บทที่ 6

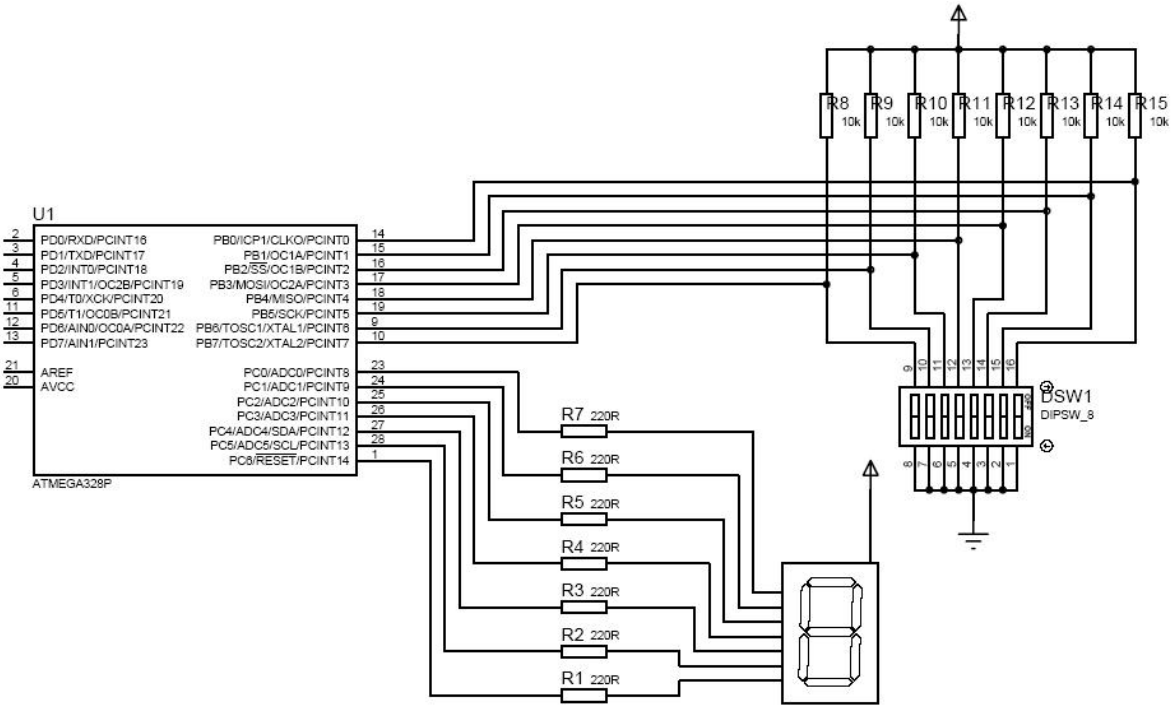
การเขียนโปรแกรมไมโครคอนโทรลเลอร์ AVR ด้วยภาษาซี

การเขียนโปรแกรมไมโครคอนโทรลเลอร์ด้วยภาษาซี ได้รับความนิยมเพิ่มมากขึ้นในปัจจุบันเนื่องจากโครงสร้างของภาษามีความง่ายกว่า ส่งผลให้การพัฒนาระบบทำได้รวดเร็วกว่าการใช้ภาษาแอสเซมบลี้ อย่างไรก็ดี การใช้ภาษาแอสเซมบลี้แม้จะยุ่งยากกว่าแต่ก็จะได้โปรแกรมที่มีขนาดเล็กและมีความเร็วในการทำงานที่สูงกว่า

ตารางที่ 6.1 เปรียบเทียบคุณสมบัติของโปรแกรมภาษาแอสเซมบลี้และภาษาซี

คุณสมบัติ	ภาษาแอสเซมบลี้	ภาษาซี
ความง่ายในการเขียน	ยากกว่า	ง่ายกว่าภาษาแอสเซมบลี้
ความเร็วในการพัฒนาโปรแกรม	ทำได้ช้า	ทำได้รวดเร็วกว่าภาษาแอสเซมบลี้
การนำโปรแกรมไปใช้ใหม่ในสถาปัตยกรรมอื่น	ทำไม่ได้	ทำได้โดยไม่ต้องดัดแปลงโปรแกรมมากนัก
ขนาดของออบเจกต์โปรแกรม	เล็กกว่าภาษาซี	ขนาดใหญ่ (กินหน่วยความจำมากกว่า)
ความเร็วในการทำงานของโปรแกรม	เร็วกว่าภาษาซี	ช้ากว่า

ตัวอย่างที่ 6.1 จงเขียนโปรแกรมเพื่ออ่านค่าจากดิพสวิทช์และแสดงจำนวนบิตที่มีสถานะลอจิกสูงออกทางแอลอีดี 7 เซกเมนต์ ดังแสดงวงจรให้เห็นในรูปที่ 6.1 กำหนดให้เขียนโปรแกรมด้วยภาษาแอสเซมบลี้และภาษาซี และทำการเปรียบเทียบคุณสมบัติของโปรแกรมที่ได้



รูปที่ 6.1 วงจรสำหรับอ่านค่าจากสวิทช์และแสดงผลออกทางแอลอีดี

โปรแกรม ch05_001.c	โปรแกรม ch05_001.asm

```
#include <avr/io.h>
int main(void)
{
   DDRB = 0x00;    // set port B as input
   DDRC = 0xFF;    // set port C as output

   unsigned char SWITCH_V, DISP, i, count, mask, test_bit;

   while (1)
   {
      //---read input switch via portB
      SWITCH_V = PINB;

      count = 0;
      mask = 0x01;

      for (i=0;i<8;i++)
      {
         test_bit =  SWITCH_V & (mask << i);

         if (test_bit)
            count++;
      }

            switch(count)
            {
               case 0: DISP= 0b00111111;   break;
               case 1: DISP= 0b00000110;   break;
               case 2: DISP= 0b01011011;   break;
               case 3: DISP= 0b01001111;   break;
               case 4: DISP= 0b01100110;   break;
               case 5: DISP= 0b01101101;   break;
               case 6: DISP= 0b01111101;   break;
               case 7: DISP= 0b00000111;   break;
               case 8: DISP= 0b01111111;   break;
               case 9: DISP= 0b01101111;   break;
               case 10:DISP= 0b01110111;   break;
               case 11:DISP= 0b01111100;   break;
               case 12:DISP= 0b00111001;   break;
               case 13:DISP= 0b01011110;   break;
               case 14:DISP= 0b01111001;   break;
               default:DISP= 0b01110001;   //15
            }
      PORTC = ~(DISP);
   }
}
```

```
.INCLUDE "m328Pdef.inc"

.EQU        ALL_PIN_OUT = 0xff
.EQU        ALL_PIN_IN  = 0x00
.DEF        VAR_A =    r16
.DEF        ZERO  =    r17
.DEF        COUNT = r18


;-----------------------------------------------------------------------------
; code segment
;-----------------------------------------------------------------------------
.CSEG
.ORG 0x0000

            ldi         VAR_A,ALL_PIN_OUT
            out         DDRC,VAR_A ;set port C as output

            ldi         VAR_A,ALL_PIN_IN
            out         DDRB,VAR_A ;set PORT B as input

            ldi         ZERO,0x00 ;ZERO <= 0

MAIN:       ;---read 4 switches from portC(lower4bits) and keep in VAR_A
            in          VAR_A,PINB              ;read 8-switches
            mov         COUNT,ZERO              ;clear count

LOOP:       lsl         VAR_A
            adc         COUNT,ZERO
            cp          VAR_A,ZERO
            brne        LOOP
            mov         VAR_A,COUNT

            call        DISP_7SEG
            rjmp        MAIN

DISP_7SEG:

            ;---put number to display in VAR_A before use this subroutine
            ldi         ZL,low(TB_7SEGMENT*2);load Z register low
            ldi         ZH,high(TB_7SEGMENT*2);load Z register high

            add         ZL,VAR_A
            adc         ZH,ZERO                 ; Z <= Z + VAR_A
            lpm                                 ;R0 <- [Z]

            com         r0
            out         PORTC,r0
            ret

;-----------------------------------------------------------------------------
; Table for 7-segment display
;-----------------------------------------------------------------------------
            ;     hgfedcba    hgfedcba
TB_7SEGMENT:
.DB         0b00111111, 0b00000110          ;0 and 1    0x3F    0x05
.DB         0b01011011, 0b01001111          ;2 and 3    0x5B    0x4F
.DB         0b01100110, 0b01101101          ;4 and 5    0x66    0x6D
.DB         0b01111101, 0b00000111          ;6 and 7    0x7D    0x07
.DB         0b01111111, 0b01101111          ;8 and 9    0x7F    0x6F
.DB         0b01110111, 0b01111100          ;A and B    0x77    0x7C
.DB         0b00111001, 0b01011110          ;C and D    0x39    0x5E
.DB         0b01111001, 0b01110001          ;E and F    0x79    0x71

.DSEG
```

ขนาดของโปรแกรมหลังคอมไพล์ 316 bytes	ขนาดของโปรแกรมหลังแอสเซมเบลอร์ 62 bytes

จะเห็นว่าขนาดของโปรแกรมที่เขียนด้วยภาษาแอสเซมบลี้มีขนาดเล็กกว่าถึง 5 เท่า อย่างไรก็ดี โปรแกรม ch05_001.c ยังมีข้อเสียคือ การใช้ switch-case ทำให้ขนาดของโปรแกรมใหญ่ วิธีการที่มีประสิทธิภาพดีกว่า คือ การใช้ Look-up table ดังแสดงในโปรแกรม ch05_002.c

โปรแกรม ch05_001.c	โปรแกรม ch05_002.c

#include <avr/io.h>
int main(void)
{
DDRB = 0x00; // set port B as input
DDRC = 0xFF; // set port C as output
 	
unsigned char SWITCH_V, DISP, i, count, mask, test_bit;
 	
while (1)
{
//---read input switch via portB
SWITCH_V = PINB;
 	
count = 0;
mask = 0x01;
 	
for (i=0;i<8;i++)
{
test_bit = SWITCH_V & (mask << i);
 	
if (test_bit)
count++;
}
 	
switch(count)
{
case 0: DISP= 0b00111111; break;
case 1: DISP= 0b00000110; break;
case 2: DISP= 0b01011011; break;
case 3: DISP= 0b01001111; break;
case 4: DISP= 0b01100110; break;
case 5: DISP= 0b01101101; break;
case 6: DISP= 0b01111101; break;
case 7: DISP= 0b00000111; break;
case 8: DISP= 0b01111111; break;
case 9: DISP= 0b01101111; break;
case 10:DISP= 0b01110111; break;
case 11:DISP= 0b01111100; break;
case 12:DISP= 0b00111001; break;
case 13:DISP= 0b01011110; break;
case 14:DISP= 0b01111001; break;
default:DISP= 0b01110001; //15
}
PORTC = ~(DISP);
}
}
	#include <avr/io.h>
	int main(void)
	{
	DDRB = 0x00; // set port B as input
	DDRC = 0xFF; // set port C as output
	
	unsigned char SWITCH_V, DISP, i, count, mask, test_bit;
	unsigned char LOOKUPTB[] = { 0b00111111,
	0b00000110,
	0b01011011,
	0b01001111,
	0b01100110,
	0b01101101,
	0b01111101,
	0b00000111,
	0b01111111,
	0b01101111,
	0b01110111,
	0b01111100,
	0b00111001,
	0b01011110,
	0b01111001,
	0b01110001 };
	while (1)
	{
	//---read input switch via portB
	SWITCH_V = PINB;
	
	count = 0;
	mask = 0x01;
	
	for (i=0;i<8;i++)
	{
	test_bit = SWITCH_V & (mask << i);
	
	if (test_bit)
	count++;
	}
	
	DISP = LOOKUPTB[count];
	
	PORTC = ~(DISP);
	}
	}

ขนาดของโปรแกรมหลังคอมไพล์ 316 bytes	ขนาดของโปรแกรมหลังคอมไพล์ 276 bytes

จะเห็นว่าโปรแกรม ch05_002.c มีขนาดเล็กกว่าโปรแกรม ch05_001.c ถึง 12.66 เปอร์เซ็นต์ ดังนั้น การเขียนโปรแกรมที่ดีจะช่วยให้ได้โปรแกรมที่มีขนาดเล็กและทำงานรวดเร็วขึ้น ซึ่งเป็นสิ่งที่ผู้เขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ควรตระหนักให้มาก เนื่องจากสภาพแวดล้อมที่มีอยู่มีทรัพยากรให้ใช้จำกัดทั้งในแง่ขนาดของหน่วยความจำและความเร็วของซีพียู

ชนิดของข้อมูล (Data type)

คอมไพเลอร์ WinAVR สนับสนุนข้อมูลชนิดจำนวนเต็มดังต่อไปนี้

```
typedef signed char int8_t;
typedef unsigned char uint8_t;
typedef int int16_t;
typedef unsigned int uint16_t;
typedef long int32_t;
typedef unsigned long uint32_t;
typedef long long int64_t;
typedef unsigned long long uint64_t;
```