TIME-MODE CIRCUITS FOR ANALOG COMPUTATION

By

VISHNU RAVINUTHULA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2006

My parents and brother have brought me to where I stand today. This work is dedicated to them for trusting me and standing by me through all hardships.

ACKNOWLEDGMENTS

I owe a special debt of gratitude to Dr. Harris for his expert guidance and stimulating discussions. His perception, insight, and experience have contributed immensely to the clarity and rigor of my research. The faith he showed was the motivating force towards my contribution. My association with him has been an enlightening and refreshing experience.

I am immensely thankful to Dr. Jose Fortes for his help when I was in a quagmire and it was a privilege working under him.

I am also grateful to the analog genius Dr. Robert Fox, for he went out of the way to help me even when I was not working under him. He is one of the few professors I feel proud that I got to work with.

This work was supported by National Aeronautics and Space Administration (NASA) under award no. NCC 2-1363 and Semiconductor Research Corporation (SRC) under Task ID: 1049 - Crosscut Research. I would like to thank Dr. M. P. Anantram, Dr. Harry Partridge, and Dr. T. R. Govindan at NASA Ames Research Center, CA, for the confidence they showed in me and all their support during my internship at NASA.

Thanks are due to the administrative staff of the Department of Electrical and Computer Engineering: Ellie, Janet, Linda, and Shannon for their co-operation. Furthermore, I take this opportunity to express my appreciation to all those who helped me in the completion of my work.

I would like to thank my former and current labmates: Pravin, Vaibhav, Rama, Du, Xiaoxiang, Yuan, Harpreet, Mark, Meena, Harsha, and Ismail for their support and encouragement. Lastly, but not the least, I thank my friends, roommates and colleagues for making my stay at UF memorable.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

Figure page

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

TIME-MODE CIRCUITS FOR ANALOG COMPUTATION

By

Vishnu Ravinuthula

August 2006

Chair: John G. Harris
Major Department: Electrical and Computer Engineering

We introduce a set of basic circuit building blocks for analog computation using a temporal step function representation for the inputs and outputs. Time-mode circuits are described that use a step function representation for computing the weighted average, weighted difference, weighted sum, scalar product, maximum, minimum, multiplication, division and thresholded difference operations. Time-mode circuits are alternatives to well-known voltage- and current-mode approaches which could be used to perform these same mathematical operations.

Time-mode circuits grow more appealing as CMOS process technologies scale since they minimize the amount of analog circuitry and use noise robust asynchronous time events as inputs and outputs. Time-mode circuits provide a seamless interface to the growing number of time-based sensors which already output compatible timing events. An example is given where a time-mode edge detector is developed to directly interface to the output of a time-to-first-spike imager. Time-mode circuits have simple architecture, provide high signal-to-noise ratio, dynamic range, consume low power, and hence, prove advantageous in architecturally complex applications like finite impulse response filters.

CHAPTER 1
INTRODUCTION

All analog signal processing circuits must represent signals using physical quantities such as voltage, current, charge, frequency or time duration. In analog literature, we have seen extensive use of voltage-mode, current-mode and charge-mode circuits that generally represent input and output signals as voltage, current and charge respectively:

- Voltage-mode circuits are the most common example, wherein voltages are used to represent both input and output signals. These circuits have a long history including classic opamp based designs [1] and GmC style circuits common in today's analog very large scale integration (VLSI) designs [2].

- Current-mode circuits are also very popular where currents are used for both inputs and outputs [3]. These designs include Barrie Gilbert's original translinear circuits that rely on the exponential voltage to current relationships of bipolar or complementary metal oxide semiconductor (CMOS) subthreshold circuits [4]. More recent log domain filters [5] are also generally considered to be current-mode circuits.

- Another physical quantity is charge, and charge-mode circuits have been employed in various applications, particularly for charge coupled devices (CCDs) [6].

These different modes of signal representation have respective advantages and drawbacks and can therefore be used in different parts of the same system. The voltage representation makes it easy to distribute a signal in various parts of a circuit, but implies a large stored energy $\frac{CV^2}{2}$ into the node's parasitic capacitance C. The current representation facilitates the summing of signals but complicates their distribution. Replicas must be created which are never exactly equal to the original signal. It has been observed that it is problematic to clearly define the distinction between current-mode and voltage mode circuits [7]. The charge representation requires time sampling but can be nicely processed by means of CCDs or switched-capacitor techniques. In actual fact, every circuit uses voltage, current and charge in its operation and sometimes semantics and philosophy are debated when definitively categorizing these classes of circuits [7].

Temporal coding is used as the dominant mode of signal representation for communication in biological nervous systems. Signals represented in this manner are easy to regenerate and this representation might therefore be preferred for long-distance transfers of information. It is discontinuous in time, but the phase information is kept in asynchronous systems. We introduce time-mode circuits as another category of analog signal processing circuits that represent input and output signals in the temporal domain. Figure 1–1 depicts block diagrams for voltage-, current-, and time-mode circuits. Time-mode circuits use temporal events, in this case voltage steps, to represent signals.

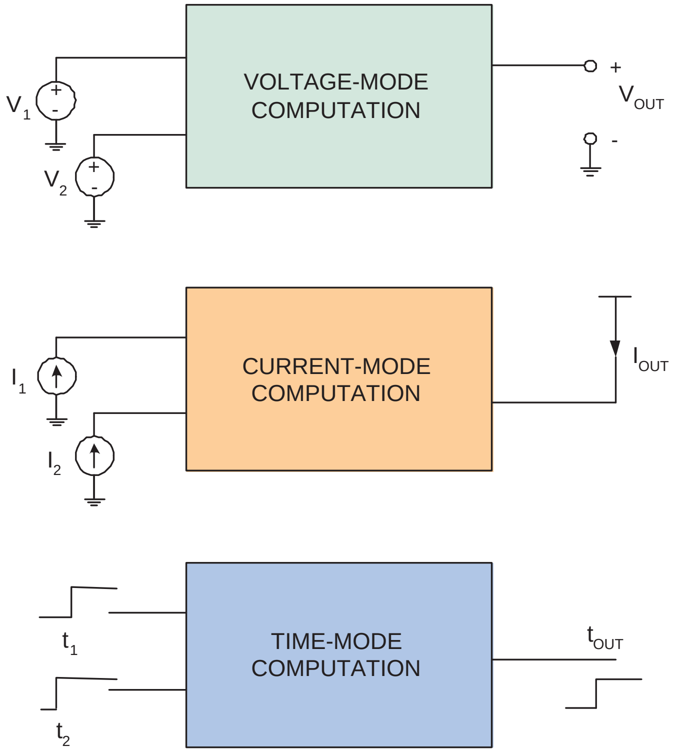

Figure 1–1: Different modes of computation.

1.1 Biological Motivation

The idea of performing computation using the timing of events is shared with the most powerful existing computer: the human brain. The brain is an analog computer, but it does not transmit continuous analog voltages, likely due to noise and cross-talk susceptibility. Instead, information is represented and transmitted using the timing of asynchronous digital-like timing pulses. However, an important difference is that time-mode circuits described in this thesis use a step function representation because of the resulting circuit simplicity compared to pulse representations in mathematical computations.

Since we are using a step function representation, we cannot represent information in terms of firing rates (where we need multiple spikes to represent an analog variable). This new approach is similar to temporal coding by single spikes [8] rather than on the traditional interpretation of analog variables in terms of firing rates.

Maass [8] points out that a spiking neuron in principle will be able to compute in temporal coding of inputs and output a linear function if its postsynaptic potential can be described or approximated by a linear function during some initial segment. As we will see in Chapter 2, time-mode circuits perform linear computations by linearly mapping the temporal inputs to a voltage across a capacitor. Also, Maass points out that networks of noisy spiking neurons are —universal approximators— they can approximate with regard to temporal coding any given continuous function of several variables. This observation is proved in Chapter 6 where we use a network of time-mode circuits to implement an approximation of the multiplication function (non-linear function).

1.2 Engineering Motivation

Independent of any biological motivation, it is also making more and more sense to consider analog computation using the timing of asynchronous events from a purely engineering perspective. Through the electronics revolution over the past decades, CMOS process technology is shrinking the usable voltage swing, wreaking havoc on traditional analog circuit design. However, the faster “digital” transistors are better able to process timing signals leading us to consider analog computation more similar to that of the brain. This trend will likely continue with nanotechnology since even smaller voltage ranges and even faster devices are promised. Of course, CMOS technology is primarily scaling in favor of faster and faster digital devices, however power consumption is beginning to limit how far these digital circuits can scale.

Time-based signal representations have been in use for many years, including such techniques as pulse-width modulation and sigma-delta converters but temporal codes are becoming even more common with the rising popularity of such techniques as class D amplifiers, spike-based sensors and even ultra-wideband (UWB) signal transmission. However, these temporal codes are typically used as temporary representations and computation is only performed after reconstruction back to a traditional analog or digital form. There are instances where amplifiers use temporal signals as inputs and outputs [9], but they do not perform computation with them.

There are architectures like the PALMO [10] where the inputs and outputs are represented by temporal signals, but using pulses. In such architectures, the input temporal pulses are immediately converted to voltage and they lose the computational advantages that the time-based representation promises. Similarly, Murray discusses the implementation of arithmetic functions like addition and multiplication using voltage or current pulses [11]. Another approach by Sarpeshkar uses pulses for scalable hybrid computation [12]. However, all the above mentioned architectures use pulses for computation with more complicated circuits than the time-mode circuits.

In this thesis we describe a set of basic circuit building blocks for computation using an analog temporal step function representation for both inputs and outputs.

1.3 Chapter Summary

The thesis is divided into the following chapters:

- Chapter 2: The Weighted Average Circuit. This chapter introduces time-mode computation by describing the details of the weighted average circuit in terms of functionality and fabricated chip measurements. We then proceed to see how new and emerging silicon/carbon-nanotube technologies affect the performance of this prototype time-mode circuit. Later we introduce a method to improve the reliability of this time-mode circuit.

- Chapter 3: SNR Comparison Of Weighted Averaging Circuits. In this chapter we derive expressions for the Signal-to-noise ratio of voltage-mode, current-mode and time-mode 2-input weighted average circuits, use these expressions to compare and contrast the SNR performances of those circuits and verify the observations with simulations.

- Chapter 4: Other Time-Mode Circuit Examples. This chapter describes other time-mode circuit examples including the weighted difference, weighted sum, scalar product, maximum, minimum and thresholded difference operations.

- Chapter 5: Application Of Time-Mode Circuits. In this chapter we talk two applications of linear time-mode computational circuits: We start with the design of a time-mode edge detector that interfaces directly to a time-to-first-spike imager. Later we describe the design of a 3-tap time-mode FIR filter. We analyze these two applications and describe the advantages of using time-mode circuits for these applications.

- Chapter 6: Non-Linear Time-Mode Computation. In this chapter we discuss two different methods of performing non-linear computation using time-mode circuits: The first method implements non-linear arithmetic using a time-mode multi-layer perceptron and the second method implements non-linear arithmetic by introducing non-linearity in the existing linear computational blocks.

- Chapter 7: Conclusion And Future Work. This chapter concludes the thesis by recapitulating all the main points of the thesis. We also discuss possible future research work.

CHAPTER 2
THE WEIGHTED AVERAGE CIRCUIT

2.1 Time-Mode Weighted Averaging Circuit

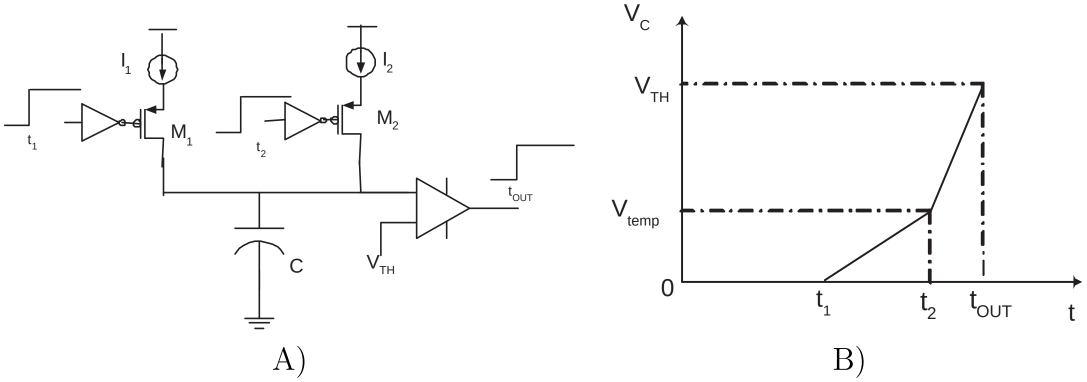

Figure 2–1. Time-mode weighted average circuit. A) Circuit schematic. B) Idealized graph showing the capacitor voltage at different time periods.

Figure 2–1A illustrates the basic elements used to perform a weighted sum of temporal signals. In general, the circuit can process many input steps but only two are shown for simplicity. The circuit consists of a single capacitor and comparator plus an inverter, current source and pfet for each input[1] . The rising edges of the input steps correspond to the time values t_1 and t_2 representing the two input values. The PMOS transistors M_1 and M_2 act as switches. The two current sources I_1 and I_2 are connected to the sources of the PMOS transistors to start charging the capacitor C when the step inputs rise. The comparator senses the voltage across the capacitor and outputs a step when the voltage reaches the threshold

[1] The inverters would not be necessary if nfets were used to sink current or if an inverted step function was used to represent input and output values. Furthermore, source signalling should be used to reduce charge injection effects as explained in [13].

voltage V_{TH}. Once the block outputs a step, an appropriate reset stage (not shown in the figure) resets the capacitor to $0V$. The current sources I_1 and I_2 charge the capacitor during different time periods as shown in Figure 2–1B.

Initially, the voltage across the capacitor (V_C) is reset to ground. For simplicity, let $t_1 < t_2 < t_{OUT}$. The capacitor voltage V_C stays at 0V until the first step arrives at time t_1. Transistor M_1 turns on and the voltage V_C linearly increases with the current source I_1 charging capacitor C. This linear increase continues until time t_2 when the second step arrives. For purposes of the following discussion, the capacitor voltage at that instant is labeled V_{temp}.

The value of V_{temp} is computed during the period t_1 to t_2 as

$$V_{temp} = \frac{I_1}{C}(t_2 - t_1) \tag{2–1}$$

Similarly, during the period t_2 to t_{OUT} (the time for the capacitor to charge to V_{TH})

$$V_{TH} - V_{temp} = \frac{I_1 + I_2}{C}(t_{OUT} - t_2) \tag{2–2}$$

Solving Eqs. 2–1 and 2–2 gives:

$$t_{OUT} = \frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{C V_{TH}}{I_1 + I_2} \tag{2–3}$$

where t_{OUT} is the time when the output step makes its transition from low to high voltage. Eq. 2–3 is symmetric with $I_1 t_1$ and $I_2 t_2$ so the assumption that $t_1 < t_2$ can be relaxed. However, we still need to assume that t_{OUT} occurs after t_1 and t_2 to ensure the validity of the equations.

The minimum value of t_{OUT} in Eq. 2–3 occurs if $t_{OUT} = t_2$ and substituting into Eq. 2–3, gives

$$I_2 t_2 - I_1 t_1 = C V_{TH} \tag{2–4}$$

Therefore for general values of t_{OUT} above the minimum value,

$$|I_2 t_2 - I_1 t_1| < CV_{TH} \tag{2–5}$$

Eq. 2–5 provides the relation to be met for Eq. 2–2 to be valid. For the special case where $I_1 = I_2 = I$, then the output step time t_{out} is the mean of t_1 and t_2 plus a programmable constant $(CV_{TH}/2I)$.

For unequal values of I_1 and I_2,

$$t_{OUT} = \frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2} \tag{2–6}$$

where $\frac{CV_{TH}}{I_1+I_2}$ is a constant. The above equation is valid, when

$$|I_1 t_1 - I_2 t_2| < CV_{TH} \tag{2–7}$$

In this case, then the output time step is the weighted average of t_1 and t_2 plus a constant.

We can summarize the results obtained above in a single equation:

$$t_{OUT} = \begin{cases} t_1 + \dfrac{CV_{TH}}{I_1}, & \text{for } |I_1 t_1 - I_2 t_2| \geq CV_{TH} \text{ and } t_1 < t_2 \quad \text{(2–8a)} \\ t_2 + \dfrac{CV_{TH}}{I_2}, & \text{for } |I_1 t_1 - I_2 t_2| \geq CV_{TH} \text{ and } t_2 < t_1 \quad \text{(2–8b)} \\ \dfrac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \dfrac{CV_{TH}}{I_1 + I_2} & \text{otherwise} \quad \text{(2–8c)} \end{cases}$$

In general for N input steps,

$$t_{OUT} = \frac{\sum_n I_n t_n}{\sum_n I_n} + \frac{CV_{TH}}{\sum_n I_n} \tag{2–9}$$

provided that V_{TH} is large enough. The circuit can be further generalized to handle negative weight values in several ways; for instance, using current sources that sink current to ground as long as the output voltage stays positive and eventually reaches V_{TH}.

Inputs t_1, t_2 and output t_{OUT} of the weighted average circuit are time-steps and these time-steps are defined within a frame (Frame 1) as shown in Figure 2–2. When a frame (Frame 1) ends, the inputs and the output steps also end and the circuit is reset. As the next frame starts (Frame 2 in the figure), the circuit would be ready to process the next set of inputs t_1 and t_2. Each frame should be long enough to allow the weighted average circuit to produce its output. For bounded frame lengths, there is a chance that the output will not occur.

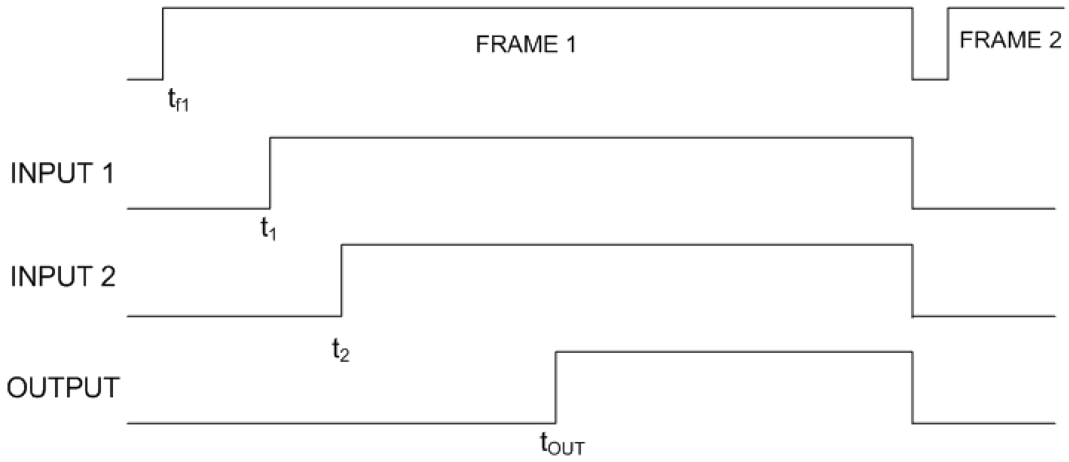

Figure 2–2. Inputs t_1, t_2 and output t_{OUT} are defined within a frame

2.1.1 Reset Stage

In the circuit shown in Figure 2–1A, we have not shown an explicit reset stage. Setting the capacitor's voltage to its initial voltage V_{TH} through a transmission gate is the functionality desired from the reset stage and this can be integrated with the application's reset stage.

Later in this dissertation, we will discuss some applications of this time-mode weighted average circuit: an edge detection circuit and a 3-tap FIR filter. These applications have custom reset stages and the reset for the basic block is integrated in these custom reset stages.

2.1.2 Measured Results

The weighted averaging circuit was fabricated using the AMI $0.6\mu m$ CMOS process. Figure 2–3 shows the measured output t_{OUT} when just one input is provided to the circuit. The output t_{OUT} is plotted for varying I. The values of C and V_{TH} in the circuit are $20pF$ and $2.5V$ respectively. Figure 2–4 also shows the output t_{OUT} when only one input (occuring at t_1) is provided to the circuit. The current source I is fixed at $1.05\mu A$. The input transition time t_1 was varied externally and the output t_{OUT} was measured and plotted. The output expected from the block $t_{OUT} = t_1 + \frac{CV_{TH}}{I}$ was also plotted. The values of C and V_{TH} in the circuit are $20pF$ and $2.5V$ respectively. The root mean squared error ($\sqrt{MSE}$) between the expected results and the measured results obtained of the output t_{OUT} obtained was $0.26\mu s$.

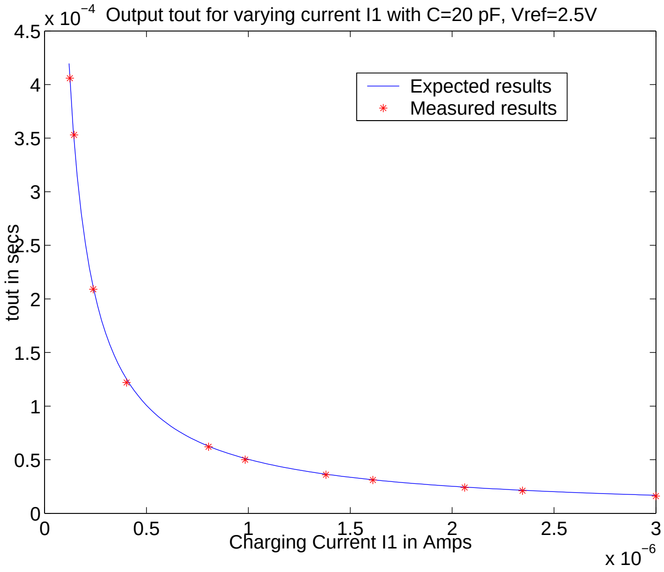

Figure 2–3. Plot of t_{OUT} for varying I ($C = 20pF$, $V_{TH} = 2.5V$). The block was given one step input.

Figure 2–5 shows the output t_{OUT} when both the inputs are provided to the circuit but the current sources I_1 and I_2 are fixed at $1.552\mu A$. The first input entering the block was fixed as $1\mu s$, $8.5\mu s$ and $32.5\mu s$ for three different sets of measurements. The input transition time t_2 was varied externally for different

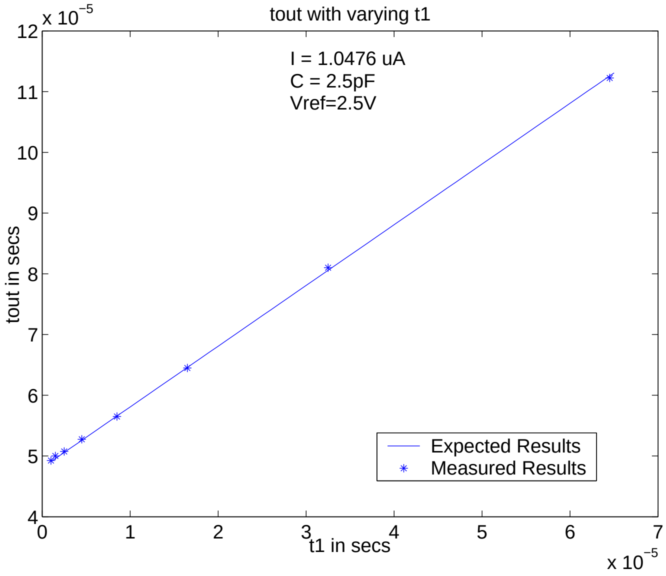

Figure 2–4. Plot of t_{OUT} for varying I ($C = 20pF$,$I = 1.0476\mu A$, $V_{TH} = 2.5V$). The block was given one step input.

values of t_1 and the output t_{OUT} was measured and plotted. The output expected from the block $t_{OUT} = \frac{t_1+t_2}{2} + \frac{CV_{TH}}{2I}$ was also plotted. The values of C and V_{TH} in the circuit are $20pF$ and $2.5V$ respectively. The $\sqrt{MSE}$ between the expected results and the measured results obtained of the output t_{OUT} obtained was $0.3\mu s$.

Figure 2–6 shows the output t_{OUT} for the case when both inputs are provided to the circuit but the current sources I_1 and I_2 are different and are fixed at $1.46\mu A$ and $0.29\mu A$ respectively. Similar to the above case, the first input entering the block was fixed as $1\mu s$, $8.5\mu s$ and $32.5\mu s$ for three different sets of measurements. The input transition time t_2 was varied externally for different values of t_1 and the output t_{OUT} was measured and plotted. The output expected from the block $t_{OUT} = \frac{I_1t_1+I_2t_2}{I_1+I_2} + \frac{CV_{TH}}{I_1+I_2}$ was also plotted. The values of C and V_{TH} in the circuit are $20pF$ and $2.5V$ respectively. The $\sqrt{MSE}$ between the expected results and the measured results obtained of the output t_{OUT} obtained was $1.9\mu s$.

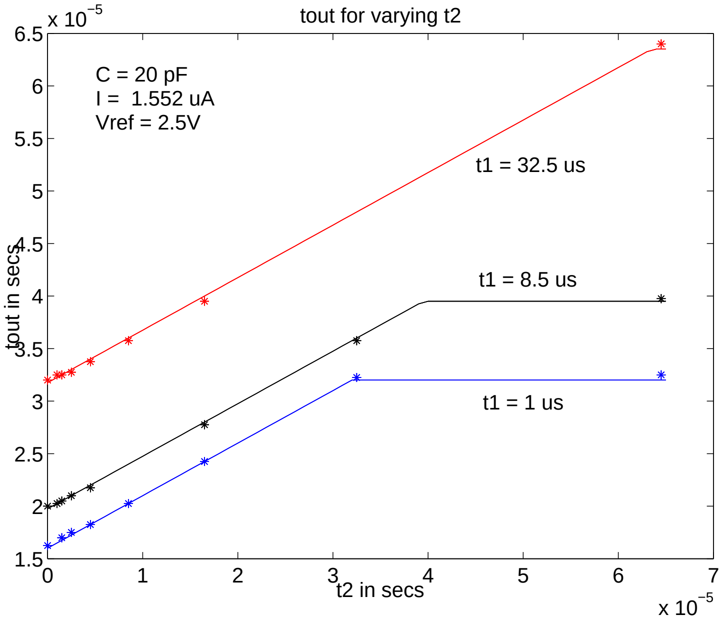

Figure 2–5. Plot of t_{OUT} for varying t_2 ($C = 20pF$,$I = 1.0476\mu A$, $V_{TH} = 2.5V$ with t_1 fixed at $1\mu s$, $8.5\mu s$ and $32.5\mu s$.)

2.1.3 Discussion

Errors in the weighted average calculation arise from a number of sources including:

- mismatches in capacitor and current source values.
- fundamental noise sources causing jitter in the timing of the input and output step functions.
- transistor time delays, for example through the comparator.

Each of these errors can be reduced somewhat with careful layout, larger circuits, more power consumption, and/or calibration procedures. These tradeoffs must be taken based on the demands of particular applications. Particular advantages and disadvantages of the weighted average circuit and other time-mode circuits must be carefully considered. It is likely that time-mode circuits will have larger dynamic ranges than conventional designs but high speed operation will be compromised since time is used in the representations. General claims are difficult, especially considering that it even difficult to cite general advantages of

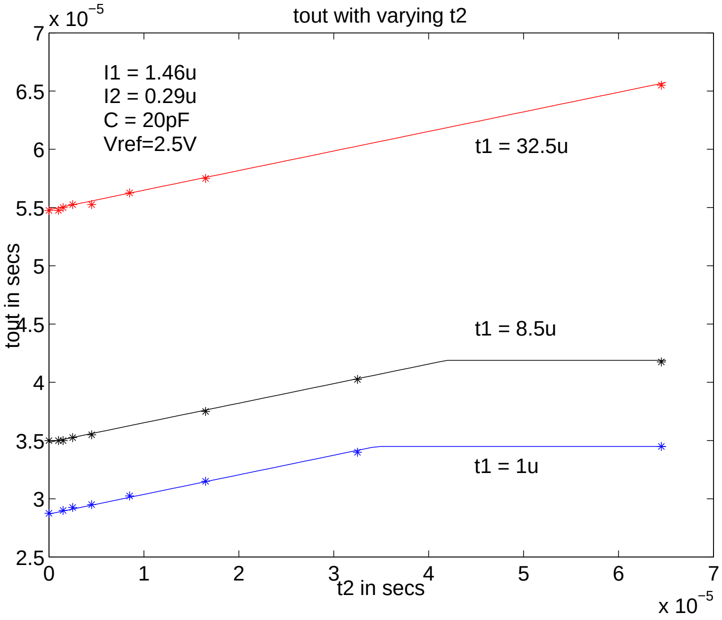

Figure 2–6. Plot of t_{OUT} for varying t_2 ($C = 20pF$,$I_1 = 1.46\mu A$, $I_2 = 0.29\mu A$, $V_{TH} = 2.5V$ with t_1 fixed at $1\mu s$, $8.5\mu s$ and $32.5\mu s$.)

current-mode circuits vs. voltage-mode circuits [7]. A big advantage of time-mode circuits however is that more and more sensors are being designed with step outputs [14][15] and the time-mode circuits can directly interface to these sensors.

Table 2–1. Measured performance characteristics of time-mode weighted averaging circuit.

Performance specification	Value
Power consumption	$0.6\mu W$
SNR	$56dB$
Differential-mode dynamic range	$62dB$
Common-mode dynamic range	Effectively infinite

By performing computation using temporal step functions, the averaging block was able to achieve almost infinite common-mode dynamic range, $62dB$ differential-mode dynamic range and SNR of $56dB$ with very low power consumption of $0.6\mu W$. This power consumption was on the order of nanowatts, when the comparator were operated in the sub-threshold region. But, the operating speed of the comparator was slow. Therefore, we had to strike a trade-off between the comparator's operating speed and the power consumption. The measured circuit specifications are tabulated in Table 2–1.

2.2 Theoretical Analysis of Signal-to-Noise Ratio and Dynamic Range

There are two sources of output noise in a time-mode circuit.

1. output noise due to timing jitter at the inputs.

2. output noise due to fundamental noise sources in the circuit.

2.2.1 Output Noise due to Timing Jitter at the Inputs

We know that the output from the block shown in Figure 2–1A is:

$$t_{OUT} = \frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2} \tag{2–10}$$

Jitter Δt_1 (across different input values it has a mean value of $\overline{\Delta t_1}$ and a variance of $\overline{\Delta t_1{}^2}$) in the input t_1, causes the following output

$$t_{OUT1} = \frac{I_1(t_1 + \Delta t_1) + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2} \tag{2–11}$$

The jitter at the output Δt_{OUT1} is given by:

$$\begin{aligned}\Delta t_{OUT1} &= t_{OUT1} - t_{OUT} \\ &= (\frac{I_1(t_1 + \Delta t_1) + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2}) - (\frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2}) \\ &= \frac{I_1 \Delta t_1}{I_1 + I_2}\end{aligned} \tag{2–12}$$

which is a simple scalar multiplication of Δt_1.

The mean of the output jitter scales accordingly as,

$$\overline{\Delta t_{OUT1}} = \frac{I_1 \overline{\Delta t_1}}{I_1 + I_2} \tag{2–13}$$

The variance of this noise can be easily derived to be,

$$\overline{\Delta t_{OUT1}{}^2} = \frac{I_1^2 \overline{\Delta t_1{}^2}}{(I_1 + I_2)^2} \tag{2–14}$$

Similarly, the variance if the output noise caused by the input time jitter Δt_2 (corresponding to input t_2),

$$\overline{\Delta t_{OUT2}}^2 = \frac{I_2^2 \overline{\Delta t_2}^2}{(I_1 + I_2)^2} \tag{2–15}$$

The total variance of the noise at the output is given by,

$$\begin{aligned} \overline{\Delta t_{OUT}}^2 &= \overline{\Delta t_{OUT1}}^2 + \overline{\Delta t_{OUT2}}^2 \\ &= (\frac{I_1^2 \overline{\Delta t_1}^2}{(I_1 + I_2)^2}) + (\frac{I_2^2 \overline{\Delta t_2}^2}{(I_1 + I_2)^2}) \\ &= \frac{I_1^2 \overline{\Delta t_1}^2 + I_2^2 \overline{\Delta t_2}^2}{(I_1 + I_2)^2} \end{aligned} \tag{2–16}$$

If $I_1 = I_2$, then

$$\overline{\Delta t_{OUT}}^2 = \frac{\overline{\Delta t_1}^2 + \overline{\Delta t_2}^2}{4} \tag{2–17}$$

Since only a fraction (25%) of the jitter at the inputs affect the output jitter, we can say that the time-mode weighted average circuit effectively reduces the jitter at the inputs.

2.2.2 Output Noise due to Fundamental Noise Sources in the Circuit

Figure 2–1A illustrates the basic elements used to perform a weighted sum of temporal signals. We already know that the output from this block is:

$$t_{OUT} = \frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{C V_{TH}}{I_1 + I_2} \tag{2–18}$$

We will now derive an expression for the signal-to-noise ratio of this time-mode weighted averaging circuit.

The first step towards the derivation is to define the signal. Assuming a differential representation, let us refer the inputs and outputs of the averaging block to its first input. For simplicity, let us assume t_1 as the first input. The two inputs to the averaging block are defined as, $\hat{t}_1 = t_1 - t_1 = 0$ and $\hat{t}_2 = t_2 - t_1$. The output is defined as $\hat{t}_{OUT} = t_{OUT} - t_1$. This output is defined as the signal. In words, the signal is defined as the output t_{OUT} of the averaging block referred to the input t_1.

Therefore, $\hat{t}_{OUT}$ is given by:

$$
\begin{aligned}
\hat{t}_{OUT} &= t_{OUT} - t_1 \\
&= (\frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{CV_{TH}}{I_1 + I_2}) - t_1 \\
&= \frac{I_1 t_1 + I_2 t_2 + CV_{TH} - I_1 t_1 - I_2 t_1}{I_1 + I_2} \\
&= \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + I_2} \qquad (2\text{–}19)
\end{aligned}
$$

There are 3 noise sources that dominate the noise performance of this circuit.

1. Noise due to the current source I_1.
2. Noise due to the current source I_2.
3. Noise due to the voltage comparator.

These noise sources are uncorrelated and therefore we can consider the impact of each of these noise sources individually on the output of the circuit.

2.2.2.1 Noise in t_{OUT} due to noise in current source I_1

Let us assume that current source I_1 is noisy with a noise current of ΔI_1. Therefore, the total current from the current source is given by $I_1 + \Delta I_1$. Since currents $I_1 + \Delta I_1$ and I_2 charge the capacitor, the new output from the weighted average circuit is given by,

$$t_{OUT1} = \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + \Delta I_1 + I_2} \qquad (2\text{–}20)$$

The noise at the output Δt_{OUT} can be calculated as shown below:

$$\begin{aligned}
\Delta t_{OUT1} &= t_{OUT1} - \hat{t}_{OUT} \\
&= \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + \Delta I_1 + I_2} - \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + I_2} \\
&= \frac{(I_2(t_2 - t_1) + CV_{TH})(-\Delta I_1)}{(I_1 + \Delta I_1 + I_2)(I_1 + I_2)} \\
&\approx \frac{(I_2 t_2 - I_2 t_1 + CV_{TH})(-\Delta I_1)}{(I_1 + I_2)^2} \qquad (2\text{–}21)
\end{aligned}$$

The variance of this noise is given by

$$\overline{\Delta t_{OUT1}{}^2} \approx \frac{(I_2 t_2 - I_2 t_1 + CV_{TH})^2(\overline{\Delta I_1{}^2})}{(I_1 + I_2)^4} \qquad (2\text{–}22)$$

2.2.2.2 Noise in t_{OUT} due to noise in current source I_2

Let us assume that current source I_2 is noisy with a noise current of ΔI_2. Therefore, the total current from the current source is given by $I_2 + \Delta I_2$. Since currents I_1 and $I_2 + \Delta I_2$ charge the capacitor, the new output from the weighted average circuit is given by,

$$t_{OUT2} = \frac{(I_2 + \Delta I_2)(t_2 - t_1) + CV_{TH}}{I_1 + I_2 + \Delta I_2} \qquad (2\text{–}23)$$

The noise at the output Δt_{OUT} can be calculated as shown below:

$$\begin{aligned}
\Delta t_{OUT2} &= t_{OUT2} - \hat{t}_{OUT} \\
&= \frac{(I_2 + \Delta I_2)(t_2 - t_1) + CV_{TH}}{I_1 + I_2 + \Delta I_2} - (\frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + I_2}) \\
&= \frac{(t_2 - t_1)(I_1 \Delta I_2)}{(I_1 + I_2 + \Delta I_2)(I_1 + I_2)} + CV_{TH}(-\frac{\Delta I_2}{(I_1 + I_2 + \Delta I_2)(I_1 + I_2)}) \\
&\approx \frac{(t_2 - t_1)(I_1 \Delta I_2)}{(I_1 + I_2)^2} + CV_{TH}(-\frac{\Delta I_2}{(I_1 + I_2)^2}) \\
&\approx \Delta I_2(\frac{I_1(t_2 - t_1) - CV_{TH}}{(I_1 + I_2)^2}) \qquad (2\text{–}24)
\end{aligned}$$

The variance of this noise is given by

$$\begin{aligned}\overline{\Delta t_{OUT2}}^2 &\approx \frac{(I_1(t_2-t_1)-CV_{TH})^2(\overline{\Delta I_2^2})}{(I_1+I_2)^4} \\ &\approx \frac{(I_1(t_1-t_2)+CV_{TH})^2(\overline{\Delta I_2^2})}{(I_1+I_2)^4} \qquad (2–25)\end{aligned}$$

2.2.2.3 Noise in t_{OUT} due to noise in the comparator

Let us assume that current sources I_1 and I_2 are noiseless and noise in the comparator is given by ΔV. The output from the weighted average circuit for these assumptions is given by,

$$t_{OUT3} = \frac{I_2(t_2-t_1)+C(V_{TH}+\Delta V)}{I_1+I_2} \qquad (2–26)$$

The noise at the output Δt_{OUT} can be calculated as shown below:

$$\begin{aligned}\Delta t_{OUT3} &= t_{OUT3} - \hat{t}_{OUT} \\ &= \frac{I_2(t_2-t_1)+C(V_{TH}+\Delta V)}{I_1+I_2} - \frac{I_2(t_2-t_1)+CV_{TH}}{I_1+I_2} \\ &= \frac{C\Delta V}{(I_1+I_2)^2} \qquad (2–27)\end{aligned}$$

The variance of this noise is given by

$$\overline{\Delta t_{OUT3}}^2 = \frac{(C\overline{\Delta V^2})}{(I_1+I_2)^4} \qquad (2–28)$$

For typical values, the noise variation denoted by Eq. 2–28 is negligible compared to the noise variances shown in Eqs. 2–22 and 2–25. Therefore, in the forthcoming calculations we neglect the noise contribution of the comparator.

Therefore, the total noise at the output of the averaging block is given by

$$\overline{\Delta t_{OUT}}^2 \approx \frac{(I_2t_2-I_2t_1+CV_{TH})^2(\overline{\Delta I_1^2})}{(I_1+I_2)^4} + \frac{(I_1t_1-I_1t_2+CV_{TH})^2(\overline{\Delta I_2^2})}{(I_1+I_2)^4} \qquad (2–29)$$

The noise in the current sources I_1 and I_2 is dominated by shot noise. In general, shot noise due to a current source I is given by [16],

$$\overline{\Delta I^2} = \frac{eI}{\tau} \tag{2–30}$$

where τ is the time during which the current source is ON and contributes to the output and e is the charge of an electron.

Current source I_1 charges the capacitor during the time period $t_{OUT} - t_1$. Noise in the current source would affect the circuit only during this time period. Therefore,

$$\begin{aligned}\tau_1 &= \hat{t}_{OUT} - \hat{t}_1 \\ &= (t_{OUT} - t_1) - (t_1 - t_1) \\ &= t_{OUT} - t_1 \\ &= \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + I_2}\end{aligned} \tag{2–31}$$

Therefore,

$$\begin{aligned}\overline{\Delta I_1}^2 &= \frac{eI_1}{\frac{I_2(t_2-t_1)+CV_{TH}}{I_1+I_2}} \\ &= \frac{eI_1(I_1 + I_2)}{I_2(t_2 - t_1) + CV_{TH}}\end{aligned} \tag{2–32}$$

Current source I_2 charges the capacitor during the time period $t_{OUT} - t_2$. Only during this time period, the noise in the current source would affect the circuit. Therefore,

$$\begin{aligned}\tau_2 &= \hat{t}_{OUT} - \hat{t}_2 \\ &= (t_{OUT} - t_1) - (t_2 - t_1) \\ &= \frac{I_2(t_2 - t_1) + CV_{TH}}{I_1 + I_2} - (t_2 - t_1) \\ &= \frac{I_1(t_1 - t_2) + CV_{TH}}{I_1 + I_2}\end{aligned} \tag{2–33}$$

Therefore,

$$\begin{aligned}\overline{\Delta I_2}^2 &= \frac{eI_2}{\frac{I_1(t_1-t_2)+CV_{TH}}{I_1+I_2}} \\ &= \frac{eI_2(I_1+I_2)}{I_1(t_1-t_2)+CV_{TH}} \qquad (2\text{–}34)\end{aligned}$$

Substituting Eqs. 2–32 and 2–34 into (2–29), we get

$$\begin{aligned}\overline{\Delta t_{OUT}}^2 &= \frac{(I_2t_2-I_2t_1+CV_{TH})^2(\frac{eI_1(I_1+I_2)}{I_2(t_2-t_1)+CV_{TH}})}{(I_1+I_2)^4}+\frac{(I_1t_1-I_1t_2+CV_{TH})^2(\frac{eI_2(I_1+I_2)}{I_1(t_1-t_2)+CV_{TH}})}{(I_1+I_2)^4} \\ &= \frac{eI_1(I_2(t_2-t_1)+CV_{TH})+eI_2(I_1(t_1-t_2)+CV_{TH})}{(I_1+I_2)^3} \\ &= \frac{eCV_{TH}(I_1+I_2)}{(I_1+I_2)^3} \\ &= \frac{eCV_{TH}}{(I_1+I_2)^2} \qquad (2\text{–}35)\end{aligned}$$

The signal-to-noise ratio is given by:

$$\begin{aligned}SNR &= \frac{(\frac{I_2(t_2-t_1)+CV_{TH}}{I_1+I_2})^2}{\frac{eCV_{TH}}{(I_1+I_2)^2}} \\ &= \frac{(I_2(t_2-t_1)+CV_{TH})^2}{eCV_{TH}} \qquad (2\text{–}36)\end{aligned}$$

Eq. 2–36 gives the SNR for a time-mode weighted average circuit.

For an averaging circuit, $I_1 = I_2 = I$. Therefore,

$$SNR = \frac{(I(t_2-t_1)+CV_{TH})^2}{eCV_{TH}} \qquad (2\text{–}37)$$

The maximum value of this SNR occurs when

$$t_2 - t_1 = \frac{CV_{TH}}{I} \qquad (2\text{–}38)$$

Substituting Eq. 2–38 into Eq. 2–37, we get

$$\begin{aligned} SNR_{MAX} &= \frac{(I(\frac{CV_{TH}}{I}) + CV_{TH})^2}{eCV_{TH}} \\ &= \frac{(CV_{TH} + CV_{TH})^2}{eCV_{TH}} \\ &= \frac{4(CV_{TH})^2}{eCV_{TH}} \\ &= \frac{4(CV_{TH})}{e} \end{aligned} \tag{2–39}$$

To quantify this peak SNR value, we substituted $C = 2pF$ and $V_{TH} = 5V$ (max value) in Eq. 2–39 and obtained a value of $84dB$. This peak value can be increased by further increasing the value of the capacitance. For a $20pF$ capacitance, we get a SNR of $88dB$.

2.2.3 Discussion

We should note that the SNR value obtained through the hand calculations shown above is an approximation. We have neglected the effect of jitter at the inputs and matching errors in the currents I_1, I_2. Since these effects affect our noise measurements and therefore the measured SNR value, we obtain different values for calculated and measured SNR values.

2.3 Scaling of Time-Mode Weighted Average Circuit with Technology

As we have seen previously time-mode circuits promise very high dynamic range and good SNR. An important question is how does the performance of time-mode circuits scale as technology scales. Since the time-mode weighted average circuit is the prototype of all the time-mode circuits discussed in this thesis, we will project how the time-mode weighted circuit performs in terms of Signal-to-Noise Ratio, Dynamic Range, Power Consumption and Energy Consumption as technology scales.

We choose the current 180nm as the reference technology and 90nm, 65nm, 45nm and 32nm as future technologies for our simulations. PTM HSpice transistor models are used for all the technologies - Predictive technology models are developed by the Nanoscale Integration and Modeling Group at Arizona State University [17].

2.3.1 Simulation Setup

We used Synopsys HSPICE to simulate the time-mode weighted average circuit in different current/future technology nodes. Low voltage cascode current mirrors were used for current sources and a five transistor single-ended differential amplifier (PMOS input pair, NMOS current mirror load) was used as a comparator. In the differential amplifier, a higher W/L ratio for the PMOS inputs and smaller W/L ratio for the loads was chosen to minimize noise and the effects of mismatch. The comparator's systematic offset would affect only the constant component - $\frac{CV_{TH}}{I_1+I_2}$ of the output and can be calibrated out later, if necessary.

To compare the performance of the weighted average circuit as technology scales, the capacitance C and reference voltage V_{TH} were kept constant across different future technology nodes. This ensures that charging currents I_1 and I_2 are the only variable circuit parameters and it allows an apt comparison of the trends in dynamic range, power and SNR of the weighted average circuit across technologies. To vary I_1 and I_2, we had to change the sizes of the transistors in the low-voltage cascode current sources. For simplification, the two charging currents I_1 and I_2 were kept constant. The transistors in the weighted average circuit (the transistors of the low-voltage cascode current mirrors, comparator and the digital switches) were sized for two possible scenarios:

- For every technology, the circuit was designed for the lowest possible current to keep all transistors in active/saturation.
- The transistor sizes were fixed for the 180nm process and the transistor sizes were scaled down by the factor with which the technology scales down.

It was argued previously that the only noise sources that contribute heavily to the noise of the time-mode weighted averaging circuit are the shot noises of the DC current sources I_1 and I_2. The noise of the comparator is negligible. To characterize the noise performance of the weighted average circuit across various future technologies, noise analysis during transient simulation is required. Since such an analysis is not readily available in our Cadence software setup, to achieve this two noise current sources (with Guassian distribution) were generated randomly in MATLAB and their RMS values were set according to equations previously derived. These current sources model the shot noise of the current sources I_1 and I_2 and are connected in parallel to their respective current sources during the simulations. These noise current sources were switched on only during the period when the time steps turned on the DC current sources.

The timing of the first step is chosen as the time reference and was kept constant for all technologies. In this way the output time had the same reference in all technologies.

2.3.2 Results and Inferences

- Power Supply: The VDD was decreased from 1.3 V to 0.6 V with scaling of technologies and the circuit worked well for all values. This shows that the time mode circuits are not dependent on voltage ranges and hence voltage supplies can be reduced easily without changing the design.

- Charging Current: With scaling of technology, charging currents I_1 and I_2 follow a monotonically decreasing trend as shown in Figure 2–7. This is because as technology scales, transistor sizes reduce and therefore, the current conducted by the transistors scale down exponentially (following the well-established large-signal long-channel/short-channel current equations).

- Dynamic Power/Average Power: As dynamic/average current scales down exponentially with technology, dynamic power/average power scales down exponentially as well. This trend can be seen in Figures 2–8 and 2–9.

- Dynamic Range: The dynamic range is defined as the maximum allowable time difference between two time steps. If $t_1 < t_2$, the maximum allowable time difference between two time steps of a weighted average circuit is given

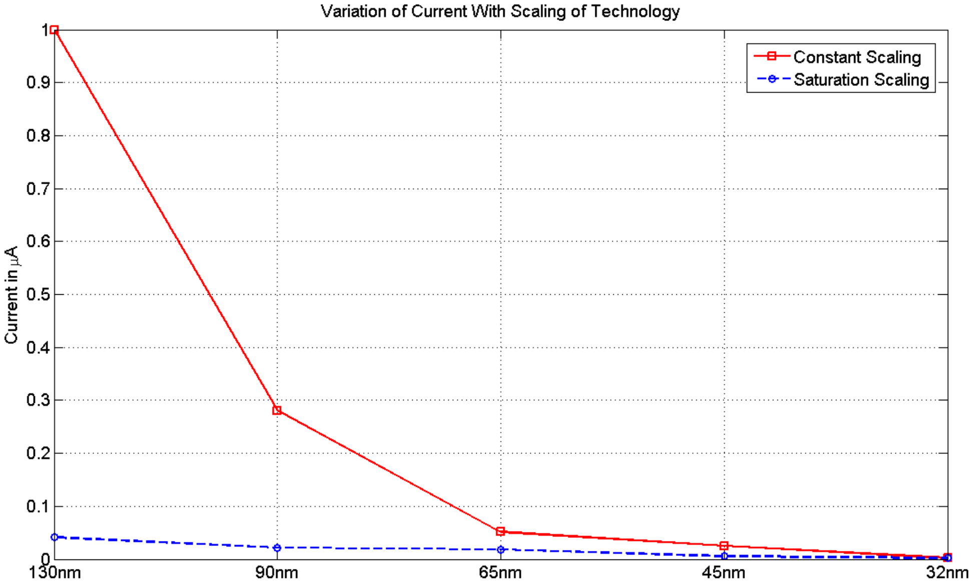

Figure 2–7. Variation of capacitor charging current with scaling technology

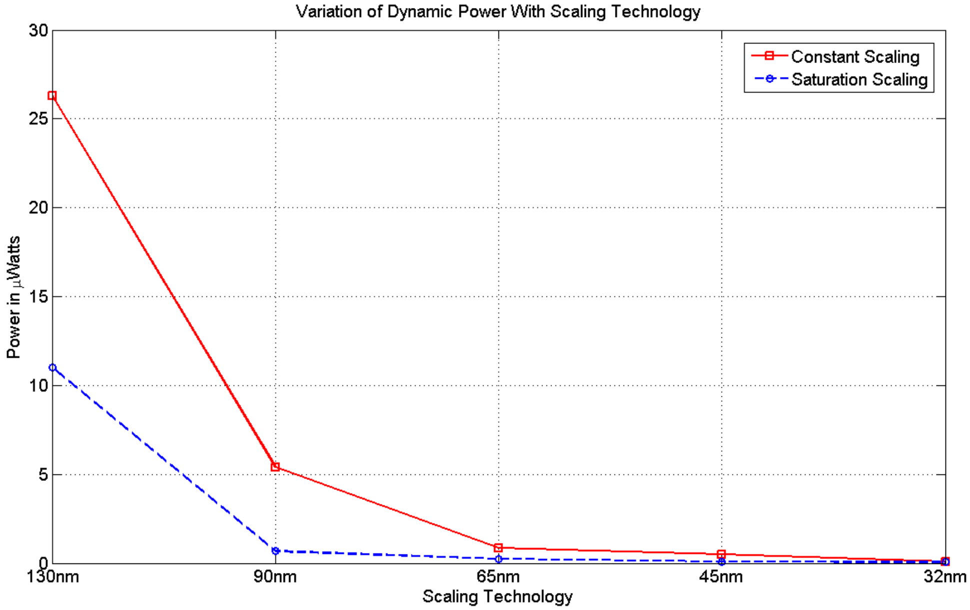

Figure 2–8. Variation of dynamic power with scaling technology

by $\frac{CV_{TH}}{I_1}$. With the capacitor C and threshold voltage V_{TH} fixed in our simulations, the dynamic range is inversely dependent on current. With an almost exponential decrease in current we would expect an exponential increase in dynamic range and the same trend can be noted in Figure 2–14.

- Energy Consumed: As power consumed by the weighted average circuit scales down exponentially with technology, so does the energy consumed by the circuit per weighted average operation. This trend can be noted from Fig. 2–10.

- Weighted Average Outputs: From Figure 2–11 comparing the simulated and calculated values of the output of the weighted average circuit, we see that the simulated results match the calculated results well. We believe that the slight difference in results as technology scales is due to inaccuracy in circuit models at low current levels.

- Output Noise: The simulated noise and calculated output noise values match closely as shown in Figure 2–12 confirming the accuracy of the derived expression for noise of this circuit. Also, noise is inversely proportional to current with all other factors remaining the same. Hence with exponential type decrease in current we see an exponential type increase in noise.

- SNR: Observations similar to those made for the output noise can also be made with the simulated and calculated SNR values as shown in Figure 2–13. The signal, which is the time taken by the weighted average circuit to produce an output, increases as we scale technology. With the noise also increasing with scaling technology we note that the ratio of signal power and noise power, the SNR, actually decreases only slightly with scaling technology. This is in accordance to what was predicted by our equations. With more detailed analysis, (with all other factors constant) we can note that the SNR is slightly dependent on the current. So as current decreases with technology, SNR also decreases. However, in Eq. 2–37, the $I(t2 - t1)$ term is much smaller than the CV_{TH} term causing SNR to be almost a constant. Also, slight variations in the simulated SNR results confirm this analysis.

For all the above mentioned performance measures, for both the transistor sizing scenarios - sizing the transistors so that the transistors are in the active region and sizing the transistors by the technology scaling factor, we see the same performance trend.

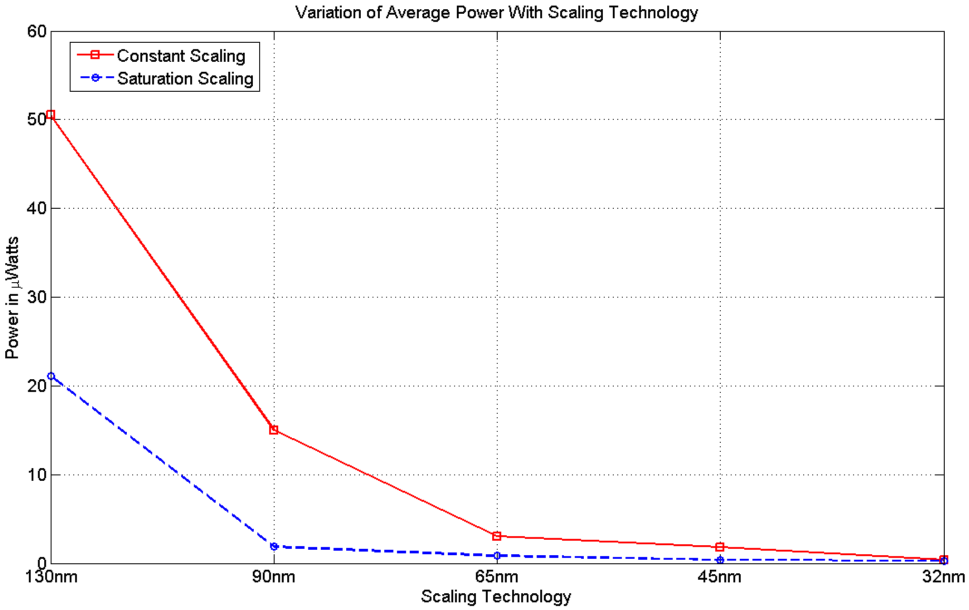

Figure 2–9. Variation of average power with scaling technology

2.3.3 Discussion

We see that the dynamic range of inputs supported by the time-mode circuits keeps increasing exponentially with scaling technology. For voltage mode and current mode circuits as technology scales the input dynamic range supported reduces (because scaling technology reduces the supply voltage and the maximum current supported by the devices). But, for time-mode circuits, as technology scales the input DR supported increases (scaling technology does not affect information stored in time). This is a very promising aspect of time-mode circuits when compared to voltage-mode and current-mode circuits.

With scaling technology, though the noise performance of the circuit gets worse the SNR stays a constant. Therefore, applications using time-mode circuits can expect the SNR performance of the time-mode circuits to remain constant with new technologies.

As technology scales, time-mode circuits become more power and energy efficient. Therefore, they can be chosen for low-power applications across different device technologies.

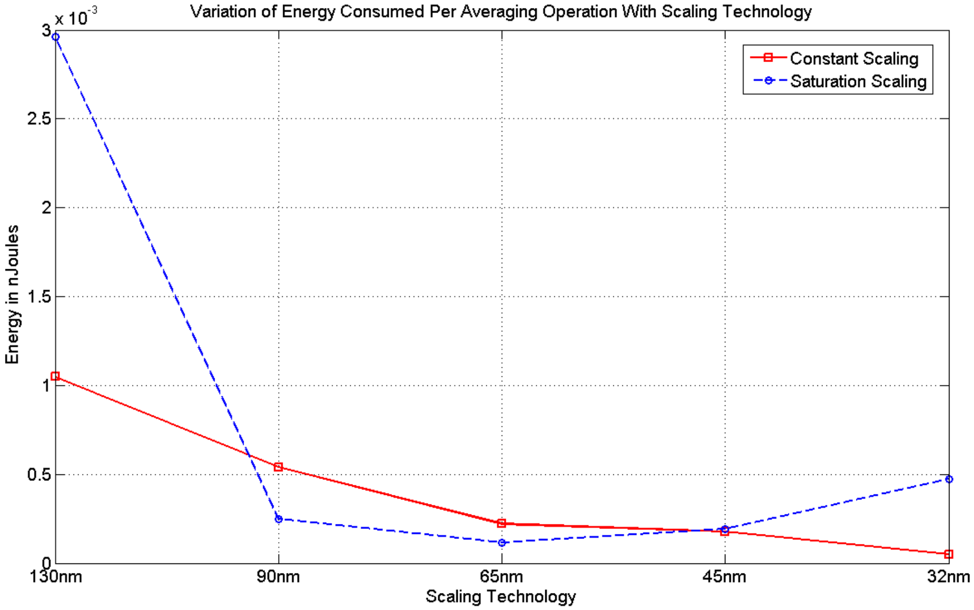

Figure 2–10. Variation of energy consumed per averaging operation with scaling technology

The results from the experiments are very encouraging and reinforce the fact that time-mode circuits would scale well with technology and show good performance.

2.3.4 Drawbacks

Time-mode circuits have shown promising results, however certain design issues can limit their performance.

- Right now the measurement of the output timing was done when the input of the comparator reached the threshold. The delay of the comparator would be another offset but this would be highly dependent on the load. When these circuits are used with larger fan out, the offset caused due to the capacitive load at the output would not remain constant.

- While the current sources are charging the capacitor the V_{DS} across the input switching transistors does not remain constant and causes variations in the

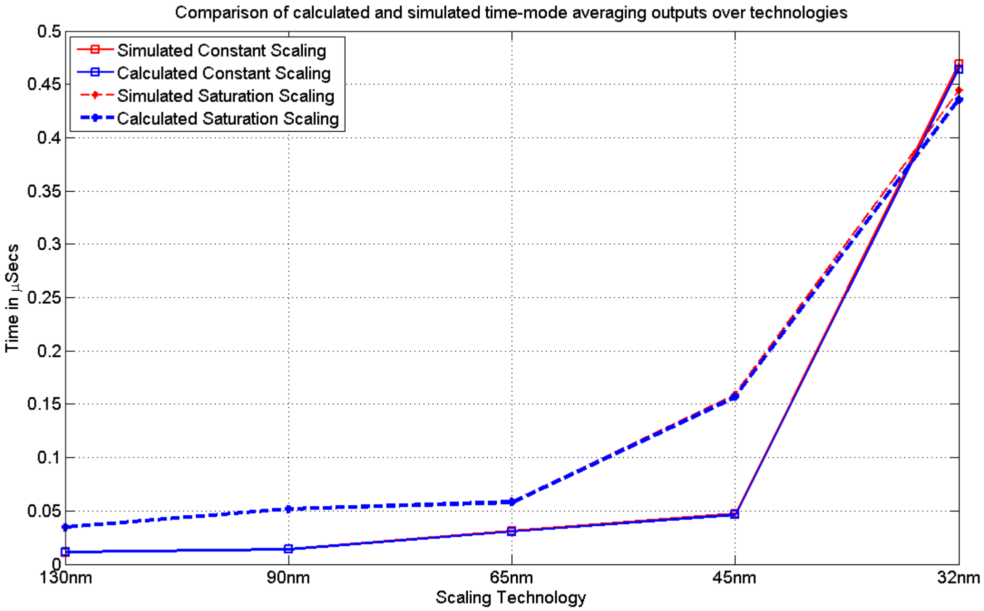

Figure 2–11. Comparison of calculated and simulated time-mode averaging outputs over technologies

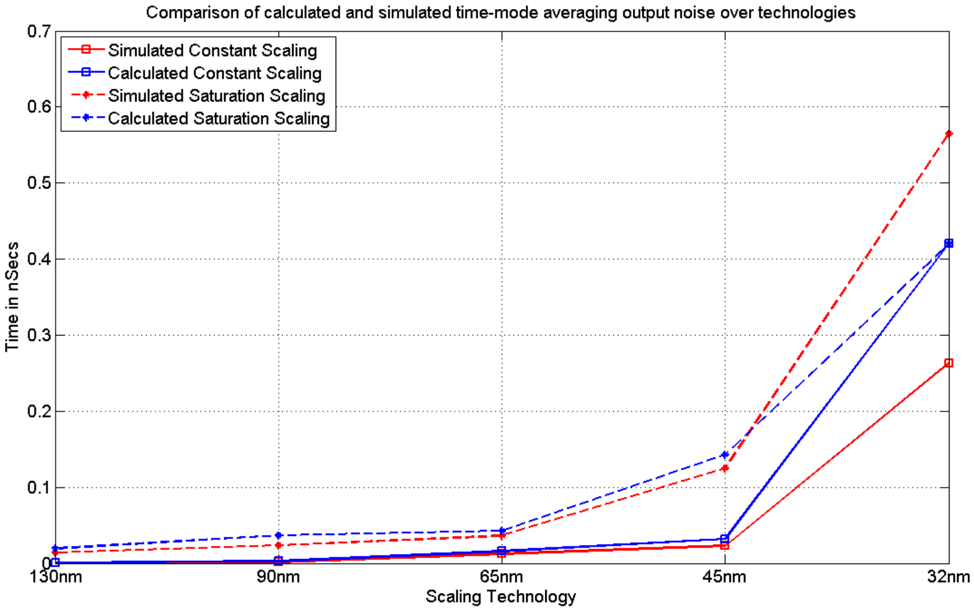

Figure 2–12. Comparison of calculated and simulated time-mode averaging output noise over technologies

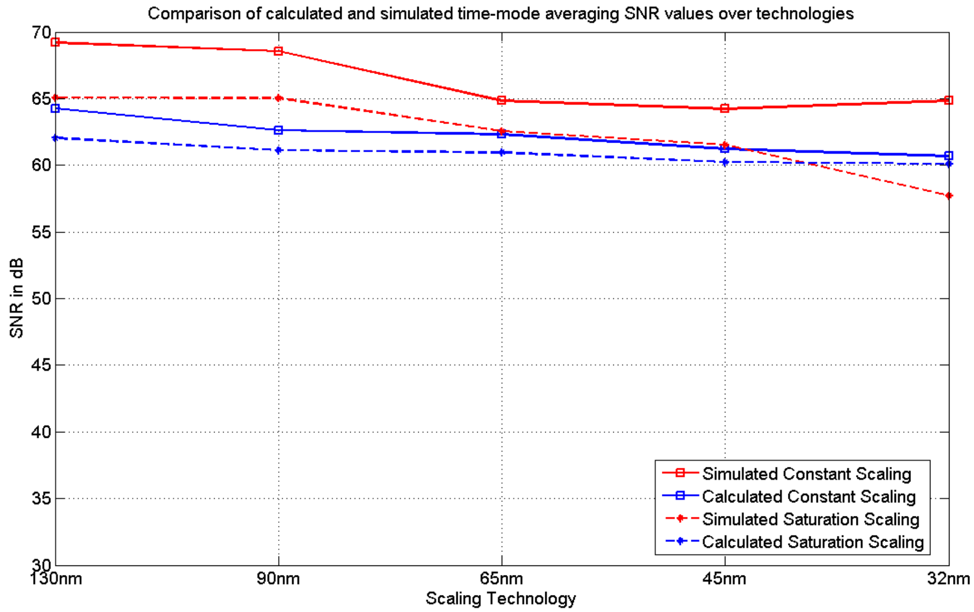

Figure 2–13. Comparison of calculated and simulated time-mode averaging SNR values over technologies

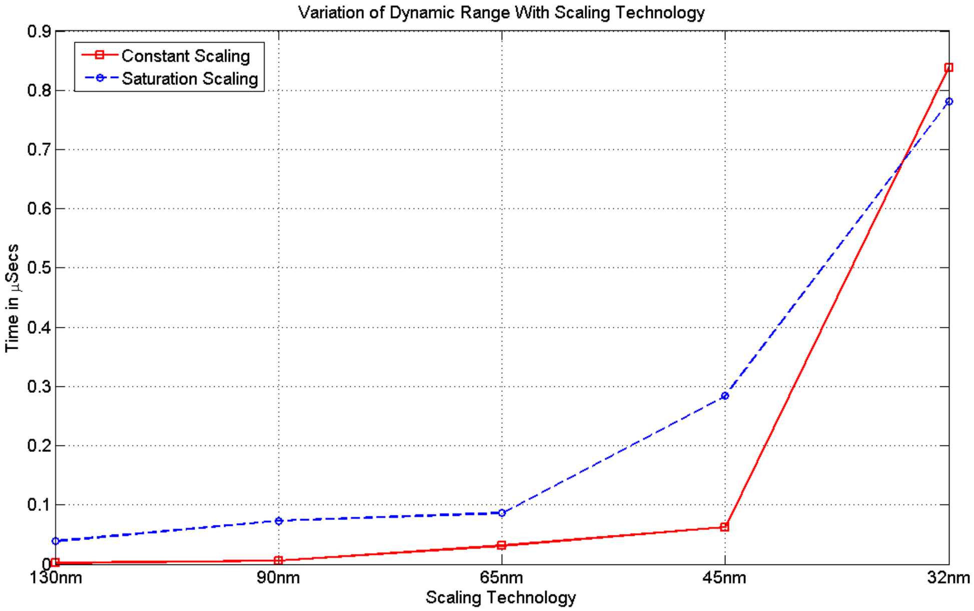

Figure 2–14. Variation of dynamic range with scaling technology

current. Having larger lengths for these transistors is critical for accurate operation.

- Input switching causes coupling between the gate and drain and we see a large spiking current initially. If the t_{OUT} to be calculated is small, this can cause problems in accuracy.

- A limitation to the accuracy of these results is due to limitations of the simulator as we try to operate at very low currents and measure up to six significant digits. This is especially true in case of noise current, where we believe the errors creep in due to numerical methods and round off. However, these precision issues do not change the trend in the results.

2.4 Carbon Nanotube Based Time-Mode Weighted Averaging Circuit

To see if the time-mode weighted averaging circuit would scale well in the nano-technology regime, we replace the traditional BSIMv3.1 AMI 0.5μm Si transistor models by carbon nanotube FET models [18] and simulate our prototype weighted average circuit. Transistors M_1, M_2, Current Sources I_1, I_2, and the inverters of the weighted averaging circuit are realized using NCNFET and PCNFET models developed by Roy et al of INAC/Purdue. The circuit is simulated using HSPICE. The NCNFET and PCNFET transistor models used in our simulations have only a single carbon nanotube representing their channels. The biggest advantage of this single carbon nanotube technology is that the transistors have extremely small gate and channel capacitances; thus, promising very high speed operation.

2.4.1 Carbon Nanotube Field Effect Transistors (CNFETs) and Their Spice Models

Carbon nanotubes are nano-diameter cylinders consisting of a single graphene sheet wrapped up to form a tube. Since their invention in the early 1990s, researchers have been actively exploring the electrical properties of these devices and their potential applications in electronics. One of the most promising applications of carbon nanotubes, the carbon nanotube transistor (CNTFET) - first reported in 1998, is currently considered as the most promising building block of a future nano-electronic era. The reason for this is not just their small size, but their inherent properties like low power dissipation, possible ballistic transport, high current densities, high mobility, low resistance and the facilitation of making transistors and interconnects using semi-conducting and metallic carbon nanotubes.

For a typical nanotube geometry of $100nm$ length and $3nm$ diameter, C is of order $4aF$. The channel resistance can be as small as $6.25k\Omega$. Therefore, the RC frequency is equal to $6.3THz$ [19]. Let us compare this frequency with the f_T of a minimum size NMOS transistor in the AMI $0.5u$ Si process. The f_T of a NMOS transistor can be roughly expressed as,

$$f_T \approx \frac{\mu}{2\pi L^2}(V_{GS} - V_{TH}) \tag{2–40}$$

with μ the mobility, L the channel length, V_{GS} the gate to source voltage and V_{TH} the threshold voltage. Substituting typical values of $\mu = 449.98cm^2/Vs$, $L = 0.6\mu m$, $V_{GS} - V_{TH} \approx 4V$ for AMI $0.5u$ Si process, we get $f_T = 80GHz$. This shows that the speed limit intrinsic to a nanotube transistor is several orders of magnitude greater than a Si transistor.

The CNFET model used in our simulations is a simplistic model that was developed to assess circuit performance of single walled semiconducting CNFETs. It is an appropriate model to evaluate delays, estimate power in circuits and simulate the performance degradation due to interconnect and device parasitics. The modeling technique used is generic in the sense that it can faithfully represent a wide range of CNFET geometries and gate materials with reasonable operating voltages and user specified temperature conditions. The model has a strong foundation on the underlying physics of operation along with necessary simplifications and assumptions. This makes a multiple-transistor circuit simulation possible.

The assumptions made to arrive at the CNFET spice model include, Bulk-type CNFETs: In the literature, two types of carbon nanotube transistors have been studied extensively. They are respectively, the Schottky barrier CNFET and the bulk-type CNFET. Though the Schottky barrier CNFET has its own advantages, the model assumes a bulk-type CNFET as this MOSFET-like device has a higher on-current and, hence, would define the upper limit of performance.

Ballistic transport: Recent experiments have demonstrated that a CNFET can typically be used in the MOSFET-like mode of operation with near ballistic transport.

2.4.2 Physics Governing the Operation of CNFET

It is a well established fact that gate voltage induces charge in the CNFET channel and also modulates the top of the energy band between the source and the drain. As the source-drain barrier is lowered, current flows between the source and the drain. Since we are dealing with ballistic transport, all scattering mechanisms are neglected.

2.4.3 Simulation Results

The rail-to-rail supply voltage used in our simulations is $0.6V$. The simulation outputs are shown in Figure 2–17. For the simulations we chose $t_1 = 1ns$, $t_2 = 3ns$, $C = 7fF$ and $V_{TH} = 0.5V$. Since the CNFET spice models are simplistic models and not ideal for analog simulations instead of the 5-transistor comparator, we chose an ideal op-amp to perform the comparator's functionality. The expected t_{OUT} and the calculated t_{OUT} values match closely and they are approximately equal to $9ns$. The small difference between the two t_{OUT} values can be attributed to the OFF current of the CNFETs charging the capacitor. The carbon nanotube transistors have very high current drive as can be seen from Figures 2–15, 2–16 and from Figure 2–18 we can see that the off-current of these carbon nanotubes is also high (in the order of 30nA). These high off-currents can produce an offset that introduces some jitter in the output. In spite of the large off currents, the average power consumed by the circuit is $0.33\mu W$.

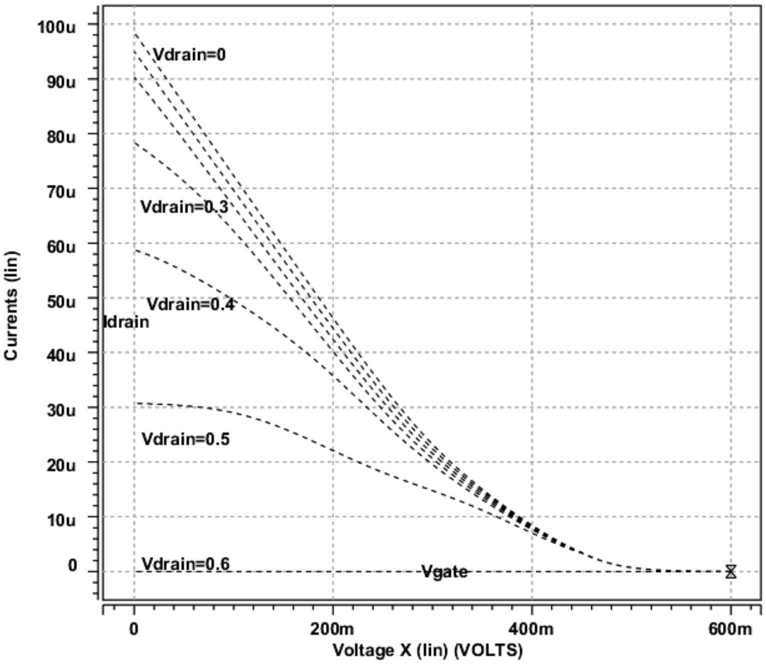

Figure 2–15. PCNFET I_D-V_{GS} plots for varying V_{DS}

2.4.4 Discussion

We did not plot the performance results of these carbon nanotube transistor based time-mode circuits with the Si technology scaling curves discussed in the previous section because these CNFET models are totally different from the PTM models used for all the Si processes. But, it would be extremely useful if we can still compare the performance numbers obtained from the scaling Si simulations and the CNFET based circuit's simulations. The carbon nanotube transistor based time-mode circuits have parasitic capacitances in the range of aF (compared to fF for the silicon based transistors), have high current drive - therefore high speed (because of the high current drive of carbon nanotube transistors) and are very power efficient (power even lower than the corresponding 32nm process based averaging circuit's power). Since nano-technology promises very attractive features

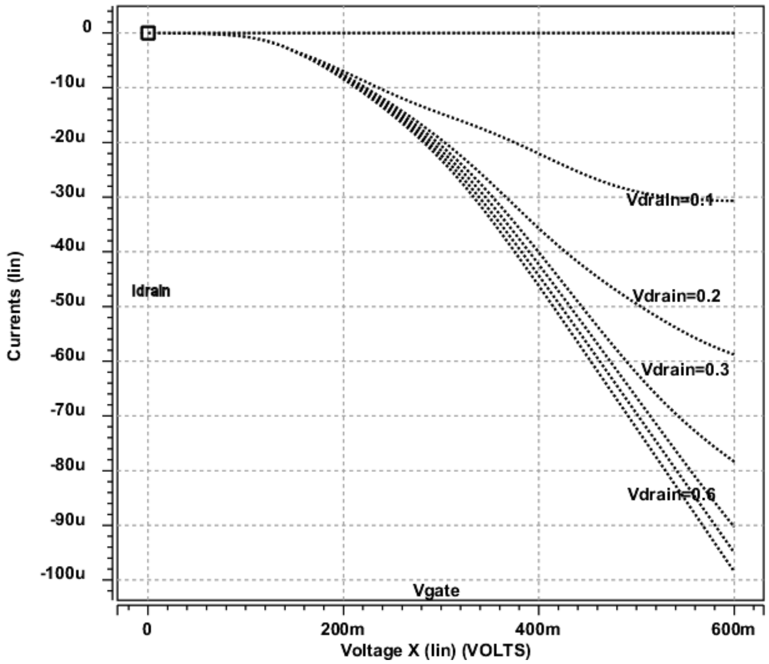

Figure 2–16. NCNFET I_D-V_{GS} plots for varying V_{DS}

and performance for time-mode circuits, we can safely say that time-mode circuits scale well into the future technologies.

2.5 Reliable Time-Mode Weighted Average Circuit

2.5.1 Motivation

The reliability of integrated circuits is a major concern for the electronics industry and becomes more of a concern as processes scale down to deep sub-micron CMOS and future nanotechnologies [20]. As these process technologies become more and more complex, higher levels of integration used in the ICs will increase the chip failure rate. These failures underscore the importance of reliability for manufacturing of nano-scale systems. It is, therefore, imperative that circuits are designed with reliability in mind.

Construction of reliable digital systems with the use of redundant components was first considered by Von Neumann for certain cases of intermittent failures of elements [21]. His ground-breaking work was extended by Dickinson and Walker for the case of permanent failures of logic elements [22]. But the works of Von

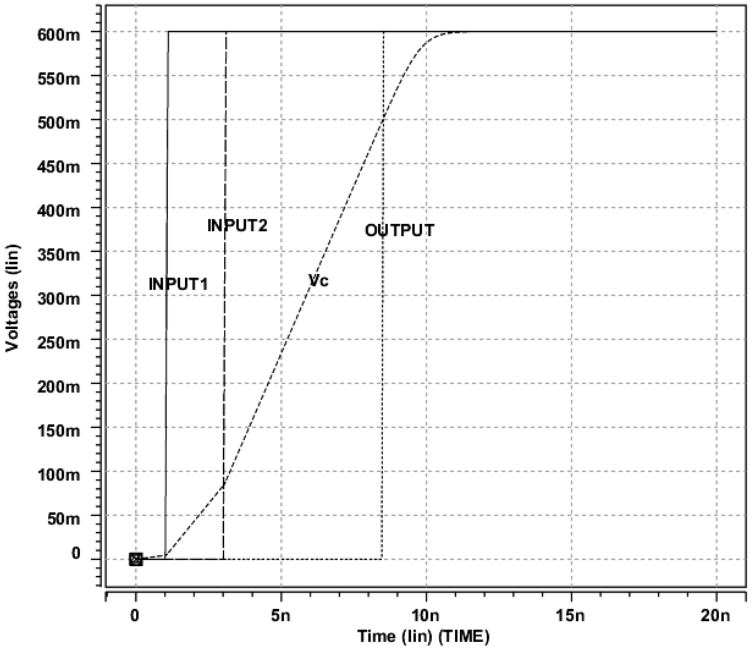

Figure 2–17. Nano-weighted average circuit simulation outputs

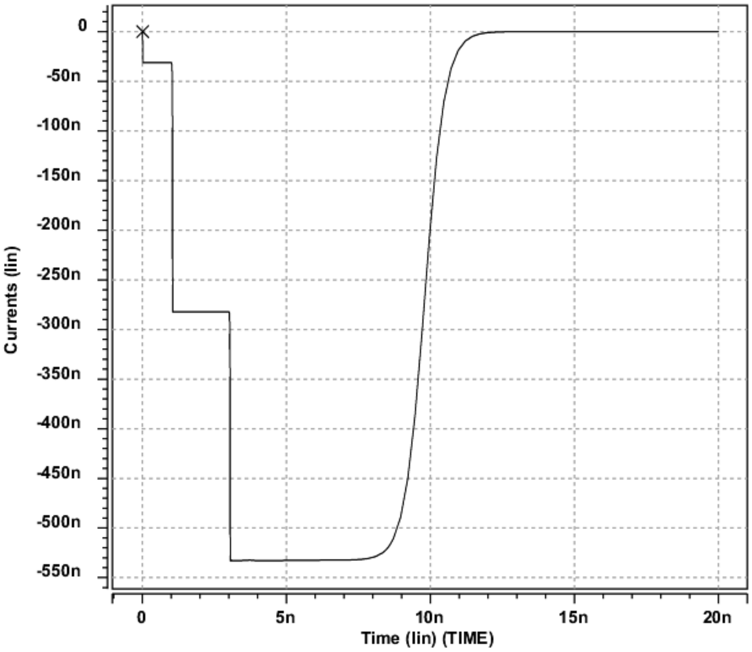

Figure 2–18. Capacitor charging/discharging current in a nano-weighted average circuit

Neumann, Dickinson and Walker and many others were all dedicated to improving the reliability of digital circuits. In this chapter, we discuss the design of reliable analog nanocomputational circuits using redundancy. As an example, we will explain the design of a reliable analog time-mode weighted average circuit.

2.5.2 Time-Mode Median Circuit

Figure 2–19 illustrates the basic elements used to find the median of an odd-number of input temporal signals. In general, the circuit can process many input steps, but only three are shown here for simplicity. The circuit consists of an inverter, a current source of value I and a PMOS transistor for each input and a current source of value $\frac{3I}{2}$ connected to the drains of the transistors M_1, M_2 and M_3. The rising edges of the input steps correspond to the time values t_1, t_2 and t_3 which represent three input values. The PMOS transistors M_1, M_2 and M_3 act as switches.

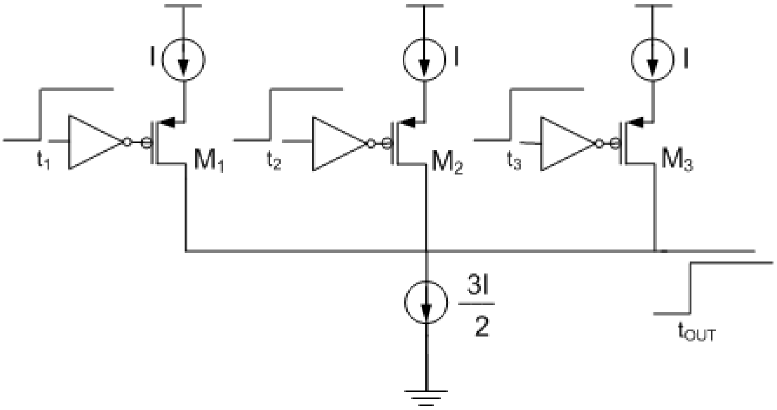

Figure 2–19. Time-mode median circuit for 3-inputs

To aid the explanation of the operation of this circuit, we assume that $t_1 < t_2 < t_3$ though such a condition is not necessary for the operation of the circuit. Since the input step making its low-high transition at time t_1 enters the median block first, it switches transistor M_1 on. Since the current source of value $\frac{3I}{2}$ is discharging the parasitic capacitance C_P, there won't be any charge built up across the parasitic capacitance. However, when the second step enters the block at time t_2, a net current of $\frac{I}{2}$ charges the parasitic capacitance and starts adding charge to the capacitor and we will get an output step at time t_{OUT}. Since the parasitic capacitance gets quickly charged after the second step enters the median block,

$$t_{OUT} \approx t_2 \tag{2–41}$$

Thus, we see that the output obtained in Eq. 2–41 is the median of the three inputs t_1, t_2 and t_3. In general, this median circuit can process N-input steps - provided that N is odd. The circuit is shown in Figure 2–20. Depending on the value of N, the value of the current source that pulls-down the capacitor's voltage is chosen to be $\frac{NI}{2}$. It is to be noted that using conventional voltage-mode and current-mode analog circuit designs it is difficult to design such a simple circuit to perform a median operation among various inputs.

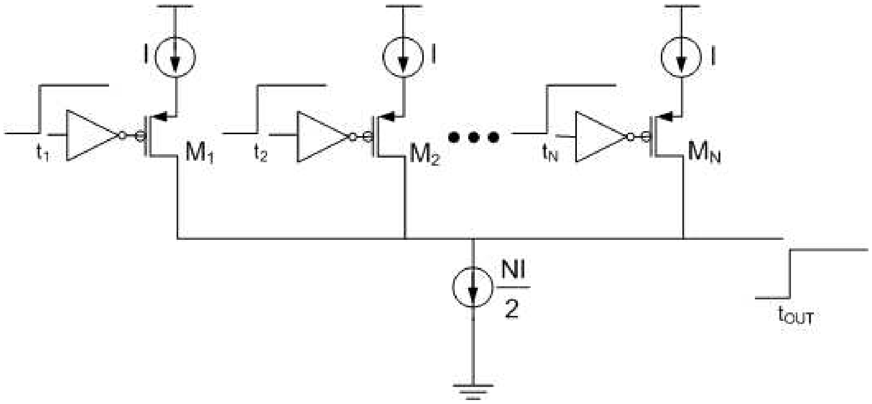

Figure 2–20. Time-mode median circuit for N-inputs

2.5.3 Redundancy in Time-Mode Computation

We will explain the concept of improving reliability using redundancy through the design of a reliable analog time-mode weighted average circuit that has an architecture similar to Von Neumann's 2-out-of-3 majority circuit shown in Figure 2–21 and perform analysis to quantify its reliability.

Von Neumann's 2-out-of-3 majority circuit shown in Figure 2–21 uses a majority circuit fed by three independent devices which operate from the same

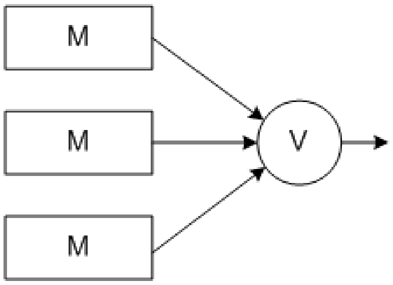

Figure 2–21. Von Neumann's two-out-of-three majority circuit

source of input information [21]. Dickinson and Walker analyzed the circuit in detail and proved that the circuit has a resultant reliability greater than that of its elements [22]. As mentioned above, their work was only applicable to digital circuits. Here, we use their concept to improve the reliability of analog circuits. We will extend the work of Dickinson and Walker to design a reliable analog time-mode weighted average circuit as shown in Figure 2–22. Von Neumann's 2-out-of-3 majority circuit is essentially used polling between inputs and can only be used for digital applications. Therefore, it is being replaced by a time-mode median circuit as shown in Figure 2–19. For our failure analysis, we assume that the median circuit never fails. The same assumption is made for the digital voting circuits discussed above. The only failures to be considered are those of the three elements (weighted average blocks) which feed the median circuit.

The time-mode weighted average block has components like current sources, digital switches, comparator and a capacitor. It is possible for any of these components to fail and introduce errors in the output of the circuit. For explanation purposes, let the output of the weighted average circuit when there are no failures in its components be t_{out}^{ideal} and the output when there are some failures be $t_{out}^{obtained}$.

The weighted average blocks can fail in two modes:

1. $t_{out}^{obtained} < t_{out}^{ideal}$. This also includes the case where due to failure there is no output from the weighted average block ($t_{out}^{obtained}$ is close to infinity).

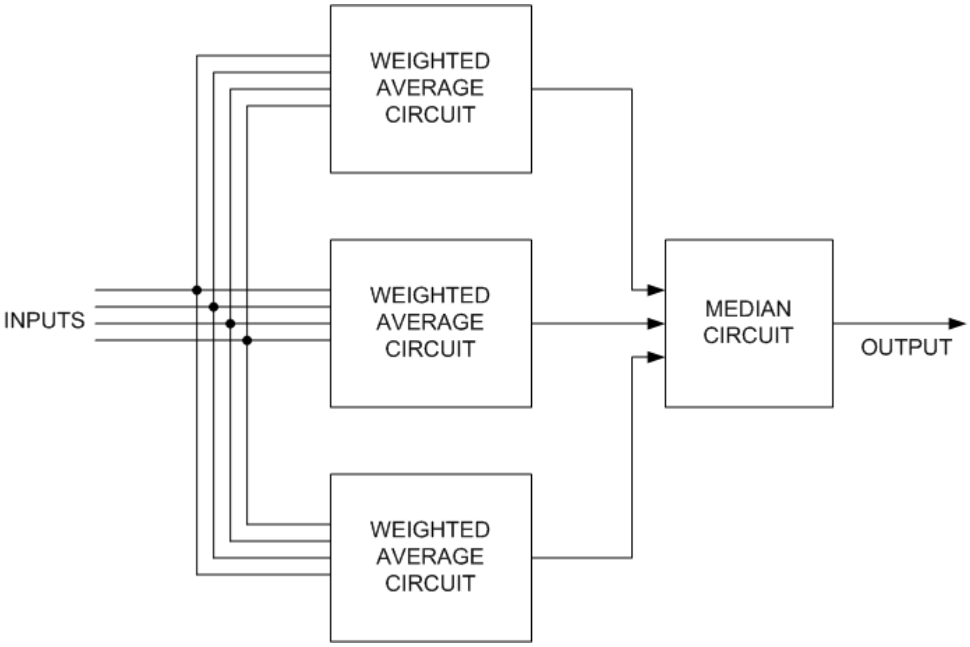

Figure 2–22. Block diagram of a reliable time-mode weighted average circuit

2. $t_{out}^{obtained} > t_{out}^{ideal}$. This also includes the case where due to failure, the weighted average block fires an output immediately after its internal nodes are reset (the reset stage is not shown in the figure) - ($t_{out}^{obtained}$ is close to zero).

Let us assume that the probability that any weighted average block will function correctly is R_0. The probability that the redundant system is not going to fail R_1 is given by the sum of the three cases mentioned below:

1. All the three time-mode weighted average blocks function correctly. The probability that the redundant system is not going to fail in this case is given by R_0^3.

2. One of the weighted average blocks fail in any of the two modes - ($t_{out}^{obtained} > t_{out}^{ideal}$) or ($t_{out}^{obtained} < t_{out}^{ideal}$) mentioned earlier and the other two function correctly, in which case the output of the system would still be correct. The probability for this case is given by $(1 - R_0)R_0^2$. Since this case can happen in three different ways, the total probability for this case is $3(1 - R_0)R_0^2$.

3. Two of the three weighted average blocks fail in this case but we assume that the elements have equal probability of failing in either of the modes. That means that there is a probability of $\frac{1}{2}$ that the two weighted average blocks will fail in opposite directions (one block firing output early and the other firing output late), in which case the output of the redundant system would still be correct. The probability that two elements fail and the others still

function correctly is $(1 - R_0)^2 R_0$ and this can happen in three different ways. Therefore, the probability for this case is given by $\frac{3}{2}(1 - R_0)^2 R_0$.

Therefore, the total probability that the redundant system is not going to fail R_1 is given by the sum of the probabilities obtained in the above mentioned three cases [22]:

$$\begin{aligned} R_1 &= R_0^3 + 3(1 - R_0)R_0^2 + \frac{3}{2}(1 - R_0)^2 R_0 \\ &= \frac{3}{2}R_0 - \frac{1}{2}R_0^3 \end{aligned} \qquad (2\text{–}42)$$

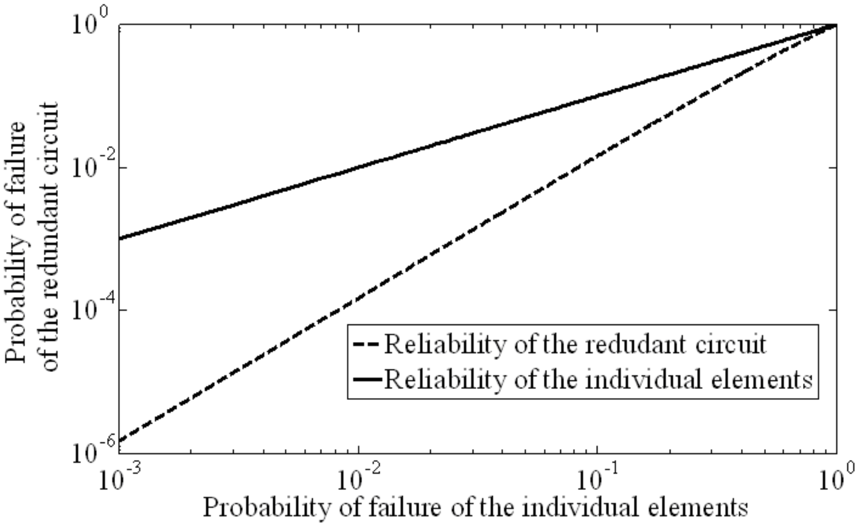

Figure 2–23. Plot showing the increase in reliability of the redundant circuit as compared to the individual elements

From the result shown in Eq. 2–42, we see that the redundant time-mode weighted average circuit is always more reliable than the individual elements. This can also be realized from the reliability curve in Figure 2–23. As shown in that figure, there is a considerable improvement in the reliability of the redundant circuit as compared to the reliability of the individual elements.

2.5.4 Discussion

We see that for the above 2 cases, redundancy provides additional reliability for the weighted average circuit. Many circuit designers would be concerned that such redundancy would increase the chip area. But, most of the real time applications would compromise on the chip area than on the reliability of the circuits. Also, this redundant weighted average circuit can be seen as a stepping stone towards improving the reliability of nanocomputational circuits.

In Chapter 2, we will see how the performance of the time-mode weighted averaging circuit with its voltage-mode and current-mode counterparts.

CHAPTER 3
SNR COMPARISON OF WEIGHTED AVERAGING CIRCUITS

We have quantified the performance of time-mode circuits in terms of key measures such as SNR, DR and power consumption. These performance metrics are not clear-cut. For instance, dynamic range is a well-defined concept in voltage-mode and current-mode but must be carefully considered for some time-mode circuits whose inputs can be arbitrarily large.

We need to compare the performance measures such as SNR and DR of time-mode circuits to corresponding voltage-mode and current-mode circuits by making a *ceteris paribus* (other things being equal) comparison. Since it is difficult to compare all possible voltage-mode, current-mode and time-mode computation circuits, we would like to start by restricting ourselves to the comparison of weighted average circuits shown in the Figures 3–1, 3–3 and the time-mode weighted averaging circuit discussed in Chapter 2. We will quantize their SNR and DR, compare their performances, and comment on them. The main criteria for the choice of these voltage and current mode weighted average circuits are low complexity and low power consumption. The choice would enable us to perform a fair comparison with the basic two-input time-mode weighted average circuit.

The first circuit operating in voltage-mode computes

$$V_{OUT} = \frac{g_1 V_1 + g_2 V_2}{g_1 + g_2} \tag{3–1}$$

where g_1 and g_2 represent the transconductances of the two OTAs in the circuit. The transconductances are set by the individual bias voltages applied to the OTAs. V_1, V_2 are the two input voltages and V_{OUT} is the output voltage of the circuit.

The second circuit operating in current-mode computes

$$I_{OUT} = \frac{e^{k_1} I_1 + e^{k_2} I_2}{e^{k_1} + e^{k_2}} \tag{3–2}$$

where I_1, I_2 are the two input currents and V_{OUT} is the output current of the circuit. k_1, k_2 are the voltages applied to the transistors in the circuit. These voltages contribute to the weights e^{k_1} and e^{k_2} applied by the circuit to compute the weighted average.

And, as we have seen in Chapter 2, the time-mode weighted averaging circuit computes

$$t_{OUT} = \frac{I_1 t_1 + I_2 t_2}{I_1 + I_2} + \frac{C V_{TH}}{I_1 + I_2} \tag{3–3}$$

As mentioned earlier, though we can come up with more efficient circuits, the voltage-mode, current-mode and time-mode averaging circuits compared in this paper are chosen such that their circuit architecture is extremely simple and consume very low power.

To quantify the SNR relations obtained for voltage-mode, current-mode and time-mode averaging circuits, let's make the following assumptions.

- no load capacitance is connected at the output node.
- for simplicity, we assume that all the transistors involved in the analyses have the same dimensions: $W = 6\mu m$ and $L = 20\mu m$.
- the transistors operate at room temperature.
- process parameters of AMI 0.5μ process are used.

3.1 Voltage-Mode Averaging Circuit

Figure 3–1 illustrates the basic elements used to perform a weighted average of voltage-mode signals V_1 and V_2. It consists of two transconductance amplifiers G_1 and G_2 connected in unity feedback configurations.

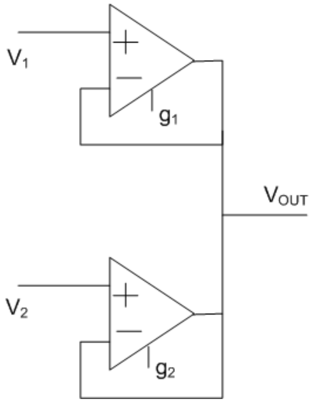

Figure 3–1. Voltage mode weighted averaging circuit

The output of the circuit is given by the equation:

$$V_{OUT} = \frac{g_1V_1 + g_2V_2}{g_1 + g_2} \tag{3–4}$$

For SNR calculations, we need to define a reference for the inputs and outputs. Let us define the input V_1 as the reference.

Now, V_{OUT} defined with respect to the reference would be given by,

$$\begin{aligned}\hat{V}_{OUT} &= V_{OUT} - V_1 \\ &= (\frac{g_1V_1 + g_2V_2}{g_1 + g_2}) - V_1 \\ &= (\frac{g_1V_1 + g_2V_2 - g_1V_1 - g_2V_1}{g_1 + g_2}) \\ &= \frac{g_2(V_2 - V_1)}{g_1 + g_2} \end{aligned} \tag{3–5}$$

There are two noises sources in this circuit as shown in Figure 3–2.

1. $\overline{\Delta v_1{}^2}$ - noise due to operational transconductance amplifier g_1 referred to its positive input.

2. $\overline{\Delta v_2{}^2}$ - noise due to operational transconductance amplifier g_2 referred to its positive input.

Since these two noise sources are not correlated, we can derive the individual contribution of each of these noise sources at the output and add up the contributions by applying superposition.

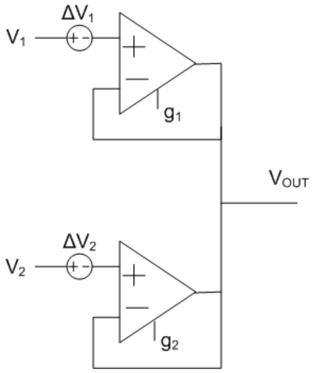

Figure 3–2. Voltage mode weighted averaging circuit with noise sources

3.1.1 Noise Contribution at the Output due to $\overline{\Delta v_1}^2$

At one particular instant in time, if the instantaneous noise of OTA1 is Δv_1, then the instantaneous noise at the output is give by,

$$\begin{aligned}\Delta V_{OUT1} &= \frac{g_2(V_2-(V_1+\Delta v_1))}{g_1+g_2} - \frac{g_2(V_2-V_1)}{g_1+g_2} \\ &= -(\frac{g_2 \Delta v_1}{g_1+g_2}) \qquad (3\text{–}6)\end{aligned}$$

The variance of the noise at the output is given by,

$$\overline{\Delta V_{OUT1}}^2 = \frac{g_2^2 \overline{\Delta v_1}^2}{(g_1+g_2)^2} \qquad (3\text{–}7)$$

Similar calculations are done for the noise contribution from OTA2.

3.1.2 Noise Contribution at the Output due to $\overline{\Delta v_2}^2$

At one particular instant in time, if the instantaneous noise of OTA2 is Δv_2, then the instantaneous noise at the output is give by,

$$\begin{aligned}\Delta V_{OUT2} &= \frac{g_2((V_2+\Delta v_2)-V_1)}{g_1+g_2} - \frac{g_2(V_2-V_1)}{g_1+g_2} \\ &= (\frac{g_2 \Delta v_2}{g_1+g_2}) \qquad (3\text{–}8)\end{aligned}$$

The variance of the noise at the output is given by,

$$\overline{\Delta V_{OUT2}}^2 = \frac{g_2^2 \overline{\Delta v_2}^2}{(g_1+g_2)^2} \qquad (3\text{–}9)$$

Therefore, the total noise contribution at the output due to the two noise sources is given by,

$$\overline{\Delta V_{OUT2}}^2 = \frac{g_2^2(\overline{\Delta v_1}^2 + \overline{\Delta v_2}^2)}{(g_1 + g_2)^2} \tag{3–10}$$

Assuming that the OTAs are basic 5-transistor differential input/single ended output OTAs, we would have noise contributions from the input transistors and the mirror transistors. Neglecting flicker noise of these transistors (valid for intermediate and high frequencies) and taking only thermal noise into our calculations, the input referred noise variances are given by:

$$\overline{\Delta V_{OUT1}}^2 = 4(\frac{8kT}{3g_1})\Delta f \tag{3–11}$$

$$\overline{\Delta V_{OUT2}}^2 = 4(\frac{8kT}{3g_2})\Delta f \tag{3–12}$$

3.1.3 Noise Bandwidth

The voltage-mode weighted averaging circuit has a pole at its output that occurs at

$$f_c = \frac{1}{2\pi R_{OUT} C_{OUT}} \tag{3–13}$$

where, C_{OUT} is the capacitance at the output node, usually defined by the load capacitance C_L.

$$R_{OUT} = \frac{1}{g_1}||\frac{1}{g_2} = \frac{1}{g_1 + g_2} \tag{3–14}$$

Hence,

$$f_c = \frac{1}{2\pi \frac{1}{g_1+g_2} C_{OUT}} = \frac{g_1 + g_2}{2\pi C_{OUT}} \tag{3–15}$$

If an amplifier has just one pole at f_c, then the noise bandwidth is given by

$$\Delta f = \frac{\pi}{2} f_c = \frac{\pi}{2}(\frac{g_1 + g_2}{2\pi C_{OUT}}) = \frac{g_1 + g_2}{4C_{OUT}} \tag{3–16}$$

The signal-to-noise ratio is given by

$$\begin{aligned}
SNR &= \frac{\frac{g_2^2(V_2-V_1)^2}{(g_1+g_2)^2}}{\frac{g_2^2(\overline{\Delta v_1{}^2}+\overline{\Delta v_2{}^2})}{(g_1+g_2)^2}} \\
&= \frac{(V_2-V_1)^2}{(\overline{\Delta v_1{}^2}+\overline{\Delta v_2{}^2})} \\
&= \frac{(V_2-V_1)^2}{4(\frac{8kT}{3g_1}+\frac{8kT}{3g_2})\Delta f} \\
&= \frac{(V_2-V_1)^2}{4(\frac{8kT}{3g_1}+\frac{8kT}{3g_2})(\frac{g_1+g_2}{4C_{OUT}})} \\
&= \frac{(V_2-V_1)^2}{4\frac{8kT}{3}(\frac{g_1+g_2}{g_1g_2})(\frac{g_1+g_2}{4C_{OUT}})} \\
&= \frac{3}{8kT}\frac{(V_2-V_1)^2 g_1 g_2 C_{OUT}}{(g_1+g_2)^2} \qquad (3\text{–}17)
\end{aligned}$$

For an averaging circuit, $g_1 = g_2$ and the SNR relation becomes,

$$SNR = \frac{3}{32kT}(V_2-V_1)^2 C_{OUT} \qquad (3\text{–}18)$$

For AMI 0.5μ process, maximum value of $V_2 - V_1$ that could be achieved $= 3.5V$ (thought the rail-to-rail voltage is 5V, to maintain the transistors of the OTA in saturation the input voltage swing would be lower). Also, through hand calculations we found out that the output node capacitance is $0.4pF$. Substituting all the values to Eq. 3–18, we get maximum SNR of $80dB$.

3.2 Current-Mode Averaging Circuit

Figure 3–3 illustrates the circuit that performs weighted average of currents I_1 and I_2. In this circuit, transistors M1-M4 operate in the sub-threshold region and they are in saturation.

We assume that in all the sub-threshold current equations below that $\kappa = 1$. By using KCL at nodes 1 and 2 in Figure 3–3, we can write

$$\begin{aligned}
I_1 &= I_S e^{\frac{K_2-V_A}{V_T}} + I_S e^{\frac{K_1-V_A}{V_T}} \\
&= I_S e^{\frac{-V_A}{V_T}}[e^{\frac{K_1}{V_T}} + e^{\frac{K_2}{V_T}}] \qquad (3\text{–}19)
\end{aligned}$$

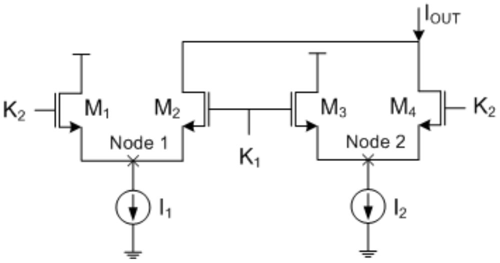

Figure 3–3. Current mode weighted averaging circuit

and

$$\begin{aligned} I_2 &= I_S e^{\frac{K_2 - V_B}{V_T}} + I_S e^{\frac{K_1 - V_B}{V_T}} \\ &= I_S e^{\frac{-V_B}{V_T}} [e^{\frac{K_1}{V_T}} + e^{\frac{K_2}{V_T}}] \end{aligned} \tag{3–20}$$

From Eqs. 3–19 and 3–20, we get

$$\frac{I_1}{I_2} = \frac{e^{\frac{-V_A}{V_T}}}{e^{\frac{-V_B}{V_T}}} \tag{3–21}$$

The output current is given by

$$I_{OUT} = I_S e^{\frac{K_1 - V_A}{V_T}} + I_S e^{\frac{K_2 - V_B}{V_T}} \tag{3–22}$$

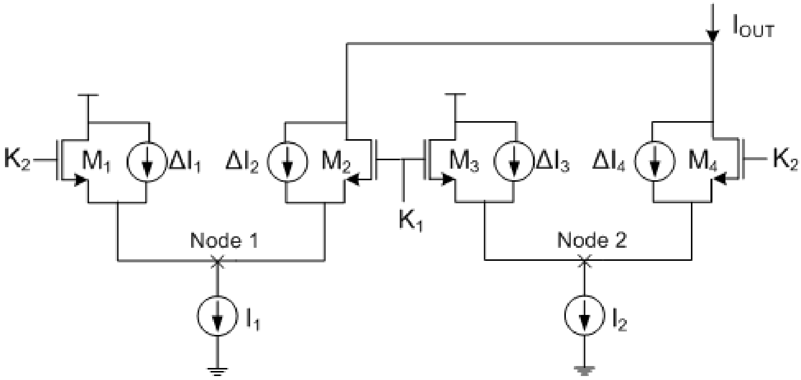

Figure 3–4. Current mode weighted averaging circuit with noise sources

Substituting Eq. 3–21 in Eq. 3–22, we get

$$\begin{aligned} I_{OUT} &= I_S e^{\frac{-V_A}{V_T}} [e^{\frac{K_1}{V_T}} + \frac{I_2}{I_1} e^{\frac{K_2}{V_T}}] \\ &= I_S e^{\frac{-V_A}{V_T}} \frac{[I_1 e^{\frac{K_1}{V_T}} + I_2 e^{\frac{K_2}{V_T}}]}{I_1} \end{aligned} \tag{3–23}$$

Using Eq. 3–19 in Eq. 3–23, we get

$$I_{OUT} = \frac{I_1 e^{\frac{K_1}{V_T}} + I_2 e^{\frac{K_2}{V_T}}}{e^{\frac{K_1}{V_T}} + e^{\frac{K_2}{V_T}}} \tag{3–24}$$

Figure 3–4 shows the current mode weighted averaging circuit with noise sources. As shown in the figure, there is noise associated with each transistor in the circuit and the equivalent noise variances can be represented using current sources connected in parallel to the transistors. The noise current source ΔI_1 sees the source resistance $\frac{1}{g_{m1}}$ of transistor M_1 and the source resistance $\frac{1}{g_{m2}}$ of transistor M_2 as shown in Figure 3–5. Since $K_1 = K_2$, $g_{m1} = g_{m2}$. Therefore, half of the current ΔI_1 flows through transistor M_2 and contributes to noise in the output current I_{OUT}. Similarly only half of noise currents ΔI_2, ΔI_3 and ΔI_4 contributes to output noise.

Therefore, the total output noise current is given by

$$\Delta I_{OUT} = \frac{\Delta I_1 + \Delta I_2 + \Delta I_3 + \Delta I_4}{2} \tag{3–25}$$

The variance in the output noise current is given by

$$\overline{\Delta I_{OUT}^2} = \frac{\overline{\Delta I_1^2} + \overline{\Delta I_2^2} + \overline{\Delta I_3^2} + \overline{\Delta I_4^2}}{4} \tag{3–26}$$

Neglecting flicker noise, the noise current of a transistor operating in the subthreshold region is given by $2KTg_m$. Substituting this noise current expression in Eq. 3–26, we get

$$\begin{aligned}\overline{\Delta I_{OUT}}^2 &= \frac{\overline{\Delta I_1}^2 + \overline{\Delta I_2}^2 + \overline{\Delta I_3}^2 + \overline{\Delta I_4}^2}{4} \\ &= \frac{(2KTg_{m1} + 2KTg_{m2})\Delta f_1 + (2KTg_{m3} + 2KTg_{m4})\Delta f_2}{4} \qquad (3\text{–}27)\end{aligned}$$

where the noise currents of transistors M_1 and M_2 would have a noise bandwidth determined by R-C time constant of node 1 and noise currents of transistors M_3 and M_4 would have a noise bandwidth determined by R-C time constant of node 2.

Since $K_1 = K_2$, for subthreshold transistors $g_{m1} = g_{m2}$. Similarly, $g_{m3} = g_{m4}$. Therefore,

$$\overline{\Delta I_{OUT}}^2 = (KTg_{m1})\Delta f_1 + (KTg_{m3})\Delta f_2 \qquad (3\text{–}28)$$

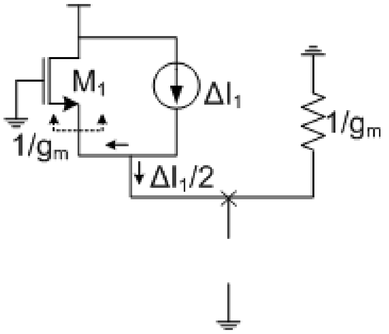

Figure 3–5. Half of the noise current from each transistor flows to the output

The pole contributed by node 1 is given by:

$$f_{node1} = \frac{1}{2\pi \frac{1}{g_{m1}||g_{m2}} C_{node1}} = \frac{g_{m1}}{\pi C_{node1}} \qquad (3\text{–}29)$$

Noise bandwidth for noise currents of transistors M_1 and M_2 is given by,

$$\Delta f_1 = \frac{\pi}{2} f_{node1} = \frac{\pi}{2}\left(\frac{g_{m1}}{\pi C_{node1}}\right) = \frac{g_{m1}}{2C_{node1}} \qquad (3\text{–}30)$$

The pole contributed by node 2 is given by:

$$f_{node2} = \frac{1}{2\pi \frac{1}{g_{m3}||g_{m4}} C_{node2}} = \frac{g_{m3}}{\pi C_{node2}} \qquad (3\text{–}31)$$

Noise bandwidth for noise currents of transistors M_3 and M_4 is given by,

$$\Delta f_2 = \frac{\pi}{2} f_{node2} = \frac{\pi}{2}(\frac{g_{m3}}{\pi C_{node2}}) = \frac{g_{m3}}{2C_{node1}} \tag{3–32}$$

as parasitic capacitance $C_{node1} = C_{node2}$.

The variance in the output noise current is given by,

$$\begin{aligned} \overline{\Delta I_{OUT}}^2 &= (KTg_{m1})\frac{g_{m1}}{2C_{node1}} + (KTg_{m3})\frac{g_{m3}}{2C_{node2}} \\ &= \frac{kT}{2C_{node1}}(g_{m1}^2 + g_{m2}^2) \\ &= \frac{kT}{2C_{node1}V_T^2}(I_1^2 + I_2^2) \end{aligned} \tag{3–33}$$

Since the discussions would get too complex, let us just focus our discussions here to current mode averaging functionality. Lets assume that $K_1 = K_2$. The output I_{OUT} in this case is given by $I_{OUT} = \frac{I_1+I_2}{2}$. The output referred to the first input I_1 is given by,

$$\begin{aligned} \hat{I}_{OUT} &= I_{OUT} - I_1 \\ &= (\frac{I_1 + I_2}{2}) - I_1 \\ &= \frac{I_1 - I_2}{2} \end{aligned} \tag{3–34}$$

The signal-to-noise ratio is given by

$$\begin{aligned} SNR &= \frac{\frac{(I_2-I_1)^2}{4}}{\frac{kT}{2C_{node1}V_T^2}(I_1^2 + I_2^2)} \\ &= \frac{V_T^2 C_{node1}(I_2 - I_1)^2}{2kT(I_1^2 + I_2^2)} \end{aligned} \tag{3–35}$$

To quantify the SNR equation, we substituted these nominal values: $C_{node1} = 0.4pF$ (as in the voltage-mode case), $I_1 = 1nA$ (low sub-threshold current) and $I_2 = 20nA$ (high subthreshold current) in Eq. 3–35. The maximum SNR that can be obtained from this circuit is $44dB$.

3.3 Discussion

Clearly, the SNR achieved by the time-mode weighted average circuit is higher than the SNR achieved by voltage-mode and current-mode weighted average circuits. The SNR values obtained from simulations differ only by 1% from the SNR values obtained through hand-calculations as shown in Figure 3–6.

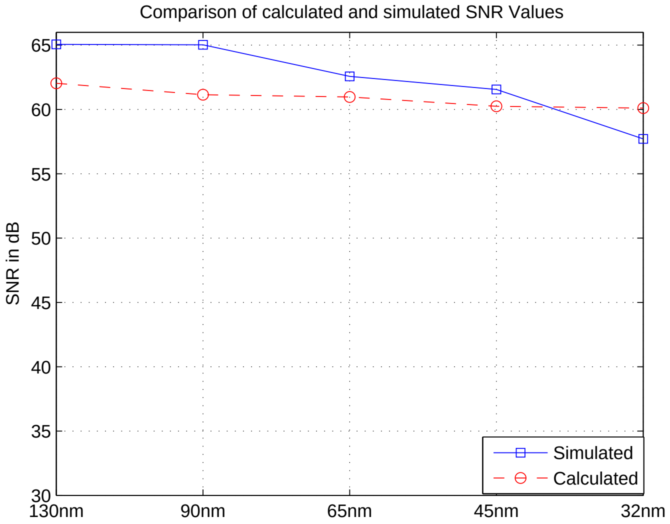

Figure 3–6. Calculated and simulated SNR values of a time-mode weighted averaging circuit over technology

So far we have just discussed a single type of time-mode circuit - the weighted averaging circuit. In Chapter 4, we will describe other time-mode computational circuits.

CHAPTER 4
OTHER TIME-MODE CIRCUIT EXAMPLES

In this chapter, we will introduce a family of time-mode circuits that can perform linear computations like weighted subtraction, weighted sum, scalar multiplication, maximum and minimum computations.

4.1 Weighted Subtraction Circuit

By replacing the PMOS transistor M_2 by an NMOS transistor and changing the direction of the I_2 of the basic block in Figure 2–1A, we obtain a circuit that can perform weighted subtraction of steps occurring at t_1 and t_2 as shown in Figure 4–1A.

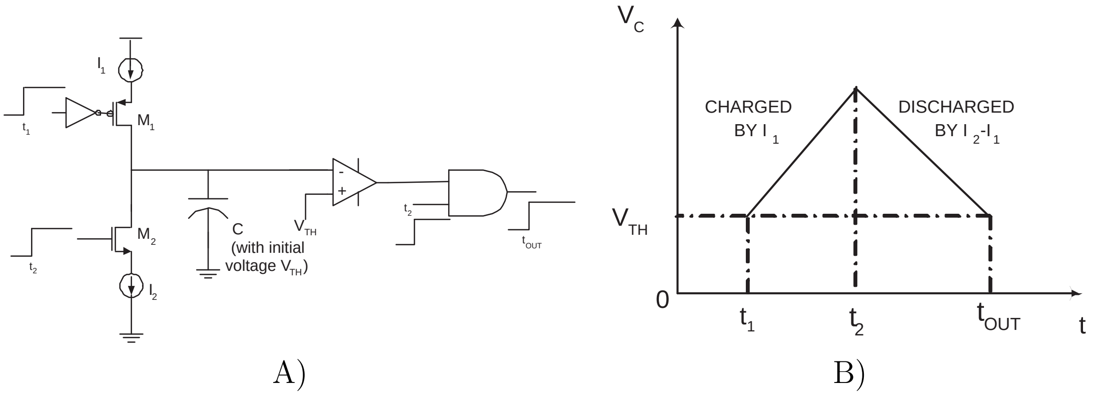

Figure 4–1. Weighted subtraction circuit. A) Circuit schematic. B) Idealized graph showing the capacitor's voltage at different time periods.

We will assume that the capacitor is initially charged to a voltage V_{TH}, $t_1 < t_2$, $I_2 > I_1$. As soon as the first step enters the block at time t_1, the current source I_1 starts to charge the capacitor. When the input step occurs at time t_2, net current $I_2 - I_1$ (with $I_2 > I_1$) starts discharging the capacitor as shown in Figure 4–1B. When the capacitor voltage reaches V_{TH}, the comparator outputs a step at time t_{OUT}. The output of the comparator contains an unwanted pulse at the reference time because the positive and negative terminals of the comparator carries the same voltage V_{TH}. The AND gate connected to the output of the comparator ensures that the output from the block contains only a step output at time t_{OUT}. Once the block outputs a step, an appropriate reset stage (not shown in the figure) resets the capacitor to $0V$.

The output t_{OUT} from the block is given by the equation,

$$t_{OUT} = \frac{I_2 t_2 - I_1 t_1}{I_2 - I_1} \tag{4–1}$$

We see from the equation above that, the block applies a weight $I_2/(I_2 - I_1)$ to t_2 and a weight $I_1/(I_2 - I_1)$ to t_1. This block has a single-ended output.

Without the assumptions made above, different outputs given out by the block can be summarized in a single equation:

$$t_{OUT} = \begin{cases} \dfrac{I_2 t_2 - I_1 t_1}{I_2 - I_1}, & \text{for } t_1 < t_2, I_2 > I_1 \quad (4\text{–}2a) \\ \dfrac{I_1 t_1 - I_2 t_2}{I_1 - I_2}, & \text{for } t_1 > t_2, I_2 < I_1 \quad (4\text{–}2b) \\ \text{No output}, & \text{otherwise} \quad (4\text{–}2c) \end{cases}$$

As in the weighted averaging circuit, inputs t_1, t_2 and output t_{OUT} are time-steps and are defined within a frame. When the frame ends, the inputs and the output steps also end and the circuit is reset. As the next frame starts, the circuit would be ready to process the next set of inputs t_1 and t_2.

4.2 Weighted Sum Circuit

The circuit shown in Figure 4–2A is again a minor modification of the basic block shown in Figure 2–1A. We will assume that the capacitor is initially charged to a voltage V_{TH}, $t_1 < t_2$, $I_2 - I_1 < I_3$. As soon as the frame starts (at time t_{REF}), net current $I_1 + I_2 - I_3$ starts to charge the capacitor as shown in Figure 4–2B. When the first temporal signal enters the block at time $\hat{t}_1$ (where $\hat{t}_1$ is defined as t_1 with respect to reference time t_{REF}) , the current source I_1 stops charging the capacitor and net current $I_2 - I_3$ charges the capacitor C. When the second signal enters the block at time $\hat{t}_2$ (where $\hat{t}_2$ is defined as t_2 with respect to reference time t_{REF}), current source I_3 discharges the capacitor. A comparator senses the voltage across the capacitor and outputs a step when the voltage reaches the threshold voltage V_{TH}. The output of the comparator would contain an unwanted pulse at the reference time because the positive and negative terminals of the comparator carry the same voltage V_{TH}. The AND gate connected to the output of the comparator ensures that the output from the block contains only a step output at time $\hat{t}_{OUT} = t_{OUT} - t_{REF}$. Once the block outputs a step and the frame ends, an appropriate reset stage (not shown in the figure) would reset the capacitor voltage to V_{TH} at reference time t_{REF}.

t_{OUT} is the time when the output step of the block, makes its transition from low to high voltage.

$$\hat{t}_{OUT} = (\frac{I_1}{I_3})\hat{t}_1 + (\frac{I_2}{I_3})\hat{t}_2 \tag{4–3}$$

From the above equation, we observe that the block computes a weighted sum of the two input time steps occurring at times $\hat{t}_1$ and $\hat{t}_2$.

An output from the block occurs when

$$(I_1 + I_2 - I_3)\hat{t}_1 + (I_2 - I_3)(\hat{t}_2 - \hat{t}_1) > 0 \tag{4–4}$$

Solving Eq. 4–4, we would get

$$I_1\hat{t}_1 + I_2\hat{t}_2 > I_3\hat{t}_2 \tag{4–5}$$

Eq. 4–5 can be interpreted as,

$$\hat{t}_2 > (\frac{I_1}{I_2 - I_3})\hat{t}_1 \tag{4–6}$$

Since $t_1 < t_2$ was assumed, it follows that $\frac{I_1}{I_2 - I_3} > 1$ or $I_3 > I_2 - I_1$.

If we assume that t_2 occurs before t_1, we would get an output from the block, when

$$(I_1 + I_2 - I_3)\hat{t}_2 + (I_1 - I_3)(\hat{t}_1 - \hat{t}_2) > 0 \tag{4–7}$$

Solving Eq. 4–7 gives $I_3 > I_1 - I_2$.

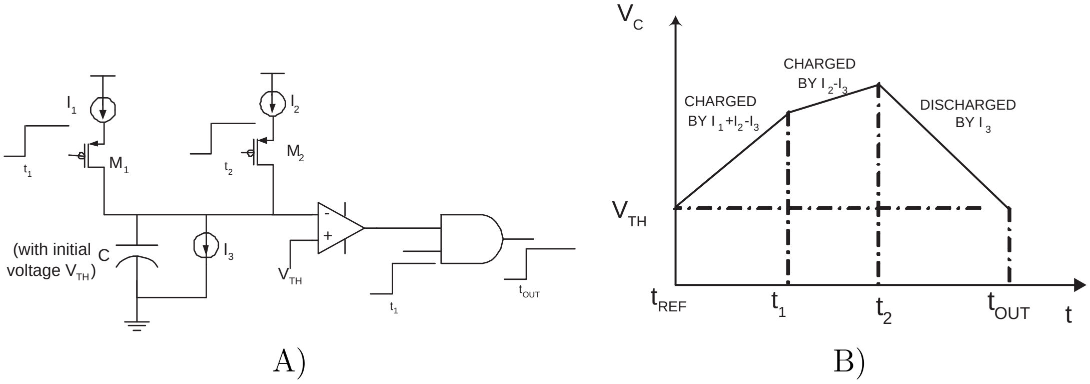

Figure 4–2. Weighted sum circuit. A) Circuit schematic. B) Idealized graph showing the capacitor's voltage at different time periods.

In both cases, $I_1 = I_2 = I$ results in

$$\hat{t}_{OUT} = \hat{t}_1 + \hat{t}_2 \tag{4–8}$$

This case corresponds to the sum of two input time steps occurring at $\hat{t}_1$ and $\hat{t}_2$. Thus, we see that by controlling the current sources, we achieve two different functionalities from the block - sum and weighted sum.

Without the assumptions made above, different outputs given out by the block can be summarized in a single equation:

$$\hat{t}_{OUT} = \begin{cases} (\frac{I_1}{I_3})\hat{t}_1 + (\frac{I_2}{I_3})\hat{t}_2, & \text{for } t_1 < t_2, I_3 > (I_2 - I_1) \text{ or } t_1 > t_2, I_3 > (I_1 - I_2) \quad \text{(4–9a)} \\ \hat{t}_1 + \hat{t}_2, & \text{for } t_1 < t_2 \text{ or } t_1 > t_2, I_1 = I_2 = I_3 \quad \text{(4–9b)} \\ \text{No output,} & \text{otherwise} \quad \text{(4–9c)} \end{cases}$$

The circuit has a single-ended output; the inputs and outputs occurring at t_1, t_2 and t_{OUT} are defined with respect to a time reference t_{REF} (start of the frame).

4.3 Scalar Multiplication Circuit

By removing the PMOS transistor M_1 that controlled current source I_1 charging the capacitor, replacing PMOS transistor M_2 by an NMOS transistor M_2 and changing the direction of the I_2 of the basic block in Figure 2–1A, we obtain a circuit that can be used for scalar multiplication of a temporal signal entering the block at time t_2 as shown in Figure 4–3A.

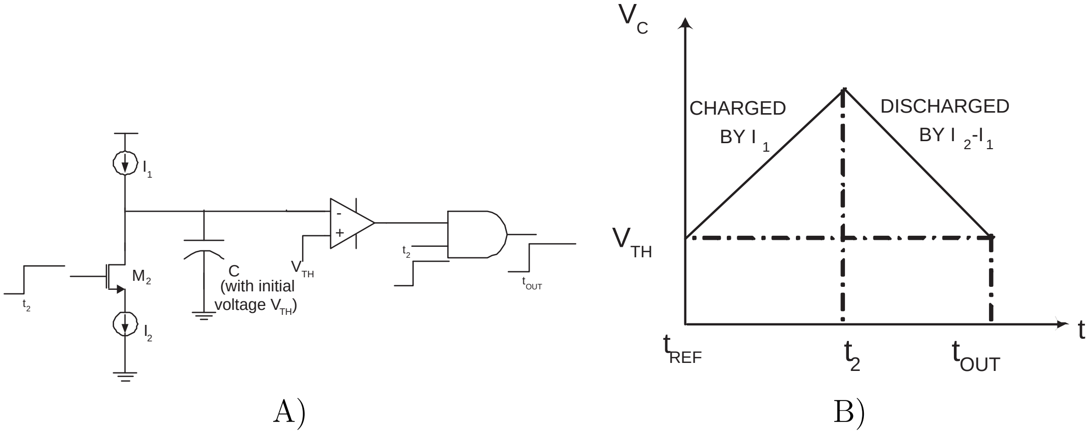

Figure 4–3. Scalar multiplication circuit. A) Circuit schematic. B) Idealized graph showing the capacitor's voltage at different time periods.

Assuming that the capacitor is initially charged to a voltage V_{TH}, the current source I_1 starts to charge the capacitor as soon as the frame starts (at time t_{REF}). The input step occurs at time $\hat{t}_2$ where, as above, $\hat{t}_2$ is defined as t_2 with respect to reference time t_{REF}. The current source $I_2 - I_1$ starts discharging the capacitor as shown in Figure 4–3B. When the capacitor voltage reaches V_{TH}, the comparator outputs a step at time t_{OUT}. The output of the comparator would also contain an unwanted pulse at the reference time because the positive and negative terminals of the comparator would carry the same voltage V_{TH}. The AND gate connected to the output of the comparator ensures that the output from the block contains only a step output at time t_{OUT}. Once the block outputs a step, an appropriate reset stage (not shown in the figure) would reset the capacitor to V_{TH} at reference time t_{REF}.

The output t_{OUT} from the block is given by the equation,

$$\hat{t}_{OUT} = (\frac{I_2}{I_2 - I_1})\hat{t}_2 \tag{4–10}$$

We see from the equation above that, the block multiplies time $\hat{t}_2$ with a scalar $I_2/(I_2 - I_1)$.

Without the assumptions made above, different outputs given out by the block can be summarized in a single equation:

$$\hat{t}_{OUT} = \begin{cases} (\frac{I_2}{I_2 - I_1})\hat{t}_2, & \text{for } I_2 > I_1 \quad (4\text{–}11a) \\ \text{No output}, & \text{for } I_2 \leq I_1 \quad (4\text{–}11b) \end{cases}$$

This block has a single-ended output and the inputs and outputs are defined with respect to a time reference t_{REF}(the start of the frame).

4.4 Maximum(MAX)/Minimum(MIN) Circuit

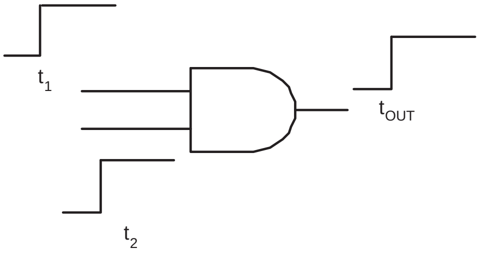

Figure 4–4. Circuit schematic of MAX circuit

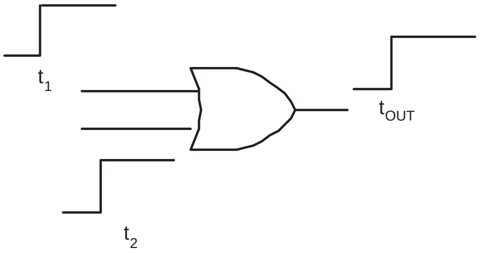

Figure 4–5. Circuit schematic of MIN circuit

The MAX and MIN circuits shown in Figures 4–4 and 4–5 support inputs and outputs that have absolute time as the reference. The output from the MAX and MIN circuits are single-ended. This block processes two temporal signals say, the time steps occurring at t_1 and t_2 as shown in the figure, and determines the $max(t_1, t_2)$ or $min(t_1, t_2)$ of the two steps. If the signal was to be represented using voltages, a complex circuit would be required to compute $max(V_1, V_2)$ or $min(V_1, V_2)$. In time-based analog computation, the circuitry to compute these functions is straightforward.

The time-mode linear computational circuits we have discussed so far and the thresholded difference block (to be discussed in Chapter 5) can be classified into different subclasses based on their output style, shown in Table 4–1.

Table 4–1. Classification of Time-mode computational circuits. Relative time reference implies that the inputs and outputs are defined with respect to a reference time (start of a frame). Absolute time reference implies that inputs and outputs are not defined with respect to a reference time.

Output	Single-ended	Differential
Absolute time reference	Weighted Averaging Circuit Weighted Subtraction Circuit MAX circuit MIN circuit	Thresholded difference block of Edge detection
Relative time reference	Sum circuit Scalar Multiplication Circuit	

So far, we have discussed time-mode computational circuits to perform computations like weighted average, weighted subtraction, weighted sum, scalar multiplication, maximum and minimum. In Chapter 6, we will discuss a couple of applications - a time-mode edge detection circuit and a time-mode 3-tap FIR filter.

CHAPTER 5
APPLICATION OF TIME-MODE CIRCUITS

5.1 Time-Mode Edge Detection Circuit

Time-mode circuits provide a seamless interface to the growing number of time-based sensors which already output compatible timing events [14], [15]. In this section, an example is given where a time-mode edge detector is developed to directly interface to the output of a time-to-first spike imager [14].

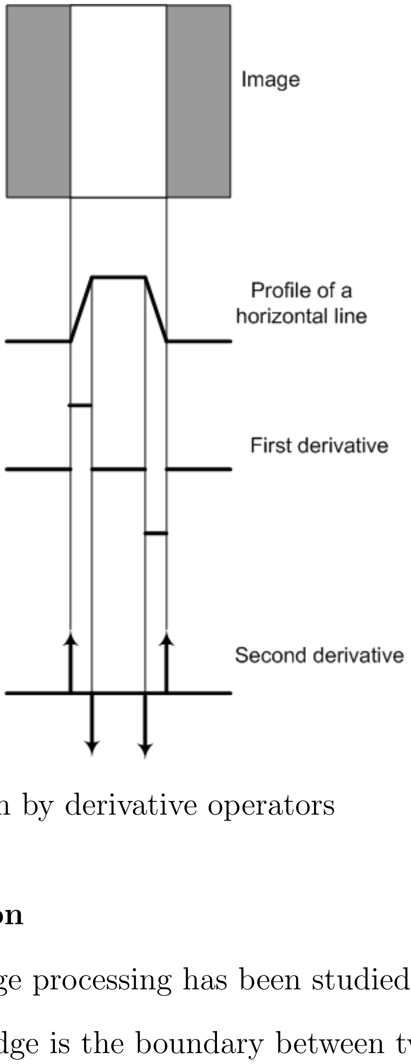

Figure 5–1. Edge detection by derivative operators

5.1.1 Basic Formulation

Edge detection in image processing has been studied for many years and is well understood [23]. An edge is the boundary between two regions with relatively distinct gray-level properties. In all the discussions below, we assume that the regions in question are sufficiently homogeneous so that the transition between two regions can be determined on the basis of gray-level discontinuities alone.

Traditionally, the idea underlying most edge-detection techniques is the computation of the local derivative operator. This concept is illustrated in Figure 5–1. The figure shows a synthetic image of a light object on a dark background, the gray-level profile along a horizontal scan line of the image, and the first and second derivatives of the profile. We note from the profile that an edge (transition from dark to light) is modeled as a ramp, rather than as an abrupt change of gray level. The first derivative of an edge modeled in this manner is 0 in all regions of constant gray level, and assumes a constant value during a gray-level transition. The second derivative, on the other hand, is 0 in all locations, except at the onset and termination of a gray-level transition. Based on these remarks, it is evident that the magnitude of the first derivative can be used to detect the presence of an edge, while the sign of the second derivative can be used to determine whether an edge pixel lies on the dark (background) or light (object) side of an edge. The sign of the second derivative in Figure 5–1 for example, is positive for pixels lying on the dark side of both the leading and trailing edges of the object, while the sign is negative for pixels on the light side of these edges. Although the discussion thus far has been limited to a one-dimensional horizontal profile, a similar argument applies to an edge of any orientation in an image.

In this chapter, we will discuss the design of a time-mode edge detector that performs a first derivative operation on the pixel outputs through a novel time-mode thresholded differencing block to detect both the presence and the sign of the edges. Significant changes in scene illuminance are typically detected with a spatial derivative operation following a spatial smoothing process that reduces high frequency noise. Figure 5–2 shows the basic data flow in the proposed time-based edge detection scheme. Initially the time steps corresponding to pixel intensities are smoothed. Next, the smoothed time steps are fed to a thresholded differencing block that finds the difference between the input steps and thresholds the result. The output of the thresholded derivative block can either be positive or negative implying a positive or negative edge between pixels.

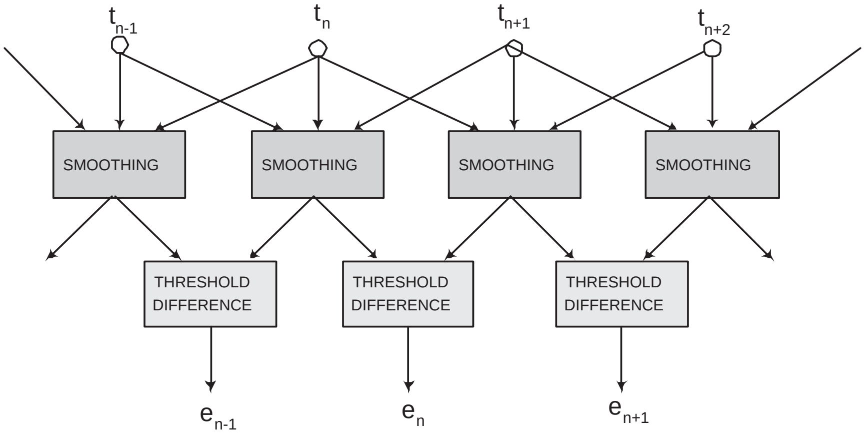

Figure 5–2. Data flow in time-mode edge detection

5.1.2 Smoothing

We have previously fabricated a time-to-first spike CMOS imager in our lab [24],[14]. This imager provides output steps whose timing encodes illumination information at each pixel. These spatial information must be smoothed to eliminate noise in the image as well as noise introduced by the electronics.

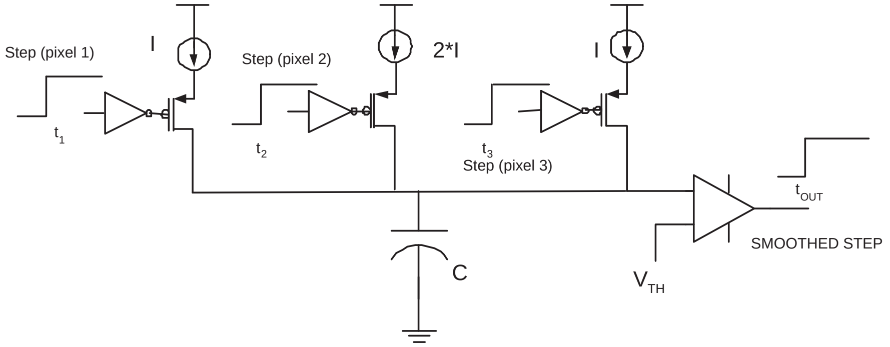

Figure 5–3. Circuit to smooth pixel intensities

Figure 5–3 shows a circuit that could be used to perform smoothing of these pixel intensities. We implement a standard convolution mask with weights of 1-2-1 by appropriately scaling the current source values. Since the circuit shown in Figure 5–3 is a special case of the weighted averaging circuit explained in Chapter 2, we can easily derive the smoothing block's output expressed below:

$$t_{OUT} = \frac{t_1 + 2t_2 + t_3}{4} + \frac{CV_{TH}}{4I} \tag{5–1}$$

5.1.3 Thresholded Difference

The threshold difference block performs a spatial first derivative operation on the smoothing circuit's outputs. By replacing one PMOS transistor by an NMOS transistor and changing the direction of the corresponding current source in the time-mode weighted averaging circuit, we obtain a circuit that can be used to obtain thresholded differences of steps shown in Figure 5–4. There are two cases to be considered assuming that V_C is initially reset to a midrange voltage:

- One of the smoothed steps enters the thresholded difference block first, starts to linearly charge (or discharge) the capacitor until it hits the positive (or negative) threshold V_{TH} (or $-V_{TH}$) before the second smoothed step enters the block. Here, we have a step from the positive (or negative) output of the block at time
$$t_{OUT} = t_1 + \frac{CV_{TH}}{I} \tag{5–2}$$
The threshold implemented by this block is CV_{TH}/I. This threshold value can be programmed by choosing desired values for V_{TH} and I.

- The two smoothed step inputs arrive within the threshold time CV_{TH}/I. Since the positive and negative current sources exactly cancel one other, no step is generated from either the positive or negative output indicating no edge between pixels. Mismatches between the two current sources will eventually cause one of the outputs to fire, but at a time much longer than the frame time of the system.

If the thresholded difference block fires an output, we can know the presence of edges between adjacent pixels. Also, depending on whether we get positive output or negative output we can infer the sign of the edges. That is, a positive output implies that pixel 1 is brighter than pixel 2. Thus, from the outputs of the

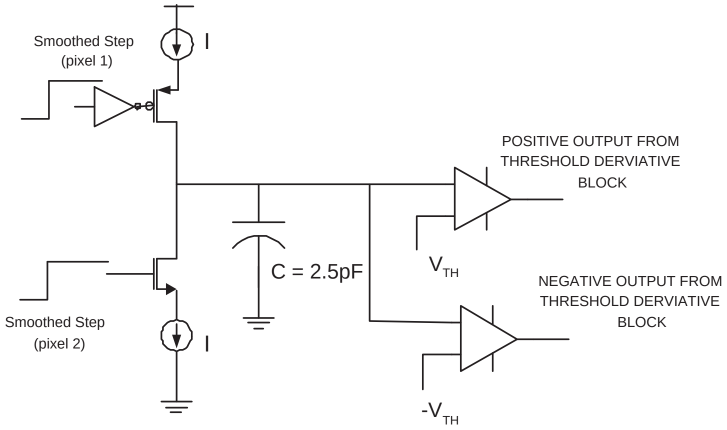

Figure 5–4. Circuit used to obtain thresholded differences on the smoothed steps

threshold differentiation block (that performs a spatial first derivative operation), we can detect both the presence and the sign of the edges.

5.1.4 Results

Using the time-mode edge detection concepts explained above, we processed a noisy JPEG image to detect edges. The whole operation is completed within 3 frames. The frames are defined by the imaging process - typically 30ms. The MATLAB simulation results are shown in Figure 5–5. In the first frame, we converted the pixel magnitude information (between 0 and 255) of each pixel to timing information using reverse coding. A bright pixel would fire earlier compared to a dark pixel, that is, with respect to the frame the bright pixel would have a smaller temporal amplitude compared to a dark pixel. In the second frame, we remove the spurious noise in the image by using time-mode smoothing circuits. After smoothing, we perform the spatial first derivative operation by running the smoothing block's outputs through time-mode thresholded difference blocks in the third frame.

For better understanding, let us restrict our analysis to 16 pixels. The original noisy image, smoothed image and the detected edges are shown in Figure 5–6. The noisy original image and the smoothed image are shown in dotted lines and solid lines respectively. The edges detected are shown special characters in the figure. From the results shown, we can infer that the time-based edge detection method is extremely accurate.

For these 16 pixels, Figure 5–7 shows the Cadence simulation outputs from different stages in the time-based edge detection process. The length of the frame and the threshold we chose for the thresholded difference block are $30ms$ and $15ms$ respectively. In the figure, the original image is shown followed by the temporal signals output by the imager. It is followed by the outputs of the smoothing and thresholded difference blocks. The edge detection circuits needs 3 frames to complete their operations. The final results indicate that only three edges were detected to be above the threshold. The power consumed by the edge detection circuits for these 16 pixels was in the order to $35\mu W$.

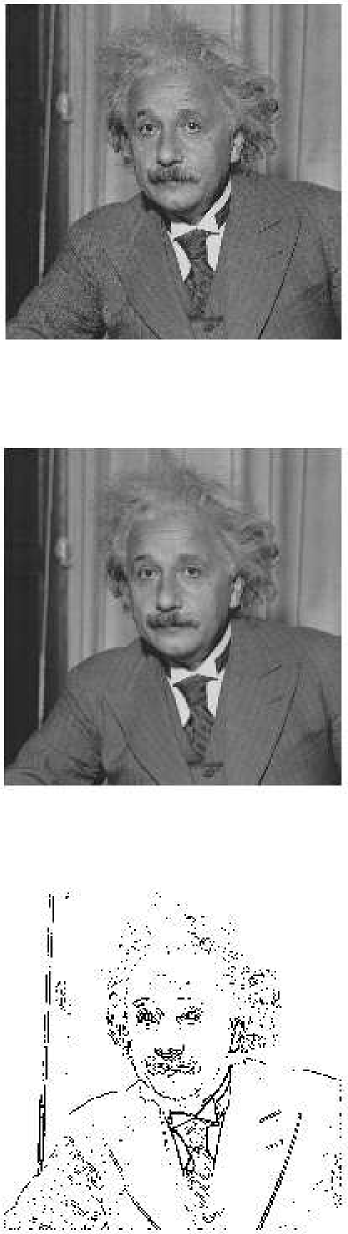

Figure 5–5. MATLAB simulation results showing the original image, smoothed image and the detected edges of an image

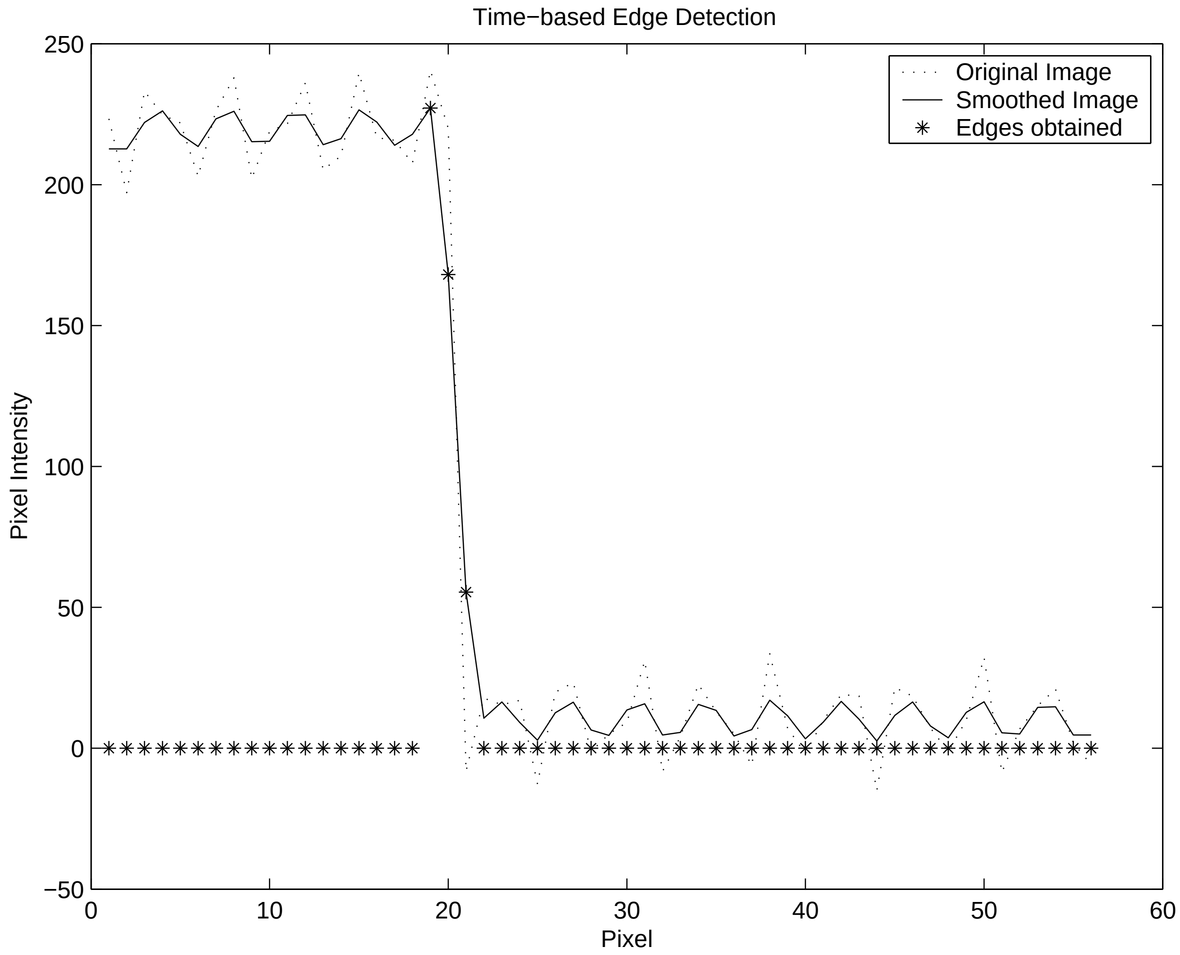

Figure 5–6. Simulation results showing the original image, smoothed image and the detected edges of a 16 pixel image

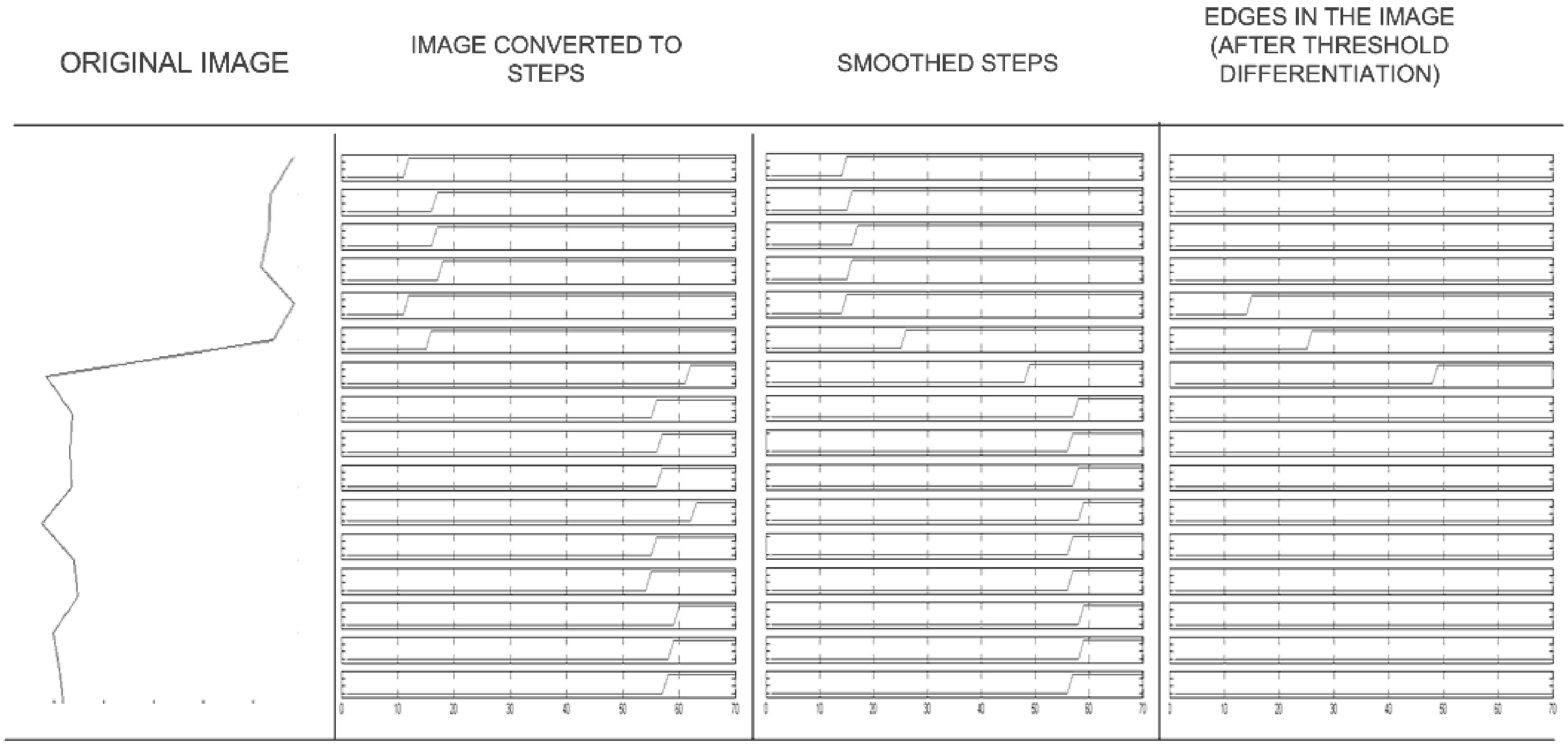

Figure 5–7. Outputs from different stages in time-mode edge detection

5.1.5 Discussion

The length of the frame that governs the operation of the time-mode edge detection circuit has a very important tradeoff. If we opt for longer frames, the DR of inputs that can be processed by the edge detection circuits is large. With short frames, the speed of the entire edge detection operation is increased and the leakage currents of the transistors in the thresholded difference block won't cause erroneous outputs.

From Eq. 5–2, we can easily infer that by controlling C, V_{TH} or I we can program the desired threshold in the thresholded difference block. But, once a edge detection chip is designed, it is tough to vary the value of C. Therefore, to vary the threshold, we should either tune V_{TH} or I off-chip.

5.2 3-Tap 1-Quadrant Time-Mode Finite Impulse Response Filter

FIR realization for a N-tap FIR filter follows directly from the convolution sum relationship written in the form:

$$y(n) = \sum_{k=0}^{N-1} w(k)x(n-k) \tag{5–3}$$

For a 3-tap filter,

$$y(n) = \sum_{k=0}^{2} w(k)x(n-k) = w(0)x(n) + w(1)x(n-1) + w(2)x(n-2) \tag{5–4}$$

From Eq. 5–4, we see that to compute the n-th sample of the output of a 3-tap FIR filter, we need the current input $x(n)$ and two previous inputs $x(n-1)$ and $x(n-2)$. For example, the output at the 3rd sampling period is given by,

$$y(3T) = w(0)x(3T) + w(1)x(2T) + w(2)x(1T) \tag{5–5}$$

If the input and output at the 3rd sampling period are represented in time by t_{IN}^{3T} and t_{OUT}^{3T} respectively, then we can implement a time-mode FIR filter if we can implement,

$$t_{OUT}^{3T} = w(0)t_{IN}^{3T} + w(1)t_{IN}^{2T} + w(2)t_{IN}^{T} \qquad (5\text{–}6)$$

From Eq. 5–6, we see that we would need inputs t_{IN}^{2T} and t_{IN}^{T} other than t_{IN}^{3T} to obtain t_{OUT}^{3T}. To do this, we can either

- delay t_{IN}^{2T} by one sampling period and t_{IN}^{T} by two sampling periods, or
- store t_{IN}^{2T} for one sampling period and t_{IN}^{T} for sampling periods,

so that these inputs would be available during the sampling period 3T when the computation shown in Eq. 5–6 is to be performed.

When the information is in time, delaying that information (information is encoded in the rising edge of a time step referenced to the start of a frame) would involve converting the time information to voltage and then converting that voltage back to time (information again is a time step but referenced to a new frame) using analog components. Since we are considering the implementation of a 3-tap FIR filter, we would need two delay stages. Since the delay stages involve analog components like current sources and capacitors that have matching constraints, we might end up with inaccurate delays that might lead to erroneous outputs. Therefore, the better option would be store the information over various sampling periods. Since, we have not yet come up with the circuit that would store time information directly in time, we convert the time information to voltage and store it on a capacitor.

5.2.1 Finite Impulse Response Computation in Time

The circuit shown in Figure 5–8 is similar to the prototype time-mode weighted average circuit except that this figure also shows the reset functionality. Three inputs that enter this block are

1. train of frames - these act as reference for the input and output steps.
2. train of input steps.

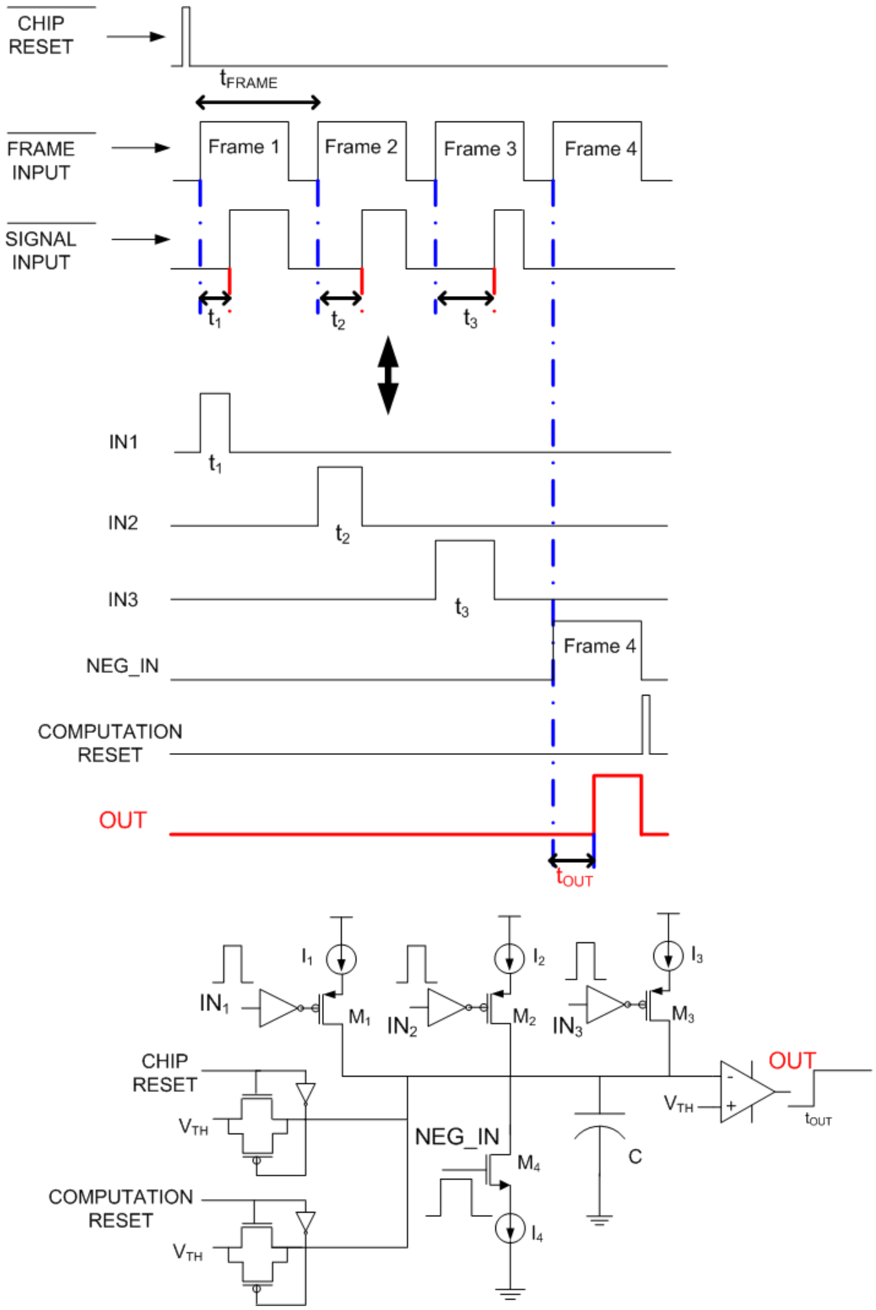

Figure 5–8. Computational block to be used in the FIR filter

3. chip reset.

Lets postpone the discussion of the input processing that should be done to the two input trains to generate signals IN1, IN2, IN3, NEG_IN and COMPUTATION RESET to the next section. These signals are the inputs to the computational block shown in Figure 5–8. Before any of the input trains enter the system, the chip reset signal would reset the capacitor's voltage to V_{TH}. As the first frame starts, IN1 generated by the input processing block turns on the switch M_1 for a period t_1. This let's the current source I_1 charge the capacitance C for the period t_1 as shown in Figure 5–9.

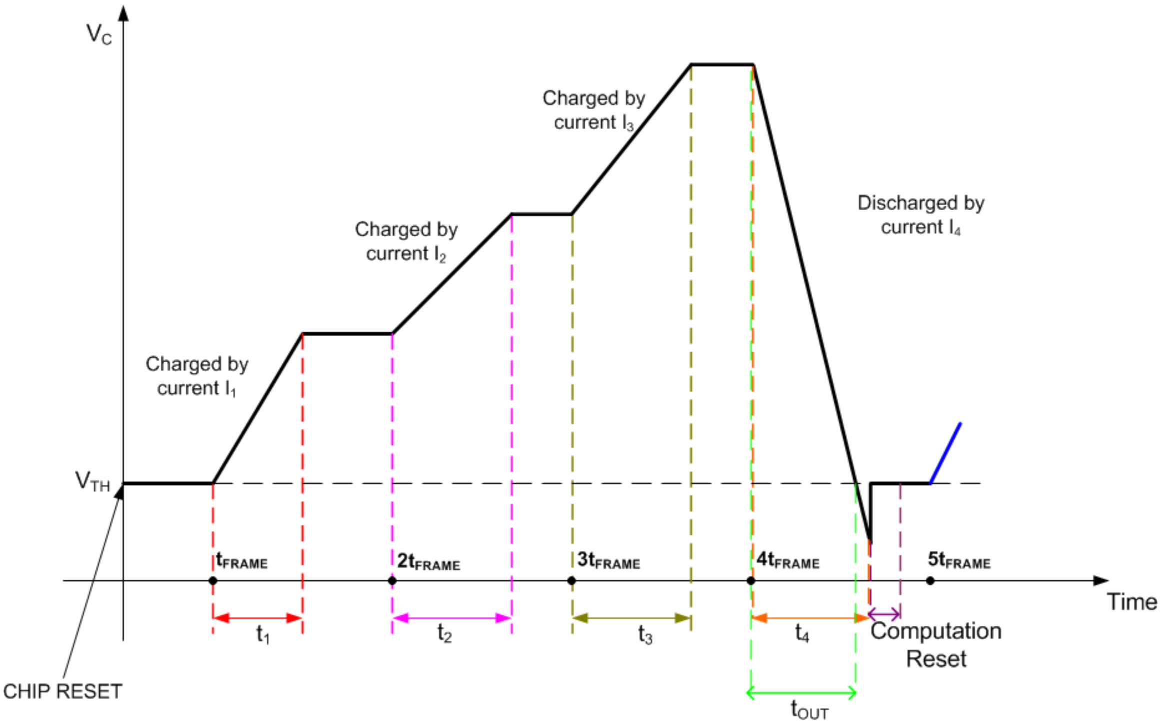

Figure 5–9. Voltage across the computational block's capacitor at various times

The voltage across the capacitor is given by,

$$V_C^1 = V_{TH} + \frac{I_1 t_1}{C} \tag{5–7}$$

The capacitor in this block performs two functionalities simultaneously:

- input t_1 is stored as voltage until the computation ends (at the end of the fourth frame) to facilitate the FIR type computation (assuming that the capacitor has minimal or no leakage).

- a weight of $\frac{I_1}{C}$ is applied to the input t_1 (though that weight is not the final weight applied to the input).

After the second frame starts, IN2 turns on the switch M_2 causing I_2 to charge the capacitance C for a period t_2. The new voltage across the capacitor is given by,

$$V_C^2 = V_{TH} + \frac{I_1 t_1}{C} + \frac{I_2 t_2}{C} \tag{5–8}$$

Now, the capacitor is holding both inputs t_1, t_2 and has applied weights $\frac{I_1}{C}$ and $\frac{I_2}{C}$ respectively. When the third frame starts, IN3 turns on the switch M_3 causing I_3 to charge the capacitance C for a period t_3. Now, the capacitor holds inputs t_1, t_2, t_3 and has applies weights $\frac{I_1}{C}$, $\frac{I_2}{C}$ and $\frac{I_3}{C}$ respectively as shown by the capacitor voltage Eq. 5–9.

$$V_C^3 = V_{TH} + \frac{I_1 t_1}{C} + \frac{I_2 t_2}{C} + \frac{I_3 t_3}{C} \tag{5–9}$$

As frame 4 starts, the signal NEG_IN turns on switch M4 and the current I_4 starts to discharge the capacitance C. With the voltage across the capacitance being continuously monitored by the comparator, voltage across the capacitance slowly decreases as I_4 discharges it and when the voltage reaches V_{TH} the comparator fires a step output. The rising time of this output step referenced to the fourth frame gives the desired output t_{OUT}.

$$\begin{aligned} t_{OUT} &= \frac{C(V_C^3 - V_{TH})}{I_4} \\ &= \frac{C(\frac{I_1 t_1}{C} + \frac{I_2 t_2}{C} + \frac{I_3 t_3}{C})}{I_4} \\ &= \frac{I_1 t_1}{I_4} + \frac{I_2 t_2}{I_4} + \frac{I_3 t_3}{I_4} \end{aligned} \tag{5–10}$$

From Eq. 5–10, we see that the computational block applies weights $\frac{I_1}{I_4}$, $\frac{I_2}{I_4}$ and $\frac{I_3}{I_4}$ to signals t_1, t_2 and t_3 and sums them together. After the block performs this computation, the capacitor voltage is reset to V_{TH} by the computation reset signal generated by the input processing block.

We made the following basic assumptions to arrive at the above result:

- The inputs t_1, t_2 and t_3 do not saturate the capacitance C.
- Frame 4's ON period is large enough to discharge the capacitance C (until the voltage across the capacitance reaches V_{TH}) and produce the output t_{OUT}.

Since, t_1, t_2, t_3 and t_{OUT} are defined with frames 1,2,3 and 4 as reference respectively, we can write Eq. 5–10 as,

$$t_{OUT}^{4T} = (\frac{I_1}{I_4})t_1^T + (\frac{I_2}{I_4})t_2^{2T} + (\frac{I_3}{I_4})t_3^{3T} \tag{5–11}$$

As mentioned in the assumptions above, the ON period of frame 4 should be large enough atleast to produce an output at time t_{OUT}. Therefore, $t_{ON}^{frame4} = t_{OUT}$. Assuming that the OFF period of the frame where the capacitor is reset to V_{TH} is extremely small, $t_{frame} \approx t_{OUT}$. Therefore, the minimum possible frame length $= t_{OUT}$ and the maximum possible sampling speed $= \frac{1}{t_{OUT}}$.

If the Eq. 5–11 can be interpreted in terms of samples then,

$$t_{OUT}(4) = (\frac{I_1}{I_4})t_{IN}(1) + (\frac{I_2}{I_4})t_{IN}(2) + (\frac{I_3}{I_4})t_{IN}(3) \tag{5–12}$$

Comparing Eq. 5–12 with the conventional 3-tap FIR equation shown below,

$$y(3) = w(2)x(1) + w(1)x(2) + w(0)x(3) \tag{5–13}$$

we see that t_{OUT} has an extra sample delay when compared to the conventional FIR output. In other words, the FIR filter's computation block has an extra pole at the origin as compared to the conventional FIR filter.

5.2.2 3-Tap 1-Quadrant Time-Mode FIR Filter Architecture

The main functional block of the 3-tap 1-quadrant time-mode FIR filter architecture is the computational block shown in Figure 5–8. Let's say that the computational block processes inputs $t_{IN}(1)$ referenced to frame 1, $t_{IN}(2)$ referenced to frame 2, $t_{IN}(3)$ referenced to frame 3 and produces an output $t_{OUT}(4)$ referenced to frame 4. The computational block gets ready to process the next set of inputs only at the end of frame 4 where the capacitor is reset to V_{TH}. The next output this block would produce is $t_{OUT}(8)$ processing $t_{IN}(5)$, $t_{IN}(6)$ and $t_{IN}(7)$. Since this block cannot produce the intermediate outputs $t_{OUT}(5)$, $t_{OUT}(6)$ and $t_{OUT}(7)$, we would need three more computational blocks to produce those outputs. Therefore, the 3-tap FIR filter would need in total, four computational blocks to continuously process the input time signals. In general, for a N-tap FIR filter, we would need N+1 computational blocks to construct the FIR filter.

The complete architecture of a 3-tap time-mode FIR filter is shown in Figure 5–10. The FIR filter block needs the same 3 inputs as the computational block - a chip reset, a train of frames and a train of time steps as shown in Figure 5–12. Every input step (example, t_1) in the train of time steps is defined with respect to a frame (example, frame 1) in the train of frames as shown in Figure 5–12. The trains of frames and time steps are fed to a input conditioning block. The architecture of the input conditioning block is shown in Figure 5–11.

The input conditioning block performs the following functions:

- the train of frames and input steps that enter the filter are decoded onto different lines so that they can be fed to the various computational blocks' inputs.

- generates the necessary charging/discharging pulses for the computational blocks.

- generates the reset pulses necessary to reset the capacitors of the computational blocks to their initial voltage V_{TH}.

Each computational block needs *3 inputs in different lines* - because we are designing a 3-tap FIR filter. Also, since there are four computational blocks that process the following different sets of inputs - (t_1,t_2,t_3), (t_2,t_3,t_4), (t_3,t_4,t_5) and

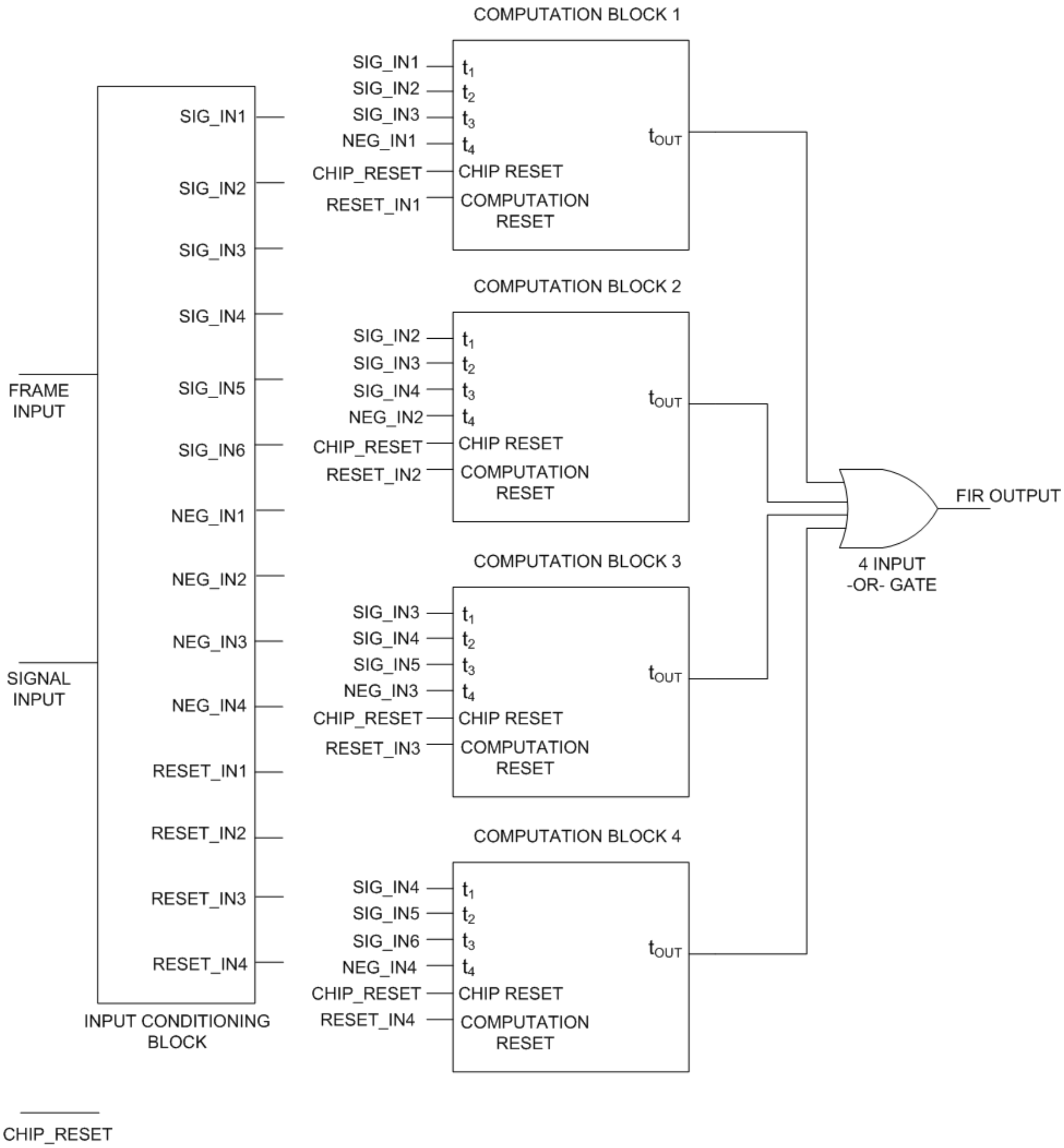

Figure 5–10. 3-tap time-mode FIR filter architecture

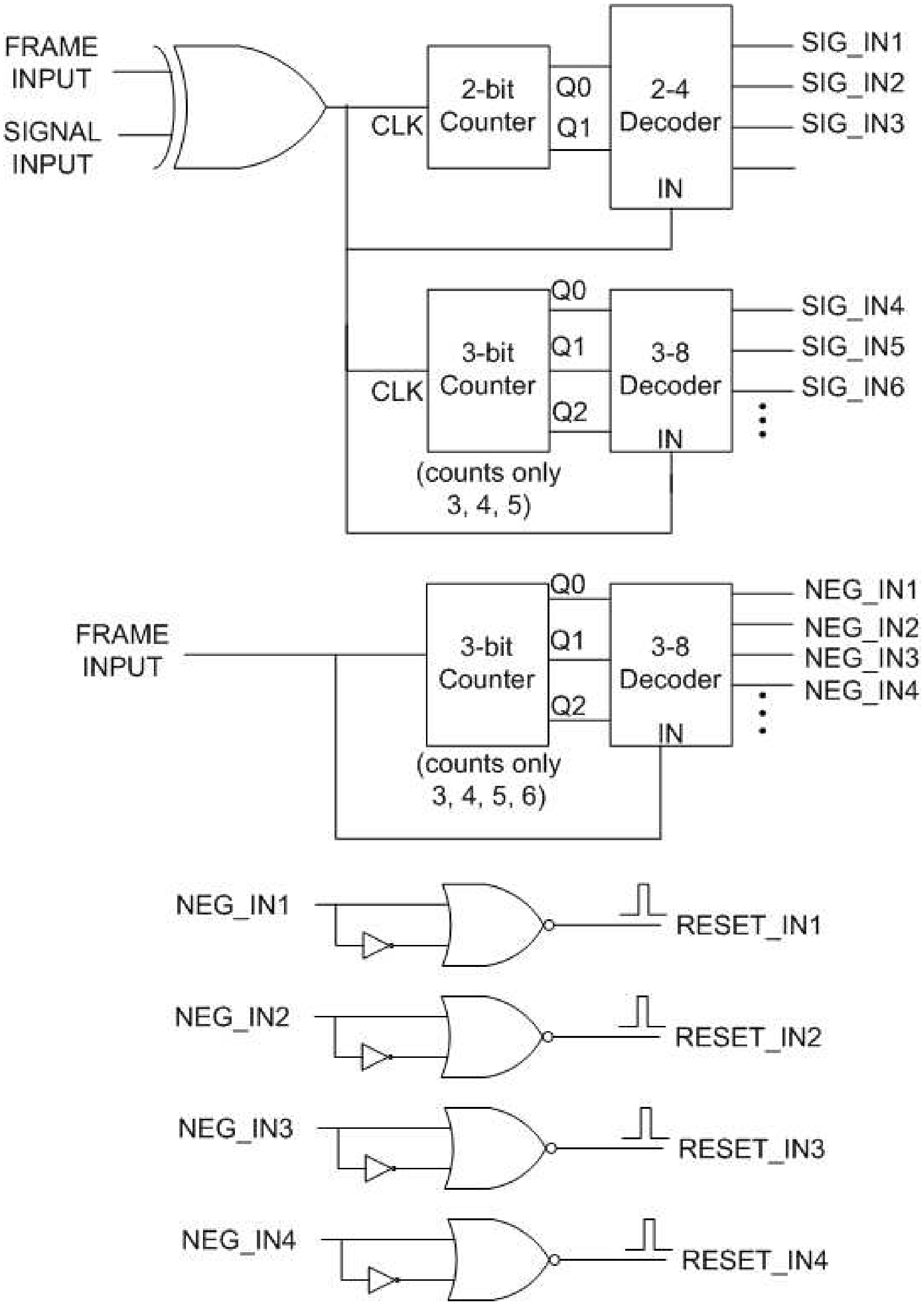

Figure 5–11. The architecture of the input conditioning block

(t_4,t_5,t_6) - that is, at every sample (frame), we would need *6 inputs in 6 different lines* for the 4 computational blocks. To keep moving these inputs at different input lines during every frame, we have used 2 counters - a 2-bit counter and a 3-bit counter (that counts between 3 and 5) - followed by decoders. Similarly, since we need the frames to discharge the capacitances of the computational blocks, we have a 3-bit counter followed by a decoder to decode the frame train onto 4 different lines.

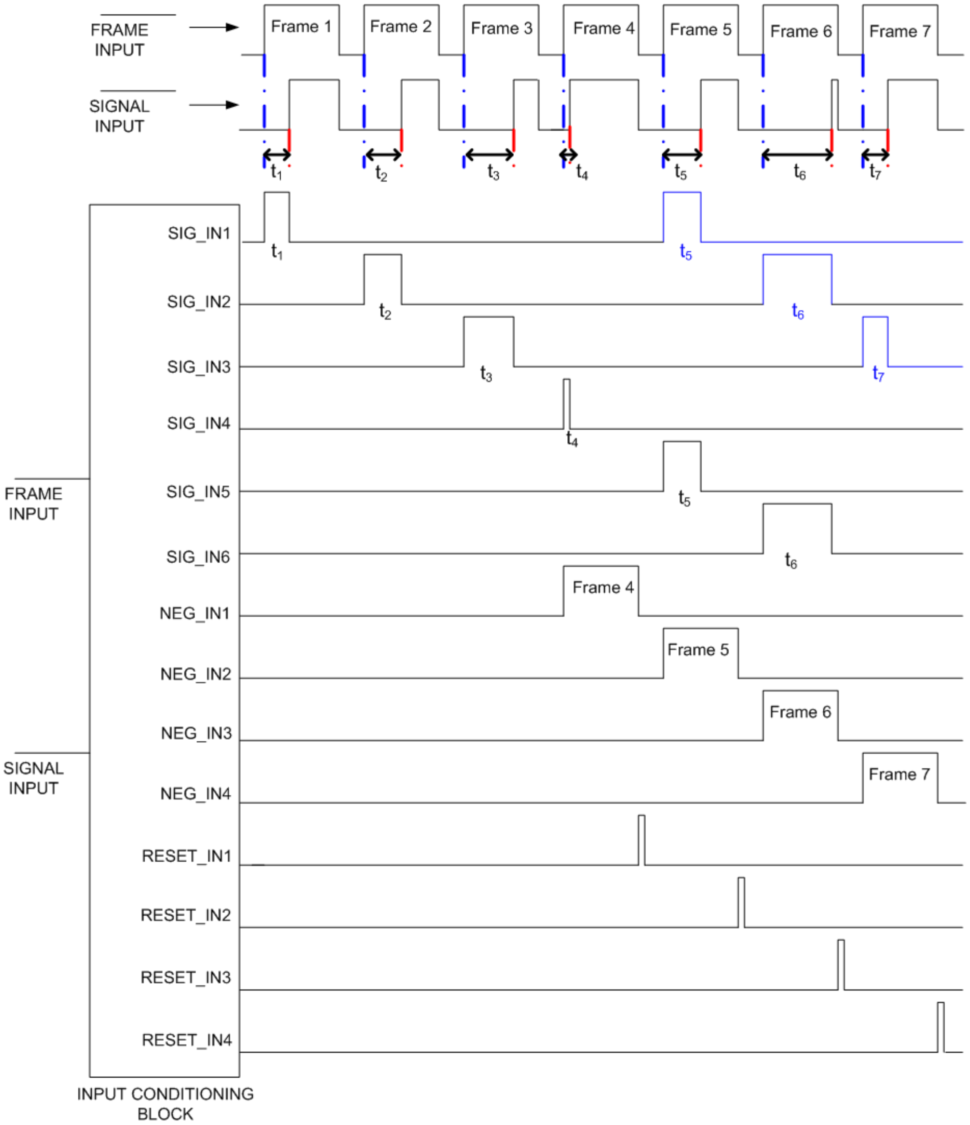

Figure 5–12. 3-tap FIR filter's input, digital preconditioning block and its outputs

The architecture shown is for a 1-quadrant FIR filter. That is, it can only process positive inputs and apply positive weights to those inputs. By adding extra circuitry, we can extend this architecture to process both positive and negative inputs and apply both positive and negative weights to those inputs.

5.2.3 Step-by-Step Description of the Functionality

1. Figure 5–13 describes the state of the computational blocks as input t_1 enters the blocks. At sampling instant t_{FRAME}, we can see only the capacitance of the computational block C_1 being charged by current I_1.

2. As input t_2 enters the blocks at sampling instant $2t_{FRAME}$,
- I_2 charges capacitor C_1, and,
- I_1 charges capacitor C_2 of computational block 2,

as shown in Figure 5–14.

3. As input t_3 enters the blocks at sampling instant $3t_{FRAME}$ as shown in Figure 5–15,
- I_3 charges capacitor C_1 of computational block 1,
- I_2 charges capacitor C_2 of computational block 2, and
- I_1 charges capacitor C_3 of computational block 3,

4. At sampling instant $4t_{FRAME}$ as frame 4 and input t_4 enter,
- frame 4 charges capacitor C_1,
- I_3 charges capacitor C_2, and,
- I_2 charges capacitor C_3

as shown in Figure 5–16.

5. As frame 4 is about to finish as shown in Figure 5–17,
- capacitor C_1 is reset to V_{TH} and it is ready for frame 5 and input t_5,
- voltage of capacitor C_2 stays a constant, and,
- voltage of capacitor C_3 stays a constant,

5.3 Simulation Results

To test the filter functionality of the circuit shown in Figure 5–10, we chose the following values for the current sources: $I_1 = 302nA$, $I_2 = 400nA$, $I_3 = 302nA$ and $I_4 = 1\mu A$ with C=$5pF$ and $V_{TH} = 2.5V$ (for a supply voltage of $5V$). The weights applied to inputs become $\frac{I_1}{I_4} = .302$, $\frac{I_2}{I_4} = .4$, $\frac{I_3}{I_4} = .302$.

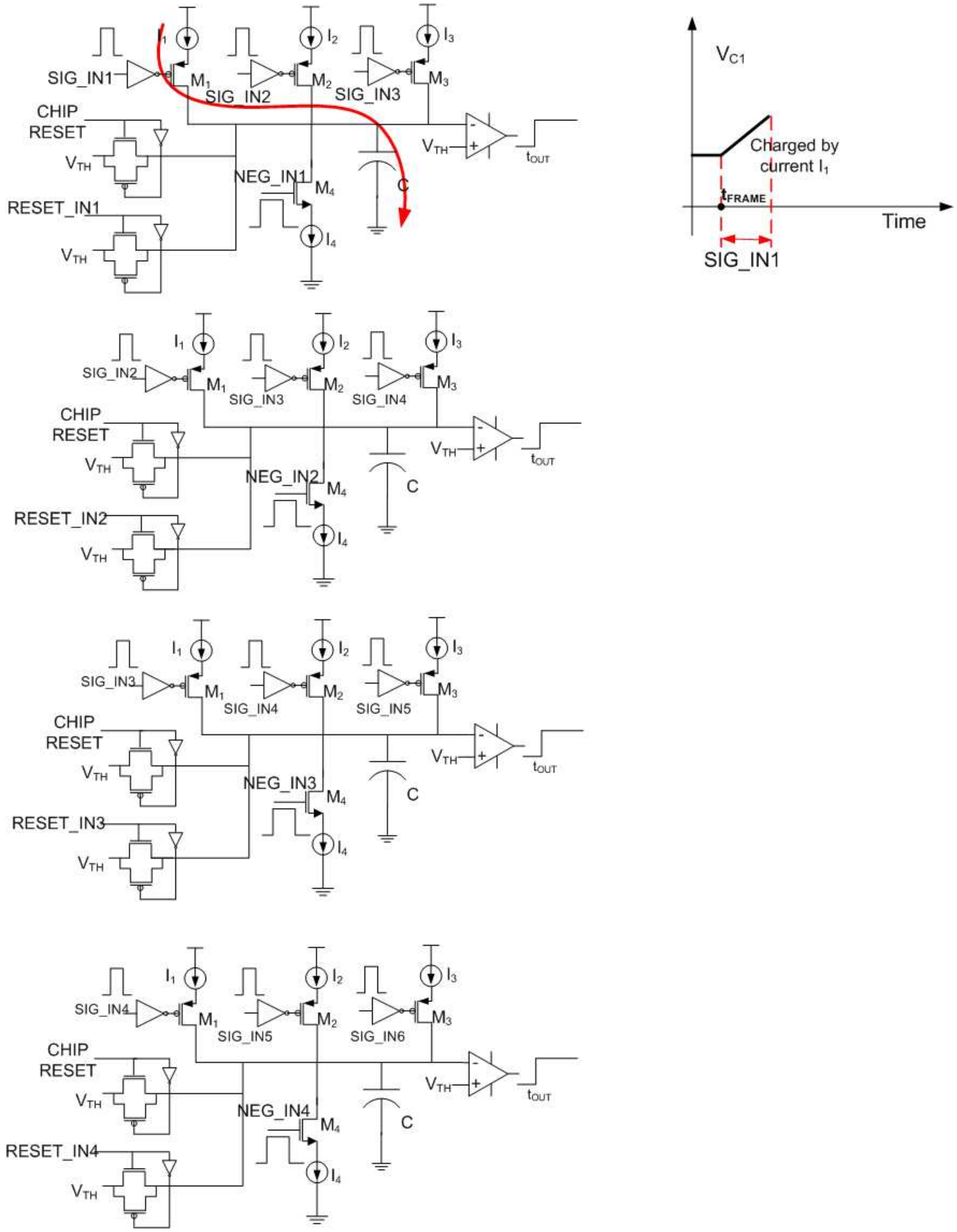

Figure 5–13. State of the FIR filter as input t_1 enters

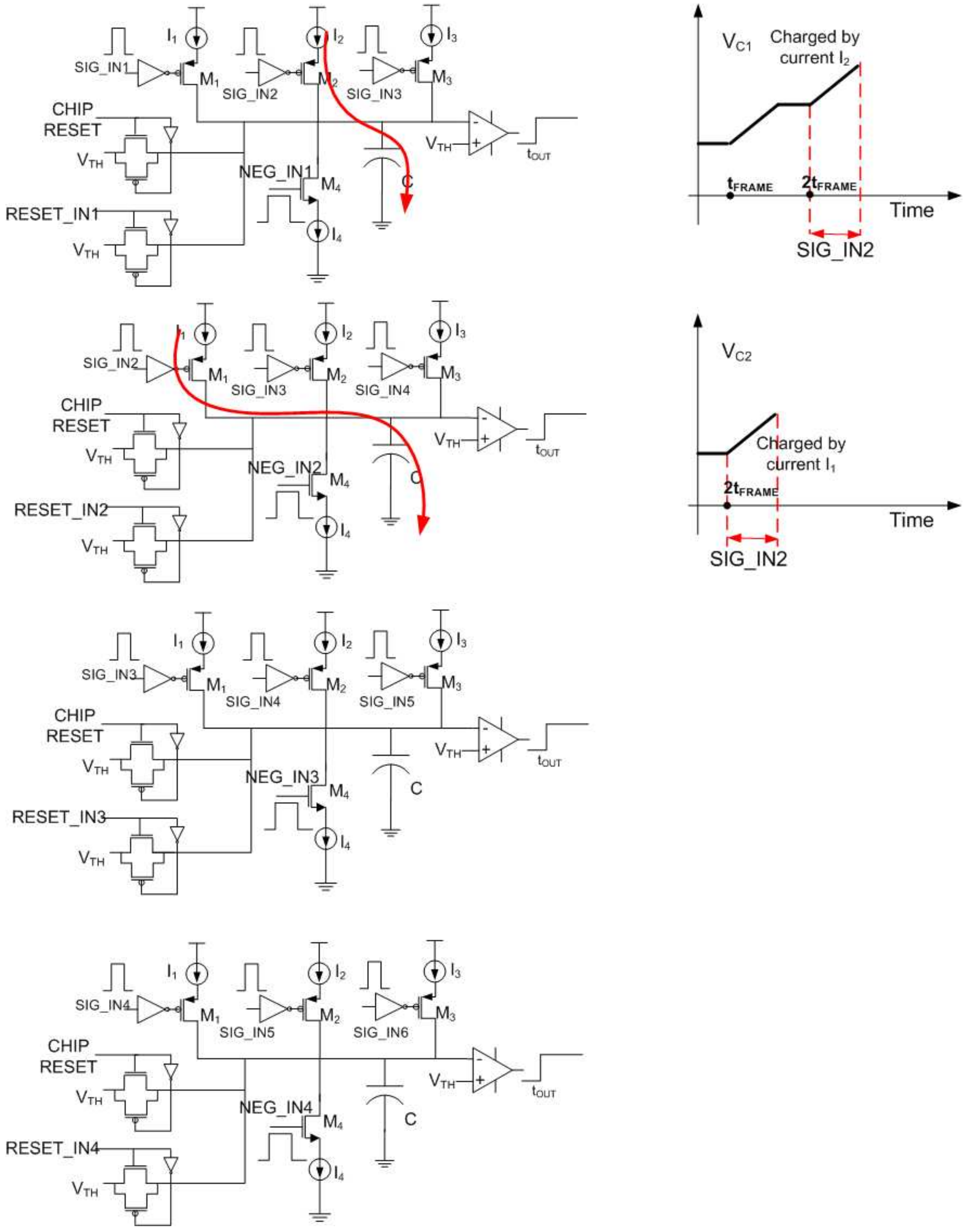

Figure 5–14. State of the FIR filter as input t_2 enters

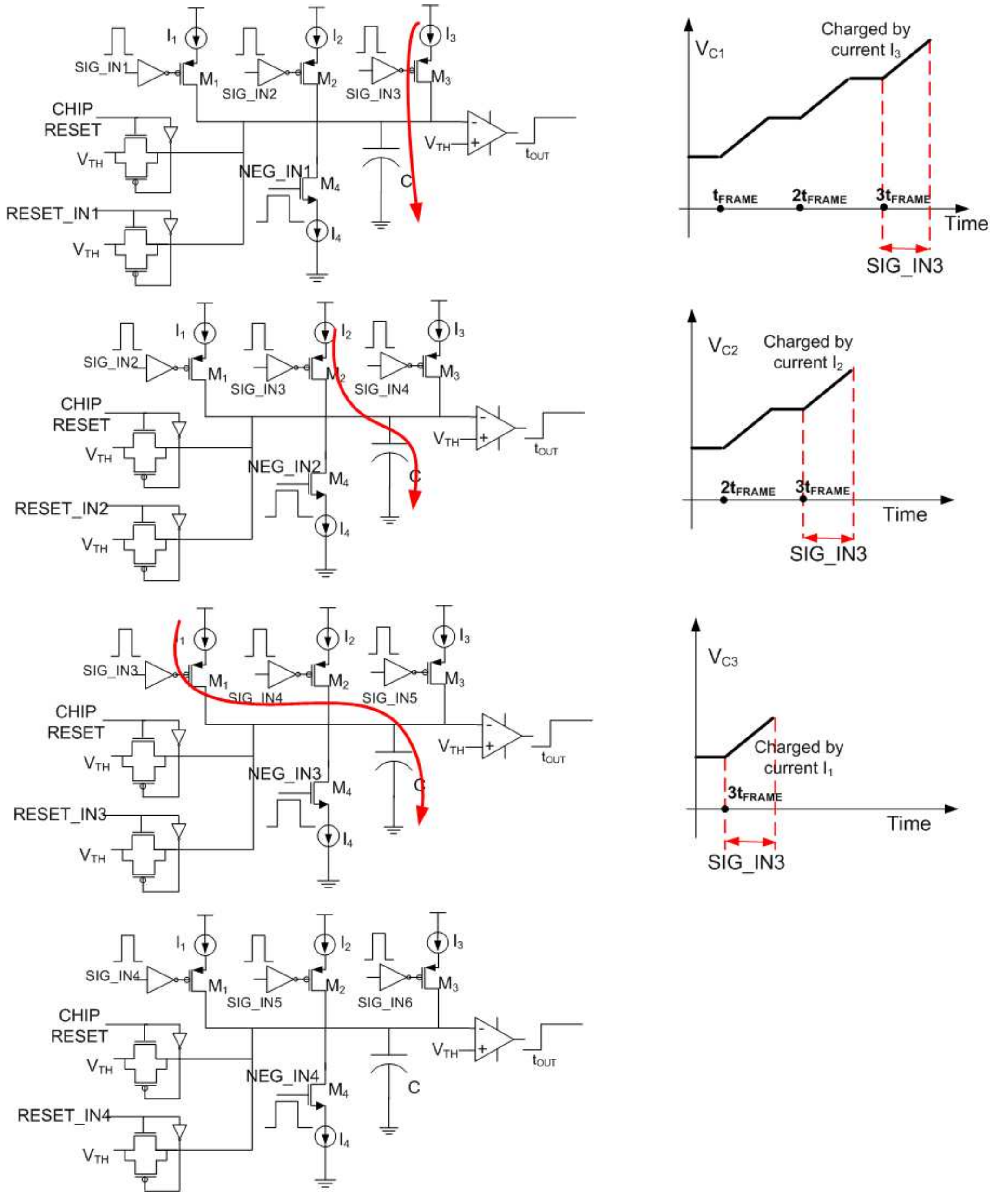

Figure 5–15. State of the FIR filter as input t_3 enters

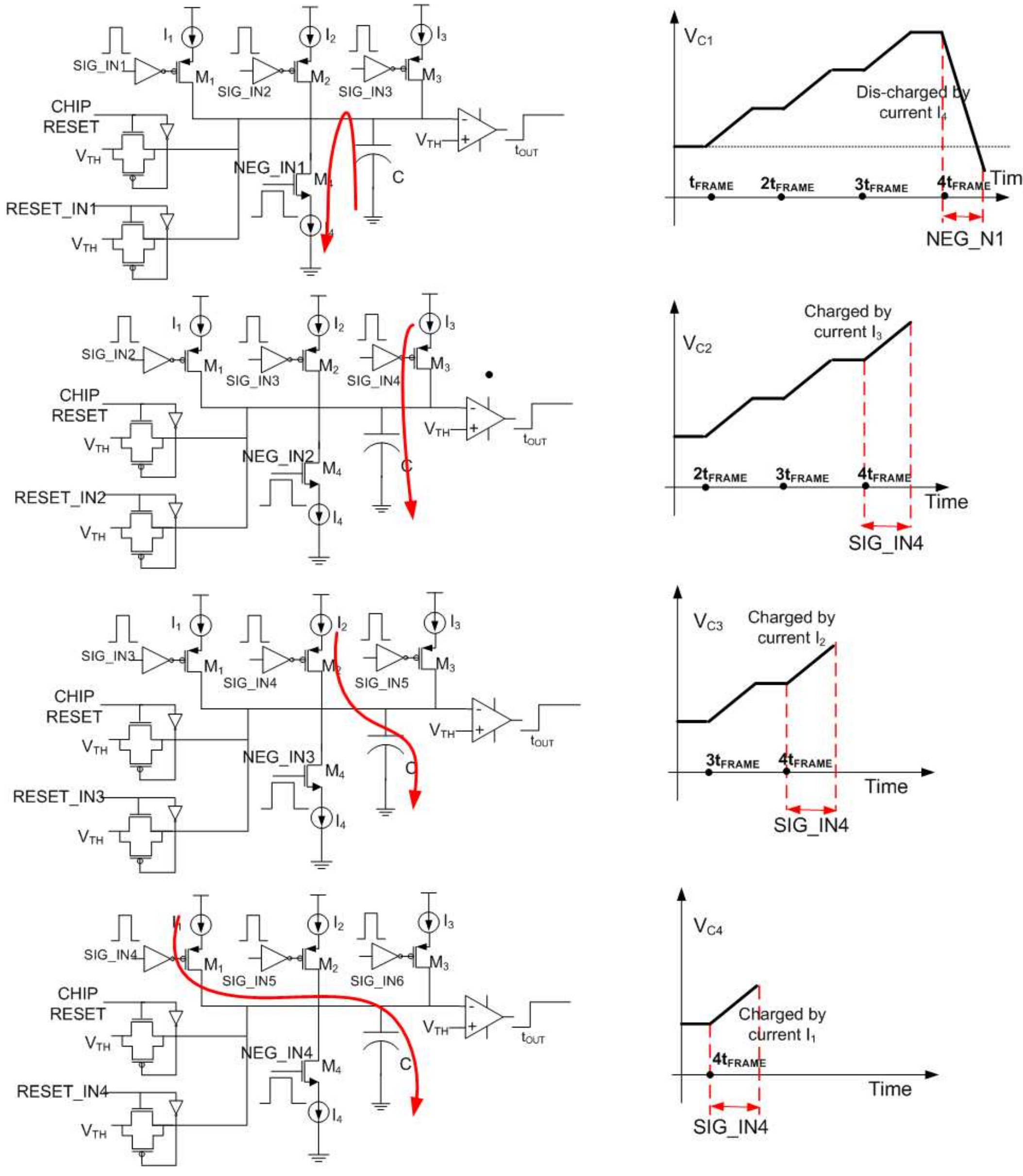

Figure 5–16. State of the FIR filter as input t_4 enters the system and with frame 4 discharging computational block 1

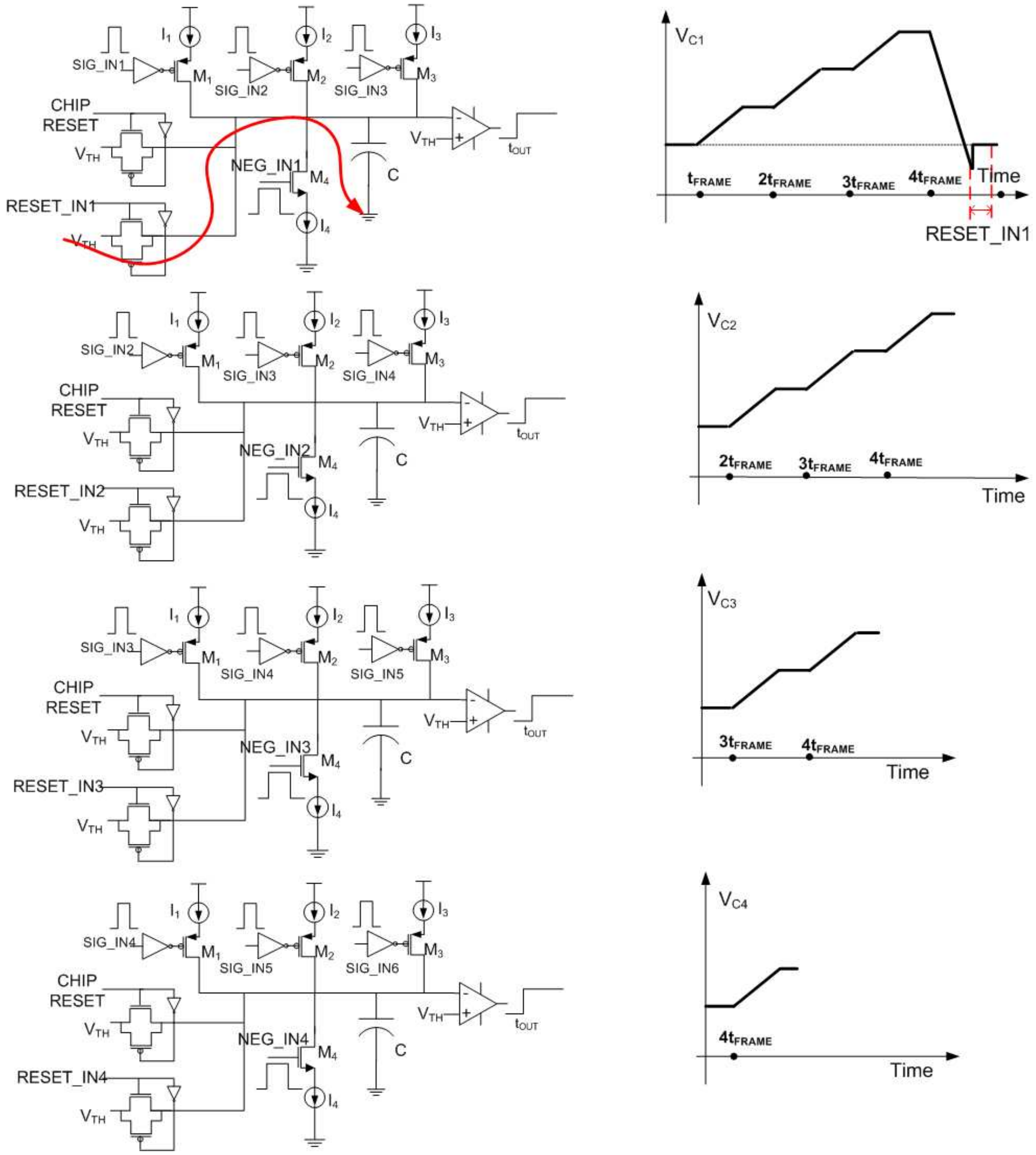

Figure 5–17. State of the FIR filter before frame 5 starts

The output of the computational block was previously derived as,

$$t_{OUT}(4) = (\frac{I_1}{I_4})t_{IN}(1) + (\frac{I_2}{I_4})t_{IN}(2) + (\frac{I_3}{I_4})t_{IN}(3) \tag{5–14}$$

The general expression for this output can be written as,

$$t_{OUT}(n) = (\frac{I_1}{I_4})t_{IN}(n-3) + (\frac{I_2}{I_4})t_{IN}(n-2) + (\frac{I_3}{I_4})t_{IN}(n-1) \tag{5–15}$$

Taking z-transform of this output, we would get

$$\begin{aligned} t_{OUT}(z) &= (\frac{I_1}{I_4})t_{IN}(z)z^{-3} + (\frac{I_2}{I_4})t_{IN}(z)z^{-2} + (\frac{I_3}{I_4})t_{IN}(z)z^{-1} \\ \frac{t_{OUT}(z)}{t_{IN}(z)} &= (\frac{I_1}{I_4})z^{-3} + (\frac{I_2}{I_4})z^{-2} + (\frac{I_3}{I_4})z^{-1} \\ H(z) &= (\frac{I_1}{I_4})z^{-3} + (\frac{I_2}{I_4})z^{-2} + (\frac{I_3}{I_4})z^{-1} \end{aligned} \tag{5–16}$$

Substituting the weights in Eq. 5–16 we get,

$$H(z) = 0.302z^{-3} + 0.4z^{-2} + 0.302z^{-1} \tag{5–17}$$

The poles and zeros of this FIR filter are shown in Figure 5–18. The sampling frequency chosen for the simulations is $100KHz$. The FIR filter's magnitude response and phase response are shown in Figures 5–19 and 5–20 respectively. From the plots, we see that the choice of coefficients $I_1 = 60.4nA$, $I_2 = 80nA$, $I_3 = 60.4nA$ and $I_4 = 100nA$ has tuned the FIR filter to function as a low pass filter with a cut-off frequency of $\approx 10KHz$, stop-band attenuation of $\approx 20dB$ and a passband attenuation of $\approx 1dB$. Similarly, by choosing different values (either positive or negative) we can come up with high pass, band pass and notch filters. It is important to note that the architecture shown in Figure 5–10 can handle only positive currents. If negative currents are to be handled, the circuit architecture would have to be altered.

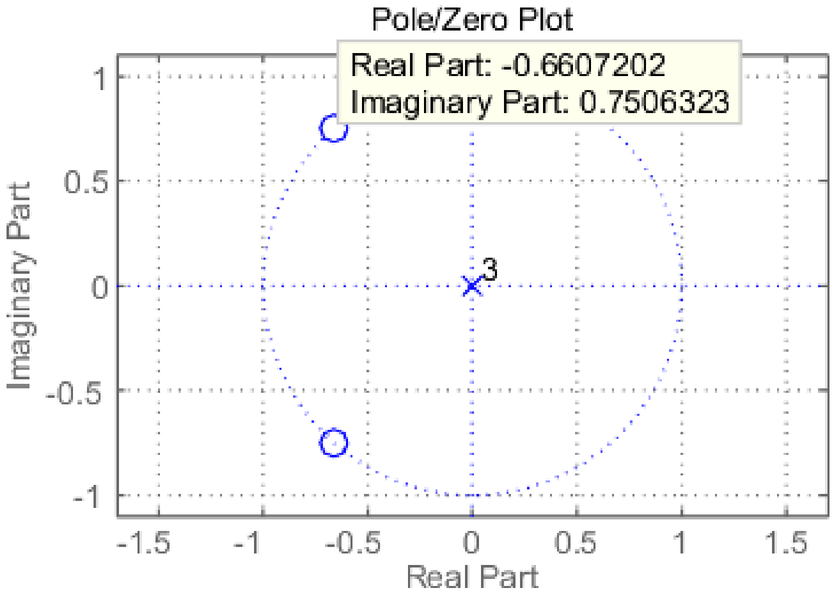

Figure 5–18. Pole-zero plots of the FIR filter

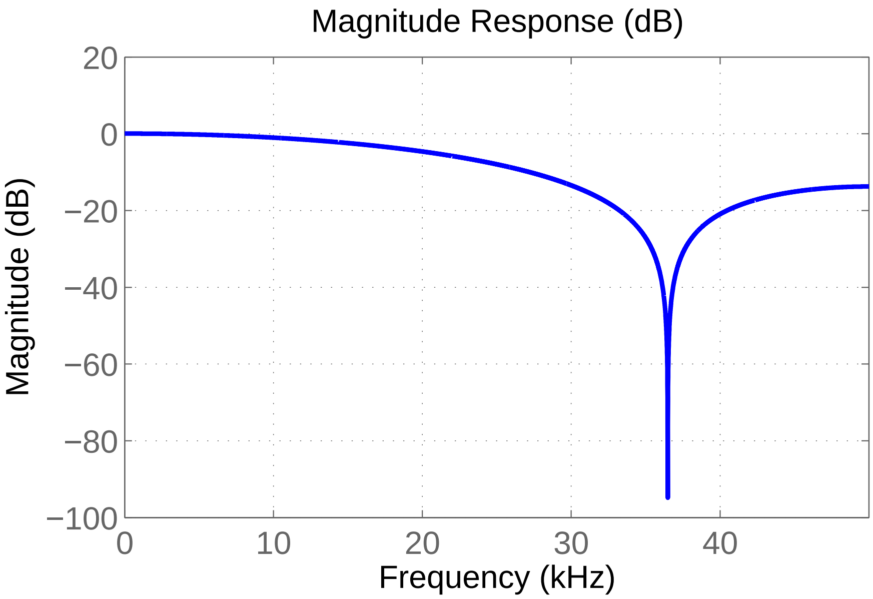

Figure 5–19: Time-mode FIR filter's magnitude response (sampling freq = 100 kHz)

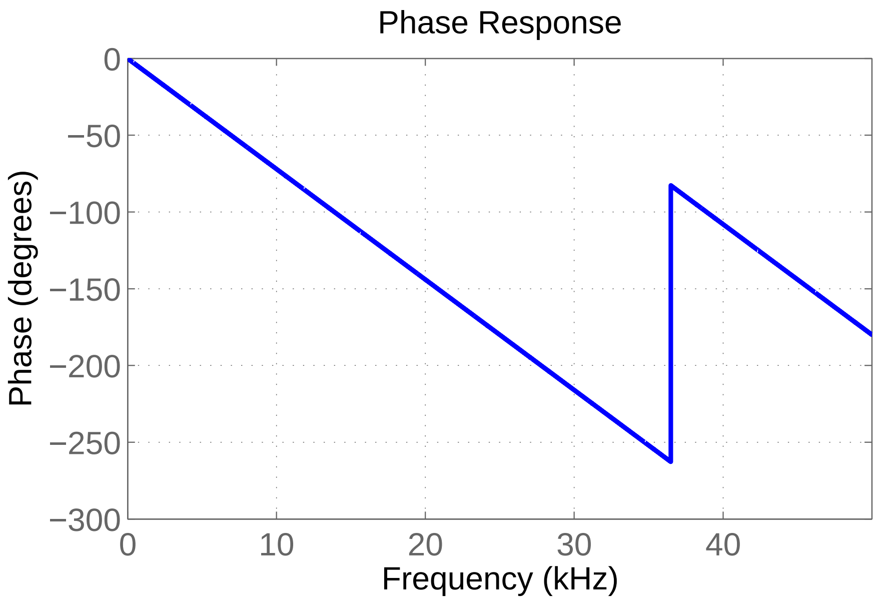

Figure 5–20. Time-mode FIR filter's phase response

The FIR filter's input and output waveforms in the time domain and frequency domain are shown in Figures 5–22 and 5–23. From the time domain waveform, we see that the output waveform is essentially a delayed version of the input waveform (as would be expected from a low pass filter) with the delay being equal to 2 samples = $20\mu s$. This 2-sample delay is also confirmed by the group delay plot shown in Figure 5–21 where the group delay of the FIR filter was obtained as 2 samples. In the freq domain, we see that the FIR filter attenuates the input signal's energy for frequencies above 10kHz by $\approx 20dB$.

Figure 5–24 shows the Cadence simulation results for a 3-tap time-mode FIR filter. The various plots shown in the figure are: Input frame train, Input time step train, outputs of the four computational blocks and the final output from the FIR filter. We can see from the figure that for $t_1 = 10\mu s$, $t_2 = 40\mu s$, $t_3 = 70\mu s$, from simulations we get $t_{OUT} = 41.1\mu s$. From hand calculations, we expect $t_{OUT} = 40.2\mu s$. The small difference between the expected output time and the expected output time should be attributed to the delay of the digital blocks processing the inputs, delay of the comparator and the leakage currents

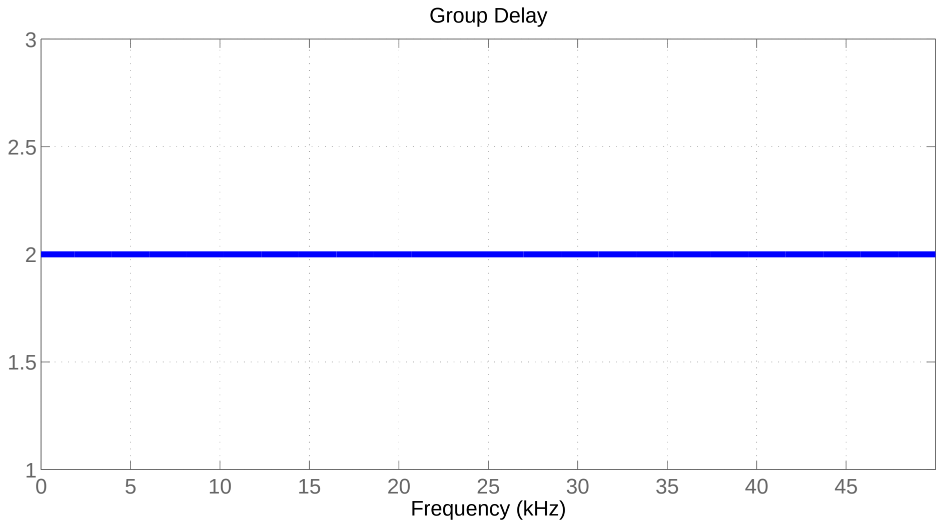

Figure 5–21. Time-mode FIR filter's group delay

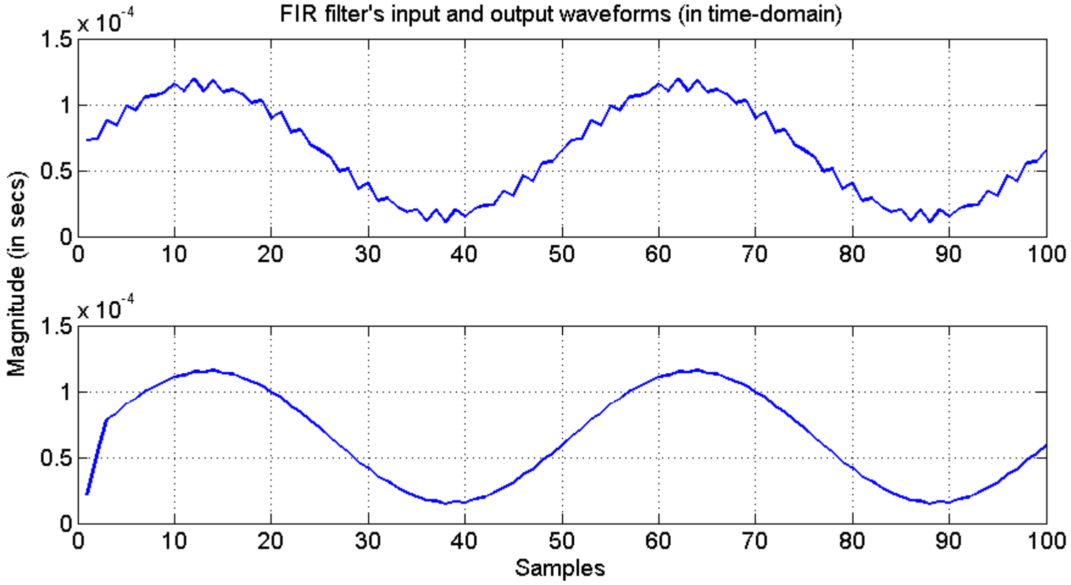

Figure 5–22. Time-mode FIR filter's input and output waveforms (in time domain)

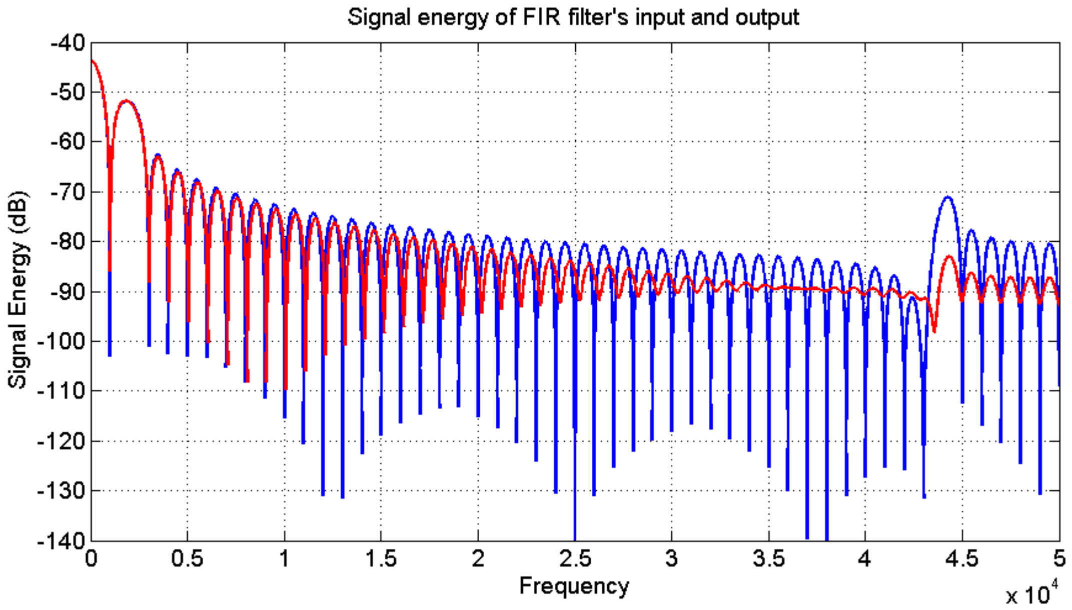

Figure 5–23. Energy of FIR filter's input and output signals

charging the capacitance. The DC power consumed by the FIR filter is $\approx 89.45\mu W$. The speed of operation of the filter is $\approx 25kHz$. There is an interesting trade-off between speed and input/output dynamic range in time-mode FIR filters. As information is represented in time, to accommodate a very high dynamic range in the inputs we may have to increase the duration of a frame. This in turn means that the sampling frequency is reduced. Therefore, the speed of operation is reduced. This, we see that there is a direct tradeoff between the speed of operation of the FIR filter and its input/output dynamic range.

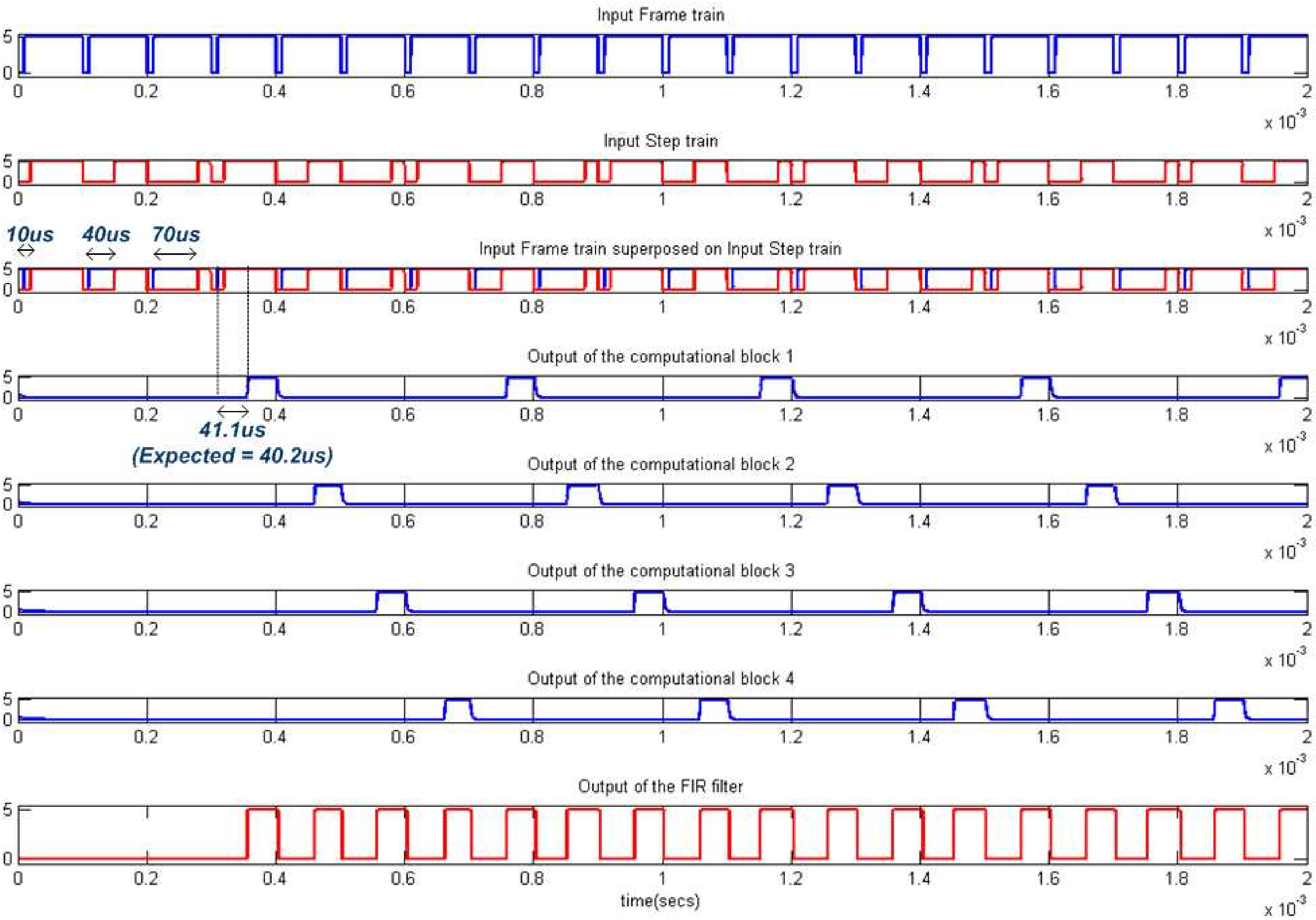

Figure 5–24. Cadence simulation results for the time-mode 3-bit FIR filter

5.4 Signal-to-Noise Ratio/Dynamic Range Analysis

The output of the basic computational block of the FIR filter is given by

$$t_{OUT} = (\frac{I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3 \tag{5–18}$$

There are 4 noise sources that dominate the noise performance of this circuit.

1. Shot noise of the DC current source I_1.
2. Shot noise of the DC current source I_2.
3. Shot noise of the DC current source I_3.
4. Shot noise of the DC current source I_4.

These noise sources are uncorrelated and therefore we can consider the impact of each of these noise sources on the output of the circuit. Also, as previously done for the SNR analysis of the prototype weighted average circuit we neglect the noise contributed by the comparator and we also neglect the noise of the reset transistors (as their contributions to the output noise would be smaller compared to the noise contributions of the DC current sources).

5.4.1 Noise in t_{OUT} due to Noise in Current Source I_1

Let us assume that current source I_1 is noisy with a noise current of ΔI_1 (and a variance of $\overline{\Delta I_1{}^2}$). Therefore, the total current from the current source is given by $I_1 + \Delta I_1$. Since currents $I_1 + \Delta I_1$, I_2, I_3 are charging the capacitor and I_4 is discharging the capacitor, the new output from the weighted average circuit is given by,

$$t_{OUT1} = (\frac{I_1 + \Delta I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3 \tag{5–19}$$

The noise at the output Δt_{OUT1} can be calculated as shown below:

$$\begin{aligned}\Delta t_{OUT1} &= t_{OUT1} - t_{OUT} \\ &= ((\frac{I_1 + \Delta I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3) - ((\frac{I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3) \\ &= (\frac{\Delta I_1}{I_4})t_1 \qquad (5\text{–}20)\end{aligned}$$

The variance of this noise is given by

$$\overline{\Delta t_{OUT1}{}^2} = (\frac{\overline{\Delta I_1{}^2}}{I_4{}^2})t_1^2 \qquad (5\text{–}21)$$

5.4.2 Noise in t_{OUT} due to Noise in Current Source I_2

Assuming that current source I_2 is noisy with a noise current of ΔI_2(and a variance of $\overline{\Delta I_2{}^2}$), the total current from the current source is given by $I_2 + \Delta I_2$. The new output from the weighted average circuit for this case is given by,

$$t_{OUT2} = (\frac{I_1}{I_4})t_1 + (\frac{I_2 + \Delta I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3 \qquad (5\text{–}22)$$

The noise at the output Δt_{OUT2} can be calculated as shown below:

$$\begin{aligned}\Delta t_{OUT2} &= t_{OUT2} - t_{OUT} \\ &= ((\frac{I_1}{I_4})t_1 + (\frac{I_2 + \Delta I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3) - ((\frac{I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3) \\ &= (\frac{\Delta I_2}{I_4})t_2 \qquad (5\text{–}23)\end{aligned}$$

The variance of this noise is given by

$$\overline{\Delta t_{OUT2}{}^2} = (\frac{\overline{\Delta I_2{}^2}}{I_4{}^2})t_2^2 \qquad (5\text{–}24)$$

5.4.3 Noise in t_{OUT} due to Noise in Current Source I_3

Let us assume that current source I_3 is noisy with a noise current of ΔI_3(and a variance of $\overline{\Delta I_3{}^2}$). Therefore, the total current from the current source is given by $I_3 + \Delta I_3$. The variance of noise at the output can be derived by following similar steps as in the above two cases. The variance of this noise is given by

$$\overline{\Delta t_{OUT3}{}^2} = (\frac{\overline{\Delta I_3{}^2}}{I_4{}^2}) t_3^2 \tag{5–25}$$

5.4.4 Noise in t_{OUT} due to Noise in Current Source I_4

Let us assume that current source I_4 is noisy with a noise current of ΔI_4 (and a variance of $\overline{\Delta I_4{}^2}$). Therefore, the total current from the current source is given by $I_4 + \Delta I_4$. Since currents I_1, I_2, I_3 charge the capacitor and $I_4 + \Delta I_4$ discharge the capacitor, the new output from the weighted average circuit is given by,

$$t_{OUT1} = (\frac{I_1}{I_4 + \Delta I_4}) t_1 + (\frac{I_2}{I_4 + \Delta I_4}) t_2 + (\frac{I_3}{I_4 + \Delta I_4}) t_3 \tag{5–26}$$

The noise at the output Δt_{OUT4} can be calculated as shown below:

$$\begin{aligned} \Delta t_{OUT4} &= t_{OUT4} - t_{OUT} \\ &= (\frac{I_1 t_1 + I_2 t_2 + I_3 t_3}{I_4 + \Delta I_4}) - (\frac{I_1 t_1 + I_2 t_2 + I_3 t_3}{I_4}) \\ &= (I_1 t_1 + I_2 t_2 + I_3 t_3)(\frac{-\Delta I_4}{I_4 (I_4 + \Delta I_4)}) \\ &\approx (I_1 t_1 + I_2 t_2 + I_3 t_3)(\frac{-\Delta I_4}{I_4^2}) \\ &= \frac{(I_1 t_1 + I_2 t_2 + I_3 t_3)}{I_4}(\frac{-\Delta I_4}{I_4}) \\ &= t_{OUT}(\frac{-\Delta I_4}{I_4}) \end{aligned} \tag{5–27}$$

The variance of this noise is given by

$$\overline{\Delta t_{OUT4}{}^2} = (\frac{\overline{\Delta I_4{}^2}}{I_4{}^2}) t_{OUT}^2 \tag{5–28}$$

Therefore, the total noise at the output of the averaging block is given by

$$\begin{aligned} \overline{\Delta t_{OUT}{}^2} &= \overline{\Delta t_{OUT1}{}^2} + \overline{\Delta t_{OUT2}{}^2} + \overline{\Delta t_{OUT3}{}^2} + \overline{\Delta t_{OUT4}{}^2} \\ &= (\frac{\overline{\Delta I_1{}^2}}{I_4{}^2}) t_1^2 + (\frac{\overline{\Delta I_2{}^2}}{I_4{}^2}) t_2^2 + (\frac{\overline{\Delta I_3{}^2}}{I_4{}^2}) t_3^2 + (\frac{\overline{\Delta I_4{}^2}}{I_4{}^2}) t_{OUT}^2 \end{aligned} \tag{5–29}$$

As previously mentioned in Chapter 2, the shot noise due to a current source I is given by,

$$\overline{\Delta I^2} = \frac{eI}{\tau} \tag{5–30}$$

where τ is the time during which the current source is ON and contributes to the output and e is the charge of an electron.

The shot noises of current sources I_1 to I_4 are given by, $\overline{\Delta I_1}^2 = \frac{eI_1}{t_1}$, $\overline{\Delta I_2}^2 = \frac{eI_2}{t_2}$, $\overline{\Delta I_3}^2 = \frac{eI_3}{t_3}$ and $\overline{\Delta I_4}^2 = \frac{eI_4}{t_{OUT}}$. Substituting these noise relations into Eq. 5–29, we get

$$\begin{aligned}
\overline{\Delta t_{OUT}}^2 &= (\frac{\overline{\Delta I_1}^2}{{I_4}^2})t_1^2 + (\frac{\overline{\Delta I_2}^2}{{I_4}^2})t_2^2 + (\frac{\overline{\Delta I_3}^2}{{I_4}^2})t_3^2 + (\frac{\overline{\Delta I_4}^2}{{I_4}^2})t_{OUT}^2 \\
&= \frac{\frac{eI_1}{t_1}t_1^2}{I_4^2} + \frac{\frac{eI_2}{t_2}t_2^2}{I_4^2} + \frac{\frac{eI_3}{t_3}t_3^2}{I_4^2} + \frac{\frac{eI_4}{t_{OUT}}t_{OUT}^2}{I_4^2} \\
&= \frac{eI_1t_1}{I_4^2} + \frac{eI_2t_2}{I_4^2} + \frac{eI_3t_3}{I_4^2} + \frac{eI_4t_{OUT}}{I_4^2} \\
&= \frac{eI_4t_{OUT}}{I_4^2} + \frac{et_{OUT}}{I_4} \\
&= 2\frac{et_{OUT}}{I_4}
\end{aligned} \tag{5–31}$$

The signal-to-noise ratio for the computational block and the FIR filter is given by,

$$\begin{aligned}
SNR &= 10\log_{10}(\frac{t_{OUT}^2}{\overline{\Delta t_{OUT}}^2}) \\
&= 10\log_{10}(\frac{t_{OUT}^2}{2\frac{et_{OUT}}{I_4}}) \\
&= 10\log_{10}(\frac{t_{OUT}I_4}{2e})
\end{aligned} \tag{5–32}$$

The equation can also be written as,

$$\begin{aligned}
SNR &= 10\log_{10}(\frac{t_{OUT}I_4}{2e}) \\
&= 10\log_{10}(\frac{I_1t_1 + I_2t_2 + I_3t_3}{2e}) \\
&= 10\log_{10}(\frac{C(V_C^3 - V_{TH})}{2e})
\end{aligned} \tag{5–33}$$

Substituting $t_{OUT} = 40.1\mu s$, $I_4 = 200nA$ in Eq. 5–31, we get the output noise variance as $\overline{\Delta t_{OUT}}^2 = 64.16as^2$ and the rms noise at the output as $8ns$. From Eq. 5–32, we get SNR$\approx 64dB$.

Since the dynamic range is given by the ratio of the maximum output (and the maximum t_{OUT} is equal to the length of the frame) to the minimum output (minimum t_{OUT} given by the noise floor), we get

$$DR = 20\log_{10}(\frac{t_{frame}}{\Delta t_{OUT}}) = 20\log_{10}(\frac{t_{OUT}}{\Delta t_{OUT}}) = SNR \tag{5–34}$$

Therefore, the DR of the FIR filter is also equal to $64dB$.

5.5 Performance of the FIR Filter under Input Time Jitter

The output of the basic computational block of the FIR filter is given by

$$t_{OUT} = (\frac{I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3 \tag{5–35}$$

Assuming that input t_1 has a time jitter Δt_1. With this time jitter, the output of the computational block would become,

$$t_{OUT1} = (\frac{I_1}{I_4})(t_1 + \Delta t_1) + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3 \tag{5–36}$$

Therefore, the noise at the output is given by

$$\begin{aligned}\Delta t_{OUT} &= t_{OUT1} - t_{OUT}\\ &= ((\frac{I_1}{I_4})(t_1 + \Delta t_1) + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3) - ((\frac{I_1}{I_4})t_1 + (\frac{I_2}{I_4})t_2 + (\frac{I_3}{I_4})t_3)\\ &= \frac{I_1 \Delta t_1}{I_4}\end{aligned} \tag{5–37}$$

The output noise variance is given by,

$$\overline{\Delta t_{OUT}}^2 = \frac{I_1^2}{I_4^2}\overline{\Delta t_1}^2 \tag{5–38}$$

For the weight chosen in our simulations $\frac{I_1}{I_4} = 0.302$, $\overline{\Delta t_{OUT}}^2 \approx 0.09\overline{\Delta t_1}^2$. Thus, we see that the effect of the input jitter is not pronounced at the output when the weights have a magnitude less than 1. If the magnitude of the weight is more than 1, the output jitter would be more than the input jitter.

5.6 Advantages of Time-Mode FIR Filters

- Typical analog FIR filter architectures [33] involve the use of Input Buffers, Track and Hold Circuits, Unity Gain Amplifiers, Multiplexers, Level Shifters and Multipliers or DACs. The architecture of the time-mode FIR filters is very simple and would occupy smaller area when compared to these analog FIR architectures. A 3-tap FIR filter needs only 4 computational blocks. In general, a N tap filter would require only $N + 1$ computational blocks.

- Since information is represented in time, the FIR filter can support very high input and output dynamic range.

- Since the time-mode FIR architecture provides a SNR of 60dB, it is very noise robust.

- If the magnitude of the weights chosen in Time-mode FIR architectures are less than one, then the FIR architecture reduces the effect of the input jitter on the output.

- The time-mode FIR architectures are very power efficient. The 3-tap time-mode FIR filter simulated above consumes just $89.45\mu W$ as compared to architectures that consume power in the order of mW.

- Matching of the capacitance is not an issue because the actual value of the capacitance does not affect the output expression (Eq. 5–10).

- Since the input and output are in time, the fan-in and fan-out for the time-mode FIR filter can be very high.

5.7 Limitations of Time-Mode FIR Filters

- Since information is represented in time, having a high dynamic range in the inputs (which means having a long frame) would inherently reduce the sampling frequency with which the FIR can function and hence the speed of the filter. Since the noise floor of this architecture is approximately $8ns$, these circuits can reach speeds upto $\approx 100MHz$, but at such high speeds these circuits would support very small dynamic range of inputs and outputs.

- With more taps, though the complexity of the analog portion of the architecture (computational blocks) reduces, more digital circuitry is needed to extract the analog inputs and cycle the inputs to feed the analog architecture.

- Matching of the current sources is very important for the accurate operation of the computational blocks of the FIR filter. Appropriate matching techniques have to be employed during the layout of the filter to make sure the current sources match properly.

- Since the switching of the digital switches can cause charge injection into the capacitance of the computational blocks and the presence of fast rising inputs on the gates of these digital switches can cause direct capacitive coupling to the capacitors, well known charge injection and capacitive coupling techniques can be used to reduce these effects, albeit with additional compexity in the architecture.

CHAPTER 6
NON-LINEAR TIME-MODE COMPUTATION

We have explored two ways of implementing non-linear arithmetic:

1. Implementing non-linear arithmetic using a time-mode multi-layer perceptron.

2. Implementing non-linear arithmetic by introducing a non-linearity in the existing linear computational blocks.

6.1 Implementing Non-Linear Arithmetic by Introducing Non-Linearity in the Existing Linear Computational Blocks

6.1.1 Time-Mode Multiplication

In this section, we will discuss how to implement the multiplication $t_1 t_2$ by introducing non-linearity in a scalar multiplication circuit. To retain proper dimensions, we must scale (divide) the output $t_1 t_2$ by a time constant τ.

The schematic, the input/output timings and the capacitor voltage waveform of a scalar multiplication circuit are shown in Figure 6–1. Let's neglect the operation of this circuit during frame 1. As shown in the figure, we see that the input t_2 is defined with respect to frame 2 (starting at time t_{F2} . The output of the circuit is given by,

$$t_{OUT} = \frac{I_1 t_2}{I_2} \tag{6–1}$$

This output t_{OUT} is defined with respect to reference frame 3 (starting at time t_{F3}). During frame 1(starting at time t_{F1}) if we can make I_1 a linear function of t_1, say $I_1 = kt_1$ where k is a constant, then

$$t_{OUT} = (\frac{k}{I_2})t_1 t_2 \tag{6–2}$$

Thus, we can achieve the non-linear multiplication function using linear computation circuits.

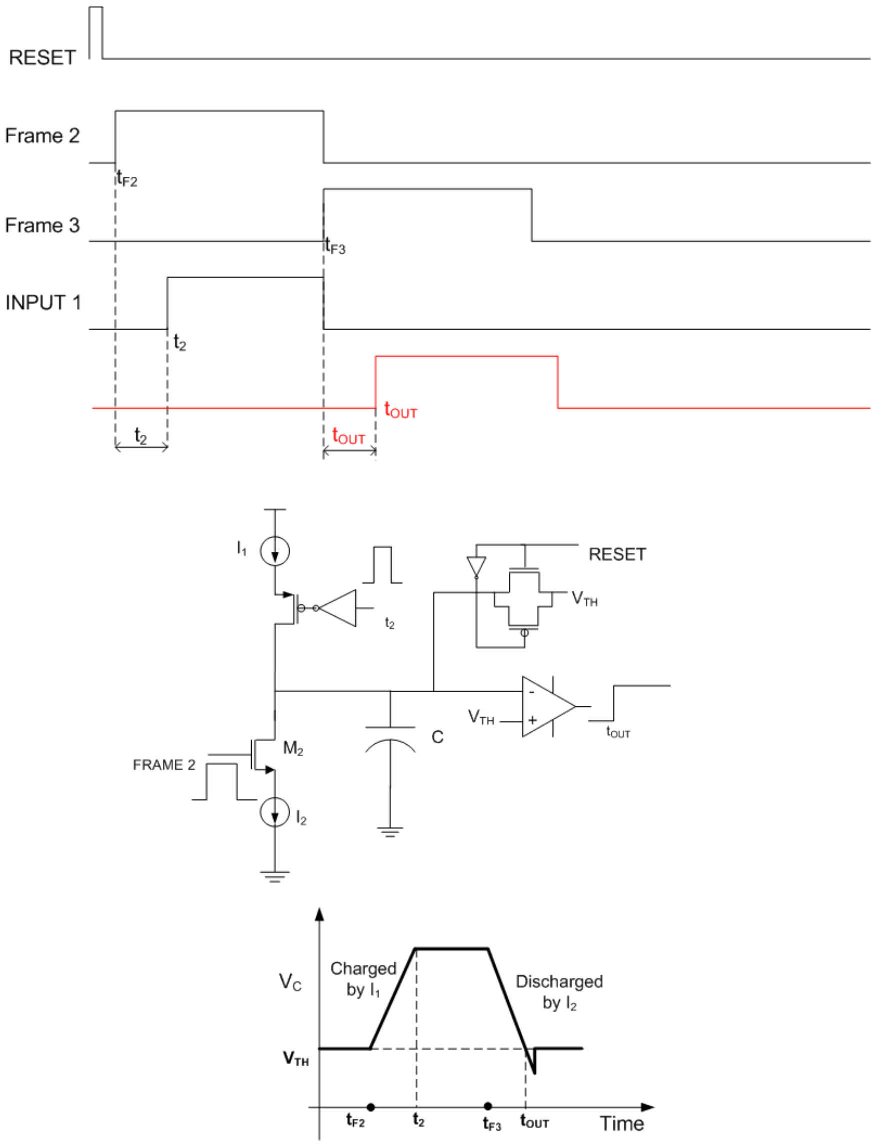

Figure 6–1. Scalar multiplication circuit

The complete 2-input time-mode multiplication circuit is shown in Figure 6–3 and the input/output timing diagrams are shown in Figure 6–2. As shown in Figure 6–3, initially the two signals - input t_1 and reference frame 1 are XORed and this XOR output controls the current source I_X charging the capacitor C_1. Thus, the time difference between t_{F1} (reference frame 1) and the input t_1 is converted to voltage across the capacitor C_1 where $V_{C1} = \frac{I_X t_1}{C_1}$. Assuming no leakage from the capacitance and zero leakage current from the switches, voltage

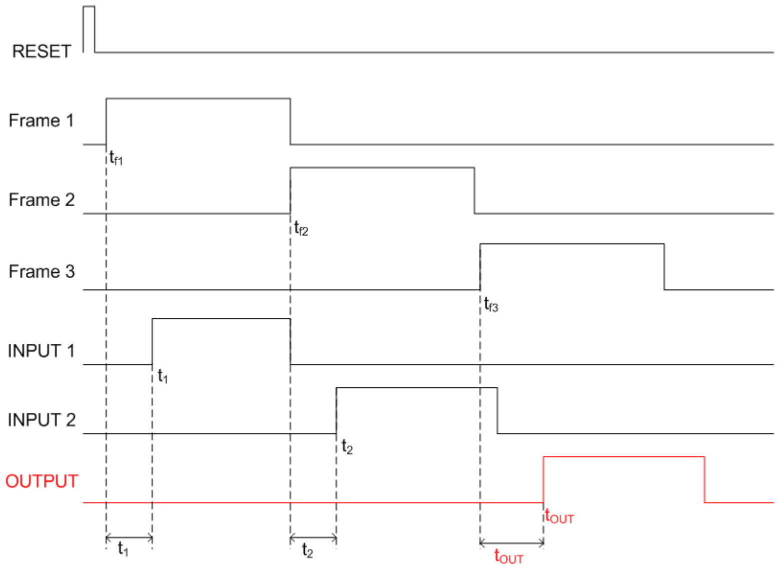

Figure 6–2. Timing details of the 2-input time-mode multiplier

V_{C1} would remain constant until the end of the third frame (The multiplier would have fired an output during the third frame. At the end of the third frame, the capacitors are reset to their initial values).

Once reference frame 2 starts, the transmission gate connected to the capacitor C_1 is turned ON. The transconductance amplifier now produces a current output,

$$\begin{aligned} I_{OUT} &= g_m V_{C1} \\ &= g_m\left(\frac{I_X t_1}{C_1}\right) \end{aligned} \tag{6–3}$$

This I_{OUT} acts as the current source I_1 in the scalar multiplication circuit shown in Figure 6–1. Therefore, the output from the 2-input time-mode multiplication circuit is given by,

$$\begin{aligned} t_{OUT} &= \frac{I_{OUT} t_2}{I_2} \\ &= \frac{g_m(\frac{I_X t_1}{C_1}) t_2}{I_2} \\ &= \frac{t_1 t_2}{(\frac{I_2 C_1}{g_m I_X})} \end{aligned} \tag{6–4}$$

Eq. 6–4 shows how the 2-input time-mode multiplication circuit produces the desired output $t_1 t_2$ scaled by the expression $\frac{I_2 C_1}{g_m I_X}$.

To arrive at the expression shown in Eq. 6–4, we have made the following assumptions.

- Input t_1 arrives earlier than input t_2.
- Time difference between the t_{F1} and t_1 doesn't saturate the capacitor C_1.

The accuracy of the output depends on the linearity of the transconductance amplifier. Since the output is valid only in the linear range of the transconductance, the dynamic range of the input t_1 supported by the multiplier depends on the linearity of the transconductance. There are many ways to improve the linearity of the transconductance and those techniques can be employed in this circuit to improve the dynamic range of inputs/outputs supported.

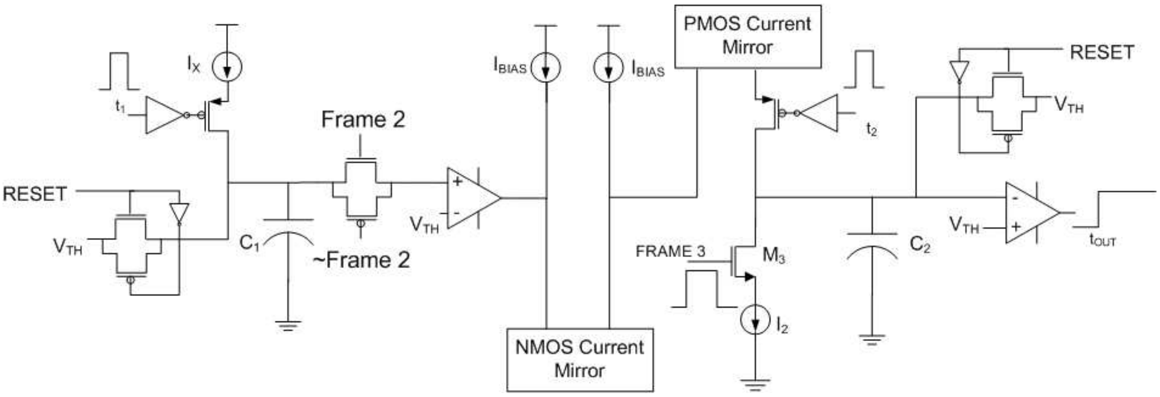

Figure 6–3. Schematic of the 2-input time-mode multiplier

6.1.2 Time-Mode Division

In this section, we will discuss how to implement $\frac{t_2}{t_1}$ by introducing non-linearity in a scalar multiplication circuit. To retain same dimensions of time, we must scale (multiply) the output $\frac{t_2}{t_1}$ by a time constant τ.

If we can make I_2 a linear function of t_1, say $I_2 = kt_1$ where k is a constant and substitute it in the output of the scalar multiplication circuit $t_{OUT} = \frac{I_1 t_2}{I_2}$, we would get,

$$t_{OUT} = (\frac{I_1}{k})\frac{t_2}{t_1} \tag{6–5}$$

Thus, we can achieve the non-linear division function using linear computation circuits.

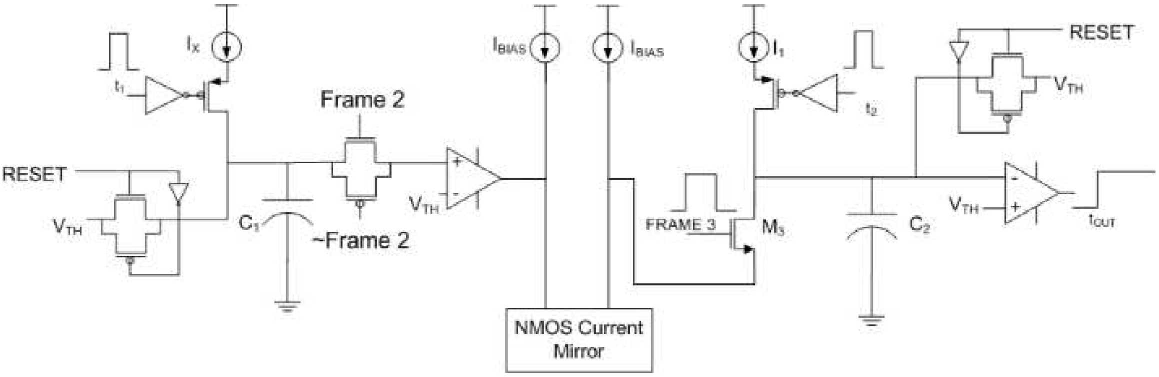

Figure 6–4. Schematic of the 2-input time-mode divider

The complete 2-input time-mode division circuit is shown in Figure 6–4 and the input/output timing diagrams are shown in Figure 6–2. As in the multiplier circuit, the input t_1 is converted to voltage across the capacitor C_1 where $V_{C1} = \frac{I_X 1 t_1}{C_1}$. The current out from the transconductance amplifier when reference frame 2 starts is given by,

$$\begin{aligned} I_{OUT} &= g_m V_{C1} \\ &= g_m(\frac{I_X t_1}{C_1}) \end{aligned} \tag{6–6}$$

This I_{OUT} acts as the current source I_2 in the scalar multiplication circuit shown in Figure 6–1. Therefore, the output from the 2-input time-mode division circuit is given by,

$$\begin{aligned} t_{OUT} &= \frac{I_1 t_2}{I_{OUT}} \\ &= \frac{I_1 t_2}{g_m(\frac{I_X t_1}{C_1})} \\ &= \frac{t_2}{t_1}(\frac{I_1 C_1}{g_m I_X}) \end{aligned} \tag{6–7}$$

Eq. 6–7 shows how the 2-input time-mode division circuit produces the desired output $\frac{t_2}{t_1}$ scaled by the expression $\frac{I_1 C_1}{g_m I_X}$. The accuracy of the output depends on the linearity of the transconductance amplifier as is in the case of the 2-input time-mode multiplier circuit.

An interesting case arises when $t_1 - t_{F1}$ is equal to zero. When $t_1 - t_{F1}$ is equal to zero, there would be no current charging C_1. So, V_{C1} does not change and it stays at its initial voltage V_{TH}. Therefore, the transconductance amplifier would not produce any current, capacitor V_{C2} will not be discharged and there won't be any output from the division circuit. This effectively means that the output from the division circuit is very high (maximum output from the division circuit is equal to the length of a frame because we reset the whole circuit at the end of each frame).

To arrive at the expression shown in Eq. 6–7, we have made the following assumptions.

- Input t_1 arrives earlier than input t_2.
- Time difference between the t_{F1} and t_1 doesn't saturate the capacitor C_1.

6.2 Implementing Non-Linear Arithmetic Using Time-Mode Multi-Layer Perceptron

Implementation of non-linear arithmetic can be done using a feedforward multi-layer perceptron (MLP) as one shown in Figure 6–5. In this chapter, we will first provide a brief introduction to the MLP, an important class of neural networks. We will then discuss a technique to implement a time-mode feed-forward MLP.

6.2.1 Time-Mode Multi-Layer Perceptron

The multilayer perceptrons (MLPs) are an important class of neural networks. Typically, the network consists of a set an input layer, one or mode hidden layers of computational nodes, and an output layer of computational nodes [30]. The input

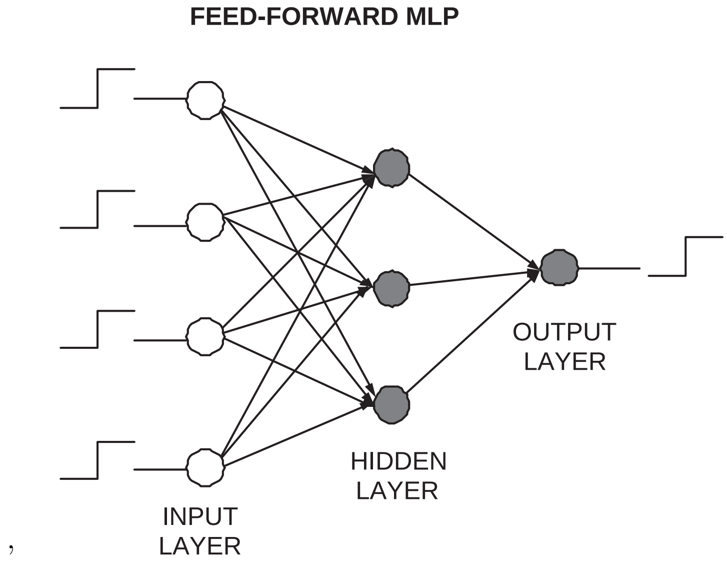

Figure 6–5. Feedforward multi-layer perceptron

signal propagates through the network in a forward direction, on a layer-to-layer basis. Each unit performes a weighted sum on its inputs and then outputs the sum through a non-linear activation function. MLPs have been applied successfully to solve many difficult and diverse problems. Usually they are trained in a supervised manner with the error back-propagation algorithm. This algorithm is based on the error-correction learning rule. As such, it may be viewed as a generalization of the least-mean-square (LMS) algorithm.

In a neural network, the neurons are organized in the form of layers. Figure 6–6 shows the simplest example of a multilayer feedforward network. This network has an input layer of source nodes, a hidden layer of neurons and an output layer of neurons. The network of Figure 6–6 is strictly a feedforward type network since there is no feedback of a signal from the output of any of the neurons to its input. This 2-input MLP has 2 source nodes, 2 hidden neurons and 1 output neuron. We will implement this 2-input MLP using time-mode computational circuits and use this neural network to implement the non-linear time operation: multiplication of two time signals.

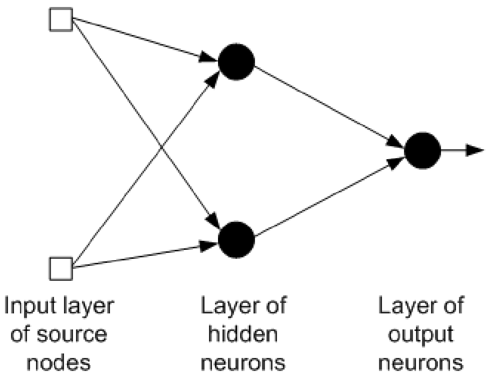

Figure 6–6. Fully connected 2-input feedforward MLP with one hidden layer and one output layer

The neuron is fundamental to the operation of the neural network. The neuron model shown in Figure 6–7 forms the basis for designing aritificial neural networks. There are three basic elements of the neuronal model:

1. A set of synapses, each of which is characterized by a weight or strength of its own. Specifically a signal x_j at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}.

2. An adder for summing the input signals, weighted by the respective synapses of the neuron.

3. An activation function for limiting the amplitude of the output of a neuron.

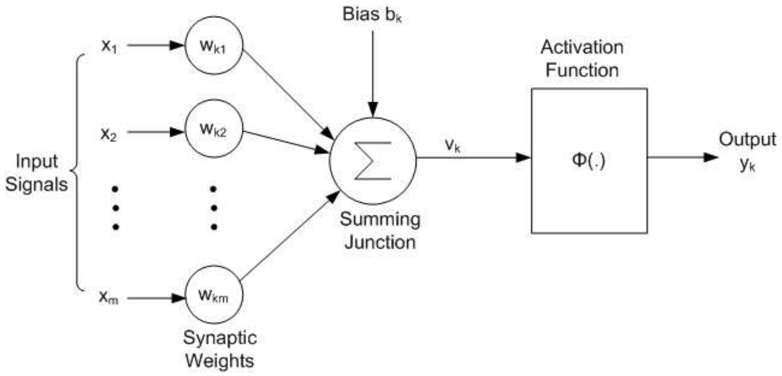

Figure 6–7. Non-linear model of a neuron

In mathematical terms, the neuron can be described by writing the following pairs of equations:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{6–8}$$

and

$$y_k = \varphi(u_k + b_k) \tag{6–9}$$

where x_1, x_2,...,x_m are the input signals; $w_{k1}, w_{k2}, ..., w_{km}$ are the synaptic weights of neuron k; u_k is the linear combiner output due to the input signals; b_k is the bias; $\varphi(.)$ is the activation function; and y_k is the output signal of the neuron. Assuming that the bias b_k is zero, Eq. 6–9 becomes,

$$y_k = \varphi(u_k) \tag{6–10}$$

6.2.2 Hardware Implementation of Time-Mode MLP

The use of analog VLSI for implementing neural architectures such as the multi-layer perceptrons allows the realization of low power and area efficient hardware structures [31]. For several years pulse-stream technique has been used by several researchers for the hardware implementation of artificial neural networks [32], which leads to very complex circuits. As shown in Chapters 2 and 3, the time-mode circuits are extremely simple and very efficient even for complex computations such as the weighted average. Therefore, it would be advantageous to use time-mode circuits in computationally intense structures such as the MLP.

We have developed a time-mode feedforward multi-layer perceptron with one hidden layer to implement a complex non-linear arithmetic operation: multiplication. Training of the weights of the MLP is done by using running a error back-propagation algorithm in a computer and then applying the resultant weights off-chip and programming the weights of the MLP. To implement the MLP using time-mode circuits, we have to first implement the non-linear model of the neuron using time-mode circuits. That is, we have to first implement the Eqs. 6–8 and 6–9.

The circuit shown in Figure 6–8 is very similar in operation to the basic computational block of the FIR filter discussed previously. This circuit performs the following computation:

$$t_{OUT} = \frac{I_1 t_1}{I_3} + \frac{I_2 t_2}{I_3} \tag{6–11}$$

where, weights $\frac{I_1}{I_3}$ and $\frac{I_2}{I_3}$ are applied to signals t_1 and t_2 respectively and summed together. In general for n inputs, we would get

$$t_{OUT} = \sum_{j=1}^{m} (\frac{I_j}{I_3}) t_j \tag{6–12}$$

Therefore, the circuit shown in Figure 6–8 can be used to perform the computation mentioned in Eq. 6–8.

The activation function denoted by $\varphi(v)$, defines the output of a neuron in terms of the induced local field v. Traditionally, three basic types of activation functions have been used: the threshold function, the piecewise-linear function and the sigmoid function. The most commonly used form of non-linearity is the sigmoidal non-linearity defined by the logistic function:

$$y_j = \frac{1}{1 + e^{-x_j}} \tag{6–13}$$

where x_j is the weighted sum of all synaptic inputs plus the bias and y_j is the output of the neuron. The presence of non-linearities ensures that the input-output relationship of the network is not the same as the single-layer perceptron. Since we have been successful in building linear time-mode computational circuits, we choose a piecewise-linear function shown in Figure 6–9 to implement the sigmoid function.

Therefore, by connecting the circuits in Figures 6–8 and 6–9, we can design a non-linear neuron shown in Figure 6–7. By connecting multiple neurons together as shown in Figure 6–6, we can implement a 2-layer multi-layer perceptron.

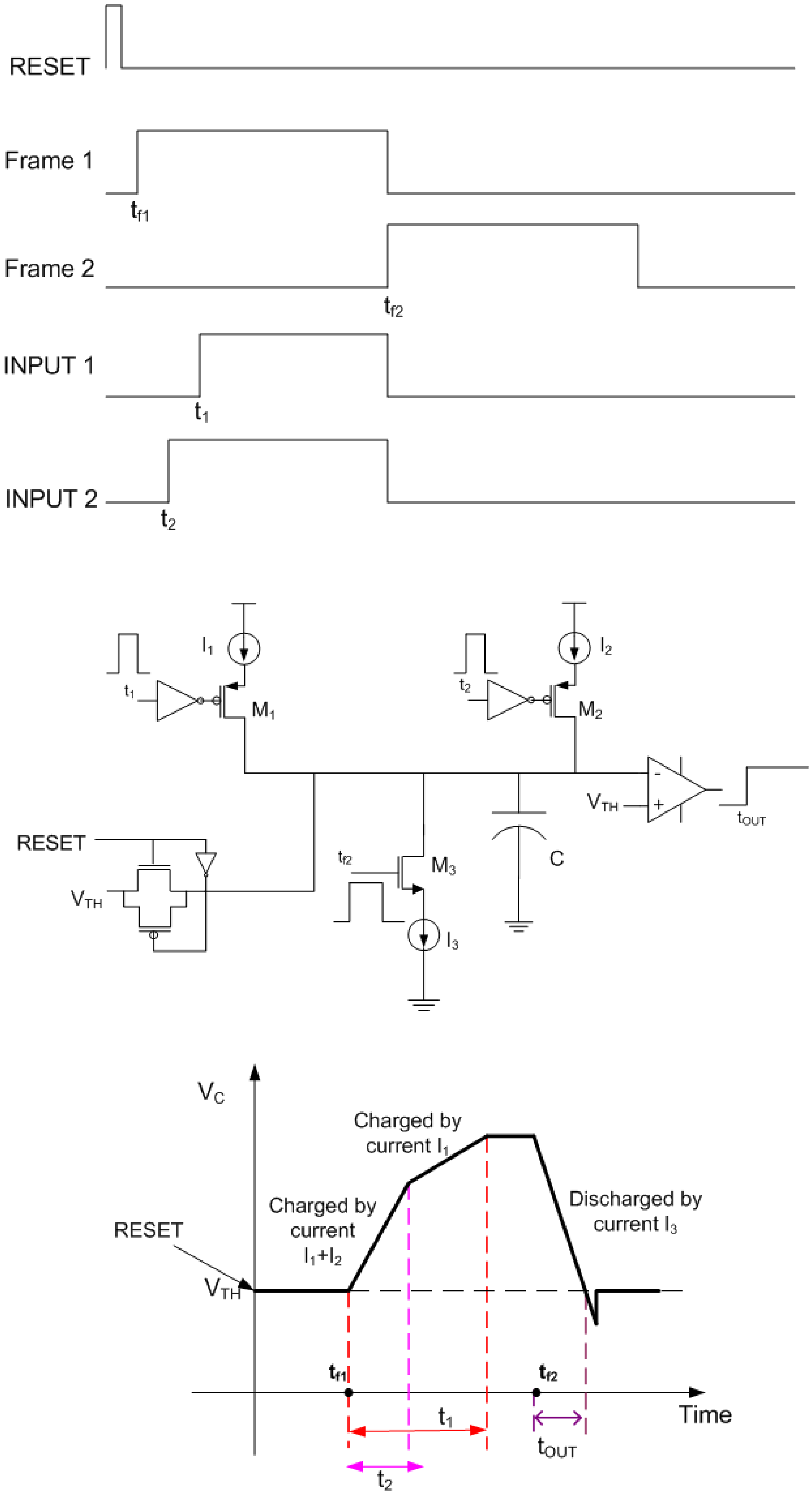

Figure 6–8. Time-mode scalar multiplication and summing circuit

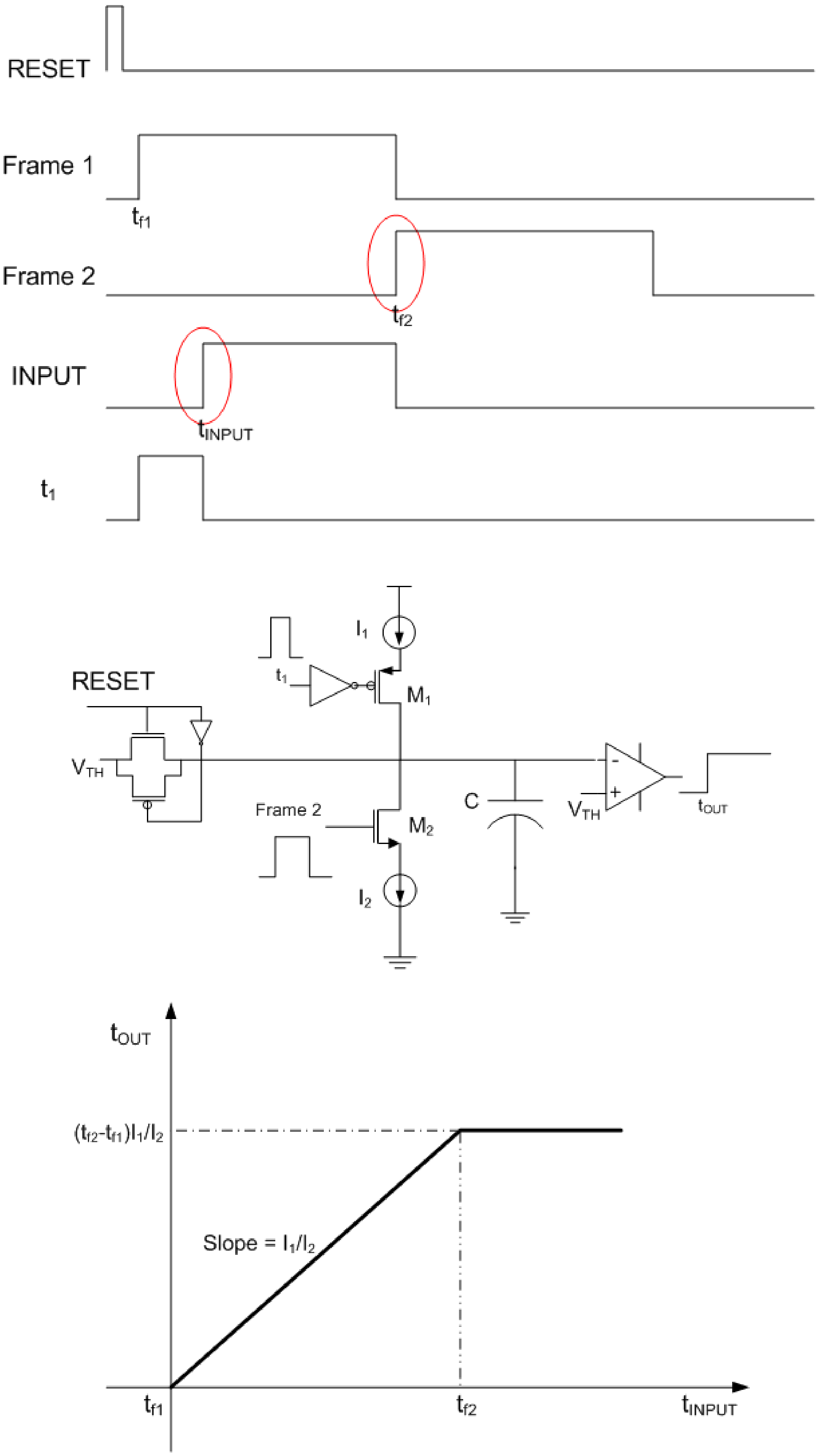

Figure 6–9. Time-mode piece-wise linear activation circuit

The designed perceptron was trained using back propagation algorithm to implement the multiplication function $\frac{t_1 t_2}{\tau}$ where t_1 and t_2 are the inputs to the perceptron and τ is a constant used to scale the output. We chose $\tau = 100\mu s$ for our simulations. The back propagation training algorithm was run over a wide range of inputs and we arrived at the following values for the weights $\frac{I_1}{I_3}$ and $\frac{I_2}{I_3}$ of the time-mode scalar multiplication and summing circuit : 4.6, 4.9 and 1.93, 1.9 for the two neurons in the hidden layer and 1.8, 4.8 for the neuron in the output layer. We also obtained a slope of 0.15 for the piece-wise linear activation function in Figure 6–9.

The MLP was tested for a wide range of inputs (ranging between $1ns$ and $200\mu s$). The test results were very promising. We obtained a low MSE as shown in Figure 6–10. The desired outputs and the actual outputs of the time-mode MLP are shown in Figure 6–11. Figure 6–12 shows cadence simulation results for a particular combination of t_1 and t_2. We can infer from the figure that the simulation results differ by approximately 3% from the desired results. This difference can be attributed to mismatch between current sources and the delay of the comparators used in various neuron circuits.

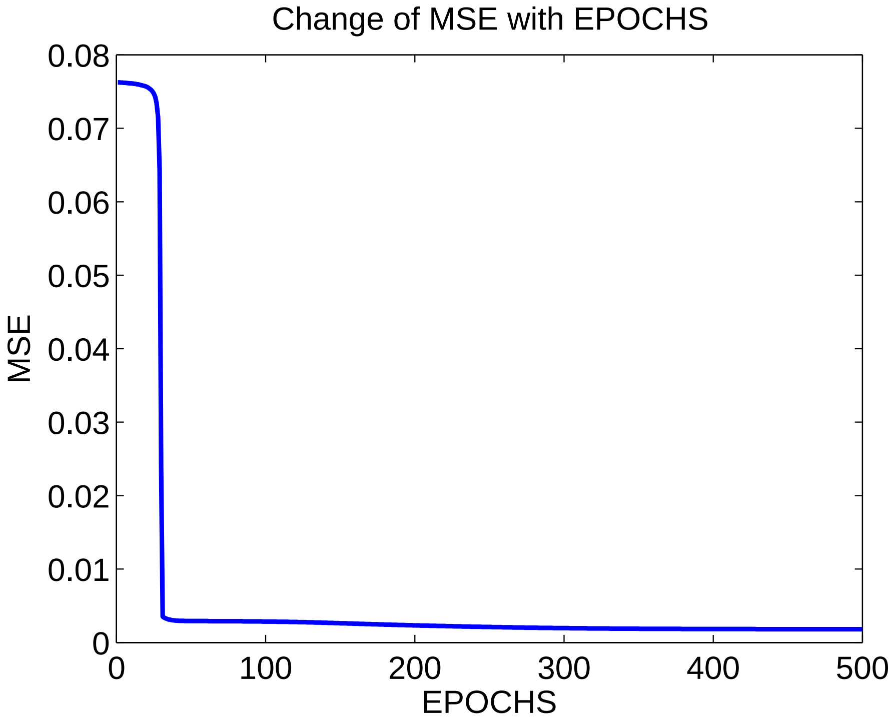

Figure 6–10. Variation of output mean square error with epochs

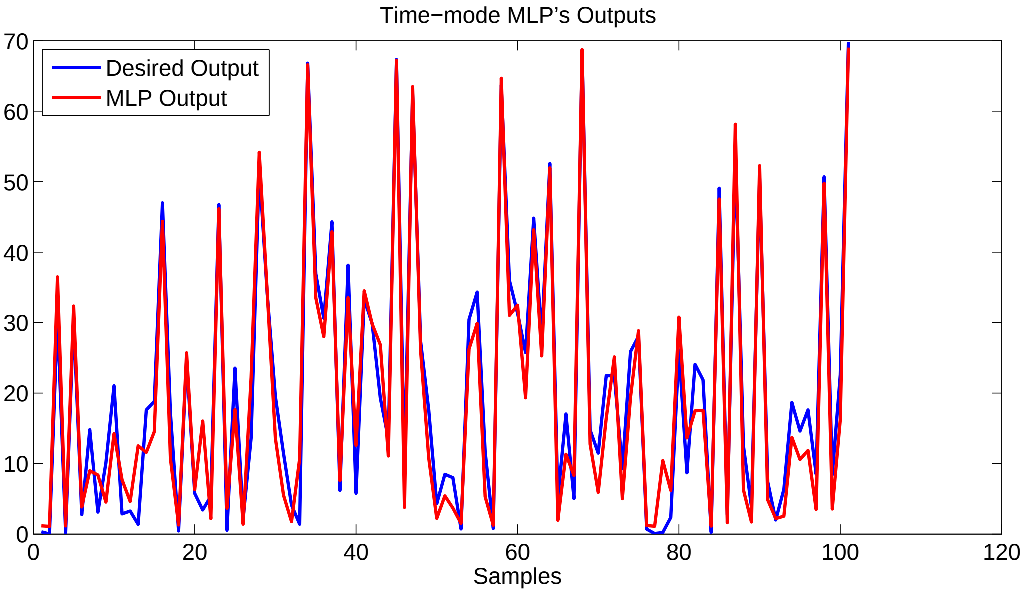

Figure 6–11. Time-mode MLP desired and actual outputs

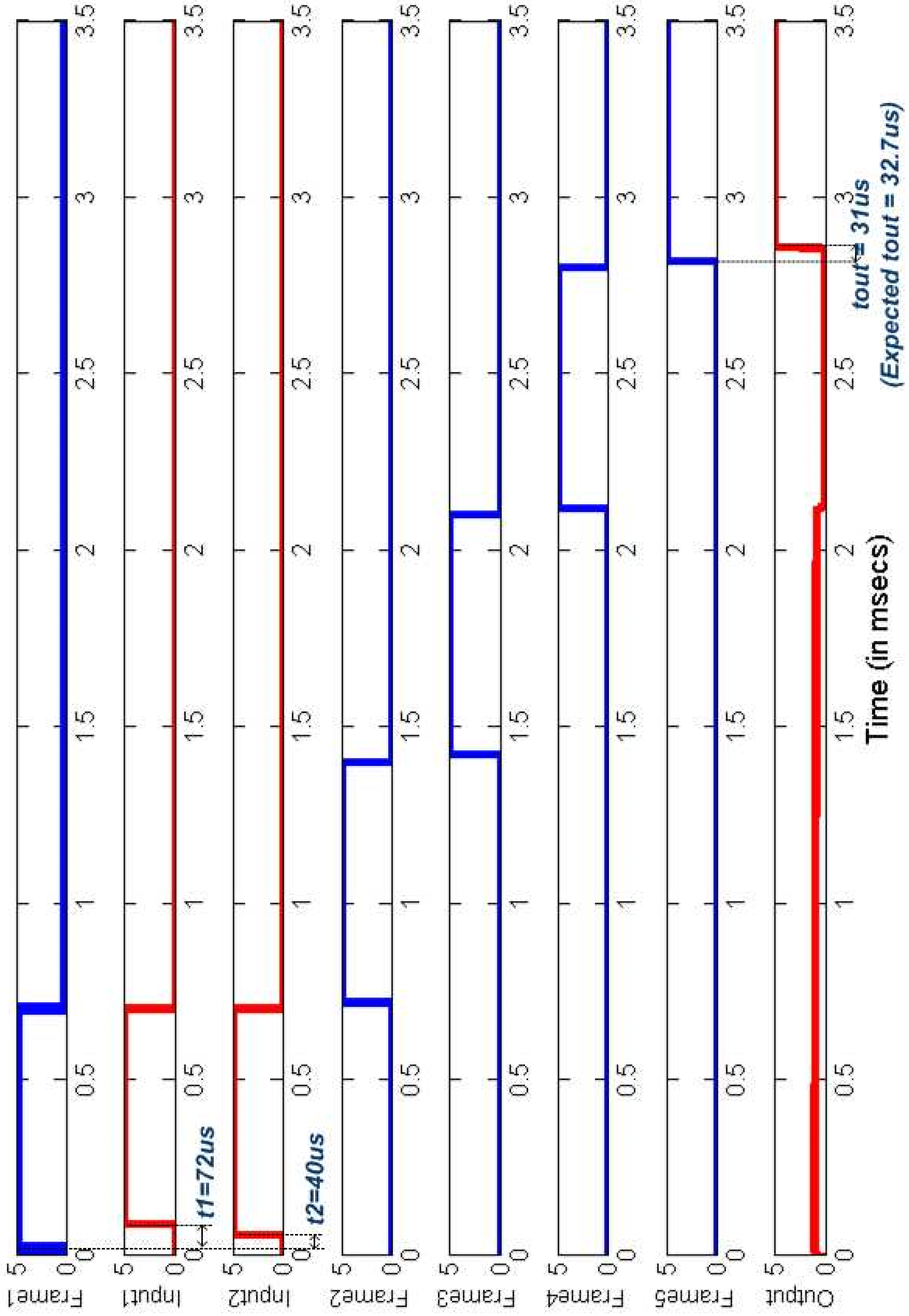

Figure 6–12. Cadence simulation results

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion

Time-mode circuits are described for computing linear and non-linear arithmetic functions - weighted average, weighted difference, weighted sum, scalar product, maximum, minimum, thresholded difference, multiplication and division operations. These basic time-mode computational blocks were used to design a time-based edge detector that naturally interfaces to our time-to-first-spike imager and a time-mode 3-tap FIR filter. Using time-mode circuits improves the circuit complexity, power consumption, chip area, SNR and DR specifications of these applications.

As technology scales, digital transistors become faster and faster while voltage-mode and current-mode analog designs become more complex. However, the performance of the time-mode circuits actually improve with scaling technologies. We have shown that time-mode circuits are expected to perform well in new silicon technologies and emerging carbon nanotube technologies.

There are some similarities between these time-mode circuits and single/dual-slope ADCs converters. These converters also charge and/or discharge a capacitor and output a step with the output voltage reaches a threshold. However, these converters have not been used for computation. As was already pointed out in Chapter 2, a large number of researchers have studied pulse-based computation but these circuits have been limited in their computational power.

There is also a striking similarity between the step function computations described here and simple models of spiking neurons. In each case, digital events from other neurons are weighted and summed, increasing the neuron cell potential.

When the cell potential reaches a fixed threshold, a digital event occurs at the output. Local processing is analog but global communication is asynchronous digital. The major difference between the architectures is that steps have been used here while neurons (and their models) use pulses. This difference may be smaller than it may initially appear since, as Maass pointed out, if the synaptic response function is approximately linear on the outset, then neurons could be implementing a weighted average computation [25].

From an engineering perspective, a step input can provide a guarantee that the neuron will eventually fire due to a single input (at least for the all-positive-weight case). Pulses are not as straightforward to process and many times engineering pulse computation boils down to just determining whether sets of pulse arrive simultaneously or not. However, pulse based computation doesn't require an external reset as does the step based computation described here.

7.2 Future work

We have only just begun our study of time-mode circuits; there is much more work to be done. In particular, expanding the range of functionality, exploring their use in real applications and further analysis of their limitations and circuit optimizations are necessary.

Currently the FIR filter design works only for positive weights. In the future, this FIR filter design will be slightly modified to work for both positive and negative weights; that is, ensure the FIR filter works in all four quadrants. This way, all possible filters can be implemented using time-mode circuits. Also, the power consumed by the FIR filter has to be optimized and compared to the power consumed by existing low-power voltage and current mode alternatives.

The time-mode MLP is in its very early stage. More work is necessary to optimize its implementation and fully characterize its performance.

Overall, time-mode circuits provide a very appealing alternative to traditional voltage-mode and current-mode computational circuits. Since time-mode computational circuits have advantages like low complexity, low power consumption, high SNR and DR, we will come up with more computational applications where the use of time-mode circuits would prove very advantageous. Further work in scaling these circuits to deep submicron silicon and nanotechnologies is necessary. If a mechanical threshold element (or neuron) replaces the functionality of the capacitor and the comparator in the time-mode weighted average circuit, the circuit implementation would be greatly simplified and it would open new doors towards integrating NEMS and time-mode computational circuits.

REFERENCES

[1] P. Horowitz and W. Hill, *The Art of Electronics*, Cambridge University Press, New York, 1989, Second Edition.

[2] D. A. Johns and K. Martin, *Analog Integrated Circuit Design*, John Wiley and Sons, Inc., New York, 1997.

[3] C. Toumazou and D. G. Haigh and F. J. Lidgey, *Analogue IC Design: The Current-Mode Approach*, Peter Peregrinus Ltd, London, 1993.

[4] B. Gilbert, "Translinear Circuits: A Proposed Classification," *Electronics Letters*, vol. 11, no. 1, pp. 14–16, 1975.

[5] D. R. Frey, "Log-domain filtering: An approach to current-mode filtering," *Inst. Elec. Eng. Proc.*, vol. 140, no. 6, pp. 406–416, Dec 1993.

[6] A. J. P. Theuwissen, *Solid-State Imaging with Charge-Coupled Devices*, Kluwer Academic Publishers, New York, 1995, First Edition.

[7] H. Schmid, "Why Ćurrent ModeĎoes Not Guarantee Good Performance," *Analog Integrated Circuits and Signal Processing*, vol. 35, pp. 79–90, 2003.

[8] W. Maass, "Fast sigmoidal networks via spiking neurons," *Neural Computation*, 9:279–304, 1997.

[9] M. Oulmane and G. W. Roberts, "A CMOS Time Amplifer for Femto-Second Resolution Timing Measurement," *IEEE International Symposium on Circuits and Systems CD*, May 2004.

[10] K. Papathanasiou and A. Hamilton, "Pulse based Signal Processing: VLSI implementation of a Palmo Filter," *IEEE International Symposium on Circuits and Systems*,vol. 1, pp. 270–273, May 1996.

[11] W. Maass, *Pulsed Neural Networks*, Chapter 3, MIT Press, Cambridge, Massachusetts, 1999.

[12] R. Sarpeshkar and M. O'Halloran, "Scalable Hybrid Computation with Spikes," *Neural Computation*,vol. 14, pp. 2003–2038, Sep 2002.

[13] G. Cauwenberghs and A. Yariv, "Fault-tolerant Dynamic Multi-level Storage in Analog VLSI," *IEEE Transactions on Circuits and Systems. Part II*,vol. 41, pp. 827–829, 1994.

[14] X. Qi and X. Guo and J. G. Harris, "A Time-to-first Spike CMOS Imager," *IEEE International Symposium on Circuits and Systems*, May 2004.

[15] H. S. Narula and J. G. Harris, "A Time-Based VLSI Potentiostat for Ion Current Measurements," *IEEE Sensors Journal*, vol. 6, no. 2, pp. 239–247, Apr 2006.

[16] F. N. H. Robinson, *Noise Fluctuations in Electronic Devices and Circuits*, Clarendon Press, Oxford, 1974.

[17] W. Zhao and Y. Cao, "New generation of Predictive Technology Model for sub-45nm design exploration," *7th IEEE Symposium on Quality Electronic Design*, pp. 585–590, 2006.

[18] A. Raychowdhury and S. Mukhopadhyay and K. Roy, "A Circuit Compatible Model of Ballistic Carbon Nanotube Field Effect Transistors," *IEEE Transactions on Computer Aided Design*, vol. 23, no. 10, pp. 1411–1420, Oct 2004.

[19] P. J. Burke, "AC Performance of Nanoelectronics: Towards a Ballistic THz Nanotube Transistor," *Solid State Electronics*, 48(10), pp. 1981–1986, 2004.

[20] M. Butts and A. DeHon, and S. C. Goldstein, "Molecular electronics: devices, systems and tools for gigagate, gigabit chips ," *Proc. IEEE/ACM Intl. Conference on Computer Aided Design*, pp. 433–440, Nov 2002.

[21] J. Von Neumann, *Probabilistic Logics*, Automata Studies, Princeton University Press, Princeton, 1956.

[22] D. E. Dickinson and R. M. Walker, "Reliability Improvement by the Use of Multiple-Element Switching Circuits," *IBM J. Res. Develop.*, vol. 2, no. 2, April 1958.

[23] R. C. Gonzalez and P. Wintz *Digital Image Processing*, Addison Wesley, Reading, 1987, Second Edition.

[24] X. Guo, "A Time-Based Asynchronous Readout CMOS Image Sensor," *Ph.D. Thesis at the University of Florida*, 2002.

[25] W. Maass, *Pulsed Neural Networks*, Chapter 2, Computing with spiking neuwons, MIT Press, Cambridge, Massachusetts, 1999.

[26] C. T. Jin and P. L. Rolandi and P. H. W. Leong, "Non-volatile programmable pulse computation cell," *Electronic Letters*, vol. 35, pp. 256–267, Issue: 17,19, Aug 1999.

[27] D. Marr, *Vision*, W. H. Freeman and Company, San Francisco, 1982.

[28] C. Mead, *Analog VLSI and Neural Systems*, Addison-Wesley, Reading, 1989.

[29] S. Norsworthy and I. Post and S. Fetterman, “A 14-bit 80-KHz Sigma-Delta A/D converter: Modelling, Design, and Performance Evaluation,” *IEEE J. of Solid State Circuits*, vol. 24, pp. 256–267, 1989.

[30] S. Haykin, *Neural Networks: A Comprehensive Foundation*,Pearson Education, Upper Saddle River, 2nd edition, July 1998.

[31] G. Cairns and L. Tarassenko, “Learning with analogue VLSP MLPs,” *Microelectronics for Neural Networks and Fuzzy Systems*, pp. 67–76, Sep 1994.

[32] J. N. Tombs and L. Tarassenko and A. F. Murray, “Novel analogue VLSI design for multilayer networks,” *IEE Proceedings of Radar and Signal Processing F*, vol.139, Issue 6, pp. 426–430, Dec 1992.

[33] X. Wang and R. R. Spencer, “A low-power 170-MHz discrete-time analog FIR filter,” *IEEE Journal of Solid-State Circuits*, vol. 33, no. 3, pp. 417–426, Mar 1998.

BIOGRAPHICAL SKETCH

Vishnu Ravinuthula finished his B.E. degree in electrical and electronics engineering from the College of Engineering, Anna University, Chennai, India, and his M.S. degree in Electrical Engineering from the University of Florida, Gainesville, U.S. in 2000 and 2003, respectively. He is currently doing doctoral research work on Bio-inspired analog circuits and nano-delay circuits at the Computational NeuroEngineering Laboratory of University of Florida under the guidance of Dr. John G. Harris. He did a summer internship at NASA Ames Research Center, California where he was working with Dr. M. P. Anantram, Dr. T. R. Govindan and Dr. Harry Partridge doing research on the pros and cons of using carbon nanotube based transistors for analog and digital applications. He will soon be working in Texas Instruments, Dallas as an analog design engineer developing the analog circuits for high-speed SERDES applications.