หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

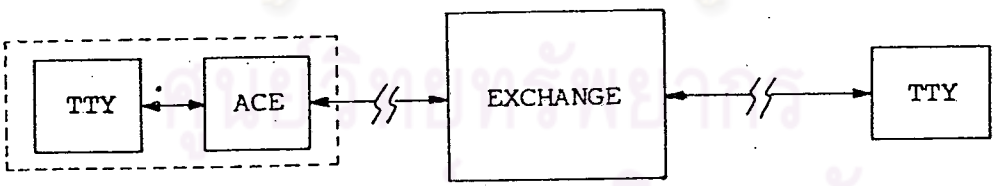
บทที่ 2

การออกแบบอินเตอร์เฟสและการทำงาน

2.1 หลักการทำงานของระบบ

เครื่องที่ได้รับการออกแบบนี้เรียกว่า เครื่องช่วยส่งเทเล็กซ์แบบอัตโนมัติ (Automatic telex Calling Equipment (ACE)) เครื่องนี้จะทำงานโดยการต่อเข้ากับวงจรเทเล็กซ์เดิม ใช้วิธีต่อเข้าระหว่างกลางระหว่างเครื่องเทเล็กซ์กับชุมสายเทเล็กซ์ ดังแสดงในรูปที่ 1

การต่อในแบบเดิม (บน) และการต่อในแบบใหม่ (ล่าง)



รูปที่ 1 ลักษณะการต่อเครื่องเพื่อใช้งานกับชุมสายเทเล็กซ์

ลักษณะการต่อเครื่องในแบบใหม่

ด้วยลักษณะการต่อดังกล่าวจะเห็นว่าตัวเครื่อง จะประกอบด้วยวงจร

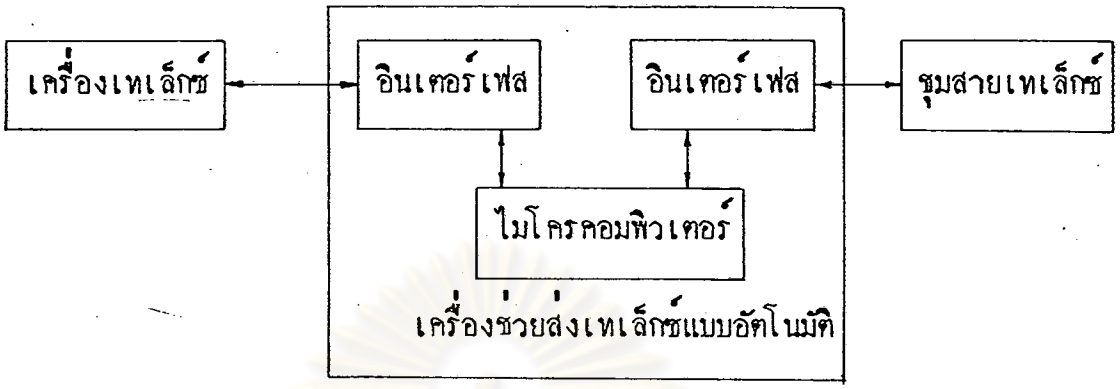
อินเตอร์เฟส 2 ส่วนคือ ส่วนหนึ่งต่อเข้ากับเครื่องเทเล็กซ์เดิม อีกส่วนต่อเข้ากับชุมสายเทเล็กซ์ ด้วยการต่อในลักษณะนี้ ถ้ามองในฐานะตัวเครื่อง เราก็จะกำหนดได้ว่าตัวเครื่องเทเล็กซ์เดิมนั้น จะทำหน้าที่เป็นอุปกรณ์ไอโอ (I/O device) ของตัวเครื่อง โดยทำหน้าที่เป็น KEYBOARD และ PRINTER และมีเอาท์พุท (OUTPUT) อีกด้านหนึ่งเป็นชุมสายเทเล็กซ์ที่จะติดต่อด้วย

ดังนั้นอาจกล่าวโดยสรุปได้ว่า ตัวเครื่องที่นำมาต่อเข้ากับเครื่องเทเล็กซ์นี้ ประกอบด้วย ซีพียู (CPU), หน่วยความจำ (MEMORY), และ ไอโอ (I/O device) ที่จำเป็นต่อการอินเตอร์เฟส โดยมีเครื่องเทเล็กซ์เดิมเป็น TERMINAL ที่ออกคำสั่งในการทำงานให้กับเครื่อง

ลักษณะการทำงานของเครื่อง ตัวเครื่องจะคอยรับคำสั่งจากเครื่องเทเล็กซ์ และส่งผลการทำงานต่อคำสั่งนั้นๆ กลับไปยังส่วน PRINTER หลังจากที่เครื่องได้รับคำสั่งมาแล้ว เครื่องจะดูว่าคำสั่งดังกล่าวนั้นต้องการให้เครื่องทำงานอะไร ตัวเครื่องก็จะเลือกโปรแกรม ที่จะทำงานดังกล่าว มาเอ็กซิคิว (EXECUTE) เพื่อที่จะทำงานตามคำสั่งนั้นๆต่อไป

ความสัมพันธ์ระหว่าง ฮาร์ดแวร์ (HARDWARE) และ ซอฟแวร์ (SOFTWARE)

หัวใจสำคัญของการทำงานของเครื่องนี้ จะเป็นส่วนของซอฟแวร์ โดยที่ตัวฮาร์ดแวร์นั้นจะเป็นแบบ ไมโครคอมพิวเตอร์ แบบธรรมดาโดยทั่วไป และมีส่วนที่ออกแบบพิเศษเพิ่มเติมคือส่วน อินเตอร์เฟส ที่จะทำการติดต่อระหว่างตัวเครื่องไมโครคอมพิวเตอร์กับตัวเครื่องเทเล็กซ์และชุมสายเทเล็กซ์ ซึ่งวงจรดังกล่าวจะทำหน้าที่เปลี่ยนระดับกำลังไฟ จากระดับ TTL เป็น HIGH LEVEL 60 V, 20 mA ที่ใช้ในการติดต่อระหว่างเครื่องเทเล็กซ์ กับชุมสายเทเล็กซ์ ดังแสดงในรูปที่ 2



รูปที่ 2 การติดต่อระหว่างเครื่องเทเล็กซ์กับชุมสายเทเล็กซ์

เครื่องช่วยส่งเทเล็กซ์แบบอัตโนมัตินี้ประกอบด้วย

1. ซีพียู (CPU) ซึ่งเป็นไมโครโปรเซสเซอร์ เบอร์ 8080
2. หน่วยความจำ (MEMORY) ซึ่งเป็น EPROM เบอร์ 2716 และ RAM เบอร์ 6116 อย่างละ 2 ตัวๆละ 2KB
3. อินเตอร์เฟสทางสื่อสาร (COMMUNICATION INTERFACE) ซึ่งเป็น IC เบอร์ 8251 จำนวน 2 ตัว
4. ไทม์เมอร์ชนิด PROGRAMMABLE INTERVAL ซึ่งใช้ IC เบอร์ 8253 จำนวน 1 ตัว

ส่วนของหน่วยความจำ อินเตอร์เฟส และไทม์เมอร์นั้น สร้างขึ้นเอง ซึ่งจะอธิบายรายละเอียดต่อไป

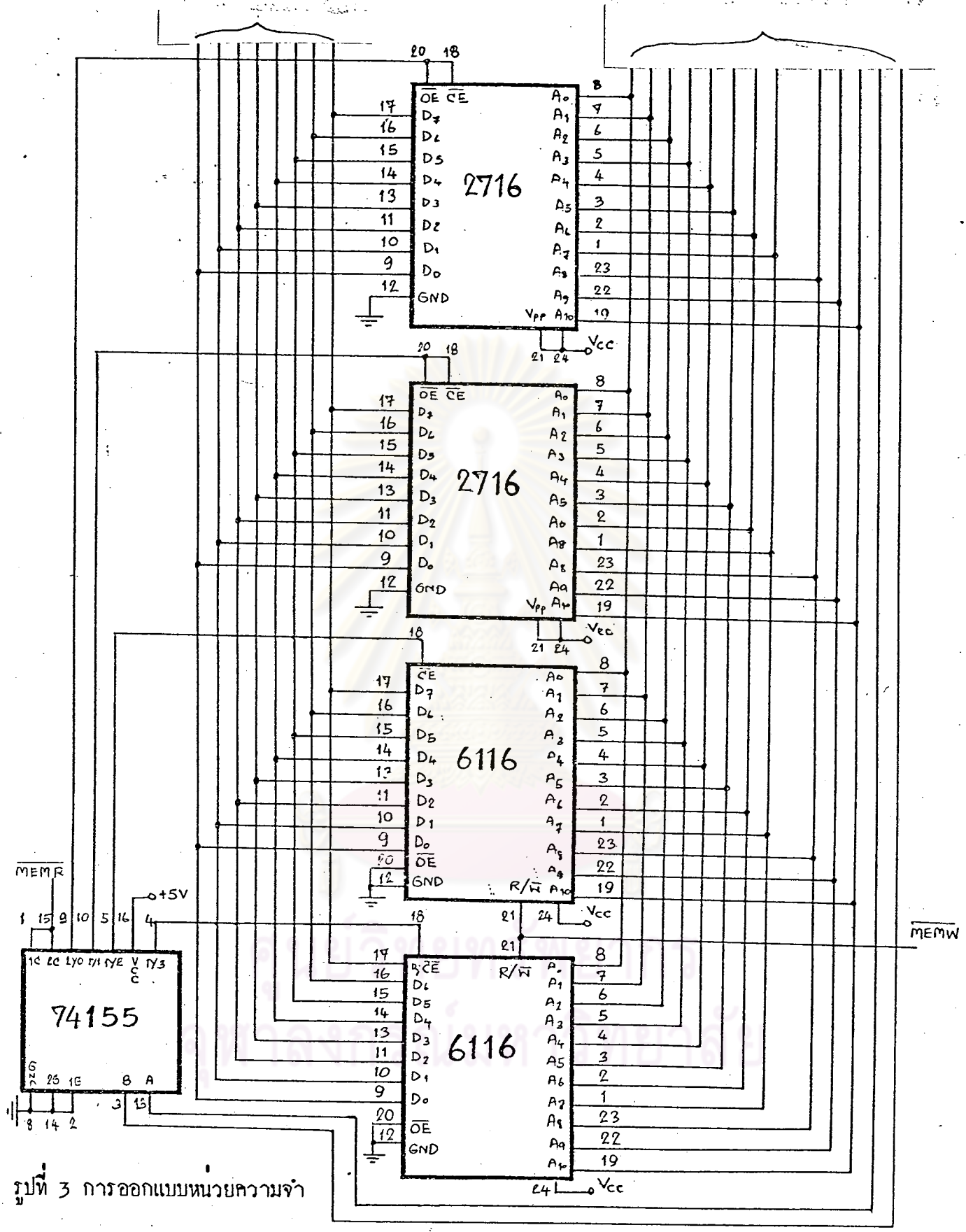
2.2 ส่วนของหน่วยความจำ

ประกอบด้วย EPROM เบอร์ 2716 จำนวน 2 ตัว แต่ละตัวมีขนาด 2K X8 จึงมีความจุ 4K ไบท์

ประกอบด้วย RAM เบอร์ 6116 จำนวน 2 ตัว แต่ละตัวมีขนาด 2K X8 จึงมีความจุ 4K ไบท์

โครงสร้างจะประกอบด้วย บัสข้อมูล (DATA BUS) 8 เส้น และบัสแอดเดรส (ADDRESS BUS) 10 เส้น ต่อขนานกับ EPROM และ RAM ทุกตัว การจะเลือกให้ EPROMหรือ RAM ตัวใดทำงาน ก็ขึ้นอยู่กับสัญญาณ $\overline{CE}$ ซึ่งต่ออยู่กับ EPROM และ RAM ทุกตัว

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย



รูปที่ 3 การออกแบบหน่วยความจำ

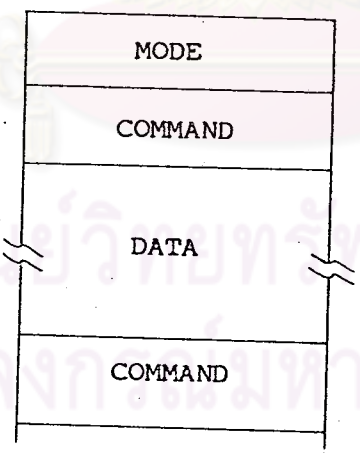
หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

2.3 การทำงานของแผงอินเตอร์เฟส (INTERFACE BOARD)

ในสภาพปกติ หลังจากที่ให้สัญญาณ รีเซ็ทกับ INTERFACE BOARDแล้ว INTERFACE BOARD ดังกล่าวก็พร้อมที่จะเริ่มต้นทำงาน ซึ่งก่อนที่จะให้ทำงานก็จะต้องให้ MODE OPERATION ก่อนซึ่ง MODE OPERATION ดังกล่าวจะใช้บอกว่า INTERFACE BOARD ดังกล่าวจะทำงานกับข้อมูล 5 หรือ 6 UNIT จำนวน STOP BIT และ SAMPLING CLOCK

เมื่อให้ MODE แล้วก็ส่งข้อมูล COMMAND INSTRUCTION เพื่อที่จะ ENABLE การทำงานของวงจรต่างๆ หลังจากนั้น INTERFACE BOARDก็พร้อมที่จะทำงานตามปกติ

ในการทำงานนี้ โปรแกรมจะใช้การอ่าน สเตตัส (STATUS) ของ INTERFACE BOARD เพื่อประกอบในการรับส่งข้อมูล และเพื่อที่จะรู้สภาวะของ LINE ในขณะนั้นๆ และโปรแกรมอาจจะส่ง COMMAND มาเป็นระยะๆเพื่อควบคุมการทำงานให้เป็นไปตามต้องการ



รูปที่ 4 FORMAT ในการทำงาน

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ส่วนประกอบที่สำคัญของ INTERFACE BOARD จะขึ้นอยู่กับการทำงานของ USART โดยที่ข้อมูลที่รับหรือส่ง, COMMAND, MODE และ สเตตัส จะส่งผ่านทางบัสข้อมูล จำนวน 8 เส้น โดยที่ว่าจะเป็นการส่งหรือรับข้อมูลชนิดใดนั้นจะกำหนดโดยสัญญาณจาก CONTROL LINE ซึ่งสัญญาณดังกล่าวจะต่อกันแบบขนาน

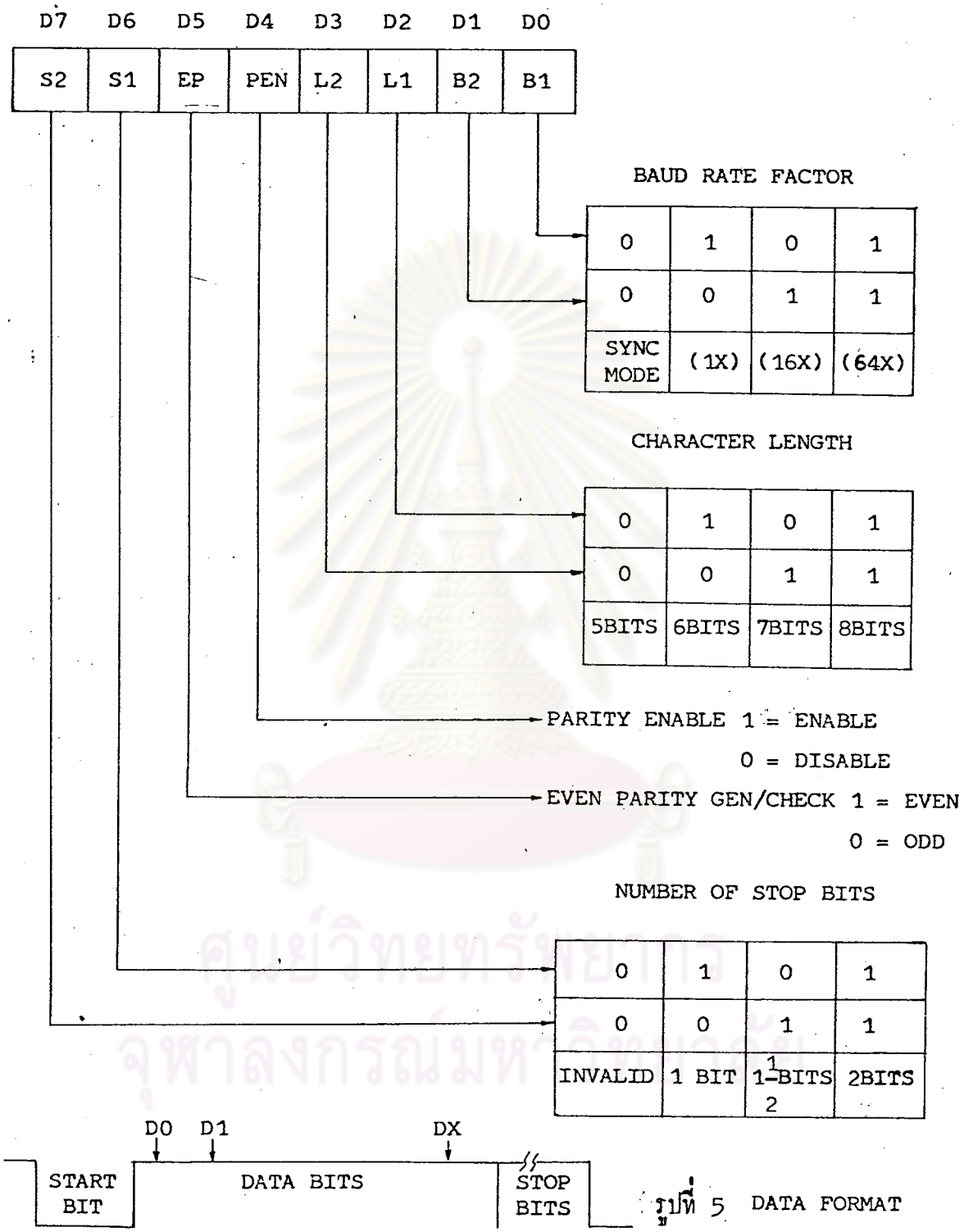
ส่วนภาค สเตตัสและ CONTROL นั้นจะทำหน้าที่ DECODE CONTROL LINE เพื่อที่จะได้รู้ว่าขณะนั้นๆจะมีความต้องการรับหรือส่งข้อมูลชนิดไหน และจะได้ไปควบคุมการทำงานส่วนอื่นๆ ให้มีความสัมพันธ์กับ USART ในการให้ข้อมูลเพิ่มเติม ทั้งนี้เนื่องจากฟังก์ชั่นบางอย่างนั้น USART ไม่สามารถทำได้ เช่นการควบคุมสภาพของ LINE หรือ STATUS LINE ขณะนั้นๆ ฯลฯ

ภาค LINE INTERFACE จะทำหน้าที่เป็น LEVEL TRANSLATOR ซึ่งจะอินเตอร์เฟสระหว่าง LOW LEVEL จาก USART กับ HIGH LEVEL จาก LINE พร้อมกับส่วนที่ทำการตรวจสอบสภาวะของ LINE

MODE INSTRUCTION

BIT	CODE	INFORMATION CONTENT
0	B1	BAUD RATE FACTOR
1	B2	
2	L1	CHARACTER LENGTH
3	L2	
4	PEN	PARITY ENABLE
5	EP	1 = EVEN PARITY, 0 = ODD PARITY
6	S1	STOP BIT
7	S2	

ตารางที่ 1 MODE BIT



รูปที่ 5 DATA FORMAT

BIT	CODE	INFORMATION CONTENT
7	EH	
6	IR	1 =RETURN 8251TO MODE INSTRUCTION
5	RTS	1 = $\overline{\text{RTS}}$ " LOW "
4	ER	1 = ERROR RESET (FE,OE)
3	SBRK	1 = TxD " LOW "
2	RxE	1 = RECEIVE ENABLE
1	DTR	1 = $\overline{\text{DTR}}$ " LOW "
0	TxEN	1 = TRANSMIT ENABLE

ตารางที่ 2 COMMAND BIT

ในตารางที่ 2 แสดงรายละเอียดดังนี้

บิท 6 INTERNAL RESET จะเซ็ทเป็น" 1 " เมื่อต้องการให้ 8251 กลับมาอยู่ในสภาพพร้อมจะรับ MODE INSTRUCTION

บิท 4 ERROR RESETเป็น" 1 " เมื่อต้องการรีเซ็ท FE และ OE

บิท 3 SEND BREAK CHARACTER เซ็ทเป็น " 1 " เมื่อต้องการให้ Tx D LINEเป็น" LOW " และเป็น " 0 " เพื่อให้ LINEเป็น normal operation

บิท 2 RxE บิทนี้จะเซ็ทเป็น" 1 " เมื่อต้องการ ENABLE RECEIVEและเป็น" 0 " เมื่อ DISABLE RECEIVE

บิท 0 TxEN บิทนี้จะเซ็ทเป็น " 1 " เมื่อต้องการ ENABLE TRANSMIT และ
เป็น " 0 " เมื่อ DISABLE TRANSMIT

บิท 5 RTS บิทนี้เป็น " 1 " จะทำให้เอาท์พุทที่ $\overline{\text{RTS}}$ เป็น " 0 "

BIT	CODE	INFORMATION CONTENT
7	DSR	
6	SYNDET	
5	FE	1 = FRAMING ERROR
4	OE	1 = OVERRUN ERROR
3	PE	1 = PARITY ERROR IS DETECTED
2	TxE	1 = NO CHARACTER TO TRANSMIT
1	RxRDY	1 = RECEIVE BUFFER FULL
0	TxRDY	1 = TRANSMIT HOLDING REGISTER EMPTY

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

DSR	SYNDET	FE	OE	PE	TxE	RxRDY	TxRDY
D7	D6	D5	D4	D3	D2	D1	DO

ตารางที่ 3 STATUS BIT

TERMINATOR STATUS WORD

บิท 5 FRAMING ERROR(FE)จะถูกเซ็ทให้เป็น " 1 " เมื่อไม่ได้รับ VALID STOP BIT และจะถูกรีเซ็ทด้วย ER bit (error reset)จาก COMMAND WORD

บิท 4 OVERRUN ERROR (OE) จะถูกเซ็ทเมื่อตัวอักษร (CHARACTER) ที่ได้รับมาก่อนหน้าไม่ถูกอ่านออกไปจากบัฟเฟอร์ ซึ่งจะมีผลให้ตัวอักษรที่ได้รับมาก่อนหายไป OE นี้จะถูกรีเซ็ทโดย ER bit

บิท 3 PARITY ERROR จะถูกเซ็ทเมื่อพบว่าเกิด PARITY ERROR และจะถูกรีเซ็ทโดย ER bit

บิท 2 TxE บิทนี้จะถูกเซ็ทเมื่อเกิดสภาวะที่ไม่มีตัวอักษร ที่จะส่งและถูกรีเซ็ทเมื่อได้รับตัวอักษรมาพร้อมที่จะให้ส่ง

บิท 1 Rx RDY บิทนี้จะเซ็ทเมื่อมีตัวอักษรพร้อมที่จะส่งให้ระบบ และ Rx ENABLE จะรีเซ็ท Rx RDY ให้พร้อมที่จะทำงานและเมื่อได้รับสัญญาณ READ DATA Rx RDYจะถูกรีเซ็ท

บิท 0 Tx RDY บิทนี้จะถูกเซ็ทเมื่อ Tx BUFFER พร้อมที่จะรับตัวอักษรที่จะส่งออกได้ และจะถูกรีเซ็ทเมื่อได้รับสัญญาณ WRITE DATA

อธิบายการทำงานของ INTERFACE BOARD

เนื่องจากอุปกรณ์ที่สำคัญใน INTERFACE BOARD คือ USART ซึ่งทำหน้าที่ต่างๆ ที่จำเป็นๆ ส่วนใหญ่การทำงานของ USART นี้จะทำงานภายใต้การควบคุมของ โปรแกรม จึงทำให้ลดส่วนฮาร์ดแวร์ ที่เกี่ยวข้องไปได้เกือบทั้งหมด

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

013330

หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

แต่มีฟังก์ชั่นอยู่ 2 อย่างที่ USART นี้ไม่ได้มีอยู่คือ

1. การตรวจสอบสภาวะของ LINE
2. การควบคุมสภาวะของ LINE

ซึ่งการเพิ่มเติม ฟังก์ชั่นดังกล่าวจะต้องใช้ส่วนประกอบอื่นมาประกอบด้วย ส่วนแรกได้แก่ CONTROL DECODE เนื่องจากการรับส่งข้อมูลแบ่งออกได้เป็น 4 ชนิดด้วยกัน คือ

1. CHARACTER ที่จะส่งออก (DATA BUS $\Longrightarrow$ USART)
2. CHARACTER ที่รับเข้า (DATA BUS $\Longleftarrow$ USART)
3. STATUS (DATA BUS $\Longleftarrow$ STATUS)
4. COMMAND (DATA BUS $\Longrightarrow$ COMMAND)

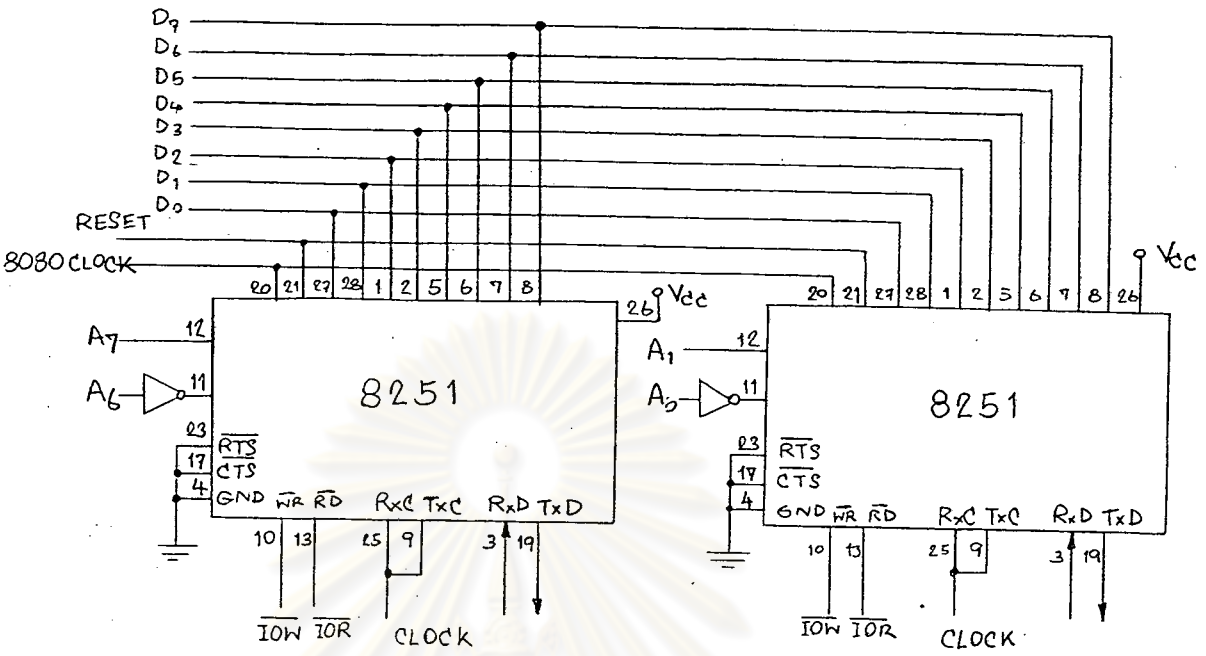
และการที่จะกำหนดว่าเป็นการรับส่งข้อมูลชนิดใด นั้นจะขึ้นอยู่กับสัญญาณ 3 ชนิด คือ

1. C/$\overline{D}$ COMMAND และ DATA
2. $\overline{RD}$ READ
3. $\overline{WR}$ WRITE

ส่วนการที่จะให้ 8251 ตัวใดทำงานนั้นจะใช้สัญญาณ CHIP ENABLE($\overline{CS}$) เป็นส่วนควบคุม

ตารางที่ 4 แสดงสัญญาณควบคุม

C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	0	1	0	8251 DATA $\Longrightarrow$ DATA BUS
0	1	0	0	DATA BUS $\Longrightarrow$ 8251 DATA
1	0	1	0	STATUS $\Longrightarrow$ DATA BUS
1	1	0	0	DATA BUS $\Longrightarrow$ CONTROL
X	1	1	0	DATA BUS $\Longrightarrow$ 3-STATE
X	X	X	1	DATA BUS $\Longrightarrow$ 3-STATE



ติดต่อด้านเครื่องผู้ใช้งาน ติดต่อด้านชุมสายเทเล็กซ์

รูปที่ 6 การออกแบบอินเตอร์เฟส

แสดงหน้าที่การทำงาน

D0-D7 เป็น บัสข้อมูลชนิด 2 ทาง ซึ่งข้อมูลของ CHARACTER ที่จะรับหรือส่ง, สเตตัส และ COMMAND จะส่งผ่านทางนี้ทั้งหมด สัญญาณทั้ง 8 เส้นนี้จะ COMMON ร่วมกันหมด

C/$\overline{D}$ CONTROL/ DATA เป็น 1 ใน 3 สัญญาณ CONTROL โดยที่สัญญาณนี้จะเป็นตัวบอกว่าข้อมูลที่จะรับส่งในบัสข้อมูลขณะนี้เป็น พวกข้อมูล หรือ พวก CONTROL โดยจะทำงานร่วมกับ $\overline{RD}$ และ $\overline{WR}$ เพื่อกำหนดทิศทางและชนิดของ

การส่งโดยที่ C/$\overline{D}$ เป็น " LOW " จะเป็นการรับส่งข้อมูล และเป็น" HIGH " จะเป็นการรับส่ง COMMAND และ STATUS

$\overline{RD}$ READ DATA และ สเตตัส สัญญาณนี้จะทำงานร่วมกับ C/$\overline{D}$ ถ้าสัญญาณนี้เป็น " LOW " จะบอกถึงว่าข้อมูลจะถูกส่งจาก 8251 ไปยังบัสข้อมูล ส่วนชนิดของข้อมูลจะขึ้นอยู่กับสัญญาณ C/$\overline{D}$ ถ้า $\overline{RD}$ เป็น " HIGH " จะไม่มีผลในการส่งข้อมูลไปบัสข้อมูล

$\overline{WR}$ WRITE DATA และ COMMAND สัญญาณนี้จะทำงานร่วมกับ C/$\overline{D}$ ถ้าสัญญาณนี้เป็น " LOW " จะบอกถึงว่าข้อมูลจะส่งจากบัสข้อมูลมายัง 8251 และถ้าเป็น " HIGH " จะไม่มีผลในการรับข้อมูลจากบัสข้อมูล

$\overline{CS}$ CHIP ENABLE สัญญาณนี้จะใช้เลือกว่า 8251 ตัวใดจะเป็นตัวที่ ทำงาน

CLK CLOCK PULSE (TTL) สัญญาณ CLOCK นี้จะใช้ในการทำงานภายในของ USART โดยปกติใช้ ϕ_2 CLOCK จาก ซีพียู

RESET ใช้เมื่อ รีเซ็ทการทำงานของ 8251 ให้อยู่ในสภาพ IDLE

$\overline{TxC}$ TRANSMITTER CLOCK เป็นสัญญาณ CLOCK ที่ใช้ในการส่งข้อมูล ซึ่งความถี่สามารถเลือกได้เป็น 1, 16 หรือ 64 เท่าของ BAUD RATE โดยจะใช้ CLOCK ความถี่เท่าใดนั้นก็เซ็ทใน MODE INSTRUCTION

$\overline{RxC}$ RECEIVE CLOCKเช่นเดียวกับ $\overline{TxC}$

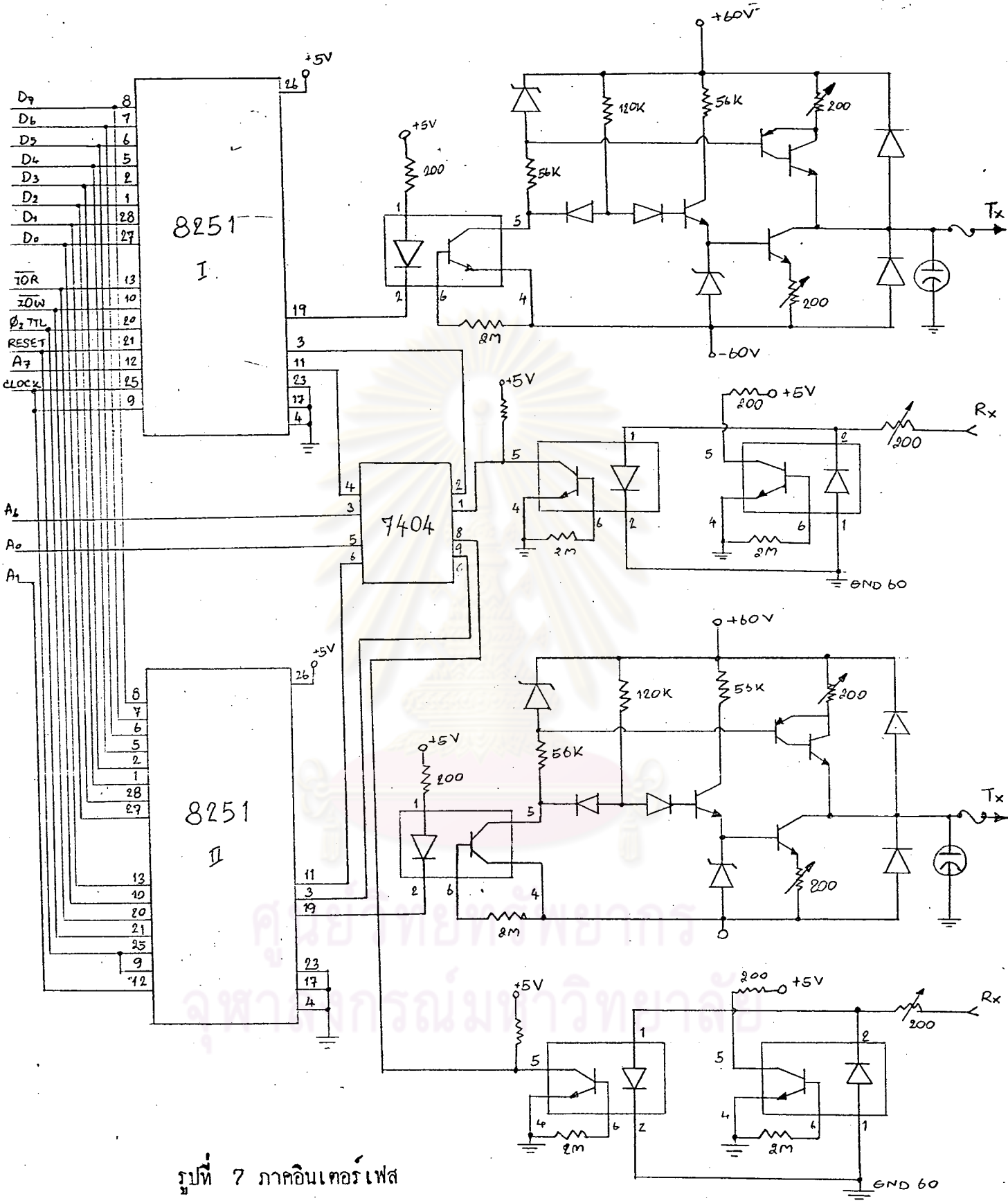
GND GROUND

VCC + 5 VOLT SUPPLY

+60 V, -60 V สำหรับใช้ส่งสัญญาณออก LINE

LINE Rx RECEIVE LINE

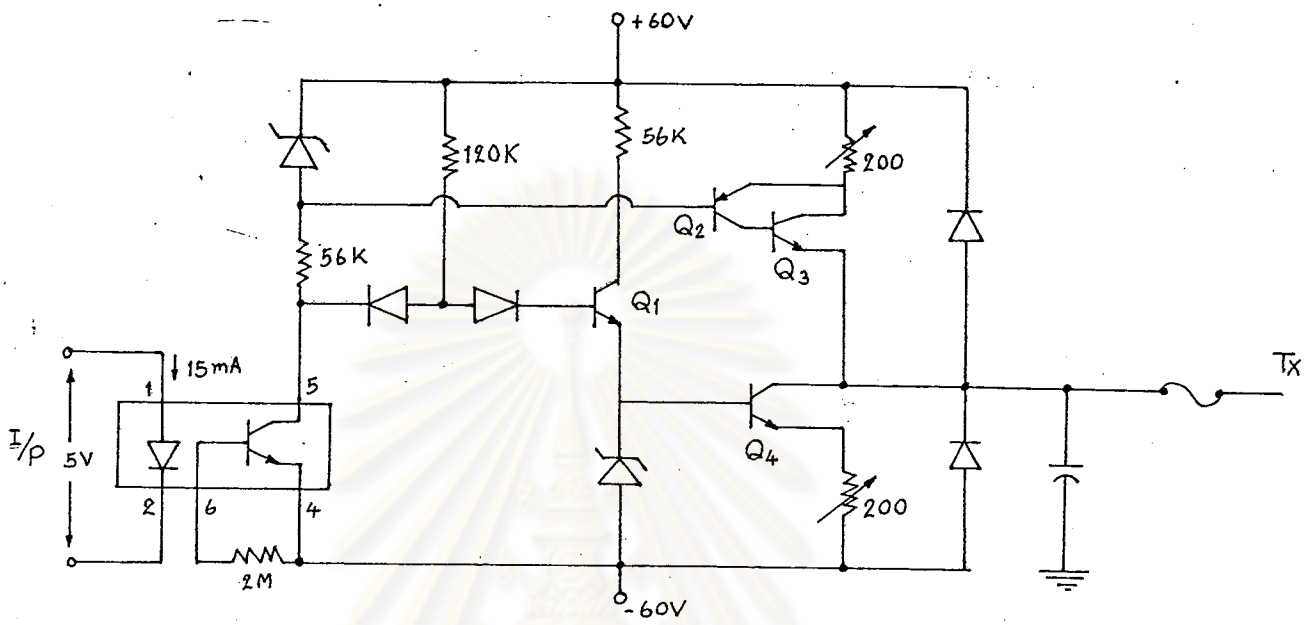
LINE Tx TRANSMIT LINE



รูปที่ 7 ภาคอินเตอร์เฟส

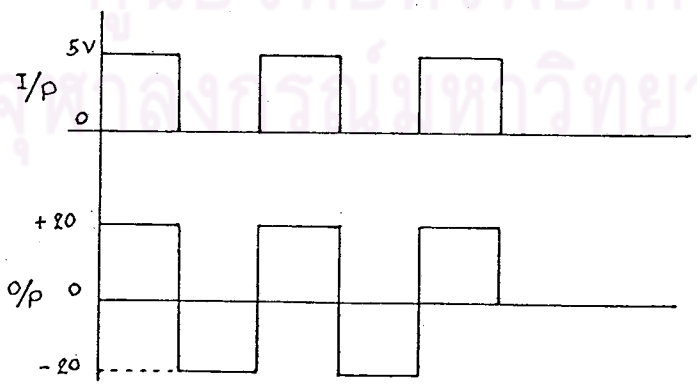
ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย

ส่วนของ LINE INTERFACE ทำหน้าที่เปลี่ยนสัญญาณจาก LOW LEVEL (TTL) ให้เป็นสัญญาณชนิด HIGH LEVEL ±20 mA และ Regulate current



รูปที่ 8 ส่วนของLINE INTERFACE

ตามรูป การต่อ Q2 และ Q3 ก็เพื่อให้ใช้เป็น PNP แบบ HIGH VOLTAGE และ HIGH POWER เนื่องจาก PNP แบบ HIGH VOLTAGE และ HIGH POWER จริงๆ นั้น มีราคาแพง วงจรนี้จะทำการเปลี่ยนแปลงสัญญาณ จาก LOW LEVEL เป็น HIGH LEVEL และ Regulate current ตามรูป



รูปที่ 9 รูปสัญญาณที่ส่วน LINE INTERFACE

หอสมุดกลาง สถาบันวิทยบริการ จุฬาลงกรณ์มหาวิทยาลัย

การทำงานในวงจรนี้ จะให้เอาท์พุท ± 20 mA Regulate current โดยแบ่งการทำงานออกเป็น 2 ส่วน คือ ส่วนที่ทำงานในช่วง +20 mA และช่วง -20mA

การทำงานในช่วง +20 mA

เมื่อมีอินพุท +5 V ผ่าน OPTO จะทำให้ OPTO ทำงานจะมีไฟ -60 V ผ่าน OPTO ไปยังเบสของ Q2 และ Q3 ทำให้เบสของ Q2 และ Q3 เป็นลบ ทำให้ Q2 และ Q3 ทำงาน จะมีกระแสไฟบวก ไหลผ่านโหลด กระแสนี้จะคงที่ เพราะว่า $I = \frac{E}{R}$

E เป็น กำลังไฟที่ ZENER DIODE ซึ่งมีค่าคงที่

R เป็นค่าความต้านทานที่ตั้งค่าไว้ ไม่เปลี่ยนแปลง

ดังนั้นค่ากระแสที่ได้จึง คงที่

ถ้าปรับค่า R200 Ω จะมีผลต่อกระแสทางเอาท์พุท ดังนี้

R66 Ω ได้กระแสคงที่ ± 20 mA

R100 Ω ได้กระแสคงที่ ± 15 mA

R150 Ω ได้กระแสคงที่ ± 10 mA

ค่า RL จะมีผลต่อ กระแสทางเอาท์พุทด้วย

เมื่อ RL มีค่า $\leq$ 400 Ω กระแสจะเพิ่มขึ้น

RL มีค่า $\geq$ 2.4K Ω กระแสจะลดลง

แต่กระแสทางเอาท์พุทจะคงที่เมื่อ RL อยู่ในระหว่างค่าดังกล่าว

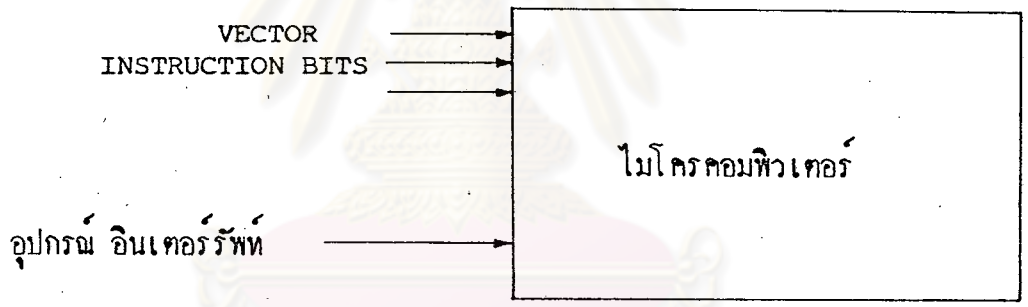
การทำงานในช่วง -20 mA

เมื่อไม่มีอินพุทผ่าน OPTO ทำให้ OPTO ไม่ทำงาน ที่เบสของ Q2 และ Q3 ไม่เป็นลบ Q2 และ Q3 จะไม่ทำงาน ขณะที่เบสของ Q1 เป็นบวก Q1 จะทำงาน ทำให้ Q4 ทำงานด้วย จะมีกระแสไฟลบจากไฟ -60 V ผ่าน Q4 ไปยังโหลด และกระแสนี้จะคงที่สม่ำเสมอ เหมือนการทำงานในช่วงบวก

2.4 การอินเตอร์รัพท์ที่ใช้กับไมโครเมอร์

อินเตอร์รัพท์ (INTERRUPT) หมายถึง การขัดจังหวะการทำงานของ ซีพียู เพื่อให้ ซีพียู หยุดการทำงานของโปรแกรมที่กำลังทำอยู่ เมื่อได้รับสัญญาณ อินเตอร์รัพท์ ซีพียู จะทำงานตามคำสั่งที่กำลังทำอยู่จนจบคำสั่งนั้นเสียก่อน จึงจะไปบริการให้กับงานที่ส่งสัญญาณอินเตอร์รัพท์เข้ามา และเมื่อทำงานที่อินเตอร์รัพท์เสร็จแล้ว ก็จะกลับไปทำงานที่โปรแกรมเดิมต่อไป การอินเตอร์รัพท์มีลักษณะคล้ายกับการที่ซีพียูกระโดดไปทำงานที่โปรแกรมย่อย จะแตกต่างกันก็คือ การกระโดดนี้ไปด้วยคำสั่ง CALL หรือ คำสั่ง CALL ที่มีเงื่อนไข แต่การอินเตอร์รัพท์จะกระโดดไปทำงานในอีกโปรแกรมหนึ่ง เมื่อซีพียูได้รับสัญญาณอินเตอร์รัพท์จากอุปกรณ์ภายนอก

การอินเตอร์รัพท์ที่ใช้ในงานนี้ ใช้วิธีการที่เรียกว่า เวคเตอร์อินเตอร์รัพท์ (VECTORED INTERRUPT) แสดงในรูปที่ 13



รูปที่ 10 เวคเตอร์อินเตอร์รัพท์

สัญญาณอินเตอร์รัพท์จะส่งมาที่ไมโครคอมพิวเตอร์ เพียงเส้นเดียว เมื่อถูกอินเตอร์รัพท์ วงจรควบคุมการอินเตอร์รัพท์จะส่งคำสั่ง RST (RESTART) เข้าไปทางบัสข้อมูล แล้วนำข้อมูลที่ได้ส่งไปที่รีจิสเตอร์ ถอดรหัสคำสั่ง (INSTRUCTION REGISTER) และดำเนินการต่อไป

แฟลกของการอินเตอร์รัพท์ (INTERRUPT FLAG)
เป็นฟลิบฟลอบที่อยู่ภายในซีพียู ฟลิบฟลอบนี้จะถูกเซ็ทให้เป็น "1" โดยคำสั่ง EI เพื่อให้ซีพียู สามารถรับสัญญาณอินเตอร์รัพท์ได้ แฟลกภายในจะถูกรีเซทให้เป็น "0" โดยคำสั่ง DI หรือจะถูกรีเซ็ททุกครั้ง เมื่อตอบสนองต่อการอินเตอร์รัพท์นั้นแล้ว

ลำดับการทำงานของการอินเตอร์รัพท์ของไมโครโปรเซสเซอร์ เบอร์ 8080

1. จะต้องมีคำสั่ง EI ซึ่งเขียนไว้ในโปรแกรมว่า โปรแกรมนั้นยอมให้เกิดการอินเตอร์รัพท์ได้

2. จะต้องมีสัญญาณ INTซึ่งเป็นสัญญาณที่บอกถึงความต้องการในการอินเตอร์รัพท์จากอุปกรณ์ภายนอกเสียก่อน

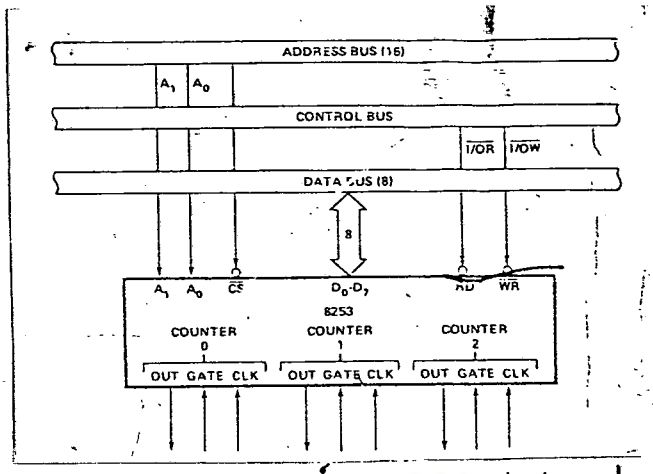
3. ซีพียู จะส่งสัญญาณ INTA (INTERRUPT ACKNOWLEDGE) ไปบอกอุปกรณ์ภายนอกว่าขณะนั้น ซีพียู พร้อมแล้วที่จะบริการอุปกรณ์ตัวนั้น ให้ส่งคำสั่ง RST เข้าไปทางบัสข้อมูลได้ และแฟลกของการอินเตอร์รัพท์จะถูกรีเซ็ท

4. ซีพียู จะเก็บแอดเดรสของโปรแกรมที่กำลังทำงานอยู่ไว้โดยเก็บไว้ที่สแตค เพื่อให้ ซีพียู สามารถกลับมาทำงานต่อในโปรแกรมเดิมได้ หลังจากบริการอินเตอร์รัพท์นั้นเสร็จเรียบร้อยแล้ว

5. ซีพียู จะกระโดดไปทำงานในโปรแกรมย่อยอินเตอร์รัพท์ตามคำสั่ง RST ซึ่งได้ส่งมาทางบัสข้อมูล

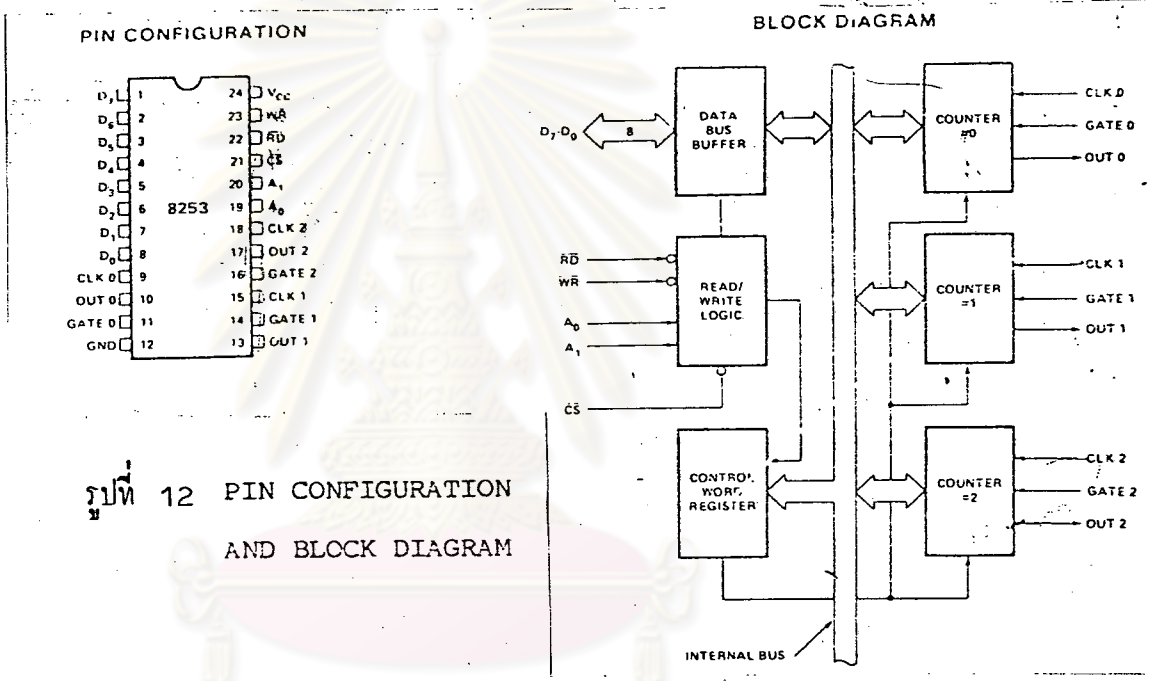
6. ในโปรแกรมย่อยอินเตอร์รัพท์ จะมีคำสั่ง RET หรือ RET แบบมีเงื่อนไข เพื่อบอกให้ ซีพียู กลับมาสู่โปรแกรมเดิมหลังจากบริการงานที่อินเตอร์รัพท์เสร็จแล้ว

เนื่องจาก 8253 ประกอบด้วย เคาว์เตอร์ 3 ตัว มีลักษณะการทำงานเหมือนกัน เคาว์เตอร์แต่ละตัวมีขนาด 16 บิท และเป็นชนิด นับถอยหลัง (DOWN COUNTER) สามารถนับค่าได้ ทั้งแบบ BINARY และ BCD การใช้งาน 8253 จึงต้องกำหนดค่าต่างๆหลายอย่าง ให้แก่ CONTROL WORD REGISTER โดยผ่านทางบัสข้อมูล เพื่อให้ควบคุม MODE ในการทำงานของเคาว์เตอร์แต่ละตัว



รูปที่ 11 การอินเตอร์เฟส

การอินเตอร์เฟส 8253 มีวิธีการต่อใช้งานของอุปกรณ์ ตามรูปที่ 11 การทำงานจะควบคุมโดยวิธีการของ ซอฟแวร์



รูปที่ 12 PIN CONFIGURATION AND BLOCK DIAGRAM

การอินเตอร์เฟสกับ 8253 โดยใช้ บัฟเฟอร์ของบัสข้อมูล ซึ่งเป็นรีจิสเตอร์ขนาด 8 บิท ชนิด 3 สภาวะ ข้อมูลที่ส่งออกหรือรับเข้า ควบคุมโดย คำสั่ง INPUT หรือ OUTPUT ของ 8080 โดยบัฟเฟอร์ของบัสข้อมูลจะมีหน้าที่แยกบัสข้อมูลภายใน 8080 ออกจากบัสข้อมูลภายนอก และยังมีหน้าที่รับส่งข้อมูลอีก 3 อย่าง คือ การโปรแกรม MODE ของ 8253 การใส่ค่าที่ต้องการให้ รีจิสเตอร์นับ และ การอ่านค่าที่กำลังนับอยู่

ลอจิกการอ่าน/เขียน รับข้อมูลจากบัสและให้ กำเนิดสัญญาณควบคุม เพื่อให้อุปกรณ์ต่างๆ ทำงาน และจะทำงานร่วมกับสัญญาณ $\overline{CS}$ (CHIP SELECT)

$\overline{RD}$ (Read)เมื่อสัญญาณที่ขา $\overline{RD}$ เป็น LOW จะทำให้ ซีพียูได้รับข้อมูลจากเคาว์เตอร์ของ 8253

$\overline{WR}$ (Write)เมื่อสัญญาณ $\overline{WR}$ เป็น LOW จะเป็นสภาวะที่ ซีพียูส่งข้อมูลเกี่ยวกับ MODE หรือค่าที่โหลดให้แก่เคาว์เตอร์ของ 8253

A_0, A_1 อินพุททั้งสองนี้จะต่ออยู่กับ บัสแอดเดรส จะใช้เป็นสัญญาณเพื่อเลือกเคาว์เตอร์ตัวหนึ่งตัวใดใน 3 ตัว ให้ทำงาน และให้ข้อมูลแก่ CONTROL WORD REGISTER เพื่อเลือก MODE ในการทำงาน

$\overline{CS}$ (Chip Select)

8253 จะทำงานเมื่อ สัญญาณ $\overline{CS}$ อยู่ในสภาวะเป็น LOW และสัญญาณ $\overline{CS}$ จะไม่มีผลรบกวนการทำงานของเคาว์เตอร์

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A_1	A_0	
0	1	0	0	0	Load Counter No. 0
0	1	0	0	1	Load Counter No. 1
0	1	0	1	0	Load Counter No. 2
0	1	0	1	1	Write Mode Word
0	0	1	0	0	Read Counter No. 0
0	0	1	0	1	Read Counter No. 1
0	0	1	1	0	Read Counter No. 2
0	0	1	1	1	No-Operation 3-State
1	X	X	X	X	Disable 3-State
0	1	1	X	X	No-Operation 3-State

ตารางที่ 5 การ CONTROL WORD

CONTROL WORD REGISTER

CONTROL WORD REGISTER จะรับข้อมูลจากบัฟเฟอร์ของบัสข้อมูล และเก็บไว้ในรีจิสเตอร์ โดยข้อมูลที่เก็บไว้จะมีหน้าที่ควบคุม MODE ในการทำงานของเคาว์เตอร์ เช่นเลือกวิธีการนับ แบบ BINARY หรือ BCD เป็นต้น การกำหนดค่าแก่รีจิสเตอร์เพื่อให้นับค่าตามที่ต้องการนั้น ต้องกำหนดค่าไว้ที่ CONTROL WORD REGISTER เสียก่อน

การโปรแกรม 8253

เราใช้ SOFTWARE กำหนด MODE ต่างๆสำหรับเคาว์เตอร์แต่ละตัว โดยเขียนโปรแกรมในการควบคุม ให้แก่ CONTROL WORD REGISTER และ ในสภาวะ $A_0 A_1 = 11$

ตารางที่ 6 CONTROL WORD FORMAT

SC1	SC0	RL1	RL0	M2	M1	M0	BCD
D7	D6	D5	D4	D3	D2	D1	D0

ความหมายของส่วนต่างๆ

ตารางที่ 7 SC - Select Counter

SC1	SC0	
0	0	แสดงการเลือก เคาว์เตอร์ # 0
0	1	แสดงการเลือก เคาว์เตอร์ # 1
1	0	แสดงการเลือก เคาว์เตอร์ # 2
1	1	ไม่ใช้ (illegal)

ตารางที่ 8 RL - Read/Load

RL1	RL0	
0	0	บันทึกค่าการนับครั้งหลังสุด
0	1	การใส่หรืออ่านค่าในไบท์ที่มีนัยสำคัญสูงสุด (most significant byte)
1	0	การอ่านหรือใส่ค่าในไบท์ที่มีนัยสำคัญต่ำสุด (least significant byte)
1	1	การอ่านหรือใส่ค่าในไบท์ที่มีนัยสำคัญต่ำสุดก่อน แล้วจึงอ่านหรือใส่ค่าในไบท์ที่มีนัยสำคัญสูงสุดทีหลัง

ตารางที่ 9 M-MODE

M2	M1	M0	MODE
0	0	0	0
0	0	1	1
X	1	0	2
X	1	1	3
1	0	0	4
1	0	1	5

ตารางที่ 10 BCD

0	การนับแบบ BINARY 16 บิท
1	การนับแบบ BCD (BINARY Coded Decimal) 4 decades

MODE 0 : อินเตอร์รัพท์เมื่อการนับครบตามค่าที่กำหนดไว้

(interrupt on terminal count)

สัญญาณเอาท์พุทเป็น LOW หลังจากกำหนดค่า MODE แล้ว และเมื่อใส่ค่าให้แก่รีจิสเตอร์แล้ว สัญญาณเอาท์พุทก็ยังคงเป็น LOW ขณะที่เคาว์เตอร์ก็จะนับไปเรื่อยๆ เมื่อนับถึงค่าสุดท้าย สัญญาณเอาท์พุทจึงจะ HIGH และจะ HIGH จนกว่าจะมีการใส่ค่าแก่รีจิสเตอร์ และกำหนด MODE ใหม่

การใส่ค่าแก่ เคาว์เตอร์ระหว่างการนับ ผลที่ตามมาคือ

1. ใส่ค่าไบท์แรก เคาว์เตอร์จะหยุดการนับ
2. ใส่ค่าไบท์ที่ 2 เป็นการเริ่มต้นการนับ

สัญญาณที่ GATE เป็น HIGH จะทำให้เกิดการนับ และจะไม่นับเมื่อเป็น LOW

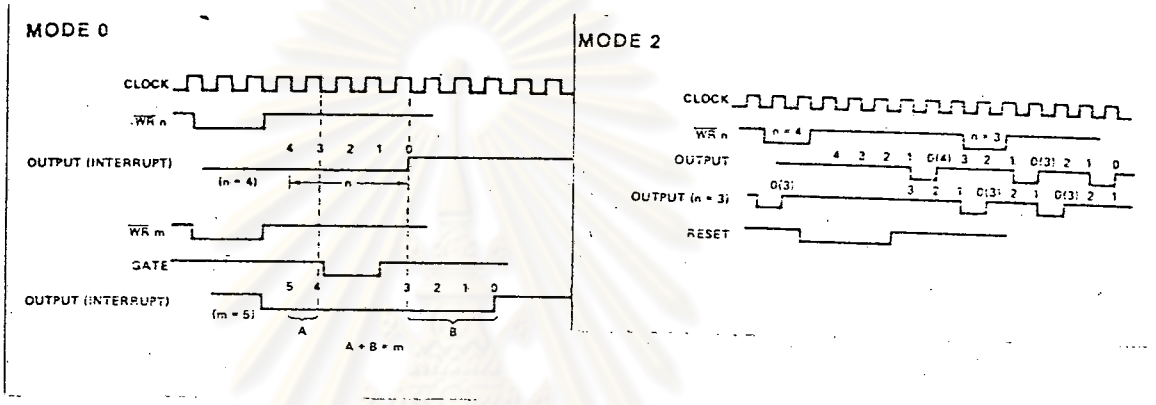
MODE 2 : การกำเนิดสัญญาณคาบเกี่ยว

(rate generator)

สัญญาณเอาท์พุท จะเป็น LOW เมื่อจบช่วงหนึ่งๆของสัญญาณคล็อกอินพุท และช่วงเวลาจากคลื่นเอาท์พุทหนึ่งไปยังอีกคลื่นหนึ่ง จะเท่ากับจำนวนของคล็อกอินพุทที่เก็บ

ไว้ในรีจิสเตอร์ ถ้ารีจิสเตอร์ถูกเปลี่ยนค่าใหม่ในช่วงเวลาของคลื่นเอาท์พุท ก็จะไม่มีผลกระทบต่อช่วงเวลาขณะนั้น

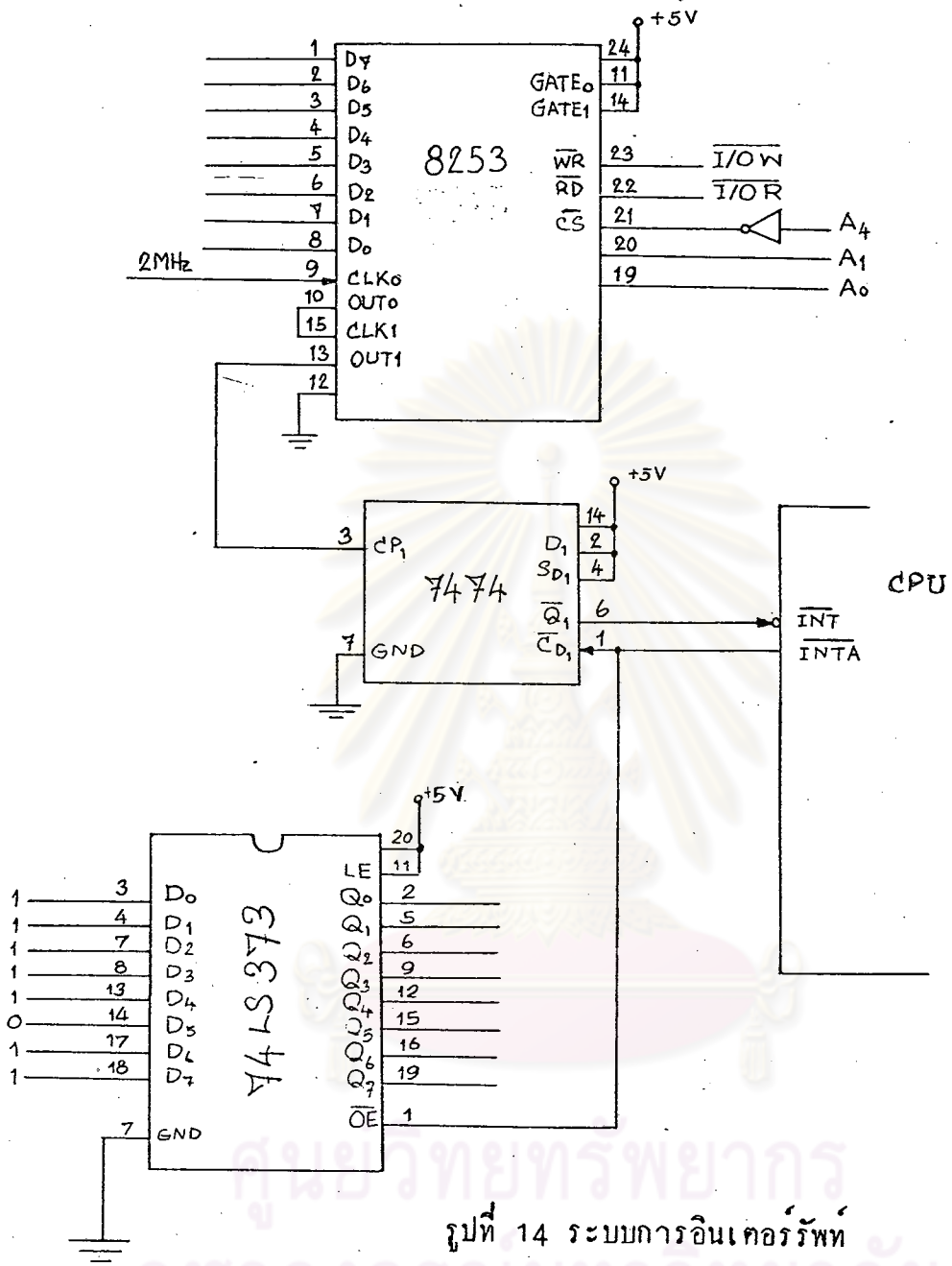
เมื่อสัญญาณ GATE เป็น LOW สัญญาณเอาท์พุทจะเป็น HIGH และเมื่อ สัญญาณ GATE เป็น HIGH เคาว์เตอร์ก็จะเริ่มนับ



รูปที่ 13 แสดง MODE

การนำ 8253 ไปใช้งานให้ทำหน้าที่เป็น REAL TIME CLOCK

เนื่องจาก ซีพียูใช้ไมโครโปรเซสเซอร์เบอร์ 8080 ซึ่งมีความถี่คล็อก 2 MHz อยู่แล้ว จึงเป็นความสะดวกอย่างมากที่จะนำความถี่นี้มาใช้ เพื่อให้ทำหน้าที่เป็น REAL TIME CLOCK โดยจะใช้วิธี หารความถี่ 2 MHz ให้เหลือความถี่ 1 Hz ด้วยเคาว์เตอร์จำนวน 2 ตัวที่มีอยู่ก็ย่อมทำได้ ในที่นี้จะใช้เคาว์เตอร์ ※ 0 และ ※ 1 ซึ่งอันที่จริงจะเลือกใช้เคาว์เตอร์ตัวไหนก็ได้ ตามหลักการนี้ จะโหลดค่า 2000 ให้แก่เคาว์เตอร์ ※ 0 เพื่อให้หารความถี่ 2 MHz จนเหลือความถี่ 1000 Hz แล้วจึงโหลดค่า 1000 ให้แก่เคาว์เตอร์ ※ 1 เพื่อหารความถี่ดังกล่าว ก็จะได้ความถี่ 1 Hz ซึ่งความถี่ 1 Hz นี้จะใช้เป็นสัญญาณไปอินเตอร์รัพท์ ซีพียูต่อไป



รูปที่ 14 ระบบการอินเตอร์รัพท์

ศูนย์วิทยทรัพยากร
จุฬาลงกรณ์มหาวิทยาลัย