DSP in VLSI

Homework 3

Interpolator

I. Purpose

In this lab., we will learn the method to implement a digital interpolator and to realize the design of customized floating-point datapath.

II. Related Algorithm

Digital interpolator is a module that is commonly seen in applications that needs to resample the discrete inputs. For example, in wirelined or wireless communication systems, the analog signal will be converted to digital signal by analog-to-digital converter (ADC) driven by sampling frequency generated from an oscillator. Because the oscillators at the transmitter and the receiver may have tiny frequency and phase deviation, sampling clock offset exists as shown in Fig. 1.

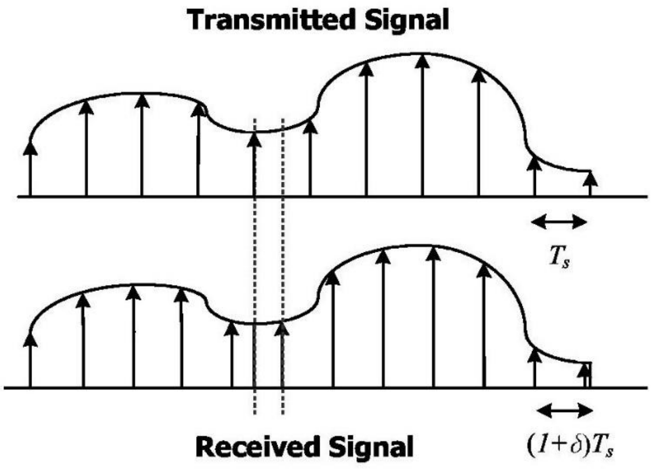

Fig. 1 Sampling clock offset

To solve the problem of sampling clock offset, we need an interpolator to generate samples with fractional delay to restore the samples that we want. According to the sampling theorem, we know that an ideal interpolator has impulse response described by $sinc(t) = \frac{sin(\frac{\pi t}{T_s})}{\frac{\pi t}{T_s}}$ in the time domain as shown in Fig. 2. In the frequency domain, the frequency response has a rectangular shape.

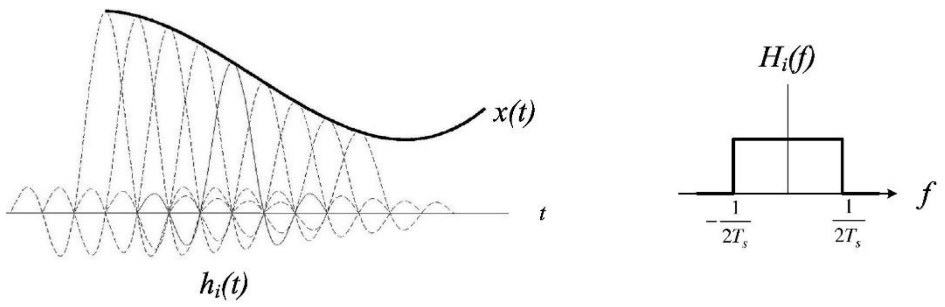

Fig. 2 Ideal interpolator.

However, it is infeasible to implement an ideal interpolator. One of the reasons is that it has an infinite length. In addition, it is not a causal filter because it contains impulse response in the left half plane. Even so, the ideal interpolator reveals the properties of an interpolator.

1. The frequency response must keep constant magnitude in the range regarding to signal bandwidth.
2. The frequency response must have linear phase in the range regarding to signal bandwidth.

There are various kinds of digital interpolators which can be seen as an alternative of ideal interpolator. Polynomial-based digital interpolators are widely used because of the following advantages.

1. The polynomial-based interpolators have closed form to describe the required interpolator coefficients.
2. The polynomial-based interpolators have good frequency response in the frequency domain.
3. The polynomial-based interpolators are easily to be implemented.

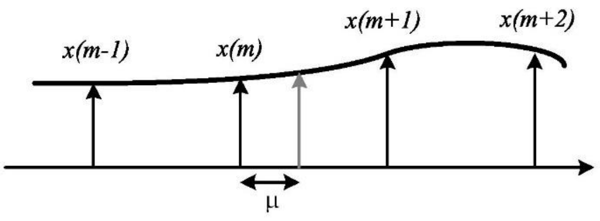

Fig. 3 Definition of sample point with fractional delay.

A. Linear interpolator

Linear interpolator is the simplest interpolator that is described by the first-order polynomial. Consider the point $x(m+\mu)$ to be interpolated in Fig. 3, where $0 \le \mu < 1$. By linear interpolation,

$$x(m+\mu) = \mu x(m+1) + (1-\mu)x(m). \tag{1}$$

B. Second-order polynomial interpolator

Linear interpolator may be too simple in some situation. To improve the quality after interpolation, we can raise the order of polynomail and the second-order interpolation can be used. Three adjacent sample points are adopted for interpolation. It uses the following equation.

$$x(m+\mu) = C_0 x(m) + C_{-1} x(m+1) + C_{-2} x(m+2) \tag{2}$$

and

$$\begin{cases} C_0 = (1-\mu)(2-\mu)/2 \\ C_{-1} = \mu(2-\mu) \\ C_{-2} = -\mu(1-\mu)/2 \end{cases} \tag{3}$$

C. Piecewise parabolic interpolator

We can also use piecewise parabolic interpolator to generate the the interpolation results. Its equation is given as follows. Four sample points are used for interpolation.

$$x(m+\mu) = C_1 x(m-1) + C_0 x(m) + C_{-1} x(m+1) + C_{-2} x(m+2), \tag{4}$$

where

$$, \quad \begin{cases} C_1 = -\alpha\mu + \alpha\mu^2 \\ C_0 = 1 + (\alpha-1)\mu - \alpha\mu^2 \\ C_{-1} = (\alpha+1)\mu - \alpha\mu^2 \\ C_{-2} = -\alpha\mu + \alpha\mu^2 \end{cases} \tag{5}$$

and α is a value between 0 and 1. In this lab., we set $\alpha = 0.5$.

D. Implementation of interpolator based on Farrow structure

Farrow structure is an efficient implementation method to realize the interpolator because of hardware sharing. The equation of interpolation can be rewritten as follows.

$$\begin{aligned} x(m+\mu) &= \sum_{i=I_1}^{I_2} x(m-i)C_i = \sum_{i=I_1}^{I_2} x(m-i) \sum_{l=0}^{N} b_l(i)\mu^l \\ &= \sum_{l=0}^{N} \mu^l \sum_{i=I_1}^{I_2} b_l(i)x(m-i) = \sum_{l=0}^{N} \mu^l \, v(l) \end{aligned} \tag{6}$$

where

$$v(l) = \sum_{i=I_1}^{I_2} b_l(i)x(m-i).$$

The piece-wise parabolic polynomial interpolator of Farrow structure is shown in Fig. 4.

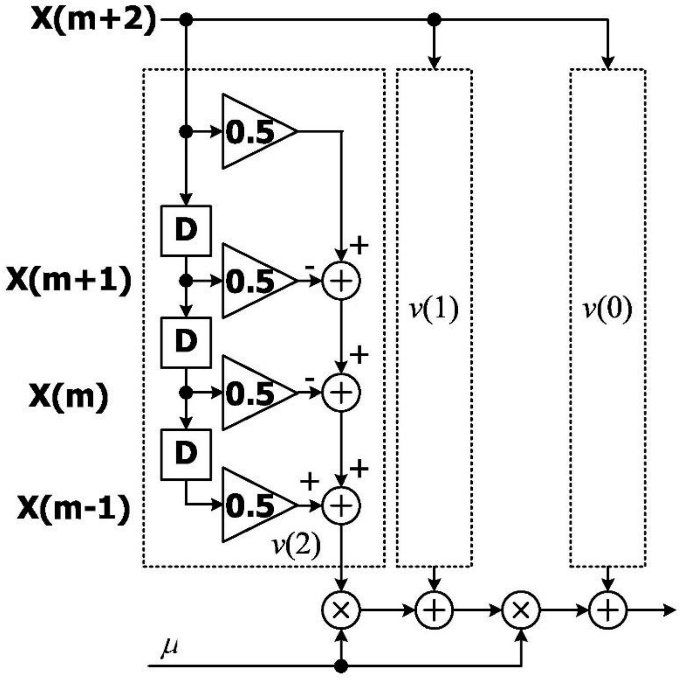

Fig. 4 Farrow structure of the piece-wise parabolic interpolator.

III. Function Descriptions (Steps)

1. Interpolator function verification in over-sampled cases.

Assume that $x_1[m] = \cos\left(2\pi\left(\frac{mT_s}{8T_s} + \frac{\phi}{5}\right)\right)$, where $\phi = \mathrm{mod}(d, 5)$ and d is the last digit of your student ID. It means that the sampling frequency is 8 times the frequency of the sinusoidal wave. Please use linear interpolator, second-order polynomial interpolator, and piecewise parabolic interpolator to interpolate the sampled waveform in the region of $16 \le m \le 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \dots \frac{8}{9}$. Compare the error between the interpolated outputs and the true results calculated by the equation.

2. Interpolator function verification in typical cases with smaller amplitude.

Assume that $x_2[m] = 2^{-15}\cos\left(2\pi\left(\frac{2mT_s}{5T_s} + \frac{\phi}{5}\right)\right)$, where ϕ is the same parameter according to your student ID. It means that the sampling frequency is 2.5 times the frequency of the sinusoidal wave. Please use linear interpolator, second-order polynomial interpolator, and piecewise parabolic interpolator to interpolate the sampled waveform in the region of $5 \leq m \leq 10$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$ Compare the error between the interpolated output and the true value.

3. **Construct bit-true model**

   Please use TF32 format to realize addition and multiplications. Write the bit-true model (simulated program in Matlab/Python) to realize TF32 additions and multiplication. Note that for simplicity, you don't need to handle NAN, infinity, subnormal number, which means if the magnitude is smaller than the smallest normal number, please use rounding to decide if it is 0 or not.

4. **Assess the performance degradation caused by finite precision.**

   Please analyze the error of piecewise parabolic interpolator if its arithmetic operations are changed to TF32 by using bit true model.

5. **Hardware reduction**

   According to Fig. 4, draw the entire block diagram of the original piecewise parabolic interpolator. Then, observe the block diagram to see if any reduction can be achieved. List at least two design approaches related with hardware sharing for saving any of D flop-flops, multipliers, adders. Draw the revised block diagram that you want to implement.

6. **Implementation**

   Please write Verilog to implement your piece-wise parabolic interpolator of Farrow structure after improvement. Please insert D flip-flops of your inputs and outputs. Check the simulation results with the following inputs in the timing diagram. Synthesis your design and check the timing report.

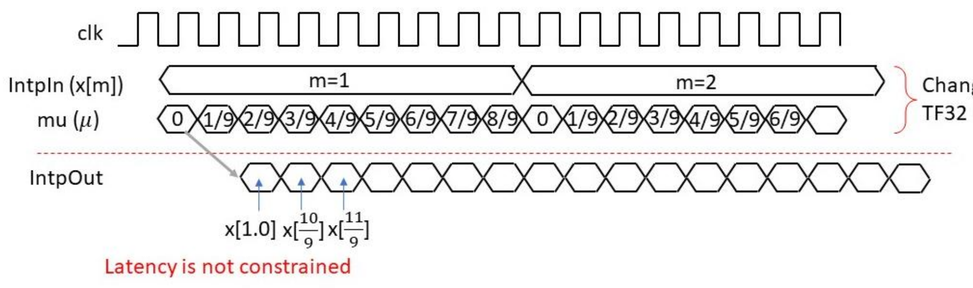

Fig. 5 Timing diagram

IV. Required Results

1. (Step 1) Show the output waveform after interpolation using linear interpolator, second-order polynomial interpolator, and piecewise parabolic interpolator to interpolate the sampled waveform in the region of $16 \leq m \leq 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. In addition, draw the error in the region of $16 \leq m \leq 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. (30%)
2. (Step 2) Show the output waveform after interpolation using linear interpolator, second-order polynomial interpolator, and piecewise parabolic interpolator to interpolate the sampled waveform in the region of $5 \leq m \leq 10$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. In addition, draw the error in the region of $5 \leq m \leq 10$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. (30%)
3. (Step 3) Please show the results that are calculated by your bit-true model for the following operands and operators. Express results both in decimal and binary representation. (S, E, F mean the sign bit, exponent field, and fraction field. All are given in binary.) (20%)

	Operand 1	Operand 2	Operation
(a)	S: 1 E: 1000_0011 F: 0000_0001_11	S: 0 E: 1000_0011 F: 0000_0001_01	Addition
(b)	S: 1 E: 1000_0011 F: 0000_0001_11	S: 1 E: 1001_0011 F: 0000_0001_01	Addition
(c)	S: 1 E: 0010_0011 F: 1100_0001_11	S: 0 E: 1000_0011 F: 1111_0001_01	Multiplication
(d)	S: 0 E: 0110_0011 F: 0011_1101_11	S: 0 E: 1000_0011 F: 0011_0011_11	Multiplication

4. (Step 4) Show the interpolation differences of the piecewise parabolic interpolator using TF32 and double-precision (default precision of Matlab) arithmetic operations. Draw the figures indicating difference of interpolated results caused by

data format for inputs $x_1[m]$ in the region of $16 \leq m \leq 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. and $x_2[m]$ in the region of $5 \leq m \leq 10$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. (20%)

5. (Step 5) Draw the complete block diagram of your first version implementation according to Fig. 4. (5%) List two design approaches of hardware sharing (The precision of the datapath must be TF32 and can not be changed.) (10%) Draw the block diagram after your improvement. (5%) Indicate the critical path (5%)

6. (Step 6) Write HDL to implement your piece-wise parabolic interpolator of Farrow structure after reduction. Please note that your input to be interpolated will change every 9 clock cycles and your μ value will change every clock cycle. Show the first 9 ($\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$) and the last 9 ($\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$) interpolated outputs of $x_1[m]$ in the region of $16 \leq m \leq 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$ and $x_2[m]$ in the region of $5 \leq m \leq 10$ in the timing diagram of behavior and post-synthesis simulations. (40%)

7. (Step 6) Show your timing report to verify the critical path of your block diagram in Q5. (10%) List your timing constraint for post-synthesis simulation. (5%)

8. (Step 6) Change your bit-true model to the new architecture after your improvement. Also depict all the errors of interpolated outputs of of $x_1[m]$ in the region of $16 \leq m \leq 32$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$ and $x_2[m]$ in the region of $5 \leq m \leq 10$ with $\mu = 0, \frac{1}{9}, \frac{2}{9}, \ldots \frac{8}{9}$. between the Verilog outputs and bit-true model (20%)

V. Reference

[1] L. Erup, F. M. Gardner and R. A. Harris, "Interpolation in digital modems. II. Implementation and performance," IEEE transactions on Communications, Vol. 41, Jun. 1993, PP. 998-1008.