Lecture Notes in Computer Science. –V.3019. –2004. -P.36-41.

Configurable microprocessor array for DSP applications

O.Maslennikov*, Ju.Shevtshenko**, A.Sergyienko**

*Polytechnica Koszalinska, Poland,
Email: oleg@moskit.ie.tu.koszalin.pl

** National Technical University of Ukraine,
Email: aser@comsys.ntu-kpi.kiev.ua

Abstract
The method for mapping parallel algorithms into FPGA is proposed which is based on programming the configurable microprocessor array. The hardware volume of the based RISC processor unit soft core is minimized, and adapted due to the used instruction subset. The method provides both high throughput and minimized hardvare volume, and speedups the design process. The method was proven in the microprocessor array for solving the linear equation system with the Toeplitz matrix.

Keywords: FPGA, DSP, processor array.

1. Introduction.

Modern DSP applications, like MPEG-4 standard packing, vocoders, etc. are characterized by both high algorithm complexity (thousands of instruction rows in the programs) and computational intensiveness (several millions and billions of operations per second). Besides, the algorithm set in the device can be substituted dynamically according to the data stream parameters, or to the application exchanging. Both hardware and power consumption minimization are of great demand. As a rule, such applications are implemented in the signal microprocessors. To achieve the high throughput, the signal microprocessors become parallel processor systems with several processor units (PUs). But the hardware of such microprocessors is not utilized very well. This is explained by that that many DSP operators could not directly mapped into the microprocessor instructions. Besides, the modern compilers for such signal microprocessors are not effective ones.

Reconfigurable computing is the real alternative both to ASICs and signal microprocessors now. Its advantage is provided by broad applicability, due to reconfiguration properties and high performance, through the potential parallelism exploitation and direct implementation of any algorithm operator. The field programmable gate arrays (FPGAs) is the most commonly used raw for the reconfigurable computing.

In the last decade the density of FPGAs, their clock frequency, and routing capabilities are increased dramatically. The high density and clock frequency of modern FPGAs provide

their extremally high throughput. For example, the Xilinx Virtex-2 FPGAs consist of several tenths of combinational multipliers, data RAMs, and it takes about 200 configurable logic block (CLB) slices per one multiplier. The expanding of FPGAs at the field of modern DSP applications is limited now because of the labor consumable process of mapping the algorithms into FPGA. One of the way to solve this problem is the intellectual property (IP) core reuse. The another way is programming the parallel system of processing units (PUs), which is configured in FPGA.

In the representation the method for mapping parallel algorithms into FPGA is proposed which is based on programming the configurable microprocessor array, and provides both high throughput and minimized hardvare volume.

2. Mapping parallel algorithms into the configurable microprocessor array.

In [1] the PU array for image processing applications is proposed which is configured in FPGA. Each PU is implemented as the IP soft core with the architecture of the well-known microcontroller i8051. The core hardware volume is exchanged in the range of 500 - 1100 CLB slices, depending on the core functionality. This means that it takes from 3 to 6 multipliers per one PU core in the Xilinx Virtex-2 FPGA. As a result, the most of multipliers are unused, and hardware utilization is not effective one. Such situation also occurs when another IP soft cores of RISC microprocessors are used in the configurable microcontroller array, like ARC core, or Leon SPARC core which consist of much more CLBs.

In the representation the method for mapping parallel algorithms into FPGA is proposed, which provides balancing the PU hardware volume with the FPGA resources. That means that the PU soft core must have the hardware volume less than 200 – 400 CLB slices, and 1 - 2 multipliers.

As the PU core the RISC_ST soft core is selected which is described in [2]. This core consists of the base core and the hardware extension unit. The base core has the RISC 16-bit architecture with the 2-staged instruction pipeline. It performs each instruction for a single clock cycle. The instruction RAM is separated from the data RAM. To achieve the high performance in the control intensive applications, the delayed branch mechanism is used. The high speed interrupts, and subroutine calls are supported by the hardware stack. After the interrupt routine end the instruction pipeline recovery has not any difficulties, because the heavy instructions, like jump instructions, delayed branch instructions, are not interruptable. The base PU core has the hardware volume only 190 CLB slices.

The PU hardware extension unit implements the proper instruction set extension. This extension is adapted to the different DSP applications. The hardware of this unit can vary depending on the given instruction set, and precision of computations. The unit for implementing the FFT algorithm is different from the unit for filter calculations, and provides, for example, the hardware implementation of the bit reverse addressing.

The PU core is described by VHDL, and runs in Xilinx Virtex-2 devices at the clock frequency, which is equal to 90 MHz. The core hardware volume is exchanged in the range of 190 - 500 CLB slices, depending on the implemented instruction set extension. The assembler was developed which generates the program codes, and outputs the table of generic constants for the hardware extension unit selection. An IP core generator was developed, which generates this PU soft core with the program RAM, constant ROM content, and proper hardware extension unit.

The parallel processor system has the ring structure. But it can be freely exchanged to any structure, which is suported by the reconfigurable nature of FPGA. The PUs interact each other by data buffers and interrupt mechanism. Such architecture provides the wave propagation processor implementation, and expanding the PU number due to the increase of the FPGA device number in the system. Due to the small PU hardware volume one FPGA device can contain more than a hundred of PUs, and provide the throughput up to ten billions of multiplications and additions per second.

The system configuring process has the following three stages. On the first stage the user microprocessor programs are designed and compiled. By this process the software pipelining for the processor array is used, which described in [3]. The derived parallel program is debugged, tested, and tuned using the behavioral model of the processor array in the VHDL simulator. Such a process can be accelerated when the hardware accelerator is attached to the simulator, which is based on the very FPGA device. By the program compiling the unused instruction codes and data addresses are fixed.

On the second stage the PU cores are generated, in which the unused units and logic cells are taken off. Each node program is mapped into the separate PU core. If the resulting architecture is SPMD - architecture, then the only one PU core is generated. When the application needs the intensive calculation of some special functions, for example, floating point operations, then the proper functional unit can be attached to the PU core. This unit has to be a fully pipelined data flow path with the high throughput, which can supersede the

throughput of the signal microprocessor. The structure of such unit is synthesized by the method, described in [4].

On the third stage all the PU cores, or copies of a single PU core are attached to the microprocessor array netlist, and the whole project is translated into the configuration file of the target FPGA device.

3. Experimental results.

The configurable microprocessor array was probed in programming the solving the linear equation system with the Toeplitz matrix. This problem is solved in DSP systems for adaptive filtering, spectrum estimating, voice coding, etc. Usually this problem is solved using the floating point data representation or the integer dates with doubled length, and specific algorithms which support the error minimization. The N+1 processor systolic array solves the N*N Toeplitz matrix problem for N iterations using the Schur algorithm [5]. Such computational schema is used in our example as well.

The disadvantage of this schema consists in that that the nodes with the division operation form the critical path. And this operation is time consumable in the RISC processors. Therefore it limits the throughput of the whole array.

To minimize the division delays the untraditional data representation is used. Each data x is represented by two integers which are numerator n_x and denominator d_x, i.e. the data is equal to the fraction $x = n_x/d_x$. At first N-1 iterations all the calculations are implemented with such a data. Multiplication, division, and addition look like:

$$x*y = n_x\, n_y /d_x\, d_y\,; \quad x/y = n_x\, d_y/d_x\, n_y\,; \quad x+y = (n_x\, d_y + n_y\, d_x\,)/\, d_x\, d_y.$$

At the last iteration denominators divide numerators to derive the algorithm results. Such data representation provides both small calculation errors and expanded dynamic range comparing to the usual integer data representation.

The PU hardware extension unit consists of two multipliers, and implements the multiplication and division for a single clock cycle, and addition for two clock cycles. To provide the minimum calculation errors each operation is finished by the normalization of resulting numerator and denominator, shifting left their codes to the equal bit number.

The PU hardware volume is equal to 380 CLB slices, 2 multipliers and 2 RAM blocks. The PU system for N=10 is fitted the 75% of the hardvare volume of XC2V1000 device. It implements the algorithm for 1.23 microseconds to take not to account the data input-output. The average speed is equal to 170 millions operations per second, like addition, multiplication, division of fractional dates. The system with up to 84 such PUs can be

configured in the XC2V8000 device, and provide approximately 1900 millions operations per second when implementing this algorithm.

4. Conclusion.

The method for mapping parallel algorithms into FPGA is proposed which is based on programming the configurable microprocessor array, and provides both high throughput and minimized hardvare volume. The proposed configurable microprocessor array is very useful in such DSP applications where logic intensive calculations, or computations of dates in the unusual format, or complex algorithm computing are of demand. These applications cover MPEG-4 packing, multichannel CELP vocoders, open key encryption systems, etc. The method was successfully proven in the microprocessor array for solving the linear equation system with the Toeplitz matrix.

References.

[1] . *Maslennikov O., Shevtshenko Ju., Sergyienko A.*. Configurable microcontroller array. *Proc. of the 3-d Int. Conf. on Parallel Computing in Electrical Engineering. PARELEC'2002,* Warsaw, Poland. 22-25 Sept., 2002. P. 47-49.

[2]. *Sergyienko A.* VHDL for computer development. *Kiev. Korneychuk,* 2003. 184 p. (In Russian).

[3]. *Sergyienko A., Kaniewski J., Maslennikov O., Wyrzykowski R.* Mapping regular algorithms into processor arrays using software pipelining. *Proceedings of the 1-st Int. Conf. on Parallel Computing in Electrical Engineering. PARELEC'2002* Bialystok, Poland. 2-5 Sept., 1998. P.197-200.

[4]. *Kanevski Ju.S., Sergienko A.M., Piech H.* A Method for the Structural Synthesis of Pipelined Array Processors. *Proc. of the First Int. Conf. "Parallel Proc. and Appl. Math. PRAM'94"* (Czestochova, Poland, 1994). P. 100-109.

[5]. *Kung S.Y.* VLSI processor arrays. *Prentice Hall, Englewood Cliffs,* 1988.