บทที่ 5
ตัวดำเนินการของโปรแกรม C51

5.1 บทนำ

โดยทั่วไปโปรแกรมภาษาซีใช้ตัวดำเนินการเป็นตัวเชื่อมในการเขียนโปรแกรมเพื่อหาผลลัพธ์ที่ต้องการ ตัวดำเนินการคือเครื่องหมายกำหนดกรรมวิธีทางคณิตศาสตร์ พีชคณิตบูลีน การเปรียบเทียบระหว่างข้อมูลสองตัวซึ่งเรียกว่าตัวถูกดำเนินการ โดยอาจมีค่าเป็นตัวเลข ข้อความ ค่าคงที่หรือตัวแปรต่างๆ สำหรับตัวดำเนินการของโปรแกรม C51 สามารถแบ่งได้ 3 กลุ่มคือ ตัวดำเนินการทางคณิตศาสตร์ ตัวดำเนินการเปรียบเทียบและลอจิก และตัวดำเนินการทางบิต ในบทนี้ได้นำเสนอตัวดำเนินการที่เกี่ยวข้องดังกล่าว

5.2 ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการทางคณิตศาสตร์เป็นตัวเชื่อมในการเขียนโปรแกรมเพื่อหาผลลัพธ์จากการคำนวณ ซึ่งสามารถกระทำกับข้อมูลได้หลายแบบ เช่น การบวก การลบ การคูณ การหาร เป็นต้น ตัวดำเนินการทางคณิตศาสตร์แสดงในตารางที่ 5.1

ตารางที่ 5.1 ตัวดำเนินการทางคณิตศาสตร์สำหรับโปรแกรม C51

ตัวดำเนินการ	ความหมาย
+	การบวก
-	การลบ
*	การคูณ
/	การหาร
%	การหารแบบเอาเศษ
++	การเพิ่มค่าขึ้นอีกค่าหนึ่ง
--	การลดค่าลงอีกค่าหนึ่ง
+=	การบวกขึ้นอีกด้วยค่าทางขวามือ
-=	การลดค่าลงอีกค่าด้วยค่าทางขวามือ
*=	การคูณด้วยค่าทางขวามือ
/=	การหารด้วยค่าทางขวามือ
%=	การหารด้วยค่าทางขวามือแบบเอาเศษ

ที่มา: ประจิน พลังสันติกุล และชัยวัฒน์ ลิ้มพรจิตรวิไล, 2550, หน้า 162.

การทำงานของตัวดำเนินการทางคณิตศาสตร์ แสดงดังตัวอย่างที่ 5.1–5.5

ตัวอย่างที่ 5.1 โปรแกรมการใช้งานตัวดำเนินการทางคณิตศาสตร์

```
1 #include <REGX52.H>
2   void main (void)
3   {
4   unsigned char a = 12;
5   unsigned char b = 12;
6   a = a+3;
7   b = b-3;
8   }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x080C
a	15
b	9

บรรทัดที่ 4 ตัวแปร a = 12

บรรทัดที่ 5 ตัวแปร b = 12

บรรทัดที่ 6 ตัวแปร a บวก 3 แล้วนำผลลัพธ์ที่ได้เก็บไว้ที่ a

บรรทัดที่ 7 ตัวแปร b ลบ 3 แล้วนำผลลัพธ์ที่ได้เก็บไว้ที่ b

ตัวอย่างที่ 5.2 โปรแกรมการใช้งานตัวดำเนินการทางคณิตศาสตร์

```
1 #include <REGX52.H>
2   void main (void)
3   {
4   unsigned char x,y,z;
5   x = 10;
6   y = x/3;
7   z=x%3;
8   }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x0800
x	10
y	3
z	1

บรรทัดที่ 4 ประกาศตัวแปร x, y, z

บรรทัดที่ 5 ตัวแปร x = 10

บรรทัดที่ 6 ตัวแปร y = x/10 เป็นการหารที่เก็บเอาเฉพาะผลลัพธ์เลขจำนวนเต็ม

บรรทัดที่ 7 ตัวแปร z = x%3 เป็นการเก็บเอาเฉพาะผลลัพธ์เลขที่เป็นเศษของการหาร

ตัวอย่างที่ 5.3 โปรแกรมการใช้งานตัวดำเนินการทางคณิตศาสตร์

```
1  #include <REGX52.H>
2    void main (void)
3    {
4    unsigned char a = 5;
5    unsigned char b = 5;
6    a++;
7    b--;
8    }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x080C
a	6
b	4

บรรทัดที่ 4 ตัวแปร a = 5

บรรทัดที่ 5 ตัวแปร b = 5

บรรทัดที่ 6 กำหนดให้ค่า a บวกกับ 1 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ a

บรรทัดที่ 7 กำหนดให้ค่า b ลบกับ 1 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ b

ตัวอย่างที่ 5.4 โปรแกรมการใช้งานตัวดำเนินการทางคณิตศาสตร์

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    int a = 100;
5    int b = 100;
6    a += 10;
7    b -= 10;
8    }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x0800
a	110
b	90

บรรทัดที่ 4 ตัวแปร a = 100

บรรทัดที่ 5 ตัวแปร b = 100

บรรทัดที่ 6 กำหนดให้ค่า a บวกกับ 10 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ a

บรรทัดที่ 7 กำหนดให้ค่า b ลบกับ 10 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ b

ตัวอย่างที่ 5.5 โปรแกรมการใช้งานตัวดำเนินการทางคณิตศาสตร์

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    int x,y,z;
5    x = y = z = 120;
6    x *= 4;
7    y /= 4;
8    z %= 4;
9    }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x088B
x	480
y	30
z	0

บรรทัดที่ 4 ประกาศตัวแปร x, y, z

บรรทัดที่ 5 ตัวแปร x, y, z มีค่าเท่ากันคือ 120

บรรทัดที่ 6 กำหนดให้ค่า x คูณ 4 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ x

บรรทัดที่ 7 กำหนดให้ค่า y หาร 4 แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ที่ y

บรรทัดที่ 8 กำหนดให้ค่า z หาร 4 แล้วเก็บเอาเฉพาะผลลัพธ์เลขที่เป็นเศษของการหาร

5.3 ตัวดำเนินการเปรียบเทียบและลอจิกระดับไบต์

ตัวดำเนินการเปรียบเทียบและลอจิกโดยทั่วไปใช้สำหรับการตัดสินใจการทำงานของโปรแกรม โดยจะให้ผลลัพธ์จากการตรวจสอบเป็นจริง (True) หรือเป็นเท็จ (False) เท่านั้น โดยหากเงื่อนไขเป็นจริงจะให้ผลลัพธ์เป็น 1 และหากเงื่อนไขเป็นเท็จจะให้ผลลัพธ์เป็น 0 การทำงานของตัวดำเนินการในกลุ่มนี้แสดงได้ดังตารางที่ 5.2

ตารางที่ 5.2 ตัวดำเนินการเปรียบเทียบและลอจิกของข้อมูลระดับไบต์

ตัวดำเนินการของข้อมูลระดับไบต์	ความหมาย
==	เท่ากับ
!=	ไม่เท่ากับ
>	มากกว่า
<	น้อยกว่า
>=	มากกว่าหรือเท่ากับ
<=	น้อยกว่าหรือเท่ากับ
!	นอต (NOT)
&&	แอนด์ (AND)
||	ออร์ (OR)

ที่มา: ประจิน พลังสันติกุล และชัยวัฒน์ ลิ้มพรจิตรวิไล, 2550, หน้า 166.

สำหรับตัวดำเนินการด้านลอจิกที่สำคัญและใช้สำหรับการตัดสินใจการทำงานของโปรแกรม ได้แก่ ตัวดำเนินการ NOT, AND และ OR สามารถอธิบายการทำงานได้ ดังนี้

5.3.1 ตัวดำเนินการ NOT ของข้อมูลระดับไบต์

ผลลัพธ์การกระทำด้วยตัวดำเนินการ NOT (!) จะได้ผลลัพธ์ที่ตรงกันข้าม ดังตารางที่ 5.3

ตารางที่ 5.3 ผลลัพธ์ที่ได้จากตัวดำเนินการ NOT (!) ของข้อมูลระดับไบต์

ค่าเริ่มต้น	ตัวดำเนินการ NOT	ผลลัพธ์ที่ได้
False	! (False)	True
True	! (True)	False

5.3.2 ตัวดำเนินการ AND ของข้อมูลระดับไบต์

ผลลัพธ์ของการตัวดำเนินการ AND จะเป็นเท็จทันทีหากตัวถูกดำเนินการตัวใดตัวหนึ่งหรือทั้งคู่เป็นเท็จ แสดงได้ดังตารางที่ 5.4

ตารางที่ 5.4 ผลลัพธ์ที่ได้จากตัวดำเนินการ AND (&&) ของข้อมูลระดับไบต์

ค่าเริ่มต้น		ตัวดำเนินการ AND	ผลลัพธ์ที่ได้
ตัวแปร 1	ตัวแปร 2		
False	False	False && False	False
False	True	False && True	False
True	False	True && False	False
True	True	True && True	True

5.3.3 ตัวดำเนินการ OR ของข้อมูลระดับไบต์

ผลลัพธ์ของตัวดำเนินการ OR (||) ของข้อมูลระดับไบต์จะเป็นจริง หากตัวถูกดำเนินการตัวใดตัวหนึ่งหรือทั้งคู่เป็นจริง แสดงได้ดังตารางที่ 5.5

ตารางที่ 5.5 ผลลัพธ์ที่ได้จากตัวดำเนินการ OR (||) ของข้อมูลระดับไบต์

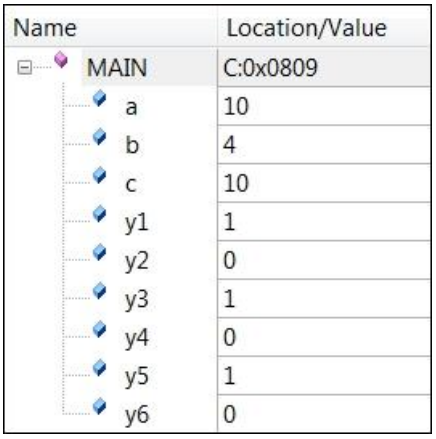
ค่าเริ่มต้น		ตัวดำเนินการ OR	ผลลัพธ์ที่ได้
ตัวแปร 1	ตัวแปร 2		
False	False	False || False	False
False	True	False || True	True
True	False	True || False	True
True	True	True || True	True

การทำงานของตัวดำเนินการเปรียบเทียบและลอจิกแสดงดังตัวอย่างที่ 5.6 และ 5.7

ตัวอย่างที่ 5.6 โปรแกรมการใช้งานตัวดำเนินการเปรียบเทียบ

```
1  #include <REGX52.H>
2  void main (void)
3  {
4    while (1)
5    {
6    unsigned char a = 10;
7    unsigned char b = 4;
8    unsigned char c = 0x0A;
9    unsigned char y1,y2,y3,y4,y5,y6;
10   y1 = a > b;
11   y2 = a > c;
12   y3 = a >= c;
13   y4 = a == b;
14   y5 = a != b;
15   y6 = a != c;
16   }
17 }
```

ผลลัพธ์ของการรันโปรแกรม



Name	Location/Value
MAIN	C:0x0809
a	10
b	4
c	10
y1	1
y2	0
y3	1
y4	0
y5	1
y6	0

บรรทัดที่ 6 ตัวแปร a = 10

บรรทัดที่ 7 ตัวแปร b = 14

บรรทัดที่ 8 ตัวแปร c = 0x0A

บรรทัดที่ 9 ตัวแปร y1, y2, y3, y4, y5, y6

บรรทัดที่ 10 เงื่อนไข a มากกว่า b เงื่อนไขเป็นจริง (y1 = 1)

บรรทัดที่ 11 เงื่อนไข a มากกว่า c เงื่อนไขเป็นเท็จ (y2 = 0)

บรรทัดที่ 12 เงื่อนไข a มากกว่าหรือเท่ากับ c เงื่อนไขเป็นจริงเพราะ 0x0A = 10 (y3= 1)

บรรทัดที่ 13 เงื่อนไข a เท่ากับ b เงื่อนไขเป็นเท็จ (y4 = 0)

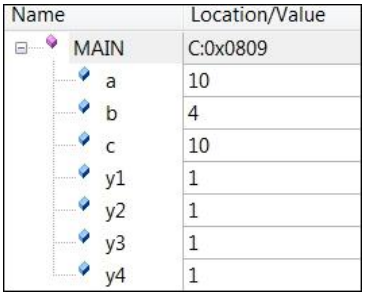
บรรทัดที่ 14 เงื่อนไข a ไม่เท่ากับ b เงื่อนไขเป็นจริง (y5 = 1)

บรรทัดที่ 15 เงื่อนไข a ไม่เท่ากับ c เงื่อนไขเป็นเท็จ (y6 = 0)

ตัวอย่างที่ 5.7 โปรแกรมการใช้งานตัวดำเนินการเปรียบเทียบ

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a = 10;
5    unsigned char b = 4;
6    unsigned char c = 0x0A;
7    bit y1, y2, y3, y4;
8    y1 = (a>b) && (a>=c);
9    y2 = (a != b) && (a>=c);
10   y3 = (a != b) || (a>=c);
11   y4 = (a != b) || !(a != b);
12   }
```

ผลลัพธ์ของการรันโปรแกรม



Name	Location/Value
MAIN	C:0x0809
a	10
b	4
c	10
y1	1
y2	1
y3	1
y4	1

บรรทัดที่ 4-6 ตัวแปร a = 10, b = 4 และ c = 0x0A

บรรทัดที่ 7 ตัวแปร y1, y2, y3, y4

บรรทัดที่ 8 เงื่อนไข (a>b) && (a >= c) เงื่อนไขเป็นจริง (y1 = 1)

บรรทัดที่ 9 เงื่อนไข a != b) && (a >= c) เงื่อนไขเป็นจริง (y2 = 1)

บรรทัดที่ 10 เงื่อนไข (a != b) || (a >= c) เงื่อนไขเป็นจริง (y3 = 1)

บรรทัดที่ 11 เงื่อนไข (a != b) || ! (a != b) เงื่อนไขเป็นจริง (y4 = 1)

5.4 ตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิต

ตัวดำเนินการทางบิตเป็นการกระทำที่เข้าถึงข้อมูลระดับบิต โดยที่ค่าข้อมูลของแต่ละบิตเป็นได้ 2 ค่าคือ 0 หรือ 1 เท่านั้น ตัวดำเนินการทางบิตที่สำคัญแสดงได้ดังตารางที่ 5.6

ตารางที่ 5.6 การกระทำของตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิต

ตัวดำเนินการระดับบิต	ความหมาย
~	กลับค่าของบิตข้อมูล
&	การแอนบิต
|	การออร์บิต
^	การเอ็กซ์คลูซีฟ-ออร์บิต
<<	เลื่อนบิตไปทางซ้าย
>>	เลื่อนบิตไปทางขวา
<<==	เลื่อนบิตไปทางซ้ายแล้วให้เท่ากับ
>>==	เลื่อนบิตไปทางขวาแล้วให้เท่ากับ
&=	ทำการแอนด์แล้วให้เท่ากับ
|=	ทำการออร์แล้วให้เท่ากับ
^=	ทำการเอ็กซ์คลูซีฟ-ออร์แล้วให้เท่ากับ

ที่มา: ประจิน พลังสันติกุล และชัยวัฒน์ ลิ้มพรจิตรวิไล, 2550, หน้า 168.

สำหรับตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิตที่สำคัญ สามารถอธิบายได้ดังนี้

5.4.1 ตัวดำเนินการ NOT ของข้อมูลระดับบิต

ผลลัพธ์ของการกระทำด้วยการกลับค่าของบิตข้อมูลจะได้ผลลัพธ์ที่ตรงกันข้าม แสดงได้ดังตารางที่ 5.7

ตารางที่ 5.7 ผลลัพธ์ของตัวดำเนินการ NOT ของข้อมูลระดับบิต

ค่าเริ่มต้น	ตัวดำเนินการ NOT	ผลลัพธ์ที่ได้
0	~ (0)	1
1	~ (1)	0

5.4.2 ตัวดำเนินการ AND ของข้อมูลระดับบิต

ผลลัพธ์ของการกระทำด้วยตัวดำเนินการแอนด์บิตจะเป็น 0 หากตัวใดตัวหนึ่งหรือทั้งคู่เป็น 0 แสดงได้ดังตารางที่ 5.8

ตารางที่ 5.8 ผลลัพธ์ของตัวดำเนินการ AND ของข้อมูลระดับบิต

ค่าเริ่มต้น		ตัวดำเนินการ AND	ผลลัพธ์ที่ได้
ตัวแปร 1	ตัวแปร 2		
0	0	0 & 0	0
0	1	0 & 1	0
1	0	1 & 0	0
1	1	1 & 1	1

5.4.3 ตัวดำเนินการ OR ของข้อมูลระดับบิต

ผลลัพธ์ของการกระทำด้วยตัวดำเนินการออร์บิตจะเป็น 1 หากตัวใดตัวหนึ่งหรือทั้งคู่เป็น 1 แสดงได้ดังตารางที่ 5.9

ตารางที่ 5.9 ผลลัพธ์ของตัวดำเนินการ OR ของข้อมูลระดับบิต

ค่าเริ่มต้น		ตัวดำเนินการ OR	ผลลัพธ์ที่ได้
ตัวแปร 1	ตัวแปร 2		
0	0	0 | 0	0
0	1	0 | 1	1
1	0	1 | 0	1
1	1	1 | 1	1

5.4.4 ตัวดำเนินการเอ็กซ์คลูซีฟ-ออร์ของข้อมูลระดับบิต

ผลลัพธ์ของการกระทำด้วยตัวดำเนินการเอ็กซ์คลูซีฟ-ออร์บิตจะเป็น 0 ทันที หากตัวถูกดำเนินการทั้งคู่มีค่าเหมือนกัน แสดงได้ดังตารางที่ 5.10

ตารางที่ 5.10 ตัวดำเนินการเอ็กซ์คลูซีฟ-ออร์ของข้อมูลระดับบิต

ค่าเริ่มต้น		ตัวดำเนินการ	ผลลัพธ์ที่ได้
ตัวแปร 1	ตัวแปร 2	เอ็กซ์คลูซีฟ-ออร์	
0	0	0 ^ 0	0
0	1	0 ^ 1	1
1	0	1 ^ 0	1
1	1	1 ^ 1	0

5.4.5 ตัวดำเนินการเลื่อนบิต

ตัวดำเนินการนี้ทำหน้าที่เลื่อนบิตข้อมูลไปทางซ้ายหรือขวา โดยการเลื่อนบิตจะต้องระบุจำนวนครั้งการเลื่อนด้วยว่า ต้องการให้มีการเลื่อนกี่ครั้ง เช่น

```
a = a << 4;      // ให้เลื่อนบิตค่าข้อมูล a ไปทางซ้าย 4 ครั้งแล้วผลลัพธ์ที่ได้เก็บไว้ที่
                    a เหมือนเดิม
b = b >> 1;      // ให้เลื่อนบิตค่าข้อมูล b ไปทางขวา 1 ครั้งแล้วผลลัพธ์ที่ได้เก็บไว้ที่
                    b เหมือนเดิม
```

ตัวอย่างการทำงานของตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิตแสดงดังตัวอย่างที่ 5.8–5.11

ตัวอย่างที่ 5.8 โปรแกรมการใช้งานตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิต

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a = 0x05;
5    unsigned char b = 0x07;
6    unsigned char y1,y2,y3,y4;
7    y1 = ~a ;
8    y2 = a & b;
9    y3 = a | b;
10   y4 = a ^ b;
11   }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x0800
a	0x05
b	0x07
y1	0xFA 'ำ'
y2	0x05
y3	0x07
y4	0x02

บรรทัดที่ 4 ตัวแปร a = 5

บรรทัดที่ 5 ตัวแปร b = 7

บรรทัดที่ 6 ตัวแปร y1, y2, y3, y4

บรรทัดที่ 7 ดำเนินการกลับค่าของบิตข้อมูลตัวแปร a แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y1

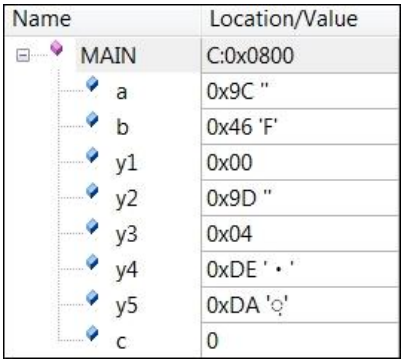
บรรทัดที่ 8 ดำเนินการแอนด์บิตระหว่าง a กับ b แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y2

บรรทัดที่ 9 ดำเนินการเอ็กซ์คลูซีฟ-ออร์บิตระหว่างตัวแปร a กับ b แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y3

ตัวอย่างที่ 5.9 โปรแกรมการใช้งานตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิต

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a, b, y1, y2, y3, y4, y5;
5    bit c;
6    a = 0x9c;
7    b = 0x46;
8    c = 0;
9    y1 = a & c;
10   y2 = a | (~c);
11   y3 = a & b;
12   y4 = a | b;
13   y5 = a ^ b;
14   }
```

ผลลัพธ์ของการรันโปรแกรม



Name	Location/Value
MAIN	C:0x0800
a	0x9C ''
b	0x46 'F'
y1	0x00
y2	0x9D ''
y3	0x04
y4	0xDE '•'
y5	0xDA '♀'
c	0

บรรทัดที่ 6 ตัวแปร a = 0x9C (0b10011100)

บรรทัดที่ 7 ตัวแปร b = 0x46 (0b01000110)

บรรทัดที่ 8 ตัวแปร c = 0

บรรทัดที่ 9 ดำเนินการแอนด์บิตระหว่างตัวแปร a กับ c ผลลัพธ์เก็บไว้ที่ตัวแปร y1

บรรทัดที่ 10 ดำเนินการออร์บิตระหว่างตัวแปร a กับ ~c ผลลัพธ์เก็บไว้ที่ตัวแปร y2

บรรทัดที่ 11 ดำเนินการแอนด์บิตระหว่างตัวแปร a กับ b ผลลัพธ์เก็บไว้ที่ตัวแปร y3

บรรทัดที่ 12 ดำเนินการออร์บิตระหว่างตัวแปร a กับ b ผลลัพธ์เก็บไว้ที่ตัวแปร y4

บรรทัดที่ 13 ดำเนินการเอ็กซ์คลูซีฟ-ออร์บิตระหว่างตัวแปร a กับ b ผลลัพธ์เก็บไว้ที่ตัวแปร y5

จากตัวอย่างที่ 5.9 สามารถแสดงรายละเอียดการดำเนินการทางบิตได้ดังตารางที่ 5.11

ตารางที่ 5.11 ผลลัพธ์ของตัวดำเนินการเชิงเปรียบเทียบและเชิงลอจิกของข้อมูลระดับบิต

ตัวแปร	**บิตข้อมูล**								**ผลลัพธ์**
	b7	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**	
a	1	0	0	1	1	1	0	0	0x9C
b	0	1	0	0	0	1	1	0	0x46
c	-	-	-	-	-	-	-	0	0
~c	-	-	-	-	-	-	-	1	1
y1 = a & c	0	0	0	0	0	0	0	0	0x00
y2 = a | (~c)	1	0	0	1	1	1	0	1	0x9D
y3 = a & b	0	0	0	0	0	1	0	0	0x04
y4 = a | b	1	1	0	1	1	1	1	0	0xDE
y5 = a ^ b	1	1	0	1	1	0	1	0	0xDA

ตัวอย่างที่ 5.10 โปรแกรมการใช้งานตัวดำเนินการเลื่อนข้อมูล

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a, y1, y2, y3, y4;
5    a = 0x01;
6    y1 = a <<1;
7    y2 = a <<2;
8    y3 = a <<3;
9    y4 = a <<4;
10   }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x0800
a	0x01
y1	0x02
y2	0x04
y3	0x08
y4	0x10

บรรทัดที่ 5 ตัวแปร a = 0x01 (0b00000001)

บรรทัดที่ 6 ตัวแปร a เลื่อนบิตข้อมูลไปด้านซ้าย 1 ครั้ง ผลลัพธ์เก็บไว้ที่ตัวแปร y1

บรรทัดที่ 7 ตัวแปร a เลื่อนบิตข้อมูลไปด้านซ้าย 2 ครั้ง ผลลัพธ์เก็บไว้ที่ตัวแปร y2

บรรทัดที่ 8 ตัวแปร a เลื่อนบิตข้อมูลไปด้านซ้าย 3 ครั้ง ผลลัพธ์เก็บไว้ที่ตัวแปร y3

บรรทัดที่ 9 ตัวแปร a เลื่อนบิตข้อมูลไปด้านซ้าย 4 ครั้ง ผลลัพธ์เก็บไว้ที่ตัวแปร y4

จากตัวอย่างที่ 5.10 สามารถแสดงรายละเอียดการดำเนินการทางบิตได้ดังตารางที่ 5.12

ตารางที่ 5.12 ผลลัพธ์ของตัวดำเนินการเลื่อนบิต

ตัวแปร	**บิตข้อมูล**								**ผลลัพธ์**
	b7	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**	
a	0	0	0	0	0	0	0	1	0x01
y1 = a << 1	0	0	0	0	0	0	1	0	0x02
y2 = a << 2	0	0	0	0	0	1	0	0	0x04
y3 = a << 3	0	0	0	0	1	0	0	0	0x08
y4 = a << 4	0	0	0	1	0	0	0	0	0x10

ตัวอย่างที่ 5.11 โปรแกรมการใช้งานตัวดำเนินการทางบิต

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a, b, c;
5    a = b = c = 0xAA;
6    a &= 0xF0;
7    b |= 0xF0;
8    c ^= 0x55;
9    }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x0800
a	0xA0 ' '
b	0xFA 'ฺ'
c	0xFF ' '

บรรทัดที่ 5 ตัวแปร a = b = c = 0xAA (0b10101010)

บรรทัดที่ 6 ตัวแปร a แอนด์บิตกับข้อมูล 0xF0 แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร a

บรรทัดที่ 7 ตัวแปร b ออร์บิตกับข้อมูล 0xF0 แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร b

บรรทัดที่ 8 ตัวแปร c เอ็กซ์คลูซีฟ-ออร์บิตกับข้อมูล 0xF0 แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร c

จากตัวอย่างที่ 5.11 สามารถแสดงรายละเอียดการดำเนินการทางบิตได้ดังตารางที่ 5.13

ตารางที่ 5.13 ผลลัพธ์ของตัวดำเนินการทางบิต

ตัวแปร	บิตข้อมูล								ผลลัพธ์
	b7	b6	b5	b4	b3	b2	b1	b0	
a = b = c	1	0	1	0	1	0	1	0	0xAA
0xF0	1	1	1	1	0	0	0	0	0xF0
0x55	0	1	0	1	0	1	0	1	0x55
a &= 0xF0	1	0	1	0	0	0	0	0	0XA0
b |= 0xF0	1	1	1	1	1	0	1	0	0xFA
c ^= 0x55	1	1	1	1	1	1	1	1	0xFF

5.5 การใช้ประโยชน์จากตัวดำเนินการทางลอจิกกับข้อมูลในระดับบิต

ในบางครั้งการเขียนโปรแกรมอาจมีความจำเป็นต้องตัดข้อมูลที่ได้บางส่วนออกเพื่อนำผลลัพธ์ที่ได้ไปใช้ประโยชน์อย่างอื่น ในที่นี้ได้ยกตัวอย่างการเลือกใช้ข้อมูลบางบิตที่ต้องการไปใช้งานอธิบายได้ดังนี้

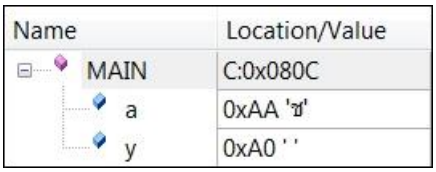
5.5.1 การนำเฉพาะข้อมูล 4 บิตบนไปใช้งาน

การนำเฉพาะข้อมูล 4 บิตบนไปใช้งาน ต้องเคลียร์ข้อมูล 4 บิตล่างให้เป็น 0 มีวิธีการทำคือนำข้อมูลดังกล่าวไปแอนด์กับ 0xF0 สามารถแสดงการใช้งานได้ดังตัวอย่างที่ 5.12

ตัวอย่างที่ 5.12 โปรแกรมการนำเฉพาะข้อมูล 4 บิตบนไปใช้งาน

```
1  #include <REGX52.H>
2  void main (void)
3    {
4      unsigned char a, y;
5      a = 0xAA;
6      y = a & 0xF0;
7    }
```

ผลลัพธ์ของการรันโปรแกรม



บรรทัดที่ 5 ตัวแปร a = 0xAA (0b10101010)

บรรทัดที่ 6 แอนด์บิตข้อมูลระหว่างตัวแปร a กับ 0xF0 แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y

จากตัวอย่างที่ 5.12 สามารถแสดงรายละเอียดการนำเฉพาะข้อมูล 4 บิตบนไปใช้งานได้ดังตารางที่ 5.14

ตารางที่ 5.14 ผลลัพธ์ของการนำเฉพาะข้อมูล 4 บิตบนไปใช้งาน

ตัวแปร	บิตข้อมูล								ผลลัพธ์
	b7	b6	b5	b4	b3	b2	b1	b0	
a	1	0	1	0	1	0	1	0	0xAA
-	1	1	1	1	0	0	0	0	0xF0
a & 0xF0	1	0	1	0	0	0	0	0	0xA0

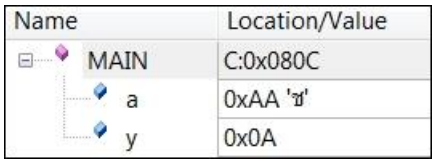
5.5.2 การนำเฉพาะข้อมูล 4 บิตล่างไปใช้งาน

การนำเฉพาะข้อมูล 4 บิตล่างไปใช้งาน ต้องเคลียร์ข้อมูล 4 บิตบนให้เป็น 0 โดยให้นำข้อมูลที่ต้องการไปแอนด์กับ 0x0F จะทำให้ค่า 4 บิตบนกลายเป็น 0 สามารถแสดงการใช้งานได้ดังตัวอย่างที่ 5.13

ตัวอย่างที่ 5.13 โปรแกรมการนำเฉพาะข้อมูล 4 บิตล่างไปใช้งาน

```
1  #include <REGX52.H>
2  void main (void)
3    {
4      unsigned char a, y;
5      a = 0xAA;
6      y = a & 0x0F;
7    }
```

ผลลัพธ์ของการรันโปรแกรม



บรรทัดที่ 5 ตัวแปร a = 0xAA (0b10101010)

บรรทัดที่ 6 ตัวแปร a แอนด์บิตกับข้อมูล 0x0F แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y

จากตัวอย่างที่ 5.13 สามารถแสดงรายละเอียดการนำเฉพาะข้อมูล 4 บิตล่างไปใช้งาน ได้ดังตารางที่ 5.15

ตารางที่ 5.15 ผลลัพธ์ของการนำเฉพาะข้อมูล 4 บิตล่างไปใช้งาน

ตัวแปร	**บิตข้อมูล**								**ผลลัพธ์**
	b7	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**	
a	1	0	1	0	1	0	1	0	0xAA
-	0	0	0	0	1	1	1	1	0x0F
a & 0x0F	0	0	0	0	1	0	1	0	0x0A

5.5.3 การเลือกแปลงข้อมูลบางบิต

ถ้าต้องการให้ข้อมูลที่บิตใดเป็น 1 ให้นำข้อมูลนั้นไปออร์กับค่าคงที่ค่าหนึ่งที่มีบิตตำแหน่งดังกล่าวเป็น 1 เนื่องจากข้อมูล 1 ออร์กับค่าใดๆ จะได้ผลลัพธ์เป็น 1 แสดงได้ดังตัวอย่างที่ 5.14

ตัวอย่างที่ 5.14 โปรแกรมการแปลงข้อมูลบางบิต

ต้องการให้บิต 0 บิต 4 และบิต 7 ของตัวแปร a เป็น 1 โดยที่บิตอื่นยังมีค่าเท่าเดิมโดยสมมุติว่าตัวแปร a มีค่าเท่ากับ 0xAA ดังนั้นค่าคงที่ที่จะนำมาออร์ด้วยคือ 10010001B หรือ 0x91

```
1  #include <REGX52.H>
2  void main (void)
3    {
4    unsigned char a, y;
5    a = 0xAA;
6    y = a | 0x91;
7    }
```

ผลลัพธ์ของการรันโปรแกรม

Name	Location/Value
MAIN	C:0x080C
a	0xAA 'ช'
y	0xBB 'บ'

บรรทัดที่ 5 ตัวแปร a = 0xAA (0b10101010)

บรรทัดที่ 6 ตัวแปร a ออร์บิตกับข้อมูล 0x91 แล้วนำผลลัพธ์เก็บไว้ที่ตัวแปร y

จากตัวอย่างที่ 5.14 สามารถแสดงรายละเอียดการทำข้อมูลบิต 0 บิต 4 และบิต 7 เป็น 1 ได้ดังตารางที่ 5.16

ตารางที่ 5.16 ผลลัพธ์ของการทำให้ข้อมูลบิต 0, 4 และ 7 มีค่าเป็น 1

ตัวแปร	บิตข้อมูล								ผลลัพธ์
	b7	b6	b5	b4	b3	b2	b1	b0	
a	1	0	1	0	1	0	1	0	0xAA
-	1	0	0	1	0	0	0	1	0x91
a | 0x91	1	0	1	1	1	0	1	1	0xBB

5.6 สรุป

ในบทนี้ผู้เรียนได้ทำความเข้าใจเกี่ยวกับตัวดำเนินการ ซึ่งเป็นประโยชน์อย่างมากในการพัฒนาโปรแกรม C51 เพราะว่าการดำเนินการต่าง ๆ ได้แก่การดำเนินการทางคณิตศาสตร์ การเปรียบเทียบและลอจิก และการกระทำทางบิต เป็นส่วนสำคัญในการดำเนินงานของโปรแกรม หากเกิดข้อผิดพลาดในส่วนการดำเนินการเหล่านี้ เช่น มีการกำหนดและใช้งานตัวดำเนินการไม่ถูกต้องตามเงื่อนไขการทำงาน อาจส่งผลกระทบผลต่อการทำงานของโปรแกรมซึ่งอาจทำให้เกิดความเสียหายต่อการพัฒนางานไมโครคอนโทรลเลอร์ได้

5.7 แบบฝึกหัดท้ายบท

แบบฝึกหัดมีทั้งหมด 4 ข้อ ให้นักศึกษาทำแบบฝึกหัดทุกข้อ

1. จงคำนวณค่าของนิพจน์ต่อไปนี้

   1.1) 13 / 4

   1.2) 13 % 4

   1.3) -13 * 4

   1.4) -13 * 4 + 8

   1.5) 10 * 7 / 4

   1.6) 5 * (6 % 4)

   1.7) 1.9 + ‘8’

2. จงหาค่าทางคณิตศาสตร์ต่อไปนี้โดยใช้โปรแกรม C51 กำหนดตัวแปร a, b, c และ d ให้มีค่าเป็น 1, 2, 3 และ 4 ตามลำดับและคำนวณค่าคำตอบออกมาเป็นชนิด float

   2.1) $\frac{ab}{a+b}$

   2.2) $a+\frac{b}{c^2}$

   2.3) $\frac{\frac{a}{b}+c}{a-\frac{b}{c}}$

   2.4) $\frac{a+b}{ab}$

   2.5) $\left(\frac{a+b}{c-d}\right)^2$

2.6) $\dfrac{\dfrac{ab+bc+ac}{a+b+c}}{abc}$

3. จงอธิบายความแตกต่างระหว่างนิพจน์ต่อไปนี้ และหาค่านิพจน์เหล่านี้เมื่อ x มีค่าเท่ากับ 0, 1 และ 10 ตามลำดับ

3.1) x && 0x01

3.2) x & 0x01

3.3) x || 0x0F

3.4) x | 0x0F

4. จงใส่ค่าผลลัพธ์ในช่องว่างให้ถูกต้อง

4.1) ตัวดำเนินการเปรียบเทียบและลอจิก

X	Y	X<Y	X<=Y	X>Y	X>=Y	X==Y	X!=Y
3	3						
3	4						
4	3						

4.2) ตัวดำเนินการทางบิต

X	Y	X&&Y	X||Y	!X	!Y	X&Y	X|Y	X!=0 && Y!=0
0	0							
0	7							
5	0							
5	7							
8	7							