İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

CHAOTIC OSCILLATOR BASED RANDOM NUMBER GENERATOR

M.Sc. Thesis by

Fidel BAYAM, B.Sc.

Department: Electronics and Communication Engineering

Programme: Electronics Engineering

MAY 2005

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

CHAOTIC OSCILLATOR BASED RANDOM NUMBER GENERATOR

M.Sc. Thesis by

Fidel BAYAM, M.Sc.

(504021207)

Date of submission : 9 May 2005

Date of defence examination: 30 May 2005

Supervisor (Chairman): Assoc. Prof. Dr. Ali ZEKİ

Members of the Examining Committee Assoc. Prof. Dr. Serdar ÖZOĞUZ

Assist. Prof. Dr. A. Şima ETANER-UYAR

MAY 2005

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

KAOTİK OSİLATÖR TABANLI RASGELE SAYI ÜRETECİ

YÜKSEK LİSANS TEZİ

Müh. Fidel BAYAM

(504021207)

Tezin Enstitüye Verildiği Tarih : 9 Mayıs 2005

Tezin Savunulduğu Tarih : 30 Mayıs 2005

Tez Danışmanı : **Doç.Dr. Ali ZEKİ**

Diğer Jüri Üyeleri **Doç. Dr. Serdar ÖZOĞUZ**

Yrd. Doç. Dr. A. Şima ETANER-UYAR

MAY 2005

ACKNOWLEDGEMENT

First I would like to thank my supervisor Assoc. Prof. Ali Zeki for his guidance and support during my B.Sc. and M.Sc. thesis.

Next I would like to thank Assoc. Prof. Serdar Özoğuz for his valuable contribution. In this work, the test chip was submitted with the research budget of Assoc. Prof. Dr. Serdar Özoğuz who is awarded by TÜBA's (Türkiye Bilimler Akademisi) Young Scientists Award Program (TÜBA-GEBİP).

I feel obliged to thank to my family, without their support, all I could have achieved would be a complete failure.

Finally, I would like to thank Keklik Alptekin for her endless support. I feel very lucky for every single day of six years we have spent together.

May 2005 Fidel Bayam

CONTENTS

TABLE LIST vi
FIGURE LIST vii
ÖZET ix
SUMMARY x

1. INTRODUCTION 1
1.1 Motivation 1
1.2 Organization of Thesis 1

2. RANDOM NUMBER GENERATION BASICS 3
2.1 Pseudo Random Number Generators (PRNGs) 4
2.2 True Random Number Generators (TRNGs) 5
2.2.1 SW-Based Generators 5
2.2.2 HW-Based Generators 6
2.3 Post-Processing 6
2.4 Statistical Tests 7
2.4.1 Frequency Test (Monobit Test) 7
2.4.2 Serial Test 7
2.4.3 Poker Test 8
2.4.4 Runs Test 8
2.4.5 Autocorrelation Test 8
2.4.6 FIPS 140-1 Statistical Tests 8
2.5 IC Random Number Generators 9
2.5.1 Direct Amplification of Noise 9
2.5.2 Oscillator Sampling 10
2.5.3 Chaos Based Generators 10

3. CHAOS 12
3.1 Chaos in Electronic Systems 12
3.2 Chua's Circuit 14

4. CHAOTIC OSCILLATOR BASED RANDOM NUMBER GENERATOR 16
4.1 Construction of Chaotic Oscillator 16
4.1.1 RLC Resonator 16
4.1.2 Basic LC Oscillator 19
4.1.3 Block Level Chaotic Oscillator 21
4.1.4 Bipolar Transistor Chaotic-LC Oscillator 24
4.1.5 MOS Transistor Chaotic-LC Oscillator 28
4.2 Random Bit Generation 31
4.2.1 Construction of the Clock Signal 33
4.2.2 Combining Bits In One Signal 34
4.2.3 Implementing Von Neumann's De-Skewing Algorithm 34
4.3 IC Implementation 35
4.3.1 Bipolar Chaotic Oscillator 36
4.3.2 MOS Transistor Chaotic-LC Oscillator 37
4.3.3 Difference Amplifier 39

4.3.4 Comparator Circuit 41
4.3.5 “OR” Circuit 42
4.3.6 DFF 44
4.3.7 EXOR Gate 45
4.3.8 Divide-by-two Circuit 46
4.3.9 Output Drivers 47
4.3.10 Toplevel Integration 48

5. CONCLUSION 50

REFERENCES 52

BIOGRAPHY 54

TABLE LIST

Page

No

Table 2.1: Conditions of runs test **9**
Table 4.1: Different parameter-sets for which chaotic oscillation occurs **23**
Table 4.2: Parameter sets that generate chaotic oscillation **30**

FIGURE LIST

Page No

Figure 2.1: Direct amplification of noise **10**
Figure 2.2: Oscillator sampling technique **10**
Figure 3.1: Chua's Circuit **14**
Figure 3.2: I-V characteristic of Chua's diode **15**
Figure 3.3: Attractor of Chua's circuit **15**
Figure 4.1: Parallel RLC circuit **16**
Figure 4.2: V_1 versus t, for R=-10Ω **17**
Figure 4.3: V_1 versus t, for R=10Ω **18**
Figure 4.4: V_1 versus t, for R=∞ **18**
Figure 4.5: The vector field of a system (a) for positive R; (b) for negative R **19**
Figure 4.6: Implementation of negative resistance with cross coupled transistors **20**
Figure 4.7: Plot of (V_1/V_T) for I=1mA, and V_T=25mV **21**
Figure 4.8: Chaotic circuit **22**
Figure 4.9: V_1 for a_1=0.6, and a_2=2 ..... **23**
Figure 4.10: Trajectory of the chaotic circuit in Figure 4.8 ..... **24**
Figure 4.11: Chaotic LC oscillator **25**
Figure 4.12: Numeric analyses result of the BJT Chaotic-LC oscillator **27**
Figure 4.13: Trajectory of the BJT Chaotic-LC Oscillator (x versus z) **27**
Figure 4.14: MOS transistor chaotic-LC oscillator **28**
Figure 4.15: Waveform of the State Variable x versus time **31**
Figure 4.16: Trajectory of the MOS Chaotic-LC Oscillator **31**
Figure 4.17: Constructed subspaces with comparator references **32**
Figure 4.18: Placement of comparator references **33**
Figure 4.19: Connections of comparators **34**
Figure 4.20: construction of clock and data signals **34**
Figure 4.21: Realization of Von-Neumann's De-Skewing Algorithm **35**
Figure 4.22: Phase Space (V_A-V_B) versus (V_C-V_D) of the BJT Chaotic Oscillator **37**
Figure 4.23: Waveform of (V_A-V_B) **37**
Figure 4.24: Phase Space of the MOS Transistor Chaotic Oscillator **38**
Figure 4.25: Waveforms of (V_A-V_B) and (V_C-V_D) differences **39**
Figure 4.26: Subtraction circuit **40**
Figure 4.27: DC characteristic of a differential pair **40**
Figure 4.28: Simulation results of subtraction circuit **41**
Figure 4.29: Comparator circuit **42**
Figure 4.30: Simulation results of comp0 and comp1 ..... **42**
Figure 4.31: OR circuit **43**
Figure 4.32: Getting V_{ref} **43**
Figure 4.33: Simulation results of OR circuit **44**
Figure 4.34: DFF circuit **44**
Figure 4.35: Simulation results of DFF **45**
Figure 4.36: EXOR gate **46**
Figure 4.37: Simulation results of EXOR **46**
Figure 4.38: Construction of signal "clk/2" **47**
Figure 4.39: Load structure was used in CML simulations **47**

Figure 4.40: Simulation results of CML drivers.....**48**
Figure 4.41: Toplevel layout of the system.....**49**

KAOTİK OSİLATÖR TABANLI RASGELE SAYI ÜRETECİ

ÖZET

Bu çalışmada, yüksek hızlı, sürekli zaman LC-kaotik osilatör tasarlanmış ve bu osilatörün çıkışları rasgele bit üretiminde kullanılmıştır. Hem Bipolar hem de MOS transistorlu osilatör versiyonları için devre deklemleri türetilmiştir. Bu denklemlerin nümerik denklem çözücü programlar yardımıyla çözülmesiyle kaotik osilasyonun sağlandığı görülmüştür. Devreler, Spectre spice simülatörü ve IHP SGB25VD 0.25µm SiGeC BiCMOS prosesi model parametreleri kullanılarak test edilmiştir. Rasgele sayı üretimi, osilatör çıkışlarının 2 farklı referansla karşılaştırılmasıyla elde edilmektedir. Oluşturulan bitlerin istatistiksel özelliklerini iyileştirmek amacıyla Von-Neumann algoritması tasarlanarak entegre edilmiştir. Üretilen çıkış bitleri periyodik olmadığından anlamlı bitlerin oluşma anlarını belirten bir saat işareti tanımlanmıştır. Rasgele sayı üretimi için gerekli olan alt bloklar yüksek hızlı çalışmaya uygun olacak şeklide Emetör Bağlamalı Lojik ve Akım Modlu Lojik aileleri kullanılarak tasarlanmıştır. Spectre simülatöründe gerçekleştirilen simülasyonlar, tasarlanan rasgele bit üretecinin yaklaşık 300Mbit/s hızında çıkış oluşturabildiğini göstermiştir. Çıkış işaretlerini cip dışına alabilmek amacıyla Akım Modlu Lojik çıkış sürecüleri tasarlanmıştır. Kaotik osilatör ve rasgele bit üreteci sistemi, IHP SGB25VD 0.25µm SiGeC BiCMOS prosesi ile gerçeklenmiş ve üretime gönderilmiştir. Çipin toplam güç harcaması 50mW mertebesindedir. Toplam kırmık alanı 1 mm x 0.5 mm'dir.

CHAOTIC OSCILLATOR BASED RANDOM BIT GENERATOR

SUMMARY

In this study, a high speed continuous time LC-chaotic oscillator was designed and utilized as a random bit generator. Circuit equations were derived for both MOS transistor and BJT versions. These equations were solved by using numeric solvers, and chaotic oscillation was observed. Spectre circuit simulator was used as the simulator. Circuits were verified by using IHP's SGB25VD 0.25µm SiGeC BiCMOS process. To generate successive '1's and '0's, two comparators with different references were used. A well-known Von-Neumann de-skewing algorithm was also implemented in order to improve statistical properties of the generated bit stream. The clock signal was constructed using the outputs of the comparators in order to define the bit generation events. The random bit generation sub-blocks were implemented as bipolar Emitter Coupled Logic (ECL) and Current Mode Logic (CML) gates. Spectre simulations showed that the average throughput of the designed random bit generator is approximately 300Mbit/s. The CML output drivers were designed to output the generated data and clock signals. The whole system, including the BJT chaotic oscillator and the random bit generation sub-blocks, were implemented in IHP's SGB25VD 0.25µm SiGeC BiCMOS process. The chaotic oscillator and the random bit generator block consume approximately 50mW power under typical conditions. Total area of the chip is 1 mm x 0.5 mm.

1. INTRODUCTION

1.1 Motivation

Random numbers are statistically independent and unbiased binary digits, and a random number generator is a system whose output consists of fully unpredictable bits. Random Number generation has a great importance in many applications. For example: numerical simulations, gaming, statistical analysis, distributed computations and cryptographic protocols. Almost all cryptographic protocols require the generation and use of secret values that must be unknown to attackers, so in cryptographic applications, the unpredictability of the output implies that the generator must also be not observable and not manipulable by any attacker.

There are two kinds of random number generators (RNGs). Pseudo Random Number Generators (PRNGs) use deterministic algorithms to generate random bits starting from the initial seed. True Random Number Generators (TRGNs) use a non-deterministic phenomenon to produce randomness [1].

True random number generation can be made by using various techniques. One of them is using chaotic circuits. Although chaos is a deterministic process, most times it is accepted to be a true random number generator due to very high initial condition sensitivity and complex behavior. For electronic systems a deterministic system is called chaotic if an infinitesimally small perturbation to its initial conditions produces a change in its behavior.

In 1961, Edward Lorentz discovered the butterfly effect coincidently while trying to forecast the weather. “Butterfly effect” is the idea that very small causes can produce dramatically out-of-proportion effects.

Chaos was first observed electronically in Chua’s circuit. Chaotic nature of Chua’s circuit was first observed by Matsumoto in 1983, and the first experimental results of Chua’s circuit which confirm the presence of chaos were taken by Zhong and Ayrom in 1984 [2].

1.2 Organization of Thesis

This thesis presents a high speed continuous time chaotic-LC oscillator and utilizes it to implement a high speed random bit generator.

Chapter 2 presents basic concepts of random number generation. Different types of random number generators and randomness tests are explained. Important IC implementations are also given.

Chapter 3 is a review of chaos theory.

Chapter 4 is a main subject of the thesis. Designed high speed chaotic LC-oscillator is explained in detail. Random bit generation method is presented in a detail. Implementation of Von-Neumann de-skewing algorithm is also given.

Chapter 5 is a review of the thesis and conclusion is given.

2. RANDOM NUMBER GENERATION BASICS

In general, random numbers can be summarized as numbers that are indistinguishable from outcomes that would arise purely by chance. With another way of saying, random numbers are the statistically independent and unbiased binary digits, that are outputs of algorithmic or device level random bit generators. The quality of a random number generator is often measured by the degree to which it produces unpredictable and unbiased output [2]

Random numbers are widely used in;

- numerical simulations
- gaming
- statistical analysis (Monte Carlo simulations)
- distributed computations,
- secure communication protocols (SSL, GSM…) and of course
- cryptography

Because security protocols rely on the unpredictability of the keys they use, random number generators for cryptographic applications must meet stringent requirements. The most important is that attackers, including those who know the RNG design, must not be able to make any useful predictions about the RNG outputs. In particular, the apparent entropy of the RNG output should be as close as possible to the bit length [1].

According to Shannon theorem, the entropy H of any message or state is:

$$H = -\sum_{i=1}^{n} p_i \log p_i \qquad (2.1)$$

where p_i is the probability of state i out of n possible states. In the case of a random number generator that produces a k-bit binary result, p_i is the probability that an output will equal i, where $0 \leq i < 2^k$. Thus, for a perfect random number generator, $p_i = 2^k$ and the entropy of the output is equal to k bits. This means that all possible outcomes are equally (un)likely, and on average the information present in the output cannot be represented in a sequence shorter than k bits [1].

Almost all cryptographic protocols require the generation and use of secret number [2]. For example:

- Conventional encryption requires the generation of unguessable keys.
- The computation of a digital signature with the Digital Signature Algorithm requires, besides the signer's private key, a value customarily called k that must be secret, and that must not be re-used.
- Standards for message encryption using the RSA algorithm generally require the use of random numbers to form message padding.
- Many challenge-response protocols require the use of a unique number, or nonce. In practice, a good way to produce a number with a large likelihood of being unique is to use a sufficiently large random number.

While a high quality random source is always best, application can be classified into two categories according to randomness requirements [2].

1. Applications which need flat statistic and unbiased bit streams but have fewer unpredictability requirements; such as numerical simulations. In this type of applications pseudo random number generators (PRNGs) can be used
2. Applications which have extremely strong unpredictability requirements but may be slightly tolerant of biased information; such as cryptographic applications. These kind of applications often need true random number generators (tRNG)

2.1 Pseudo Random Number Generators (PRNGs)

In many applications where a random bit stream is required, pseudorandom number generators (PRNGs) are used. Applications where flat statistic and unbiased bits are enough, such as numerical simulations, PRNGs can be safely used. In addition properly-implemented and seeded PRNGs are also suitable for most cryptographic applications, however great care must be taken in the development, testing, and selection of PRNG algorithms. PRNGs use a deterministic process to generate bit sequence, starting from an initial seed. Since the adopted algorithms are usually public, the seed is the only source of randomness and the actual entropy of the output can never exceed the entropy of the seed. Therefore, it is critical that a PRNG be properly seeded from a source of true randomness. In most cases, it must be properly seeded from a source of true randomness.

The output of a PRBG is not random; in fact, the number of possible output sequences is at most a small fraction of all possible binary sequences. The intent is to take a small truly random sequence and expand it to a sequence of much larger length [3].

2.2 True Random Number Generators (TRNGs)

Since it is impossible to create true randomness from within a deterministic system, True Random Number Generators (TRGNs) use a non-deterministic source to produce randomness.

In the applications, the random source can be constructed of dedicated hardware devices; otherwise; the random source can use software procedures to extract random processes from the platform on which the generator is implemented. Generators of the first type are commonly called hardware-based (HW); generators of the second type are software-based (SW) [3].

2.2.1 SW-Based Generators

Generally, SW-based generators are implemented on computer systems and the values typically exploited as raw stream sources are obtained from:

- event timings:
    - mouse movements and clicks
    - keystrokes
    - disk and network accesses
- data depending on the history of the system and/or large amount of events:
    - system clock
    - I/O buffers
    - Load and network statistics

The behavior of such processes can vary considerably depending on various factors, such as the computer platform, and entropy is mostly low and difficult to evaluate as well as the actual robustness with respect to observation and manipulation.

A well-designed software random bit generator should utilize as many good sources of randomness as are available. Using many sources guards against the possibility of a few of the sources failing, or being observed or manipulated by an adversary. Although employing these protections, it’s had to say that most PRNGs are immune to attacks. In 1996 two researchers found that Netscape’s random number

generator seed was derived from “just three quantities: the time of day, the process ID, and the parent process ID. Thus, an adversary who can predict these three values can apply the well-known MD5 algorithm to compute the exact seed generated [1].

2.2.2 HW-Based Generators

HW-Based generators use a non-deterministic, physical phenomenon to produce randomness. Most operate by measuring unpredictable natural processes. Examples of such physical phenomena include [3]:

1. elapsed time between emission of particles during radioactive decay;
2. thermal noise from a semiconductor diode or resistor;
3. the frequency instability of a free running oscillator;
4. flip-flop metastability
5. the amount a metal-insulator-semiconductor capacitor is charged during a fixed period of time;
6. air turbulence within a sealed disk drive which causes random fluctuations in disk drive sector read latency times; and
7. sound from a microphone or video input from a camera.

2.3 Post-Processing

In practice, every kind of raw random source can present defects as offset, auto-correlation or cross-correlation. The probability of the producing a ‘1’ bit may not be equal to ½ because of interference with other signals (cross-correlation) or the probability of the generated output bit depends on previous bits because of bandwith limitations (auto-correlation).

There are various techniques for generating truly random bit sequences from the output bits of such a defective generator; such techniques are called de-skewing techniques.

One of the well-known de-skewing algorithms is the Von Neumann’s algorithm [3]. According to Von Neumann’s algorithm;

- “01” sequences must be converted into ‘0’ bit,
- “10” sequences must be converted into ‘1’ bit

- “00” and “11” sequences must be discarded

2.4 Statistical Tests

Statistical tests are designed to measure the quality of a generator. While it is impossible to give a mathematical proof that a generator is indeed a random bit generator, the tests help detect certain kinds of weaknesses the generator may have. This is accomplished by taking a sample output sequence of the generator and subjecting it to various statistical tests. Each statistical test determines whether the sequence possesses a certain attribute that a truly random sequence would be likely to exhibit; the conclusion of each test is not definite, but rather probabilistic. An example of such an attribute is that the sequence should have roughly the same number of 0’s as 1’s. If the sequence is deemed to have failed any one of the statistical tests, the generator may be rejected as being non-random; alternatively, the generator may be subjected to further testing. On the other hand, if the sequence passes all of the statistical tests, the generator is accepted as being random. More precisely, the term “accepted” should be replaced by “not rejected”, since passing the tests merely provides probabilistic evidence that the generator produces sequences which have certain characteristics of random sequences. There are 5 basic statistical tests [3].

2.4.1 Frequency Test (Monobit Test)

The purpose of this test is to determine whether the number of ‘0’bits and in binary sequence (s) of length n are approximately same as would be expected from a random sequence. Let n_0, and n_1 denote the number of ‘0’bits and ‘1’bits in sequence s, respectively. The statistic used is

$$X_1 = \frac{(n_0 - n_1)^2}{n} \tag{2.2}$$

2.4.2 Serial Test

The purpose of this test is to determine whether the number of occurrences of 00, 01, 10 and 11 as subsequences of binary sequence s are same, as would be expected for a random sequence. Let n_0, and n_1 denote the number of ‘0’bits and ‘1’bits in s, and let n_{00}, n_{01}, n_{10}, and n_{11} denotes number of occurrences of “00”, “01”, “10”, and “11” respectively. Note that $n_{00} + n_{01} + n_{10} + n_{11} = (n - 1)$ since the subsequences are allowed to overlap. The statistic used is

$$X_2 = \frac{4}{n-1}\left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2\right) - \frac{2}{n}\left(n_0^2 + n_1^2\right) + 1 \tag{2.3}$$

2.4.3 Poker Test

Let m be a positive integer such that $\left[\frac{n}{m}\right] \geq 5\left(2^m\right)$ and let. $k = \left[\frac{n}{m}\right]$. Divide the sequence s into k non-overlapping parts each of length m, and let n_i be the number of occurrences of the i^{th} type of sequence of length m, $1 \leq i \leq 2^m$. The poker test determines whether the sequences of length m each appear approximately the same number of times in s, as would be expected for a random sequence. The statistic used is

$$X_3 = \frac{2^m}{k}\left(\sum_{i=1}^{2^m} n_i^2\right) - k \tag{2.4}$$

The poker test is a generalization of the frequency test: setting m = 1in the poker test yields the frequency test.

2.4.4 Runs Test

The purpose of the runs test is to determine whether the number of runs (of either zeros or ones) of various lengths in the sequence s is as expected for a random sequence. The expected number of gaps (or blocks) of length i in a random sequence of length n is $e_i = (n - i + 3)/2^{i+2}$. Let k be equal to the largest integer i for which. Let B_i, G_i be the number of blocks and gaps, respectively, of length i in s for each i, $1 \leq i \leq k$. The statistic used is

$$X_4 = \sum_{i=1}^{k} \frac{(B_i - e_i)^2}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)^2}{e_i} \tag{2.5}$$

2.4.5 Autocorrelation Test

The purpose of this test is to check for correlations between the sequence s and (non-cyclic) shifted versions of it.

2.4.6 FIPS 140-1 Statistical Tests

Federal Information Processing Standards (FIPS) specifies four statistical test for randomness with FIPS 140-1 standard [4]:

- monobit test. The number n_1 of '1'bits in s must satisfy 9654 < n1 < 10346.
- poker test. The statistic X_3 defined by equation (2.4) is computed for m = 4, and the poker test is passed if 1,03 <X_3 < 57,4 is satisfied.
- runs test. The number B_i and G_i of blocks and gaps, respectively, of length i in s are counted for each i, $1 \leq i \leq 6$ (For the purpose of this test, runs of length greater than 6 are considered to be of length 6). The runs test is

passed if the 12 counts B_i, G_i, $1 \leq i \leq 6$, are each within the corresponding interval specified by the following Table 2.1.

Table 2.1: Conditions of runs test

Length of run	Required interval
1	2267-2733
2	1079-1421
3	502-748
4	223-402
5	90-223
6	90-223

- long run test. The long run test is passed if there are no runs of length 34 or more.

2.5 IC Random Number Generators

The increasing usage and importance of network communication and cryptography results in an increasing to use integrated RNGs. There are a various IC implementations of RNGs, and the important ones are explained below.

2.5.1 Direct Amplification of Noise

The direct amplification technique shown in Figure 2.1 uses a high-gain high-bandwidth amplifier to process the small ac voltage produced by a noise source such as thermal or shot noise, is in the order of μV making the RNG very sensitive to signal coupling. The noise must be amplified to a level where it can be accurately processed with no bias by a clocked comparator. This is the most popular RNG technique for single-chip solutions where shielding of the noise source is possible. The lack of adequate shielding from power supply and substrate signals in an IC environment prohibits the exclusive use of this method for IC-based cryptographic systems [4].

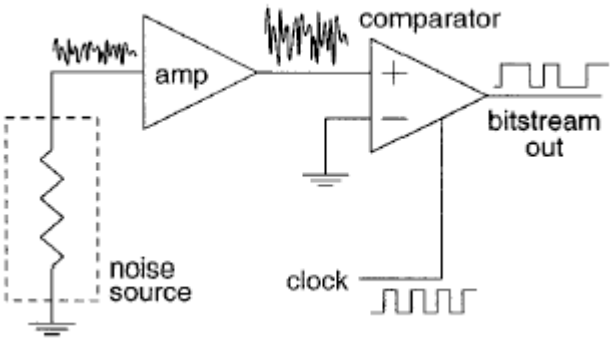

Figure 2.1: Direct amplification of noise

2.5.2 Oscillator Sampling

Oscillator based RNGs have a advantage over direct amplification of noise technique even in the presence of sinusoidal signal coupling. Oscillator based RNGs use oscillator timing jitter as a source of randomness. Two or more oscillators are combined to produce a random bit stream. In Figure 2.2, a low frequency oscillator samples the output of a high frequency oscillator using a D flip-flop. The level of randomness depends on the mean frequency separation of the oscillators and the amount jitter. If the low frequency oscillator period has a standard deviation much greater than the fast oscillator period, then the states for two successive sampled times can be considered uncorrelated and therefore the output bit stream is random in nature [1].

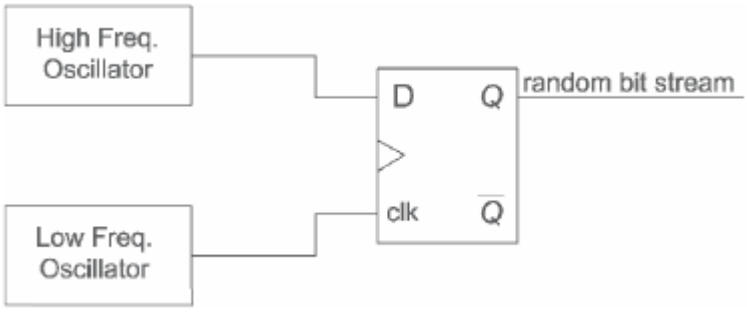

Figure 2.2: Oscillator sampling technique

In some applications, to improve statistical properties of generated bits, and achive a high bit rates the oscillation frequency of VCO is controlled by other RNG [4].

2.5.3 Chaos Based Generators

A deterministic system is called chaotic if an infinitesimally small perturbation to its initial conditions produces a change in its behavior that grows exponentially with time. Chaos will be examined in detain in the next chapter.

While chaos is a concept completely different from randomness, it is important in random-number generation for the following reason: If an RNG is chaotic, and if there is some inescapable uncertainty in any contribution to its state (e.g., due to thermal noise), then simply by waiting for a certain length of time, namely the time required for the exponential growth of that uncertainty to reach the magnitude of the system's gross state, it can be assumed that the state of the system is unknowable. By waiting a sufficient length of time between samplings, is can be possible to sample high-quality random bits from a chaotic system that is otherwise deterministic [2]. This time period changes from implementation to implementation.

3. CHAOS

The chaos theory is often ascribed to Edward Lorenz. Edward Lorenz was a meteorologist at MIT who showed that weather is chaotic and ultimately unpredictable. In 1961, he used term “butterfly effect” to explain his theory. “Butterfly effect” is the idea that very small causes can produce dramatically out-of-proportion effects. The notion that "the flap of a butterfly's wings in Brazil" might "set off a tornado in Texas" was presented in a lecture by Edward Lorenz to illustrate the impossibility of perfect weather prediction even if all known causes and effects could be measured. The butterfly effect is an illustration of sensitive dependence on initial conditions.

In other way of saying, chaos is the certain systems, in both nature and mathematics, appear to be governed by chance, but can be shown to be deterministic through analysis, phase space maps and computer models. These systems exhibit a sensitive dependence on initial conditions, so that even small variations in their starting conditions will produce wildly differing results.

Every model able to produce chaotic behavior must be a non-linear, dynamical system. Simply, dynamic system is a system in a motion. The swing of a pendulum, boiling water, weather, the growth of populations and the interactions between atoms and molecules are all examples of dynamical systems. Some are predictable and some are not, but they are all systems whose future motions depend entirely on past movements. Dynamical systems signal their presence through three factors:

- they are dynamic, that is, subject to lasting changes
- they are complex, that is, depend on many factors
- they are iterative, that is, the laws that govern their behavior can be described by feedback.

Regarding non-linearity, it can be say that systems and phenomena that do not move predictably by following a clear pattern are said to be non-linear.

3.1 Chaos in Electronic Systems

For electronic systems a deterministic system is called chaotic if an infinitesimally small perturbation to its initial conditions produces a change in its behavior that

grows exponentially with time. So, it is impossible to make accurate long-term predictions about the behavior of the system. Chaotic signals are non-periodic in time domain and trajectory of the system cannot go through the same point twice [5]

Chaotic circuits can be used in:

- analog signal processing applications as a dither source to improve the performance of other blocks. For instance, dithering can be used to whiten the noise floor of $\Sigma\Delta$ modulators, as well as to reduce the (idle channel) spurious tones, which are introduced during quantization of direct current (dc) inputs (audible in voice-band applications) [6]. Also, dithering can be used to improve the integral nonlinearity of high-performance Nyquist-rate analog-to-digital converters [7].
- ranging systems, the nonperiodicity of chaotic signals, as well as the rapid decorrelation of their time-shifted sequences, make the use of chaos an interesting coding technique for high resolution radar systems [8].
- chaos-based digital communication systems as a generator of communication carriers [9].
- random number generation frequently. Although chaotic oscillator is a deterministic system, most times it is accepted as a TRNG. Since a small changes, affects its behavior, they can be thought as a noise amplificatory.

Chaotic circuits can be classified into autonomous or nonautonomous systems, depending on whether the system is able or not to self-sustain chaotic oscillations without any external excitation. Most of the IC implementations are autonomous systems. Another possible classification is between discrete-time or continuous-time, depending on whether the system evolution is described by nonlinear difference or differential equations, respectively [10].

Autonomous discrete-time systems (or discrete maps) can be generally described by the following qth (delay) order n-dimensional finite-difference equation (FDE),

$$x(k+q) = F\left[x(k+q-1),...,x(k)\right] \tag{3.1}$$

where k=0,1,2… symbolizes the discrete time variable, x(k) represents the state vector of the system at the *k*th discrete time instant and F is a n-dimensional time-invariant vector field. Autonomous continuous-time systems are defined by the ODE ordinary differential equations (ODE),

$$\frac{dx(t)}{dt} = F(x(t)) \tag{3.2}$$

where x(t) is the state vector of the system (also trajectory) and F is the nonlinear vector field that defines the direction and speed of a trajectory at every point in the state space and at every instant of time [10].

In order to exhibit chaos electronically, an autonomous circuit consisting of resistors, capacitors, and inductors must contain:

- at least one nonlinear element (sign, absolute value, hysteresis etc.)
- at least one negative resistor (to supply energy to the system)
- at least three energy-storage elements

3.2 Chua's Circuit

Chua's circuit is the simplest electronic circuit that generates chaos [11] (see Figure 3.1).

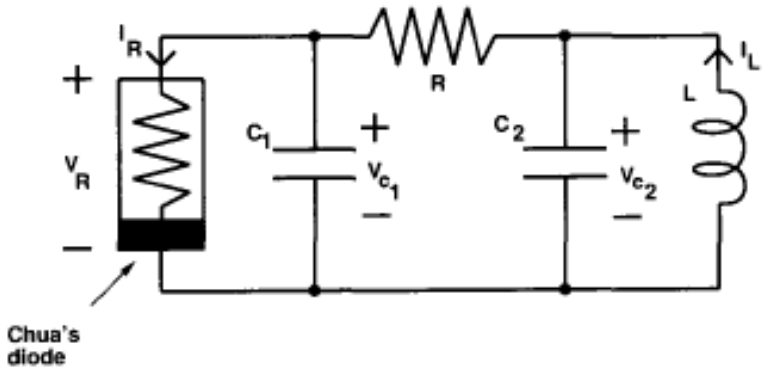

Figure 3.1: Chua's Circuit

It consists of;

- A linear inductor L
- A linear resistor R
- Two linear capacitors C1 and C2 and
- A single voltage controlled nonlinear resistor called Chua's diode.

I-V characteristic of Chua's Diode is shown in Figure 3.2.

The state equations of Chua's circuit are:

$$\frac{dv_1}{dt} = \frac{1}{C_1}\left[G(v_2 - v_1) - f(v_1)\right] \tag{3.3}$$

$$\frac{dv_2}{dt} = \frac{1}{C_2}\left[G(v_1 - v_2) + i_3\right] \tag{3.4}$$

$$\frac{di_3}{dt} = -\frac{1}{L}v_2 \tag{3.5}$$

where, $G = \frac{1}{R}$, and $f(v_1) = G_b v_1 + \frac{1}{2}(G_a - G_b)\{|v_1 + E| - |v_1 - E|\}$.

These equations form different attractors for different parameter values. In Figure 3.3 one of the attractors obtained from computer simulation of Chua's circuit is shown.

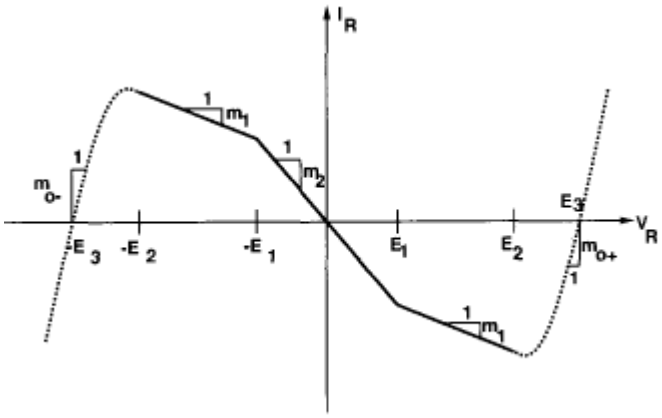

Figure 3.2: I-V characteristic of Chua's diode

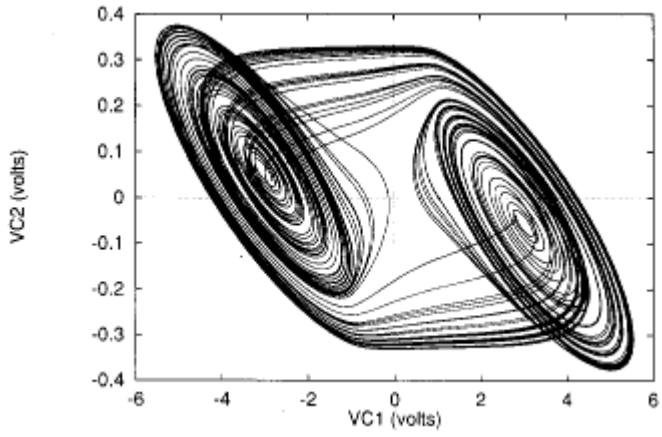

Figure 3.3: Attractor of Chua's circuit

4. CHAOTIC OSCILLATOR BASED RANDOM NUMBER GENERATOR

The ultimate goal of this research was to integrate a chaotic oscillator, and use this oscillator outputs to generate a successive bit stream that passes randomness tests. A high speed negative-g_m LC oscillator was selected as the oscillator [12]. Random bits were generated with the method described in [13].

By adding new elements to the well known LC oscillator, chaotic oscillation was obtained. The state equations of the oscillator were derived and solved by using various numeric solvers.

4.1 Construction of Chaotic Oscillator

Before substituting equations for the proposed chaotic oscillator, deriving equations for familiar RLC resonator may be helpful in order to illustrate the concepts in dynamical systems theory, and introduce ideas of stability and oscillation,.

4.1.1 RLC Resonator

The parallel-tuned RLC resonant circuit consists of two linear, lossless passive energy-storage elements (L, C) and a linear resistor R (See Figure 4.1).

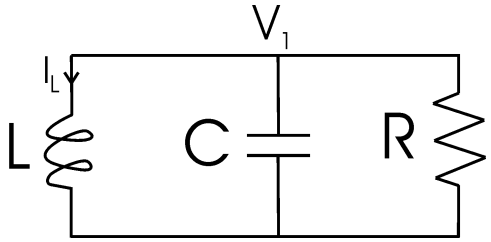

Figure 4.1: Parallel RLC circuit

This circuit can be described by a system of ordinary differential equations of the form;

$$\dot{X}(t) = F(X(t), t), \quad X(0) = X_0 \tag{4.1}$$

where $X(t)$ is called a state vector, and $F(X(t), t)$ is called a vector field. If the vector field depends only on the state, and is independent of time t, then the system is said to be autonomous and can be written as [8];

$$\dot{X} = F(X), \quad X(0) = X_0 \tag{4.2}$$

For the circuit in Figure 4.1, with the selection of V_1 and i_L as state variables, two state equations can be written as;

$$\frac{dV_1}{dt} = -\frac{1}{C} i_L - \frac{V_1}{RC} \tag{4.3}$$

$$\frac{d\dot{I}_L}{dt} = \frac{V_1}{L} \tag{4.4}$$

By applying variable transformation as $V_1 = x, i_L = y,$ and $t_n = \frac{t}{RC}$; the above state equations for the circuit become independent of time t;

$$\dot{x} = -Ry + x \tag{4.5}$$

$$\dot{y} = \frac{RC}{L} x \tag{4.6}$$

Without solving these equations, the behavior of the circuit can easily be explained with respect to the polarity of R. If R is positive then the resistor is said to be dissipative. The energy initially stored in the capacitor and inductor is dissipated, and $V_1(t)$ and $\dot{I}_L(t)$ approach zero either monotonically or in the form of exponentially decaying sinusoids. If R is negative, the resistor has negative dissipation; it supplies energy to the rest of the circuit. Energy stored in the circuit increases with time. This circuit simply oscillates when the R infinite.

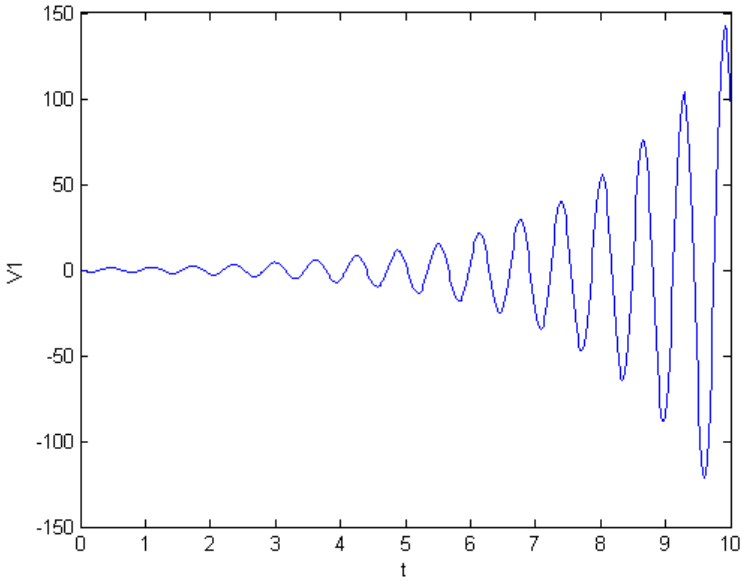

Figure 4.2: V_1 versus t, for R=-10Ω

To see the behavior of the circuit visually, equations (4.5) and (4.6) can be solved by using MATLAB. For L=C and a given initial condition-set; the typical voltage V_1 waveforms for negative, positive and infinite R situations are shown in Figure 4.2, 4.3, and 4.4 respectively.

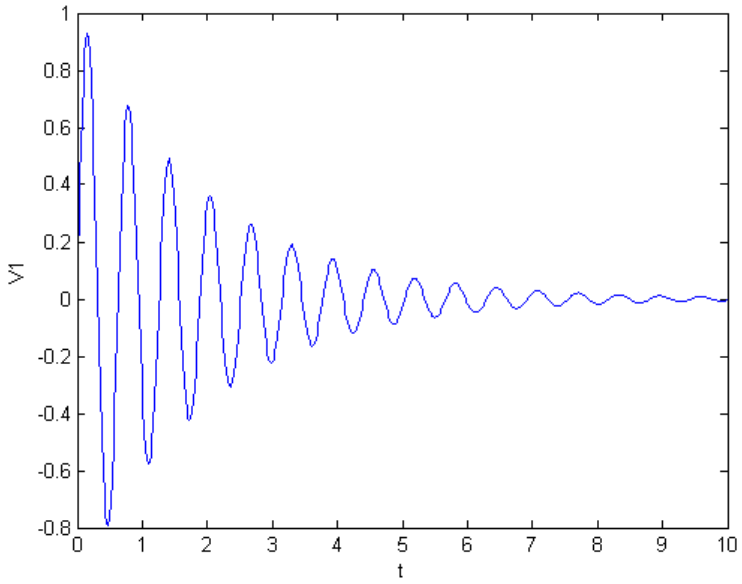

Figure 4.3: V_1 versus t, for R=10Ω

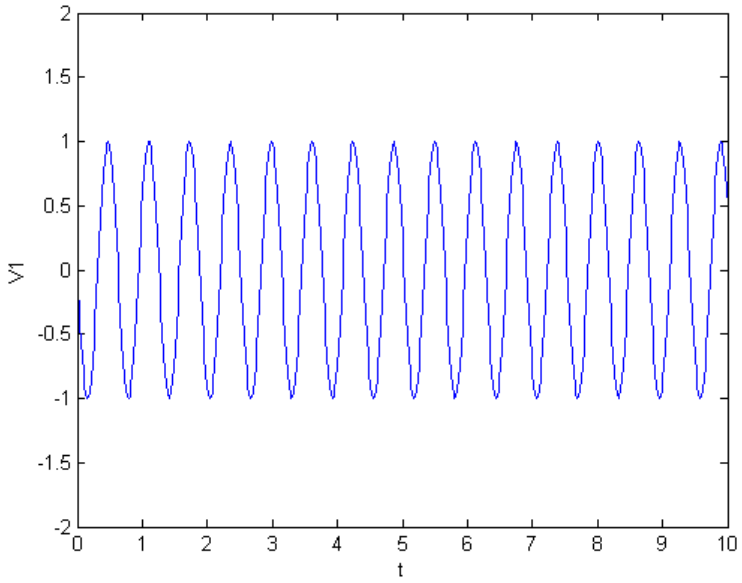

Figure 4.4: V_1 versus t, for R=∞

The trajectories of a system are also shown in Figure 4.5a,b for positive, and negative R situations respectively (In infinite R situation, the trajectory becomes a simple circle).

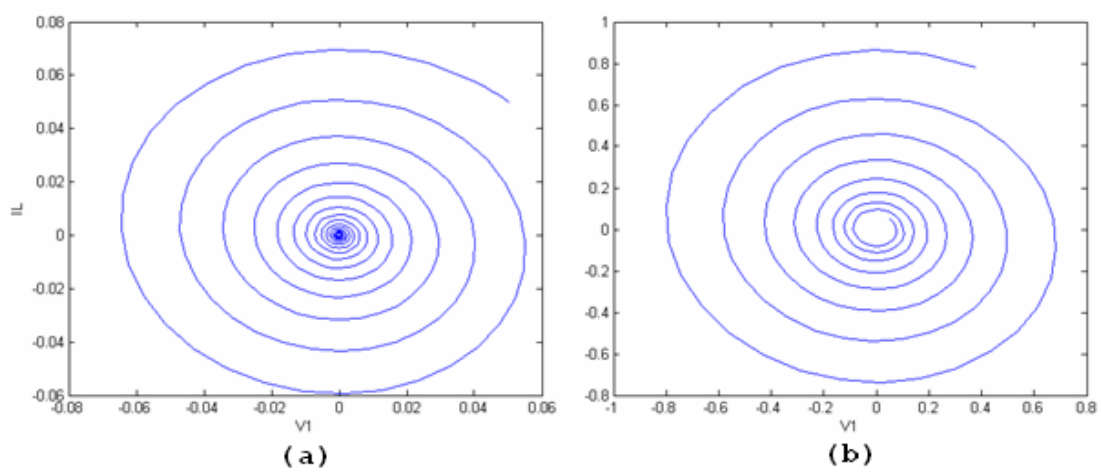

Figure 4.5: The vector field of a system (a) for positive R; (b) for negative R

For positive R values, trajectories are pushed together as they track spiral towards the origin, which is the equilibrium point of the RLC circuit (Figure 4.5-a). For negative R values trajectories are stretched apart as they track a spiral away from the equilibrium point - the origin (Figure 4.5-b). Although the last situation corresponds to a continuous oscillation, it neither has a physical meaning (A real oscillator must possess a nonlinearity to limit the amplitude of the oscillation), nor is sufficient to produce a chaos.

Indeed, in physical systems there is always a positive R that comes from the parasitic resistance of the inductor. Thus the energy stored in the resonator is dissipated even without the need of an external positive resistor. In order to compensate resonator losses and achieve oscillation, active devices that show negative resistance must be used.

4.1.2 Basic LC Oscillator

Most LC oscillators employ a cross-coupled transistor pair as a negative resistance. Figure 4.6 shows one of the basic configurations of classical LC oscillators. This configuration provides a symmetrical nature to the oscillator.

Since the circuit has a symmetric nature, when node A has a common mode voltage V_C and differential voltage $+V_1$, the voltage of node B can be written as V_C-V_1.

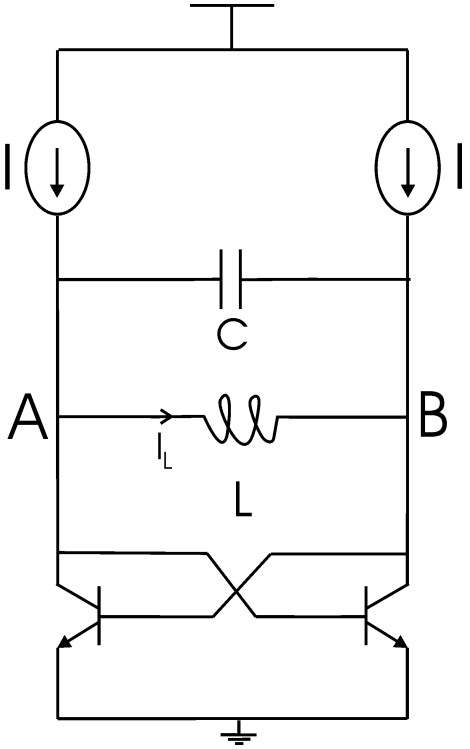

Figure 4.6: Implementation of negative resistance with cross coupled transistors

By applying KCL to node A and B, two equations can be written;

$$C\frac{d(2V_1)}{dt} + I_L + I_S e^{\frac{V_C - V_1}{V_T}} - I = 0 \tag{4.7}$$

$$-C\frac{d(2V_1)}{dt} - I_L + I_S e^{\frac{V_C - V_1}{V_T}} - I = 0 \tag{4.8}$$

where I_S and V_T are the reverse saturation current of the transistor and thermal voltage respectively. By adding, and subtracting (4.7) and (4.8) we conclude with;

$$C\frac{dV_1}{dt} = -\frac{1}{2}i_L - \frac{I_S e^{\frac{V_C}{V_T}}}{4}(e^{-\frac{V_1}{V_T}} - e^{\frac{V_1}{V_T}}) \tag{4.9}$$

$$I_S e^{\frac{V_C}{V_T}} = \frac{2I}{(e^{-\frac{V_1}{V_T}} + e^{\frac{V_1}{V_T}})} \tag{4.10}$$

Rearranging (4.9) by using (3.10) and the tanh definition, one of the state equations of the circuit is reached:

$$C\frac{dV_1}{dt} = -\frac{1}{2}I_L + \frac{I}{2}\tanh(V_1 / V_T) \tag{4.11}$$

The other state equation comes from the voltage across the inductor L:

$$L\frac{dI_L}{dt} = 2V_1 \tag{4.12}$$

If (V_1/V_T), and I_L are chosen as state variables and normalization is applied to time, equations (4.11) and (4.12) can be rearranged with using $(V_1/V_T) = x$, $I_L = y$ and $t_n = \frac{t}{\sqrt{LC}}$

$$\dot{x} = -\frac{1}{2V_T}y + \frac{I}{2V_T}\tanh(x) \tag{4.13}$$

$$\dot{y} = 2V_T x \tag{4.14}$$

These equations form the vector field of the circuit in Figure 4.6, and can be solved by using MATLAB. For I=1mA, V_T=25mV, and a given initial condition-set, the obtained waveform of the state variable (V_1/V_T) is shown in Figure 4.7.

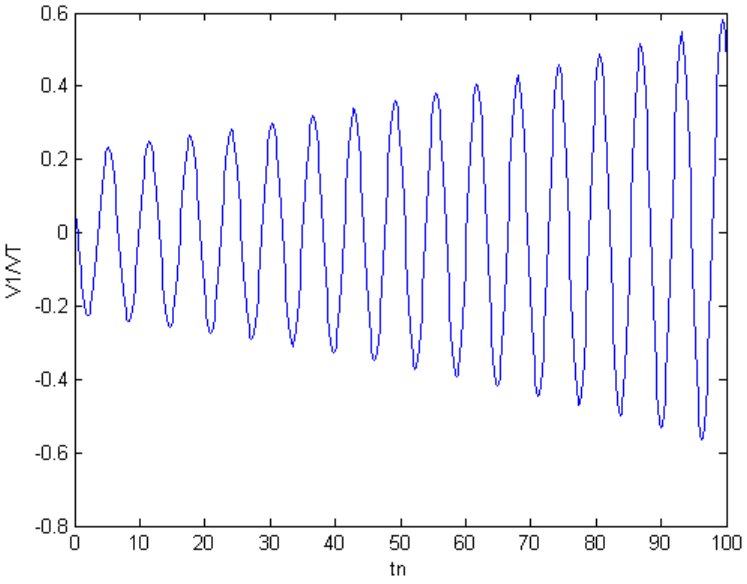

Figure 4.7: Plot of (V_1/V_T) for I=1mA, and V_T=25mV

Since there is no amplitude limiting mechanism included, again increasing oscillation is observed.

4.1.3 Block Level Chaotic Oscillator

As discussed in Chapter 3, in order to produce chaos in an electronic system three conditions must be satisfied:

- the system must have at least one nonlinear element

- at least one locally active resistor
- at least three state variables.

By making the appropriate modifications in the circuit seen in Figure 4.6, a sustained chaotic oscillation can be maintained. In Figure 4.8 the new version of the RLC circuit is shown. A resistor (R_2) and a capacitor (C_2) couple are included in order to produce one more state variable. The Signum function (sgn) is employed as the nonlinear element. It maintains nonlinearity by sourcing a current to node V_2 or sinking a current at node V_2 according to the polarity of the node voltage V_1.

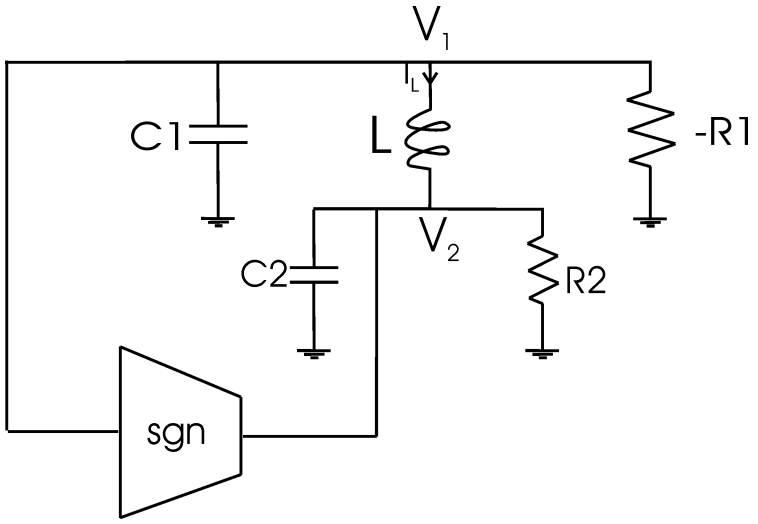

Figure 4.8: Chaotic circuit

For the circuit shown in Figure 4.8, the state equations can be written as;

$$C_1 \frac{dV_1}{dt} = -I_L + \frac{V_1}{R_1} \tag{4.15}$$

$$C_2 \frac{dV_2}{dt} = \text{sgn}(V_1) + I_L - \frac{V_2}{R_2} \tag{4.16}$$

$$L \frac{dI_L}{dt} = V_1 - V_2 \tag{4.17}$$

If V_1, RI_L, and V_2 are chosen as variables, equations (4.15-17) can be rearranged using the variable transformation; $V_1 = x$, $RI_L = y$, $V_2 = z$ and time normalization $t_n = \frac{t}{RC}$.

$$\dot{x} = -y + a_1 x \tag{4.18}$$

$$\dot{y} = \frac{R^2}{(L/C)}(x - z) \tag{4.19}$$

$$\dot{z} = R\,\mathrm{sgn}(x) + y - a_2 z \qquad (4.20)$$

where $a_1 = R/R_1$, and $a_2 = R/R_2$

These equations can be solved by using MATLAB numeric solver ODE45. With the selection of $\frac{R^2}{(L/C)} = 1$, and L=C, it is seen that chaotic oscillation occurs for some values of a_1, and a_2. In Table 4.1, three sets of variables (a_1, a_2) that provide chaotic oscillation is shown.

Table 4.1: Different parameter-sets for which chaotic oscillation occurs

	a_1	a_2
1	0.6	2
2	0.8	1.6
3	1	1.4

Figure 4.9 shows the waveform of node voltage V_1 for the first parameter-set in Table 4.1.

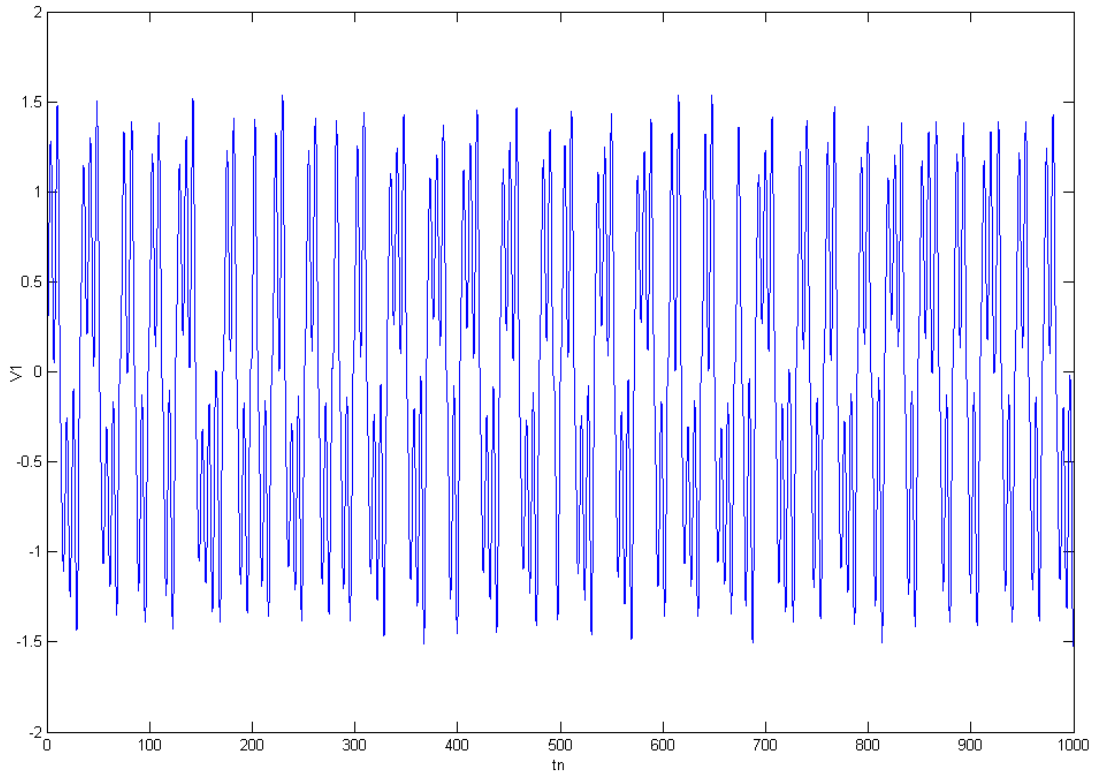

Figure 4.9: V_1 for a_1=0.6, and a_2=2

Figure 4.10 is a plot of V_1 versus V_2, in other words the trajectory of the system in Figure 4.8.

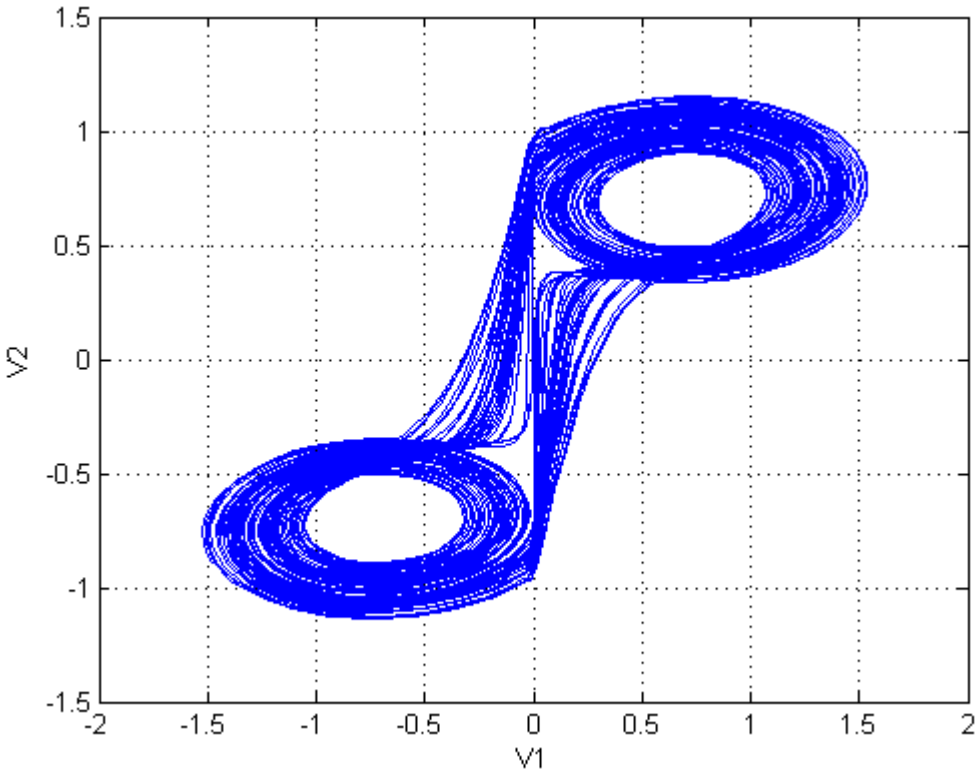

Figure 4.10: Trajectory of the chaotic circuit in Figure 4.8

4.1.4 Bipolar Transistor Chaotic-LC Oscillator

By combining the concepts of classical negative-gm LC oscillator and the circuit that uses the signum function, the final chaotic oscillator has been reached (Figure 4.11) [12]. This circuit has been derived from the classical negative-g_m LC oscillator, by adding a parallel RC_3 section (like the R_2C_2 section in Figure 4.8), and a differential pair stage, to realize the signum-like function.

Similar to the classical LC oscillator in Figure 4.6, if node A has a common mode voltage of V_C and a differential voltage component of $+V_1$, then node B has a common mode voltage of V_C but differential voltage of $-V_1$. Similarly, the voltage values of node C, and D can be written as V_C+V_2 and V_C-V_2 respectively, as a result of the symmetrical nature of the circuit.

The inductance, placed between nodes A and C has a current value of $I_L - i_L$, whereas the other one has $I_L + i_L$.

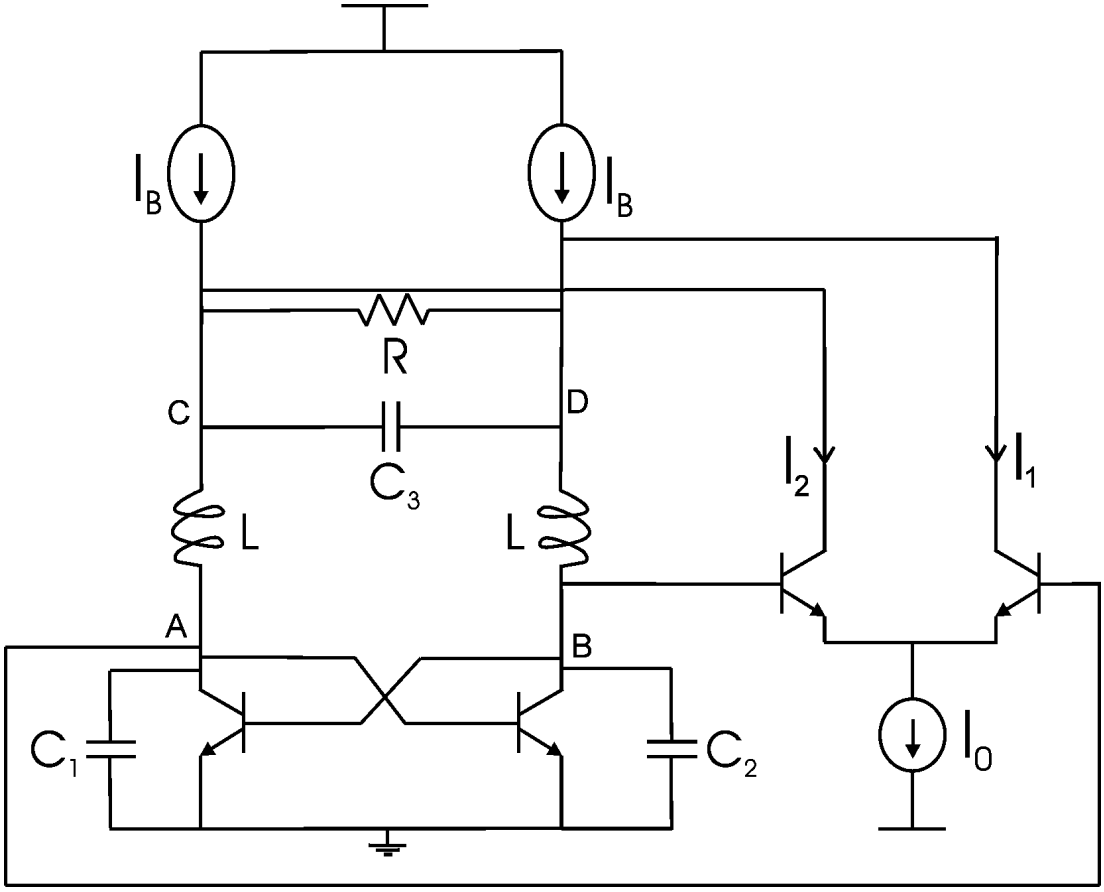

Figure 4.11: Chaotic LC oscillator

Applying KCL at node A, B, C and D, with the assumption of $C_1=C_2=C_3=C$ yields the following equations;

$$C\frac{d(V_C+V_1)}{dt}+I_S e^{\frac{V_C-V_1}{V_T}}-I_L+i_L=0 \tag{4.21}$$

$$C\frac{d(V_C-V_1)}{dt}+I_S e^{\frac{V_C+V_1}{V_T}}-I_L-i_L=0 \tag{4.22}$$

$$I_L-i_L+I_2+\frac{2V_2}{R}+C\frac{d(2V_2)}{dt}-I_B=0 \tag{4.23}$$

$$I_L+i_L+I_1-\frac{2V_2}{R}-C\frac{d(2V_2)}{dt}-I_B=0 \tag{4.24}$$

where I_S and V_T are the reverse saturation current of the transistor and the thermal voltage respectively.

By rearranging (4.21) and (4.22), the following equations can be written:

$$C\frac{dV_1}{dt}=-i_L+\frac{I_S e^{\frac{V_C}{V_T}}}{2}\left(e^{\frac{V_1}{V_T}}-e^{-\frac{V_1}{V_T}}\right) \tag{4.25}$$

$$C\frac{dV_C}{dt} = I_L - \frac{I_S e^{\frac{V_C}{V_T}}}{2}\left(e^{\frac{V_1}{V_T}} + e^{-\frac{V_1}{V_T}}\right) \tag{4.26}$$

By combining (4.23) and (4.24),

$$C\frac{dV_2}{dt} = -\frac{V_2}{R} + \frac{i_L}{2} + \frac{I_1 - I_2}{4} \tag{4.27}$$

is achieved. In this equation $I_1 - I_2$ represents the output current difference of the differential pair and can be written in terms of the tail current I_0 and the differential input voltage as, $V_A - V_B = 2V_1$.

$$I_1 - I_2 = I_0 \tanh(\frac{V_1}{V_T}) \tag{4.28}$$

Equation (4.27) can be rearranged by using (4.28), as

$$C\frac{dV_2}{dt} = -\frac{V_2}{R} + \frac{i_L}{2} + \frac{I_O}{4}\tanh\left(\frac{V_1}{V_T}\right) \tag{4.29}$$

Equations (4.25), (4.26) and (4.29) are the state equations of the chaotic oscillator circuit, but there is one more equation that comes from the voltage-current relationship of inductance:

$$\left(V_C + V_2\right) - \left(V_C + V_1\right) = L\frac{d(I_L - i_L)}{dt} \tag{4.30}$$

By making the appropriate simplifications;

$$V_1 - V_2 = L\frac{d(i_L)}{dt} \tag{4.31}$$

is reached.

The 4 equations (4.25, 4.26, 4.29, and 4.31) are the state equations of the circuit shown in Figure 4.11.

By scaling all voltage values with an arbitrary scaling voltage V_s; choosing state variables as $x = \frac{V_1}{V_S}$, $y = \frac{\dot{I}_L R}{V_S}$, $z = \frac{V_2}{V_S}$, and $x_C = \frac{V_C}{V_S}$; scaling time t with RC ($t_n = \frac{t}{RC}$); and taking $R = \sqrt{\frac{L}{C}}$; the state equations are transformed into a simple and dimensionless form:

$$\dot{x} = -y + \frac{be^{ax_C}}{2}\left(e^{ax} - e^{-ax}\right) \tag{4.32}$$

$$\dot{y} = x - z \tag{4.33}$$

$$\dot{x} = -z + \frac{y}{2} + \frac{c}{4}\tanh(ax) \tag{4.34}$$

$$\dot{x}_C = c - \frac{c_S e^{ax_C}}{2}\left(e^{ax} + e^{-ax}\right) \tag{4.35}$$

where $a = V_S / V_T$, $b = I_S R / aV_T$, $c = I_0 R / aV_T$, and $d = (I_B - \frac{I_0}{2})R / aV_T$.

These state equations can be solved by using numeric solvers, and again for different sets of parameters a, b, c and d, chaotic oscillation occurs. Figure 4.12 and 4.13 are obtained for a=0.5, b=10^{-4}, c=8.8 and d=2.

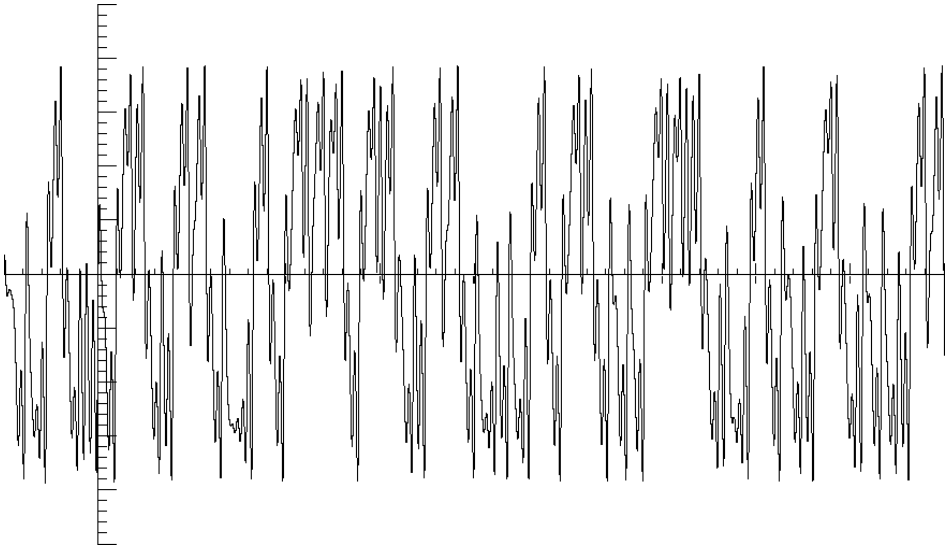

Figure 4.12: Numeric analyses result of the BJT Chaotic-LC oscillator

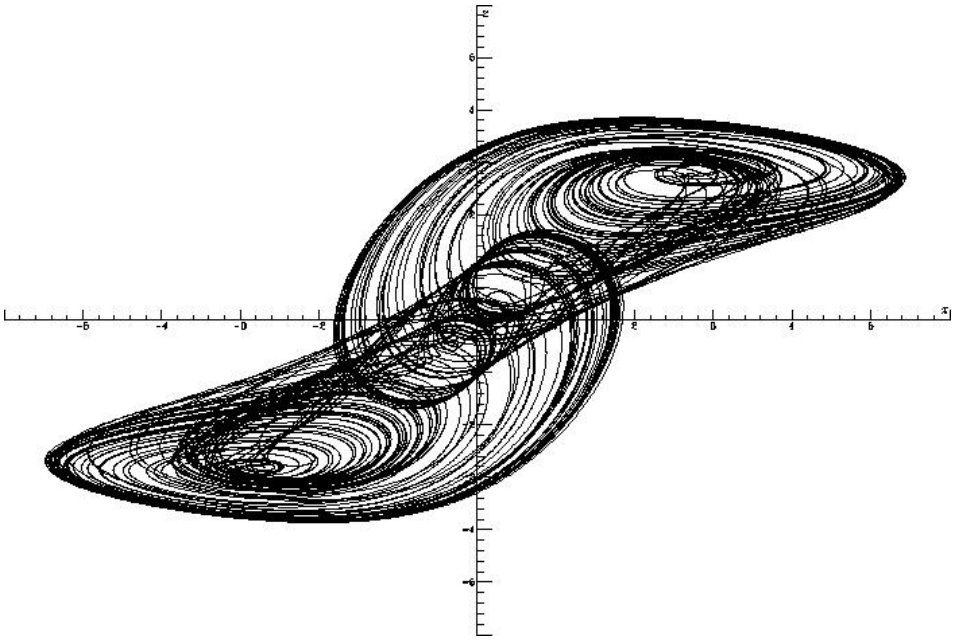

Figure 4.13: Trajectory of the BJT Chaotic-LC Oscillator (x versus z)

4.1.5 MOS Transistor Chaotic-LC Oscillator

The MOS transistor version of the chaotic-LC oscillator is also possible (Figure 4.14)

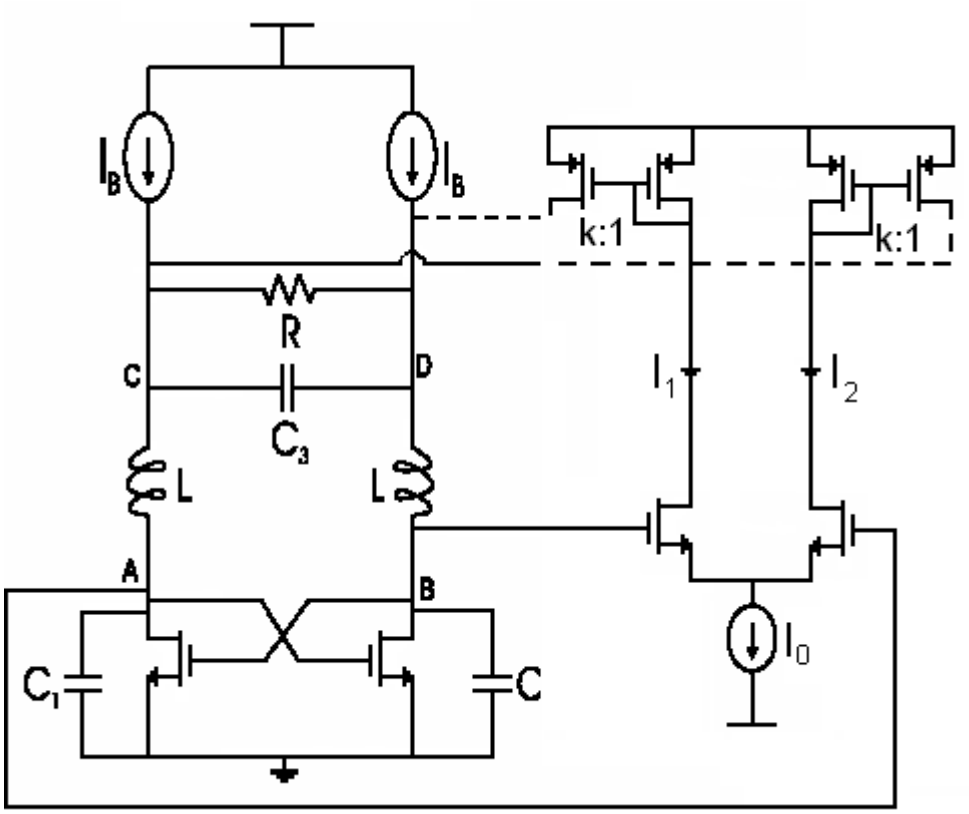

Figure 4.14: MOS transistor chaotic-LC oscillator

Comparing Figure 4.11 and 4.14, it can be seen that transistor type is not the only difference. While currents I_1 and I_2 are sunk from nodes C and D in Figure 4.11, these currents are sourced to nodes C and D with a gain of k in Figure 4.14.

The nonlinearity in this chaotic oscillator, whose basic structure has been explained in Chapter 4.1.3, has been established by using a transimpedance amplifier block with a signum functionality. While the bipolar differential pair maintains this functionality with a good approximation thanks to the exponential nature of the BJTs, the MOS differential pair cannot achieve a good approximation due to the square-law behavior of MOS transistors. However, it has been seen that in the MOS differential pair, the requested approximation can be achieved when the currents I_1 and I_2 are multiplied with a gain of k. The circuit structure has been modified accordingly. The currents I_1 and I_2 could be mirrored once more – this time with NMOS transistors – in order to make a structure, which would be more similar to the BJT version. This modification was not needed.

The currents being sourced this way have made another modification in connections necessary. In the BJT version, the transistor whose base is connected to node A

has its collector at node D, which causes current to be sunk from D according to the voltage value at node A. This connection has been changed in the MOS version and current is being sourced to C according to the voltage value at node A.

Applying KCL at node A, B, C and D, with the assumption of $C_1=C_2=C_3=C$ yields the following equations:

$$C\frac{d(V_C+V_1)}{dt}+\frac{\beta}{2}(V_C-V_1-V_{TN})^2-I_L+i_L=0 \tag{4.36}$$

$$C\frac{d(V_C-V_1)}{dt}+\frac{\beta}{2}(V_C+V_1-V_{TN})^2-I_L-i_L=0 \tag{4.37}$$

$$I_L-i_L-kI_2+\frac{2V_2}{R}+C\frac{d(2V_2)}{dt}-I_B=0 \tag{4.38}$$

$$I_L+i_L-kI_1-\frac{2V_2}{R}-C\frac{d(2V_2)}{dt}-I_B=0 \tag{4.39}$$

where $\beta=\mu_n C_{ox}\left(\frac{W}{L}\right)_{cross-coupled}$ and V_{TN} is the threshold voltage of the NMOS transistors.

By rearranging (4.36) and (4.37), the following equations can be written:

$$C\frac{dV_1}{dt}=-i_L+\beta(V_C-V_{TN})V_1 \tag{4.40}$$

$$C\frac{dV_C}{dt}=I_L+\frac{\beta}{2}\left[(V_C-V_{TN})^2+V_1^2\right] \tag{4.41}$$

By combining (4.38) and (4.39):

$$C\frac{dV_2}{dt}=-\frac{V_2}{R}+\frac{i_L}{2}+\frac{k}{4}(I_1-I_2) \tag{4.42}$$

(4.42) is achieved. In equation (4.42) (I_1-I_2) is the output current difference of the MOS differential pair and by rearranging (4.42) using the output current-input differential voltage relationship of the MOS differential pair, the following equation can be derived:

$$C\frac{dV_2}{dt}=-\frac{V_2}{R}+\frac{i_L}{2}+\frac{k}{4}\begin{cases}2\beta_n\left(\sqrt{\frac{I_0}{\beta_n}-V_1^2}\right)V & ,V_1^2\le\frac{I_0}{2\beta_n}\\ \pm I_0\end{cases} \tag{4.43}$$

Here, $\beta=\mu_n C_{ox}\left(\frac{W}{L}\right)_{diff-pair}$

By scaling all voltage values with an arbitrary scaling voltage V_s; choosing state variables as; $x = \frac{V_1}{V_S}$, $y = \frac{\dot{I}_L R}{V_S}$, $z = \frac{V_2}{V_S}$, and $x_C = \frac{V_C}{V_S}$; scaling time t with RC ($t_n = \frac{t}{RC}$); and taking $R = \sqrt{\frac{L}{C}}$; the state equations are transformed into a simple and dimensionless form:

$$\dot{x} = -y + b(x_C - c_{th})x \tag{4.44}$$

$$\dot{y} = x - z \tag{4.45}$$

$$\dot{z} = -z + \frac{y}{2} + \frac{k}{4}\begin{cases} 2b_n\left(\sqrt{\frac{c_0}{b_n} - x^2}\right)x & , x^2 \le \frac{c_0}{2b_n} \\ \pm c_0 \end{cases} \tag{4.46}$$

$$\dot{x}_C = c_b - \frac{b_0}{2}\left[(x_C - c_{th})^2 + x^2\right] \tag{4.47}$$

where $b = \beta RV_S$, $b_n = \beta_n RV_S$ $c_{th} = V_{th}/V_S$, $c_0 = I_0 R/V_S$ and $c_b = I_L R/V_S$.

These state equations again generate chaos for different set of parameters. In Table 4.2, four sets of parameters that generate chaotic oscillation are shown.

Table 4.2: Parameter sets that generate chaotic oscillation

	b	b_n	c_{th}	c_0	c_b	k
1	1	2.6	1	2.2	0.7	1
2	1	0.4	1	0.5	0.7	5
3	1	0.3	1	0.3	0.7	7
4	0.5	0.3	1	0.3	0.7	9

Figure 4.15, and Figure 4.16 show the waveform of the state variable x, and the trajectory of the system (x versus z) respectively for the second parameter set in Table 4.2.

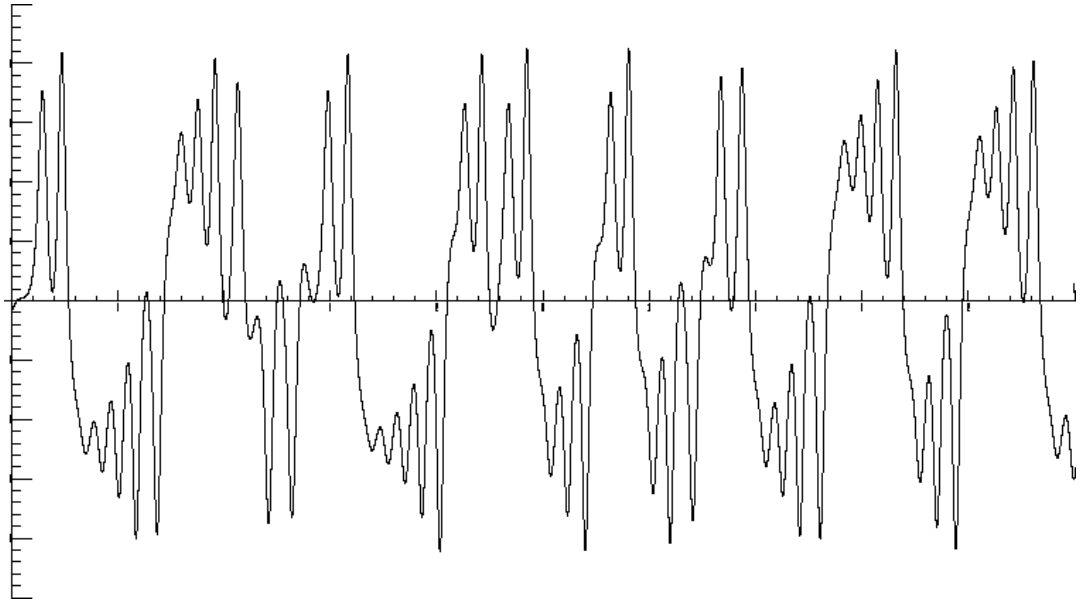

Figure 4.15: Waveform of the State Variable x versus time

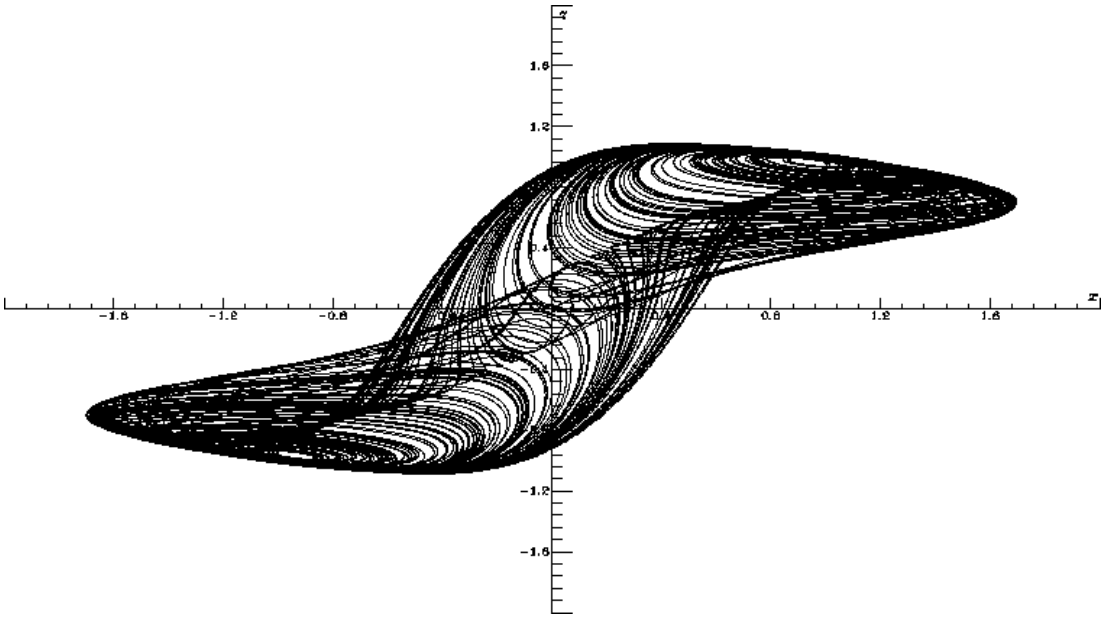

Figure 4.16: Trajectory of the MOS Chaotic-LC Oscillator

4.2 Random Bit Generation

The chaotic oscillator presented in the last section, is a continuous time autonomous system, which exhibits a double-scroll attractor. This chaotic oscillator has been used as a source for the RBG.

Bits are produced by using the V_A-V_B voltage difference which is one of the state variables of the chaotic oscillator. To generate '1's and '0's, two comparators with different references are used [13] (Figure 4.17). These two comparators divide the state space into three sub regions; V_0, V_{tr}, and V_1. One of the comparators is

responsible for detecting the jumps between the scrolls, and the reference of this comparator has been set to the mid-value of the oscillation. The other comparator is used for detecting the crossings inside one scroll, and the reference of this comparator has been selected as a design variable.

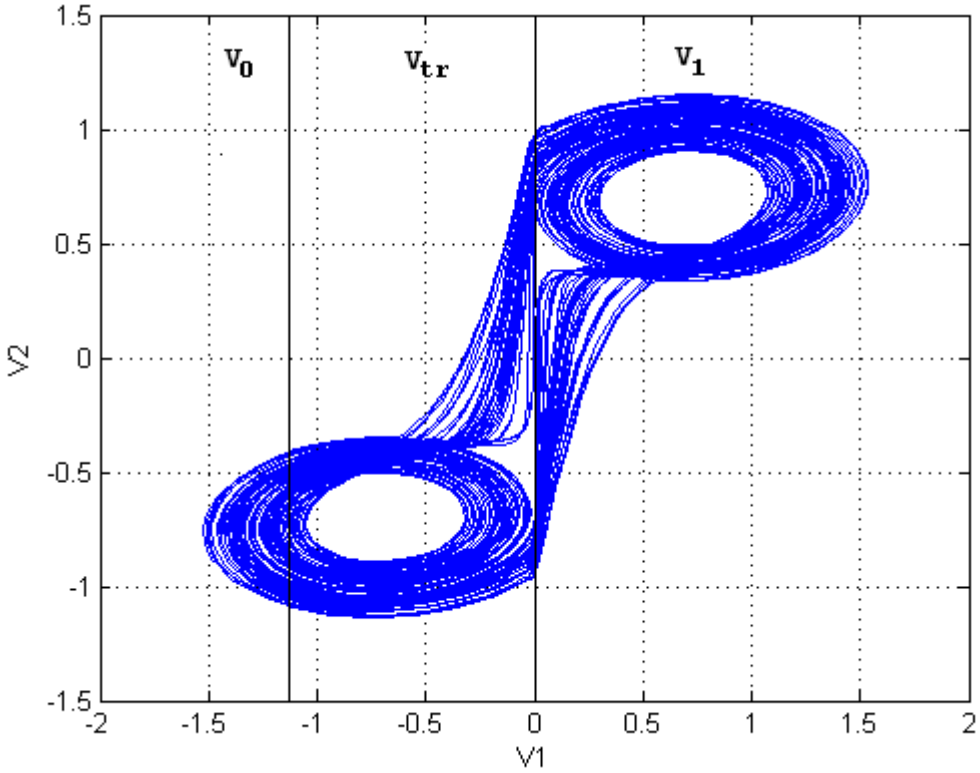

Figure 4.17: Constructed subspaces with comparator references

A '1' bit is generated when the trajectory passes from region V_{tr} to region V_1 and a '0' bit is generated when the trajectory passes from region V_{tr} to region V_0.

The time domain representation of the reference placement is shown in Figure 4.18. In Figure 4.18, the x-axis is the voltage signal (V_A-V_B), which is one of the state variables of the system described in section 4.1.4

It is easy to see that by using the bit generation procedure that has just been explained, and applying the comparator references as seen in Figure 4.18, the resulting number of '1's will be significantly less than that of '0's. This nonsymmetrical replacement of references can be overcome by a proper selection of the reference of comp0 (ref0), and applying Von-Neumann de-skewing algorithm as explained in chapter 2.

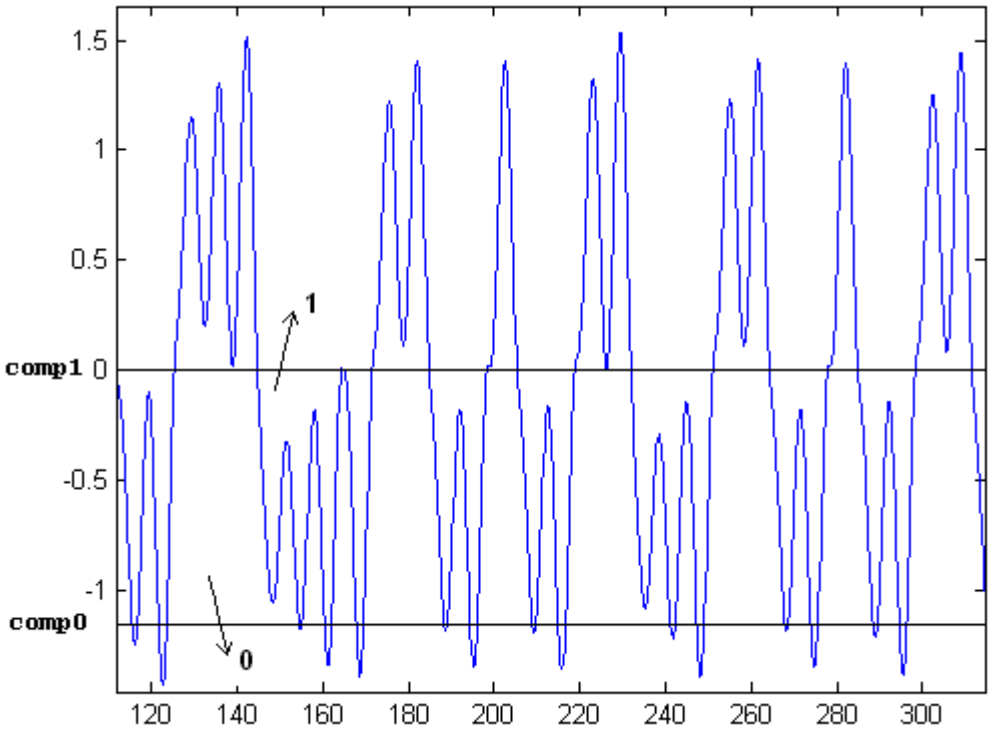

Figure 4.18: Placement of comparator references

Up to here, the ‘1’s have been generated at the output of the comparator comp1 at 0 to 1 crossings, and ‘0’s at the output of comparator comp0 at 1 to 0 crossings. It is obvious that this is not an appropriate way of outputting random numbers, therefore another method should be found.

4.2.1 Construction of the Clock Signal

In data transport protocols, it is common to synchronize the data on the line with a synchronization signal, namely the clock. Therefore combining the generated ‘1’s and ‘0’s in one signal and defining a clock signal would be a better way of outputting random bits.

The clock signal must point to the meaningful bits. As a result, it has to be constructed using the comparator outputs.

The first step of clock generation is to make the two comparators similar in the way they generate ‘1’s and ‘0’s. In order to achieve that, the reference and signal inputs of comp0 have been interchanged. Thus comp0 generates ‘0’s during 0 to 1 crosses; while comp1 keeps generating ‘1’s during 0 to 1 crossings.

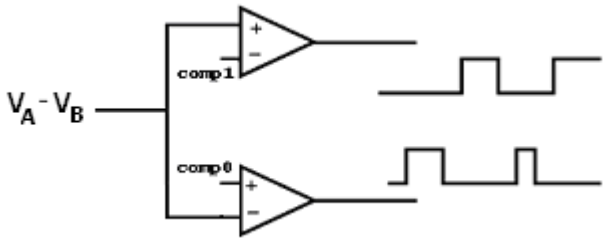

Figure 4.19: Connections of comparators

This new arrangement gives us an opportunity to combine the two comparator outputs by using an OR function. The output of this OR function is the clock signal that we need.

4.2.2 Combining Bits In One Signal

Since the two comparators are taking the same signal as input and reference of comp0 is smaller than comp1; the output of comp1 has already been set to '0' whenever comp0 produces '0'. This means that sampling the output of comp1 with the constructed clock signal gives us the random bit stream that has been generated by system the (Figure 4.20).

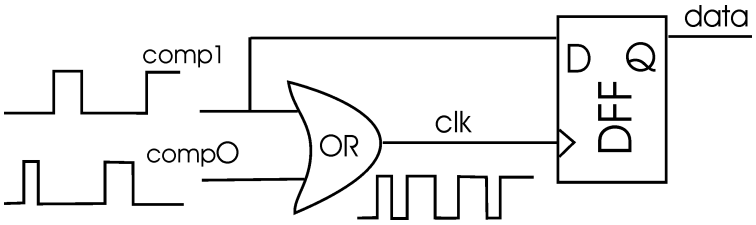

Figure 4.20: construction of clock and data signals

4.2.3 Implementing Von Neumann's De-Skewing Algorithm

As explained in chapter 2, generated bits may have cross or auto correlation related defects. For example, if the reference of comp0 is selected improperly, the resulting number of '0's would be significantly higher that that of '1's. De-skewing can be applied to the generated bits to prevent correlation related defects. Von Neumann's well known de-skewing algorithm was implemented to improve the statistical properties of the produced bits. With this de-skewing algorithm "01" sequences are converted into "0"; "10" sequences are converted into "1"; "00" and "11" sequences are discarded.

Although this de-skewing would change the generated bit sequences, since the meaningful bits have been defined by the generated clock signal, the manipulations to implement Von Neumann's de-skewing algorithm must be applied to the clock

signal. The complete block level diagram, where de-skewing algorithm is realized, is shown in Figure 4.21.

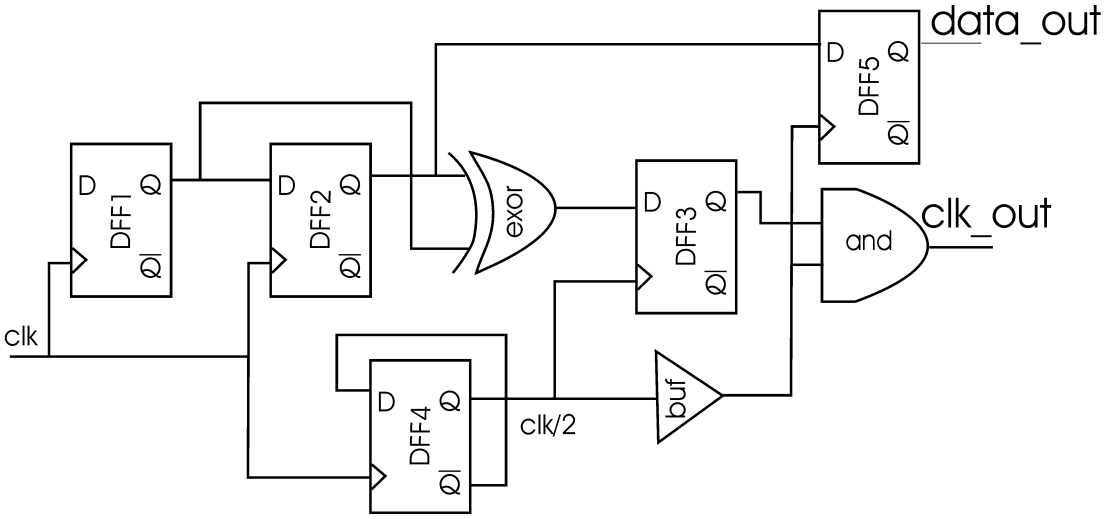

Figure 4.21: Realization of Von-Neumann's De-Skewing Algorithm

In figure 4.21:

- Data signals have been stored into two consecutive DFFs (DFF1, DFF2).
- To eliminate the "00"s and "11"s and keep the "01"s and "10"s, the outputs of these two DFFs have been EXORed.
- To process the non-overlapping bit pairs only, the clock signal has been divided by two again using DFF (DFF4).
- To get the final clock signal (clk_out), outputs of EXOR and DFF4 have been combined by using AND function. If the EXOR operation results "0", the signal "clk/2" is disabled by this AND gate.
- DFF5 has been used to synchronize the data signal with the newly constructed clock signal, and it produces the final data.

4.3 IC Implementation

In the first steps of the design process, AustriaMicroSystems (AMS) 0.35μm SiGe BiCMOS process had been chosen for production and the oscillator circuits had been designed and simulated by using the parameters of this process. Afterwards, out of financial reasons, the process to be used was changed to the IHP 0.25μm SiGe-C BiCMOS process, and the design was completed in this process and sent along for production.

It was proven numerically in section 4.1 that the designed oscillators can generate chaotic oscillation. By choosing an appropriate parameter-set out of the sets that

generate chaotic oscillation and using the definitions of these parameters, the values of the circuit elements have been identified. Although different sets of parameters produce chaos in numeric analyses, some parameter sets are not meaningful in physical basis. In other words, not all of the parameter sets that numerically generate chaos can produce suitable L, C and R values for IC implementation.

With the existence of transistors with high transit frequency, the RBG blocks were implemented using standard ECL and CML structures.

Cadence Spectre circuit simulator was used as the simulator. Supply voltages were ±1.2V. Simulations have been verified for various corner parameters, and temperature was swept from -20 ^{0}C to 50 ^{0}C.

4.3.1 Bipolar Chaotic Oscillator

R, L, C, I_B, and I_0 were calculated for the parameter values a=0.5, b=10-4, c=8.8, and d=2. With using definitions of these four parameters - $a = V_S / V_T$, $b = I_S R / aV_T$, $c = I_0 R / aV_T$, and $d = (I_B - \frac{I_0}{2}) R / aV_T$ - and choosing L=11.7nH we conclude with the other element values as R=180Ω, C=350fF, and DC bias currents I_0=600μA, I_B=710μA.

L=11.7nH was chosen to be able to use standard inductances supplied by the vendor. By doing this, we can avoid design and layout of inductances customly. Since, these parameters do not restrict the selection of transistor geometries, transistor geometries were determined in such a way that maximum transit frequency and current gain occur for the transistors at given current levels.

After first simulations:

- I_B was changed to 800μA
- C values were reduced in order to compensate parasitic capacitances at node A,B,C and D
- In contrast to the differential pair, the cross coupled stage transistors were changed with large area ones, to be able to reduce the base resistance which was not included in the state equations

Figure 4.22 and 4.23 show the observed phase space corresponding to (V_A-V_B) versus (V_C-V_D) and time domain waveform of (V_A-V_B) respectively.

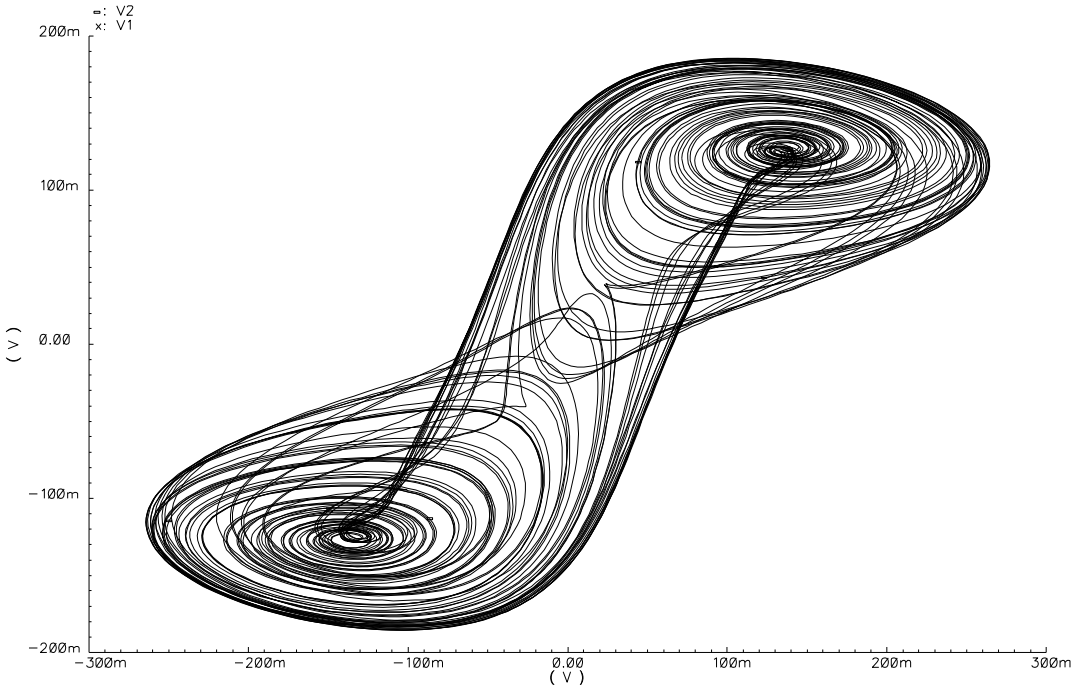

Figure 4.22: Phase Space (V_A-V_B) versus (V_C-V_D) of the BJT Chaotic Oscillator

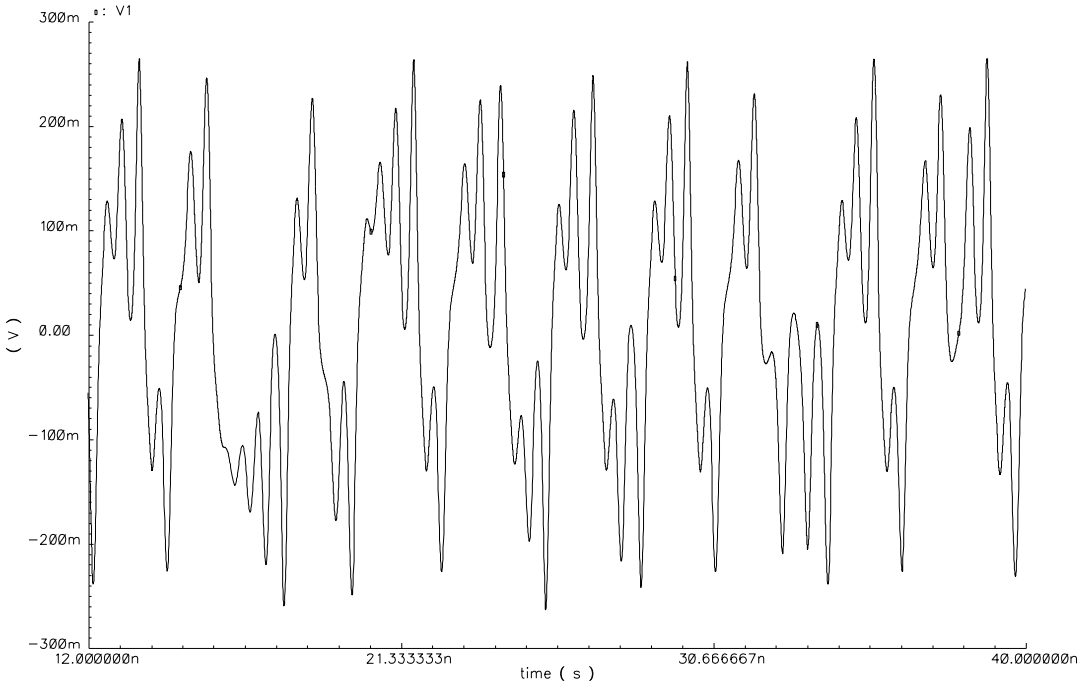

Figure 4.23: Waveform of (V_A-V_B)

4.3.2 MOS Transistor Chaotic-LC Oscillator

Although for the bipolar version, there are some parameter sets that result suitable element values for IC implementation, for the MOS version, no parameter set that was found that supplies acceptable L values could not be found. Despite this known inconvenience, MOS version was constructed, and simulated by using the AMS

model parameters. R, L, C, I_B and I_0 values were calculated for the second parameter set of Table 4.2, i.e. -b=1, b_n=0.4, c_{th}=1, c_0=0.5, c_b=0.7, and k=5-. By using definitions of these parameters - $b = \beta R V_S$, $b_n = \beta_n R V_S$, $c_{th} = V_{th} / V_S$, $c_0 = I_0 R / V_S$ and $c_b = I_L R / V_S$ - we conclude with passive element values as R=1kΩ, C=10pF, L=10nH, and bias currents as I_0=250μA, I_B=-275μA. Since I_B results a negative value, the direction of current sources I_B were changed, and the design was finalized.

Figure 4.24 and 4.25 show the Spectre simulation results of circuit in Figure4.14. In Figure 4.24 phase space corresponding to (V_A-V_B) versus (V_C-V_D) and in Figure 4.25 time domain waveforms of (V_A-V_B) are shown.

The large L values result an obligation of using off-chip inductance, and reduce the oscillation frequency. Because of these inconveniences the MOS version chaotic-LC oscillator was not chosen as the golden version, and the design was later carried on by using the bipolar version.

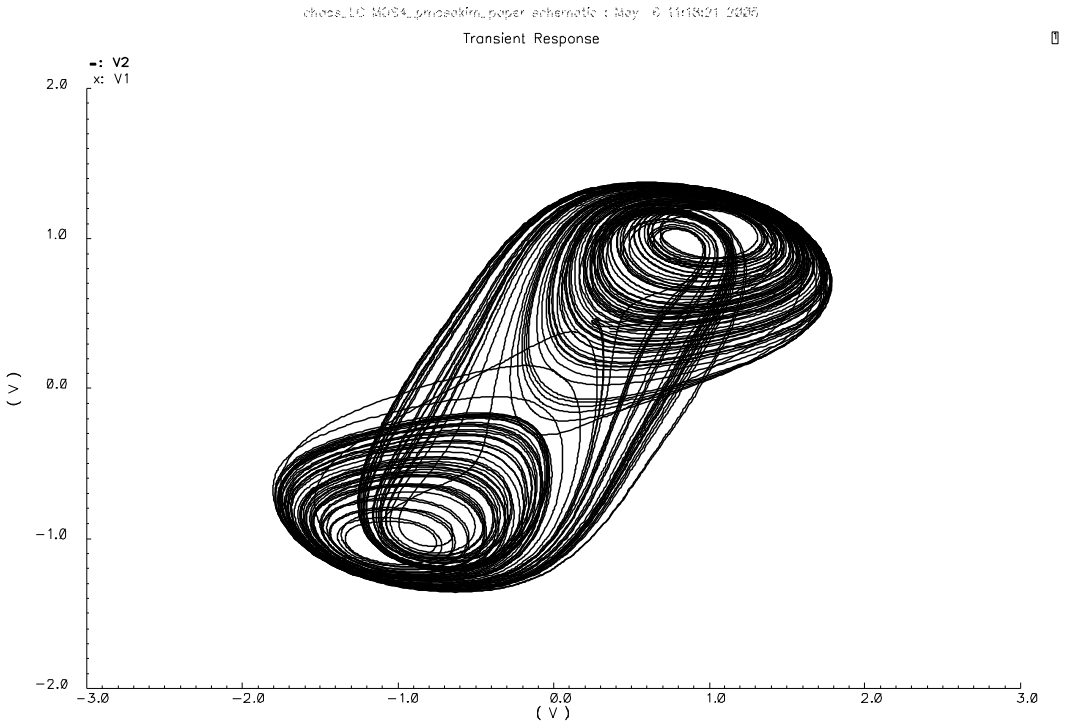

Figure 4.24: Phase Space of the MOS Transistor Chaotic Oscillator

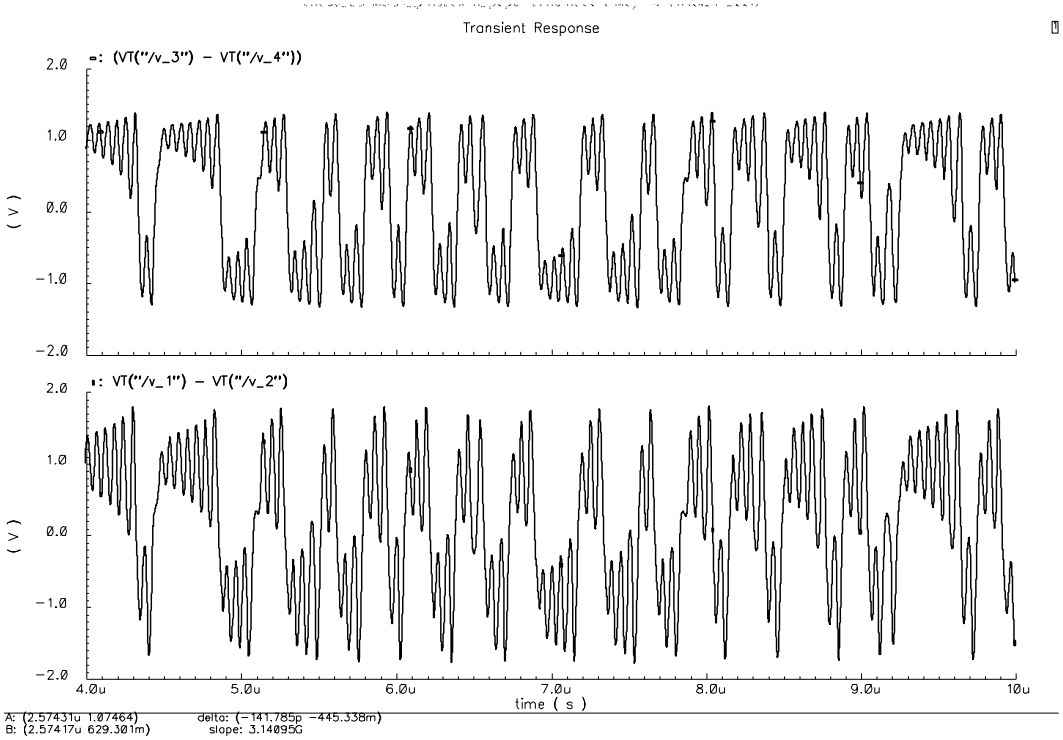

Figure 4.25: Waveforms of (V_A-V_B) and (V_C-V_D) differences

4.3.3 Difference Amplifier

As explained in chapter 1.2, V_A-V_B difference is needed to generate random bits. Since the oscillation has high frequency components, opamp-based closed-loop topologies are not suitable as the subtraction circuit. Because of their simplicity and high frequency operation capability, open-loop architectures such as differential pairs can be good candidates.

Although the oscillation amplitude in V_A and V_B is approximately 600mVpp (meaning that 500mVpp linear region is needed), the output of the differential pair saturates when the input voltage difference reaches $4V_T$=100mV. Resistive degeneration was applied to widen the linear region of the differential pair. In Figure 4.26, the schematic view of the difference amplifier is shown.

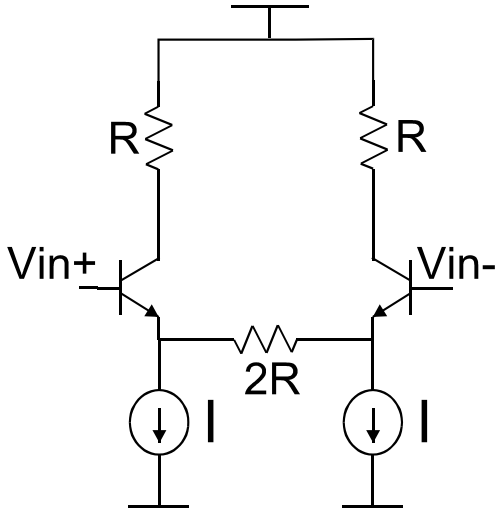

Figure 4.26: Subtraction circuit

The V_{in}-V_{out} characteristic of the circuit in Figure 4.26 for different degeneration resistance values are shown in Figure 4.27.

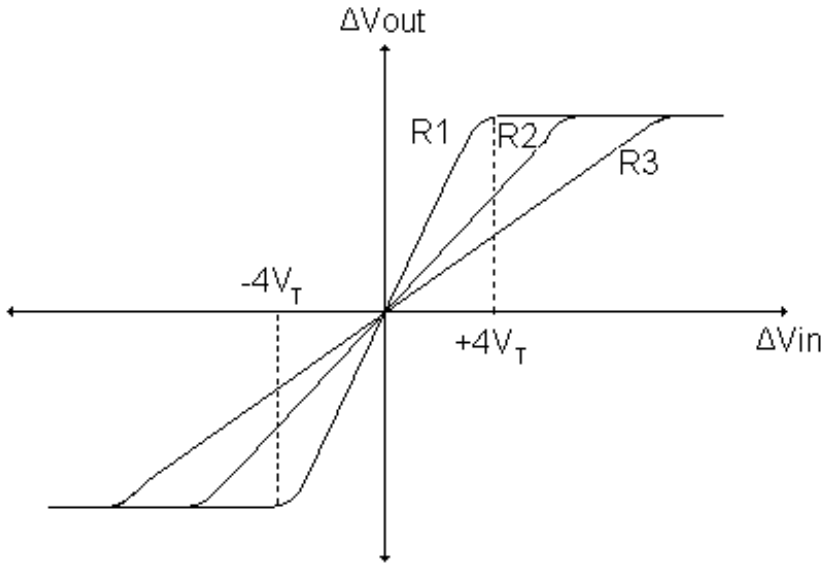

Figure 4.27: DC characteristic of a differential pair

In Figure 4.27, R_1=0, and $R_3 > R_2 > R_1$. Bigger the degeneration resistance, results the bigger linear region. R=600Ω was chosen to maintain the desired linear region. The simulation results of the subtraction circuit are shown in Figure 4.28.

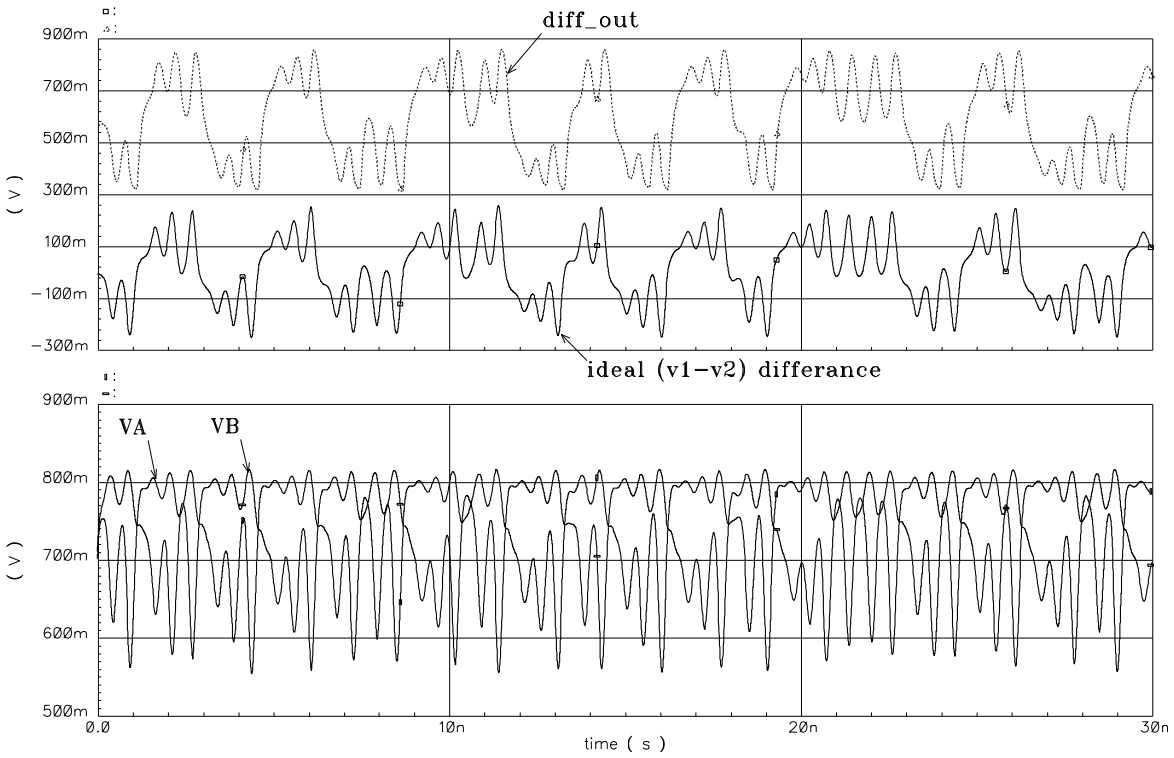

Figure 4.28: Simulation results of subtraction circuit

In Figure 4.28, the output of difference amplifier is shown with the voltage waveforms of node V_A, V_B and the ideal V_A-V_B difference which derived using Spectre calculator function. From Figure 4.28 it is seen that the differences of V_A and V_B voltages are taken properly with the help of applied linearization.

4.3.4 Comparator Circuit

Again a differential pair was used as a comparator. Unlike the subtraction circuit, a wide linear region is not needed in the comparator, so resistive degeneration was not applied. In the signal flow, when the signals reached the comparator, there is no defined clock signal yet. Because of the absence of the clock signal, well known “latch” structures with positive feedback could not be used. Instead, the desired gain was achieved by using three differential stages in cascade. The final schematic of comparator circuit is shown in Figure 4.29. The simulation results of comp0 and comp1 are shown in Figure 4.30.

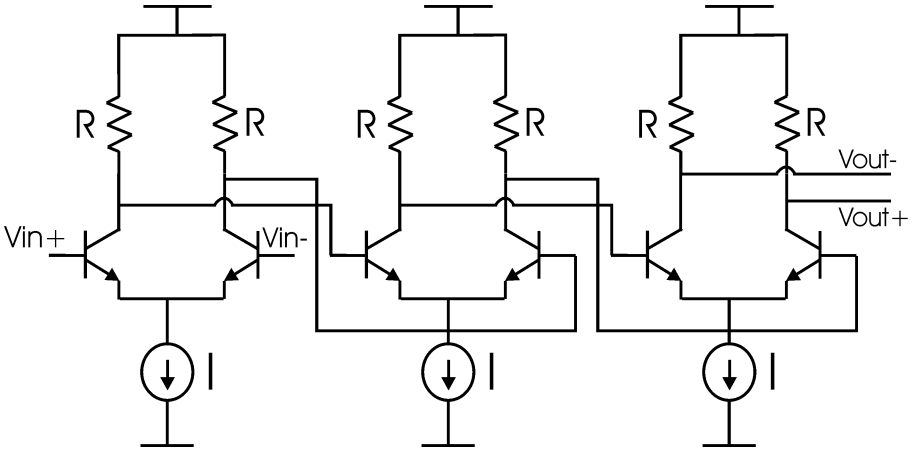

Figure 4.29: Comparator circuit

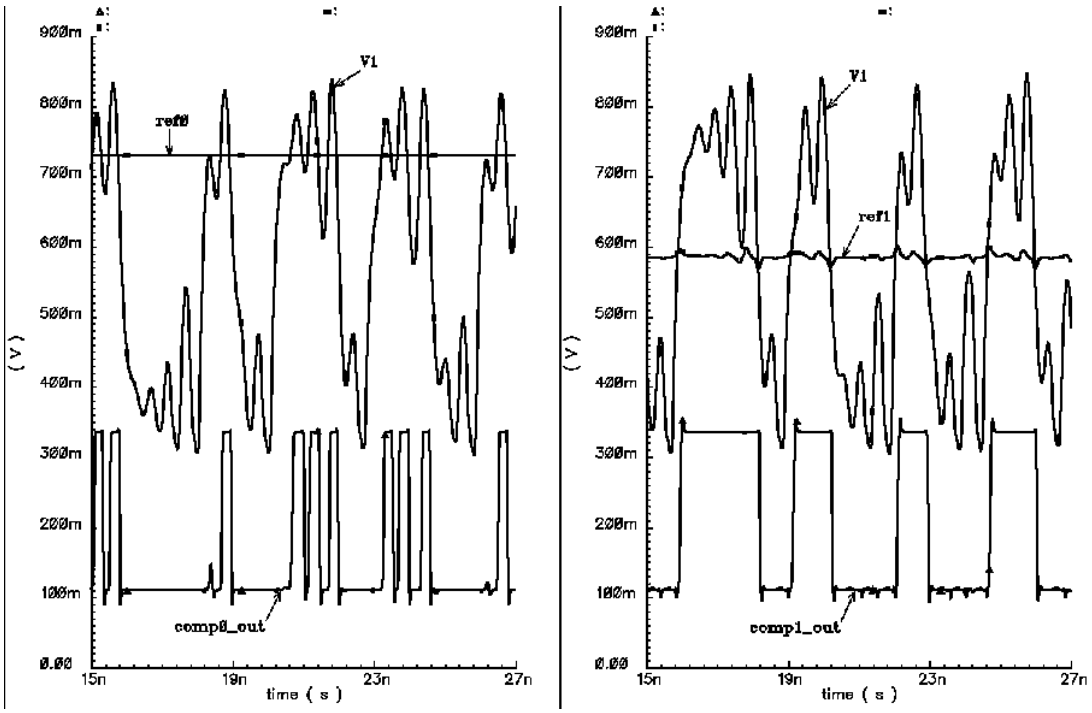

Figure 4.30: Simulation results of comp0 and comp1

As seen in the Figure 4.30, the outputs of comparator circuits go to logic one, if the inputs exceed reference levels, and otherwise comparators produce logic zero.

4.3.5 "OR" Circuit

OR operation was realized by using a standard ECL logic gate. Since the output of OR circuit is a clock signal, and it drives three flip-flops, relatively stronger emitter followers were placed at the outputs of the OR gate. The schematic of the OR circuit is shown in Figure 4.31 (The emitter followers are not included).

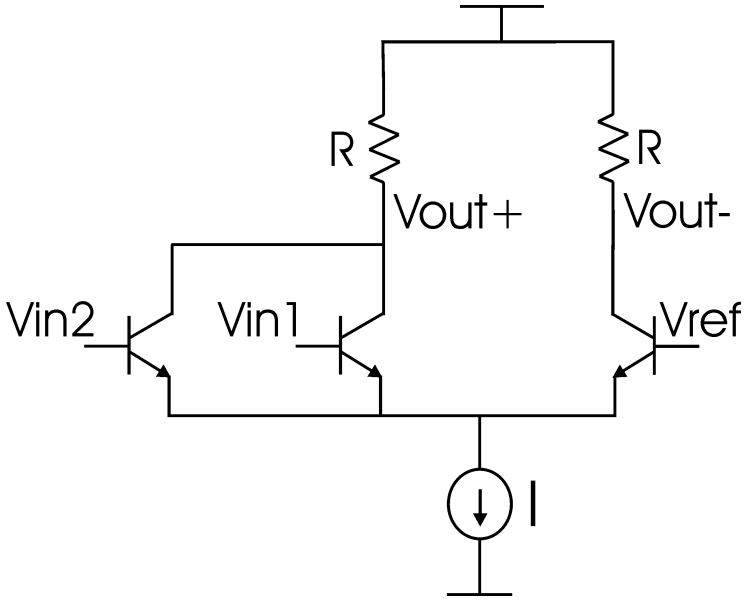

Figure 4.31: OR circuit

The operation of OR circuit is same as the comparator. If any one of Vin1 and Vin2 exceeds Vref, the output of the circuit is inverted. To maintain the proper operation, Vref must be placed at the middle of the swing of Vin1,Vin2 signals. Since Vin1 and Vin2 signals are the outputs of comparator circuit, Vref can be obtained with the same structure, and element values that were used in the comparator as shown in Figure 4.32.

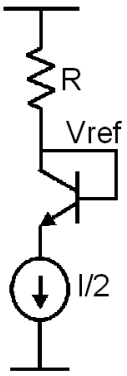

Figure 4.32: Getting V_{ref}

The simulation results of the OR circuit is shown in Figure 4.33. In Figure 4.33, last two waveforms are the inputs of OR circuit, while the first one is the output of OR circuit or “clk” signal. Whenever any of the comp0 or comp1 outputs go to logic high, the output of the OR circuit also goes to high and the clk signal is constructed.

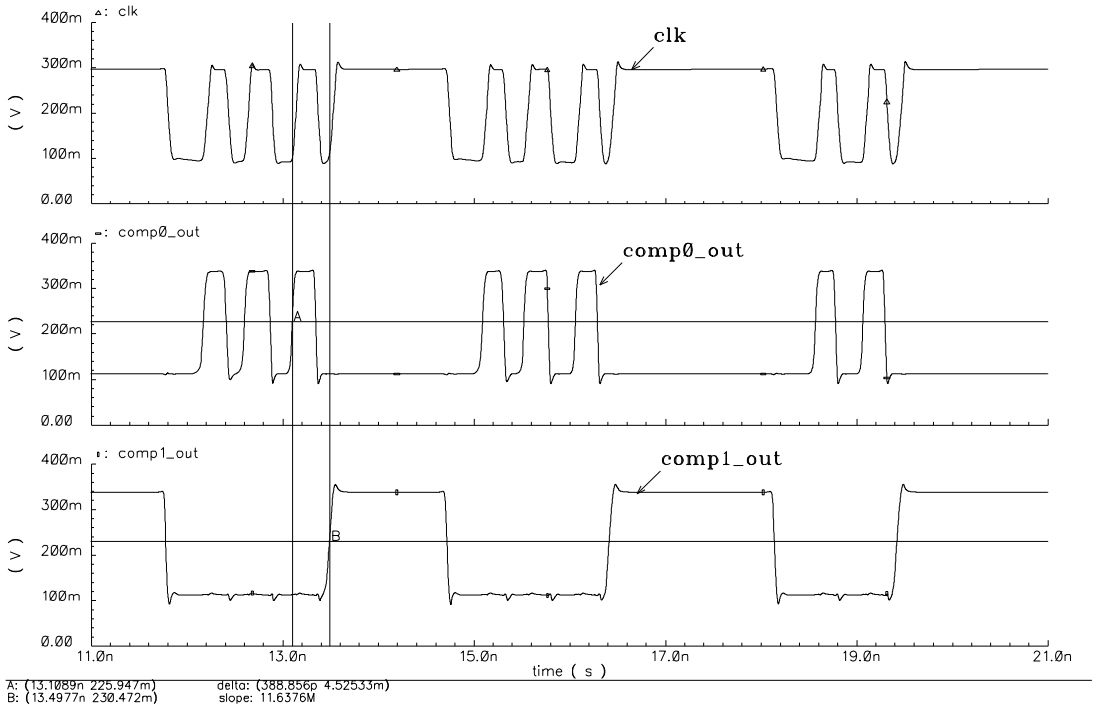

Figure 4.33: Simulation results of OR circuit

4.3.6 DFF

DFF was realized by using two CML latches that receive inverted clocks. The circuit schematic and simulation results of DFF are shown in Figure 4.34, and Figure 4.35 respectively.

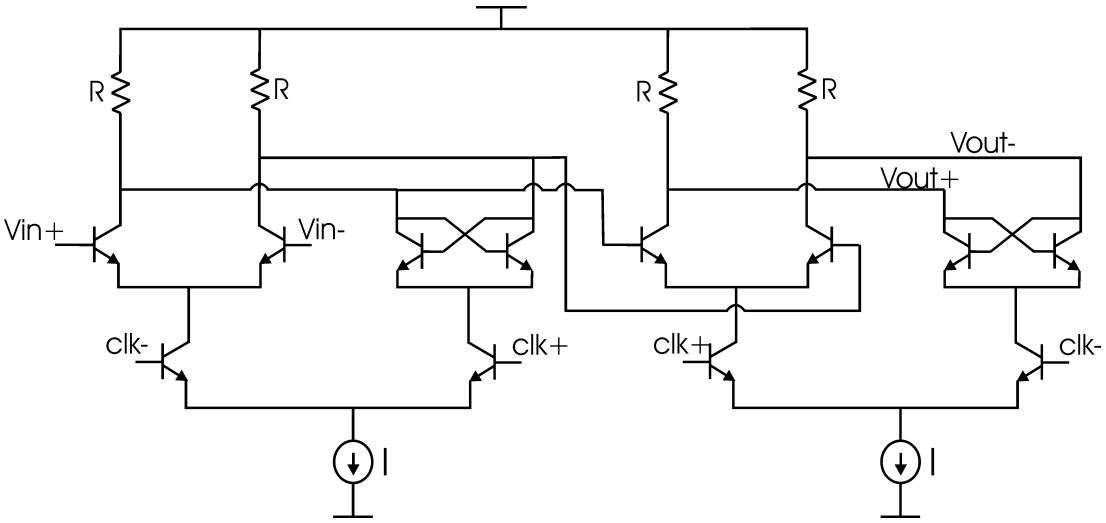

Figure 4.34: DFF circuit

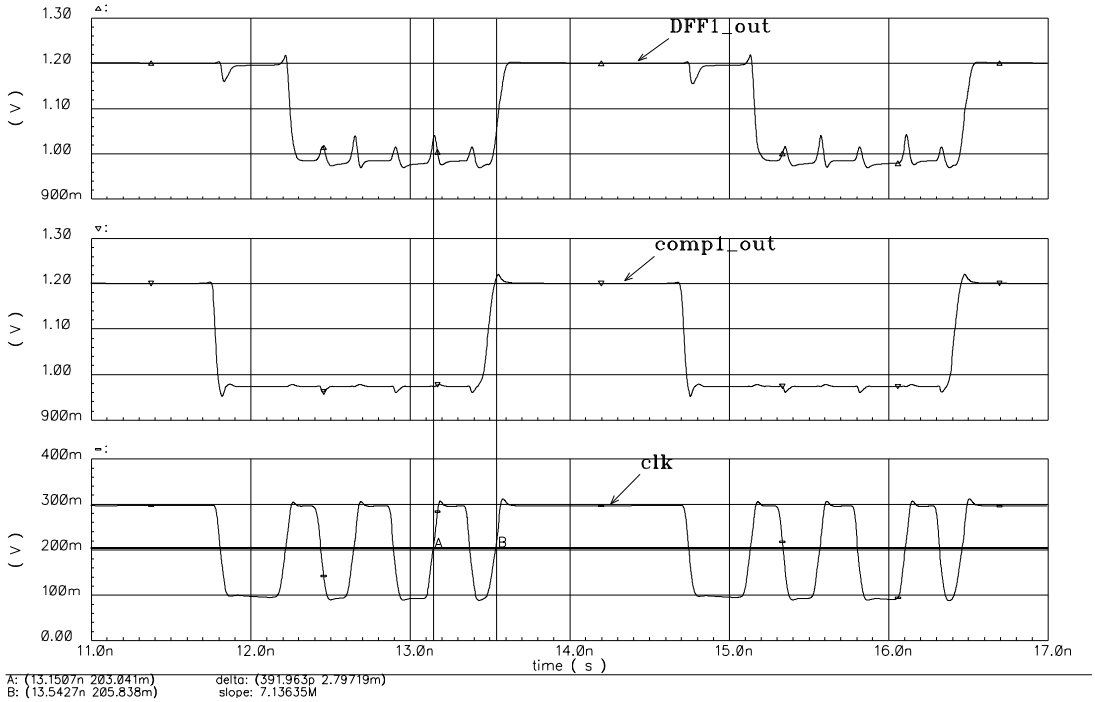

Figure 4.35: Simulation results of DFF

From Figure 4.35 it is seen that DFF circuit works correctly. At the rising edge of clock signal it transfers the input data to the output.

The glitches that occur at the output signal in clock transitions are not such a high level that degrades the circuit functionality.

4.3.7 EXOR Gate

As explained in Section 4.2, in Von-Neumann de-skewing algorithm so as to keep 01, 10 sequences and discard 00,11 sequences EXOR gate was used. The well-known CML structure was used as the EXOR gate. In Figure 4.36, schematic of EXOR gate, and in Figure 4.37 simulation results are shown.

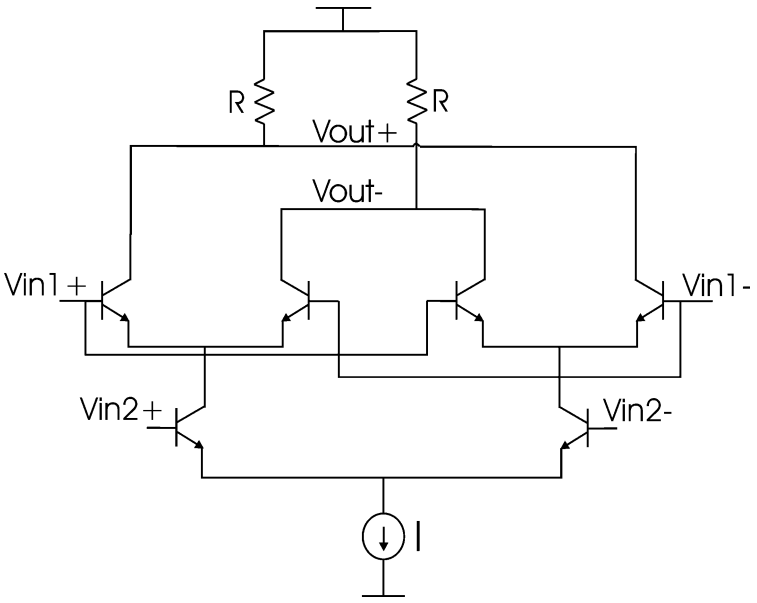

Figure 4.36: EXOR gate

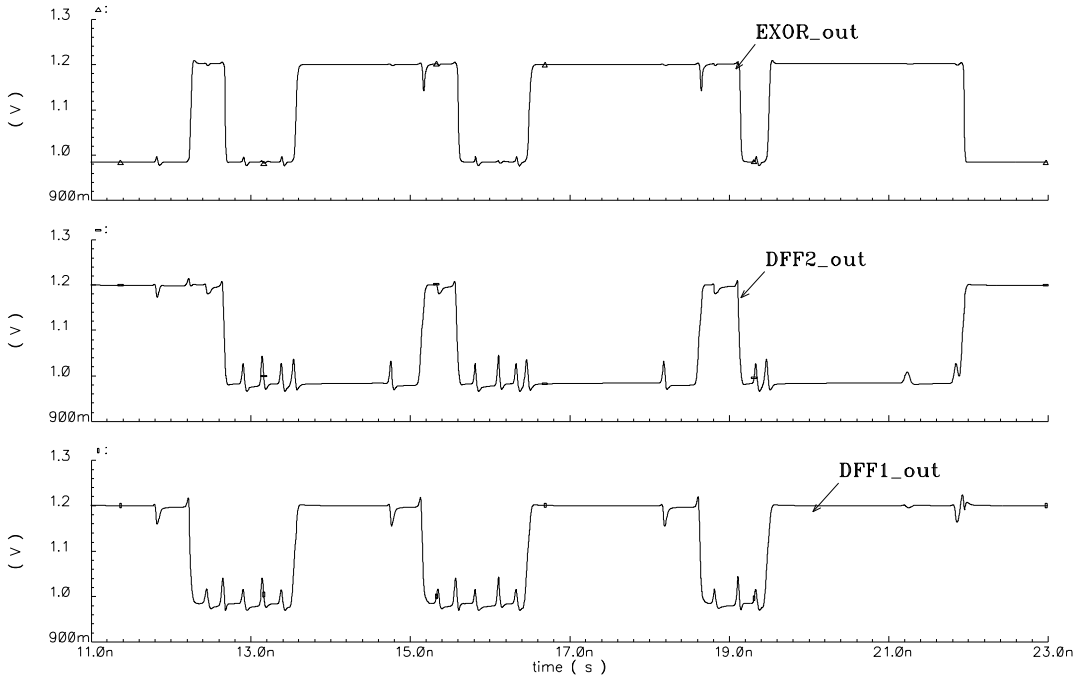

Figure 4.37: Simulation results of EXOR

From Figure 4.37 it is seen that fhe EXOR gate, produce logic“1”, if the inputs are complement of each other. Otherwise it produces logic”0”.

4.3.8 Divide-by-two Circuit

Divide-by-two circuit was realized using designed DFF, and shorting $\overline{Q}$ output and input of flip-flop. Simulation result is given in Figure 4.38.

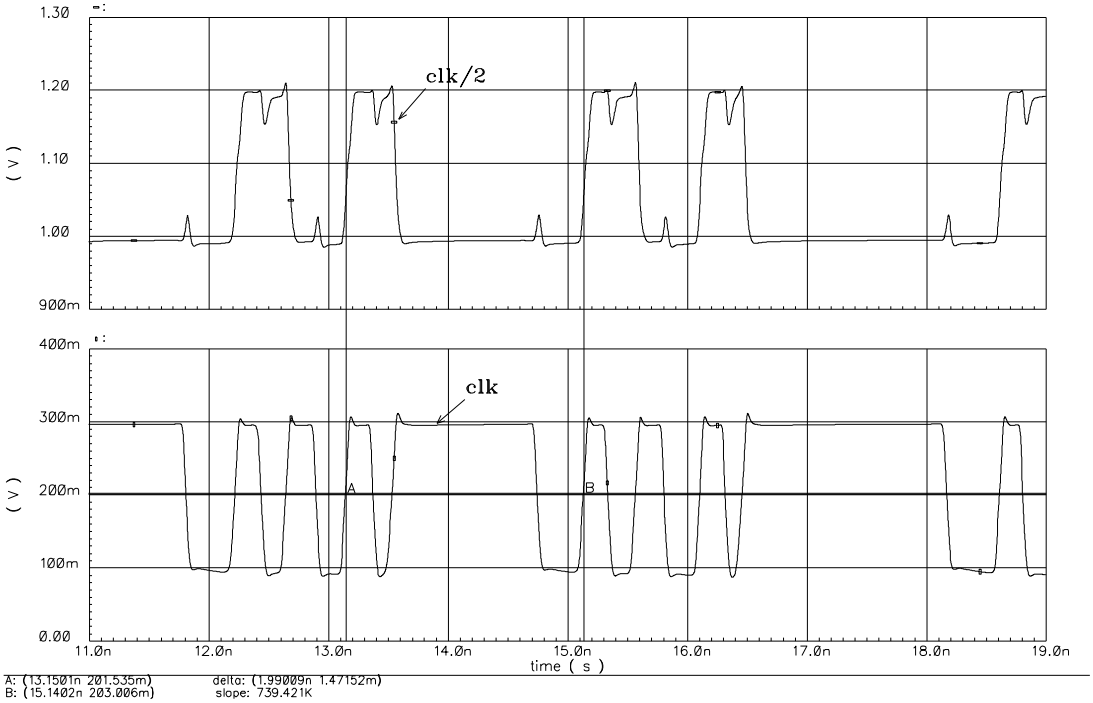

Figure 4.38: Construction of signal "clk/2"

4.3.9 Output Drivers

CML drivers with on-chip 50Ω termination resistors are designed and used as output derivers. In order to maintain a desired swing, current I was selected 8mA. In order to represent the loads that drivers will face in reality, the equivalent structure shown in Figure 4.39 was constructed and simulations were carried out with this structure. In FigureXXX, C_{pad}, $L_{bondwire}$,$R_{bodnwire}$,C_{pin} are pad capacitance, bondwire inductance, parasitic bondwire resistance, and pin capacitance respectively. The values were taken as C_{pad}=100f, $L_{bondwire}$=5nH, $R_{bodnwire}$=5Ω and C_{pin}=5pF. Simulation results of CML drivers are shown in Figure 4.40

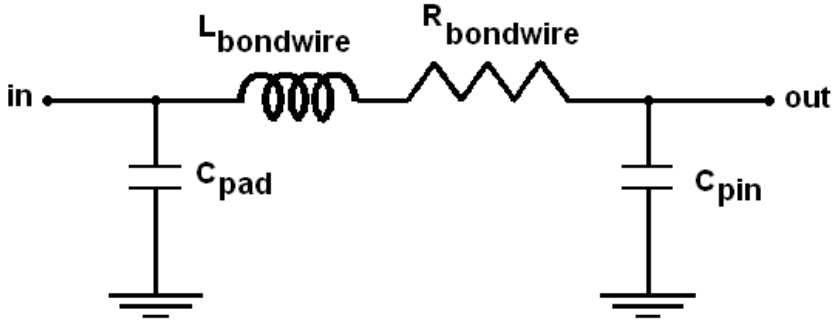

Figure 4.39: Load structure was used in CML simulations

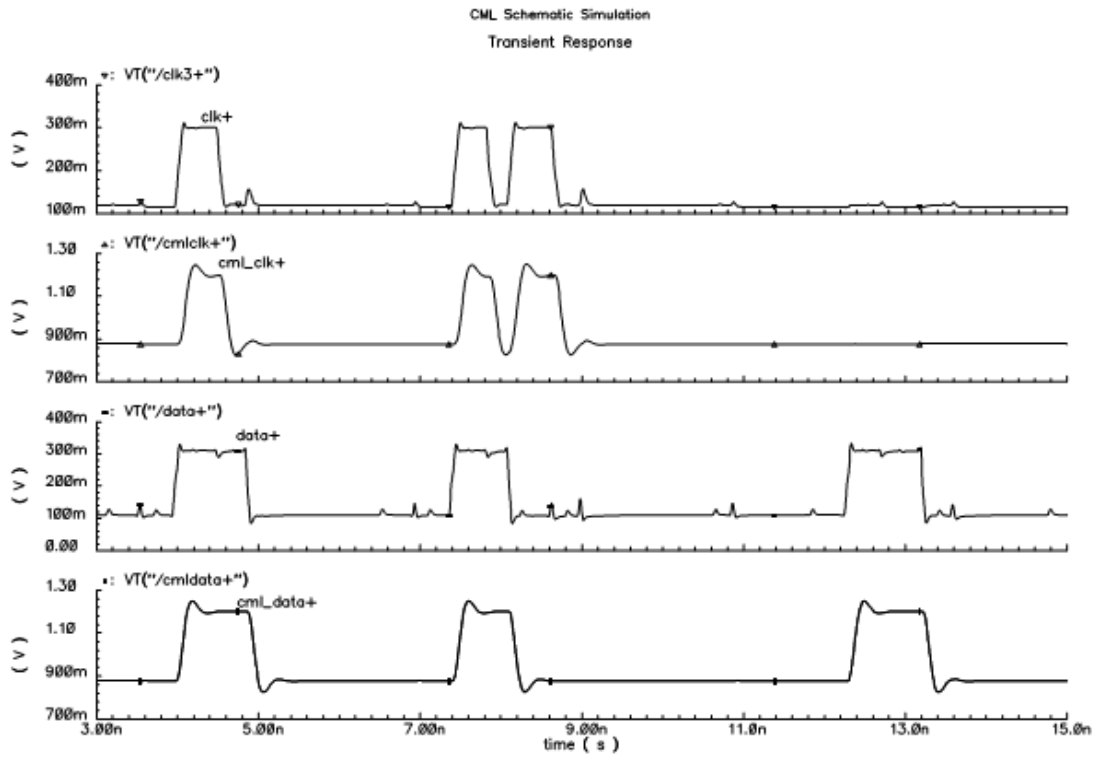

Figure 4.40: Simulation results of CML drivers

Since the used load model is relatively high compared to the loads of internal circuits, rise and fall times of data and clock signals are decreased at the output of CML buffers as expected. However, this decreasing is not effect the operation.

4.3.10 Toplevel Integration

In toplevel integration the supply lines of chaotic oscillator and consecutive RBG blocks were separated to prevent cross-correlation as far as possible. Coupling capacitances were placed between V_{CC} and V_{EE} lines to get better AC coupling between the supply lines. RF probe pads were placed at the output of the subtraction circuit to avail monitoring the oscillation directly. Total silicon area is 1,09mmx0.57mm. Power dissipation is approximately 50mW in the core circuits and 150mW at CML IO pad drivers. The final layout of the chip is shown in Figure 4.42.

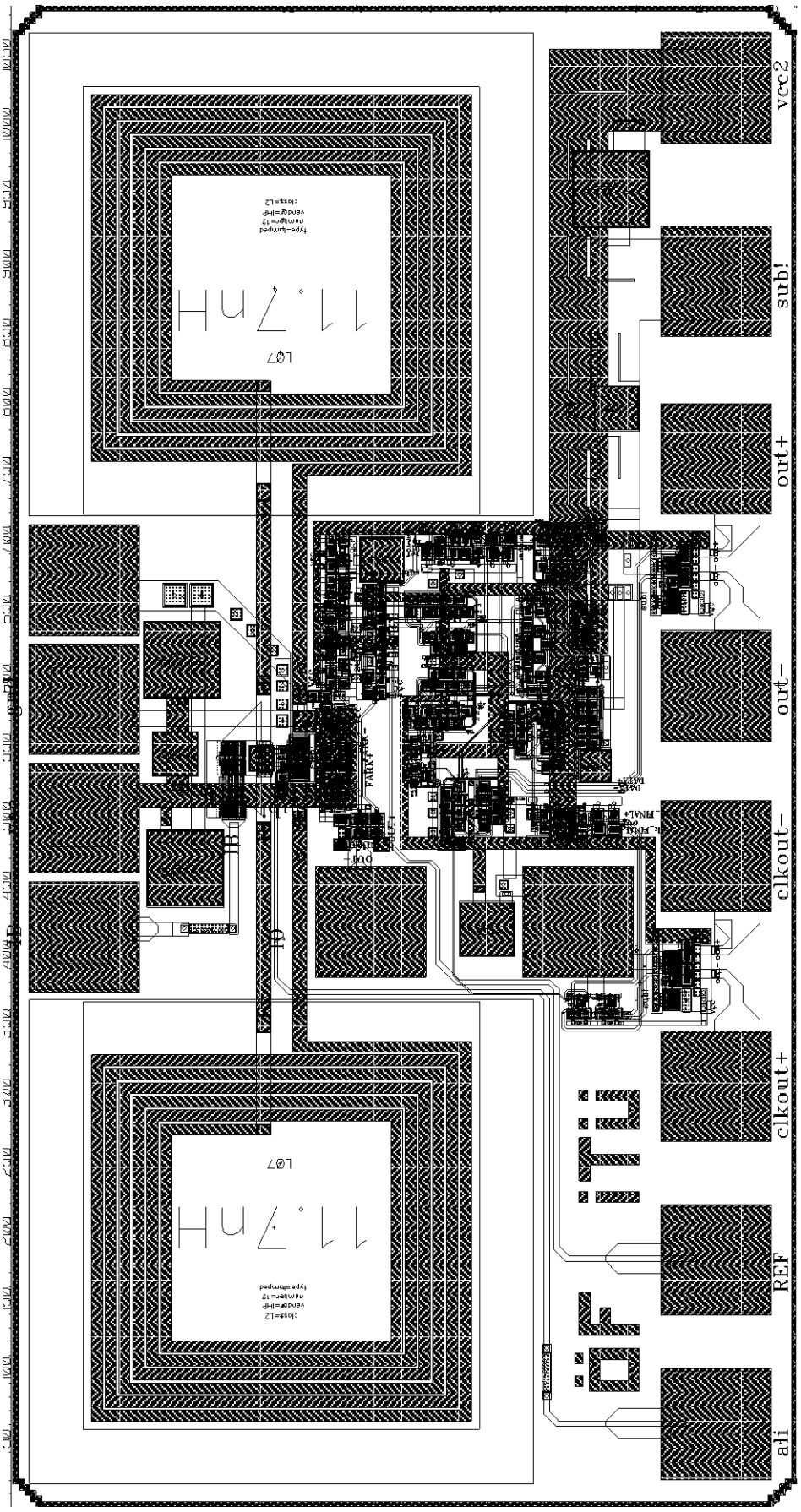

Figure 4.41: Toplevel layout of the system

5. CONCLUSION

The ultimate goal of this research was to integrate a negative-gm LC chaotic oscillator, and use this oscillator outputs to generate a successive bit stream that passes random tests.

The design is carried out using IHP 0.25 µm SiGeC BiCMOS technology, which has three different types of bipolar transistors (high speed, high voltage and standard), high-Q MIM (Metal Insulator Metal) capacitor, salicided and unsalicided polysilicon resistors, 4 level Al interconnect metal and 2µm thick fourth level metal to realize high-Q inductors.

Achieved throughput of random bit generation system is approximately 300Mbit/s. There are a few RNGs that exhibit comparable bit rates to our design. Also there is no reported chaotic oscillator in the literature operating at frequencies similar to our design.

From Spectre simulations, it is observed that when component values deviate from real values because of process variations by changing I_B current the circuit can be derived to chaotic oscillation. In order to guarantee the chaotic oscillation, I_B and I_O were taken out of the chip so that they would be driven from outside.

A test PCB will be designed and measurements will be done on receiving the chip samples from production.

In order to examine statistical properties of the generated bit stream, 150.000 bits were taken from Spectre simulations and each 20.000 bits were subjected to FIPS-140-1 tests. The stream was failed in poker test. Although chaotic oscillator generates very complex behavior, it is a deterministic system indeed. Without noise injection it may be hard to pass statistical tests. It is thought that, after fabrication of test chip, generated bit streams will pass all tests.

The BJT version of the chaotic oscillator and the random bit generation block was experimentally tested by using discrete components. In order to use standard discrete components, the passive element values and DC biasing currents were changed accordingly. The generated bit streams were read via the parallel port of a personal computer. A bit stream with a length of 1,500,000 was obtained. It was divided into 75 groups each having a length of 20,000 bits and each block was subjected to the FIPS-1-140 test suit. The first 27 blocks passed the statistical tests.

However, 5 of the following 48 blocks failed to tests. It is thought that this is due to the cross-correlations resulting from the improper setup. When the bit stream was subjected to the tests by skipping one of each consecutive bits, all sequences passed the statistical tests.

From Spectre simulations, it is observed that bipolar version of chaotic-LC oscillator can oscillate in frequencies up to 5GHz, by choosing another parameter set and using smaller inductance values. In this way, the throughput of the circuit can be also increased.

It is also assumed that another good RNG could be designed by sampling the output of the chaotic oscillator by a slower oscillator.

REFERENCES

[1] **Jun, B. and Kocher P.,** 1999, The Intel® Random Number Generator, *Cryptography Research, Inc. white paper prepared for Intel Corporation*.

[2] **Cryptography Research Inc.,** 2003, Evaluation of VIA C3 Nehemiah Random Number Generator, Cryptography Research Inc.

[3] **Menezes, A.J., Van Oorschot, P.C.**, **and Vanstone S.A.,** 1997. Handbook of Applied Cryptography. CRC Press, Florida.

[4] **Stefanou, N., Sonkusale S.R.,** 2004. High Speed Array of Oscillator-based Truly Binary Random Number Generators, *IEEE ISCAS '04,* pp. 505 -508 May 2004.

[5] **Kennedy, M.P.,** 1993. Three Steps to Chaos - Part I: Evolution, *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40, 640-656.

[6] **Hein, S.,** 1993. Exploiting chaos to suppress spurious tones in general double-loop $\Sigma\Delta$ modulators, *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 651–659, Oct. 1993.

[7] **Jewett, R., Poulton, K., Hsieh, K.-C. and Doernberg, J.,** 1997. A 12b 128MSample/s ADC with 0.05LSB DNL, *ISSCC Digest of Technical Papers,* San Francisco, CA, Feb. 1997, pp. 138–139.

[8] **Walker, W. T.,** Radar system utilizing chaotic coding, *U.S. Patent 5 321 409,* June 1994.

[9] **Kolumban, G., Kennedy, M. P. and Chua, L. O.,** The role of synchronization in digital communications using chaos—Part II: Chaotic modulation and chaotic synchronization, *IEEE Trans. Circuits Syst. I,* vol. 45, pp. 1129–1140, Nov. 1998.

[10] **Delgado-Restituto, M., Rodriguez-Vazquez, A.,** 2002. Integrated Chaos Generators, *Proceedings of the IEEE,* 90, 747-767.

[11] **Chua, L.O., Wu, C.W.**, **Huang A. and Zong G.,** 1993. A Universal Circuit for Studying and Generating Chaos – Part I : Routes to Chaos, *IEEE*

[12] **Ozoguz, S., Ates, O., Elwakil, A.S.,** 2005. An Integrated Circuit Chaotic Oscillator and Its Application for High Speed Random Bit Generation, *IEEE Proceedings of the 2005 International Symposium on Circuits and Systems* , Kobe, Japan, May 23-26.

[13] **Yalcin, M.E. , Suykens, J.A.K. and Vandewalle J.,** 2004. True Random Bit Generaton from a Double Scroll Attractor, *IEEE Transactions on Circuits and Systems I : Fundemantal Theory and Applications*, 51, 1395-1404.

BIOGRAPHY

Fidel Bayam was born in Bergama, TURKEY in 1981. He graduated from Bergama High School in 1997. In 2002, he received B.Sc. degree in Electronics and Communication Engineering from Istanbul Technical University where he started to continue M.Sc. degree in the same year. He has been working as a design engineer at ETA ASIC Design Center since 2002. His research interests are high frequency IC design and high speed data converters.