Kazimierz KRZYWICKI, Grzegorz ANDRZEJEWSKI
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI,
ul. prof. Z. Szafrana 2, 65-516 Zielona Góra

Interfejsy wymiany danych w systemach rozproszonych

Mgr inż. Kazimierz KRZYWICKI

Absolwent Wydziału Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. Członek Polskiego Towarzystwa Informatycznego. Obecnie jest słuchaczem studiów doktoranckich na Uniwersytecie Zielonogórskim. Zainteresowania badawcze obejmują zagadnienia z zakresu projektowania i implementacji rozproszonych systemów wbudowanych, automatyki i robotyki oraz syntezy układów reprogramowalnych FPGA.

e-mail: k.krzywicki@weit.uz.zgora.pl

Dr inż. Grzegorz ANDRZEJEWSKI

Dr inż. Grzegorz Andrzejewski urodził się w roku 1970. Studia ukończył w roku 1995 na Politechnice Poznańskiej. Stopień doktora z zakresu informatyki uzyskał na Politechnice Szczecińskiej w roku 2002. Jego zainteresowania naukowe ukierunkowane są na zagadnienia modelowania i syntezy systemów sterowania cyfrowego.

e-mail: g.andrzejewski@iie.uz.zgora.pl

Streszczenie

W artykule przedstawiono analizę wydajności wybranych metod wymiany danych i protokołów komunikacji w prostych systemach rozproszonych. Dokonano porównania niezbędnej ilość przesyłanych danych przy użyciu najbardziej popularnych protokołów transmisji oraz protokołu zaproponowanego przez autorów. Metody przeanalizowano pod kątem wydajności w zależności od rodzaju sterowania i ilości modułów wykonawczych rozproszonego systemu sterowania. Zmniejszenie obciążenia infrastruktury systemu wbudowanego pozwala na skrócenie czasu reakcji układu, a modularna budowa na szybką rekonfigurację i zmniejszenie kosztów całego systemu sterowania.

Słowa kluczowe: systemy rozproszone, wymiana danych, protokoły komunikacyjne.

Data exchange interfaces in distributed systems

Abstract

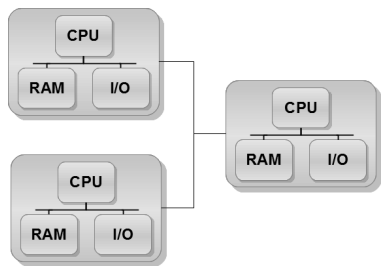
The paper deals with the performance analysis of selected methods of data exchange and communication protocols in simple distributed systems. There is presented the comparison of three main methods of distributed system implementation i.e.: central module control, hybrid module control and distributed system control. Each of them is implemented by different communication protocols. The authors made a calculation of the necessary amount of transmitted data between these communication protocols commonly used in the industry (Modbus RTU representing central module control, Profibus-DP representing hybrid module control) as well as the protocol proposed by the authors called the CloudBus representing the distributed system control. The CloudBus is a new method of process synchronization and communication between a various number of the modules in the distributed system. It allows for fast reconfiguration of the developed system platform. The performance of the methods is analysed depending on the type of control and the number of modules implementing a distributed control system. Based on the presented results one can conclude that the CloudBus protocol allows significant savings in the amount of the transmitted data. A greater number of modules results in bigger data traffic savings, especially when compared to the amount of the transmitted data by the Modbus RTU protocol.

Keywords: distributed systems, data exchange, communication protocols.

1. Wprowadzenie

Rozwój przemysłowych systemów sterowania wymusił zaprojektowanie dedykowanych protokołów komunikacji do wymiany danych i synchronizacji pracy poszczególnych jednostek obliczeniowych w systemach wbudowanych. W chwili obecnej część z nich jest jawna, co umożliwia ich analizę oraz porównanie efektywności działania. Do najpopularniejszych protokołów należy zaliczyć protokoły z rodziny: Modbus oraz Profibus.

Obecne systemy wbudowane coraz częściej stają się systemami rozproszonymi (rys. 1) z pewną ilością (n) modułów wykonawczych. Każdy z nich stanowi zarazem pojedynczą jednostkę, jak również element składowy całego systemu. Istnieje wiele standardów komunikacyjnych, możliwych do wykorzystania w systemach rozproszonych. W artykule skupiono się na najbardziej popularnych (Modbus RTU i Profibus-DP) w kontekście autorskiego protokołu – CloudBus.



Rys. 1. Ogólny schemat systemu rozproszonego
Fig. 1. General model of the distributed system

2. Metody wymiany danych

Komunikację między modułami można podzielić na trzy główne metody realizacji ze sterowaniem: centralnym, hybrydowym oraz zdecentralizowanym.

Metoda sterowania centralnego polega na wykonywaniu algorytmu sterującego przez jednostkę nadrzędną (*master*). Wymiana informacji w systemie odbywa się na zasadzie odpytania wszystkich modułów podrzędnych (*slave*) o stan obsługiwanych przez nie sygnałów wejściowych i przekazanie do wszystkich modułów informacji o stanie sygnałów wyjściowych.

Sterowanie hybrydowe polega na przekazywaniu tokena między jednostkami nadrzędnymi (*master*). Urządzenie posiadające token zarządza magistralą danych i określonymi urządzeniami podrzędnymi (*slave*).

W sterowaniu rozproszonym (zdecentralizowanym) każdy z modułów posiada wyodrębnione zadanie sterowania, które samodzielnie realizuje. Wymiana informacji dotyczy punktów synchronizacji poszczególnych, wyodrębnionych podprocesów. W pozostałych sytuacjach komunikacja pomiędzy modułami składającymi się na cały system nie zachodzi.

Modbus RTU

Protokół Modbus jest jednym z najstarszych protokołów używanych w automatyce przemysłowej [2, 4, 6] (opublikowany w 1979r.). Z racji prostej budowy i łatwej implementacji używany jest on nadal w nieskomplikowanych systemach wbudowanych.

Ogólna budowa ramki protokołu Modbus RTU zaprezentowana została w tab. 1. Protokół ten reprezentuje metodę sterowania centralnego, z pojedynczą jednostką nadrzędną (*master*) oraz określoną liczbą urządzeń podrzędnych (*slave*).

Tab. 1. Podstawowa ramka danych dla protokołu Modbus RTU
Tab. 1. Basic data frame of the Modbus RTU protocol

ST	ADD	FUNC	DATA	CRC	ED

ST – oznacza 3 bajty początku ramki; ADD – bajt adresu docelowego; FUNC – bajt sterujący; DATA – tablica danych; CRC – 2 bajty sumy kontrolnej; ED – 3 bajty końca ramki.

Dla celów niniejszego porównania wykorzystano najprostszy format ramki danych, aby uzyskać możliwie maksymalną wydajność i najmniejszą ilość danych niezbędną do przesłania zapewniającą poprawne funkcjonowanie systemu.

Profibus-DP

Protokół Profibus-DP [1, 5] opiera się na wykorzystaniu dwóch trybów pracy urządzeń tj.: *master* (kontrola magistrali danych) i *slave* (tryb pasywny). W przypadku wykorzystania większej ilości urządzeń *master* niezbędne jest przekazywanie pomiędzy nimi tokena, który określa zdolność danego urządzenia *master* do kontroli magistrali.

Tab. 2. Podstawowa ramka danych, żądania dla protokołu Profibus-DP
Tab. 2. Basic request data frame of the Profibus-DP protocol

SD1	DA	SA	FC	FCS	ED

Podstawowa ramka żądania bez tablicy danych przedstawiona została w tab. 2, gdzie: SD1 – oznacza bajt początku ramki; DA – bajt adresu docelowego; SA – bajt adresu źródłowego; FC – bajt sterujący; FCS – bajt kontrolny; ED – bajt końca ramki.

Tab. 3. Podstawowa ramka danych, odpowiedzi dla protokołu Profibus-DP
Tab. 3. Basic response data frame of the Profibus-DP protocol

SD3	DA	SA	FC	DATA	FCS	ED

Rozbudowana ramka odpowiedzi o stałej długości tablicy danych przedstawiona została w tab. 3, gdzie: SD3 – oznacza bajt początku ramki; DA – bajt adresu docelowego; SA – bajt adresu źródłowego; FC – bajt sterujący; DATA – tablica danych; FCS – bajt kontrolny; ED – bajt końca ramki.

Tab. 4. Podstawowa ramka danych, tokena dla protokołu Profibus-DP
Tab. 4. Basic token data frame of the Profibus-DP protocol

SD4	DA	SA

Ramka przekazania tokena przedstawiona została w tab. 4, gdzie: SD4 – oznacza bajt początku ramki; DA – bajt adresu docelowego; SA – bajt adresu źródłowego.

Dodatkowo w przypadku protokołu Profibus-DP należy uwzględnić minimum 33 bity na okres synchronizacji między żądaniami.

Protokół ten reprezentuje metodę sterowania hybrydowego.

Protokół autorski CloudBus

Zaproponowany przez autorów protokół [3] *CloudBus* realizuje metodę sterowania zdecentralizowanego (rozproszonego), gdzie wszystkie urządzenia (moduły) funkcjonują na równych prawach. Metoda bazuje na tym, że jeżeli moduł potrzebuje informacji spoza własnego zasobu zmiennych, to dopiero wtedy wysyła (rozgłasza) do systemu konkretne zapytanie o stan określonej zmiennej. Moduł, który jest odpowiedzialny za zarządzanie tą zmienną odpowiada do systemu dopiero wtedy, gdy zmienna przyjmie wartość, o którą dotyczyło zapytanie (tab. 5).

Tab. 5. Podstawowa ramka danych dla protokołu CloudBus
Tab. 5. Basic data frame of the CloudBus protocol

CNT	FUNC	VARS	DATA	CRC

Pole CNT – oznacza 1 bajt długości ramki; FUNC – kod instrukcji (np. zapytania o warunki, bądź odpowiedź); VARS i DATA reprezentują bitową tablicę zmiennych i ich wartości; CRC – 1 bajt sumy kontrolnej.

3. Analiza wydajności

Analizę i obliczenie wydajności poszczególnych metod wymiany danych zrealizowano przyjmując:

- 8 modułów wykonawczych: Modbus RTU (8 modułów wykonawczych *slave* i 1 moduł nadzoru *master*; Profibus-DP (8 modułów master); CloudBus (8 modułów równorzędnych)
- 16 wejść/wyjść cyfrowych w module,
- 4 wejść/wyjść analogowych w module po 16 bitów,
- zasoby współdzielone między modułami stanowią nie więcej niż 10% wszystkich zmiennych w systemie w związku z tym przyjęto, że niezbędne jest przesłanie co najmniej 64 bajtów danych (13 wej/wyj cyfrowych oraz 2 wej/wyj analogowe),
- dla metod opartych o: Profibus-DP i CloudBus, przyjęto skrajnie pesymistyczny wariant sprawdzania przez każdy moduł – pozostałych modułów i ustawiania stanów ich wyjść,
- dodatkowo dla protokołu Profibus-DP pominięto transfer danych niezbędny do przekazywania tokena między modułami *master*, ewentualną wymianę danych z modułami *slave* oraz czas synchronizacji magistrali między kolejnymi żądaniami,
- ramka danych (pole *DATA*) zawierająca stany wejść/wyjść cyfrowych oraz analogowych przedstawiona została w tab. 6. Pole IN/OUT reprezentuje stany wejść/wyjść cyfrowych – maksymalnie 2 bajty (16 bitów, po 1 bit na każde wejście/wyjście cyfrowe), ADC*n*(H) oraz ADC*n*(L) reprezentują starszą oraz młodszą cześć z danego wejścia/wyjścia analogowego – maksymalnie 8 bajtów (po 2 bajty na każde wejście/wyjście analogowe)

Tab. 6. Ramka danych zawierająca stan wejść/wyjść cyfrowy i analogowych
Tab. 6. Basic data frame of the digital and analog inputs/outputs

IN/OUT	…	ADC1(H)	ADC1(L)	…

Powyższe założenia, szczególnie dotyczące uproszczonej budowy ramek danych, pominięcia czasów synchronizacji magistrali i czasów przetwarzania jednostek *master*, mają na celu eliminację wszystkich dodatkowych czynników, które mogłyby zbędnie zwiększyć ilość przesłanych danych oraz czasu odpowiedzi układu, a tym samym spowodować nieprawidłową interpretację uzyskanych rezultatów.

W tab. 7 przedstawiono zestawienie ilości bajtów niezbędnych do transmisji.

Schemat przesłania danych dla poszczególnych protokołów prezentuje się następująco:

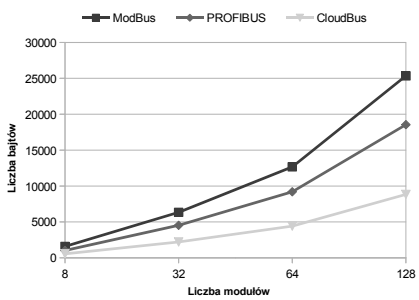
- Modbus RTU:
  Moduł *master* musi odpytać (*Pytanie* – po 10 bajtów dla każdego z 8 modułów wykonawczych) wszystkie moduły *slave* o stan ich wejść, następnie moduły *slave* kolejno odpowiadają (*Odpowiedź* – po 94 bajty każdy z modułów). Po dokonaniu przetwarzania moduł *master* przekazuje kolejno do modułów *slave* informacje o ustawieniu odpowiednio ich wyjść (*Ust. wyjść* – po 94 bajty dla każdego z 8 modułów). W przypadku protokołu Modbus przyjęto, że wszystkie dane (stany wejść/wyjść cyfrowych i analogowych) znajdują się w jednym rejestrze, dzięki czemu odczyt/zapis następuje w jednym cyklu transmisji. Dokonywanie ww. transferu danych przy użyciu kolejnych komend pobierania stanu pojedynczych wejść/wyjść spowodowałoby sztuczne – kilkukrotne zawyżenie ilości przesyłanych danych,
- Profibus-DP:
  Moduł *master* (posiadający token w danej jednostce czasu) odpytuje (*Pytanie* – po 6 bajtów na moduł) kolejno pozostałe moduły (7 modułów) zawierające wejścia/wyjścia systemu. Następnie, moduły przekazują *Odpowiedź* (po 70 bajtów), po czym następuje ustawienie wyjść (*Ust. wyjść*, również po 70 bajtów).
- CloudBus:
  Jeden z modułów, rozgłasza do systemu informację, że potrzebuje informację o stanie danego wejścia/wyjścia spoza własne-

go zbioru (*Pytanie* – 69 bajtów), następnie pozostałe moduły (7 modułów) odpowiadają (*Odpowiedź* – każdy po 69 bajtów). Ze względu na budowę i charakterystykę systemu [3] nie występuje tutaj proces ustawiania wyjść (*Ust. wyjść*), ponieważ, każdy z modułów realizuje wybrany fragment sterowania, za który jest odpowiedzialny, a komunikacja jest ograniczona i realizowana wtedy i tylko wtedy, gdy jest to konieczne dla wybranych punktów synchronizacji i wymiany danych współdzielonych w systemie.

Tab. 7. Porównanie ilości przesyłanych danych dla protokołów komunikacyjnych
Tab. 7. Traffic amount comparison for different communication protocols

Protokół	Długość ramki [bajty]			
	Pytanie	Odpowiedź	Ust. wyjść	Suma
Modbus RTU	80	752	752	1584
Profibus-DP	42	490	490	1022
CloudBus	69	483	0	552

Poniższy wykres (rys. 2) przedstawia zależność między liczbą modułów a ilością bajtów niezbędnych do transmisji. Ilość przesłanych danych rośnie wraz ze wzrostem ilości modułów. Wzrost ten nie jest proporcjonalny, tzn. protokół CloudBus, będzie zwiększał swoją przewagę nad pozostałymi protokołami, wraz ze wzrostem ilości modułów w systemie.



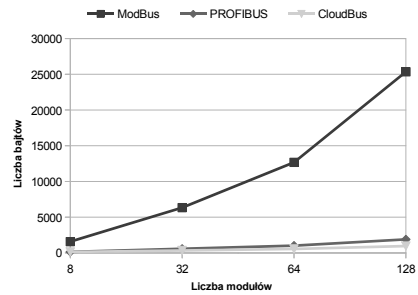
Rys. 2. Ilość przesyłanych danych w zależności od ilości modułów
Fig. 2. Traffic amount vs. the number of modules

W przypadku protokołów Profibus-DP i CloudBus przyjęto skrajnie niekorzystny wariant komunikacji ze wszystkimi modułami. W warunkach rzeczywistego rozproszonego systemu sterowania sytuacja taka jest bardzo mało prawdopodobna, gdyż oznaczałaby poważne błędy już na poziomie fazy projektu systemu. Praktyka wskazuje na przyjęcie progu ilości zmiennych współdzielonych między modułami na poziomie 10% oraz 10% progu ilości wszystkich warunków synchronizacji w systemie. Uwzględniając powyższe, w tab. 8 zaprezentowano wyniki ilości przesyłanych danych przy nałożeniu powyższych ograniczeń.

Tab. 8. Porównanie ilości przesyłanych danych dla protokołów komunikacyjnych w przypadku narzucenia ograniczeń w zależności od ilości modułów
Tab. 8. Traffic amount comparison for different communication protocols under restrictions in the term of the number of modules

Liczba modułów	Protokół / liczba przesłanych bajtów		
	Modbus RTU	Profibus-DP	CloudBus
8	1584	146	138
32	6336	584	345
64	12672	1022	552
128	25344	1898	966

Analizując otrzymane wyniki widać znaczne zmniejszenie ilości przesyłanych danych. Zysk w ilości przesłanych bajtów przy zastosowaniu powyższych ograniczeń pomiędzy protokołem CloudBus, a Modbus RTU wynosi ponad 25 razy na korzyść protokołu CloudBus. Porównując uzyskane wyniki (tab. 7 i tab. 8), narzucenie ograniczeń ilości zmiennych współdzielonych i punktów synchronizacji dla protokołów Profibus-DP i CloudBus spowodowało w przypadku dla 8 modułów, zmniejszenie kilkukrotnie ilości przesyłanych danych. Zysk wzrasta (szczególnie dla protokołu CloudBus) wraz ze wzrostem liczby modułów w systemie. Dla protokołu Modbus RTU brak jakiegokolwiek zysku wynika z zastosowanej metody wymiany danych (ze sterowaniem centralnym), gdzie niezbędne jest odpytanie przez moduł *master* wszystkich urządzeń *slave*. Na rys. 3 przedstawiono wykres zależności między ilością modułów a ilością przesłanych danych przy założeniu ograniczeń na podstawie tab. 8.



Rys. 3. Ilość przesyłanych danych w zależności od ilości modułów w przypadku narzucenia ograniczeń
Fig. 3. Traffic amount vs. the number of modules under restrictions

4. Wnioski

W artykule przedstawiono porównanie wybranych, popularnych metod sterowania i protokołów komunikacji z protokołem zaproponowanym przez autorów. Protokoły Modbus RTU i Profibus-DP pomimo ich rozbudowanej funkcjonalności, ograniczono strukturalnie do niezbędnego minimum. Miało to na celu eliminację wszelkich dodatkowych czynników, które mogłyby mieć wpływ na uzyskane wyniki, tj. dodatkową ilość przesyłanych danych pomiędzy modułami. Dodatkowo, do obliczeń, zarówno w przypadku protokołu CloudBus, jak i Profibus-DP przyjęto maksymalnie pesymistyczny wariant wymiany danych między wszystkimi modułami. Miało to jednak na celu wykazanie różnic między poszczególnymi metodami wymiany danych. Analiza wykazała kilkukrotne zmniejszenie ilości przesyłanych danych na rzecz protokołu CloudBus względem pozostałych. Przy założeniu progu ilości zmiennych współdzielonych między modułami na 10% oraz ilości warunków synchronizacji na 10% wszystkich warunków synchronizacji w systemie możliwym wydaje się uzyskanie nawet kilkudziesięciokrotnej oszczędność w ilości przesyłanych danych w porównaniu do metody ze sterowaniem centralnym (Modbus RTU). Oszczędność ta, będzie wzrastać wraz ze wzrostem ilości modułów pracujących w systemie rozproszonym. Wartości progów ograniczających mogą zostać dowolnie dobrane poprzez odpowiedni dobór ilości modułów i podział realizowanych zadań.

5. Literatura

[1] Solnik W, Zajda Z.: Sieci przemysłowe Profibus DP i MPI w automatyce, Oficyna Wydawnicza Politechniki Wrocławskiej, 2010.
[2] Chipkin P.: Modbus For Field Technicians, Chipkin Automation Systems, 2011.
[3] Krzywicki K., Andrzejewski G.: Concurrent process synchronization in distributed systems, XV International PHD Workshop – OWD 2013.
[4] Modbus Application Protocol Specification v1.1b, Modbus Organization
[5] IEC 61158, International Electrotechnical Commission. Digital Data Communications for Measurement and Control, 1999.
[6] Jaszczak S., Małecki K., Sokołowski R.: Wykorzystanie OPC i ModBus w zarządzaniu siecią sterowników PLC, Pomiary Automatyka Kontrola 2012, Vol. 58, nr 7, 602-604.

otrzymano / received: 12.04.2014
przyjęto do druku / accepted: 02.06.2014 *artykuł recenzowany / revised paper*