A HIGH PERFORMANCE PARALLEL ARCHITECTURE FOR LINEAR FEEDBACK SHIFT REGISTER

Dhinesh. H[1], Adhiyaman. G[2], Esakki Muthu. A[3]

[1,2,3]*Student, Department of Electronics and Communication, Adhiparasakthi Engineering College, Melmaruvathur, Tamilnadu, India*

---***---

Abstract - Low power dissipation is very critical in today's electronic designs. Linear feedback shift registers are an efficient way of describing and generating certain sequences in hardware implementations. A linear feedback shift register composed of a shift register R which contains a sequence of bits and a feedback function F which is the bit sum (XOR) of the entries of the shift register. The performance on parallel architecture of LFSR reduces dynamic power consumption significantly, compared to the conventional architecture and showed the way to generate multiple outputs. The proposed method significantly reduces dynamic power dissipation, simplifies the design process for single and multiple output generation, and eliminate the need of some hardware. The achievable rate and power reduction to improve the performance in parallel architecture by implementing Integrated clock gating. The Integrated clock gating achieves substantial reduction on the power consumption by reducing the gate count and dynamic power dissipation

Key Words: High performance, Linear feeback shift register. Clock gating, Parallel architecture.

1. INTRODUCTION

The main challenging areas in VLSI are performance, cost, testing, area, reliability and power. The applications require low power dissipation for circuit implementation. Linear feedback shift registers (LFSR's) are an efficient way of describing and generating certain sequences in hardware implementations. A linear feedback shift register is composed of a shift register R which contains a sequence of bits and a feedback function f which is the bit sum (XOR) of a subset of the entries of the shift register.

A design must contain these enable conditions in order to use and benefit from clock gating. This clock gating process can also save significant die area as well as power, since it removes large numbers of mux and replaces them with clock gating logic. This clock gating logic is generally in the form of "integrated clock gating" (ICG) cells. However, the clock gating logic will change the clock tree structure, since the clock gating logic will sit in the clock tree

1.1 Linear Feedback Shift Register

In computing, a linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state. The most commonly used linear function of single bits is exclusive-or (XOR). Thus, an LFSR is most often a shift register whose input bit is driven by the XOR of some bits of the overall shift register value.

The initial value of the LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an LFSR with a well-chosen feedback function can produce a sequence of bits that appears random and has a very long cycle.

Applications of LFSRs include generating pseudo-random numbers, pseudo-noise sequences, fast digital counters, and whitening sequences. Both hardware and software implementations of LFSRs are common. The mathematics of a cyclic redundancy check, used to provide a quick check against transmission errors, are closely related to those of an LFSR.

1.2 Objective

- To implement a parallel architecture with high performance of a linear feedback shift register.
- To reduce gate count and dynamic power consumption.
- The proposed design achieves substantial reduction on power consumption

2. PROPOSED METHODOLOGY

2.1 MULTIPLE CLOCK GATING

- To reduce the dynamic power dissipation, Multiple clock gating is used.
- Clock gating process can also save significant die area as well as power, since it removes large number of multiplexers and replaces them with clock gating logic.
- Using clock gating, it can save power by reducing unnecessary clock activities inside the gated module.

2.1.1 PARALLEL LFSR STRUCTURE

A parallel linear-feedback shift register (LFSR) is a shift register whose input bit is a linear function of its previous state. The most commonly used linear function of single bits is exclusive-or (XOR). Thus, an LFSR is most often a shift register whose input bit is driven by the XOR of some bits of the overall shift register value.

The initial value of the p-LFSR is called the seed, and because the operation of the register is deterministic, the stream of values produced by the register is completely determined by its current (or previous) state. Likewise, because the register has a finite number of possible states, it must eventually enter a repeating cycle. However, an p-LFSR with a well-chosen feedback function can produce a sequence of bits that appears random and has a very long cycle.

2.1.2 CLOCK GATING

Clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation. Clock gating saves power by adding more logic to a circuit to prune the clock tree. Pruning the clock disables portions of the circuitry so that the flip-flops in them do not have to switch states. When not being switched, the switching power consumption goes to zero, and only leakage currents are incurred Clock gating works by taking the enable conditions attached to registers, and uses them to gate the clocks. A design must contain these enable conditions in order to use and benefit from clock gating. This clock gating process can also save significant die area as well as power, since it removes large numbers of mux and replaces them with clock gating logic. This clock gating logic is generally in the form of "integrated clock gating" (ICG) cells. However, the clock gating logic will change the clock tree structure, since the clock gating logic will sit in the clock tree.

Clock gating logic can be added into a design in a variety of ways:

1. Coded into the register transfer level (RTL) code as enable conditions that can be automatically translated into clock gating logic by synthesis tools (fine grain clock gating).
2. Inserted into the design manually by the RTL designers (typically as module level clock gating) by instantiating library specific integrated clock gating (ICG) cells to gate the clocks of specific modules or registers.
3. Semi-automatically inserted into the RTL by automated clock gating tools. These tools either insert ICG cells into the RTL or add enable conditions into the RTL code. These typically also offer sequential clock gating optimisations.

Any RTL modifications to improve clock gating will result in functional changes to the design (since the registers will now hold different values) which need to be verified.

Sequential clock gating is the process of extracting/propagating the enable conditions to the upstream/downstream sequential elements, so that additional registers can be clock gated.

Although asynchronous circuits by definition do not have a "clock", the term perfect clock gating is used to illustrate how various clock gating techniques are simply approximations of the data-dependent behaviour exhibited by asynchronous circuitry.

As the granularity on which you gate the clock of a synchronous circuit approaches zero, the power consumption of that circuit approaches that of an asynchronous circuit: the circuit only generates logic transitions when it is actively computing.

Chip intended to run on batteries or with very low power such as those used in the mobile phones, wearable devices, etc. would implement several forms of clock gating together.

At one end is the manual gating of clocks by software, where a driver enables or disables the various clocks used by a given idle controller. On the other end is automatic clock gating, where the hardware can be told to detect whether there's any work to do, and turn off a given clock if it is not needed. These forms interact with each other and may be part of the same enable tree. For example, an internal bridge or bus might use automatic gating so that it is gated off until the CPU or a DMA engine needs to use it, while several of the peripherals on that bus might be permanently gated off if they are unused on that board.

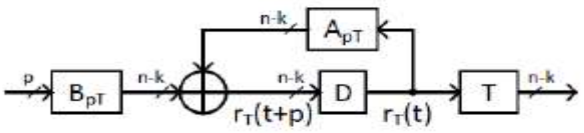

Figure 2.1 Transformed LFSR parallel architecture

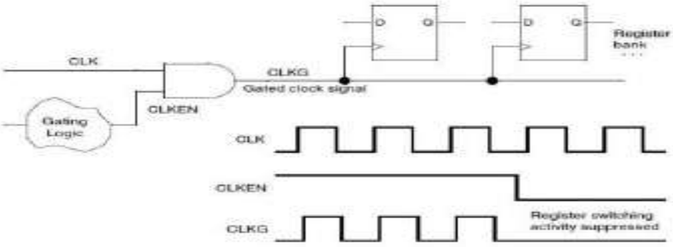

Figure 2.2 Clock gating

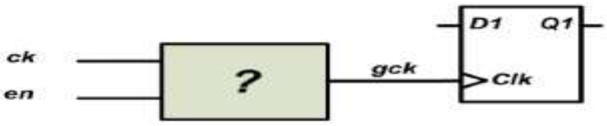

Figure 2.3 Reference Block of Clock gating

2.2 OPERATIONS OF CLOCK GATING

The DFF shown in Figure 2.3, has main clock *ck* and it has an enable signal, en, which determines when the DFF should operate (in the register file example, the write enable signal, wr_en, can be used for clock gating purposes). When en is logic high, the gated clock gck, will be equal to *ck*. This is the basic concept of clock gating.

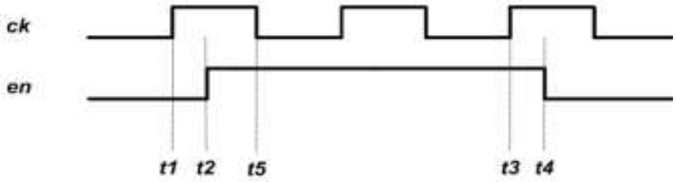

Figure 2.4 Reference waveform of Clock gating

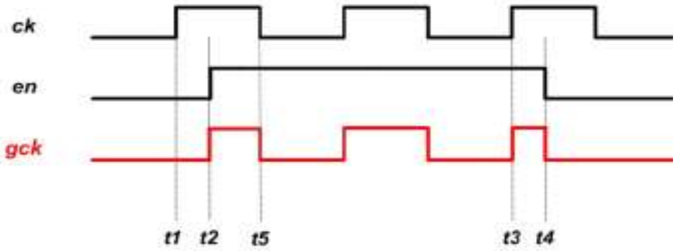

Figure 2.5 sequence pulsed waveform of Clock gating

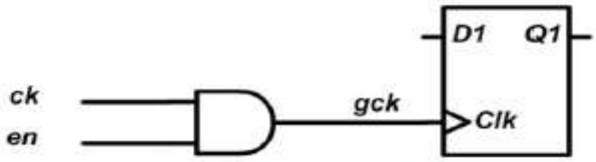

Figure 2.6 Logical gate in Clock gating

Since the DFF shown in Figure 2.4 is sensitive to the positive edge of the clock, if the *en* signal too comes from devices that change state at the rising edge of the clock. In Figure 6.4, the clock signal, *ck*, goes from low to high at *t=t1*; some time later, the *en* signal transitions to high at *t=t2*. The time difference *t2-t1* corresponds to the delay of the circuitry that produces the *en* signal. In this case, *t2-t1*will correspond to the delay of the FFs that store this particular state of the FSM plus the delay of the combinational circuit that generates the *en* signal from the Finite State Machine state(FSM). Hence, the transitions of en will occur some time after the rising edge of ck.

Let's use the above example waveforms to find a circuit that can generate an appropriate gated clock, gck, for Figure 2.3. From t2 to t4, the en signal is high and gck must be equal to ck. What if en is logic low? Should gck be high or low in this case? First, we assume that, for en=0, gck is set to low. Then, the red waveform shown in Figure 2.5 is obatined. To generate this waveform, replace the unknown circuit of Figure 2.3 with an AND gate as shown in Figure 6.6.

There are a few problems with this clock gating. First, do we really need a rising edge for gck at *t=t2*? One may think that the rising edge of gck at *t=t2* is the delayed version of the rising edge of *ck* at *t=t1*; however, note that, at *t=t1*, we have *en=0* and, hence, the clock edge of *ck* must not reach the FF. Thus, this gating arrangement produces an undesired rising-edge transition.

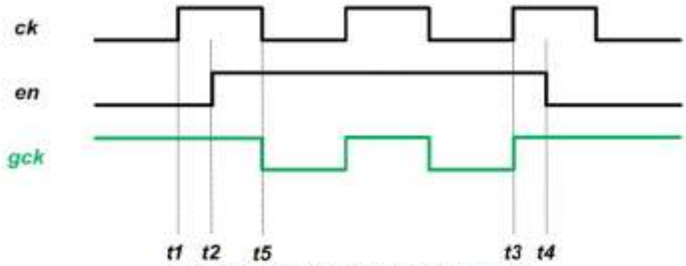

Figure 2.7 Modified pulsed waveform of Clock gating

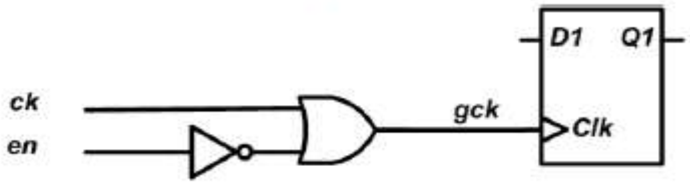

Figure 2.8 logical circuit of Clock gating

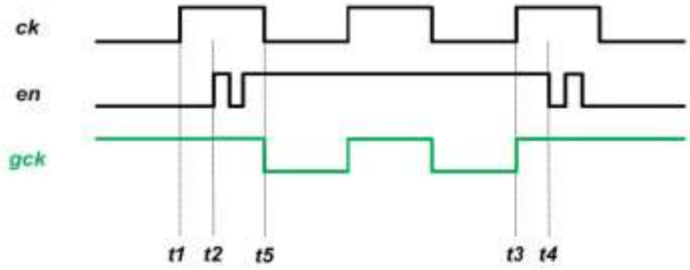

Figure 2.9 Resultant waveform of Clock gating

Another issue is that the pulse width from *t2* to *t5* is shorter than that of *ck*. A very short pulse can cause the DFF to malfunction.

When *en* is logic high, the gated clock, gck, will be equal to *clk* but, for en=0, gck will be set to high. Then, the green waveform is obtained as shown in Figure 2.7. To generate this waveform, the logical circuit is used as shown in Figure 2.8

The gck generated in Figure 2.7 doesn't have the rising edge at *t=t2*. With the circuit of Figure 6.8, a rising edge is presented to the DFF only when *en* is high and a rising edge occurs on ck. Moreover, the generated pulse width cannot be shorter than that of the original clock.

The synchronous system triggers changes at the rising edge of the clock. Also, the logic that generates the *en* signal has a delay that is less than half the period of *ck*. This means that *en* will start changing state right after the rising edge of ck (at *t=t1*) and will reach its final value before the next falling edge of ck, i.e., before *t=t5*. However, before reaching its final value, the en signal can have glitches.

2.2.1 CLOCK GATING IN FPGA

Clock gating is a common technique used to reduce power consumption in the context of application-specific integrated circuit (ASIC) design. However, in FPGAs, we normally avoid gating the clock. This is mainly due to the fact that, in an FPGA, dedicated nets and buffers are utilized to appropriately route the clock signal to different parts of the chip.

Clock gating can interfere with the clock distribution network, for example, by forcing the clock signal to go through a general-purpose lookup table. If clock gating is utilized in FPGAs, it's up to the designer to check that the synthesized circuits are safe.

However, power optimization software packages can be used to apply the concepts of clock gating in order to reduce the power consumption of the circuit. For example, Xilinx has an option called "Intelligent Clock Gating" which uses the clock enable pin in a slice to neutralize superfluous switching activity. The technique is different from the classic clock gating discussed in this article because Intelligent Clock Gating doesn't actually create new clocks. Instead, Xilinx's technique uses clock enable pins of slices to disable registers that don't contribute to the circuit's operation for a given clock cycle.

2.2.2 SUMMARY OF CLOCK GATING

- Classic clock gating can significantly reduce power consumption. This can be done, for example, by switching off the clock signal for DFFs that don't change state.

- For a synchronous system in which the logic is driven by the rising edge of the clock, use an OR gate to generate the gated clock. In this case, it has an correct timing along with resilience to glitches.

- For a synchronous system in which the logic is driven by the falling edge of the clock, use an AND gate to generate the gated clock.

- In FPGAs, classic clock gating is discouraged because it can lead to unexpected or undesired functionality.

3. RESULT AND DISCUSSION

3.1 RTL SCHEMATIC OF LFSR

Figure 3. 1 describes the schematic of linear feedback shift register designed in the Xilinx suite. LFSR is an shift register that takes a linear function of previous state as an input.

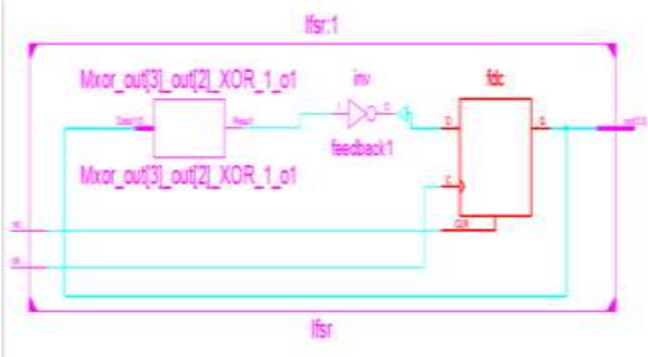

Figure 3. 1 Schematic Diagram of Linear Feedback Shift Register

3.2 OUTPUT OF THE LFSR

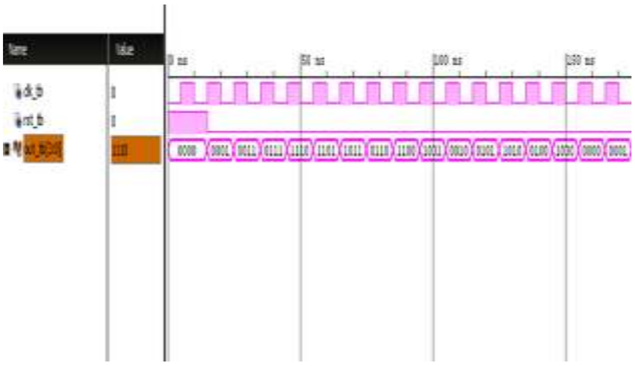

Figure3. 2

3.3 RTL SCHEMATIC FOR PARALLEL LFSR WITH CLOCK GATING

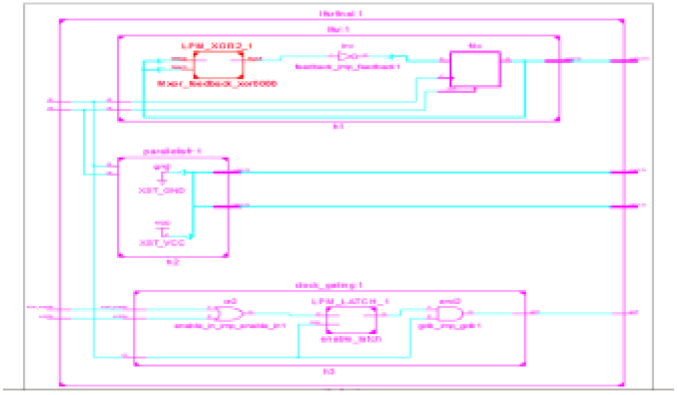

Figure 3.3 Schematic for Parallel Linear Feedback shift register

In figure 3.3, schematic diagram of parallel linear feedback shift register designed in the Xilinx suite. It consumes more power than linear feedback shift register. When clock gating is given as the input to parallel LFSR the consumption of clock pulse gets reduced using Control logic contains XOR and NOR gate.

3.4 OUTPUT OF THE PARALLEL LFSR WITH CLOCK GATING

In figure 3.4, the output waveform of parallel linear feedback shift register obtained in Model Sim 6.4a. The clock gated input is given to the parallel LFSR which reduces the unwanted clock supply given to shift registers. By the help of clock gating input, power dissipation occurred in flipflop gets reduced in such a way that the performance of the circuit in increased.

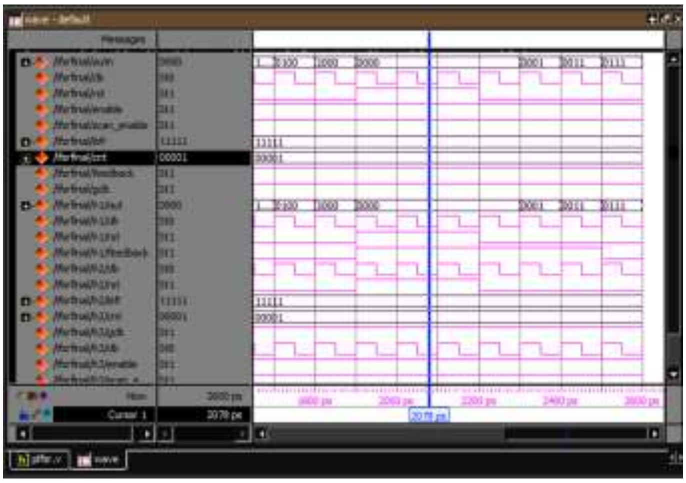

Figure3. 4 Output Waveform for Parallel Linear Feedback shift register

3.5 SCHEMATIC OF 4 STAGE PARALLEL LFSR WITH INTEGRATED CLOCK GATING

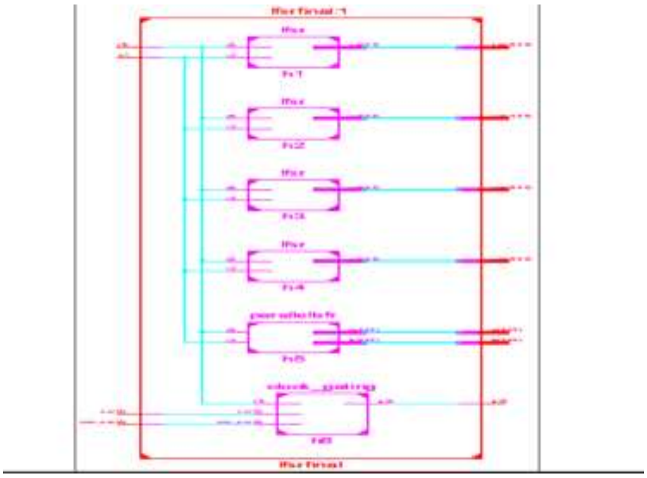

Figure 3.5 Schematic for 4 stage Parallel Linear Feedback shift register

In figure 3.5, schematic diagram of 4 Stage parallel linear feedback shift register designed in the Xilinx suite. In this schematic the Integrated clock gating is given as the input to parallel LFSR. So, the consumption of clock pulse gets reduced using Control logic. The common clock is given by dividing the clock pulse as input with the help of Integrated Clock Gating (IGC) to the 4 Stage parallel LFSR.

3.6 OUTPUT OF THE 4 STAGE PARALLEL LFSR WITH INTEGRATED CLOCK GATING

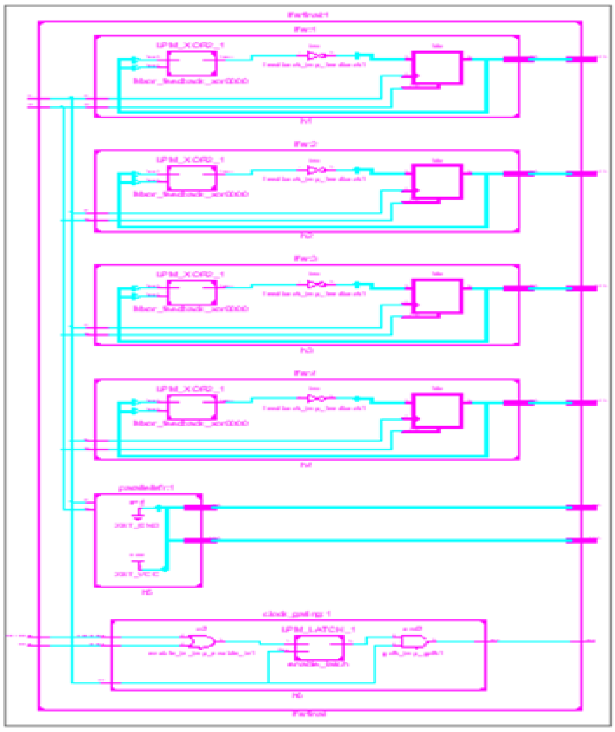

Figure 3.6 RTL Schematic Diagram for 4 stage Parallel LFSR

In figure 3. 6, RTL schematic diagram of 4 Stage parallel linear feedback shift register designed in the Xilinx suite. In this schematic the Integrated clock gating is given as the input to parallel LFSR. So, the consumption of clock pulse gets reduced using Control logic. The common clock is given by dividing the clock pulse as input with the help of Integrated Clock Gating (IGC) to the 4 Stage parallel LFSR. By implementing the 4stage parallel Linear Feedback Shift register the performance is increased by reducing dynamic power consumption and delay using clock gating.

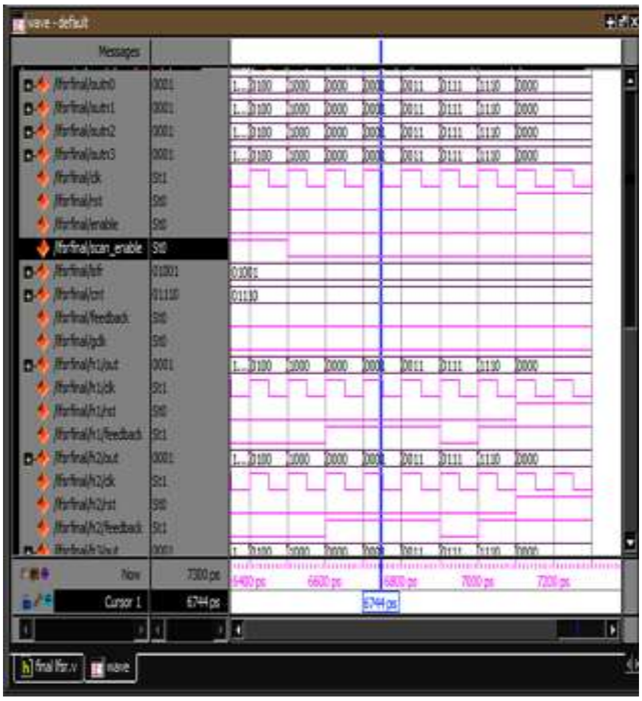

Figure 3. 7 Output Waveform for 4 Stage Parallel LFSR

In figure 3.7, the final output waveform of 4 stage parallel linear feedback shift register obtained in Model Sim 6.4a. The output of LFSR is given as 4bit in which the corresponding output is obtained by shifting the bit in parallel stage and clock pulse is consumed with the help of clock gating. By the help of clock gating input, power dissipation occurred in flipflop gets reduced in such a way that the performance of the circuit in increased. By giving common clock input with the help of Integrated Clock Gating (IGC) the performance is improved by reducing power consumption.

4. CONCLUSION

The shift register reduces area, delay and power consumption by replacing clock signal with clock gating signal. The Clock gating reduces delay, area and power consumption which is implemented in Linear Feedback Shift Register. The power dissipation problem between flipflops in shift registers is solved using Integrated Clock Gating in parallel architecture of linear feedback shift register. This technique can significantly reduce power consumption. This can be done, by switching off the clock signal for D Flip Flop's that don't change state. Clock gating process can also save significant die area as well as power, since it removes large numbers of mux and replaces them with clock gating logic. This clock gating logic is generally in the form of "Integrated clock gating" (ICG) cells. This project shows a high performance parallel LFSR architecture that effectively shift the complexities of power dissipation with the help of an Integrated clock gating. As a result, the proposed design achieves substantial reduction on power consumption without increasing the critical path delay. The Future work will address the efficient parallel design of long LFSRs.

REFERENCES

[1] M. Ayinala and K. K. Parhi, "High-speed parallel architectures for linear feedback shift registers," IEEE Trans. on Signal Process., vol. 59, no. 9, pp. 4459-4469, Sep. 2011

[2] N.Chen and Z. Yan, "High-performance designs of AES transformations," Proc. IEEE Intl. Symp. Circuits and Syst., pp. 2906-2909, May 2009.

[3] C. Cheng and K. K. Parhi, "High-speed parallel CRC implementation based on unfolding, pipelining, and retiming," IEEE Trans. on Circuits and Syst.-II, vol. 53, no. 10, pp. 1017 - 1021, Oct. 2006.

[4] J.H.Derby, "High-speed parallel CRC implementation on state space transformation" Proc. IEEE Global Commun. Conf., pp. 166-170, Nov. 2001.

[5] J. Jung, et. al., "Efficient parallel architecture for linear feedback shift registers," IEEE Trans. on Circuits and Syst.-II, vol. 62, no. 11, pp. 1068- 1072, Nov. 2015.

[6] C. Kennedy and A. Reyhani-Masoleh, "High-speed CRC computations using improved state-space transformation," Proc. IEEE Intl. Conf. Electro/ Info. Tech., pp. 9-14, 2009.

[7] T.B. Pei, and C. Zukowski, "High-speed parallel CRC circuits in VLSI," IEEE Trans. on Commun., vol. 40, no. 4, pp. 653-657, Apr. 1992.

[8] G. Hu, J. Sha, and Z. Wang, "High-speed parallel LFSR architectures based on improved state-space transformations," IEEE Trans. on VLSI Syst. vol. 25, no. 3, pp. 1159-1163, Mar. 2017.

[9] N. Wu, et. al., "Improving common subexpression elimination algorithm

with a new gate-level delay computing method," Proc. World Congress

on Engr. and Computer Science, Oct. 2013.

[10] X. Zhang and K. K. Parhi, "High-speed architectures for parallel long BCH

encoders," IEEE Trans. on VLSI Syst., vol. 13, no. 7, pp. 872-877, Jul. 2005.