8051 MICROCONTROLLER: C PROGRAMMING

WHY PROGRAM THE 8051 IN C

- It is easier and less time consuming to write in C as compare 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 send hex values for 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++);
}
```

- 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
      }
  }
```

- Write an 8051 C program to read the P1.0 and P1.1 bits and issue an ASCII character to P0 according to the following table.

P1.1	P1.0	Decision
0	0	Send '0' to P0
0	1	Send '1' to P0
1	0	Send '2' to P0
1	1	Send '3' to P0

```
#include <reg51.h>

void main(void)

 {

      unsigned char z;

      z=P1;

      z=z&0x3;
```

```
switch (z)
    {
        case(0):
         {
            P0='0';
            break;
         }
        case(1):
         {
            P0='1';
            break;
          }
         case(2):
          {
            P0='2';
            break;
           }
```

```
        case(3):
          {
             P0='3';
              break;
           }
       }
   }
```

INPUT/OUTPUT PROGRAMMING

- Write an 8051 C program to count from 00 to FFH on the LEDs connected to port P1.

```
#include <reg51.h>
void main(void)
 {
    P1=00;                    //clear P1
    for (;;)                  //repeat forever
   {
      P1++;                   //increment P1
   }
 }
```

- Write an 8051 C program to get a byte of data from P1, wait 0.5 second, and then send it to P2.

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

- 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.

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

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

```
#include <reg51.h>
sbit mybit=P2^4;
void main(void)
 {
    while (1)
     {
        mybit=1;              //turn on P2.4
        mybit=0;             //turn off P2.4
      }
 }
```

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

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

- 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.

```
#include <reg51.h>

void MSDelay(unsigned int);

sbit Dsensor=P1^1;

sbit Buzzer=P1^7;
```

```
void main(void)
 {
    Dsensor=1;                          //make P1.1 an input
    while (1)
      {
        while (Dsensor==1)//while it opens
           {
               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++);
   }
```

- Write an 8051 C program to get the status of bit P1.0, save it and send it to P2.7 Continuously.

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

ACCESSING SFR ADDRESSES 80-FFH

- Another way to access the SFR RAM space 80-FFH is to use the sfr data type.

P0	Addr	P1	Addr	P2	Addr	P3	Addr	Port's Bit
P0.0	80	P1.0	90	P2.0	A0	P3.0	B0	D0
P0.1	81	P1.1	91	P2.1	A1	P3.1	B1	D1
P0.2	82	P1.2	92	P2.2	A2	P3.2	B2	D2
P0.3	83	P1.3	93	P2.3	A3	P3.3	B3	D3
P0.4	84	P1.4	94	P2.4	A4	P3.4	B4	D4
P0.5	85	P1.5	95	P2.5	A5	P3.5	B5	D5
P0.6	86	P1.6	96	P2.6	A6	P3.6	B6	D6
P0.7	87	P1.7	97	P2.7	A7	P3.7	B7	D7

Bit Addresses for all Ports

- We can also access a single bit of any SFR if we specify the bit address.
- There is no need to use #include<reg51.h> statement.
- This allows us to access any byte of the SFR RAM space 80-FFH.
- This is the method widely used for the new generation of 8051 microcontroller.

- Write an 8051 C program to toggle all the bits of P0, P1, P2 continuously with a 250ms delay. Use the sfr keyword to declare the port addresses.

```
sfr P0=0x80;
sfr P1=0x90;
sfr P2=0xA0;
 void MSDelay(unsigned int);
 void main(void)
   {
       while (1)
         {
            P0=0x55;
            P1=0x55;
            P2=0x55;
           MSDelay(250);
           P0=0xAA;
           P1=0xAA;
           P2=0xAA;
           MSDelay(250);
          }
 }
```

- Write an 8051 C program to turn bit P1.5 on and off 50,000 times.

```
sbit MYBIT=0x95;
void main(void)
  {
      unsigned int z;
      for (z=0;z<50000;z++)
        {
          MYBIT=1;
          MYBIT=0;
         }
   }
```

DATA SERIALIZATION USING 8051 C

- Serializing data is a way of sending a byte of data one bit at a time through a single pin of microcontroller. There are two ways to transfer the byte of data serially:

✓ Using the serial port. When using the serial port, the programmer has very limited control over the sequence of data transfer.

✓ The second method of serializing data is to transfer data one bit at a time and control the sequence of data and spaces in between them.

- Write a C program to send out the value 44H serially one bit at a time via P1.0. The LSB should go out first.

```
#include <reg51.h>
sbit P1b0=P1^0;
sbit regALSB=ACC^0;
void main(void)
 {
      unsigned char conbyte=0x44;
      unsigned char x;
     ACC=conbyte;
     for (x=0;x<8;x++)
      {
        P1b0=regALSB;
        ACC=ACC>>1;
      }
 }
```

SHIFTING RIGHT

- Write a C program to send out the value 44H serially one bit at a time via P1.0. The MSB should go out first.

```
#include <reg51.h>
sbit P1b0=P1^0;
sbit regAMSB=ACC^7;
void main(void)
  {
     unsigned char conbyte=0x44;
     unsigned char x;
    ACC=conbyte;
    for (x=0;x<8;x++)
     {
       P1b0=regAMSB;
      ACC=ACC<<1;
     }
  }
```

SHIFTING LEFT

Write a C program to bring in a byte of data serially one bit at a time via P1.0. The LSB should come in first.

```
#include <reg51.h>
sbit P1b0=P1^0;
sbit ACCMSB=ACC^7;
bit membit;
void main(void)
{
   unsigned char x;
   for (x=0;x<8;x++ )
    {
        membit=P1b0;
        ACCMSB=membit;
        ACC=ACC>>1;

    }
}
```

SHIFTING RIGHT

- Write a C program to bring in a byte of data serially one bit at a time via P1.0. The MSB should come in first.

```
#include <reg51.h>
sbit P1b0=P1^0;
sbit regALSB=ACC^0;
bit membit;
void main(void)
 {
    unsigned char x;
    for (x=0;x<8;x++)
     {
        membit=P1b0;
        regALSB=membit;
        ACC=ACC<<1;

     }
 }
```

SHIFTING LEFT

DATA CONVERSION PROGRAMS IN C

- ASCII numbers:

Key	ASCII (hex)	Binary	BCD
0	30	0110000	0000 0000
1	31	0110001	0000 0001
2	32	0110010	0000 0010
3	33	0110011	0000 0011
4	34	0110100	0000 1000
5	35	0110101	0000 1001
6	36	0110110	0000 1100
7	37	0110111	0000 0111
8	38	0111000	0000 1000
9	39	0111001	0000 1001

- Write an 8051 C program to convert packed BCD 0x29 to ASCII and display the bytes on P1 and P2.

```
#include <reg51.h>
void main(void)
{
    unsigned char x,y,z;
    unsigned char mybyte=0x29;
    x=mybyte&0x0F;
    P1=x|0x30;
    y=mybyte&0xF0;
    y=y>>4;
   P2=y|0x30;
 }
```

- Write an 8051 C program to convert ASCII digits of ‘4’ and ‘7’ to packed BCD and display them on P1.

```
#include <reg51.h>
void main(void)
{
   unsigned char bcdbyte;
   unsigned char w='4';
   unsigned char z='7';
   w=w&0x0F;
  w=w<<4;
  z=z&0x0F;
  bcdbyte=w|z;
  P1=bcdbyte;
 }
```