

**Implementation of Programmable IIR Filters for Software-Defined Radio on FPGA**

# 1. Project Description

Background:

Software-defined radio (SDR) technology has revolutionized wireless communication systems by enabling flexible and reconfigurable radio platforms. One critical component of an SDR system is the digital signal processing (DSP) stage, where filters play a crucial role in extracting, manipulating, and transmitting signals efficiently. In this project, we aim to design and implement programmable Infinite Impulse Response (IIR) filters on an FPGA platform to enhance the signal processing capabilities of an SDR receiver.

Problem Statement:

Traditional SDR systems often use fixed filters that are limited in their adaptability to varying signal characteristics and interference conditions. The goal of this project is to develop programmable IIR filters that can dynamically adjust their parameters to effectively suppress interference while maintaining high signal fidelity. This will enable the SDR receiver to handle a wide range of signals and adapt to changing environmental conditions in real-time.

Project Objectives:

1. Design a set of programmable IIR filters with adjustable parameters such as cutoff frequency, bandwidth, and filter order.
2. Implement the IIR filter architecture using Verilog/VHDL, considering fixed-point or floating-point arithmetic based on resource availability and performance requirements.
3. Develop user-friendly interfaces to configure filter parameters and control filter operation.
4. Integrate the programmable IIR filters into the receive chain of an SDR system, ensuring compatibility with other signal processing blocks.
5. Optimize resource utilization of the IIR filter implementation through algorithmic optimization, resource sharing, and pipelining.
6. Validate the functionality and performance of the programmable IIR filters through comprehensive testing and performance analysis.

Approach:

* **Filter Design:** Select appropriate filter specifications based on SDR requirements, considering passband and stopband frequencies, stopband attenuation, and filter order.

* **IIR Filter Implementation:** Choose a suitable IIR filter topology (e.g., Butterworth, Chebyshev, Elliptic) and implement the filter structure using Verilog/VHDL.

* **Tuning and Configuration Interface**: Design user-friendly interfaces using communication protocols (e.g., UART, SPI) to enable external configuration of filter settings.

* **Integration with SDR Receiver Chain:** Develop interface logic to interface the IIR filters with ADC data streams and downstream processing modules.

* **Resource Optimization:** Explore optimization techniques to reduce resource utilization, such as algorithmic optimization and pipelining.

* **Testing and Validation:** Develop comprehensive testbenches to validate the functionality and performance of the programmable IIR filters under different operating conditions.

**2/3. Proposed Solution & Work Breakdown:**

1. Filter Design Parameters:

* + Passband Frequency: 100 kHz to 2.5 MHz
  + Stopband Frequency: 105 kHz to 2,505 kHz
  + Transition Band: 5 kHz
  + Passband Attenuation: 1 dB
  + Stopband Attenuation: 60 dB
  + Filter Order: 10 to 14 taps based on frequency requirements

1. IIR Filter Implementation:

* + Topology: Cascade IIR Filter of Elliptic topology
  + Implementation: Verilog/VHDL using fixed-point arithmetic
  + Coefficient Configuration: USART protocol for coefficients and input from the host PC

1. Tuning and Configuration Interface:

* + User Interface: Designed for configuring filter parameters (cutoff frequency, bandwidth)
  + Communication Protocol: Implemented USART protocol for coefficient and input data transmission from the host PC to the FPGA

1. ADC Interfacing with IIR Filter:

* + Integration: Developed interface logic for interfacing the IIR filters with ADC data streams
  + Compatibility: Ensured compatibility with downstream processing modules in the SDR receiver chain

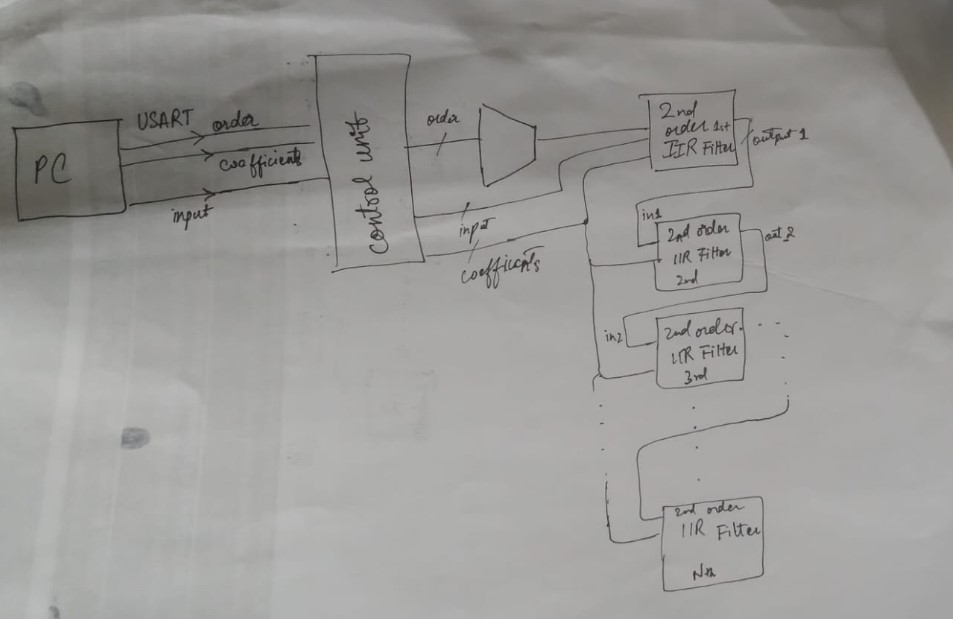
1. Resource Optimization:

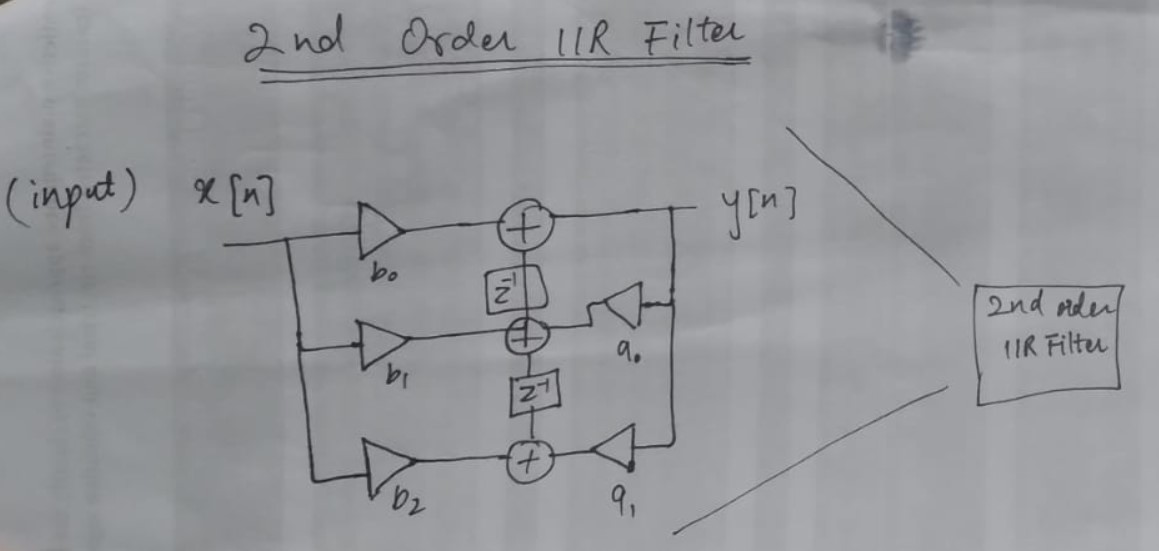
* + Techniques: Explored optimization techniques including algorithmic optimization, resource sharing, and pipelining
  + Synthesis: Utilized FPGA synthesis tools for synthesis and optimization to meet timing constraints and minimize resource utilization

1. Testing and Validation:

* + Testbenches: Developed comprehensive testbenches to validate functionality and performance under various operating conditions
  + Verification: Conducted simulation-based verification and hardware testing on FPGA hardware to ensure correct filter behavior and compliance with design specifications

1. Micro Architecture Diagram





# 4. Tasks Allocation

|  |  |
| --- | --- |
| **Group Member** | **Task** |
| Muhammad Omais | Implementation of Filter and Communication between host and kit |
| Ahmad Razi Ullah | Control unit and instantiation of filter taps |
| Muhammad Hassan Javed | Filter specifications, topology, and coefficients generation |
| Muhammad Saad Faran Malik | Testbenches and validation |

**5. Outcome:**

The user sets the input specifications (frequency) through pc, based on which the input, order of filter and its coefficients are fed to FPGA via USART Communication protocol and get stored in memory. The control unit processes the order provided by user and based on it selects the number of taps for designing the filter. In this way the taps of filter get programmed based on the frequency specifications.