Fall
2024

FIR Filter Design

PROJECT 1 CE6325 VLSI DESIGN: VERILOG HDL

LAMIN JAMMEH NET-ID: DAL852207

Project Description:

A Finite Impulse Response (FIR) filter was designed with pre-determined specifications. The filter order, and the Data_in size were parameterized to allow the design to be scalable. The filter takes in a sample input, processes the sample input and passes it to an output register

Design Specifications

Filter order: 5
Filter coefficients: [7, 8, 9, 12, 4]
Data_in size: 4
Data_out size: 16

Data flow of the design

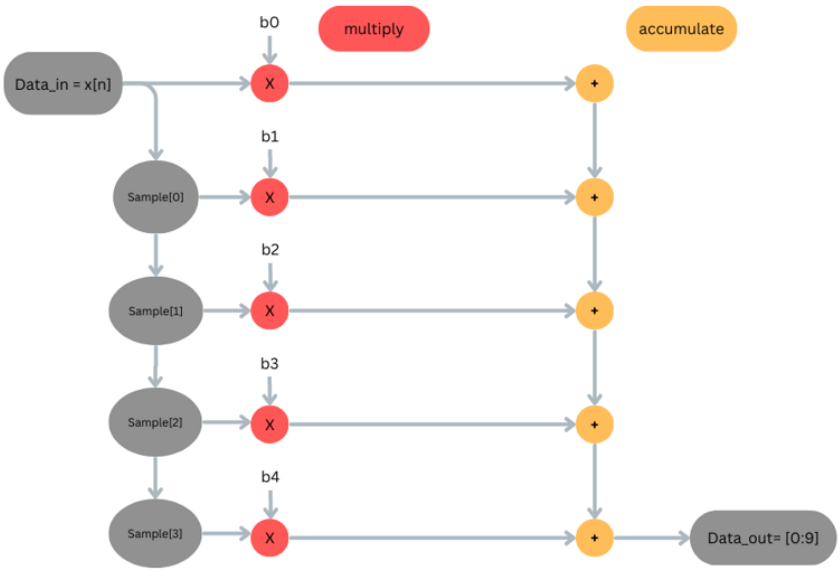

Figure1: Finite Impulse Response (FIR) Filter using Multiply and Accumulate

Design Netlist

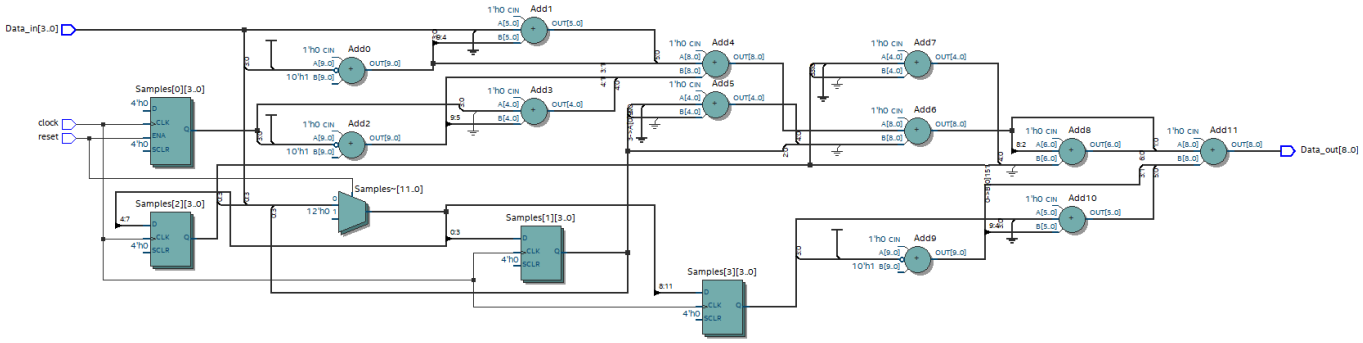

Figure 2: Netlist for the FIR filter

Testbench Process Flow

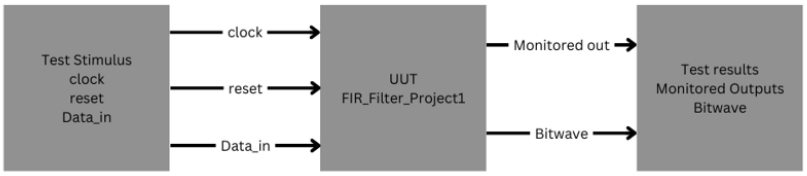

Figure 3: Shows the Testbench Process flow

Table 1 showing Filter out with when Data_in = decimal (6)

Data_in		Sample[k]	Filter Coefficient (bn)		bn * Sample[k]		Acc @ filter stage
6		6	7		42	acc0	42
	Data_in	6	8	bn * Sample[k]	48	acc1	90
		6	9		54	acc2	144
		6	12		72	acc3	216
		6	4		24	Data_out/acc4	240

Table 1 shows that Data_in will go shift through all of Sample[k] when enough time is allowed before changing the value at the input. The value in each register of Sample[k] is multiplied by the filter

coefficient (bn). The values of the multiply is accumulated or summed to form the output of the filter.

$$Mux = b_n * Sample[k]$$

$$acc = \sum_{n}^{0} Mux_n$$

Register Transistor Logic (RTL) Simulation Result

Table3 shows the observed values from the simulation results

```
# run -all
# Time:     0  | Data_in: 0  | Data_out:    x  | reset: 1
# Time:     5  | Data_in: 0  | Data_out:    0  | reset: 1
# Time:    10  | Data_in: 0  | Data_out:    0  | reset: 0
# Time:   110  | Data_in: 6  | Data_out:    0  | reset: 0
# Time:   115  | Data_in: 6  | Data_out:   42  | reset: 0
# Time:   125  | Data_in: 6  | Data_out:   90  | reset: 0
# Time:   135  | Data_in: 6  | Data_out:  144  | reset: 0
# Time:   145  | Data_in: 6  | Data_out:  216  | reset: 0
# Time:   155  | Data_in: 6  | Data_out:  240  | reset: 0
# Time:   210  | Data_in: 3  | Data_out:  240  | reset: 0
# Time:   215  | Data_in: 3  | Data_out:  219  | reset: 0
# Time:   225  | Data_in: 3  | Data_out:  195  | reset: 0
# Time:   235  | Data_in: 3  | Data_out:  168  | reset: 0
# Time:   245  | Data_in: 3  | Data_out:  132  | reset: 0
# Time:   255  | Data_in: 3  | Data_out:  120  | reset: 0
# Time:   310  | Data_in: 2  | Data_out:  120  | reset: 0
# Time:   315  | Data_in: 2  | Data_out:  113  | reset: 0
# Time:   325  | Data_in: 2  | Data_out:  105  | reset: 0
# Time:   335  | Data_in: 2  | Data_out:   96  | reset: 0
# Time:   345  | Data_in: 2  | Data_out:   84  | reset: 0
# Time:   355  | Data_in: 2  | Data_out:   80  | reset: 0
# Time:   410  | Data_in: 5  | Data_out:   80  | reset: 0
# Time:   415  | Data_in: 5  | Data_out:  101  | reset: 0
# Time:   420  | Data_in: 5  | Data_out:  101  | reset: 1
# Time:   425  | Data_in: 5  | Data_out:    0  | reset: 1
# Time:   430  | Data_in: 5  | Data_out:    0  | reset: 0
# Time:   435  | Data_in: 5  | Data_out:   35  | reset: 0
# Time:   445  | Data_in: 5  | Data_out:   75  | reset: 0
# Time:   455  | Data_in: 5  | Data_out:  120  | reset: 0
# Time:   465  | Data_in: 5  | Data_out:  180  | reset: 0
# Time:   475  | Data_in: 5  | Data_out:  200  | reset: 0
# Time:   530  | Data_in: 2  | Data_out:  200  | reset: 0
# Time:   535  | Data_in: 2  | Data_out:  179  | reset: 0
# Time:   545  | Data_in: 2  | Data_out:  155  | reset: 0
# Time:   555  | Data_in: 2  | Data_out:  128  | reset: 0
# Time:   565  | Data_in: 2  | Data_out:   92  | reset: 0
# Time:   575  | Data_in: 2  | Data_out:   80  | reset: 0
# Time:   630  | Data_in: 0  | Data_out:   80  | reset: 0
# Time:   635  | Data_in: 0  | Data_out:   66  | reset: 0
# Time:   645  | Data_in: 0  | Data_out:   50  | reset: 0
# Time:   655  | Data_in: 0  | Data_out:   32  | reset: 0
# Time:   665  | Data_in: 0  | Data_out:    8  | reset: 0
# Time:   675  | Data_in: 0  | Data_out:    0  | reset: 0
# Time:   730  | Data_in: 2  | Data_out:    0  | reset: 0
# Time:   735  | Data_in: 2  | Data_out:   14  | reset: 0
# Time:   745  | Data_in: 2  | Data_out:   30  | reset: 0
# Time:   755  | Data_in: 2  | Data_out:   48  | reset: 0
# Time:   765  | Data_in: 2  | Data_out:   72  | reset: 0
# Time:   775  | Data_in: 2  | Data_out:   80  | reset: 0
# Time:   830  | Data_in: 4  | Data_out:   80  | reset: 0
# Time:   835  | Data_in: 4  | Data_out:   94  | reset: 0
# Time:   845  | Data_in: 4  | Data_out:  110  | reset: 0
# Time:   855  | Data_in: 4  | Data_out:  128  | reset: 0
# Time:   865  | Data_in: 4  | Data_out:  152  | reset: 0
# Time:   875  | Data_in: 4  | Data_out:  160  | reset: 0
# ** Note: $stop    : C:/Users/lmnjm/OneDrive/Documents/Grad School/UTD/Fall 2024/EEDG 6325/Projects/Project 1/FIR_Filter_Project1_tb.v(66)
#    Time: 930 ps  Iteration: 0  Instance: /FIR_Filter_Project1_tb
```

When Data_in =6 Data_out=240 @ next rising clock after Data_in goes through all the filter taps or stages

@ reset = 1 Data_out = 0 since the Sample register is cleared @reset high. This happens @ next rising clock

Simulation Wave form