Valery SALAUYOU, Tomasz GRZEŚ
POLITECHNIKA BIAŁOSTOCKA, WYDZIAŁ INFORMATYKI

Metody kodowania stanów wewnętrznych automatu skończonego minimalizujące pobór mocy

Prof. dr hab. inż. Valery SALAUYOU

Ukończył w 1978 r. studia na Wydziale Matematyki Stosowanej w Białoruskim Państwowym Uniwersytecie w Mińsku. W 1986 r. obronił pracę doktorską, a w 2003 r. uzyskał tytuł doktora habilitowanego. Od 25 lat pracuje w dziedzinie projektowania logicznego systemów cyfrowych.

e-mail: walsol@wi.pb.edu.pl

Mgr inż. Tomasz GRZEŚ

Ukończył studia w Instytucie Informatyki Politechniki Białostockiej w 1999 r. Od 2000 r. jest asystentem w Instytucie Informatyki, a następnie na Wydziale Informatyki Politechniki Białostockiej. Jego zainteresowania naukowe to problemy minimalizacji mocy w układach cyfrowych opartych o struktury programowalne CPLD i FPGA.

e-mail: grzes@chilan.com

Streszczenie

Opisano badania trzech algorytmów kodowania stanów wewnętrznych automatu skończonego: algorytmu kodowania kolumnowego, algorytmu „wyżarzania” oraz algorytmu sekwencyjnego. Głównym zadaniem wymienionych algorytmów jest zakodowanie stanów wewnętrznych automatu skończonego w taki sposób, aby moc pobierana przez automat skończony była jak najmniejsza. Badania eksperymentalne, które przeprowadzono na standardowych układach testowych, potwierdziły wyższość opracowanego przez autorów algorytmu sekwencyjnego.

Słowa kluczowe: automat skończony, pobór mocy, kodowanie stanów wewnętrznych.

The State Assignment of Finite State Machine for Minimizing the Power Dissipation

Abstract

The reduction of the power dissipation is of extreme importance for mobile, battery-operated systems as well as for increasing the speed and performance of the digital systems. Based on the CMOS gate model we can prove that power dissipation depends on the applied assignment. Thus using the particular state assignment method lead to minimization of the power dissipation. In this paper three algorithms of the FSM internal states assignment were described: column-based, annealing and sequential. The main aim of those algorithms were to minimize the power dissipation in the sequential circuits by assigning the state codes with as minimal Hamming distance as possible. Experimental results show that sequential algorithm can reduce about 10% more power than other discussed algorithms.

Keywords: finite state machine, power dissipation, state assignment.

1. Wprowadzenie

Problem minimalizacji mocy układów ma ogromne znaczenie przy konstruowaniu systemów cyfrowych zasilanych bateryjnie, jak również do zwiększenia wydajności i szybkości systemu. Większość układów produkowanych jest w technologii CMOS, które cechuje pobór mocy tylko w czasie przełączania. Moc pobieraną przez element a układu można obliczyć z przedstawionego poniżej równania [4]:

$$P_a = \frac{1}{2} \cdot \frac{V_{CC}^2}{T_{cycle}} \cdot N_a \cdot C_a \qquad (1)$$

gdzie: V_{CC} – napięcie zasilające układ; T_{cycle} – czas względem którego rozpatrywana jest aktywność przełączania; C_a – pojemność wyjściowa elementu a; N_a – aktywność przełączania (średnia ilość zmian stanu wyjścia w czasie trwania jednego cyklu T_{cycle}) elementu a.

Aby zmniejszyć pobór mocy należy zminimalizować wartości występujące po prawej stronie równania (1), jednakże jedynym parametrem niezależnym od technologii jest aktywność przełączania N_a. Jedną z metod zmniejszenia aktywności przełączania układów sekwencyjnych jest zmiana sposobu kodowania stanów wewnętrznych automatu. Klasycznymi przykładami rozwiązania problemu są [6], gdzie zastosowano niedeterministyczny algorytm polegający na „wygrzewaniu” (ang. *annealing*), jak również [1], gdzie zastosowano kodowanie kolumnowe (ang. *column-based approach*).

2. Sformułowanie problemu

Automaty skończone można opisywać za pomocą dyskretnych łańcuchów Markowa $\{X^t \mid t \in T\}$ ze skończoną liczbą stanów $A = \{a_1, \ldots, a_M\}$ przy założeniu, że przestrzeń czasu T jest dyskretna [2].

Prawdopodobieństwa statyczne, określające prawdopodobieństwo znalezienia automatu w określonym stanie, można obliczyć korzystając z równań Chapmana-Kołmogorowa [5]. Dla każdego stanu a_i automatu skończonego ($1 \le i \le M$) można zapisać równanie:

$$P(a_i) = \sum_{k=1}^{M} P(a_k) \cdot P(I_{ki}) \qquad (2)$$

gdzie: $P(a_i)$ – prawdopodobieństwo statyczne stanu a_i; $P(a_k)$ – prawdopodobieństwo statyczne stanu a_k; $P(I_{ki})$ – prawdopodobieństwo pojawienia się wektora wejściowego I_{ki}, który spowoduje przejście automatu ze stanu a_k do stanu a_i [7].

Prawdopodobieństwo zmiany stanu z a_i na a_j, pod warunkiem, że automat znajduje się w stanie a_i jest zależne od prawdopodobieństwa pojawienia się na wejściu wektora I_{ij}, który spowoduje zmianę stanu z a_i na a_j [3]. W związku z tym prawdopodobieństwo zmiany przez automat skończony stanu z a_i na a_j $P(a_i \to a_j)$ można wyliczyć korzystając z równania (3):

$$P(a_i \to a_j) = P(a_i) \cdot P(I_{ij}) \qquad (3)$$

Automat Można przedstawić w postaci grafu, którego węzłami są stany automatu, natomiast krawędzie odpowiadają przejściom między stanami. Ponieważ zmiana stanu automatu z a_i na a_j wymaga przełączenia takiej samej ilości przerzutników, co zmiana stanu z a_j na a_i, wagi poszczególnych krawędzi można przedstawić jako [1]:

$$w_{i,j} = P(a_i \to a_j) + P(a_j \to a_i) \qquad (4)$$

Implementując automat skończony w postaci układu sekwencyjnego każdemu stanowi a_i należy przypisać kod c_i. W związku z tym zbiorowi stanów automatu A odpowiada zbiór C kodów stanów automatu skończonego, $C = \{c_1, c_2, \ldots, c_M\}$. Każdy kod musi być ortogonalny ze wszystkimi pozostałymi kodami stanów automatu skończonego. Ilość bitów kodu R może być wartością z zakresu [int $\log_2 M$, M].

Niech c_i^l oznacza l-ty bit kodu stanu a_i. Odległość Hamminga $H(c_i, c_j)$, określona jest jako ilość bitów w kodach c_i i c_j znajdujących się na tych samych pozycjach mających przeciwne wartości, może być opisana wzorem (5):

$$H(c_i, c_j) = \sum_{l=1}^{N} c_i^l \oplus c_j^l \quad (5)$$

Problem znalezienia sposobu kodowania, który doprowadzi do minimalizacji poboru mocy można przedstawić w postaci zadania całkowitoliczbowego programowania liniowego (ang. *Integer Linear Programming*).

$$Min\left(\sum_{i=1}^{M}\sum_{j=1}^{M} w_{i,j} \cdot H(c_i, c_j)\right) \quad (6)$$

przy ograniczeniach:

$$\sum_{l=1}^{N} c_i^l \oplus c_j^l \geq 1,\ \forall c_i, c_j,\ c_i \neq c_j \quad (7)$$

Ponieważ zadanie opisane równaniami (6) i (7) jest problemem NP-zupełnym, stosuje się rozwiązania heurystyczne.

3. Algorytmy kodowania stanów wewnętrznych

Algorytm kodowania kolumnowego bazuje na pojęciu klasy nierozróżnialności kodów stanów wewnętrznych [1]. Kody stanów mające identyczne kody częściowe należą do tej samej klasy nierozróżnialności, co można zapisać jako:

$$c_i, c_j \in C_{k,l} \Leftrightarrow c_i^m = c_j^m, \forall 1 \leq m \leq l \quad (8)$$

Kodowanie zakończy się sukcesem tylko wtedy, gdy ilość kodów w każdej klasie jest ograniczona zależnością (9):

$$\left|C_{k,l}\right| \leq 2^{R-l} \quad (9)$$

Wykorzystując klasy nierozróżnialności można zakodować stany wewnętrzne „bit po bicie", bez konieczności wykonywania operacji na całym kodzie. W trakcie przypisywania wartości bitów do kolejnych kodów stanów automatu należy przestrzegać ograniczenia (9). Algorytm działający na podstawie opisanych powyżej zależności, został przedstawiony w [1].

Algorytm wyżarzania jest algorytmem niedeterministycznym, w którym przypisania kodów odbywają się w sposób losowy. Wymaga zdefiniowania funkcji γ określającej moc wydzieloną przez automat przy przejściu ze stanu a_i do dowolnego stanu (10):

$$nowa(w_{i,j}) = stara(w_{i,j}) \cdot \left(H(c_i, c_j) + 1\right) \quad (10)$$

Algorytm działający na zasadzie wyżarzania został opisany w [6].

W algorytmie sekwencyjnym kodowanie uzależnione jest od przypisanych uprzednio kodów stanów wewnętrznych automatu.

Niech K^R oznacza zbiór wszystkich wartości kodów stanów o długości R bitów. Liczność zbioru K^R wynosi 2^R. Wartość R należy do przedziału [int $\log_2 M$, M].

Poniżej przedstawiono algorytm sekwencyjny kodowania stanów automatu skończonego.

1. Wybierz dwa stany a_i i a_j, dla których $w_{i,j}$ jest największe.
2. Dla stanów a_i oraz a_j przypisz dowolne kody c_i oraz c_j ze zbioru K^R, dla których $H(c_i, c_j) = 1$. Usuń przypisane kody ze zbioru K^R.
3. Powtarzaj kroki 4-5, aż wszystkie stany zostaną zakodowane.
4. Wybierz ze zbioru A stan a_i, któremu nie przypisano kodu i dla którego suma wag krawędzi grafu łączących stan a_i ze stanami, dla których kody są już przypisane, osiąga wartość maksymalną. Jeżeli takich stanów jest kilka wybierz spośród nich ten stan a_i, dla którego $\sum_{j=1}^{M} w_{ij} = min$.
5. Wybierz ze zbioru kodów K^R kod c_i, dla którego funkcja γ ma wartość minimalną. Usuń kod ze zbioru K^R.

4. Badania eksperymentalne

Badania zostały przeprowadzone dla standardowych testów (benchmark) [8] przy założeniu następujących wartości: C = 3pF, f = 5MHz, V_{CC} = 5V oraz $P(x_i = 1) = 0{,}5$. W tab.1 zebrano wyniki dla trzech omówionych algorytmów kodowania oraz dodatkowo dla kodowania binarnego oraz „one-hot". Kolumna „Benchmark" zawiera nazwę układu testowego, w kolumnie „Stany" umieszczono ilość stanów układu testowego, kolumna „Rej." określa ilość rejestrów wykorzystanych w kodowaniu, poza kodowaniem „one-hot", gdzie długość kodu jest równa ilości stanów automatu. Dla każdego sposobu kodowania podano wartości obliczonej mocy („P" w mW) oraz procentowy stosunek tej wartości do wartości maksymalnej („%max").

Tab. 1. Wyniki badań eksperymentalnych algorytmów kodowania stanów wewnętrznych

Tab. 1. Experimental results of the states assignment using different algorithms

Benchmark	Stany	Rej.*	Binarne		One-hot		Kolumnowe		Wyżarzanie		Sekwen-cyjne	
			P	% max	P	% max	P	% max	P	% max	P	% max
bbtas	6	3	134,5	80,9	166,3	100	83,2	50	112,5	67,7	83,2	50
bee-count	7	3	113,3	66,8	169,6	100	108,9	64,2	139,8	82,4	89,4	52,7
cse	16	4	58,1	51,3	85,4	75,5	55,9	49,4	113,1	100	45,0	39,8
dk14	7	3	239,7	77,7	308,6	100	207,3	67,2	252,0	81,7	223,7	72,5
dk15	4	2	186,9	70,5	265,1	100	159,5	60,2	183,9	69,4	159,5	60,2
dk16	27	5	401,1	100	360,8	89,9	377,5	94,1	386,0	96,2	309,1	77,1
dk17	8	3	297,0	94,4	314,6	100	194,1	61,7	299,3	95,2	195,0	62,0
dk512	15	4	298,6	79,6	375	100	319,8	85,3	301,3	80,4	238,8	63,7
ex1	20	5	259,0	100	238,6	92,1	157,7	60,9	223,6	86,3	138,6	53,5
ex2	19	5	250,6	96,4	228,0	87,8	134,7	51,9	259,8	100	134,3	51,7
ex3	10	4	274,7	100	245,1	89,2	172,5	62,8	252,8	92,0	160,4	58,4
ex6	8	3	274,0	91,0	301	100	189,2	62,9	231,7	77,0	199,2	66,2
keyb	19	5	138,6	67,3	205,8	100	121,3	59,0	120,5	58,6	104,4	50,7
lion	4	2	93,8	66,7	140,6	100	70,3	50	93,8	66,7	70,3	50
mark1	15	4	207,3	63,2	281,3	85,7	178,5	54,4	328,1	100	178,3	54,4
mc	4	2	120,5	75	160,7	100	80,4	50	120,5	75	80,4	50
modu-lo12	12	4	171,9	91,7	187,5	100	93,8	50	156,3	83,3	93,8	50
opus	10	4	150,2	61,6	243,9	100	133,4	54,7	144,8	59,4	133,4	54,7
planet	48	6	422,8	100	359,8	85,1	286,1	67,7	317,5	75,1	208,1	49,2
pma	24	5	252,5	100	199,3	78,9	105,6	41,8	141,2	55,9	104,8	41,5
s1	20	5	329,3	100	274,2	83,3	250,7	76,1	268,8	81,6	201,0	61,0
s1488	48	6	72,7	66,6	109,2	100	65,0	59,5	90,0	82,5	65,5	60,0
s1494	48	6	73,1	65,4	109,2	97,7	64,6	57,8	111,8	100	65,7	58,8
s298	218	8	301,2	77,7	282,2	72,8	252,9	65,2	387,9	100	214,0	55,2
s386	13	4	170,3	67,5	252,4	100	162,0	64,2	223,8	88,6	145,5	57,6
sand	32	5	215,6	100	181,2	84,0	130,0	60,3	165,7	76,9	115,4	53,5
sse	16	4	226,0	89,5	252,4	100	162,0	64,2	209,9	83,2	146,5	58,0
styr	30	5	133,0	69,5	191,3	100	112,4	58,7	152,1	79,5	104,5	54,6
tbk	32	5	263,2	100	213,1	81,0	221,3	84,1	246,3	93,6	193	73,3
tma	20	5	158,4	100	130,1	82,2	65,4	41,3	67,4	42,6	65,3	41,2
train11	11	4	101,9	82,0	124,3	100	87,0	70,0	99,2	79,8	63,5	51,1
Średnia			206,1	82,3	224,4	93,1	154,9	61,3	200,0	81,0	139,6	55,9

Z zestawienia przedstawionego w tab.1 wynika, że najniższą średnią wartość mocy dla badanych układów osiągnięto przy zastosowaniu kodowania sekwencyjnego (139,6). Niewiele gorszy wynik (154,9) został osiągnięty przy zastosowaniu kodowania kolumnowego. Najgorsze wyniki dało zastosowanie kodowania „one-hot". Porównując dwa najlepsze algorytmy (kodowanie kolumnowe i sekwencyjne) okazuje się, że kodowanie sekwencyjne jest lepsze w 20 przypadkach, w 6 daje takie same wyniki, natomiast w 5 gorsze (dk14, dk17, ex6, s1488, s1494) w porównaniu do kodowania kolumnowego. Najlepszy wynik dało kodowanie sekwencyjne w 26 przypadkach, w 11 kodowanie kolumnowe.

Tab. 2. Wpływ długości kodu na moc wydzielaną przez układ
Tab. 2. The dependence of the power dissipation on the state code width

Benchmark	2	3	4	5	6	7	8
3-4 stany							
dk15	159,5	155,5	155,5				
lion	70,3	70,3	70,3				
mc	80,4	80,4	80,4				
5-8 stanów							
bbtas		83,2	105,2	105,2	105,2		
beecount		89,4	89,4	89,4	89,4	89,4	
dk14		223,7	222,4	222,4	222,4	222,4	
dk17		195,0	194,1	194,1	194,1	194,1	194,1
ex6		199,2	191,2	191,2	191,2	191,2	191,2

Benchmark	4	5	6	7	8	9	10	11	12	13	14	15	16
9-16 stanów													
cse	45,0	44,7	44,7	44,7	44,7	44,7	44,7	44,7	44,7	44,7	44,7	44,7	44,7
dk512	238,8	232,1	232,1	232,1	232,1	232,1	232,1	232,1	232,1	232,1	232,1	232,1	
ex3	160,4	160,4	160,4	160,4	160,4	160,4	160,4						
ex4	92,0	92,9	93,1	93,1	93,1	93,1	93,1	93,1	93,1	93,1	93,1		
mark1	178,3	174,8	175,2	175,2	174,6	174,6	174,6	174,6	174,6	174,6	174,6	174,6	
modulo12	93,8	109,4	109,4	109,4	125,0	140,6	156,3	156,3	156,3				
opus	133,4	133,3	133,3	133,3	133,3	133,3	133,3						
s386	145,5	145,5	145,5	145,5	145,5	145,5	145,5	145,5	145,5	145,5			
sse	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5	146,5
train11	63,5	63,5	63,5	63,5	63,5	63,5	63,5	63,5					
17-32 stany													
dk16		307,0	304,8	304,8	304,8	304,8	304,8	304,8	304,8	304,8	304,8	304,8	304,8
ex1		138,6	138,6	138,6	138,6	138,6	138,6	138,6	138,6	138,6	138,6	138,6	138,6
ex2		134,3	134,3	134,3	134,3	134,3	134,3	134,3	134,3	134,3	134,3	134,3	134,3
keyb		104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4
pma		104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4
s1		201,0	200,0	200,0	200,0	200,0	200,0	200,0	200,0	200,0	200,0	200,0	200,0
s208		89,1	89,1	89,1	89,1	89,1	89,1	89,1	89,1	89,1	89,1	89,1	89,1
sand		116,3	111,8	109,9	108,8	108,7	108,7	108,7	108,7	108,7	108,7	108,7	108,7
styr		104,5	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4	104,4
tbk		193,0	166,4	161,9	157,4	153,0	149,4	145,9	142,4	138,8	135,3	131,8	128,2
tma		65,3	65,3	65,3	65,3	65,3	65,3	65,3	65,3	65,3	65,3	65,3	65,3
33-64 stany													
planet			217,8	216,3	219,3	228,4	247,0	247,2	247,2	247,2	247,2	247,2	247,2
s1488			65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7
s1494			65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7	65,7
powyżej 64 stanów													
s298					211,6	208,8	207,8	206,3	206,4	204,7	203,6	202,5	201,2

W tab.2 przedstawiono wpływ długości kodu na wartość mocy pobieranej przez układ. Badania zostały przeprowadzone dla algorytmu sekwencyjnego. Kolumna „Benchmark" zawiera nazwę układu testowego. Kolejne kolumny określają długość kodu R, dla której przeprowadzone zostały obliczenia.

Zmiana ilości bitów kodu R od int $\log_2 M$+1 do M pozwoliła na poprawę wyników w 13 przypadkach na 33 (39%). Kodowanie przy R = int $\log_2 M$ dało najlepsze wyniki tylko w 3 przypadkach. W pozostałych przypadkach zmiana wielkości R nie powodowała zmiany poboru mocy.

W 10 przypadkach optymalne rozwiązanie uzyskano przy R = int $\log_2 M$+1. W związku z tym podczas stosowania metody sekwencyjnej w większości przypadków wystarczy przeprowadzić kodowanie dla dwóch wartości R: R = int $\log_2 M$ oraz R = int $\log_2 M$+1, a z uzyskanych wyników wybrać lepszy.

Należy zwrócić również uwagę na przykład tbk, gdzie zwiększenie ilości bitów kodu R z 5 do 16 spowodowało zmniejszenie pobieranej mocy o 33%.

5. Wnioski

Przeprowadzone badania pokazały znaczne różnice w jakości kodowania przeprowadzonego algorytmami: kodowania kolumnowego, wyżarzania i sekwencyjnym. Najlepsze średnie wyniki osiągnął algorytm sekwencyjny, który może stanowić alternatywę dla pozostałych dwóch algorytmów.

Zależność mocy pobieranej przez układ od ilości bitów kodu stanów wewnętrznych okazała się w większości przypadków niewielka, jednakże dla kilku układów zwiększenie kodu pozwoliło na znaczące zmniejszenie ilości pobieranej mocy. W przypadku układu s298 uzyskano zmniejszenie poboru mocy o ok. 5%, natomiast w przypadku układu tbk redukcja mocy sięgnęła 33%.

W dalszych badaniach należy skupić się nad dostosowaniem przedstawionych algorytmów do stosowania w syntezie układów sekwencyjnych realizowanych na strukturach programowalnych.

6. Literatura

[1] Benini L., DeMicheli G., State Assignment for Low Power Dissipation. In IEEE Journal on Solid-state Circuits, Vol. 30, No. 3 (1995), pp. 259-268.

[2] Freitas A. T., Oliveira A. L., Implicit Resolution of the Chapman-Kolmogorov Equations for Sequential Circuits: An Application in Power Estimation. In Proceedings of Design, Automation and Test in Europe Conference and Exhibition (DATE) 2003, pp. 10764-10769.

[3] Grześ T., Salauyou V., Metody obliczania mocy w układach cyfrowych. „Pomiary, Automatyka, Kontrola" nr 7bis (2006), str. 101-102.

[4] Grzes T., Sequential Circuits Power Modeling for Low Power Design. Proceedings of XVI Ukrainian-Polish Conference „CAD in Machinery Design. Implementation and Educational Problems" (CADMD'2006), Polyana, Ukraine, May 22-23, 2006, pp. 54-56.

[5] Pedram M., Power simulation and estimation in VLSI circuits, In "The VLSI Handbook", Edited by W-K. Chen, The CRC Press and the IEEE Press, 1999.

[6] Roy K., Prasad S. C., Circuit Activity Based Logic Synthesis for Low Power Reliable Operations. In IEEE Trans. on VLSI Systems, Vol. 1, No. 4 (1993), pp. 503-513.

[7] Tsui C.-Y., Monteiro J., Pedram M., Devadas S., Despain A. M., Lin B., Power Estimation Methods for Sequential Logic Circuits. In IEEE Trans. on VLSI Systems, Vol. 3, No. 3 (1995), pp. 404-416.

[8] Yang S., Logic Synthesis and Optimization Benchmarks User Guide:Version 3.0. In Technical Report, Microelectronics Center of North Carolina, 1991, 43 p.

Artykuł recenzowany