An Efficient Parallel Architecture Using Register-in-Logic Element for Digital Signal Processing

Sung-il Bae, Hyun-Jin Kim, Hyuntae Park, Sungho Kang †
anet0727@yonsei.ac.kr, {nagicman, radiob}@soc.yonsei.ac.kr, shakng@yonsei.ac.kr
Department of Electrical and Electronic Engineering
Yonsei University, Seoul, South Korea

Summary
In general processors, a logic unit and a register file exist individually. This architecture has a simple structure but creates data dependency. The data dependency is partially solved through various technical studies. In this paper, we propose an efficient parallel architecture to compensate the demerits of typical designs. The new architecture consists of modified logic blocks with general-purpose registers inside. These blocks are located in parallel like a reconfigurable computing hardware. The proposed architecture eliminates structural problems and guarantees high performance. We evaluate it in basic digital signal processing functions compared with a commercial DSP processor. In the results, we obtain 2~4 times faster performance than the typical architecture in digital signal processing

Key words:
DSP, Reconfigurable Computing, Data Dependency

1. Introduction

The continued progress of semiconductor technology has enabled the "system-on-a-chip" (SoC) to become a reality. In this sense, processor designers have proposed new paradigm architecture. Recent processor points to a parallel computing architecture that combines a multiple module with a high performance logic element. This trend shows that high performance is much more needed than small complexity and low power consumption. Application specific integrated circuits (ASICs) and general-purpose processors speak for current processor platforms. Market requires the optimum structure to specified function submitting high complexity. Power is not an impact factor to processor design any more. The power consumption seizes designer for a long time. But, we consider power consumption for only mobile applications these days [1-3].

Most processors are constructed via the typical architecture. The conventional architecture is composed of register file, arithmetic and logic unit (ALU), multiplier, program memory, data bus, and main controller. This composition has many problems like data dependency and branch prediction. These problems block efficient operations in the iterative calculation within the nested loop because the recursive transmission for results is continued. This calculation is widely used on the digital signal processing function. Therefore, the conventional architecture is not suitable for the latest digital signal processing functions any more. A reconfigurable computing (RC) is suggested to solve problem, but the adaptation to various applications is practically not on time [4-6].

In this paper, we propose an efficient parallel architecture with register-in-logic block. The proposed architecture has high performance and efficiency. Besides, the complexity of the proposed architecture is similar to the conventional processor

2. Proposed Architecture

A basic structure of the conventional processor is composed of an ALU, a register file, a controller, and a data bus. This structure makes the processor architecture more hierarchical. It is easy to improve the performance through the addition of logic blocks and the modification of the data routing. Because the register file and the logic block exist independently each other, it is too easy to apply the change of register set to the architecture. However, the conventional architecture has several problems; the data dependency, the difficult application to hardware parallelism, and the hardness to accelerate for specified

2.1 Attenuating Data Dependency

First of all, the data dependency is limit to improve the performance in a pipelining architecture. Many technical approaches are studied to reduce its effect in a *Writeback* process. The data dependency also creates the branch prediction problem that occurs the delay to make the decision whether the branch is taken or not. It causes the performance degradation. When the number of operands is small, the performance degradation by the data dependency is small and the technique reducing this effect is so simple.

However, when there are many operands, it is the critical bottleneck of the processor performance. In order to apply a hardware parallelism to the original architecture, number of the logic blocks must be increased. It can complicate the data bus and the register file that manages a register transmission. Practically, in the register file, an area of the data bus is more critical than that of a register itself. In the

register file, the data bus selects output registers according to the number of reading operands. Also, it selects input registers according to the number of writing operands. If many operands are defined, than the various applications can be offered. To the contrary, the hardware complexity increases exponentially. To solve this problem, a SMT (simultaneous multi-thread) architecture is proposed. It can increase the utility factor of processors but it causes the increment of the hardware complexity.

Next, the basic architecture of processors is unable to accelerate a specified function. It supports limited sub-routines or functions within the pre-defined instruction set. For the unsupported functions, we only have to solve it by flattening and reconfiguring them. It means that the applications for specified functions are not optimized and performed with low efficiency. This problem can be solved to raise the clock speed. But, the clock raise is constrained by a limitation of a manufacturing process for the semiconductor. The RC architecture makes the processors improve structural performance by using a programmable array platform. It guarantees high performance when its data path is rearranged according to the specified function or program. However, whenever a program is changed, an optimized data routing must be generated and set. Namely, the RC architecture is reprogrammable logic array without the consideration for the registers

2.2 Register-in-Logic

To solve the problems of the conventional architecture without the hardware overhead and the performance decrease, we propose an efficient architecture which has the modified logic block. It includes both the advantages of the conventional architecture and the parallel architecture. The main features are as follows:

- No register file
- Real-time reconfigurable data path
- Parallel processing
- Free for the data dependency

The register file makes many problems related the data transmission. We eliminate the register file to solve the data dependency. The complementary logic to the register file must be integrated the register anywhere. Therefore, there is no register file in the proposed architecture. Instead of the register file, a special logic block includes a register. The register inside is substituted for the function of the register file. It exists in the logic block, and contains a logic operation result. It is unnecessary to consume time for writing a result back into the register file. This architecture does not need the *writeback* sequence any more. Then it reduces an effect of the data dependency. As the result of an analysis for the conventional architectures, the number of the registers for storing the final result or outgoing value is very small. Most registers are used as the storages containing an intermediary computing result. It is time-consuming to schedule these registers. So, the proposed architecture locates these registers on each logic element and minimizes threads for scheduling them. We manage a few registers to report a final result like the conventional design.

The proposed architecture can be assembled lots of the modified logic element like the RC architectures. It can accelerate the system performance with the hardware parallelism. A 32-register system is implemented by grouping 16 logic elements having 2 registers inside individually. Generally, the register file is so complicated in the system with many registers. If number of the registers is increased, the complexity is also increased. Because the hardware complexity and path delay of data routing is greater than them of register, the increment of the register becomes a bottleneck. In the result of a logic synthesis, an area of pure register cells of the register file with 32 32-bit registers is similar to a routing area between registers. In additions, an area of the logic element is so small compared to the register file include a multiplier.

Figure 1 shows that the proposed logic element integrates 2 registers called register-in-logic element. There are an ALU, a multiplier, and two shifters. An architecture with a complete logic -the ALU, the multiplier, and the shifter- is the highest performance while the hardware complexity is too high. So, we must constrain an all-in-one style. Actually, we decide the integration of the multiplier according to required specification. We classify the logic element into a RegALU and a RegMUL. The RegALU is a basic element that contains the ALU and the shifter. The RegMUL is an expert element that is added the multiplier to the RegALU logic element. The type of logic element is decided according to the complexity and the performance required.

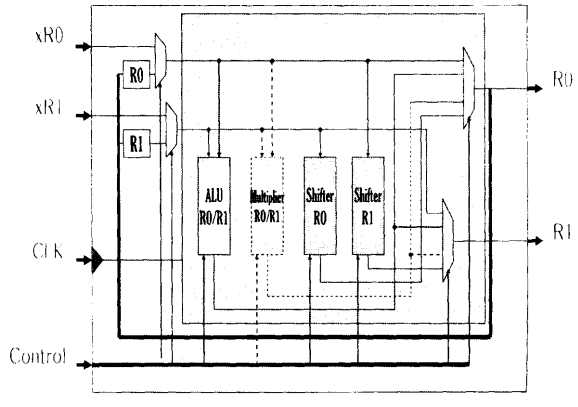


Fig. 1. Register-in-Logic element with 2 registers

Table 1 shows the area comparison for the register file and the logic element. Accordingly, the hardware complexity of the proposed architecture with 16 logic elements is not very large. A small complexity of the basic logic element attenuates the increase of the hardware overhead with the multiple logic elements. Therefore, the proposed architecture organizes a parallel structure without a sudden hardware increase.

Table 1. Area Comparison

Type	Area Synthesized
32 32-bit register file	1.2M
32-bit ALU	20K
32-bit Multiplier	265K
Register-in-Logic (ALU)	25K
Register-in-Logic (Multiplier)	272K

Figure 2 shows a 32-register system assembled by the register-in-logic logic elements. It is similar to the RC architecture carrying 16 processing elements. But, the RC and the proposed architecture are different from each other in meaning of a real-time modification of the data routing strategy. Besides, we configure the data path to be capable to connect from one logic element to any logic element. It makes an architecture more effective. Figure 3 shows the connectivity of the proposed parallel architecture is very effective. 4 left-side logic elements can connect 4 right-side logic elements for a few cycles. Naturally, other logic element can connect any logic element by the same method. For the correlated functions, it manifests the efficiency of this routing strategy.

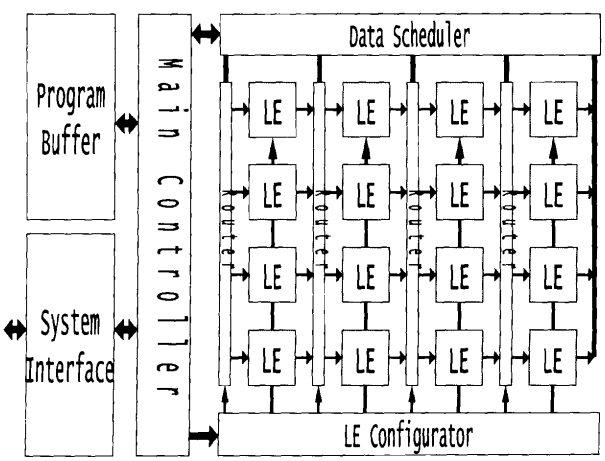


Fig. 2. 32-register DSP system with register-in-logic elements

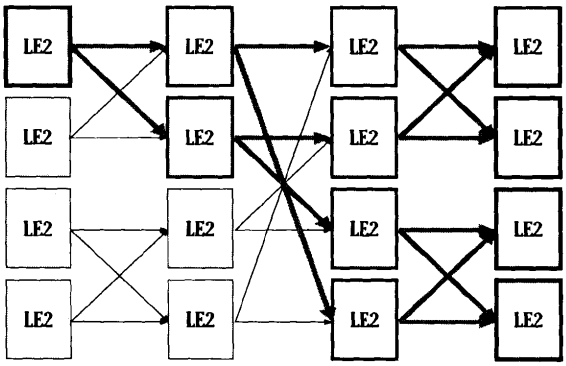


Fig. 3. Data-path structure for the proposed architecture

3. Performance Evaluation

In order to prove the effectiveness of the proposed architecture, we have performed simulation on the DSP applications because the application of the digital signal processing is a performance index for processors. We chose TMS320C6xx DSP series of Texas Instrument as a comparative architecture. The TMS320C6xx series consists of two multipliers, four ALUs, and two data-path units [7]. First, we compared the cycle counts of basic filter functions, FIR (finite impulse response) and IIR (infinite impulse response) filter functions. A generalized FIR filter of N filter coefficients, *h(k)*, is defined as:

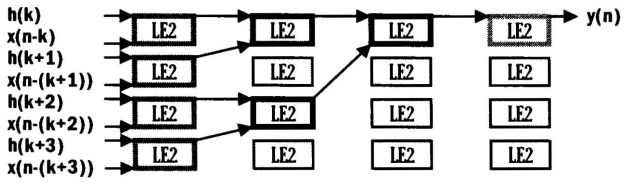
$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \qquad (1)$$

Figure 4 (a) shows the parallel structure of the FIR filter applied to the proposed architecture. Most filter functions have an iterative accumulation of the multiplications. In the proposed architecture, we layer the calculation into the multiplication and the accumulation. Each logic element row is performed in parallel. A 4-th order IIR filter is defined as:

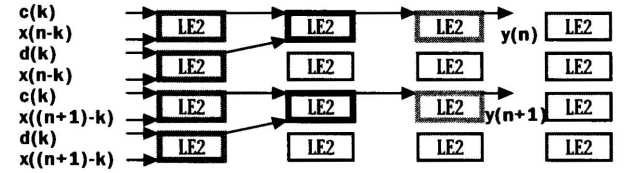
$$y(n) = \sum_{k=0}^{4} c(k)x(n-k) - \sum_{k=1}^{4} d(k)y(n-k) \qquad (2)$$

Figure 4 (b) shows the parallel structure of the IIR filter function. Compared to the FIR filter function, a structure of the IIR filter function is simple. Next, we compared cycle count for FFT (fast Fourier transform) application. Because the FFT function is more complex than the filter function, the parallel structure of the proposed architecture becomes more complicate. Figure 4 (c) shows the FFT application. The utilization of the logic element is high while the data path among the logic elements is more complicate. A radix-2 FFT is defined as:

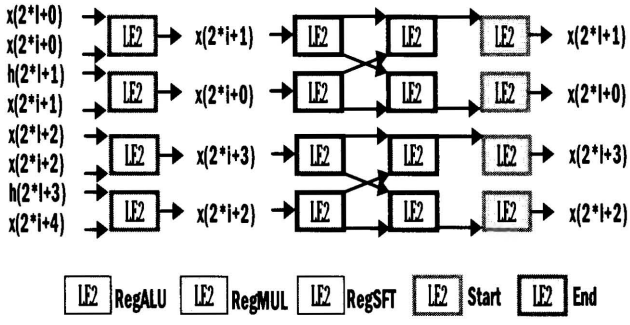
$$X(k) = \sum_{n=0}^{N-1} x_n W_N^{kn}, k = 0,..., N-1 \quad (3)$$



(a) FIR filter



(b) IIR filter



(c) Radix-2 FFT

Fig. 4. Parallel architectures for DSP functions

Table 2 shows the comparison result of cycle counts for each digital signal function. We choose the TI DSP processors to compare the performance because they are successful commercial processors with a high performance and a good architecture. We extract the cycle counts from library document offered. Since the extracted values are ideally optimized to the TI's core, there is some error to measure a real count in the practical program. Table 2 shows that the proposed architecture is 2~4 times faster than the commercial core. Most digital signal processing functions consist of the combination of the arithmetic and the multiplication. The FIR and the IIR filter functions are combined these formulas about half-and-half. For the filter functions, the proposed architecture is 2 times faster than the commercial product. The FFT function is organized by the combination of more complex formulas. But, in the FFT function, the proportion of the multiplication is comparatively high. For this case, the proposed parallel architecture is 4 times faster than the commercial core. It means that the register-in-logic element is more effective to process the complex functions.

Table 2. The performance comparison

DSP Function	Commercial DSP	Register-in-Logic	Improvement
FIR	2,068	803	258%
IIR	830	402	206%
FFT	4,250	1,122	379%

4. Conclusion

The conventional architecture of the processor consists of the logic block, the register file, and so on. It involves many problems like the data dependency because of the existence of the register file. And the recent RC architecture performs a reconfiguration optimized an individual program to obtain high performance. The basic architecture has simple, but low efficiency. A reconfigurable parallel architecture has high performance, but it is too difficult to process the real-time adaptation. To complete these problems, we combine advantages of the conventional architectures to make high efficient architecture. We propose a new parallel architecture having register-in-logic blocks. There is no register file that causes the data dependency. It has an efficient data path to increase functionality. And it can reconfigure a routing structure on time. We compare the cycle counts for some digital signal processing function with the commercial processor. The proposed parallel architecture is maximal 4 times faster than the conventional architecture. As results, the register-in-logic element system is more efficient to perform DSP applications without increasing the complexity.

References

[1] Nigen Horspool and Peter Gorman, The ASIC handbook, Prentice Hall, Inc., 2001.

[2] David A. Patternson and John L. Hennessy, Computer Organization & Design: The Hardware/Software Interface, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1997.

[3] Zhaohui Liu, Kevin Dickson, and John V. McCanny, "Application-Specific Instruc-tion Set Processor for SoC Implementation of Modern Signal Processing Algorithms", IEEE transactions on Circuits and Systems, Vol. 52, Issue 4, pp.755-765, April 2005.

[4] Kiran Bondalapati and Viktor K. Prasanna, "Reconfigurable Computing Systems", Invited paper on proceeding of the IEEE, vol.90, No.7, pp. 1201-1217, July 2002.

[5] Li-Hsun Chen, Oscal T.-C.Chen, and Ruey-Liand Ma, "A High-Efficiency Recon-figurable Digital Signal Processor for Multimedia Computing", Proceedings of In-ternational Symposium on Circuits and Systems, vol. 2, Session II, pp.768-771, May 2003.

[6] Chin-Long Wey and Jin-Fu Li, "Design of Reconfigurable Array Multipliers and Multiplier-Accumulators", Proceedings on IEEE Asia-Pacific Conference on Circuits and Systems, Vol. 1, pp.37-40, Dec. 2004.

[7] "Signal Processing Examples Using TMS320C64x: Digital Signal Processing Library (DSPLIB)", Application Report, Texas Instruments, http://www.ti.com/ZhaD. Awduche, "Requirements for Traffic Engineering Over MPLS", *IETF Request for Comments*, RFC 2707, September 1999.

Sung-il Bae received his B.S. degree in electronic engineering from Kyungpook National University, South Korea, in 1998. And, he received his M.S. degree in electrical & electronic engineering from Yonsei University in 2000. He is pursuing his Ph.D. degree in the same department at the same university. His research areas include the network processor design and ARM system.

Hyun-Jin Kim received his B.S. and M.S. degrees in electronic engineering from Yonsei University, South Korea, in 1997 and 1999, respectively. In 1999, he joined Samsung Electro-Mechanics, where he designed analog devices. In 2004, he leaved a company. Since 2004, he is pursuing his Ph.D. degree in the same department at Yonsei University. His research areas include the reconfigurable computing and DSP processors.

Hyuntae Park received his B.S. in electric and electronic engineering from Yonsei University in 2004. And, he is pursuing his M.S. degree in the same department at the same university. His research areas include the adaptive routing and the network look-up algorithm.

Sungho Kang received B.S. degree from Seoul National University in 1986 and the M.S. and Ph.D. degrees from the University of Texas, Austin, in 1988 and 1992, respectively. In 1989 he joined Schlumberger and worked on a computer science as a member of research scientist. In 1992 he joined Motorola, and worked on a semiconductor systems design technology as a member of senior staff engineer. Since 1994, he is currently a professor in the Yonsei University, Seoul, Korea. His research focuses on the VLSI design and testing.