

Andrzej SKORUPSKI¹, Marek SZYPROWSKI², Paweł KERNTOPF^{2,3}

¹ WYŻSZA SZKOŁA Menedżerska, ul. Kawęczyńska 36, 03-772 Warszawa

² POLITECHNIKA WARSZAWSKA, WYDZIAŁ ELEKTRONIKI I TECHNIK INFORMACYJNYCH, ul. Nowowiejska 15/19, 00-665 Warszawa

³ UNIwersytet Łódzki, WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ, ul. Pomorska 149/153, 90-236 Łódź

Algorytm syntezy kombinacyjnych układów odwracalnych

Dr inż. Andrzej SKORUPSKI

Do 2008 r. adiunkt w Instytucie Informatyki Politechniki Warszawskiej, obecnie docent w Wyższej Szkole Menedżerskiej w Warszawie. Autor wielu publikacji dotyczących projektowania układów cyfrowych i architektury komputerów. Jego zainteresowania naukowe są związane z praktyką inżynierską. Jest współprojektantem laboratoryjnych systemów prototypowania urządzeń i systemów cyfrowych wykorzystywanych zarówno w dydaktyce, jak i w pracach badawczych.

e-mail: ask@ii.pw.edu.pl



Mgr inż. Marek SZYPROWSKI

Ukończył studia magisterskie na Wydziale Elektroniki i Techniki Informatycznych Politechniki Warszawskiej. Obecnie odbywa studia doktoranckie w Instytucie Informatyki na tym Wydziale. Jego zainteresowania naukowe koncentrują się wokół układów odwracalnych, które stanowiły temat jego pracy magisterskiej.

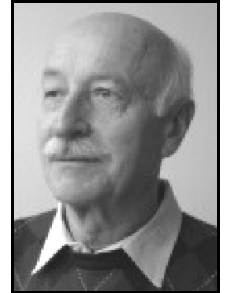
e-mail: M.Szyprowski@ii.pw.edu.pl



Dr hab. inż. Paweł KERNTOPF

Ukończył studia na Wydziale Elektroniki i Techniki Informatycznych Politechniki Warszawskiej. Obecnie pracuje na stanowisku profesora nadzwyczajnego w Instytucie Informatyki na tym Wydziale i w Katedrze Fizyki Teoretycznej i Informatyki na Wydziale Fizyki i Informatyki Teoretycznej Uniwersytetu Łódzkiego. Jego zainteresowania naukowe to synteza układów logicznych, odwracalne układy logiczne, kwantowe układy logiczne, binarne i wielowartościowe diagramy decyzyjne.

e-mail: P.Kerntopf@ii.pw.edu.pl



SAT [15, 2], ale nawet dla niewielu wejść nie udało się opracować metod syntezy prowadzących zawsze do układów o minimalnej liczbie bramek. W pracy zaprezentowano algorytm syntezy oparty na nowej koncepcji przedstawiania zamian wierszy w tablicy prawdy i na tej podstawie wyboru kolejnej bramki w układzie. Podano przykład zastosowania tego algorytmu do syntezy układów o trzech wejściach i trzech wyjściach z bramek Toffoliego.

2. Bramki odwracalne

Dla funkcji trzech zmiennych jest 12 bramek Toffoliego, które są najczęściej rozpatrywane w literaturze. Każda z nich realizuje trzy zrównoważone funkcje boolowskie, tj. mające w tablicy prawdy tyle samo wartości 1, co wartości 0. Na rys. 1 pokazano cztery z tych bramek T0, C0-2, C0-1 i N0, mające wyjścia sterowane na linii x_0 , oraz zamieniane przez nie wiersze w tablicy prawdy. Analogicznie, bramki mające wyjścia sterowane na linii x_1 oznaczane będą przez T1 (zamienia wiersze 5 i 7 w tablicy prawdy), C1-0 (zamienia wiersze 1 i 3 oraz 5 i 7), C1-2 (zamienia wiersze 4 i 6 oraz 5 i 7) oraz N1 (zamienia wiersze 0 i 2, 1 i 3; 4 i 6 oraz 5 i 7), a bramki mające wyjścia sterowane na linii x_2 będą oznaczane przez T2 (zamienia wiersze 3 i 7), C2-0 (zamienia wiersze 1 i 5 oraz 3 i 7), C2-1 (zamienia wiersze 2 i 6 oraz 3 i 7) oraz N2 (zamienia wiersze 0 i 4, 1 i 5; 2 i 6 oraz 3 i 7).

Łatwo zauważyć, że każda z bramek Toffoliego zamienia wiersze różniące się tylko na jednej pozycji, która odpowiada linii, na której umieszczone jest wyjście sterowane tej bramki. Ponadto, można zauważyć, że żadna bramka poza bramkami N nie zamienia wiersza, w którym wszystkie zmienne są równe zero (tzn. wiersza 0), natomiast każda bramka zamienia wiersz, w którym wszystkie zmienne są równe jeden (tzn. wiersz 7).

Streszczenie

W pracy przedstawiono koncepcję nowego algorytmu syntezy układów odwracalnych. Jest on oparty na oryginalnej reprezentacji zamiany wierszy w tablicy prawdy. Dla układów o trzech wejściach i trzech wyjściach sformułowano kryteria takiego doboru bramek, aby otrzymać układ zbliżony do optymalnego. Następnie podano przykład zastosowania przedstawionego algorytmu do syntezy układów o trzech wejściach i trzech wyjściach z bramek Toffoliego.

Słowa kluczowe: odwracalne układy logiczne, bramki Toffoliego.

Algorithm for reversible circuit synthesis

Abstract

A gate or circuit is reversible if there is one-to-one correspondence between its input signals and output signals, i.e. if they implement bijective functions. Research on reversible logic circuits is motivated by advances in quantum computing, nanotechnology and low-power design. Recently, the attention has been focused on the synthesis of reversible circuits built from the NCT library of gates consisting of NOT, CNOT and Toffoli gates. This paper presents a novel algorithm for synthesis of reversible circuits. It is based on a new representation of row exchanges in the truth table. There is described how each possible row exchange determines the set of subsequent gates in a circuit, basing on the newly introduced cube of row exchanges. Next, the criteria for the choice of NCT reversible gates are formulated. For an exemplary function, the presented algorithm generates an optimal reversible circuit with 3 inputs and 3 outputs. It can also be generalized to any number of inputs and outputs.

Keywords: reversible logic circuits, Toffoli gates.

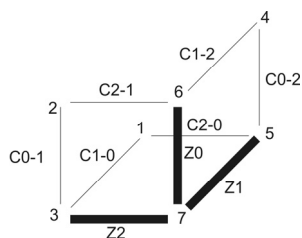
1. Wstęp

Układy odwracalne realizują wzajemnie jednoznaczne przekształcenia. Są intensywnie badane ze względu na zastosowania w obliczeniach kwantowych, nanotechnologiach, układach o małym poborze energii, kryptografii i przetwarzaniu sygnałów [1-4].

Szczególnie intensywne prace prowadzone są nad synteza układów odwracalnych [1, 2, 5, 6]. Zaproponowano algorytmy, oparte na przekształceniach tablic prawdy [7], wyrażeniach boolowskich [8], widmach funkcji boolowskich [9, 10], diagramach decyzyjnych [11, 2], permutacjach [12], teorii grup [13, 14] i narzędziach

x_2 —●— $y_2 = x_2$ x_1 —●— $y_1 = x_1$ x_0 —⊕— $y_0 = x_0 \oplus x_1 x_2$	T0	Zamienia wiersze 6 i 7
x_2 —●— $y_2 = x_2$ x_1 —●— $y_1 = x_1$ x_0 —⊕— $y_0 = x_0 \oplus x_2$	C0-2	Zamienia wiersze 4 i 5 oraz 6 i 7
x_2 —●— $y_2 = x_2$ x_1 —●— $y_1 = x_1$ x_0 —⊕— $y_0 = x_0 \oplus x_1$	C0-1	Zamienia wiersze 2 i 3 oraz 6 i 7
x_2 —●— $y_2 = x_2$ x_1 —●— $y_1 = x_1$ x_0 —⊕— $y_0 = x_0 \oplus 1$	N0	Zamienia wiersze 0 i 1, 2 i 3, 4 i 5 oraz 6 i 7

Rys. 1. Graficzne oznaczenia bramek Toffoliego i zamieniane przez nie wiersze
Fig. 1. Pictorial representations of Toffoli gates and corresponding numbers of exchanged rows in a truth table



Rys. 2. Sześcian zamiany wierszy
Fig. 2. Cube of exchanged rows

Zamianę wierszy dokonywaną przez bramki można pokazać graficznie na sześcianie (kostce), co przedstawiono na rys. 2. Na sześcianie pominięto bramki negacji N_0 , N_1 , N_2 ponieważ założono, że w wierszu zerowym tablicy prawdy wszystkie funkcje zrównoważone mają wartość 0. Pogrubione krawędzie sześcianu oznaczone przez Z_0 , Z_1 i Z_2 oznaczają zbiór wszystkich trzech bramek umieszczonych odpowiednio na liniach 0, 1 i 2.

Na rys. 2 widać, że zamiana wiersza 2 z jednym z pozostałych wierszy może być dokonana tylko za pomocą bramek C_0-1 i C_2-1 . Posługując się tym sześcianem można ułożyć tablicę zamiany wierszy. Tablica taka ma wskazywać, którą bramkę należy zastosować jako pierwszą, aby dany wiersz (źródłowy) znalazł się w określonym miejscu (w wierszu docelowym). Wiersze źródłowe są umieszczone w wierszach tablicy na rys. 3, a wiersze docelowe są umieszczone w kolumnach tej tablicy. Pokazana na rys. 3 tablica zamiany jest wykorzystywana do wyznaczenia pierwszej bramki danej zamiany. Jeśli wiersze te różnią się na jednej pozycji, to można je zamienić jedną bramką. Natomiast, jeśli wiersze te różnią się na dwóch pozycjach, to trzeba użyć dwóch bramek, aby je zamienić. Jeśli zamieniane wiersze różnią się na trzech pozycjach, to ich zamiana wymaga trzech bramek. Rys. 3 wskazuje tylko pierwszą bramkę.

Wiersz nr	Wstawić na pozycję						
	1	2	3	4	5	6	7
1	-	C0-1	C1-0	C0-2	C2-0	Z0	T1,C1-2 T2,C2-1
2	C1-0	-	C0-1	C1-2	Z1,C0-2 C2-0	C2-1	T0 T2
3	C1-0	C0-1	-	C0-2 C1-2	Z1	Z0, C2-1	Z3
4	C2-0	C2-1	Z2,C0-1 C1-0	-	C0-2	C1-2	Z2
5	C2-1	C2-1 C0-1	Z2	C1-2	-	Z0, C1-2	Z2
6	C1-0 C2-0	C2-1	Z2	C1-2	Z1,C0-2	-	Z1
7	C1-0 C2-0	C0-1 C2-1	Z2	C0-2 C1-2	Z1	Z0	-

Rys. 3. Tablica zamiany wierszy
Fig. 3. Table of exchanging rows

3. Własności układów odwracalnych

W niniejszej pracy przyjmujemy założenie, że wszystkie funkcje układu odwracalnego mają wartość zero w wierszu zerowym. Jeśli nie jest ono spełnione, to należy najpierw dla każdej funkcji, która nie spełnia tego warunku, umieścić bramkę negacji na linii odpowiadającej tej funkcji.

Zakłada się także, że wybór bramek do realizacji danej funkcji odwracalnej będzie następował przez modyfikację lewej strony tablicy prawdy, a więc zamieniane będą wiersze wejściowe. Możliwe jest dualne podejście, w którym wybiera się bramki na podstawie prawej strony tabeli, czyli zmieniając wiersze wyjściowe. Porównując lewą stronę tablicy prawdy z jej prawą stroną można

wyznaczyć konieczne zamiany wierszy. Maksymalna liczba zamian wierszy wynosi 7.

Jak można zauważyć na rys. 3, niektóre zamiany wierszy mogą być dokonane tylko na jeden sposób, ale niektóre – na wiele sposobów. Algorytm poszukiwania minimalnego układu odwracalnego ma za zadanie wyznaczyć ciąg bramek (układ realizujący funkcję odwracalną jest kaskadą bramek), który dokonuje takich zamian wierszy, że na wyjściu ostatniej bramki ciągu pojawia się zadana funkcja.

4. Algorytm syntezy układów odwracalnych

Niech będzie dana tablica prawdy funkcji odwracalnej trzech zmiennych. Najdłuższa kaskada układu optymalnego ma 8 bramek [12, 6]. Dlatego algorytm ich poszukiwania składa się z ośmiu iteracji. W każdej z nich trzeba wyznaczyć zbiór bramek, których można użyć w następnej iteracji, a więc bramek, które można wstawić w kolejne miejsce ciągu. W pierwszej z nich wyznacza się zbiór bramek, które mogą być umieszczone na pierwszym miejscu ciągu. Zbiór ten określany jest na podstawie tablicy prawdy. Ze zbioru tego kolejno wybiera się bramki i tworzy się nową tablicę prawdy z zamienionymi przez tą bramkę wierszami. Nowa tablica prawdy stanowi podstawę dla następnej iteracji.

Rozpatrzmy pierwszą iterację. Na podstawie tablicy prawdy należy wyznaczyć wszystkie konieczne zamiany wierszy. Z poprzednich rozważań wynika, że zamian tych może być co najwyżej siedem. Z rys. 3 dla każdej z nich wyznacza się bramki, które rozpoczynają tę zamianę. Bramki te należą do zbioru bramek, które można umieścić jako pierwsze w ciągu.

Proponowany algorytm polega na wyznaczeniu kryteriów dla takiego doboru bramek, który prowadzi do optymalnego lub prawie optymalnego rozwiązania. W każdej z iteracji stosowane są następujące kryteria:

1. Dla każdej zamiany wierszy, której trzeba dokonać w tablicy prawdy zadanej funkcji, określa się liczbę pozycji, które ulegają zmianie i wybiera się bramki umieszczone na liniach, które zmieniają dane pozycje. Następnie, wybiera się bramki, które prowadzą do największej liczby zmian.
2. Jeśli jest więcej niż jedna taka zmiana, to poszukuje się bramki, która jest dla nich (lub ich części) wspólna i wybiera się tę bramkę. Jeśli nie ma takiej bramki, to rozpatruje się kolejno następne takie zmiany.
3. Jeśli po zastosowaniu bramek istnieją zmiany, dla których zmieniana jest więcej niż jedna pozycja, to ze zbioru bramek wybiera się te, które mają wyjście sterowane na innej linii niż rozpatrywana wcześniej bramka.
4. Spośród zamian wierszy wybiera się te, które mogą być realizowane przez pojedyncze bramki. Oprócz zamiany wiersza 7, są to następujące zamiany: 1 i 3 (C_1-0), 1 i 5 (C_2-0), 2 i 3 (C_0-1), 2 i 6 (C_2-1), 4 i 5 (C_0-2) oraz 4 i 6 (C_1-2).

5. Przykład działania algorytmu

Rozpatrzmy funkcję pokazaną na rys. 4, która zamienia wiersze 1 i 6, zaś nie zmienia pozostałych wierszy.

$x_2x_1x_0$	$y_2y_1y_0$	Zamieniane wiersze
000	000	-
001	110	1 → 6
010	010	-
011	011	-
100	100	-
101	101	-
110	001	6 → 1
111	111	-

Rys. 4. Zamieniane wiersze dla przykładowej funkcji
Fig. 4. Exchanged rows for the exemplary function

Z rys. 3 wynika, że aby wstawić wiersz 1 na pozycję 6, można użyć bramek Z0. Natomiast, aby wstawić wiersz 6 na pozycję 1, trzeba użyć bramek C1-0 lub C2-0. Załóżmy, że zostanie wybrana bramka C1-0. Na rys. 5 w kolumnie nazwanej bramka C1-0 pokazano dokonane przez tę bramkę zamiany wierszy wejściowych. Zatem będą to wejścia bramki następnej.

$x_2x_1x_0$	$y_2y_1y_0$	C1-0	Zamieniane wiersze
000	000	000	-
001	110	011	3 → 6 (2)
010	010	010	-
011	011	001	1 → 3 (1)
100	100	100	-
101	101	111	7 → 5 (1)
110	001	110	6 → 1 (3)
111	111	101	5 → 7 (1)

Rys. 5. Wiersze zamieniane przez bramkę C1-0
Fig. 5. Rows exchanged by gate C1-0

Na rys. 5 w kolumnie „Zamieniane wiersze” pokazano aktualnie wymagane zamiany wierszy wraz z liczbą zmienianych pozycji (w nawiasie). Najbardziej liczną jest zmiana 6→1 i ona decyduje o wyborze następnej bramki. Z trzeciego punktu algorytmu wynika, że należy wybrać bramkę C2-0. Na rys. 6 pokazano tablicę prawdy po zastosowaniu tej bramki wraz z koniecznymi zmianami.

$x_2x_1x_0$	$y_2y_1y_0$	C1-0	C2-0	Zamieniane wiersze
000	000	000	000	-
001	110	011	111	7 → 6 (1)
010	010	010	010	-
011	011	001	101	5 → 3 (2)
100	100	100	100	-
101	101	111	011	3 → 5 (2)
110	001	110	110	6 → 1 (3)
111	111	101	001	1 → 7 (2)

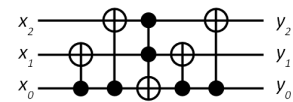
Rys. 6. Wiersze zamieniane przez bramkę C2-0
Fig. 6. Rows exchanged by gate C2-0

Ponownie najbardziej liczną jest zmiana 6→1 i ona decyduje o wyborze następnej bramki. Ponieważ nadal istnieją zmiany dotyczące więcej niż jednej pozycji (oprócz zmiany 7→6), to należy zmienić linię. Po liniach 1 i 2 jest kolej na linię 0. Na tej linii żadna bramka nie zmienia wiersza 1, więc należy zamienić wiersz 6. Może on zostać zmieniony na linii 0 przez jedną z trzech bramek: T0, C0-1 lub C0-2. Wybierzmy bramkę T0 (procedurę należy powtórzyć dla pozostałych bramek, ale wybranie bramki C0-1 i C0-2 prowadzi do układów o większej liczbie bramek).

$x_2x_1x_0$	$y_2y_1y_0$	C1-0	C2-0	T0	Zamieniane wiersze
000	000	000	000	000	-
001	110	011	111	110	-
010	010	010	010	010	-
011	011	001	101	101	5 → 3 (2)
100	100	100	100	100	-
101	101	111	011	011	3 → 5 (2)
110	001	110	110	111	7 → 1 (2)
111	111	101	001	001	1 → 7 (2)

Rys. 7. Wiersze zamieniane przez bramkę T0
Fig. 7. Rows exchanged by gate T0

Na podstawie rys. 3 można zauważyć, że przecięciem zbiorów bramek są bramki C1-0 i C2-0. Rozpatrując w analogiczny sposób jeszcze jeden krok otrzymuje się dwa rozwiązania: C1-0, C2-0, T0, C1-0, C2-0 oraz C1-0, C2-0, T0, C2-0, C1-0, w których ostatnie dwie bramki mogą być użyte w dowolnej kolejności. Te dwa rozwiązania stanowią część rozwiązań optymalnych, których dla tej funkcji jest sześć. Na rys. 8 pokazano jeden z dwóch otrzymanych optymalnych układów. Wspomniane wyżej rozgałęzienia prowadzą do znalezienia pozostałych rozwiązań optymalnych.



Rys. 8. Jeden z optymalnych układów dla przykładowej funkcji
Fig. 8. One of the optimal circuits for the exemplary function

6. Podsumowanie

W artykule opisany został nowy algorytm syntezy układów odwracalnych, którego działanie zostało przedstawione dla przykładu syntezy układu o trzech wejściach i trzech wyjściach, jednak jego zaprogramowana wersja może być stosowana dla funkcji o większej liczbie zmiennych.

Praca była wykonana w ramach realizacji grantu MNiSzW nr 4180/B/T02/2010/38.

7. Literatura

- [1] De Vos A.: Reversible Computing. Fundamentals, Quantum Computing, and Applications. Wiley-VCH, Berlin 2010.
- [2] Wille R., Drechsler R.: Towards a Design Flow for Reversible Logic. Springer, Dordrecht 2010.
- [3] Nielsen M., Chuang I.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge 2000.
- [4] Axelsen H. B., Glück R., De Vos A., Thomsen M. K., MicroPower: Towards low-power microprocessors with reversible computing. <http://ercim-news.ercim.eu/en79/special-theme/micropower-towards-low-power-microprocessors-with-reversible-computing>, 2010.
- [5] Kerntopf P.: Synteza odwracalnych układów logicznych, Pomiar Automatyka Kontrola, vol. 53, nr 7, 2007, pp.78-80.
- [6] Szyprowski M., Kerntopf P.: Porównanie efektywności heurystycznych miar złożoności odwracalnych funkcji boolowskich, Pomiar Automatyka Kontrola, vol. 55, nr 8, 2009, pp. 581-583.
- [7] Miller D.M., Maslov D., Dueck G.W.: A transformation based algorithm for reversible logic synthesis, Proc. 40th Design Automation Conference, Anaheim, CA, USA, 2003, pp. 318-323.
- [8] Gupta P., Agrawal A., Jha N.K.: An algorithm for synthesis of reversible logic circuits, IEEE Trans. on Computer-Aided Design, 2006, vol. 25, pp. 2317-2330.
- [9] Miller D.M., Maslov D.: Spectral Techniques for Reversible Logic Synthesis, Proc. 6th Int. Symp. on Representations and Methodology of Future Computing Technology, Trier, Germany, 2003, pp. 56-62.
- [10] Maslov D., Dueck G.W., Miller D.M.: Techniques for the synthesis of reversible Toffoli networks, ACM Transactions on Design Automation of Electronic Systems, 12, 4 (Sept. 2007), article 42: pp. 1-28.
- [11] Kerntopf P.: A new heuristic algorithm for reversible logic circuit synthesis, Proc. 41st Design Automation Conference, San Diego, USA, 2004, pp. 834-837.
- [12] Shende V.V., Prasad A.K., Markov I.L., Hayes J.P.: Synthesis of reversible logic circuits, IEEE Trans. on Computer-Aided Design, 2003, vol. 22, no 6, pp. 710-722.
- [13] Van Rentergem Y., De Vos A., De Keyser K.: Using group theory in reversible computing. Proc. IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, July 2006.
- [14] Yang G., Song X., Hung W.N.N., Xie F., Perkowski M.A.: Group theory based synthesis of binary reversible circuits. Proc. 3rd International Conference on Theory and Applications of Models of Computation, Lecture Notes in Computer Science, 3959, 2006, pp. 365-374.
- [15] Grosse D., Wille R., Dueck G. and Drechsler R.: Exact Multiple Control Toffoli Network Synthesis with SAT Techniques, IEEE Trans. on CAD, vol. 28, 2009, pp. 703-715.