Assignment 3

Rick Veens Studentno: 0912292 Barry de Bruin Studentno: -
r.veens@student.tue.nl -@student.tue.nl

June 5, 2015

Lab assignment 3a

1. Requirements

The assignment is to design and implement a FIR filter named filter that:

1. Uses as little resources as possible and is maximally sequential. In particular at most 1 multiplier may be used.
2. Conforms to the 4-phase asynchronous protocol for both input and output.
3. Can run at a clock frequency of 100 Mhz.
4. Honors changes in the coefficients (after a finite delay).
5. May produce a finite length interval of start-up noise

1.1 Analysis of requirements

The FIR filter should make use of only one DSP-unit, since the internal clock frequency is significantly higher than the expected sample rate. Furthermore the coefficients should not be buffered but must be connected as wires instead of buffering them in registers. This ensures that requirement 1 and 4 can be satisfied. For requirement 3, the clock frequency should be at least 100MHz, which will be checked in the post-synthesis report. Lastly, the asynchronous ack/req protocol will be used since there needs to be some synchronization between the testbench (44.1KHz sample rate) and the filter (100MHz sample rate).

2. System architecture

Figure one shows the global architecture of the FIR filter.

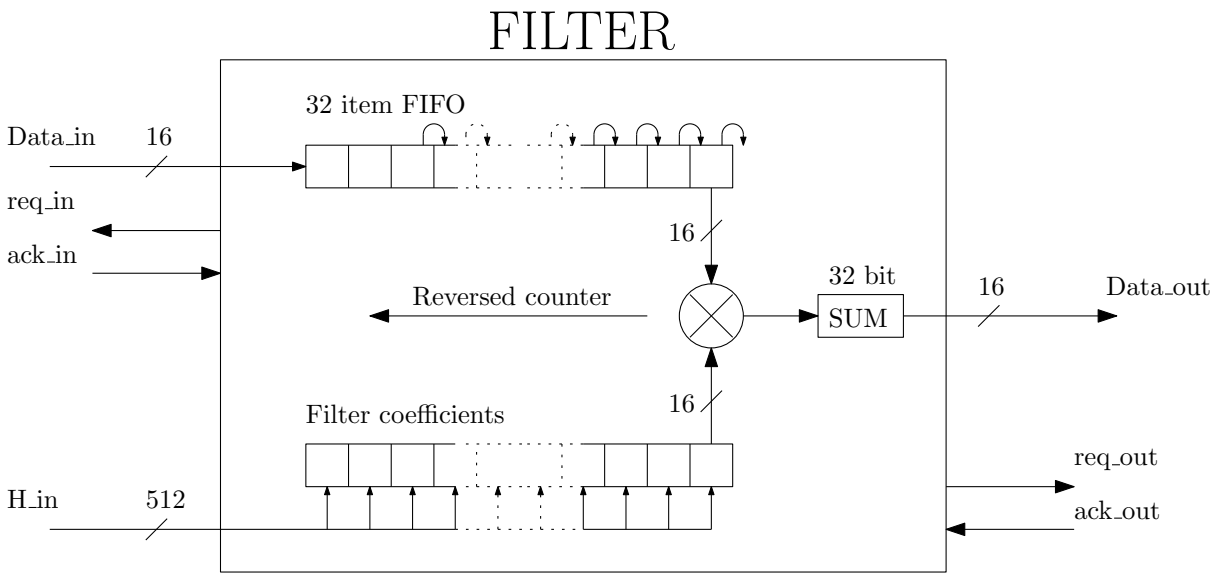

Figure 1: System overview

Unlike the FIR filter of lab2, the filter has only one tap. After each clock edge the multiplier/accumulator will multiply a sample that is stored in the memory with it's corresponding filter coefficient. An index makes sure that this operation moves through the FIFO in 32 clock cycles. The index sample is also shifted one place to the right. After 32 cycles the last tap has been calculated and the first place in the FIFO is empty. The system will output the calculated sample, and request a new data item, which will be put on the first index.

The whole procedure will then restart.

4. Design choices

5. Functional correctness

all requirements?

6. Resource usage

Table 1 summarizes the resources used by the FIR filter. This list is obtained by running the synthesis step in Xilinx ISE and extracted from the *Summary* and *Device Utilization* report.

Resource	**Available**	**Utilized**	**Percentage utilized**
Flip Flops	54576	1064	1%
Slice LUTs	27288	527	1%
DSP48A1s	58	1	1%
BRAM	116	0	0%
Bonded IOBs	218	40	18%

Table 1: General resource usage overview

The number of flipflops used in the design are:

$$\begin{aligned} Total_{\#reg} &= sum + data + coef + cnt + state + flowcontrol \\ &= 32 + 16 \cdot 32 + 32 \cdot 16 + 5 + 1 + 2 \\ &= 1064 \end{aligned}$$

This number matches the synthesis result.

There is only one DSP unit used, since the filter is fully sequential. This results in a significant resource reduction since the DSP units are quite expensive. According to the summary report, almost all slice LUTs are used for logical ports. They are used to model the 5 bit counter that counts up to 32. The number of Bonded IOB's represents the number of pins used on the FPGA. These pins are occupied by the clk, rst, h_in, h_enabled, $2 \cdot 16$ input and output pins and 2 pairs of ack/req pins.

Furthermore some additional resources are used to enable the possibility to shift the data in the FIFO every clock cycle.

7. System throughput and latency

The minimum sample time estimation is extracted from the synthesis report under *Timing Report*.

Minimum period: 8.421ns (Maximum Frequency: 118.750MHz)

This time is equal to the largest critical path (the calculation of the tap is the critical path in this design).
(min) sample time Ts and (max) sample frequency fs, both after synthesis and after placement & routing.

8. Simulation results

Time en FFT plot

Lab assignment 3b

1. Requirements

The specific requirements for the strength reduced FIR filter are:

1. The design may use at most 3 multipliers.
2. Conforms to the 4-phase asynchronous handshake protocol for both input and output.
3. Can run at a clock frequency of 100 Mhz.
4. Should have about 4 times the sample frequency as the sequential implementation.
5. Honors changes in the coefficients (after a finite delay).
6. May produce a finite length interval of start-up noise.

Comparison between two filters

In your final report, compare the output signals of the strength reduced filter with the output of the sequential filter, by generating a difference signal for a representative subset of samples. Make sure you align the outputs correctly, because the amount of start-up noise of the 2 implementations may differ. If the difference signal is non-zero, explain the differences. Further reporting guidelines are on the courses website.

Appendix

all verilog code