ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1982

Seria: INFORMATYKA z. 3 Nr kol. 706

Eugeniusz KORBIEL

Instytut Informatyki
Czasu Rzeczywistego
Politechniki Śląskiej

METODA SYNTEZY MINIMALNYCH, WIELOWYJŚCIOWYCH SIECI KOMBINACYJNYCH NA BAZIE DEKODERA

Streszczenie: W artykule przedstawiono metodę syntezy wielowyjściowych sieci kobminacyjnych na bazie dekodera, w oparciu o zdefiniowane D-implikanty.

Proponowaną metodę porównano na przykładach literaturowych z metodami klasycznymi, uzyskując lepsze rozwiązania zarówno ze względu na szybkość projektowania, jak i liczbę połączeń oraz elementów scalonych.

1. WPROWADZENIE

Problem minimalizacji sieci kombinacyjnych jest jednym z najbardziej klasycznych problemów teorii układów przełączających i mającym bardzo bogatą literaturę, począwszy od prac Quine'a, Karnaugha i Mc Cluskey'a, aż po ostatnie lata, gdyż wciąż jeszcze dominują rozwiązania bramkowe. Dopiero szerokie rozpowszechnienie takich elementów, jak FPLA oraz urządzeń do ich programowania uczyni ten problem historycznym, nie wcześniej jednak, niż za kilka lat. Już pojawienie się przed laty dekodera w postaci scalonej dla wielu projektantów było znacznym ułatwieniem w realizacji wielowyjściowych sieci, przynajmniej dla kilku zmiennych. Nietrudno jednak stwierdzić i to już na prostych przykładach, że typowy sposób realizacji wielowyjściowej sieci w oparciu o dekoder, gdzie każdą z funkcji wyjściowych realizuje się niezależnie [2], daje układy nieminimalne zarówno ze względu na liczbę połączeń, jak i na liczbę elementów scalonych w stosunku do metod klasycznych.

Problem minimalizacji wielowyjściowych sieci kombinacyjnych w oparciu o dekoder postawiony został w [1], ale podany tam sposób realizacji jest na tyle nieefektywny przez swą pracochłonność i to przy wynikach nie zawsze lepszych od klasycznych, że stawia to pod znakiem zapytania jego przydatność w praktyce.

Tematem niniejszego artykułu będzie metoda realizacji wielowyjściowej sieci o minimalnej liczbie połączeń i elementów.

2. OPIS METODY

W klasycznych metodach efekt minimalizacji (redukcja zmiennych i operacji) związany jest zazwyczaj z warunkiem sąsiedniości elementarnych iloczynów zupełnych (EIZ), czyli tworzenie sześcianów k-wymiarowych, pokrywających sześciany 0÷k-1-wymiarowe [3], określających podzbiory EIZ, wchodzące w skład funkcji lub iloczynów funkcji wyjściowych [4, 5, 6].

Zauważmy, że ograniczenie to nie dotyczy sieci opartej na dekoderze, który generuje pełny zbiór EIZ. Problem sprowadza się więc do wygenerowania podzbiorów EIZ, wchodzących w skład funkcji lub ich iloczynów, przy czym zawsze możemy znaleźć podzbiory rozłączne.

Zbiór n funkcji wyjściowych określony jest rodziną M^1 podzbiorów EIZ, tworzących te funkcje.

$$M^1 = \left\{ M_1^1, M_2^1, \ldots M_n^1 \right\},$$

gdzie M_i^1 - podzbiór EIZ określający i-tę funkcję wyjściową.

Z kolei rodzinę M^i zdefiniujemy jako rodzinę tych podzbiorów EIZ, które wchodzą w skład iloczynów i-funkcji.

$$M^i = \left\{ M_1^i, M_2^i, \ldots, M_s^i \right\}, \quad s = \binom{n}{i},$$

gdzie M_k^i - k-ta kombinacja iloczynów i funkcji spośród n funkcji wyjściowych

$i \in \left\{ 1, 2, \ldots, n \right\}$, przy czym

$$M^n = M_1^n = M_1^1 \cap M_2^1 \cap \ldots \cap M_n^1$$

Jak łatwo zauważyć, wszystkie EIZ wchodzące w skład podzbiorów tworzących daną rodzinę z definicji wchodzą zawsze w skład wszystkich rodzin o niższych indeksach, a więc

$$M^n \subset M^{n-1} \subset \ldots \subset M^1$$

Chcąc wygenerować takie rodziny podzbiorów EIZ, dla których dowolny EIZ wystąpiłby tylko jeden raz, wprowadzimy zbiór D_k^i, określony następująco:

$$D_k^i = M_k^i - \bigcup_{s=1}^{\binom{n}{i+1}} M_s^{i+1}, \quad k \in \left\{ 1, 2 \ldots, \binom{n}{i} \right\}$$

Operacja powyższa generuje tym samym rodziny $D^1, D^2, \ldots, D^n$, a D_k^i zawiera wszystkie EIZ wchodzące w skład k-tej kombinacji iloczynów i funkcji spośród n funkcji wyjściowych oraz nie wchodzących do żadnego z iloczynów i+1 funkcji.

Definicja 1

D-implikantem nazwiemy każdy niepusty zbiór D_k^i.
Zilustrujemy powyższy opis przykładem dla trzech funkcji wyjściowych A, B i C (n=3).

Przykład 1

Dla prostoty pominiemy znak iloczynu zbiorów $\cap$,· a znak sumy $\cup$ zastąpimy zwykłym plusem, pamiętając jednak, że mamy do czynienia ze zbiorami:

$$M_1 = \{A, B, C\}$$
$$M_2 = \{AB, AC, BC\}$$
$$M_3 = \{ABC\}$$

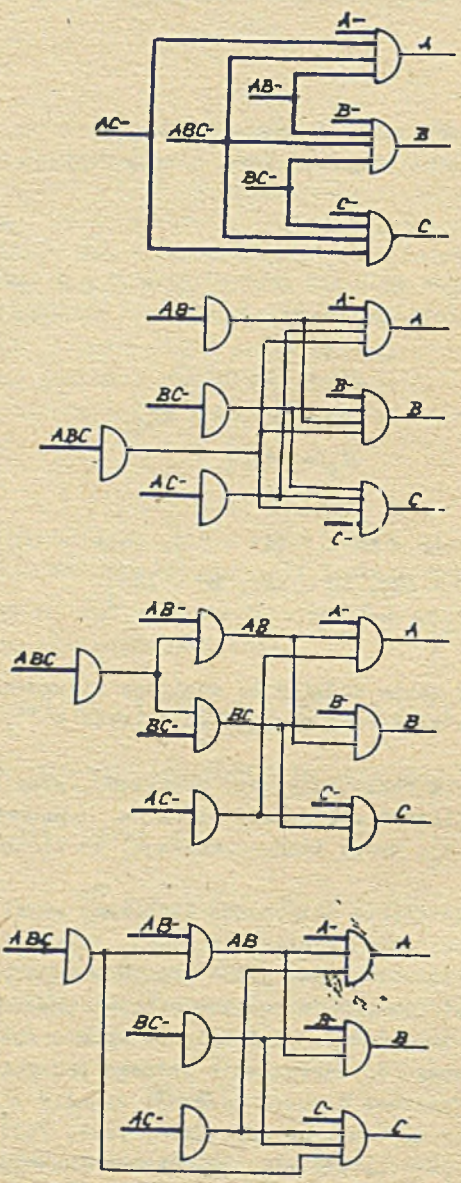
oraz

$$D_1 = \{A-(AB + AC), B-(AB + BC), C-(AC + BC)\}$$
$$D_2 = \{AB - ABC, AC - ABC, BC - ABC\}$$
$$D_3 = \{ABC\}$$

Jakkolwiek powyższy zapis rodziny D_1 sugeruje niezgodność z [1], łatwo zauważyć, że sumowanie wystarczy ograniczyć do tych M^{i+1}, które zawierają się w M_k^i.

Rysunek 1 przedstawia realizacje tego przykładu, przy czym dla uproszczenia zapisu D-implikantu opuszczono człon za minusem, co wystarczy dla odróżnienia D_k^i od M_k^i. Grube linie oznaczają, że liczba połączeń może być większa od 1.

Na rysunku 1a przedstawiono dotychczas stosowany zwykły sposób jednopoziomowej (nie licząc dekodera) realizacji sieci, jakkolwiek w przypadku ogólnym przy większej liczbie wejść jeden poziom sumujący może nie wystarczyć. Pozostałe rysunki prezentują możliwe warsje kaskadowej realizacji tej sieci za pomocą D-implikantów, zależnie od wyboru sposobu sumowania D-implikantów, koniecznego do zapewnienia pełnego pokrycia podzbiorów EIZ funkcji wyjściowych. Ponieważ dekoder generuje sygnały w logice negatywnej, sumowanie zachodzi na bramkach AND (7408, 7411, 7421 [7]), za wyjątkiem tych bramek wyjściowych, które są związane z funkcjami określonymi przez swoje składniki jedynek, co pociąga za sobą konieczność użycia bramek NAND. Dla funkcji określonej czynnikami zera bramka wyjściowa jest także bramką AND.



Rys. 1. Realizacja trójwyjściowej sieci w oparciu o dekoder: a) niezależna realizacja funkcji wyjściowych; b), c), d) przykłady rozwiązań metodą D-implikantów

Jak wynika z tego przykładu, reguła sumowania jest taka, że D-implikant i-tego rzędu opisany i-literowym literałem może być sumowany tylko z tymi D-implikantami niższych rzędów, których literały całkowicie się w nim zawierają, a ponadto sumarycznie całkowicie go pokrywają. W naszym przykładzie D-implikant ABC może być sumowany z następującymi D-implikantami AB- i BC-, AC- i BC-, AB- i AC-, AB- i C-, AC- i B-, BC- i A-, oraz A- i B- i C-. Wyboru dokonujemy w oparciu o zasadę wykorzystywania bramek o możliwie małej liczbie wejść lub bramek wolnych dostępnych w układzie, co prowadzi do minimalnej liczby użytych do realizacji elementów scalonych. Elastyczność wyboru wynika strukturalnie z dużej jednorodności sieci opartej na bramkach AND, które przy tym nie zmieniają logiki na każdym poziomie, jak bramki NOR czy NAND.

Warto zauważyć, że kierując się tymi samymi przesłankami ekonomicznymi podano w [8] ekonomiczny sposób realizacji dekodera za pomocą trzypoziomowej kaskady dwuwejściowych bramek AND, co istotnie obniżyło ilość połączeń i użytych elementów w stosunku do jednopoziomowej realizacji na bramkach czterowejściowych.

Łatwo stwierdzić, że zależnie od sposobu pokrycia ilość połączeń wyjściowych z bramki związanej z D-implikantem i-tego rzędu może wahać się od 2 do i. Pozwala to oszacować liczbę połączeń dla każdego typu realizacji sieci z dekoderem (przy założeniu braku ograniczenia liczby wejść na bramce).

Dla zwykłej realizacji jednopoziomowej (np. rys. 1a) dostajemy:

$$N_a = \sum_{i=1}^{n} \sum_{k=1}^{\binom{n}{i}} i \cdot d_k^i,$$

gdzie d_k^i jest mocą zbioru D_k^i.

Natomiast dla realizacji kaskadowej liczba połączeń zawiera się pomiędzy:

$$N_b = \sum_{k=1}^{n} d_k^1 + \sum_{i=2}^{n} \sum_{k=1}^{\binom{n}{i}} (d_k^i + i)$$

oraz

$$N_c = \sum_{k=1}^{n} d_k^1 + \sum_{i=2}^{n} \sum_{k=1}^{\binom{n}{i}} (d_k^i + 2)$$

Wzory na N_b i N_c są ścisłe, o ile $d_k^i \geqslant 2$, gdyż dla $d_k^i = 0$ znikają oczywiście całe nawiasy, a dla $d_k^i = 1$ równają się odpowiednio i lub 2, chyba że w ostatnim przypadku następuje sumowanie z D-implikantem wyższego rzędu, co wymaga użycia bramki i wyrażenie w nawiasie na N_c jest ścisłe. Praktycznie więc zachodzi zawsze $(i > 1)$:

$$i \cdot d_k^i \geqslant d_k^i + i \geqslant d_k^i + 2, \quad \text{a więc} \quad N_a \geqslant N_b \geqslant N_c,$$

przy czym różnice rosną w miarę wzrostu i, czyli efekty będą lepsze w miarę wzrostu liczby funkcji wyjściowych n, gdyż $1 \leqslant i \leqslant n$, a także dla większych d_k^i.

Można prześledzić to bliżej dla przypadku szczególnego

$$\bigwedge_{i,k} \left\{ d_k^i = l \right\}, \quad l \geqslant 2,$$

wtedy:

$$N_a = \sum_{i=1}^{n} \sum_{k=1}^{\binom{n}{i}} il = l \cdot \sum_{i=1}^{n} i\binom{n}{i} = l \cdot n \cdot 2^{n-1}$$

$$N_b = \sum_{k=1}^{n} l + \sum_{i=2}^{n} \sum_{k=1}^{\binom{n}{i}} (l+i) = nl + l \sum_{i=2}^{n} \binom{n}{i} + \sum_{i=2}^{n} i\binom{n}{i} =$$

$$= 2^{n-1}(2l + n) - (l + n)$$

$$N_c = \sum_{k=1}^{n} l + \sum_{i=2}^{n} \sum_{k=1}^{\binom{n}{i}} (l+2) = nl + (l+2) \sum_{i=2}^{n} \binom{n}{i} =$$

$$= (l+2)2^n - l - 2(n+1),$$

co potwierdza nasze przypuszczenia. Zauważmy przy tym, że wpływ ograniczonej ilości wejść bramek rzeczywistych jest znacznie większy dla N_a niż dla N_b i N_c, gdyż realizacje kaskadowe umożliwiają realizację sieci na bramkach o małej liczbie wejść.

3. PORÓWNANIE Z METODĄ GREERA-THOMPSONA [1]

Podstawowa różnica leży w odmiennej definicji D-implikantu, co rzutuje na efektywność metod. W [1] nie tworzy się podzbiorów rozłącznych, co powoduje, że każdy podzbiór M_k^i generuje nie pojedynczy D-implikant, ale aż 2^{n-i} D-implikantów, przy czym M_k^i jest również D-implikantem. Dla n funkcji daje to $\sum_{i=1}^{n} 2^{n-i} \binom{n}{i}$ D-implikantów, spośród których dokonuje się wyboru pokrycia, analogicznie do metody prostych implikantów. Ponadto cała metoda jest definicyjnie powiązana z siatkami Karnaugha jako narzędziem opisu i realizacji, co znacznie powiększa trudności w posługiwaniu się nią. Na przykład dla n=4 mamy do czynienia z 65 siatkami reprezentującymi D-implikanty, spośród których wybieramy pokrycie.
Dla n=6 czyni to już 665 siatek, co przy uwzględnieniu nawet, że część będzie zawierała same zera, nie zachęca projektanta do posługiwania się tą metodą.

Niniejsza praca, proponując rozłączne D-implikanty, ogranicza ich maksymalną ilość do $\sum_{i=1}^{n} \binom{n}{i} = 2^n - 1$ dla n funkcji, co dla n=4 i 6 daje odpowiednio 15 i 63, przy czym wybór pokrycia wynika natychmiast z podanej powyżej reguły sumowania D-implikantów.

Porównajmy oba podejścia na przykładzie zaczerpniętym właśnie z [1].

Przykład 2

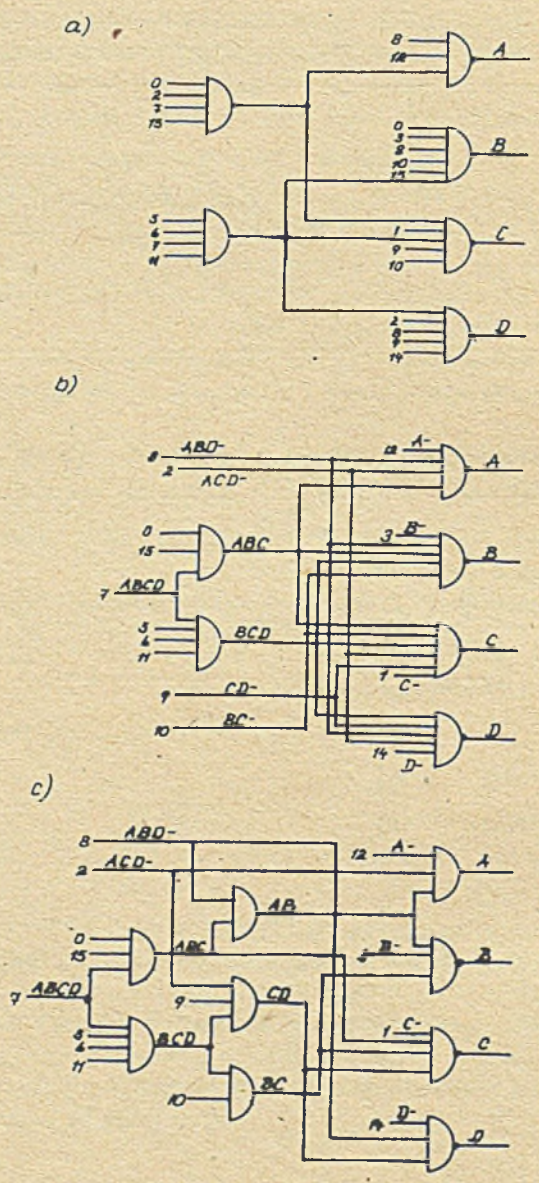
Dane są zbiory EIZ, określające 4 funkcje:

$$A = \{0\ 2\quad 78\quad 12\quad 15\}$$
$$B = \{0\quad 3\ 5678\ 1011\quad 15\}$$
$$C = \{012\quad 567\ 91011\quad 15\}$$
$$D = \{\quad 2\quad 56789\quad 11\quad 14\}$$

Rysunek 2a przedstawia optymalną wg [1] realizację, która wymagała skonstruowania 60 niezerowych siatek Karnaugha oraz dokonania spomiędzy nich wyboru pokrycia. Realizacja ta wymaga 27 połączeń, 5 elementów scalonych SSI (3 x 7430, 1 x 7421, 1/3 x 7410, czyli w myśl konwencji autorów $4\frac{1}{3}$) wraz dekodera.

Wygenerujmy z kolei rozłączne zbiory D-implikantów.

$$D_1 = \{A-;\ B-;\ C-;\ D-\} = \{12;\ 3;\ 1;\ 14\}$$
$$D_2 = \{BC-;\ CD-\} = \{10;\ 9\}$$
$$D_3 = \{ABC-;\ ABD-;\ ACD-;\ BCD-\} = \{0;\ 15;\ 8;\ 2;\ 5,\ 6,\ 11\}$$
$$D_4 = \{ABCD\} = \{7\}$$



Rys. 2. Realizacja sieci z przykładu 2

a) rozwiązanie z [1] (nierozłączne D-implikanty); b) dwupoziomowe rozwiązanie dla rozłącznych D-implikantów; c) trzypoziomowe rozwiązanie dla rozłącznych D-implikantów

Już najprostsza realizacja (rys. 2b) daje rezultat identyczny pod względem kosztów z poprzednim, przy praktycznie natychmiastowej realizacji, a trzypoziomowa kaskada z rys. 2c pozwala zrealizować te funkcje na 4 elementach scalonych, o koszcie liczonym wg [1] równym $3\frac{1}{6}$ (bez dekodera).

Zauważmy przy tym, że zapisywanie rodzin D_i może być pominięte i sieci można rysować bezpośrednio z zapisu numerycznego.

Porównanie obu metod na tym przykładzie dowodzi, że tylko proponowana tutaj metoda łączy zalety szybkiej i ekonomicznej realizacji sieci opartej na dekoderze, gdyż nadmiarowość nierozłącznych D-implikantów oraz niewygodny sposób opisu metody podanej w [1] uniemożliwiają praktyczne jej wykorzystywanie już dla kilku funkcji wyjściowych.

4. METODA D-IMPLIKANTÓW A METODY KLASYCZNE

Newralgicznym punktem przy tym porównaniu jest przyjęcie pewnego równoważnika bramkowego SSI dla dekodera, który jest 24-nóżkowym elementem MSI.

W [1] autorzy traktują dekoder jak zwykły element bramkowy, przyjmując równoważnik równy 1, z czym jednak trudno się zgodzić, choć autorom znakomicie to ułatwiło dyskusję przy porównywaniu kosztów sieci. Ponieważ jednak można łatwo zakwestionować także inne propozycje, jak np. przelicznik cenowy, końcówkowy (24/14), mocy pobieranej itp. lub ich kombinacje, problem ten zostawia się projektantom do indywidualnego rozstrzygnięcia. Zauważmy jednak, że udział dekodera w koszcie sieci maleje ze wzrostem n, który z kolei silnie wpływa na ilość prostych implikantów.

Rozwiązując przykład 2 metodą prostych implikantów, dostajemy sieć z 17 bramek, które łącznie z negacjami zmiennych wymagają użycia 9 elementów scalonych przy 63 połączeniach, podczas gdy uprzednio wymagało to 4 elementów + dekoder i 27 połączeń + 4 wejścia dekodera. Wynika to z faktu, że D-implikanty nie są ograniczone warunkiem sąsiedniości i liczebności (2^k) EIZ, tak jak proste implikanty. Prosty implikant będzie efektywniejszy od D-implikantu tylko wtedy, gdy utworzy bramkę o mniejszej ilości wejść niż D-implikant.

Ponieważ prosty implikant może być utworzony tylko przez 2^k EIZ i zrealizowany na bramce o r-k wejściach, gdzie r jest ilością zmiennych, zajdzie to dla warunku:

$$2^k > r-k$$

Ostra nierówność wynika z faktu, że prosty implikant może wymagać dodatkowo negacji zmiennych wejściowych.

Optymalnym rozwiązaniem sieci może być więc uwzględnienie takiej sytuacji w metodzie D-implikantów. Należy po prostu sprawdzić, czy dla EIZ

tworzących D_k^i zachodzą powyższe warunki i ewentualnie zastąpić D_k^i lub jego część implikantem prostym i wprowadzić odpowiednią bramkę NAND do sieci.

Jednakże dla większej ilości funkcji metoda prostych implikantów jest mało efektywna i stosuje się w tych przypadkach metody maszynowe [5].

Porównajmy więc odpowiednie realizacje sieci na poniższym przykładzie.

Przykład 3 [5, str. 56]

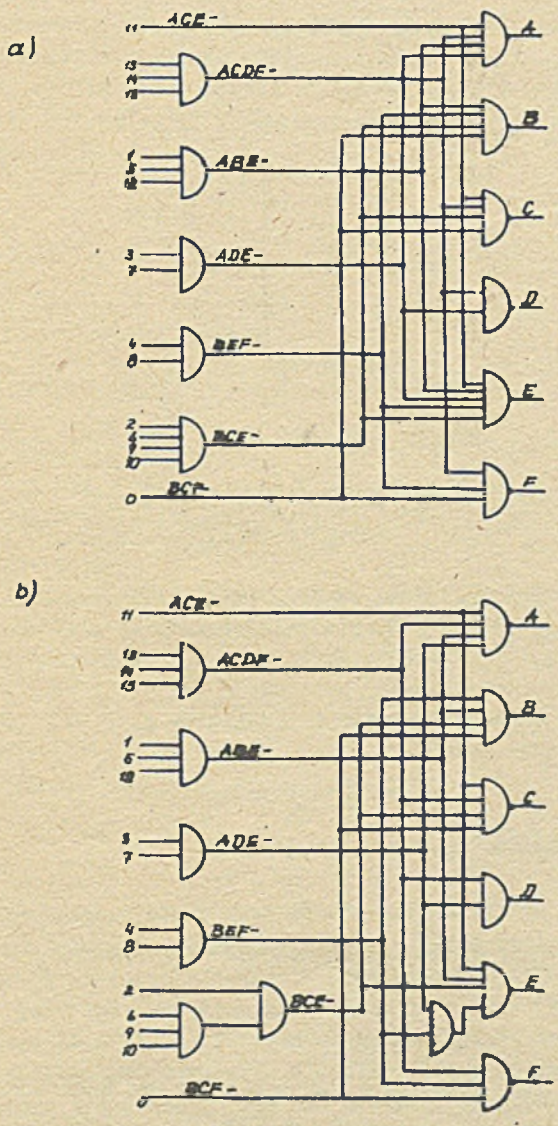
$$A = \{1\ 3\ 5\ 7\quad 1112131415\}$$
$$B = \{012\ 456\quad 910\quad 12\}$$
$$C = \{0\ 2\quad 6\quad 91011\quad 131415\}$$
$$D = \{\quad 3\quad 7\quad 131415\}$$
$$E = \{123456789101112\}$$
$$F = \{0\quad 4\quad 8\quad 131415\}$$

Realizacja metodą prostych implikantów będzie wymagała 10 elementów scalonych i ponad 70 połączeń, natomiast podane w [5] rozwiązanie maszynowe wymaga przy optymalnym wykorzystaniu bramek tylko 6 elementów scalonych przy 45 połączeniach i pięciu poziomach sieci.

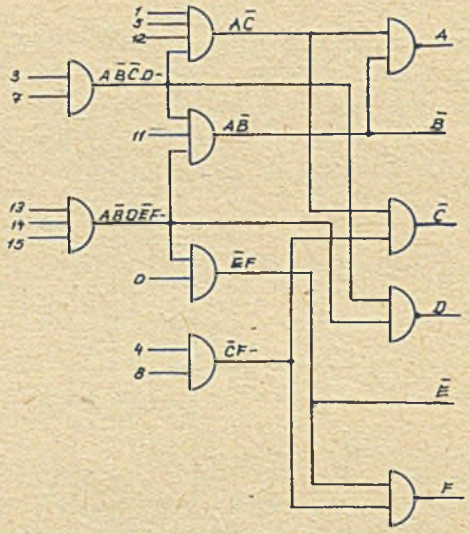
Rysunek 3a przedstawia schemat realizacji tej sieci metodą D-implikantów, który poprzez wykorzystanie wolnych bramek można zrealizować na 5 elementach scalonych i 38 połączeniach (rys. 3b) + oczywiście dekoder, przy czym jego koszt jest chyba z nawiązką kompensowany kosztem czasu przygotowania danych i pracy komputera, który zresztą podaje wynik w postaci równań, w oparciu o które należy dopiero zbudować wielopoziomową zazwyczaj sieć.

Ponadto szybkość realizacji sieci metodą D-implikantów pozwala na porównanie w krótkim czasie rozmaitych wersji realizacji sieci. Tak np. można przejść dla dowolnej z funkcji ze składników jedynki na czynniki zera lub odwrotnie. Zróbmy to dla najliczniejszych zbiorów B, C i E, oznaczając ich dopełnienia przez $\bar{B}$, $\bar{C}$ i $\bar{E}$.

$$A = \{1\ 3\ 5\ 7\quad 1112131415\}$$
$$\bar{B} = \{\quad 3\quad 7\quad 11\quad 131415\}$$
$$\bar{C} = \{1\ 345\ 78\quad 12\}$$
$$D = \{\quad 3\quad 7\quad 131415\}$$
$$\bar{E} = \{0\quad 131415\}$$
$$F = \{0\quad 4\quad 8\quad 131415\}$$



Rys. 3. Realizacja sieci z przykładu 3:
a) schemat realizacji metodą D-implikantów; b) ten sam schemat z uwzględnieniem niewykorzystanych uprzednio bramek



Rys. 4. Realizacja sieci z przykładu 3 dla funkcji A, D i F, określonych swymi składnikami jedynek oraz funkcji B, C i E, określonych swymi czynnikami zera

Rysunek 4 przedstawia realizację tej sieci, którą można zrealizować już na 4 elementach scalonych (pozostawiając jeszcze 3 wolne bramki) i 24 połączeniach + dekoder. Czytelnik może łatwo sprawdzić, że istnieje przynajmniej kilka jeszcze możliwości realizacji tej sieci przy podobnym koszcie, wykorzystując dopełnienia innych podzbiorów EIZ, co uwypukla elastyczność proponowanej metody.

5. WNIOSKI I UWAGI KOŃCOWE

Prezentowana tu metoda D-implikantów jest chyba jedyną metodą, która skutecznie łączy zalety szybkiej i ekonomicznej realizacji wielowyjściowych sieci kombinacyjnych, przy czym efekty ekonomiczne rosną w miarę wzrostu liczby wyjść sieci. Ograniczeniem stosowalności tej metody jest liczba zmiennych, ze względu na wykładnicze od niej uzależnienia liczby EIZ. Praktyczną granicą będzie liczba 6 zmiennych, wymagająca 4 dekoderów typu 74154, co jednak wystarczy dla wielu, jeśli nie większości zagadnień.

Metoda D-implikantów nie zapewnia likwidacji hazardu statycznego i dynamicznego w sieci. Z tego względu zrealizowana tą metodą sieć może sterować układami o stałej czasowej rzędu przynajmniej kilkusek nanosekund

(praktycznie więc wszystkie układy automatyki przemysłowej) albo występować w pętlach sprzężeń sekwencyjnych układów synchronicznych. Niemniej zauważmy, że to ograniczenie dotyczy niemal wszystkich metod realizacji sieci kombinacyjnych.

Jak powiedziano na wstępie, szerokie wprowadzenie elementów FPLA rozwiąże problem realizacji wielowyjściowych sieci dla wielu zmiennych, ale niewykluczone, że dla małej ilości zmiennych koszt sieci oparty na dekoderze może być konkurencyjny w stosunku do ceny elementu FPLA i metoda może pozostać aktualna.

LITERATURA

[1] GREER C.R., THOMPSON R.A.: Combinational Logic with decoder. IEEE Trans. on Comp., C-27, no 9. September 1978.

[2] WAGNER F., BUDKA M.: Synteza układów logicznych realizowanych na bazie demultiplekserów. Zeszyty Naukowe Pol.Śl., Automatyka z. 34, Gliwice 1976.

[3] HARRISON M.A.: Wstęp do teorii sieci przełączających i teorii automatów. PWN, Warszawa 1973.

[4] SIWINSKI J.: Układy przełączające w automatyce. WNT, Warszawa 1968.

[5] Praca zbiorowa pod kierunkiem prof. dr inż. MAJEWSKIEGO W. i dr inż. ALBICKIEGO A.: Projektowanie cyfrowych układów telekomunikacyjnych. WKiŁ, Warszawa 1977.

[6] TRACZYK W.: Układy cyfrowe automatyki. WNT, Warszawa 1976.

[7] The TTL data book for design engineers. Texas Instruments Inc. 1976.

[8] WAGNER F.: Projektowanie urządzeń cyfrowych. WNT, Warszawa 1978.

Wpłynęło do Redakcji: 25.04.1980 r.

W ostatecznej formie przyjęto: 22.04.1981 r.

Recenzent: Doc.dr hab. inż. Jan Piecha

МЕТОД СИНТЕЗА МИНИМАЛЬНЫХ МНОГОВЫВОДНЫХ КОМБИНАЦИОННЫХ СЕТЕЙ НА БАЗЕ ДЕКОДЕРА

Р е з ю м е

В статье представлен метод синтеза многовыводных комбинационных сетей на базе декодера, опираясь на определениие D-импликанты. Предлагаемый метод сопоставлен на литературных примерах с классическими методами, получая лучшие результаты как ввиду скорости проектирования, так и числа соединений и интегральных схем.

A METHOD OF OPTIMAL SYNTHESIS OF MULTIOUTPUT COMBINATIONAL LOGIC NETWORKS WITH A DECODER

S u m m a r y

In this paper a method for the fast and optimal synthesis of multioutput combinational logic networks is described. This method is based on the D-implicants, which are defined here.

It has been found that the D-implicant method, in comparison with classical methods, gives minimal number of connections and integrated circuits.