DSP-Specific Field-Programmable VLSI and Its CAD Environment

Masanori Hariyama, Sho Ogata, Michitaka Kameyama
Graduate School of Information Sciences
Tohoku University
Aoba 6-6-05, Aramaki, Aoba, Sendai, Miyagi, 980-8579, Japan
Email: {hariyama@,ogata@kameyama.,kameyama@}ecei.tohoku.ac.jp

***Abstract*— A field-programmable VLSI processor (FPVLSI) is proposed based on a bit-serial pipeline architecture and a mesh network that greatly reduce complexity of a programmable interconnection network. An area-efficient shift-register-based cell is also proposed based on regularity of bit-serial operation. To minimize an inter-cell network, functionality of a programmable switch block is realized only by programming cells. Moreover, direct allocation of a control/data flow graph (CDFG) is introduced to localize data transfer. As a result, an ultra-highly-parallel cellular array is achieved that can efficiently exploit high parallelism of DSP applications. The FPVLSI is designed in a 0.18μm CMOS design rule. Its performance at 700MHz is 4.7 to 9 times higher than that of a conventional FPGA in typical applications under constraints of the same chip area and the same power consumption.**

I. Introduction

Field-programmable gate arrays (FPGAs) are widely used for implementing special-purpose processors. FPGAs are cost-effective and flexible since the FPGA consists of an array of processing units whose function and interconnection can be programmed after fabrication [1], [2].

Figure 1 shows a structure of a typical FPGA. The FPGA has lookup tables (LUTs) to serve as programmable computation elements (logic blocks). Logic blocks are connected to each other through programmable switch blocks. Each switch block consists of many programmable switches. The major disadvantage of FPGAs is low performance from the following reasons:

- The delay and area of switch block are large due to a large number of programmable switches in the switch block.
- Propagation delay between logic blocks is large since a logic block is connected to another logic block through many switch blocks.

To solve the problems, this paper proposes a field programmable VLSI(FPVLSI) based on following concepts.

Bit-serial pipeline architecture: Bit-serial pipeline architecture greatly reduces the delay and area of interconnection network since the data is transferred between cells through 1-bit interconnection so that the area of interconnection can be greatly reduced. The area reduction leads to increase of the number of cells, that is, a high degree of spatial parallelism. Moreover, the bit-serial pipeline architecture achieves the high utilized ratio of a cell irrespective of a word length.

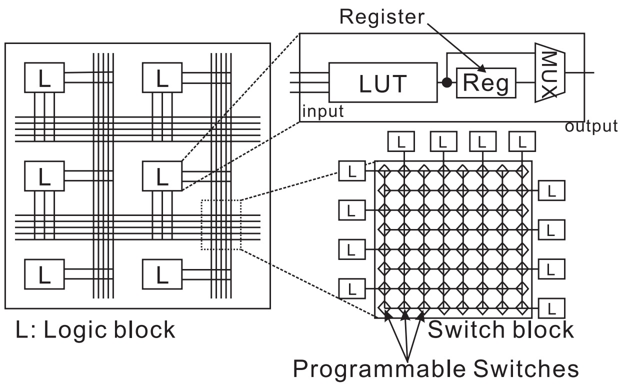

Fig. 1. Block diagram of a basic FPGA

Area-efficient cell for bit-serial architecture: A cell should be capable of performing operations, storing intermediate results, and controlling bit-serial operations. To implement these three functions area-efficiently, the cell is constructed by shift-register-based lookup tables. Moreover, functionality of a programmable switch block is realized only by programming cells to minimize an inter-cell network. As a result, a high degree of spatial parallelism is achieved.

Direct allocation of a control/data flow graph: To execute an algorithm represented by a control/data flow graph (CDFG), direct allocation of the CDFG is introduced to map each operation in a CDFG into a single cell. The direct allocation make the multiplexers for input selection be unnecessary so that data transfer between cells can be localized.

As a result, an ultra-highly-parallel cellular array is achieved that fully exploit high degree of parallelism of DSP(Digital Signal Processing) applications.

II. Architecture

A. Overview of the FPVLSI

Figure 2 shows the structure of the FPVLSI. The cell consists of processing elements(PE) and a switch block as shown in Fig. 3. The PE mainly consists of the two shift-register-based look-up tables(LUTs) with three inputs as shown in Fig. 4. Each cell is connected to only four neighbor cells through 1-bit switch blocks so that the number of programmable switches is greatly reduced in comparison with the conventional FPGA. Hence, the area and the delay of the switch block become much smaller than those in the FPGA.

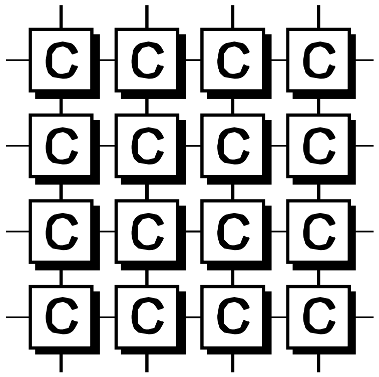

Fig. 2. Mesh-connected cellular array.

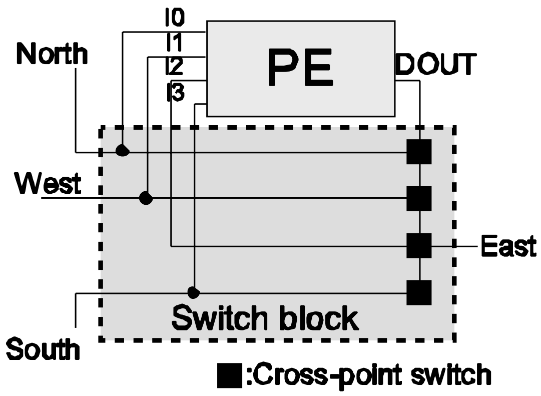

Fig. 3. Cell structure.

Note that each input of the PE is directly connected to the outputs of the four neighbor cells. The functionality of the switch block is realized by programming the PE. In comparison with the previous version of the FPVLSI[3] with input programmable switches, the area of the PE becomes slightly larger to realize the switch block functionality, while the number of the cross-point switches is reduced to about 50%. As a result, the cell area is reduced to about 60%.

B. *Cell using shift-register-based lookup tables*

The cell in the FPVLSI must have three different functions: an arithmetic/logic function, a memory function, and a control function. For an area-efficient design, shift-register-based LUTs are used in the PE as shown in Fig.4. Each of an arithmetic/logic function, a memory function and a control function can be implemented using two shift-register-based LUTs cell as follows:

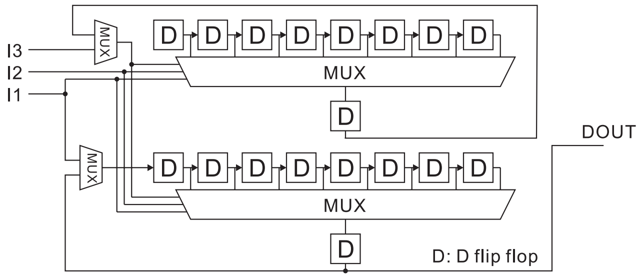

Fig. 4. Block diagram of the PE.

Arithmetic/logic function: The arithmetic/logic function is implemented as two LUTs. The LUT consists of the 8D flip flops with output multiplexer. The 3-input LUT can implement an arbitrary logic function of 3 inputs. Moreover, the function of a full adder is implemented using two LUTs. One LUT generates a sum, and the other LUT generates a carry. In a memory mode, I0 is used as the input of the shift register. In a control mode, a one-hot counter is implemented to generate a signal for the word termination by feeding the LUT output back to the left-serial input.

Memory function: The memory function is implemented as a shift register because each bit of a memory is always accessed sequentially in bit-serial architecture. As a result, the bit-by-bit memory access can be performed without a counter, which is necessary in the conventional bit-serial architecture. and relatively larger than the ALU and the memory module.

Control function: The control function is used to reset the carry-propagation registers for bit-serial arithmetic operations. The control function is implemented by a one-hot counter using shift register. The counter has the ”1” in the least index bit at reset and shifts it by one bit toward the greatest index bit at each clock cycle, then ”1” is produced after n clock cycles.

III. CAD ENVIRONMENT

Figure 5 shows the design flow for the FPVLSI. A behaviour is given by data-flow graph(DFG) representation. The given DFG is mapped onto the FPVLSI through the procedures shown in the figure. The search is controlled based on simulated annealing like mapping for conventional FPGAs. The most remarkable feature of the mapping algorithm is that the placing and routing are combined into the same loop. In mapping algorithms for conventional FPGAs, placing and routing are separated to reduce search time. In other words, a total place-and-route problem is divided into small two problems: a place one and a route one. This is because conventional FPGAs have a large amount of interconnection resource, and finding a feasible solution for routing is relatively easy even if placing and routing are separated. However, in the FPVLSI, the interconnection resource is limited, and a feasible solution for routing is difficult to be found after a solution for placing is fixed.

A cost function is essential to find a good solution in a reasonable time. A summation of a pseudo-wire length using a Manhattan distance is typical one. However, it may result in a local optimum. For example, let us map a DFG shown in Fig. 6(a) onto the FPVLSI. To eliminate redundancy in search, O_0 is fixed on a specified cell. The placing is done from a node closer to O_0. Assumed that a mapping shown in Fig. 6(b) is a current optimal solution, and that O_2 is selected. If we use a summation of pseudo-wire lengths from O_2 to all the nodes

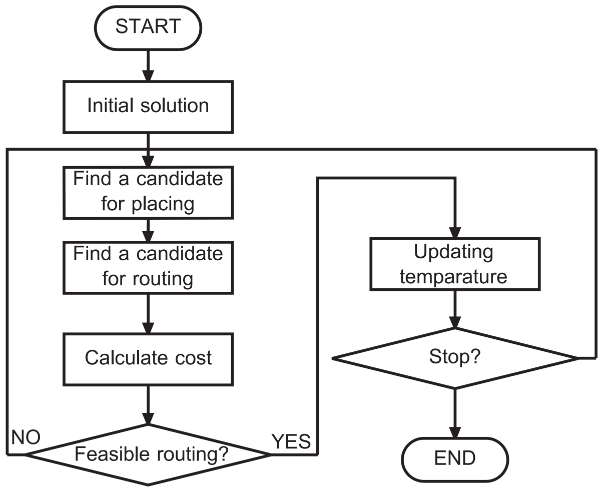

Fig. 5. Design flow for FPVLSI.

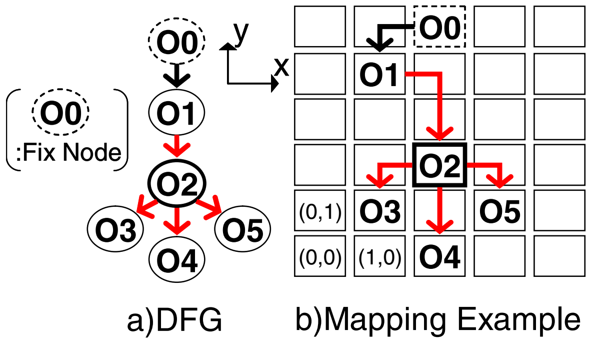

Fig. 6. Calculating a cost function.

connected to O_2 as a cost function, O_2 is located to $(2, 1)$. This is because O_2 has more connected nodes: $O_3, O_4 and O_5$ on the underside. Hence, O_2 is located far from O_0, and it results in a larger area. To solve this problem, we use a summation of pseudo-wire lengths from neighbours that are connected to O_0 as a cost function. By using this cost function, nodes are gradually pulled to O_0, and it results in a smaller area.

IV. Evaluation

Figure 7 shows a micrograph of the FPVLSI with 64 cells designed in a $0.18\mu m$ CMOS design rule. The clock frequency is evaluated to be 700MHz by the HSPICE simulation. For comparison, an FPGA based on the XILINX XC4000E architecture[1] is also designed, where the FPGA has dedicated interconnection for fast carry ripple. Let us evaluate the performance of the FPVLSI using three typical DSP applications: FFT, stereo matching[4], FIR filtering under the constraints of the same chip area and the same power dissipation. Assumed that the chip area is 100mm in the follwoing.

First, let us evaluate the performance of the FPVLSI using 16-point FFT with 16-bit fixed-point. FFT consists of butterfly operations shown in Fig. 9(a). The butterfly operation is mapped onto the FPGA and FPVLSI as shown in Figs.

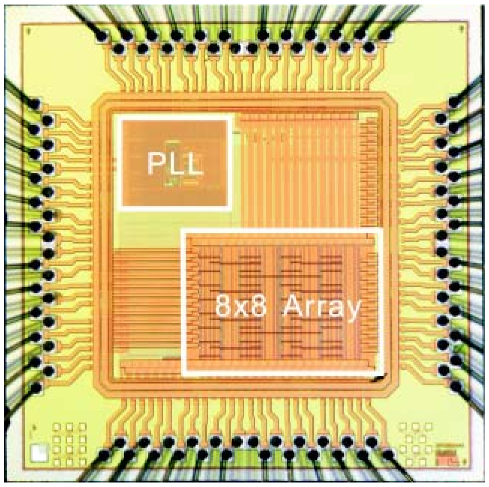

Fig. 7. Micrograph of the FPVLSI (0.18μm CMOS, 2.8mm$\times$2.8mm).

9(b) and (c), respectively. The butterfly unit on the FPGA is much larger than the one on the FPVLSI because the FPGA implementation uses a bit-parallel architecture. Note that FPGA implementation has more un-used logic blocks than FPVLSI because of its large routing area. As a result, the number of the FFT units on the FPGA is much smaller than that on the FPVLSI as shown in Table I. The FPVLSI implementation has 13 FFT units, and FPVLSI can exploit the parallelism of FFT. Since the FFT can be performed in parallel for all the data sets, the performance increases in proportion to the number of FFT units. The clock frequency is estimated to be 700MHz independently of the chip size because no global control is necessary in the FPVLSI as described above. In spite of the bit-serial pipeline architecture, the processing time of a FFT unit on the FPVLSI is only 1.4 times longer than that on the FPGA. This is because the interconnection delay is greatly reduced because of the simple interconnection network as shown in Table I. The total processing time of the FPVLSI-based FFT processor is 9 times higher than that of the FPGA-based one under the conditions of the same chip area and the same power dissipation.

Next, let us evaluate the performance of the FPVLSI using 16bit 4th-FIR Table II shows the comparison result. The performance of the FPVLSI-based implementation is 7 times higher than that of the FPGA-based implementation.

Finally, let us evaluate the performance of the FPVLSI using an 16bit SAD(Sum of Absolute Differences) computation. The SAD computation is used to measure similarity between two vectors. For vectors with N elements, AD(absolute difference)-and-ADD operations are iterated N times. The SAD computation is mainly used for image matching such as stereo matching, MPEG encoding and so on. Table III shows the comparison result. The performance of the FPVLSI-based implementation is 4.7 times higher than that of the FPGA-based implementation. The result is slightly worse than results on the applications because AD operation includes conditional branch, and it decreases the throughput.

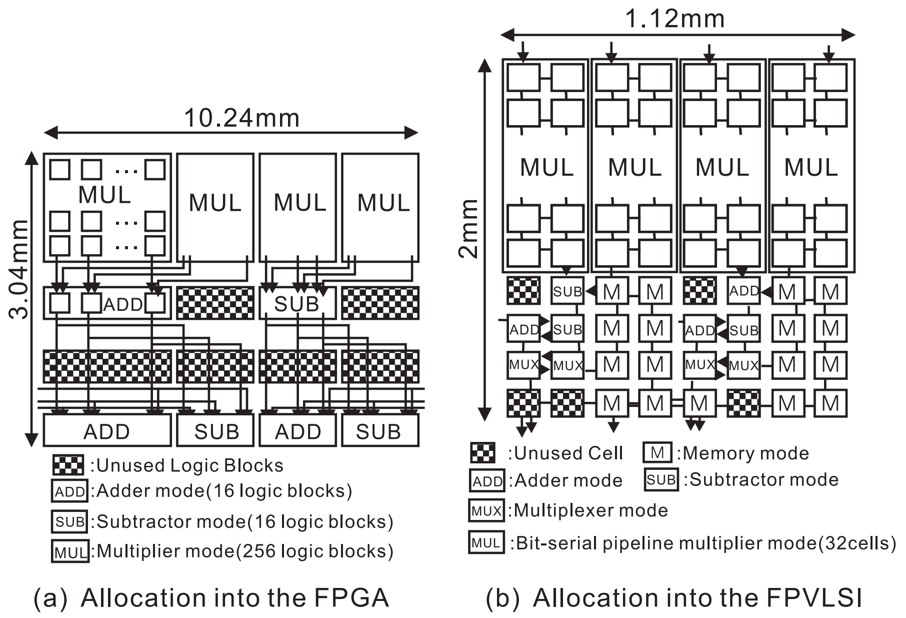
10.24mm
3.04mm
MUL
ADD
SUB
▩ :Unused Logic Blocks
ADD :Adder mode(16 logic blocks)
SUB :Subtractor mode(16 logic blocks)
MUL :Multiplier mode(256 logic blocks)
(a) Allocation into the FPGA
1.12mm
2mm
M
MUX
▩ :Unused Cell
M :Memory mode
ADD :Adder mode
SUB :Subtractor mode
MUX :Multiplexer mode
MUL :Bit-serial pipeline multiplier mode(32cells)
(b) Allocation into the FPVLSI

Fig. 9. Allocation of FFT.

TABLE I

PERFORMANCE COMPARISON USING FFT.

	FPVLSI	FPGA
Number of FFT units	13	1
Processing time of a single unit per operation [ns]	22.3	16.0
Interconnection delay in the processing time [%]	42%	78%
Average processing time per operation[ns]	1.7	16.0

TABLE II

PERFORMANCE COMPARISON USING FIR FILTERING.

	FPVLSI	FPGA
Number of FIR units	9	1
Processing time of a single unit per operation [ns]	22.8	20.0
Interconnection delay in the processing time [%]	56%	83%
Average processing time per operation[ns]	2.5	20.0

TABLE III

PERFORMANCE COMPARISON USING STEREO MATCING.

	FPVLSI	FPGA
Number of SAD units	208	15
Processing time of a single unit per operation [ns]	111	19
Interconnection delay in the processing time [%]	39%	78%
Average processing time per operation[ns]	0.53	2.53

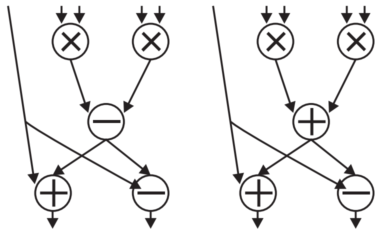

Fig. 8. CDFG of a butterfly operation.

V. CONCLUSION

High-performance field-programmable VLSI is proposed based on bit-serial pipeline architecture. Bit-serial pipeline architecture is also useful for inter-chip interface since it reduces the number of I/O pins and allows to use high-speed serial data-transfer scheme. High-level synthesis for direct allocation is also the challenging problem. The bit-serial pipeline architecture imposes a lot of timing constraints on allocation. These features of the FPVLSI may make it difficult to find an optimal solution in allocation.

REFERENCES

[1] W. Carter, K. Duong, R. H. Freeman, H. Hsieh, J. Y. Ja, J. E. Mahoney, L. T. Ngo, S. L. Sze, *A user programmable reconfigurable gate array*, Proc. Custom Integrated Circuits Conf., pp.232-235(1986).

[2] P. Chow, S. O. Seo, J. Rose, K. Chung,G. Paez-Monzon, I. Rahardja, *The design of an SRAM-based field-programmable gate array-Part I : Architecture*, IEEE Trans. VLSI Syst., vol.7, no.2, pp.191-197, 1999.

[3] N. Ohsawa, M. Hariyama, M. Kameyama, *Program-Counter-Less Bit-Serial Field-Programmable VLSI Processor with Mesh-Connected Cellular Array Structure*, IEEE Computer Society Annual Symposium on VLSI, 2004,pp.258-259,2004.

[4] M. Hariyama, M. Kameyama *VLSI Processor for Reliable Stereo Matching Based on Window-Parallel Logic-in-Memory Architecture* , Digest of Technical Paper 2004 Symposium on VLSI Circuits VLSI Symposium, pp.166-169,2004.