XVIII Seminarium

ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE' 2008
Oddział Gdański PTETiS
Referat nr 34

WERYFIKACJA CZASÓW OBLICZEŃ HEURYSTYCZNYCH ALGORYTMÓW REDUKCJI POBORU MOCY UKŁADÓW CYFROWYCH CMOS

Władysław SZCZEŚNIAK

Wyższa Szkoła Zarządzania, Wydział Informatyki, ul. Pelplińska 7, 80-335 Gdańsk
tel: 058 5527997 fax: 058 7690869 e-mail: ws205@o2.pl

Streszczenie: W pracy zaprezentowano przeprowadzoną komputerową weryfikację czasów obliczeń piętnastu nowoutworzonych algorytmów heurystycznych dla potrzeb redukcji poboru mocy cyfrowych układów CMOS. W zrealizowanych badaniach eksperymentalnych wykorzystano ogólnodostępne przykłady testowe ISCAS, zaczerpnięte z laboratorium CBL.
Uzyskane wyniki pozwalają na akceptację nowoopracowanych algorytmów redukcji poboru mocy układów CMOS z punktu widzenia ich złożoności obliczeniowej.

Słowa kluczowe: redukcja poboru mocy, cyfrowe układy CMOS, heurystyczne algorytmy redukcji poboru mocy.

1. WPROWADZENIE

Redukcja poboru mocy układów elektronicznych, stosowanych w urządzeniach przenośnych, jest jednym z najważniejszych kryteriów procesu ich projektowania. Dotyczy to w szczególności układów cyfrowych CMOS, stosowanych we współczesnych procesorach. Z literatury [1, 3, 4, 5, 7] znanych jest wiele rozwiązań, pozwalających na obniżenie poziomu poboru mocy pobieranej ze źródła zasilającego, co pozwala na wydłużenie czasu jego pracy.
Każdy z opracowanych algorytmów heurystycznych charakteryzuje się określoną redukcją poboru mocy, która mocno zależy od struktury optymalizowanego układu. Stąd też w literaturze przedmiotu spotyka się tak wiele różnych algorytmów. Z uwagi na to, iż problem ten należy do klasy NP [2] uzyskiwane rozwiązania charakteryzują się suboptymalnością.

W niniejszej pracy przeprowadzono komputerową weryfikację czasów obliczeń utworzonych heurystycznych algorytmów redukcji poboru mocy cyfrowych układów CMOS. Przetestowano 15 różnych połączeń algorytmu optymalizacji i szeregowania zadań, tworzących nowe algorytmy heurystyczne.

2. SFORMUŁOWANIE PROBLEMU

Dla potrzeb omawianego zagadnienia projektowany układ cyfrowy traktowany jest jako algorytm, reprezentowany przez graf przepływu danych *DFG(V,E)*, gdzie: *V* reprezentuje zbiór wierzchołków (operacji) o liczności n ($|V|=n$), a E - zbiór krawędzi (danych przekazywanych pomiędzy operacjami). Cykl operacji reprezentowanych przez graf *G* może być wykonany przy użyciu pewnego zbioru jednostek funkcjonalnych. Aczkolwiek algorytm opisany grafem wykonywany jest przy użyciu ustalonego zbioru jednostek funkcjonalnych *(ang. resources)*, to w danym cyklu zegarowym dostępne są tylko te, które nie wykonują w danej chwili żadnych obliczeń. Pomiędzy cyklami zegarowymi wyniki operacji są przechowywane w rejestrach, które nie są częścią algorytmu. Zadanie polega więc na takim przyporządkowaniu operacji do jednostek funkcjonalnych, aby część z nich mogła pracować przy obniżonym napięciu zasilania - V_{dd}, co oznacza zmniejszenie wydajności obliczeniowej, przy zachowaniu niezmienionej wydajności całego układu. Na przykład, jeśli możliwe jest takie przyporządkowanie operacji do którejś z jednostek funkcjonalnych, aby w co drugim cyklu zegarowym była ona nadal dostępna, to możliwe jest obniżenie jej napięcia zasilania, powodujące dwukrotny spadek jej wydajności, bez zmniejszenia wydajności całego układu/systemu.

Dla zadanego algorytmu (grafu) *DFG(V,E)* w wyniku syntezy wysokiego poziomu uzyskuje się diagram Gantta *SG* [2, 4] (nazywany również w literaturze grafem uszeregowania/szeregowania zadań *(ang. Scheduling Graph)* [4], w którym każdy wierzchołek jest przyporządkowany do odpowiedniej jednostki funkcjonalnej i cyklu zegarowego w taki sposób, że wszystkie zależności od wyników wcześniejszych operacji są zachowane. Każda kolumna *SG* reprezentuje jedną jednostkę funkcjonalną, a każdy wiersz - kolejny takt zegara [7].

Zadanie syntezy wysokiego poziomu jest ograniczone liczbą jednostek funkcjonalnych (np. sumatorów, układów mnożących, bramek logicznych, itd.) oraz dopuszczalnym czasem wykonywania algorytmu (określony jako maksymalna liczba cykli zegarowych) [3, 4]. Jeśli możliwe jest spowolnienie niektórych jednostek funkcjonalnych bez pogarszania wydajności całego systemu, obniżane jest ich napięcie zasilania (V_{dd}), co prowadzi do redukcji poboru mocy. W tym przypadku mocy dynamicznej $Pd \sim V_{dd}^2$, pobieranej ze źródła zasilającego [3, 7].

Recenzent: Dr inż. Grzegorz Blakiewicz – Wydział Elektroniki, Telekomunikacji i Informatyki Politechnika Gdańska

W prezentowanej pracy przeprowadzono obliczenia testowe dla dwóch zbioru jednostek funkcjonalnych, a mianowicie wyznaczonych przy pomocy algorytmów ASAP [3] i MNP [6].

Celem niniejszej pracy jest wyznaczenie czasów obliczeń dla piętnastu nowoutworzonych algorytmów, wyszczególnionych w tablicy 1., będących połączeniem algorytmu optymalizacji i szeregowania zadań.

3. TWORZENIE ZBIORU ALGORYTMÓW HEURYSTYCZNYCH DLA POTRZEB REDUKCJI POBORU MOCY UKŁADÓW CMOS

Dla zwiększenia poziomu redukcji poboru mocy układów cyfrowych CMOS dokonano połączenia algorytmu optymalizacji i algorytmu szeregowania zadań i zastosowano następujące heurystyczne algorytmy optymalizacji:

- R_1 - algorytm sekwencyjnie spowalniający jednostki funkcjonalne, uwzględniający możliwość zamiany operacji pomiędzy jednostkami tego samego typu,
- R_2 - algorytm sekwencyjnie spowalniający jednostki funkcjonalne bez możliwości zamiany operacji pomiędzy jednostkami tego samego typu,
- R_3 - algorytm dokonujący optymalizacji metodą zachłanną

oraz niżej wymienione algorytmy szeregowania zadań:

- S_1 - algorytm charakteryzujący się przydzielaniem zadań możliwie blisko początku harmonogramu,
- S_2, S_3, - algorytmy charakteryzujące się przydzielaniem zadań możliwie blisko końca harmonogramu, różniące się sposobem wyboru jednostki funkcjonalnej,
- S_4 - algorytm charakteryzujący się tworzeniem harmonogramu o większym stopniu pokrewieństwa pomiędzy zadaniami przyporządkowanymi do tej samej jednostki funkcjonalnej,
- S_5 - algorytm charakteryzujący się tworzeniem harmonogramu o bardziej równomiernym obciążeniu jednostek funkcjonalnych tego samego typu.

W tablicy 1. zawarto informację o sposobie tworzenia badanych algorytmów, będących wynikiem połączenia wybranego algorytmu optymalizacji i szeregowania zadań.

Tablica 1. Wyszczególnienie heurystycznych algorytmów optymalizacji i szeregowania będących przedmiotem przeprowadzonych eksperymentów.

Algorytm heurystyczny	Algorytm optymalizacji	Algorytm szeregowania
DIS	R_1	S_1
DIL	R_1	S_2
DIH	R_1	S_3
DIM	R_1	S_4
DIU	R_1	S_5
DCS	R_2	S_1
DCL	R_2	S_2
DCH	R_2	S_3
DCM	R_2	S_4
DCU	R_2	S_5
DRS	R_3	S_1
DRL	R_3	S_2
DRH	R_3	S_3
DRM	R_3	S_4
DRU	R_3	S_5

Występujące tutaj algorytmy R_1-R_5 i S_1-S_5 są dobrze znane z literatury dotyczącej zagadnień redukcji poboru mocy [1, 3, 5, 6, 7], syntezy wysokiego poziomu [3, 4] i szeregowania zadań [2, 3].

I tak na przykład wymieniony w tablicy 1. algorytm DIS jest połączeniem algorytmu optymalizacji R_1 (w poprzednich pracach autora nazywany IIOI [7]) z algorytmem szeregowania zadań S_1, w literaturze występuje jako ASAP [3].

4. WYNIKI EKSPERYMENTALNE

Poniżej zaprezentowano wyniki badań eksperymentalnych dla algorytmów wymienionych w tablicy 1.

Wspomniane piętnaście algorytmów heurystycznych dla potrzeb redukcji poboru mocy cyfrowych układów CMOS zaimplementowano w języku Java.

Algorytmy testowano przy pomocy przykładowych układów (benchmarków), pobranych z laboratorium CBL [8].

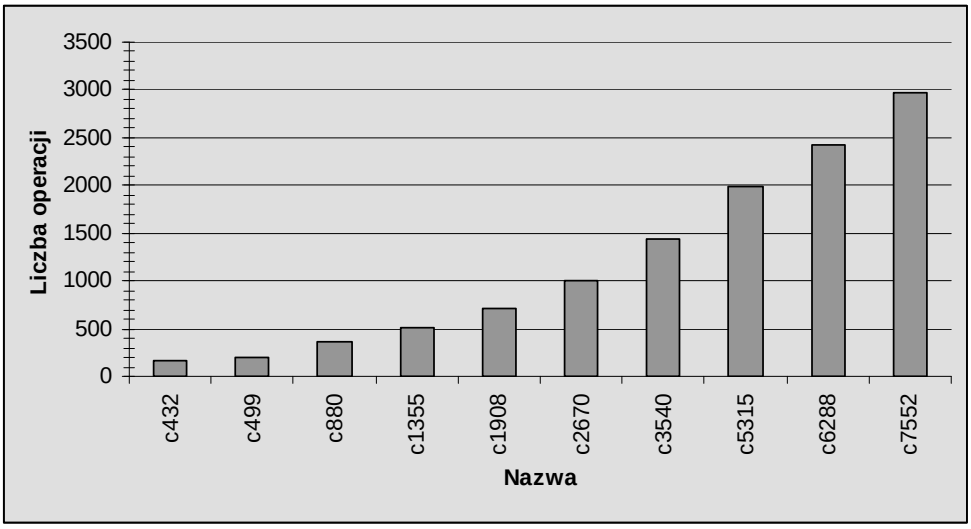
Dla zbadania zaprezentowanych algorytmów w możliwie szerokim zakresie przypadków testowych wybrano 10 przykładów o różnym stopniu złożoności. Najprostszy z układów testowych (c432) zawierał 160 operacji, podczas gdy najbardziej złożony - ponad 16 razy więcej (c7552: 2678 operacji). Liczby operacji dla wszystkich wykorzystanych przykładów testowych przedstawiono na rysunku 1.

Przykłady testowe pobrane z laboratorium CBL musiały zostać poddane modyfikacjom ze względu na wymagania algorytmów szeregowania zadań. Algorytmy te wymagają definicji powiązań między operacjami w postaci grafu acyklicznego. Oryginalne układy testowe reprezentowały układy z przerzutnikami, a więc nie spełniały tego warunku. Modyfikacja, której poddano przykłady testowe, polegała na usunięciu wszystkich przerzutników. Połączenia z operacji do przerzutnika zastąpiono przy tym połączeniami do wyjść systemowych, a połączenia z przerzutnika do operacji zastąpiono połączeniami z wejść systemowych.

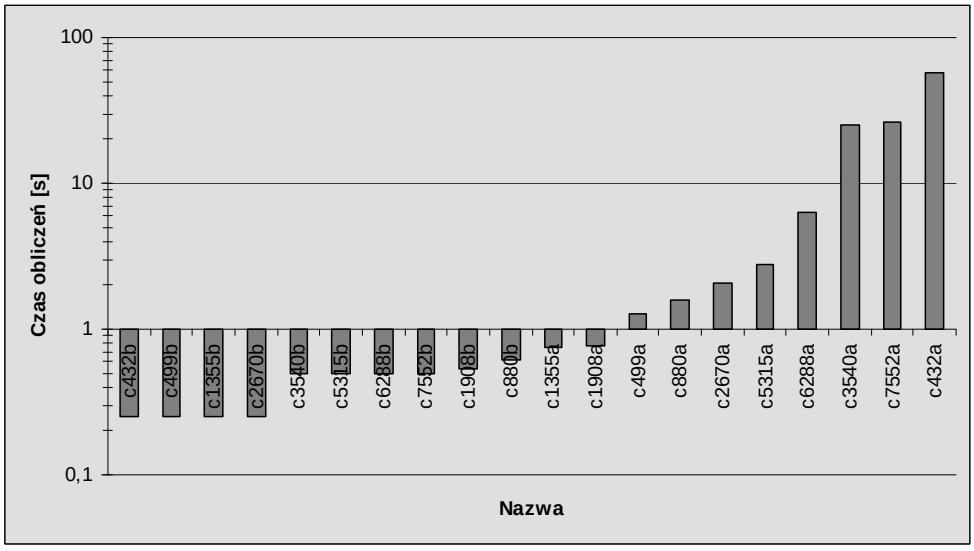
Dodatkowo, dla pełniejszego zbadania przedstawionych algorytmów dla każdego przykładu testowego użyto dwóch zestawów jednostek funkcjonalnych, oznaczonych jako (a) i (b). Liczba jednostek funkcjonalnych w zestawach typu (a) określana była za pomocą algorytmu ASAP [3]. Na skutek tego zestawy typu (a) mogą zawierać pewną nadmiarowość jednostek funkcjonalnych. W celu sprawdzenia przedstawionych algorytmów w warunkach bardziej ograniczonej liczby jednostek funkcjonalnych dla każdego przypadku testowego utworzono zestaw (b). Liczba jednostek funkcjonalnych w zestawach typu (b) określana była za pomocą algorytmu MNP (ang. *Minimization the Number of Procesing elements)* [6]. Spośród wszystkich zestawów jednostek funkcjonalnych (typu a i b łącznie) minimalny zbiór miał liczność 21, a maksymalny - 569. Należy tutaj zaznaczyć, iż redukcja poboru mocy dla zestawów typu (b) jest zadaniem znacznie trudniejszym ze względu na dużo większą liczbę operacji przyporządkowanych do jednostki funkcjonalnej, a co za tym idzie dużo mniejszą liczbę wolnych cykli zegara.

Uzyskane minimalne i maksymalne czasy obliczeń dla algorytmu DIS przedstawiono na rysunku 2. Indeksem *a* zaznaczono tutaj czasy obliczeń w przypadku wyznaczenia zbioru jednostek funkcjonalnych algorytmem ASAP, natomiast *b* MNP.

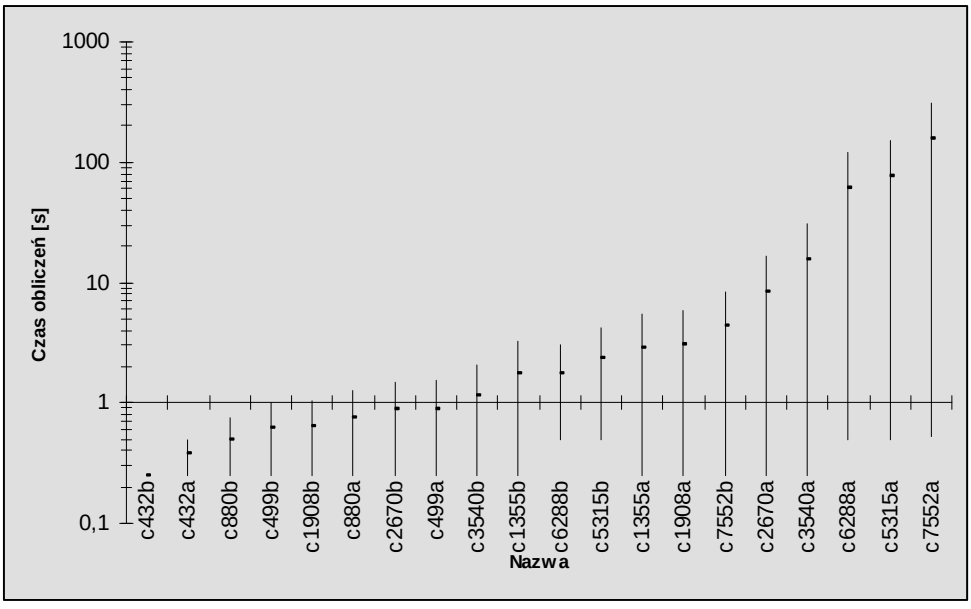
Wskaźniki minimalnej i maksymalnej redukcji poboru mocy (zdefiniowany jako względne, procentowe zmniejszenie poboru mocy pobieranej ze źródła zasilającego) dla badanych algorytmów zawierał się



Rys.1. Liczba operacji w zastosowanych przykładach testowych



Rys. 2. Czasy obliczeń dla przykładów testowych dla algorytmu DIS



Rys. 3. Minimalny i maksymalny czasy obliczeń dla algorytmów wymienionych w tablicy 1. dla rozpatrywanych przykładów testowych

w granicach od 5 do 60 procent i był w dużym stopniu zależny od struktury grafu *DFG* (przykładu testowego).
Należy tutaj zaznaczyć, iż stosując określone algorytmy zaobserwowano różne wartości wskaźnika redukcji poboru mocy dla rozpatrywanych przykładów testowych.
Wyznaczone czasy obliczeń przedstawia rys. 3.

5. WNIOSKI

Przeprowadzone badania utworzonych algorytmów heurystycznych pozwalają stwierdzić, iż połączenie algorytmu optymalizacji i szeregowania zadań daje dodatkową redukcję poboru mocy w akceptowalnym czasie obliczeń. Uzyskane zależności czasowe potwierdzają również wyniki analityczne wyznaczone jako wypadkowe złożoności obliczeniowe rozpatrywanych algorytmów.

Zmiana przykładu testowego przy przeprowadzaniu optymalizacji wszystkimi piętnastoma algorytmami prowadziła do zmiany algorytmu dającego najlepsze rozwiązanie. Nie pozwoliło to na wytypowanie jednego algorytmu gwarantującego największą redukcję poboru mocy. W celu najlepszego wykorzystania opracowanych algorytmów należy wprowadzić ich adaptacyjny dobór, zależny od określonych cech charakterystycznych projektowanego (optymalizowanego) układu.

6. BIBLIOGRAFIA

1. Benini L., Bogliolo A., De Micheli G.: A survey of design techniques for system-level dynamic power management. IEEE Trans. on Very Large Scale Integration (VLSI) Systems, vol. 8 3, s. 299–316, June 2000, ISSN 1063-8210
2. Giaro K., Szcześniak W.: Formalizm i metody szeregowania zadań dla potrzeb redukcji poboru mocy cyfrowych układów CMOS, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki PG, Gdańsk 2006, Nr 22, s. 55 - 62, ISSN 1425-5766
3. Rabaey J. M., Pedram M. (Eds.): Low Power Design Methodologies, Kluwer Academic Publishers, 1996
4. Schmitz M.T., Al-Hashimi B.M., Eles P.: System-Level Techniques for Energy-Efficient Embedded Systems, Kluwer Academic Publishers, 2004, ISBN 1-4020-7750-5
5. Szcześniak W., Szcześniak P.: Algorytmiczne metody redukcji poboru mocy w cyfrowych układach CMOS, II Krajowa Konferencja Technologie Informacyjne, Zeszyty Naukowe Wydziału ETI Politechniki Gdańskiej, Gdańsk 2004, s. 859-866. ISBN 83-917681-5-5
6. Szcześniak P., Szcześniak W.: Dobór optymalnej liczby jednostek funkcjonalnych dla realizacji syntezy wysokiego poziomu układów cyfrowych, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Nr 21, Gdańsk 2005, s. 237-245, ISSN 1425-5766
7. Szcześniak W., Voss B., Theisen M., Becker J., Glesner M.: Influence of high-level synthesis on average and peak temperatures of CMOS circuits, Microelectronics Journal, vol. 32, s. 855-862, Oct. 2001, ISSN 0026-2692
8. Collaborative Benchmarking Laboratory, www.cbl.ncsu.edu

COMPUTATIONAL TIME VERIFICATION OF HEURISTIC ALGORITHMS FOR LOW POWER DESIGN OF CMOS CIRCUITS

Keywords: low power design, digital CMOS circuits, heuristic low power design algorithms

This paper presents a computer verification of computational complexity of 15 newly elaborated heuristic algorithms for low power design of digital CMOS circuits. The verified algorithms were tested against a set of commonly available ISCAS benchmarks from CBL laboratory. The computational complexities of the tested heuristic algorithms were verified experimentally.