Chapter 2

Switched-Current FIR Filters

An overview of the switched-current (SI) FIR filter architecture will be presented in this chapter and the architecture will be described. Before presenting the circuit design of the FIR filter, it is helpful to review some basic concepts of FIR filters in general.

2.1 FIR Filters: Basic Theory

A Finite-Impulse-Response (FIR) filter, by definition, has an impulse response that is finite in length, that is, the response is zero outside a finite interval. The impulse response can be written as a finite sum of shifted unit impulses as follows.

$$h[n] = \sum_{k=0}^{M} h_k \delta[n-k] = \begin{cases} h_k, & 0 \leq k \leq M, \\ 0, & \text{otherwise.} \end{cases} \tag{2.1}$$

The system behavior can also be described as an input-output relationship. The corresponding difference equation for (2.1), relating the input to the output, is given by

$$y[n] = \sum_{k=0}^{M} h_k x[n-k] \tag{2.2}$$

For an FIR system, either (2.1) or (2.2) can be used to describe the response of the filter. To envision the operation and structure of an FIR filter, a signal-flow graph depicted in Fig. 2.1, gives a clearer representation. Fig. 2.1 shows

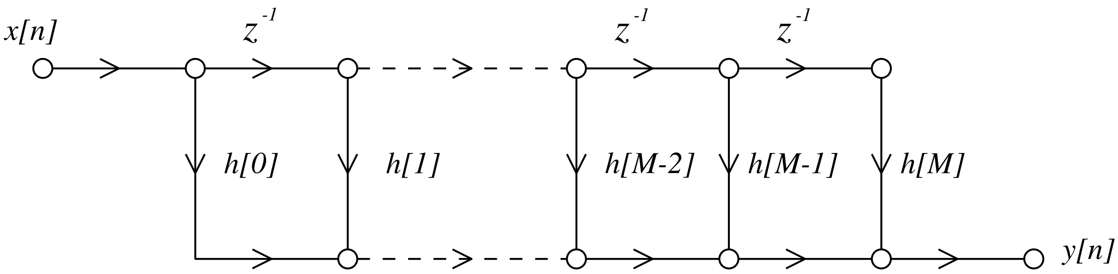


Figure 2.1 Signal-flow graph of an FIR filter in direct-form-1

a direct-form-1 representation of a typical FIR filter. The input signal $x[n]$ at each tap is weighted by the appropriate coefficient; the sum of all these weighted inputs give the output $y[n]$. Therefore, the output is just a weighted average of the input, where the tap coefficients are the corresponding impulse response values. This structure is also known as a *tapped delay-line* or a *transversal filter* structure. The name derives from the chain of the delay elements, z^{-1}, shown across the top of the filter structure of Fig. 2.1.

Linear-Phase In most data-communication systems, where transmitted pulses represent data values, it is important to maintain constant group-delay, or equivalently, linear-phase, in the communication channel, otherwise the pulse shapes will be distorted.[1] Hence, a linear phase FIR filter is highly

[1]This phase-distortion phenomenon can be pictured as waves crashing on a beach. The rounded wave-front, which travels in deep water gets distorted near the beach, producing first a flat face and then crashing over in a tube shape. The reason for the change in shape of the wave-front is nonlinear phase shifts. The speed of the wave is proportional to the

desirable and can be achieved easily by making the impulse response have even symmetry about the midpoint.

2.2 Overview of SI-FIR Filter Architecture

Traditionally, analog tapped-delay-line filters have been realized as shown in Fig. 2.2, where an input signal is sampled N-times for an N-tap filter. Problems with this type of filter are that multiple re-sampling of the signal degrades performance. A significant increase in performance can be gained if the signal does not have to be re-sampled, as in the proposed architecture of Fig. 2.3.

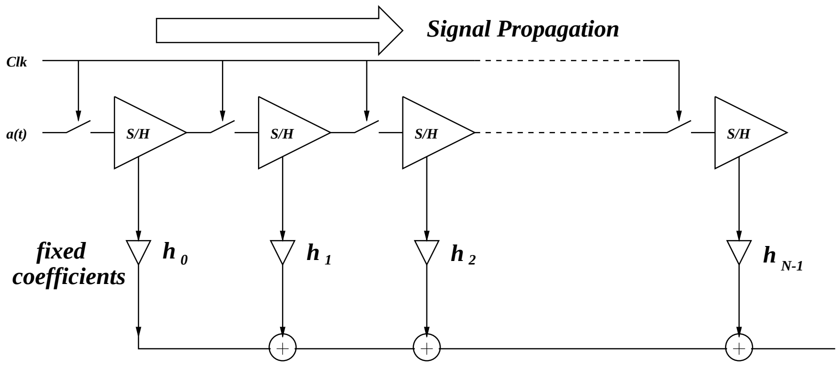


Figure 2.2 Standard analog tapped-delay-line FIR filter, where the signal is re-sampled N-times for an N-tap filter

Rotating Coefficients In this filter, the S/H circuits sample only once every N clock cycles and hold the signal value for the remaining $N-1$ cycles. In order to have the coefficient h_0 be associated with the most recent

depth of the water, therefore the top of the wave travels faster than the bottom. This difference is not important in deep water, but becomes significant in shallow water and therefore, the wave-front changes shape as it approaches the beach.

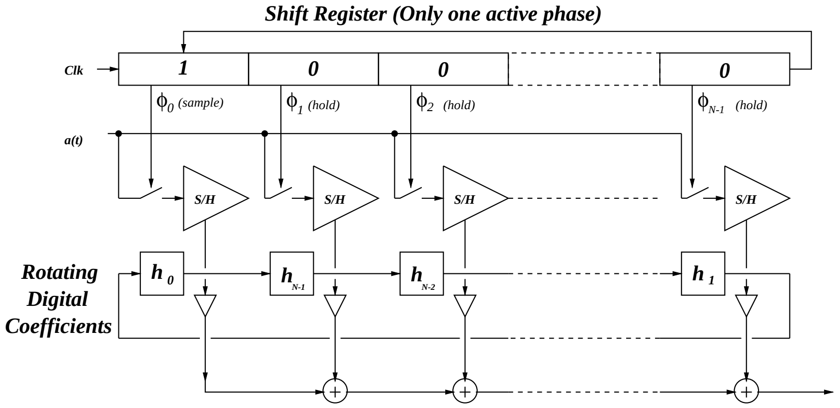


Figure 2.3 Alternative analog FIR filter, where signal is sampled once and held for N-cycles for an N-tap filter

sample, the coefficients must rotate one position for each clock cycle. However, it is much easier to rotate digital coefficients, which can be delayed and resampled using a shift-register without loss of accuracy, than to re-sample the sensitive analog signal. A similar approach was used by Xu in a 9-tap equalizer design [16], where a rotating switch matrix was used to connect stationary S/H outputs to stationary coefficients. However in the circuit of [16] the filter taps used analog control of g_m cells to change coefficient values. The circuit of Xu requires a set of analog voltages to set coefficient values and, therefore, lacks ease of programmability.

2.3 Realization of SI-FIR Filter Using Current Memory Cells and Current-Division

A block diagram of the actual SI-FIR filter realization is shown in Fig. 2.4. A transconductor acts as the front-end interface circuit and converts the analog differential input voltage signal to differential currents. The transconductance is fixed by an external resistor. The signal current is then sampled and

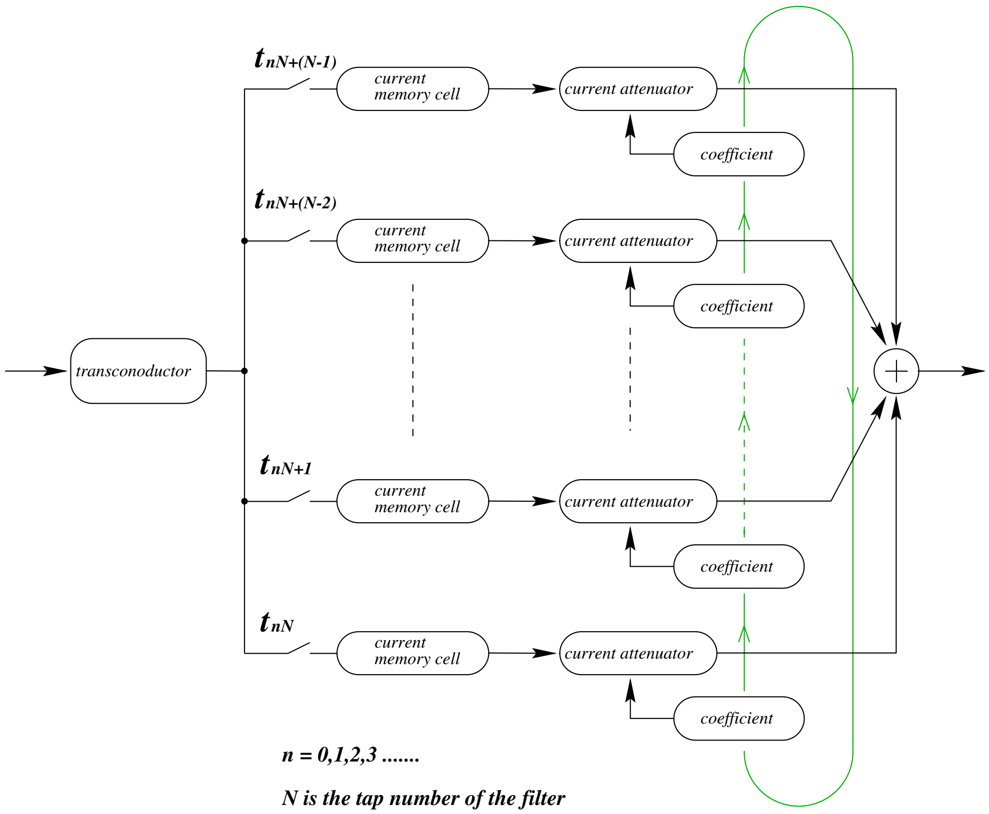


Figure 2.4 Architecture of the proposed FIR filter

held by class-AB current-memory cells [17, 18, 19]. These current signals, unlike in the conventional CCD FIR filter, are kept held within the current memory cells and are not propagated through the taps. Hence, the loss of signal is solely determined by the leakage of the memory cells rather than by the propagation loss. As the sampled current is held, the coefficients have to advance by one tap for each clock-cycle. This is equivalent to propagating the held signal transversally with the coefficients kept fixed at each tap.

Filter-Coefficient Implementation Using Current-Division The sampled currents are then passed to the current-division ladders, the attenuation of which is determined by coefficients which are stored in digital form in static D flip-flops. Finally, the output currents from the individual taps are added directly without extra circuitry. This sum of the weighted currents from all individual taps in any one clock period is the output of the filter.

2.3.1 Illustration of Filter Operation

To illustrate the operation of the filter in a pictorial way, Fig. 2.5 shows the internal signals at all the filter taps at various time intervals. The rows represent the current-memory-cell taps across the filter, and the columns represent the signal in a particular tap at different times.

Coefficient Programming Mode In Fig. 2.5, only the first N time intervals are shown — this is the programming period. Initially, the filter is in the reset state; all the coefficients are reset to zero. At time $t = 0$, the filter is set to programming mode. On the first half-cycle, the first coefficient shifts in, while the input signal is being sampled. Then on the second half-cycle, at time $t = T/2$, the signal is held by the current memory cells in tap 1. At time T, the second coefficient shifts in. The first tap is in the hold-state, while the current memory cell in the second tap is sampling the input signal. Simultaneously, the first coefficient synchronously shifts to the second tap. This process continues until the first N coefficients are programmed into the filter. Valid filter outputs are available on the second half-cycle of each clock period throughout the programming mode.

Fig. 2.5(a) shows the *static* property of the *signals* within the filter — once the signal is sampled, it is held for the next $N - 1$ clock cycles, while Fig. 2.5(b) shows the *dynamic* property of the *coefficients* — coefficients will rotate one position on each clock cycle. To derive the output from these *held signals in current memory cells*

time \ taps	1	2	3	...	N - 1	N
NT	x(T)	x(2T)	x(3T)	...	x((N-1)T)	x(NT)
(N -1)T	x(T)	x(2T)	x(3T)	...	x((N-1)T)	0
(N - 2)T	x(T)	x(2T)	x(3T)	...	0	0
(N - 3)T	x(T)	x(2T)	x(3T)	...	0	0
(N - 4)T	x(T)	x(2T)	x(3T)	...	0	0
:	:	:	:	:	:	:
3T	x(T)	x(2T)	x(3T)	...	0	0
2T	x(T)	x(2T)	0	...	0	0
T	x(T)	0	0	...	0	0
0	0	0	0	...	0	0

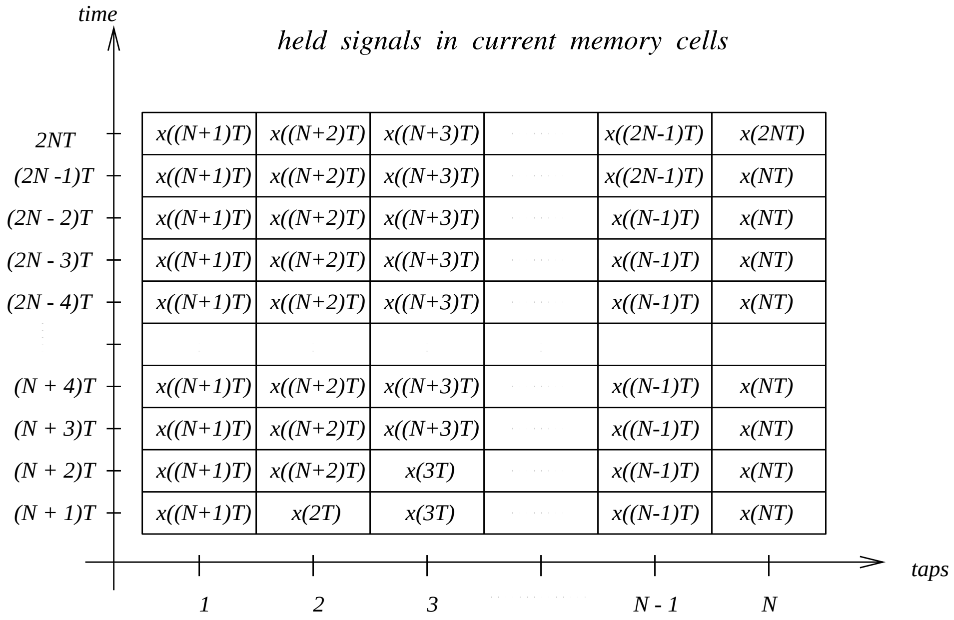
(a) Signals in the current-memory cells within the first N time periods

coefficients at each filter tap

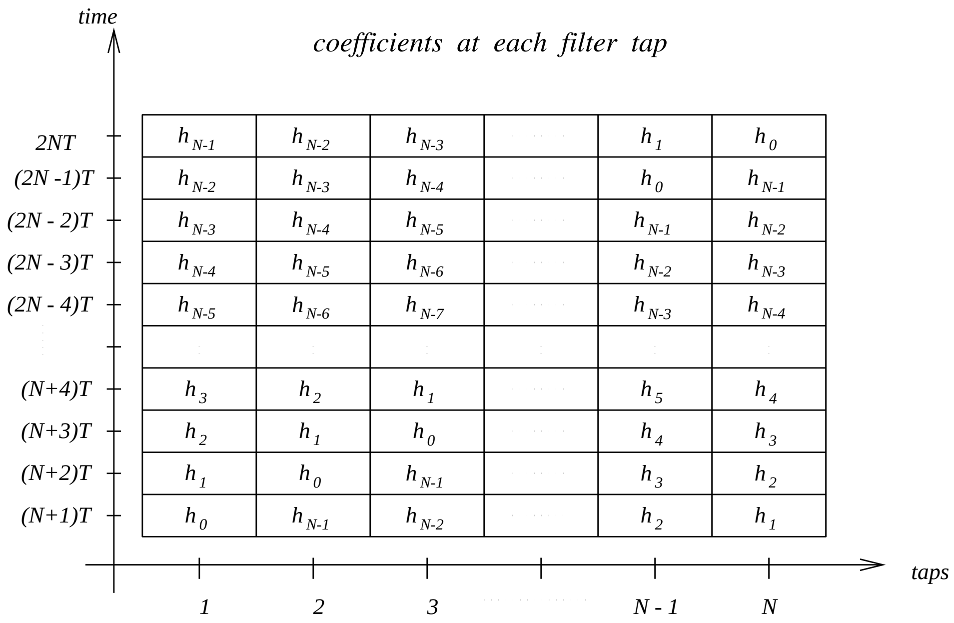
time \ taps	1	2	3	...	N - 1	N
NT	h_{N-1}	h_{N-2}	h_{N-3}	...	h_1	h_0
(N -1)T	h_{N-2}	h_{N-3}	h_{N-4}	...	h_0	0
(N - 2)T	h_{N-3}	h_{N-4}	h_{N-5}	...	0	0
(N - 3)T	h_{N-4}	h_{N-5}	h_{N-6}	...	0	0
(N - 4)T	h_{N-5}	h_{N-6}	h_{N-7}	...	0	0
:	:	:	:	:	:	:
3T	h_2	h_1	h_0	...	0	0
2T	h_1	h_0	0	...	0	0
T	h_0	0	0	...	0	0
0	0	0	0	...	0	0

(b) Coefficients in the shift register within the first N time periods

Figure 2.5 A diagram that illustrates the operation of the filter within the first N periods



(a) Signals in the current-memory cells within the $N+1$ to $2N$ time period



(b) Coefficients in the shift register within the $N+1$ to $2N$ time period

Figure 2.6 A diagram that illustrates the operation of the filter within the $N+1$ to $2N$ period diagrams, simply take the corresponding rows in each diagram (a) and (b) and perform the *dot-product* as in (2.3).

$$y(T) = \begin{bmatrix} h_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix}' \begin{bmatrix} x(T) \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.3a}$$

For example, to calculate the output at time $2T$, take the rows corresponding to $2T$ in each diagram and form the dot-product as in (2.3b).

$$y(2T) = \begin{bmatrix} h_1 \\ h_0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix}' \begin{bmatrix} x(T) \\ x(2T) \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.3b}$$

The same method is continued to find the output of the filter in normal operation as illustrated in Fig. 2.6, which shows the status of the next N time periods. The output at time $(N-1)T$ is given in (2.3c),

$$y((N-1)T) = \begin{bmatrix} h_{N-2} \\ h_{N-3} \\ h_{N-4} \\ \vdots \\ h_1 \\ h_0 \\ 0 \end{bmatrix}' \begin{bmatrix} x(T) \\ x(2T) \\ x(3T) \\ \vdots \\ x((N-2)T) \\ x((N-1)T) \\ 0 \end{bmatrix} \tag{2.3c}$$

whereas the output at time NT is given by (2.3d).

$$y(NT) = \begin{bmatrix} h_{N-1} \\ h_{N-2} \\ h_{N-3} \\ \vdots \\ h_2 \\ h_1 \\ h_0 \end{bmatrix}' \begin{bmatrix} x(T) \\ x(2T) \\ x(3T) \\ \vdots \\ x((N-2)T) \\ x((N-1)T) \\ x(NT) \end{bmatrix} \tag{2.3d}$$

2.3.2 Pseudo-Differential Design: Implementation of Negative Coefficients

The previous diagrams have shown a single-ended signal-flow for simplicity. In the actual implementation, the circuits are differential. A true balanced differential realization is difficult to implement. However, we use a pseudo-differential structure to gain most of the benefits of cancellation of common-mode errors, such a clock-feedthrough, and to facilitate the generation of negative coefficients.

To explain further how the architecture is implemented, we can explore the structure of a single tap, as illustrated in Fig. 2.7, in more detail. The FIR filter requires four-quadrant multiplication. The input signal and the filter coefficients can be either positive or negative. A positive signal current means that the current flows into the current memory cell, while a negative signal value indicates that the current will flow out of the memory cell. To generate a four-quadrant multiplication, dual memory cells are used — one with positive current output and one with negative. Negative multiplier coefficients can then be implemented simply by using a switch. If the coefficient

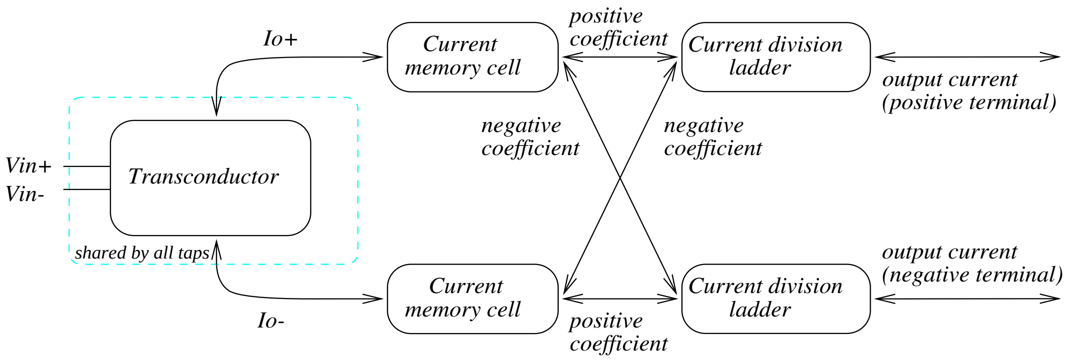


Figure 2.7 One tap of the FIR filter

is positive the current is passed directly to the current-division ladder circuit. However, if the coefficient is negative, a cross-connection is made.

2.4 Signal Discrepancy in Sample-State vs. Hold-State

The output of this analog FIR filter, is a sampled-data analog signal consisting of discrete jumps at the clock-rate. The sampling nature of the filter leads to a discrepancy between the output signal in the sample-state and the hold-state. The fact is that during the leading half period (sampling) of any clock cycles, one of the taps is sampling the signal and is not available to be processed by the following circuitry. Therefore, the output is the weighted sum of the past $N-1$ signals, excluding the current sample at that particular moment. Thus, a small difference exists at the output in the hold-state (valid data) and the sample-state (invalid data) as illustrated in Fig. 2.8.

The error between the output in the two states can be obtained without going through a complex calculation. As the coefficient h_0 is coincident with the most recent sample, the only term not contributing to the output is given

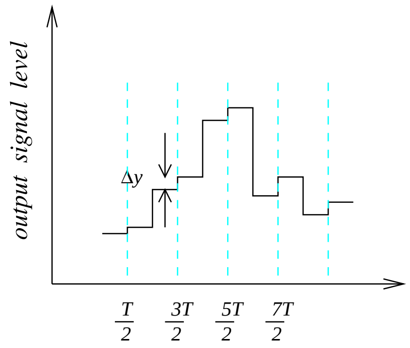


Figure 2.8 Output signal difference between the sample state and hold state

by

$$\Delta y = h_0 x(nT) \tag{2.4}$$

during sampling. There are two factors affecting Δy in (2.4), namely the coefficient h_0 and the current input signal $x(nT)$. There is no error, (i.e., Δy equals 0), when either h_0 or $x(nT)$ equals 0. The polarity of Δy also depends on the polarities of h_0 and $x(nT)$ respectively.

For the application as an analog equalizer followed by an ADC, this discrepancy is not important to a first-order. All that is important is that the ADC sample the data at the correct instant when the output data is valid. However, for high-speed applications, it may be important to give the ADC a longer setup time to make a valid data conversion. In this case the wasted half-clock cycle when the SI-FIR data is invalid can be important.

Addition of Latency for Reduced Glitching and High Sample-Rates

If it is desired to have the data valid for the entire clock cycle, this can be easily accomplished by adding an extra current memory cell which has a multiplier coefficient of zero during the sample-state. This will add one tap to the filter of value zero and add a latency of one clock cycle to the output.

However, there will no longer be a glitch in the output in the middle of the clock cycle as the new data becomes active.

This technique of adding latency can be extended to further increase the speed of the filter. Consider a case where the settling time of the current memory cell takes three clock cycles. In this case, three extra memory cells can be added with corresponding coefficients of zero. This will add latency of three clock cycles to the output, but it will allow the filter to clock at a higher rate.

The maximum clock-rate of the SI-FIR filter, maximum signal frequency, and accuracy will be discussed in the following chapter, where details of the transistor-level circuit design of each block will be presented.