

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

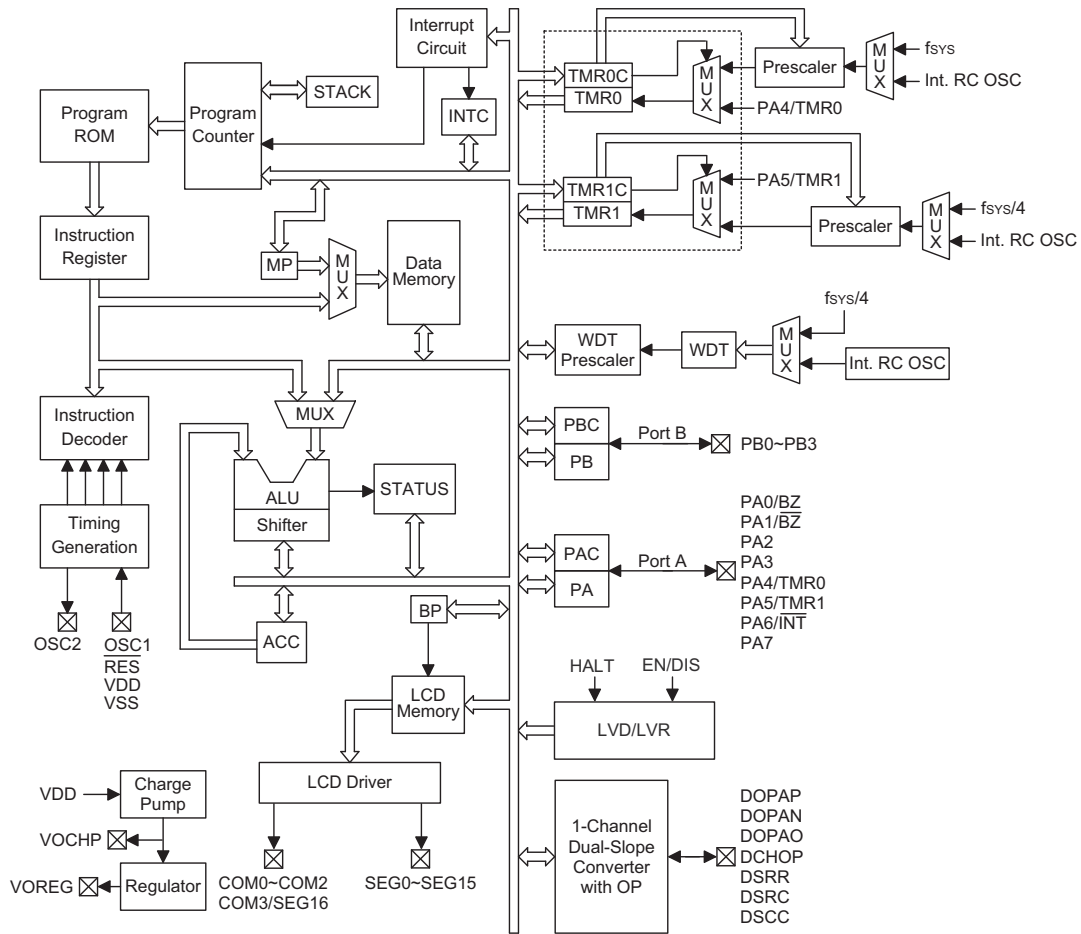
盛群之名称和标识、Holtek 标识、HT-IDE、HT-ICE、Marvel Speech、Music Micro、Adlib Micro、Magic Voice、Green Dialer、PagerPro、Q-Voice、Turbo Voice、EasyVoice 和 HandyWriter 都是盛群半导体公司在台湾地区和其它国家的注册商标。

著作权

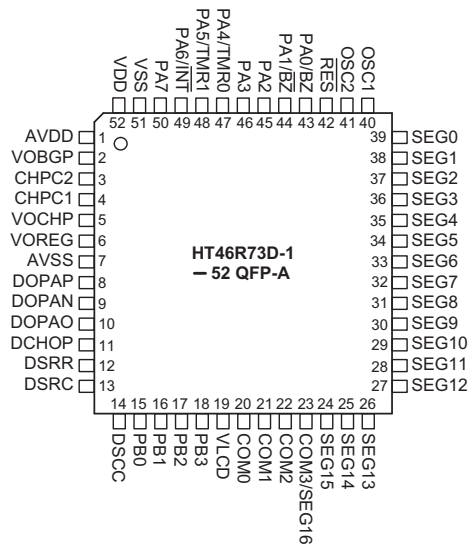
Copyright © 2008 by HOLTEK SEMICONDUCTOR INC.

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>

方框图



引脚图



引脚说明

引脚名称	输入/输出	掩膜选项	说明
PA0/BZ PA1/BZ PA2 PA3 PA4/TMR0 PA5/TMR1 PA6/ $\overline{\text{INT}}$ PA7	输入/输出	唤醒功能 上拉电阻 蜂鸣器	8 位双向输入/输出口。每一位可由掩膜选项设置为唤醒输入。可由软件设置为 CMOS 输出、带或不带上拉电阻(由上拉电阻选项决定: 位选择)的斯密特触发输入。BZ、 $\overline{\text{BZ}}$ 、TMR0、TMR1 和 INT 分别与 PA0、PA1、PA4、PA5 和 PA6 共用引脚。
PB0~PB3	输入/输出	上拉电阻	4 位双向输入/输出口。可由软件设置为 CMOS 输出、带或不带上拉电阻(由掩膜选项决定)的斯密特触发输入。
VLCD	输入	—	LCD 电源。
COM0~COM2 COM3/SEG16	输出	1/2、1/3 或 1/4Duty	COM0~COM3 是驱动 LCD 的 Common 输出。由配置选项设置 LCD 的占空比。当 1/2 或 1/3duty 被选择, 则 COM3/SEG16 口即被设置为 SEG16
SEG0~SEG15	输出	Segment 输出	驱动 LCD 的 Segment 输出。
VOBGP	模拟输出	—	Band gap 电压输出(供内部使用)
VOREG	输出	—	3.3V 稳压输出
VOCHP	输出	—	充电泵输出(需要连接一电容)
CHPC1	—	—	充电泵电容正极
CHPC2	—	—	充电泵电容负极
DOPAN DOPAP DOPAO DCHOP	模拟输入 模拟输出	—	双积分转换器前置放大器相关引脚。DOPAN 是 OPA 负输入引脚, DOPAP 是 OPA 正输入引脚, DOPAO 是 OPA 输出引脚, DCHOP 是 OPA 断路器引脚。
DSRR DSRC DSCC	模拟输入/ 模拟输出	—	双积分转换器主要 RC 电路。DSRR 是输入引脚或参考电压引脚, DSRC 是积分器的负极输入, DSCC 是比较器的负极输入。
OSC1 OSC2	输入 输出	晶振或 RC 振荡	OSC1、OSC2 连接外部 RC 或晶振, 以产生内部系统时钟。在外部 RC 振荡方式下, OSC2 是系统时钟四分频的输出口。
$\overline{\text{RES}}$	输入	—	斯密特触发复位输入。
VDD	—	—	正电源。
VSS	—	—	负电源。
AVSS	—	—	模拟地。
AVDD	—	—	模拟正电源。

极限参数

电源供应电压..... $V_{SS}-0.3V \sim V_{SS}+6.0V$	储存温度..... $-50^{\circ}\text{C} \sim 125^{\circ}\text{C}$
端口输入电压..... $V_{SS}-0.3V \sim V_{DD}+0.3V$	工作温度..... $-40^{\circ}\text{C} \sim 85^{\circ}\text{C}$
端口总灌电流 150mA	端口总源电流 -100mA
总功耗 500mW	

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	—	f _{SYS} =4MHz	2.2	—	5.5	V
		—	f _{SYS} =8MHz	3.3	—	5.5	V
I _{DD1}	工作电流 (晶体振荡)	5V	无负载, f _{SYS} =8MHz, ADC 关闭	—	4	8	mA
I _{DD2}	工作电流 (晶体振荡或 RC 振荡)	3V	无负载, ADC 关闭,	—	0.8	1.5	mA
		5V	f _{SYS} =4MHz	—	2.5	4	mA
I _{DD3}	工作电流 (晶体振荡或 RC 振荡)	3V	无负载, ADC 关闭,	—	0.5	1	mA
		5V	f _{SYS} =2MHz	—	1.5	3	mA
I _{DD4}	工作电流 (晶体振荡或 RC 振荡)	5V	V _{REGO} =3.3V, f _{SYS} =4MHz, ADC 打开, ADCCCLK=125KHz (所有其它模拟器件 全部关闭)	—	3	5	mA
I _{STB1}	静态电流 (看门狗关闭)	3V	无负载, 系统 HALT,	—	—	1	μA
		5V	HALT 时 LCD 关闭	—	—	2	μA
I _{STB2}	静态电流 (看门狗打开)	3V	无负载, 系统 HALT, HALT 时 LCD 关闭,	—	2.5	5	μA
		5V	ADC 关闭	—	8	15	μA
I _{STB3}	静态电流 (看门狗关闭, 内部 12kHz 振荡打开)	3V	无负载, 系统 HALT, HALT 时 LCD 关闭,	—	2	5	μA
		5V	ADC 关闭	—	6	10	μA
I _{STB4}	静态电流 (看门狗关闭, 内部 12kHz 振荡打开)	3V	无负载, 系统 HALT, HALT 时 LCD 打开,	—	17	30	μA
		5V	1/2bias, V _{LCD} =V _{DD}	—	34	60	μA
I _{STB5}	静态电流 (看门狗关闭, 内部 12kHz 振荡打开)	3V	无负载, 系统 HALT, HALT 时 LCD 打开,	—	13	25	μA
		5V	1/3bias, V _{LCD} =V _{DD}	—	28	50	μA
V _{IL1}	输入/输出口、TMR0, TMR1 和 $\overline{\text{INT}}$ 的低电平 输入电压	—	—	0	—	0.3 V _{DD}	V
V _{IH1}	输入/输出口、TMR0, TMR1 和 $\overline{\text{INT}}$ 的高电平 输入电压	—	—	0.7V _{DD}	—	V _{DD}	V
V _{IL2}	低电平输入电压(RES)	—	—	0	—	0.4 V _{DD}	V
V _{IH2}	高电平输入电压(RES)	—	—	0.9 V _{DD}	—	V _{DD}	V
V _{LCD}	LCD 最高电压	—	—	0	—	V _{DD}	V
V _{LVR}	低电压复位	—	—	2	2.1	2.2	V
V _{LVD}	低电压检测	—	—	2.2	2.3	2.4	V
I _{OL1}	输入/输出口逻辑输出 灌电流	3V	V _{OL} =0.1V _{DD}	4	8	—	mA
		5V		10	20	—	mA
I _{OH1}	输入/输出口逻辑输出 源电流	3V	V _{OH} =0.9V _{DD}	-2	-4	—	mA
		5V		-5	-10	—	mA

I _{OL2}	LCD Common 和 Segment 电流	3V	V _{OL} =0.1V _{DD}	210	420	—	μA
		5V		350	700	—	μA
I _{OH2}	LCD Common 和 Segment 电流	3V	V _{OH} =0.9V _{DD}	-80	-160	—	μA
		5V		-180	-360	—	μA
R _{PH}	输入/输出口的上拉电阻	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
充电泵和稳压器							
V _{CHP1}	输入电压	—	充电泵打开	2.2	—	3.6	V
			充电泵关闭	3.7	—	5.5	V
V _{REGO}	输出电压	—	无负载	3	3.3	3.6	V
V _{REGDP1}	稳压器输出下降 (与无负载时比较)	—	V _{DD} =3.7V~5.5V 电压泵关闭 电流≤10mA	—	100	—	mV
V _{REGDP2}		—	V _{DD} =2.4V~3.6V 电压泵打开 电流≤6mA	—	100	—	mV
双积分 AD 转换器, 放大器和带隙							
V _{RFGO}	参考发生器输出	—	@3.3V	1.45	1.5	1.55	V
V _{RFGTC}	参考发生器温度系数	—	@3.3V	—	50	—	Ppm/ C
V _{ADOFF}	输入偏移范围	—	—	—	500	800	μV
V _{ICMR}	共模输入范围	—	放大器, 空载	0.2	—	V _{REGO} -1	V
		—	积分器, 空载	1	—	V _{REGO} -0.2	V

交流电气特性
T_a=25℃

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 (RC 振荡)	—	2.2V~5.5V	400	—	4000	kHz
	系统时钟 (晶体振荡)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
f _{INRC}	内部 RC 振荡	3V	—	—	12	—	kHz
		5V		—	15	—	kHz
f _{TIMER}	定时器输入频率 (TMR0/TMR1)	—	2.2V~5.5V	0	—	4000	kHz
t _{WDTOSC}	看门狗振荡器周期	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t _{RES}	外部复位低电平脉宽	—	—	1	—	—	μs
t _{SST}	系统启动延迟时间	—	上电或从 HALT 状态唤醒	—	1024	—	t _{SYS}
t _{LVR}	低电压复位延迟时间	—	—	0.25	1	2	ms
t _{INT}	中断脉冲宽度	—	—	1	—	—	μs

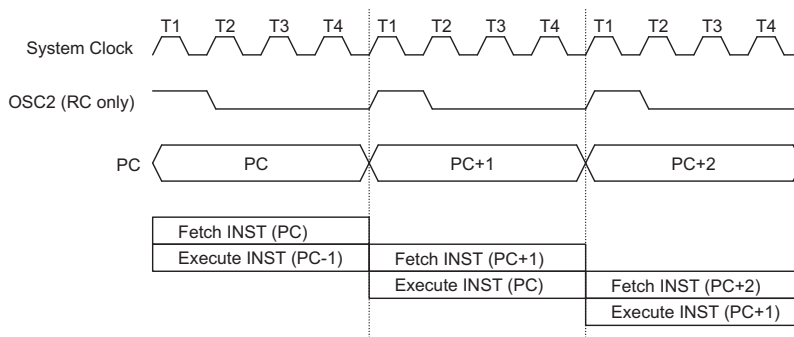
 注: t_{SYS} = 1/f_{SYS}

系统功能说明

指令执行时序

单片机的系统时钟由晶振或外部 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

12 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 4096 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(程序计数器低字节寄存器)赋值、子程序调用、初始化复位、中断或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											
	*11	*10	*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	0	0
外部中断	0	0	0	0	0	0	0	0	0	1	0	0
定时/计数器 0 溢出	0	0	0	0	0	0	0	0	1	0	0	0
定时/计数器 1 溢出	0	0	0	0	0	0	0	0	1	1	0	0
ADC 中断	0	0	0	0	0	0	0	1	0	0	0	0
条件跳跃	Program Counter+2											
装载 PCL	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
跳转，子程序调用	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
子程序返回	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

注： *11 ~ *0：程序计数器位
#11 ~ #0：指令代码位

S11 ~ S0：堆栈寄存器位
@7 ~ @0：PCL 位

程序存储器 — EPROM

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 4096×15 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

该地址为外部中断服务程序保留。当 $\overline{\text{INT}}$ 引脚有触发信号输入，如果中断允许且堆栈未滿，则程序会跳转到 004H 地址开始执行。

- 地址 008H

该地址为定时/计数器 0 服务程序保留。当定时/计数器 0 溢出，如果中断允许且堆栈未滿，则程序会跳转到 008H 地址开始执行。

- 地址 00CH

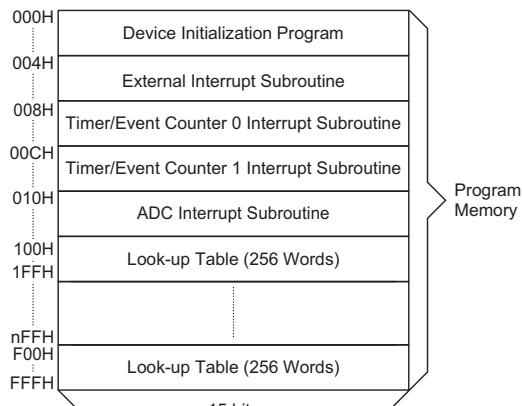
该地址为定时/计数器 1 中断服务程序保留。当定时/计数器 1 溢出，如果中断允许且堆栈未滿，则程序会跳转到 00CH 地址开始执行。

- 地址 010H

该地址为 AD 转换中断服务程序保留。当 AD 转换中断发生，如果中断允许且堆栈未滿，则程序会跳转到 010H 地址开始执行。

- 表格区

ROM 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]” (查最后一页表格)，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。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。



Note: n ranges from 1 to E

程序存储器

指令	表格区											
	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC[m]	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL[m]	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

表格区

注：*11 ~ *0 : 表格地址位

P11 ~ 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 Program Counter 的值。HT46R73D-1 有 4 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(Program Counter)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 4 个返回地址会被保留。

数据存储器 — RAM

数据存储器的 Bank0 由 123×8 位组成，分为两个功能区间：特殊功能寄存器(27×8)和通用数据存储器(96×8)，数据存储器单元大多数是可读/写的，但有些是只读的。在所有 Bank 中，特殊功能寄存器是重叠的。

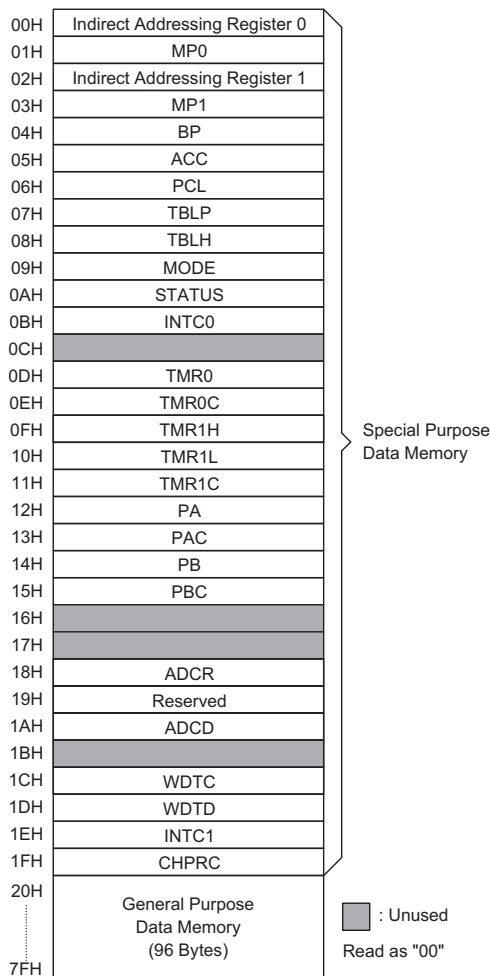
在 20H 之前的空间保留给系统以后扩展使用，读取这些地址的返回值为“00H”。通用数据寄存器地址从 20H 到 7FH，用来存储数据和控制信息。所有的数据存储器单元都能直接执行算术、逻辑、递增、递减和循环操作。除了一些特殊位外，数据存储器的每一位都可通过“SET[m].i”置位或由“CLR[m].i”复位。而且都可以通过间接寻址指针(MP0：01H/MP1：03H)进行间接寻址。

Bank1 是 LCD 数据存储区，将 BP 值设为“01H”，便可通过间接寻址指针 MP1 访问 Bank1。当 BP 值为“01H”，通过 MP1 来间接读写地址从 40H 到 50H 的数据存储空间，将会改变 Bank1 中内容。不管 BP 为何值，直接寻址数据寄存器只会改变 Bank0 中内容。

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] 或 [02H] 的读/写操作，都是访问由 MP0(01H)/MP1(03H)或所指向的 RAM 单元。间接读取地址 00H 或 02H 得到的值为 00H，间接写入此地址，不会产生任何操作。间接寻址指针 MP0 和 MP1 是 7 位寄存器。

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 7 位寄存器 MP0 只能用于寻址数据存储器，而 MP1 能用于寻址数据存储器 and LCD 显示存储器。



数据存储器

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。

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, 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, 读出为“0”

STATUS (0AH) 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 和 PDF 标志只受看门狗溢出、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

HT46R73D-1 提供一个外部中断、两个内部定时/计数器中断、一个 AD 转换中断。中断控制寄存器 0(INTC0: 0BH)和中断控制寄存器 1(INTC1: 1EH)包含了中断控制位和中断请求标志，中断控制位用来设置中断允许/禁止。

只要有中断子程序被服务，其余的中断全部都被自动禁止(通过清除 EMI 位)，这种做法的目的在于防止中断嵌套。这时如果有其它中断发生，只有中断请求标志会被记录下来。如果在中断服务程序中有另一个中断需要响应，程序员可以置位 EMI、INTC0 和 INTC1 所对应的位，以便进行中断嵌套。如果堆栈已满，则中断并不会被响应，一直到堆栈指针(SP)发生递减后才会响应。如果需要中断立即得到响应，应避免堆栈饱和。

位	符号	功 能
0	EMI	总中断控制位(1=允许; 0=禁止)
1	E EI	外部中断控制位(1=允许; 0=禁止)
2	ETOI	定时/计数器 0 中断控制位(1=允许; 0=禁止)
3	ETI1	定时/计数器 1 中断控制位(1=允许; 0=禁止)
4	EIF	外部中断请求标志(1=有; 0=无)
5	TOF	定时/计数器 0 中断请求标志(1=有; 0=无)
6	T1F	定时/计数器 1 中断请求标志(1=有; 0=无)
7	—	只作内部测试用 使用时必须写入 ‘0’; 否则会发生不可预知的错误

INTC0 (0BH) 寄存器

位	符号	功 能
0	EADI	A/D 转换中断控制位(1=允许; 0=禁止)
1~3,5~7	—	未用, 读出为 “0”
4	ADF	A/D 转换中断请求标志(1=有; 0=无)

INTC1 (1EH) 寄存器

所有的中断都具有唤醒能力。当有中断被服务，系统会将程序计数器的内容压入堆栈，然后再跳转至中断服务程序的入口。但这时只有程序计数器的内容被压入堆栈，如果其它寄存器和状态寄存器的内容会被中断程序改变，从而会破坏主程序的控制流程的话，程序员应该事先将这些数据保存起来。

外部中断是由 $\overline{\text{INT}}$ 引脚电平变化触发的(可由掩膜设置为上升沿触发、下降沿触发或两者皆可触发)，其中断请求标志位(EIF; INTC0 的第 4 位)会被置位。如果中断允许，且堆栈未满，当发生外部中断时，会产生地址 004H 的子程序调用；而中断请求标志 EIF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。

内部定时/计数器 0 中断是由定时/计数器 0 溢出触发的，其中断请求标志(TOF; INTC0 的第 5 位)会被置位。如果中断允许，且堆栈未满，当发生定时/计数器中断时，会产生地址 008H 的子程序调用；而中断请求标志 TOF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。定时/计数器 1 的中断响应过程与定时/计数器 0 相同，只是其中断请求标志为 T1F (INTC0 的第六位)，中断入口地址为 0CH。

A/D 转换中断是由 A/D 转换完成触发的，其中断请求标志(ADF; INTC1 的第 4 位)会被置位。如果中断允许，且堆栈未满，当发生 A/D 转换中断时，会产生地址 10H 的子程序调用；而中断请求标志位 ADF 和总中断控制位 EMI 会被清除，以禁止其它中断响应。

在执行中断子程序期间，其它的中断请求会被屏蔽，直到执行 RETI 指令或 EMI 和相关中断控制位被置位(当然，此时堆栈未满)。如果要从中断子程序返回，只要执行 RET 或 RETI 指令即可。其中，RETI 指令会自动置位 EMI，以允许中断服务，而 RET 则不会。

如果中断在两个连续的 T2 脉冲的上升沿之间发生，且中断响应允许，那么在下两个 T2 脉冲之间，该中断会被服务。如果同时发生中断请求，其优先级如下表示，这些中断可以通过清除 EMI 位来进行屏蔽。

中断源	优先级	中断向量
外部中断	1	04H
定时/计数器 0 中断	2	08H
定时/计数器 1 中断	3	0CH
A/D 转换中断	4	10H

一旦中断请求标志(T0F、T1F、EIF、ADF)被置位, 会一直保留在 INTC0 和 INTC1 寄存器中, 直到中断被响应或用软件指令清除为止。

建议不要在中断服务程序中使用“CALL”指令来调用子程序。因为中断随时都可能发生, 而且需要立刻给予响应。如果只剩下一层堆栈, 而中断不能被很好地控制, 原先的控制序列很可能因为在中断子程序中执行“CALL”指令而使堆栈溢出, 从而发生混乱。

振荡电路

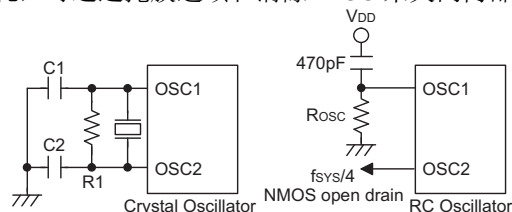
HT46R73D-1 有三种振荡方式: 外部 RC 振荡, 外部晶体振荡和内部 12kHz RC 振荡。外部 RC 振荡和晶振是用来作为系统时钟而内部 12kHz 是用来作为定时器时钟。

在 IDLE 模式, 系统时钟将会停止运行, 但若 IRC 允许 (IRCC=0), 内部 RC 振荡将会继续运行。在 HALT 模式, 若 IRC 禁止 (IRCC=1), 系统时钟和内部 RC 振荡都会停止运行, 但若看门狗打开, 内部 RC 也会继续运行。中断、PA 下降沿、看门狗溢出、定时/计数器溢出或请求标志位的置位 (0→1) 都会使系统离开 IDLE 或 HALT。

如果选用晶体振荡方式, 在 OSC1 和 OSC2 之间需要连接一个晶体, 用来提供晶体振荡器所需的反馈和相移, 除此之外, 不再需要其它外部元件。另外, 在 OSC1 和 OSC2 之间也可使用谐振器来取代晶体振荡器, 但是在 OSC1 和 OSC2 需要多连接两个电容。

如果选用外部 RC 振荡方式, 在 OSC1 与 VSS 之间需要接一个外部电阻, 其阻值为 100kΩ 到 2.4MΩ; 而 OSC2 上会输出系统频率的 4 分频信号, 可用于同步外部逻辑。

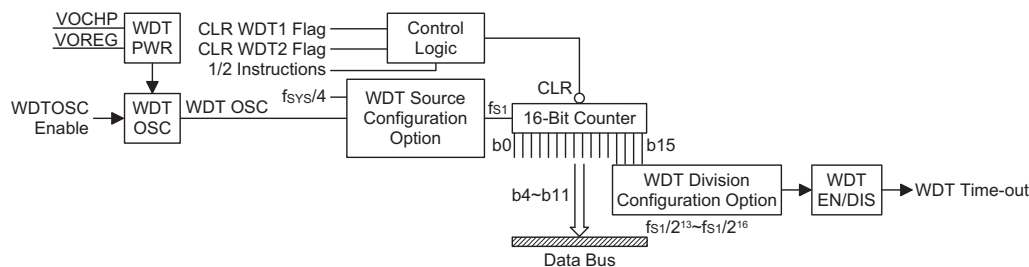
内部 RC 振荡器是一个自由振荡内部 RC 振荡器, 并不需要连接任何外部元件。当系统进入暂停模式时, 系统时钟会停止, 但在 WDT 或 IRC 时钟使能的情况下, 内部 RC 振荡器会继续工作, 其振荡周期大约为 65μs/5V。如果需要降低功耗, 可通过掩膜选项和清除 IRCC 来关闭内部 RC 振荡器。



系统振荡器

看门狗定时器 — WDT

WDT 的时钟来源可由掩膜选项设置为内部 RC 振荡 (内部 RC 振荡, 注意: 在本节中的描述对于 WDT 振荡器同内部 RC 振荡器是同一个振荡器) 或指令时钟 (系统时钟 4 分频)。WDT 主要用来防止程序运行故障和程序跳入一死循环而导致不可预测的结果。WDT 可由掩膜选项设置为打开或关闭, 如果在关闭状态, WDT 计数器还是会同样工作, 只是其计数溢出不会导致器件复位。因此当 WDT 除能时, WDT 计数寄存器还是能被读出来或是被清除。此功能是专门设计, 使软件设计者可以通过获取 WDT 的当前频率值来检测当前外部的温度系数是否发生变化, 以做为是否调整外部模拟信号量的依据。为了达到降低功耗的目地, 可以通过控制寄存器 (WDTC: WDTOSC) 来选择 WDT 功能的使能或除能。



看门狗定时器

位	符号	功 能
0~1	WDTPWR0~WDTPWR1	WDT 电源源选择 01:WDT 电源来自 VOCHP 10:WDT 电源来自于调整器 00/11:推荐使用 VOCHP 做为 WDT 电源源, 以防止干扰导致 WDT 掉电
2~3	WDTOSC0~WDTOSC1	WDT 振荡器的使能位 (WDTOSC1:0) = 01:WDT OSC 除能 10:WDT OSC 使能 00/11:推荐选择 10 使能 WDT OSC
4~7	—	未用

WDTC(1CH)寄存器

注意：如果 WDT 使能及 WDT 时钟来源选为 WDT 时钟，那么 WDTOSC 寄存器的初始值为 (1, 0)，反之被设置为 (0, 1)。

位	符号	功 能
0~7	WDTD0~WDTD7	WDT 计数数据寄存器 此寄存器只读，用于温度系数的调整。

WDTD(1DH)寄存器

WDT 时钟(f_{s1})可以通过内部分频以得到一个更久的溢出周期。通过配置选项选择不同的分频系数以得到 $2^{13} \sim 2^{16}$ 之间的 4 个选择。

有两个寄存器与 WDT 相关：WDTC 及 WDTD。其中 WDTC 是控制寄存器，用来选择 WDT 电源的来源及 WDT 功能是否使能。WDTD 是 WDT 计数寄存器。

WDTPWR 位为 WDT 电源源选择位，默认的 WDT 电源源为 VOCHP。而使用调整器输出电压做为 WDT 电源主要是为了获得一个精确的温度调整系数。当然在此应用前应保证调整器模块是处于工作状态中。WDTOSC 位是用来选择是否使能 WDT OSC (12kHz)。如果没有选用 WDT OSC，那么应该对它除能来降低功耗。当 WDT 振荡器关闭，不管 IRCC 寄存器如何设置，WDT 时钟都会关闭。当 WDT 振荡器打开时，通过 IRCC 寄存器决定掉电模式的情形。

如果 WDT 时钟源为内部 RC 振荡(RC 振荡周期一般为 65 μ s)，该频率可再经过 $2^{12} \sim 2^{15}$ (由掩膜选项设置)的分频。最小的 WDT 溢出周期大约是 300mS~600mS。但溢出时间会因为温度、VDD 以及芯片参数的变化而变化。通过设置 WDT 的掩膜选项，可以得到更长的溢出周期。如果 WDT 的溢出时间选为 2^{15} 分频，最大的溢出时间可达到 2.3s~4.7s(分频系数为 $2^{15} \sim 2^{16}$)。

如果 WDT 的时钟源为指令时钟，则在 HALT 状态时，WDT 会停止计数而失去保护功能；此时只能靠外部逻辑复位来重新启动系统。如果系统运用在强干扰的环境中，建议选用内部 RC 振荡器，因为 HALT 模式会使系统时钟停止，看门狗也就失去了保护的功能。

在正常运行时，WDT 溢出会使系统复位并置位 TO 标志；但在 HALT 模式下，WDT 溢出只产生“热复位”，只有程序计数器 Program Counter 和堆栈指针 SP 被复位。要清除 WDT 的值可以有三种方法：外部复位(低电平输入到 \overline{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。

蜂鸣器输出

蜂鸣器功能提供一个可变频率输出的功能，以适用于像压电蜂鸣器驱动、或其它需要精确频率输出的场合。BZ 和 $\overline{\text{BZ}}$ 是一对和输入/输出引脚 PA0 与 PA1 共用的蜂鸣器输出。可通过掩膜选项选择 PA0 和 PA1 都是普通输入/输出、BZ 和 $\overline{\text{BZ}}$ 输出、以及单一 BZ 输出而 PA1 作为普通输入/输出。要注意的是 $\overline{\text{BZ}}$ 引脚是 BZ 引脚的反向输出，两者配合可产生更强的输出功率驱动蜂鸣器等器件。

蜂鸣器时钟来源为内部时钟 f_s 经过分频器得到，其分频系数可通过掩膜选项选择为 $f_s/2^2$ 到 $f_s/2^9$ 之间。

f_s 可通过掩膜选项选择为内部 RC 振荡源或系统时钟四分频。需要注意的是蜂鸣器输出频率是完全由掩膜选项控制，包括其时钟 f_s 来源和分频系数，没有特殊功能寄存器与蜂鸣器频率有关。

如果掩膜选项确定 PA0 和 PA1 是 BZ 和 $\overline{\text{BZ}}$ 蜂鸣器输出，必须将 PA 控制寄存器 PAC0 和 PAC1 设为输出。若要蜂鸣器正常工作，需要将 PA0 置为高；若清零，则 PA0 和 PA1 都输出低电平。PA 寄存器的 PA0 位可作为 BZ 和 $\overline{\text{BZ}}$ 蜂鸣器输出的控制开关。注意 PA 寄存器的 PA1 位不能控制 $\overline{\text{BZ}}$ 蜂鸣器引脚 PA1。

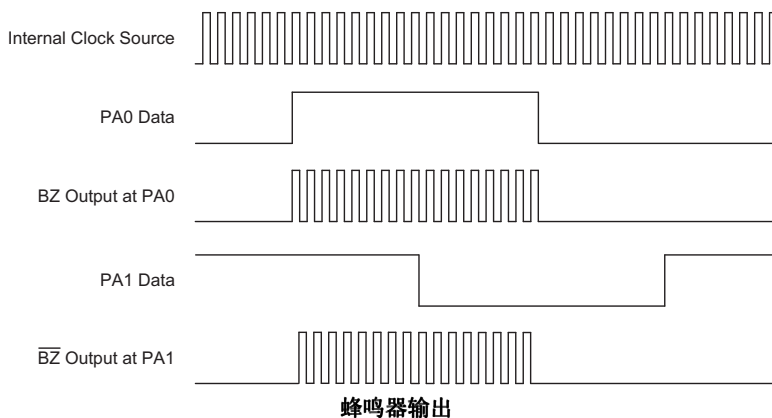
如果掩膜选项确定 PA0 为 BZ 输出，那么 PA1 可被用作普通输入/输出。蜂鸣器正常工作，需要通过 PAC0 将 PA0 设为输出。若 PA 寄存器中 PA0 置为高，则引脚 PA0 可输出蜂鸣器驱动输出；若清为零，则引脚 PA0 保持低电平。这样输出数据位 PA0 被用来作为 BZ 输出开关控制。尽管掩膜选项中选择为 BZ 输出，但 PA 控制寄存器 PAC0 置为高，则 PA0 可用作普通的输入口。

注意，若将相应口设为输入，无论掩膜选项中选择如何，都会作为普通输入使用。这样可使得这些引脚既可作为蜂鸣器输出，也可作为输入使用；实际功能可通过程序修改端口控制寄存器完成。

PAC 寄存器 PAC.0	PAC 寄存器 PAC.1	PA 寄存器 PA.0	PA 寄存器 PA.1	输出功能
0	0	0	X	PA0=0, PA1=0
0	0	1	X	PA0=BZ, PA1= $\overline{\text{BZ}}$
0	1	0	X	PA0=0, PA1=Input
0	1	1	X	PA0=BZ, PA1=Input
1	0	0	X	PA0=Input, PA1=0
1	1	X	X	PA0=Input, PA1=Input

PA0/PA1 引脚功能

注意：“X”表示任意值



注意：上图中 PA0 和 PA1 都被选择为 BZ 和 $\overline{\text{BZ}}$ 输出，其端口控制寄存器都被设置为输出。PA1 位对蜂鸣器输出不起作用。

暂停模式 — HALT

暂停模式是由 HALT 指令来实现的，暂停模式时系统状态如下：

- 系统振荡器停振，但内部 RC 振荡器会继续振荡(如果选择内部 RC 振荡)。
- RAM 和寄存器内容保持不变。
- WDT 被清除并重新开始计数(如果 WDT 时钟来源为 RC 振荡)。
- 所有输入/输出口都保持其原有状态。
- 置位 PDF 标志，清除 TO 标志。
- 如果 IRC 时钟打开，LCD 驱动器仍然运行；IRCC=0，LCD 在 Halt 模式下打开。

以下操作可以使系统离开暂停模式或空闲模式：外部复位、中断、PA 口下降沿信号或看门狗定时器溢出。其中，外部复位会使系统初始化，WDT 溢出则会发生“热复位”。通过检测 TO 和 PDF 标志，即可了解系统复位的原因。PDF 标志可由系统上电或执行“CLR WDT”指令清除，由 HALT 指令置位。TO 标志由 WDT 溢出置位，同时产生唤醒，但只有程序计数器 Program Counter 和堆栈指针 SP 被复位，其它都保持其原有的状态。

PA 口唤醒和中断唤醒可做为正常运行的继续。PA 口的每一位都可以由掩膜选项设置为唤醒功能。如果是由输入/输出口唤醒，程序会从下一条指令开始运行。如果是由中断唤醒，可能会发生两种情况：如果中断禁止或中断允许但堆栈已满，程序将会从下一条指令开始运行；如果中断允许且堆栈未滿，则会产生一般的中断响应。如果在进入 HALT 模式之前，中断请求标志位已被置“1”，则中断唤醒功能被禁止。

当发生唤醒，系统需要额外花费 $1024t_{SYS}$ (系统时钟周期)的时间，才能重新正常运行，也就是说，唤醒之后会插入一个等待周期。如果唤醒是由中断产生的话，则实际中断子程序的执行会延迟一个以上的周期。如果唤醒导致下一条指令执行，那么在等待周期执行完成之后，会立即执行该指令。

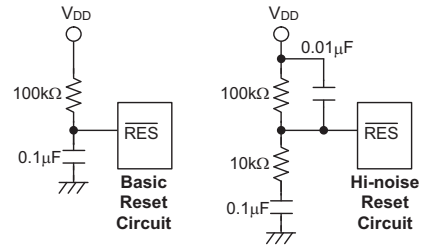
为减小功耗，在进入暂停模式之前，应小心处理所有的输入/输出口状态。

复位

总共有三种方法会产生初始复位:

- 正常运行时由 $\overline{\text{RES}}$ 引脚发生复位。
- 在暂停模式下由 $\overline{\text{RES}}$ 引脚发生复位。
- 正常运行时由看门狗定时器溢出发生复位。

暂停模式和空闲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”, 只有程序计数器 Program Counter 和堆栈指针 SP 被复位, 而系统其它部分都保持原有状态。在其它复位状态下, 某些寄存器不会改变。在初始复位时, 大部分寄存器会复位成初始的状态。通过检测 PDF 和 TO 标志, 即可判断出各种不同的复位原因。



复位电路

注意: 大多数应用中使用基本复位电路即可, 如方案中外部噪音或杂讯比较厉害, 就请选用高抗杂讯电路

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 $\overline{\text{RES}}$ 发生复位
0	1	暂停模式下 $\overline{\text{RES}}$ 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

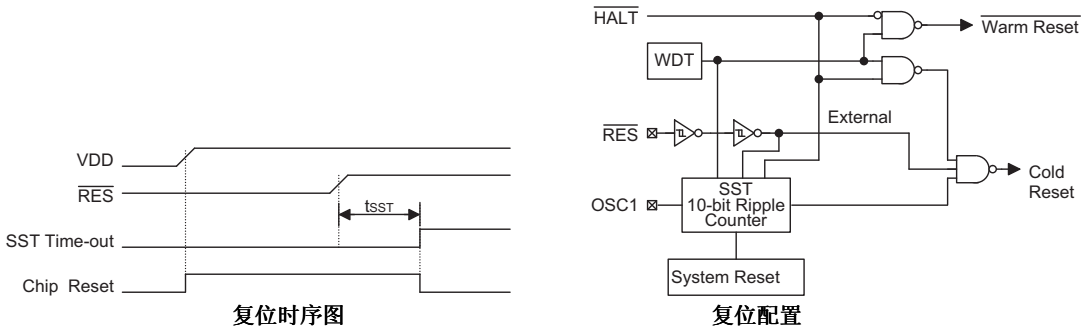
为了保证系统振荡器起振并稳定运行, 系统复位(包括上电复位、WDT 溢出或由 $\overline{\text{RES}}$ 端复位)或由暂停状态唤醒时, 系统启动定时器(SST)提供了一个额外的延迟时间, 共 1024 个系统时钟周期。

系统复位时, SST 会被加在复位延时中; 由暂停模式唤醒也会加入 SST 延迟。

系统复位(包括上电复位、正常运行时 WDT 溢出或由 $\overline{\text{RES}}$ 端复位)需要额外增加一个加载掩膜选项 (Option) 的时间。

系统复位时各功能单元的状态如下所示:

Program Counter	000H
中断	禁止
预分频器、分频器	清除
WDT	清除, 在主系统复位后, WDT 开始计数
定时/计数器	停止
输入/输出口	输入模式
堆栈指针 SP	指向堆栈顶部



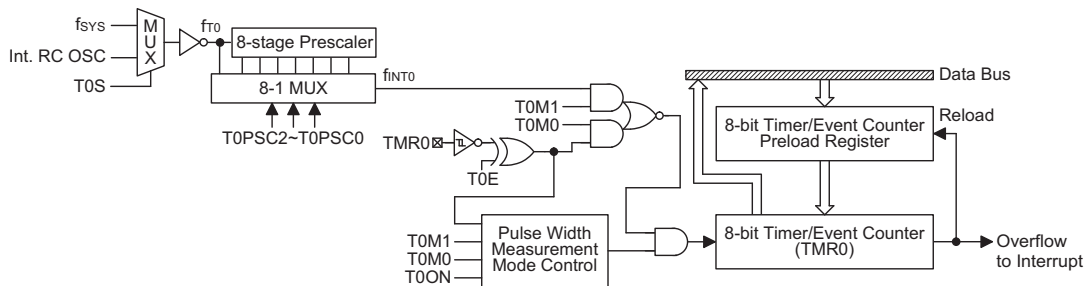
有关寄存器的状态如下：

寄存器	上电复位	正常运行期间		暂停模式	
		WDT 溢出	RES 端复位	RES 端复位	WDT 溢出*
MP0	1xxx xxxx	1uuu uuuu	1uuu uuuu	1uuu uuuu	1uuu uuuu
MP1	1xxx xxxx	1uuu uuuu	1uuu uuuu	1uuu uuuu	1uuu uuuu
BP	---- --0	---- --0	---- --0	---- --0	---- --u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
Program Counter	0000H	0000H	0000H	0000H	0000H
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
MODE	xx0x 01xx	xx0x 01xx	xx0x 01xx	xx0x 01xx	uuuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
PBC	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
ADCR	0000 x000	0000 x000	0000 x000	0000 x000	uuuu xuuu
ADCD	---- -111	---- -111	---- -111	---- -111	---- -uuu
WDTC	---- ss01	---- ss01	---- ss01	---- ss01	---- uuuu
WDTD	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
INTC1	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
CHPRC	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

- 注： 1. “*” 表示“热复位”；
 2. “u” 表示不变化；
 3. “x” 表示不确定；
 4. “s” 要特别留意，它由选项表格决定(详情请看 WDT 部分的内容)

定时/计数器

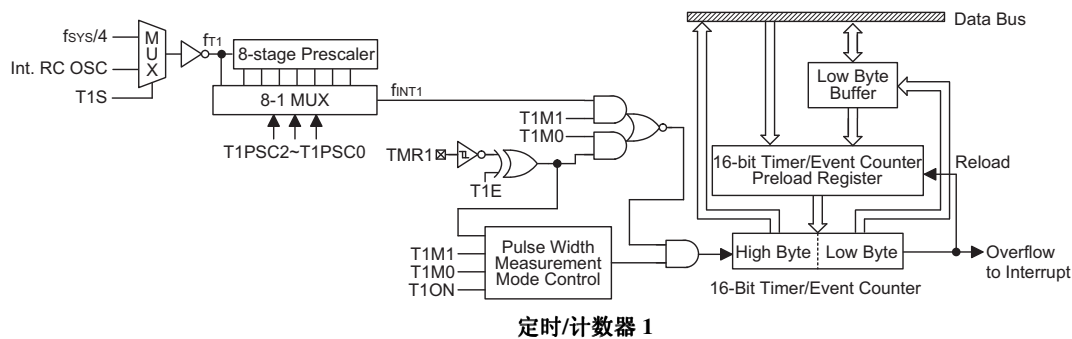
HT46R73D-1 提供两个定时/计数器。定时/计数器 0 是一个 8 位可编程向上计数器，它的时钟来源可以是外部信号输入或内部时钟。如果用做内部定时器方式，则时钟源为 f_{SYS} 或内部 RC。定时/计数器 1 是一个 16 位可编程向上计数器，其时钟来源可以是外部信号输入或内部时钟。如果用做内部定时器方式，则可通过特殊功能寄存器选择时钟源为 $f_{SYS}/4$ 或内部 RC。外部信号输入可以用来计数外部事件、测量时间间隔、测量脉冲宽度或产生一个精确的时基信号。



定时/计数器 0

位	符号	功能
0 1 2	TOPSC0 TOPSC1 TOPSC2	定义预分频器级数, TOPSC2, TOPSC1, TOPSC0= 000: $f_{INT0}=f_{T0}$ 001: $f_{INT0}=f_{T0}/2$ 010: $f_{INT0}=f_{T0}/4$ 011: $f_{INT0}=f_{T0}/8$ 100: $f_{INT0}=f_{T0}/16$ 101: $f_{INT0}=f_{T0}/32$ 110: $f_{INT0}=f_{T0}/64$ 111: $f_{INT0}=f_{T0}/128$
3	TOE	定义定时/计数器 TMR0 的触发方式 在事件计数模式 (TOM1, TOM0) = (0, 1): 1: 在下降沿计数 0: 在上升沿计数 在脉冲宽度测量模式 (TOM1, TOM0) = (1, 1): 1: 在上升沿开始计数, 下降沿停止计数 0: 在下降沿开始计数, 上升沿停止计数
4	TOON	打开/关闭定时/计数器(0=关闭, 1=打开)
5	TOS	TMR0 时钟来源选项 (0= f_{SYS} , 1=内部 RC)
6 7	TOM0 TOM1	定义工作模式(TOM1, TOM0): 01=事件计数模式(外部时钟) 10=定时模式(内部时钟) 11=脉宽测量模式(外部时钟) 00=未用

TMR0C (0EH) 寄存器



定时/计数器 1

位	符号	功能
0 1 2	T1PSC0 T1PSC1 T1PSC2	定义预分频器级数, T1PSC2, T1PSC1, T1PSC0= 000: $f_{INT1}=f_{T1}$ 001: $f_{INT1}=f_{T1}/2$ 010: $f_{INT1}=f_{T1}/4$ 011: $f_{INT1}=f_{T1}/8$ 100: $f_{INT1}=f_{T1}/16$ 101: $f_{INT1}=f_{T1}/32$ 110: $f_{INT1}=f_{T1}/64$ 111: $f_{INT1}=f_{T1}/128$
3	T1E	定义定时/计数器 TMR1 的触发方式 在事件计数模式 (T1M1, T1M0) = (0, 1): 1: 在下降沿计数 0: 在上升沿计数 在脉冲宽度测量模式 (T1M1, T1M0) = (1, 1): 1: 在上升沿开始计数, 下降沿停止计数 0: 在下降沿开始计数, 上升沿停止计数
4	T1ON	打开/关闭定时/计数器(0=关闭, 1=打开)
5	T1S	TMR0 时钟来源选项 (0= $f_{SYS}/4$, 1=内部 RC)
6 7	T1M0 T1M1	定义工作模式(T1M1, T1M0): 01=事件计数模式(外部时钟) 10=定时模式(内部时钟) 11=脉宽测量模式(外部时钟) 00=未用

TMR1C (11H) 寄存器

有两个与定时/计数器 0 有关的寄存器, TMR0(0DH)和 TMR0C(0EH)。写入 TMR0 会将初始值装入到定时/计数器 0 的预置寄存器中, 而读 TMR0 则会取得定时/计数器 0 的内容。TMR0C 是定时/计数器 0 控制寄存器, 用来定义定时/计数器 0 一些选项。有三个与定时/计数器 1 有关的寄存器, TMR1H(0FH)、TMR1L(10H)和 TMR1C(11H)。写 TMR1L 只能将数据写到低字节缓冲器 (8 位), 而写 TMR1H 会把指定数据和低字节缓冲器的数据的数据分别写到 TMR1H 和 TMR1L 预置寄存器中。

定时/计数器 1 预置寄存器的内容只有在写入 TMR1H 时才会被改变而写 TMR1L 不会改变预置寄存器的值。读取 TMR1H 会把 TMR1H 的内容送至目标单元, 而 TMR1L 的值被送至低字节缓冲器中。读 TMR1L 将读取低字节缓冲器的值。换言之, 定时/计数器的低字节内容是无法直接读取的。必须先读取 TMR1H, 将定时/计数器 1 的低字节内容送至低字节缓冲器。TMR1C 是定时/计数器 1 控制寄存器, 用来定义定时/计数器 1 一些选项。

T0M0、T0M1(TMR0C)和 T1M0、T1M1(TMR1C)用来定义定时/计数器的工作模式。外部事件计数模式是用来记录外部事件的，其时钟来源为外部（TMR0、TMR1）引脚输入。定时器模式是一个常用模式，其时钟来源为内部时钟。脉宽测量模式可以测量 TMR0 或 TMR1 引脚高/低电平的脉冲宽度，其时钟来源为内部时钟。

无论是定时模式还是外部事件计数模式，一旦开始计数，定时/计数器 0(1)会从寄存器当前值向上计到 FFH(FFFFH)。一旦发生溢出，定时/计数器会从预置寄存器中重新加载初值，并开始计数；同时置位中断请求标志(T0F; INTC0 的第 5 位, T1F; INTC0 的第 6 位)。在脉宽测量模式，当 T0ON/T1ON 与 T0E/T1E 是 1 时，只要 TMR0(TMR1)引脚有一个上升沿信号(如果 T0E/T1E 是 0，则为下降沿信号)，定时/计数器就会开始计数，直到 TMR0(TMR1)脚电平恢复，同时 T0ON/T1ON 被清零。即使有新的脉冲产生，上次测量的结果也会保存在寄存器中。换句话说，一次只能测量一个脉冲宽度。重新置位 T0ON/T1ON 后，可以继续测量。注意，在该模式下，定时/计数器是跳变触发而不是电平触发。当计数器溢出时，定时/计数器会从预置寄存器中重新加载初值，并置位中断请求标志，这与其它两种模式一样。

要启动计数器，只要置位 T0ON/T1ON(TMR0C/TMR1C 的第 4 位)。在脉宽测量模式下，T0ON/T1ON 在测量结束后会被自动清除；但在另外两种模式中，T0ON/T1ON 只能由指令来清除。定时/计数器 0/1 的溢出可以做为唤醒信号。不管是什么模式，只要写 0 到 ET0I 或 ET1I 即可禁止定时/计数器中断服务。

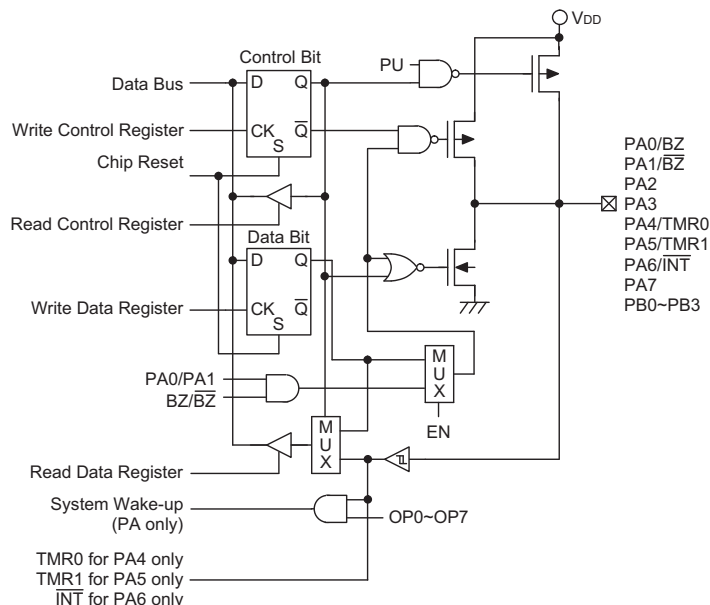
在定时/计数器停止计数时，写数据到定时/计数器的预置寄存器中，同时会将该数据写入到定时/计数器。但如果在定时/计数器运行时这么做，数据只能写入到预置寄存器中，直到发生溢出时才会将数据从预置寄存器加载到定时/计数器寄存器。

读取定时/计数器时，计数会被停止，以避免发生错误；计数停止会导致计数错误，程序员必须注意到这一点。强烈建议先将数据写入 TMR0 和 TMR1 寄存器中，再打开相应的定时/计数器。

TMR1C 的第 0~2 位用于定义定时/计数器 1 的预分频系数。

输入/输出口

HT46R73D-1 有 12 位双向输入/输出口，记为 PA 和 PB，其分别对应 RAM 地址[12H]和[14H]，所有端口都可以进行输入/输出操作。输入时，端口没有锁存功能，输入信号必须在 MOV A, [m](m=12H 或 14H) 指令的 T2 上升沿到来前准备好；输出时，端口有锁存功能，端口上的数据会保持不变直到执行下一个写入操作。



输入/输出口

每个输入/输出口都有一个控制寄存器(PAC, PBC)，用来控制输入/输出状态。利用控制寄存器，可对 CMOS 输出、带或不带上拉电阻的斯密特触发输入通过软件动态地进行改变。做为输入时，对应的控制寄存器应设置为“1”。输入信号来源也取决于控制寄存器，如果控制寄存器的值为“1”，那么读取的是引脚状态；如控制寄存器的值为“0”，则读取的是内部锁存器的值。后者可能会在‘读-修改-写’指令中发生。

做为输出时，只能采用 CMOS 输出。控制寄存器对应 RAM 地址 13H、15H。

系统复位之后，这些输入/输出口会是高电平或浮空状态(由上拉电阻选项决定)。每一个输入/输出锁存位都能用“SET [m].i”或“CLR [m].i”指令置位或清除(m=12H 或 14H)。

有些指令会先输入数据，然后进行输出操作。例如：“SET [m].i”，“CLR [m].i”，“CPL [m]”，“CPLA[m]”这些指令会先将整个端口状态读入 CPU 中，接着执行所定义的运算(位操作)，然后再将结果写入锁存器或累加器中。

PA 的每一个口都具有唤醒系统的能力。

所有的输入/输出口都有上拉电阻选项。一旦选择了上拉电阻选项，输入/输出口就加上了上拉电阻。如果不选择上拉电阻，必须注意在输入模式下，不带上拉电阻的 I/O 口会产生浮空状态。

PA0、PA1、PA4、PA5 和 PA6 分别和 BZ、 \overline{BZ} 、TMR0、TMR1 和 \overline{INT} 共用引脚。

PA0、PA1 与 BZ、 \overline{BZ} 共用引脚，如果选择 BZ/ \overline{BZ} 功能，则 PA0/PA1 在输出模式时的输出信号将是蜂鸣器信号，而在输入模式始终保持其原来的功能。一旦选择 BZ/ \overline{BZ} 功能，蜂鸣器的输出信号只受 PA0 数据寄存器控制。

PA0/PA1 的输入/输出功能如下所示:

PA0 I/O	I	I	O	O	O	O	O	O	O	O
PA1 I/O	I	O	I	I	I	O	O	O	O	O
PA0 模式	X	X	C	B	B	C	B	B	B	B
PA1 模式	X	C	X	X	X	C	C	C	B	B
PA0 数据	X	X	D	0	1	D ₀	0	1	0	1
PA1 数据	X	D	X	X	X	D ₁	D	D	X	X
PA0 引脚状态	I	I	D	0	B	D ₀	0	B	0	B
PA1 引脚状态	I	D	I	I	I	D ₁	D	D	0	B

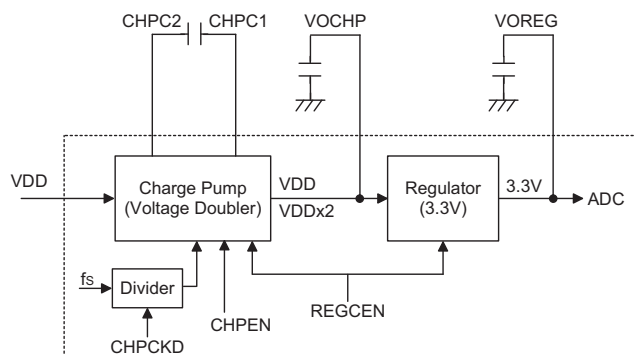
注: “I” 输入; “O” 输出
 “D、D₀、D₁” 数据
 “B” 蜂鸣器选项, BZ 或 \overline{BZ}
 “X” 任意值
 “C” CMOS 输出

建议用软件将未使用和没有外接的输入/输出口设置为输出模式, 以防止这些端口在输入浮空时增加系统的功耗。

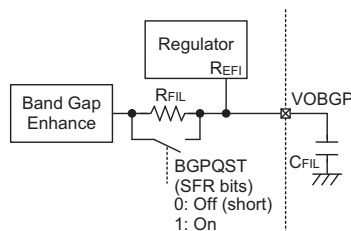
充电泵和稳压器

HT46R73D-1 提供一个充电泵和一个稳压器。

充电泵可由程序控制打开或关闭, 它由 VDD 作为输入, 可输出 2 倍 VDD 电压 (即 $2 \times VDD$)。稳压器用于产生稳定的 3.3V 电压, 作为 AD 转换电路和外部传感器电源, 也可供给其它相关电路使用。为确保稳压器输出为稳定的 3.3V, 必须要使充电泵输出大于 3.6V。



另外, 还提供了一个 1.5V 的 band gap 电压发生器, 做为低温条件下的参考电压。这个电压相当于在零校准时引进的一个直流参考电压, R_{FIL} 等效值约为 100 kΩ, C_{FIL} 的推荐值为 10μF。



注: VOBGP 为内部信号, 外部仅需要连接器件 C_{FIL}

有一寄存器 CHPRC 与此模块相关。它是充电泵/稳压器控制寄存器，可以控制充电泵和稳压器的打开与关闭，以及设置充电泵时钟分频系数。

位	符号	功能
0	REGCEN	打开或关闭充电泵/稳压器（1=打开；0=关闭）
1	CHPEN	打开或关闭充电泵（1=打开；0=关闭） 注意：若 REGCEN=0，则此位无效
2	BGPQST	Band gap 电压快速启动功能 0:电阻被短路，快速启动 1:电路接入，正常的 RC 模式 为了保证快速启动成功，每次 REGCEN 位从 0 到 1，BGPQST 位也必执行从 0 到 1 的动作，且 0 状态最少保持 2ms 时间。
3~7	CHPCKD0~ CHPCKD4	充电泵时钟分频，此 5 位可产生 1~32 的分频系数 其等式：充电泵时钟 = $(f_{SYS}/16) / (CHPCKD+1)$

CHPRC (1FH) 寄存器

CHPCKD4~0 位用于设置充电泵工作时钟分频。实际频率计算公式为：

实际充电泵时钟频率 = $(f_{SYS}/16) / (CHPCKD+1)$ 。

建议充电泵工作频率为 20kHz，应用程序需要设置正确值得到预期的频率。例如：若时钟为 4MHz 时，CHPCKD 应为 11；若时钟为 2MHz 时，CHPCKD 应为 5。

REGCEN	CHPEN	电压泵	VOCHP 引脚	稳压器	VOREG 引脚	OPA ADC	描述
0	X	OFF	V _{DD}	OFF	高阻态	关闭	整个模块关闭，OPA/ADC 掉电
1	0	OFF	V _{DD}	ON	3.3V	打开	V _{DD} 电压高于 3.6V (V _{DD} > 3.6V)
1	1	ON	2 × V _{DD}	ON	3.3V	打开	V _{DD} 电压低于 3.6V (V _{DD} = 2.2V 到 3.6V)

寄存器 CHPRC 中 REGCEN 位是充电泵/稳压器打开/关闭位。若关闭，则充电泵和稳压器都会被关闭以降低功耗。当 REGCEN=0，此模块将进入掉电模式并忽略 CHPEN 位，AD 转换和 OPA 也会关闭以降低功耗。

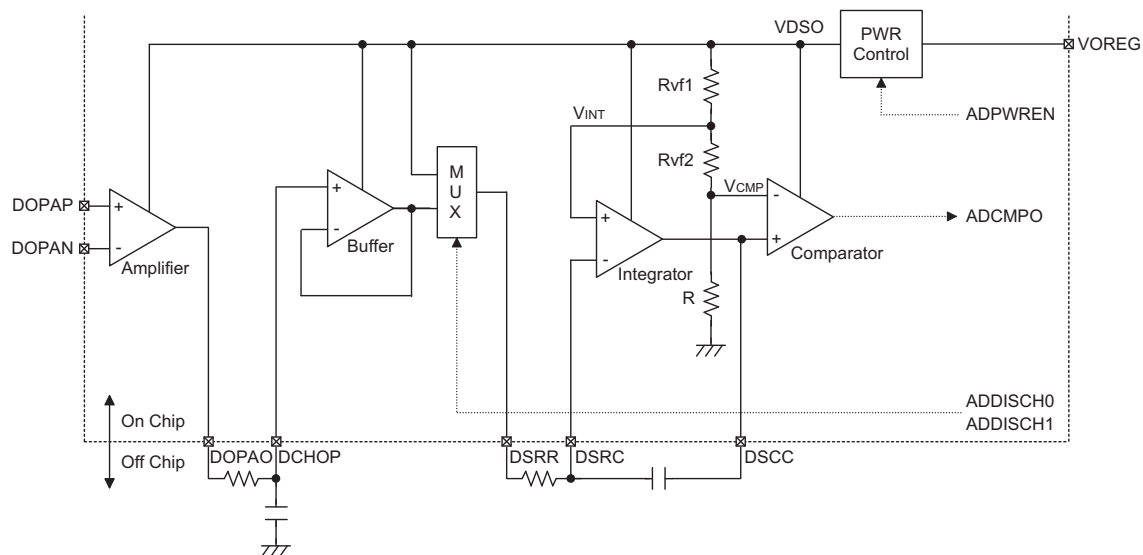
当 REGCEN=1，稳压器将会使能。若 CHPEN=1，充电泵使能并且输出电压等于输入电压两倍 (2V_{DD})，此电压作为稳压器输入。若 CHPEN=0，充电泵除能并且输出电压等于输入电压 (V_{DD})。

必须注意 V_{DD} 电压，若小于 3.6V，则需将 CHPEN 置 1 使能充电泵；否则需将 CHPEN 清 0。若充电泵除能且 V_{DD} 电压低于 3.6V，则稳压器输出将得不到保证。

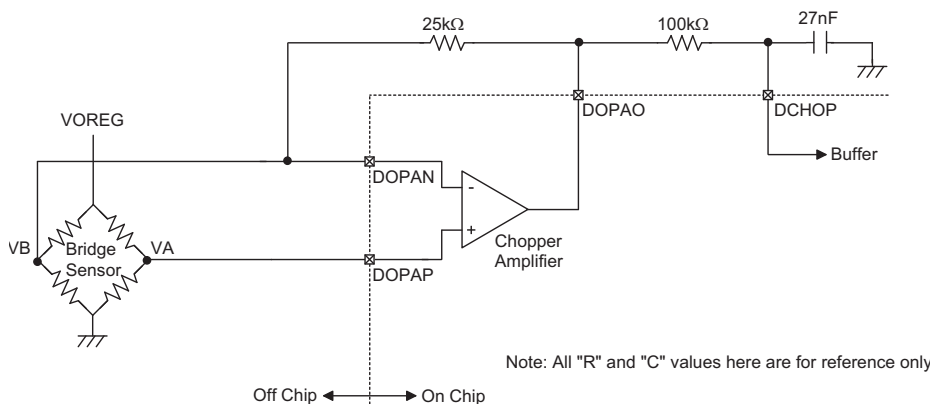
AD 转换器 — 双积分

HT46R73D-1 提供一个双积分 AD 转换器。它包括一个放大器、电压跟随器、积分器和比较器。

有两个与 AD 转换有关的寄存器，ADCR 和 ADCD。ADCR 寄存器是 AD 转换的控制寄存器，用来控制 AD 转换的开/关，断路器的开/关，充电/放电以及用于读取比较器输出状态。ADCD 寄存器用于设置断路器时钟分频。

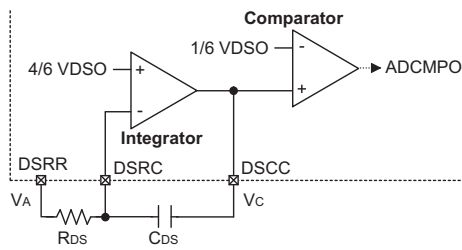


注： V_{INT} 和 V_{CMP} 信号可通过寄存器设置，使其来自不同电阻分压。



ADPWREN 位 (ADCR 的第 0 位) 用于控制 AD 转换器开/关。ADCCKEN 位 (ADCR 的第 6 位) 用于控制断路器时钟开/关。当 ADCCKEN=1, 断路器时钟将打开, 其时钟频率由 ADCD 寄存器设置。AD 转换器包含 OPA、缓冲器、积分器和比较器。而 band gap 电压发生器是一个独立部分, 它会自动根据稳压器使能而使能, 反之亦然。应用程序可控制各功能打开与关闭, 以降低功耗。充放电控制位 ADDISCH1~0 (ADCR 的第 1、2 位) 用于控制双积分电路充电与放电。ADCMPO 位 (ADCR 的第 3 位) 是只读位, 用于表示比较器输出, 而 ADINTM 位用来设置产生 AD 转换中断的 ADCMPO 触发模式。

下面描述基于 ADRR0=0。放大器和缓冲器组成前置放大器将传感器信号进行放大。



积分器、比较器、DSRR 与 DSRC 之间电阻 (R_{DS})、DSRC 与 DSCC 之间电容 (C_{DS}) 组成双积分 AD 转换主要部分。

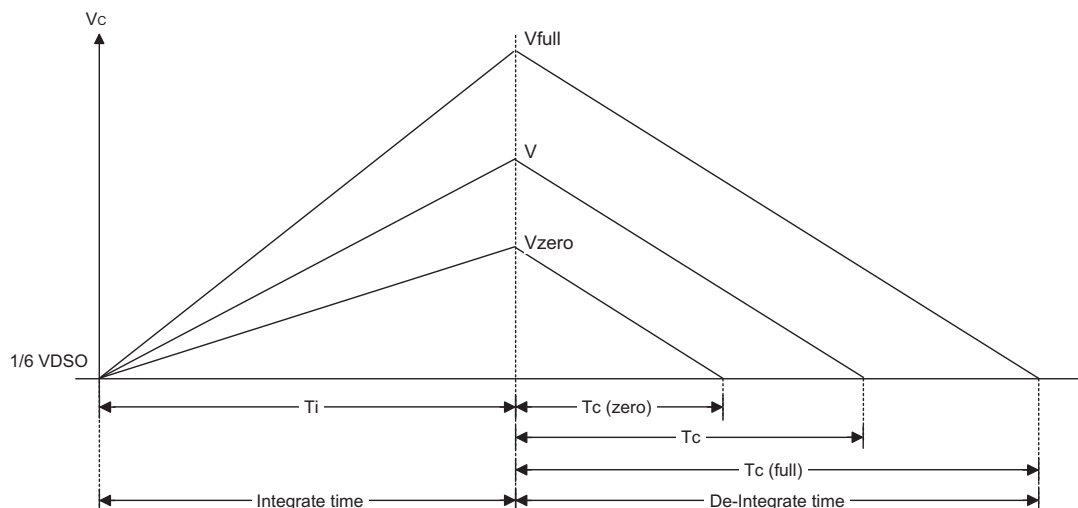
当 VC (DSCC 引脚电压) 降低低于 $1/6V_{DSO}$ 时, 比较器输出将会由高变低。

在一般的应用中, 程序可通过开关控制 AD 转换器进行固定时间 (T_i) 积分, 然后切换到放电模式。

当 V_c 降至低于 $1/6V_{DSO}$ (比较器会改变状态) 的这段时间记为反向积分时间 (T_c)。通过公式 1 可计算出输入电压 V_A 。

$$\text{公式 1: } V_A = (1/3) \times V_{DSO} \times (2 - T_c/T_i) \\ (\text{ADRR0}=0 \text{ 时})$$

提示: 必须注意选择正确的 R_{DS} , C_{DS} 来确定 T_i , 使 V_c 介于 $5/6V_{DSO}$ 和 $1/6V_{DSO}$ 之间 (例如: V_{FULL} 不能高于 $5/6V_{DSO}$, V_{ZERO} 不能低于 $1/6V_{DSO}$)



位	符号	功能
0	ADPWREN	双积分电路电源开关 0: 电源关闭 1: 电源来自稳压器
1~2	ADDISCH0~ ADDISCH1	设置 AD 转换器充电/放电 (ADDISCH1:0) 00: 保留 01: 充电 (积分器输入连接至缓冲器输出) 10: 放电 (积分器输入连接至 VDSO) 11: 保留
3	ADCMPO	双积分 AD 转换器---比较器输出 只读, 写此位指令被忽略 在放电阶段, 当积分器输出电压低于参考电压, ADCMPO 将由高变低
4~5	ADINTM0~ ADINTM1	ADC 积分器中断模式定义, 此两位用来定义 ADCMPO 数据中断触发模式: 00: 不触发中断 01: 上升沿触发中断 10: 下降沿触发中断 11: 上升/下降沿触发中断
6	ADCCKEN	AD 转换器断路器时钟开关 0: 关闭 1: 打开 (时钟频率由 ADCD 寄存器设置)
7	ADRR0	AD 转换器电阻选择 0: (V_{INT} , V_{CMP}) = ($4/6V_{OREG}$, $1/6V_{OREG}$) 1: (V_{INT} , V_{CMP}) = ($4.4/6V_{OREG}$, $1/6V_{OREG}$)

ADCR (18H) 寄存器

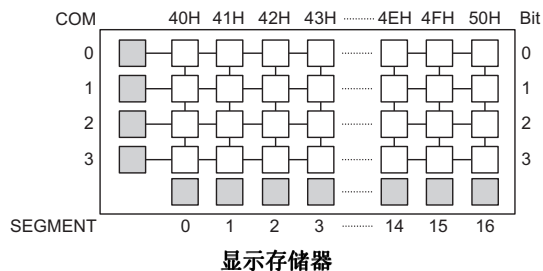
位	符号	功能
0	ADCD0	定义断路器时钟 (需 ADCCKEN 使能), 建议频率为 10kHz 其断路器频率: 0: 时钟频率= (f _{SYS} /32) /1 1: 时钟频率= (f _{SYS} /32) /2 2: 时钟频率= (f _{SYS} /32) /4 3: 时钟频率= (f _{SYS} /32) /8 4: 时钟频率= (f _{SYS} /32) /16 5: 时钟频率= (f _{SYS} /32) /32 6: 时钟频率= (f _{SYS} /32) /64 7: 时钟频率= (f _{SYS} /32) /128
1	ADCD1	
2	ADCD2	
3~7	—	保留

ADCD (1AH) 寄存器

LCD 显示存储器

HT46R73D-1 为 LCD 显示提供一个嵌入式数据存储器区域。这个区域位于第一段数据存储器(RAM Bank 1)的 40H 到 50H 单元。存储器段指针 Bank Pointer(BP; RAM 的 04H 单元)是通用存储器与 LCD 显示存储器之间切换的开关。当 BP 被置“1”，任何数据写入 40H~50H(用 MP1 和 R1 间接寻址访问)将会影响 LCD 的显示。当 BP 被清“0”，任何数据写入 40H~50H 意味着访问一般意义上的数据存储器。LCD 显示存储器能被读出和写入，但是只能通过间接寻址模式，并使用 MP1 来进行。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，可以控制显示或不显示。下图为显示存储器和 LCD 显示模块之间的映射关系。

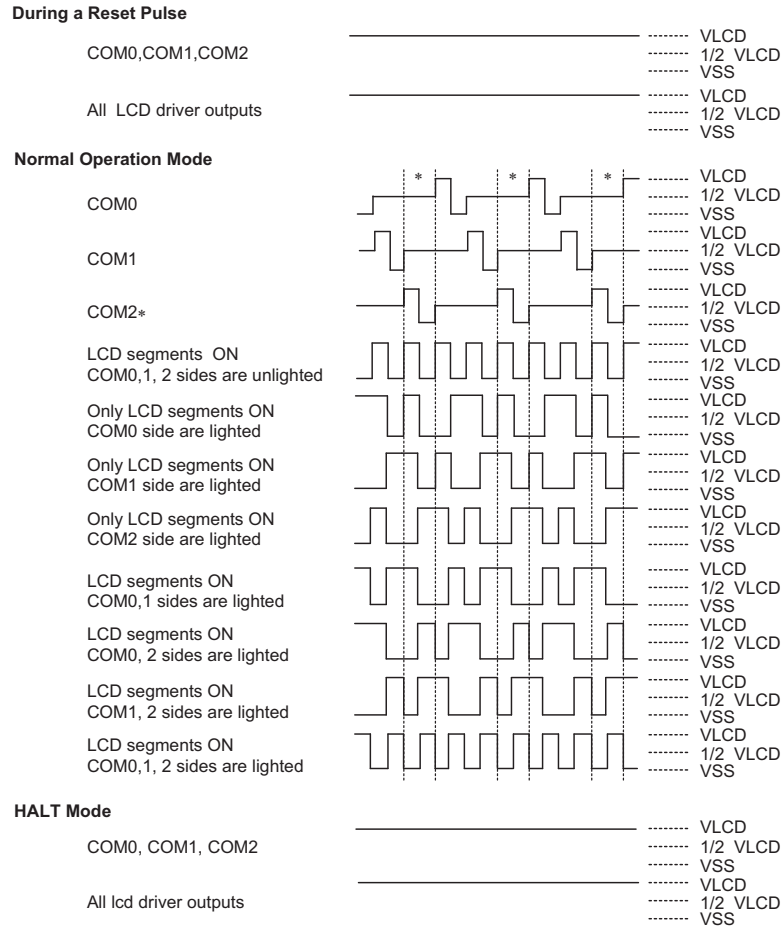
LCD 时钟由 IRC 经过分频器提供，由掩膜选项选择其频率为内部 RC/3 或内部 RC/4。注意：LCD 时钟分频由掩膜选项控制。



显示存储器

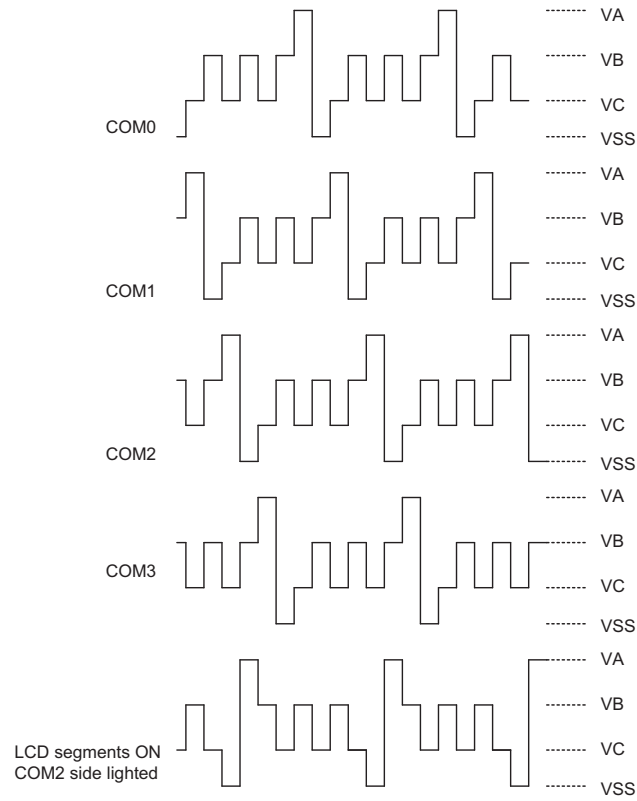
LCD 驱动输出

HT46R73D-1 的 LCD 驱动器输出数目可以由掩膜选项确定为 16×4, 17×3 或 17×2 (1/2 或 1/3 占空比)。LCD 驱动器偏压产生方式只有“R”型。LCD 驱动器偏置电压值可以由掩膜选项设置为 1/2 bias 或 1/3 bias。



Note: "*" Omit the COM2 signal, if the 1/2 duty LCD is used.

LCD 驱动输出 (1/3Duty,1/2 Duty, R Type)



Note: 1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD

LCD 驱动输出 (1/4Duty)

低电压复位/检测功能

系统具有低电压检测(LVD)和低电压复位(LVR)功能,可由掩膜选项设置为打开或关闭。如果选择 LVD 功能,用户可以通过 MODE.3 来打开/关闭低电压检测,通过 MODE.5 来读取低电压检测的状态。

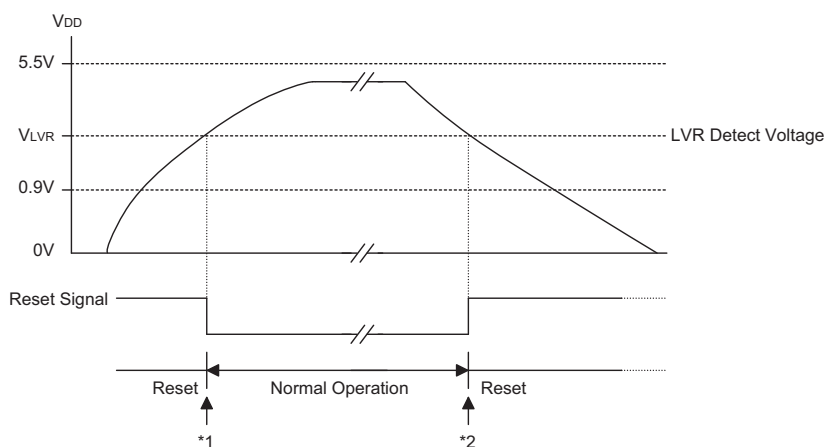
LVR 与外部复位信号有相同的功能,都会使芯片复位。但是在 HALT 状态下, LVR 和 LVD 是没有作用的。

为了监控器件的工作电压, HT46R73D-1 提供低电压复位功能。如果器件的工作电压在 $0.9V \sim V_{LVR}$ 之间,例如电池电压的变化,那么 LVR 会自动使器件产生内部复位。

LVR 功能说明如下:

- 低电压($0.9V \sim V_{LVR}$)的状态必须持续 1ms 以上。如果低电压的状态没有持续 1ms 以上,那么 LVR 会忽视它而不去执行复位功能。
- LVR 通过与外部 \overline{RES} 信号的“或”的功能来执行系统复位。

V_{DD} 与 V_{LVR} 之间的关系如下所示:



低电压复位

注: *1: 要保证系统振荡器起振并稳定运行,在系统进入正常运行以前, SST 提供额外的 1024 个系统时钟周期的延迟。

*2: 因为低电压状态必须保持 1ms 以上,因此进入复位模式就要有 1ms 的延迟。

MODE 寄存器的定义如下表:

位	标号	功能
0~1,4,6~7	—	未定义, 不确定值
2	IRCC	在 HALT 模式下, IRC 使能或除能选择位 0: IRC 和内部 RC 打开。1: IRC 关闭
3	LVDC	低电压检测打开/关闭(1/0)
5	LVDO	低电压检测输出(1/0) 1: 检测到低电压,只读;0: 没有检测到低电压。

MODE (09H) 寄存器

工作模式

HT46R73-D 有三种工作模式。系统时钟源来自外部 RC 振荡或晶振，它的工作模式有正常工作状态、空闲状态、暂停状态三种。如下表所示。

HALT 指令	IRCC	看门狗	外部 RC 振荡	IRC 时钟	系统时钟	工作模式
未执行	x	—	打开	打开	外部 RC 振荡	正常模式
执行	0	x	关闭	IRC 打开, 内部 RC 振荡打开	关闭	空闲模式
	1	打开	关闭	IRC 关闭, 内部 RC 振荡打开	关闭	暂停模式
		关闭	关闭	IRC 关闭, 内部 RC 振荡关闭	关闭	暂停模式

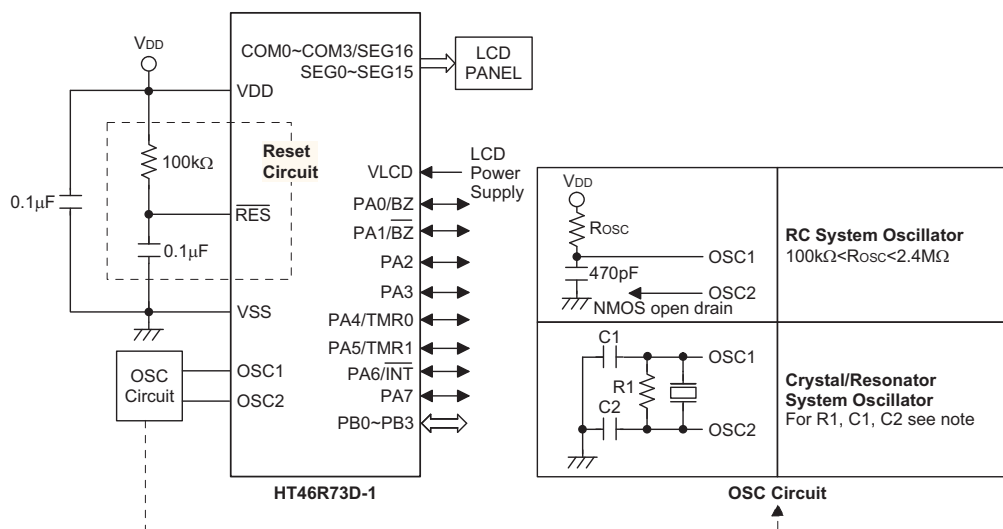
注：WDTOSC0~1 寄存器应设置为使能，否则，内部 RCOSC 将一直关闭。关于 WDTOSC 的细节部分请查看 WDT 部分。

掩膜选项

下表列出了所有掩膜选项。所有选项必须正确定义，以保证系统正常运行。

掩膜选项
系统振荡模式选项。 有两种选择方式：晶体振荡或 RC 振荡
f_S 时钟源选项。 有两种选择方式：内部 RC 或 $f_{SYS}/4$ 。
WDT 时钟源选项。 有两种选择方式：内部 RC 或 $f_{SYS}/4$ 。
WDT 打开/关闭选项。 由掩膜选项决定，WDT 打开或关闭。
WDT 溢出周期选项。 有四种选择的方式：WDT 时钟源的 $2^{16}/f_S$ 、 $2^{15}/f_S$ 、 $2^{14}/f_S$ 或 $2^{13}/f_S$ 分频。
CLR WDT 次数选项。 这个选项定义用指令清除 WDT 的方法。“One time”指用“CLR WDT”指令功能清除 WDT。“Two times”指的是必须要用 CLR WDT1 和 CLR WDT2 二条指令来清除 WDT。
蜂鸣器输出频率选项。 有八种输出频率供选择： $f_S/2^2 \sim f_S/2^9$ 。“ f_S ”是由掩膜选项确定的时钟源频率。
Wake-up 选项。 这个选项用来设置唤醒功能。外部引脚 PA 的下降沿，具有将系统从 HALT 模式唤醒的能力（按位设置）。
上拉电阻选项。 这个选项用来设置输入/输出口在输入模式时，是否带有内部上拉电阻。PA 和 PB 可以独立设置(按位设置)。
输入/输出与其它功能共用引脚选项。 PA0/ \overline{BZ} 、PA1/ \overline{BZ} ：PA0 和 PA1 可以设置为一般输入/输出口或蜂鸣器输出。
LCD COM 选择。 有三种方式可选择：2COM (1/2 duty)，3COM (1/3 duty) 或 4COM (1/4duty)。
LCD 偏压选项。 有二种选择：1/2 bias 或 1/3 bias。
LCD 驱动器时钟源选项。 有两种频率信号可选择：内部 RC 频率/3~内部 RC 频率/4。
LCD 在 HALT 模式开关选项。
LVR 选项。LVR 打开或关闭。
LVD 选项。LVD 打开或关闭。
INT 触发方式选项：禁止，上升沿触发，下降沿触发，或两者皆可触发

应用电路



注：1. 晶体/陶瓷谐振器为系统振荡器

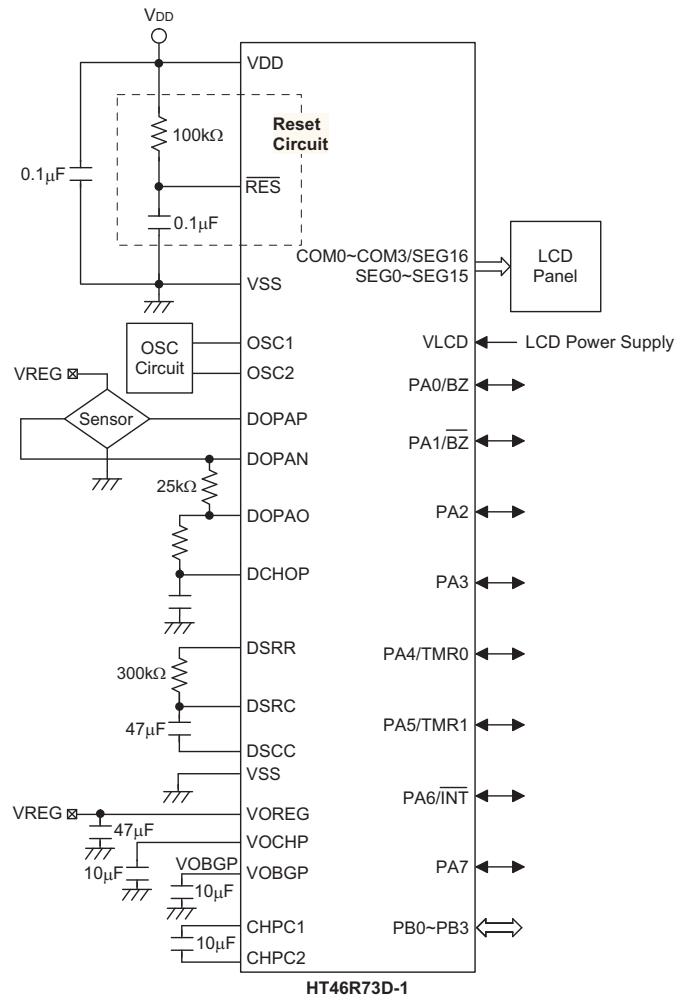
以晶体振荡器而言，仅有部分晶体是需要 C1 与 C2 来校准其振荡精度。而对于陶瓷谐振器来说，基本上都需要 C1 与 C2 来确保可以正常起振。在多数方案中，R1 没有必要使用。当 LVR 功能没有启用，如果要求当电压低于工作电压时晶体必须要停振，就有必要加上电阻 R1。C1 与 C2 的确切数据可以根据晶体/陶瓷谐振器的规格说明来选定。

2. 复位电路

复位电路的电阻及电容值的选取应确保工作电压 VDD 在 RES 引脚上上升到高之前能稳定，并将数值维持在正常允许的数据之内。为了防止噪声干扰，RES 脚的引线应尽可能地越短越好。

3. 对于应用中从复位电路进入的噪声干扰处理及振荡器外部器件的细节，可以参看应用范例 HA0075S。

图例



指令集介绍

指令集

任何单片机成功运作的核心在于它的指令集，此指令集为一组程序指令码，用来指导单片机如何去执行指定的工作。在盛群单片机中，提供了丰富且易变通的指令，共超过六十条，程序设计师可以事半功倍地实现他们的应用。

为了更加容易了解各式各样的指令码，接下来按功能分组介绍它们。

指令周期

大部分的操作均只需要一个指令周期来执行。分支、调用或查表则需要两个指令周期。一个指令周期相当于四个系统时钟周期，因此如果在 8MHz 的系统时钟振荡器下，大部分的操作将在 0.5 μ s 中执行完成，而分支或调用操作则将在 1 μ s 中执行完成。虽然需要两个指令周期的指令通常指的是 JMP、CALL、RET、RETI 和查表指令，但如果牵涉到程序计数器低字节寄存器 PCL 也将多花费一个周期去加以执行。即指令改变 PCL 的内容进而导致直接跳转至新地址时，需要多一个周期去执行。例如“CLR PCL”或“MOV PCL, A”。对于跳转命令必须注意的是，如果比较的结果牵涉到跳转动作将多花费一个周期，如果没有则需一个周期即可。

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用三种 MOV 的指令，数据不但可以从寄存器转移至累加器(反之亦然)，而且能够直接移动立即数到累加器。数据传送最重要的应用之一是从接收端口接收数据或者传送数据到输出端口。

算术运算

算术运算和数据处理是大部分单片机应用所需具备的能力，在盛群单片机内部的指令集中，可直接实现加与减的运算。当加法的结果超出 255 或减法的结果少于 0 时，要注意正确的处理进位和借位的问题。INC、INCA、DEC 和 DECA 指令提供了对一个指定地址的值加一或减一的功能。

逻辑和移位运算

标准逻辑运算例如 AND、OR、XOR 和 CPL 全都包含在盛群单片机内部的指令集中。大多数牵涉到数据运算的指令，数据的传送必须通过累加器。在所有逻辑数据运算中，如果运算结果为零，则零标志位将被置位，另外逻辑数据运用形式还有移位指令，例如 RR、RL、RRC 和 RLC 提供了向左或向右移动一位的方法。移位指令常用于串行端口的程序应用，数据可从内部寄存器转移至进位标志位，而此位则可被检验，移位运算还可应用在乘法与除法的运算组成中。

分支和控制的转换

程序分支是采取使用 JMP 指令跳转至指定地址或使用 CALL 指令调用子程序的形式，两者之不同在于当子程序被执行完毕后，程序必须马上返回原来的地址。这个动作是由放置在子程序里的返回指令 RET 来实现，它可使程序跳回 CALL 指令之后的地址。在 JMP 指令中，程序则只是跳到一个指定的地址而已，并不需如 CALL 指令般跳回。一个非常有用的分支指令是条件跳转，跳转条件是由数据存储器或指定位来加以决定。遵循跳转条件，程序将继续执行下一条指令或略过且跳转至接下来的指令。这些分支指令是程序走向的关键，跳转条件可能是外部开关输入，或者是内部数据位的值。

位运算

供数据存储器中单个位的运算指令是盛群单片机的特性之一。这特性对于输出端口位的规划尤其有用，其中个别的位或端口的引脚可以使用“SET [m].i”或“CLR [m].i”指令来设定其为高位或低位。如果没有这特性，程序设计师必须先读入输入输出的 8 位数据，处理这些数据，然后再输出正确的新数据。这种读入-修改-写出的过程现在则被位运算指令所取代。

查表运算

数据的储存通常由寄存器完成，然而当处理大量固定的数据时，它的存储量常常造成对个别存储器的不便。为了改善此问题，盛群单片机允许在程序存储器中设定一块数据可直接存储的区域，只需要一组简易的指令即可对数据进行查表。

其它运算

除了上述功能指令外，其它指令还包括用于省电的“HALT”指令和使程序在极端电压或电磁环境下仍能正常工作的看门狗定时器控制指令。这些指令的使用则请查阅相关的章节。

指令集摘要

下列表格是按照指令功能来分类描述的，可作为基本指令的参考，其使用了如下惯例。

表格惯例：

x: 立即数

m: 数据存储器地址

A: 累加器

I: 0~7 号位

Addr: 程序存储器地址

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：
 $[m] \leftarrow \overline{[m]}$

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：
 $ACC \leftarrow \overline{[m]}$

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

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，结果放入累加器
 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: $Program\ Counter \leftarrow Program\ Counter + 1$
 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: $ACC \leftarrow ACC \text{ “OR” } [m]$
 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: $ACC \leftarrow ACC \text{ “OR” } x$
 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: $[m] \leftarrow ACC \text{ “OR” } [m]$
 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: $Program\ Counter \leftarrow Stack$
 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: $Program\ Counter \leftarrow Stack$
 $ACC \leftarrow x$
 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： $Program\ Counter \leftarrow Stack$
 $EMI \leftarrow 1$

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： $[m].0 \leftarrow [m].7, [m].(i+1) \leftarrow [m].i; (i=0\sim6)$

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： $ACC.0 \leftarrow [m].7, ACC.(i+1) \leftarrow [m].i; (i=0\sim6)$

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： $[m].(i+1) \leftarrow [m].i; (i=0\sim6)$
 $[m].0 \leftarrow C$
 $C \leftarrow [m].7$

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： $ACC.(i+1) \leftarrow [m].i; (i=0\sim6)$
 $ACC.0 \leftarrow C$
 $C \leftarrow [m].7$

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： $[m].7 \leftarrow [m].0, [m].i \leftarrow [m].(i+1); \quad (i=0\sim6)$

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： $ACC.7 \leftarrow [m].0, ACC.i \leftarrow [m].(i+1); \quad (i=0\sim6)$

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： $[m].i \leftarrow [m].(i+1); \quad (i=0\sim6)$

$[m].7 \leftarrow C$

$C \leftarrow [m].0$

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： $ACC.i \leftarrow [m].(i+1); \quad (i=0\sim6)$

$ACC.7 \leftarrow C$

$C \leftarrow [m].0$

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： $ACC \leftarrow ACC + [\bar{m}] + C$

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： $[m] \leftarrow ACC + [\bar{m}] + C$

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

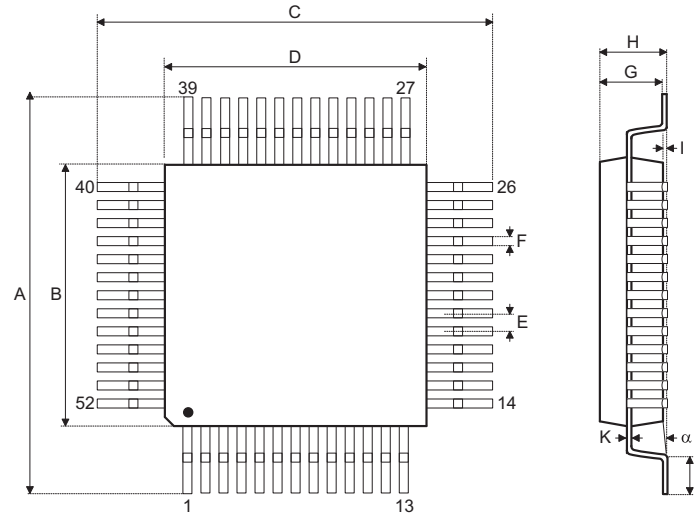
TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

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

52-pin QFP (14×14) 外形尺寸



符号	尺寸(单位: mm)		
	最小	正常	最大
A	17.3	—	17.5
B	13.9	—	14.1
C	17.3	—	17.5
D	13.9	—	14.1
E	—	1	—
F	—	0.4	—
G	2.5	—	3.1
H	—	—	3.4
I	—	0.1	—
J	0.73	—	1.03
K	0.1	—	0.2
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号香槟广场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 © 2008 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>