

MCS—48单片计算机

MCS—96单片计算机

应 用 手 册



中国英特尔计算机用户协会

目 录

第一篇 MCS—48 系统	(1)
---------------------	-------

第一章 结构	(1)
--------------	-------

结构	(1)
运算部件	(1)
程序存贮器	(2)
数据存贮器	(3)
输入/输出	(4)
测试和中断输入	(5)
程序计数器和堆栈	(5)
程序状态字	(6)
条件转移逻辑	(7)
中断	(7)
中断定时	(8)
定时器/计数器	(9)
时钟和定时电路	(10)
复位	(10)
单步	(14)
掉电方式	(15)
外部访问方式	(16)
同步方式	(16)
空闲方式	(17)
引脚说明	(18)
编程、检验和 EPROM 擦除	(21)

第二章 MCS—48 扩展系统	(23)
-----------------------	--------

引言	(23)
程序存贮器的扩展	(23)
取指周期	(23)
扩展的程序存贮器的寻址 (超过2K)	(24)

程序存储器的区开关	(24)
中断程序	(24)
I/O 口信息的恢复	(25)
扩展举例	(25)
数据存储器扩展	(26)
读/写周期	(26)
外部数据存储器寻址	(26)
数据存储器扩展举例	(26)
输入/输出扩展	(27)
I/O 扩展器件	(27)
I/O 口特性	(27)
用标准外设扩展I/O	(27)
存储器和I/O扩展器组合	(29)
I/O扩展举例	(29)
多片MCS—48系统	(30)
存储器区开关	(32)
控制信号小结	(32)
口的特性	(32)
BUS口操作	(32)
口2操作	(33)
 第三章 MCS—48指令系统	 (34)
引言	(34)
数据传送	(34)
累加器操作	(35)
寄存器操作	(35)
标志	(35)
转移指令	(35)
子程序	(36)
定时器指令	(36)
控制指令	(36)
输入/输出指令	(37)
指令系统说明	(37)
MCS—48指令系统的符号	(38)
8048AH/8748H/8049AH/8749H指令表摘要	(39)

第四章 ICETM—49A (59)

MCSR—48 在线仿真器.....	(59)
用户系统内的调试功能.....	(60)
成批测试.....	(60)
硬件/软件开发的结合	(60)
实时跟踪.....	(60)
存储器映象.....	(60)
内部存储器.....	(60)
外部存储器.....	(61)
符号调试.....	(61)
硬件.....	(61)
跟踪缓冲器.....	(61)
控制板.....	(61)
仿真板.....	(62)
电缆卡.....	(62)
软件.....	(62)

第五章 MCS—48应用举例..... (64)

引言.....	(64)
硬件例子.....	(64)
I/O扩充技术	(74)
8049H 仿真线路说明.....	(82)
软件例子.....	(86)
单精度二进制乘.....	(88)
中断处理.....	(89)
双字节处理系统.....	(89)
模拟输入/输出	(94)
查表技术.....	(98)

第二篇 MCS—96系统..... (102)

第一章 MCS—96绪论..... (102)

微控制器的继续发展.....	(102)
MCS—96概述.....	(102)

MCS—96的应用	(104)
MCS—96系列开发支持工具	(105)
MCS—96软件开发包	(105)
ASM—96 宏汇编程序	(105)
PL/M—96	(105)
支持硬件开发	(105)
MCS—96调试车间	(105)
Insite 库	(105)
MCS—96产品	(105)
 第二章 结构综述	 (107)
引言	(107)
CPU操作	(107)
CPU总线	(107)
CPU寄存器堆	(108)
RALU 控制	(108)
RALU	(108)
基本定时	(109)
内部定时	(109)
存贮空间	(110)
寄存器堆	(110)
备用存贮空间	(110)
内部 ROM	(111)
存贮控制器	(111)
系统总线	(111)
READY	(112)
RAM 空间	(112)
专用寄存器	(112)
掉电	(115)
中断结构	(116)
中断源	(116)
中断控制	(116)
中断优先级的编程	(117)
中断定时	(118)
定时器	(119)
定时器 1	(119)

定时器 2	(119)
定时器中断	(119)
定时器有关部份	(120)
高速输入	(120)
HSI 模式	(120)
HSI FIFO	(121)
HSI 中断	(121)
HSI 状态	(123)
高速输出	(123)
HSIO 公用引脚	(123)
HSIO CAM	(123)
HSO 状态	(124)
清 HSO	(124)
软件定时器	(124)
模拟量输入	(124)
A/D 精度	(125)
A/D 命令	(125)
A/D 结果	(126)
脉冲宽度可调输出(D/A)	(127)
串行端口	(128)
串行端口模式	(128)
多处理器通讯	(129)
串行端口的控制	(129)
波特率的确定	(129)
I/O 端口 0、1、2、3 和 4	(129)
端口 0	(129)
端口 1	(130)
端口 2	(131)
端口 3 和端口 4	(131)
状态和控制寄存器	(131)
I/O 控制寄存器	(131)
IOC0	(131)
IOC1	(131)
I/O 状态寄存器 0	(131)
I/O 状态寄存器 1	(131)
监控定时器	(132)
禁止监控器	(132)

复位.....	(133)
复位信号.....	(133)
复位状态.....	(133)
复位同步方式.....	(134)
引脚说明.....	(134)
引脚列表.....	(136)
 第三章 MCS—96软件设计.....	(138)
引言.....	(138)
操作数类型.....	(138)
字节类型.....	(138)
字类型.....	(138)
短整类型.....	(139)
整数类型.....	(139)
位类型.....	(139)
双字类型.....	(139)
长整数类型.....	(139)
操作数寻址.....	(139)
寄存器直接寻址.....	(140)
间接寻址.....	(140)
自增型间接寻址.....	(140)
立即寻址.....	(141)
短变址寻址.....	(141)
长变址寻址.....	(141)
ZERO 寄存器寻址.....	(142)
堆栈指针寄存器寻址.....	(142)
汇编语言寻址模式.....	(142)
直接寻址.....	(142)
立即寻址.....	(142)
程序状态字.....	(142)
中断标记.....	(143)
状态标记.....	(143)
指令系统.....	(144)
软件标准和规定.....	(150)
寄存器使用.....	(150)
32位操作数寻址.....	(150)

子程序联接.....	(150)
中断系统运用.....	(151)
全部封锁.....	(151)
挂起中断寄存器.....	(151)
中断屏蔽寄存器.....	(152)
中断向量.....	(152)
关键区段.....	(154)
I/O 编程须知.....	(155)
I/O 口编程.....	(155)
读I/O 状态寄存器1	(156)
向HSO单元发送命令.....	(156)
高速I/O中断	(157)
存取寄存器映象I/O	(157)
例1. 串行I/O通道编程	(158)
例2. 用HSO单元产生 PWM	(161)
例3. 用HSI单元测量脉冲宽度	(166)
例4. 扫描A/D通道.....	(168)
例5. 表格检索和插值.....	(170)
指令系统详述.....	(174)

第四章 硬件设计资料..... (214)

硬件连接综述.....	(214)
硬件的正确连接方法.....	(214)
电源.....	(214)
其它连接.....	(214)
振荡器.....	(215)
复位.....	(216)
同步方式.....	(218)
监控定时器的禁止.....	(218)
掉电路.....	(219)
驱动和接口电平.....	(219)
准双向端口.....	(219)
准双向硬件连接.....	(220)
输入端口, 模拟和数字.....	(221)
漏极开路的端口.....	(221)
HSO引脚, 输出控制和总线引脚.....	(222)

模拟接口·····	(222)
模拟输入·····	(222)
模拟输出·····	(223)
I/O 定时·····	(223)
HSO输出·····	(223)
HSI 输入取样·····	(224)
标准的I/O 端口·····	(224)
串行端口定时·····	(224)
模式 0 ·····	(224)
模式 1 的定时关系·····	(225)
模式 2 和模式 3 的定时关系·····	(226)
总线定时和存贮器的连接·····	(226)
总线功能·····	(226)
定时规范·····	(226)
READY线的使用 ·····	(228)
INST线的使用 ·····	(228)
地址译码·····	(228)
系统查验实例·····	(229)
I/O 端口的重新组织·····	(232)
噪声的抑制·····	(233)
引脚封装和环境条件·····	(234)
 第五章 MCS—96数据表·····	 (235)

第一篇 MCS—48系统

第一章 结 构

本章介绍了MCS—48系列的功能块和各个组件。MCS—48 系列的结构，如图 1—1 所示：

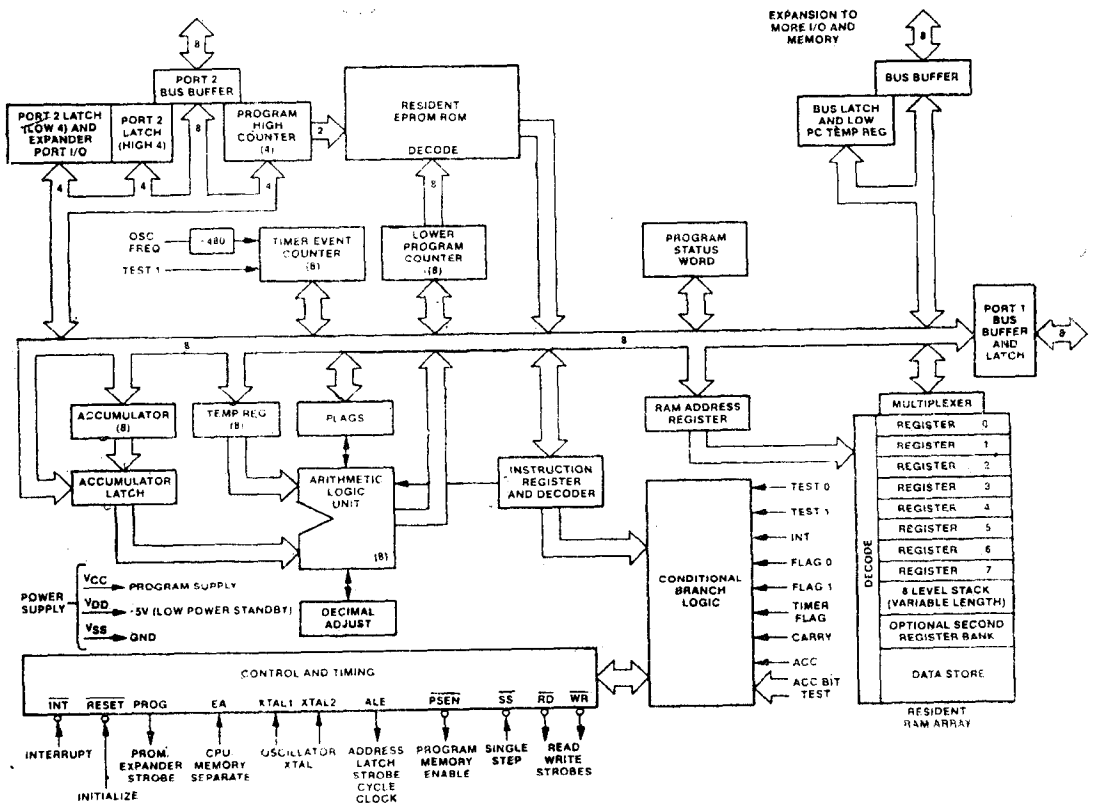


图1—1 8748H/8048H/8749AH/8050AH方块图

运算部件

这个部件具有8048AH的数据处理功能，可分成下列几部分：

- 算术逻辑运算器

- 累加器
- 进位标志
- 指令译码器

标准操作是累加器中的数和通过内部总线来自另一个信号源的数在ALU中相加，并将结果存入累加器或另一个寄存器。

下面分别介绍各个部件的功能：

指令译码器：每个程序指令中的操作码部分（OP Code）存入指令译码器，并转换成输出信号去控制运算器中的各个功能块。这些信号控制数据的源和目的寄存器及ALU所有的执行功能。

算术逻辑运算器：ALU接收来自一个或两个源的8位数字代码，在指令译码器的控制下产生八位结果，ALU可以实现下述功能：

- 代进位或不代进位加
- 与、或、异或
- 加1/减1
- 按位取补
- 左移、右移
- 半字节交换
- BCD、十进制调正

如果ALU操作产生的结果超出8位，就在程序状态字内设置进位标志。

累加器：在处理器中是一个非常重要的数字寄存器，是输入到ALU的一个数字源，并常作为ALU操作结果的目的地。到I/O和存贮器的数据和从I/O口存贮器来的数据一般也通过累加器。

程序存贮器

程序存贮器中包括1024、2048、4096个8位字长的字，由程序计数器确定地址。在8748和8749内的存贮器是可编程和可擦除的EPROM。8049AH、8051AH内的存贮器是屏蔽可编程ROM，8035AHL/8039AHL/8040AHL没有内部程序存贮器，用外部存贮器件。程序码在以上各种型号中是可互换的。要存取8050AH程序存贮器的上2K，或其它MCS—48器件，一个可选择的存贮器单元，必须执行JUMP或CALL指令才可以穿过2K的边界。

在程序存贮器中有三个单元非常重要，如图1—2所示

0单元

启动处理机的复位线，从0号单元取出第一个指令。

3单元

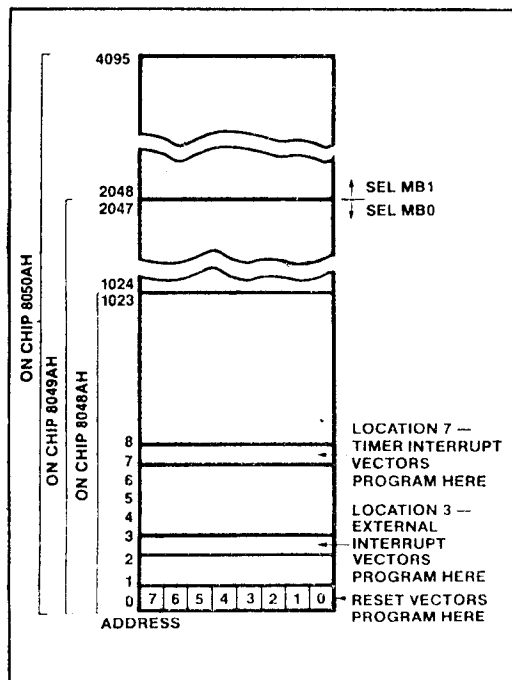
启动处理机中断输入线（如果允许中断），使控制转移到3号单元的子程序。

7单元

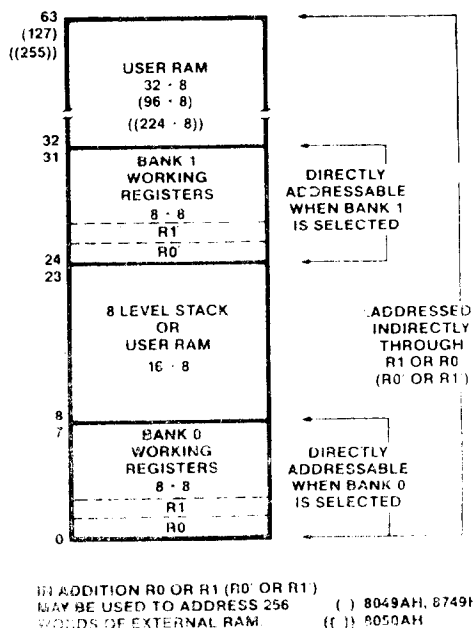
由于定时/计数器溢出导致定时/计数器中断（如允许），使控制转移到7号单元的子程序。

因此复位后执行的第一个指令存在0号单元，外部中断服务子程序的第一个字存在3

号单元，定时/计数器服务子程序的第一个字存在7号单元。程序存储器即能存放程序指令，又能存放常数。这样可使用MOVP和MOVP3很容易的查数据表。



F1—2 程序存储器影像



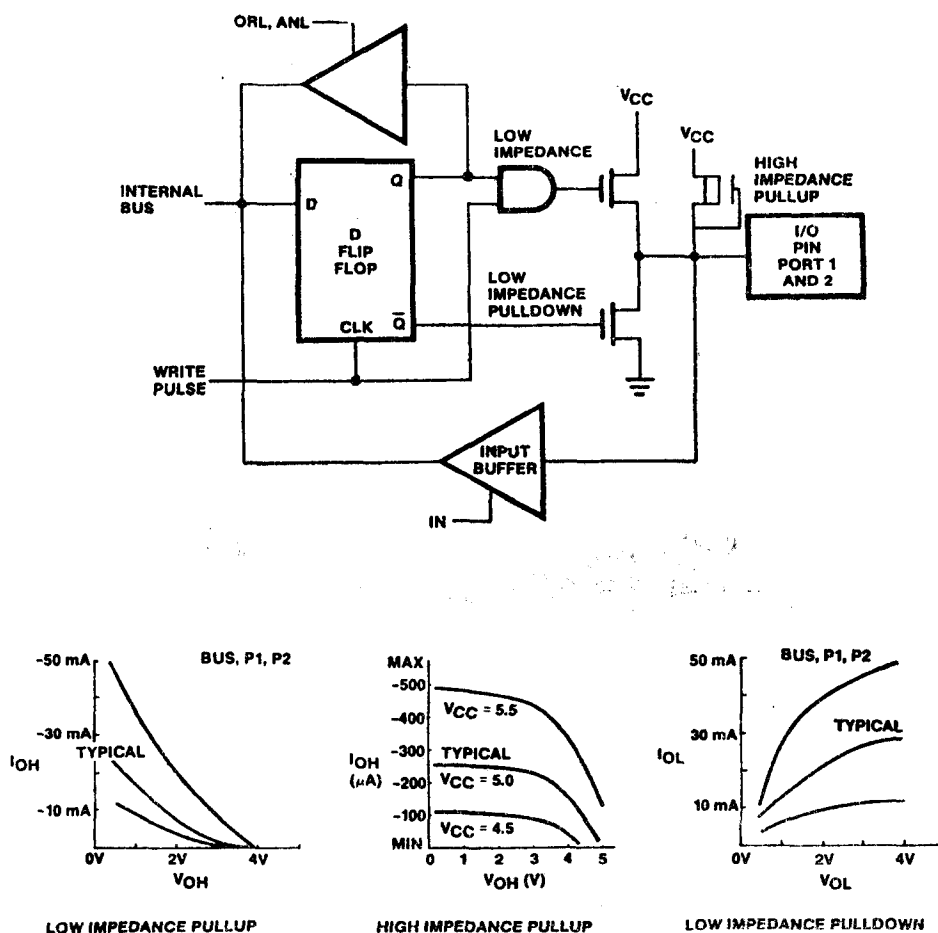
F1—3 数据存储器影像

数据存储器

在8048AH、8049AH和8050AH内数字存储器为64、128、256个字，字长为8位的存储器。所有单元可通过两个RAM指针寄存器中的任一个寄存器间接寻址，它们驻留在地址为0和1的寄存器阵列。如图1-3所示。这个阵列中前八个单元（0—7）叫做工作寄存器，因为这些寄存器很容易寻址，因此可由几个指令直接寻址。常用它们存放经常存取的中间结果。用DTNZ指令允许程序员在单一指令中对寄存器减一，并测试，使工作寄存器当作程序循环计数器得到充分地利用。

通过执行寄存器区开关指令（SELRB）RAM24—31单元（代替0—7单元）工作寄存器可直接寻址。第二个工作寄存器区可以作为第一个区的扩展区或在中断服务程序期间，由区开关保留主程序用的0区寄存器。如果不用第二个区，24—31单元还是和普通的RAM一样寻址。因为两个RAM指针R₀和R₁寄存器是工作寄存器阵列的一部分，所以区开关有效的产生另外两个寄存器指针（R₀/和R₁/），这样加上R₀、R₁就容易地同时存取RAM的四个分离的工作区。RAM的8—23号单元具有两重作用，包括用作程序计数器堆栈。这些单元在子程序期间可以用栈指针寻址，也可用RAM指针寄存器R₀和R₁寻址。如果子程

序嵌套少于八层，就不需要这些栈寄存器，可当作一般的RAM单元使用。每少一层子程序嵌套，可多两个RAM单元。



These graphs are for informational purposes only and are not guaranteed minimums or maximums.

图1—4 “准双向口” 结构

输入/输出

8048有27条线，可以用作输入或输出线。这些线分成输入、输出和双向口三个口，每个口8条线，三条“测试”输入线用条件转移指令测试时可以改变程序的顺序。

端口1和端口2：

端口1和端口2的每个口具有八位字和相同的特性，写到这些口的数据被锁存，直到重新写为止。做为输入口这些线没有锁存，即输入必须保持到由输入指令读入为止。输入与TTL电平相容，输出能驱动一标准TTL负载。

端口1和端口2这些线称为准双向线，因为具有特殊的输出电路结构，尽管输出被锁存起来，但每个线可以做输入、输出或两个同时存在。图1—4详细地表示出电路的结构，每条线通过高阻抗电阻上拉到V_{cc}，为TTL高电平提供足够的电流源；也可以用一个标准的TTL门将其下拉，这样可允许同一脚即可输入，又可输出。为了提高“0”到“1”的开关速度，当写“1”到这条线时，就暂时（ $\approx \frac{1}{2}$ 机器周期）接入一个比较低的电阻，当写“0”到这条线时，一个低电阻克服这个轻微的上拉高电阻并提高TTL吸收电流能力。因为下拉低电阻是低阻抗器件，所以任一条线作为输入时必须首先写“1”，复位使所有线初始化为高阻“1”状态。这种结构可在同一引脚实现输入和输出，同时也允许同一口有输入和输出线。这种准双向口和ANL及ORL逻辑指令为在一个8位处理器中处理单线输入和输出提供了有效的方法。一个口的任何输出操作（ORL、ANL或MOV）将驱动该口所有引脚上的低阻抗器件。如果最终结果是高阻抗（“1”）状态，就会暂时接通下拉器件。这也适用于同一口内输入线与输出线混合的情况。ORL和ANL是读/写操作，注意到这一点是重要的。执行它们时微机“读”这个口，根据该指令修改数据，然后把数据“写”回这个口。“写”（实质是OUTL指令）时重新使低电阻暂时上拉，尽管数据仍是一个“1”不变。这专门适用于一个口的输入和输出混合的情况。

总线

总线也是双向八位口，具有相应的输入和输出选通脉冲。如果没有用双向口的必要，总线可以作静态锁存的输出口或没有锁存的输入口，输入和输出线在这个口不能混合用。

作为静态口数据的写和锁存用OUTL指令，输入用INS指令。INS和OUTL指令在相应的RD和WR输出线上产生选通脉冲，然而用静态口方式时一般不使用这两个指令。双向口用MOVX指令来读出或写入这个口。写入这个口时在WR输出线产生脉冲，在WR的后沿输出数据有效。读出时在RD输出线产生脉冲，输入数据在RD的后沿有效。总线在不写入和读出时是高阻状态。

测试和中断输入

有三个脚用作输入和用条件转移指令来测试，这就是T₀、T₁和INT。这些口允许转移到程序分支。这些口的内容不需要读到累加器。T₀、T₁和INT脚还有其它功能。

程序计数器和堆栈

程序计数器是独立的计数器、程序计数器栈是由数据存储器阵列中的寄存器对实现的。程序计数器在8048AH中为10位、8049AH为11位，8050为12位、可以分别对片内程序存储器的1024、2048、或4096字寻址。最高位用于外部程序存储器（看图1-5），复位时程序计数器初始为零。

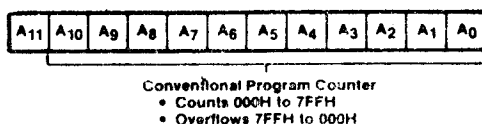


图1—5 程序计数器

中断或 CALL 子程序时, 程序计数器的内容存放到由八个寄存器对组成的程序计数器栈内, 如图1-6所示。使用的寄存器对由三位栈指针决定, 它们是程序状态字(PSW)的一部分。数字RAM的8—23单元用作栈寄存器存储程序计数器和 PSW 的高四位、如图1-6所示。栈指针初始化为000时指向RAM 的8和9单元。第一个子程序转移或中断使程序计数器内容存放到 RAM 阵列中的8和9单元。执行另一个CALL栈指针就加1指向10和11单元。子程序内嵌套子程序可以达到8层, 栈不会溢出。如溢出, 则最下面的单元(8和9单元)将被重新写且原内容丢失, 因为栈指针从111溢出为000, 也可以从000下溢出为111。

由返回指令(RET或RETR)发信号给子程序的结尾, 引起栈指针减1并使结果寄存器对的内容传送到程序计数器中。

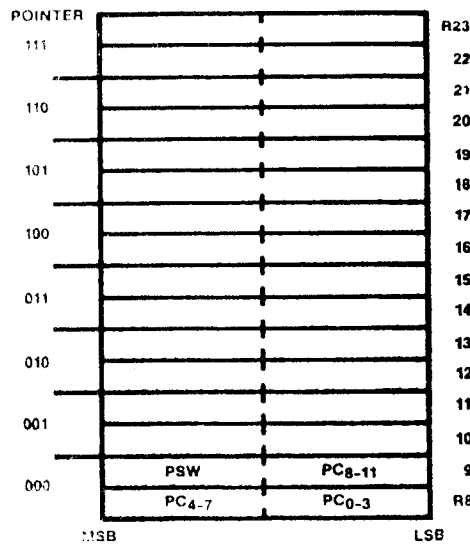


图1—6 程序计数器堆栈

程序状态字

8位状态字可以从累加器装入、移出。这8位状态字叫做程序状态字(PSW)。图1—7表示字内的信息。程序状态字实际上是整个机器触发器的集合, 可以作为一个整体读和写。写入PSW的能力允许机器掉电后容易恢复机器的状态。

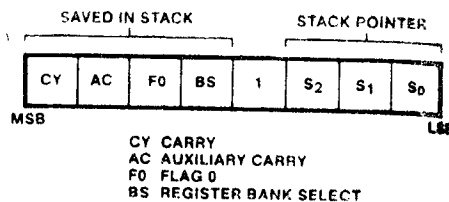


图1—7

当调用子程序或中断向量时PSW的高四位存入程序计数器栈中，并能用 RETR 指令来存贮。RET返回指令不更新PSW。

PSW各位的意义如下：

0~2位：栈指针 (S_0 、 S_1 、 S_2)

3位：不用（读时为“1”）

4位：工作寄存器区开关位

0 = 0区

1 = 1区

5位：标志位0 (F_0) 可由用户控制。可被清0或取反，并能用条件转移指令 JFO 测试。

6位：辅助进位 (AC) 由ADD指令产生的进位位，供十进制调整指令DAA使用。

7位：进位标志，它标志前面操作产生累加器溢出。

条件转移逻辑

微处理机的条件转移逻辑可用用户程序选通几种要测试的内部或外部条件。根据表1—1的条件，用条件转移指令可以改变程序执行的顺序。

表1—1转移条件

被 测 试 器 件	转 移 条 件	
累加器	全○	非全○
累加器位	—	1
进位标志	○	1
用户标志 (F_0 , F_1)	—	1
计时器溢出标志	—	1
测试输入 (T_0 , T_1)	○	1
中断输入 (INT)	○	—

中 断

在INT脚上加一个低电平“0”启动中断序列。中断为电平触发并且低电平有效，以允许几个中断源在输入脚上“线或”（“WIRE ORING”）图1—8是8048AH中断逻辑图。在每个程序周期取样这个中断线。当测试到输入时，在完成当前指令的所有周期后立即转移到程序存储器3号单元的子程序。对于两个周期的指令，只在第二个周期内中断线取样。INT 必保持三个机器周期的低电位，以保证正常的中断操作。在任何子程序调用中，程序计数器的内容和程序状态字都存入堆栈，有关这个操作的说明请参见前面的程序计数

器和堆栈部分。程序存储器的3号单元通常存一条无条件转移指令。它转到存在程序存储器其它地方的中断服务子程序。中断服务子程序的结尾是执行返回和恢复状态指令RETR。这个中断系统为单级中断,当允许一个中断后,就不允许其它中断只有执行RETR指令后,才重新允许中断逻辑输入。这发生在RETR指令第二周期的开始。这同样适用于由于定时器溢出产生的内部中断。如果同时检测到内部定时/计数器产生的中断及外部中断,就执行外部中断。如果需要,可以引入第二个外部中断,方法是允许定时器/计数器中断,在计数器内装入FFH(比溢出少1)及启动事件计数器方式。T₁脚从“1”跳变到“0”引起转移到7号单元的中断向量。

中断定时

这个中断输入在程序控制下可用ENT和DIS指令来允许或关闭。在复位时关中断,一直保持到用户程序允许(开中)为止。在从服务程序返回执行RETR指令前,必须移去中

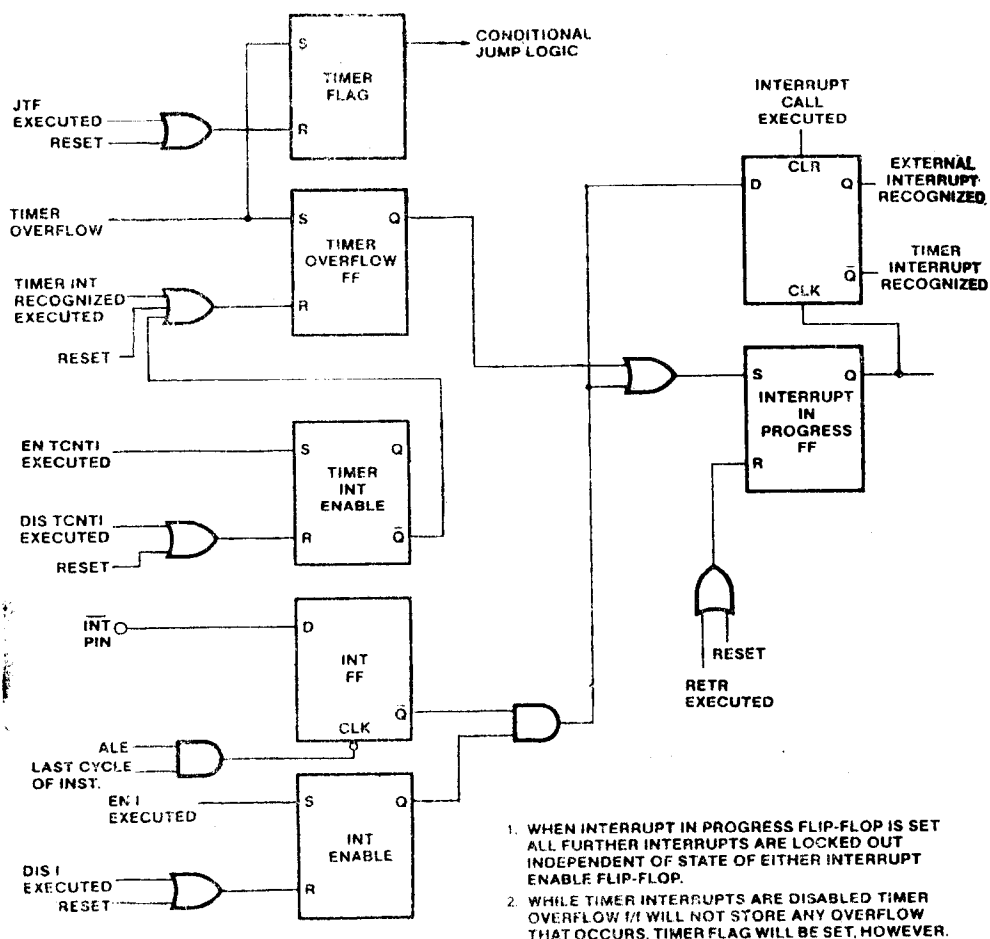


图1—8 中断逻辑