Prof. dr hab. inż. Jerzy Honczarenko
Mgr inż. Mariusz Sosnowski
Politechnika Szczecińska
Zakład Zautomatyzowanych Systemów Wytwarzania

IDEA OPROGRAMOWANIA UKŁADU STEROWANIA BADAWCZEGO ELASTYCZNEGO SYSTEMU WYTWARZANIA

W referacie przedstawiono ideę oprogramowania sterującego Badawczym Elastycznym Systemem Wytwarzania na podstawie wygenerowanych harmonogramów w dowolnym oprogramowaniu. Omówiono strukturę systemu, którego architektura sterowania jest swobodnie kształtowana w zależności od potrzeb użytkownika, oraz opracowane algorytmy, programów i podprogramów sterujących, stanowiących „otwarty system" do badań nowych form planowania, harmonogramowania i sterowania produkcją.

IDEA OF SOFTWARE SYSTEM CONTROL RESEARCH OF FLEXIBLE MANUFACTURING SYSTEM

The paper presents conception of software control research of flexible manufacturing system according on the generated schedules in any software. The structures of the system have been looked into which means and the architecture of control to form in subjection from user's needs free. It algorithms, programs and subprograms have control and making up "the open system" to research of new forms the planning, schedules and control the production.

1. WPROWADZENIE

W Zakładzie Zautomatyzowanych Systemów Wytwarzania Politechniki Szczecińskiej zbudowano badawczy elastyczny system wytwarzania. System ma cechy rzeczywistego systemu, lecz wykonany jest w postaci zminiaturyzowanej. Celem budowy systemu jest umożliwienie prowadzenia doświadczalnej weryfikacji komputerowych symulacji nowych metod planowania, harmonogramowania oraz wdrażania nowych metod sterowania produkcją.

Referat jest kontynuacją badań nad systemem, dla którego wyniki wcześniejszych doświadczalnych weryfikacji i próby sterowania przy użyciu sztucznej inteligencji zostały opisane w publikacjach [1][2].

Celem referatu jest przedstawienie oprogramowania sterującego badawczym elastycznym systemem wytwarzania, swobodnie kształtowanego w zależności od potrzeb użytkownika. Opracowany główny program i podprogramy sterujące, stanowią „otwarty system" do badań nowych form planowania, harmonogramowania i sterowania produkcją.

2. ZAŁOŻENIA PROJEKTOWE UKŁADU STEROWANIA

Podczas budowy systemu przyjęto, że ma on umożliwiać prowadzenie doświadczalnej weryfikacji różnych metod planowania, harmonogramowania i sterowania produkcją również z wykorzystaniem sztucznej inteligencji.

Założono otwartą budowę i modułową strukturę sterowania, z zastosowaniem lokalnych sieci komputerowych w celu zapewnienia elastycznego przepływu informacji sterujących.

Stąd **kolejno** wypłynęły następujące założenia pozwalające na swobodną rozbudowę i modyfikację:

- ✓ Układ sterowania badawczego systemu powinien pozwalać na tworzenie różnych typów architektur sterowania, począwszy od scentralizowanej, hierarchicznej, hybrydowej po rozproszoną.
- ✓ Sterowanie systemem powinno być kształtowane swobodnie, w zależności od celu i zadania badawczego, tzn. może być dostosowane do potrzeb użytkownika.
- ✓ Łatwość reorganizacji architektury sterowania zapewniona będzie dzięki modułowej budowie pozwalającej na swobodną wymianę rodzaju i kolejności informacji przez lokalną sieć komputerową.
- ✓ Każdy podsystem funkcjonalny jest sterowany przez oddzielny komputer podsystemowy, a komputery elastycznie powiązane siecią maja przypisane numery identyfikacyjne.
- ✓ Każdy z programów zainstalowany w oddzielnym komputerze podsystemowym spełnia wydzielone zadanie podczas sterowania systemem.
- ✓ Pojedyncze funkcje sterujące programuje się w postaci podprogramów, tworzonych w języku wyższego rzędu.
- ✓ Każdemu elementarnemu przemieszczeniu układarki, wózka i robota odpowiada jeden podprogram sterujący, wywoływany zgodnie z zadanym harmonogramem.

W pierwszym etapie prac badawczych zbudowano układ sterowania o architekturze scentralizowanej. W drugim etapie prac przewidziano zbudowanie architektury rozproszonej z zastosowaniem metod agentowych i holonicznych.

3. STRUKTURA PROGRAMOWA UKŁADU STEROWANIA

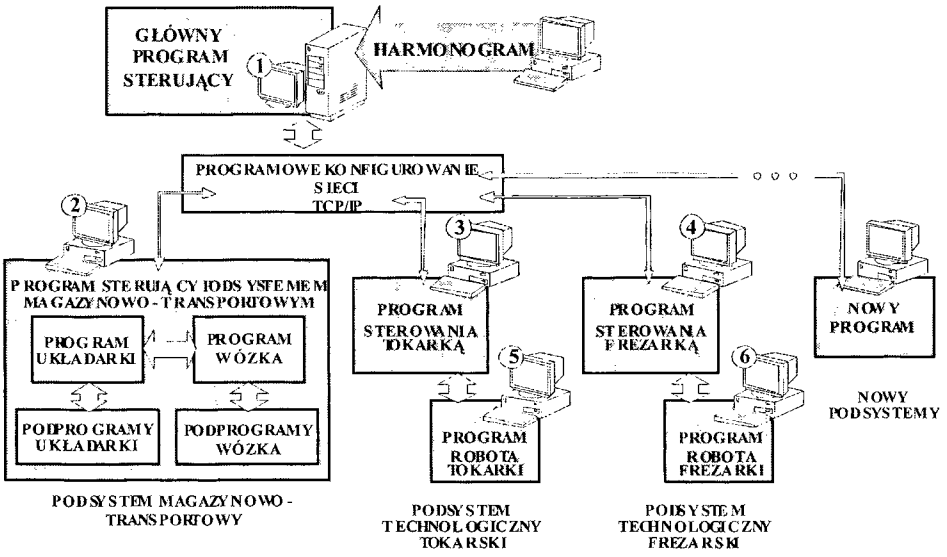
Na podstawie założeń zaprojektowano strukturę systemu składającego się z sześciu komputerów, z których jeden pełni funkcję komputera głównego a pozostałe pełnią funkcję komputerów sterujących elementami wykonawczymi poszczególnych podsystemów, co przedstawiono na rys. 1.

Funkcję wymiany informacji pomiędzy poszczególnymi programami pełni komputer nadrzędny ①. W komputerze nadrzędnym zainstalowany jest główny program sterujący, który umożliwia kontrolę i sterowanie całym systemem. Wprowadzone tu informację są przekazywane do kolejnych programów sterujących podsystemami magazynowo-transportowym oraz podsystemów technologicznych w celu ich realizacji. W przypadku architektury scentralizowanej komputer nadrzędny staje się komputerem

centralnym współpracującym ze wszystkimi pozostałymi komputerami systemu sterowania, natomiast w przypadku sterowania rozproszonego uczestniczy on zarówno w procesie sterowania jak i podejmuje decyzje opierające się na wzajemnej negocjacji i koordynacji działań.

Sterowanie systemem można realizować metodą off-line lub on-line. W pierwszym przypadku harmonogramy pracy będą generowane w innym komputerze np. w biurze technologa i przesyłane do komputera nadrzędnego za pomocą Internetu. Wprowadzony do głównego programu harmonogram pracy tworzony może być w dowolnym oprogramowaniu. Wygenerowany harmonogram w postaci pliku tekstowego jest kompleksową informacją na temat wszystkich ruchów urządzeń znajdujących się w systemie, od pobrania pierwszej palety przedmiotowej z magazynu do odłożenia na regał magazynu ostatniej dostarczonej przez wózek.

Przy sterowaniu on-line, wykorzystując metody sztucznej inteligencji, harmonogramy pracy tworzone są dynamicznie w komputerze nadrzędnym i korygowane w miarę napływu zleceń z zewnątrz.



Rys. 1. Struktura programowa badawczego elastycznego systemu wytwarzania

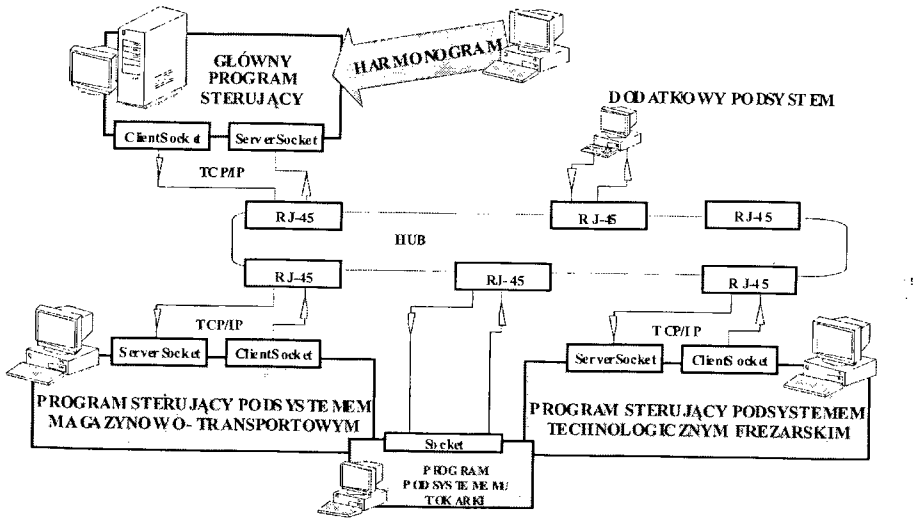
Wygenerowany harmonogram z modelu symulacyjnego lub operacyjnego, przesłany zostaje w postaci procedur sterujących (plik tekstowy) do głównego programu sterującego w celu jego realizacji.

Komputery ②, ③, ④ wraz z oprogramowaniem odpowiedzialne są za pracę poszczególnych podsystemów. W podsystemie magazynowo-transportowym komputer ② steruje wózkiem transportowym, układarką regałową oraz czyta informacje ze stanowisk odkładczych i magazynu regałowego. W podsystemie technologicznym tokarskim komputer ⑤ steruje robotem, a komputer ③ steruje tokarką. Komputer

sterujący tokarką pełni rolę nadrzędną w stosunku do komputera sterującego robotem. Podobnie w podsystemie technologicznym frezarskim nadrzędny komputer ④ steruje frezarką, a komputer ⑥ robotem. Komputery podsystemowe uzyskują dane z nadrzędnego komputera i przekazują informacje do odpowiednich urządzeń wykonawczych.

Komunikacja między programami zainstalowanymi w komputerach ①, ②, ③, ④ odbywa się za pomocą protokołu sieciowego TCP/IP. Ideę wymiany informacji sterujących przedstawiono na rys. 2.

Każdy z programów sterujących wchodzących w skład sieci internetowej ma własny numer identyfikacyjny IP. Adres IP jest informacją o odpowiednim adresacie (programie), do którego są przesyłane odpowiednie informacje sterujące. W momencie dołączenia dodatkowego programu (podsystemu) następuje automatycznie przyznawany adres IP, dzięki czemu istnieje możliwość łatwej reorganizacji architektury sterowania. Zaletą tego rozwiązania jest swobodne przyłączanie programów sterujących nowych podsystemów jednak rozwiązanie to ograniczone jest liczbą wejść do koncentratora HUB.



Rys. 2. Wymiana informacji sterujących pomiędzy programami

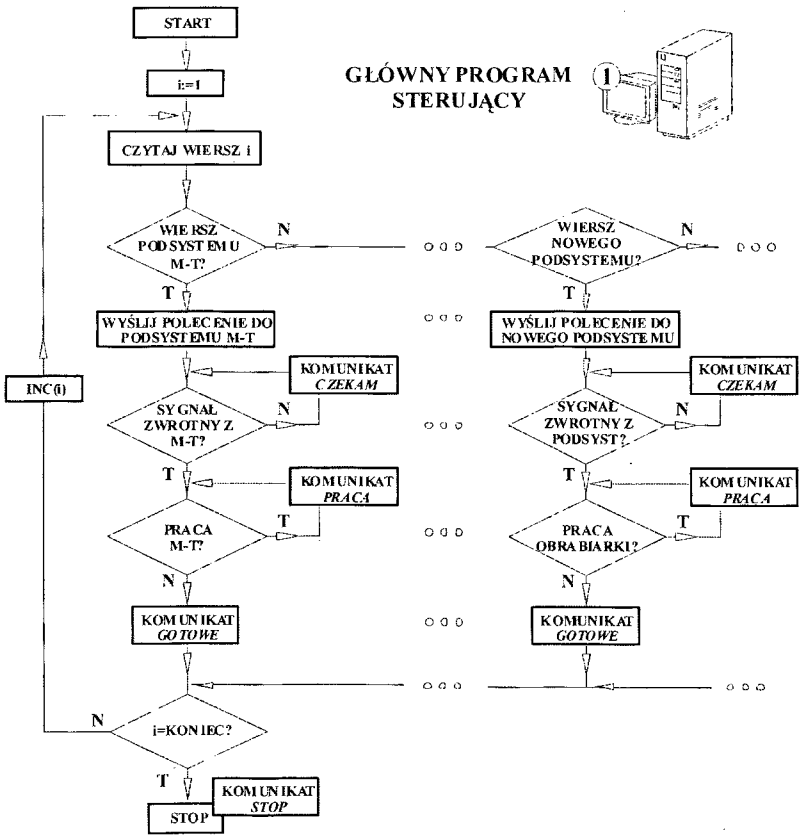
4. GŁÓWNY PROGRAM STERUJĄCY

Harmonogram pracy systemu przesłany do komputera nadrzędnego w metodzie off-line lub w nim wygenerowany metodą on-line przetwarzany jest w algorytm sterowania systemem, co przedstawiono na rys. 3.

Algorytm głównego programu pozwala na wczytywanie do pamięci poszczególnych sekwencji sterujących z harmonogramu, a następnie przesyłaniu ich do odpowiednich podsystemów w celu jego realizacji. Pojedyncze polecenie sterujące wysłane do określonego podsystemu musi być potwierdzone sygnałem zwrotnym, a następnie

wyświetlony zostaje stosowny komunikat. Algorytm pracy systemu kończy się w momencie wczytania ostatniego wiersza z harmonogramu. Główny program sterujący został opracowany i zaimplementowany w języku Delphi 6, a przykładowe okno z udziałem sekwencji sterujących wózkiem i układarką przedstawiono na rys. 4.

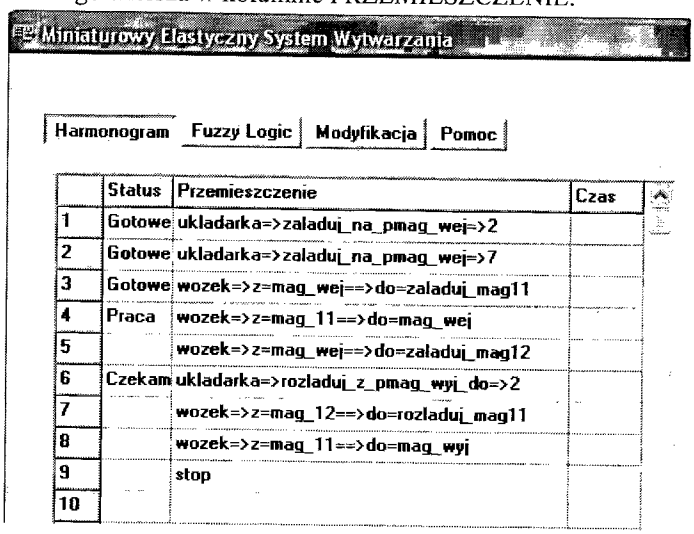
Program składa się z kilku zakładek: *Harmonogram*, *Fuzzy Logic*, *Modyfikacja* i *Pomoc*. Dla zakładki *Harmonogram* w głównym programie sterującym, podstawowym elementem jest centralnie umieszczona tablica, do której wczytuje się wierszami sekwencje procedur sterujących utworzone na podstawie harmonogramu produkcji. Po ukończeniu określonej sekwencji ruchu następuje przeskok do kolejnego wiersza harmonogramu. Każde przemieszczenie kontrolowane jest przez program sterujący, który w kolumnie STATUS wyświetla aktualny stan danego wiersza.



Rys. 3. Algorytm głównego programu sterującego

Przykładowy harmonogram, przedstawiony na rys. 4, składa się z wierszy sterujących, z których trzy pierwsze zostały aktualnie wykonane przez program sterujący. Przykładowo, sekwencja nr 6 związana jest z przemieszczeniem wózka tzn. pobraniem palety ze stanowiska odkładczego tokarskiego *mag_11* i odłożeniem na stanowisko

wyjściowe *mag_wyj* do magazynu. Cały program sterujący kończy się w momencie wczytania ostatniego wiersza w kolumnie PRZEMIESZCZENIE.



	Status	Przemieszczenie	Czas
1	Gotowe	ukladarka=>zaladuj_na_pmag_wej=>2	
2	Gotowe	ukladarka=>zaladuj_na_pmag_wej=>7	
3	Gotowe	wozek=>z=mag_wej==>do=zaladuj_mag11	
4	Praca	wozek=>z=mag_11==>do=mag_wej	
5		wozek=>z=mag_wej==>do=zaladuj_mag12	
6	Czekam	ukladarka=>rozladuj_z_pmag_wyj_do=>2	
7		wozek=>z=mag_12==>do=rozladuj_mag11	
8		wozek=>z=mag_11==>do=mag_wyj	
9		stop	
10			

Rys. 4. Fragment głównego programu sterującego

W zakładce *Modyfikacja* wbudowano interpreter COM (ang. *Component Object Model*), która umożliwia rozbudowę algorytmu pracy o dodatkowe sekwencje sterujące a tym samym o dodatkowe podsystemy [4].

W zakładce *Pomoc*, przedstawiono zasadę obsługi głównego programu sterującego.

W zakładce *Fuzzy Logic* sterowanie systemem odbywa się on-line z wykorzystaniem logiki rozmytej. W artykule [2] przedstawiono ideę i działanie systemu z wykorzystaniem wnioskowania rozmytego.

5. PROGRAMY STERUJĄCE PODSYSTEMAMI

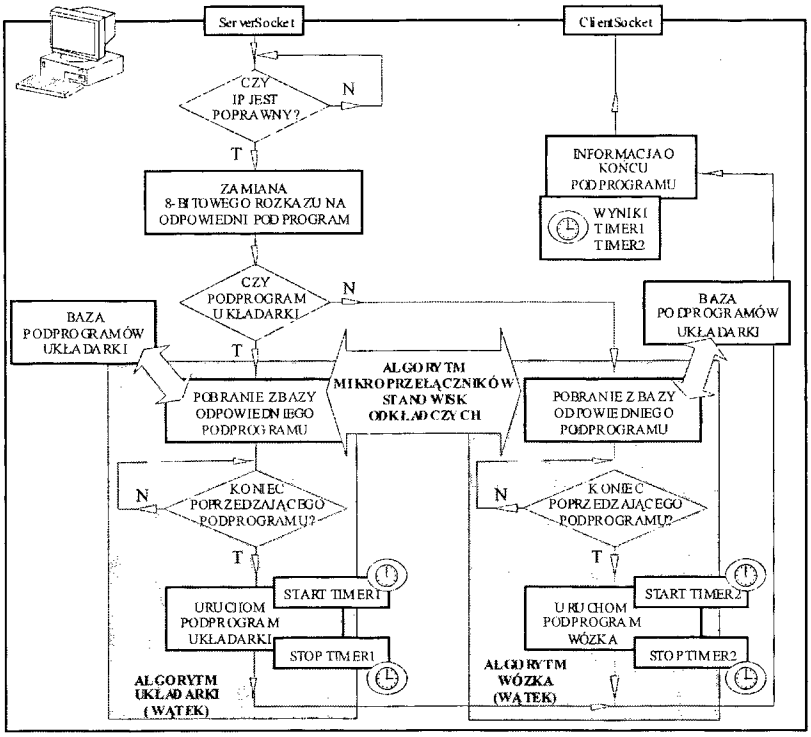
5.1. Program sterujący podsystemem magazynowo-transporotwym

Program sterujący pracą układarki regałowej i wózka transportowego zainstalowany jest w komputerze podsystemu magazynowo-transportowego. Algorytm pracy podsystemu magazynowo-transportowego przedstawiono na rys. 5.

Oprócz plików uruchomieniowych zainstalowane są dwie bazy, w których znajdują się podprogramy sterujące układarką regałową i wózkiem transportowym. Podprogramy zbudowane są w postaci dynamicznie uruchamialnych plików [2].

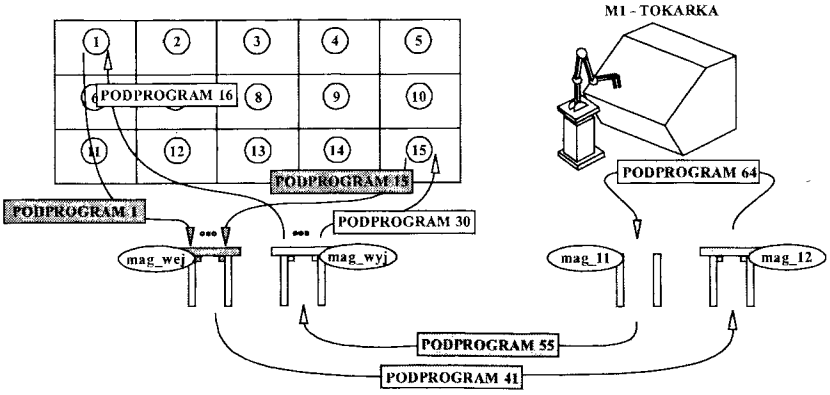
Program zawarty w komputerze magazynowo-transportowym składa się z niezależnie działających algorytmów (wątków), sterujących pracą układarki i wózka, co przedstawiono na rys. 5. Pomiędzy głównymi wątkami zaimplementowano wątek, którego zadaniem jest nadzorowanie informacji sterujących między układarką a wózkiem celem zapobiegania kolizji. Odczyt stanu wszystkich mikroprzełączników stanowisk odkładczych w systemie jest dokonywany na bieżąco. Bloki baz

podprogramów układarki i wózka działają w nieskończonej pętli, pobierane są z nich właściwe biblioteki sterujące, natomiast dwa stopery odliczają czas ich realizacji.



Rys. 5. Algorytm pracy podsystemu magazynowo-transportowego

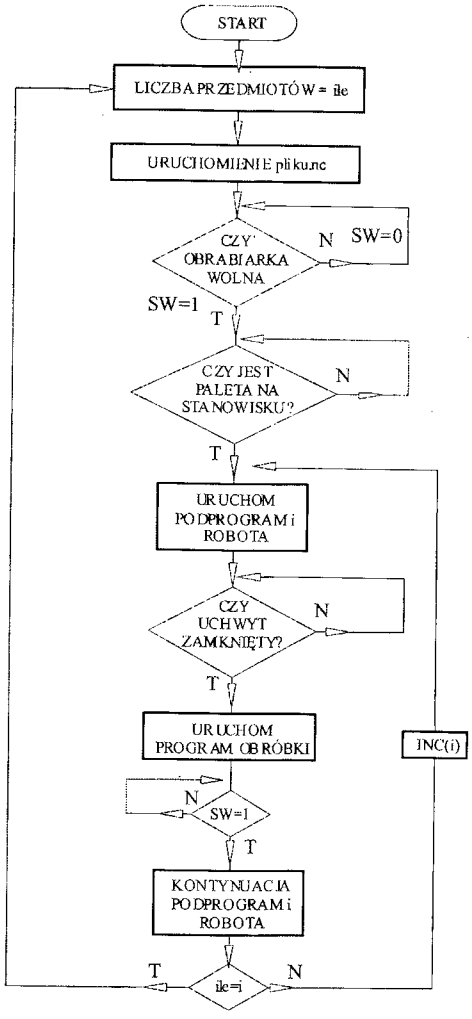
Dla zrealizowania wszystkich założonych możliwych przemieszczeń opracowano 30 podprogramów sterujących przemieszczeniami układarki oraz 54 podprogramy sterujące pracą wózka, co przedstawiono fragmentarycznie na rys. 6.



Rys. 6. Podprogramy sterujące podsystemem magazynowo-transportowym

5.2. Implementacja programów sterujących podsystemami technologicznymi

Tokarka i frezarka pracujące w podsystemach technologicznych są sterowane za pomocą oprogramowania zakupionego wraz z maszynami i zainstalowanego w oddzielnych komputerach – pełniących funkcję układów sterowania tych obrabiarek. Do współpracy obrabiarki z robotem wykorzystano przede wszystkim gotowe komendy służące do sterowania urządzeń zewnętrznych. Komendy umożliwiają sterowanie poziomami logicznymi na wyjściach TTL I/O interfejsu obrabiarki. Algorytm pracy na przykładzie podsystemu frezarskiego przedstawia rys. 7.



Rys. 7. Algorytm działania oprogramowania podsystemów technologicznych

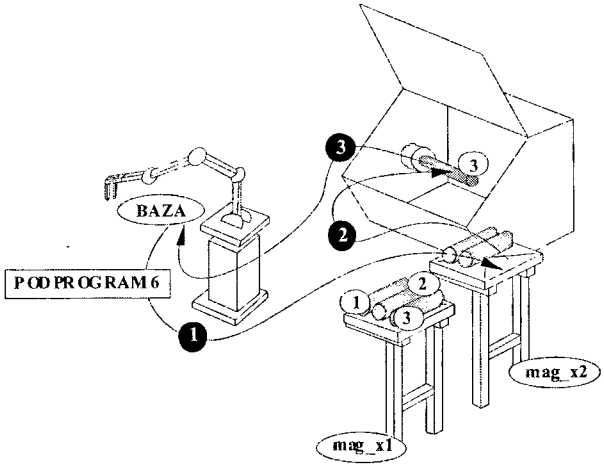
Robot wysyła dwa sygnały do frezarki: *otwórz uchwyt* i *zamknij uchwyt*, natomiast frezarka wysyła do robota cztery sygnały: *uchwyt otwarty*, *uchwyt zamknięty*, *obrabiarka wolna* (koniec obróbki), *obrabiarka zajęta*.

Z powodu ograniczonej liczby wyjść obrabiarki sygnały *obrabiarka wolna*, *obrabiarka zajęta* są wysyłane za pomocą czujnika dotykowego znajdującego się w przestrzeni roboczej oznaczonego jako *SW*. Przycisk jest uruchamiany przez narzędzie osadzone we wrzecionie frezarki. Zaletą takiego rozwiązania jest pewność, że nie dojdzie do konfliktu między wrzecionem obrabiarki, a ramieniem robota obsługującego (czujnik dotykowy *SW* umieszczony z dala od przestrzeni roboczej robota). Logika działania zastosowanego przycisku przebiega następująco: frezarka wolna – *sygnał 1*, frezarka zajęta – *sygnał 0*. Takie odwrócenie logiki jest wymuszone kwestiami bezpieczeństwa. W przypadku zakłóceń w komunikacji, takich jak np. przerwa w dopływie informacji sterujących lub uszkodzony przewód, układ sterowania robota odczytuje jako *obrabiarka zajęta* i nie wprowadza robota w przestrzeń roboczą frezarki.

W podsystemie tokarskim sygnał *obrabiarka wolna* jest wysyłany do robota przez układ sterowania po zakończeniu programu obróbki.

Do sterowania pracą robota przegubowego opracowano podprogramy sterujące poszczególnymi cyklami przemieszczeń. Zakładając, że na palecie znajdują się 3 przedmioty dla każdego robota opracowano 3 podprogramy sterujące. Każdy podprogram realizuje przemieszczenie przedmiotu w relacji paleta – obrabiarka – paleta. Podprogram zaczyna się od pobrania przedmiotu z palety i włożeniu do uchwytu obrabiarki. Zgodnie z przedstawionym algorytmem podprogram zostaje przerwany po zamknięciu uchwytu przedmiotowego w obrabiarce (i wycofaniu robota w bezpieczną pozycję). Kontynuacja podprogramu następuje po wykonaniu obróbki i obrabiany przedmiot zostaje odłożony na paletę.

Zaimplementowane podprogramy sterujące robotem, są identyczne dla podsystemu technologicznego tokarskiego i frezarskiego, co przedstawiono na rys. 8.



Rys. 8. Podprogramy podsystemów technologicznych

Wszystkie podprogramy sterujące pracą układarki regałowej, wózka transportowego i robota przegubowego zostały opracowane, skompilowane oraz przetestowane w języku Borland Delphi 6, przy użyciu modułu DLL Wizard. Podprogram zbudowany jest w postaci biblioteki doładowywanej dynamicznie z rozszerzeniem dll (ang. *dynamic link library*). Budowa podprogramów została zamieszczona w publikacjach [1][2].

6. PODSUMOWANIE

Opracowane oprogramowanie umożliwia sterowanie pracą systemu na podstawie harmonogramów produkcji, które wcześniej mogą być tworzone przy udziale różnych metod np. w oprogramowaniach symulacyjnych.

Dzięki otwartej architekturze programowej oraz użyciu komponentów obsługujących technologię internetową możliwe stało się swobodne kształtowanie dowolnej architektury sterowania.

Każdy podsystem funkcjonalny sterowany jest za pomocą oddzielnego komputera, a poszczególne funkcje sterujące zapisane są w postaci podprogramów, które wywoływane są zgodnie z zadanym harmonogramem pracy.

Opracowany główny program sterujący przy udziale podprogramów dynamicznych, umożliwia programową rozbudowę systemu bez przerabiania głównego algorytmu sterowania systemem.

LITERATURA

[1] HONCZARENKO J., SOSNOWSKI M., *Zastosowanie modułów mikroprocesorowych do sterowania badawczym elastycznym systemem wytwarzania,* POSTĘPY ROBOTYKI, tom 2, praca zbiorowa pod redakcją Krzysztofa Tchonia, Wydawnictwo Komunikacji i Łączności, Warszawa 2005, str. 87-94.

[2] HONCZARENKO J., SOSNOWSKI M., *O możliwości wykorzystania logiki rozmytej do sterowania elastycznym systemem wytwarzania,* AUTOMATION 2005, Konferencja Naukowo-Techniczna AUTOMATYZACJA - NOWIŚCI I PERSPEKTYWY, Warszawa 2005, str.86-95.

[3] METZGER P., *Automation 2004. Architektura komputerów zgodna z IBM PC.* wydanie VII, Wydawnictwo HELION, Gliwice 2002.

[4] PACHECO X.,TEIXEIRA S., *Delphi 6, Vademecum profesjonalisty*, tom 1, Wydawnictwo HELION, Gliwice 2002.