Design of a 200MS/s, 8-bit Time based Analog to Digital Converter in 65nm CMOS Technology

Ahmed Abdelaziz Mohamed Mohamed
Mohamed Abdelkader Mohamed Mahmoud
Ahmed Ali Hassan Ali

Supervised by
Dr. Hassan Mostafa
Dr. Mohamed Refky Amin

A thesis submitted to Cairo University
for the degree of
Bachelor
in
Electronics and Electrical Communications Engineering
Cairo University

July 2014

To our Parents and Families

Abstract

Analog-to-Digital Converters (ADCs) is a very important block in mixed analog digital/systems. The whole world is trying to reach best technology scaling targeting better speed, cost and power. Digital nanometer-scale complementary metal-oxide-semiconductor (CMOS) takes the advantage of technology scaling in terms of gate delay and area so system designers try to increase the percentage of digital part in the system. Technology scaling reduces supply voltage and intrinsic gain so disadvantages are faced by analog designers so two broad trends has been taken into consideration in ADC researches. The first trend emphasizes relaxation of analog domain precision and the recovery accuracy, this is done by making a digitally-assisted analog design. This trend helps to reduce power consumption. The second trend is representing the signals in the time domain. Due to technology scaling digital systems give better resolution by reducing the gate delay. So representing the signal as period of time, rather than as a voltage, can take advantage of technology scaling in terms of reducing power consumption and die area.

This thesis proposes a Time-based ADC that consists of two main blocks. The first is the voltage-to-time converter (VTC) or analog-to-time converter (ATC), which is mainly based on current-starved inverter architecture with some modifications. The VTC receives an analog voltage input and produces a series of pulses. The VTC focuses on pulse position modulation (PPM) which means that the delay of each pulse in the series of pulses produced is proportional to the input. The second is the time-to-digital converter (TDC) which is mainly based on the vernier delay line (VDL) method. Two level of the vernier are used which are coarse and fine levels. The coarse level is responsible of propagating the signals produced from VTC with big steps until catch up occurs then the signals at catch up is sent the fine level to propagate implicitly backward with small steps. Then calculating the steps by which the lag signal exceeded the lead one when catch up occurred at the coarse level, Finally, subtracting the binary output of the fine level from the binary output of the coarse level resulting in the digital output of the time based ADC.

A 8-bit 200MS/s ADC in 65nm CMOS is proposed which achieves an effective number of bits (ENOB) of 7.6 bits.

Acknowledgments

We would like to express our sincere gratitude to our supervisors, Dr. Hassan Mostafa and Dr. Mohamed Refky, without whom, this project would have not been possible, and for their continuous support and guidance throughout the project. we are also grateful for the Eng. Ali El-Hussein for his support with many tools that were helpful throughout the project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The importance of analog-to-digital converters (ADCs) comes from introducing the powerful digital processors in the mid-twentieth century. After a long life in analog world, there is a need to decrease the gap between continuous analog information and discrete digital information. The digital systems could process inputs from the real world [1].

But, that wasn't enough, the scaling trends of very large scale integration (VLSI) CMOS processes continued to bring us every year to higher speed and lower power digital circuitry. The international technology road map for semiconductors (2003 edition) has predicted that this scaling trend will continue until well into the next decade. For example, the transistor minimum gate length and the power suppl voltage are predicted to reach 7nm and 0.5V respectively by 2018 [2].

1.1 Motivation

As technology scales, the improvement in the analog part is small compared to that of the digital part. The reason is that the supply voltage reduction that accompanies technology scaling results in lower voltage swing.

Small voltage swing causes two problems. The first one is the low signal to noise ratio (SNR) because the noise level does not decrease with the same ratio as the supply voltage decrease with. The second problem arises from the fact that the threshold voltage of the transistor too does not decrease with the same ratio as the supply voltage, this results in making transistor cascoding difficult. Thus, the design of the operational amplifier (op-amp), which is a main building block in the ADC, becomes difficult, and a new solution must be found [3].

New implementations of ADCs based on time quantization have been arose to help reducing of the drawbacks such as the reduced voltage swing, and to take advantage of the high speed circuitry brought about by the VLSI scaling. One common feature in these implementations is that the signal is represented in the time domain during data

conversion. These implementations are called time-based ADCs since their resolution is determined in the time domain, which is a main difference from conventional ADCs [2].

1.2 Proposed work

In this thesis, A high resolution TDC of 3.9ps is reached, and a reduced number of elements is achieved instead of using 256 D-FLIP FLOPs and the double of last number for delay elements. This is done by using the coarse and fine concept with two level vernier delay line (VDL), The number of elements has been reached to use only 32 D-FLIP FLOPs and 64 delay elements. In the VTC, A large dynamic range of 291mV with high linearity and high sensitivity of 3.43ps/mV are reached also.

1.3 Thesis Organization

This thesis consists of eight chapters, including this introduction. Chapter 2 introduces a literature review of the analog to digital converters, their specifications, and some types of direct conversion ADC. Flash ADC, successive approximation ADC, pipelined ADC, and delta-sigma ADC are reviewed.

In Chapter 3, an introduction to the time based ADC is given with a review of some TADC types. Dual slope TADC, and oversampling TADC based on phase modulation or frequency modulation are reviewed. Also, an analysis of basic VTC circuit is reviewed.

In Chapter 4, the analysis of the first main block of the time-based ADC which is VTC is presented, including an explanation of the circuit on which the design proposed is based, the challenges that have been faced in this design, improvements to make best use of the old design and then the derivation of output delays using timing Diagram.

In Chapter 5, an overview of many TDCs ideas, there advantages and disadvantages, and what types can help is presented. In this chapter four types of TDC is reviewed: Single counter TDC, Cyclic Pulse-Shrinking TDC, Simple flash TDC, and Vernier TDC.

In Chapter 6, The implementation of the proposed two levels VDL TDC is introduced. The chapter presents the implementations of the basic elements in the circuit: the delay element, D-FLIP FLOP, and the interface circuit. Then shows the overall implementation which consists of coarse level, interface circuit, fine level, and the read out circuit. Also, some of the challenges and problems that have been faced during work is shown.

Chapter 7 shows how the performance of the overall TADC can be measured, gives a special concern to the effective number of bits (ENOB) ,and then presents the simulation results of the TADC proposed in this thesis.

Chapter 2

Literature Review

2.1 Analog to Digital Converter

In this chapter, a definition of ADC, its function and the basic concepts needed in analysis of ADCs is described. Then different types of signal representations will be reviewed. ADC characteristics are also will be discussed, then finally review some different types of ADCs.

2.1.1 Signal Representation

There are four types of signal that may be dealt with in ADC through data conversion, Analog signals, sample-and-hold signals, asynchronous digital signals and digital signals, which are displayed in Figure (2.1) (a), (b), (c) and (d), respectively.

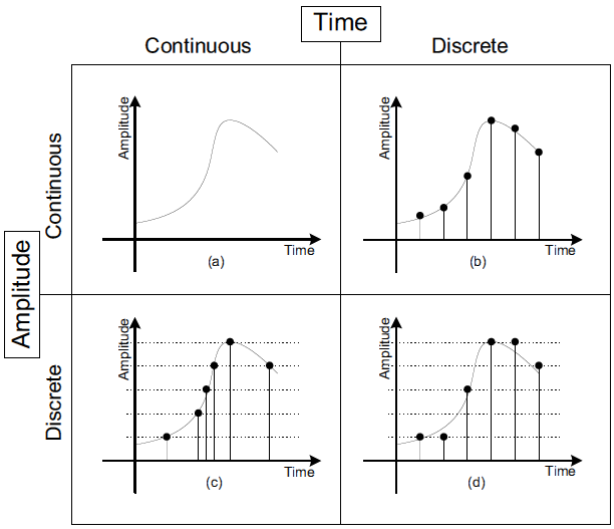


Figure 2.1: Signal representations. (a) Analog Digital, (b) Sample-and-hold signal, (c) Asynchronous digital signal, and (d) Digital signal

ADC is a device that converts analog continuous signal (continuous in time and amplitude) to digital discrete signal (discrete time and amplitude). Each digital code is a quantized version of the sampled analog signal at the time instant corresponding to it. Figure 2.2 shows a 3-bit ADC's output. It's reverse operation is done using a digital to analog converter (DAC) [4].

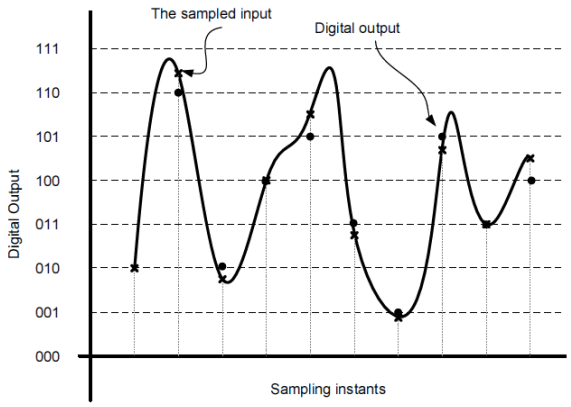


Figure 2.2: Example of 3-bit ADC [3]

2.1.2 Sampling

Sampling is obtain the signal values from a continuous signal every regular interval of time as shown in figure 2.3. The sampling interval is denoted as T_s. The sampling rate or sampling frequency $f_s = 1/T_s$, must be at least twice the input signal bandwidth (BW) f_{BW}. This condition is called Nyquist criterion [5].

$$f_s \geq 2 * f_{BW} \tag{2.1}$$

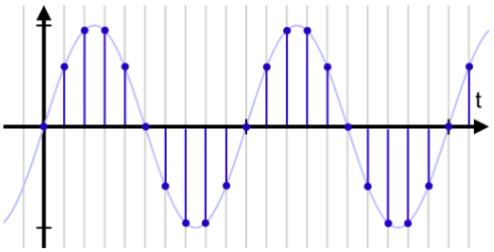


Figure 2.3: Sample of an analog signal

If sampling rate used in the system doesn't meet the Nyquist criterion, aliasing occurs the original signal couldn't be reconstructed from its samples and this happens when T_s is very large which means loss of information. Figure 2.4 shows an example of large T_s ($f_s < 2 * f_{BW}$) and how aliasing occurs in the reconstructed signal.

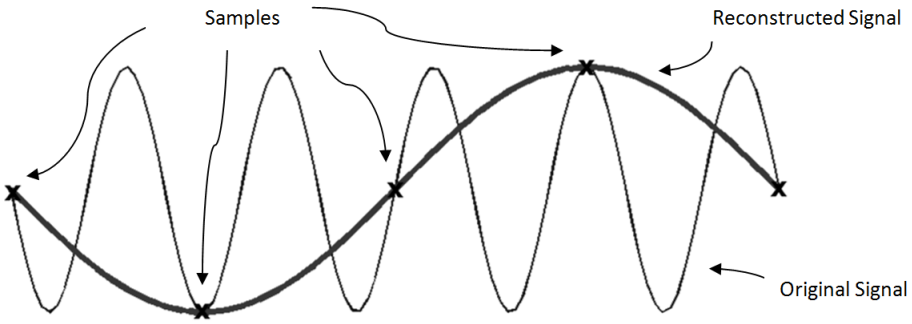


Figure 2.4: Example of aliasing.

Frequency spectrum of the signal makes understanding aliasing easy. Assume having a signal in time domain and its corresponding frequency spectrum is as shown in Figure 2.5. If fs meets the criterion of Nyquist, the frequency spectrum of the sampled signal will be as shown in Figure 2.6.a. While if it doesn't meet Nyquist criterion, aliasing occurs as shown in Figure 2.6.b. Reconstructing the original signal from samples in Figure 2.6.b can't be done due to distorted parts (dotted parts in figure 2.6.b).

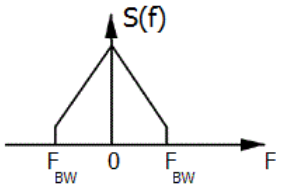


Figure 2.5: Frequency Spectrum of a signal

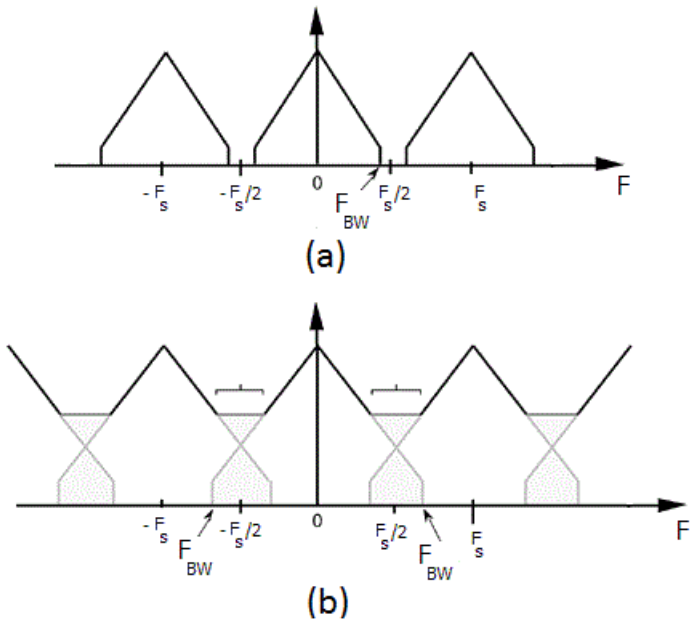


Figure 2.6: The frequency spectrum of samples in case of (a) $f_s \geq 2 * f_{BW}$ or (b) $f_s < 2 * f_{BW}$

To avoid aliasing, an aliasing filter is usually used to band limit the input signal before applying the ADC.

2.1.3 Quantization and Quantization error

It is an undesired phenomenon which happens when a system produces the same output for a certain range of input. To understand this phenomenon, it is required to illustrate the resolution first. The resolution of ADC is defined as the smallest change in input that makes a change in output digital code. The resolution is expressed in terms of the least significant bit (LSB) as

$$LSB = \frac{Full\,Scale\,input\,of\,ADC}{2^{number\,of\,bits\,contained\,in\,output\,digital\,code}} \tag{2.2}$$

The problem arises when the analog value being sampled falls between two digital "steps" like second sample in Figure 2.2 as it falls between the two levels (110 and 111). When this happens, the analog value must be represented by the nearest digital value, resulting in a very slight error so the second sample is represented by 110 as it is the nearest digital value to it. This error is called quantization error. Thus having a small resolution of ADC as possible will result in reducing the quantization error. In other words small resolution means using greater number of bits. Figure 2.7.a shows a signal and its quantization while Figure 2.7.b shows the quantization noise. Quantization noise is considered the only source of error in ideal ADC. For Ideal ADC, quantization error is always in the range of

$$-0.5 * LSB < Quatization\,error < 0.5 * LSB \tag{2.3}$$

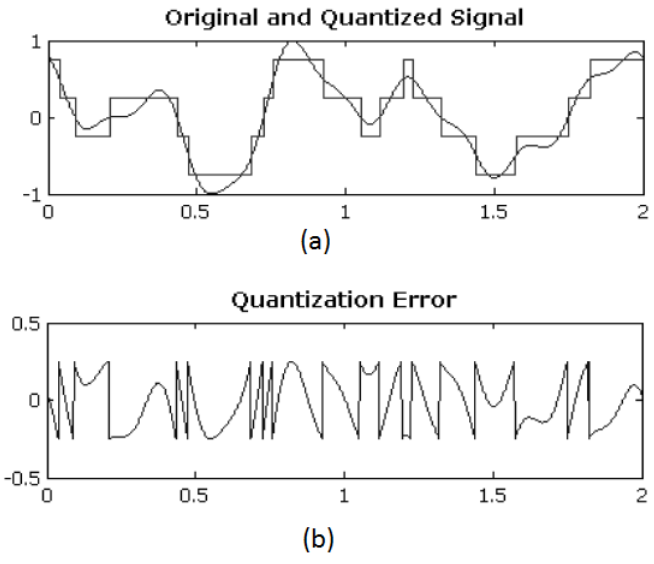


Figure 2.7: (a) Original and Quantized signals , (b) Quantization noise

2.2 ADC specifications

Deciding which type of ADCs is suitable for a certain application, depends on understanding the ADC specifications.

2.2.1 Static specifications

New types of error, other than the quantization error, are introduced due to non idealities in circuit implementation of the ADC. In this section, some of errors that are independent on time (static) is reviewed.

2.2.1.1 Offset and Gain errors

The offset error is the deviation of the ADC characteristics line (the line connecting the mid-point of ADC levels) to the right or to the left, from the one of the ideal ADC characteristics line [6]. The offset error is generated from the operational amplifier's offset voltage. The offset error affects all the digital code with the same effect and is removed using calibration. While the gain error is the deviation of the slope of the ADC characteristics line from that of the ideal ADC after removing the offset error [6]. Figure 2.8 shows the offset error in line of actual ADC which is deviated for the shifted actual by the value of offset error. So it is first removed by calibration to be at shifted actual. The gain error is shown in Figure 2.8 and measured from the line after removing offset error.

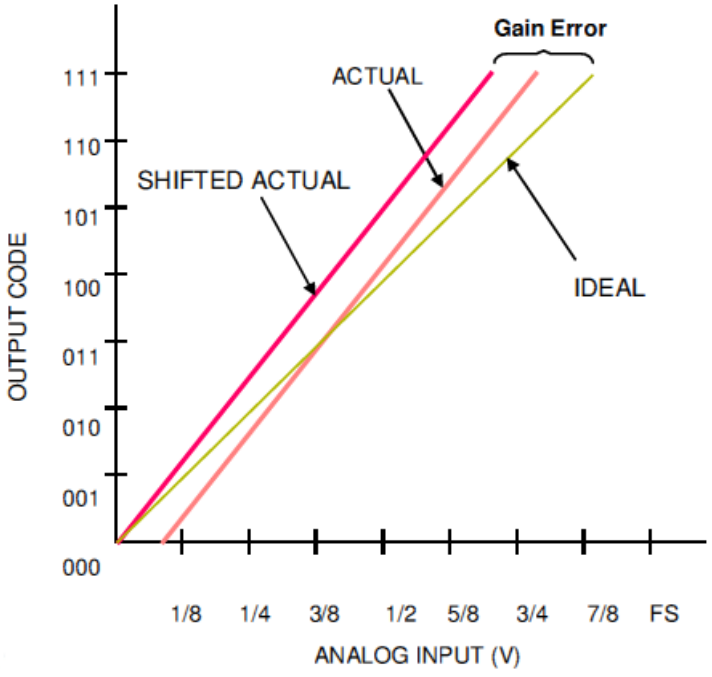


Figure 2.8: Example showing the Gain and offset errors.

2.2.1.2 The absolute accuracy and relative accuracy

The absolute inaccuracy is defined to be the maximum deviation of the actual analog value from the ideal one. It includes offset, gain and linearity errors and may be variable for every individual quantization value. While the relative inaccuracy is defined to be the deviation that still remains even after the offset and gain errors have been removed.

2.2.1.3 Differential non Linearity (DNL)

DNL is defined to be the deviation of the every individual code's width from the ideal 1 LSB [4]. Figure 2.9 shows an example of DNL in ADC. DNL for each individual code is expressed as

$$DNL_k = \frac{width\,of\,every\,individual\,code\,k - 1\,LSB}{1\,LSB}$$

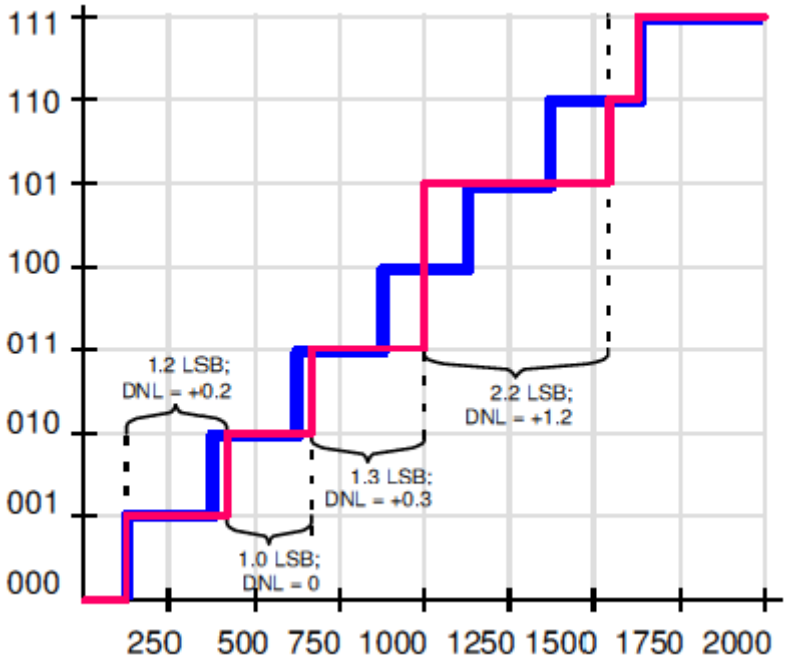


Figure 2.9: Example of DNL error in ADC.

2.2.1.4 Integral non linearity (INL)

INL is defined to be the deviation of the actual transfer function from a straight line [5]. Figure 2.10 shows an example of INL in ADC. INL is expressed as

$$INL(k) = \sum_{i=0}^{k} DNL$$

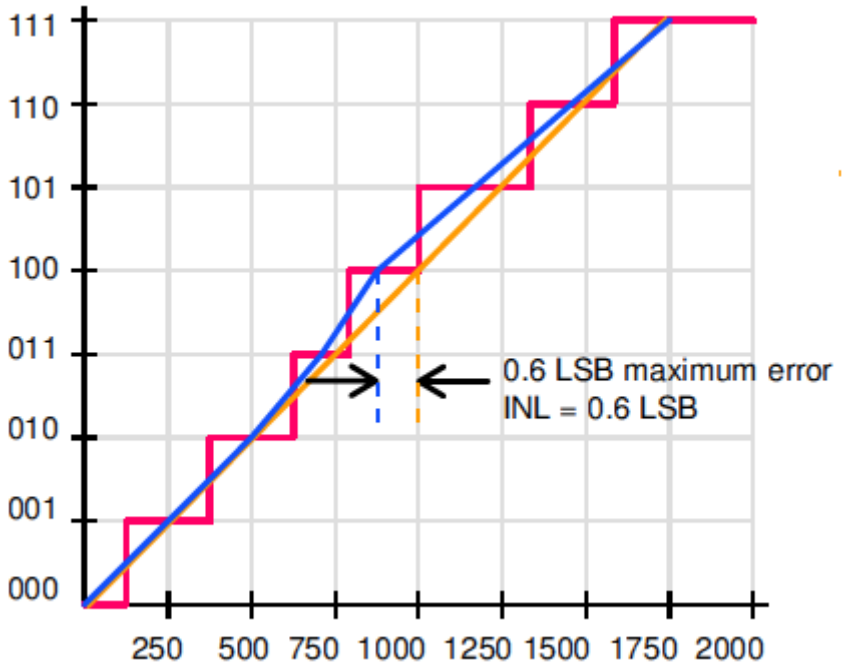


Figure 2.10: Example of INL error in ADC

2.2.1.5 Missing Codes

Missing codes is said to be found in ADC when one of the ADC's digital output codes is skipped. No missing codes is said to be guaranteed if DNL error doesn't exceed 1 LSB or if INL error doesn't exceed 0.5 LSB. Figure 2.11 shows an example of a missing code in ADC.

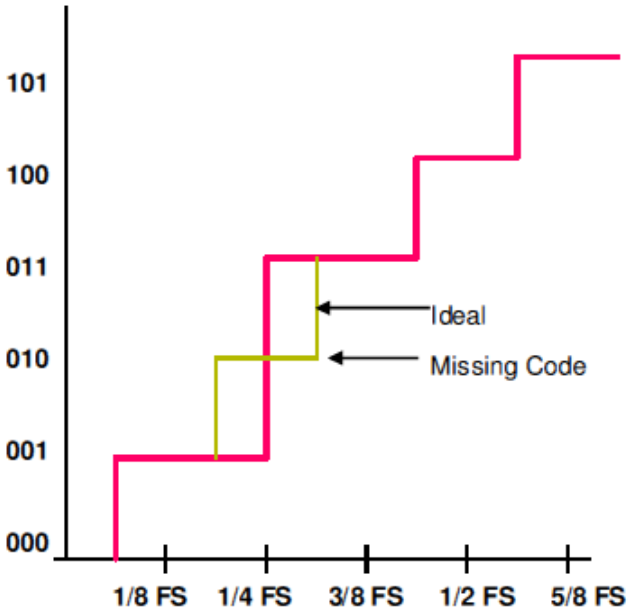


Figure 2.11: Example of a missing code in ADC

2.2.2 Dynamic specifications

In this section, Some of ADC specifications which are dependent on time is reviewed.

2.2.2.1 ADC sampling rate and conversion time

The ADC Sampling rate is defined as the speed at which the ADC can convert continuously analog input samples into digital code and is equal to the inverse of conversion time. The conversion time of ADC is defined as the time that the converter takes to complete a single conversion including the acquisition time of the analog input signal.

2.2.2.2 Signal to Noise Ratio (SNR)

SNR is defined as the ratio of the full scale input signal power to the noise power at the output of ADC. SNR is given by

$$SNR = 10log(\frac{Full\,scale\,input\,power}{quantization\,noise\,power + circuit\,noise\,power})\,dB$$

After ignoring circuit noise power (quantization noise only), SNR is expressed as

$$SNR = 6.02D + 1.76\,dB$$

as seen from last equation, as the number of bits increases, the system SNR increases.

2.2.2.3 Signal to Noise and Distortion Ratio (SNDR)

SNDR is defined to be the ratio of the full scale input signal power to the noise plus distortion power. Figure 2.12 shows a signal with harmonic distortion. SNDR is given by

$$SNDR = 10log(\frac{Full\,scale\,input\,power}{noise\,power + distortion\,power})\,dB$$

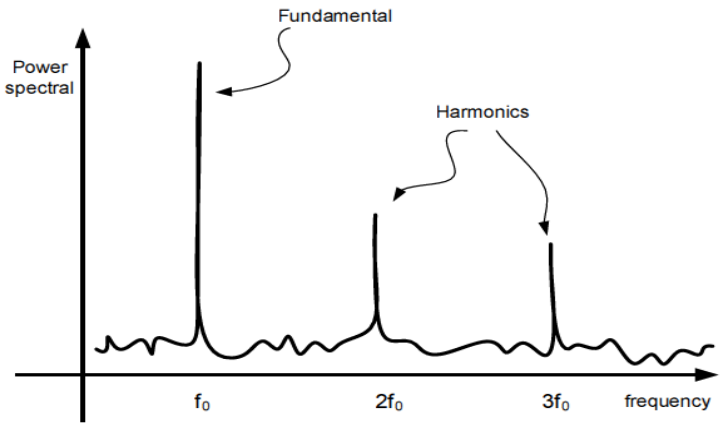


Figure 2.12: Signal with harmonic distortion.

2.2.2.4 Effective Number of Bits (ENOB)

SNDR (which includes noise and distortion powers) is less than ideal ADC SNR (which includes only the quantization noise), The actual number of bits will be less than that gotten from

$$SNR = 6.02D + 1.76\,dB$$

Effective number of bits (ENOB) is the number of bits that if it is substituted in the previous equation the value of SNDR will equal to the SNR value. Thus we can express ENOB as

$$ENOB = \frac{SNDR|_{dB} - 1.76}{6.02}$$

2.3 Conventional ADC types

ADCs are classified according to two ways. The first way is in which sampling is performed and have two types. The first type is the Nyquist rate ADCs like flash ADC, successive approximation ADC, and pipelined ADC. The second type is the oversampling ADCs like sigma-delta modulator. The oversampling conversion technique avoids many of difficulties found with Nyquist ADCs like using of anti-aliasing analog filters.

The second way is in which conversion is performed and divided into two types. The first type is the direct conversion of analog input into digital output. The ADCs of the first type are called Conventional ADCs. The second type is the indirect conversion and this is done by first converting the analog input signal into an intermediate representation such as time. Then converting this intermediate representation into digital code. This thesis propose an ADC of the second type. The ADCs of the second type are called time based ADCs (TADC).

Conventional or direct conversion ADCs are divided into Nyquist rate ADCs in which sampling frequency is equal to twice the maximum frequency of input signal bandwidth [4] and oversampling ADCs in which sampling frequency is so big compared to input signal frequency [4]. For applications that require high input signal frequency, Nyquist ADCs are better than oversampling ADCs. Oversampling is better for low input signal frequency applications but also require high resolution.

Flash ADC, successive approximation ADC and piplined ADC are Nyquist rate Conventional ADCs. Sigma delta (or delta sigma) modulator is an oversampling conventional ADC. Dual slope ADC is a Nyquist rate time based ADC while Voltage controlled oscillator (VCO) is an oversampling time based ADC. In this chapter some Conventional ADC types are reviewed. In chapter 3, some time based ADC types are reviewed.

2.3.1 Comparator

Comparator is the simplest ADC. It is considered to be a 1-bit ADC as shown in Figure 2.13. If Vin is greater than a certain value (V_{TH}) the output is 1 , otherwise the output is 0. There is no ADC without using at least one comparator its architecture of some sort.

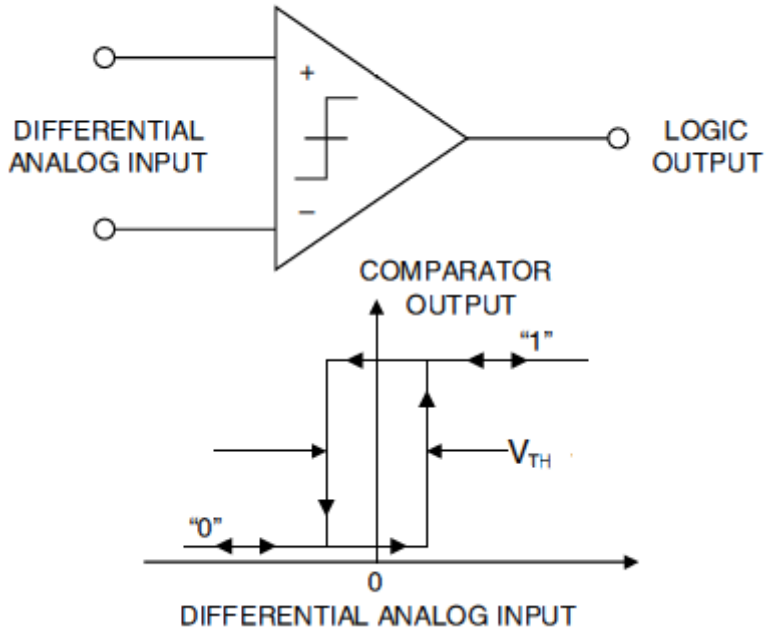


Figure 2.13: The Comparator : A 1 bit ADC

2.3.2 Nyquist rate ADCs

In this section, some types of conventional Nyquist rate ADCs are reviewed.

2.3.2.1 Flash ADC

The fastest type of conventional ADCs is the Flash ADC [6]. Sample of analog input voltage is compared with $2^N - 1$reference values using $2^N - 1$ comparators where N is the number of bits. As shown in Figure 2.14, using a resistive divider with2^N resistors generate the reference voltages. Each reference voltage exceeds the one immediately below it by one LSB. One input of comparators is connected with the sample of analog input voltage while the other is connected to the reference voltages. The output of each comparator is either ”1” when the sample of the analog input voltage is higher than the reference voltage connected to it. Otherwise, the output is ”0”. The output of comparators produce the thermometer code. Then the thermometer code is decoded to the digital code output.

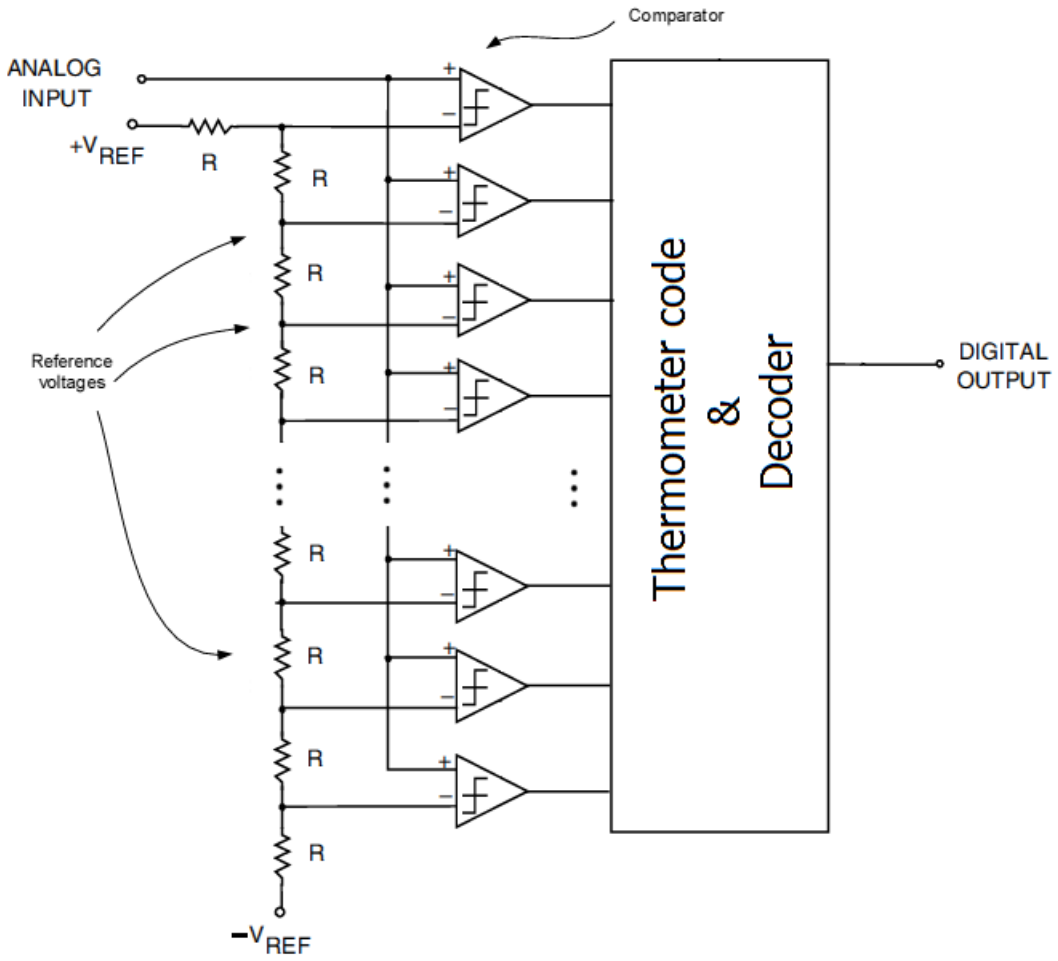


Figure 2.14: The flash ADC.

The main disadvantage of the flash ADC is that it requires a very large number of comparators compared to other types of ADCs. Thus, leading to consuming large area and large power. This disadvantage makes this type of ADC typically impractical for high resolution (greater than 8 bit). The large number of comparators connected to analog input results in a large parasitic capacitance at the input terminal, thus limiting the speed of converter and requiring a power-hungry buffer at the input terminal [4].

2.3.2.2 Successive approximation ADC

The majority of ADC market is taken to Successive approximation ADC for medium and high resolution [4].

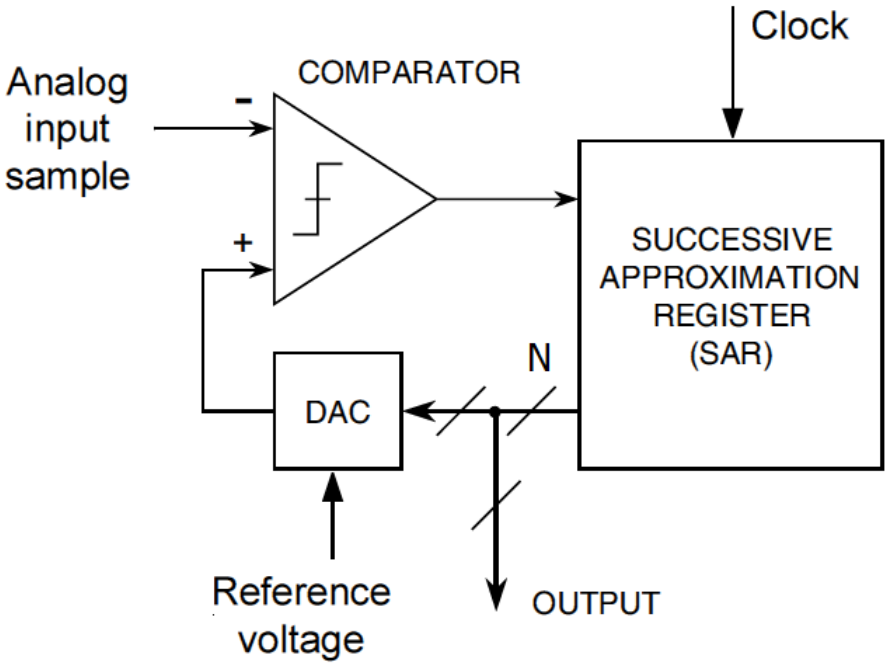


Figure 2.15: The SAR ADC.

Figure 2.15 displays the SAR ADC's basic implementation. The main blocks are a comparator, a DAC, and a successive approximation register (SAR). First, The SAR's bits are set to be at midscale. In other words, all the bits of the successive approximation register (SAR) except the most significant bit (MSB) are reset to "0" while the MSB is set to "1". The SAR output is driven to the DAC with a certain reference voltage. If the DAC output exceeds the analog input sample, The MSB bit in the SAR is set to "0", otherwise it is left "1". The next most significant bit is then set to "1". Then, compare the DAC output with the analog input sample. If the DAC output is the greater, this bit in the SAR is set to "0", otherwise it is left "1". The process is repeated till least significant bit. When all the bits are tested and set to "0" or "1" according to comparison explained, the contents of the SAR is equivalent to the value of the analog input sample, and the conversion is therefore completed. In Figure 2.16 the flow chart of conversion in SAR ADC is displayed.

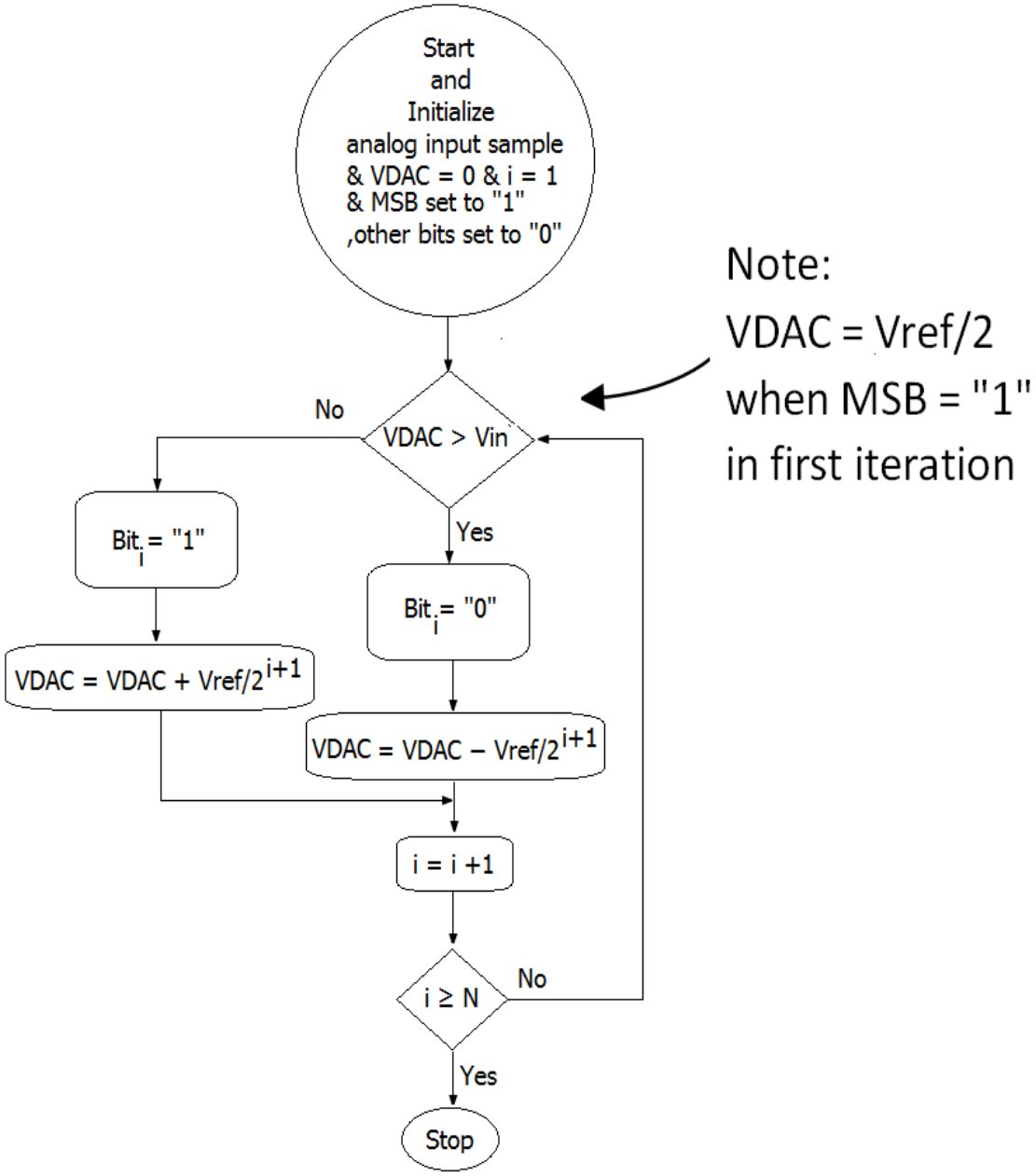


Figure 2.16: SAR algorithm's flow chat.

Assuming a Vin sample = 45, the reference voltage = 64 and 5-bit SAR ADC. Figure 2.17 shows an example of conversion algorithm of SAR ADC [7].

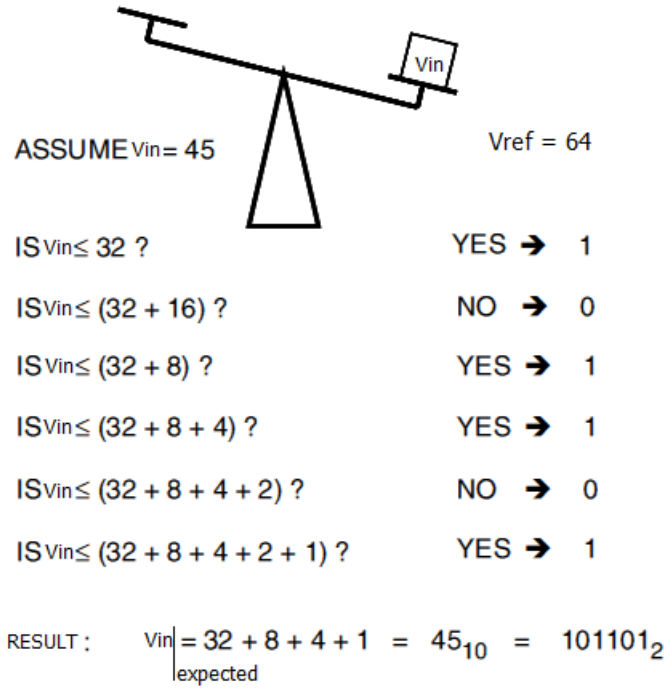


Figure 2.17: 5-bit SAR ADC example.

2.3.2.3 Pipelined ADC

The pipelined ADC takes the advantage of flash ADC in its throughput and that of SAR ADC in its resolution. It utilizes both advantages in one integration with lower area and power. Figure 2.18 shows the pipelined ADC architecture. It consists of N stages, each stage excluding the last stage, has a sample and hold (S/H), M-bits flash ADC of low number of comparators resulting in lower area and power, M-bits DAC, summer, and a gain block with gain equal to 2^N. While, the last stage is only M bits flash ADC.

Pipelined ADCs are based on the concept of sub-ranging as shown in Figure 2.19. Each stage samples its input (the output of the previous stage) except the first stage takes the input analog signal. Then, the flash ADC in every stage converts the input sample to M-bits. Then, the output of flash ADC is taken to the input of flash DAC. The output of flash DAC is then removed from the input sample to the stage to form residue. The residue is then gained up by 2^N and fed to the next stage. As shown in Figure 2.19, as a result of using pipelining, only 16 comparators is used to reach 6-bit resolution instead of 64 comparators.

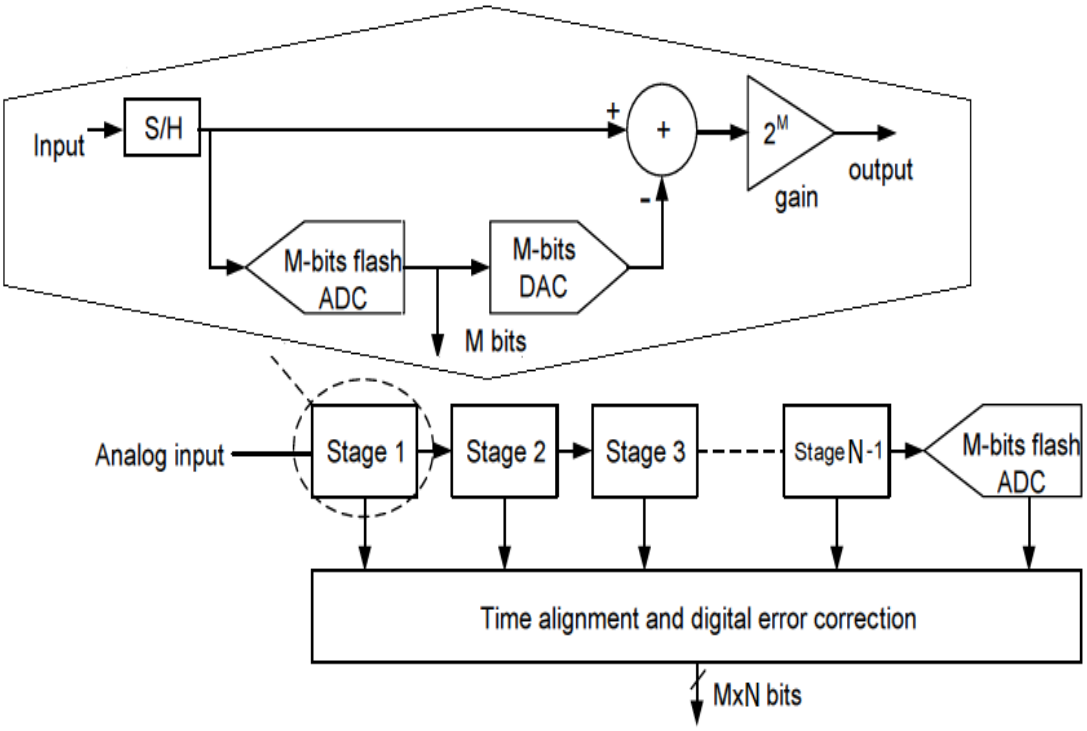


Figure 2.18: Pipelined ADC.

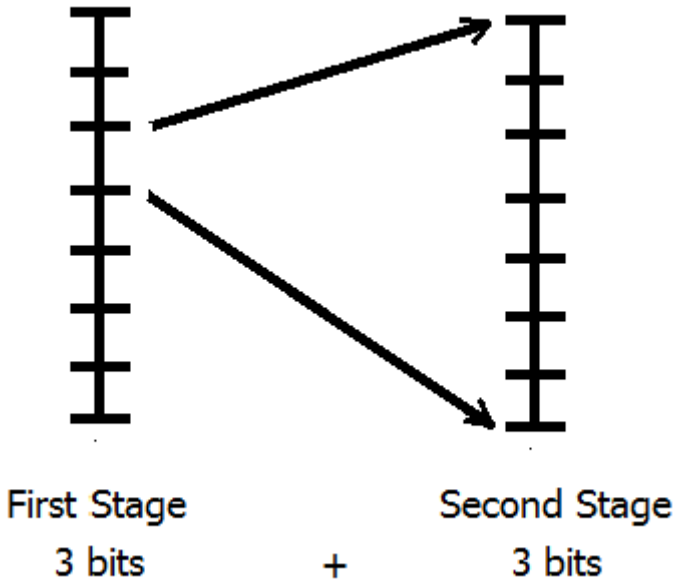


Figure 2.19: 6-bit pipelined ADC

2.3.3 Oversampling ADCs

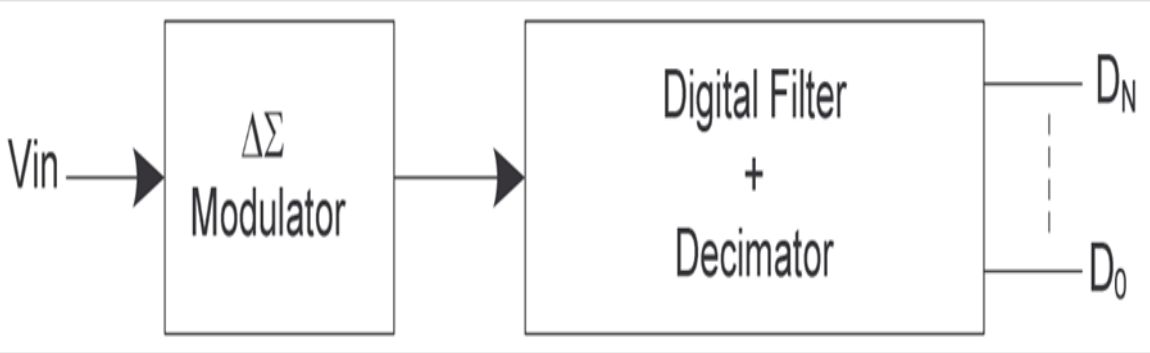
2.3.3.1 Delta-Sigma ($\Delta - \Sigma$) ADC

Delta Sigma ($\Delta\Sigma$) ADC's are typically used in low frequency high resolution applications.The advantages of$\Delta\Sigma$ ADC can be summarized in the following:

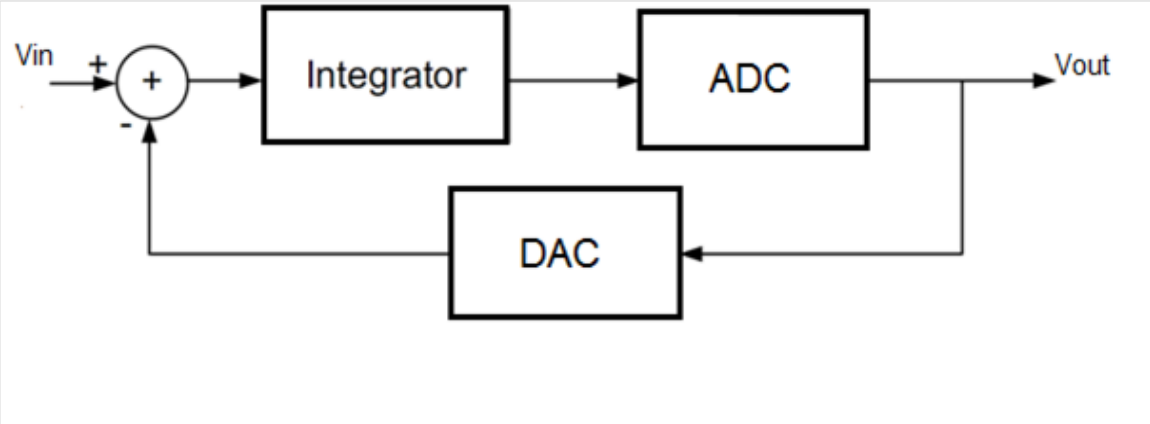
1. There are no precise requirements on analogue building blocks as the linearity of the ADC is not dependent on component matching. Also it may take advantage of the low cost, and low power digital filtering.

2. It relaxes the transition band requirements for analogue anti-aliasing filters.

3. It reduces the baseband quantization noise power and most importantly trades speed for resolution. The $\Delta\Sigma$ ADC depicted in Figure 2.20.a consists of a $\Delta\Sigma$ modulator as shown in Figure 2.20.b and a decimation filter [1] [8].



(a)



(b)

Figure 2.20: (a) Block diagram of $\Delta\Sigma$ ADC, (b) Block diagram of $\Delta\Sigma$ modulator)

The purpose of the $\Delta\Sigma$ Modulator is to convert the analogue input voltage in to a 1-bit pulse stream. The loop filter/integrator can be either switched capacitor or contin-

[1] Decimation is the process of reducing the sampling rate of a signal, A system component that performs decimation is called a decimator

uous time. Switched capacitor filters are easier to implement on silicon then continuous time and the frequency characteristics scale with the clock rate. The purpose of the digital filter is to remove the out of band quantization noise and provides anti-aliasing to allow re-sampling at a lower sampling rate. There are numerous ΔΣ ADC architecture and the choice usually involves trade-offs between resolution, circuit complexity and stability. Through extensive simulations, it was found that a 1st-order ΔΣ ADC with an over-sampling ratio (OSR) of 32 is sufficient to achieve femtosecond resolution avoiding stability and complexity issues often associated with higher order converters. A block diagram of the 1st-order ΔΣ ADC is shown in Figure 2.21. It consists of an integrator and a single bit quantizer. The oversampling ratio (OSR) of the modulator is given by the following equation

$$OSR = \frac{f_s}{2 * f_B}$$

Where f_s is the sampling frequency and fs is the input signal bandwidth. For this application the OSR is set to 32, in order to provide to appropriating noise shaping that is required to achieve the high resolution time measurement [8].

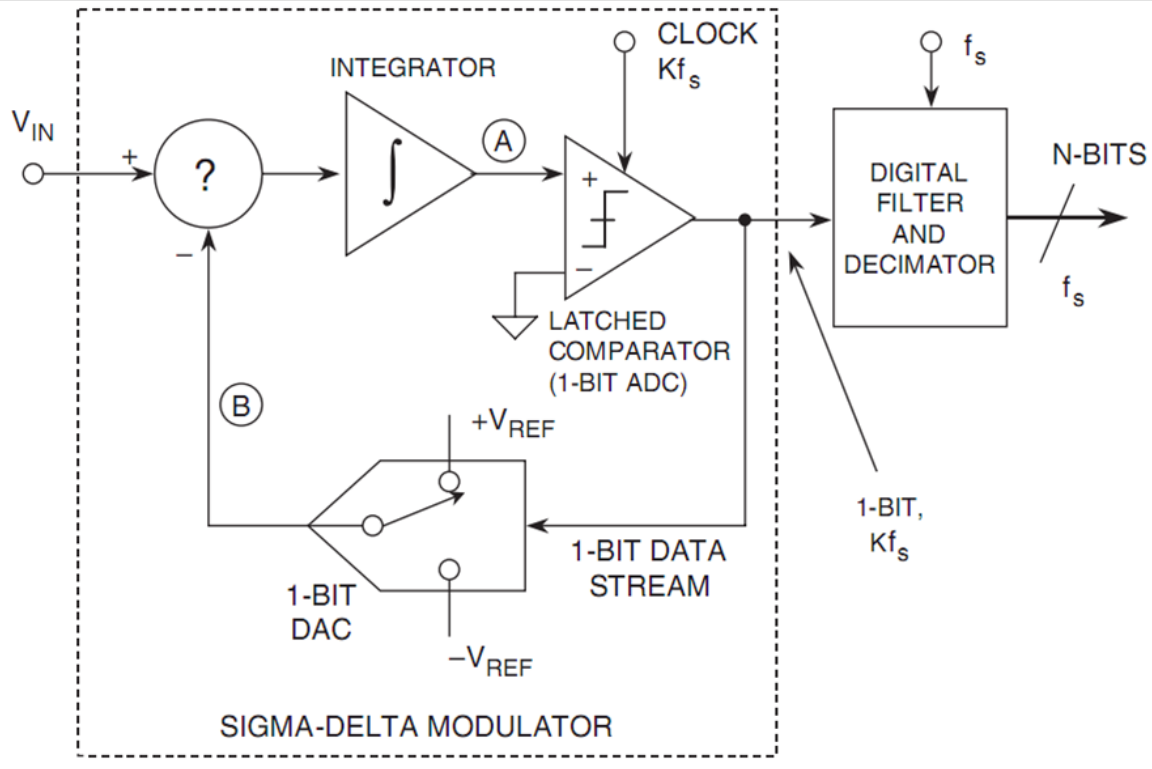


Figure 2.21: Block diagram of the 1st Order ΔΣ ADC [handbook]

By equations we could see that doubling the oversampling ratio of this circuit reduces the noise by 9 dB and provides 1.5 bits of extra resolution. But, without feedback, doubling the oversampling ratio of this circuit reduces the noise only by 3 dB and provides

only 0.5 bits of extra resolution. This shows the importance of the feedback roll in the system.

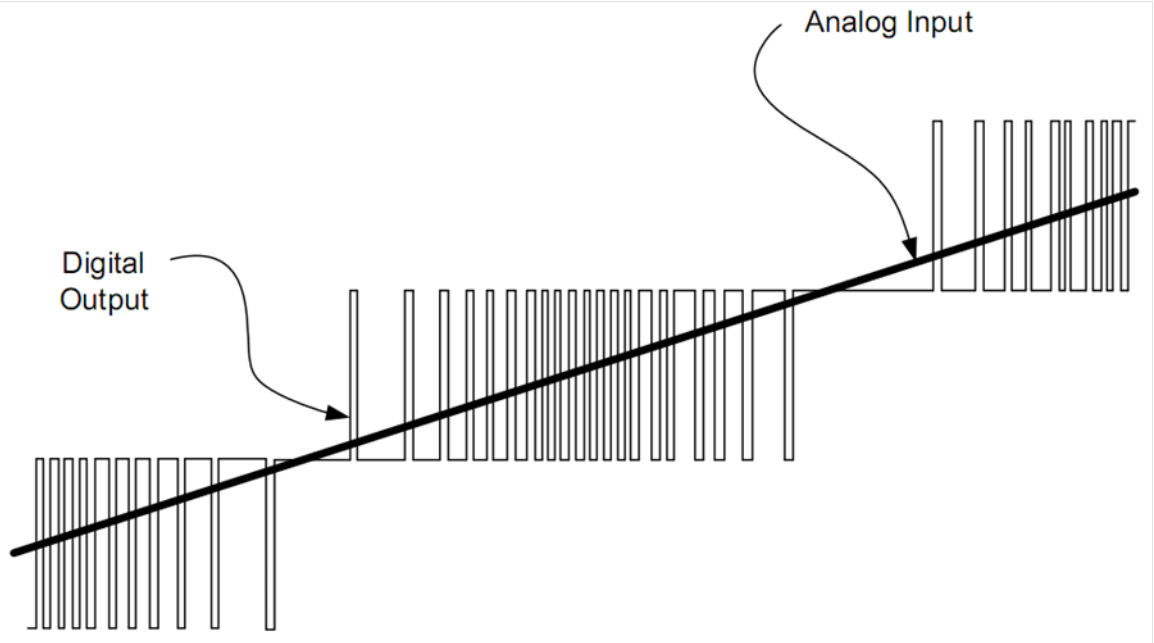


Figure 2.22: Example of a multilevel quantizer output when the input is a ramp[3]

Figure 2.22 shows an example of a multilevel quantizer output when the input is a ramp. As we can see the average of the digital output tries to follow the input [3].

Figure 2.23 shows the second order sigma-delta modulator. We can see that the differences between the second and the first order modulator are the existence of another integrator (accumulator), and another feedback path from the output.

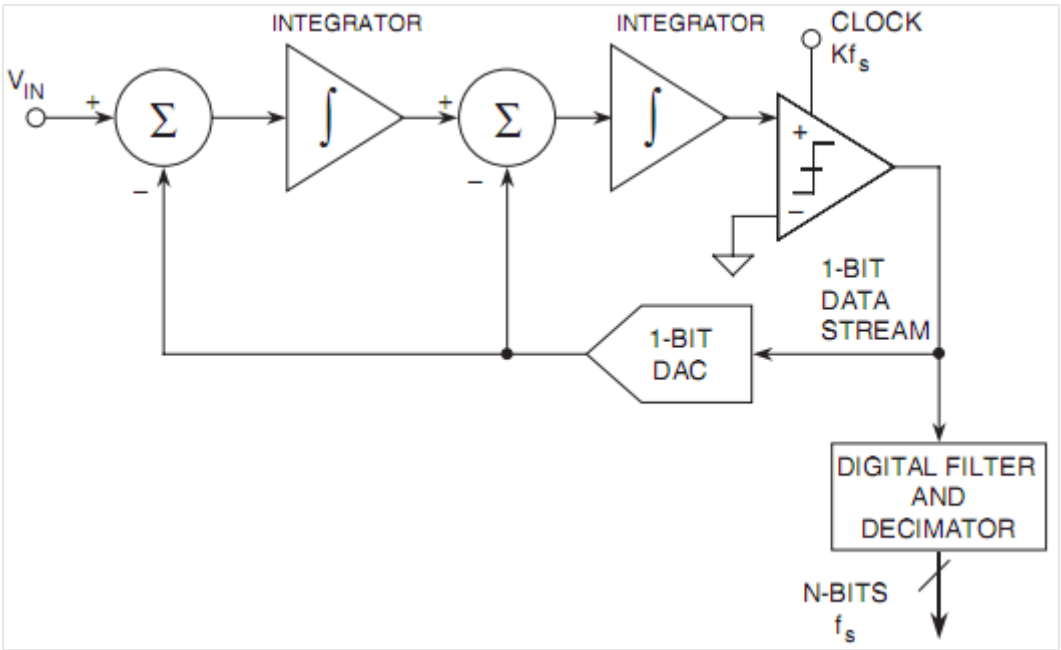


Figure 2.23: Second order ΔΣ ADC [handbook]

We can find for this second order that doubling the OSR results in decreasing the quantization error by 15 dB and provides 2.5 bits of extra resolution. This technique can be extended to higher order loop ADC by adding more feedback loops to the circuit. In general, when a modulator has L loops, doubling the OSR results in decreasing the quantization error by $3(2L-1)$ dB and provides $(L–1/2)$ extra bits of resolution. The problem behind not using L order sigma-delta modulator, where L is greater than 2, is the stability of the system. Due to feedback, signal at the input of the quantizer may accumulate. This leads to overload the modulator, and makes the modulator unstable [3].

Chapter 3

Time Based Analog To Digital Converter

3.1 Introduction

A time based analog to digital converter (TADC) converts the analog to digital conversion in an indirect manner by first converting the analog input to time representation, and then quantized this time representation into digital code. So, TADC uses indirect way of conversion from analog to digital. When we scale technology the voltage swing decreases. The SNR degrades as noise does not scale with the same ratio. Cascading becomes very difficult as supply voltage becomes around1 volt [3].

This section will review some examples of TADC and as stated before TADCs are indirect ADCs. Figure 3.1 shows main types of TADC

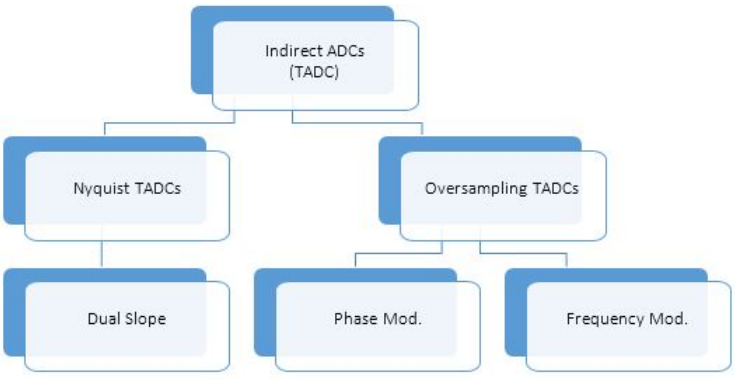


Figure 3.1: Main types of TADC

3.2 Nyquist TADCs

Nyquist ADCs are type of ADC where the sampling frequency is twice input signal frequency (Nyquist rate).

3.2.1 Dual slope TADC

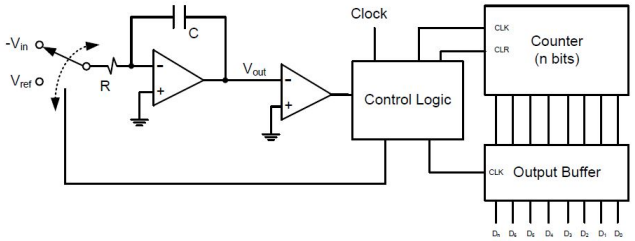


Figure 3.2: Dual slope ADC[3]

Figure 3.2 shows Dual slope Nyquist rate ADC the conversion is done through two stages. The first stage is when the input switch is connected to V_{in} , in this stage the switch remains connected to V_{in} for Fixed time equal to T_1 so at this stage the output of integrator will be equal to

$$V_{out} = -\int_0^t \frac{V_{in}}{RC} dt = \frac{V_{in} t}{RC} \tag{3.1}$$

during this stage and at the end of this stage

$$V_{out} = \frac{V_{in} T_1}{RC}. \tag{3.2}$$

The second stage is when the switch is connected to known voltage V_{ref} . where the V_{out} will be equal to

$$V_{out} = -\int_{T_1}^t \frac{V_{ref}}{RC} dt + \frac{V_{in} T_1}{Rc} \tag{3.3}$$

during this stage. so according to the last relation

$$V_{out} = -\frac{V_{ref}}{RC}(t - T_1) + \frac{V_{in} T_1}{Rc} \tag{3.4}$$

will becomes zero after certain time T_2 which will be equal to

$$T_2 = T_1 \frac{V_{in}}{RC} \tag{3.5}$$

The counter will count time taken by Vout to become zero,T_2, where we can use last relation between to calculate V_{in}.

3.3 Oversampling TADC

Oversampling ADCs are type of ADC where the sampling frequency is very large compared to input signal frequency. Here the conversion is divided to two stage. In first stage the input voltage is converted to time or frequency representation and second stage this time or frequency representation is converted to digital output.

So in Oversampling ADC the input analog voltage modulates a reference signal, there are two types of modulation phase and frequency modulation.The main types of oversampling TADC is shown in figures 3.3 and 3.4

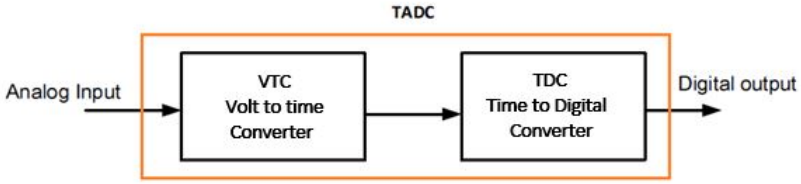


Figure 3.3: Simplified diagram of VTC/TDC based TADC

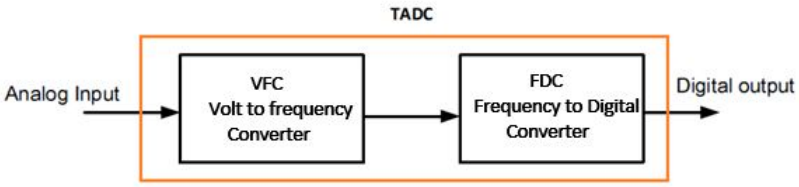


Figure 3.4: Simplified diagram of VFC/FDC based TADC

3.3.1 Phase modulation

In phase modulation the angle is varied linearly with the message signal [9].

Let the angle is

$$\Theta = 2\Pi F_c + Km(t) \tag{3.6}$$

where $m(t)$ is the input signal, So the modulated signal is

$$S(t) = A_c \cos\Theta \tag{3.7}$$

.So here we makes input volt modulates reference signal $S(t)$ by modulating the angle of $S(t)$. In fact phase modulation is used in VTCs by making input volt linearly modulates the phase of a pulse reference signal. which means that the input volt introduce some delay in a reference pulse as will be discussed later in VTC Chapter.

3.3.2 Frequency modulation

In frequency modulation the instantaneous frequency is varied linearly with the message signal [9].

Let the instantaneous frequency is

$$f(t) = f_c + Km(t) \tag{3.8}$$

where f_c represent the carrier demodulated frequency, So here we makes input volt modulates reference signal frequency, So the angle of the modulated signal will be

$$\Theta(t) = 2\Pi f_c t + 2\Pi K \int_0^t m(\gamma) d\gamma \tag{3.9}$$

and thus the time domain modulated signal will be

$$S(t) = A_c \cos(2\Pi f_c t + 2\Pi K \int_0^t m(\gamma) d\gamma) \tag{3.10}$$

The frequency modulation is a non linear modulation process as the modulated signal $S(t)$ is a nonlinear function of input signal $m(\gamma)$ [3]. In volt to frequency converter the input analog volt modulates the frequency of a reference signal the deviation of the frequency is then sensed by frequency to digital converter FDC which converts this deviation to digital output.

Chapter 4

Voltage to Time Converter (VTC)

This chapter will present the analysis of the first main block of the time-based ADC which is VTC including an explanation of the basic element in VTC and the circuit on which the design proposed is based, the challenges that have been met in this design, improvements to make best use of old design and derivation of output delays using timing Diagram.

Several VTC's for different applications have been proposed. Most of these VTC's cores are based on basic current starved as it will be discussed in this chapter. In [10] Djemouai proposed a basic current starved inverter differential delay cell and in [11] added weak cross coupled inverters to shorten the transition times of the inverters. Dudek proposed a similar VTC but added a weak CMOS nfet with its gate connected to the supply in order to ensure that the VTC operates at very low input voltages [12]. Watanabe proposed a delay unit which consists of a series of inverters with sources of PMOS connected to the input voltage [13]. Gray proposed inverter delay units in which the clock derives NMOS gates and the delay is controlled through the bias voltages of the PMOS gates resulting in lower switching noise but higher current consumption [14]. these previous VTC's are not suitable for high speed high resolution time based ADC because they are not sufficiently linear or conversion is not sensitive.

Pekau proposed a VTC that makes use of all above circuits with a new linearization scheme targeting improved linearity and higher sensitivity but still not sufficiently high [15]. Figure 4.1 shows improvement in linearity and linearity error of VTC proposed of Pekau but still need improvement on linearity to be used for High resolution TDCs. A highly linear VTC circuit is proposed in 2013 which achieves higher linearity and higher sensitivity compared to all previous [16]. It is referred to be pulse width modulation (PWM) as it will be discussed in this chapter.

The proposed VTC in this thesis is build upon the design of [16] and in the following section the basic circuit design and analysis is explained.

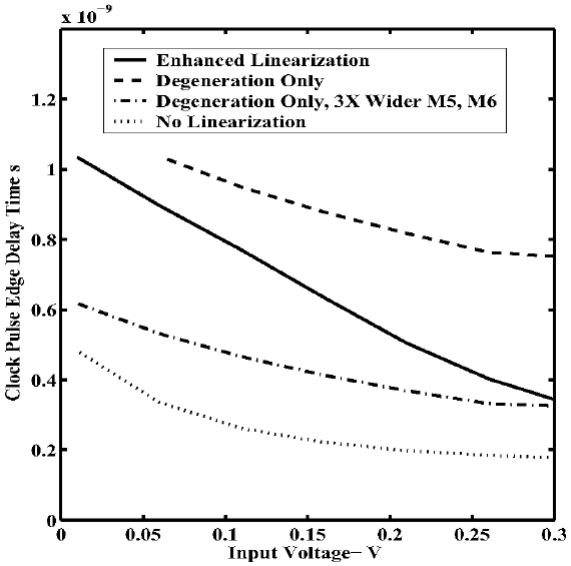


Figure 4.1: Simulated transient clock pulse edge delay versus input voltage for the VTC proposed in [15] with different linearization methods

4.1 Current starved inverter

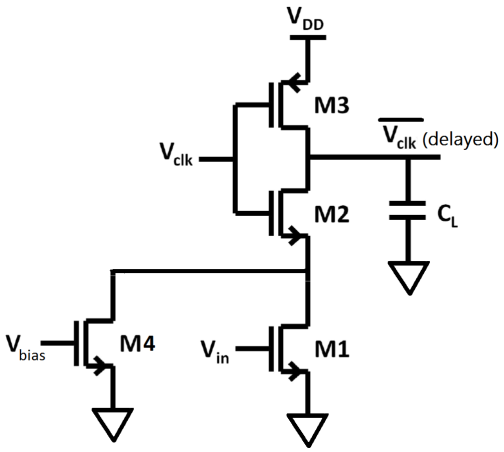


Figure 4.2: Current Starved Inverter

The most simplified schematic of the VTC circuit is the basic current starved inverter as shown in figure 4.2. The output of this circuit is a delayed version of the Vclk. This delay is changing with Vin. The current moving in M1 is changing with Vin according to equation

$$I = \frac{\mu_n Cox \frac{w}{l}}{2}(V_{GS} - v_{th})^2$$

where VGS = Vin, This current is used for charging and discharging the CL, so it affects the delay produced on Vclk. This delay is constant for the same Vin. M4 is weak

device with low aspect ratio used to ensure that the inverter operates at very low input voltages.

4.2 Basic Design and Analysis

Figure 4.3 shows the block diagram of VTC in [16]. The analog input voltage, Vin , is applied to two current starved circuits t_{RISE} and t_{FALL} shown in Figure 4.4 [16]. The rise time of the inverter in tRISE circuit is controlled by Vin through PMOS Pb2. Pb3 is a weak minimum size transistor is used to be another current path when Pb2 is OFF (when Vin is close to supply voltage VDD). Similarly, the fall time of the inverter in t_{FALL} circuit is controlled by Vin through NMOS Na2. Na1 is a weak minimum size transistor is used to be another current path when Na2 is OFF (when Vin is below the threshold voltage of Na2). VCLK , the input clock is applied to the t_{FALL}circuit, the output is V2 which is an inverted delayed version of V_{CLK} and the falling delay is controlled by the input voltage Vin.

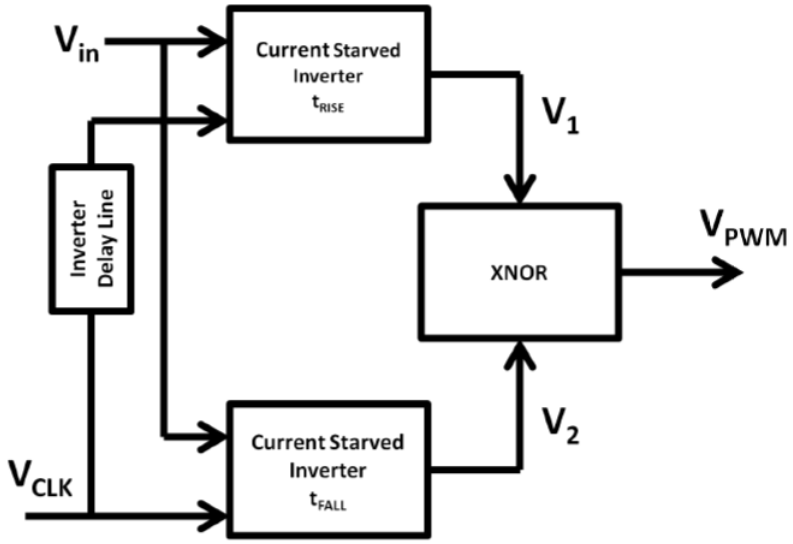


Figure 4.3: Block diagram of the VTC circuit [16].

On other hand VCLK is applied to an inverter delay line (i.e. odd chain of CMOS inverters) and the output ($V_{CLK_{bar}}$) is then applied to the t_{RISE} circuit. Accordingly, the output V1 is a delayed version of V_{CLK} and its rising delay is controlled by the input voltage Vin. V1 and V2 are then applied to CMOS XNOR as inputs and the result is a pulse width modulated output Vpwm.

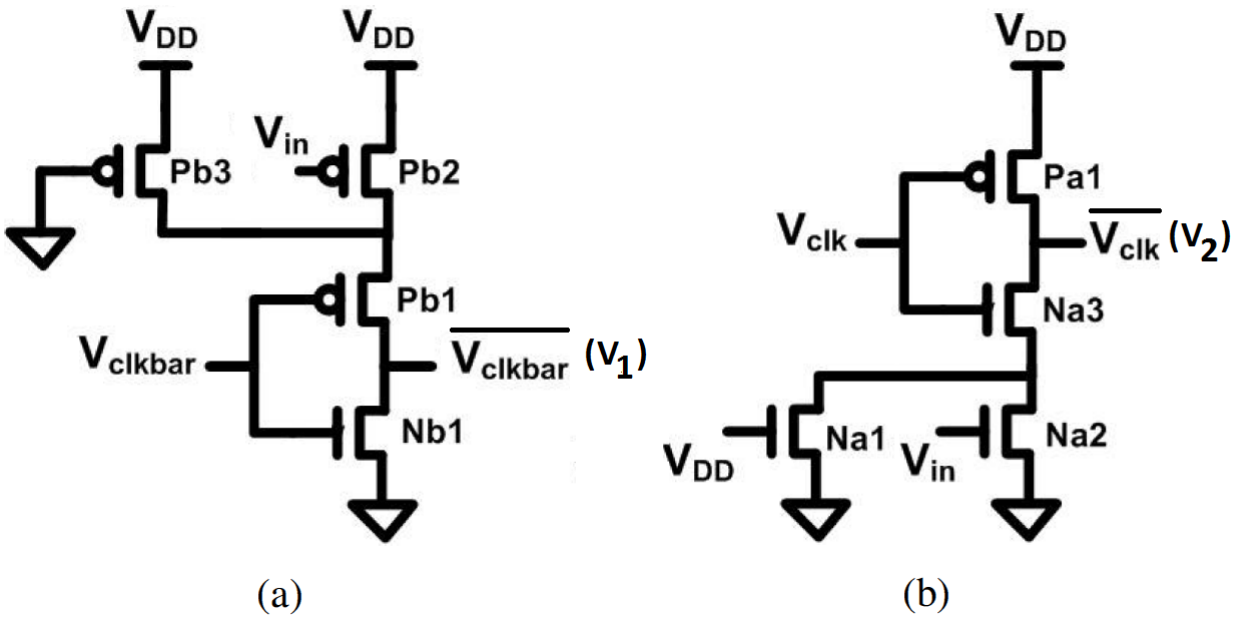


Figure 4.4: (a) The tRISE current starved inverter circuit and (b) the tFALL current starved inverter circuit [16].

Figure 4.5 shows the timing diagram of a given value of V_{in} , the output of XNOR circuit is the V_{pwm} has two pulses. The first pulse has width equals to$\Delta + t_r - t_f$ where Δ is delay due to inverter delay line which is constant, $t_r$is the delay due to t_{RISE} current starved inverter which is controlled by V_{in} and t_f is the delay due to t_{FALL} current starved inverter which is also controlled by V_{in}. The second pulse has width equals to Δ which is fixed as it is independent of V_{in}. As value of V_{in} increases,t_f is decreased as the gate-to-source voltage of NMOS Na2 (i.e.$V_{GS} = V_{in}$) increases, whereas t_r is increased as the source-to-gate voltage of PMOS Pb2 (i.e.$V_{SG} = V_{DD} - V_{in}$) decreases. The VTC proposed in [16] stated that it is essential to first pulse always larger than zero in all dynamic range and that is corresponds to have:

$$\Delta = min\{(t_{f_{max}} - t_{r_{min}})\} + \Theta \tag{4.1}$$

where $\Theta$is a safety margin and is selected to be the delay of one CMOS inverter with minimum size.

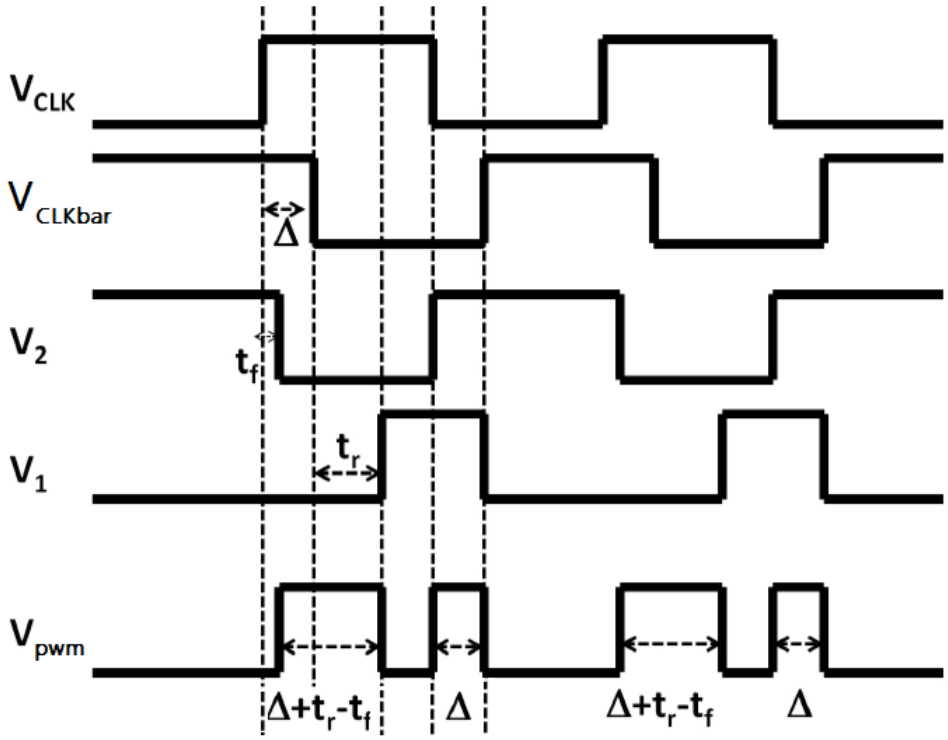


Figure 4.5: Timing diagram of the VTC circuit for a given value of Vin [16].

The VTC proposed in [16] sizing is chosen to have maximum sensitivity and minimum linearity error for CLK frequency equals 500 MHZ. It provides a dynamic input voltage range of 150mV. outside this range the linearity error is larger than 1%. The sensitivity of the circuit of VTC is defined to be the slope of the$t_{pwm} - V_{in}$ curve and found to be 3.46 ps/mV in [16]. The difference betweent_{RISE} andt_{FALL} helps in providing higher linearity and sensitivity as they are inversely controlled by Vin as shown in Figure 6. When applying two different inputs to VTC circuit in [16], Figure 4.6.a shows output when Vin = 200mV while Figure 4.6.b when Vin The frequency spectrum of samples in case of (a) V_{in} = 200mV or (b) V_{in}= 350mV. It is clear that as Vin increases the first pulse is increasing while the second pulse is fixed (resulting from Δ).

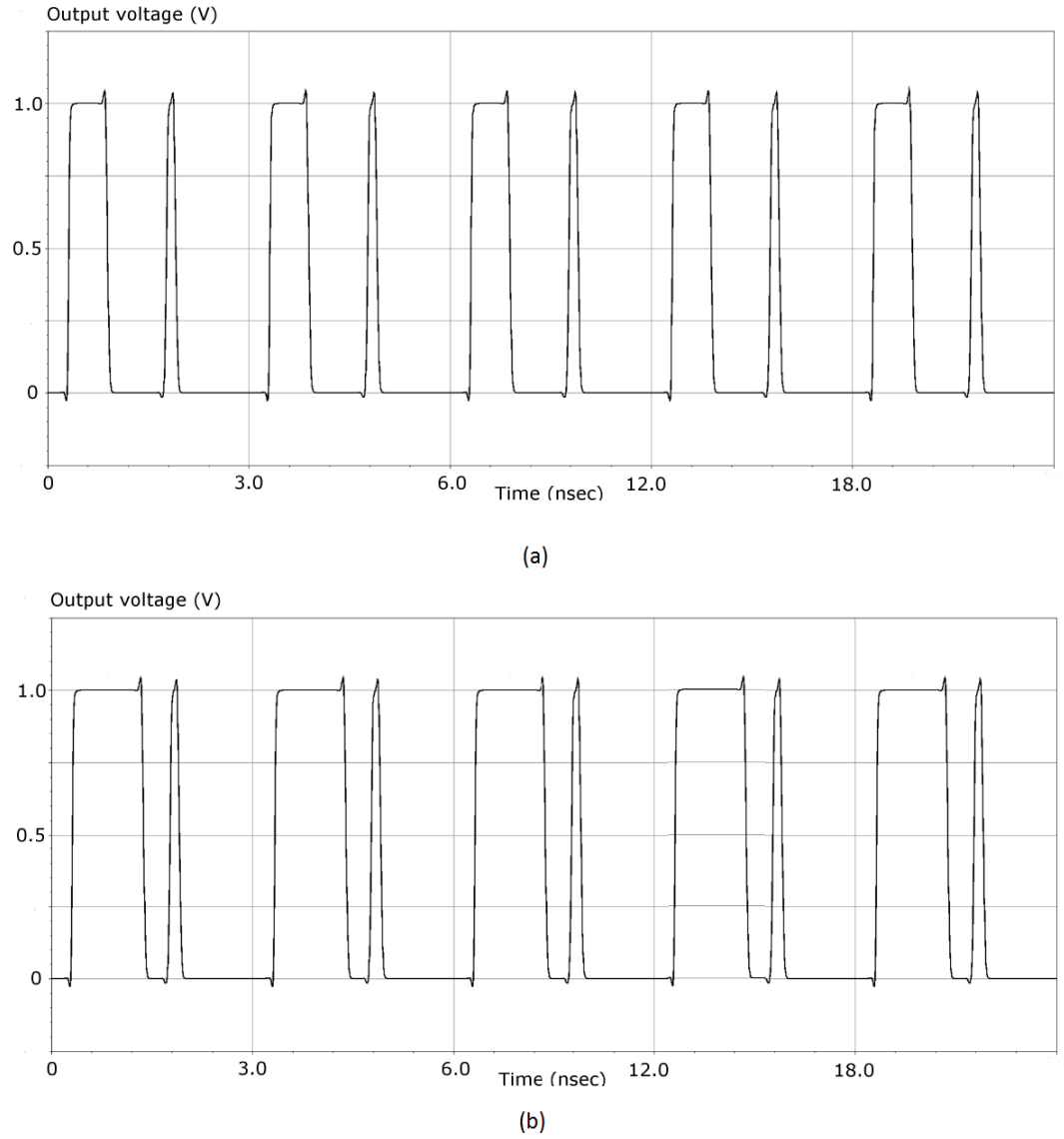


Figure 4.6: Output voltage pulses V_{pwm} when (a) V_{in} = 200mV and (b) V_{in}= 350mV [16].

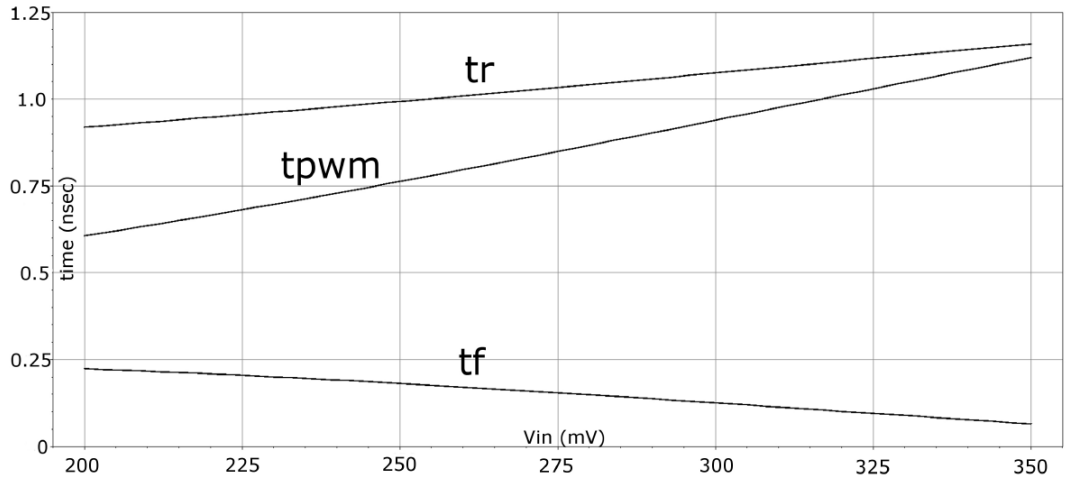


Figure 4.7: Rising delayt_r , falling delay tf and t_{pwm} versus the input analog voltage V_{in} [16].

4.3 Challenges and proposed Solutions.

The design of TDC in this thesis requires to have a VTC based on PPM, but VTC in [16] is referred to be PWM. Two questions arise here: (1) What makes this thesis decides to use a PWM design in spite of being based on another modulation of that required in the time based ADC proposed in this thesis ?, (2)How can PWM to be used to make a PPM ? and this is the first challenge.

The reason for choosing the VTC in [16] is:

- High linearity and high sensitivity.
- Digital assisted system (the analog part becomes very small in the whole TADC structure).
- Built on 65nm technology.

4.3.1 Conversion from PWM to PPM

To change the PWM to be PPM. From the previous section it is shown that output is of two pulses. The first pulse is variable and dependent on Vin while the second is fixed for all outputs. The VTC is said to based on Pulse Position Modulator (PPM) or Pulse Width Modulator (PWM), depending on the delay whether is applied to one or both edges of the input clock pulses. PPM is about producing shifted versions of a single pulse where the shift. While PWM is about The idea is to use the positive edge of the first pulse as a clock signal for a D flip-flop to get a START signal and to use the negative edge of the same pulse for producing a STOP signal. The difference between START and STOP signals corresponds to the width difference (or t_{pwm}). In other words the STOP

signal is a delayed version of START signal at a given input by delay1. and at another input is another STOP signal is a delayed version of another START signal by delay2. The difference between delay1 and delay2 is constant ($\Delta new = delay_1 - delay_2$) if we move with a constant step in Vin and this exactly equivalent to the width of the first pulse obtained in [16].

Figure 4.8 shows the timing diagram of the improvement proposed on VTC to be used as PPM. To implement this timing diagram. The second pulse width is fixed so it will not be used to represent the input. So, it removing it will not affect the output. This is done using 2-input AND gate. The AND inputs are V_{CLK} andV_{pwm} and the output is V_{PWM} as shown in Figure 8. Now, START and STOP signals be obtained from V_{PWM}. By using the pulse (V_{PWM}) as a clock for a D flip-flop (with initial state of zero) and input of V_{DD} , START signal (Qrf) is the output of D flip-flop. But, after defining the dynamic range of the VTC, this START signal should be shifted by a constant shift which is equal to the width of first V_{PWM} obtained from the first V_{in} in dynamic range. This is done because the TDC as it will be discussed later in chapter 4 measures the difference between the START and STOP signal which changes by step (or resolution) equals to Δ_{new} so it is needed to have the lowest difference to be '0' (START_delay - STOP_delay = 0) at lowest V_{in} in dynamic range. This is shown in Figure 4.9 So, the difference will be 0 then Δ_{new} then $2 * \Delta_{new}$ as moving by Vin step equivalent to Δ_{new}.

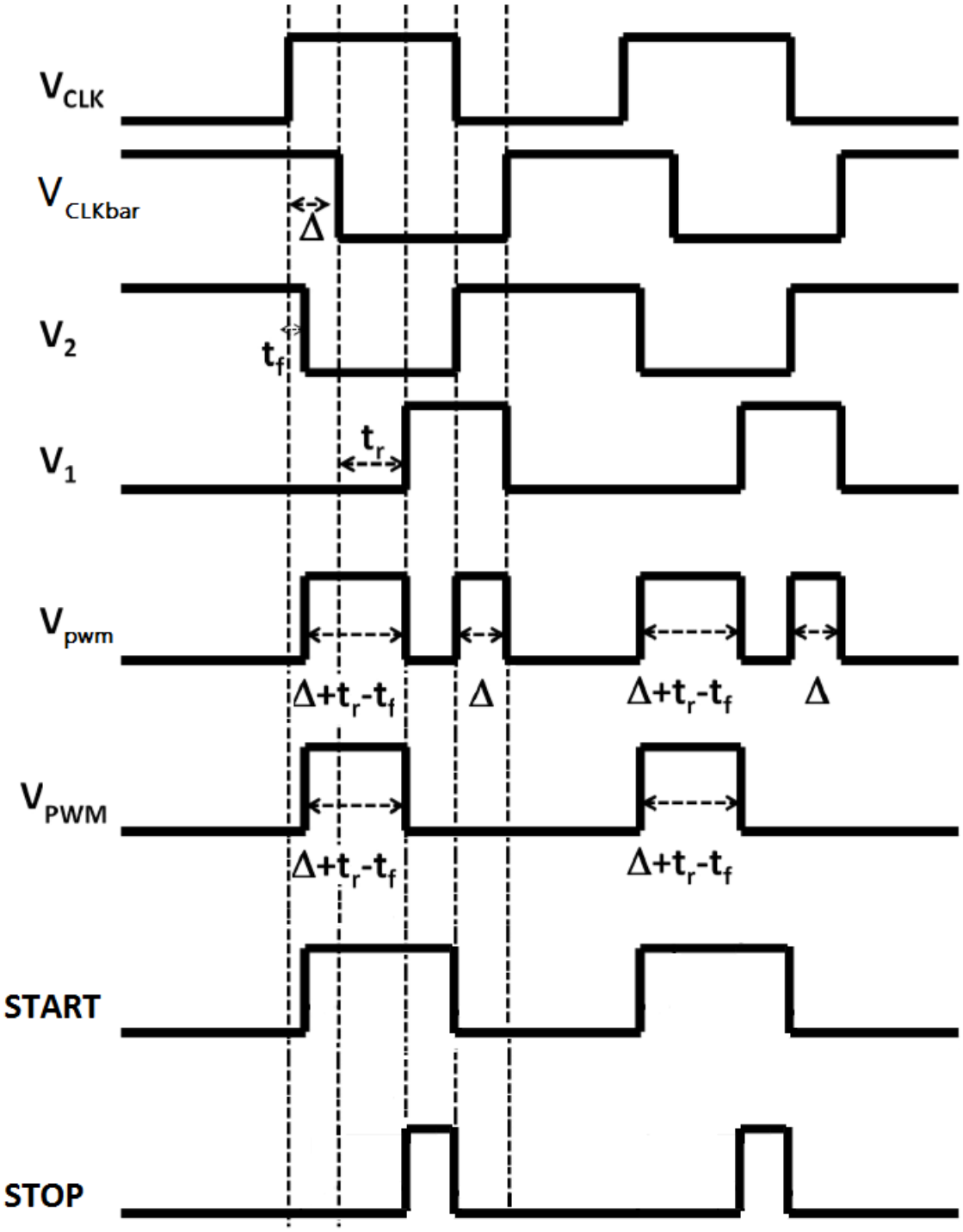


Figure 4.8: Timing diagram of the Proposed Improvement of VTC circuit for a given value of V_{in}.

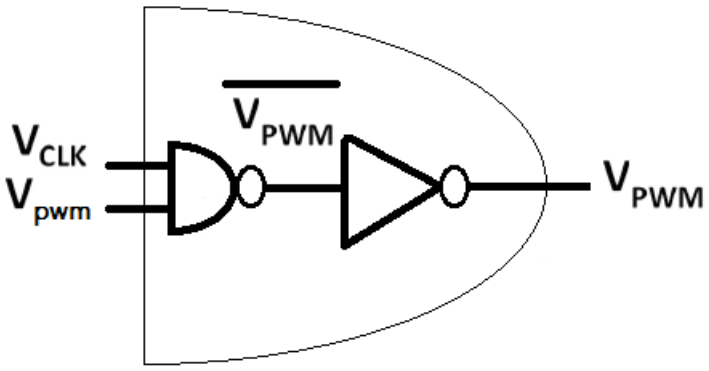


Figure 4.9: Using NAND followed by NOT to form AND.

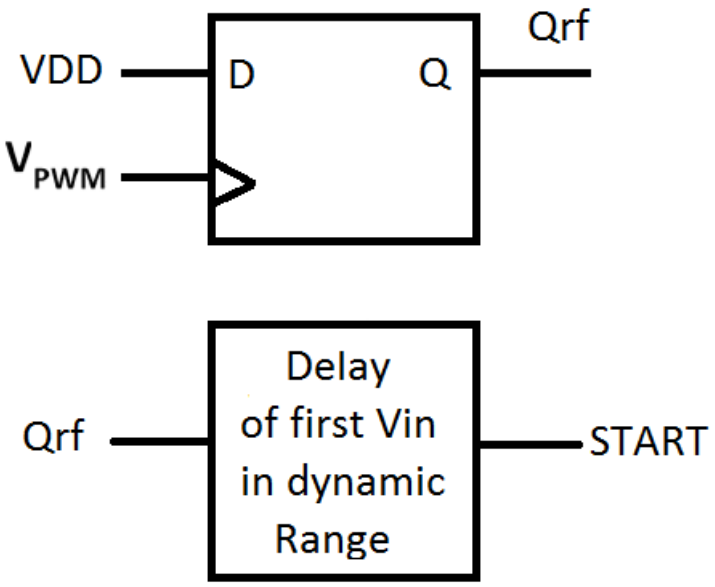


Figure 4.10: Obtaining START signal from VPWM using D flip-flop.

STOP signal is obtained by using 2-input AND gate, its inputs are the START signal and V'_{PWM} (inverted pulse of V_{PWM}) obtained from NAND gate shown in Figure 4.10. START and STOP signals are reset every V_{CLK}.

4.3.2 Large Dynamic Range

The second challenge is the high resolution in the proposed design in this thesis (8 bit), the TDC as it will be discussed in next chapter, it measures resolution of 3.9ps. In other words Δ_{new} = 3.9ps over a range of 999ps (≈1ns). To have a perfect 1 LSB = 1mV in V_{in} which corresponds to 1 LSB = 3.9ps in time difference, dynamic range of at least 256mv is required. There are only three tracks to be moved on. The first track is to maintain the

high sensitivity of the VTC in[16] which was 3.46ps/mV or try to increase it to 3.9ps, but on the other hand dynamic range need to be increased from 150mV to 256mV at least or more. The second track is to maintain low dynamic range (150mV), but on the other hand dramatically high sensitivity is needed (6.6667ps/mV) which is nearly impossible till now. The third track is to have lower sensitivity say 2ps/mV, but on the other hand a dynamic range of 500mV or larger is needed which is impossible with $V_{DD} = 1.2$ used in this technology and also bad due to low sensitivity.

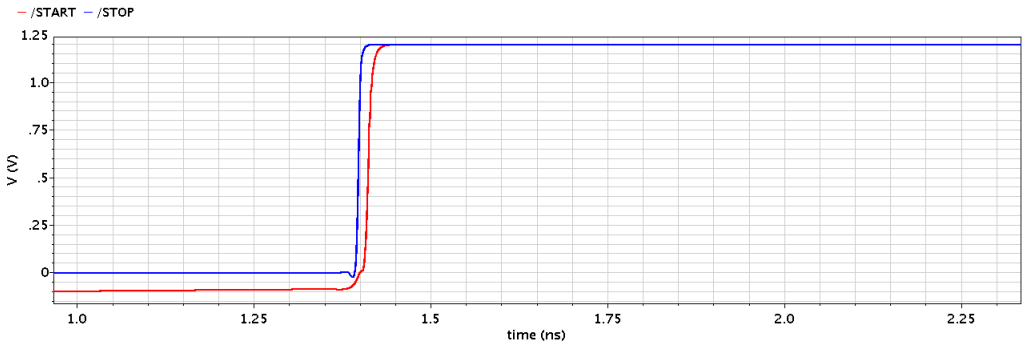
The most suitable and better solution is using the first track so that maintaining high sensitivity and targeting higher dynamic range which is challenging but not impossible. Sizing is made on trise and tfall to achieve the required dynamic range as in table 4.1 .

Transistor	Length	Width
Na_1	60n	200n
Na_2	60n	9u
Na_3,Nb_1	60n	15u
Pa_1,Pb_1	500n	30u
Pb_2	400n	1u
Pb_3	250n	200n

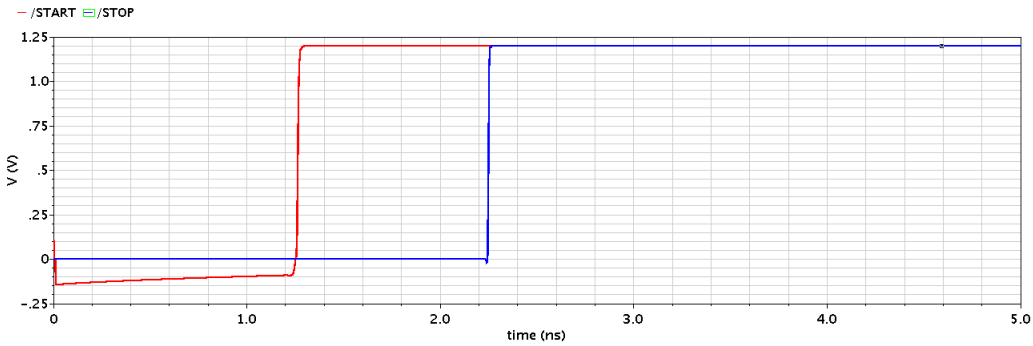
Table 4.1: Sizing of TRISE and TFALL

4.4 Simulation Results and Discussions

The voltage sensitivity and linearity of the proposed VTC in this thesis we simulated by parametric analysis on sizing of the main blocks of VTC in[16] which are t_{RISE} ,t_{FALL} and inverter delay line. Also, by sweeping the DC input voltage V_{in} and measuring the output signals START and STOP. These measurements are performed by using transient analysis with industrial hardware calibrated CMOS 65nm transistor device models, provided by TSMC. Figure 4.11.a and 4.11.b shows the output signals START and STOP when$V_{in} = 384mV$ (first input in dynamic range) and $V_{in} = 675mV$ (last input in dynamic range), respectively. in Figure 4.11.a START signal lead the STOP signal by 10.1 ps although it is required to be zero (START - STOP = -10.1ps) and that is equal to setup time of D flip-flop so that it can operate in TDC as it will be discussed later



(a)



(b)

Figure 4.11: Output signals START and STOP (a) Vin = 384mV and (b) Vin = 675mV.

Figure 4.12 shows the difference between START and STOP signals versus input analog voltage V_{ib} that changes from 384mV to 675mV providing a dynamic input range of 291mV. Outside this range error is larger than 1.5%. The sensitivity of this VTC circuit is defined as the slope of the delay-V_{in} curve which is denoted by ρ= 3.43 ps/mV.

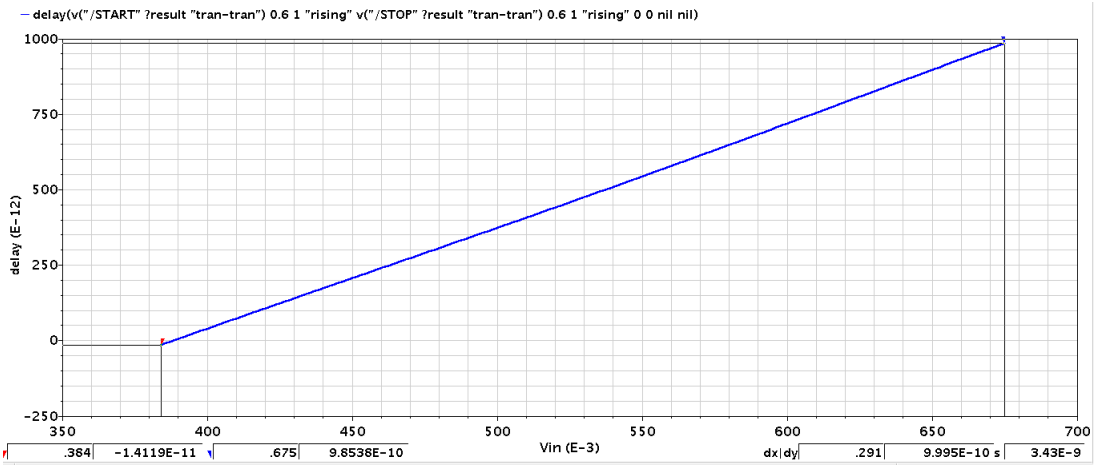


Figure 4.12: The difference between START and STOP signals versus input analog voltage V_{in}.

4.5 Layout of VTC basic elements

4.5.1 T_{RISE}

Area = 32.7*3.1 $(um)^2$

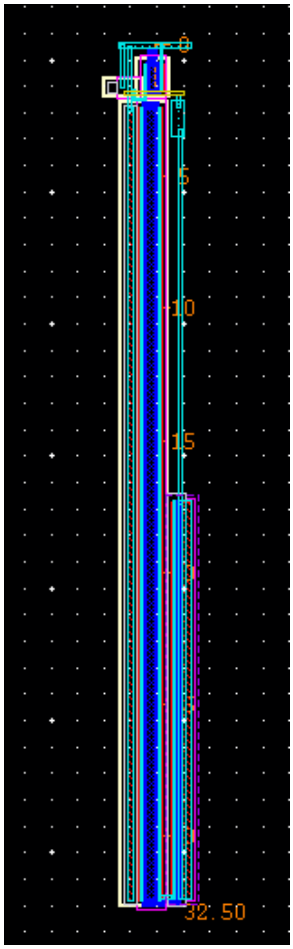


Figure 4.13: TRise Layout

4.5.2 T_{FALL}

Area = 5.4*31 $(um)^2$

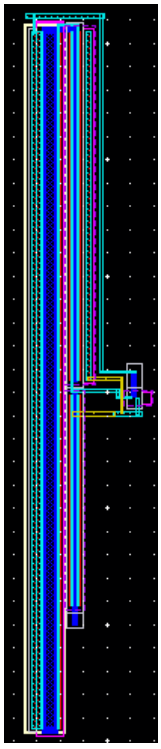


Figure 4.14: TFALL Layout

4.5.3 Delay Inverter Line

Area = 7*2.3 $(um)^2$

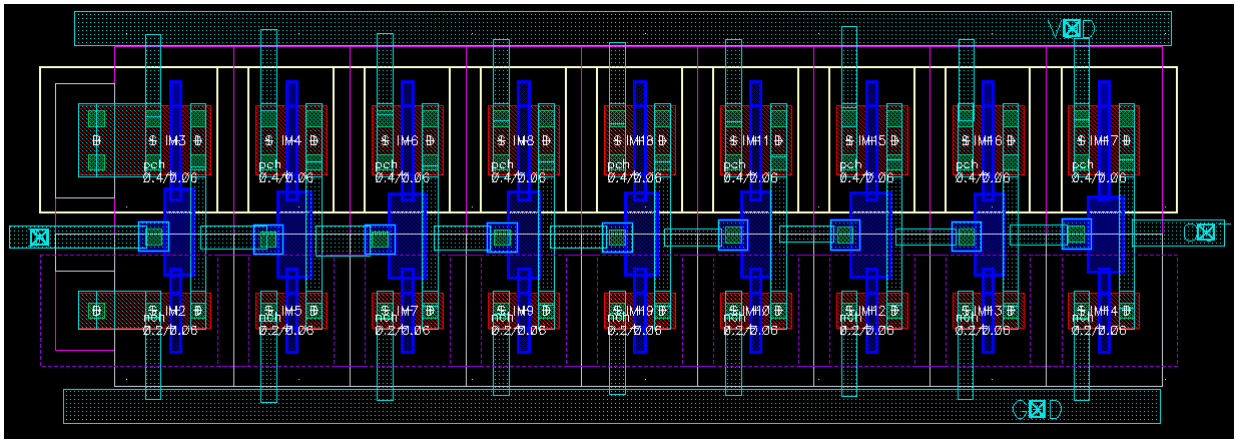


Figure 4.15: Delay Inverter Line Layout

Chapter 5

Time to Digital Converter (TDC)

5.1 Introduction

The basic task of TDC is to convert the time representation which corresponds to the analog input into a digital code. The conversion is done by comparing the edges to output signals generated from VTC[1]. In this section, we will refer to the signals as LEAD (START) and LAG (STOP) signal. So, the task of the TDC is to convert the time interval between a LEAD and a LAG to a digital output.

The function of the TDC can be done using a simple counter, as shown in the following section, by making the counter starts to count when the start signal enables it, And stops counting when stop signal reaches. The high resolution required in most applications, in order of picoseconds, makes the implementation of the TDC using a simple counter very complex and impractical because of the need to unreasonably high clock counter frequency [3] [17]

5.2 Types of TDC

5.2.1 Single counter TDC

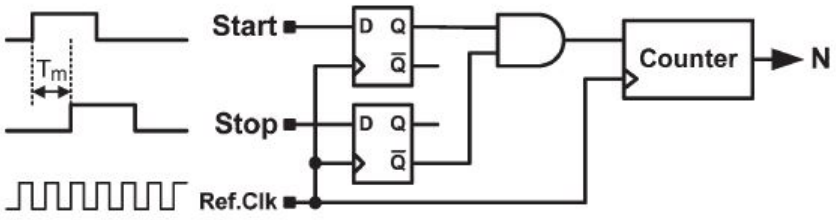


Figure 5.1: Single counter TDC[18]

[1]The VTC modulates a reference signal and generate two inputs for TDC the reference signal (LEAD signal) and delayed version of it (LAG signal) which is corresponding to input analog signal

Figure 5.1 shows the simplest structure of a counter TDC. It uses a counter running on external high frequency clock to measure the time difference (T_m) between two rising edges. The counter stops when the START signal catches the STOP signal. One of the disadvantages of this type of TDC is that its resolution is limited by the clock period of the Reference clock.

5.2.2 Cyclic Pulse-Shrinking TDC

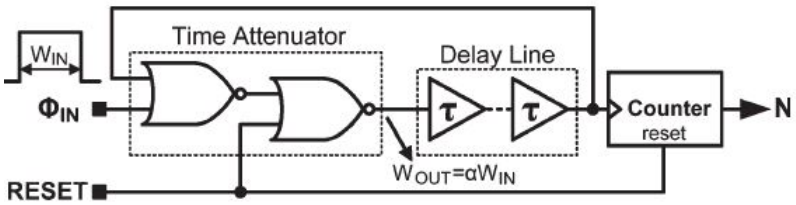


Figure 5.2: Cyclic Pulse-Shrinking TDC[18]

Using a pulse-shrinking circuit in a feedback loop, as in figure 5.2 . The width of an input pulse W_{IN} is reduced as it propagates around the feedback loop by factor of α. The pulse keep propagating till the width vanishes at this moment the counter stops giving an indication of this pulse width. The delay of the feedback is implemented using chain of inverters to make the feedback delay larger than the duration of the pulse to take the input in the first cycle only.[18]

5.2.3 Simple flash TDC

Figure 5.3 shows simple TDC, The LEAD signal propagates along a chain of delay elements of equal delay. The output of every delay element is connected to the input of a D-flip-flop. As shown in figure 9.4, when the lead signal catches up the lag signal the output of the D-flip-flop will be one.

The outputs of D-flip-flops are represented by a thermometer code which represent the time difference between the lead and the lag signal. This type of TDC uses only digital delay elements and D-flip-flops. The resolution of this TDC is given by a delay introduced by one delay element[12],[18]. Figure 5.4 shows timing diagram of simple flash TDC.

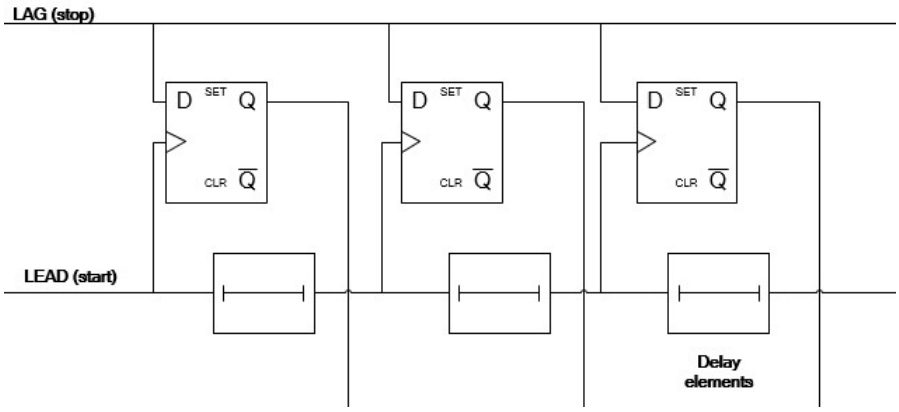


Figure 5.3: Simple TDC

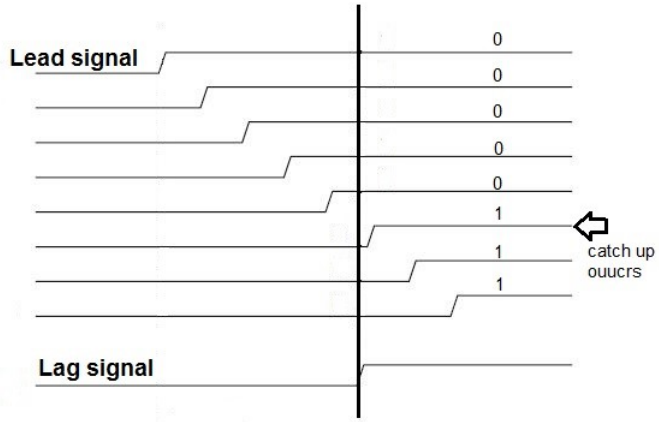


Figure 5.4: Time diagram of LEAD and LAG signals

5.2.4 Vernier TDC

The improvement of the resolution of flash TDC can be achieved by using a Vernier method. In a Vernier delay line (VDL) two delay buffer chains are used. The basic configuration is depicted in Figure 5.5 . The measured time difference is represented by LEAD and LAG signals.

The technique is based on a Vernier principle. The delay of one buffer in the upper delay chain(faster signal) is slightly greater than the delay of one buffer in the lower delay chain(slower signal). As the lead and Lag signals propagate in their cross ponding delay chains, the time difference between the lead and the lag pulse is decreased in each Vernier stage[2] by resolution.

The position in the delay line, at which the LEAD signal catches up with the LAG signal, gives information about the measured time figure 5.6 . In each stage, LEAD and LAG signals are the inputs to D-flip-flops which indicate when the catch up occurs.[18]. Figure 5.6 shows timing diagram of Vernier TDC

[2]The vernier stage consists of upper delay element, D-flip-flops and the lower delay element.

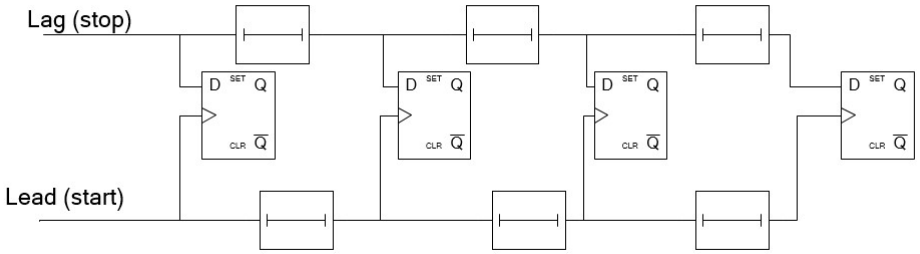


Figure 5.5: Basic structure for VDL TDC

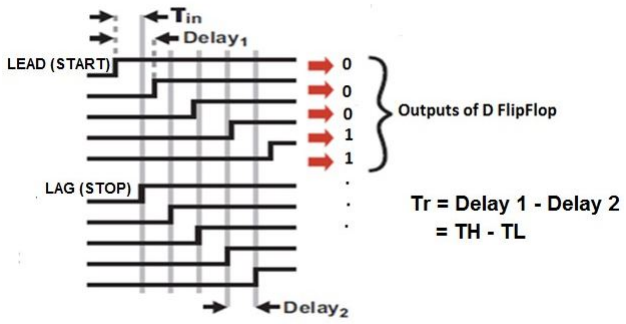


Figure 5.6: Time diagram of 2 level VDL

5.2.4.1 Vernier Ring TDC

The Vernier delay line has a lot of drawbacks as long conversion time and large number of elements. For example, if you need an 8-bit TADC, you will need around 256 D-flip-flop and 512 delay element. Therefore, we must find a way to reduce the number of elements.

There are two famous ways to reduce the number of elements, the Vernier ring and the two level VDL. In Vernier ring technique we make the start and the stop signals loop in limited number of resources more than one time as shown in Figure 5.7[19].

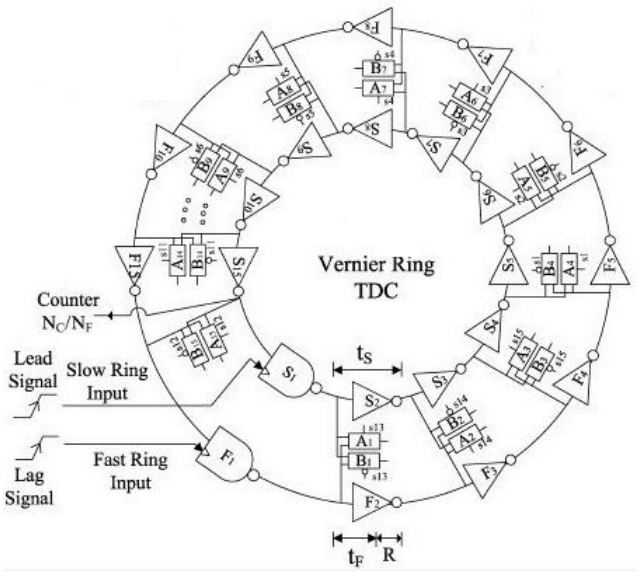


Figure 5.7: Vernier Ring TDC[19]

Here the LEAD and LAG signals use only 15 stage to reach a resolution of 12 bits[19]. But this technique is very complex due to the need of arbiters which indicate at which cycle the catch up occurs. Also this technique puts a lot of constrains on the input frequency.

5.2.4.2 Multi-level Vernier TDC

The other way to implement the VDL with less number of elements is the two level VDL technique. This idea is similar to that of Vernier caliper. We have a coarse level and a fine level. The full dynamic range is divided by the number of delay elements to represent the coarse level resolution. This coarse resolution is divided by the same number of elements to represent the fine level resolution. In other words, in the coarse VDL we can achieve coarse Resolution (T_{CLK}/N), and the signals will be sent to the fine VDL by the help of an interface circuit. Fine resolution (T_{CLK}/N^2) can be achieved in the second VDL[12].

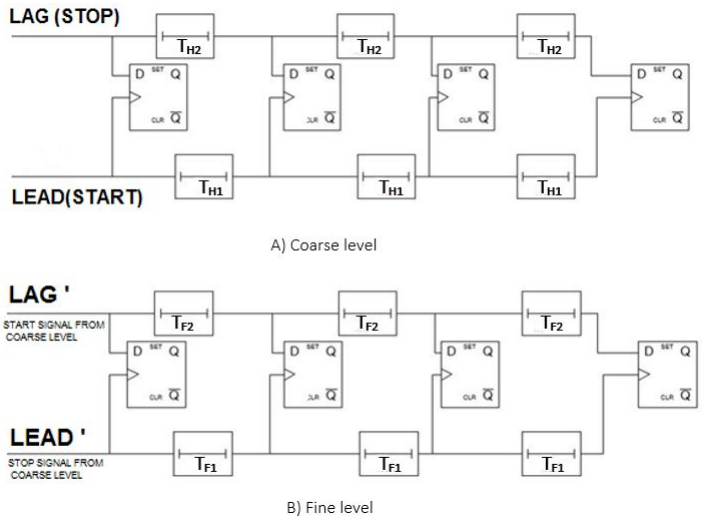


Figure 5.8: Coarse level and fine levels

Figure 5.8 shows the Coarse VDL and Fine VDL. In Coarse VDL, the delay time in down line is T_{H1} which is slightly greater than that of the upper line T_{H2}. The time resolution in is $T_{RCoarse} = T_{H1} - T_{H2} = T_{CLK}/N$. The LEAD signal is the input to the branch with larger delay to make it try to catch the LAG signal. In Fine VDL, the two kinds of delay time are T_{F1} and T_{F2},as in the coarse level the time resolution in Fine VDL is $T_{RFine} = T_{F1}–T_{F2} = T_{CLK}/N^2$[20].

In the coarse level after the catch up occurs as in figure 5.9, the two signals will be transmitted to the fine level which will measure the difference between them with its fine resolution.This figure shows 2 signals having 7 time slots delay between them assume coarse level to have 5 time slots resolution and fine level to have 1 time slot resolution.

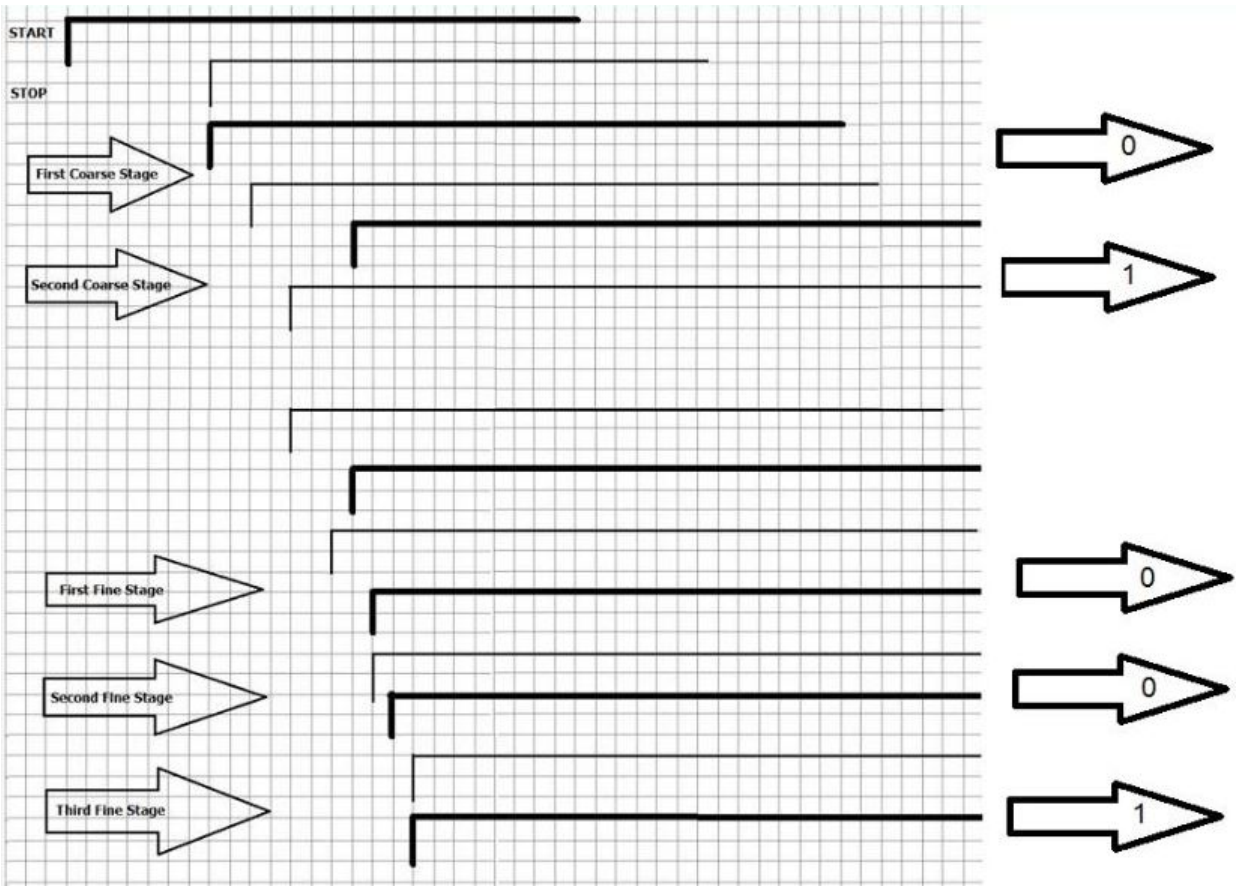


Figure 5.9: Timing diagram of Multi Level VDL showing coarse and fine levels

\In two level VDL architecture, how signals transfer from first level to second level is a critical design. An interface circuit is used to transfer signals correctly.

The NOR gates can detect the HI/LO transition, so we will use NOR gates connected to outputs of each two successive D-flip-flops as in Figure 5.10. For example, if the LEAD signal catches up the LAG signal in the ith Vernier delay element then only $C(i)$ will become HI and so we need to send the two signals ($D(i)$ and $M(i)$) to the fine VDL[20].

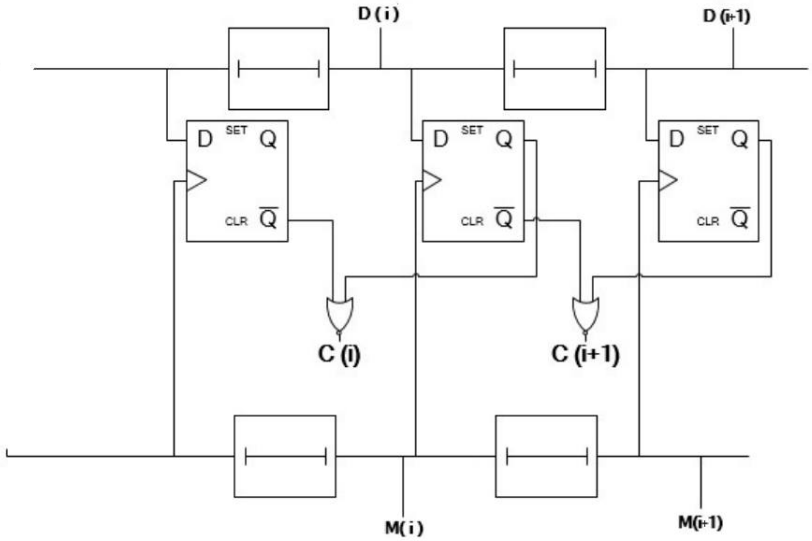


Figure 5.10: Coarse level after connecting NOR gates

If the first D-flip-flop output is one this mean that the two signals have no delay between them.The interface circuit will be as shown in Figure 5.11. Only the output $C(i)$ of NOR gate is HI and can make the signals $D(i)$ and $M(i)$pass to second level VDL for more precise resolving[20].

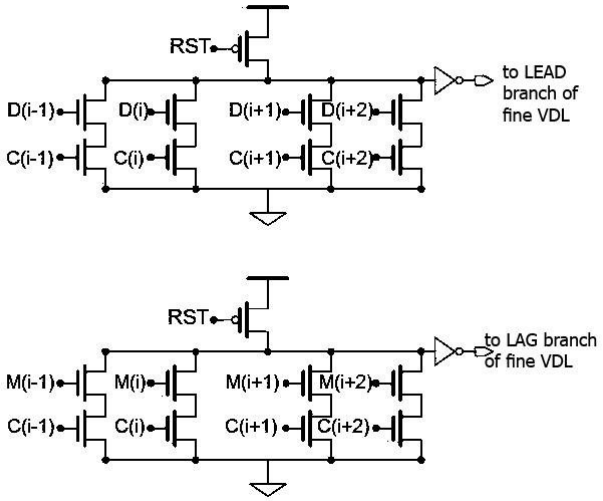


Figure 5.11: Interface circuit

From the above we can see that the signals C (the output of NORs) should turning on the branch of transistors in the interface circuit first, then when the delayed signals come to that branch the interface circuit passes it directly to the fine level. Theoretically the interface circuit can work perfectly according to that visualization, but in real world the signals C depend on the according delay signals that should be transferred to the fine level, but C should come first as we said!.

To can understand this problem let us take this example: Signal D(i) enters the D-

FLIP-FLOP then the output of the D-FLIP-FLOP will change according to signal D(i) in some pico-seconds, then the corresponding NOR output C(i) will be change too but not instantaneously!, it will take a few pico-seconds. So there is many pico-seconds between the signal D(i) and the signal C(i), the signal C(i) is the late one, but we said that it should come first to turn on the transistor! There is a problem here! , the signal D(i) we want to send to the fine level will pass before C(i) open the branch and will be lost!, what is the solution! The solution is to take a delayed version from D(i) to guarantee that the signal C(i) turned on the branch indeed.

We know it is not the required signal really, it is a delayed version, but we will delay the other signal with the required delay to keep the difference delay between them. Now take a look at Figure 5.12, you will see that signal D(i) is token after one stage of C(i), so the branch of the interface circuit will be ready to open and pass the signal ones it comes. The same thing is considered for signal M. But we should notice that we must delay both signals by the same delay to keep the difference between them to measure it by the FINE LEVEL.

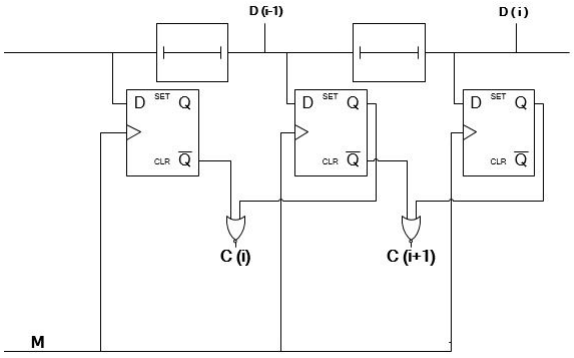


Figure 5.12: Coarse level after connecting NOR gates considering the delay problem

Chapter 6

Implementation of The proposed 8-bit TDC Using 2 Levels VDL

6.1 D-flip-flop:

The implementation of the D-flip-flop is shown in Figure 6.1. The D-flip-flop acts as an arbiter block, it gives an indication when the catch up occurs between the LEAD and LAG signals.

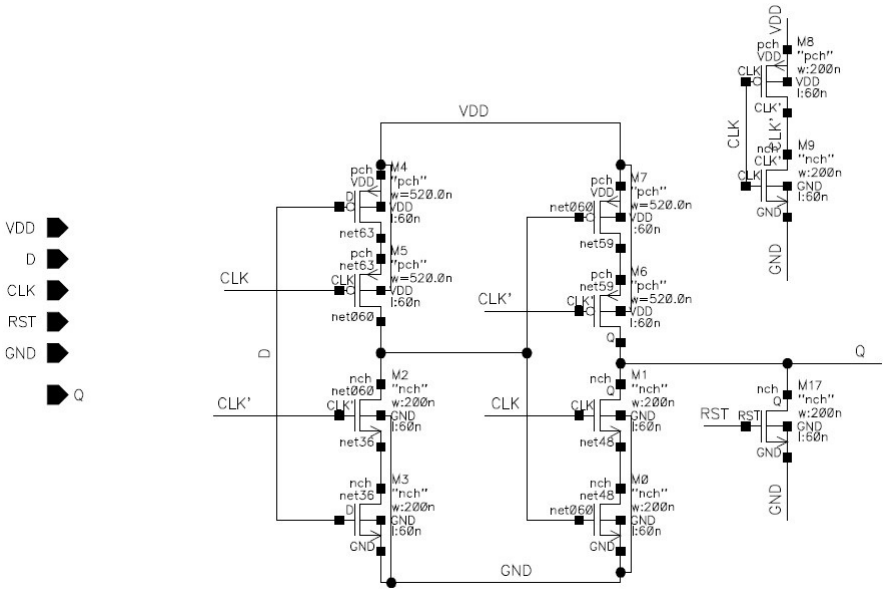


Figure 6.1: D-flip-flop implementation on cadence

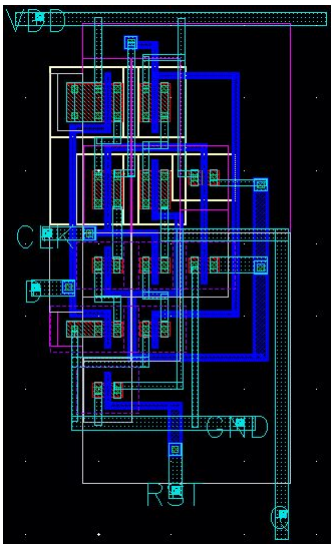


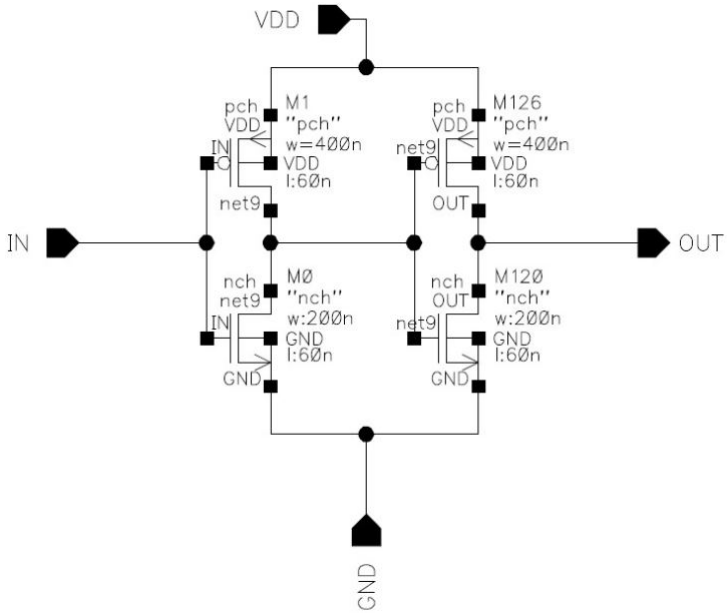
Figure 6.2: LAYOUT of D-Flip-Flop

One should take care of the setup time of D-flip-flop as if the two signals are with same delay (similar to each other) the output will be zero, this is due to the set-up time. This TDC take into consideration this problem by making the VTC introduce a delay to the LEAD signal equal to the set-up time.Figure 6.2 shows layout of D-flip-flop in Figure 6.1.

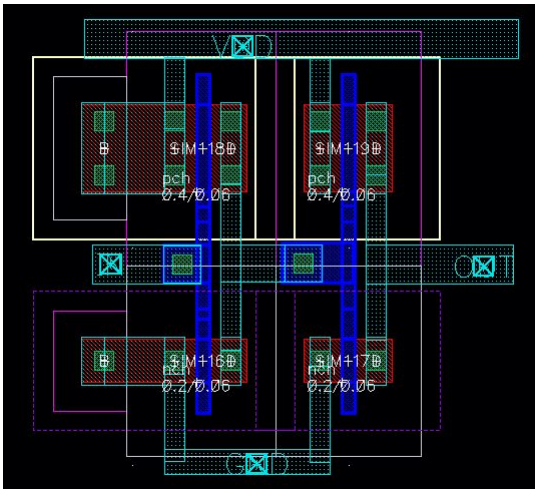
6.2 Delay element

The delay element is the most important block In the Vernier TDC as it defines the resolution. In both coarse level and fine level the resolution is defined by the difference between the delay of the upper line and that of the down line. So, the coarse resolution equal T_{H1}-T_{H2} and the fine resolution is equal to T_{F1}-T_{F2}.

The delay element is implemented using chain of buffers, Figure 6.3 shows the buffer (building block) of Delay element.



(a)

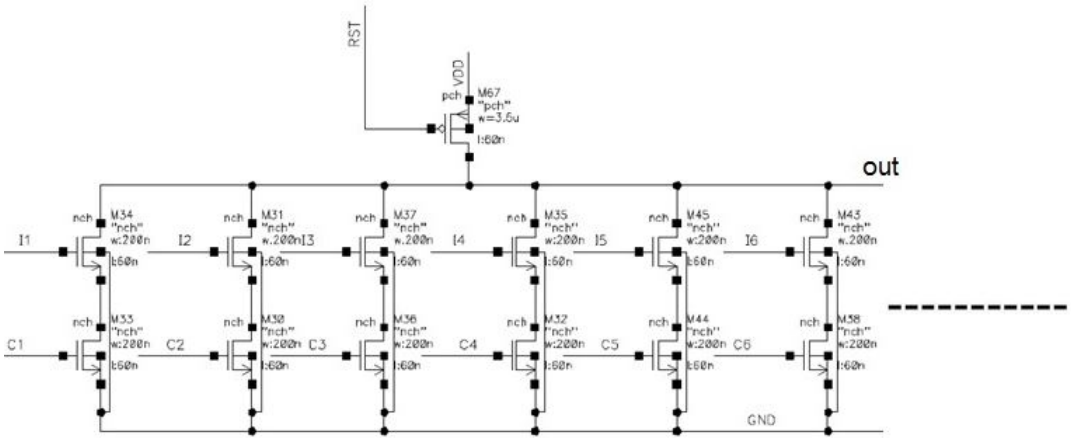


(b)

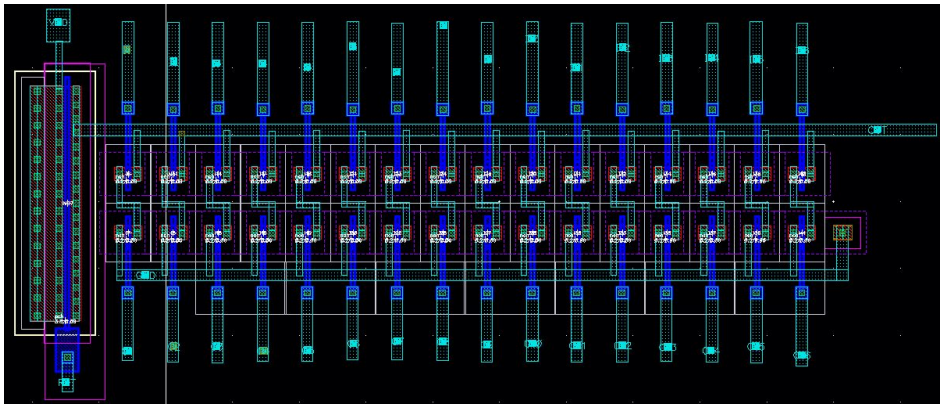
Figure 6.3: (a)Buffer the building block of Delay element (b)LAYOUT of one block of Delay element used

Chain of buffers can be used to achieve desired delay, for example we used a chain of 10 buffers to implement the upper delay element of coarse level (T_{H1}).

6.3 Interface Circuit



(a)



(b)

Figure 6.4: (a) The Interface circuit (b) LAYOUT of the interface circuit

The interface circuit is implemented using a NMOS as shown in Figure 6.4. The number of branches are equal the number of stages in the coarse level which is equal that of fine level.Interface circuit must be designed to have minimum delay so that it does not affect the total conversion time of TDC. its size is (13.84*5.045)

As stated before this technique of 2 level VDL needs 2 interface circuit one to transmit the LEAD signal and other to transmit the LAG signal. In fact interface circuit does not transmit a signal, the output of these two interface signals will be two pulses having the same delay as the LEAD and LAG signals at the moment of catch up.

6.4 Constructing coarse and fine levels

6.4.1 Building block

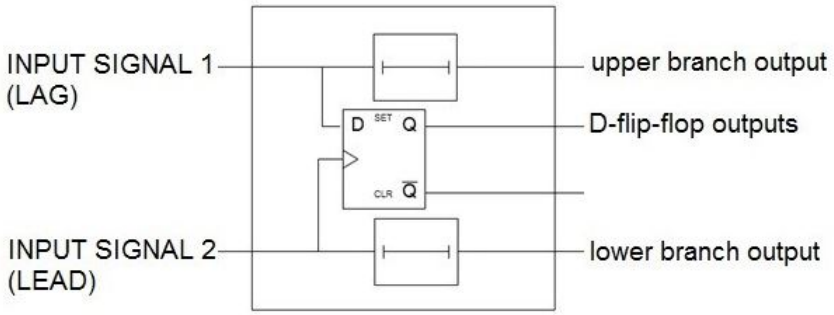


Figure 6.5: Building Block

This building block, in Figure 6.5, is the main block of coarse and fine levels, it is repeated to form our required number of bits, here we need 8-bits TDC so we need 256 levels. Due to using of 2 levels VDL we will use 16 blocks in coarse level and also the same number in fine level. [1].

6.4.2 Coarse level

Our VTC has dynamic range 291 mV in 1 nano-seconds, we divide this dynamic range to 256 level (8-bit), So by using two level VDL coarse will have 16 stages of building block and resolution of 1 nano-seconds/16 = 62.5 pico-seconds. So the difference between the two delay elements must equal 62.5 pico-seconds. Here in this design T_{H1}=125 pico-seconds, T_{H2}=62.5 pico-seconds.Figure 6.6 shows the layout of the building block of coarse level this building block is repeated 16 times to form the full coarse level.Its size is (21*6.7)

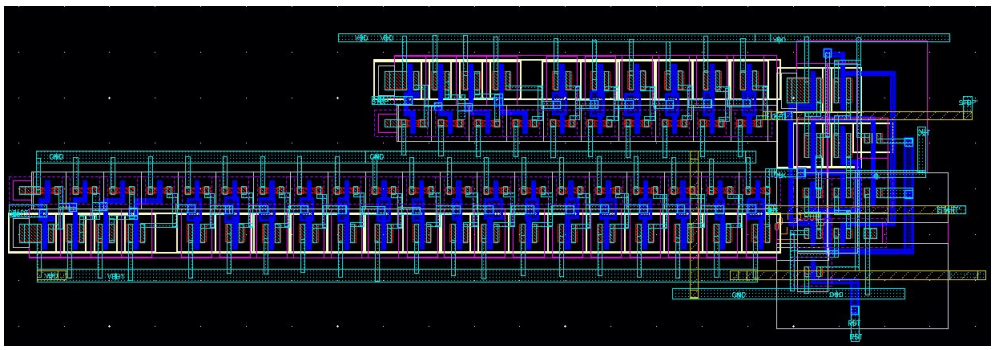


Figure 6.6: LAYOUT of building block of coarse level

[1]NOR gates will be connected to the D-flip-flops output, as stated in section, to be used in the interface circuit.

6.4.3 Fine level

In a same manner, the resolution of fine level will equal to 62.5/16=3.9 pico-seconds. Here in this design T_{F1}=10 pico-seconds, T_{F2}=13.9 pico-seconds.Figure 6.7 shows the layout building block of fine level this building block is repeated 16 times in row to form the full fine level of proposed TADC.Its size is (5.9*6.7)

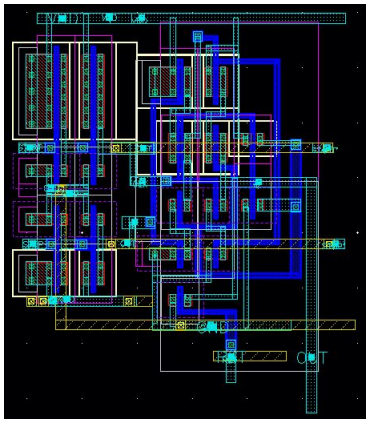


Figure 6.7: LAYOUT of building block of fine level

6.5 Structure of TDC

The final structure of TDC will be as shown in Figure 6.8. Here the inputs to TDC are the two output signals from VTC, where it is required to measure the delay between them and convert this time delay to digital output.

These two signals first enter the coarse level till a catch up occurs where the interface circuit begins to play its role by passing the signals at catch up to the fine level for more analysis to obtain better accuracy.

After that the outputs of the coarse and fine levels enter the read out circuit to give the digital output corresponding to the delay between the two outputs of VTC.

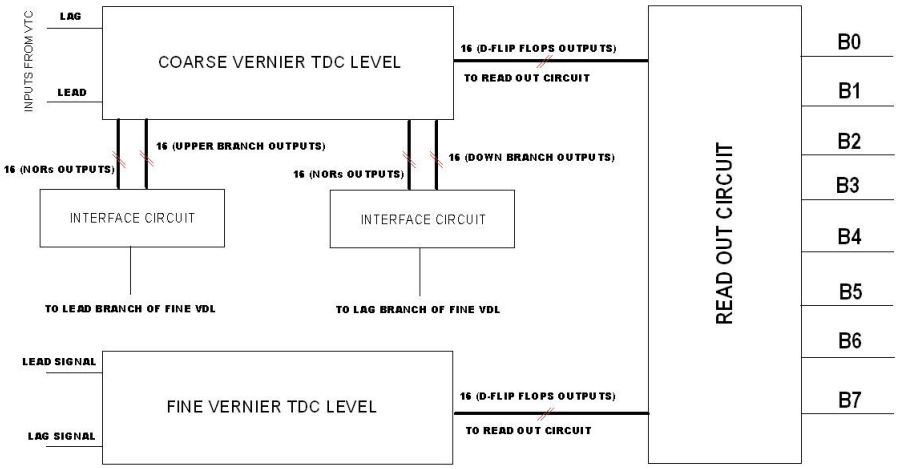


Figure 6.8: Final block diagram of TDC

6.6 Challenges and solutions

6.6.1 D-flip-flop

The main problem in the D FLIP-FLOP is the setup time[2]. We want the setup time to be zero, although this is realizable but it makes a big delay between the output of the D FLIP-FLOP and the input. This big delay will increase the problem we mentioned before, it will make even the delayed signal $D(i+1)$ reach the branch of interface circuit before the signal $C(i)$. So we want to compromise between the setup time and the delay between input and output of the D FLIP-FLOP. We reached a setup time equals 10 ps with input output delay equals 17 ps. But what will we do with this problem of setup time? Easily we entered the LEAD and LAG signal at the beginning with delay difference equals -10 ps not zero ps. That will make the D FLIP-FLOPs work correctly, but we should make up for this 10 ps delay before the signals enter the FINE LEVEL to can read the right delay and the right corresponding binary output.

6.6.2 Delay Element

Delay element is a circuit that outputs a delayed version of the input signal as we know. There are a lot of designs of delay element, some is programmable and others are not [21, 22]. The problem is that the same design and sizing of the delay elements will outputs different delays if the load changes, and of course the loads of the delay elements in the circuit will change from one to another especially at the edges. We solved this problem by making the delay element in the form of series buffers, which will not affect the previous delay element and make it change greatly. The problem still exists at the edges, so we used extra stages at the edges of both COARSE and FINE LEVELS.

6.6.3 Interface Circuit

We start our coarse design using the one chain delay element (simple flash TDC). This design is preferred when we don't need high resolution because its resolution is the delay of the delay element as we said before. So we begin the design with using it in the coarse and using the Vernier design in the fine according to the need of 3.9 ps resolution. But due to having just one chain of delay elements (the upper one) in the coarse, many errors arose in the circuit, affected the performance greatly, caused the circuit acting wrong. The errors can be summarized in the following:

- FAN-OUT ERROR
- CURRENT LEAKAGE ERROR

[2]Setup time is the minimum amount of time the data signal should be held steady before the clock event so that the data are reliably sampled by the clock. This applies to synchronous input signals to the flip-flop.

6.6.3.1 Fan-Out Error

The CMID signal goes to the sixteen transistors in the interface circuit as shown in figures 5.11 and 5.12. Due to large number of fan-out, the signal becomes so sluggish as shown in Figure 6.9. It takes about 100 pico-seconds to turn from low to high unlike other signals that take about just 10 pico-seconds. In figure 6.9, despite both of “in”, ”CMID” signals are beginning at the same time (5ns), but there is a difference about 72 pico-seconds between them. So this is a big problem that affects the circuit and make TDC can’t read the difference between the two signals correctly then can’t out the correct 8-bit binary output.

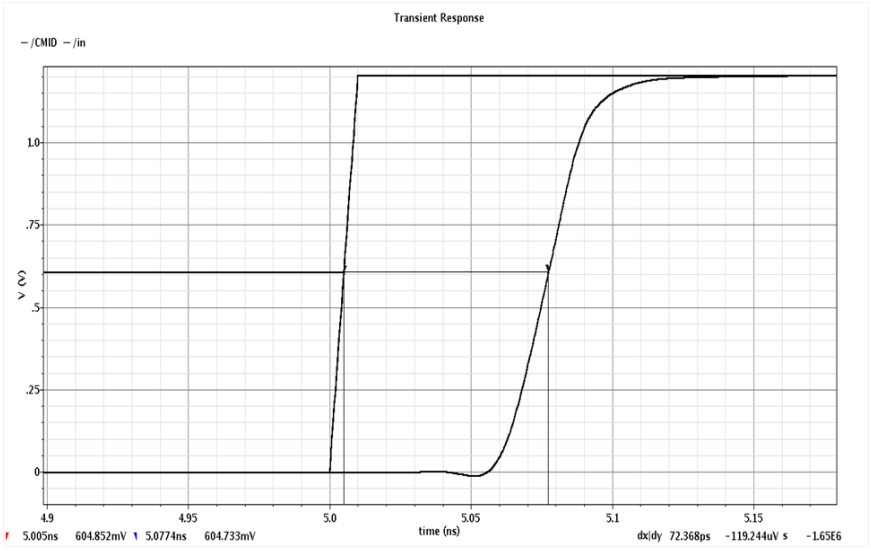


Figure 6.9: Fan-out problem

6.6.3.2 Current Leakage Error

From our understanding to how the interface circuit work and how it transmits the difference between the signals, we know that delayed copies are entered the transistors of the interface circuit, for example for the upper chain in the figure 5.12 the signal $D(i-1)$ enters the left transistor, $D(i)$ enters the one after it, and $D(i+1)$ enters the right transistor on the figure.

But for the lower chain we can find that the signal M enters the sixteen transistors in the same time when it just comes, so the amount of delay due to leakage current in the transistors will vary with no constrains and will be variable between the interface circuit that out the LEAD SIGNAL and the interface circuit that out the LAG SIGNAL. This problem will lead to a big error which is the difference between the LEAD and LAG signals will not equal to the right number or the right amount of delay, but will still some error in the range of 15 pico seconds which will cause wrong reading with more than two counts, we can see that in figure 6.10.

This figure is token when the signal $C(13)$ (the output of NOR) is high, then the signals entering the interface circuit and wanted to be out are CMID1-i15, and the resulting signals from the interface circuit are START- STOP. START- STOP should equals CMID1-i15, but we can see here that CMID1-i15=10.88 ps, and START- STOP=24.46 ps, which means there is an error equals about 14 ps, that error is due to the leakage current which results in from the one chain of delays design.

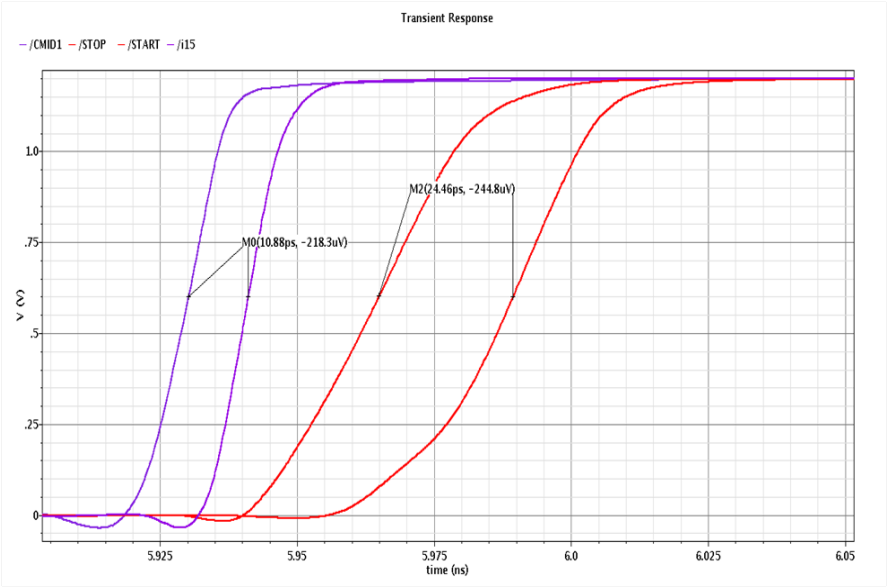


Figure 6.10: Current leakage problem

But using the vernier design (two chains of delay elements) solved these errors,

- It is obvious that it solves the fan-out problem as there will not be just a wire who feeds a lot of inputs
- It solved the other problem too by making both interface signals generating the lead and lag signals vary in the same manner, so the difference between them will not be affected and will be the right delay we want.

6.7 Read Out Circuit

For now, we have got the results of both coarse and fine levels in the form of thermometer code, the READ OUT circuit is aiming to calculate the 8-bit binary output from these thermal codes. Based on our understanding of the two level vernier delay line design we know that every count (every delay element)in the coarse level represents sixteen counts in binary, and the fine level is the extra counts (the amount of excess delay) need to be subtracted from the coarse to give the right output binary number. So we can conclude

the equation represents the READ OUT circuit as follow

$$BINARY O/P = 16 * COARSE-FINE \tag{6.1}$$

Both of COARSE and FINE in binary. We can see the READ OUT circuit block diagram in the Figure 6.11

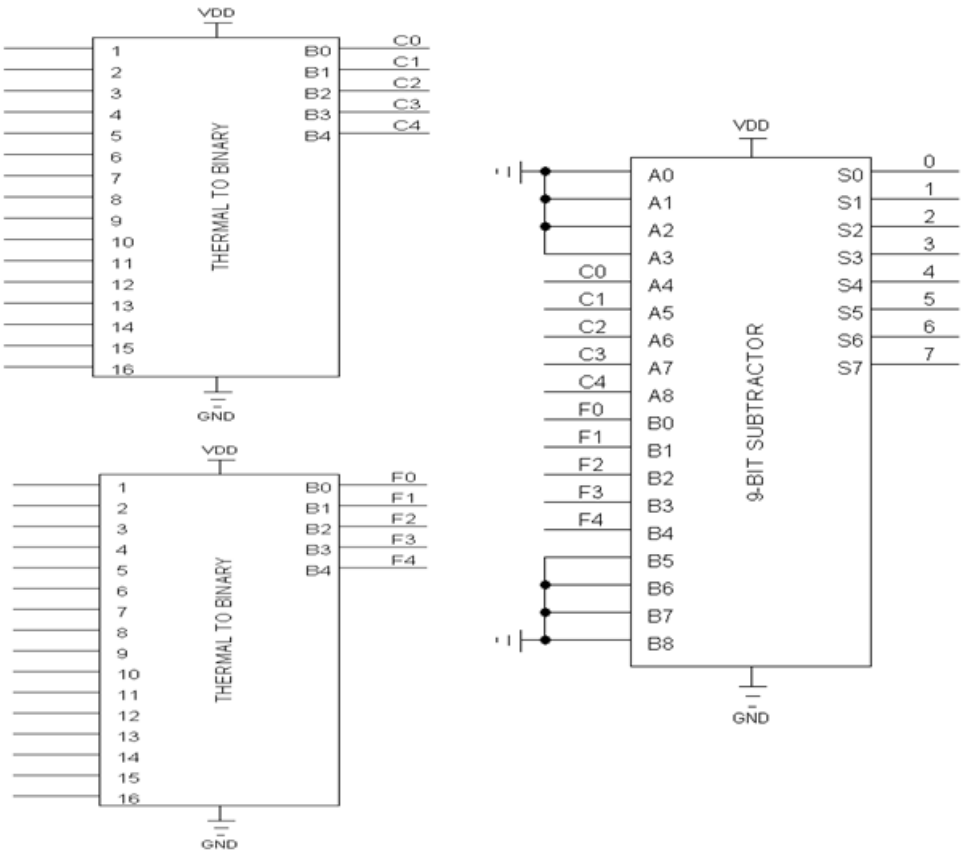


Figure 6.11: Read-out circuit

6.7.1 Thermometer to Binary Encoder

There are many common designs for this purpose and many researches as well, Three of them were the most attractive for us,

- ROM-BASED THERMAL TO BINARY ENCODER
- FAT-TREE THERMAL TO BINARY ENCODER
- MUX-based THERMAL TO BINARY.

Most of these circuits work in the same flow, converting the thermal code to one hot code then to the binary, Table 6.1 and Figure 6.12 shows the flow from ROM-BASED and the same thing for FAT-TREE.[23]

Thermometer code	one-hot code	Binary code
0000000	0000000	000
0000001	0000001	001
0000011	0000010	010
0000111	0000100	011
0001111	0001000	100
0011111	0010000	101
0111111	0100000	110
1111111	1000000	111

Table 6.1: TM2B flow table

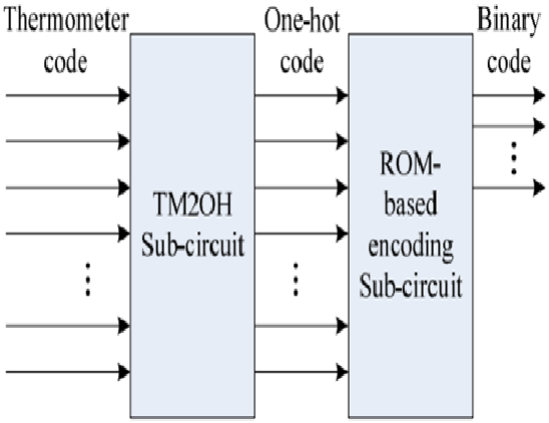


Figure 6.12: TM2B flow schematic

6.7.1.1 Rom-Based TM2B Encoder

It consists of the two parts we mentioned, the way it works on is so easy. The sub-circuit TM2OH is implemented by 2-input AND gate with one input is inverted. Figure 6.13 shows a TM2OH sub-circuit of a 7-to-3 ROM-based TM2B encoder. outputs can be determined by the formula:

$$OUT_i = IN_i.(IN_{i+1}) \tag{6.2}$$

Where OUT_i is the output at the position$i$ and IN_i,IN_{i+1} and i,$i+1$are the input at the position respectively.. That is Except the highest significant output .[23]

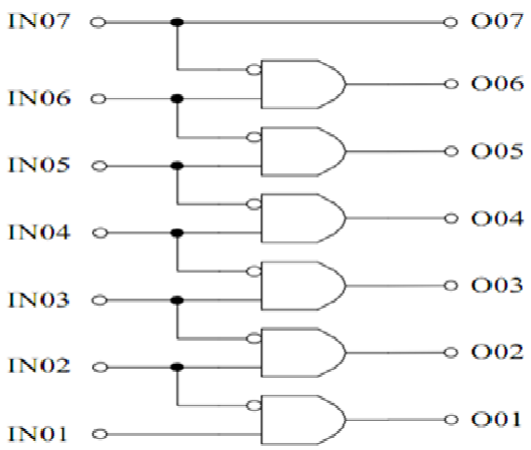


Figure 6.13: TM2OH sub-circuit of a 7-to-3 ROM-based TM2B encoder

The second part IS ROM-based encoding sub-circuit, it is a ROM-structure circuit that doesn't contain address decoding circuit. The content of ROM is directly the binary value output of the TM2B circuit. It does not need the address decoding circuit it can be used to access ROM structure directly because its input is a one-hot code. Figure 6.14 shows the structure of a 7-to-3 ROM-based TM2B encoder as we explained. We can see also the meaning of the thermometer code on figure.

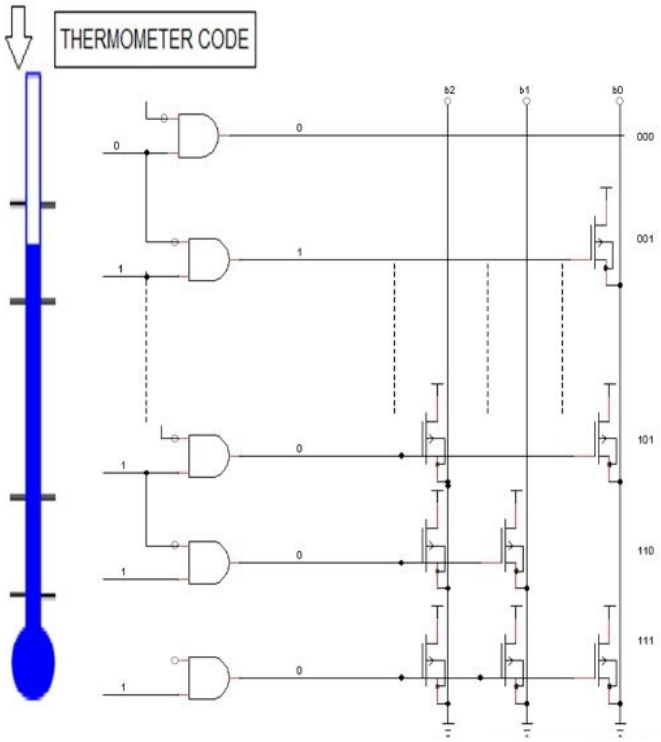


Figure 6.14: A 7-to-3 ROM-based TM2B encoder

6.7.1.2 Fat-Tree TM2B Encoder

It consists of the two parts too, the first sub-circuit is similar to the ROM-BASED design And the second one is a logic circuit. With an $2^n - 1$inputs, each output is an OR operation of $2^n - 1$inputs. The formulas that convert one-hot code to binary code are shown at equations 17,18, and 19.

$$b_0 = h_1 + h_3 + h_5 + h_7 \tag{6.3}$$

$$b_1 = h_2 + h_3 + h_6 + h_7 \tag{6.4}$$

$$b_2 = h_4 + h_5 + h_6 + h_7 \tag{6.5}$$

Where b_0,b_1 ,b_2 are the outputs in binary and are the inputs in form of thermometer code.

Reusing some sub-circuits, the number of OR gates are reduced. Figure 6.15 is the fat-tree encoding sub-circuit of a 7-to-3 fat-tree TM2B encoder. The OR result of input $h_6$and inputh_7 is reused to generate output b_1 This structure can be easy to extend to higher number of inputs.

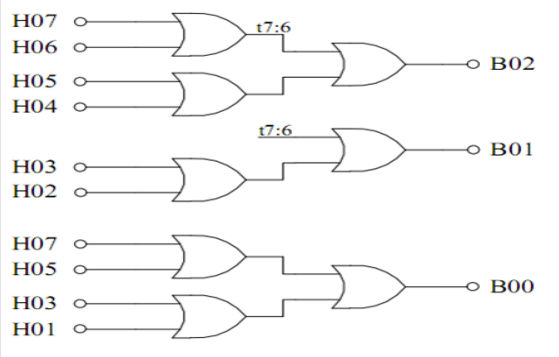


Figure 6.15: A 7-to-3 fat-tree OH2B encoder

6.7.1.3 MUX-based TM2B Encoder

MUX-based TM2B encoder converts thermometer code to binary code directly at one conversion. The encoder uses multiplexers as basic gate to implement converting function. A 7-to-3 MUX-based TM2B encoder is shown in Figure 6.16. As shown in the figure, the circuit has a high structure. Hence, the circuit can extend to higher input encoder easily [23].

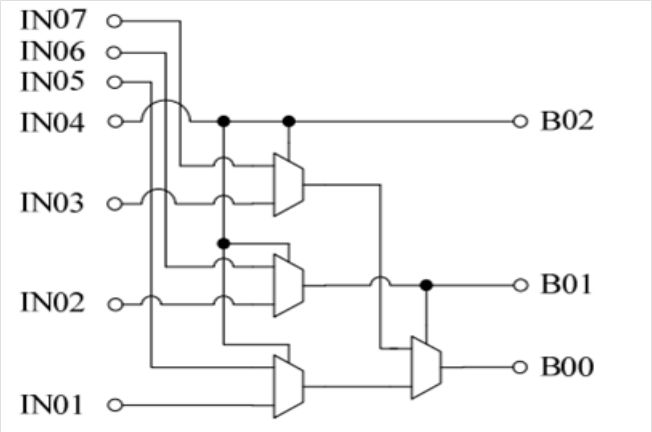


Figure 6.16: A 7-to-3 MUX-based TM2B [23]

6.7.2 Comparison TM2B encoder circuits [23]

To compare the complexity of the circuits, three different 63-to-6 TM2B with BEC [3] encoders are designed in three methods, ROM-based encoder, fat-tree encoder, and MUX-based encoder.

Circuits are designed with Cadence tool. The library was used is TSMC25. The number of requirement transistors of these three methods is shown in TABLE 6.2.a.

As shown in the table, fat-tree encoder requires the highest number of transistors.

The MUX-BASED circuit requires fewer transistors than fat-tree circuit but little more transistors than ROM-based circuit.

To compare the power consumption, three TM2B circuits were simulated with Cadence tool. Input generator generates value for 63 inputs of tested 63-to-6 TM2B circuits. The input values are change from the maximum value to the minimum value.Binary value outputs of TM2B circuits are measured to verify the correction of circuits. Power consumption is monitored in simulating process and reported at the end of the simulating process. Average power consumption of three circuits operating at 500 MHz is shown in Table 6.2.b. As shown in the table, ROM-base encoder circuit consumes most power, whereas fat-tree circuit consumes least power. The MUX-BASED circuit requires only a little more power than fat-tree circuit. And now here is a comparison between the previous techniques from the point of view both of area and power consumption

[3]BIT ERROR CORRECTION: it's an additional circuit to detect the bubbles error, for example: for the thermometer code 00011110111 there is an error, it's the bubble (zero) between the ones, BEC circuit corrects this error to be 00011111111

Circuits	Transistor requirement
ROM-based + BEC circuit	714
Fat-tree + BEC circuit	832
MUX-based + BEC circuit	722

(a)

Circuits	Average power consumption (mW)
ROM-based + BEC circuit	1.831
Fat-tree + BEC circuit	0.942
MUX-based + BEC circuit	1.211

(b)

Table 6.2: comparison between TM2B encoders [23]

6.7.3 Implementation of TM2B Circuit Used In TADC Proposed

We used the FAT-TREE ENCODER design because it consumes the minimum power. We extended the design to make it 16 to 5 TM2B, the TM2OH sub-circuit is just as we explained before, but for 16 inputs now as we can see figure 5.7.7. The other part of the design is extended too to achieve 16 to 4 TM2B encoder2 (actually we made 16 to 5 TM2B encoder but the fifth bit B4 was token directly from H16 because we will just need it in one case, the case of the sixteen count which the binary output will be 10000, so we can take it directly as we did) The binary outputs can be determined from the equations 20, 21, 22, 23 and 24

$$B_0 = H_1 + H_3 + H_5 + H_7 + H_9 + H_{11} + H_{13} + H_{15} \tag{6.6}$$

$$B_1 = H_2 + H_3 + H_6 + H_7 + H_{10} + H_{11} + H_{14} + H_{15} \tag{6.7}$$

$$B_2 = H_4 + H_5 + H_6 + H_7 + H_{12} + H_{13} + H_{14} + H_{15} \tag{6.8}$$

$$B_3 = H_8 + H_9 + H_{10} + H_{11} + H_{12} + H_{13} + H_{14} + H_{15} \tag{6.9}$$

$$B_4 = H_4 \tag{6.10}$$

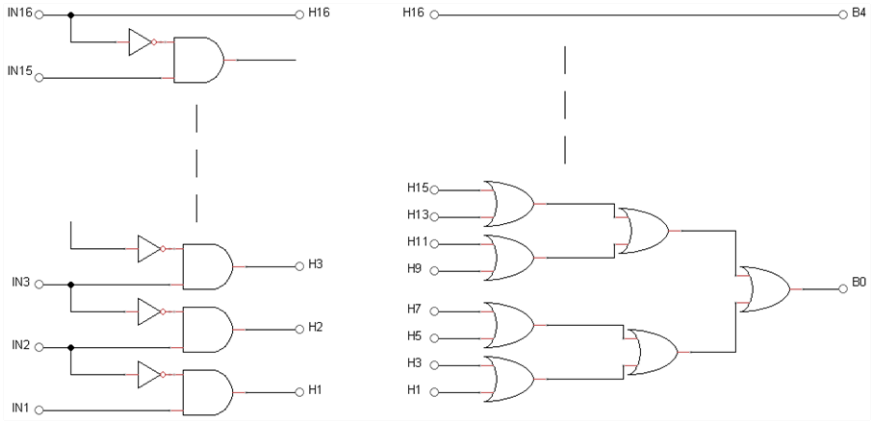


Figure 6.17: PROPOSED TH2B

Subtractor

After we got the binary of the coarse and fine, we will need a multiplier and a subtractor to compute the equation

$$BINARYO/P = 16 * COARSE–FINE. \tag{6.11}$$

. We don't need a general multiplier but only a one which can multiply by sixteen. Actually it is just a four bit shift!. So the binary of the first term in the equation will be nine bits, hence we will just need a 9-bit subtractor. We used the common architecture of subtractors, nine full adders are used as we can see in Figure 6.18. The first term of the equation is the binary $A8A7....A1A0$, the second term is $B8B7.....B1B0$. the second term is inverted before it enter the full adder, each carry is entered the next full adder, and the first full adder we enter a "1" as we know from basics of digital . The output will be a 8-bit binary number $(s8s7.....s1s0)$.

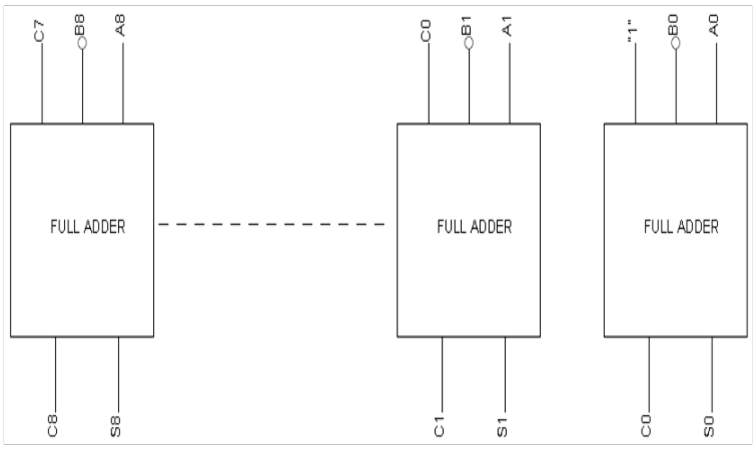


Figure 6.18: Proposed 9-bit subtractor design

We can see also in Figure 6.19 the design of the full adder which consist of two half adders and an OR gate. The design of the half adder too is shown in Figure 6.20 and

consists of a XOR and an AND gate.

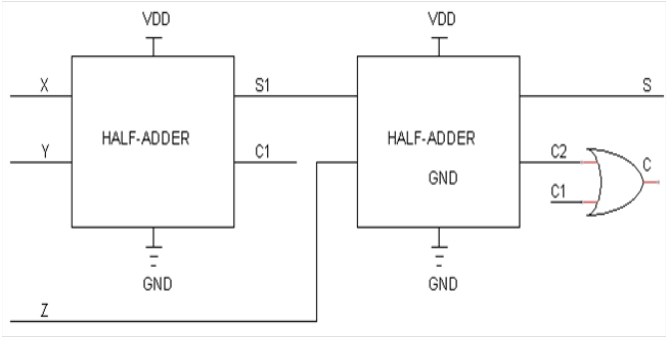


Figure 6.19: Full adder design

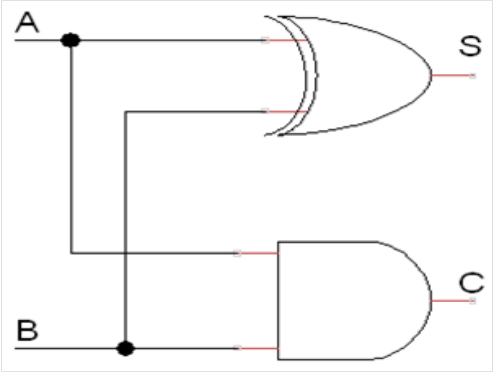


Figure 6.20: Half adder design

Chapter 7

Measuring Performance of TADC & Simulation Results of the Proposed TADC

7.1 Effective Number Of Bits (ENOB)

one important property of any converter including ADCs, VTCs and TDCs is Linearity. There are different metrics for quantifying linearity, including differential non-linearity (DNL), integral non-linearity (INL), total harmonic distortion (THD), and signal to noise and distortion ratio (SNR).

One of the important metrics for measuring the performance of a TADC is the signal to noise and distortion ratio (SNR) which is frequently described in terms of effective number of bits (ENOB). The value of ENOB describes the overall accuracy of the converter.

ENOB is the most commonly used for data converters, due to its intuitive nature. For example, if one is told that a 4-bit ADC has an ENOB of 3.2 bits, this is much easier to grasp than being told that the SNR is 21 dB (even though the two metrics are exactly equivalent)[9]. The common formula for calculating the ENOB is

$$ENOB = \frac{SNR - 1.76}{6.02} \tag{7.1}$$

where SNR has units of dB and ENOB is measured in bits.

There is a common technique for measuring ENOB: it is the fast Fourier transform (FFT) method. In the FFT method, a data record of a specified length is recorded from the output of the converter. The record can include all samples, or every Mth sample (decimation). For best results, the test must be arranged so that an exact integer number of input cycles occur during the test period - this is known as coherent sampling. In other words, the following should be true: $f_0 = (n/c)f_s$ where f_0 is the signal frequency,f_s is

the sampling frequency, n is the total number of samples in the record, and c is the integer number of cycles in the record. Any integer can be used for c in order to test different input frequencies.

Once the record has been taken, an FFT is performed on the data to obtain the frequency response. The SNR can then be calculated directly from the FFT data, and the ENOB is calculated using the standard formula[1]. Another method that has been used will be seen in Appendix **B** by cadence and matlab.

7.2 Simulation results of proposed TADC

After we connected VTC with TDC we made simulations on many distinct dc values to can see the linearity of the overall TADC.Table 7.1 shows 31 inputs in mv with constant step between every input equals to 9.7mv and the corresponding output in binary and decimal. The fourth column shows the difference between outputs for every two successive steps.

V_{in}(mVolt)	Digital Output	Decimal Output	Diff (current - next)
384	0000 0000	0	8
393.7	0000 1000	8	9
403.4	0001 0001	17	8
413.1	0001 1001	25	9
422.8	0010 0010	34	8
432.5	0010 1010	42	8
442.2	0011 0010	50	8
451.9	0011 1010	58	9
461.6	0100 0011	67	8
471.3	0100 1011	75	8
481	0101 0011	83	8
490.7	0101 1011	91	8
500.4	0110 0100	100	8
510.1	0110 1100	108	9
519.8	0111 0101	117	8
529.5	0111 1101	125	9
539.2	1000 0110	134	8
548.9	1000 1110	142	9
558.6	1001 0111	151	8
568.3	1001 1111	159	9
578	1010 1000	168	9
587.7	1011 0001	177	9
597.4	1011 1010	186	9
607.1	1100 0011	195	8
616.8	1100 1011	203	9
626.5	1101 0100	212	9
636.2	1101 1101	221	9
645.9	1110 0110	230	8
655.6	1110 1110	238	9
665.3	1111 0111	247	8
675	1111 1111	255	

Table 7.1: Result for 31 different input DC volt and obtained digital code

We can see that for constant input voltage steps, the corresponding output steps are almost constant, which means the TADC achieves high linearity. We can see the plot of the input voltage to the output digital at Figure 7.1.

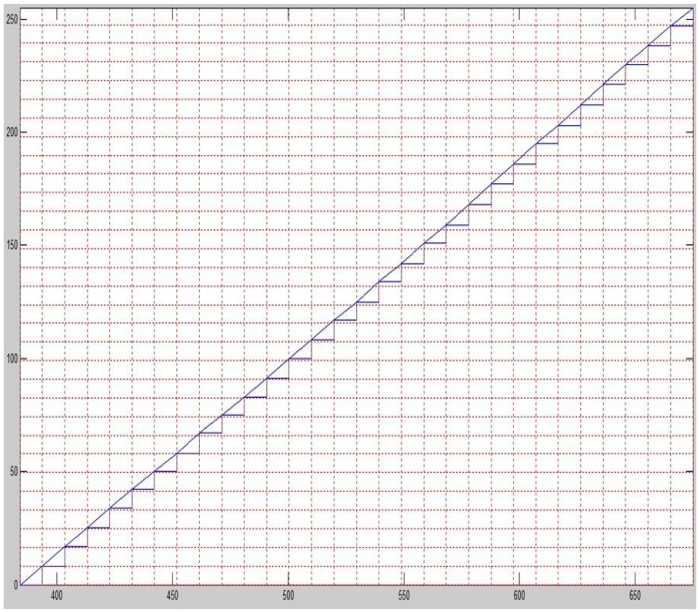


Figure 7.1: Plot diagram of the obtained results

We can see too the high linearity which will results a high effective number of bits. Now to calculate the ENOB we will enter a sine-wave with frequency 30 MHZ over a period of cycle, we will use cadence and matlab to calculate the ENOB as we will see that in Appendix B in detail, but we can see in figure 7.2 the input signal, the signal after sample and hold and the reconstructed signal. We can see that the reconstructed signal is so close to the signal after sample and hold at time we will sample1 (to redraw the signal we will sample at points in the reconstructed signal to know the value of the signal at this cycle) Finally after doing all the steps , we calculated the ENOB and got ENOB of 7.6 bits which is a good result.

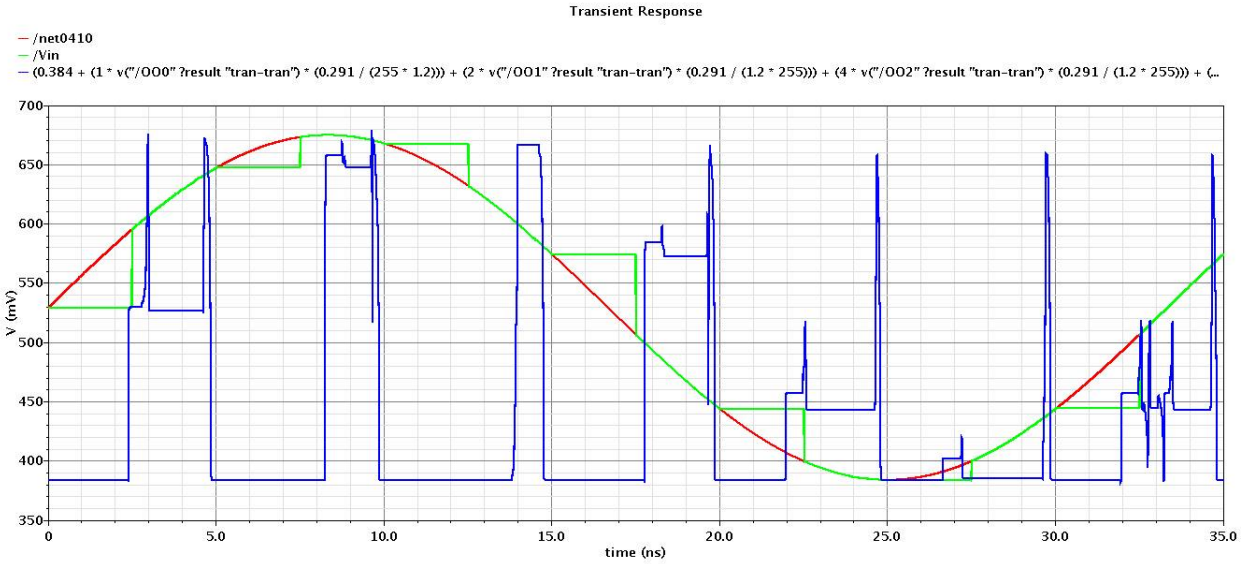


Figure 7.2: Input signal,sampled Vin and digital output after DAC equation

7.3 Layout of the basic elements of Read Out circuit

7.3.1 Nand Layout

Area of NAND = 1.88*2.1 $(um)^2$

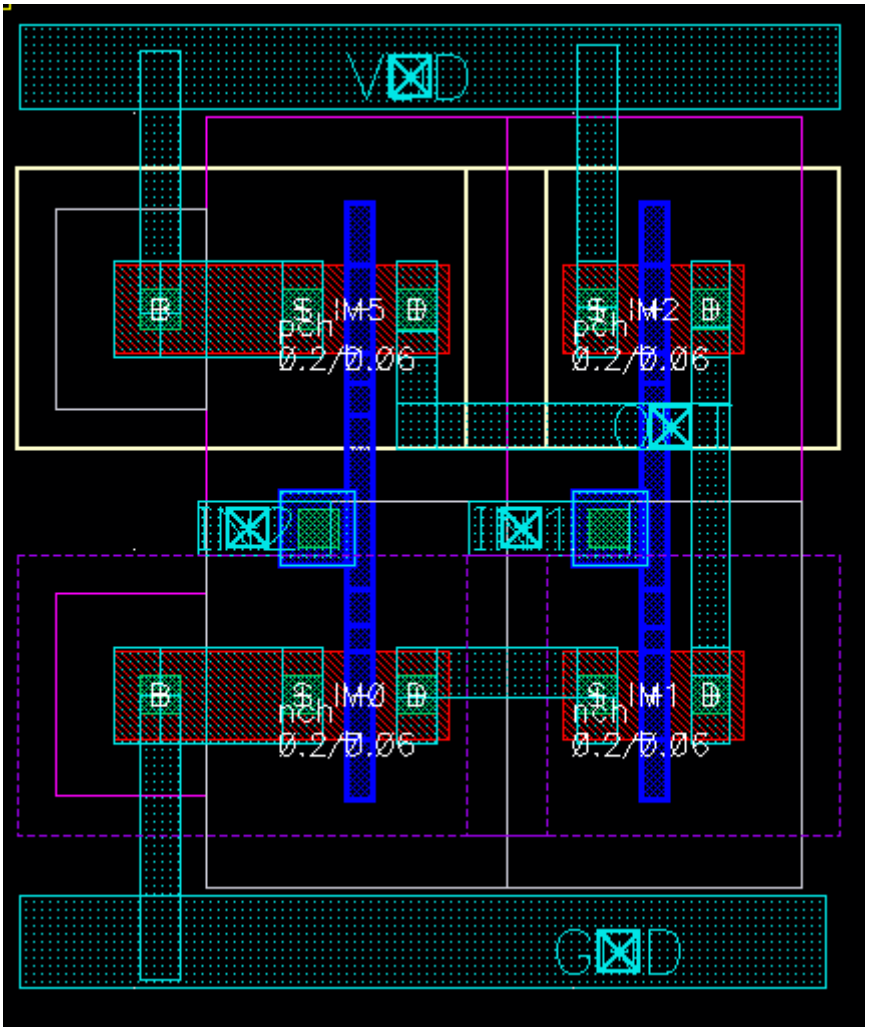


Figure 7.3: NAND layout

7.3.2 NOR

Area of NOR = 1.88*2.1 $(um)^2$too.

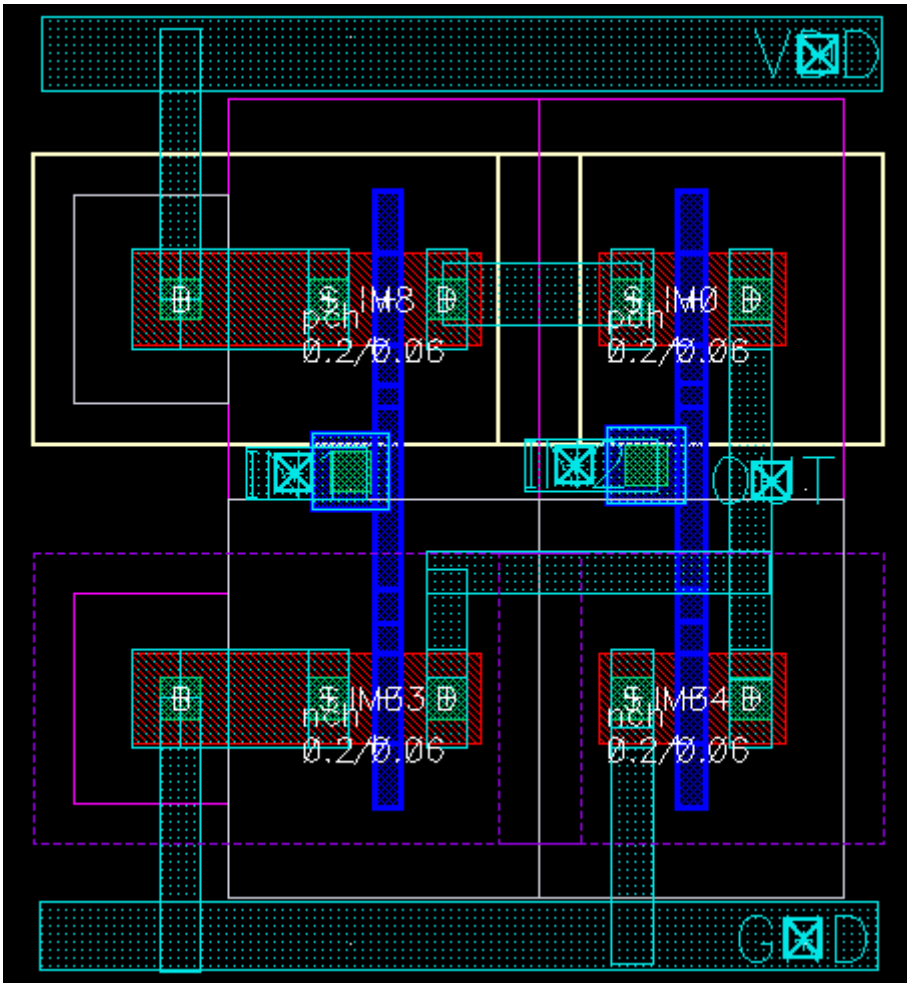


Figure 7.4: NOR layout

The read out circuit is implemented using repeated copies of these two basic cells.

Chapter 8

Conclusions

The time based ADC employs a completely different architecture from the conventional ADC and quantizes time at predefined amplitude intervals.

An 8-bits 200 MSPS time-based ADC with its two main blocks is proposed. A highly linear VTC of dynamic range of 291mv with linearity error of 1-3% and high sensitivity of 3.43ps/mV is reached. This VTC is referred to be PPM.

The TDC is based on the two levels Vernier delay line was introduced, a resolution of 3.9ps has been achieved. This 2 level VDL reduces the required number of delay elements and D-flip-flops.

The two level VDL outputs are a two thermometer codes (one for the coarse and one for the fine) which converted to a binary output using the proposed READ OUT circuit over two stages, the first one is done by converting the thermometer code to a binary code using the fat tree design, and the second stage we used a subtractor to get the binary output.

The ENOB of 7.6 bits for input sine-wave frequency equals 30MHZ, which represents an acceptable signal to noise (quantization noise) ratio.

Appendix A

Layout tutorial using cadence

This appendix shows how to make layout after finishing schematic (circuit design).

If we want to make a layout for the following schematic

Steps:

1. Select “Layout XL” from “Launch menu”

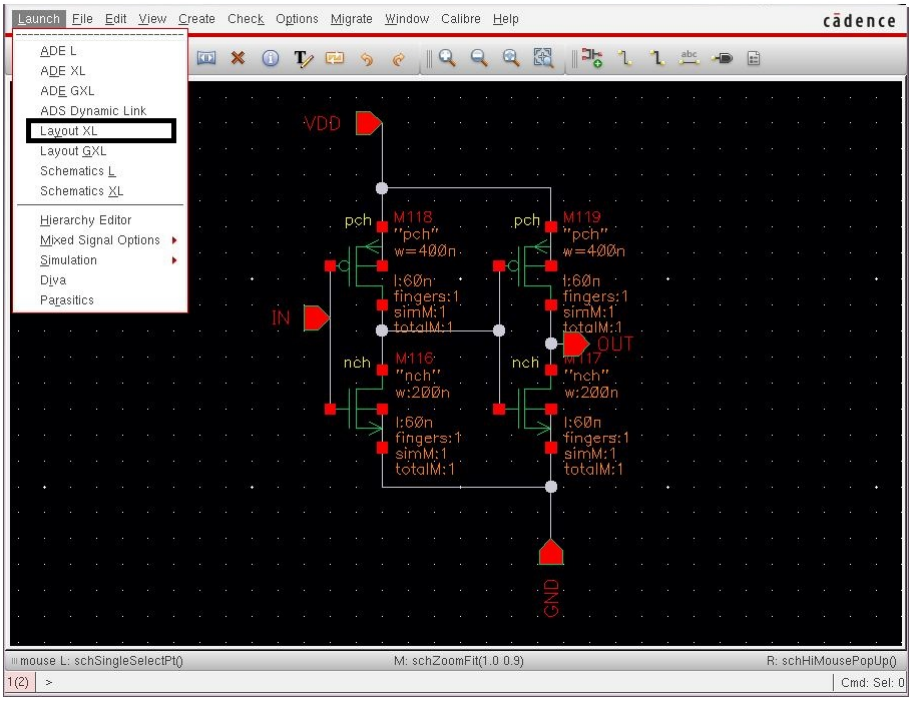


Figure A.1: LAYOUT Step 1

2. From the “Startup option” window select “create new” and set configuration to be “Automatic”

Figure A.2: LAYOUT Step 2

3. Dialogue box appear asking if you want to create a layout cell view in this library we press ”OK”

4. New empty window appears where we can start our layout

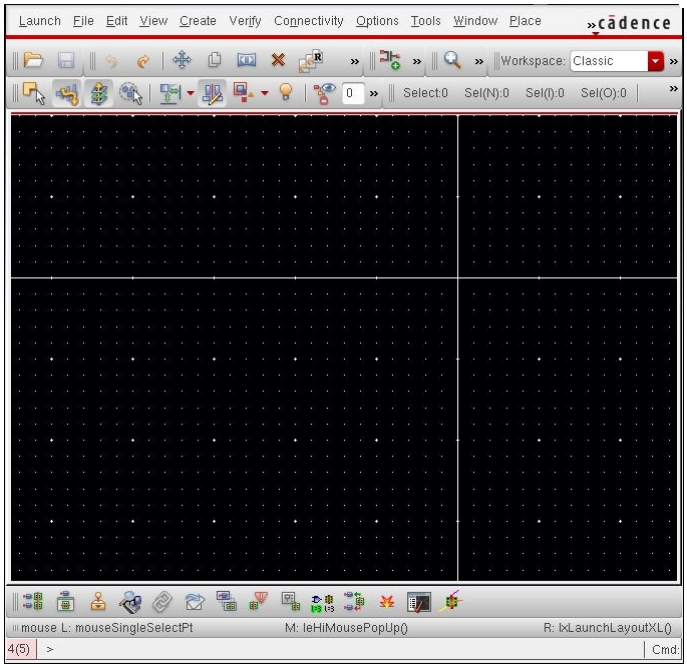


Figure A.3: LAYOUT Step 4

5. Also a window called “LSW” appears, this window contain all layers that we will use to make layout.

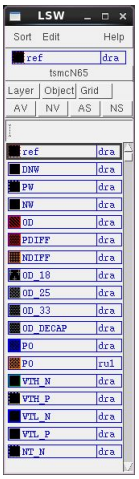


Figure A.4: LAYOUT Step 5

6. In cadence there is a feature which we can use to make cadence automatically generate all transistors layout from schematic view also I/O pins. To use this feature we select “All from source” from “Connectivity–>Generate” menu

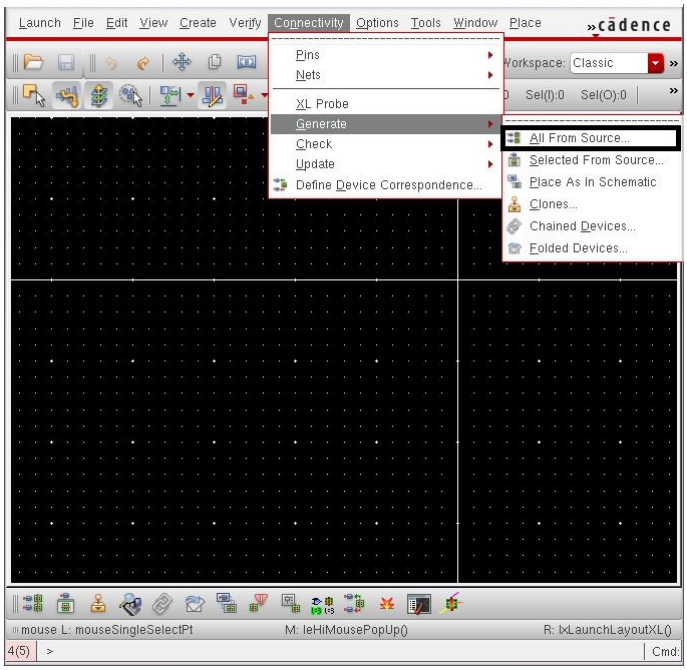


Figure A.5: LAYOUT Step 6

7. Here we select what we need to generate so we select “Instances” and “I/O pins”

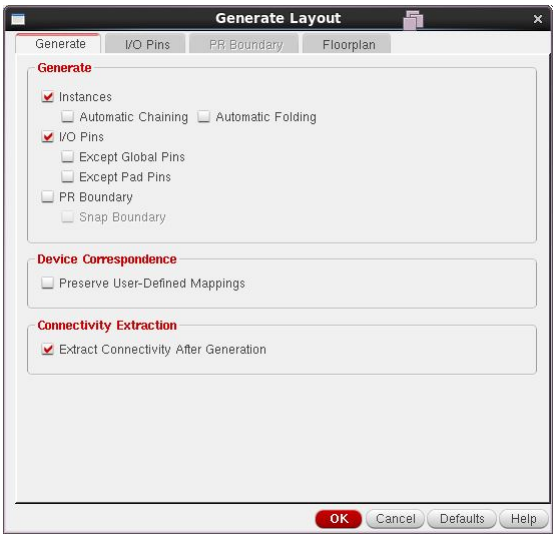


Figure A.6: LAYOUT Step 7

8. From “I/O pins” tab we select pins to be created on “Metal 1” as shown in figure. Also we must select “Create Label AS–> Label” to create labels on pins to be able to distinguish different pins.

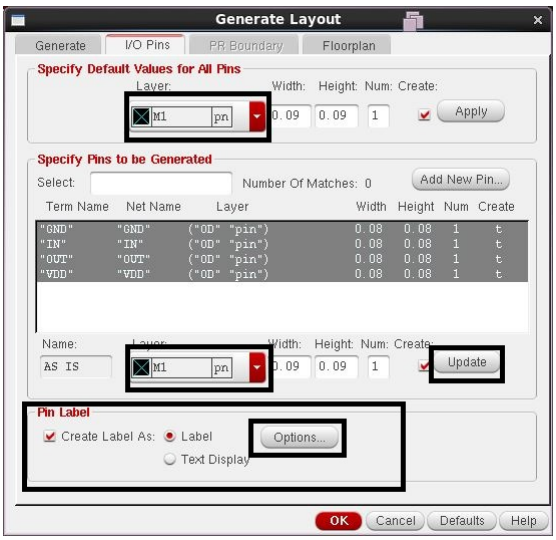


Figure A.7: LAYOUT Step8

9. From label options we enter a height of “0.1” and make the layer of pin as “text”

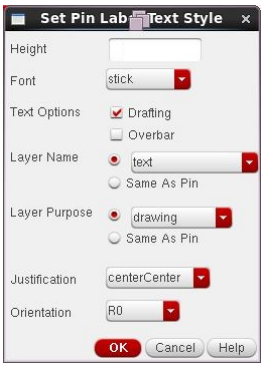


Figure A.8: LAYOUT Step 9

10. After pressing OK this what will appear to us, it generates all transistors in schematic and all pins

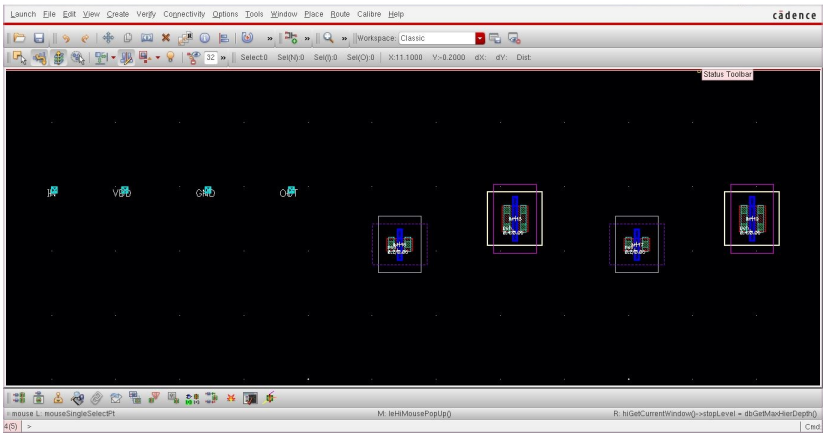


Figure A.9: LAYOUT Step 10

11. All we have to do is to begin to connect our transistor as in schematic but we will a problem during dragging any instance as mouse is set to move with constant snap spacing, To modify this we press ”e” to open

“Display Option” window and make spacing as shown in figure.

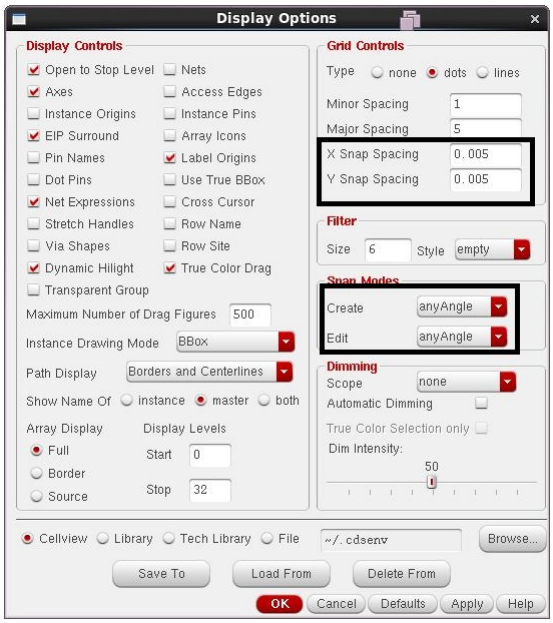


Figure A.10: LAYOUT Step 11

12. Then we drag instance and align them as shown in figure[1]

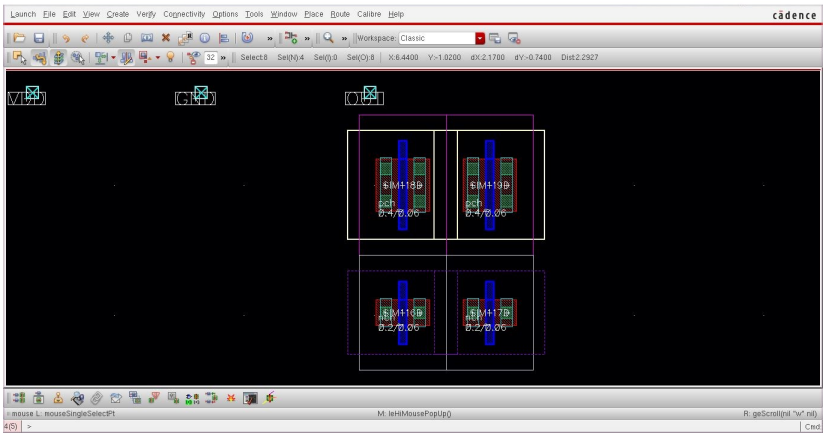


Figure A.11: LAYOUT Step 12

13. To connect bulk terminal we select one transistor only in each well and press “q” and select from parameter “bodytie_typeL–> Integrated”.[2]

[1] take in consideration the design rules

[2] bodytie_typeL integrated connects the Bulk to source

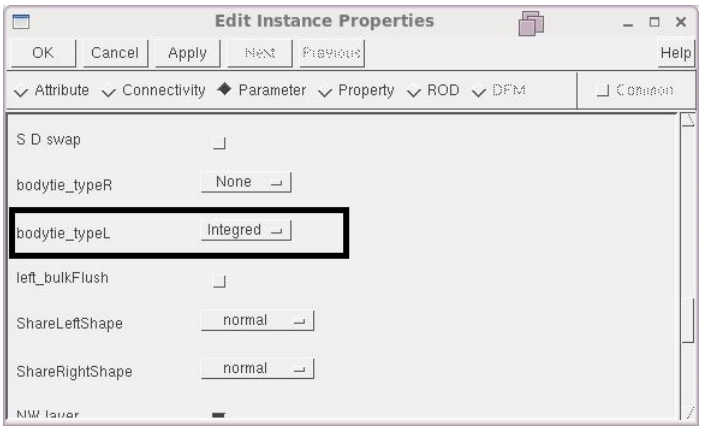


Figure A.12: LAYOUT step 13

14. After that we make connections as in schematic with the help of “LSW” window, by selecting desired layer and pressing “r” to rectangle plot any layer.

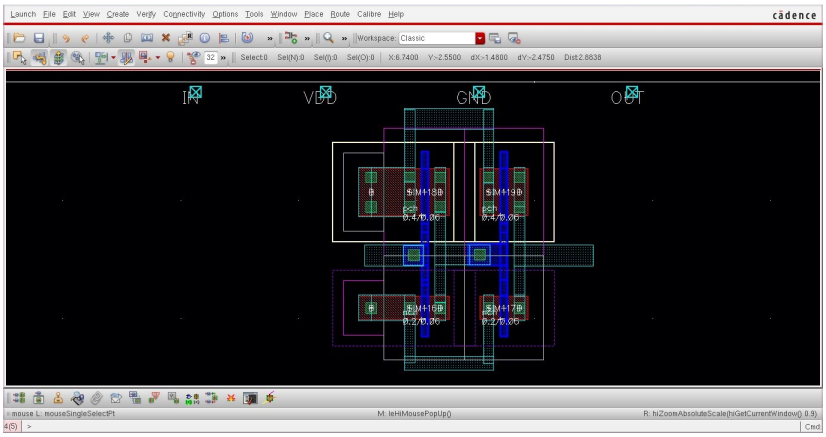


Figure A.13: LAYOUT Step 14

15. Then we select all pins, press ”q”, select “Common” and make label layer “Metal 1” as shown in figure.

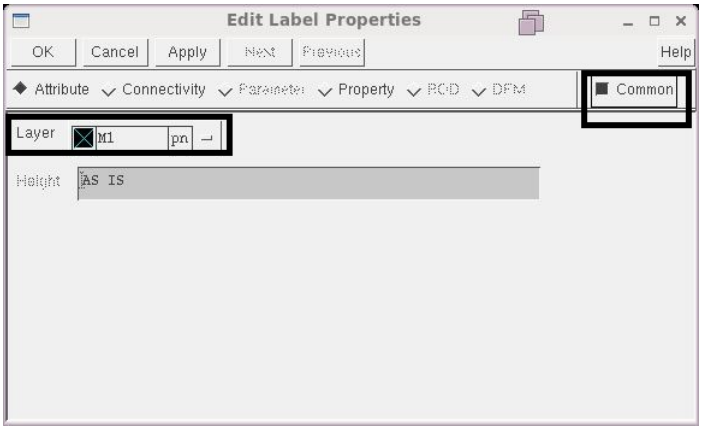


Figure A.14: LAYOUT Step 15

16. Connect pins as in schematic.

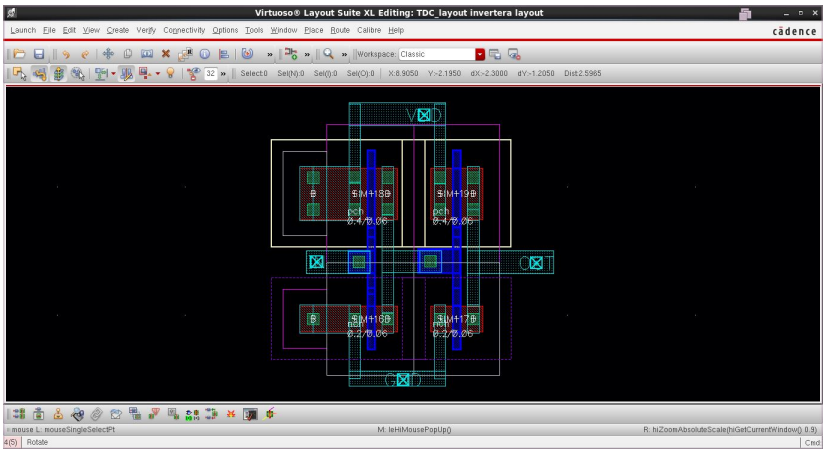


Figure A.15: LAYOUT Step 16

17. Till this we have finished the layout but we must go through some checks, First check is DRC check which checks errors in design rules

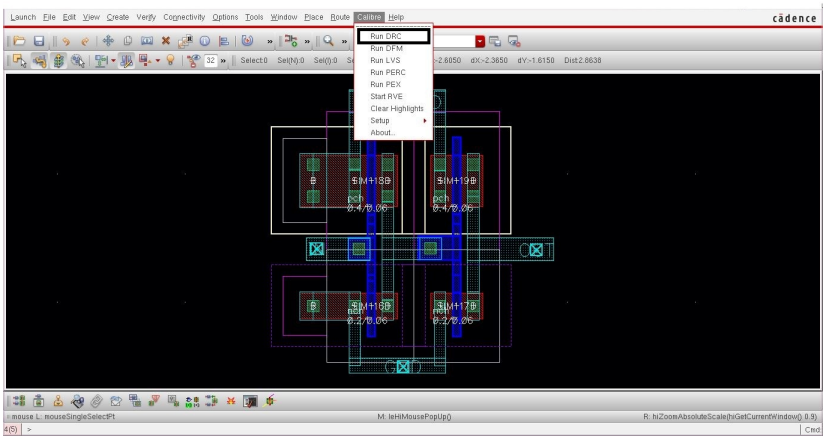


Figure A.16: LAYOUT Step 17

18. After selecting DRC this window appears which asks for the rules file we browse for “calibre.drc” file

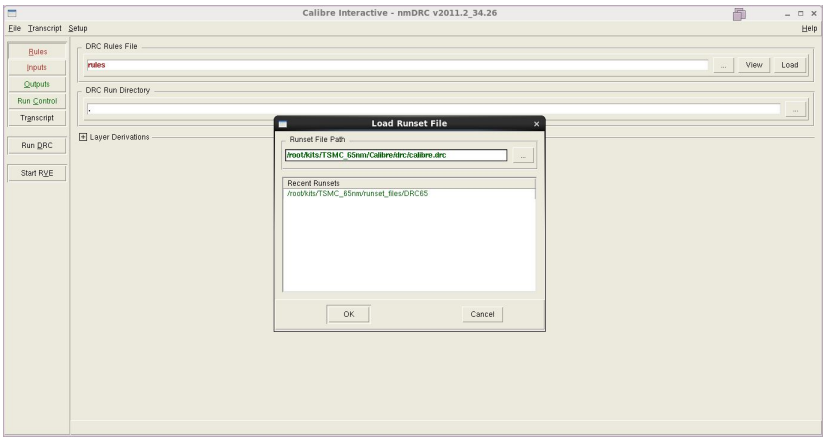


Figure A.17: LAYOUT Step 18

19. We make sure that the option “Export from layout viewer” in Inputs tab.

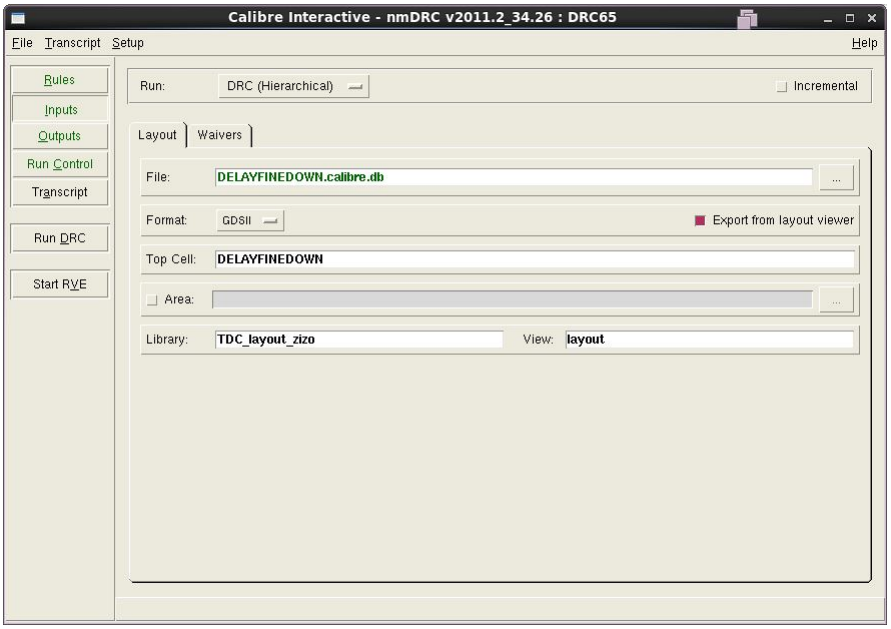


Figure A.18: LAYOUT Step 19

20. Click “Run DRC”. A window will appear which contains error in layout (if there is an error).

21. Second check is LVS “layout versus schematic” this is an important check as it make sure that all connections are the same as schematic.

22. From calibre menu we choose “LVS”, as in DRC a window appears where we select the rules file.

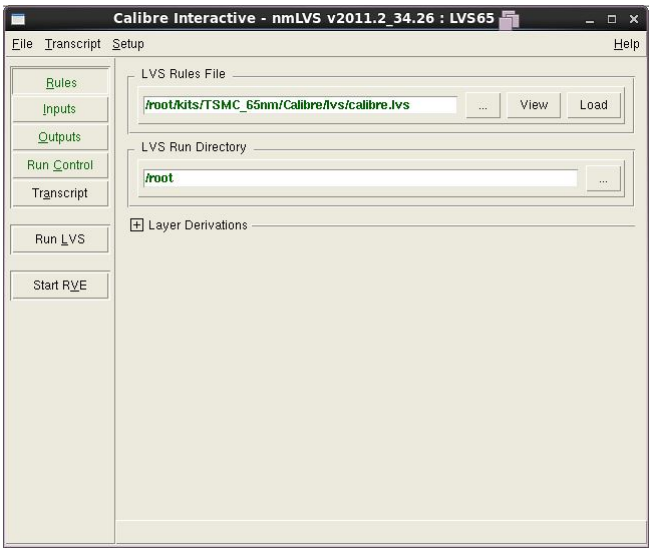


Figure A.19: LAYOUT Step 22

23. We make sure that the option “Export from layout viewer” in Inputs tab.

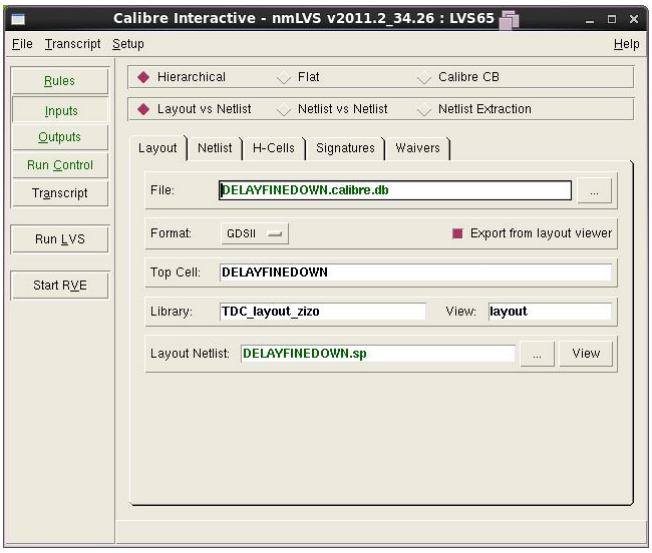


Figure A.20: LAYOUT Step 23

24. Click “Run LVS”, a window will appear, you must see correct sign and the smiley face to make sure that there is no error.

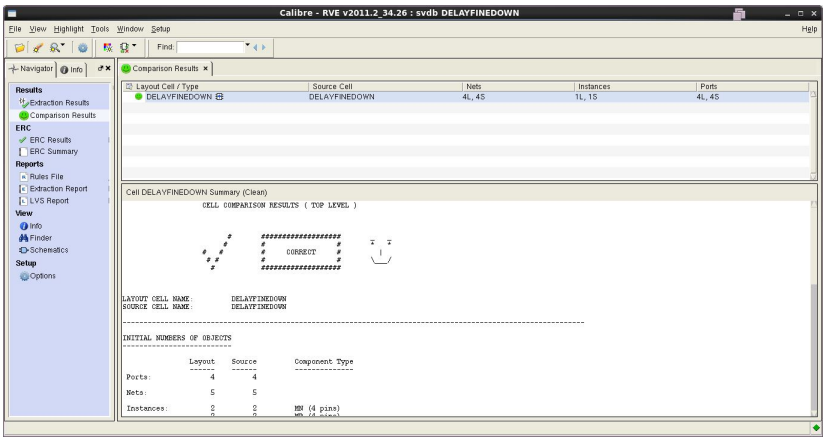


Figure A.21: LAYOUT Step 24

25. In this step we make the extraction of parasitic due to layout, from calibre menu we choose “Run PEX”.

26. This window appears which asks for the rules file we browse for “calibre.rcx” file

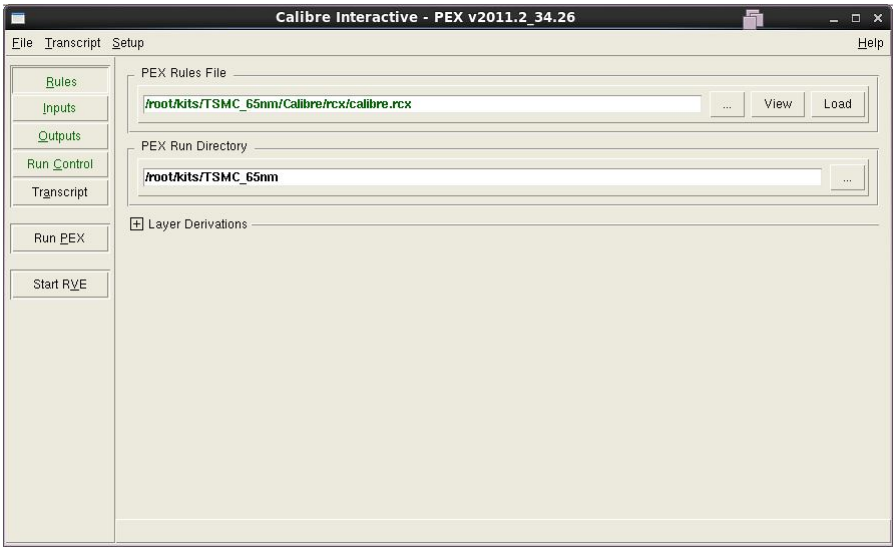


Figure A.22: LAYOUT Step 26

27. We make sure that the option “Export from layout viewer” in Inputs tab.

Figure A.23: LAYOUT Step 27

28. In the output tab we make sure of selecting extraction of “R-C-CC” parasitic.Also make sure that the output will be in format “CALIBRE VIEW” from “LAYOUT”[3]

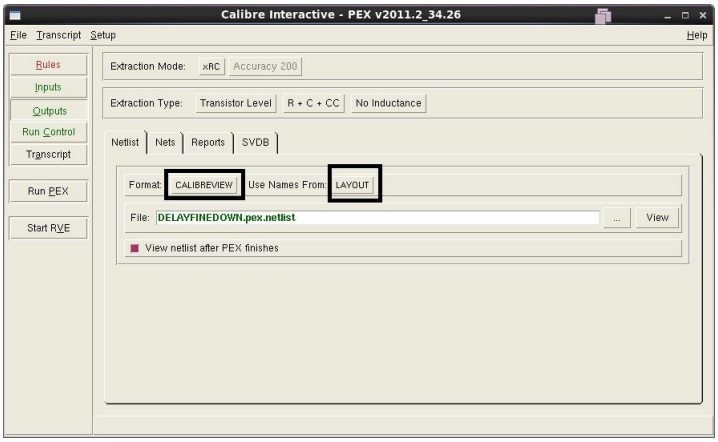


Figure A.24: LAYOUT Step 28

29. A window appears where we can see the parasitic are extracted (capacitors and resistances).[4]

[3] calibre view is a cell view which will contain parasitic elements

[4] you may need to make zoom to see the parasitic elements

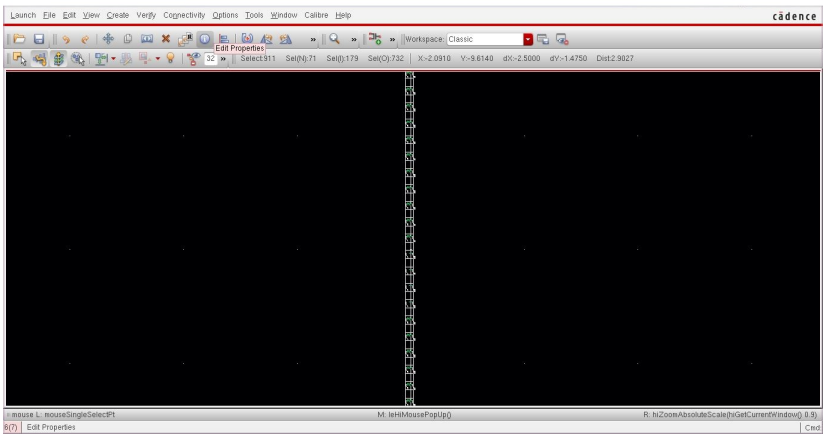


Figure A.25: LAYOUT Step 29

30. A cell view named “calibre” is created in the cell, This cell view contains all parasitic

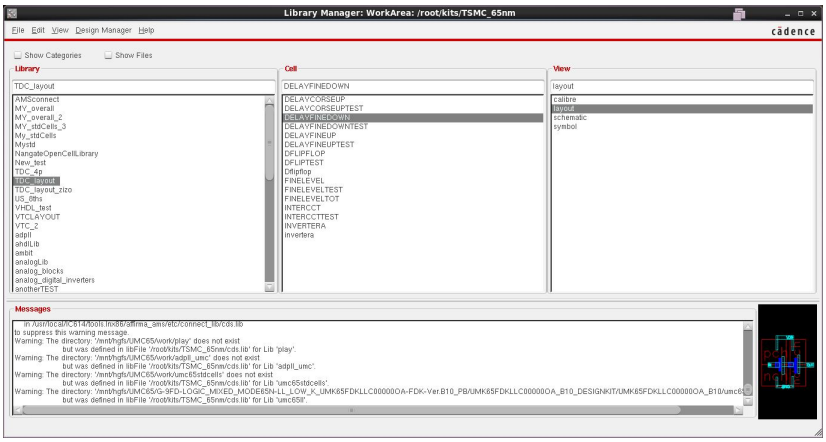


Figure A.26: LAYOUT Step 30

31. Now, We need to simulate parasitic components extracted from drawn layout to know its effect on the circuit functionality: We assume, that you have drawn a test bench file before to test functionality as shown

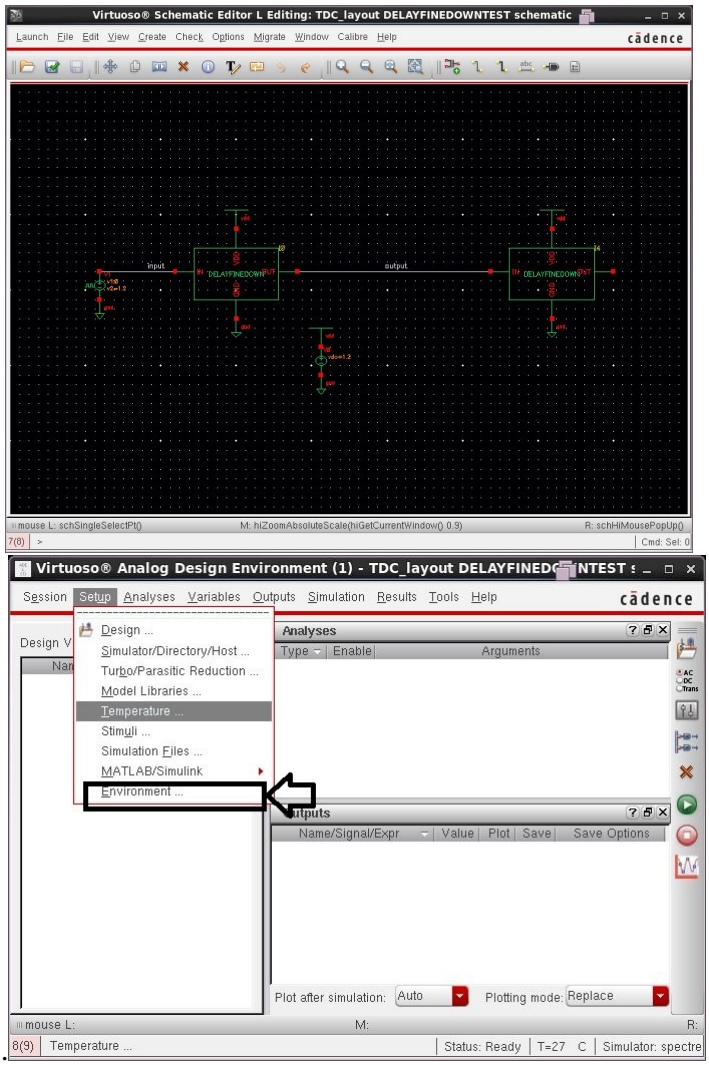


Figure A.27: LAYOUT Step 31

32. Now we will include a file called calibre view as shown, to simulate the parasitic.This is done by writing “calibre” before schematic as shown.

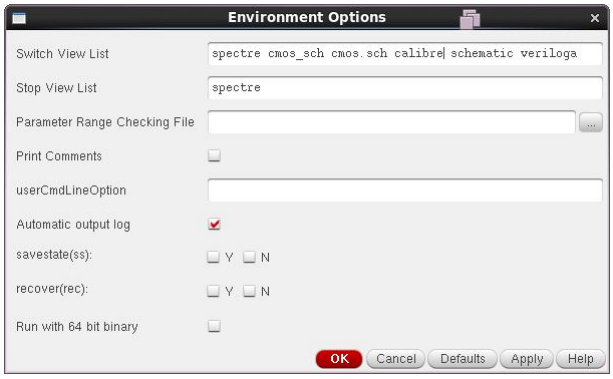


Figure A.28: LAYOUT Step 32

Now when running simulation the simulator tool will take into consideration the parasitic elements due to layout.

Appendix B

ENOB

This appendix shows how to calculate effective number of bits for the overall TADC.

Steps:

1-enter the input sine-wave with 30 MHZ, dc and amplitude as shown in figure for our example (dynamic range 145.5*2=291 mV, negative peak at 384mV and positive peak at 675mV).

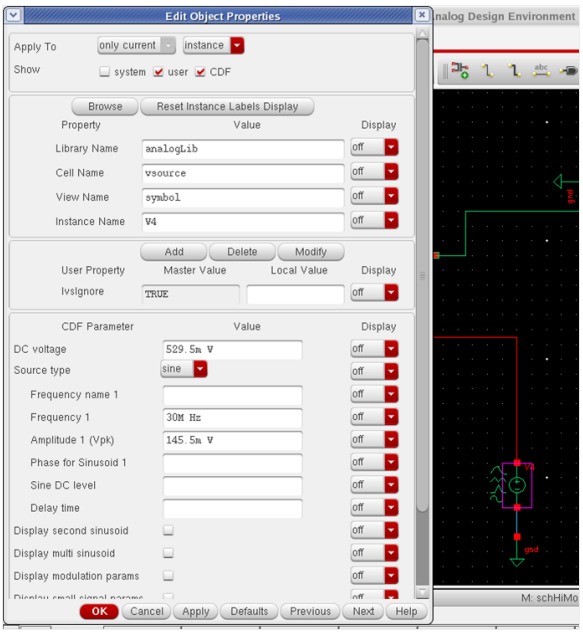


Figure B.1: ENOB calculation step 1

2-Use sample and hold to sample and hold the sine wave to input it to the VTC. The figure shows the input sine and the output of the sample and hold (that will be implemented from steps 3 to 10) for one period.

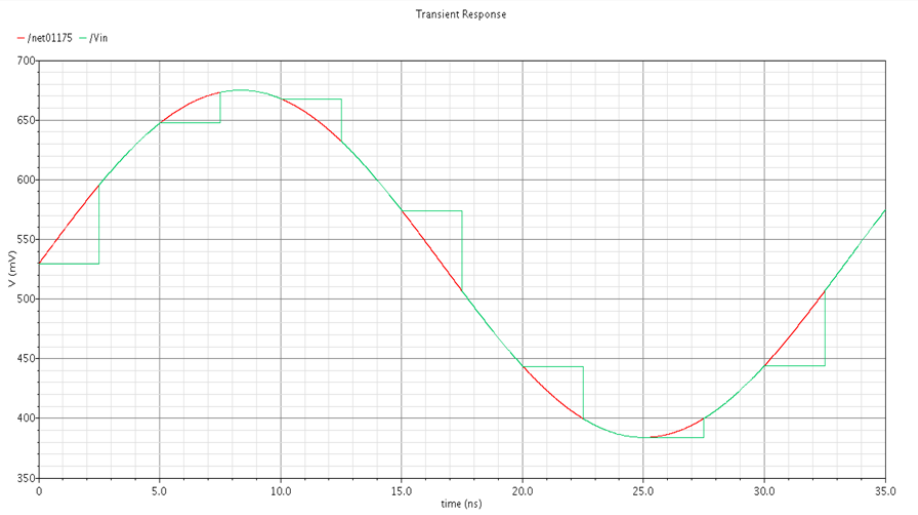


Figure B.2: ENOB calculation step 2

This ideal sample and hold was implemented using a VerilogA code and added to cadence as a symbol.

To write a VerilogA code and making it as a symbol we go in the following steps:

3- From file choose new then create a library.

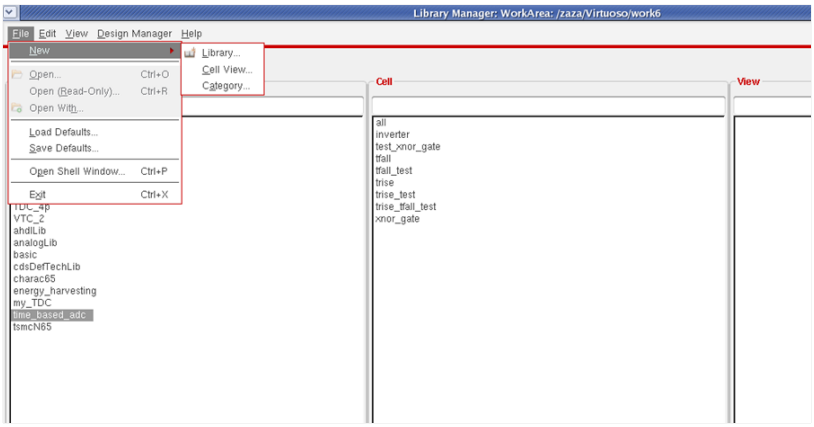


Figure B.3: ENOB calculation step 3

4- Name the library.

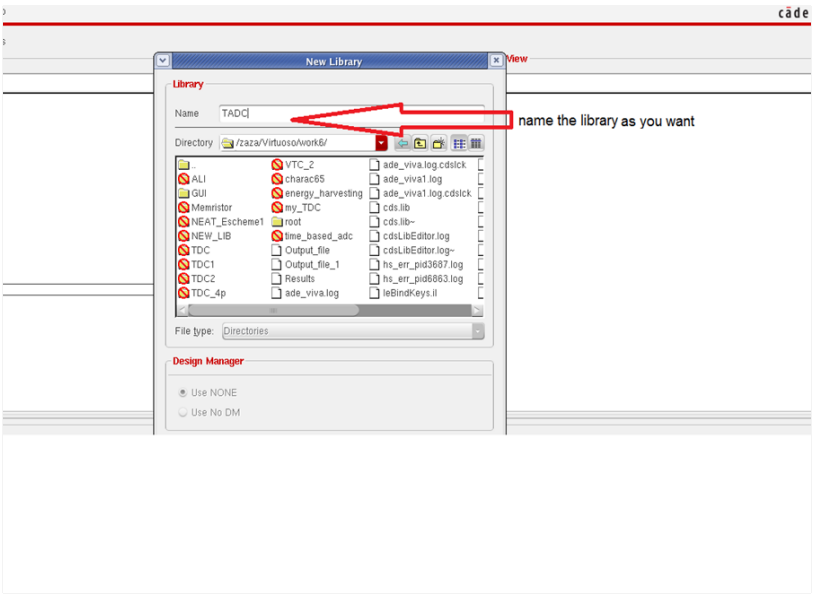


Figure B.4: ENOB calculation step 4

5- Create a cell view.

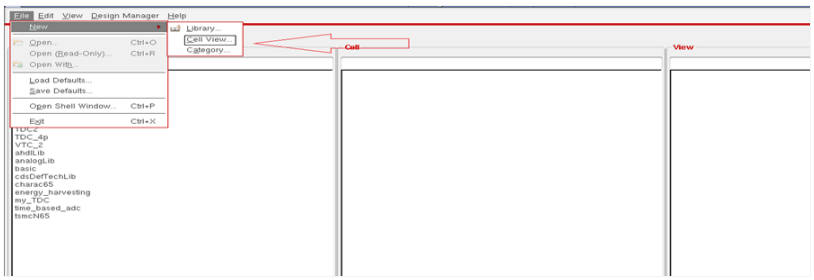


Figure B.5: ENOB calculation step 5

6- Name the cell view, choose VerilogA and press OK.

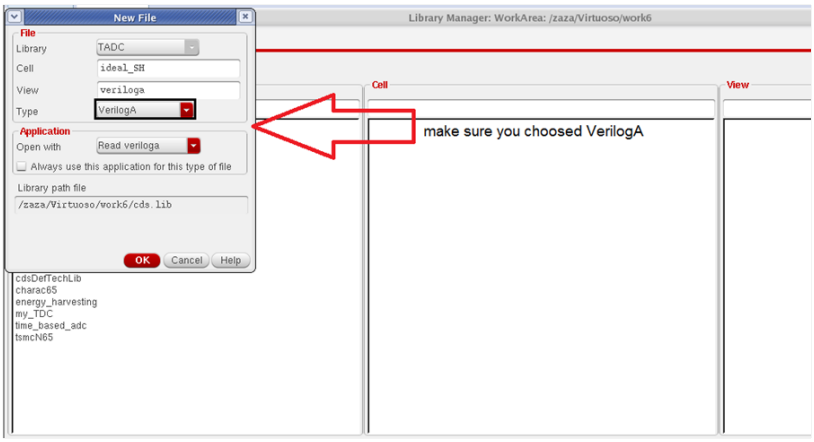


Figure B.6: ENOB calculation step 6

7- The shown window will be opened to write the code.

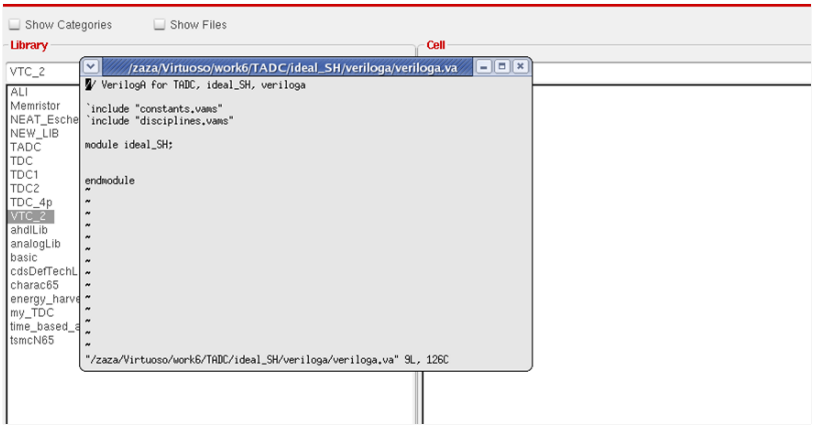


Figure B.7: ENOB calculation step 7

8- Write the code of the ideal sample and hold in the window as shown in the figure below.

```
`include "constants.vams"
`include "disciplines.vams"

 `include "discipline.h"
module sample_hold(in, out, clk);
input in, clk;
output out;
voltage in, out, clk;
parameter real clk_vth = 0.5;
real v;
analog begin
@(initial_step)
v=V(in);
if (analysis("static") || (V(clk) > clk_vth))

v = V(in);//V(in); // passing phase

@(cross(V(clk) - clk_vth,0))
v = V(in); // sampling phase
V(out)<+ v;
end

endmodule
```

Figure B.8: ENOB calculation step 8

Then save and close.

This is the code to be easily copied:

include “constants.vams”

include “disciplines.vams”

include “discipline.h”

module sample_hold(in,out,clk);

input in,clk;

output out;

voltage in, out, clk;

parameter real clk_vth = 0.5;

real v;

analog begin

@(initial_step)

v=V(in);

If (analysis(“static”) || (V(clk) > clk_vth))

v=V(in); //V(in); // passing phase

@(cross(V(clk) – clk_vth,0))

v = V(in); // sampling phase

V(out)<+v;

end

endmodule

9- You can find the sample and hold as a symbol in the cell view as shown, open it.

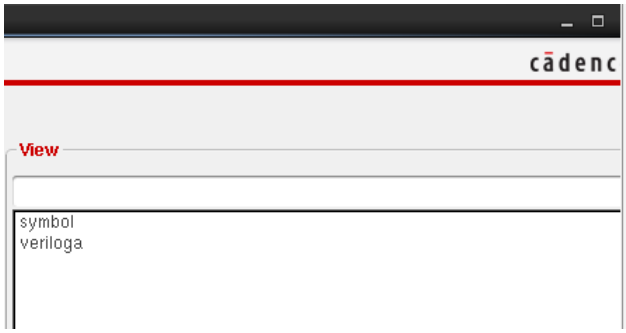


Figure B.9: ENOB calculation step 9

10- The sample and hold is ready to be used

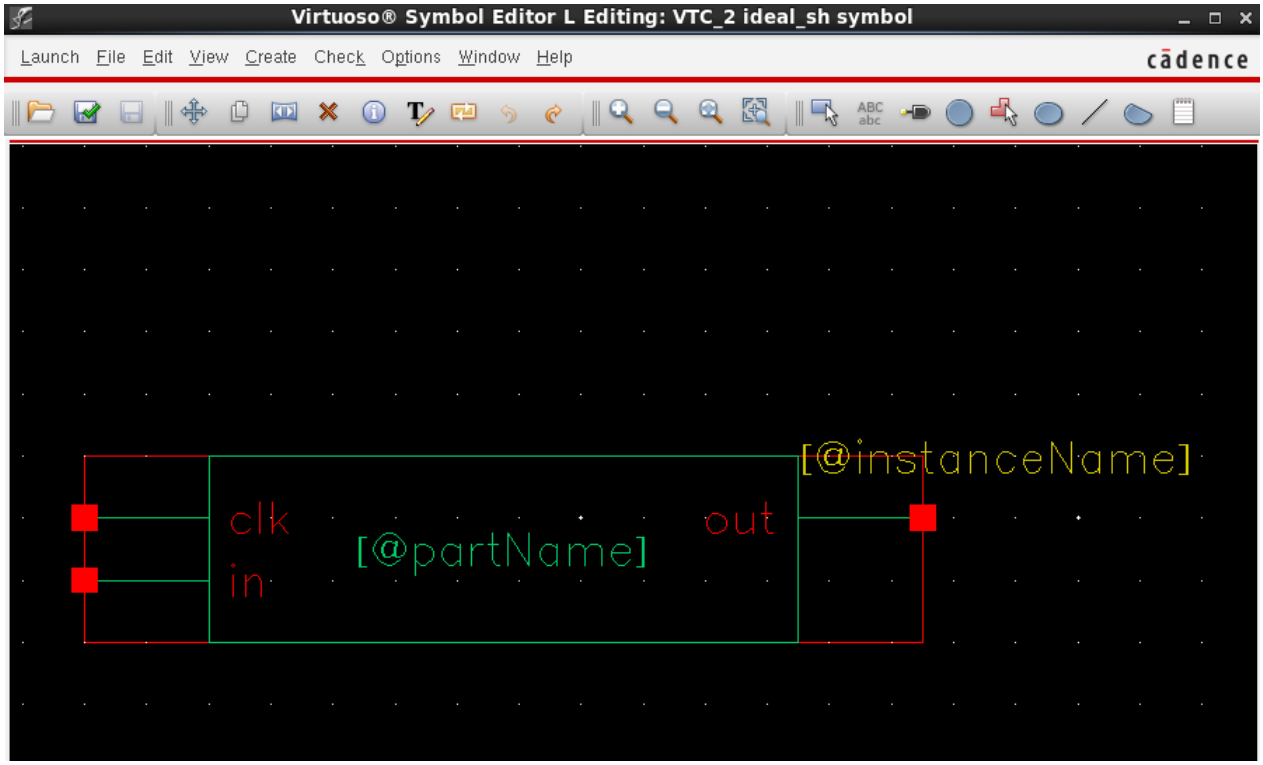


Figure B.10: ENOB calculation step 10

11- Now run the simulation of the TADC and after it finishes open the calculator and draw the following equation: (0.384+(1 * v(”/OO0” ?result ”tran-tran”) * (0.291/255)) +

(2 * v(”/OO1” ?result ”tran-tran”) * (0.291/255)) + (4 * v(”/OO2” ?result ”tran-tran”) * (0.291/255)) + (8 * v(”/OO3” ?result ”tran-tran”) * (0.291/255)) + (16 * v(”/OO4” ?result ”tran-tran”) * (0.291/255)) + (32 * v(”/OO5” ?result ”tran-tran”) * (0.291/255))+(64 * v(”/OO6” ?result ”tran-tran”) * (0.291/255)) + (128 * v(”/OO7” ?result ”tran-tran”) * (0.291/255)))

Where OO0, OO1, ... , OO7 are the 8-bits outputs at the schematic.

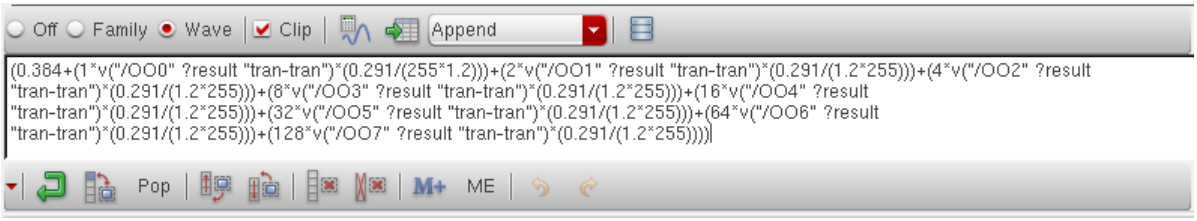


Figure B.11: ENOB calculation step 11

The drawn equation and run period from the sampled signal is as shown.

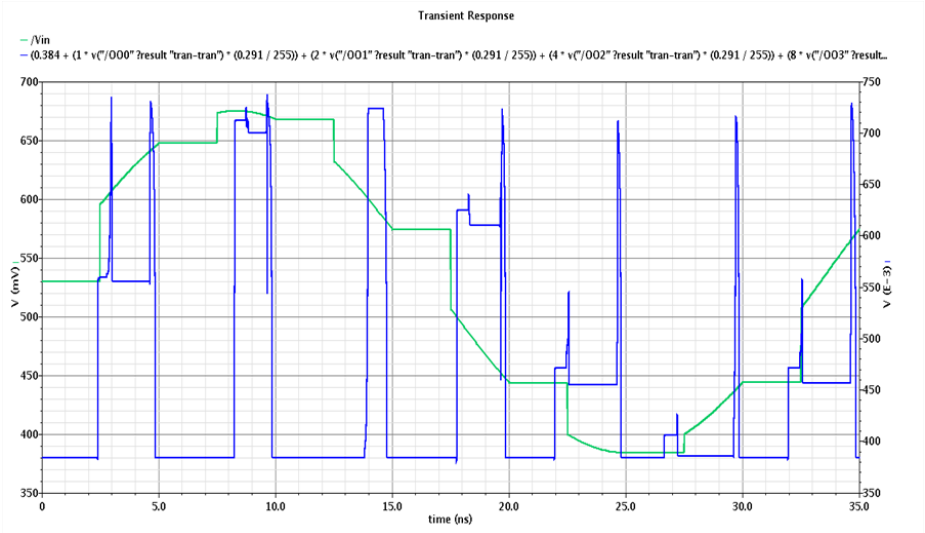


Figure B.12: ENOB calculation result

we sample the outputs at 4.25ns ,9.25ns ,,, 34.2ns

12- From the tools choose table.

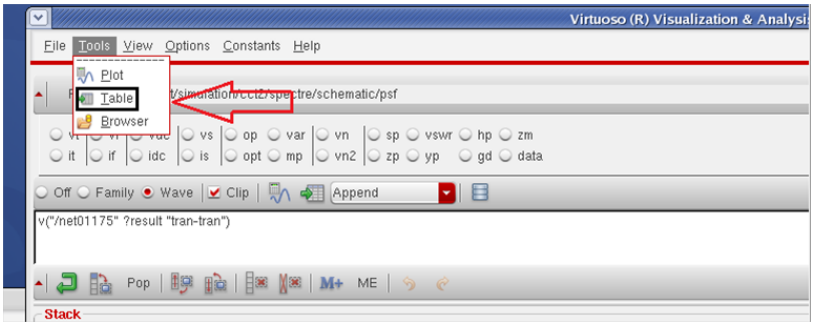


Figure B.13: ENOB calculation step 12

13-Choose the period, we choose it 5ns for our design, choose the end of simulation to be 170ns and choose where to take the samples.

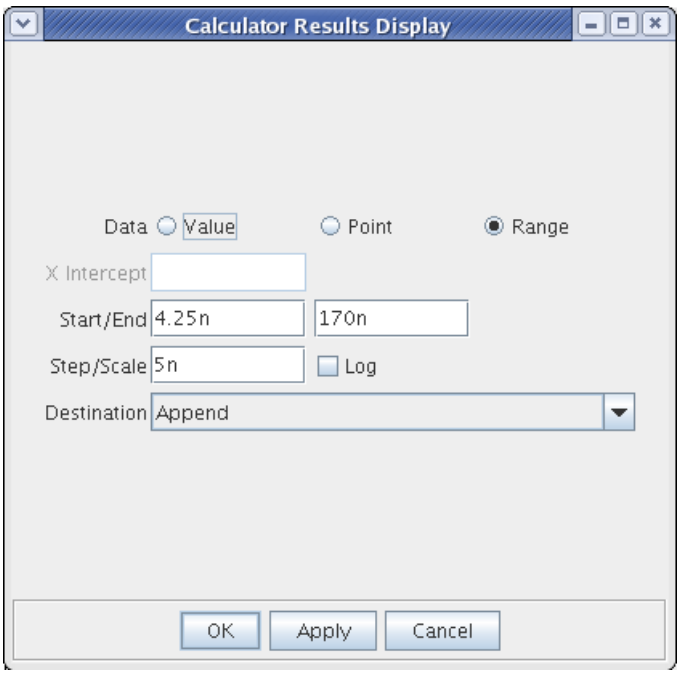


Figure B.14: ENOB calculation step 13

14- The values of the reconstructed signal is shown now at sample time.

time (s)	(0.384 + (1 ...
4.25E-9	526.6E-3
9.25E-9	647.6E-3
14.25E-9	667.0E-3
19.25E-9	572.3E-3
24.25E-9	443.3E-3
29.25E-9	385.1E-3
34.25E-9	444.5E-3
39.25E-9	573.4E-3
44.25E-9	668.1E-3
49.25E-9	646.5E-3
54.25E-9	525.5E-3
59.25E-9	412.5E-3
64.25E-9	392.0E-3
69.25E-9	483.3E-3
74.25E-9	614.5E-3
79.25E-9	673.9E-3
84.25E-9	613.4E-3
89.25E-9	483.3E-3
94.25E-9	392.0E-3
99.25E-9	412.5E-3
104.3E-9	526.6E-3
109.3E-9	647.6E-3
114.3E-9	667.0E-3
119.3E-9	572.3E-3
124.3E-9	443.3E-3
129.3E-9	385.1E-3
134.3E-9	444.5E-3
139.3E-9	573.4E-3
144.3E-9	668.1E-3
149.3E-9	646.5E-3
154.3E-9	525.5E-3
159.3E-9	412.5E-3
164.3E-9	392.0E-3
169.3E-9	493.5E-3

Figure B.15: ENOB calculation step 14

15- We can take it as excel file too, choose file -> save as CVS

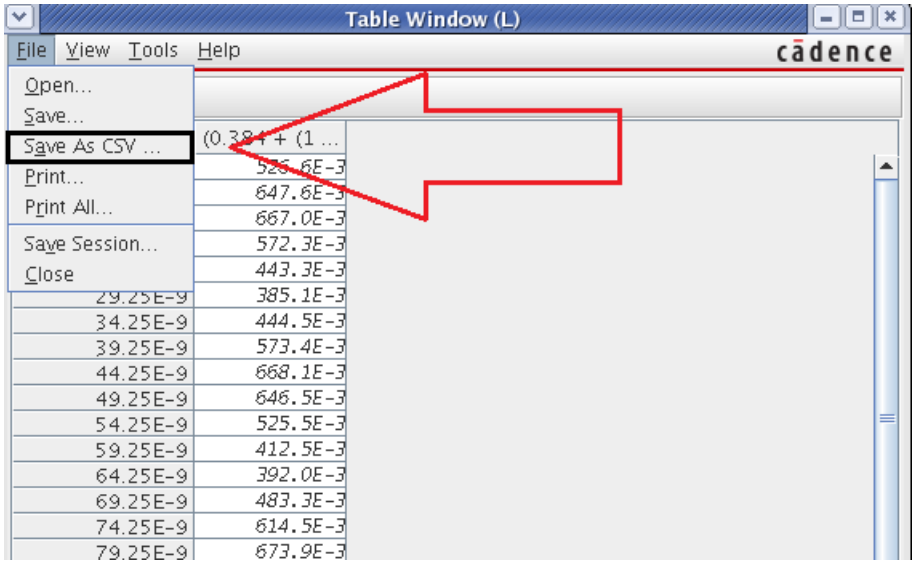


Figure B.16: ENOB calculation step 15

16 -Choose the place you want to save in with the name you want and click save.

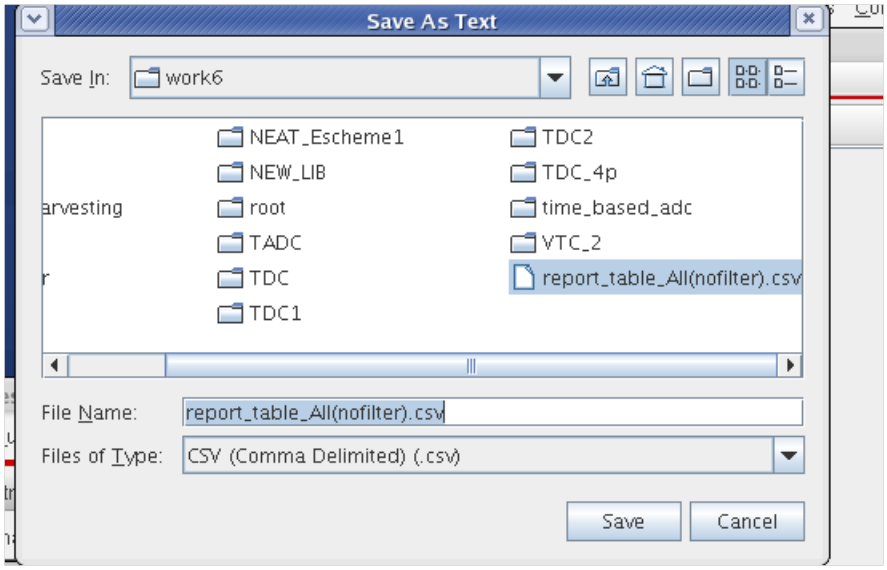


Figure B.17: ENOB calculation step 16

17- Open a new Matlab file and write the following function

function [snrdB,ptotdB,psigdB,pnoisedB] = calcSNR(vout,f,fBL,fBH,w,N)

fBL=ceil(fBL); fBH=ceil(fBH); signal=(N/sum(w))*sinusx(vout(1:N).*w,f,N); % Extracts sinusoidal

signal noise=vout(1:N)-signal; % Extracts noise components

stot=((abs(fft((vout(1:N).*w)’))).^2); % Bitstream PSD

ssignal=(abs(fft((signal(1:N).*w)’))).^2; % Signal PSD

snoise=(abs(fft((noise(1:N).*w)’))).^2; % Noise PSD

pwsignal=sum(ssignal(fBL:fBH)); % Signal power

pwnoise=sum(snoise(fBL:fBH)); % Noise power

snr=pwsignal/pwnoise;

snrdB=dbp(snr);

norm=sum(stot(1:N/2));%/sum(vout(1:N).^2)*N; % PSD normalization

if nargout > 1 ptot=stot/norm;

ptotdB=dbp(ptot);

end if

nargout > 2

psig=ssignal/norm;

psigdB=dbp(psig);

end if

nargout > 3

pnoise=snoise/norm;

pnoisedB=dbp(pnoise);

end

this function called calcSNR which will be used to calculate SNR .

18- Open a new Matlab file and write the following code

data_stream = data.*255/(0.291);

dc_offset = sum(data_stream)./length(data_stream);

data_stream = data_stream-dc_offset;

N = length(data_stream); %Length of your timedomain data (remove %some samples for settling

w = hann(N)/(N/4); %Hann window

bw = 60e6; % Base-band

Fs = 200e6;

Fin = 30e6; %must not be multiple of fs

f = Fin/Fs; % Normalized signal frequency

fB = N*(bw/Fs); % Base-band frequency bins(the BW you are looking at)

[snr,ptot,psig,pnoise] = calcSNR(data_stream,f,3,fB,w,N);

ENOB = (snr - 1.76)/6.02; % Equivalent resolution in bits

This code take a matrix called ‘data’ (output data from cadence) and perform calculation of SNR using SNR defined in step 1. As shown in code above Fs , Fin ,Bw must be defined according to the parameter in the under test ADC. In the proposed TADC we have Fs=200 MHz and we test a Fin of 30 MHz.

19- From Matlab window select “Import data”.

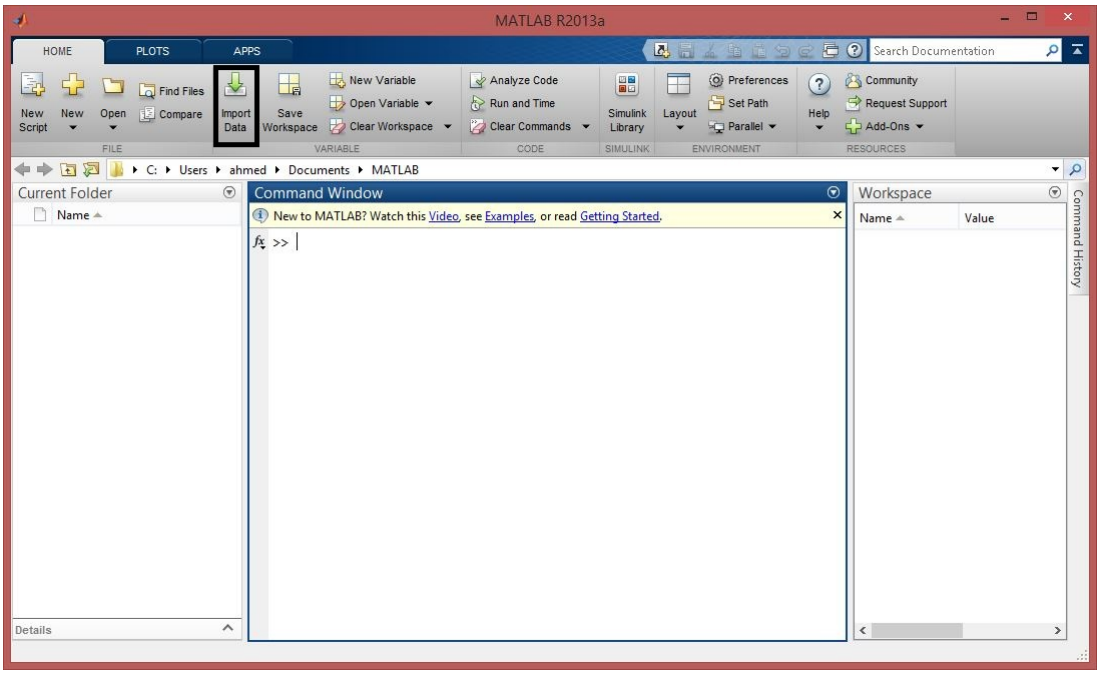


Figure B.18: ENOB calculation step 19

20- Select file generated from cadence

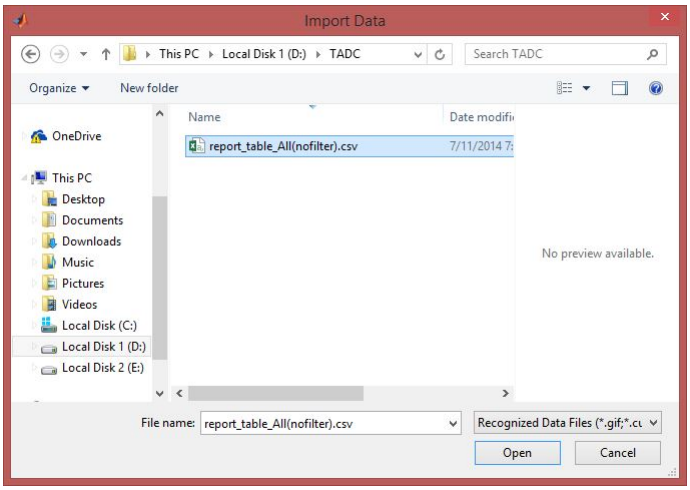


Figure B.19: ENOB calculation step 20

21- Select columns which contain output volt and press import selection

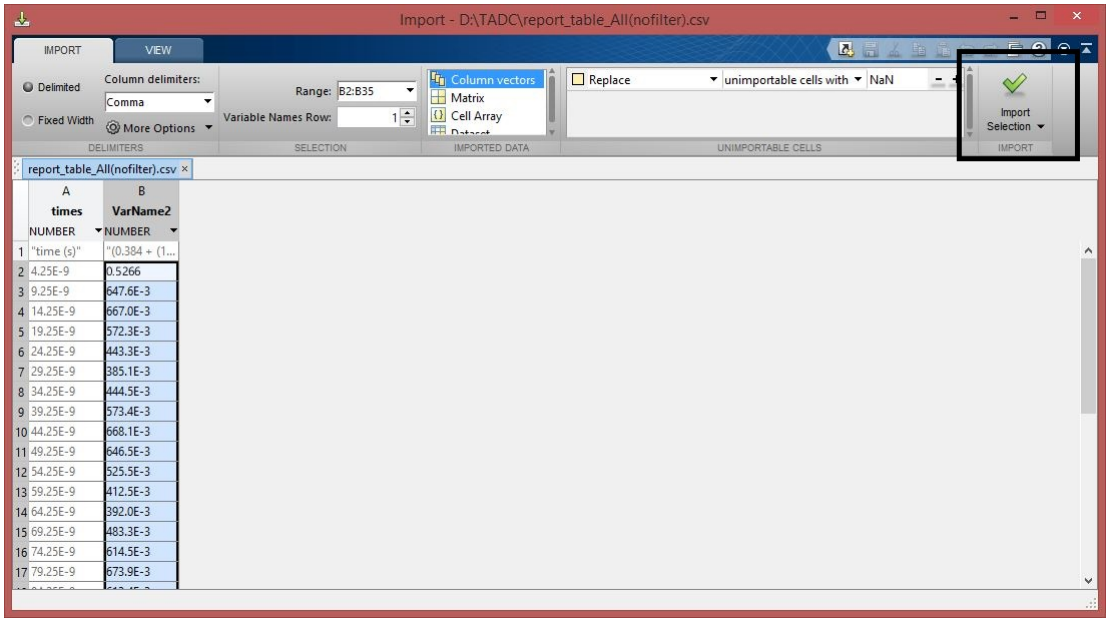


Figure B.20: ENOB calculation step 21

22- A matrix will be created in work space rename it to “data” as it will be used by the above code

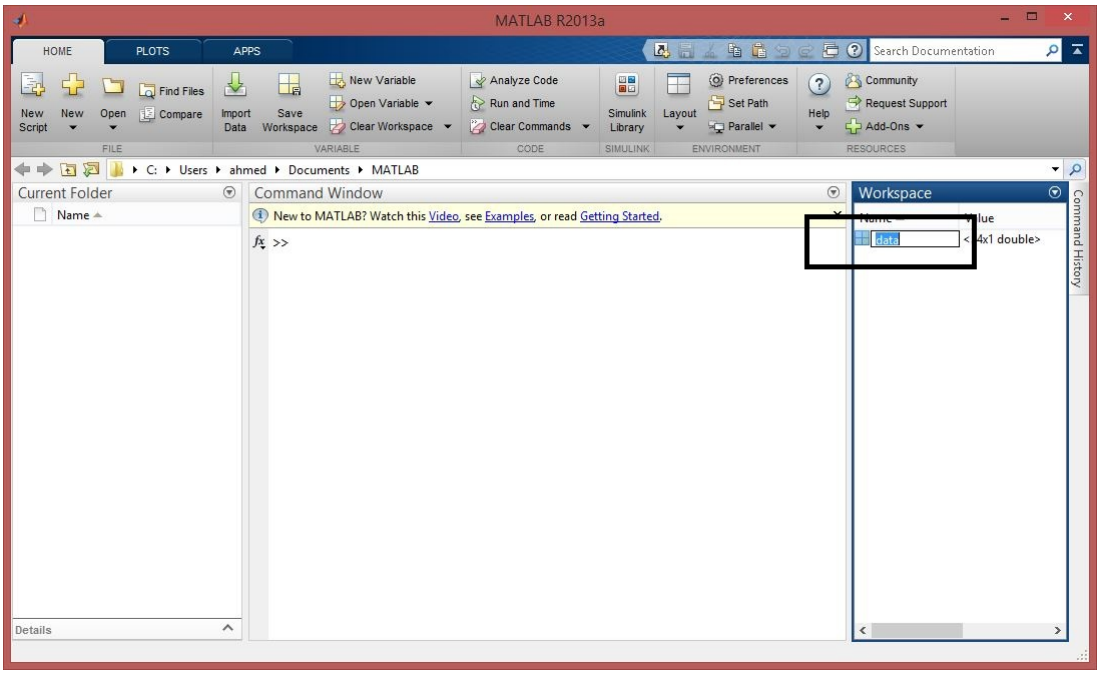


Figure B.21: ENOB calculation step 22

The imported data contain the samples calculated by cadence, you can repeat these data for several cycles using simple Matlab code or by double click on data and repeat values in the array. But to be able to use above code number of samples must be multiple of 4.

23- Run the code in Matlab (code written in step 2). A variable will be created contain the calculated ENOB.

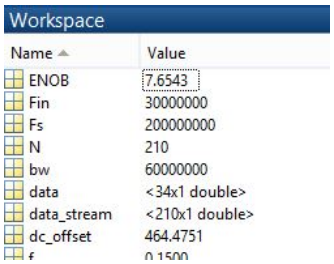


Figure B.22: ENOB calculation step 23

Bibliography

[1] Andrew Robert Macpherson, “A Time-Based 5GS/s CMOS Analog-to-Digital Converter, CALGARY, ALBERTA,MAY”, 2013.

[2] Dazhi wei, “TIME-BASED ANALOG-TO-DIGITAL CONVERTERS”, 2005.

[3] Mohamed Amin , “Design of a Time Based Analog to Digital Converter”, Waterloo, Ontario, Canada, 2009.

[4] D. Johns, and K. Martin, Analog Integrated Circuit Design. John Wiley, 1997.

[5] A. Oppenheim, and A. Willsky, Signal and System. Second Edition. Prentice Hall signal processing series, 2004.

[6] P. E. Allen, and D. R. Holberg, CMOS Analog Circuit Design. Second Edition. Oxford university press, 1987.

[7] Walt Kester, “Data Conversion Handbook” ,2005.

[8] Matthew Collins, “On-Chip Time Measurement Architectures and Implementation” , may 2009.

[9] S. Haykin, “Communication Systems”. Forth Edition, John Wiley & sons Inc,2001.

[10] A. Djemouai, M. Sawan, and M. Slamani, ”New 200 MHz Frequency Using Notch Filtering at the VTC Input or by Using Switches Locked Loop Based on New Frequency-To-Voltage Converters Approach,” Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS’99), pp. 89–92, 1999.

[11] ____ , ”New CMOS integrated pulse width modulator for voltage conversion applications,” in Proceedings of the 7th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2000), vol. Jounieh, Lebanon, Dec. 2000, pp. 116-119.

[12] P. Dudek and S. Szczepanski, ”A High Resolution CMOS Time-to-Digital Converter Using a Vernier Delay Line,” IEEE Journal of Solid State Circuits, vol. 35, no. 2, pp. 240–247, Feb. 2000.

[13] T. Watanabe, T. Mizuno, and Y. Makino, "An all-digital analog-to-digital converter with 12 -,uv LSB using moving-average filtering," IEEE Journal of Solid State Circuits, vol. 38, no. 1, pp. 120–125, Jan. 2003.

[14] C. Gray, L. Wentai, W. V. Noije, T. H. Jr., and R. Cavin, "A sampling technique and its CMOS implementation with I Gb/s bandwidth and 25 ps resolution," IEEE Joumrnal of Solid State Circuits, vol. 29, no.3, pp. 340-349, Mar. 1994.

[15] H. Pekau, A. Yousif, and J. Haslett, "A CMOS integrated linear voltage-to-pulse delay-time converter for time based analog-to-digital converters,"Proceedings of IEEE International Symposium on Circuits and Systems. ISCAS 2006, pp. 2373-2376, 2006.

[16] Mostafa, H., and Y. Ismail, "Highly-Linear Voltage-to-Time Converter (VTC) Circuit for Time-Based Analog-to-Digital Converters (T-ADCs)", IEEE International Conference on Electronics, Circuits, and Systems (ICECS'13), Abu Dhabi, United Arab Emirates, IEEE, pp. 149 - 152, 2013.

[17] S. Henzler, S. Koeppe, D. Lorenz, W. Kamp, R. Kuenemund, and D. Schmitt-Landsiedel, "A Local Passive Time Interpolation Concept for Variation-Tolerant High-Resolution Time-to-Digital Conversion," IEEE Journal of Solid-State Circuits, vol. 43, no. 7, pp. 1666- 1676, July 2008.

[18] Gordon W. Roberts, and Mohammad Ali-Bakhshian, "A Brief Introduction to Time-to-Digital and Digital-to-Time Converters",2010 .

[19] Jianjun Yu, Fa Foster Dai, and Richard C. Jaeger, "A 12-Bit Vernier Ring Time-to-Digital Converter in 0.13 .m CMOS Technology",2010.

[20] G. H. Li and H. P. Chou , "A High Resolution Time-to-Digital Converter Using Two-Level Vernier Delay Line Technique",IEEE,2007.

[21] Mohammad Maymandi-Nejad and Manoj Sachdev, "Digitally Programmable Delay Element Design and Analysis", 2002.

[22] Nihar R. Mahapatra , Alwin Tareen , and Sriram V., "COMPARISON AND ANALYSIS OF DELAY ELEMENTS" ,2000.

[23] Bui Van Hieu, Seunghyun Beak, Seunghwan Choi, Jongkook Seon, Taikyeong Ted. Jeong , "Thermometer-to-binary Encoder with Bubble Error Correction (BEC) Circuit for Flash Analog-to-Digital Converter (FADC)" ,2010 .