ข้อเสนอโครงร่างวิทยานิพนธ์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

****

ชื่อโครงการ

Mixed-Mode BIST ที่ประหยัดทรัพยากรสำหรับทดสอบสมองกลฝังตัว
บนอุปกรณ์ลอจิกแบบโปรแกรมได้

Low Cost Mixed-Mode BIST for Testing Embedded FPGA Core

ผู้ทำการวิจัย

ชื่อ-สกุล นางสาวปิยนาฏ คงทิม รหัสนักศึกษา 5110120025
สาขาวิชาวิศวกรรมคอมพิวเตอร์ หลักสูตร วิศวกรรมศาสตรมหาบัณฑิต
คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

อาจารย์ที่ปรึกษาวิทยานิพนธ์

ดร. วรรณรัช สันติอมรทัต อาจารย์ที่ปรึกษาหลัก

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยสงขลานครินทร์

1. ความสำคัญและที่มาของหัวข้อวิจัย

ปัจจุบันเทคโนโลยีด้านอุปกรณ์อิเล็กทรอนิกส์ได้พัฒนาไปอยู่ในยุคของนาโนเทคโนโลยี ทั้งทางด้านประสิทธิภาพ ความสามารถการทำงานและขนาดของอุปกรณ์ โดยมีการออกแบบให้ในชิปหนึ่งชิ้นประกอบไปด้วยวงจรอิเล็กทรอนิกส์มากมาย เช่น ไมโครโปรเซสเซอร์, วงจรอนาลอก, หน่วยความจำ เป็นต้น ซึ่งชิปอิเล็กทรอนิกส์เหล่านี้มีขนาดเล็ก ต้องการประสิทธิภาพสูง ทั้งทางด้านความเร็ว ด้านกำลังไฟฟ้าและด้านสมรรถนะ อย่างไรก็ดีขนาดของชิปที่เล็กจะกำหนดขนาดของส่วนประกอบภายในด้วย ซึ่งส่งผลให้อัตราการผลิตต่อหนึ่งหน่วยยูนิค (Yield) ลดลง เนื่องจากเทคโนโลยีสมัยใหม่ ที่เน้นการผลิตชิปให้มีคุณสมบัติข้างต้น ข้อผิดพลาดของขบวนการผลิตนอกจากจะมีโอกาสเกิดขึ้นมากแล้ว ยังมีชนิดข้อผิดพลาดใหม่ๆเกิดขึ้นตามมาอีกด้วย ซึ่งหากชิปที่ถูกผลิตมีข้อผิดพลาด อาจทำให้การทำงานไม่ถูกต้องหรือไม่ทำงาน โดยผู้ผลิตอาจได้รับความเสียหายมาก จากการถูกส่งคืนของผู้รับซื้อได้

การทดสอบ (Testing) เป็นกระบวนการตรวจสอบการทำงานของชิปก่อนที่จะเข้าสู่กระบวนการผลิต และช่วยลดระยะเวลากระบวนการออกแบบและผลิตสู่ท้องตลาด โดยจำลองการทำงานของชิป ตรวจจับข้อผิดพลาดและวิเคราะห์ผลการทำงาน วิธีการหนึ่งที่สามารถทำได้ก็คือการประยุกต์ใช้อุปกรณ์ลอจิกแบบโปรแกรมได้ แต่การใช้อุปกรณ์ชนิดนี้มีข้อจำกัดทางด้านหน่วยความจำและทรัพยากรที่มีให้ใช้ ดังนั้นการนำอุปกรณ์ลอจิกแบบโปรแกรมได้มาใช้สำหรับการทดสอบวงจรที่ถูกออกแบบจะต้องคำนึงถึงข้อมูลการทดสอบ วิธีการทดสอบ เพื่อใช้หน่วยความจำและทรัพยากรที่เหมาะสม

วิทยานิพนธ์นี้จึงมีเป้าหมายในการวิจัยเพื่อหาวิธีการทดสอบที่เหมาะสมสำหรับการทดสอบวงจรสมองกลฝังตัวบนอุปกรณ์ลอจิกแบบโปรแกรมได้ ให้มีความสามารถในการกำเนิดข้อมูลทดสอบ เพื่อครอบคลุมการทดสอบ (Test Coverage) ได้ 100 เปอร์เซ็นต์ ข้อมูลการทดสอบ (Test Pattern) เวลาในการทดสอบน้อยลง (Test time) อีกทั้งประหยัดทรัพยากรของอุปกรณ์ลอจิกแบบโปรแกรมได้

2. วัตถุประสงค์ของโครงการ

1.1 เพื่อประยุกต์ใช้อุปกรณ์ลอจิกแบบโปรแกรมได้สำหรับการทดสอบวงจรที่ออกแบบ

1.2 เพื่อพัฒนาวิธีการทดสอบโดยใช้ Mixed-Mode BIST

3. ประโยชน์ที่คาดว่าจะได้รับ

3.1. สามารถพัฒนาวิธีทดสอบที่สามารถครอบคลุมการทดสอบของวงจร 100 เปอร์เซ็นต์ ด้วยอุปกรณ์ลอจิกแบบโปรแกรมได้

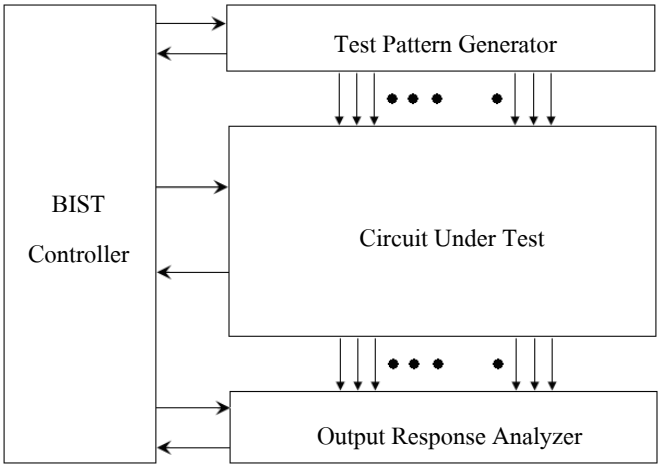
3.2. สามารถลดการใช้ข้อมูลการทดสอบ เวลาในการทดสอบ ทรัพยากรของอุปกรณ์ลอจิกแบบโปรแกรมได้ แต่คุณภาพการทดสอบไม่ลดลง

4. ทฤษฎีและหลักการ

4.1 Built-In Self-Test (BIST)

BIST เป็นรูปแบบพื้นฐานในการออกแบบวงจรให้สามารถทดสอบและวิเคราะห์ผลการทดสอบ เพื่อการทดสอบอุปกรณ์อิเล็กทรอนิกส์ที่มีขนาดเล็กในระดับนาโนเมตรให้สามารถทดสอบได้ด้วยตัวเอง อย่างมีประสิทธิภาพและไม่จำเป็นต้องใช้ซอฟแวร์ที่มีราคาแพง [1] โครงสร้างการทำงานของ BIST แสดงดังรูปที่ 1

BIST ประกอบได้ด้วย 3 ส่วน คือวงจรกำเนิดชุดข้อมูลการทดสอบ (Test Pattern Generator หรือ TPG) ส่วนที่สองคือวงจรทดสอบ (Circuit Under Test หรือ CUT) และวงจรวิเคราะห์ผลการทดสอบ (Output Response Analyzer หรือORA) วงจรควบคุมการทำงานของ BIST (BIST Controller) ทำหน้าที่ในการควบคุมและประสานงานกันระหว่างส่วนการกำเนิดข้อมูล ส่วนวงจรทดสอบ และส่วนวงจรวิเคราะห์ผลการทดสอบ



รูปที่ 1 โครงสร้างการออกแบบวงจรด้วยวิธีการทดสอบแบบ BIST [2]

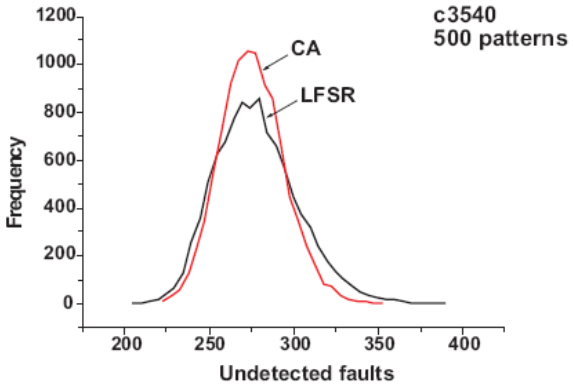
4.2 วงจรกำเนิดชุดข้อมูลการทดสอบ

วงจรกำเนิดชุดข้อมูลการทดสอบเป็นเทคนิคการกำเนิดข้อมูลทดสอบเพื่อตรวจจับความผิดพลาดของวงจรภายใต้การทดสอบ ซึ่งสามารถทดสอบได้ใกล้เคียง 100 เปอร์เซ็นของความผิดพลาดทั้งหมด ทั้งยังเป็นส่วนสำคัญในลดระยะเวลาการทดสอบและทรัพยากรที่ใช้ [1]

เทคนิคการกำเนิดข้อมูลทดสอบมีหลายเทคนิคด้วยกันแต่ที่นิยมมากที่สุดคือ Pseudo-Random Pattern Generator (PRPG) เนื่องจากวงจรมีขนาดเล็ก ใช้จำนวนหน่วยความจำน้อย

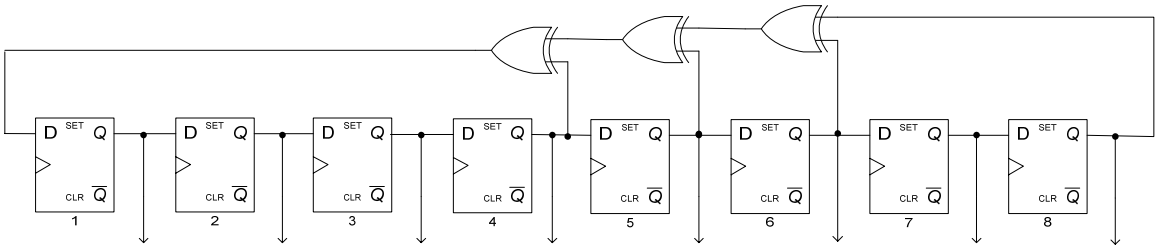
4.2.1 Pseudo Random Pattern Generator

เป็นเทคนิคการกำเนิดข้อมูลด้วยวิธีการสุ่มแต่สามารถคำนวณชุดข้อมูลชุดถัดไปที่จะเกิดขึ้นได้ โดยที่จำนวนของชุดข้อมูลที่ได้มีค่าเท่ากับ 2^n-1 การกำเนิดข้อมูลการทดสอบจากสมการโพลิโนเมียล (Polynomial) PRPG มีด้วยกัน 2 วิธีคือ Cellular Automata (CA) และ Linear Feedback Shift Register (LFSR) จากการทดลอง [3] โดยมีจำนวนชุดข้อมูลที่ใช้ทั้งหมด 500 ชุด ทดสอบด้วยวงจร ISCAS c3540 ผลการทดลองแสดงดังรูปที่ 2 จากรูปจุดสูงสุดของกราฟในแนวแกน X แสดงให้เห็นว่าจำนวนของความผิดพลาดที่ไม่สามารถตรวจพบได้ของ LFSR ค่าน้อยกว่า CA กล่าวคือ LFSR สามารถตรวจจับความผิดพลาดได้ดีกว่าวิธี CA จึงเป็นเหตุผลให้เลือกใช้เทคนิคกำเนิดข้อมูลทดสอบแบบ LFSR

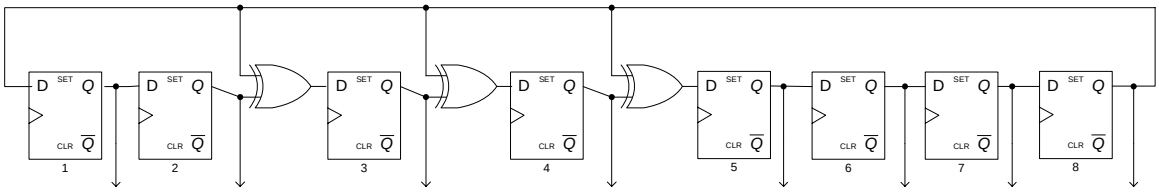


รูปที่ 2 จำนวนความผิดพลาดที่เกิดขึ้นในวงจรแล้วไม่สามารถตรวจพบได้
ระหว่าง CA และ LFSR ของวงจร c3540

4.2.1.1 LFSR เป็นเทคนิคกำเนิดข้อมูลทดสอบที่ใช้สมการโพลิโนเมียลในการเลื่อนบิตข้อมูลเป็นลำดับๆ มีอินพุทเป็นเชิงเส้น และมีเอาท์พุทที่เลื่อนตำแหน่งออกในแบบอนุกรมและแบบขนาน LFSR ประกอบไปด้วย D Flip-Flop และ Exclusive–OR (XOR) โดย XOR เกตเชื่อมต่อระหว่าง D Flip-Flop และชี้ตำแหน่งการป้อนกลับของข้อมูล เทคนิค LFSR มี 2 ชนิดคือ External LFSR หรือ Type I และ Internal LFSR หรือ Type II โดยวงจร Internal LFSR มี XOR อยู่ระหว่าง D Flip-Plop สองตัว แต่ External LFSR มีตำแหน่งของ XOR gate อยู่ภายนอก ซึ่งแสดงดังรูปที่ 3 และ 4 ตามลำดับ



รูปที่ 3 Internal LFSR หรือ Type II ที่มีสมการ $f(x) = X^8+X^4+X^3+X^2+1$



รูปที่ 4 External LFSR หรือ Type I ที่มีสมการ $f(x) = X^8+X^6+X^5+X^4+1$

สมการโพลิโนเมียลที่ใช้ในการสร้างวงจร LFSR แสดงดังสมการที่ (1)

$$f(x) = C_{n-1}X^{n-1} + C_{n-2}X^{n-2} + C_{n-3}X^{n-3} + \ldots + C_1X^1+1 \qquad (1)$$

จากสมการที่ 1 ค่า C คือค่าคงที่ของสมการ ค่าคงที่นี้จะใช้ในการบอกตำแหน่งป้อนกลับของเอาท์พุทและ n คือจำนวนเอาท์พุทของข้อมูลในรูปแบบขนานหรือจำนวนของ D Flip-Flop นั่นเอง เช่น n = 8 สามารถเขียนให้อยู่ใน

รูปของสมการ Polynomial ของ LFSR ในรูปแบบ Externalได้ $f(x) = X^8+X^6+ X^5+ X^4+1$ จากสมการเอาท์พุทของ D Flip-Flop ตำแหน่งที่ 8 ย้อนกลับและ XOR กับเอาท์พุทของ D Flip-Flop ตำแหน่งที่ 6, 5 และ 4 เพื่อเป็นอินพุทให้กับ D Flip-Flop ตำแหน่งที่ 1 ซึ่งแสดงดังรูปที่ 4

การอธิบายการป้อนกลับของเอาท์พุทของ D Flip-Flop สามารถเขียนในอยู่ในรูปของเซตของการเกิด XOR เกตซึ่งเรียกว่า tap ได้ tap = [8, 6, 5, 4] ซึ่งก็คือ มีการป้อนกลับเอาท์พุทจาก D Flip-Flop ตำแหน่งที่ 8, 6, 5 และ 4 การเขียนในลักษณะของเซตเพื่อให้ง่ายในการเข้าใจและลดรูปของสมการลง

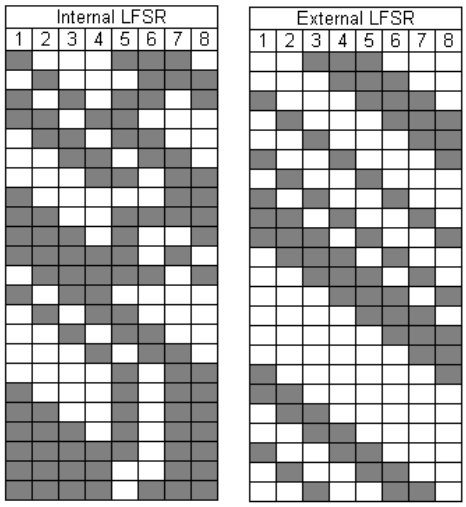
จำนวนข้อมูลทดสอบสูงสุดที่ LFSR สามารถกำเนิดได้เท่ากับ 2^n-1 และจะเรียกจำนวนนี้ว่า Maximum Sequence หรือ M-sequence ซึ่งเอาท์พุทที่ LFSR ไม่สามารถสร้างได้คือ ชุดข้อมูลที่ทุกตำแหน่งเป็น 0 เช่นถ้าจำนวน n เท่ากับ 8 ได้จำนวนการทดสอบ 2^8-1 = 255 สมการโพลิโนเมียลที่ใช้ในการสร้างวงจร LFSR ที่สามารถกำเนิดชุดข้อมูลได้สูงสุดเท่ากับจำนวน M-sequence จะเรียกสมการโพลิโนเมียลว่าเป็น Primitive Polynomial จากตัวอย่าง $f(x) = X^8+X^6+ X^5+ X^4+1$ เป็น Primitive Polynomial

การกำเนิดข้อมูลด้วยวิธี LFSR ทั้งสองรูปแบบมีข้อเสียที่จะเกิด Structural Dependency และ Correlate ของข้อมูลซึ่งมีรายละเอียดดังนี้

- Structural Dependency คือกลุ่มข้อมูลที่ได้จากการวงจรกำเนิดข้อมูลมีรูปแบบขึ้นอยู่กับโครงสร้างของเทคนิคที่ใช้ วงจรกำเนิดข้อมูล LFSR มีจำนวนของเอาท์พุทเท่ากับ 8 มีสมการโพลิโนเมียล $f(x) = X^8+X^6+ X^5+ X^4+1$ สำหรับ Internal LFSR และสำหรับ External LFSR มีสมการโพลิโนเมียล $f(x) = X^8+X^4+ X^3+ X^2+1$ ผลของการกำเนิดข้อมูลดังรูปที่ 5 โดยสีเข้มแทนด้วยบิตข้อมูลที่มีค่า ‘1’ และสีอ่อนแทนด้วยค่า ‘0’ ในตารางการกำเนิดข้อมูลด้วย External LFSR แสดงให้เห็นถึงข้อมูลที่มีลักษณะเกิดการเลื่อนตำแหน่งของข้อมูลไปเรื่อยๆ แต่เมื่อพิจารณา Internal LFSR จะมีการกระจายของกลุ่มข้อมูลมากขึ้นแต่ก็ยังเกิด Structural Dependency อยู่ Structural Dependency ที่เกิดขึ้นส่งผลให้ชุดข้อมูลทดสอบที่ถูกเลื่อนสู่วงจรทดสอบมีชุดข้อมูลใกล้เคียงกับข้อมูลก่อนหน้า ถ้าโครงสร้างของวงจรทดสอบแบบ STUMP จะทำให้เซลล์เพื่อนบ้านถูกทดสอบด้วยข้อมูลชุดเดิม [1] ซึ่งไม่ทำให้จำนวนความผิดพลาดที่ทดสอบได้เพิ่มขึ้น

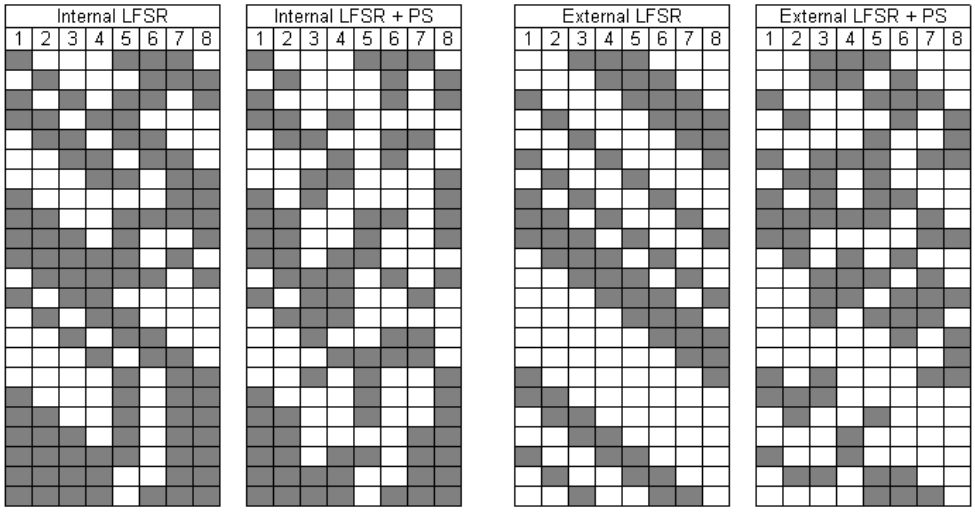
- Correlated คือลักษณะข้อมูลที่สร้างขึ้นแล้วมีลักษณะสัมพันธ์กับข้อมูลชุดอื่น เช่น ข้อมูลในข้อมูลทดสอบมีจำนวนข้อมูล 3 ชุด 110X0, 1X011 และ 11X01 ถ้าข้อมูลในชุดที่ 2 ตำแหน่งที่ 2 ถูกเปลี่ยนให้เป็น 1 และข้อมูลในชุดที่ 3 ตำแหน่งที่ 3 ถูกเปลี่ยนให้เป็น 0 ข้อมูลทั้ง 3 ชุดมีรูปแบบเป็น Correlated ในบิตที่ 1, 2 และ 3 ส่วนในบิตที่ 4 และ 5 ไม่เกิด Correlated

การแก้ไข Structural Dependency และ Correlated ของชุดทดสอบทำได้โดยการเพิ่มการกระจายตัวของข้อมูลทดสอบ ด้วยการใช้วงจรเลื่อนเฟส (Phase Shifter)



รูปที่ 5 การเปรียบเทียบการเกิด Structural Dependency ของ LFSR ทั้งสองแบบ

การเพิ่มวงจรเลื่อนเฟสเป็นวิธีการลดการเกิด Structural Dependency ทำให้ข้อมูลที่ได้จากวงจร Internal LFSR หรือ External LFSR มีการกระจายตัวของตำแหน่งของข้อมูลมากขึ้น จากรูปที่ 6 หมายเลข ‘1’ ที่ได้วงจร LFSR ทั้งสองแบบแสดงด้วยสีเข้ม และหมายเลข ‘0’ แสดงด้วยสีขาว เมื่อเปรียบเทียบชุดข้อมูลที่ได้จาก External LFSR และข้อมูลที่ได้จาก External LFSR รวมกับ Phase Shifter จะแสดงให้เห็นการกระจายตัวของหมายเลขหนึ่งและลดความเป็นรูปแบบของวงจรเลื่อนบิตลง



รูปที่ 6 การกระจายตัวของข้อมูลเมื่อใช้ LFSR ร่วมกับวงจรเลื่อนเฟส

4.2.2 วงจรเลื่อนเฟส

วงจรเลื่อนเฟสเป็นวงจรการเลื่อนลำดับของเลขฐานสองที่ถูกกำเนิดโดย LFSR ก่อนที่ข้อมูลจะถูกส่งไปยังวงจรภายใต้การทดสอบ การสร้างวงจรเลื่อนเฟสเกิดจากการนำ XOR เกตมาเชื่อมโยงกันระหว่างเอาท์พุทของ LFSR ในแต่ละบิตข้อมูล เพื่อให้ได้กลุ่มข้อมูลที่เกิดการเลื่อนลำดับเพื่อเพิ่มการกระจายตัวของข้อมูลและลดการเกิด Structural Dependency [4][5]

วิธีการสร้างวงจรเลื่อนเฟส [5] สามารถทำได้โดยการใช้ชุดข้อมูลที่กำเนิดโดยวิธีการ LFSR แล้วนำชุดข้อมูลที่ได้มาสร้างเป็นวงจรเลื่อนเฟสโดยมีสิ่งที่ต้องพิจารณาดังนี้

B คือจำนวนเอาท์พุทของวงจรเลื่อนเฟสที่ต้องการ

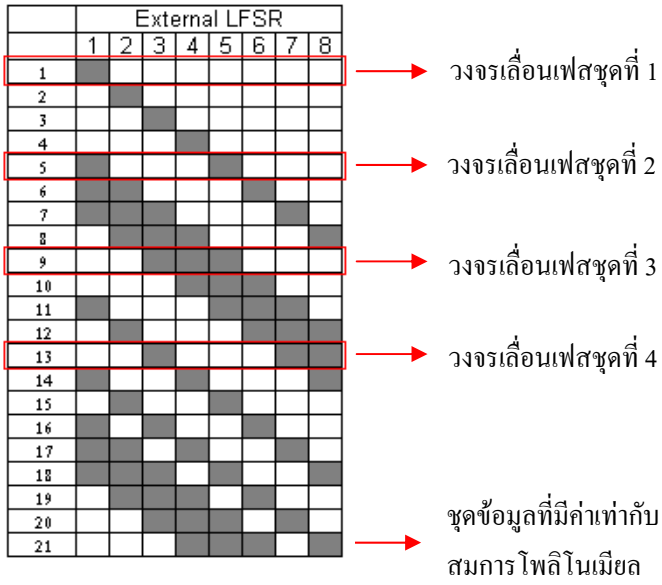
L คือจำนวนน้อยที่สุด ที่จะได้ชุดข้อมูลชุดถัดไป

I คือจำนวนหมายเลขหนึ่งที่ปรากฏในชุดข้อมูล

ซึ่งสามารถพิจารณาเป็นขั้นตอนได้ดังนี้

1) กำเนิดข้อมูลด้วยวิธี LFSR โดยการเริ่มข้อมูลชุดแรกโดยกำหนดให้ค่าเริ่มต้นเป็นค่าที่มี 1 เพียงตำแหน่งเดียว เช่น 00000001 ที่จำนวน D Flip-Flop เท่ากับ 8
2) กำหนดค่า L, B และ I ที่จะใช้ในการพิจารณา
3) กำเนิดข้อมูลจนกว่าจะได้ชุดข้อมูลเท่ากับค่าของสมการโพลิโนเมียล

ดังตัวอย่างที่มีสมการโพลิโนเมียล $f(X) = X^8+X^6+X^5+X^4+1$ กำหนดชุดข้อมูลเริ่มต้น 10000000 ใช้ค่า L= 4, B=4 และจำนวนชุดข้อมูลที่ต้องการเป็น I = 3 สามารถสร้างชุดข้อมูลได้สูงสุด 255 ชุด แต่ชุดข้อมูลที่นำมาพิจารณาแค่ 21 ชุดข้อมูลเนื่องจากชุดข้อมูล ชุดที่ 21 มีค่าเท่ากับค่าของสมการโพลีโนมียล เมื่อดูจากรูปที่ 7 ชุดข้อมูลชุดแรกที่ถูกกำเนิดเป็นชุดแรกที่เลือกในการสร้างวงจรเลื่อนเฟสชุดข้อมูลถัดไปก็นับจากชุดแรกไปอีก 4 ชุดคือข้อมูลที่ B=2 เนื่องจากชุดข้อมูลดังกล่าวมีจำนวนหมายเลขหนึ่งน้อยกว่าค่า I ที่กำหนดไว้ ถ้ามีจำนวนหมายเลขหนึ่งมากกว่าค่า I ก็จะพิจารณาข้อมูลชุดถัดไปจนกว่าจะเจอตำแหน่งที่มีค่าตำแหน่งที่มีจำนวนหมายเลขหนึ่งน้อยกว่าหรือเท่ากับค่า I และทำเช่นนี้จนครบ



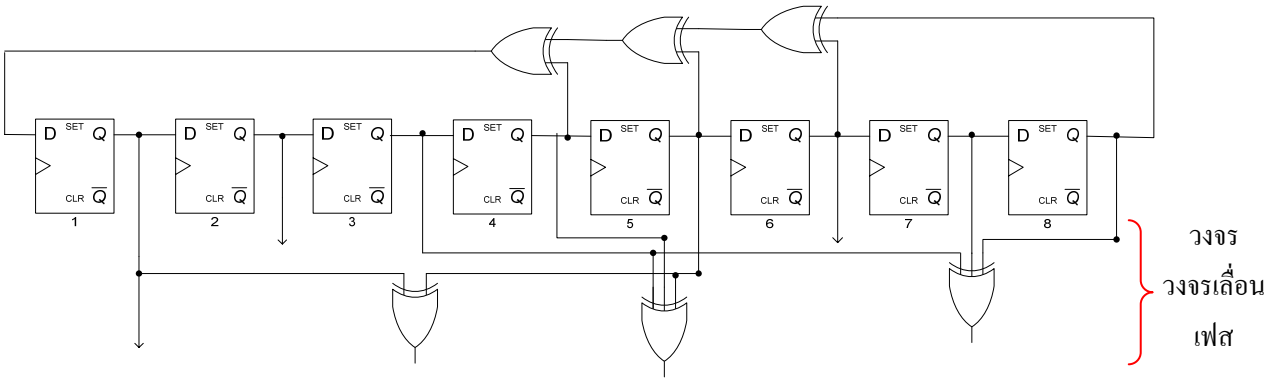
รูปที่ 7 ชุดข้อมูลที่ใช้ในการสร้างวงจรวงจรเลื่อนเฟส

วิธีการหาสมการสำหรับสร้างวงจรเลื่อนเฟสวิธีการดังกล่าวนี้ทำให้ชุดข้อมูลที่ใช้ในการพิจารณามีจำนวนน้อยลง ไม่จำเป็นต้องกำเนิดข้อมูลทุกความเป็นไปได้ของ LFSR ทั้งยังสามารถกำหนดจำนวนของ tap ของอินพุทและเอาท์พุทที่จะได้จากวงจรได้อีกด้วย

ข้อมูลที่ถูกเลือกเมื่อนำมาเขียนในรูปแบบของเมตริกซ์ ให้จำนวนหลักเป็นจำนวน D Flip-Flop ของ LFSR และให้แถวเป็นจำนวนเอาท์พุทของวงจรเลื่อนเฟสสามารถเขียนได้ดังนี้

$$H = \begin{pmatrix} 10000000 \\ 10001000 \\ 00011100 \\ 00100011 \end{pmatrix}$$

เมื่อพิจารณาจากเมตริกซ์นำมาสร้างวงจรเลื่อนเฟสโดยแถวแรกของเมตริกซ์เป็นชุดแรกของวงจรเลื่อนเฟส หมายเลข 1 คือตำแหน่งที่จะมีเอาท์พุทของ LFSR มาผ่านตัวดำเนินการ XOR แล้วออกเป็นเอาท์พุทของวงจรเลื่อนเฟส จากเมตริกซ์ H สามารถสร้างวงจรเลื่อนเฟสได้ดังรูปที่ 8



รูปที่ 8 วงจรเลื่อนเฟสที่รวมกับวงจร LFSR

การกำเนิดข้อมูลแบบ PRPG ด้วยวิธี LFSR เป็นวิธีการที่ชุดข้อมูลที่สามารถกำเนิดชุดข้อมูลได้จำนวนมาก แต่ข้อมูลที่กำเนิดมานั้นไม่สามารถที่จะตรวจจับความผิดพลาดที่เกิดขึ้นในวงจรได้ทั้งหมด ความผิดพลาดที่ PRPG ไม่สามารถตรวจจับได้ เรียกว่า ความผิดพลาดแบบ Random Pattern Resistant (R.P.R.) หรือความผิดพลาดที่ตรวจจับได้ยาก จึงได้มีการพัฒนาส่วนที่เป็น Deterministic Test Pattern Generator (DTPG) เพื่อเพิ่มประสิทธิภาพการทดสอบให้สามารถตรวจจับความผิดพลาดที่ตรวจจับได้ยาก

4.2.3 DTPG

เป็นการสร้างวงจรที่ใช้ในการเพิ่มประสิทธิภาพตรวจจับข้อผิดพลาดที่เป็น R.P.R ในวงจรอิเล็กทรอนิกส์ โดยวิธีการพัฒนาสามารถแบ่งได้ 3 แบบ [1] ดังนี้

1. ROM Compression เป็นวิธีที่พัฒนาขึ้นเพื่อช่วยลดขนาดของข้อมูลการทดสอบ โดยใช้เทคนิคการบีบอัด ซึ่งช่วยให้ขนาดของหน่วยความจำที่ใช้ในการเก็บข้อมูลลดลง แต่จะเป็นการเพิ่มวงจรขยายข้อมูลที่ถูกบีบอัดไว้

2. LFSR Reseeding เป็นวงจรที่เกิดจากการสร้าง LFSR และใช้ LFSR มากำเนิดวงจร DTPG ด้วยแต่ไม่มีการกำหนดค่าของอินพุท ใช้การคำนวณชุดข้อมูลทดสอบไว้ล่วงหน้าแล้วค่อยเลื่อนชุดข้อมูลดังกล่าวสู่วงจร LFSR โดยเรียกข้อมูลที่เกิดคำนวณชุดข้อมูลทดสอบไว้ล่วงหน้าว่า Seed

3. Embedding Deterministic Pattern เป็นวิธีการนำชุดข้อมูลการทดสอบของ PRPG ที่ไม่สามารถตรวจสอบความผิดพลาดได้มาทำการเปลี่ยนแปลงในบางบิต เช่นวิธี bit-Fixing, bit blipping หรือการ mapping logic

การเพิ่มวงจร DTPG ในกระบวนการทดสอบเพื่อลดจำนวนความผิดพลาดที่ตรวจจับไม่ได้ด้วย PRPG แต่ข้อเสียในการสร้างวงจร DTPG จำเป็นต้องใช้ทรัพยากรเพิ่มมากขึ้น การพัฒนา DTPG จึงควบคู่ไปกับการลดการใช้

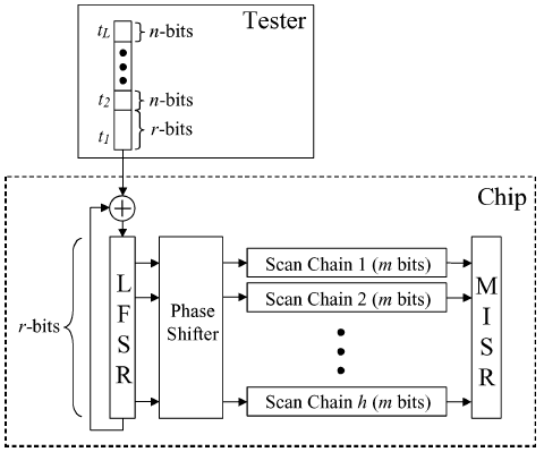
จำนวนทรัพยากร วิธีการสร้างวงจร DTPG ด้วยวิธีการ Reseeding จึงเป็นวิธีที่นำมาใช้ในการพัฒนาให้สามารถลดขนาดของข้อมูลที่ใช้ในการทดสอบและลดระยะเวลาการทดสอบได้

4.2.3.1 Reseeding

การทำ Reseeding ของวงจร DTPG เป็นการกำเนิดเซตของวงจรทดสอบที่ใช้วงจรที่ได้จากการปรับปรุงวงจร LFSR เพื่อกำเนิดชุดข้อมูลการทดสอบให้สามารถทดสอบความผิดพลาดได้มากขึ้น ในการคำนวณเซตของ แต่ละ Seed จะขึ้นอยู่กับแต่ละ Test Vector ที่ต้องการจะทดสอบ พื้นที่ในการเก็บข้อมูลทั้งหมดจึงมีขนาดใหญ่ แต่ด้วยข้อจำกัดของการทรัพยากรที่ใช้ในการพัฒนา จึงได้มีการพัฒนาการกำเนิดข้อมูลด้วยวิธี Reseeding ที่ลดขนาดของชุดข้อมูลลงเพื่อให้ใช้เวลาน้อยลงในการเลื่อนบิตข้อมูลสู่วงจร LFSR

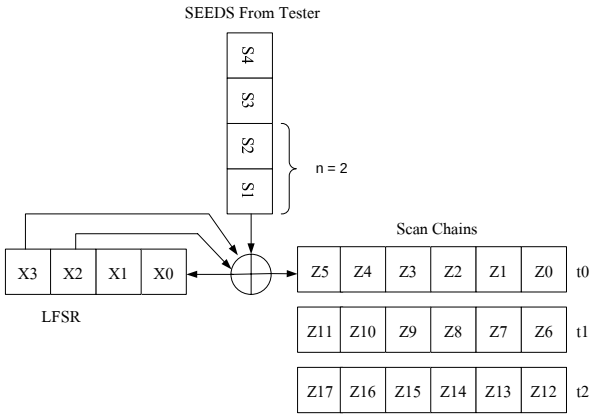
Static Reseeding เป็นการคำนวณชุดข้อมูลของแต่ละ Test Vector ที่มีการกำหนดค่าคงที่ของขนาดของชุดข้อมูลไว้คือ ค่า S_{max}+4 หรือ S_{max}+20 [6] โดย S_{max} คือจำนวนข้อมูลที่มีการเฉพาะเจาะจงค่าของบิตข้อมูลใน Test Vector นั้นๆ เพื่อที่จะสามารถตรวจจับความผิดพลาดได้ทุกจุด ซึ่งวิธีการนี้ต้องใช้ข้อมูลขนาดใหญ่ ต้องใช้พื้นที่ในการเก็บข้อมูลจำนวนมาก จำนวนข้อมูลที่ป้อนให้กับวงจร LFSR ก็จะมีจำนวนเยอะตามไปด้วย จึงได้มีการพัฒนาเพื่อลดขนาดของข้อมูลด้วยวิธี Partial Reseeding

Partial Reseeding [7][8] เป็นวิธีการหนึ่งที่ใช้การลดขนาดของเซตข้อมูลของแต่ละ Seed ลง จำนวนเซตของ Seed จะขึ้นอยู่กับขนาดของความยาวของ Scan chain ถ้าขนาดของ LFSR มีขนาดเท่ากับ r ข้อมูลที่ใช้กำหนดค่าเริ่มต้นมีขนาดเท่ากับขนาดของวงจร LFSR แต่ข้อมูลชุดถัดไป เท่ากับจำนวนสัญญาณนาฬิกาที่เหลือเพื่อเลื่อนบิตข้อมูลให้เต็ม Scan Chain ถ้า Scan chain มีค่าเท่ากับ m บิต ข้อมูลที่จะถูกเลื่อนเข้าสู่ LFSR จะเท่ากับ n บิต ทำให้ n < r เพราะข้อมูลของ Seed ที่ถูกเลื่อนไปก่อนหน้าจะมีผลย้อนกลับมาในสัญญาณนาฬิกาถัดไปด้วย XOR เกตที่อยู่ระหว่าง Tester และ LFSR โครงสร้างของ Partial Reseeding แสดงดังรูปที่ 9



รูปที่ 9 Partial Reseeding [7]

วิธีการเลื่อนบิตข้อมูลจาก Tester ด้วยวิธี Partial Reseeding ใช้การเลื่อนบิตข้อมูลเป็นลำดับ (Serial) กล่าวคือ บิตของข้อมูลที่เลื่อนจาก Tester ที่เวลา t=0 จะมีผลต่อบิตของข้อมูลที่เวลา t=1 ซึ่งทำให้ข้อมูลที่จะถูกเลื่อนไปยัง Scan Chain เป็นลำดับด้วย ส่งผลให้ไม่สอดคล้องกับโครงสร้างของ Scan Chain แบบ STUMP ที่จะทำงานในรูปแบบขนาน ซึ่งวิธีการนี้จะต้องใช้เวลาในการทดสอบ และสามารถทดสอบได้ทีละจุดเท่านั้น



รูปที่ 10 ตัวอย่างการสร้างสมการของ Partial Reseeding

เมื่อพิจารณาจากรูปที่ 10 ตัวอย่างการสร้างสมการของวิธีการเลื่อนบิตข้อมูลจาก Tester ด้วยวิธีการแบบ Partial Reseeding จากรูปประกอบไปด้วยวงจร LFSR ขนาด 4 D Flip-Flop, Seeds ที่จะถูกเลื่อนจาก Tester แต่ละชุดของ Seed มีค่า n เท่ากับ 2 และเซลล์ตรวจกวาด (Scan Cell) จำนวน 3 เซลล์ เมื่อเวลา t0 ชุดข้อมูลที่ปรากฏที่ Scan Chain เป็นค่าข้อมูลที่ตั้งค่าเริ่มต้นให้กับวงจร LFSR จะไม่มีการเลื่อนบิตข้อมูลจาก Tester ที่เวลา t1 จะเป็นการเลื่อนบิตข้อมูลจาก LFSR และทำการ XOR กับข้อมูลจาก Tester ซึ่งข้อมูลชุดดังกล่าวจะมีผลต่อเซลล์ตรวจกวาดที่ Z6-Z11 จากรูปทำให้ได้สมการของการเลื่อนบิตของ Seeds ที่ทำงานประสานกับวงจร LFSR ได้ดังสมการที่ 2

$$
\begin{aligned}
Z6 &= X1 \oplus X2 \oplus X3 \oplus S1 \\
Z7 &= X0 \oplus X1 \oplus X2 \oplus S2 \\
Z8 &= X0 \oplus X1 \oplus X2 \oplus X3 \\
Z9 &= X0 \oplus X1 \oplus X3 \oplus S1 \\
Z10 &= X0 \oplus X3 \oplus S1 \oplus S2 \\
Z11 &= X3 \oplus S2
\end{aligned}
\qquad (2)
$$

ที่เวลา t2 จะเป็นการเลื่อนบิตข้อมูลจาก LFSR และทำการ XOR กับข้อมูลจาก Tester ซึ่งข้อมูลชุดดังกล่าวจะมีผลต่อเซลล์ตรวจกวาดที่ Z12-Z17 แสดงดังสมการที่ 3

$$
\begin{aligned}
Z12 &= X1 \oplus S1 \oplus S3 \\
Z13 &= X1 \oplus S2 \oplus S4 \\
Z14 &= X0 \oplus S1 \\
Z15 &= X2 \oplus X3 \oplus S1 \oplus S2 \oplus S3 \\
Z16 &= X1 \oplus X2 \oplus S1 \oplus S2 \oplus S3 \oplus S4 \\
Z17 &= X0 \oplus X1 \oplus S1 \oplus S2 \oplus S4
\end{aligned}
\qquad (3)
$$

เทคนิค Partial Reseeding ใช้การเลื่อนบิตข้อมูลจาก Tester แบบลำดับ ทำให้ข้อมูลที่จะถูกเลื่อนไปยังเซลล์ตรวจกวาดเกิดการขัดแย้งกันและมีบางชุดของข้อมูลทดสอบทำให้สมการดังกล่าวไม่เป็นจริง เมื่อพิจารณาจากสมการที่ 2 โดยพิจารณาที่ทุกบิตข้อมูลในข้อมูลทดสอบมีค่าเป็น ‘1’โดย LFSR มีค่าเท่ากับ 1000 จะได้ว่า $X0 = 1, X1 = 0$ $X2 = 0, X3 = 0$ เพื่อให้ Z6 = 1 จึงทำให้ S1= 1 เท่านั้น Z7 มีค่าเท่ากับ 1 ก็ต่อเมื่อ S2 =0 เมื่อพิจารณาที่ Z9 จะ เกิดการขัดแย้งกันเนื่องจาก X0 = 1 และ S1 =1 จึงทำให้ Z9 เป็น ‘0’ ตลอด ไม่มีโอกาสที่จะเป็น ‘1’ ซึ่งการขัดแย้งที่เกิดขึ้นนี้เป็นการพิสูจน์ให้เห็นว่า เทคนิคดังกล่าวไม่สามารถกำเนิดข้อมูลได้ครบตรงกับข้อมูลทดสอบที่ต้องการได้เสมอไป

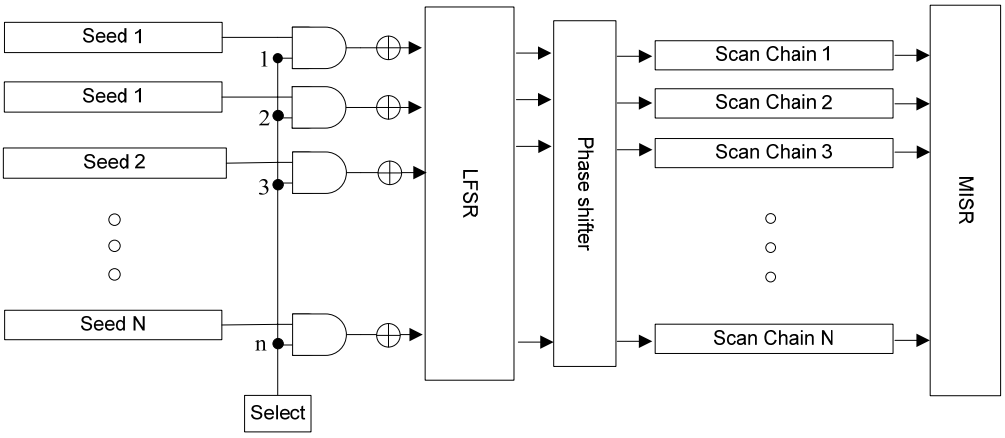
4.4 แนวทางการวิจัย

จากปัญหาของการกำหนดข้อมูลด้วยวิธี Partial Reseeding ที่ได้กล่าวข้างต้น ผู้วิจัยจึงได้เสนอวิธีการเพิ่มประสิทธิภาพการทดสอบเพื่อให้ลดระยะเวลาการทดสอบลง แต่ครอบคลุมการทดสอบ 100 เปอร์เซ็นต์ ลดจำนวนข้อมูลการทดสอบ อีกทั้งยังสามารถกำเนิดข้อมูลให้ตรงกับข้อมูลทดสอบที่ต้องการได้ทุกกรณี โดยการพัฒนาและดัดแปลงจากวิธี Reseeding ให้มีการทำงานแบบขนาน หรือ Multi-Reseeding การวิเคราะห์วงจรด้วย Logic Cones รวมทั้งการใช้วิธีการจัดเรียงวงจรตรวจกวาด ซึ่งผู้วิจัยคาดหวังว่าจะมีผลให้การคำนวณ Seed ได้เร็วและมีขนาดน้อยลง โดยมีหลักการดังต่อไปนี้

Multi-Reseeding

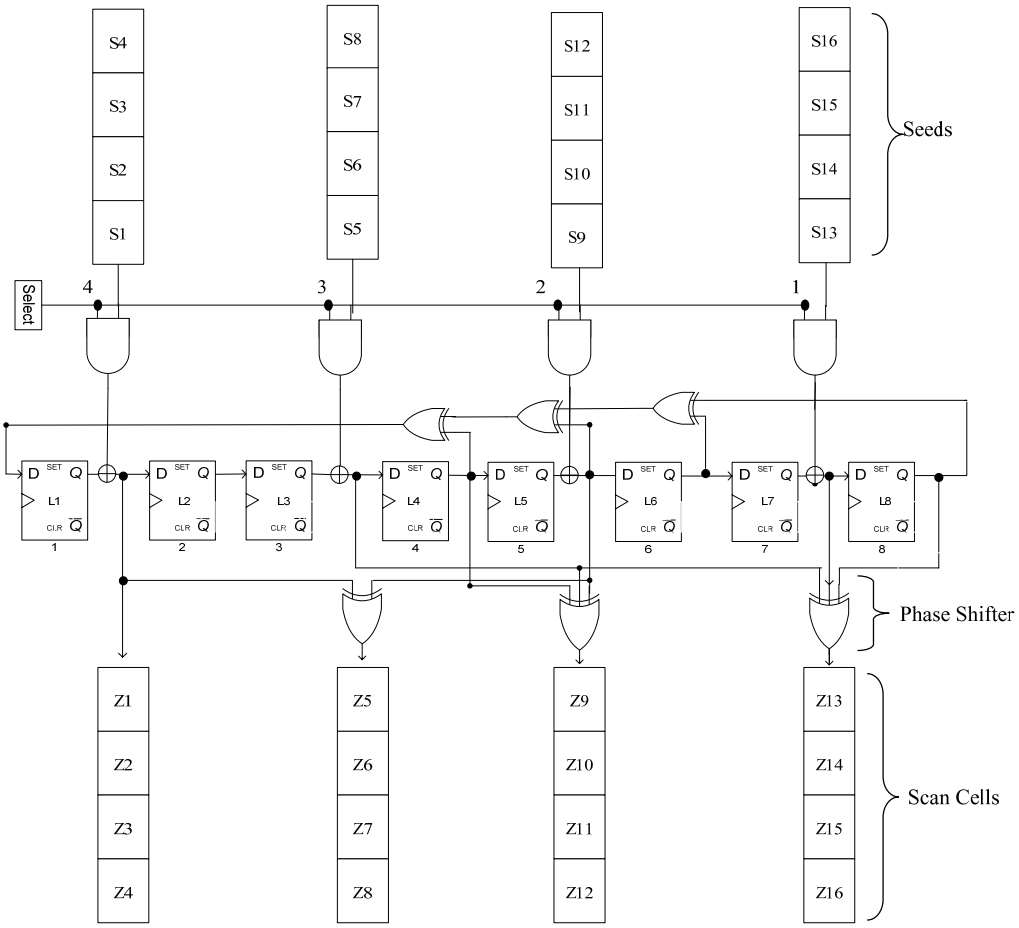
Multi-Reseeding เป็นการป้อนชุดข้อมูลเพื่อที่จะเจาะจงข้อมูลทดสอบของวงจร ให้สามารถครอบคลุมข้อมูลการทดสอบได้ 100 เปอร์เซ็นต์ โดยใช้การป้อนชุดข้อมูลในแบบขนานเข้าสู่โครงสร้างของเซลล์ตรวจกวาดแบบ STUMP เพื่อลดระยะเวลาการทดสอบลงแต่ไม่ลดประสิทธิภาพของการทดสอบ

วิธีการทำงานของ Multi-Reseeding ใช้การเลื่อนบิตข้อมูล Seeds ในรูปแบบขนานโดยจะมี AND เกต เลือกอินพุทเข้าสู่ LFSR เมื่อต้องการเอาท์พุทจาก LFSR สัญญาณ Select จะเท่ากับ ‘0’ และเมื่อต้องการให้นำข้อมูลจาก Seed เข้าสู่ LFSR สัญญาณ Select เท่ากับ ‘1’ โดยแต่ละชุดของ Seed ถูกควบคุมด้วยแต่ละบิตของขาสัญญาณ Select เพื่อให้สามารถควบคุมการเลื่อนบิตข้อมูลของ Seed เข้าสู่วงจรการทดสอบได้ง่ายและให้สามารถเจาะจงตำแหน่งและบิตข้อมูลที่ต้องการจะทดสอบได้ โครงสร้างของ Multi-Reseeding แสดงดังรูปที่ 11 และการระบุตำแหน่งของ Seed ที่จะมีผลต่อเซลล์ใน Scan Chains แสดงดังรูปที่ 12



รูปที่ 11 โครงสร้าง Multi-Reseeding

Multi-Reseeding จากรูปที่ 11 มีกลุ่มของ Seeds ที่ได้จากการคำนวณข้อมูลทดสอบทั้งหมด 4 ชุด, LFSR, วงจรการปรับเฟสที่มีเอาท์พุท 4 เอาท์พุท และเอาท์พุทของวงจรการปรับเฟสเป็นอินพุทให้กับเซลล์ตรวจกวาด โดย Seed แต่ละชุดใช้ตรวจจับข้อผิดพลาดของแต่ละเซลล์ ซึ่ง Seed ที่ประกอบด้วย S1, S2, S3 และ S4 ใช้เพื่อตรวจจับข้อผิดพลาดในเซลล์ตรวจกวาดที่ประกอบด้วยเซลล์ Z4, Z3, Z2 และ Z1 ส่วนในเซลล์ตรวจกวาดที่ประกอบด้วยเซลล์ Z12, Z11, Z10 และ Z9 จะถูกตรวจสอบด้วยข้อมูลใน Seed ที่ S9, S10, S11 และ S12 เป็นต้น



รูปที่ 12 Multi-reseeding

เมื่อพิจารณาที่รูปที่ 12 สามารถคำนวณหาสมการ Multi-Reseeding ที่มีความสัมพันธ์กับเซลล์ต่างๆในเซลล์ตรวจกวาดได้ โดยค่าเริ่มต้นของ LFSR มีค่าเท่ากับ 10000000 ที่เวลา t = 0 ชุดข้อมูลที่เก็บใน Seed จะถูกเลื่อนเข้าสู่วงจร LFSR และ วงจรปรับเฟสแบบขนาน ซึ่งชุดแรก ที่ประกอบด้วย S1, S2, S3 และ S4 จะมีผลต่อเซลล์ตรวจกวาดในลำดับสุดท้ายของเซลล์ตรวจกวาด คือเซลล์ที่ Z16, Z12, Z8 และ Z4 สมการที่มีผลต่อเซลล์ในเซลล์ตรวจกวาด ณ ที่เวลาต่างๆ แสดงดังสมการที่ 4 - 7

t = 0 ค่าเริ่มต้นของ LFSR = 10000000

$$\begin{aligned} Z4 &= S1 \oplus L1 \\ Z8 &= S1 \oplus S9 \oplus L5 \oplus L1 \\ Z12 &= S5 \oplus S9 \oplus L3 \oplus L4 \oplus L5 \\ Z16 &= S13 \oplus S5 \oplus L8 \oplus L7 \oplus L3 \end{aligned} \quad (4)$$

t = 1, LFSR = 01000000

$$\begin{aligned} Z3 &= S2 \oplus L1 \\ Z7 &= S2 \oplus S10 \oplus L5 \oplus L1 \\ Z11 &= S6 \oplus S10 \oplus L3 \oplus L4 \oplus L5 \\ Z15 &= S14 \oplus S6 \oplus L8 \oplus L7 \oplus L3 \end{aligned} \quad (5)$$

t = 2, LFSR = 00100000

$$\begin{aligned} Z2 &= S3 \oplus L1 \\ Z6 &= S3 \oplus S11 \oplus L5 \oplus L1 \\ Z10 &= S7 \oplus S11 \oplus L3 \oplus L4 \oplus L5 \\ Z14 &= S15 \oplus S7 \oplus L8 \oplus L7 \oplus L3 \end{aligned} \quad (6)$$

t = 3, LFSR = 00010000

$$\begin{aligned} Z1 &= S4 \oplus L1 \\ Z5 &= S4 \oplus S12 \oplus L5 \oplus L1 \\ Z9 &= S8 \oplus S12 \oplus L3 \oplus L4 \oplus L5 \\ Z13 &= S16 \oplus S8 \oplus L8 \oplus L7 \oplus L3 \end{aligned} \quad (7)$$

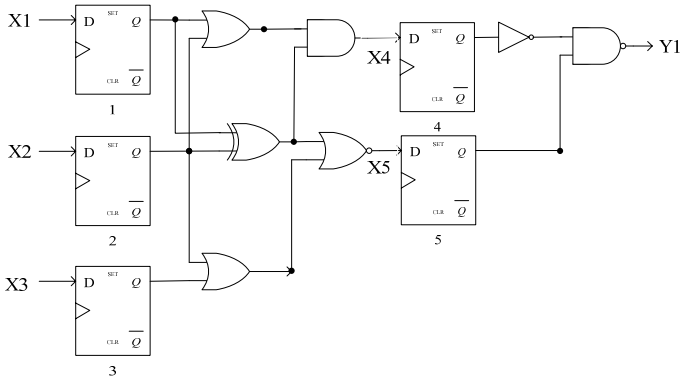
ถ้าต้องข้อมูลในชุดทดสอบมีค่าเท่ากับ ‘1’ สามารถคำนวณชุดทดสอบของ Seed ดังสมการที่ (8)

$$\begin{array}{llll} S1=0 & S5=1 & S9=0 & S13=0 \\ S2=1 & S6=1 & S10=0 & S14=0 \\ S3=1 & S7=0 & S11=0 & S15=0 \\ S4=1 & S8=0 & S12=0 & S16=1 \end{array} \quad (8)$$

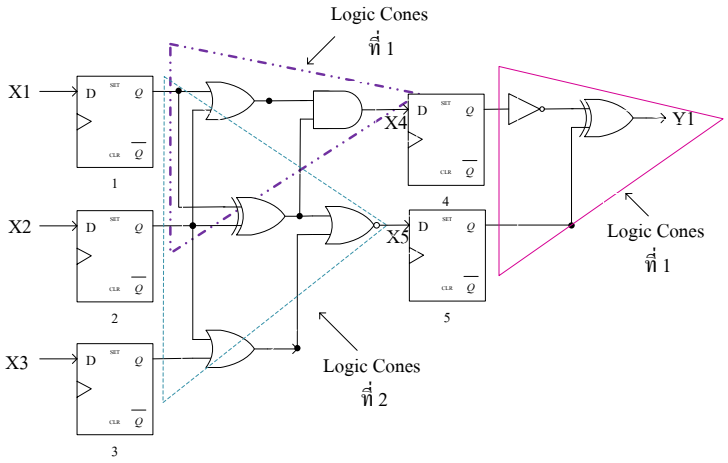
การพัฒนา Reseeding แบบ Multi-Reseeding นี้สามารถจะเจาะจงความผิดพลาดได้ทุกตำแหน่งความผิดพลาดและสามารถลดระยะเวลาการทดสอบได้ แต่ข้อมูลที่ต้องใช้ในแต่ละ Seed นั้นต้องมีขนาดเท่ากับจำนวนของเซลล์ตรวจกวาดในแต่ละ Scan chain ผู้วิจัยจึงนำเสนอแนวคิดในการพิจารณา Logic Cones ของเซลล์ตรวจกวาด โดยมีเป้าหมายให้การคำนวณ Seed มีขนาดของข้อมูลน้อยที่สุด

Logic Cones

การพิจารณา Logic Cones [2] เป็นวิธีการวิเคราะห์วงจรโดยการพิจารณากลุ่มของอินพุทที่มีเอาท์พุทเป็นตัวเดียวกัน เพื่อลดจำนวนชุดทดสอบของวงจร และสามารถนำมาใช้กับวงจรเซลล์ตรวจกวาดได้ดังรูปที่ 13 เป็นกลุ่มของเซลล์ตรวจกวาดที่มีการเชื่อมโยงด้วยวงจรตรรกะ เมื่อพิจารณาด้วย Logic Cones สามารถแบ่งกลุ่มของเซลล์ตรวจกวาดได้ 3 กลุ่ม กลุ่มที่ 1 ประกอบด้วย D Flip-Flop ที่ 1 และ 2 มีเอาท์พุทเป็น X4 กลุ่มที่ 2 ประกอบด้วย D Flip-Flop ที่ 1, 2 และ 3 มีเอาท์พุทเป็น X5 ซึ่ง X5 และ X4 เป็นอินพุทให้กับ Logic Cones กลุ่มที่ 3 ที่ประกอบด้วย D Flip-Flop ที่ 4 และ 5 ซึ่งมีเอาท์พุทเป็น Y1 การแบ่งกลุ่มวงจรตรวจกวาดด้วย Logic Cones แสดงดังรูปที่ 14



รูปที่ 13 วงจรตรวจกวาด



รูปที่ 14 การพิจารณาวงจรตรวจกวาดด้วย Logic Cones

การจัดเรียงวงจรตรวจกวาด เป็นวิธีการเรียงกลุ่มของ Flip-Flop เพื่อช่วยให้ง่ายในการควบคุมและติดตามผลค่าข้อมูล ร่วมด้วยการพิจารณา Logic Cones เพื่อให้วงจรตรวจกวาดที่อยู่ในกลุ่มของ Logic Cones เดียวกันอยู่ใน Scan Chains เดียวกันหรืออยู่ในลำดับตำแหน่งภายใน Scan Chains ที่มีความสัมพันธ์กัน หรือจัดเรียงกลุ่มเซลล์ตรวจกวาดที่มีการค่าข้อมูลเป็น ‘0’ และ ‘1’ บิตแยกออกจากกลุ่มไม่ทราบสถานะ เพื่อให้ชุดข้อมูลที่จะเลื่อนเข้าสู่ Scan Chains สามารถลดขนาดลงได้และสามารถกำเนิดข้อมูลได้ครบตรงกับข้อมูลทดสอบที่ต้องการได้ง่ายขึ้น

5. ขอบเขตของการวิจัย

5.1 พัฒนาวิธีการทดสอบข้อผิดพลาดของวงจรที่ออกแบบ เพื่อประยุกต์ใช้กับอุปกรณ์ลอจิกแบบโปรแกรมได้

5.2 เปรียบเทียบผลลัพธ์ของวิธีการทดสอบที่นำเสนอว่าสามารถประหยัดหน่วยความจำและทรัพยากรของอุปกรณ์ลอจิกแบบโปรแกรมได้

6. ขั้นตอนและวิธีการดำเนินการวิจัย

ขั้นที่ 1: ศึกษาแนวทาง และวิธีการดำเนินงานวิจัย
ขั้นที่ 2: ศึกษาวิธีการทดสอบวงจร
ขั้นที่ 3: ศึกษาและทดสอบวิธีการสร้างวงจร
ขั้นที่ 4: ออกแบบระบบ สำหรับวิธีทดสอบวิธีการสร้างวงจร
ขั้นที่ 5: พัฒนาระบบ ทดสอบจำนวนทรัพยากรที่ใช้ และจำนวน fault coverage ของวิธีที่ได้
ขั้นที่ 6: ศึกษาการนำการทดสอบด้วยอุปกรณ์ลอจิกแบบโปรแกรมได้
ขั้นที่ 7: ประยุกต์การทดสอบด้วยอุปกรณ์ลอจิกแบบโปรแกรมได้
ขั้นที่ 8: พัฒนาระบบ ทดสอบการทำงาน ทดสอบจำนวนทรัพยากรที่ใช้ และจำนวน fault coverage
ขั้นที่ 9: ปรับปรุงและทดสอบระบบทั้งระบบ
ขั้นที่ 10: สรุปผล จัดทำรายงานฉบับสมบูรณ์

7. แผนการดำเนินงาน

เดือน	ขั้นตอนการดำเนินงาน									
	1	2	3	4	5	6	7	8	9	10
มิ.ย. 50										
ก.ค. 51										
ส.ค. 51										
ก.ย. 51										
พ.ย. 51										
ธ.ค. 51										
ม.ค. 52										
ก.พ. 52										
มี.ค. 52										
มิ.ย. 52										
ก.ค. 52										
ส.ค. 52										
ก.ย. 52										
พ.ย. 52										
ธ.ค. 52										
ม.ค. 52										
ก.พ. 52										

เอกสารอ้างอิง

[1] Stroud, Charles E (2002), **A Designer's Guide to Built-in Self-Test**, Vol. 19: Springer

[2] Laung-Terng Wang, Cheng-Wen Wu and Xiaoqing Wen, (2006) **VLSI Test Principles and Architectures design for testability**, page 308-309, San Francisco: Elsevier

[3] P. Fišer, H. Kubatova, **Pseudorandom Testing – A Study of the Effect of the Generator Type,** Czech Technical University in Prague, 2005

[4] J. Rajski, N. Tamarapalli, and J. Tyszer, **Automated synthesis of large. phase shifters for built-in self-test**, Proc. ITC, pp. 1047–1056, 1998.

[5] D. Kagaris, **A unified method for phase shifter computation**, ACM Transactions on Design Automation of Electronic Systems, pp. 157–167, 2005

[6] Ahmad A. Al-Yanani, Subhasish Mitra, and Edward J. MacCluskey, **Short Papers: Optimized Reseeding by Seed ordering and Encoding**, IEEE International on Computer-Aided of Integrated circuits and systems, pp. 265-270, 2006

[7] C.V. Krishna and N.A. Touba, **Hybrid BIST Using an Incrementally Guided LFSR**, Proc. of IEEE Symposium on Defect and Fault Tolerance, pp. 217-224, 2003.

[8] C.V. Krishna, A. Jas, and N.A. Touba, **Test Vector Encoding Using Partial LFSR Reseeding**, Proc. of IEEE International Test Conference, pp. 885-893, 2001.

[9] C.V. Krishna, Albijit Jas and N.A. Touba, **Achieving High Encoding Efficiency With Partial Dynamic LFSR Reseeding**, ACM Transactions on Design Automation of Electronic System, pp.500-516, 2004

[10] Gang Zeng and Hideo Ito, **Hybrid BIST for System-on-a-Chip Using an Embedded FPGA Core**, Proc. of IEEE VLSI Test Symposium, 2004