

ANALYSIS OF AN EFFICIENT LINEAR FEEDBACK SHIFT REGISTER FOR VLSI APPLICATIONS

Submitted in partial fulfillment of the requirements for the award of Bachelor of
Engineering Degree in Electronics and Communication Engineering

by

A.SALMANFAARIS (39130391)

SAIDU GOWTHAM KUMAR (39130388)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

SATHYABAMA

**INSTITUTE OF SCIENCE AND
TECHNOLOGY (DEEMED TO BE
UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119**

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of a **SALMAN FAARIS A** (39130391) And **SAIDU GOWTHAM KUMAR** (39130388) who carried out the project entitled "**ANALYSIS OF AN EFFICIENT LINEAR FEEDBACK SHIFT REGISTER FOR VLSI APPLICATIONS**" under my supervision from December 2022 to April 2023.

Internal Guide

Mr.M.A.MUTHIAH.,M.Tech

Head of the Department

Dr. T. RAVI M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

I **Salman Faaris A** (39130391) And **Saidu Gowtham Kumar** (39130388) Hereby Declare That The Project Report Entitled “**Analysis Of An Efficient Linear Feedback Shift Register For Vlsi Applications**” Done By Us Under The Guidance Of **Mr.M.A.Muthiah.,M.Tech.,(Ph.D).**, Dept. Of Electronics And Communication Engineering, Sathyabama Institute Of Science And Technology, Chennai, Is Submitted In Partial Fulfillment Of The Requirements Of The Award Of Bachelor Of Engineering Degree In **Electronics And Communication Engineering**.

DATE:

PLACE: CHENNAI

SIGNATURE OF CANDIDATES

1.

2.

ACKNOWLEDGEMENT

I am pleased to acknowledge our sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

I convey my thanks to **Dr. N.M.NANDHITHA M.E.,Ph.D., Dean**, School of Electricals and Electronics Engineering and **Dr. T. RAVI M.E.,Ph.D.**, Head of the Department of Electronics and Communication Engineering for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to our Project Guide **Mr.M.A.MUTHIAH.,M.TECH.,(Ph.D.)**, Dept. of Electronics and Communication Engineering, for his valuable guidance, suggestions and constant encouragement for the successful completion of our project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	vii
	LIST OF FIGURES	viii
	LIST OF TABLES	ix
1	INTRODUCTION	1
	1.1 Introduction	1
	1.2 Motivation	2
	1.3 Shift Register	3
	1.3.1. SISO	4
	1.3.2 SIPO	5
	1.3.3 PISO	6
	1.3.4 USES	6
	1.4 Linear Feedback Shift Register	7
	1.4.1 LFSR Implementation	7
	1.4.2 One to many Versus Many to One	9
	1.5 Adders	10

	1.6 Half Adder	11
	1.7 Full Adder	14
	1.8 Half Adder And Full Adder Using K-Map	16
	1.8.1 Logical Expression of Sum , Carry	17
	1.9 Organization Of The Report	19
2	LITERATURE SURVEY	21
	2.1 Literature Survey	21
	2.2 Literature Summary	28
	2.3 Objective Of the Project	28
3	Procedures Of Existing System	29
	3.1 Low Power FlipFlop	29
	3.2 Low Power LFSR	31
4	PROPOSED METHODOLOGY	34
	4.1 Proposed Flipflop	34
	4.2 HSPICE SOFTWARE	35
	4.2.1 Simulated Screen	37
	4.2.2 Cosmoscope	38
	4.2.3 Power	38
	4.2.4 Delay	39

	4.2.5 Power Delay Product	39
5	RESULTS AND DISCUSSIONS	40
	5.1 Results	40
	5.2 Simulation Results Of Partial Product	41
	5.3 Simulation Result Of PPS	41
	5.4 Simulation Result Of Full Adder	42
	5.5 Simulation Of D-Flip flop	42
	5.6 Performance Analysis	43
	5.6.1 Performance analysis Of Half Adder	43
	5.6.2 Performance Analysis Of Partial Product	44
	5.6.3 Performance Analysis Of Program Stored	45
	5.6.4 Performance Analysis Of Pps	46
	5.6.5 Performance Analysis Of Full Adder Gate	47
	5.6.6 Performance analysis Of Low Power FlipFlop	49
6	CONCLUSION	50
	REFERENCES	52

ABSTRACT

VLSI plays an important role in circuit designing, it reduces the size of the electronic circuits which will be helpful in designing the compact electronic devices. This project focuses on the linear feedback shift register, that must be efficient in data storage and should be reduced in size. The tool going to be used for the implementation of the efficient linear feedback shift register is Xilinx Virtex-4 FPGA. The VLSI tool will evaluate its performance in terms of the logic, speed and the memory requirement in FPGA. The linear feedback shift register performs discrete step operations that shifts the bits one position to the left and replaces the vacated bit by the exclusive-or of the bits shifted off and the bit previously at a given tap position in the register. Linear Feedback Shift Registers (LFSRs) are widely used in various VLSI applications such as cryptography, error correction coding, and digital signal processing. In this paper, we present an analysis of an efficient LFSR design for VLSI applications. The proposed LFSR design is based on a Galois LFSR structure, which offers better performance in terms of speed and area efficiency compared to conventional Fibonacci LFSRs. We begin by reviewing the basic concepts of LFSRs and their application in VLSI design. We then present a detailed analysis of the proposed LFSR design, including its Galois LFSR structure, feedback polynomial, and implementation details. We also discuss the benefits of using the proposed LFSR design in VLSI applications, such as its low power consumption and high clock frequency.

LIST OF FIGURES

Figure No	Figure Name	Page No
1.1	Shift register	4
1.2	SISO Shift Register	5
1.3	SIPO Shift Register	6
1.4	PISO Shift Register	6
1.5	LFSR With XOR Feedback	9
1.6	Comparison Of Alternative Tap Selections	9
1.9	Half Adder	13
1.10	2-bit half Adder Truth table	14
1.12	Full Adder	16
1.13	Truth Table Of Full Adder	16
3.1	CMOS DFF	29
4.1	Block Diagram Of Flip flop	34
4.2	HSPUI Interface	36
4.2.1	AvanWaves Interface	37
4.2.2	CosmoScope	38
5.1	Simulation Results Of Half Adder	40
5.2	Simulation result Of Partial Product	41
5.3	Simulation Result Of PPS	41
5.4	Simulation Results Of Full Adder	42
5.5	Simulation Of D-Flipflop	42

LIST OF TABLES

TABLE NO	Table Name	Page No
5.1	Design Analysis Of 130NM Half Adder	43
5.2	Design Analysis Of 32NM Half Adder	43
5.3	Design Of 130NM Partial Product	44
5.4	Design Of 32NM Partial Product	44
5.5	Design Analysis Of 130NM Program Stored	45
5.6	Design Analysis Of 32NM Program Stored	45
5.7	Design Analysis Of 130NM PPS	46
5.8	Design Analysis Of 32NM PPS	46
5.9	Design Analysis Of 130NM Full Adder Gate	47
5.10	Design Analysis Of 32NM Full Adder	47
5.11	Design Analysis Of 130NM Low Power Flipflop	48

CHAPTER 1

INTRODUCTION

1.1 Introduction

Very large-scale integration was made possible with the wide adoption of the MOS transistor, originally invented by Mohamed M. Atalla and Dawon Kahng at Bell Labs in 1959. Atalla first proposed the concept of the MOS integrated circuit chip in 1960, followed by Kahng in 1961, both noting that the MOS transistor's ease of fabrication made it useful for integrated circuits. General Microelectronics introduced the first commercial MOS integrated circuit in 1964. In the early 1970s, MOS integrated circuit technology allowed the integration of more than 10,000 transistors in a single chip. This paved the way for VLSI in the 1970s and 1980s, with tens of thousands of MOS transistors on a single chip (later hundreds of thousands, then millions, and now billions).

The first semiconductor chips held two transistors each. Subsequent advances added more transistors, and as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now known retrospectively as small-scale integration (SSI), improvements in technique led to devices with hundreds of logic gates known as medium-scale integration (MSI). Further improvements led to large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and billions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large-scale integration above VLSI. Terms like ultra-large-scale integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot. Terms suggesting greater than VLSI levels of integration are no longer in widespread use.

In 2008, billion-transistor processors became commercially available. This became more commonplace as semiconductor fabrication advanced from the then-current generation of 65 nm processes. Current designs, unlike the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high-performance logic blocks like the SRAM (static random-access memory) cell, are still designed by hand to ensure the highest efficiency.

1.2 Motivation

Multipliers play a very important role in many digital systems such as calculators, microprocessors, digital filters, and digital signal processor. They are also used in implementation of discrete Fourier transform, Correlation, Range measurement. The performance of any digital system is usually measured in terms of the used multipliers, since the process of the multiplication is very slow. Multiplication is nothing but an add and shift algorithm which is considered as series of repeated addition. That is the multiplicand adds itself the multiplier number of times. To implement this multiplication, it requires a large number of hardware elements and also it will operate at lower speed. Shift and Add, Reducing the Number of Partial Products, Radix-4 Booth Recoding and Sign Extension Tricks. The design of an efficient High speed Radix-4 Booth multiplier for both signed and unsigned numbers.

1.3 Shift Register

A shift register is a type of digital circuit using a cascade of flip-flops where the output of one flip-flop is connected to the input of the next. They share a single clock signal, which causes the data stored in the system to shift from one location to the next. By connecting the last flip-flop back to the first, the data can cycle within the shifters for extended periods, and in this configuration they were used as computer memory, displacing delay-line memory systems in the late 1960s and early 1970s.

In most cases, several parallel shift registers would be used to build a larger memory pool known as a "bit array". Data was stored into the array and read back out in parallel, often as a computer word, while each bit was stored serially in the shift registers. There is an inherent trade-off in the design of bit arrays; putting more flip-flops in a row allows a single shifter to store more bits, but requires more clock cycles to push the data through all of the shifters before the data can be read back out again.

It is to be noted here that data can be transferred in or out of the register either serially or parallelly. So, the data bit movement inside the shift register give rise to various configurations which are as follows:

- **SISO**: Serial-in Serial-out: It permits the insertion of data serially and taking the output also in a serial manner.
- **SIPO**: Serial-in Parallel-out: Here the data is inserted serially either from the left or right direction. But the output is taken parallelly.
- **PISO**: Parallel-in Serial-out: This type of shift register allows the parallel input of data bit, but the output is taken serially.
- **PIPO**: Parallel-in Parallel-out: PIPO shift register permits both in and out of data bit in a parallel manner.

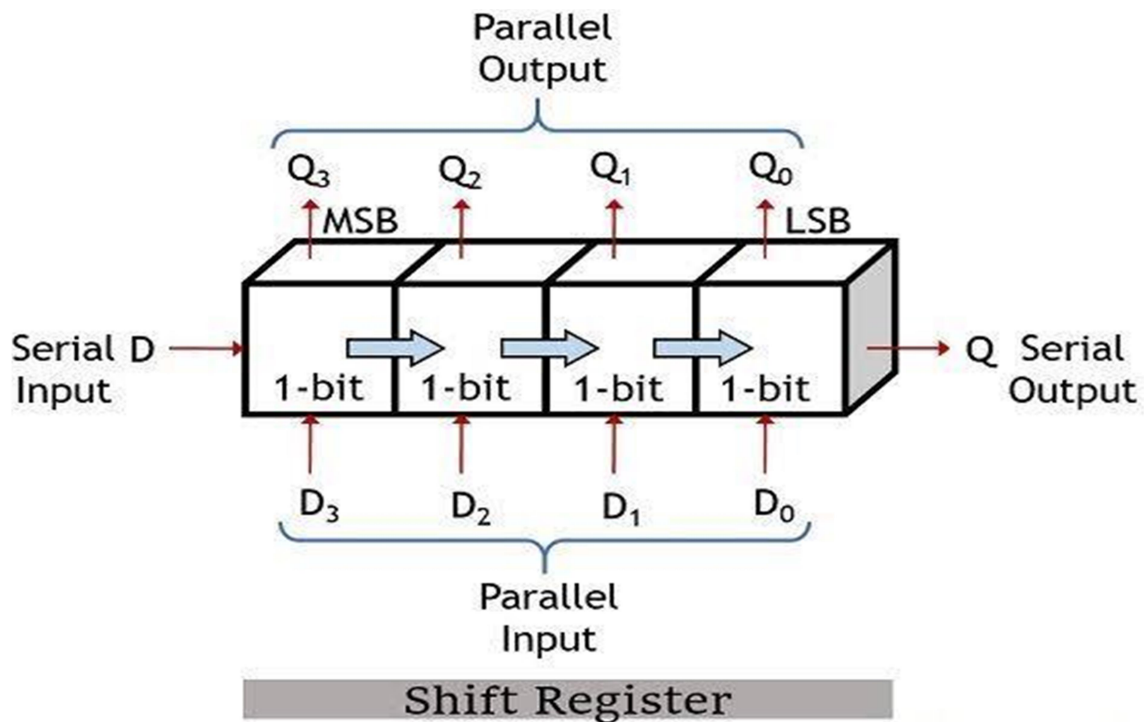


Fig 1.1 Shift Register

1.3.1 Serial In Serial Out (SISO)

These are the simplest kind of shift registers. The data string is presented at "data in" and is shifted right one stage each time "data advance" is brought high. At each advance, the bit on the far left (i.e. "data in") is shifted into the first flip-flop's output. The bit on the far right (i.e. "data out") is shifted out and lost.

The data is stored after each flip-flop on the "Q" output, so there are four storage "slots" available in this arrangement, hence it is a 4-bit register. To give an idea of the shifting pattern, imagine that the register holds 0000 (so all storage slots are empty). As "data in" presents 1,0,1,1,0,0,0,0 (in that order, with a pulse at "data advance" each time—this is called clocking or strobing) to the register, this is the result. The right hand column corresponds to the right-most flip-flop's output pin, and so on.

So the serial output of the entire register is 00010110. It can be seen that if data were to be continued to input, it would get exactly what was put in (10110000), but offset by four "data advance" cycles. This arrangement is the hardware equivalent of a queue. Also, at any time, the whole register can be set to zero by bringing the reset (R) pins high.

This arrangement performs *destructive readout* – each datum is lost once it has been shifted out of the right-most bit.

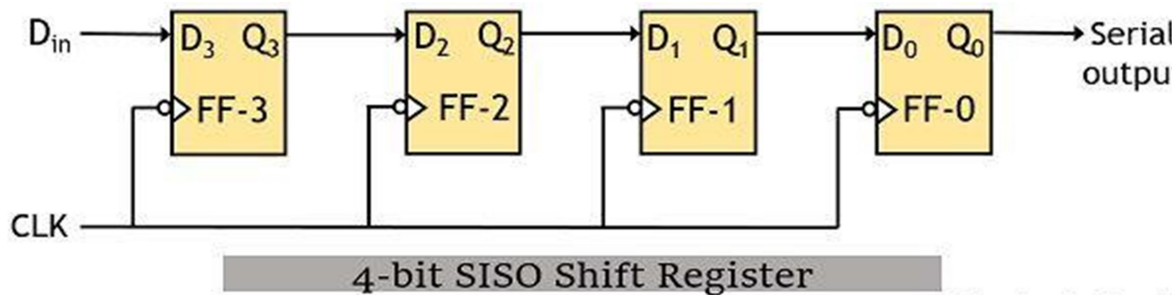


Fig 1.2 SISO Shift Register

1.3.2 Serial In Parallel Out (Sipo)

This configuration allows conversion from serial to parallel format. Data input is serial, as described in the SISO section above. Once the data has been clocked in, it may be either read off at each output simultaneously, or it can be shifted out.

In this configuration, each flip-flop is edge triggered. All flip-flops operate at the given clock frequency. Each input bit makes its way down to the Nth output after N clock cycles, leading to parallel output.

In cases where the parallel outputs should not change during the serial loading process, it's desirable to use a latched or buffered output. In a latched shift register (such as the 74595) the serial data is first loaded into an internal buffer register, then upon receipt of a load signal the state of the buffer register is copied into a set of output registers. In general, the practical application of the serial-in/parallel-out shift register is to convert data from serial format on a single wire to parallel format on multiple wires.

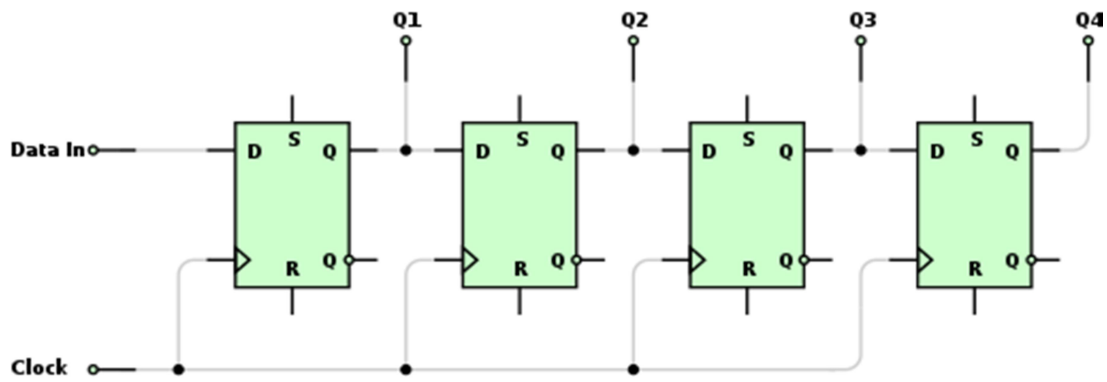


Fig 1.3 SIPO Shift Register

1.3.3 Parallel In Serial Out

This configuration has the data input on lines D1 through D4 in parallel format, D1 being the most significant bit. To write the data to the register, the Write/Shift control line must be held LOW. To shift the data, the W/S control line is brought HIGH and the registers are clocked. The arrangement now acts as a PISO shift register, with D1 as the Data Input. However, as long as the number of clock cycles is not more than the length of the data-string, the Data Output, Q, will be the parallel data read off in order.

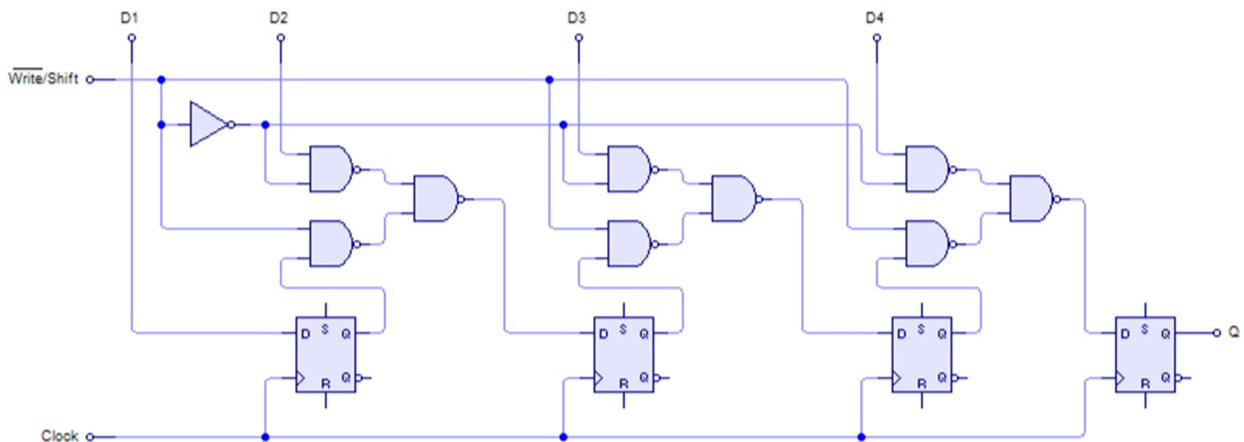


Fig 1.4 PISO Shift Register

1.3.4 SHIFT REGISTER USES

One of the most common uses of a shift register is to convert between serial and parallel interfaces. This is useful as many circuits work on groups of bits in parallel, but serial interfaces are simpler to construct. Shift registers can be used as simple delay circuits. Several bidirectional shift registers can also be connected in parallel for a hardware implementation of a stack.

SIPO registers are commonly attached to the output of microprocessors when more general-purpose input/output pins are required than are available. This allows several binary devices to be controlled using only two or three pins, but more slowly than by parallel output. The devices in question are attached to the parallel outputs of the shift register, and the desired state for all those devices can be sent out of the microprocessor using a single serial connection. Similarly, PISO configurations are commonly used to add more binary inputs to a microprocessor than are available – each binary input (such as a button or more complicated circuitry) is attached to a parallel input of the shift register, then the data is sent back via serial to the microprocessor using several fewer lines than originally required.

Many computer languages include instructions to "shift right" and "shift left" the data in a register, effectively dividing by two or multiplying by two for each place shifted.

1.4 Linear Feedback Shift Register(LFSR)

A type of shift register called a linear feedback shift register (lfsr) uses a linear feedback function to shift its contents. lfsrs consist of a series of flip flops, with the output of each flip flop connected to the input of the next flip flop. the flip flops shift the contents of the register to the right (or left) every clock cycle, and the feedback function calculates the new value for the first flip flop based on a linear combination of the current values of the other flip-flops .

The feedback function is typically implemented using exclusive-or (xor) gates that combine specific flip-flop outputs in a specific pattern. the inputs to the xor gates are usually selected based on a polynomial representation of the feedback function.

The polynomial representation of the feedback function is derived from the desired length of the LFSR sequence and the characteristic polynomial of the LFSR.

The characteristic polynomial of an LFSR is defined as the polynomial whose roots correspond to the cycle lengths of the LFSR sequence. the feedback function is designed such that the LFSR sequence generated by the shift register repeats after a certain number of clock cycles, which is determined by the characteristic polynomial of the lfsr. the efficiency of an LFSR design can be evaluated based on Several Factors, Including Power Consumption, Area Overhead, Speed, And Complexity. The Power Consumption Of An Lfsr Is Dependent On Factors Such As Clock Frequency, Supply Voltage, And The Number Of Flip-Flops Used In The design. The area overhead is the additional circuit area required to implement the LFSR, which includes the area required for the flip-flops, feedback function, and other control logic.

The speed of an LFSR is determined by the clock frequency, the propagation delay of the flip-flops, and the delay of the feedback function. The complexity of an LFSR design is determined by the number of flip-flops, the feedback function, and any additional control logic required to implement the design. In the paper "Analysis of an Efficient Linear Feedback Shift Register for VLSI Applications," the authors propose a novel LFSR design that reduces power consumption and area overhead while maintaining high speed and low complexity. The proposed design utilizes a modified feedback function that reduces the number of XOR gates required to implement the feedback function. The authors analyse the proposed design using simulations and comparisons with existing LFSR designs, demonstrating its superiority in terms of performance and efficiency. The results show that the proposed LFSR architecture is capable of achieving high throughput with reduced power consumption and area overhead, making it an ideal choice for VLSI applications.

1.4.1 LSFR IMPLEMENTATION

LFSRs are simple to construct and are useful for a wide variety of applications, but are often sadly neglected by designers. One of the more common forms of LFSR is formed from a simple shift register with feedback from two or more points, or *taps*, in the register chain.

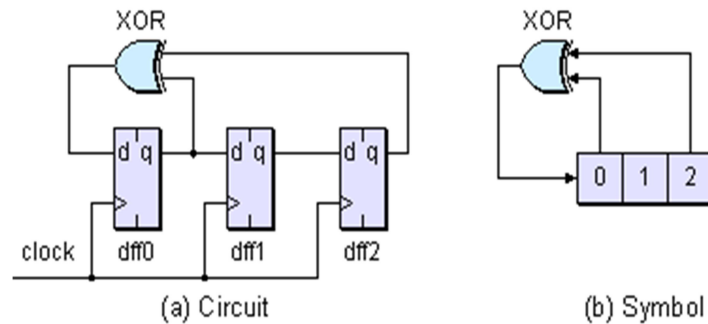


Fig 1.5 LFSR with XOR feedback

The taps in this example are at bit 0 and bit 2, and can be referenced as [0,2]. All of the register elements share a common clock input, which is omitted from the symbol for reasons of clarity. The data input to the LFSR is generated by XOR-ing or XNOR-ing the tap bits; the remaining bits function as a standard shift register. The sequence of values generated by an LFSR is determined by its feedback function (XOR to XNOR) and tap selection. For example, consider two 3-bit XOR based LFSRs with different tap selections.

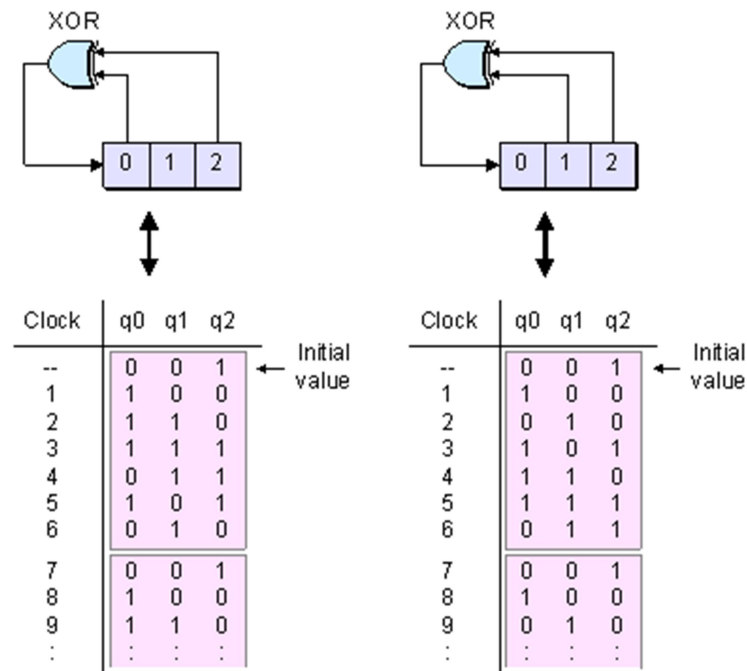


Fig 1.6 Comparison of alternative tap selections

Both LFSRs start with the same initial value but, due to the different taps, their sequences rapidly diverge as clock pulses are applied. In some cases an LFSR will end up cycling round a loop comprising a limited number of values. However, both of the LFSRs shown in *Fig 2* are said to be of *maximal length* because they sequence through every possible value (excluding all of the bits being 0) before returning to their initial values. A binary field with 'n' bits can assume 2^n unique values, but a maximal-length LFSR with 'n' register bits will only sequence through $(2^n - 1)$ values. This is because LFSRs with XOR feedback paths will not sequence through the value where all the bits are 0, while their XNOR equivalents will not sequence through the value where all the bits are 1.

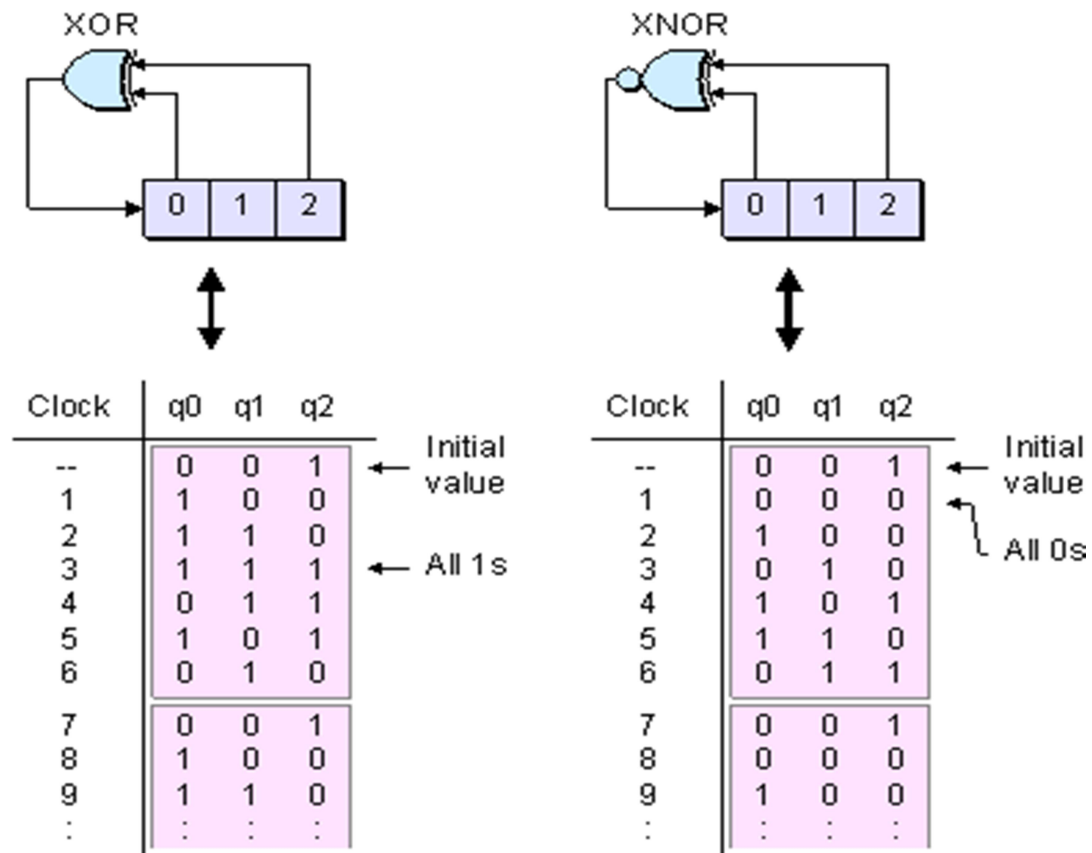


Fig 1.7 Comparison of XOR to XNOR feedback paths.

1.4.2 One-To-Many Versus Many-To-One Implementations

Consider the case of an 8-bit LFSR, for which the minimum number of taps that will generate a maximal-length sequence is four. In the real world, XOR gates only have two inputs, so a four-input XOR function has to be created using three XOR gates arranged as two levels of logic. Even in those cases where an LFSR does support a minimum of two taps, you may actually wish to use a greater number of taps such as eight (which would result in three levels of XOR logic).

The problem is that increasing the levels of logic in the combinational feedback path can negatively impact the maximum clocking frequency of the function. One solution is to transpose the many-to-one implementations discussed above into their one-to-many counterparts.

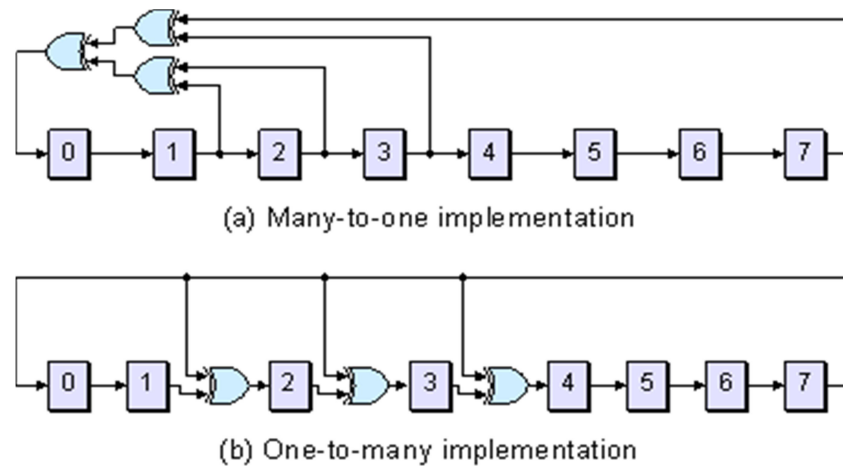


Fig 1.8 One-to-many versus many-to-one implementations.

The traditional many-to-one implementation for the eight-bit LFSR has taps at [7,3,2,1]. To convert this into its one-to-many counterpart, the most-significant tap (which is always the most significant bit) is fed back directly into the least significant bit, and is also individually XORed with the other original taps (bits [3,2,1] in this example). Note that although both styles result in maximal-length LFSRs, the actual sequences of values will differ between them. But the main point is that using the one-to-many style means that there is never more than one level of combinational logic in the feedback path, irrespective of the number of taps being employed.

1.5 Adders

An adder or summer is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic units, but also in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators, and similar operations.

Although adders can be constructed for many numerical representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers, it is trivial to modify an adder into an adder subtractor. Other signed number representations require more logic around the basic adder.

Binary carry-propagate adders have been extensively published, heavily attacking problems related to carry chain problem. Binary adders evolve from linear adders, which have a delay approximately proportional to the width of the adder, e.g. ripple-carry adder (RCA), to logarithmic - delay adders, such as the carry-look ahead adder (CLA). There are some additional performance enhancing schemes, including the carry- increment adder and the Ling adder that can further enhance the carry chain, however, in Very Large Scale Integration (VLSI) digital systems, the most efficient way of offering binary addition involves utilizing parallel-prefix trees, this occurs because they have the regular structures that exhibit logarithmic delay. Parallel-prefix adders compute addition in two steps: one to obtain the carry at each bit, with the next to compute the sum bit based on the carry bit. Unfortunately, prefix trees are algorithmically slower than fast logarithmic adders, such as the carry propagate adders, however, their regular structures promote excellent results when compared to traditional CLA adders.

This happens within VLSI architectures because a carry-look ahead adder, such as the one implemented in one of Motorola's processors, tends to implement the carry chain in the vertical direction instead of a horizontal one, which has a tendency to increase both wire density and fan-in/out dependence. Therefore, although logarithmic adder structures are one of the fastest adders algorithmically, the speed efficiency of the carry-look ahead adder has been hampered by diminishing returns given the fan-in and fan-out dependencies as well as the heavy wire load distribution in the vertical path. In fact, a traditional carry-look ahead adder implemented in VLSI can actually be slower than traditional linear-based adders, such as the Manchester carry adder.

1.6 Half Adder

So, coming to the scenario of half adder, it adds two binary digits where the input bits are termed as augend and addend and the result will be two outputs one is the sum and the other is carry. To perform the sum operation, XOR is applied to both the inputs, and AND gate is applied to both inputs to produce carry.

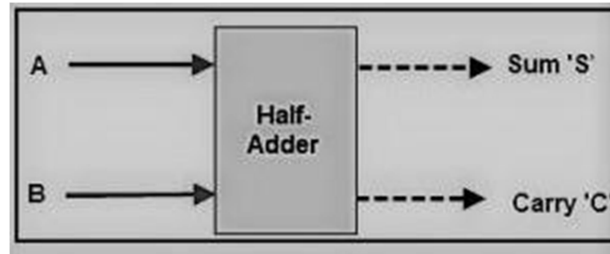


Fig 1.9 Half Adder

Whereas in the full adder circuit, it adds 3 one-bit numbers, where two of the three bits can be referred to as operands and the other is termed as bit carried in. The produced output is 2-bit output and these can be referred to as output carry and sum. By using a half adder, you can design simple addition with the help of logic gates. Let's see an example of adding two single bits.

The 2-Bit Half Adder Truth Table is as Below :

INPUTS		OUTPUTS	
A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Fig 1.10 2-bit half adder truth table

These are the least possible single-bit combinations. But the result for $1 + 1$ is 10, the sum result must be re-written as a 2-bit output. Thus, the equations can be written as

The output '1' of '10' is carry-out. 'SUM' is the normal output and 'CARRY' is the carry-out. Now it has been cleared that a 1-bit adder can be easily implemented with the help of the XOR Gate for the output 'SUM' and an AND Gate for the 'Carry'. For instance, when we need to add two 8-bit bytes together, then it can be implemented by using a full-adder logic circuit. The half-adder is useful when you want to add one binary digit quantity.

A way to develop two-binary digit adders would be to make a truth table and reduce it. When you want to make a three binary digit adder, the half adder addition operation is performed twice. In a similar way, when you decide to make a four-digit adder, the operation is performed one more time. With this theory, it was clear that the implementation is simple, but development is a time taking process.

The simplest expression uses the exclusive OR function:

$$\text{Sum} = A \text{ XOR } B$$

$$\text{Carry} = A \text{ AND } B$$

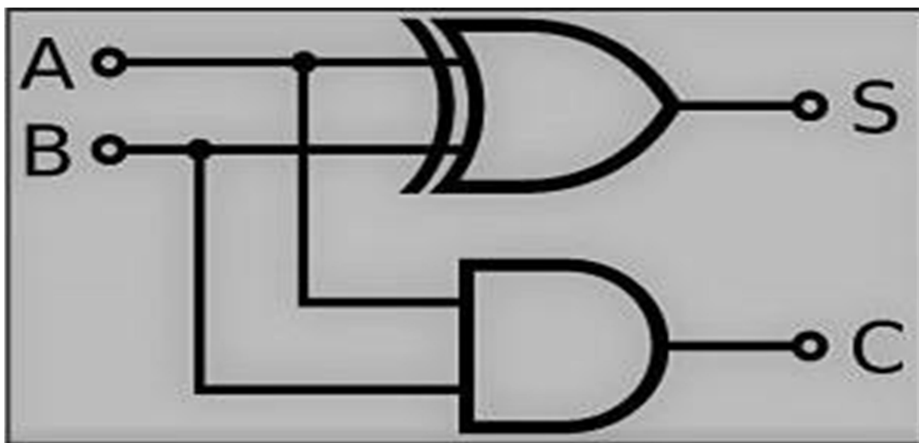


Fig 1.11 Logic circuit for half adder

And an equivalent expression in terms of the basic AND, OR, and NOT is:

$$\text{SUM} = A.B + A.B'$$

The implementation of half adder can be done through high-speed CMOS digital logic integrated circuits like 74HC series which includes the SN74HC08 (7408) & SN74HC86 (7486).

The main reason to call these binary adders like Half Adders is, that there is no range to include the carry bit using an earlier bit. So, this is a main limitation of HAS once used like binary adder particularly in real-time situations which involve adding several bits. So this limitation can be overcome by using the full adders.

1.7 Full Adder

This adder is difficult to implement when compared to half-adder.

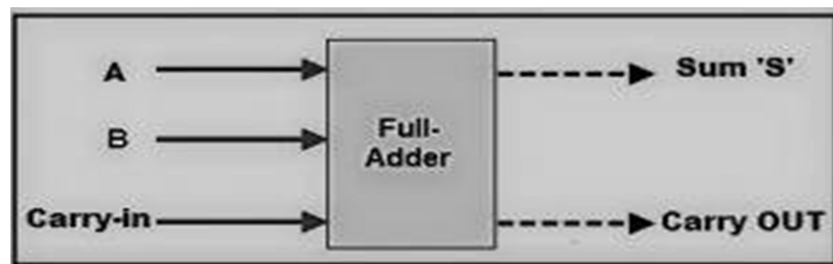


Fig 1.12 Full Adder

The full-adder has three inputs and two outputs, whereas half adder has only two inputs and two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.

INPUTS			OUTPUT	
A	B	C-IN	C-OUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Fig 1.13 Truth table of full adder

The difference between a half-adder and a full-adder is that The output carry is designated as C-OUT and the normal output is represented as S which is 'SUM'. With the above full adder truth-table, the implementation of a full adder circuit can be understood easily. The SUM 'S' is produced in two steps: By XOR ing the provided inputs 'A' and 'B'. The result of A XOR B is then XOR ed with the C-IN. If any of the half adder logic produces a carry, there will be an output carry. So, C-OUT will be an OR function of the half-adder Carry outputs. Take a look at the implementation of the full adder circuit shown below.

This generates SUM and C-OUT is true only when either two of three inputs are HIGH, then the C-OUT will be HIGH. So, we can implement a full adder circuit with the help of two half adder circuits. Initially, the half adder will be used to add A and B to produce a partial Sum and a second-half adder logic can be used to add C-IN to the Sum produced by the first half adder to get the final S output.

If any of the half adder logic produces a carry, there will be an output carry. So, C-OUT will be an OR function of the half-adder Carry outputs. Take a look at the implementation of the full adder circuit shown below. The implementation of larger logic diagrams is possible with the above full adder logic a simpler symbol is mostly used to represent the operation. Given below is a simpler schematic representation of a one-bit full adder

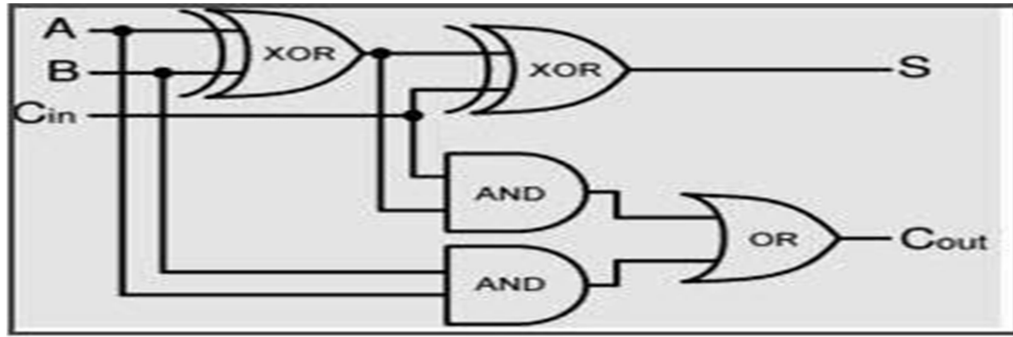


Fig 1.13 Logic diagram of full adder

With this type of symbol, we can add two bits together, taking a carry from the next lower order of magnitude, and sending a carry to the next higher order of magnitude. In a computer, for a multi-bit operation, each bit must be represented by a full adder and must be added simultaneously. Thus, to add two 8-bit numbers, you will need 8 full adders which can be formed by cascading two of the 4-bit blocks.

1.8 Half Adder And Full Adder Using K-Map

Even The Sum And Carry Outputs For Half Adder Can Also Be Obtained With The Method Of Karnaugh Map (K-Map). The Half Adder And Full Adder Boolean Expression Can Be Obtained Through K-Map. So, The K-Map For These Adders Is

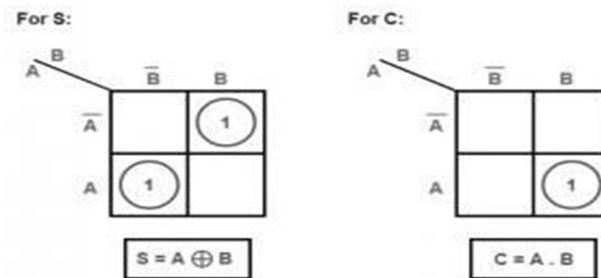


Fig 1.14 Half Adder Of K-Map

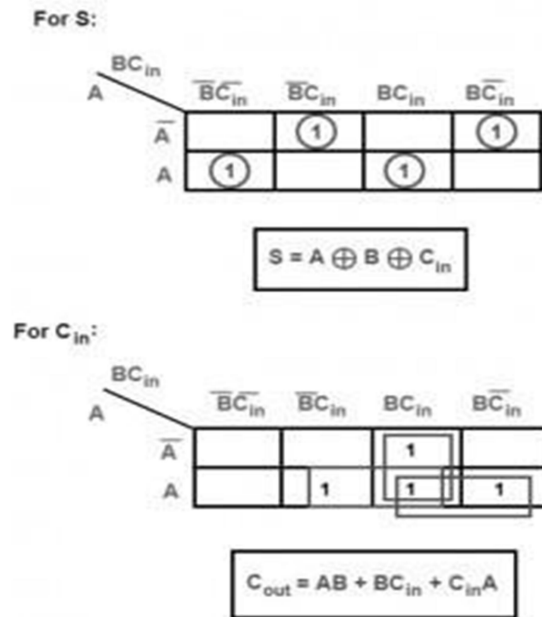


Fig 1.15 Full adder of k-map

1.8.1 Logical Expression of SUM and Carry

The logical expression of sum (S) can be determined based on the inputs mentioned in the table.

$$= A' B' C_{in} + A' B C C_{in}' + A B' C_{in}' + AB C_{in}$$

$$= C_{in}(A' B' + AB) + C_{in}' (A' B + A B')$$

$$= C_{in} \text{ EX - OR}(A \text{ EX - OR } B)$$

$$= (1,2,4,7)$$

The logical expression of the carry (C_{out}) can be determined based on the inputs mentioned in the table.

$$= A' B C_{in} + A B' C_{in} + AB C_{in}' + AB C_{in}$$

$$= AB + BC_{in} + AC_{in} = (3, 5, 6, 7)$$

With the above-mentioned truth tables, the results can be obtained and the procedure is: A combinational circuit combines the different gates in the circuit where those can be an encoder, decoder, multiplexer and de multiplexer. The characteristics of combinational circuits are as follows.

The output at any instant of time is based only on the levels that are present at input Terminals. It does not use any memory. The previous state of input does not have any effect on the present state of the circuit. It can have any number of inputs and m number of outputs.

CHAPTER 2

LITERATURE SURVEY

This chapter discusses the literature survey and summary. It also give idea about analysis of an efficient radix 4 booth multiplier for low power applications.

2.1 LITERATURE SURVEY

Manohar Ayinala and Keshab K. Parhi, “EFFICIENT PARALLEL VLSI ARCHITECTURE FOR LINEAR FEEDBACK SHIFT REGISTERS” Linear feedback shift register (LFSR) is an important part of the cyclic redundancy check (CRC) operations and BCH en-coders. This paper presents a novel high speed parallel LFSR architecture based on parallel Infinite Impulse Response (IIR) filter design, pipelining and retiming algorithms. A new for-mulation is proposed to modify the LFSR into the form of an IIR filter. Then pipelining and retiming algorithms are applied to further reduce the critical path in the parallel architecture. A comparison between the proposed and previous architectures shows that our parallel architecture achieves a critical path same as that of previous designs with a reduced hardware cost.

Raj Kumar Mistri and Rahul Ranjan, “IC Layout Design of 4-bit Universal Shift Register using Electric VLSI Design System” It is considered that improved other performance specifications such as less delay, high noise immunity and suitable ambient temperature conditions are the prime factors. In this paper two different techniques are used for designing a 4-bit Universal shift register (USR) and then a comparison is made about area and average delay. First one is the Transmission Gate (TG) technique and the second one is the GDI Technique. This paper describes the design of an Integrated Circuit (IC) layout for a 4-bit USR. The layout was designed by use of an open source software namely Electric VLSI Design System which is an Electronic Design Automation (EDA) tool. LTspiceXVII is used as a simulator to carry out the simulation work.

Srinivasan Krishnaswamy and Harish K. Pillai, "On the Number of Linear Feedback ShiftRegisters With a Special Structure"Given a primitive polynomial of degree n , we deal with the problem of finding the number of possible linear feedback shift register realizations, with m -input &-output delay elements, such that the corresponding characteristic polynomial is $x^n - 1$. We show the equivalence between these realizations and a set of specially structured matrices. Furthermore, the number of realizations is computed for some special cases.

Shivshankar Mishra, Ram Racksha Tripathi and Devendra Kr. Tripathi, "Implementation of Configurable Linear Feedback Shift Register in VHDL" This paper focus on the implementation of configurable linear feedback shift register (CLFSR) in VHDL and evaluates its performance with respect to logic, speed and memory requirement in FPGA. Behavioral implementation of LFSR in VHDL is configurable in terms of number of bits in the LFSR, the number of taps, positions of each tap in the shift register stage and seed value of LFSR. The target device used for implementation of LFSR is Xilinx Virtex-4 FPGA. For waveforms simulation and synthesis of CLFSR Xilinx ISE 9.2i tool is used. The output and timing report are also discussed.

Walter Aloisi and Rosario Mita, "Gated-Clock Design of Linear-Feedback Shift Registers" They presented a method to reduce the power consumption of the popular linear feedback shift register. The proposed scheme is based on the gated clock design approach and it can offer a significant power reduction, depending on technological characteristics of the employed gates. Moreover, the analytical condition that must be satisfied to achieve a power reduction of the gated-clock circuit has been found. Theoretical analysis was validated through many transistor-level SPECTER simulations in the CADENCE environment by using the 0.35- μ m digital standard cells technology supplied by AMS. Simulation results have shown a power reduction of about 10% with a mean error of about 3% with respect to theoretical derivations.

Sergey S. Krivenko, “Many-to-many Linear-feedback Shift Register” There are both the one-to-many and many-to-one types of linear-feedback shift registers. In this paper we filled the gap between them. Our linear-feedback shift register has many-to-many configuration. It generates the Galois field differently from the finite field formed Galois linear-feedback shift register. We consider inferences for appropriate primitive polynomials. Our principle can be implemented very efficiently on common microprocessors.

Chao Cheng, Keshab K. Parhi, “High Speed VLSI Architecture for General Linear Feedback Shift Register (LFSR) Structures” Based on previous three-step high-speed VLSI architecture for LFSR structures, this paper proposes an improved three-step LFSR architecture with both higher hardware efficiency and speed. Generator polynomials for the first and third steps are constructed with iterative small length polynomials, which can in turn be easily handled by proposed look-ahead pipelining algorithm. A new scheme is also proposed for cutting down the iteration bound of the LFSR structure in the second step. This architecture can be applied to any LFSR structure for high-speed parallel implementation. For example, for the parallel BCH (8191, 7684) encoder with different unfolding factors J from 8 to 32, the proposed design can achieve speedup of 2.83% to 15.78% and XOR gates savings of 9.67% to 26.28%

Abdullah Mamun and Rajendra Katti, “A NEW PARALLEL ARCHITECTURE FOR LOW POWER LINEAR FEEDBACK SHIFT REGISTERS” Low power dissipation is very critical in today's electronic designs. Components which are widely used in design, such as sequence generators like linear feedback shift registers (LFSR), should consume as little power as possible. Two recent works on parallel architecture of LFSR, one by M. Lowy and another by M. E. Hamid and C. I. H. Chen, have reduced dynamic power consumption significantly compared to the conventional architecture and showed the way to generate multiple outputs. In this paper we propose design improvements on these parallel architectures. The proposed method reduces dynamic power dissipation significantly, simplifies the design process for single and multiple output generation, and eliminates the need of some hardware.

Reda H. Seireg and Andre G. Vacroux, "STEADY STATE ALIASING ERROR PROBABILITY FOR MULTI-INPUT LINEAR FEEDBACK SHIFT REGISTERS" The traditional way to calculate the Steady State Aliasing Error Probability (SS-AEP) is to calculate a general expression for the AEP with the length of the input pattern equal to n and then limiting this expression. In this paper, we use Markov state diagrams to model the Multi-Input Linear Feedback Shift Register (MISR) for two cases (MISR with primitive polynomials and MISR with non-primitive polynomials) and we calculate the SS-AEP by solving the local balance equations. This new approach is straightforward and easy to handle. Moreover, in some cases the new results differ from those previously published.

R.saraswathi and R.manikandan, "Using classifiers to predict linear feedback shift registers" some new ideas that justify the use of artificial intelligence techniques in cryptanalysis are presented. The main objective of that paper was to show that the theoretical next bit prediction problem can be transformed into a classification problem, and this classification problem could be solved with the aid of some AI algorithms. In particular, they showed how a well-known classifier called c4.5 could predict the next bit generated by a linear feedback shift register (LFSR, a widely used model of pseudorandom number generator) very efficiently and, most importantly, without any previous knowledge over the model used. In this paper we look for other classifiers, apart from c4.5, that could be useful in the prediction of LFSRs. We conclude that the selection of c4.5 in [1] was adequate, because it shows the best accuracy of all the classifiers tested. However, we have found other classifiers that produce interesting results, and we suggest that these algorithms must be taken into account in the future, when trying to predict more complex LFSR-based models. Finally, we show some other properties that make the c4.5 algorithm the best choice for this particular cryptanalytic problem.

Mohmed Sharaf I and Hala A.K.Mansour, “A COMPLEX LINEAR FEEDBACK SHIFT REGISTER DESIGN FOR THE A5 KEYSTREAM GENERATOR” A5 Encryption / Decryption algorithm depends on the use of linear feedback shift registers (LFSR) in order to generate the key-stream bits. The A5-1 algorithm uses three different LFSRs With a majority function to add non-linearity, while A5-2 uses four LFSRs. In this paper a new mdti-configurable LFSR design variant of the A5 algorithm is proposed, in which the inverse taps of the LFSR are used to generate the output key-stream bits. The new LFSR design, called a Complex LFSR, techniques can be effectively applied in software or hardware to a communications stream to provide a low cost and high speed encryption capability.

Yugal Maheshwari, Kleber Stangherlin, Derek Wright and Manoj Sachdev, “Ultra Low-power, Low-energy Static Single-phase Clocked Flip-flop” Flip-flops are a key component of digital integrated circuits and substantially affect their power and energy consumption. In this paper, an ultra low-power, contention-free, static single-phase 3-transistor clock load flip-flop is described and referred to as 19-T Ultra Low-power Flip-flop (ULFF). Simulation results in CMOS 65 nm technology show that at nominal conditions and Data Activity (DA) of 10%, the ULFF has 56% and 7% low-power consumption compared to the 18-T Single-phase Clocked Static Flip-flop (18 TSPC), and Topologically Compressed Flip-flop (TCFF), respectively. Similarly, ULFF has energy-efficiency comparable to Low-power at Low-data activity Flip-flop (LLFF) at low DAs, and 18TSPC at high DAs. A 19-transistor contention-free static single-phase 3-transistors clock load ultra low-power flipflop referred to as ULFF. Simulation results show that ULFF has low-power consumption compared to all the other considered flip-flops. Its energy consumption at nominal supply voltage is amongst the lowest in all flip-flops. In conclusion, ULFF is best suited for low-power and low-energy applications.

Abdullah Hasan and Khaldoon Abugharbieh “A D-Type Flip-Flop with Enhanced Timing Using Low Supply Voltage” This work proposes a novel master-slave latch D-type Flip-Flop. It consists of a reset-set slave latch and an asymmetrical single data input master latch. By reducing the number of stages and removing signal conditioning circuitry in the master latch, setup time has been significantly reduced and power consumption has improved. The proposed flip-flop is competitive to other state of the art low power flip-flops in addition to the conventional Transmission Gate Flip-flop (TGFF) in terms of performance, power consumption and area. In simulations, the proposed flip-flop has achieved up to 71.5% improvement in setup time, 36.5% improvement in D-Q delay time and up to 56.5% less power delay product (PDP) with 10% data activity compared to Topologically Compressed Flip-Flop (TCFF), which is a low power flip-flop. Further, it has achieved 11% smaller circuit area compared with TGFF. This work includes the proposed flip-flop’s circuit schematic, layout design and simulations using Hspice tool with 28nm CMOS technology and a 1V supply voltage at 1 GHz clock (CLK). This work presented a novel low power flip-flop design, which is built from 22 transistors only. By reducing data path stages and clock network transistors count to only 6, the proposed design has achieved high performance keeping low power consumption. When simulated at the slowest process conditions, the proposed flip-flop has achieved 57ps setup time and 336ps D-Q delay time, which is almost the same as TGFF. In terms of power, the proposed design consumes only 0.54 μ W with 10% data activity, which is 34% of the power consumed by TGFF. Moreover, the proposed flip-flop has proved to be functional at 1.8GHz with 100% data activity.

Neha Sharma and shailza “Low Power Design of Various D-Flip-Flop Techniques using CNFET: A Comparative Study” Technology advancement leads to device operation at sub-threshold level and must be scaled down to nanometer range. Eventually speed and power related issues arise in logic circuits. D-Flip-Flop (DFF) is the heart of the memory storage system. The work in this paper shows the basic implementation of different design techniques of D Flip Flop using Carbon Nanotube

Field Effect Transistor (CNFET) as a low power element. It is analyzed and compared with existing conventional CMOS technology using the HSPICE simulation tool at a 32nm technology node with 1.42nm CNT (Carbon Nanotube) diameter. The power delay product (PDP) simulation is carried out. DFF based on CMOS, C2MOS (Clocked CMOS), POWER PC (Phase clock), GDI MUX (Gate Diffusion Input Multiplexer), and TSPC (True single phase clocked) using CNFET has 76.74%, 71.16%, 35.28%, 62.62% and 60% less PDP compared to CMOS logic. It clearly depicts that the DFFs designed using CNFET have better performance. The work done in this paper gives a comparative study of Carbon Nanotube transistor-based sequential circuit D Flip Flop techniques like CMOS, C2MOS, GDI MUX, POWER PC, TSPC and basic inverter circuit with the CMOS in 32nm technology. CNFET has been used for low power operation. The simulated results clearly reveal that the sequential circuits designed using CNFET are more power efficient than CMOS. TSPC DFF has been proved to be the fastest one with the least number of transistor requirements. The flexible nature of CNFET with its control of threshold voltage with the tube diameter. The reduction in tube diameter eventually reduces drain to source current, which in turn lowers power dissipation in CNFET based circuits as compared to MOS.

2.2 Literature Summary

Dynamic power dissipation depends on the switching activity of the circuit. We analyzed power consumption of TG based D flip-flop at different technology nodes and power saving obtained by applying dynamic XOR based clock gating technique to this flip-flop. This work deals with implementation of a transmission gate based D flip-flop in 3 different technology nodes namely 32 nm, .22 nm and 16 nm. The circuit level simulation result of D flip-flop shows power consumption with and without clock gating at the several frequencies of operation and several data activity factors at these technology nodes. Although the power dissipation decreases with the lower technology node, the additional power saving may be obtained using the dynamic XOR based clock gating approach at higher frequency of operation and low data activity, which has been investigated in this research work.

2.3 OBJECTIVE OF THE PROJECT

1. -To increase the efficiency of the Flip Flop for Low Power applications.
2. -To compare the characteristics of Power, Delay and Power Delay Product. of MOSFET Technology in terms of power, Delay and power delay product.
3. -In order to improve Efficiency Low Power Techniques will be applied in all Circuits.
4. -To identify the Feasibility of Linear feedback shift register.

CHAPTER 3

PROCEDURES OF EXISTING SYSTEM

3.1 Low Power Flip Flop

D-flip flop is used in digital circuits for temporary storage of data or a delay element. D-flip flop is written in CMOS technology MOSFET using 130nm and 32nm technology.

The LCG circuit diagram has DFF, multiplier, adder and mux. To measure the power, delay and PDP, first we have to write the code for each device in the LCG block diagram in HSPICE. Coming to the DFF, we will write the nodes for each transistor as D = 2 CK = 3 , CKbar = 4 After giving the nodes we will write the code and stimulation is done in the HSPICE software.

After simulation it has been proven that compared to existing systems, proposed LCG Architecture serves efficiently in terms of power, delay and power delay product. Capacitance of a diode is directly proportional to its size. This enhances its speed in turn.

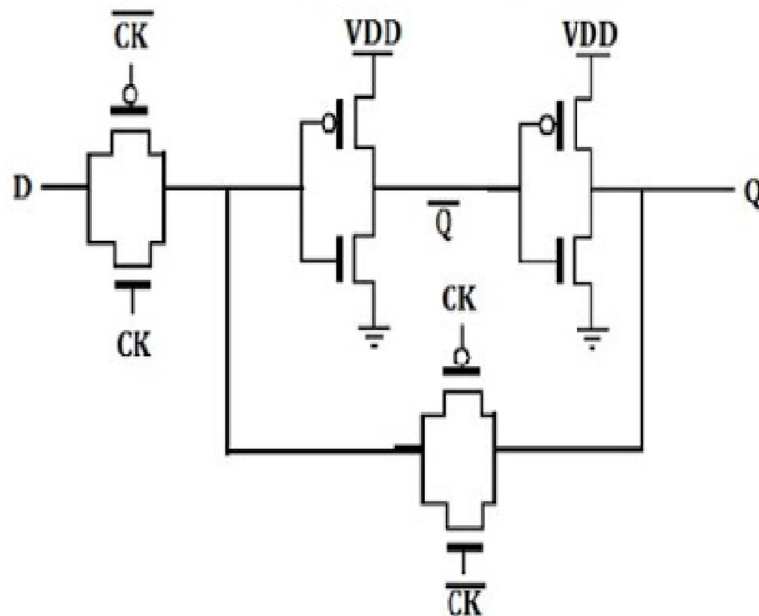


Fig 3.1 Cmos Technology Dff

The LCG circuit diagram has DFF, multiplier, adder and mux. To measure the power, delay and PDP, first we have to write the code for each device in the LCG block diagram in HSPICE. Coming to the DFF, we will write the nodes for each transistor as D=2, CK=3, CKbar=4. After giving the nodes we will code and stimulation is done in the HSPICE software.

After stimulation we will see the power which is nearly 0.00158% and 0.0005293% which is higher than other DFF in MOSFET under 32nm, 130nm and 0.00001037% in CNTFET under 32nm. And the delay we get is 563.86p, 499.84p in MOSFET under 32nm, 130nm and 0.077p in CNTFET under 32nm. Finally, the product of power and delay will be the PDP (Power -Delay product). capacitance of a diode is directly proportional to its size. This enhances its speed in turn.

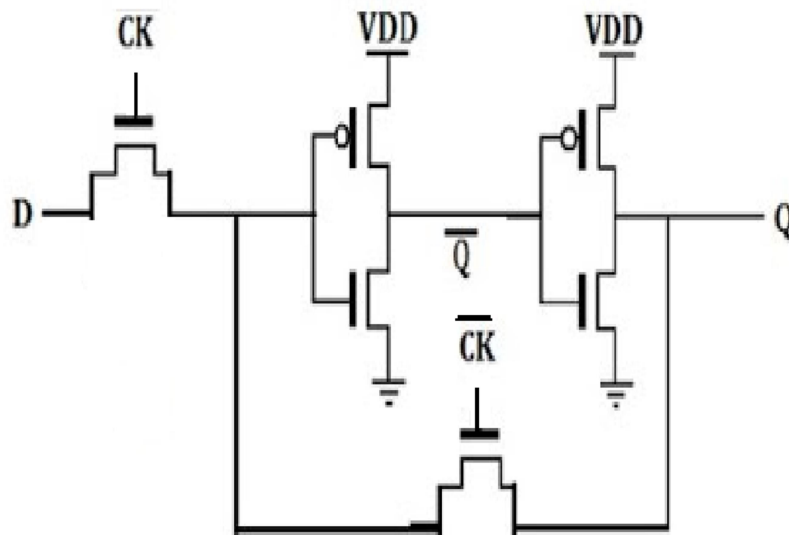


Fig 3.2 MODIFIED CMOS TECHNOLOGY DFF

The modified DFF has seven transistors (two PMOS transistors and four NMOS transistors) which has low power, low delay and lower PDP than existing DFF. We will replace the modified DFF from existing in the LCG circuit diagram to measure the power, delay and PDP. After replacement, we will write the code and stimulation is done in HSPICE software. The power, delay and PDP will be reduced in MOSFET under 32nm, 130nm and CNTFET under 32nm.

The power is reduced to 0.00145% and 0.0000242% in MOSFET under 32nm, 130nm and 0.00000887% in CNTFET under 32nm. From the MOSFET under 32nm, 130nm and CNTFET under 32nm, the CNTFET is best to use and also the more power, delay and PDP reduce in CNTFET only.

3.2 Low Power LFSR

VLSI primary demanding sectors are power, area and performance. The portable device implications are expanding quickly. So circuits should be intended for low power dispersal. Dissipation of power is getting higher in test mode than in typical mode. Subsequently, power optimization at testing is an essential angle. The Present LFSR (Linear Feedback Shift Register) is modified by using the transition controller module. The transition controller is utilized to raise the interrelation between successive test patterns. Simulation and synthesis of the proposed approach are done by using Altera Quartus Prime 16.0 version.

The enormous improvements of Integration fields have driven the gate increment in chips and speed upgrade in chips accompanying minimal consumption of power and area. Integrated Circuit(IC) testing is compulsory because of the increment in density of the device. Integrated circuit examining is relevant in functionality intensification of chips. Testing percussion will affect the ambience of the chip and in addition those included in the strategy. Requirement of time and cost is very obligatory in design progress to consummate the eminent level of quality. EDA tools are utilized to design and test ICs.

Recent devices utilize well advanced processing technology, however, it keeps up the small size by utilizing transistor channel reduction techniques. Because of design complexity, the structural testing is preferred which produces more transition

amid testing at a short span of time. So it dissipates much higher power amid testing, which is not the situation amid typical operation mode. Design complexity enlargement enforces the big demand for recent EDA tools and test procedures. Finding a productive way to overcome the demand is vital for today's system designers.

The test pattern generator has been designed based on LFSR in a low transition manner. This architecture is called LT- GLFSR (Low Transition Generalized Linear Feedback Shift Register). It incorporates GLFSR with bipartite procedure. It is called interpolation of interceder pattern between successions of patterns produced by GLFSR

The bit swapping low transition generalized linear feedback shift register is utilized to reduce the transition between patterns which has been implemented by using a multiplexer which will abate the transition in between the patterns. With this technique, created patterns have a more noteworthy level of arbitrariness and high correlation between continuous patterns yet it has marginally increased transition in the arrangement of patterns produced.

Customarily, increment of transition between progressive test patterns will increase the power consumption. So it will enlarge the power consumption in pattern generators and CUT. By expanding the enable signals to initiate the GLFSR, to decrease the quantity of transition. In the proposed technique, LT-GLFSR can be enacted by four non-overlapping enable signals. This enables the signal to actuate the test design generator halfway and stay out of gear when time of test pattern generation.

The less toggling progressive pattern is produced by using a transition controller. This transition controller is used to decrease the switching of sequent test patterns. Based on the switching of successive test patterns, it will generate patterns. These transition controllers are intended to get the low power test patterns. LFSR with the transition controller approach is utilized to impart the less switching patterns. Modification of progressive test pattern is done by utilizing transition controller intercede pattern interpolation technique. Consider the following test pattern generation by using LFSR Transition controller interpolation technique.

The advanced progressive Quartus Prime design software incorporates a design for FPGAs, SoCs, and CPLDs from design entry and synthesis to optimization, verification, and simulation. Quartus Prime design software is prepared to enhance the productivity of next-generation devices with numerous million logic elements.

The Quartus Prime software comprises newer and faster, scalable algorithms, a new hierarchical database infrastructure, and a novel incorporated compiler technology. The Quartus Prime Edition 16.0 software was used for simulation and synthesis of the proposed design.

CHAPTER 4

PROPOSED METHODOLOGY

4.1 Proposed Flip Flop

The TG based flip-flop in Figure. 1 is realized using TG in the forward path and clocked inverters in the feedback path. This is a negative edge triggered master-slave flip-flop with CN as clock pulse and CPI as inverted clock pulse. The master stage activates for the positive clock and slave stage for the negative clock. Feedback clocked inverters are used in order to overcome input noises. For sizing the transistors, all transistors in the forward path are given high driving capacity to lower the clock-to-Q delay of flip flop and hence to improve its speed of operation. Feedback inverters are kept at minimum size to reduce the area of the overall circuit. The ratio from pmos to nmos taken are 1.5:1, 1:1, 1:1 for 32 nm, 22 nm and 16 nm technology nodes. The supply voltages chosen are 0.45 V, 0.45 V and 0.35 V for 32 nm, 22 nm, 16 nm.

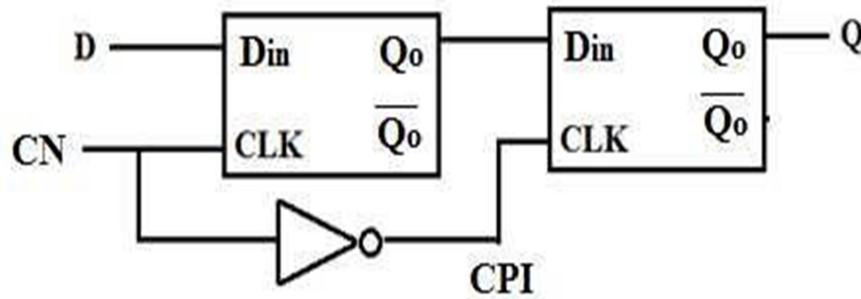


Fig 4.1 Block diagram of flip flop

And hence to improve its speed of operation. Feedback inverters are kept at minimum size to reduce the area of the overall circuit. The ratio from pmos to nmos taken are 1.5:1, 1:1, 1:1 for 32 nm, 22 nm and 16 nm technology nodes. The supply voltages chosen are 0.45 V, 0.45 V and 0.35 V for 32 nm, 22 nm, 16 nm respectively.

The dynamic XOR gate-based clock gating circuit is shown in Figure. 2 [6]. It is a dynamic logic XOR gate with D and Q as its two inputs and its output as gated clock CG_bar. The gated clock (CG) is generated by passing CG_bar through an inverter. For clock gated TG based D flip flop, the inputs CN and CPI of flip-flop circuit of Figure. 1 are replaced with CG and CG_bar respectively.

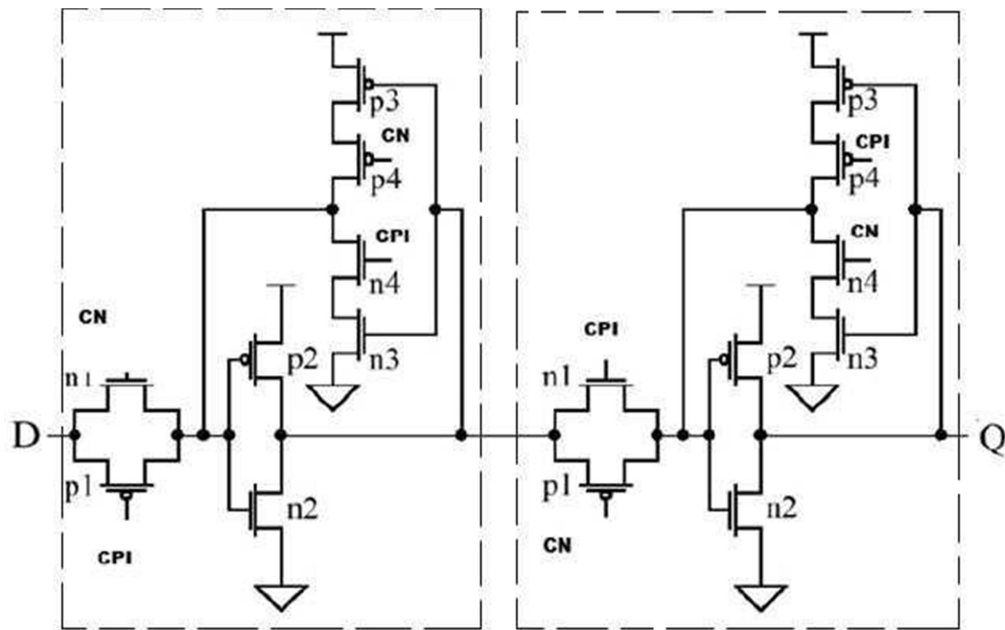


Fig 4.2 proposed Flip flop

4.2 HSPICE SOFTWARE

Hewlett Simulation Program with Integrated Circuit is the full name of HSPICE. Unlike board-level designs made up of individual components, integrated circuits cannot be bread boarded before being manufactured. Furthermore, due to the high cost of photolithographic masks and other manufacturing requirements, it is critical to design the circuit to be as close to perfect as possible before it is designed. Until committing to producing an integrated circuit, the industry-standard method of verifying circuit operation at the transistor level is to simulate the circuit with SPICE.

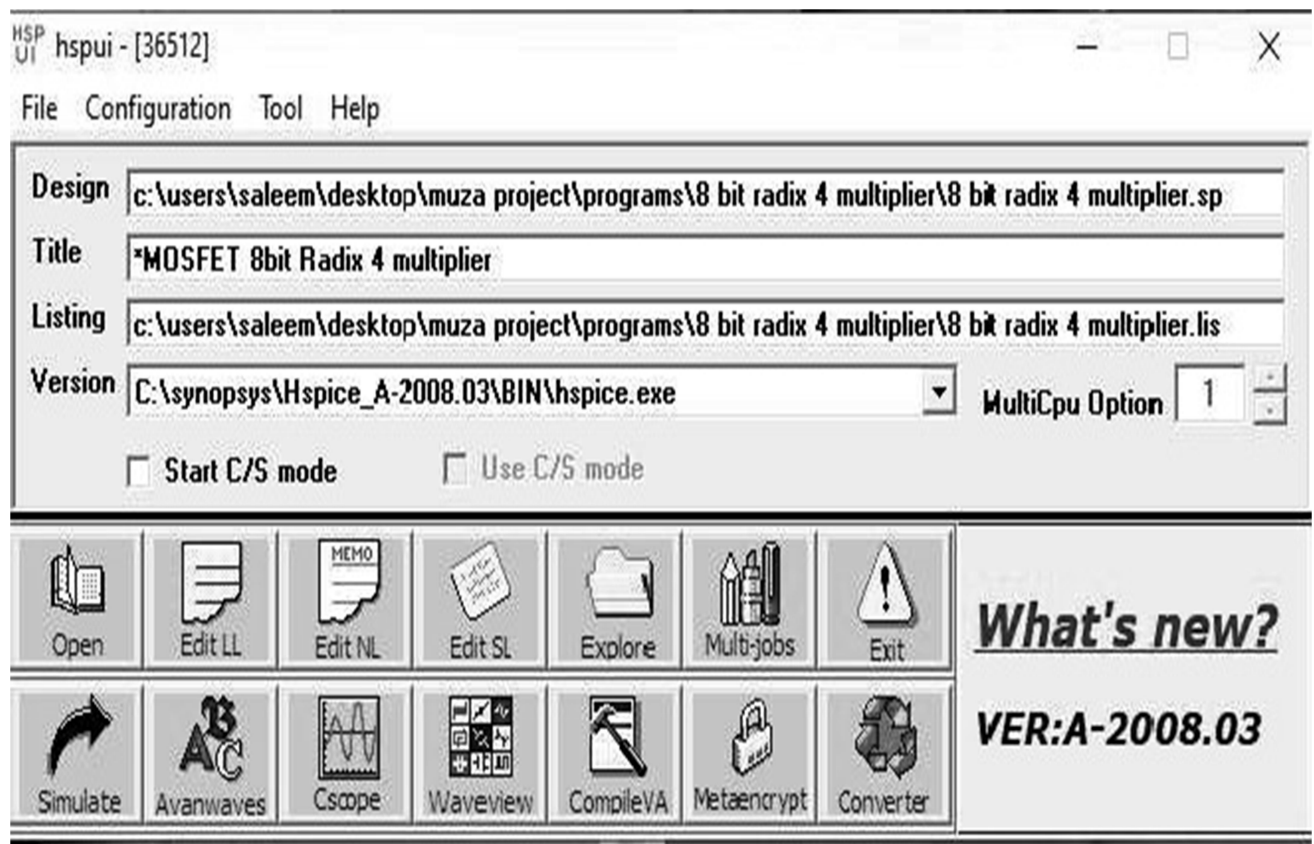


FIG 4.2 HSPUI INTERFACE

The general procedure for analyzing a circuit is to use an editor (vi, emacs, etc.) to create an input file, run HSPICE to generate graph and hardcopy data files, and run MetaWaves to view, customize, and print the resulting graphs. It is usually easier to start with an example input file (such as those at the end of this handout) and modify it to match the desired circuit. The commands found in the example input files and summary in this handout are sufficient to perform simple simulations. For more sophisticated analyses consult the HSPICE USERS MANUAL.

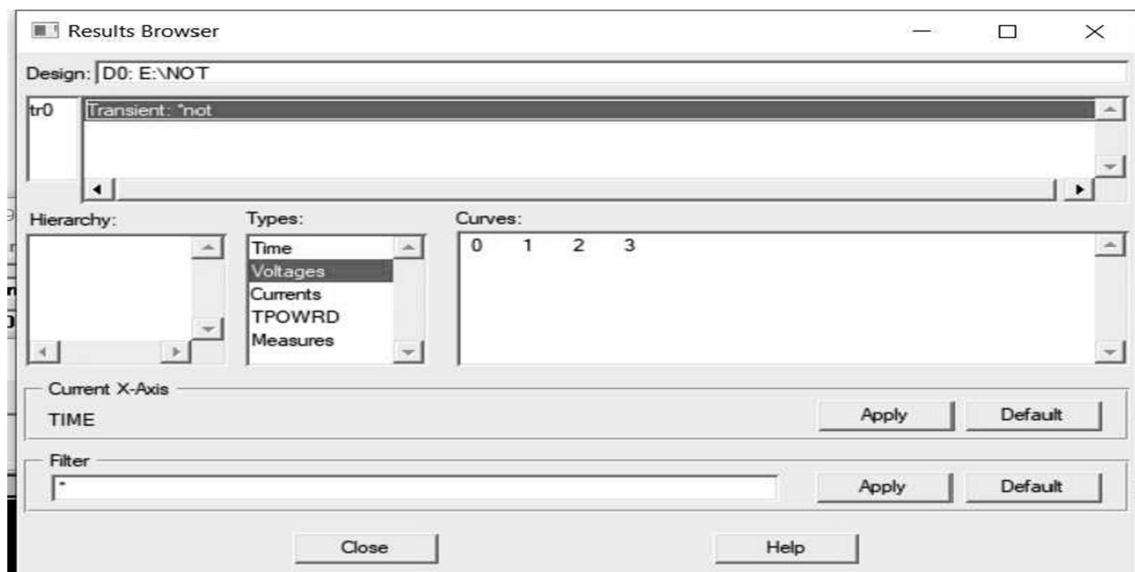


FIG 4.2 Results Browser

4.2.1 SIMULATED SCREEN

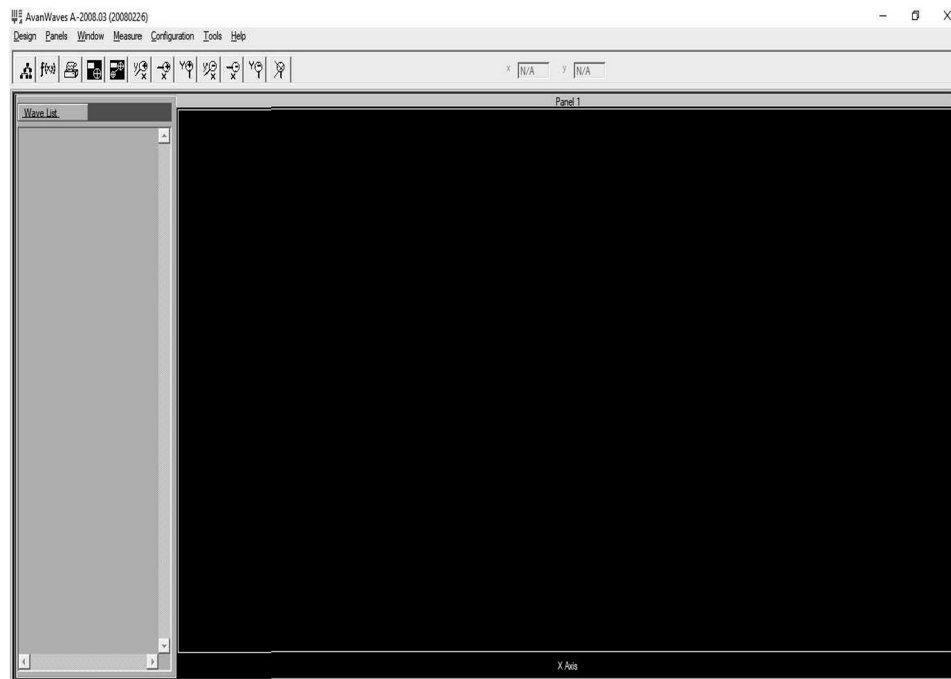


FIG 4.2.1 INTERFACE

4.2.2 COSMOSCOPE

Cosmoscope is a software which is used to calculate the Delay value.

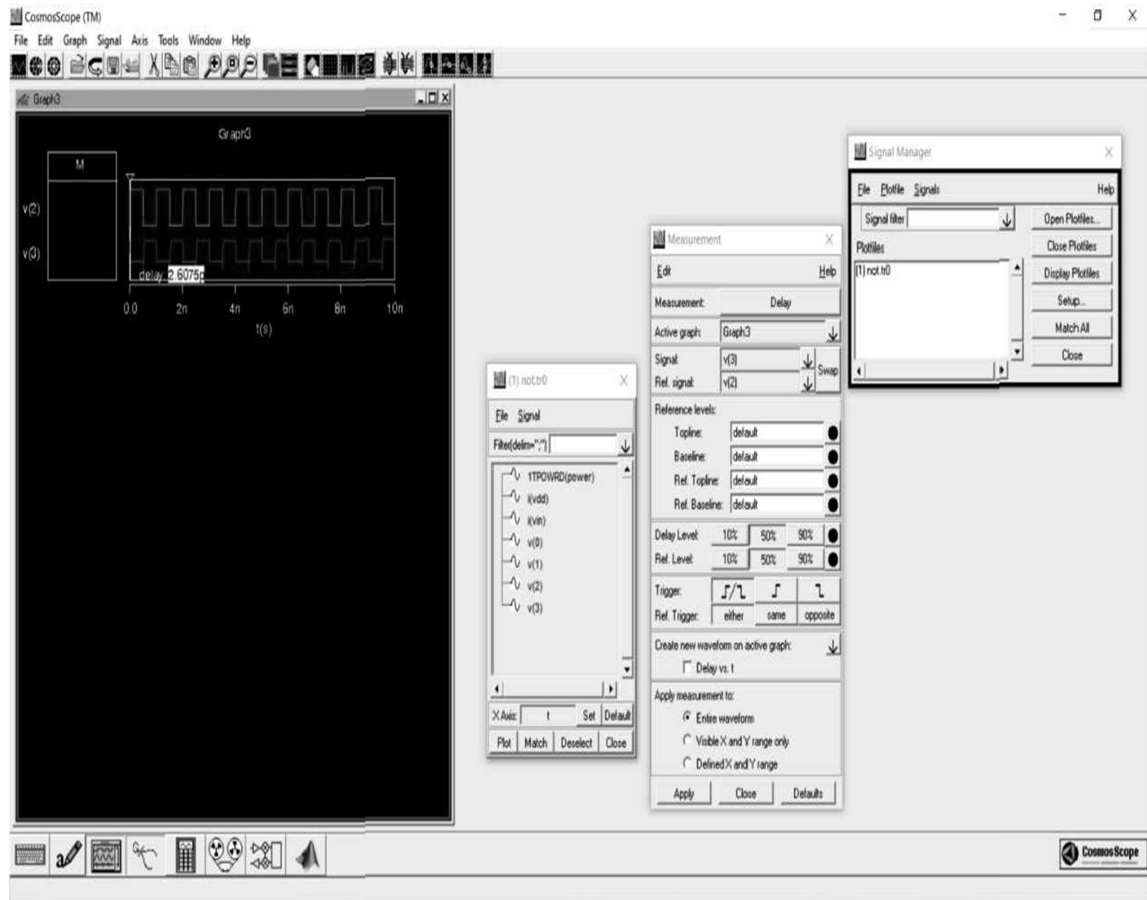


FIG 4.2.2 COSMOSCOPE

4.2.3 POWER

Electric power is the rate at which electrical energy is transmitted by an electric circuit per unit of time. It is the pace at which work is completed. Electrical power is the rate at which electrical energy is transmitted by an electric circuit per unit time in terms of an electric circuit.

The unit of the Electric Power = Watt

$$\text{Power} = V \times I \text{ (W)}$$

$$I = Q/t$$

Where:

W= power in watts

V = potential in volts

I = current in amps

Q = charge in coulombs

t= time in seconds

4.2.4 Delay

The time it takes for a signal to reach its destination is referred to as delay. The time it takes for the consequence of a shift in input to be noticeable at the output is known as the delay of a logic gate.

4.2.5 Power Delay Product

Power Delay Product is the product of the power and delay.

$$\text{PDP} = \text{Power} \times \text{Delay}$$

The power delay product (PDP) is mainly used to determine the performance of a digital integrated circuit. In order to understand the meaning of the PDP, consider square wave $V_{in}(t)$ is applied to the CMOS Inverter; the output voltage $V_{out}(t)$.

CHAPTER 5

RESULT AND DISCUSSIONS

The existing and proposed architecture are simulated by using Avan Waves simulator. Output of the existing radix 4 booth multipliers and proposed radix 4 booth multipliers are analyzed from the results and it is concluded that the proposed architecture reduces the power consumption compared with the existing system. The Synthesis is done using Hspice.

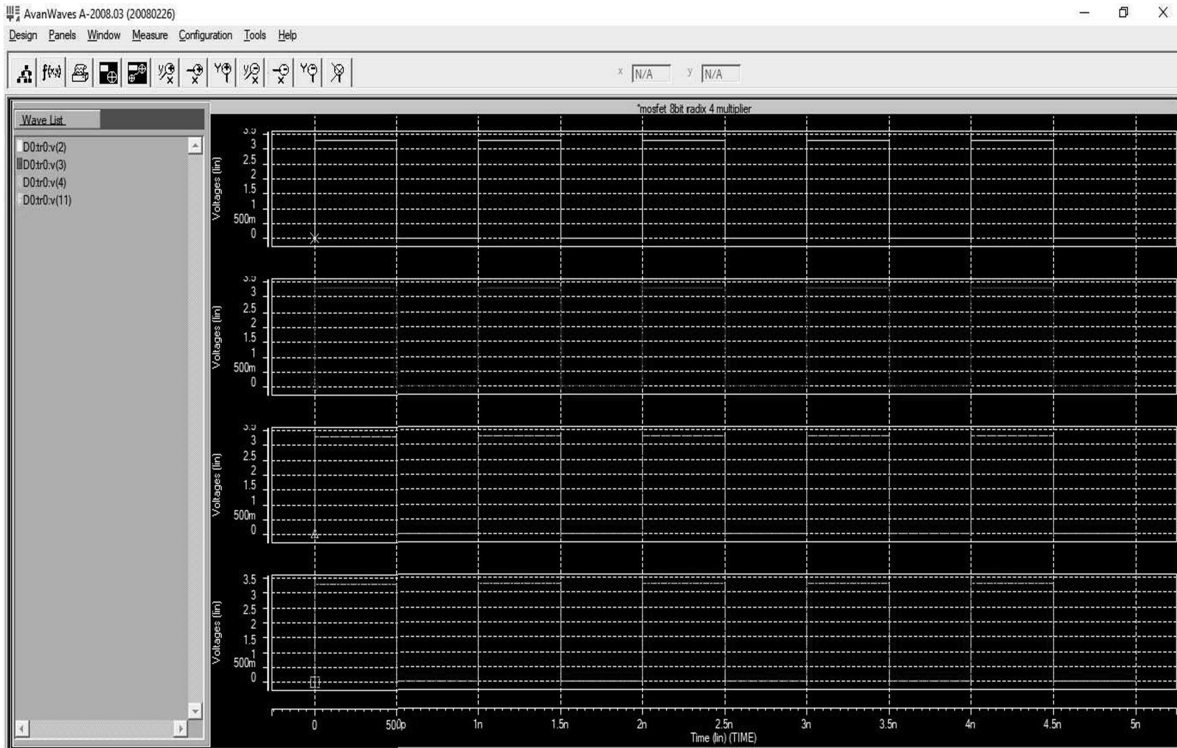


Fig 5.1 Simulation result of Half Adder

Figure 5.1 shows the simulation output of a HA. The inputs given were a and b the output is taken by c and d.

5.2 SIMULATION RESULT OF PARTIAL PRODUCT



Fig:5.2 Simulation Result Of Partial Product

Figure 5.2 shows the simulation output of a PARTIAL PRODUCT. The inputs given were a, b and the output is taken by c.

5.3 SIMULATION RESULT OF PPS

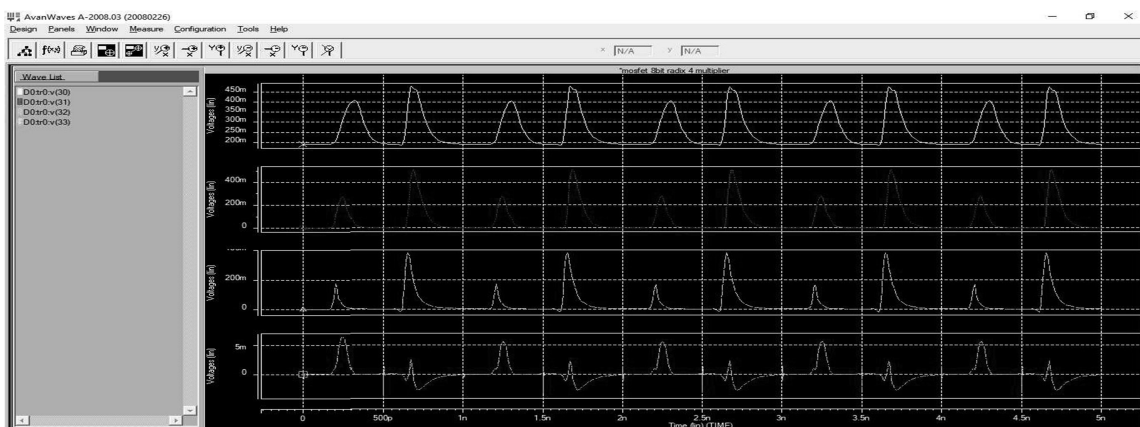


Fig 5.3 Simulation result of Pps

Figure 5.3 shows the simulation output of a Full Adder. The inputs given were a, b

and the output is taken by c

5.4 Simulation Result Of Full Adder

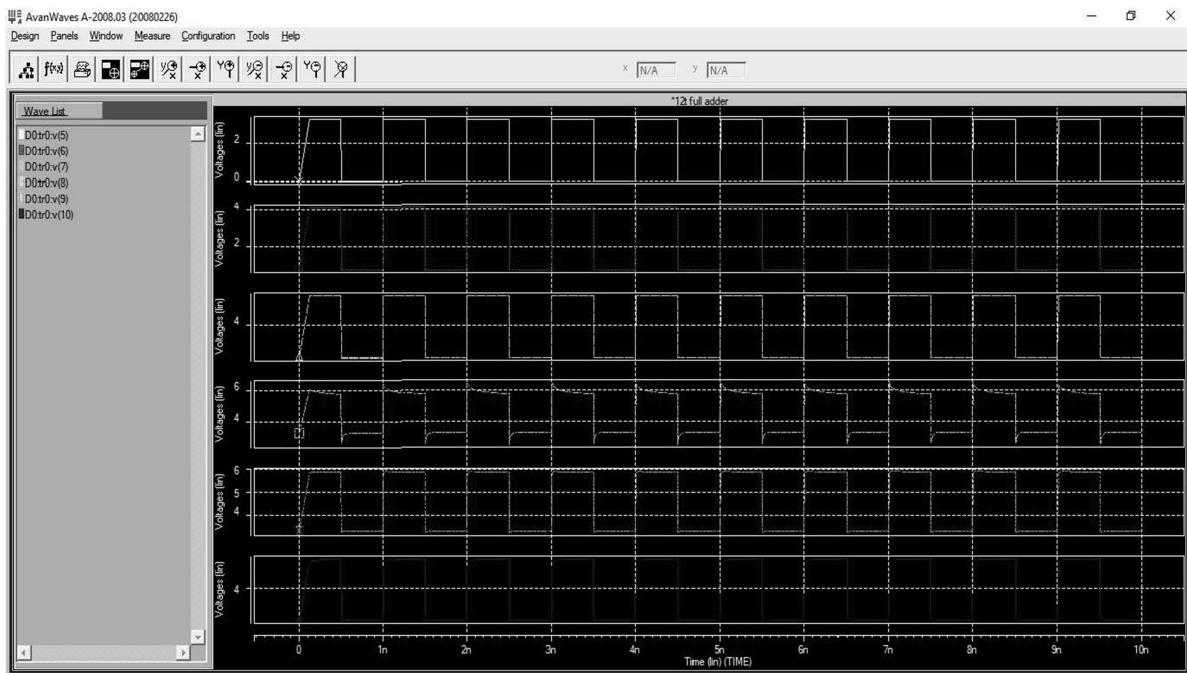


Figure 5.4: Simulation results for Full Adder Figure

5.4 shows the simulation output of a Full Adder. The inputs given were a, b and the output is taken by c.

5.5 Simulation Of D-Flip Flop



Fig 5.5 Simulation of D-Flip flop

5.6 PERFORMANCE ANALYSIS

The performance analysis of HALF ADDER, PARTIAL PRODUCT, PPS, FULL ADDER, D-FLIP FLOP in terms of power has been given below.

5.6.1 Performance Analysis Of Half Adder

Table 5.1 Design analysis of 130NM HALF ADDER

HALF ADDER (130 NM) PARAMETERS	VALUES
POWER	$1.556 \times 10^{-5} \text{W}$
DELAY	$46.594 \times 10^{-12} \text{s}$
POWERDELAY PRODUCT	$72.5002 \times 10^{-17} \text{J}$

5.6.2 Performance Analysis Of Partial Product

HALF ADDER (32NM) PARAMETERS	VALUES
POWER	$4.849 \times 10^{-8} \text{W}$
DELAY	558.58F s

Table 5.3: Design of 130NM PARTIAL PRODUCT

Table 5.4 Design analysis of 32NM PARTIAL PRODUCT

5.6.3 Performance Analysis Of Program Stored

Table 5.5 Design analysis of 130NM PROGRAM STORED

Table 5.6 Design analysis of 32 NM PROGRAM STORED

5.6.4 Performance Analysis Of Pps

Table 5.7: Design analysis of 130NM PPS

Table 5.8: Design analysis of 32NM PPS

5.6.5 Performance Analysis Of Full Adder Gate

Table 5.9: Design analysis of 130NM FULL ADDER

Table 5.10: Design analysis of 32NM FULL ADDER

5.6.6 Performance Analysis Of Low Power Flip Flop

Table 5.11: Design analysis of 130 NM low power flip flop

EXISTING FLIP FLOP			
PROPOSED FLIP FLOP			

CHAPTER 6

CONCLUSION

The TG based D flip-flop and 1 and clock gated TG based D flip flop as explained in subsection 2.2 are simulated for three technology nodes namely 32 nm, 22 nm, 16 nm. The Figure. 3 shows the transient responses of the TG based D flip-flop without clock gating. The output Q follows the D at every negative edge of the clock pulse.

The transient responses of the TG based D flip-flop with clock gating. The output Q follows the D at every negative edge of the gated clock pulse. The instantaneous power waveform of TG D flip flop without clock gating .while instantaneous power waveform of TG D flip flop with clock gating.

The analysis assumes following parameters:

fclk: frequency of clock signal. The clock periods are selected as 200 ns, 60 ns, 20 ns and 5 ns which gives fclk as 5 MHz, 16.67 MHz, 50 MHz and 200 MHz respectively. Data activity factor: % change in data in every clock cycle [if data do not changes then data activity factor is 0 %, if data changes every twelve clock cycle then data activity factor is 8.33 % (~8 %), if data changes every six clock cycle then data activity factor is 16.66 % (~16 %), if data changes every three clock cycle then data activity factor is 33.33 % (~33 %), if data 'D' changes every clock cycle then data activity factor is 100, etc.]. The data activity factor reduces the switching at the internal nodes of the flip flop, i.e. if data 'D' does not change then switching at internal nodes of flip flop will reduce. Rise and fall times: The rise and fall times kept equal value of 50 ps for data, clock and enable signal. Pavg: Average Power dissipation in watt .% Power savings: Reduction in power after clock gating with respect to power without clock gating.

Firstly the average power dissipation of TG D flip flop with and without clock gating circuitry are compared for the different technology nodes at 16.67 MHz clock frequency and 33% data activity as shown in Table 1.

Technology Nodes (fclk = 16.67 MHz, Data activity=33%)	Avg. Power dissipation (in nW)	
	W/O clock Gating	With Clock Gating
32 nm	6.6904	4.9423
22 nm	5.5186	4.8927
16 nm	4.0103	4.5460

Fig 6.1 Data activity

The work presented an analysis of a TG based D flip-flop with and without clock gating technique to study power dissipation and power saving considering the data activity factor and frequency. The results were also compared for 3 deep sub- micron technology nodes to include the effect of technology scaling. It is concluded that in TG based D flip flop this type of clock gating is suitable for saving power only at higher clock frequency and low data activity factor. The effect of clock gating at lower technology node reduces either at low clock frequency or at higher data activity factor. But, in 16 nm technology node, at low data activity clock gating is less effective to save average power since leakage power is the more dominant component). The high speed and/or low data activity circuits designed using higher technology nodes (where normal power dissipation is generally more) the presented clock gating technique in a TG based D flip-flop is advantageous for larger power saving.

REFERENCES

- [1] Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators, Application Note, Xilinx Inc.
- [2] Linear Feedback Shift Registers in Virtex Devices. Application Note, Xilinx Inc.
- [3] Mohammed Gazi. J, Abdul Mubeen Mohammed: Design of Random Testing Circuit Based on LFSR for the External Memory Interface. International Journal of Application or Innovation in Engineering & Management (IJAIEEM), vol. 2, issue 3, pp.145- 150, March 2013.
- [4] Shikha Kakar, Balwinder Singh and Arun Khosla: Implementation of BIST Capability using LFSR Techniques in UART. International Journal of Recent Trends in Engineering, Vol 1, No. 3, pp. 301- 304, May 2009.
- [5] Khushboo Sewak, Praveen Rajput, Amit Kumar Panda.: FPGA Implementation of 16 bit BBS and LFSR PN Sequence Generator: A Comparative Study. In the proceedings of IEEE Student Conference on Electrical, Electronics and Computer Sciences, MANIT Bhopal, India, pp. 1-2, March 2012.
- [6] Rai S Katti, Sudarshan K Srinivasan,: Efficient hardware implementation of a new pseudo-random bit sequence generator. In the proceeding of IEEE International Symposium on Circuits and Systems, ISCAS 2009, pp. 1393-1396, 2009.
- [7] Amit Kumar Panda, Praveen Rajput, Bhawna Shukla.: Design of Multi Bit LFSR PNRG and Performance comparison on FPGA using VHDL. International Journal of Advances in Engineering & Technology (IJAET), Vol. 3, Issue 1, pp. 566-571, March 2012.
- [8] J. Bhasker: A VHDL Primer. Prentice Hall India, 3rd edition.
- [9] Saranyadevi S and Thangavel M.: A Low Power Structure Design of 2D-LFSR and Encoding Techniques for BIST. International Journal of Advanced Science and

Technology. Vol. 18, pp 11-22, May 2010.

[10] Chien-In Henry Chen, Kiran George: Configurable two-dimensional linear feedback shift registers for parallel and serial built-in self-test. IEEE transactions on Instrumentation and Measurement, Vol. 53, no. 4, pp. 1005–1014, Aug. 2004.

[11] Avinash Ajane, Paul M. Furth, Eric E. Johnson, Rashmi Lakkur Subramanyam,: Comparison of binary and LFSR counters and efficient LFSR decoding algorithm. In the proceeding of 54th IEEE International Midwest Symposium on Circuits and Systems, pp. 1–4, 2011.

[12] Amit Kumar Panda, Praveena Rajput, Bhawna Shukla: FPGA Implementation of 8, 16 and 32 Bit LFSR with Maximum Length Feedback Polynomial using VHDL. International Conference on Communication Systems and Network Technologies, pp.769-773, 2012.

[13] Dr. R.V. Kshirsagar, Purushottam Y. Chawke: Design of 8 and 16 Bit LFSR with Maximum Length Feedback Polynomial & Its pipelined Structure Using Verilog HDL. International Journal on Recent and Innovation Trends in Computing and Communication, Vol. 2 issue 11, pp 3337- 3339, November 2014.

[14] Shruti Hathwalia, Meenakshi Yadav: Design and Analysis of a 32 Bit Linear Feedback Shift Register Using VHDL. International Journal of Engineering Research and Applications, Vol. 4, Issue 6 (version 6), pp. 99-102, June 2014.

[15] Stephen Lim, Andy Miller: LFSRs as Functional Blocks in Wireless Application. Xilinx application note, XAPP220 (v1.1), pp 1-10, 2001.

[16] Chao Cheng, Keshab K Parhi: High speed VLSI architecture for general linear feedback shift register (LFSR) structures. Conference Record of the Forty-Third Conference on Signals, Systems and Computers, pp. 713–717, 2009.

[17] Muthiah, D., Arockia Basil Raj, A.: Implementation of high-speed LFSR design

with parallel architectures. International Conference on Computing, Communication and Applications (ICCCA), pp. 1–6, 2012.

[18] Kitsos P, Sklavos N, Zervas N and Koufapavlou O (2001),” Reconfigurable linear feedback shift register (LFSR) for the .bluetooth System”, IEEE Proceedings International Conference on Electronics, Circuits, and Systems, Malta, September 2001.

[19] Rose O ,”A Stream Cipher Based on Linear Feedback over GF (29”, IEEE Press, 1998.

[20] Abramovici. M, Brewt. MA , Friedman AD,”Digital Systems Testing and Testable Design”, IEEE Press, New York. 1990.

[21] NIST Special Publication 800-22” A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”, 2001.

[22] Golomb S,” Shift Register Sequences”, Aegean Park Press, Laguna Hills, Ca, Revised Edition 1982.

[23] Schneier. B, “Applied Cryptography” Prentice Hall, 1998.