บรรณานุกรม

National Semiconductor, TTL DATA BOOK, Santo Clora, U.S.A., 1976

____. CMOS Integraled Circuits, Santo Clora, U.S.A., 1975

Fairchild, TTL DATA BOOK, California, U.S.A., 1978

____. CMOS DATA BOOK, California, U.S.A., 1977

____. FULL LINE CONDENSED CATALOG, California, U.S.A., 1978

Signatics, DIGITAL, LINEAR, CMOS DATA BOOK, California, U.S.A., 1974

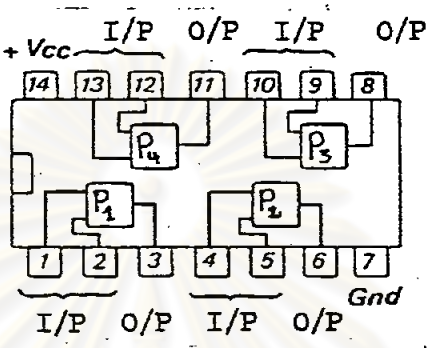
Peter R.Rony, David G. Larsen, Jonothon A. Titus, THE 8080A BUGBOOK,
Howord W. Soms & Co., Ins, Indianopolis, Indiana, U.S.A.,1977

Intel, MCS - 80 USER'S MANUAL, Santa Clara, California, U.S.A., 1977

ภาคผนวก ก

แสดงรายละเอียดและ FLOW CHART ในการทดสอบ IC ประเภทต่าง ๆ ซึ่งแบ่งเป็นประเภท ๆ ดังนี้

ก. IC ประเภทที่ ๑ ใช้ SOCKET #1 (มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๑



รูปที่ ก.๑ IC ประเภทที่ ๑

จากรูปที่ ก.๑ มีทั้งหมด 4 GATE ในที่นี้แบ่งเป็น ๔ ส่วน (PART) ได้แก่ P_1, P_2, P_3 และ P_4

- P_1 มีขา ๑ และ ๒ เป็น INPUT/ ขา๓ เป็น OUTPUT
- P_2 มีขา ๔ และ ๕ เป็น INPUT/ ขา ๖ เป็นOUTPUT
- P_3 มีขา ๙ และ ๑๐ เป็นINPUT/ ขา ๘ เป็น OUTPUT
- P_4 มีขา ๑๒ และ ๑๓ เป็น INPUT/ขา ๑๑ เป็น OUTPUT

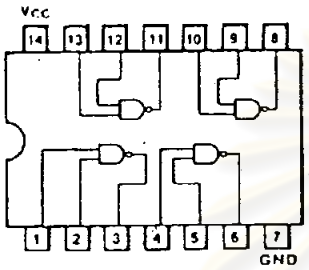
โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อยที่ สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่มีลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปแทน ก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภทที่ ๑ นี้ได้แก่

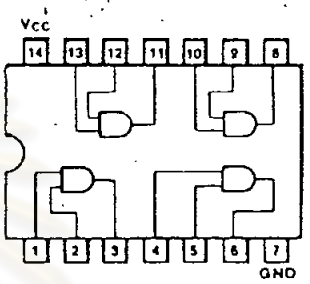
- 2 INPUT NAND GATE เช่น7400,7403, 7426, 7437, 7438, 74132, และ 74C00 เป็นต้น
- 2 INPUT AND GATE เช่น 7408, 74C09 และ 74C08
- 2 INPUT OR GATE เช่น 7432 และ74C32
- 2 INPUT EXCLUSIVE - OR GATE เช่น7486 และ74C86
- TRI STATE เช่น 74125, 74126

ตัวอย่าง IC LOGIC DIAGRAM

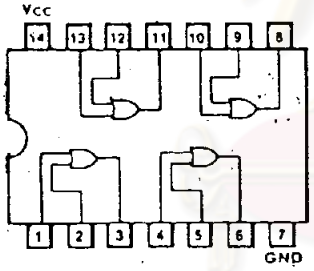
9002, 54/7400,
54H/74H00, 54S/74S00,
54LS/74LS00, 9012,
54H/74H01, 54/7403,
54S/74S03, 54LS/74LS03,
7426, 54LS/74LS26
54/7437, 54LS/74LS37,
54/7438, 74LS38, 54/74132,
54S/74S132, 74LS132



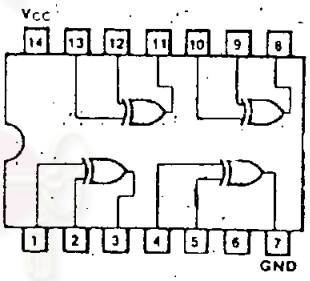
547408, 54H/74H08,
54S/74S08, 54LS/74LS08
54/7409, 54S/74S09,
54LS/74LS09



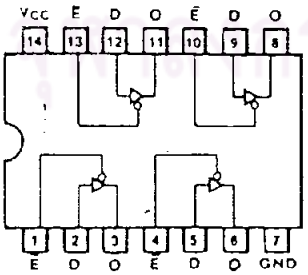
54/7432, 54S/74S32
54LS/74LS32



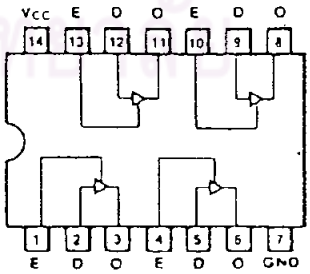
54/7486, 54S/74S86,
54LS/74LS86, 54LS/74LS136



54/74125, 54LS/74LS125



54/74126, 54LS/74LS126



แนวความคิดในการเขียนโปรแกรมเพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๑

PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๑ ข้อมูลที่ SET ดังนี้ (จะต้อง SET ก่อนจะทดสอบ แล้วบอกผู้ใช้ว่า READY)

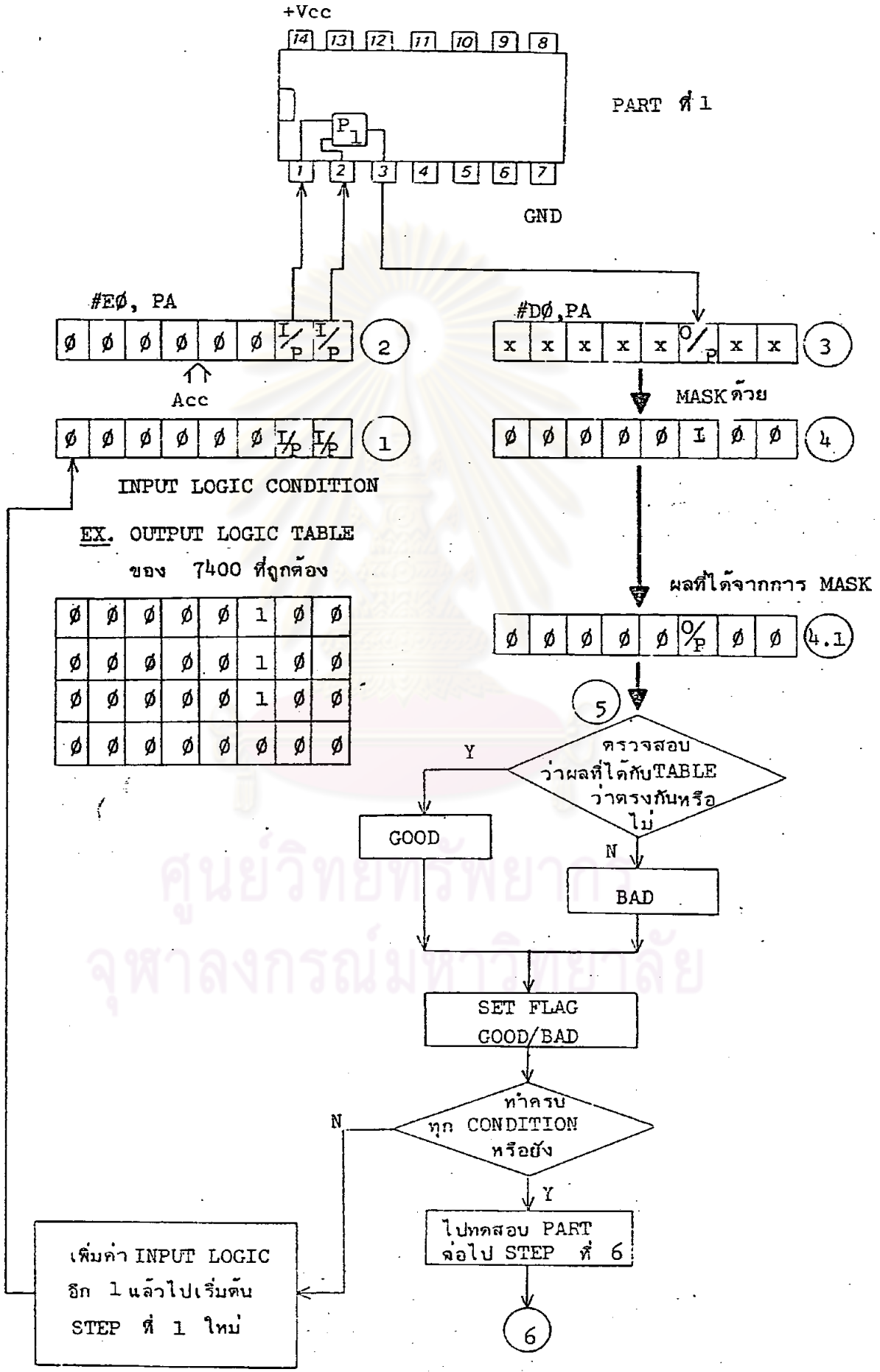
F1,PB

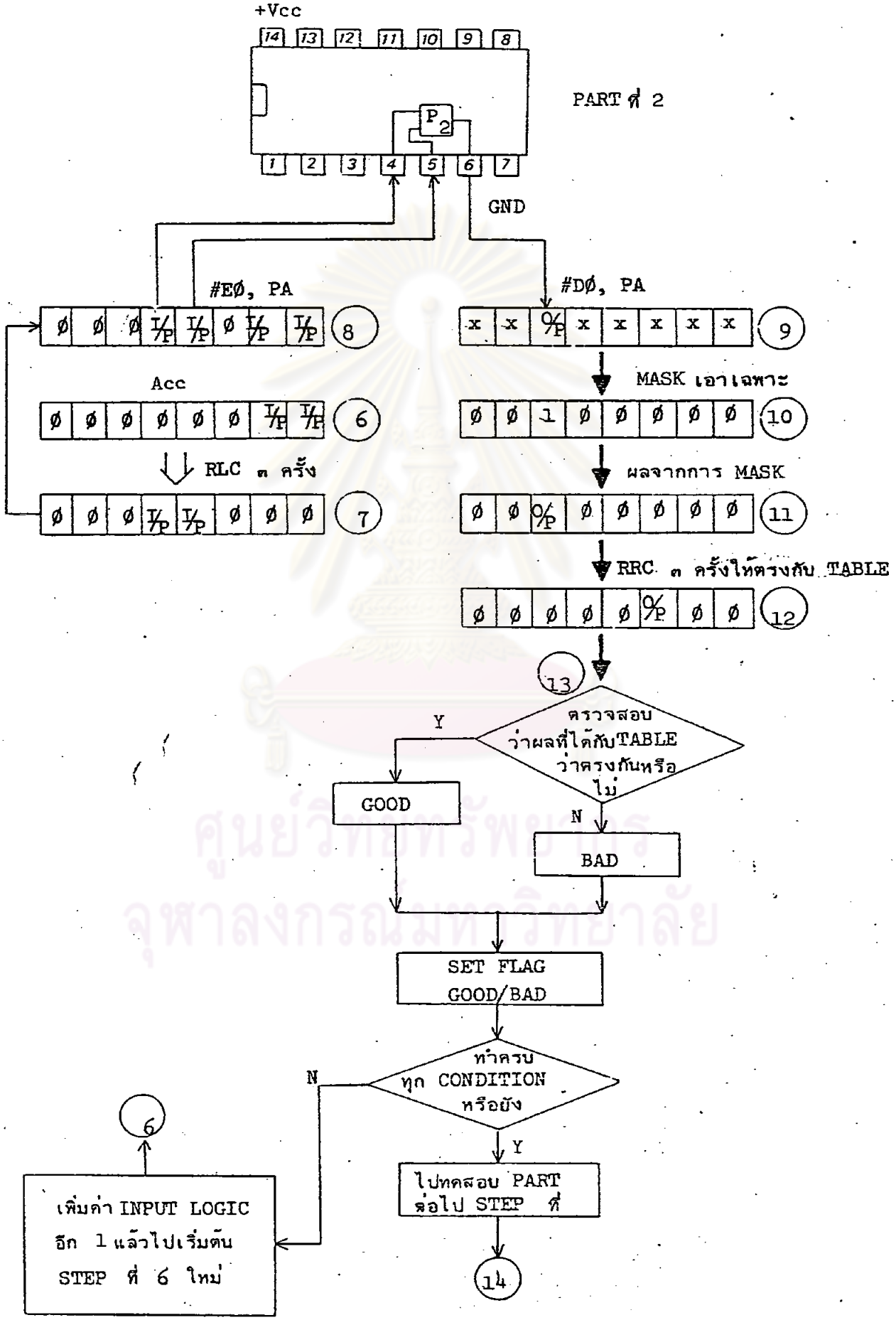
X'Ø2' | Ø | Ø | Ø | Ø | Ø | Ø | 1 | Ø |

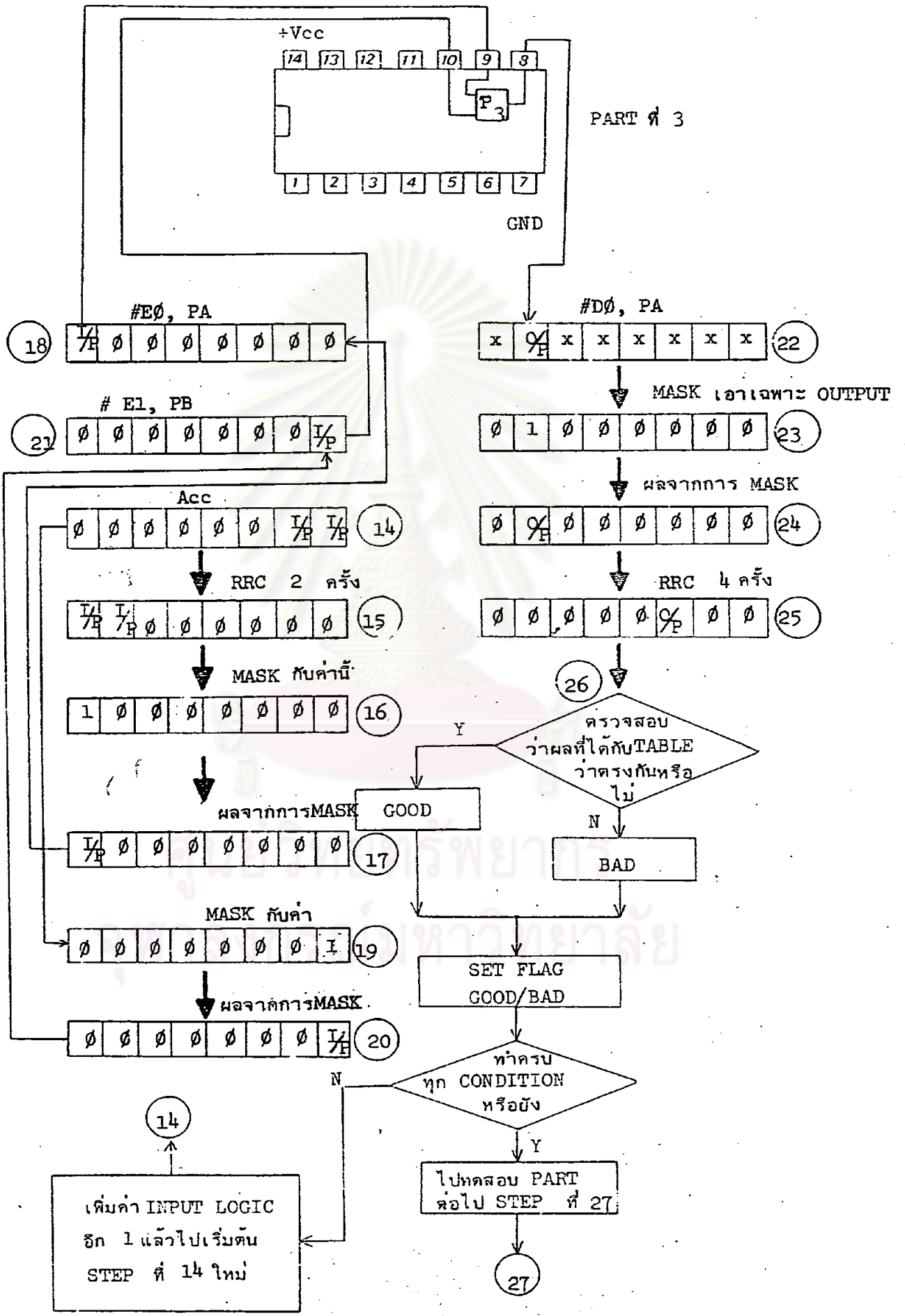
#FØ, PA

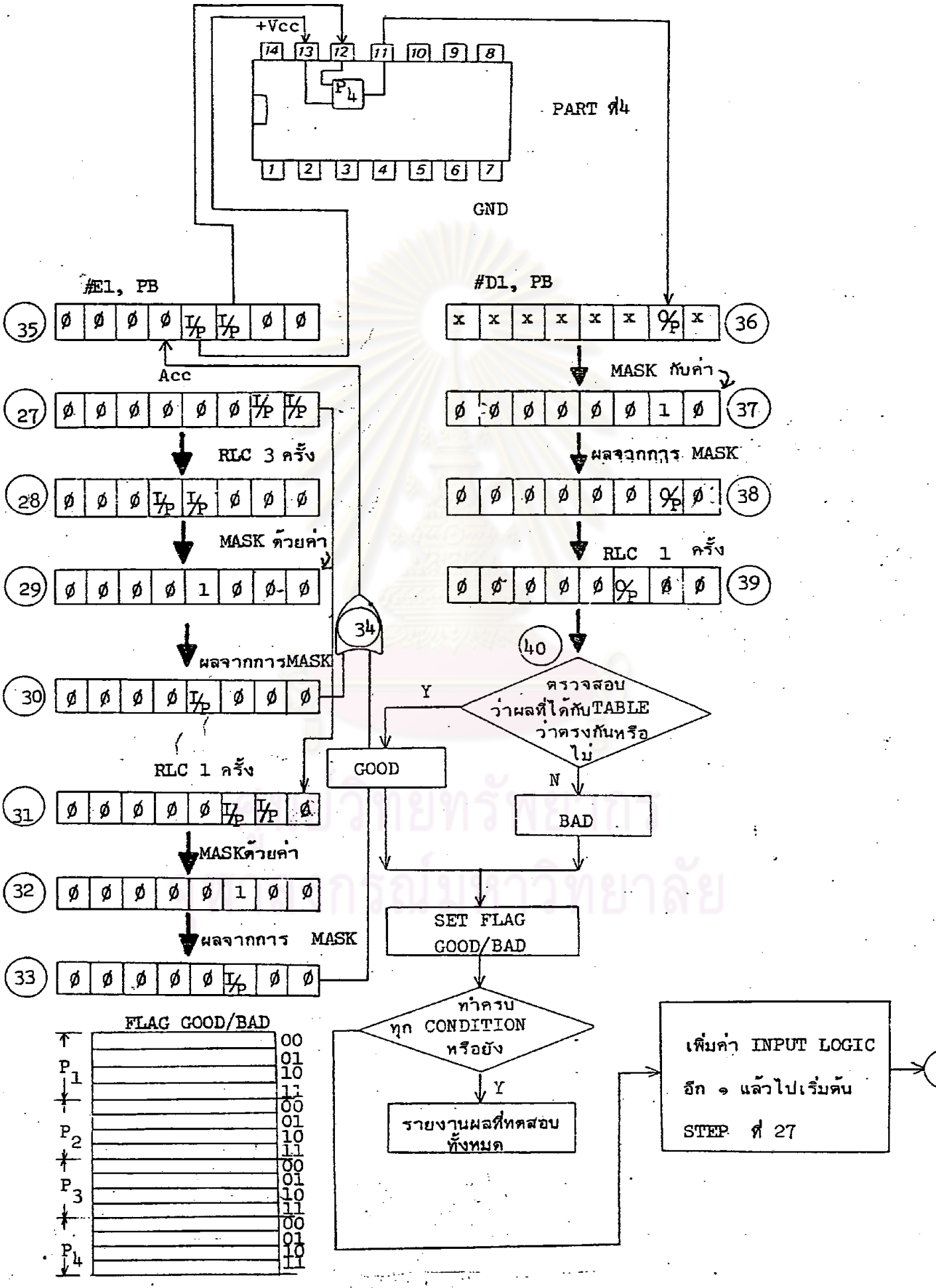
| Ø | 1 | 1 | Ø | Ø | 1 | Ø | Ø | X '54'

FLOW CHART ที่ ก.๑ เป็น FLOW CHART ของโปรแกรมทดสอบ IC ประเภทที่ ๑

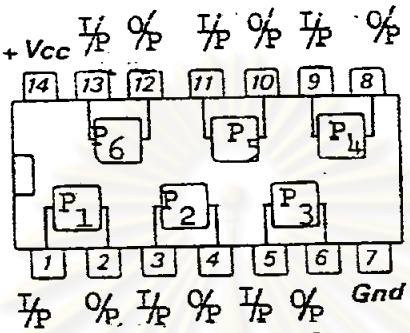



+Vcc
14 13 12 11 10 9 8
P2
1 2 3 4 5 6 7
PART ที่ 2
GND
#EØ, PA
#DØ, PA
8
9
Acc
6
MASK เอาเฉพาะ
10
RLC ๓ ครั้ง
7
ผลจากการ MASK
11
RRC ๓ ครั้งให้ตรงกับ TABLE
12
13
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
N
Y
6
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 6 ใหม่
ไปทดสอบ PART ต่อไป STEP ที่
14


+Vcc
14 13 12 11 10 9 8
P3
1 2 3 4 5 6 7
GND
PART ที่ 3
#EØ, PA
18
I/P Ø Ø Ø Ø Ø Ø Ø
E1, PB
21
Ø Ø Ø Ø Ø Ø Ø I/P
Acc
Ø Ø Ø Ø Ø Ø I/P I/P
14
RRC 2 ครั้ง
I/P I/P Ø Ø Ø Ø Ø Ø
15
MASK กับค่านี้
1 Ø Ø Ø Ø Ø Ø Ø
16
ผลจากการMASK
I/P Ø Ø Ø Ø Ø Ø Ø
17
MASK กับค่า
Ø Ø Ø Ø Ø Ø Ø I
19
ผลจากการMASK
Ø Ø Ø Ø Ø Ø Ø I/P
20
#DØ, PA
x O/P x x x x x x
22
MASK เอาเฉพาะ OUTPUT
Ø 1 Ø Ø Ø Ø Ø Ø
23
ผลจากการ MASK
Ø O/P Ø Ø Ø Ø Ø Ø
24
RRC 4 ครั้ง
Ø Ø Ø Ø Ø O/P Ø Ø
25
26
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
N
Y
14
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 14 ใหม่
ไปทดสอบ PART ต่อไป STEP ที่ 27
27


+Vcc
14 13 12 11 10 9 8
P4
PART ที่4
1 2 3 4 5 6 7
GND
#E1, PB
#D1, PB
35
36
Acc
MASK กับค่า
27
37
RLC 3 ครั้ง
ผลจากการ MASK
28
38
MASK ด้วยค่า
RLC 1 ครั้ง
29
39
34
40
ผลจากการMASK
ตรวจสอบ
ว่าผลที่ได้กับTABLE
ว่าตรงกันหรือ
ไม่
Y
30
GOOD
N
BAD
RLC 1 ครั้ง
31
MASKด้วยค่า
32
ผลจากการ MASK
SET FLAG
GOOD/BAD
33
ทำครบ
ทุก CONDITION
หรือยัง
Y
รายงานผลที่ทดสอบ
ทั้งหมด
เพิ่มค่า INPUT LOGIC
อีก ๑ แล้วไปเริ่มต้น
STEP ที่ 27
FLAG GOOD/BAD
P1
P2
P3
P4
00
01
10
11

ข. IC ประเภทที่ ๒ ใช้ SOCKET #1(มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๒



รูปที่ ก.๒ IC ประเภทที่ ๒

จากรูปที่ ก.๒ มีทั้งหมดมี 6 GATEในที่นี้แบ่งเป็น ๖ ส่วน (PART) ได้แก่ P_1, P_2, P_3, P_4, P_5 และ P_6

- P_1 มีขา ๑ เป็น INPUT/ ขา ๒ เป็น OUTPUT
- P_2 มีขา ๓ เป็น INPUT/ ขา ๔ เป็น OUTPUT
- P_3 มีขา ๕ เป็น INPUT/ ขา ๖ เป็น OUTPUT
- P_4 มีขา ๘ เป็น INPUT/ ขา ๘ เป็น OUTPUT
- P_5 มีขา ๑๑เป็น INPUT/ ขา ๑๐ เป็นOUTPUT
- P_6 มีขา ๑๓เป็น INPUT/ ขา ๑๒ เป็น OUTPUT

โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อยที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่มีลักษณะของขา เหมือนกัน เพียงแต่ เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปเท่านั้น ก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภทที่ ๒ นี้ ได้แก่

- BUFFER เช่น 7407, 7417
- INVERTE เช่น 7404, 7414, 7416,74C04, 74C14,4069 และ4004

แนวความคิดในการเขียนโปรแกรม เพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ในรูปที่ ก.๒

PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๒ ข้อมูลที่ SET ดังนี้ (จะต้อง SET ก่อนจะทดสอบแล้วบอกผู้ใช้ว่า READY)

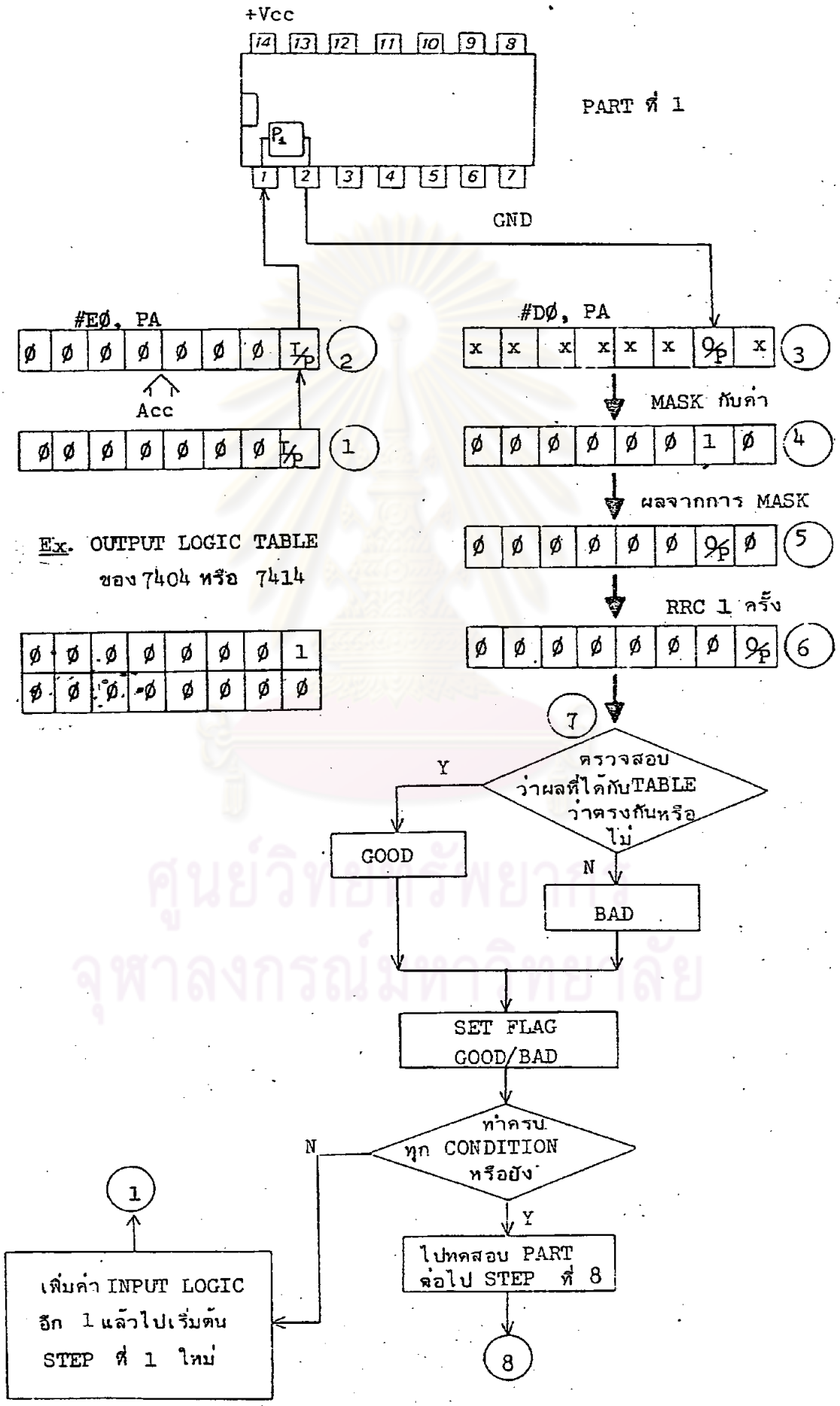
x'Ø5'

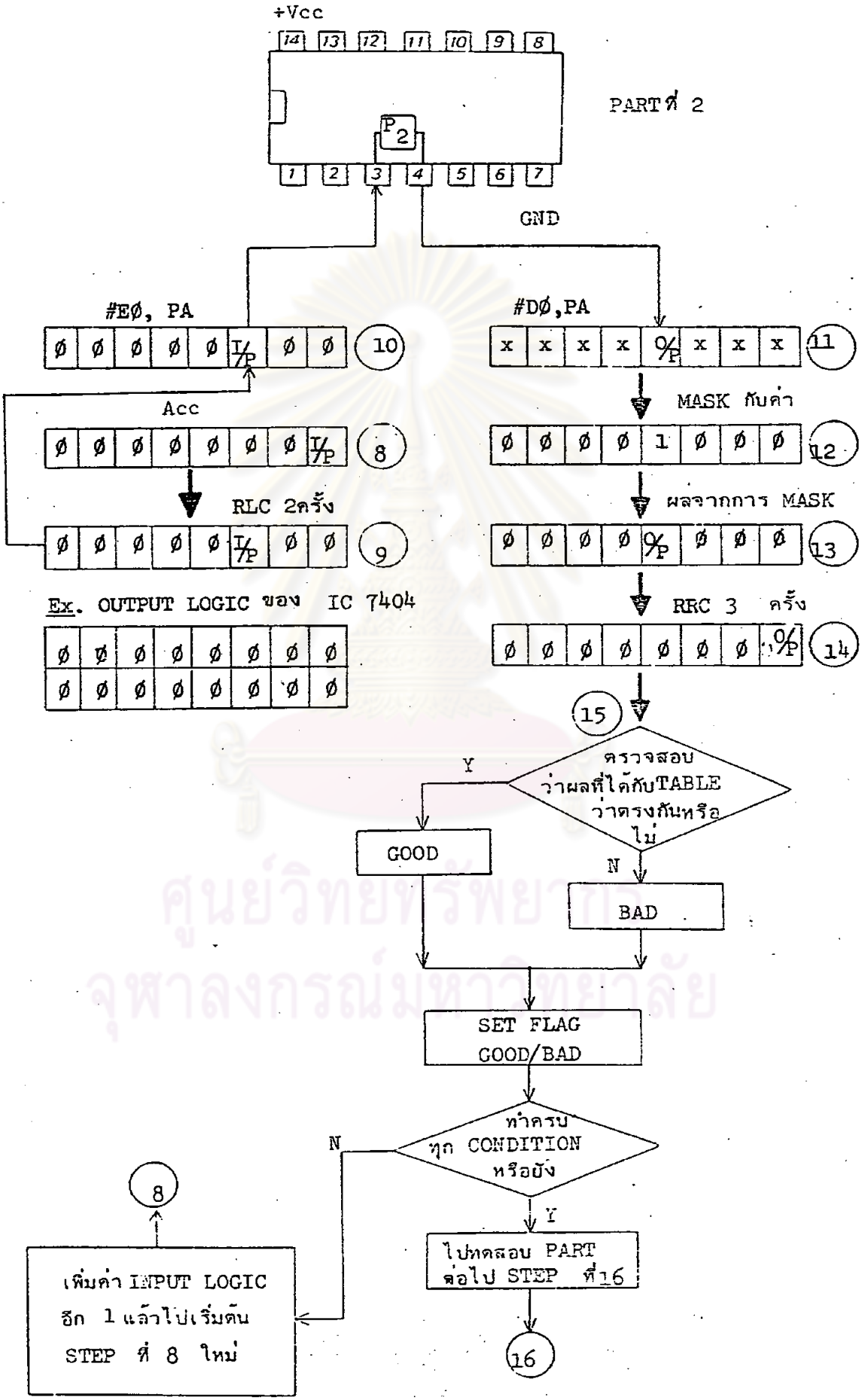
Ø	Ø	Ø	Ø	Ø	1	Ø	1

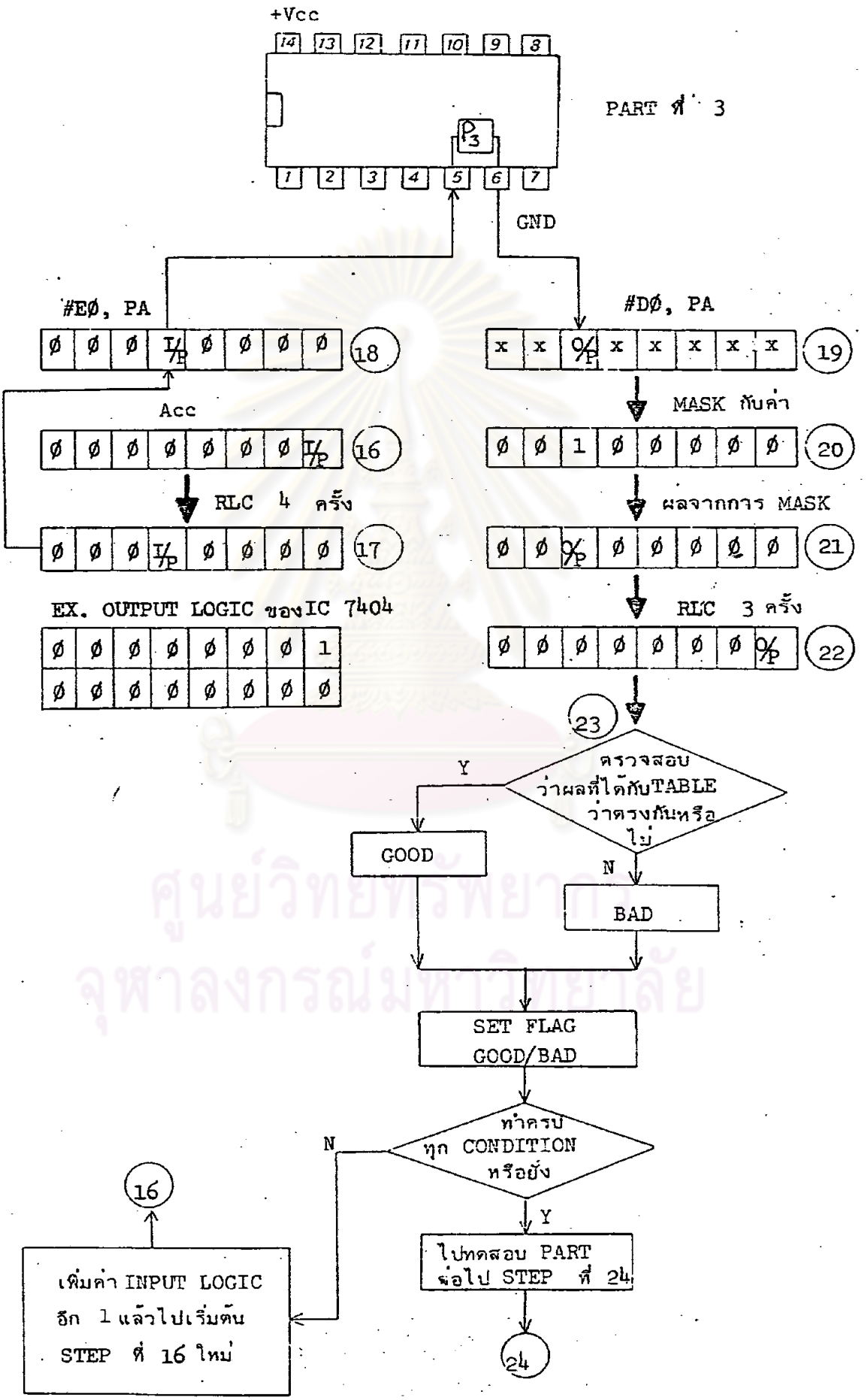
Ø	1	1	Ø	1	Ø	1	Ø

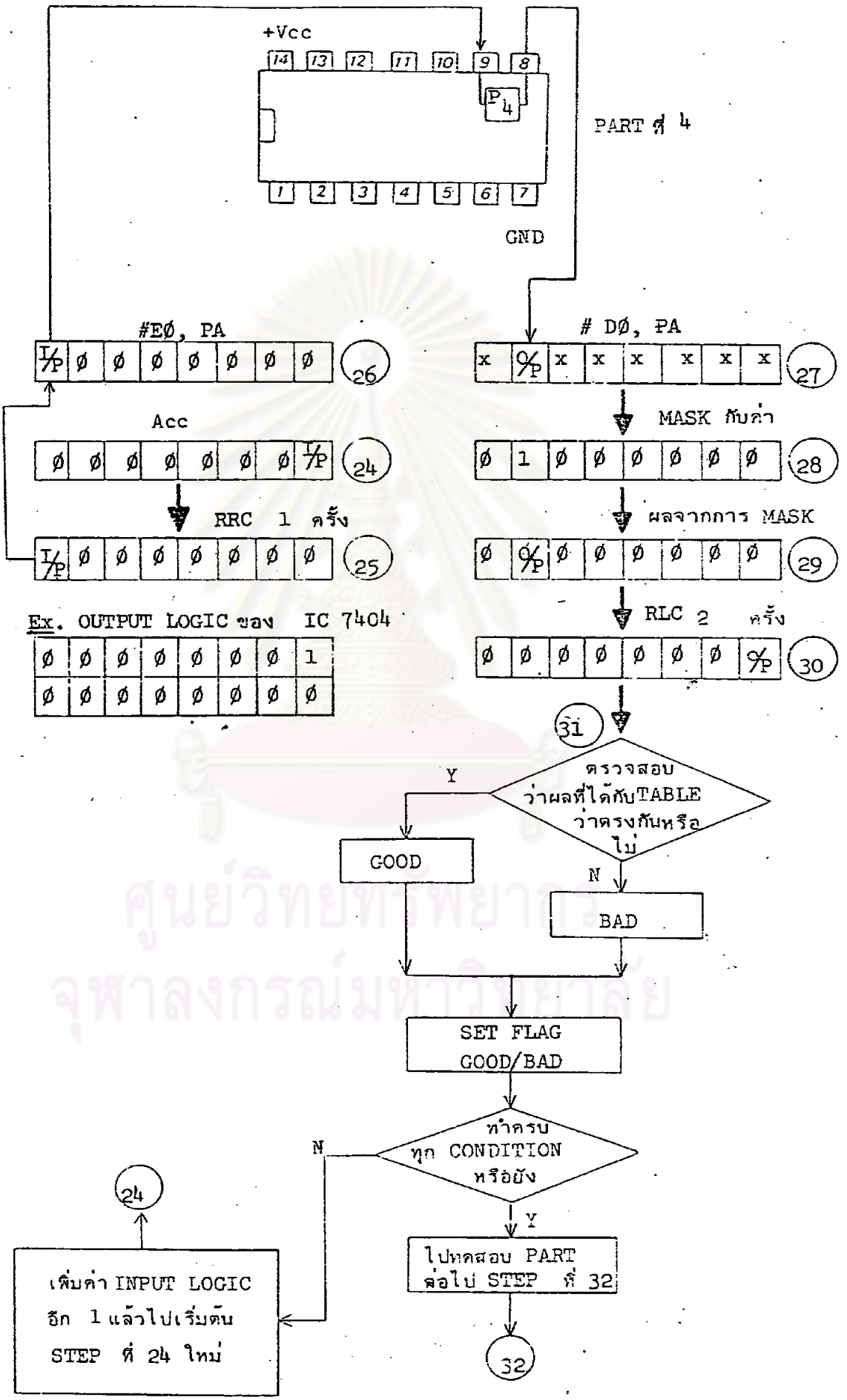
x'6A'

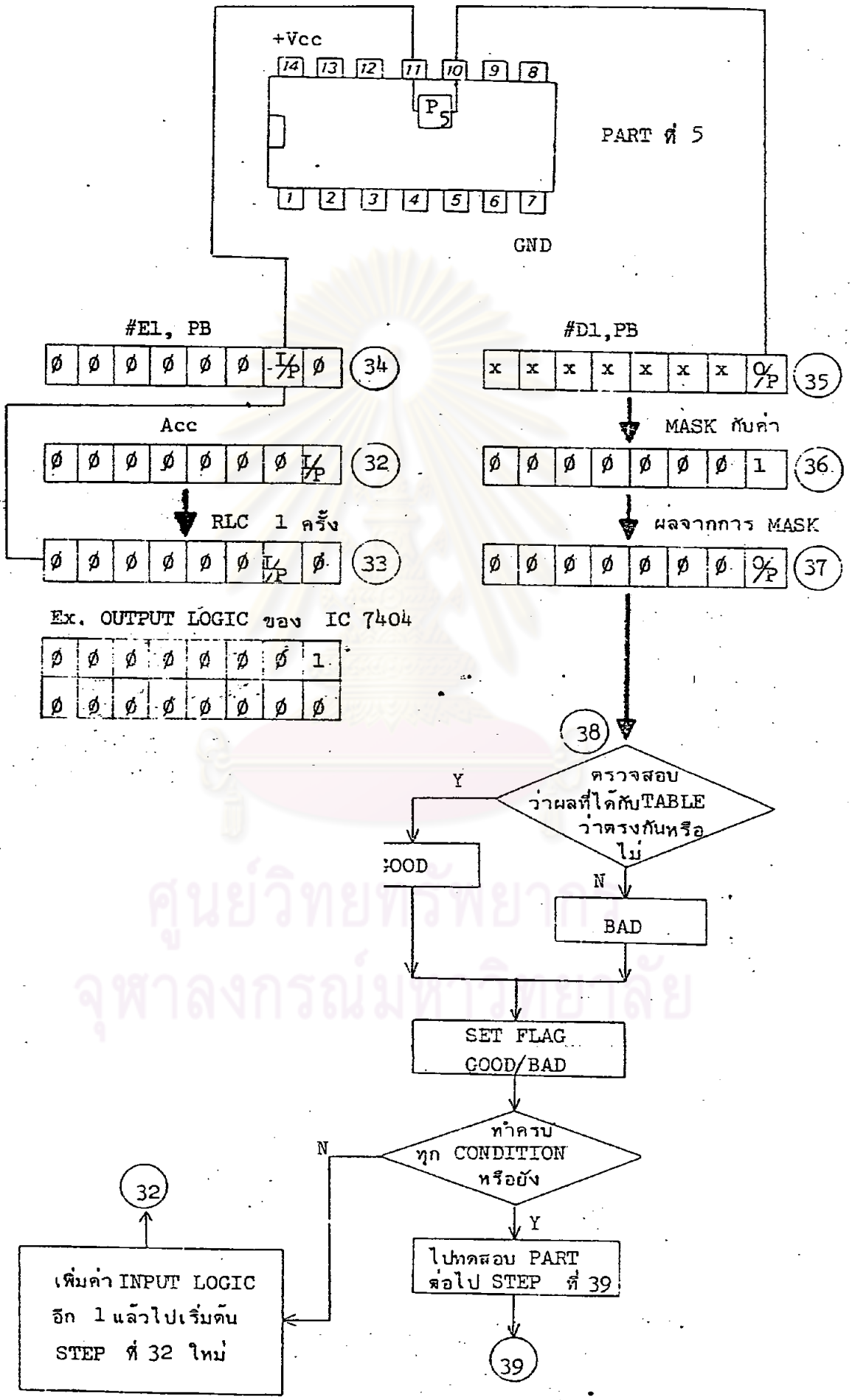
FLOW CHART ที่ ก.๒ FLOW CHART ของโปรแกรมทดสอบ IC ประเภท ที่ ๒

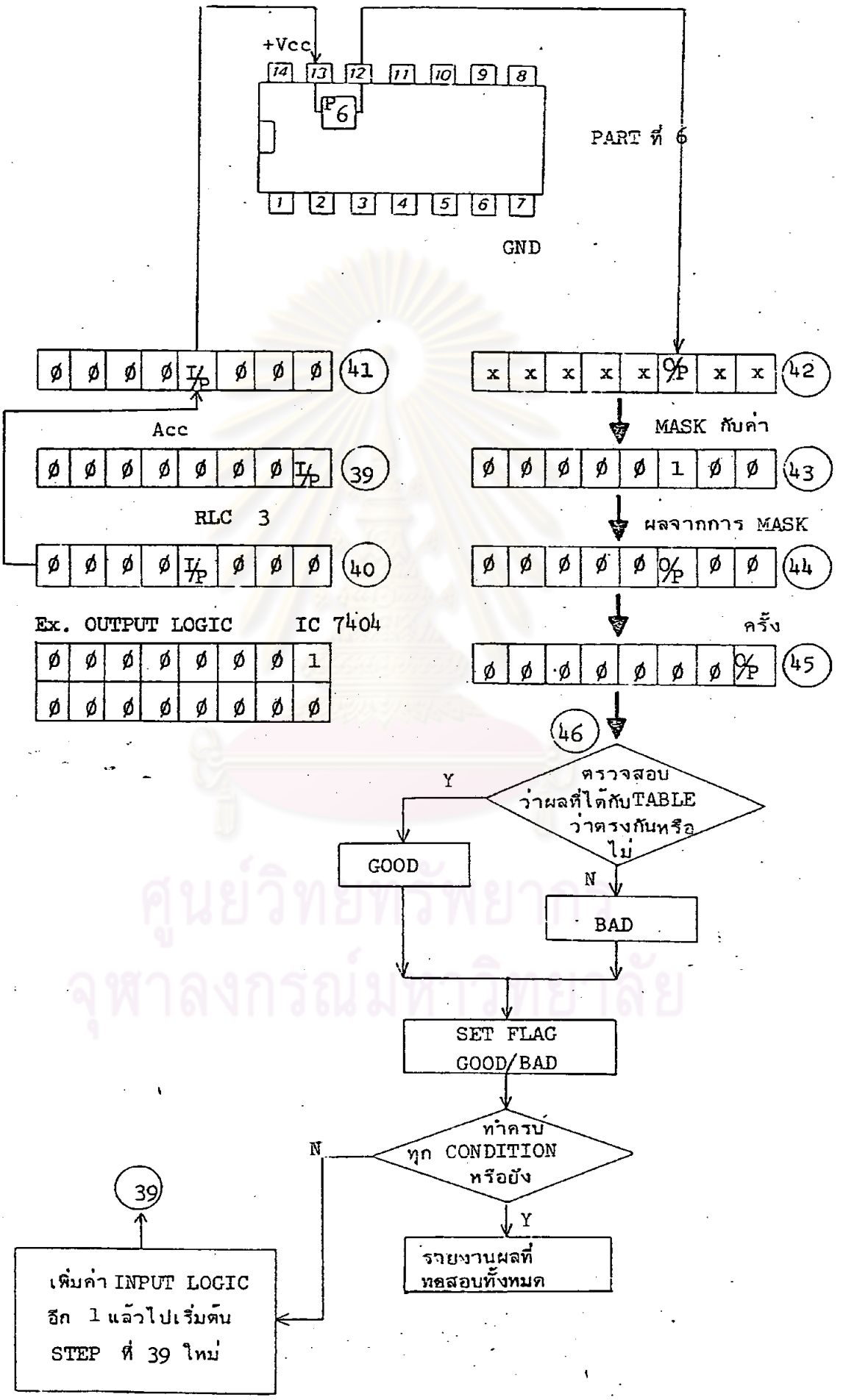




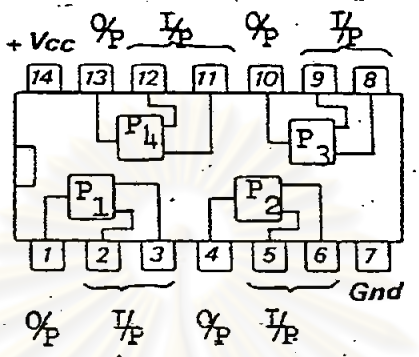



+Vcc
14 13 12 11 10 9 8
P4
1 2 3 4 5 6 7
GND
PART ที่ 4
#EØ, PA
I/P Ø Ø Ø Ø Ø Ø Ø
26
DØ, PA
x O/P x x x x x x
27
Acc
Ø Ø Ø Ø Ø Ø Ø I/P
24
MASK กับค่า
Ø 1 Ø Ø Ø Ø Ø Ø
28
RRC 1 ครั้ง
I/P Ø Ø Ø Ø Ø Ø Ø
25
ผลจากการ MASK
Ø O/P Ø Ø Ø Ø Ø Ø
29
RLC 2 ครั้ง
Ø Ø Ø Ø Ø Ø Ø O/P
30
Ex. OUTPUT LOGIC ของ IC 7404
Ø Ø Ø Ø Ø Ø Ø 1
Ø Ø Ø Ø Ø Ø Ø Ø
31
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
N
Y
24
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 24 ใหม่
ไปทดสอบ PART ต่อไป STEP ที่ 32
32




+Vcc
14 13 12 11 10 9 8
P 6
1 2 3 4 5 6 7
GND
PART ที่ 6
Ø Ø Ø Ø I/P Ø Ø Ø 41
x x x x x O/P x x 42
Acc
MASK กับค่า
Ø Ø Ø Ø Ø Ø Ø I/P 39
Ø Ø Ø Ø Ø 1 Ø Ø 43
RLC 3
ผลจากการ MASK
Ø Ø Ø Ø I/P Ø Ø Ø 40
Ø Ø Ø Ø Ø O/P Ø Ø 44
Ex. OUTPUT LOGIC IC 7404
ครั้ง
Ø Ø Ø Ø Ø Ø Ø 1
Ø Ø Ø Ø Ø Ø Ø Ø
Ø Ø Ø Ø Ø Ø Ø O/P 45
46
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
N
Y
39
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 39 ใหม่
รายงานผลที่ทดสอบทั้งหมด

ค. IC ประเภทที่ ๓ ใช้ SOCKET #1(มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๓



รูปที่ ก.๓ IC ประเภทที่ ๓

จากรูป ก.๓ มีทั้งหมด 4 GATE ในที่นี้แบ่งเป็น ๔ ส่วน (PART) ได้แก่ P_1, P_2, P_3 และ P_4

- P_1 มีขา ๒ และ ๓ เป็น INPUT/ ขา ๑ เป็น OUTPUT
- P_2 มีขา ๕ และ ๖ เป็น INPUT/ ขา ๔ เป็น OUTPUT
- P_3 มีขา ๘ และ ๙ เป็น INPUT/ ขา ๑๐ เป็นOUTPUT
- P_4 มีขา ๑๑ และ ๑๒ เป็น INPUT /ขา ๑๓ เป็น OUTPUT

โปรกแกรมที่ใช้ในการทดสอบ จะเป็นโปรแกรมย่อยที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่มีลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปเท่านั้นก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภทที่ ๓ นี้ได้แก่

- 2 INPUT NAND GATEเช่น 7401
- 2 INPUT NOR GATE เช่น7402, 7428 และ 7433

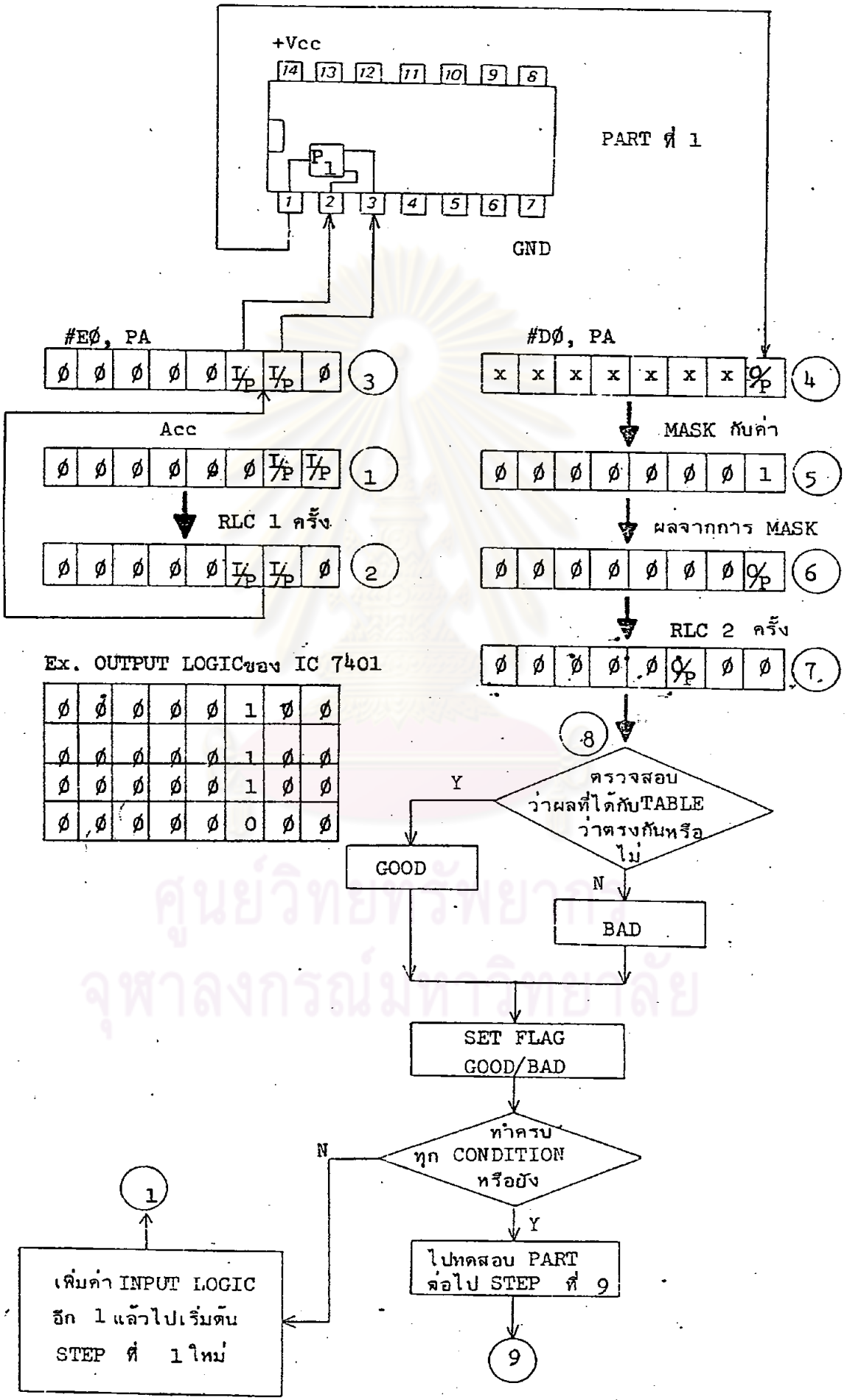
แนวความคิดในการเขียนโปรแกรม เพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๓

PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๓ ข้อมูลที่ SET ดังนี้

(จะต้อง SET ก่อนจะทดสอบแล้วบอกผู้ใช้ว่า READY)

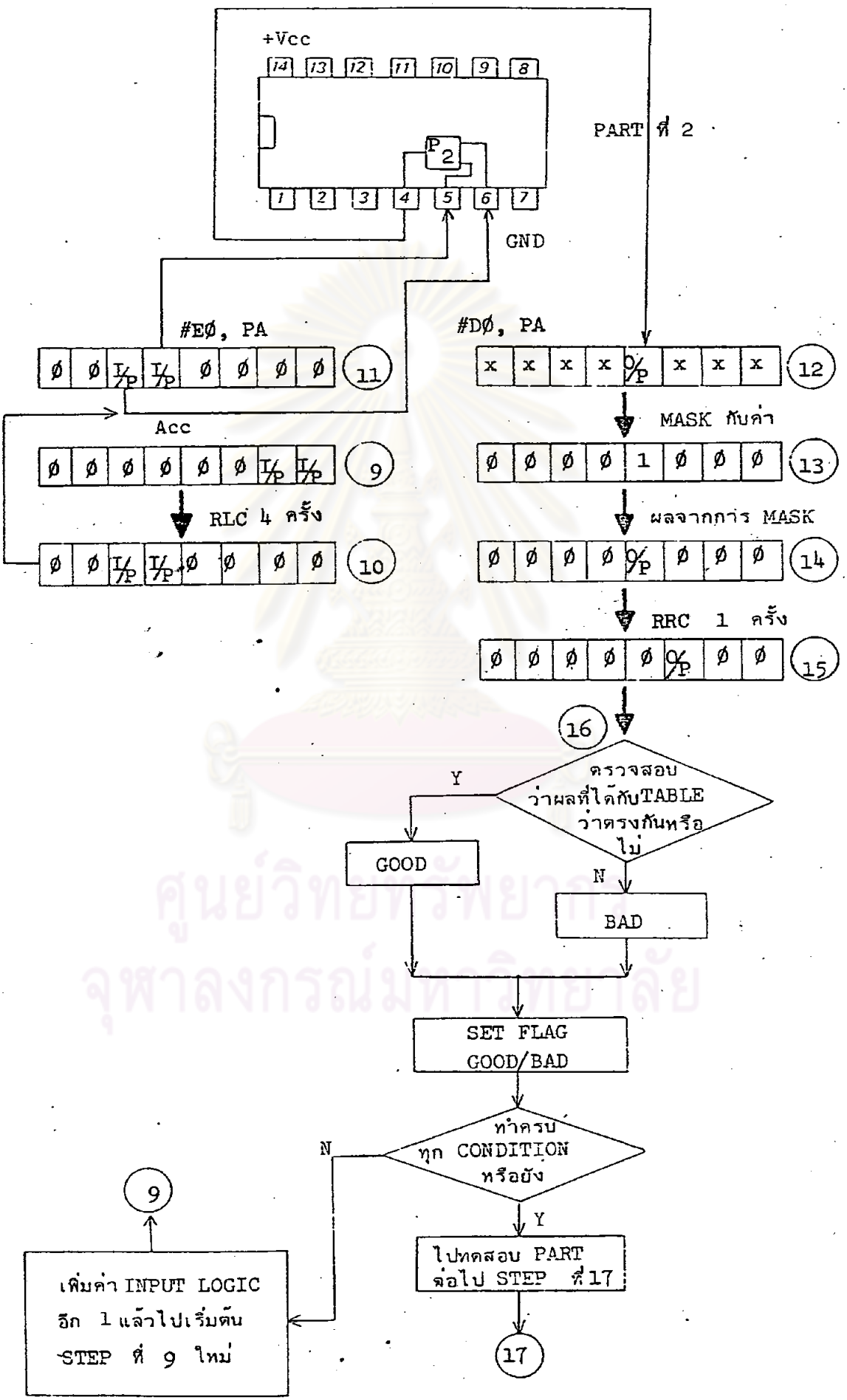
	#F1, PB								#FØ, PA								
x'Ø9'	Ø	Ø	Ø	Ø	1	Ø	Ø	1	Ø	Ø	Ø	Ø	1	Ø	Ø	1	x'Ø9'

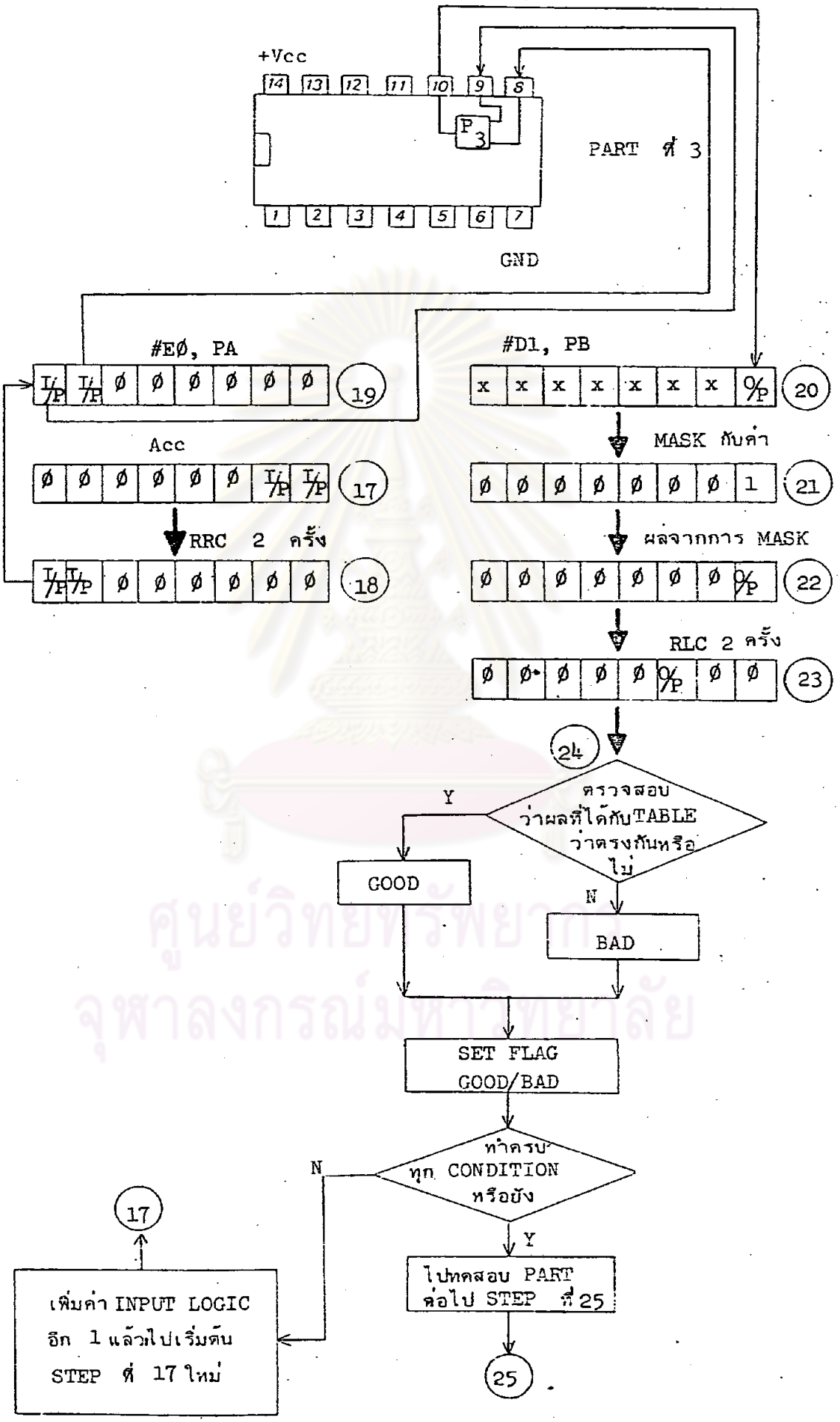
FLOW CHART ที่ ก.๓ FLOW CHART ของโปรแกรมทดสอบ IC ประเภทที่ ๓

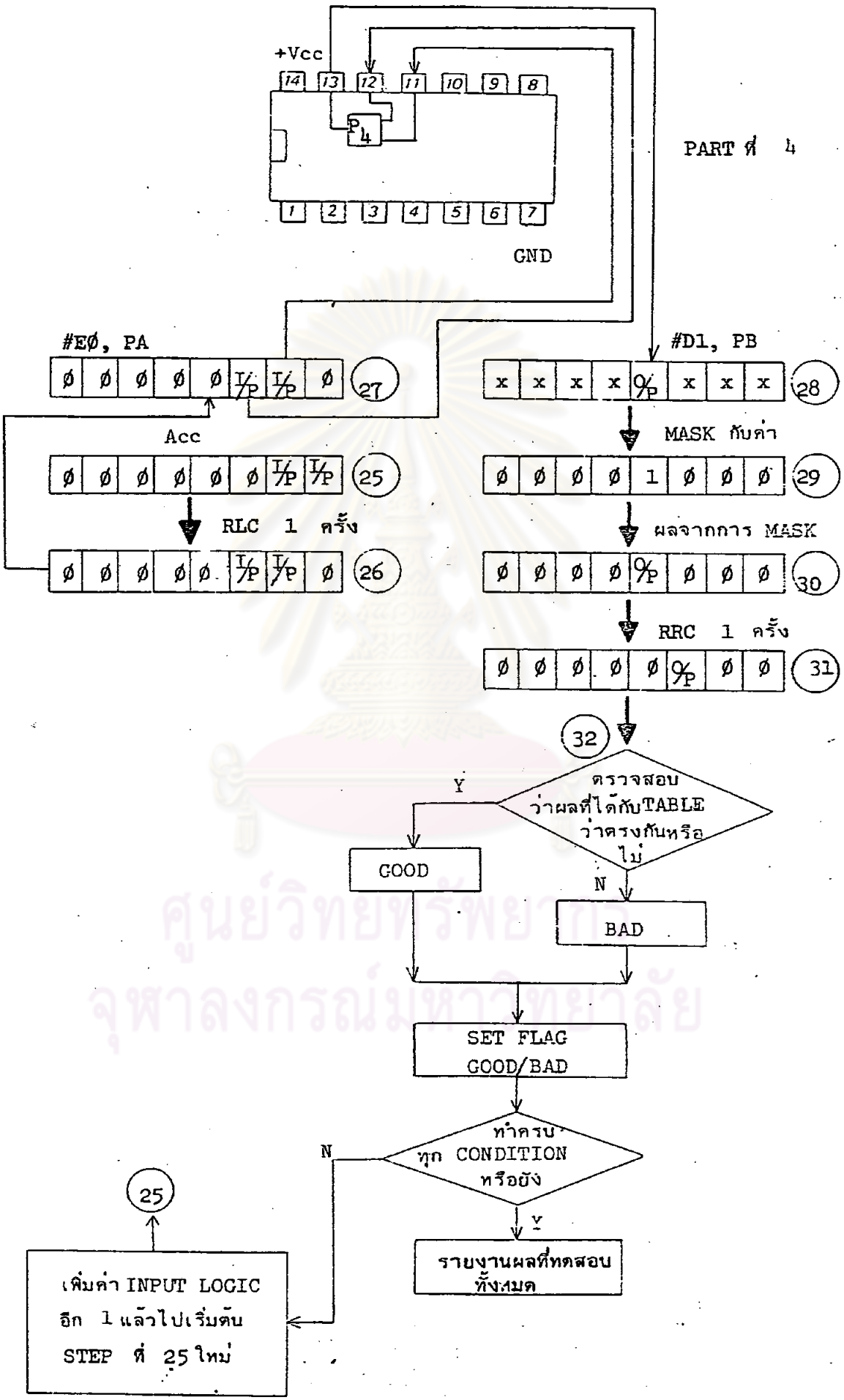


Ex. OUTPUT LOGICของ IC 7401

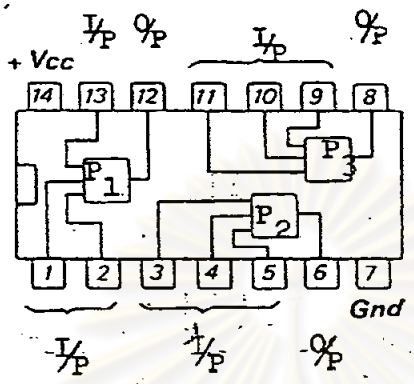
Ø	Ø	Ø	Ø	Ø	1	Ø	Ø
Ø	Ø	Ø	Ø	Ø	1	Ø	Ø
Ø	Ø	Ø	Ø	Ø	1	Ø	Ø
Ø	Ø	Ø	Ø	Ø	0	Ø	Ø







ง. <u>IC ประเภทที่ ๔</u> ใช้ SOCKET #1 (มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๔



รูปที่ ก.๔ IC ประเภทที่ ๔

จากรูปที่ ก.๔ มีทั้งหมด 3 GATE ในที่นี้แบ่งเป็น ๓ ส่วน (PART) ได้แก่ P_1, P_2, และ P_3

- P_1 มีขา ๑, ๒ และ ๑๓ เป็น INPUT/ ขา ๑๒ เป็น OUTPUT
- P_2 มีขา ๓, ๔ และ ๕ เป็น INPUT/ขา ๖ เป็น OUTPUT
- P_3 มีขา ๙, ๑๐ และ ๑๑ เป็น INPUT /ขา ๘ เป็น OUTPUT

โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อยที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่มีลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปเท่านั้น ก็สามารถทดสอบIC แต่ละชนิดได้

IC ประเภทที่ ๔ นี้ได้แก่

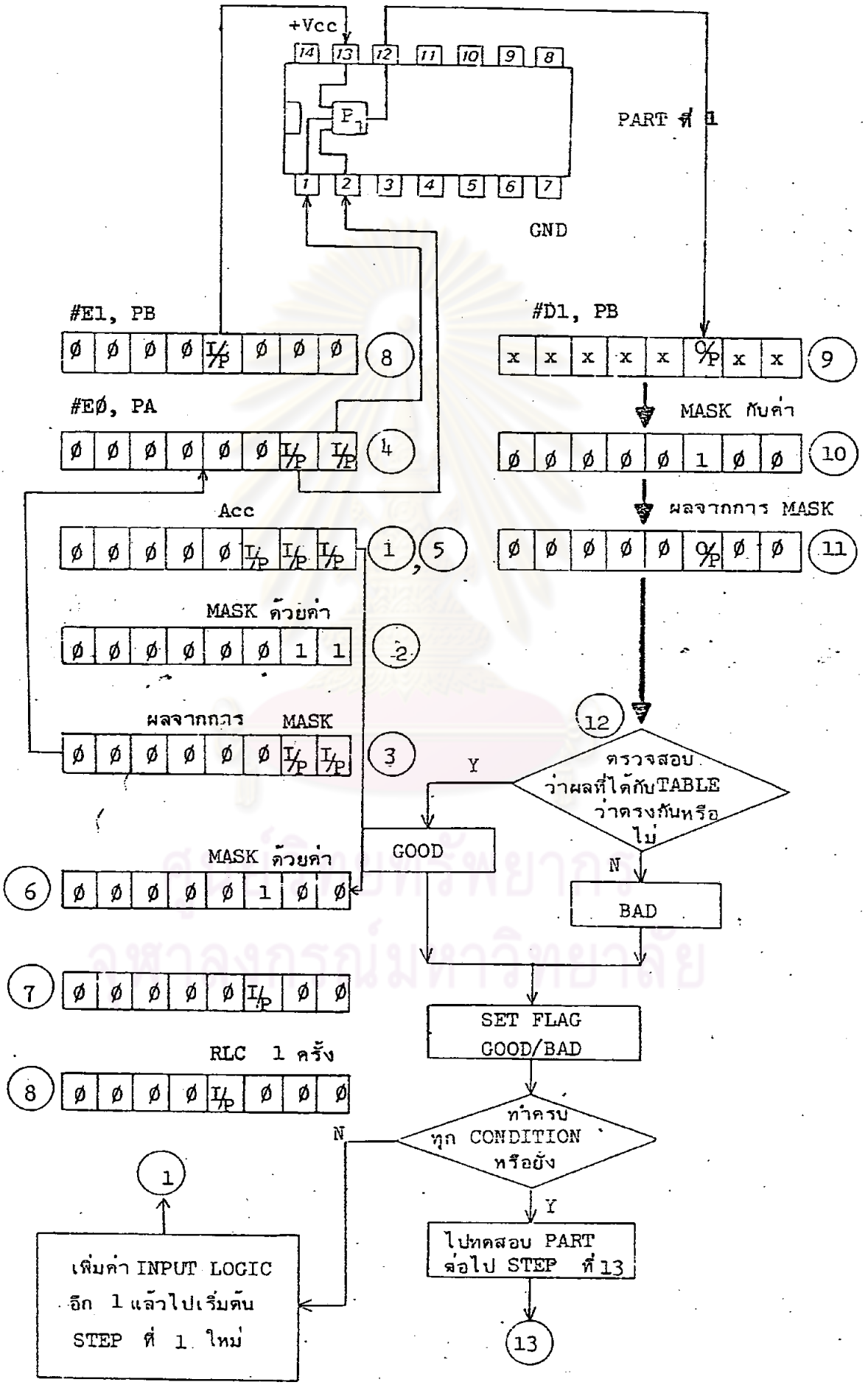
- 3 INPUT NAND GATE เช่น7410, และ 7412
- 3 INPUT NOR GATE เช่น 7427
- 3 INPUT AND GATE เช่น 7411, และ7415

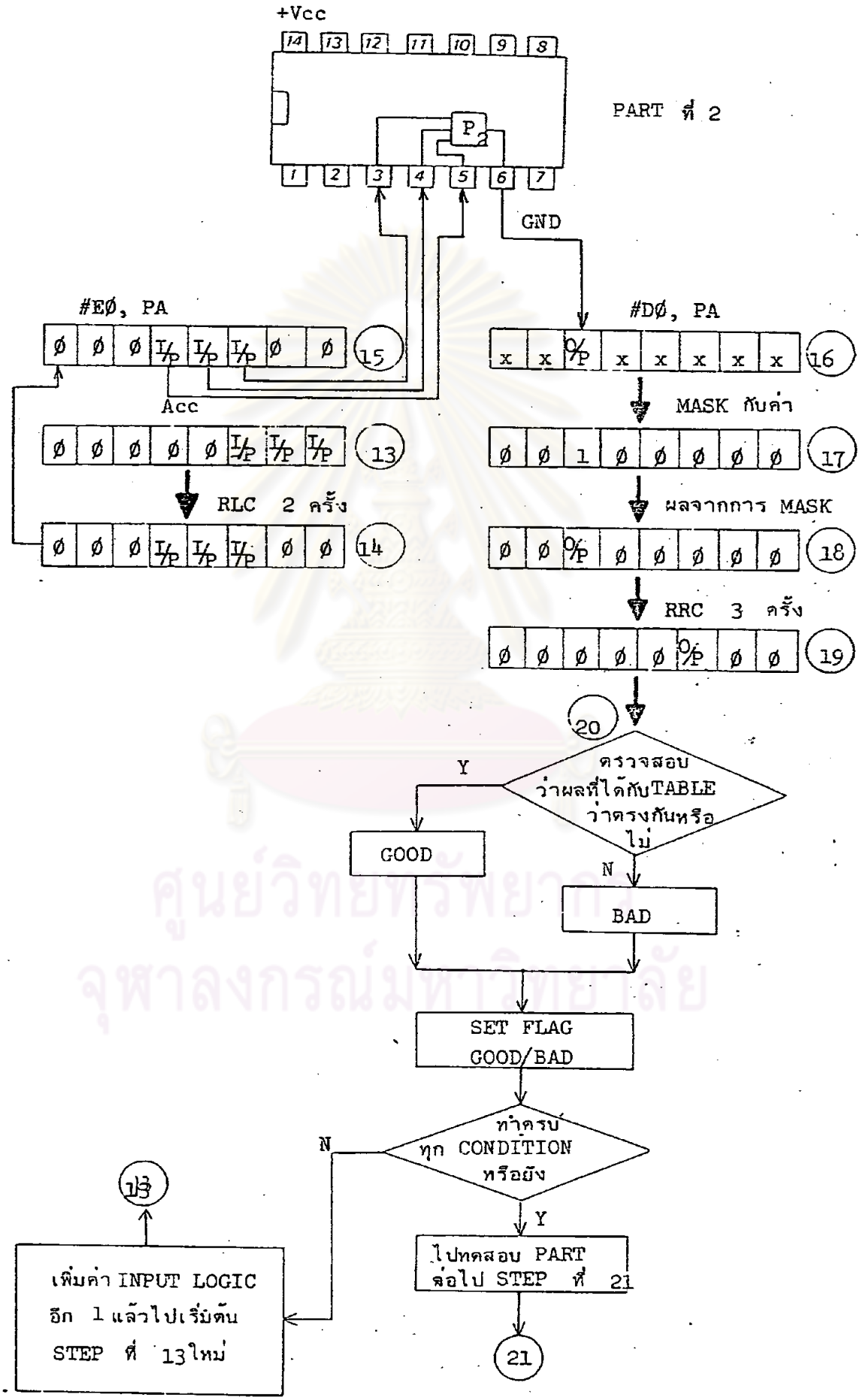
แนวความคิดในการเขียนโปรแกรม เพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๔

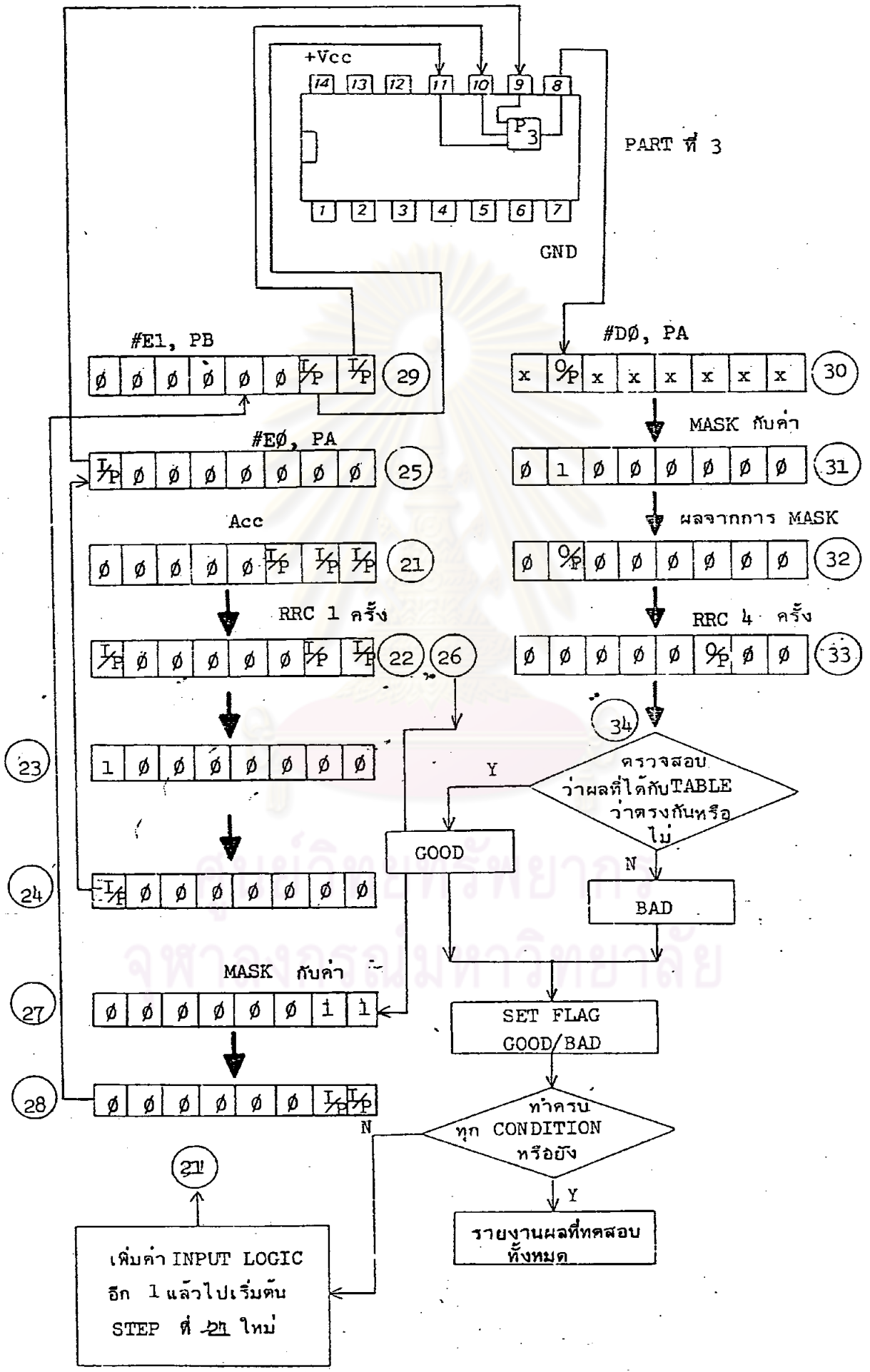
PIN CONTROL สำหรับ SET ขา ICประเภทที่ ๔ ข้อมูลที่ SET ดังนี้

	#F1, PB								#FØ, PA								
x'Ø4'	Ø	Ø	Ø	Ø	Ø	1	Ø	Ø	Ø	1	1	Ø	Ø	Ø	Ø	Ø	x'6Ø'

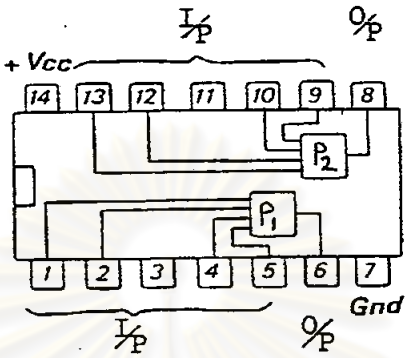
FLOW CHARTที่ ก.๔ FLOW CHART ของโปรแกรมทดสอบ IC ประเภทที่ ๔




+Vcc
14 13 12 11 10 9 8
P
1 2 3 4 5 6 7
PART ที่ 2
GND
#EØ, PA
Ø Ø Ø I/P I/P I/P Ø Ø
15
Acc
Ø Ø Ø Ø Ø I/P I/P I/P
13
RLC 2 ครั้ง
Ø Ø Ø I/P I/P I/P Ø Ø
14
#DØ, PA
x x O/P x x x x x
16
MASK กับค่า
Ø Ø 1 Ø Ø Ø Ø Ø
17
ผลจากการ MASK
Ø Ø O/P Ø Ø Ø Ø Ø
18
RRC 3 ครั้ง
Ø Ø Ø Ø Ø O/P Ø Ø
19
20
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
N
Y
13
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 13 ใหม่
ไปทดสอบ PART ต่อไป STEP ที่ 21
21


+Vcc
14 13 12 11 10 9 8
P3
1 2 3 4 5 6 7
GND
PART ที่ 3
#E1, PB
29
#DØ, PA
30
#EØ, PA
25
MASK กับค่า
31
Acc
21
ผลจากการ MASK
32
RRC 1 ครั้ง
22
26
RRC 4 ครั้ง
33
23
34
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
24
MASK กับค่า
27
SET FLAG GOOD/BAD
28
ทำครบทุก CONDITION หรือยัง
N
Y
รายงานผลที่ทดสอบทั้งหมด
21
เพิ่มค่า INPUT LOGIC อีก 1 แล้วไปเริ่มต้น STEP ที่ 21 ใหม่

จ. IC ประเภทที่ ๔ ใช้ SOCKET # 1 (มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๔



รูปที่ ก.๔ IC ประเภทที่ ๔

จากรูปที่ ก.๔ มีทั้งหมด ๒ GATE ในที่นี้แบ่งเป็น ๒ ส่วน (PART) ได้แก่ P_1 และ P_2

- P_1 มีขา ๑, ๒, ๔, และ ๕ เป็น INPUT/ขา ๖ เป็น OUTPUT
- P_2 มีขา ๔, ๑๐, ๑๑ และ ๑๓ เป็น INPUT /ขา ๘ เป็น OUTPUT

โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อยที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่ลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปเท่านั้นก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภทที่ ๔ นี้ได้แก่

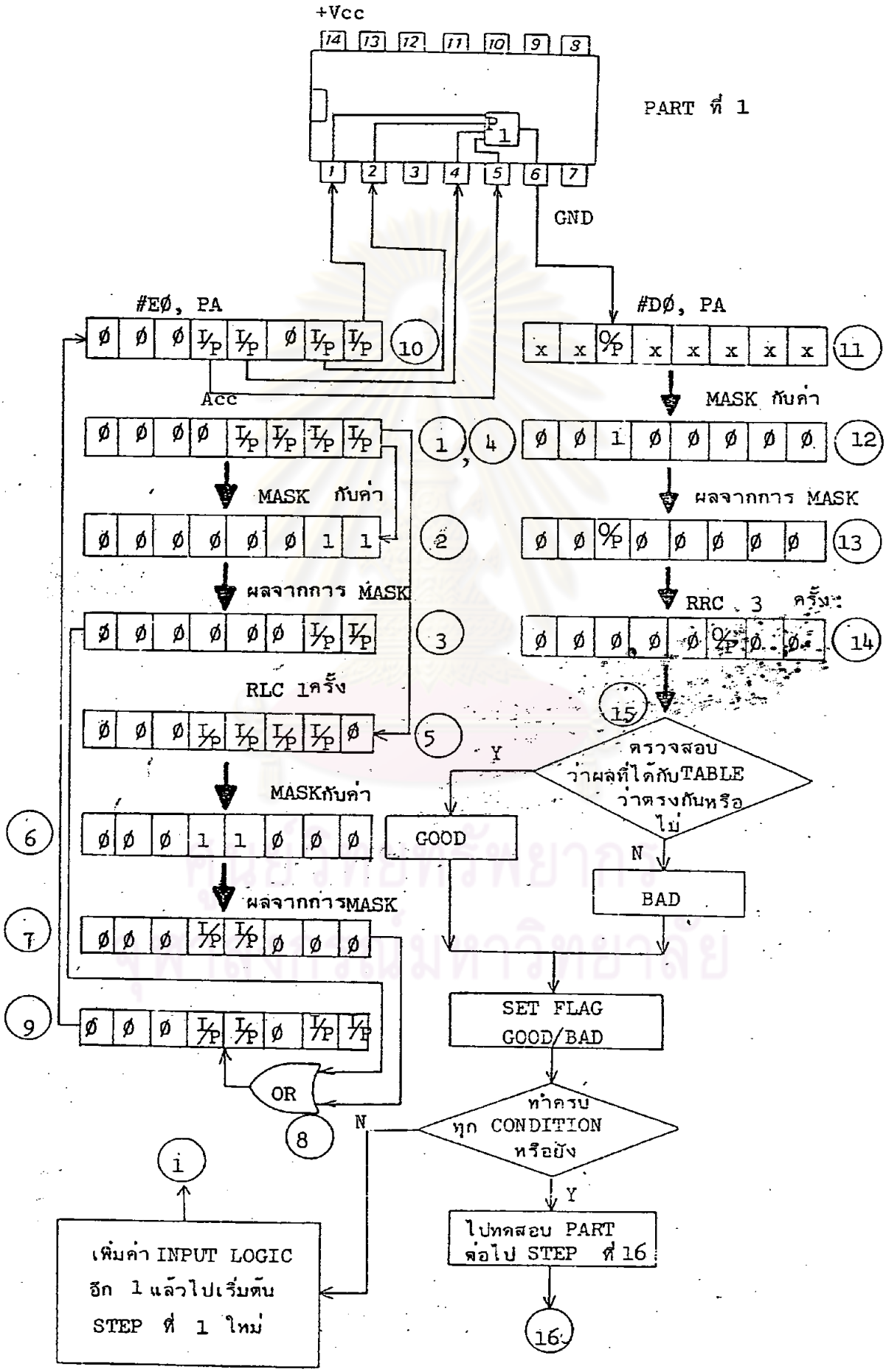
- DUAL 4 INPUT NAND GATE เช่น 7420, 7422, 7440, 74140 และ 74C20

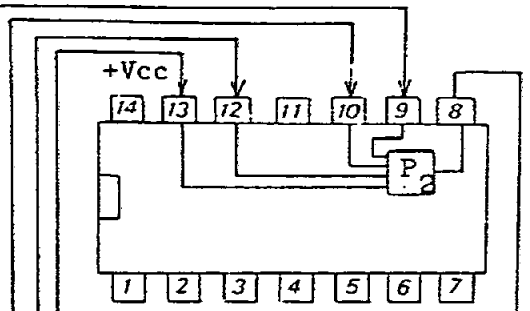
แนวความคิดในการเขียนโปรแกรมเพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๔

PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๔ ข้อมูลที่ SET ดังนี้

	#F1, PB								#FØ, PA								
X'ØØ'	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	1	1	Ø	Ø	Ø	Ø	Ø	X'6Ø'

FLOW CHART ที่ ก.๔FLOW CHART ของโปรแกรมทดสอบ IC ประเภทที่ ๔





PART ที่ 2

#EØ, PA

I/P	Ø	Ø	Ø	Ø	Ø	Ø	Ø

28

#E1, CRB

Ø	Ø	Ø	Ø	I/P	I/P	Ø	I/P

27

Ø	Ø	Ø	Ø	I/P	I/P	I/P	I/P

16

17, 20, 23 RRC 1 ครั้ง

I/P	Ø	Ø	Ø	Ø	I/P	I/P	I/P

MASK กับค่า

1	Ø	Ø	Ø	Ø	Ø	Ø	Ø

18

ผลจากการMASK

$\overline{I/P}$	Ø	Ø	Ø	Ø	Ø	Ø	Ø

19

MASK กับค่า

Ø	Ø	Ø	Ø	Ø	Ø	Ø	1

21

ผลจากการMASK

Ø	Ø	Ø	Ø	Ø	Ø	Ø	I/P

22

MASK กับค่า

Ø	Ø	Ø	Ø	1	1	Ø	Ø

24

ผลจากการMASK

Ø	Ø	Ø	Ø	I/P	I/P	Ø	Ø

25

OR

26

#DØ, PA

x	O/P	x	x	x	x	x	x

29

MASK กับค่า

Ø	1	Ø	Ø	Ø	Ø	Ø	Ø

30

ผลจากการ MASK

Ø	O/P	Ø	Ø	Ø	Ø	Ø	Ø

31

RRC 4 ครั้ง

Ø	Ø	Ø	Ø	Ø	O/P	Ø	Ø

32

33 ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่

Y → GOOD

N → BAD

GOOD / BAD → SET FLAG GOOD/BAD

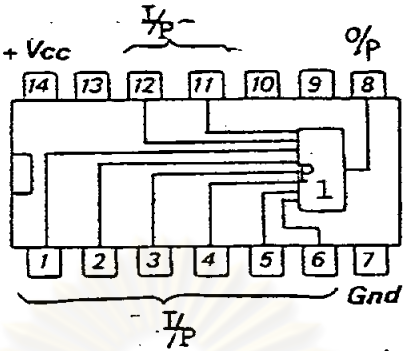
ทำครบทุก CONDITION หรือยัง

N → 16

Y → รายงานผลการทดสอบทั้งหมด

16 → เพิ่มค่า INPUT LOGIC อีกหนึ่งแล้วไปเริ่มต้น STEP ที่ 16 ใหม่

๓. IC ประเภทที่ ๖ ใช้ SOCKET # 1 (มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๖



รูปที่ ก.๖ IC ประเภทที่ ๖

จากรูป ก.๖ มีทั้งหมด ๑ GATE ในที่นี้แบ่งเป็น ๑ ส่วน (PART) ได้แก่ P_1

- P_1 มีขา ๑, ๒, ๓, ๔, ๕, ๖ ๑๑ และ ๑๒ เป็น INPUT /ขา ๘ เป็น OUTPUT

โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อยที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกันแต่มีลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปเท่านั้นก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภทที่ ๖ นี้ ได้แก่

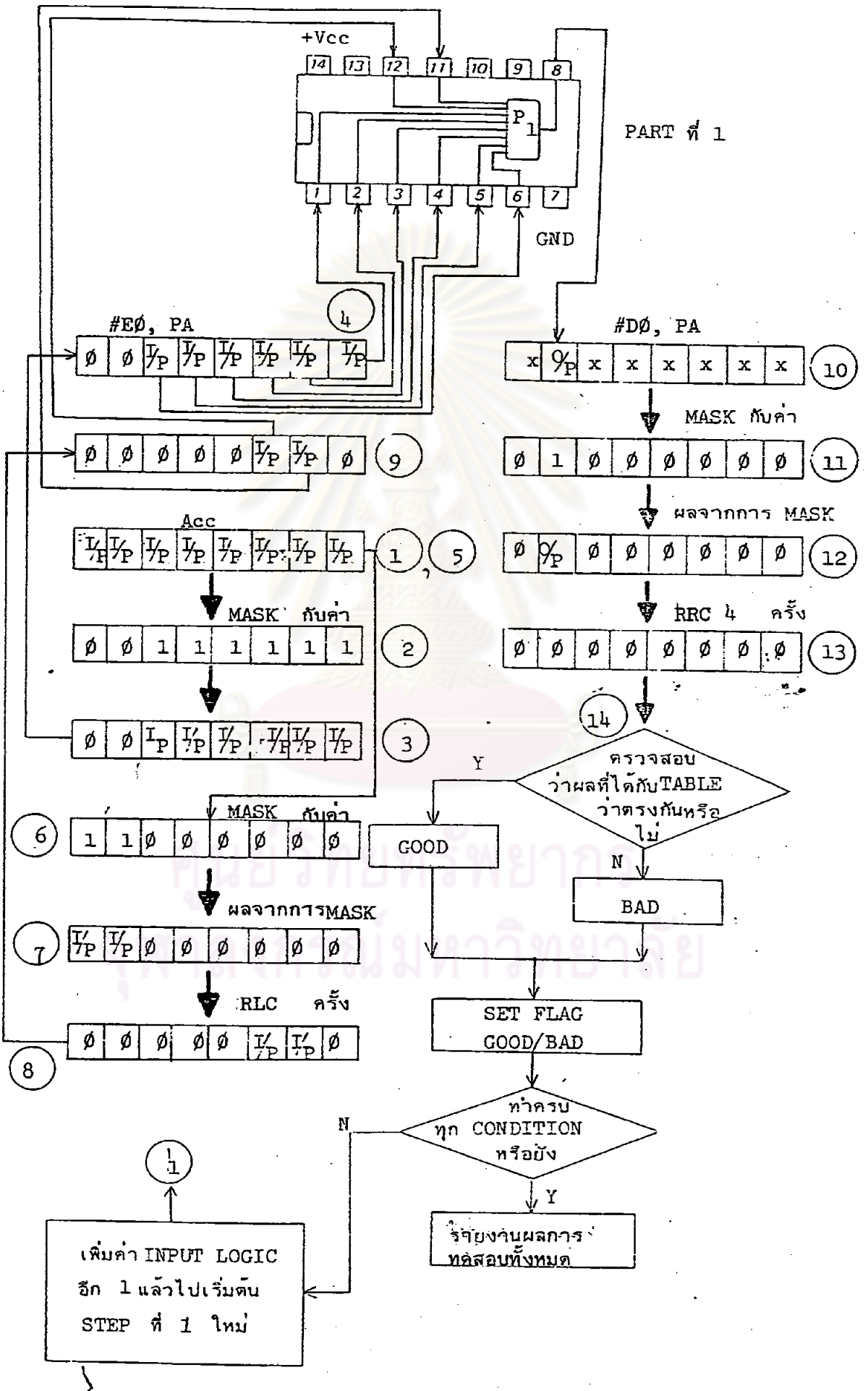
- 8 INPUT NAND GATE เช่น 7430

แนวความคิดในการเขียนโปรแกรมเพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๖

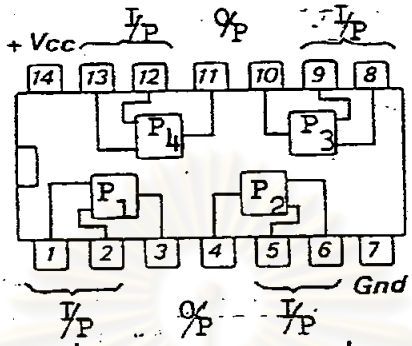
PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๖ ข้อมูลที่ SET ดังนี้

	#F1, PB								#FØ, PA								
X'ØØ'	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	Ø	1	Ø	Ø	Ø	Ø	Ø	Ø	X'4Ø'

FLOW CHART ที่ ก.๖ FLOW CHART ของโปรแกรมทดสอบ IC ประเภทที่ 6



ช. IC ประเภทที่ ๗ ใช้ SOCKET # 1 (มี ๑๔ ขา) Vcc อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๗



รูปที่ ก.๗ IC ประเภทที่ ๗

จากรูป ก.๗ มีทั้งหมด 4 GATEในที่นี้แบ่งเป็น ๔ ส่วน (PART) ได้แก่ P1, P2, P3 และP4

- P_1 มีขา ๑ และ ๒ เป็น INPUT/ ขา ๓ เป็น OUTPUT
- P_2 มีขา ๕ และ ๖ เป็น INPUT/ ขา ๔ เป็น OUTPUT
- P_3 มีขา ๘ และ ๙ เป็น INPUT/ ขา ๑๐ เป็น OUTPUT
- P_4 มีขา ๑๒ และ ๑๓ เป็น INPUT/ ขา ๑๑ เป็น OUTPUT

โปรแกรมที่ใช้ในการทดสอบจะเป็นโปรแกรมย่อมที่สามารถใช้ได้กับ IC GATE ที่ทำหน้าที่ต่างกัน แต่มีลักษณะของขาเหมือนกัน เพียงแต่เปลี่ยน LOGIC OUTPUT TABLE ที่แตกต่างกันออกไปแทน ก็สามารถทดสอบ IC แต่ละชนิดได้

IC ประเภที่ ๗ นี้ได้แก่

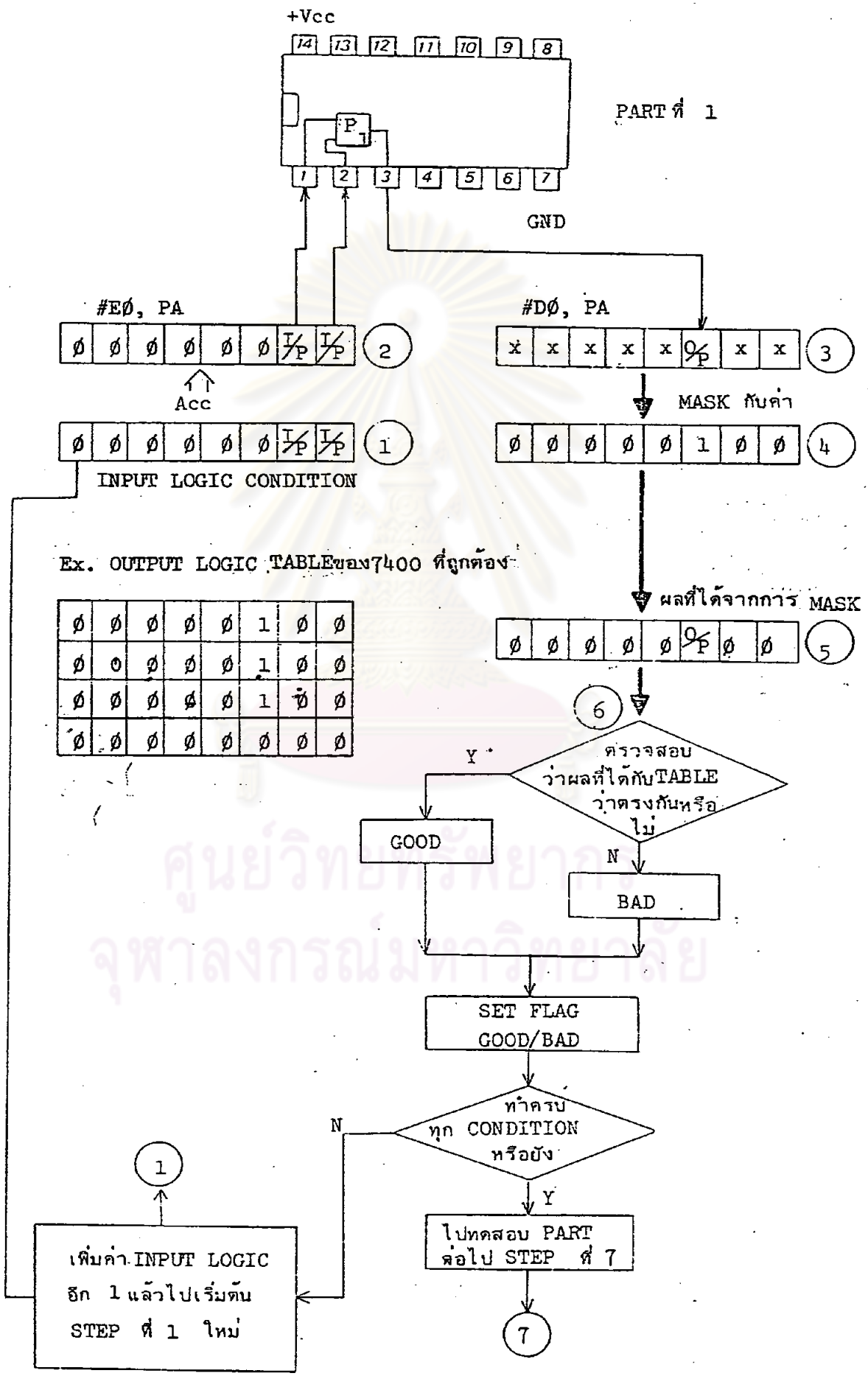
- QUAD 2 INPUT EXCLUSIVE OR GATE เช่น 74LS266, 74LS386, 74C86, 4030 และ 4070 (CMOS)
- QUAD 2 INPUT NAND GATE เช่น 4011 (CMOS)
- QUAD 2 INPUT NOR GATE เช่น 4001 (CMOS)
- QUAD 2 INPUT OR GATE เช่น 4071 (CMOS)

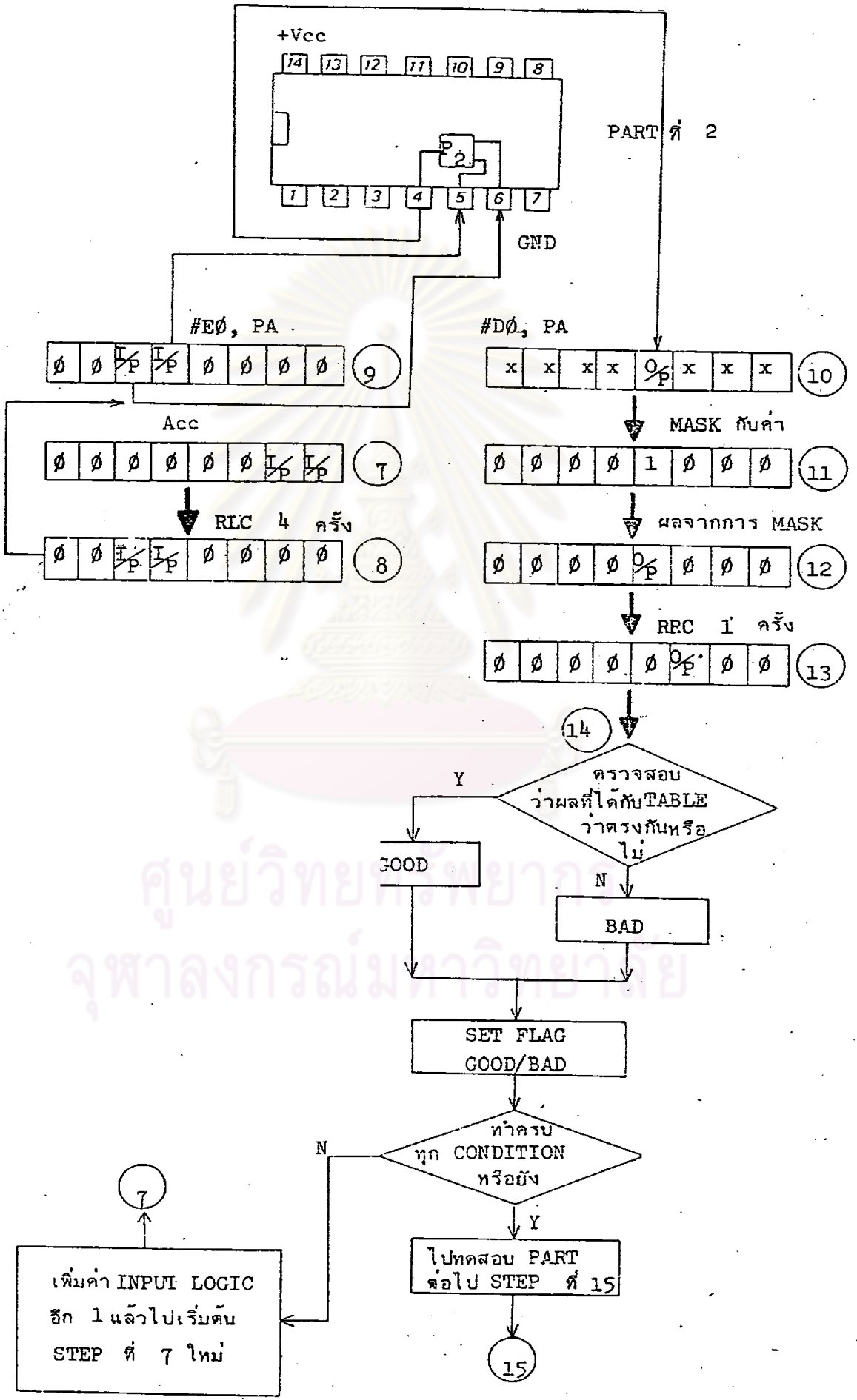
แนวความคิดในการเขียนโปรแกรมเพื่อจะทดสอบ IC พวกนี้ ดู FLOW CHART ที่ ก.๗

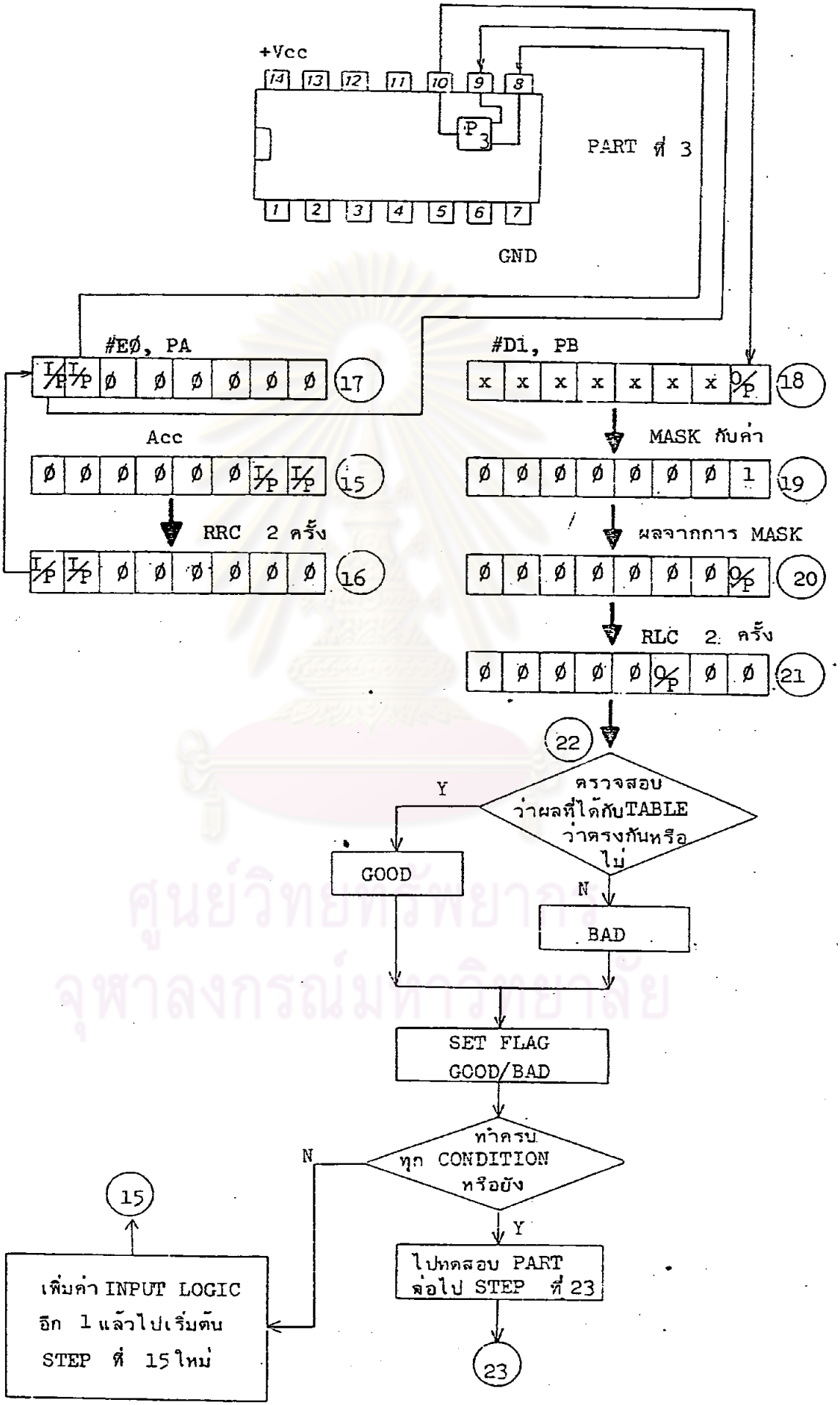
PIN CONTROL สำหรับ SET ขา IC ประเภทที่ ๗ ข้อมูลที่ SET ดังนี้ (จะต้อง SET ก่อนจะทดสอบแล้วบอกผู้ใช้ว่า READY)

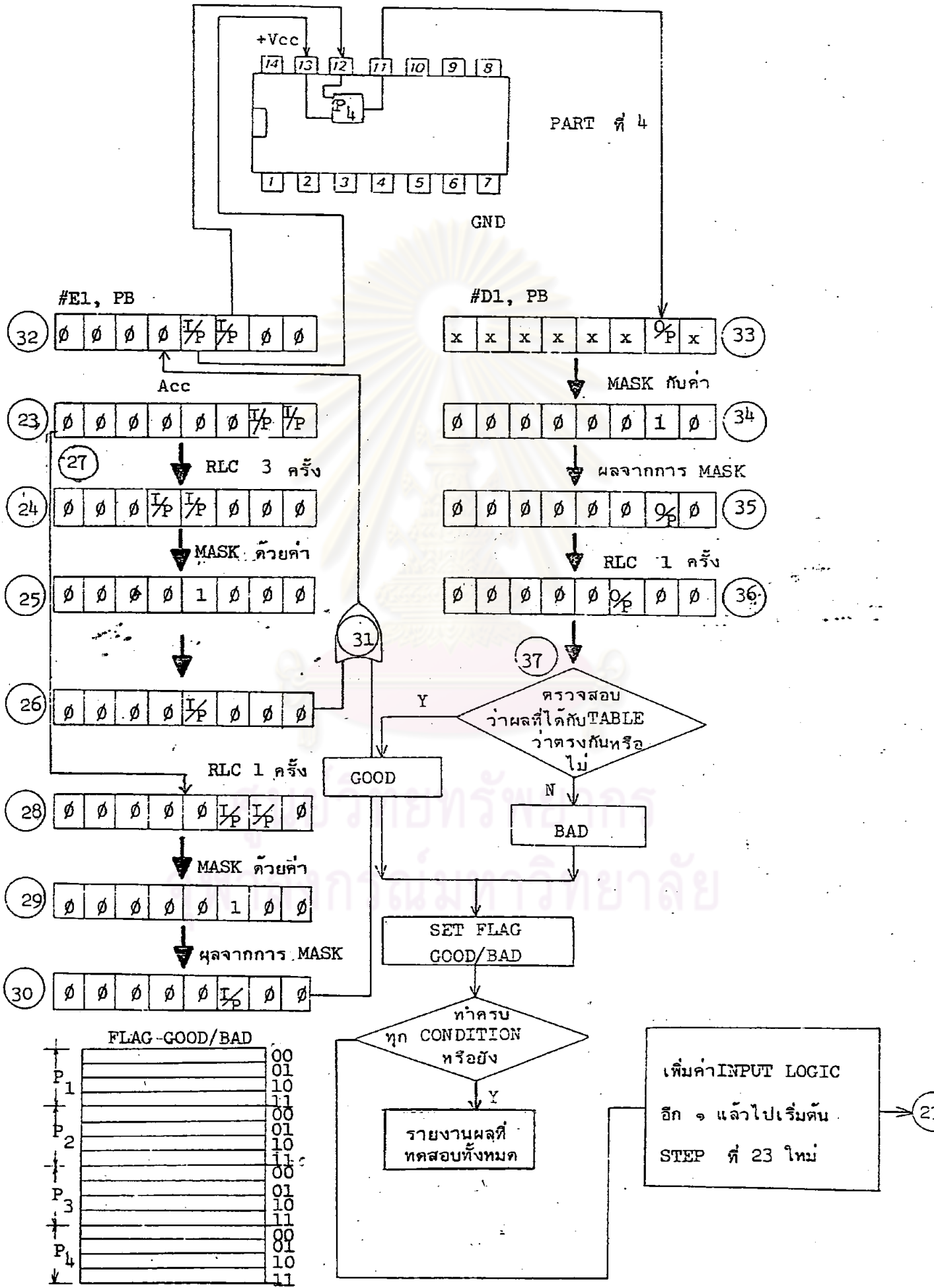
	#F1, PB								#FØ, PA								
X 'Ø3'	Ø	Ø	Ø	Ø	Ø	Ø	1	1	Ø	Ø	Ø	Ø	1	1	Ø	Ø	X'ØC'

FLOW CHART ที่ ก.๗ FLOW CHART ของโปรแกรมทดสอบIC ประเภทที่ ๗

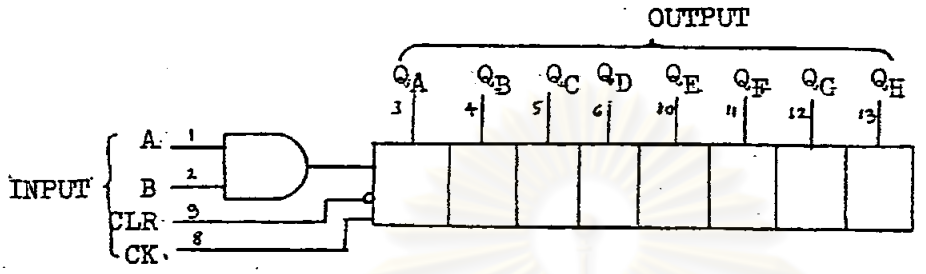







+Vcc
14 13 12 11 10 9 8
P4
PART ที่ 4
1 2 3 4 5 6 7
GND
#E1, PB
32
Ø Ø Ø Ø I/P I/P Ø Ø
Acc
23
Ø Ø Ø Ø Ø Ø I/P I/P
27
RLC 3 ครั้ง
24
Ø Ø Ø I/P I/P Ø Ø Ø
MASK ด้วยค่า
25
Ø Ø Ø Ø 1 Ø Ø Ø
26
Ø Ø Ø Ø I/P Ø Ø Ø
31
RLC 1 ครั้ง
28
Ø Ø Ø Ø Ø I/P I/P Ø
MASK ด้วยค่า
29
Ø Ø Ø Ø Ø 1 Ø Ø
ผลจากการ MASK
30
Ø Ø Ø Ø Ø I/P Ø Ø
#D1, PB
33
x x x x x x O/P x
MASK กับค่า
34
Ø Ø Ø Ø Ø Ø 1 Ø
ผลจากการ MASK
35
Ø Ø Ø Ø Ø Ø O/P Ø
RLC 1 ครั้ง
36
Ø Ø Ø Ø Ø O/P Ø Ø
37
ตรวจสอบ ว่าผลที่ได้กับTABLE ว่าตรงกันหรือไม่
Y
N
GOOD
BAD
SET FLAG GOOD/BAD
ทำครบทุก CONDITION หรือยัง
Y
รายงานผลที่ทดสอบทั้งหมด
เพิ่มค่าINPUT LOGIC อีก ๑ แล้วไปเริ่มต้น STEP ที่ 23 ใหม่
23
FLAG-GOOD/BAD
P1 00 01 10 11
P2 00 01 10 11
P3 00 01 10 11
P4 00 01 10 11

ข. IC ประเภทที่ ๔ ใช้ SOCKET # 1 (มี ๑๔ ขา V_{cc} อยู่ขา ๑๔ และ GND อยู่ขา ๗) มีลักษณะภายในดังรูปที่ ก.๔ (สำหรับทดสอบ IC เบอร์ ๗๔๑๖๔ โดยเฉพาะ)



V_{cc} PIN 14

GND PIN 7

รูปที่ ก.๔ ลักษณะภายใน IC เบอร์ 74164

การทดสอบ IC เบอร์ 74164 แบ่งเป็นขั้นตอนดังต่อไปนี้

ขั้นตอนที่ ๑ CLEAR SHIFT REGISTER โดย SET ขา CLEAR (CLR) ให้มีขา LOGIC"0" รับผลเข้าระบบตรวจสอบว่าได้ข้อมูลจากOUTPUTเป็น LOGIC "0" จริง หรือไม่

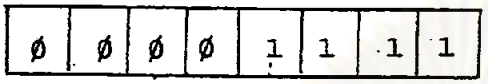
ขั้นตอนที่ ๒ ทดสอบ CONDITION ที่ A และ B เป็น L(LOW) ทั้งคู่ (DATA INPUT = ØØ) โดย SET ให้ขา CLR เป็น H(HIGH) หมายถึง วงจร SHIFT REGISTER สามารถทำงานได้แล้วSIMULATE สัญญาณ CLOCK (CK) โดยใช้การส่งข้อมูลสลับกันระหว่าง H กับ L จะได้ลักษณะเป็นสัญญาณ CLOCK แล้วรับผลกลับ ประมวลผลที่ได้ว่าถูกต้องหรือไม่แล้วทำขั้นตอนที่ ๓ ต่อไป

ขั้นตอนที่ ๓, ๔ และ ๕ ทำเช่นเดียวกับขั้นตอนที่ ๒ แต่ขา INPUT A กับ B มีค่าดังนี้

ขั้นตอนที่	B	A
3	0	1
4	1	0
5	1	1

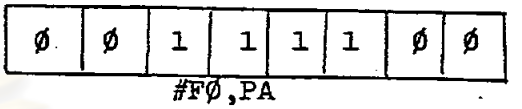
โปรแกรมทดสอบจะทำตาม FLOW CHART ที่ ก.๘

SET PIN CONTROLLER สำหรับIC เบอร์ 74164 ดังนี้



#F1 ,PB

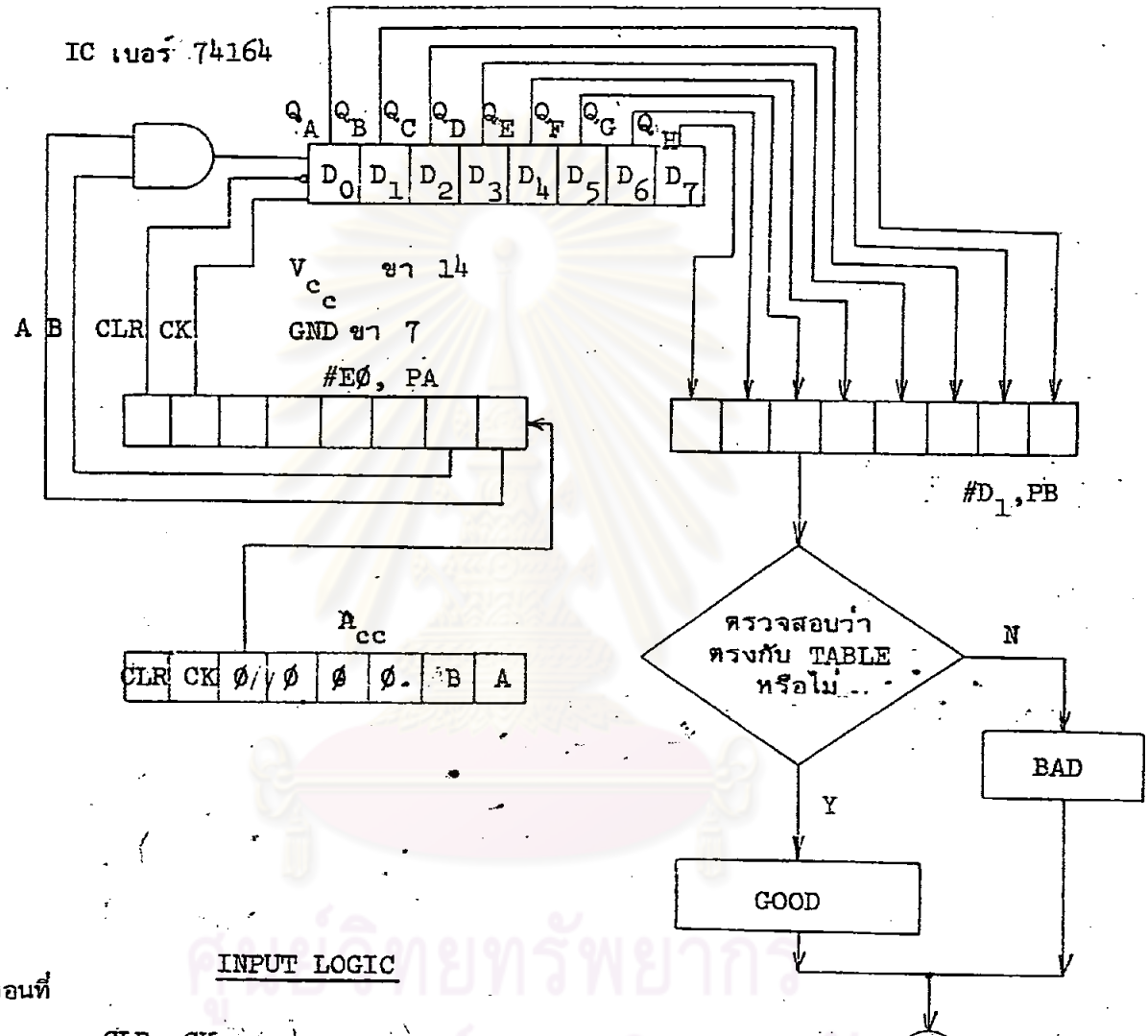
X'ØF'



#FØ,PA

X'2C'

FLOW CHART ที่ ก.๔ แสดงFLOW CHART ของโปรแกรมทดสอบ ICเบอร์ 74164 ประเภทที่ ๔



ขั้นตอนที่

INPUT LOGIC

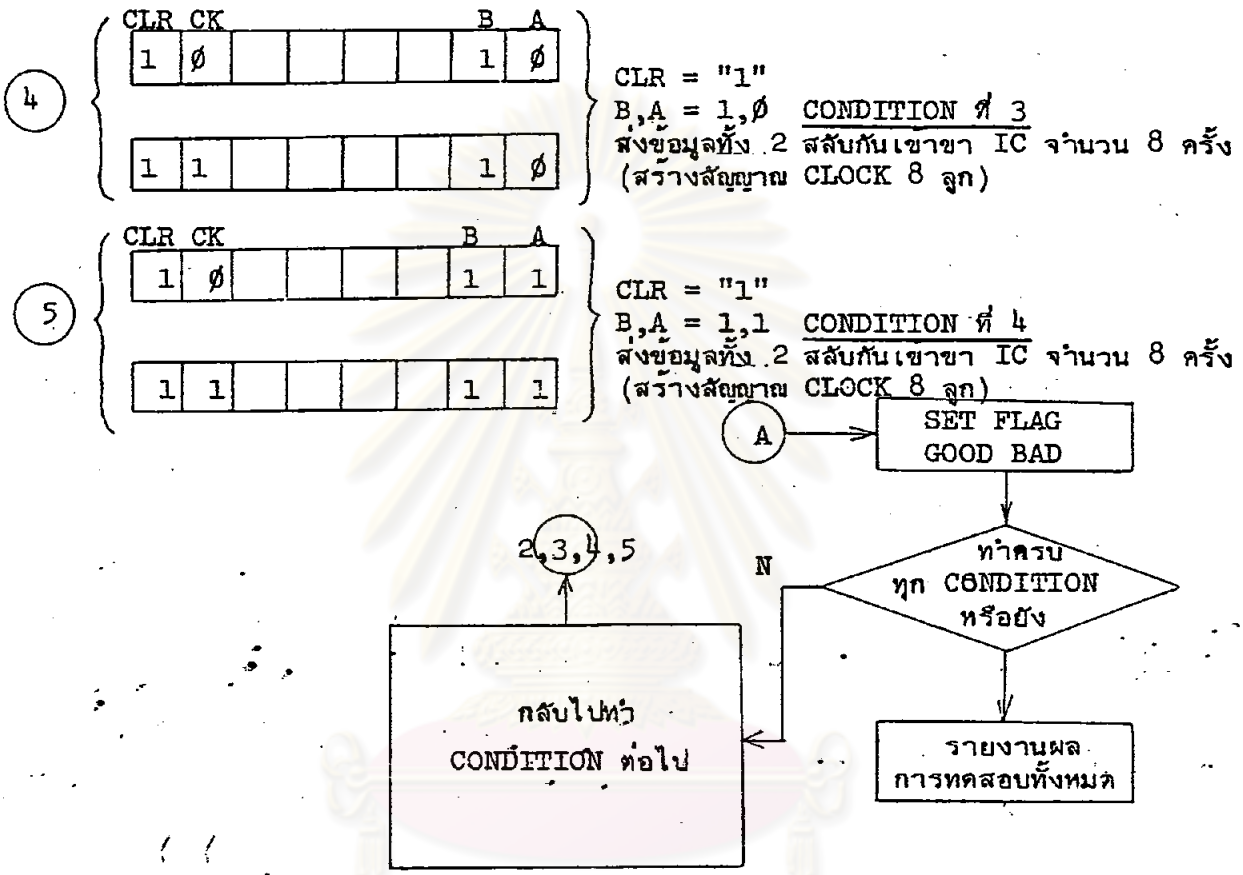
(1) CLR CK — Ø Ø Ø Ø Ø Ø Ø Ø ← CLEAR SHIFT REGISTER CLR = "0"

(2) CLR CK — 1 Ø ... Ø Ø ; 1 1 ... Ø Ø

CLR = "1"
B,A = ØØ CONDITION ที่ 1
ส่งข้อมูลทั้ง 2 สลับกันเข้าขา IC จำนวน 8 ครั้ง
(สร้างสัญญาณ CLOCK 8 ลูก)

(3) CLR CK — 1 Ø ... Ø 1 ; 1 1 ... Ø 1

CLR = "1"
B,A = Ø1 CONDITION ที่ 2
ส่งข้อมูลทั้ง 2 สลับกันเข้าขา IC จำนวน 8 ครั้ง
(สร้างสัญญาณ CLOCK 8 ลูก)



ตาราง OUTPUT DATA TABLE

STEP	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
1	ø	ø	ø	ø	ø	ø	ø	ø	
2	ø	ø	ø	ø	ø	ø	ø	ø	
3	ø	ø	ø	ø	ø	ø	ø	ø	
4	ø	ø	ø	ø	ø	ø	ø	ø	
5	ø	ø	ø	ø	ø	ø	ø	1	CK1
	ø	ø	ø	ø	ø	ø	1	1	CK2
	ø	ø	ø	ø	ø	1	1	1	CK3
	ø	ø	ø	ø	1	1	1	1	CK4
	ø	ø	ø	1	1	1	1	1	CK5
	ø	ø	1	1	1	1	1	1	CK6
	ø	1	1	1	1	1	1	1	CK7
	1	1	1	1	1	1	1	1	CK8

OUTPUT ของ SHIFT REGISTER
ซึ่งมี 8 แบบ เมื่อ INPUT BA = 1

ภาคผนวก ข

โปร แกรม START

เริ่ม START PROGRAM ที่ ADDRESS ๒๐๐๐ นี้ เป็น PROGRAM INITIALIZE DEVICE ทั้งหมดว่าให้ทำ MODE ใด ตามที่ออกแบบ HARDWARE ไว้

```
2000 3E9B       MVI  A,9B   ; SET IC 8255 ให้เป็น INPUT ทั้ง 3 PORT
2002 D3D3       OUT  D3     ; IC LOGIC OUT PUT MONITOR
2004 3E80       MVI  A,80   ; SET IC 8255 ให้เป็น OUTPUT ทั้ง 3 PORT
2006 D3E3       OUT  E3     ; IC INPUT LOGIC DRIVER
2008 3E80       MVI  A,80   ; SET IC 8255 ให้เป็น OUTPUT ทั้ง 3 PORT
200A D3F3       OUT  F3     ; IC PIN CONTROLLER
200C 3E90       MVI  A,90   ; SET IC 8255 ให้ PORT A เป็น INPUT, PORT B
                            ; และ PORT C เป็น OUTPUT
200E D363       OUT  63     ; KEYBOARD และ SOCKET DISPLAY
2010 3E80       MVI  A,80   ; SET IC 8255 ให้เป็น OUTPUT ทั้ง 3 PORT
2012 D3B3       OUT  B3     ; BINARY INPUT DISPLAY
2014 3E80       MVI  A,80   ; SET IC 8255 ให้เป็น OUTPUT ทั้ง 3 PORT
2016 D3C3       OUT  C3     ; BINARY OUTPUT DISPLAY
2018 3E00       MVI  A,00   ; ตัดเสียงกระดิ่ง
201A D300       OUT  00     ; CLEAR ALARM
201C 00         NOP         ;
201D 00         NOP         ;
201E 00         NOP         ;
201F 00         NOP         ;
2020 CD4626     CALL 264C   ; เรียก PROGRAM ที่รับเลขเบอร์ IC จาก
                            ; KEYBOARD และ DISPLAY
2023 21004E     LXI  H,4E00 ; MEMORY ที่เก็บหมายเลขเบอร์ IC ที่กดเข้ามาที่
                            ; ADDRESS 4E00 เลขตัวแรก
2026 7E         MOV  A,M    ; นำค่า MEMORY มาตรวจหมายเลขอะไร
2027 00         NOP         ;
```

```
2028 00         NOP          ;
2029 00         NOP          ;
202A 00         NOP          ;
202B 00         NOP          ;
202C FE05       CPI  05      ; ตรวจเลขว่าเป็นเลข 5 หรือไม่
202E CA5020     JZ   2050    ; ถ้าใช่ไปทำ PROGRAM ที่ ADDRESS 2050 (ตรวจหมายเลขถัดไป)
2031 FE07       CPI  07      ; ถ้าไม่ใช่ตรวจว่าเป็นเลข 7 หรือไม่
2033 CA5020     JZ   2050    ; ถ้าใช่ไปทำ PROGRAM ที่ ADDRESS 2050 (ตรวจหมายเลขถัดไป)
2036 00         NOP          ;
2037 00         NOP          ;
2038 00         NOP          ;
2039 00         NOP          ;
203A 00         NOP          ;
203B 00         NOP          ;
203C 00         NOP          ;
203D 00         NOP          ;
203E 00         NOP          ;
203F 00         NOP          ;
2040 FEFF       CPI  FF      ; ตรวจว่าเป็นค่า FF หรือไม่
2042 CA4820     JZ   2048    ; ถ้าใช่กลับไปเริ่มตรวจหมายเลขใหม่
2045 CD4C26     CALL 264C    ; ไปรับหมายเลขใหม่(ต้องกดเบอร์ IC ใหม่)
2048 C32020     JMP  2020    ; กลับไปตรวจหมายเลขใหม่
```

โปรแกรม P1

ส่งเสียง ALARM TONE ยาวและตรวจ KEY ว่ากดอะไรเข้ามา

```
204B D310       OUT  10      ; ส่งเสียง ALARM TONE ยาว
204D C3B920     JMP  20B9    ; กลับไปตรวจ KEY ว่ากด KEY ใดเข้ามา
```

```
                    ตรวจหมายเลขถัดไป
2050 23         INX  H          ; MEMORY ADDRESS ที่ 4E01
2051 7E         MOV  A,M        ; ข้อมูลจาก MEMORY เก็บที่ A
2052 FE04       CPI  04         ; ตรวจว่าเป็นเลข 4 หรือไม่
2054 CA0021     JZ   2100       ; ถ้าใช่ไปทำ PROGRAM ที่ ADDRESS 2100
                                ; (ตรวจเลขถัดไป)
2057 CD6820     CALL 2068       ; ถ้าไม่ใช่แสดงว่าไม่มี IC เบอร์นี้อยู่
205A CD2326     CALL 2623       ; SET MEMORY DISPLAY เป็นFF ทุก LOCATION และ
                                ; CLEAR DISPLAY
                                ; ตรวจว่ามีใครกด KEY อะไรเข้ามาทางKEYBOARD
205D CD0028     CALL 2800       ;
2060 FE16       CPI  16         ; ตรวจดูว่ากดปุ่ม CLEAR หรือไม่
2062 CA6D20     JZ   206D       ; ถ้ากดไปทำ STEP ที่ ADDRESS 206D
2065 C35D20     JMP  205D       ; ถ้าไม่ใช่ WAITING LOOP

                    โปรแกรม ย่อย P2
              แสดงผล DISPLAYว่า NO HAVE IC
2068 3E80       MVI A,80        ; UNKNOWN IC

206A D300       OUT 00          ; SET LED UNKNOWN IC

206B 00         NOP             ;
206C C9         RET             ;
206D 3E00       MVI  A,00       ; CLEAR DISPLAY UNKNOWN IC
206F D3 00      OUT  00         ; LED UNKNOWN IC
2071 C34C26     JMP  264C       ; ไปรอรับเบอร์ IC ใหม่ที่กดเข้ามา
          โปรแกรมย่อย DECODE HEX 4 BIT UPPER เป็น 7 SEGMENT
2074 78         MOV  A,B        ; นำค่า INPUT LOGIC ไว้ที่ A
2075 E6F0       ANI  F0         ; MASK เอา 4 BIT บน (UPPER)
2077 0F         RRC             ;
2078 0F         RRC             ; ย้าย 4 BIT บนมาเป็น 4 BIT ล่าง
```

```
2079 0F          RRC          ;
207A 0F          RRC          ;
207B CD0222      CALL 2202    ; LOOK UP TABLE จากHEX เป็น7-SEGMENT
207E C9          RET          ;
```

โปรแกรมย่อย P3

SET PIN IC๑๔ขา;ขาINPUT/OUTPUT=1,2/3; 4,5/6; 9,10/8; 12,13/11

ประเภทที่ ๑ SOCKET#1

```
207F 00          NOP          ;
2080 3E64        MVI  A,64    ;A = 64   (8 6 3 : 0 1 1 0 0 1 0 0 ; 9 5 4 2 1)
2082 D3F0        OUT  F0      ;PORT A 8255 PIN CONTROLLER
2084 3E02        MVI  A,02    ;A = 02   (11 : 0 0 0 0 0 0 1 0 ; 13 12 10)
2086 D3F1        OUT  F1      ;PORT B 8255 PIN CONTROLLER
2088 C9          RET          ;
```

โปรแกรมย่อย P4

LED แสดง IC SOCKET อันที่ 1 (14ขา V_{cc} ขา14,GND ขา 7)

```
2089 3E01        MVI  A,01    ; A = 01, PC0 = SOCKET # 1
208B D362        OUT  62      ;{PORT C 8255 KEYBOARD และIC SOCKET
                               ;{DISPLAY
208D C9          RET          ;
```

โปรแกรมย่อย P5

LED แสดง IC SOCKET อันที่ 2 (14ขา,V_{cc} ขา 4,GND ขา 11)

```
208E 3E02        MVI  A,02    ; A = 02, PC1=SOCKET # 2
2090 D362        OUT  62      ; PORT C 8255 KEY&IC SOCKET DISPLAY
2092 C9          RET          ;
```

โปรแกรมย่อย P 6

LED แสดง IC SOCKETอันที่3 (14 ขาV_{cc} ขา5,GND ขา10)

```
2093 3E04       MVI  A,04    ; A = 04, PC2=SOCKET # 3
2095 D362       OUT  62      ; PORT C 8255 KEY&IC SOCKET DISPLAY
2097 C9         RET          ;
```

โปรแกรมย่อย P 7

LED แสดง IC SOCKET อันที่ 4 (16 ขา,V_{cc} ขา 16, GND ขา 8)

```
2098 3E08       MVI  A,08    ; A = 08, PC3=SOCKET # 4
209A D362       OUT  62      ; PORT C 8255 KEY&IC SOCKET DISPLAY
209C C9         RET          ;
```

โปรแกรมย่อย P 8

LEDแสดง IC SOCKET อันที่ 5 (16 ขา,V_{cc} ขา5, GND ขา 13)

```
209D 3E10       MVI  A,10    ; A = 10, PC4=SOCKET # 5
209F D362       OUT  62      ; PORT C 8255 KEY & IC SOCKET DISPLAY
20A1 C9         RET          ;
```

โปรแกรมย่อย P 9

LED แสดง IC SOCKET อันที่ 6 (16 ขา,V_{cc} ขา 5, GND ขา 12)

```
20A2 3E20       MVI  A,20    ; A = 20, PC5 = SOCKET # 6
20A4 D362       OUT  62      ; PORT C 8255 KEY & IC SOCKET DISPLAY
20A6 C9         RET          ;
```

โปรแกรมย่อย P 10

LED แสดง IC SOCKET อันที่ 7 (24 ขา, V_{cc} ขา 24, GND ขา 12)

```
20A7 3E40       MVI  A,40    ; A = 40, PC6 = SOCKET # 7
20A9 D362       OUT  62      ; PORT C 8255 KEY & IC SOCKET DISPLAY
20AB C9         RET          ;
20AC 00         NOP          ;
20AE 00         NOP          ;
20B0 00         NOP          ;
```

โปรแกรมย่อย P 11

LED แสดง"READY" DISPLAY

```
20B1 3E04       MVI  A,04    ; A = 04 , D2
20B3 D300       OUT  00      ; LED แสดง "READY" ติด
20B5 C9         RET          ;
```

โปรแกรมย่อย P 12

FAST/SLOW TEST

```
20B6 21FE4F     LXI  H,4FFE  ; STATUS ของ FAST/SLOWอยู่ที่ 4FFE
20B9 CD0028     CALL 2800    ; รอรับผลจาก KEYBOARD
20BC FE13       CPI  13      ; ตรวจว่าเป็นปุ่ม SLOW หรือไม่
20BE CAC920     JZ   20C9    ; ถ้าใช่รอการกด KEY ปุ่ม TEST
20C1 FE17       CPI  17      ; ตรวจว่าเป็นปุ่ม TEST หรือไม่
20C3 CAD420     JZ   20D4    ; ถ้าใช่ SET STATUS 4FFE เป็น 00
20C6 C39821     JMP  2198    ; ถ้าไม่ใช่ไปตรวจว่าเป็นปุ่ม CLEAR ต่อไป
20C9 CD0028     CALL 2800    ; รอรับผลจาก KEYBOARD
20CC FE17       CPI  17      ; ตรวจว่าเป็นปุ่ม TEST
20CE C22321     JNZ  2123    ; ถ้าไม่ใช่ส่งเสียง ALARM (กด KEY ผิด)
```

```
20D1 36FF       MVI  M,FF     ; ปิดท้ายด้วย FF หลังเบอร์ IC ที่กดเข้ามา
20D3 C9         RET           ;
```

โปรแกรมย่อย P 13

SET FLAG FAST TEST = ๐๐

```
20D4 3600       MVI  M,00     ;
20D6 C9         RET           ;
```

โปรแกรมย่อย P 14

ลักษณะขา IC—ประเภท 1/เลือก SOCKET/แจ้ง READY /ตรวจ FAST/หรือ SLOW TEST

```
20D7 CD8020     CALL 2080     ; PIN CONTROL ประเภทที่ 1
20DA CD8920     CALL 2089     ; เลือก SOCKET # 1 LED ติด
20DD CDB120     CALL 20B1     ; แจ้งพร้อมแล้ว (READY)
20E0 CDB620     CALL 20B6     ; ตรวจดูว่ากดปุ่ม SLOW หรือไม่กดแล้วแต่ผู้ใช้
20E3 C9         RET           ;
```

โปรแกรมย่อย P 15

ตรวจ FLAG FAST/SLOW TEST (F/S FLAG)ถ้า SLOW ก็จะหน่วงเวลา(DELAY)ให้เลย

```
20E4 E5         PUSH H        ;
20E5 D5         PUSH D        ;
20E6 C5         PUSH B        ; SAVE REGISTER ทั้งหมด
20E7 F5         PUSH PSW      ;
20E8 21FE4F     LXI  H,4FFE   ; F/S FLAG อยู่ที่ 4FFE
20EB 7E         MOV  A,M      ; เอาค่าที่ FLAG เข้าไปไว้ที่ A
20EC FEFF       CPI  FF       ; ตรวจว่าเป็น SLOW หรือไม่
20EE C2F420     JNZ  20F4     ; ถ้าไม่เป็นทดสอบขั้นตอนต่อไป
20F1 CD4D27     CALL 274D     ; ถ้าเป็นทดสอบแบบช้า(SLOW)ก็หน่วงเวลา(DELAY) 1 SEC
```

```
20F4 F1          POP  PSW   ;
20F5 C1          POP  B     ;  คืนค่าเก่าให้กับ REGISTER นั้นอย่างเดิม
20F6 D1          POP  D     ;
20F7 E1          POP  H     ;
20F8 C9          RET         ;
```

โปรแกรมย่อย P16

4 BIT LOWER DISPLAY (D_3-D_0) จากฐาน 2 เป็นฐาน 16 ในรูปของ 7 SEGMENT

B - เป็น INPUT

```
20F9 78          MOV  A,B    ; B ไปเก็บไว้ที่ A
20FA E60F        ANI  0F     ; MASK 4 BIT หลัง D3-D0 ไว้
20FC CD0222      CALL 2202   ; DECODE ว่าเลข BINARY ตรงกับ 7 SEGMENT เลขฐาน 16 อะไรผลลัพท์อยู่ที่ A
20FF C9          RET         ;
```

ต่อโปรแกรม DECODE เบอร์ IC

ในช่อง DIGIT ที่ 3

```
2100 23          INX  H      ; เลื่อนข้อมูลมาที่ตัวเลข IC ตัวที่ 3
2101 7E          MOV  A,M    ; ข้อมูลใน MEMORY มาเก็บเข้าที่ A
2102 FE11        CPI  11     ; ตรวจว่าเป็นเลขหรืออักษร L=(11)HEX
2104 CA1921      JZ   2119   ; ถ้าใช่ L ไปตรวจว่าเลขหรืออักษรต่อไปเป็น S หรือเป็นตัวเลข
2107 FE12        CPI  12     ; ถ้าไม่ใช่ L ตรวจว่าใช่ S หรือไม่
2109 CAD321      JZ   21D3   ; ถ้าใช่ตรวจว่าเลขต่อไปเป็นเลขอะไรของตารางพวก IC ประเภท SHOCKY
210C FE10        CPI  10     ; ถ้าไม่ใช่ S ตรวจต่อไปว่าเป็น H หรือไม่
210E CADF21      JZ   21DF   ; ถ้าใช่ก็ไปที่ตารางของพวก IC ประเภท HIGH SPEED
2111 FE0C        CPI  0C     ; ถ้าไม่ใช่ตรวจว่าเป็น C หรือไม่
2113 CAEB21      JZ   21EB   ; ถ้าใช่ก็ไปที่ตารางของพวก IC ประเภทตระกูล CMOS
2116 C3F721      JMP  21F7   ; ไปทำ PROGRAM ต่อเฉพาะ IC ที่มีเบอร์เป็นตัวเลขล้วน ๆ
2119 23          INX  H      ; นำตัวเลขหลักที่ 4 มาตรวจ
211A 7E          MOV  A,M    ; นำค่าเลขตัวที่ 4 จาก MEMORY ไว้ที่ A
```

```
211B FE12       CPI  12      ; ตรวจว่าเป็นตัว S หรือไม่
211D CA2B21     JZ   212B    ; ถ้าใช่ไปทำตามตารางของ IC ประเภท LS
2120 C3C721     JMP  21C7    ; ไปทำต่อที่โปรแกรมตารางของ IC ประเภท L
2123 D310       OUT  10      ; ส่งเสียง ALARM เสียง TONE เดียว
2125 C3C920     JMP  20C9    ; กลับไปตรวจว่ากดปุ่ม TEST มาหรือไม่

                โปรแกรมตรวจต่อ IC ประเภท LSxxx

2128 00         NOP          ;
2129 00         NOP          ;
212A 00         NOP          ;
212B 23         INX  H       ; เลื่อนเลขบัดไปหลังอักษร LS (=x1)
212C 7E         MOV  A,M     ; ย้ายข้อมูลจาก MEMORY เข้า A (A=X1)
212D 4F         MOV  C,A     ; SAVE A ไว้ที่ C
212E E5         PUSH H       ; SAVE HL ไว้ที่ M
                             ; LOAD POINTER ชี้ตาราง PROGRAM ของ IC
212F 11C00D     LXI  D,0DC0  ; ที่จะวิ่งไปทำการทดสอบ
                             ; LOAD POINTER ชี้ตารางเบอร์ ICพวก
2132 21600D     LXI  H,0D60  ; 54/74LSXXX
2135 7E         MOV  A,M     ; เอาหมายเลขเบอร์ IC ในตารางมาเพื่อตรวจสอบ
2136 FEFF       CPI  FF      ; ตรวจสอบว่าเท่ากับค่าFF หรือไม่(หมดข้อมูลแล้ว)
2138 CA4C26     JZ   264C    ; ถ้าเท่ากับ00แจ้ง NO IC ไม่มี IC เบอร์นี้อยู่
213B 47         MOV  B,A     ; ถ้าไม่เท่ากับ FF SAVE Aไว้ที่ B
213C E6F0       ANI  F0      ; MASK 4 BIT หน้า (D7-D4)
213E 0F         RRC          ;
213F 0F         RRC          ;
2140 0F         RRC          ; เลื่อนข้อมูลจาก D7-D4 เป็น D3-D0
2141 0F         RRC          ;
2142 B9         CMP  C       ; เปรียบเทียบเบอร์ IC(X1X2X3)ตัวแรก X1
2143 CA4D21     JZ   214D    ; เท่ากันตรวจเลขถัดไปอีก
```

```
2146 23          INX  H      ;
                                 } ถ้าไม่เท่าเลื่อนไปตรวจข้อมูลเบอร์ IC ถัดไป
2147 23          INX  H      ;
2148 13          INX  D      ;
                                 } พร้อมเลื่อนข้อมูล PROGRAM ด้วย
2149 13          INX  D      ;
214A C33521      JMP  2135   ; กลับไปตรวจใหม่ลักษณะเดิม
214D 78          MOV  A,B    ; เอาข้อมูลเบอร์IC ที่ SAVE ไว้มาเก็บที่ A
214E 44          MOV  B,H    ;
                                 } SAVE POINTER ข้อมูลเบอร์ IC
214F 4D          MOV  C,L    ;
2150 E1          POP  H      ; เรียก POINTER ที่ชี้ข้อมูลเบอร์IC ที่กดเข้ามา
                             ; ของเก่าคืนมา
2151 C5          PUSH B      ; SAVE ข้อมูลใน BC REGISTER
2152 23          INX  H      ; เลื่อน POINTER ข้อมูลเลขถัดไป (=X2)
2153 4E          MOV  C,M    ; เอาเลขเบอร์(=X2)ไปเก็บที่ C เพื่อไปเปรียบเทียบ
2154 E60F        ANI  0F     ; MASK ข้อมูลเอาเฉพาะ4BIT หลัง(D3-D0)
2156 B9          CMP  C      ; เปรียบเทียบว่าข้อมูลเบอร์IC ตรงกับข้อมูลเบอร์-
                             ; IC ที่กดเข้ามาหรือไม่
2157 CA7021      JZ   2170   ; ถ้าเท่ากันไปตรวจเลขถัดไปคือ X3
215A 2B          DCX  H      ; ไม่เท่าถอยไปเลขเก่าก่อน X1 เพื่อตรวจใหม่
215B 7E          MOV  A,M    ; นำข้อมูลเบอร์IC X1 เก็บที่ A
215C C1          POP  B      ; คืนค่า POINTER ข้อมูลเบอร์ IC กลับเข้า BC
215D E5          PUSH H      ; SAVE POINTER เบอร์IC ที่กดเข้ามา
215E 60          MOV  H,B    ;
                                 } นำ POINTERข้อมูลเบอร์IC คืนที่เดิม
215F 69          MOV  L,C    ;
2160 4F          MOV  C,A    ; นำข้อมูลเบอร์IC X1 เก็บจาก A ไว้ที่ C
2161 C34621      JMP  2146   ; ไปตรวจข้อมูลเบอร์ ICถัดไป
2164 E5          PUSH H      ;
2165 210023      LXI  H,2300 ;
2168 4F          MOV  C,A    ;
2169 0600        MVI  B,00   ;
```

```
216B 09          DAD  B       ;
216C 7E          MOV  A,M     ;
216D E1          POP  H       ;
216E C9          RET          ;
216F 00          NOP          ;
2170 23          INX  H       ; เลื่อน POINTER ข้อมูลถัดไป (=X3)
2171 7E          MOV  A,M     ; นำ X3 ใน M เก็บที่ A
2172 FEFF        CPI  FF      ; เปรียบเทียบว่าเท่ากับFF (หมดข้อมูลเบอร์IC ทั้งหมดที่มีอยู่แล้วตรวจหมดแล้วไม่พบ)
2174 CA4C26      JZ   264C    ; ถ้าจบแล้ว(เท่ากับ FF)ไปแจ้งว่า NO IC
2177 C1          POP  B       ; ถ้าไม่เท่าคืนPOINTER ชี้ข้อมูลเบอร์ IC ที่ BC
2178 E5          PUSH H       ; SAVE POINTER ชี้เบอร์ IC ที่กดเข้ามา
2179 60          MOV  H,B     ;
217A 69          MOV  L,C     ; คืน POINTERชี้ข้อมูลเบอร์ ICจาก BCเข้า HL
217B 23          INX  H       ; เลื่อนค่า POINTER ชี้ข้อมูลเบอร์ IC ถัดไป
217C 7E          MOV  A,M     ; ย้ายข้อมูลเบอร์ IC จาก M มา A
217D 44          MOV  B,H     ;
217E 4D          MOV  C,L     ; SAVE POINTER ข้อมูลเบอร์ IC ที่ BC
217F E1          POP  H       ; คืนPOINTER ชี้ขอมูลที่กดเข้ามา X3
2180 C5          PUSH B       ; SAVE PONITER ข้อมูลเบอร์ IC
2181 4E          MOV  C,M     ; นำค่าM=X3 เก็บในC เพื่อเปรียบเทียบต่อไป
2182 47          MOV  B,A     ; SAVE A ไว้ที่ B
2183 E6F0        ANI  F0      ; MASK 4BIT หน้า (D7-D4)
2185 0F          RRC          ;
2186 0F          RRC          ; เคลื่อนย้ายจาก 4 BIT หน้า(D7-D4) เป็น
2187 0F          RRC          ; 4 BIT หลัง (D3-D0)
2188 0F          RRC          ;
2189 B9          CMP  C       ; เปรียบเทียบว่าข้อมูลเบอร์IC ในตารางกับข้อมูลเบอร์ ICที่กดเข้ามา
218A CAA021      JZ   21A0    ; ถ้าเท่ากันไปตรวจต่อไป
```

```
218D 2B          DCX  H        ;
218E 2B          DCX  H        ;  ไม่เท่าเลื่อนค่า POINTER ชี้เบอร์ IC ที่กดเข้ามาถอยหลังไป 2 ที่
218F 7E          MOV  A,M      ; นำค่าเบอร์IC ที่กดเข้ามาไว้ที่ A(A=X2)
2190 C1          POP  B        ; คืนค่าBC ที่ SAVE ไว้กลับที่เดิมซึ่งเก็บค่า POINTER ข้อมูลเบอร์ IC
2191 E5          PUSH H        ; SAVE POINTER เบอร์ IC ที่กดเข้ามา
2192 60          MOV  H,B      ;
2193 69          MOV  L,C      ;  คืน POINTER ชี้ข้อมูลเบอร์ IC จาก BC เข้ามาเก็บที่ HL ตามลำดับ
2194 4F          MOV  C,A      ; SAVE ค่าเบอร์ IC ที่กดเข้ามาจาก A ไปเก็บที่C
2195 C34721      JMP  2147     ; กลับไปตรวจข้อมูลเบอร์IC ถัดไปใหม่อีก
2198 FE16        CPI  16       ; ตรวจว่าเป็นปุ่ม CLEAR หรือไม่
219A CA4524      JZ   2445     ; ถ้าใช่กลับไปเคลีย LED ทั้งหมดแล้วเริ่มต้นใหม่ใหม่
219D C34B20      JMP  204B     ; ส่งเสียงALARM TONE ยาวและไปตรวจ KEYBOARD
21A0 23          INX  H        ; เลื่อน POINTER เบอร์IC ที่กดเข้ามาเพิ่มอีก
21A1 7E          MOV  A,M      ; เคลื่อนย้ายค่าเบอร์IC ที่กดเข้ามาไว้ที่ A
21A2 E60F        ANI  0F       ; MASK 4 BITหลัง (D3-D0)
21A4 C1          POP  B        ; คืน POINTER ชี้เบอร์IC ในTABLE ไว้ที่BC
21A5 E5          PUSH H        ; SAVE HL REGISTER
21A6 60          MOV  H,B      ;
21A7 69          MOV  L,C      ;  เปลี่ยนPOINTER เป็นเบอร์ IC จากตาราง
21A8 4F          MOV  C,A      ; SAVE A (เบอร์IC ที่กดเข้ามา) ไว้ที่ C
21A9 7E          MOV  A,M      ; นำข้อมูลเบอร์ IC จากตารางเข้าไว้ที่ A
21AA E60F        ANI  0F       ; MASK 4 BIT หลัง (D3-D0)
21AC B9          CMP  C        ; เปรียบเทียบข้อมูลทั้งสอง
21AD CAC021      JZ   21C0     ; ถ้าเท่ากันไปRUN PROGRAM ทดสอบICเบอร์นั้นได้
21B0 44          MOV  B,H      ;
21B1 4D          MOV  C,L      ;  ถ้าไม่เท่ากันSAVE POINTER ไว้ที่ BC
21B2 E1          POP  H        ; คืนค่า POINTER เบอร์ IC ที่กดเข้ามา
```

```
21B3 C5          PUSH B      ; SAVE POINTER เบอร์ IC จากตาราง
21B4 2B          DCX H       ; ลดค่า POINTERเบอร์ IC ที่กดเข้ามา
21B5 C38D21      JMP 218D    ; กลับไปตรวจใหม่
21B8 2B          DCX H       ;
21B9 C38D21      JMP 218D    ;
21BC CA4C26      JZ 264C     ;
21BF C9          RET         ;
21C0 EB          XCHG        ; สลับที่ HL กับ DE
21C1 56          MOV D,M     ; นำค่าADDRESSชี้ PROGRAM(A15-A9)ไว้ที่D
21C2 23          INX H       ; เพิ่มค่าPOINTER ชี้ ADDRESS PROGRAM อีก 1
21C3 5E          MOV E,M     ; นำค่า ADDRESSชี้ PROGRAM (A8-A0)
21C4 EB          XCHG        ; สลับที่ DE กับ HL
21C5 E9          PCHL        ; ไปทำตามคำสั่งที่ HL ADDRESS ทดสอบเบอร์ IC นั่น ๆ
21C6 00          NOP         ; NO OPERATE
21C7 7E          MOV A,M     ; นำข้อมูลใน MEMORY เก็บที่ A
21C8 4F          MOV C,A     ; นำข้อมูลใน A เก็บไว้ที่ C REGISTER
21C9 E5          PUSH H      ; SAVE HL
21CA 113000D     LXI D,0D30  ; POINTER ชี้ TABLE โปรแกรมที่จะทำตามคำสั่งสำหรับเบอร์ IC 54/74Lxxx
21CD 210000D     LXI H,0D00  ; POINTER ชี้ TABLEเบอร์IC ประเภท54/74Lxxx
21D0 C33521      JMP 2135    ; กลับไปตรวจเบอร์ IC
21D3 23          INX H       ; เพิ่ม HL = HL + 1
21D4 4E          MOV C,M     ;
21D5 E5          PUSH H      ; SAVE HL
21D6 11B00C      LXI D,0CB0  ; POINTERชี้ TABLE โปรแกรมที่จะทำตามคำสั่งสำหรับเบอร์ IC 54/74Sxxx
21D9 21300C      LXI H,0C30  ; POINTERชี้ TABLEเบอร์ IC ประเภท54/74Sxxx
21DC C33521      JMP 2135    ; กลับไปตรวจเบอร์ IC
```

```
21DF 23          INX  H        ; เพิ่ม HL = HL + 1
21E0 4E          MOV  C,M      ;
21E1 E5          PUSH H        ; SAVE HL
21E2 1170ØC      LXI  D,ØC7Ø   ; POINTERชี้ TABLE โปรแกรมที่จะทำตามคำสั่ง
                               ; สำหรับเบอร์ IC 54/74Hxxx
21E5 2100ØC      LXI  H,ØCØØ   ; POINTERชี้ TABLE เบอร์ ICประเภท 54/74Cxxx
21E8 C33521      JMP  2135     ; กลับไปตรวจเบอร์ IC
21EB 23          INX  H        ; เพิ่ม HL = HL + 1
21EC 4E          MOV  C,M      ;
21ED E5          PUSH H        ; SAVE HL
21EE 1170ØE      LXI  D,ØE7Ø   ; POINTER ชี้ TABLE โปรแกรมที่จะทำตามคำสั่ง
                               ; สำหรับเบอร์ IC 54/74Cxxx
21F1 2120ØE      LXI  H,ØE2Ø   ; POINTER ชี้ TABLE เบอร์IC ประเภท54/74Cxxx
21F4 C33521      JMP  2135     ; กลับไปตรวจเบอร์ IC
21F7 4E          MOV  C,M      ;
21F8 E5          PUSH H        ; SAVE HL
21F9 1170ØB      LXI  D,ØB7Ø   ; POINTER ชี้ TABLE โปรแกรมที่จะมีตามคำสั่ง
                               ; สำหรับเบอร์ IC 54/74xxx
21FC 2100ØB      LXI  H,ØBØØ   ; POINTER ชี้ TABLE เบอร์IC ประเภท54/74xxx
21EF C33521      JMP  2135     ; กลับไปตรวจเบอร์ IC
```

โปรแกรายย่อย P 17

สำหรับ LOOK UP TABLE จาก HEX เป็น 7-SEGMENT DISPLAY

```
2202 E5          PUSH H        ; SAVE HL
2203 C5          PUSH B        ; SAVE BC
2204 2100ØF      LXI  H,ØFØØ   ; POINTER ชี้ TABLE 7-SEGMENT DISPLAY ที่
                               ; ADDRESS ØFØØ
2207 4F          MOV  C,A      ; INPUT A เลข HEX เก็บไว้ที่ C
2208 0600        MVI  B,ØØ     ; CLEAR B=ØØ
220A 09          DAD  B        ; HL=HL+BC
220B 7E          MOV  A,M      ; เอาข้อมูลที่ได้จาก MEMORY เก็บไว้ที่ A
                               ; ( A เป็น OUTPUT )
220C C1          POP  B        ; คืน BC ที่ SAVE ไว้
```

```
220D E1          POP  H          ; คืน HL ที่ SAVE ไว้
220E C9          RET              ;

                 โปรแกรมย่อย  P 18

แสดงผลในรูปของเลขฐาน ๑๖และ BINARY ทาง LED 2 DIGIT สำหรับ INPUT LOGIC =

                 -CONDITION ที่ DRIVE เข้า IC.

220F CDF920      CALL 20F9       ; เรียกโปรแกรมย่อย MASK เฉพาะ 4 BIT LOWER
                                 ; และแปลง CODE เป็น 7-SEGMENT
2212 D37B        OUT  7B         ; DISPLAY ออก LED 7-SEGMENT #7B (DIGIT16)0
2214 CD7420      CALL 2074       ; เรียกโปรแกรมย่อย MADK เฉพาะ 4 BIT UPPER
                                 ; และแปลง CODE เป็น 7-SEGMENT
2217 D37A        OUT  7A         ; DISPLAY ออก LED 7-SEGMENT #7A (DIGIT16)1
2219 78          MOV  A,B        ; B เก็บค่า INPUT LOGIC ที่เป็นเลข BINARY
                                 ; ส่งไปเก็บที่ A
221A D3C0        OUT  C0         ; DISPLAY ออก 8255 ที่เป็นเลข BINARY PORT A,
                                 ; DEVICE #C0 (INPUT LOGIC DISPLAY)
221C C9          RET             ;

                 โปรแกรมย่อย  P 19

แสดงผล BINARY OUTPUT LOGIC ของภายใน IC ประเภทที่ ๑(พวกที่ลักษณะภายในแบบ 7400)

221D 00          NOP             ;
221E 00          NOP             ;
221F 00          NOP             ;
2220 F5          PUSH PSW        ; SAVE A และ STATUS FLAG
2221 0F          RRC             ; ROTATE ไปทางขวา ๑ ครั้ง } เพื่อเลื่อนOUTPUT
2222 0F          RRC             ; ROTATE ไปทางขวา ๑ ครั้ง } ให้มาตรงBITที่0
2223 CD6922      CALL 2269       ; BINARY OUTPUT ออก PORT A , DEVICE #B0
                                 ; พร้อมออก 7-SEGMENT เป็นเลขฐาน ๑๖ ด้วย
2226 F1          POP  PSW        ; คืน A และ STATUS FLAG ดังเดิม
2227 C9          RET             ;

                 โปรแกรมย่อย  P 20

แสดงผลเป็น HEX. OUTPUT LOGIC,1 DIGIT

2228 C5          PUSH B          ; SAVE B C
2229 47          MOV  B,A        ; เอา A ไปเก็บในB ( A เป็น BINARY )
```

```
222A CDF920     CALL 20F9     ;เรียกโปรแกรมย่อย MASK 4 BIT LOWER และ CODE เป็น 7-SEGMENT
222D D37F       OUT  7F       ; ส่ง OUTPUT ไปที่ LED 7-SEGMENT#7F
222F C1         POP  B        ; คืน BC อย่างเดิม
2230 C9         RET           ;
```

โปรแกรมย่อย P 21

SET GOOD FLAG และ DELAY นาน ๑ วินาที สำหรับทดสอบแบบ FAST/SLOW

```
2231 EB         XCHG          ; สลับ HL กับ DE
2232 36FF       MVI  M,FF     ;เอาค่า FF ไปเก็บที่ M DE (ถ้า IC ส่วนนั้นดี ก็จะ SET เป็น FF = GOOD)
2234 CD4A23     CALL 234A     ;เรียกโปรแกรมย่อย แสดงทาง LED ว่า GOOD และ DELAY นาน ๑ วินาที
2237 EB         XCHG          ; สลับ HL กับ DE
2238 C9         RET           ;
```

โปรแกรมย่อย P 22

SET BAD FLAGและDELAY นาน ๑ วินาที สำหรับทดสอบแบบ FAST/SLOW

```
2239 EB         XCHG          ; สลับ HL กับ DE
223A 3600       MVI  M,00     ;เอาค่า 00 ไปเก็บที่ M DE (ถ้า IC ส่วนนั้นเสีย ก็จะ SET เป็น 00=BAD)
223C CD3E23     CALL 233E     ;เรียกโปรแกรมย่อย แสดงทาง LED ว่า BAD และ DELAY นาน ๑ วินาที
223F EB         XCHG          ; สลับ HL กับ DE
2240 C9         RET           ;
```

โปรแกรมย่อย P 23

แสดงผล INPUT LOGIC ในลักษณะเลขฐาน 16 และ DRIVE LOGIC เข้าขา INPUT ของIC และรับผลกลับ

```
2241 CD0F22     CALL 220F     ; เรียกโปรแกรมแสดงผลเลขฐาน16 INPUT LOGIC
2244 D3E0       OUT  E0       ;ส่ง LOGIC เข้าINPUT ของขา IC ของ GATE ส่วนที่กำลังตรวจสอบ PORTA # E0
2246 DBD0       IN   D0       ;รับผลกลับเข้าระบบ เพื่อนำไปประมวลผลที่ได้ 8255 PORTA # D0
2248 C9         RET           ;
```

โปรแกรมย่อย P 24

ตรวจสอบว่าที่ได้ตรงกับ DATA หรือไม่ SET FLAG GØØD -BAD และ DELAY นาน ๑ วินาที

```
2249 4E          MOV  C,M    ; เอา DATA ผลลัพท์ที่ถูกต้องจาก TABLE
                             ; เก็บไว้ที่ C
224A 91          SUB  C      ; เอา A=A-C(A=DATAจากผลที่ได้,C=DATAจริงที่ถูก)
224B C25222      JNZ  2252   ; ถ้าไม่เท่ากันก็เสีย(BAD) ไป SET BAD FLAG
                             ; และ DELAY ๑ วินาที
224E CD3122      CALL 2231   ; เรียกโปรแกรม SET GOOD FLAG และ
                             ; DELAY นาน ๑ วินาที
2251 C9          RET         ;
2252 CD3922      CALL 2239   ; เรียกโปรแกรม SET BAD FLAG และ DELAY
                             ; นาน ๑ วินาที
2255 C9          RET         ;
```

โปรแกรมย่อย P 25

แสดงผล BINARY LOGIC OUTPUT ตรวจสอบผลที่ได้ว่าตรงกันหรือไม่ตรวจสอบว่าทำครบ 4 CONDITION

```
2256 CD2Ø22      CALL 2220   ; แสดงผล BINARY LOGIC OUTPUT
2259 CD4922      CALL 2249   ; ตรวจสอบว่าผลที่ได้ตรงกับ DATA หรือไม่,
                             ; SETFLAG GOOD/BAD และ DELAY
225C 23          INX  H      ; เพิ่มHL=HL+1,HL เก็บ ADDRESS POINTER
                             ; ชี้ข้อมูลพวก TRUTH TABLE (OUTPUT LOGIC)
225D 13          INX  D      ; เพิ่ม DE=DE+1,DE POINTER ชี้ MEMORY ที่เก็บ
                             ; ผลการตรวจสอบว่า GOODหรือ BADแต่ละCONDITION
225E 78          MOV  A,B    ; B=INPUT LOGIC CONDITION ที่ DRIVE
                             ; เข้าขา INPUT ของ IC เก็บที่ A
225F FE03        CPI  03     ; เพื่อตรวจสอบว่าทำครบ 4 CONDITION หรือยัง
2261 C9          RET         ;
```

โปรแกรมย่อย P 26

ถอย POINTER (HL)ที่ชี้ผลลัพท์ที่ถูก ไปเริ่มต้นใหม่ ๔ ตัว

และ SET INPUT LOGIC B=ØØ เริ่มใหม่

```
2262 2B          DCX  H      ; HL = HL - 1
2263 2B          DCX  H      ; HL = HL - 1
2264 2B          DCX  H      ; HL = HL - 1
2265 2B          DCX  H      ; HL = HL - 1
```

```
2266 0600       MVI  B,00    ; SET B = 00
2268 C9         RET          ;
```

โปรแกรมย่อย P 27

แสดงผล BINARY LOGIC OUTPUT ที่ LED และ HEX. LOGIC OUTPUT ที่ 7-SEIMENT

```
2269 D3B0       OUT  B0      ; แสดงผลออก LED 8255,PORT A, DIVICE # B0
                             ; (BINARY LOGIC OUTPUT)
226B CD2822     CALL 2228    ; แสดงผลออกทาง LED 7-SEGMENT
                             ; เป็นเลขฐาน 16 (HEX)
226E C9         RET          ;
```

โปรแกรมย่อย P 28

ทดสอบ IC ประเภทที่ ๑ ส่วนที่ ๑ (PART 1) SOCKET#1

```
226F 00         NOP          ;
2270 00         NOP          ;
2271 3E06       MVI  A,06    ; SET ค่าA=(06)HEX เพื่อให้แสดงออก
                             ; 7-SEGMENT เป็นเลข ๑
2273 D377       OUT  77      ; ส่งข้อมูลจาก A-(06)HEX ออกไป LED
                             ; 7 SEGMENT ติดเป็นเลข ๑
2275 0600       MVI  B,00    ; SET INPUT LOGIC B=00 CONDITION 00
2277 CD0F22     CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เฉพาะเลขฐาน
                             ; ๑๖
227A 78         MOV  A,B     ; เอาค่าใน B เก็บไว้ที่ A
227B CD4422     CALL 2244    ; DRIVE INPUT LOGIC เข้าขา IC และรับผลกลับ
                             ; มาประมวล
227E E604       ANI  04      ; MASK BIT ที่ 3(D3) OUTPUT ของ PART ที่ 1
2280 CD5622     CALL 2256    ; ตรวจสอบดูว่าเท่าหรือไม่ ทดสอบจนครบ
                             ; 4 CONDITION
2283 C8         RZ           ; ครบ 4 CONDITION แล้ว RETURN
2284 00         NOP          ;
2285 00         NOP          ;
2286 04         INR  B       ; ไม่ครบเพิ่มค่า INPUT LOGIC อีก ๑
2287 C37722     JMP  2277    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 29

ทดสอบ IC ประเภทที่ ๑ ส่วนที่ ๒ (PART 2) SOCKET#1

```
228A CD4A24     CALL 244A    ; SET ค่าEA ปิดท้ายผลการทดสอบ PART ที่ ๑
228D 3E5B       MVI  A,5B    ; A=(5B)HEX="2" 7-SEGMENT
228F D377       OUT  77      ; แสดงผลว่ากำลังทดสอบส่วนที่ ๒ (PART 2)
                             ; 7-SEGMENT ติดเลข "2"
2291 CD6222     CALL 2262    ; ถอยหลังกลับไปดู TRUTH TABLE เริ่มต้นใหม่ และ
                             ; SET CONDITION แรกใหม่ =ØØ
2294 CDØF22     CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เฉพาะเลขฐาน
2297 78         MOV  A,B     ; เคลื่อน INPUT LOGIC ที่จะขับเข้าขา IC
                             ; จาก B ไว้ที่ A
2298 Ø7         RLC          ; เคลื่อน INPUT LOGIC ให้ตรงกับขา IC
2299 Ø7         RLC          ;
229A Ø7         RLC          ;
229B CD4422     CALL 2244    ; DRIVE INPUT LOGIC เข้าขา IC และรับผล
                             ; กลับมาประมวล
                             ; MASK BITที่ 5(D5) OUTPUT ของPART ที่ ๒
229E E62Ø       ANI  2Ø      ;
22AØ ØF         RRC          ; เลื่อนผลที่ได้ให้ตรงกลับข้อมูลใน TABLE
22A1 ØF         RRC          ;
22A2 ØF         RRC          ;
22A3 CD5622     CALL 2256    ; ตรวจสอบว่าผลที่ได้ต่างกับที่เป็นจริงหรือไม่,ตรวจว่า
                             ; ครบ 4 CONDITION
22A6 C8         RZ           ; ถ้าครบการTESTทั้ง 4 CONDITIONแล้วRETURN
22A7 ØØ         NOP          ;
22A8 ØØ         NOP          ;
22A9 Ø4         INR  B       ; ถ้าไม่ครบ 4 conditionให้เพิ่มB=B+1 คือ
                             ; INPUT LOGIC STAGE ต่อไป
22AA C39422     JMP  2294    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 3Ø

ทดสอบ IC ประเภทที่ ๑ ส่วนที่ ๓ (PART 3) SOCKET#1

```
22AD CD4A24     CALL 244A    ; SETค่า'EA' ปิดท้ายผลการทดสอบPART ที่ ๒
22BØ 3E4F       MVI  A,4F    ; A=(4F)HEX ลักษณะตรงกับเลข " 3"ของ
                             ; 7-SEGMENT
```

```
22B2 D377        OUT  77      ;แสดงผลทาง LED 7-SEGMENT เป็นเลข 3 ~ DEVICE#77
22B4 CD6222      CALL 2262    ;ถอยหลังกลับไปดูTRUTH TABLE เริ่มต้นใหม่ และ SET CONDITION
22B7 CD0F22      CALL 220F    ;แสดงผล INPUT LOGIC ทาง LED เฉพาะเลขฐาน ๑๖
22BA 78          MOV  A,B     ;เคลื่อน INPUT LOGIC ที่จะขับเข้าขา IC จาก B ไว้ที่ A
22BB 0F          RRC          ;
22BC 0F          RRC          ;เลื่อน INPUT LOGIC ให้ตรงขา IC
22BD E680        ANI  80      ; MASK เอา BIT 7(D7) ออกไปใช้งาน
22BF D3E0        OUT  E0      ; ส่ง LOGIC เข้าขา IC เพียง ๑ ขา (ขา ๙)
22C1 78          MOV  A,B     ;เคลื่อน INPUT LOGIC ที่จะรับเข้าขา IC จาก B ไว้ที่ A
22C2 E601        ANI  01      ; MASK เอา BIT 0(D0) ออกไปใช้งาน
22C4 D3E1        OUT  E1      ; ส่ง LOGIC เข้าขาIC อีกขาหนึ่ง (ขา ๑๐)
22C6 DBD0        IN   D0      ; รับ OUTPUT ที่ได้เข้าระบบ
22C8 E640        ANI  40      ; MASK เอาเฉพาะ BIT 6 (D6)
22CA 0F          RRC          ;
22CB 0F          RRC          ;เลื่อนผลที่ได้จาก OUTPUTให้ตรงกับข้อมูลใน TABLE
22CC 0F          RRC          ;
22CD 0F          RRC          ;
22CE CD5622      CALL 2256    ;ตรวจสอบว่าผลที่ได้เท่ากับที่เป็นจริงหรือไม่, ตรวจว่าครบ 4 CONDITION
22D1 C8          RZ           ;ถ้าครบการ TEST ทั้ง 4 CONDITIONล้ว RETURN
22D2 00          NOP          ;
22D3 00          NOP          ;
22D4 04          INR  B       ;ถ้าไม่ครบ4 CONDITIONให้เพิ่มB=B+1 คือ INPUT LOGIC STAGE ต่อไป
22D5 C3B722      JMP  22B7    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 31

ทดสอบ IC ประเภทที่ ๑ ส่วนที่ ๔ (PART 4) SOCKET#1

```
22D8 CD4A24      CALL 244A    ; SET ค่า'EA' บัตท้ายผลการทดสอบPART ที่ ๓
```

```
22DB CDFC22      CALL 22FC   ; CLEAR LOGIC INPUT ที่ค้างอยู่ที่ PORT ให้เป็น Ø
22DE 3E66        MVI  A,66   ; SET A=(66)HEX=เลข ๔ ของ 7-SEGMENT
22E0 D377        OUT  77     ; แสดงผลทาง LED 7-SEGMENT เป็นเลข ๔, DEVICE#77
22E2 CD6222      CALL 2262   ; ถอดหลังกลับไปดู TRUTH TABLE เริ่มต้นใหม่ และ SET CONDITION แรกใหม่
22E5 CD0F22      CALL 220F   ; แสดงผล INPUT LOGIC ทาง LED เฉพาะเลขฐาน 16
22E8 CD5B23      CALL 235B   ; เรียกโปรแกรมช่วยทดสอบ PORT ที่ 4 เรื่องของ INPUT LOGIC
22EB D3E1        OUT  E1     ; ส่ง LOGIC เข้าขา IC ทดสอบ (DEVICE#E1)
22ED DBD1        IN   D1     ; รับผลที่ได้กลับ ทาง DEVICE#D1
22EF E602        ANI  Ø2     ; MASK เอา BIT ที่ 1($D_1$)
22F1 Ø7          RLC         ; เลื่อน OUTPUT ที่ได้ให้ตรงกับข้อมูลใน TABLE
22F2 CD5622      CALL 2256   ; ตรวจสอบว่าผลที่ได้เท่ากับที่เป็นจริงหรือไม่, ตรวจว่าครบ 4 CONDITION
22F5 CA5423      JZ   2354   ; ถ้าครบ การทดสอบไปทำการปิดท้ายผลลัพท์ด้วย EE และ CLEAR PORT แล้ว RETURN กลับไป
22F8 Ø4          INR  B      ; ถ้าไม่ครบ 4 CONDITION ให้เพิ่ม B=B+1 คือ INPUT LOGIC STAGE ต่อไป
22F9 C3E522      JMP  22E5   ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 32

CLEAR LOGIC INPUT ที่ค้างอยู่ที่ PORT ให้เป็น Ø

```
22FC 3EØØ        MVI  A,ØØ   ; A = ØØ
22FE D3EØ        OUT  EØ     ; CLEAR PORT A, DEVICE # EØ
2300 D3E1        OUT  E1     ; CLEAR PORT B, DEVICE # E1
2302 D3E2        OUT  E2     ; CLEAR PORT C, DEVICE # E2
2304 C9          RET         ;
```

โปรแกรมย่อย P 33

CLEAR INPUT/OUTPUT BINARY DISPLAY

```
2305 ØØ          NOP         ;
2306 ØØ          NOP         ;
```

```
2307 ØØ          NOP          ;
2308 ØØ          NOP          ;
2309 3EØØ        MVI  A,ØØ    ; SET A = ØØ
230B D3BØ        OUT  BØ      ;
230D D3B1        OUT  B1      ; CLEAR BINARY INPUT LOGIC
230F D3B2        OUT  B2      ;
2311 D3CØ        OUT  CØ      ;
2313 D3C1        OUT  C1      ; CLEAR BINARY OUTPUT LOGIC
2315 D3C2        OUT  C2      ;
2317 C9          RET          ;
```

โปรแกรมย่อย P 34

CLEAR LED 7-SEGMENT ที่แสดง BAD PART, DEVICE # 78-7F ตามลำดับ

```
2318 3EØØ        MVI  A,ØØ    ; SET A = ØØ
231A D378        OUT  78      ; CLEAR 7-SEGMENT # 78
231C D379        OUT  79      ; CLEAR 7-SEGMENT # 79
231E D37A        OUT  7A      ; CLEAR 7-SEGMENT # 7A
2320 D37B        OUT  7B      ; CLEAR 7-SEGMENT # 7B
2322 D37C        OUT  7C      ; CLEAR 7-SEGMENT # 7C
2324 D37D        OUT  7D      ; CLEAR 7-SEGMENT # 7D
2326 D37E        OUT  7E      ; CLEAR 7-SEGMENT # 7E
2328 D37F        OUT  7F      ; CLEAR 7-SEGMENT # 7F
232A C9          RET          ;
```

โปรแกรมย่อย P 35

CLEAR LED 7-SEGMENT ที่แสดงเบอร์ IC ตั้งแต่ #70-77 ตามลำดับ

```
232B 3EØØ        MVI  A,ØØ    ; SET A = ØØ
```

```
232D D370       OUT   70      ;CLEAR 7-SEGMENT # 70
232F D371       OUT   71      ;CLEAR 7-SEGMENT # 71
2331 D372       OUT   72      ;CLEAR 7-SEGMENT # 72
2333 D373       OUT   73      ;CLEAR 7-SEGMENT # 73
2335 D374       OUT   74      ;CLEAR 7-SEGMENT # 74
2337 D375       OUT   75      ;CLEAR 7-SEGMENT # 75
2339 D376       OUT   76      ;CLEAR 7-SEGMENT # 76
233B D377       OUT   77      ;CLEAR 7-SEGMENT # 77
233D C9         RET           ;
```

โปรแกรมย่อย P 36

แสดงผลทาง LED ว่า BAD และ DELAY นาน ๑ วินาที

```
233E F5         PUSH PSW      ;SAVE A และ STATUS FLAG
233F 3E01       MVI  A,01     ;SET ค่า A=01(ตรงกับ DATA BIT ที่0=D0="1")
2341 D300       OUT  00       ;ส่ง OUT ให้หลอด LED ติด (LED นี้จะมีอักษร
                              ;กำกับว่า BAD)
2343 D310       OUT  10       ;พร้อมทั้งส่งเสียงออกมา 1 TONE
2345 F1         POP  PSW      ;คืน A และ STATUS FLAG , กลับไปอย่างเดิม
2346 CDE420     CALL 20E4     ;เรียกโปรแกรมย่อย DELAY นาน ๑ วินาที
2349 C9         RET           ;
```

โปรแกรมย่อย P 37

แสดงผลทาง LED ว่า GOOD และ DELAY นาน ๑ วินาที

```
234A F5         PUSH PSW      ;SAVE A และ STATUS FLAG
234B 3E02       MVI  A,02     ;SET ค่า A=02 (ตรงกับ DATA BIT ที่ 1
                              ;D1="1")
234D D300       OUT  00       ;ส่ง OUT ให้หลอด LED ติด ( LED นี้จะมีอักษร
                              ;กำกับว่า GOOD)
234F F1         POP  PSW      ;คืน Aและ STATUS FLAG กลับไปอย่างเดิม
2350 CDE420     CALL 20E4     ;เรียกโปรแกรมย่อย DELAY นาน ๑ วินาที
2353 C9         RET           ;
```

โปรแกรมย่อย P38

SET ค่าปิดท้าย เมื่อทำการตรวจสอบเสร็จสิ้นทุก PART แล้วและรายงานผล

```
2354 CDCF25     CALL 25CF    ; SET 'EE' ปิดท้ายผลลัพท์ที่ได้และ CLEAR PORT
2357 CD6E23     CALL 236E    ; รายงานผลทั้งหมดว่า GOOD หรือ BAD
235A C9         RET          ;
```

โปรแกรมย่อย P39

โปรแกรมช่วยในการทดสอบ PART 4 ของ IC ประเภทที่ ๑

```
235B 78         MOV  A,B     ; เคลื่อน INPUT LOGIC ที่จะขับเข้าขา IC จาก B ไว้ที่ A
235C Ø7         RLC          ;
235D Ø7         RLC          ; เลื่อน LOGIC CONDITION ให้ตรงกับขา INPUT
235E Ø7         RLC          ; ของ IC ขา 13
235F E6Ø8       ANI  Ø8      ; MASK เฉพาะBIT ที่ 3(D3)ขา IC ขาที่ 13
2361 4F         MOV  C,A     ; SAVE LOGIC ที่จะ DRIVE ขา 13ไว้ก่อนที่ C
2362 78         MOV  A,B     ; เอาINPUT LOGIC เข้าบาใหม่ที่ A
2363 Ø7         RLC          ; เลื่อน LOGIC CONDITION ให้ตรงกับขา INPUT ของ IC ขา 12
2364 E6Ø4       ANI  Ø4      ; MASK เฉพาะ BIT ที่ 2(D2) ขา IC ที่ 12
2366 B1         ORA  C       ; รวมผล LOGIC ที่จะ DRIVE เข้า IC เข้าด้วยกันก่อน DRIVE
2367 C9         RET          ;
```

โปรแกรมย่อย P4Ø

ROTATE RIGHT CARRY

```
2368 ØF         RRC          ;
2369 ØF         RRC          ;
236A ØF         RRC          ;
236B ØF         RRC          ;
236C ØF         RRC          ;
236D C9         RET          ;
```

```
236E CD0B24    CALL 240B    ;CLEAR MEMORY ที่แสดงผล BAD PART และ
                            ;LED 7 SEGMENT # 78-7F
2371 CD5E24    CALL 245F    ;CLEAR FLAG GOOD/BADและCLEAR I/O BINARY
                            ;DISPLAY
2374 D377      OUT  77      ; CLEAR LED7-SEGMENT#77
2376 216010    LXI  H,1060  ; POINTERชี้จุดเริ่มต้นผลลัพธ์ในการทดสอบแต่ละPART
2379 0601      MVI  B,01    ; SET B=01 บอกให้ทราบว่าเป็น PART 1
237B 7E        MOV  A,M     ; เอาผลลัพธ์มาทดสอบ
237C FE00      CPI  00      ; เปรียบเทียบ=00 หรือไม่ (00 = BAD)
237E C28B23    JNZ  238B    ; ถ้าไม่เท่ากับ 00 ไปตรวจต่อไปที่ 238B
2381 78        MOV  A,B     ; ถ้า BAD เอาค่า 01 ใน B ไปเก็บใน A
2382 CD0222    CALL 2202    ; LOOK UP TABLE จาก HEX เป็น7-SEGMENT DISPLAY
2385 CDA423    CALL 23A4    ; เอาค่าที่ได้เก็บใน MEMORY ที่ตรงกับ PART นั้น ๆ
2388 C35524    JMP  2455    ;SET FLAG GOOD/BAD ว่าBAD และทดสอบ
                            ; ผลที่ได้ต่อไป

238B FEEA      CPI  EA      ;ตรวจว่าจบ PART หรือยัง
238D CA9923    JZ   2399    ;ถ้าหมด PART แล้วตรวจผล PART ต่อไป
2390 FEEE      CPI  EE      ;ตรวจว่าถึง PART สุดท้ายหรือยัง
2392 CA9E23    JZ   239E    ;ถ้าถึง PART สุดท้าย แสดงผลทาง 7-SEGMENT
                            ;ว่ามีส่วนใดเสีย
2395 23        INX  H       ;เพิ่มค่า HL = HL + 1
2396 C37B23    JMP  237B    ;กลับไปทดสอบว่า GOOD หรือ BAD ต่อไป

2399 04        INR  B       ;เพิ่ม B = B+1
239A 23        INX  H       ;เพิ่มค่า HL = HL + 1
239B C37B23    JMP  237B    ;กลับไปทดสอบว่า GOOD หรือ BAD ต่อไป
```

```
239E D320       OUT  20        ;ส่งเสียงว่า สรุปผลได้ดังนี้ (เสียง 4 STEP)
23A0 CD2224     CALL 2422      ;สรุปผลโดยแสดงผลทาง LED7 - SEGMENT
23A3 C9         RET            ;
```

โปรแกรมย่อย P41

PART ใดเสียก็จะเอาค่านั้นไปเก็บใน MEMORY ที่ตรงกับ PART นั้น ๆ

```
23A4 F5         PUSH PSW       ; SAVE Aและ FLAG
23A5 E5         PUSH H         ; SAVE HL
23A6 FE06       CPI  06        ; ตรวจว่าเป็น PART1 ที่เสียใช่หรือไม่(06=PART1)
23A8 CAD323     JZ   23D3      ; ถ้าใช่เอาค่า 06 เก็บใน MEMORY ที่ 4A09
23AB FE5B       CPI  5B        ; ตรวจว่าเป็นPART2ที่เสียใช่หรือไม่(5B=PART2)
23AD CADA23     JZ   23DA      ; ถ้าใช่นำค่า5B เก็บใน MEMORY ที่ 4E0A
23B0 FE4F       CPI  4F        ; ตรวจว่าเป็น PART 3 ที่เสียใช่หรือไม่ (4F=PART3)
23B2 CAE123     JZ   23E1      ; ถ้าใช่นำค่า4F เก็บใน MEMORY ที่ 4E0B
23B5 FE66       CPI  66        ; ตรวจว่าเป็นPART4 ที่เสียใช่หรือไม่(66=PART4)
23B7 CAE823     JZ   23E8      ; ถ้าใช่นำค่า 66เก็บในMEMORY ที่ 4E0C
23BA FE6D       CPI  6D        ; ตรวจว่าเป็นPART5 ที่เสียใช่หรือไม่(6D=PART5)
23BC CAEF23     JZ   23EF      ; ถ้าใช่นำค่า 6D เก็บใน MEMORY ที่ 4E0D
23BF FE7D       CPI  7D        ; ตรวจว่าเป็น PART6ที่เสียใช่หรือไม่(7D=PART6)
23C1 CAF623     JZ   23F6      ; ถ้าใช่นำค่า 7D เก็บใน MEMORY ที่ 4E0E
23C4 FE07       CPI  07        ; ตรวจว่าเป็นPART7 ที่เสียใช่หรือไม่ (07=PART7)
23C6 CAFD23     JZ   23FD      ; ถ้าใช่นำค่า 07 เก็บใน MEMORY ที่ 4E0F
23C9 FE7F       CPI  7F        ; ตรวจว่าเป็น PART8ที่เสียใช่หรือไม่(7F=PART8)
23CB CA0424     JZ   2404      ; ถ้าใช่นำค่า 7F เก็บใน MEMORY ที่ 4E10
23CE D310       OUT  10        ; ส่งเสียง ERROR สำหรับว่าตรวจแล้วไม่เจอ PART ที่แจ้งว่าเสีย
23D0 C35A23     JMP  235A      ; RETURN
```

โปรแกรมย่อย P42

เอาผลที่ได้เก็บไว้ใน MEMORY PART 1 ที่ (4EØ9)

```
23D3 21094E     LXI  H,4EØ9  ; ENTRY POINT ของ PART 1 ที่ 4EØ9
23D6 77         MOV  M,A     ;
23D7 E1         POP  H       ;
23D8 F1         POP  PSW     ;
23D9 C9         RET          ;
```

โปรแกรมย่อย P43

เอาผลที่ได้เก็บไว้ใน MEMORY PART 2 ที่ (4EØA)

```
23DA 210A4E     LXI  H,4EØA  ; ENTRY POINT ของ PART 1 ที่ 4EØA
23DD 77         MOV  M,A     ;
23DE E1         POP  H       ;
23DF F1         POP  PSW     ;
23E0 C9         RET          ;
```

โปรแกรมย่อย P44

เอาผลที่ได้เก็บไว้ใน MEMORY PART 3 ที่ (4EØB)

```
23E1 210B4E     LXI  H,4EØB  ; ENTRY POINT ของ PART 1 ที่ 4EØB
23E4 77         MOV  M,A     ;
23E5 E1         POP  H       ;
23E6 F1         POP  PSW     ;
23E7 C9         RET          ;
```

โปรแกรมย่อย P45

เอาผลที่ได้เก็บไว้ใน MEMORY PART 4 ที่ (4EØC)

```
23E8 210C4E     LXI  H,4EØC  ;ENTRY POINT ของ PART 1 ที่4EØC
23EB 77         MOV  M,A     ;
23EC E1         POP  H       ;
23ED F1         POP  PSW     ;
23EE C9         RET          ;
```

โปรแกรมย่อย P46

เอาผลที่ได้เก็บไว้ใน MEMORY PART 5 ที่ (4EØD)

```
23EF 210D4E     LXI  H,4EØD  ; ENTRY POINT ของ PART 5อยู่ที่ 4EØD
23F2 77         MOV  M,A     ;
23F3 E1         POP  H       ;
23F4 F1         POP  PSW     ;
23F5 C9         RET          ;
```

โปรแกรมย่อย P47

เอาผลที่ได้เก็บไว้ใน MEMORY PART6 ที่ (4EØE)

```
23F6 210E4E     LXI  H,4EØE  ; ENTRY POINT ของ PART 6 อยู่ที่ 4EØE
23F9 77         MOV  M,A     ;
23FA E1         POP  H       ;
23FB F1         POP  PSW     ;
23FC C9         RET          ;
```

โปรแกรมย่อย P48

เอาผลที่ได้เก็บไว้ใน MEMORY PART 7 ที่ (4E0F)

```
23ED 210F4E     LXI  H,4E0F  ; ENTRY POINT ของ PART 7 ที่ 4E0F
2400 77         MOV  M,A     ;
2401 E1         POP  H       ;
2402 F1         POP  PSW     ;
2403 C9         RET          ;
```

โปรแกรมย่อย P49

เอาผลที่ได้เก็บไว้ใน MEMORY PART 8 ที่ (4E10)

```
2404 21104E     LXI  H,4E10  ; ENTRY POINT ของ PART 8 อยู่ที่ 4E10
2407 77         MOV  M,A     ;
2408 E1         POP  H       ;
2409 F1         POP  PSW     ;
240A C9         RET          ;
```

โปรแกรมย่อย P50

CLEAR MEMORY ที่แสดงผลส่วนที่เสีย (BAD PART) และ CLEAR LED7-SEGMENT#78-7F

```
240B F5         PUSH PSW     ; SAVE Acc และFLAG
240C E5         PUSH H       ; SAVE HL
240D C5         PUSH B       ; SAVE BC
240E 0608       MVI  B,08    ; B เก็บจำนวน DIGIT=8DIGITของ7-SEGMENT
2410 21094E     LXI  H,4E09  ; POINTER ข้อมูล BAD PART DISPLAYเริ่มที่4E09
2413 3E00       MVI  A,00    ; SET A = 00
2415 77         MOV  M,A     ;
2416 23         INX  H       ; CLEAR MEMORY ทั้ง 8ที่ตั้งแต่ 4E09-4E10
2417 05         DCR  B       ;
```

```
2418 C21524     JNZ  2415    ;
241B CD1823     CALL 2318    ; CLEAR LED7-SEGMENT DISPLAY,DIVICE#78-7F
241E C1         POP  B       ; คืน BC
241F E1         POP  H       ; คืน HL
2420 F1         POP  PSW     ; คืน Acc และFLAG
2421 C9         RET          ;
```

โปรแกรมย่อย P 51

แสดงผลว่าส่วนใดเสีย (BAD PART) หรือ GATE ใดเสียทาง LED 7-SEGMENT

```
2422 21094E     LXI  H,4E09  ; POINTER ชี้จุดเริ่มต้นของ MEMORY PART
                             ; ที่เก็บผลไว้ว่าดีหรือเสีย
2425 7E         MOV  A,M     ;
2426 D378       OUT  78      ; 7-SEGMENT # 78 แสดง PART ที่ 1
2428 23         INX  H       ; ชี้ MEMORY ถัดไป (4E0A)
2429 7E         MOV  A,M     ;
242A D379       OUT  79      ; 7-SEGMENT # 79 แสดงPART ที่ 2
242C 23         INX  H       ; ชี้ MEMORY ถัดไป (4E0B)
242D 7E         MOV  A,M     ;
242E D37A       OUT  7A      ; 7-SEGMENT # 7A แสดง PART ที่ 3
2430 23         INX  H       ; ชี้ MEMORY ถัดไป (4E0C)
2431 7E         MOV  A,M     ;
2432 D37B       OUT  7B      ; 7-SEGEMENT # 7Bแสดง PART ที่ 4
2434 23         INX  H       ; ชี้ MEMORY ถัดไป (4E0D)
2435 7E         MOV  A,M     ;
2436 D37C       OUT  7C      ; 7-SEGMENT # 7C แสดง PART ที่ 5
2438 23         INX  H       ; ชี้ MEMORY ถัดไป (4E0E)
2439 7E         MOV  A,M     ;
243A D37D       OUT  7D      ; 7-SEGMENT # 7D แสดง PART ที่ 6
```

```
243C 23          INX  H       ; ดู MEMORY ถัดไป (4E0F)
243D 7E          MOV  A,M     ;
243E D37E        OUT  7E      ; 7-SEGMENT # 7E แสดง PART ที่ 7
2440 23          INX  H       ; ดูMEMORY ถัดไป (4E10)
2441 7E          MOV  A,M     ;
2442 C36A24      JMP  246A    ; ไปทำต่อที่ 246A
```

โปรกแกรมย่อย P 52

CLEAR LED GOOD/BAD แล้วกลับไปเริ่มต้น PROGRAM ใหม่

```
2445 D350        OUT  50      ; CLEAR LED แสดงผล GOOD/BAD
2447 C30020      JMP  2000    ; กลับไปเริ่มต้นโปรแกรมใหม่
```

โปรแกรมย่อย P 53

SETค่า 'EA' บิดท้ายผลการทดสอบแต่ละ PART

```
244A EB          XCHG         ; สลับ HL กับ DE
244B 36EA        MVI  M,EA    ; SET ค่า'EA' บิดท้ายข้อมูล เมื่อทดสอบหมด
244D 23          INX  H       ; เพิ่ม HL= HL+ 1
244E EB          XCHG         ; สลับ HL กลับ DE
244F C9          RET          ;
```

โปรแกรมย่อย P 54

SETค่า EE เพื่อบิดท้ายผลการทดสอบสุดท้าย

```
2450 EB          XCHG         ; สลับ HL กับ DE
2451 36EE        MVI  M,EE    ; SET ค่า'EE' ไว้ท้ายข้อมูล ผลทั้งหมดเมื่อเสร็จ
                              ; สิ้นการทดสอบ
2453 EB          XCHG         ; สลับ HL กับ DE
2454 C9          RET          ;
```

โปรแกรมย่อย P 55

SET FLAG ที่ 4 FFC ซึ่งแสดง BAD = (FF)HEX

```
2455 E5        PUSH H       ; SAVE HL
2456 21FC4F    LXI  H,4FFC  ; FLAG GOOD/BAD อยู่ที่  4FFC
2459 36FF      MVI  M,FF    ; SET เป็นFF (แสดงว่า BAD)
245B E1        POP  H       ; คืน  HL
245C C39A23    JMP  239A    ; กลับไปทดสอบ PART  ต่อไป
```

โปรแกรมย่อย P 56

CLEAR FLAG ที่4FFC ซึ่งแสดง GOOD หรือ BAD และ CLEAR BINARY DISPLAY ทั้งหมด

```
245F E5        PUSH H       ; SAVE HL
2460 21FC4F    LXI  H,4FFC  ; FLAG GOOD/BADอยู่ที่  4FFC
2463 3600      MVI  M,00    ; CLEAR FLAG M4FFC = 00
2465 E1        POP  H       ; คืนHL
2466 CD0923    CALL 2309    ; CLEAR INPUT และ OUTPUT BINARY DISPLAY
2469 C9        RET          ;
```

โปรแกรมย่อย P 57

ช่วยแสดง PART ที่เสียทาง LED # SEGMENT และแสดงผลทาง LED ว่า GOOD/BAD

```
246A D37E      OUT  7E      ; 7-SEGMENT # 7Fแสดง PART ที่ 8
246C 21FC4F    LXI  H,4FFC  ; PIONTER FLAG GOOD/BAD
246F 7E        MOV  A,M     ; เอามาตรวจสอบเก็บไว้ที่ A
2470 FEFF      CPI  FF      ; ตรวจสอบว่าBAD หรือไม่ (FF = BAD)
2472 C27E24    JNZ  247E    ; ถ้าไม่ BAD แสดงผลว่า GOOD
2475 3E01      MVI  A,01    ; ถ้า BADแสดงผลว่าBAD, SET A=01(ตรงกับ$D_0$)
2477 D300      OUT  00      ; แสดงผลทาง LED ว่า BAD
2479 D310      OUT  10      ; ส่งเสียง 1TONE ยาว
247B C38224    JMP  2482    ; ไปตรวจว่าจะทดสอบต่อหรือไม่โดยตรวจทางKEYBOAR
```

โปรแกรมย่อย P 58

แสดงผลว่า GOOD ทาง LED และกลับไปตรวจว่าจะทดสอบต่อหรือไม่ โดยตรวจ KEYBOARD

```
247E 3E02       MVI  A,02    ; SET A = 02 (ตรงกับ D1)
2480 D300       OUT  00      ; แสดงผลที่หลอด LED ว่า GOOD
2482 CD0028     CALL 2800    ; ดูว่ากด KEYBOARD อะไรเข้ามา
2485 FE15       CPI  15      ; ตรวจว่าจะทดสอบต่อหรือไม่ (ปุ่ม CONTINUE)
2487 CACB25     JZ   25CB    ; ถ้าต้องการทดสอบต่อไป CLEAR 7-SEGMENT BAD PART DISPLAY ทั้งหมดแล้วรอคำสั่ง TEST
248A FE16       CPI  16      ; ตรวจว่า กดปุ่ม CLEAR หรือไม่
248C CA4524     JZ   2445    ; ถ้ากดปุ่ม CLEAR กลับไปเริ่มต้น PROGRAM ใหม่
248F D310       OUT  10      ; ส่งเสียง ERROR กด KEY ผิด 1TONE ยาว
2491 C38224     JMP  2482    ; จนรอว่ากดปุ่มอะไรต่อไป
2494 C9         RET          ;
```

โปรแกรมย่อย P 59

SET PIN IC 14 ขา , ขาINPUT/ ขา OUTPUT = ½, 3/4, 5/6, 9/8, 11/10, 13/12

IC ประเภทที่ 2 SOCKET#1

```
2495 3E6A       MVI  A,6A    ;
2497 D3F0       OUT  F0      ; ส่ง OUTPUT ออกไปทาง PORT A # F0
2499 3E05       MVI  A,05    ;
249B D3F1       OUT  F1      ; ส่ง OUTPUT ออกไปทาง PORT B # F1
249D C9         RET          ;
```

โปรแกรมย่อย P 60

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 1

```
249E 3E06       MVI  A,06    ; '06' = เลข 1
24A0 D377       OUT  77      ; แสดงผล 7-SEGMENT # 77
24A2 C9         RET          ;
```

โปรแกรมย่อย P 61

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 2

```
24A3 3E5B       MVI  A,5B    ; '5B' =  เลข 2
24A5 D377       OUT  77      ;แสดงผล 7-SEGMENT # 77
24A7 C9         RET          ;
```

โปรแกรมย่อย P 62

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 3

```
24A8 3E4F       MVI  A,4F    ; '4F' = เลข 3
24AA D377       OUT  77      ; แสดงผล 7-SEGMENT # 77
24AC C9         RET          ;
```

โปรแกรมย่อย P 63

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 4

```
24AD 3E66       MVI  A,66    ; '66' = เลข 4
24AF D377       OUT  77      ; แสดงผล 7 - SEGMENT # 77
24B1 C9         RET          ;
```

โปรแกรมย่อย P 64

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 5

```
24B2 3E6D       MVI  A,6D    ; '6D' =  เลข 5
24B4 D377       OUT  77      ; แสดงผล 7-SEGMENT # 77
24B6 C9         RET          ;
```

โปรแกรมย่อย P 65

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 6

```
24B7 3E7D       MVI  A,7D    ; '7D' =  เลข 6
```

```
24B9 D377          OUT  77          ; แสดงผล 7-SEGMENT # 77
24BB C9            RET              ;
```

โปรแกรมย่อย P66

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 7

```
24BC 3E07          MVI  A,07        ; '07' = เลข 7
24BE D377          OUT  77          ; แสดงผล 7-SEGMENT # 77
24C0 C9            RET              ;
24C1 3E3E          MVI  A,3E        ;
24C3 07            RLC              ;
24C4 D377          OUT  77          ;
24C6 C9            RET              ;
```

โปรแกรมย่อย P67

แสดงผลทาง 7-SEGMENT ว่าเป็น PART 8

```
24C7 3E7F          MVI  A,7F        ; '7F' = เลข 8
24C9 D377          OUT  77          ; แสดงผล 7-SEGMENT # 77
24CB C9            RET              ;
```

โปรแกรมย่อย P68

แสดงผล BINARY และ HEX ที่ 7-SEGMENT และตรวจสอบผลตรงกับ TRUTH TABLE หรือไม่

```
24CC F5            PUSH PSW         ; SAVE A และFLAG
24CD CD6922        CALL 2269        ; แสดงผล BINARY LOGIC OUTPUT ที่ LED
                                    ; และ HEX LOGIC OUTPUT ที่ 7-SEGMENT
24D0 F1            POP  PSW         ; คืน A และ FLAG
24D1 CD4922        CALL 2249        ; ตรวจสอบว่าผลที่ได้ตรงกับ DATA หรือไม่ SET FLA
                                    ; GOOD/BAD และ DELAY นาน • วินาที
                                                              (0กับ 1)
24D4 C3C525        JMP  25C5        ; ไปตรวจว่าทำครบ 2 CONDITION ทั้งหมดหรือยัง
```

โปรแกรมย่อย P69

ถอยหลังกลับไปดู DATA เริ่มต้นใหม่และ SET INPUT LOGIC เริ่มต้นที่ ØØ ใหม่

```
24D7 2B          DCX  H      ; ถอยหลังกลับไปดู DATA จาก TRUTH TABLE -
24D8 2B          DCX  H      ; เริ่มใหม่อีก
24D9 Ø6ØØ        MVI  B,ØØ   ; SET CONDITION แรกใหม่คือ ØØ
24DB C9          RET         ;
```

โปรแกรมย่อย P 7Ø

ส่ง INPUT LOGIC เข้าขา IC และรับผลกลับ

```
24DC D3E1        OUT  E1     ; ส่ง INPUT LOGIC เข้าขา IC เพื่อทดสอบ,
                             ; DIVICE # E1
24DE DBD1        IN   D1     ; รับผล OUTPUT ได้กลับเข้าระบบ, DIVICE # D1
24EØ C9          RET         ;
```

โปรแกรมย่อย P 71

CLEAR 7-SEGMENT, DIVICE 77-7F และCLEAR INPUT/OUTPUT BINARY DISPLAY

```
24E1 116Ø1Ø      LXI  D,1Ø6Ø ; POINTER ชี้ที่เก็บผลการทดสอบเริ่มต้นที่จุดนี้
24E4 CD1823      CALL 2318   ; CLEAR LED 7-SEGMENT, DIVICE # 78-7F
24E7 CD0923      CALL 2309   ; CLEAR INPUT/OUTPUT BINARY DISPLAY
24EA D377        OUT  77     ; CLEAR 7-SEGMENT, DIVICE # 77
24EC C9          RET         ;
```

โปรแกรมย่อย P 72

ทดสอบ IC ประเภทที่2 ส่วนที่ 1(PART 1) SOCKET#1

```
24ED CD9E24      CALL 249E   ; แสดงทาง 7-SEGMENT เป็นเลข 1(PART 1)
24FØ Ø6ØØ        MVI  B,ØØ   ; CONDITION แรกคือ Ø
24F2 CDØF22      CALL 22ØF   ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน 16
24F5 78          MOV  A,B    ; ส่ง LOGIC CONDITION จาก B ไว้ที่ A
                             ; เพื่อเตรียมส่งออก
```

```
24F6 CD4422      CALL 2244    ;ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
24F9 E602        ANI  02      ;MASK เอา BIT ที่ 1 (D1)
24FB 0F          RRC          ;เลื่อน DATA ให้ตรงกับ TRUTH TABLE (ทางขวา ๑ ครั้ง)
24FC CDCC24      CALL 24CC    ;แสดงผล BI/HEX OUTPUT LOGIC , ทดสอบว่าผลที่ได้ถูกต้องหรือไม่
24FF CA0625      JZ   2506    ;ถ้าทำการทดสอบครบทุก CONDITION แล้วทดสอบ PART ที่ 2 ต่อไป
2502 04          INR  B       ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
2503 C3F224      JMP  24F2    ; กลับไปทดสอบ CONDITION ใหม่ต่อไป

                 โปรแกรมย่อย  P 73
        ทดสอบ IC ประเภทที่ 2 ส่วนที่ 2 (PART 2) SOCKET #-1
2506 CD4A24      CALL 244A    ; SET ค่าEA' บิดท้ายผลการทดสอบ PART ที่ 1
2509 CDA324      CALL 24A3    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART ที่ 2
250C CDD724      CALL 24D7    ; ถอยหลังกลับไปดู DATA เริ่มต้นใหม่และ SET CONDITION แรกเป็น 0
250F CD0F22      CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน16
2512 78          MOV  A,B     ; ส่ง LOGIC CONDITIONจากB ไว้ที่A แล้วเตรียมส่งออก
2513 07          RLC          ; } เลื่อนข้อมูลให้ตรงกับขา IC
2514 07          RLC          ; } (ทางซ้าย ๒ ครั้ง)
2515 CD4422      CALL 2244    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
2518 E608        ANI  08      ; MASKเอาBIT ที่ 3 (D3)
251A 0F          RRC          ; } เลื่อนข้อมูลเพื่อให้ตรงกับ TRUTH TABLE
251B 0F          RRC          ; } (ทางขวา ๓ ครั้ง)
251C 0F          RRC          ; }
251D CDCC24      CALL 24CC    ; แสดงผล BI/HEX OUTPUT LOGIC, ทดสอบว่าผลที่ได้ถูกต้องหรือไม่
2520 CA2725      JZ   2527    ; ถ้าทำการทดสอบครบทุก CONDITION แล้วทดสอบ PART ที่ 3 ต่อไป
2523 04          INR  B       ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
2524 C30F25      JMP  250F    ; กลับไปทดสอบ CONDITION ใหม่ต่อไป
```

โปรแกรมย่อย P 74

ทดสอบ ICประเภทที่ 2 ส่วนที่ 3 (PART 3) SOCKET#1

```
2527 CD4A24     CALL 244A     ; SET ค่า 'EA' บอกท้ายผลการทดสอบ PART ที่ 2
252A CDA824     CALL 24A8     ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART ที่ 3
252D CDD724     CALL 24D7     ; ถอยหลังกลับไปดู DATA เริ่มต้นใหม่และ SET CONDITION แรกเป็น Ø
2530 CD0F22     CALL 220F     ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน 16
2533 78         MOV  A,B      ; ส่ง LOGIC CONDITION จาก B ไว้ที่ A เพื่อเตรียมส่งออก
2534 Ø7         RLC           ;
2535 Ø7         RLC           ;  เลื่อนข้อมูลให้ตรงกับขา IC
2536 Ø7         RLC           ;  (ทางซ้าย ๔ ครั้ง)
2537 Ø7         RLC           ;
2538 CD4422     CALL 2244     ; ส่ง LOGIC DRIVE เข้าขา ICและรับผลกลับ
253B E620       ANI  20       ; MASK เอา BIT ที่ 5 ($D_5$)
253D Ø7         RLC           ;
253E Ø7         RLC           ;  เลื่อนข้อมูลเพื่อให้ตรงกับ TRUTH TABLE (ทางขวา ๓ ครั้ง)
253F Ø7         RLC           ;
2540 CDCC24     CALL 24CC     ; แสดงผล BI/HEX OUTPUT LOGIC, ทดสอบว่าผลที่ได้ถูกต้องหรือไม่
2543 CA4A25     JZ   254A     ; ถ้าทำการทดสอบครบทุก CONDITION แล้วทดสอบ PART ที่ 3 ต่อไป
2546 Ø4         INR  B        ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
2547 C33Ø25     JMP  2530     ; กลับไปทดสอบ CONDITION ใหม่ต่อไป
```

โปรแกรมย่อย P 75

ทดสอบ ICประเภทที่ 2 ส่วนที่ 4 (PART 4) SOCKET#1

```
254A CD4A24     CALL 244A     ; SETค่า 'EA' บอกท้ายผลการทดสอบ PART ที่ 3
254D CDAD24     CALL 24AD     ; แสดงผลทาง 7-SEGMENT ว่าเป็นPART ที่ 4
2550 CDD724     CALL 24D7     ; ถอยหลังกลับไปดู DATA เริ่มต้นใหม่และ SET CONDITION แรกเป็น Ø
2553 CD0F22     CALL 220F     ; แสดงผล INPUT LOGIC ทาง LEDเป็นเลขฐาน16
2556 78         MOV  A,B      ; ส่ง LOGIC CONDITION จาก Bไว้ที่ A เพื่อเตรียมส่งออก
```

```
2557 ØF          RRC          ; เลื่อนข้อมูลให้ตรงกับขา IC (ทางขวา ๑ ครั้ง)
2558 CD4422      CALL 2244    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
255B E640        ANI  40      ; MASK เอา BIT ที่ 6 (D6)
255D Ø7          RLC          ; เลื่อนข้อมูลที่ได้ให้ตรงกับ TRUTH TABLE
255E Ø7          RLC          ; (ทางซ้าย ๒ ครั้ง)
255F CDCC24      CALL 24CC    ; แสดงผล BI/HEX OUTPUT LOGIC, ทดสอบว่าผลที่ได้ถูกต้องหรือไม่
2562 CA6925      JZ   2569    ; ถ้าทำการทดสอบครบทุก CONDITION แล้วทดสอบ PART ที่ 5 ต่อไป
2565 Ø4          INR  B       ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
2566 C35325      JMP  2553    ; กลับไปทดสอบ CONDITION ใหม่ต่อไป
```

โปรแกรมย่อย P 76

ทดสอบ ICประเภทที่ 2 ส่วนที่ 5 (PART 5) SOCKET#1

```
2569 CDFC22      CALL 22FC    ; CLEAR LOGIC ที่ค้างอยู่ที่ PORT ให้หมดไป
256C CD4A24      CALL 244A    ; SETค่า'EA'ปิดท้ายผลการทดสอบPART ที่ 4
256F CDB224      CALL 24B2    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART ที่ 5
2572 CDD724      CALL 24D7    ; ถอยหลังหลับไปดู DATA เริ่มต้นใหม่ และ SET CONDITION แรกเป็น Ø
2575 CDØE22      CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน16
2578 78          MOV  A,B     ; ส่ง LOGIC CONDITION จาก Bไว้ที่ A เพื่อเตรียมส่งออก
2579 Ø7          RLC          ; เลื่อนข้อมูลให้ตรงกับขา IC(ทางซ้าย ๑ ครั้ง)
257A CDDC24      CALL 24DC    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
257D E601        ANI  Ø1      ; MASK เอา BIT ที่ Ø (DØ)
257F CDCC24      CALL 24CC    ; แสดงผล BI/HEX OUTPUT LOGIC, ทดสอบว่าผลที่ได้ที่ได้ถูกต้องหรือไม่
2582 CA8925      JZ   2589    ; ถ้าทำการทดสอบครบทุก CONDITION แล้วทดสอบ PART ที่ 6 ต่อไป
2585 Ø4          INR  B       ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
2586 C37525      JMP  2575    ; กลับไปทดสอบ CONDITION ใหม่ต่อไป
```

โปรแกรมย่อย P 77

ทดสอบประเภทที่ ๒ ส่วนที่ ๖ (PART-5) SOCKET # 1

```
2589 CDFC22     CALL 22FC    ; CLEAR LOGIC ที่ค้างอยู่ที่ PORT ให้หมดไป
258C CD4A24     CALL 244A    ; SET ค่า 'EA' ปิดท้ายผลการทดสอบPART ที่ 5
258F CDB724     CALL 24B7    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART ที่ 6
2592 CDD724     CALL 24D7    ; ถอยหลังกลับไปดู DATA เริ่มต้นใหม่และ SET
                             ; CONDITION แรกเป็น Ø
2595 CD0F22     CALL 220F    ; แสดงผลINPUT LOGIC ทาง LEDเป็นเลขฐาน16
2598 78         MOV  A,B     ; ส่ง LOGIC CONDITION จากB ไว้ที่ A เพื่อเตรียม
                             ; ส่งออก
2599 Ø7         RLC          ;
259A Ø7         RLC          ; เลื่อนข้อมูลให้ตรงกับขา IC
259B Ø7         RLC          ; (ทางซ้าย ๓ ครั้ง)
259C CDDC24     CALL 24DC    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
259F E604       ANI  04      ; MASK เอา BIT ที่ 2 ($D_2$)
25A1 ØF         RRC          ; เลื่อนข้อมูลที่ได้ให้ตรงกับ TRUTH TABLE
25A2 ØF         RRC          ; (ทางขวา ๒ ครั้ง)
25A3 CDCC24     CALL 24CC    ; แสดงผล BI/HEX OUTPUT LOGIC, ทดสอบว่า
                             ; ผลที่ได้ถูกต้องหรือไม่
25A6 CA5423     JZ   2354    ; ถ้าทำการทดสอบครบทุก CONDITION แล้ว SET
                             ; ค่า EE ปิดท้าย และสรุปผลต่อไป
25A9 Ø4         INR  B       ; ไม่ครบเปลี่ยน INPUT LOGIC ขั้นต่อไป
25AA C39525     JMP  2595    ; กลับไปทดสอบ CONDITION ใหม่ต่อไป
25AD ØØ         NOP          ;
25AE ØØ         NOP          ;
25AF ØØ         NOP          ;
25BØ ØØ         NOP          ;
25B1 ØØ         NOP          ;
25B2 ØØ         NOP          ;
25B3 ØØ         NOP          ;
25B4 CDE124     CALL 24E1    ; CLEAR 7-SEGMENT, DEVICE 77-7F และ
                             ; CLEAR I/O BINARY DISPLAY
25B7 CDE626     CALL 26E6    ; ทดสอบIC ประเภทที่ 1 SOCKET # 1เช่น 7400
25BA C9         RET          ;
```

```
25BB ØØ          NOP           ;
25BC ØØ          NOP           ;
25BD ØØ          NOP           ;
25BE ØØ          NOP           ;
25BF CD9524      CALL 2495     ;  SET PIN ขาIC ประเภทที่ 2
25C2 C3DA2Ø      JMP  2ØDA     ;  แสดง LED ติดที่ SOCKET#1, แสดง READY
                                  ตรวจว่ากดปุ่ม SLOW หรือไม่
```

โปรแกรมย่อย P 78

ตรวจผลว่าวิ่งครบ CONDITION หรือยัง ของ IC ประเภทที่ 2

```
25C5 23          INX  H        ; HL = HL + 1
25C6 13          INX  D        ; DE = DE + 1
25C7 78          MOV  A,B      ; เอาค่า INPUT LOGIC เก็บที่ A เพื่อเปรียบเทียบ
25C8 FEØ1        CPI  Ø1       ; เปรียบเทียบว่า INPUT LOGICเป็นØ1 หรือยัง
25CA C9          RET           ;
```

โปรแกรมย่อย P 79

((CLEAR LOGIC DRIVEและCLEAR BAD PART DISPLAY

```
25CB CD1823      CALL 2318     ; ไป CLEAR LED -7 SEGMENT BAD PART
25CE C9          RET           ;
```

โปรแกรมย่อย P 8Ø

SETค่า 'EE' ท้ายการทดสอบทั้งหมด และ CLEAR INPUT DRIVER

```
25CF CD5Ø24      CALL 2450     ; SET ค่า'EE'ปิดท้ายผลลัพท์ที่วิเคราะห์แล้ว
25D2 CDFC22      CALL 22FC     ; CLEAR LOGIC OUTPUT ที่ค้างอยู่ที่PORT ให้เป็นØ
25D5 C9          RET           ;
```

โปรแกรมย่อย P 81

โปรแกรมทดสอบ IC ประเภทที่ 2 SOCKET ที่#1 ได้แก่ 7404 05, 06, 14, 16

```
25D6 CDBF25     CALL 25BF     ; SETขาIC SOCKET #1 และตรวจว่ากดปุ่มอะไรเข้ามา
25D9 21240F     LXI  H,0F24   ; POINTER ชี้ TRUTH TABLE DATA OUTPUT
25DC CDE124     CALL 24E1     ; CLEAR DISPLAY
25DF CDED24     CALL 24ED     ; เริ่มทดสอบ 7404,
25E2 C3D625     JMP  25D6     ; กลับไปตรวจว่าจะทดสอบต่อหรือไม่
```

โปรแกรมย่อย P 82

โปรแกรมทดสอบ IC ประเภทที่ 2 SOCKET ที่ #1 ได้แก่ 7407, 17

```
25E5 CDBF25     CALL 25BF     ; {SET ขาIC SOCKET #1 และตรวจว่ากดปุ่ม
                              ; {TEST/CLEAR/CONT/SLOW
25E8 21260F     LXI  H,0F26   ; POINTER ชี้ TRUTH TABLE DATA OUTPUT
25EB CDE124     CALL 24E1     ; CLEAR DISPLAY
25EE CDED24     CALL 24ED     ; เริ่มทดสอบ 7407,17
25F1 C3E525     JMP  25E5     ; กลับไปตรวจว่าจะทดสอบต่อหรือไม่
```

โปรแกรมย่อย P 83

SET ขาIC ประเภทที่ 3 SOCKET ที่ #1 ได้แก่ 7402

```
25F4 CD7726     CALL 2677     ; SET ขา IC
                              ; {แสดง READY และตรวจว่ากดปุ่ม TEST หรือ
25F7 CDDA20     CALL 20DA     ; {CLEAR หรือ SLOW
25FA C9         RET           ;
```

โปรแกรมย่อย P 84

รอรับการกด KEY เบอร์ IC เข้ามา ถ้าเข้ามาแล้วตรวจว่าเป็นปุ่มใด เก็บข้อมูลที่ LOOK UP TABLE ที่ได้ไว้ใน A สำหรับแสดงผลทาง 7-SEGMENT DISPLAY

```
25FB 00         NOP           ;
25FC 00         NOP           ;
25FD 00         NOP           ;
```

```
25FE 00          NOP          ;
25FF E5          PUSH H       ; SAVE HL
2600 CD0028      CALL 2800    ; รอว่ากด KEY อะไรเข้ามา
2603 0600        MVI B,00     ; เมื่อกด KEY เข้ามาแล้ว CLEAR B
2605 FE13        CPI 13       ; X'13=19=19KEY ตรวจว่าจะเกิน 19 KEYหรือไม่
2607 D26526      JNC 2665     ; { ถ้ามากกว่า (13 )HEX ไปตรวจว่าเป็น KEY ใดกดเข้ามา
260A 21000F      LXI H,0E00   ; ถ้าน้อยกว่า (13 )HEX POINTER ชี้ตารางข้อมูล
260D 4F          MOV C,A      ; }
260E 09          DAD B        ; } LOOK UP TABLE
260F 7E          MOV A,M      ; เอาข้อมูลจากตารางไว้ที่ A
2610 E1          POP H        ; คืน HL กลับที่เดิม
2611 C9          RET          ;

                 โปรแกรมย่อย  P 85
                 ROTATE ทางซ้าย ๔ ครั้ง
2612 07          RLC          ;
2613 07          RLC          ;
2614 07          RLC          ;
2615 07          RLC          ;
2616 07          RLC          ;
2617 C9          RET          ;
2618 FE16        CPI 16       ; ตรวจว่าเป็นปุ่ม CLEAR หรือไม่
261A CA4C26      JZ 264C      ; { ถ้ากดปุ่ม CLEAR ไป CLEAR DISPLAY และ SET MEMORY
261D FE17        CPI 17       ; ตรวจว่าเป็นปุ่ม ENTER หรือไม่
261F CA2320      JZ 2023      ; ถ้ากดปุ่ม ENTER ไปเข้า PROGRAM ตรวจเบอร์ I
2622 C9          RET          ;
```

โปรแกรมย่อย P 86

SET MEMORY DISPLAY ให้เป็น FF ทุก ๆ LOCATION จาก 4EØØ ถึง 4EØ7

และลบ DISPLAY DIGIT #0-7

```
2623 E5          PUSH H       ; SAVE HL ไว้
2624 D5          PUSH D       ; DAVE DE ไว้
2625 F5          PUSH PSW     ; SAVE A และ FLAG
2626 16Ø8        MVI  D,Ø8    ; COUNTER INDEX 8 จำนวน (8 DIGIT)
2628 21Ø04E      LXI  H,4EØØ  ; POINTER MEMORY LOCATION 4EØØ
262B 36FF        MVI  M,FF    ; ค่าที่จะ SET ใน MEMORY =FF
262D 23          INX  H       ; เพิ่ม HL = HL = 1
262E 15          DCR  D       ; ลดค่า D = D - 1
262F C22B26      JNZ  262B    ; ถ้าไม่ครบ 8กลับไปทำต่อที่262B ถ้าครบทำคำสั่งต่อไป
2632 F1          POP  PSW     ; คืน A และ FLAG กลับที่เดิมจากที่ SAVE เอาไว้
2633 D1          POP  D       ; คืน DC ที่ SAVE เอาไว้
2634 E1          POP  H       ; คืน HL ที่ SAVE
2635 F5          PUSH PSW     ; SAVE AและFLAG
2636 3EØØ        MVI  A,ØØ   ; SET A = ØØ
2638 D37Ø        OUT  7Ø      ; CLEAR LED 7-SEGMENT # 7Ø
263A D371        OUT  71      ; CLEAR LED 7 - SEGMENT # 71
263C D372        OUT  72      ; CLEAR LED 7-SEGMENT # 72
263E D373        OUT  73      ; CLEAR LED 7-SEGMENT # 73
2640 D374        OUT  74      ; CLEAR LED 7-SEGMENT # 74
2642 D375        OUT  75      ; CLEAR LED 7-SEGMENT # 75
2644 D376        OUT  76      ; CLEAR LED 7-SEGMENT # 76
2646 CD5226      CALL 2652    ; เรียกPROGRAM ดับ LED 7-SEGMENT#77 ถึง7F
2649 ØØ          NOP          ;
264A F1          POP  PSW     ; คืน A และFLAG
264B C9          RET          ;
```

```
264C CD2326     CALL 2623    ; { SET MEMORY = FF ตั้งแต่ 4EØØ ถึง 4EØ7 และ
                               { CLEAR DISPLAY 7Ø-76
264F C30427     JMP  27Ø4    ; { ตรวจ KEYBOARD ว่าถูกกดเบอร์ IC เบอร์อะไร
                               { เข้ามาพร้อม DISPLAY ทาง 7-SEGMENT ด้วย
                               { ถ้าไม่มีการกด CPU จะรออยู่

2652 D377       OUT  77      ; ดับ LED 7-SEGMENT # 77
2654 D378       OUT  78      ; ดับ LED 7-SEGMENT # 78
2656 D379       OUT  79      ; ดับ LED 7-SEGMENT # 79
2658 D37A       OUT  7A      ; ดับ LED 7-SEGMENT # 7A
265A D37B       OUT  7B      ; ดับ LED 7-SEGMENT # 7B
265C D37C       OUT  7C      ; ดับ LED 7-SEGMENT # 7C
265E D37D       OUT  7D      ; ดับ LED 7-SEGMENT # 7D
2660 D37E       OUT  7E      ; ดับ LED 7-SEGMENT # 7E
2662 D37F       OUT  7F      ; ดับ LED 7-SEGMENT # 7F
2664 C9         RET          ;
2665 ØØØØ       NOP          ;
2667 ØØØØØØ     NOP          ;
266A FE14       CPI  14      ; ตรวจว่าเป็นปุ่ม ENTER ใช่หรือไม่
266C CA9826     JZ   2698    ; ถ้าใช่ไปเตรียมทดสอบแจ้ง READY
266F ØØØØ       NOP          ;
2671 ØØØØØØ     NOP          ;
2674 C31826     JMP  2618    ; ถ้าไม่ใช่กลับไป ตรวจที่เป็นปุ่ม CLEAR หรือTEST
                               ใช่หรือไม่
```

โปรแกรมย่อย P87

SETขา IC 14 ขา ขา INPUT/OUTPUT = 2,3/1; 5,6/4; 8,9/10; 11,12/13

```
                SOCKET#1 ประเภทที่ 3
2677 3E09       MVI  A,09    ; 
2679 D3F0       OUT  F0      ; ส่ง OUTPUT ไปที่ PORT A#FØ
267B 3E09       MVI  A,09    ;
267D D3F1       OUT  F1      ; ส่ง OUTPUT ไปที่ PORT A # F1
267F C9         RET          ;
```

```
2680 0000          NOP          ;
2682 0000          NOP          ;
2684 0000          NOP          ;
2686 E1            POP  H       ; SAVE HL
2687 71            MOV  M,C     ;
2688 CD0028        CALL 2800    ; รอรับผลจาก KEY BOARD ว่ากดอะไรเข้ามา
268B FE16          CPI  16      ; ถ้ามีตรวจว่าเป็นปุ่ม CLEAR เข้ามาหรือไม่
268D CA4C26        JZ   264C    ; ถ้าเป็นปุ่ม CLEAR ไป CLEAR DISPLAY และ
                                  SET MEMORY
2690 FE17          CPI  17      ; ตรวจว่าเป็นปุ่ม TEST หรือไม่
2692 CA2320        JZ   2023    ; ถ้าเป็นปุ่ม TEST ก็ไป DECODEเบอร์IC ที่ กดเข้ามา
2695 C38826        JMP  2688    ; ถ้าไม่ใช่กลับไปรอว่าจะกดปุ่ม CLEAR หรือ TEST
2698 0000          NOP          ;
269A 0000          NOP          ;
269C 0000          NOP          ;
269E 00000000      NOP          ;
26A1 0000          NOP          ;
26A3 0000          NOP          ;
26A5 0000          NOP          ;
26A7 00000000      NOP          ;
```

โปรแกรมย่อย P 88

ทดสอบ IC ประเภทที่ 1 SOCKET ที่ # 1 ได้แก่ 7408, 74132 (NAND GATE)

```
26AA CDD720        CALL 20D7    ; SET ขา IC สำหรับ 7408, 74132 และ
                                  ตรวจ KEYBOARD
26AD 21280F        LXI  H,0F28  ; POINTER ชี้ TRUTH TABLE ที่ 0F28
26B0 CDB425        CALL 25B4    ; ทดสอบ IC 7408, 74132
26B3 C3AA26        JMP  26AA    ; กลับไปรอดูว่าจะทดสอบซ้ำต่ออีกหรือ CLEAR หรือ
                                  CONT. หรือ SLOW TEST
```

โปรแกรมย่อย P89

ทดสอบ IC ประเภทที่ 1 SOCKETที่ # 1 ได้แก่ 7432 (OR GATE)

```
26B6 CDD720     CALL 20D7     ; SET ขา IC สำหรับ 7432 และตรวจ KEYBOARD
26B9 212C0F     LXI H,0E2C    ; POINTER ชี้ TRUTH TABLE ที่ 0F2C
26BC CDB425     CALL 25B4     ; ทดสอบ IC 7432
26BF C3B626     JMP 26B6      ; กลับไปรอดูว่าจะทดสอบซ้ำต่ออีก(COUNT) หรือ CLEAR หรือ SLOW TEST
```

โปรแกรมย่อย P90

ทดสอบ IC ประเภทที่ 1 SOCKET ที่ # 1ได้แก่ 7486 (EXCLUSIVE - OR GATE)

```
26C2 CDD720     CALL 20D7     ; SET ขา IC สำหรับ 7486 และตรวจ KEYBOARD
26C5 21300F     LXI H,0F30    ; POINTER ชี้ TRUTH TABLE ที่ 0F30
26C8 CDB425     CALL 25B4     ; ทดสอบ IC 7486
26CB C3C226     JMP 26C2      ; กลับไปดูว่าจะทดสอบซ้ำต่ออีก CONT , หรือ CLEAR หรือ SLOW TEST
```

โปรแกรมย่อย P91

ทดสอบ IC ประเภทที่ 1 SOCKET ที่ #1 ได้แก่ 74125 (6 TRI - STATE INVERTER)

```
26CE CDD720     CALL 20D7     ; SET ขา IC สำหรับ 74125 และตรวจ KEYBOARD
26D1 21340F     LXI H,0F34    ; POINTER ชี้ TRUTH TABLE ที่ 0F34
26D4 CDB425     CALL 25B4     ; ทดสอบ IC 74125
26D7 C3CE26     JMP 26CE      ; กลับไปรอดูว่าจะทดสอบซ้ำต่ออีก(CONT) หรือ CLEAR หรือ SLOW TEST
```

โปรแกรมย่อย P92

ทดสอบ IC ประเภทที่ 1 SOCKET ที่ #1 ได้แก่ 74126 (6TRI - STATE BUFFER)

```
26DA CDD720     CALL 20D7     ; SET ขาIC สำหรับ 74126 และตรวจKEYBOARD
26DD 21380F     LXI H,0F38    ; POINTER ชี้ TRUTH TABLE ที่0F38
26E0 CDB425     CALL 25B4     ; ทดสอบ IC 74126
26E3 C3DA26     JMP 26DA      ; กลับไปรอดูว่าจะทดสอบซ้ำต่ออีก (CONT)หรือ CLEAR หรือ SLOW TEST
```

โปรแกรมย่อย P93

ทดสอบ IC ประเภทที่ • SOCKET # 1 ได้แก่ 74ØØ, 74132

```
26E6 CD7122     CALL 2271    ; ทดสอบส่วนที่ 1 (PART 1)
26E9 CD8A22     CALL 228A    ; ทดสอบส่วนที่ 2 (PART 2)
26EC CDAD22     CALL 22AD    ; ทดสอบส่วนที่ 3 (PART 3)
26EF CDD822     CALL 22D8    ; ทดสอบส่วนที่ 4 (PART 4)
26F2 C9         RET          ;
```

โปรแกรมย่อย P94

ทดสอบ IC เบอร์ 74ØØ, 74132

```
26F3 CDD720     CALL 20D7    ; {SET ขา IC สำหรับ 74ØØ, 74132และ ตรวจ KEYBOARD
26F6 2120ØF     LXI H,ØF2Ø   ; POINTER ชี้ TRUTH TABLE ที่ ØF2Ø
26F9 CDE124     CALL 24E1    ; CLEAR 7-SEGMENT, CLEAR DISPLAY BINARY
26FC CDE626     CALL 26E6    ; ทดสอบ 74ØØ, 74132
26FF C3F326     JMP 26F3     ; {กลับไปรอดูว่าจะทดสอบซ้ำต่ออีก (CONT) หรือ SLOW หรือ TEST
```

((โปรแกรมย่อย P95

รอรับเบอร์ IC ที่กดเข้ามาและแสดงผลทาง 7-SEGMENT พร้อมกับเก็บข้อมูลเบอร์ICที่ได้ไว้ในMEMORY

```
2702 ØØ         NOP          ;
2703 ØØ         NOP          ;
2704 2100 4E    LXI H,4EØØ   ; POINTER ชี้ที่เก็บ DATA เมื่อ KEY เบอร์ ICเข้ามา
2707 CD3527     CALL 2735    ; {รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØØ
270A D370       OUT 7Ø       ; แสดงผลทาง 7-SEGMENT #7Ø
270C CD3527     CALL 2735    ; {รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ1
270F D371       OUT 71       ; แสดงผลทาง 7-SEGMENT #71
2711 CD3527     CALL 2735    ; {รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ2
2714 D372       OUT 72       ; แสดงผลทาง 7-SEGMENT #72
```

```
2716 CD3527     CALL 2735    ; รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ3
2719 D373       OUT  73      ; แสดงผลทาง 7-SEGMENT #73
271B CD3527     CALL 2735    ; รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ4
271E D374       OUT  74      ; แสดงผลทาง 7-SEGMENT #74
2720 CD3527     CALL 2735    ; รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ5
2723 D375       OUT  75      ; แสดงผลทาง 7-SEGMENT # 75
2725 CD3527     CALL 2735    ; รอการกด KEY รับผลจากการ KEY เก็บใน MEMORY ADDR 4EØ6
2728 D376       OUT  76      ; แสดงผลทาง 7-SEGMENT #76
272A CD3527     CALL 2735    ; รอการกด KEY รับผลจากทาง KEY เก็บใน MEMORY ADDR 4EØ7
272D D377       OUT  77      ; แสดงผลทาง 7-SEGMENT #77
                             ; รอการกด KEY เข้ามาอันสุดท้าย ถ้ากดก็จะ
272F CDFF25     CALL 25FF    ;
2732 C3ØØ2Ø     JMP  2ØØØ    ; กลับไปเริ่มต้น โปรแกรม ใหม่หมด
```

โปรแกรมย่อย P96

รอการกด KEY เข้ามา, เก็บข้อมูลลง MEMORY พร้อมกับ LOOK UP TABLE สำหรับ 7-SEGMENT DISPLAY และเลื่อน MEMORY ถัดไป

```
2735 CDFF25     CALL 25FF    ; รอว่ากด KEY ใดเข้ามาแล้ว LOOK UP TABLE ได้ CODE ใหม่สำหรับ 7-SEGMENT DISPLAY
2738 71         MOV  M,C     ; เมื่อมีการกด KEY เข้ามาก่อน LOOK UP TABLE ไว้ใน MEMORY (เบอร์ IC )
2739 23         INX  H       ; HL = HL + 1
273A C9         RET          ;
```

โปรแกรมย่อย P97

SET ขา IC ทดสอบ IC ประเภทที่ 3 SOCKETที่ #1ได้แก่ 7402

```
273B CDF425     CALL 25F4    ; SET ขา IC สำหรับ 74Ø2
273E 213CØF     LXI  H,ØF3C  ; POINTER ชี้ TRUTH TABLE ที่ ØF3C
2741 CDE124     CALL 24E1    ; CLEAR 7-SEGMENT, CLEAR DISPLAY BINARY
2744 CDE527     CALL 27E5    ; ทดสอบ 74Ø2
2747 C33B27     JMP  273B    ; กลับไปรอดูว่าจะทดสอบซ้ำอีก (CONT) หรือ CLEAR หรือ SLOW TEST
```

โปรแกรมย่อย P98

```
                    PROGRAM DELAY 1 SEC.
274A ØØ             NOP         ;
274B ØØ             NOP         ;
274C ØØ             NOP         ;
274D E5             PUSH H      ; SAVE HL
274E C5             PUSH B      ; SAVE BL
274F D5             PUSH D      ; SAVE DE
2750 26B0           MVI  H,B0   ; SETค่าH=BØ(สามารถADJUST DELAY TIME ได้)
2752 2E02           MVI  L,Ø2   ; SET ค่าL=Ø2
2754 4C             MOV  C,H    ; C=BØ
2755 0605           MVI  B,05   ; B=Ø5
2757 1668           MVI  D,68   ; D=68
2759 1EC1           MVI  E,C1   ; E=C1
275B 1D             DCR  E      ; E=E-1
275C C25B27         JNZ  275B   ; ถ้า E≠ØØ ให้กลับไปทำที่ 275B
275F 15             DCR  D      ; D=D-1
2760 C25927         JNZ  2759   ; ถ้า D≠ØØ ให้กลับไปทำที่ 2759
2763 05             DCR  B      ; B=B-1
2764 C25727         JNZ  2757   ; ถ้า B≠ØØ ให้กลับไปทำที่ 2757
2767 2D             DCR  L      ; L=L-1
2768 C25527         JNZ  2755   ; ถ้า L≠ØØ ให้กลับไปทำที่ 2755
276B D1             POP  D      ; คืน DE
276C C1             POP  B      ; คืน BC
276D E1             POP  H      ; คืน HL
276E C9             RET         ;
```

โปรแกรมย่อย P99

ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1 ส่วนที่ 1 (PART 1)

```
276F CD9E24     CALL 249E    ; แสดง PART ที่ 1 ทาง 7-SEGMENT #77
2772 0600       MVI  B,00    ; CLEAR B=00 เป็น DATA INPUT เริ่มต้น 00
2774 CD0F22     CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน 16
2777 78         MOV  A,B     ; ส่ง LOGIC CONDITION จาก B ไว้ที่ A เพื่อ
                             ; เตรียมส่งออก.
2778 07         RLC          ; เลื่อนข้อมูลให้ตรงกับขา IC (ทางซ้าย ๑ ครั้ง).
2779 CD4422     CALL 2244    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
277C E601       ANI  01      ; MASK เอา BIT ที่ 0 (D0)
277E 07         RLC          ; เลื่อนข้อมูลเพื่อให้ตรงกับ TRUTH TABLE
277F 07         RLC          ; (ทางซ้าย ๒ ครั้ง)
2780 CD5622     CALL 2256    ; แสดงผล BINARY LOGIC OUTPUT ตรวจสอบ
                             ; ผลที่ได้ว่าตรงกันหรือไม่ ตรวจสอบว่าทำครบ 4 CONDITI
2783 C8         RZ           ; ถ้าครบ 4 CONDITION ก็จะ RETURN
2784 04         INR  B       ; ไม่ครบเลื่อนไปทดสอบ CONDITION ถัดไป
2785 C37427     JMP  2774    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P100

ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1 ส่วนที่ 2 (PART 2)

```
2788 CD4A24     CALL 244A    ; SET ค่า 'EA' ปิดท้ายผลทดสอบ PART ที่ 1
278B CDA324     CALL 24A3    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART 2
278E CD6222     CALL 2262    ; ถอยหลังกลับไปดู TRUTH TABLE เริ่มต้นใหม่และ
                             ; SET CONDITION แรกใหม่
2791 CD0F22     CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน 16
2794 78         MOV  A,B     ; ค่าใน B เก็บที่ A
2795 07         RLC          ; เลื่อน INPUT LOGIC ให้ตรงขา IC
2796 07         RLC          ; (ทางซ้าย ๔ ครั้ง)
2797 07         RLC          ;
2798 07         RLC          ;
```

```
2799 CD4422      CALL 2244    ; ส่ง LOGIC DRIVE เข้าขา IC และรับผลกลับ
279C E608        ANI  08      ; MASK เอา BIT ที่ 3 (D3)
279E 0F          RRC          ; เลื่อนผลที่ได้ให้ตรงกับ TRUTH TABLE
279F CD5622      CALL 2256    ; แสดงผล BINARY LOGIC OUTPUT ตรวจสอบผลที่ได้ว่าตรงกันหรือไม่
27A2 C8          RZ           ; ตรวจสอบว่าทำครบ 4 CONDITION ถ้าครบ 4 CONDITION ก็จะ RETURN
27A3 04          INR  B       ; ไม่ครบเลื่อนไปทดสอบ CONDITION ถัดไป
27A4 C39127      JMP  2791    ; กลับไปทดสอบ CONDITION ต่อไป

                 โปรแกรมย่อย  P 101
          ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1 ส่วนที่ 3 (PART 3)
27A7 CD4A24      CALL 244A    ; SET ค่า'EA' ปิดท้ายผลทดสอบ PART ที่ 1
27AA CDA824      CALL 24A8    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART 3
27AD CD6222      CALL 2262    ; ถอยหลังกลับไปดู TRUTH TABLE เริ่มต้นใหม่ และ SET CONDITION แรกใหม่
27B0 CD0F22      CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน 16
27B3 78          MOV  A,B     ; ค่าใน B เก็บที่ A
27B4 0F          RRC          ; เลื่อน INPUT LOGIC ให้ตรงขา IC
27B5 0F          RRC          ; (ทางขวา ๒ ครั้ง)
27B6 D3E0        OUT  E0      ; ส่ง LOGIC DRIVE เข้าขา IC
27B8 DBD1        IN   D1      ; รับผลกลับ
27BA E601        ANI  01      ; MASK BIT ที่ 0 (D0)
27BC 07          RLC          ; เลื่อน LOGIC OUTPUT ที่ได้ให้ตรงกับ -
27BD 07          RLC          ; -TRUTH TABLE
27BE CD5622      CALL 2256    ; แสดงผล BINARY LOGIC OUTPUT ตรวจสอบผลที่ได้ว่าตรงกันหรือไม่ตรวจสอบว่าทำครบ4CONDITION
27C1 C8          RZ           ; ถ้าครบ 4 CONDITIONก็จะ RETURN
27C2 04          INR  B       ; ไม่ครบเลื่อนไปทดสอบ CONDITION ถัดไป
27C3 C3B027      JMP  27B0    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 1Ø2

ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1 ส่วนที่ 4 (PART 4)

```
27C6 CD4A24     CALL 244A   ; SET ค่า'EA' ปิดท้ายผลทดสอบ PART ที่ 1
27C9 CDAD24     CALL 24AD   ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART 4
27CC CD6222     CALL 2262   ; ถอยหลังกลับไปอ TRUTH TABLE เริ่มต้นใหม่และ
                            ; SET CONDITION แรกใหม่
27CF CD0F22     CALL 220F   ; แสดงผล INPUT LOGIC ทาง LED เป็นเลขฐาน16
27D2 78         MOV A,B     ; ค่าใน B เก็บที่ A
27D3 07         RLC         ; เลื่อน INPUT LOGIC ให้ตรงขา IC
                            ; (ทางซ้าย ๑ ครั้ง)
27D4 D3E1       OUT E1      ; ส่ง LOGIC DRIVE เข้าขา IC
27D6 DBD1       IN D1       ; รับผลกลับ
27D8 E608       ANI 08      ; MASK BIT ที่ที่ 3 ($D_3$)
27DA 0F         RRC         ; เลื่อน LOGIC OUTPUT ที่ได้ให้ตรงกับ
                            ; TRUTH TABLE
27DB CD5622     CALL 2256   ; แสดงผล BINARY LOGIC OUTPUT ตรวจสอบผลที่
                            ; ได้ว่าตรงกันหรือไม่ตรวจสอบว่าทำครบ4CONDITION
27DE CA5423     JZ 2354     ; ถ้าครบ 4 CONDITIONไปสรุปผล
27E1 04         INR B       ; ไม่ครบเลื่อนไปทดสอบ CONDITION ถัดไป
27E2 C3CF27     JMP 27CF    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 1Ø3

ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1 ได้แก่ 7402 ทุก ๆ ส่วน(PART)

```
27E5 CD6F27     CALL 276F   ; ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1
                            ; ส่วนที่ 1 (PART 1)
27E8 CD8827     CALL 2788   ; ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1
                            ; ส่วนที่ 2 (PART 2)
27EB CDA727     CALL 27A7   ; ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1
                            ; ส่วนที่ 3 (PART 3)
27EE CDC627     CALL 27C6   ; ทดสอบ IC ประเภทที่ 3 SOCKET ที่ 1
                            ; ส่วนที่ 4 (PART 4)
27F1 C9         RET         ;
27F2 CD2022     CALL 2220   ; แสดงผล BINARY OUTPUT LOGIC ของภายในIC
                            ; ส่วนที่ 1 IC ประเภท A
27F5 CD4922     CALL 2249   ; ตรวจสอบว่าผลที่ได้ตรงกับ DATA หรือไม่,
                            ; SET FLAG G/B และ DELAY
```

```
27F8 23        INX  H      ; เพิ่มค่า HL = HL + 1
27F9 13        INX  D      ; เพิ่มค่า DE = DE + 1
27FA 78        MOV  A,B    ; {เอาค่าB เก็บไว้ที่A(B เป็นตัวเก็บINPUT LOGIC CONDITION)
27FB FE07      CPI  07     ; {เปรียบเทียบว่าค่าA=07 คือทำครบทุก CONDITION หรือยัง
27FD C9        RET         ;
27FE 00        NOP         ;
27FF 00        NOP         ;
```

โปรแกรมย่อย P 104

KEY INPUT SCAN ตรวจดูว่ากดเลขอะไรเข้ามาA REGISTER เป็น OUTPUT

```
2800 CD0D28     CALL 280D    ; เรียกโปรแกรม KEY SCAN
2803 47         MOV B,A      ; เก็บค่าผลที่ได้ไว้ที่ Bว่าเป็น KEY ใดหรือไม่มีการกด
2804 3AFF4F     LDA 4FFF     ; เอาค่าจาก LOCATION 4FFF ไว้ที่ A เป็น FLAG บอกว่ามีการกด KEY
2807 A7         ANA A        ; เมื่อ AND ระหว่าง A กับ A
2808 CA0028     JZ 2800      ; ผล ถ้าเป็น A=00 กับไปเริ่มต้นใหม่
280B 78         MOV A,B      ; ถ้ามีการกดแน่ เอาค่าที่ได้ว่าเป็น KEY ใดเก็บไว้ที่ A
280C C9         RET          ;
```

โปรแกรมย่อย P 105

KEY SCAN และส่งผลที่ได้ทาง A REGISTER

```
280D CD3128     CALL 2831    ; เรียกโปรแกรม SCAN 3 COLUME ($PB_4$, $PB_5$ $PB_6$#X'61')
2810 3C         INR A        ; SET ค่า A=00 (โดยเดิม A=FF เมื่อ A+1=FF+1=00)
2811 CA2C28     JZ 282C      ; เมื่อ A=00 ไม่มีการกดไปทำ STEP 282C
2814 CD6628     CALL 2866    ; ถ้ามีการกด KEY เข้ามา DELAY ประมาณ 10mS.
2817 CD3128     CALL 2831    ; กลับไปตรวจว่ากด KEYอีกว่ากดเข้ามาจริง
281A 47         MOV B,A      ; SET B=A เก็บค่าผลที่ได้ว่าเป็น KEY ใดที่ B
281B 3C         INR A        ; A = A+1
281C CA2C28     JZ 282C      ; ถ้ามีการกดKEY คือ A=00 กลับไปทำ STEP=282Cอีก
281F 3AFF4F     LDA 4FFF     ; ถ้ามีการกดKEY A≠00 แล้วเอาค่าที่อยู่ใน LOCATION 4FFF มาไว้ที่ A
2822 A7         ANA A        ; A AND ด้วย A
2823 C21428     JNZ 2814     ; ถ้าA≠00 ให้กลับไปตรวจใหม่ (ทำPROGRAM ที่ ADDRESS 2814 )
2826 3D         DCR A        ; ถ้าไม่ให้A=A-1 (ถ้า A=00, A=00-1=FF)
2827 32FF4F     STA 4FFF     ; STORGEค่า A ไว้ที่ ADDRESS 4FFF
282A 78         MOV A,B      ; SET ค่าA=B
282B C9         RET          ;
```

```
282C 06FF        MVI  B,FF    ; SET B=FF
282E C32728      JMP  2827    ; กลับไปทำ LOCATION 2827
```

โปรแกรมย่อย P 106

KEY SCAN (SCAN ๓ แถว แนวตั้งรับผลแนวนอน ๘ แถว)

```
2831 1600        MVI  D,00    ; CLEAR D=00 เป็นตัวนับจำนวนBIT ในแนวนอน 0-7 BIT
2833 42          MOV  B,D     ; SET B=00 เป็นค่าที่จะเอาไปบวกและอักษรกับ พวกตัวเลขใช้ในตอน LOOK UP TABLE
2834 3EEF        MVI  A,EF    ; ให้ A=EF เพื่อ SCAN แถวแรก BIT ที่ 4 ของ PORT B (PB4) ดู ckt รูป ๓.๓๖
2836 D361        OUT  61      ; ส่ง LOGIC '0' ไปในแถว BIT ที่ 4ของ PORT B
2838 DB60        IN   60      ; รับผลแนวนอนกลับทางPORT A ใช้8BIT(มี ๘แถวนอน)
283A EEFF        XRI  FF      ; ตรวจว่ามีการกด KEY เข้ามาหรือไม่ ใช้ EXCLUSIVE OR ด้วยEF กับ A
283C C25B28      JNZ  285B    ; ถ้าA=00 คือมีการกดKEY ก็ไปทำการตรวจว่ากดเลขอะไรเข้ามา
283E 0608        MVI  B,08    ; ถ้าไม่มีใครกด A=00 SET ค่าB=08 (ใช้ค่านี้ในตอน LOOK UP TABLE 8-F)
2841 3EDF        MVI  A,DF    ; ให้ A=DF เพื่อSCAN แถวที่ ๒BIT ที่๕ของPORTB
2843 D361        OUT  61      ; ส่งLOGIC '0'ไปในแถว BIT ที่5 ของ PORT B
2845 DB60        IN   60      ; รับผลแนวนอนกลับทาง PORT A
2847 EEFF        XRI  FF      ; ตรวจว่ามีการกดKEY เข้ามาหรือไม่ ใช้EXCLUSIVE OR ด้วยFF กับ A
2849 C25B28      JNZ  285B    ; ถ้า A≠00 คือมีการกด KEY ก็ไปทำการตรวจว่ากดเลขอะไรเข้ามา
284C 0610        MVI  B,10    ; ถ้าไม่มีใครกด A≠00 SET ค่า B=10 (ใช้ค่านี้ในตอน LOOK UP TABLE)
284E 3EBF        MVI  A,BF    ; ให้A=BF เพื่อSCAN แถวที่3BIT ที่ 6ของPORT B(PB6)
2850 D361        OUT  61      ; ส่งLOGIC'0' ไปในแถว3BITที่ 5 ของPORT B
2852 DB60        IN   60      ; รับผลแนวนอนกลับทาง PORT A
2854 EEFF        XRI  FF      ; ตรวจว่ามีการกด KEYเข้ามาหรือไม่EXCLUSIVE OR ด้วยFF กับ A
2856 C25B28      JNZ  285B    ; ถ้าA≠00 คือมีการกด KEY ก็ไปทำการตรวจว่ากดเลขอะไรเข้ามา
2859 3D          DCR  A       ; ถ้าไม่มีใครกดA≠00 SET ค่า A=A-1(A=00-1=FF)
285A C9          RET          ;
```

โปรแกรมย่อย P 1Ø7

ตรวจ กับ LOOK_UP_TABLE

```
285B ØF        RRC        ;ตรวจว่าเป็นBIT ไหน(แถวนอนไหน)เริ่มต้นจาก
                          ;BIT Ø ถึง 7 ตามลำดับ
285C DA6328    JC  2863   ;ถ้ามี CARRY ถ้าตรวจเจอ BIT นั้นไปทำการLOOK
                          ;UP TABLE ต่อไป
285F 14        INR D      ;ถ้าไม่มี CARRY ให้ D=D+1 เป็น COUNTER
                          ;บอกว่าเป็น BIT ใดเข้ามา
2860 C35B28    JMP 285B   ; กลับไปตรวจอีกว่าเป็น BIT ใดแน่
2863 7A        MOV A,D    ;เอาค่าที่ได้ว่าเป็น BIT ใด(แถวนอน)เข็บที่ A
2864 B0        ORA B      ;OR ค่า A กับค่าB ซึ่งเป็นค่าที่จะไป LOOK UP
                          ;TABLE ว่าเป็นเลขหรือเป็นอักษรอะไร
2865 C9        RET        ;
```

โปรแกรมย่อย P 1Ø8

หน่วยเวลา (DELAY TIME) 10mS.

```
2866 1648      MVI D,48   ;
2868 1E0C   ┌─→ MVI E,ØC   ;
286A 1D     │   DCR E      ;
286B C26A28 │   JNZ 286A   ;
286E 15     │   DCR D      ;
286F C26828 └── JNZ 2868   ;
2872 C9        RET        ;
```

โปรแกรมย่อย P 1Ø9

ทดสอบ IC เบอร์ 7401, 7439 (SOCKETแบบIC เบอร์ 7402) 2INPUT NAND GATEx4

```
2873 CDF425    CALL 25F4  ;SET ขาIC(I/P,O/P)แสดง READY, WAIT FAST/
                          ;SLOW TEST
2876 21200F    LXI H,ØF2Ø ;ENTRYชี้ TRUTH TABLE
2879 CDE124    CALL 24E1  ;CLEAR 7SEG.DISPLAY, CLEARE BINARY DISPLAY
287C CDE527    CALL 27E5  ;โปรแกรมทดสอบIC ประเภทที่ ๓,๑๔ ขา
287F C37328    JMP 2873   ;กลับตรวจสอบใหม่ถ้าต้องการ
```

โปรแกรมย่อย P 11Ø

SET ขา IC ประเภทที่ 4 SOCKET # 1 INPUT/OUTPUT=1, 2, 13/12, 3, 4, 5/6, 9, 10, 11/8

```
2882 3E60    MVI A,60    ; 8 6 | Ø 1 1 Ø Ø Ø Ø Ø | 9 5 4 3 2 1
2884 D3F0    OUT F0      ; SETขาIC ให้เป็นINPUT และOUTPUT ที่ DEVICE#FØ
2886 3E04    MVI A,Ø4    ; 12 | Ø Ø Ø Ø Ø 1 Ø Ø | 13 11 10
2888 D3F1    OUT F1      ; SERขาICให้เป็น INPUTและOUTPUTที่ DEVICE#F1
288A C9      RET         ;
```

```
288B CD8228  CALL 2882   ; SET ขา IC ว่าขาใดเป็นINPUT ขาใดเป็น OUTPUT เป็น IC ประเภทที่
288E CDDA20  CALL 20DA   ; เลือก SOCKET #1 จะแสดง LED/ แจ้ง READY ตรวจกดปุ่ม SLOW หรือไม่
2891 C9      RET         ;
```

โปรแกรมย่อย P 111

ลดค่า HL 1, 2, 3, 4, 5, 6, 7, 8, ครั้งแล้วแต่ความต้องการและ SET B=ØØ ใช้คอนกลับไปทั้งต้น DRIVE INPUT ใหม่ใน CONDITION

```
2892 2B      DCX H       ;
2893 2B      DCX H       ;
2894 2B      DCX H       ;
2895 2B      DCX H       ;
2896 2B      DCX H       ;
2897 2B      DCX H       ;
2898 2B      DCX H       ;
2899 2B      DCX H       ;
289A 0600    MVI B,ØØ    ; SET B=ØØ กลับไปตั้งต้น CONDITION แรกในการ STEST คือ ØØ
289C C9      RET         ;
```

โปรแกรมย่อย ---P 112

ทดสอบ IC ประเภทที่ 4 ใช้ SOCKET #1 ในส่วนที่ 1ZPART 1)ได้แก่ 7410, 12, 27และ 15

```
289D CD9E24    CALL 249E    ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART 1
28A0 0600      MVI B,00     ; CONDITION แรกเป็น00 ( B เป็นตัวเก็บ
                            ; LOGIC CONDITION)
28A2 CD0F22    CALL 220F    ; แสดงผล INPUT LOGIC ในรูปของเลขฐาน 16
                            ; ที่ 7-SEGMENT 2 DIGIT
28A5 78        MOV A,B      ; เอาค่า INPUT LOGIC CONDITION ใน B มา
                            ; เก็บไว้ที่ A เตรียมส่งออก
28A6 E603      ANI 03       ; MASK เอา 2 BIT หลังคือ BIT D0และ D1
28A8 D3E0      OUT E0       ; ส่ง LOGIC เข้าขา INPUT ขา 1และขา2 ของ IC
28AA 78        MOV A,B      ; เอา LOGIC CONDITIONที่SAVE ไว้ในB มาไว้ที่ A
28AB E604      ANI 04       ; MASK เอา BIT ที่ 2 คือ D2
28AD 07        RLC          ; เลื่อนข้อมูลให้ข้อมูล INPUT LOGIC ให้ตรงขา
                            ; 13 ของ IC
28AE D3E1      OUT E1       ; ส่ง INPUT LOGIC เข้าขา13 ของIC
28B0 DBD1      IN D1        ; รับ OUTPUT LOGIC ที่ของ IC ส่วนที่หนึ่ง
                            ; เข้าระบบ
28B2 E604      ANI 04       ; MASK เอา BIT ที่เป็น OUTPUT คือ BIT ที่ 2
                            ; ตรงกับขา 12 ของ IC
28B4 CDF227    CALL 27F2    ; แสดงผล BINARY OUTPUT LOGIC ทาง LED ตรวจ
                            ; สอบผลที่ตรวจว่าครบ CONDITION หรือไม่
28B7 CABE28    JZ 28BE      ; ตรวจดูแล้วถ้าทดสอบครบ CONDITION ก็ให้ไปทดสอบ
                            ; PART ที่ 2 ต่อไป
28BA 04        INR B        ; ถ้าไม่ครบเพิ่มค่า INPUT LOGIC CONDITION
                            ; อีก 1
28BB C3A228    JMP 28A2     ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 113

ทดสอบ IC ประเภทที่ 4 SOCKET # 1 ส่วนที่ 2 (PART 2)

```
28BE CD3629    CALL 2936    ; CLEAR ค่าINPUT LOGIC เก่าที่ค้างอยู่ให้เป็น 00
                            ; และ SET 'EA' ปิดท้ายใน MEMORY REPORT
28C1 CDA324    CALL 24A3    ; บอกว่าเป็นส่วนที่ 2 ทาง 7-SEGMENT
28C4 CD9228    CALL 2892    ; ถอยPOINTER ที่ชี้ TRUTH TABLE กลับไปตั้งต้น
                            ; ใหม่ SET INPUT LOGIC=00 (B=00)
28C7 CD0F22    CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED แบบ BINARY
28CA 78        MOV A,B      ; เอา INPUT LOGIC กรณีต่าง ๆ ไว้ที่A เตรียมขับ
                            ; เข้าขา INPUT ของ IC
28CB 07        RLC          ; เลื่อน INPUT LOGICให้ตรงกับขาINPUT ของ IC
28CC 07        RLC          ;
```

```
28CD CD4422      CALL 2244    ; ส่ง INPUT LOGIC เข้าขา INPUT ของ IC ส่วนที่ 2 แล้วรับผลจากขา 6 กลับ
28D0 E620        ANI  20      ; MASK เฉพาะ BIT ที่ 5 ซึ่งตรงกับขา OUTPUT ขา6 ของ IC ส่วนที่ 2
28D2 0F          RRC          ;
28D3 0F          RRC          ; เลื่อนค่า OUTPUT LOGIC ที่ได้ให้ตรงกับ TRUTH-
28D4 0F          RRC          ; TABLE
28D5 CDF227      CALL 27F2    ; แสดงผล OUTPUT ที่ได้ในลักษณะ BINARY ทาง LED ตรวจสอบว่า GOOD/BAD และทำครบทุกกรณี ?
28D8 CADF28      JZ   28DF    ; ถ้าทำครบทุก CONDITION แล้วไปทดสอบ PART ที่ 3 ต่อไป
28DB 04          INR  B       ; ถ้าไม่ครบเพิ่มค่า INPUT LOGIC อีก 1
28DC C3C728      JMP  28C7    ; กลับไปทดสอบ CONDITION ต่อไป

                 โปรแกรมย่อย P 114

ทดสอบ IC ประเภทที่ 4 SOCKET # 1 ส่วนที่ 3 (PART 3)
28DF CD4A24      CALL 244A    ; SET ค่า 'EA' ใน MEMORY REPORT หลังผลลัพท์ที่เก็บไว้
28E2 CDA824      CALL 24A8    ; บอกว่าเป็นส่วนที่ 3 ทาง 7-SEGMENT
28E5 CD9228      CALL 2892    ; ถอย POINTER ที่ชี้ TRUTH TABLE กลับไปตั้งต้นใหม่ SET INPUT LOGIC=ØØ (B=ØØ)
28E8 CD0F22      CALL 220F    ; แสดงผล INPUT LOGIC ทาง LED แบบ BINARY
28EB 78          MOV  A,B     ; เอา INPUT LOGIC กรณีต่าง ๆ ไว้ที่ A เตรียมขับเข้าขา INPUT ของ IC
28EC 0F          RRC          ; เลื่อน INPUT LOGIC ให้ตรงกับขา IC
28ED 4F          MOV  C,A     ; SAVE A ไว้ที่ C
28EE E680        ANI  80      ; MASK BIT ที่ 7 ($D_7$) ซึ่งตรงกับขา 9
28F0 D3E0        OUT  E0      ; ขับ INPUT LOGIC เข้าขา9 ของ IC LATCH ไว้
28F2 79          MOV  A,C     ; เอาค่า INPUT LOGIC ที่ SAVE ไว้กลับมาที่ A
28F3 E603        ANI  03      ; MASK ของ BIT ที่ 0 และที่ 1 ($D_0$ และ $D_1$) ซึ่งตรงกับขา INPUT ของ IC ขา10 และ11
28F5 D3E1        OUT  E1      ; ขับ INPUT LOGIC เข้าขา IC ทั้ง ๒ ขา คือขา 10 และขา 11
28F7 DBD0        IN   D0      ; รับผลจาก OUTPUT กลับ
28F9 E640        ANI  40      ; MASK เอา BIT ที่ 6 มาซึ่งตรงกับ OUTPUT LOGIC ที่ได้จากขา 8 ของ IC
```

```
28FB 0F          RRC          ;
28FC 0F          RRC          ;  เลื่อน OUTPUT LOGIC ที่ได้ให้ตรงกับ
28FD 0F          RRC          ;  TRUTH TABLE
28FE 0F          RRC          ;
28FF CDF227      CALL 27F2    ; แสดงผล OUTPUT ที่ได้ในลักษณะ=BINARY ทางLED
                              ; ตรวจสอบว่าGOOD/BAD และทำครบทุกกรณี ?
2902 CA5423      JZ   2354    ; ถ้าทำครบทุก CONDITION ก็ไปสรุปผลทั้งหมดอีกที่ว่า
                              ; GOOD และ BAD-PART ใด
2905 04          INR  B       ; ถ้าไม่ครบเพิ่มค่า INPUT LOGIC อีก 1
2906 C3E828      JMP  28E8    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 115

ทดสอบ IC ประเภทที่ 4 SOCKET #1สำหรับ IC เบอร์ 7410 และ 12

```
2909 CD8B28      CALL 288B    ; SET ขาIC ประเภทที่ แสดงเลือก SOCKET
                              ; ที่ 1 ตรวจสอบ KEY ต่าง ๆ
290C 2140 0F     LXI  H,0F40  ; SET POINTERชี้ TRUTH TABLE ของ IC
                              ; เบอร์ 7410, 12
290F CDE124      CALL 24E1    ; CLEAR 7-SEGMENT และ CLEAR I/O BINARY
                              ; DISPLAY
2912 CD9D28      CALL 289D    ; ทดสอบ IC ประเภทที่4 เริ่ม PART ที่ 1 ก่อน
2915 C30929      JMP  2909    ; กลับไปตรวจว่าต้องการจะ TEST ต่อหรือไม่
```

โปรแกรมย่อย P 116

ทดสอบ IC ประเภทที่ 4 SOCKET # 1สำหรับ IC เบอร์ 7427

```
2918 CD8B28      CALL 288B    ; SET ขา IC ประเภทที่ 4 แสดงเลือกSOCKET #
                              ; ที่ 1,ตรวจสอบ KEY ต่าง ๆ
291B 2148 0F     LXI  H,0F48  ; SET POINTER ชี้ TRUTH TABLE ของ IC
                              ; เบอร์ 7427
291E CDE124      CALL 24E1    ; CLEAR 7-SEGMENT และ CLEAR I/O BINARY
                              ; DISPLAY
2921 CD9D28      CALL 289D    ; ทดสอบ IC ประเภทที่ 4 เริ่ม PART ที่ 1 ก่อน
2924 C31829      JMP  2918    ; กลับไปตรวจว่าต้องการจะ TEST ต่อหรือไม่
```

โปรแกรมย่อย P 117

ทดสอบ ประเภทที่ 4 SOCKET # 1 สำหรับ IC เบอร์ 7411, 15

```
2927 CD8B28    CALL 288B    ;SET ขา IC ประเภทที่4 แสดงเลือกSOCKET # ที่ 1,ตรวจสอบ KEY ต่าง ๆ
292A 2150ØF    LXI  H,ØF5Ø  ;SET POINTER ชี้ TRUTH TABLE ของ IC เบอร์ 7411, 15
292D CDE124    CALL 24E1    ;CLEAR 7-SEGMENT และ CLEAR I/O BINARY DISPLAY
2930 CD9D28    CALL 289D    ; ทดสอบ IC ประเภทที่4 เริ่มPART ที่ 1 ก่อน
2933 C32729    JMP  2927    ; กลับไปตรวจว่าต้องการจะ TEST ต่อหรือไม่
```

โปรแกรมย่อย P 118

SET ค่า 'EA' ปิดท้ายผลการทดสอบและ CLEAR LOGIC INPUT ที่ค้างอยู่ที่ PORT

```
2936 CD4A24    CALL 244A    ; SET ค่า'EA' ปิดท้ายผลทดสอบ
2939 CDFC22    CALL 22FC    ; CLEAR LOGIC INPUT ที่ค้างอยู่ที่ PORT
293C C9        RET          ;
```

โปรแกรมย่อย P 119

SET ขา IC 14ขา, ขา INPUT/ ขา OUTPUT=1, 2, 3, 4, 5/6; 9, 10, 11, 12, 13/8

ประเภทที่ 5 SOCKET # 1

```
293D 3E60      MVI  A,60    ; [Ø|1|1|Ø|Ø|Ø|Ø|Ø] ขา๗ไม่ใช้(NO CONNECT)
293F D3F0      OUT  F0      ;ส่ง OUTPUT ออกไปทาง PORT A # FØ SETขา INPUT/OUTPUT
2941 3E00      MVI  A,00    ; [Ø|Ø|Ø|Ø|Ø|Ø|Ø|Ø] ขา ๑๑ไม่ใช้ (NO CONNECT)
2943 D3F1      OUT  F1      ;ส่ง OUTPUT ออกไปทาง PORT B#F1 SETขา INPUT/OUTPUT
2945 CDDA20    CALL 20DA    ;เลือก SOCKET # 1 LED ติด, แจ้ง READY, ตรวจว่า SLOW หรือ TEST
2948 C9        RET          ;
```

โปรแกรมย่อย P 12Ø

แสดง OUTPUT LOGIC ตรวจสอบผลว่า GOOD หรือ BAD, ตรวจทำถึง CONDITION สุดท้ายหรือยัง

```
2949 CD2022    CALL 2220    ;แสดงผล BINARY OUTPUT LOGIC แบบตัวเลขฐาน 16
294C CD4922    CALL 2249    ;ตรวจว่าผลที่ได้ตรงกับข้อมูลหรือไม่และSET GOOD หรือ BAD แล้ว DELAY 1 SEC
294F 13        INX  D       ;เพิ่มค่า POINTER ชี้ผลของการทดสอบใน MEMORY อีกหนึ่ง
```

```
2950 79          MOV  A,B    ;เอาค่า INPUT LOGIC CONDITION จาก Bไปเก็บที่ A
2951 FE0E        CPI  0E     ;ตรวจว่าทำถึงก่อน INPUT LOGIC CONDITION สุดท้ายหรือยัง
2953 C25729      JNZ  2957   ;ถ้าไม่เท่าไปตรวจว่าทำถึง CONDITION สุดท้ายหรือยัง
2956 23          INX  H      ;เพิ่มค่าPOINTER ชี้ TRUTH TABLE อีกหนึ่ง
2957 FE0F        CPI  0F     ;ตรวจสอบว่าทำครบทุก CONDITION หรือยัง
2959 C9          RET         ;
```

โปรแกรมย่อย P 121

ทดสอบ IC ประเภทที่ 5 SOCKET #1ส่วนที่ 1 (PART 1)

```
295A CD9E24      CALL 249E   ; แสดงผลทาง7-SEGMENT ว่าเป็น PART 1
295D 0600        MVI  B,00   ;CLEAR B=00(SET INPUT LOGIC เป็น 00 CONDITION แรก)
295F CD0F22      CALL 220F   ;แสดงผล INPUT LOGIC ลักษณะเลขฐาน 16 และเลขฐาน 2
2962 78          MOV  A,B    ; เตรียม INPUT LOGIC ที่จะส่งออกไว้ที่ A
2963 E603        ANI  03     ;MASK เอาเฉพาะ BIT 1และ BIT 0 ($D_1$ และ $D_0$ ตามลำดับ)
2965 4F          MOV  C,A    ; SAVE INPUT LOGIC ที่ MASK ไว้ที่ A ก่อน
2966 78          MOV  A,B    ; เอา INPUT LOGIC มาไว้ที่ A อีก
2967 07          RLC         ;เลื่อน INPUT LOGIC 2 BIT หน้าคือ BIT 3 และ BIT 2 ให้ตรงกับขา ICคือขา 5และขา 4ตามลำดับ
2968 E618        ANI  18     ;MASK เอาเฉพาะBIT 5 และ BIT 4 ($D_4$ และ $D_3$ ตามลำดับ)
296A B1          ORA  C      ;OR เพื่อรวม INPUT LOGIC DATA ที่สมบูรณ์ที่ตรงกับขา INPUT ของ IC พอดี
296B CD4422      CALL 2244   ;DRIVE LOGIC เข้าขาของ IC แล้วรับผลที่ได้กลับเข้าระบบ
296E E620        ANI  20     ;MASK เอาเฉพาะBIT ที่ตรงOUTPUT ของส่วนที่กำลังทดสอบอยู่
2970 0F          RRC         ;เลื่อนข้อมูล OUTPUT ที่ได้ให้ตรงกับ TRUTH
2971 0F          RRC         ; TABLE โดย ROTATE ทางขวา ๓ ครั้ง
2972 0F          RRC         ;
2973 CD4929      CALL 2949   ;แสดงผล OUTPUT LOGIC แบบ HEX ตรวจว่าทำครบทุก CONDITION หรือยัง
2976 CA7D29      JZ   297D   ; ถ้าทำครบไปทดสอบ PART ต่อไป
2979 04          INR  B      ; ถ้าไม่ครบเพิ่มค่า INPUT LOGIC อีก 1
297A C35F29      JMP  295F   ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 122

ทดสอบ IC ประเภทที่ 5 SOCKET # 1 ส่วนที่ 2 (PART 2)

```
297D CD4A24     CALL 244A   ; SET ค่า 'EA' บิตท้ายผลการทดสอบแต่ละ PART
                            แสดงผลทาง 7-SEGMENT ว่าเป็น PART 2
2980 CDA324     CALL 24A3   ;
2983 2B         DCX  H      ; เลื่อนPOINTER ชี้ OUTPUT LOGIC TABLE
                            ค่าถัดไป
2984 0600       MVI  B,00   ; START INPUT LOGIC CONDITION = 00
2986 CD0F22     CALL 220F   ; แสดงผล INPUT LOGIC ลักษณะเลขฐาน 16
                            และเลขฐาน 2
2989 78         MOV  A,B    ; เตรียม INPUT LOGIC ที่จะส่งออกไว้ที่ A
298A 0F         RRC         ; เลื่อน INPUT LOGIC ให้ตรงกับขา IC
298B 4F         MOV  C,A    ; SAVE ค่า A ไว้ที่ C
298C E680       ANI  80     ; MASK เอา BIT 7 ซึ่งตรงกับขา 9ของ IC
298E D3E0       OUT  E0     ; ส่ง INPUT LOGIC เข้าขา 9 ของ IC
                            ( LACTH ไว้)
2990 79         MOV  A,C    ; เอาค่า SAVE ไว้ใน C มาไว้ที่ A
2991 E601       ANI  01     ; MASKเอาBIT 0 เพื่อตรงกับขา10 ของ IC
2993 D5         PUSH D      ; SAVE ค่า DE Reg. ใน MEMOR
2994 57         MOV  D,A    ; SAVE ค่า INPUT LOGIC สำหรับขา10 ใน D
2995 78         MOV  A,B    ; เอาค่า INPUT LOGIC จากB ไว้ที่ A
2996 E60C       ANI  0C     ; MASK BIT 2 และ BIT 3 ตรงกับขา 12
                            และขา 13 ของ IC
2998 B2         ORA  D      ; รวมINPUT LOGIC ที่จะขับเข้าขา10, 12 และ
                            13 เข้าด้วยกัน
2999 D1         POP  D      ; เอาค่า DE ใน MEMORY คืนที่เดิม
299A D3E1       OUT  E1     ; ส่ง INPUT LOGIC เข้าขา10, 12 และ 13
                            ของ IC
299C DBD0       IN   D0     ; รับผลจาก OUTPUT ของ IC เข้าระบบ
299E E640       ANI  40     ; MASK เอาเฉพาะBIT 6 ที่เป็น OUTPUT
29A0 0F         RRC         ;
29A1 0F         RRC         ; เลื่อนค่าOUTPUT ให้ตรงกับ TABLE
29A2 0F         RRC         ;
29A3 0F         RRC         ;
29A4 CD4929     CALL 2949   ; แสดง OUTPUT LOGIC ตรวจสอบผลว่า GOOD หรือ
                            BAD,ตรวจทำครบทุก CONDITION หรือยัง
29A7 CA5423     JZ   2354   ; ถ้าทำครบทุก CONDITION หรือยัง ถ้าครบแสดง
                            ผลการตรวจสอบ
29AA 04         INR  B      ; เพิ่มค่า INPUT LOGIC อีกหนึ่ง
29AB C38629     JMP  2986   ; กลับทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 123

ทดสอบ IC เบอร์ 7420 (4 INPUT NAND GATE),7422,40,140, 13, 74C02

```
29AE CD3D29     CALL 293D    ; {SET ขา IC ตาม IC ประเภทที่ 5 และตรวจ
                              ; {KEY ว่าจะ CLEAR ,CONT, SLOW
29B1 21480F     LXI H,0F48   ; {POINTER ชี้ OUTPUT TRUTH TABLE ของ IC
                              ; {เบอร์ 7420
29B4 CDE124     CALL 24E1    ; {CLEAR 7-SEGMENT และ OUTPUT BINARY
                              ; {DISPLAY
29B7 CD5A29     CALL 295A    ; ทดสอบ IC ประเภทที่ 5
29BA C3AE29     JMP 29AE     ; {กลับไปทดสอบว่าจะทดสอบต่อหรือ CLEAR หรือ
                              ; {ทดสอบแบบ SLOW
```

โปรแกรมย่อย P 124

ทดสอบ IC เบอร์ 7421 (4 INPUT AND GATE)

```
29BD CD3D29     CALL 293D    ; {SET ขา IC ตาม IC ประเภทที่ 5 และตรวจ
                              ; {KEY ว่าจะ CLEAR, CONT, SLOW
29C0 212C0F     LXI H,0F2C   ; {POINTER ชี้ OUTPUT TRUTH TABLE ของ IC
                              ; {เบอร์ 7421 (4 INPUT AND GATE)
29C3 CDE124     CALL 24E1    ; CLEAR 7-SEGMENTและ OUTPUT BINARY DISPLAY
29C6 CD5A29     CALL 295A    ; ทดสอบ IC ประเภทที่ 5
29C9 C3BC29     JMP 29BC     ; {กลับไปทดสอบว่าจะทดสอบต่อหรือ CLEAR หรือทดสอบ
                              ; {แบบ SLOW
```

โปรแกรมย่อย P 125

ทดสอบ IC ประเภทที่ 7 SOCKET # 1 ทุก ๆ PART โดยใช้ MODUIE จากโปรแกรมอื่น

```
29CC CD7122     CALL 2271    ; ทดสอบ IC ประเภทที่ 1 SOCKET # 1 PART 1
29CF CD8827     CALL 2788    ; ทดสอบ IC ประเภทที่ 3 SOCKET # 1 PART 2
29D2 CDA727     CALL 27A7    ; ทดสอบ IC ประเภทที่ 3 SOCKET # 1 PART 3
29D5 CDD822     CALL 22D8    ; ทดสอบ IC ประเภทที่ 1 SOCKET # 1 PART 4
29D8 C9         RET          ;
```

โปรแกรมย่อย P 126

SET ขาIC ประเภทที่ 7 ขา INPUT/OUTPUT = 1,2/3; 5,6/4; 8,9/10 และ 12,13/11

```
29D9 3E0C       MVI  A,0C    ; SET ขา 1,2/3 และ 5,6/4 8,9 9INPUT)
29DB D3F0       OUT  F0      ; ส่งข้อมูลไป SET PORT A # F 0
29DD 3E03       MVI  A,03    ; SET ขา 10(OUTPUT) และ 12,13/11
29DF D3F1       OUT  E1      ; ส่งข้อมูลไป SET PORT B # F1
29E1 CDDA20     CALL 20DA    ; เลือก SOCKET # LED ดิด, แจ้ง READY ตรวจ KEY ว่า SLOW TEST
29E4 C9         RET          ;
```

โปรแกรมย่อย P 127

ทดสอบ IC เบอร์ 74266 (QUAD EXCLUSIVE - NOR GATE)

```
29E5 CDD929     CALL 29D9    ; SET ขา IC ประเภทที่ ๗
29E8 21320F     LXI  H,0F32  ; POINTER ชี้ LOGIC OUTPUT TABLE
29EB CDE124     CALL 24E1    ; CLEAR 7 SEGMENT HEX/BINARY DISPLAY
29EE CDCC29     CALL 29CC    ; ทดสอบ IC ประเภทที่ ๗
29F1 C3E529     JMP  29E5    ; กลับไปทดสอบว่าจะทดสอบต่อหรือ CLEAR หรือ ทดสอบแบบ SLOW
```

โปรแกรมย่อย P 128

SET ขา IC ประเภทที่ 6, SOCKET # 1

```
29F4 3E40       MVI  A,40    ; SET ขา 1, 2, 3, 4, 5, 6 เป็น INPUT/ ขา 8 เป็น OUTPUT
29F6 D3F0       OUT  F0      ; SET PIN CONTROLLERตามคำสั่งนั้น
29F8 3E00       MVI  A,00    ; SET ขา 11, 12 เป็น INPUT นอกนั้น DON't CARE
29FA D3F1       OUT  F1      ; SET PIN CONTROLLER ตามคำสั่งนั้น
29FC CDDA20     CALL 20DA    ; เลือก SOCKET # 1 LED ดิด แสดงREADY, ตรวจสอบ KEY
29FF C9         RET          ;
```

โปรแกรมย่อย P 129

แสดง OUTPUT LOGIC ตรวจสอบผลว่า GOOD หรือ BAD ตรวจทำถึง CONDITION สุดท้ายหรือยัง

```
2A00 CD2022     CALL 2220   ; แสดงผล BINARY OUTPUT LOGIC
2A03 CD4922     CALL 2249   ; ตรวจสอบว่าผลที่ได้ตรงกับ DATA หรือไม่
                            ; SET GOOD/BAD
2A06 13         INX  D      ; เพิ่มค่า POINTER ชี้ผลของการทดสอบใน MEMORY
                            ; อีกหนึ่ง
2A07 78         MOV  A,B    ; นำค่า INPUT LOGIC CONDITION ไว้ที่ A
2A08 FEFE       CPI  FE     ; ตรวจสอบว่าทดสอบถึง CONDITION ก่อนสุดท้าย
                            ; หรือยัง
2A0A C20E2A     JNZ  2A0E   ; ถ้ายังไปตรวจสอบว่าถึง CONDITION สุดท้าย
                            ; หรือยัง
2A0D 23         INX  H      ; เพิ่มค่า POINTER ชี้ TRUTH TABLE อีกหนึ่ง
2A0E FEFF       CPI  FF     ; ตรวจสอบว่าถึง CONDITION สุดท้ายหรือยัง
2A10 C9         RET         ;
```

โปรแกรมย่อย P 130

ทดสอบ IC ประเภทที่ 6 SOCKET ที่ 1 ส่วนที่ 1(PART 1)ในที่นี้มีส่วนเดียว

```
2A11 CD9E24     CALL 249E   ; แสดงผลทาง 7-SEGMENT ว่าเป็น PART 1
2A14 0600       MVI  B,00   ; SET START INPUT LOGIC CONDITION
                            ; = 00 เก็บที่ B
2A16 CD0F22     CALL 220F   ; แสดงผลทางเลขฐาน 16 และ BINARY ของ
                            ; INPUT LOGIC CONDITION
2A19 78         MOV  A,B    ; เตรียมส่ง INPUT LOGIC ไว้ที่ A
2A1A E63F       ANI  3F     ; MASK เอา BIT 0, 1, 2, 3, 4,และ 5
                            ; ตรงกับขา 1, 2, 3, 4, 5 และ 6
2A1C D3E0       OUT  E0     ; ส่ง INPUT LOGIC เข้าขา IC ดังกล่าวแล้ว
2A1E 78         MOV  A,B    ; ส่วนขาที่เหลือเอาค่า INPUT LOGIC
                            ; มาทำอีก
2A1F E6C0       ANI  C0     ; โดยMASK เอา BIT 6 และ 7 ตรงกับขา 11
                            ; และ 12 ตามลำดับ
2A21 07         RLC         ;
2A22 07         RLC         ; เลื่อน INPUT LOGIC ให้ตรงกับขา 11 และ12
2A23 07         RLC         ;
2A24 D3E1       OUT  E1     ; ส่ง INPUT LOGICเข้าขา IC ที่นำมาทดสอบ
2A26 DBD0       IN   D0     ; รับผลที่ได้กลับเข้าระบบ
2A28 E640       ANI  40     ; MASKเอา BIT 6ซึ่งเป็นค่า OUTPUT LOGIC
2A2A 0F         RRC         ; เลื่อน OUTPUT LOGIC ให้ตรงกับ TABLE
```

```
2A2B ØF          RRC          ;
2A2C ØF          RRC          ;
2A2D ØF          RRC          ;
2A2E CDØØ2A      CALL 2AØØ    ; แสดง OUTPUT LOGIC, ตรวจสอบผลว่าGOOD หรือ BAD และตรวจว่าครบทุกCONDITION หรือยัง
2A31 CA5423      JZ   2354    ; ถ้าทำครบ CONDITIONรายงานผลการทดสอบทั้งหมด
2A34 Ø4          INR  B       ; เพิ่มค่า INPUT LOGIC CONDITION อีก 1
2A35 C3162A      JMP  2A16    ; กลับไปทดสอบ CONDITION ต่อไป
```

โปรแกรมย่อย P 131

ทดสอบ IC ประเภทที่ 6 SOCKET ที่ 1สำหรับ IC เบอร์ 7430

```
2A38 CDF429      CALL 29F4    ; SET ขา IC ประเภทที่ 6, SOCKET ที่ 1
2A3B 21480F      LXI  H,ØF48  ; POINTER ชี้ LOGIC OUTPUT TABLE
2A3E CDE124      CALL 24E1    ; CLEAR 7-SEGMENTและI/P, O/P BINARY DISPLAY
2A41 CD112A      CALL 2A11    ; ทดสอบ IC ประเภทที่ 6
2A44 C3382A      JMP  2A38    ; กลับไปตรวจว่าต้องการจะ TEST ต่อหรือไม่
```

((โปรแกรมย่อย P 132

DISPLAY OUTPUT 74164 และจัด OUTPUT ให้เรียงลำดับกันจาก QA - QH

```
2A47 DBDØ        IN   DØ      ; รับผลจาก OUTPUT ของ IC 74164 (QA, QB, QCและ QD)
2A49 E63C        ANI  3C      ; MASK เอาเฉพาะ Q_A, Q_B, Q_C และ Q_D
2A4B ØF          RRC          ; เลื่อนให้ตรงกับ BIT 0, 1, 2และ 3 ตามลำดับ
2A4C ØF          RRC          ;
2A4D C5          PUSH B       ; SAVE BC ไว้
2A4E 4F          MOV  C,A     ; SAVE OUTPUT ที่ได้ไว้ที่ C(Q_A, Q_B, Q_C และ Q_H)
2A4F DBD1        IN   D1      ; รับผลจาก OUTPUT ของ IC 74164 (Q_E, Q_F Q_Gและ Q_H)
2A51 E60F        ANI  ØF      ; MASK เอา 4 BIT หลัง (Q_E, Q_F, Q_Gและ Q_H)
2A53 CD1326      CALL 2613    ; เลื่อนให้ตรงกับ BIT 4, 5, 6และ 7 ตามลำดับ
```

```
2A56 B1          ORA  C      ;รวม OUTPUT เข้าด้วยกัน โดยที่ Q_A-Q_H
                              ;ตรงกับ BIT 0-7 ตามลำดับ
                              ;คือค่าที่ SAVE ไว้ใน MEMORY ให้ BC
2A57 C1          POP  B      ;
2A58 D3B0        OUT  B0     ;ส่งออกDISPLAY ทาง LED BINARY DISPLAY
2A5A C9          RET         ;
```

```
2A5B CDF920      CALL 20F9   ;DECODE จาก 4 BIT(D3-D0) ฐาน ๒ เป็น
                             ;7-SEGMENT
2A5E D37F        OUT  7F     ;DISPLAY ออก 7-SEGMENT # 7F
2A60 CD7420      CALL 2074   ;DECODE จาก 4 BIT(D7-D0) ฐาน ๒ เป็น
                             ;7-SEGMENT
2A63 D37E        OUT  7E     ;DISPLAY ออก 7-SEGMENT # 7E
2A65 C9          RET         ;
```

โปรแกรมย่อย P 133

SET ขา IC ประเภทที่ 8SOCKET # 1 IC เบอร์ 74164

```
2A66 3E3C        MVI  A,3C   ;SET ขา 1, 2, 8 และ9 เป็น INPUTและขา
                             ;3, 4, 5, 6 เป็น OUTPUT
2A68 D3F0        OUT  F0     ;SET PIN CONTROLLER PORT A # FØ
2A6A 3E0F        MVI  A,0F   ;SETขา 10, 11, 12และ 13 เป็น OUTPUT
2A6C D3F1        OUT  F1     ;SET PIN CONTROLLER PORT B # F1
2A6E CDDA20      CALL 20DA   ;เลือก SOCKET #1 LED ติดแจ้งREADY ตรวจ
                             ;FAST/SLOW TEST
2A71 C9          RET         ;
```

โปรแกรมย่อย P 134

CLEAR (CLR) เตรียมการก่อน DISPLAY

```
2A72 E680        ANI  80     ;MASKเอา BIT 7
2A74 07          RLC         ;
2A75 07          RLC         ;  เลื่อนให้ตรงกับ BINARY DISPLAY
```

```
2A76 07          RLC          ;
2A77 32094E      STA  4E09    ;SAVE INPUT 'CLR'ไว้ที่ 4E09
2A7A C9          RET          ;

        โปรแกรมย่อย  P 135
    CLOCK (CK) เตรียมการก่อน DISPLAY
2A7B E640        ANI  40      ;MASK เอา BIT 6
2A7D 0F          RRC          ;
2A7E 0F          RRC          ; เลื่อนให้ตรงกับ BINARY DISPLAY
2A7F 0F          RRC          ;
2A80 320A4E      STA  4E0A    ;SAVE INPUT 'CK'ไว้ที่ 4E0A
2A83 C9          RET          ;

        โปรแกรมย่อย  P 136
    DISPLAY INPUT AและB
2A84 E603        ANI  03      ;MASK เอา BIT 0 กับ BIT 1 ซึ่งเป็น A และ B ที่INPUT
2A86 D3C0        OUT  C0      ;ส่ง INPUT ที่ได้ออกทาง LED (BINARY DISPLAY)
2A88 C9          RET          ;

        โปรแกรมย่อย  P 137
    ทดสอบ IC ประเภทที่ 8
2A89 D3E0        OUT  E0      ;ส่ง INPUT LOGIC เข้าขา IC ที่นำมาทดสอบ
2A8B CD0B24      CALL 240B    ;CLEAR MEMORY ที่แสดงผลส่วนที่เสียและ CLEAR 7-SEGMENT # 78 -7F
2A8E C5          PUSH B       ;SAVE BC ไว้ใน MEMORY
2A8F 47          MOV  B,A     ;นำค่า A SAVE ไว้ที่ B (INPUT LOGIC)
2A90 CD722A      CALL 2A72    ;SET ค่า CLR FLAG ใน MEMORY (BIT 7)
2A93 78          MOV  A,B     ;นำค่า B ไว้ที่ A (INPUT LOGIC)
```

```
2A94 CD7B2A     CALL 2A7B    ; SETค่า CK FLAGใน MEMORY (BIT 6)
2A97 CDDF2A     CALL 2ADF    ; เตรียมสัญญาณ CLR และ CK
2A9A D3C2       OUT C2       ; DISPLAY CLR และ CK
2A9C 78         MOV A,B      ; นำค่า B ไว้ที่ A (INPUT LOGIC)
2A9D CD842A     CALL 2A84    ; DISPLAY INPUT A และ B
2AA0 CD472A     CALL 2A47    ; DISPLAY ผลที่ได้จาก OUTPUT Q_A-Q_H
2AA3 CDE420     CALL 20E4    ; {ตรวจ FLAG FAST/SLOW TEST ถ้า SLOW จะหน่วงเวลาให้เลย
2AA6 C1         POP B        ; เอาค่า BC เก่าที่ SAVE ไว้คืน
2AA7 C9         RET          ;

                โปรแกรม P 138
เริ่มต้น TEST IC ประเภทที่ 8 SOCKET # 1
                STEPแรกCLEAR 8 BIT SHIFT REG. INPUT CLR=Ø
2AA8 3E00       MVI A,00     ;
2AAA CD892A     CALL 2A89    ; CLEAR SHIFT REGISTER
2AAD CD492Z     CALL 2249    ; ตรวจสอบผลที่ได้ว่าตรงกับข้อมูลหรือไม่
2AB0 23         INX H        ; {เลื่อน POINTER ไปดูข้อมูลที่เป็น INPUT LOGIC ถัดไป
2AB1 13         INX D        ; เลื่อนPOINTER ที่เก็บผลลัพท์
2AB2 0600       MVI B,00     ; {CLEAR B=ØØ, Bเป็นINPUT A และ B ตอนนี้ CONDITION ØØ
2AB4 0E08       MVI C,08     ; กำหนดจำนวน CK ๘ ครั้ง
2AB6 3E80       MVI A,80     ; {A เป็นCLR และ CR, CONDITIONแรกCK=Ø, CLR
2AB8 B0         ORA B        ; {รวมข้อมูลที่จะขับเข้าขา IC เข้าด้วยกัน CLK, CK, A และ B (INPUT LOGIC)
2AB9 CD892A     CALL 2A89    ; ทดสอบ IC ประเภทที่ 8
2ABC 3EC0       MVI A,C0     ; ทำ CK ให้เป็น MASK = '1'
2ABE B0         ORA B        ; {รวมข้อมูลที่ขับเข้าขา IC เข้าด้วยกัน CLK, CK, A และ B
2ABF CD892A     CALL 2A89    ; ทดสอบ IC ประเภทที่ 8
2AC2 0D         DCR C        ; ลดค่า C อีก 1 (C=C-1)
                ถ้าไม่ครบ CLOCK 8 ลูก ให้กลับไปทดสอบต่อ
2AC3 C2B62A     JNZ 2AB6     ;
```

```
2AC6 CD5922     CALL 2259    ; ถ้าทำครบ CLOCK 8 ลูกแล้วตรวจสอบผลที่ได้กับ
                             ; TABLE และทำครบทุก CONDITION หรือยัง
2AC9 CA5423     JZ   2354    ; ถ้าทำครบรายงานผลการทดสอบทั้งหมด
2ACC 04         INR  B       ; เพิ่มค่า B อีก 1 เพื่อทดสอบ CONDITION ต่อไป
2ACD C3B42A     JMP  2AB4    ; กลับไปทดสอบ CONDITION ต่อไป

                    โปรแกรมย่อย  P 139
        ทดสอบ IC เบอร์ 74164 เป็น IC ประเภทที่ 8ใช้ SOCKET # 1
2AD0 CD662A     CALL 2A66    ; SET ขา IC ประเภทที่ 8
2AD3 21580F     LXI  H,0F58  ; POINTER ชี้ OUTPUT LOGIC TABLE
2AD6 CDE124     CALL 24E1    ; CLEAR 7-SEGMENT INPUT/OUTPUT BINARY
2AD9 CDA82A     CALL 2AA8    ; ทดสอบ IC ประเภทที่ 8
2ADC C3D02A     JMP  2AD0    ; กลับไปตรวจว่าจะทดสอบอีกหรือไม่ CLEAR หรือ
                             ; SLOW TEST

                    โปรแกรมย่อย  P 140
          เตรียมสัญญาณ CLR และ CK เพื่อจะ DISPLAY
2ADF E5         PUSH H       ;
2AE0 D5         PUSH D       ; SAVE ค่าใน HL, DE และ BC
2AE1 C5         PUSH B       ;
2AE2 21094E     LXI  H,4E09  ; SET POINTER ไปที่ CLR FLAG
2AE5 0000       NOP          ;
2AE7 7E         MOV  A,M     ; เอาค่าใน CLR FLAG ไว้ที่ A
2AE8 23         INX  H       ; เลื่อนค่าPOINTER ไปชี้ที่ CK FLAG
2AE9 4E         MOV  C,M     ; เอาค่าใน CR FLAG ไว้ที่ C
2AEA B1         ORA  C       ; รวมสัญญาณ LOGIC เข้าด้วยกัน
2AEB 00         NOP          ;
2AEC 0000000    NOP          ;
2AEF C1         POP  B       ;
```

```
2AF0 D1          POP  D       ; คืนข้อมูลที่ SAVE ไว้กลับ BC, DE, HL ตามเดิม
2AF1 E1          POP  H       ;
2AF2 C9          RET           ;
```

โปรแกรมย่อย P 141

ทดสอบ IC ประเภทที่ 7 IC เบอร์ 74C86, 74LS266, 74LS386

```
2AF3 CDD929      CALL 29D9     ; SAVE ขา IC ประเภทที่ 7
2AF6 21300F      LXI  H,0F30   ; POINTER ชี้ OUTPUT LOGIC TABLE
                               ; CLEAR 7-SEGMENT และ INPUT/OUTPUT BINARY
2AF9 CDE124      CALL 24E1     ;
2AFC CDCC29      CALL 29CC     ; ทดสอบ IC ประเภทที่ 7 SOCKET # 1
2AFF C3F32A      JMP  2AF3     ; กลับไปทดสอบว่าจะทดสอบต่อหรือ CLEAR หรือ
                               ; ทดสอบแบบ SLOW
```

```
2B02 00          NOP           ;
2B03 00          NOP           ;
2B04 00          NOP           ;
2B05 00          NOP           ;
2B06 00          NOP           ;
2B07 00          NOP           ;
2B08 00          NOP           ;
2B09 00          NOP           ;
2B0A 00          NOP           ;
2B0B 00          NOP           ;
2B0C 00          NOP           ;
2B0D 00          NOP           ;
2B0E 00          NOP           ;
```

โปรแกรมย่อย P142

เคลื่อนย้ายข้อมูลของ IC ที่จะนำมาทดสอบเก็บไว้ที่ AREA ที่จะ PROCESS โดย LOAD ค่า HL และ DE ก่อนตามลำดับ

```
2C00 7E       MOV  A,M    ; นำข้อมูลจากMEMORY มาเก็บไว้ที่ A ก่อน
2C01 FEFF     CPI  FF     ; ตรวจสอบดูว่าหมดข้อมูลหรือยัง(FF ตัวที่ 1)
2C03 C2162C   JNZ  2C16   ; ถ้ายังกลับไป LOAD ข้อมูลลงในAREA ที่เตรียม
                          ; ไว้ PROCESS
2C06 4F       MOV  C,A    ; ถ้าใช่ SAVE ข้อมูลไว้ที่ C
2C07 23       INX  H      ; ไปดูตัวถัดไป
2C08 7E       MOV  A,M    ; นำข้อมูลมาตรวจอีก
2C09 FEFF     CPI  FF     ; ตรวจว่าเท่ากับ FF หรือไม่ (เป็นแสดง
                          ; ว่าหมดข้อมูลแน่)
2C0B C8       RZ          ; ถ้าหมดข้อมูลจริง RET
2C0C EB       XCHG        ; สลับ POINTER ไปชี้ AREA ที่จะไปเก็บ
2C0D 71       MOV  M,C    ; นำข้อมูลที่ SAVE ตัวแรกไปเก็บใน MEMORY
2C0E 23       INX  H      ; เลื่อน ADDRESS ที่จะเก็บไปอีก ๑ ที่
2C0F 77       MOV  M,A    ; นำข้อมูลใน A เก็บใน MEMORY
2C10 EB       XCHG        ; สลับ POINTER ไปชี้ข้อมูลของ IC
                          ; เลื่อนค่า POINTER ชี้ ADDRESS ใน AREA
2C11 13       INX  D      ; ที่จะPROCESS ถัดไป
                          ; เลื่อนค่าPOINTER ชี้ ADDRESS ข้อมูล IC
2C12 23       INX  H      ; ถัดไป
2C13 C3002C   JMP  2C00   ; กลับไปเคลื่อนย้ายข้อมูลถัดไป
```



```
                          ; สลับPOINTER ไปชี้ ADDRESS ใน AREA
2C16 EB       XCHG        ; ที่จะ PROCESS
2C17 C30F2C   JMP  2C0F   ; ไปเคลื่อนย้ายข้อมูลไปไว้ใน AREA ที่จะPROCESS
```

โปรแกรมย่อย P143

เคลื่อนย้ายข้อมูลจากข้อมูลของ IC ที่อยู่ใน MEMORY ไปเก็บที่ AREA ที่เตรียมไว้ที่จะ PROCESS (ทุก ๆ ข้อมูลของ IC)

```
2C1A 11C04F     LXI  D,4FC0  ; POINTER ชี้ SOCKET # buffer area
2C1D CD002C     CALL 2C00    ; เคลื่อนย้ายข้อมูลไปยัง ADDRESS ที่ต้องการ
2C20 11C14F     LXI  D,4FC1  ; POINTER ชี้ PINCONTROL BUFFER AREA
2C23 CD002C     CALL 2C00    ; เคลื่อนย้ายข้อมูลไปยัง ADDRESS ที่ต้องการ
2C26 110040     LXI  D,4000  ; POINTER ชี้ INPUT LOGIC AREA
2C29 CD002C     CALL 2C00    ; เคลื่อนย้ายข้อมูลไป ยัง ADDRESS ที่ต้องการ
2C2C
2C2F
2C32 110048     LXI  D,4800  ; POINTER ชี้ OUTPUT TRUTH TABLE AREA
2C35 CD002C     CALL 2C00    ; เคลื่อนย้ายข้อมูลไปยัง ADDRESS ที่ต้องการ
2C38
2C3B
2C3E C9         RET          ;
```

โปรแกรมย่อย P144

DISPLAY ว่าใช้ SOCKET #

```
2C3F 3AC04F     LDA  4FC0    ; LOAD ข้อมูลจากM ซึ่งเป็น SOCKET #
2C42 D362       OUT  62      ; DISPLAY ใช้ SOCKET ใด
2C44 C9         RET          ;
```

โปรแกรมย่อย P145

ทดสอบ IC ทุกประเภท

```
2C45 CD1A2C     CALL 2C1A    ; เคลื่อนย้ายข้อมูลIC ทั้งหมดเข้า AREA ที่จะPROCESS
2C48 21C14F     LXI  H,4FC1  ; POINTER ชี้ ADDRESS PIN CONTROLL BUFFER
```

```
2C4B 7E          MOV  A,M     ; นำข้อมูลPIN CONTROL เก็บไว้ที่ A
                              PORT A - PIN CONTROL SETขา IC INPUT/
2C4C D3F0        OUT  F0      ; OUTPUT
2C4E 23          INX  H       ; เลื่อน ADDRESS ของ PIN CONTROL ถัดไป
2C4F 7E          MOV  A,M     ; นำข้อมูล PIN CONTROL เก็บไว้ที่ A
                              PORT B - PIN CONTROL, SETขา IC
2C50 D3F1        OUT  F1      ; INPUT/OUTPUT
2C52 23          INX  H       ; เลื่อน ADDRESS ของ PIN CONTROL ถัดไป
2C53 7E          MOV  A,M     ; นำข้อมูล PIN CONTROL เก็บไว้ที่ A
2C54 D3F2        OUT  F2      ; PORT C-PIN CONTROL, SET ขาIC INPUT/
                              OUTPUT
2C56 CD3F2C      CALL 2C3F    ; เลือก SOCKET ใดก่อนโดย
                              แจ้ง READY ผู้ใช้เสียบ IC ได้ และตรวจดูว่า
2C59 CDDD20      CALL 20DD    ; กดปุ่ม SLOW หรือไม่กดปุ่ม หรือไม่
2C5C C35F2C      JMP  2C5F    ; เรียกโปรแกรมทดสอบ
```

โปรแกรมย่อย P146

ส่ง INPUT LOGIC เข้าขา IC ทั้ง 3 PORT

```
2C5F 210040      LXI  H,4000  ; POINTER ชี้ INPUT LOGIC CONDITION
2C62 7E          MOV  A,M     ; นำค่า INPUT LOGIC จาก MEMORY เก็บไว้ที่ A
2C63 32C44F      STA  4FC4    ; INPUT/BUFFER DISPLAY
2C66 D3E0        OUT  E0      ; ส่ง INPUT LOGIC เข้าขา IC ทาง PORT A
2C68 23          INX  H       ; เลื่อนไปดูข้อมูลถัดไป
2C69 7E          MOV  A,M     ; นำค่า INPUT LOGIC จาก MEMORYเก็บไว้ที่ A
2C6A 32C54F      STA  4FC5    ; INPUT/OUTPUT BUFFER DISPLAY
2C6D D3E1        OUT  E1      ; ส่ง INPUT LOGIC เข้าขา IC ทาง PORT B
2C6F 23          INX  H       ; เลื่อนไปดูข้อมูลถัดไป
2C70 7E          MOV  A,M     ; นำ INPUT LOGIC จาก MEMORYเก็บไว้ที่ A
2C71 32C64F      STA  4FC6    ; INPUT/OUTPUT BUFFER DISPLAY
2C74 D3E2        OUT  E2      ; ส่ง INPUT LOGICเข้าขา ICทาง PORT C
                              CLEAR A=00 (DISPLAY INPUT FLAG
2C76 0000        NOP
```

```
2C78  ØØØØØØ      NOP
2C7B  ØØØØØØ      NOP
2C7E  23           INX  H        ; เลื่อนไปดูข้อมูลถัดไป
2C7F  7E           MOV  A,M      ;นำข้อมูลถัดไปจาก MEMORY  เก็บที่ A
2C8Ø  23           INX  H        ; เลื่อน  ADDRESS ถัดไป
2C81  FEFD         CPI  FD       ;ตรวจว่าจะต้องทำ CONDITION  นี้ต่อไปหรือไม่
2C83  CA932C       JZ   2C93     ;ถ้าต้องทำกลับไป DRIVE INPUT LOGIC  อีก
2C86  FEFE         CPI  FE       ;ว่า CONDITION  นี้จบหรือยัง
2C88  CAEC2C       JZ   2CEC     ;ไปตรวจสอบOUTPUT  ว่าได้ผลตรงกับTABLE  หรือไ
2C8B  FEFF         CPI  FF       ;ตรวจว่าทำครบทุก CONDITION  หรือยัง
2C8D  C2ØØ2Ø       JNZ  2ØØØ     ;ถ้ายังกลับไปตรวจCONDITION  ถัดไป
2C9Ø  C3FE2C       JMP  2CFE     ;(DATA ERROR)
```

```
2C93  E5           PUSH H        ; SAVE HL
2C94  CD9C2C       CALL 2C9C     ; ทำการรับผลและทดสอบ
2C97  E1           POP  H        ; คือ HL  ที่ SAVE  ไว้ที่เดิม
2C98  23           INX  H        ; เลื่อนไปดูข้อมูล
2C99  C3622C       JMP  2C62     ; INPUT LOGIC  ถัดไป
```

โปรแกรมย่อยที่ 4 7

รับข้อมูลไปประมวลผลจากขา OUT ของ IC

```
2C9C  DBDØ         IN   DØ       ; รับผลจาก PORT A
2C9E  32C44F       STA  4FC4     ; INPUT/OUTPUT BUFFER DISPLAY (PA)
2CA1  DBD1         IN   D1       ; รับผลจาก  PORT B
2CA3  32C54F       STA  4FC5     ; INPUT/OUTPUT BUFFER DISPLAY (PB)
2CA6  DBD2         IN   D2       ; รับผลจาก  PORT C
```

```
2CA8 32C64F     STA  4FC6    ; INPUT/OUTPUT BUFFER DISPLAY (PB)
2CAB ØØØØ       NOP
2CAD ØØØØØØ     NOP
2CB0 ØØØØØ      NOP
2CB3 2AC84F     LHLD 4FC8    ; POINTER ชี้ที่เก็บผลลัพท์
2CB6 EB         XCHG         ; SAVE ไว้ที่ DE
2CB7 2AC94F     LHLD 4FC9    ; OUTPUT TABLE
2CBA C3C32C     JMP  2CC3    ; นำค่าข้อมูลผลจากการทดสอบได้จาก PORT A


2CBD CD4922     CALL 2249    ; ตรวจสอบว่าที่ได้ตรงกับ TABLE หรือไม่ แล้วถ้าทดสอบแบบ SLOW จะ DELAY ให้ด้วย
2CC0 23         INX  H       ; เลื่อนค่าPOINTER TRUTH ข้อมูลถัดไป
2CC1 13         INX  D       ; เลื่อน ADDRESS ของ MEMMOที่เก็บผลลัพทแต่ละ CONDITION
2CC2 C9         RET          ;
```

โปรแกรมย่อย P148

ตรวจผลการทดสอบว่าตรงกันหรือไม่

```
2CC3 3AC44F     LDA  4FC4    ; นำข้อมูลผลจากการทดสอบที่ได้จาก PORT A
2CC6 47         MOV  B,A     ; SAVE A ไว้ที่ B
2CC7 3AC14F     LDA  4FC1    ; LOAD PIN CONTROL PORT A
2CCA A0         ANA  B       ; MASK เอาเฉพาะข้อมูล OUTPUT
2CCB CDBD2C     CALL 2CBD    ; ตรวจสอบผลที่ได้ว่าตรงกับ TRUTH TABLE หรือไม่ แล้ว SAVE ผลวิเคราะห์ไว้
2CCE 3AC54F     LDA  4FC5    ; นำข้อมูลผลจากการทดลองที่ได้จาก PORT B
2CD1 47         MOV  B,A     ; SAVE A ไว้ที่ B
2CD2 3AC24E     LDA  4EC2    ; LOAD PIN CONTROL PORT B
2CD5 A0         ANA  B       ; MASK เอาเฉพาะข้อมูล OUTPUT
2CD6 CDBD2C     CALL 2CBD    ; ตรวจสอบผลที่ได้ว่าตรงกับ TRUTH TABLE หรือไม่ แล้ว SAVE ผลวิเคราะห์ไว้
```

```
2CD9 3AC64F     LDA  4FC6    ; นำข้อมูลผลจากการทดสอบที่ได้จาก PORT B
2CDC 47         MOV  B,A     ; SAVE Aไว้ที่ B
2CDD 3AC34F     LDA  4FC3    ; LOAD PIN CONTROL PORT C
2CE0 A0         ANA  B       ; MASK เอาเฉพาะ OUTPUT

2CE1 CDBD2C     CALL 2CBD    ; ตรวจสอบผลที่ได้ว่าตรงกับ TRUTH TABLE หรือไม่
                             ; แล้ว SAVE ผลที่วิเคราะห์ไว้
2CE4 22C94F     SHLD 4FC9    ; เก็บค่า ADDRESS ชี้ OUTPUT TABLE
2CE7 EB         XCHG         ; HL = DE
2CE8 22C84F     SHLD 4FC8    ; เก็บค่า ADDRESS ชี้ผลลัพท์
2CEB C9         RET          ;

2CEC E5         PUSH H       ; SAVE HL
2CED CD9C2C     CALL 2C9C    ; ทำการรับผลและทดสอบ
2CF0 2AC84F     LHLD 4FC8    ; เอาค่า POINTER ชี้ผลลัพธ์ไว้ที่ HL
2CF3 36EA       MVI  M,EA    ; SET ค่า EA บอกจบ PART แล้ว
2CF5 23         INX  H       ; เลื่อน POINTER ชี้ผลลัพธ์อีก 1
2CF6 22C84F     SHLD 4FC8    ; SAVE ค่าชี้ผลลัพธ์ไว้ที่เดิม
2CF9 E1         POP  H       ; คืน HL ที่ SAVE ไว้ (ค่า ENTRY INPUT
                             ; LOGIC TABLE)
2CFA 23         INX  H       ; เลื่อนไปดูข้อมูล IN LOGIC ถัดไป
2CFB C3622C     JMP  2C62    ;

2CFE E5         PUSH H       ; SAVE HL
```

```
2CFF CD9C2C     CALL 2C9C    ; ทำการรับผลและทดสอบ
2D02 2AC84F     LHLD 4FC8    ; เอาค่า POINTER ชี้ผลลัพธ์ไว้ที่ HL
2D05 36EE       MVI  M,EE    ; SET  ค่า EE บอกจบ CONDITION สุดท้ายแล้ว
2D07 E1         POP  H       ;
2D08 CD6E23     CALL 236E    ; สรุปผลการทดสอบทั้งหมด
2D0B C9         RET          ;

;
                             POINTER ชี้ข้อมูลทั้งหมดของIC ที่จะนำมาทดสอบ
2D0C 210009     LXI  H,0900  ; เบอร์นั้น
2D0F CD452C     CALL 2C45    ; ทดสอบ  IC ได้
2D12 C30C2D     JMP  2D0C    ;

2D15 00         NOP          ;
2D16 00         NOP          ;
2D17 00         NOP          ;
2D18 00         NOP          ;
2D19 00         NOP          ;
2D1A 00         NOP          ;
2D1B 00         NOP          ;
2D1C 00         NOP          ;
2D1D 00         NOP          ;
2D1E 00         NOP          ;
2D1F 00         NOP          ;
2D20 00         NOP          ;
```

```
2D21 E5        PUSH H          ; SAVE HL
2D22 CD0028    CALL 2800       ; รับข้อมูลจาก KEYBOARD
2D25 00        NOP             ;
2D26 00        NOP             ;
2D27 FE16      CPI  16         ; CLEAR  หรือไม่
                               ; ถ้าใส่ CLEAR DISPLAY และ CLEAR MEMORY
2D29 CA8B2D    JZ   2D8B       ; 4E00-4E07 แล้วสามารถแก้ใส่เข้าข้อมูลใหม่ได้
2D2C FE00      CPI  00         ; ตรวจดูว่ากดเลข 0 เข้ามาหรือไม่
                               ; LOAD ค่าเลข 0 ลงใน MEMORYแล้วDISPLAY
2D2E CA3B2D    JZ   2D3B       ; ออก 7-SEGMENT
2D31 FE01      CPI  01         ; ตรวจดูว่ากดเลข  เข้ามาหรือไม่
                               ; LOAD ค่าเลข 1 ลงใน MEMORYแล้ว DISPLAY
2D33 CA402D    JZ   2D40       ; ออก 7-SEGMENT
2D36 D310      OUT  10         ; ส่งเสียง ALARM
2D38 C3222D    JMP  2D22       ; กลับไปตรวจ KEY ใหม่
```

```
2D3B 4F        MOV  C,A        ; SAVE เลข 0 ไว้ที่ C
2D3C 3E3F      MVI  A,3F       ; 3F เป็นค่า= เลข 0 ออก 7-SEGMENT
2D3E E1        POP  H          ;
2D3F C9        RET             ;
```

```
2D40 4F        MOV  C,A        ; SAVE เลข 1 ไว้ที่ C
2D41 3E06      MVI  A,06       ; 06 เป็นค่า= เลข 1 ออก 7-SEGMENT
2D43 E1        POP  H          ;
2D44 C9        RET             ;
```

โปรแกรมย่อย P149

รับข้อมูลจาก KEYBOARD แล้วแสดงออก 7 - SEGMENT

```
2D45 F5          PUSH PSW     ;
2D46 C5          PUSH B       ;
2D47 D5          PUSH D       ; SAVE REG, ทั้งหมด
2D48 E5          PUSH H       ;
                              POINTER ชี้ที่เก็บ DATA เมื่อ KEY
2D49 2100 4E     LXI  H,4E00  ; เบอร์ IC เข้ามา
                              รอการกด KEY รับผลจาก MEMORY
2D4C CD992D      CALL 2D99    ; ADDR 4E 00
2D4F D370        OUT  70      ; แสดงผลทาง TSER. DIGIT ที่ 1
2D51 CD992D      CALL 2D99    ; รอการกด KEY รับผลจากการKEY
                              เก็บใน MEMORY ADDR 4E 01
2D54 D371        OUT  71      ; แสดงผลทาง 7 - SEG DIGIT 2
                              รอการกด KEY รับผลจากการ KEY
2D56 CD992D      CALL 2D99    ; เก็บใน MEMORY ADDR 4002
                              แสดงผลทาง 7 - SEGMENT DIGIT
2D59 D372        OUT  72      ; ที่ 3
                              รอการกด KEY รับผลจากการ KEY
2D5B CD992D      CALL 2D99    ; เก็บใน MEMORY ADDR 4E 03
2D5E D373        OUT  73      ; แสดงผลทาง7 - SEGMENT DIGIT ที่ 4
                              รอการกด KEY รับผลจากการ KEY เก็บใน
2D60 CD992D      CALL 2D99    ; MEMORY ADDR 4E04
2D63 D374        OUT  74      ; แสดงผลทาง7 - SEG, DIGIT ที่ 5
                              รอการกด KEY รับผลจากการ KEY เก็บใน
2D65 CD992D      CALL 2D99    ; MEMORY ADDR 4E05
2D68 D375        OUT  75      ; แสดงผลทาง 7 - SEG DIGIT ที่ 6
                              รอการกด KEY รับผลจากการ KEY เก็บใน
2D6A CD992D      CALL 2D99    ; MEMORY ADDR 4E06
2D6D D376        OUT  76      ; แสดงผลทาง 7 - SEG, DIGIT ที่ 7
                              รอการกด KEY รับผลจากการ KEY เก็บใน
2D6F CD992D      CALL 2D99    ; MEMORY ADDR 4E07
2D72 D377        OUT  77      ; แสดงผลทาง 7 - SEG, DIGIT ที่ 8
2D74 CD0028      CALL 2800    ; รับข้อมูลจาก KEY
2D77 FE14        CPI  14      ; ตรวจว่ากดปุ่ม ENTER หรือไม่
2D79 CA7F2E      JZ   2E7F    ; ถ้ากด RETURN
2D7C FE16        CPI  16      ; ตรวจว่ากดปุ่ม CLEAR หรือไม่
2D7E CA882E      JZ   2E88    ; CLEAR เริ่มรับข้อมูลใหม่
2D81 D310        OUT  10      ; ALARM
2D83 C3742D      JMP  2D74    ; กลับไปตรวจว่ากด ENTER หรือ CLEAR หรือไม่
```

```
2D86 E1          POP  H        ;
2D87 D1          POP  D        ;  คืนของเก่าที่ SAVE ไว้
2D88 C1          POP  B        ;
2D89 F1          POP  PSW      ;
2D8A C9          RET            ;
```

```
2D8B ØØ          NOP            ;
2D8C CD2B23      CALL 232B      ;  CLEAR DISPLAY ทั้ง8DIGIT และSET
                                   MEMORY ทั้งหมด FF ทุกที่
2D8F C3452D      JMP  2D45      ;
```

โปรแกรมทดสอบ P150

รับข้อมูลจาก KEYBOARD และแปลงรหัสเรียบร้อย

```
2D92 CD452D      CALL 2D45      ;  รับข้อมูล SAVE ไว้ใน
                                   เอาข้อมูลที่ใดใน MEMORY แปลงเป็น
2D95 CDC42D      CALL 2DC4      ;  1 BYTE
2D98 C9          RET            ;
```

```
2D99 CD212D      CALL 2D21      ;
2D9C 71          MOV  M,C       ;
2D9D 23          INX  H         ;
2D9E C9          RET            ;
```

โปรแกรมย่อย P(5)

		รวมข้อมูลจาก MEMORY	ลงเหลือ 1 BYTE LOAD ข้อมูลจากMEMCRY เลื่อนไหอยู่ในที่ BIT 7
2D9F 7E	MOV A,M	;	
2DA0 0F	RRC	;	
2DA1 77	MOV M,A	;	SAVE ไว้ที่เดิม
2DA2 23	INX H	;	
2DA3 7E	MOV A,M	;	LOAD ข้อมูลจาก MEMORY
2DA4 0F	RRC	;	เลื่อนให้อยู่ที่ BIT 6
2DA5 0F	RRC	;	
2DA6 77	MOV M,A	;	SAVE ไว้ที่เดิม
2DA7 23	INX H	;	
2DA8 7E	MOV A,M	;	LOAD ข้อมูลจาก MEMORY
2DA9 0F	RRC	;	
2DAA 0F	RRC	;	เลื่อนให้อยู่ที่ BIT 5
2DAB 0F	RRC	;	
2DAC 77	MOV M,A	;	SAVE ไว้ที่เดิม
2DAD 23	INX H	;	
2DAE 7E	MOV A,M	;	LOAD ข้อมูลจาก MEMORY
2DAF 0F	RRC	;	
2DB0 0F	RRC	;	เลื่อนให้อยู่ที่ BIT 4
2DB1 0F	RRC	;	
2DB2 0F	RRC	;	
2DB3 77	MOV M,A	;	SAVE ไว้ที่เดิม
2DB4 23	INX H	;	
2DB5 7E	MOV A,M	;	LOAD ข้อมูลจาก MRMORY
2DB6 07	RLC	;	
2DB7 07	RLC	;	

```
2DB8 07          RLC          ;
2DB9 77          MOV  M,A     ; SAVE ไว้ที่เดิม
2DBA 23          INX  H       ;
2DBB 7E          MOV  A,M     ; LOAD ข้อมูลจาก MEMORY
2DBC 07          RLC          ; เลื่อนให้อยู่ที่ BIT 2
2DBD 07          RLC          ;
2DBE 77          MOV  M,A     ; SAVE ไว้ที่เดิม
2DBF 23          INX  H       ;
2DC0 7E          MOV  A,M     ; LOAD ข้อมูลจาก MEMORY
2DC1 07          RLC          ; เลื่อนให้อยู่ที่ BIT 1
2DC2 77          MOV  M,A     ; SAVE ไว้ที่เดิม
2DC3 C9          RET          ;
```

โปรแกรมย่อย P1S2

รวมข้อมูลใน MEMORY ที่ ROTAE จาก 8 LOCATION เป็น 1 BYTE

```
2DC4 E5          PUSH H       ; SAVE HL, BC
2DC5 C5          PUSH B       ;
2DC6 21004E      LXI  H,4E00  ; POINTERชี้ที่ 4E00
2DC9 CD9F2D      CALL 2D9F    ; เลื่อน BIT ให้ตรงกับความเป็นจริง
2DCC 21004E      LXI  H,4E00  ; POINTER ชี้ที่ 4E00
2DCF 0E07        MVI  C,07    ; มี 8 จำนวน
2DD1 CDD72D      CALL 2DD7    ; OR ทั้ง 8 LOCATION
2DD4 C1          POP  B       ;   คือ BC, และ HL
2DD5 E1          POP  H       ;
2DD6 C9          RET          ;
```

```
2DD7 7E          MOV  A,M    ;
2DD8 23          INX  H      ;
2DD9 46          MOV  B,M    ;
2DDA B0          ORA  B      ;
2DDB 0D          DCR  C      ;
2DDC C2D82D      JNZ  2DD8   ;
2DDF C9          RET          ;
```

โปรแกรมย่อย P153

ทดสอบแบบ MANUAL TEST ช่วงรับข้อมูล

```
2DE0 CD292E      CALL 2E29   ; DISPLAY "SOCKET"
2DE3 CD922D      CALL 2D92   ; รับ ข้อมูลเข้าระบบ
2DE6 32C04F      STA  4FC0   ; STORGE SOCKET ที่เลือกใช้
2DE9 CD302E      CALL 2E30   ; DISPLAY "PIN CONTROL"
2DEC CD922D      CALL 2D92   ; รับข้อมูลเข้าระบบ
2DEF 32C14F      STA  4FC1   ; PA PIN CON
2DF2 CD922D      CALL 2D92   ; รับข้อมูลเข้ามา
2DF5 32C24F      STA  4FC2   ; PB PIN CON
2DF8 CD922D      CALL 2D92   ; รับข้อมูลเข้ามา
2DFB 32C34F      STA  4FC3   ; PC PIN CON
2DFE CD372E      CALL 2E37   ; DISPLAY "INPUT LOGIC"
2E01 210040      LXI  H,4000 ; POINTERที่เก็บ INPUT LOGIC
2E04 CD922D      CALL 2D92   ; รับข้อมูลเข้ามา
2E07 FEFF        CPI  FF     ; ตรวจว่าจบข้อมูล INPUT LOGIC หรือยัง
2E09 CA112E      JZ   2E11   ; ถ้าเท่ากับ FFจบข้อมูล INPUT LOGIC
2E0C 77          MOV  M,A    ; ไม่เท่า SAVE ข้อมูล
2E0D 23          INX  H      ; เลื่อนไป ADDRESS ถัดไป
2E0E C3042E      JMP  2E04   ; กลับไปรับข้อมูลต่อ
```

```
2E11 77          MOV  M,A      ;
2E12 CD3E2E      CALL 2E3E     ; DISPLAY OUTPUT LOGIC
2E15 210048      LXI  H,4800   ; POINTER ชี้ที่เก็บ OUTPUT LOGIC
2E18 CD922D      CALL 2D92     ; รับข้อมูล
2E1B FEFF        CPI  FF       ; ตรวจว่าจบข้อมูลหรือยัง
2E1D CA252E      JZ   2E25     ;
2E20 77          MOV  M,A      ; ข้อมูลเก็บใน MEMORY
2E21 23          INX  H        ; เลื่อนค่า ADDRESS ถัดไป
2E22 C3182E      JMP  2E18     ; ไป PROGRAM ทดสอบ IC

2E25 77          MOV  M,A      ;
2E26 C3482C      JMP  2C48     ;
```

โปรแกรมทดสอบ P154

DISPLAY SOCKET

```
2E29 215D2E      LXI  H,2E5D   ; POINTER ชี้ข้อมูลแสดง "SOCKET"
2E2C CD752E      CALL 2E75     ; แสดงออก 7-SEGMENT
2E2F C9          RET           ;
```

โปรแกรมย่อย P155

DISPLAY PIN CONTROL

```
2E30 21632E      LXI  H,2E63   ; POINTER ชี้ข้อมูลแสดง "PIN 6"
2E33 CD752E      CALL 2E75     ; แสดงออก 7-SEGMENT
2E36 C9          RET           ;
```

โปรแกรมย่อย P156

DISPLAY INPUT

```
2E37 21692E     LXI  H,2E69  ;  POINTER ชี้ข้อมูลแสดง"INPUT"
2E3A CD752E     CALL 2E75    ;  แสดงออก 7-SEGMENT
2E3D C9         RET          ;  ---
```

โปรแกรมย่อย P157

DISPLAY OUTPUT

```
2E3E 216F2E     LXI  H,2E6F  ;  POINTER ชี้ข้อมูลแสดง "OUTPUT"
2E41 CD752E     CALL 2E75    ;  แสดงออก 7 - SEGMENT
2E44 C9         RET          ;
```

โปรแกรมย่อย P158

DISPLAY ออก 7-SEGMENT, # 78 - 7D แถวล่าง

```
2E45 7E         MOV  A,M     ;
2E46 D378       OUT  78      ;  DISPLAY 7-SEGMENT # 78
2E48 23         INX  H       ;
2E49 7E         MOV  A,M     ;
2E4A D379       OUT  79      ;  DISPLAY 7-SEGMENT # 79
2E4C 23         INX  H       ;
2E4D 7E         MOV  A,M     ;
2E4E D37A       OUT  7A      ;  DISPLAY 7-SEGMENT # 7A
2E50 23         INX  H       ;
2E51 7E         MOV  A,M     ;
2E52 D37B       OUT  7B      ;  DISPLAY 7-SEGMENT # 7B
2E54 23         INX  H       ;
```

```
2E55 7E          MOV   A,M      ;
2E56 D37C        OUT   7C       ;  DISPLAY 7-SEGMENT # 7C
2E58 23          INX   H        ;
2E59 7E          MOV   A,M      ;
2E5A D37D        OUT   7D       ;  DISPLAY 7-SEGMENT # 7D
2E5C C9          RET            ;
```

ข้อมูลสำหรับตัวอักษรต่าง ๆ

```
2E5D 6D
2E5E 3F
2E5F 39          'SOCKET '
2E60 76
2E61 79
2E62 31730006
2E65 37
2E66 00          'PIN C'
2E67 00
2E68 39
2E69 0637
2E6B 73          'INPUT'
2E6C 3E31
2E6E 00
2E6F 3F
2E70 3E31        'OUTPUT'
2E72 73
2E73 3E31
2E75 CD2B23      CALL 232B      ; CLEAR 7-SEGMENT แถวบน
```

```
2E78 CD6628       CALL 2866     ;  DELAY 10mS
2E7B CD452E       CALL 2E45     ;  กลับไปรับข้อมูลใหม่
2E7E C9           RET           ;
```

```
2E7F CD2B23       CALL 232B     ;  CLEAR 7-SEGMENT แถวบน
2E82 CD6628       CALL 2866     ;  DELAY 10mS
2E85 C3862D       JMP  2D86     ;  กลับไปคืนของที่ SAVE ไว่แล้ว RETURN
```

```
2E88 CD2B23       CALL 232B     ;  CLEAR 7-SEGMENT แถวบน
2E8B C3492D       JMP  2D49     ;  กลับไปรับข้อมูลใหม่
```

โปรแกรมทดสอบ

IC 2 I/P NAND GATE

```
400 CD1408      CALL 0814     ; โชว์ DISPLAY แสดงว่าเป็น NAND GATE
                              กลับไปทำโปรแกรมทดสอบ IC 2 I/P
403 CDFB26      CALL 26FB     ; NAND GATE
406 C9          RET           ;
```

โปรแกรมย่อยทดสอบ

IC 2 I/P AND GATE

```
407 CD0A08      CALL 080A     ; โชว์ DISPLAY แสดงว่าเป็น AND GATE
40A CDAA26      CALL 26AA     ; กลับไปทำโปรแกรมทดสอบ IC 2 I/P
                              AND GATE
40D C9          RET           ;
```

โปรแกรมทดสอบ

IC 2 I/P OR GATE

```
40E CD0F08      CALL 080F     ;
                              โชว์ DISPLAY แสดงว่าเป็น OR GATE
411 CDB626      CALL 26B6     ; กลับไปทำโแรแกรมทดสอบ
                              IC OR GATE
414 C9          RET           ;
```

โปรแกรมทดสอบ

IC 2 I/P EXCLUSIVE OR GATE

```
415 CD2308      CALL 0823     ; โชว์ DISPLAY แสดงว่าเป็นEXCLUSIVE
                              OR GATE
418 CDC226      CALL 26C2     ; กลับไปทำโปรแกรมทดสอบ IC EXCLUSIVE OR
41B C9          RET           ;
```

โปรแกรมทดสอบ

IC TRI STATE # 74125

```
41C CD1E08      CALL 081E     ;  โชว์ DISPLAY แสดงว่าเป็น TRI-STATE
41F CDCE26      CALL 26CE     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC TRI-STATE # 74125
422 C9          RET           ;
```

โปรแกรมทดสอบ

IC INVERTOR

```
423 CD0008      CALL 0800     ;  โชว์DISPLAY แสดงว่าเป็น IC INVERTOR
426 CDD625      CALL 25D6     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC INVERTOR
429 C9          RET           ;
```

โปรแกรมทดสอบ

IC BUFFER

```
42A CD0508      CALL 0805     ;  โชว์DISPLAY แสดงว่าเป็น IC BUFFER
42D CDE525      CALL 25E5     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC BUFFER
430 C9          RET           ;
```

โปรแกรมทดสอบ

IC 2 I/P NOR GATE

```
431 CD1908      CALL 0819     ;  โชว์ DISPLAY แสดงว่าเป็น IC NOR GATE
434 CD3B27      CALL 273B     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC 2 I/P NOR GATE
437 C9          RET           ;
```

โปรแกรมทดสอบ

IC 2 I/P NAND GATE # 7401, 7439

```
438 CD1408      CALL 0814     ;  โชว์ DISPLAY แสดงว่าเป็นIC NAND GATE
43B CD7328      CALL 2873     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC NAND GATE
43E C9          RET           ;
```

โปรแกรมทดสอบ

IC 3 I/P NAND GATE # 7410, 7412

```
43F CD1408      CALL 0814     ;  โชว์DISPLAY แสดงว่าเป็นIC NAND GATE
442 CD0929      CALL 2909     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC 3 I/P NAND GATE
445 C9          RET           ;
```

โปรแกรมทดสอบ

3 I/P AND GATE # 7411

```
446 CD0A08      CALL 080A     ;  โชว์DISPLAY แสดงว่าเป็นIC AND GATE
449 CD2729      CALL 2927     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC 3 I/P AND GATE
44C C9          RET           ;
```

โปรแกรมทดสอบ

3 I/P NOR GATE # 7427

```
44D CD1908      CALL 0819     ;  โชว์ DISPLAY ว่าเป็น IC NOR GATE
450 CD1829      CALL 2918     ;  กลับไปทำโปรแกรมทดสอบ
                                 IC 3 I/P NOR GATE
453 C9          RET           ;
```

โปรแกรมทดสอบ

IC 4 I/P NAND GATE # 7420, 7422, 7440, 7413

```
454 CD1408      CALL 0814     ;  โชว์ DISPLAY แสดงว่าเป็น NAND GATE
457 CDAE29      CALL 29AE     ;  กลับไปทำโปรแกรมทดสอบ 4 I/P NAND GATE
45A C9          RET           ;
```

โปรแกรมทดสอบ

IC TRI-STATE # 74126

```
45B CD1E08      CALL 081E     ;  โชว์ DISPLAY แสดงว่าเป็น TRI-STATE
45E CDDA26      CALL 26DA     ;  กลับไปทำโปรแกรมทดสอบ IC TRI-STATE
461 C9          RET           ;
```

โปรแกรมทดสอบ

IC AND GATE 4 INPUT # 7421

```
462 CD0A08      CALL 080A     ;  โชว์ DISPLAYแสดงว่าเป็น AND GATE
465 CDBC29      CALL 29BC     ;  กลับไปทำโปรแกรมทดสอบ IC AND GATE
468 C9          RET           ;
```

โปรแกรมทดสอบ

IC NAND GATE # 7430

```
469 CD1408      CALL 0814     ;  โชว์ DISPLAY แสดงว่า NAND GATE
46C CD382A      CALL 2A38     ;  กลับไปทำโปรแกรมทดสอบ IC NAND GATE
46F C9          RET           ;
```

โปรแกรมทดสอบ

IC SHIFT REGISTER

```
470 CD5008      CALL 0850     ; โชว์ DISPLAY แสดงว่าเป็น SHIFT REGISTER
473 CDD02A      CALL 2AD0     ; กลับไปทำโปรแกรมทดสอบ
                                IC SHIFT REGISTER
476 C9          RET           ;
```

โปรแกรมทดสอบ

INVERTOR

```
0800 3E01       MVI   A,01      ; DISPLAY INVERTOR
0802 D390       OUT   90        ; LED สว่างแสดงผลว่าเป็น INVERTOR
0804 C9         RET             ;
```

โปรแกรมทดสอบ

BUFFER

```
0805 3E02       MVI   A,02      ; DISPLAY BUFFER
0807 D390       OUT   90        ; LED สว่าง แสดงว่าเป็น BUFFER
0809 C9         RET             ;
```

โปรแกรมทดสอบ

AND GATE

```
080A 3E04       MVI   A,04      ; DISPLAY AND GATE
080C D390       OUT   90        ; LED สว่างแสดงว่าเป็น AND GATE
080E C9         RET             ;
```

โปรแกรมทดสอบ

OR GATE

```
080F 3E08       MVI   A,08      ; DISPLAY OR GATE
0811 D390       OUT   90        ; LED สว่างแสดงว่าเป็น OR GATE
0813 C9         RET             ;
```

โปรแกรมทดสอบ

NAND GATE

```
0814 3E10       MVI   A,10      ; DISPLAY NAND GATE
0816 D390       OUT   90        ; LED สว่าง แสดงว่าเป็น NAND GATE
0818 C9         RET             ;
```

โปรแกรมทดสอบ

NOR GATE

```
0819 3E20       MVI   A,20      ; DISPLAY NOR GATE
081B D390       OUT   90        ; LED สว่าง แสดงว่าเป็น NOR GATE
081D C9         RET             ;
```

โปรแกรมทดสอบ
TRI - STATE

```
081E 3E40       MVI   A,40      ; DISPLAY TRI - STATE
0820 D390       OUT   90        ; LED สว่าง แสดงว่าเป็นTRI - STATE
0822 C9         RET             ;
```

โปรแกรมทดสอบ

EX - OR GATE

```
0823 3E80       MVI   A,80      ; DISPLAY EX - OR GATE
0825 D390       OUT   90        ; LED สว่าง แสดงว่าเป็น EX - OR GATE
0827 C9         RET             ;
```

โปรแกรมทดสอบ

RS - FLIP FLOP

```
0828 3E01       MVI   A,01      ;  DISPLAY RS-FLID FLOP
082A D391       OUT   91        ;  LED สว่างแสดงว่าเป็น RS-FLIP FLOP
082C C9         RET             ;
```

โปรแกรมทดสอบ

D - FLIP FLOP

```
082D 3E02       MVI   A,02      ;  DISPLAY D-FLIP FLOP
082F D391       OUT   91        ;  LED สว่างแสดงว่าเป็น D-FLIP FLOP
0831 C9         RET             ;
```

โปรแกรมทดสอบ

JK FLIP FLOP

```
0832 3E04       MVI   A,04      ;  DISPLAY JK FLID FLOP
0834 D391       OUT   91        ;  LED สว่างแสดงว่าเป็น JK FLIP FLOP
0836 C9         RET             ;
```

โปรแกรมทดสอบ

ENCODER

```
0837 3E08       MVI   A,08      ;  DESPLAY ENCODER
0839 D391       OUT   91        ;  LED สว่างแสดงว่าเป็น ENCODER
083B C9         RET             ;
```

โปรแกรมทดสอบ

DECODER

```
083C 3E10       MVI   A,10      ; DISPLAY DECODER
083E D391       OUT   91        ; LED สว่างแสดงว่าเป็นDECODER
0840 C9         RET             ;
```

โปรแกรมทดสอบ

MULTIPLEXER

```
0841 3E20       MVI   A,20      ; DISPLAY MULTIPLEXER
0843 D391       OUT   91        ; LED สว่างแสดงว่าเป็น MULTIPLEXER
0845 C9         RET             ;
```

โปรแกรมทดสอบ

DEMULTIPLEXER

```
0846 3E40       MVI   A,40      ; DISPLAY DEMULTIPLEXER
0848 D391       OUT   91        ; LED สว่างแสดงว่าเป็น DEMULTIPLEXER
084A C9         RET             ;
```

โปรแกรมทดสอบ

COUNTER

```
084B 3E80       MVI   A,80      ; DISPLAY COUNTER
084D D391       OUT   91        ; LED สว่างแสดงว่าเป็น COUNTER
084F C9         RET             ;
```

โปรแกรมทดสอบ

SHIFT REGISTER

```
0850 3E01       MVI   A,01      ; DISPLAY SHIFT REGISTER
0852 D392       OUT   92        ; LED สว่างแสดงว่าเป็น SHIFT REGISTER
0854 C9         RET             ;
```

โปรแกรมทดสอบ

COMPARATOR

```
0855 3E02       MVI   A,02      ; DISPLAY COMPARATOR
0857 D392       OUT   92        ; LED สว่างแสดงว่าเป็น COMPARATOR
0859 C9         RET             ;
```

โปรแกรมทดสอบ

OTHER IC

```
085A 3E80       MVI   A,80      ; DISPLAY OTHER IC
085C D392       OUT   92        ; LED สว่าง แสดงว่าเป็น OTHER IC
085E C9         RET             ;
```



```
085F 3E00       MVI   A,00      ;
0861 D390       OUT   90        ; RESET PORT A
0863 D391       OUT   91        ; RESET PORT B
0865 D392       OUT   92        ; RESET PORT C
0867 C9         RET             ;
```

```
000 31FF1F      LXI   SP,1FFF ; TOP OF STACK
003 3E80        MVI   A,80    ;
005 D393        OUT   93      ;
007 3E16        MVI   A,16    ;
009 D380        OUT   80      ;   INITIALIZE 8259
00B 3E01        MVI   A,01    ;
00D D381        OUT   81      ;
00F FB          EI            ;
010 C30020      JMP   2000    ;ไปทำโปรแกรมทดสอบ IC


100 C30002      JMP   0200    ; INTERRUPT IC SHORT CCT.


200 F3          DI            ; DISABLE INTERRUPT
201 3E80        MVI   A,80    ;
203 32D74F      STA   4FD7    ;   DISPLAY IC SHORT CCT.
206 CD0003      CALL  0300    ;
209 D330        OUT   30      ; ส่งเสียง ALARM
20B CD4A27      CALL  274A    ;   DELAY 2 SEC
20E CD4A27      CALL  274A    ;
211 D340        OUT   40      ; RESET ALARM
213 3E20        MVI   A,20    ; RESET INTERRUPT SERVICE
215 D380        OUT   80      ; ROUTINE
217 FB          EI            ; ENABLE INTRRUPT
218 C9          RET           ;
```

โปรแกรม DISPLAY GOOD/BAD/IC SHORT CCT

```
3Ø0 E5          PUSH H          ;
3Ø1 C5          PUSH B          ;
3Ø2 21D04F      LXI  H,4FD0     ;
3Ø5 ØE07        MVI  C,Ø7       ;
3Ø7 CDD72D      CALL 2DD7       ;
3ØA C1          POP  B          ;
3ØB E1          POP  H          ;
3ØC C9          RET             ;
```

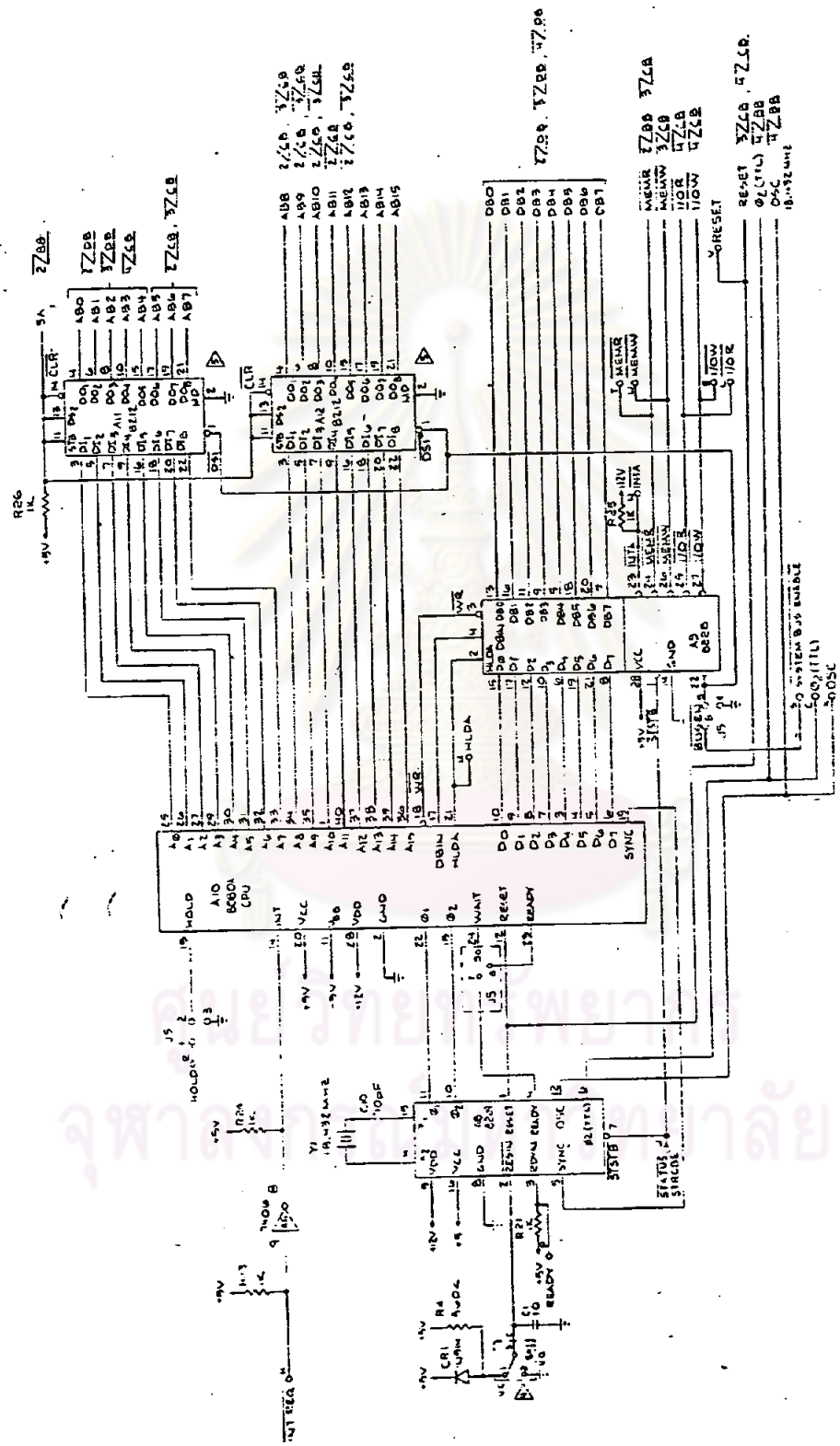
ข้อมูลเบอร์ ICและ POINTER ชี้ PROGRAM

```
ØBØØ ØØ AF Ø3 AF 26 AF 37 AF 38 AF 13 2A Ø8 AF Ø9 AF
ØB1Ø 32 AF 86 AF 12 5A 12 6A Ø4 AF Ø5 AF Ø6 AF 14 AF
ØB2Ø 16 AF Ø7 AF 17 AF Ø2 AF 28 AF 33 AF Ø1 AF 39 AF
ØB3Ø 1Ø AF 11 AF 12 AF 27 AF 2Ø AF 22 AF 4Ø AF 13 AF
ØB4Ø 21 AF 3Ø AF 16 4A ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
ØB5Ø ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
ØB6Ø ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ
ØB7Ø Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 Ø7 Ø4 Ø7
ØB8Ø Ø4 ØE Ø4 15 Ø4 1C Ø4 5B Ø4 23 Ø4 23 Ø4 23 Ø4 23
ØB9Ø Ø4 23 Ø4 2A Ø4 2A Ø4 31 Ø4 31 Ø4 31 Ø4 38 Ø4 38
ØBAØ Ø4 3F Ø4 46 Ø4 3F Ø4 4D Ø4 54 Ø4 54 Ø4 54 Ø4 54
ØBBØ Ø4 62 Ø4 69 Ø4 7Ø ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ

ØCØØ ØØ AF Ø1 AF Ø8 AF Ø4 AF Ø5 AF 1Ø AF 11 AF 15 AF
ØC1Ø 2Ø AF 22 AF 4Ø AF 21 AF 3Ø AF FF FF FF FF FF FF
ØC2Ø FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØC3Ø ØØ AF Ø3 AF 13 2A Ø8 AF Ø9 AF 32 AF 86 AF Ø4 AF
ØC4Ø Ø5 AF Ø2 AF 1Ø AF 11 AF 15 AF 2Ø AF 22 AF 4Ø AF
ØC5Ø 14 ØA 3Ø AF BØ 26 F3 26 F3 26 F3 26 AA 26 AA 26
ØC6Ø B6 26 C2 25 D6 FF FF FF FF FF FF FF FF FF FF FF
ØC7Ø Ø4 ØØ Ø4 ØØ Ø4 Ø7 Ø4 23 Ø4 23 Ø4 3F Ø4 46 Ø4 46
ØC8Ø Ø4 54 Ø4 54 Ø4 54 Ø4 62 Ø4 69 FF FF FF FF FF FF
ØC9Ø FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØCAØ FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØCBØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 Ø7 Ø4 Ø7 Ø4 ØE Ø4 15 Ø4 23
ØCCØ Ø4 23 Ø4 31 Ø4 3F Ø4 46 Ø4 46 Ø4 54 Ø4 54 Ø4 54
ØCDØ Ø4 54 Ø4 69 FF FF FF FF FF FF FF FF FF FF FF FF
ØCEØ FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØCFØ FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØDØØ ØØ AF Ø3 AF 26 AF Ø8 AF Ø9 AF 32 AF Ø4 AF Ø5 AF
ØD1Ø Ø2 AF 1Ø AF 11 AF 2Ø AF 3Ø AF FF FF FF FF FF FF
ØD2Ø FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØD3Ø Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 Ø7 Ø4 Ø7 Ø4 ØE Ø4 23 Ø4 23
ØD4Ø Ø4 31 Ø4 3F Ø4 46 Ø4 54 Ø4 69 FF FF FF FF FF FF
ØD5Ø FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØD6Ø ØØ AF Ø3 AF 26 AF 37 AF 38 AF 13 2A Ø8 AF Ø9 AF
ØD7Ø 31 AF 86 AF 13 6A 12 5A 12 6A Ø4 AF Ø5 AF 14 AF
ØD8Ø Ø2 AF 28 AF 33 AF Ø1 AF 1Ø AF 11 AF 1A F1 5A F2
ØD9Ø 7A F2 ØA F2 2A F4 ØA F1 3A F2 1A FØ 26 6A 3Ø AF
ØDAØ FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØDBØ FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
ØDCØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 ØØ Ø4 Ø7 Ø4 Ø7
ØDDØ Ø4 ØE Ø4 15 Ø4 15 Ø4 1C Ø4 5B Ø4 23 Ø4 23 Ø4 23
ØDEØ Ø4 31 Ø4 31 Ø4 31 Ø4 38 Ø4 3F Ø4 46 Ø4 3F Ø4 46
ØDFØ Ø4 4D Ø4 54 Ø4 54 Ø4 54 Ø4 54 Ø4 62 Ø4 15 Ø4 69
ØE2Ø ØØ AF Ø8 AF 32 AF Ø4 AF 14 AF Ø2 AF 1Ø AF 2Ø AF
ØE3Ø 86 AF 3Ø AF FF FF FF FF FF FF FF FF FF FF FF FF
ØE7Ø Ø4 ØØ Ø4 Ø7 Ø4 ØE Ø4 23 Ø4 23 Ø4 31 Ø4 3F Ø4 54
ØE8Ø Ø4 15 Ø4 69 FF FF FF FF FF FF FF FF FF FF FF FF

ØFØØ 3F Ø6 5B 4F 66 6D 7D Ø7 7F 6F 77 7C 39 5E 79 71
ØF1Ø 76 38 ED 1E 52 3E 16 17 FF FF FF FF FF FF FF FF
ØF2Ø Ø4 Ø4 Ø4 ØØ Ø1 ØØ ØØ Ø1 ØØ ØØ ØØ Ø4 ØØ Ø4 Ø4 Ø4
ØF3Ø ØØ Ø4 Ø4 ØØ ØØ Ø4 Ø4 Ø4 Ø4 ØØ Ø4 Ø4 Ø4 ØØ ØØ ØØ
ØF4Ø Ø4 Ø4 Ø4 Ø4 Ø4 Ø4 Ø4 ØØ Ø4 ØØ ØØ ØØ ØØ ØØ ØØ ØØ
ØF5Ø ØØ ØØ ØØ ØØ ØØ ØØ ØØ Ø4 ØØ ØØ ØØ ØØ FF FF FF FF
```

ภาคผนวก ค

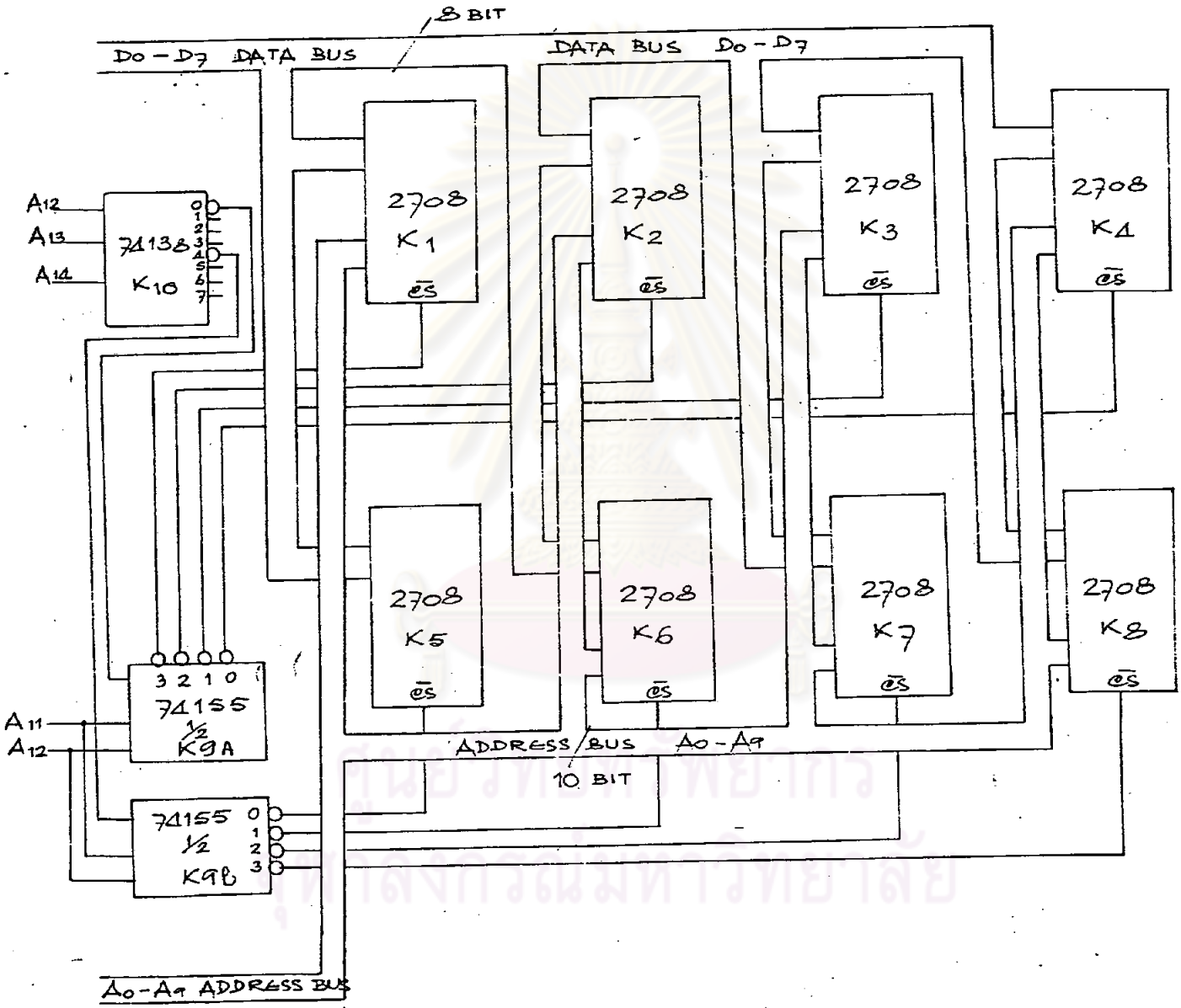
ภาคผนวกนี้เป็นการแสดงองค์ประกอบของ MICROPROCESSOR ที่นำมาใช้กับเครื่องทดสอบวงจรประมวลเชิงเลข จากรูปที่ ๓.๑

ก) แสดงวงจร CPU ใช้ PROCOPROCESSOR เบอร์ 8080 ดังรูปที่ ๓.๑

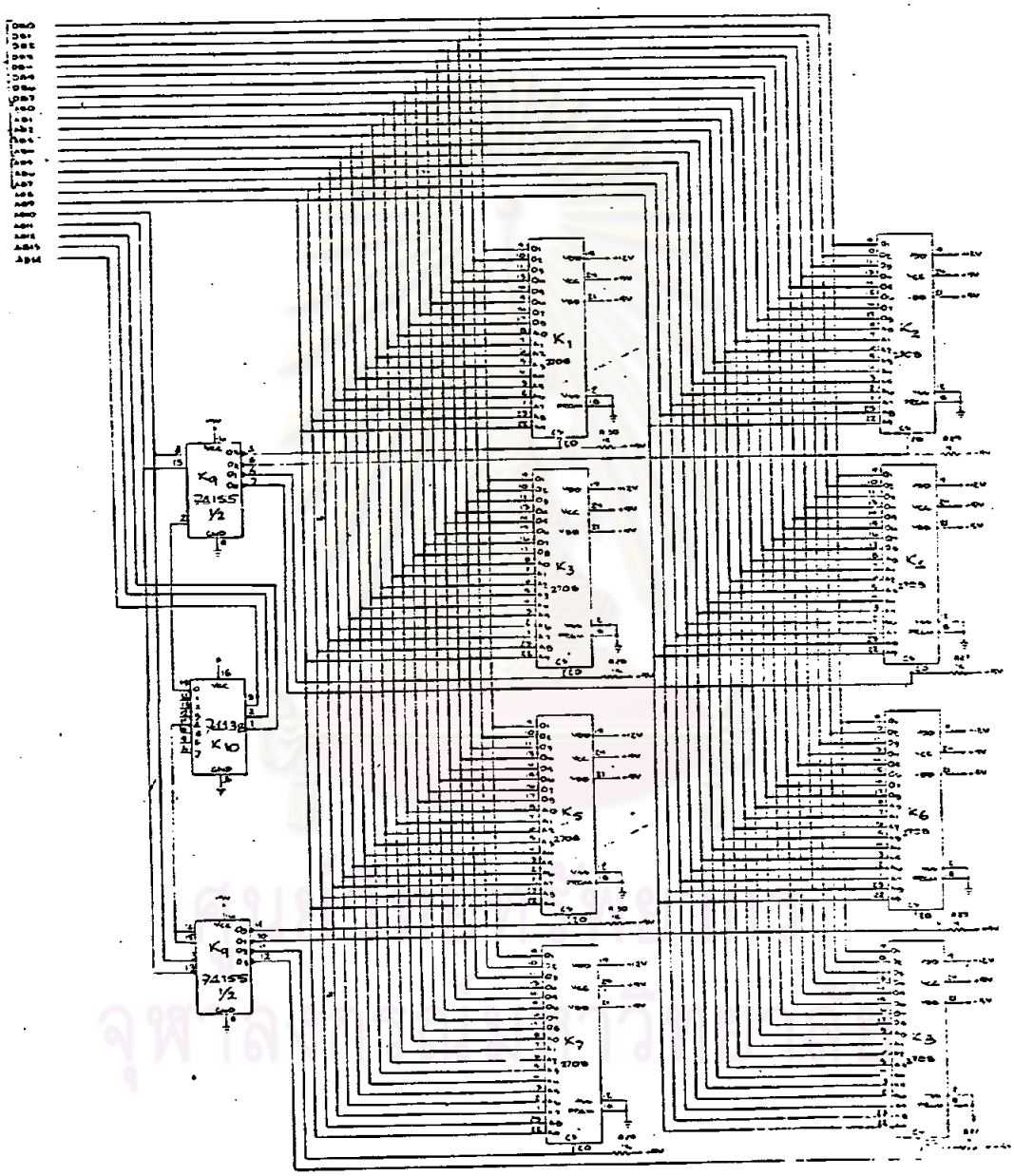


รูปที่ ๓.๑ วงจร MPU ที่ประกอบขึ้นอย่างสมบูรณ์ที่ใช้กับเครื่องทดสอบนี้

ข) หน่วยความจำที่เป็น FIXED PROGRAM จะมี ลักษณะ BLOCK DIAGRAM ของวงจร MEMORY ที่เป็น EPROM ขนาด 8K BYTE ใช้เบอร์ IC 2708 จำนวน ๘ ตัว ดังรูปที่ ข.๒ และวงจรจริงในรูปที่ ค.๓

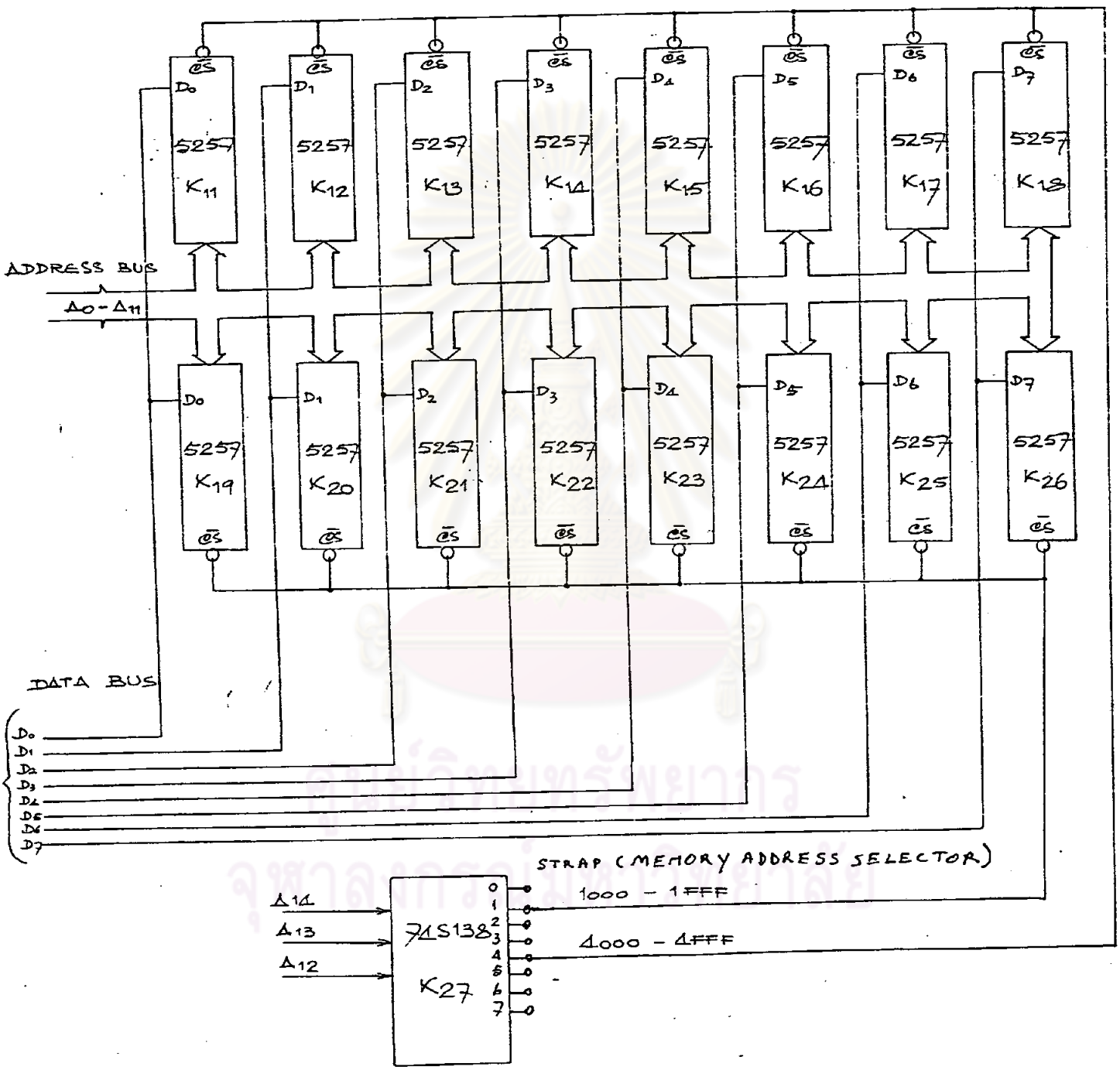


รูปที่ ค.๒ แสดง BLOCK DIAGRAM MEMORY ที่เป็นประเภทถาวร EPROM ขนาด 8k BYTE

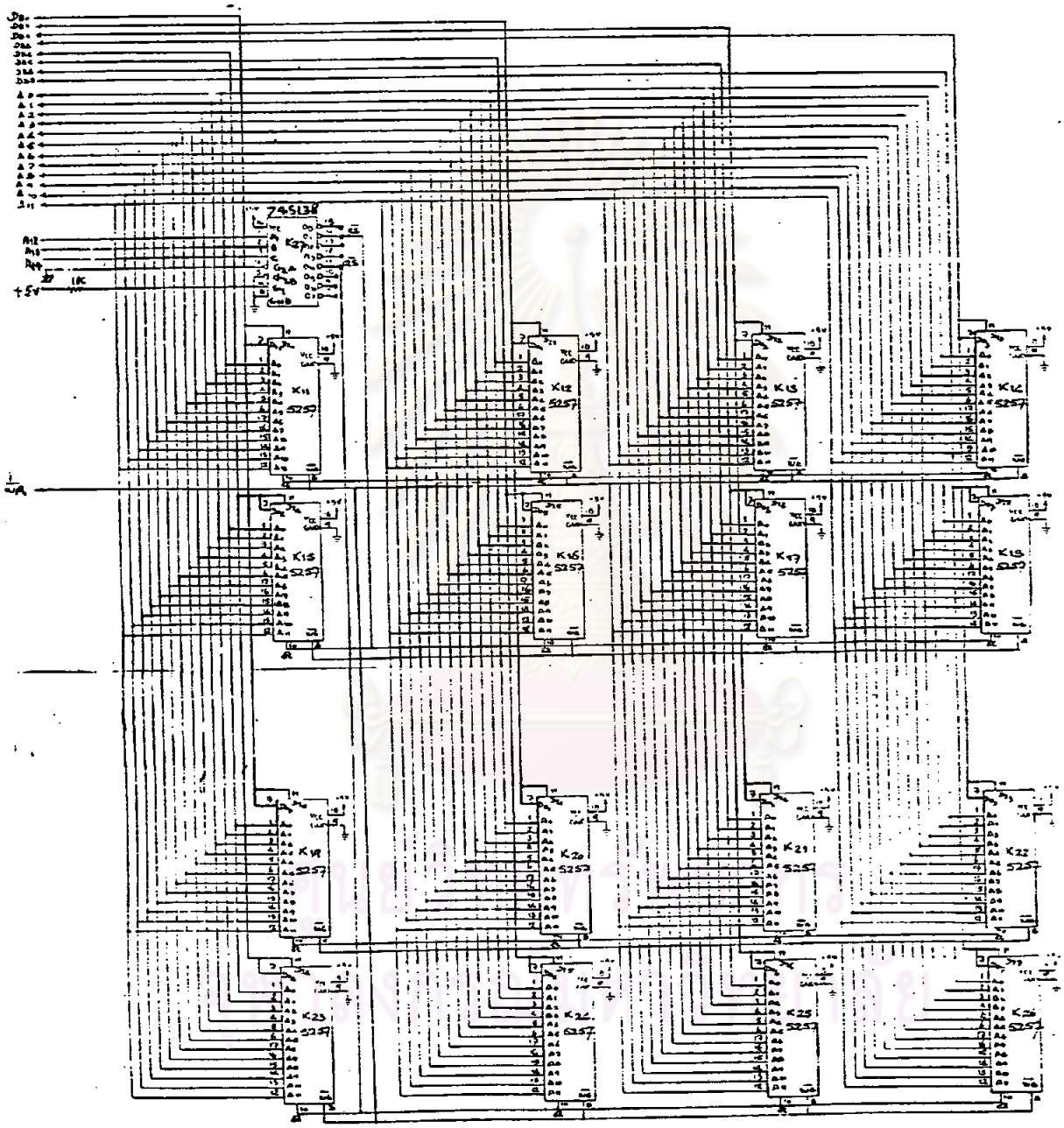


รูปที่ ฅ.๓ วงจร EPROM 8K BYTE

ค) หน่วยความจำที่ทำหน้าที่เป็นตัวเก็บข้อมูลชั่วคราวที่ใช้กับเครื่องทดสอบวงจรประมวลเชิงเลข ขนาด 8K RAM ดูรูป BLOCK DIAGRAM ในรูปที่ ค.๔ และวงจรจริงในรูป ที่ ค.๕



รูปที่ ค.๔ BLOCK DIAGRAM วงจร RAM ขนาน 8KBYTE



รูปที่ ค.๔ วงจร RAM 8K BYTE

ภาคผนวก ง

TTL CHARACTERISTICS

GLOSSARY

Currents — Positive current is defined as conventional current flow into a device. Negative current is defined as conventional current flow out of a device. All current limits are specified as absolute values.

I_{CC} **Supply Current** — The current flowing into the V_{CC} supply terminal of a circuit with the specified input conditions and the outputs open. When not specified, input conditions are chosen to guarantee worst case operation.

I_{IH} **Input HIGH Current** — The current flowing into an input when a specified HIGH voltage is applied.

I_{IL} **Input LOW Current** — The current flowing out of an input when a specified LOW voltage is applied.

I_{OH} **Output HIGH Current** — The leakage current flowing into a turned off open-collector output with a specified HIGH output voltage applied. For an output with an internal pull-up circuit, the I_{OH} is the current flowing out of the output when it is in the HIGH state.

I_{OL} **Output LOW Current** — The current flowing into an output when it is in the LOW state.

I_{OS} **Output Short Circuit Current** — The current flowing out of a HIGH-state output when that output is short circuited to ground (or other specified potential).

I_{OZH} **Output OFF Current HIGH** — The current flowing into a disabled 3-state output with a specified HIGH output voltage applied.

I_{OZL} **Output OFF Current LOW** — The current flowing out of a disabled 3-state output with a specified LOW output voltage applied.

Voltages — All voltages are referenced to the ground pin. Negative voltage limits are specified as absolute values (i.e., -10 V is greater than -1.0 V).

V_{CC} **Supply Voltage** — The range of power supply voltage over which the device is guaranteed to operate within the specified limits.

$V_{CD(Max)}$ **Input Clamp Diode Voltage** — The most negative voltage at an input when a specified current is forced out of that input terminal. This parameter guarantees the integrity of the input diode, intended to clamp negative ringing at the input terminal.

V_{IH} **Input HIGH Voltage** — The range of input voltages that represents a logic HIGH in the system.

$V_{IH(Min)}$ **Minimum Input HIGH Voltage** — The minimum allowed input HIGH in a logic system. This value represents the guaranteed input HIGH threshold for the device.

V_{IL} **Input LOW Voltage** — The range of input voltages that represents a logic LOW in the system.

$V_{IL(Max)}$ **Maximum Input LOW Voltage** — The maximum allowed input LOW in a system. This value represents the guaranteed input LOW threshold for the device.

GLOSSARY (Cont'd)

$V_{OH(Min)}$ **Output HIGH Voltage** — The minimum voltage at an output terminal for the specified output current I_{OH} and at the minimum value of V_{CC}.

$V_{OL(Max)}$ **Output LOW Voltage** — The maximum voltage at an output terminal sinking the maximum specified load current I_{OL}.

V_{T+} **Positive-Going Threshold Voltage** — The input voltage of a variable threshold device (i.e., Schmitt Trigger) that is interpreted as a V_{IH} as the input transition rises from below $V_{T-(Min)}$.

V_{T-} **Negative-Going Threshold Voltage** — The input voltage of a variable threshold device (i.e., Schmitt Trigger) that is interpreted as a V_{IL} as the input transition falls from above $V_{T+(Max)}$.

AC Switching Parameters

f_{max} **Toggle Frequency/Operating Frequency** — The maximum rate at which clock pulses may be applied to a sequential circuit. Above this frequency the device may cease to function.

t_{PLH} **Propagation Delay Time** — The time between the specified reference points, normally 1.5 V (1.3 V for LS) on the input and output voltage waveforms, with the output changing from the defined LOW level to the defined HIGH level.

t_{PHL} **Propagation Delay Time** — The time between the specified reference points, normally 1.5 V (1.3 V for LS) on the input and output voltage waveforms, with the output changing from the defined HIGH level to the defined LOW level.

t_w **Pulse Width** — The time between 1.5 V (1.3 V for LS) amplitude points on the leading and trailing edges of a pulse.

t_h **Hold Time** — The interval immediately following the active transition of the timing pulse (usually the clock pulse) or following the transition of the control input to its latching level, during which interval the data to be recognized must be maintained at the input to ensure its continued recognition. A negative hold time indicates that the correct logic level may be released prior to the active transition of the timing pulse and still be recognized.

t_s **Setup Time** — The interval immediately preceding the active transition of the timing pulse (usually the clock pulse) or preceding the transition of the control input to its latching level, during which interval the data to be recognized must be maintained at the input to ensure its recognition. A negative setup time indicates that the correct logic level may be initiated sometime after the active transition of the timing pulse and still be recognized.

t_{PHZ} **Output Disable Time (of a 3-State Output) from HIGH Level** — The time between the 1.5 V (1.3 V for LS) level on the input and a voltage 0.5 V below the steady state output HIGH level with the 3-state output changing from the defined HIGH level to a high impedance (off) state.

t_{PLZ} **Output Disable Time (of a 3-State Output) from LOW Level** — The time between the 1.5 V (1.3 V for LS) level on the input and a voltage 0.5 V above the steady state output LOW level with the 3-state output changing from the defined LOW level to a high impedance (off) state.

t_{PZH} **Output Enable Time (of a 3-State Output) to a HIGH Level** — The time between the 1.5 V (1.3 V for LS) levels of the input and output voltage waveforms with the 3-state output changing from a high impedance (off) state to a HIGH level.

GLOSSARY (Cont'd)

t_{PZL} — Output Enable Time (of a 3-State Output) to a LOW Level — The time between the 1.5 V (1.3 V for LS) levels of the input and output voltage waveforms with the 3-state output changing from a high impedance (off) state to a LOW level.

t_{rec} — Recovery Time — The time between the 1.5 V (1.3 V for LS) level on the trailing edge of an asynchronous input control pulse and the same level on a synchronous input (clock) pulse such that the device will respond to the synchronous input.

Miscellaneous

C — Marking code letter indicating that the device is guaranteed to meet the specifications for the Commercial temperature range.

D — Package code letter for ceramic Dual In-line Packages.

F — Package code letter for ceramic flatpaks.

M — Marking code letter indicating that the device is guaranteed to meet the specifications for the Military temperature range.

P — Package code letter for plastic Dual In-line Packages.

QB — Marking code indicating in-house 38510, level B reliability screening (military grade only).

QM, QR — Marking code indicating Matrix VI commercial/industrial reliability screening.

XC, XM — Shorthand for the commercial or military temperature range specifications or devices; the letter X stands for the code letter of any package in which the device is available.

DEFINITION OF SYMBOLS AND TERMS — CMOS

CURRENTS — Positive current is defined as conventional current flow into a device. Negative current is defined as conventional current flow out of a device.

I_{IN} — (Input Current) — The current flowing into a device at specified input voltage and V_{DD}.

I_{OH} — (Output HIGH Current) — The drive current flowing out of the device at specified HIGH output voltage and V_{DD}.

I_{OL} — (Output LOW Current) — The drive current flowing into the device at specified LOW output voltage and V_{DD}.

I_{DD} — (Quiescent Power Supply Current) — The current flowing into the V_{DD} lead at specified input and V_{DD} conditions.

I_{OZH} — (Output OFF Current HIGH) — The leakage current flowing into the output of a 3-state device in the "OFF" state at a specified HIGH output voltage and V_{DD}.

I_{OZL} — (Output OFF Current LOW) — The leakage current flowing out of a 3-state device in the "OFF" state at a specified HIGH output voltage and V_{DC}.

I_{IL} — (Input Current LOW) — The current flowing into a device at a specified LOW level input voltage and a specified V_{DD}.

I_{IH} — (Input Current HIGH) — The current flowing into a device at a specified HIGH level input voltage and a specified V_{DD}.

I_{DDL} — (Quiescent Power Supply Current LOW) — The current flowing into the V_{DD} lead with a specified LOW level input voltage on all inputs and specified V_{DD} conditions.

I_{DDH} — (Quiescent Power Supply Current HIGH) — The current flowing into the V_{DD} lead with a specified HIGH level input voltage on all inputs and specified V_{DD} conditions.

I_Z — (OFF State Leakage Current) — The leakage current flowing into the output of a 3-state device in the "OFF" state at a specified output voltage and V_{DD}.

VOLTAGES — All voltages are referenced to V_{SS} (or V_{EE}) which is the most negative potential applied to the device.

V_{DD} — (Drain Voltage) — The most positive potential on the device.

V_{IH} — (Input HIGH Voltage) — The range of input voltages that represents a logic HIGH level in the system.

V_{IL} — (Input LOW Voltage) — The range of input voltages that represents a logic LOW level in the system.

V_{IH} (min) — (Minimum Input HIGH Voltage) — The minimum allowed input HIGH level in a logic system.

V_{IL} (max) — (Maximum Input LOW Voltage) — The maximum allowed input LOW level in a system.

V_{OH} — (Output HIGH Voltage) — The range of voltages at an output terminal with specified output loading and supply voltage. Device inputs are conditioned to establish a HIGH level at the output.

V_{OL} — (Output LOW Voltage) — The range of voltages at an output terminal with specified output loading and supply voltage. Device inputs are conditioned to establish a LOW level at the output.

V_{SS} — (Source Voltage) — For a device with a single negative power supply, the most negative power supply, used as the reference level for other voltages. Typically ground.

V_{EE} — (Source Voltage) — One of two (V_{SS} and V_{EE}) negative power supplies. For a device with dual negative power supplies, the most negative power supply used as a reference level for other voltages.

ANALOG TERMS

R_{ON} — (ON Resistance) — The effective "ON" state resistance of an analog transmission gate, at specified input voltage, output load and V_{DD}.

ΔR_{ON} — ("Δ" ON Resistance) — The difference in effective "ON" resistance between any two transmission gates of an analog device at specified input voltage, output load and V_{DD}.

AC SWITCHING PARAMETERS

f_{MAX} — (Toggle Frequency/Operating Frequency) — The maximum rate at which clock pulses may be applied to a sequential circuit with the output of the circuit changing between 10% of V_{DD} and 90% of V_{DD}. Above this frequency the device may cease to function. See Figure 7-15.

t_{PLH} — (Propagation Delay Time) — The time between the specified reference points, normally 50% points on the input and output voltage waveforms, with the output changing from the defined LOW level to the defined HIGH level. See Figure 7-14.

t_{PHL} — (Propagation Delay Time) — The time between the specified reference points, normally 50% points on the input and output voltage waveforms, with the output changing from the defined HIGH level to the defined LOW level. See Figure 7-14.

t_{TLH} — (Transition Time, LOW to HIGH) — The time between two specified reference points on a waveform, normally 10% to 90% of V_{DD}, which is changing from LOW to HIGH. See Figure 7-14.

t_{THL} — (Transition Time, HIGH to LOW) — The time between two specified reference points on a waveform, normally 90% to 10% of V_{DD}, which is changing from HIGH to LOW. See Figure 7-14.

t_w — (Pulse Width) — The time between 50% amplitude points on the leading and trailing edges of pulse.

t_h — (Hold Time) — The interval immediately following the active transition of the timing pulse (usually the clock pulse) or following the transition of the control input to its latching level, during which interval the data to be recognized must be maintained at the input to ensure its continued recognition. A negative hold time indicates that the correct logic level may be released prior to the active transition of the timing pulse and still be recognized.

t_s — (Set-up Time) — The interval immediately preceding the active transition of the timing pulse (usually the clock pulse) or preceding the transition of the control input to its latching level, during which interval the data to be recognized must be maintained at the input to ensure its recognition. A negative set-up time indicates that the correct logic level may be initiated sometime after the active transition of the timing pulse and still be recognized.

t_{PHZ} — (3-State Output Disable Time, HIGH to Z) — The time between the specified reference points, normally the 50% point on the Output Enable input voltage waveform and a point representing a 0.1 V_{DD} drop on the Output voltage waveform of a 3-state device, with the output changing from the defined HIGH level to a high impedance OFF state.

t_{PLZ} — (3-State Output Disable Time, LOW to Z) — The time between the specified reference points, normally the 50% point on the Output Enable input voltage waveform and a point representing a 0.1 V_{DD} rise on the Output voltage waveform of a 3-state device, with the output changing from the defined LOW level to a high impedance OFF state.

t_{PZH} — (3-State Output Enable Time, Z to HIGH) — The time between the specified reference points, normally the 50% point on the Output Enable input voltage waveform and a point representing 0.5 V_{DD} on the Output voltage waveform of a 3-state device, with the output changing from a high impedance OFF state to the defined HIGH level.

t_{PZL} — (3-State Output Enable Time, Z to LOW) — The time between the specified reference points, normally the 50% point on the Output Enable input voltage waveform and a point representing 0.5 V_{DD} on the Output voltage waveform of a 3-state device, with the output changing from a high impedance OFF state to the defined LOW level.

t_{rec} — (Recovery Time) — The time between the end of an overriding asynchronous input, typically a Clear or Reset input, and the earliest allowable beginning of a synchronous control input, typically a Clock input, normally measured at 50% points on both input voltage waveforms.

t_{CW} — (Clock Period) — The time between 50% amplitude points on the leading edges of a clock pulse.

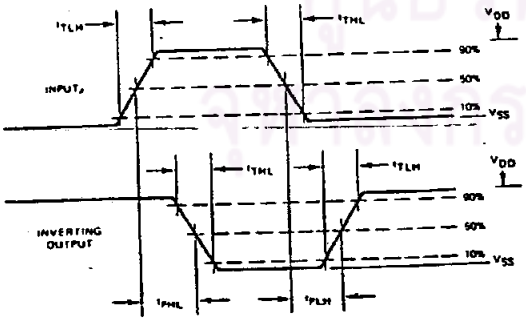


Fig. 7-14. Propagation Delay, Transition Time

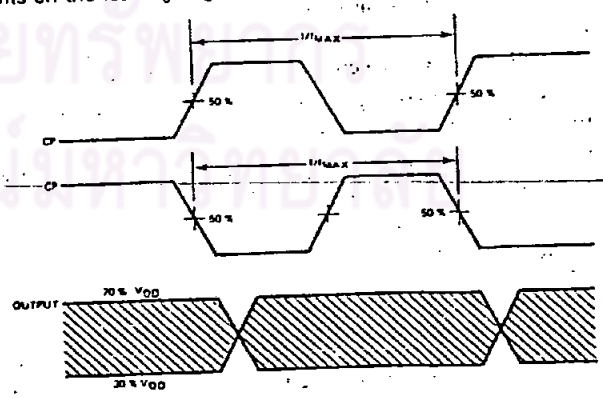


Fig. 7-15. Maximum Operating Frequency

ประวัติ

นายสุนทร วาศนา เกิดวันที่ ๒๔ พฤศจิกายน ๒๔๙๕ เกิดที่ ๕/๑ ถนนพิชัย อำเภอดุสิต กรุงเทพมหานคร จบการศึกษาขั้นปริญญาตรี คณะวิศวกรรมเทคโนโลยี จาก วิทยาลัยเทคโนโลยีและอาชีวะศึกษา ปี ๒๕๒๐

ขณะนี้ทำงานที่การสื่อสารแห่งประเทศไทย กองเทเล็กซ์ แผนกวางแผน ตำแหน่ง นายช่างโทรคมนาคม ระดับ ๕