บทที่ 4

โปรแกรมการให้กำเนิดรูปแบบสัญญาณทดสอบโดยอัตโนมัติ โดยใช้วิธีรายการลดทอน

โปรแกรมให้กำเนิดรูปแบบสัญญาณทดสอบโดยอัตโนมัติ ใช้ทฤษฎีทางคณิตศาสตร์ดีสครีต เกี่ยวกับเรื่อง เซต ฟังก์ชัน Intersection และทฤษฎีโครงสร้างข้อมูลเพื่อสร้างแบบจำลองวงจร รายการจุดเสีย รายการลดทอนและการค้นหาข้อมูลในหน่วยความจำ การท่องเข้าไปในกราฟเป็นแบบการค้นหาทางลึกผสมกับการค้นหาแนวกว้าง เทคนิคการเขียนโปรแกรมเป็นแบบ Recursive เข้ามาช่วยในการเขียนโปรแกรม

4.1 ข่ายระบบงานของโปรแกรมเอทีพีจี

ข่ายระบบงานของโปรแกรมเอทีพีจี แสดงดังรูปที่ 4.1 แสดงขั้นตอนการทำงานของโปรแกรม ประกอบด้วย

4.1.1 การสร้างวงจรในแบบจำลองของกราฟแบบไม่มีทิศทางในหน่วยความจำ

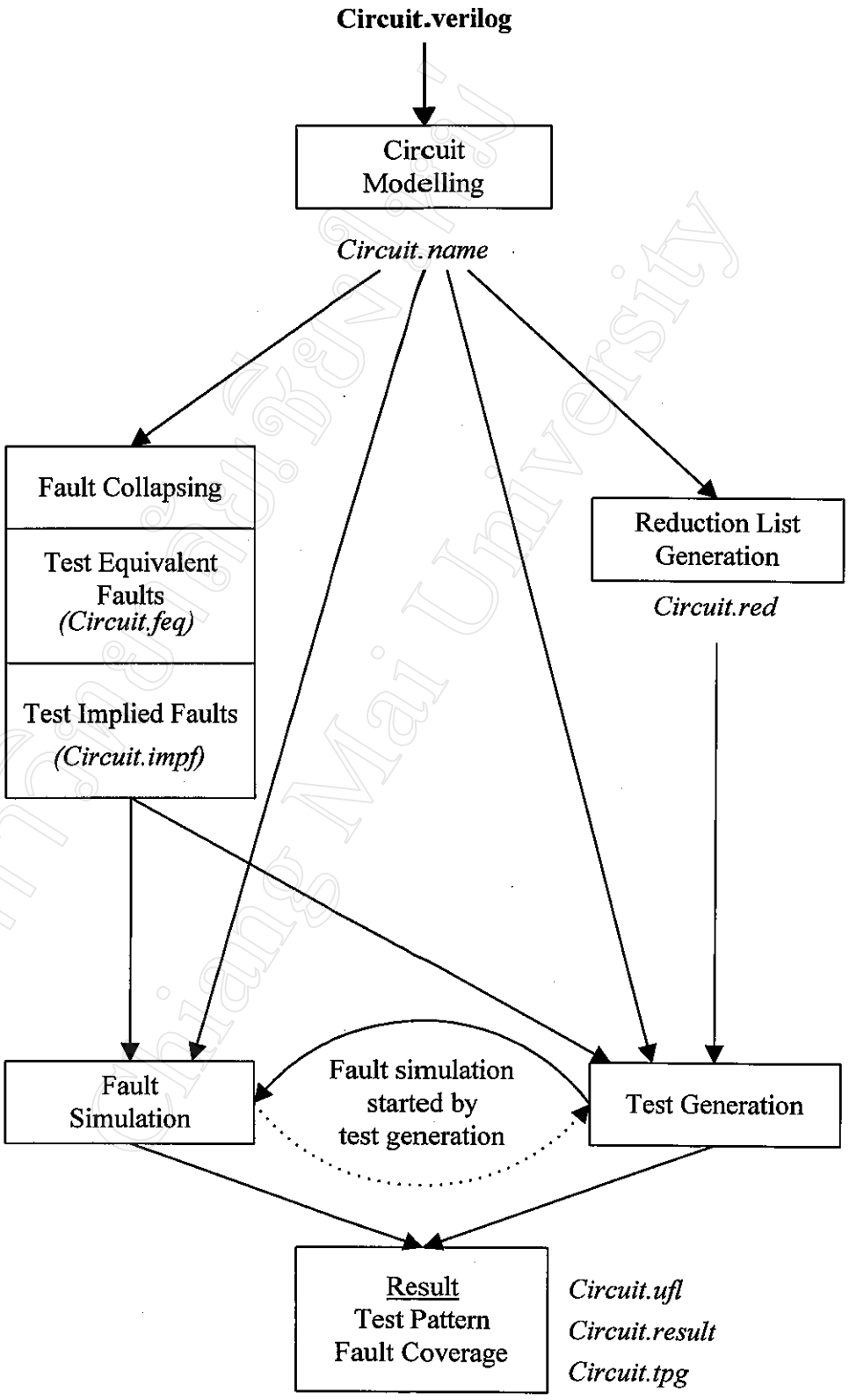
4.1.2 การยุบรวมจุดเสีย (Fault Collapsing)

4.1.3 การสร้างรายการลดทอน (Reduction List Generation)

4.1.4 การจำลองการทำงานจุดเสีย (Fault Simulation)

4.1.5 การให้กำเนิดรูปแบบสัญญาณทดสอบโดยอัตโนมัติ (ATPG)

ข่ายระบบงานของโปรแกรมเอทีพีจี แสดงดังรูปที่ 4.1 มีรายละเอียดการทำงาน และวิธีการทำดังต่อไปนี้



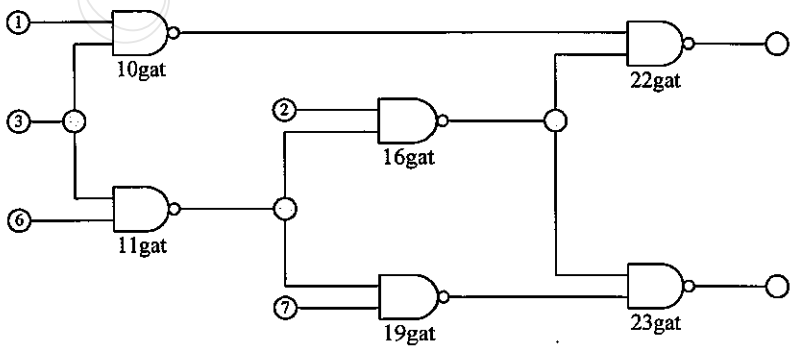
รูปที่ 4.1 ข่ายระบบงานของ
การให้กำเนิดรูปแบบสัญญาณทดสอบโดยอัตโนมัติ
(Automatic Test Pattern Generation)

4.2 โปรแกรมสร้างแบบจำลองวงจรในรูปของกราฟแบบไม่มีทิศทาง

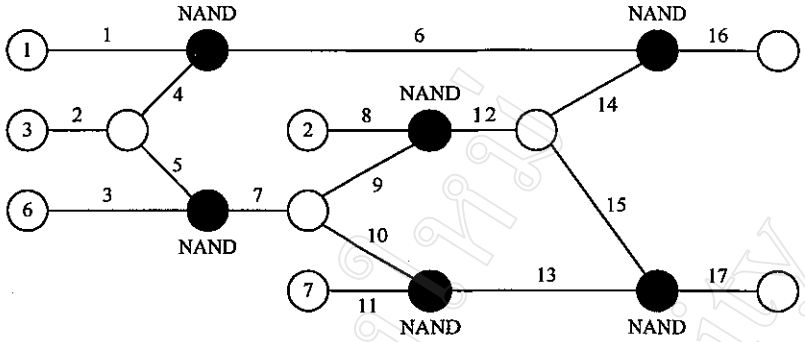
วงจรมาตรฐานของ ISCAS Benchmark Circuits ลักษณะไฟล์ของวงจรจะอยู่ในรูปแบบรายการข่ายวงจร (Network List, NetList)

ตัวอย่างการสร้างกราฟ จากวงจร c17.vm ซึ่งมีรายการข่ายวงจร ดังต่อไปนี้

```
module
c17(22gat,23gat,1gat,2gat,3gat,6gat,7gat);
// no. of NAND:    6
parameter sgd=1;
input 1gat; input 2gat;
input 3gat; input 6gat;
input 7gat;
output 22gat;
output 23gat;
nand #sgd nand_1(10gat,1gat,3gat);
nand #sgd nand_2(11gat,3gat,6gat);
nand #sgd nand_3(16gat,2gat,11gat);
nand #sgd nand_4(19gat,11gat,7gat);
nand #sgd nand_5(22gat,10gat,16gat);
nand #sgd nand_6(23gat,16gat,19gat);
endmodule
```



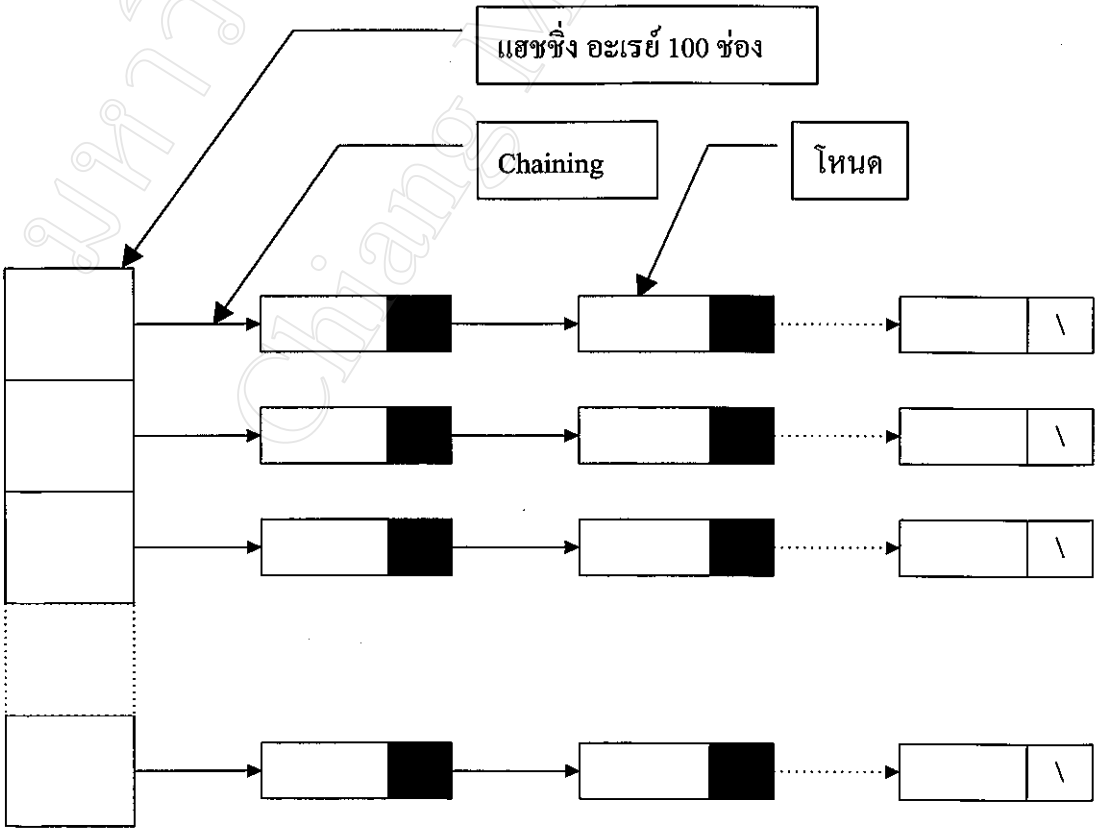
รูปที่ 4.2 วงจร c17.vm



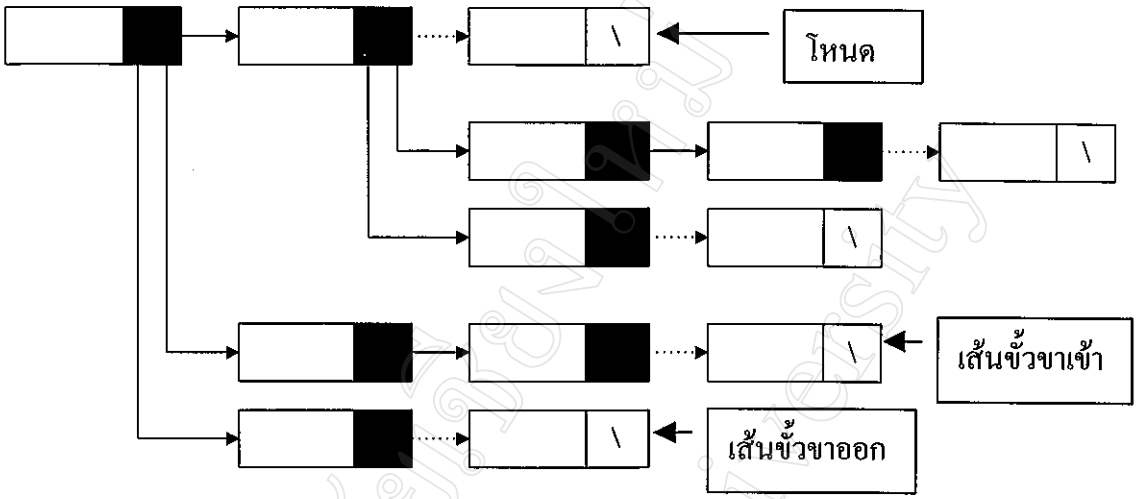
รูปที่ 4.3 รูปกราฟของวงจร c17.vm

จากรูปที่ 4.3 การสร้างจะประกอบไปด้วยโครงสร้างของโหนดและของเส้น โดยใช้โครงสร้างข้อมูลแบบรายการเดี่ยว (Single Link List) รายการคู่ (Double Link List) [3] และแฮชชิ่ง (Hashing) [3] ใช้วิธีเชื่อมโยงแบบลูกโซ่ (Chaining) ใช้ในการค้นหาโหนดเพื่อสร้างเส้นในการเชื่อมต่อที่เป็นขั้วขาเข้ากับขั้วขาออกของเกต

การสร้างรูปกราฟของวงจรในหน่วยความจำมีลักษณะการเชื่อมต่อดังรูปที่ 4.4 รูปที่ 4.5 ตามลำดับ



รูปที่ 4.4 แสดงการเชื่อมต่อระหว่างแฮชชิ่งกับโหนด



รูปที่ 4.5 แสดงการเชื่อมต่อระหว่างโหนดกับเส้น

4.3 วิธีการท่องเข้าไปในกราฟ

การวิเคราะห์เพื่อตรวจสอบการทำงาน และเส้นทางเดินของแต่ละโหนดของกราฟ โดยทั่วไปแล้วมีอยู่ 2 หลักวิธี คือ หลักวิธีการค้นหาแนวกว้าง (Breadth - first search) [3] และหลักวิธีการค้นหาทางลึก (Depth-first search) [3] หลักวิธีแนวกว้างนั้นจะใช้คิว (Queue) เป็นโครงสร้างข้อมูลช่วยเพื่อเก็บรักษาโหนดไว้สำหรับขบวนการต่อไป ส่วนหลักวิธีหาทางลึกนั้นจะใช้สแตก (Stack)

ระหว่างการทำงานนั้นแต่ละโหนดของกราฟจะอยู่ในสถานะใดสถานะหนึ่งใน 3 สถานะดังต่อไปนี้

1.) สถานะพร้อม (Ready State) เป็นสถานะเริ่มต้นของแต่ละโหนด

2.) สถานะรอ (Waiting State) แต่ละโหนดจะอยู่ในคิวหรือสแตกเพื่อรอที่จะทำงาน

3.) สถานะโปรเซส (Process State) แต่ละโหนดจะถูกดำเนินการตามขั้นตอน

4.3.1 หลักวิธีหาค้นหาแนวกว้าง (Breadth - first search)

จะเริ่มต้นขึ้นที่จุดสตาร์ท สมมติให้โหนด A เป็นจุดสตาร์ท จากนั้นตรวจสอบเพื่อนบ้านของโหนด A ทั้งหมด จากนั้นตรวจสอบเพื่อนบ้านของเพื่อนบ้านของโหนด A ไปเรื่อยๆ โดยใช้คิวเก็บโหนดเพื่อรอที่จะดำเนินการต่อไป ขั้นตอนของหลักวิธีนี้มีดังนี้

1.) เริ่มต้นทำโหนดทั้งหมดให้อยู่ในสถานะพร้อม

2.) เลือนโหนด A ไว้ในคิว แล้วเปลี่ยนสถานะเป็นรอ

3.) ทำซ้ำข้อ 4 และข้อ 5 จนกระทั่งคิวว่าง

4.) เอาโหนดที่ตำแหน่ง Front ชี้อยู่มาโปรเซสแล้วเปลี่ยนสถานะเป็นโปรเซส

5.) เอาโหนดเพื่อนบ้านทั้งหมดที่ตำแหน่ง Front ชี้อยู่ต่อท้ายคิว แล้วเปลี่ยนสถานะเป็นรอ

6.) จบการทำงาน

4.3.2 หลักวิธีค้นหาทางลึก (Depth – first search)

สมมติให้โหนด A เป็นจุดเริ่มต้น จากนั้นตรวจสอบแต่ละโหนดตามเส้นทางของโหนด A คือได้เพื่อนบ้านของโหนด A จากนั้นตรวจสอบทีละโหนดเพื่อนบ้านของเพื่อนบ้านโหนด A ไปเรื่อยๆ จนถึงโหนดสุดท้ายแล้วย้อนกลับตามเส้นทางเดิมเพื่อตรวจสอบโหนดอื่นๆ ต่อไปขั้นตอนของหลักวิธีนี้มีดังนี้

1.) เริ่มต้นทำให้โหนดทั้งหมดอยู่ในสถานะพร้อม

2.) เก็บโหนด A ลงสแตก และเปลี่ยนสถานะเป็นรอ

3.) ทำซ้ำข้อ 4 และข้อ 5 จนกระทั่งสแตกว่าง

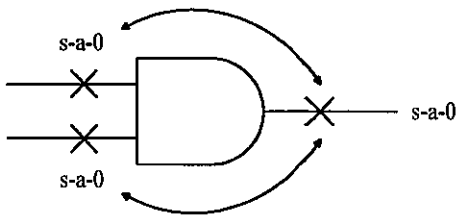
4.) ตรวจสอบโหนดชั้นบนสุดของสแตก และเปลี่ยนสถานะเป็นโปรเซส

5.) เลื่อนสแตกไปที่โหนดเพื่อนบ้านของ A ทั้งหมด ยังคงอยู่ในสถานะพร้อม และเปลี่ยนสถานะเป็นรอ

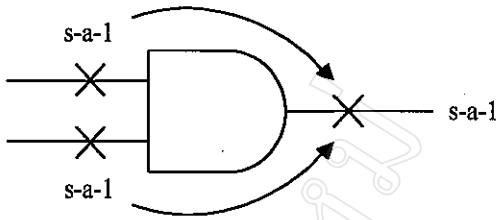
6.) จบการทำงาน

4.4 การยุบรวมจุดเสีย

เป็นกระบวนการยุบรวมจุดเสียประเภทเดียวกันเข้าด้วยกันเป็นกลุ่มๆ (Fault Class) เพื่อไม่จำเป็นต้องเสียเวลาทดสอบจุดเสียทุกจุดในวงจร โดยแต่ละกลุ่มของจุดเสียนั้นจะมีจุดเสียเพียง 1 จุด ที่เป็นผู้แทนจุดเสียทั้งหมดในกลุ่มนั้น เพื่อนำไปใช้ในการหารูปแบบสัญญาณทดสอบ การยุบรวมจุดเสียโดยใช้วิธีการทดสอบจุดเสีย ที่สมมูล (Test Equivalent Faults) กับการทดสอบจุดเสียที่สมนัย (Test Implied Faults) [22]



รูปที่ 4.6 การทดสอบจุดเสียที่สมมูล ของ AND เกต [22]

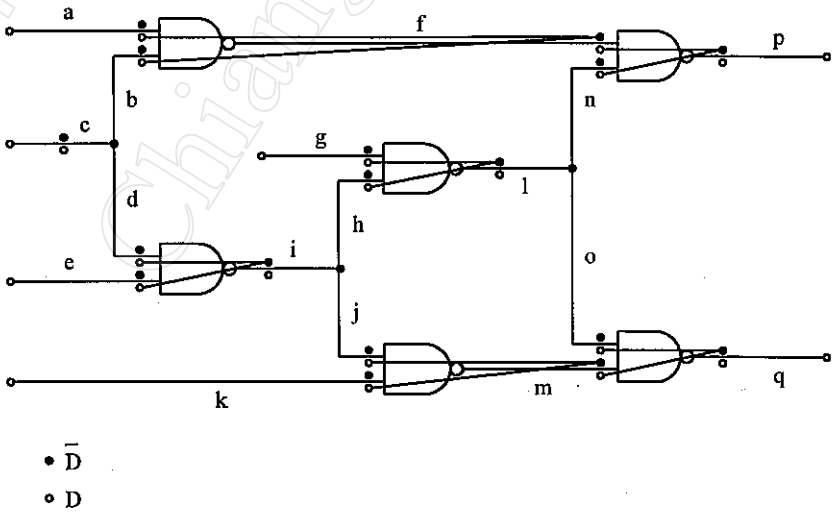


รูปที่ 4.7 การทดสอบจุดเสียที่สมนัย ของ AND เกต [22]

ตารางที่ 4.1 กฎการยุบรวมจุดเสีย (Fault Collapsing Rules)

ชนิดของ Gate	Fault	Fault Collapsed Into
NOT	A s-a-0	Z s-a-1
NOT	A s-a-1	Z s-a-0
AND	A s-a-0	Z s-a-0
NAND	A s-a-0	Z s-a-1
OR	A s-a-1	Z s-a-1
NOR	A s-a-1	Z s-a-0

A คือ ขั้วขาเข้าของเกต, Z คือ ขั้วขาออกของเกต

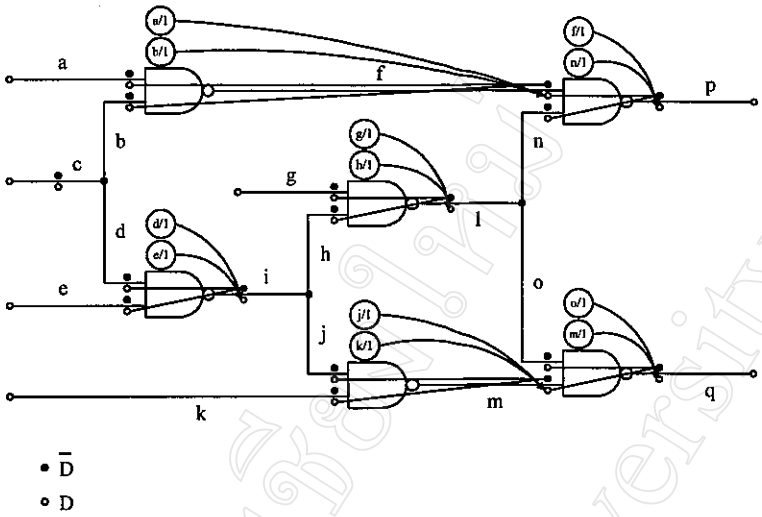


รูปที่ 4.8 แสดงการยุบรวมจุดเสียโดยใช้วิธีการทดสอบจุดเสียที่สมมูล

ตัวอย่างการยุบรวมจุดเสียของวงจร c17.vm ผลลัพธ์ที่ได้จากการทดสอบจุดเสียที่สมมูลที่เป็นประเภทเดียวกันเข้าด้วยกันเป็นกลุ่ม ๆ จากรูปที่ 4.8 ดังแสดงในตารางที่ 4.2

ตารางที่ 4.2 การรวมจุดเสียที่สมมูลเข้าด้วยกันเป็นกลุ่ม

ผู้แทนจุดเสีย (Fault Representative)	กลุ่มของจุดเสียประเภทเดียวกัน
a/1	{a/1}
f/1	{f/1, a/0, b/0}
b/1	{b/1}
c/1	{c/1}
c/0	{c/0}
d/1	{d/1}
e/1	{e/1}
i/1	{i/1, d/0, e/0}
i/0	{i/0}
g/1	{g/1}
h/1	{h/1}
l/1	{l/1, g/0, h/0}
l/0	{l/0}
j/1	{j/1}
k/1	{k/1}
m/1	{m/1, j/0, k/0}
n/1	{n/1}
o/1	{o/1}
p/1	{p/1, f/0, n/0}
p/0	{p/0}
q/1	{q/1, o/0, m/0}
q/0	{q/0}



รูปที่ 4.9 แสดงการยุบรวมจุดเสียโดยใช้วิธีการทดสอบจุดเสียที่สมนัย

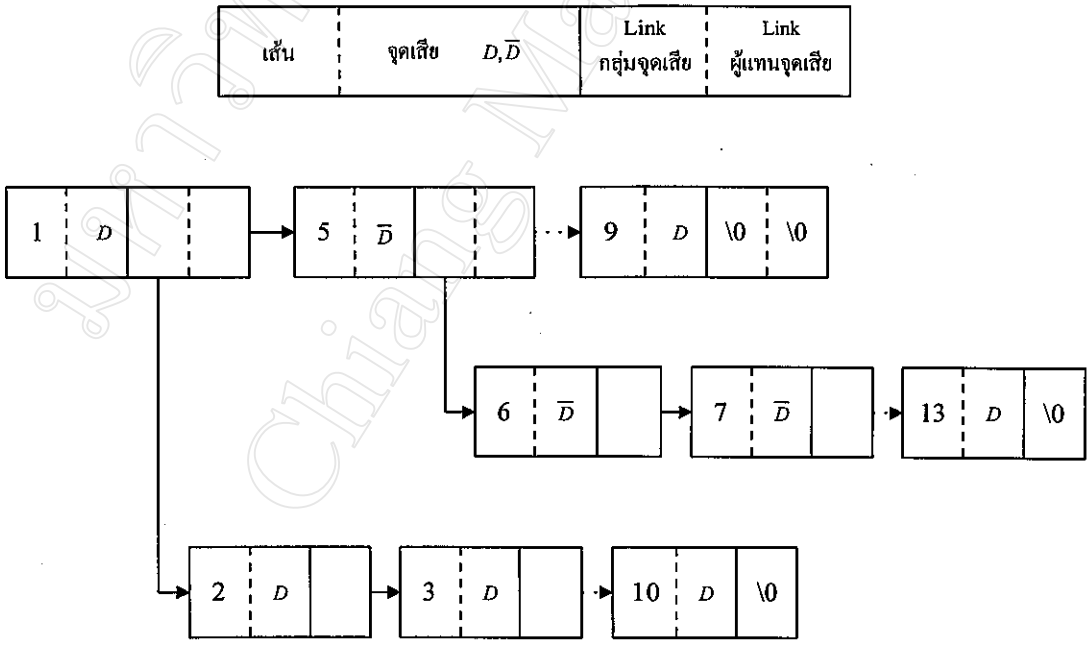
ตารางที่ 4.3 การรวมจุดเสียที่สมนัยเข้าด้วยกันเป็นกลุ่ม

ผู้แทนจุดเสีย (Fault Representative)	กลุ่มของจุดเสียประเภทเดียวกัน
a/1	{a/1} and {p/1, f/0, n/0}
f/1	{a/0, b/0} and {p/0}
b/1	{b/1} and {p/1, f/0, n/0}
c/1	{c/1}
c/0	{c/0}
d/1	{d/1} and {i/0}
e/1	{e/1} and {i/0}
i/1	{i/1, d/0, e/0}
g/1	{g/1} and {l/0}
h/1	{h/1} and {l/0}
l/1	{g/0, h/0}
j/1	{j/1} and {q/1, o/0, m/0}
k/1	{k/1} and {q/1, o/0, m/0}
m/1	{m/1, j/0, k/0} and {q/0}
n/1	{n/1} and {p/0}
o/1	{o/1} anf {q/0}

ผลลัพธ์ที่ได้การทดสอบจุดเสียที่สมนัยที่เป็นประเภทเดียวกันเข้าด้วยกันเป็นกลุ่ม ๆ จากรูปที่ 4.9 ดังแสดงในตารางที่ 4.3

หลังจากการทดสอบจุดเสียที่สมนัย สามารถยุบรวมจุดเสียได้เพิ่มขึ้น เพราะว่าจากรูปที่ 4.9 a/1, b/1 ทดสอบจุดเสียที่สมนัย f/0 ได้ และ f/0 สามารถทดสอบจุดเสียที่ สมมูล p/1 ได้ ดังนั้นจึงสามารถนำกลุ่มของจุดเสีย p/1 = {p/1, f/0, n/0} ไปรวมกับกลุ่มของ จุดเสีย a/1 และ b/1 ได้ และ j/1, k/1 ทดสอบจุดเสียที่สมนัย m/0 ได้ และ m/0 สามารถทดสอบจุดเสียที่สมมูล q/1 ได้ ดังนั้นจึงสามารถนำกลุ่มของจุดเสีย q/1 = {p/1, f/0, n/0} ไปรวมกับกลุ่มของ จุดเสีย j/1 และ k/1 ได้ o/1, m/1 ทดสอบจุดเสียที่สมนัย q/0 ได้ สามารถนำ q/0 ไปรวมกับ o/1 และ m/1 ได้ d/1, e/1 ทดสอบจุดเสียที่สมนัย i/0 ได้ g/1, h/1 ทดสอบจุดเสียที่สมนัย l/0 และ p/1, n/1 ทดสอบจุดเสียที่สมนัย p/0 ได้ จึงสามารถยุบรวมจุดเสียมาไว้ในกลุ่มเดียวกันได้

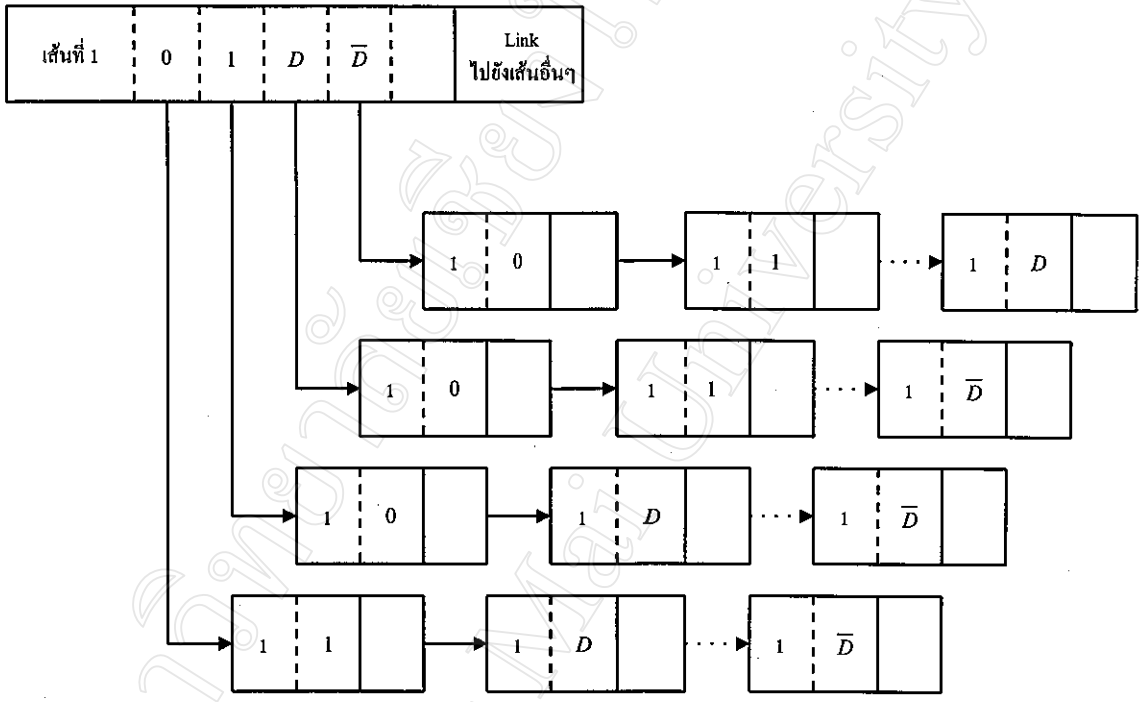
การยุบรวมจุดเสียมีอยู่ 2 ขั้นตอน คือ การทดสอบจุดเสียที่สมมูล และการทดสอบจุดเสียที่สมนัย จุดเสียที่ถูกยุบรวมกันเป็นกลุ่มเรียบร้อยแล้ว เก็บไว้ในโครงสร้างข้อมูลแบบรายการโยงเดี่ยวในหน่วยความจำ และเก็บไว้ในแฟ้มข้อมูล



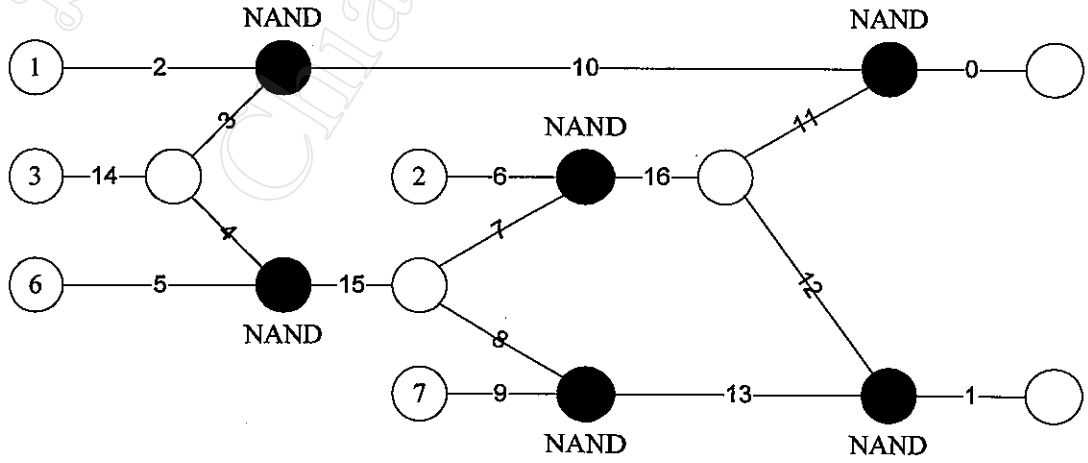
รูปที่ 4.10 โครงสร้างข้อมูลการยุบรวมจุดเสีย

4.5 การให้กำเนิดรายการลดทอน

รายการลดทอนจะถูกเก็บไว้ในไฟล์กับในหน่วยความจำโดยใช้โครงสร้างข้อมูลแบบรายเดี่ยวเป็นตัวเก็บอยู่ในหน่วยวามจำ และเก็บในแฟ้มข้อมูล มีโครงสร้างดังต่อไปนี้



รูปที่ 4.11 แสดงการเก็บรายการลดทอนในโครงสร้างข้อมูลแบบรายการเดี่ยว



รูปที่ 4.12 วงจร c17.vm ตัวอย่างการสร้างรายการลดทอน

พิจารณารูปที่ 4.12 เป็นวงจร c17.vm แสดงตัวอย่างรายการลดทอนที่ได้มาจากการทำการแพร่ค่าตรรกไปข้างหน้ากับ Contrapositive Law (บทที่ 3) ซึ่งมีรายการลดทอนดังนี้

Full Reduction List Generation

D (s-a-0)

B (s-a-1)

line[0]=0

11~0 : 11~D : 11~B : 16~B : 16~D : 16~0 : 10~B : 10~D : 10~0 : 0~B : 0~D : 0~1 :

line[0]=1

0~B : 0~D : 0~0 :

line[0]=D

10~0 : 10~D : 11~0 : 11~D : 16~D : 16~0 : 0~B : 0~1 : 0~0 :

line[0]=B

10~0 : 10~B : 11~0 : 11~B : 16~B : 16~0 : 0~D : 0~1 : 0~0 :

line[1]=0

13~0 : 13~D : 13~B : 16~0 : 16~D : 16~B : 12~B : 12~D : 12~0 : 1~B : 1~D : 1~1 :

line[1]=1

15~D : 15~0 : 15~B : 1~B : 1~D : 1~0 :

line[1]=D

12~0 : 12~D : 16~D : 16~0 : 13~0 : 13~D : 15~0 : 15~B : 1~B : 1~1 : 1~0 :

line[1]=B

12~0 : 12~B : 16~B : 16~0 : 13~0 : 13~B : 15~0 : 15~D : 1~D : 1~1 : 1~0 :

line[2]=0

10~B : 10~D : 10~0 : 2~B : 2~D : 2~1 :

line[2]=1

2~B : 2~D : 2~0 :

line[2]=D

10~D : 10~0 : 2~B : 2~1 : 2~0 :

line[2]=B

10~B : 10~0 : 2~D : 2~1 : 2~0 :

line[3]=0

10~B : 10~D : 10~0 : 14~1 : 14~D : 14~B : 3~B : 3~D : 3~1 :

line[3]=1

14~0 : 14~D : 14~B : 3~B : 3~D : 3~0 :

line[3]=D

10~D : 10~0 : 14~0 : 14~1 : 14~B : 3~B : 3~1 : 3~0 :

line[3]=B

10~B : 10~0 : 14~0 : 14~1 : 14~D : 3~D : 3~1 : 3~0 :

line[4]=0

15~B : 15~D : 15~0 : 8~B : 8~D : 8~0 : 7~B : 7~D : 7~0 : 14~1 : 14~D : 14~B : 4~B :

4~D : 4~1 :

line[4]=1

14~0 : 14~D : 14~B : 4~B : 4~D : 4~0 :

line[4]=D

15~D : 15~0 : 8~D : 8~0 : 7~D : 7~0 : 14~0 : 14~1 : 14~B : 4~B : 4~1 : 4~0 :

line[4]=B

15~B : 15~0 : 8~B : 8~0 : 7~B : 7~0 : 14~0 : 14~1 : 14~D : 4~D : 4~1 : 4~0 :

line[5]=0

15~B : 15~D : 15~0 : 8~B : 8~D : 8~0 : 7~B : 7~D : 7~0 : 5~B : 5~D : 5~1 :

line[5]=1

5~B : 5~D : 5~0 :

line[5]=D

15~D : 15~0 : 8~D : 8~0 : 7~D : 7~0 : 5~B : 5~1 : 5~0 :

line[5]=B

15~B : 15~0 : 8~B : 8~0 : 7~B : 7~0 : 5~D : 5~1 : 5~0 :

line[6]=0

16~B : 16~D : 16~0 : 12~B : 12~D : 12~0 : 11~B : 11~D : 11~0 : 6~B : 6~D : 6~1 :

line[6]=1

6~B : 6~D : 6~0 :

line[6]=D

16~D : 16~0 : 12~D : 12~0 : 11~D : 11~0 : 6~B : 6~1 : 6~0 :

line[6]=B

16~B : 16~0 : 12~B : 12~0 : 11~B : 11~0 : 6~D : 6~1 : 6~0 :

line[7]=0

16~B : 16~D : 16~0 : 12~B : 12~D : 12~0 : 11~B : 11~D : 11~0 : 15~1 : 15~D : 15~B : 4~0 : 4~D : 4~B : 14~B : 14~D : 14~0 : 5~B : 5~D : 5~0 : 7~B : 7~D : 7~1 :

line[7]=1

15~0 : 15~D : 15~B : 7~B : 7~D : 7~0 :

line[7]=D

16~D : 16~0 : 12~D : 12~0 : 11~D : 11~0 : 15~0 : 15~1 : 15~B : 5~D : 5~0 : 14~0 : 14~D : 4~D : 4~0 : 7~B : 7~1 : 7~0 :

line[7]=B

16~B : 16~0 : 12~B : 12~0 : 11~B : 11~0 : 15~0 : 15~1 : 15~D : 5~B : 5~0 : 14~0 : 14~B : 4~B : 4~0 : 7~D : 7~1 : 7~0 :

line[8]=0

13~B : 13~D : 13~0 : 15~1 : 15~D : 15~B : 4~0 : 4~D : 4~B : 14~B : 14~D : 14~0 : 5~B : 5~D : 5~0 : 8~B : 8~D : 8~1 :

line[8]=1

15~0 : 15~D : 15~B : 8~B : 8~D : 8~0 :

line[8]=D

13~D : 13~0 : 15~0 : 15~1 : 15~B : 5~D : 5~0 : 14~0 : 14~D : 4~D : 4~0 : 8~B : 8~1 : 8~0 :

line[8]=B

13~B : 13~0 : 15~0 : 15~1 : 15~D : 5~B : 5~0 : 14~0 : 14~B : 4~B : 4~0 : 8~D : 8~1 : 8~0 :

line[9]=0

13~B : 13~D : 13~0 : 9~B : 9~D : 9~1 :

line[9]=1

9~B : 9~D : 9~0 :

line[9]=D

13~D : 13~0 : 9~B : 9~1 : 9~0 :

line[9]=B

13~B : 13~0 : 9~D : 9~1 : 9~0 :

line[10]=0

0~B : 0~D : 0~0 : 3~0 : 3~D : 3~B : 14~B : 14~D : 14~0 : 2~B : 2~D : 2~0 : 10~B :

10~D : 10~1 :

line[10]=1

10~B : 10~D : 10~0 :

line[10]=D

0~D : 0~0 : 2~0 : 2~D : 3~0 : 3~D : 14~D : 14~0 : 10~B : 10~1 : 10~0 :

line[10]=B

0~B : 0~0 : 2~0 : 2~B : 3~0 : 3~B : 14~B : 14~0 : 10~D : 10~1 : 10~0 :

line[11]=0

0~B : 0~D : 0~0 : 16~1 : 16~D : 16~B : 6~0 : 6~D : 6~B : 15~0 : 15~D : 15~B : 7~B :

7~D : 7~0 : 11~B : 11~D : 11~1 :

line[11]=1

16~0 : 16~D : 16~B : 11~B : 11~D : 11~0 :

line[11]=D

0~D : 0~0 : 16~0 : 16~1 : 16~B : 15~0 : 15~D : 7~D : 7~0 : 6~D : 6~0 : 11~B : 11~1 :

11~0 :

line[11]=B

0~B : 0~0 : 16~0 : 16~1 : 16~D : 15~0 : 15~B : 7~B : 7~0 : 6~B : 6~0 : 11~D : 11~1 :

11~0 :

line[12]=0

1~B : 1~D : 1~0 : 16~1 : 16~D : 16~B : 6~0 : 6~D : 6~B : 15~0 : 15~D : 15~B : 7~B :

7~D : 7~0 : 12~B : 12~D : 12~1 :

line[12]=1

16~0 : 16~D : 16~B : 12~B : 12~D : 12~0 :

line[12]=D

1~D : 1~0 : 16~0 : 16~1 : 16~B : 15~0 : 15~D : 7~D : 7~0 : 6~D : 6~0 : 12~B : 12~1 : 12~0 :

line[12]=B

1~B : 1~0 : 16~0 : 16~1 : 16~D : 15~0 : 15~B : 7~B : 7~0 : 6~B : 6~0 : 12~D : 12~1 : 12~0 :

line[13]=0

1~B : 1~D : 1~0 : 9~0 : 9~D : 9~B : 15~0 : 15~D : 15~B : 8~B : 8~D : 8~0 : 13~B : 13~D : 13~1 :

line[13]=1

13~B : 13~D : 13~0 :

line[13]=D

1~D : 1~0 : 8~0 : 8~D : 15~D : 15~0 : 9~0 : 9~D : 13~B : 13~1 : 13~0 :

line[13]=B

1~B : 1~0 : 8~0 : 8~B : 15~B : 15~0 : 9~0 : 9~B : 13~D : 13~1 : 13~0 :

line[14]=0

15~B : 15~D : 15~0 : 10~B : 10~D : 10~0 : 8~B : 8~D : 8~0 : 7~B : 7~D : 7~0 : 4~B : 4~D : 4~1 : 3~B : 3~D : 3~1 : 14~B : 14~D : 14~1 :

line[14]=1

4~B : 4~D : 4~0 : 3~B : 3~D : 3~0 : 14~B : 14~D : 14~0 :

line[14]=D

15~D : 15~0 : 10~D : 10~0 : 8~D : 8~0 : 7~D : 7~0 : 4~B : 4~1 : 4~0 : 3~B : 3~1 : 3~0 : 14~B : 14~1 : 14~0 :

line[14]=B

15~B : 15~0 : 10~B : 10~0 : 8~B : 8~0 : 7~B : 7~0 : 4~D : 4~1 : 4~0 : 3~D : 3~1 : 3~0 : 14~D : 14~1 : 14~0 :

line[15]=0

16~B : 16~D : 16~0 : 13~B : 13~D : 13~0 : 12~B : 12~D : 12~0 : 11~B : 11~D : 11~0 : 8~B : 8~D : 8~1 : 7~B : 7~D : 7~1 : 1~B : 1~D : 1~1 : 5~0 : 5~D : 5~B : 14~0 : 14~D : 14~B : 4~B : 4~D : 4~0 : 15~B : 15~D : 15~1 :

line[15]=1

8~B : 8~D : 8~0 : 7~B : 7~D : 7~0 : 15~B : 15~D : 15~0 :

line[15]=D

16~D : 16~0 : 13~D : 13~0 : 12~D : 12~0 : 11~D : 11~0 : 8~B : 8~1 : 8~0 : 7~B : 7~1 : 7~0 : 1~B : 1~1 : 4~0 : 4~D : 14~D : 14~0 : 5~0 : 5~D : 15~B : 15~1 : 15~0 :

line[15]=B

16~B : 16~0 : 13~B : 13~0 : 12~B : 12~0 : 11~B : 11~0 : 8~D : 8~1 : 8~0 : 7~D : 7~1 : 7~0 : 1~D : 1~1 : 4~0 : 4~B : 14~B : 14~0 : 5~0 : 5~B : 15~D : 15~1 : 15~0 :

line[16]=0

12~B : 12~D : 12~1 : 11~B : 11~D : 11~1 : 1~B : 1~D : 1~0 : 0~B : 0~D : 0~0 : 7~0 : 7~D : 7~B : 15~B : 15~D : 15~0 : 6~B : 6~D : 6~0 : 16~B : 16~D : 16~1 :

line[16]=1

12~B : 12~D : 12~0 : 11~B : 11~D : 11~0 : 16~B : 16~D : 16~0 :

line[16]=D

12~B : 12~1 : 12~0 : 11~B : 11~1 : 11~0 : 1~D : 1~0 : 0~D : 0~0 : 6~0 : 6~D : 7~0 : 7~D : 15~D : 15~0 : 16~B : 16~1 : 16~0 :

line[16]=B

12~D : 12~1 : 12~0 : 11~D : 11~1 : 11~0 : 1~B : 1~0 : 0~B : 0~0 : 6~0 : 6~B : 7~0 : 7~B : 15~B : 15~0 : 16~D : 16~1 : 16~0 :

4.6 การให้กำเนิดรูปแบบสัญญาณทดสอบ

การให้กำเนิดรูปแบบสัญญาณทดสอบมีขั้นตอนดังต่อไปนี้

1.) เลือกผู้แทนจุดเสียจากรายการจุดเสีย

2.) แพร่ค่าตรรกไปข้างหน้าหรือการทำภาพฟังก์ชัน

3.) เลือกค่าตรรกจุดเสีย ($\overline{D}, D$) ที่ขั้วขาออกหลักของวงจร

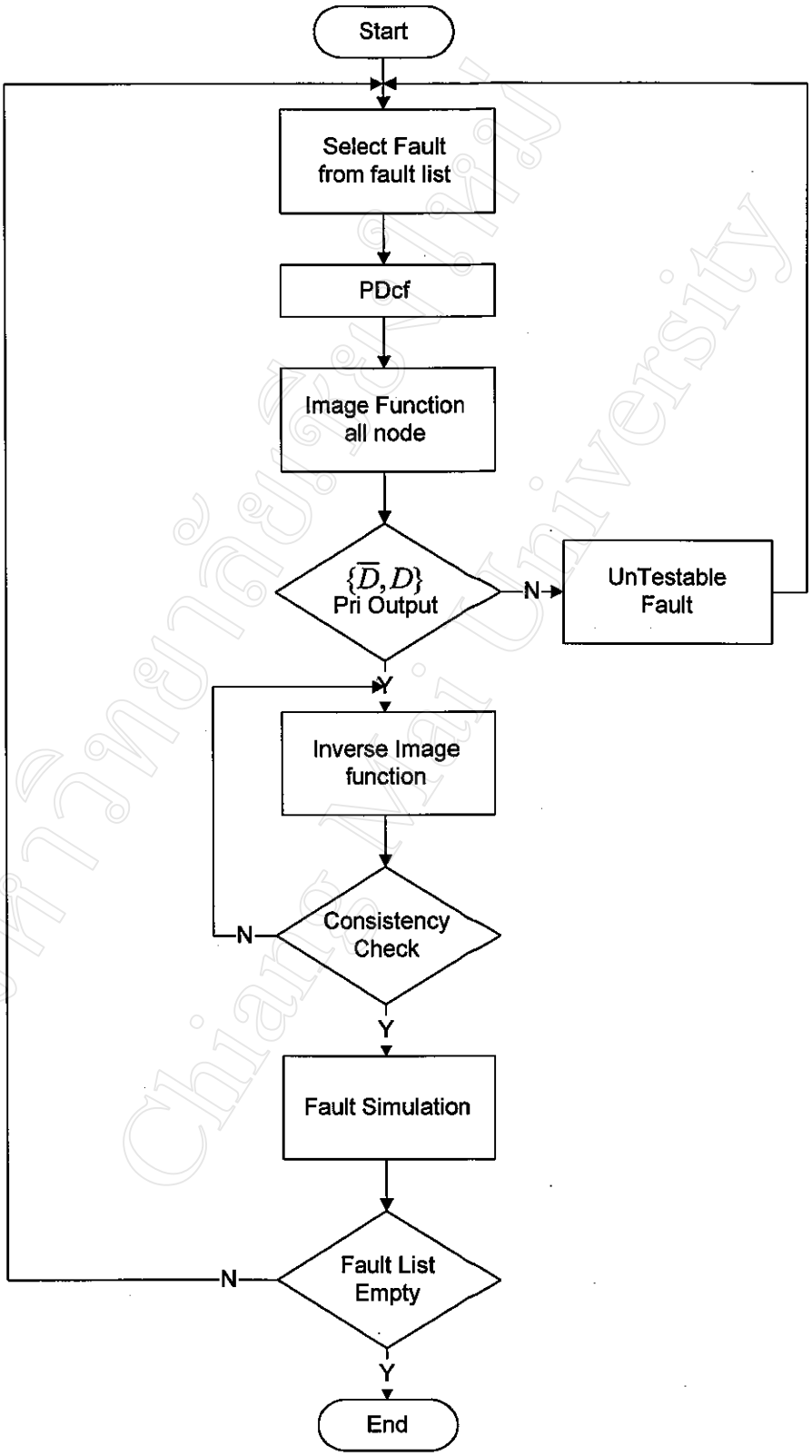
4.) ทำส่วนกลับภาพฟังก์ชันจากขั้วขาออกหลักไปจนถึงขั้วขาเข้าหลักของวงจร

5.) ทำการตรวจเช็คความถูกต้องในระหว่างการทำส่วนกลับภาพฟังก์ชัน

6.) จำลองการทำงานจุดเสีย

7.) ตัดจุดเสียออกจากรายการจุดเสียถ้าจุดเสียจุดนั้นสามารถทดสอบได้

8.) กลับไปทำข้อ 1 ใหม่จนกว่าจุดเสียถูกทดสอบทั้งหมด



รูปที่ 4.13 ผังงานการให้กำเนิดรูปแบบสัญญาณทดสอบโดยอัตโนมัติ