SEL0337

8051 Programação em C

Prof. Dr. Marcelo A. C. Vieira

Linguagem C

- Linguagem que pode ser utilizada atualmente na programação de quase todos os microcontroladores;
- Há microcontroladores com instruções otimizadas para programação em C;
- O compilador transforma as instruções em C no código em assembly;
- Há a possibilidade de colocar instruções em assembly no código em C;
- Geralmente o programa fica maior e menos eficiente em linguagem C, mas a facilidade na programação é muito maior;

C e Assembly

- loop em Assembly

```
      MOV R1,#255
ABC:  MOV P1,R1
      DJNZ R1,ABC
```

for-loop em C

```
int z;
for (z=255; z>0; z--)
  P1=z;
```

Estrutura de um programa C

- Diretivas de pré-processamento
- Declaração de variáveis globais
- Declaração de protótipos de funções
- Definições das funções
- Programa principal

Exemplo de um programa C

```
#include <at89x52.h>   //diretiva de pré-processamento

int x,y;    //declaração de variáveis globais

int soma(int, int);     //declaração de protótipo de função

void main()     //Programa principal
{
    x = 10;
    y = 20;
   x = soma(x,y);
}

int soma(int a, int b)   // definição de função
{
    return a+b;
}
```

Linguagem C

- O programa deve ser salvo na extensão **`*.c`**
- A primeira função a ser executada é:

```
void main(void)
{
}
```

ela não recebe e não retorna nenhum parâmetro

- Comentários:

```
// comentário em uma linha
/* comentário em mais de
uma linha */
```

Linguagem C – SDCC 8051

- Há uma biblioteca para cada microcontrolador que especifica os registradores especiais (SFR)

  - Usar o nome do SFR com o at89x52.h

```
#include <at89x52.h>
void main(void) {
P1=0x55;   // MOV P1,#55h
}
```

Linguagem C – SDCC 8051

- Para carregar um valor em um registrador, basta colocar o nome dele:

```
P1 = 0;
TMR0L = 0x0F;
P2 = 255;
```

Linguagem C

- Representação numérica:

```
P1 = 255; // decimal
P1 = 0xff; // hexadecimal
P1 = 0377; // octal
P1 = 0b11111111; // binário
```

Tipos de Dados (Data Types)

Tipos de Dados (SDCC) para o 8051

- unsigned char, signed char
- unsigned int, signed int,
- Single bit: sbit (SFR) , bit (bit-addressable RAM)
- Special function register: sfr

- O compilador C alocará um espaço na RAM para as variáveis (char, int & bit).

Data Type	Size in Bits	Data Range/Usage
unsiged char	8-bit	0 to 255
(signed) char	8-bit	–128 to +127
unsigned int	16-bit	0 to 65535
(signd) int	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

Operadores em C

Linguagem C

- Operadores aritméticos:

Operador	Ação
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão inteira
++	Incremento
- -	Decremento

Linguagem C

- Operadores bit a bit:

Operador	Ação
&	AND
|	OR
^	XOR
~	Complemento
>>	Deslocamento à direita
<<	Deslocamento à esquerda

Linguagem C

- Operadores relacionais:

Operador	Ação
==	Igual a
>	Maior que
<	Menor que
!=	Diferente de
>=	Maior ou igual a
<=	Menor ou igual a

Linguagem C

- Operadores relacionais booleanos:

Operador	Ação
&&	E
||	OU
!	NOT

Exemplos de Rotinas em C do 8051

Unsigned Char

- Tipo de dado mais usado para o 8051
- 8-bit data type
- Range do unsigned char: 0-255 (00-FFH)
- Quando usar unsigned char?
  - Para estabelecer valor de um contador (Exemplo_1)
  - Para “string” de caracteres ASCII (Exemplo_2)
  - Para alterar valor das portas (Exemplo_3)

EXEMPLO_1 (UNSIGNED CHAR)

Programa em C do 8051 para enviar valores de 00 a FF para a porta P1.

```
#include <at89x52.h>
void main(void)
{
  unsigned char z;
  for (z=0; z<=255; z++)
    P1=z;
}
```

EXEMPLO_2 (UNSIGNED CHAR)

Programa em C do 8051 para enviar valores hexadecimais dos caracteres ASCII → 0,1,2,3,4,5,A,B,C,D para a porta P1.

```
#include <at89x52.h>
void main(void){
   unsigned char mynum[]="012345ABCD";
   unsigned char z;
   for (z=0; z<10; z++)
      P1=mynum[z];
}
```

Nota: A "string" "**012345ABCD**" é armazenada na RAM interna e pode ser alterada.

EXEMPLO_3 (UNSIGNED CHAR)

Programa em C do 8051 para alternar (toggle) todos os bits da porta P1 continuamente.

```
#include <at89x52.h>
void main(void) {
unsigned char S1=0x55;
unsigned char S2=0xAA;
 for (z=0; z<=255; z++){
    P1=S1;
    P1=S2;
  }
}
```

Signed Char

- 8-bit data type
- Representação em complemento de 2
- Range do signed char: -128 ---127 (00-FFH)
- Quando usar o signed char?
  - Para apresentar uma determinada quantidade numérica, por exemplo a temperatura de algum objeto (Exemplo_4)

EXEMPLO_4 (SIGNED CHAR)

Programa em C do 8051 para enviar valores de -4 a 4 para a porta P1.

```
#include <at89x52.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];
}
```

Integer

- 16-bit data type
  - Range do unsigned int: 0 - 65535
  - Range do signed int: -32768 - 32767
- Como a memória RAM do 8051 é de 8-bits e o tipo de dado int necessita de 2 bytes da RAM, só se deve usar o tipo int se for realmente necessário.
- Deve-se usar o tipo unsigned char ao invés de int.

EXEMPLO_5 (UNSIGNED INT, SBIT)

Programa em C do 8051 para comutar (toggle) 50.000 vezes o bit 0 da porta P1.

```
#include <at89x52.h>
__sbit      MYBIT;
  void main(void) {
       unsigned int z;
       for (z=0;z<50000;z++) {
       MYBIT=0;
       P1_0=MYBIT;
       MYBIT=1;
       P1_0=MYBIT;
       }
   }
```

Exemplos de Rotinas de Atrasos (Time Delay) em C do 8051

Time Delay (Rotinas de Atraso) em C

- Tres fatores afetam a precisão de rotinas de Atraso em C:
  - A frequência do oscilador
  - O ciclo de máquina do 8051
  - O compilador C usado

Exemplo_6 e Exemplo_7

EXEMPLO_6 (TIME DELAY)

Programa em C do 8051 que comuta (toggle) todos os bits da porta P1 continumente em loop infinito com atraso(Delay).

```
#include <at89x52.h>
  void main(void)
  {
       unsigned char x;
       while (1) {
              P1=0x55;
              for (x=0;x<40;x++); //Atraso ???
              P1=0xAA;
              for (x=0;x<40;x++); //Atraso ???
                  }
  }
```

EXEMPLO_7 (1/2)

Programa em C do 8051 que comuta (toggle) todos os bits da porta P1 continuamente com atraso de 250 ms.

```
#include <at89x52.h>
  void MSDelay(unsigned int);
  void main(void)
{
      while(1)    {         //repete sempre
      P1=0x55;
            MSDelay(250);   //Atraso 250 ms
      P1=0xAA;
            MSDelay(250);
                  }
}
```

EXEMPLO_7 (2/2)

Conhecendo-se o clock do 8051, deve-se calcular (ou medir via simulador) o valor dado a j para que se tenha uma rotina de 1ms de atraso.

```
void MSDelay(unsigned int itime) {
     unsigned int i,j;
     for (i=0; i<itime; i++)
     for (j=0; j<1275; j++); //1ms delay
  }
```

Programação das Portas de I/O em C do 8051

Acesso aos SFR's

- Usar o nome do SFR com o at89x52.h

```
#include <at89x52.h>
void main(void) {
P1=0x55;   // MOV P1,#55h
}
```

- Usar o tipo de dado sfr e declarar no programa

```
void main(void) {
  __sfr   __at  0x90  P1;       // endereço da P1
          P1 = 0x55;            // P1 = 55h
  }
```

EXEMPLO_8 (SFR DECLARADO NO PROGRAMA)

Programa em C do 8051 que troca (toggle) todos os bits de P0, P1 e P2 continuamente com atraso de 250 ms

```
__sfr __at 0x80 P0;   // declarado no programa
__sfr __at 0x90 P1;   // declarado no programa
__sfr __at 0xA0 P2;   // declarado no programa
void MSDelay(unsigned int itime) {
        unsigned int i,j;
        for (i=0; i<itime; i++)
        for (j=0; j<1275; j++); //1ms delay
        }
void main(void) {
  while(1) {          //repete sempre
    P0 = 0x55;
    P1 = 0x55;
    P2 = 0x55;
 MSDelay(250);  //atraso de 250 ms
  P0=0xAA;
  P1=0xAA;
  P2=0xAA;
 MSDelay(250);  //atraso de 250 ms
}
}
```

SFR endereçáveis à BIT

- Para carregar um valor em um bit, deve-se colocar o nome do registrador e o endereço do bit (0, 1, 2...):

```
P1_5 = 1;
P2_1 = 0;
```

- Ou, pode apenas colocar o nome do bit, se conhecido:

```
EA = 1;
IT0 = 0;
EX0 = 0;
```

Acesso a um Bit do SFR

- Usar o tipo sbit e o nome do SFR com at89x52.h

```
#include <at89x52.h>
        void main(void) {
        __sbit MYBIT = P1_5;
        MYBIT = 1;
        }
```

- Usar o tipo sbit para declarar o bit do SFR diretamente no programa

```
    void main(void) {
        __sbit  __at  0x95 P1_5; // Endereço de P1.5
        P1_5 = 1;
        }
```

EXEMPLO_9 (SBIT DECLARADO NO PROGRAMA)

Programa em C (8051) que liga e desliga o bit P1.5 50.000 vezes.

```
__sbit __at 0x95 MYBIT;   // endereço de P1.5
void main(void) {
  unsigned int z;
  for (z=0;z<50000;z++) {
      MYBIT=1;
    MYBIT=0;
}
}
```

Acesso a Bit da RAM interna (área endereçável a bit)

- O tipo de dado bit é usado para acessar um bit da RAM interna endereçável a bit de 20H-2FH.

Programa em C (8051) que lê o bit da Porta P1.0, salva na RAM interna e envia para a Porta P2.7

```
#include <at89x52.h>
__sbit inbit;
__sbit outbit;
__bit membit;
void main(void) {

  while(1) {          //repete sempre
          inbit=P1_0;
          P2_7=outbit;
    membit = inbit;
    outbit = membit;
         }
         }
```

Assembly do 8051 “inline” (dentro de um Programa em C)

Assembly "inline"

- Restrições no uso dos Labels:
  - Os "Labels"devem ter a forma

nnnnn$:

onde nnnnn é um número menor que 100

Cada conjunto de códigos em Assembly deve estar entre as palavras-chaves __asm e __endasm;

Exemplo:

```
__asm
        mov b,#0xff
00001$:
        djnz b,00001$
__endasm ;
```

Rotina de Delay em Assembly

Exemplo: Programa que alterna todos os bits da Porta P0 com atraso de aproximadamente 1 ms

```
#include <at89x52.h>
void delay(void);
void main(void)
{
        while(1)
                {
                P0=0;
                delay();
                P0=0xFF;
                delay();
                }
 }
void delay(void) {
        __asm
        mov     r1, #0x3
00001$: mov     r0, #0xB2
00002$: djnz    r0, 00002$
        djnz    r1, 00001$
        __endasm;
 }
```

OPERACÕES LÓGICAS EM C DO 8051

Operações Lógicas Bit-a bit em C

AND &

```
0x35 & 0x0F = 0x05
```

```
  0011 0101
AND 0000 1111
----------
  0000 0101
```

OR |

```
0x04 | 0x68 = 0x6C
```

```
  0000 0100
OR  0110 1000
----------
  0110 1100
```

Exclusive-OR ^

```
0x54 ^ 0x78 = 0x2C
```

```
  0101 0100
XOR 0111 1000
----------
  0010 1100
```

Complemento ~

```
~0x55 = 0xAA
```

```
NOT 0101 0101
-------------
    1010 1010
```

EXEMPLO_10: OPERAÇÕES LÓGICAS BIT-A BIT

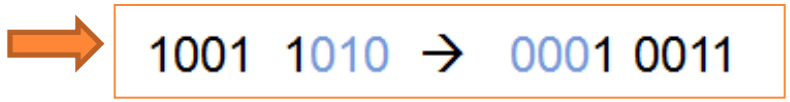
Programa para realizar operações Lógicas AND, OR, Exculsive OR , NOT mostrando o resultado nas Portas de I/O

```
#include <at89x52.h>
void main(void) {
  P0=0x35 & 0x0F; // 35h AND 0Fh = 05h
  P1=0x04 | 0x68; // 04h OR 68h = 6Ch
  P2=0x54 ^ 0x78; // 54h XOR 78h = 2Ch
  P3=~0x55;       // NOT 55h = AAh
 }
```

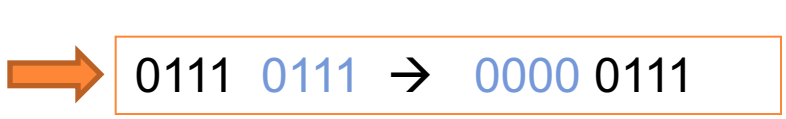
Operacões de Deslocamento de bit (Shift) em C

Shift Right >> (Deslocamento à Direita)

`0x9A >> 3 = 0x13`
desloca à direita 3 bits

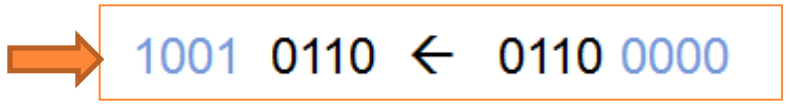


`0x77 >> 4 = 0x07`
desloca à direita 4 bits



Shift Left << (Deslocamento à Esquerda)

`0x96 << 4 = 0x60`
desloca à esquerda 4 bits



Exemplo_11 Operacões de Deslocamento de bit

Programa para realizar operações de Deslocamento à direita e à esquerda mostrando o resultado nas Portas de I/O

```
#include <at89x52.h>
void main(void) {
  P0=0x9A >> 3;  // shift right de 3 Bits
  P1=0x77 >> 4;  // shift right de 4 Bits
  P2=0x96 << 4; // shift left de 4 Bits

 }
```

Uso de Interrupções em C do 8051

INTERRUPÇÕES DO 8051

Para usar as Interrupções do 8051 em C, deve-se gerar as Sub-rotinas de Atendimento de Interrupção de acordo com:

```
void subint (void) __interrupt (1) __using (0)
{
...
}
```

O número entre parênteses significa o Banco de Registradores usado na Interrupção.

O número entre parênteses significa a Interrupção usada.

Interrupt #	Description	Vector Address
0	External 0	0x0003
1	Timer 0	0x000b
2	External 1	0x0013
3	Timer 1	0x001b
4	Serial	0x0023
5	Timer 2 (8052)	0x002b
...		...
n		0x0003 + 8*n

Interrupções do 8051

```
void timer_isr (void) __interrupt (1) __using (2)
{
...
}
```

Se a palavra-chave __using(n) for omitida, o compilador usará como default o Banco 0.

Cuidado ao utilizar o Banco 1, pois ele é normalmente reservado para uso da Pilha.

Interrupções do 8051

Se existirem múltiplas fontes de Interrupção, as Sub-rotinas de Atendimento podem estar presentes em qualquer ordem.

Um protótipo da Sub-rotina de Atendimento deve estar presente.

Habilitar Interrupções

- As Interrupções devem ser Habilitadas diretamente no Programa Principal.
- EA = 1
- EX0 = 1
- Observar o arquivo at89x52.h para uso correto dos nomes dos bits de programação das Interrupções.

Desabilitar Interrupções

Desabilitar diretamente no programa : EA = 0 , EX0 = 0 ..... Etc.

A palavra-chave __critical pode ser associada a um bloco ou a uma função declarada como "critica".

O SDCC gerará código para desabilitar todas as Interrupções ao entrar em uma "função crítica" e restaurará as habilitações das Interrupções ao retornar.

```
int foo () __critical
{
...
...
}
```

Uma Função critica: Interrupções não serão atendidas .


```
__critical{ i++; }
```

Um comando ou um bloco critico.

Comunicação Serial em C do 8051

Transmissão Serial

Exemplo: Transmitir o caractere ASCII da letra "A" para a saída serial em 9600,N,8,1

```
#include <at89x52.h>
void putchar(char c);
void init_serial(void);
unsigned char c;
void main(void){
init_serial();
c='A';
putchar(c);
}
void init_serial(void){
SCON=0x50;
TMOD=0x20,
TH1=253;
TL1=253;
TR1=1;
}
void putchar (char c) {
while (!TI)
SBUF = c;
TI = 0;
}
```

Comunicação Serial

Exemplo: Programa que espera um caractere pelo canal Serial e o envia pelo canal Serial na taxa de 9600,N,8,1

```
#include <at89x52.h>
void putchar(char c);
char getchar (void);
void init_serial(void);
unsigned char  d;
        void main(void){
        init_serial();
        d=getchar();
        putchar(d);
        }
                void init_serial(void){
                SCON=0x50;
                TMOD=0x20,
                TH1=253; // Serial programada para
                TL1=253; // 9600,N,8,1
                TR1=1;
                }
        void putchar (char c) { // Envia um caracter para o SBUF
        while (!TI)
        SBUF = c;
        TI = 0;
        }
                char getchar (void){ // Lê um caracter do SBUF
                char d;
                while(!RI);
                d=SBUF;
                RI=0;
                return d;
                }
```

FIM