8051 MICROCONTROLLER: C PROGRAMMING

DR. VIBHU JATELY

ASSISTANT PROFESSOR-SG

SCHOOL OF COMPUTER SCIENCE

UPES

UNIVERSITY OF TOMORROW

WHY PROGRAM THE 8051 IN C

- It is easier and less time consuming to write in C as compared to assembly language.
- C is easier to modify and update.
- Code available in function libraries can be used.
- C code is portable to other microcontrollers with little or no modifications.

COMPILERS

- Compilers produce hex files that is downloaded to ROM of microcontroller
- The size of hex file is the main concern

✓ Microcontrollers have limited on-chip ROM

✓ Code space for 8051 is limited to 64K bytes

C programming is less time consuming, but has larger hex file size than assembly language.

DATA TYPES IN 8051 C

A good understanding of C data types for 8051 can help programmers to create smaller hex files. Following are the most useful and widely used data type for the 8051 microcontroller.

- ✓ Unsigned char
- ✓ Signed char
- ✓ Unsigned int
- ✓ Signed int
- ✓ Sbit (single bit)
- ✓ Bit and sfr

Unsigned Char

- It is an 8-bit data type that takes a value in the range of 0-255 (00 to FFH).
- It is one of the most widely used data type for the 8051.
- It is used for setting the counter value, ASCII characters.
- C compilers use the signed char as the default data type if we do not put the keyword unsigned in front of the char.
- In declaring variables, a careful attention has to be paid to the size of the data and try to use unsigned char instead of int if possible.

Signed Char

- Signed char is an 8-bit data type that uses the most significant bit (D7 of D7-D0) to represent the –ve or +ve value.
- Only 7 bits for the magnitude of the signed numbers, giving values from -128 to 127.
- Situations where + and – are needed to represent a given quantity such as temperature, the use of signed char data type is must.

Unsigned int

- Unsigned int is a 16-bit data type that takes a value in the range of 0 to 65535 (0000-FFFFH).
- Define 16-bit variables such as memory addresses
- Set counter values of more than 256
- We should not use the int data type unless we have to. Since registers and memory accesses are in 8-bit chunks, the misuse of int variables will result in a larger hex file.
- C compiler uses signed int as the default if we do not use the keyword unsigned.

Signed int

- Signed int is a 16-bit data type that uses the most significant bit (D15 of D15-D0) to represent the –ve or +ve value.
- As a result, only 15 bits are there for the magnitude of the number or values from -32,768 to 32,767.

Sbit (Single bit)

- Sbit is a widely used 8051 C data type designed specifically to access single-bit addressable registers.
- Some of the SFR are bit addressable, e. g. port P0-P3.
- Sbit can be used to access the individual bits of the ports.

Bit and sfr

- The bit data type allows access to single bits of bit-addressable memory spaces 20-2FH.
- SFR data type is used to access byte size SFR registers.

Some Widely Used Data Types for 8051 C

Data Type	Size in Bits	Data Range/Usage
Unsigned char	8-bit	0 to 255
Signed char	8-bit	-128 to 127
Unsigned int	16-bit	0 to 65535
Signed int	16-bit	-32,768 to 32,767
Sbit	1-bit	SFR bit-addressable
Bit	1-bit	RAM bit-addressable
Sfr	8-bit	RAM addresses 80-FFH

Time Delays

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

✓ Using a simple for loop

✓ Using the 8051 timers

In creating a time delay using a for loop, the following factors must be kept in mind as they can affect the accuracy of the delay:

- The 8051 design

✓ The number of machine cycle

✓ The number of clock periods per machine cycle

- The crystal frequency connected to the X1 – X2 input pins

- Compiler choice

✓ C compiler converts the C statements and functions to Assembly language instructions

✓ Different compilers produce different code

- Write an 8051 C program to send values (0-255)00-FFH to port P1.

```
 #include <reg51.h>
void main(void)
{
unsigned char z;
for (z=0;z<=255;z++)
    P1=z;
}
```

- Write an 8051 C program to display ASCII characters of 0,1,2,3,4,5,A,B,C, D to port P1.

```
#include <reg51.h>
void main(void)
{
unsigned char mynum []=“012345ABCD”;
unsigned char z;
for (z=0;z<10;z++)
  P1=mynum [z];
}
```

- Write an 8051 C program to toggle all the bits of P1 continuously.

```
#include <reg51.h>
void main(void)
 {
   for (; ;)                              // repeat forever
      {
           p1=0x55;
           p1=0xAA;
      }
}
```

- Write an 8051 C program to send values of -4 to 4 to port P1.

```
#include <reg51.h>
void main(void)
 {
    char mynum[]={+1,-1,+2,-2,+3,-3,+4,-4};
    unsigned char z;
    for (z=0;z<8;z++)
        P1=mynum[z];
}
```

- Write an 8051 C program to toggle bit D0 of the port P1 (P1.0) 50,000 times.

```
#include <reg51.h>
sbit MYBIT=P1^0;
void main(void)
 {
    unsigned int z;
    for (z=0;z<=50000;z++)
      {
          MYBIT=0;
          MYBIT=1;
       }
}
```

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

```
#include <reg51.h>
void main(void)
{
  unsigned int x;
  for (;;)                            //repeat forever
   {
     p1=0x55;
     for (x=0;x<40000;x++);     //delay size
     p1=0xAA;
     for (x=0;x<40000;x++);
   }
}
```

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

```
#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++);
}
```

(ASSUME CRYTSAL FREQUENCY OF 11.0592MHz)

(1 Machine Cycle require 12 crystal pulses)

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

```
# include>reg.51.h>
Void MSdelay (unsigned int);
Void man (void)
{
While(1)
          {
             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++)
    }
```

LOGIC OPERATORS IN 8051 C

- One of the most important and powerful feature of the C language is its ability to perform bit manipulation.
- There are several bit wise operators in C language, AND, OR, EX-OR, Inverter, Shift Right and Shift Left operators.
- The bit-wise operators are widely used in software engineering for embedded systems and control operations.

Bit-wise Logic Operators for C

		AND	OR	EX-OR	Inverter
A	B	A&B	A|B	A^B	Y=~B
0	0	0	0	0	1
0	1	0	1	1	0
1	0	0	1	1	
1	1	1	1	0	

- Run the following program on your simulator and examine the results.

```
#include <reg51.h>
void main(void)
  {
       P0=0x35 & 0x0F;                    //ANDing
       P1=0x04 | 0x68;                    //ORing
       P2=0x54 ^ 0x78;                    //XORing
       P0=~0x55;                          //inversing
       P2=0x77 >> 4;                      //shifting right 4
       P0=0x6 << 4;                       //shifting left 4
   }
```

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

```
#include <reg51.h>

void MSDelay(unsigned int);

void main(void)

 {

     P0=0x55;

     P2=0x55;

     while (1)

       {

          P0=~P0;

          P2=~P2;

          MSDelay(250);

        }

   }
```

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

- Write an 8051 C program to toggle all the bits of P0, P1, P2 continuously with a 250ms delay. Use the Ex-OR operator.

```
#include <reg51.h>

void MSDelay(unsigned int);

void main(void)

 {

    P0=0x55;

    P1=0x55;

    P2=0x55;

    while (1)

      {

         P0=P0^0xFF;

         P1=P1^0xFF;

         P2=P2^0xFF;

         MSDelay(250);

      }

  }
```

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

- Write an 8051 C program to get bit P1.0 and send it to P2.7 after inverting it.

```
#include <reg51.h>
sbit inbit=P1^0;
sbit outbit=P2^7;
bit membit;
void main(void)
 {
  while (1)
    {
        membit=inbit;           //get a bit from P1.0
        outbit=~membit;        //invert it and send it to P2.7
      }
   }
```