Giới thiệu về C với 8051

TS Nguyễn Hồng Quang

Chương trình C đầu tiên

```
#include <reg51.h>
void main(void)
{
   for (;;)
   {
        P1=0x55;
        P1=0xAA;
   }
}
```

```
#include <reg51.h>
void main(void)
{
   while(1)
   {
        P1=0x55;
        P1=0xAA;
   }
}
```

Kiểu dữ liệu thường dùng

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	-32768 to +32767
sbit	1-bit	SFR bit-addressable only
bit	1-bit	RAM bit-addressable only
sfr	8-bit	RAM addresses 80 – FFH only

Ví dụ với sbit

```
#include <reg51.h>
sbit mybit=P2^4;
void main(void)
{
    while (1)
    {
        mybit=1; //turn on P2.4
        Delay(1000);
        mybit=0; //turn off P2.4
    }
}
```

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

Loại bộ nhớ

Memory Type	Description
code	Program memory (64 KBytes); accessed by opcode MOVC @A+DPTR.
data	Directly addressable internal data memory; fastest access to variables (128 bytes).
idata	Indirectly addressable internal data memory; accessed across the full internal address space (256 bytes).
bdata	Bit-addressable internal data memory; supports mixed bit and byte access (16 bytes).
xdata	External data memory (64 KBytes); accessed by opcode MOVX @DPTR.
far	Extended RAM and ROM memory spaces (up to 16MB); accessed by user defined routines or specific chip extensions (Philips 80C51MX, Dallas 390).
pdata	Paged (256 bytes) external data memory; accessed by opcode MOVX @Rn.

Ví dụ

- char **data** var1;
- char **code** text[] = "ENTER PARAMETER:";
- unsigned long **xdata** array[100];
- float **idata** x,y,z;
- unsigned int **pdata** dimension;
- unsigned char **xdata** vector[10][4][4];
- char **bdata** flags;
- data char *x; // Old-Style Memory Type Declaration
  char *data x; // New-Style Memory Type Declaration

Phép toán logic

- Toán hạng logic
  - AND (&&), OR (||), and NOT (!)
- Toán hạng theo bit
  - AND (&), OR (|), EX-OR (^), Inverter (~),
  - Shift Right (>>), and Shift Left (<<)

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

Ví dụ lệnh sử dụng bit

```
#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;
             }
         }
}
```

Chuyển đổi Hex - ASCII

```
#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;
}
```

```
#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;
}
```

Mặt nạ bit

- Bước 1. Tạo ra số nguyên để đại diện cho từng trạng thái của bit (hoặt nhóm bit). Ví dụ

```
enum {
  FIRST = 0x01, /* 0001 binary */
  SECND = 0x02, /* 0010 binary */
  THIRD = 0x04, /* 0100 binary */
  FORTH = 0x08, /* 1000 binary */
  ALL = 0x0f    /* 1111 binary */
};
```

Mặt nạ bit

- Một cách khác

```
enum {
        FIRST = 1 << 0,
        SECND = 1 << 1,
        THIRD = 1 << 2,
        FORTH = 1 << 3,
        ALL = ~(~0 << 4)
};
```

- Dòng cuối cùng thường dùng để bật tắt một nhóm bit

```
1111 1111 /* ~0 */
1111 0000 /* ~0 << 4 */
0000 1111 /* ~(~0 << 4) */
```

Thao tác với mặt nạ bit

```
unsigned flags = 0;

flags |= SECND | THIRD | FORTH;      /* (1110). */
flags &= ~(FIRST | THIRD);          /* (1010). */
flags ^= (THIRD | FORTH);           /* (1100). */

if ((flags & (FIRST | FORTH)) == 0)
    flags &= ~ALL;                  /* (0000). */
```

- Keys:
  1. **Toán tử|** dùng để tổ hợp các mặt nạ, toán tử **~** dùng để đảo dấu tất cả các bit (mọi bit là 1 *trừ những bit được che mặt nạ*).
  2. **|=** dùng để set bits.
  3. **&=** dùn để reset bits.
  4. **^=** dùng để đảo dấu bits.
  5. **&** dùng để chọn bits (cho việc kiểm tra trạng thái).

Các Macros cho từng bít

```
#define BitSet(arg,posn) ((arg) |  (1L << (posn)))
#define BitClr(arg,posn) ((arg) & ~(1L << (posn)))
#define BitFlp(arg,posn) ((arg) ^  (1L << (posn)))
#define BitTst(arg,posn) ((arg) &  (1L << (posn)))


enum {FIRST, SECND, THIRD};
unsigned flags = 0;

flags = BitSet(flags, FIRST);  /* Set first bit. */
flags = BitFlp(flags, THIRD);  /* Toggle third bit.
*/
if (BitTst(flags, SECND) == 0) /* Test second bit.
*/
     flags = 0;
```

Ví dụ về check sum 8bit

Ví dụ ta có 4 số 25H, 62H, 3FH, 52H.
(a) Tìm checksum byte.

25H
+ 62H
+ 3FH The checksum byte là 2's
+ 52H của 18H, giá trị là E8H
118H

(b) Kiểm tra giá checksum byte

25H
+ 62H
+ 3FH
+ 52H
+ E8H
200H (dropping the carries)

(c) Nếu số 62H bị thay đổi thành 22H,

25H
+ 22H
+ 3FH
+ 52H
+ E8H
1C0H

Ví dụ về checksum 8bit

```
#include <reg51.h>
void main(void)
{
    unsigned char
      mydata[]={0x25,0x62,0x3F,0x52};
    unsigned char sum=0, x
    unsigned char chksumbyte;
    for (x=0;x<4;x++)
    {
        P2=mydata[x];
        sum=sum+mydata[x];
        P1=sum;
    }
    chksumbyte=~sum+1;
    P1=chksumbyte;
}
```

```
#include <reg51.h>
void main(void)
{
    unsigned char mydata[]
    ={0x25,0x62,0x3F,0x52,0xE8};
    unsigned char shksum=0;
    unsigned char x;
    for (x=0;x<5;x++)
        chksum=chksum+mydata[x];
    if (chksum==0)
        P0='G';
    else
        P0='B';
}
```


Ví dụ về Timer 0 – mode 1:16bit

```
#include <reg51.h>
sbit mybit=P1^5;
sbit SW=P1^7;
void T0M1Delay(unsigned char);
void main(void){
SW=1;
while (1) {
    mybit=~mybit;
    if (SW==0)
        T0M1Delay(0);
    else
        T0M1Delay(1);
    }
}
```

```
void T0M1Delay(unsigned char c){
    TMOD=0x01;
    if (c==0) {
        TL0=0x67;
        TH0=0xFC; }
    else {
        TL0=0x9A;
        TH0=0xFD;
    }
    TR0=1;
    while (TF0==0);
    TR0=0;
    TF0=0;
}
```

FC67H = 64615
65536 – 64615 = 921
921 × 1.085 µs = 999.285 µs
1 / (999.285 µs × 2) = 500 Hz

Ví dụ Timer 0, mode 2-8bit

```
#include <reg51.h>
void T0M2Delay(void);
sbit mybit=P1^5;
void main(void){
    unsigned char x,y;
    while (1) {
    mybit=~mybit;
    for (x=0;x<250;x++)
          for (y=0;y<36;y++)
          T0M2Delay();
    }
}
```

```
void T0M2Delay(void){
TMOD=0x02;
TH0=-23;
TR0=1;
while (TF0==0);
TR0=0;
}
```

256 – 23 = 233
23 × 1.085 µs = 25 µs and
25 µs × 250 × 40 = 250 ms

Vào ra cổng nối tiếp

```
#include <reg51.h>
void SerTx(unsigned char);
void main(void){
      TMOD=0x20; //use Timer 1, mode 2
      TH1=0xFD; //9600 baud rate
      SCON=0x50;
      TR1=1; //start timer
      while (1) {
            SerTx('Y');
            SerTx('E');
            SerTx('S');
      }
}
void SerTx(unsigned char x){
      SBUF=x; //place value in buffer
      while (TI==0); //wait until transmitted
      }
```

Nhận dữ liệu cổng nối tiếp

```
#include <reg51.h>
void main(void){
unsigned char mybyte;
      TMOD=0x20; //use Timer 1, mode 2
      TH1=0xFA; //4800 baud rate
      SCON=0x50;
      TR1=1; //start timer
      while (1) { //repeat forever
         while (RI==0); //wait to receive
         mybyte=SBUF; //save value
         P1=mybyte; //write value to port
         RI=0;
         }
}
```

Thay đổi tốc độ cho cổng nối tiếp

```
#include <reg51.h>
sbit MYSW=P2^0; //input switch
void main(void){
unsigned char z;
unsigned char Mess1[]="Normal
    Speed";
unsigned char Mess2[]="High Speed";
TMOD=0x20; //use Timer 1, mode 2
TH1=0xFF; //28800 for normal
SCON=0x50;
TR1=1; //start timer
      if(MYSW==0) {
      for (z=0;z<12;z++) {
            SBUF=Mess1[z]; //place
               value in buffer
            while(TI==0); //wait for
               transmit
            TI=0;
            }
      }
else {
   PCON=PCON|0x80; //for high
   speed of 56K
   for (z=0;z<10;z++) {
      SBUF=Mess2[z]; //place
   value in buffer
   while(TI==0); //wait for
   transmit
   TI=0;
   }
}
```

Bảng vector ngắt

Interrupt Number	Description	Address
0	EXTERNAL INT 0	0003h
1	TIMER/COUNTER 0	000Bh
2	EXTERNAL INT 1	0013h
3	TIMER/COUNTER 1	001Bh
4	SERIAL PORT	0023h

Interrupt Number	Address
0	0003h
1	000Bh
2	0013h
3	001Bh
4	0023h
5	002Bh
6	0033h
7	003Bh
8	0043h
9	004Bh
10	0053h
11	005Bh
12	0063h
13	006Bh
14	0073h
15	007Bh

Interrupt Number	Address
16	0083h
17	008Bh
18	0093h
19	009Bh
20	00A3h
21	00ABh
22	00B3h
23	00BBh
24	00C3h
25	00CBh
26	00D3h
27	00DBh
28	00E3h
29	00EBh
30	00F3h
31	00FBh

Ví dụ sử dụng với ngắt Timer

```
#include <reg51.h>
sbit SW =P1^7;
sbit IND =P1^0;
sbit WAVE =P2^5;
void timer0(void) interrupt 1 {
WAVE=~WAVE; //toggle pin
}
void main() {
      SW=1; //make switch input
      TMOD=0x02;
      TH0=0xA4; //TH0=-92
      IE=0x82; //enable interrupt for timer 0
      while (1) {
         IND=SW; //send switch to LED
      }
}
```

Ví dụ tổng hợp về ngắt

```
#include <reg51.h>
sbit WAVE =P0^1;
void timer0() interrupt 1 {
    WAVE=~WAVE; //toggle pin
}
void serial0() interrupt 4 {
    if (TI==1) {
      TI=0; //clear interrupt
    }
    else {
    P0=SBUF; //put value on pins
    RI=0; //clear interrupt
    }
}
```

```
void main() {
      unsigned char x;
      P1=0xFF; //make P1 an input
      TMOD=0x22;
      TH1=0xF6; //4800 baud rate
      SCON=0x50;
      TH0=0xA4; //5 kHz has =200us
      IE=0x92; //enable interrupts
      TR1=1; //start timer 1
      TR0=1; //start timer 0
      while (1) {
      x=P1; //read value from pins
      SBUF=x; //put value in buffer
      P2=x; //write value to pins
      }
}
```