บทที่ 5

โปรแกรมต้นแบบสำหรับวิเคราะห์การทำงานของวงจรตรรกะ

5.1 บทนำ

หลักการต่าง ๆ ที่ได้กล่าวไปแล้วนั้น ได้นำมาใช้ในการพัฒนาโปรแกรมต้นแบบสำหรับวิเคราะห์การทำงานของวงจรตรรกะ โปรแกรมที่พัฒนาขึ้นให้ชื่อว่า LAP ซึ่งย่อมาจาก Logic Analysis Program โปรแกรมนี้ได้ทำการพัฒนาขึ้นมาโดยใช้งานกับไมโครคอมพิวเตอร์ IBM PC/XT ซึ่งใช้งานกันอย่างแพร่หลายมีอุปกรณ์และมีโปรแกรมช่วยในการพัฒนาซอฟต์แวร์เป็นจำนวนมาก เราได้เลือกใช้ภาษา C ในการพัฒนาโปรแกรมเนื่องจากมีข้อดีหลายประการ เช่น

- ภาษา C มีขีดความสามารถสูง และเป็นภาษาโครงสร้าง ซึ่งเหมาะสมในการพัฒนาโปรแกรมขนาดใหญ่
- เป็นภาษาที่นิยมกันอย่างแพร่หลายในปัจจุบันสำหรับใช้ในการพัฒนาโปรแกรมสำเร็จรูปและมีการใช้งานบนคอมพิวเตอร์แทบทุกชนิด
- มีความเป็นมาตรฐานมาก ทำให้สามารถพัฒนาต่อไปบนคอมพิวเตอร์อื่นซึ่งมีความสามารถสูงขึ้นได้ในอนาคตโดยเฉพาะคอมพิวเตอร์ซึ่งใช้ระบบปฏิบัติการ UNIX

โปรแกรมต้นแบบสำหรับวิเคราะห์การทำงานของวงจรตรรกะ LAP ที่พัฒนาขึ้นมานี้มีขีดความสามารถดังต่อไปนี้

1. สามารถวิเคราะห์การทำงานของ วงจรตรรกะซึ่งประกอบไปด้วยเกตต่อไปนี้ คือ AND,OR,NAND,NOR,INVERTOR,XOR,DELAY GATE,ACTIVE LOW ENABLE TRISTATE BUFFER,FLIP-FLOP โดยจะต่อกันแบบใดก็ได้ จำนวนเท่าใดก็ได้ จำกัดที่หน่วยความจำของเครื่องที่ใช้

2. การทำงานใช้สถานะทางตรรกะ 4 สถานะคือ HIGH (H), LOW(L) HIGH IMPEDANCE (Z) และ UNKNOWN (X)

3. สามารถกำหนดสถานะเริ่มต้นและค่าเวลาประวิงของเกตแต่ละตัวได้อย่างอิสระ

4. การแสดงผล ทำในรูปกราฟฟิค สามารถแสดงผลเป็นแผนภาพสัญญาณเวลา ณ.จุดใดในวงจร โดยสามารถแสดงได้พร้อมกันครั้งละ 15 โนด

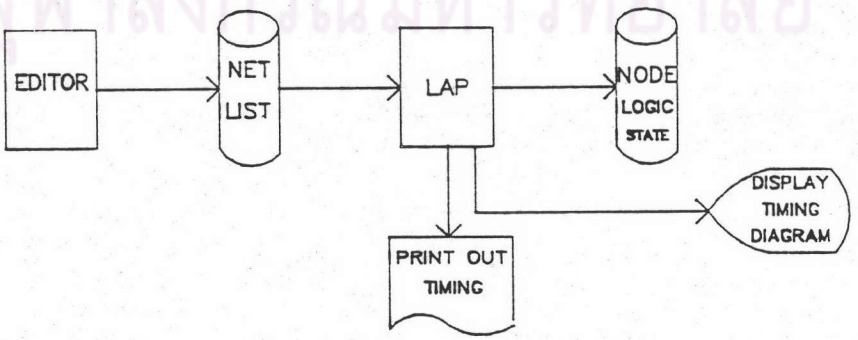
5. สามารถกำหนดส่วนของวงจรให้อยู่ในรูปแมคโครเพื่อนำมาใช้งานหลายครั้ง หรือนำไปใช้กับโปรแกรมอื่นได้

ในบทนี้จะเริ่มด้วยการกล่าวถึง รูปแบบและการใช้งานของโปรแกรม จากนั้นจะกล่าวถึงข้อมูลทางเทคนิคภายในของโปรแกรมเพื่อให้สามารถใช้งานได้อย่างมีประสิทธิภาพ โดยอาศัยความเข้าใจถึงกลไกการทำงานของโปรแกรม

5.2 ขั้นตอนในการใช้งานโปรแกรม LAP

ในการใช้งานโปรแกรม LAP นั้น จะเริ่มด้วยการสร้าง netlist ไฟล์ ซึ่งเป็นไฟล์ที่จะบอกถึงลักษณะของวงจรที่จะวิเคราะห์ เกตที่ใช้ในวงจรและการเชื่อมต่อของเกตหรือแหล่งกำเนิดสัญญาณต่าง ๆ ในวงจร netlist ไฟล์ นี้เป็นไฟล์แบบ text ปกติ ซึ่งสามารถสร้างขึ้นโดยอาศัยโปรแกรม text editor ใดก็ได้รูปแบบของ netlist ไฟล์นี้จะกล่าวถึงในตอนต่อไป

เมื่อเริ่มทำการวิเคราะห์ใช้คำสั่งใน LAP ทำการเรียกไฟล์ netlist (มี extension .LAP) เข้ามา จากนั้นแล้วจะสามารถใช้คำสั่งภายในต่าง ๆเพื่อทำการวิเคราะห์การทำงาน ตรวจสอบค่าตรรกะที่โนดต่างๆ แสดงสัญญาณเวลา ขั้นตอนดังกล่าวแสดงในแผนภาพในรูปที่ 5.1



รูปที่ 5.1 โครงสร้างการใช้งานโปรแกรม LAP

การแก้ไขวงจรสามารถทำได้โดยการแก้ไข netlist ด้วยโปรแกรม editor หลังจากนั้นก็เก็บเข้าในแผ่นจานแม่เหล็กเพื่อนำมาทำการวิเคราะห์การทำงานด้วยโปรแกรม LAP ต่อไป

5.3 คำสั่งของโปรแกรม LAP

ในการใช้งานโปรแกรมวิเคราะห์การทำงานของวงจรตรรก LAP นั้นสามารถแบ่งคำสั่งออกเป็น 3 กลุ่ม ด้วยกัน คือ

1. คำสั่งป้อนข้อมูล เป็นคำสั่งซึ่งใส่ไว้ใน netlist เพื่อกำหนดลักษณะต่างๆ ของวงจร เกตที่ใช้ การเชื่อมต่อของเกตและโนดต่าง ๆ การกำหนดเงื่อนไขเริ่มต้นของวงจร สัญญาณอินพุตต่าง ๆ ที่เข้ามายังวงจร

2. คำสั่งควบคุมการวิเคราะห์การทำงาน เป็นชุดคำสั่งที่ควบคุมการทำงานของโปรแกรมวิเคราะห์การทำงานของวงจรตรรกะ คำสั่งควบคุมการแสดงผลการตั้งค่าและแสดงค่าพารามิเตอร์ต่างๆ ของวงจร

3. คำสั่งทั่วไป เช่น คำสั่งแสดงรายการแฟ้มข้อมูลบนจานแม่เหล็ก คำสั่งลบแฟ้มข้อมูลเป็นต้น

5.4 คำสั่งป้อนข้อมูล

ในการกล่าวถึงคำสั่งป้อนข้อมูลนั้นจะกล่าวรวมถึงรูปแบบของ netlist(รูปที่ 5.2) โปรแกรม LAP ใช้รวมไปด้วย ข้อมูลที่ต้องใช้ในการวิเคราะห์การทำงานของวงจรตรรกะนั้น ประกอบไปด้วยรายการของเกตในวงจร ชนิดของเกตแต่ละตัว การเชื่อมต่อระหว่างโนดต่างๆ การกำหนดโมเดลของเกตจากผู้ใช้ สัญญาณภายนอกที่ป้อนเข้าวงจรและการกำหนดแมคโครซึ่งจะทำการอธิบายไปตามลำดับ ดังนี้

```
*------------------------------
*  Example of netlist in LAP
*------------------------------
74GS04   MODEL INV   1
74GS00   MODEL AND2  1
*---------------------------
*  Macro definition example
*---------------------------
EDGE    MACRO
IN   OUT
N2   N3   N4
G1   74GS04    IN   N2
G2   74GS04    N2   N3
G3   74GS04    N3   N4
G4   74GS00    IN   N4   OUT
ENDM
*---------------------
*  Signal  definition
*---------------------
CLK   CLOCK   INPUT   60   100
SIG1 SIGNAL S1    3
10    H
23    L
22    H
*--------------
*  Circuit
*--------------
GATE1   NAND2    A       B       C
GATE2   74GS04   B       X
B1      BUS      B       X       D
B1      BUS      10
ELM1    EDGE    INPUT   OUT1
*--------------------
* INITIAL CONDITION
*--------------------
A    NODE   H
B    NODE   H
C    NODE   L
D    NODE   X
```

รูปที่ 5.2 ตัวอย่าง netlist ซึ่งใช้ใน LAP

5.4.1 <u>คำสั่งกำหนดเกต</u>

เป็นคำสั่ง ซึ่งกำหนด ชื่อ ชนิด และการเชื่อมต่อของเกตภายในวงจร โดยมีรูปแบบทั่วไปดังนี้ คือ

<ELEMENT NAME> < MODEL > < I/O PIN LIST >

ELEMENT NAME คือ ชื่อของเกต มีความยาวไม่เกิน 6 ตัวอักษร

MODEL คือ ชนิดของเกตตรรกะ นั้น

I/O PINLIST คือ รายการของ อินพุต-เอาต์พุต โนด ของเกต ซึ่งจำนวน และลำดับตามหน้าที่จะขึ้นกับชนิดของเกตนั้น ๆ และเป็นไปดังรูปที่ 5.3

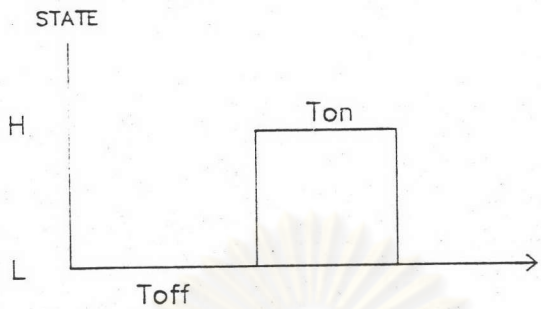
```
AND[2-8]   <IN 1..8>     <OUT>
OR[2-8]    <IN 1..8>     <OUT>
NAND[2-8]  <IN 1..8>     <OUT>
NOR[2-8]   <IN 1..8>     <OUT>
XOR        <IN1> <IN2>   <OUT>
INV        <IN>  <OUT>
BUFF       <IN>  <OUT>
DRVL       <IN>  <OUT>
BUS        <IN 1..20>    <OUT>
TFFC  <T> <CLK> <PR>  <CLR> <Q>   <Qbar>
DFFC  <D> <CLK> <PR>  <CLR> <Q>   <Qbar>
JKFC  <J> <K>   <CLK> <PR>  <CLR> <Q>   <Qbar>
TFF   <T> <CLK> <PR>  <CLR> <Q>   <Qbar>
DFF   <D> <CLK> <PR>  <CLR> <Q>   <Qbar>
JKF   <J> <K>   <CLK> <PR>  <CLR> <Q>   <Qbar>
TFNC  <T> <CLK> <PR>  <CLR> <Q>   <Qbar>
DFNC  <D> <CLK> <PR>  <CLR> <Q>   <Qbar>
JKNC  <J> <K>   <CLK> <PR>  <CLR> <Q>   <Qbar>
TFN   <T> <CLK> <PR>  <CLR> <Q>   <Qbar>
DFN   <D> <CLK> <PR>  <CLR> <Q>   <Qbar>
JKN   <J> <K>   <CLK> <PR>  <CLR> <Q>   <Qbar>
```

รูปที่ 5.3 แสดงถึงรายการของเกต และขาอินพุตเอาต์พุต

5.4.2 <u>คำสั่งกำหนดแหล่งกำเนิดสัญญาณ</u>

แหล่งกำเนิดสัญญาณที่ใช้ในโปรแกรมแบ่งออกเป็น 2 ชนิด คือ สัญญาณนาฬิกา (CLOCK) และสัญญาณทั่วไป (SIGNAL)

สัญญาณนาฬิกาหมายถึง พัลส์ที่มีคาบเวลาแน่นอนซึ่งแบ่งเป็นช่วงที่มีค่าตรรกะ H (Ton) และค่าตรรกะ L (Toff) ดังรูปที่ 5.4



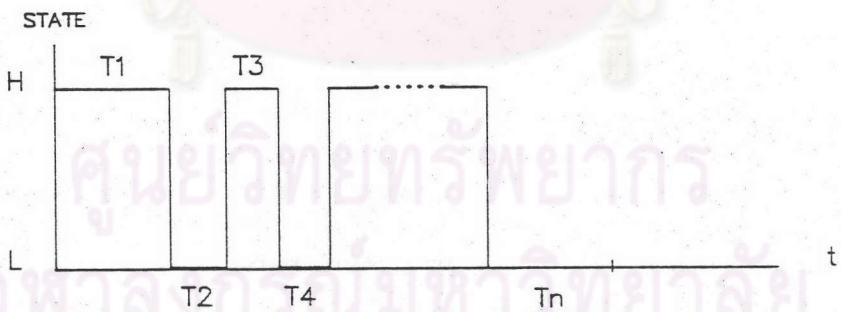
รูปที่ 5.4 สัญญาณนาฬิกา

รูปแบบของคำสั่งจะเป็นดังนี้

< CLOCK NAME > CLOCK < OUTPUT NODE > < TON> <TOFF>

โดยที่ Ton และ Toff มีหน่วยเป็น MRT (ในโปรแกรมจะให้ 1 MRT คือ 1 nSec)

สัญญาณทั่วไป คือ สัญญาณซึ่งมีลักษณะการเปลี่ยนแปลงในแบบใด ๆก็ได้ไม่เป็นรายคาบแน่นอน



รูปที่ 5.5 สัญญาณแบบทั่วไป

ในการกำหนดสัญญาณแบบทั่วไปจะแบ่งรูปคลื่นเป็นช่วง ๆ ด้วยกัน ถ้าสัญญาณแบ่งเป็น n ช่วงดังรูปที่ 5.5 แล้ว รูปแบบคำสั่งจะเป็นดังนี้

< SIGNALNAME > SIGNAL <OUTPUT NODE> <NO. OF PERIOD = N>

< T_1 > < $STATE_1$ >

< T_2 > < $STATE_2$ >

< T_N > < $STATE_n$ >

คาบเวลา T_1, T_2, ...T_n มีหน่วยเป็น MRT STATE มี 4 สถานะคือ H,L,X,Z

5.4.3 คำสั่งกำหนดโมเดลของผู้ใช้

กรณีที่ต้องการกำหนดค่าพารามิเตอร์บางอย่างขึ้นเอง เช่น ค่าเวลาประวิงให้กับเกตบางตัวนั้น สามารถทำได้โดยใช้คำสั่ง MODEL ซึ่งมีรูปแบบ ดังนี้

< USER MODEL NAME > MODEL < BUILDIN MODELNAME > < DELAY >

เช่น กรณีที่มีเกตบางตัวมี ค่าเวลาประวิง 5 MRT สามารถทำได้โดยกำหนด โมเดลใหม่ขึ้นมาใช้ สมมุติว่าชื่อ 74AS00

74AS00 MODEL NAND2 5

จากนั้นสามารถทำการเรียกใช้ได้ เช่น เดียวกับโมเดลภายในปกติ

5.4.4 คำสั่งกำหนดเงื่อนไขเริ่มต้น

โดยปกติเมื่อโปรแกรม LAP อ่านวงจรเข้าไป เก็บในหน่วยความจำนั้นจะตั้งค่าสถานะทางตรรกะของทุกโนดเป็น LOW แต่ผู้ใช้สามารถทำการกำหนดสถานะใหม่ให้กับโนดต่างๆได้ โดยการใส่ไว้ใน netlist หรือกำหนดค่าเมื่ออยู่ในโปรแกรม LAP โดยใช้คำสั่ง SET NODE ที่จะกล่าวต่อไป

การกำหนดสถานะของโนดใน netlist นั้นทำได้โดยใส่คำสั่งรูปแบบนี้

<NODE NAME> NODE <STATE>

STATE คือ สถานะที่กำหนดให้โนด

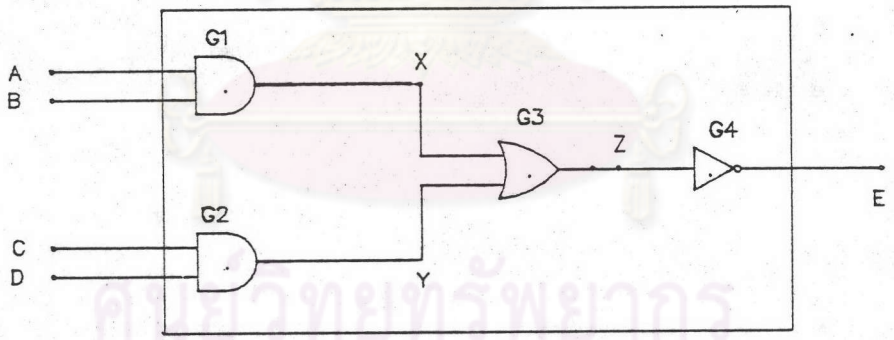
NODE NAME คือ ชื่อโนดที่ต้องการกำหนดสถานะ

นอกจากนั้นยังมีคำสั่งซึ่งกำหนดเวลาประวิงแก่บัส

<BUS NAME> BUS <TIME DELAY>

5.4.5 <u>คำสั่งเกี่ยวกับแมคโคร</u>

การใช้งานแมคโครแบ่งออกเป็น 2 ขั้น คือการกำหนดแมคโครและการเรียกใช้ซึ่งอธิบายได้ดังตัวอย่างต่อไปนี้



รูปที่ 5.6 แมคโครของเกต AOI

หากต้องการกำหนดวงจรในรูป 5.6 เป็นแมคโคร มีชื่อ AOI ข้อมูลกำหนดแมคโครใน netlist จะเป็นดังนี้

```
AOI       MACRO
A  B  C  D  E
X  Y  Z
G1  AND2    A    B    X
G2  AND2    C    D    Y
G3  OR2     X    Y    Z
G4  INV     Z    E
ENDM
```

บรรทัดแรกเป็นชื่อของแมคโครตามด้วยคำสั่ง MACRO บรรทัดที่ 2 เป็นรายการโนดภายนอก (EXTERNAL NODE) บรรทัดที่ 3 เป็นรายการโนดภายใน(INTERNAL NODE) จากนั้นก็เป็นรายการของเกตภายใน แมคโคร ซึ่งอยู่ในรูปแบบเดียวกับการ netlist ธรรมดา ปิดท้ายด้วยคำสั่ง ENDM เป็นการจบการกำหนดแมคโคร ในการเรียกใช้แมคโครทำการเรียกเสมือนว่าแมคโคร เป็นโมเดลธรรมดา เพียงแต่การเชื่อมต่อของจุดอินพุต เอาท์พุต จะเรียงตามบรรทัดที่กำหนดโนดภายนอก ตัวอย่างเช่นการเรียกใช้แมคโครตามตัวอย่างจะทำได้ดังนี้

```
GATE1       AOI       P Q R S T
```

จากตัวอย่างนี้เป็นการกำหนดให้ GATE1 เป็นเกตชนิด AOI มีขาอินพุตคือ P,Q,R และ S และขาเอาท์พุต T

5.5 คำสั่งควบคุมการวิเคราะห์การทำงาน

เมื่อเรียกใช้งานโปรแกรม LAP แล้วจะปรากฏข้อความดังรูปที่ 5.7

```
MicroLAP 1.0  : Microcomputer Logic Analysis Program
Developed By  : Putchong Uthayopas,B815754
                Department of Electrical Engineering
                Faculty of Engineering
                Chulalongkorn University
                March 1988
memory available : 413916 Byte
```

รูป 5.7 การแสดงผลเมื่อเริ่มโปรแกรม LAP

เมื่อ พรอมพ์ LAP> ถูกแสดงออกมาแสดงว่าในขณะนี้โปรแกรม LAP พร้อมที่จะทำการรับคำสั่งจากผู้ใช้ซึ่งเป็นคำสั่งควบคุมการวิเคราะห์การทำงานและคำสั่งทั่วไป ซึ่งในหัวข้อนี้จะกล่าวถึงคำสั่งควบคุมก่อน ทีละคำสั่งตามลำดับ

5.5.1 คำสั่ง LOAD

คำสั่งนี้เป็นคำสั่งให้อ่าน netlist เข้ามายังโปรแกรมเพื่อทำการวิเคราะห์การทำงาน รูปแบบคำสั่งเป็นดังนี้

LOAD < FILE NAME >

แฟ้มข้อมูลที่จะเรียกเข้ามาจะต้องลงท้ายด้วย "LAP" เสมอเช่น CIRCUIT.LAP

5.5.2 คำสั่ง LIST

คำสั่ง LIST เป็นคำสั่งที่แสดงวงจรและข้อมูลต่าง ๆ ที่เก็บอยู่ภายในวงจร ภายในคำสั่ง LIST เองแบ่งเป็นคำสั่งย่อยหลายคำสั่งดังต่อไปนี้

LIST	แสดงวงจรทั้งหมดในหน่วยความจำ
LIST SIGNAL	แสดงเฉพาะแหล่งกำเนิดสัญญาณ
LIST GATE	แสดงสถานะทางตรรกะของโนดต่าง ๆ
LIST FIO	แสดงรายการ fan in,fan out ของแต่ละโนด
LIST MODEL	แสดงโมเดลของเกตทั้งหมดรวมทั้งโมเดลที่กำหนดขึ้นโดยผู้ใช้จากคำสั่ง MODEL
LIST MACRO	แสดงแมคโครที่ผู้ใช้กำหนดขึ้น ซึ่งเก็บอยู่ในหน่วยความจำในปัจจุบัน

5.5.3 คำสั่ง SET

เป็นคำสั่งสำหรับตั้งค่าพารามิเตอร์ต่าง ๆ ในขณะนี้มีเพียงคำสั่งเดียวคือ SET NODE ซึ่งเป็นคำสั่งให้ทำการตั้งค่าตรรกะที่โนดใด ๆ ในวงจร โดยมีรูปแบบดังนี้

SET NODE < NODENAME > < STATE >

หากส่วนที่เป็นชื่อ โนดใช้คำว่า ALL จะหมายถึงการตั้งค่าตรรกะให้กับทุกโนดในวงจร

5.5.4 คำสั่ง SIM

เป็นคำสั่งให้เครื่องทำการวิเคราะห์การทำงานของวงจรที่เก็บไว้ในหน่วยความจำ ซึ่งรายละเอียดของการทำงานจะกล่าวถึงในตอนต่อ ๆ ไป

5.5.5 คำสั่ง DISPLAY

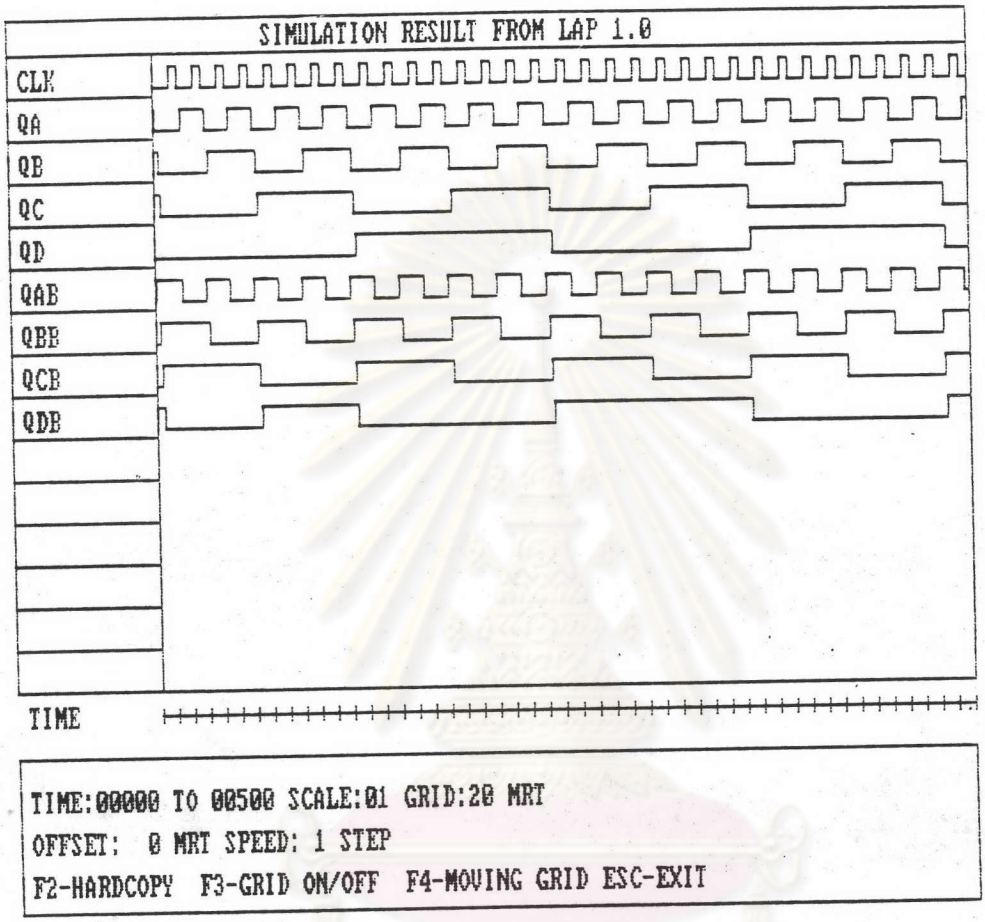
คำสั่งนี้เป็นคำสั่งซึ่งควบคุมการแสดงแผนภาพสัญญาณเวลาซึ่งได้มาจากการวิเคราะห์การทำงานของวงจรด้วยคำสั่ง SIM คำสั่งนี้แบ่งเป็นหลายคำสั่งย่อยด้วยกันคือ

- DISPLAY NODE เป็นคำสั่งที่ทำหน้าที่กำหนดว่าจะทำการแสดงผลที่โนดใดบ้างมากที่สุด 15 โนดพร้อมกันดังตัวอย่างในรูปที่ 5.8

```
LAP>
LAP> disp node
NO.  of node to display (less than 15) : 9
< 1>  NAME : CLK
< 2>  NAME : QA
< 3>  NAME : QB
< 4>  NAME : QC
< 5>  NAME : QD
< 6>  NAME : QAB
< 7>  NAME : QBB
< 8>  NAME : QCB
< 9>  NAME : QDB
LAP>
```

รูปที่ 5.8 การใช้งานคำสั่ง DISPLAY NODE

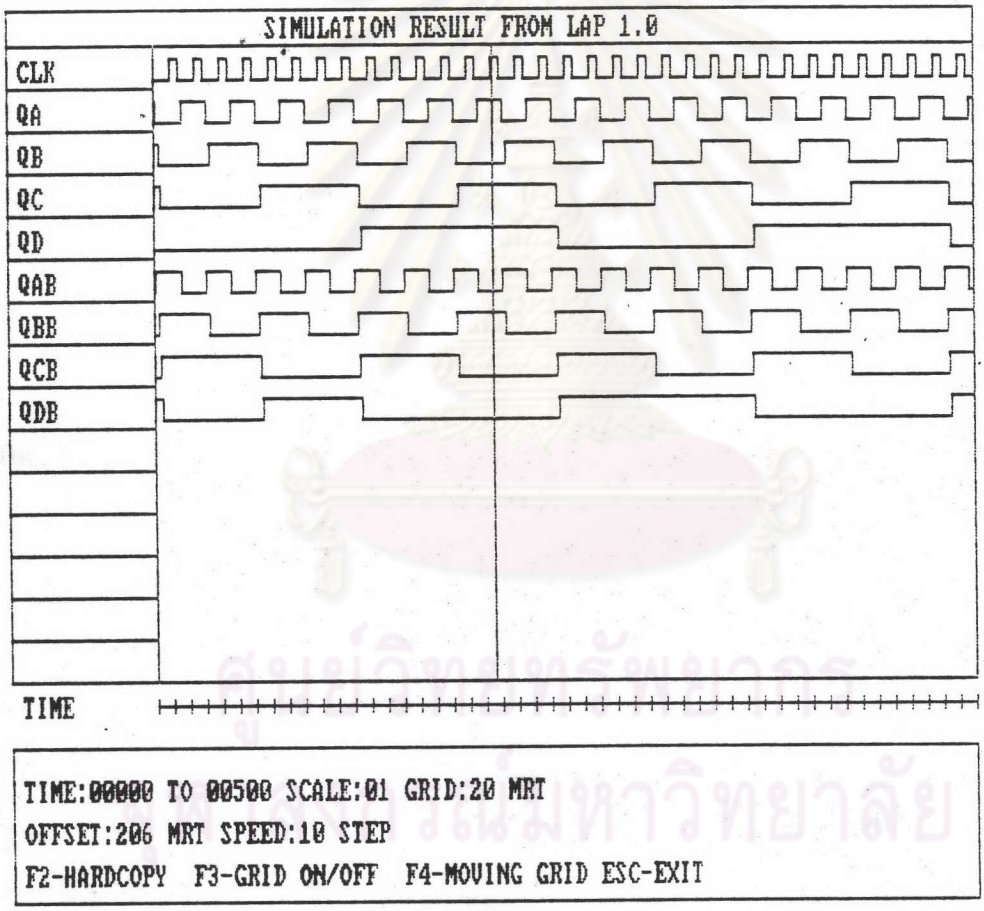
- DISPLAY เป็นคำสั่งให้ทำการแสดงผลแผนภาพสัญญาณเวลาในรูปกราฟฟิคบนจอภาพ (เป็นจอภาพแบบโมโนโครมใช้การ์ด แสดงผลแบบ Hercules)

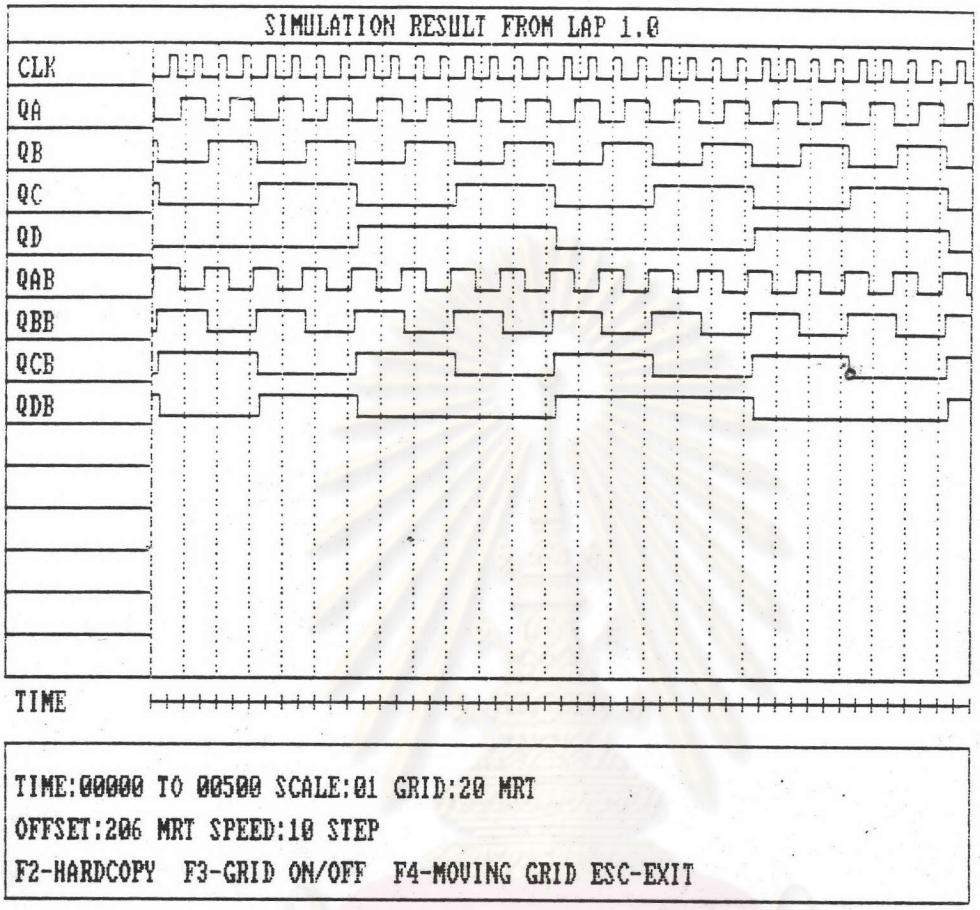


รูปที่ 5.9 การแสดงผลโดยคำสั่ง DISPLAY

รูปที่ 5.9 เป็นการแสดงแผนภาพสัญญาณเวลาของแต่ละโนดที่ถูกเลือกโดยใช้คำสั่ง DISPLAY NODE ส่วนทางด้านล่างเป็นการแสดงข้อมูลต่าง ๆ ที่เกี่ยวข้องกับการแสดงผล เช่น เวลาที่แสดงผลเริ่มจากเวลาเท่ากับ 0 ถึง 500 การแสดงผลจะแสดงได้ 500 จุด SCALE เป็นการบอกอัตราขยายว่าเป็นเท่าไร หาก SCALE มีค่ามากขึ้น การแสดงผลจะได้น้อยจุดลงแต่จะทำการเปรียบเทียบผลได้ง่ายขึ้น ส่วน GRID เป็นการบอกความถี่ของการแสดงเส้นกริดบอกเวลา การแสดงเส้นกริดทำได้โดยกดปุ่ม F3 บนแป้นพิมพ์ ซึ่งจะแสดงกริดบอกเวลาแบบคงที่ หากกด F4 จะมีกริดชนิดเคลื่อนที่ปรากฏบนจอภาพซึ่งสามารถเลื่อนได้โดยใช้ปุ่มลูกศรขวา (--->) และซ้าย (<---) บนแป้นพิมพ์เวลาที่กริดเคลื่อนที่ปรากฏจะแสดงอยู่หลังคำว่า OFFSET ในช่วงแสดงสถานะส่วน SPEED หมายถึงความเร็วที่กริดเคลื่อน

ที่จะเคลื่อนไป เมื่อกดปุ่มลูกศร 1 ครั้ง การเพิ่มลดความเร็วทำโดยใช้ปุ่ม "+" และ " - "
การพิมพ์รูปภาพบนจอภาพที่ปรากฏอยู่นั้นสามารถพิมพ์ออกทางเครื่องพิมพ์ได้โดยใช้ปุ่ม F2
ดังรูปที่ 5.10





รูปที่ 5.10 เส้นกริดแบบต่างๆ ในการแสดงผล

- DISPLAY SET เป็นคำสั่งตั้งค่าพารามิเตอร์บางประการก่อนการแสดงผล เช่น ค่า SCALE เวลาที่เริ่มต้นการแสดงผลและความถี่ของเส้นกริดคงที่

```
LAP>disp set
** Display parameter setting **
Display scale[01]>8
Display start at step <0-1024>[00000]>220
Display grid every[020]>1
```

รูปที่ 5.11 การใช้คำสั่ง DISPLAY SET

5.5.6 คำสั่ง SAVE

คำสั่งนี้จะทำการเก็บค่าโนด และสถานะปัจจุบันลงไปในแผ่นจานแม่เหล็ก เพื่อนำไปใช้งานต่อไป โดยเก็บไว้ในรูปไฟล์ที่สามารถนำไปรวมใน netlist ได้ทันที คำสั่ง SAVE มีรูปแบบดังนี้

SAVE < FILENAME >

ผลที่ได้แสดงในรูปที่ 5.12

```
*--------------------*
*     Logic state    *
*--------------------*
     INA     NODE      L
      PR     NODE      H
     CLR     NODE      H
      QA     NODE      L
     QAB     NODE      H
     INB     NODE      L
      QB     NODE      H
     QBB     NODE      L
      QC     NODE      L
     QCB     NODE      H
      QD     NODE      L
     QDB     NODE      H
     R01     NODE      L
     R02     NODE      H
     CLK     NODE      L
```

รูปที่ 5.12 ผลที่ได้จากคำสั่ง SAVE

5.5.7 คำสั่ง NEW

เป็นคำสั่งลบข้อมูลทั้งหมดในหน่วยความจำเพื่อทำงานกับวงจรใหม่ต่อไป

5.5.8 คำสั่ง QUIT

เป็นคำสั่งออกจากโปรแกรม LAP กลับเข้าสู่ระบบปฏิบัติการ MS-DOS

5.6 คำสั่งทั่วไป

คำสั่งทั่วไปเป็นคำสั่งที่นอกเหนือไปจากคำสั่งเกี่ยวกับการวิเคราะห์การทำงานทั้งนี้เพื่อให้การใช้งาน โปรแกรมเป็นไปได้ง่ายขึ้น คำสั่ง เหล่านี้มีรายละเอียดต่อไปนี้

5.6.1 คำสั่ง DIR

เป็นคำสั่งแสดงรายการข้อมูลบนแผ่นจานแม่เหล็ก หากสั่ง DIR อย่างเดียว จะแสดงไฟล์ที่มีนามสกุล .LAP หากตามด้วย pathname ก็จะแสดงรายการไฟล์นั้นออกมา

```
LAP>dir
DIRECTORY of *.LAP

  SIMCOMB .LAP   7493    .LAP   7493A   .LAP   BUSTEST .LAP   EDGB    .LAP
  EDGE1   .LAP   JKEDGB  .LAP   NETEXAM .LAP   NEWOSC  .LAP   OSC     .LAP
  OSC1    .LAP   ZTEST   .LAP   SIMCOMB1.LAP   DELTA   .LAP   KKL     .LAP
         15 FILE(S)    6830080 Bytes available on disk
LAP>
```

รูปที่ 5.13 การทำงานของคำสั่ง DIR

5.6.2 คำสั่ง DEL

เป็นคำสั่งลบไฟล์บนแผ่นจานแม่เหล็ก มีรูปแบบดังนี้

DEL < FILENAME >

5.6.3 คำสั่ง COPY

เป็นคำสั่งที่รูปแบบดังนี้

COPY < FILE1 > < FILE2 >

คำสั่งนี้จะทำสำเนาข้อมูลจากไฟล์ FILE1 ไปยังไฟล์ FILE2

5.6.4 คำสั่ง HELP

เป็นคำสั่งให้พิมพ์ข้อความในไฟล์ LAP.HLP บนจอภาพซึ่งไฟล์นี้จะเก็บข้อความวิธีใช้งานโปรแกรม LAP ไว้

5.6.5 คำสั่ง MEM

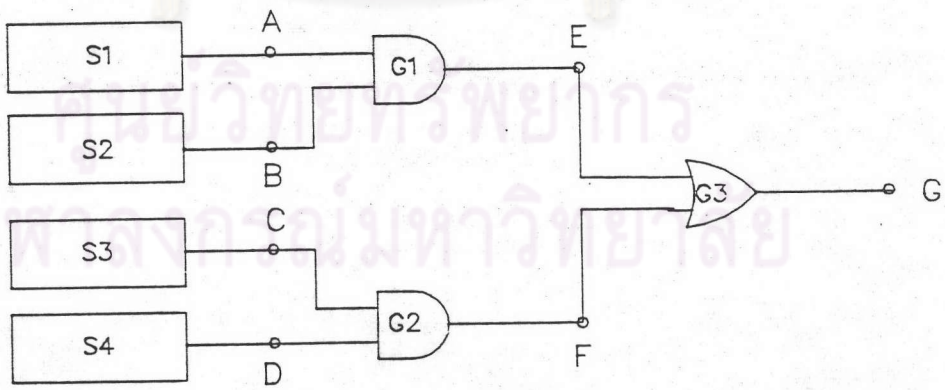
คำสั่งนี้จะทำการแสดงขนาดของหน่วยความจำซึ่งเหลืออยู่ให้ใช้งาน

5.7 โครงสร้างข้อมูลและการทำงานของโปรแกรม LAP

หลักจากที่ได้กล่าวถึงการใช้งานและคำสั่งต่าง ๆ ในโปรแกรม LAP หัวข้อนี้จะกล่าวถึงการทำงานภายในของโปรแกรมและโครงสร้างข้อมูลที่ใช้ในการทำงานแต่ละขั้นตอนดังต่อไปนี้

5.7.1 การเก็บข้อมูลวงจรในหน่วยความจำ

เมื่อพิจารณาวงจรที่ทำการวิเคราะห์การทำงานจะพบว่า ข้อมูลที่จะต้องเก็บประกอบไปด้วย เกต, โนด, แหล่งกำเนิดสัญญาณ ดังตัวอย่างในรูปที่ 5.14

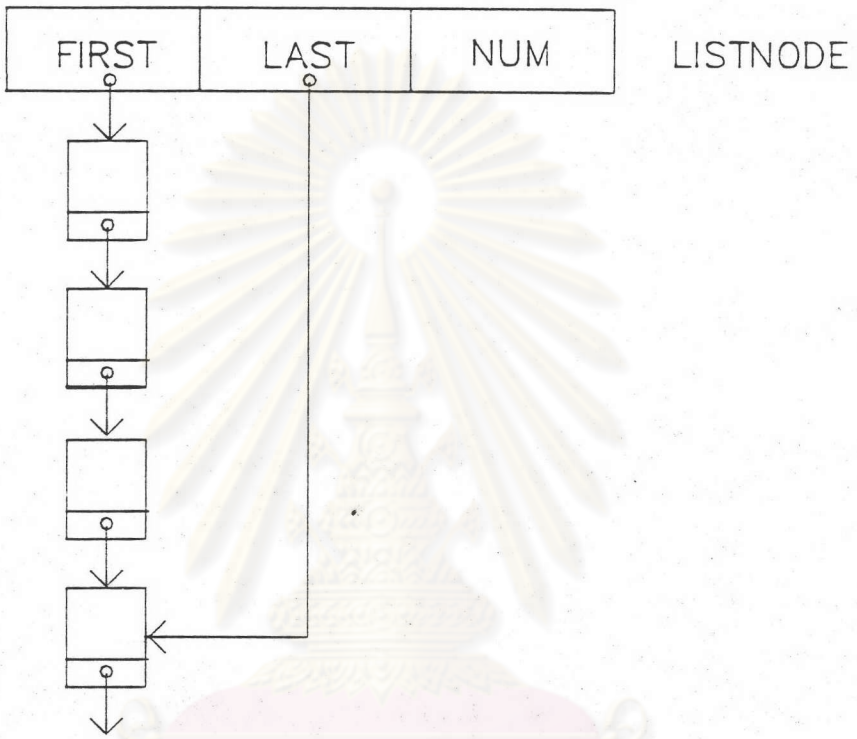


รูปที่ 5.14 ตัวอย่างวงจรที่วิเคราะห์การทำงานได้

ในตอนต้นนี้ จะกล่าวถึงการเก็บข้อมูลของวงจรที่ไม่มีแมคโครก่อนส่วนในเรื่องการทำงานของแมคโครนั้นจะกล่าวถึงโดยละเอียดอีกครั้งหนึ่ง

การเก็บข้อมูลเหล่านี้ทำโดยการใช้ linked list ทั้งนี้เนื่องจากการเพิ่มเติม

ข้อมูลทำได้ง่าย และสามารถรับข้อมูลซึ่งไม่ทราบจำนวณได้ ลักษณะของ linked list ที่ใช้จะเป็นดังรูปที่ 5.15



รูปที่ 5.15 linked list ที่ใช้งานใน LAP

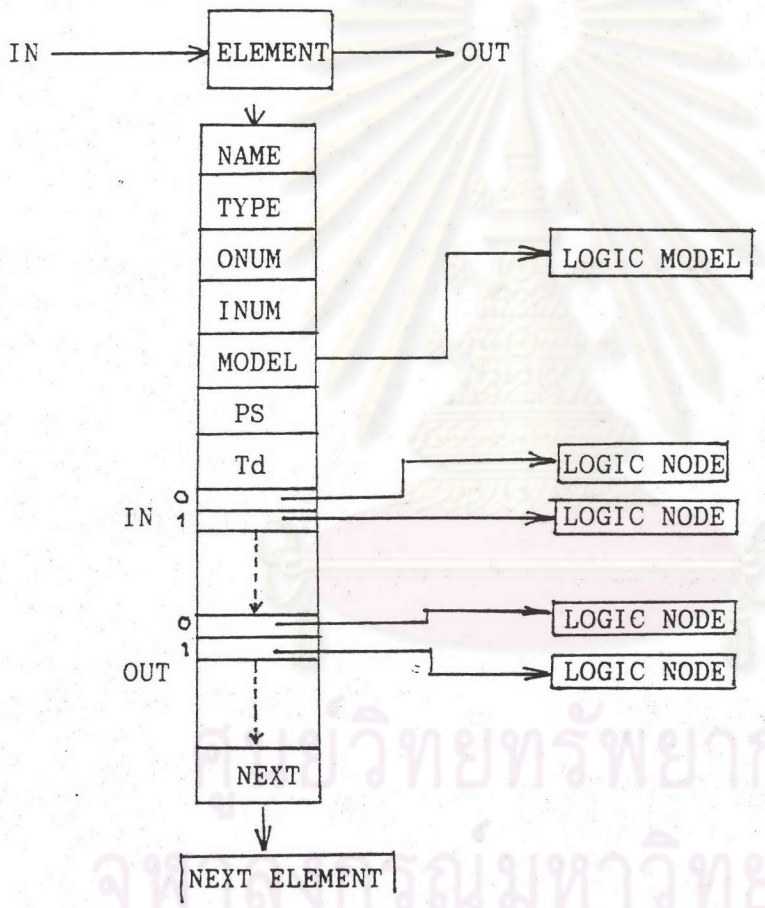
linked list ที่ใช้จะประกอบด้วยโครงสร้างข้อมูลแบบ structure ชื่อ LISTNODE ประกอบด้วย 3 ส่วน คือ first เป็นตัวชี้ไปยังข้อมูลตัวแรกใน linked list last เป็นตัวชี้ไปยังข้อมูลตัวสุดท้ายใน linked list ส่วน num คือ จำนวนข้อมูลในlinked list สำหรับเกต , โนด, แหล่งกำเนิดสัญญาณจะเก็บใน linked list ลักษณะนี้โดยใน next ทำหน้าที่เป็นตัวชี้ไปยังข้อมูลตัวถัดไป (ตัวสุดท้ายตัวชี้จะเป็นค่าคงที่ NULL ในภาษา C ซึ่งทำให้รู้ว่าเป็นข้อมูลสุดท้ายในlinked list) กรณี linked list ว่าง first และ last จะมีค่าเป็น NULL ส่วน num จะมีค่าเป็น 0

ในกรณีของโมเดลของเกตนั้นจะต่างไปจากที่กล่าวมา เนื่องจากเราแบ่งโมเดลออกเป็น 2 แบบ แบบแรกคือ <u>โมเดลที่สร้างไว้ในโปรแกรม</u> (build in model) ส่วนแบบที่ 2 คือ โมเดลที่ผู้ใช้กำหนดมาจากแบบแรกแต่เปลี่ยน พารามิเตอร์ โดยใช้คำสั่ง

MODEL เรียกว่า <u>โมเดลกำหนดโดยผู้ใช้</u> (user define model) ในโมเดลแบบสร้างไว้ในโปรแกรมนั้น การเก็บข้อมูลทำในลักษณะของ array ส่วนโมเดลกำหนดโดยผู้ใช้เก็บข้อมูลในแบบ linked list เช่นที่ได้กล่าวไปแล้ว

5.7.2 <u>โครงสร้างข้อมูลของ เกต โนด และแหล่งกำเนิดสัญญาณ</u>

ในหัวข้อนี้เราจะกล่าวลงไปถึงรายละเอียดของโครงสร้างข้อมูลและหน้าที่ของแต่ละฟิลด์ในโครงสร้างข้อมูลตามลำดับไป สำหรับเกตจะมีโครงสร้างข้อมูลดังรูปที่ 5.16

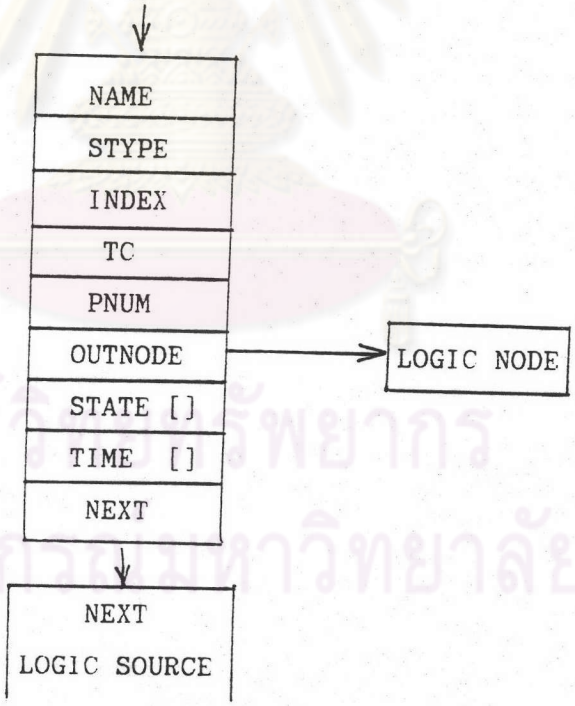


รูปที่ 5.16 โครงสร้างข้อมูลของเกต

- NAME เก็บชื่อเกต
- TYPE เก็บชนิดของเกตมี 2 ชนิด คือ XTERN สำหรับเกตปกติและ INTERN สำหรับแมคโคร

- ONUM,INUM เป็นจำนวนของอินพุต เอาต์พุตของเกตนั้น เป็นตัวเลขที่ได้มาจากโมเดลของเกต
- MODEL เป็นตัวชี้ไปยังโมเดลของเกตนั้น
- PS คือ สภาวะปัจจุบัน ใช้สำหรับวงจรแบบลำดับ
- IN,OUT เป็นข้อมูลแบบ array เก็บตัวชี้ไปยังอินพุตโนด และเอาท์พุตโนด ซึ่งเกตนั้นต่ออยู่ IN มีขนาด MAXIPIN ออกมีขนาด MAXOPIN ซึ่งเป็นค่าคงที่กำหนดไว้ในโปรแกรม ใน LAP MAXIPIN มีค่า 8 ส่วน MAXOPIN มีค่า 2
- NEXT เป็นตัวชี้ไปยังข้อมูลเกตตัวต่อไปใน linked list

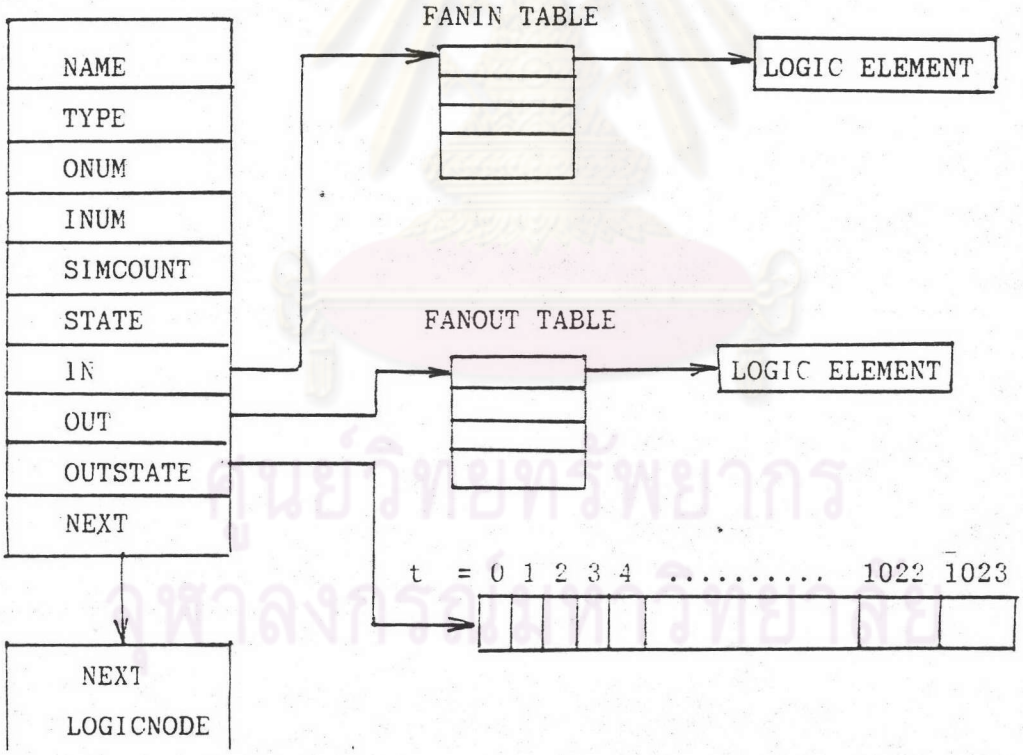
สำหรับแหล่งกำเนิดสัญญาณ จะมีโครงสร้างข้อมูลดังรูปที่ 5.17



รูปที่ 5.17 โครงสร้างข้อมูลของแหล่งกำเนิดสัญญาณ

แต่ละฟิลด์ในโครงสร้างข้อมูลของแหล่งกำเนิดสัญญาณ จะมีรายละเอียด ดังนี้

- NAME เก็บชื่อของแหล่งกำเนิดสัญญาณนั้น
- STYPE เก็บชนิดของแหล่งกำเนิดสัญญาณซึ่งแบ่งเป็น 2 ชนิด คือ LSRC สำหรับสัญญาณทั่วไป และ CLKSRC สำหรับสัญญาณนาฬิกา
- INDEX,TC,PNUM ใช้ในการทำงานในขณะกำลังทำการวิเคราะห์การทำงาน จะกล่าวถึงต่อไปในตอนที่เกี่ยวกับการวิเคราะห์การทำงาน
- OUTNODE เป็นตัวชี้ไปยังเอาต์พุตโนดของแหล่งกำเนิดสัญญาณนี้
- STATE, TIME เก็บค่าสถานะทางตรรกะ และเวลาที่มีค่าสถานะนี้ ตัวแปรทั้งสองเป็น array มีขนาด MAXSBREAK ซึ่งเป็นตัวคงที่ในโปรแกรม มีค่าเท่ากับ 20
- NEXT เป็นตัวชี้ไปยังข้อมูลของแหล่งกำเนิดสัญญาณตัวต่อไปใน link listed



รูปที่ 5.18 โครงสร้างข้อมูลของโนด

โครงสร้างข้อมูลของโนดในวงจรมีรายละเอียด ดังรูปที่ 5.18

- NAME เก็บชื่อของโนดนั้น
- TYPE ชนิดของโนดซึ่งมี 2 ชนิด คือ XTERN สำหรับ โนดธรรมดา และ

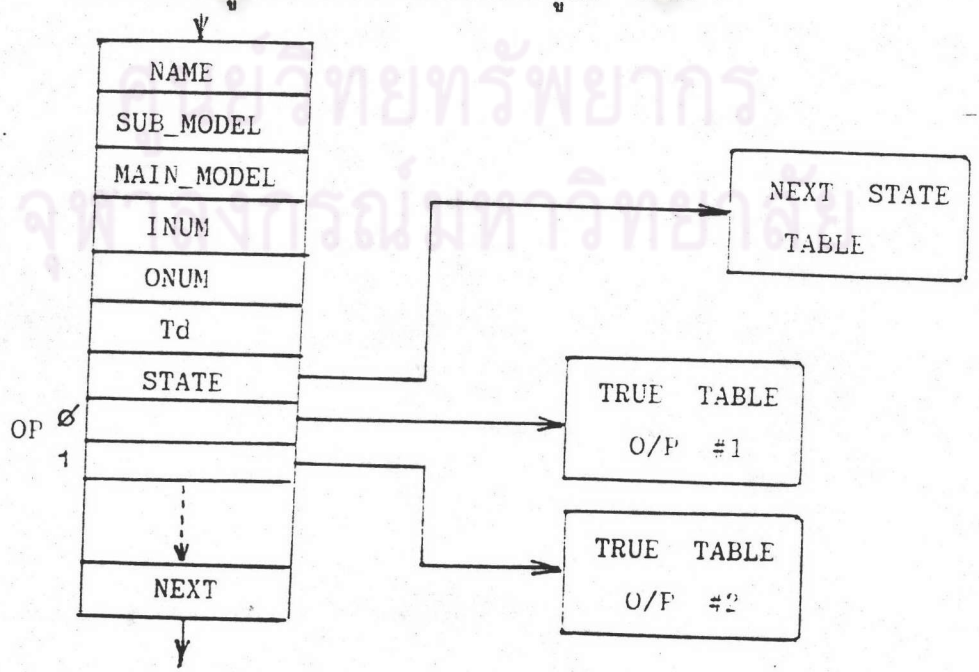
INTERN สำหรับ แมคโครโนด

- ONUM, INUM จำนวน fan in, fan out ของโนดนั้นซึ่งค่ามากสุด กำหนดไว้โดยค่าคงที่ MAXFANIN, MAXFANOUT
- SIMCOUNT เป็นค่าที่ใช้นับจำนวนข้อมูลที่ถูกจัดไว้ใน time queue สำหรับโนดนั้น ใช้เพื่อตรวจสอบสัญญาณยอดแหลม
- STATE สถานะทางตรรกะในขณะปัจจุบันของโนดนั้น
- IN เป็นตัวชี้ไปยังตาราง fan in ซึ่งจะเก็บตัวชี้ไปยังเกตที่ต่อกับตัวโนดนี้
- OUT เป็นตัวชี้ไปยังตาราง fan out ซึ่เก็บค่าตัวชี้ไปยัง เกตซึ่งถูกขับโดยโนดนี้
- OUTSTATE ชี้ไปยังตารางซึ่งใช้เก็บค่าที่ได้จากการวิเคราะห์การทำงานของวงจร มีขนาด 1024 ไบท์
- NEXT ชี้ไปยังโนดถัดไปใน linked list

การใช้งานของข้อมูแต่ละฟิลด์ในโครงสร้างข้อมูลจะกล่าวถึงในหัวข้อต่อไป

5.7.3 <u>โครงสร้างข้อมูลของโมเดล</u>

โครงสร้างข้อมูลของโมเดลจะมีลักษณะดังรูปที่ 5.19



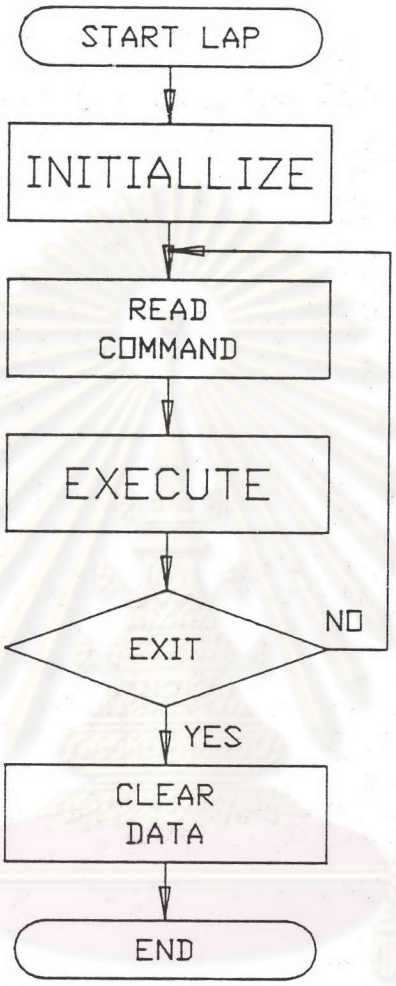
รูปที่ 5.19 โครงสร้างข้อมูลของโมเดล

- NAME ชื่อของโมเดล
- SUB_MODEL คือ ชนิดย่อยของโมเดลซึ่งใช้ในการพิจารณาว่าจะส่งการทำงานไปยังโปรแกรมย่อยใด มีด้วยกัน 10 แบบคือ AND ,OR, NAND, NOR, XOR, BUFF, TFF, DFF, JKFF, DRIVER
- MAIN_MODEL ใช้แยกชนิดของโมเดลหลัก ซึ่งแบ่งออกเป็น 4 แบบ คือ
  - SEQN สำหรับวงจรแบบลำดับ
  - COMB สำหรับวงจรแบบจัดหมู่
  - FLPF สำหรับวงจรแบบลำดับเฉพาะคือฟลิปฟลอป
  - BUS สำหรับเกตบัส
- INUM, ONUM คือ จำนวนขั้วเข้าและออกของเกต
- Td คือ เวลาประวิงแผ่กระจายของเกตนั้น
- STATE คือ ตัวชี้ไปยังตารางซึ่งเก็บค่าสถานะถัดไปใช้เฉพาะวงจรแบบลำดับเท่านั้น
- OP[] เป็นข้อมูลแบบ array ซึ่งเก็บค่าตัวชี้ไปยังจุดเริ่มต้นของตารางค่าความจริงของขั้วออกแต่ละขั้ว
- NEXT เป็นตัวชี้ไปยังโมเดลต่อไปใน linked list

โมเดลซึ่งสร้างอยู่ในโปรแกรมมีจำนวน 45 โมเดลจะมีการเก็บในลักษณะของ array ส่วนโมเดลซึ่งผู้ใช้กำหนดเพิ่มโดยใช้คำสั่งทาง MODEL นั้นจะเก็บไว้ในลักษณะของ linked list เช่นเดียวกับเกตและโนดในวงจร

5.8 การทำงานของโปรแกรม LAP

การทำงานของโปรแกรม LAP นั้นเริ่มจากส่วนหลักซึ่งทำหน้าที่รับคำสั่งจากผู้ใช้โดยขึ้นข้อความ LAP> เมื่อผู้ใช้ป้อนคำสั่งก็จะตีความและทำงานตามคำสั่งซึ่งการทำงานของโปรแกรมหลักจะมีลักษณะตามแผนภาพ ในรูป 5.20



รูปที่ 5.20 การทำงานของโปรแกรมหลัก

โปรมแกรมหลักจะประกอบไปด้วยการ Initialize ก่อนเริ่มการทำงานซึ่งจะเตรียมการหลายสิ่ง เช่น เตรียมโครงสร้างข้อมูลแบบ linked list ที่จะเก็บเกตและโนดต่างๆไว้ จองเนื้อที่สำหรับโนดอ้างอิง คือ Vcc และ Gnd เนื่องจากทั้งสองโนดนี้ จะต้องมี fan out มากเป็นพิเศษ จึงตรียมไว้โดยมีค่า MAXREFOUT = 500 นอกจากนั้นยังตั้งค่าให้ตัวแปรที่ใช้ร่วมกันในโปรแกรมด้วย

เมื่อจบการ initialize แล้วโปรแกรมจะอ่านคำสั่งจากผู้ใช้ทีละบรรทัดและแยกออกเป็นส่วนๆส่งให้โปรแกรมส่วน execute ซึ่งโปรแกรมในส่วนนี้จะทำการตีความคำสั่งซึ่งแบ่งออก ได้เป็น 3 ประเภทด้วยกันคือ

ก) คำสั่งอ่านเขียนข้อมูลจากแผ่นจานแม่เหล็ก

ข) คำสั่งควบคุมการวิเคราะห์การทำงาน

ค) คำสั่งทั่วไป

การทำงานจะเป็นไปในลักษณะดังกล่าว จนกว่าจะมีคำสั่ง QUIT หรือ EXIT ก็จะทำการล้างข้อมูล และสิ่งต่างๆที่จองไว้ในระบบ พร้อมทั้งหยุดการทำงานของโปรแกรม

5.9 การอ่านข้อมูลเข้ามายังหน่วยความจำ

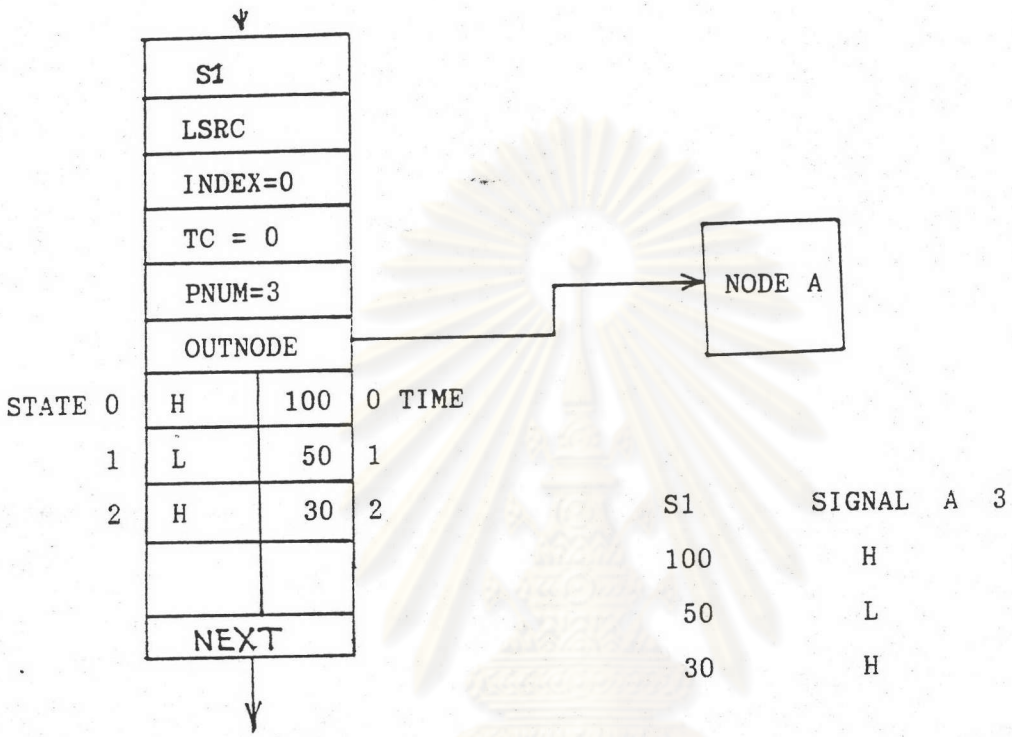
การทำงานของโปรแกรม LAP นั้น ต้องมีการอ่านข้อมูลจาก netlist เข้ามายังหน่วยความจำ ซึ่งจะทำงานเมื่อผู้ใช้ ใช้คำสั่ง LOAD ซึ่งได้กล่าวไปแล้วในหัวข้อก่อนหน้านี้ โปรแกรมจะอ่านข้อมูลจากแผ่นจานแม่เหล็กครั้งละบรรทัด และตรวจสอบบรรทัดที่เป็น COMMENT หรือ บรรทัดว่าง หากพบก็จะข้ามไป ถ้าเป็นข้อมูลที่เกี่ยวกับวงจร ก็จะส่งให้กับโปรแกรมย่อย ซึ่งจะแยกชนิดของเกต เพื่อเก็บลงยังโครงสร้างข้อมูล โดยแบ่งออกเป็น 7 ประเภท ดังนี้

1. SIGNAL
2. CLOCK
3. BUS
4. NODE
5. MODEL
6. MACRO
7. LOGIC ELEMENT (GATE)

การแยกประเภทเกต จะอาศัยคำสำคัญ (Reserve word) ซึ่งอยู่ใน คำที่ 2 ของบรรทัดนั้น การเก็บข้อมูลของแต่ละประเภทจะกล่าวถึงในหัวข้อต่อไป

5.9.1 การเก็บข้อมูลของแหล่งกำเนิดสัญญาณ

CLOCK และ SIGNAL จะใช้โครงสร้างข้อมูลเดียวกันในการเก็บข้อมูลโดยที่เรามองว่า CLOCK เป็น SIGNAL ชนิดหนึ่ง เพียงแต่ในการป้อนข้อมูล ใช้รูปแบบที่ง่ายกว่า ดังแสดงไว้ในรูปที่ 5.21 และ 5.22

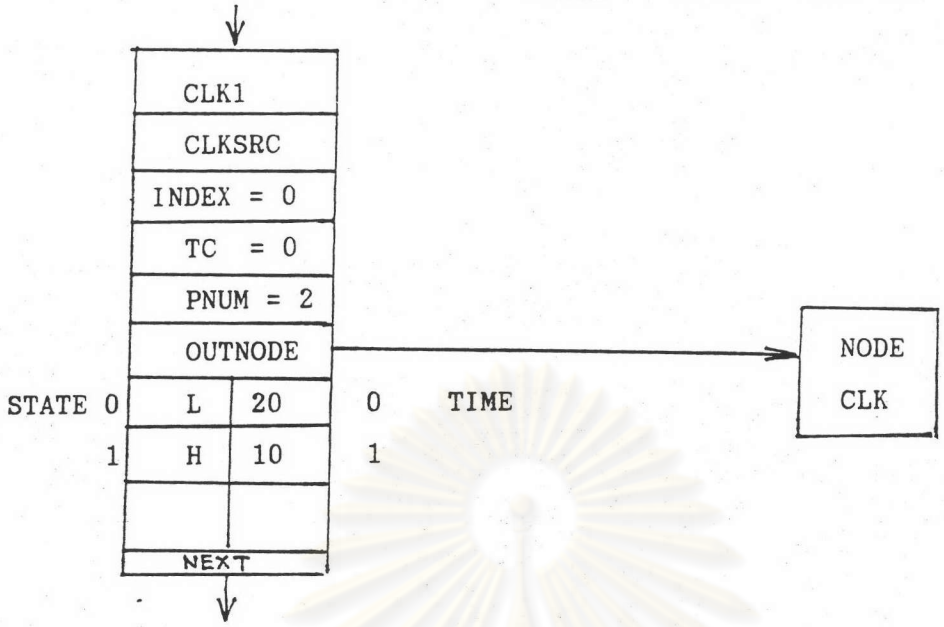


รูปที่ 5.21 การเก็บค่าแหล่งกำเนิดสัญญาณชนิด SIGNAL

เมื่อรับข้อมูลเข้ามาโปรแกรมจะทำการตรวจสอบความถูกต้องของข้อมูลบางประการ เช่น จำนวนของ Break point ต้องไม่มากกว่า MAXSBREAK เท่ากับ 20 จากนั้นจะจองเนื้อที่ในหน่วยความจำและเพิ่มเข้าไปใน linked list ที่เก็บข้อมูล ตั้งค่า OUTNODE ให้ชี้ไปยังโนดซึ่งถูกขับด้วยแหล่งกำเนิดสัญญาณนี้ จากนั้นจะอ่านข้อมูลของคาบเวลาเข้ามาเก็บดังภาพที่ 5.21 ตัวแปร INDEX และ TC นั้น จะถูกใช้ในช่วงวิเคราะห์การทำงานเพื่อนับคาบเวลาของสัญญาณ

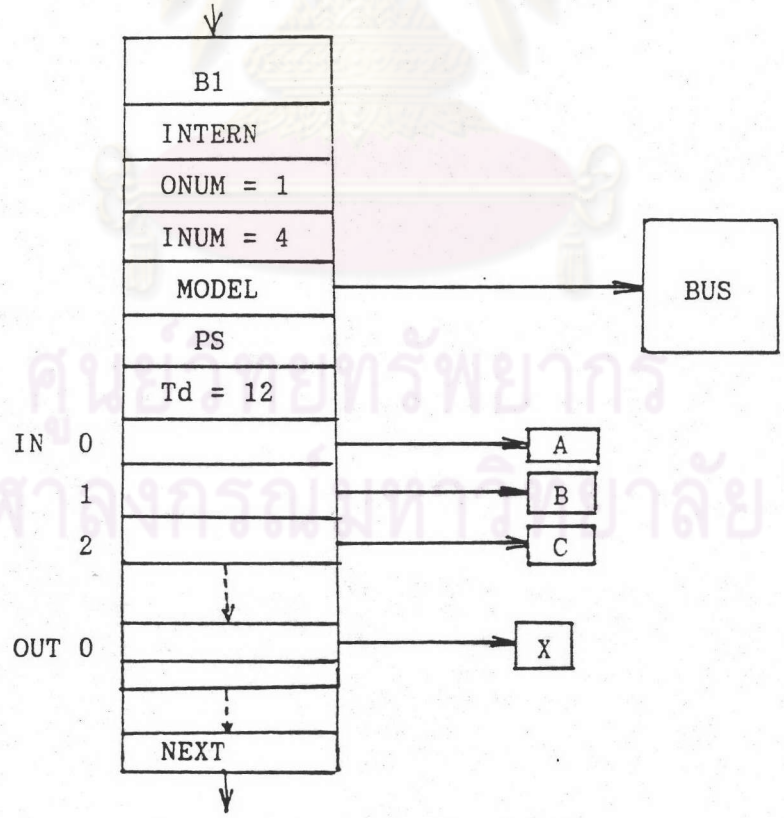
การเก็บ CLOCK นั้นจะเป็นลักษณะเดียวกับ SIGNAL เพียงแต่ค่า PNUM ซึ่งเป็นจำนวนของ break point จะมีค่าคงที่ = 2 ค่าเวลา T_{ON} และ T_{OFF} จะกลายเป็นจุด break point ค่าหนึ่งเท่านั้น

```
                                        CLK1    CLOCK   CLK  10  20
```



รูปที่ 5.22 การเก็บค่าของ CLOCK

```
B1      BUS     A     B     C     D     X
B1      BUS     .     12
```



รูปที่ 5.23 การเก็บข้อมูลของ BUS

5.9.2 การเก็บข้อมูลของบัส (BUS)

โครงสร้างข้อมูลของบัสจะมีลักษณะเช่นเดียวกับเกตธรรมดาทุกประการ (รูป 5.23) เนื่องจากในโปรแกรมจะมองบัสในลักษณะของเกตแบบจัดหมู่พิเศษชนิดหนึ่ง แต่ในการรับข้อมูลของบัสไม่สามารถกำหนดค่าเวลาประวิงพร้อมกันได้ เป็นเพราะเราไม่ทราบจำนวนขั้วเข้าที่แน่นอนของบัสขณะที่ป้อนข้อมูล การกำหนดค่าเวลาประวิงจึงต้องทำแยกออกไป

5.9.3 การเก็บข้อมูลของโนด

โดยปกติเมื่อมีการเรียกใช้ เกตต่างๆ แหล่งกำเนิดสัญญาณ หรือ บัส จะมีการตรวจสอบโนดที่เกตนั้นอ้างถึงอยู่แล้ว ในกรณีที่โนดนั้นยังไม่เคยปรากฏมาก่อนในวงจรจะมีการสร้างโนดใหม่ขึ้นมา แต่ในกรณีซึ่งมีโนดปรากฏอยู่ในวงจรอยู่แล้ว เกตนั้นก็จะได้รับตัวชี้ไปยังโนดซึ่งจะนำมาเก็บในโครงสร้างข้อมูลของเกตอีกทีหนึ่ง ในการสร้างโนดขึ้นมานั้นจะมีการแบ่งออกเป็น 2 ชนิดด้วยกัน คือ XTERN สำหรับโนดปกติ และ INTERN สำหรับโนดภายในของแมคโคร อย่างไรก็ดีโนดสามารถถูกตั้งค่าได้ก่อนการวิเคราะห์การทำงานโดยการสั่ง SET NODE หรือใส่ค่าเริ่มต้นไว้ใน netlist ดังที่ได้กล่าวไปแล้ว

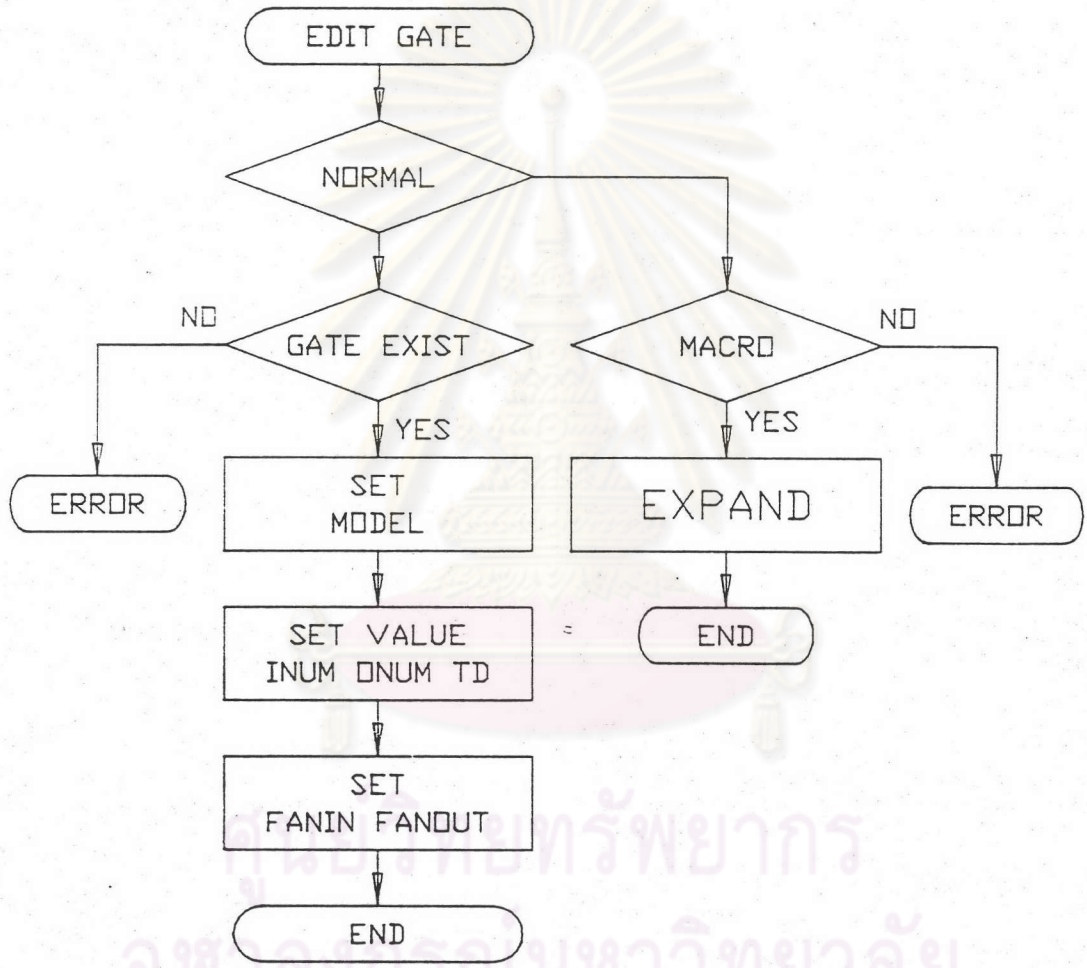
5.9.4 การเก็บข้อมูลของโมเดล

โมเดลของเกตจะมีการเก็บข้อมูลดังในรูป 5.19 ในหัวข้อย่อย 5.7.3 ซึ่งสิ่งที่สามารถแก้ไขได้อย่างเดียวในโมเดล คือ ค่าเวลาประวิงโดยโปรแกรมจะเปิดโอกาสให้ผู้ใช้กำหนดเวลาประวิงได้จากคำสั่งจาก MODEL ซึ่งการทำงานจะทำโดยการทำสำเนา MODEL ซึ่งมีอยู่แล้วในโปรแกรมออกมา โมเดลที่สร้างไว้ในโปรแกรมจะมีการเก็บเป็น array โมเดลใหม่จะถูกใส่ไว้ใน linked listed ที่สร้างขึ้นมาเพื่อเก็บโมเดลที่ผู้ใช้กำหนดขึ้น โดยมีค่าเวลาประวิงและชื่อใหม่ตามที่กำหนดให้

5.9.5 การรับข้อมูลของเกต

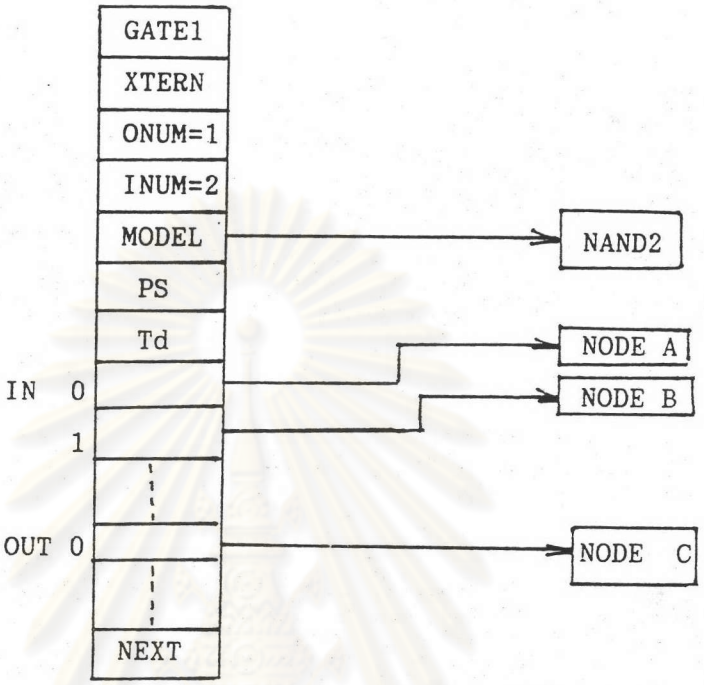
ในการรับข้อมูลของเกตนั้น โปรแกรมจะทำการตีความโดยแบ่งเกตเป็น 2 ชนิด คือ เกตแบบธรรมดากับแมคโคร กรณีที่เป็นเกตธรรมดาก็จะทำการเก็บไว้ใน

linked list แต่กรณีที่เป็นเกตแบบแมคโคร จะเรียกโปรแกรมย่อยมาทำการกระจายแมคโคร อีกทีหนึ่ง โครงสร้างข้อมูลของเกตตรรกะจะเป็นไปดังรูปที่ 5.16 ซึ่งอาจเขียนเป็นแผนภาพได้ดังรูป



รูปที่ 5.24 แผนภาพของการรับข้อมูลเกต

GATE1 NAND2 A B C



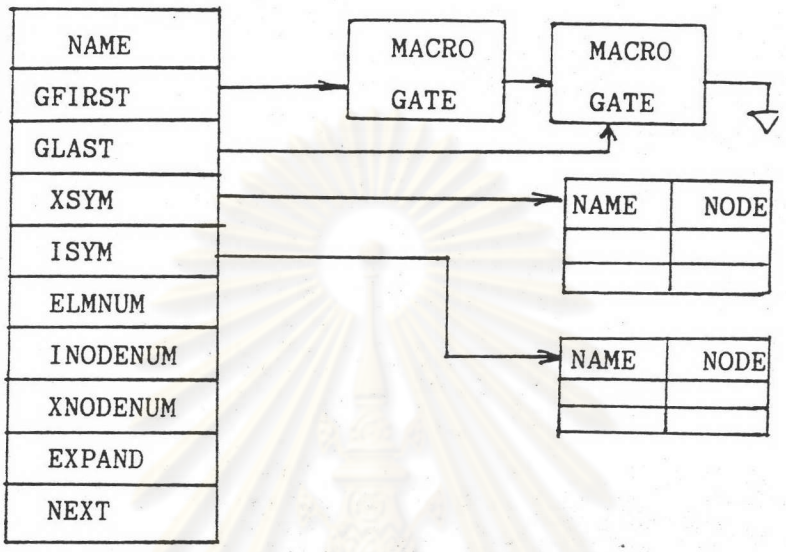
รูปที่ 5.25 การเก็บเกตในหน่วยความจำ

5.10 การทำงานของแมคโคร

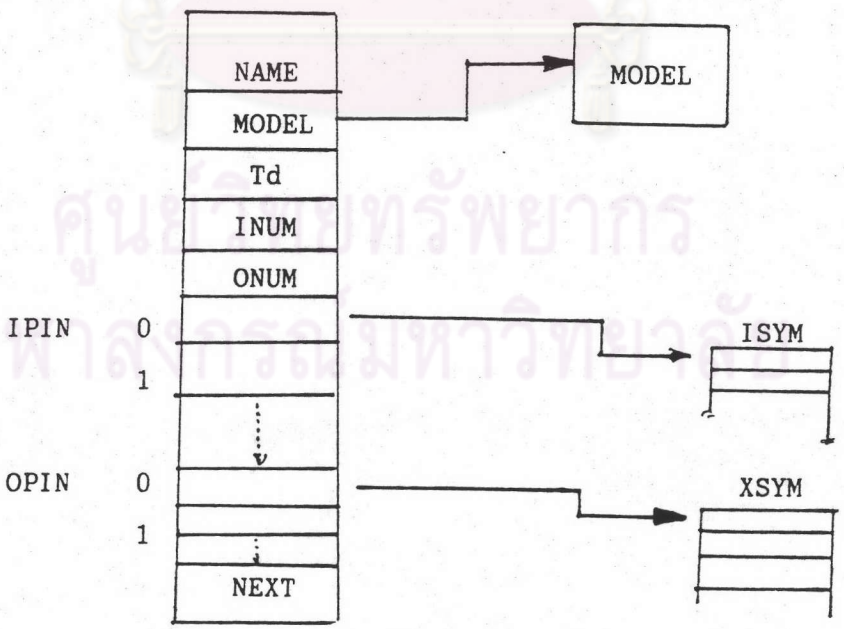
ในการทำงานของแมคโครนั้นจะแบ่งออกเป็น 2 ขั้นตอนด้วยกัน คือ ขั้นแรกเป็นช่วงที่โปรแกรมตรวจพบคำสั่งกำหนดแมคโคร โปรแกรมจะต้องอ่านแมคโครนั้นเข้ามาในหน่วยความจำ ขั้นตอนต่อมาเมื่อมีการเรียกใช้เกตนั้นโปรแกรมจะต้องทำการกระจายแมคโครนั้นออกและใส่ลงไปใน linked list ของเกตและโนดซึ่งจะแยกบรรยายเป็น 2 หัวข้อดังนี้

5.10.1 การรับข้อมูลของแมคโคร

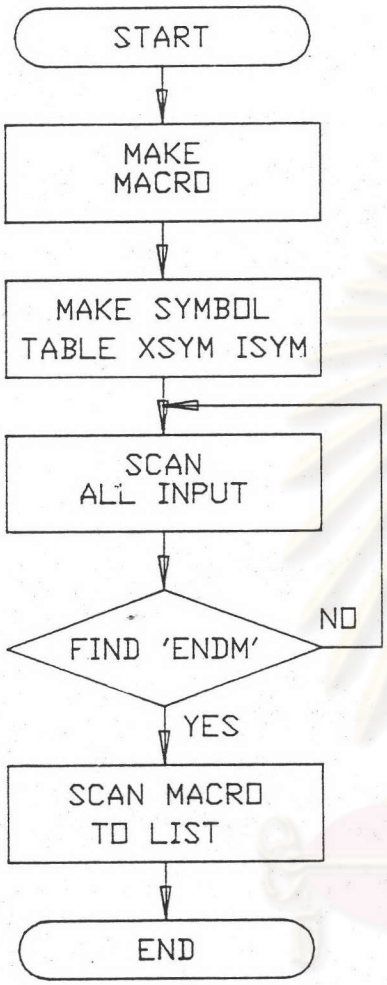
โครงสร้างข้อมูลทั้งหมดที่เกี่ยวข้องกับแมคโครจะเป็นดังรูปที่ 5.26 (ก) และ (ข) โดยที่ MACRO จะเป็นโครงสร้างข้อมูลหลักซึ่งจะเป็นตัวเก็บพารามิเตอร์ต่างๆที่ควบคุมการทำงานของแมคโคร MACROGATE เป็นโครงสร้างข้อมูลสำหรับเกตภายใน XSYM และ ISYM เป็นตารางเก็บชื่อโนดทั้งภายในและภายนอก EXPAND เป็นตัวนับจำนวนครั้งที่แมคโครนี้ถูกกระจายออกไป การทำงานจะเป็นไปดังรูปที่ 5.27

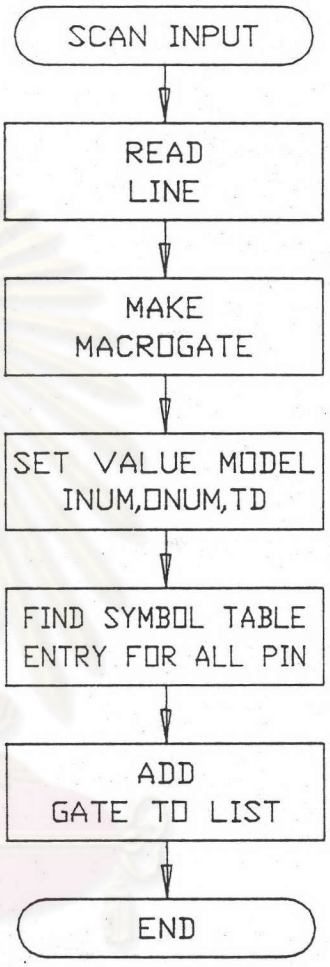


รูปที่ 5.26(ก) แมคโครและตารางสัญลักษณ์



รูปที่ 5.26 (ข) MACROGATE





(ก) การรับข้อมูลแมคโคร (ข) การ SCAN INPUT

รูปที่ 5.27 ขั้นตอนการรับข้อมูลจากแมคโคร

เมื่อโปรแกรมพบคำว่า MACRO จะเข้าสู่ส่วนที่เก็บค่าแมคโครในหน่วยความจำ การทำงานจะเป็นตามรูปที่ 5.27 (ก) การสร้างตารางความสัญลักษณ์จะทำโดยการหาขนาดตารางในช่วงบรรทัดที่บอกโนดภายนอกภายใน จากนั้นจะทำการลอกชื่อโนดนำมาใส่ในตารางโดยโนดภายนอกใส่ตาราง XSYM โนดภายในใส่ตาราง ISYM เพื่อใช้ในการกระจายต่อไป จากนั้นโปรแกรมจะทำการตรวจบรรทัดต่อมาที่จะบอกถึงการจัดวงจรของเกตภายในแมคโคร จนกว่าจะพบคำว่า "ENDM" จึงเป็นอันสิ้นการทำงาน

การตรวจการป้อนเกตตามรูปที่ 5.27 (ข) นั้น โปรแกรมจะอ่าน netlist ของโปรแกรมขึ้นมา 1 บรรทัด จากนั้นจะจองหน่วยความจำเพื่อเก็บค่าของเกตภายในแมคโคร (MACROGATE) หาตัวชี้ไปยังโมเดลของเกตนั้นมาใส่ในฟิลด์โมเดลและทำการตั้งค่าเวลาประวิง จำนวนขั้วเข้า (INUM) ขั้วออก(ONUM) ขั้นต่อมา คือ หาค่าตำแหน่งในตารางสัญลักษณ์ของโนดที่ต่อกับขั้วทั้งหมด โดยการเทียบชื่อโนดและนำตัวชี้ที่ได้มาเก็บไว้ยัง IPIN และ OPIN ซึ่งจะนำไปใช้งานต่อไปในขั้นของการกระจายแมคโคร

5.11 <u>การกระจายแมคโคร</u>

การกระจายแมคโครจะเกิดขึ้นเมื่อวงจรตรวจพบว่ามีการเรียกใช้งานแมคโครเกิดขึ้นในช่วงที่มีการรับข้อมูลซึ่งการกระจายแมคโครจะเกิดขึ้นตามแผนภาพดังรูปที่ 5.28

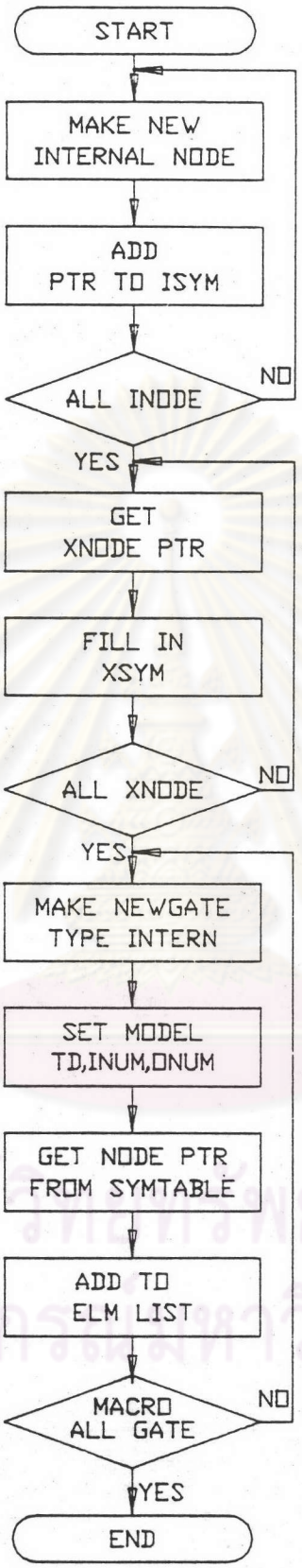
ในการกระจายแมคโครนั้น ขั้นแรกคือการตรวจดูตาราง ISYM จากนั้นสร้างโนดภายในทั้งหมดขึ้นมาใหม่ และนำค่าตัวชี้ที่ได้ไปใส่ไว้ในตารางสัญลักษณ์ ISYM ในฟิลด์โนด

ขั้นต่อมา ทำเช่นเดียวกันกับโนดภายนอก เพียงแต่โนดภายนอกนั้นต้องได้มาจากการตรวจสอบใน linked list ของโนดเพื่อที่จะทราบตำแหน่งของโนดในหน่วยความจำ ค่าตัวชี้ที่ได้นำมาใส่ในตาราง XSYM

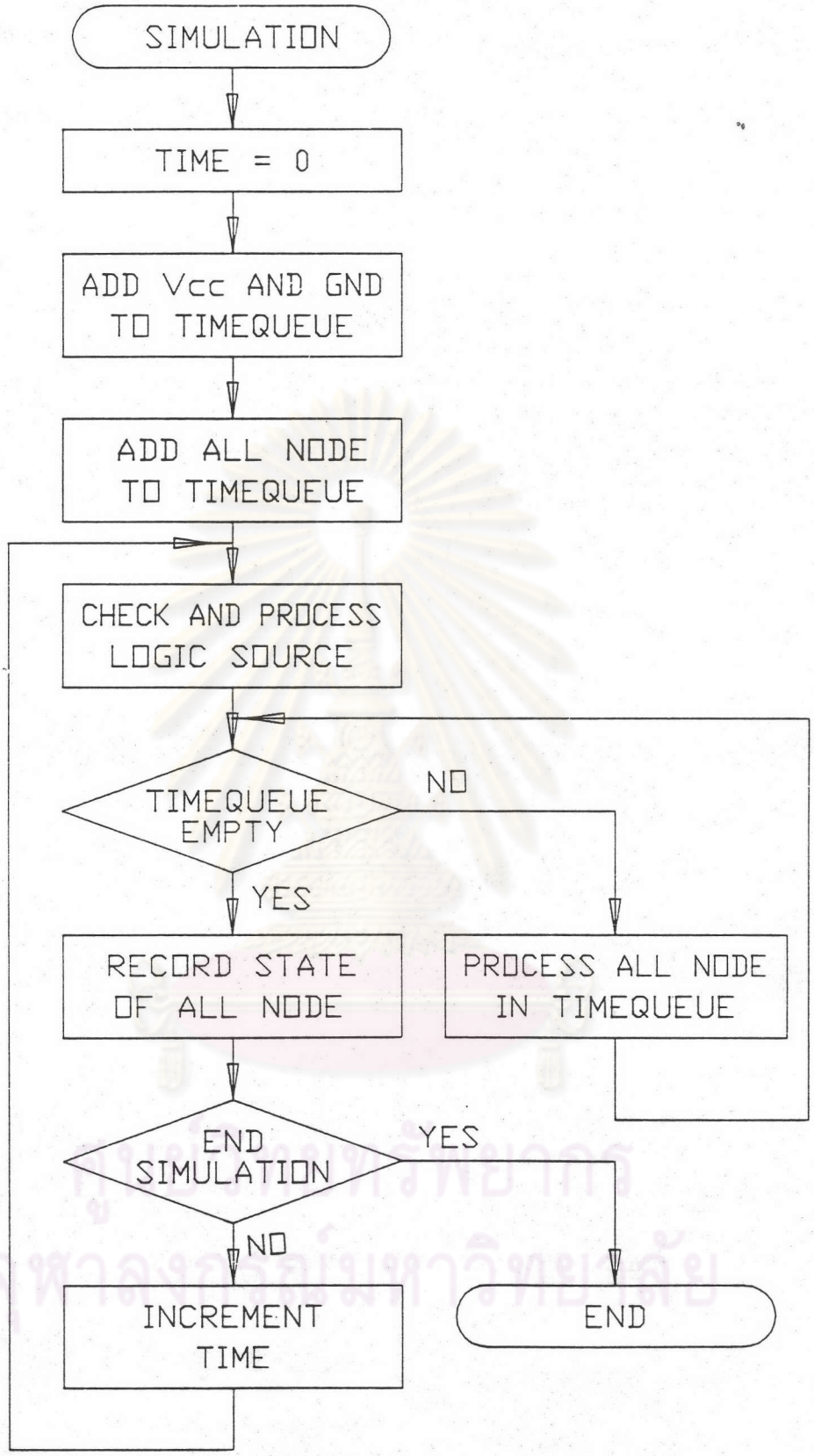
จากนั้นเป็นการสร้างเกตใหม่โดยมีชนิด INTERN และนำเข้าไปต่อยัง linked list ของเกต โดยที่ค่าพารามิเตอร์ต่างๆจะนำมาจากค่าที่เก็บในแมคโครเกต ค่าตัวชี้ไปยังโนดที่ต่ออยู่จะถูกนำมาใส่จากตารางสัญลักษณ์ เมื่อเสร็จสิ้นการทำงานในขั้นนี้กับเกตภายในแมคโครทุกตัว แมคโครก็จะถูกกระจายเข้าไปเป็นส่วนหนึ่งของวงจรในหน่วยความจำ

5.11 <u>การทำงานของส่วนวิเคราะห์การทำงาน</u>

เมื่อทำการอ่านข้อมูลทั้งหมดของวงจรเข้ามายังหน่วยความจำแล้ว โปรแกรมจะวิเคราะห์การทำงานของวงจรตรรกะเมื่อใช้คำสั่ง SIM การวิเคราะห์การทำงานของวงจรตรรกะ แสดงในแผนภาพในรูปที่ 5.29



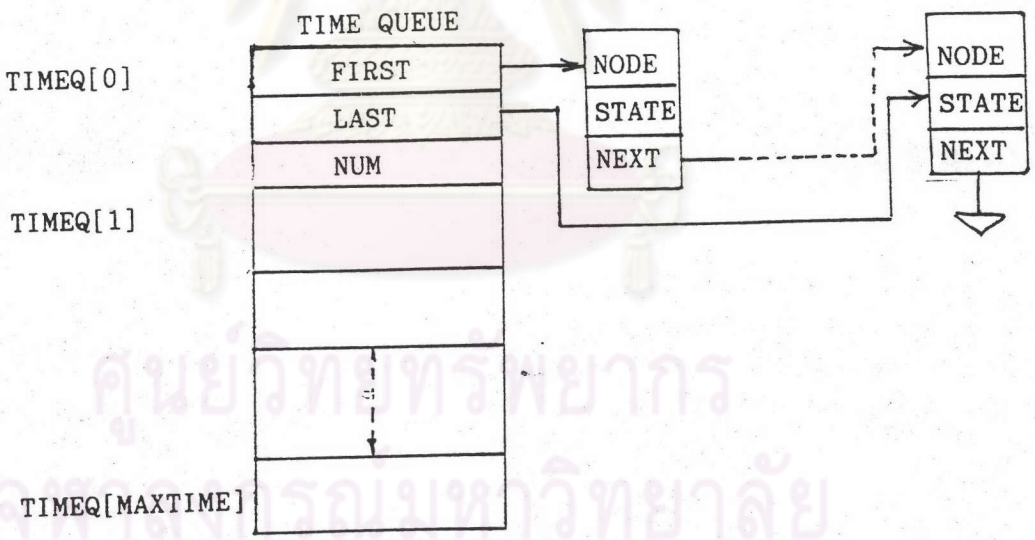
รูปที่ 5.28 การกระจายแมคโคร



รูป 5.29 การวิเคราะห์การทำงาน

ในการวิเคราะห์การทำงานจะเริ่มต้นด้วยการตั้งค่าเวลาให้เป็น 0 จากนั้นทำการใส่ค่า VCC และ GND ซึ่งเป็นโนดอ้างอิงใน time queue เพื่อวิเคราะห์สถานะที่มีการเปลี่ยนแปลงของโนดอ้างอิงเมื่อใส่พลังงานให้วงจร ขั้นต่อมาคือการใส่ค่าโนดทุกโนดลงใน Time queue ที่เวลา 0 เพื่อให้ค่าเริ่มต้นของวงจรถูกทำการวิเคราะห์ การทำงานจะเริ่มด้วยการตรวจสอบว่ามีการเปลี่ยนแปลงค่าที่แหล่งกำเนิดสัญญาณหรือไม่ ถ้ามีก็ใส่การเปลี่ยนแปลงของสัญญาณลงไปที่ time queue เวลานั้น จากนั้นทำการตรวจสอบและจัดการกับการ time queue ในช่วงเวลาดังกล่าว หลังขั้นตอนนี้จะบันทึกค่าไว้เพื่อแสดงผลต่อไป การทำงานจะเป็นดังนี้จนสิ้นสุดการวิเคราะห์การทำงาน คือ ที่เวลาเท่ากับ 1024 MRT

5.12 การจัดการ time queue ในโปรแกรม

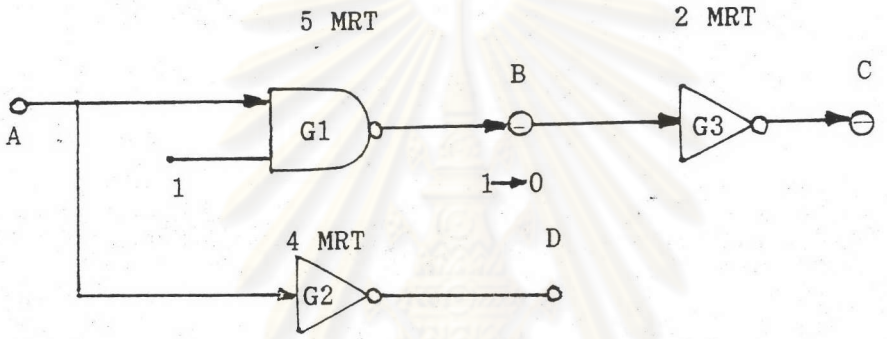


รูปที่ 5.30 โครงสร้างของ time queue

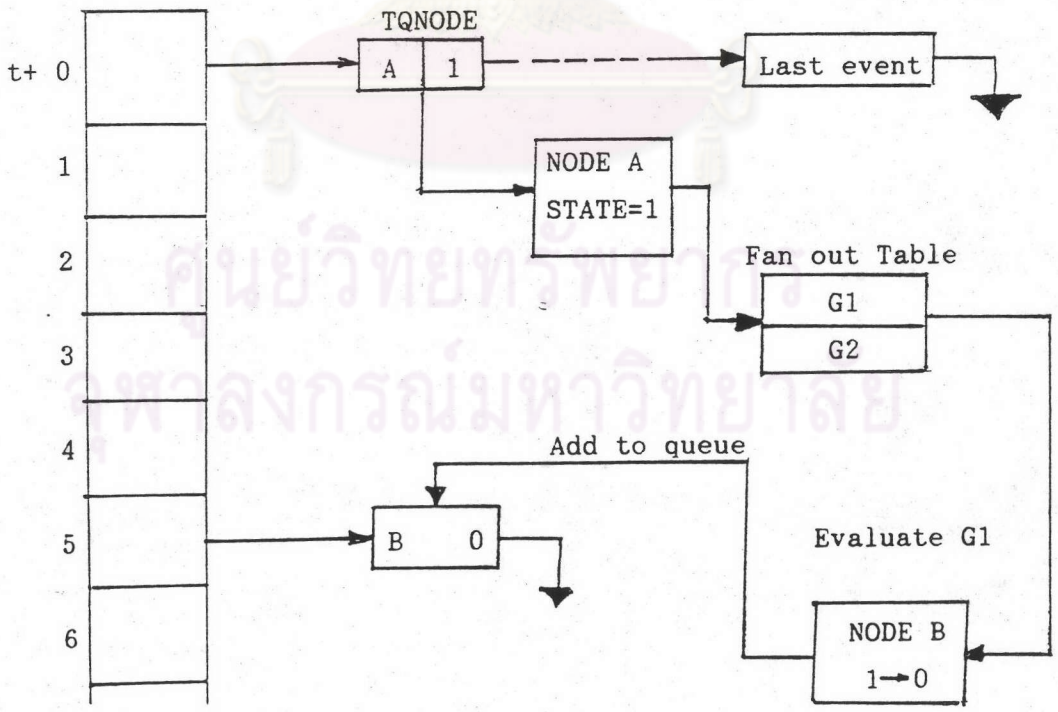
จากรูปที่ 5.30 time queue จะประกอบด้วยโครงสร้างข้อมูล 2 ชนิดด้วยกัน คือ ตัว time queue และ TQNODE ซึ่งใช้เก็บ event ใน time queue โครงสร้างข้อมูล

ใน TQNODE นั้น ฟิลด์ชื่อ node จะเป็นตัวเก็บชี้ไปยังโนดที่จะมีการเปลี่ยนแปลง และ STATE เป็นที่เก็บสถานะใหม่ของโนดนั้น TQNODE จะถูกเก็บในลักษณะ linked list ซึ่งควบคุมโดย time queue ส่วน time queue จะเป็นตัวแปรชนิดarray ชื่อ TIMEQ มีขนาด MAXTIME โดยที่ MAXTIME คือ เวลามากที่สุดที่จะทำการวิเคราะห์วงจรมีค่าเท่ากับ 1024 MRT โดยมี FIRST เป็นตัวชี้ไปยัง TQNODE ตัวแรก LAST ชี้ไปยังตัวสุดท้ายและNUM เป็นจำนวน TQNODE ใน linked list ซึ่งก็คือ จำนวน event ที่เกิดขึ้นในวงจรในขณะนั้นนั่นเอง

ในการอธิบายการทำงานของ time queue จะทำโดยอาศัยตัวอย่างในรูปที่ 5.31

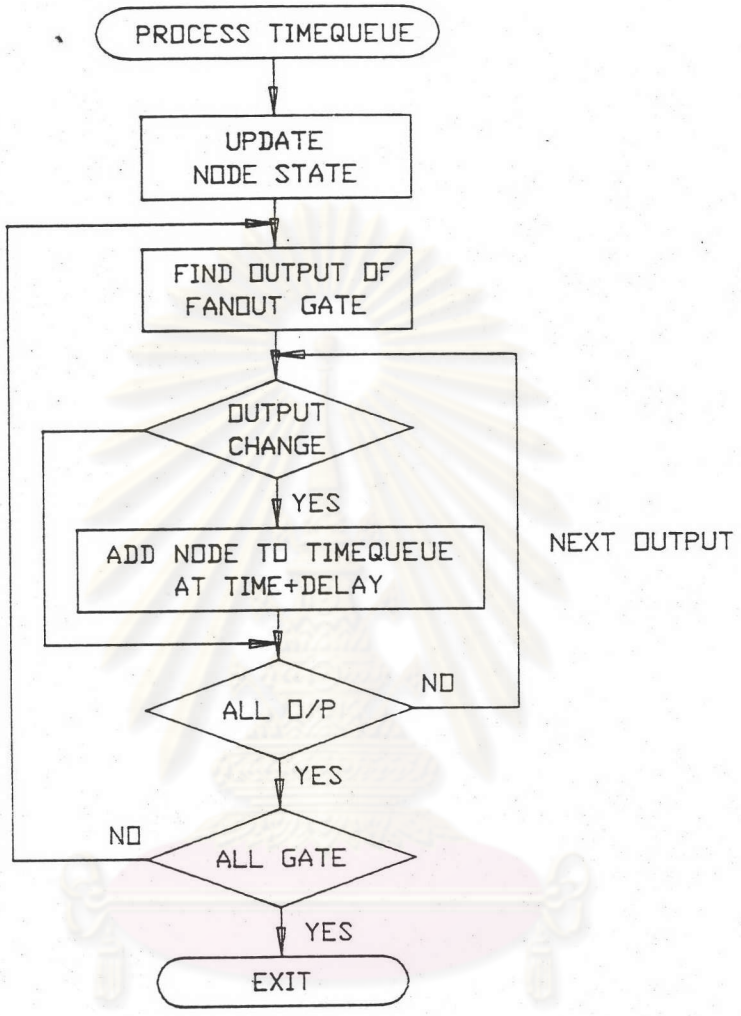


ก) วงจรแสดงการทำงานของ time queue



ข) ขั้นตอนการทำงานของ time queue

รูปที่ 5.31 แสดงการทำงานของ time queue



รูปที่ 5.32 แผนภาพแสดงการทำงานของ time queue

จากรูป 5.31 โปรแกรมทำงานมาจนถึงเวลา t พบว่าใน time queue มี event เกิดที่โนด A ซึ่งจะทำให้โนด A เปลี่ยนสถานะใหม่เป็น 1 โปรแกรมจะเปลี่ยนค่าสถานะที่เก็บไว้ที่โนด A ให้เป็น 1 จากนั้นจะทำการตรวจสอบ fan out table ของโนด A เพื่อหาเกตที่ถูกขับโดยโนด A พบว่าเป็นเกต G1 และ G2 โปรแกรมจะหาค่าตรรกะของเกต G1 จะเห็นว่าเกต G1 มีการเปลี่ยนสถานะด้านออกจาก 1 เป็น 0 ทำให้เกิด event ขึ้นที่โนด B โดยโนด B มีค่าตรรกะใหม่เป็น 0

ขั้นตอนต่อมาจะมีการเพิ่มการเปลี่ยนแปลงที่โนด B ลงใน time queue ที่เวลาเท่ากับ t + 5 เนื่องจากเวลาประวิงของเกต G1 มีค่า 5 MRT การเพิ่ม event ใน

time queue ทำโดยการสร้าง TQNODE ซึ่งเก็บตัวชี้ไปยังโนด B และสถานะในอนาคตของ B ซึ่งเป็นค่าตรรกะ 0 ไว้และเชื่อมต่อกับ linked list ที่เวลาเท่ากับ t + 5

โปรแกรมจะทำงานเช่นเดียวกันกับ G2 และจะเปลี่ยนไปทำงานกับ TQNODE ตัวต่อไปจนกว่าจะหมดค่าใน linked list ที่เวลานั้น ซึ่งการจัดการกับ time queue ก็จะเลื่อนไปยังเวลาต่อไป

การทำงานทั้งหมดที่กล่าวมาจะสรุปได้ดังแผนภาพในรูปที่ 5.32

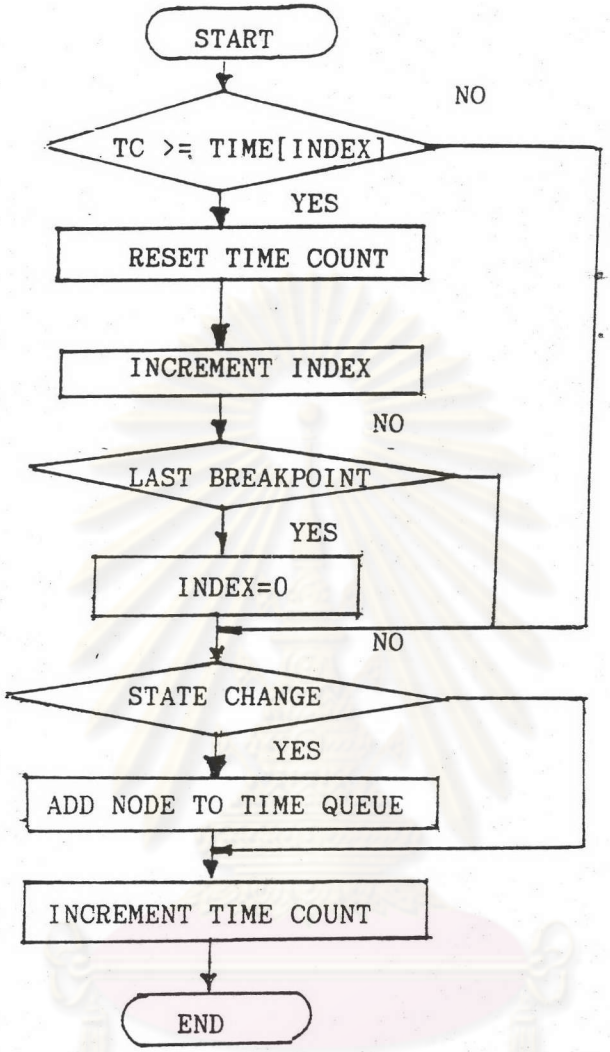
5.13 การทำงานของแหล่งกำเนิดสัญญาณ

ในการวิเคราะห์การทำงานในทุก MRT นั้นจะต้องมีการตรวจสอบการเปลี่ยนแปลงของแหล่งกำเนิดสัญญาณทุกตัวในวงจรก่อน และทำการเพิ่มโนดที่การเปลี่ยนแปลงลงไปใน time queue

รูปที่ 5.17 ซึ่งแสดงโครงสร้างข้อมูลของแหล่งกำเนิดสัญญาณ จะมีตัวแปร 3 ตัว คือ pnum ซึ่งเป็นจำนวน breakpoint ของแหล่งกำเนิดสัญญาณนั้น tc เป็นตัวนับจำนวนคาบเวลาภายใน 1 breakpoint และ index เป็นตัวชี้ในตารางว่าปัจจุบันอยู่ที่ break point จุดใด

ในการทำงานนั้นทุกค่าเวลาที่ทำการวิเคราะห์ จะมีการเพิ่มค่าตัวแปร tc หาก tc มากกว่าคาบเวลาใน break point นั้น ก็จะทำการเพิ่มค่า index ขึ้น 1 และตั้งค่า tc กลับเป็น 0 ใหม่ ซึ่งเป็นการเลื่อนไปยัง Breakpoint ต่อไป กรณีที่ index มีค่ามากกว่าจำนวน break point ใน pnum แล้ว index จะถูกตั้งให้เป็น 0 ซึ่งเป็นการทวนซ้ำของแหล่งกำเนิดสัญญาณ

ทุกช่วงเวลานี้จะมีการอ่านค่าสถานะออกมาเปรียบเทียบกับโนดที่ถูกขับด้วยแหล่งกำเนิดสัญญาณนั้น หากมีการเปลี่ยนค่าตรรกะแล้วโนดนั้นจะถูกเพิ่มลงไปใน time queue การทำงานทั้งหมดสามารถสรุปได้ดังแผนภาพในรูปที่ 5.33



รูปที่ 5.33 แสดงการทำงานของแหล่งกำเนิดสัญญาณ