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

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

技术相关信息

- 应用范例
  - HA0075S MCU 复位电路及振荡电路应用

特性

CPU 特性

- 工作电压：
  f_{SYS} = 4MHz：2.2V～5.5V
  f_{SYS} = 8MHz：3.0V～5.5V
  f_{SYS} = 12MHz：4.5V～5.5V
- 在 V_{DD}=5V，系统频率为 12MHz 时，指令周期为 0.33μs
- 休眠模式和唤醒功能可降低功耗
- 振荡模式：
  外部高频晶振 – HXT
  外部 RC – ERC
  内部高频 RC – HIRC
  外部低频晶振 – LXT
- 3 种工作模式：正常模式，低速模式，休眠模式
- 内部集成 4MHz，8MHz 和 12MHz 振荡器，不需要增加外部元器件
- 2048×15 位的程序存储器 ROM
- 96×8 位的数据存储器 RAM
- 看门狗定时器
- LIRC 振荡用于看门狗时钟
- 所有指令都可在 1 个或 2 个指令周期内完成
- 查表指令
- 63 条功能强大的指令系统
- 4 层硬件堆栈
- 位操作指令
- 低电压复位功能
- 提供多种封装类型

周边特性：

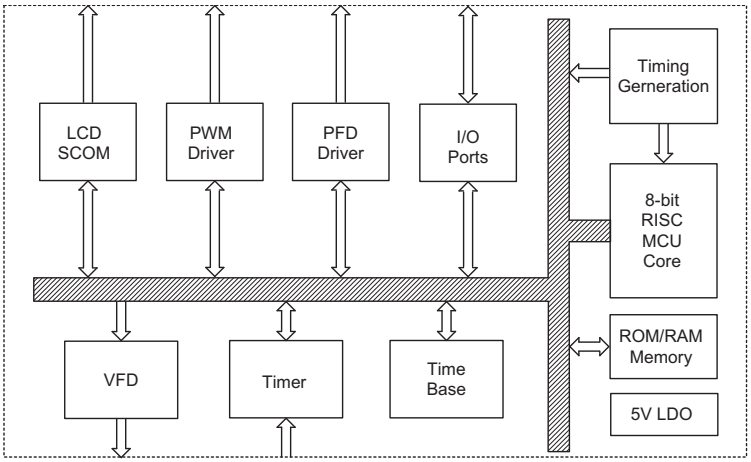
- 21 个双向输入/输出口
- 4 个软件控制 SCOM 口 1/2 bias LCD 驱动
- 一个与 I/O 口复用的外部中断输入
- 一个 8 位可编程定时/计数器，具有溢出中断和预分频功能
- 时基功能
- PFD 功能
- 集成直流 24V 转 5V 的 LDO 稳压器
- 蜂鸣器和灯丝 5V 转 24V 输出电平转换器
- 24 位移位寄存器/锁存器，用于驱动 VFD 面板 24 栅格/段输出。
- 集成 3 线串行 VFD 接口，用于栅格/段显示控制
- 52-pin QFP 封装

概述

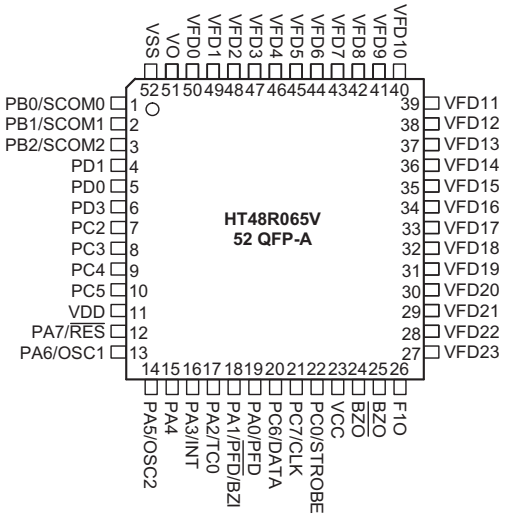
HT48R065V 是一款 24V VFD 型 8 位高性能精简指令集型单片机，应用相当广泛。秉承 HOLTEK 单片机具有的低功耗、I/O 灵活、定时器功能、振荡类型可选、休眠和唤醒功能、看门狗和低电压复位等丰富的功能选项，使得该单片机具有极高的性价比，其内部集成了系统振荡器 HIRC，提供三种频率选择，可以广泛适用于各种应用，例如工业控制，消费类产品，家用电器子系统控制等。

方框图

以下为主要功能模块的方框图



引脚图



引脚说明

引脚名称	功能	OPT	I/T	O/T	说明
PA0/ PFD	PA0	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	PFD	CTRL0	—	CMOS	PFD 输出
PA1//$\overline{PFD}$/BZI	PA1	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	$\overline{PFD}$	CTRL0	—	CMOS	PFD 互补输出
	BZI	—	ST	—	VFD 控制接口
PA2/TC0	PA2	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	TC0	—	ST	—	外部定时器 0 时钟输入脚
PA3/INT	PA3	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	—	ST	—	外部中断输入
PA4/TC1	PA4	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	TC1	—	ST	—	外部定时器 1 时钟输入脚-HT48R065 没有 TC1
PA5/OSC2	PA5	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OSC2	CO	—	OSC	振荡器引脚
PA6/OSC1	PA6	PAPU PAWK	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OSC1	CO	OSC	—	振荡器输入脚
PA7/$\overline{RES}$	PA7	PAWK	ST	NMOS	通用 I/O 口，可通过寄存器设置唤醒功能
	$\overline{RES}$	CO	ST	—	复位输入脚
PB0/SCOM0	PB0	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM0	SCOMC	—	SCOM	软件控制 1/2bias 的 LCD COM 口
PB1/SCOM1	PB1	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM1	SCOMC	—	SCOM	软件控制 1/2bias 的 LCD COM 口
PB2/SCOM2	PB2	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM2	SCOMC	—	SCOM	软件控制 1/2bias 的 LCD COM 口
PC0/STROBE	PC3	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STROBE	—	ST	—	VFD 控制接口
PC1/F1	PC1	—	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	F1	—	ST	—	带斯密特触发器的灯丝控制输入脚
PC2~PC5	PCn	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PC6/DATA	PC6	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	DATA	—	ST	—	VFD 控制接口
PC7/CLK	PC7	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CLK	—	ST	—	VFD 控制接口
PD0,PD1,PD3	PDn	PDPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
VDD	VDD	—	PWR	—	正电源
VSS	VSS	—	PWR	—	负电源、接地

引脚名称	功能	OPT	I/T	O/T	说明
VFD23~VFD0	VFDn	—	—	HV	VFD 面板的高压栅格/段输出脚
VCC	VCC	—	PWR	—	高压电源的正极，用于驱动 VFD 灯丝引脚 F1O,BZO 和$\overline{BZO}$输出。建议外部连接 10uF 的电容到 PCB 板，以减少雷击电压影响。
VO	VO	—	—	PWR	LDO 稳压输出（5V），建议外部连接电容到 PCB 板
BZO, $\overline{BZO}$	BZO	—	—	HV	高压蜂鸣器互补输出信号脚
	$\overline{BZO}$	—	—	HV	
F1O	F1O	—	—	HV	高压灯丝输出信号脚

注意： I/T：输入类型； O/T：输出类型
OPT：通过配置选项（CO）或者寄存器选项来配置
PWR：电源；CO：配置选项
ST：斯密特触发输入； CMOS：CMOS 输出
SCOM：软件控制的 LCD COM
HXT：高频晶体振荡器
LXT：低频晶体振荡器
HV：高压输出脚
用于 VFD 驱动接口时，输入/输出引脚需设置为输出。

极限参数

电源供应电压 ..... Vss-0.3V 至 Vss +6.0V
端口输入电压 ..... Vss-0.3V 至 V_{DD}+0.3V
I_{OL} 总电流 100mA
总功耗 500mW
储存温度 ..... -50℃至 125℃
工作温度..... -40℃至 85℃
I_{OH} 总电流 . ..... -100mA

注意：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℃

符号	参数	测试条件			最小	典型	最大	单位
		V_{DD}	V_{CC}	条件				
V_{DD}	工作电压	—	—	f_{SYS} = 4MHz	2.2	—	5.5	V
				f_{SYS} = 8MHz	3.0	—	5.5	V
				f_{SYS} = 12MHz	4.5	—	5.5	V
I_{DD1}	工作电流 (HXT，HIRC，ERC)	3V	—	无负载, f_{SYS} = 4MHz	—	0.8	1.2	mA
		5V	—		—	1.5	2.25	mA
I_{DD2}	工作电流 (HXT，HIRC，ERC)	3V	—	无负载, f_{SYS} = 8MHz	—	1.4	2.1	mA
		5V	—		—	2.8	4.2	mA
I_{DD3}	工作电流 (HXT，HIRC，ERC)	5V	—	无负载, f_{SYS} = 12MHz	—	4	6	mA
I_{DD4}	工作电流 (HIRC+LXT，低速模式 LXTLP=1)	3V	—	无负载, f_{SYS} = 32768Hz （LXT on OSC1/OSC2，LVR 关闭, LXTLP=1）	—	5	10	μA
		5V	—		—	12	24	μA
I_{STB1}	静态电流 (LIRC On, LXT Off)	3V	—	无负载，休眠模式	—	—	5	μA
		5V	—		—	—	10	μA
I_{STB2}	静态电流 (LIRC Off, LXT Off)	3V	—	无负载，休眠模式	—	—	1	μA
		5V	—		—	—	2	μA
I_{STB3}	静态电流 (LIRC Off, LXT On, LXTLP=1)	3V	—	无负载，休眠模式 （LXT on OSC1/OSC2）	—	—	5	μA
		5V	—		—	—	10	μA
V_{IL1}	I/O、TCn 和 INT 的低电平输入电压	—	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	I/O、TCn 和 INT 的高电平输入电压	—	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	低电平输入电压 ($\overline{RES}$)	—	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	高电平输入电压 ($\overline{RES}$)	—	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{LVR1}	低电压复位 1	—	—	V_{LVR} = 4.2V	3.98	4.2	4.42	V
V_{LVR2}	低电压复位 2	—	—	V_{LVR} = 3.15V	2.98	3.15	3.32	V
V_{LVR3}	低电压复位 3	—	—	V_{LVR} = 2.1V	1.98	2.1	2.22	V
I_{OL1}	I/O 口灌电流	3V	—	V_{OL} = $0.1V_{DD}$	4	8	—	mA
		5V	—		10	20	—	mA
I_{OH1}	I/O 口源电流	3V	—	V_{OH} = $0.9V_{DD}$	-2	-4	—	mA
		5V	—		-5	-10	—	mA
I_{OL}	PA7 灌电流	5V	—	V_{OL} = $0.1V_{DD}$	2	3	—	mA
R_{PH}	上拉电阻	3V	—	—	20	60	100	kΩ
		5V	—	—	10	30	50	kΩ

符号	参数	测试条件			最小	典型	最大	单位
		V_{DD}	V_{CC}	条件				
I_{SCOM}	SCOM 驱动电流	5V	—	SCOMC，ISEL[1:0] = 00	17.5	25.0	32.5	μA
				SCOMC，ISEL[1:0] = 01	35	50	65	μA
				SCOMC，ISEL[1:0] = 10	70	100	130	μA
				SCOMC，ISEL[1:0] = 11	140	200	260	μA
V_{SCOM}	LCD COM 口输出电压	5V	—	无负载	0.475	0.500	0.525	V_{DD}
V_{CC}	VFD,F1O,BZO, $\overline{BZO}$ 输出电源	—	—	—	12	—	24	V
I_{CC1}	逻辑工作电流 1	5V	18V	无负载，VFD 输出均为低电平，CLK=100kHz	—	70	110	μA
			24V		—	90	135	μA
I_{CC2}	逻辑工作电流 2	5V	18V	无负载，VFD 输出均为低电平，CLK=100kHz	—	70	110	μA
			24V		—	90	135	μA
I_{CC3}	蜂鸣器工作电流	5V	18V	无负载，BZI 输入为 50 kHz	—	100	150	μA
			24V		—	120	180	μA
I_{CC4}	灯丝工作电流	5V	18V	无负载，F1 输入为 50 kHz	—	80	120	μA
			24V		—	100	150	μA
I_{STB4}	静态电流（LDO 一直开启，WDT 使能/除能）	5V	18V	无负载	—	70	105	μA
			24V		—	90	135	μA
I_{OL2}	F1O 灌电流	5V	18V	V_{OL}=0.1V_{CC}	2.5	5.0	—	mA
			24V		5.0	10.0	—	mA
I_{OH2}	F1O 源电流	5V	18V	V_{OH}=0.9V_{CC}	-15	-30	—	mA
			24V		-22	-25	—	mA
I_{OL3}	BZO/$\overline{BZO}$灌电流	5V	18V	V_{OL}=0.1V_{CC}	15	30	—	mA
			24V		25	50	—	mA
I_{OH3}	BZO/$\overline{BZO}$源电流	5V	18V	V_{OH}=0.9V_{CC}	-15	-30	—	mA
			24V		-25	-50	—	mA
I_{OL4}	栅格/段灌电流	5V	18V	V_{OL}=0.1V_{CC}	2.5	5.0	—	mA
			24V		5.0	10.0	—	mA
I_{OH4}	栅格/段源电流	5V	18V	V_{OH}=0.9V_{CC}	-6	-12	—	mA
			24V		-10	-20	—	mA

注意：静态电流(I_{STB1}~I_{STB3})和 I_{DD4} 的测试条件为：设置所有输入/输出口为输入模式，并且连接至 V_{DD}。

交流电气特性

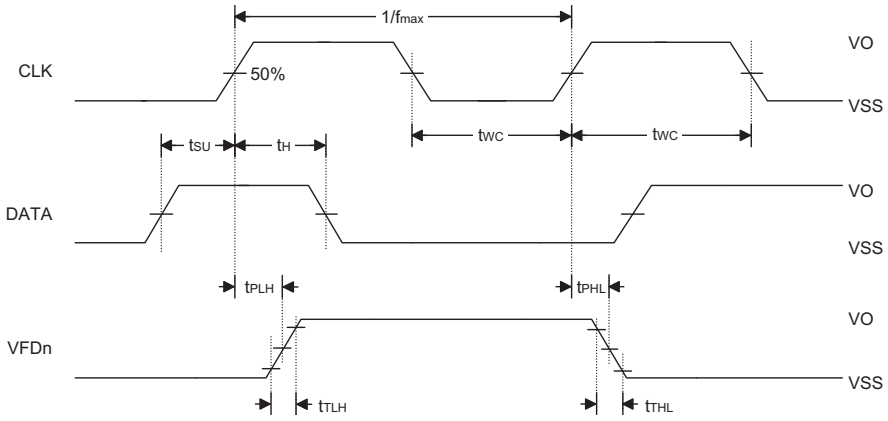
Ta=25℃

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	—	2.2V ~ 5.5V	32	—	4000	kHz
			3.0V ~ 5.5V	32	—	8000	kHz
			4.5V ~ 5.5V	32	—	12000	kHz
f_{HIRC}	系统时钟（HIRC）	3V/5V	Ta=25℃	-2%	4	+2%	MHz
		3V/5V	Ta=25℃	-2%	8	+2%	MHz
		5V	Ta=25℃	-2%	12	+2%	MHz
		3V/5V	Ta=0~70℃	-5%	4	+5%	MHz
		3V/5V	Ta=0~70℃	-5%	8	+5%	MHz
		5V	Ta=0~70℃	-5%	12	+5%	MHz
		2.2V~3.6V	Ta=0~70℃	-8%	4	+8%	MHz
		3.0V~5.5V	Ta=0~70℃	-8%	4	+8%	MHz
		3.0V~5.5V	Ta=0~70℃	-8%	8	+8%	MHz
		4.5V~5.5V	Ta=0~70℃	-8%	12	+8%	MHz
		2.2V~3.6V	Ta=-40℃~85℃	-12%	4	+12%	MHz
		3.0V~5.5V	Ta=-40℃~85℃	-12%	4	+12%	MHz
		3.0V~5.5V	Ta=-40℃~85℃	-12%	8	+12%	MHz
		4.5V~5.5V	Ta=-40℃~85℃	-12%	12	+12%	MHz
f_{ERC}	系统时钟（ERC）	5V	Ta=25℃，R=120kΩ*	-2%	4	+2%	MHz
		5V	Ta=0~70℃，R=120kΩ*	-5%	4	+5%	MHz
		5V	Ta=-40℃~85℃，R=120kΩ*	-7%	4	+7%	MHz
		2.2V~5.5V	Ta=-40℃~85℃，R=120kΩ*	-11%	4	+11%	MHz
f_{LXT}	系统时钟（LXT）	—	—	—	32768	—	Hz
f_{TIMER}	定时器输入频率（TCn）	—	2.2V~5.5V	0	—	4000	kHz
			3.0V~5.5V	0	—	8000	kHz
			4.5V~5.5V	0	—	12000	kHz
f_{LIRC}	LIRC 振荡器	3V	—	5	10	15	kHz
		5V	—	6.5	13	19.5	kHz
t_{RES}	外部复位低电压脉宽	—	—	1	—	—	μs
t_{SST}	系统启动延时周期	—	HXT/LXT	—	1024	—	t_{SYS}
			ERC/IRC（配置选项控制）	—	2	—	t_{SYS}
				—	1024	—	t_{SYS}
t_{INT}	中断脉冲宽度	—	—	1	—	—	μs
t_{LVR}	低电压复位宽度	—	—	0.25	1	2	ms
t_{RSTD}	复位延时时间	—	—	—	100	—	ms
t_{PHL}	传播延迟时间（Clock to VFD Output）	—	V_{CC}=15V	—	100	200	ns
t_{PLH}	传播延迟时间（Strobe to VFD Output）			—	100	200	ns
t_{THL} t_{TLH}	传输时间	—	V_{CC}=15V	—	40	80	ns

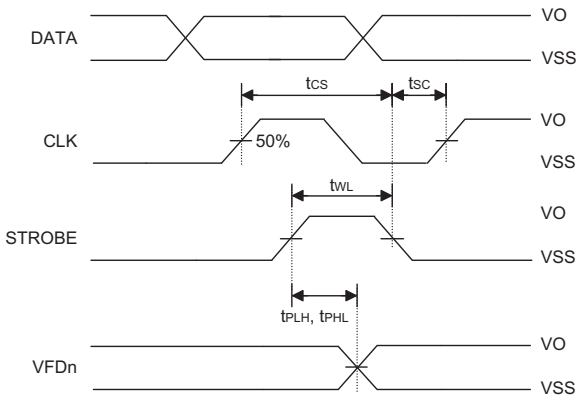
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
t_{SU}	数据建立时间	—	V_{CC}=15V	—	10	20	ns
t_{CS}	建立时间（Clock to Strobe）	—	V_{CC}=15V	—	10	20	ns
t_H	保持时间（Data to Clock）	—	V_{CC}=15V	—	10	20	ns
t_{SC}	保持时间（Clock to Strobe）	—	V_{CC}=15V	—	75	150	ns
t_r,t_f	时钟输入上升沿或下降沿时间	—	V_{CC}=15V	—	—	20	ns
t_{WC}	时钟脉冲宽度	—	V_{CC}=15V	—	40	83	ns
t_{WL}	选通脉冲宽度	—	V_{CC}=15V	—	35	70	ns
f_{max}	最大时钟输入频率	—	V_{CC}=15V	—	8	—	MHz

注意：
1. $t_{SYS} = 1/f_{SYS}$
2. *: 表示电阻的公差会影响外部 RC 的频率，建议使用精密度较高的电阻。
3. 为了保持内部 HIRC 振荡器频率的准确性，需要在 VDD 和 VSS 之间接入一个 0.1μF 的去耦电容，并且尽可能地靠近单片机。

交流波形



数据传播延迟，建立和保持时间



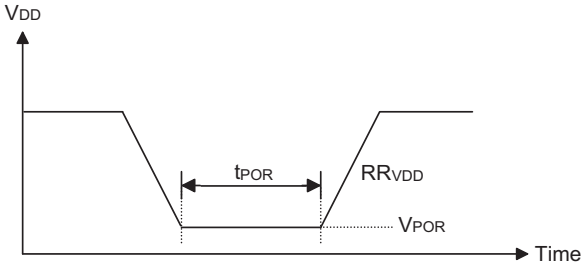
选通脉冲传播延迟，建立和保持时间

LDO 特性

符号	参数	测试条件			最小	典型	最大	单位
		V_{DD}	V_{CC}	条件				
V_O	LDO 输出电压	—	—	—	4.7	5.0	5.3	V
I_{OUT}	最大 LDO 输出电流	—	—	VCC≥5V	10	—	—	mA
ΔVLNR	线性调整率	—	—	V_{IN}=(V_{OUT}+0.1V)~24V, I_{OUT}=1mA	—	0.2	—	%/V
ΔVLDR	负载调整率	—	—	I_{OUT}=100μA~20 mA C_{OUT}=10pF	—	0.1	0.2	%/mA
VDRO	输入输出电压差	—	—	I_{OUT}=1mA	25	30	35	mV

上电复位特性

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{POR}	上电复位电压	—	—	—	—	100	mV
RR_{VDD}	上电复位电压速率	—	—	0.035	—	—	V/ms
t_{POR}	V_{DD} 保持为 V_{POR} 的最小时间	—	—	1	—	—	ms

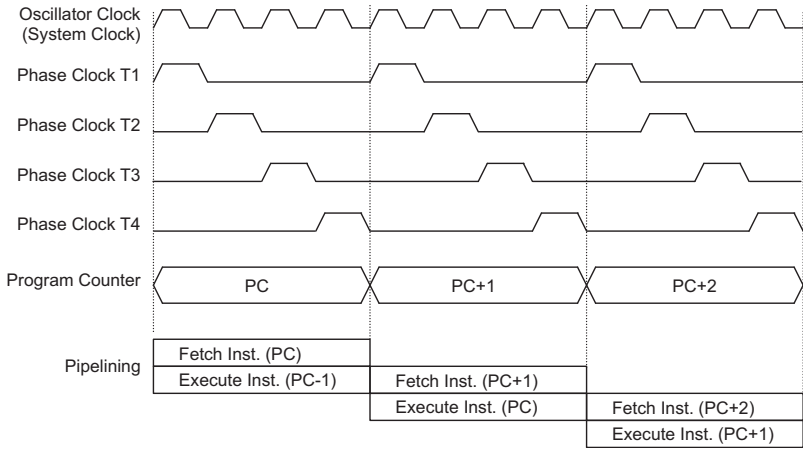


系统结构

内部系统结构是盛群单片机具有良好性能的主要因素。由于采用 RISC 结构，此系列单片机具有高运算速度和高性能的特点。通过流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令外，其它指令都能在一个指令周期内完成。8 位 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器或 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供具有最大可靠度和灵活性的实用性控制系统时，仅需要少数的外部器件。

时序和流水线结构

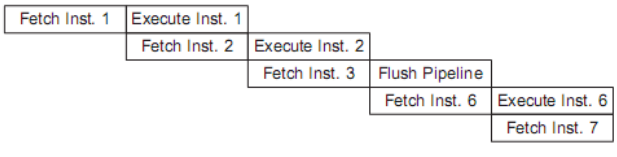
主系统时钟由晶体/陶瓷振荡器，或者 RC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并抓取一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时间周期构成一个指令周期。虽然指令的抓取和执行发生在连续的指令周期，但单片机流水线结构会保证指令在一个指令周期内被有效执行。除非程序计数器的内容被改变，如子程序的调用或者跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



系统时序和流水线

如果指令牵涉到分支，例如跳转或调用等指令，则需要两个指令周期才能完成指令执行。需要一个额外周期的原因是程序先用一个周期取出实际要跳转或调用的地址，再用另一个周期去实际执行分支动作，因此用户需要特别考虑额外周期的问题，尤其是在执行时间要求较严格的时候。

```
1            MOV A,[12H]
2            CALL DELAY
3            CPL [12H]
4            :
5            :
6  DELAY:    NOP
```



指令捕捉

程序计数器 PC

在程序执行期间，程序计数器用来指向下一个要执行的指令地址。除了 “JMP”和“CALL”指令需要跳转到一个非连续的程序存储器地址之外，它会在每条指令执行完成以后自动加一。选择不同型号的单片机，程序寄存器的宽度会因程序存储器的容量的不同而不同。然而只有较低的 8 位，即所谓的程序低字节寄存器 PCL，可以被用户直接读写。

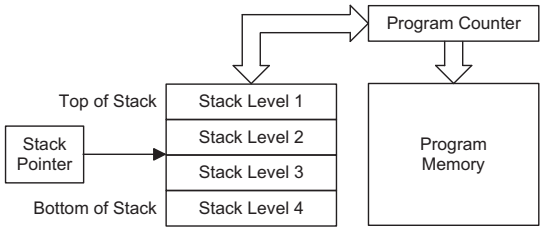
当执行的指令要求跳转到不连续的地址时，如跳转指令、子程序调用、中断或者复位等，单片机通过加载所需要的位址到程序寄存器来控制程序，对于条件跳转指令，一旦条件符合，在当前指令执行时取得的下一条指令将会被舍弃，而由一个空指令周期来取代。

程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。

程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。有关 PCL 寄存器的更多信息请参考特殊功能寄存器章节的说明。

堆栈

堆栈是一个特殊的存储器空间，用来保存程序计数器中的值。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 来指示的，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，程序计数器中的内容会被压入堆栈；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，且有非屏蔽的中断发生，则只有中断请求标志位会被置位，而中断响应会被禁止，直到堆栈指针发生递减（执行 RET 或 RETI 指令），中断才会被响应。这个特性提供程序设计者简单的方法来预防堆栈溢出。然而即使堆栈已满，CALL 指令仍然可以执行，从而造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这样会造成不可预期的程序分支指令的执行错误。

算术逻辑单元 –ALU

算术逻辑单元是单片机中很重要的部分，执行指令集中的算术和逻辑运算。ALU 连接到单片机的数据总线，在接收相关的指令码后执行需要的算术与逻辑运算，并将结果储存在指定的寄存器，当 ALU 计算或操作时，可能导致进位、借位或其它状态的改变，而相关的状态寄存器会因此更新内容以显示这些改变，ALU 所提供的功能如下：

- 算术运算：ADD、ADDM、ADC、ADCM、SUB、SUBM、SBC、SBCM、DAA
- 逻辑运算：AND、OR、XOR、ANDM、ORM、XORM、CPL、CPLA
- 移位运算：RRA、RR、RRCA、RRC、RLA、RL、RLCA、RLC
- 递增和递减：INCA、INC、DECA、DEC
- 分支判断：JMP、SZ、SZA、SNZ、SIZ、SDZ、SIZA、SDZA、CALL、RET、RETI

程序存储器

程序存储器用来存放用户代码即存储程序。此系列的单片机提供一次可编程的存储器（OTP），使用者可以编写他们的应用代码到芯片中。OTP 型单片机提供使用者以灵活的方式自由开发他们的应用，这对于需要除错或者需要经常升级和改变程序的产品是很有帮助的。

结构

程序存储器的容量有 2K×15 到 8K×16 等类型。程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口，数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。

特殊向量

程序存储器中某些地址保留用作诸如复位和中断的入口等特殊用途。

- 复位向量

该向量是保留用做单片机复位后的程序起始地址。在芯片初始化后，程序将会跳转到这个地址并开始执行。

- 外部中断向量

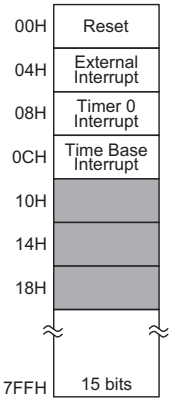
该向量为外部中断服务程序使用。当外部中断引脚发生边沿跳变时，如果中断允许且堆栈未满，则程序会跳转到该地址开始执行。外部中断有效边沿转换类型由 CTRL1 寄存器指定设定是高到低，还是低到高有效或者两者都可以触发。

- 定时/计数器 0 中断向量

该内部中断向量为定时/计数器使用，当定时/计数器发生溢出，如果中断允许且堆栈未满，则程序会跳转到相应的地址开始执行。

- 时基中断向量

该内部向量为时基中断使用，当时基溢出发生，如果中断允许且堆栈未满，则程序将跳转到相应地址开始执行。



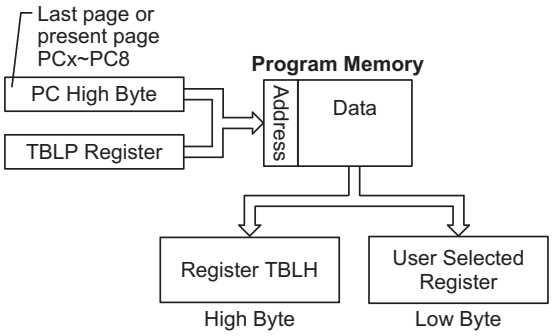
程序存储器结构

查表

程序存储器中的任何地址都可以定义为一个表格，以便存储固定的数据。使用查表指令时，查表指针需要先行设定，其方式是将表格的低位地址放在表格指针寄存器 TBLP 中。这个寄存器定义的是表格较低的 8 位地址。

在设定完表格指针后，表格数据可以使用“TABRDC[m]”或“TABRDL[m]”指令从程序所在的存储器的当前页或者存储器的最后一页中查表来读取。当这些指令执行时，程序存储器的表格的低字节，将会传输到用户所指定的数据存储器中。程序存储器表格的高字节，将会传输到特殊寄存器 TBLH 中。传输数据中任何未定义的字节将会读取为“0”。

下图为查表寻址/数据流程图：



指令	表格地址											
	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TABRDC[m]	PC11	PC10	PC9	PC8	@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

注意： 1. PC11 ~ PC8：当前程序计数器位
2. @7~@0：表格指针 TBLP 位

查表范例

以下范例说明在芯片中表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器的最后一页，在此 ORG 伪指令中的值为 700H，即 1K 程序存储器（HT48R065V）最后一页存储器的起始地址，而表格指针的初始值则为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 706H，即最后一页起始地址后的第六个地址。注意，假如“TABRDC [m]”指令被使用，则表格指针指向当前页。在这个例子中，表格数据的高字节等于零，而当“TABRDL [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

因为 TBLH 寄存器是只读寄存器，不能重新储存，若主程序和中断服务程序都使用表格读取指令，应该注意它的保护。使用表格读取指令，中断服务程序可能会改变 TBLH 的值，若随后在主程序中再次使用这个值，则会发生错误。因此建议避免同时使用表格读取指令。然而在某些情况下，如果同时使用表格读取指令是不可避免的，则在执行任何主程序的表格读取指令前，中断应该先除能，另外要注意，所有与表格相关的指令，都需要两个指令周期去完成操作。

表格读取程序范例

```
tempreg1 db ?                   ; temporary register #1
tempreg2 db ?                   ; temporary register #2
        :
        :
mov a,06h                       ; initialise table pointer - note that this address
                                ; is referenced
mov tblp, a                     ; to the last page or present page
        :
        :
tabrdl    tempreg1              ; transfers value in table referenced by table pointer
                                ; to tempreg1
                                ; data at prog.memory address “706H” transferred to
                                ; to tempreg1 and TBLH
dec   tblp                      ; reduce value of table pointer by one
tabrdl    tempreg2              ; transfers value in table referenced by table pointer
                                ; to tempreg2
                                ; data at prog.memory address “705H” transferred to
                                ; tempreg2 and TBLH
                                ; in this example the data “1AH” is transferred to
                                ; tempreg1 and data “0FH” to register tempreg2
                                ; the value “00H” will be transferred to the high byte
                                ; register TBLH
        :
        :
org 700h                        ; sets initial address of last page

dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
        :
        :
```

数据存储器

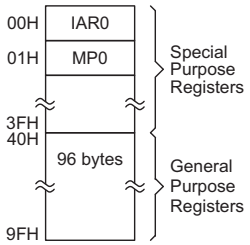
数据存储器是内容可以更改的 8 位 RAM 内部存储器，用来存储临时数据。

结构

数据存储器分为两个部分，第一部分是特殊功能寄存器，这些寄存器有特定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，而有些是被加以保护而不对用户开放。第二部分是通用数据存储器，所有地址都可在程序的控制下进行读取和写入。

数据存储器的两个部分，即特殊和通用数据存储器，位于连续的地址。全部 RAM 为 8 位宽度，而数据储存器长度因所选择的单片机型号的不同而不同。所有单片机的数据存储器的开始地址都是 00H。

所有单片机的程序需要一个读/写的存储区，让临时数据可以被储存和再使用。该 RAM 区域就是通用数据存储器。这个数据存储区可让用户进行读取和写入操作。使用“SET[m].i”和“CLR[m].i”指令可对个别位进行设置或复位的操作，方便用户在数据存储器中进行位操作。

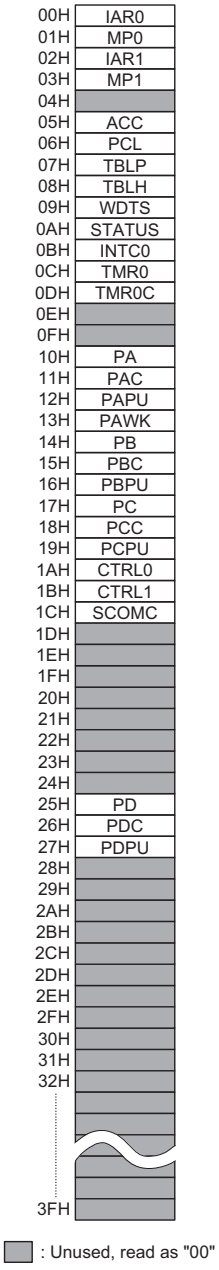


数据存储器结构

注意：使用“SET[m].i”和“CLR[m].i”可对大多数的数据存储区进行位操作，少数专用位除外。数据存储区也可以通过存储器指针间接寻址。

特殊数据存储器

这个区域的数据存储器是存放特殊寄存器的，它和单片机的正确操作密切相关。大多数寄存器是可以读取和写入，只有一些是被写保护而只可读取的，相关的介绍请参考特殊功能寄存器的部分。需注意，任何读取指令对于未定义的地址读取将返回“00H”的值。


00H IAR0
01H MP0
02H IAR1
03H MP1
04H
05H ACC
06H PCL
07H TBLP
08H TBLH
09H WDTS
0AH STATUS
0BH INTC0
0CH TMR0
0DH TMR0C
0EH
0FH
10H PA
11H PAC
12H PAPU
13H PAWK
14H PB
15H PBC
16H PBPU
17H PC
18H PCC
19H PCPU
1AH CTRL0
1BH CTRL1
1CH SCOMC
1DH
1EH
1FH
20H
21H
22H
23H
24H
25H PD
26H PDC
27H PDPU
28H
29H
2AH
2BH
2CH
2DH
2EH
2FH
30H
31H
32H
3FH
: Unused, read as "00"

特殊数据存储器

特殊功能寄存器

为了确保单片机能正常工作，数据存储器中设置了一些内部寄存器。这些寄存器确保内部功能（定时器，中断等）和外部功能（输入/输出口数据控制）的正确工作。特殊功能寄存器和通用数据存储器起始地址之间，有一些未定义的数据存储器，被保留用来做未来扩充，若从这些地址读取数据将会返回 00H 值。

间接寻址寄存器 — IAR0，IAR1

间接寻址寄存器 IAR0 和 IAR1，位于数据存储区，并没有实际的物理地址。间接寻址方式是使用间接寻址寄存器或者存储器指针对数据操作，以取代定义在实际存储器地址的直接存储器寻址方式。在间接寻址寄存器上的任何动作，将对间接寻址指针（MP0 或 MP1）所指定的存储器地址产生对应的读/写操作。IAR0 和 MP0，IAR1 和 MP1 对数据存储器中数据的操作是成对出现的。间接寻址寄存器不是实际存在的，直接读取 IAR 寄存器将会返回 00H 的结果，而直接写入此寄存器则不做任何操作。

间接寻址指针 — MP0， MP1

该系列单片机提供两个间接寻址指针，即 MP0 和 MP1。由于这些指针在数据存储器中能像普通的寄存器一样被写入和操作，因此提供了一个有效的间接寻址和数据追踪的方法。当对间接寻址寄存器进行任何操作时，单片机所指向的实际地址是由间接寻址指针所指定的地址。以下范例说明如何清除一个具有 4 个 RAM 地址的区块，它们已经事先被定义成地址 adres1 到 adres4。

间接寻址程序范例

```
data . section    'data'
adres1    db    ?
adres2    db    ?
adres3    db    ?
adres4    db    ?
block     db    ?
code. section   at 0 code
org 00h

start:
    mov a,04h                  ;setup size of block
    mov block,a
    mov a,offset adres1        ; Accumulator loaded with first RAM address
    mov mp0,a                  ; setup memory pointer with first RAM address

loop:
    clr IAR0                   ; clear the data at address defined by MP0
    inc mp0                    ; increment memory pointer
    sdz block                  ; check if last memory location has been cleared
    jmp loop
continue:
```

在以上的例子中，没有提及具体的数据存储器地址。

累加器 — ACC

对于任何单片机来说，累加器是相当重要的，且与 ALU 所完成的运算有密切关系，所有的 ALU 得到的运算结果都将暂存在累加器中，如果没有累加器，ALU 必须在每次进行如加法、减法和移位等运算时，将结果写入数据存储器中，这样会造成程序编写和时间的负担。另外，数据传输通常涉

及到累加器的临时储存功能，如在用户定义的寄存器和另一个寄存器之间，由于两者之间的不能直接传送数据，因此需要通过累加器来传送数据。

程序计数器低字节寄存器 — PCL

为了提供额外的程序控制功能，程序计数器的低字节被设置在数据存储器的特殊功能区域，程序员可对此寄存器进行操作，很容易直接跳转到其它程序地址。直接给 PCL 寄存器赋值将导致程序直接跳转到专用程序存储器某一地址，然而，由于寄存器只有 8 位的长度，因此只允许在本页的程序存储器中跳转。注意，使用这种运算时，会插入一个空指令周期。

状态寄存器 — STATUS

这 8 位寄存器包括零标志位（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）、和看门狗溢出标志位（TO）。这些标志位同时记录单片机的状态数据和算术/逻辑运算。

除了 TO 和 PDF 标志位以外，状态寄存器的其它位像其它大多数寄存器一样可以被改变。但是任何数据写入状态寄存器将不会改变 TO 和 PDF 标志位。另外，执行不同指令操作后，与状态寄存器相关的运算将会得到不同的结果。TO 标志位只会受系统上电、看门狗溢出、或执行“CLR WDT”或者“HALT”指令的影响。PDF 指令只会受执行“HALT”或“CLR WDT”指令或系统上电的影响。

Z、OV、AC 和 C 标志位通常反映最近的运算操作的状态

另外，当进入一个中断程序或者执行子程序调用时状态寄存器将不会自动压入到堆栈中保存。假如状态寄存器的内容很重要，且中断子程序会改变状态寄存器的内容，则需要保存备份以备恢复。注意，状态寄存器的 0~3 位可以读取和写入。

- **状态寄存器**

位	7	6	5	4	3	2	1	0
名称	—	—	TO	PDF	OV	Z	AC	C
R/W	—	—	R	R	R/W	R/W	R/W	R/W
POR	—	—	0	0	x	x	x	x

“x”表示未知

Bit 7,6 未定义,读为“0”

Bit 5 **TO**：看门狗溢出标志位
0：系统上电或者执行“CLR WDT”或“HALT”指令
1：WDT 溢出

Bit 4 **PDF**：暂停标志位
0：系统上电或执行“CLR WDT”指令
1：执行“HALT”指令将会置位 PDF 位。

Bit 3 **OV**：溢出标志位
0：不发生溢出时
1：当运算结果高两位的进位状态异或结果为 1 时

Bit 2 **Z**：零标志位
0：算数运算或者逻辑运算的结果不为零时
1：算数运算或者逻辑运算的结果为零时

Bit 1 **AC**：辅助进位标志位
0：没有辅助进位时
1：当低字节的加法造成进位或者高字节的减法没有造成借位时

Bit 0 **C**：进位标志位
0：没有进位时
1：当加法造成进位或者减法没有造成借位时，同时移位指令也会影响 C 标志位。

输入/输出口和控制寄存器

在特殊功能寄存器中，输入/输出寄存器（PA、PB 等）和相关的控制寄存器（PAC、PBC 等）

是很重要的，这些寄存器在数据存储器有特定的地址。输入/输出口寄存器用来传送端口上的输入和输出数据，而这些控制寄存器是用来设置引脚的状态，以决定是输出口还是输入口。若要设定一个引脚为输入，需将控制寄存器相应的位设置为高，若引脚设定为输出，则控制寄存器相应的位设置为低。程序初始化期间，在从输入/输出口读取数据之前，需要先设置好控制寄存器以确定引脚的输入输出状态。使用 SET[m].i 和 CLR[m].i 可以对寄存器单独的位进行灵活设置。在程序中可以直接通过改变输入/输出状态控制寄存器中的某一位来改变端口输入/输出状态，这是此系列单片机十分有用的特性。

系统控制寄存器 – CTRL0，CTRL1，CTRL2

这些寄存器是用来控制各种内部功能，如 PFD 控制、PWM 控制、系统时钟选择、LXT 低功耗控制，外部中断边沿触发类型、看门狗定时器使能、时基分频和 LXT 振荡器使能控制。

- **CTRL0 寄存器**

位	7	6	5	4	3	2	1	0
名称	—	—	—	—	PFDEN1	PFDEN0	LXTLP	CLKMOD
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4　未定义,读为 ”0”

Bit 3，2　**PFDEN1,PFDEN0**:PFD/$\overline{\text{PFD}}$使能／除能
00：都除能
01：保留位
10：PFD使能
11：PFD和$\overline{\text{PFD}}$都使能
当PFD/$\overline{\text{PFD}}$两者都除能时，可作普通引脚使用。

Bit 1　**LXTLP**:LXT振荡器低功耗控制位
0：LXT振荡器快速启动模式
1：LXT振荡器低功耗模式

Bit 0　**CLKMOD**:系统时钟模式选择位
0：高速模式-使用HIRC作为系统时钟
1：低速模式-使用LXT作为系统时钟，HIRC振荡器停止。
只有配置选项选择HIRC+LXT为振荡器时，Bit0、Bit1的设定才是有效的。

- **CTRL1 寄存器**

位	7	6	5	4	3	2	1	0
名称	INTEG1	INTEG0	TBSEL1	TBSEL0	WDTEN3	WDTEN2	WDTEN1	WDTEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	0	1	0	1	0

Bit 7, 6 **INTEG1**, **INTEG0**: 外部中断边沿触发类型
00: 关闭中断
01: 上升沿触发
10: 下降沿触发
11: 双边沿触发

Bit 5, 4 **TBSEL1**, **TBSEL0**: 时基周期选择位
00: $2^{10}\times(1/f_{TP})$
01: $2^{11}\times(1/f_{TP})$
10: $2^{12}\times(1/f_{TP})$
11: $2^{13}\times(1/f_{TP})$

Bit 3~0 **WDTEN3**, **WDTEN2**, **WDTEN1**, **WDTEN0**: WDT 功能使能
1010: WDT 关闭
其它值: WDT 使能 – 建议值为 0101
如果“watchdog timer enable”配置选项被选中，则看门狗定时器将总是开启的，且 WDTEN3~WDTEN0 控制位无效

注意： 只有当 WDT 配置选项是除能的，并且位 WDTEN3~WDTEN0 = 1010 时，WDT 才能关闭。
而只要 WDT 配置选项是使能的，或者位 WDTEN3~WDTEN0≠1010，WDT 就开启。

唤醒功能寄存器 – PAWK

当单片机进入休眠模式以后，多种方式可唤醒单片机，使其继续正常运行。其中一种方式是通过有唤醒功能的 I/O 口的下降沿唤醒，而这个控制寄存器就是用来设置 PA 口是否具有唤醒功能。

上拉电阻寄存器 - PAPU, PBPU, PCPU, PDPU

当 I/O 口设置为输入，则可以设置使用内部连接上拉电阻，从而省去外接上拉电阻。这些寄存器即用来选择 I/O 引脚是否连接到内部上拉电阻。

软件控制的 COM 口寄存器 – SCOMC

PB 口的 PB0~PB3 引脚可以用作 COM 口来驱动外部液晶面板，SCOMC 寄存器即用来设置这些引脚正确的偏压值以实现 LCD 驱动功能。

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗之间可以达到最优化。振荡器选项是通过配置选项和寄存器来完成的。

系统振荡器概述

振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。外部振荡器需要一些外围器件，而集成的两个内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。

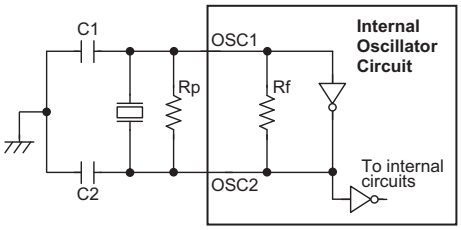
振荡类型	名称	频率范围	引脚
外部晶振	HXT	400kHz~12MHz	OSC1/OSC2
外部 RC	ERC	400kHz~12MHz	OSC1
内部高速 RC	HIRC	4，8 或 12MHz	—
外部低速晶振	LXT	32768Hz	OSC1/OSC2
内部低速 RC	LIRC	13kHz	—

系统时钟配置

此系列的单片机有五个系统振荡器，包括三个高速振荡器和两个低速振荡器。高速振荡器有外部晶体/陶瓷振荡器-HXT，外部 RC 振荡器-ERC 和内部 RC 振荡器-HIRC。两个低速振荡器包括外部 32768Hz 振荡器-LXT 和内部 13kHz（VDD=5V）振荡器-LIRC。

外部晶体/陶瓷振荡器–HXT

对于晶体振荡器，只要简单地将晶体连接至 OSC1 和 OSC2，则会产生振荡所需的相移及反馈，而不需其它外部的器件。为保证某些低频率的晶体振荡和陶瓷谐振器的振荡频率更精准，建议连接两个小容量电容 C1 和 C2 到 VSS，具体数值与客户选择的晶体/陶瓷晶振有关。



Note: 1. Rp is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

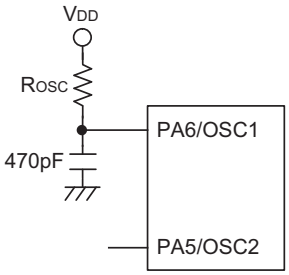
晶体/陶瓷振荡器 – HXT

晶体振荡器 C1 和 C2 值		
晶体频率	**C1**	**C2**
12MHz	8pF	10pF
8MHz	8pF	10pF
4MHz	8pF	10pF
1MHz	100pF	100pF
注意：C1 和 C2 数值仅作参考用		

晶体振荡器电容推荐值

外部 RC 振荡器 – ERC

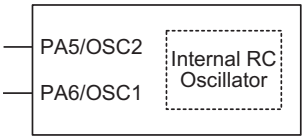
使用外部 RC 电路可配置低成本的系统振荡器，只需要在 OSC1 和 VDD 之间连接一个阻值约在 24 kΩ 到 1.5MΩ 之间的电阻，OSC1 与 VSS 之间连接一个电容。系统频率由外部所接电阻的大小决定，外部电容并不会影响振荡器的频率值，在这里只起到稳定的作用。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 VDD、温度以及芯片制成工艺不同的影响减至最低程度。这里，提供一个电阻/频率的参考：使用外部 120K 电阻连接到 5V 电源电压，在 25℃下，振荡器的频率为 4MHz，容差 2%。外部 RC 振荡器仅使用 OSC1 引脚，OSC1 与 PA6 引脚共用，此时 PA5 引脚可以作为普通的 I/O 口使用。



外部 RC 振荡器 – ERC

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有三种固定的频率：4MHz，8MHz，12MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 VDD、温度以及芯片制成工艺不同的影响减至最低程度。在电源电压为 3V 或者 5V 及温度为 25℃的条件下，4MHz，8MHz，12MHz 这三个固定频率的容差为 2%。如果选择了该内部时钟，无需额外的引脚；PA5 和 PA6 可以作为通用 I/O 口使用。

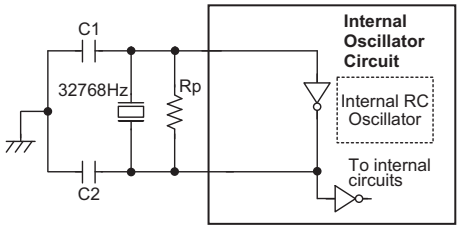


Note: PA5/PA6 used as normal I/Os

内部 RC 振荡器 – HIRC

外部 32768Hz 晶体振荡器–LXT

当系统进入休眠模式，系统时钟关闭以降低功耗。然而在某些应用需要在省电模式下保持定时/计数功能能继续运行，系统需要提供额外的系统时钟。配置选项提供了高速时钟和低速时钟搭配使用的选项，LXT 振荡器即用来提供这样的低速时钟。LXT 振荡器由 32768Hz 晶振接到 OSC1 和 OSC2 引脚。为了保证 LXT 起振和频率的精确，建议使用两个小容量电容 C1 和 C2，具体数值与客户选择的 32768Hz 晶振有关。另外，外部并联的反馈电阻 R_P 也是必需的。LXT 振荡器需要和 HIRC 振荡器搭配使用。



Note: 1. Rp, C1 and C2 are required.
2. Although not shown pins have a parasitic capacitance of around 7pF.

外部 LXT 振荡器

LXT 振荡器 C1 和 C2 的值		
晶体振荡器频率	**C1**	**C2**
32768Hz	8pF	10pF
注意：1. C1 和 C2 的值仅供参考 2. 推荐 R_P=5M~10MΩ		

32768Hz 晶体振荡器电容推荐值

LXT 振荡器低功耗功能

LXT 振荡器有两种模式：快速启动模式和低功耗模式。可以使用寄存器 CTRL0 中的 LXTLP 位来选择模式。

LXTLP 位	LXT 时钟模式
0	快速启动
1	低功耗

系统上电后，LXTLP 会自动清零以确保 LXT 振荡器工作在快速启动模式。在快速启动模式中，LXT 振荡器将起振并且快速的稳定下来。在 LXT 振荡器完全起振后，可设置 LXTLP 位为 1，LXT 振荡器将进入低功耗模式。振荡器将继续运行，但功耗降低，振荡电路只有在 LXT 起振时需要较大的电流。在电池及低功耗应用中，建议在上电两秒后再设置 LXTLP 位为 1，以保证最小的功耗。

注意，无论 LXTLP 位设置为何值，LXT 振荡器都能保持正常工作，不同之处在于，低功耗模式下启动需要更长的时间。

内部低速振荡器 – LIRC

LIRC 是一个完全独立运行的片内 RC 振荡器，无需外部器件，在常温 5V 条件下，振荡频率值为 13kHz。当单片机进入休眠模式，系统时钟将停止运行。但 WDT 振荡器会继续运行以保持 WDT 的功能。然而，在某些需要节省功耗的应用中，可通过配置选项来关闭 LIRC 以降低功耗。

工作模式

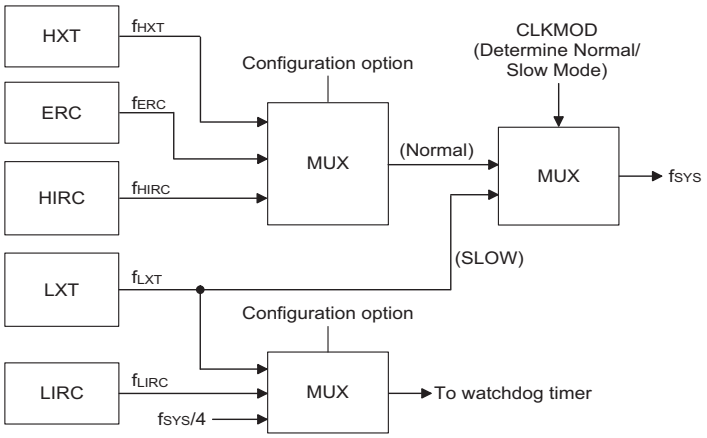
使用 LXT 低速振荡器和一个高速振荡器，系统可以工作在下面几个不同的模式：正常模式、低速模式和休眠模式。

模式的类型与选择

使用较高频率的振荡器将获得较高的性能，但是功耗也较大，使用较低频率的振荡器则刚好相反。若单片机能够动态的切换高速和低速振荡器，则单片机就具有足够的灵活性来优化性能/功耗比，该特性对于低功耗的应用特别重要。

单片机如果使用了 LXT 振荡器，则需要使用内部 RC 振荡器 HIRC 作为高速振荡器。如果使用 HXT 或者 ERC 作为高速系统时钟，共用该引脚被占用，因此不能外接 LXT 振荡器。寄存器 CTRL0 中的 CLKMOD 位用来切换系统时钟从高速 HIRC 振荡器到低速 LXT 振荡器。当执行 HALT 指令后，单片机进入休眠模式，LXT 振荡器将继续运行。LXT 晶振连接到 OSC1/OSC2 脚，且 LXT 将一直运行。

注意，只有在 HIRC+LXT 振荡器配置下，对 CLKMOD 的操作才是有效的。



系统时钟配置

工作模式	OSC1/OSC2 配置				
	HXT	ERC	HIRC	HIRC+LXT	
				HIRC	LXT
正常模式	运行	运行	运行	运行	运行
低速模式	—	—	—	停止	运行
休眠模式	停止	停止	停止	停止	运行

注意：“-”表示未使用

工作模式控制

切换模式

通过设置 CTRL0 寄存器中的 CLKMOD 位和 HALT 指令，可实现单片机工作模式之间的切换。CLKMOD 位用来设置系统时钟是高速或者低速振荡器，从而使系统工作在正常模式或者低速模式。执行 HALT 指令将强制系统进入休眠模式（LXT 停止）。HALT 指令的执行与 CLKMOD 位的设置无关。

当执行 HALT 指令，LXT 振荡器停止运行，系统进入休眠模式，将发生下面的情况：

- 系统振荡器将被关闭，应用程序将停止在“HALT”指令处
- 在 RAM 和寄存器上的内容保持不变
- 如果 WDT 时钟源是来自 LIRC 振荡器或者 LXT 振荡器，则 WDT 将被清除然后再重新计数；若来源于系统时钟，则停止计数
- 所有输入/输出端口状态保持不变
- STATUS 寄存器中，PDF 标志位被置位，TO 标志位被清零

静态电流控制

要使系统静态电流降到最小，为微安级，除了需要单片机进入休眠模式，还要考虑到电路的设计。特别要注意输入/输出口的状态。所有高阻抗输入引脚需要接高电平或低电平，否则引脚浮空会造成内部振荡进而增大电流的消耗。另外还需要注意单片机输出端口上的负载，尽量减少拉电流或与其它 CMOS 输入相连。

如果配置选项使能看门狗振荡器 LIRC，当进入休眠模式，振荡器继续振荡，并继续消耗能量。对电源消耗敏感的应用，使用系统时钟作为 WDT 时钟是更好的选择。如果配置 LXT 使能，当进入休眠模式时也将消耗一定的能量。置位 LXTLP（CTRL0.1）也可使 LXT 振荡消耗最小。

唤醒

当系统进入休眠模式下，可以通过以下几种方式唤醒：

- 外部复位
- PA 口下降沿
- 系统中断
- WDT 溢出

若由外部$\overline{RES}$引脚唤醒，系统会经过完全复位的过程。若由 WDT 溢出唤醒，则看门狗计数器将被复位清零。这两种唤醒方式都会使系统复位，可以通过状态寄存器中 TO 和 PDF 位来判断它的唤醒源。系统上电或执行清除看门狗的指令，PDF 被清零；执行 HALT 指令，PDF 将被置位。看门狗计数器溢出将会置位 TO 标志并唤醒系统，同时复位程序计数器和堆栈指针，其它标志保持原有状态。

端口 PA0~PA7 中的每个位都可以通过 PAWK 寄存器独立选择唤醒功能。PA 口唤醒后，程序将执行“HALT”指令后的其他指令。

如果系统是通过中断唤醒，则有两种情况，假如中断除能或中断使能但堆栈已满，系统唤醒后继续执行“HALT”指令的其他指令，相应的中断服务程序只有在中断使能后或堆栈空闲后被执行；假如中断使能且堆栈未满，则正常的中断响应将会发生。如果系统进入休眠模式之前外部中断请求标志位被置为“1”，则相关中断的唤醒功能无效。

无论是哪种方式唤醒，单片机从唤醒回到正常运行都需要一定的延迟时间，延时的时长请参照下面的表格：

唤醒源	振荡器类型	
	ERC，IRC	Crystal
外部$\overline{RES}$	$t_{RSTD}+t_{SST1}$	$t_{RSTD}+t_{SST2}$
PA 口	t_{SST1}	t_{SST2}
中断		
WDT 溢出		

注：1.t_{SYS}（系统时钟）
2. t_{RSTD} 为上电延时，典型值为 100ms
3. t_{SST1}=2 或者 1024 t_{SYS}
4. t_{SST2}= 1024 t_{SYS}

唤醒延迟时间

看门狗定时器

看门狗定时器的功能在于防止如电磁的干扰等外部不可控制事件，所造成的程序不正常动作或跳转到未知的地址。

看门狗的使用

当 WDT 溢出时，会产生系统复位的动作。当 WDT 配置选项设置为除能，则任何相关的指令操作都无效。通过配置选项和两个内部的寄存器 WDTS 和 CTRL1 可以设置不同的 WDT 选项。通过配置选项和数据存储器中特殊功能寄存器 CTRL1 的 WDTEN 位，来使能看门狗定时器。

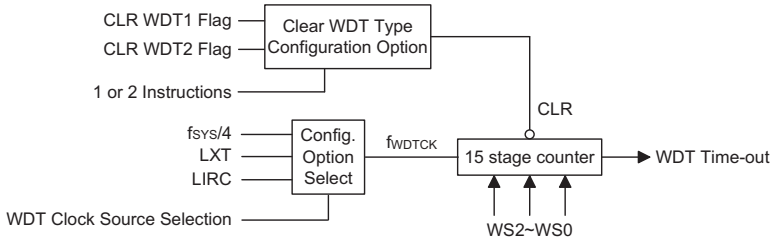
配置选项	CTRL1 寄存器	WDT 功能
除能	除能	OFF
除能	使能	ON
使能	×	ON

看门狗定时器开/关控制

如果 WDT 配置选项除能且 CTRL1 寄存器中 WDTEN3~WDTEN0 位被写入 1010B，看门狗定时器将被关闭。此时 WDTEN 中的值为系统上电时的默认值。虽然向 CTRL1 的 WDTEN3~WDTEN0 位写入其它的任何数字可开启看门狗定时器，但为了最大程度保护它，建议向这些位写入 0101B。

通过配置选项，看门狗定时器可以选择三种不同的时钟源：LXT，$f_{SYS}/4$ 或 LIRC。注意，选择 $f_{SYS}/4$ 时钟作为 WDT 的时钟源，当系统进入休眠模式时，指令时钟会停止且 WDT 将失去其保护功能。对于干扰比较大的应用环境，推荐使用 LIRC 振荡器或者 LXT 作为 WDT 的时钟源。分频比由 WDTS 寄存器的第 0，1 和 2 位，即 WS0、WS1 和 WS2 位来决定。如果 WS0、WS1 和 WS2 都置 1，分频比例为 1:128，即可提供最大溢出周期。

系统在正常运行状态下，WDT 溢出将导致芯片复位，并置位状态标志位 TO。但是在系统处于休眠模式时， 如果 WDT 发生溢出，系统将从休眠中唤醒，置位状态寄存器中的 TO，并且它只复位程序计数器 PC 和 SP。有三种方法可以用来清除 WDT 的内容，第一种是外部硬件复位(RES 引脚低电平)，第二种是通过软件指令，而第三种是通过“HALT”指令。使用软件指令有两种方法去清除看门狗寄存器，需要由配置选项选择。第一种选择是使用单一“CLR WDT”指令，而第二种是使用“CLR WDT1”和“CLR WDT2”两个指令。对于第一种选择，只要执行“CLR WDT”便清除 WDT。而第二种选择，需要交替执行“CLR WDT1”和“CLR WDT2”两者才能成功的清除 WDT。关于第二种选择，如果“CLR WDT1”正被使用来清除 WDT，接着再执行这条指令将是无效的，只有执行“CLR WDT2”指令才能清除 WDT。同样的“CLR WDT2”指令已经执行后，只有接着执行“CLR WDT1”指令才可以清除看门狗定时器。



看门狗定时器

- **WDTS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	WS2	WS1	WS0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	1	1	1

Bit 7~3： 未定义， 读为“0”

Bit 2~0： **WS2**，**WS1**，**WS0**：WDT 溢出周期选择

000：$2^8 t_{WDTCK}$

001：$2^9 t_{WDTCK}$

010：$2^{10} t_{WDTCK}$

011：$2^{11} t_{WDTCK}$

100：$2^{12} t_{WDTCK}$

101：$2^{13} t_{WDTCK}$

110：$2^{14} t_{WDTCK}$

111：$2^{15} t_{WDTCK}$

复位和初始化

复位功能是任何单片机中基本的部分，使得单片机可以设定一些与外部参数无关的先置条件。最重要的复位条件是在单片机首次上电以后，经过短暂的延迟，内部硬件电路使得单片机处于预期的稳定状态并开始执行第一条程序指令。上电复位以后，在程序执行之前，部分重要的内部寄存器将会被设定为预先设定的状态。程序计数器就是其中之一，它会被清除为零，使得单片机从最低的程序存储器地址开始执行程序。

除上电复位以外，即使单片机处于正常工作状态，有些情况的发生也会迫使单片机复位。譬如当单片机上电后已经开始执行程序，$\overline{\text{RES}}$脚被强制拉为低电平。这种复位为正常操作复位，单片机中只有一些寄存器受影响，而大部分寄存器不会改变，在复位引脚恢复至高电平后，单片机可以正常运行。另一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。

另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位，这种复位与与$\overline{\text{RES}}$脚拉低复位方式相同。

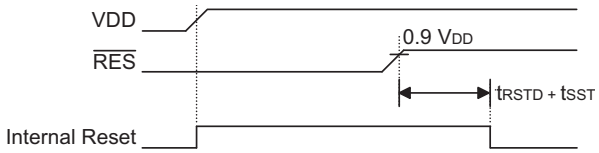
复位功能

包括内部和外部事件触发复位，单片机共有五种复位方式：

- 上电复位

这是最基本且不可避免的复位，发生在单片机上电后。除了保证程序存储器从开始地址执行，上电复位也使得其它寄存器被设定在预设条件，所有的输入/输出端口控制寄存器在上电复位时会保持高电平，以确保上电后所有引脚被设定为输入状态。

虽然单片机有一个内部 RC 复位功能，如果电源上升缓慢或者上电时电源不稳定，内部 RC 振荡可能导致芯片复位不良，所以推荐使用和$\overline{\text{RES}}$引脚连接的外部 RC 电路。由 RC 电路所造成的时间延迟使得$\overline{\text{RES}}$引脚在电源供应稳定前的一段延长周期内保持在低电平。在这段时间内，单片机的正常操作是被禁止的。$\overline{\text{RES}}$引脚达到一定电压值后，再经过延迟时间 t_{RSTD} 单片机可以开始进行正常操作。下图中 SST 是系统延迟周期 System Start-up Timer 的缩写。

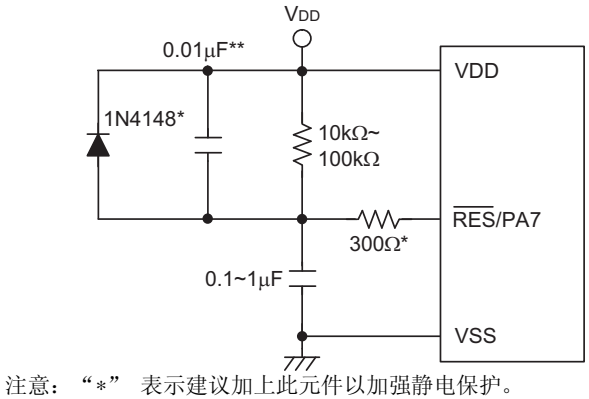


注意：t_{RSTD}为上电延迟时间，典型值为100ms。

上电复位时序图

在许多应用场合，可以在 VDD 和$\overline{RES}$之间接入一个电阻，在 VSS 与$\overline{RES}$之间接入一个电容作为外部复位电路。与$\overline{RES}$脚上所有相连接的线段必须尽量短以减少噪声干扰。

当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。



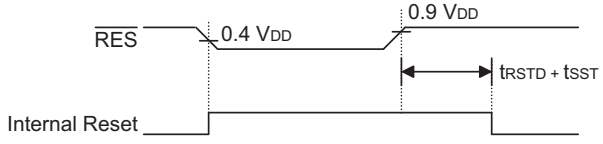
注意：“*” 表示建议加上此元件以加强静电保护。
“**”表示建议在电源有较强干扰场合加上此元件。

外部$\overline{RES}$电路

欲知有关外部复位电路的更多信息可参考 HOLTEK 网站上的应用范例 HA0075S

- $\overline{RES}$引脚复位

当单片机正常工作时，$\overline{RES}$引脚通过外部硬件（如外部开关）强迫拉至低电平时，此种复位形式即会发生。这种复位方式和其它的复位方式一样，程序计数器会被清除为零且程序从头开始执行。

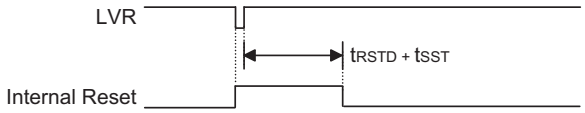


注意：t_{RSTD}为上电延迟时间，典型值为100ms。

$\overline{RES}$复位时序图

• 低电压复位-LVR

单片机具有低电压复位电路，用来监测它的电源电压，可通过配置选项进行选择。例如在更换电池的情况下，单片机供应的电压可能会落在 0.9V~V_{LVR} 的范围内，这时 LVR 将会自动复位单片机。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过交流电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。V_{LVR} 参数值可通过配置选项进行设定。

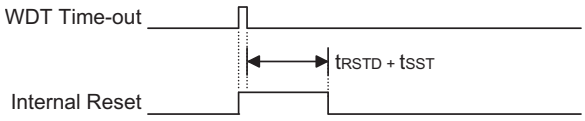


注意：t_{RSTD}为上电延迟时间，典型值为100ms。

低电压复位时序图

• 正常运行时看门狗溢出复位

除了看门狗溢出标志位 TO 将被设为 1 之外，正常运行时看门狗溢出复位和$\overline{RES}$复位相同。

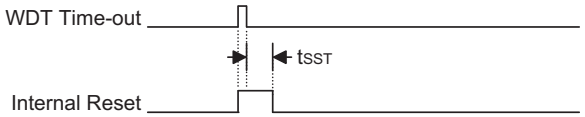


注意：t_{RSTD}为上电延迟时间，典型值为100ms。

正常运行时看门狗溢出时序图

•休眠时看门狗溢出复位

休眠时看门狗溢出复位和其它种类的复位有些不同,除了程序计数器与堆栈指针将被清 0 及 TO 位被设为 1 外，绝大部分的条件保持不变。图中 t_{SST} 的详细说明请参考交流电气特性。



注意：如果系统时钟为ERC或者HIRC时，通过配置选项可以选择t_{SST}为1024或者2个时钟周期。
如果系统时钟为HXT或者LXT，则t_{SST}为1024个时钟周期。

休眠时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位，即 PDF 和 TO 位存放在状态寄存器中，由休眠功能或看门狗计数器等几种控制器操作控制。复位标志位如下所示：

T0	PDF	复位条件
0	0	上电时的$\overline{\text{RES}}$复位
u	u	正常模式或低速模式时的$\overline{\text{RES}}$复位或 LVR 复位
1	u	正常模式或低速模式时的 WDT 溢出复位
1	1	休眠模式时的 WDT 溢出复位

注意：“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器	WDT 清除并重新计时
定时/计数器	所有定时/计数器停止
预分频器	定时/计数器之预分频器内容清除
输入/输出口	所有 I/O 设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

不同的复位形式对单片机内部寄存器的影响是不同的。为保证复位后程序能正常执行，了解寄存器在特定条件复位后的设置是非常重要的，下表即为不同方式复位后内部寄存器的状况。

寄存器	上电复位	$\overline{\text{RES}}$或 LVR 复位	WDT 溢出 (正常模式)	WDT 溢出 (休眠模式)
PCL	0000 0000	0000 0000	0000 0000	0000 0000
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	- xxx xxxx	-uuu uuuu	- uuu uuuu	- uuu uuuu
WDTS	- - - - -111	- - - - -111	- - - - -111	- - - - - uuu
STATUS	- -00 xxxx	- -uu uuuu	- -1u uuuu	- - 11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	- uuu uuuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1000	0000 1000	0000 1000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAWK	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	- 000 0000	- 000 0000	- 000 0000	-uuu uuuu
PB	- - 11 1111	- - 11 1111	- - 11 1111	- -uu uuuu
PBC	- - 11 1111	- - 11 1111	- - 11 1111	- -uu uuuu
PBPU	- -00 0000	- 000 0000	- 000 0000	- -uu uuuu
PC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	- - - - 1111	- - - - 1111	- - - - 1111	- - - - uuuu
PDC	- - - - 1111	- - - - 1111	- - - - 1111	- - - - uuuu
PDPU	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
CTRL0	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
CTRL1	1000 1010	1000 1010	1000 1010	uuuu uuuu
SCOMC	0000 0000	0000 0000	0000 0000	uuuu uuuu

注意：“-”表示未定义
“x”表示未知
“u”表示不改变

输入/输出端口

盛群单片机的输入/输出口控制具有很大的灵活性。每一个引脚都在用户程序控制下都可被设定为输入或输出，所有引脚的上拉电阻设置以及指定引脚的唤醒设置也都由软件控制，这些特性也使得此类单片机在广泛应用上都能符合开发的需求。

作为输入操作，输入引脚无锁存功能，也就是说输入数据必须在执行“MOV A, [m]”,T2 的上升沿准备好，m 为端口地址。对于输出操作，所有数据都是被锁存的，且保持不变直到输出锁存被重写。

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去外部上拉电阻，当引脚规划为输入时，可由内部连接到一个上拉电阻，这些上拉电阻可通过寄存器 PAPU，PBPU、PCPU 和 PDPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。注意，PA7 引脚没有上拉电阻功能。

PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入休眠模式状态，单片机的系统时钟将会停止以降低功耗，此功能对于电池及低功耗应用很重要。唤醒单片机有很多种方法，其中之一就是使 PA0~PA7 的其中一个引脚从高电平转为低电平。使用暂停指令“HALT”迫使单片机进入休眠模式状态后，处理器将会一直保持低功耗状态，直到 PA 口上被选为唤醒输入的引脚电平发生下降沿跳变。这个功能特别适合于通过外部开关来唤醒的应用。注意，PA0~PA7 是可以通过设置 PAWK 寄存器来单独选择是否具有唤醒功能。

PAWK, PAC, PAPU, PBC, PBPU, PCC, PCPU, PDC, PDPU 寄存器

寄存器名称	POR	位							
		7	6	5	4	3	2	1	0
PAWK	00H	PAWK7	PAWK6	PAWK5	PAWK4	PAWK3	PAWK2	PAWK1	PAWK0
PAC	FFH	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	00H	—	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PBC	3FH	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	00H	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PCC	FFH	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	00H	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PDC	0FH	—	—	—	—	PDC3	PDC2	PDC1	PDC0
PDPU	00H	—	—	—	—	PDPU3	PDPU2	PDPU1	PDPU0

“—” 未定义，读为“0”

PAWKn：PA 口唤醒功能使能
0: 除能
1: 使能

PACn/PBCn/PCCn/PDCn：I/O 类型选择
0: 输出
1: 输入

PAPUn/PBPUn/PCPUn/PDPUn：上拉功能使能
0: 除能
1: 使能

输入/输出端口控制寄存器

每一个输入/输出口都具有各自的控制寄存器（PAC,PBC,PCC,PDC）用来控制输入/输出状态。从而每个 I/O 引脚都可以通过软件控制，动态的设置带上拉电阻或者不带上拉电阻。所有的 I/O 端口的引脚都各自对应于 I/0 端口控制的某一位。若 I/O 引脚要实现输入功能，则对应的控制寄存器的位需要设置为“1”，这时程序指令可以直接读取输入脚的逻辑状态。若控制寄存器相应的位被设定为“0”，则此引脚被设置为 COMS 输出。当引脚设置为输出状态时，程序指令读取的是输出端口寄存器的内容。注意，如果对输出口做读取动作时，程序读取到的是内部输出数据锁存器中的状态，而不是输出引脚上实际的逻辑状态。

引脚共用功能

引脚的共用功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。多功能输入/输出的功能选择，有些是由配置选项进行设定，有些则是在应用程序中进行控制。

- 外部中断输入

外部中断引脚 INT 与一个 I/O 引脚共用。为了使用该引脚作为外部中断输入引脚，需要正确设置 INTC0 寄存器中的有关位。此外，还需要通过端口控制寄存器中的 PAC3 位来设置该引脚为输入脚。如果需要，可以通过上拉电阻寄存器来选择带上拉电阻。注意即使该引脚被配置为外部中断输入，引脚的输入/输出功能将依然存在。

- 外部定时/计数器输入

定时/计数器引脚 TC0 与输入/输出引脚共用。如果设定为定时/计数器的输入，则需要通过设置外部定时/计数器控制寄存器相应的位将外部定时/计数器配置为外部事件计数模式或者脉冲宽度测量模式，同时该引脚需要通过端口控制寄存器设置为输入，上拉电阻也可以通过上拉电阻寄存器进行设置。注意，即使该引脚被配置为外部定时/计数器输入，输入/输出功能依然存在。

- PFD 输出

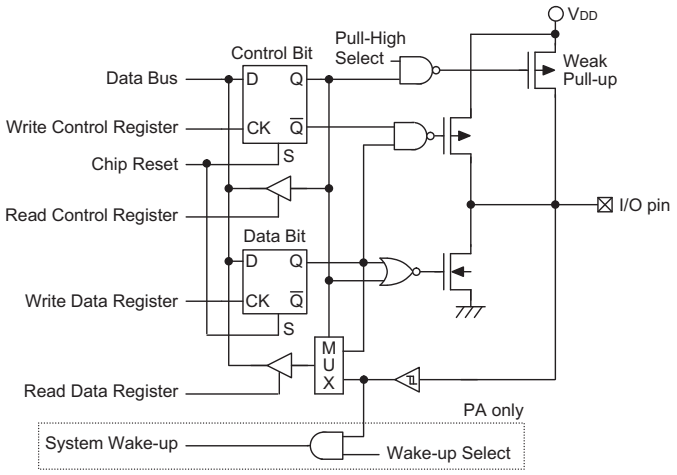
此系列单片机均提供有单通道和双通道的 PFD 信号输出，与输入/输出引脚共用。PFD 的输出可通过 CTRL0 寄存器进行设置。注意端口控制寄存器相应的位需要设置为输出高才能使能 PFD 的输出。如果端口控制寄存器被设置为输入，即使正确设置了 PFD 的输出，该引脚都只作为普通逻辑输入脚，并且允许选择上拉电阻。

- 软件控制的 COM 口驱动引脚

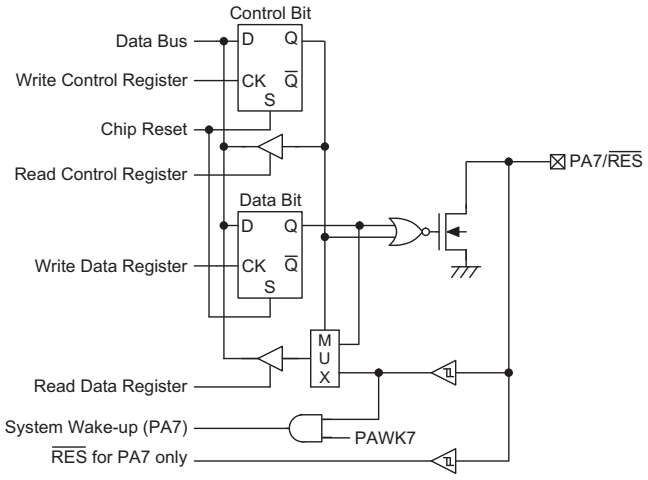
PB 端口中的 PB0~PB3 能被用做 LCD COM 的驱动引脚。该功能由 SCOMC 寄存器来控制。通过设置 SCOMC 寄存器，可以使 PB0~PB3 四个引脚上输出驱动 LCD 所需要的 1/2 bias 偏置电压。

输入/输出引脚结构

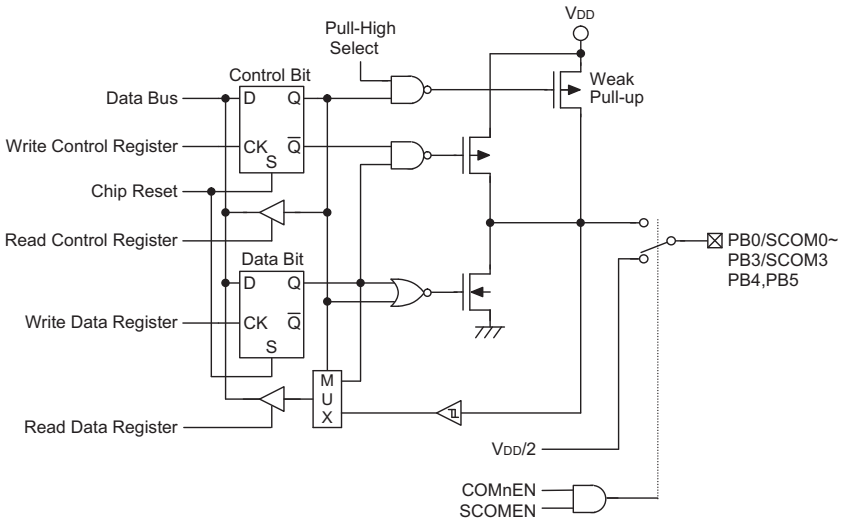
下图为输入/输出引脚的内部结构图。输入/输出引脚的准确逻辑结构图可能与此图不同，这里只是为了方便对功能的理解提供的一个参考。



通用输入/输出端口



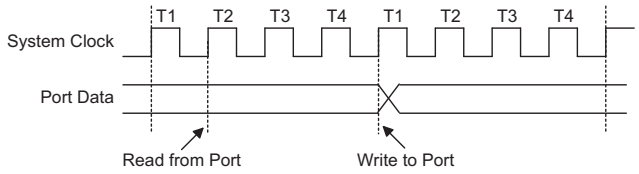
PA7 NMOS 输入/输出端口



PB 输入/输出端口

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入/输出数据及端口控制寄存器都将被设为逻辑高。所有输入/输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或者使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读-修改-写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。



读写时序图

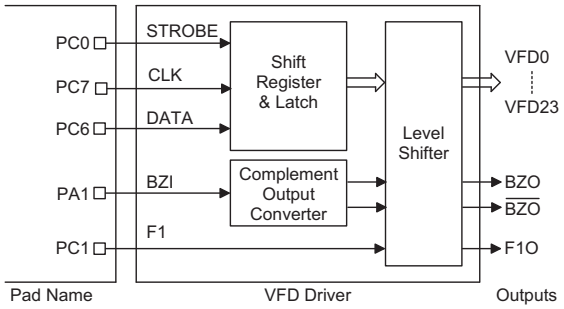
PA0~PA7 的每个引脚可通过 PAWK 寄存器设置带唤醒功能。单片机处于休眠模式时，有很多方法可以唤醒单片机，其中之一就是通过 PA 任一引脚电平从高到低的转换的方式，可以设置 PA 口一个或者多个引脚具有唤醒功能。

VFD 驱动器

该单片机具有 VFD 功能，用来驱动 VFD 面板的高压灯丝和蜂鸣器。单片机使用串行通讯接口将显示数据传送到 VFD 驱动器中的 24 位移位寄存器。VFD 驱动器将移位寄存器中的数据转成 VFD 驱动信号，并且提供必要的电平转换。数据只能从单片机单向传送到 VFD 驱动器，反之则不行。

VFD 接口

下图为单片机与 VFD 驱动器之间的五线接口方框图。



VFD 驱动器

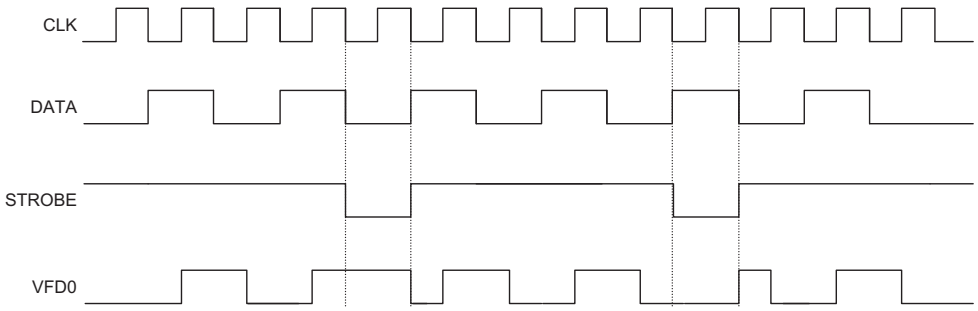
单片机与 VFD 驱动器之间的数据传输由三线接口（CLK,DATA 和 STROBE）来控制。数据为单向传输方式，且单片机的输入/输出口必选设置为输出。

蜂鸣器的控制输入口 BZI 经由 VFD 驱动的转换器转变为一对互补输出 BZO 和$\overline{BZO}$，且此对互补输出也将通过电平转换器转换为高电压输出。VFD 驱动器的灯丝控制输入口 F1 也将转换为高电压输出 F1O，用于灯丝的开/关控制。

24 位移位寄存器/锁存器

单片机与 VFD 驱动器间使用串行数据传输，需首先将数据写入到位于 VFD 驱动的 24 位移位寄存器中。这 24 位中 VFD0~VFD15 口用于控制 VFD 面板的段，VFD16~VFD23 口用于控制 VFD 面板的栅格，控制方式描述如下：

- 通过“DATA”和“CLK”口将数据传送至内部的 24 位移位寄存器，数据在时钟上升沿写入移位寄存器，此数据为 VFD0~VFD23 口要输出的显示数据。只有当 STROBE 口为高时，VFD 输出才发生改变。而当 STROBE 口为低时，只修改移位寄存器中的数据，VFD 输出保持不变。
- 通过“STROBE”口将移位寄存器中的数据锁存到输出口 VFD0~VFD23，当 STROBE 口为高时，移位寄存器数据将会锁存在 VFD 口上。注意，STROBE 口为电平触发而非边沿触发。



VFD 显示控制时序图

下表为 24 位移位寄存器/锁存器功能表：

Clock	Strobe	Data	VFD0	VFDn
↑	0	×	不变	不变
↑	1	0	0	VFDn-1
↑	1	1	1	VFDn-1
↓	1	1	不变	不变

注意：“×”表示不影响。
“VFDn”表示 VFD1 ~VFD23。

编程注意事项

上电之后，所有的输入/输出口将自动设置为输入。然而当 PB2~PB5 用于 VFD 驱动时，单片机上电之后必须将他们设置输出。而 VFD 接口控制口设置为输入将导致错误的 VFD 显示操作。建议上电初始化时，配置选项设置 VFD 控制口带上拉电阻，使得这些引脚保持固定的高电平，直到这些口设置为输出。

编程应用范例

以下例子说明了 VFD 显示数据如何通过单片机编程。

```
;It's necessary to define strobe/clk/data for related I/O port.
Data_2_register:                    ; send data to vfd driver
      mov     a,024d                ; shift register counter
      mov     count,a
      clr     strobe                ; strobe = 0
data_2_register_1:
      clr     clk                   ; clk = 0
      set     data                  ; data = 1
      snz     vfd_grid.7
      clr     data                  ; data = 0
      rlc     vfd_segl              ; shift data to vfd[7:0]
      rlc     vfd_segh              ; shift data to vfd[15:8]
      rlc     vfd_grid              ; shift data to vfd[22:16]
      set     clk                   ; clk = 1 (rising edge)
      sdz     count
      jmp     data_2_register_1
      set     strobe                ; strobe = 1, vfd output
      ret
```

定时/计数器

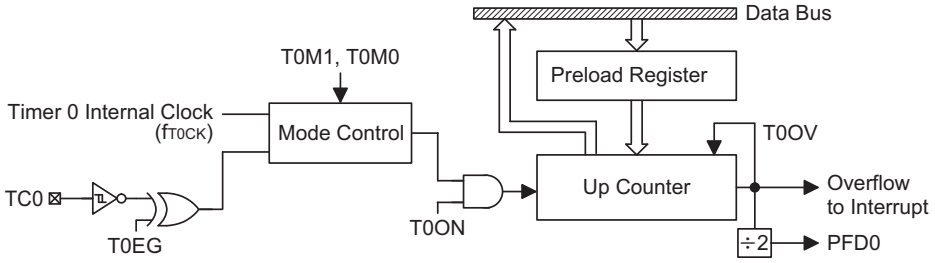
定时/计数器在任何单片机中都是一个很重要的部分，提供程序设计者一种实现和时间有关功能的方法。这些芯片具有一个 8 位的向上计数器。定时/计数器有三种不同的工作模式，可以当作一个普通定时器、外部事件计数器或脉冲宽度测量使用。并且提供了一个内部时钟分频器，以扩大定时器的范围。

有两种和定时/计数器相关的寄存器。第一种类型的寄存器是用来存储实际的计数值，赋值给此寄存器可以设定初始值，读取此寄存器可获得定时/计数器的内容；第二种类型的寄存器为定时器控制寄存器，用来定义定时/计数器工作模式和定时设置。定时/计数器的时钟源可来自内部时钟源或外部定时器引脚。

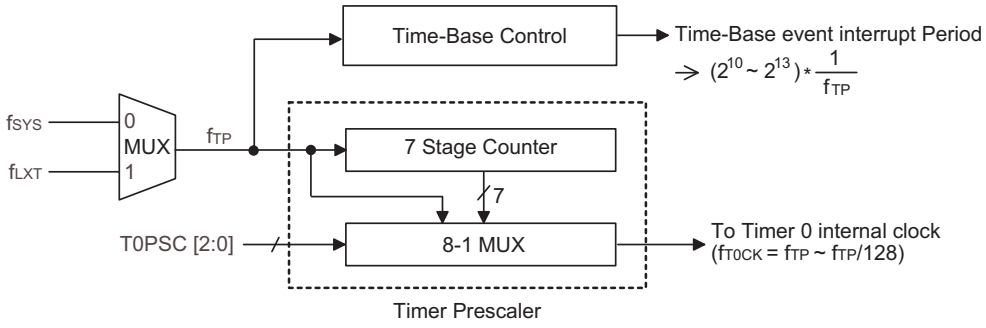
配置定时/计数器输入时钟源

定时/计数器的时钟源可有多种选择，可以是内部时钟，也可以是外部引脚。当定时/计数器工作在定时器模式或脉冲宽度测量模式时，使用内部时钟作为时钟源。对于某些定时/计数器，内部时钟首先由分频器分频，分频比由定时器控制寄存器的位 T0PSC0~ T0PSC2 来确定。对于定时/计数器 0，内部时钟源可以通过 TMR0C 寄存器的 T0S 位来选择 f_{SYS} 或者 LXT 振荡器。

当定时/计数器在事件计数模式时，使用外部时钟源，时钟源由外部时钟输入引脚 TC0 提供。每次外部引脚由高电平到低电平或者由低电平到高电平(由 T0EG 位决定)进行转换时，计数器增加一。



8 位定时/计数器 0 结构图



Timer/Time Base 的时钟源结构图

定时/计数寄存器 - TMR0

定时/计数寄存器 TMR0，是位于特殊数据存储器内的特殊功能寄存器，用于储存定时器的当前值。在用作内部定时且收到一个内部计数脉冲或用作外部计数且外部定时/计数器引脚发生状态跳变时，此寄存器的值将会加一。定时器将从预置寄存器所载入的值开始计数，到 FFH 时定时器溢出且会产生一个内部中断信号。定时器的值随后被预置寄存器的值重新载入并继续计数。

注意，上电后预置寄存器处于未知状态。为了得到定时器的最大计算范围 FFH，预置寄存器需要先清为零。定时/计数器在关闭条件下，写数据到预置寄存器，会立即写入实际的定时器。而如果定时/计数器已经打开且正在计数，在这个周期内写入到预置寄存器的任何新数据将保留在预置寄存器，直到溢出发生时才被写入实际定时器。

定时/计数控制寄存器 - TMR0C

Holtek 单片机灵活的特性也表现在定时器的多功能上，定时/计数器能提供三种不同的工作模式，由相应的控制寄存器来选择定时/计数器的工作方式。

定时/计数控制寄存器为 TMRC，配合相应的 TMR 寄存器控制定时/计数器的全部操作。在使用定时器之前，需要先正确地设定定时/计数控制寄存器，以便保证定时器能正确操作，而这个过程通常在程序初始化期间完成。

定时/计数控制寄存器的第 7 位和第 6 位，即 T0M1/T0M0，用来设定定时器的工作模式。定时/计数控制寄存器的第 4 位即 T0ON，用于定时器开关控制，设定为逻辑高时，计数器开始计数，而清零时则停止计数。定时/计数控制寄存器的第 0~2 位用来控制输入时钟预分频器。如果使用外部时钟源，预分频器位将不起作用。如果定时/计数器工作在外部事件计数模式或脉冲宽度测量模式，T0EG 位即 TMRC 寄存器的第 3 位将可用来选择上升沿或下降沿触发。T0S 位用来选择内部时钟源。

- **TMR0C 寄存器**

位	7	6	5	4	3	2	1	0
名称	T0M1	T0M0	T0S	T0ON	T0EG	T0PSC2	T0PSC1	T0PSC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	0	0	0

Bit 7,6 **T0M1**, **T0M0**: 选择 Timer0 工作模式
00: 无可用模式
01: 计数器模式
10: 定时器模式
11: 脉冲宽度测量模式

Bit 5 **T0S**: 定时器时钟源
0: f_{SYS}
1: LXT 振荡器
T0S 用来选择 Timer0，时基和 PWM 的时钟源 f_{TP}，
如果 PWM 使能， f_{TP} 固定 f_{SYS} ，忽略 T0S 的设置。

Bit 4 **T0ON**: 定时/计数器使能
0: 除能
1: 使能

Bit 3 **T0EG**:
计数器有效边沿选择
0: 在上升沿计数
1: 在下降沿计数
脉冲宽度测量有效边沿选择
0: 在下降沿启动计数，在上升沿停止计数
1: 在上升沿启动计数，在下降沿停止计数

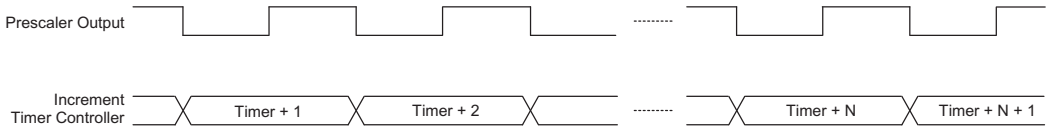
Bit 2~0 **T0PSC2**, **T0PSC1**, **T0PSC0**: 选择定时器预分频比
定时器内部时钟 =
000: f_{TP}
001: $f_{TP}/2$
010: $f_{TP}/4$

011: $f_{TP}/8$
100: $f_{TP}/16$
101: $f_{TP}/32$
110: $f_{TP}/64$
111: $f_{TP}/128$

定时器模式

在这个模式下，定时器可以用来测量固定时间间隔，当定时器发生溢出时，就会产生一个内部中断信号。为使定时/计数器工作在定时器模式，T0M1/T0M0 需要设置成 1 和 0。

在定时器模式中， f_{SYS}、$f_{SYS}/4$ 或者 LXT 振荡器被用来当定时器的输入时钟源。然而，该定时器时钟源被预分频器进一步分频，分频比是由定时器控制寄存器的 T0PSC2~T0PSC0 位来确定。定时器控制寄存器第 4 位，即 T0ON 位需要设为逻辑高，才能令定时器工作。每次内部时钟由高到低的电平转换都会使定时器值增加一。当定时器计数已满即溢出时，会产生中断信号且定时器会重新载入预置寄存器的值，然后继续计数。定时器溢出以及相应的内部中断产生也是唤醒暂停模式的一种方法。通过设置中断寄存器 INTC0 中的位 ET0I 为 0 ，可以禁止计数器中断。



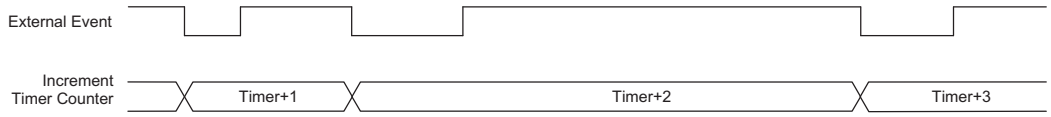
定时器模式时序图

外部事件计数模式

定时/计数器工作在外部事件计数模式，可以通过定时/计数器来记录发生在 TC0 引脚的外部逻辑事件变化的次数。为使定时/计数器工作在外部事件计数模式，T0M1/T0M0 需要设置成 0 和 1。

在外部事件计数模式，外部定时脚 TC0 被用来当定时/计数器的计时源且不被内部预分频器进一步分频。在设置完定时/计数控制寄存器其它位，定时/计数器控制寄存器第 4 位，即 T0ON 位需要设为逻辑高，才能使计数器工作。当定时控制寄存器第 3 位，即 T0EG 设置为逻辑低时，每次外部计数引脚接收到由低到高电平的转换将使计数器加一。而当 T0EG 为逻辑高时，每次外部定时/计数器引脚接收到由高到低电平的转换将使计数器加一。当计数器计数满，即溢出时会产生中断信号且计数器会重新加载预置寄存器的值，然后继续计数。计数器溢出中断可通过设置相应的中断寄存器中的定时/计数器中断使能位为 0 而禁止。

由于外部时钟引脚和普通输入/输出引脚共用，为了确保工作在外部事件计数模式，要注意两点。首先是要将定时/计数器的工作模式设定在事件计数模式，其次是确定端口控制寄存器将这个引脚设定为输入状态。注意，在外部事件计数模式下，当单片机工作在休眠模式时也保持对外部 TC0 引脚的事件计数功能。当计数器溢出时，将产生一个定时器中断，并且可以作为唤醒暂停模式的一种方法。



事件计数器模式时序图（T0EG=1）

脉冲宽度测量模式

定时/计数器工作在脉冲宽度测量这个模式，可以测量外部定时器引脚上的外部脉冲宽度。为使定时/计数器工作在脉冲宽度测量模式，T0M1/T0M0 需要设置 1 和 1。

在脉冲宽度测量模式中，f_{SYS}，$f_{SYS}/4$ 或者 LXT 作为 8 位定时/计数器的内部时钟源，并可被预

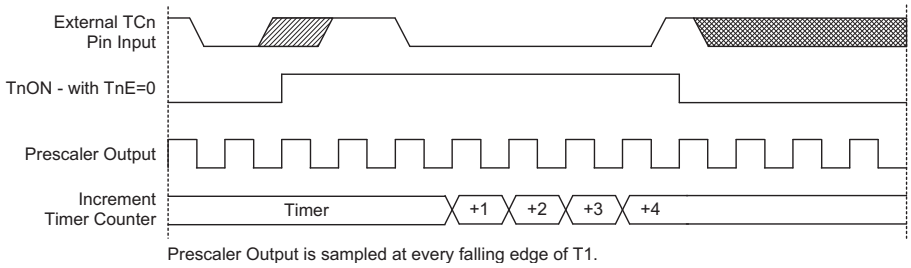
分频器进一步分频。分频比由预分频选择位 T0PSC2~T0PSC0，即定时控制寄存器的第 2~0 位来确定。在设置完定时/计数控制寄存器其它位，定时器控制寄存器第 4 位，即 T0ON 位需要设为逻辑高，才能使定时/计数器工作。然而，只有当在外部定时器引脚上接收到有效的逻辑转换时，定时/计数器才真正开始启动计数。

当定时控制寄存器第 3 位，即 T0EG 设置为逻辑低时，每次外部定时器引脚接收到由高到低电平的转换时将开始计数直到外部定时/计数器引脚回到它原来的高电平。此时使能位将自动清除为 0 以停止计数。而当 T0EG 为逻辑高时，每次外部定时器接收到由低到高电平的转换时将开始计数直到外部定时/计数器引脚回到它原来的低电平。同样使能位将自动清除为 0 以停止计数。注意，在脉冲宽度测量模式中，当外部定时器上的外部控制信号回到它原来的电平时，使能位将自动地清除为 0。而在其它两种模式，使能位只能在程序控制下清除为 0。

可以通过程序读取定时/计数器当前值，获得 TC0 外部引脚的信号脉冲宽度。当使能位重新复位，任何出现在外部定时器引脚上信号脉冲将被忽略。直到使能位被程序重新置高，开始重新测量外部脉冲。这种方式使得测量窄脉冲将会很容易实现。

注意，在这种模式下，定时/计数器是通过外部定时器引脚上的逻辑转换来控制，而不是通过逻辑电平。当定时/计数器计满，即溢出时会产生中断信号且定时/计数器会重新加载预置寄存器的值，然后继续向上计数。定时/计数器溢出中断可通过设置相应的中断寄存器中的定时/计数器使能位为 0 而禁止。

由于 TC0 引脚和普通输入/输出引脚共用，为了确保工作在脉冲宽度测量模式，要注意两点。首先是要将定时/计数器的工作模式设定在脉冲宽度测量模式，其次是确定端口控制寄存器将这个引脚设定为输入状态。



脉冲宽度测量模式时序图（T0EG=0）

预分频器

TMR0C 寄存器的 T0PSC0~T0PSC2 位用来确定定时/计数器的内部时钟的分频比，从而能够设置更长的定时器溢出周期。

PFD 功能

PFD（Programmable Frequency Divider）提供了一个可编程分频器，适用于像蜂鸣器驱动或者其它需要精确频率的场合。

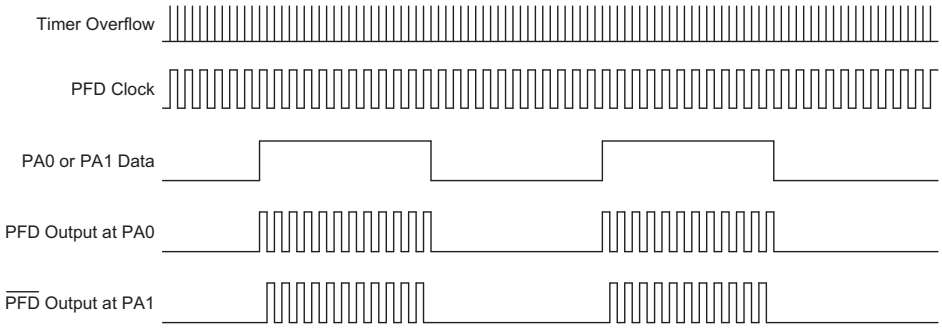
对于不同的单片机，PFD引脚可做单一的输出PFD，也可以做成对输出PFD和$\overline{\text{PFD}}$，此引脚是与输入/输出口共用的，由CTRL0寄存器来控制。要注意的是$\overline{\text{PFD}}$引脚是PFD引脚的互补输出，两者配合可以产生更强的输出功率以驱动蜂鸣器等器件。控制寄存器CTRL0中的PFDEN[1:0]可选择单一的PFD输出，或一对输出PFD和$\overline{\text{PFD}}$提供给需要双输出的设备。

PFD 功能的时钟源是定时/计数器溢出信号，由 CTRL0 中的 PFDCS 来控制。该时钟源可以来自定时/计数器 0。输出频率的控制可以通过设置适当值到预分频器和定时寄存器来达到要求的分频比。计数器将开始从预置值开始向上计数，直到满，此时，将产生一个溢出信号，导致 PFD 和$\overline{\text{PFD}}$输出改变状态。然后，计数器将自动从预置寄存器中加载预置值，并且继续向上计数。

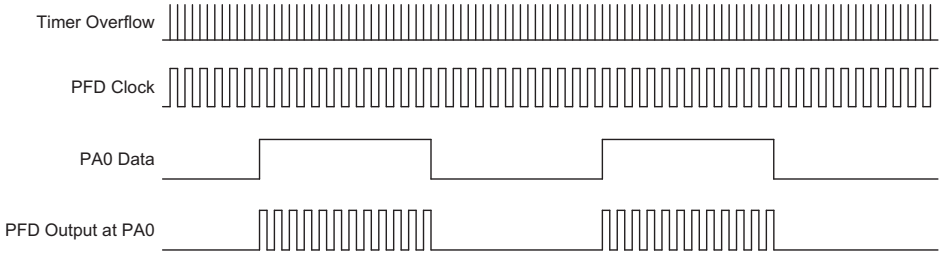
如果 CTRL0 寄存器已经选择 PFD 功能，为了能对 PFD 输出进行操作，PA 控制寄存器 PAC 的设置是至关重要的，需要设置 PFD 引脚为输出。当其中一只引脚设置为 PFD 输出时，另一个引脚可作为普通的数据输入引脚使用。然而，如果两个引脚都设置为输入，则 PFD 功能将关闭。对于具

有双输出 PFD 的单片机而言，PA0 设置为高会激活 PFD。而具有单输出 PFD 的单片机，PA1 需要设置为高来激活 PFD。输出数据位可以用来作为 PFD 输出的开关控制位。需要注意的是，如果输出数据位清零，PFD 输出将一直输出低。

如果使用晶体振荡器作为系统时钟，通过该方法能产生非常准确的频率。



PFD功能-互补输出



PFD功能-单通道输出

输入/输出接口

当定时/计数器运行在计数或者脉冲宽度测量模式下，定时/计数器需要使用外部定时器引脚以确保正确的动作。由于该引脚是共用引脚，因此需要正确的将其配置为定时/计数器输入引脚。这可以通过定时/计数器控制寄存器的模式选择位来选择是计数模式或者脉冲宽度测量模式。此外，相应的端口控制寄存器位需要被设置为高，来确保该引脚是作为输入脚。即使该引脚用作定时/计数器输入，任何连接到这个引脚的上拉电阻将仍然是有效的。

编程注意事项

当定时/计数器工作在定时器模式时，内部的系统时钟作为定时器的时钟源，因此与单片机所有运算都能同步。在这个模式下，当定时器寄存器溢出时，微控制器将产生一个内部中断信号，使程序进入相应的内部中断向量。对于脉冲宽度测量模式，定时器时钟源同样使用内部的系统时钟，然而，只有正确的逻辑条件出现在定时器输入引脚时，定时器才开始运行。当这个外部事件没有和内部定时器时钟同步时，只有当下一个定时器时钟到达时，单片机才会看到这个外部事件，因此在测量值上可能有小的差异，需要程序设计者在程序应用时加以注意。同样的情况发生在定时器设置为外部事件计数模式时，它的时钟来源是外部事件，与内部系统时钟或者定时器时钟不同步。

当读取定时/计数器值或写数据到预置寄存器时，计数时钟会被禁止以避免发生错误，但这样做可能会导致计数错误，所以程序设计者应该考虑到这点。在第一次使用定时/计数器之前，要仔细确认有没有正确地设定初始值。中断控制寄存器中的定时器使能位需要正确的设置，否则相应定时/计数器内部中断仍然无效。定时/计数器控制寄存器中的触发边沿选择、定时/计数器工作模式和时钟源控制位也需要正确的设定，以确保定时/计数器按照应用需求而正确的配置。在定时/计数器打开之前，需要确保先载入定时/计数器寄存器的初始值，这是因为在上电后，定时/计数器寄存器中的初始值是未知的。定时/计数器初始化后，可以使用定时/计数器控制寄存器中的使能位来打开或

关闭定时器。

当定时/计数器产生溢出，中断控制寄存器中相应的中断请求标志将置位。若中断允许，将会依次产生一个中断信号。不管中断是否允许，在省电状态下，定时/计数器的溢出也会产生唤醒。这种情况可能发生在外部信号变化的计数模式中。定时/计数器向上计数直至溢出并唤醒系统。若在省电模式下，不需要定时器中断唤醒系统，可以在执行“HALT”指令之前将相应中断请求标志位置位。

- **定时/计数器应用范例**

这个例子说明了如何设置定时/计数器的寄存器，如何设置和控制中断。另外还需注意怎样通过寄存器的第 4 位来启停定时/计数器。此应用范例设置定时/计数器为定时模式，时钟来源于内部的系统时钟。

- **PFD 编程应用范例**

```
org     04h             ; external interrupt vector

org     08h             ; Timer Counter 0 interrupt vector
jmp     tmr0int         ; jump here when Timer 0 overflows
:       :
org     20h             ; main program
:       :
;internal Timer 0 interrupt routine
tmr0int:
:
;Timer 0 main program placed here
:
:
begin:
;setup Timer 0 registers
mov     a,09bh          ; setup Timer 0 preload value
mov     tmr0,a
mov     a,081h          ; setup Timer 0 control register
mov     tmr0c,a         ; timer mode and prescaler set to /2
;setup interrupt register
mov     a,00dh          ; enable master interrupt and both timer interrupts
mov     intc0,a
:       :
set     tmr0c.4         ; start Timer 0
:       :
```

时基功能

此单片机具有时基功能，用来产生一个有规律的时间间隔信号。

时基功能中时间长度可以通过内部 13 级计数器设置时钟源的分频比来实现，而分频比则是由 CTRL1 寄存器中的 TBSEL0 和 TBSEL1 来设置。另外，TMR0C 寄存器中的 T0S 位可以用来选择时基的时钟源。

当时基溢出时，将产生一个时基中断信号。需要注意的是，时基中断时钟源和定时/计数器的时钟源相同，在编程的时候要多加小心。

中断

中断是单片机一个重要功能。当发生外部中断或内部中断（如定时/计数器和时基），系统会中止当前的程序，而转到相对应的中断服务程序中。此系列每一款单片机均提供一个外部中断和多个内部中断，外部中断由 INT 引脚信号触发，而内部中断由定时/计数器和时基溢出控制。

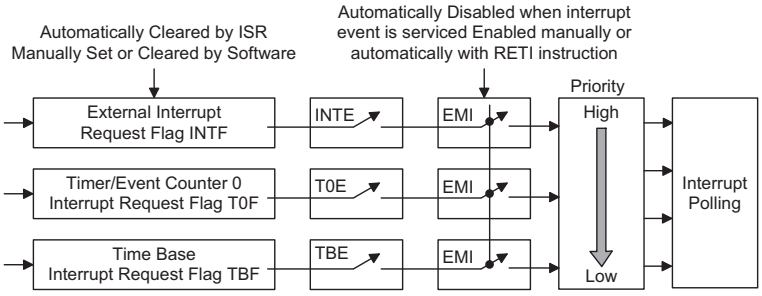
中断寄存器

所有中断允许和请求标志均由 INTC0 和 INTC1 寄存器控制。通过控制相应的中断使能位实现对应中断的打开或关闭。当发生中断，相应有中断请求标志将被置位。总中断请求标志清零将关闭所有中断允许。

中断操作

定时/计数器溢出、时基事件或者外部中断引脚上有一个有效的边沿信号都会产生一个中断请求，如果相应的中断允许，系统将要执行指令的下条地址压入堆栈，并将相应的中断向量地址加载至 PC 中，然后从此向量取下条指令。中断向量处通常为跳转指令，以跳转到相应的中断服务程序。中断服务程序必须以 RETI 指令返回，系统将先前压入堆栈的地址返回 PC，以继续执行中断发生时的程序。

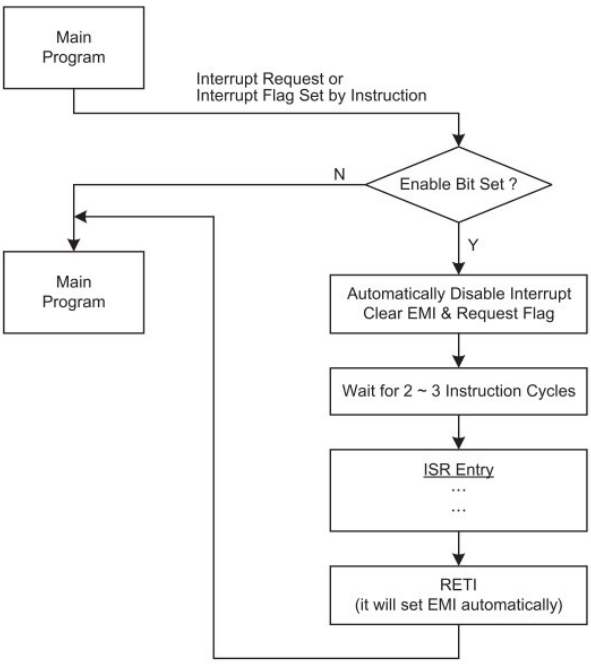
各个中断使能位以及相应的请求标志位，以优先级的顺序如下图所示。



中断示意图

一旦中断子程序被响应，所有其它的中断将被屏蔽（系统自动清除 EMI 位），这个方式可以防止中断嵌套。如果其它的中断请求发生在此期间，只有中断请求标志位会被置位。也可以开启中断嵌套，即在某个中断服务子程序中置位 EMI，此时如果有另一个中断发生则允许响应此中断。如果堆栈已满，即使此中断使能，中断请求也不会被响应，直到 SP 减少为止。如果要求中断服务程序立即响应，则堆栈必须避免成为储满状态。

当中断请求产生后，需要插入 2 个或 3 个指令周期，程序才能跳转到相应的中断向量地址。而单片机在休眠模式被唤醒时，需要插入 3 个指令周期，程序才能跳转到响应的中断向量地址。



中断流程图

中断优先级

当中断发生在两个连续的 T2 脉冲上升沿之间时，如果相应的中断请求被允许，中断将在后一个 T2 脉冲响应。下表指出在同时提出请求的情况下的优先权。中断请求可以通过重新设定 EMI 位来加以屏蔽。

中断源	优先级	向量
外部中断	1	04H
定时/计数器0溢出中断	2	08H
时基溢出中断	3	0CH

当外部中断和内部中断均被使能，如果同时发生中断，则外部中断永远优先处理，首先被响应。使用中断寄存器适当地屏蔽个别中断，可以防止同时发生的情况。

外部中断

要使外部中断发生，总中断控制位 EMI、外部中断使能位 INTE 需要先被置位。外部中断通过外部 INT 引脚上的电平转换来触发，并置位外部中断请求标志位 INTF。通过 INTEG0 和 INTEG1 位(CTRL1 寄存器的第 6 位和第 7 位)可以设置外部中断触发方式为下降沿触发、上升沿触发或者双边沿触发，也可以设置关闭外部中断功能。

INTEG1	INTEG0	边沿触发类型
0	0	外部中断关闭
0	1	上升沿触发
1	0	下降沿触发
1	1	双边沿触发

外部中断与 PA3 共用引脚，如果 INTC0 中相应的外部中断使能位被置位并且在 CTRL1 寄存器中也设置了中断边沿触发类型，PA3 将只能被作为外部中断输入口使用，同时 PAC.3 需将 PA3 设

为输入口。当中断使能、堆栈未满且外部中断产生时，将调用位于地址 04H 处的子程序。当进入外部中断服务程序时，外部中断请求标志位 INTF， EMI 位都会被自动清零以屏蔽其它中断。注意，即使作为外部中断引脚，PA3 依然可以设置带有上拉电阻功能。

- **INTC0 寄存器**

位	7	6	5	4	3	2	1	0
名称	—	TBF	T0F	INTF	TBE	T0E	INTE	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **TBF**：时基中断请求标志
0：无效
1：有效

Bit 5 **T0F**: 定时/计数器 0 中断请求标志位
0: 无效
1: 有效

Bit 4 **INTF**: 外部中断请求标志位
0: 无效
1: 有效

Bit 3 **TBE**: 时基中断使能
0: 除能
1: 使能

Bit 2 **T0E**: 定时/计数器 0 中断使能
0: 除能
1: 使能

Bit 1 **INTE**: 外部中断使能
0: 除能
1: 使能

Bit 0 **EMI**: 总中断使能
0: 除能
1: 使能

定时/计数器中断

要产生定时/计数器中断，总中断控制位 EMI 和相应的定时/计数器中断使能位 TnE 需要先被置位。当定时/计数器发生溢出，相应的中断请求标志位 TnF 将置位并触发定时/计数器中断。若中断使能，堆栈未满，当发生定时/计数器中断时，将调用相应定时器中断子程序。当定时/计数器中断被响应时，中断请求标志位 TnF 被复位且 EMI 被清零以除能其它中断。

时基中断

要产生时基中断，总中断使能位 EMI 和时基中断使能位 TBE 需要先被置位。当时基发生溢出，将置位时基请求标志位 TBF，并触发时基中断。当中断被允许且堆栈未满，一旦时基发生溢出，将调用相应时基中断子程序。当时基中断响应时，时基中断标志位 TBF 被复位且 EMI 位被清零以除能其它中断。

编程注意事项

通过除能中断使能位，可以屏蔽中断请求。然而，一旦请求标志位被置位，它将保存在中断寄存器中，直到相应的中断被响应或被软件指令清除。

建议用户不要在中断子程序中使用“Call 子程序”指令。中断通常发生在不可预料的情况或需要立即执行的某些应用。假如只剩下一层堆栈且没有控制好中断，一旦“Call 子程序”在中断子程序中执行时，将破坏原来的控制序列。

所有的中断都具有将处于休眠模式的单片机唤醒的功能。但只有程序计数器被压入堆栈中，一旦中断服务程序使寄存器和状态寄存器中的内容发生改变，则会破坏想要的控制序列，因此需要事先将这些数据保存起来。

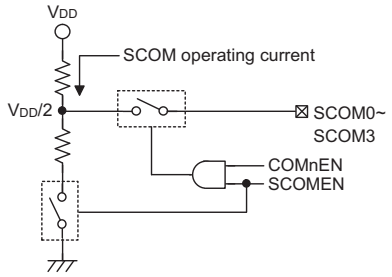
带 SCOM 功能的 LCD

此系列的单片机可以用来驱动外部 LCD 面板。LCD 驱动的 COM 口（SCOM0~SCOM3）与 PB0~PB3 引脚共用。LCD 控制信号(COM & SEG)由软件编程实现。

LCD 操作

单片机通过设置 PB0~PB3 作为 COM 引脚,其它输出口作为 SEG 引脚，以驱动外部的液晶面板。LCD 驱动功能是由 SCOMC 寄存器来控制，另外，该寄存器可设置 LCD 的开启和关闭以及输出偏压值等功能，使得 COM 口输出 $V_{DD}/2$ 的电压，从而实现 1/2biasLCD 的显示。

SCOMC 寄存器中的 SCOMEN 位是 LCD 驱动的主控制位，它与 COMnEN 位搭配共同设置 PB 端口是否用于 LCD 驱动。注意，作为 LCD 驱动时，PBC 控制寄存器不需要设置为输出，PB 即可输出 $V_{DD}/2$ 电压。



SCOM电路

SCOMEN	COMnEN	引脚功能	O/P Level
0	X	I/O	0或1
1	0	I/O	0或1
1	1	SCOMN	$V_{DD}/2$

输出控制

LCD 偏压控制

LCD 驱动器可以提供多种驱动电流选择以适应不同 LCD 面板的需求。通过设置 SCOMC 寄存器中 ISEL0 位和 ISEL1 位可以配置不同的驱动电流。

• **SCOMC寄存器**

位	7	6	5	4	3	2	1	0
名称	—	ISEL1	ISEL0	SCOMEN	COM3EN	COM2EN	COM1EN	COM0EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 保留位
1: 不可预测的操作 – 该位不能设置为高
0: 正确电平值 – 该位必须清除为 0

Bit 6,5 **ISEL1**, **ISEL0**: 选择 SCOM 工作电流 (V_{DD}=5V)
00: 25μA
01: 50μA
10: 100μA
11: 200μA

Bit 4 **SCOMEN**: SCOM 模块打开/关闭控制位
0: 关闭
1: 打开
如果 SCOMEN=1，COMnEN 能打开 SCOMn

Bit 3 **COM3EN**: 选择 PB3 或者 SCOM3
0: GPIO
1: SCOM3

Bit 2 **COM2EN**: 选择 PB2 或者 SCOM2
0: GPIO
1: SCOM2

Bit 1 **COM1EN**: 选择 PB1 或者 SCOM1
0: GPIO
1: SCOM1

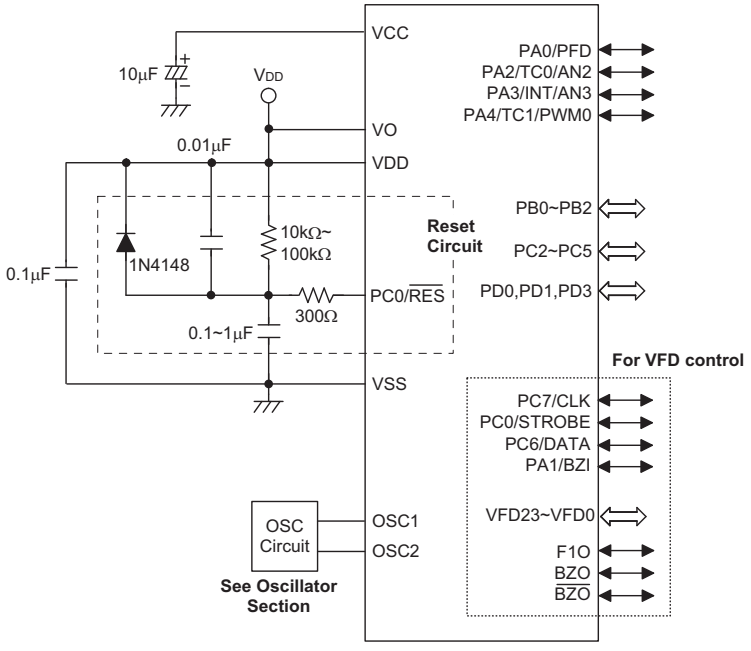
Bit 0 **COM0EN**: 选择 PB0 或者 SCOM0
0: GPIO
1: SCOM0

配置选项

配置选项在烧写程序时写入芯片。通过 HT-IDE 的软件开发环境，使用者在开发过程中可以选择配置选项。当配置选项烧入单片机后，无法再通过应用程序修改。所有位需要按系统的需要定义，具体内容可参考下表：

编号	选　　项
1	看门狗定时器：打开或关闭
2	看门狗定时器时钟源：LXT，LIRC 或者 $f_{SYS}/4$ 注：如果 WDT 时钟源来自 LXT，在 OSC 配置选项需要选择 LXT 振荡器
3	清除看门狗指令：1 条或 2 条
4	系统振荡器配置选项：HXT，HIRC，ERC，HIRC+LXT
5	LVR 功能：使能或禁止
6	LVR 电压：2.1V，3.15V 或 4.2V
7	$\overline{\text{RES}}$ 或 PA7 选择
8	系统启动延时时间： 1024 个时钟或 2 个时钟（为 HIRC/ERC 选择 t_{SST}）
9	内部 RC: 4MHz, 8MHz 或 12MHz

应用电路



指令集介绍

简介

任何单片机成功运作的核心在于它的指令集，此指令为一组程序指令码,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集，共超过 60 条，程序设计师可以事半功倍地实现他们的应用。

为了更容易的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 0.5μs 中执行完成，而分支或调用操作则将在 1μs 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，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

单片机程序的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，如同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 指令跳转到指定地址或使用 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 指令跳回。一个非常有用的分支指令是条件跳转，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

提供数据存储器单一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-写出的程序现在则由位指令所取代。

查表运算

数据的储存通常由寄存器完成，然而当处理大量的数据时，其庞大与复杂的内容常造成对指定存储器储存上的不便，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取的区域，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。

指令设定一览表

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

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	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
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

助记符	说明	周期	影响标志位
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
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，并送至数据存储器和 TBLH	2 注	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2 注	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据储存器	1 注	无
SET [m]	置位数据储存器	1 注	无
CLR WDT	清除看门狗定时器	1	TO，PDF
CLR WDT1	预清看门狗定时器	1	TO，PDF
CLR WDT2	预清看门狗定时器	1	TO，PDF
SWAP [m]	交换数据储存器的高低字节，结果放入数据存储器	1 注	无
SWAPA [m]	交换数据储存器的高低字节，结果放入 ACC	1	无
HALT	进入暂停模式	1	TO，PDF

注意： 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，如果没有跳转发生，则只需一个周期即可。
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”和“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”被连续执行后，TO和PDF标志位会被清零，除此外TO和PDF 标志位保持不变。

指令定义

ADC A, [m] Add Data Memory to ACC with Carry
指令说明 将指定数据存储器、累加器和进位标志位的内容相加后，把结果储存回累加器。
功能表示 ACC ← ACC + [m] + C
影响标志位 OV , Z , AC , C

ADCM A, [m] Add ACC to Data Memory with Carry
指令说明 将指定数据存储器、累加器和进位标志位的内容相加后，把结果储存回指定数据存储器。
功能表示 [m] ←ACC + [m] + C
影响标志位 OV , Z , AC , C

ADD A, [m] Add Data Memory to ACC
指令说明 将指定数据存储器和累加器的内容相加后，把结果储存回累加器。
功能表示 ACC ← ACC + [m]
影响标志位 OV , Z , AC , C

ADD A, x Add immediate data to ACC
指令说明 将累加器和立即数的内容相加后，把结果储存回累加器。
功能表示 ACC ← ACC + x
影响标志位 OV , Z , AC , C

ADDM A, [m] Add ACC to Data Memory
指令说明 将指定数据存储器和累加器的内容相加后，把结果储存回指定数据存储器。
功能表示 [m] ←ACC + [m]
影响标志位 OV , Z , AC , C

AND A, [m] Logical AND Data Memory to ACC
指令说明 将存在累加器和指定数据存储器中的数据作AND的运算，然后把结果储存回累加器。
功能表示 ACC ← ACC“AND”[m]
影响标志位 Z

AND A, x Logical AND immediate data to ACC
指令说明 将存在累加器中的数据和立即数作AND的运算，然后把结果储存回累加器。
功能表示 ACC ← ACC“AND”x
影响标志位 Z

ANDM A, [m] Logical AND ACC to Data Memory
指令说明 将存在指定数据存储器和累加器中的数据作AND的运算，然后把结果储存回数据存储器。
功能表示 [m] ← ACC“AND”[m]
影响标志位 Z

CALL addr Subroutine call
指令说明 无条件地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继续执行程序，由于此指令需要额外的运算，所以为一个2周期的指令。
功能表示 Stack ← Program Counter + 1

Program Counter ← addr
影响标志位 None

CLR [m] Clear Data Memory
指令说明 指定数据存储器中的每一位均清除为0。
功能表示 [m] ← 00H
影响标志位 None

CLR [m].i Clear bit of Data Memory
指令说明 指定数据存储器中的i位清除为0。
功能表示 [m].i ← 0
影响标志位 None

CLR WDT Clear Watchdog Timer
指令说明 将TO、PDF 标志位和WDT全都清零。
功能表示 WDT cleared
TO ← 0
PDF ← 0
影响标志位 TO , PDF

CLR WDT1 Pre-clear Watchdog Timer
指令说明 将TO、PDF 标志位和WDT全都清零，请注意此指令要结合CLR WDT2一起动作且必须交替执行才有作用，重复执行此项指令而没有与CLR WDT2交替执行将无任何作用。
功能表示 WDT cleared
TO ← 0
PDF← 0
影响标志位 TO , PDF

CLR WDT2 Pre-clear Watchdog Timer
指令说明 将TO、PDF标志位和WDT全都清零，请注意此指令要结合CLR WDT1一起动作且必须交替执行才有作用，重复执行此项指令而没有与CLR WDT1交替执行将无任何作用。
功能表示 WDT cleared
TO ← 0
PDF← 0
影响标志位 TO , PDF

CPL [m] Complement Data Memory
指令说明 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0变1。
功能表示 [m] ← $[\overline{m}]$
影响标志位 Z

CPLA [m] Complement Data Memory with result in ACC
指令说明 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0变1，而结果被储存回累加器且数据存储器中的内容不变。
功能表示 ACC←$[\overline{m}]$
影响标志位 Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory
指令说明 将存在累加器中的内容数值转换为BCD（二进制转成十进制）数值，如果低4位大于9或AC标志位被置位，则在低4位加上一个6，不然低4位的内容不变，如果高4位大于9或C标志位被置位，则在高4位加上一个6，十进制的转换主要是依照累加器和标志位状况，分别加上00H、06H、60H或66H，只有C标志位也许会被此指令影响，它会指出原始BCD数是否大于100，并可以进行双精度十进制数相加。
功能表示 [m] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← ACC + 66H
影响标志位 C

DEC [m] Decrement Data Memory
指令说明 将在指定数据存储器内的数据减1。
功能表示 [m] ← [m] – 1
影响标志位 Z

DECA [m] Decrement Data Memory with result in ACC
指令说明 将在指定数据存储器内的数据减1，把结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC ← [m] – 1
影响标志位 Z

HALT Enter power down mode
指令说明 此指令停止程序的执行并且关闭系统时钟，但数据存储器和寄存器的内容仍被保留，WDT和预分频器(Prescaler)被清零，暂停标志位PDF被置位且WDT溢出标志位TO 被清零。
功能表示 TO ← 0
PDF ← 1
影响标志位 TO , PDF

INC [m] Increment Data Memory
指令说明 将指定数据存储器内的数据加1。
功能表示 [m] ← [m] + 1
影响标志位 Z

INCA [m] Increment Data Memory with result in ACC
指令说明 将指定数据存储器内的数据加1，把结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC ← [m] + 1
影响标志位 Z

JMP addr Jump unconditionally
指令说明 程序计数器的内容被指定地址所取代，程序由新地址继续执行，当新地址被加载入时，必须插入一个空指令周期，所以此指令为2个周期的指令
功能表示 Program Counter ← addr
影响标志位 None

MOV A, [m] Move Data Memory to ACC
指令说明 将指定数据存储器的内容复制到累加器中。
功能表示 ACC← [m]
影响标志位 None

MOV A, x Move immediate data to ACC
指令说明 将立即数载入至累加器中。
功能表示 ACC ← x
影响标志位 None

MOV [m], A Move ACC to Data Memory
指令说明 将累加器的内容复制到指定数据存储器。
功能表示 [m] ← ACC
影响标志位 None

NOP No operation
指令说明 空操作，接下来顺序执行下一条指令。
功能表示 No operation
影响标志位 None

OR A, [m] Logical OR Data Memory to ACC
指令说明 将存在累加器和指定数据存储器中的数据作OR的运算，然后把结果储存回累加器。
功能表示 ACC □ ACC“OR”[m]
影响标志位 Z

OR A, x Logical OR immediate data to ACC
指令说明 将存在累加器中的数据和立即数作OR的运算，然后把结果储存回累加器。
功能表示 ACC ← ACC“OR”x
影响标志位 Z

ORM A, [m] Logical OR ACC to Data Memory
指令说明 将存在指定数据存储器和累加器中的数据作OR的运算，然后把结果储存回数据存储器。
功能表示 [m] ← ACC“OR”[m]
影响标志位 Z

RET Return from subroutine
指令说明 将堆栈区的数据取回至程序计数器，程序由取回的地址继续执行。
功能表示 Program Counter←Stack
影响标志位 None

RET A, x Return from subroutine and load immediate data to ACC
指令说明 将堆栈区的数据取回至程序计数器且累加器载入立即数，程序由取回的地址继续执行。
功能表示 Program Counter ← Stack
ACC←x
影响标志位 None

RETI Return from interrupt
指令说明 将堆栈区的数据取回至程序计数器且中断功能通过EMI位重新被使能，EMI是控制中断使能的主中断位(寄存器INTC的第0位)，如果在执行RETI指令之前还有中断未被响应，则这个中断将在返回主程序之前被响应。
功能表示 Program Counter ←Stack
EMI ← 1
影响标志位 None

RL [m] Rotate Data Memory left
指令说明 将指定数据存储器的内容向左移1个位，且第7位移回第0 位。
功能表示 [m].(i+1) ← [m].i ; (i = 0~6)
[m].0 ← [m].7
影响标志位 None

RLA [m] Rotate Data Memory left with result in ACC
指令说明 将指定数据存储器的内容向左移1个位，且第7位移回第0位，而移位的结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC.(i+1) ← [m].i ; (i = 0~6)
ACC.0 ←[m].7
影响标志位 None

RLC [m] Rotate Data Memory Left through Carry
指令说明 将指定数据存储器的内容连同进位标志位向左移1个位，第7位取代进位位且原本的进位标志位移至第0位。
功能表示 [m].(i+1) ← [m].i ; (i = 0~6)
[m].0 ← C
C ← [m].7
影响标志位 C

RLCA [m] Rotate Data Memory left through Carry with result in ACC
指令说明 将指定数据存储器的内容连同进位标志位向左移1个位，第7位取代进位位且原本的进位标志位移至第0位，而移位的结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC.(i+1) ← [m].i ; (i = 0~6)
ACC.0 ← C
C ← [m].7
影响标志位 C

RR [m] Rotate Data Memory right
指令说明 将指定数据存储器的内容向右移1个位，且第0位移回第7位。
功能表示 [m].i ← [m].(i+1) ; (i = 0~6)
[m].7 ← [m].0
影响标志位 None

RRA [m] Rotate Data Memory right with result in ACC
指令说明 将指定数据存储器的内容向右移1个位，且第0位移回第7位，而移位的结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC.i ← [m].(i+1) ; (i = 0~6)
ACC.7 ← [m].0
影响标志位 None

RRC [m] Rotate Data Memory right through Carry
指令说明 将指定数据存储器的内容连同进位标志位向右移1个位，第0位取代进位位且原本的进位标志位移至第7位。
功能表示 [m].i ← [m].(i+1) ; (i = 0~6)
[m].7← C
C ← [m].0
影响标志位 C

RRCA [m] Rotate Data Memory right through Carry with result in ACC
指令说明 将指定数据存储器的内容连同进位标志位向右移1个位，第0位取代进位位且原本的进位标志位移至第7位，而移位的结果储存回累加器且数据存储器中的内容不变。
功能表示 ACC.i ← [m].(i+1) ; (i = 0~6)
ACC.7 ← C
C ← [m].0
影响标志位 C

SBC A, [m] Subtract Data Memory from ACC with Carry
指令说明 将累加器中的数据与指定数据存储器内容和进位标志位的反相减，把结果储存回累加器。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。
功能表示 ACC ← ACC – [m] – $\overline{C}$
影响标志位 OV , Z , AC , C

SBCM A, [m] Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明 将累加器中的数据与指定数据存储器内容和进位标志位的反相减，把结果储存回数据存储器。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。
功能表示 [m] ← ACC – [m] – $\overline{C}$
影响标志位 OV , Z , AC , C

SDZ [m] Skip if Decrement Data Memory is 0
指令说明 将指定数据存储器的内容先减去1后，如果结果为0，则程序计数器再加1跳过下一条指令，由于取得下一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，则程序继续执行下面的指令。
功能表示 [m] ← [m] – 1
Skip if [m] = 0
影响标志位 None

SDZA [m] Skip if decrement Data Memory is zero with result in ACC
指令说明 将指定数据存储器的内容先减去1后，如果结果为0，则程序计数器再加1 跳过下一条指令，此结果会被储存回累加器且指定数据存储器中的内容不变，由于取得下一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，则程序继续执行下面的指令。
功能表示 ACC ← [m] – 1
Skip if ACC = 0
影响标志位 None

SET [m] Set Data Memory
指令说明 将指定数据存储器的每一个位置位为1。
功能表示 [m] ← FFH
影响标志位 None

SET [m].i Set bit of Data Memory
指令说明 将指定数据存储器的第i位置位为1。
功能表示 [m].i ← 1
影响标志位 None

SIZ [m] Skip if increment Data Memory is 0
指令说明 将指定数据存储器的内容先加上1后，如果结果为0，则程序计数器再加1 跳过下一条指令，由于取得下一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，则程序继续执行下面的指令。
功能表示 [m] ← [m] + 1
Skip if [m] = 0
影响标志位 None

SIZA [m] Skip if increment Data Memory is zero with result in ACC
指令说明 将指定数据存储器的内容先加上1后，如果结果为0，则程序计数器再加1跳过下一条指令，此结果会被储存回累加器且指定数据存储器中的内容不变，由于取得下一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，则程序继续执行下面的指令。
功能表示 ACC← [m] + 1
Skip if ACC = 0
影响标志位 None

SNZ [m].i Skip if bit i of Data Memory is not 0
指令说明 如果指定数据存储器的第i位不为0，则程序计数器再加1跳过下一条指令，由于取得下一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，程序继续执行下面的指令。
功能表示 Skip if [m].i ≠ 0
影响标志位 None

SUB A, [m] Subtract Data Memory from ACC
指令说明 将累加器中内容减去指定数据存储器的数据，把结果储存回累加器。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。
功能表示 ACC ← ACC – [m]
影响标志位 OV , Z , AC , C

SUBM A, [m] Subtract Data Memory from ACC with result in Data Memory
指令说明 将累加器中内容减去指定数据存储器的数据，把结果储存回数据存储器。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。
功能表示 [m] ← ACC – [m]
影响标志位 OV , Z , AC , C

SUB A, x Subtract immediate Data from ACC
指令说明 将累加器中内容减去立即数，把结果储存回累加器。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。
功能表示 ACC ← ACC – x
影响标志位 OV , Z , AC , C

SWAP [m] Swap nibbles of Data Memory
指令说明 将指定数据存储器的低4位与高4位互相交换。
功能表示 [m].3 ~ [m].0 ↔ [m].7 ~ [m].4
影响标志位 None

SWAPA [m] Swap nibbles of Data Memory with result in ACC
指令说明 将指定数据存储器的低4位与高4位互相交换，然后把结果储存回累加器且数据存储器的内容不变。
功能表示 ACC.3 ~ ACC.0 ← [m].7 ~ [m].4
ACC.7 ~ ACC.4 ← [m].3 ~ [m].0
影响标志位 None

SZ [m] Skip if Data Memory is 0
指令说明 如果指定数据存储器的内容为0，则程序计数器再加1跳过下一条指令，由于取得下一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，程序继续执行下面的指令。
功能表示 Skip if [m] = 0
影响标志位 None

SZA [m] Skip if Data Memory is 0 with data movement to ACC
指令说明 将指定数据存储器的内容复制到累加器，如果值为0，则程序计数器再加1跳过下一条指令，由于取得下一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，程序继续执行下面的指令。
功能表示 ACC ← [m]
Skip if [m] = 0
影响标志位 None

SZ [m].i Skip if bit i of Data Memory is 0
指令说明 如果指定数据存储器第i 位为0，则程序计数器再加1 跳过下一条指令，由于取得下一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，程序继续执行下面的指令。
功能表示 Skip if [m].i = 0
影响标志位 None

TABRDC [m] Read table (current page) to TBLH and Data Memory
指令说明 将表格指针TBLP所指的程序代码低字节(当前页)移至指定数据存储器且将高字节移至TBLH。
功能表示 [m] ← 程序代码(低字节)
TBLH ← 程序代码(高字节)
影响标志位 None

TABRDL [m] Read table (last page) to TBLH and Data Memory
指令说明 将表格指针TBLP所指的程序代码低字节(最后一页)移至指定数据存储器且将高字节移至TBLH。
功能表示 [m] ← 程序代码(低字节)
TBLH ← 程序代码(高字节)
影响标志位 None

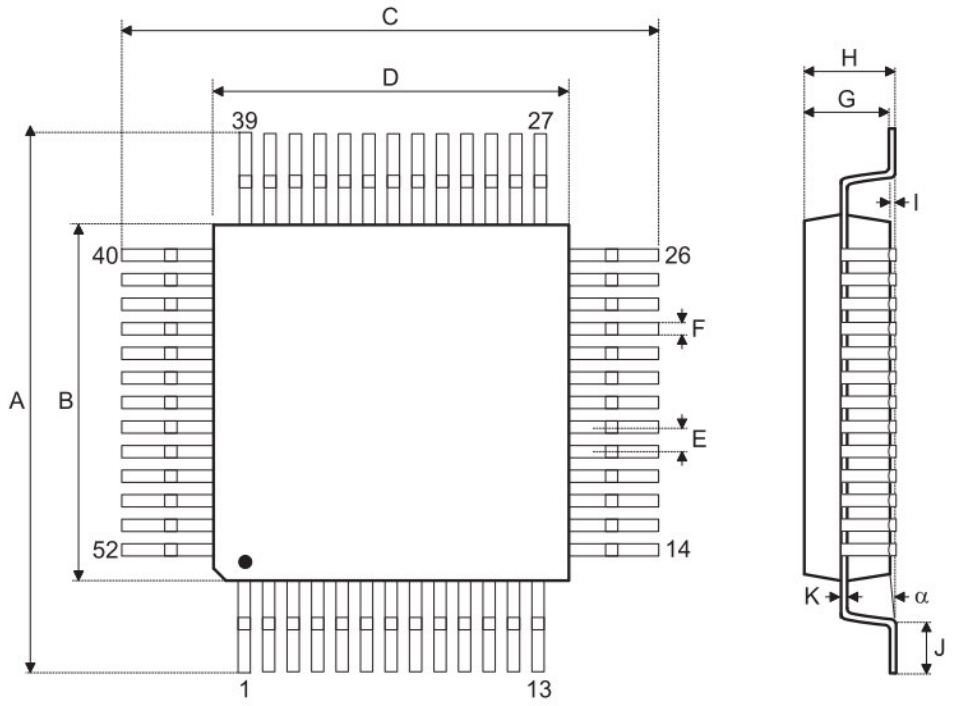
XOR A, [m] Logical XOR Data Memory to ACC
指令说明 将存在累加器和指定数据存储器中的数据作XOR的运算，然后把结果储存回累加器。
功能表示 ACC ← ACC“XOR”[m]
影响标志位 Z

XORM A, [m] Logical XOR ACC to Data Memory
指令说明 将存在指定数据存储器和累加器中的数据作XOR的运算，然后把结果储存回数据存储器。
功能表示 [m] ← ACC“XOR”[m]
影响标志位 Z

XOR A, x Logical XOR immediate data to ACC
指令说明 将存在累加器中的数据和立即数作XOR的运算，然后把结果储存回累加器。
功能表示 ACC ← ACC“XOR”x
影响标志位 Z

封装信息

52-pin QFP（10mm×10mm）外形尺寸



符号	尺寸（mm）		
	最小值	典型值	最大值
A	17.30	—	17.50
B	13.90	—	14.10
C	17.30	—	17.50
D	13.90	—	14.10
E	—	1.00	—
F	—	0.40	—
G	2.50	—	3.10
H	—	—	3.40
I	—	0.10	—
J	0.73	—	1.03
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: http://www.holtek.com