

Valery SALAUYOU, Marek GRUSZEWSKIPOLITECHNIKA BIAŁOSTOCKA, WYDZIAŁ INFORMATYKI,
ul. Wiejska 45A, 15-351 Białystok**Komparatory hierarchiczne – metody opisu, wyniki syntezy****Dr hab. inż. Valery SALAUYOU**

Dr hab. inż. Valery Salauyou w latach 1980-1984 pracował jako programista w Mińsku na Białorusi. W latach 1984-2002 był pracownikiem dydaktycznym w Białoruskim Państwowym Uniwersytecie Informatyki i Radioelektroniki w Mińsku, gdzie uzyskał tytuł doktora habilitowanego nauk technicznych. Jednocześnie od 1992 roku pracuje jako adiunkt Wydziału Informatyki Politechniki Białostockiej. Zainteresowania naukowe: projektowanie systemów cyfrowych na układach programowalnych.



e-mail: valsol@mail.ru

Mgr inż. Marek GRUSZEWSKI

Ukończył studia na Wydziale Elektroniki Politechniki Warszawskiej w 1984 r. W latach 1984-1991 zatrudniony był w PDP PAN „SONOPAN”. W 1992 r. rozpoczął pracę w Instytucie Informatyki Politechniki Białostockiej. Od 2001 r. do 2013 r. kierował Laboratorium Informatyki Technicznej. Obecnie zatrudniony jest na Wydziale Informatyki PB w Laboratorium Wydziału Informatyki na stanowisku starszego specjalisty ds. informatyki. Zainteresowania naukowe: synteza układów cyfrowych.



e-mail: m.gruszewski@pb.edu.pl

Streszczenie

Praca poświęcona jest problematyce syntezy komparatorów binarnych w strukturach CPLD/FPGA. Opracowano metody opisu komparatorów w postaci piramidalnych struktur hierarchicznych. Do budowy komparatorów wykorzystano język Verilog i edytor graficzny. Badania eksperymentalne wykonano dla komparatorów 64-bitowych w środowisku Quartus II firmy Altera. Przeprowadzone badania wykazały, że istnieją struktury hierarchiczne, które są bardziej efektywne od wbudowanej funkcji *lpm_compare* pakietu Quartus II. W najlepszym przypadku uzyskano zmniejszenie maksymalnego czasu propagacji o 44%.

Słowa kluczowe: komparator binarny, język Verilog, komparator o strukturze hierarchicznej, funkcja *lpm_compare*, struktury CPLD/FPGA.

Hierarchical comparators – description methods, synthesis results**Abstract**

The paper deals with the problem of binary comparator synthesis in CPLD/FPGA structures. Comparators were built with the usage of the Verilog language and the Quartus II graphics editor [10]. Section 1 describes the notion of a digital comparator, its basic usage [1-4] and research directions [6-10]. Section 2 presents the general hierarchical structure of the comparator (Fig. 2). Section 3 describes the method of building new hierarchical structures of 64-bit comparators. Section 4 presents the results of experimental research. Comparators were built and tested in the Altera Quartus II environment. In the experimental research, the 64-bit hierarchical comparators were compared with the 64-bit comparator built with the direct usage of the *lpm_compare* library function of the Quartus II package. The research was conducted on three CPLD families (MAX 3000 A, MAX II and MAX V) and two FPGA families (Cyclone III and Arria II GX). Three parameters were compared: implementation cost, maximum propagation delay and overall power dissipation. The conducted research demonstrates the existence of hierarchical structures which are better than the in-built *lpm_compare* function. For the MAX 3000 A family, the implemented hierarchical methods of comparator synthesis show the improved results: 32% in the implementation cost, 44% in the maximum propagation delay and 18% in the overall power dissipation. The improved results for Arria II are as follows: 17% in the implementation cost and 26% in the maximum propagation delay.

Keywords: binary comparator, Verilog language, hierarchical comparator, *lpm_compare* function, CPLD/FPGA structures.

1. Wprowadzenie

Komparator binarny (nazywany dalej komparatorem) należy do układów kombinacyjnych i służy do porównywania wartości dwóch słów binarnych, np. A i B. Komparator tradycyjny ma trzy wyjścia i realizuje następujące funkcje: G ($A > B$), E ($A = B$), L ($A < B$), gdzie $A = a_n \dots a_1$ oraz $B = b_n \dots b_1$.

Komparatory należą do podstawowych komponentów systemów cyfrowych. Są kluczowymi elementami w szerokim zakresie zastosowań, tj.: w procesach obliczeniowych (grafika oraz

przetwarzanie obrazów/sygnalów [1]), w układach testujących (analizatory sygnałów, wbudowane układy samotestujące [2]), w procesach poszukiwania i sortowania danych [3], jako komponenty w procesorach ogólnego przeznaczenia: pamięci asocjacyjne (skojarzeniowe), bufor TLB (*Translation Lookaside Buffer*), bufor BTB (*Branch Target Buffer*) i wiele innych bloków porównywania argumentów w CPU [4].

Rozbudowany układ cyfrowy na ogół przedstawiany jest jako zespół standardowych i/lub oryginalnych bloków funkcjonalnych. Do takich bloków funkcjonalnych należą m.in. komparatory. Najbardziej rozpowszechnioną dzisiaj bazą elementową techniki cyfrowej są złożone programowalne układy logiczne (CPLD) oraz bezpośrednio programowalne macierze bramek (FPGA) [5].

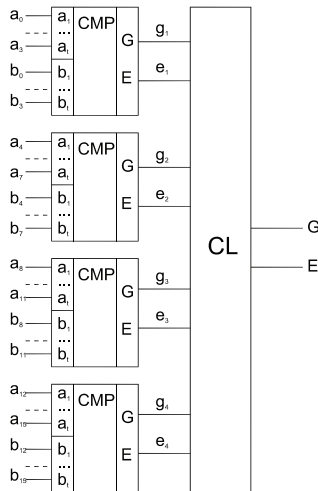
W technice obliczeniowej zauważa się stałą tendencję do zwiększania wielkości słów binarnych. Długość słów rośnie szczególnie szybko w systemach telekomunikacji, a także w urządzeniach przetwarzania i przesyłania informacji. Przy projektowaniu systemów cyfrowych pojawia się więc potrzeba opracowania efektywnych metod syntezy komparatorów w strukturach CPLD/FPGA, pracujących ze słowami binarnymi o dużych rozmiarach (komparatorów dużych rozmiarów). Przy syntezie komparatorów wystarczy zrealizować tylko dwie funkcje G i E . Funkcja L zawsze może być określona na podstawie dwóch pierwszych na podstawie zależności:

$$L = \bar{G} \cdot \bar{E}. \quad (1)$$

Istnieje wiele rodzajów komparatorów binarnych oraz metod ich syntezy, klasycznych oraz nowatorskich. W ostatnich latach zwraca się dużą uwagę na projektowanie komparatorów o dużej szybkości działania, niskim koszcie realizacji i małym poborze mocy. Przykłady wydajnych architektur komparatorów zaprezentowano w pracach [6-10]. W pracy [10] przedstawiono metody syntezy komparatorów z wykorzystaniem języka Verilog: równoległe, szeregowo, równoległo-szeregowo, a także równoległo-szeregowo z wykorzystaniem edytora graficznego. W niniejszej pracy przedstawione są wyniki syntezy hierarchicznych komparatorów w strukturach CPLD/FPGA. Wyniki syntezy porównano z biblioteczną makrofunkcją *lpm_compare* środowiska Quartus II firmy Altera.

2. Hierarchiczne struktury komparatorów

Hierarchiczna budowa komparatora dużego rozmiaru ma postać piramidalnej struktury złożonej z bloków komparatorów mniejszego rozmiaru. Każdy blok piramidalnej struktury może zawierać z kolei inną hierarchiczną strukturę. W ten sposób można utworzyć wiele poziomów hierarchii oraz budować komparatory dowolnego rozmiaru. Na rys. 1 przedstawiono przykład komparatora 16-bitowego zbudowanego z czterech bloków 4-bitowych.



Rys. 1. Hierarchiczna struktura komparatora 16-bitowego zbudowanego z bloków 4-bitowych

Fig. 1. Hierarchical structure of a 16-bit comparator built of 4-bit blocks

Wyjściowe funkcje G oraz E generuje układ logiczny CL na podstawie następujących wyrażeń:

$$G = g_4 + e_4 \cdot g_3 + e_4 \cdot e_3 \cdot g_2 + e_4 \cdot e_3 \cdot e_2 \cdot g_1, \quad (2)$$

$$E = e_1 \cdot e_2 \cdot e_3 \cdot e_4. \quad (3)$$

Rozbudowując wyrażenia (2) i (3) dla F bloków otrzymano uogólnione funkcje G oraz E :

$$G = g_F + e_F \cdot g_{F-1} + e_F \cdot e_{F-1} \cdot g_{F-2} + \dots + e_F \cdot e_{F-1} \cdot \dots \cdot e_2 \cdot g_1, \quad (4)$$

$$E = e_1 \cdot \dots \cdot e_F. \quad (5)$$

Na podstawie wyrażeń (4) i (5) zbudowano układy kombinacyjne CL dla wszystkich rozpatrywanych hierarchicznych struktur komparatorów.

3. Zastosowanie hierarchicznej metody do syntezy komparatorów 64-bitowych

Podczas realizacji komparatorów w strukturach CPLD/FPGA za pomocą hierarchicznej metody syntezy pojawiają się następujące zagadnienia:

- określenie najlepszej metody syntezy bloków pierwszego poziomu;
- określenie najlepszej struktury hierarchicznej dla każdej klasy układów programowalnych.

Do rozwiązania powyższych zagadnień przyjęto metodę badań eksperymentalnych. Do syntezy komparatorów pierwszego poziomu wykorzystano dwie metody, które wykazały wysoką efektywność podczas badań eksperymentalnych, przeprowadzonych w pracy [10]. Pierwsza metoda - M2, wykorzystuje do opisu komparatora blok *always* w języku Verilog. Druga metoda - *lpm*, to wbudowana makrofunkcja *lpm_compare* pakietu Quartus II.

Do określenia najlepszej struktury hierarchicznej zbudowano w środowisku Quartus II różne struktury hierarchiczne komparatorów 64-bitowych. Jako najmniejsze bloki pierwszego poziomu przyjęto komparatory 4-bitowe.

Hierarchiczne struktury komparatorów 64-bitowych można w takim przypadku budować z komparatorów (bloków) mniejszego rozmiaru o szerokości 4, 8, 16 i 32 bitów. Do oznaczenia hierarchicznej struktury komparatora przyjęto następujące wyrażenie:

$$M_NxT_metoda[LC], \quad (6)$$

gdzie: M - ilość bitów bloku drugiego stopnia, $M = NxT$; N - ilość bloków pierwszego stopnia; T - ilość bitów bloku pierwszego stopnia; *metoda* - sposób syntezy pierwszego stopnia; *LC* - zastosowanie buforów LCELL na wyjściach bloków pierwszego stopnia. Podczas budowy następnego stopnia wyrażenie (6) powtarza się (tylko w miejsce wyrażenia T_metoda wstawia się wyrażenie bloku niższego stopnia).

Bufer LCELL jest buforem z jednym wejściem i jednym wyjściem. Zastosowanie takich buforów na wyjściach komparatorów pierwszego stopnia powoduje, że dany punkt układu obowiązkowo będzie realizowany na wyjściu makrokomórki CPLD/FPGA. Za pomocą buforów LCELL można dokonywać dekompozycji złożonych układów, co w niektórych przypadkach powoduje zmniejszenie ich kosztu realizacji oraz zwiększenie szybkości działania. Na przykład zapis *64_4x16_4x4_lpm_LC* oznacza hierarchiczną strukturę komparatora 64-bitowego, która zbudowana jest z czterech bloków 16-bitowych. Bloki te składają się z 4 kolejnych bloków 4-bitowych pierwszego stopnia zrealizowanych za pomocą metody (funkcji) *lpm_compare*. Na wyjściach tych bloków (komparatorów) zastosowano bufory LCELL. Oznaczenia części zastosowanych w badaniach struktur zamieszczono w opisach odpowiednich tabel z wynikami pomiarów (tabela 1-2).

4. Badania eksperymentalne hierarchicznych struktur komparatorów 64-bitowych

Efektywność hierarchicznych struktur komparatorów 64-bitowych badano za pomocą pakietu Quartus II wersja 13.0 sp1 Web Edition (64-bit) oraz 13.1 Web Edition (64-bit) firmy Altera. Badania przeprowadzono dla trzech rodzin CPLD (MAX 3000A, MAX II i MAX V) oraz dla dwóch rodzin FPGA (Cyclone III i Arria II GX). W tym celu porównano wszystkie komparatory, zbudowane w oparciu o opisane wcześniej struktury hierarchiczne z komparatorem 64-bitowym, który zbudowano z bezpośrednim wykorzystaniem bibliotecznej parametryzowanej makrofunkcji *lpm_compare* firmy Altera (*64_lpm*). Jako kryteria porównawcze przyjęto: koszt realizacji C mierzony ilością wykorzystanych elementów logicznych CPLD/FPGA, maksymalny czas propagacji D z wejścia na wyjście komparatora mierzony w nanosekundach [ns] oraz całkowite rozproszenie mocy P mierzone w miliwatach [mW].

Praca przedstawia tylko część wyników badań ze względu na bardzo dużą ilość eksperymentów. W tabeli 1 przedstawiono wyniki badań eksperymentalnych komparatora *64_lpm* oraz hierarchicznych struktur komparatorów 64-bitowych, w których do syntezy bloków pierwszego poziomu wykorzystano metodę M2, a w tabeli 2 wyniki badań tych samych struktur hierarchicznych tylko z buforami LCELL na wyjściach komparatorów pierwszego poziomu.

Na podstawie uzyskanych wyników dokonano wyboru najmniejszych wartości poszczególnych parametrów oraz odpowiadających im struktur hierarchicznych i porównano je z wartościami tych parametrów uzyskanymi bezpośrednio za pomocą funkcji *lpm_compare*. W grupie układów CPLD największą poprawę parametrów uzyskano dla rodziny MAX 3000A: C - 32%, D - 44%, P - 18%. W grupie układów FPGA największą poprawę parametrów uzyskano dla rodziny Arria II: C - 17%, D - 26%.

Porównano również hierarchiczne struktury komparatorów 64-bitowych lepsze od komparatora *64_lpm*. Dobre struktury pod względem ilości lepszych wyników to: *64_4x16_M2* - cztery lepsze wyniki, *64_4x16_4x4_lpm*, *64_4x16_2x8_2x4_lpm* - trzy lepsze wyniki. Najlepsza struktura hierarchiczna ma konfigurację *64_4x16_* - osiem lepszych wyników. Dobre struktury ze względu na koszt realizacji mają konfigurację: *64_4x16_4x4_*, *64_4x16_2x8_2x4_*, *64_2x32_8x4_* - pięć lepszych wyników. Dobre struktury ze względu na szybkość działania mają konfigurację: *64_4x16_* - sześć lepszych wyników, *64_2x32_* - pięć lepszych wyników.

Tab. 1. Wyniki badań eksperymentalnych komparatora 64_lpm oraz 64-bitowych komparatorów o strukturze hierarchicznej, w których do syntezy bloków pierwszego poziomu wykorzystano metodę M2

Tab. 1. Results of experimental research on 64_lpm comparator and 64-bit hierarchical comparators in which first level block synthesis was conducted with the usage of the M2 method

CPLD/FPGA	Par.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	64_lpm
MAX 3000A	C	92	92	104	80	86	88	103	81	82	98	99	86	92	85	105	117
	D	29	29	29	29	29	29	29	64	29	29	29	29	29	29	30	36
	P	291	267	311	250	272	262	303	252	270	275	302	259	281	256	308	304
MAX II	C	110	113	116	110	106	113	106	108	112	112	112	110	110	113	110	107
	D	17	13	16	14	15	13	15	12	16	14	17	14	17	13	17	12
	P	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46
MAX V	C	110	113	116	110	106	113	106	108	112	112	112	110	110	113	110	107
	D	32	24	36	25	28	24	30	22	36	25	35	23	31	25	31	21
	P	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07
Cyclone III	C	110	112	109	110	109	112	109	108	103	113	103	110	115	112	115	107
	D	19	14	18	12	15	14	15	13	16	16	16	14	17	14	17	13
	P	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67
Arria II GX	C	82	72	95	85	86	77	86	86	87	77	87	86	86	75	86	83
	D	20	14	20	17	22	14	21	19	18	20	18	18	16	21	16	19
	P	332	332	332	332	332	332	332	332	332	332	332	332	332	332	332	332

1. 64_16x4_M2, 2. 64_8x8_M2, 3. 64_8x8_2x4_M2, 4. 64_4x16_M2, 5. 64_4x16_4x4_M2, 6. 64_4x16_2x8_M2, 7. 64_4x16_2x8_2x4_M2, 8. 64_2x32_M2, 9. 64_2x32_8x4_M2, 10. 64_2x32_4x8_M2, 11. 64_2x32_4x8_2x4_M2, 12. 64_2x32_2x16_M2, 13. 64_2x32_2x16_4x4_M2, 14. 64_2x32_2x16_2x8_M2, 15. 64_2x32_2x16_2x8_2x4_M2

Tab. 2. Wyniki badań eksperymentalnych komparatora 64_lpm oraz 64-bitowych komparatorów o strukturze hierarchicznej, w których do syntezy bloków pierwszego poziomu wykorzystano metodę M2 z buforami LCELL

Tab. 2. Results of experimental research on 64_lpm comparator and 64-bit hierarchical comparators in which first level block synthesis was conducted with the usage of the M2 method with LCELL buffers

CPLD/FPGA	Par.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	64_lpm
MAX 3000A	C	85	90	85	89	85	90	85	112	85	96	85	89	85	99	85	117
	D	20	29	20	29	20	29	20	36	20	29	20	29	20	29	20	36
	P	298	264	298	263	298	264	298	297	298	273	298	263	298	277	298	304
MAX II	C	119	120	112	115	111	120	111	110	112	120	112	115	111	120	111	107
	D	15	12	15	12	18	14	20	12	13	13	13	12	17	13	19	12
	P	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46	46
MAX V	C	119	120	112	115	111	120	111	110	112	120	112	115	111	120	111	107
	D	25	22	26	23	35	23	37	22	25	25	26	22	34	25	40	21
	P	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07	0,07
Cyclone III	C	115	120	114	115	111	120	111	110	116	121	116	115	111	121	111	107
	D	14	14	16	13	18	17	19	12	13	14	15	13	14	14	18	13
	P	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67	67
Arria II GX	C	84	72	78	83	74	72	74	84	84	69	84	83	74	69	74	83
	D	23	17	17	14	17	15	16	17	14	14	14	17	17	14	17	19
	P	332	332	332	332	332	332	332	332	332	332	332	332	332	332	332	332

1. 64_16x4_M2_LC, 2. 64_8x8_M2_LC, 3. 64_8x8_2x4_M2_LC, 4. 64_4x16_M2_LC, 5. 64_4x16_4x4_M2_LC, 6. 64_4x16_2x8_M2_LC, 7. 64_4x16_2x8_2x4_M2_LC, 8. 64_2x32_M2_LC, 9. 64_2x32_8x4_M2_LC, 10. 64_2x32_4x8_M2_LC, 11. 64_2x32_4x8_2x4_M2_LC, 12. 64_2x32_2x16_M2_LC, 13. 64_2x32_2x16_4x4_M2_LC, 14. 64_2x32_2x16_2x8_M2_LC, 15. 64_2x32_2x16_2x8_2x4_M2_LC

5. Wnioski

Wyniki badań eksperymentalnych wykazały możliwość zbudowania komparatorów dużych rozmiarów, które dla niektórych rodzin układów CPLD/FPGA mają lepsze parametry (koszt realizacji, maksymalny czas propagacji, całkowite rozproszenie mocy) od parametrów komparatora z bezpośrednim wykorzystaniem funkcji *lpm_compare* (64_lpm). W szczególności do grupy najlepszych struktur należą: 64_4x16_M2, 64_4x16_4x4_lpm, 64_4x16_2x8_2x4_lpm.

Przedstawioną metodę syntezy można wykorzystywać dalej do budowy i badań komparatorów 128-bitowych i większych. Można też zbudować i przebadать struktury hierarchiczne, w których najmniejsze bloki pierwszego poziomu będą 8-bitowe lub 16-bitowe.

6. Literatura

- [1] Parhami B.: Efficient hamming weight comparators for binary vectors based on accumulative and up/down parallel counters, IEEE Trans. Circuits Syst., vol. 56, no. 2, p. 167–171, 2009.
- [2] Jarmolik W., Gruszewski M.: Nowy sposób projektowania uniwersalnego modułu do samotestowania układów hybrydowych, Elektronika, nr 4, s. 26–28, 2001.

- [3] Cheng S.W.: Arbitrary Long Digit Sorter HW/SW Co-Design, Proceedings of IEEE Asia and South Pacific Design Automation Conference, p. 538–543, 2003.
- [4] Suzuki H., Kim C. H., Roy K.: Fast tag comparator using diode partitioned domino for 64-bit microprocessor, IEEE Trans. Circuits Syst. I, vol. 54, no. 2, p. 322–328, 2007.
- [5] Solov'ev V. V.: Proektirovanie cifrovyyh sistem na osnove programmirovemykh logičeskikh integral'nyh shem, Moskva, Gorčaćaã liniã - Telekom, s. 636, 2001.
- [6] Chuang P., Li D., Sachdev M.: A Low-Power High-Performance Single-Cycle Tree-Based 64-Bit Binary Comparator IEEE Transactions on Circuits And Systems-II: Express Briefs, vol. 59, no. 2, 2012.
- [7] Deb S., Chaudhury S.: High-Speed Comparator Architectures for Fast Binary Comparison, Third International Conference on Emerging Applications of Information Technology (EAIT), p. 454–457, 2012.
- [8] Deb S.: A Novel Architecture for Binary Comparison Using Time Division De-multiplexing Technique, Third International Conference on Emerging Applications of Information Technology (EAIT), p. 478–482, 2012.
- [9] Hauser A., Chichester I.: High-Speed 64-Bit Binary Comparator using Two Stages, European Journal of Engineering and Innovation, vol. 11, 2013.
- [10] Gruszewski M.: Metody syntezy komparatorów z wykorzystaniem języka Verilog w środowisku Quartus II, Elektronika, nr 1, s. 72–77, 2014.

otrzymano / received: 12.04.2014

przyjęto do druku / accepted: 02.06.2014

artykuł recenzowany / revised paper