

An Efficient Variable-Length Tap FIR Filter Chip

Sung Hyun Yoon and Myung H. Sunwoo

School of Electrical and Electronic Eng.
Ajou Univ.
Suwon, Kyungki-Do, 442-749
Tel : +11-82-331-219-2369
Fax : +11-82-331-212-9531
email : sunwoo@madang.ajou.ac.kr

Abstract - This paper proposes a novel VLSI architecture for an FIR filter chip providing variable-length taps. To change the number of taps, we propose two special features called a data-reuse structure and a recurrent-coefficient scheme. These features consist of several MUXs and registers and reduce the number of gates over 20 % compared with existing chips using an address generation unit and a modulo unit. Since parallel multipliers occupy a large VLSI area, a filter chip using bit-serial multipliers meeting the real-time requirement specification can dramatically save the area. We propose a modified bit-serial multiplication algorithm to compute two partial products in parallel, and thus, the proposed filter can be twice faster than previous filters using bit-serial multipliers. We developed VHDL models and performed logic synthesis using the SYNOPSISTM CAD tool with the 0.8 μ m SOG cell library (HSG30042). The chip has only 9,507 gates, was fabricated, and is running at 77 MHz.

I. INTRODUCTION

Digital filters can be divided into two categories: finite impulse response (FIR) filters; and infinite impulse response (IIR) filters. Although FIR filters, in general, require higher taps than IIR filters to obtain similar frequency characteristics, FIR filters are widely used because they have linear phase characteristics, guarantee stability and are easy to implement with multipliers, adders and delay elements [1,2].

The number of taps in digital filters varies according to applications. In commercial filter chips with the fixed number of taps [3], zero coefficients are loaded to registers for unused taps and unnecessary calculations have to be performed. To alleviate this problem, the FIR filter chips providing variable-length taps have been widely used in many application fields [4-6]. However, these FIR filter chips use memory, an address generation unit, and a modulo unit to access memory in a circular manner.

The paper proposes two special features called a data-reuse structure and a recurrent-coefficient scheme to provide variable-length taps efficiently. Since the proposed architecture only requires several MUXs, registers, and a feedback-loop, the number of gates can be reduced over 20 % than existing chips [4-6].

Since parallel multipliers occupy a large VLSI area, eliminating multipliers is desirable as long as the real-time requirement specification is satisfied. To perform multiplications without using parallel multipliers, bit-serial methods [7,8] and the Quasi-serial method [10,11] have been proposed. The bit-serial method executes add-and-shift operations which need an M -bit register, two $2M$ -bit registers and a $2M$ -bit adder for an $M \times M$ -bit multiplication. This algorithm takes M cycles for an $M \times M$ -bit multiplication. The Quasi-serial method offers an alternative to add-and-shift techniques for a two's complement multiplication. The hardware size of the Quasi-serial method is bigger than that of the bit-serial method [7,8]. The algorithm takes $2M - 1$ cycles for an $M \times M$ -bit multiplication.

To reduce the number of cycles, we propose the modified bit-serial algorithm producing two partial products in parallel, which is twice faster than previous filters using bit-serial multipliers [7,9] and has small hardware overhead.

This paper is organized as follows. Section II describes the proposed bit-serial algorithm for FIR filters and section III describes the new VLSI architecture. Section IV presents the chip implementation of the proposed filter and its performance evaluation. Finally, section V contains concluding remarks.

II. THE PROPOSED BIT-SERIAL ALGORITHM

In general, FIR filtering is described by a simple convolution operation as expressed in the equation (1)

$$y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n-k] \quad (1)$$

where $x[n]$, $y[n]$, and $h[n]$ represent data input, filtering output, and a coefficient, respectively and N is the filter order.

The equation using the bit-serial algorithm for a FIR filter can be represented as

$$y[n] = \sum_{k=0}^{N-1} \sum_{j=0}^{M-1} (h_j[k] \cdot 2^j) \cdot x[n-k] \quad (2)$$

where the h_j , N and M are the j^{th} bit of the coefficient h , the

number of taps and the number of coefficient bits, respectively. The bit-serial algorithm [7,8] sequentially inspects bits of a multiplier from the least significant bit (LSB) to the most significant bit (MSB). If the inspected bit is '1', an M -bit multiplicand is added to the most significant half word of a double length accumulator ($2M$ -bit register). Otherwise, the addition is not performed. The content of the accumulator is shifted one bit to the right whenever each bit of the multiplier is inspected. After all bits of the multiplier are considered, the accumulator contains the final product.

To reduce the number of multiplication cycles, we modified the bit-serial algorithm. Multiplier bits can be separated into even bits and odd bits. Based on the equation (2), the proposed filter equation can be rewritten as follows

$$y[n] = \sum_{k=0}^{N-1} \sum_{j=0}^{\frac{M}{2}-1} (h_{2j}[k] \cdot 2^{2j} + h_{2j+1}[k] \cdot 2^{2j+1}) \cdot x[n-k] \quad (3).$$

Two separated bits are stored in two $M/2$ -bit shift registers. To generate two partial products simultaneously, the M -bit multiplicand is logically anded (AND) by two shifted multipliers. Accordingly, we can obtain two partial products in every cycle.

The bit-serial algorithm [7,8] requires $N \times M$ [7,9] cycles. In contrast, the proposed algorithm can reduce the number of cycles to $N \times M/2$.

III. THE PROPOSED ARCHITECTURE

The proposed architecture providing variable-length taps mainly consists of the data-reuse structure for a data block and the recurrent-coefficient scheme for a coefficient block. We implement the chip having the number of taps up to 64 and the word-length of data and coefficients is 8-bit. However, the numbers of filter taps and data bits can be increased.

A. The data-reuse structure for data block

Fig. 1 shows the proposed data-reuse structure that consists of sixty-four 8-bit registers, one 8-bit 8×1 MUX, and eight MUX cells (MC1 ~ MC8). Each MUX cell consists of two 2×1 MUXs. To get the N^{th} order filtering output, $N-1$ previous input values should be reused by using a delay line or a circular buffer. The data-reuse structure feeds $N-1$ previous input values to the computation unit and also feeds them to a circular buffer.

The circular buffer is composed of $N-1$ registers, a feedback-loop and MUX cells. The 8×1 MUX selects a register among the 56th register through the 63rd register, and transfers the value of the selected register to the 64th

register and also to the feedback-loop at the same time. One of MUX cells determined by N receives the input data through the feedback-loop, and transfers the current input data to one of the top registers in Fig. 1 at every N^{th} cycle. The other MUX cells make the delay line by connecting registers. Therefore, the number of filter taps determines the number of registers connected in a circular manner.

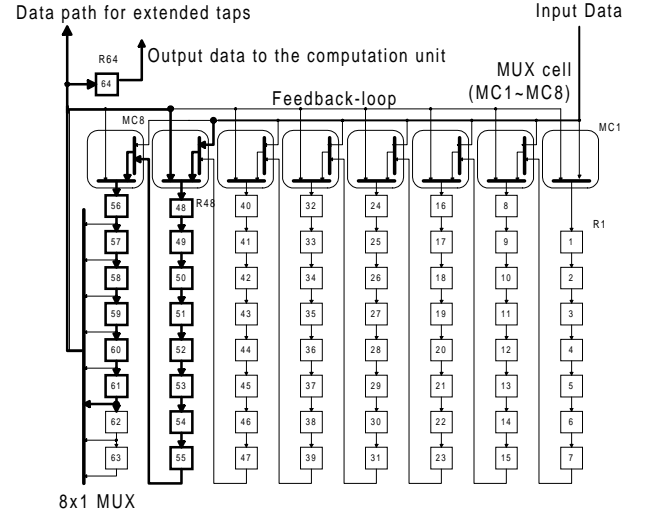


Fig. 1 The data-reuse structure for data block

For example, assume that the number of taps is 15. The input data are $D_0, D_1, \dots, D_k, \dots$, and the coefficients are C_0, C_1, \dots, C_{14} . For 15 tap filtering, 15 registers form a circular buffer which are the registers in bold boxes shown in Fig. 1. Only 15 input data, $D_k \sim D_{k+14}$, are required to get the k^{th} filtering output and $D_{k+1} \sim D_{k+15}$ are required for the $k+1^{\text{th}}$ filtering output. After calculating the k^{th} filtering output, D_k is eliminated and the new input data D_{k+15} is loaded into the circular buffer. The external control pins can set the connection for the circular buffer.

Since the data-reuse structure supplies data for the computation unit, the proposed architecture does not require an address generation unit and a modulo unit used in existing filter chips [4-6]. In addition, this feature requires smaller hardware having an 8×1 MUX, eight MUX cells, and the feedback-loop. When the number of taps increases to 128, only eight more MUX cells and 64 more registers are required. As the number of taps increases with the power of 2, the hardware size increases linearly. Therefore, it can have good scalability.

B. The recurrent-coefficient scheme for coefficient block

Fig. 2 shows the coefficient block consisting of sixty-four 8-bit registers. The number of MUX cells in the recurrent-coefficient scheme can be reduced compared with

the data-reuse structure. If the number of taps (N) is less than 32, we can modify N -tap to Nr -tap as repeating r times of N -tap to make larger than 33-tap based on the equation (4)

$$33 \leq Nr \leq 64 \quad (4).$$

Since the number of taps less than 32 can be rearranged between 33-tap and 64-tap by repeating taps, only four MUX cells are required.

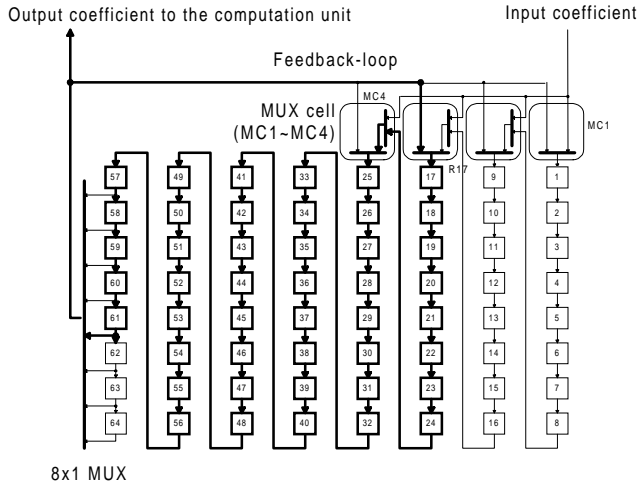


Fig. 2 The recurrent-coefficient scheme for coefficient block

For example, when the coefficients for the 15-tap FIR filter are C_0, C_1, \dots, C_{14} , the scheme repeats the coefficients 3 or 4 times to make the 45- or 60-tap filter. If we select the 45-tap filter, the first repeated coefficients are stored in the registers from 61st to 47th shown in Fig. 2. The second repeated coefficients are stored in the registers from 46th to 32nd, and the third repeated coefficients are stored in the registers from 31st to 17th. This feature can make the circular buffer containing three repeated coefficient sequences from 17th to 61st through the feedback-loop. The external control pins can set the connection for the circular buffer.

The coefficient block shown in Fig. 2 requires only one 8×1 MUX and four MUX cells, because of the recurrent-coefficient scheme. We can implement the 128-tap filter by adding four more MUX cells and 64 registers. In data and coefficient blocks, regardless of the number of taps the critical path is always the same and is appeared on one 8×1 MUX and one 2×1 MUX. Hence, we can increase the number of taps without sacrificing the clock speed.

C. The computation unit

Fig. 3 shows the pipelined computation unit that is composed of two 4-bit Parallel-Input-Serial-Output Shift-

Right registers (PISO-SR1, PISO-SR2), two Partial Product Generators (PPGs), a Partial Product Summation (PPS) unit, 16-bit Group CLA adder and registers.

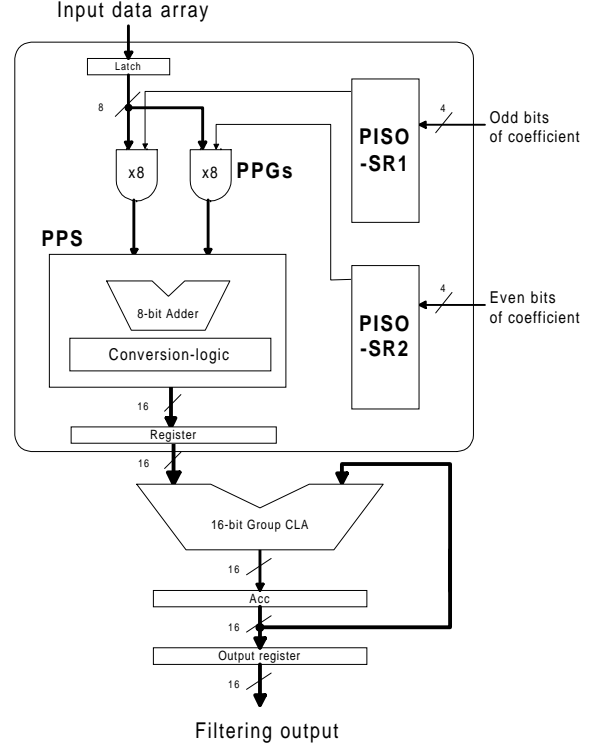


Fig. 3 The computation unit

The bits in a coefficient are separated into even bits and odd bits to generate two partial products at the same time. The separated coefficient bits are stored in two 4-bit PISO-SRs and are shifted out to PPGs. The input data from the data block is stored in the 8-bit latch. Two PPGs generate two partial products every cycle via logical AND operations of data bits and coefficient bits. PPS adds two partial products and the results of PPS are converted to the 16-bit stream at the conversion-logic. The odd operand is shifted 1-bit left and is applied to the 8-bit adder in PPS. This feature can eliminate a shift register in PPS, and thus, there is no timing and hardware overhead for a shift operation.

The 16-bit group CLA adds the converted stream to the accumulated partial products. A multiplication requires 4 cycles and the N -tap filtering output is obtained in $4N$ cycles. The accumulator (Acc) transfers the accumulated value to the output register after $4N$ clock cycles to obtain the filtering output.

The proposed architecture can perform twice faster than previous filters using bit-serial multipliers [7,9] due to the proposed algorithm, and thus, it can reduce filtering cycles by half.

IV. THE CHIP IMPLEMENTATION AND PERFORMANCE EVALUATION

We implemented behavior and structure models of the proposed architecture using the VHDL and simulated the models using the SYNOPSISTM CAD tool. First, the behavior models realize every unit component, and then these components are mapped using the structure models. We simulated the filtering operation supporting variable-length taps.

For gate-level logic synthesis, we optimized the VHDL models using the SYNOPSISTM CAD tool. We used the HyundaiTM 0.8 μm SOG cell library (HSG30042) and performed function and timing simulations using the CADENCETM. We verified the results between the VHDL models and the synthesized models. The total gate count for the proposed FIR filter chip is only 9,507 and the clock frequency is 77 MHz. Fig. 4 shows the layout of the implemented FIR filter chip.

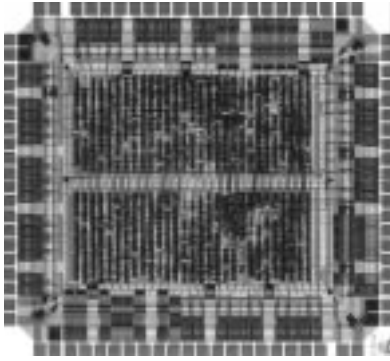


Fig. 4 The layout of the chip

The existing chips [4-6] supporting variable-length taps certainly consist of more than 890 gates except the storage devices, such as memories based on the estimated numbers of gates for various components described in [12]. However, our architecture consists of 712 gates except registers. In addition, the existing computation unit using the bit-serial algorithm [7] is estimated about 800 gates based on [12]. The number of gates for the proposed computation unit requires 882 gates but we can improve the processing speed by two times with little hardware overhead.

For a 64-tap FIR filter, this chip can operate at 77 MHz and can provide 313 Ksamples/second. For a 2-D convolution filter, the chip can provide 2.293 Mpixels/second and can process 35 frames for 256 x 256 pixels in real-time. The chip can be extended for larger size filters or for high speed 2-D convolution filters. If 256 taps are required for 1-D signal processing, we can use four filter units connected in serial shown in Fig. 5 and obtain 313 Ksamples/second.

In addition, three chips connected in parallel for a 2-D

convolution filter shown in Fig. 6 can obtain 7.078 Mpixels/second and process 27 frames for 512 x 512 pixels in real-time. To improve processing speed, we can increase the number of computation units as shown in Fig. 7. These different configurations can be implemented in a chip since one filter unit has only 9,507 gates and one computation unit has only 882 gates.

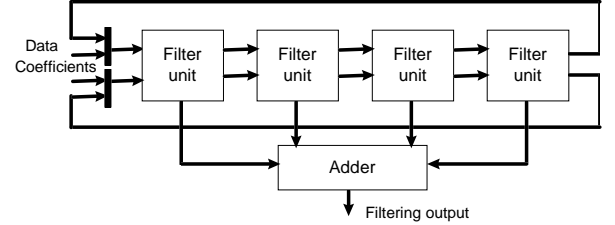


Fig. 5 The extended 256-tap filter

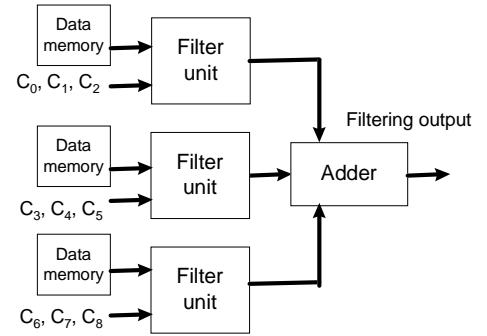


Fig. 6 A 2-D convolution filter with a 3 x 3 window

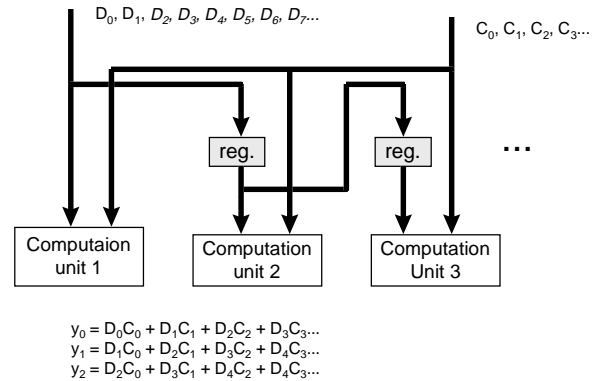


Fig.7 The parallel computation units

V. CONCLUSIONS

This paper has proposed the new VLSI architecture for a FIR filter chip providing variable-length taps and presented its chip design and implementation. We verified that the new architecture features, such as the data-reuse structure and the recurrent-coefficient scheme, can reduce the number

of gates about 20 %. In addition, the computation unit using the modified bit-serial algorithm performs twice faster than previous filters using bit-serial multipliers without increasing the hardware size. We have actually implemented the proposed chip, is running at 77 MHz, and is completely verified.

REFERENCES

- [1] Sanjit K. Mitra and James F. Kaiser, Handbook Digital Signal Processing, WILEY-INTERSCIENCE, 1993.
- [2] John G. Proakis and Dimitris G. Manolakis, Digital Signal Processing : Principles, Algorithms, & Applications, 1996.
- [3] Harris Semiconductor Inc., Digital Signal Processing Databook, HSP43168, 1994.
- [4] Motorola Inc., Motorola Semiconductor Technical Data, DSP56200.
- [5] Harris Semiconductor Inc., Digital Signal Processing Databook, HSP4312, HSP43481, 1994.
- [6] Advanced RISC Machines Limited (ARM), ARM7DM, 1996.
- [7] Robert E. Morley, Jr., Gray E. Christensen, Thomas J. Sullivan, Orly Kamin, "The Design of a Bit-Serial Coprocessor to Perform Multiplication and Division on a Massively Parallel Architecture," in Proc. IEEE, The 2nd Symposium on the Frontiers of Massively Parallel Computation, Fairfax, U.S.A., pp. 419-422, Oct. 1998.
- [8] Israel Koren, Computer Arithmetic Algorithms, pp. 29-32, 1993.
- [9] Woo Jin Oh and Yong Hoon Lee, "Implementation of Programmable Multiplierless FIR Filters with Powers-of-Two Coefficients," IEEE Trans. on Circuits and Syst., vol. 42, no. 8, Aug. 1995.
- [10] Earl E. Swartzlander, Jr., "The Quasi-Serial Multiplier," IEEE Trans. Comput., vol. C-22, pp. 317-321, Apr. 1973.
- [11] T. G. MaDanel and R. K. Guha, "The Two's Complement Quasi-Serial Multiplier," IEEE Trans. Comput., C-24, pp. 1233-1235, 1975.
- [12] S. Ward, P. Barton, J. B. G. Roberts, and B. J. Stanier, "Figure of merit for VLSI implementations of digital signal processing algorithms," Proc. Inst. Elec., vol. 131, Part F, pp. 64-70, Feb. 1984.