100% OPERATIONAL EFFICIENT BIT-SERIAL PROGRAMMABLE FIR DIGITAL FILTERS

P. Kalivas, A. Tsirikos, P. Bougas, and K.Z.Pekmestzi

School of Electrical & Computer Engineering Department Computer Science, National Technical University of Athens,
9 Heroon Polytechneiou, Zographou Campus 157 80 Athens, Greece
phone: + 30 210722500, fax: + 30 210722428, email: {paraskevas, andreas, paul, pekmes}@microlab.ntua.gr
web: www.microlab.ntua.gr

ABSTRACT

A new scheme for the implementation of programmable FIR digital filters with 100% operational efficiency is presented in this paper. The term *100% operational efficiency* implies that no zero bits have to be inserted between successive input data words in order the filter input to be synchronized with the filter output. Both the input data and the filter output are in two's complement LSB-first bit-serial form. The coefficients are in two's complement bit-parallel form. All the intermediate results and the filter output are produced and handled in full precision. The proposed scheme is based on a special serial-parallel multiplier that operates with 100% efficiency. We exploit the internal registers and the free accumulation input in this multiplier to reduce the hardware complexity of the filter significantly. The proposed scheme is compared from the aspect of hardware complexity and efficiency with other bit-serial schemes.

1. INTRODUCTION

Many practical applications require FIR digital filters with large number of taps. The parallel implementations of such filters require a vast amount of hardware. When the filter coefficients are constant numbers significant hardware reductions can be achieved by eliminating the hardware that corresponds to zero coefficient bits [1-3]. When the application requires the filter coefficient to change on the fly, programmable FIR filters are required and discrete parallel multipliers must be used for the implementation of taps. Therefore, fully parallel implementations are unsuitable in such cases.

On the other hand, when the number of filter taps is relatively small the hardware implementation of parallel schemes is feasible. However, there are DSP applications, like the processing of audio signals, where the sample rate is significantly low compared to the operational frequency of modern circuits. In such applications bit-serial implementations are more hardware efficient and have significantly lower power dissipation than the parallel. The later is a crucial factor in modern designs due to the widespread use of mobile devices. Moreover, the decreased area of the serial modules leads to low power consumption due to decreased leakage current that in the sub-micron technology becomes the dominant factor. Consequently, the efficient design of bit-serial FIR digital filters is worthy.

In this paper we propose a bit-serial filter scheme based on a serial-parallel multiplier that operates with 100% efficiency, namely no zero bits are required between input data words. Also, in the proposed scheme all the operations are perform in full precision. It is based on a modified version of a special serial-parallel multiplier presented in [4] that operates with 100% efficiency. By exploiting the internal registers and the free accumulation input of this multiplier we achieve to reduce the total number of required delay element and consequently the total required hardware.

The paper is organized as follows. In Section 2 we present bit-serial FIR filter architectures. The detailed implementation of the proposed filter scheme is given in Section 3. In Section 4 we compare the proposed scheme with the schemes presented in Section 2.

2. BIT-SERIAL FIR FILTER ARCHITECTURES

In Fig. 1a and 1b the structure of a *k*-tap FIR filter is shown in direct and transpose form.

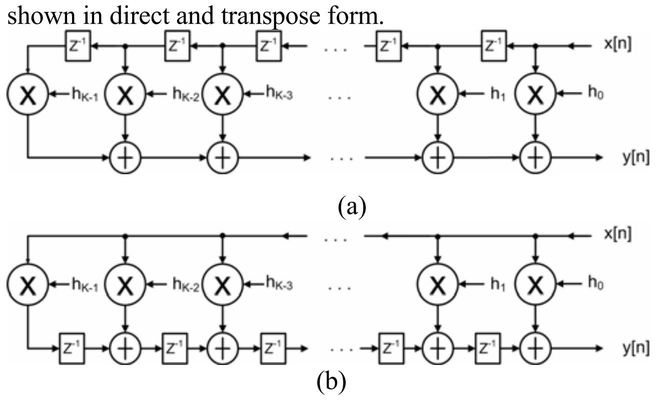
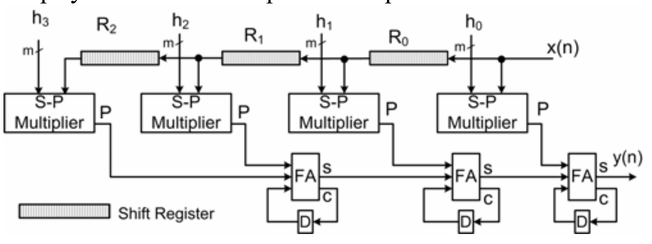


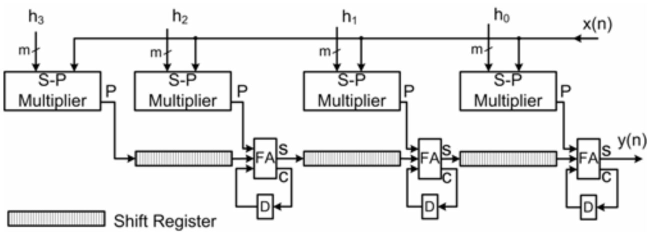
Fig. 1 The structure of *k*-tap FIR filter in (a) direct (b) transpose form.

Throughout the paper the following notation is used. The bit-lengths of the input data *x(n)* and the filter coefficients h_i are represented as *w* and *m* respectively. The bit-length of the full precision intermediate product P_i is $w+m$. The intermediate sums and the filter output are $w+m+\log_2 k$ bit wide. The extra $\log_2 k$ bits are required for avoiding the accumulation overflow. The combination of a multiplier with an adder, which is the basic element of a FIR filter structure, is called Multiply-Accumulate (M-A) unit in the rest of the paper.

The bit-serial implementations of the structures in Fig. 1a and 1b for $k = 4$ are given in Fig. 2a and 2b. The shift registers shown in the figures correspond to the delay elements in Fig. 1. Their bit-length is $w + m + \log_2 k$ bits, equal to bit-length of the filter output. $m + \log_2 k$ zero bits must be inserted between successive input data words to keep synchronized the input and output of the filter.



(a)



(b)

Fig. 2 Bit-serial implementation of a 4-tap FIR filter in (a) direct form (b) transpose form.

The major disadvantage of the circuits in Fig. 2 is the need for $m + \log_2 k$ zeros between successive input data words. Thus, the operational efficiency of the circuit drops to 50% (assuming $m + \log_2 k \simeq w$). A way to get around this problem is to use a serial-parallel multiplier that operates with 100% efficiency. Such a multiplier is presented in [4]. The detailed circuit of this multiplier is shown in Fig. 3.

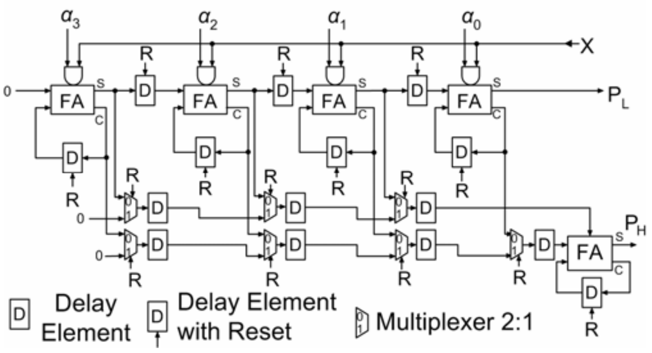
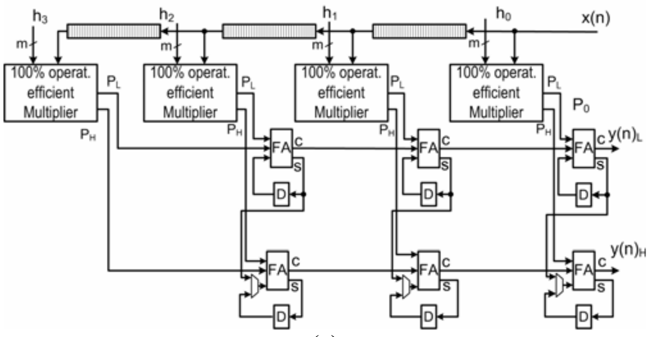


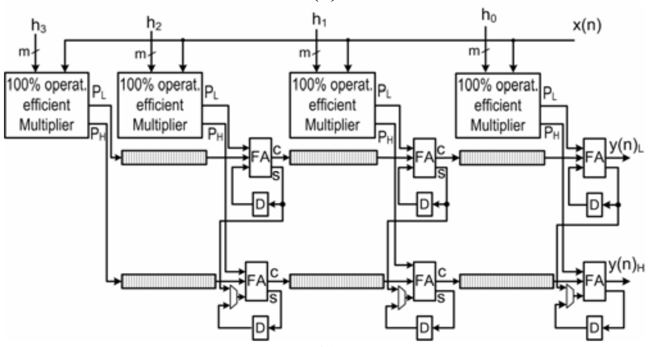
Fig. 3 The 100% operational efficient serial-parallel multiplier presented in [4].

It consists of a serial-parallel (S-P) multiplier, two shift registers, and a bit-serial adder. The multiplication cycle lasts $w + m$ clocks where w and m are the bit lengths of the multiplicand X and the multiplier a respectively. During the first w clocks the least significant part of the product is being obtained from the output P_L . At the w-th clock cycle, the carry and sum delay elements of the S-P multiplier contain the most significant part of the product in carry-save form. At this clock cycle the control signal R is activated (becomes high) and downloads the most significant part into the shift registers. During the last m clocks the contents of the shift registers are being shifted through the bit serial adder and the most significant part of the product is obtained in binary form from the output P_H while the S-P multiplier is involved in the computation of the next product. For simplicity, the unsigned version of the multiplier in [4] is show in Fig. 3. But the same circuit with slight modifications can be used for two's complement multiplication.

A direct and a transpose FIR filter implementation based on the above multiplier are shown Fig. 4. These implementations operate with 100% efficiency. The most and least significant parts of the result $y(n)_H$ and $y(n)_L$ are obtained from different outputs. The bit-length of registers shown in Fig. 4 is w assuming that $m + \log_2 k \le w$. This assumption is true for the most practical FIR filter applications.



(a)



(b)

Fig. 4 A bit-serial implementation of a 4-tap FIR filter with 100% operational efficiency in (a) direct form (b) transpose form.

3. THE PROPOSED FIR FILTER SCHEME

The disadvantage of the circuits in Fig. 4 is the large number of delays, especially in the transpose form. The direct form requires fewer shift registers but has the serious disadvantage of the increased combinational delay due to the direct connection of the bit serial adders in the accumulation line. A pipelined adder tree has been proposed [5-6] for accumulating the intermediate results. However, it leads to circuits with latency proportional to the number of filter taps. Thus, we focus on the design of FIR filters in transpose form since they are of immediate response and have minimum combinational delay.

In this paper we suggest the exploitation of the free sum input at the left end of the multiplier in Fig. 3 as well as its shift registers in order to achieve an implementation with significantly lower hardware complexity compared to that in Fig. 4b. Therefore, we use a modified version of the multiplier in Fig. 2 that is shown in Fig. 5.

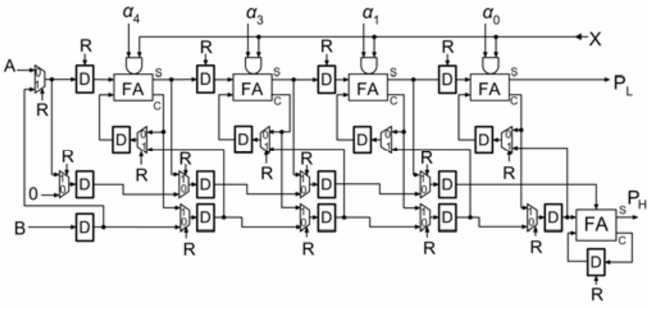


Fig 5. The modified multiplier of [4] that is used as M-A unit in the proposed filter scheme.

The major modification in relation to the multiplier in Fig. 3 is that when the most significant part of the result is downloaded into the two shift registers the content of the lower shift register is uploaded into the carry delay elements of the S-P multiplier. In other words, the content of lower shift register is interchanged with the contents of the carry delay elements of S-P multiplier.

By exploiting the free sum input, which is denoted with *A* in Fig. 5, and the bit-serial adder that converts the most significant part of the result in binary form, we can use the above circuit as M-A unit. The proposed filter that uses the multiplier in Fig. 5 as M-A unit is shown in Fig. 6. Also, in this circuit, we use the lower shift register of the modified multiplier to store the intermediate results produced by each M-A unit. Thus, the bit-length of external registers $R_{L,i}$ and $R_{H,i}$ is reduced to $w-m$ bits.

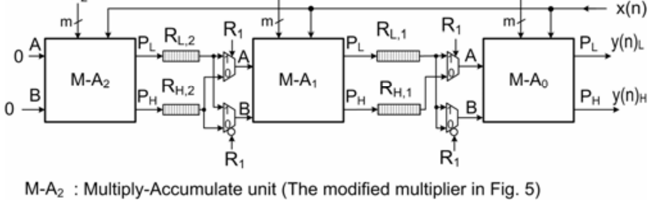


Fig. 6. The proposed bit-serial scheme for a 3-tap FIR filter.

A description of the operation of the circuit in Fig. 6 follows. We denote the intermediate result, that is produced by M-A$_i$ unit when x(n) enters the circuit, by P_i^n. Also, we denote the most and least significant part of P_i^n by $P_{i,L}^n$ and $P_{i,H}^n$. For the sake of the discussion, we consider $P_{i,L}^n$ and $P_{i,H}^n$ consisted of two parts according to the following equations:

$$P_{i,L}^n = P_{i,L1}^n 2^m + P_{i,L0}^n \, , \; P_{i,H}^n = P_{i,H1}^n 2^m + P_{i,H0}^n \, . \qquad (1)$$

According to these equations $P_{i,L0}^n$ and $P_{i,H0}^n$ consist of the *m* least significant bits of $P_{i,L}^n$ and $P_{i,H}^n$ respectively while $P_{i,L1}^n$ and $P_{i,H1}^n$ consist of the $w-m$ most significant bits. Since we have a transpose FIR filter, P_i^n must be added to P_{i-1}^{n+1}. Upon the completion of the extraction of $P_{i,L}^n$ from M-A$_i$ unit, $P_{i,L0}^n$ is stored in the lower shift register of M-A$_{i-1}$ while $P_{i,L1}^n$ is in register $R_{L,i}$. When M-A$_{i-1}$ starts the computation of P_{i-1}^{n+1}, $P_{i,L0}^n$ is uploaded from the lower shift register into the carry delay elements of M-A$_{i-1}$. Thus, the quantity $P_{i,L0}^n$ is added to P_{i-1}^{n+1} in parallel, as initial value of the carry delay elements. The next two parts of P_i^n, namely $P_{i,L1}^n$ and $P_{i,H0}^n$ (in this turn) are added serially to P_{i-1}^{n+1} during the next *w* clock cycles through the input *A* of M-A$_{i-1}$. The last part of P_i^n, namely the term $P_{i,H1}^n$, represents the accumulation overflow bits (we always assume $m + \log_2 k \le w$). This term is appended to the most significant part of P_{i-1}^{n+1} through the input *B* of M-A$_{i-1}$, when the most significant part of P_{i-1}^{n+1} has been downloaded into the shift registers and is being shifted through the bit-serial adder.

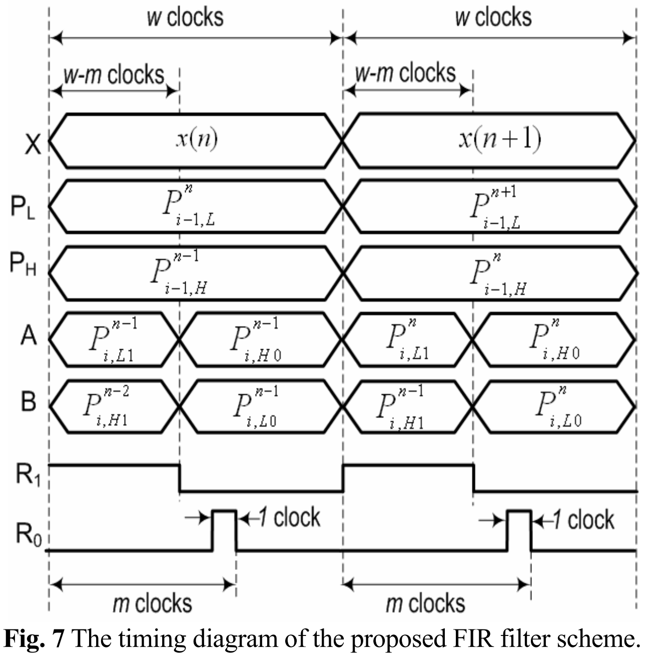


Fig. 7 The timing diagram of the proposed FIR filter scheme.

The above description is clarified by the timing diagram in Fig. 7. This diagram shows the quantities that enter the

Table 1 Comparison of filter schemes.

Filter Scheme	Hardware complexity per filter tap	Transistors (w=16, m=12)	Operation	Efficiency (Throughput/ Hardware)*1000
Fig. 2b.	$(m+1)\,FA + mAND + (3m+1)\,D$	926	50%	0,526
Fig. 4b	$(m+3)\,FA + mAND + (2m+1)\,D^* + (4m+1)\,D + (2m+1)\,MUX$	1812	100%	0,544
Propose scheme (Fig. 6)	$(m+1)\,FA + mAND + (m+1)\,D^* + (2w+1)\,D + (3m+1)\,MUX$	1344	100%	0,731

FA: Full-Adder (22 Tr), AND: 2-input AND gate (6 tr), D: Delay Element (16 Tr) D*: Delay Element with Reset (20 tr) MUX: Multiplexer 2:1 (6 tr) [8]

M-A$_{i-1}$ unit and the results obtained from its outputs. It also shows the timing of the control signals R_1 and R, where R is the signal used in M-A$_{i-1}$ for downloading the most significant part (shown in Fig. 5).

A disadvantage of the circuit in Fig. 6 is the broadcasting of the lines for input data and the control signals R_1 and R. A solution to this problem is to apply retiming [7]. We can remove delay elements from the shift registers shown in Fig. 6 and insert new ones into the lines of *x(n)*, R_1 and R by drawing vertical retiming cuts between adjacent M-A units. Another solution is to use the systolic version of the multiplier in Fig. 3 which is also presented in [4].

4. COMPARISON

In Table 1 the proposed scheme is compared with the filter schemes presented in Fig. 2b and Fig. 4b from the aspect of hardware complexity and efficiency. We include only the transpose filter forms in this comparison. Because the filter schemes have different operational efficiency we use the quantity *E=Throughput/Hardware*1000* as measure of the circuit performance.

Table 1 reveals the clear advantage of the proposed scheme. The second scheme in Table 1 achieves 100% operational efficiency but requires twice as much hardware as the first scheme in this table. Therefore it has about the same circuit performance as the first scheme. On the other hand, the proposed filter has about the one third of the delay elements of the second scheme and consequently its circuit performance is increased by 45-50% compared with the other two schemes.

5. CONCLUSION

In this paper we propose a 100% operational efficient bit-serial FIR filter based on a special multiplier, which also operates with 100% efficiency. We exploit the internal shift registers and the free sum input of this multiplier and use it as a full multiply-accumulate unit. This technique yields a filter with far more higher circuit performance, since it achieves 100% operational efficiency with only 40% increase in hardware complexity compared with a bit-serial FIR filter scheme that operates with 50% efficiency.

REFERENCES

[1] R.A. Hawley, B.C. Wong, T. Lin, J. Laskowski, and H. Samueli, "Design Techniques for Silicon Compiler Implementations of High-Speed FIR Digital Filters," *IEEE Journal of Solid-State Circuits*, Vol. 31, pp. 656-667, May 1996.

[2] R. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 677-688, Oct. 1996.

[3] J. B. Evans, "Efficient FIR filter architectures suitable for FPGA implementation," *IEEE Transactions on Circuits and Systems II*, Vol. 41, pp. 490-493, 1994.

[4] K. Pekmestzi, P. Kalivas and N. Mosxopoulos, "Long unsigned systolic serial multipliers and squarers," *IEEE Trans. On Circuits Syst. II*, vol. 48, no.3, pp. 316-321, Mar. 2001.

[5] C. J. Nicol., P. Larsson, K. Azadet and J. H. O'Neill, " A low-power 128-tap digital adaptive equalizer for broadband modems", *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, November 1997, pp. 1777-1790.

[6] J.-R. Choi, L.-H. Jang, S.-W. Jung and J.-H. Choi, "Structured design of a 288-tap FIR filter by optimized partial product tree compression", *IEEE Journal of Solid-State Circuits*, vol. 32, no 3, March 1997, pp. 468-467.

[7]K. K. Parhi, *VLSI Digital Signal Processing Systems*. John Willey & Sons Inc., 1999.

[8] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Addison-Wesley Publishing Company, 1994.