DSP in VLSI

Homework 2

Digital Filter

I. Purpose

In this homework., we will learn to implement digital filter and meet the high throughput requirement using parallel processing or pipeline techniques.

II. Related Algorithm

Digital filter is an indispensable component in signal processing and communication systems. It can be classified into two categories, i.e. infinite impulse response (IIR) filter and finite impulse response (FIR) filter. The FIR filter is stable and has linear phase. Thus, it is widely used in various applications. The IIR filter has the feedback path, which results in good efficiency to obtain excellent frequency response with fewer arithmetic components.

Assume that the length of the FIR filter is L. The coefficient is denoted as $h[i]$, for $i = 0,1, \dots, L-1$. The Z-transform of FIR filter is represented by

$$H(z) = \sum_{i=0}^{L-1} h[i] z^{-i} \quad (1)$$

The impulse response of an FIR filter is its coefficients, i.e. the filter output signal is the convolution of the filter input signal and filter coefficients. If the input signal is denoted as $x[n]$ and output signal is denoted as $y[n]$, then

$$y[n] = \sum_{i=0}^{L-1} h[i] x[n-i]. \quad (2)$$

In other words,

$$y[n] = h[0]x[n] + h[1]x[n-1] + h[2]x[n-2] + h[L-1]x[n-L+1]. \quad (3)$$

From Eq. (3), we know that every output $y[n]$ needs the current and past inputs $x[n] \sim x[n-L+1]$ to complete the computation. Thus, storage of length L is required.

Generally speaking, the FIR filter can be implemented in two ways, namely direct form and transpose form. Direct form FIR filter is given in Fig. 1 based on Eq. (2). In the direct form architecture, there exists a long critical path consisting of a multiplier and several adders. Hence, it is not suitable for high-speed operation. However, all the register word-lengths are the same as that of the input.

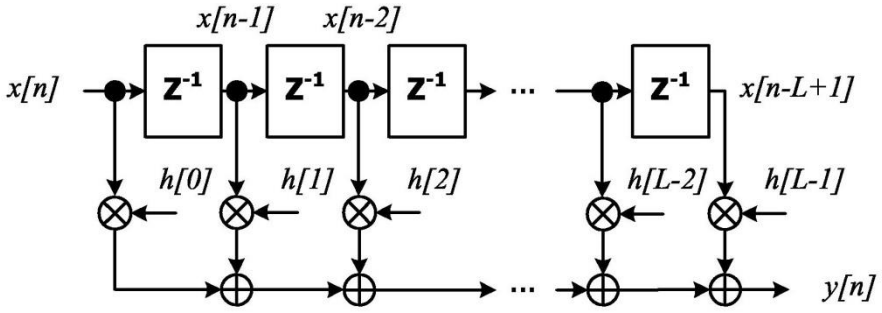

Fig. 1 FIR filter in direct form.

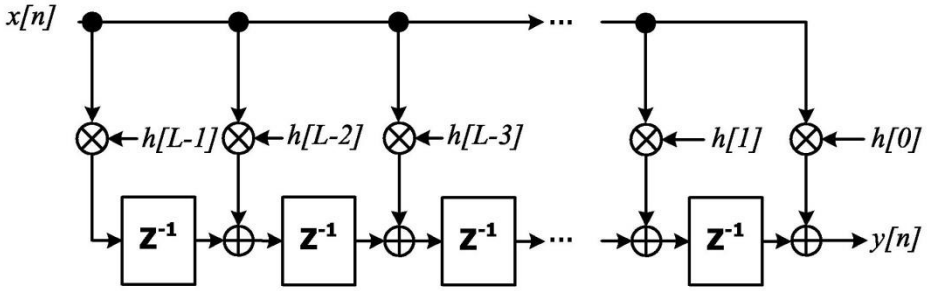

Fig. 2 FIR filter in transposed form.

We can also implement the FIR filter by the transposed form architecture, as shown in Fig. 2. In this architecture, the critical path is reduced to one multiplier and one adder. Thus, it is suitable for high-speed operation. However, if higher throughputs are required, we may need pipeline or parallel processing techniques to accelerate computation.

In addition, quantization is an important issue in filter design. If longer word-lengths are used, the better filter performance can be achieved. However, the hardware cost increases for larger multipliers or adders. If shorter wordlengths are used, performance degradation may not be acceptable. Hence, we need to evaluate the performance to decide the proper word-length.

III. Function Description

1. Filter coefficients

   Given a filter with the following continuous-time impulse response is given as follows.

$$h_p(t)=\frac{1}{\sqrt{T}}\left\{(1-\alpha)\,\mathrm{sinc}\left(\frac{(1-\alpha)t}{T}\right)+\alpha\left[\mathrm{sinc}\left(\frac{\alpha t}{T}+\frac{1}{4}\right)\cos\left(\frac{\pi t}{T}+\frac{\pi}{4}\right)+\mathrm{sinc}\left(\frac{\alpha t}{T}-\frac{1}{4}\right)\cos\left(\frac{\pi t}{T}-\frac{\pi}{4}\right)\right]\right\}$$

Insert $t = nT_s$ and $T = 4T_s = 1$. Namely, 4 points are sampled in one symbol period. We will implement a $17(= 4 \times 4 + 1)$-tap causal FIR filter with $\alpha = 0.5$. Note that you need to acquire the filter coefficient for $-2T \leq nT_s \leq 2T$ and make it causal (impulse response on the right half plane).

2. Evaluate performance

   Draw the frequency domain response of the filter including phase and magnitude. Please indicate the proper unit for X-axis. For magnitude response, please express it in dB and adjust the range of Y-axis to be 60dB so that we can know the effect of attention and the 3dB bandwidth.

   Feed the input signal given as follows into the filter

   $$x[n] = \cos\left(-\frac{2\pi n}{64}\right) + \sin\left(\frac{2\pi n}{3}\right) \qquad \text{for } 0 \leq n \leq 128.$$

   Observe the output time-domain waveform.

3. Quantization

   Please use transposed form to implement 17-tap square-root raised-cosine FIR filter. Decide the word-lengths of each data path after additions and multiplications **based on the simulation results from Matlab/Python** so that the RMSE error between the fixed-point output and the floating-point output is less than 2^{-a}. If $p = 0, a = 11$. Otherwise, $a = 10$, where $p = mod(d_1, 2)$ and d_1 is the last digit of your student ID.

4. Implementation

   Please implement the transposed form using the word-lengths that you decide in step 3. Note that modularity and parametric design can ease your loading. Use $x[n]$ as the input. Please add D flip-flops in the input port $x[n]$ and output port $y[n]$. Check the behavior simulation results.

   Find out the critical path of your design. Show the numbers of adders and multipliers in the critical path and list the timing information from the report of max delay. Set proper timing constraint to achieve the maximal operating frequency in this case as shown in Fig. 3. Check the post-**SYNTHESIS** simulation results to ensure its correctness. Compare both the behavior simulation and post-**SYNTHSIS** simulation results with the Matlab floating-point results. Draw the errors.

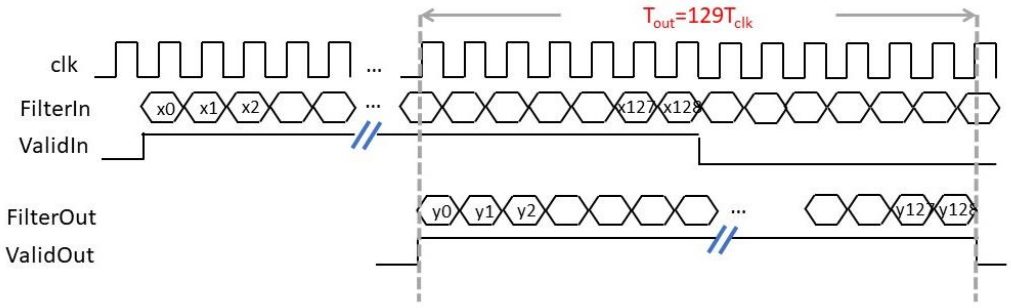

Fig. 3 Post **SYNTHESIS** timing diagram before speedup

5. Acceleration using pipeline or parallel processing

   Now, we want to achieve a 2X speedup. Use either the pipeline or parallel processing technique to accelerate your throughput. Set proper operating frequency if you choose to use pipeline technique or proper input/output style (FilterIn1, FilterIn2 v.s. FilterOut1, FilterOut2) if you choose to use 2-level parallel processing. **You have to generate $y[0]$ to y[128] within $T_{out}/2$**. Evaluate the performance to examine the correctness of functionality and successful speedup. Compare the area report (for cell based design flow) or the resource utilization (for FPGA design flow) before speedup and after speedup.

IV. Required Results

1. (Step 1) Please use Matlab/Python to draw the **impulse response** and **frequency response** (containing magnitude and phase) of the 17-tap square-root raised-cosine FIR filter. Note that you need to use dB scale for the magnitude of the frequency response and use radian for the phase of frequency response versus normalized frequency. The x-axis must be marked with the correct label (20%). Please explain whether the filter is high-pass, band-pass or low-pass and why. (5%)
2. (Step 2) Draw the time-domain input and output waveforms after you feed 128-point input (20%)
3. (Step 3) To show how you determine the word-length, please use the word-length of coefficients as the X-axis and error as the Y-axis. Scan the quantization error versus the word-lengths for at least 6 settings**.** (Four figures each having at least 6 word-length settings in its x-axis). According the following simulation results, mark the final word-length settings in the block diagram of the transposed form FIR filter. (20%)
   a. Output error versus input word-lengths
   b. Output error versus coefficient word-lengths

c. Output error versus word-lengths after multiplication
d. Output error versus word-lengths after addition

4. (Step 3) Show the fixed-point output of time domain signal. (10%) Compare to the floating-point output and draw the error. (10%) Show the frequency response of the fixed-point direct-form FIR filter.(10%) Compare to the frequency response of floating-point results and draw the error of the magnitude response in passband (within 3dB bandwidth), expressed in dB (5%)
5. (Step 4) Print out the behavior timing diagram for the first 20 samples and the last 20 samples (10%). Show the errors of hardware outputs and Matlab floating-point results of transposed from FIR versus for $0 \leq n \leq 128 + \boldsymbol{d}$ by figures, where $\boldsymbol{d}$ **is the latency of your design and is not constrained, but you have to observe all the outputs until output becomes 0. (10%)**
6. (Step 4) Paste the timing report about the critical path. Show the numbers of adders and multipliers in the critical path. (5%). List your timing constraint to achieve the highest operating frequency (5%). Print out the post-**SYNTHESIS** timing diagram for the first 20 samples and the last 20 samples (10%). and show the errors of hardware outputs and Matlab floating-point results of transposed from FIR versus for $0 \leq n \leq 128 + \boldsymbol{d}$ (10%).
7. (Step 5) Show that $\boldsymbol{y[0]}$ **to y[128] can be generated within** $\boldsymbol{T_{out}/2}$ in the timing diagram. Explain the way that you use to accelerate the throughput and draw the block diagram and indicate your critical path. (15%). Show your synthesis report of max delay to verify your explanations. (5%)
8. (Step 5) Show the post-**SYNTHESIS** simulation results for the first 20 samples and the last 20 samples and indicate why your throughput is doubled (10%). Show the errors of hardware outputs and Matlab floating-point results of transposed from FIR versus for $0 \leq n \leq 128 + \boldsymbol{d}$ by figure. (10%) Show the area (resources) before and after speedup from the synthesis report. (10%)
9. Please upload your report containing the required results and your Verilog codes as well as Matlab/Python codes to NTUCool.