MCS－51 单片机指令快速记忆方法总结

单片机是一种集成在电路芯片，是采用超大规模集成电路技术把具有数据处理能力的中央处理器 CPU 随机存储器 RAM、只读存储器 ROM、多种 I/O 口和中断系统、定时器/计时器等功能（可能还包括显示驱动电路、脉宽调制电路、模拟多路转换器、A/D 转换器等电路）集成到一块硅片上构成的一个小而完善的计算机系统。

MCS-51 是指由美国 INTEL 公司生产的一系列单片机的总称，这一系列单片机包括了好些品种，如 8031，8051，8751，8032，8052，8752 等，其中 8051 是最早最典型的产品。

学习单片机，除了搞清单片机内部功能、存储空间分配及 I/O 接口外，还应掌握其指令系统。MCS－51 共有 111 条指令，现介绍我们总结出的快速记忆 MCS－51 指令的方法，供大家参考。

大家都知道，汇编语言指令由操作码、操作数两部分组成。MCS－51 使用汇编语言指令，它共有 44 个操作码助记符，33 种功能，其操作数有＃data、direct、Rn、@Ri 等。这里先介绍指令助记符及其相关符号的记忆方法。

一、助记符号的记忆方法

1 表格列举法

把 44 个指令助记符按功能分为五类，每类列表记忆。此处从略，请读者自己总结。

2 英文还原法

单片机的操作码助记符是该指令功能的英文缩写，将缩写还原成英语原文，再对照汉语有助于理解其助记符含义，从而加强记忆。例如：

增量 INC－Incremect

减量 DNC－Decrement

短转移 SJMP－Short jump

长转移 LJMP－Long jump

比较转移 CJNE－Compare jump not equality

绝对转移 AJMP－Absolute jump

空操作 NOP－No operation

交换 XCH－Exchange

加法 ADD－Addition

乘法 MUL－Multiplication

除法 DIV－Division

左环移 RL－Rotate left

进位左环移 RLC－Rotate left carry

右环移 RR－Rotate right

进位右环移 RRC－Rotate right carry

3 功能模块记忆法

单片机的 44 个指令助记符，按所属指令功能可分为五大类，每类又可以按功能相似原则为 2～3 组。这样，化整为零，各个击破，实现快速记忆。

1）数据传送组

2）加减运算组：

MOV 内部数据传送

ADD 加法

MOVC 程序存储器传送

ADDC 带进位加法

MOVX 外部数据传送

SUBB 带进位减法

3）逻辑运算组

4）子程序调用组：

ANL 逻辑与

LCALL 长调用

ORL 逻辑或

ALALL 绝对调用

XRL 逻辑异或

RET 子程序返回

二、指令的记忆方法

1 指令操作数的有关符号

MCS－51 的寻址方式共有六种：立即数寻址、直接寻址、寄存器寻址、寄存器间址、变 址寻址、相对寻址。我们必须掌握其表示的方法。

1）立即数与直接地址。ata 表示八位立即数，＃data16 表示是十六位立即数，data 或 direct 表示直接地址。

2）Rn(n=0－7)、A、B、CY、DPTR 寄存器寻址变量。

3）@R0、@R1、@DPTR、SP 表示寄存器间址变量。

4）DPTR＋A、PC＋A 表示变址寻址的变量。

5）PC＋rel（相对量）表示相对寻址变量。

记住指令的助记符，掌握不同寻址方式的指令操作数的表示方法，为我们记忆汇编指 令打下了基础。MCS－51 指令虽多，但按功能可分为五类， 其中数据传送类 28 条，算术运算类 24 条，逻辑操作类 25 条，控制转移类 17 条，布尔位操作 类 17 条。在每类指令里，根据其功能，抓住其源、 目的操作数的不同组合，再辅之以下方法，是完全能记住的。 我们约定，可能的目的操作数按（＃ data/direct/A/Rn/@Ri ）顺序表示。

对于 MOV 指令，其目的操作数按 A、Rn、direct、@Ri 的顺序书写，则可以记住 MOV 的 15 条指令。例如以累加器 A 为目的操作数，可写出如下 4 条指令。

MOV A，＃ data/direct/A/Rn/@Ri

以此类推，写出其它指令。

MOV Rn，＃data/direct/A

MOV direct，＃ data/direct/A/Rn/@Ri

MOV @Ri，＃data/direct/A

2 指令图示记忆法

图示记忆法是把操作功能相同或相似、但其操作数不同的指令，用图形和箭头将目 的、源操作数的关系表示出来的一种记忆方法。 例如：由助记符 MOV、MOVX、MOVC 组成的送数组指令，可以用图 1、2 帮助记忆。

由助记符 CJNE 形成的四条指令，也可以用图示法表示，如图 3。 CJNE A，＃data，rel CJNE A，direct，rel CJNE @Rn，＃data，rel CJNE @Ri，＃data，rel

另外，对于由（ANL、ORL、ARL）形成的 18 条逻辑操作指令，有关 A 的四条环移指令， 也可以用图示法表示，请读者自行画出记忆。

3 相似功能归类法

在 MCS－51 指令中，我们发现部分指令其操作码不同，但功能相似，而操作数则完全一 样。相似功能归类法就是把具有这样特点的指令放在一起记忆， 只要记住其中的一条，其余的也就记住了。如加、减法的十二条指令，与、或、非的十八 条指令，现列举如下。

ADD/ADDC/SUBB A， ＃ data/direct/Rn/@Ri

ANL/ORL/XRL A，＃ data/direct/Rn/@Ri

ANL/ORL/XRL direct，＃data/a

上述每一排指令，功能相似，其操作数都相同。其它的如加 1(INC)、减 1(DEC)指令也可照此办理。

4 口诀记忆法

对于有些指令，我们可以把相关的功能用精练的语言编成一句话来记忆。如 PUSH direct 和 POP direct 这两条指令。 初学者常常分不清堆栈 SP 的变化情况，为此编成这样一句话：(SP 的内容)加 1(direct 的内 容)再入栈，(SP 的内容)弹出(到 direct 单元)SP 才减 1。 又如乘法指令中积的存放，除法指令中被除数和除数以及商的存放，都可以编成口诀记忆如下：

MUL AB

高位积(存于)B，低位积(存于)A。

DIV AB

A 除以 B，商(存于)A 余(下)B。

上面介绍了几种快速记忆单片机指令的方法，希望能起到抛砖引玉的作用，相信读者在学习单片机的过程中能找到适合自己的方法来记忆。但是，有了好的方法还不够，还需要实践，即多读书上的例题和别人编写的程序，自己再结合实际编写一些程序。只有这样， 才能更好更快地掌握单片机指令系统。

建议

学单片机之初，你必须懂一些数字电路，若对数字电路中的一般概念都很模糊，最好还是再补习一下再来学单片机。接下来你最好先选一种单片机机种进行学习，因为目前单片机机种较多，其结构和指令均不相同，若这种学两天，那种学两天往往会滩多嚼不烂。这里建议你最好先学 8051 单片机，因为 8051 方面的书籍、资料、器材都较多。PIC 和 AVR 以及其它类型的单片机虽有其长处，但现在的书籍、资料以及器件供应并不理想，不太适合初学者选择。若你对这些并不在意的话那选择后者进行学习也未尚不可。

我们建议你选择 8051 单片机开始学习的原因还在于 8051 家族的派生品很多，例如 ATMEL 公司的 AT89C51 系列单片机就是完全兼容 MSC-51 8051 系列的（也就是说，AT89C51 的指令、管脚、内部主要结构，以及用法与 MSC-51 相同），他不但兼容，而且还有不少创新，比如他的程序存储器可以电擦、写，一片 IC 就拥有了过去单片机的最小系统，不需要以前所谓的 373 和 EPROM 元件；所以，实验时的电路连接、电路板自制都比较容易，加上目前其价格较底，你学习的片子也可以做产品，做产品的片子也可以做实验，当然 AVR 系列也有这些特点；而 PIC 及其它系列在这一点上则显得不太理想。

购买单片机的书籍最好是书的前面你能看懂，而书的后面你不懂，若前后都看不懂的书最好先别买，因为这本书短时间内不会对你起多大作用。当然若不是把书当资料查也不必买前后你都懂的书，因为它对你来说有点浅。应以原理书籍为主。其次可以购买一些应用方面的书籍以便参考。

电子技术本身与实验离不开，若光靠看书是很难理解其原理和学会单片机开发的。你应该购置相关单片机的芯片、编程器、实验板，以及开发他的相关软件。并以边看书边实验的方式进行学习其效果将明显好的多！由于初学，不可能购置很多昂贵的设备，建议学习用的单片机芯片其程序储存器是可以反复可擦写的，如 AT89C 系列或 AVR 系列。这样，在学习烧写时是无后顾之忧的。

现在来谈谈单片机开发的步骤。想让单片机按你的意思（想法）完成一项任务，必须先编写供其使用的程序，编写单片机的程序应使用该单片机可以识别的“语言”，否则你将是对“石”弹琴。目前较流行的有汇编和 C 语言；汇编语言可以精确的控制单片机工作的每一步，而 C 语言则注重结果，不必关心单片机具体的每一步。习惯上宜先学汇编语言后学 C 语言，这样可以对单片机有一个更深的了解，再说，就是用 C 语言编程，在需要精确控制时还需要嵌入汇编语句。当然，也有一开始就用 C 语言的，后来再学汇编；若你学过计算机的 Turbo C ，开始就学单片机的 C 也许会更快一些。

单片机程序是用文本编辑器编写的纯文本文件，象我们平常在 windows 计事本中用汉语写计划一样，先这件事后那件事的去写，以所使用单片机语言的语法，按我们的想法把单片机要做的事“一件一件”的依次写下来，遇到“有些事”是重复的，就指明在什么什么地方已有说明（跳转），在正常安排中若有其它突发事件出现，必须写一段突发事件处理计划（中断）... ...。最后保存文件的扩展名应与所使用的语言要求的名字一致；我们汉语的文章一般保存为*.txt 扩展名，而汇编语言的文件扩展名一般应为*.asm；有的开发系统则有自己的规定，如用 Keil C51 开发系统，编写的汇编程序扩展名为*.a51；当然 Keil C51 开发系统也有自己的编辑器，不必用 Windows 中的记事本。

无论我们使用汇编语言，还是 C 语言编写的程序，只是给我们看的，这个程序还必须经过与该语言对应的软件将我们能看懂的汇编或 C“翻译”（编译）成所用单片机可以识别的代码。将单片机可以识别的代码烧写（编程）到单片机程序存储器中，单片机装的实际电路中才能依你的“计划”去工作。

对于 8051 系列单片机来说，Keil C 开发系统具有编辑、编译、模拟单片机 C 语言程序的功能，也能编辑、编译、模拟汇编语言程序；对于初学者，开始编写的程序难免出现语法错误或其它不规范的语句，由于 Keil C 编译时对错误语句提示的是英文，不太好理解，若用汇编的话，可使用 DOS 下的宏汇编编译器 ASM51；他可以对出错语句进行中文提示；你源程序的注释部分还可以使用中文，这更便于你今后对程序的维护。

编译出的代码一般扩展名为*.hex 或*.bin；这个代码文件必须送到单片机中单片机在电路中才能按你的“计划”去工作。将这个代码文件送到单片机中的工具就是编程器，与电脑连接的编程器一般都通过并口或者串口与编程器的硬件连接，也有相应的服务程序；在连接好电脑与编程后运行其服务程序，在服务程序中先选择所要编程的单片机型号，再调入前面所得到的代码文件，接下来就用编程器将这个代码文件烧写到单片机中。到此，单片机开发的一个过程就大致完成。

当然，你不可能一次就把你的“计划”用单片机的语言完美正确的将源程序写好（就是我们平常制订的计划在实际中也有修改的），这就需要反复修改源程序，反复编译、烧写到单片机中、反复将单片机装到电路中去实验。由于单片机执行每一条语句所用的时间很短，有时你无法得到其中间的结果，也无法判断程序出错的位置，这时你可以使用软件模拟的方法，让程序一步一步的执行，每执行一步，通过查看单片机中各关键数据的变化情况，来找到错误或没按你“计划”执行的语句，从而达到排错的目的。若你资金不成问题的话，也可以购买单片机仿真器，他可以取代实际电路中的单片机，在电脑的控制下一步一步的去排错。实际上无论软件模拟（仿真）还是硬件仿真，其功能远不止这里讲的这一点点。