Part1: Design of a 2 GHz 64 sample Analog Memory and 7 bit SAR ADC in 0.13μm CMOS Part2: Testing of an 800 MHz $\Delta\Sigma$ Modulator

A Project Report

submitted by

NIMIT NIGANIA

in partial fulfilment of the requirements

for the award of the dual degree of

BACHELOR OF TECHNOLOGY

and

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.

May 28, 2011

THESIS CERTIFICATE

This is to certify that the thesis titled **Part1: Design of a 2 GHz 64 sample Analog Memory and 7 bit SAR ADC in 0.13μm CMOS**
Part2: Testing of an 800 MHz $\Delta\Sigma$ Modulator, submitted by **Nimit Nigania**, to the Indian Institute of Technology Madras, for the award of the degree of **Bachelor of Technology** and **Master of Technology**, is a bona fide record of the work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Nagendra Krishnapura
Advisor,
Assistant Professor,
Dept. of Electrical Engineering,
IIT-Madras, 600 036

Place: Chennai
Date: 27 May 2011

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Nagendra Krishnapura for his guidance from the beginning till the end of the project. I am indebted to him for his ideas, valuable guidance and the interesting topics he has taught me.

I would like to thank various faculty members of IIT Madras under whom I have done courses and laboratories. I would like to thank Dr. Shanthi Pavan, Dr. K Sridharan and Dr. Nitin Chandrachoodan whose stimulating and inspiring lectures motivated me to work in circuits.

I would like to thank Vikas for helping me learn various aspects of testing a chip. I would not have been able to finish in time if not for the help of Lokesh who taught me layout and gave me some of his layout secrets. I would also like to thank Ankesh, Kunal, Debashish, Santosh,Animesh, Srinivas and Amrith, my friends in the VLSI lab and the DSDL lab for their support.

Last but not least, I express my sincere gratitude to my batchmates specially Harshit, Mayank and Sujay for the interesting discussions we had between us and who helped make this journey easy for me.

I would like to dedicate this thesis to my parents who always stood by me and motivated me to go further.

ABSTRACT

This thesis is divided into 2 parts. In the first part the design of an 2 GHz analog memory and 7 bit SAR(successive approximation) ADC(analog to digital converter) on 0.13μm technology is discussed. Analog memories are used for sampling signals at very high speed which are not possible using a ADC. The proposed design stores 64 samples of an input pulse sampled at 2 GHz and then a SAR ADC digitizes this at a much lower speed. The memory designed as part of this thesis is for applications in high speed physics experiments where we need to sample an input pulse available only for a short duration and not a continuously arriving waveform. The SAR ADC designed is a 7-bit asynchronous ADC and consuming 250μW of power having an SNDR of 43.7dB.

The second part of the thesis contains testing results of 800 MHz $\Delta\Sigma$ modulator compensated for more than unity feedback delay. The chip tested as part of this thesis contains a fast feedback path to compensate for delays which are greater than one clock cycle. This led to realize very high sampling speeds on 0.18μm technology. The ADC tested was a 4-bit modulator with sampling speed of 800MHz. The chip consumes $47.6mW$ of power from a 1.8V supply. The measured dynamic range is 75.5 dB for OSR of 25 and is 64dB for OSR of 12.5.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

ADC	Analog to Digital Converter
SAR	Successive Approximation
ELD	Excess Loop Delay
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
LPF	Low pass filter
LSB	Least Significant Bit
MSA	Maximum Stable Amplitude
MSB	Most Significant Bit
NTF	Noise Transfer Function
SDM	Sigma Delta Modulator
SNR	Signal to Noise Ratio
DR	Dynamic Range

Part I

Design of a 2 GHz 64 sample Analog Memory and 7 bit SAR ADC in 0.13μm CMOS

CHAPTER 1

Introduction

In most of the applications we see analog waveforms get digitised by directly using an analog to digital converter(ADC). But for many applications like high energy physics experiments or laboratory instrumentation continuous sampling and digitisation is not required. We only need to sample an analog pulse and store a snapshot of it in an analog memory. These sampled values are then read out and digitized using a much slower ADC. Using this technique rather than directly using a very high speed ADC saves power, cost and can give us a higher sampling rate [3].

The aim of the design is to digitize and store a pulse sampled at 2GHz. Since it is only a pulse which is needed to be sampled and there is a requirement for a low power design and hence a SAR ADC is being used to digitize the pulse. The SAR ADC designed is an asynchronous 7bit ADC. An analog memory is also needed as the SAR ADC(which is used because of its low power consumption) cannot work at input sample speeds as high as 2GHz. For example a 7 bit SAR ADC needs 7 cycles to digitize a single sample of the input. So for a input sampled at 2GHz we will need about $7 * 2 = 14$GHz of internal clock which is very difficult. In this design since we have to sample only a pulse and not a continuous arriving time signal so we sample the pulse at 2GHz and store it in an array of capacitors(64 in number) and then read and digitize the sampled value at a slow pace of the ADC. This thesis contains a circuit level implementation of the proposed analog memory and the SAR ADC. The proposed design is a viable alternative to continuous time analog to digital converters for applications where continuous sampling of the input waveform is not required.

Organization

Chapter 2 contains a general overview and working of an analog memory. It also explains the top level architecture and the working of the proposed analog memory design.

Chapter 3 discusses the basic storage cell used in the memory and also about the amplifier used for reading out the sampled values. It also discusses the circuit level implementation of the read control and the write control signal generation needed for the read and the write cycle of the memory.

Chapter 4 introduces the Successive approximation(SAR) ADC and discusses the top level implementation and working of the proposed SAR ADC.

Chapter 5 discusses the circuit implementation of the various blocks needed in the SAR ADC.

Chapter 6 contains the simulation results of the proposed design.

Chapter 7-8 These chapters make up the 2nd part of the thesis. They discuss about testing a 800 MHz $\Delta\Sigma$ modulator compensated for more than unity feedback delay.

Chapter 9 Concludes the thesis and discusses possible future work.

CHAPTER 2

Analog memory overview

In this chapter the basic concept of an analog memory is explained. We initially look at some already existing analog memories. After that the architecture of the proposed analog memory is discussed along with its functionality.

2.1 Introduction to analog memories

Figure 2.1 shows a basic design of an analog memory. It consists of N storage cells which can store N samples of the input analog signal. The input signal V_{in} is common to all the cells and its value at different time instants t_1, t_2, t_N are stored in cells $Cell_1$, $Cell_2$ till $Cell_N$. Each storage cell has its own write (q_{wi}) and read (q_{ri}) control signals. After all the N input samples are stored in the memory cell the values stored in the cells are read out in a sequence by asserting q_{r1} first then q_{r2} and so on. The values read out are then digitized by an ADC whose input sampling speed decides the read out time of the analog memory. Generally the time between consecutive read outs is much larger than the input sampling time during the write phase which greatly relaxes the requirements on the ADC. Hence a low speed ADC with low power consumption can be used.

For many designs for getting a much higher performance a calibration phase is followed before the signal is sampled to the memory. This calibration is used to help compensate for the device mismatch, non-linearity, variations etc between the memory cells. In the calibration phase a known input signal is applied and stored and then the output is read out. The transfer function of the input sample and the output sample is calculated for each cell. The inverse of this transfer function is then calculated to correct the acquired data. In the proposed design a

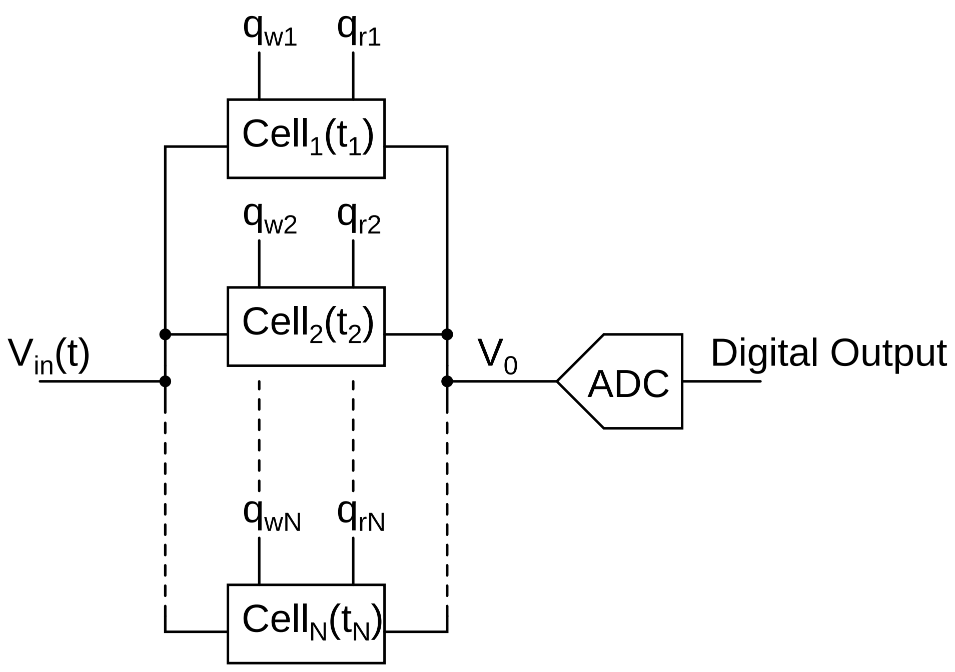


Figure 2.1: A basic analog memory with N storage cells

differential version of the analog memory is presented which reduces the effects of offset to a significant extent and thus not requiring calibration.

Most of the analog memory cells are realized using transistors as voltage switches and use capacitors to store the input samples [3]. Each storage cell can have its own buffer/opamp for the read cycle or there can be just one common opamp for all the storage cells. Memories with buffers in each cell occupy large area and also consume significant power compared to using just a single opamp [4][5]. The work analysed here will look at memories with a single opamp for the read phase. Apart from the memory array and the opamp an elaborate circuit for generating the read and write signals for each of the memory cell is required. The input signal sampling speed is limited by the speed of the write signal generation circuit. The next chapter will discuss the architecture of the proposed analog memory and also describe its operation.

2.2 Design of a 2 GHz analog memory

Analog memory is designed using a switched capacitor topology. An array of capacitors is used to store the sampled values of the input and then they are read one by one as shown in Figure 2.2.The capacitors are sized taking into account the sampling time (2GHz) and also so that they are significant compared to the parasitics of the switches it is connected to (to prevent leakage and parasitic effects). The switches were sized so that their on resistance is small enough to sample a signal around 1GHz on the capcaitor. The non-linearity will be introduced due to the *write* switch but the V_{gs} of the *qw* switch remains at $600mV$ so it will not introduce non-linearity. The *write* switch was sized so that its on resistance was much less than the on resistance of the *qw* switch so as to reduce the non-linearity in the sampled value.

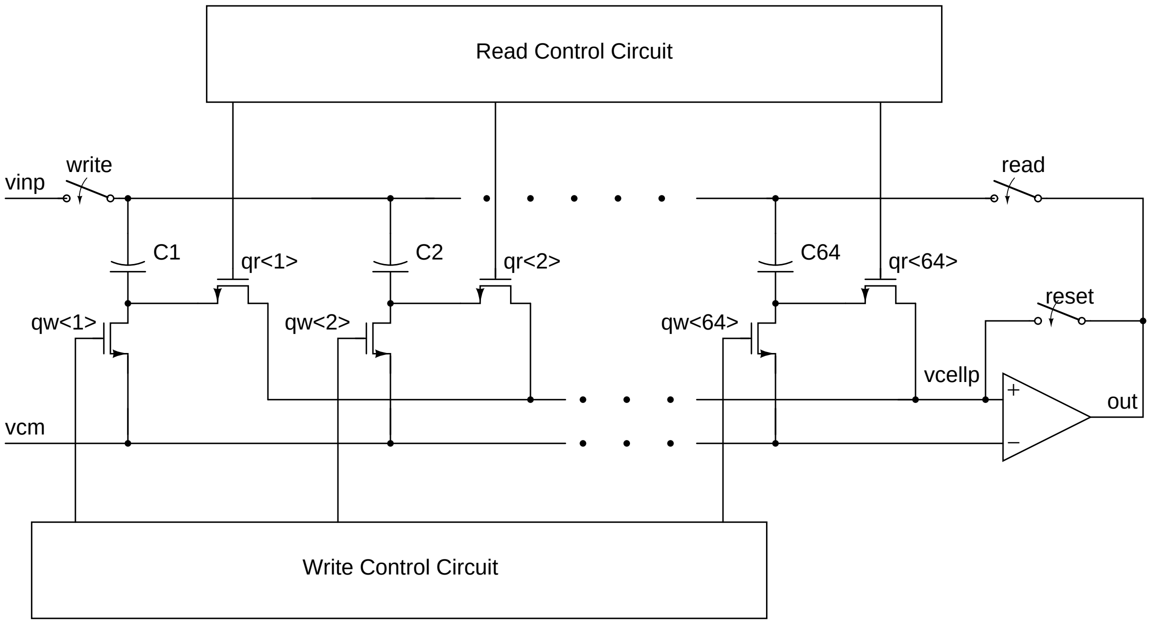


Figure 2.2: Analog memory (single ended equivalent)

In the schematic shown in Figure 2.2 *qw* and *qr* are the write and read signals for each cell respectively. The common mode voltage is labelled as *vcm* (which is $0.6V$). During the write phase $qr = 0$ and $qw = 1$ for a particular memory cell and input data is written on the capacitor. A totally of 64 such cells are used in the design to store the 64 samples of the input pulse sampled at $2GHz$.

The 64 capacitors of the 64 memory cells which hold the sampled value are labeled as C_1 through C_{64}. At the start of the write operation the *write* signal is asserted and the signals qw_1 through to qw_{64} corresponding to their respective memory cell are asserted and deasserted in a sequence. Since we are sampling the input signal at 2 GHz hence each of the *qw* signals have a width of $500ps$. After the write operation is complete we start the read operation. For the read operation the *write* signal is deasserted and the *read* signal is asserted. Now to read the values on each memory cell we assert the read signal *qr* for each memory cell and read out the values using the opamp. The read and the write signals

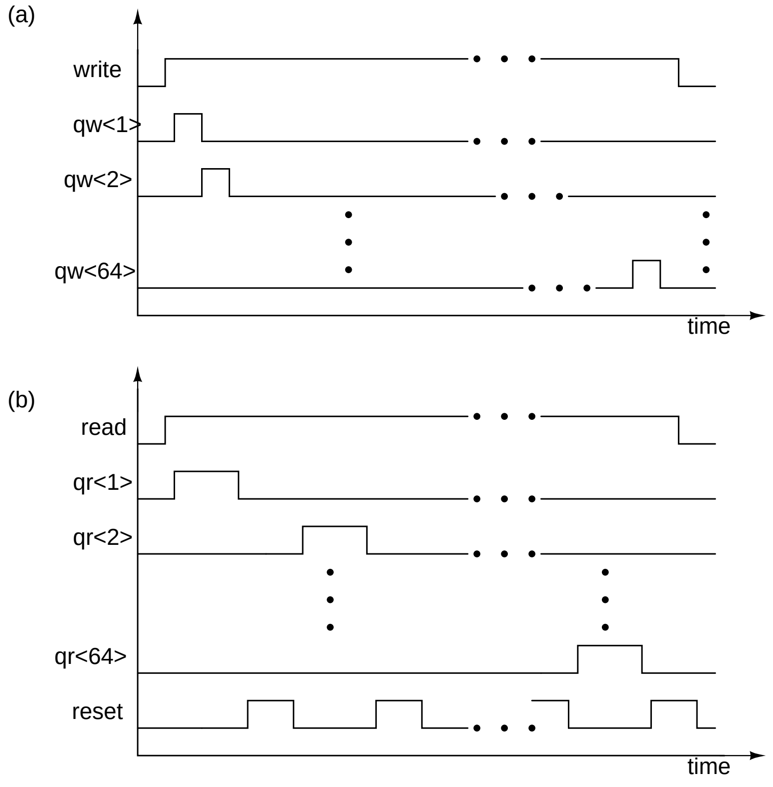


Figure 2.3: (a)Write signals (b)Read signals

are shown in Figure 2.3. During the time the sampled valued is being read out it is also continiously sampled on the capacitors of the SAR ADC which follows the analog memory. After every read operation of the cell we also need to reset the opamp outputs to the vcm values [3]. Resetting is very important to prevent charge sharing between the storage cells and the parasitics due to the switches and also the parasicitcs introduced in the layout. This will be discussed further in the next chapter. The SAR ADC performs the digitisation of the sample read out during this reset cycle before the next sample comes at the start of the next read pulse.

To mitigate the offsets due to charge injection and also due to charge sharing a differential version of the memory circuit is implemented. The differential version of the circuit is shown in Figure 2.4.

In the next couple of chapters the different components that make up the memory will be explained. First we will discuss the memory cell and the opamp, then the read and write control circuit will be described. Finally the architecture and operation of the SAR ADC will be explained.

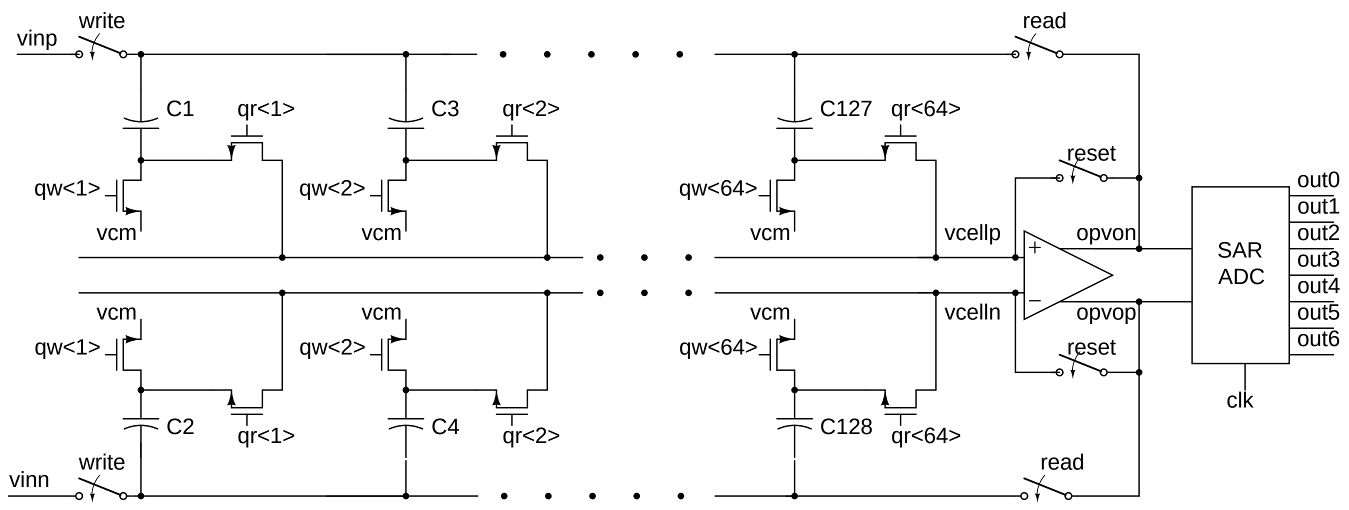


Figure 2.4: Differential version of the analog memory

CHAPTER 3

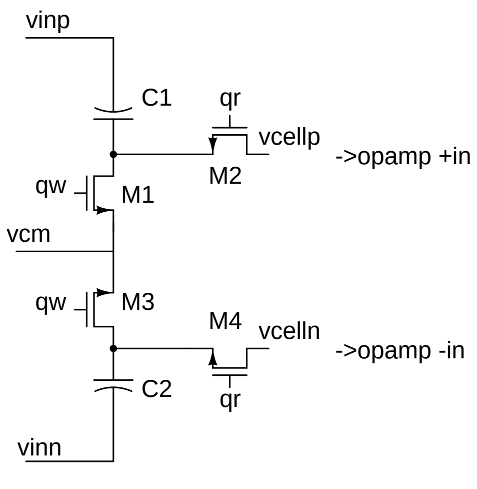
Analog memory circuit

Till this section the basic architecture of the analog memory has been explained. This section contains the circuit implementation of basic building blocks of the memory. The analog memory cell and the cascoded amplifier will be discussed first, then we will discuss about the read and write control circuits which form a very important part of the design.

3.1 Memory cell

As discussed in the previous chapter each storage element used in the memory is basically a capacitor with switches used for the read and write operation. Figure 3.1 shows a single memory cell along with the transistor sizes. After an input signal is sampled on the capacitor there will be charge injection due to the switching off of switches M1 and M3, the offset is same in both the capacitors.

Since a differential implementation is used hence when we read out the values stored in the capacitor the value read out by the opamp will be the common mode voltage plus the actual input signal sampled on the capacitor without the offset. If say for example while writing $vinp = vcm + vd$ and $vinn = vcm - vd$ then the charge stored across C1 corresponds to a voltage difference of vd similarly it is $-vd$ for C2. Now due to charge injection say the voltage changes to $vd + v_{inj}$ for C1 and $-vd + v_{inj}$ for C2. Then during the read cycle the opamp outputs will go to $vcm + vd$ and $vcm - vd$. As for the opamp inputs both $vcellp$ and $vcelln$ will go to $vcm + v_{inj}$. Hence the charge across the capacitor(C1/C2) still has the offset but the voltage read out corresponds to the actual input sampled which in this case is vd and $-vd$.



C1,C2: 125fF
M1,M3: (3u/120n)
M2,M4: (360n/120n)

Figure 3.1: Memory cell

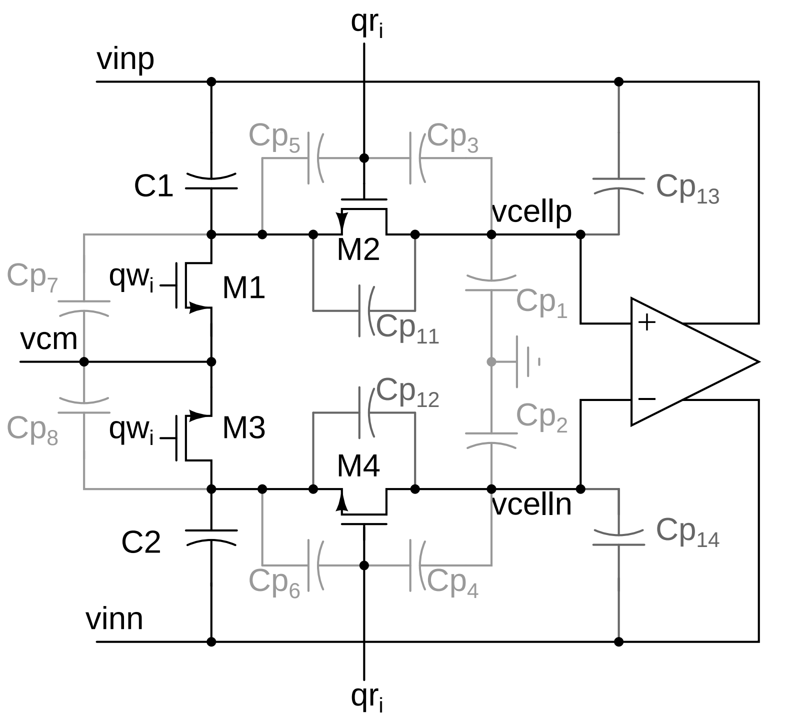


Figure 3.2: Memory cell with parasitics

Figure 3.2 shows the various parasitics present in the circuit. These become more pronounced in the layout of the circuit which leads to errors in the read out sample. Only parasitic capacitors across which charge changes are the ones which cause the offset. We now look at the effect of important parasitics on the read out sample value.

- Cp_1/Cp_2 - Before the read operation since the opamp is reset hence the voltage across these capacitors are vcm. Also after the read out operation the voltage remains the same at vcm or vcm plus some offset. In either case the voltage across C1/C2 change by the same amount hence not having any affect on the output.
- Cp_3/Cp_4- The voltage across these capacitors change by 1.2V as during the read operation both the gates of M2/M4 go to Vdd. Here again the offset due to this extra charge is same for both C1/C2 hence not having any affect on output. It only affects the input terminals of the opamp which will have same offset. A similar argument holds for capacitors Cp_7/Cp_8.
- Cp_{13}/Cp_{14}- This parasitic was found to affect the output significantly. As during the reset phase both $vinp$ and $vcellp$ were reset to vcm hence there was no charge across the parasitics Cp_{13}/Cp_{14}. And after the read operation the voltage across Cp_{13} went to the charge stored in $C1$ and voltage across Cp_{14} went to the charge stored in $C2$. Since the offsets caused by these were of opposite signs hence it appeared at the output. If say C1 had a voltage of vd across it and C2 had $-vd$. Then due to this parasitic the output read out will be:

  By charge conservation at the top plate of $C1/C2$

  $$(Voutp - vcm) * Cp_{13} + (Voutp - vcm) * C1 = vd * C1 \tag{3.1}$$

  $$\therefore Voutp = vcm + \frac{vd * C1}{Cp_{13} + C1} \tag{3.2}$$

  $$Similarly, Voutn = vcm - \frac{vd * C2}{Cp_{14} + C2} \tag{3.3}$$

  For $vd = 100mV, C1/C2 = 100fF$ and $Cp_{13}/Cp_{14} = 50aF$; as we have 64 cells which have both of the capacitor terminals as common these parasitics effectively become $50aF * 64 = 3.2fF$. For these values $Voutp = 96.9mV and Voutn = 102.1mV$ Hence in the layout both the input and the output were kept in such as way so as to minimize this capacitance.
- The parasitics Cp_{11}/Cp_{12} were also found to affect the output to a considerable extent. As again let us take a case in which the voltage sampled in

the capacitors $C1$ and $C2$ were vd and $-vd$ respectively. We assume all the other 63 remaining capacitors have no charge across them. Now after the read operation the output of the opamp should ideally go to $vcm + vd$ and $vcm - vd$. This causes the source of M2 and M4 for all the 63 remaining capacitors to $vcm+vd$ and $vcm-vd$. Hence the voltage across the parasitics corresponding to Cp_{11}/Cp_{12} in the remaining 63 cells go to $+vd$ and $-vd$. This again causes an error in the read out value. By charge conservation at the top plate of $C1/C2$

$$(Voutp - vcm) * (Cp_{11} * 63) + (Voutp - vcm) * C1 = vd * C1 \tag{3.4}$$

$$Voutp = vcm + \frac{vd * C1}{Cp_{11} * 63 + C1} \tag{3.5}$$

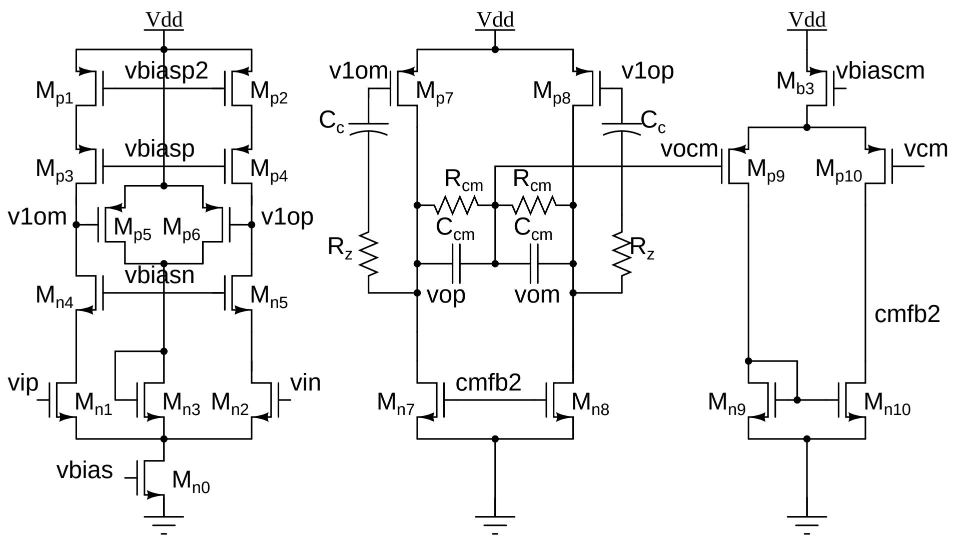
$$Similarly, Voutn = vcm - \frac{vd * C2}{Cp_{12} * 63 + C2} \tag{3.6}$$

Hence the error comes out to be similar to that calculated in the previous case due to Cp_{13}/Cp_{14}. But in this case there will always be a capacitance in the range of $50aF$ between the gate and the drain of the read switches hence there is no way to compensate for this error in the layout, other than using calibration technique discussed in the previous chapter.

Hence the differential implementation will take care of charge injection or any other kind of charge variations which affect both the capacitors in the same memory cell equally. But for parasitics which cause offsets of opposite sing in the storage capacitors, the error comes out at the output.

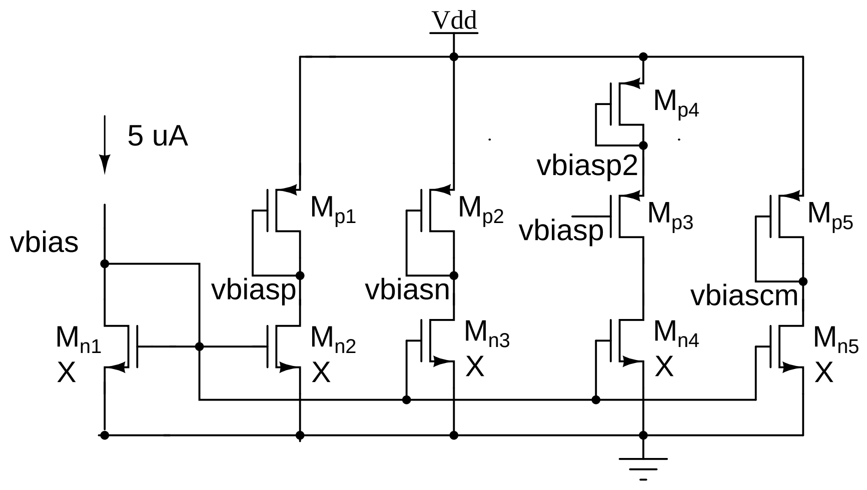
3.2 Two stage cascoded amplifier

As the input samples were already sampled at high speed and then read out at a low speed hence the requirements of the amplifier were relaxed. The only constraints that were important were the DC gain which decides the error and also the load which affects the bandwidth. The load for the opamp comes mainly from the SAR ADC, as described previously the input samples are directly read out to the DAC capacitors of the SAR ADC. For the current design this load came out to be in the range of $1.5pF$. The open loop unity gain frequency of the opamp was chosen to be around 100 MHz. To meet these requirements a 2 stage Miller-compensated opamp was designed as shown in Figure 3.3.



M_{n0}- 4(0.2/0.12)	$M_{p1,2}$- 10(0.8/.6)	$M_{p7,8}$- 120(0.2/.12)	R_z- 260 Ω
M_{n3}- 1(0.2/0.12)	$M_{p5,6}$- 3(0.2/.012)	$M_{n9,10}$- 1(0.8/.6)	C_{cm}- 25fF
$M_{n1,2}$- 5(0.8/.6)	$M_{p3,4}$- 12(0.2/0.12)	$M_{p9,10}$- 4(0.8/.6)	R_{cm}- 500 KΩ
$M_{n4,5}$- 15(0.2/0.12)	$M_{n7,8}$- 60(0.2/.12)	C_c- 70fF	

Figure 3.3: Opamp schematic



X = (0.2μ / 0.12μ)	M_{p4} = (1μ / 0.6μ)
M_{p1} = (0.32μ / 0.12μ)	M_{p5} = (0.5μ / 0.12μ)
M_{p2} = (1.2μ / 0.15μ)	M_{p3} = (2.5μ / 0.12μ)

Figure 3.4: Opamp biasing circuit

Figure 3.4 shows the biasing circuit used for the amplifier. We used a cascoded structure to realize a high DC gain and to keep the unity gain frequency within the desired range. The lengths of the input transistors were chosen to be high so as to provide high DC gain. For the second stage the width of the transistors were chosen to be very high so as to take care of the load and keep the second pole away from the first pole.

Since during the read cycle as discussed before the inputs of the opamp go to the common mode voltage plus the offset voltage due to charge injection and other parasitics. Hence there was a requirement for a strong common mode feedback circuitry. As single common mode feedback circuit was not found enough to make the output settle in the desired time(as this constituted a 3rd order system) so a separate common mode feedback circuitry was used for both the first and the second stage.

The opamp consumes 470μW of power and has a DC gain of 2200. The magnitude and the phase plot of the opamp for an output load of 1.6pF is shown in Figure 3.5. The bandwidth is 100MHz and the phase margin is 65 degrees.

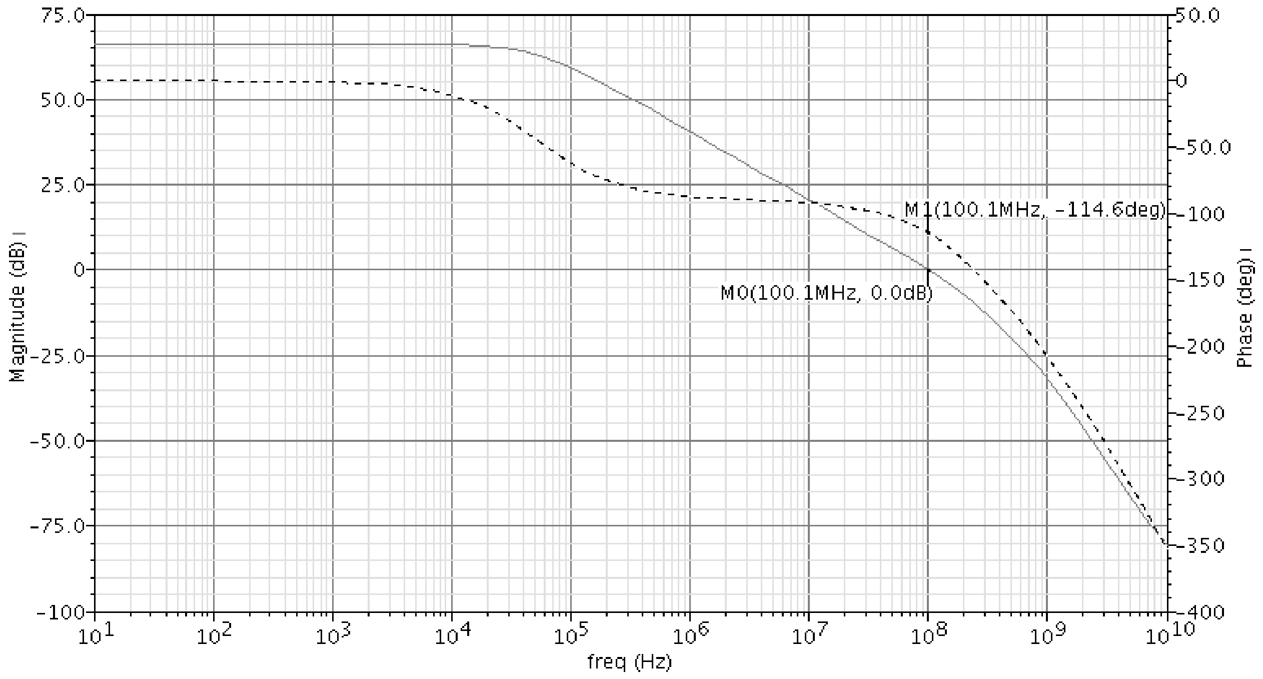


Figure 3.5: Magnitude and phase plot of the opamp

3.3 Read control circuit

The input samples are stored in an array of 64 capacitors which requires 64 pulses both for the read and the write cycle. As the input signal has to be sampled at 2 GHz we need to generate write signals of 500ps width. The pulses generated by the write control circuit is very crucial and all the 64 pulses(each of width 500ps) have to be consecutive one after the other. As for the read control signals timing is not so crucial as the time period is user defined and are of much longer width(around 50ηs period) compared to the write signals(500ps). Also it is followed by an *asynchronous* SAR ADC so matching of time period or jitter in the read signal is not a major problem. The maximum speed of the read control circuit is limited by the maximum sampling rate of the asynchronous SAR ADC and also the time taken to reset the output of the opamps.

The 64 read control signals are generated using a 2 phase shift register logic [3][6]. The logic used is shown in Figure 3.6. The shift register is initialized by keeping both the clocks high and setting the input to be low. Before generating the read pulses we first generate 2 non overlapping clocks which are used to control the signal propagation. The signal generation starts by asserting the serial input high for one clock cycle. Then this signal keeps propagating through each inverter stage every alternate cycle as and when the clocks keep toggling. The output read signal generated from the method described above has the period which equals the period of the non-overlapping clocks. So we will get 64 consecutive read pulses. But to prevent errors in the output due to charge sharing as explained in the previous chapter we always reset the opamp after every read operation. Hence we want a reset cycle between 2 read pulses. To do this we just AND the read pulses with one of the non overlapping clocks, this will make the width of read pulse same as that of the input clock as shown in Figure 3.6. The reset signal can then be taken as to be the other clock, so that it is high when there are no read pulses. The input signal is the output from the previous stage and the output pulse is AND with one of the input clocks to generate generate the required read pulse.

The sizes of the inverters are chosen to be enough to drive the output loads.

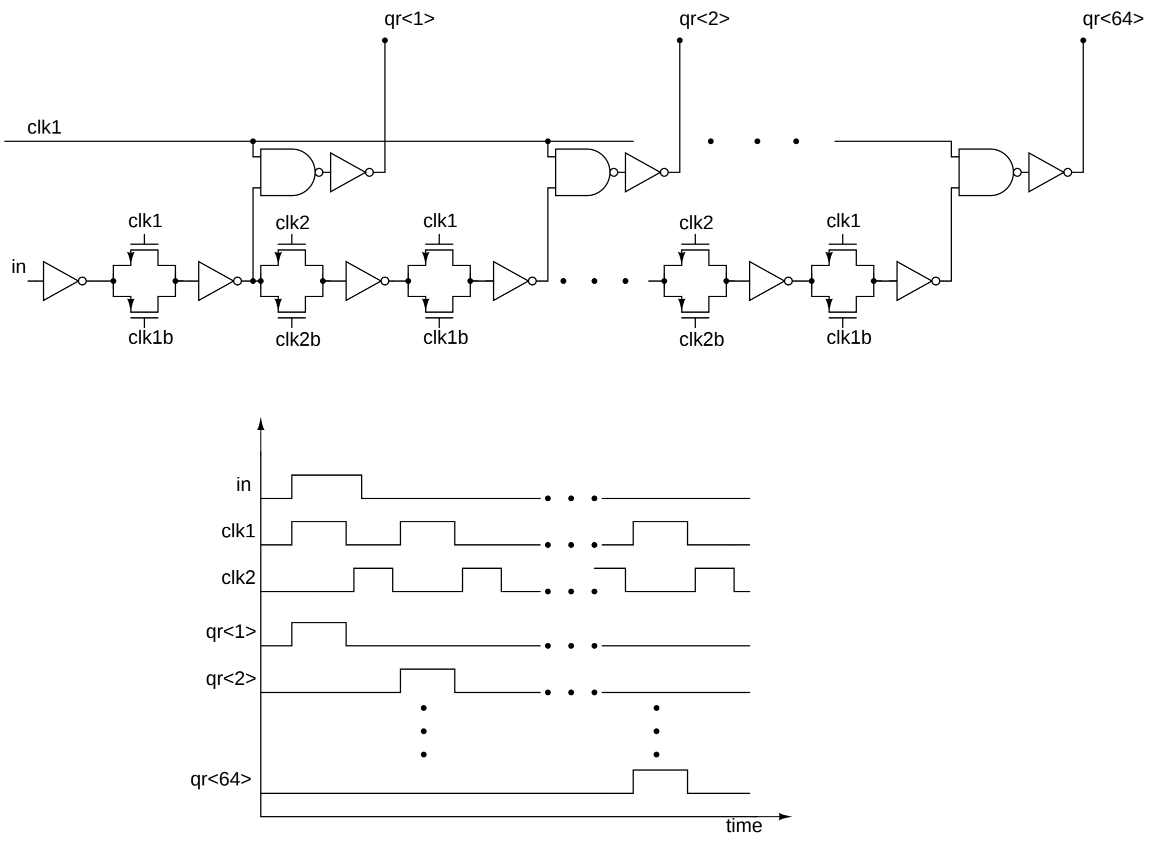


Figure 3.6: Read control circuit and timing diagram

3.4 Write control circuit

Many of the analog memories use shift register based topology for generating the pulses but their performance degrades at speeds more than a hundred MHz [3].So for generating the write pulses in the proposed design a delay chain is used whose delay is controlled by the help of a DLL(a delay locked loop). The delay chain along with the write signals is shown in Figure 3.7. The delay through the chain is sensitive to fabrication process variations, the operating temperature and the power supply level. In the proposed circuit by the help of a DLL(which basically acts as a feedback circuit for controlling the delay) we make the sampling frequency independent of these parameters. In a DLL we take an input clock or a pulse and pass it through an array of delay elements which are basically buffers whose delay can be controlled by some control voltage. After the input is passed through the buffers the output is compared with the input which was fed to the buffer array. This comparison is done by the phase detector which gives a UP or DOWN signal depending on the relative phases of the 2 clocks(the input and the delayed clock). This UP and DOWN signals are then fed to a charge pump. The charge pump steers currents into or out of a capacitor (whose voltage called as control voltage) depending on the input signals(UP/DOWN). So if the UP signal is high we increase the voltage across the capacitor and it is opposite for DOWN. This control voltage Vc is used to control the delay of the buffer array and completes the feedback loop. So in the locked condition the control voltage remains constant and there is no phase difference between the input clock and the delayed version of it.

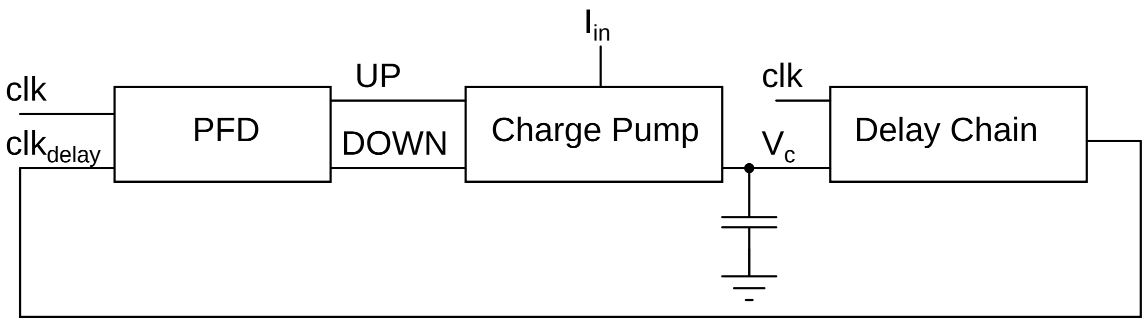


Figure 3.7: Delay locked loop

3.4.1 Delay locked loop(DLL)

DLL or a delay locked loop is a negative feedback circuit for locking an input clock with a delayed version of it. It mainly consists of 3 blocks each is discussed in the coming sections. The overall schematic of the DLL after putting together all the three components is shown in Figure 3.7.

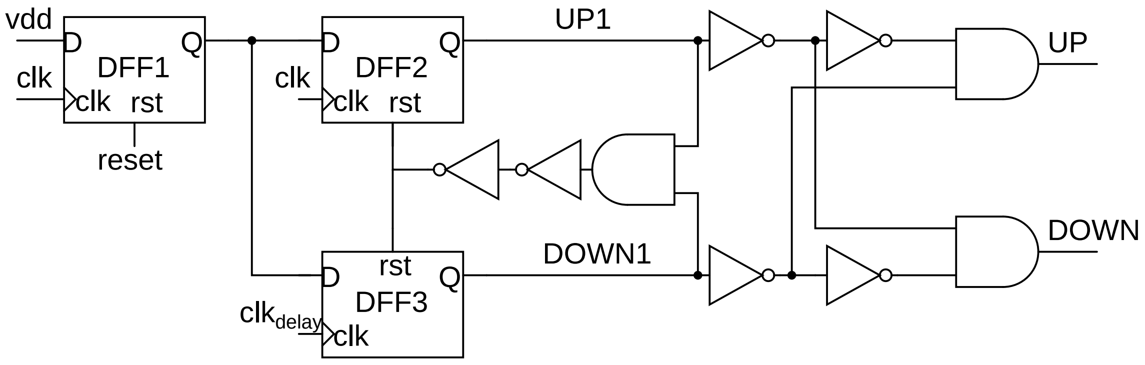


Figure 3.8: Phase frequency detector

Phase frequency detector(PFD)

The phase frequency detector is designed similar to ones in common PLL's(Phase Locked Loops) and is shown in Figure 3.8. It detects the phase difference between the input clock and its delayed version. It then asserts the UP signal if the delayed signal lags the input clock and it asserts the DOWN signal for the other case. These UP and DOWN signals are fed to the charge pump which controls the control voltage of the delay chain. After both the input pulses arrive making UP1 and DOWN1 go high, the reset circuitry resets the DFF's. Due to the delay in the reset circuitry the UP1 and DOWN1 signals stay on for the duration of the reset delay. Ideally if we give the UP1 and DOWN1 signal directly to the charge pump then we expect that the current sources pump and pull equal amount of current keeping the control voltage constant. But due to the process variations and mismatch between the charge pump current sources these push and pull currents tend to be unequal hence changing the control voltage. The circuit in Figure 3.8 compensates for this error by making the outputs of the PFD zero as soon as both UP1 and DOWN1 go high without waiting for the delay due to the reset path [1]. The logic used is to simply AND the UP1 signal with the complement of DOWN1 signal and call it the UP of the PFD. We use a similar logic for the DOWN output. Hence UP and DOWN signals are never go high simultaneously, this compensates for the current mismatch problem in the charge pump.

In a DLL we are comparing the input clock and its delayed version hence for the very first pulse of the input *clk* the PFD would assert the UP signal causing the control voltage to rise, so as to make the delayed signal(corresponding to the first pulse) match the first edge of the input clock Figure 3.9. But the delayed clock will never match the first edge, so after a few cycles the circuit saturates. To compensate for this problem we place an extra DFF labelled DFF1 before the inputs of both DFF2 and DFF3 as shown in Figure 3.8. Placing DFF1 makes the PFD skip the first edge of *clk* and make the comparison start when the next edge of the inputs arrive as shown in Figure 3.9. The DFF1 flip-flop is reset initially along with the whole circuit.

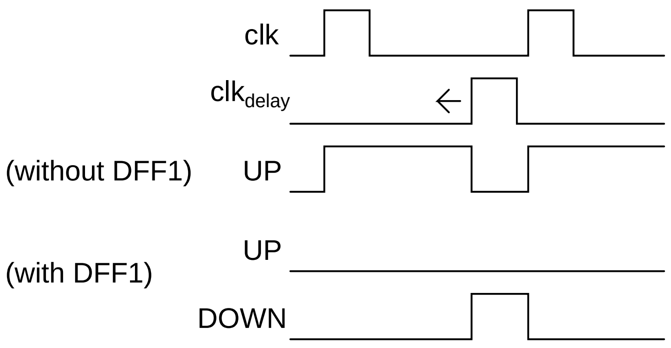
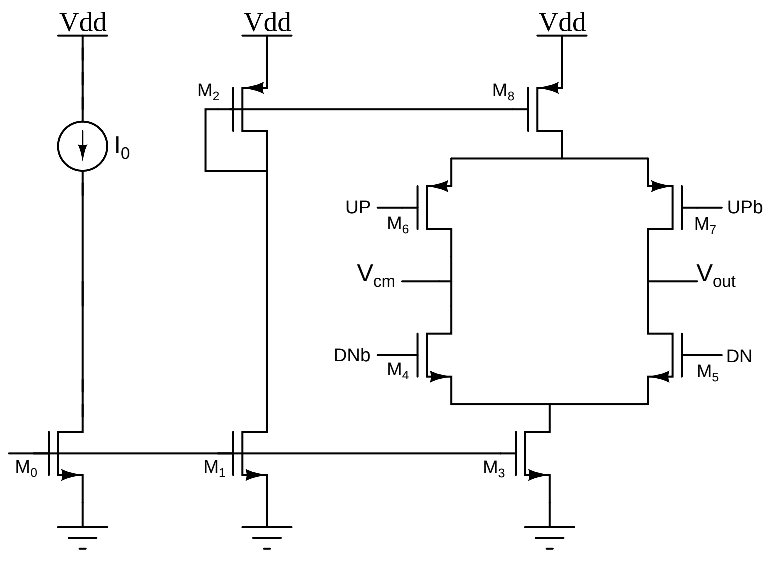


Figure 3.9: Skipping the first input clock edge

Charge pump

The charge pump just steers current into or out of the control voltage capacitor depending on the UP and DOWN signals from the PFD. If the UP signal is high then it means the delayed signal is lagging the input clock hence it pushes current into the capacitor raising the control voltage. Increasing the control voltage reduces the delay of the delay chain hence reducing the phase difference between the two clocks before the start of the next cycle. This goes on until both the signals are locked and have zero phase difference. Figure 3.10 shows the schematic of the charge pump which is similar to the one used in [1].



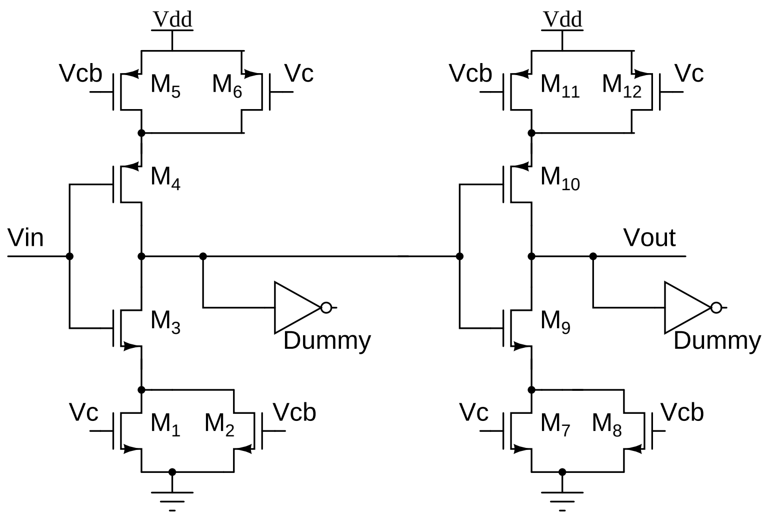
M_0, M_1	2.12um/1.5um	M_4, M_5	1.2um/0.12um
M_2	3.6um/1.5um	M_6, M_7	4.02um/0.12um
M_3	10.6um/1.5um	M_8	18um/1.5um
I_0, V_{cm}	10uA, 600mV		

Figure 3.10: Charge pump, [1]

Delay chain

The delay chain consists of 64 delay elements. Each delay element can be considered to be a buffer whose delay is controlled via a control voltage which in turn controls the on resistance of inverter transistors. The circuit of a single delay block in the delay chain is shown in Figure 3.11. The heart of the delay block has 2 inverters(the other 2 inverter symbols represent the dummy load) with extra transistors to control the currents though the PMOS and the NMOS transistor of the inverter. For the first part of the figure the delay is controlled by transistors M1 and M5 whose on resistance is determine by Vc and Vcb respectively. We also see that we have 2 more transistor in the first part of the circuit, these are M2 and M6. These are used in order to make the delay chain work even when control voltages fall much below the cut-off voltages of M1 and M5.

The overall architecture of the delay is also chosen so as to make sure that the rise and fall delays remain exactly the same. Maintaining equal delays is very



M_1 ,M_7 -(1.52u/120n)	M_4,M_{10} -(2u/120n)
M_2,M_8 -(280n/120n)	M_6,M_{12} -(280n/120n)
M_3,M_9 -(1u/120n)	M_5,M_{11} -(4.98u/120n)

Figure 3.11: A single delay block

important for the write control circuit used in the design. Since there are 64 delay blocks in the delay chain so even a slight mismatch between the rise and fall delays of a delay block may severely affect the final delayed output. If rise delay is more than fall delay then the input pulse width will decrease as it moves through the delay chain and it might finally die down before reaching the output. Similarly when fall delay is more, the input pulse width will keep increasing as it moves through the delay chain hence violating the 500ps criteria set by design specifications.

In the proposed design of the delay chain the rise and fall delay remains exactly the same unlike the design proposed by [2](ref: gunter thesis). For the case when the input rises the output delay is determined by the delay due to M1 and M11. Similarly when the input is falling the delay to the output is determined by M5 and M7. Since delay due to M11 is same as M5 and delay due to M1 is same as M7 hence the total delay is same in both case, that is

$$delay(M1) + delay(M11) = delay(M5) + delay(M7) \tag{3.7}$$

3.4.2 Generating 64 consecutive pulses

Now for the analog memory we need 64 pulses each of width 500ps. So ideally if we use a clock of period $500ps * 64 = 32n$ then we can get 64 pulses. In this case the delay of each buffer stage is $500ps$ so that after 64 buffer stages we will have a total delay of 32ns which is the clock period of the input clock and hence they should lock. But getting a delay of $500ps$ from each buffer is very difficult given we want a square pulse at the output with short rise and fall times. So instead of 1 buffer to get a delay of 500ps we use 4 buffers to get a delay of 500ps. In total this will require a total of $64 * 4 = 256$ buffers. But using 256 buffers will cause a lot of mismatch problems due to process and temperature variations and also the output becomes skewed when it reaches the last buffer. Also using 256 buffers will consume a very large area since along with the buffer we also need to keep a dummy so that each buffer stage is loaded similarly. So in oder to generate the write pulses using a low number of buffers we create a buffer with less number of output pulses(again all pulses of width $500ps$) and then multiplex them to create 64 pulses. In the proposed design a total of 16 write pulses are generated hence requiring $16 * 4 = 64$ buffer stages and an input clock period of $500ps * 16 = 8ns$. So every $8ns$ we generate 16 consecutive pulses each of width $500ps$. To multiplex we create 4 write enable pulses each switched on for 8ns and we AND these 16 pulses with the write pulses one by one. So initially when the first write enable signal is high the first 16 write pulses are generated and when the *write2* signal gets high after 8ns(and write1 goes low) then the next writes pulses are generated and so on till we generate 64 write pulses over 4 write cycles. The write enable pulses are generated using D flip flops. The overall schematic of the write pulse generator(without the DLL) is shown in Figure 3.12.

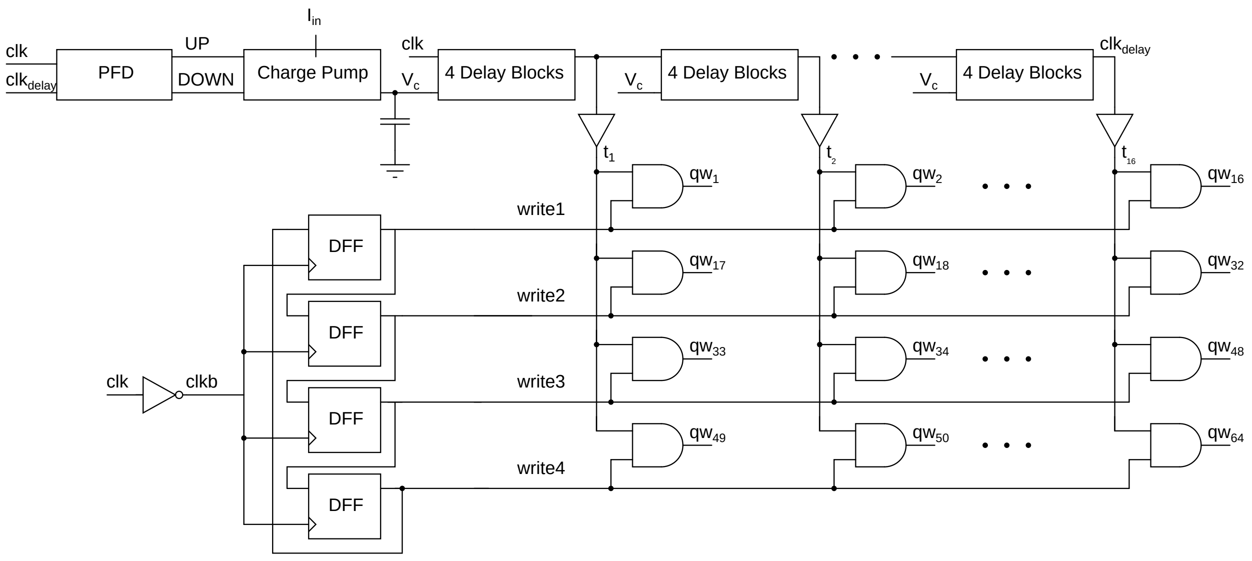


Figure 3.12: Multiplexing 16 write signals to generate 64 pulses

Here t(1:16) are the 16 pulses which are multiplexed over 4 cycles. And qw(1:64) are the final output 64 write pulses. Also write1, write2 and so on are the write enable signals used to select the t(1:16) pulses from the delay chain. The buffer stage has 4 buffers to generate one pulse(so these 4 buffer create delay of 500ps) , hence we use 16 of these 4 buffer stages to generate the t(1:16) pulses. Vc is the control voltage and Vcb is used to control the pmos transistors inside the charge pump.

CHAPTER 4

SAR ADC concept and design

4.1 Successive approximation ADC

A Successive Approximation ADC(SAR ADC) is an analog to digital converter which samples and converts an input signal using a method similar to binary search. It is a multiple step process and it compares the input with closer/accurate quantization levels after each step. SAR ADCs are useful because of their very high resolution and accuracy and extremely low power requirements. This combination of features makes these ADCs useful for applications like battery-powered instruments, industrial controls, bio-electronics and data/signal acquisition [7]. In earlier technologies the input sampling speed was a huge limitation but these days with high speed technology SAR ADC with speeds in the range of hundreds of MHz have been realized.

The basic architecture of ADC consists mainly of 3 parts. These are the DAC, the Latch or comparator and the SAR logic as shown in Figure 4.2. After the input is sampled the input is compared with the middle of the quantizer range. Depending on the output of the comparator the SAR logic sets the DAC output voltage to either the positive half of the quantizer range(Vref/2) or the negative half(-Vref/2). Again the comparison is done and the DAC output is again set to a closer quantizer value for comparison. This goes on till the input is compared with its closest quantization level as shown in Figure 4.1. Since for every input we need N internal cycles for N-bit resolution hence the internal clock for SAR operation is N times the input sampling clock. But a SAR ADC's speed limited by:

- The settling time of the DAC, which must settle to within the resolution of the overall converter, for example, $1/2LSB$.

- The comparator, which must resolve small differences between the input and the DAC output.
- The logic overhead

To summarize the primary advantages of SAR ADCs are low power consumption, high resolution and accuracy, and a small form factor [7]. Because of these benefits, SAR ADCs can often be integrated with other larger functions. The main limitations of the SAR architecture are the lower sampling rates and the requirements that the building blocks, the DAC and the comparator, be as accurate as the overall system.

This thesis contains the details of the design of an asynchronous 7-bit SAR ADC with input sampling speed of 100MS/sec on 0.13μm technology.

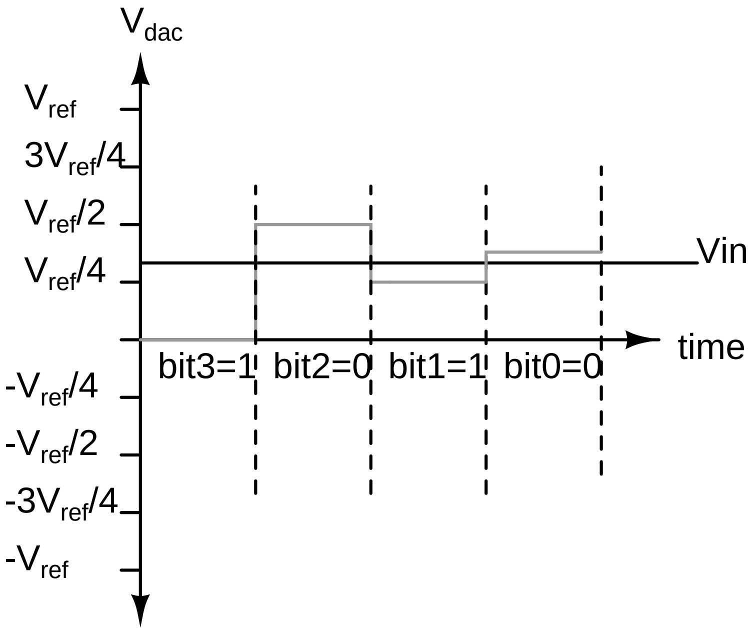


Figure 4.1: input comparison cycles

4.2 SAR architecture

The 7 bit SAR ADC designed is an asynchronous design as shown in Figure 4.2. The outputs of the comparator/latch delayed by a fixed time has been used as the internal clock in the design [8]. As it is a 7bit SAR ADC it needs 7 cycles to digitize one input sample. So if the input is sampled at 100 MHz then the internal asynchronous clock runs at $7 * 100 = 700$MHz. The quantizer range is from -Vref to Vref where Vref is $200mV$. The digital logic used during a conversion cycle in the SAR ADC starts with the 1^{st} comparison of the input done with the middle of the quantizer range that is 0. The digital code corresponding to 0 is 1000000. The output of this comparison decides the most significant bit. If the input is greater than zero then the first bit is 1 else it is 0. The 2^{nd} comparison is done with either positive or negative half of Vref depending on the output of the previous comparison. The output of the 2^{nd} comparison decides the 2^{nd} bit, similar logic holds for the rest of the bits. So if the input is greater than zero and smaller than $Vref/2$ then the first 2 bits of the output are 10 and say if input is less than zero and less than $-Vref/2$ then the first 2 bits of the output code is 00. Similarly we keep on dividing the quantizer range into halves in which the input is present. The halve to be considered depends on the output of the previous comparison.If the input is say $153mV$ then:

1^{st} comparison is done with $0mV$ => 7^{th} bit is 1 (input is greater than 0)
2^{nd} comparison is done with $100mV$ => 6^{th} bit is 1
3^{rd} comparison is done with $150mV$ => 5^{th} bit is 1
4^{th} comparison is done with $175mV$ => 4^{th} bit is 0 (input is smaller than $175mV$)
5^{th} comparison is done with $162.5mV$ => 3^{rd} bit is 0
6^{th} comparison is done with $156.25mV$ => 2^{nd} bit is 0
7^{th} comparison is done with $153.125mV$ => 1^{st} bit is 0

Hence the final digitized output is 1110000

The overall architecture of the ADC looks as shown in Figure 4.2

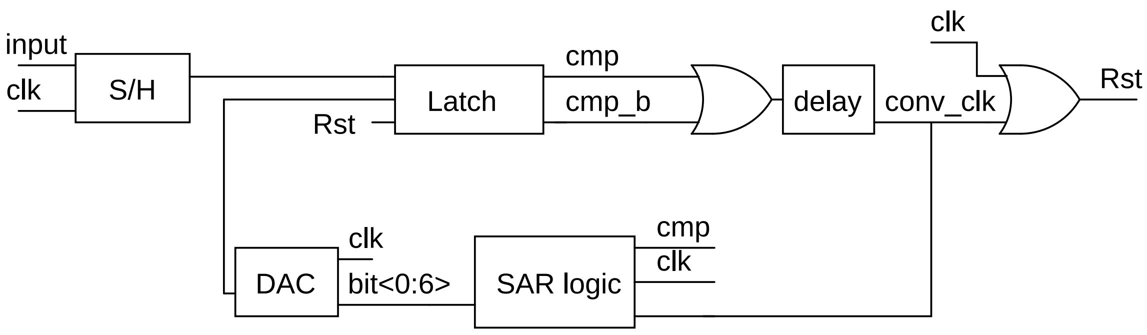


Figure 4.2: Architecture of the proposed SAR ADC

4.3 SAR ADC functionality

Initially the input signal(which are basically the samples read out from the analog memory) is sampled on the capacitor array of the DAC. In the actual design the sample and hold circuit shown separately in the Figure 4.2 is integrated with the DAC so the input gets directly sampled on the DAC capacitors. During this sampling cycle the *clk* signal is high. This signal also resets the value of the SAR logic block(digital logic block) to 1000000 which corresponds to a comparison voltage of $0V$(middle of the quantizer range) for the initial comparison. After the input has been sampled we start the conversion operation. First we compare the input signal with $0V$ DAC output using the *Latch*. The outputs of the comparator *cmp* and *cmpb* which are complement of each other rise to supply($Vdd = 1.2V$) and ground. These outputs are then *OR*ed and passed through a fixed delay(which decides the operating frequency of the SAR ADC). This delayed OR output *conv_clk* and also *Rst* is used as the asynchronous clock [8]. Since the OR of the outputs of the comparator will go high after every comparison they have been used as the internal clock for the ADC. The *Rst* serves as the reset for the latch. As soon as *Rst* goes high after some fixed delay the values of comparator outputs are reset to ground which itself deasserts the *Rst* again after the same fixed delay(this is how *Rst* acts like the internal clock). Also during the time *conv_clk* is high the new digital code for the next comparison is generated and the DAC converts this digital code to a reference voltage for the next comparison. All this operation is done before the *Rst* is again set low (this was because of the resetting of the latch) hence limiting the delay of the delay block or limiting the input sampling frequency. So after the new DAC output is generated and the *Rst* is deasserted the next comparison starts. Again the same cycle repeats and this takes place 7 times with each comparison deciding each bit of the final 7 bit output. After the seven cycles are complete the next input is sampled and digitized and we go over the process described above once again. The next chapter will discuss about the circuit level implementation of each of the SAR ADC blocks.

CHAPTER 5

SAR circuit design

In the last chapter the concept of the SAR ADC was discussed and how it was able to digitize an analog signal sample in as many steps as the number of qunatization bits. Also the overall architecture and the working of the proposed SAR ADC design was discussed. This chapter contains details of each of the SAR ADC blocks namely the *SAR digital logic*, the *DAC* and the *Latch*.

5.1 SAR digital logic

The digital logic to implement the SAR logic is done using a cascade of DFF(D-flip flops), 2 arrays of 7 DFF each were used. Initially for the first comparison we set the digital output (which is then converted by the DAC and compared with the input signal) to 1000000 which corresponds to zero. This is done by the *clk* signal, that is the digital logic is reset when the input is being sampled on the DAC capacitors. The schematic is shown in Figure 5.1

In Figure 5.1 the *conv_clk* is the asynchronous clock which was generated by the delayed *OR* of the latch outputs. *cmp* is the output of the comparator(it is 1 when input sampled signal is greater than the DAC output and zero otherwise). *out0* to *out6* are the outputs of the digital logic block(which are later converted by the DAC to voltages for comparison with the input). After every cycle we set the bit *s0* to *s7* in a sequence using the top array of DFF. We also reset the *s*bit which was set in the previous cycle and assign the output corresponding to that bit to the output of the comparator.

At the start we set *s0* which causes the output to be 1000000. Then for every subsequent cycle of conversion we set the next digital output bit to 1 and set the previous bit to the value depending on the result of the comparator. For example after 3 comparisons(the 3rd rising edge of conv_clk) we set *s3* and set the 3rd output bit that is out2 to the result of the 3rd comparison. The value of digital block *out[0:6]* after 7 cycles of conversion gives us the final digitized output corresponding to that input sample.

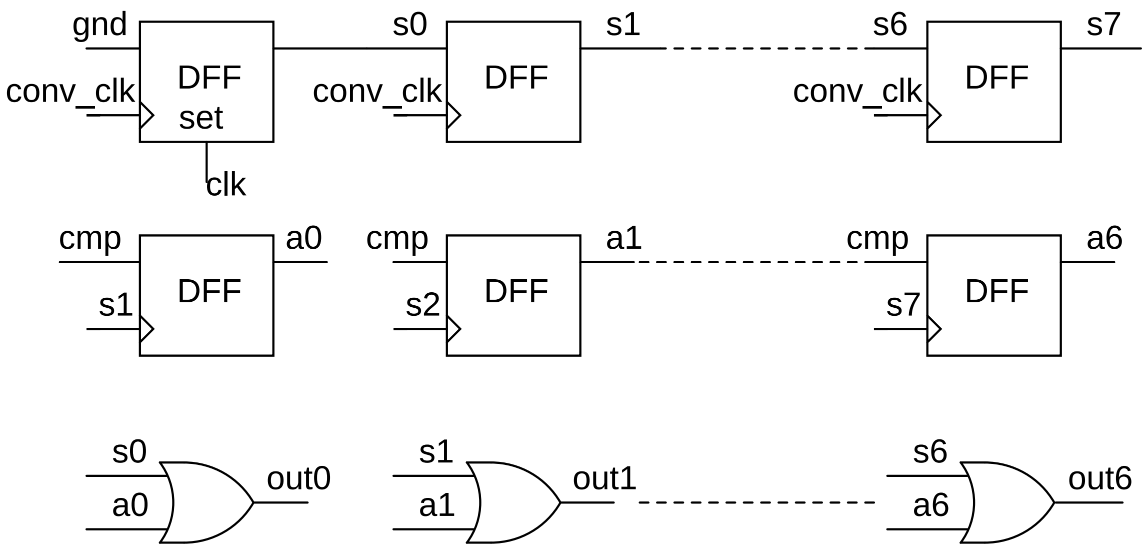


Figure 5.1: SAR digital logic

5.2 Latch

Latch is basically a comparator which compares two input signals. Since the ADC design is asynchronous the latch was designed to only have once reset signal and the remaining two input signals and the two output signals. The schematic with the sizes of the transistors used is shown in Figure 5.2. Here the *Rst* signal is the reset signal and *Rstb* is its complement. The *Rst* signal was generated by the delayed OR of the latch outputs as discussed in the previous section. Let us now take a simple case say initially the Latch was reset and the outputs were grounded. If $ip > im$ then the current through M1 will be more which will pull down the drain of M3. This will start to turn on the first inverter and its output(om) goes down hence pulling up *op*. As *op* goes up the NMOS of the 1st inverter further switches on pulling *om* further down which in turn pulls up *op* further. Finally *op* goes to Vdd and *om* goes to ground. The input referred offset due to the Latch has not been take into account during the design but it needs to be compensated by the help of a preamplifier which should be placed before the latch.

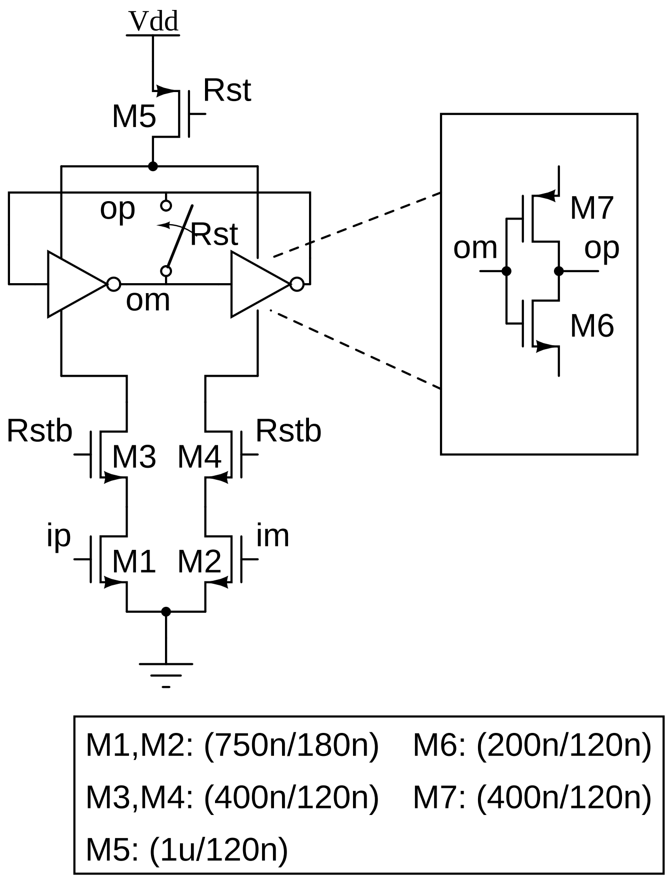


Figure 5.2: Latch with reset

5.3 DAC(digital to analog converter)

For the DAC the set up and down approach [9] is used which has the potential to consume 75% less energy compared to conventional SAR DAC techniques [10][11] . Figure 5.3 shows the schematic of the DAC designed for a 7-bit case.

Initially the input signals *vip* and *vin* are sampled on the top plates of the capacitor and the bottom plate of the capacitors are charged to Vref. Then the switches connect them to either vref or ground depending on the digital conversion output after each cycle. The switches used to realize the input sampling switches

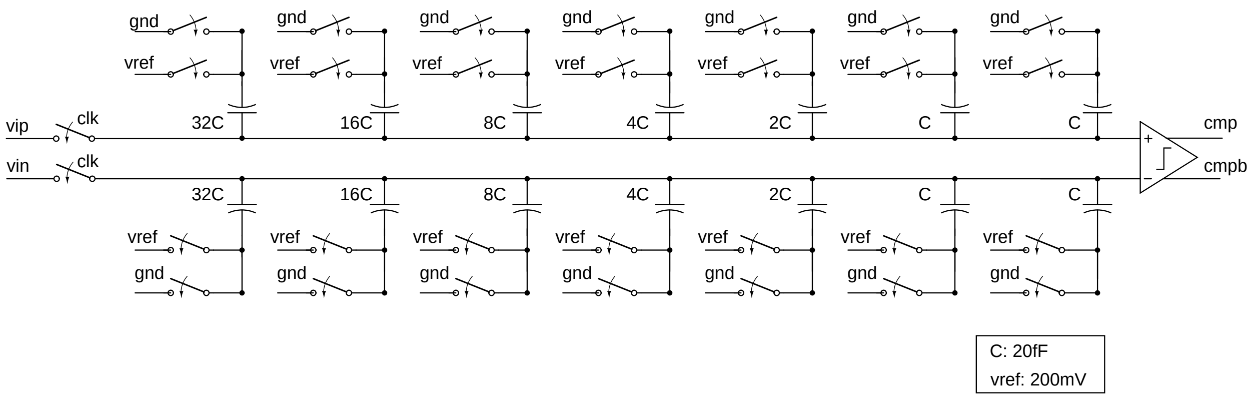


Figure 5.3: 7 bit DAC

were boot-strapped as shown in Figure 5.4. Boot-strapping reduced input dependent charge injection and decreased the harmonic distortion. Using the boot strap switch when *clkb* was high the capacitor was charged to Vdd . When *clk* goes high to track the input then the gate of the NMOS was at $vin + Vdd$ hence the gate−source voltage of the transistor was $(Vdd + vin) - (vin)$ which is Vdd and remains so irrespective of the input signal. Hence when this switch turns off then the charge injected on the capacitor is the same for all input samples.

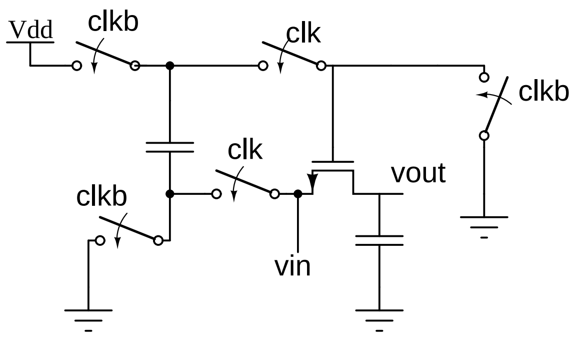


Figure 5.4: Boot-strap switch for input sampling

For the first comparison all the capacitors are connected to Vref and we compare the input with zero, that is *vip* is compared with *vin.* Now if say the input is greater than 0 then for the next comparison(the digital output corresponding to which is 1100000) the top left most capacitor(32C) is connect to gnd from vref. This reduces the value on the line on which *vip* was sampled by $vref/2$. So effectively we now compare $vip{-}vref/2$ with *vin* or to put it simply the input signal is compared with $vref/2$. After every step the switch logic changes the DAC output voltages reducing the range with which the input is compared hence justifying the SAR functionality. The DAC architecture following the set up and down approach for a 3-bit case is shown in Figure 5.5. For conventional techniques the average switching energy [12] for an n-bit case is

$$E_{avg,n_bit} = \sum_{i=1}^{n} 2^{n+1-2i}(2^i - 1)CV_{ref}^2 \tag{5.1}$$

The average switching energy [12] for n-bit case with the set-up and down approach is

$$E_{avg,n_bit} = \sum_{i=1}^{n-1} (2^{n-2-i})CV_{ref}^2 \tag{5.2}$$

The energy consumed in each switching step is also written above each of the arrows in Figure 5.5 and on an average this comes to about 75% less than conventional switch capacitor techniques.

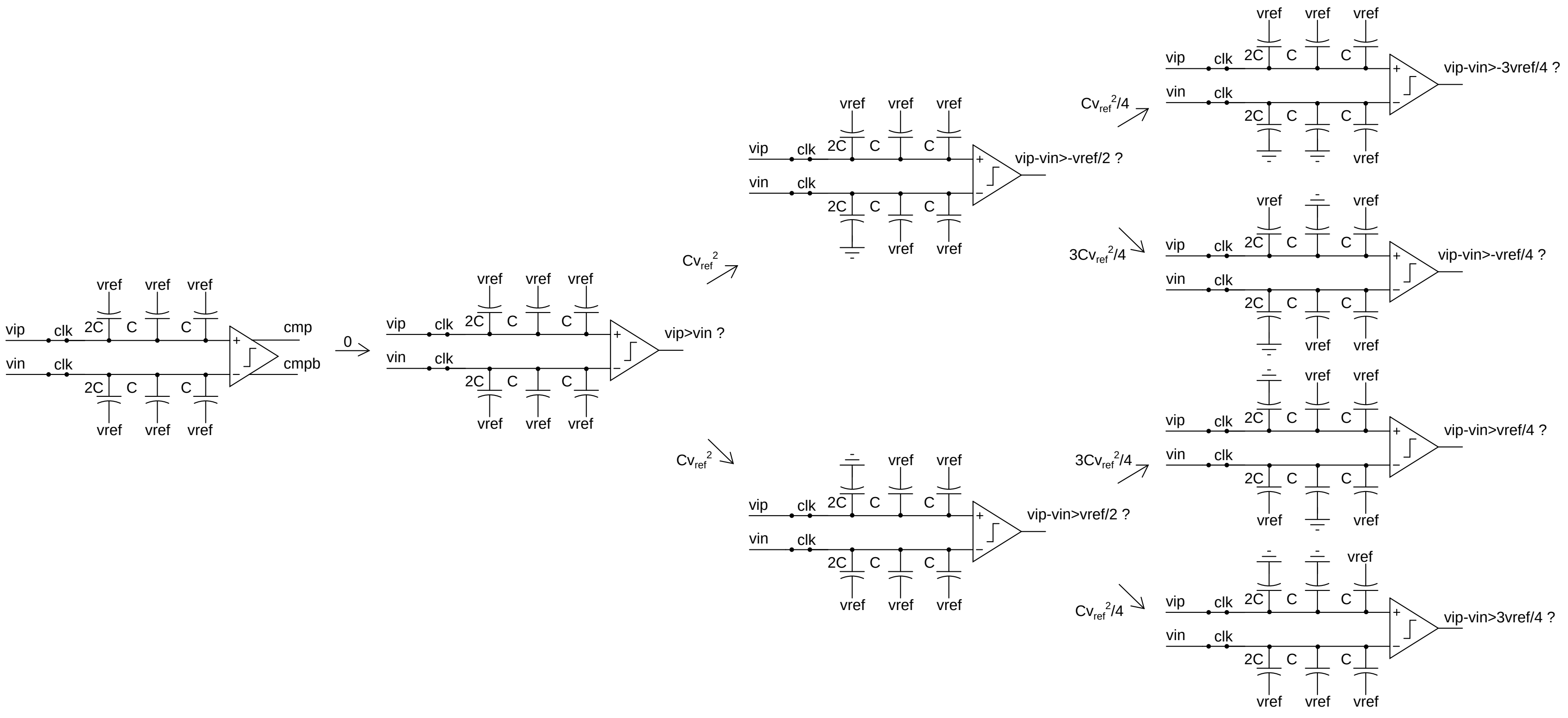


Figure 5.5: A 3-bit DAC using the set-up and down approach

CHAPTER 6

Simulation results

6.1 Layout

The analog memory proposed in the previous chapters was then laid out and the performance was analyzed. The floorplan of the modulator is shown in Figure 6.1. As mentioned in Chapter 3 we saw how parasitics between different wires caused error in the read out value. Hence layout was done so as to minimize the parasitics mentioned in Chapter 3. The total area of the layed out memory is 530 μmX515 μm. Owing to the way the write signals were generated that is 16 main pulses multiplexed over 4 cycles, we dedicated one column of memory cells to one cycle of 16 write signals. So totally for 4 different write enable signals or 4 cycles of the 16 main write pulses we created a 16X4 array of memory cells. Enough space was given between the memory blocks so as to route the read signals. The first 16 read signals were routed to the first column of memory cells and next 16 to the 2nd column and so on. The DLL which generates the control voltage to control the delay of the delay chain was put separately. The empty spaces were also filled with bypass capacitors. The opamp occupies 37.5 μmX90 μm.

6.2 Simulation result

The main reason for the noise and distortion at the output is due to the inaccuracies in the write signals. Since a 7-bit ADC was used after the analog memory hence we would like the total distortion introduced by the analog memory to be less than the distortion due to the ADC. If the noise due to the quantizer is considered to be white noise then ideally we expect the SNR for an ADC to be

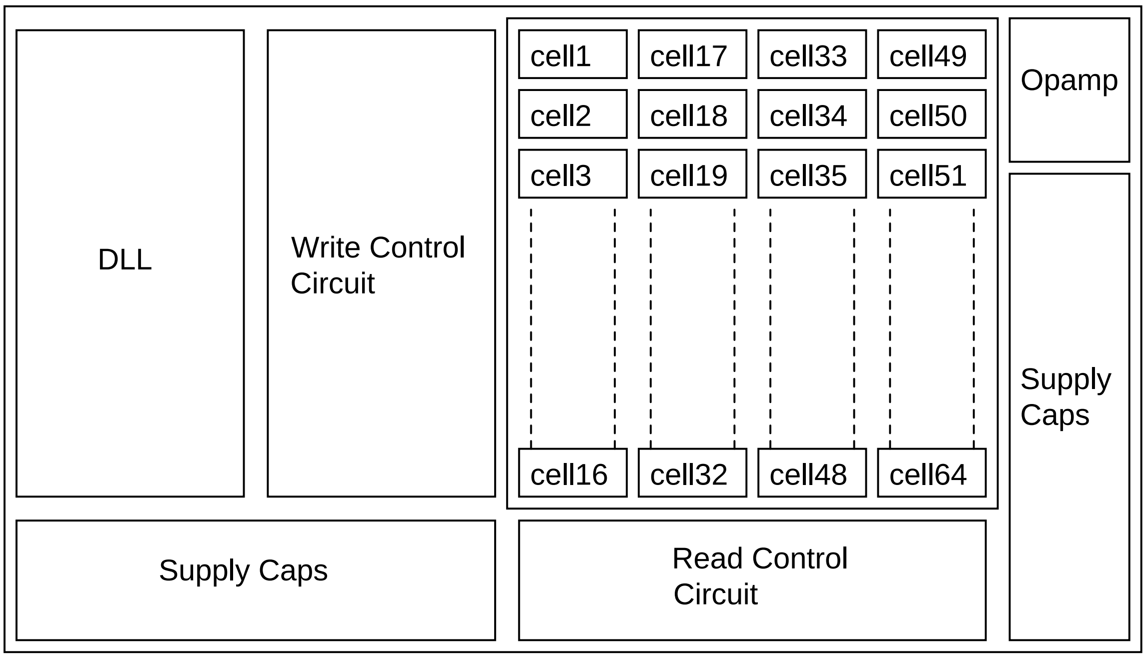


Figure 6.1: Floorplan of the design

$6N + 1.73 = 43.73$dB where N is the number of quantization bits. The layed out version of the analog memory for an input at 31.25MHz and an output load of $1.5pF$ was found to have a SNDR of 61dB. The schematic had an SNDR of 67.34dB. Both the simulations were for the *tt* corner. The output of the fully layed out version for different input frequencies is shown in Figure 6.2. The SNDR for inputs of 31.25 MHz, 93.75 MHz and 281.25 MHz are 61dB, 52.1dB and 42.7dB respectively. The output plot for an input pulse (obtained by giving a step to a resistor and capacitor in series having a time constant of 5ns) is shown in Figure 6.3, the first sample has a high error as it corresponds to the very fast rising edge of the pulse. The maximum power consumed by the analog memory is 2.6mW during the write phase. The switches in the analog memory mainly the *write* switch and the *qw* switch of each memory cell were sized so that their distortion was much less than the overall distortion. To find the reason for the distortion in the laid out design inaccuracies in the write signal was simulated using Matlab. For a sinusoid sampled at 2GHz if the sampling instant inaccuracy varies linearly from 0 to 3ps for the 16 main write pulses, the output plot compared with the layout is shown in Figure 6.4. This explains why in the proposed design as well the inaccuracy of the sampling time causes close to same amount of error.

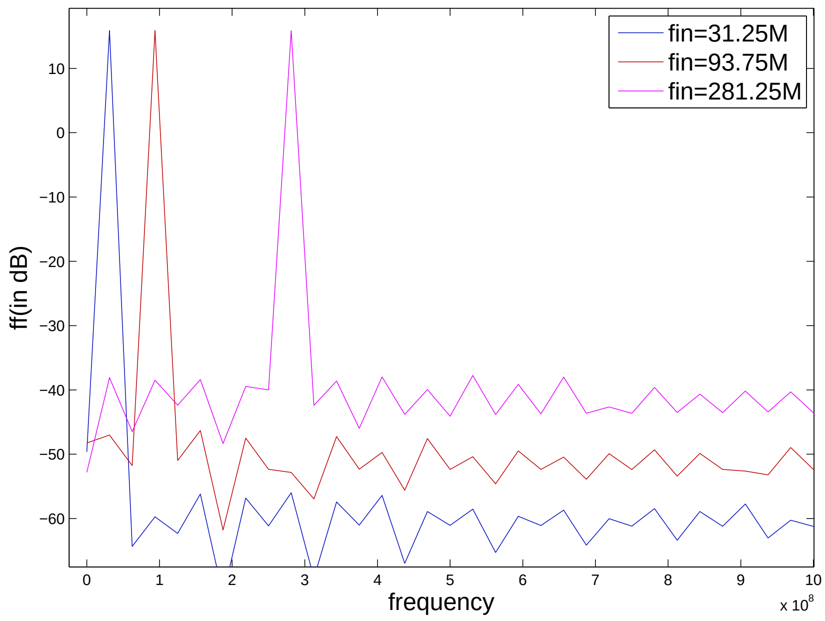


Figure 6.2: FFT of output samples for different input frequencies

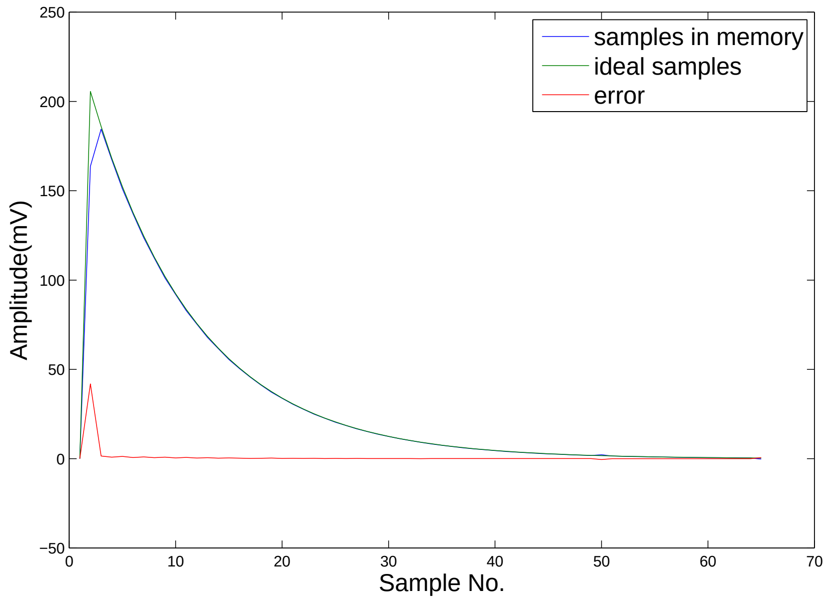


Figure 6.3: Output samples read out for an input pulse

For the proposed 7 bit SAR ADC when simulated for an input sinusoid at 2.8MHz and input sampling frequency of 50MHz the total SNDR comes out to

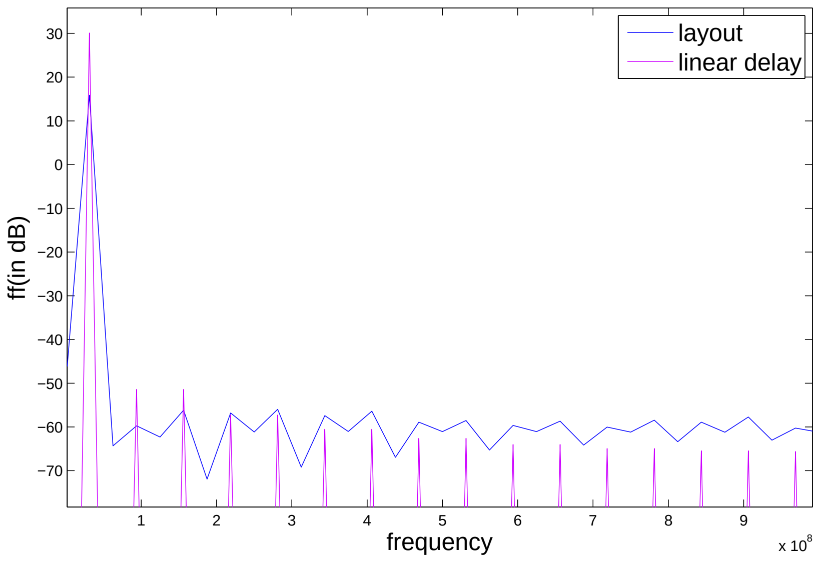


Figure 6.4: Effect of inaccurate sampling compared with the layout

be 44.07dB which is comparable to the expected value of 43.78dB. The fft of the output digitized value is shown in Figure 6.5. For input at $fs/2$ the SNDR is 43dB and when we use an ideal input sampling switch the SNDR comes out to be 44dB. The SAR ADC consumes 262 μW of power from 1.2V supply.

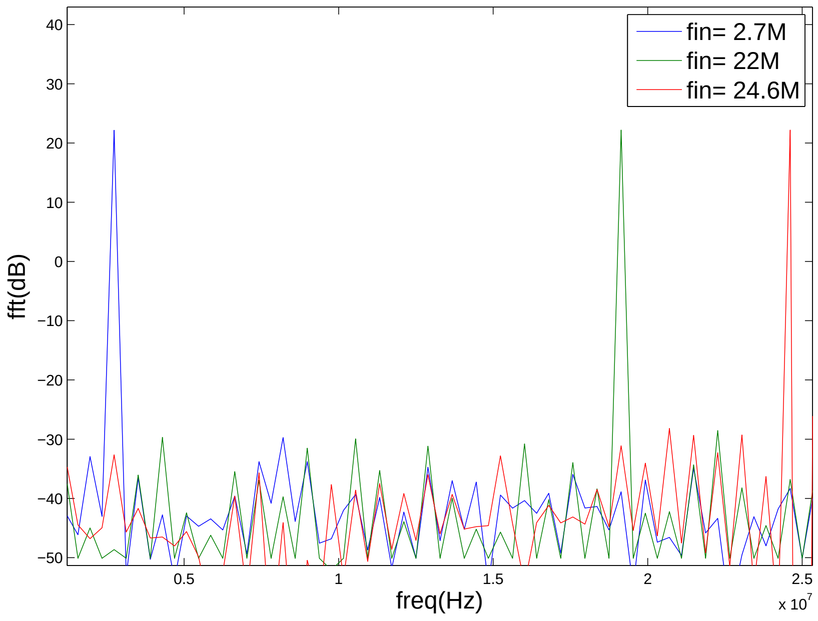


Figure 6.5: FFT of output samples for different input frequencies

Part II

Testing of an 800 MHz $\Delta\Sigma$ Modulator

CHAPTER 7

Introduction

Continuous time $\Delta\Sigma$ modulators have become popular over the years for wireless and signal processing applications. They are high speed and consume low power. But another important property that they possess is noise shaping. CT(Continuous time) $\Delta\Sigma$ modulators filter out the quantization noise in the signal band thus making the quantizer look like having more number of bits(hence better quantization) for quantization. Hence we can as well design a simple 1 bit quantizer and make it look like 4 or 5bit quantizer by the help of the noise shaping property. Also CT time $\Delta\Sigma$ modulators have an inherent anti aliasing property thus saving the need of a explicit high order anti-alias filter which saves both power and area. In a given technology the sampling frequency of a CT $\Delta\Sigma$ modulators is limited by the excess loop delay present in the loop because of the quantizer and the DAC. Most of modulators have ELD(excess loop delay) compensation but they are mostly valid for ELD less than one clock cycle [13][14]. This again limits the sampling frequency. Thus to realize higher sampling frequency we need to design the modulator which can compensate for much higher ELD. The chip [2] which is tested as part of this thesis work employs one such technique which can compensate for ELD between one to 2 clock cycle delays [15]. This led to a very high sampling frequency for a 4-bit $\Delta\Sigma$ modulator designed on 0.18μm technology. The chip with as $OSR = 25$ was tested to work at 800,MHz and the DR was 75dB the SNR_{max} was 67dB and the $SNDR_{max}$ was 65dB. The sampling rate of 800 MHz is the highest reported for a 0.18μm technology. The bandwidth is 16MHz/32MHz and it consumes 47.6mW from a 1.8V supply. Also a new high speed data acquisition technique to capture the ADC data directly from the FPGA(field programmable gate array) is also discussed along with its potential applications for testing with other similar chip test setups.

Organization

This part of the thesis has been organised as follows

Chapter8 discusses the whole test setup used to test the chip along with the measurement results. Finally a new data acquisition technique using Chipscope tool thus replacing the need of a logic analyzer is described.

The details of the chip which was tested can be found in [2] and also Appendix A, which discusses the effect of excess loop delay on a delta sigma modulator and also discusses commonly used techniques to compensate them. Appendix A further discusses the compensation technique used in the chip being tested and also gives an overview of the chip architecture.

CHAPTER 8

Test setup and measurement results

8.1 Test chip

The pins details of the test chip is shown in the Figure 8.1. The chip has 48pins and occupies 36.77mm^2. A snapshot of the chip is shown in Figure 8.2.

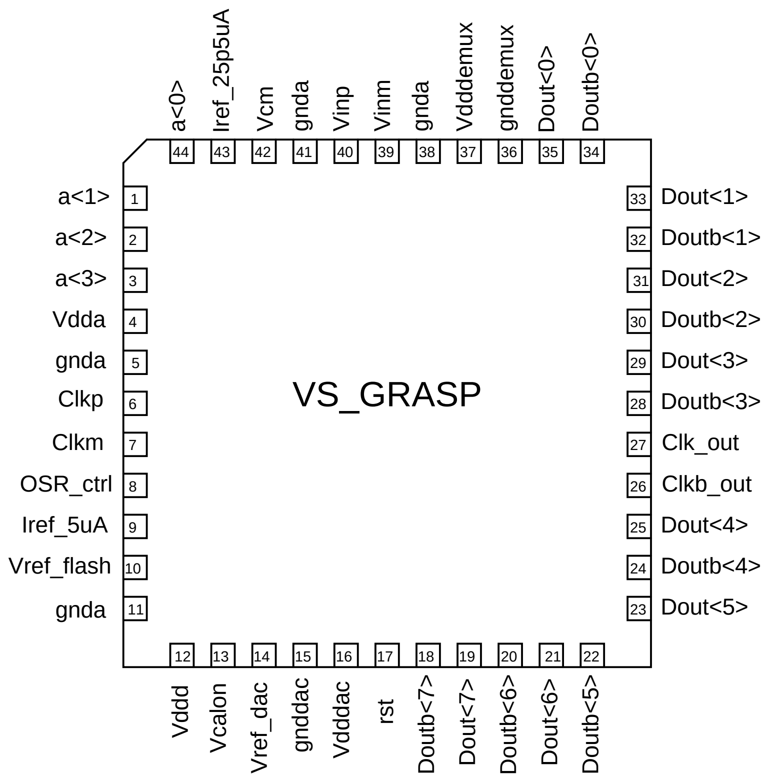


Figure 8.1: Chip pin details, [2]

8.2 Test setup

The top level block diagram of the test setup is shown in Figure 8.3 .A signal source(Agilent 33250A) drives an input tone to a passband filter which filters out

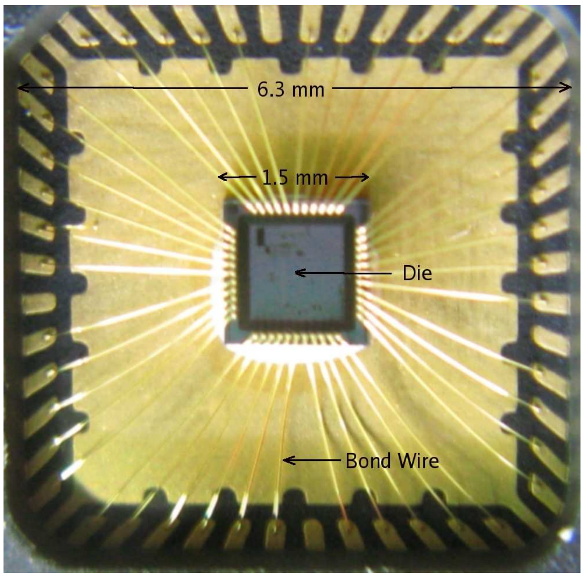


Figure 8.2: Chip used for testing, [2]

the harmonics of the signal source. This filtered signal is then passed through a balun transformer(North-Hills 100KHz-20MHz balun) which converts this single ended signal to a differential signal. The clock signal is generated by a signal source (Agilent E4422B), which generates low-jitter sine wave at 800 MHz. The

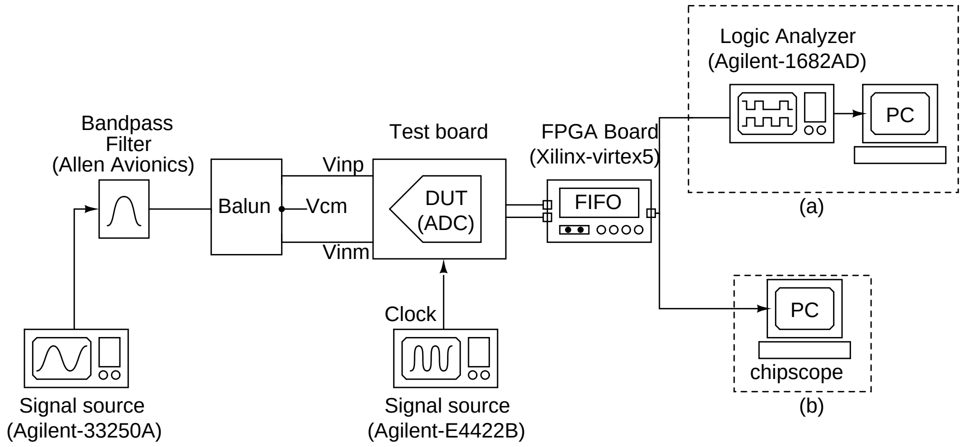


Figure 8.3: Test setup using either (a)logic analyzer (b)Chipscope

clock and the signal is then sent to the test board containing the ADC. The ADC outputs are then passed through LVDS buffers and are written on an FIFO at $f_s/2$. The FIFO is implemented on an FPGA(Xilinx Virter 5 board) and it stores the 8bit ADC outputs on its FIFO. This data stored in the FIFO is then read out to a Logic Analyzer at a much slower speed. Since the Logic analyzer speed is limited to around 200MHz hence we first have to write data to a FPGA rather than capturing it directly from the LVDS buffer outputs. The logic analyzer is a costly and scarce instrument hence a new way of capturing data was used by the help of a tool called Chipscope provided Xilinx. Details of test setup using chipscope will be discussed in the last section of this chapter. Figure 8.4 shows a snapshot of the test set-up.

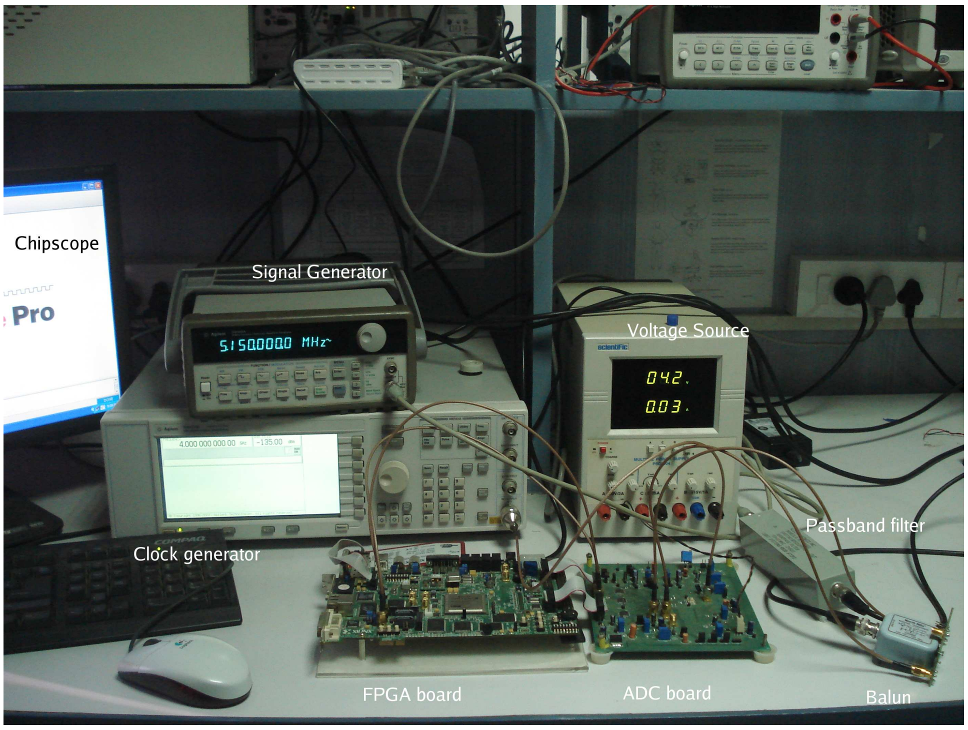


Figure 8.4: Test setup

8.3 Test board

Various test board were used for testing. Initially a 2 layer board(Board1) with socket chip connection was used. After the functionality was verified a new test board(Board2) was used on which the chip was directly soldered instead of using a socket connection also the ground plane of the part of the board with the modulator was separated from the ground plane of the LVDS part of the board. Its snapshot with labels is shown in the Figure 8.5. The measurement results presented in the next section contains readings taken from this board. The dimension of the board was 14.5cmX16.5cm. Finally another new board was designed containing the ADC chip on one board and the other board containing the LVDS buffers; both these board will be collectively referred to as board3. For board3 the extra circuitry which allowed us to control DAC supply voltages,flash ADC supply and common mode voltages on board2 was excluded. During the course of testing it was also observed that the packaging of the chip seemed to have an affect on the output as packages by *SPEL* were not running at 800MHz on board3.

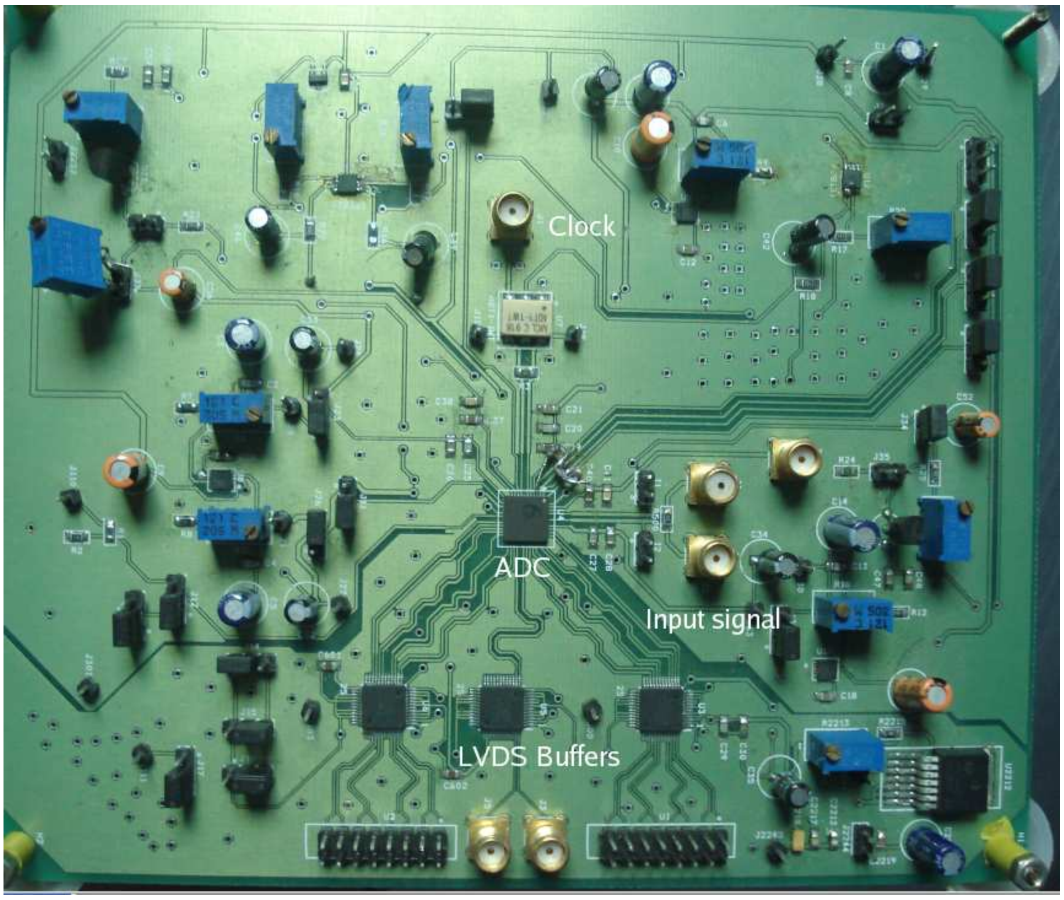


Figure 8.5: Board used for the results

8.4 Measurement results

The chip consumes $47.6mW$ of power from a 1.8V supply. Figure 8.6 shows the idle channel response of the modulator.

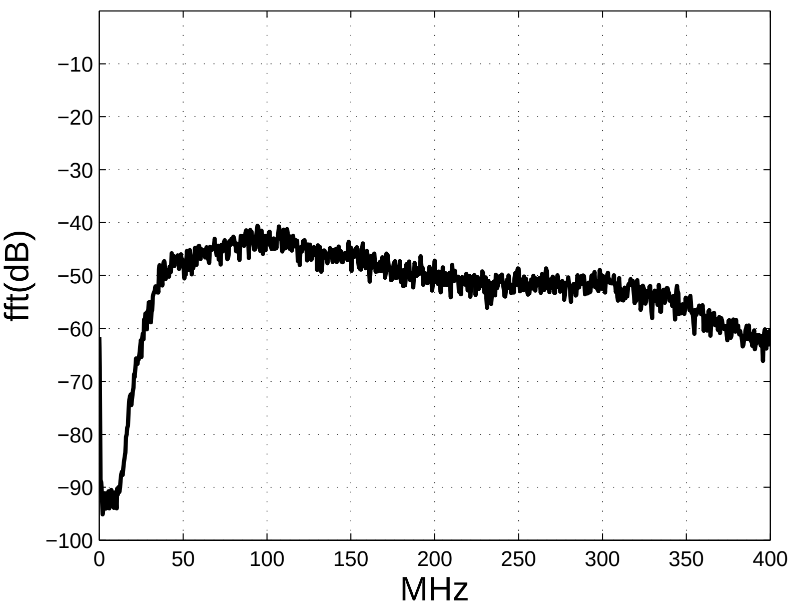


Figure 8.6: Idle channel noise

For an input of -1.9dBFS the output of the modulator is shown in Figure 8.7, the SQNR is 66.6dB and SNDR 65dB. The tone at $fs/4$ is due to the input reference subtractor in the flash ADC. As discussed in the previous chapter the DAC calibraton scheme used in the design significantly reduces the distortion at the output and when compared to a case with calibration turned on the plots look like as shown in Figure 8.8. The harmonics with calibration on are about 77dB below the input tone.

The plot of SNDR for different input amplitudes is shown in Figure 8.9. The measured dynamic range is 75.5 dB for OSR of 25.

For the case with OSR of 12.5 the plot for a 5MHz input at -6.7dBFS is shown in Figure 8.10. The SNR and SNDR were both 57dB. Also the plot of SNDR over different amplitudes is shown in Figure 8.11. The measured dynamic range is 64dB.

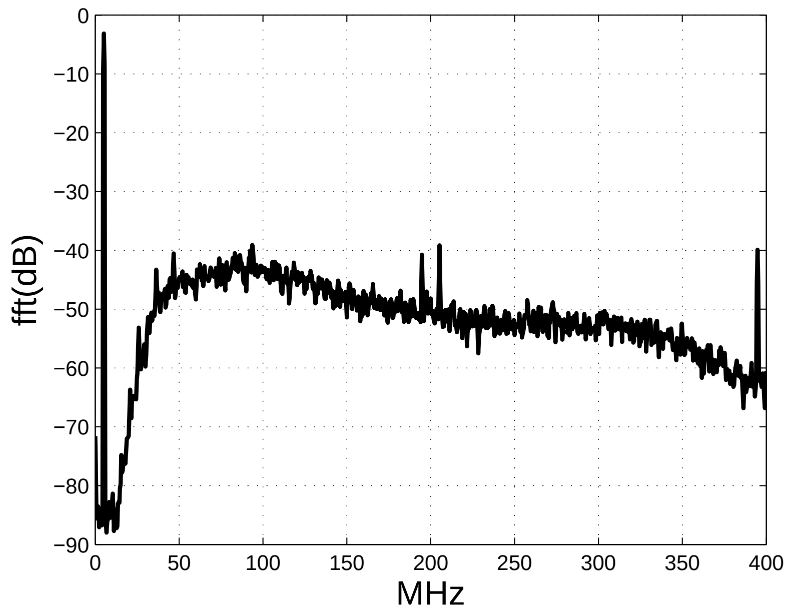


Figure 8.7: Output for OSR of 25 and input at 5.2MHz

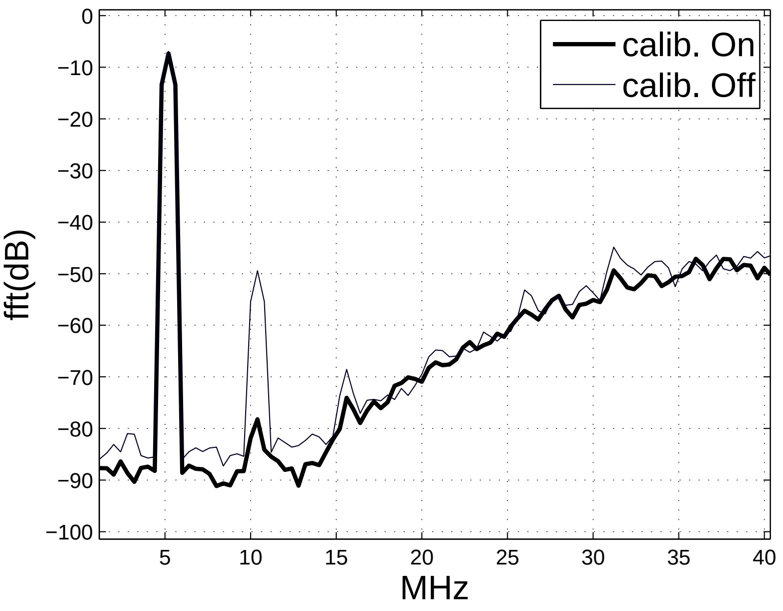


Figure 8.8: NTF with and without calibration

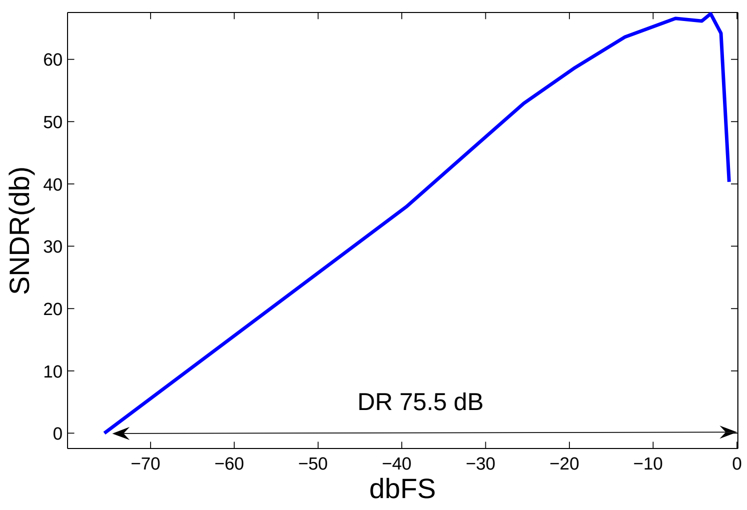


Figure 8.9: SNDR vs input amplitude for OSR= 25

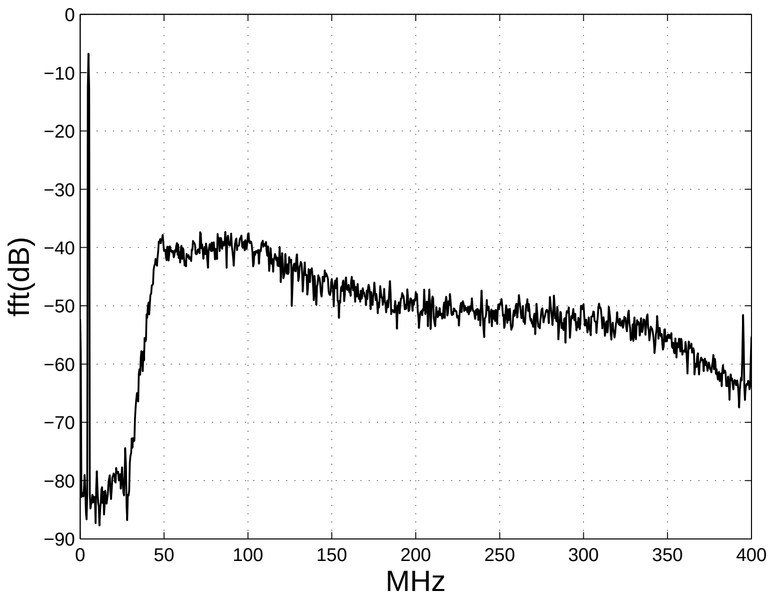


Figure 8.10: Output for OSR of 12.5 and input of 5MHz

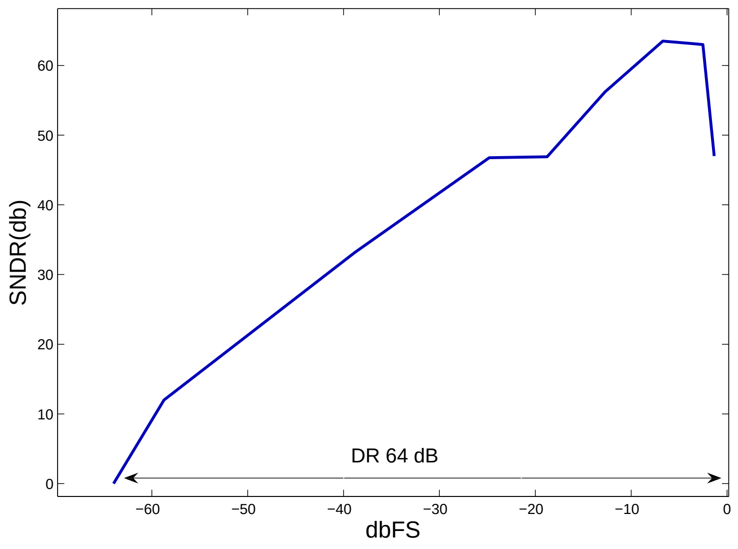


Figure 8.11: SNDR vs input amplitude for OSR= 12.5

8.5 Using Chipscope for high speed data acquisition

In the earlier section,we discussed how we were using the logic analyzer along with the FPGA board to capture the data from the ADC. Since logic analyzers are scarce and costly instruments so there was a need to come up with a way to directly capture the data from the FPGA itself. Also capturing data into the logic analyzer and then transferring it to the main machine was also a time consuming process. By using a tool called Chipscope [16] which is provided as part of the Xilinx ISE package it is possible to capture data at high speed on the FPGA itself. It can also capture data at very high speeds(the speed of the design on the FPGA) which is much higher than the maximum sampling rate possible via a commercially available logic analyzer.

8.5.1 Chipscope overview

Chipscope is a tool provided by Xilinx for debugging on board FPGA signals. It samples the signals needed to be analyzed and stores them on the FPGA FIFO.

These are then read to the Chipscope GUI on your host computer. Chipscope consists of 3 modules which can be added to an FGPA code they are:

- ICON(integrated controller): It is a module which acts as the communication link betwen your PC and the chipscope module on the FPGA board.
- VIO(Virtual Input/Output): This module is used to drive signals and monitor them in real time on your FPGA. It is very useful for debugging purposes and can be treated as virtual push buttons.
- ILA(Integrated Logic Analyzer): A module that is used to trigger and capture the desired signals. It is like a digital logic analyzer or an oscilloscope which we can use for debugging.

For our design we will be needing only the Chipscope ICON core which is used in all designs and the ILA core. We currently are not using the VIO core but it can be used to drive our FPGA signals for example the reset, read and write signals on our FPGA board. The number of control ports of the ICON core must match the number of other chipscope modules(VIO or ILA) used in a design.

8.5.2 Using Chipscope in the design

To start using chipscope we first need to create all the modules that we need for analyzing and capturing the data. Then after creating these modules we add them to the top level of our verilog design file which we then burn on the FPGA(a Virterx5 board was used for all the measurements). We will first of all need the Chipscope ICON core(which is necessary for all the designs). Then since we only want to capture and analyze data hence we need an ILA core and we don't need any VIO core. The number of ILA cores depends on the number of signals we want to capture which range from a few bits in some designs to hundreds of bits in some other designs. Each ILA core has the capability to sample a 32 bit signal [17]. It is like storing 32 bits samples in a FIFO and each FIFO acts like one ILA core.

For our delta sigma modulator we need to only capture 8 output digitized bits. We also need to capture one read signal which is the output from our top level as it will act as the trigger signal for our chipscope module.

Once we have finalized the specifications we start generating the cores [17]

- First, we start generating the cores by the help of Xilinx Core-generator tool present in the Xilinx ISE tool. We then start creating an ICON core. After selecting the correct device we select the number of ports which is one for our case(as we will be needing just one ILA core). We then click next.
- We now generate the ILA core in a similar fashion. We select the number of bits to be analyzed which is 9 bits(8 outputs along with a read signal trigger). Then we set the number of trigger bits which we set to all the same 9 bits which are to be analyzed(we will have the option of selecting which bit to trigger on in the chipscope GUI). We also set the box which says that data is same as trigger.
- Finally, after creating all the cores we now instantiate all of them at the top level design in a way similar to how we instantiate a simple verilog module. We map the data pins of the ILA core(which will be captured by Chipscope) to the signals we want to sample in our top level design(to map analyzer signals to signals in other sub modules we need to bring them out as pins to the top level). We also need to map the clock of the ICON core. This is the clock on which all the data will be sampled by the ILA core. After the mapping and instantiation is complete we generate the bit file and then burn it on the FPGA.

All the above steps need to be carried out only once in order to burn all the chipscope cores on the FPGA. Once the all the module are working on the FPGA board we then start to capture data. To do this we start the Chipscope analyzer software provided by the Xilinx ISE toolkit. In the console we select Xilinx USB cable to connect the machine to the FPGA via its USB port. After the console detects all the chipscope modules in our FPGA, then all the signals that were declared in the ILA module will appear. We then open the trigger module and select the 9th bit or the read signal bit as our trigger and set its trigger condition to *1*, such that the system samples the FPGA signals when the read switch is turned on at the sampling clock as specified in the ICON core [17].

After doing all the above we save the whole project. Next time we just open the project click on the trigger button and then set the read switch on the FPGA board to start capturing data. The captured data can then be exported to matlab readable file.

CHAPTER 9

Conclusion and Future Work

9.1 Conclusion

A high speed analog memory was successfully designed and layed out as part of this thesis work. The performance degraded significantly in the layed out version compared to the schematic owing to the parasitics present in the layout. But the circuit was able to perform its functionality of sampling and storing a pulse and keeping the error within the required specifications. Also in order to digitize the stored pulse a low power SAR ADC was also designed at the transistor level. The performance of the standlone ADC was found to be close to the performance of a ideal 7-bit SAR ADC.

The second part of the thesis contained the testing results of a high speed and high resolution continuous time $\Delta\Sigma$ ADC. The testing work was done on various boards and the best performance was summarized in the last chapter. The chip using a sampling rate of 800 MHz had a dynamic range of $75dB$ and $64dB$ for a bandwidth of 16 MHz and 32 MHz respectively. The sampling rate of 800 MHz was found to be the highest reported for 0.18μm technology.

9.2 Future work

9.2.1 Analog memory

Since the schematic was found to have the desired performance, hence only changes in the layout needs to be made in order to make it match to the schematic. Although various ways of laying out the memory was tried but the distortion introduced due the parasitics was tough to overcome. The main reason for the poor performace was initially thought to be due to the inaccuracies in the write signals but even after using ideal write and read signals there was still the same amount of degradation. A alternative way to layout the memory cells probably by sacrificing the area in oder to get better performane might be an option.

9.2.2 Testing the high speed $\Delta\Sigma$ ADC

During the course of testing various variations in the outputs were observed for different test setups. It was found that the board3 as explained in the previous chapter was not found to work at 800 MHz and it worked only till 750 MHz although its results for lower frequencies matched the results from board2(from which the final results were used). Two different packaging of the chip was used for the 2 boards; it needs to be investigated whether the packaging itself caused such a difference between the results as even the chips from the same package were found to have variations (more peaking, higher noise floor). Finally the noise output of the signal generator was found to be close to the noise floor of the ADC, so using a better signal generator might help. Also the distortions introduced due to the balun transformer needs to be more accurately analyzed.

APPENDIX A

Excess loop delay compensation and the chip architecture

The maximum frequency of operation in a CT $\Delta\Sigma$ modulator is limited by the delay in the quantizer and the feedback DAC. The delay in the quantizer is caused because it takes a finite amount of time for the comparator in the Flash ADC to differentiate the inputs, this is known as regeneration time which acts like a loop delay. Any extra errors due to non-linearity of the quantizer is taken care by the loops filter at it is shaped out of the signal band (these errors can be considered like an extra error to the quantization error and hence get shaped out). The DAC circuit converts the digitized output of the quantizer to an analog signal which is then fedback to the input. But mismatch between the DAC elements causes an error which appears directly at the input [18]. Since the loop gain of the signal in the signal bandwidth is unity hence the error due to DAC also appears at the output. To prevent this different techniques are used for example DWA(data weighted averaging). Introducing these extra circuitry reduces the DAC error at the expense of extra delay(the DAC clock has to be delayed). All these factors mainly contribute to the ELD(excess loop delay).

Due to excess loop delay present in the loop the DAC will have to be clocked at a time greater than the delay($t > \tau_d$). But when the DAC pulse is delayed by some time say τ_d then the equivalent DT loop filter order increases by one [13]. This seriously affects the noise shaping and increases the inband noise and also reduces the maximum stable amplitude. If this delay becomes too large then the modulator will become unstable. Figure A.1 shows the effect of ELD on noise shaped idle channel output of the modulator. The next section will discuss about common ELD compensation techniques. After that we will discuss about the ELD technique used in the chip which is being tested as part of this thesis work.

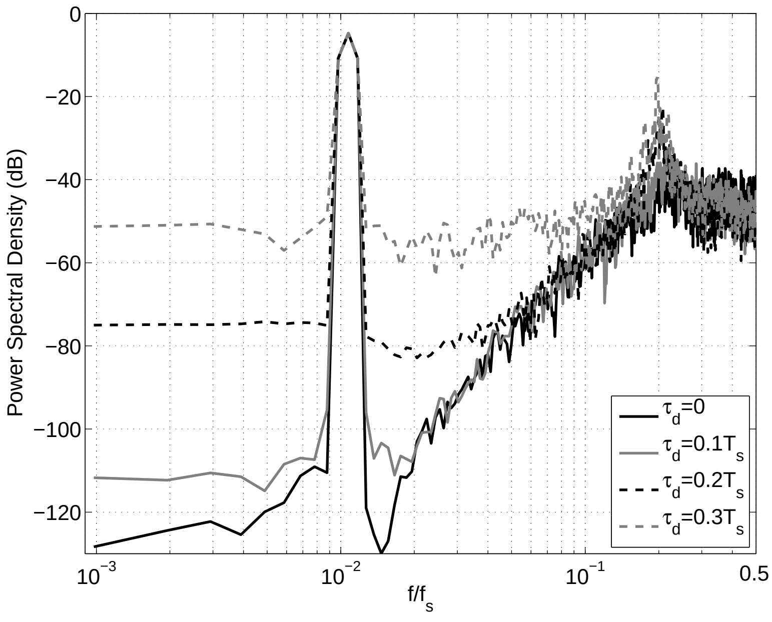


Figure A.1: Effect of ELD on Noise Shaping, [2]

A.1 Commonly used compensation techniques

One of the techniques used for compensating ELD is by introducing one clock cycle delay in the feedback path and then using another DAC to provide the missing sample. Due to introduction of one clock cycle delay the second sample of the impulse response of the main CT loop is zero. The extra new compensation path(with less delay) introduced compensates for this sample without affecting the other samples produced by the main DAC [19]. Hence the samples of the impulse response of the overall feedback loop becomes equivalent to that of the DT modulator hence compensating the ELD. The schematic in the Figure A.2 shows this.

Another more commonly used technique is to directly add the scaled DAC output to the output of the loop filter. The scaling constant depends on the delay of the feedback path. This technique is also shown in Figure A.3

The scaling consatant is realized by adding a differentiator after the output of the 1^{st} loop filter. This is done by passing the output from the first loop filter

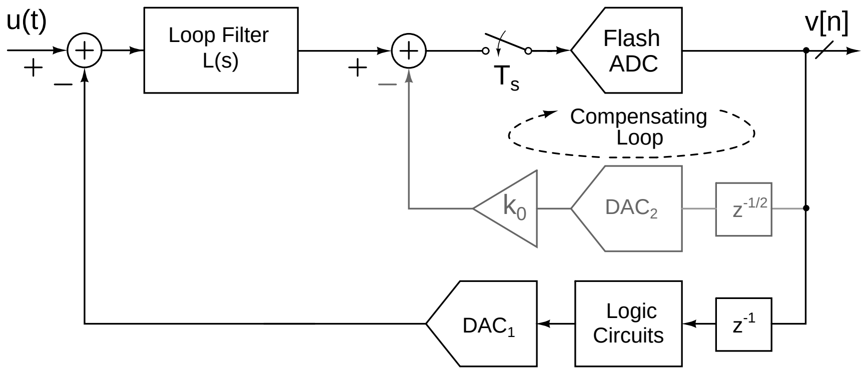


Figure A.2: ELD compensation using an extra DAC, [2]

integrator to the virtual node of the summing opamp using a capacitor.

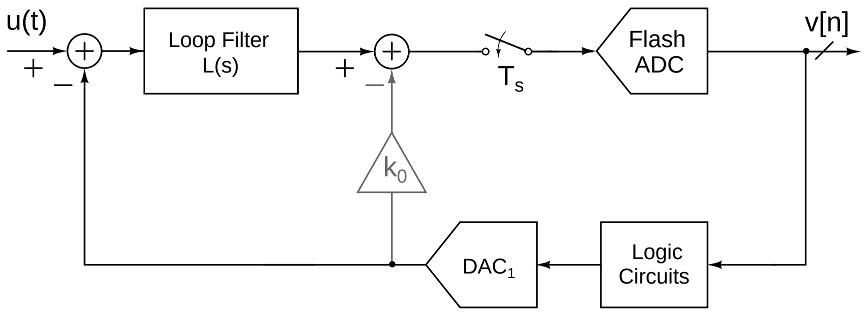


Figure A.3: ELD compensation using a differentiating path in the loop filter, [2]

A.1.1 Limitations of commonly used ELD techniques

The techniques proposed above have a lot of limitations. The most important being the excess loop delay has to be less than one clock cycle. This limits the maximum sampling speed of the modulator. Also the technique in which we introduced an extra DAC, the DAC loads the output of the quantizer and takes a larger area.

Let us now consider the technique in which we used a direct path to add a sample directly from the output of the DAC. As the loop delay increases the contribution from this direct path will keep increasing such that the overall NTF looks exactly similar to the ideal case without any delay [14]. For large delays it becomes very difficult to realize large gains keeping the delay low. Also as soon the delay in the feedback path becomes greater than one clock cycle then the second sample of the impulse response of the feedback path will be zero. Multiplying it with a constant will not be able to make the second sample non zero. This increases the order of the NTF and hence degrades the noise attenuation in the signal band and causes more peaking compared to ideal case.

The only way to compensate this excess delay of one clock cycle is to introduce an extra feedback path which is independent of this delay.

A.2 Compensating for more than unity cycle excess loop delay

The technique used in the chip [2] which is being tested, employs a local feedback circuit with lesser delay in order to compensate for the ELD of greater than unity clock cycle [15]. Figure A.4 shows a additional feedback loop to compensate for the missing second sample of the original loop response. The delay of this additional loop is significantly less (lesser than unity clock cycle) as it uses only a single sample and hold circuit. To make the response of the modulator similar to the ideal case we first find out the sampled response of the loop. To do this we break the loop and apply a impulse at the input of the quantizer. The samples of the output are then compared with the ideal response. The second sample is adjusted by the help of the scaling coefficient in the local feedback path. The remaining samples are adjusted by changing the loop filter coefficients such that the remaining response is equivalent to that of the ideal case. We now calculate the new NTF due to the introduction of this loop. We break the loop at point X at the start of the quantizer and find the transfer function from the error introduced in the quantizer to the output [2]. It is also easily seen from the figure that the transfer function of the fast extra feedback loop is $1/(1+az^{-1})$. We can model the flash ADC by adding a quantizaton error e_q. Hence the overall transfer function looks like

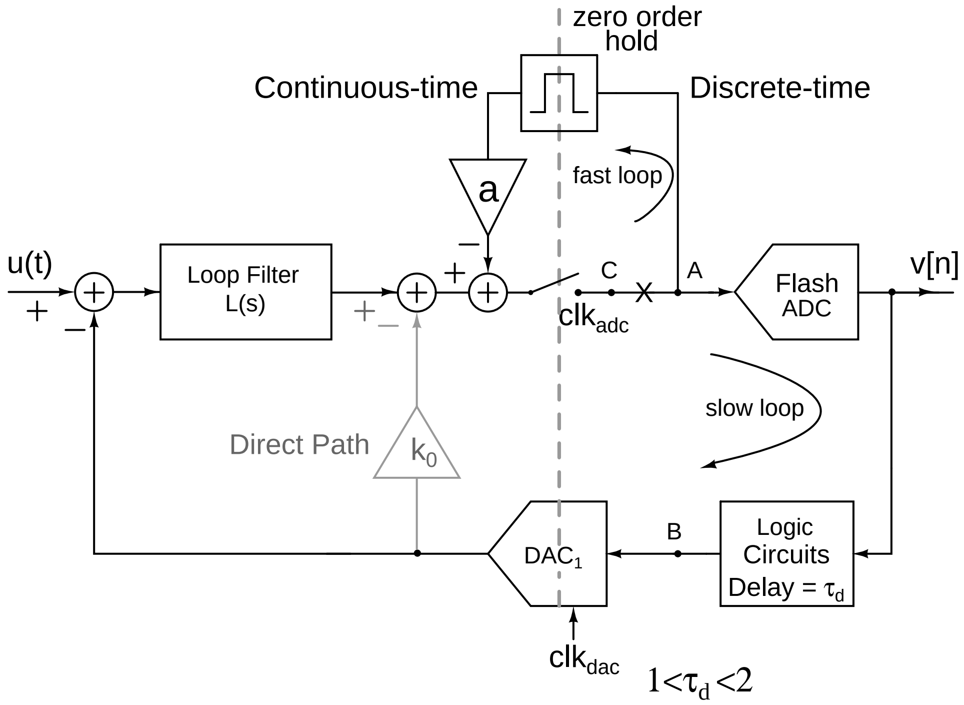


Figure A.4: ELD compensation using an extra fast feedback path, [2]

$$e_q(z) - V(z)[L_d(z)z^{-2} + k_0 z^{-2}](\frac{1}{1+az^{-1}}) = V(z) \tag{A.1}$$

$$\frac{V(Z)}{e_q(z)} = \frac{1+az^{-1}}{1+az^{-1}+(k_0+L_d(z))z^{-2}} \tag{A.2}$$

Since the coefficient of the loop filter were adjusted to match the samples hence the denominator remains the same as the ideal case. We now see an extra zero due to the compensation loop. Hence the new NTF in terms of ideal NTF is written as

$$NTF_{new} = (1+az^{-1})NTF_{ideal}(z) \tag{A.3}$$

The NTF of the compensated loop for a fourth order $\Delta\Sigma$ compared to the ideal case when the ELD is 1.5 looks as shown in Figure A.5. As we can see from the figure for $OSR = 25$ there is an 7.5dB increase in the quantization noise. Also by simulation it was seen that for higher OBG the performance becomes similar to that of the ideal case with a order less [2]. Hence to get a performance of a ideal 3rd order loop filter using this compensation technique we will have to design it for 4th order.

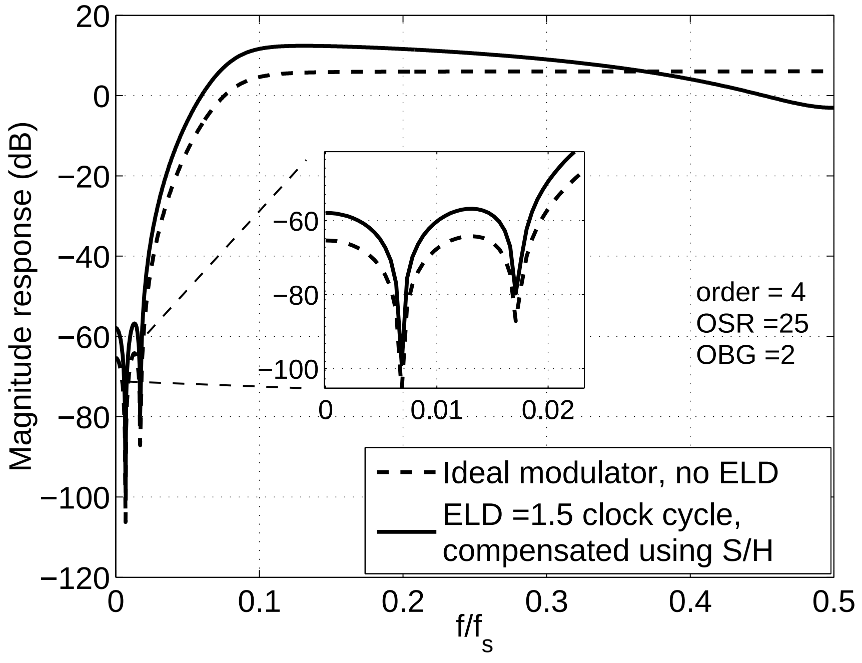


Figure A.5: Effect of the extra zero on the NTF, [2]

A.3 Architecture of the $\Delta\Sigma$ modulator

A.3.1 Loop filter

This section will briefly describe the overall architecture of the chip. More details regarding the chip can be found in [2]. The chip which was tested as part of this thesis work was designed for an OBG of 2 and an OSR of 25. A forth order loop filter with optimized zeros was used. The over all modulator looks as shown in Figure A.6

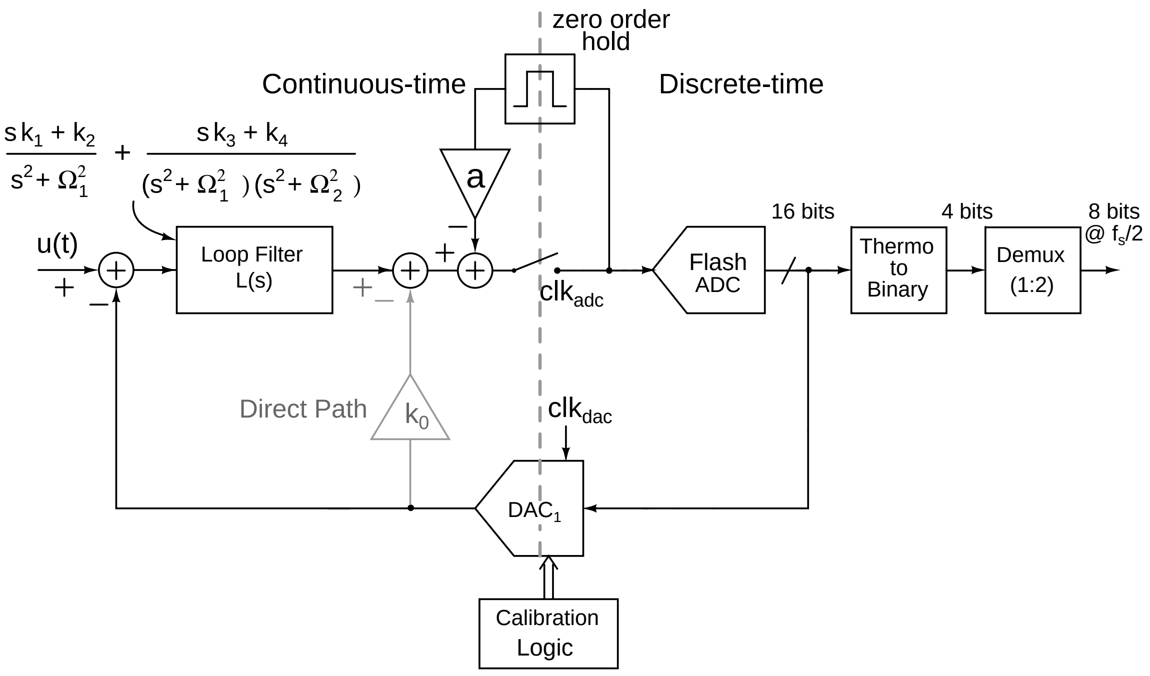


Figure A.6: Block diagram of the ADC, [2]

The loop filter chosen was a cascade of 4 integrators with feedforward summation(CIFF) [2]. To realize the integrators of the loop filter and a Active RC integrator was used. Active RC integrators were chosen over G_mC because of their higher linearity. The NTF for the modulator can be derived from the ideal case. Using the Schreier $\Delta\Sigma$ toolbox [20] in Matlab we know that the NTF of an ideal system for OSR of 25 and OBG of 2 is

$$NTF_{ideal} = \frac{(z^2 - 1.98z + 1)(z^2 - 1.98z + 1)}{(z^2 - 1.204z + 0.3771)(z^2 - 1.43z + 0.6585)} \tag{A.4}$$

$$NTF_{new} = (1 + az^{-1})NTF_{ideal}(z) \tag{A.5}$$

The first opamp for the loop filter was designed using a feedforward compensated architecture as it has high bandwidth and consumes low power(at the expense of signal swing). The other opamps(2nd,3rd and 4th) were designed using as simple 2 stage miller compensated opamps as these opamps are less crucial compared to the 1^{st} opamp.

A.3.2 Fast feedback ELD compensating path

To realize the fast feedback path we need two simple components one of them is the sample and hold part and the other is the constant multiplication part. The sample and hold is realized using a 2 stage track and hold circuit. That is while one stage is tracking the other holds and vice versa. The switches used on the track and hold circuit is boot-strap switch to reduce the distortion. For constant scaling part we use a G_m cell which is a simple voltage controlled current source. For a scaling constant of a_0 the G_m of the cell is kept at a_0/R where R is the feedback resistance(from the output to virtual ground) of the summing ompamp used after the loop filter.

A.3.3 Flash ADC

The 4 bit quantizer is realized using a flash ADC. Flash ADC architecture offers high speed at the expense of more power and area; since speed is critical here so a flash ADC was used. In this architecture the input is compared will all the possible quantization levels at the same instant. So for a 4bit ADC we will need $2^4 - 1 = 15$ comparators to compare the input with the quantization levels. The output of these comparators will be an array of 1's and 0's which is called a thermometer code. This then has to be converted to binary using a separate circuitry outside the modulator, the DAC uses the thermometer code itself. The reference levels are generated using resistive ladders and a differential version of the circuit is implemented.

A.3.4 4 bit DAC

The DAC converts the digital outputs of the quantizer to analog signals which can be compared with the input signal in a CT modulator. The DAC topology used in [2] is a complementary current steering DAC which feeds the current to the loop filter. This architecture is used because the total noise PSD is 2 times that of current steering sources. Whereas in the more commonly used DAC architecture using NMOS current steering and PMOS current source the total noise PSD is 4 times that of the current source noise.

We know that mismatch in the DAC cells can seriously affect the performance of the modulator as any kind of error introduced due to the DAC comes directly at the output. To reduce mismatch various techniques like dynamic element matching or data weighted averaging can be used so that the mismatch between the DAC cells on an average is minimised [18]. In the current modulator the DAC uses a *calibration scheme* to reduce the mismatch. The basic idea is to charge the gate capacitance of the NMOS to a value such that it draws the reference current when it is switched on. So in one cycle we connect the reference current source to the M1 and let the capacitor charge to the reference value and when we want to use it we disconnect the reference current source and connect the NMOS as shown in Figure A.7

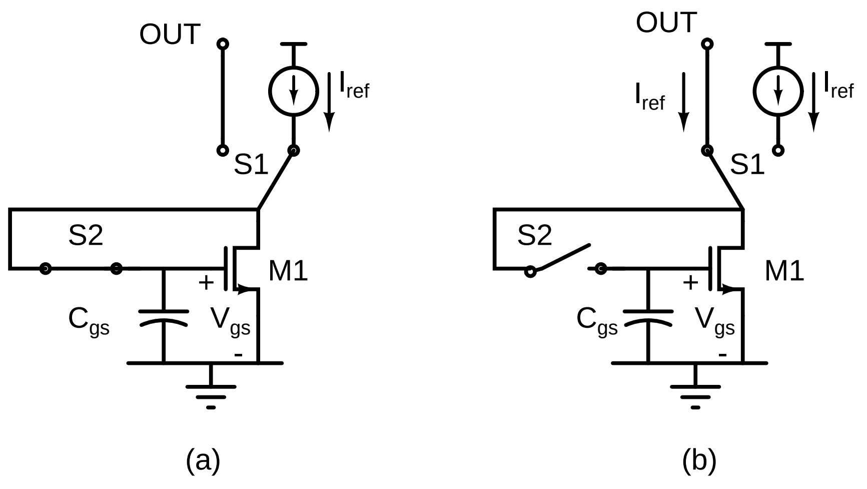


Figure A.7: Calibratoin of a DAC cell (a)Calibration Phase (b)Output phase, [2]

A similar technique is used for this chip in which each DAC cell has two reference currents sources and two reference current transistors. This is because when one of the current sources is being used by the DAC the other gets calibrated. Once the calibration of the other cell is complete the two sources switch their functionality and now the other source enters the calibration phase . Also since we have say N DAC cells not all the N transistors (which are not connected to the DAC output) in these cells are being calibrated, only one transistor is calibrated and then the others in the other cells are calibrated in a sequence(to generated this sequence we require some extra switching circuitry). So among the N DAC cells each of the transistors connected to the calibration circuit is calibrated in a sequence once by one and inside each DAC cell as soon as once of the current source gets calibrated it gets connected to DAC output and the other transistor connects to the calibration circuit as it will then be calibrated in the next cycle after the calibration of the other DAC cells are complete.

REFERENCES

[1] H. Vaishnav, *Design of High Speed Front End Circuitry for Neutrino Detectors.* IIT Madras, June 2011.

[2] V. Singh, *Design of a High Speed High Resolution Continuous-Time Delta Sigma Modulator.* IIT Madras, July 2010.

[3] G. M. Haller, *High Speed, High Resolution Analog Waveform Sampling in VLSI Technology.* Stanford University, March 1994.

[4] J.Walker, S. Chae, S. Shapiro and R. Larsen, "Microstore-The Stanford Analog Memory Unit," *IEEE Transactions on Nucl. Sci.*, vol. 32, pp. 616–621, Feb. 1985.

[5] E. Franchi, M. Tartagni, R. Guerrieri and G. Baccaranie, "Random Access Analog Memory for Early Vision," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 1105–1109, July. 1992.

[6] C. MEad and L. Conway, *Introduction to VLSI systems.* Addison-Wesley, 1980.

[7] [Online]: http://www.maxim-ic.com/app-notes/index.mvp/id/1080.

[8] T. Tulabandhula and Y. Mitikiri, "A 20MS/s 5.6 mW 6b asynchronous ADC in 0.6μ CMOS," in *22nd International Conference on VLSI Design 2009, VLSID.*, 2009.

[9] Liu, Chun-Cheng; Chang, Soon-Jyh; Huang, Guan-Ying; Lin, Yin-Zu, " A 0.92mW 10-bit 50-MS/s SAR ADC in 0.13μm CMOS process," in *2009 Symposium on VLSI Circuits*, pp. 236 – 237, Aug 2009.

[10] B. Ginsburg and A. Chandrakasan, "500-MS/s 5-bit ADC in 65-nm CMOS With Split Capacitor Array DAC," *IEEE Journal of Solid-State Circuits,*, vol. 42, pp. 739–747, March. 2007.

[11] V. Hariprasath, J. Guerber, S.-H. Lee and U.-K. Moon, "Merged capacitor switching based SAR ADC with highest switching energy-efficiency," *Electronics Letters*, vol. 46, pp. 620–621, April. 2010.

[12] You-Kuang Chang, Chao-Shiun Wang and Chorng-Kuang Wang, "A 8-bit 500-KS/s Low Power SAR ADC for Bio- Medical Applications," in *IEEE Asian SolidState Circuits Conference*, pp. 228 – 231, 2008.

[13] J. Cherry and W. Snelgrove, "Excess loop delay in continuous-time delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 4, pp. 376–389, 1999.

[14] S. Pavan, "Excess loop delay compensation in continuous-time delta-sigma modulators," *IEEE Transactions on Circuits and Systems II: Express briefs*, vol. 55, no.11, pp. 1119–1123, November 2008.

[15] Vikas Singh, Nagendra Krishnapura, Shanthi Pavan, "Compensating for Quantizer Delay in Excess of One Clock Cycle in Continuous-Time $\Delta\Sigma$ Modulators," *IEEE Transactions on Circuits and Systems*, vol. 43, pp. 351–360, Feb. 2010.

[16] [Online].Available: www.xilinx.com/tools/cspro.htm.

[17] [Online].Available: www.stanford.edu/ phartke/chipscope_tutorial.pdf.

[18] S. Pavan and N. Krishnapura, "Oversampling Analog-to-Digital Converter Design," in *21st International Conference on VLSI Design 2008, VLSID.*, pp. 7–7, 2008.

[19] S. Yan and E. Sanchez-Sinencio, "A continuous-time $\Sigma\,\Delta$ modulator with 88-dB dynamic range and 1.1-MHz signal bandwidth," *IEEE Journal of Solid-State Circuits*, vol. 39, pp. 75–86, Jan. 2004.

[20] R. Schreier, "$\Delta\,\Sigma$ Toolbox," *[Online]. Available: http://www.mathworks.com.*