第7章 STC单片机汇编语言编程模型

何宾

2018.03

本章主要内容

- 汇编语言程序结构
- 汇编代码中段的分配
- 汇编语言符号及规则
- 汇编语言操作数描述
- 汇编语言控制描述
- 单片机端口控制汇编语言程序设计
- 单片机中断汇编语言程序设计

汇编语言程序结构

实际上，所谓的汇编语言程序就是按照一定的规则组合在一起的机器语言助记符和汇编器助记符命令。

- **这些按一定规则组合在一起的汇编语言助记符机器指令，能通过软件开发环境的处理，转换成可以在STC 8051 CPU上按照设计要求运行的机器代码。**

汇编语言程序结构

```
NAME main                          ; 声明模块
my_seg SEGMENT  CODE               ; 声明代码段my_seg
        RSEG    my_seg             ; 切换到代码段my_seg
TABLE:    DB     3,2,5,0xFF        ; 声明四个常数

myprog SEGMENT CODE                ; 声明代码段myprog
        RSEG    myprog             ; 切换到代码段myprog
        LJMP main                  ; 在程序存储器地址0x0000的位置跳转
        USING 0                    ; 使用第0组寄存器
        ORG 100H                   ; 定位到代码段100H的位置
```

汇编语言程序结构

```
main:   MOV  DPTR, #TABLE        ; 将TABLE表的地址送
                                 ; 给DPTR寄存器
        MOV    A,#3              ; 将立即数3送到累加器A中
        MOVC   A,@A+DPTR         ; 将(A)+(DPTR)所指向的程序
                                 ; Flash的内容送给累加器A
         MOV    P1,0             ; 给P1端口清零
        MOV    P1,A              ; 将累加器A的内容送到P1端口
END
```

注：进入本书配套资料\STC_example\例子7-1\目录下，打开并参考该设计

汇编代码中段的分配

在一个由汇编语言所构建的程序代码中，包括：

- 绝大部分代码都是机器语言助记符。这些程序代码中的机器语言助记符经过软件处理后变成机器指令（机器码），然后保存在程序存储器中。通常地，将保存程序代码的区域称为代码段。
- 根据程序设计的复杂度，需要提供代码中所需数据所在的位置。这些需要操作的数可能保存在不同的存储空间中。因此，就需要在程序中明确的说明这些数据所存放的位置。
  - 在STC 15系列单片机中，提供了片内基本RAM、片内扩展RAM等存储单元。通常地，将保存数据的区域统称为数据段。

注：更具体的划分，会根据不同的CPU的结构有所不同。

汇编代码中段的分配

- 在运行程序的过程中，可能还需要对8051 CPU内功能部件的状态进行保存和恢复操作。
  - 这是由堆栈机制实现的。在这种情况下，就需要明确指出保存这些运行状态的存储空间大小和位置。

注：在任何一个程序中，必须要有代码段，而其它段的存在与否，由具体的程序模型决定。

汇编代码中段的分配
--CODE段

CODE段，也称为代码段，它是用来保存程序中汇编助记符描述的机器指令部分。

- CODE放在STC单片机中的程序Flash存储空间。
- CODE段可以由MOVC指令，并且通过DPTR寄存器进行访问。

汇编代码中段的分配
--CODE段

代码清单　定义和访问CODE段的代码

```
my_seg  SEGMENT  CODE                    ; 定义为CODE段
        RSEG    my_seg
TABLE:    DB     1,2,4,8,0x10            ; 定义常数表
myprog SEGMENT CODE                     ; 定义CODE段
        RSEG    myprog
        MOV     DPTR, #TABLE            ; 加载TABLE的地址
        MOV     A,#3                    ; 加载偏移量
        MOVC    A,@A+DPTR               ;通过MOVC指令访问
END
```

汇编代码中段的分配
--BIT段

在8051汇编语言中,BIT段可以用来保存比特位，可以通过指令系统中的位操作指令来访问BIT段。

注：可以通过位操作指令访问特殊功能寄存器SFR。

- 可位寻址的地址只能是可以被8整除的地址。
  - 典型地，包括：80H，88H，90H，98H，0A0H,，0A8H，0B0H，0B8H，0C0H，0C8H，0D0H，0D8H，0E0H， 0E8H，0F0H和0F8H地址空间。

汇编代码中段的分配
--BIT段

代码清单　定义和访问BIT段的代码

```
mybits SEGMENT BIT                    ; 定义BIT段
       RSEG    mybits
FLAG:    DBIT   1                     ; 保留1位空间
P1       DATA   90H                   ; 8051 SFR 端口1
GREEN_LED  BIT    P1.2                ; 在端口P1的第2引脚P1.2定义
                                       符号GREEN_LED
myprog SEGMENT CODE                   ; 定义为代码段
       RSEG    myprog
       LJMP    main                   ;无条件跳转到main
       ORG  100H                      ;定位到100H的位置
```

汇编代码中段的分配
--BIT段

```
main:     SETB    GREEN_LED              ; P1.2 = 1
          JB      FLAG, is_on            ; 到DATA的直接访问
SETB    FLAG
          CLR     ACC.5                  ; 复位ACC的第5位
          :
is_on:    CLR     FLAG
          CLR     GREEN_LED              ; P1.2 = 0
END
```

注：进入本书配套资料\STC_example\例子7-2\目录下，打开并参考该设计

汇编代码中段的分配
--IDATA段

在8051汇编语言中，可以定义IDATA段。

- 在IDATA段可以定义少量的变量，这些变量将最终保存在STC单片机的片内RAM的高地址和低地址区域中。

注：IDATA的低128个字节和DATA段重叠。

- 可以通过寄存器R0或者R1，间接寻址保存在IDATA段中的变量。

汇编代码中段的分配
--IDATA段

代码清单　定义和访问IDATA段的代码

```
myvars SEGMENT IDATA          ; 定义IDATA段
       RSEG    myvars
BUFFER:  DS     100            ; 保留100个字节
myprog SEGMENT CODE           ; 定义CODE段
       RSEG    myprog
       LJMP    main           ;无条件跳转到main
       ORG     100H           ;定位到100H的位置
main:  MOV     R0,#BUFFER     ; 将BUFFER的地址加载到R0寄存器
       MOV     A,@R0          ; 读缓冲区的内容到寄存器A
       INC     R0             ; R0内的地址递增
       MOV     @R0,A          ; 将A内容写到BUFFER+1存储空间
END
```

注：进入本书配套资料\STC_example\例子7-3\目录下，打开并参考该设计

汇编代码中段的分配
--DATA段

在8051汇编语言中，定义了DATA段，该段指向STC单片机内部数据RAM的低128个字节。

- 通过直接和间接寻址方式，程序代码可以访问在DATA段中的变量。

汇编代码中段的分配
--DATA段

代码清单　定义和访问DATA段的代码

```
myvar SEGMENT DATA                    ; 定义DATA段
        RSEG    myvar
VALUE:    DS     1                    ; 在DATA空间保存一个字节
IO_PORT2  DATA   0A0H                 ; 特殊功能寄存器
VALUE2    DATA   20H                  ; 存储器的绝对地址
myprog SEGMENT CODE                   ; 定义CODE段
        RSEG    myprog
        LJMP    main                  ;无条件跳转到main
        ORG     100H                  ;定位到100H的位置
```

汇编代码中段的分配
--DATA段

```
main:   MOV    A,IO_PORT2          ; 直接访问DATA
        ADD    A,VALUE
        MOV    VALUE2,A
        MOV    R1,#VALUE           ; 加载VALUE 的值到R1
        ADD    A,@R1               ; 间接访问VALUE
END
```

注：进入本书配套资料\STC_example\例子7-4\目录下，打开并参考该设计

汇编代码中段的分配
--XDATA段

在8051汇编语言中，定义了XDATA段，XDATA段指向扩展RAM区域。

- 通过寄存器DPTR和MOVX指令，程序代码就可以访问XDATA段。
- 对于一个单页的XDATA存储空间来说，也可以通过寄存器R0和R1访问。

汇编代码中段的分配
--XDATA段

代码清单　定义和访问XDATA段的代码

```
my_seg SEGMENT XDATA                ; 定义XDATA段
       RSEG    my_seg
XBUFFER:  DS     2                  ; 保留2个字节存储空间
myprog SEGMENT CODE                 ; 定义CODE段
       RSEG    myprog
       LJMP    main                 ; 无条件跳转到main
       ORG     100H                 ; 定位到100H的位置
```

汇编代码中段的分配
--XDATA段

```
main:   MOV    DPTR,#XBUFFER      ; XBUFFER的地址送到DPTR寄存器
        CLR    A                  ; 累加器A清0
        MOVX   @DPTR,A            ; 将累加器A的内容送给DPTR指向的
                                  ; XBUFFER区域
        INC    DPTR               ; 寄存器DPTR的内容加1
        CLR    A                  ; 累加器A清0
        MOVX   @DPTR,A            ; 累加器A的内容送给DPTR指向的
END                               ; XBUFFER区域
```

注：进入本书配套资料\STC_example\例子5-5\目录下，打开并参考该设计