# 오류 검출 선형 되먹임 시프트 레지스터

신화수, 최소연, 김현규, 류도현, 유호영
충남대학교
전화: (042)821-6285, E-mail: hyyoo@cnu.ac.kr

# Fault Tolerant Linear Feedback Shift Register

Hwasoo Shin, Soyeon Choi, Hyeonkyu Kim, Dohyun Ryu, and Hoyoung Yoo
Chungnam National University

요 약

선형 되먹임 시프트 레지스터는 다항식에 따라 입력되는 값들이 이전 상태의 선형 함수로 계산되어 정해진 패턴을 주기적으로 반복하며 출력하는 순차회로의 일종이다. 무작위 패턴 생성 특성을 활용하여 내장자체시험과 같은 테스팅이나 스트림 암호를 생성하는 용도로 많이 이용된다. 본 논문에서는 선형 되먹임 시프트 레지스터의 현재 패턴을 통해 다음 패턴의 패리티를 예측함으로써 열악한 외부환경에 의해 발생할 수 있는 오류를 검출하고, 정상 패턴으로 정정한다. 오류가 발생하면 이전패턴으로 복귀시킴으로써 신뢰성을 높이고 시간적 낭비를 최소화한다.

Abstract

Linear Feedback Shift Register (LFSR) is a special type of sequential logics that generate a new state based on a generator polynomial. It is widely used to generate a pseudo random sequences for BIST (Built-In Self Test) and stream cipher systems. In this paper, we propose a Fault Tolerant Linear Feedback Shift Register (FT-LFSR), which detects and collects an error by comparing an actual parity and a predicted parity. When an error occurs, the proposed FT-LFSR returns to the previous pattern so as to improve reliability and to reduce a waste of time.

**Keywords**: 선형 되먹임 시프트 레지스터, 오류 검출, 오류 정정, 고장 허용시스템, 오류 감지 시스템

## Ⅰ. 서론

선형 되먹임 시프트 레지스터(Linear Feedback Shift Register, LFSR)[1]은 순차회로의 일종으로, 다항식에 따라 입력되는 이전 상태 값들의 선형 함수의 계산을 통해 예측 가능한 난수를 발생시켜 스트림 암호생성이나 내장자체시험(Built-In Self Test, BIST) 등의 용도로 많이 이용된다[2]. 그림 1 은 $x^4+x^3+1$ 의 다항식을 갖는 4 비트 선형 되먹임 시프트 레지스터의 구조이다[1]. LFSR 은 다항식에 따라 정해진 패턴을 주기적으로 반복하며 출력으로 내보낸다. 열악한 외부 환경에서 LFSR 은 플립플롭 내부 혹은 플립플롭 간의 데이터 전송 과정에서 오류가 발생할 가능성이 존재한다. 이러한 오류의 발생이 암호화나 테스트 결과의 신뢰성을 낮추고 시간적 손해를 입힌다. 따라서 오류의 발생을 미리 감지하여 정정할 수 있는 오류 검출 기능이 필요하다.

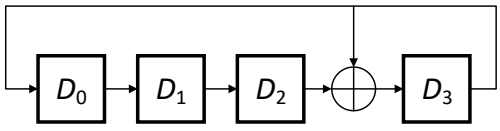


그림 1. 전통적인 4 비트 선형 되먹임 시프트 레지스터

## Ⅱ. 본론

본 논문에서는 선형 되먹임 시프트 레지스터의 오류를 검출하기 위해 패리티 부호를 적용한다[3]. 우선 효율적인 패리티 부호 적용을 위하여 그림 1 과 같이 전통적인 선형 되먹임 시프트 레지스터의 패리티를 분석하였다. 표 1 에 의하면 최상위 비트인 $D_3$ 가 1 이면 레지스터의 패리티가 반전되고, 최상위 비트인 $D_3$ 가 0 이면 패리티가 유지되는 특성을 확인할 수 있다.

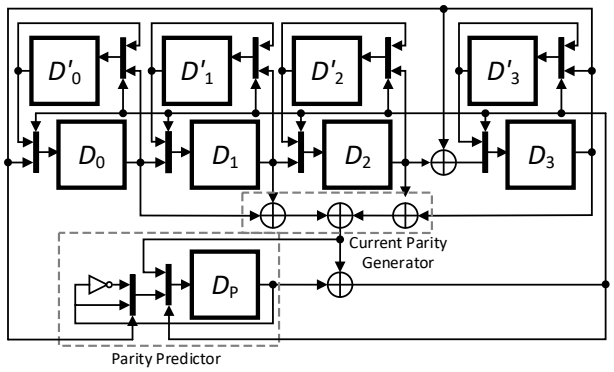


그림 2. 4 비트 오류 검출 선형 되먹임 시프트 레지스터

이러한 선형 되먹임 시프트 레지스터의 특성을 활용하여 오류 검출 선형 되먹임 시프트 레지스터(Fault Tolerant Linear Feedback Shift Register, FT-LFSR)의 구조를 그림 2 와 같이 제안한다. 기존의 선형 되먹임 시프트 레지스터에 1) 패리티 생성부, 2) 패리티 예측부, 3) 레지스터 상태 저장부를 추가하여 구현하였다. 패리티 생성부는 현재 레지스터의 패리티를 XOR 하여 현재 패리티를 생성하고, 패리티 예측부는 앞서 설명한 패리티 획득 특성에 따라 패리티를 예측한다. 다시 말하면 $D_3$ 의 출력에 따라 이전 패리티 $D_p$ 를 반전하거나 유지한다. 끝으로 레지스터 상태 저장 부는 오류정정을 위하여 레지스터의 값을 저장하는 용도로 사용된다.

패리티 생성부와 패리티 예측부의 패리티가 일치하면 오류가 없는 정상적인 패턴으로 판단하여 시프트 레지스터는 정상 동작을 한다. 추가적으로 레지스터 상태 저장부 $D'_{0-3}$ 에 현재 시프트 레지스터의 출력과 현재 패리티 발생기의 출력을 저장하여, 오류가 발생할 경우 이를 활용할 수 있도록 대비한다. 패리티 생성부와 패리티 예측부의 패리티가 일치하지 않으면 오류가 발생한 것으로 판단하여 레지스터 상태 저장부 $D'_{0-3}$ 에 저장해둔 값으로 복귀하여 동작을 다시 진행한다. 이를 통해 지정된 초기 값으로 복귀하는 시스템과는 달리 직전 값으로 복귀함으로 시간 소모를 최소화 할 수 있다.

## Ⅲ. 실험결과 및 결론

본 논문에서는 LFSR 과 FT-LFSR 을 4 비트, 7 비트, 15 비트에 대하여 CMOS 180 nm 공정을 사용하여 200 MHz 로 합성했다. 표 2 는 각 LFSR 과 FT-LFSR 의 합성 결과를 나타낸다. 실험결과에 따르면 Equivalent Gate Count 는 오류 검출 기능을 적용한 경우가 적용하지 않은 LFSR 에 비해 약 70% 증가했고, Critical Path Delay 또한 평균적으로 약 70% 증가한다.

선형 되먹임 시프트 레지스터의 플립플롭 내부 혹은 플립플롭 간의 데이터 전송과정에서 발생하는 오류에 대처하기 위한 오류 검출 선형 되먹임 시프트 레지스터를 제안하였다. 제안하는 오류 검출 선형 되먹임 시프트 레지스터는 오류를 검출하여 패턴을 안정시킬 수 있어 시스템의 신뢰성을 높이고 시간적 이득을 얻을 수 있다.

표 1. 4 비트 선형 되먹임 시프트 레지스터의 출력 패턴과 패리티

| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $P$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 0 | 1 |

| $D_0$ | $D_1$ | $D_2$ | $D_3$ | $P$ |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |

표 2. 전통적인 LFSR 과 FT-LFSR 의 합성결과 비교

| LFSR | | Equivalent Gate Count | Critical Path Delay |
|---|---|---|---|
| 4-bit | Conventional | 63.41 | 1.85 |
| | Fault Tolerant | 110.47 | 3.27 |
| 7-bit | Conventional | 108.47 | 2.07 |
| | Fault Tolerant | 180.58 | 3.70 |
| 15-bit | Conventional | 228.63 | 2.59 |
| | Fault Tolerant | 379.51 | 4.30 |

## ACKNOWLEDGEMENTS

이 논문은 중소기업청이 지원하는 중소기업청 지원사업(S2646718)과 IDEC 의 지원으로 수행된 연구임.

## 참고문헌

[1] A. Poorghanad, A. Sadr, A. Kashanipour, "Generating High Quality Pseudo Random Number Using Evolutionary methods", *IEEE International Conference on Computational Intelligence and Security*, Vol. 1, pp. 331-335, Dec. 2008

[2] E. J. McCluskey, "Built-In Self-Test Techniques", *IEEE Design & Test of Computers*, Vol. 2, pp. 21-28, April.1985

[3] M. Nicolaidis, "Carry checking/parity prediction adders and ALUs", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol.11, pp. 121-128, April.2003