

盛群知识产权政策

专利权

盛群半导体公司在全球各地区已核准和申请中之专利权至少有 160 件以上，享有绝对之合法权益。与盛群公司 MCU 或其它产品有关的专利权并未被同意授权使用，任何经由不当手段侵害盛群公司专利权之公司、组织或个人，盛群将采取一切可能的法律行动，遏止侵权者不当的侵权行为，并追讨盛群公司因侵权行为所受之损失、或侵权者所得之不法利益。

商标权

盛群之名称和标识、Holtek 标识、HT-IDE、HT-ICE、Marvel Speech、 Music Micro、 Adlib Micro、 Magic Voice、 Green Dialer、 PagerPro、 Q-Voice、 Turbo Voice、 EasyVoice 和 HandyWriter 都是盛群半导体公司在台湾地区和其它国家的注册商标。

著作权

Copyright © 2007 by HOLTEK SEMICONDUCTOR INC.

规格书中所出现的信息在出版当时相信是正确的，然而盛群对于规格内容的使用不负责任。文中提到的应用其目的仅仅是用来做说明，盛群不保证或不表示这些应用没有更深入的修改就能适用，也不推荐它的产品使用在会由于故障或其它原因可能会对人身造成危害的地方。盛群产品不授权使用于救生、维生器件或系统中做为关键器件。盛群拥有不事先通知而修改产品的权利，对于最新的信息，请参考我们的网址 <http://www.holtek.com.tw>; <http://www.holtek.com.cn>

技术相关信息

- [工具信息](#)
- [问答集](#)
- [应用范例](#)

特性

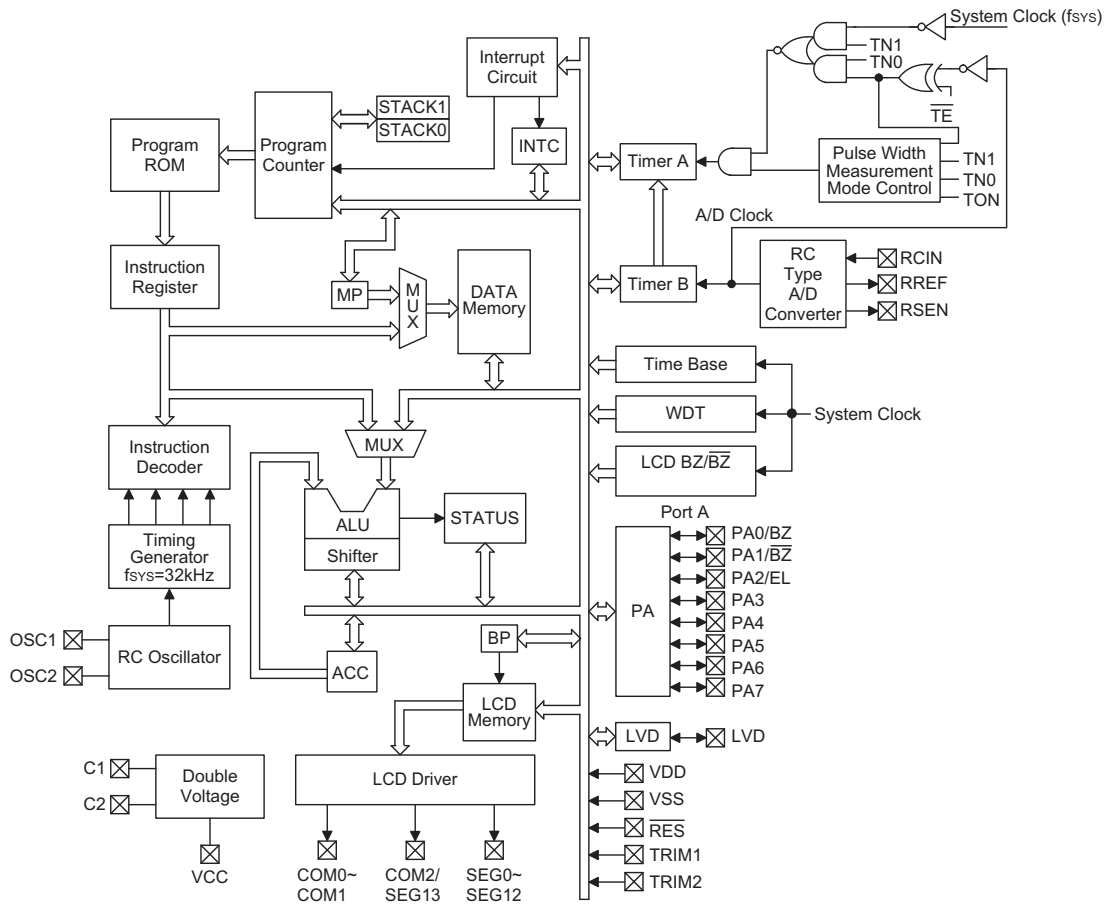
- 工作电压：1.2V~2.2V
- 8 个双向输入/输出口
- 内置 32kHz~128kHz RC 振荡(外接电阻)
- RC 型 A/D 转换
- 看门狗定时器
- 1K×16 程序存储器 ROM
- 32×8 数据存储器 RAM
- 时基 (TB)
- 蜂鸣器输出(BZ, $\overline{\text{BZ}}$)
- EL 载波输出
- 一个外部可调低电压检测器
- 13×3 或 14×2 段的液晶显示驱动电路
- HALT 和唤醒功能可降低功耗
- 2 层硬件堆栈
- 位操作指令
- 查表指令，表格内容字长 16 位
- 当系统时钟为 128kHz 时，指令周期为 31μs
- 指令执行时间为 1 或 2 个指令周期
- 63 条指令
- 44-pin QFP 封装

概述

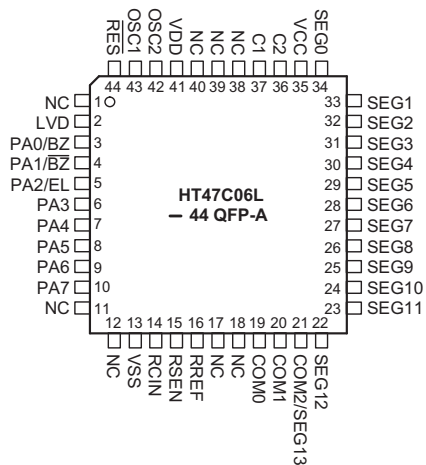
HT47C06L 是 8 位高性能精简指令集单片机。专门为有模拟信号输入（如传感器信号）的产品而设计。单一周期指令和流水线结构使它适用于高速应用。

低功耗、I/O 使用灵活、计数器、振荡类型选择、RC 型 A/D 转换器、LCD 驱动、暂停和唤醒功能，使这款单片机可以广泛应用于电阻到频率转换的 A/D 测量，例如传感器测量、遥控测量，特别是当作临床体温计单片机使用。

方框



引脚图



引脚说明

引脚名称	输入/输出	功能说明
RES	输入	斯密特触发复位输入，低电平有效。
PA0/BZ PA1/ \overline{BZ}	输入/输出 输入/输出	2 位双向输入/输出口，每个位都具有唤醒输入。PA0 和 PA1 分别与 BZ 和 \overline{BZ} 公用引脚。一旦 PA0 和 PA1 选择为蜂鸣器输出，那么输出信号来自内部蜂鸣器时钟发生器。软件指令决定引脚是 CMOS 输出或斯密特触发输入，可由掩膜选项决定是否有上拉电阻。
PA2/EL	输入/输出	1 位双向输入/输出口，具有唤醒输入。PA2 和 EL 载波输出公用引脚。一旦 PA2 选为 EL 载波输出，那么输出信号来自内部 EL 载波时钟发生器。软件指令决定引脚是 CMOS 输出或斯密特触发输入，可由掩膜选项决定是否有上拉电阻。
PA3~ PA7	输入/输出	5 位双向输入/输出口，每个位都具有唤醒输入。软件指令决定引脚是 CMOS 输出或斯密特触发输入，可由掩膜选项决定是否有上拉电阻。
VSS	—	电源负极，地。
VCC,C1,C2	—	倍压电路， $VCC=2\times VDD$ VCC: LCD 工作电压，需在 VCC 与 VSS 之间接入一个电容。 C1 ,C2: VCC 的转换引脚，需在 C1 与 C2 之间接入一个电容。
COM0~COM1 COM2/SEG13	输出	LCD 的 1/3 占空比的掩膜选项决定 COM2/SEG13 引脚是作为 LCD 面板的 SEG 13 驱动输出还是作为 COM2 驱动输出。COM0~COM1 是 LCD 的 common 输出。
SEG0~SEG12	输出	LCD 面板 Segment 驱动输出。
VDD	—	电源正极。
LVD	—	低电压检测，需在 VSS 与 LVD 之间接入一个电阻。
RCIN	输入	RC 型 A/D 转换 RC 振荡输入引脚。
RREF	输出	RC 型 A/D 转换参考电阻连接引脚。
RSEN	输出	RC 型 A/D 转换传感器电阻连接引脚。
OSC1 OSC2	—	系统振荡输入引脚，需在 OSC1 与 OSC2 之间接入一个电阻。
TRIM1~TRIM2	输入	测试输入引脚，正常使用时不必连接。

极限参数

电源供应电压..... $V_{SS} -0.3V \sim V_{SS} +2.5V$
 端口输入电压..... $V_{SS} -0.3V \sim V_{DD} +0.3V$
 端口总灌电流150mA
 总功耗 500mW

储存温度..... $-50^{\circ}C \sim 125^{\circ}C$
 工作温度..... $-40^{\circ}C \sim 85^{\circ}C$
 端口总源电流-100mA

注：这里只强调额定功率，超过极限参数所规定的范围将对芯片造成损害，无法预期芯片在上述标示范围外的工作状态，而且若长期在标示范围外的条件下工作，可能影响芯片的可靠性。

直流电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	1.2	1.5	2.2	V
V _{CC}	LCD 电压	—	V _{CC} =2×V _{DD}	2.4	3	4.4	V
V _{LVD}	低电压检测电压	—	*R _{LVD} =30kΩ	1.25	1.3	1.35	V
I _{DD}	工作电流	1.5V	无负载, f _{sys} =32kHz, A/D 关闭, LVD 关闭	—	10	20	μA
			无负载, f _{sys} =32kHz, A/D 打开, LVD 关闭 *R=30kΩ, *C=2200pF	—	30	60	μA
			无负载, f _{sys} =128kHz, A/D 关闭, LVD 关闭	—	15	30	μA
			无负载, f _{sys} =128kHz, A/D 打开, LVD 关闭 *R=30kΩ, *C=2200pF	—	35	70	μA
I _{LVD}	LVD 电流	1.5V	LVD 打开	—	50	100	μA
I _{STB1}	静态电流 (LVD 关闭, LCD 关闭)	1.5V	无负载, 系统 HALT, A/D 关闭, LVD 关闭	—	—	1	μA
I _{STB2}	静态电流 (LCD 打开)	1.5V	无负载, f _{sys} =32kHz, A/D 关闭, LVD 关闭	—	5	10	μA
			无负载, f _{sys} =128kHz, A/D 关闭, LVD 关闭	—	8	16	μA
V _{IL1}	输入/输出口的低电平输入电压	—	—	0	—	0.3V _{DD}	V
V _{IH1}	输入/输出口的高电平输入电压	—	—	0.8V _{DD}	—	V _{DD}	V
V _{IL2}	低电平输入电压($\overline{\text{RES}}$)	—	—	0	—	0.4V _{DD}	V
V _{IH2}	高电平输入电压($\overline{\text{RES}}$)	—	—	0.9V _{DD}	—	V _{DD}	V
I _{OL1}	输入/输出口灌电流 (PA0/BZ, PA1/ $\overline{\text{BZ}}$, PA2/EL, PA3~PA7)	1.5V	V _{OL} =0.15V	0.5	0.8	—	mA
I _{OH1}	输入/输出口源电流 (PA0/BZ, PA1/ $\overline{\text{BZ}}$, PA2/EL, PA3~PA7)	1.5V	V _{OH} =1.35V	-0.3	-0.6	—	mA
I _{OL2}	输入/输出口灌电流 (RREF, RSEN)	1.5V	V _{RREF, RSEN} =0.15V	4	7	—	mA
I _{OH2}	输入/输出口源电流 (RREF, RSEN)	1.5V	V _{RREF, RSEN} =1.35V	-3	-5	—	mA
I _{OL3}	Common 灌电流	1.5V	V _{OL} =0.3V(1/2bias)	50	100	—	μA
I _{OH3}	Common 源电流	1.5V	V _{OH} =2.7V(1/2bias)	-50	-100	—	μA
I _{OL4}	Segment 灌电流	1.5V	V _{OL} =0.3V(1/2bias)	50	100	—	μA
I _{OH4}	Segment 源电流	1.5V	V _{OH} =2.7V(1/2bias)	-50	-100	—	μA
R _{PH}	上拉电阻	1.5V	V _{IL} =0V	75	150	300	kΩ

注: *R 代表 RC 型 A/D 转换器的电阻

*C 代表 RC 型 A/D 转换器的电容

*R_{LVD} 值对于不同的批次可能会有所不同

交流电气特性

Ta=25°C

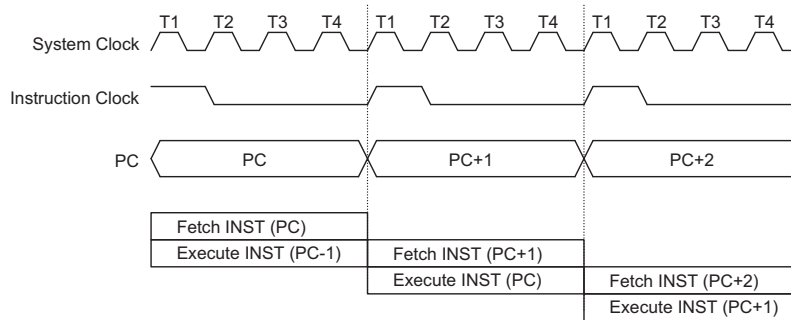
符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
f _{SYS}	系统时钟	1.5V	外接电阻	25	—	154	kHz
t _{RES}	外部复位低电平脉宽	1.5V	—	100	—	—	μs
f _{AD}	A/D 转换频率	1.5V	—	—	—	50	kHz

系统功能说明

指令执行时序

HT47C06L 的系统时钟由需要外接电阻的内置 RC 振荡器产生。该时钟在芯片内部被分成四个互不重叠的时钟周期。一个指令周期包括四个系统时钟周期。

指令的读取和执行是以流水线方式进行的，这种方式在一个指令周期进行读取指令操作，而在下一个指令周期进行解码与执行该指令。因此，流水线方式使多数指令能在一个周期内执行完成。但如果涉及到的指令要改变程序计数器的值，就需要花两个指令周期来完成这一条指令。



指令执行时序

程序计数器 — PC

10 位的程序计数器(PC)控制程序存储器 ROM 中指令执行的顺序，它可寻址整个 ROM 范围的 1024 个地址。

取得指令码以后，程序计数器会自动加一，指向下一个指令码的地址。但如果执行跳转、条件跳跃、向 PCL 赋值、子程序调用、初始化复位、内部中断、外部中断、子程序返回等操作时，PC 会载入与指令相关的地址而非下一条指令地址。

当遇到条件跳跃指令且符合条件时，当前指令执行过程中读取的下一条指令会被丢弃，取而代之的是一个空指令周期，随后才能取得正确的指令。反之，就会顺序执行下一条指令。

程序计数器的低字节(PCL)是一个可读写的寄存器(06H)。对 PCL 赋值将产生一个短跳转动作，跳转的范围为当前页 256 个地址。

当遇到控制转移指令时，系统也会插入一个空指令周期。

模式	程序计数器									
	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
初始化复位	0	0	0	0	0	0	0	0	0	0
定时/计数器中断	0	0	0	0	0	0	0	1	0	0
时基中断	0	0	0	0	0	0	1	0	0	0
条件跳跃	PC+2									
装载 PCL	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
跳转、子程序调用	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
从子程序返回	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

注： *9 ~ *0 : 程序计数器位
S9~ S0 : 堆栈寄存器位

#9~ #0 : 指令代码位
@7 ~ @0 : PCL 位

程序存储器

程序存储器用来存放要执行的指令代码，以及一些数据、表格和中断入口。程序存储器有 1024×16 位，程序存储器空间可以用程序计数器或表格指针进行寻址。以下列出的程序存储器地址是系统专为特殊用途而保留的：

- 地址 000H

该地址为程序初始化保留。系统复位后，程序总是从 000H 开始执行。

- 地址 004H

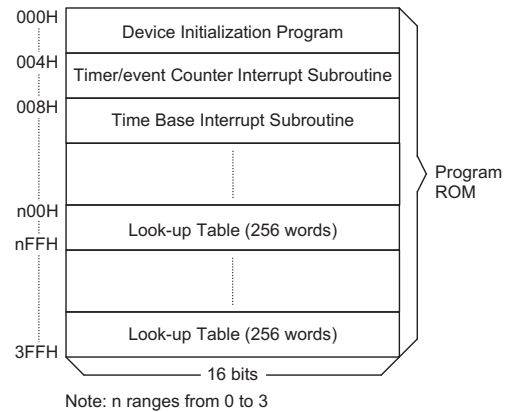
该地址为定时/计数器中断服务程序保留。当定时/计数器 A 或 B 发生溢出，如果中断允许且堆栈未满，则程序会跳转到 004H 地址开始执行。

- 地址 008H

该地址为时基中断服务程序保留。当时基发生溢出，如果中断允许且堆栈未满，则程序会跳转到 008H 地址开始执行。

- 表格区

程序存储器空间的任何地址都可做为查表使用。查表指令“TABRDC [m]”(查当前页表格，1 页=256 个字)和“TABRDL [m]”(查最后页表格)，会把表格内容低字节传送给 [m]，而表格内容高字节传送到 TBLH 寄存器(08H)。只有表格内容的低字节被传送到目标地址中，而高字节被传送到表格内容高字节寄存器 TBLH。表格内容高字节寄存器 TBLH 是只读寄存器。表格指针(TBLP)是可读/写寄存器(07H)，用来指明表格地址。在查表之前，要先将表格地址写入 TBLP 中。TBLH 寄存器只可读，不可写。如果主程序和中断服务程序(ISR)都用到查表指令，主程序中 TBLH 的值可能会因为 ISR 中执行的查表指令而发生变化，产生错误。也就是说，要避免在主程序和中断服务程序中都使用查表指令。但如果必须这样做的话，我们可以在查表指令前先将中断禁止，在保存了 TBLH 的值后再开放中断以避免发生错误。所有与表格有关的指令都需要两个指令周期的执行时间。这里提到的表格区都可以做为正常的程序存储器来使用。



程序存储器

指令	表格区									
	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	@7	@6	@5	@4	@3	@2	@1	@0

表格区

注：*9~*0：表格地址位

P9~P8：当前程序指针位

@7~@0：表格指针位

堆栈寄存器 — STACK

堆栈寄存器是特殊的存储器空间，用来保存 PC 的值。HT47C06L 有 2 层堆栈，堆栈寄存器既不是数据存储器的一部分，也不是程序存储器的一部分，而且它既不能读出，也不能写入。堆栈的使用是通过堆栈指针(SP)来实现的，堆栈指针也不能读出或写入。当发生子程序调用或中断响应时，程序计数器(PC)的值会被压入堆栈；在子程序调用结束或中断响应结束时(执行指令 RET 或 RETI)，堆栈将原先压入堆栈的内容弹出，重新装入程序计数器中。在系统复位后，堆栈指针会指向堆栈顶部。

如果堆栈已满，并且发生了不可屏蔽的中断，那么只有中断请求标志会被记录下来，而中断响应会被抑制，直到堆栈指针(执行 RET 或 RETI 指令)发生递减，中断才会被响应。这个功能可以防止堆栈溢出，使得程序员易于使用这种结构。同样，如果堆栈已满，并且发生了子程序“CALL”调用，那么堆栈会发生溢出，首先进入堆栈的内容将会丢失，只有最后的 2 个返回地址会被保留。

数据存储器 — RAM

数据存储器由 52×8 位组成，分为两个功能区间：特殊功能寄存器和通用数据存储器(32×8)，数据存储器单元大多数是可读/写的，但有些只读的。特殊功能寄存器包括间接寻址寄存器 0(00H)，间接寻址指针寄存器 0(MP0; 01H)，间接寻址寄存器 1(02H)，间接寻址指针寄存器 1(MP1; 03H)，存储器段指针寄存器(BP; 04H)，累加器(ACC; 05H)，程序计数器低字节寄存器(PCL; 06H)，表格指针寄存器(TBLP; 07H)，表格内容高字节寄存器(TBLH; 08H)，时基控制寄存器(TBC; 09H)，状态标志寄存器(STATUS; 0AH)，中断控制寄存器 0(INTC; 0BH)，输入/输出寄存器(PA; 12H)，输入/输出控制寄存器 (PAC; 13H)，定时/计数器 A 高、低位字节寄存器(TMRAH; 20H, TMRAL; 21H)，定时/计数器控制寄存器(TMRC; 22H)。定时/计数器 B 高、低位字节寄存器(TMRBH; 23H, TMRBL; 24H)，RC 振荡型 A/D 转换控制寄存器(ADCR; 25H)。

其余在 60H 之前的空间保留给系统以后扩展使用，读取这些地址的返回值为“00H”。通用数据寄存器地址从 60H 到 7FH，用来存储数据和控制信息。

所有的数据存储器单元都能直接执行算术、逻辑、递增、递减和循环操作。除了一些特殊位外，数据存储器的每一位都可通过“SET[m].i”置位或由“CLR[m].i”复位。而且都可以通过间接寻址指针 (MP0; 01H, MP1; 03H) 进行间接寻址。

间接寻址寄存器

地址 00H 和 02H 是间接寻址寄存器，并无实际的物理区存在。任何对[00H]和[02H]的读/写操作，都是访问由 MP0(01H)和 MP1(03H)所指向的 RAM 单元。间接读取 00H 或 02H 地址得到的值为 00H，间接写入此地址，不会产生任何操作。

间接寻址寄存器之间不支持数据传送功能。间接寻址指针 MP0 和 MP1 是 8 位寄存器。MP0 只能用于数据存储器，而 MP1 能用于数据存储器 and LCD 显示存储器。

累加器 — ACC

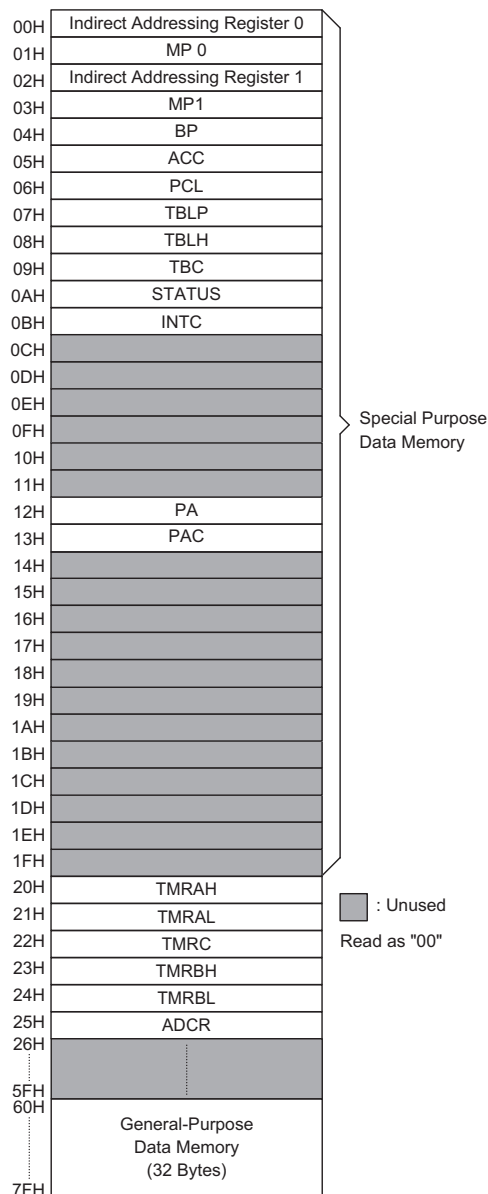
累加器(ACC)与算术逻辑单元(ALU)有密切关系。它对应于 RAM 地址 05H，做为运算的立即数据。存储器之间的数据传送必须经过累加器。

算术逻辑单元 — ALU

算术逻辑单元(ALU)是执行 8 位算术、逻辑运算的电路，它提供有以下功能：

- 算术运算(ADD, ADC, SUB, SBC, DAA)
- 逻辑运算(AND, OR, XOR, CPL)
- 移位运算(RL, RR, RLC, RRC)
- 递增和递减(INC, DEC)
- 分支判断(SZ, SNZ, SIZ, SDZ...)

ALU 不仅可以储存数据运算的结果，还会改变状态寄存器的值。



数据存储器

状态寄存器 — STATUS

8 位的状态寄存器(0AH)，由零标志位(Z)、进位标志位(C)、辅助进位标志位(AC)、溢出标志位(OV)、暂停标志位(PDF)和看门狗定时器溢出标志位(TO)组成。该寄存器不仅记录状态信息，而且还控制操作顺序。

位	符号	功能
0	C	如果在加法运算中结果产生了进位或在减法运算中结果不产生借位，则 C 被置位；反之，C 被清除。它也可被循环移位指令影响。
1	AC	如果在加法运算中低 4 位产生了进位或减法运算中低 4 位不产生借位，则 AC 被置位；反之，AC 被清除。
2	Z	如果算术或逻辑运算的结果为零，则 Z 被置位；反之，Z 被清除。
3	OV	如果运算结果向最高位进位，但最高位并不产生进位输出，则 OV 被置位，反之亦然；否则，OV 被清除
4	PDF	系统上电或执行“CLR WDT”指令，PDF 被清除；执行“HALT”指令，PDF 被置位。
5	TO	系统上电、执行“CLR WDT”或“HALT”指令，TO 被清除；WDT 定时溢出，TO 被置位。
6~7	—	未用，读数为“0”

状态寄存器

除了 PDF 和 TO 标志外，状态寄存器的其它位都可以用指令改变。任何对状态寄存器的写操作都不会改变 PDF 和 TO 的值。对状态寄存器的操作可能会导致与预期不一样的结果。TO 标志只受系统上电、看门狗溢出、“CLR WDT”指令或“HALT”指令的影响。PDF 标志只受系统上电、“CLR WDT”指令或“HALT”指令的影响。

标志位 Z、OV、AC 和 C 反映的是最近一次操作的状态。

在进入中断程序或子程序调用时，状态寄存器不会被自动压入堆栈。如果状态寄存器的内容是重要的，而且子程序会影响状态寄存器的内容，那么程序员必须事先将 STATUS 的值保存好。

中断

HT47C06L 提供了一个内部定时器/计数器中断和一个内部时基中断。中断控制寄存器(INTC; 0BH)包含了中断控制位，用来控制中断允许/禁止以及中断请求标志。

位	标志	功能
0	EMI	控制总中断(1=允许；0=禁止)
1	ETI	控制定时/计数器中断(1=允许；0=禁止)
2	ETBI	控制时基中断(1=允许；0=禁止)
3	—	未定义，读取值为“0”
4	TF	定时/计数器中断请求标志(1=请求；0=无)
5	TBF	时基中断请求标志(1=请求；0=无)
6~7	—	未定义，读取值为“0”

INTC (0BH) 寄存器

一旦有中断子程序被服务，其余的中断全部被禁止(通过清除 EMI 位)，这种做法的目的在于防止中断嵌套。在执行中断服务子程序期间，可能会发生其它的中断请求，但只有中断请求标志会被记录下来。如果要实现中断嵌套，使用者可以在中断子程序中置位 EMI 及 INTC 中对应的中断控制位。如果堆栈已满，即使有关的中断允许，该中断申请并不会被响应，一直到堆栈指针(SP)发生递减才会响应。如果需要立即服务，应避免让堆栈发生饱和。

当有中断服务请求，会将程序计数器值压栈，然后再转至中断服务程序的入口。但要记住这时只有程序计数器的内容被压栈，如果寄存器和状态寄存器的内容被中断程序改变，而导致该中断服务程序会破坏预期控制序列的话，使用者应该事先将这些数据储存起来。

内部定时/计数器中断是由置位定时/计数器中断申请标志(TF; INTC 的第 4 位)的方式来启动的, 而该中断申请标志是由定时器 A 或定时器 B 溢出产生的。如果中断允许, 而此时堆栈尚未用满, 而且 TF 也已置位的话, 会调用地址 04H 子程序。此时不仅其相关的中断申请标志(TF)会被清除, 而且 EMI 位也会被清除, 以禁止其他中断。

时基中断是由置位时基中断申请标志位(TBF; INTC 的第 5 位)的方式来启动的, 而该中断申请标志是由时基时钟产生的。如果中断允许, 而此时堆栈尚未用满, 而且 TBF 也已置位的话, 会调用地址 08H 子程序。此时不仅其相关的中断申请标志(TBF)会被清除, 而且 EMI 位也会被清除, 用以禁止任何进一步的中断。

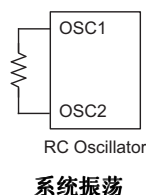
执行中断子程序期间, 其他的中断响应会被暂停, 直到执行 RETI 指令或是将 EMI 位和其相关中断控制位设为“1”(此时堆栈尚未用满)为止。若欲从此中断子程序返回, 只要执行 RET 或 RETI 指令即可。其中, RETI 会设定 EMI 位, 用以允许中断服务, 而 RET 则否。

如果在两个连续的 T2 脉冲上升沿之间发生中断, 则在这两个 T2 后面的 T2 脉冲该中断会被服务。而如果同时发生中断申请, 其顺序会依照下表所显示, 这个顺序可以通过设定各中断相关的控制位来改变顺序。

中断源	优先级	中断向量
定时/计数器中断	1	04H
时基中断	2	08H

振荡电路

HT47C06L 只有一个外部 RC 振荡器, 通过外部 RC 产生系统时钟。HALT 模式停止系统振荡器并忽略外部信号以节省电源。如果使用 RC 振荡, 那么需在 OSC1 与 OSC2 之间接一个电阻。RC 振荡器提供了最廉价的解决方案, 然而振荡器的频率会随着 VDD, 温度和芯片自身的参数发生变化。因此, 它不适合需要精确的振荡器频率的计时操作的场合。



看门狗定时器 — WDT

WDT 的时钟来源(f_s)是系统时钟 f_{SYS} 。WDT 是用来防止程序的不正常运行或是跳到未知或不希望去的地址, 而导致不可预见的结果。WDT 可以被掩膜选择禁止。如果在关闭状态, 所有与 WDT 有关的指令操作都是没有作用的。

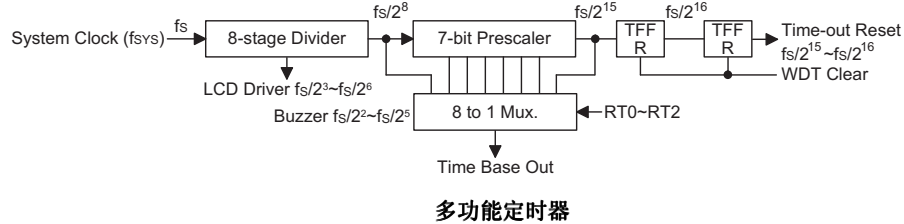
在 HALT 状态时, 如果 f_{OSC} 打开, WDT 会继续计数直到溢出使系统唤醒。

看门狗定时器溢出在正常操作时不仅启动系统复位(Chip Reset), 并置位状态位 TO。在暂停模式中, 这溢出会启动热复位(Warm Reset), 但只有程序计数器和状态指针会复位为零。要清除 WDT 的值可以有三种方法: 外部复位(低电平输入到 RES 端)、清除看门狗指令或 HALT 指令。清除看门狗指令有“CLR WDT”和“CLR WDT1”、“CLR WDT2”二组指令。这两组指令中, 只能选择其中一组, 由掩膜选项决定。如果选择“CLR WDT”, 那么只要执行“CLR WDT”指令就会清除 WDT。如果选择“CLR WDT1”和“CLR WDT2”, 那么二条指令要交替使用才会清除 WDT, 否则, WDT 会由于溢出而使系统复位。

看门狗定时器的溢出时间周期为 $f_j/2^{15} \sim f_j/2^{16}$ 。执行“CLR WDT”指令只能清除 WDT 的最后两级锁存器。

多功能计时器

HT47C06L 为 WDT 和时基提供了具有不同溢出周期的多功能定时器。此多功能定时器由一个八阶除法器及一个七位预分频器所组成，使用的时钟源来自系统时钟。多功能定时器为 LCD 驱动电路提供可选择的频率信号(范围从 $f_s/2^3 \sim f_s/2^6$)，并为蜂鸣器输出电路提供可选择的频率信号(范围从 $f_s/2^2 \sim f_s/2^5$)，频率由掩膜选择。为了正确地显示，建议选择 4KHz 作为 LCD 驱动信号。



时基 —TB

时基用来提供一个有规律的内部中断。溢出周期的范围为 $f_s/2^8 \sim f_s/2^{15}$ ，可通过软件编程实现。写数据到 RT2, RT1, RT0 (RTCC 的第 2, 1, 0 位; 09H) 将产生各种溢出周期。如果时基溢出产生，系统置位相关的中断请求标志位 (TBF; INTC1 第 5 位)。如果中断允许，堆栈未满，那么就产生一个地址在 08H 的子程序调用。在 HALT 状态时，如果 f_{OSC} 打开，时基仍旧工作而且可以唤醒系统。如果在进入 HALT 之前，TBF 被置位，则唤醒功能无效。

RT2	RT1	RT0	时基分频级数
0	0	0	2^8
0	0	1	2^9
0	1	0	2^{10}
0	1	1	2^{11}
1	0	0	2^{12}
1	0	1	2^{13}
1	1	0	2^{14}
1	1	1	2^{15}

暂停模式 — HALT

暂停模式是由 HALT 指令来实现的，具有下列功能：

- f_{OSC} 和 f_{SYS} 工作还是停止取决于 LCD 选项，但 T1 被关闭
- 不改变 RAM 和寄存器的内容
- 看门狗定时器清除并重新计数
- 所有输入/输出口都维持其原有状态
- 置位 PDF 标志并且清除 TO 标志
- 可由掩膜选择液晶显示驱动电路仍继续运作还是关闭
- 时基停止还是保持工作取决于 LCD 选项

PA 口唤醒和中断这两种唤醒方式可以视为正常运行的继续，PA 口上每个位在掩膜选择时都可以单独选择，用来唤醒系统。一旦从输入/输出口启动唤醒之后，程序即从下一条指令重新开始运行。但如果是从中断被唤醒的话，此时可能会发生两种情况。如果相关中断都被禁止，或该中断被允许且堆栈已用满的话，程序会从下一条指令重新开始运行。但如果该中断被允许，但堆栈尚未用满，则会产生一般中断响应。

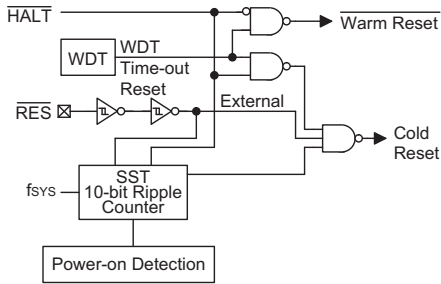
当进入“HALT”状态以前某个中断请求位被置位，那么系统不能用这个中断来唤醒。

如果唤醒是由于中断响应的話，实际中断子程序的执行会延时大约一个以上的周期。如果唤醒事件导致 HALT 下一条指令执行，会立即执行下一条指令。另外，为减少电源损耗，在进入暂停模式之前，应小心处理所有的输入/输出管脚状态。

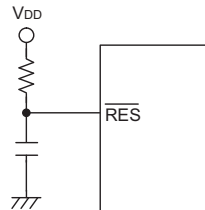
复位

总共有 3 种方法会产生系统复位，如下所示：

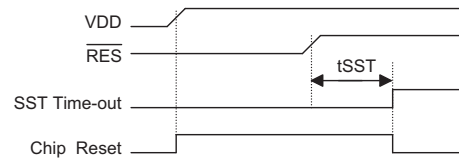
- 正常操作时由 $\overline{\text{RES}}$ 引脚发生复位
- 在暂停模式由 $\overline{\text{RES}}$ 引脚发生复位
- 正常操作时由看门狗定时器溢出发生复位



Reset Configuration
复位框图



Reset Circuit
复位电路



Reset Timing Chart
复位时序

暂停模式中的看门狗定时器溢出与其它系统复位状况不同，因为看门狗定时器溢出会执行热复位，用来重新设置程序计数器和堆栈指针，并保持其它电路原有的状态。少数寄存器在其它复位状态皆不会改变，大部分寄存器一旦符合复位条件时，会复位成初始的状态。通过检测 PDF 和 TO 这两个标志，程序即可区别出各种不同的系统复位。

TO	PDF	复位原因
0	0	上电时 $\overline{\text{RES}}$ 发生复位
u	u	正常运行时 $\overline{\text{RES}}$ 或 $\overline{\text{LVR}}$ 发生复位
0	1	暂停模式下 $\overline{\text{RES}}$ 或 $\overline{\text{LVR}}$ 发生复位
1	u	正常运行时 WDT 溢出
1	1	暂停模式下 WDT 溢出

注：“u”表示不变

系统复位时各功能单元的状态如下所示：

项目	复位后情况
程序计数器 PC	000H
中断	禁止
预分频器、除法器	清除
看门狗定时器、时基	清除、复位后看门狗开始计数
定时/计数器	关闭
输入/输出口	输入模式
堆栈指针 SP	指向堆栈的顶端

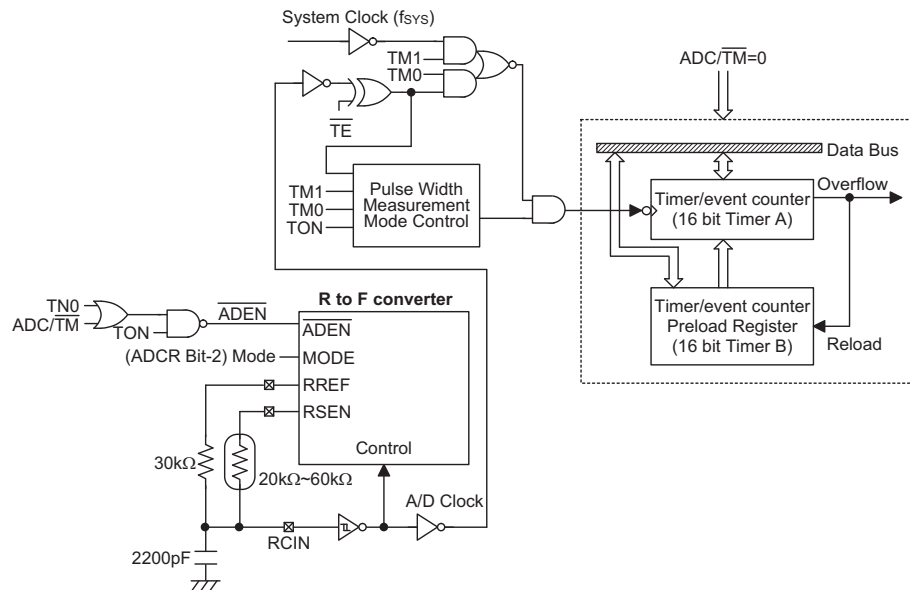
有关寄存器的状态如下:

寄存器	复位(上电)	WDT 溢出 (正常运作)	RES复位 (正常运作)	RES复位 (暂停模式)	WDT 溢出 (暂停模式)
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PC	000H	000H	000H	000H	000H*
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBC	---- -111	---- -111	---- -111	---- -111	---- -uuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC	--00 -000	--00 -000	--00 -000	--00 -000	--uu -uuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
TMRAH	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMRAL	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMRC	-000 1---	-000 1---	-000 1---	-000 1---	-uuu u---
TMRBH	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMRBL	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	---x 0000	---x 0000	---x 0000	---x 0000	---u uuuu

注: “*”表示“热复位”
 “u”表示“不变”
 “x”表示“未知”

定时/计数器

HT47C06L 提供一个 16 位定时/计数器或一个 RC 型 A/D 转换器。ADC/TM 位(ADCR 寄存器的第 1 位)用来决定定时器 A 和定时器 B 是用作 16 位的定时/计数器还是用作 RC 型 A/D 转换器。



定时/计数器

当 ADC/TM 为“0”时, TMRAL、TMRAH、TMRBL、TMRBH 组成了 16 位的定时/计数器。TMRBL 和 TMRBH 组成一个预置寄存器, 分别用来存放定时/计数器初始值的低字节和高字节。

当定时/计数器的时钟来源来自系统时钟 (f_{SYS}) 或外部时钟 (A/D 时钟引脚: RCIN)。外部时钟输入允许用户去计算外部事件, 测量外部 RC 型的 A/D 时钟, 测量时间长度或脉宽、或产生一个精确的时基信号。

总共有六个与定时/计数器工作模式有关的寄存器，分别是 TMRAH([20H])、TMRAL([21H])、TMRC([22H])、TMRBH([23H])、TMRBL([24H])和 ADCR([25H])。若写入 TMRBL 只能将数据写入低字节内部缓冲器中，但若写入的是 TMRBH 则可将数据和低字节内部缓冲器的内容同时写入定时/计数器预置寄存器(16位)之中。改变定时/计数器预置寄存器的内容，只可被写入 TMRBH 之动作改变，但若写入 TMRBL 则可维持定时/计数器预置寄存器的内容不受改变。

同样的，若读取 TMRAH 则可将 TMRAL 传送至低字节内部缓冲器之中，以避免发生计时错误。然而，若读取 TMRAL，则只读回低字节内部缓冲器的内容。换言之，定时/计数器的低字节数据并不能直接读取。若欲读取该低字节的数据，必须先读取 TMRAH，以便将定时/计数器的低字节数据传送至内部低字节缓冲器之中。

TMRC 为定时/计数器控制寄存器，用来定义定时/计数器的某些选项。定时/计数器的控制寄存器可以定义定时/计数器的工作模式、计数的允许或禁止以及计数的触发沿。

位	名称	功能
0~2	—	未定义，读取时为“0”
3	TE	定义定时/计数器 TMR 作用沿： 外部事件计数器模式(TM1, TM0)=(0, 1) 1: 下降沿计数 0: 上升沿计数 脉冲宽度测量模式(TM1, TM0)=(1, 1) 1: 上升沿开始计数，下降沿停止计数 0: 下降沿开始计数，上升沿停止计数
4	TON	允许/禁止定时器计数(0=禁止, 1=允许)
5 6	TM0 TM1	定义操作方式(TM1, TM0) 10=定时器模式(内部时钟: f_{SYS}) 01=计数器模式(外部时钟: A/D 时钟输入脚 RCIN) 11=脉冲宽度测量模式(RCIN, f_{SYS}) 00=未定义
7	—	未定义，读取时为“0”

TMRC (22H) 寄存器

写入定时器 B 就可以将定时/计数器的初始值放到预置寄存器中，而读取定时器 A 就可以得到定时/计数器的内容。定时器 B 是定时/计数器的初始值预置寄存器。

TM0 和 TM1 用来定义操作模式。计数器模式是用来计数外部事件，这表示时钟来源(A/D 时钟)为外部 RCIN 引脚的输入。定时器模式作为普通定时器使用，其时钟来源为内部时钟 (f_{SYS})。最后，脉冲宽度测量模式能够对外部信号 (A/D 时钟引脚: RCIN) 的高电平或低电平的持续时间进行计数，计数的时钟来源为系统时钟 (f_{SYS})。

在事件计数、A/D 时钟或内部定时器模式下，一旦定时/计数器开始计数即从定时/计数器的现行内容(TMRAH 和 TMRAL)开始计数至 FFFFH。若发生溢出，计数器即从定时/计数器预置寄存器(TMRBH 和 TMRBL)重新装入加载值，并同时置位中断请求标志(TF; INTC 的第 4 位)。

在脉冲宽度测量模式下，当 TON 和 TE 位的值都为 1 时，一旦 RCIN 收到由低电平到高电平(如果 TE 位的值为“0”，则为由高电平到低电平)的转变信号，计数器就会开始数，直到 A/D 时钟回到原来的电平为止，并且会将 TON 位清零。测量的结果会依然存放在定时/计数器之中，也就是说一次只能计数一次脉冲的宽度。而当 TON 位重新置位为“1”，只要收到跳变脉冲，测量就会再次执行下去。在脉冲测量模式中，定时/计数器并不会根据逻辑电压来计数，其根据的标准为信号的转变沿。一旦发生计数溢出，计数器会从定时/计数器加载寄存器重新装入初值，同时还会发出中断请求，这种情况和定时和计数这两个模式一样。

若欲启动计数器运行，只要将定时器启动位(TON: TMRC 的第 4 位)的值设为“1”即可。在脉冲宽度测量模式中，TON 位在测量周期完成后，会自动被清除。但在其它两种模式中，TON 位只可以用指令清除。

若在定时/计数器关闭的情况下，将数据写入定时/计数器的预置寄存器也会将该数据重新载入定

时/计数器之中。但若定时/计数器已经开启，写入定时/计数器的数据只会保存在定时/计数器的预置寄存器中。这时定时/计数器并不会马上被改变而会继续计数下去，直到发生溢出为止，此时再由加载寄存器装入新的初始值。

一旦定时/计数器(读取 TMRAH)的数据被读取，会将时钟禁止，以避免发生错误。将时钟禁止可能会导致计数错误，所以程序编写工程师必须考虑清楚才行。

我们强烈建议在打开定时/计数器前先将要加载的数据写入到 TMRBL、TMRBH、TMRAL 和 TMRAH 中去，因为在系统初始化后，TMRBL、TMRBH、TMRAL 和 TMRAH 的值是未知的。

下例为定时/计数器的定时模式(禁止中断):

```

clr  tmrc
clr  adcr.1          ; 设置为定时/计数器模式
clr  intc.4         ; 清除定时/计数器的中断请求标志位
mov  a, low (65536-1000) ; 置定时器初值
mov  tmrbl, a       ; 计数1000然后定时器溢出
mov  a, high (65536-1000)
mov  tmrbh, a

mov  a, 01010000b   ; 定时器时钟来源为fsys并且允许定时器计数
mov  tmrc, a

```

P10:

```
clr  wdt
```

RC 型 A/D 转换器

HT47C06L 有一个 RC 型的 A/D 转换通道，包含两个可编程 16 位向上计数的计数器，计数器 A 的时钟来源可以是系统时钟($f_{\text{SYS}}=32\text{kHz}$)，计数器 B 的时钟来源是外部 RC 振荡电路。当 ADC/ $\overline{\text{TM}}$ 位为“1”时(寄存器 ADCR 的第 1 位)，TMRAL、TMRAH、TMRBL、TMRBH 组成了 A/D 转换器。

A/D 转换定时器 B 的时钟来源可以来自 RREF~RCIN 振荡器、RSEN~RCIN 振荡器、RCIN 外部时钟输入。定时器 A 的时钟来源为系统时钟(设置 TM1, TM0=1, 0)。

总共有六个与 A/D 转换器有关的寄存器，分别是 TMRAH、TMRAL、TMRC、TMRBH、TMRBL 和 ADCR。内部定时器时钟输入到 TMRAH 和 TMRAL 中，A/D 时钟输入到 TMRBH 和 TMRBL 中。 $\overline{\text{OV}}\overline{\text{B}}/\overline{\text{O}}\overline{\text{V}}\overline{\text{A}}$ 位(ADCR 寄存器的第 0 位)用来设置定时器 A 或定时器 B 溢出作为定时/计数器中断信号。在 A/D 转换模式下，当定时器 A 或定时器 B 溢出时 TON 位被清除并且计数器停止计数。写入 TMRAH/TMRBH 就是对定时器 A/定时器 B 设置初值，读取 TMRAH/TMRBH 就是读取定时器 A/定时器 B 的内容，写入 TMRAL/TMRBL 只能将数据写入内部缓冲器的低位字节，但若写入的是 TMRAH/TMRBH 则可将数据和低字节内部缓冲器的内容写入定时器 A/定时器 B(16 位)之中。改变定时 A/定时器 B 的内容，只可被写入 TMRAH/TMRBH 之动作改变，但若写入 TMRAL/TMRBL 则可维持定时 A/定时器 B 的内容不改变。

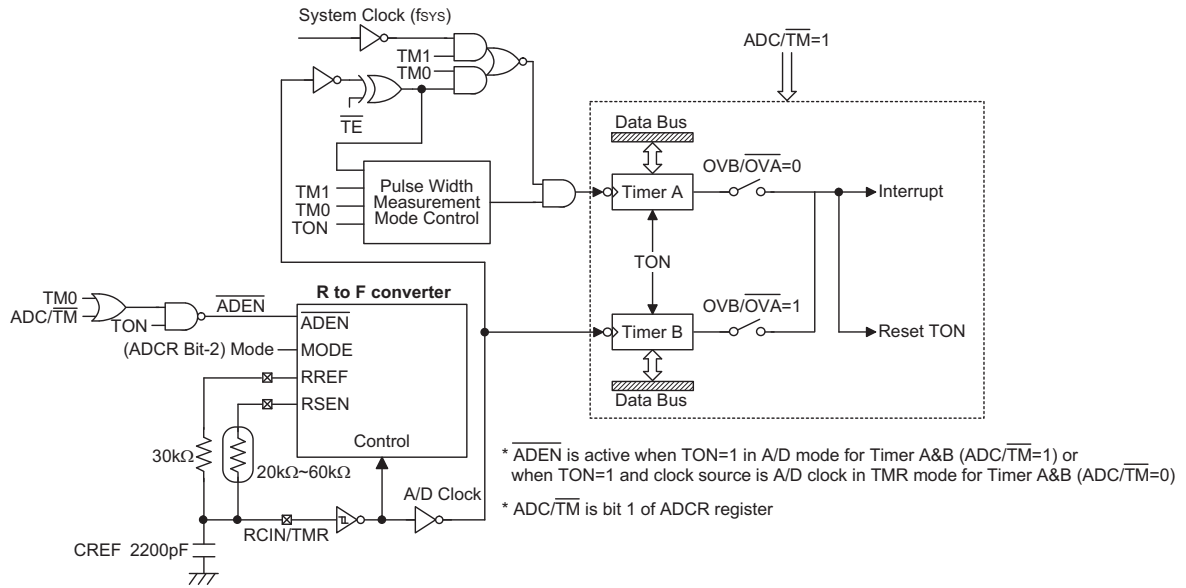
若读取 TMRAH/TMRBH 则可将 TMRAL/TMRBL 传送至低字节内部缓冲器之中，以避免发生计时错误。然而，若读取 TMRAL/TMRBL，则只读回低字节内部缓冲器的内容。换言之，定时器 A/定时器 B 的低字节数据并不能直接读取。若欲读取该低字节的数据，必须先读取 TMRAH/TMRBH，以便将定时/计数器的低字节数据传送至内部低字节缓冲器之中。

寄存器 ADCR 的 2 位用来决定选取哪一组电阻、电容来组成 TMRBH 和 TMRBL 的振荡输入。

寄存器 TMRC 的 TM0 和 TM1 用来决定定时器 A 的时钟来源。建议定时器 A 的时钟来源使用系统时钟。

当 TON 位(TMRC 的第 4 位)置为“1”时，定时器 A 和定时器 B 就开始计数，直到定时器 A 或定时器 B 发生溢出，此时，定时/计数器便置位中断请求标志(TF; INTC 的第 4 位)，同时计数器 A 和计数器 B 停止计数并 TON 位被清为“0”。

当 TON 位置为“1”时，那么 TMRBL、TMRBH、TMRAL 和 TMRAH 不能进行读写操作。只有在定时/计数器关闭并且使用“MOV”指令时，才能对这四个寄存器进行读写操作。



RC 型 A/D 转换器

位	标志	功能
0	OVB/ \overline{OVA}	在 RC 型 A/D 转换模式下, 该位用来定义定时/计数器中断来自定时器 A 溢出或定时器 B 溢出 (0=定时器 A 溢出, 1=定时器 B 溢出) 在定时/计数器模式下, 该位空缺
1	ADC/ \overline{TM}	设定定时/计数器或 RC 型 A/D 转换器允许 (0=定时/计数器允许, 1=A/D 转换器允许)
2	MODE	定义 A/D 转换工作模式: 0=RREF~CREF 振荡 (参考电阻和参考电容) 1=RSEN~CREF 振荡 (电阻型传感器和参考电容)
3	BON	低电压检测使能/除能 (0=除能; 1=使能)
4	BLF	低电压标志位 (0=电池电压正常, 1=电池为低电压)
5~7	—	未定义, 读取时为“0”

ADCR(25H)寄存器

下例是 RC 型 AD 转换模式(定时器 A 溢出):

```

clr    tmrc
clr    adcr.l           ; 设置定时器模式
clr    intc.4          ; 清除定时/计时器中断请求标志位
mov    a, low (65536-1000) ; 置TIMER A初值
mov    tmral, a        ; 计数1000后溢出
mov    a, high (65536-1000)
mov    tmrah, a
mov    a, 00000010b    ; RREF~CREF; 设置RC型ADC模式; 设置TIMER A溢出
mov    adcr,a
mov    a, 00h          ; 置TIMER B初值
mov    tmrbl, a
mov    a, 00h
mov    tmrbh, a
mov    a, 01010000b    ; TIMER A的时钟来源为fSYS并且允许计数
mov    tmrc, a
    
```

```

p10:
  clr    wdt
  snz    intc.4          ; 判断定时/计数器中断请求标志位
  jmp    p10
  clr    intc.4          ; 清除定时/计数器中断请求标志位
                          ; 程序继续

```

下例是RC型AD转换模式(定时器B溢出):

```

  clr    tmrc
  clr    adcr.1          ; 设置定时器模式
  clr    intc.4          ; 清除定时/计数器中断请求标志位
  mov    a, 00h          ; 置TIMERA初值
  mov    tmral, a
  mov    a, 00h
  mov    tmrah, a
  mov    a, 00000011b    ; RREF~CREF; 设置RC型ADC模式; 设置TIMERB溢出
  mov    adcr,a
  mov    a, low (65536-1000) ; 置TIMERB初值
  mov    tmrbl, a        ; 计数1000后溢出
  mov    a, high (65536-1000)
  mov    tmrbh, a
  mov    a, 00110000b    ; TIMERA的时钟来源为fsys并且允许计数
  mov    tmrc, a

p10:
  clr    wdt
  snz    intc.4          ; 判断定时/计数器中断请求标志位
  jmp    p10
  clr    intc.4          ; 清除定时/计数器中断请求标志位
                          ; 程序继续

```

输入/输出口

HT47C06L 有一个 8 位双向输入/输出口 PA，对应 RAM 中的[12H]。所有的输入/输出口都可以作为输入和输出，就输入而言，这些输入/输出口并不具有锁存功能，也就是说，所有输入在 MOV A, [m]指令(m=12H)的 T2 上升沿准备好。就输出而言，所有数据被锁存住，而且不受任何影响，直到输出锁存被写入新的值为止。

PAC 是 PA 口对应的控制寄存器，它控制每个 I/O 的设置。使用控制寄存器，可对 CMOS 输出或带上拉电阻的斯密特触发输入在软件下动态地进行改变。要设置为输入功能，相应的控制寄存器必须写 1，同时输入/输出口会被自动加上上拉电阻。信号源的输入也取决于控制寄存器。如果控制寄存器的某位值为 1，那么输入信号是读取这个引脚的状态，但是如果控制寄存器的某位值为 0，那么锁存器的内容将会被送到内部总线。后者可以在“读—修改—写”指令中发生。对于输出功能，只能设置为 CMOS 输出。这些控制寄存器是对应于 RAM 的 13H 地址。系统复位后，这些输入/输出口默认为输入状态，为高电平或浮接(由掩膜选项决定)。每一个输入/输出锁存位都可以用 SET [m].i 或 CLR [m].i 指令置位或清零 (m=12H)。有些指令会先输入数据，然后才输出运行结果。举例来说：“SET [m].i”，“CLR [m].i”，“CPL [m]”和“CPLA[m]”这些指令会先将整个口状态读入 CPU 中，接着执行所定义的运算，最后再将执行的结果写入锁存或是累加器中。

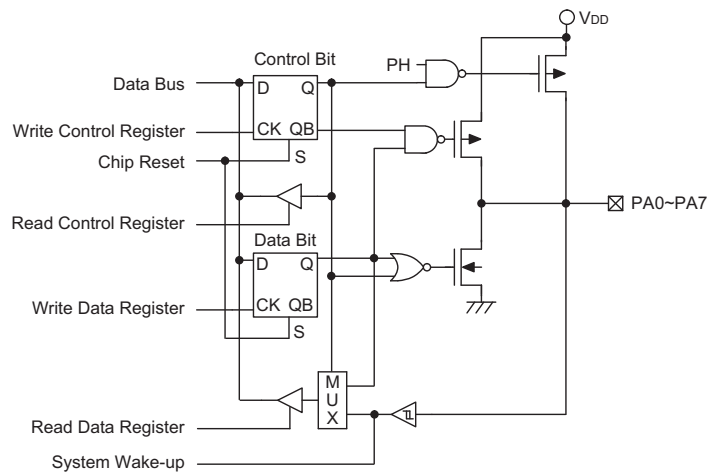
PA 的每个口都具有唤醒系统的能力。

PA0/PA1 分别与 $\overline{\text{BZ}}$ / $\overline{\text{BZ}}$ 共用引脚。如果选择蜂鸣器功能，则 PA0/PA1 在输出模式时的输出信号为蜂鸣器 $\overline{\text{BZ}}$ / $\overline{\text{BZ}}$ 信号，在输入模式始终保持它原来的功能。一旦选择 $\overline{\text{BZ}}$ / $\overline{\text{BZ}}$ ，4kHz 蜂鸣器输出信号由 PA0/PA1 的数据控制。

PA0/PA1 输入/输出功能如下表所示:

PA1 数据寄存器	PA0 数据寄存器	PA1, PA0 引脚功能
0 (CLR PA.1)	0 (CLR PA.0)	PA0=BZ, PA1= $\overline{\text{BZ}}$
1 (SET PA.1)	0 (CLR PA.0)	PA0=BZ, PA1=0
X	1 (SET PA.0)	PA0=0, PA1=0

掩膜选项

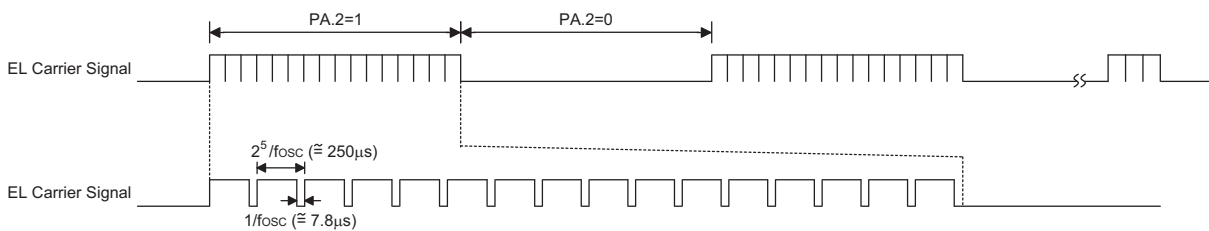


输入/输出口

注: 框图中没有显示 BZ 和 EL 模式功能

PA2 与 EL 载波信号共用引脚。如果选定为 EL 载波输出, 则 PA2 输出模式时的输出信号为 EL 载波信号, 输入模式保持原有的功能。由 PA2 数据寄存器控制 EL 载波信号 (输出模式) 输出。PA2/EL 的真值表罗列如下:

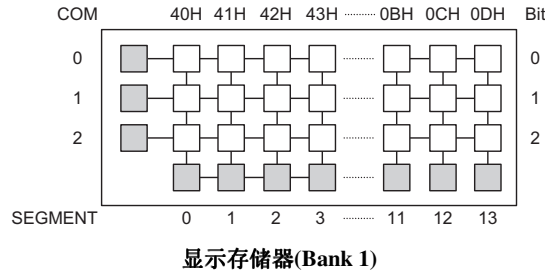
PA2 数据寄存器	PA2 引脚功能
0 (CLR PA.2)	PA2=0
1 (SET PA.2)	PA2=EL 载波输出



EL 时序 ($f_{osc}=128\text{kHz}$)

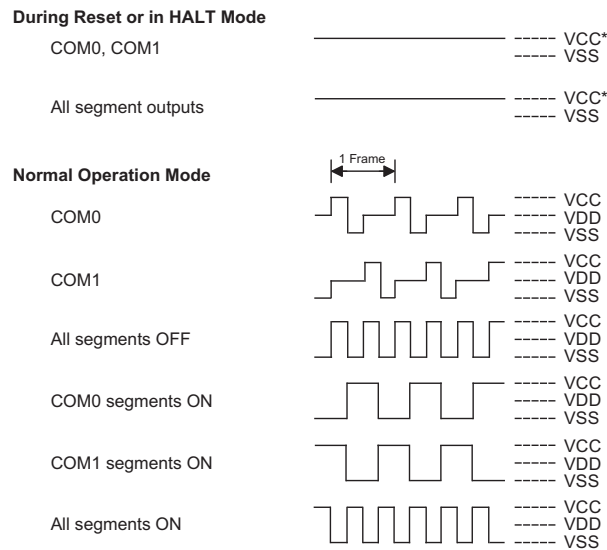
LCD 液晶显示存储器

HT47C06L 为液晶显示提供一块嵌入式的数据存储区。它的存储区域为 RAM 的 BANK1 内的 40H~4DH，存储器段指针(BP; RAM 的 04H 单元)是在 RAM 和 LCD 显示存储器之间的切换开关。当 BP 被置“01H”，任何数据写入 40H~4DH 将会影响 LCD 的显示。当 BP 被清“00H”，任何数据写入 40H~4DH 意味着访问通用数据存储器。LCD 只能通过 MP1 进行间接寻址来访问。当数据被写入显示数据区域，这些数据自动地被 LCD 驱动器读取来产生相应的 LCD 驱动信号。把“1”或“0”写入显示存储器的相应位，来控制显示或不显示。图示表达了显示存储器和 HT47C06L 上 LCD 显示模式之间的对应显示关系。



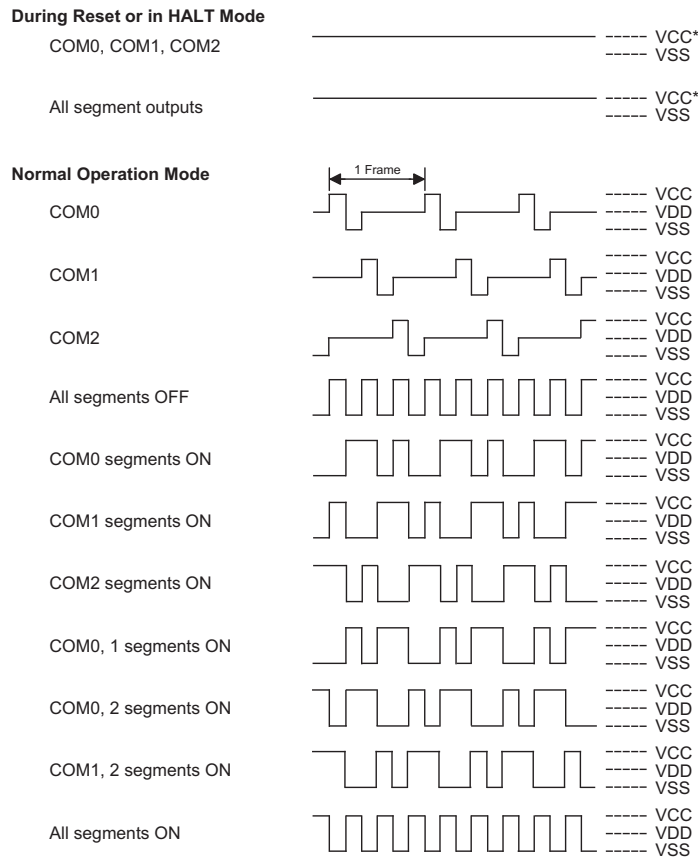
液晶显示驱动输出

HT47C06L 的 LCD 液晶显示驱动器数，可通过掩膜选择为 14x2 或 13x3 (1/2 或 1/3 占空比)。LCD 的驱动偏压方式只有电容式偏压。在 C1 和 C2 引脚之间要接上一个电容，在 VCC 脚和地之间接上一个电容。



LCD 驱动输出(1/2Duty,1/2Bias)

注：在工作模式下，"VCC" $=2 \times V_{DD}$
 在 LCD 关闭或复位时，"VCC*" $=V_{DD}$



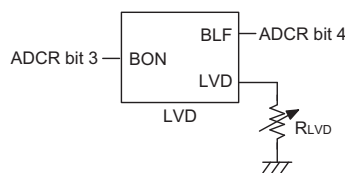
LCD 驱动输出(1/3Duty,1/2Bias)

注：在工作模式下，“VCC”=2×V_{DD}
 在 LCD 关闭或复位时，“VCC*”=V_{DD}

低电压检测 — LVD

HT47C06L 为电池应用系统提供了一个低电压检测功能。如果 LVD 打开，当电池电压低于指定电压值时，低电压标志位被置位 (BLF;ADCR 的第 4 位)。选择合适的外部 R_{LVD}，指定的电压值可以设置为 1.3V±0.05V。低电压检测电路关闭或打开，可通过对 BON (ADCR 第 3 位) 写“1”或“0”来控制。当 BON 为“0”时,BLF 无效。

检测完电压之后，把 BON 设置为“0”，防止 LVD 消耗直流电流。

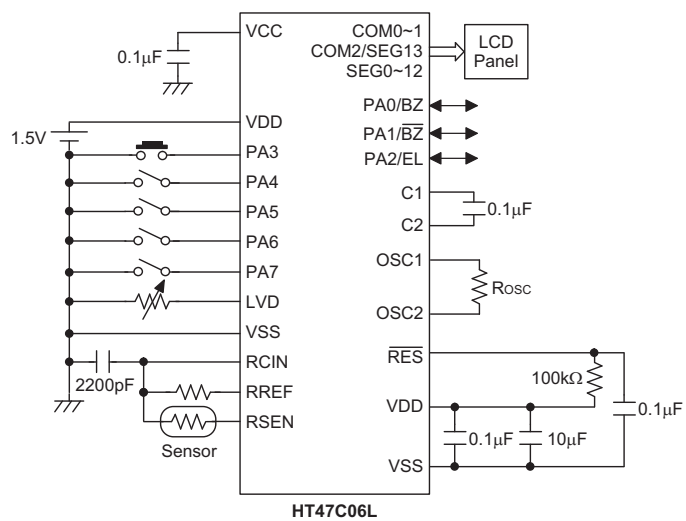


掩膜选择

下表列出了 HT47C06L 的掩膜选择，这些选择必须设定清楚，以确保系统运作正常。

编号	掩膜选择
1	看门狗定时器允许/禁止选择。 (0=允许; 1=禁止)
2	蜂鸣器输出频率选择。 共有四种蜂鸣器输出频率，为 $f_{SYS}/2^2 \sim f_{SYS}/2^5$
3	确定 PA0 和 PA1 的输出功能： 0= 一般输出功能 1= BZ 输出，PA0 输出 BZ，PA1 输出 \overline{BZ}
4	确定 PA2 输出功能： PA2 为一般输出或 EL 载波输出
5	当 CPU 处于 HALT 状态时，振荡/LCD 关闭或打开： 0=HALT 时，振荡/LCD 关闭 1=HALT 时，振荡/LCD 打开
6	PA0 ~PA7 输入模式时，上拉电阻选项 (0: 带上拉; 1: 不带上拉)
7	LCD common 选择, 有两种形式可以选择: 2common(1/2 占空比)或 3 common(1/3 占空比)。如果选择 3 common, 则 segment 输出引脚"SEG13"要设为 common 输出。
8	LCD 驱动时钟选择。 共有四种频率可供 LCD 选择: $f_{SYS}/2^3 \sim f_{SYS}/2^6$

应用电路



应用电路

指令集摘要

助记符	说明	指令周期	影响标志位
算术运算			
ADD A,[m]	ACC 与数据存储器相加, 结果放入 ACC	1	Z,C,AC,OV
ADDM A,[m]	ACC 与数据存储器相加, 结果放入数据存储器	1 ⁽¹⁾	Z,C,AC,OV
ADD A,x	ACC 与立即数相加, 结果放入 ACC	1	Z,C,AC,OV
ADC A,[m]	ACC 与数据存储器、进位标志相加, 结果放入 ACC	1	Z,C,AC,OV
ADCM A,[m]	ACC 与数据存储器、进位标志相加, 结果放入数据存储器	1 ⁽¹⁾	Z,C,AC,OV
SUB A,x	ACC 与立即数相减, 结果放入 ACC	1	Z,C,AC,OV
SUB A,[m]	ACC 与数据存储器相减, 结果放入 ACC	1	Z,C,AC,OV
SUBM A,[m]	ACC 与数据存储器相减, 结果放入数据存储器	1 ⁽¹⁾	Z,C,AC,OV
SBC A,[m]	ACC 与数据存储器、进位标志相减, 结果放入 ACC	1	Z,C,AC,OV
SBCM A,[m]	ACC 与数据存储器、进位标志相减, 结果放入数据存储器	1 ⁽¹⁾	Z,C,AC,OV
DAA [m]	将加法运算中放入 ACC 的值调整为十进制数, 并将结果放入数据存储器	1 ⁽¹⁾	C
逻辑运算			
AND A,[m]	ACC 与数据存储器做“与”运算, 结果放入 ACC	1	Z
OR A,[m]	ACC 与数据存储器做“或”运算, 结果放入 ACC	1	Z
XOR A,[m]	ACC 与数据存储器做“异或”运算, 结果放入 ACC	1	Z
ANDM A,[m]	ACC 与数据存储器做“与”运算, 结果放入数据存储器	1 ⁽¹⁾	Z
ORM A,[m]	ACC 与数据存储器做“或”运算, 结果放入数据存储器	1 ⁽¹⁾	Z
XORM A,[m]	ACC 与数据存储器做“异或”运算, 结果放入数据存储器	1 ⁽¹⁾	Z
AND A,x	ACC 与立即数做“与”运算, 结果放入 ACC	1	Z
OR A,x	ACC 与立即数做“或”运算, 结果放入 ACC	1	Z
XOR A,x	ACC 与立即数做“异或”运算, 结果放入 ACC	1	Z
CPL [m]	对数据存储器取反, 结果放入数据存储器	1 ⁽¹⁾	Z
CPLA [m]	对数据存储器取反, 结果放入 ACC	1	Z
递增和递减			
INCA [m]	递增数据存储器, 结果放入 ACC	1	Z
INC [m]	递增数据存储器, 结果放入数据存储器	1 ⁽¹⁾	Z
DECA [m]	递减数据存储器, 结果放入 ACC	1	Z
DEC [m]	递减数据存储器, 结果放入数据存储器	1 ⁽¹⁾	Z
移位			
RRA [m]	数据存储器右移一位, 结果放入 ACC	1	无
RR [m]	数据存储器右移一位, 结果放入数据存储器	1 ⁽¹⁾	无
RRCA [m]	带进位将数据存储器右移一位, 结果放入 ACC	1	C
RRC [m]	带进位将数据存储器右移一位, 结果放入数据存储器	1 ⁽¹⁾	C
RLA [m]	数据存储器左移一位, 结果放入 ACC	1	无
RL [m]	数据存储器左移一位, 结果放入数据存储器	1 ⁽¹⁾	无
RLCA [m]	带进位将数据存储器左移一位, 结果放入 ACC	1	C
RLC [m]	带进位将数据存储器左移一位, 结果放入数据存储器	1 ⁽¹⁾	C
数据传送			
MOV A,[m]	将数据存储器送至 ACC	1	无
MOV [m],A	将 ACC 送至数据存储器	1 ⁽¹⁾	无
MOV A,x	将立即数送至 ACC	1	无
位运算			
CLR [m].i	清除数据存储器的位	1 ⁽¹⁾	无
SET [m].i	置位数据存储器的位	1 ⁽¹⁾	无

助记符	说明	指令周期	影响标志位
转移			
JMP addr	无条件跳转	2	无
SZ [m]	如果数据存储器为零, 则跳过下一条指令	1 ⁽²⁾	无
SZA [m]	数据存储器送至 ACC, 如果内容为零, 则跳过下一条指令	1 ⁽²⁾	无
SZ [m].i	如果数据存储器的第 i 位为零, 则跳过下一条指令	1 ⁽²⁾	无
SNZ [m].i	如果数据存储器的第 i 位不为零, 则跳过下一条指令	1 ⁽²⁾	无
SIZ [m]	递增数据存储器, 如果结果为零, 则跳过下一条指令	1 ⁽³⁾	无
SDZ [m]	递减数据存储器, 如果结果为零, 则跳过下一条指令	1 ⁽³⁾	无
SIZA [m]	递增数据存储器, 将结果放入 ACC, 如果结果为零, 则跳过下一条指令	1 ⁽²⁾	无
SDZA [m]	递减数据存储器, 将结果放入 ACC, 如果结果为零, 则跳过下一条指令	1 ⁽²⁾	无
CALL addr	子程序调用	2	无
RET	从子程序返回	2	无
RET A,x	从子程序返回, 并将立即数放入 ACC	2	无
RETI	从中断返回	2	无
查表			
TABRDC [m]	读取当前页的 ROM 内容, 并送至数据存储器 and TBLH	2 ⁽¹⁾	无
TABRDL [m]	读取最后页的 ROM 内容, 并送至数据存储器 and TBLH	2 ⁽¹⁾	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 ⁽¹⁾	无
SET [m]	置位数据存储器	1 ⁽¹⁾	无
CLR WDT	清除看门狗定时器	1	TO,PDF
CLR WDT1	预清除看门狗定时器	1	TO ⁽⁴⁾ ,PDF ⁽⁴⁾
CLR WDT2	预清除看门狗定时器	1	TO ⁽⁴⁾ ,PDF ⁽⁴⁾
SWAP [m]	交换数据存储器的高低字节, 结果放入数据存储器	1 ⁽¹⁾	无
SWAPA [m]	交换数据存储器的高低字节, 结果放入 ACC	1	无
HALT	进入暂停模式	1	TO,PDF

注: x: 立即数

m: 数据存储器地址

A: 累加器

i: 第 0~7 位

addr: 程序存储器地址

√: 影响标志位

—: 不影响标志位

(1): 如果数据是加载到 PCL 寄存器, 则指令执行周期会被延长一个指令周期(四个系统时钟)。

(2): 如果满足跳跃条件, 则指令执行周期会被延长一个指令周期(四个系统时钟); 否则指令执行周期不会被延长。

(3): (1)和(2)

(4): 如果执行 CLR WDT1 或 CLR WDT2 指令后, 看门狗定时器被清除, 则会影响 TO 和 PDF 标志位; 否则不会影响 TO 和 PDF 标志位。

ADC A, [m] 累加器与数据存储器、进位标志相加，结果放入累加器
 说明：本指令把累加器、数据存储器值以及进位标志相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + [m] + C$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADCM A, [m] 累加器与数据存储器、进位标志相加，结果放入数据存储器
 说明：本指令把累加器、数据存储器值以及进位标志相加，结果存放到存储器。
 运算过程： $[m] \leftarrow ACC + [m] + C$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A, [m] 累加器与数据存储器相加，结果放入累加器
 说明：本指令把累加器、数据存储器值相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADD A, x 累加器与立即数相加，结果放入累加器
 说明：本指令把累加器值和立即数相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + x$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

ADDM A, [m] 累加器与数据存储器相加，结果放入数据存储器
 说明：本指令把累加器、数据存储器值相加，结果存放到数据存储器。
 运算过程： $[m] \leftarrow ACC + [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

AND A, [m] 累加器与数据存储器做“与”运算，结果放入累加器
 说明：本指令把累加器值、数据存储器值做逻辑与，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC \text{ “AND” } [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

AND A, x 累加器与立即数做“与”运算，结果放入累加器
 说明：本指令把累加器值、立即数做逻辑与，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC \text{ “AND” } x$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ANDM A, [m] 累加器与数据存储器做“与”运算，结果放入数据存储器
 说明：本指令把累加器值、数据存储器值做逻辑与，结果存放到数据存储器。
 运算过程： $[m] \leftarrow ACC \text{ “AND” } [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CALL addr 子程序调用
 说明：本指令直接调用地址所在处的子程序，此时程序计数器加一，将此程序计数器值存到堆栈寄存器中，再将子程序所在处的地址存放到程序计数器中。
 运算过程： $Stack \leftarrow Program Counter + 1$
 $Program Counter \leftarrow addr$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m] 清除数据存储器
 说明：本指令将数据存储器内的数值清零。
 运算过程： $[m] \leftarrow 00H$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m].i 将数据存储器的第 i 位清“0”
 说明：本指令将数据存储器内第 i 位值清零。
 运算过程： $[m].i \leftarrow 0$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR WDT 清除看门狗定时器
 说明：本指令清除 WDT 计数器(从 0 开始重新计数)，暂停标志位(PDF)和看门狗溢出标志位(TO)也被清零。
 运算过程： $WDT \leftarrow 00H$
 $PDF \& TO \leftarrow 0$
 影响标志位

TO	PDF	OV	Z	AC	C
0	0	—	—	—	—

CLR WDT1 预清除看门狗定时器
 说明：必须搭配 CLR WDT2 一起使用，才可清除 WDT 计时器(从 0 开始重新计数)。当程序只执行过该指令，没有执行 CLR WDT2 时，系统只会不会将暂停标志位(PDF)和计数溢出位(TO)清零，PDF 与 TO 保留原状态不变。

运算过程: $WDT \leftarrow 00H^*$
 $PDF\&TO \leftarrow 0^*$

影响标志位

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

CLR WDT2 预清除看门狗定时器

说明: 必须搭配 CLR WDT1 一起使用, 才可清除 WDT 计时器(从 0 开始重新计数)。当程序只执行过该指令, 没有执行 CLR WDT1 时, 系统只会不会将暂停标志位(PDF)和计数溢出位(TO)清零, PDF 与 TO 保留原状态不变。

运算过程: $WDT \leftarrow 00H^*$
 $PDF\&TO \leftarrow 0^*$

影响标志位

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

CPL [m] 对数据存储器取反, 结果放入数据存储器

说明: 本指令是将数据存储器内保存的数值取反。

运算过程: $[m] \leftarrow [\bar{m}]$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

CPLA [m] 对数据存储器取反, 结果放入累加器

说明: 本指令是将数据存储器内保存的值取反后, 结果存放在累加器中。

运算过程: $ACC \leftarrow [\bar{m}]$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DAA [m] 将加法运算后放入累加器的值调整为十进制数，并将结果放入数据存储器
 说明 本指令将累加器高低四位分别调整为 BCD 码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，并且内部进位标志 AC1= \overline{AC} ，即 AC 求反；否则原值保持不变。如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”再加 AC1，并把 C 置位；否则 BCD 调整就执行对原值加 AC1，C 的值保持不变。结果存放到数据存储器中，只有进位标志位(C)受影响。

操作 如果 ACC.3~ACC.0 > 9 或 AC=1
 那么 [m].3~[m].0 \leftarrow (ACC.3~ACC.0)+6, AC1= \overline{AC}
 否则 [m].3~[m].0 \leftarrow (ACC.3~ACC.0), AC1=0
 并且
 如果 ACC.7~ACC.4+AC1 > 9 或 C=1
 那么 [m].7~[m].4 \leftarrow (ACC.7~ACC.4)+6+ AC1, C=1
 否则 [m].7~[m].4 \leftarrow (ACC.7~ACC.4)+ AC1, C=C

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

DEC [m] 数据存储器内容减 1，结果放入数据存储器
 说明 本指令将数据存储器内的数值减一再放回数据存储器。
 运算过程: [m] \leftarrow [m]-1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DECA [m] 数据存储器内容减 1，结果放入累加器
 说明 本指令将存储器内的数值减一，再放到累加器。
 运算过程: ACC \leftarrow [m]-1

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

HALT 进入暂停模式
 说明 本指令终止程序执行并关掉系统时钟，RAM 和寄存器内的数值保持原状态，WDT 计数器清“0”，暂停标志位(PDF)被设为 1，WDT 计数溢出位(TO)被清为 0。

运算过程: Program Counter \leftarrow Program Counter+1

PDF \leftarrow 1

TO \leftarrow 0

影响标志位

TO	PDF	OV	Z	AC	C
0	1	—	—	—	—

INC [m] 数据存储器的内容加 1，结果放入数据存储器
 说明：本指令将数据存储器内的数值加一，结果放回数据存储器。
 运算过程： $[m] \leftarrow [m]+1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

INCA [m] 数据存储器的内容加 1，结果放入数据存储器
 说明：本指令是将存储器内的数值加一，结果放到累加器。
 运算过程： $ACC \leftarrow [m]+1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

JMP addr 无条件跳转
 说明：本指令是将要跳到的目的地直接放到程序计数器内。
 运算过程： $Program\ Counter \leftarrow addr$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A, [m] 将数据存储器送至累加器
 说明：本指令是将数据存储器内的数值送到累加器内。
 运算过程： $ACC \leftarrow [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV A, x 将立即数送至累加器
 说明：本指令是将立即数送到累加器内。
 运算过程： $ACC \leftarrow x$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

MOV [m], A 将累加器送至数据存储器
 说明：本指令是将累加器值送到数据存储器内。
 运算过程： $[m] \leftarrow ACC$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

NOP 空指令
 说明: 本指令不作任何运算, 而只将程序计数器加一。
 运算过程: $\text{Program Counter} \leftarrow \text{Program Counter} + 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

OR A, [m] 累加器与数据存储器做“或”运算, 结果放入累加器
 说明: 本指令是把累加器、数据存储器值做逻辑或, 结果放到累加器。
 运算过程: $\text{ACC} \leftarrow \text{ACC} \text{ “OR” } [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

OR A, x 累加器与立即数做“或”运算, 结果放入累加器
 说明: 本指令是把累加器值、立即数做逻辑或, 结果放到累加器。
 运算过程: $\text{ACC} \leftarrow \text{ACC} \text{ “OR” } x$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

ORM A, [m] 累加器与数据存储器做“或”运算, 结果放入数据存储器
 说明: 本指令是把累加器值、存储器值做逻辑或, 结果放到数据存储器。
 运算过程: $[m] \leftarrow \text{ACC} \text{ “OR” } [m]$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

RET 从子程序返回
 说明: 本指令是将堆栈寄存器中的程序计数器值送回程序计数器。
 运算过程: $\text{Program Counter} \leftarrow \text{Stack}$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RET A, x 从子程序返回, 并将立即数放入累加器
 说明: 本指令是将堆栈寄存器中的程序计数器值送回程序计数器, 并将立即数送回累加器。
 运算过程: $\text{Program Counter} \leftarrow \text{Stack}$
 $\text{ACC} \leftarrow x$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RETI 从中断返回
 说明: 本指令是将堆栈寄存器中的程序计数器值送回程序计数器, 与 RET 不同的是它使用在中断程序结束返回时, 它还会将中断控制寄存器 INTC 的 0 位(EMI)中断允许位置 1, 允许中断服务。

运算过程: $\text{Program Counter} \leftarrow \text{Stack}$
 $\text{EMI} \leftarrow 1$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RL [m] 数据存储器左移一位, 结果放入数据存储器
 说明: 本指令是将数据存储器内的数值左移一位, 第 7 位移到第 0 位, 结果送回数据存储器。

运算过程: $[\text{m}].0 \leftarrow [\text{m}].7, [\text{m}].(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RLA [m] 数据存储器左移一位, 结果放入累加器
 说明: 本指令是将存储器内的数值左移一位, 第 7 位移到第 0 位, 结果送到累加器, 而数据存储器内的数值不变。

运算过程: $\text{ACC}.0 \leftarrow [\text{m}].7, \text{ACC}.(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RLC [m] 带进位将数据存储器左移一位, 结果放入数据存储器
 说明: 本指令是将存储器内的数值与进位标志左移一位, 第 7 位取代进位标志, 进位标志移到第 0 位, 结果送回数据存储器。

运算过程: $[\text{m}].(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

$[\text{m}].0 \leftarrow \text{C}$

$\text{C} \leftarrow [\text{m}].7$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RLCA [m] 带进位将数据存储器左移一位, 结果放入累加器
 说明: 本指令是将存储器内的数值与进位标志左移一位, 第七位取代进位标志, 进位标志移到第 0 位, 结果送回累加器。

运算过程: $\text{ACC}.(\text{i}+1) \leftarrow [\text{m}].\text{i}; (\text{i}=0\sim 6)$

$\text{ACC}.0 \leftarrow \text{C}$

$\text{C} \leftarrow [\text{m}].7$

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RR [m] 数据存储器右移一位，结果放入数据存储器
 说明：本指令是将存储器内的数值循环右移，第 0 位移到第 7 位，结果送回数据存储器。
 运算过程： $[m].7 \leftarrow [m].0$, $[m].i \leftarrow [m].(i+1)$; (i=0~6)
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRA [m] 数据存储器右移一位，结果放入累加器
 说明：本指令是将数据存储器内的数值循环右移，第 0 位移到第 7 位，结果送回累加器，而数据存储器内的数值不变。
 运算过程： $ACC.7 \leftarrow [m].0$, $ACC.i \leftarrow [m].(i+1)$; (i=0~6)
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RRC [m] 带进位将数据存储器右移一位，结果放入数据存储器
 说明：本指令是将存储器内的数值加进位标志循环右移，第 0 位取代进位标志，进位标志移到第 7 位，结果送回存储器。
 运算过程： $[m].i \leftarrow [m].(i+1)$; (i=0~6)
 $[m].7 \leftarrow C$
 $C \leftarrow [m].0$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RRCA [m] 带进位将数据存储器右移一位，结果放入累加器
 说明：本指令是将数据存储器内的数值加进位标志循环右移，第 0 位取代进位标志，进位标志移到第 7 位，结果送回累加器，数据存储器内的数值不变。
 运算过程： $ACC.i \leftarrow [m].(i+1)$; (i=0~6)
 $ACC.7 \leftarrow C$
 $C \leftarrow [m].0$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

SBC A,[m] 累加器与数据存储器、进位标志相减，结果放入累加器
 说明：本指令是把累加器值减去数据存储器值以及进位标志的取反，结果放到累加器。
 运算过程： $ACC \leftarrow ACC + [\overline{m}] + C$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SBCM A,[m] 累加器与数据存储器、进位标志相减，结果放入数据存储器
 说明：本指令是把累加器值减去数据存储器值以及进位标志取反，结果放到数据存储器。
 运算过程： $[m] \leftarrow ACC + [\overline{m}] + C$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SDZ [m] 数据存储器减 1，如果结果为“0”，则跳过下一条指令
 说明：本指令是把数据存储器内的数值减 1，判断是否为 0，若为 0 则跳过下一条指令，即如果结果为零，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。
 运算过程：如果 $[m]-1=0$ ，跳过下一条指令执行再下一条。
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SDZA [m] 数据存储器减 1，将结果放入累加器，如果结果为“0”，则跳过下一条指令
 说明：本指令是把数据存储器内的数值减 1，判断是否为 0，为 0 则跳过下一行指令并将减完后数据存储器内的数值送到累加器，而数据存储器内的值不变，即若结果为 0，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。
 运算过程：如果 $[m]-1=0$ ，跳过下一条指令执行再下一条。
 $ACC \leftarrow ([m]-1)$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m] 置位数据存储器
 说明：本指令是把存储器内的数值每个位置为 1。
 运算过程： $[m] \leftarrow FFH$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m].i 将数据存储器的第 i 位置“1”
 说明：本指令是把存储器内的数值的第 i 位置为 1。
 运算过程： $[m].i \leftarrow 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZ [m] 数据存储器加 1，如果结果为“0”，则跳过下一条指令
 说明：本指令是把数据存储器内的数值加 1，判断是否为 0。若为 0，跳过下一条指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程：如果 $([m]+1=0)$ ，跳过下一行指令； $[m] \leftarrow [m]+1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZA 数据存储器加 1，将结果放入累加器，如果结果为“0”，则跳过下一条指令
 说明：本指令是把数据存储器内的数值加 1，判断是否为 0，若为 0 跳过下一条指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)，并将加完后存储器内的数值送到累加器，而数据存储器的值保持不变。否则执行下一条指令(一个指令周期)。

运算过程：如果 $[m]+1=0$ ，跳过下一行指令； $ACC \leftarrow ([m]+1)$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SNZ [m].i 如果数据存储器的第 i 位不为“0”，则跳过下一条指令
 说明：本指令是判断数据存储器内的数值的第 i 位，若不为 0，则程序计数器再加 1，跳过下一行指令，放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以取得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程：如果 $[m].i \neq 0$ ，跳过下一行指令。
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SUB A, [m] 累加器与数据存储器相减，结果放入累加器
 说明：本指令是把累加器值、数据存储器值相减，结果放到累加器。

运算过程： $ACC \leftarrow ACC + [\bar{m}] + 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUB A, x 累加器与立即数相减，结果放入累加器
 说明：本指令是把累加器值、立即数相减，结果放到累加器。

运算过程： $ACC \leftarrow ACC + \bar{x} + 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SUBM A, [m] 累加器与数据存储器相减，结果放入数据存储器
 说明：本指令是把累加器值、存储器值相减，结果放到存储器。
 运算过程： $[m] \leftarrow ACC + \overline{[m]} + 1$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SWAP [m] 交换数据存储器的高低字节，结果放入数据存储器
 说明：本指令是将数据存储器的低四位和高四位互换，再将结果送回数据存储器。
 运算过程： $[m].7 \sim [m].4 \leftrightarrow [m].3 \sim [m].0$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SWAPA [m] 交换数据存储器的高低字节，结果放入累加器
 说明：本指令是将数据存储器的低四位和高四位互换，再将结果送回累加器。
 运算过程： $ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$
 $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m] 如果数据存储器为“0”，则跳过下一条指令
 说明：本指令是判断数据存储器内的数值是否为0，为0则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。
 运算过程：如果 $[m] = 0$ ，跳过下一行指令。
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZA [m] 数据存储器送至累加器，如果内容为“0”，则跳过下一条指令
 说明：本指令是判断存储器内的数值是否为0，若为0则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。并把存储器内值送到累加器，而存储器的值保持不变。否则执行下一条指令(一个指令周期)。
 运算过程：如果 $[m] = 0$ ，跳过下一行指令，并 $ACC \leftarrow [m]$ 。
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m]. i 如果数据存储器的第 i 位为“0”，则跳过下一条指令
 说明：本指令是判断存储器内第 i 位值是否为 0，若为 0 则跳过下一行指令，即放弃在目前指令执行期间所取得的下一条指令，并插入一个空周期用以得正确的指令(二个指令周期)。否则执行下一条指令(一个指令周期)。

运算过程：如果 [m].i = 0，跳过下一行指令。
 影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDC [m] 读取 ROM 当前页的内容，并送至数据存储器 and TBLH
 说明：本指令是将表格指针指向程序寄存器当前页，将低字节送到存储器，高字节直接送到 TBLH 寄存器内。

运算过程：[m] ← 程序存储器低字节
 TBLH ← 程序存储器高字节

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

TABRDL [m] 读取 ROM 最后一页的内容，并送至数据存储器 and TBLH
 说明：本指令是将 TABLE 指针指向程序寄存器最后页，将低字节送到存储器，高字节直接送到 TBLH 寄存器内。

运算过程：[m] ← 程序存储器低字节
 TBLH ← 程序存储器高字节

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

XOR A, [m] 累加器与立即数做“异或”运算，结果放入累加器
 说明：本指令是把累加器值、数据存储器值做逻辑异或，结果放到累加器。
 运算过程：ACC ← ACC “XOR” [m]

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XORM A, [m] 累加器与数据存储器做“异或”运算，结果放入数据存储器
 说明：本指令是把累加器值、数据存储器值做逻辑异或，结果放到数据存储器。

运算过程：[m] ← ACC “XOR” [m]

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XOR A, x 累加器与数据存储器做“异或”运算，结果放入累加器
 说明：本指令是把累加器值与立即数做逻辑异或，结果放到累加器。

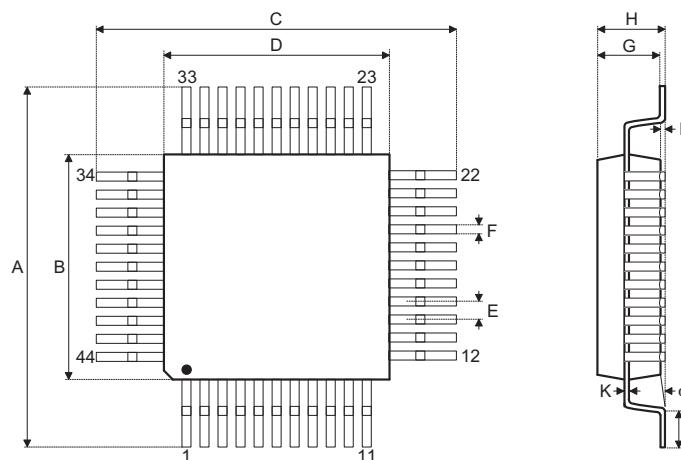
运算过程：ACC ← ACC “XOR” x

影响标志位

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

封装信息

44-pin QFP (10×10)外形尺寸



符号	尺寸 (单位: mm)		
	最小	典型	最大
A	13	—	13.40
B	9.90	—	10.10
C	13	—	13.40
D	9.90	—	10.10
E	—	0.80	—
F	—	0.30	—
G	1.90	—	2.20
H	—	—	2.70
I	—	0.10	—
J	0.73	—	0.93
K	0.10	—	0.20
α	0°	—	7°

盛群半导体股份有限公司（总公司）

新竹市科学工业园区研新二路3号
电话: 886-3-563-1999
传真: 886-3-563-1189
网站: www.holtek.com.tw

盛群半导体股份有限公司（台北业务处）

台北市南港区园区街3之2号4楼之2
电话: 886-2-2655-7070
传真: 886-2-2655-7373
传真: 886-2-2655-7383 (International sales hotline)

盛扬半导体有限公司（上海业务处）

上海宜山路889号2号楼7楼 200233
电话: 021-6485-5560
传真: 021-6485-0313
网站: www.holtek.com.cn

盛扬半导体有限公司（深圳业务处）

深圳市南山区科技园科技中三路与高新中二道交汇处生产力大楼A单元五楼 518057
电话: 0755-8616-9908, 8616-9308
传真: 0755-8616-9722

盛扬半导体有限公司（北京业务处）

北京市西城区宣武门西大街甲129号金隅大厦1721室 100031
电话: 010-6641-0030, 6641-7751, 6641-7752
传真: 010-6641-0125

盛扬半导体有限公司（成都业务处）

成都市东大街97号香樟广场C座709室 610016
电话: 028-6653-6590
传真: 028-6653-6591

Holtek Semiconductor (USA), Inc.（北美业务处）

46712 Fremont Blvd., Fremont, CA 94538
电话: 510-252-9880
传真: 510-252-9885
网站: www.holtek.com

Copyright © 2007 by HOLTEK SEMICONDUCTOR INC.

使用指南中所出现的信息在出版当时相信是正确的，然而盛群对于说明书的使用不负任何责任。文中提到的应用目的仅仅是用来做说明，盛群不保证或表示这些没有进一步修改的应用将是适当的，也不推荐它的产品使用在会由于故障或其它原因可能会对人身造成危害的地方。盛群产品不授权使用于救生、维生器件或系统中做为关键器件。盛群拥有不事先通知而修改产品的权利，对于最新的信息，请参考我们的网址 <http://www.holtek.com.tw>