ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1988
Seria:AUTOMATYKA z.94 Nr kol.970

Wojciech GOMÓŁKA
Politechnika Śląska

WYKORZYSTANIE SIECI GRAFCET W AUTOMATYZACJI PROCESÓW DYSKRETNYCH I ROBOTYCE

Streszczenie. W pracy przedstawiono zasady tworzenia opisu funkcjonowania systemu dyskretnego za pomocą sieci działań GRAFCET. Przedstawiono także zasady wykorzystania sieci do syntezy sterownika dla takiego procesu oraz generowania oprogramowania dla programowalnego sterownika binarnego.

1. Wstęp

Posiadanie nawet najbardziej skomplikowanego sprzętu nie wystarcza do prawidłowego rozwiązania zadania automatyzacji procesu przemysłowego. Aby zadanie takie było jak najlepiej rozwiązane, należy przede wszystkiem :

- wiedzieć, jak proces przemysłowy jest zorganizowany, jak funkcjonuje i jakie funkcje są przypisane każdemu elementowi procesu;
- określić i przeanalizować struktury i algorytmy sterowania ;
- określić i dobrać zbiór środków technicznych , które umożliwią praktyczną realizację zadań sterowania.

Realizacja dwóch ostatnich etapów jest niemożliwa bez prawidłowej realizacji etapu pierwszego , w wielu przypadkach realizowanego poprawnie wyłącznie przez specjalistę "procesowca". Z drugiej strony , pojęcia używane przez specjalistę "automatyka" dla zdefiniowania zadań algorytmów sterowania są często trudne do zrozumienia przez "procesowca" . Prowadzi to więc często do sytuacji, w których obydwoje muszą poświęcić sporo czasu na rozpoznanie zasad funkcjonowania procesu przemysłowego oraz systemu sterowania tym procesem. Nie jest to oczywiście problemem dla przypadku prostych procesów przemysłowych oraz dla standardowych zadań sterowania . Staje się to jednak kłopotliwe , gdy proces przemysłowy jest bardziej złożony, a zadania sterowania odbiegają od zadań typowych takich jak "stabilizacja" ,"nadążanie",.... Dotyczy to zwłaszcza systemów , które mają być sterowane za pomocą układów logiki programowanej (np.elastyczne systemy produkcji,roboty przemysłowe,magazyny wysokiego składowania itp.) lub też zadań sterowania w sytuacjach nietypowych (sytuacje awaryjne , rozruchy ,odstawianie ....).

Dlatego też w drugiej połowie lat siedemdziesiątych wiele

organizacji rządowych w krajach uprzemysłowionych przystąpiło do prac nad standaryzacją opisu zadań sterowania oraz opisu funkcjonowania procesów przemysłowych do realizacji tych zadań. Między innymi, wynikiem prac prowadzonych we Francji przez takie organizacje jak AFCET, ADEPA oraz UTE było pojawienie się nowej formy opisu dyskretnych procesów sekwencyjnych dla celów sterowania nazwanej GRAFCET (fr. Graphe de Commande Etape-Transition) zatwierdzonej w roku 1982 normą francuską NF-C-03-190.

GRAFCET jest formalnym sposobem opisu i analizy funkcjonowania systemu przydatnym zwłaszcza tam, gdzie działanie systemu jest działaniem sekwencyjnym i może być zdekomponowane na pewną liczbę etapów podstawowych. Zasadnicze zalety opisu typu GRAFCET są następujące:

- jest niezależny od sposobu technicznej realizacji zadania sterowania;
- umożliwia racjonalny wybór zmiennych stanu (ich liczba nie musi być minimalna);
- umożliwia zapisanie w sposób jednoznaczny zasad funkcjonowania systemu zarówno bez sterowania, jak i przy włączonym sterowaniu;
- umożliwia synchronizację zadań realizowanych równolegle lub też zadań związanych z wyborem określonej sekwencji działań;
- jest wygodny do opisu systemów o dużej liczbie zmiennych wejściowych, wyjściowych i zmiennych stanu;
- skraca i upraszcza w znacznym stopniu całość prac związanych z analizą i syntezą układów i algorytmów sterowania.

W pracy przedstawione zostaną podstawowe zasady tworzenia opisu systemu za pomocą sieci GRAFCET oraz przedstawione zostaną zasady syntezy sterownika lub programu algorytmu sterowania dla procesu opisanego tą siecią.

2. Pojęcia podstawowe

GRAFCET jest siecią zdefiniowaną przez czwórkę uporządkowaną:

$$\langle E, T, A, M_o \rangle$$

gdzie:

1) $E = \langle E_1, E_2, \ldots E_m \rangle$ jest skończonym zbiorem etapów.

Etap odpowiada pewnemu charakterystycznemu zachowaniu się systemu (całości lub jakiejś jego części), niezmiennemu w odniesieniu do jego wejść i wyjść. Gdy zachowanie to odpowiada definicji danego etapu, to mówi się, że etap ten jest etapem aktywnym. W danej chwili można mieć do czynienia z większą liczbą etapów aktywnych. Aktywność etapów może

ulegać zmianie wg reguł ewolucyjnych omówionych w dalszej części pracy. Etap przedstawiany jest za pomocą kwadratu z umieszczonym wewnątrz numerem tego etapu. Jeżeli etap jest aktywny, zaznacza się to za pomocą kropki w danym kwadracie. Etap początkowy (inicjujący) jest oznaczany za pomocą kwadratu o podwójnej linii.

Etapowi przyporządkowane mogą być pewne operacje . Operacją nazywa się zadanie lub zbiór zadań realizowanych na danym etapie, gdy dany etap jest etapem aktywnym . Operacje powiązane z danym etapem są przedstawiane w postaci prostokątów połączonych poziomo z kwadratem odpowiadającym danemu etapowi. Opis operacji (w postaci słownej lub symbolicznej) umieszczony musi być w prostokącie odpowiadającym danej operacji .

2) - $T = (t_1, t_2, \ldots t_k)$ jest skończonym zbiorem przejść . Przejście (tranzycja) wskazuje możliwości warunkowej zmiany aktywności etapów (np. przejścia z etapu na etap). Jest ono przedstawiane za pomocą pogrubionej linii poziomej umieszczonej na odpowiednim połączeniu pomiędzy etapami . Gdy przejście jest numerowane, numer ten jest umieszczony w nawiasach okrągłych po lewej stronie danego przejścia . Pomiędzy kolejnymi etapami może występować tylko jedno przejście.

Każdemu przejściu jest przypisany pewien warunek logiczny zwany warunkiem przejścia . Warunki te są predykatami opisanymi za pomocą zmiennych wejściowych , wyjściowych systemu oraz zmiennych reprezentujących stan niektórych etapów . Spełnienie predykatu jest warunkiem koniecznym przy realizacji danego przejścia. Predykat dla warunku przejścia jest wpisywany na prawo od linii odpowiadającej danemu przejściu.

3) - $A = (a_1, a_2, \ldots, a_k)$ jest skończonym zbiorem połączeń skierowanych wskazujących kierunki oraz możliwości przejścia z etapu na etap . Dzielą się one na:

- połączenia proste , wskazujące możliwość przechodzenia z jednego etapu na drugi ; są one przedstawiane za pomocą linii pionowych i wskazują normalny kierunek zmian (od góry do dołu),
- połączenia wielokrotne , wskazujące możliwość przechodzenia z jednego etapu na wiele innych , lub z wielu etapów na jeden .

Połączenia te dzielą się z kolei na :

a) połączenia wyboru (dysjunkcyjne) , wskazujące możliwość przechodzenia z danego etapu na jeden z wielu ; są one przedstawiane za pomocą pojedynczej linii poziomej ,

b) *połączenia jednoczesnej realizacji (koniunkcyjne)*, wskazujące możliwość przechodzenia z danego etapu na wiele innych, realizowanych równolegle. Są one przedstawiane za pomocą podwójnej linii poziomej.

4) - $M = (m_1, m_2, \ldots, m_m)$ jest zbiorem zmiennych boolowskich takich, że $m_i = 1$, oznacza, iż etap "i" jest etapem aktywnym. Zbiór M_o oznacza zbiór znaczników inicjujących, reprezentujących etapy aktywne w stanie początkowym.

3. Podstawowe reguły konstrukcji opisu systemu za pomocą sieci GRAFCET.

Stworzenie poprawnego opisu systemu za pomocą sieci GRAFCET wymaga przestrzegania pewnej liczby reguł konstrukcji takiej sieci. Reguły te reprezentowane są przez zbiór tzw. reguł syntaktycznych oraz przez sześć reguł ewolucyjnych.

3.1. Reguły syntaktyczne związane z etapem i operacją.

- Dwa etapy nie mogą następować bezpośrednio po sobie. Muszą być one zawsze rozdzielone przejściem;
- etapy muszą uwzględniać wszystkie zmiany w zachowaniu się systemu;
- każdemu etapowi należy przypisać absolutnie wszystkie operacje realizowane na tym etapie;
- liczba etapów jest nieograniczona.

3.2. Reguły syntaktyczne związane z przejściem i warunkami przejścia.

- Dwa przejścia (tranzycje) nie mogą występować bezpośrednio po sobie. Muszą być zawsze rozdzielone etapem;
- ze względu na możliwość wystąpienia połączeń wielokrotnych liczba przejść może być różna od liczby etapów;
- każdemu przejściu można przyporządkować tylko jeden warunek;
- każdy warunek winien zawierać całą informację logiczną związaną z możliwościami realizacji danego przejścia;
- każdy warunek przejścia może zawierać dowolną liczbę operatorów logicznych;
- ten sam warunek przejścia może być przypisany wielu przejściom (tranzycjom) w sieci GRAFCET;
- nie można dawać identycznych warunków przejścia w dwóch bezpośrednio po sobie następujących przejściach; sytuacja taka powoduje bowiem przeskok etapu znajdującego się pomiędzy tymi przejściami;
- nie można nigdy uwzględniać w warunkach przejścia informacji związanej z operacją, która nie występuje w danej sieci GRAFCET. Nieobecność takiej operacji może bowiem doprowadzić do blokady

realizacji etapów związanej z oczekiwaniem na informację o stanie "nieobecnej" operacji .

3.3. Reguły syntaktyczne związane z połączeniami skierowanymi .

- Połączenia pomiędzy etapami powinny być przedstawiane w postaci linii pionowych . Normalny kierunek realizacji etapów jest zatem "z góry na dół" , a wszelkie odstępstwa od tej reguły winny być ściśle opisane ;
- należy unikać krzyżowania się połączeń ;
- połączenia wielokrotne koniunkcyjne muszą rozpoczynać się od jednego wspólnego warunku przejścia . Także zakończenie takich połączeń musi być związane z jednym wspólnym warunkiem przejścia (może to być np. iloczyn logiczny warunków dla poszczególnych przejść dla końcowych etapów każdej gałęzi połączenia);
- w przypadku połączeń wielokrotnych dysjunkcyjnych każdy łuk połączenia musi rozpoczynać się i kończyć osobnym warunkiem przejścia .

3.4. Reguły ewolucyjne dla sieci GRAFCET.

W czasie tworzenia i analizy sieci GRAFCET należy zawsze uwzględniać następujące reguły ewolucyjne :

Reguła 1. Inicjalizacja.

Na początku realizacji zadania przynajmniej jeden z etapów musi być aktywny bez żadnych warunków wstępnych . Etap taki nazywany jest etapem początkowym (inicjującym) . Z reguły etap ten jest etapem "spoczynkowym" i nie przypisuje mu się żadnej operacji . Sieć GRAFCET może zwierać większą liczbę takich etapów .

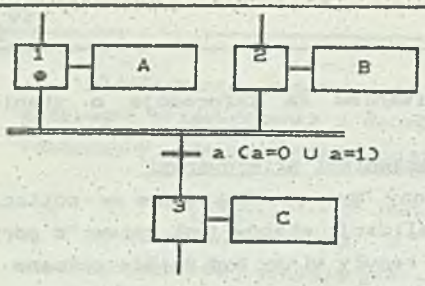
Reguła 2. Realizacja przejścia.

Dane przejście pomiędzy etapami jest "przygotowane", gdy wszystkie poprzedzające je etapy są aktywne. Przejście może być "zrealizowane", gdy jest przygotowane i gdy przypisany mu warunek przejścia jest prawdziwy (tzn. równy jedynce logicznej). Wówczas też realizacja przejścia jest natychmiastowa i obligatoryjna .

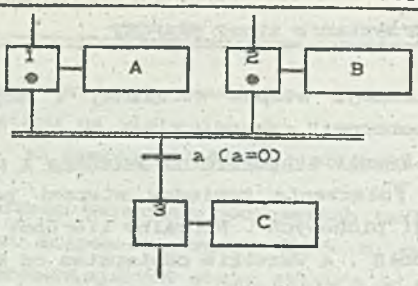
Reguła 3. Konsekwencje realizacji przejścia.

Realizacja przejścia powoduje natychmiast :
- aktywację wszystkich następujących po nim etapów ;
- dezaktywację wszystkich etapów poprzedzających to przejście

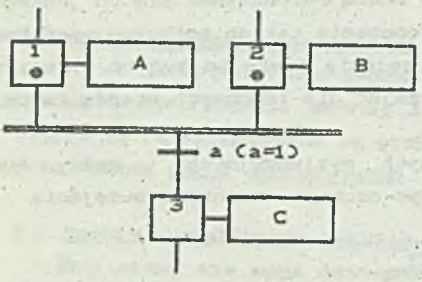
Przykład : Interpretację reguł 2 i 3 pokazano na rys.1 . Na rys.1a przejście nie jest przygotowane, gdyż etap 2 jest nieaktywny. Z kolei przejście na rys.1b jest przygotowane, gdyż oba etapy 1 i 2 są aktywne. Przejście na rys.1c może być realizowane, gdyż etapy 1,2 są aktywne, zaś warunek przejścia jest prawdziwy. Wreszcie na rys.1d pokazano efekt realizacji przejścia , tzn. etapy 1,2 przestały być aktywne ,



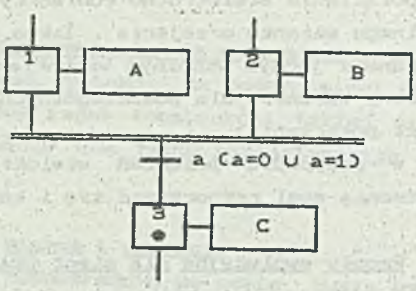
Rys.1a.



Rys.1b



Rys.1c



Rys.1d

Rys.1. Interpretacja reguł ewolucyjnych 2 i 3
Fig.1. Interpretation of evolution rules 2 and 3

Reguła 4. Jednoczesna realizacja przejść.

Jeżeli spełnione są warunki dla jednoczesnej realizacji wielu przejść , przejścia te muszą być zrealizowane .Sytuacja taka może pojawić się, gdy kilka przejść jest związanych z tymi samymi warunkami przejścia , lub gdy w kilku z nich znajduje się taka sama informacja . Reguła ta jest wykorzystywana przy dekompozycji sieci GRAFCET na kilka współzależnych sieci GRAFCET .

Reguła 5. Priorytet aktywacji etapów.

Jeżeli jakiś etap może być w danym momencie aktywny lub nieaktywny , zawsze staje się on aktywnym .

Reguła 6. Konsekwencje aktywacji etapu.

Aktywacja etapu pociąga za sobą realizację wszystkich bez wyjątku operacji związanych z danym etapem .Dezaktywacja etapu powoduje z kolei przerwanie wszystkich operacji przypisanych danemu etapowi .

4. Podstawowe rodzaje operacji w sieci GRAFCET.

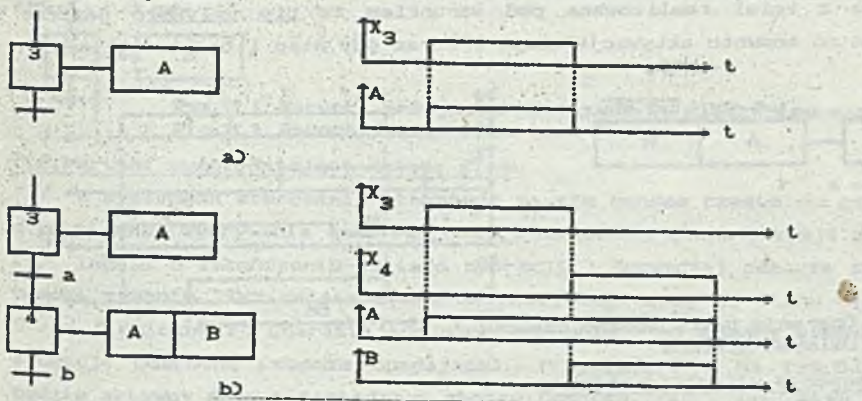
Analiza operacji wykorzystywanych w sieci GRAFCET wymaga określenia wzajemnych zależności pomiędzy stanem aktywności etapu a

warunkami realizacji przyporządkowanych mu operacji . Zazwyczaj stan każdego etapu opisany jest przez zmienną logiczną X_i równą 1, gdy etap jest aktywny , oraz równą 0 w przypadku przeciwnym . Zależności pomiędzy sposobami realizacji operacji dla danego etapu aktywnego wymagają zatem zdefiniowania podstawowych typów operacji , które mogą być na danym etapie realizowane .

4.1. Operacja ciągła.

Jest to operacja , która jest realizowana tak długo, jak długo dany etap jest aktywny . Przykładowo, na rys. 2a operacja A związana z etapem 3 jest realizowana tak długo, jak długo etap ten jest aktywny.

Szczególnym przypadkiem operacji ciągłej jest operacja z podtrzymaniem , dla której realizacja jest rozłożona na kilka etapów. Wówczas rozkaz realizacji danej operacji musi być powtarzany na wszystkich etapach zawiązanych z tą operacją . W praktyce przemysłowej opracje takie występują, gdy ma się do czynienia z monostabilnymi urządzeniami wykonawczymi . Przykład operacji cigłej z podtrzymaniem pokazano na rys. 2b.



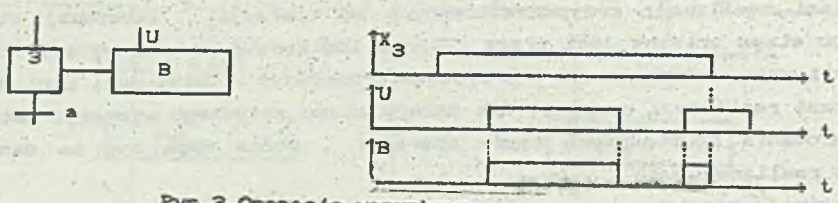
Rys. 2. Operacje ciagłe
Fig. 2. Continuous operations

4.2. Operacja ciągła - warunkowa.

Operacja warunkowa jest operacją ciągłą , której realizacja jest uwarunkowana spełnieniem pewnego warunku logicznego U .

"Operacja" = X_i and U

Warunek logiczny jest oznaczany za pomocą wskaźnika w górnej części prostokąta opisującego daną operację . Tego typu operacje umożliwiają proste przedstawianie warunków zabezpieczających poprawność realizacji związanych np. ze sterowaniem procesów. Przykładowo , operacja B z rys. 3 będzie realizowana , gdy etap 3 jest aktywny i gdy warunek logiczny U będzie spełniony .

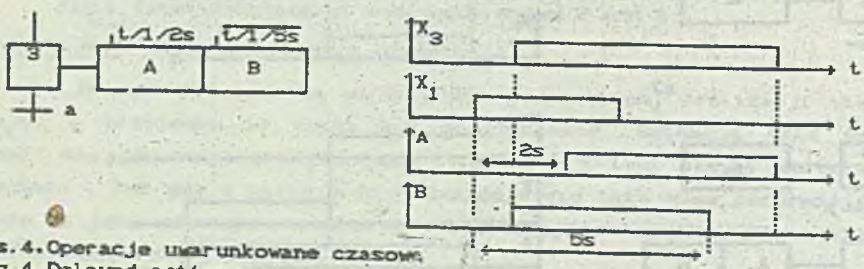


Rys.3.Operacje warunkowe
Fig.3.Conditional operations

4.3.Operacje uwarunkowane czasowo.

Operacje uwarunkowane czasowo są to operacje warunkowe, dla których warunkiem logicznym jest czas . Warunek taki zaznacza się symbolem : t/i/q (sek), gdzie i jest numerem etapu zawierającego operację zliczania czasu , zaś q jest odcinkiem czasu, jaki musi upłynąć od momentu aktywacji etapu "i" .

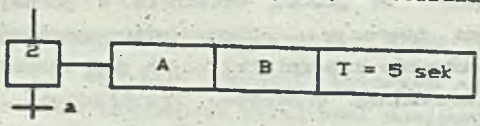
Przykładowo, na rys.4 przedstawiono dwie operacje A i B . Operacja A będzie zrealizowana pod warunkiem, że upłyną 2 sekundy od momentu aktywacji etapu 1 oraz gdy etap 3 będzie aktywny ; zaś operacja B będzie z kolei realizowana pod warunkiem, że nie upłynęło jeszcze 5 sekund od momentu aktywacji etapu 1 (oraz gdy etap 1 będzie aktywny) .



Rys.4.Operacje uwarunkowane czasowo
Fig.4.Delayed actions

4.4.Operacje zliczania czasu .

Pomiędzy różnymi operacjami przypisanymi dla danego etapu może być także operacja zliczania czasu . Operację taką zaznacza się za pomocą zapisu " T = q (sek) " wewnątrz prostokąta oznaczającego operację (q oznacza wówczas odcinek czasu do zliczania). Przykładowo, na rys.5 z etapem 3 związane są trzy operacje A,B oraz odliczania odcinka czasu 5 sekund (dokładnie rozpoczęcie odliczania odcinka czasu 5 sekund).



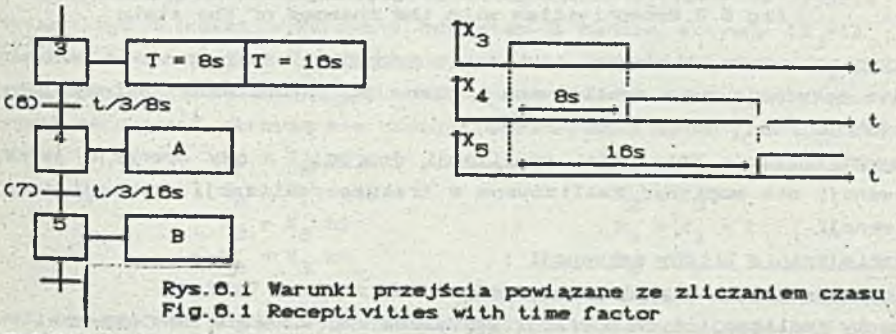
Rys.5.Operacja zliczania czasu
Fig.5.Action of time counting

5. Szczególne przypadki warunków przejścia .

Warunki logiczne odpowiadające przejściu z etapu na etap mogą czasami przyjmować specyficzną postać wynikającą z przebiegu procesu . Są to najczęściej warunki związane ze zliczaniem czasu lub warunki uwzględniające zmiany stanu różnych wielkości .

5.1. Warunki związane z odliczaniem czasu.

Opis warunku związanego z odliczaniem czasu jest taki sam jak dla operacji uwarunkowanych czasowo . Przykładowo , dwie operacje inicjacji odliczania czasu przypisane są do etapu 3 . Warunek przejścia (6) będzie prawdziwy po 8 sekundach od momentu aktywacji etapu 3 , zaś warunek (8) po 16 sekundach .



Rys.6.1 Warunki przejścia powiązane ze zliczaniem czasu
Fig.6.1 Receptivities with time factor

5.2. Warunki uwzględniające zmianę stanu.

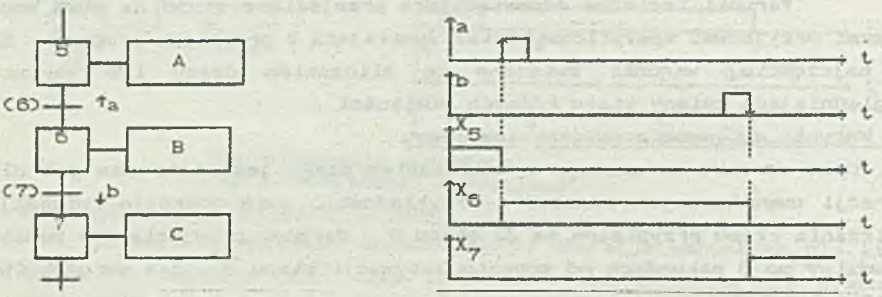
W systemach sterowania binarnego ma się bardzo często do czynienia z problemami wykrywania oraz reagowania na zmiany stanu jakiejś zmiennej (np. impuls o zakończeniu jakiejś operacji). Zazwyczaj oznacza się za pomocą symbolu "↑a" zmianę stanu zmiennej logicznej ze stanu "logiczne 0" na "logiczne 1" (narastające zbocze); oraz za pomocą symbolu "↓a" sytuację odwrotną (zbocze opadające). Przykładowo , na rys.6.2 etap 6 będzie aktywny przy narastającym zboczu impulsu "a" , zaś etap 7 przy zboczu opadającym impulsu "b" .

6. Modyfikacje sieci GRAFCET.

Dla opisu zadania sterowania można skonstruować wiele różnych sieci GRAFCET opisujących dokładnie jedno i to samo zadanie. Istnieją jednak przypadki, gdy jedna struktura może być bardziej wygodna i bardziej czytelna od innych . Wymaga to jednak wykonania pewnych operacji, które mogą być realizowane na strukturach sieci .

Do operacji tych zaliczyć można :

- synchronizacje sekwencji ,gdy w dwóch różnych sekwencjach chce się zapewnić jednoczesną aktywację etapu ;



Rys. 6.2. Warunki przejścia uwzględniające zmiany stanu
Fig. 6.2. Receptivities with the changes of the state

- synchronizacje kolejności realizacji sekwencji . gdy wymagane jest, aby jedna sekwencja była realizowana dopiero po zakończeniu jakiejś innej sekwencji ;
- synchronizacje kolejności realizacji operacji . gdy operacje jednej sekwencji nie mogą być realizowane w trakcie realizacji operacji innej sekwencji ;
- zmniejszanie liczby sekwencji ;
- redukcje liczby etapów w sekwencji , itp..

Sposoby realizacji tych operacji można znaleźć w normie NF-C-03-190 .

7. Wykorzystanie sieci GRAFCET do syntezy sterownika.

Istnieją dwie metody bezpośredniego wykorzystania sieci GRAFCET do syntezy sterownika sekwencyjnego :

- w oparciu o przerzutniki RS ;
- w oparciu o liczniki programowane .

W niniejszej pracy omówiona zostanie pierwsza metoda. Wykorzystuje się w niej fakt, iż zmienna logiczna określająca stan etapu (X = 1, gdy etap aktywny lub X = 0, gdy etap nie jest aktywny) może być przyporządkowana wyjściu Q przerzutnika RS . Synteza sterownika sprowadza się zatem do określenia wyjść przerzutników RS oraz do utworzenia pewnej sieci kombinacyjnej odpowiadającej wyjściom-realizowanym operacjom .

Rozważmy metodę syntezy sterownika w oparciu o następujący przykład.

Przykład

Należy zaprojektować sterownik dla wiertarki pokazanej na rys. 7a. Zakładamy , że wiertło kręci się ze stałą prędkością. Wiertarka zaopatrzona jest w układ napędowy E sterowany czterema sygnałami:
-sygnałem D oznaczającym opuszczanie głowicy wiertarki;
-sygnałem M oznaczającym podnoszenie głowicy wiertarki;
-sygnałem PV oznaczającym powolny ruch głowicy (w górę lub w dół);
-sygnałem GV oznaczającym szybki ruch głowicy.
Układ wiertarki zaopatrzony jest także w trzy wyłączniki krańcowe

b0,b1,b2 dla zlokalizowania położenia głowicy.
Cykl realizacji operacji wiercenia jest następujący. Rozpoczyna się on przez naciśnięcie przycisku monostabilnego "m" .Następnie od położenia b0 do b1 głowica opuszczana jest z dużą prędkością , zaś od położenia b1 do b2 z prędkością małą .Po dotarciu do położenia b2 powinien nastąpić szybki ruch głowicy w górę do położenia b0 i w tym położeniu układ czeka na następne przyciśnięcie przycisku "m" .

Sieć GRAFCET dla tak określonego zadania pokazano na rys.7b. Zawiera ona 4 etapy , a więc sterownik będzie się składał z czterech przerzutników RS. Przykładowo, dla etapu 2 , wyjście X_2 przerzutnika drugiego ma być równe 1, gdy $X_1=1$ i gdy $m.b0=1$. Zatem wejście ustawiające S_2 winno być sterowane iloczynem $X_1.m.b0$. Wyjście X_2 powinno być z kolei wysterowane, gdy etap 3 będzie aktywny ($X_3=1$) ; a zatem wejście zerujące R_2 będzie sterowane sygnałem logicznym X_3 .
Postępując podobnie dla pozostałych przerzutników , oraz uwzględniając sygnał zerujący I otrzyma się następujące równania wejść przerzutników:

$$S_1 = X_4.b0 + I \qquad R_1 = X_2$$
$$S_2 = X_1.m.b0 \qquad R_2 = X_3$$
$$S_3 = X_2.b1 \qquad R_3 = X_4 + I$$
$$S_4 = X_3.b2 \qquad R_4 = X_1 + I$$

Należy jeszcze określić wyjścia sterownika odpowiadające realizowanym operacjom (sygnałom sterującym). I tak np. sygnał D winien być równy 1, gdy etap 2 lub 3 są aktywne ; czyli :

$$D = X_2 + X_3$$

Końcowy schemat sterownika pokazano zatem na rys.7c.

Reguły syntezy sterownika za pomocą przerzutników RS można zatem określić następująco:

1. Niech r_{i-1} jest warunkiem przejścia dla etapu i . Ustawienie wyjścia X_i na 1 jest realizowane przez równanie logiczne dla wejścia ustawiającego S_i :

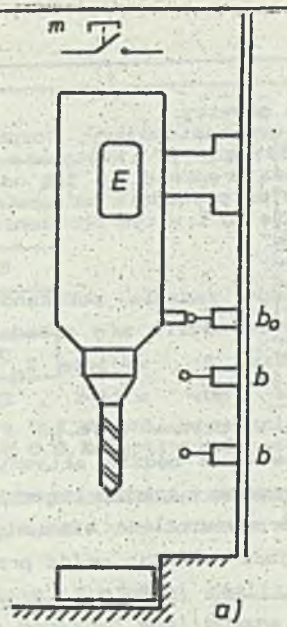
$$S_i = X_{i-1} \cdot r_{i-1}$$

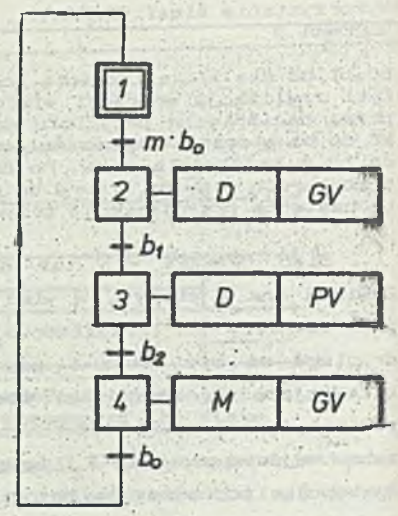
2. Ustawienie wyjścia X_i na zero logiczne jest realizowane przez równanie logiczne dla wejścia zerującego R_i:

$$R_i = X_{i+1} + I$$

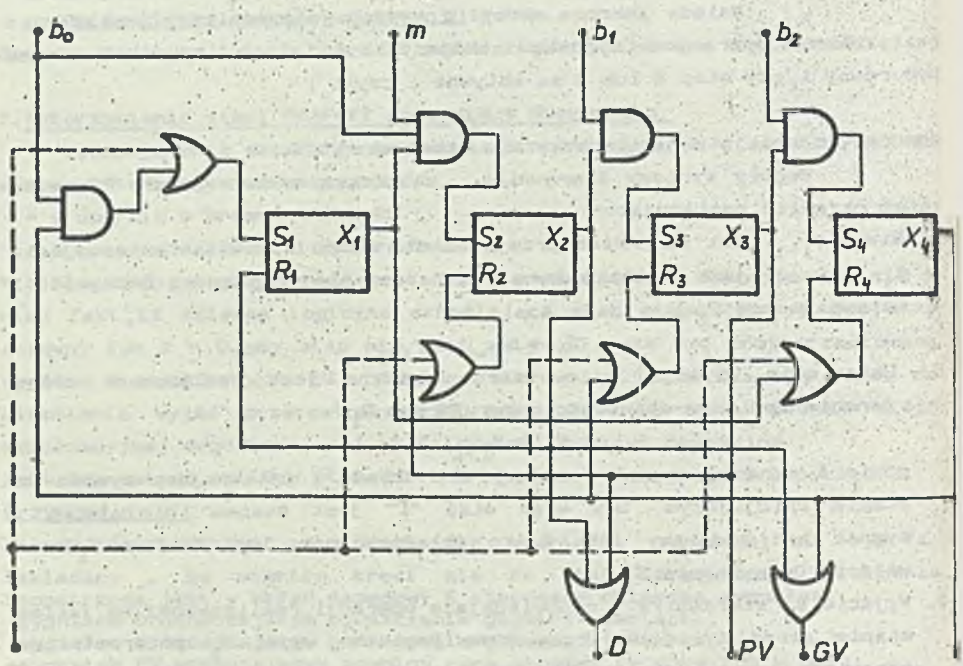
gdzie I oznacza sygnał zerujący dla układu , ustawiający system na etapie inicjującym .Gdy więc etap "i" jest etapem inicjującym , sygnał ten podawany jest (jako składnik sumy logicznej z S_i) na wejście ustawiające S_i.

3. Wyjście A_j sterownika , odpowiadające operacji realizowanej na jakimś etapie, określone jest przez sumę logiczną wyjść X_k przerzutników odpowiadających etapom, na których operacja A_j ma być realizowana.



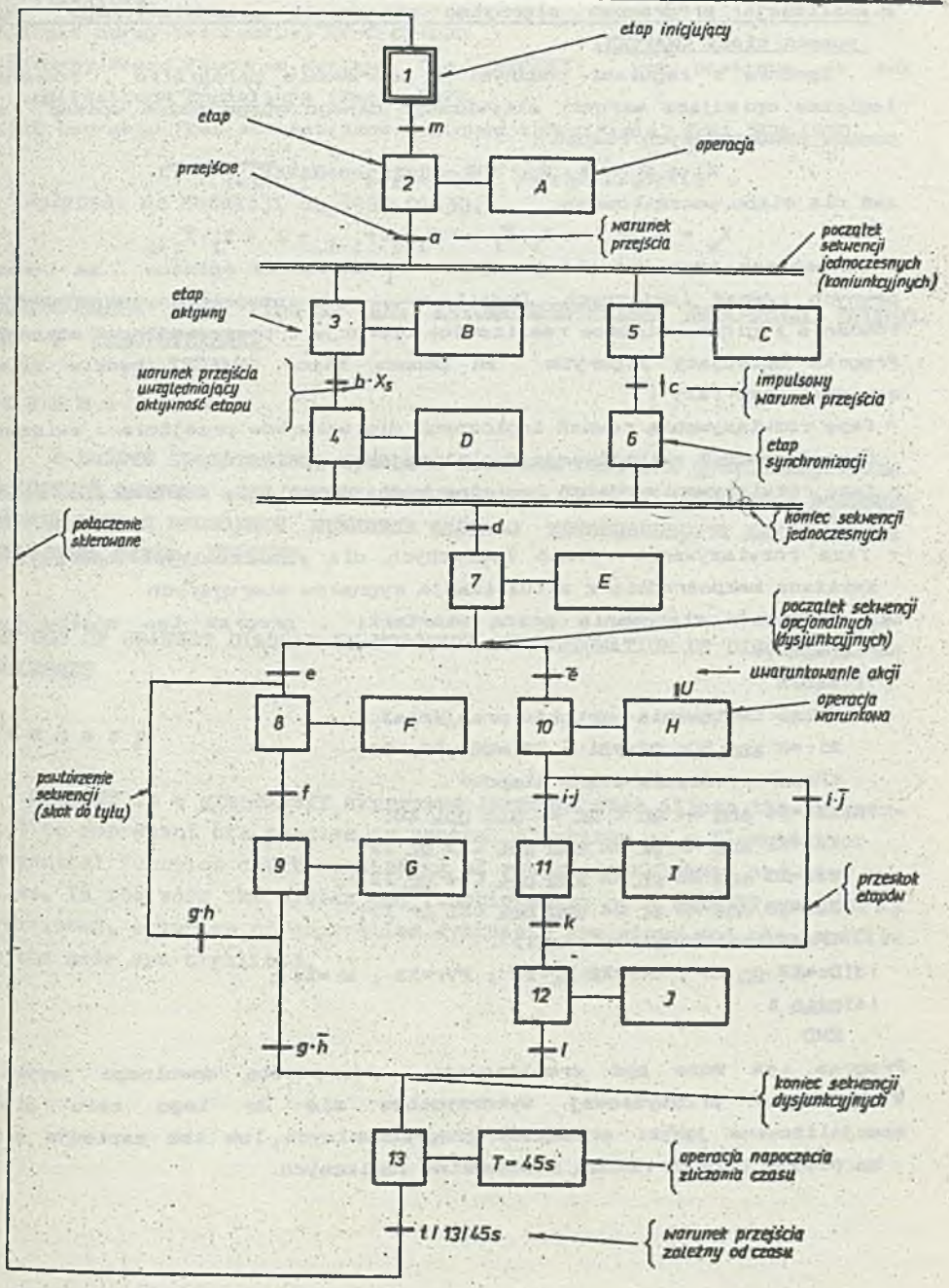


c)



Rys. 7. Przykład syntezy sterownika (c) dla wiertarki (a) w oparciu o GRAFCET (b)

Fig. 7. An example of controller (c) synthesis for machine (a) on the basis of GRAFCET (b)



Rys. 8. Podstawowe określenia spotykane w sieci GRAFCET
Fig. 8. Basic notions for GRAFCET network

8. Realizacja programowa algorytmu sterowania przedstawionego za pomocą sieci GRAFCET.

Zgodnie z regułami podanymi w poprzednim paragrafie , równania logiczne opisujące warunki aktywizacji danego etapu można opisać za pomocą następujących równań :

$$X_i = S_i + X_i \cdot \overline{R}_i = X_{i-1} \cdot r_{i-1} + X_i \cdot (\overline{X_{i+1} + I})$$

zaś dla etapu początkowego:

$$X_i = S_i + I + X_i \cdot \overline{R}_i = X_{i-1} \cdot r_{i-1} + I + X_i \cdot \overline{X}_{i+1}$$

W równaniach tych warunki przejścia r_i są także opisane za pomocą pewnych równań logicznych. Dodatkowo , opis uzupełniony musi być o równania logiczne wiążące realizowane operacje z poszczególnymi etapami. Program opisujący algorytm za pomocą sieci GRAFCET będzie zatem zawierał trzy fazy :

- fazę rozwiązywania równań logicznych dla warunków przejścia , związaną bezpośrednio z aktualizacją danych wejściowych ;
- fazę rozwiązywania równań logicznych dla określenia stanu etapów sieci GRAFCET ;
- fazę rozwiązywania równań logicznych dla realizowanych operacji , związaną bezpośrednio z aktualizacją sygnałów sterujących .

Dla przykładu sterowania pracą wiertarki , program ten mógłby być następujący:

```
!1!BEGIN
    <Faza testowania warunków przejścia>
    R1:=M and B0; R2:=B1 ; R3:=B2 ;R3:=B0;
    <Faza określania stanu etapów>
!2!X1:=X4 and R4 or I or X1 and not X2;
    X2:=X1 and R1 or X2 and not (X3 or I);
    X3:=X2 and R2 or X3 and not (X4 or I);
    X4:=X3 and R3 or X4 and not (X1 or I);
    <Faza realizacji operacji>
!3!D:=X2 or X3 ; GV:=X2 or X4 ; PV:=X3 ; M:=X4 ;
!4!goto 1
    END
```

Program ten może być zrealizowany za pomocą dowolnego języka. W praktyce przemysłowej wykorzystuje się do tego celu albo specjalizowane języki automatów programowalnych, lub też zapisuje się za pomocą języków równań i schematów logicznych.

LITERATURA.

[1]Opis normy francuskiej NF-C-03-190.

[2]Bossy,Brard,Faugeres,Merlaud "Le GRAFCET , sa pratique et ses applications"Educalivre ,Paris 1979.

[3]C.Laurgeau "Les automatismes logiques industriels" Edit.SCM 1979.

Recenzent:Doc.dr hab.inż.T.Sawik

Wpłynęło do Redakcji do 1988-04-30.

ИСПОЛЬЗОВАНИЕ СЕТИ ГРАФСЕТ ДЛЯ АВТОМАТИЗИРОВАНИЯ ДИСКРЕТНЫХ ПРОЦЕССОВ И РАБОТОТЕХНИКИ

Р е з ю м е

В работе представлены принципы создания описания функционирования дискретной системы при помощи сети действий ГРАФСЕТ. Приведены основные правила сети и оговорены принципы синтеза командаппарата для процесса описанного сетью ГРАФСЕТ.

THE USE OF GRAFSET DIAGRAM IN ROBOTICS AND AUTOMATION OF DISCRETE PROCESSES

S u m m a r y

GRAFSET is a graphicaly expressed language that allows the programmist to represent his process or control algorithm as a flowchart or sequential function chart consisting of steps, transitions and flow lines. In the work the syntax and functionality of a GRAFSET program is overviewed, examples of controller synthesis are given and the benefits to the user are highlited.