Design and Implementation of Low Power and Low Area FIR Filter Using CSM Architecture

Abstract: Low power and low area are the two important requirement of low power FIR filter used in wireless mobile communication system. In this paper the new low power and low area FIR filter architecture called CSM (Constant Shift Method) architecture is proposed. The proposed architecture will operate in different filter coefficient wordlength. The proposed FIR filter can be efficiently implemented by using Binary Subexpression Elimination algorithm. The proposed architectures will be implemented and tested on Spartan 3, Field-Programmable Gate Array and synthesized using Xilinx ISE and ModelSim. The proposed architecture will give good area and power reduction compared to the existing FIR filter.

Keywords: FIR filter, low area, low power, binary subexpression elimination.

N. C. Sendhil Kumar
Research scholar, St. Peter's University, Chennai, India

Dr. E. Logashanmugam
Professor and Head, Dept of ECE, Sathya bama University, Chennai, India
1. Introduction:
FIR filters are mainly used in wireless mobile communication systems for application such as pulse shaping, channel equalization, channelization and matched filtering, due to their linear phase property and absolute stability[1]. The filter which used in mobile system should consume low power and low area.

The power consumption and area required depends on number of multipliers used in the filter architecture. Multipliers are used to multiply the filter coefficients. In the proposed method the algorithm called binary subexpression elimination [2] is used to reduce the number of multipliers used in the filter. The low complexity multipliers are produced by canonical signed digit which is based on common sub expression method [3]. The aim of (CSE) Common Subexpression Elimination is to find the occurrence of identical group of bit pattern. This method is developed to [5], to minimize the logical depth to increase the speed of the operation. The FIR filter based on Binary Common Subexpression Elimination (BCSE) method, which provide low area when compare to the FIR filter [3]-[5]. In [7] FIR filter is designer based on pseudo floating point method used to reduce the complexity of the filter and it is designed to operate in fixed coefficient wordlength. The method [7] is only suitable for fixed coefficient wordlength. In [7]-[8] describes the FIR filter which works on infinite filter coefficient wordlength. These designs consist of fully programmable multiply-accumulate based processor, used to store filter coefficient in register. The architecture of the processor consists of control unit, program memories and a data path with a single MAC unit. The data path consists of multiplier, 32-bit accumulator and a 16-bit adder/subtractor. The processor is used to reduce the delay in the data path. But the main disadvantage of this processor is it consumes more area and power. In [8], the comparison between speech based algorithm and general purpose processor was given. It shows that the power consumption of general purpose processor is four times greater than the speech based algorithm.

There are several methods to reduce the area and power usage of FIR filter. This paper proposed the new architecture called CSM to reduce power and area usage. This architecture will be tested and implemented in Spartan 3, FPGA and synthesized in ModelSim and Xilinx ISE.

2. About CSE Method:
This section consists of basic information about CSE algorithm, which deals how to eliminate
the reoccurrence of binary pair. An n-bit binary number can produce \( 2^n - (n + 1) \) BCS. For
example a 3-bit binary representation consist of four BCS, they are [0 1 1], [1 0 1], [1 1 0],
[1 1 1] (more than one non-zero bit). This BCS are represented as [0 1 1] = \( x_1 = 2^{-1}x + 2^{-2}x \),
[1 0 1] = \( x_2 = x + 2^{-2}x \), [1 1 0] = \( x_3 = x + 2^{-1}x \), [1 1 1] = \( x_4 = x + 2^{-1}x + 2^{-2}x \), where \( x \) is the
filter input. The other BCS such as ([0 0 1], [0 1 0], [1 0 0]) which does not require for adder
implementation, because they have only one non zero bit. Thus the normal implementation
required only five adders. By using BCS algorithm the equation can be rearranged and \( x_1 \) can
be obtained from \( x_3 \) by a right shift operation and it is given by
\[
x_1 = 2^{-1}x + 2^{-2}x = 2^{-1}(x + 2^{-1}x)
\]
(1)
And also \( x_4 \) can be obtained from \( x_3 \) and it is given by
\[
x_4 = x + 2^{-1}x + 2^{-2}x = x_4 + 2^{-1}x
\]
(2)
Thus the BCS algorithm requires only three adders to implement \( x_1 \) to \( x_4 \). In BCS method the
number of adders required for n-bit binary number is denoted as \( 2^{n-1} - 1 \). The number of
adders required to implement FIR filter using BCS method is less than the number of adder
used in CSD method.

3. Architecture Of Proposed Filter:
The proposed FIR filter architecture is based on transposed direct form, which is shown in
fig. 1

![Figure 1: Transposed Direct of FIR filter](image)

PE represents the processing element of the FIR filter. PE-\( n \) represents the \( n^{th} \) processing
element of the filter. The PE used here is different from other by its operation. The number of
processing element required is dependent on order of the filter or number of taps in the FIR
filter, (for e.g. if the filter is third order means it required three processing elements). PE is
used to multiply the filter coefficient with the help of add and shift unit. The architecture of proposed FIR filter is shown in fig. 2

![Architecture of proposed system](image)

*Figure 2: Architecture of proposed system*

The PE in the FIR filter is used to produce the partial products by multiplying the filter coefficients with the filter input \((h \ast x[n])\) or in parallel way when parallel PE is employed. The important use of PE is to reduce the number of multiplier for power consumption and to reduce the circuit complexity for area usage. Each block in the proposed filter architecture is explained below

i. Shift and Add unit: Add and Shift is one of the best way to reduce the multiplication complexity. The architecture of add and shift unit is shown in fig. 3. The add and shift unit used in the existing system is very complex when compared to the add and shift unit in the proposed system. The proposed system use BCS (binary common Subexpression)- based add and shift unit.

![Architecture of proposed add and shift unit](image)

*Figure 3: Architecture of proposed add and shift unit*
Suppose the 3-bit input ranging from [0 0 0] to [1 1 1], the BC method is used to eliminate the repeated binary value present in the input coefficient. The total number of BCS formed is denoted by

\[ 2^{-1} - (n + 1) \]

\( n \) is the number of bit

ii. Multiplexer unit: The output from the add and shift unit is common to all multipliers in the CSM architecture. The multiplier unit is used to select the correct output from the add and shift unit. The multiplier used in the CSM architecture is 8:1/4:1, because it uses 8/4 input from the add and shift unit. The filter coefficients which stored in the LUT is used as the selection signal for the multipliers. In CSM architecture the filter coefficients are previously stored in the LUT. The number of multiplier required is depend on the number of groups after partitioning of the filter coefficients

iii. Final Shifter unit: After all intermediate additions the final shifter unit will perform the shifting operation and the output of 16-bit input is given below

\[ y = 2^{-1}x + 2^{-6}x + 2^{-15}x + 2^{-16}x \]  

(3)

By coefficient partitioning, we obtain

\[ y = 2^{-4}(x + 2^{-2}x) + 2^{-15}(x + 2^{-1}x) \]  

(4)

After obtaining the intermediate term \((x + 2^{-2}x)\) and \((x + 2^{-1}x)\) from the add and shift unit, then the final shifter unit will perform the shift operations \(2^{-4}\) and \(2^{-15}\).

iv. Final Adder Unit: This unit is used to obtain the sum of all the intermediate addition of \(2^{-4}(x + 2^{-2}x)\) and \(2^{-15}(x + 2^{-1}x)\).

\[ h = 2^{-1}(x + 2^{-1}x + 2^{-2}x + 2^{-3}(x + 2^{-1}x + 2^{-2}x) + 2^{-6}(x + 2^{-1}x) \]  

(7)

The terms \((x + 2^{-1}x + 2^{-2}x)\) and \((x + 2^{-1}x)\) obtained from the add and shift unit. The intermediate sums that shown in (7) is obtained by three multiplexers, two 8:1 mux for first two terms and one 4:1 mux for last two filter coefficients.

In fig. 4 \(r_1 \) to \(r_6\) denotes the output of the mux1 to mux 6. The shift operation is given as follows
\[
y = 2^{-1}r_1 + 2^{-4}r_2 + 2^{-7}r_3 + 2^{-10}r_4 + 2^{-13}r_5 + 2^{-16}r_6 \quad (8)
\]

**Figure 4: Architecture of CSM**

The number of multiplier unit required is given by \( [n/3] \), where \( n \) is the bit length of the filter coefficient. Consider the 8-bit filter coefficient \( h = "0.11111111" \). The length of the filter coefficient is 8. In this case it requires 8 multipliers. The output \( y = h^*x \) is expressed as

\[
y = 2^{-1}x + 2^{-2}x + 2^{-3}x + 2^{-4}x + 2^{-5}x + 2^{-6}x + 2^{-7}x + 2^{-8}x \quad (5)
\]

By partitioning (5),

\[
h = 2^{-1}(x + 2^{-2}x + 2^{-3}x + 2^{-4}x + 2^{-5}x + 2^{-6}x + 2^{-7}x) \quad (6)
\]

The shift obtained by partitioning the 16-bit coefficient into 3-bit group is given by

\[
y = 2^{-1}[(r_1 + 2^{-3}r_2) + 2^{-6}[(r_3 + 2^{-3}r_4) + 2^{-6}(r_5 + 2^{-3}r_6)]] \quad (9)
\]

Substituting \( (r_1 + 2^{-3}r_2), (r_3 + 2^{-3}r_4) \) and \( (r_5 + 2^{-3}r_6) \) as \( r_7, r_8, r_9 \), respectively

\[
y = 2^{-1}[r_7 + 2^{-6}(r_8 + 2^{-6}r_9)] \quad (10)
\]

Substituting \( (r_8 + 2^{-6}r_9) \) as \( r_{10} \)

\[
y = 2^{-1}(r_7 + 2^{-6}r_{10}) \quad (11)
\]

Substituting \( (r_7 + 2^{-6}r_{10}) \) as \( r_{11} \)

\[
y = 2^{-1}(r_{11}) \quad (12)
\]

The equations from (8) to (12) represents the correspondence output of fig. 4. The main advantage of CSM architecture is that all the shift operation is constant irrespective to the
filter coefficient. The method used in the CSM architecture is to split the filter coefficients into three bits, these groups are used as the selector signal of multiplexer.

5. Synthesis Results:

The synthesis results of the proposed CSM architecture is shown in this section. Xilinx ISE simulator is used for simulation. Table 1 shows the simulation output of area usage and table 2 show the simulation result of power usage of proposed CSM architecture.

<table>
<thead>
<tr>
<th>Logic utilization</th>
<th>Used</th>
<th>Available</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of 4 input LUTs</td>
<td>Proposed 48</td>
<td>Existing 61</td>
<td>Proposed 7,168</td>
</tr>
<tr>
<td>Logic Distribution</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of occupied slices</td>
<td>Proposed 26</td>
<td>Existing 106</td>
<td>Proposed 3,584</td>
</tr>
<tr>
<td>Number of slices containing only related logic</td>
<td>Proposed 26</td>
<td>Existing 106</td>
<td>Proposed 26</td>
</tr>
<tr>
<td>Number of slices containing unrelated logic</td>
<td>Proposed 0</td>
<td>Existing 0</td>
<td>Proposed 26</td>
</tr>
</tbody>
</table>

Table 1: Area usage of CSM architecture for four taps using 16 bit filter coefficients

<table>
<thead>
<tr>
<th>Name</th>
<th>Power (w)</th>
<th>Used</th>
<th>Total available</th>
<th>Utilization (1%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Proposed</td>
<td>Existing</td>
<td>Proposed</td>
<td>Total available</td>
</tr>
<tr>
<td>Clocks</td>
<td>0.071</td>
<td>0.101</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Logic</td>
<td>0.002</td>
<td>0.001</td>
<td>91</td>
<td>63</td>
</tr>
<tr>
<td>Signals</td>
<td>0.019</td>
<td>0.012</td>
<td>108</td>
<td>229</td>
</tr>
<tr>
<td>Ios</td>
<td>0.049</td>
<td>0.048</td>
<td>18</td>
<td>43</td>
</tr>
<tr>
<td>Total Quiescent Power</td>
<td>0.059</td>
<td>0.059</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Total Dynamic Power</td>
<td>0.260</td>
<td>0.283</td>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>Total Power</td>
<td>0.319</td>
<td>0.342</td>
<td>....</td>
<td>....</td>
</tr>
</tbody>
</table>

Table 2: Power usage of CSM architecture for four taps using 16 bit filter coefficients

The shift and add units are employed that can generate 3 bit BCS which uses only three adders. The number of adder requires to implement n bit BCS is $2^{n-1} - 1$. Therefore the shift and
add units are capable of generating four bit, five bit, and six bit BCS that requires 7, 15 and 13 adders. The Logic Depth (LD) of five bit and six bit BCS based shift and add units are same thus the 3bits BCS based shift and add units requires fewer number of adders than the four bit BCS based shift and add units for the same LD. In the proposed CSM architecture 8:1 multiplexers are require. If four bit BCS was used instead of 3bit BCS the complexity of shift and add units and multiplexer unit would have increased, where as the complexity of final adder unit decrease.

The above table shows the area and power usage of the proposed and existing system for four tap FIR filter. The number of slices used in proposed system is 48 and the number of slices used in existing system is s 61. And the power usage of existing system is 0.342(w) and power usage of proposed system is 0.319(w).

6. Conclusion:

The main objective of the article is to reduce the power and area usage of the FIR filter. Here CSM approach is used for implementing higher order filters with low complexity and low power, the CSM architecture also operates in high speed also due to using pipeline techniques in final stage. It was simulated using Xilinx ISE simulator and it will synthesized by using CMOS technology with high precision of 16- bits. The proposed method is a new approach for low complexity and low power FIR filters.
References:


