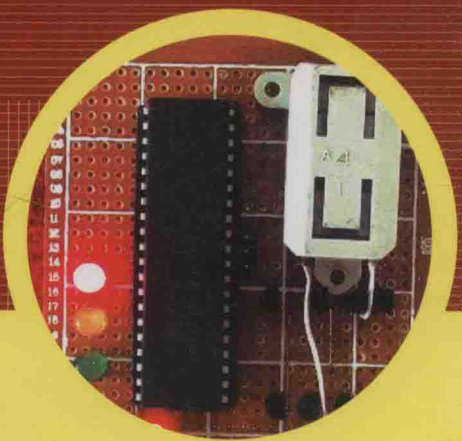
51单片机C语言开发教程

刘理云　编著

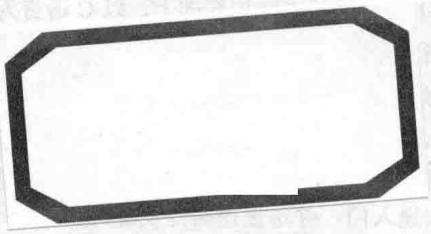
60个实用案例快速入门

详细、可移植的源代码

课程平台讲解、答疑解惑



化学工业出版社



51单片机C语言开发教程

刘理云　编著

化学工业出版社

·北京·

本书在引导读者认识C51单片机基本结构基础上，以C语言为设计语言，通过60个案例、详细的源代码介绍了C51单片机程序开发的各项细节，包括单片机应用系统仿真开发、接口应用技术、中断系统与定时/计数器设计、串行接口技术等。程序代码经典，可移植性强：大部分代码写成傻瓜式，对C51单片机可直接套用，也容易移植到AVR、PIC等单片机中去，节省了开发时间。

全书案例丰富，程序代码可靠，并可以在相应的平台下载，帮助单片机开发人员、电子爱好者以及从事智能电子产品开发的人员快速入门，并迅速提高开发能力。

图书在版编目(CIP)数据

51单片机C语言开发教程 / 刘理云编著. —北京：化学工业出版社，2017.9
ISBN 978-7-122-30134-5

Ⅰ. ①5… Ⅱ. ①刘… Ⅲ. ①单片微型计算机-C语言-程序设计 Ⅳ. ①TP368.1②TP312.8

中国版本图书馆CIP数据核字（2017）第161851号

责任编辑：刘丽宏　　文字编辑：孙凤英
责任校对：王　静　　装帧设计：刘丽华

出版发行：化学工业出版社（北京市东城区青年湖南街13号　邮政编码100011）
印　　装：三河市延风印装有限公司
787mm×1092mm　1/16　印张22　字数572千字　　2017年9月北京第1版第1次印刷

购书咨询：010-64518888（传真：010-64519686）　售后服务：010-64518899
网　　址：http: // www. cip. com. cn
凡购买本书，如有缺损质量问题，本社销售中心负责调换。

定　　价：68.00元

前　言

对于刚接触单片机的学习者，似乎都很迷茫，C 语言、汇编语言、电路、开发板，不知道从哪儿开始学起。其实在学习单片机原理与应用系统开发时，只有在学习一些理论知识的基础上，多阅读单片机应用系统开发案例，注重单片机应用系统开发实践训练，才能透彻地理解和掌握单片机结构与原理，才能更快更好地掌握单片机应用知识和单片机应用系统开发技能。

本书结合笔者多年的教学和实践经验，根据案例阅读与实践训练在学习单片机应用系统开发过程中的重要性，用 C 语言为编程语言，由易到难，循序渐进地讲述 51 单片机的硬件结构和功能应用，以及 C 语言为 51 单片机编程的方法。

全书具有以下特点：

一、内容全面，既有基础知识介绍，又重视开发技能、技巧的说明，初学者可以掌握全面的硬件原理和编程技巧，建立比较完善的知识体系。

二、案例丰富，源代码可靠，讲解循序渐进，详细生动，为读者打好基础，增强学习兴趣和信心。

三、相应课程平台帮读者答疑解惑，读者可以少走弯路，快速入门。

笔者基于世界大学城平台创建了一个自助学习空间，有丰富的学习资源和教学课件等，课程空间链接：http://www.worlduc.com/SpaceShow/Index.aspx?uid=359771。

本书由刘理云编著，祖国建审稿。在编写本书过程中，参阅了许多文献资料，在此谨向各位作者表示诚挚的感谢。

单片机和电子技术的知识发展迅猛，涉及的应用面广，知识更新快，加上笔著者水平有限，虽经努力，但书中仍难免有不足之处，恳请广大读者指正。

刘理云

目 录

第1章 C51单片机基本结构与最小应用系统

单片机就是一块集成芯片，在这块集成芯片里集成了中央处理部件（CPU）、存储器（RAM、ROM）、定时/计数器和各种输入/输出（I/O）接口等，能够完成一些特殊功能，而它的功能的实现要靠使用者自己来编程完成。编程的目的就是控制这块芯片的各个引脚在不同时间输出不同的电平（高电平或低电平），进而控制与单片机各个引脚相连接的外围电路的电气状态。可见单片机就是一台计算机，其全称是单片微型计算机（Single Chip Micro-computer），是微型计算机发展中的一个重要分支。由于单片机原来就是为实时控制应用而设计制造的，故又称为微控制器（Microcontroller）。

1.1 51单片机的基本结构

1.1.1 51单片机内部的逻辑结构

51单片机芯片内部结构非常复杂，但作为单片机的用户，只需要了解其逻辑结构和功能就足够了。51单片机芯片内部集成了微型计算机所需的基本功能部件，它由CPU、振荡与时钟电路、程序存储器、数据存储器、定时/计数器、串行口、并行口、总线扩展控制、中断等部分组成，各部分之间通过内部总线相连接，如图1-1所示。

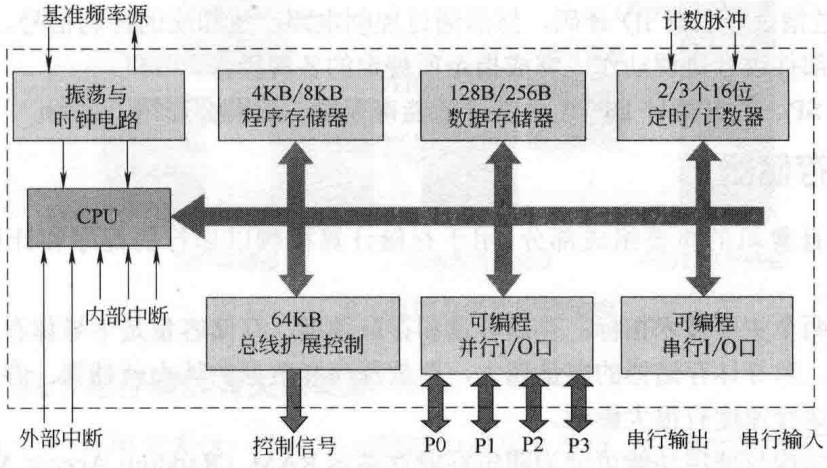


图1-1 51单片机简化逻辑结构图

51单片机有多种型号的产品：

普通型（51子系列）有：8051、8031、8751、89C51、89S51等型号。

增强型（52子系列）有：8032、8052、8752、89C52、89S52等型号。

它们的结构基本相同，其主要差别反映在存储器的配置上。

8031片内没有程序存储器；

8051内部设有4KB的掩模ROM程序存储器；

8751是将8051片内的ROM换成EPROM；

89C51则换成4KB的闪速EEPROM；

89S51结构同89C51， 4KB的闪速EEPROM可在线编程。

增强型的存储容量为普通型的一倍。另外普通型只有2个16位的定时/计数器，而增强型有3个16位定时/计数器。

1.1.2 CPU

CPU是单片机的核心部件，它由运算器和控制器等部件组成。

（1）运算器ALU

运算器ALU由一个加法器、两个8位暂存器（TMP1与TMP2）、8位的累加器A、寄存器B和程序状态寄存器PSW以及一个布尔处理器组成（累加器A、寄存器B和程序状态寄存器PSW是特殊功能寄存器，后面专门介绍）。运算器ALU可以对8位数据进行加、减、乘、除、加1、减1、比较、BCD码十进制调整等算术运算和与、或、异或、求反、循环等逻辑运算，并且能够完成数据传送、移位、判断和程序转移等操作。利用布尔处理器能够对位数据进行传送、逻辑运算、判断和程序转移等操作。位信号处理能力是51系列单片机的重要特色。

（2）控制器

控制器是用来控制计算机工作的部件，它包括程序计数器PC、指令寄存器IR、指令译码器ID、堆栈指针SP、数据指针DPTR、时钟发生器和定时控制逻辑等。

① 程序计数器PC（Program Counter）。程序计数器PC是一个16位的专用寄存器，PC中存放的内容是：CPU将要执行的下一条指令的地址，可对64KB程序存储器直接寻址，每读取指令的一个字节，PC的内容自动加1，故称为程序计数器。

② 指令寄存器IR和指令译码器ID的功能：从程序存储器取出的指令先存放指令寄存器IR中，再送指令译码器ID译码，然后通过控制电路产生相应的控制信号，控制CPU内部及外部有关部件进行协调动作，完成指令所规定的各种操作。

堆栈指针SP、数据指针DPTR是特殊功能寄存器，后面会介绍。

1.1.3 存储器

存储器是计算机的重要组成部分，用于存储计算机赖以运行的程序和计算机处理的对象——数据。

存储器有两个主要技术指标：存储容量和存取速度。存储容量是半导体存储器存储信息量大小的指标。半导体存储器的容量越大，存放程序和数据的能力就越强。存储器的存取速度对计算机的运行速度有很大影响。

存储器按结构与使用功能可分为随机存取存储器RAM（Random Access Memory）和只读存储器ROM（Read Only Memory）两类。随机存取存储器RAM又称读写存储器，它的数

据既可以从RAM中读数据，又可以将数据写入RAM，但掉电后RAM中存放的信息将丢失。RAM适宜存放原始数据、中间结果及最后的运算结果，因此又被称作数据存储器。只读存储器ROM在计算机运行时只能执行读操作，但掉电后ROM中存放的数据不会丢失。ROM适宜存放程序、常数、表格等，因此又称为程序存储器。

随着半导体存储技术的发展，新的可现场改写信息的非易失存储器逐渐被广泛采用，且发展速度很快。主要有快擦写FLASH存储器，新型非易失静态存储器NVSRAM和铁电存储器FRAM。这些存储器的共同特点是：从原理上看，它们属于ROM型存储器，但是从功能上看，它们又可以随时改写信息，因而作用又相当于RAM。新型的非易失性存储器在这两类存储器之间搭起了一座跨越沟壑的桥梁。

51单片机的存储器结构与常见的微型计算机的配置方法不同,它将程序存储器和数据存储器分开,各有自己的寻址方式、控制信号和功能。程序存储器用来存放程序和始终要保留的常数。数据存储器存放程序运行中所需要的常数和变量。

从物理空间看，51单片机有四个存储器地址空间：片内数据存储器、片外数据存储器、片内程序存储器、片外程序存储器。

51单片机存储器物理结构如图1-2所示：

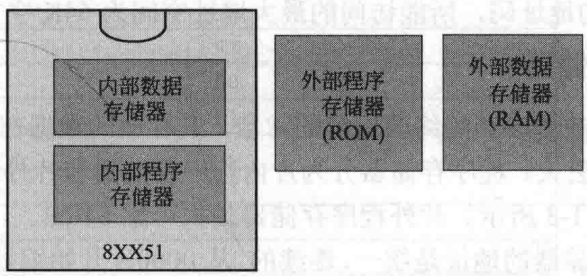


图1-2 51单片机存储器物理结构图

从逻辑上看，51单片机有三个存储器空间：片内数据存储器、片外数据存储器、片内片外统一编址的程序存储器。

51单片机的存储器逻辑结构如图1-3所示。

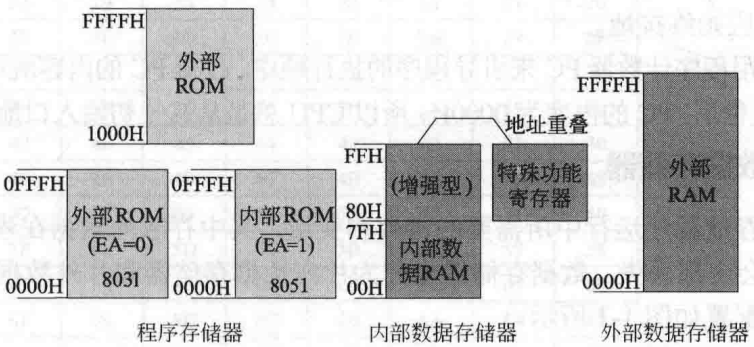


图1-3 51单片机的存储器逻辑结构

1.1.3.1 几个与存储器有关的概念

• **位（bit）**：位是用来表达一个二进制信息的最基本单位。在存储器中，一位信息“1”或“0”的存储是由具有记忆功能的半导体电路（如触发器）实现的。

·字节（byte）：字节是计算机处理或存储数据的最常用基本单位，一个字节由一定位（8位）数的二进制代码组成，常简写成“B”。

·字长：一般将计算机中一个字节的二进制代码的长度称为字长。计算机的字长越长，它能代表的数值就越大，能表示的数值的有效位数也越多，计算的精度就越高，它能处理的信息也越复杂。但是，字节位数越多，用来表示二进制代码的逻辑电路也越多，使得计算机的结构变得庞大，电路变得复杂，造价也越昂贵。用户通常要根据不同的任务选择不同字长的计算机。单片微型计算机使用的字长以8位的为主，也有4位、16位和32位的。

·容量：存储器的容量是指在一块芯片中所能存储的信息位数。例如：2832 E^2PROM芯片的容量为32768 bit。存储器的容量，更多的是用字节数表示，如2832的存储容量可写成：4K×8bit，通常表示为4KB。

·存储单元及其地址：一个字节所占用的存储空间称为一个存储单元，4K字节的存储器就有4K存储单元。微型计算机存取信息，一般是以字节为单位，存取信息首先就要寻找信息字节的存储单元，为了准确无误地存放和取用数据，每个存储单元都有一个用二进制数编的号，即地址码，用地址码标识其所处的物理空间位置。地址码的有效位数，可以根据存储容量的大小而定。换句话说，地址码的位数也可以反映计算机的寻址范围。51单片机最多可以使用16位二进制的地址码，所能访问的最大地址空间为64K字节。

1.1.3.2 程序存储器

程序存储器用来存放程序和始终要保留的常数，其存放的数据在程序运行过程中不会发生改变，断电后也不会丢失。程序存储器分为片内程序存储器和片外程序存储器，51单片机的程序存储器配置如图1-3所示。片外程序存储器最大可配64KB。

内部和外部程序存储器的地址是统一、连续的，从0000H开始编址，最大地址是0FFFFH。从图1-3可看出，内部程序存储器和外部程序存储器的低4KB的地址是重叠的，在使用时，程序存储器0000H～0FFFH（低4KB）是使用内部的还是使用外部的，由31脚（$\overline{EA}$）来区分。31脚（$\overline{EA}$）接高电平时，CPU将首先访问内部存储器，当指令地址超过0FFFH时，自动转向片外ROM去取指令。31脚（$\overline{EA}$）接低电平时，CPU将从外部程序存储器的0000H开始访问。8031单片机无内部程序存储器，地址从0000H～FFFFH都是外部程序存储空间，故31脚（$\overline{EA}$）应始终接地。

51单片机中用程序计数器PC来引导程序的执行顺序，改变PC的内容就可以改变程序执行的顺序。当系统复位后，PC的内容为0000H，所以CPU总是从这一初始入口地址开始执行程序。

1.1.3.3 数据存储器

数据存储器存放程序运行中所需要的常数和变量，其中存放的数据在程序运行中可能发生改变，断电后会全部丢失。数据存储器也分为片内数据存储器和片外数据存储器，51单片机的数据存储器配置如图1-3所示。

（1）片外数据存储器

片外数据存储器RAM也有64KB寻址区，在地址上与程序存储器ROM重叠，51单片机通过不同的信号来选通ROM或RAM，当从外部ROM取指令时，用选通信号$\overline{PSEN}$，而当从外部RAM读/写数据时，采用读/写信号$\overline{RD}$或$\overline{WR}$来选通，从内部ROM取指令或读/写内部RAM不要用选通信号，因此不会因为地址重叠而出现混乱。

51单片机的外部数据存储器和外部I/O口实行统一编址，并使用相同的$\overline{RD}$或$\overline{WR}$作选通控制信号。

（2）内部数据存储器

内部数据存储器是最灵活的存储区，51单片机片内RAM分成性质不同的几个区：00H～7FH单元组成的低128字节地址空间的RAM区；80H～FFH单元组成的高128字节的RAM区（仅在增强型中有这一区）；80H～FFH地址空间内的专用寄存器区（又称特殊功能寄存器）。在89C51等普通型中只有低128字节的RAM区和128字节的专用寄存器区。

51单片机的片内RAM分成工作寄存器区、位寻址区和数据缓冲区三部分（如表1-1所示）。

表1-1 51单片机片内RAM的结构和功能分区

		D7	D6	D5	D4	D3	D2	D1	D0	
工作寄存器区	00H	R0								工作寄存器0组
	01H	R1								
	⋮	⋮								
	07H	R7								
	08H	R0								工作寄存器1组
	09H	R1								
	⋮	⋮								
	0FH	R7								
	10H	R0								工作寄存器2组
	11H	R1								
	⋮	⋮								
	17H	R7								
	18H	R0								工作寄存器3组
	19H	R1								
	⋮	⋮								
	1FH	R7								
位寻址区	20H	07	06	05	04	03	02	01	00	
	21H	0F	0E	0D	0C	0B	0A	09	08	
	22H	17	16	15	14	13	12	11	10	
	23H	1F	1E	1D	1C	1B	1A	19	18	
	24H	27	26	25	24	23	22	21	20	
	25H	2F	2E	2D	2C	2B	2A	29	28	
	26H	37	36	35	34	33	32	31	30	
	27H	3F	3E	3D	3C	3B	3A	39	38	
	28H	47	46	45	44	43	42	41	40	
	29H	4F	4E	4D	4C	4B	4A	49	48	
	2AH	57	56	55	54	53	52	51	50	
	2BH	5F	5E	5D	5C	5B	5A	59	58	
	2CH	67	66	65	64	63	62	61	60	
	2DH	6F	6E	6D	6C	6B	6A	69	68	
	2EH	77	76	75	74	73	72	71	70	
	2FH	7F	7E	7D	7C	7B	7A	79	78	
数据缓冲区	30H									
	31H									
	⋮	⋮								
	7FH									

① 工作寄存器区　片内RAM低端的00H～1FH共32个单元，分成4个工作寄存器组，每组占8个单元。

- 寄存器0组：地址00H～07H；
- 寄存器1组：地址08H～0FH；
- 寄存器2组：地址10H～17H；
- 寄存器3组：地址18H～1FH。

每个工作寄存器组都有8个寄存器，分别称为：R0，R1，…，R7。程序运行时，只能有一个工作寄存器组作为当前工作寄存器组，其余的可以作一般的RAM使用。当前程序使用的工作寄存器选组由程序状态字PSW的RS1和RS0位定。

RS1	RS0	选寄存器组
0	0	0组
0	1	1组
1	0	2组
1	1	3组

复位后，系统默认寄存器0组为当前工作寄存器组。

② 位寻址区　片内RAM低端的20H～2FH共16个单元是位寻址区。该区的每一存储位都有位地址，位地址范围为00H～7FH，既可以按字节寻址，也可以按位寻址。关键字bit可以定义存储于位寻址区中的位变量。bit型变量的定义方法如下：

```
bit  flag;   // 定义一个位变量 flag
bit  flag=1; // 定义一个位变量 flag 并赋初值 1
```

③ 数据缓冲区　片内RAM的30H～7FH单元是数据缓冲区，或称通用RAM区，共有80个单元，用于存放用户数据。数据缓冲区只能以字节为单位执行操作。除选中的寄存组以外的存储器均可以作为通用RAM区。

在一个实际的程序中，往往需要一个后进先出的RAM区，以保存CPU的现场，这种后进先出的缓冲器区称为堆栈区。51单片机的堆栈原则上可以设在内部RAM的任意区域内，但一般设在30H～7FH的范围内。栈顶的位置由堆栈指针SP指出，初始化时SP指向07H。

（3）特殊功能寄存器 SFR（Special Function Register）

特殊功能寄存器区（或称专用寄存器区）中离散地布置了18个专用寄存器，它们分散地分布在内部RAM地址空间范围80H～FFH内。其中，DPTR、T0和T1分别由2个字节组成，所以，专用寄存器共占用21个字节。各专用寄存器的名称、符号、字节地址和位地址如表1-2所示。

表1-2　51单片机的特殊功能寄存器

D7			位地址				D0	字节地址	SFR	寄存器名
P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0	80	P0*	P0端口
87	86	85	84	83	82	81	80			
								81	SP	堆栈指针
								82	DPL	数据指针
								83	DPH	
SMOD								87	PCON	电源控制

续表

D7			位地址				D0	字节地址	SFR	寄存器名
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	88	TCON*	定时器控制
8F	8E	8D	8C	8B	8A	89	88			
CATE	C/T	M1	M0	GATE	C/T	M1	M0	89	TMOD	定时器模式
								8A	TL0	T0 低字节
								8B	TL1	T1 低字节
								8C	TH0	T0 高字节
								8D	TH1	T1 高字节
P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	90	P1*	P1 端口
97	96	95	94	93	92	91	90			
SM0	SM1	SM2	REN	TB8	RB8	TI	RI	98	SCON*	串行口控制
9F	9E	9D	9C	9B	9A	99	98			
								99	SBUF	串行口数据
P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0	A0	P2*	P2 端口
A7	A6	A5	A4	A3	A2	A1	A0			
EA			ES	ET1	EX1	ET0	TX0	A8	IE*	中断允许
AF	—	—	AC	AB	AA	A9	A8			
P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0	B0	P3*	P3 端口
B7	B6	B5	B4	B3	B2	B1	B0			
			PS	PT1	PX1	PT0	PX0	B8	IP*	中断优先权
—	—	—	BC	BB	BA	B9	B8			
CY	AC	F0	RS1	RS0	OV	—	P	D0	PSW*	程序状态字
D7	D6	D5	D4	D3	D2	D1	D0			
								E0	A*	A 累加器
E7	E6	E5	D4	E3	E2	E1	E0			
								F0	B*	B 寄存器
F7	F6	F5	F4	F3	F2	F1	F0			

除表中所列的21个寄存器之外，SFR的其余单元是预留的，不能访问和使用，若访问，则得到的是一个随机数。

21个寄存器中标有位地址的单元有11个（在其英文缩写名处标有*号），它们可以字节寻址，也可以位寻址；其中许多位既有地址码，又有位名称，使用很方便。其余的寄存器单元仅以字节寻址。

① 累加器A（Accumulator） 累加器A有时写成ACC，字节地址E0H，是一个8位寄存器， A表示累加器本身，而ACC侧重于表示累加器对应的地址：E0H。在CPU中，累加器A是工作最频繁的寄存器，在进行算术和逻辑运算时，通常用累加器A存放两个操作数中的一个，而运算结果又存放在累加器A或AB中。

② 寄存器B 寄存器B也是一个8位寄存器，字节地址F0H。一般用于乘、除法指令，它与累加器配合使用。运算前，寄存器B中存放乘数或除数，在乘法或除法完成后用于存放乘积的高8位或除法的余数。寄存器B还有许多类似于累加器A的功能。

③ 程序状态字PSW（Program Status Word） PSW是一个8位寄存器，字节地址D0H，它的各位用于设置当前工作寄存器组、提供位累加器、跟踪程序执行后的状态，并建立有关标志等。PSW中各位状态（除RS1、RS0外）通常是在指令执行过程中自动形成的，也可以由用户根据需要用指令设置，PSW中各位的意义说明如下：

CY	AC	F0	RS1	RS0	OV	X	P

• **CY（PSW.7）**：进/借位标志位，CY 也常写作 C。在执行加法（或减法）运算指令时，如果运算中最高位向前有进位(或借位)，则 CY 位由硬件自动置 1；否则 CY 清 0。CY 也是进行位操作时的位累加器。对于无符号数的加减法可通过 CY 来判断存放在累加器 A 中的结果是否产生溢出，如果 CY=1，反映 A 中的数据已超出了以补码形式表示的一个 8 位无符号数的范围（0～+255）。

• **AC（PSW.6）**：辅助进/借位标志，也称半进位标志。当执行加法（或减法）操作时，如果运算中（和或差）的低半字节（位 3）向高半字节有进位（或借位），则 AC 位将被硬件置 1，否则 AC 被清 0。

• **F0（PSW.5）**：用户标志位。供用户根据自己的需要对 F0 位定义，以作为软件标志，需用指令置位或复位。

• **RS0 和 RS1（PSW.3 和 PSW.4）**：工作寄存器组选择控制位。这两位的值可以决定选择哪组工作寄存器为当前工作寄存器组。上电复位后，RS1=0、RS0=0；CPU 自动选择第 0 组为当前工作寄存器组。用户用指令可以改变 RS1 和 RS0 的值，以切换当前的工作寄存器组，其关系在前面介绍工作寄存器区时已介绍。

• **OV（PSW.2）**：溢出标志位。运算时累加器中内容有溢出，OV 位自动置 1；无溢出时，OV=0。

51 单片机的数据运算通常使用补码，运算结果放回累加器 A。OV=1 反映 A 中的数据已超出了以补码形式表示的一个 8 位有符号数的范围（–128～+127）。

在做加法时，最高、次高二位之一有进位，或做减法时，最高、次高二位之一有借位时 OV 将被置位。如果用 C7 和 C6 分别表示最高和次高二位的进/借位，则有 OV=C7⊕C6。

执行乘法指令也会影响 OV 标志：积>255 时 OV=1，否则 OV=0（OV=0 时，只需要从 A 中取积）。

执行除法指令也会影响 OV 标志：如 B 中所放除数为 0，OV=1，否则 OV=0。

• **X（PSW.1）**：保留位，51 单片机未用，52 子系列单片机有用。

• **P（PSW.0）**：奇偶标志。P 标志跟踪 A 中 1 的奇偶个数，奇为 1，偶为 0。此标志常用于串行通信，通过奇偶校验可以检验数据传输的可靠性。

④ 堆栈指针 SP（Stack Pointer） 堆栈指针 SP 是一个 8 位专用寄存器，字节地址 81H。堆栈的概念：“栈”是存放物资的仓库，如货栈等。在微处理器中设有堆栈，堆栈的主要作用是在处理子程序调用和中断操作等问题时，保存返回地址和保护现场信息。51 单片机中的堆栈是片内 RAM 中的一部分区域，位置不固定，而利用堆栈指针寄存器 SP 指定其位置。

【例 1-1】执行语句 `SP=0x60;`

SP 中的内容 0x60 就是堆栈指针的指向，也就是堆栈区的地址。

51 单片机的堆栈区，以堆栈指针 SP 的初值为界，向地址高的方向伸展。数据出/入堆栈遵循“后进先出”的原则。入栈时，堆栈指针 SP 先加 1，然后数据压入 SP 指向的单元；出栈时，数据先从 SP 指向的单元弹出，堆栈指针 SP 再减 1。

SP 的复位初值=07H，表示 51 单片机默认的堆栈区从片内 RAM 的 07H 单元开始，入栈数据从 08H 单元开始存放。由于片内 RAM 的 08H 单元是工作寄存器 1 组的 R0 寄存器，一般的情况下，这是不合适的。因此，在实际的初始化程序中，要安排指令调整 SP 的初值。