

技术相关信息

- [工具信息](#)
- [FAQs](#)
- [应用范例](#)
 - [HA0017S HT49 MCU系列微控制器读写HT24 系列应用范例](#)
 - [HA0024S HT49 MCU RTC实时时钟的使用](#)
 - [HA0025S HT49 MCU中的Time base\(时基\)的使用说明](#)
 - [HA0026S HT49 MCU输入/输出的使用](#)
 - [HA0027S HT49 MCU定时/计数器的使用](#)
 - [HA0075S MCU复位电路和振荡电路的应用范例](#)

特性

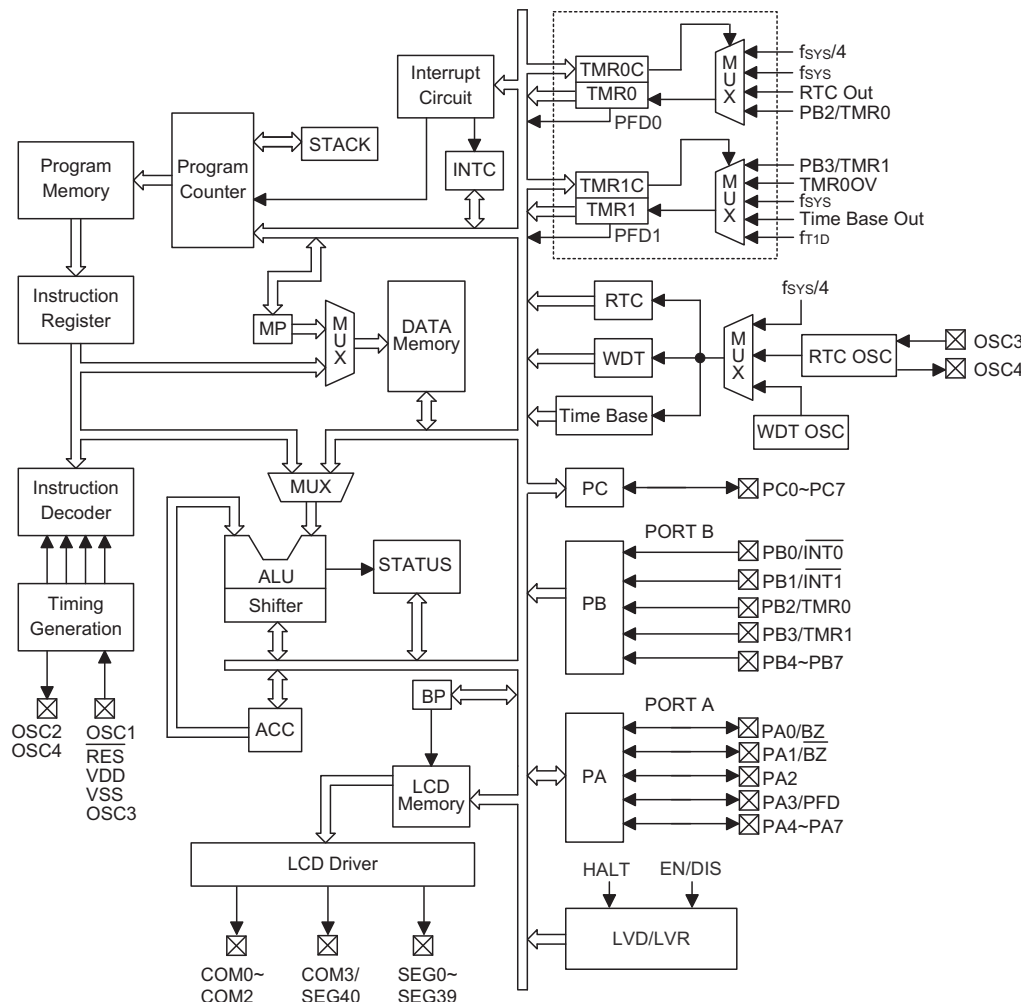
- 工作电压：
HT49R70A-1/HT49C70-1:
fsys = 4MHz 2.2V~5.5V
fsys = 8MHz 3.3V~5.5V
HT49C70L:
fsys = 500kHz 1.2V~2.2V
- 8 位输入口
- 16 位双向输入/输出口
- 2 个外部中断输入
- 具有预分频器及中断功能的一个 8 位和一个 16 位的可编程定时/计数器
- 带 41×2、41×3、40×4 段的 LCD 驱动器
- 8K×16 位的程序存储器 ROM
- 224×8 位的数据存储器 RAM
- 实时时钟 (RTC)
- 实时时钟 (RTC) 的 8 位预分频器
- 看门狗定时器
- 蜂鸣器输出
- 内置晶体、RC、32768Hz 晶体的振荡电路
- 提供暂停和唤醒功能，以降低功耗
- 16 层堆栈
- 位操作指令
- 16 位查表指令
- 系统时钟为 8MHz 时，指令周期为 0.5 μs (HT49R70A-1/HT49C70-1)
- 系统时钟为 500kHz 时，指令周期为 8μs (HT49C70L)
- 63 条指令
- 所有指令都可在 1 或 2 个指令周期内完成
- HT49R70A-1/HT49C70-1 具有低电压复位/检测功能
- 64-pin LQFP 和 100-pin LQFP 封装

概述

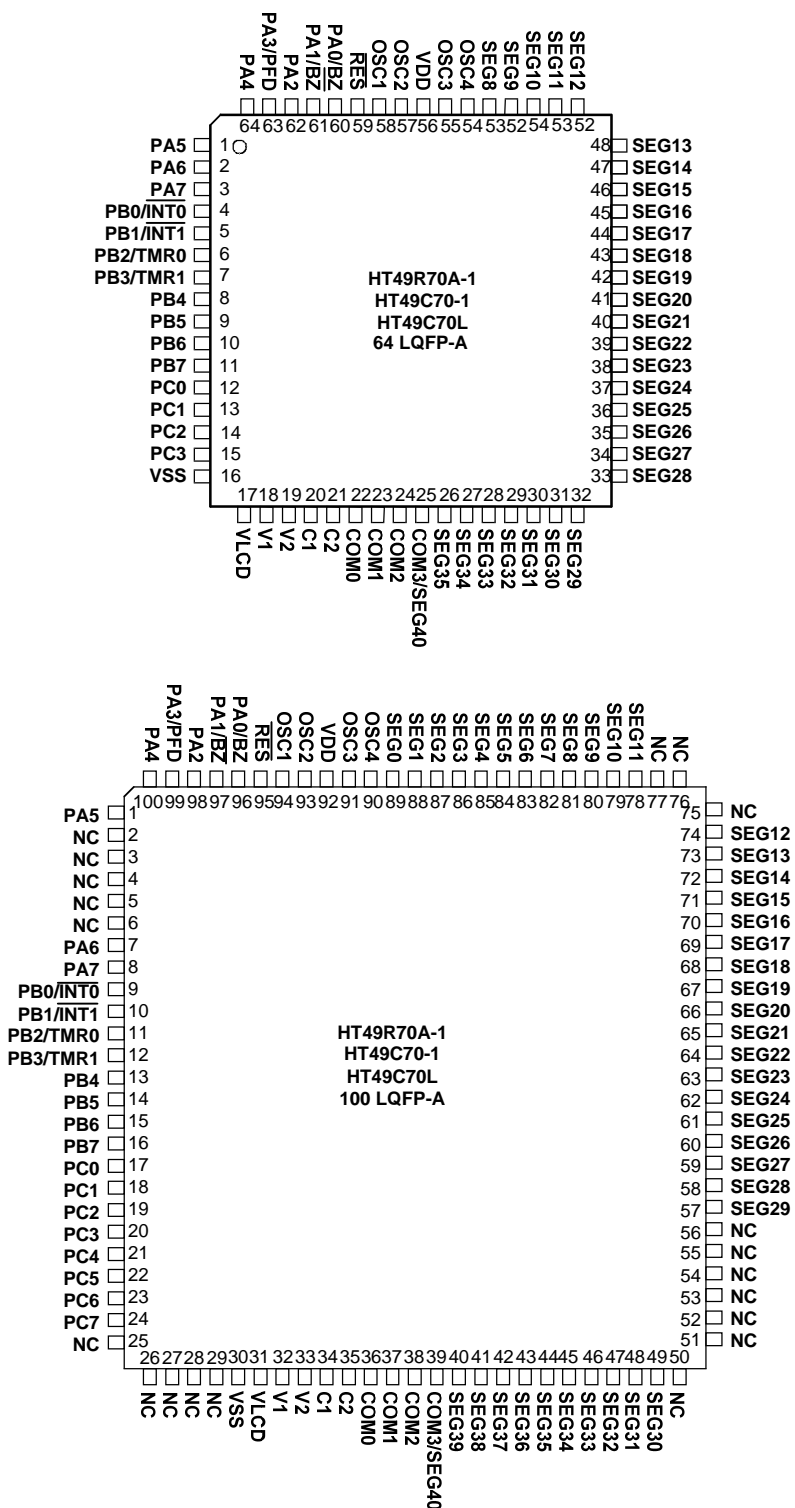
HT49R70A-1/HT49C70-1/HT49C70L 是 8 位高性能精简指令集单片机，专门为需要 LCD 显示功能的产品而设计。配置版本 HT49C70-1 和 HT49C70L 与 OTP 版本 HT49R70A-1 引脚和功能完全相同。HT49C70L 为低电压版本，工作电压最低可为 1.2V，可适用于一节电池的应用系统。

具有低功耗、I/O 使用灵活、可编程分频器、计数器、振荡类型选择、暂停和唤醒、另有灵活的蜂鸣器输出以及 LCD 显示功能，使这款单片机功能更多样性。可以广泛应用于电子秤、电表、气表、定时器、计算器、遥控器和其他带 LCD 显示功能的工业或家用消费类产品等系统中。

方框图



引脚图



引脚说明

引脚名称	输入/输出	配置选项	说 明
PA0/BZ PA1/BZ PA2 PA3/PFD PA4~PA7	输入/输出	唤醒功能 上拉电阻 CMOS 或 NMOS	PA0~PA7 是带斯密特触发输入的 8 位双向输入/输出端口。每一位能被配置选项设置为唤醒输入。PA0~PA3 为 CMOS 输出或 NMOS 输入/输出(由配置选项确定有无上拉电阻)。PA4~PA7 是带上拉电阻的 NMOS 输入/输出。PA0~PA1 由配置选项确定为输入输出引脚或蜂鸣器输出。PA3 由配置选项确定为输入输出引脚或 PFD 输出。
PB0/INT0 PB1/INT1 PB2/TMR0 PB3/TMR1 PB4~PB7	输入	—	PB0~PB7 是 8 位带上拉电阻的斯密特触发输入输出。PB0、PB1 分别软件设置为输入引脚或外部中断控制引脚 (INT0) 和 (INT1)。而 PB2、PB3 也可软件设置为输入引脚或定时/计数器 TMR0、TMR1 输入引脚。
PC0~PC7	输入/输出	上拉电阻 CMOS 或 NMOS	PC0~PC7 是双向 8 位的有斯密特触发输入能力的输入/输出口。这些端口能由配置选项确定为 CMOS 输出, NMOS 输入(上拉电阻可由 OPTION 决定)/输出。
V2	输入	—	在 HT49C70L 中, 为 LCD 电源输入 在 HT49R70A-1/HT49C70-1 中, 为电压泵
VLCD	输入	—	在 HT49C70L 中, 为电压泵 在 HT49R70A-1/HT49C70-1 中, 为 LCD 电源输入
V1, C1, C2	输入	—	电压泵。
COM0~COM2 COM3/ SEG40	输出	1/2,1/3 或 1/4 占空比	SEG40 可由配置选项被确定为 LCD 的段(segment)或公共端的输出驱动器。COM2~COM0 是作为 LCD 面板公共端的输出极。
SEG0~SEG39	输出	—	LCD 面板段驱动器输出。
OSC1 OSC2	输入 输出	晶体 或 RC	OSC1 和 OSC2 连接一个 RC 或一个晶体(由配置选项确定)来产生内部系统时钟。在 RC 方式下, OSC2 是一个系统时钟四分频的输出端。系统时钟也可来源于 RTC 振荡器。如果系统时钟来源于 RTC 振荡器, OSC1 和 OSC2 引脚悬空。
OSC3 OSC4	输入 输出	RTC 或系统时钟	实时时钟振荡器。OSC3 和 OSC4 连接到一个 32768Hz 的晶体振荡器来产生定时时间或连接到系统时钟源(由配置选择决定)。无内置电容
VDD	—	—	正电源
VSS	—	—	负电源, 接地
RES	输入	—	斯密特触发复位输入, 低电平有效。

极限参数
HT49R70A-1/HT49C70-1

 电源供应电压 ----- $V_{SS} - 0.3V \sim V_{SS} + 6.0V$

 端口输入电压 ----- $V_{SS} - 0.3 \sim V_{DD} + 0.3V$

 I_{OL} 总电流 ----- 150mA

最大功耗-----500mW

 储存温度 ----- $-50^{\circ}C \sim 125^{\circ}C$

 工作温度----- $-40^{\circ}C \sim 85^{\circ}C$

 I_{OH} 总电流----- -100mA

HT49C70L

电源供应电压 ----- $V_{SS} - 0.3V \sim V_{SS} + 2.5V$ 储存温度 ----- $-50^{\circ}C \sim 125^{\circ}C$
 端口输入电压 ----- $V_{SS} - 0.3 \sim V_{DD} + 0.3V$ 工作温度----- $-40^{\circ}C \sim 85^{\circ}C$
 I_{OL} 总电流 ----- 50mA I_{OH} 总电流----- 30mA
 最大功耗-----150mW

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。

直流特性
HT49R70A-1 和 HT49C70-1
Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	测试条件				
V _{DD}	工作电压	—	LVR 关闭, f _{sys} = 4MHz	2.2	—	5.5	V
			f _{sys} = 8MHz	3.3	—	5.5	V
V _{LCD}	LCD 供电电源 (注*)	—	VA ≤ 5.5V	2.2	—	5.5	V
I _{DD1}	工作电流(晶体振荡)	3V	无负载, f _{sys} = 4MHz	—	1	2	mA
		5V		—	3	5	mA
I _{DD2}	工作电流(RC 振荡)	3V	无负载, f _{sys} = 4MHz	—	1	2	mA
		5V		—	3	5	mA
I _{DD3}	工作电流(RC, 晶体振荡)	5V	无负载, f _{sys} = 8MHz	—	4	8	mA
I _{DD4}	工作电流 (f _{sys} = RTC 振荡)	3V	无负载	—	0.3	0.6	mA
		5V		—	0.6	1	mA
I _{STB1}	静态电流(*f _s = T1)	3V	无负载, 系统 HALT 时关闭 LCD	—	—	1	μA
		5V		—	—	2	μA
I _{STB2}	静态电流 (*f _s = RTC 振荡)	3V	无负载, 系统 HALT HALT 时开启 LCD, 电容型	—	2.5	5	μA
		5V		—	10	20	μA
I _{STB3}	静态电流 (*f _s = WDT RC 振荡)	3V	无负载, 系统 HALT HALT 时开启 LCD, 电容型	—	2	5	μA
		5V		—	6	10	μA
I _{STB4}	静态电流 (*f _s = RTC 振荡)	3V	无负载, 系统 HALT 开 LCD, 电阻型 1/2 偏压	—	17	30	μA
		5V		—	34	60	μA
I _{STB5}	静态电流 (*f _s = RTC 振荡)	3V	无负载, 系统 HALT 开 LCD, 电阻型 1/3 偏压	—	13	25	μA
		5V		—	26	50	μA
I _{STB6}	静态电流 (*f _s = WDT RC 振荡)	3V	无负载, 系统 HALT 开 LCD, 电阻型 1/2 偏压	—	14	25	μA
		5V		—	28	50	μA
I _{STB7}	静态电流 (*f _s = WDT RC 振荡)	3V	无负载, 系统 HALT 开 LCD, 电阻型 1/3 偏压	—	10	20	μA
		5V		—	20	40	μA
V _{IL1}	I/O口、TMR、 \overline{INT} 输入低电压	—	—	0	—	0.3V _{DD}	V
V _{IH1}	I/O口、TMR、 \overline{INT} 输入高电压	—	—	0.7V _{DD}	—	V _{DD}	V
V _{IL2}	输入低电压(\overline{RES})	—	—	0	—	0.4V _{DD}	V
V _{IH2}	输入高电压(\overline{RES})	—	—	0.9V _{DD}	—	V _{DD}	V

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OL1}	I/O 口灌电流	3V	V _{OL} = 0.1 V _{DD}	6	12	—	mA
		5V		10	25	—	mA
I _{OH1}	I/O 口源电流	3V	V _{OH} = 0.9 V _{DD}	-2	-4	—	mA
		5V		-5	-8	—	mA
I _{OL2}	LCD com/seg 口电流	3V	V _{OL} = 0.1 V _{DD}	210	420	—	μA
		5V		350	700	—	μA
I _{OH2}	LCD com/seg 口电流	3V	V _{OH} = 0.9 V _{DD}	-80	-160	—	μA
		5V		-180	-360	—	μA
R _{PH}	上拉电阻	3V	—	20	60	100	KΩ
		5V		10	30	50	KΩ
V _{LVR}	低电压复位电压	—	—	2.7	3.2	3.6	V
V _{LVD}	低电压检测电压	—	—	3.0	3.3	3.6	V

注意：“*” VA 值参考 LCD 驱动部分。
“*fs” 请参照 WDT 时钟选择说明

HT49R70L
T_a = 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	—	2.2	V
I _{DD1}	工作电流(晶体振荡)	1.5V	无负载, f _{sys} = 455KHz	—	60	100	μA
I _{DD2}	工作电流(RC 振荡)	1.5V	无负载, f _{sys} = 400KHz	—	50	100	μA
I _{DD4}	工作电流 (f _{sys} = RTC 振荡)	1.5V	无负载	—	2.5	5	μA
I _{STB1}	静态电流(*fs=T1)	1.5V	无负载, 系统 HALT 时关闭 LCD	—	0.1	0.5	μA
I _{STB2}	静态电流 (*fs= RTC 振荡)	1.5V	无负载, 系统 HALT HALT 时开启 LCD, 电容型	—	1	2	μA
I _{STB3}	静态电流(*fs= WDT RC 振荡)	1.5V	无负载, 系统 HALT HALT 时开启 LCD, 电容型	—	0.5	1	μA
V _{IL1}	I/O口、TMR、 $\overline{\text{INT}}$ 输入低电压	—	—	0	—	0.3V _{DD}	V
V _{IH1}	I/O口、TMR、 $\overline{\text{INT}}$ 输入高电压	—	—	0.8V _{DD}	—	V _{DD}	V
V _{IL2}	输入低电压($\overline{\text{RES}}$)	—	—	0	—	0.4V _{DD}	V
V _{IH2}	输入高电压($\overline{\text{RES}}$)	—	—	0.9V _{DD}	—	V _{DD}	V
I _{OL1}	I/O 口灌电流	1.5V	V _{OL} = 0.1 V _{DD}	0.4	0.8	—	mA
I _{OH1}	I/O 口源电流	1.5V	V _{OH} = 0.9 V _{DD}	-0.3	-0.6	—	mA
R _{PH}	上拉电阻	1.5V	—	75	150	300	KΩ
V _{LVR}	低电压复位电压	—	—	2.7	3.2	3.6	V
V _{LVD}	低电压检测电压	—	—	3.0	3.3	3.6	V

注：“*” VA 值参考 LCD 驱动部分。

“*fs” 请参照 WDT 时钟选择说明。

交流特性
HT49R70A-1 和 HT49C70-1
Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
f _{SYS1}	系统时钟 (晶体振荡)	—	2.2V~5.5V	400	—	4000	kHZ
		—	3.3V~5.5V	400	—	8000	kHZ
f _{SYS2}	系统时钟 (RC 振荡)	—	2.2V~5.5V	400	—	4000	kHZ
		—	3.3V~5.5V	400	—	8000	kHZ
f _{SYS3}	系统时钟 32768Hz 晶体振荡	—	—	—	32768	—	Hz
f _{RTCOSC}	实时时钟频率	—	—	—	32768	—	Hz
f _{TIMER}	定时器输入频率(TMR)	—	2.2V~5.5V	0	—	4000	kHZ
		—	3.3V~5.5V	0	—	8000	kHZ
t _{WDTOSC}	看门狗振荡器周期	3V	—	45	90	180	μs
		5V		32	65	130	μs
t _{RES}	外部复位低电平脉冲宽度	—	—	1	—	—	μs
t _{SST}	系统启动延迟周期	—	从 HALT 状态唤醒	—	1024	—	*t _{SYS}
t _{LVR}	低电压复位宽度	—	—	0.5	1	2	ms
t _{INT}	中断脉冲宽度	—	—	1	—	—	μs

 注: *t_{sys}= 1/f_{SYS1}, 1/f_{SYS2} 或 1/f_{SYS3}
HT49C70L
Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
f _{SYS1}	系统时钟 (晶体振荡)	—	1.2V~2.2V	400	—	500	kHz
f _{SYS2}	系统时钟 (RC 振荡)	—	1.2V~2.2V	400	—	500	kHz
f _{SYS3}	系统时钟 32768Hz 晶体振荡	—	—	—	32768	—	Hz
f _{RTCOSC}	实时时钟频率	—	—	—	32768	—	Hz
f _{TIMER}	定时器输入频率(TMR)	—	1.2V~2.2V	0	—	500	kHz
t _{WDTOSC}	看门狗振荡器周期	1.5V	—	35	70	140	μs
t _{RES}	外部复位低电平脉冲宽度	—	—	10	—	—	μs
t _{SST}	系统启动延迟周期	—	从 HALT 状态唤醒	—	1024	—	*t _{SYS}
t _{LVR}	低电压复位宽度	—	—	0.5	1	2	ms
t _{INT}	中断脉冲宽度	—	—	10	—	—	μs

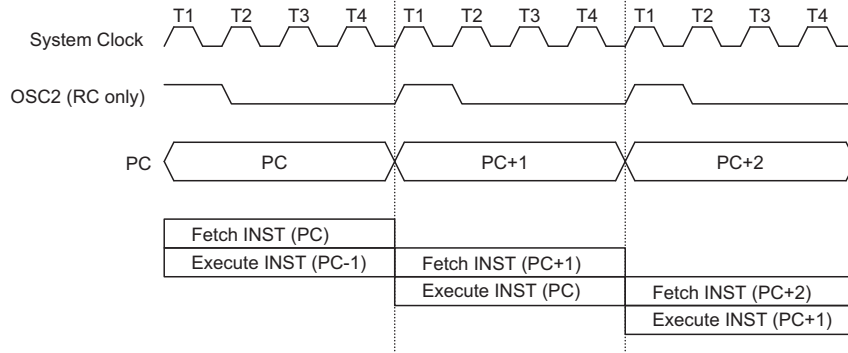
 注: *t_{sys}= 1/f_{SYS1}, 1/f_{SYS2} 或 1/f_{SYS3}

功能说明

指令执行时序

系统频率由晶体振荡、RC 振荡或 32768Hz 晶体振荡产生。系统内部对此频率进行四分频,产生四个不重叠的时钟周期作为系统时钟,一个指令周期包含四个系统时钟周期。

指令读取与执行是以流水线方式来进行的。这种方式允许指令在前一个指令周期进行读取操作,而在当前指令周期里进行解码与执行。这种流水线方式能在一个指令周期里有效地执行每条指令。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

程序计数器为 13 位宽,是指令字的标志器,控制了程序的流程。单片机从程序计数器所指向的程序存储器里抓取指令来执行,程序计数器(PC)最大可以对 8192 个地址进行寻址。

取得指令字以后,程序计数器(PC)会自动指向下一个指令字的地址。PC 将指向包含下一指令码的存储器字

但是如果执行跳转、条件跳转、写 PCL 寄存器、子程序调用、中断、复位初始化或由中断子程序返回等情况,程序计数器(PC)会载入相对应的地址,而非下一个地址来操作程序流程。

举例而言,如果执行条件单跳跃指令,当条件符合时,则在执行此条指令时装入的下一条指令会被抛弃,且一个空指令周期(dummy cycle)会被插入,程序计数器才会指向正确的指令字地址,因此需要两个指令周期。其他情况,程序计数器指向下一条指令字的地址。

模 式	程序计数器												
	*12	*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	0
外部中断 0	0	0	0	0	0	0	0	0	0	0	1	0	0
外部中断 1	0	0	0	0	0	0	0	0	0	1	0	0	0
定时/计数器 0 溢出	0	0	0	0	0	0	0	0	0	1	1	0	0
定时/计数器 1 溢出	0	0	0	0	0	0	0	0	1	0	0	0	0
时基中断	0	0	0	0	0	0	0	0	1	0	1	0	0
RTC 中断	0	0	0	0	0	0	0	0	1	1	0	0	0
短跳转	PC+2												
装载 PCL	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
跳转, 子程序调用	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
从子程序返回	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

程序计数器

注意: *12~*0: 程序计数器位

S12~S0: 堆栈寄存器位

#12~#0: 指令代码位

@7~@0: PCL 位

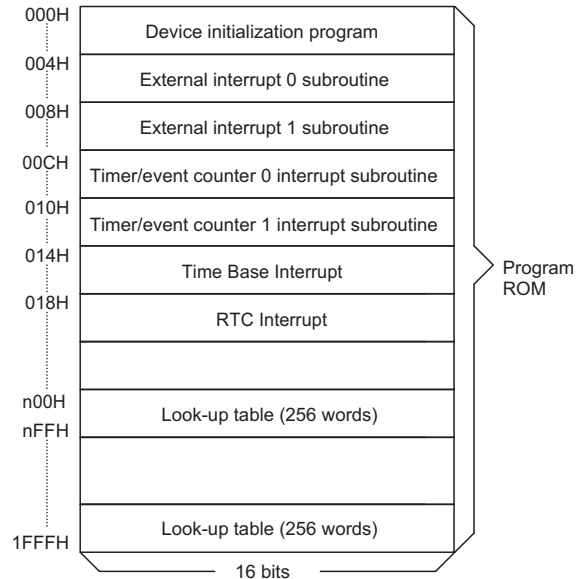
程序计数器的低字节单元(PCL)是一个可读写的暂存器(06H),如果写入一个值将会产生一个一个短程跳跃动作,这个短程跳跃范围是 256 个地址。同样在此情况下也会插入一个空指令周期。

程序存储器

程序存储器 (ROM) 用来存放要执行的指令代码。还包括数据, 表格和中断入口。程序存储器为 8192×16 位。程序存储器 (ROM) 的空间都可以由程序计数器 (PC) 或表格指针 (Table Pointer) 来寻址。

下列出的程序存储器 (ROM) 的地址是专为特殊用途而保留的。

- **地址 000H**
此地址保留给程序初始化之用。当系统复位时, 程序会从 000H 地址开始执行。
- **地址 004H**
此地址保留给外部中断服务使用。当中断允许, 且堆栈未滿, 则一旦INT0输入脚有下降沿信号, 就能产生中断, 程序会从 004H 地址开始执行外部中断服务程序。
- **地址 008H**
此地址保留给外部中断服务程序使用。当中断允许, 且堆栈未滿, 则一旦INT1输入脚有下降沿信号, 就能产生中断, 程序会从 008H 地址开始执行外部中断服务程序。
- **地址 00CH**
此地址保留给定时/计数器 0 中断服务使用。当中断允许, 且堆栈未滿, 则一旦定时/计数器 0 发生溢出时, 就能产生中断, 程序会从 00CH 地址开始执行中断服务程序。
- **地址 010H**
此地址保留给定时/计数器 1 中断服务使用。当中断允许, 且堆栈未滿, 则一旦定时/计数器 1 发生溢出时, 就能产生中断, 程序会从 010H 地址开始执行中断服务程序。
- **地址 014H**
此地址保留给时基中断服务使用。当中断允许, 且堆栈未滿, 则一旦时基中断产生, 程序会从 014H 地址开始执行中断服务程序。
- **地址 018H**
此地址保留给实时时钟中断服务使用。当中断允许, 且堆栈未滿, 则一旦实时中断产生, 程序会从 018H 地址开始执行中断服务程序。



程序存储器

• **表格区**

ROM 内的任何区域都可用来作为查表区使用。查表指令为 TABRDC [m] 与 TABRDL [m]。TABRDC [m]是查当前页的内容[1 页=256 个字 (word)]。TABRDL [m]是查最后一页的内容。[m] 为数据被存入的地址。在执行 TABRDC [m]指令 (或 TABRDL [m] 指令) 后, 将会传送当前页(或最后一页)上的数据内容的低位字节到[m], 而数据内容的高位字节传送到 TBLH(08H)。只有表格的低位字节会送到定义的目的地址, 而表格中的其他位则送到 TBLH 的低位。TBLH 为只读寄存器, 而表格指针 (TBLP) 是可以读写的寄存器 (07H), 用来指明表格地址。在访问表格以前, 通过对 TBLP 寄存器赋值来指明表格地址。查表指令要花两个指令周期来完成这一条指令的操作。按照用户的需要, 表格地址这些位置可以作为正常的程序存储器来使用。

指令	表格地址												
	*12	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC[m]	P12	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL[m]	1	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

表格区

注意: *12 ~ *0 : 表格地址位 P12 ~ P8 : 当前程序计数器位 @7 ~ @0 : 表格指针位

堆栈寄存器 — STACK

堆栈寄存器 (STACK) 是一个用来保存 PC 值的特殊存储单元, 此系列有 16 层堆栈。堆栈寄存器既不是数据存储器的部分, 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。栈将要弹出的地址的由堆栈指针 (SP) 来索引, 堆栈指针也不能读出与写入。一旦发生子程序调用或中断被响应时, 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被响应结束时 (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。

如果堆栈满了, 并且发生了不可屏蔽的中断, 那么中断请求标志将会被记录下来, 而不会响应该中断, 一旦堆栈指针 (由 RET 或 RETI) 发生递减时, 才会响应中断。这个功能防止堆栈溢出, 使得程序员易于使用这种结构。但是堆栈已满, 接着又执行一个子程序调用 (CALL), 那么堆栈会产生溢出, 而使首先进入堆栈的内容丢失。堆栈只会保留最近的十六个返回地址。

数据存储器

数据存储器 RAM 由 224×8 位组成。它可分成两部分, 即特殊功能寄存器和通用数据存储器。这两部分的大多数单元可以读写, 但有一些是只读的。

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 (BP; 04), 累加器 (ACC; 05H), 程序计数器低位字节寄存器 (PCL; 06H), 表格指针寄存器 (TBLP; 07H), 表格高位字节寄存器 (TBLH; 08H), 实时时钟控制寄存器 (RTCC; 09H), 状态寄存器 (STATUS; 0AH), 中断控制寄存器 0 (INTC0; 0BH), 定时/计数器 0 (TMR0; 0DH), 定时/计数器 0 控制寄存器 (TMR0C; 0EH), 定时/计数器 1 高位寄存器 (TMR1H; 0FH), 定时/计数器 1 低位寄存器 (TMR1L; 10H), 定时/计数器 1 控制寄存器 (TMR1C; 11H), I/O 寄存器 (PA; 12H, PB; 14H, PC; 16H), 中断控制寄存器 1 (INTC1; 1EH)。另外, 通用数据存储器的 20H~FFH 地址作为程序数据和控制信息使用。

所有的 RAM 区单元都能直接执行算术、逻辑、递增、递减和移位等运算。除了某些特殊的位以外, RAM 中的每一位都可以由 SET[m].i 和 CLR[m].i 指令来置位和复位。这些 RAM 都可通过存储器指针寄存器 0 (MP0; 01H) 和存储器指针寄存器 1 (MP1; 03H) 来间接寻址。

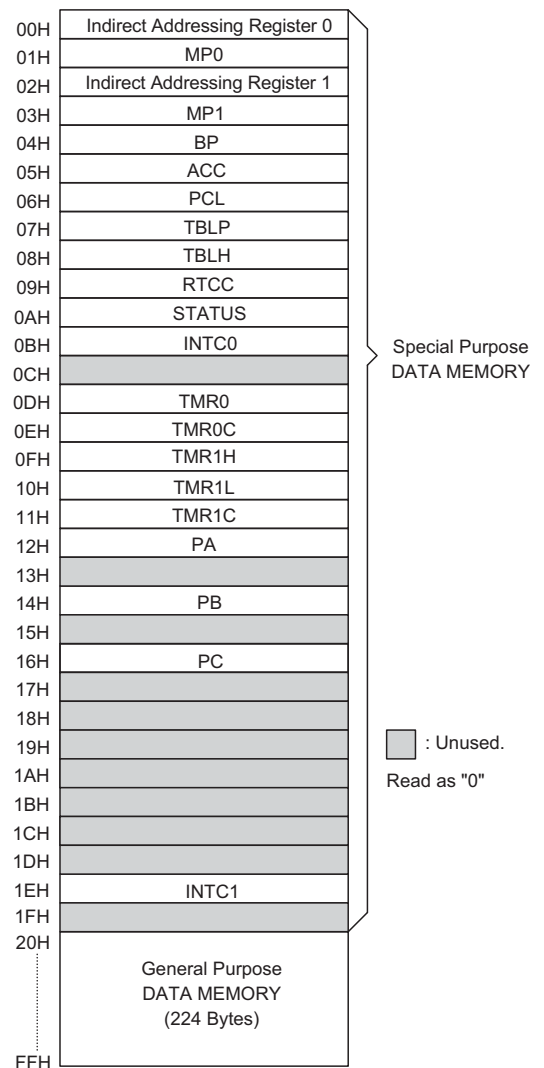
间接寻址寄存器

地址 00H 和 02H 作为间接寻址寄存器。它们都没有实际的物理结构。任何对 [00H] 和 [02H] 的读/写操作, 都会访问由 MP0[01H] 和 MP1[03H] 所指向的 RAM 单元。间接地读取 00H 或 02H 单元, 将会得到值 00H, 而间接地写入 00H 或 02H 单元, 则不会产生任何结果。

间接寻址寄存器之间不支持数据传送功能。间接寻址指针寄存器 MP0 和 MP1 的宽度都是 8 位, 被用来访问由间接寻址寄存器所对应的 RAM。MP0 只能用于数据存储器, 而 MP1 能用于数据存储器和 LCD 显示存储器。

累加器 — ACC

累加器 (ACC) 与算术逻辑单元 (ALU) 运算相联系。它对应于 RAM 的地址 05H, 并能对立即数进行操作, 在数据存储器间的数据传送都必须经过累加器。



数据存储器

算术逻辑单元 — ALU

算术逻辑单元是执行 8 位算术，逻辑运算的电路。它提供如下的功能：

- 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)
- 递增和递减运算 (INC, DEC)
- 分支转移 (SZ, SNZ, SIZ, SDZ ...)

算术逻辑单元 ALU 不仅会保存运算的结果而且会改变状态寄存器的值。

状态寄存器 — STATUS

状态寄存器 (0AH) 的宽度为 8 位，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。

除了 TO 和 PDF 以外，状态寄存器中的位都可用指令来改变。任何写到状态寄存器的数据不会改变 TO 或 PDF 标志位。注意与状态寄存器有关的操作可能会导致与预期不一样的结果。系统上电、看门狗定时器溢出、执行 HALT 指令或清除看门狗定时器都能改变 TO 和 PDF 标志位。而 Z、OV、AC 和 C 标志位都反映了最近一次操作的状态。

进入中断程序或执行子程序调用时，状态寄存器不会自动压入堆栈。如果状态寄存器的内容是重要的，而且子程序可能会改变状态寄存器的内容，那么程序员必须事先将其保存好，以免被破坏。

符号	位	功 能
C	0	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,那么被 C 置位;反之, C 被清除。它也可被一个循环移位指令影响。
AC	1	在加法运算中低四位产生了进位或减法运算中在低四位不产生借位,AC 被置位;反之, AC 被清除。
Z	2	算术运算或逻辑运算的结果为零则 Z 被置位;反之, Z 被清除。
OV	3	如果运算结果次高位向最高位进位,但最高位并不产生进位输出,或情况相反,则 OV 被置位;反之, OV 被清除。
PDF	4	系统上电或执行了 CLR WDT 指令, PDF 被清除。执行 HALT 指令 TD 被置位。
TO	5	系统上电或执行了 CLR WDT 指令,或 HALT 指令, TO 被清除。WDT 定时溢出, TO 被置位。
—	6	未定义,读出为“0”
—	7	未定义,读出为“0”

STATUS(0AH) 寄存器

中断

系统提供二个外部中断、二个内部定时/计数器中断、一个内部时基中断和一个实时时钟中断。中断控制寄存器 0 (INTC0; 0BH) 和中断控制寄存器 1 (INTC1; 1EH) 包含了中断控制位以及中断请求标志位, 中断控制位是用来设置中断允许/禁止。

一旦有中断响应, 其它的中断都被禁止 (通过清除 EMI 位)。这种方法目的在于防止中断嵌套。这时如有其它中断请求发生, 只会把这个中断请求的标志记录下来。如果在一个中断服务程序中有另一个中断需要响应的话, 程序员可以设置 EMI 位及 INTC0 或 INTC1 所对应位来允许中断嵌套服务。如果堆栈已满, 即使相关的中断允许, 该中断请求也不会被响应。直到堆栈指针发生递减时才会响应。如果需要立即得到中断服务, 则必须避免堆栈饱和。

所有的中断都具有唤醒功能。当中断响应时, 系统会将程序计数器 (PC) 压入堆栈, 转移到指定的 ROM 地址的中断入口。只有程序计数器 (PC) 的内容压入堆栈。如果寄存器和状态寄存器的内容会被中断服务程序改变, 从而破坏主程序的执行流程的话, 那么程序员必须事先将这些数据保存起来。

外部中断是由 $\overline{\text{INT0}}$ 或 $\overline{\text{INT1}}$ 脚上、下降沿信号触发的, 会置位相关的中断请求位 (EIF0, INTC0 的第 4 位, EIF1, INTC1 的第 5 位)。当中断允许, 堆栈未满, 外部中断触发时, 那么将会产生 04H 或 08H 单元的子程序调用。系统将会清除中断请求标志 (EIF0 或 EIF1) 和 EMI 位以禁止其他的中断响应。

内部定时/计数器 0 中断产生, 会置位定时/计数器 0 中断请求标志位 (T0F, INTC0 的第 6 位), 中断的请求标志一般是由定时/计数器 0 溢出产生的。当中断允许, 堆栈未满, 并且 T0F 已被置位, 就会产生 0CH 单元的子程序调用。系统将会清除中断请求标志位 (T0F) 和 EMI 位, 以禁止其他中断的响应。内部定时/计数器 1 的运作方式与之相同, 相关的中断请求标志位是 T1F (INTC1 的第 4 位), 而它的子程序调用的地址是 10H 单元。

时基中断产生, 会置位时基中断请求标志 (TBF, INTC1 的第 5 位), 中断的请求是由有规律的时基信号产生的。当中断允许, 堆栈未满, 并且 TBF 已被置位, 就会产生 14H 单元的子程序调用。系统将清除中断请求标志位 (TBF) 和 EMI 位, 以禁止其他中断的响应。

实时时钟中断产生, 会置位实时时钟中断请求标志 (RTF, INTC1 的第 6 位), 中断的请求是由有规律的实时时钟信号产生的。当中断允许, 堆栈未满, 并且 RTF 已被置位, 就会产生 18H 单元的子程序调用。系统将清除该中断请求标志位 (RTF) 和 EMI 位, 以禁止其他中断的响应。

单片机在执行中断子程序期间, 会屏蔽其他的中断请求, 直到执行 RETI 指令或是 EMI 位和相关的中断控制位都被置 1 (堆栈未满时)。若要从中断子程序返回时, 只要执行 RET 或 RETI 指令。RETI 指令将会自动置位 EMI 允许中断服务, 而 RET 则不会自动置位 EMI。

如果中断在两个连续的 T2 脉冲的上升沿之间发生, 且中断被允许的话, 那么在后来的二个 T2 周期, 该中断会被响应。如果同时发生中断服务请求, 那么下列表中列出了中断服务优先等级。这种优先权地位也可以通过 EMI 位的复位来屏蔽。

No.	中 断 源	优 先 权	中 断
A	外部中断 0	1	04H
B	外部中断 1	2	08H
C	定时/计数器 0 中断	3	0CH
D	定时/计数器 1 中断	4	10H
E	时基中断	5	14H
F	实时时钟中断	6	18H

中断控制寄存器 (INTC0) 其 RAM 地址是 0BH, 由定时/计数器 0 中断请求标志位 (T0F)、外部中断 1 请求标志位 (EIF1)、外部中断 0 请求标志位 (EIF0)、定时/计数器 0 允许位 (ET0I)、外部中断 1 允许位 (EEI1)、外部中断 0 允许位 (EEI0) 和主中断控制允许位 (EMI) 组成。中断控制寄存器 (INTC1) 其 RAM 地址是 1EH, 由实时时钟中断请求标志位 (RTF)、时基中断请求标志位 (TBF)、定时/计数器 1 中断请求标志位 (T1F)、实时时钟中断允许位 (ERTI)、时基中断允许位 (ETBI) 和定时/计数器 1 允许位 (ET1I) 组成。EMI、EEI、EEI0、EEI1、ET0I、ET1I、ETBI 和 ERTI 都是用来控制中断的允许/禁止的状态的。这些位防止响应正在进行中断服务中的中断请求。一旦置位中断请求标志位 (RTF, TBF, T0F, T1F, EIF1, EIF0), 它们将在 INTC1 或 INTC0 中被保留下来, 直到中断响应或标志位被软件指令清除。

建议不要在中断子程序中使用“CALL”指令来调用子程序。因为中断经常随机发生在不可预见的情况中或某一个确定的应用程序可能要求立即服务的情况下, 在这种时候, 如果只剩下一个堆栈, 若此时中断不能很好地被控制, 在中断服务程序中调用了子程序调用的操作, 可能使原先的控制序列被破坏。

位	符号	功	能
0	EMI	总中断控制位	允许=1, 禁止=0
1	EEI0	外部中断 0 控制位	允许=1, 禁止=0
2	EEI1	外部中断 1 控制位	允许=1, 禁止=0
3	ET0I	定时/计数器 0 中断控制位	允许=1, 禁止=0
4	EIF0	外部中断 0 请求标志位	有=1, 无=0
5	EIF1	外部中断 1 请求标志位	有=1, 无=0
6	T0F	内部定时/计数器 0 中断请求位	有=1, 无=0
7	—	未使用位, 读出为 “0”	

INTC0(0BH) 寄存器

位	符号	功	能
0	ET1I	定时/计数器 1 中断控制位	允许=1, 禁止=0
1	ETBI	时基中断控制位	允许=1, 禁止=0
2	ERTI	实时时钟中断控制位	允许=1, 禁止=0
3	—	未使用位, 读出为 “0”	
4	T1F	定时/计数器 1 中断请求位	有=1, 无=0
5	TBF	时基中断请求位	有=1, 无=0
6	RTF	实时时钟中断请求位	有=1, 无=0
7	—	未使用位, 读出为 “0”	

INTC1(1EH) 寄存器

振荡器

针对各方面的应用的要求，多种振荡器的选择给设计者提供了灵活的应用方案。当要求系统的看门狗时钟在最大值时，系统提供三种振荡电路以供选择。所有的振荡器的选定是根据电路设计来要求的。

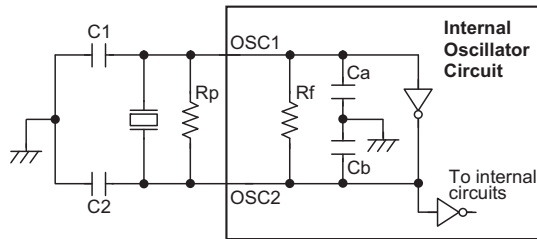
三种产生系统时钟的方式为：

- 外部晶体振荡.
- 外部 RC 振荡
- 外部 RTC 振荡

任意一种振荡器可设计为系统时钟。更多关于振荡器的资料可以在盛群半导体公司的网站的应用范例 HA0075S 中查到。

外部晶体振荡/谐振器

简单的方式即在 OSC1 与 OSC2 之间连接一个晶体，用来提供晶体振荡所要的反馈 (Feedback) 和相移 (Phase Shift)，这种设计不需要连接外部电容。但是，对于一些晶体和大多数谐振器，为了确保起振并产生精确的系统时钟，可以外加两个小容值的电容，C1 和 C2。C1 和 C2 的容值的选择可参考以下提供的规格书。一些设计需外接并联反馈电阻 R_p 来帮助系统起振。



Note: 1. R_p is normally not required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

在电压为 5V，室温为 25℃情况下的内部 Ca、Cb 和 Rf 的典型参数		
Ca	Cb	Rf
11~13Pf	13~15Pf	270 KΩ

HT49R70A-1 和 HT49C70-1 的内部振荡器元件参数

晶体振荡器 C1、C2 的典型参数			
晶体振荡频率	C1	C2	CL
8MHz	TBD	TBD	TBD
4MHz	TBD	TBD	TBD
1MHz	TBD	TBD	TBD

注： 1. C1 和 C2 的值只是参考值。 2. CL 是生产商规定的电容值。

HT49R70A-1 和 HT49C70-1 的晶体振荡器电容参数

晶体振荡器 C1、C2 的典型参数			
晶体振荡频率	C1	C2	CL
3.58MHz	TBD	TBD	TBD
1MHz	TBD	TBD	TBD
455KHz	TBD	TBD	TBD

注： 1. C1 和 C2 的值只是参考值。

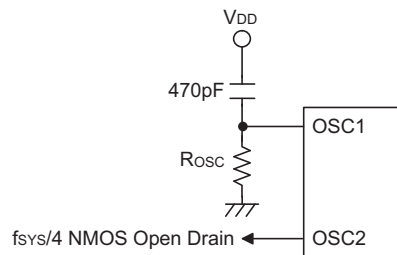
HT49R70A-1 和 HT49C70-1 的 RC 振荡器电容参数

晶体振荡器 C1、C2 的典型参数			
晶体振荡频率	C1	C2	CL
455KHz	TBD	TBD	TBD

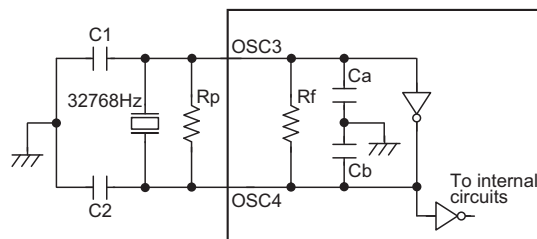
注： 1. C1 和 C2 的值只是参考值。

HT49C70L 的 RC 振荡器电容参数
外部 RC 振荡器

采用外部 RC 振荡，那么在 OSC1 与地之间要接一个外接电阻，与 VDD 之间接一个电容。对于 HT49R70A-1/HT49C70-1 其阻值范围为 $24K\Omega \sim 1M\Omega$ ；对于 HT49C70L 其阻值范围为 $560K\Omega \sim 1M\Omega$ 。在 OSC2 端接一个上拉电阻，可获得系统时钟四分频信号输出，它可以用于同步系统外部逻辑。注意 OSC2 输出是一种 NMOS 的开路类型，如果它被用作内部频率输出，应该加一个上拉电阻。在 RC 振荡是一种低成本的方案，但是振荡频率由于 V_{DD} ，温度及芯片自身参量的漂移而产生误差。因此，它不能满足要求精确的振荡频率对时间敏感的应用。对于外部电阻值可以参考盛群半导体网站的资料。温度和 V_{DD} 的特性参数表。注意微控制器电路的外部电阻，确定系统振荡器的频率。外部电容不会影响振荡器的频率。相反要选用晶体振荡。在 OSC1 与 OSC2 之间连接一个晶体，用来提供晶体振荡所要的反馈 (Feedback) 和相移 (Phase Shift)。除此以外，不再需要其他外部元件。另外，可在 OSC1 与 OSC2 之间接一个陶瓷谐振器 (Resonator) 来取代晶体振荡用来得到频率产生，但是需要分别在 OSC1 和 OSC2 外接一个电容器。


外部 RTC 振荡器

当微控制器进入低功耗模式时，系统时钟状态的选择使微控制器工作在低功耗模式。显然，当微控制器工作在低功耗模式需要保持内部功能，如此像计数器操作。在这种振荡电路中，只能在 OSC3 与 OSC4 之间连接 32.768KHz 的晶体。为了使系统时钟频率更精确的产生，需加两个小容值的电容，C1 和 C2。可以根据盛群半导体的规格书为 C1 和 C2 选择适当的容值。外部并联反馈电阻 R_p ，一般是不需要的，只是在帮助系统时钟起振时才用到。用 32.768KHz 的晶体可以使系统时钟工作在低功耗模式。



Note: 1. R_p is normally not required.
2. Although not shown OSC3/OSC4 pins have a parasitic capacitance of around 7pF.

在电压为 5V，室温为 25°C 情况下的内部 C_a 、 C_b 和 R_f 的典型参数		
C_a	C_b	R_f
TBD	TBD	TBD

RTC 谐振器内部元件参数

RTC 谐振器的 C1、C2 典型参数			
晶体振荡频率	C1	C2	CL
32768Hz	TBD	TBD	TBD
注： 1. C1 和 C2 的值只是参考值。 2. CL 是生产商规定的电容值。			

32768 HZ 谐振器的电容参数

当进入低功耗模式，系统时钟被选择为定时器和看门狗定时器时，这个 32768 Hz 振荡器将保持正常运行。当在正常模式下，有一个时钟延在 RTC 振荡器模式下出现，需要等待 RTC 振荡器启动。在 RTCC 寄存器的这个 QOSC 位需提供快速启动功能和短暂的延时。在正常工作模式下，这个位将清零，同时将使 RTC 快速起振。显然，附加功耗的增加将要求程序置 QOSC 位为“1”，大约 2 秒钟后系统才启动正常工作。注意，QOSC 位被置为任意状态对系统振荡是无影响的，RTC 振荡器仍正常工作，仅仅是功耗的增大。

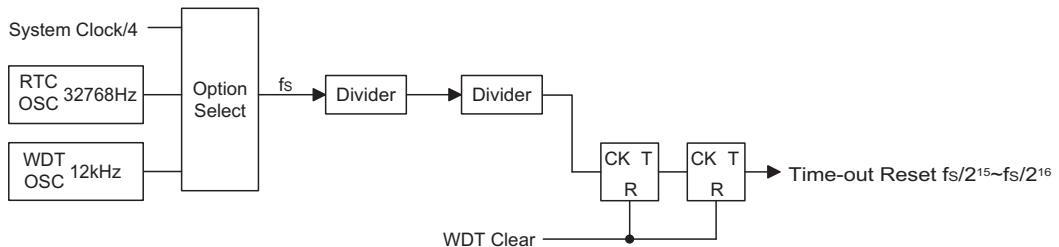
WDT 看门狗振荡器

WDT 看门狗振荡器是一个在 5V 电压时周期为 65μs 的内部 RC 振荡器。不需连接任何外部元件。当系统进入暂停模式时，系统时钟会停止，但是 WDT 振荡器仍然以看门狗模式工作。如果要降低功耗，可在配置选项中关闭 WDT 振荡器。

看门狗定时器

WDT 的时钟源是由专用的 RC 振荡器 (WDT 振荡器)、指令时钟 (系统时钟 4 分频) 或实时时钟振荡器 (RTC) 来提供。看门狗定时器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 可以由配置选项设置为打开或关闭。如 WDT 关闭，所有与 WDT 有关的执行都等同一个空操作。

在选定了 WDT 的时钟源后，它的溢出周期为 $fs/2^{15} \sim fs/2^{16}$ 。



看门狗定时器

如果 WDT 时钟源为内部 WDT 振荡器的话，那么溢出时间会因为温度、V_{DD} 以及芯片参数的变化而变化。如果选择了指令时钟为时钟源，在 HALT 状态时，WDT 会停止计数而失去保护功能。只能由外部逻辑来重新启动系统。因此若单片机工作在干扰很大的环境中，强烈建议使用片内的 RC 振荡器 (WDT OSC)，因为 HALT 模式会使系统时钟停止运作。

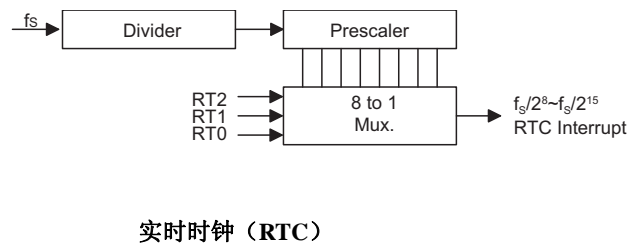
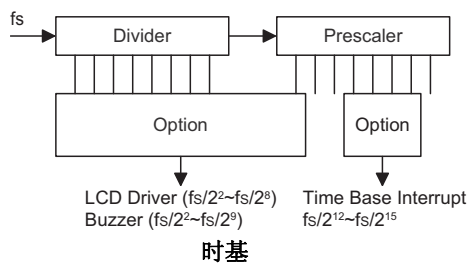
在正常运作下，WDT 溢出会使系统复位并置位 TO。但在 HALT 模式下，溢出执行热复位，只复位 PC 程序计数器和堆栈指针 SP。要清除 WDT 的值 (包括 WDT 预分频器) 可以有三种方法：外部复位 (低电平输入到 RES 端)、软件指令和 HALT 指令三种。软件指令由 CLR WDT 和 CLR WDT1 及 CLR WDT2 二组指令组成。这两组指令中，只能选取其中一种。选择的方式由配置选项的 CLR WDT 次数选项决定。如果“CLR WDT”被选择 (即 CLR WDT 次数为 1)，那么只要执行 CLR WDT 指令就会清除 WDT。在 CLR WDT1 和 CLR WDT2 被选择的情况下 (即 CLR WDT 次数为 2)，那么要交替执行二条指令才可清除 WDT，否则，会由于 WDT 的溢出而使系统复位。

多功能定时器

系统为 WDT、时基和 RTC 提供了具有不同溢出周期的多功能定时器。多功能定时器由一个七级预分频器和一个 8 位分频器组成。使用的时钟来自 WDT OSC、RTC OSC 或指令时钟（系统时钟四分频）。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 $f_s/2^2 \sim f_s/2^8$ ），并为蜂鸣器输出电路提供可选择的频率信号（范围从 $f_s/2^2 \sim f_s/2^9$ ），频率由配置选择。为了正确地显示，建议选择 4kHz 左右的信号作为 LCD 驱动信号。

时基

时基指的是用一个周期性的溢出来产生一个有规律的内部中断。溢出周期的范围从 $f_s/2^{12} \sim f_s/2^{15}$ ，由配置选项确定。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，堆栈未滿，那么就产生一个地址在 14H 的子程序调用。时基溢出信号也能被作为定时/计数器 1 的时钟源来获得更长的溢出周期。



实时时钟 — RTC

实时时钟（RTC）的运作情况和时基是一样的。它是用来提供一个有规律的内部中断。它的溢出周期的范围从 $f_s/2^8 \sim f_s/2^{15}$ ，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 RTC 溢出产生，相关的中断请求标志位（RTF；INTC1 第 6 位）被置位。如果中断是允许的，并且堆栈未滿，那么就产生一个地址在 18H 的子程序调用。实时时钟溢出信号也能被作为定时/计数器 0 的时钟源来获得更长的溢出周期。

RT2	RT1	RT0	RTC 实时时钟分频级数
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}

注意：“*” 不建议使用

暂停模式 — HALT

暂停模式是由 HALT 指令来实现的，有如下功能：

- 系统振荡器关闭，但 WDT 或 RTC 振荡器会继续运行（如果选择 WDT 振荡器或 RTC）。
- RAM 及寄存器的内容保持不变。
- WDT 被清除并重新计数（如果 WDT 的时钟是来自 WDT 振荡器或 RTC）。
- 所有的输入/输出端口都保持其原先状态。
- 置位 PDF 标志位，清除 TO 标志位。
- LCD 驱动器仍然运行（如果 WDT OSC 或 RTC OSC 被选择）。

外部复位、中断、外部输入一个下降沿的信号到 PA 口，或 WDT 溢出均可使系统唤醒。外部复

位能使系统初始化，而 WDT 溢出唤醒复位为热复位。可以通过判断 TO 和 PDF 状态，确定系统复位的原因。系统上电复位和执行 CLR WDT 指令，可清除 PDF 标志位；而执行 HALT 指令，则会置位 PDF。如 WDT 产生溢出，置位 TO 标志位，还能产生唤醒，但只复位程序计数器 PC 和堆栈指针 SP，其他都保持原状态。

PA 口唤醒和中断唤醒可作为正常运行的继续，PA 口的每一位都可以通过配置选项来单独设定为对系统的唤醒。如果是输入输出唤醒的话，程序会从 HALT 后的下一条命令继续执行。如果是中断唤醒的话，则会产生二种情况：如果相关的中断被禁止或中断是允许的，但堆栈已满，程序会从 HALT 后的下一条命令继续执行。如果中断允许并且堆栈未满，那么会响应这个中断。

如果进入暂停模式以前，某个中断请求位已经被置位，那么系统不能用这个中断来唤醒。当唤醒事件发生时，要延迟 1024 t_{sys} （系统时钟周期），系统才会重新正常运行。这就是说，在唤醒后被插入了一个等待时间。如果是中断唤醒，那么实际中断程序的执行就被延迟了一个周期以上。如果唤醒导致下一条指令执行，那么在一个等待周期结束后指令就立即被执行。

为了减小功耗，在进入暂停模式之前必须要小心处理输入输出状态。

复位

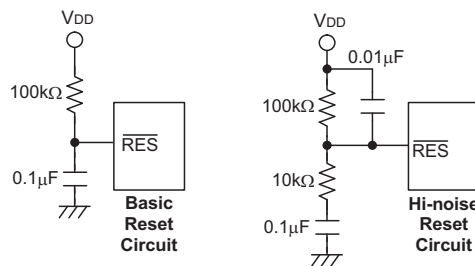
总共有三种方法会产生初始复位：

- 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，热复位只复位程序计数器 PC 和堆栈指针 SP，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。

TO	PDF	复位条件
0	0	电源上电复位
u	u	正常运作时由 $\overline{\text{RES}}$ 发生复位
0	1	由 $\overline{\text{RES}}$ 唤醒暂停模式
1	u	正常运作时发生看门狗定时器超时
1	1	由看门狗定时器唤醒暂停模式

注意：“u”表示未变。



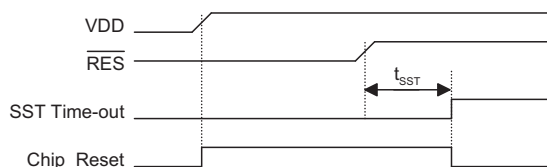
复位电路

注意：上图是大多数电路应用的是基础的复位电路，如果想防范外部噪声的干扰，可以采用高噪声复位电路。

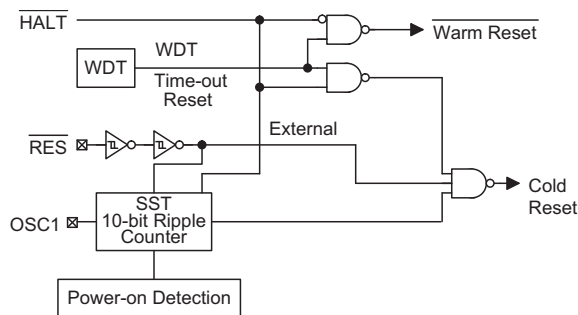
为了保证系统振荡器起振并稳定运行，系统复位(包括上电复位、看门狗定时器溢出或由 $\overline{\text{RES}}$ 端复位)或由暂停状态唤醒时，系统启动定时器(SST)提供了一个额外的延迟时间，共 1024 个系统时钟周期。系统复位时，SST 会被加在复位延时中。由暂停模式唤醒也会启动 SST 延迟。当系统上电、正常运行时 WDT 溢出或 RES 脚复位，系统需要额外增加一个加载 Option 的时间。

系统复位时各功能单元的状态如下所示：

程序计数器(PC)	000H
中断	禁止
预分频器	清除
看门狗定时器	清除, 复位后看门狗定时器开始计数
定时/计数器	关闭
输入/输出口	输入模式
堆栈指针	指向堆栈的顶端



复位时序图



复位框图

特殊功能寄存状态概述表

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)*
TMR	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
TMR1H	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1L	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
PC	000H	000H	000H	000H	000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	---- --0	---- --0	---- --0	---- --0	---- --u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
STATUS	--00 xxxx	--lu uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-000 -000	-uuu -uuu
RTCC	--00 0111	--00 0111	--00 0111	--00 0111	--uu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

- 注意: 1. “*” 表示热复位。
 2. “u” 表示不变化。
 3. “x” 表示不确定。

定时/计数器

系统提供两个定时/计数器。一个定时/计数器包含一个 8 位可编程的向上计数的计数器；一个定时/计数器包含一个 16 位可编程的向上计数的计数器

定时/计数器 0 的时钟来源可以是系统时钟、指令时钟（四分频的系统时钟）、RTC 溢出信号或是外部的时钟源。系统时钟源或是系统时钟四分频通过配置选项来设定。

定时/计数器 1 的时钟来源可以是来自 TMR0 的溢出、系统时钟、时基的溢出信号、系统时钟四分频或是外部时钟源，前三项时钟源由配置选项确定。外部时钟输入允许用户去计算外部事件，测量时间间隔或脉宽、或产生一个精确的时基信号；而使用内部时钟的话，则允许用户去产生一个精确的时基信号。

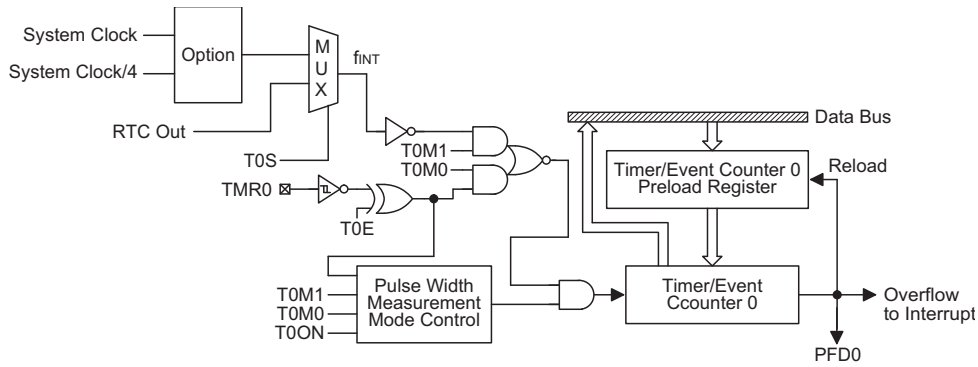
有两个寄存器与定时/计数器 0 相关联，即 TMR0 ([0DH]) 和 TMR0C ([0EH])。写入 TMR0 会将初始值装入到定时/计数器的预置寄存器中，而读 TMR0 则会获得定时/计数器的内容。TMR1C 是定时/计数器控制寄存器，其定义操作模式、计时允许或禁止、触发边缘。

有三个寄存器与定时/计数器 1 相关联，即 TMR1H ([0FH])、TMR1L ([10H]) 和 TMR1C ([11H])。有两个物理寄存器对应 TMR0 (TMR1) 的位置。若写入 TMR1L 只能将数据写入低字节内部缓冲器中，但若写入的是 TMR1H 则可将数据和低字节内部缓冲器的内容写入定时/计数器加载寄存器（16 位）之中。改变定时/计数器加载寄存器的内容，只可被写入 TMR1H 之动作改变，但若写入 TMR1L 则可维持定时/计数器加载寄存器的内容不受改变。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 TMR1H，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。TMR1C 是定时/计数器控制寄存器，其定义某些选项。

T0M0 和 T0M1 (T1M0 和 T1M1) 位，定义工作模式。外部事件计数模式用来记录外部事件，时钟来源由外部 (TMR0, TMR1) 引脚输入。定时器模式是作为一个普通的定时器功能，时钟来源为内部时钟。脉冲宽度测量模式能用来测量外部引脚 (TMR0, TMR1) 上的高电平或低电平的宽度，时钟来源是内部时钟

名称	位	功能
—	0~2	未定义，读取时为“0”
T0E	3	定义定时/计数器 TMR0 作用沿： 外部事件计数器模式(T0M1, T0M0)=(0, 1) 1: 下降沿计数 0: 上升沿计数 脉冲宽度测量模式(T0M1, T0M0)=(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
T0ON	4	允许/禁止定时器计数 (0=禁止, 1=允许)
T0S	5	2 选 1 多通道选择，控制定时/计数器的时钟来源 (0=RTC 输出,1=系统时钟或是系统时钟 4 分频)
T0M0 T0M1	6 7	定义工作模式 (T0M1, T0M0) 01=计数器模式 (外部时钟) 10=定时器模式 (系统时钟) 11=脉冲宽度测量模式 (外部时钟) 00=未定义

TMR0C(0EH) 寄存器

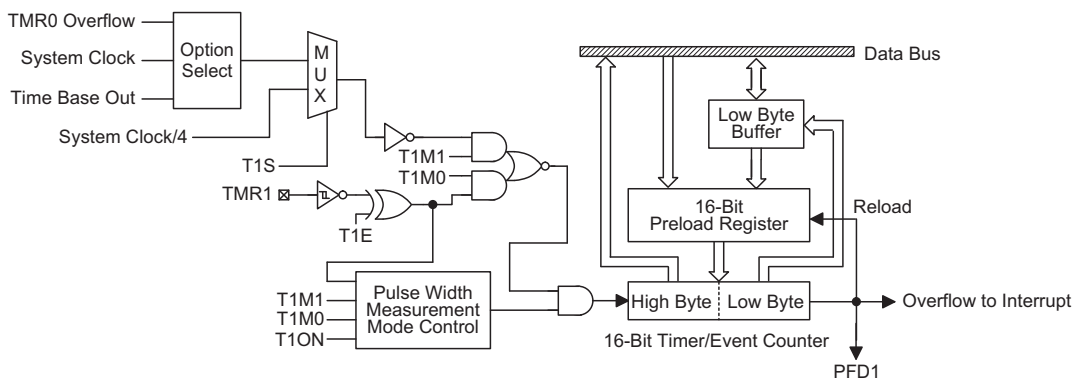


定时/计数器 0

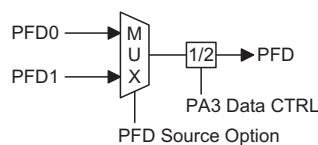
在外部事件计数或定时器模式中，定时/计数器会从当前定时/计数器中的数值开始向上计数，到 0FFH (0FFFFH) 结束。如果产生溢出，计数器会从定时/计数器预置寄存器重新装载计数值并且同时产生相应的中断请求标志 (T0F; INTC0 第 6 位, T1F; INTC1 的第 4 位)。

在脉冲宽度测量模式时，其 T0ON/T1ON 和 T0E/T1E 位皆为 1 时，如果引脚 TMR0/TMR1 接收到一个上升沿信号（如 T0E/T1E 值为 0，则为下降沿信号）时，计数器会开始计数直至 TMR0/TMR1 引脚回到原来的电平为止，并且会将 T0ON/T1ON 清零，只有这种模式 T0ON/T1ON 会自动清零，其它模式 T0ON/T1ON 位只可以用指令清除。计数器停止计数，测量的结果则保留在定时/计数器之中，而且即使再收到一个跳变信号也不会改变。换句话说，脉冲宽度测量模式一次只能测量一个脉冲。只要 T0ON/T1ON 位又被置位，则当引脚 TMR0/TMR1 接到跳变脉冲，测量周期会再次执行下去。在脉冲宽度测量模式中，定时/计数器并不会根据逻辑电平来计数，其根据的标准为信号的跳变沿。一旦发生计数器溢出，计数器会从定时/计数器加载寄存器重新装入，同时还会发出中断请求，这个情况和外部事件计数模式和定时器模式一样。

要启动计数运作，只要将定时器启动 ON 位 (T0ON: TMR0C 的第 4 位; T1ON: TMR1C 的第四位) 被置成为 1。在脉宽测量模式中 T0ON/T1ON 在测量周期结束后自动被清除。但在另外两个模式中，T0ON/T1ON 只能由指令来复位。定时/计数器的溢出是唤醒的信号源之一。并且可由配置选项，设定 PA3 用作 PFD 输出 (可编程分频器)。配置只有一个 PFD 信号 (PFD0 或 PFD1) 提供给 PA3。不管处于何种模式，若写 0 到 ET0I 或 ET1I 位即可禁止相应的中断服务。当 PFD 功能被选时，执行 “CLR [PA].3” 指令来允许 PFD 输出和执行 “SET [PA].3” 指令来禁止 PFD 输出。



定时/计数器 1



PFD 时钟来源选择

名称	位	功能
—	0~2	未定义，读取时为“0”
T1E	3	定义定时/计数器 TMR0 作用沿： 外部事件计数器模式(T1M1, T1M0)=(0, 1) 1: 下降沿计数 0: 上升沿计数 脉冲宽度测量模式(T1M1, T1M0)=(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
T1ON	4	允许/禁止定时器计数 (0=禁止, 1=允许)
T1S	5	2 选 1 多通道选择，控制定时/计数器的时钟来源 (0=配置选项时钟,1=系统时钟 4 分频)
T1M0 T1M1	6 7	定义操作方式 (T1M1, T1M0) 01=计数器模式 (外部时钟) 10=定时器模式 (内部时钟) 11=脉冲宽度测量模式 (外部时钟) 00=未定义

TMR1C(11H) 寄存器

在定时/计数器为关闭的状态下，写数据到定时/计数器的预置寄存器之中，同时也会将数据装入定时/计数器中。但是若定时/计数器处于工作状态，写到定时/计数器的数据只会被保留在定时/计数器的预置寄存器中，直到定时/计数器发生计数溢出才会将数据从预置寄存器加载到定时/计数器寄存器中。

如果要读取定时/计数器数据（读 TMR0/TMR1），计数会被停止。计数停止会导致计数错误，所以程序员必须注意到这一点。

强烈建议首先装载一个指定的值到 TMR0/TMR1 寄存器，然后启动定时/计数器作正常的运作。因为 TMR0/TMR1 的初始值是不确定的。

鉴于定时/计数器的配置，在任何要使用定时/计数器的时候，为了避免不可预料的结果，用户都必须特别注意，第一次开启然后关闭定时/计数器。在这以后定时/计数器开始正常工作。下面给出一个例子，用一个 8 位的定时器和一个 16 位的定时器叠加成一个 24 位的定时器。

START:

```

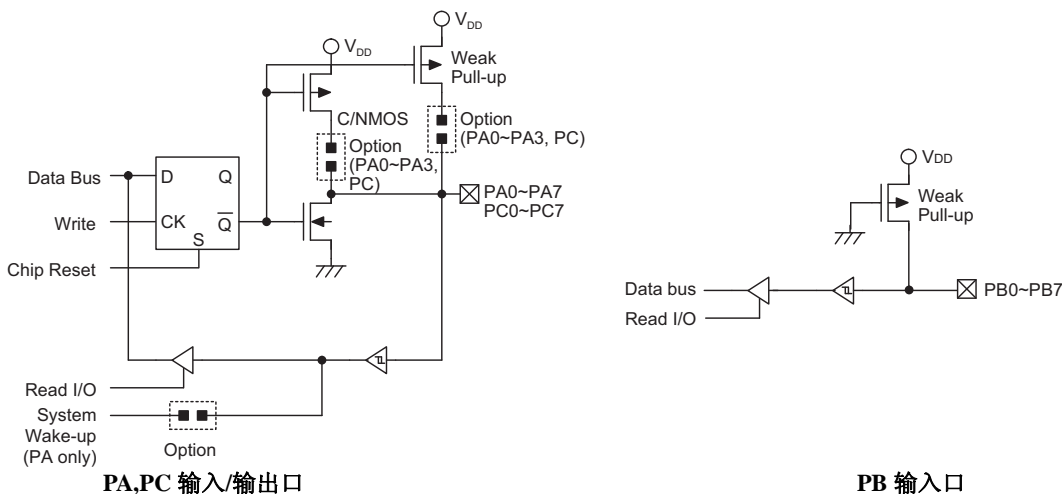
mov    a, 09h           ;置位 ETOI 和 ETI
mov    intc0, a         ;来开放定时/计数器 0 中断和主中断
mov    a, 01h           ;置位 ET1I
mov    intc1, a         ;来开放定时/计数器 1 中断
mov    a, 80h           ;设定定时/计数器 1 工作在计时模式
mov    tmr1c, a         ;并且选择配置时钟为时钟来源
mov    a, 0a0h          ;设定定时/计数器 0 工作在计时模式
mov    tmr0c, a         ;并且选择系统时钟 4 分频为时钟源
set    tmr1c.4          ;第一次开启定时/计数器 1
clr    tmr1c.4          ;第一次关闭定时/计数器 1
mov    a, 00h           ;为 TMR0/TMR1 赋初值
mov    tmr0, a
mov    a, 00h
mov    tmr1l, a
mov    tmr1h, a

```

```
set    tmr0c.4      ;正常工作
set    tmr1c.4
END
```

输入/输出端口

系统包含一个 16 位的双向输入/输出端口 PA 和 PC，和一个 8 位的输入端口 PB。PA、PB 和 PC，其分别对应 RAM 的[12H]，[14H]和[16H]。由配置选项确定，PA0~PA3 可以被设置成 CMOS 输出，或带或不带上拉电阻的 NMOS 输入/输出。PA4~PA7 总是带上拉电阻和作为 NMOS 输入/输出。如果选择为 NMOS 输入，PA 的每一位带都有唤醒功能。PB 只能作为输入端口，PC 可以被配置成 CMOS 输出，或带或不带上拉电阻的 NMOS 输入/输出。PA，PB 和 PC 作为输入端口时，不具有锁存功能，即输入数据必须在 MOV A, [m] (m=12H,14H) 指令的 T2 上升沿被准备好。对 PA 和 PC 口输出而言，所有的数据被锁存并保持不变，直到输出锁存器被改写。



PA,PC 输入/输出口

PB 输入口

当 PA 和 PC 的结构为漏极开路 NMOS 型时，必须要注意：从引脚读数据前，应该先写“1”到相关位来禁止 NMOS 器件。既首先执行“SET[m].i” (i=0~7 仅对 PA 而言) 来禁止相关位的 NMOS 器件，然后“MOV A, [m]”来获得稳定的数据。

芯片复位后，这些输入口都会是高电平或浮空状态（可以由配置选项确定）。每一个输出锁存位都能被“MOV [M],A” (m=12H 或 16H) 指令置位或清除。

某些指令先输入数据然后进行输出操作。例如，SET [m].i，CLR [m].i，CPL [m]和 CPLA [m] 指令，读取整个输入口的状态到 CPU，执行这个被指定的操作（位操作），然后将结果写回这个锁存器或累加器。

当 PA 或 PC 口作为输入输出口时，相应的配置选项应该设为带或不带上拉电阻的 NMOS。一旦设为 CMOS 输出，就不能作为输入输出口使用了。

PA 或 PC 口输入是从相应的 PA 或 PC 管脚读取的。当作为带或不带上拉电阻的 NMOS 时，用户需要小心处理与之有关的读-改-写指令。因为读-改-写指令会先将整个口状态读入，进行相应的操作，然后将结果写回到数据存储器。当进行读操作时，默认的管脚状态（由于负载影响或悬空状态产生）会被读入，就会产生错误。

有三个功能脚与PA口复用：PA0/ BZ，PA1/ \overline{BZ} 和PA3/PFD。BZ和 \overline{BZ} 是蜂鸣驱动输出端，PFD是可编程分频输出端。如果用户需要使用BZ/ \overline{BZ} 或PFD功能，相应的PA口要设为CMOS输出。蜂鸣输出信号由PA0 和PA1 数据寄存器控制，如下表：

PA1 数据寄存器	PA0 数据寄存器	PA0/PA1 管脚状态
0	0	PA0=BZ, PA1= \overline{BZ}
1	0	PA0=BZ, PA1=0
X	1	PA0=0, PA1=0

注意：“X”表示未定义

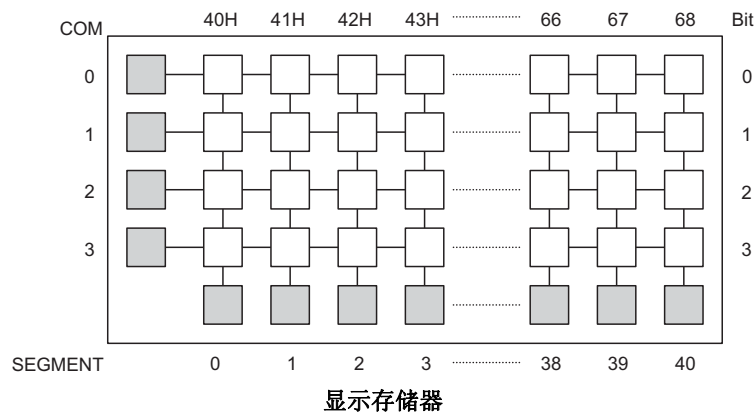
PFD 输出信号由 PA3 数据寄存器和定时/计数器状态控制。PFD 输出信号的频率也依赖定时/计数器溢出周期。PFD 输出频率和控制信号如下表：

定时/计数器	定时/计数器预置值	PA3 数据寄存器	PA3 管脚状态	PFD 输出频率
关闭	X	0	U	X
关闭	X	1	0	X
打开	N	0	PFD	$f_{INT}/[2 \times (256-N)]$
打开	N	1	0	X

注意：“X”表示未定义
“U”表示未知

LCD 显示存储器

系统为 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(bank 1)的 40H~68H 单元。存储器段指针 BANK POINTER (BP; RAM 的 04H 单元) 是 RAM 和 LCD 显示存储器之间切换的开关。当 BP 设置 “01”，通过 MP1 间接寻址写入 40H~68H 的数据将会影响 LCD 的显示。当 BP 被清 “0”，MP1 间接寻址写入的数据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 驱动输出

LCD 驱动器的输出数目可以由配置选项确定为 41×2 或 41×3 或 40×4 (即 $1/2$ 或 $1/3$ 或 $1/4$ 占空比)。HT49R70A-1/HT49C70-1 的 LCD 驱动器偏压产生方式可以分为 “R” 型或 “C” 型，而 HT49C70L 的偏压产生方式只能为 C 型。如果选为 “R” 型，不需要外接电容器；如果为 “C” 型，需要在 C1 和 C2 之间外接一个电容器。HT49R70A-1/HT49C70-1 的 LCD 驱动器偏置电压值可以由配置选项确定为 $1/2$ bias 或 $1/3$ bias，而 HT49C70L 的偏置电压值只能为 $1/2$ bias。如果选择为 $1/2$ bias，则引脚 V2 到地需要接一个电容；如果选择为 $1/3$ bias 的话，则需要两个电容到地分别地接到引脚 V1、V2。

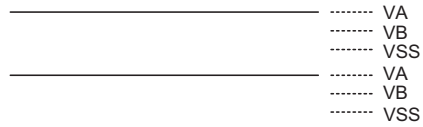
HT49R70A-1/HT49C70-1 的 LCD 偏置电压值选择有两种： $1/2$ bias 或 $1/3$ bias。

HT49R70A-1/HT49C70-1 的 LCD 偏置电压产生方式选择有两种：R 型或 C 型。

During a reset pulse

COM0,COM1,COM2

All LCD driver outputs



Normal operation mode

COM0

COM1

COM2*

LCD segments ON
COM0,1,2 sides are unlighted

Only LCD segments ON
COM0 side are lighted

Only LCD segments ON
COM1 side are lighted

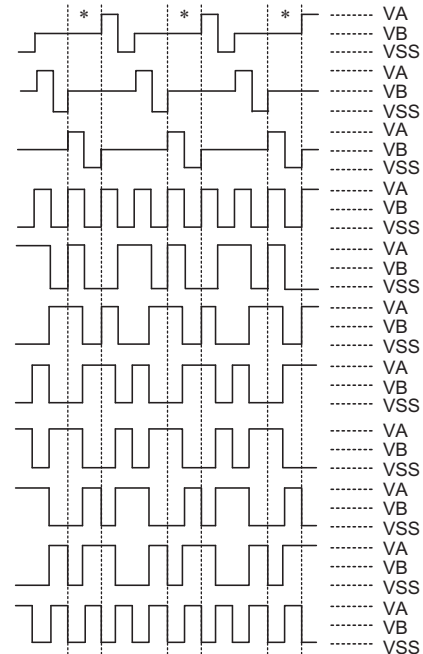
Only LCD segments ON
COM2 side are lighted

LCD segments ON
COM0,1 sides are lighted

LCD segments ON
COM0,2 sides are lighted

LCD segments ON
COM1,2 sides are lighted

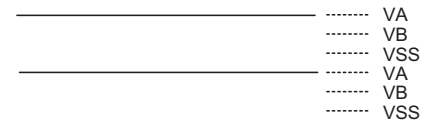
LCD segments ON
COM0,1,2 sides are lighted



HALT Mode

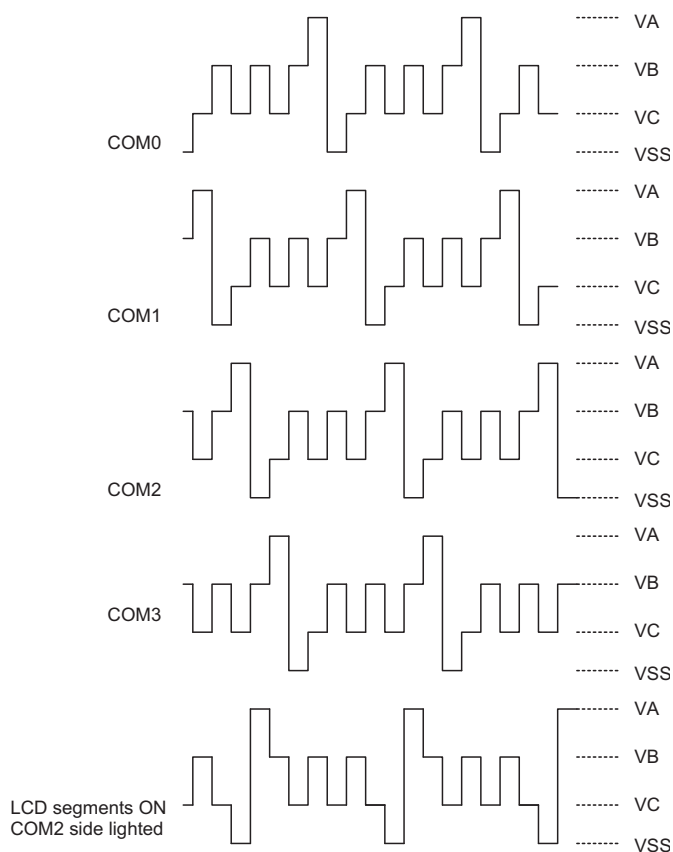
COM0,COM1,COM2*

All LCD driver outputs



Note: "*" Omit the COM2 signal, if the 1/2 duty LCD is used.
VA=VLCD, VB=1/2 VLCD for HT49R70A-1/HT49C70-1
VA=2V2, VB=V2, C type for HT49C70L

LCD 驱动输出 (1/3 duty, 1/2 bias, R/C 型)



Note: 1/4 duty, 1/3 bias, C type: "VA" 3/2 VLCD, "VB" VLCD, "VC" 1/2 VLCD
 1/4 duty, 1/3 bias, R type: "VA" VLCD, "VB" 2/3 VLCD, "VC" 1/3 VLCD
 1/3 bias only for HT49R70A-1/HT49C70-1

LCD 驱动输出

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，这两个功能可以由配置选项选择是否打开。LVD 可以由配置选项打开或关闭。如果选择打开 LVD（低电压检测），用户可以通过 RTCC.3 来打开低电压检测，通过 RTCC.5 来读取低电压检测的状态。否则，低电压检测无效。

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

RTCC 寄存器的定义如下表：

位	标号	读/写	复位	功能
0~2	RT0~RT2	读/写	111B	通过控制 8 选 1 多路器输入来选择实时钟预置寄存器的分频输出比例
3	LVDC*	读/写	0	低电压检测打开/关闭 (1/0)
4	QOSC	读/写	0	32768Hz 晶振快速起振 0/1: 快速/慢速
5	LVDO*	读	0	低电压检测输出 (1/0), 1: 检测到低电压
6~7	—	—	—	未定义, 读出为 0

注：“*”HT49R70A-1/HT49C70-1

RTCC(09H) 寄存器

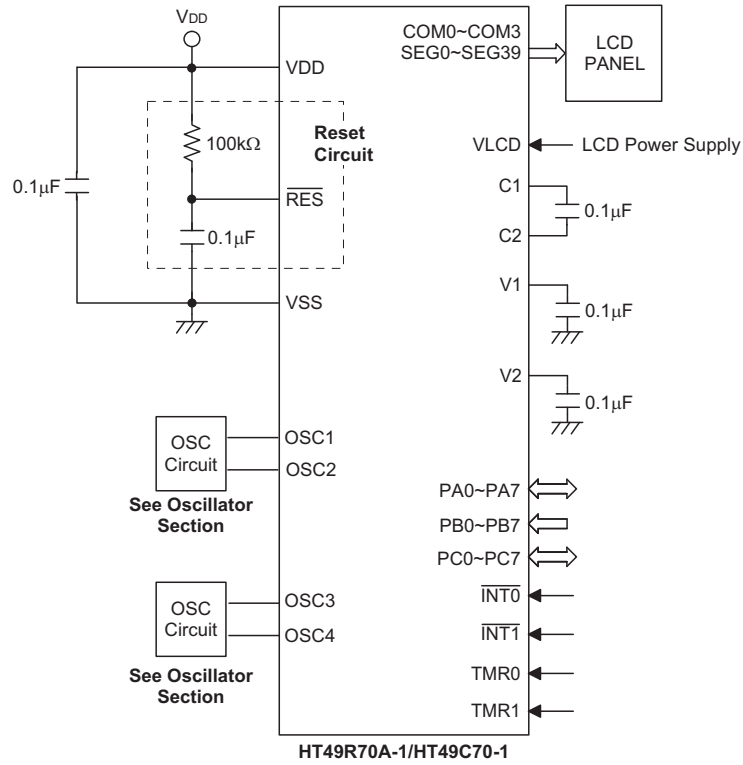
配置选项

下面的表格列出了配置选项，所有选项必须正确定义以确保系统正确的功能。

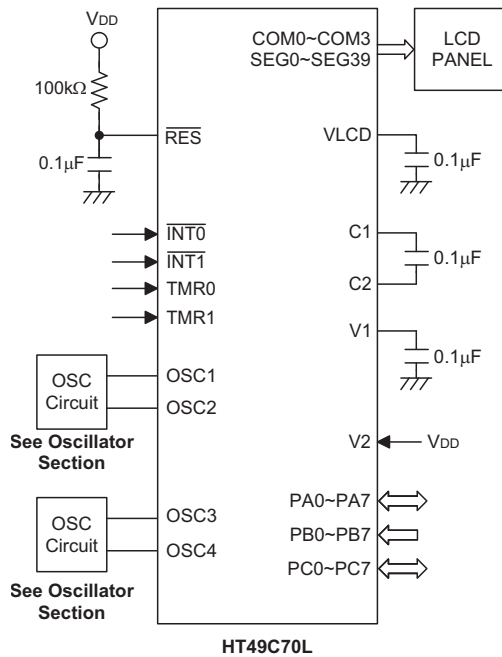
配置选项
OSC 类型的选项。 这个选项确定是否选择一个 RC 或晶体或 32768HZ 晶体来作为系统时钟。
WDT, RTC 和时基的时钟源选项。 有三种选择的方式：系统时钟四分频或 RTC OSC 或 WDT OSC。
WDT 打开/关闭选项。 由配置选项，WDT 打开或关闭。
CLR WDT 次数选项。这个选项定义用指令清除 WDT 的方法。“One time”指用“CLR WDT”指令功能清除 WDT。“Two times”指的是必须要用 CLR WDT1 和 CLR WDT2 两条指令来清除 WDT。
时基溢出周期选项。 时基溢出周期范围从 $\text{clock}/2^{12} \sim \text{click}/2^{15}$ 。“clock”是由配置选项确定的时钟源频率。
蜂鸣器输出频率选项。 有八种输出频率供选择： $\text{clock}/2^2 \sim \text{clock}/2^9$ 。“clock”是由配置选项确定的时钟源频率。
Wake-up 选项。 这个选项来定义激活唤醒功能。外部的 I/O 引脚（仅 PA 具有）电平的下降沿，使芯片从 HALT 状态中唤醒的能力。
上拉电阻的选项。 该选项确定 PA0~PA3 或 PC 的上拉电阻配置。（PB 和 PA4~PA7 总是带上拉电阻）
PA0~PA3 和 PC0~PC7 的 CMOS 或 NMOS 选择。 PA0~PA3 和 PC0~PC7 每位可以分别被选为 CMOS 或 NMOS 的结构。当选择 CMOS 时，相关的引脚只能用作输出。当选择 NMOS 时，相关的引脚能用作输入或输出。（PA4~PA7 总是 NMOS）
定时/计数器 0 的时钟源选项。 有两种选择：系统时钟或系统时钟四分频。
定时/计数器 1 的时钟源选项。 有三种选择：TMR0 的溢出，系统时钟或时基的溢出。
输入输出引脚与其它功能共享选项。 PA0/BZ, PA1/BZ: PA0 和 PA1 能被置成输入输出引脚或蜂鸣器输出。PA3/PFD: PA3 能被置成输入输出引脚或 PFD 输出。
LCD 公共端选项。 有三种选择：2 公共端（1/2 占空比），或 3 公共端（1/3 占空比），或 4 公共端（1/4 占空比）。如果选择了 4 公共端，那么段输出引脚“SEG40”将被作为一个公共端输出。
LCD 偏置电压的选项。 有二种选择：1/2 偏压或 1/3 偏压。
LCD 偏置电压种类的选项。 这个选项确定是 R 型偏压还是 C 型偏压。
LCD 驱动器时钟选项。有七种频率信号选择为 LCD 驱动器电路： $\text{fs}/2^2 \sim \text{fs}/2^8$ 。“fs”是由配置选项确定的时钟源频率。
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
PFD 选项。 如果 PA3 被选作为 PFD 输出，有二种选择；一种是 PFD0 作为 PFD 输出，另一种是 PFD1 作为 PFD 输出。PFD0, PFD1 分别是定时/计数器 0 和定时/计数器 1 的定时器溢出信号。

应用电路

HT49R70A-1/HT49C70-1 应用电路



HT49C70L 应用电路



指令集

简介

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，提供了丰富且灵活的指令，共超过六十条，程序设计者可以事半功倍地实现他们的应用。

为了更加容易理解各种各样的指令码，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”指令。对于跳转指令必须注意的是，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输入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必需具备的能力，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。不同的移位指令可满足不同的应用需要。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

提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设置尤其有用，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出口的 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建立一个表格作为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。

指令集概要

惯例

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

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 ^注	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 ^注	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 ^注	Z,C,AC,OV
SBC A,[m]	ACC 与数据存储器、进位标志的反相减, 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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	无
RLCA [m]	带进位将数据存储器左移一位, 结果放入 ACC	1	C

助记符	说明	指令周期	影响标志位
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 ^注	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 ^注	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 ^注	无
SET [m].i	置位数据存储器的位	1 ^注	无
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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	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 ^注	无
CALL addr	子程序调用	2	无
RET	从子程序返回	2	无
RETA,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 ^注	无
TABRDL [m]	读取最后页的 ROM 内容, 并送至数据存储器 and TBLH	2 ^注	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 ^注	无
SET [m]	置位数据存储器	1 ^注	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,PDF
CLR WDT2	预清除看门狗定时器	1	TO,PDF
SWAP [m]	交换数据存储器的高低字节, 结果放入数据存储器	1 ^注	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: 1、对跳转指令而言, 如果比较的结果牵涉到跳转即需 2 个周期, 如果没有发生跳转, 则只需一个周期。

2、任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

3、对于“CLR WDT1”或“CLR WDT2”指令而言, TO 和 PDF 标志位也许会受执行结果影响, “CLR WDT1”和“CLR WDT2”被连续地执行后, TO 和 PDF 标志位会被清除, 除此之外 TO 和 PDF 标志位保持不变。

指令定义

- ADC A, [m]** Add data memory and carry to the accumulator
 说明：将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + [m] + C$
 影响标志位：OV、Z、AC、C
- ADCM A, [m]** Add the accumulator and carry to the accumulator
 说明：将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
 运算过程： $[m] \leftarrow ACC + [m] + C$
 影响标志位：OV、Z、AC、C
- ADD A, [m]** Add data memory to the accumulator
 说明：将指定的数据存储器和累加器内容相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + [m]$
 影响标志位：OV、Z、AC、C
- ADD A, x** Add immediate data to the accumulator
 说明：将累加器和立即数相加，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC + x$
 影响标志位：OV、Z、AC、C
- ADDM A, [m]** Add the accumulator to the data memory
 说明：将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
 运算过程： $[m] \leftarrow ACC + [m]$
 影响标志位：OV、Z、AC、C
- AND A, [m]** Logical AND accumulator with data memory
 说明：将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC \text{ "AND" } [m]$
 影响标志位：Z
- AND A, x** Logical AND immediate data to the accumulator
 说明：将累加器中的数据和立即数做逻辑与，结果存放到累加器。
 运算过程： $ACC \leftarrow ACC \text{ "AND" } x$
 影响标志位：Z
- ANDM A, [m]** Logical AND data memory with the accumulator
 说明：将指定数据存储器内容和累加器中的数据做逻辑与，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：Z

CALL	addr	Subroutine call
说明:		无条件的调用指定地址的子程序, 此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈, 接着载入指定地址并从新地址执行程序。由于指令需要额外的运算, 所以此指令为 2 个周期。
运算过程:		Stack ← Program Counter+1 Program Counter ← addr
影响标志位:		无
CLR	[m]	Clear data memory
说明:		将指定数据存储器的内容清零。
运算过程:		[m] ← 00H
影响标志位:		无
CLR	[m].i	Clear bit of data memory
说明:		将指定数据存储器的 i 位内容清零。
运算过程:		[m].i ← 0
影响标志位:		无
CLR	WDT	Clear Watchdog Timer
说明:		WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。
运算过程:		WDT ← 00H PDF & TO ← 0
影响标志位:		TO、PDF
CLR	WDT1	Preclear Watchdog Timer
说明:		PDF 和 TO 标志位都被清 0。必须配合 CLR WDT2 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT1, 而没有执行 CLR WDT2 时, PDF 与 TO 保留原状态不变。
运算过程:		WDT ← 00H PDF & TO ← 0
影响标志位:		TO、PDF
CLR	WDT2	Preclear Watchdog Timer
说明:		PDF 和 TO 标志位都被清 0。必须配合 CLR WDT1 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT2, 而没有执行 CLR WDT1 时, PDF 与 TO 保留原状态不变。
运算过程:		WDT ← 00H PDF & TO ← 0
影响标志位:		TO、PDF
CPL	[m]	Complement data memory
说明:		将指定存储器中的每一位取逻辑反, 相当于从 1 变 0 或从 0 变 1。
运算过程:		[m] ← $\overline{[m]}$
影响标志位:		Z

CPLA	[m]	Complement data memory
说明:		将指定数据存储器中的每一位取逻辑反, 相当于从 1 变 0 或从 0 变 1, 结果被存放回累加器且数据寄存器的内容保持不变。
运算过程:		$ACC \leftarrow [\bar{m}]$
影响标志位:		Z
DAA	[m]	Decimal-Adjust accumulator for addition
说明:		将累加器中的内容转换为 BCD (二进制转成十进制) 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”, 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”。BCD 转换实质上是根据累加器和标志位执行 00H, 06H, 60H 或 66H 的加法运算, 结果存放在数据存储器。只有进位标志位 C 受影响, 用来指示原始 BCD 的和是否大于 100, 并可以进行双精度十进制数的加法运算。
操作:		$[m] \leftarrow ACC+00H$ 或 $[m] \leftarrow ACC+06H$ $[m] \leftarrow ACC+60H$ 或 $[m] \leftarrow ACC+66H$
影响标志位:		C
DEC	[m]	Decrement data memory
说明:		将指定数据存储器的内容减 1。
运算过程:		$[m] \leftarrow [m]-1$
影响标志位:		Z
DECA	[m]	Decrement data memory and place result in the accumulator
说明:		将指定数据存储器的内容减 1, 把结果存放回累加器并保持指定数据存储器的内容不变。
运算过程:		$ACC \leftarrow [m]-1$
影响标志位:		Z
HALT		Enter power down mode
说明:		此指令终止程序执行并关掉系统时钟, RAM 和寄存器的内容保持原状态, WDT 计数器和分频器被清“0”, 暂停标志位 PDF 被置位 1, WDT 溢出标志位 TO 被清 0。
运算过程:		$PDF \leftarrow 1$ $TO \leftarrow 0$
影响标志位:		TO、PDF
INC	[m]	Increment data memory
说明:		将指定数据存储器的内容加 1。
运算过程:		$[m] \leftarrow [m]+1$
影响标志位:		Z
INCA	[m]	Increment data memory and place result in the accumulator
说明:		将指定数据存储器的内容加 1, 结果存放回累加器并保持指定的数据存储器内容不变。

运算过程:	$ACC \leftarrow [m]+1$
影响标志位:	Z
JMP addr	Directly jump
说明:	程序计数器的内容无条件地由被指定的地址取代，程序由新的地址继续执行。当新的地址被加载时，必须插入一个空指令周期，所以此指令为 2 个周期的指令。
运算过程:	$PC \leftarrow addr$
影响标志位:	无
MOV A, [m]	Move data memory to the accumulator
说明:	将指定数据存储器的内容复制到累加器。
运算过程:	$ACC \leftarrow [m]$
影响标志位:	无
MOV A, x	Move immediate data to the accumulator
说明:	将 8 位立即数载入累加器。
运算过程:	$ACC \leftarrow x$
影响标志位:	无
MOV [m], A	Move the accumulator data to memory
说明:	将累加器的内容复制到指定的数据存储器。
运算过程:	$[m] \leftarrow ACC$
影响标志位:	无
NOP	No operation
说明:	空操作，顺序执行下一条指令。
运算过程:	$PC \leftarrow PC+1$
影响标志位:	无
OR A, [m]	Logical OR accumulator with data memory
说明:	将累加器中的数据和指定的数据存储器内容逻辑或，结果存放到累加器。
运算过程:	$ACC \leftarrow ACC \text{ "OR" } [m]$
影响标志位:	Z
OR A, x	Logical OR immediate data to the accumulator
说明:	将累加器中的数据和立即数逻辑或，结果存放到累加器。
运算过程:	$ACC \leftarrow ACC \text{ "OR" } x$
影响标志位:	Z
ORM A, [m]	Logical OR data memory with accumulator
说明:	将存在指定数据存储器中的数据和累加器逻辑或，结果放到数据存储器。
运算过程:	$[m] \leftarrow ACC \text{ "OR" } [m]$
影响标志位:	Z

RET	Return from subroutine
说明:	将堆栈寄存器中的程序计数器值恢复, 程序由取回的地址继续执行。
运算过程:	$PC \leftarrow Stack$
影响标志位:	无
RET A, x	Return and place immediate data in the accumulator
说明:	将堆栈寄存器中的程序计数器值恢复且累加器载入指定的立即数, 程序由取回的地址继续执行。
运算过程:	$PC \leftarrow Stack$ $ACC \leftarrow x$
影响标志位:	无
RETI	Return from interrupt
说明:	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被相应, 则这个中断将在返回主程序之前被相应。
运算过程:	$PC \leftarrow Stack$ $EMI \leftarrow 1$
影响标志位:	无
RL [m]	Rotate data memory left
说明:	将指定数据存储器的内容左移 1 位, 且第 7 位移到第 0 位。
运算过程:	$[m].(i+1) \leftarrow [m].i \quad (i=0\sim6)$ $[m].0 \leftarrow [m].7$
影响标志位:	无
RLA [m]	Rotate data memory left and place result in the accumulator
说明:	将指定数据存储器的内容左移 1 位, 且第 7 位移到第 0 位, 结果送到累加器, 而指定数据存储器的内容保持不变。
运算过程:	$ACC.(i+1) \leftarrow [m].i \quad (i=0\sim6)$ $ACC.0 \leftarrow [m].7$
影响标志位:	无
RLC [m]	Rotate data memory left through carry
说明:	将指定数据存储器的内容连同进位标志左移 1 位, 第 7 位取代进位标志且原本的进位标志移到第 0 位。
运算过程:	$[m].(i+1) \leftarrow [m].i \quad (i=0\sim6)$ $[m].0 \leftarrow C$ $C \leftarrow [m].7$
影响标志位:	C
RLCA [m]	Rotate left through carry and place result in the accumulator
说明:	将指定数据存储器的内容连同进位标志左移 1 位, 第 7 位取代进位标志且原本的进位标志移到第 0 位, 移位结果送回累加器, 但是指定数据寄存器的内容保持不变。
运算过程:	$ACC.(i+1) \leftarrow [m].i \quad (i=0\sim6)$

		ACC.0 \leftarrow C
		C \leftarrow [m].7
影响标志位:		C
RR	[m]	Rotate data memory right
说明:		将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。
运算过程:		[m].i \leftarrow [m].(i+1) (i=0~6) [m].7 \leftarrow [m].0,
影响标志位:		无
RRA	[m]	Rotate right and place result in the accumulator
说明:		将指定数据存储器的内容循环右移 1 位, 第 0 位移到第 7 位, 移位结果存放到累加器, 而指定数据存储器的内容保持不变。
运算过程:		ACC.i \leftarrow [m].(i+1) (i=0~6) ACC.7 \leftarrow [m].0
影响标志位:		无
RRC	[m]	Rotate data memory right through carry
说明:		将指定数据存储器的内容连同进位标志右移 1 位, 第 0 位取代进位标志且原本的进位标志移到第 7 位。
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:		C
RRCA	[m]	Rotate right through carry and place result in the accumulator
说明:		将指定数据存储器的内容连同进位标志右移 1 位, 第 0 位取代进位标志且原本的进位标志移到第 7 位, 移位结果送回累加器, 但是指定数据寄存器的内容保持不变。
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:		C
SBC	A,[m]	Subtract data memory and carry from the accumulator
说明:		将累加器减去指定数据存储器的内容以及进位标志的反, 结果存放到累加器。如果结果
果结果		为负, C 标志位清除为 0, 反之结果为正或 0, C 标志位设置为 1。
运算过程:		ACC \leftarrow ACC - [m] - \overline{C}
影响标志位:		OV、Z、AC、C
SBCM	A,[m]	Subtract data memory and carry from the accumulator
说明:		将累加器减去指定数据存储器的内容以及进位标志的反, 结果存放到数据存储器。如果
如果		结果为负, C 标志位清除为 0, 反之结果为正或 0, C 标志位设置为 1。

运算过程: $ACC \leftarrow ACC - [m] - \bar{C}$

影响标志位: OV、Z、AC、C

SDZ [m] Skip if decrement data memory is 0

说明: 将指定的数据存储器的内容减 1, 判断是否为 0, 若为 0 则跳过下一条指令, 由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果不为 0, 则程序继续执行下一条指令。

运算过程: $[m] \leftarrow [m] - 1$, 如果 $[m]=0$ 跳过下一条指令执行

影响标志位: 无

SDZA [m] Decrement data memory and place result in ACC, skip if 0

说明: 将指定数据存储器内容减 1, 判断是否为 0, 如果为 0 则跳过下一条指令, 此结果将存放到累加器, 但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果不为 0, 则程序继续执行下一条指令。

运算过程: $ACC \leftarrow [m] - 1$, 如果 $ACC=0$ 跳过下一条指令执行

影响标志位: 无

SET [m] Set data memory

说明: 将指定数据存储器的每一位设置为 1。

运算过程: $[m] \leftarrow FFH$

影响标志位: 无

SET [m].i Set bit of data memory

说明: 将指定数据存储器的第 i 位设置为 1。

运算过程: $[m].i \leftarrow 1$

影响标志位: 无

SIZ [m] Skip if increment data memory is 0

说明: 将指定的数据存储器的内容加 1, 判断是否为 0, 若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果不为 0, 则程序继续执行下一条指令。

运算过程: $[m] \leftarrow [m] + 1$, 如果 $[m]=0$ 跳过下一条指令执行

影响标志位: 无

SIZA [m] Increment data memory and place result in ACC, skip if 0

说明: 将指定数据存储器的内容加 1, 判断是否为 0, 如果为 0 则跳过下一条指令, 此结果会被存放到累加器, 但是指定数据存储器的内容不变。由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果不为 0, 则程序继续执行下一条指令。

运算过程: $ACC \leftarrow [m] + 1$, 如果 $ACC=0$ 跳过下一条指令执行

影响标志位: 无

SNZ	[m].i	Skip if bit I of the data memory is not 0
说明:		判断指定数据存储器的第 i 位, 若不为 0, 则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果为 0, 则程序继续执行下一条指令。
运算过程:		如果[m].i≠0, 跳过下一条指令执行
影响标志位:		无
SUB	A, [m]	Subtract data memory from the accumulator
说明:		将累加器的内容减去指定的数据存储器的数据, 把结果存放到累加器。如果结果为负, C 标志位清除为 0, 反之结果为正或 0, C 标志位设置为 1。
运算过程:		$ACC \leftarrow ACC - [m]$
影响标志位:		OV、Z、AC、C
SUBM	A, [m]	Subtract data memory from the accumulator
说明:		将累加器的内容减去指定数据存储器的数据, 结果存放到指定的数据存储器。如果结果为负, C 标志位清除为 0, 反之结果为正或 0, C 标志位设置为 1。
运算过程:		$[m] \leftarrow ACC - [m]$
影响标志位:		OV、Z、AC、C
SUB	A, x	Subtract immediate data from the accumulator
说明:		将累加器的内容减去立即数, 结果存放到累加器。如果结果为负, C 标志位清除为 0, 反之结果为正或 0, C 标志位设置为 1。
运算过程:		$ACC \leftarrow ACC - x$
影响标志位:		OV、Z、AC、C
SWAP	[m]	Swap nibbles within the data memory
说明:		将指定数据存储器的低 4 位和高 4 位互相交换。
运算过程:		$[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$
影响标志位:		无
SWAPA	[m]	Swap data memory and place result in the accumulator
说明:		将指定数据存储器的低 4 位和高 4 位互相交换, 再将结果存放到累加器且指定数据寄存器的数据保持不变。
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:		无
SZ	[m]	Skip if data memory is 0
说明:		判断指定数据存储器的内容是否为 0, 若为 0, 则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期, 所以此指令为 2 个周期的指令。如果结果不为 0, 则程序继续执行下一条指令。
运算过程:		如果[m] = 0, 跳过下一条指令执行
影响标志位:		无
SZA	[m]	Move data memory to ACC, skip if 0

说明: 将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程: $ACC \leftarrow [m]$ ，如果 $[m] = 0$ ，跳过下一条指令执行

影响标志位: 无

SZ [m]. i Skip if bit I of the data memory is 0

说明: 判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

运算过程: 如果 $[m].i = 0$ ，跳过下一条指令执行

影响标志位: 无

TABRDC [m] Move the ROM code(current page) to TBLH and data memory

说明: 将表格指针 TBLP 所指的程序代码低字节（当前页）移至指定的数据存储器且将高字节移至 TBLH。

运算过程: $[m] \leftarrow$ 程序代码（低字节）

$TBLH \leftarrow$ 程序代码（高字节）

影响标志位: 无

TABRDL [m] Move the ROM code(last page) to TBLH and data memory

说明: 将表格指针 TBLP 所指的程序代码低字节（最后一页）移至指定的数据存储器且将高字节移至 TBLH。

运算过程: $[m] \leftarrow$ 程序代码（低字节）

$TBLH \leftarrow$ 程序代码（高字节）

影响标志位: 无

XOR A, [m] Logical XOR accumulator with data memory

说明: 将累加器的数据和指定的数据存储器内容逻辑异或，结果存放到累加器。

运算过程: $ACC \leftarrow ACC \text{ "XOR" } [m]$

影响标志位: Z

XORM A, [m] Logical XOR data memory with accumulator

说明: 将累加器的数据和指定的数据存储器内容逻辑异或，结果放到数据存储器。

运算过程: $[m] \leftarrow ACC \text{ "XOR" } [m]$

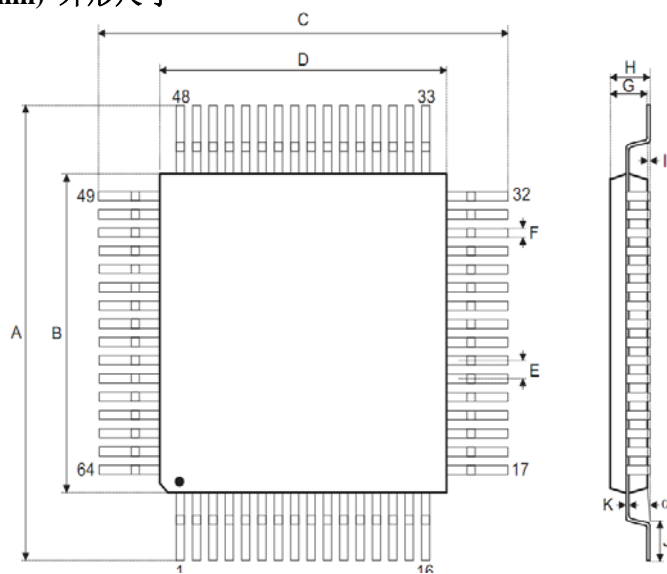
影响标志位: Z

XOR A, x Logical XOR immediate data to the accumulator

说明: 将累加器的数据与立即数逻辑异或，结果存放到累加器。

运算过程: $ACC \leftarrow ACC \text{ "XOR" } x$

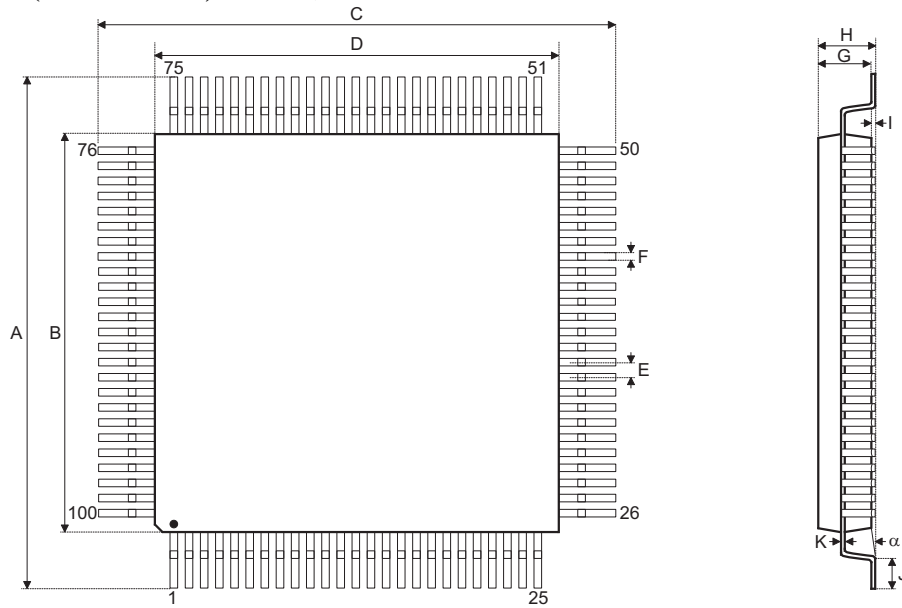
影响标志位: Z

封装尺寸
64-pin LQFP (7mm×7mm) 外形尺寸


符号	尺寸 (单位: inch)		
	最小	一般	最大
A	0.350	—	0.358
B	0.272	—	0.280
C	0.350	—	0.358
D	0.272	—	0.280
E	—	0.016	—
F	0.005	—	0.009
G	0.053	—	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	—	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸 (单位: mm)		
	最小	一般	最大
A	8.90	—	9.10
B	6.90	—	7.10
C	8.90	—	9.10
D	6.90	—	7.10
E	—	0.40	—
F	0.13	—	0.23
G	1.35	—	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	—	0.75
K	0.09	—	0.20
α	0°	—	7°

100-pin LQFP (14mm×14mm)外形尺寸



符号	尺寸(单位: inch)		
	最小	正常	最大
A	0.626	—	0.634
B	0.547	—	0.555
C	0.626	—	0.634
D	0.547	—	0.555
E	—	0.020	—
F	—	0.008	—
G	0.053	—	0.057
H	—	—	0.063
I	—	0.004	—
J	0.018	—	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸(单位: mm)		
	最小	典型	最大
A	15.90	—	16.10
B	13.90	—	14.10
C	15.90	—	16.10
D	13.90	—	14.10
E	—	0.50	—
F	—	0.20	—
G	1.35	—	1.45
H	—	—	1.60
I	—	0.10	—
J	0.45	—	0.75
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（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: 86-755-8616-9908, 86-755-8616-9308

传真: 86-755-8616-9722

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

46729 Fremont Blvd., Fremont, CA 94538, USA

电话: 1-510-252-9880

传真: 1-510-252-9885

网站: www.holtek.com

Copyright © 2011 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>