

## Low-Complexity Design of FIR Filter Implementation

<sup>1</sup>R.VENKATESH , <sup>2</sup>SANTHOSH KUMAR

<sup>1</sup>Department of Electronics and Communication Engineering  
CMR Engineering College, Hyderabad. [Venkateshrapolu2008@gmail.com](mailto:Venkateshrapolu2008@gmail.com)

<sup>2</sup>Department of Electronics and Communication Engineering AVANTHI Engineering college, HYDERABAD.  
[Santhosh\\_gangi@yahoo.com](mailto:Santhosh_gangi@yahoo.com)

### Abstract—

this paper presents a programmable digital finite impulse response (FIR) filter for low-power applications. A 10-tap programmable FIR filter was implemented and fabricated in CMOS 0.25- m technology based on the proposed architectural and circuit-level techniques. The chip's core contains approximately 130 K transistors and occupies 9.93 mm<sup>2</sup> areas.

The architecture is based on a computation sharing multiplier (CSHM) which specifically add and shift operation. Efficient circuit-level techniques, namely a new carry-select adder using fast all one finding logic used to improve and 16 percent shorter delay than the original dual ripple-carry carry-select adder. And conditional capture flip-flop (CCFF), are also used to further improve power and performance.

**Index Terms**—Dual transition skewed logic, programmable finite impulse response (FIR) filter.

### I. INTRODUCTION

One of the most widely used operations in DSP is finite-impulse response (FIR) filtering. The FIR filter performs the weighted summations of input sequences and is widely used in video convolution functions, signal preconditioning, and various communication applications. Recently, due to the high-performance requirement and increasing complexity of DSP and multimedia communication applications, FIR filters with large filter taps are required to operate with high sampling rate, which makes the filtering operation very computationally intensive.

In the proposed FIR filter architecture, the Computation sharing multiplier (CSHM) [1] is efficiently used for the low-complexity design of the FIR filter.

Canonical-signed-digit [2] and signed-power-of-two [3] coefficient representations are widely used in the parallel implementation of FIR filters. Using those Techniques, the FIR filtering operation can be simplified to add and shift operations. Common sub expressions elimination [4], [5] and differential coefficients method [6], [7] also explore low-complexity design of FIR filters by minimizing the number of additions in filtering operations.

The main idea of CSHM is to represent the multiplications in the FIR

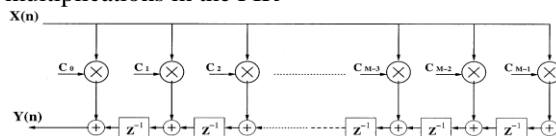


Fig. 1. Transposed direct form (TDF) FIR filter.

Adders are critical components of the ALU's (Arithmetic Logic Unit) or DSP (Digital Signal Processing) chips [10]. Among various adders, the carry-select adder (CSA) is intermediate regarding speed and area and widely used in mobile applications [11]. CSAs with very large sizes can be constructed hierarchically by combining smaller 'block' adders [12]. Fig. 5 shows the conventional CSA which consists of two ripple carry adders (RCAs) in each block (except block1), one for  $C_{in}$  (from lower block) = 0 and the other for  $C_{in}$  (from lower block) = 1. Instead of using dual RCAs, Chang proposed a 29.2% reduced area CSA with 5.9% speed penalty by replacing one RCA to an add-one circuit [13].

Flip-flops are also crucial elements from both a delay and power standpoint. Conditional capture flip-flop (CCFF) [14] is explained and used in our filter design. CCFF is a dynamic style flip-flop that has a negative setup time and small clock-to-output delay. Moreover, depending on data switching activity, CCFF also reduces the power consumption.

Filtering operations as a combination of add and shift operations over the common computation results. The common computations are identified and those are shared without additional memory area. This sharing property enables the computation sharing multiplier approach that achieves high performance and low power in FIR filter implementation.

## II. FIR FILTER ARCHITECTURE

### A. CSHM Algorithm and Architecture

The input–output relationship of the linear time invariant (LTI) FIR filter can be described as where represents the length of the FIR filter, the's are the filter coefficients and denotes the data sample at time constant. Fig. 1 shows a transposed direct form (TDF) implementation of the FIR filter. We notice that the TDF implements a product of the coefficient vector with the scalar by all the coefficients simultaneously. In the sequel, such products will be referred to as a *vector scaling* operation [1]. In the vector scaling operations, we can carefully select a set of small bit sequences so that the same multiplication result can be obtained by only add and shift Operations.

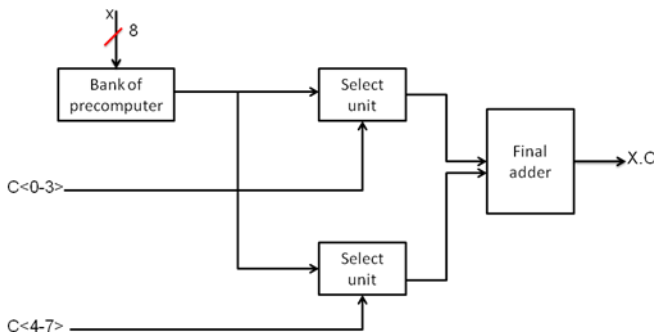


Fig. 2. Computation sharing multiplier (CSHM) architecture

For instance, a simple vector scaling operation can be decomposed as. If, and are available, the entire multiplication process is significantly simplified to a few add and shift operations. We refer to these chosen basic bit sequences as *alphabets*. Also, an *alphabet set* is a set of alphabets that spans all the coefficients in vector. In the above example, the alphabet set is. In this example, is computed once and the result is shared to calculate both and, this shows the concept of computation sharing in the vector Scaling operation.

CSHM architecture is based on the algorithm explained above. Fig. 2 shows the CSHM architecture. CSHM is composed of a precomputer, select units and final adders (S&As). The precomputer performs the multiplication of alphabets with input. Since alphabets are small bit sequences, the multiplication with input and alphabets can be done without seriously compromising the performance. Once the multiplications of alphabets with input are calculated by the precomputer, the outputs are shared by the entire S&As, which is the main advantage of CSHM. In order to cover every possible coefficient and perform general multiplication operation, we used eight alphabets in the precomputer [1]. S&As perform appropriate select/shift and add operations required to obtain the

multiplication output. The select unit is composed of SHIFTER, MUX (8:1), ISHIFTER, and AND gate. To find the correct alphabet, SHIFTERS perform the right shift operation until they encounter 1 and send an appropriate *select* signal to MUXes (8:1). SHIFTERS also send the exact shifted values (*shift* signal) to ISHIFTERS. The MUXes (8:1) select the correct answer among the eight precomputer outputs, ISHIFTERS simply inverse the operation performed by SHIFTERS (barrel shifter). [15] When the coefficient input is 0000, we cannot obtain a zero output with shifted value of the precomputer outputs. Simple AND gates are used to deal with the zero (0000) coefficient input. The final adder adds the outputs of the select units to obtain the final multiplication output. An example of the multiplication procedure is shown in Fig. 2.

The 17X 17 CSHM, shown in Fig. 3, is implemented using 0.25- m TSMC standard cell library. As shown in Fig. 2, the CSHM is composed of a precomputer and S&As. In our CSHM implementation, the input is represented in two's complement format and coefficient is in sign and magnitude format. The output of the CSHM is also in two's complement format. *Precomputer*: The multiplications performed by the

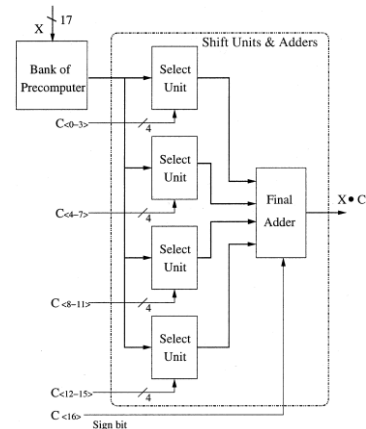


Fig. 3. 17X17 CSHM structure.

Precomputer are simply implemented using the new carry-Select adder, which is proposed in Section III-A. Fig. 4 shows the basic structures of and the precomputer structure. *Select Unit*: As shown in Fig. 2, the select unit is composed of SHIFTER, MUX, ISHIFTER, and AND gates. Since SHIFTER is directly connected to the coefficients, it does not lie on the critical path. Static CMOS design with minimum size is used for SHIFTER implementation. ISHIFTER lies on the critical path and the maximal shift width is 3 bits. A barrel shifter [2] is used since the signal has to pass through at most one transmission gate in the barrel shifter. The MUX using pass-transistor logic was implemented to achieve a compact and high-

speed design. *Final Adder*: The final adder is the largest component in the S&A, which sums the outputs of four select units. The carry-save array [2] and the new carry-select adder presented in Section III-A are used for high performance. As mentioned before, the input data is in two's complement format, the coefficient in sign and magnitude, and the final adder output in two's complement. In our CSHM design shown in Fig. 3, the sign bit of coefficient is not used and is considered as a positive number in the select unit. The XOR gate array is efficiently used for controlling the sign of the final adder output. When the coefficient is a positive number (when the sign bit is '0'), since the output of the final adder has the same sign as input data, the inputs of final adder can be added without sign conversion. When the coefficient is a negative number (when the sign bit is '1'), since the output of the final adder has a different sign than the input data, the inputs of the final adder should be converted to numbers with the opposite sign. The architecture is easily realized using the XOR gate array. The addition of the coefficient sign bit and input least significant bit (LSB) can be merged into the carry-select adder.

**B. FIR Filter Based on CSHM**

Using the 17 CSHM presented in the previous section, a 10-tap FIR filter with programmable coefficients has been implemented for fabrication. FIR filter can be implemented in *direct form* (DF) [1] or *transposed direct form* (TDF) architecture (Fig. 1). In the DF FIR filter, a large adder in the final stage lies on the critical path and it slows down the FIR filter. For high-performance filter structure, TDF is used in our implementation. In the TDF of the FIR filter shown in Fig. 1, multipliers are replaced by S&As and a precomputer is connected to the input. Therefore, as shown in Fig. 4, the FIR filter using CSHM consists of one precomputer and ten S&As. We can easily see from the figure that the precomputer outputs are shared by all the S&As. In other words, the computations are performed only once for all's and these values are shared by the entire S&As for generating. The CSHM scheme efficiently removes the redundant computations in the FIR filtering operation, which leads to low-power and high-performance design.

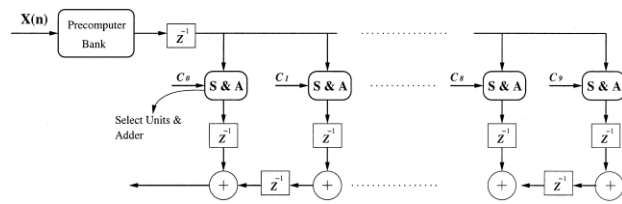


Fig. 4 Architecture of FIR filter based on CSHM.

**III. CIRCUIT LEVEL TECHNIQUES**

**A. HIGH-PERFORMANCE CARRY-SELECT ADDER USING FAST ALL ONE FINDING LOGIC**

The blocks in the conventional carry-select adder consists of two ripple carry adders, one for  $C_{in}=0$  and the other for  $C_{in}=1$ . If the results for  $C_{in}=0$  is known as  $S^0$  the result for  $C_{in}=1$  ( $S^1$ ) can be obtained by adding one to  $S^0$ . Thus, an add-one circuit can replace the ripple-carry adder for  $C_{in}=1$  to reduce the area in a block. To design an efficient add-one circuit,

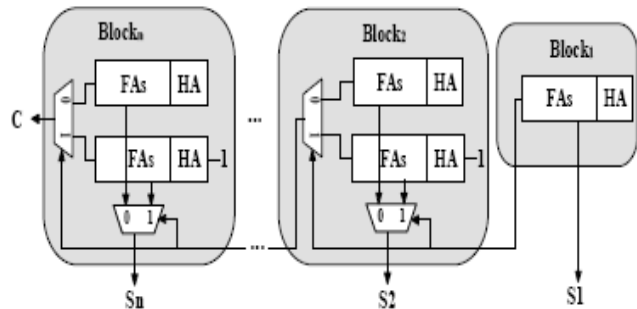


Fig.5. Conventional CSA using dual RCAs with n blocks

The first zero finding circuit is used in implement scheme [13]. Adding one to the result for  $C_{in}=0$  ( $S^0$ ), if the  $S^0$  is the first zero count from the least significant bit, the  $S^1$  is just inverting each bit of  $S^0$  starting from the least significant bit until the  $S^0$  bit (included), and other bit(s) remain the same. The 4-bit add-one circuit architecture used by Chang showed [10].

The full adder(FA) cell consists of a two-level NAND gate for carry output and two-level two-input exclusive-or gates with the critical delays. And the delay in the unit of the two input NAND gate was illustrated in [9]. The carry-chain is the critical path in the CSA, so the critical path increase 1.5 units in every block compared with the original RSA structure. As shown in [9], the implementation of the multiplexer with control signal  $C_{in}$  requires three two-input NAND gates and an inverter for the complementary signal of  $K_{in}$  [13]. Thus, the delay from  $C_{in}$  to  $C_{out}$  in the unit of the two input NAND gate is 2.5.

The carry-chain between blocks is the critical path in the CSA; it includes the delay of multiplexers and the wire delay. When another structure of multiplexer showed in was used in proposed CAS,[9] the delay time through the multiplexer becomes very small. Furthermore, in every block, compared with the original CSA structure, the circuit used to determine the  $C_{out}$  when  $C_{in}=1$  increased the delay of the critical path. Instead, a new add-one

circuit is proposed to eliminate this delay and further reduce the area as shown in Fig. 6

According to the previous complement scheme, The sum result  $S^k$  is either the  $S_k^0$  or the  $S_k^1$  ( $S_k^0$  or inverted  $S_k^0$ ), thus the  $S^k$  can be chosen from  $S_k^0$  and inverted  $S_k^0$  through a multiplexer [9]. The select signals were shown in following equations, and the P signals was generated from the first zero finding circuit to decide whether the  $S_k^1$  is  $S_k^0$  or inverted  $S_k^0$ .

$$sel_1 = c_{in} \cdot \overline{P_1}$$

$$sel_2 = c_{in} \cdot \overline{P_2}$$

$$sel_3 = c_{in} \cdot \overline{P_3}$$

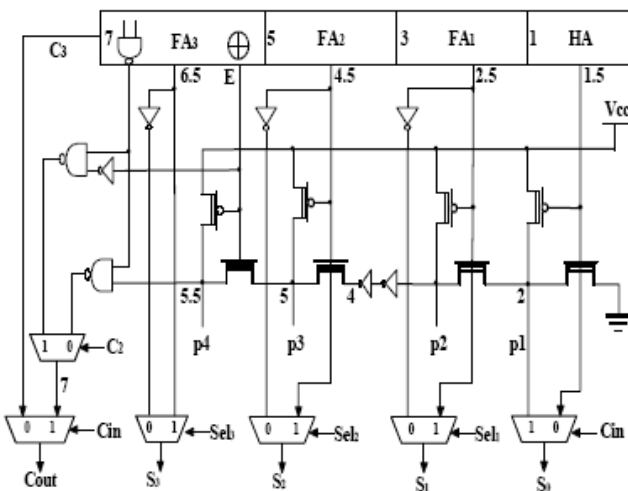


Fig. 6. Proposed CSA with fast all-one finding circuit

The carry out of a block can be chosen between the carry out of the RCA and the carry out of the add one circuit. So the fast all-one finding scheme depends on two points in a block, one is that the two carry out will be different if and only if all sums from the RCA and the Cin are equal to one, another is that if all sum bits from the RCA in a block are equal to one, the inputs of every FA or HA in the block will be different. So we can use an exclusive-or in the highest bit to determine whether all sums from the RCA are equal to one. In Fig.6, if P4 is 0, the four-bit sum from the RCA and C2 (the carry of FA2) should be '1111' and 0 respectively. The delay of P4 is reduced without extra transistors because the value of exclusive-or is contained in the FA3. Furthermore, if  $P_4 = 0$  or  $a_3 \cdot b_3 = 1$ , Cout should be 1. So, the output of the carry

$$C_{out} = C_3 + C_{in} \cdot (\overline{P_3} + a_3 \cdot b_3 + (a_3 \oplus b_3) \cdot C_2$$

#### IV. RESULTS AND COMPARISON

The designed 10-tap FIR filter with programmable coefficients was fabricated in the TSMC 0.25- m CMOS technology. The die has an area of 9.91 mm and includes more than 130 000 transistors. For the test and measurement of the chip, input test patterns are generated and output patterns and waveforms are monitored using a logic analyzer. Separate power supplies for the core and PADs allow exact measurement of the power consumption of the core of the chip.

Table I the implementation of the 4 bit dual ripple-carry adder and proposed carry-select adder with number of transistor and speed is listed in Table 1. The first row shows the conventional adder, the second row shows the modified adder and third row proposed carry select adder. Proposed CSA is minimum number of transistor and also some percent speed improved.

Comparisons of result of CSA

| Logic styles                    | No. of transistor | Speed (ns) |
|---------------------------------|-------------------|------------|
| Conventional carry select adder | 280               | 116        |
| Modified carry select adder     | 245               | 50         |
| Proposed carry select adder     | 182               | 116        |

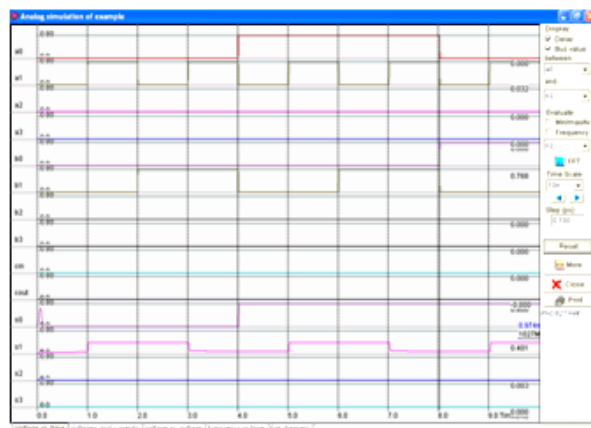


Fig.7.Result of Conventional CSA

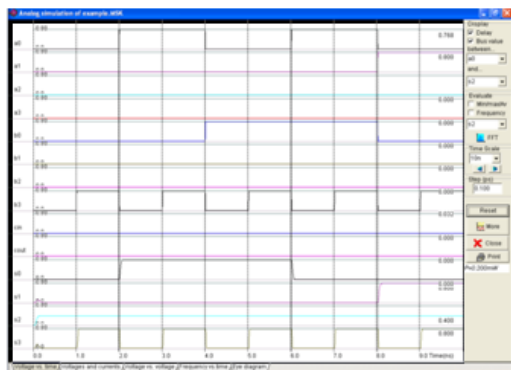


Fig.8.Result of Proposed CSA

## V. CONCLUSION

An FIR filter based on CSHM was implemented using 0.25- $\mu$ m technology. The CSHM algorithm specifically targets reduction of redundant computation in FIR filtering operation. Using the CSHM scheme, the multiplications in vector scaling operation were significantly simplified to add and shift operations of alphabets multiplied by input. These common computations were shared by the sequence of operations in vector scaling operations. Adders and flip-flops are critical components in CSHM and FIR filter implementation. Circuit-level techniques for the high performance carry select adder and flip-flop were proposed and used in the full-custom FIR filter implementation. The CSHM scheme and circuit-level Techniques helped to achieve high-performance FIR filtering operation. The proposed CSHM architecture is also applicable to adaptive filter and matrix multiplication implementation. The ideas presented in this paper will help the design of DSP algorithms and their implementation for high-performance applications.

## REFERENCES

- [1.] H. Samueli, "An improved search algorithm for the design of multiplierless FIR filter with powers-of-two coefficients," *IEEE Trans. Circuits Syst.*, vol. 36, pp. 1044–1047, July 1989.
- [2.] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans. Circuits Syst. II*, vol. 43, pp. 677–688, Oct. 2008
- [3.] M. Potkonjak, M. Srivastava, and A. P. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination," *IEEE*

*Trans. Computer-Aided Design*, vol. 15, pp. 151–165, Feb. 1996.

- [4.] N. Sankarayya, K. Roy, and D. Bhattacharya, "Algorithms for lowpowerhigh speed FIR filter realization using differential coefficients," *IEEE Trans. Circuits Syst. II*, vol. 44, pp. 488–497, June 1997
- [5.] K. Muhammad and K. Roy, "A graph theoretic approach for synthesizing very low-complexity high-speed digital filters," *IEEE Trans. Computer-Aided Design*, vol. 21, pp. 204–216, Feb. 2002.
- [6.] W. Jeong, K. Roy, and C. Koh, "High-performance low-power carry-select adder using dual transition skewed logic," in *Proc. ESSCIRC*, 2001, pp. 172–175.
- [7.] Yan Sun., xin zhang. And xi jin. 'High performance carry select adder using all one finding logic', Second Asia International Conference on Modelling and Simulation. June 2008.
- [8.] J. Woopyo, and R. Kaushik. "RobustHigh-Performance Low-Power Carry Select Adder", *Design Automation Conference*, pp. 503 - 506, Jan. 2010
- [9.] K. Rawat, T. Darwish, and M. Bayoumi, "A low power and reduced area carry select adder", The 45th Midwest Symposium on *Circuits and Systems.*, vol.1, pp. 467-470, 2002.
- [10.] N. Weste and K. Eshragian, "Principle of CMOS VLSI designs: a system perspective", 2nd ed., *Addison-Wesley*, 2010.
- [11.] T. Y. Chang and M. J. Hsiao, "Carry-select adder using ripple-carry adder", *Electron. Lett.* vol. 34, pp. 2101-2103, 2000.
- [12.] B. S. Kong *et al.*, "Conditional capture flip-flop for statistical power reduction," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2000, pp. 290–291.
- [13.] J. M. Rabaey, *Digital Integrated Circuits: A Design Perspective*. Englewood Cliffs, NJ: Prentice-Hall, 1996
- [14.] Jongsun Park, Woopyo Jeong, Hamid Mahmoodi-Meimand, *Student Member, IEEE*, Yongtao Wang, Hunsoo Choo, and Kaushik Roy, *Fellow, IEEE* "Computation Sharing Programmable FIR Filter for Low-Power and High-Performance Applications".
- [15.] Y. C. Lim and S. R. Parker, "FIR filter design over a discrete powers-of-two coefficient space," *IEEE Trans. Acoust., Speech Signal Processing*, vol. ASSP-31, pp. 583–591, June 1983.

#### **AUTHOR PROFILES**

R.Venkatesh received his M.Tech (Communication Engineering) degree from VIT University, TamilNadu, India, in 2010. He is serving as Lecture in the department of Electronics and Communication, CMR Engineering College, Hyderabad, India, since 2014. His research interests include Digital signal processing, Mobile Wireless and Multimedia Technology, FPGA, Digital systems and Speech and video Processing.

G.Santhosh received his M.Tech (Communication Engineering) degree from VIT University, TamilNadu, India, in 2010. He is serving as Lecture in the department of Electronics and Communication, AVANTHI Engineering College, Hyderabad, India, since 2014. He has served as Lecturer in the department of Electronics and Communication, Guru Nanak Institute of Engineering & Technology, Hyderabad, India. His research interests include Digital signal processing, Digital systems, Mobile Wireless and Image Processing and Artificial Intelligence.