Two Electron Integrals calculation accelerated with Double Precision exp() Hardware Module

Maciej Wielgosz[2], Marcin Pietroń[2], Ernest Jamro[1,2], Paweł Russek[1,2] , Kazimierz Wiatr[1,2]

1.Akademia Górniczo-Hutnicza, al. Mickiewicza 30, 30-059 Kraków
2.ACK Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków
email:wielgosz/pietron/jamro/russek/wiatr@agh.edu.pl

Abstract

FPGA implementation of double precision exponential function module is presented. The module will be incorporated in the Gaussian system to accelerate extremely time consuming exponential function evaluation. The exp function is accelerated on SGI RASC[1] board with two Virtex-4 LX200 FPGA. The exp() function alone occupies less than 3% Virtex-4 LX200 FPGA. Exp() arguments are fetched to the FPGA's and results are sent back to processors over the system bus working at speed of NUMAlink 6,4 GB/s. The exponential module reaches the processing speed of 200 MHz, the external memory interface limits the number of operation to two exp() every clock cycle per a FPGA. The overall end-to-end algorithm execution speedup that authors expect to achieve 4x as compared to a sequential implementation of the algorithm executed on a single 2 GHz Intel Itanium2 microprocessor.

Gaussian

Gaussian[2] is a software environment, employed by chemists, chemical engineers, biochemists, physicists and others for research in established and emerging areas of chemical and physics interest. Starting from the basic laws of quantum mechanics, *Gaussian* predicts the energies, molecular structures, and vibration frequencies of molecular systems, along with numerous molecular properties derived from these basic computation types. Since the majority of the operations conducted with *Gaussian* are high precision, data of double precision are ubiquitous in this application.

Profiling of the *Gaussian*

Employment of well known profilers (e.g. *gnuprof*) was the first step to profile Gaussian application. Results of using these profilers were not satisfying (e.g. binaries produced by *gprof* after compiling *Gaussian* can not be debug). Consequently a new, dedicated profiling tool was developed. This tool is able to determine data dependencies (e.g. data dependencies in the loop). Such the environment is very helpful in finding hotspots which can be accelerated with FPGA. There are a lot of exponential functions in source code but only several of them are heavily employed in most common chemical computations tasks. For example while computing benzene molecule the exp() function is executed a few millions times. One of the exp() function hot spots is the subroutine responsible for functional computation in solving the Hartree-Fock equation.

Exponential function

Many hardware examples of single precision FP exponential function implementations can be found [3,4,5,6] contrary to efficient double precision standard ones [7], which are unknown to the authors. This disproportion results from the fact that commonly known table-based or polynomial methods are not straightforward applicable to this double precision elementary function. Therefore some novel solutions were adopted to the proposed exp() calculation module not only to preserve compatibility to double precision standard but also to achieve high processing speed (200 MHz) and satisfying accuracy. The exp() module is fully pipelined

(max. pipeline latency is 28 clks). Equations (1) and (2) are a mathematical background of exp() module solution presented hereby. Eq. (1) produces directly the final result exponent field - 2^a. Eq. (2) is implemented using 3 independent Look-Up Table (LUT) memories: e^{b1}, e^{b2}, e^{b3}. The expression p is a very small number therefore the formula e^p is approximated by $1+p$ according to Taylor expansion series (3).

$$e^x = e^{a*ln2+b} = 2^a.e^b \quad (1)$$

$$e^b = e^{b1}.e^{b2}.e^{b3}.e^p \quad (2)$$

$$e^p = 1 + p + ... \quad (3)$$

The final result is the product of partial exp() expressions according to (2). Each LUT memory (Fig.1) has 9-bit wide input (optimal for Virtex 4 512×32bit BRAMs) which allows to spare much logic resources (that would be otherwise absorbed by one huge LUT) at the expanse of two additional multipliers. The sign of the input argument is considered only in integer part, (1) expression *a,* which results in further drop of occupied area.

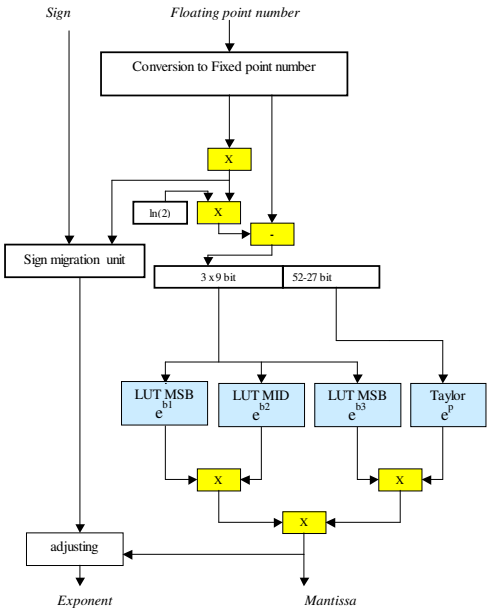

Fig. 1. Block diagram of exp() module.

Dedicated fixed-point, non-zero-input argument and reduced-width multipliers are employed which results in area decrease and increase of speed. Instead of multiplying shift and add operation is performed (Fig.2.).
Multipliers are fed with adjustable width of the argument depending on the number of guard bits that may vary from 0 to 8. Multipliers outputs are also determined by a number of guard bits.

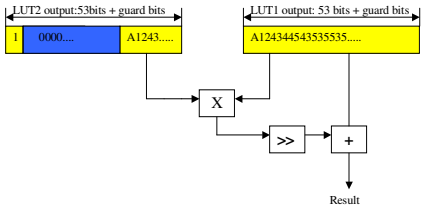

Fig. 2. Non-zero-input argument multiplier

Conclusion

Double precision exponential module was design and also profiling of *Gaussian* was performed which resulted in precise information on exp() function instances in the code. The main challenge that authors are to face now is integration of exp() modules with extracted parts of the source so the whole system achieves expected speedup. The VHDL language has been chosen for the implementation together with Aldec Active-HDL environment.

References

[1] SGI *RASC RC100 Blade Manual*, version 2.1 http://techpubs.sgi.com/library/manuals/4000/007-4718-006/pdf/007-4718-006.pdf.

[2] Gaussian 03 Online Manual, update 02 http://www.gaussian.com/g_ur/g03mantop.htm.

[3] Detrey J., de Dinechin F, *Table-based polynomials for fast hardware function evaluation*, 16th IEEE International Conference on Application-Specific Systems, Architectures, and Processors (ASAP'05), Samos, Greece, July 2005, pp. 328-333.

[4] Doss C.C., Riley R.L., Jr., *FPGA-Based Implementation of a Robust IEEE-754 Exponential Unit*, 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04), pp. 229- 238.

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

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