Pearson New International Edition

The 8051 Microcontroller
A Systems Approach

Mazidi McKinlay Mazidi

First Edition

ALWAYS LEARNING™

PEARSON®

Pearson New International Edition

The 8051 Microcontroller
A Systems Approach
Mazidi McKinlay Mazidi
First Edition

PEARSON®

Bit and sfr

The bit data type allows access to single bits of bit-addressable memory spaces 20–2FH. Notice that while the sbit data type is used for bit-addressable SFRs, the bit data type is used for the bit-addressable section of RAM space 20–2FH. To access the byte-size SFR registers, we use the sfr data type. We will see the use of sbit, bit, and sfr data types in the next section. See Table 1.

Time delay

There are two ways to create a time delay in 8051 C:

1. Using a simple `for` loop
2. Using the 8051 timers

In either case, when we write a time delay we must use the oscilloscope to measure the duration of our time delay. Next, we use the `for` loop to create time delays.

In creating a time delay using a `for` loop, we must be mindful of three factors that can affect the accuracy of the delay.

1. The 8051 design: The original 8051 microcontroller was designed in 1980. Since then, both the fields of IC technology and microprocessor architectural design have seen great advancements. For instance, the number of machine cycles and the number of clock periods per machine cycle vary among different versions of the 8051/52 microcontroller. While the original 8051/52 design used 12 clock periods per machine cycle, many of the newer generations of the 8051 use fewer clocks per machine cycle. For example, the DS5000 uses 4 clock periods per machine cycle, while the DS89C4x0 uses only 1 clock per machine cycle.
2. The crystal frequency connected to the X1–X2 input pins: The duration of the clock period for the machine cycle is a function of this crystal frequency.
3. Compiler choice: The third factor that affects the time delay is the compiler used to compile the C program. When we program in Assembly language, we can control the exact instructions and their sequences used in the DELAY subroutine. In the case of C programs, it is the C compiler that converts the C

Table 1. Some Widely Used Data Types for 8051 C

Data Type	Size in Bits	Data Range/Usage
unsigned char	8-bit	0 to 255
char (signed)	8-bit	–128 to +127
unsigned int	16-bit	0 to 65535
int (signed)	16-bit	–32,768 to +32,767
sbit	1-bit	SFR bit-addressable only
bit	1-bit	RAM bit-addressable only
sfr	8-bit	RAM addresses 80–FFH only

statements and functions to Assembly language instructions. As a result, different compilers produce different code. In other words, if we compile a given 8051 C program with different compilers, each compiler produces different hex code.

For the above reasons, when we write time delays for C, we must use the oscilloscope to measure the exact duration. Look at Examples 6 through 8.

Example 6

Write an 8051 C program to toggle bits of P1 continuously forever with some delay.

Solution:

```
// Toggle P1 forever with some delay in between "on" and "off".
#include <reg51.h>
void main(void)
  {
    unsigned int x;
    for(;;)                      //repeat forever
      {
        P1=0x55;
        for(x=0;x<40000;x++);   //delay size unknown
        P1=0xAA;
        for(x=0;x<40000;x++);
    }
  }
```

Example 7

Write an 8051 C program to toggle bits of P1 ports continuously with a 250 ms delay.

Solution:

This program is tested for the DS89C4x0 with XTAL = 11.0592 MHz.

```
#include <reg51.h>
void MSDelay(unsigned int);
void main(void)
  {
    while(1)    //repeat forever
      {
        P1=0x55;
        MSDelay(250);
        P1=0xAA;
        MSDelay(250);
      }
  }
void MSDelay(unsigned int itime)
  {
    unsigned int i, j;
    for(i=0;i<itime;i++)
      for(j=0;j<1275;j++);
  }
```

Run the above program on your Trainer and use the oscilloscope to measure the delay.

Example 8

Write an 8051 C program to toggle all the bits of P0 and P2 continuously with a 250 ms delay.

Solution:

```
//This program is tested for the DS89C4x0 with XTAL = 11.0592 MHz
#include <reg51.h>
void MSDelay(unsigned int);
void main(void)
  {
    while(1)    //another way to do it forever
      {
        P0=0x55;
        P2=0x55;
        MSDelay(250);
        P0=0xAA;
        P2=0xAA;
        MSDelay(250);
      }
  }
void MSDelay(unsigned int itime)
  {
    unsigned int i, j;
    for(i=0;i<itime;i++)
      for(j=0;j<1275;j++);
  }
```

REVIEW QUESTIONS

1. Give the magnitude of the unsigned char and signed char data types.
2. Give the magnitude of the unsigned int and signed int data types.
3. If we are declaring a variable for a person's age, we should use the ___ data type.
4. True or false. Using a `for` loop to create a time delay is not recommended if you want your code be portable to other 8051 versions.
5. Give three factors that can affect the delay size.

2: I/O PROGRAMMING IN 8051 C

In this section, we look at C programming of the I/O ports for the 8051. We look at both byte and bit I/O programming.

Byte size I/O

Ports P0–P3 are byte-accessible. We use the P0–P3 labels as defined in the 8051/52 C header file. Examples 9–11 provide a better understanding of how ports are accessed in 8051 C.

Example 9

LEDs are connected to bits P1 and P2. Write an 8051 C program that shows the count from 0 to FFH (0000 0000 to 1111 1111 in binary) on the LEDs.

Solution:

```
#include <reg51.h>
#define LED P2          //notice how we can define P2
void main(void)
  {
    P1=00;              //clear P1
    LED=0;              //clear P2
    for(;;)             //repeat forever
      {
        P1++;           //increment P1
        LED++;          //increment P2
      }
  }
```

Example 10

Write an 8051 C program to get a byte of data from P1, wait 1/2 second, and then send it to P2.

Solution:

```
#include <reg51.h>
void MSDelay(unsigned int);
void main(void)
  {
    unsigned char mybyte;
    P1=0xFF;                //make P1 an input port
    while(1)
      {
        mybyte=P1;          //get a byte from P1
        MSDelay(500);
        P2=mybyte;          //send it to P2
      }
  }

void MSDelay(unsigned int itime)
  {
    unsigned int i, j;
    for(i=0;i<itime;i++)
      for(j=0;j<1275;j++);
  }
```

Example 11

Write an 8051 C program to get a byte of data from P0. If it is less than 100, send it to P1; otherwise, send it to P2.

Solution:

```
#include <reg51.h>
void main(void)
  {
    unsigned char mybyte;
    P0=0xFF;               //make P0 an input port
    while(1)
      {
        mybyte=P0;         //get a byte from P0
        if(mybyte<100)
          P1=mybyte;       //send it to P1 if less than 100
        else
          P2=mybyte;       //send it to P2 if more than 100
      }
  }
```

Bit-addressable I/O programming

The I/O ports of P0–P3 are bit-addressable. We can access a single bit without disturbing the rest of the port. We use the sbit data type to access a single bit of P0–P3. One way to do that is to use the `Px^y` format where *x* is the port 0, 1, 2, or 3, and *y* is the bit 0–7 of that port. For example, `P1^7` indicates P1.7. When using this method, you need to include the `reg51.h` file. Study Examples 12–15 to become familiar with the syntax.

Example 12

Write an 8051 C program to toggle only bit P2.4 continuously without disturbing the rest of the bits of P2.

Solution:

```
//toggling an individual bit
#include <reg51.h>
sbit mybit = P2^4;    //notice the way single bit is declared
void main(void)
{
  while(1)
    {
      mybit=1;        //turn on P2.4
      mybit=0;        //turn off P2.4
    }
  }
```

Example 13

Write an 8051 C program to monitor bit P1.5. If it is high, send 55H to P0; otherwise, send AAH to P2.

Solution:

```
#include <reg51.h>
sbit mybit = P1^5;    //notice the way single bit is declared
void main(void)
  {
    mybit=1;          //make mybit an input
    while(1)
      {
        if(mybit==1)
          P0=0x55;
        else
          P2=0xAA;
      }
  }
```

Example 14

A door sensor is connected to the P1.1 pin, and a buzzer is connected to P1.7. Write an 8051 C program to monitor the door sensor, and when it opens, sound the buzzer. You can sound the buzzer by sending a square wave of a few hundred Hz.

Solution:

```
#include <reg51.h>
void MSDelay(unsigned int);
sbit Dsensor = P1^1;  //notice the way single bit is defined
sbit Buzzer = P1^7;
void main(void)
  {
    Dsensor=1;        //make P1.1 an input
    while(Dsensor==1)
      {
        buzzer=0;
        MSDelay(200);
        buzzer=1;
        MSDelay(200);
      }
  }

void MSDelay(unsigned int itime)
  {
    unsigned int i, j;
    for(i=0;i<itime;i++)
      for(j=0;j<1275;j++);
  }
```