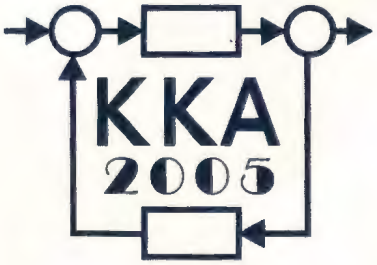
XV Krajowa Konferencja Automatyki

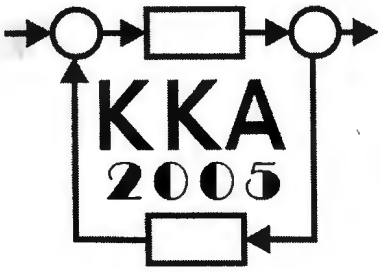
Tom II



Redaktorzy:
Zdzisław Bubnicki
Roman Kulikowski
Janusz Kacprzyk

XV Krajowa Konferencja Automatyki

Tom II



Redaktorzy:
Zdzisław BUBNICKI
Roman KULIKOWSKI
Janusz KACPRZYK

ORGANIZATOR
Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk
WSPÓŁORGANIZATORZY
Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

ORGANIZATOR

Komitet Automatyki i Robotyki Polskiej Akademii Nauk
Instytut Badań Systemowych Polskiej Akademii Nauk

WSPÓŁORGANIZATORZY

Politechnika Warszawska
Przemysłowy Instytut Automatyki i Pomiarów
Polskie Stowarzyszenie Pomiarów, Automatyki i Robotyki

KOMITET PROGRAMOWY

Przewodniczący	Zdzisław BUBNICKI
Zastępca Przewodniczącego	Roman KULIKOWSKI

CZŁONKOWIE

Stanisław BAŃKA	Michał BIAŁKO
Mikołaj BUSŁOWICZ	Władysław FINDEISEN
Ryszard GESSING	Henryk GÓRECKI
Jakub GUTENBAUM	Jerzy JÓZEFCZYK
Stanisław KACZANOWSKI	Tadeusz KACZOREK
Janusz KACPRZYK	Jerzy KLAMKA
Józef KORBICZ	Zbigniew KOWALSKI
Krzysztof KOZŁOWSKI	Juliusz L. KULIKOWSKI
Krzysztof KUŹMIŃSKI	Kazimierz MALANOWSKI
Krzysztof MALINOWSKI	Wojciech MITKOWSKI
Antoni NIEDERLIŃSKI	Władysław PEŁCZEWSKI
Tadeusz PUCHAŁKA	Leszek RUTKOWSKI
Stanisław SKOCZOWSKI	Roman SŁOWIŃSKI
Jerzy ŚWIĄTEK	Andrzej ŚWIERNIAK
Ryszard TADEUSIEWICZ	Piotr TATJEWSKI
Krzysztof TCHOŃ	Leszek TRYBUS
Jan WĘGLARZ	Andrzej P. WIERZBICKI

KOMITET ORGANIZACYJNY

Przewodniczący	Roman KULIKOWSKI
Zastępcy Przewodniczącego	Janusz KACPRZYK
	Stanisław KACZANOWSKI
	Tadeusz KACZOREK
	Krzysztof MALINOWSKI
Członkowie	Roman OSTROWSKI
	Tadeusz PUCHAŁKA
	Dariusz WAGNER
Sekretarze naukowi	Jan STUDZIŃSKI
	Jan W. OWSIŃSKI

ISBN 83-89475-01-4

Druk: ARGRAF, Warszawa

ROBOTY

MODERNIZACJA STEROWNIKA ROBOTA IRb–6

Andrzej ENGLOT*, Aleksander MAZGAJ**

* Politechnika Krakowska, Wydział Inżynierii Elektrycznej i Komputerowej
ul. Warszawska 24, 31-155 Kraków, e-mail: englot@pk.edu.pl

** Politechnika Krakowska, Wydział Inżynierii Elektrycznej i Komputerowej
ul. Warszawska 24, 31-155 Kraków, e-mail: amazgaj@pk.edu.pl

Streszczenie: W artykule omówiona została modernizacja układu sterowania popularnego w Polsce robota IRb-6. Dokonano tego poprzez zaprojektowanie a następnie zbudowanie płyty sterownika w oparciu o układ FPGA firmy Xilinx oraz mikrokontroler Atmel. Dzięki temu dotychczasowe możliwości robota zostały rozszerzone o sterowanie w czasie rzeczywistym z komputera PC przez port szeregowy. Przykładowy program sterujący napisany został w oparciu o system czasu rzeczywistego RTLinux.

Słowa kluczowe: robotyka, irb-6, układy programowalne FPGA, systemy czasu rzeczywistego.

1. WSTĘP

Robot przemysłowy IRb-6 opracowany został w latach siedemdziesiątych przez firmę ASEA. W latach osiemdziesiątych produkowany był również w Polsce. W roku 1991 zastąpiony został przez model nowocześniejszy IRp-6. Podstawową zaletą robota IRp-6 w porównaniu z IRb-6 jest nowoczesny sterownik umożliwiający między innymi łatwiejsze i bardziej elastyczne programowanie. Część mechaniczna i napędowa robota, a zatem i jego dynamika, pozostały bez zmian. Z względu na to celowa wydawała się taka modernizacja sterownika robota IRb-6, która umożliwiałaby sterowanie robotem przez zewnętrzny komputer PC. Pierwszej modernizacji w Instytucie Automatyki Politechniki Krakowskiej dokonano w latach 2001 – 2002 [3]. Zbudowany został sterownik robota, który przez łącze RS-232 odbierał wyznaczone przez komputer PC przyrosty ruchu robota i przesyłał je do sterowników osi. Pomiar położenia osi dokonywano za pomocą karty przetworników analogowo-cyfrowych RT-DAC [5]. Obecna modernizacja obejmuje rezygnację z karty pomiarowej i uzupełnienie sterownika robota o własny układ pomiaru odchyłek położenia osi.

2. UKŁAD PODSTAWOWY STEROWNIKA ROBOTA IRB-6

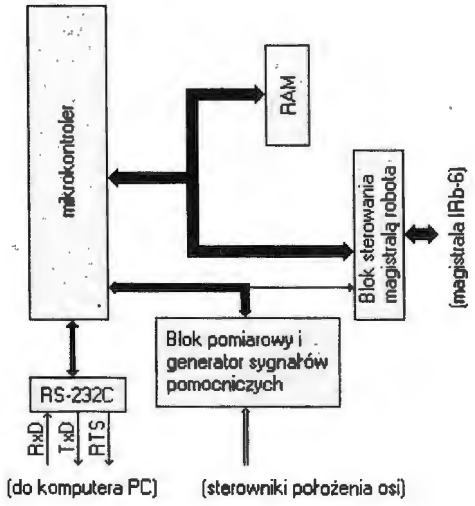
Oryginalny sterownik robota IRb-6 zbudowany został na czterech płytach [1, 2]:

- mikroprocesora,
- identyfikacji przerwań i generowania sygnałów pomocniczych,
- pamięci programu (PROM – 8 kB),
- pamięci danych i programów użytkownika (RAM – 8 kB).

Elementy elektroniczne tych płyt współpracują ze sobą poprzez magistralę robota, do której dołączone są również układy zewnętrzne takie jak: sterowniki położenia osi oraz blok wejść i wyjść cyfrowych. Sterownik robota zbudowany został z podzespołów dostępnych w latach siedemdziesiątych: mikroprocesora Intel 8008 z zegarem 0,73 MHz, elementów pamięci RAM i EPROM o pojemnościach 256 bajtów.

3. WPROWADZONE MODYFIKACJE

Zmodernizowany układ (rys. 1) zbudowany jest na jednej płycie. Wykorzystuje on 8 bitowy mikrokontroler Atmel taktowany zegarem o częstotliwości 22,118 MHz., układ programowalny Xilinx oraz 8 kB pamięci RAM przeznaczonej na programy użytkownika. Pamięć programu – systemu operacyjnego (8kB Flash) – zawarta jest w mikrokontrolerze.
Mikrokontroler steruje pracą osi i innych podzespołów robota według programu zawartego w pamięci, a także umożliwia dwustronną komunikację po łączu RS-232 z zewnętrznym komputerem PC (rys.2). W układzie Xilinx zaprogramowano 5 szesnastobitowych liczników, których zadaniem jest pomiar szerokości prostokątnych sygnałów z rezolwerowego układu pomiaru odchyłki położenia osi robota. Logika znajdująca się w tym

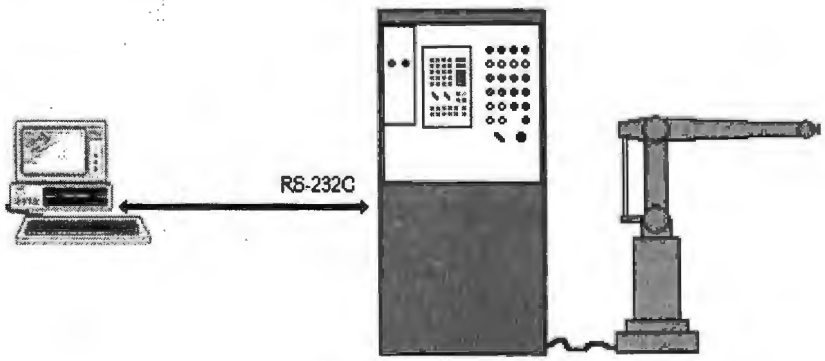


Rys.1. Zmodyfikowany układ sterownika robota.

układzie generuje również wzajemnie uzależnione czasowo sygnały potrzebne do pracy robota.

4. UZYSKANE MOŻLIWOŚCI APLIKACYJNE

Komunikacja komputer PC – sterownik robota umożliwia zastosowanie bardziej złożonych algorytmów



Rys.2. Układ sterowania robotem z wykorzystaniem komputera.

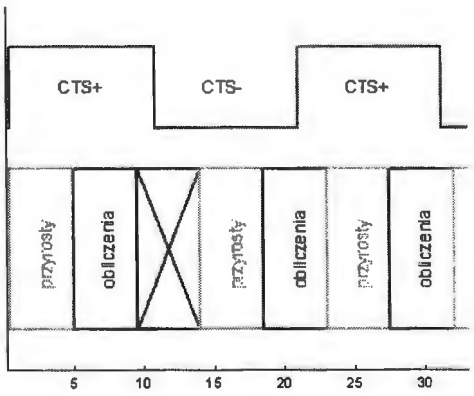
sterowania, w tym dostępnych w oprogramowaniu MATLAB.

Podczas pracy systemu układ Xilinx wyznacza odchyłki położenia osi robota od wartości zadanej na podstawie pomiaru wypełnienia sygnałów prostokątnych z układów rezolwerowych sterowników położenia. Częstotliwość tych sygnałów wynosi 125 Hz. Mikrokontroler odczytuje zawartość liczników z układu Xilinx, które zawierają wartości powyższych odchyłek i przesyła te dane po łączu RS do komputera PC. Na podstawie tych informacji wyznaczane są w komputerze kolejne przyrosty położenia osi robota, które następnie przesyłane są po łączu RS do mikrokontrolera. Mikrokontroler przekazuje te dane przez wewnętrzną magistralę robota do sterowników położenia osi. Cykl pracy robota IRb-6 wynosi 10,2 ms. W celu synchronizacji pracy mikrokontrolera i komputera PC, na początku każdego cyklu mikrokontroler zmienia stan linii RTS na przeciwny. Sygnał ten łączem RS-232C przesyłany jest do komputera PC i odbierany na linii CTS komputera.

Możliwe są dwa sposoby synchronizacji cyklu robota i mikrokomputera.

Pierwszy polega na tym, że czas cyklu systemu operacyjnego komputera wynosi 5 ms. Na początku swojego cyklu komputer sprawdza, czy nastąpiła zmiana sygnału CTS. Jeżeli tak, to przeprowadzane jest wysłanie wcześniej wyliczonych przyrostów ruchu robota do portu szeregowego, odbiór zmierzonej odchyłki położenia robota i wyliczenie położenia. Jeżeli w danym takcie nie ma zmiany sygnału CTS a była w takcie poprzednim, wyznaczane są nowe przyrosty ruchu robota według zaprojektowanego algorytmu. Jeżeli w danym takcie komputera nie ma zmiany CTS i nie było jej również w takcie poprzednim, komputer nie wykonuje żadnej czynności związanej ze sterowaniem. Przy tak przyjętym czasie próbkowania komputera, żaden cykl pracy robota (zmiana sygnału CTS) nie zostanie opuszczony. Omówiony sposób współpracy sterownika robota z komputerem PC przedstawiony jest na rys. 3 i zalecany przy oprogramowaniu Real-Time Windows Target.

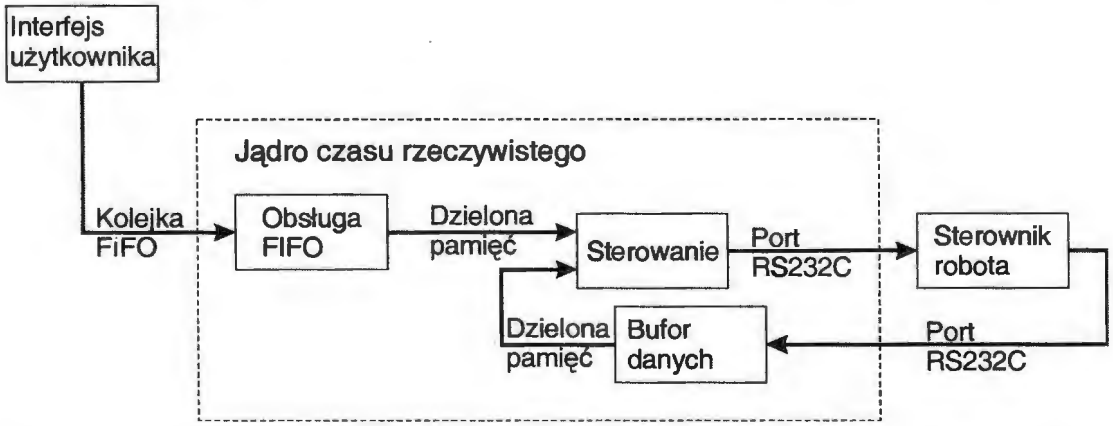
Drugi sposób, możliwy do zastosowania w systemie RTLinux, polega na uruchamianiu odpowiednich zadań podczas obsługi przerwania spowodowanego zmianą sygnału CTS.



Rys. 3. Synchronizacja komputera PC ze sterownikiem robota.

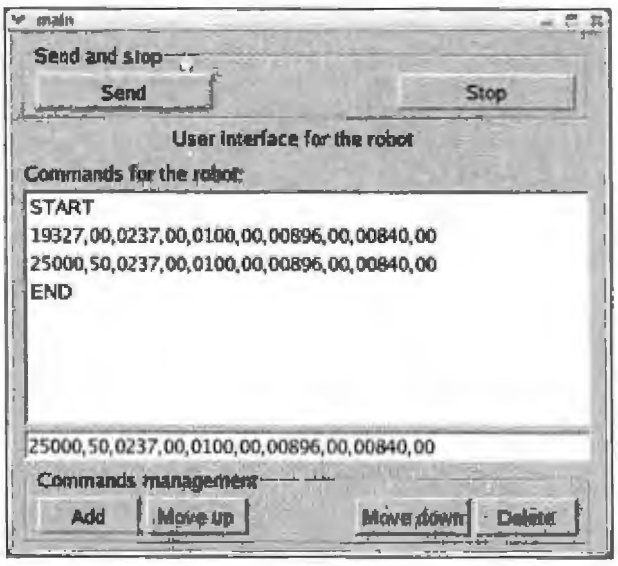
5. PRZYKŁADOWE STEROWANIE Z ZASTOSOWANIEM SYSTEMU CZASU RZECZYWISTEGO RTLINUX

Wstępnie działanie układu przetestowano z zastosowaniem twardego systemu czasu rzeczywistego RTLinuxFree 3.1, będącego nakładką na jądro Linux'a 2.4 [4]. W pracy modyfikacji poddano jądro 2.4.20. Na rys. 4 przedstawiono strukturę ładowalnego modułu, będącego zadaniem czasu rzeczywistego, wraz z łączami zapewniającymi komunikację z konsolą użytkownika umożliwiającą zadawanie trajektorii, jaką robot ma wykonać, oraz ze sterownikiem robota.



Rys. 4. Schemat blokowy programu w komputerze PC.

Procedura wykonywania zadanej trajektorii przez robota rozpoczyna się od ustalenia tejże trajektorii przez użytkownika a następnie przesłania jej do zadania czasu rzeczywistego. Dokonane to może zostać przez przykładowy interfejs przedstawiony na rys. 5, do którego utworzenia zastosowano środowisko programistyczne Qt. W celu przesłania tych danych z przestrzeni użytkownika do zadania czasu rzeczywistego zastosować można jeden z dostępnych w RTLinux'ie mechanizmów komunikacji międzyprocesowej. W prezentowanym projekcie wykorzystano kolejką czasu rzeczywistego FIFO.



Rys. 5. Interfejs użytkownika.

Procedura asynchroniczna obsługująca tę kolejkę po stronie zadania czasu rzeczywistego, określona na rys. 4 jako *Obsługa FIFO*, po odebraniu całego zbioru danych, przepisuje je do bufora znajdującego się w zadaniu cyklicznym – *Sterowanie*. Przepisanie tych danych zabezpieczone jest obiektem typu *mutex* uniemożliwiającym dostęp do danych przez dwa lub więcej procesów równocześnie. W tym przypadku dokonywane jest to w celu zabezpieczenia regulatora przed wyznaczaniem sterowania dla dwóch różnych czasowo zbiorów danych.

Kolejna procedura asynchroniczna stosowana jest do odbioru danych przychodzących poprzez port RS232C ze zmodyfikowanej karty sterownika robota. Dane te, zawierające aktualną odchyłkę położeń osi robota od wartości sygnału zadanego, zapisywane są następnie w buforze znajdującym się we wspomnianym wcześniej zadaniu cyklicznym. Również w tej procedurzej zastosowano mechanizm *mutex* służący temu samemu celowi, co poprzednio.

Zarówno zadana trajektoria, jaki i otrzymane ze sterownika robota dane o aktualnym położeniu służą do wyznaczenia sterowania ale również do nadzoru i zatrzymania pracy robota (wstrzymania wysyłania przyrostów) w przypadkach krytycznych tj. gdy odchyłki położenia robota są znaczne a sterowanie nie ma możliwości zniwelowania tej różnicy. Dokonywane jest to w wątku cyklicznym, wykonywanym, co 5 ms, i dopiero po przyjściu informacji o oczekiwaniu robota na nową wartość sterowania – sygnał CTS. Wyznaczone sterowanie wysyłane jest następnie poprzez port RS232C do sterownika robota.

6. WNIOSKI KOŃCOWE

Poprzez przyjęcie przedstawionej struktury sterowania robotem tzn. poprzez zmodyfikowanie układu sterownika robota oraz zastosowanie zaprezentowanego systemu czasu rzeczywistego, staje się możliwa implementacja różnego rodzaju algorytmów służących wyznaczeniu wartości sygnału sterującego, przeprowadzającego kiść robota z położenia początkowego do zadanego położenia docelowego po przyjętej trajektorii.

Dodatkowo, korzystając z otwartości systemu Linux można rozbudowywać niskim kosztem zaprezentowany układ o kolejne elementy np. o sterowanie układem z nadzorem w postaci obrazu z kamery. Natomiast za-

stosowanie bibliotek środowiska inżynierskiego MATLAB – wersja dla systemu Unix – prowadzi do możliwości wykorzystania bardziej rozbudowanych rodzin regulatorów.

Początkowe testy układu wykazały pełną poprawność działania układu. Zarówno sterownik robota zbudowany w oparciu o układ FPGA i mikrokontroler jak i program znajdujący się w nim a również na komputerze PC działały bezbłędnie.

Przedstawiony sterownik jest układem stosunkowo prostym. Zastępuje on oryginalny sterownik robota, pozostawiając pozostałe układy sterowania, czyli sterowniki położenia osi i układy regulacji mocy bez zmian. Praca tego układu podlega jednak pewnym ograniczeniom związanym z wyznaczaniem analogowych sygnałów zadanych dla układów regulacji mocy przez sterownik położenia osi. Po pierwsze wysłany z komputera sygnał przyrostów robota zmienia sygnał zadany układu regulacji mocy po 2 taktach robota. Drugim istotniejszym ograniczeniem jest maksymalna ilość przyrostów wysyłanych przez komputer w jednym takcie robota. Wynosi ona 80 przyrostów. Ponieważ sygnał zadany układu regulacji mocy zależy od różnicy między ilością przyrostów wysłanych przez system a ilością przyrostów wykonanych przez silnik, pełną moc (pełną szybkość ruchu) uzyskuje się po pewnej liczbie taktów robota. Z rozważań, które nie są tu zamieszczone, liczba taktów robota, po której silnik osiąga pełną moc wynosi 80 + 100, czyli po czasie około 1 sek.

MODERNISATION OF THE IRb-6 ROBOT CONTROLLER

This paper describes a modernisation of the robot IRb-6 control circuit. It has been done through the design and after that building the enhanced controller on the basis of Xilinx FPGA circuit and Atmel microcontroller. Due to it capabilities of the robot have been expanded to the direct real-time control through serial port using a computer.

Literatura

[1] *Dokumentacja serwisowa robota IRb-6*. Przemysłowy Instytut Automatyki i Pomiarów MERA-PIAP, 1987.

[2] *Dokumentacja techniczno-ruchowa robotów IRb-6 i IRb-60*. Przemysłowy Instytut Automatyki i Pomiarów MERA-PIAP, 1987.

[3] Englot A. (2002) Zmodernizowany sterownik robota IRb-6. *Pomiary Automatyka Robotyka*, 11, 26-28.

[4] Lal K., Rak T., Orkisz K. (2003) *RTLinux system czasu rzeczywistego*. Helion, Gliwice

[5] *RT-DAC multi I/O board. User's manual*. INtelligent TEchnology for COntrol LTD, 1998.

Instytut Badań Systemowych
Polskiej Akademii Nauk

ISBN 83-89475-01-4