ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Area Efficient Pipelined Design for Digit-Serial Fir Filters

Divya.A[*1], Santhakumar.K[2]

[*1] PG Student, Department. of ECE, Nandha Engineering College, Erode, India

[2] Assistant Professor, Department of ECE, Nandha Engineering college, Erode, India

divya27691@gmail.com

Abstract

FIR filters are being designed using HDL languages to enhance the speed of the system. In the whole system if the speed of the individual block is enhanced, the overall speed of the system is enhanced. In order to obtain effective utilization hardware is done by applying the pipelining technique. Pipelining is an implementation technique in which multiple instructions are overlapped in execution. The proposed design of this paper is an attempt to optimize the system speed with minimal cost and hardware. In a filter the pipelining of multiplication is achieved by shifts and addition method. Pipelined technique may reduce area, delay and enhance speed as compared to common sub-expression elimination algorithm.

Keywords: FIR filter, Pipelining, common sub-expression elimination.

Introduction

Finite impulse response (FIR) filters are of great importance in digital signal processing (DSP) systems since their characteristics in linear-phase and feed-forward implementations make them very useful for building stable high-performance filters. The problem of designing FIR filters has received a significant amount of attention during the last decade, because the filters require a large number of multiplications, leading to excessive area, delay and power consumption, even if implemented in a full custom integrated circuit. The direct and transposed-form FIR filter implementations, both architectures have similar complexity in hardware; the transposed form is generally preferred because of its higher performance and power efficiency.

The multiplier block of the digital FIR filter in its transposed form as shown in Fig. 1(a), where the multiplication of filter coefficients with the filter input is realized, has significant impact on the complexity and performance of the design because a large number of constant multiplications are required. This is generally known as the multiple constant multiplications (MCM) operation shown in Fig. 1(b) and is also a central operation and performance bottleneck in many applications such as, digital audio and video processing, wireless communication, and computer arithmetic. This method replaces all traditional multipliers by an MCM block following the transposed direct form structure of FIR filter. The MCM block is internally built with cost-efficient shifters and adders. An algorithm for optimization of a transpose form FIR filter with constant coefficients by an exhaustive search is presented. Exhaustive search method is to find multiple common bit patterns in a FIR filter coefficient array. By removing multiple occurrences of the same pattern, and calculating these only once, hardware is saved. For larger filters, this leads to important area reductions. During the optimization process, several techniques are used to obtain a cheaper implementation. First, all multiplications in a FIR filter are splitted into sequences of add-shift operations. A Canonical signed Digit (CSD) representation is used. It is a representation with a minimum number of non-zero bits, and therefore a minimum number of adder-subtractors are necessary for the expansion. In the next step, the set of coefficients to implement is subject to Common Sub-expression Elimination (CSE). This involves finding multiple bit-patterns in the set of coefficients and coding these repetitions as common Sub-expressions into the filter structure. Hardware then is saved by the reduction of the number of adders necessary.

In this paper, pipelined technique is proposed. A pipeline is a set of data processing elements connected in series, so that the output of one element is the input of the next one. In most of the cases we create a pipeline by dividing complex operations into simpler operations. We can also say that instead of taking a bulk thing and processing it at

ISSN: 2277-9655
Impact Factor: 1.852

once, we break into smaller pieces and process it one after another to enhance the speed, and reduce area and delay when compared to the common sub-expression algorithm.

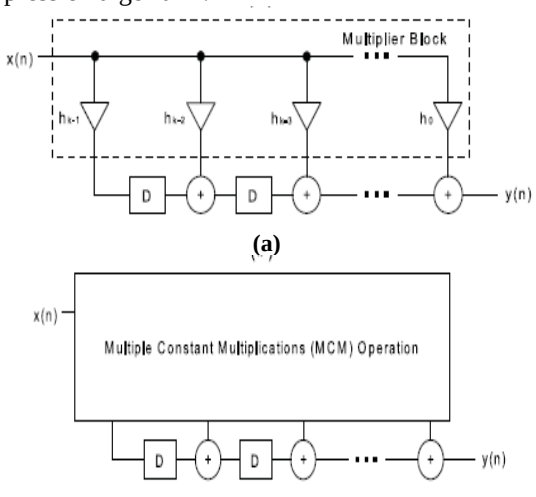

Fig. 1. FIR filter implementations. (a) Transposed form. (b) Transposed form with an MCM block.

II. DESIGN FLOW OF FIR FILTERS

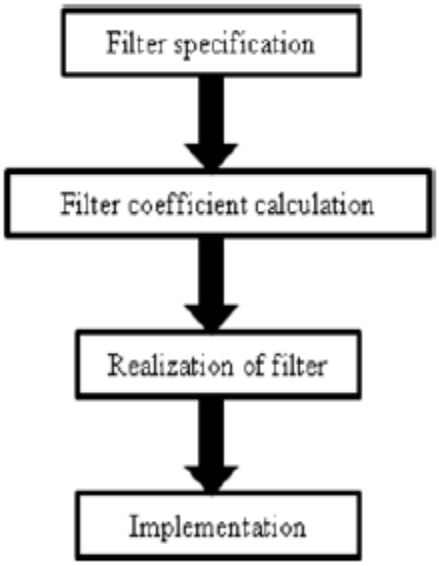

Fig. 2. Design flow of FIR filter

The design of a digital filter involves following five steps as indicated in Fig. 2.

Filter Specification: This may include stating the type of filter, for example low pass filter, the desired amplitude and/or phase responses and the tolerance, the sampling frequency, the word length of the input data.

Filter Coefficient Calculation: The coefficient of a transfer function H(z) is determined in this step, which will satisfy the given specification. The choice of calculation method will be influenced by several factors. The most important of which are the critical requirements i.e. specification.

Realization: This involves converting the transfer function into a suitable filter network or structure.

IMPLEMENTATION: This involves producing the software code and/or hardware and performing the actual filtering.

Common Sub-Expression Elimination

The CSE algorithms initially extract all possible sub-expressions from the representation of the constants when they are defined under binary, canonical signed digit (CSD), or minimal signed digit (MSD). Then, they find the best sub-expression, generally the most common, to be shared among the constant multiplications.

For the shift-adds implementation of constant multiplications, a straightforward method, generally known as digit-based recoding, initially defines the constants in binary. Then, for each "1" in the binary representation of the constant, according to its bit position, it shifts the variable to obtain the result. As a simple example, consider the constant multiplications Eq. 1 and Eq. 2. Their decompositions in binary are listed as follows:

$29x = (11101)_{bin}\, x = x << 4 + x << 3 + x << 2 + x$ (1)

$43x = (101011)_{bin}\, x = x << 5 + x << 3 + x << 1 + x$ (2)

Which requires six addition operations as illustrated in Fig. 3. as shown

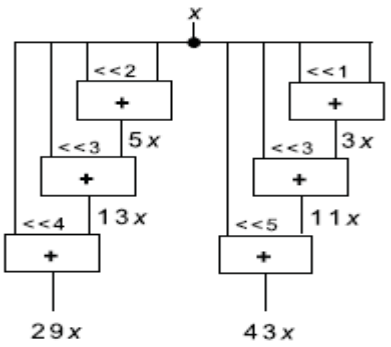

Fig. 3. Shift-adds implementation of 29x and 43x (without partial product sharing)

However, the digit-based recoding technique does not exploit the sharing of common partial products, which allows great reductions in the number of operations and, consequently, in area and power dissipation of the MCM design at the gate level. Hence, the fundamental optimization problem, called the MCM problem, is defined as finding the minimum number of addition and subtraction operations that implement the constant multiplications,. Note that, in bit-parallel design of constant multiplications, shifts can be realized using

ISSN: 2277-9655
Impact Factor: 1.852

only wires in hardwire without representing any area cost.

Considering the shift-add implementation of 29x and 43x, the common sub-expression elimination (CSE) algorithm gives a solution with four operations by finding the most common partial products 3x = $(11)_{bin}$ x and 5x = $(101)_{bin}$ x when constants are defined under binary, as illustrated in Fig. 4.

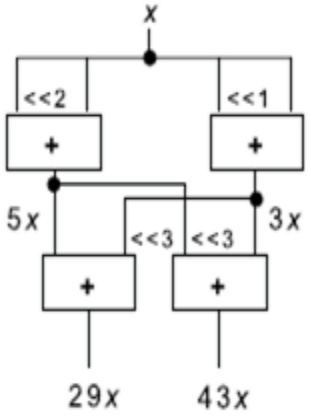

Fig. 4. CSE algorithm (with partial product sharing)

Pipelining

In pipelining, any operation along the critical path is broken into smaller, quicker operation with registers between levels, so as to get a smaller critical path delay. As a result there will be an increase in operating frequency which leads to higher throughput. Pipelining transformation leads to a reduction in the critical path, which can be exploited to either increase the clock speed or sample speed or to reduce power consumption at same speed. Moreover some real time application requires faster input rates where direct structures cannot be used. In such cases we can use pipelining by introducing latches along the critical data path.

The critical path or the minimum time required for processing a new sample is limited by 1 multiply time and 2 add times, i.e., if T_M is the time taken for multiplication and T_A is the time needed for addition operation then the "sample period" Eq. 3 is given by,

$T_{sample} \geq T_M + 2T_A$ (3)

Therefore, the sampling frequency (also referred to as the throughput or the iteration rate) Eq. 4 is given by

$F_{sample} = 1/(T_M + T_A)$ (4)

The following points give the significances of pipelining;

1. The speed of architecture (or the clock period) is limited by the longest path between any 2 latches or between an input and a latch or between a latch and an output or between the input and the output.
2. This longest path or the "critical path" can be reduced by suitably placing the pipelining latches in the architecture.

Proposed Method

In proposed method, 4 tap pipelined FIR filter structure is presented as shown in Fig. 5. The transposed direct form with two additional registers (one for multiplication and one for adder registers) is shown in figure.

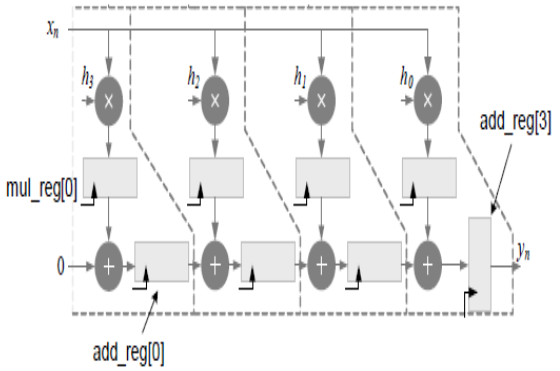

Fig. 5. Pipelined FIR filter structure

All the multiplications of a variable at the time and the result is stored in a multiplication registers such as mul_reg[0], mul_reg[1], mul_reg[2], mul_reg[3]. The stored results are added and then it is delayed by the pipelined registers. Therefore any operation along the critical path is broken into smaller, quicker operation with register between levels, so as to get a smaller critical path and delay. Moreover real time applications require faster input rates where direct structures cannot be used. In such cases we can use pipelined transposed direct form by introducing latches along the critical data path.

Simulation View of Pipelined Fir Filter

The four tap pipelined FIR filter structure is simulated in Xilinx 9.1i ISE simulator and the gate level results are obtained.

Schematic Diagram Of Pipelined Fir Filter

The schematic diagram for four tap pipelined FIR filter structure is shown in Fig. 6.

ISSN: 2277-9655
Impact Factor: 1.852

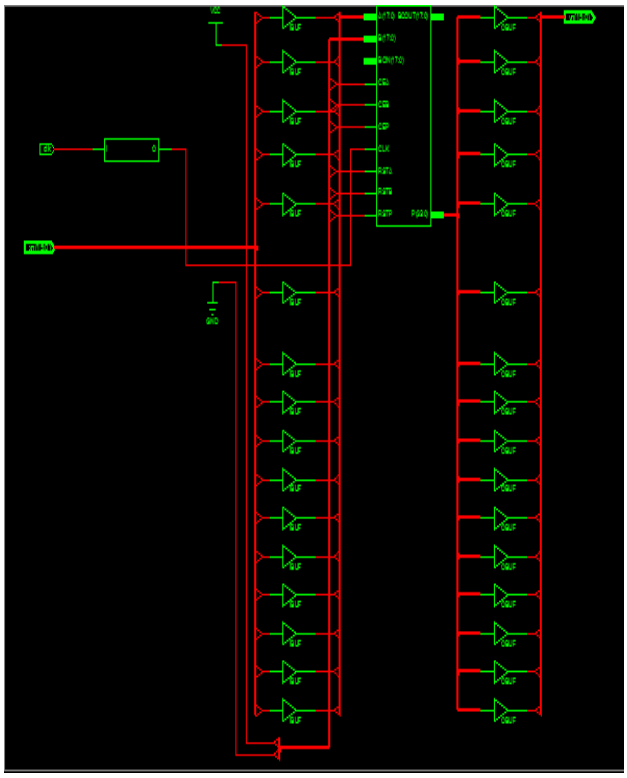

Fig. 6. Schematic diagram of Pipelined FIR filter

Output Waveform Of Pipelined Fir Filter

The output waveform of four tap pipelined FIR filter is shown in Fig. 7. In this the input signal x_n is multiplied by the coefficients h0, h1, h2, h3 at the same time and it is stored in mul_reg. The pipelining process is taken and the output is stored in y_n.

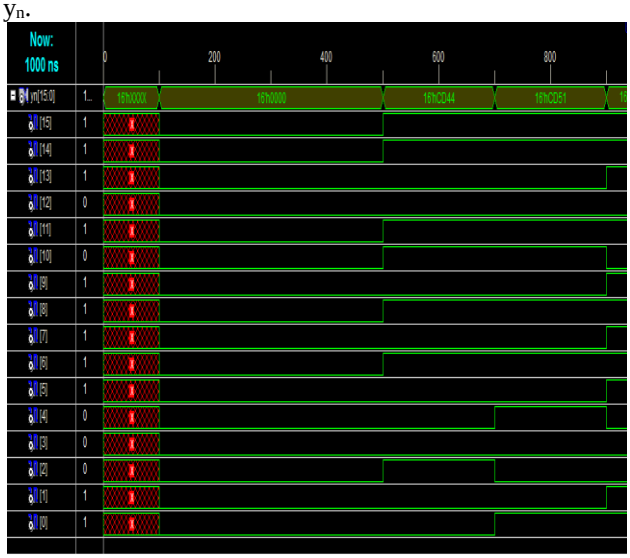

Fig. 7. Output waveform of Pipelined FIR filter

Power And Area Comparison Table

The Table I shows that the pipelined FIR filter under shift-adds architecture yields significant area reduction than common sub-expression algorithm. Power consumption is also less in pipelined FIR filter when compared to common sub-expression algorithm.

Table I. Power and area comparison table

Algorithm	Architecture	Parameter	result obtained
Pipelined architecture	**Shift-adds**	**Area** **Power(mW)**	**3** **81**
Common Sub-expression Elimination	**Shift-adds**	**Area** **Power(mW)**	**56** **84**

Power And Area Comparison Chart

The power and area for binary two's complement, common sub-expression elimination algorithm and pipelined architecture is analyzed with the help of graph.

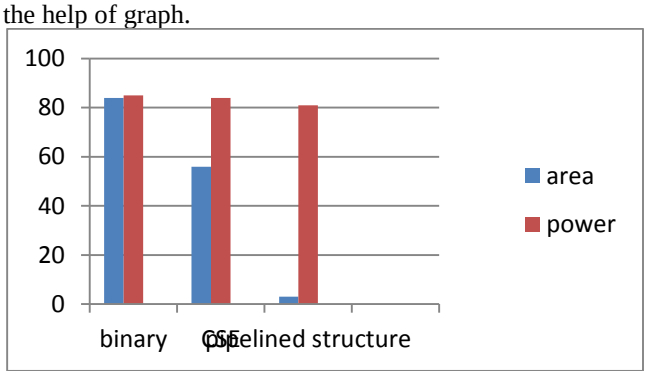

Fig. 8. Power and area comparison chart

Fig. 8. Show that the power and area comparison chart. The area of CSE-CSD is reduced when compared to the binary. And also, the power consumption is less.

Conclusion

The cost effective design of four tap FIR filter has been presented using pipelining operations with shift and add architecture. The simulated model has been synthesized using Xilinx synthesis tool. The results shows that the pipelined FIR filter structure require less area and power consumption when compared to common sub-expression elimination algorithm. Hence the system speed and throughput is improved.

References

[1] *A. Avizienis, "Signed-digit number representations for fast parallel arithmetic," IRE Trans. Electron. Comput., vol. 10, no. 3, pp. 389-400, Sep. 1961.*

[2] *A. Dempster and M. Macleod, "Constant integer multiplication using minimum*

adders," IEEE Proc.-Circuits, Devices, syst., vol. 141, no. 5, pp. 407-413, Oct. 1994.

[3] A. Dempster and M. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," IEEE Trans. Circuits syst. II, Exp. Briefs, vol. 42, no. 9, pp. 569-577, Sep. 1995.

[4] C. Wallace, "A suggestion for a fast multiplier," IEEE Trans. Electron. Comput., vol. 13, no. 1, pp. 14-17, Feb. 1964.

[5] F. Dittmann, B. Kleinjohann, and A. Rettberg, "Efficient bit-serial constant multiplication for FPGAs," in Proc. NASA Symp. VLSI Design, 2003, pp. 1-6.

[6] H. Nguyen and A. Chatterjee, "Number-splitting with shift-and-add decomposition for power and hardware optimization in linear DSP synthesis," IEEE Trans. Very Large Scale Integration. (VLSI) Syst., vol. 8, no. 4, pp. 419-424, Aug. 2000.

[7] H. Suzuki, Y. –N.Chang, and K. Parhi, "Performance tradeoffs in digit-serial DSP systems," in Proc. Asilomar Conf. signals, syst. Comput., 1998, pp. 1225-1229.

[8] I. –C. Park and H. –J. Kang, "Digital filter synthesis based on minimal signed digit representation," in Proc. DAC, 2001, pp. 468-473.

[9] K. Johansson, O. Gustafsson, A. Dempster, and L. Wanhammar, "Algorithm to reduce the number of shifts and additions in multiplier blocks using serial arithmetic," in Proc. IEEE Medit. Electrotecch. Conf., May 2004, pp. 197-200.

[10] K. Johansson, O. Gustafsson, and L. Wanhammar, "Multiple constant multiplication for digit-serial implementation of low power fIR filters,' WSEAS Trans. Circuits Syst., vol. 5, no. 7, pp. 1001-1008, 2006.

[11] K. Parhi, "A systematic approach for design of digit-serial signal processing architectures," IEEE Trans. Circuits Sys., vol. 38, no. 4, pp. 358-375, Apr. 1991.

[12] L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Design of digit –serial FIR filters: Algorithms, architectures, and a CAD tool," IEEE Trans. Very large scale integration systems., vol. 21, no. 3, pp. 498-511, Mar. 2013.

[13] L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 6, pp. 1013-1026, Jun. 2008.

[14] L. Aksoy, E. Costa, P. Flores, and J. Monteiro., "Optimization of area in digital FIR filters using gate-level metrics," in Proc. DAC, 2007, pp. 420-423.

[15] M. Potkonjak, M. Srivastava., and A. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination," IEEE Trans. Comput-Aided Design Integr. Circuits syst., vol. 15, no. 2, pp. 151-165, Feb. 1996.

[16] P. Cappello and K. Steiglitz, "Some complexity issues in digital signal processing," IEEE Trans. Acoust., Speech, Signal Process., vol. 32, no.5, pp. 1037-1041, Oct. 1984.

[17] R. Hartley and P. Corbett, "Digit-serial processing techniques," IEEE Trans. Circuits Syst., vol. 37, no.6, pp. 707-719, Jun. 1990.

[18] R. Hartley, "Sub-expression sharing in filters using canonic signed digit multipliers," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 43, no. 10, pp. 677-678, Oct 1996.

[19] Y. –H. Ho, C. –U. Lei, H. -.K. Kwan, and N. Wong, "Global optimization of common subexpressions for multiplierless synthesis of multiple constant multiplications," in Proc. ASPDAC, 2008, pp. 119-124.

[20] Y. Voronenko and M. Puschel, "Multiplierless multiple constant multiplication," ACM Trans. Algor., vol. 3, no. 2, pp. 1-39, May 2007.