Leszek TRYBUS - kierownik projektu
Ryszard LENIOWSKI
Wacław IRZEŃSKI

Politechnika Rzeszowska
Zakład Automatyki i Informatyki

WIELOPROCESOROWY 32-BITOWY UKŁAD STEROWANIA ROBOTA LABORATORYJNEGO

Grant nr 7 1202 9101

W pracy zaprezentowano 32-bitowy, transputerowy układ sterowania robota laboratoryjnego, pracujący z okresem próbkowania pojedynczych miliseckund. Pozwala on realizować zaawansowane metody sterowania manipulatorem robota przemysłowego. Chodzi tutaj przede wszystkim o sterowanie z wykorzystaniem wyliczanego na bieżąco (*on-line*) modelu dynamiki manipulatora oraz o sterowanie adaptacyjne. Omówiono problem współpracy systemów transputerowych z urządzeniami zewnętrznymi. W przyjętym do realizacji rozwiązaniu, podsystem WE/WY komunikuje się z siecią transputerów za pomocą adapterów linku typu IMS C011.

1. WPROWADZENIE

Z dotychczasowych doświadczeń autorów zdobytych podczas budowy robota laboratoryjnego KRĘPY [2,3,4,12] wynika, że komputer realizujący złożone algorytmy sterowania musi posiadać znaczną moc obliczeniową. Z tego względu proponowanym bardzo często rozwiązaniem są układy wieloprocesorowe. Przykładem jest regulator, w którym do sterowania dwoma ramionami robota Adept One wykorzystano dwa współpracujące ze sobą mikroprocesory typu Motorola 68000 [5]. Jeden realizuje właściwy algorytm sterowania wykorzystując elementy modelu manipulatora. Drugi na bazie

pomiaru kątów przegubowych i sterowań dokonuje ciągłej identyfikacji parametrów dynamicznych ramion. Tak aktualizowane parametry są sukcesywnie dostarczane do pierwszego mikroprocesora, aktualizując stosowany w algorytmie sterowania model. Oba mikroprocesory działają więc równolegle, wymieniając co jakiś czas potrzebne informacje.

W przypadku typowych układów mikroprocesorowych zapewnienie wzajemnej współpracy wymaga stosowania dodatkowych rozwiązań układowych. Przykładowo, wykorzystując do komunikacji wspólny obszar pamięci należy zbudować odpowiedni układ arbitrażowy eliminujący ewentualne konflikty w sytuacji równoczesnego żądania dostępu do tej pamięci przez więcej niż jeden mikroprocesor. Może stwarzać to pewne komplikacje na etapie projektowania.

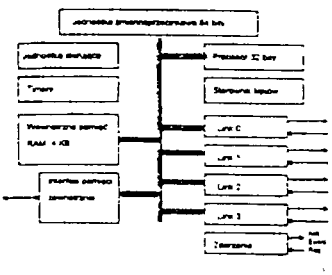
Alternatywnym rozwiązaniem jest zastosowanie do budowy regulatora cyfrowego transputerów. Transputer jest procesorem ze zredukowaną liczbą instrukcji (*RISC*) posiadającym pamięć lokalną oraz wyposażonym w linki umożliwiające jego bezpośrednie łączenie z innymi transputerami [6]. Wobec tego może on być stosowany zarówno w systemach jednoprocesorowych, jak i w sieciach z wieloma realizowanymi równolegle procesami. Procesy te są wykonywane na tym samym transputerze, względnie na kilku różnych.

Do programowania transputerów stosuje się specjalizowany język Occam [6,7], jak również powszechnie używane języki programowania wysokiego poziomu (C, Pascal, Fortran) uzupełnione o biblioteki procedur i funkcji do realizacji obliczeń równoległych i komunikacji poprzez kanały [8]. Z punktu widzenia programisty nie jest przy tym ważne czy program będzie realizowany na pojedynczym transputerze, czy na sieci. W programie występują bowiem jedynie komunikujące się poprzez kanały procesy, a informację na temat ich przyporządkowania do poszczególnych transputerów zawiera osobny, łatwo modyfikowalny plik. Dzięki temu adaptacja programu, np. po dodaniu nowych transputerów do sieci jest bardzo łatwa.

Biorąc to pod uwagę zdecydowano, że do budowy cyfrowego regulatora robota zostanie zastosowany układ transputerowy. Wykorzystano do tego celu kartę Fast9 produkowaną przez brytyjską firmę Quintek Ltd. Jest to typowa karta rozszerzenia przeznaczona do współpracy z mikrokomputerami klasy PC.

2. WIELOPROCESOROWY UKŁAD STEROWANIA ROBOTA LABORATORYJNEGO

Transputer IMS T800 jest 32 bitowym układem cyfrowym VLSI wykonanym w technologii CMOS [6]. Jego architekturę wewnętrzną przedstawiono na rys.1. 32-bitowy procesor zawiera dekoder instrukcji, jednostkę arytmetyczno-logiczną, wskaźnik instrukcji, wskaźnik roboczy oraz rejestr operandu. Może on adresować bezpośrednio 4 KB wewnętrznej pamięci RAM i do 4 GB pamięci zewnętrznej. Podczas kontaktu z pamięcią zewnętrzną wykorzystywany jest układ interfejsu. Oprócz wymienionych już trzech rejestrów (wskaźników: instrukcji i roboczego oraz rejestru operandu) procesor



transputera IMS T800 zawiera jeszcze trzy dodatkowe rejestry oznaczone jako A, B i C. Rejestry te tworzą strukturę stosu (z rejestrem A na szczycie) i są wykorzystywane podczas realizacji rozkazów. Przykładowo dla rozkazu dodawania *add* operandami są wartości przechowywane na szczycie stosu (w rejestrach A, B), a wynik jest przesyłany z powrotem na szczyt stosu (do rejestru A).

Rys.1. Architektura wewnętrzna transputera IMS T800

Pojedynczy transputer może współbieżnie realizować kilka procesów o wysokim lub niskim priorytecie. Proces o niskim priorytecie jest wykonywany do momentu zawieszenia w oczekiwaniu na zakończenie komunikacji lub upływ zadanego okresu czasu. Jeśli jednak lista procesów aktywnych nie jest pusta to wtedy, najpóźniej po 2 ms proces jest przerywany automatycznie (realizuje to odpowiedni program szeregujący wbudowany w układ sterowania transputera) i umieszczany na końcu listy, a do realizacji wybierany jest proces z początku listy. Mechanizm ten pozwala zatem dzielić czas procesora, gwarantując współbieżną realizację kilku procesów. Takie automatyczne przełączanie nie ma miejsca w przypadku procesów o wysokim priorytecie. Są one wykonywane w całości, chyba że wcześniej zostaną przerwane oczekując na zakończenie transmisji lub upływ określonego odcinka czasu.

Do komunikacji pomiędzy procesami wykorzystywane są kanały. Jeśli oba komunikujące się procesy działają na tym samym transputerze, kanałem jest pojedyncze słowo pamięci wewnętrznej RAM. W przeciwnym razie jest nim link. W obu przypadkach procesy nadający i odbierający informację muszą być do tego gotowe. Nie jest przy tym ważne, który z nich jest gotów jako pierwszy. Transmisja danych poprzez link odbywa się w sposób szeregowy ze standardową szybkością 10 Mb/s. Alternatywnie może być ona realizowana również z szybkościami 5 lub 20 Mb/s.

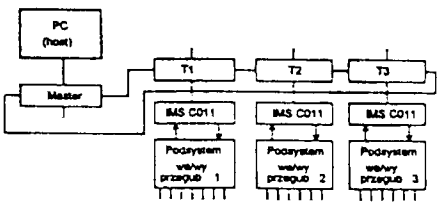
Do budowy układu sterowania wykorzystano kartę Quintek-Fast9 współpracującą z mikrokomputerami klasy PC [9,10]. Posiada ona 9 gniazd, w których mogą być umieszczone transputery np. typu IMS T800. Każdy transputer ma własną pamięć RAM o pojemności do 4 MB. W typowym przypadku jej pojemność wynosi 1 MB za wyjątkiem pamięci transputera współpracującego z mikrokomputerem, w którym umieszczono kartę Fast9 (master-transputer) - ma ona 4 MB.

Mikrokomputer (*host*), w którym umieszczono kartę Fast9 współpracuje z siecią transputerów IMS T800 poprzez *Link0* master-transputera. Ten z kolei wraz z pozostałymi trzema (T1-T3) tworzy pokazaną na rys.2 strukturę pierścieniową. Połączenia pomiędzy kolejnymi transputerami struktury (*Link0-link2*) są stałe. Wolne linki transputerów T1, T2 i T3 (*Link1, Link3*) połączono z programowo konfigurowalną przełącznicą (układ typu IMS C004) [8]. Układ IMS C004 pozwala równocześnie realizować połączenia pomiędzy

dwoma dowolnymi, doprowadzonymi do niego sygnałami. Dzięki temu można tworzyć dodatkowe kanały komunikacyjne między transputerami T1-T3.

Ponieważ celem prowadzonych prac było wykonanie regulatora cyfrowego robota, zatem zasadnicze znaczenie miało zapewnienie współpracy transputerów realizujących algorytm sterowania z urządzeniami zewnętrznymi.

Najbardziej efektywną metodą współpracy zewnętrznych urządzeń wejścia-wyjścia z siecią transputerową zbudowaną w oparciu o kartę Quintek-Fast9 jest ich dołączenie do linków transputerów za pośrednictwem adapterów typu IMS C011. Rys.2. ilustruje w sposób poglądowy przyjętą metodę współpracy. Założono, że z każdym przegubem robota laboratoryjnego KRĘPY współpracuje jeden transputer. Układ IMS C011 (adaptera linku) jest elementem pośredniczącym. Z jednej strony komunikuje się on porzez link z transputerem, a z drugiej przy pomocy dwóch ośmiobitowych, jednokierunkowych szyn z podsystemem WE/WY.



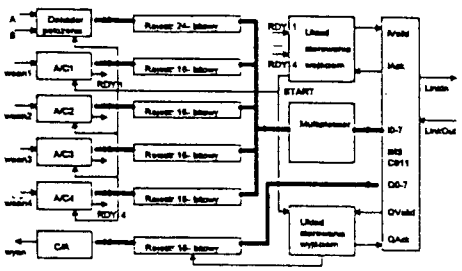
Rys.2. Współpraca sieci transputerowej z urządzeniami wejścia/ wyjścia.

Zapewnia to bardzo dużą szybkość transmisji danych oraz krótki czas reakcji systemu na zdarzenia zewnętrzne. Równocześnie nie ma tu potrzeby ingerencji w układowe rozwiązania zastosowane na karcie Quintek-Fast9, co byłoby nieodzowne w przypadku włączenia urządzeń zewnętrznych w przestrzeń adresową transputerów.

3. PODSYSTEM WEJŚCIA-WYJŚCIA

Na rys.3. przedstawiono schemat blokowy jednego z trzech kanałów podsystemu wejścia-wyjścia transputerowego regulatora cyfrowego. Układ sterowania wejściem oraz ośmiobitowy multiplekser cyfrowy zapewniają transmisję danych pomiarowych do regulatora, a układ sterowania wyjściem wraz z zespołem rejestrów ośmiobitowych z regulatora. Generowany w regulatorze sygnał *START* równocześnie rozpoczyna odczyt oraz zapis danych. Na początku inicjowany jest cykl przetwarzania we wszystkich przetwornikach A/C, dzięki czemu sygnały pomiarowe dotyczą tej samej chwili czasowej. Oprócz tego *START* ustawia oba pokazane na rys.3. układy sterowania w stan początkowy. Po zakończeniu przetwarzania regulator może odczytać informację. Sygnały gotowości (*RDY1, RDY4*) gwarantują, że transmisja rozpocznie się dopiero, gdy dane będą dostępne. Układ sterowania wejściem dokonuje wyboru wejść multipleksera, które mają być

aktualnie przekazane, przy pomocy linii adresowych zgodnie z przyjętą wcześniej kolejnością. Zmiana stanu tych linii następuje po przetransmitowaniu bajtu W przypadku transmisji wyjściowej układ sterowania generuje impulsy zapisu do rejestrów ośmiobitowych. Podobnie jak poprzednio kolejność zapisu do tych rejestrów jest z góry określona.



Rys.3. Schemat blokowy podsystemu wejścia-wyjścia

Podstawowym mierzonym sygnałem jest położenie osi silnika. Pomiar ten jest w pełni cyfrowy. Zastosowany enkoder optyczny generuje 1000 impulsów na jeden obrót tarczy. Poprzez czytanie wszystkich zboczy (narastających i opadających) czterokrotnie zwiększono rozdzielczość pomiaru położenia osiągając dokładność po stronie ramienia 4000*84 impulsów/obrót. Sygnały te trafiają na wejście enkodera położenia z 24-bitowym licznikiem rewersyjnym i detektorem kierunku zapisanym w pamięci PROM. Licznik rewersyjny zlicza aktualne położenie osi silnika, które jest zapamiętywane w rejestrach pośrednich a następnie kierowane do trzech ośmiobitowych rejestrów buforowych. Rejestry buforowe zapewniają bezkolizyjne przekazywanie danych na wewnętrzną magistralę danych wejściowych I0-7.

Do wewnętrznej magistrali I0-7 podpięte są również bloki czterech przetworników analogowo-cyfrowych przetwarzających sygnały obiektowe WEAN0 - WEAN3. Zawierają one układy typu ADS7800 firmy Burr-Brown [14]. Odczyt sygnału analogowego inicjuje sygnał R/C. W odpowiedzi na to przetwornik wystawia po czasie około 100ns sygnał BUSY informujący o przeprowadzanej konwersji danych. Konwersja danych kończy się po 2.7μs, o czym informuje wysoki stan BUSY. 12-bitowa dana transmitowana jest do adaptera IMS C011 w postaci dwóch bajtów. Kolejność odczytu danych ustalają sygnały R2 i R3. Przyjęte rozwiązanie zapewnia wyższe parametry przetwarzania sygnałów analogowych niż układ z jednym przetwornikiem, multiplekserem i układem pamiętającym. Koszt realizacji obu realizacji jest zbliżony.

Podsystem wejścia-wyjścia ma jedno wyjście analogowe oznaczone symbolem WYAN. W układzie zastosowano szybki (czas konwersji 150 ns), 12-bitowy przetwornik cyfrowo-analogowy typu AD565A firmy Analog-Devices. Ponieważ układ ten nie posiada wewnętrznych rejestrów zastosowano zewnętrzne popularne układy typu 74LS374..

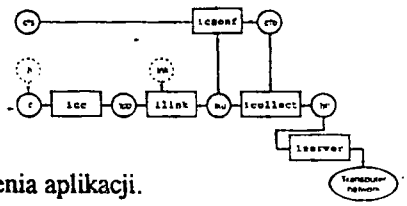
W celu elastycznego konfigurowania podukładu wejść-wyjść wprowadzono układ dekodera wejść analogowych-DWA. W skład DWA wchodzą klucze K0-K3 ściśle odpowiadające wejściom analogowym WEAN0-WEAN3. W celu odłączenia pewnego kanału analogowego należy jednocześnie ustawić klucz w pozycji "rozwarty" i odłączyć je programowo. Cykl pracy podsystemu wejścia-wyjścia uruchamia niski stan R/C podawany na wszystkie bloki wejściowe. W trakcie trwającej dość długo operacji przetwarzania danych przez układy A/D (3 s) obsługiwane jest wyjście analogowe a następnie enkoder położenia. Po tym etapie następuje odczyt rejestrów R3-R10 przechowujących dane z wejść analogowych.

4. WARSTWA PROGRAMOWA STEROWNIKA

W języku Quintek-ANSI C realizowany jest model programowania procesów równoległych zgodny z notacją CSP, w którym niezależne, wykonywane równolegle procesy komunikują się ze sobą za pośrednictwem kanałów. Model ten w sposób naturalny daje się odwzorować na sieci transputerów połączonych wzajemnie linkami.

Przygotowanie i uruchomianie programów na sieci transputerowej odbywa się z pomocą kilku narzędzi programistycznych. Narzędzia te tworzące tzw. "ANSI C Toolset" to: a) kompilator (icc) , który posiada funkcje kreujące procesy równoległe oraz realizujące komunikację poprzez kanały. b) linker (ilink) - łączący oddzielnie skompilowane moduły oraz dołącza zbiory biblioteczne, c) konfigurator (iconf) - tworzący informację o alokacji modułów programu w sieci transputerów. d) konsolidator (icollect), e) program ładujący (iserver)., który dodatkowo dostarcza narzędzi do komunikacji z host-komputerem w czasie wykonywania programu transputerowego (w czasie *run-time*).

Proces tworzenia aplikacji przedstawiono schematycznie na rys.4. W kółkach zapisano przyjmowane standardowo rozszerzenia zbiorów generowanych na poszczególnych etapach uruchamiania programu. W razie potrzeby mogą być one zmienione.



Rys. 4. Proces tworzenia aplikacji.

Z punktu widzenia programisty najważniejsze są nowe konstrukcje języka C umożliwiające współbieżną realizację procesów oraz sposób opisu topologii sieci transputerowej akceptowany prze konfigurator *iccof*. W celu umożliwienia prowadzenia obliczeń współbieżnych język Quintek-ANSI C został uzupełniony o nowe typy danych oraz funkcje biblioteczne. Dodatkowe typy danych definiowane w zbiorach nagłówkowych *process.h, channel.h, semaphore.h* to:

- Process : struktura, w której przechowywana jest informacja o procesie zawierająca między innymi wskaźniki do obszaru roboczego procesu w pamięci operacyjnej, początkową wartość wskaźnika instrukcji oraz wskaźnik do kodu procesu,
- Channel : wskaźnik (*pointer*) wykorzystywany podczas implementacji kanału. Wskazuje on na zmienną kanałową realizującą, zgodnie z notacją CSP jednokierunkowe łącze między dwoma procesami,
- Semaphore : struktura przechowująca informacje o semaforze.

Funkcje biblioteczne działające na tych zmiennych realizują następujące operacje:
a) tworzenie, uruchamianie i szeregowanie procesów,
b) komunikacja za pośrednictwem kanałów,
c) działania na semaforach.

Część funkcji uruchamia proces w trybie asynchronicznym. Tak uruchomiony proces działa niezależnie od procesu który go powołał, w szczególności może on być realizowany również po zakończeniu głównego modułu programu.

Pozostałe funkcje rozpoczynają procesy w trybie synchronicznym. Polega to na tym, że sterowanie powraca do procesu, z którego wydano funkcję rozpoczęcia procesu typu *ProcPar* dopiero po zakończeniu wszystkich rozpoczętych tą funkcją procesów.

Należy wyraźnie podkreślić, że wymiana informacji między dwoma procesami wymaga gotowości obu - jeden musi realizować operację zapisu (*Out*), a drugi odczytu (*In*) odpowiedniej porcji danych. W przeciwnym razie jeden z pary takich procesów zostanie zawieszony w oczekiwaniu na komunikację. Odpowiada to modelowi obliczeń współbieżnych zgodnemu z notacją CSP.

Topologia sieci transputerowej

Aby określić rozłożenie modułów programowych na sieci transputerowej oraz skojarzyć kanały z linkami należy stworzyć zbiór opisu konfiguracji. Zbiór ten zapisany w specjalnym języku (transputer configuration language) jest przetwarzany poprzez konfigurator *icconf* do postaci akceptowalnej przez konsolidator. W zbiorze opisu topologii sieci definiuje się:
- sieć programową złożoną z procesów połączonych kanałami wejścia-wyjścia.
- sieć sprzętową, w skład której wchodzą procesory (transputery.)
- odwzorowanie sieci programowej na sieć sprzętową.

W języku opisu konfiguracji przyjęto do opisu sieci programowej i sprzętowej wspólną składnię opartą na deklaracji węzłów (*nodes*) z połączeniami definiowanymi przy pomocy wyrażeń języka. Każdy węzeł sieci posiada kilka atrybutów, których ilość i natura zależą od wspólnego atrybutu *element*. Atrybut ten w szczególności określa czy dany węzeł jest węzłem sieci programowej (procesem), czy sprzętowej (procesorem).

Określone w ten sposób węzły sieci są następnie łączone za pośrednictwem kanałów przy pomocy dyrektywy *connect* .Kanały mogą być przy tym jednokierunkowe (programowe - zgodnie z notacją CSP) lub dwukierunkowe (linki transputerów). Ich rodzaj zależy od tego, jaka sieć jest aktualnie opisywana.

Sieć programowa

Sieć programowa składa się z predefiniowanych węzłów *process* połączonych jednokierunkowymi kanałami wejścia-wyjścia. Opis sieci obejmuje więc definicje węzłów i połączeń między nimi. Dodatkowo zawiera on także informacje pozwalające skojarzyć procesy ze skompilowanymi i zlinkowanymi modułami programowymi.

Sieć sprzętowa

W skład sieci sprzętowej wchodzą węzły typu *processor* połączone ze sobą linkami. Węzeł *processor* ma oprócz atrybutu *element* jeszcze dwa atrybuty:
- type : typ transputera,
- memory : rozmiar dostępnej pamięci.

Linki należy traktować jako specjalny atrybut definiowany jednoznacznie wyborem typu procesora (atrybutem *type*). Programista nie może więc zmieniać jego wartości. Do łączenia linków traktowanych jako tablice o rozmiarze zgodnym z typem procesora (najczęściej równym 4) służy dyrektywa *connect*. Jest ona używana w identyczny sposób jak przy opisie sieci programowej. Jedyną różnicą jest to, że linki są kanałami dwukierunkowymi. W poniższym przykładzie zdefiniowano połączenie, w którym zerowy link pierwszego transputera jest połączony z trzecim linkiem drugiego transputera:

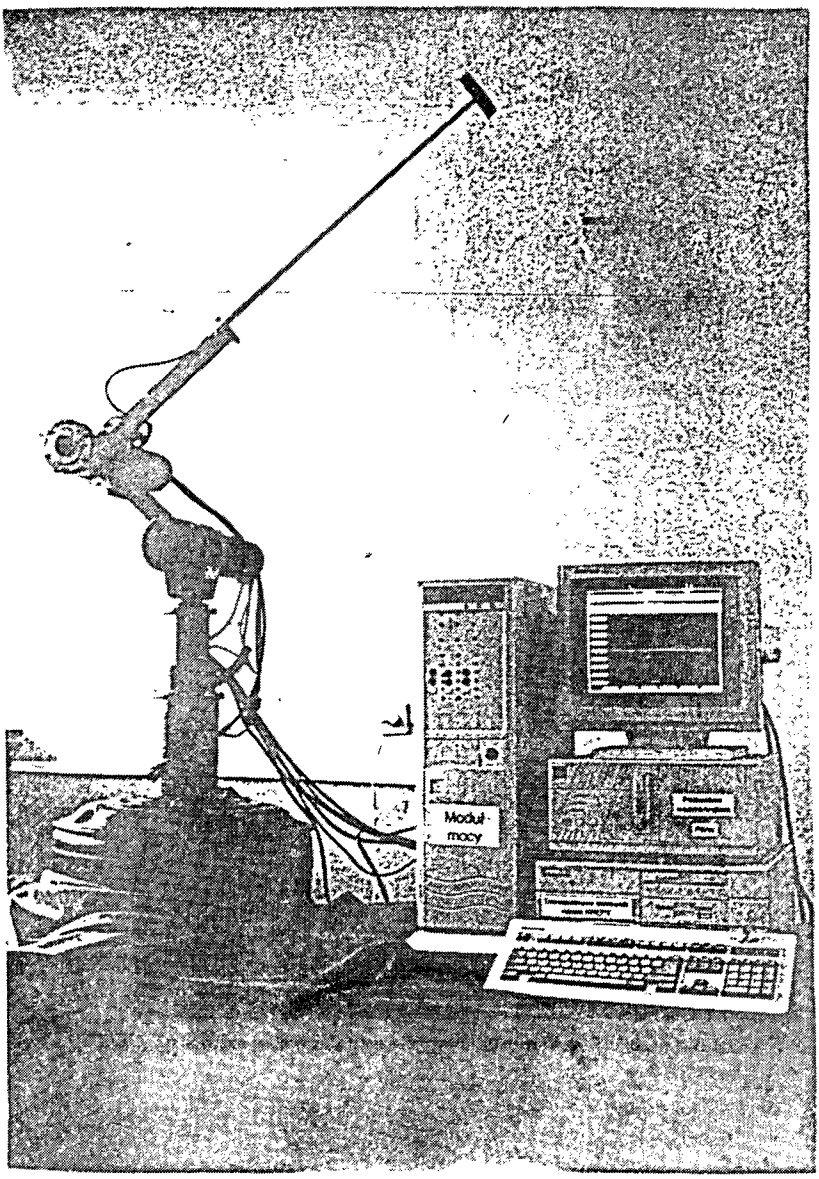
Odwzorowanie sieci sprzętowej na sieć programową

Opis odwzorowania sieci programowej na sieć sprzętową jest ostatnim etapem definiowania topologii układu. Określa się w nim, na których procesorach (węzłach sieci programowej) zostaną umieszczone poszczególne procesy. Dokonuje się tu również skojarzenia programowych kanałów wejścia-wyjścia z linkami. Odwzorowanie to jest realizowane przy użyciu dyrektywy *place*.

5. ROZWIĄZANIE SPRZĘTOWE

Pierwotne założenia przewidywały wykonanie układu sterowania w standardzie podwójnej Eurokarty. Jednak ze względu na koszty i dostępność gotowych i tanich elementów mechanicznych przyjęto standard PC, zmieniając złącza bazowe na zalecane dla urządzeń prototypowych. Karta Quintek-Fast9 oraz płyta z adapterami linku C011 są umieszczone w obudowie Host-komputera (IBM 386/387). Podsystem WE/WY zbudowano w formie indywidualnego modułu elektronicznego zamkniętego w typowej obudowie Baby-PC. Moduł składa się z zasilacza oraz płyty bazowej, na której przymocowano sześć pakietów. Jedna para pakietów obsługuje pojedynczy przegub robota laboratoryjnego. Ponadto w skład układu sterowania wchodzi wzmacniacz mocy (trzy kanały) oraz moduł zawierający filtry analogowe, wzmacniacze instrumentalne oraz układy dopasowujące, których celem jest sprzęganie przetworników z podsystemem WE/WY.

Stanowisko badawcze z transputerowym stemem sterowania i robotem laboratoryjnym KRĘPY przedstawia rys.5.



Rys.5. Stanowisko badawcze z transputerowym sterownikiem i robotem KRĘPY.

PODSUMOWANIE

W związku z postępem technologicznym w elektronice i elektrotechnice należy spodziewać się, że w niedalekiej przyszłości nawet w tańszych robotach znajdą zastosowanie układy wieloprocesorowe. Obecnie, na świecie trwają intensywne badania w tym zakresie [11,15]. Konieczne jest zatem dążenie do opanowania projektowania i wytwarzania takich systemów w kraju.

Opisany w artykule układ sterowania stanowi początkowy etap badań nad wieloprocesorowymi układami sterowania robotów. Obecnie w ZAiI PRz kontynuowane są prace nad działającym w sposób równoległy adaptacyjnym, wielowymiarowym algorytmem sterowania robotem KRĘPY.

LITERATURA

1. J.J.Craig, "Introduction to Robotics Mechanics and Control", Addison-Wesley Publishing Company, 1986
2. L.Trybus, "Dobór nastaw serwomechanizmów PID o strukturze jak w robotach IRb", Prace Nauk. ICT Pol. Wrocł., Konf. Nr 33, 1988
3. W.Irzeński, "Dokładne pozycjonowanie robotów przemysłowych", praca doktorska, Wydział Budowy Maszyn i Lotnictwa, Pol. Rzesz., 1992
4. R.Leniowski, "KRĘPY - robot laboratoryjny o sterowaniu cyfrowym, charakterystyka urządzenia", Prace Nauk. ICT Pol. Wrocł., Konf. Nr 38, 1990
5. J.J.Craig, "Adaptive Control of Mechanical Manipulators", Addison-Wesley Publishing Company, 1988
6. "The Transputer Databook", INMOS Databook Series, INMOS Ltd, 1989
7. Praca zbiorowa, "Programowanie współbieżne - wybrane zagadnienia", Pol. Śląsk., Skrypt nr 1638, 1991
8. "ANSI C Toolset. User Manual", INMOS Ltd, 1990
9. "The Fast9. A Nine Transputer Board for the IBM PC. User Manual", Quintek Ltd, 1988
10. "Quintek Information", Quintek Ltd, 1991
11. "M.I.Barlow, P.Konnanov, S.E.Burge, "Analogue I/O strategies for transputers", Microprocessors and Microsystems, Vol 13, No 6 July/August 1989
12. R.Leniowski, "Model matematyczny i metoda kompensacji wibracji robota z elastycznym ramieniem", praca doktorska, Wydzail Automatyki, Informatyki i Elektroniki, Pol. Śląska, Gliwice 1992
13. "Dynaserv-Direct Drive Servo Actuators", Yokogawa Corporation of America, Lake Geneva, inf. PCIM, oct. 1989, no.105
14. "Burr-Brown Integrated Circuits Data Book-vol. 33", Burr-Brown Corporation, Tuscon, USA, 1989
15. D.T.Pham, Hu Houseng, J.Pote, "A transputer-based system for locating parts and controlling an idustrial robot", Robotica, vol.8, pp.97-103, 1990