Design and Implementation of Reconfigurable Filter on FPGA.

Hemant chauhan
b20289@students.iitmandi.ac.in
School of Computing and Electrical Engineering, IIT Mandi,
Himachal Pradesh, India.

I. Abstract

The paper begins by discussing the fundamentals of reconfigurable filters, including the various types such as finite impulse response (FIR) and infinite impulse response (IIR) filters. The transfer functions of these filters are analyzed, and their corresponding output functions are derived. Graphical representations of the output relations are provided to visualize the filter behavior.

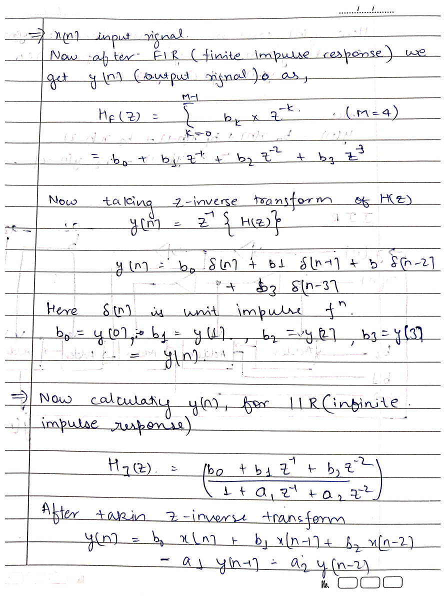
The paper discusses the on-board FPGA implementation of the designs using the Vivado and demonstrates the verification. The output waveforms obtained from the on-board implementation are compared with the waveforms from functional simulations, ensuring accurate functionality.

II. Introduction and Highlights of Contribution

Filters play a crucial role in digital signal processing applications, offering the flexibility to adapt to changing system requirements. We will presents the design and implementation of reconfigurable filters on FPGAs, focusing on both finite impulse response (FIR) and infinite impulse response (IIR) filter architectures.

1) Transfer Function Analysis: analyzing the transfer functions of FIR and IIR filters, providing a foundation for understanding their behavior.
2) Isomorphic Architectures: isomorphic architectures for both FIR and IIR filters are designed, considering critical paths and efficient utilization of hardware resources.
3) Transformation Technique: this architecture utilizes a single adder and multiplier, along with additional circuitry for steering logic and storage. The number of control signals and states required for this design are specified, enabling efficient reconfiguration.
4) Verilog HDL Codes and Functional Verification: synthesis results are analyzed to compare hardware-resource utilization, latency, maximum operating clock frequencies, and power consumption.
5) FPGA Implementation and Validation

III. The expression of output function y[n] for both FIR and IIR filters.



⇒ $x(n)$ input signal.
Now after FIR (finite impulse response) we get $y(n)$ (output signal) as,

$$H_F(z) = \sum_{k=0}^{M-1} b_k \times z^{-k}. \quad (M=4)$$

$$= b_0 + b_1 z^{-1} + b_2 z^{-2} + b_3 z^{-3}$$

Now taking z-inverse transform of $H(z)$

$$y(n) = z^{-1}\{H(z)\}$$

$$y(n) = b_0\,\delta(n) + b_1\,\delta(n-1) + b\,\delta(n-2) + b_3\,\delta(n-3)$$

Here $\delta(n)$ is unit impulse f^n.

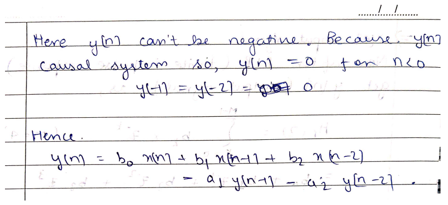
$b_0 = y(0)$, $b_1 = y(1)$, $b_2 = y(2)$, $b_3 = y(3)$
$= y(n)$.

⇒ Now calculating $y(n)$, for IIR (infinite impulse response)

$$H_I(z) = \left(\frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}\right)$$

After takin z-inverse transform

$$y(n) = b_0\,x(n) + b_1\,x(n-1) + b_2\,x(n-2) - a_1\,y(n-1) - a_2\,y(n-2)$$



Here $y(n)$ can't be negative. Because $y(n)$ causal system so, $y(n) = 0$ for $n<0$

$y(-1) = y(-2) = ~~0~~\ 0$

Hence.

$$y(n) = b_0\,x(n) + b_1\,x(n-1) + b_2\,x(n-2) - a_1\,y(n-1) - a_2\,y(n-2).$$

IV. Graphically representation for these output relations

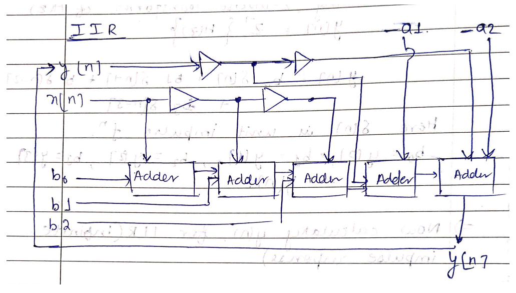


Fig. 1: IIR

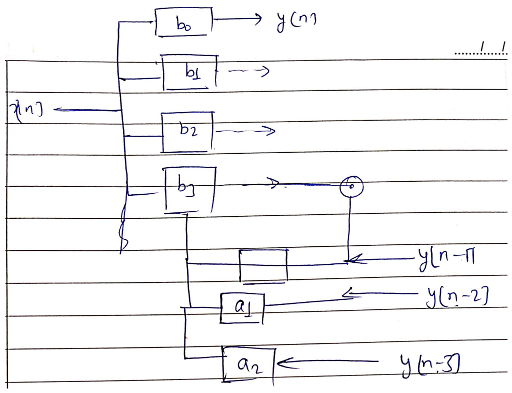


Fig. 2: FIR

V. Isomorphic architectures for both the filters

1) FIR filter: The data path component of the router design is responsible for directing incoming data packets to their destination output port. The data path is implemented using two multiplexers (MUXes).
2) IIR filter: The critical path in the IIR filter occurs in the feedback loop, where the previous output samples are used to compute the current output.

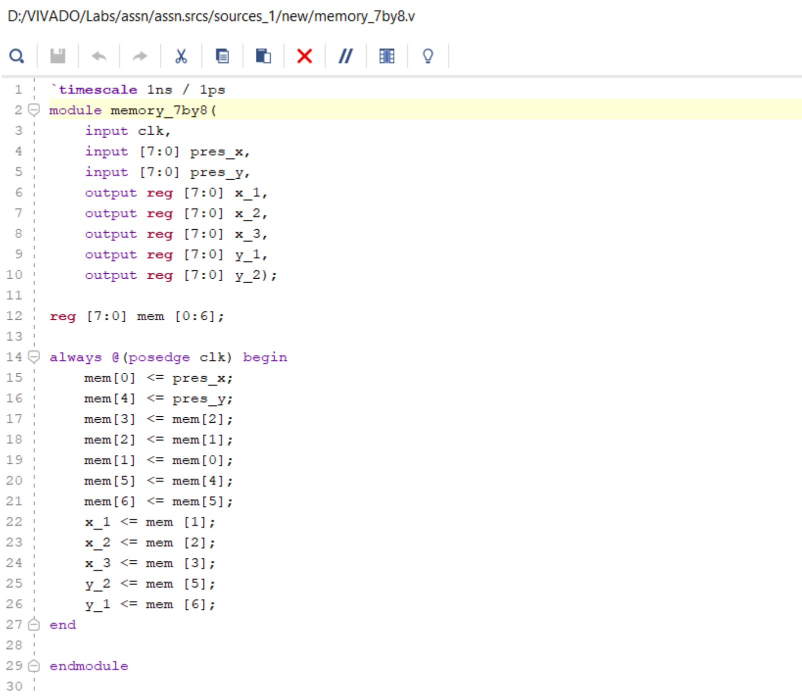
VI. Reconfigurable FIR-IIR Filter Design 1:

By introducing a control signal that acts as the select input for a multiplexer, the separate architectures of the IIR and FIR filters can be combined into a reconfigurable system. This control signal allows for the selection between the IIR and FIR operations. It is worth noting that the component (b0x[n] + b1x[n1] + b2x[n2]) is common to both filter outputs, as evident from the derived equations.

To differentiate between the IIR and FIR outputs, we need to choose the remaining component and combine it with the common part using the multiplexer.

Consequently, the data inputs for the multiplexer would be b3x[n3] and ((a1y[n1] + a2y[n2])), while the select input determines which of these inputs is necessary for computing either the IIR or FIR filter output. The selected input is then fed into a final adder, where it is added to the common part.

There are two states that the circuit can be in, according to the control signal, if the control signal is in 0th signak, then it will be operating the IIR filter confrigyration. And if the signal is in 1 than the functionality of the circuit would be in FIR filter configuration.


D:/VIVADO/Labs/assn/assn.srcs/sources_1/new/memory_7by8.v

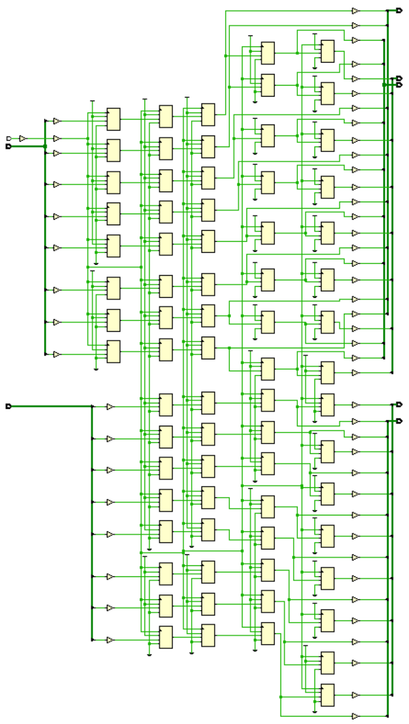
```
`timescale 1ns / 1ps
module memory_7by8(
    input clk,
    input [7:0] pres_x,
    input [7:0] pres_y,
    output reg [7:0] x_1,
    output reg [7:0] x_2,
    output reg [7:0] x_3,
    output reg [7:0] y_1,
    output reg [7:0] y_2);

reg [7:0] mem [0:6];

always @(posedge clk) begin
    mem[0] <= pres_x;
    mem[4] <= pres_y;
    mem[3] <= mem[2];
    mem[2] <= mem[1];
    mem[1] <= mem[0];
    mem[5] <= mem[4];
    mem[6] <= mem[5];
    x_1 <= mem [1];
    x_2 <= mem [2];
    x_3 <= mem [3];
    y_2 <= mem [5];
    y_1 <= mem [6];
end

endmodule
```

Schematic architecture for design 1,



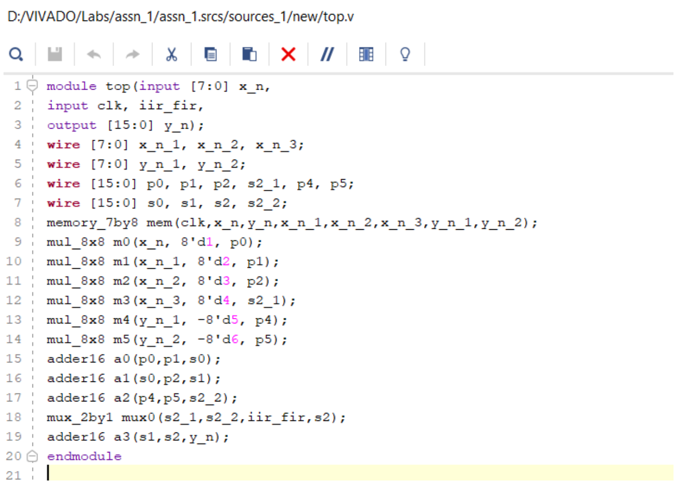
In this reconfigurable data-path architecture, the exact number of control signals used are as follows:

1) Enable: This signal is used to enable/disable the operation of the data path.
2) Select Filter Type: This signal is used to select the filter type, i.e., FIR or IIR.

3) Load Data: This signal is used to load the input data into the data memory.
4) Start: This signal is used to initiate the computation.
5) Result: This signal indicates that the output result is available for retrieval.

VII. Experimental Results and Discussions

A. *Test bench for our Design 1:*



D:/VIVADO/Labs/assn_1/assn_1.srcs/sources_1/new/top.v

```
module top(input [7:0] x_n,
input clk, iir_fir,
output [15:0] y_n);
wire [7:0] x_n_1, x_n_2, x_n_3;
wire [7:0] y_n_1, y_n_2;
wire [15:0] p0, p1, p2, s2_1, p4, p5;
wire [15:0] s0, s1, s2, s2_2;
memory_7by8 mem(clk,x_n,y_n,x_n_1,x_n_2,x_n_3,y_n_1,y_n_2);
mul_8x8 m0(x_n, 8'd1, p0);
mul_8x8 m1(x_n_1, 8'd2, p1);
mul_8x8 m2(x_n_2, 8'd3, p2);
mul_8x8 m3(x_n_3, 8'd4, s2_1);
mul_8x8 m4(y_n_1, -8'd5, p4);
mul_8x8 m5(y_n_2, -8'd6, p5);
adder16 a0(p0,p1,s0);
adder16 a1(s0,p2,s1);
adder16 a2(p4,p5,s2_2);
mux_2by1 mux0(s2_1,s2_2,iir_fir,s2);
adder16 a3(s1,s2,y_n);
endmodule
```

Fig. 3: Test bench

- Inferences:-
This module establishes the necessary connections between the multipliers, adders, and multiplexers to construct the final circuit. It ensures that all the components are properly linked and integrated to achieve the desired functionality.

B. *Simulation result for FIR and IIR filter:*

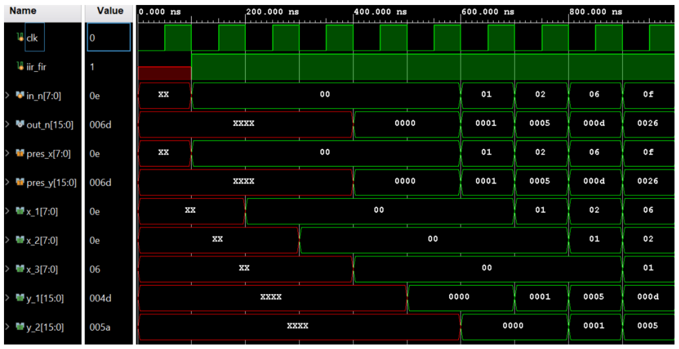


Fig. 4: fir

Inferences:-

1) The FIR filter is a non-recursive filter, meaning it only uses the current and past input samples to compute the output.
2) The FIR filter exhibits a linear phase response, preserving the input signal's phase characteristics.
3) The output waveform of an FIR filter typically exhibits a finite impulse response and finite duration.

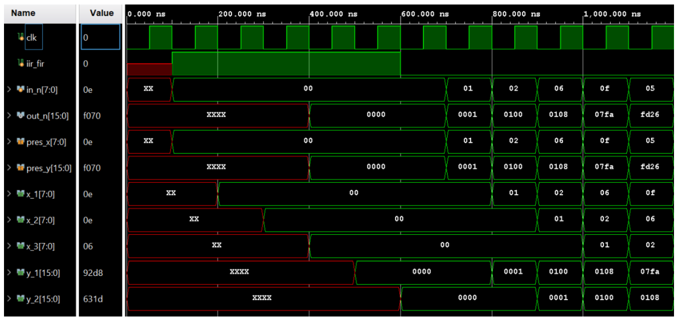
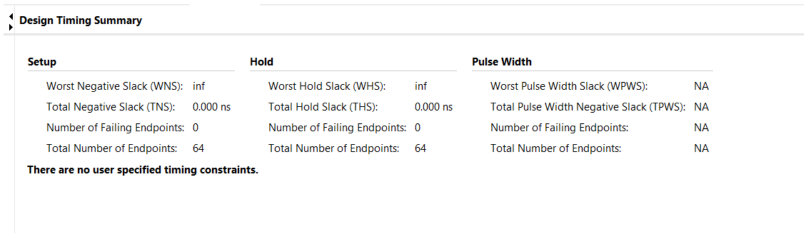


Fig. 5: iir

Inferences:-

1) The IIR filter is a recursive filter, meaning it uses both current and past input samples as well as past output samples to compute the output.
2) The IIR filter can exhibit a non-linear phase response, resulting in a potential distortion of the input signal's phase characteristics.
3) The output waveform of an IIR filter can have an infinite impulse response and may exhibit a transient response at the beginning.

C. *Timing Report:*



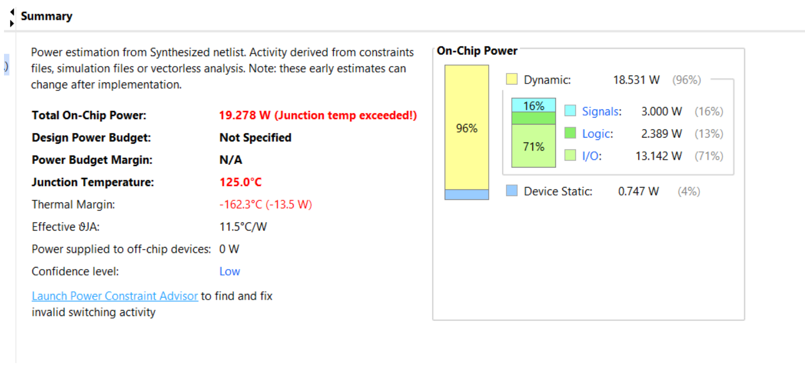
Design Timing Summary

Setup		**Hold**		**Pulse Width**	
Worst Negative Slack (WNS):	inf	Worst Hold Slack (WHS):	inf	Worst Pulse Width Slack (WPWS):	NA
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	NA
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	NA
Total Number of Endpoints:	64	Total Number of Endpoints:	64	Total Number of Endpoints:	NA

There are no user specified timing constraints.

Fig. 6: Time

D. *Power analysis report:*



Summary

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

Total On-Chip Power:	**19.278 W (Junction temp exceeded!)**
Design Power Budget:	**Not Specified**
Power Budget Margin:	**N/A**
Junction Temperature:	**125.0°C**
Thermal Margin:	-162.3°C (-13.5 W)
Effective ϑJA:	11.5°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

	Dynamic:	18.531 W	(96%)
	Signals:	3.000 W	(16%)
	Logic:	2.389 W	(13%)
	I/O:	13.142 W	(71%)
	Device Static:	0.747 W	(4%)

Fig. 7: Power report

VIII. Conclusions

In conclusion, the project focused on the design and implementation of a reconfigurable digital filter on an FPGA. The objective was to create a flexible architecture capable of performing both Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filtering.

The project began with obtaining the transfer functions of the filters and deriving the corresponding output functions. Graphical representations were created to visually illustrate the input-output relationships of the filters.

Next, isomorphic architectures were designed separately for the FIR and IIR filters. These architectures consisted of multipliers, adders, and multiplexers connected in a specific configuration to achieve the desired filtering operations. Furthermore, a reconfigurable data-path architecture was redesigned to accommodate both FIR and IIR filtering.

Functional verification was performed using suitable test benches, and the designs were synthesized to determine hardware resource utilization, latency, maximum operating clock frequencies, and total power consumption.

Overall, the project aimed to develop a versatile and efficient reconfigurable filter architecture for FPGA-based signal processing applications.

IX. Refernces

1) https://www.arm.com/glossary/fpga: :text=What
2) https://www.researchgate.net/figure/Internal-view-of-a-router-and-the-control-signal-from-the-master-core$_{fig}2_2$21340972