ECE574 Final Project Report: Reconfigurable Partially Pipelined RAM-based FIR Filter

Brian B Cheng
Rutgers University Department of Electrical and Computer Engineering

1 Abstract

This report presents a Verilog implementation of a reconfigurable Finite Impulse Response (FIR) filter designed for heterogeneous FPGAs. The model is highly flexible, with customizable data width, filter depth, and the number of parallel multiply-accumulate (MAC) pipelines, all configurable through top-level parameters. The filter modifies the traditional direct-form FIR structure by replacing the input shift register with RAM elements and the stationary weights with ROM elements. Input samples are written to RAM as new samples arrive, with the write address counting downward. Concurrently, weights and samples are read from the ROMs and RAMs into their respective MAC units, with read addresses counting upward. This counter-directional address counting achieves the "shift register" effect of a traditional direct-form filter. The method is demonstrated with a 256-tap FIR filter featuring a 24-bit data width and eight parallel MAC pipelines. A testbench simulation provides a noisy sine wave as input, demonstrating the filter's performance in producing a clean sine wave output, verified in the waveform diagrams.

2 Introduction

Finite Impulse Response (FIR) filters are commonly used in digital signal processing to remove unwanted frequencies from a noisy signal while preserving the desired frequency bands. They are particularly well-suited for implementation on FPGAs due to their unique balance between low latency, parallelism, and hardware flexibility. When implementing FIR filters on FPGAs, several key design constraints must be considered. Hardware constraints include the number utilization of multipliers (which are costly), adders, and registers used. Performance constraints include throughput, output latency, and placement area. Efficient design requires careful trade-offs between these constraints to meet the application's performance and resource requirements. This report presents an FIR filter that prioritizes placement area by opting for

RAM and ROM elements instead of registers, and sacrifices low-latency for higher throughput via extensive usage of parallel pipelining. At the same time however, the usage of RAM and ROM instead of registers hinders the higher throughput as memory access throughput is slower than register throughput. Thus, the main benefit of this implementation is the placement profile on the FPGA chip. A smaller placement profiles can yield fewer wire nets and less congested routing, which may also contribute to lower power usage.

3 Related Work

Related Work.

4 Data Description

Not actually sure what this is upposed to be... The filter is highly flexible can be easily reconfigured by the user via the top level parameters DATA_WIDTH, FIR_DEPTH, and PIPELINES. The parameter DATA_WIDTH defines the bit-width of the input signal, filter weights, and the output signal. The parameter FIR_DEPTH defines the number of 'taps' in the filter, or in other words, the number of multiply-accumulates (MAC) performed per input signal sample. The parameter PIPELINES defines the number of parallel hardware MAC units in the filter, or in other words, the number of 'time-slices' per input signal sample. This paper demonstrates a 256 tap FIR Filter with 8 parallel MAC pipelines on 24-bit data, that is to say, with DATA_WIDTH=24, FIR_DEPTH=256, and PIPELINES=8. With 8 parallel MAC pipelines, each MAC unit performs 256/8=32 multiply-accumulates per input signal sample. The 256 weights are generated via python scipy's firwin() function for a low-pass filter with a cutoff frequency of 2KHz. An input signal of 200Hz sine wave plus a 10KHz sine wave to represent noise is fed into the filter.

5 Method Description

Why RAM-based shift register instead of Flip-Flop-based shift register?

24-bit signals =¿ 24 registers per filter tap.

For example, 256 taps * 24-bits = 6144 registers just to implement the input shift register.

On a 7-series Xilinx FPGA, each CLB has only 8 registers.

6144 / 8 registers =¿ 768 CLBs.

On the other hand, 3 RAM or ROM elements take up the same space profile of just 5 CLBs.

8 MAC pipelines =¿ 8 RAMs and 8 ROMs =¿ 16 memory elements / 3 memory elements per BRAM sites =¿ 6 BRAM sites =¿ space profile of 30 CLBs.

6 Model Description

Model Description.

7 Experimental Procedure and Results

Experimental Procedure and Results

8 Conclusion

Conclusion

9 References

References