Politecnico di Torino
III Facoltà di Ingegneria

Laboratory 2
Digital arithmetic

Master degree in Electrical Engineering

Authors: Group 21

Nicola Dilillo, Stefano Moncalvo, Lorenzo Carrano

November 20, 2021

Many thanks to Prof. Mariagrazia Graziano for providing us with this template.

Contents

CHAPTER 1

Reference model development

1.1 Introduction

The goal of this laboratory is to design a Finite Impulse Filter filter (FIR) with a cut frequency of 2 kHz. Filter has is design according two parameter: order and number of bits. The order employ for the following filter is 10 and the number of bits are 9.

Before starting with filter design it's needed to develop a prototype version that ensure the final result of request implementation.

1.2 Design the filter with Matlab/Octave

First step is the generation of coefficients. To do this Matlab function fir1 has been used. The coefficients are shown in table 1.1.

Number	Quantize	Normalize
0	-1	1
1	-2	1
2	-4	1
3	8	0
4	35	1
5	50	1
6	35	1
7	8	1
8	-4	1
9	-2	1
10	-1	1

Table 1.1: All coefficients.

Always staying in Matlab and using the previous coefficients, a further Matlab script is executes in order to perform different simulation with prototype filter with a cut-off frequency of 2 kHz and a sampling frequency of 10 kHz. The input signal used is an average between two sinusoidal waves respectively at 500 HZ and 4.5 kHz. After this execution two files have been generated:

1. *sample.txt*, which contains the sample values that have fed the input of FIR;

2. *result.txt*, which contains the output values that has been elaborated from our FIR.

1.3 C prototype

A C program language has been written to have a fixed point implementation of FIR that use the following formula:

$$y_i = \sum_{n=0}^{10} x_{i-n} \cdot b_n$$

Thanks this program is possible to evaluate the performance of fixed version respect to Matlab execution.

1.3.1 Evaluate the THD

The purpose of this script is to evaluate the Total Harmonic Distortion (THD) trying to react a value that is maximum -30dB. If an amount of tollerance is avaiable maybe will be possible to reduce the bit numbers and so to reduce the size of FIR design.

In first hand, with 9 bits used for data, the THD is -40 dB. Trying to reduce the number of bits to 8 the value of THD obtain is still acceptable, it's -33 dB. When a further reduction has been applied the value of THD a not allowed value is returned, with 7 bits THD is -27dB.

In the end, for the final implementation of FIR 8 bits have been used in order to achieve the THD request and to reduce the area.

CHAPTER 2

VLSI implementation

2.1 Starting architecture development

The purpose of this section is to develop in VHDL the architecture of the previously designed filter. The architecture of the filter is composed by four elements:

- Adders
- Multipliers
- Flipflops
- Registers

The 8-bit input is recived and then propagated through a chain of 10 registers; the output of each register is multiplied by the corresponding coefficient, and the results are summed together to form the filter's output. All registers use VIN as an enable signal, in order to avoid unwanted propagation of data. The VIN signal, delayed of two clock cycles, is also used to drive VOUT. Every input and output signal is loaded or produced by registers or flipflops, to reduce the risk of interference from external signals.

2.2 Simulation

The design was simulated using a testbench written in both Verilog and VHDL. The testbench is composed of four disinct entities:

- **clk_gen:** generates a clock signal of the specified frequency, and a reset signal.
- **data_maker:** reads the samples.txt file and provides an input every clock cycle and its validity using the VIN signal.
- **data_sink:** recives the outputs of the filter every clock cycle and writes them in the output.txt file if VOUT is equal to 1.
- **tb_fir:** is the testbench top entity written in Verilog.

At the end of the simulation the values stored in Output.txt were compared with the ones produced by the C prototype. The two files are equal, which means that the filter is behaving correctly.

2.3 Logic synthesis

After the simulation the design must be synthetized. To estimate the maximum working clock frequency of the filter, the clock period in the design compiler is set to 0 ns. In this way the compiler optimizes the circuit as much as possible, and the negative slack of the timing report corresponds to the maximum clock frequency.

After running the synthesis at the computed frequency the area is evaluated.

It is requested to set the frequency to 25% of the maximum value. After the synthesis a new area estimation is produced.

The constraints on the clock have a significant role in the estimation of the area: by allowing the frequency to be lower, the size of the circuit will be smaller.

The Design Compiler produces the Verilog netlist of the synthetized circuit and a .sdf file containing the circuit's delays. Those files are used by Modelsim to generate

CHAPTER 3

Advanced architecture development