Design and Implementation of Polyphase Decimation Filter

Pooja Jacob
Research Scholar, VLSI and Embedded Systems
St. Joseph's College of Engineering and Technology
Palai, India
poojaplath@gmail.com

Mr.Anoop B.N
Asst. Professor, Electronics and Communication
St. Joseph's College of Engineering and Technology
Palai, India
anoopcem@gmail.com

Abstract—**In most of today's applications like computer communication, multimedia systems, digital audio broadcast equipments etc, a very high quality sample rate converter(SRC) is required. Most high quality SRC's employ a digital filter that provides the required quality by upsampling the data to a high sampling rate followed by downsampling the data to the required output sampling rate. Polyphase Decimation filters (PPD) are a class of digital filters that operates at a lower sampling rate compared to conventional filters.The objective of this paper is to design and implement a polyphase decimation filter operating at a frequency fs/M for an input clock frequency fs and a decimation factor M. PPD was implemented and functionally verified using Xilinx ISE simulator 14.6**

Keywords-VHDL,SRC,Polyphase Decomposition*,*Decimation

I. INTRODUCTION

Digital signal processing algorithms are increasingly employed in many applications like wireless communication, consumer electronics such as cellular telephones and digital cameras. Advancements in Field Programmable Gate Arrays provide a new option for efficient implementation of DSP algorithms[1].A frequent task in digital signal processing is to adjust the sampling rate according to the signal of interest.

Sample rate conversion(SRC) is the process of changing sampling rate of data stream from a specific sampling rate to another sampling rate.SRC is a necessary component in many of today's applications, like CDs, Audio players, Tape recorders, WiMAX, WiFi etc. Systems with different sampling rates are referred to as multirate systems[2].Polyphase is a method of doing sampling rate conversion that leads to very efficient implementation.

Decimation[3] reduces the sampling rate at the output of a system so that another system with a lower sampling rate can receive this signal as input. A narrow filter followed by a downsampler is referred to as a decimator. Decimator can reduce the sampling rate up to the limit called the Nyquist rate, according to which the sampling rate must be higher than the bandwidth of the signal to avoid aliasing. Multiple stages of decimation can reduce computational and memory requirements of filters. Also it reduces the cost for processing because the memory and calculations for implementing a DSP system is proportional to the sampling rate.Reducing sampling rate results in a cheaper implementation. Downsampling by a factor M is implemented by keeping every Mth sample and throwing away M-1 samples in between. Polyphase Decimation Filter is a digital filter(FIR/IIR) which is implemented in a polyphase decomposition[4]. In Polyphase implementation first the signals are decimated and then filtered. The filter used in this paper is an FIR filter, because in FIR filters output is a function of past inputs only. Therefore only the outputs that will be used needs to be calculated, while in IIR filters for each input output needs to be calculated.

II. NOBLE IDENTITIES

Noble identities describe the property of reverse ordering the filter and upsampler/downsampler as shown in Fig.1.

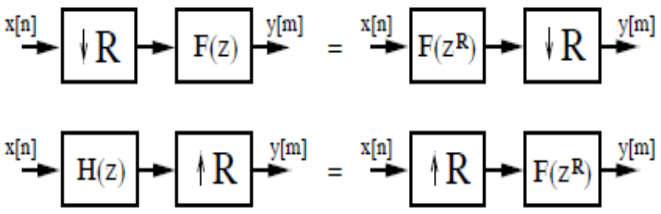

Fig.1: Equivalent Multirate Systems(Noble Identities)

For decimator the Noble identity follows:

$$(\downarrow R)\, F(z) = F(z^R)\, (\downarrow R) \qquad (1)$$

Downsampling followed by filtering can reduce the filter length $F(z^R)$ by a factor R.

For interpolator the Noble identity follows:

$$F(z)\ (\uparrow R) = (\uparrow R)\, F(z^R) \qquad (2)$$

In an interpolator putting the filter before expander results in an R-times shorter filter. Both the identities are very useful when it comes to polyphase implementation.

III. POLYPHASE DECOMPOSITION

While decimating, a signal is downsampled by throwing away the intermediate samples which will result in aliasing. To prevent aliasing an anti-aliasing lowpass filter, H(z) is employed before downsampler as shown in Fig.2.

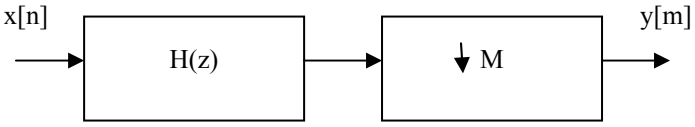

Fig.2 : Decimator with anti-aliasing filter and downsampler

The decimator shown in Fig.2 is computationally inefficient because it throws away the filtered samples. By using noble identity it is possible to rearrange the structure such that the filtered outputs are not thrown away. For that first the filter H(z) should be decomposed to polyphase components.

$$H(z) = \sum_{i=0}^{M-1} z^{-i} H_i(z^M) \qquad (3)$$

In other words ,the coefficients h(n) are decomposed into M polyphase filters, each with N/M taps, where N is the number of taps in the prototype filter and M is the decimation factor. This decomposition can be written in Z - transform of h(n) in the following way:

$$\begin{aligned} & h(0) + h(M+0)z^{-M} + h(2M+0)z^{-2M} + \ldots.. \\ & h(1)z^{-1} + h(M+1)z^{-(M+1)} + h(2M+1)z^{-(2M+1)} + \ldots \\ H(z) = \; & h(2)z^{-2} + h(M+2)z^{-(M+2)} + h(2M+2)z^{-(2M+2)} + \ldots \\ & \ldots\ldots\ldots\ldots\ldots\ldots \\ & h(M-1)z^{-(M-1)} + h(2M-1)z^{-(2M-1)} + \ldots\ldots \qquad (4) \end{aligned}$$

or,

$$\begin{aligned} H(z) = \; & H_0(z^M) \\ & + z^{-1}H_1(z^M) \\ & + z^{-2}H_2(z^M) \qquad (5) \\ & + \ldots\ldots\ldots \\ & + z^{M-1}H_{M-1}(z^M) \end{aligned}$$

Each row in equation (5) represents a polyphase filter.Fig.3 performs processing according to equation (5) and then performs downsampling.

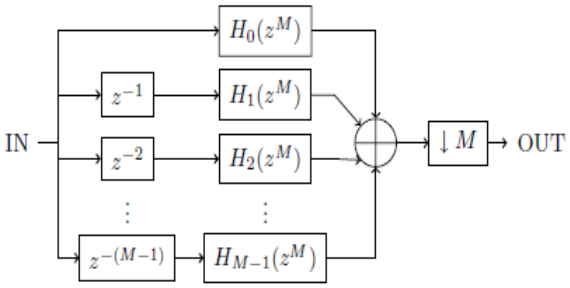

Fig.3: Polyphase filter implementation based on equation (5)

Now Noble identity can be applied by first downsampling and then polyphase filtering. The block diagram showing this operation is shown in Fig.4.

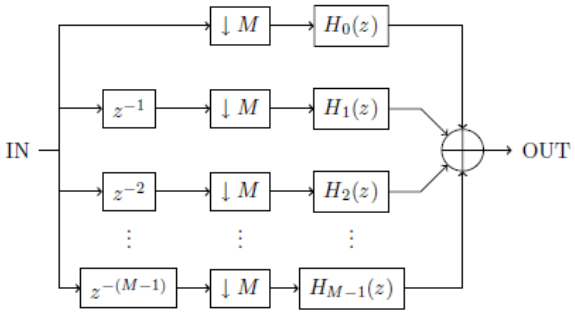

Fig.4 : Decimation polyphase filter using Noble identity

IV. DECIMATOR FILTER IMPLEMENTATION

Decimator filter implementation was based on a Finite State Machine (FSM) which divides the input stream as even and odd samples to half the sampling rate(M=2).FIR(Finite Impulse Response)[5] low pass filter was employed for filtering followed by downsampling. The FIR filter used was a Transposed form of a 4-Tap FIR Filter as shown in Fig.5:

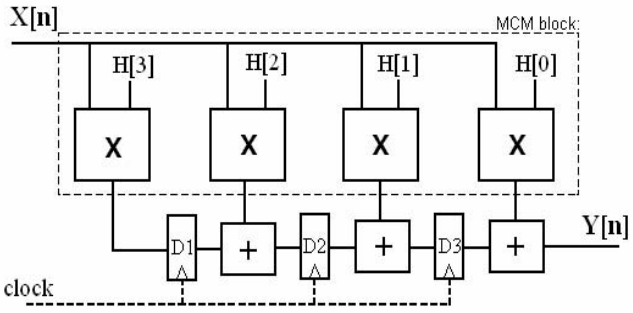

Fig.5 :Transposed form of a 4-tap FIR Filter Implementation

An FIR Filter is given by the equation:

$$Y[n] = \sum_{i=0}^{N-1} H[i]\, X[n-i] \qquad (6)$$

where N is the number of coefficients(or taps) of the filter, X is the input signal, Y is the output signal, Y[n] is the current output sample and H represents the filter coefficients.

A fully parallel implementation of FIR filters was used. The architecture involves multiplying all coefficients with the same input data. The circuit is faster because the registers are between the adder blocks reducing the critical path. Due to register localization most of the glitching activity is reduced, resulting in a significant power reduction.

The FSM based on which the decimator filter was implemented is shown in Fig 6:

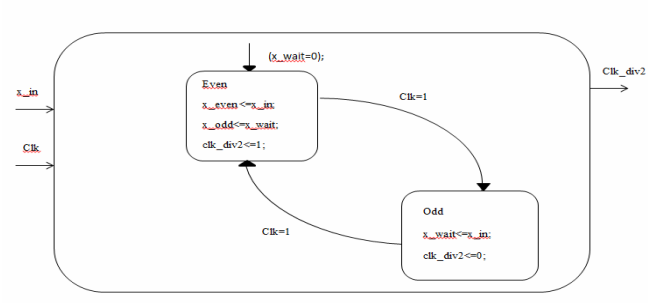

Fig.6: State machine flowchart

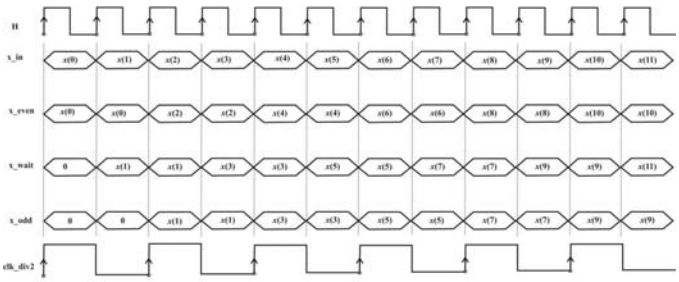

Fig.7: The decimator timing diagram

By carefully designing the coefficients, the structure allows to obtain a high performance and easier implementation of the decimator filter.

V. POLYPHASE DECIMATION FILTER IMPLEMENTATION

For Polyphase filter implementation first downsampling by a factor two was done and then filtering using linear convolution technique was done.

Convolution is a mathematical process that relates output, y(t),of a linear time invariant system to its input ,x(t),and impulse response, h(t).Convolution can be expressed as:

$$y(t) = x(t) * h(t) \qquad (7)$$

The continuous time relationship is described by convolution integral

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau \qquad (8)$$

For infinite discrete sequences, this integral relation reduces to convolution sum and is given by:

$$y(n) = \sum_{k=-\infty}^{\infty} h(k).x(n-k) \qquad (9)$$

Linear convolution takes two functions of an independent variable and convolves them using convolution sum formula given in equation(9).It is a correlation of one function with time-reversed version of the other function. It includes a flip, multiply and sum while shifting one function with respect to the other. Convolution is a filtering process, or more appropriately filtering is an application of convolution.

Sample MATLAB Code Based on which the filter was designed is given below:

MATLAB implementation

```
x=0:10;
h=[1 2 3 4 5 6];
y=filter(h,1,x)
y_dec=y(1:2:end)
p0=h(1:2:end)
p1=h(2:2:end)
x0=[0 x(3:2:end)]
x1=[0 x(2:2:end)]
y_poly=filter(p0,1,x0)+filter(p1,1,x1)
```

In the MATLAB code above ‘x’ represents the filter inputs, ‘h’ represents the coefficients, ‘y’ represents the filtered output and ‘y_dec’ represents the decimated output. In polyphase implementation first the filter inputs and the coefficients are decimated and is represented by x0,x1,p0,p1 respectively. Finally the decimated values are filtered and ‘y_poly’ represents the polyphase filter outputs.

VI. SIMULATION RESULTS

First a 4-Tap FIR Filter was designed in VHDL using the transposed filter architecture and the results were compared with the MATLAB results. The coefficients used for simulation was -5,-4 ,3 ,2

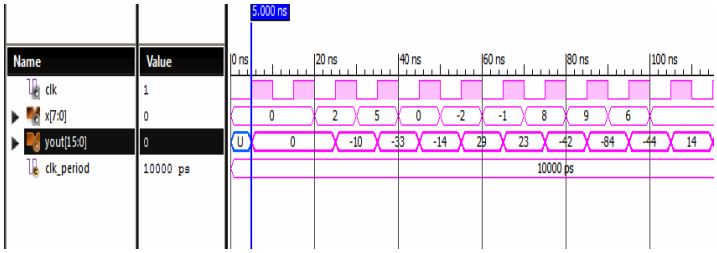

Fig 8:VHDL implementation of a 4-tap FIR Filter

Decimator filter was implemented using the FSM in Fig.6 First the input samples were filtered using transposed FIR filter architecture and the filtered output was divided into even and odd samples to half the sampling rate.

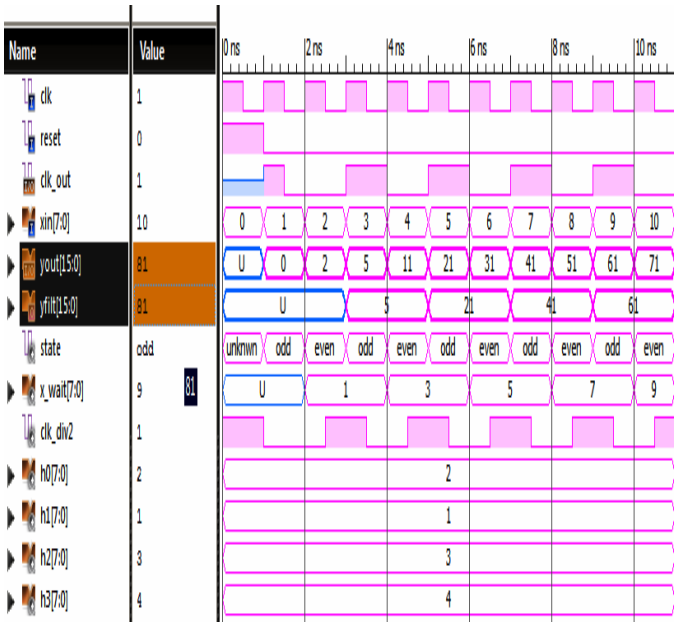

Fig 9: VHDL implementation of a decimator filter

In the above figure 'yout' represents the filtered output and 'yfilt' represents the decimated filter outputs. When reset ='1' the FSM goes to an unknown state.'clk_out' represents the output clock which is downsampled by a factor M=2. The coefficients used were 2,1,3,4.The clk input was 1 ns and the output clock was observed to be 2 ns.

The polyphase filter implementation was based on the MATLAB code above and the structure could downsample upto a factor '3'.clk_div2 in Fig.10 shows the output downsampled clock with a clock period of 3us and 'y9' in the figure shows the polyphase filter outputs array. In polyphase implementation inputs(x) and coefficients(h) were first decimated by '3'and was stored in arrays a0,a1,a2 ,p0,p1,p2 respectively. The results were compared with the MATLAB outputs.

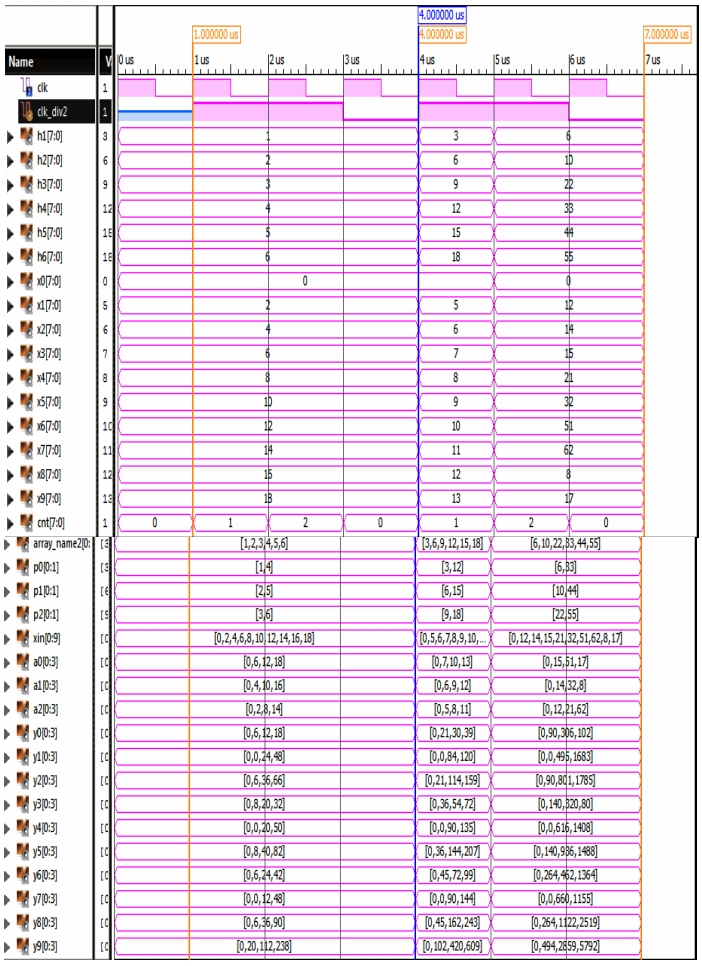

Fig.10:VHDL implementation of Polyphase decimation filter

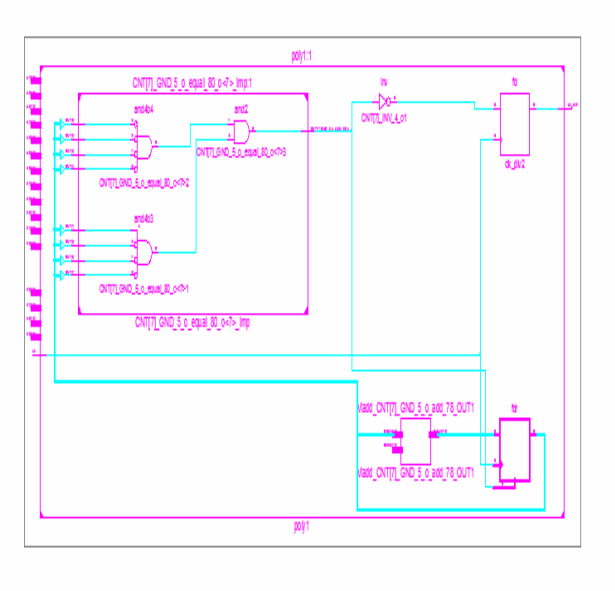

Fig.11:RTL Schematics Of Polyphase Filter

VII. CONCLUSION

Though the implementation of signal processing systems on ASICs provide better optimized devices, the cost involved will be high. But with advancements in FPGA, filter implementation on FPGA utilizing dedicated hardware resources can effectively achieve application specific integrated circuit(ASIC)-like performance while reducing development time and cost. In this paper, design of FIR filter, Decimation filter and Polyphase Decimation filters was done and the result was verified on Xilinx ISE Simulator 14.6 platform. Implementation was done of Virtex-6 FPGA board and the results prove that polyphase decimation filter can reduce the sampling rate of the input data stream compared to conventional filters.

REFERENCES

[1] R. Seals and G. Whapshott, 1997 Programmable Logic: PLDs and FPGAs. UK: Macmillan

[2] Uwe Meyer-Baese., Digital Signal Processing with Field Programmable Gate Arrays, Third Edition, ISBN 978-3-540-72612-8 Springer Berlin Heidelberg NewYork.

[3] E. B. Hogenauer, ―"An economical class of digital filters for decimation and interpolation", IEEE Transactions on Acoustic, Speech and Signal Processing, ASSP29(2):155-162, 1981.

[4] P. P. Vaidyanathan, V. C. Liu, "Classical sampling theorems in the context of multirate and poly-phase digital filter bank structures," IEEE Trans. Acoust., Speech, Signal Processing, vol. 36, pp. 1480-1495, Sept. 1988.

[5] N. Sankarayya, K. Roy, and D. Bhattacharya, "Algorithms for low power high speed FIR filter realization using differential coefficients," IEEETrans. Circuits Syst. II, vol. 44, pp. 488–497, June 1997

AUTHORS PROFILE

Mr. Anoop B.N received the Bachelors degree in Electronics and Communication from College of Engineering, Munnar. He received his Masters degree in Signal Processing from National Institute Of Technology, Calicut. He is currently working as an Assistant Professor in the Department of Electronics and Communication at St.Joseph's College of Engineering and Technology, Palai

Pooja Jacob received the Bachelors degree in Electronics and Communication from College of Engineering, Munnar affiliated to Cochin University of Science and Technology in 2012.Presently she is doing M.Tech in VLSI and Embedded Systems from St.Joseph's College of Engineering and Technology, Palai.