CHAPTER V

DESIGN OF A NEW RECONFIGURABLE ARCHITECTURE FOR IMPLEMENTING HIGH SPEED FIR FILTER

5.1 Introduction

One of the most widely used operations performed in DSP is finite impulse Response (FIR) filtering. In many applications, in order to achieve high spectral containment and/or noise attenuation, the FIR filters with fairly large numbers of taps are necessary. Many previous efforts for reducing power consumption of FIR filter generally focus on the optimization of the filter coefficients while maintaining a fixed filter order. In such approaches, FIR filter structures are simplified to add and shift operations, and minimizing the number of additions/subtractions is one of the main goals of the research. However, one of the drawbacks encountered in this approach is that once the filter architecture is decided, the coefficients cannot be changed; therefore, this technique is not applicable to the FIR filter with programmable coefficients.

Approximate signal processing techniques are also used for the design of low power digital filters (Dempster.A.G 2002). FIR filter order dynamically varies according to the stop band energy of the input signal. However, the approach suffers from slow filter-order adaptation time due to energy computations in the feedback mechanism. Previous studies show that sorting both the data samples and filter coefficients before the convolution operation has a desirable energy-quality characteristic of FIR filter. However, the overhead associated with the real-time sorting of incoming samples is too large. Reconfigurable FIR filter architectures are previously proposed for low power and delay implementations or to realize various frequency responses using a single filter. For low power architectures, variable input word-length and filter taps, different coefficient word-lengths, and dynamic reduced signal representation techniques are used. In those
works, large overhead is incurred to support reconfigurable schemes such as arbitrary nonzero digit assignment or programmable shift.

In this method (Hosangadi, A 2005), propose a simple, yet efficient, low power reconfigurable FIR filter architecture, where the filter order can be dynamically changed depending on the amplitude of both the filter coefficients and the inputs. In other words, when the data sample multiplied to the coefficient is so small as to mitigate the effect of partial sum in FIR filter, the multiplication operation can be simply canceled. The filter performance degradation can be minimized by controlling the error bound as small as the quantization error or signal to noise power ratio (SNR) of the given system. The primary goal of this work is to reduce the dynamic power of the FIR filter.

5.1.1. Transposed direct form of an FIR filter

As shown in Fig. 5.1, FIR filtering operation performs the weighted summations of input sequences, called as convolution sum, which are frequently used to implement the frequency selective low-pass, high-pass, or band-pass filters. Generally, since the amount of computation and the corresponding power consumption of FIR filter are directly proportional to the filter order, if the filter order can be dynamically changed by turning off some of the multipliers, significant power savings can be achieved. However, performance degradation should be carefully considered when the order of the filter is changed.

Fig5.1: Transposed direct form of an FIR filter
5.2 Programmable Shifts Method (PSM)

In this section, the architecture of the proposed FIR filter is presented (Mahesh.R 2010). Our architecture is based on the transposed direct form FIR filter structure as shown in Fig. 5.1.

![Architecture of the PSM method.](image)

The FIR filter architecture can be realized in a serial way in which the same PE is used for generation of all partial products by convolving the coefficients with the input signal \((h \ast x[n])\) or in a parallel way, where parallel PE architectures are employed. The first option is used when High speed and Delay are of prime concern. Following are the upcoming steps for multiplying the coefficients into the data.
The steps involved in PSM are as follows:

Step 1: Obtain the BCSs from filter coefficients using BCSE algorithm.
Step 2: Store the resultant coefficients in the prescribed format in the LUT.
Step 3: Get the input x.
Step 4: Get the coefficients from the LUT and use as the select signal for the multiplexers and the PS.
Step 5: Perform the final shifting function on the output of the multiplexer using PS.
Step 6: Perform the addition of intermediate sums using the final adder unit.
Step 7: Store the final result, h*x, in the delay unit ‘D’.
Step 8: Go to step 4 if the coefficients in the LUT are not finished, else go to 3.

The functions of different blocks of the PE are explained below.

1) Shift and Add Unit:

It is well known that one of the efficient ways to reduce the complexity of multiplication operation is to realize it using shift and add operations (Dempster.A.G 1995). In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, we use the BCSs-based shift and add unit in our PSM architectures. The architecture of shift and add unit is shown in Fig. 5.2. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Fig. 5.3, “x >>k ”represents the input x shifted right by k units. All the 3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bit number are generated using only three adders, whereas a conventional shift and add unit would require five adders. Since the shifts to obtain the BCSs are known beforehand, PS is not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit.

2) Multiplexer Unit:

The multiplexer units are used to select the appropriate output from the shift and add unit. All multiplexers will share the output of the shift and add units. The inputs to
the multiplexers are the 8/4 inputs from the shift and add unit and hence 8:1/4:1 multiplexer units are employed in the architecture. The select signals of the multiplexers are the filter coefficients which are previously stored in a look up table (LUT). The PSM architectures basically differ in the way filter coefficients are stored in the LUT. The number of multiplexers in the PSM is dependent on the number of non-zero operands in the coefficient for the worst case after the application of BCSE algorithm.

3) Final Shifter Unit:

The final shifter unit will perform the shifting operation after all the intermediate additions (i.e., intra-coefficient additions) are done. This can be illustrated using the output expression shift and add unit is shown in Fig. 5.3. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Fig. 3, “x >>k ”represents the input x shifted right by k units. All the 3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bit number are generated using only three adders, whereas a conventional shift and add unit would require five adders. Since the shifts to obtain the BCSs are known beforehand, PS is not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit.

4) Final Adder Unit:

This unit will compute the sum of all the intermediate additions 2−4(x +2−2x ) and 2−15(x +2−1x). As the filter specifications of different communication standards are different, the coefficients change with the standards. In conventional reconfigurable filters, the new coefficient set corresponding to the filter specification of the new communication standard is loaded in the LUT. Subsequently, the shift and add unit performs a bitwise addition after appropriate shifts. On the contrary, the PSM architectures perform a binary common sub expression (BCS)-wise addition (instead bitwise addition). Thus, the same hardware architecture can be used for different filter specifications to achieve the necessary reconfigurability. Moreover, the proposed BCS-based shift and add unit reduces addition operations and hence offers hardware
complexity reduction. Architecture can be used for different filter specifications to achieve the necessary reconfigurability. Moreover, the proposed BCS-based shift and add unit reduces addition operations and hence offers hardware complexity reduction.

Fig 5.3. Architecture of Shift/Add.

5.2.1 Architecture of PSM

The PSM is based on the BCSE algorithm. The PSM architecture presented in this section incorporates reconfigurability into BCSE. The PSM has a pre-analysis part in which the filter coefficients are analyzed using the BCSE algorithm. Thus, the redundant computations (additions) are eliminated using the BCSs and the resulting coefficients in a coded format are stored in the LUT. The number of multiplexer units required can be obtained from the filter coefficients after the application of BCSE. The number of multiplexers is selected after considering the number of non-zero operands (BCSs and unpaired bits) in each of the coefficients after the application of the BCSE algorithm. The number of multiplexers will be corresponding to the number of non-zero operands for the
worst-case coefficient (worst-case coefficient being defined as co-efficient that has the maximum number of non-zero operands).

The architecture of PE for PSM is shown in Fig. 5.4. The coefficient word length is fixed as 16 bits. The statistical analysis for various filters with coefficient precision of 16 bits and different filter lengths was performed, and it was found that the maximum number of non-zero operands is 5 for any coefficient of the filter.

Fig.5.4. Architecture of PE for PSM.
The LUT consists of two rows of 18 bits for each coefficient of the form SDDDDXXDDDDXXMMMMML and DDDDDXXDDDDXXDDDDXX, where “S” represents the sign bit, “DDDD” represents the shift values from $2^0$ to $2^{-15}$ and “XX” represents the input “x” or the BCSs obtained from the shift and add unit. In the coded format, XX = “01” represents “x”, “10” represents $x + 2^{-1}x$, “11” represents $x + 2^{-2}x$, and “00” represents $x + 2^{-1}x + 2^{-2}x$, respectively. Thus, the two rows can store up to five operands which is the worst case number of operands for a 16-bit coefficient. In most of the practical coefficients, the number of operands is less than the worst case number of operands, 5. In that case, “MMMML” can be used to avoid unnecessary additions. The values “MMMM” will be given as ‘select’ signal to the Mux6 and “L” to Mux8. “MMMML” indicates the presence of five operands.

5.3. Architecture of Proposed Reconfigurable FIR Filter

In this section, direct form (DF) architecture of the reconfigurable FIR filter is presented, which is shown in Fig 5.5 (a). In order to monitor the amplitudes of input samples and cancel the right multiplication operations, amplitude detector (AD) in Fig 5.5 (b) is used. When the absolute value is smaller than the threshold $C_{th}$, the output of AD is set to “1”. The design of AD is dependent on the input threshold $x_{th}$, where the fan-in’s of AND and OR gate are decided by $x_{th}$. If $C_{th}$ and $x_{th}$ have to be changed adaptively due to designer's considerations, AD can be implemented using a simple comparator. In the proposed reconfigurable filter, if we turn off the multiplier by considering each of the input amplitude only, then, if the amplitude of input abruptly changes for every cycle, the multiplier will be turned on and off continuously, which incurs considerable switching activities.

5.3.1 Multiplier Control Signal Decision window (MCSD)

This architecture is used to solve the switching problem. Using ctrlsignal generator inside MCSD, the number of input samples consecutively smaller than $x_{th}$ are counted and the multipliers are turned off only when consecutive input samples are
smaller than \( c_{th} \). As an input smaller than threshold \( x_{th} \), the output of the AD is set to “1”, the counter is counting up. When the counter reaches, the ctrl signal in the figure changes to “1”, which indicates that consecutive small inputs are monitored and the multipliers are ready to turn off. One additional bit, \( in_{ct\rightarrow n} \) in Fig.5.5 (a), is added and it is controlled by ctrl.

The \( in_{ct\rightarrow n} \) accompanies with input data all the way in the following flip-flops to indicate that the input sample is smaller than \( x_{th} \) and the multiplication can be cancelled when the coefficient of the corresponding multiplier is also smaller than \( c_{th} \).

Once the \( in_{ct\rightarrow n} \) signal is set inside MCSD, the signal does not change outside MCSD and holds the amplitude information of the input. A delay component is added in front of the first tap for the synchronization between \( x^*(n) \) and \( in_{ct\rightarrow n} \) in Fig 5.5(a) since one clock pulse is needed due to the counter in MCSD. In case of adaptive filters, additional ADs for monitoring the coefficient amplitudes are required. However, in the FIR filter with fixed or programmable coefficients, since the amplitude of coefficients ahead is known, extra AD modules for coefficient monitoring are not needed. When the amplitudes of input and coefficient are smaller than \( x_{th} \) and \( c_{th} \), respectively, the multiplier is turned off by setting control signal \( \varphi_n \) to “1”.

The area overheads of the proposed reconfigurable filter are flip-flops for \( in_{ct\rightarrow n} \) signals, AD and ctrl signal generator inside MCSD and the modified gates in Fig 5.5 (b) for turning off multipliers. Those overheads can be implemented using simple logic gates, and a single AD is needed for input \( x(n) \) monitoring as specified in Fig 5.5(a). Consequently, the overall circuit overhead for implementing reconfigurable filter is as small as a single multiplier.

Multiplier Control Signal Decision window (MCSD) is used to reduce the frequency of switching activity. There is a control signal generator within the MCSD window which reduces the switching problem. The control signal generator consists of an
internal counter which counts the number of input samples for which the condition $x[n] < x_{th}$ is satisfied.

That is, the counter starts counting up whenever the output of the amplitude detector (ad_out) is set to “1”. When the count reaches certain values (say m), the output of the control signal generator (ctrl) becomes “1”, where m is the size of MCSD. So, a “1” value on ctrl is an indication that m consecutive small inputs are monitored and multipliers are ready to turn off. The signal ctrl also controls an additional signal $in_{ct-n}$. This signal accompanies with the input data all the way in the following flip-flops to indicate that $x[n] < x_{th}$, and the multiplications can be cancelled if the corresponding filter coefficient amplitude is smaller than $c_{th}$.

The procedure of turning the multiplier off after getting the filter coefficient and data sample smaller than the corresponding threshold is as follows. When the amplitude of the input data sample and filter coefficient is smaller than the threshold, the signal $\varphi n$ is set to “1”. Whenever the signal $\varphi n$ is set to logic “1”, the multiplier will be turned off by a simple logic circuit and the output in turn forced to zero.

An important thing to consider while designing the reconfigurable filter section is the values for the threshold $x_{th}$ and $c_{th}$, which has a significant impact on the filter performance and power consumption. If the threshold values are very large, it can give rise to large power savings at the cost of filter performance.

On the other hand if the threshold values are small, then the power savings become trivial. Similarly the values of m, indicating the size of MCSD also have significant impact on the power savings. So, if m becomes larger, then the number of input samples that makes multipliers turned off decreases.
5.4 Results & Simulation

The results presented establish a clear area advantage of the proposed FIR filter architecture over prior architecture for typical FIR filter parameters with comparable maximum clock rates. MCSD architecture achieved high clock frequency compared to PSM architecture. Due to logic depth high in PSM it will operate at lesser speed, but in the proposed architecture gives high speed and low power. These techniques were validated on Spartan-III devices where the observation of significant area and
power reductions over traditional Distributed Arithmetic based techniques and multiplier less technique were found.

Fig5.6. Simulation Result of Proposed Reconfigurable FIR filter.
Fig5.7 Synthesis Result of Proposed Reconfigurable FIR filter for delay and frequency.
Fig 5.8 Synthesized RTL view of Proposed Reconfigurable FIR filter.
Table 5.1 Comparison of Delay, Clock Frequency and Power of PSM and MCSD.

<table>
<thead>
<tr>
<th>METHODS</th>
<th>PSM</th>
<th>MCSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delay (ns)</td>
<td>6.296</td>
<td>4.319</td>
</tr>
<tr>
<td>Clock Frequency (MHz)</td>
<td>158.827</td>
<td>231.511</td>
</tr>
<tr>
<td>Power (mW)</td>
<td>0.300</td>
<td>0.268</td>
</tr>
</tbody>
</table>

Fig. 5.9 Performance comparison of PSM and MCSD.
5.5 Conclusion

This research work proposed a novel approach, namely Multiple Control Signal Decision Window (MCSD) technique, for implementing reconfigurable higher order filters for low power and high speed applications. The MCSD method provides the flexibility of changing the filter coefficient word lengths dynamically. The proposed scheme implemented on Spartan-III XC3S200-5PQ-208 FPGA and synthesized. The proposed reconfigurable architecture provides high speed and low power than the existing Programmable Shift Methods.