บทที่ 2
ทฤษฎีพื้นฐานที่เกี่ยวข้อง

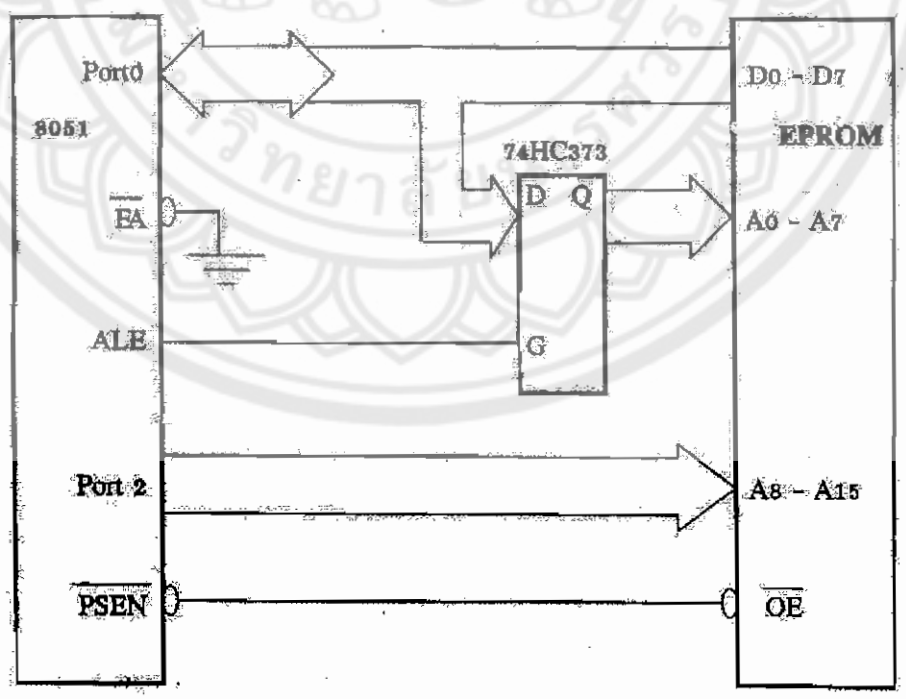
2.1 สถาปัตยกรรมและการควบคุมไมโครคอนโทรลเลอร์ตระกูล MCS – 51

2.1.1 การจัดหน่วยความจำ [1]

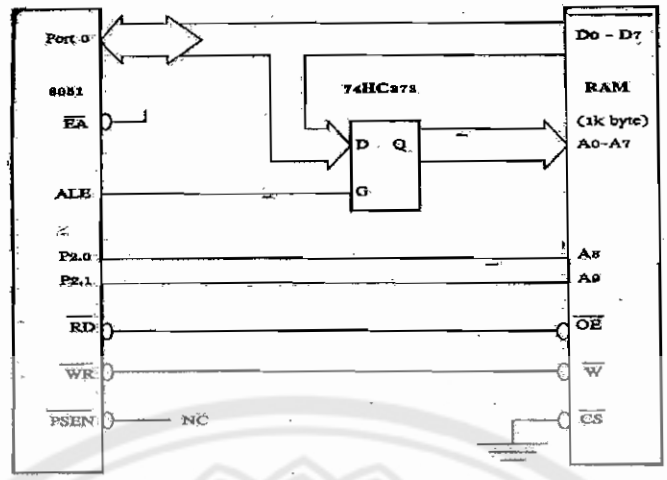
ในระบบของ MCS-51 จะมีการแบ่งหน่วยความจำเหมือนกับซีพียูทั่วไปคือ จะแบ่งเป็น 2 ลักษณะตามชนิดของข้อมูลที่เก็บดังนี้

- หน่วยความจำข้อมูล (data memory)
- หน่วยความจำโปรแกรม (program memory)

ถ้าจะพูดกันง่าย ๆ หน่วยความจำข้อมูลก็หมายถึงหน่วยความจำส่วนที่เป็นแรม (ram) ซึ่งเราสามารถอ่านหรือเขียนข้อมูลเปลี่ยนแปลงได้ตลอดเวลา แต่ไม่สามารถปรับโปรแกรมบนหน่วยความจำส่วนนี้ได้ ส่วนหน่วยความจำโปรแกรมจะหมายถึงหน่วยความจำที่อ่านได้อย่างเดียว (Rom) ซึ่งบรรจุโปรแกรมที่จะให้ MCS-51 ทำงาน โดยหน่วยความจำทั้ง 2 ประเภทนี้จะถูกแยกออกจากกันด้วยคำสั่งทางซอฟต์แวร์และลักษณะการติดต่อทางฮาร์ดแวร์ด้วย กล่าวคือ จะมีคำสั่งเฉพาะสำหรับการติดต่อกับหน่วยความจำชนิดใดชนิดหนึ่งและจัดสัญญาณสโตรบในการติดต่อกับหน่วยความจำแต่ละชนิดแยกต่างหากกันด้วย ดังรูปที่ 2.1 และรูปที่ 2.2

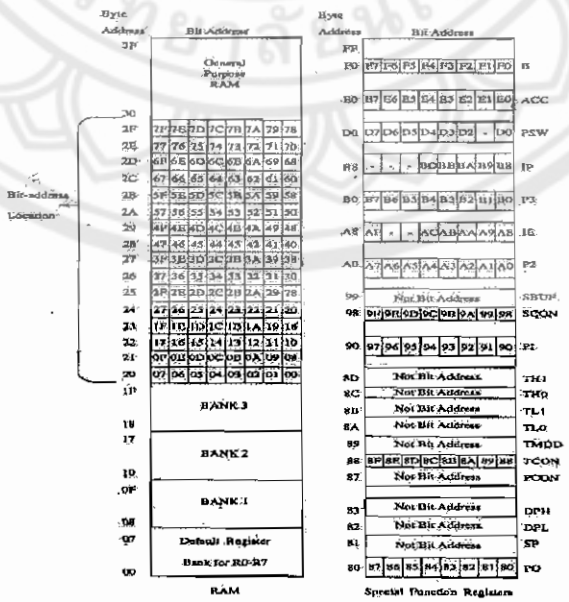


รูปที่ 2.1 การเชื่อมต่อกับหน่วยความจำโปรแกรม



รูปที่ 2.2 การเชื่อมต่อกับหน่วยความจำข้อมูล

2.1.1.1 หน่วยความจำโปรแกรม ใน MCS-51 จะแบ่งหน่วยความจำโปรแกรมออกเป็น 2 ส่วน คือ หน่วยความจำโปรแกรมภายในและหน่วยความจำโปรแกรมภายนอก หน่วยความจำโปรแกรมภายในก็คือหน่วยความจำประเภท ROM หรือ EPROM ที่อยู่ภายในตัว MCS-51 เบอร์ 8051หรือ 8751 ตามลำดับ ซึ่งมีขนาด 4 กิโลไบต์ส่วนหน่วยความจำโปรแกรมภายนอกก็คือ หน่วยความจำที่ต่ออยู่ภายนอกตัว MCS-51 โดยอาจจะเป็นการต่อเพื่อขยายหน่วยความจำเพิ่มเนื่องจากหน่วยความจำภายในไม่พอหรืออาจต่อเป็นหน่วยความจำโปรแกรมภายนอกทั้งหมดเลยก็ได้ในกรณีของ 8031 เพราะภายในตัวมันไม่มีหน่วยความจำโปรแกรมเหมือนเบอร์ 8051 หรือ 8751 โดย MCS-51 สามารถเลือกให้รันโปรแกรมในหน่วยความจำโปรแกรมภายในหรือภายนอกก็ได้ โดยควบคุมที่ขา EA (ขา 31) แต่สำหรับเบอร์ 8031 จะต้องต่อขา EA ลงกราวนด์เสมอ การจัดแบ่งพื้นที่หน่วยความจำโปรแกรมภายในและภายนอก แสดงในรูปที่ 2.3



รูปที่ 2.3 การจัดพื้นที่หน่วยความจำข้อมูล

2.1.1.2 หน่วยความจำข้อมูล หน่วยความจำข้อมูลของ MCS-51 ก็ถูกแบ่งเป็น 2 ส่วนเหมือนกันคือ หน่วยความจำข้อมูลภายในและภายนอก โดยหน่วยความจำข้อมูลภายนอกจะเข้าถึงได้ก็ด้วยคำสั่ง MOVX เท่านั้น สำหรับหน่วยความจำข้อมูลภายในของ 8031/8051 และ 8751 จะมีทั้งหมด 256 ไบต์ โดยแบ่งเป็น 128 ไบต์ ส่วนบน ซึ่งให้เป็นที่อยู่ของรีจิสเตอร์ฟังก์ชันพิเศษ และอีก 128 ไบต์ส่วนล่าง ซึ่งจะถูกใช้งานทั่งไป

จะเห็นว่าพื้นที่หน่วยความจำข้อมูลภายในส่วน 128 ไบต์ล่าง (ตำแหน่ง 00H-7FH) จะถูกแบ่งเป็น 3 ส่วนคือ ส่วนของรีจิสเตอร์แบงค์ (00H-1FH) , ส่วนพิเศษที่สามารถเข้าถึงตำแหน่งบิตได้โดยตรง (20H-2FH) และส่วนที่ใช้งานทั่งไป (30H-7FH)

ส่วนของรีจิสเตอร์แบงค์มีทั้งหมด 4 แบงค์ แต่เราสามารถใช้ได้ครั้งละแบงค์เท่านั้น การเลือกใช้แบงค์ไหนอยู่ที่เรากำหนดค่าในรีจิสเตอร์ PSW

คำสั่งที่ใช้ในการติดต่อกับหน่วยความจำข้อมูล ซึ่งจะแบ่งลักษณะการกำหนดเลขที่อยู่ของหน่วยความนำข้อมูลได้ 4 โหมดด้วยกันคือ

โหมดการกำหนดเลขที่อยู่รีจิสเตอร์ จะเป็นการติดต่อกับข้อมูลที่อยู่ในรีจิสเตอร์โดยตรง ตัวอย่างเช่น

```
MOV   A , R0        ; ย้ายค่าใน R0 ไปไว้ที่ ACC
ADD   A , R3        ; บวกค่าใน ACC กับค่าใน R3 แล้วนำผลลัพธ์ไปไว้ใน ACC
```

โหมดการกำหนดเลขที่อยู่โดยตรง จะเป็นการติดต่อกับข้อมูลที่ตำแหน่งของแรมภายใน 128 ไบต์ ตัวอย่างเช่น

```
MOV   A, 40H        ; ย้ายค่าข้อมูลในหน่วยความจำแรมตำแหน่ง 40H ไปเก็บไว้ใน ACC
ADD   A,  41H       ; บวกข้อมูลใน แรมตำแหน่ง 41 H กับข้อมูลใน ACC
```

โหมดการกำหนดเลขที่อยู่ข้อมูลโดยทันที จะเป็นการติดต่อกับข้อมูลคงที่ ตัวอย่างเช่น

```
MOV   A, #40H       ; นำค่า 40H ไปเก็บใน Acc
ADD   A, #41H       ; บวกค่า41H เข้ากับค่าใน Acc ตำแหน่ง 41H
```

โหมดการกำหนดเลขที่อยู่รีจิสเตอร์โดยอ้อม จะเป็นการติดต่อกับข้อมูลที่ใช้ค่าในตัวรีจิสเตอร์ R0 หรือ R1 เป็นตัวชี้ตำแหน่ง ตัวอย่างเช่น

```
MOV   A, @RO        ; นำค่าข้อมูลในหน่วยความจำที่ถูกชี้ด้วย RO ไปเก็บใน ACC
MOV   A, @R1        ; บวกค่าจองข้อมูลในหน่วยความจำที่ถูกชี้ด้วย R1 กับค่าใน ACC
```

2.1.2 ตัวจับเวลา / ตัวนับ (Timer/Counter) [1]

ในเบอร์ 8031/8051 และ 8751 จะมีตัวจับเวลา / ตัวนับ ขนาด 16 บิด จำสวน 2 ตัว ไทเมอร์/เคาน์เตอร์ 0 และ ไทเมอร์/เคาน์เตอร์ 1 ส่วนเบอร์ 8032/8052 จะมีเพิ่มอีก 1 ตัว โดยแต่ละตัวสามารถที่

จะกำหนดให้ทำงานเป็นตัวจับเวลาหรือตัวนับได้โดยการเซตหรือเครียร์บิต C/T ที่ตัวรีจิสเตอร์ควบคุม TMOD ซึ่งอยู่ในกลุ่มรีจิสเตอร์ฟังก์ชันพิเศษ

ในการกำหนดให้ทำงานเป็นตัวจับเวลา ตัวรีจิสเตอร์ TH1 และ TL1 ซึ่งทำหน้าที่เป็นตัวเก็บค่าจำนวนพัลส์ที่เข้ามาจะเพิ่มค่าทุก ๆ แมชชีนไซเกิล โดยแต่ละแมชชีนไซเกิลจะประกอบด้วย 12 คาบออสซิลเลเตอร์ ดังนั้นอัตราการนับแต่ละครั้งจะใช้เวลาเท่ากับ 1/12 ของความถี่ออสซิลเลเตอร์ซึ่งส่วนใหญ่จะใช้ในงานอินเตอร์รัพต์ RTC (Real Time Clock)

และถ้าให้ทำงานเป็นตัวนับ รีจิสเตอร์ตัวนับจะเพิ่มค่าทุกครั้งที่มีการเปลี่ยนแปลงสถานะจาก "1" เป็น "0" ที่ ขา T0 หรือ T1 โดยอัตราความถี่สูงสุดที่สามารถนับได้ต้องไม่เกิน 1/24 ของความถี่ออสซิลเลเตอร์

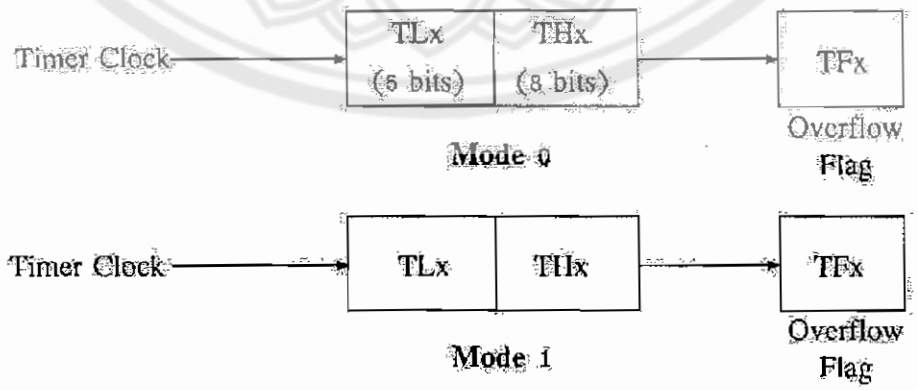
ตัวจับเวลา/ตัวนับ สามารถโปรแกรมให้มันทำงานได้ต่างกันถึง 4 โหมด โดยการตั้งค่าให้รีจิสเตอร์ TMOD ซึ่งจะกล่าวถึงการทำงานของแต่ละโหมดดังนี้

โหมด 0 รีจิสเตอร์ตัวนับจะถูกกำหนดให้มี 13 บิต ประกอบด้วยรีจิสเตอร์ TH1 8 บิต และ TL1 อีก 5 บิต อันดับต่ำซึ่งสามารถกำหนดให้เป็นตัวจับเวลาหรือตัวนับได้โดยเซตหรือเคลียร์ที่บิต C/T ในตัวรีจิสเตอร์ TMOD การทำงานของรีจิสเตอร์ตัวนับจะนับขึ้นครั้งละ 1 เมื่อมีสัญญาณเข้ามา 1 ลูก และเมื่อนับจนเป็น "1" หมดทุกบิต ก็จะกลับมาเป็น "0" หมดทุกบิตใหม่ ซึ่งจะเป็นการเกิดโอเวอร์โฟลว์ไปทดแฟลกอินเตอร์รัพต์ TF1 ให้เป็น "1" ลักษณะวงจรเป็นดังรูปที่ 2.4

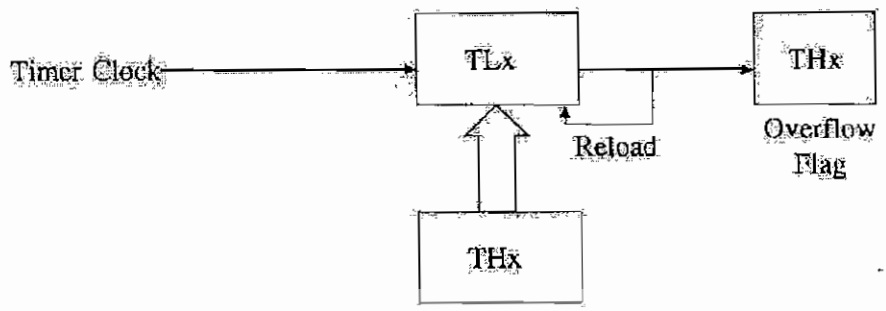
โหมด 1 การทำงานจะเหมือนกับโหมด 0 ทุกอย่าง ยกเว้นรีจิสเตอร์ตัวนับจะเป็นขนาด 16 บิตข

โหมด 2 จะใช้รีจิสเตอร์ TL1 เป็นตัวนับเพียงตัวเดียวและเมื่อ TL1 นับจนเป็น "1" หมดทุกบิต ก็จะมีการโหลดค่าจากรีจสเตอร์ TH เข้าไปไว้ใน TL1 โดยอัตโนมัติ และทำการทดแฟลกอินเตอร์รัพต์ TF1 ให้เป็น "1" ค่าใน TH1 นี้ เราสามารถตั้งค่าได้ด้วยซอฟต์แวร์ ลักษณะวงจรแสดงในรูปที่ 2.5

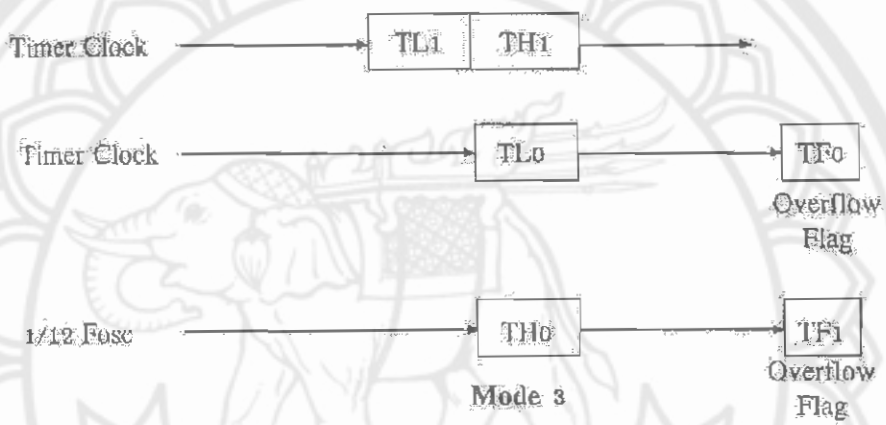
โหมด 3 เป็นการเพิ่มตัวจับเวลาขึ้นอีก 1 ตัว แต่จะเป็นขนาด 8 บิตทั้งคู่ ซึ่งลักษณะการทำงานอื่น ๆ จะเหมือนกับ โหมด 0 การทำงานแสดงในรูปที่ 2.6



รูปที่ 2.4 วงจรการทำงานของตัวจับเวลา/ตัวนับที่ 1 ในโหมด 0 และ โหมด 1



รูปที่ 2.5 วงจรการทำงานของตัวจับเวลา/ตัวนับที่โหมด 2



รูปที่ 2.6 วงจรการทำงานของตัวจับเวลา/ตัวนับที่ 0 ในโหมด 3

-รายละเอียดบิตควบคุมในรีจิสเตอร์ TMOD และ รีจิสเตอร์TCON

ตารางที่ 2.1 บิตควบคุมในรีจิสเตอร์ TMOD

TIMER/COUNTER 1				TIMER/COUNTER 0			
GATE	C/T	M1	M0	GATE	C/T	M1	M0

GATE : เซตเป็น "1" จะเป็นการอีนาเบิลตัวจับเวลา/ตัวนับให้ถูกควบคุมด้วยขา INTx ต้อง มีสถานะสูงและบิต TRX ใน TCON ต้องเซตเป็น"1" จึงจะเริ่มทำงาน แต่ถ้า GATE เป็น "0" ตัวจับเวลา/ตัวนับจะถูกควบคุมให้เริ่มทำงานด้วยบิต TRX เท่านั้น

C/T : เป็นบิตควบคุมในการเลือกทำงานเป็นตัวจับเวลาหรือตัวนับ ถ้าเป็น "0" จะทำงานเป็นตัวจับเวลา ถ้าเป็น"1" ทำงานเป็นตัวนับ

M1, M0 : เป็นตัวเลือกโหมดการทำงานดังนี้

ตารางที่ 2.2 การเซตโหมดการทำงานของไทเมอร์

M1	M0	โหมดการทำงาน
0	0	โหมด 0
0	1	โหมด1
1	0	โหมด2
1	1	โหมด 3

-แสดงบิตควบคุมที่อยู่ในรีจิสเตอร์ TCON

ตารางที่ 2.3 บิตควบคุมที่อยู่ในรีจิสเตอร์ TCON

TF1	TR1	TF0	TR0	**IE1**	IT1	IE0	IT0

TFx ; เป็นแฟลกอินเตอร์รัพต์จะถูกเซตด้วยฮาร์ดแวร์เมื่อเกิดโอเวอร์โฟลว์ของตัวจับเวลา / ตัวนับ และจะเคลียร์ตัวเองโดยอัตโนมัติเมื่อทำงานอินเตอร์รัพต์นั้นเสร็จเรียบร้อยแล้ว

TRx ; เป็นบิตควบคุมให้ตัวจับเวลา/ ตัวนับเริ่มทำงาน โดยการเซตให้เป็น"1" และให้หยุดทำงานด้วยการเคลียร์ให้เป็น "0" โดยซอฟต์แวร์

IEx ; เป็นแฟลกอินเตอร์รัพต์จากสัญญาณภายนอก เซตด้วยฮาร์ดแวร์เมื่อมีสัญญาณของการอินเตอร์รัพต์ปรากฏที่ขา INTx และจะเคลียร์ตัวเองโดยอัตโนมัติ เมื่อกระโดดไปทำงานบริการอินเตอร์รัพต์ที่ขอมาเรียบร้อยแล้ว

ITx ; เป็นบิตควบคุมรูปแบบสัญญาณอินเตอร์รัพต์ภายนอกจะเซต/เคลียร์ด้วยซอฟต์แวร์โดยถ้าเซตเป็น"1" จะถูกอินเตอร์รัพต์ด้วยสัญญาณขอบขาลง และถ้าเคลียร์ เป็น "0" จะถูกอินเตอร์รัพต์ด้วยสัญญาณระดับแรงดันต่ำ

2.1.3 พอร์ตอนุกรม [1]

MCS-51 จะมีพอร์ตอนุกรมเป็นแบบฟูลดูเพล็กซ์ สามารถที่จะส่งและรับข้อมูลได้พร้อมกัน เพราะมีบัฟเฟอร์ 2 ตัว ใช้ในการับตัวหนึ่ง และส่งตัวหนึ่ง โดยโครงสร้างของรีจิสเตอร์บัฟเฟอร์ ทั้ง 2 ตัวนั้นจะแยกกันแต่การติดต่อจะใช้ชื่อเดียวกับคือ SBUF พอร์ตอนุกรมของ MCS-51 สามารถที่จะโปรแกรมให้ทำงานได้แตกต่างกัน 4 โหมด

โหมด 0 ข้อมูลจะเข้าและออกทางขา RXD โดยการเลื่อนสัญญาณนาฬิกาออกที่ขา TXD ข้อมูลจะเป็น 8 บิต โดยจะส่งบิตนัยสำคัญต่ำ ก่อน อัตราบอต จะคงที่ที่ 1/12 ของความถี่ออสซิลเลเตอร์

โหมด 1 เป็นการรับ/ส่ง ข้อมูลขนาด 10 บิต โดยการส่งออกทางขา TXD และรับเข้าทางขา RXD รูปแบบบิตจะประกอบด้วย 1 บิต สตาร์ท เป็น"0" , 8 บิตข้อมูล และ 1 สต็อปบิตเป็น "1" อัตราเร็วแปรผันได้ตามการตั้งตัวจับเวลาตัวที่ 1 โดยมีสูตรดังนี้

$$\text{อัตราเร็ว} = \frac{2^{SMOD}}{32} \times \frac{\text{ความถี่ออสซิลเลเตอร์}}{12\times(256 - TH1)}$$

โหมด 2 เป็นการรับส่งข้อมูลขนาด 11 บิต เข้าทางขา RXD และส่งออกทางขา TXD ประกอบด้วย 1 บิต สตาร์ตมีค่า "0" , 9 บิตข้อมูลและ 1 บิตสต๊อป โดยการรับข้อมูลบิตใส่บิตที่ 9 ไว้ใน TB8 ของ SCON ก่อน อัตราเร็ว สามารถเลือกได้ 2 อัตรา คือ 1-32 หรือ 1-64 ของความถี่ออสซิเลเตอร์ขึ้นอยู่กับการเซตบิต SMOD ในรีจิสเตอร์ PCON

โหมด 3 จะเหมือนกับโหมด 2 ทุกอย่าง ยกเว้นอัตราบอตจะแปรผันตามการตั้งตัวจับเวลา 1 ซึ่งจะใช้สูตรเดียวกับโหมด1

-รายละเอียดบิตควบคุมที่อยู่ในรีจิสเตอร์ SCON

ตารางที่ 2.4 บิตควบคุมที่อยู่ในรีจิสเตอร์ SCON

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 , SM1 : เป็นตัวกำหนดโหมดการใช้งานของพอร์ตอนุกรมดังนี้

ตารางที่ 2.5 การเซตโหมดการทำงานของพอร์ตอนุกรม

SM0	SM1	โหมด
0	0	0
0	1	1
1	0	2
1	1	3

SM2 : ควบคุมอีนาเบิลการใช้โปรเซสเซอร์หลายตัวในการสื่อสารซึ่งกันและกันใน โหมด 2 และ 3

REN : ตัวอีนาเบิลอนุกรมการรับเมื่อเซตเป็น "1" และถ้าเป็น "0" เป็นการดิสเอเบิล

TB8 : เป็นตัวเก็บข้อมูลบิตที่ 9 ที่จะส่งในโหมด 2 และ 3

RB8 :เป็นตัวรับข้อมูลบิตที่ 9 ในโหมด 2 และ3 ส่วนในโหมด 1 จะสต๊อปบิต

TI : เป็นแฟลกอินเตอร์รัพต์การเซตด้วยฮาร์แวร์ที่ปลายช่วยของบิตที่ 8 ในโหมด 0 หรือที่จุดเริ่มต้นของบิตสต๊อปในโหมดอื่น ในการส่งแบบอนุกรมของทุก โหมดจะต้องเคลียร์บิตนี้ด้วยโปรแกรมหลังการส่ง

RI : เป็นแฟลกอินเตอร์รัพต์การรับ เซตด้วยฮาร์แวร์ที่ปลายช่วยของบิตที่ 8 ในโหมด 0 หรือจุดครึ่งของช่วงบิตสต๊อปในโหมดอื่น ในการรับแบบอนุกรมจะต้องเคลียร์บิดนี้ด้วยโปรแกรมหลังการรับทุกครั้ง

2.1.4 การอินเตอร์เฟสแบบอนุกรม ของ MCS-51 [2]

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 ได้บรรจุวงจรในส่วนการอินเตอร์เฟสแบบอนุกรมไว้ภายในตัวด้วยซึ่งในส่วนนี้นับได้ว่าเป็นอีกภาคหนึ่งที่มีความซับซ้อนมาก อีกทั้งฮาร์ดแวร์หรือบอร์ด MCS-51 ก็ได้ออกแบบวงจรในส่วนมอนิเตอร์เองก็มีรูทีนสำหรับเรียกใช้งานไว้เสร็จสรรพแล้วดังนั้นคงจะกล่าวได้แต่เพียงการทำงานพื้นฐานและการเขียนโปรแกรมใช้งานกับ MCS-51 บอร์ดนี้เฉพาะโหมดที่ใช้งานกันเท่านั้น

SM0 , SM1 : เป็นตัวกำหนดโหมดการใช้งานของพอร์ตอนุกรมดังนี้

ตารางที่ 2.6 ลักษณะการทำงานในโหมดต่าง ๆ ของพอร์ตอนุกรม

SM0	SM1	โหมด	ลักษณะการทำงาน	อัตราอัตราเร็ว
0	0	0	ชิฟต์รีจิสเตอร์	Fose/12
0	1	1	8 บิต UART	กำหนดค่าเองได้
1	0	2	9 บิต UART	fose/32 หรือ fose/64
1	1	3	9 บิต UART	กำหนดค่าเองได้

- บิต SM2 จะถูกอีนาเบิลเมื่อทำการติดต่อสื่อสารระหว่างโปรเซสเซอร์หลายตัว ในโหมด 2 และ 3 โดยถ้า SM2 ถูกเซตนั่นคือบิต RI ไม่ถูกเซตหรือข้อมูลบิดที่ 9 ที่เก็บใน RB8 มีค่า เป็น "0" และสำหรับในโหมด 1 ถ้าบิต SM 2 ถูกเซตนั่นคือบิต RI ไม่ถูกเซตหรือแสดงว่าบิตสิ้นสุดไม่ได้รับเข้ามา และสำหรับในโหมด 0 บิตนี้จะมีค่าเป็น "0"

- บิต REN ทำหน้าที่อีนาเบิลการรับข้อมูลผ่านพอร์ตอนุกรม สามารถเซตเพื่ออีนาเบิลและเคลียร์เพื่อดิสเอเบิลการรับข้อมูลได้

- บิต TB8 เป็นสถานะของบิตที่ 9 สำหรับการส่งข้อมูลในโหมด 2 และ 3 สามารถเซตและเคลียร์ได้จากซอฟต์แวร์

- บิต RB8 ในโหมด 2 และ 3 บิตนี้คือสถานะของบิตที่ 9 ที่ได้รับเข้ามา สำหรับในโหมด 1 ถ้าบิต SM2 เป็น "0" บิตนี้คือบิตสิ้นสุดที่รับเข้ามา และสำหรับในโหมด 0 บิตนี้ไม่ถูกใช้งาน

- บิต TI ทำหน้าที่เป็นอินเตอร์รัพต์แฟลกในการส่งข้อมูล จะถูกเซตโดยฮาร์ดแวร์เมื่อสิ้นสุดการส่งข้อมูลบิตที่ 8 ในการทำงานโหมด 0 หรือถูกเซตที่จุดเริ่มต้นของบิตสิ้นสุดในการทำงานโหมดอื่น ๆ และบิตนี้ต้องถูกเคลียร์จากซอฟต์แวร์

- บิต RI ทำหน้าที่เป็นอินเตอร์รัพต์แฟลกในการับข้อมูล จะถูกเซตโดยฮาร์แวร์เมื่อสิ้นสุดการรับข้อมูลบิตที่ 8 ในการทำงานโหมด 0 หรือถูกเซตที่ช่วงระหว่าง กลางของบิตสิ้นสุดในโหมดอื่น ๆ (ยกเว้นบางกรณีสัมพันธ์กับบิต SM2) และบิตนี้ต้องถูกเคลียร์จากซอฟต์แวร์

การอินเตอร์เฟสผ่านพอร์ตอนุกรมสามารถเลือกทำงานได้หลายโหมด ซึ่งในบางโหมดนั้นอาจไม่ต้องให้ความสนในเลยก็ได้ เพราะมันจะถูกเลือกก็ต่อเมื่อทำงานเป็นระบบไมโครคอนโทรลเลอร์หลายตัวเชื่อมต่อกันเป็นเน็ตเวอร์ก

การอินเตอร์เฟสผ่านพอร์ตอนุกรมมีรีจิสเตอร์พิเศษ (SFR) ควบคุมการทำงานคือรีจิสเตอร์ SCON อยู่ที่ตำแหน่งแอดเดรส 098H ทำหน้าที่หลักคือควบคุมการอินเตอร์เฟสแบบอนุกรมในตัวไมโครคอนโทรลเลอร์และเป็นตัวกำหนดโหมดการทำงาน ซึ่งแสดงหน้าที่ของแต่ละตำแหน่งบิตของรีจิสเตอร์ SCON

2.1.5 บัฟเฟอร์สำหรับการรับและส่งข้อมูล [2]

ภายในตัวไมโครคอนโทรลเลอร์บิตข้อมูลที่ต้องการทำการส่งและรับผ่านพอร์ตอนุกรมจะถูกส่งและรับลักษณะแบบชิฟต์ เช่นเดียวกับการทำงานของชิฟต์รีจิสเตอร์ในกรณีที่ทำการส่งข้อมูล บิตข้อมูลทั้งหมดจะถูกโหลดลงสู่บัฟเฟอร์ ซึ่งทำงานเป็นชิฟต์รีจิสเตอร์แบบขนาน หลังจากนั้นจึงทำการชิฟต์ข้อมูลออกโดยเลื่อนไปทีละบิตด้วยบิตเรตหรืออัตราเร็ว ที่กำหนดไว้แล้วในตรงข้ามกัน กล่าวคือบิตข้อมูลที่รับได้จะถูกนำมารวบรวมไหม่ด้วยการชิฟต์บิตข้อมูลเข้าทีละบิตไปยังชิฟต์รีจิสเตอร์เมื่อได้รับครบแล้ว จึงทำการอ่านข้อมูลนั้นไปใช้งานแบบขนานเข้าสู่ระบบ

ชิฟต์รีจิสเตอร์ที่กล่าวถึงนี้นำหน้าที่เป็นเช่นเดียวกับบัฟเฟอร์ เพื่อพักข้อมูลในการส่งและรับข้อมูลซึ่งไมโครคอนโทรลเลอร์ก็ได้จัดเตรียมไว้แล้ว นั่นคือรีจิสเตอร์พิเศษ (SFR) ชื่อ SBUF อยู่ที่ตำแหน่งแอดเดรส 099H มันจะทำหน้าที่เป็นได้ทั้งบัฟเฟอร์รับข้อมูลในกรณีระบบทำการรับข้อมูลจากภายนอกและเป็นบัฟเฟอร์เพื่อพักข้มูลก่อนส่งออกสู่ระบบในกรณีระบบทำการส่งข้อมูลซึ่งผู้เขียนโปรแกรมต้องกำหนดการใช้คำสั่งอ่านหรือเขียนข้อมูลไปยังรีจิสเตอร์ SBUF ตามรูปแบบของการติดต่อที่ต้องการในขณะนั้นและตรงกับกรณีด้วย

2.1.6 อัตราเร็วในการสื่อสารผ่านพอร์ตอนุกรม [2,4]

ถ้าเคยใช้ไมโครคอนโทรลเลอร์เบอร์ 8051/8031 ที่ใช้คริสตอลความถี่ 12 เมกะเฮิรตซ์นั้นจะพบว่าไม่สามารถกำหนดอัตราเร็ว ในการสื่อสารผ่านพอหรืออนุกรมที่ ความเร็ว 4800 bps (Bit per seconds)ได้อย่างถูกต้องสมบูรณ์ทั้งนี้เพราะค่าอัตราเร็วต้องขึ้นอยู่กับความถี่ของสัญญาณนาฬิกาของระบบ ด้วย ซึ่งมีค่าเท่ากับ 12 MHz/12 = 1 MHz ปัญหานี้ถูกแก้ไขให้หมดไป โดยใช้ภาคกำเนิดอัตราเร็วพิเศษ ที่มีอยู่ใน 80C535 หรือกล่าวได้ว่าใน 80C535 นั้นถูกออกแบบมาให้มีไทเมอร์ตัวหนึ่งที่ทำงานอย่างอิสระและทำหน้าที่เฉพาะในการกำเนิดอัตราเร็วเพื่อการสื่อสารผ่านพอร์ตอนุกรมเท่านั้น

การควบคุมให้ภาคกำเนิดอัตราเร็วเริ่มทำงาน ทำได้โดยการเซตบิต 7 ในรีจิสเตอร์หน้าที่พิเศษ ADCON ซึ่งทำได้ด้วยคำสั่งเซตบิตง่าย ๆ เท่านั้น เพราะรีจิสเตอร์ ADCON สามารถเข้าได้ในระดับบิต สำหรับการกำหนดอัตราอัตราเร็วทำได้โดยควบคุมบิต 7 ในรีจิสเตอร์ PCON เพื่อเลือกอัตราอัตราเร็วระหว่าง 4800 bps (PCON 7= 0) และ 9600 bps (PCON 7 = 1) ดังแสดงตัวอย่างโปรแกรมการกำหนดอัตราอัตราเร็วในรูปที่ 2.4 และอย่างลืมว่ารีจิสเตอร์ PCON นั้นไม่สามารถเข้าถึงได้ในระดับบิต ดังนั้นเพื่อป้องกันไม่ให้บิตอื่น ๆ ในรีจิสเตอร์เกิดการเปลี่ยนแปลง เมื่อต้องการแก้ไขค่าบิตในรีจิสเตอร์ PCON จึงควรเลือกใช้คำสั่ง ANL และ ORL มาใช้ให้เหมาะสม

2.1.7 สัญญาณนาฬิกาของระบบ [2]

ในบางครั้งนั้น การออกแบบวงจรหรือการออกแบบด้านฮาร์ดแวร์เพื่อให้ทำงานร่วมกับไมโครคอนโทรลเลอร์ 80C535 นั้นจำเป็นต้องมีการใช้สัญญาณนาฬิกาของระบบภายใน 80C535 ด้วย ทั้งนี้ก็เพื่อให้จังหวะการทำงานของระบบเป็นไปอย่างเข้าจังหวะกัน ไม่เกิดการผิดพลาดไมโครคอนโทรลเลอร์ 80C535 จึงถูกออกแบบให้สามารถป้อนสัญญาณนาฬิกาของระบบ (มีค่าเท่ากับความถี่ของคริสตอบที่ต่อร่วมอยู่กับไมโครคอนโทรลเลอร์หารด้วยค่า 12) ออกที่ขา 30 หรือขา พอร์ต P1.6 ได้ด้วยโดยต้องทำการเซตบิต 6 ในรีจิสเตอร์ ADCON และเขียนค่า 1 ไปยังขาพอร์ต P1.6 เพื่ออีนาเบิลขาพอร์ตให้ทำงานเป็นขาเอาต์พุต ซึ่งถูกควบคุมผ่านลอจิกเกต AND ภายใน ดังนั้นรูปแบบการเขียนโปรแกรมจะเป็นดังนี้

```
SETB ADCON.6
SETB P 1.6
```

หลังจากประมวลคำสั่งทั้งสอบนี้แล้ว สัญญาณนาฬิกาของระบบภายใน 80C535 จะถูกส่งออกจากขาพอร์ต P1.6 ไว้ให้ต่อใช้งานร่วมกับวงจรภายนอกได้ทันที

2.1.8 การคำนวณอัตราเร็ว [4]

การคำนวณอัตราเร็วเป็นจุดสำคัญ โหมดการอินเตอร์เฟสที่จะกล่าวเน้นก็คือโหมด 1 ซึ่งเป็นโหมดที่พื้นฐานที่สุดในโหมดนี้อัตราอัตราเร็วถูกกำหนดได้จากไทเมอร์ 1 หรือ ไทเมอร์ 2 ตัวใดตัวหนึ่ง

แต่อย่าลืมว่าไทเมอร์ 2 นั้น จะไม่มีในไมโครคอนโทรลเลอร์เบอร์ 8031 และ 8051 ดังนั้นเพื่อเขียนโปรแกรมให้ใช้งานได้กับไมโครคอนโทรลเลอร์ทุกเบอร์จึงควรมุ่งเน้นเลือกใช้ไทเมอร์ 1 ในการกำหนดอัตราเร็วจะดีกว่า

สัญญาณจากไทเมอร์ 1 ที่เลือกใช้ก็คือพัลส์แสดงการโอเวอร์โฟลว์จากการนับ ซึ่งถูกป้อนไปยังภาคควบคุมการส่งและรับข้อมูล ดังนั้นเพื่อให้ไทเมอร์สามารถกำเนิดโอเวอร์โฟลว์พัลส์ได้อย่างต่อเนื่องจึงต้องตั้งการทำงานของไทเมอร์เป็นโหมด 2 คือทำงานแบบ 8 บิต โหลดค่าใหม่อัตโนมัติ นั่นคือไทเมอร์จะเกิดการโอเวอร์โฟลว์ทุก n ไมโครวินาที ในขณะที่ฐานเวลาภายในเป็น 1 เมกะเฮิรตซ์ เพราะว่าบน MCS-51 บอร์ดนั้นกำหนดให้ใช้คริสตอลออสซิลเลเตอร์ความถี่ 12 เมกะเฮิรตซ์ แต่ทั้งนั้นผู้เขียนโปรแกรมสามารถกำหนดให้โอเวอร์โฟลว์พัลส์จากไทเมอร์ถูกหารด้วย 2 หรือไม่ได้ด้วยการกำหนดสถานะของบิต SMOD และเมื่อป้อนเข้าสู่ระบบควบคุมการรับและส่งข้อมูลสัญญาณยังถูกหารด้วย 16 อีก ครั้งหนึ่ง ดังนั้นถึงที่สุดแล้วสามารถสรุปเป็นสูตรคำนวณได้ดังนี้

เมื่อ SMOD = 1 ใช้สูตร

$$\text{อัตราเร็ว} = \frac{\text{อัตราการโอเวอร์โฟลว์ของไทเมอร์ 1}}{16}$$

เมื่อ SMOD = 0 ใช้สูตร

$$\text{อัตราเร็ว} = \frac{\text{อัตราการโอเวอร์โฟลว์ของไทเมอร์ 1}}{32}$$

หลังจากกำหนดค่าต่าง ๆ เรียบร้อย โปรแกรมก็จะเริ่มสั่งให้ไทเมอร์ 1 ทำงาน โดยเซตบิต TCON.6 ในบรรทัดที่ 26 ขณะนี้ก็พร้อมแล้วที่จะทำการรับและส่งข้อมูลด้วยอัตราอัตราเร็วนี้

ทางเลือกอีกทางหนึ่งสำหรับผู้ที่ใช้งานไมโครคอนโทรลเลอร์เบอร์ 8052 และ 8032 คือใช้ ไทเมอร์ 2 เป็นตัวทำหน้าที่ กำหนดอัตราเร็ว โดยทำการเซตบิต TCLK และ RCLK ในรีจิสเตอร์ T1CON แต่เนื่องจากรีจิสเตอร์ T2CON นั้นจะมีค่าเป็น 00H เสมอหลังจากที่รีเซตเครื่องทุกครั้ง ดังนั้นเมื่อเริ่มต้นทำงานใหม่ของไมโครคอนโทรลเลอร์เบอร์ 8052 หรือ8032 จึงจำเป็นต้องใช้ไทเมอร์ 1 เป็นตัวกำหนดอัตราเร็วก่อนแล้วจึงเปลี่ยนไปใช้ไทเมอร์ 2 ได้ ซึ่งเห็นได้ว่าไม่ค่อยสะดวกเท่าไดนัก ส่วนมากจึงมักยึดถือให้ใช้ไทเมอร์ 1 เป็นตัวทำหน้าที่หลัก

อัตราอัตราเร็วมาตรฐานที่ใช้งานมีตั้งแต่ 1200 , 2400 , 4800 และอื่น ๆ การกำหนดอัตราเร็วให้ได้ค่าตามมาตรฐานนี้สามารถทำได้ใกล้เคียงมาก เมื่อใช้คริสตอลเป็นความถี่ 11.0592 เมกะเฮิรตซ์ ซึ่งดีกว่าการใช้คริสตอล 12 เมกะเฮิรตซ์ แต่ว่าค่าเวลาในส่งรอบการทำงานของโปรเซสเซอร์จะถูกเปลี่ยนเป็นค่าที่ตดเลขทศนิยม ซึ่งถ้าหากใช้คริสตอลเป็น 12 เมกะเฮิรตซ์ ค่าเวลาของวงรอบการทำงานนี้จะ

เป็น 1 ไมโครวินาที พอดี ทำให้ง่ายต่อการคำนวณค่าว่าดีอย่างก็เสียอย่าง ด้วยเหตุนี้ในไมโครคอนโทรลเลอร์รุ่งใหม่ ๆ จึงออกแบบให้มีภาคกำเนิดอัตราเร็วบนชิปภายในซึ่งทำให้ได้ค่าตรงตามมาตรฐานที่ใช้งานอย่างพอดี ในขณะเดียวกันก็ยังใช้คริสตอล 12 เมกะเฮิรตซ์ เป็นค่ากำหนดช่วงเวลาวงรอบการทำงานของโปรเซสเซอร์หรือเรียกได้ว่ามีแต่ข้อดีไม่มีข้อเสียเลย

ในไมโครคอนโทรลเลอร์ MCS-51 จะมีวงจรเคาน์เตอร์และไทเมอร์ (counters and timers) อยู่ด้วยเช่นในเบอร์ 8051 จะมีเคาน์เตอร์อยู่ 2 ตัว สำหรับใช้ในการนับค่าต่าง ๆ แต่เมื่อนำเคาน์เตอร์ที่มีอยู่มาป้อนสัญญาณนาฬิกาจากภายในระบบแทน เคาน์เตอร์ก็อาจถูกใช้เป็นวงจรไทเมอร์ได้ วงจรไทเมอร์แต่ละตัวสามารถทำงานได้ในหลายโหมดแล้วแต่ใช้งานจะกำหนด การโปรแกรมให้ทำงานในโหมดใดทำได้โดยการเซตและรีเซตบิตในรีจิสเตอร์พิเศษที่ควบคุมการทำงานของวงจรเคาน์เตอร์ และไทเมอร์ที่เกี่ยวข้อง

รีจิสเตอร์พิเศษสำหรับควบคุมโหมดการทำงานของไทเมอร์ คือ TMOD เป็นรีจิสเตอร์พิเศษ (SFR) ที่แอดเดรส 089H และไม่สามารถเข้าถึงได้ระดับบิต และรีจิสเตอร์พิเศษสำหรับควบคุมการทำงานของไทเมอร์ คือ TCON เป็นรีจิสเตอร์พิเศษที่แอดเดรส 088H และสามารถเข้าถึงได้ระดับบิต

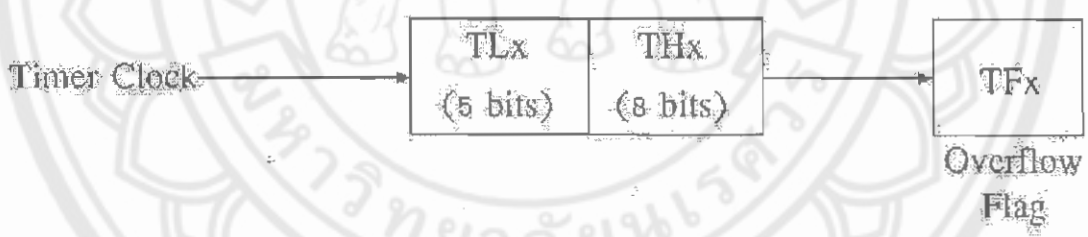
วงจรเคาน์เตอร์และไทเมอร์ของไมโครคอนโทรลเลอร์ MCS-51 เป็นแบบ 16 บิต โดยที่ในวงจรเคาน์เตอร์/ไทเมอร์ 1 จะแบ่งเป็นไบต์ต่ำและไบต์สูง ซึ่งในไบต์ต่ำหรือมีชื่อเรียกว่า TL1 จะอยู่ที่แอดเดรส 08BH และในไบต์สูง หรือ TH1 จะอยู่ที่แอดเดรส 08DH การกำหนดโหมดการทำงานของเคาน์เตอร์/ไทเมอร์ 1 นี้ทำได้โดยการกำหนดที่บิต TMOD.4 และ TMOD.5 ถ้าทั้ง 2 บิตนี้เป็น "0" เคาน์เตอร์จะถูกเซตให้ทำงานแบบ 13 บิต หรือหมด 0 สำหรับการทำงานในโหมด 0 นี้แสดงได้ดังรูปที่ 19 จะเห็นได้ว่ามีการติดต่อกับสัญญาณภายนอกด้วยบางส่วนโดยการควบคุมที่บิด C/T ในรีจิสเตอร์ TMOD ถ้าบิต C/T เป็น "0" นั่นคือเลือกใช้สัญญาณภายนอกมาใช้งานแทน

ระดับลอจิกของตำแหน่งบิต TR1 , GATE และระดับบอจิกที่ขา INT1 เป็นผลต่อการควบคุมการทำงานของวงจรเคาน์เตอร์/ไทเมอร์ ตัวอย่างเช่น ถ้าให้บิด GATE เป็น "0" การควบคมให้เคาน์เตอร์ทำงานและหยุดทำงานทำได้โดยการควบคุมที่บิด TR1 ได้และสำหรับตำแหน่งบิตและขาสัญญาณในโหมดนี้และโหมดอื่น ๆ ก็มีผลต่อการควบคุมเช่นเดียวกัน

M0,M1 เป็นบิตกำหนดโหมดการทำงานของไทเมอร์/เคาน์เตอร์

ตารางที่ 2.7 ลักษณะการทำงานในโหมดต่าง ๆ ของไทเมอร์

M1	M0	โหมดการทำงาน
0	0	กำหนดให้เป็นไทเมอร์/เคาน์เตอร์ขนาด 13 บิต โดยให้ TL ทำงานเพียง 5 บิต
0	1	กำหนดให้เป็นไทเมอร์/เคาน์เตอร์ขนาด 16 บิต
1	0	ทำงานแบบ 8 บิต โหลดค่าใหม่อัตโนมัติ โดยให้ TH เก็บค่าสำหรับป้อนค่าใหม่อัตโนมัติให้กับ YL เมื่อ TL เกิดโอเวอร์โฟลว์
1	1	สำหรับไทเมอร์ 0 กำหนดให้ TL0 ทำงานเป็นไทเมอร์/ เคาน์เตอร์ขนาด 8 บิต ถูกควบคุมตามแบบมาตรฐานเช่นเดียวกับในโหมดอื่น ๆ สำหรับ TH0 ทำงานเป็นไทเมอร์ขนาด 8 บิต ถูกควบคุมโดยบิตควบคุม TR1 เพียงสัญญาณเดียว
1	1	สำหรับไทเมอร์/เคาน์เตอร์ 1 ในโหมดนี้กำหนดให้หยุดทำงาน

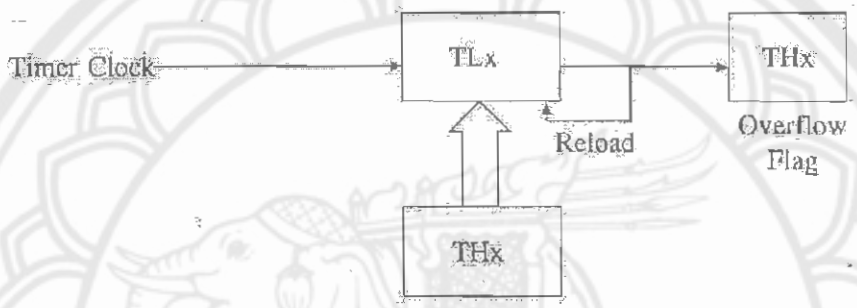


รูปที่ 2.7 การทำงานของไทเมอร์/เคาน์เตอร์ในโหมด 0

อีกแฟลกหนึ่งที่เกี่ยวข้องในส่วนนี้คือ TF1 ซึ่งมันจะถูกเซตทุกครั้งเมื่อวงจรเคาน์เตอร์ทำการนับจนเปลี่ยนค่าจากค่าสูงสุดกลับเป็นศูนย์อีกครั้ง หรือพูดอีกอย่าง ก็คือ เกิดการโอเวอร์โฟลว์ แฟลกนี้สามารถตรวจสอบได้จากผู้เขียนโปรแกรมและนำไปเป็นอินเตอร์รัพต์เพื่อใช้งานในโปรแกรมที่เขียนขึ้นได้ แต่นั่นหมายถึงผู้ใช้งานต้องทำการเซตในบิต IE.3 ของรีจิสเตอร์ IE ด้วย เพื่อให้อีนาเบิลแฟลกนี้ด้วย

ความถี่สัญญาณนาฬิกาของระบบที่ให้กับวงจรเคาน์เตอร์/ไทเมอร์ ในกรณีที่กำหนดค่า C/T = "0" วงจรเคาน์เตอร์จะนับสัญญาณนาฬิกโดยคิดจากความถี่ของคริสตอลที่ต่อกับไมโครคอนโทรลเลอร์หารด้วยค่า 12 ในกรณีของ MCS-51 บอร์ด ที่ใช้ในที่นี้ก็คือได้ผลลัพธ์เป็นความถี่ 1 เมกะเฮิรตซ์

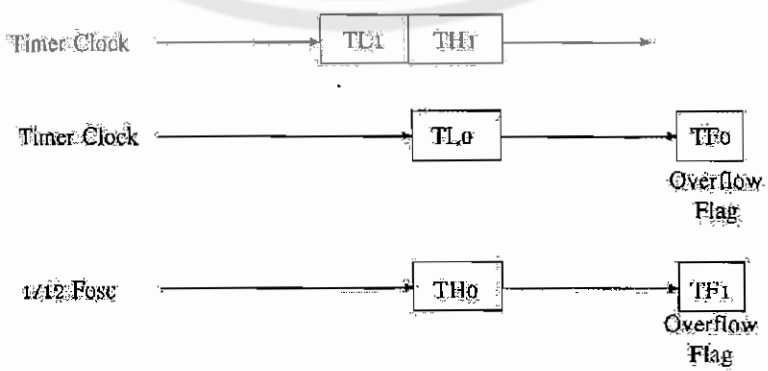
การทำงานในโหมดอื่น ๆ มีดังนี้ ในโหมด 1(TM- OD.4 = 1และ TMOD .5 = 0) และวงจรเคาน์เตอร์จะทำแทนในโหมด 2 วงจรเคาน์เตอร์ถูกกำหนดให้ทำงานแบบ 8 บิต โดยค่าที่ทำการนับจะอยู่ที่รีจิสเตอร์ TL1 และเมื่อTL1 นับค่าจนเกิดการโอเวอร์โฟลว์มันจะถูกป้อนค่าใหม่ (reload) โดยค่าที่เก็บใน TH1 เป็นการทำงานแบบโหลดค่าอัตโนมัติ (8bit auto reload) การทำงานในโหมดนี้แสดงดังในรูปที่ 2.8 สำหรับในโหมด 3 วงจรเคาน์เตอร์/ไทเมอร์ 1 นี้จะไม่สามารถใช้งานได้



รูปที่ 2.8 การทำงานของไทเมอร์/เคาน์เตอร์ในโหมด 2

ในโปรแกรมมอนิเตอร์ EMON51 วงจรเคาน์เตอร์/ไทเมอร์ 1 ถูกใช้ในภารสร้างอัตราบ็รว (baud rate generator) ดังนั้นมันจึงไม่สามารถนำไปใช้งานในการเขียนโปรแกรมของผู้ใช้งานได้ ยกเว้นเสียแต่ถ้าไม่ต้องการใช้งานติดต่อกับภายนอกผ่านการอินเตอร์เฟสแบบอนุกรม

การทำงานของวงจรเคาน์เตอร์/ไทเมอร์ 0 มีการทำงานเช่นเดียวกับวงจรเคาน์เตอร์/ไทเมอร์ 1 แตกต่างกันที่ตำแหน่งของบิตและรีจิสเตอร์ในการควบคุมและใช้งานอีกประการหนึ่งที่แตกต่างกันก็คือ วงจรเคาน์เตอร์/ไทเมอร์ 0 สามารถทำงานในโหมด 3 ได้ (TMOD.0 = 1 , TMOD.1 = 1) ในโหมดนี้มีการทำงานแบบวงจรเคาน์เตอร์/ไทเมอร์ขนาด 8 ลิด 2 ตัว แยกอิสระต่อกันดังแสดงในรูปที่ 2.9



รูปที่ 2.9 การทำงานของไทเมอร์/เคาน์เตอร์ 0 ในโหมด 3

ข้อสังเกตหนึ่งวงจรเคาน์เตอร์/ไทเมอร์ 8 บิต ตัวที่ 2 (ด้านล่างของรูป) ใช้บิตควบคุมวงจรเคาน์เตอร์/ไทเมอร์ 1 ด้วย ด้วยเหตุนี้จึงไม่สามารถใช้บิตนี้ในเคาน์เตอร์ไทเมอร์ 1 ได้ เมื่อกำหนดให้วงจรเคาน์เตอร์/ไทเมอร์ 0 ทำงานในโหมด 3

2.2 วงจรในภาคแปลงสัญญาณและยกระดับสัญญาณ

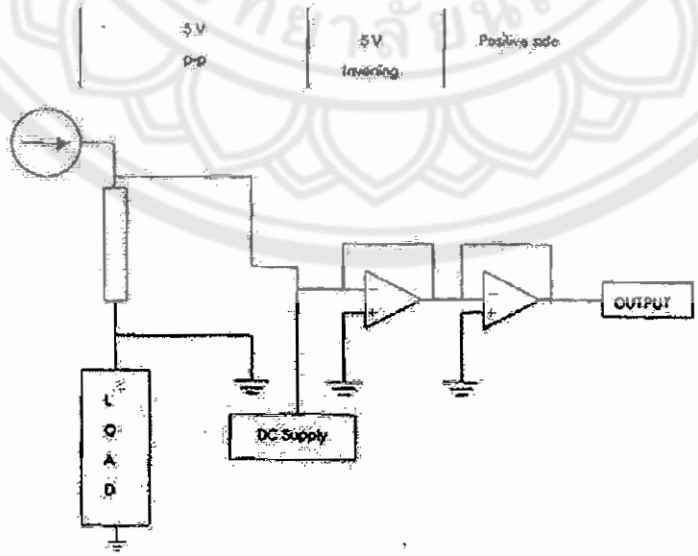
2.2.1 การแปลงสัญญาณกระแสเป็นสัญญาณแรงดัน (Current-to-Voltage Conversion)[3]

เมื่อมีการส่งสัญญาณแบบกระแสไปยังปลายทางอีกด้านหนึ่งจะต้องมีการแปลงรูปแบบของสัญญาณกระแสให้กลับเป็นสัญญาณแรงดันเพื่อนำไปใช้งาน ตัวอย่างวงจรที่ใช้ในการแปลงสัญญาณกระแสเป็นสัญญาณแรงดัน (Current-to-Voltage Conversion) แสดงในรูปด้านล่าง โดยที่หลักการทำงานของวงจรจะทำการแปลงสัญญาณกระแสที่ส่งมาให้เป็นแรงดันแตกต่างจากผลของ R_{span} ออปแอมป์ U_1 และตัวต้านทาน R_1,R_2 จากวงจรขยายผลต่างแรงดัน โดยที่ $R_i >> R_{span}$

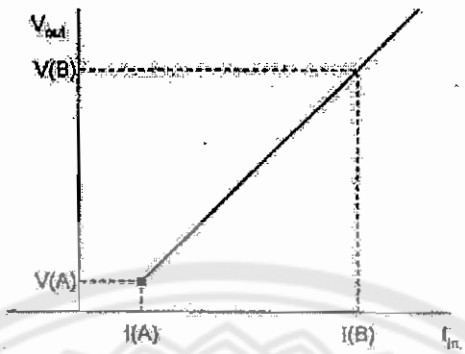
ซึ่งออฟเซต หรือการปรับแก้ซีโร่ จะถูกจัดการโดยความต้านทาน R_{pot} และออปแอมป์ U_2 ผลที่ได้ทางด้านเอาท์พุตจาก U_1 จะขึ้นกับสัญญาณกระแส I_{in} โหลดลอย R_{span} อัตราขยาย R_f/R_i และแรงดันซีโร่ออฟเซต V_2

$$V_2 = \frac{R_2}{R_1} \times IR_{SPAN} + V_Z$$

ค่าของอุปกรณ์ที่ใช้กำหนดจุดการทำงานของซีโร่และสแปนสามารถพิจารณาได้จากกราฟแสดงการแปลงสัญญาณกระแสเป็นสัญญาณแรงดันดังรูปที่ 2.10



รูปที่ 2.10 วงจรการแปลงสัญญาณกระแสเป็นสัญญาณแรงดัน



รูปที่ 2.11 กราฟการแปลงสัญญาณกระแสเป็นสัญญาณแรงดัน

ที่จุดการทำงาน A , $V(A) = (R_f/R_i)\, I(A)R_{span} + V_z$

ที่จุดการทำงาน B , $V(B) = (R_f/R_i)\, I(B)R_{span} + V_z$

ถ้านำสมการที่จุดการทำงาน B ลบด้วยสมการที่จุดการทำงาน A เพื่อกำจัดผลของ V_z จะได้ว่า

$$V(A) - V(B) = (R_f/R_i)\, R_{span}\, [I(B) + I(A)]$$

จัดรูปสมการใหม่ให้อยู่ในรูปความสัมพันธ์ของ R_{span} จะได้ว่า

$$R_{span} = [(V(A) - V(B))/(R_f/R_i)\, (I(B) + I(A))]$$

และสามารถหาความสัมพันธ์ของ V_z ได้จากสมการที่จุดการทำงาน A

$$V_z = V(A) - (R_f/R_i)\, I(A)R_{span}$$

2.2.2 วงจรบวกแรงดัน [3]

จากหลักการของวงจรพื้นฐานของออปแอมป์ เราสามารถนำออปแอมป์ไปประยุกต์ใช้งานได้หลายรูปแบบ เช่น เป็นวงจรบวกแรงดัน วงจรขยาย ผลรวมของแรงดัน วงจรลบแรงดัน และวงจรขยายผลต่างของแรงดัน เป็นต้น

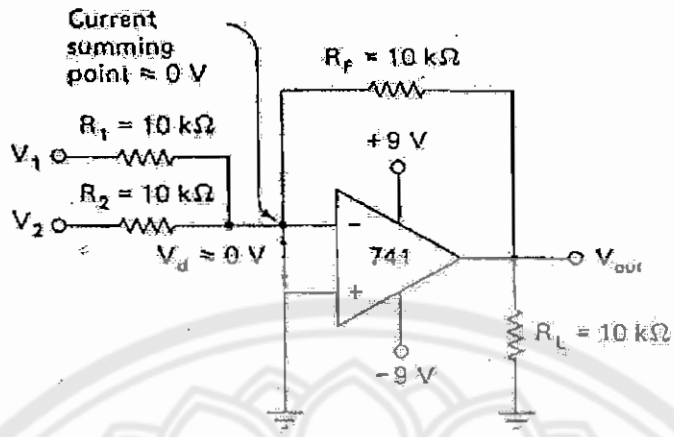
ตามที่แสดงให้เห็นในรูป2.12 จะแสดงวงจรบวกแรงดัน และตารางตัวอย่าง ซึ่งจากรูปสามารถอธิบายโดยใช้หลักที่ว่า ผลรวมเชิงพีชคณิตของกระแส ณ จุดกราวด์เสมือนจะทำให้เกิดแรงดันเอาท์พุทซึ่งมีขนาดเท่ากับผลคูณของ RF กับผลรวมของกระแสเหล่านี้ ดังนั้นแรงดันเอาท์พุท จึงเปรียบเสมือนผลรวมเชิงพีชคณิตของแรงดันอินพุททั้งหมดด้วย

นั่นคือ $V\ รวม = I_F R_F = (I_1 + I_2 + I_3 + \ldots . I_n)\, R_F$

และเนื่องจาก $R_1 = R_2 = \ldots\ldots = R_n = R_F$ จะได้ว่า

$$V\ รวม = I_1 R_1 + I_2 R_2 + \ldots\ldots . I_n R_n$$

$$\therefore \quad V\ รวม = V_1 + V_2 + \ldots\ldots + V_n$$



รูปที่ 2.12 วงจรบวกแรงดัน

ตารางที่ 2.8 ผลลัพธ์ของวงจรบวกแรงดัน

Input voltage		Output voltage
V_1	V_2	Algerbraic sum
+1	+1	-2
+1	-1	0
+2	+1	-3
-1	+1	0
-1	+2	-1
-2	+1	+1

$$V_{out} = -[(R_F/R_1)\,V_1 + (R_F/R_2)\,V_2 + \ldots\ldots + (R_F/R_n)\,V_n]$$

เมื่อ $R_1 = R_2 = \ldots\ldots = R_n = R_F$

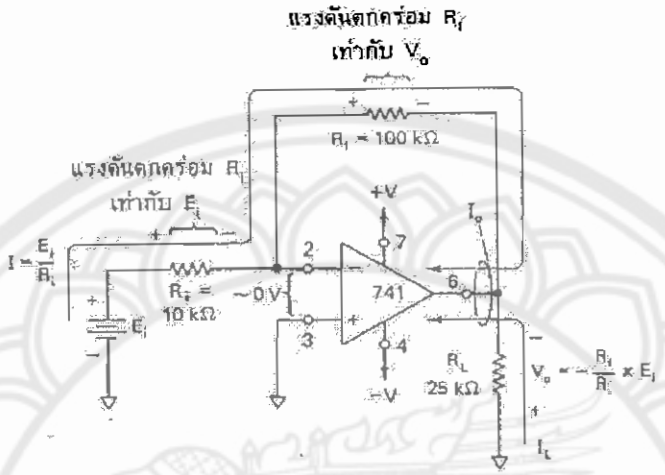
$$V_{out} = -(V_1 + V_2 + \ldots\ldots + V_n)$$

2.2.3 วงจรขยายแบบกลับเฟส [3]

วงจรในรูปที่ 2.13 เป็นวงจรที่ขยายได้ทั้งสัญญาณ AC และ DC อัตราการขยายแบบลูปปิด A_{CL} จาก E_i ไปสู่ V_o ขึ้นอยู่กับ R_f และ R_i

1. ในกรณี V_o ไม่อิ่มตัว (Saturation) ความต่างศักย์ระหว่างขาอินพุตบวกและลบ (E_d) จะเท่ากับศูนย์

2. ปริมาณกระแสที่ไหลเข้าขาอินพุตทั้งสองจะมีค่าน้อยมาก จนคิดว่าไม่มีกระแสไหลเข้า



รูปที่ 2.13 วงจรขยายแบบกลับเฟส

จากวงจรในรูปที่ 2.13 เมื่อป้อนแรงดัน $+E_i$ ให้กับขาอินพุตลบ (ขาอินเวอร์ติ้ง) ผ่านตัวต้านทานอินพุต R_i และให้ R_f ซึ่งเป็นตัวต้านทานป้อนกลับต่ออยู่ระหว่างขาเอาต์พุตและอินพุตลบ ส่วนขาอินพุตบวกต่ออยู่กับกราวด์ สิ่งที่ต้องการหาก็คือ A_{CL} ว่าจะมีค่าเท่าไร ซึ่งหาได้จากสูตร V_O/E_i โดย V_O หาได้ตามขั้นตอนต่อไปนี้

จากรูปที่ 2.13 แรงดันบวกถูกป้อนเข้าที่ขาลบของออปแอมป์ R_i จะทำหน้าที่ในการแปลงแรงดันให้เป็นกระแส I จากนั้น R_f จะทำหน้าที่แปลงกระแส I ให้กลับไปอยู่ในรูปของแรงดันอีกครั้ง โดยเป็นแรงดันที่เป็นสัดส่วนกับ E_i

จากข้อที่ว่าความต่างศักย์ระหว่างขาบวกและขาลบเป็นศูนย์ ซึ่งหมายความว่าแรงดันของทั้ง 2 ขาต้องเท่ากัน ดังนั้นเมื่อขาอินพุตบวกในวงจรนี้จึงเป็นกราวด์ ขาอินพุตลบจึงเสมือนต่ออยู่กับกราวด์ด้วย กระแสที่ไหลผ่าน R_i จึงเกิดความต่างศักย์ระหว่าง E_i และกราวด์ ตามกฎของโอห์มได้กระแสดังนี้

$$I = E_i / R_i$$

จากข้อที่ว่าจะไม่มีกระแสไหลเข้าขาอินพุตทั้ง 2 ของออปแอมป์ ดังนั้นกระแส I ทั้งหมดจาก R_i จะไหลผ่านไปยัง R_f ทำให้เกิด V_{Rf} ขึ้น (โดยไม่มีการไหลเข้าอินพุตลบ)

$$V_{Rf} = I \times R_f = E_i / R_i \times R_f$$

จาก V_{RF} ที่ได้เป็นแรงดันตกคร่อม Rf ซึ่งเกิดจากความต่างศักย์ระหว่างกราวด์กับ VO ตามทิศของกระแสในขณะนี้ กระแสไหลสืบเนื่องมาจาก I ผ่านกราวด์ซึ่งอยู่ทางซ้ายของ R_f มายัง VO ซึ่งอยู่ทางขวาของ R_f แสดงให้เห็นว่า V_O ในตอนนี้มีแรงดันต่ำกว่ากราวด์คือ เป็นลบ (แรงดันตกคร่อมขา 2 และขา 6 ของออปแอมป์มีค่าเท่ากับ 0 โวลต์) ดังนั้นในขณะที่ E_i เป็นบวกเทียบกับกราวด์ V_O ก็จะเป็นลบเมื่อเทียบกับกราวด์ (จะได้หักล้างเป็น 0 โวลท์พอดี) เพราะฉะนั้นถ้าคิดว่าขนาดของ V_O เท่ากับ V_{Rf} แล้วเครื่องหมายของ V_O ก็จะกลับกันกับ V_{Rf} ดังนั้นจะได้

$$V_O = -E_i \times (R_f / R_i)$$

จากนิยามของอัตราการขยายแบบลูปปิด A_{CL} เท่ากับ V_O / E_i จะได้

$$\begin{aligned} A_{CL} &= -V_O / E_i \\ &= (-E_i \times [R_f / R_i]) \times (1 / E_i) \\ &= -R_f / R_I \end{aligned}$$

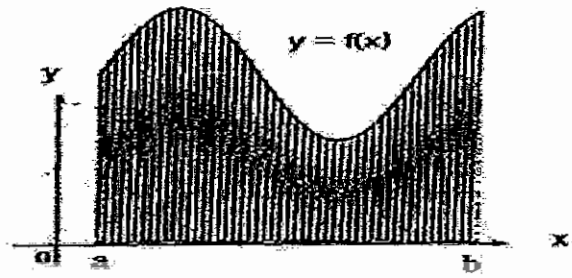
จากเครื่องหมายลบ แสดงว่าขั้วของ V_O จะกลับกับทางด้าน E_i ซึ่งจุดนี้เองที่ทำให้วงจรขยายนี้จึงได้ชื่อว่า วงจรขยายแบบกลับเฟส (Inverting Amplifier) ซึ่งอัตราการขยายของวงจรนี้จะขึ้นอยู่กับ R_f และ R_i เท่านั้น

2.3 ผลบวกรีมันน์และอินทิกรัลจำกัดเขต

สัญญลักษณ์ซิกมาของผลรวมจำกัด (**Sigma Notation for Finite Sums**)

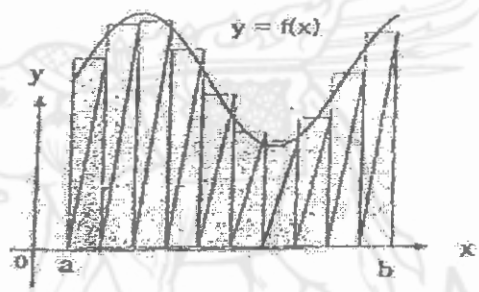
$$\sum_{k=1}^{n} a_k = a_1 + a_2 + a_3 + \ldots\ldots + a_n$$

เมื่อ a เป็นเทอมของผลรวม โดย a_1 เป็นเทอมที่ 1 , a_2 เป็นเทอมที่ 2 , a_k เป็นเทอมที่ 2และ a_n เป็นเทอมที่ n และเทอมสุดท้าย เรียกตัวแปร k ว่า ดัชนีของการบวก (Index of Summation) ค่าของ k วิ่งจาก 1 ถึง n และเรียก 1 ว่า ลิมิตล่าง (Lower Limit) และเรียก n ว่า ลิมิตบนของการบวก (Upper Limit of Summation)

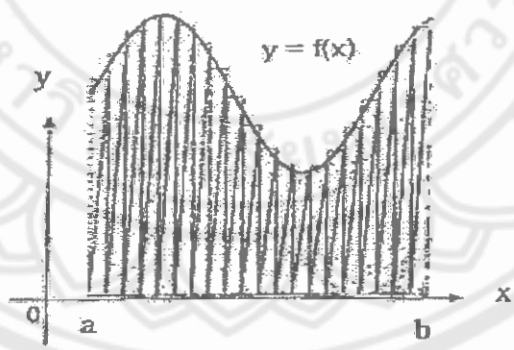


รูปที่ 2.14 พื้นที่ใต้กราฟ y = f(x)

เราสามารถประมาณพื้นที่ในบริเวณส่วนที่แรเงาในรูปที่ 2.14 ได้โดยการแบ่งบริเวณใต้เส้นโค้งออกเป็นสี่เหลี่ยมผืนผ้าให้สิบรูปเท่า ๆ กัน ดังแสดงในรูป 2.15 และแล้วแบ่งสี่เหลี่ยมผืนผ้าให้เล็กลงอีกเป็นยี่สิบรูปเท่า ๆ กันดังแสดงในรูป 2.16



รูปที่ 2.15 การแบ่งพื้นที่ใต้กราฟเป็นสี่เหลี่ยมสิบรูป



รูปที่ 2.16 การแบ่งพื้นที่ใต้กราฟเป็นสี่เหลี่ยมยี่สิบรูป

สำหรับวิธีการหาพื้นที่โดยใช้ผลรวมของรีมันน์เริ่มต้นด้วยการแบ่งช่วง [a,b] ออกเป็นช่วงย่อย ๆ n ช่วง แต่ละช่วงกว้าง $[b-a]/n$ เท่ากันหมด เขียนแทนด้วย Δx จุดแบ่งซึ่งอยู่ระหว่าง a และ b คือ $x_1, x_2, x_3, \ldots\ldots, x_{n-1}$ เมื่อ $a = x_1$, และ $b = x_{n+1}$ แล้วลากเส้นตามแนวดิ่งผ่านจุดแบ่งเหล่านี้ พื้นที่จึงถูกแบ่งออกเป็น n ส่วนเล็ก ๆ (Strips) ถ้าเราประมาณค่าพื้นที่แต่ละส่วนด้วยพื้นที่ของสี่เหลี่ยมผืนผ้าที่แรเงาเล็ก ๆ ชิ้นใด ๆ ซึ่งเรียกว่าสี่เหลี่ยมผืนผ้าชิ้นที่ i ซึ่งพื้นที่มีค่าเท่ากับกว้างคูณยาวเพราะว่าด้าน

กว้างของสี่เหลี่ยมผืนผ้าชิ้นที่ i คือ Δx และด้านยาวแทนด้วยส่วนสูง $f(x_i)$ และพื้นที่ของสี่เหลี่ยมผืนผ้าชิ้นที่ i แทนด้วย A_i
ดังนั้น

$$A_i = f(x_i)\,\Delta x$$

และพื้นที่ใต้เส้นโค้ง $y = f(x)$ กับแกน x บนช่วง [a,b] สามารถประมาณด้วย $\sum_{i=1}^{n} f(x_i)\,\Delta x$
ถ้า n มีค่ามากขึ้น ๆ ความกว้างของช่วงแต่ละช่วงจะเล็กลง ๆ ค่าประมาณของพื้นที่ดังกล่าวจะเข้าใกล้พื้นที่ใต้เส้นโค้ง A นั่นคือ

$$A = \lim_{n\to\infty} \sum_{i=1}^{n} f(x)\,\Delta x$$

ถ้าลิมิตของผลรวมซิกมานี้หาค่าได้เป็นจำนวนจริงลิมิตนี้เราเรียกว่า อินทิกรัลจำกัดเขต (**Definite Integral**) ของ f(x) จาก a ไป b และเขียนได้ดังนิยามต่อไปนี้

นิยาม ถ้า f เป็นฟังก์ชันต่อเนื่องบนช่วง [a,b] อินทิกรัลจำกัดเขต ของ f จาก a ไป b เขียนแทนด้วย

$$\int_a^b f(x)dx = \lim_{n\to\infty} \sum_{i=1}^{n} f(x)\,\Delta x$$

ถ้าลิมิตหาค่าได้ เมื่อ $\Delta x = (b-a)/n$ และ x_i เป็นค่าใด ๆ ในช่วงที่ I เราเรียก a ว่า ลิมิตล่างของอินทิกรัล และเรียก b ว่า ลิมิตบนของอินทิกรัล ถ้าลิมิตในนิยามนี้หาค่าได้ เราจะกล่าวว่า f หา อินทิเกรตแบบรีมันน์ได้บน [a,b] หรือ กล่าวย่อ ๆ ว่า f อินทิเกรตได้ (**Intrigrable**) บน [a,b]