Akademia Górniczo – Hutnicza
im. Stanisława Staszica w Krakowie
Wydział Elektrotechniki, Informatyki, Automatyki i Elektroniki
Katedra Elektroniki

Autoreferat rozprawy doktorskiej

Implementacja w układach FPGA wybranych operacji zmiennoprzecinkowych

mgr inż. Maciej Wielgosz

Promotor: Prof. dr hab. inż. Kazimierz Wiatr

Kraków, 2010

Wstęp

Systemy komputerowe o wielkich mocach obliczeniowych umożliwiają obecnie realizację obliczeń w oparciu o układy logiki rekonfigurowanej FPGA (ang. *Field Programmable Gate Array*), których pojemność i szybkość znacząco wzrosła w ostatnim czasie i układy te śmiało konkurują z innymi rozwiązaniami HPC (ang. *High Performance Computing*) tworząc podwaliny nowej dziedziny obliczeń HPRC (ang. *High Performance Reconfigurable Computing*).

Zastosowanie układu o elastyczniej strukturze wewnętrznej umożliwia dostosowanie architektury systemu obliczeniowego do aktualnie realizowanego algorytmu, co w kontekście tej pracy oznacza m.in. możliwość implementacji szerokiej gamy pośrednich formatów zapisu liczb zmiennoprzecinkowych. Takie podejście wpływa bezpośrednio na możliwy do uzyskania zakres precyzji wykonywania operacji oraz reprezentacji danych, krytyczny w obliczeniach naukowo-technicznych ze względu na wymaganą dużą dokładność obliczeń. Możliwość płynnego dobory precyzji obliczeń na poszczególnych etapach przetwarzania prowadzi również do znacznej oszczędności zajmowanych zasobów i pozwala jednocześnie uzyskać przyspieszenie wykonywania operacji.

Cel i teza pracy

W ostatnim czasie powstało wiele modułów zmiennoprzecinkowych dedykowanych dla logiki rekonfigurowalnej. Są to jednak zazwyczaj realizacje pojedynczych, niewielkich funkcji matematycznych [1, 2, 3, 4, 5], których implementacja i testy porównawcze z innymi rozwiązaniami (procesory wielordzeniowe, karty graficzne, układy ASIC) nie dają miarodajnej informacji, co do potencjału rozwiązań opartych o logikę programowalną w systemach HPC. W przypadku umieszczenia niewielkich modułów w strukturach akceleratora sprzętowego dość znacznie uwidaczniają się koszty związane z transferem danych i podziałem algorytmu. Dlatego zrodził się pomysł kompletnej implementacji w układach FPGA większego algorytmu, tak by w pełni ocenić korzyści wynikające ze sprzętowej realizacji operacji zmiennoprzecinkowych. W tym celu zaprojektowane i zaimplementowane zostały własne moduły sprzętowe, które albo nie są dostępne komercyjnie albo też umożliwią uzyskanie większej wydajności obliczeniowej.

Ważne jest, aby wybrana do sprzętowej realizacji część większego algorytmu, stanowiła obliczeniowo znaczący fragment oraz charakteryzowała się cechami predysponującymi ją do implementacji w strukturach logiki programowalnej. Wybrany fragment algorytmu kwantowo-chemicznego, którym jest generacja funkcji orbitalu atomowego [6] jest dobrym kandydatem do akceleracji z wykorzystaniem układów FPGA, gdyż stanowi on jądro absorbujące stosunkowo dużą moc obliczeniową.

Celem pracy jest sprawdzenie możliwości realizacji zaawansowanych obliczeń zmiennoprzecinkowych w układach logiki reprogramowalnej. Należy, zatem zaadoptować wybrane istniejące algorytmy zmiennoprzecinkowe do warunków logiki programowalnej, a następnie dokonać ich realizacji sprzętowej, aby otrzymać konieczne zasoby sprzętowe oraz parametry określające wydajność obliczeniową, które pozwolą oszacować skuteczność proponowanych rozwiązań.

W pracy sformułowano następującą tezę:

Zastosowanie układów logiki programowalnej FPGA umożliwia implementację specjalizowanej architektury wydajnego akceleratora sprzętowego realizującego obliczenia zmiennoprzecinkowe dla potrzeb naukowo-technicznych (np. chemii kwantowej.

Rezultaty i wyniki badań

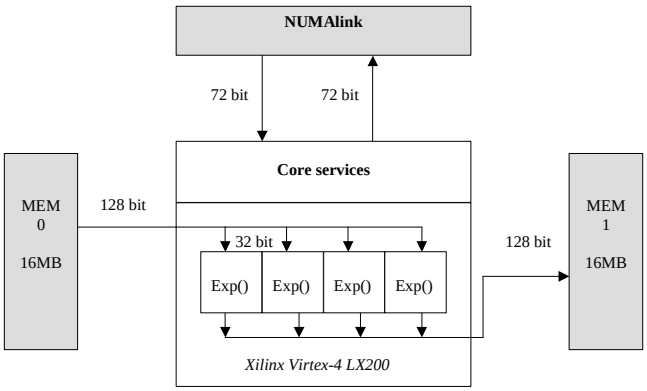
W wyniku prowadzonych prac powstał szereg modułów zmiennoprzecinkowych, które następnie wykorzystane zostały jako elementy składowe systemu obliczeniowego generującego wartości funkcji orbitalnej w punkcie trójwymiarowej siatki przestrzennej (*ang. grid*) [6].

Pierwszym elementem zaimplementowanym w układach FPGA przedstawionym w tej pracy, była funkcja exp(), dla której opracowano dedykowany sprzętowy algorytm tablicowo-wielomianowy zarówno dla pojedynczej jak i podwójnej precyzji w standardzie IEEE-754. W wyniku przeprowadzonych badań zaproponowana została architektura pozwalająca spełnić wymagania dotyczące pożądanej precyzji obliczeń przy jednoczesnym ograniczeniu zajętości zasobów logicznych. Było to możliwe dzięki zastosowaniu w ramach wspomnianej struktury rozwiązań takich jak migracja znaku liczby do części całkowitej wyniku oraz mnożarek o skróconej szerokości. Znaczącym ulepszeniem początkowej architektury wspomnianego modułu exp() była również redukcja szerokości wewnętrznych magistral danych, co w konsekwencji umożliwiło zastąpienie operacji mnożenia długich argumentów operacją dodawania o znacznie zmniejszonej szerokości.

Tab. 1 Porównanie zajętości zasobów logicznych modułów exp() zaimplementowanych w układzie Xilinx Virtex-4 LX200 [7,8]

	Użyte zasoby logiczne		
Rodzaj modułu	*# LUT (4 wejściowe)*	*# przerzutniki*	*# 18-Kb BRAMs*
Moduł exp() pojedynczej precyzji	820(0.5%)	926 (0.5%)	2 (0.6%)
Moduł exp() podwójnej precyzji	4283 (3%)	5757 (3%)	6 (1.8%)

Na podstawie wyników implementacji zamieszczonych w Tab. 1 można zauważyć, że przyjęta precyzja obliczeń ma znaczący wpływ na zajętość zasobów logicznych układu programowalnego.



Rys. 1 Rozmieszenie modułów exp() pojedynczej precyzji na platformie RASC

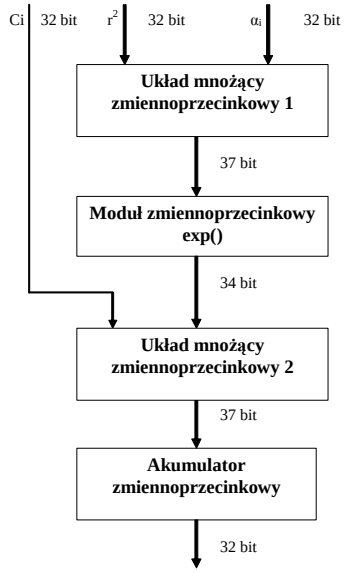
Możliwe do uzyskania przyspieszenie obliczeń jest zdeterminowanie poprzez szerokość interfejsu komunikacyjnego, która jest ograniczona. Dlatego też redukcja szerokości argumentu wejściowego prowadzi również do zwiększenia efektywnej równoległości obliczeń (Rys. 1).

Kolejnym elementem zrealizowanym w układach FPGA był moduł obliczający eksponencjalą część orbitalu atomowego, wyrażona następującą formuła matematyczną:

$$\chi_e(r) = \sum_i c_i e^{-\alpha_i r^2} \quad \textbf{(1)}$$

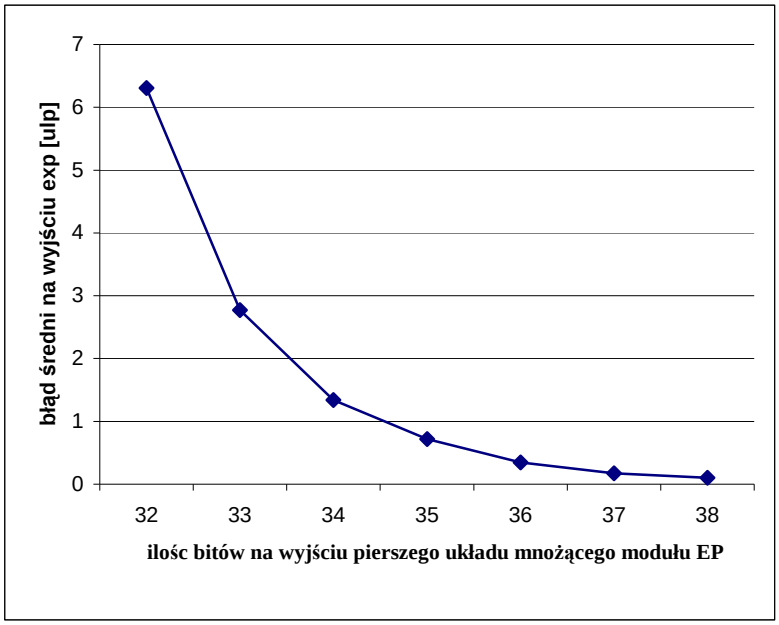
gdzie wartości c_i oraz α_i są współczynnikami bazy atomowej, w jakiej prowadzone są obliczenia, r^2 jest natomiast kwadratem odległości danego punktu w siatce przestrzennej centrowaną na każdym atomie[6].

Zaproponowana i zaimplementowana została w pełni potokowa architektura realizująca wspomnianą zależność (1). W wyniku przyjętego postępowania wykonano szereg modułów zmiennoprzecinkowych (Rys. 2), pracujących z częstotliwością 200 MHz [9,10,11].



Rys. 2 Schemat blokowy modułu obliczającego część eksponencjalną funkcji orbitalnej

Należy nadmienić, że wspomniane moduły są skalowane, umożliwiając dobór precyzji obliczeń w szerokim zakresie, co pozwoliło na modyfikację wewnętrznej szerokości magistrali danych w celu uzyskania większej dokładności wyniku. Jako przykład podana może zostać zależności błędu średniego na wyjściu modułu exp() w funkcji szerokości bitowej układu mnożącego znajdującego się bezpośrednio przed nim w strukturze logiki EP (Rys. 3).



Rys. 3 Zależność błędu średniego na wyjściu jednostki exp() w zależności od szerokości bitowej pierwszego układu mnożącego (Rys. 2)

Opóźnienie potokowe wprowadzone przez moduł obliczania części eksponencjalnej orbitalu atomowego (EP) wynosi odpowiednio 37 taktów zegara dla pojedynczej oraz 45 dla podwójnej precyzji obliczeń.

Tab. 2 Porównanie zajętości zasobów logicznych dla modułów EP zaimplementowanych w układzie Xilinx Virtex-4 LX200

	Użyte zasoby logiczne		
Rodzaj modułu	*# LUT (4 wejściowe)*	# przerzutniki	*# 18-Kb BRAMs*
Moduł EP pojedynczej precyzji	2229 (1%)	1975(1%)	2(0.006%)
Moduł EP podwójnej precyzji	8684 (4.8%)	7891(4%)	6(0.02%)

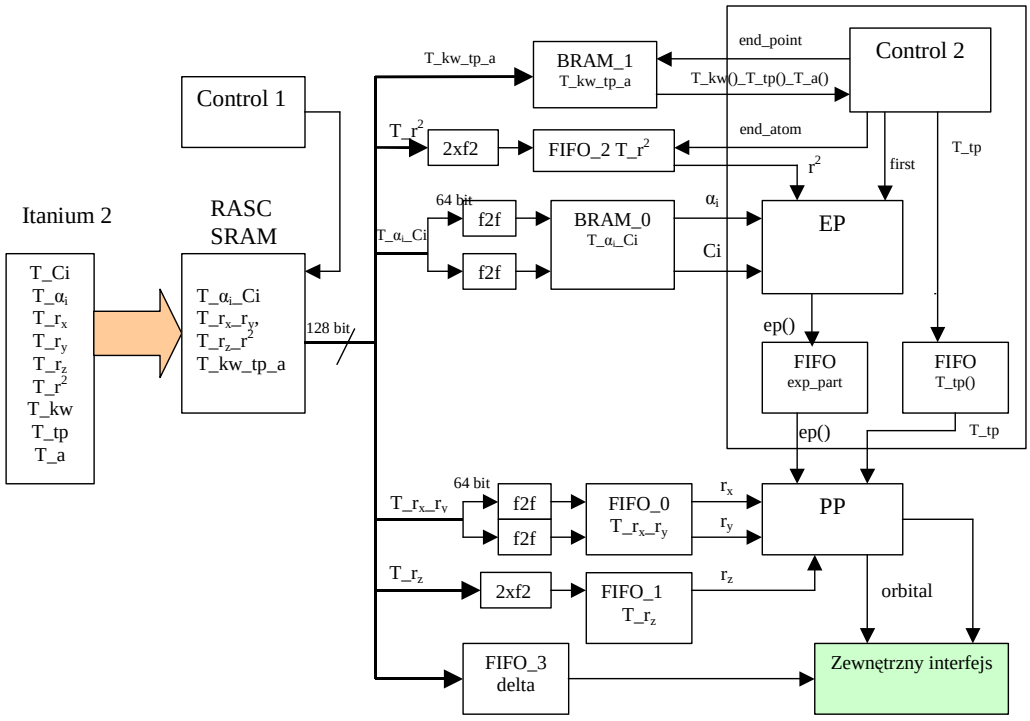
Najbardziej zaawansowanym modułem przedstawionym w niniejszej pracy jest sprzętowa implementacja funkcji obliczającej wartość orbitalu atomowego w punkcie.

Funkcja ta jest wyrażona następującą formułą matematyczną:

$$\chi_{klm}(r) = C_x \cdot C_y \cdot C_z \cdot r_x^k r_y^l r_z^m \sum_i c_i e^{-\alpha_i r^2} \quad \textbf{(2)}$$

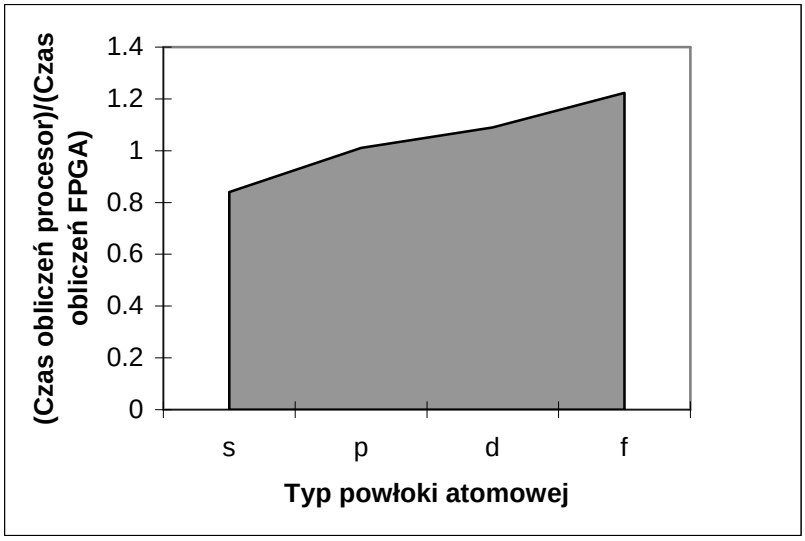
gdzie r_x, r_y, r_z , $r^2 = r_x^2 + r_y^2 + r_z^2$ określają położenie danego punktu rzutowanego na każdy atom w siatce przestrzennej[6], natomiast C_x, C_y, C_z są współczynnikami normalizacji. Wykładniki *k, l, m* zależą od typu powłoki atomowej (*s, p, d* lub *f*). Natomiast wartości *Ci* oraz α_i są współczynnikami bazy atomowej, w jakiej prowadzone są obliczenia.

Trudność wykonania tego modułu wynikała m.in. z silnej asymetrii liczby strumieni danych wejściowych dostarczanych do modułu w porównaniu z jednym tylko potokiem danych wynikowych. Należy przy tym zauważyć, że akcelerator RASC został wyposażony w pojedynczą magistralę danych wejściowych oraz wyjściowych, o szerokości 128 bitów.



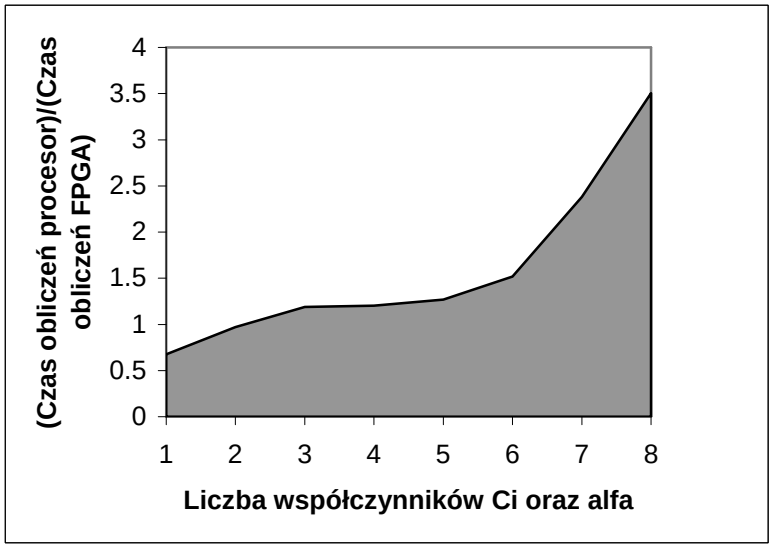
Rys. 4 Schemat blokowy systemu obliczającego funkcję orbitalną

Czas wykonania operacji obliczania wartości funkcji orbitalnej w punkcie z wykorzystaniem akceleratora RASC zależy w znacznym stopniu od rodzaju prowadzonych obliczeń. W przypadku cząsteczki wody był on zbliżony do wyniku uzyskanego z wykorzystaniem procesora Itanium 2 1.6 GHz. Dla bloku 512 punktów czas ten wynosi 2885 us, natomiast czas wykonania tych samych obliczeń na platformie RASC to około 3174 us . Należy zauważyć, że w przypadku cząsteczek z wyraźną dominacją wyższych powłok przewagę uzyskuje implementacja z wykorzystaniem układów FPGA, co zostało przedstawione na poniższych wykresach (Rys. 5, Rys. 6).



Rys. 5 Uzyskana akceleracja obliczania funkcji orbitalnej w zależności od typu powłoki, dla jakiej prowadzone są obliczenia

Ilość współczynników C_i oraz α_i (Rys. 6) decyduje również o efektywności wykorzystania jednostki EP, a tym samym wpływa na ogólną wydajność obliczeniową modułu obliczającego wartości orbitalu atomowego w punkcie.



Rys. 6 Uzyskana akceleracja obliczania funkcji orbitalnej w zależności od liczby współczynników

W docelowym systemie generującym macierz potencjału korelacyjno-wymiennego moduł obliczający wartości funkcji orbitalnej w punkcie będzie zaimplementowany dla pojedynczej precyzji obliczeń, gdyż taka dokładność została uznana jako wystarczająca na podstawie przeprowadzonych badań. Implementacja obliczeń w standardzie zmiennoprzecinkowym podwójnej precyzji wiązałaby się z około trzykrotnym zwiększeniem wielkości zajmowanych zasobów logicznych układu FPGA.

Tab. 3 Wyniki implementacji modułu obliczania funkcji orbitalnej dla pojedynczej precyzji obliczeń (Xilinx Virtex-4 LX200)

	Użyte zasoby logiczne		
Rodzaj modułu	*# LUT (4 wejściowe)*	# przerzutniki	*# 18-Kb BRAMs*
Moduł Orbital	8034 (4.5%)	7025 (3.4%)	14 (4.1%)
Moduł Orbital + core services	17365 (9%)	20972 (11%)	37(11%)

Podsumowanie

Przedstawione moduły zmiennoprzecinkowe zostały zaimplementowane z wykorzystaniem języka opisu sprzętu VHDL. Zaproponowano architekturę potokową odznaczająca się dużą efektywnością przetwarzanych danych. Ponadto wykonana implementacja wyposażona jest w parametryzację, która umożliwia dobór zarówno precyzji obliczeń jak i etapów potokowości przetwarzania, co w konsekwencji wpływa na zajętość zasobów logicznych.

Stopień akceleracji obliczeń z wykorzystaniem układów FPGA zależy w szczególności od:

- proporcjonalnego udziału zaimplementowanego fragmentu do całości obliczeń,
- struktury zaimplementowanej receptury obliczeniowej,
- możliwość redukcji precyzji obliczeń w ramach algorytmu,
- doboru właściwej precyzji obliczeń dla poszczególnych bloków składowych algorytmu (np. bity ochronne),
- możliwości uproszczenia struktury modułów składowych z wykorzystaniem operacji bitowych bez utraty precyzji (np. zastąpienie operacji mnożenia dodawaniem o zredukowanej szerokości),
- przepustowości magistrali komunikacyjnej łączącej akcelerator sprzętowy z systemem obliczeniowym.

Do oryginalnych rozwiązań autora zaprezentowanych w tej pracy, należy zaliczyć:

- wykonanie w pełni sprzętowej implementacji operacji obliczania atomowej funkcji orbitalnej w punkcie,
- opracowanie i wykonanie jednostek umożliwiających osadzenie modułu obliczania atomowej funkcji orbitalnej w strukturze akceleratora RASC,
- zaprojektowanie i realizację sprzętową modułu obliczania części eksponencjalnej orbitalu atomowego,
- wykonanie jednostki obliczającej cześć wielomianową orbitalu atomowego,
- modyfikacja tablicowo-wielomianowej metody obliczania funkcji exp() poprzez wprowadzenie operacji mnożenia o skróconej szerokości, migracji znaku do części całkowitej wyniku oraz wykorzystanie układów dodających o zredukowanej szerokości argumentu zamiast układów mnożących,
- opracowanie środowiska do symulacji i testowania poprawności działania projektu,

- opracowanie w języku C szeregu procedur sterujących pracą akceleratora RASC z poziomu procesora oraz algorytmów formatowania danych przesyłanych do układu FPGA,
- zaprojektowanie efektywnego mechanizmu składowania i buforowania danych w hierarchicznej strukturze akceleratora RASC, umożliwiającego prowadzenie obliczeń dla cząstek o różnej strukturze i rozmiarze.

Uzyskane wyniki implementacji i przeprowadzonych badań wykazują poprawność przyjętej w pracy tezy. Implementacja obliczeń zmiennoprzecinkowych w układach FPGA prowadzi do zysku obliczeniowego, który jest zależny od szeregu czynników przedstawionych w pracy.

Literatura

[1] J.Detrey, F. de Dinechin, *A parameterizable floating-point logarithm operator for FPGAs*, Proceedings of 39th Asilomar Conference on Signals, Systems & Computers IEEE 2005, s. 1186-1190.

[2] N. Takagi, *Powering by a Table Look–up and a Multiplication with Operand Modification*, IEEE Transactions on Computers, 47(11) 1998, s. 1216–1222,

[3] P. T. P. Tang, *Table Look-up Algorithms for Elementary Functions and their Error Analysis*, Argonne National Lab. Report, MCS-P194-1190, January 1991.

[4] H.T. Bui, S. Tahar, *Design and Synthesis of an IEEE-754 Exponential Function*, IEEE Canadian Conference on Electrical and Computer Engineering Shaw Conference Center, Edmonton, Alberta, Canada May 9-12 1999, vol.1, s. 450-455

[5] P.T.P. Tang, *Table-Driven Implementation of the Exponential Function in IEEE Floating-Point Arithmetic*, ACM Transactions on Mathematical Software (TOMS), Volume 15 , Issue 2 (June 1989), s. 144 – 157

[6] M. Makowski, *Podstawy chemii kwantowej – opracowanie*, Wydział Chemii UJ, Kraków 30 grudnia 2007

[7] M. Wielgosz, E. Jamro, K. Wiatr, *Moduł obliczający eksponenty implementowanej w układach FPGA*, Pomiary, Automatyka, Kontrola vol. 53, nr 7/2007, s.27-29

[8] E. Jamro, M. Wielgosz, K. Wiatr, *FPGA implementation of 64-bit exponential function for HPC*, Proceedings of the 17th International conference on field programmable logic and applications, Amsterdam, Netherlands, 27-29 August 2007, s. 718-721

[9] M. Wielgosz, E. Jamro, K. Wiatr, *Acceleration of Exponential funtion on Reconfigurable application specific computing platform*, Computing and Informatics (ISSN 1335-9150, IF = 0.349), Vol. 28, 2009, s. 1001–1011

[10] M. Wielgosz, E. Jamro, K. Wiatr, *Accelerating calculations on the RASC platform. A case study of the exponential function*, Applied Reconfigurable Computing, ARC'2009, Springer-Verlag, LNCS 5453, s. 306-311

[11] M. Wielgosz, E. Jamro, K. Wiatr, *Highly Efficient Structure of 64-Bit Exponential Function implemented In FPGAs*, Applied Reconfigurable Computing, ARC'2008, Springer-Verlag 2008, LNCS 4943, s. 274 – 279