第3章 汇编语言及仿真设计基础

汇编语言概述

汇编语言指令格式

- 一般格式：〔标号:〕 操作码 〔操作数〕 〔;注释〕
- 标号区段是当前指令行的符号地址，其值等于当前指令的机器码首字节在ROM中的存放地址，标号由英文字母开头的1～6个字符组成，不区分大小写，以英文冒号结尾；
- 操作码区段是指令的操作行为，由操作码助记符表征。51单片机共有42个操作码助记符，各由2～5个英文字符组成，不区分大小写；
- 操作数区段是指令的操作对象，根据指令的不同功能，操作数可以是3个、2个、1个或无操作数。操作数大于1时，操作数之间要用英文逗号隔开；
- 注释区段是对指令的解释性说明，用以提高程序的可读性，可以用任何文字描述，以英文分号“;”开始，无须结束符号；
- 汇编语言指令中的标号、操作码和操作数都是字符大小写不敏感的，而且在各区段之间的空格数也不影响指令的功能。

描述操作数的简记符号

- #data 代表一个8位的立即数（常数）
- #data16 代表一个16位的立即数（常数）
- Rn 代表R0～R7中的某个工作寄存器（n = 0～7）
- Ri 代表R0或R1工作寄存器（i = 0或1）
- direct 代表128B范围内某个片内RAM的具体地址，也可以是SFR的名称或地址
- addr16 代表64KB(216)范围内某个RAM或ROM的具体地址
- addr11 代表2KB(211)范围内某个RAM或ROM的具体地址
- rel 代表-128～+127字节范围内某个RAM或ROM地址的偏移量
- bit 代表RAM或SFR中某个位单元的具体地址
- / 代表将随后的位状态取反
- $ 代表当前指令的首地址
- @ 代表以寄存器中的数据作为单元地址

用简记符可以替换具体指令中的操作数，使之还原为指令原型，从而可以在指令手册中查看到该条指令的功能。

MCS-51指令系统简介

- 指令——CPU用于指挥功能部件完成某一指定动作的指示和命令；
- 指令系统——CPU全部指令的集合。MCS-51单片机指令系统共有111条指令，按照实现的基本功能可划分为4大类。

数据传送与交换类指令

- 数据传送类指令的基本通式为：<transfer> <dest>,<src>，它表示将源操作数（src）内容传送给目的操作数（dest），传送后源操作数中的内容不变；
- 共使用8种操作码助记符，MOV用于访问片内RAM，MOVX用于访问片外RAM，MOVC用于访问程序存储器，XCH和XCHD用于字节交换，SWAP用于A内半字节交换，PUSH和POP用于堆栈操作；

数据传送目的和源的快捷记忆法（图3.1）
- 立即数和ROM地址只能作为源操作数（单向箭头）；
- ROM只能用MOVC操作码助记符并通过A进行数据传送（单向箭头）；
- 片外RAM只能用MOVX操作码助记符并通过A进行数据传送（双向箭头）；
- PUSH和POP只能在堆栈和direct间操作（单向箭头）；
- 位数据传送只能在C（即Cy标志位）与bit间进行（双向箭头）。

算术运算类指令

- 算术运算类指令共有24条，可实现加、减、乘、除4种基本运算功能；
- 共使用8种操作码助记符，其中ADD和ADDC用于加法运算，SUBB用于减法运算，MUL和DIV用于乘法和除法运算，INC和DEC用增1和减1，DA用于十进制数加法调整。

特点：算术运算指令一般对程序状态字寄存器PSW中的CY、AC、OV和P四个标志位有影响，

算术运算指令快捷记忆法（图3.5）
- 算术运算指令共涉及7类操作数，即@Ri、A、Rn、#data、B、direct和DPTR；
- 单向箭头表示只能从源到目的方向，弧线箭头表示源和目的相同，箭头线旁边的文字是相应的操作码；
- 除INC和DEC操作码外，算术运算都要以A为目标操作数，即A必须参与运算并存放运算结果。

逻辑运算及移位类指令

逻辑运算及移位类指令共有34条，可以实现二进制数的与、或、异或、求反、置1、清零、移位等逻辑操作。共使用10种操作码助记符，其中ANL、ORL和XRL分别用于逻辑与、逻辑或和逻辑异或运算，CPL用于求反运算，SETB和CLR用于置位和清零运算，RL、RLC、RR和RRC用于循环移位。

特点：逻辑运算指令中不以累加器A为目标寄存器的指令均不影响PSW中任何标志位，带进位的移位指令影响CY位

逻辑运算指令快捷记忆法（图3.7）
- 共涉及5种操作数，分别是@Ri，A，Rn，#data，direct；
- 除direct与#data的逻辑关系外，其余逻辑运算都与A有关，且几乎都以A为目的操作数；
- 位运算都以C为目的操作数。

控制转移类指令

- 控制转移类指令共有22条，主要功能是通过改变程序计数器PC的内容，进而实现程序转移功能；
- 共涉及18种操作码助记符，其中LJMP、AJMP、SJMP、JMP是无条件转移指令，JZ、JNZ、JC、JNC、JB、JNB、JBC、CJNE、DJNZ是条件转移指令，LCALL、ACALL、RET、RETI是子程序调用及返回指令，NOP是空操作指令。

条件转移类指令快捷记忆法（图3.10）
- direct和Rn都可实现“减一非零转移”；
- @Ri、A、Rn可分别与#data，A与direct可实现“比较不等转移”；
- A可实现为零或非零转移；
- 所有条件转移都只能是在rel的范围内进行，即-128～+127。
- 无条件转移指令中LJMP、AJMP 、SJMP和JMP的最大转移范围分别是65535，2047，-128～127和65535。

寻址方法汇总

1. 直接寻址——指令中包含direct形式操作数的寻址方式称为直接寻址，其中direct既可以是片内RAM的低128字节地址，也可以是不含A、B、C和DPTR在内的其他特殊功能寄存器名。
2. 寄存器寻址——指令中包含寄存器形式操作数的寻址方式称为寄存器寻址，其中寻址寄存器只能是Rn、A、B、CY和DPTR 5种类型。
3. 寄存器间接寻址——指令中包含“@间址寄存器”形式操作数的寻址方式称为寄存器间接寻址，其中间址寄存器只能由R0、R1或DPTR三个寄存器兼任。
4. 立即寻址——指令中包含#data或#data16形式操作数的寻址方式称为立即寻址。
5. 变址寻址——指令中包含“@A+基址寄存器”形式操作数的寻址方式称为变址寻址方式，其中基址寄存器只能由DPTR或PC兼任。
6. 位寻址——指令中包含bit形式操作数的寻址方式称为位寻址方式，其中bit形式的位地址可以是片内RAM中可位寻址区内的位地址，也可以是SFR中的位地址或位名称。
7. 相对寻址——指令中包含rel形式操作数的寻址方式称为相对寻址方式，其中rel是片内RAM或ROM 8位地址偏移量。

伪指令

常用的几种伪指令
- ORG（程序起点）——ORG 地址或表达式——用于定义汇编程序或查表数据在ROM中存放的起始地址
- EQU（等值指令）——符号名 EQU 数或汇编符号——用于将一个数值或汇编符号赋给该标示符
- DATA（数据地址赋值）——符号名 DATA 内存字节地址——用于将一个片内RAM的地址赋给该标示符
- BIT（位地址赋值）——符号名 DATA 位地址或位名称——用于将一个位地址或位名称赋给该标示符
- DB（定义字节）——〔标号:〕DB 字节值或字符串——用于将字节值或字符串依次存入标号开始的存储单元中
- END（结束汇编）——END——用于指示汇编源程序段结束

汇编程序仿真设计基础

汇编程序设计步骤

1. 分析问题，确定算法或解题思路——必须具体问题具体分析，通过认真比较从中挑选最佳方案。
2. 画流程图——可充分地表达程序的设计思路，将问题与程序联系起来，便于阅读、理解程序，查找错误。
3. 编写程序——用汇编指令对流程图中的各部分加以具体实现。如果流程复杂，可以采取分别编写各个模块程序，然后汇总成完整程序的做法。
4. 调试与修改——进行反复调试和修改，直至问题完全排除。

汇编程序编译方法

约定采用*.asm格式的汇编程序，采用基于Keil for 8051的编译方法

Proteus准备

创建新工程中→在创建固件项目单选框中选择8051、80C51、Keil for 8051三个选项→不勾选“创建快速启动文件”选项→单击“确定”按钮结束设置。

汇编程序的仿真设计

1. 添加并保存*.asm文件→打开添加的汇编程序空白页面；
2. 编辑程序文件→对源程序进行录入，编辑，保存；
3. 程序编译→调用Keil for 8051进行程序编译，根据编译结果进行查错；
4. 仿真运行→对可执行文件进行仿真运行。

汇编程序应用举例

电路分析

P2口外接8个发光二极管，P2。0-P2.7轮流输出低电平可产生流水灯效果

编程要点

编码初值应为0FEH（保证D1亮灯其余灭灯）此后，不断将亮灯编码值进行循环左移输出，亮灯位也将随之由上向下变化；循环左移7次后再改为循环右移，则亮灯位将随之由下向上变化。

程序中用到了3种伪指令
- 语句ORG 30H表示本程序编译后产生的机器码将从ROM 30H地址开始依次加载；
- 语句CYC1 EQU 200表示，程序中的CYC1标识符汇编时将会用常数200代替；
- 语句END用于通知编译器源程序结束，如果缺少END，程序编译时就会报错。