A Low power Implementation of for BCH encoders and cyclic redundancy check Using LFSR

[1]Muddu Haripriya & [2]Swetha Mammidi

[1]PG Scholar, Dept. of ECE, Sri Venkatesa Perumal *College of Engineering* and Technology, Puttur, Chittoor, AP, India.
[2]Assistant Professor, Dept. of ECE, Sri Venkatesa Perumal *College of Engineering* and Technology, Puttur, Chittoor, AP, India.

Gmail Id: mudduharipriya171@mail.com & swethamammidi@gmail.com

Abstract— Linear feedback shift registers (LFSRs) are used to implement BCH encoders and cyclic redundancy check (CRC), which are broadly used in digital communication systems. Previous parallel LFSR designs adopt a state-space transformation that shortens the feedback data path and reduces the gate count. Transformations have been designed to minimize the total gate count of the three involved matrix multiplications. However, the transformation matrix multiplication is only active for one clock cycle at the end. In this paper, we propose an alternative transformation matrix construction that effectively shifts the complexity from the other two matrices, which are active in every clock cycle, to the transformation matrix without increasing the critical path or the total gate count. For an example CRC-32, the proposed design achieves 33% power and 8% gate count reductions without compromising the achievable clock frequency.

I INTRODUCTION

The highly emerging digital VLSI system design in the communication field requires fast generation of random patterns for error detection, VLSI circuits testing, data encryption and decryption. The test patterns for the above applications are generated by Linear Feedback Shift Register (LFSR). The 8 bit pattern generation circuit to generate a pattern X^7+X^5 +X^4+X^3+1 is shown in Figure 1. As the technology advances, the need for low power consumption circuit increases. Thus, the circuit is generally expected to be designed in such a way that it should consume less power, occupy minimum area with improved response time. The use of flip-flop with activated clock in the register design consumes more power which is not sufficient for high throughput, so pulsed latches are used in the place of flip-flops in this proposed work.

For reducing the power consumption of the device, various methodologies are available in the literature. Dropping the number of transitions is one of the means for power optimization. Transitions are reduced by swapping the bits and applying clock to half part of the circuit. Clock gating is also employed for power optimization. Although various optimization techniques are implemented for minimizing the power consumption of the device, they are not eventually much effective by the means of reducing the response time and area. Like power optimization techniques, techniques for minimizing the area and increasing the speed are also employed in [1]-[5]. The conventional method of serial to parallel architecture and pipelining algorithms are used to increase the speed of the shift register. Also calculation of output value only by considering the past feedback value in the transposed serial architecture, increases the speed. The transformation from long LFSR sequence to several short LFSR sequence in series reduces the overhead. Though several techniques are used to reduce the power, area and speed, they are not efficient in terms of critical path delay.

II LINEAR FEED BACK SHIFTREGISTER

Implement 8-bit Fibonacci type LFSR and 8-bit Galois type LFSR using the same characteristic polynomial:

$$X8 + X6 + X5 + X4 + 1$$

Measure and display the phase shift between serial output sequences of both LFSRs (i.e. number of clock cycles the output sequence of Galois type LFSR lags behind the output sequence of Fibonacci type LFSR, or vice versa). Simulate and implement the project on FPGA development board. If the initial seed value of the Fibonacci type LFSR is a hexadecimal number formed from two least significant digits of the student ID code (in case of 00 move one digit to the left until the number is non-zero), which initial seed value of the Galois type LFSR allows to eliminate the phase shift altogether? Optionally, try to make the polynomial and LFSR seeds reconfigurable on-the-fly, measure and display the length of the LFSR output sequence (number of bits in the output sequence before it starts to repeat itself).

Overview of the basic elements of the synchronous sequential logic

In contrast to combinational logic (e.g. comparator and different adders discussed in the previous labs), the output of sequential logic depends not only on the current input values, but also on the previous input history. This means that sequential logic

possesses an internal state that changes in accordance with the applied sequence of input values. The result is that the same set of input values can generate a different output since the internal state of the sequential logic element might have been different. The sequential circuits can be of asynchronous or synchronous type. In asynchronous circuits the internal state starts changing immediately in response to the change of inputs. In synchronous circuits the internal state can change only during certain discrete times that are determined with the clock signal. Clock signal is essentially a series of repetitive pulses with fixed frequency. Synchronous circuits can be configured to change internal state at the rising state (clock signal changes from low to high) or falling edge (clock signal changes from high to low). This manual focuses only on the synchronous type sequential circuits. The basic element of synchronous sequential logic is a flip-flop. It has two stable states (low and high) that can be changed by applying appropriate values to flip-flop's control inputs. A D-type flip-flop is in Figure 1 (as indicated with the "D" letter on its data input). The triangle-shaped input denotes clock signal input, meaning this sequential circuit is synchronous (edge-sensitive). Reset and enable are two commonly used control signals that govern the state change process. When reset input is active, the flip-flow goes into predefined reset state (usually low). Reset input usually has the highest priority, meaning that other control signals are ignored when it is active. Enable input, when set, allows the state change to take place. When enabled (reset is inactive and enable is active) the D-type flip-flop stores the value that is fed to the data input D on the active edge. The internal state of the flip-flop is then propagated to the data output Q.

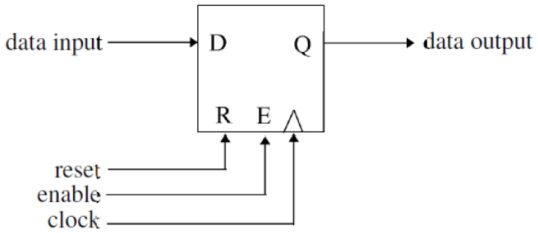

Figure 1: Block Diagram of D-type Flip-flop

One flip-flop can store only one bit of information. However, the data items in digital systems can be much bigger. In order to store multi-bit data flip-flops are grouper into arrays called registers (Figure 2). Each flip-flop has individual input/output that can be accessed in parallel, but all of them share clock and control signals (reset is also shared, but is not shown in Figure 2) that allow joint operation. When arranged in such a way, the data comprising of several bits can be simultaneously written to/read from all the flip-flops.

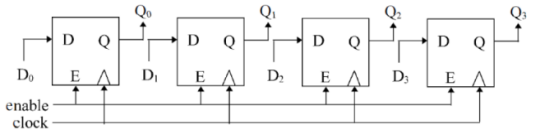

Figure 2: Block Diagram of 4-bit Register Built from Four D-type Flip-flops

Another possible arrangement of flip-flops in a register is to cascade them (data output of one flip-flop goes to the data input of the next one). This type of register is called shift register. An example of a shift register connected in serial-in, parallel-out (SIPO) fashion is shown in Figure 3. All flip-flops share clock and control signals for joint operation, and the content of all flip-flops can be read in parallel (hence "parallel-out"). However, saving of data is performed via 1- bit input D (i.e. serially, hence "serial-in"). Thus it requires four clock cycles to input (shift in) a 4-bit data into such register. Technically, in the same manner data can be read out serially as well (shifted out).

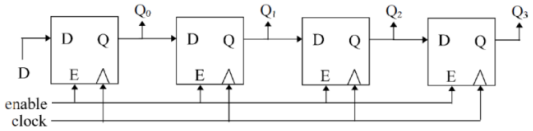

Figure 3: Block Diagram of 4-bit SIPO Shift Register Built from Four D-type Flip-flops

There is a special type of shift register called linear feedback shift register (LFSR). The name indicates that such shift register has a feedback which is a linear function (usually XOR) of its input and current state. LFSRs are mostly used for generation of pseudo-random sequences that find application in cryptography, circuit testing and digital communication among others.

The feedback function of LFSR is usually represented with characteristic polynomial. When the polynomial is known, there are basically two ways to implement LFSR: as Fibonacci type LFSR or as Galois type LFSR. A 4-bit Fibonacci type LFSR with characteristic polynomial X4 + X3 + 1 is shown in Figure 3.1.4. The value of the highest degree in the polynomial indicates the size of LFSR (four bits in this particular case). To construct a Fibonacci type LFSR from a given characteristic polynomial the flip-flops in the shift register are numbered sequentially starting from a flip-flop that is closest to the input. The output of each numbered flip-flop can then be mapped to terms of the polynomial with the corresponding power (as it is shown in Figure 3.1.4). The term 1 (essentially X0) in the polynomial is mapped to the data input of the LFSR. The outputs of the flip-flops that correspond to the terms which are present in the characteristic polynomial are then XOR-ed together to form the feedback of LFSR.

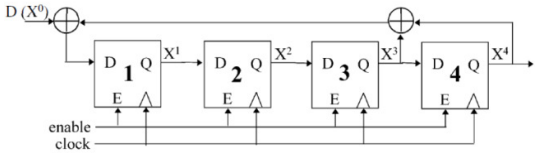

Figure 4: Block Diagram of a 4-bit Fibonacci LFSR with X4 + X3 + 1 Characteristic

Flip-flop stores the value from input "D" XOR-ed with outputs of flip-flops 3 and 4 ("Q(2)" and "Q(3)" respectively) as it is defined in the characteristic polynomial X4 + X3 + 1. Quite often LFSRs are designed as pure generators without data input. In this case input D (X0) is omitted from the feedback equation (even though it is always present in the characteristic polynomial). For such configuration of the LFSR the feedback is completely determined by its current state. Since the total number of possible states is finite (e.g. a 4-bit LFSR has 24-1=15 possible states excluding the one with all zeros) and the operation of the LFSR is deterministic, at some point LFSR will get into its initial state (the seed) and will start to repeat the same sequence of states all over again. The length of this cycle depends on the selected characteristic polynomial. For example, characteristic polynomial X4 + X3 + 1 produces a maximum sequence of all 15 states (excluding the all zeros state) before entering a repeat cycle.

Another widely used sequential circuit is a counter. Counter stores the number of clock cycles it has been active (enabled). The principle structure of a generic 4-bit counter is shown in Figure 5. It comprises of a 4-bit register whose inputs and outputs are connected to a combinational logic block that computes the next digit based on value of the one that is currently stored. The exact logic inside the "next value function" depends on the numerical system that is being employed. It is possible to count in, for example, binary, BCD or Gray encoded formats. Even an LFSR in generator configuration can be considered a counter. Similarly, the direction of counting can be implemented as incrementing or decrementing.

Reversible Logic has received great attention in the recent years due to their ability to reduce the power dissipation which is the main requirement in low power VLSI design. Quantum computers are developed using reversible logic circuits. In 1960 R. Landauer showed that, circuits and systems of high technology constructed using irreversible hardware result in energy dissipation due to information loss. According to him, the loss of one bit of information issipates at least KTln2 joules of energy, in which K refers to Boltzmann's constant and T is the absolute temperature at which the operation is performed. In 1973 Bennett showed that, one can avoid KTln2 joules of energy dissipation by designing the circuits using reversible logic gates.

[1] An introduction to reversible latches by J E Rice This paper introduces the details behind two proposed reversible SR latch designs: one design based on the Fredkin gate and one design based on the Toffoli gate. Both of these designs are verified to behave correctly as compared to the traditional, irreversible SR latch and a short comparison is made. We find that the designs are very similar in behaviour, and little can be found to choose between them [2] A review of reversible logics and it's application in logic design by Nahadi banu and Dibya saha. This paper presents the primitive reversible gates which are gathered from literature and this paper helps researchers/designers in designing higher complex computing circuits using reversible gates. The paper can further be extended towards the digital design development using reversible logic circuits which are helpful in quantum computing, low power CMOS, nanotechnology, cryptography, optical computing, DNA computing, digital signal processing (DSP), quantum dot cellular automata, communication, computer graphics. [3] A Distinguish between reversible and conventional logic gates by B Raghu kanth and B murali Krishna.In this paper, we compared 4-bit reversibleadder/subtractor circuit using DKG gate. [4] Design of sequential circuit element using reversible logic gates by Bhagyalakshmi H R and Venkatesha M K In this paper an attempt is made to realize the basic sequential elements using two new reversible logic gates namely VB-1 and VB - 2. This will help to build more complex reversible sequential circuits.[5] Comparison of cost metrics for reversible and quantum logic synthesis by Dmitri Maslov and D Micheal miller The results in Section 5 lead us to the ollowing conclusion. Minimization of Toffoli gate count as a criterion for a reversible synthesis method is not optimal and even for small parameters may result in a seemingly small circuit which may be as far off a technologically favorable implementation as a factor of 8. It is natural to expect that for circuits with more lines the difference will grow. We suggest that the commonly used gate count metric should be replaced with a metric that accounts for the different costs of large building block (i.e., Toffoli gates), such as the weighted gate cost presented here. Using such a metric would lead to the technologically favorable circuit (b).[6] Efficient building blocks of reversible sequential circuits and V. Kamakoti In This paper proposed the design of a new reversible logic gate which is shown to be advantageous for synthesizing multi valued reversible logic. The proposed reversible gate is utilized for efficiently designing the RS latch. The paper proposes the designs of the reversible Flip Flops and Latch using the proposed gate, Fredkin gate, Toffoli and the Feynman gate. The designs are compared with the existing design and are shown to be have an improvement by a factor of 2 to 6. The proposed designs utilized the fact that the reversible circuits constructed from the logic are better in terms

of logic complexity and garbage minimization than the one obtained by converting the irreversible designs by replacing gates appropriately. The proposed designs are highly optimized. The number of garbage outputs generated and the number of gates used in the designs are very small as compared to the earlier implementations.

III SIMULATION RESULTS

If a substructure, also called a common term, appears in multiple output formulas, then this substructure only needs to be computed once and it can be shared in the computations of multiple outputs. The complexity of constant matrix multiplications is more accurately estimated by applying SS. The SS for achieving optimal gate count reduction is an NP-complete problem. Also different SS schemes lead to trade-offs on the gate count and CPD. To achieve high clock frequency in parallel LFSRs, SS needs to be applied with constraints on the CPD. Although many SS schemes targeting at gate count reduction have been proposed, only a few of them address the critical path issue associated with sharing substructures. In [9], the CPD is computed by constructing a restriction graph describing the SS, and a newly identified substructure is only adopted if it does not lead to CPD violation. In [10], the CPD is computed through updating a delay matrix along with identifying substructions for sharing. Matrices are more friendly to computer execution than restriction graphs. In this section, a simplified vector-based CPD computation method for SS is proposed, and considerations in substructure selection that help to reduce the CPD are discussed. In our SS scheme, first consider that the 2-term pattern appearing most in the outputs is shared each time, since an XOR gate has two inputs. If there is a tie, pick a pattern randomly. Also substructures participate in the pattern matching recursively. Consider an example that computes y0 = x0+x1+x2+x3+x5, y1 = x0 + x1 + x2 + x3 + x4, and y2 = x2 + x3 + x4 + x5. The substructures identified for sharing in iteration 1, 2, and 3 are x6 = x2 + x3, x7 = x6 + x4 and x8 = x0 + x1, respectively. The SS can be expressed by a graph as shown in Fig. 5. Instead of converting this graph to a restriction graph and then apply a recursive process to compute the CPD, a depth vector, d, can be kept and updated along with the identifications of the substructures. d is initially an all-zero vector, when a substruction xl = xi + xj is adopted, an entry dl = maxfxi; xjg + 1 is added to the vector. In the above example, d = [0000001], [00000012], [000000121] in iteration 1, 2, and 3, respectively. In the substructure-shared output formula, a term is either an original input or a shared substructure.

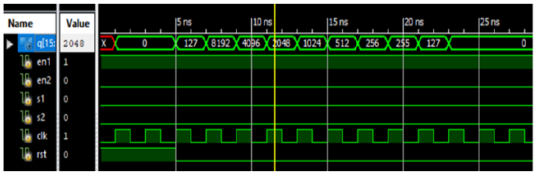

Fig:5 Simulation results of LFSR

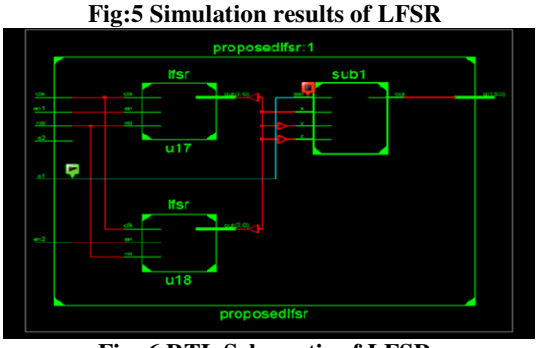

Fig: 6 RTL Schematic of LFSR

Denote the numbers of terms whose values in the final depth vector are 0; 1; 2; _ _ _ by s0; s1; s2; _ _ _ . Then the CPD for computing the output can be derived using Algorithm B. In this algorithm, smax is the last nonzero si.

The num in iteration i is the number of intermediate items available to be added up after i levels of XOR gates, assuming 2-input XORs are used to add up inputs, substructures, and intermediate results as early as possible in a tree structure. This algorithm generates the same result as that in [9], [10] with simpler interpretations and computations. In the above example, y1 is computed as x7 + x8 using SS. d7 = 2 and d8 = 1 in the final depth vector. Hence s0 = 0; s1 = 1, and s2 = 1. Applying Algorithm B, num is initialized to s0 = 0, and becomes 1, 2 and 1 in the iterations that i = 1; 2 and 3, respectively. The iteration breaks when i = 3 and CPD = 3.

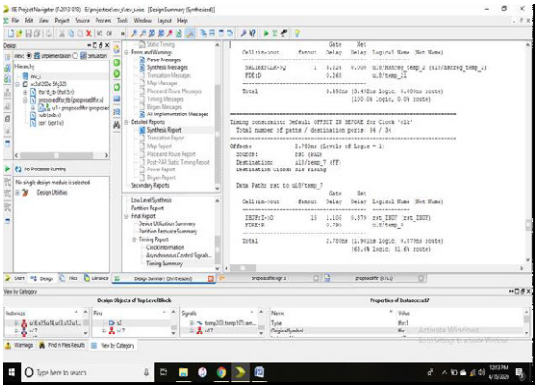

Fig: 7 Total Propagation Delay

IV CONCLUSION & FUTURE SCOPE

A low power test pattern generator has been proposed which consists of a modified low power linear feedback shift register (LP-LFSR). The seed generated from (LP-LFSR) is Ex-ored with the single input changing sequences generated from gray code generator, which effectively reduces the switching activities among the test patterns. Thus the proposed method significantly reduces the power

consumption during testing mode with minimum number of switching activities using LP-LFSR in place of conventional LFSR in the circuit used for test pattern generator. From the implementation results, it is verified that the proposed method gives better power reduction compared to the exiting method.

REFERENCES

[1] BalwinderSingh, Arun khosla and Sukhleen Bindra "Power Optimization of linear feedback shift register(LFSR) for low power BIST" , 2009 IEEE international Advance computing conference(IACC 2009) Patiala,India 6-7 March 2009.

[2] Y.Zorian, "A Distributed BIST control scheme for complex VLSI devices," Proc. VLSI Test Symp., P.4-9,1993.

[3] P.Girard," survey of low-power testing of VLSI circuits," IEEE design and test of computers, Vol. 19,no.3,PP 80-90,May-June 2002.

[4] Mechrdad Nourani,"Low-transition test pattern generation for BIST- Based Applications", IEEE TRANSACTIONS ON COMPUTERS, Vol57,No.3 ,March 2008.

[5] BOYE and Tian-Wang Li," A novel BIST scheme for low power testing," 2010 IEEE.

[6] R.S.Katti,X.Y.Ruan , and H.Khattri," Multiple-Output Low-Power Linear feedback shift register design," IEEETrans.circuitsSyst.I,Vol.53,No.7,pp-1487-1495,July 2006.

[7] P.Girard, L.Guiller, C.Landrault, S.Pravossoudovitch,and H.J.Wunderlich," A modified clock scheme for a low power BIST test pattern generator," 19th IEEE proc. VLSI test Symp.,CA,pp-306- 311,Apr-May 2001.

[8] S.Wang and S.K.Gupta," DS-LFSR: a BIST LFSR for low switching activity," IEEE Trans.computer-aided design of Integrated circuits and systems, Vol. 21,No.7,pp.842-851,July 2002.

[9] I.Voyiatzis,A.paschalis,D.Nikolos and C.Halatsis, "An efficient built-in self test method for robust path delay fault testing," Journal of electronic testing: Theory and applications Vol.8,No.2,pp-219-222,Apr-
1996.

[10] S.C.Lei, J.Guo, L.Cao, Z.Ye.Liu, and X.M.Wang,"SACSR: A low power BIST method for sequential circuits,: Academic Journal of XI'AN jiaotonguniversity(EnglishEdition),Vol.20,no.3,pp.155- 159,2008.

[11] R.H.He,X.W.Li and Y.Z.Gong," A scheme for low power BIST test pattern generator," micro electronics & computer,no.2,pp.36-39 Feb.2003.

[12] S.C. Lei, X.Y.Hou ,Z.B.Shao and F.Liang," A class of SIC circuits: theory and application in BIST design," IEEE trans. circuits syst. II,