



个人护理 Flash 单片机

# HT45F3420/HT45F3430

版本 : V1.30   日期 : 2016-12-01

[www.holtek.com](http://www.holtek.com)

## 目录

<b>特性 .....</b>	<b>6</b>
CPU 特性 .....	6
周边特性 .....	6
<b>概述 .....</b>	<b>7</b>
选型表 .....	7
方框图 .....	8
引脚图 .....	8
引脚描述 .....	9
极限参数 .....	13
直流电气特性 .....	14
交流电气特性 .....	16
A/D 转换器电气特性.....	17
OCP 电气特性 .....	18
OVP 电气特性 .....	19
DLL 电气特性 .....	19
LVR 电气特性 .....	19
上电复位电气特性 .....	20
系统结构 .....	20
时序和流水线结构 .....	20
程序计数器 .....	21
堆栈 .....	22
算术逻辑单元 – ALU .....	22
<b>Flash 程序存储器 .....</b>	<b>23</b>
结构 .....	23
特殊向量 .....	23
查表 .....	23
查表范例 .....	24
在线烧录 – ICP .....	25
片上调试 – OCDS.....	25
<b>RAM 数据存储器 .....</b>	<b>26</b>
结构 .....	26
通用数据存储器 .....	26
特殊功能数据存储器 .....	26
<b>特殊功能寄存器描述 .....</b>	<b>29</b>
间接寻址寄存器 – IAR0, IAR1 .....	29
存储器指针 – MP0, MP1.....	29
存储区指针 – BP .....	30
累加器 – ACC .....	30
程序计数器低字节寄存器 – PCL.....	30

查表寄存器 – TBLP, TBLH.....	30
状态寄存器 – STATUS.....	31
<b>EEPROM 数据存储器.....</b>	<b>33</b>
EEPROM 数据存储器结构 .....	33
EEPROM 寄存器 .....	33
从 EEPROM 中读取数据 .....	35
写数据到 EEPROM 中 .....	35
写保护 .....	35
EEPROM 中断 .....	35
编程注意事项 .....	35
<b>振荡器 .....</b>	<b>37</b>
振荡器概述 .....	37
系统时钟配置 .....	37
高速内部 RC 振荡器 – HIRC .....	37
内部 32kHz 振荡器 – LIRC .....	38
辅助振荡器 .....	38
<b>工作模式与系统时钟 .....</b>	<b>38</b>
系统时钟 .....	38
系统工作模式 .....	39
控制寄存器 .....	40
工作模式切换 .....	41
待机电流注意事项 .....	44
唤醒 .....	45
<b>看门狗定时器 .....</b>	<b>45</b>
看门狗定时器时钟源 .....	45
看门狗定时器控制寄存器 .....	45
看门狗定时器操作 .....	46
<b>复位和初始化 .....</b>	<b>47</b>
复位功能 .....	47
复位初始条件 .....	49
<b>输入 / 输出端口 .....</b>	<b>53</b>
上拉电阻 .....	53
PA 口唤醒 .....	54
输入 / 输出端口控制寄存器 .....	54
引脚共用功能 .....	55
输入 / 输出端口源电流控制 – 仅用于 HT45F3430 .....	60
输入 / 输出引脚结构 .....	61
编程注意事项 .....	62
<b>集成电压分压器电路 .....</b>	<b>63</b>
功能性描述 .....	63
<b>定时器模块 – TM .....</b>	<b>65</b>
简介 .....	65
TM 操作 .....	65
TM 时钟源 .....	65

TM 中断 .....	65
TM 外部引脚 .....	66
TM 输入 / 输出引脚控制寄存器 .....	66
编程注意事项 .....	66
<b>标准型 TM – STM .....</b>	<b>67</b>
标准型 TM 操作 .....	68
标准型 TM 寄存器 .....	68
标准型 TM 工作模式 .....	72
<b>A/D 转换器.....</b>	<b>81</b>
A/D 简介 .....	81
A/D 转换器寄存器 .....	82
A/D 转换器操作 .....	85
A/D 转换器输入信号 .....	86
A/D 转换率与时序图 .....	87
A/D 转换步骤 .....	88
编程注意事项 .....	88
A/D 转换功能 .....	89
A/D 转换程序范例 .....	89
<b>过电流保护 – OCP .....</b>	<b>92</b>
OCP 操作 .....	92
OCP 控制寄存器 .....	92
输入电压范围 .....	95
失调校准 .....	95
<b>过电压保护 – OVP .....</b>	<b>96</b>
OVP 操作 .....	96
OVP 控制寄存器 .....	96
失调校准 .....	98
<b>高精度互补输出 PWM 发生器 .....</b>	<b>98</b>
功能性描述 .....	98
高精度 PWM 寄存器 .....	99
PWM 发生器 .....	102
延迟锁相环 .....	102
死区时间插入 .....	104
保护与反相控制 .....	105
编程注意事项 .....	105
<b>自动关闭控制 – 仅用于 HT45F3430 .....</b>	<b>106</b>
功能性描述 .....	106
<b>LCD SCOM 功能 – 仅用于 HT45F3430.....</b>	<b>107</b>
LCD 操作 .....	107
LCD 偏压电流控制 .....	107
<b>中断 .....</b>	<b>108</b>
中断寄存器 .....	108
中断操作 .....	113
外部中断 .....	114

OCP 中断 .....	114
OVP 中断 .....	115
多功能中断 .....	115
A/D 转换器中断 .....	115
时基中断 .....	115
EEPROM 中断 .....	116
TM 中断 .....	117
中断唤醒功能 .....	117
编程注意事项 .....	117
<b>应用电路 .....</b>	<b>118</b>
LED 手电筒 .....	118
电子烟 .....	118
<b>指令集 .....</b>	<b>119</b>
简介 .....	119
指令周期 .....	119
数据的传送 .....	119
算术运算 .....	119
逻辑和移位运算 .....	119
分支和控制转换 .....	120
位运算 .....	120
查表运算 .....	120
其它运算 .....	120
<b>指令集概要 .....</b>	<b>121</b>
惯例 .....	121
<b>指令定义 .....</b>	<b>124</b>
<b>封装信息 .....</b>	<b>136</b>
8-pin SOP (150mil) 外形尺寸 .....	137
10-pin MSOP 外形尺寸 .....	138
16-pin NSOP (150mil) 外形尺寸 .....	139
24-pin SSOP (150mil) 外形尺寸 .....	140

## 特性

### CPU 特性

- 工作电压:
  - ◆  $f_{SYS}=8MHz$ : 2.2V~5.5V
- $V_{DD}=5V$ , 系统时钟为 8MHz 时, 指令周期为 0.5 $\mu s$
- 提供暂停和唤醒功能, 以降低功耗
- 2 种内部振荡器 – 无需外部元器件
  - ◆ 8MHz 高速振荡器 – HIRC
  - ◆ 32kHz 低速振荡器 – LIRC
- 多种工作模式: 正常、低速、空闲和休眠
- 所有指令都可在 1~2 个指令周期内完成
- 查表指令
- 63 条功能强大的指令
- 4 层堆栈
- 位操作指令

### 周边特性

- Flash 程序存储器:  $1K \times 14 \sim 2K \times 15$
- RAM 数据存储器:  $64 \times 8 \sim 128 \times 8$
- True EEPROM 存储器:  $32 \times 8 \sim 64 \times 8$
- 看门狗定时器功能
- 多达 22 个双向 I/O 口
- 可编程 I/O 端口源电流, 用于 LED 驱动应用 – 仅用于 HT45F3430
- 软件控制的 1/2 bias 4-SCOM LCD 驱动功能 – 仅用于 HT45F3430
- 多达 2 个引脚与外部中断口共用
- 2 个引脚集成电压分压器电路
- 定时器模块用于时间测量、比较匹配输出、捕捉输入、PWM 输出及单脉冲输出功能
- 双时基功能, 可提供固定时间的中断信号
- 带互补输出的高精度 PWM
- 多通道 12-bit A/D 转换器
- 带中断的过电流保护功能
- 带中断的过电压保护功能
- 低电压复位功能
- Flash 程序存储器烧录可达 100,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 数据存储器烧录可达 1,000,000 次
- True EEPROM 数据存储器数据可保存 10 年以上

- 封装类型: 8-pin SOP/10-pin MSOP – HT45F3420,  
16-pin NSOP/24-pin SSOP – HT45F3430

## 概述

该系列单片机专门应用于电池功率控制，是一款 8 位具有高性能精简指令集的 Flash 型单片机。此系列单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了极大的方便。存储器方面还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，该系列单片机包含了一个多通道的 12-bit A/D 转换器，一个使用灵活的定时器模块，可提供定时功能、脉冲产生功能、捕捉输入、比较匹配输出及 PWM 产生功能。内部看门狗定时器、低电压复位等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。保护特性还包含了过电流保护和过电压保护功能。

此系列单片机内建了高速 HIRC 和低速 LIRC 振荡器作为系统振荡器功能选项，无需外部元器件。

包含 I/O 使用灵活、时基功能等其它特性，使该系列单片机可灵活应用于如电子个人护理产品及其它应用。

## 选型表

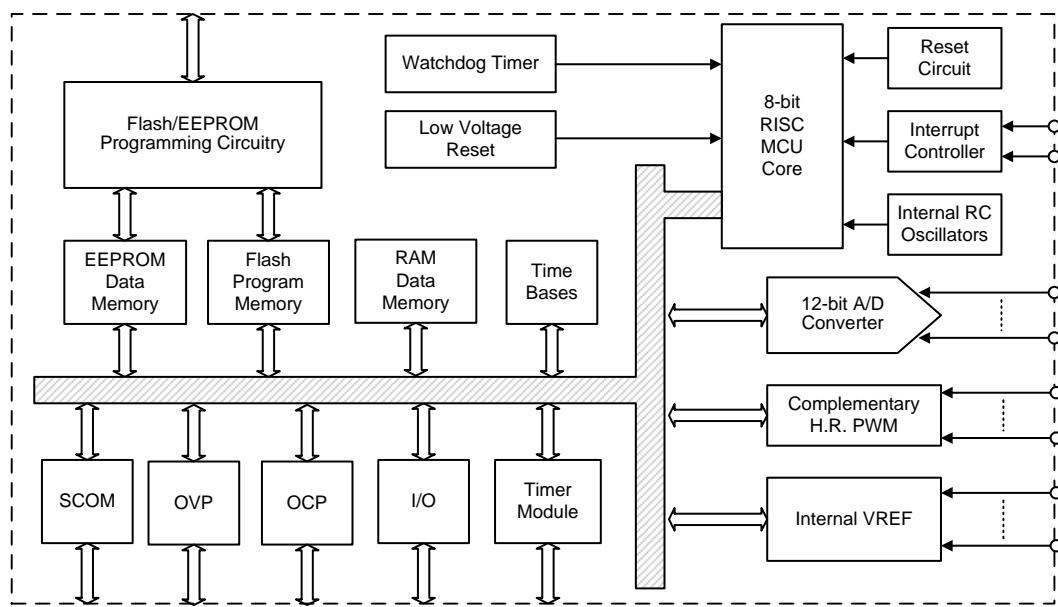
对此系列的单片机而言，大多数的特性参数都是一样的。主要差异在于存储器容量、I/O 数、外部中断数、A/D 转换器通道数、LCD/LED 驱动功能以及封装类型。下表列出了各单片机的主要特性。

单片机型号	V <sub>DD</sub>	程序存储器	数据存储器	数据EEPROM	I/O	外部中断	A/D转换器
HT45F3420	2.2V~5.5V	1K×14	64×8	32×8	8	1	12-bit×4
HT45F3430	2.2V~5.5V	2K×15	128×8	64×8	22	2	12-bit×8

单片机型号	LCD驱动器	LED驱动器	定时器模块	时基	堆栈	封装
HT45F3420	—	—	10-bit STM×1	2	4	8SOP 10MSOP
HT45F3430	4-SCOM	√	10-bit STM×1	2	4	16NSOP 24SSOP

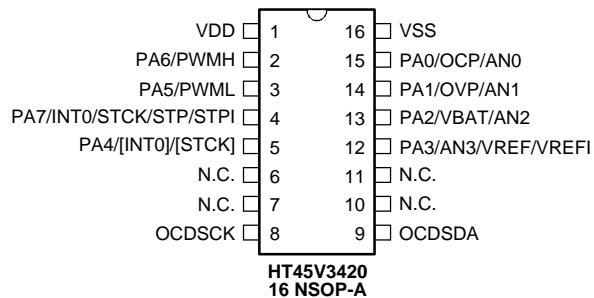
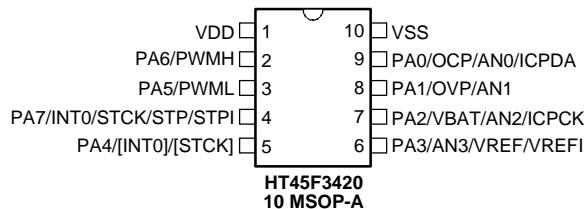
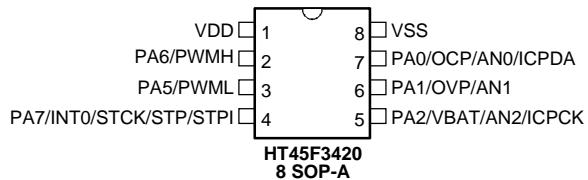
注：当单片机存在多种封装类型时，选型表反应最多引脚封装的情况。

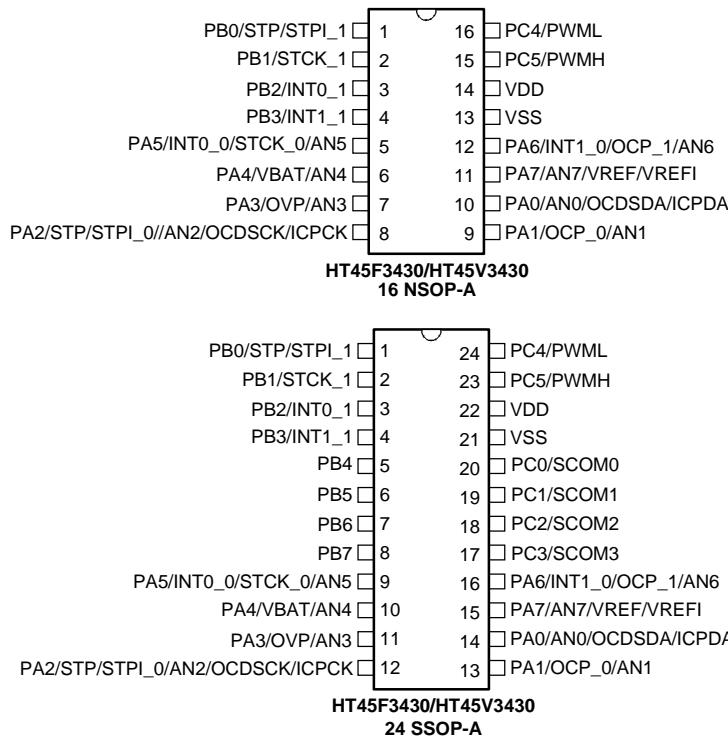
## 方框图



注：4-SCOM LCD 驱动功能仅存在于 HT45F3430。

## 引脚图





注：1. 引脚名称带后缀“\_0”或“\_1”表示非固定引脚输出重置位置，可由 IFS0 寄存器进行选择。  
 2. 若共用引脚同时有多种输出，所需引脚共用功能由相应的软件控制位决定。  
 3. OCDSC 和 OCDSDA 引脚为 OCDS 专用引脚，仅用于 HT45V3420 和 HT45V3430 EV 芯片。

## 引脚描述

除了电源引脚和一些相关的转换器控制引脚外，该系列单片机的所有引脚都以它们的端口名称进行标注，例如 PA0、PA1 等，用于描述这些引脚的数字输入 / 输出功能。然而，这些引脚也与其它功能共用，如模数转换器，定时器模块引脚等。每个引脚的功能如下表所述，而引脚配置的详细内容见规格书其它章节。该章节的引脚描述是针对较大封装的单片机，这些引脚并非都存在于小封装的单片机内。

### HT45F3420

引脚名称	功能	OPT	I/T	O/T	描述
PA0/OCP/AN0/ ICPDA	PA0	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	OCP	PAS0	AN	—	OCP 输入
	AN0	PAS0	AN	—	A/D 转换器输入通道 0
	ICPDA	—	ST	CMOS	ICP 数据

引脚名称	功能	OPT	I/T	O/T	描述
PA1/OVP/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	OVP	PAS0	AN	—	OVP 输入
	AN1	PAS0	AN	—	A/D 转换器输入通道 1
PA2/VBAT/AN2/ ICPCK	PA2	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	VBAT	PAS0	AN	—	VBAT 输入
	AN2	PAS0	AN	—	A/D 转换器输入通道 2
	ICPCK	—	ST	CMOS	ICP 时钟
PA3/AN3/VREF/ VREFI	PA3	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	AN3	PAS0	AN	—	A/D 转换器输入通道 3
	VREF	PAS0	AN	—	A/D 转换器参考输入
	VREFI	PAS0	AN	—	PGA 输入，用于 A/D 转换器参考
PA4/INT0/STCK	PA4	PAWU PAPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	INT0	IFSO	ST	—	中断输入
	STCK	IFSO	ST	—	STM 时钟输入
PA5/PWML	PA5	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。 (始终带有 30K poly 下拉电阻)
	PWML	PAS1	—	CMOS	互补 PWM 输出 (始终带有 30K poly 下拉电阻)
PA6/PWMH	PA6	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	PWMH	PAS1	—	CMOS	互补 PWM 输出
PA7/INT0/STP/ STCK/STPI	PA7	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	INT0	IFSO	ST	—	外部中断输入
	STP	PAS1	—	CMOS	STM 输出
	STCK	IFSO	ST	—	STM 时钟输入
	STPI	PAS1	ST	—	STM 捕捉输入
VDD	VDD	—	PWR	—	数字正电源供电
VSS	VSS	—	PWR	—	数字负电源供电
<b>OCDS EV</b>					
OCDSCK	OCDSCK	—	ST	—	片上调试系统时钟 (仅用于 OCDS EV)
OCDSDA	OCDSDA	—	ST	CMOS	片上调试系统数据 (仅用于 OCDS EV)

注： I/T： 输入类型

O/T： 输出类型

OPT： 通过寄存器选项来配置

PWR: 电源  
CMOS: CMOS 输出

ST: 施密特触发输入  
AN: 模拟信号

HT45F3430

引脚名称	功能	OPT	I/T	O/T	描述
PA0/AN0/OCDSDA /ICPDA	PA0	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	AN0	PAS0	AN	—	A/D 转换器输入通道 0
	OCDSDA	—	ST	CMOS	OCDS 数据 (仅用于 EV 芯片)
	ICPDA	—	ST	CMOS	ICP 数据
PA1/OCP_0/AN1	PA1	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	OCP_0	PAS0 IFSO	AN	—	OCP 输入
	AN1	PAS0	AN	—	A/D 转换器输入通道 1
PA2/STP/STPI_0/AN2 /OCDSCK/ICPCK	PA2	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	STP	PAS0	—	CMOS	STM 输出
	STPI_0	PAS0 IFSO	ST	—	STM 捕捉输入
	AN2	PAS0	AN	—	A/D 转换器输入通道 2
	OCDSCK	—	ST	—	OCDS 时钟 (仅用于 EV 芯片)
	ICPCK	—	ST	CMOS	ICP 时钟
PA3/OVP/AN3	PA3	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	OVP	PAS0 IFSO	AN	—	OVP 输入
	AN3	PAS0	AN	—	A/D 转换器输入通道 3
PA4/VBAT/AN4	PA4	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	VBAT	PAS1	AN	—	VBAT 输入
	AN4	PAS1	AN	—	A/D 转换器输入通道 4
PA5/INT0_0/STCK_0 /AN5	PA5	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	INT0_0	PAS1 IFSO	ST	—	外部中断输入
	STCK_0	PAS1 IFSO	ST	—	STM 时钟输入
	AN5	PAS1	AN	—	A/D 转换器输入通道 5

引脚名称	功能	OPT	I/T	O/T	描述
PA6/INT1_0/OCP_1/AN6	PA6	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	INT1_0	PAS1 IFS0	ST	—	外部中断输入
	OCP_1	PAS1 IFS0	AN	—	OCP 输入
	AN6	PAS1	AN	—	A/D 转换器输入通道 6
PA7/AN7/VREF/VREFI	PA7	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能。
	VREF	PAS1	AN	—	ADC 参考输入
	VREFI	PAS1	AN	—	PGA 输入，用于 A/D 转换器参考
	AN7	PAS1	AN	—	A/D 转换器输入通道 7
PB0/STP/STPI_1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	STP	PBS0	—	CMOS	STM 输出
	STPI_1	PBS0 IFS0	ST	—	STM 捕捉输入
PB1/STCK_1	PB1	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	STCK_1	IFS0	ST	—	STM 时钟输入
PB2/INT0_1	PB2	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	INT0_1	IFS0	ST	—	外部中断输入
PB3/INT1_1	PB3	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	INT1_1	IFS0	ST	—	外部中断输入
PB4	PB4	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
PB5	PB5	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
PB6	PB6	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
PB7	PB7	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
PC0/SCOM0	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	SCOM0	PCS0	—	AN	SCOM 输出
PC1/SCOM1	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	SCOM1	PCS0	—	AN	SCOM 输出

引脚名称	功能	OPT	I/T	O/T	描述
PC2/SCOM2	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	SCOM2	PCS0	—	AN	SCOM 输出
PC3/SCOM3	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	SCOM3	PCS0	—	AN	SCOM 输出
PC4/PWML	PC4	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。(始终带有 30K poly 下拉电阻)
	PWML	PCS0	—	CMOS	互补 PWM 输出 (始终带有 30K poly 下拉电阻)
PC5/PWMH	PC5	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻。
	PWMH	PCS0	—	CMOS	互补 PWM 输出
VDD	VDD	—	PWR	—	正电源供电
VSS	VSS	—	PWR	—	负电源供电

注: I/T: 输入类型

O/T: 输出类型

OPT: 通过寄存器选项来配置

ST: 施密特触发输入

PWR: 电源

AN: 模拟信号

CMOS: CMOS 输出

## 极限参数

电源供应电压	.....V <sub>SS</sub> -0.3V~V <sub>SS</sub> +6.0V
输入电压	.....V <sub>SS</sub> -0.3V~V <sub>DD</sub> +0.3V
储存温度	.....-50°C~125°C
工作温度	.....-40°C~85°C
I <sub>OL</sub> 总电流	.....80mA
I <sub>OH</sub> 总电流	.....-80mA
总功耗	.....500mW

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

## 直流电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
V <sub>DD</sub>	工作电压 (HIRC)	—	f <sub>SYS</sub> =f <sub>HIRC</sub> =8MHz	V <sub>LVR</sub>	—	5.5	V
		—	f <sub>SYS</sub> =f <sub>HIRC</sub> /2=4MHz	V <sub>LVR</sub>	—	5.5	V
		—	f <sub>SYS</sub> =f <sub>HIRC</sub> /4=2MHz	V <sub>LVR</sub>	—	5.5	V
		—	f <sub>SYS</sub> =f <sub>HIRC</sub> /8=1MHz	V <sub>LVR</sub>	—	5.5	V
I <sub>DD</sub>	工作电流 (HIRC)	3V	无负载, 所有外设除能, f <sub>SYS</sub> =f <sub>HIRC</sub> /2=4MHz	—	0.4	0.6	mA
		5V	—	0.8	1.2	mA	
		3V	无负载, 所有外设除能, f <sub>SYS</sub> =f <sub>HIRC</sub> =8MHz	—	0.8	1.2	mA
		5V	—	1.6	2.4	mA	
	工作电流 (LIRC)	3V	无负载, 所有外设除能, f <sub>SYS</sub> =f <sub>LIRC</sub> =32kHz	—	10	20	μA
		5V	—	30	50	μA	
I <sub>STB</sub>	待机电流 (SLEEP mode)	3V	无负载, 所有外设除能,	—	1.5	3	μA
		5V	WDT on	—	3	5	μA
	待机电流 (IDLE0 mode)	3V	无负载, 所有外设除能,	—	3	5	μA
		5V	f <sub>SUB</sub> on	—	5	10	μA
	待机电流 (IDLE1 mode, HIRC)	3V	无负载, 所有外设除能,	—	180	250	μA
		5V	f <sub>SUB</sub> on, f <sub>SYS</sub> =f <sub>HIRC</sub> /2=4MHz	—	400	600	μA
		3V	无负载, 所有外设除能,	—	360	500	μA
		5V	f <sub>SUB</sub> on, f <sub>SYS</sub> =f <sub>HIRC</sub> =8MHz	—	600	800	μA
V <sub>IL</sub>	I/O 口或输入引脚 低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	0.2V <sub>DD</sub>	V
V <sub>IH</sub>	I/O 口或输入引脚 高电平输入电压	5V	—	3.5	—	5.0	V
		—	—	0.8V <sub>DD</sub>	—	V <sub>DD</sub>	V
I <sub>OL</sub>	I/O 口灌电流 (PA5, PA6 除外) (HT45F3420)	3V	V <sub>OL</sub> =0.1V <sub>DD</sub>	18	36	—	mA
		5V		33	66	—	mA
	PA5, PA6 引脚灌电流 (HT45F3420)	3V	V <sub>OL</sub> =0.1V <sub>DD</sub>	16	32	—	mA
		5V		40	80	—	mA
	I/O 口灌电流 (PC4, PC5 除外) (HT45F3430)	3V	V <sub>OL</sub> =0.1V <sub>DD</sub>	16	32	—	mA
		5V		32	64	—	mA
	PC4, PC5 引脚灌电流 (HT45F3430)	3V	V <sub>OL</sub> =0.1V <sub>DD</sub>	16	32	—	mA
		5V		40	80	—	mA

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
I <sub>OH</sub>	I/O 口源电流 (PA5, PA6 除外) (HT45F3420)	3V	V <sub>OH</sub> =0.9V <sub>DD</sub>	-3	-6	—	mA
		5V		-7	-14	—	mA
	PA5, PA6 引脚源电流 (HT45F3420)	3V	V <sub>OH</sub> =0.9V <sub>DD</sub>	-18	-38	—	mA
		5V		-40	-80	—	mA
	I/O 口源电流 (PC4, PC5 除外) (HT45F3430)	3V	V <sub>OH</sub> =0.9V <sub>DD</sub> , SLEDCn[m+1, m]=00B (n=0,1..., m=0 or 2 or 4 or 6)	-1.0	-2.0	—	mA
		5V		-2.0	-4.0	—	mA
		3V	V <sub>OH</sub> =0.9V <sub>DD</sub> , SLEDCn[m+1, m]=01B (n=0,1..., m=0 or 2 or 4 or 6)	-1.75	-3.5	—	mA
		5V		-3.5	-7.0	—	mA
		3V	V <sub>OH</sub> =0.9V <sub>DD</sub> , SLEDCn[m+1, m]=10B (n=0,1..., m=0 or 2 or 4 or 6)	-2.5	-5.0	—	mA
		5V		-5.0	-10	—	mA
		3V	V <sub>OH</sub> =0.9V <sub>DD</sub> , SLEDCn[m+1, m]=11B (n=0,1..., m=0 or 2 or 4 or 6)	-5.5	-11	—	mA
		5V		-11	-22	—	mA
	PC4, PC5 引脚源电流 (HT45F3430)	3V	V <sub>OH</sub> =0.9V <sub>DD</sub>	-18	-38	—	mA
		5V	V <sub>OH</sub> =0.9V <sub>DD</sub>	-40	-80	—	mA
R <sub>PH</sub>	I/O 口上拉电阻	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
R <sub>PL</sub>	PA5 引脚下拉电阻 (HT45F3420)	3V	—	-50%	30	50%	kΩ
		5V	—	-50%	30	50%	kΩ
	PC4 引脚下拉电阻 (HT45F3430)	3V	—	-50%	30	50%	kΩ
		5V	—	-50%	30	50%	kΩ

## 交流电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
f <sub>SYS</sub>	系统时钟 (HIRC)	V <sub>LVR</sub> ~5.5V	f <sub>SYS</sub> =f <sub>HIRC</sub> =8MHz	—	8	—	MHz
	系统时钟 (LIRC)	V <sub>LVR</sub> ~5.5V	f <sub>SYS</sub> =f <sub>LIRC</sub> =32kHz	—	32	—	kHz
f <sub>HIRC</sub>	高速内部 RC 振荡 (HIRC)	5V	T <sub>a</sub> =25°C	-2%	8	+2%	MHz
		5V±0.5V	T <sub>a</sub> =0°C~70°C	-5%	8	+5%	MHz
		5V±0.5V	T <sub>a</sub> =-40°C~85°C	-7%	8	+7%	MHz
		2.2V~5.5V	T <sub>a</sub> =0~70°C	-7%	8	+7%	MHz
		2.2V~5.5V	T <sub>a</sub> =-40°C~85°C	-10%	8	+10%	MHz
f <sub>LIRC</sub>	低速内部 RC 振荡 (LIRC)	5V	T <sub>a</sub> =25°C	-10%	32	+10%	kHz
		5V±0.5V	T <sub>a</sub> =-40°C~85°C	-40%	32	+40%	kHz
		2.2V~5.5V	T <sub>a</sub> =-40°C~85°C	-50%	32	+60%	kHz
t <sub>TC</sub>	STCK 输入脉宽	—	—	0.3	—	—	μs
t <sub>INT</sub>	外部中断最小脉宽	—	—	10	—	—	μs
t <sub>RSTD</sub>	系统复位延迟时间 (POR 复位, LVR 硬件复位, WDT 软件复位)	—	—	25	50	100	ms
	系统复位延迟时间 (WDT 溢出硬件冷复位)	—	—	8.3	16.7	33.3	ms
t <sub>SST</sub>	系统启动时间 (从 HALT 唤醒, HALT 状态下 f <sub>SYS</sub> off)	—	f <sub>SYS</sub> =f <sub>H</sub> ~f <sub>H</sub> /64, f <sub>H</sub> =f <sub>HIRC</sub>	16	—	—	t <sub>HIRC</sub>
		—	f <sub>SYS</sub> =f <sub>SUB</sub> =f <sub>LIRC</sub>	2	—	—	t <sub>LIRC</sub>
	系统启动时间 (低速模式 ↔ 正常模式)	—	f <sub>HIRC</sub> off → on (HTO=1)	16	—	—	t <sub>HIRC</sub>
	系统启动时间 (从 HALT 唤醒, HALT 状态下 f <sub>SYS</sub> on)	—	f <sub>SYS</sub> =f <sub>H</sub> ~f <sub>H</sub> /64, f <sub>SYS</sub> =f <sub>HIRC</sub>	2	—	—	t <sub>H</sub>
		—	f <sub>SYS</sub> =f <sub>LIRC</sub>	2	—	—	t <sub>SUB</sub>
	系统启动时间 (WDT 溢出硬件冷复位)	—	—	0	—	—	t <sub>H</sub>
t <sub>EERD</sub>	EEPROM 读周期	—	—	—	2	4	t <sub>SYS</sub>
t <sub>EEWR</sub>	EEPROM 写周期	—	—	—	2	4	ms

## A/D 转换器电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
V <sub>DD</sub>	A/D 转换器工作电压	—	—	2.2	—	5.5	V
V <sub>ADI</sub>	A/D 转换器输入电压	—	—	0	—	V <sub>REF</sub>	V
V <sub>REF</sub>	A/D 转换器参考电压	—	—	2.2	—	V <sub>DD</sub>	V
DNL	非线性微分误差	3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =0.5μs	-3	—	+3	LSB
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =10μs	-3	—	+3	LSB
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =0.5μs	-4	—	+4	LSB
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =10μs	-4	—	+4	LSB
INL	非线性积分误差	3V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =0.5μs	-4	—	+4	LSB
		5V	SAINS[3:0]=0000B, SAVRS[1:0]=01B, V <sub>REF</sub> =V <sub>DD</sub> , t <sub>ADCK</sub> =10μs	-4	—	+4	LSB
		3V	无负载, t <sub>ADCK</sub> =0.5μs	—	0.2	0.4	mA
		5V	无负载, t <sub>ADCK</sub> =0.5μs	—	0.3	0.6	mA
t <sub>ADCK</sub>	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t <sub>ADC</sub>	A/D 转换时间 (包括采样和保持时间)	—	—	—	16	—	t <sub>ADCK</sub>
t <sub>ON2ST</sub>	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
I <sub>PGA</sub>	使能 PGA 增加的电流	3V	无负载	—	420	500	μA
		5V	无负载	—	460	550	μA
V <sub>CM</sub>	PGA 共模电压范围	3V	—	V <sub>SS</sub> -0.3	—	V <sub>DD</sub> -1.4	V
		5V	—	V <sub>SS</sub> -0.3	—	V <sub>DD</sub> -1.4	V
V <sub>OR</sub>	PGA 最大输出电压范围	3V	—	V <sub>SS</sub> +0.1	—	V <sub>DD</sub> -0.1	V
		5V	—	V <sub>SS</sub> +0.1	—	V <sub>DD</sub> -0.1	V
Ga	PGA 增益精度	3V	—	-5	—	+5	%
		5V	—				

## OCP 电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
I <sub>OCP</sub>	OCP 工作电流	3V	OCPEN [1:0]=01B DAC V <sub>REF</sub> =2.5V	—	260	350	μA
		5V	OCPEN [1:0]=01B DAC V <sub>REF</sub> =2.5V	—	350	420	μA
V <sub>OS_CMP</sub>	比较器输入失调电压	3V	无校准 (COF[4:0]=10000B)	-15	—	15	mV
		5V	无校准 (COF[4:0]=10000B)	-15	—	15	mV
		3V	校准后	-4	—	4	mV
		5V	校准后	-4	—	4	mV
V <sub>HYS</sub>	迟滞	3V	—	20	40	60	mV
		5V	—	20	40	60	mV
V <sub>CM_CMP</sub>	比较器共模电压范围	3V	—	V <sub>SS</sub>	—	V <sub>DD</sub> -1.4	V
		5V	—	V <sub>SS</sub>	—	V <sub>DD</sub> -1.4	V
V <sub>OS_OPA</sub>	OPA 输入失调电压	3V	无校准 (OOF[5:0]=100000B)	-15	—	15	mV
		5V	无校准 (OOF[5:0]=100000B)	-15	—	15	mV
		3V	校准后	-4	—	4	mV
		5V	校准后	-4	—	4	mV
V <sub>CM_OPA</sub>	OPA 共模电压范围	3V	—	V <sub>SS</sub> +0.2	—	V <sub>DD</sub> -1.4	V
		5V	—	V <sub>SS</sub> +0.2	—	V <sub>DD</sub> -1.4	V
Gain	OPA 增益误差	3V	全增益 (V <sub>CM_OPA</sub> >0.2V)	-5	—	5	%
		5V	全增益 (V <sub>CM_OPA</sub> >0.2V)	-5	—	5	%
DNL	非线性微分误差	3V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1	—	+1	LSB
		5V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1	—	+1	LSB
INL	非线性积分误差	3V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1.5	—	+1.5	LSB
		5V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1.5	—	+1.5	LSB

## OVP 电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
I <sub>OVP</sub>	OVP 工作电流	3V	OVPEN=1, DAC V <sub>REF</sub> =2.5V	—	45	54	μA
		5V	OVPEN=1, DAC V <sub>REF</sub> =2.5V	—	56	72	μA
V <sub>os</sub>	输入失调电压	3V	校准后	-4	—	4	mV
		5V	校准后	-4	—	4	mV
V <sub>HYS</sub>	迟滞	5V	—	20	40	60	mV
V <sub>CM</sub>	共模电压范围	3V	—	V <sub>SS</sub>	—	V <sub>DD</sub> - 1.4	V
		5V	—	V <sub>SS</sub>	—	V <sub>DD</sub> - 1.4	V
DNL	非线性微分误差	3V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1	—	+1	LSB
		5V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1	—	+1	LSB
INL	非线性积分误差	3V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1.5	—	+1.5	LSB
		5V	DAC V <sub>REF</sub> =V <sub>DD</sub>	-1.5	—	+1.5	LSB

## DLL 电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
I <sub>DLL</sub>	工作电流	3V	DLLEN=1	—	0.9	1.2	mA
		5V	DLLEN=1	—	1.5	2	mA
f <sub>DLL</sub>	工作频率	2.2V~5.5V	f <sub>HIRC</sub> =8MHz	-10%	8	+10%	MHz

## LVR 电气特性

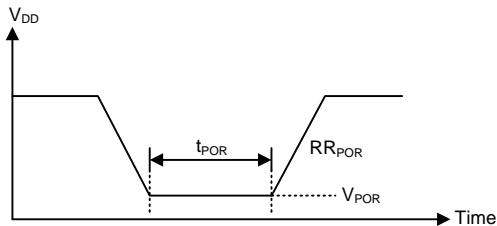
T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
V <sub>DD</sub>	工作电压	—	—	1.9	—	5.5	V
V <sub>LVR</sub>	低压复位电压	—	LVR 使能, 2.1V	-5%	2.1	+5%	V
V <sub>BG</sub>	Bandgap 参考电压	—	—	-5%	1.04	+5%	V
t <sub>LVR</sub>	最小低电压复位时间	—	—	120	240	480	μs

## 上电复位电气特性

T<sub>a</sub>=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V <sub>DD</sub>	条件				
V <sub>POR</sub>	上电复位电压	—	—	—	—	100	mV
R <sub>R POR</sub>	上电复位电压速率	—	—	0.035	—	—	V/ms
t <sub>POR</sub>	V <sub>DD</sub> 保持为 V <sub>POR</sub> 的最长时间	—	—	1	—	—	ms

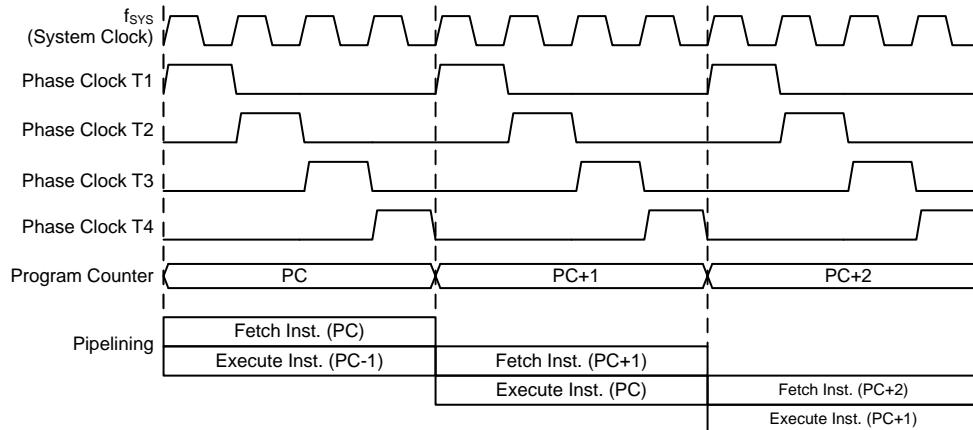


## 系统结构

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需要多一个指令周期外，大部分指令能在一个指令周期内完成。8-bit 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和 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，且可以直或接或间接寻址。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 I/O 和 A/D 控制系统时，仅需要少数的外部器件。这些使得此系列单片机适用于低成本和批量生产的控制应用。

## 时序和流水线结构

主系统时钟由 HIRC 或 LIRC 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 时钟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。



系统时序与流水线

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]	Fetch Inst. 1	Execute Inst. 1			
2 CALL DELAY		Fetch Inst. 2	Execute Inst. 2		
3 CPL [12H]			Fetch Inst. 3	Flush Pipeline	
4 :				Fetch Inst. 6	Execute Inst. 6
5 :					Fetch Inst. 7
6 DELAY: NOP					

指令捕捉

## 程序计数器

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。只有较低的 8 位，即所谓的程序计数器低字节寄存器 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。

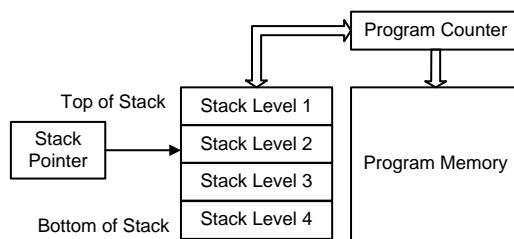
单片机型号	程序计数器	
	程序计数器高字节	PCL 寄存器
HT45F3420	PC9~PC8	PCL7~PCL0
HT45F3430	PC10~PC8	PCL7~PCL0

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。注意，当这样一个程序跳转要执行时，会插入一个空指令周期。PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

## 堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。

如果堆栈已满，且有非屏蔽的中断发生，中断请求标志会被置位，但中断响应将被禁止。当堆栈指针减少（执行 RET 或 RETI），中断将被响应。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 指令仍然可以被执行，而造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这可能导致不可预期的程序分支指令执行错误。若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



## 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操作，并将结果存储在指定的寄存器，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

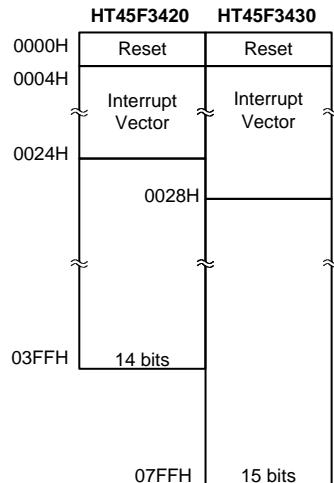
## Flash 程序存储器

程序存储器用来存放用户代码即储存程序。程序存储器为 Flash 类型意味着可以多次重复编程，方便用户使用同一芯片进行程序的修改。使用适当的单片机编程工具，此系列单片机提供用户灵活便利的调试方法和项目开发规划及更新。

### 结构

程序存储器的容量为  $1K \times 14 \sim 2K \times 15$ 。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。

单片机型号	容量
HT45F3420	$1K \times 14$
HT45F3430	$2K \times 15$



程序存储器结构

### 特殊向量

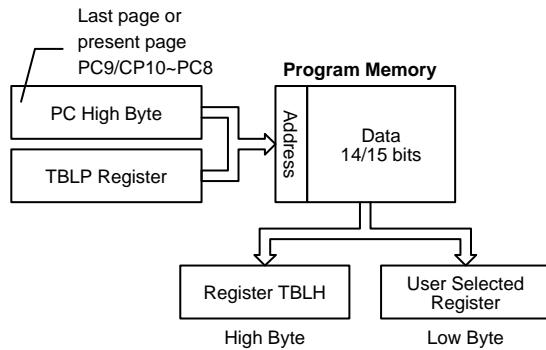
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 0000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

### 查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的地址放在表格指针寄存器 TBLP 中。这个寄存器定义表格总的地址。

在设置完表格指针后，表格数据可以使用“TABRDC [m]”或“TABRDL [m]”指令分别从程序存储器查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到使用者在指令中所指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊寄存器。所发送的高字节中未用到的位读为“0”。

下图为查表中寻址 / 数据流程：



## 查表范例

以下范例说明如何定义并从单片机中获取表格指针和表格数据。这个例子使用的是利用“ORG”指令储存在程序存储器中的一个原始数据表。ORG 指令的值“700H”位于 HT45F3430 单片机中 2K 程序存储器内最后一页的起始地址。表格指针设置的初始值为“06H”，以确保从数据表格读取的第一笔数据位于程序存储器地址“706H”，或是最后一页起始地址后的第六个地址。值得注意的是，若“TABRDC[m]”指令被使用，则表格指针指向 TBLP 寄存器所指定的起始地址。在这个例子中，表格数据的高字节等于零，而当“TABRDC[m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

由于 TBLH 寄存器为只读寄存器，无法重新储存，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，另外要注意的是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 low table pointer - note that this address
; is referenced
mov tblp,a ; to the last page or the present page
:
:
tabrd1 tempreg1 ; transfers value in table referenced by table pointer
; data at program memory address "706H" transferred to
; tempreg1 and TBLH
dec tblp ; reduce value of table pointer by one
tabrd1 tempreg2 ; transfers value in table referenced by table pointer
; data at program memory address "705H" transferred to
; tempreg2 and TBLH.
; In this example the data "1AH" is transferred to
; tempreg1 and data "0FH" to register tempreg2
:
:
org 700h ; sets initial address of program memory
dc 00Ah,00Bh,00Ch,00Dh,00Eh,00Fh,01Ah,01Bh

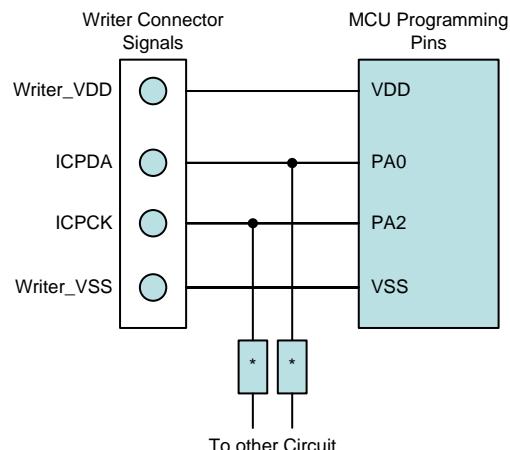
```

## 在线烧录 – ICP

Flash 型程序存储器的提供使得用户可以方便简单地在同一芯片上进行程序的更新和修改。另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧写，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

Holtek 烧录器引脚	MCU 在线烧录引脚	功能
ICPDA	PA0	烧写串行数据 / 地址
ICPCK	PA2	烧写时钟
VDD	VDD	电源
VSS	VSS	地

芯片内部程序存储器和 EEPROM 存储器都可以通过 4 线的接口在线进行烧录。其中一个单独的引脚用于数据串行下载或上传、一条用于时钟、另外两条用于提供电源。芯片在线烧录的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。



注：\* 可能为电阻或电容。若为电阻则其值必须大于  $1k\Omega$ 。若为电容则其值必须小于  $1nF$ 。

## 片上调试 – OCDS

Holtek 提供 EV 芯片 HT45V3420 和 HT45V3430 用于 HT45F3420 和 HT45F3430 单片机的仿真。这些 EV 芯片提供片上调试功能 (OCDS) 用于开发过程中单片机的调试。除了片上调试功能，EV 芯片和实际 MCU 在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，用 EV 芯片来仿真实际单片机芯片的性能。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV 芯片进行调试时，OCDSDA 和 OCDSCK 引脚上的其它共用功能对 EV 芯片无效。这两个 OCDS 引脚与 ICP 引脚共用，因此在线烧录时仍用作 Flash 存储器烧录引脚。关于 OCDS 功能的详细描述，参考“Holtek e-Link for 8-bit MCU OCDS User's Guide”文档。

Holtek e-Link 引脚	EV 芯片引脚	引脚描述
OCDSDA	OCDSDA	片上调试数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
GND	VSS	地

## RAM 数据存储器

数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

### 结构

数据存储器分为两部分，第一部分是特殊功能数据存储器，这些寄存器有固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，但有些被加以保护而不对用户开放。第二部分是通用数据存储器，该存储器内所有地址都可在程序的控制下进行读取和写入。

整体数据存储器被分为 2 个 Bank。除了在 40H 的 EEC 寄存器只能在 Bank1 中被寻址外，特殊功能数据寄存器在所有 Bank 中可被访问。可通过设置存储区指针的值实现不同数据存储器 Bank 之间的切换。此系列单片机数据存储器的起始地址为 00H。

单片机型号	RAM 数据存储器	地址
HT45F3420	特殊功能数据存储器	Bank0: 00H~3FH Bank1: 00H~40H (EEC 在 40H)
	通用数据存储器: 64×8	Bank0: 40H~7FH
HT45F3430	特殊功能数据存储器	Bank0: 00H~7FH Bank1: 00H~7FH (EEC 在 40H)
	通用数据存储器: 128×8	Bank0: 80H~FFH

### 通用数据存储器

所有的单片机程序需要一个读 / 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使用者进行读取和写入的操作。在程序控制下使用位操作指令可对个别的位做置位或复位的操作，极大地方便了用户在数据存储器内进行位操作。

### 特殊功能数据存储器

这个区域的数据存储器是存放特殊寄存器的，这些寄存器与单片机的正确操作密切相关。大多数的寄存器可进行读取和写入，只有一些是被写保护而只能读取的，相关细节的介绍请参看有关特殊功能寄存器的部分。要注意的是，任何读取指令对存储器中未定义的地址进行读取将返回 “00H”。

	Bank0	Bank1	Bank0	Bank1
00H	IAR0		20H	SADC0
01H	MP0		21H	SADC1
02H	IAR1		22H	SADC2
03H	MP1		23H	PAS0
04H	BP		24H	PAS1
05H	ACC		25H	STMC0
06H	PCL		26H	STMC1
07H	TBLP		27H	STMDL
08H	TBLH		28H	STMDH
09H	Unused		29H	STMAL
0AH	STATUS		2AH	STMAH
0BH	SMOD		2BH	Unused
0CH	Unused		2FH	Unused
0DH	INTEG		30H	PWMP
0EH	INTC0		31H	PWMD
0FH	INTC1		32H	DLL
10H	INTC2		33H	CPR
11H	MFI		34H	Unused
12H	Unused		35H	CPOR
13H	Unused		36H	Unused
14H	PA		37H	Unused
15H	PAC		38H	OCP0
16H	PAPU		39H	OCP1
17H	PAWU		3AH	OCPDA
18H	IFS0		3BH	OCPICAL
19H	WDTC		3CH	OCPCCAL
1AH	TBC		3DH	OVPC0
1BH	SMOD1		3EH	OVPC1
1CH	EEA		3FH	OVPDA
1DH	EED		40H	Unused EEC
1EH	SADOL			
1FH	SADOH			

 : Unused, read as "00"

### 特殊功能数据存储器结构 – HT45F3420

Bank 0~1		Bank 0~1		Bank 0		Bank 1	
00H	IAR0	20H	SADC0	40H	Unused	EEC	
01H	MP0	21H	SADC1	41H	PB		
02H	IAR1	22H	SADC2	42H	PBC		
03H	MP1	23H	PAS0	43H	PBPU		
04H	BP	24H	PAS1	44H	PC		
05H	ACC	25H	STMC0	45H	PCC		
06H	PCL	26H	STMC1	46H	PCPU		
07H	TBLP	27H	STMDL	47H	PBS0		
08H	TBLH	28H	STMDH	48H	PCS0		
09H	Unused	29H	STMAL	49H	Unused		
0AH	STATUS	2AH	STMAH	4AH	ASCR		
0BH	SMOD	2BH	Unused		4BH	SCOMC	
0CH	Unused	2CH	Unused		4CH	SLEDC0	
0DH	INTEG	2DH	Unused		4DH	SLEDC1	
0EH	INTC0	2EH	Unused		4EH	Unused	
0FH	INTC1	2FH	Unused		7FH	Unused	
10H	INTC2	30H	PWMP				
11H	MFI	31H	PWMD				
12H	Unused	32H	DLL				
13H		33H	CPR				
14H	PA	34H	Unused				
15H	PAC	35H	CPOR				
16H	PAPU	36H	Unused				
17H	PAWU	37H					
18H	IFSO	38H	OCPC0				
19H	WDTC	39H	OCPC1				
1AH	TBC	3AH	OCPDA				
1BH	SMOD1	3BH	OCPOCAL				
1CH	EEA	3CH	OCPCCAL				
1DH	EED	3DH	OVPC0				
1EH	SADOL	3EH	OVPC1				
1FH	SADOH	3FH	OVPDA				

□ : Unused, read as 00H

### 特殊功能数据存储器结构 – HT45F3430

## 特殊功能寄存器描述

大部分特殊功能寄存器的细节将在相关功能章节描述，但有几个寄存器需在此章节单独描述。

### 间接寻址寄存器 – IAR0, IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储区，但其并没有实际的物理地址。间接寻址的方法准许使用间接寻址寄存器和存储器指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。在间接寻址寄存器 IAR0 和 IAR1 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。它们总是成对出现，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问任何 Bank。因为这些间接寻址寄存器不是实际存在的，直接读取将返回 “00H”的结果，而直接写入此寄存器则不做任何操作。

### 存储器指针 – MP0, MP1

该系列单片机提供两个存储器指针，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般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由相关的存储器指针所指定的地址。MP0 和间接寻址寄存器 IAR0 用于访问 Bank 0，而 MP1 和 IAR1 根据 BP 寄存器可以访问所有的 Bank。直接寻址只能在 Bank 0 中使用，使用 MP1 和 IAR1 可间接访问所有的数据 Bank。

以下例子说明如何清除一个具有 4 个 RAM 地址的内容，它们已事先定义成地址 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:
```

需注意，在例中并没有确定 RAM 地址。

## 存储区指针 – BP

该系列单片机中，数据存储器被分为 2 个 bank，即 Bank 0~Bank 1。可以通过设置存储区指针 (Bank Pointer) 值来访问不同的数据存储区。BP 指针的第 0 位用于选择数据存储器的 Bank 0 或 Bank 1。

复位后，数据存储器会初始化到 Bank 0，但是在暂停模式下的 WDT 溢出复位，不会改变数据存储器的存储区号。应该注意的是特殊功能数据存储器不受存储区的影响，也就是说，不论是在哪一个存储区，都能对特殊功能寄存器进行读写操作。数据存储器的直接寻址总是访问 Bank 0，不影响存储区指针的值。要访问 Bank1，则必须要使用间接寻址方式。

### BP 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	DMBPO
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 未定义，读为“0”

Bit 0 **DMBPO**: 数据存储器 Bank 选择位

0: Bank 0

1: Bank 1

## 累加器 – ACC

对任何单片机来说，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 ALU 得到的运算结果都会暂时存在 ACC 累加器里。若没有累加器，ALU 必须在每次进行如加法、减法和移位的运算时，将结果写入到数据存储器，这样会造成程序编写和时间的负担。另外数据传送也常常牵涉到累加器的临时储存功能，例如在使用者定义的一个寄存器和另一个寄存器之间传送数据时，由于两寄存器之间不能直接传送数据，因此必须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，程序计数器低字节设置在数据存储器的特殊功能区域内，程序员可对此寄存器进行操作，很容易的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程序存储器的某一地址，然而由于寄存器只有 8 位长度，因此只允许在本页的程序存储器范围内进行跳转，而当使用这种运算时，要注意会插入一个空指令周期。

## 查表寄存器 – TBLP, TBLH

这两个特殊功能寄存器对存储在程序存储器中的表格进行操作。TBLP 为表格指针，指向表格数据存储的地址。它们的值必须在任何表格读取指令执行前加以设定，由于它们的值可以被如“INC”或“DEC”的指令所改变，这就提供了一种简单的方法对表格数据进行读取。表格读取数据指令执行之后，表格数据高字节存储在 TBLH 中。其中要注意的是，表格数据低字节会被传送到使用者指定的地址。

## 状态寄存器 – STATUS

该 8-bit 寄存器由零标志位 (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。这些算术 / 逻辑操作和系统运行标志位是用来记录单片机的运行状态。

除了 TO 和 PDF 标志外，状态寄存器中的位像其它大部分寄存器一样可以被改变。任何数据写入到状态寄存器将不会改变 TO 或 PDF 标志位。另外，执行不同的指令后，与状态寄存器有关的运算可能会得到不同的结果。TO 标志位只会受系统上电、看门狗溢出或执行“CLR WDT”或“HALT”指令影响。PDF 标志位只会受执行“HALT”或“CLR WDT”指令或系统上电影响。

Z、OV、AC 和 C 标志位通常反映最新运算的状态。

- C: 当加法运算的结果产生进位，或减法运算的结果没有产生借位时，C 被置位，否则 C 被清零，同时 C 也会被带进位的移位指令所影响。
- AC: 当低半字节加法运算的结果产生进位，或低半字节减法运算的结果没有产生借位时，AC 被置位，否则 AC 被清零。
- Z: 当算术或逻辑运算结果是零时，Z 被置位，否则 Z 被清零。
- OV: 当运算结果高两位的进位状态异或结果为 1 时，OV 被置位，否则 OV 被清零。
- PDF: 系统上电或执行“CLR WDT”指令会清零 PDF，而执行“HALT”指令则会置位 PDF。
- TO: 系统上电或执行“CLR WDT”或“HALT”指令会清零 TO，而当 WDT 溢出则会置位 TO。

另外，当进入一个中断程序或执行子程序调用时，状态寄存器不会自动压入到堆栈保存。假如状态寄存器的内容是重要的且子程序可能改变状态寄存器的话，则需谨慎的去做正确的储存。

### STATUS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	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” 未知

Bit 7~6 未定义，读为“0”

Bit 5 **TO:** 看门狗溢出标志位

0: 系统上电或执行“CLR WDT”或“HALT”指令后  
1: 看门狗溢出发生

Bit 4 **PDF:** 暂停标志位

0: 系统上电或执行“CLR WDT”指令后  
1: 执行“HALT”指令

Bit 3 **OV:** 溢出标志位

0: 无溢出  
1: 运算结果高两位的进位状态异或结果为1

Bit 2 **Z:** 零标志位

0: 算术或逻辑运算结果不为0  
1: 算术或逻辑运算结果为0

Bit 1 **AC:** 辅助进位标志位

0: 无辅助进位  
1: 在加法运算中低四位产生了向高四位进位，或减法运算中低四位不发生从高四位借位

Bit 0 **C:** 进位标志位

0: 无进位  
1: 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发生借位  
C 也受带进位的移位指令的影响。

## EEPROM 数据存储器

该系列单片机的一个特性是内建 EEPROM 数据存储器。“Electrically Erasable Programmable Read Only Memory”为电可擦可编程只读存储器，由于其非易失的存储结构，即使在电源掉电的情况下存储器内的数据仍然保存完好。这种存储区扩展了 ROM 空间，对设计者来说增加了许多新的应用机会。EEPROM 可以用来存储产品编号、校准值、用户特定数据、系统配置参数或其它产品信息等。EEPROM 的数据读取和写入过程也会变的更简单。

### EEPROM 数据存储器结构

该系列单片机的 EEPROM 数据存储器容量为  $32 \times 8 \sim 64 \times 8$ 。由于映射方式与程序存储器和数据存储器不同，该存储器不能像其它类型的存储器一样寻址。使用 Bank0 中的地址和数据寄存器以及 Bank1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

单片机型号	容量
HT45F3420	$32 \times 8$
HT45F3430	$64 \times 8$

### EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储器总的操作，即地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 寄存器位于 Bank 0 和 Bank1 中，它们能像其它特殊功能寄存器一样直接被访问。而 EEC 位于 Bank1 中，只能通过存储器指针 MP1 和间接寻址寄存器 IAR1 进行间接读取或写入。由于 EEC 控制寄存器位于 Bank 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，存储器指针 MP1 必须先设为“40H”，存储区指针寄存器 BP 则设置为“01H”。

寄存器名称	位							
	7	6	5	4	3	2	1	0
EEA (HT45F3420)	—	—	—	D4	D3	D2	D1	D0
EEA (HT45F3430)	—	—	D5	D4	D3	D2	D1	D0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM 控制寄存器列表

### EEA 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	D4	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 未定义，读为“0”

Bit 4~0 数据 EEPROM 地址

数据 EEPROM 地址 bit 4~bit 0

### EEA 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	D5	D4	D3	D2	D1	D0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义, 读为“0”

Bit 5~0 数据 EEPROM 地址

数据 EEPROM 地址 bit 5~bit 0

### EED 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0 数据 EEPROM 数据

数据 EEPROM 数据 bit 7~bit 0

### EEC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WREN	WR	RDEN	RD
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 **WREN:** 数据 EEPROM 写使能位

0: 除能

1: 使能

此位为数据 EEPROM 写使能位, 向数据 EEPROM 写操作之前需将此位置高。将此位清零时, 则禁止向数据 EEPROM 写操作。

Bit 2 **WR:** EEPROM 写控制位

0: 写周期结束

1: 激活写周期

此位为数据 EEPROM 写控制位, 由应用程序将此位置高将激活写周期。写周期结束后, 硬件自动将此位清零。当 WREN 未先置高时, 此位置高无效。

Bit 1 **RDEN:** 数据 EEPROM 读使能位

0: 除能

1: 使能

此位为数据 EEPROM 读使能位, 向数据 EEPROM 读操作之前需将此位置高。将此位清零时, 则禁止向数据 EEPROM 读操作。

Bit 0 **RD:** EEPROM 读控制位

0: 读周期结束

1: 激活读周期

此位为数据 EEPROM 读控制位, 由应用程序将此位置高将激活读周期。读周期结束后, 硬件自动将此位清零。当 RDEN 未首先置高时, 此位置高无效。

注: 在同一条指令中 WREN、WR、RDEN 和 RD 不能同时置为“1”。WR 和 RD 不能同时置为“1”。

## 从 EEPROM 中读取数据

从 EEPROM 中读取数据，EEC 寄存器中的读使能位 RDEN 先置为高以使能读功能。EEPROM 中读取数据的地址要先放入 EEA 寄存器中。若 EEC 寄存器中的 RD 位被置高，一个读周期将开始。若 RDEN 位未被设置而 RD 位置高则不能开始读操作。若读周期结束，RD 位将自动清除为“0”，数据可以从 EED 寄存器中读取。数据在其它读或写操作执行前将一直保留在 EED 寄存器中。应用程序将轮询 RD 位以确定数据可以有效地被读取。

## 写数据到 EEPROM 中

EEPROM 中写入数据的地址要先放入 EEA 寄存器中，写入的数据需存入 EED 寄存器中。EEC 寄存器中的写使能位 WREN 先置为高以使能写功能，然后 EEC 寄存器中的 WR 位需立即置高以开始写操作，这两条指令必须连续执行。总中断位 EMI 在写周期开始前应当被清零，写周期开始后再将其使能。需注意若 WR 位已置为高而 WREN 位还未被设置则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 写中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。从而，应用程序将轮询 WR 位以确定写周期是否结束。

## 写保护

防止误写入的写保护有以下几种。单片机上电后控制寄存器中的写使能位将被清除以杜绝任何写入操作。上电后存储区指针 BP 将重置为“0”，这意味着数据存储器 Bank 0 被选中。由于 EEPROM 控制寄存器位于 Bank 1 中，这增加了对写操作的保护措施。在正常程序操作中确保控制寄存器中的写使能位被清除将能防止不正确的写操作。

## EEPROM 中断

EEPROM 写周期结束后将产生 EEPROM 写中断，需先通过设置相关中断寄存器的 DEE 位使能 EEPROM 中断。当 EEPROM 写周期结束，DEF 请求标志位将被置位。若总中断，EEPROM 中断使能且堆栈未满，将跳转到相应的中断向量中执行。当 EEPROM 中断被响应，EEPROM 中断标志 DEF 将自动复位。总中断 EMI 位也将自动清零以除能其他中断。

## 编程注意事项

必须注意的是数据不会无意写入 EEPROM。在没有写动作时写使能位被正常清零可以达到定期保护功能。存储区指针 BP 也可以正常清零以阻止进入 EEPROM 控制寄存器存在的 Bank 1。尽管没有必要，写一个简单的读回程序以检查新写入的数据是否正确还是应该考虑的。WREN 位置位后，EEC 寄存器中的 WR 位需立即置位，以确保写周期正确地执行。写周期执行前总中断位 EMI 应先清零，写周期开始执行后再将此位重新使能。注意，单片机不应在 EEPROM 读或写操作完全完成之前进入空闲或休眠模式，否则 EEPROM 读或写操作将失败。

## 程序范例

### 从 EEPROM 中读取数据 – 轮询法

```
MOV A, EEPROM_ADRES           ; user defined address
MOV EEA, A
MOV A, 040H                   ; setup memory pointer MP1
MOV MP1, A                     ; MP1 points to EEC register
MOV A, 01H                     ; setup Bank Pointer
MOV BP, A
SET IAR1.1                     ; set RDEN bit, enable read operations
SET IAR1.0                     ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0                     ; check for read cycle end
JMP BACK
CLR IAR1                        ; disable EEPROM write
CLR BP
MOV A, EED                      ; move read data to register
MOV READ_DATA, A
```

### 写数据到 EEPROM – 轮询法

```
MOV A, EEPROM_ADRES           ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA             ; user defined data
MOV EED, A
MOV A, 040H                   ; setup memory pointer MP1
MOV MP1, A                     ; MP1 points to EEC register
MOV A, 01H                     ; setup Bank Pointer
MOV BP, A                      ; BP points to data memory bank 1
CLR EMI
SET IAR1.3                     ; set WREN bit, enable write operations
SET IAR1.2                     ; start Write Cycle - set WR bit -
                                ; executed immediately after set WREN bit
SET EMI
BACK:
SZ IAR1.2                     ; check for write cycle end
JMP BACK
CLR IAR1                        ; disable EEPROM write
CLR BP
```

## 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方面可以达到最优化。振荡器选择是通过控制寄存器完成的。

### 振荡器概述

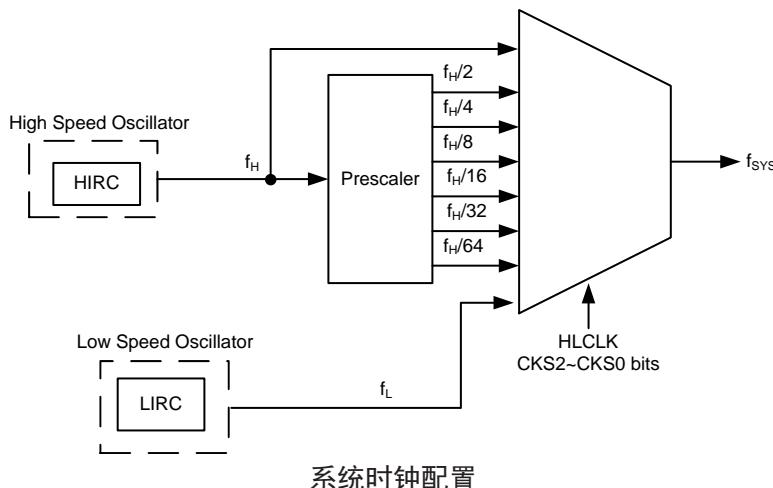
振荡器除了作为系统时钟源，还作为看门狗定时器和时基中断的时钟源。高度集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

类型	名称	频率
内部高速 RC	HIRC	8MHz
内部低速 RC	LIRC	32kHz

振荡器类型

### 系统时钟配置

该系列单片机有两个系统振荡器，即一个高速振荡器和一个低速振荡器。高速振荡器为内部 8MHz RC 振荡器。低速振荡器为内部 32kHz RC 振荡器。使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位和 CKS2~CKS0 位段决定的，系统时钟可动态选择。



系统时钟配置

### 高速内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个高度集成的系统振荡器，无需其它外部元器件。该内部 RC 振荡器的固定频率为 8MHz 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  $V_{DD}$ 、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 5V 且温度为 25°C 时，HIRC 的固定振荡频率的容差为 2%。

## 内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个低频振荡器。这是一个完全集成的 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元器件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响减至最低。

## 辅助振荡器

低速振荡器除了提供一个系统时钟源外，也用来为其它单片机功能提供时钟源，如看门狗定时器和时基中断等功能。

## 工作模式与系统时钟

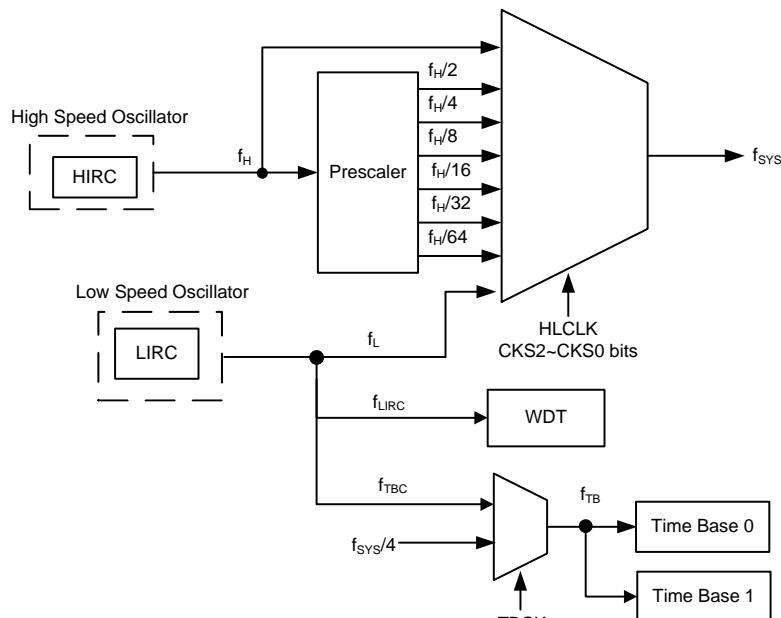
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。盛群单片机提供了高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得最佳性能 / 功耗比。

## 系统时钟

该系列单片机为 CPU 和外围功能操作提供了两种不同的时钟源。用户使用寄存器编程可获取多种时钟，进而使系统时钟获取最大的应用性能。

主系统时钟可来自高频时钟源  $f_H$  或低频时钟源  $f_L$ ，通过 SMOD 寄存器中的 HLCLK 位和 CKS2~CKS0 位段进行选择。高频系统时钟来自 HIRC 振荡器，低频系统时钟源来自 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频  $f_H/2 \sim f_H/64$ 。

一个额外的内部时钟可用于外围电路，即  $f_{TBC}$ 。 $f_{TBC}$  源自 LIRC 振荡器。该时钟可作为时基中断功能和 STM 的时钟源。



单片机时钟配置

注：当系统时钟源  $f_{SYS}$  由  $f_H$  切换到  $f_L$ ，高速振荡器将停止以节省耗电。因此没有  $f_H \sim f_H/64$  的时钟可供外部电路使用。

## 系统工作模式

单片机有 5 种不同的工作模式，每种有它自身的特性，根据应用中不同的性能和功耗要求可选择不同的工作模式。单片机正常工作有两种模式：正常模式和低速模式。剩余的 3 种工作模式：休眠模式、空闲模式 0 和空闲模式 1 用于单片机 CPU 关闭时以节省耗电。

工作模式	说明			
	CPU	f <sub>SYS</sub>	f <sub>LIRC</sub>	f <sub>TBC</sub>
正常模式	On	f <sub>H</sub> ~f <sub>H</sub> /64	On	On
低速模式	On	f <sub>L</sub>	On	On
空闲模式 0	Off	Off	On	On
空闲模式 1	Off	On	On	On
休眠模式	Off	Off	On	Off

### 正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。该模式下单片机正常工作的时钟源来自 HIRC 振荡器。高速振荡器频率可被分为 1~64 的不等比率，实际的比率由 SMOD 寄存器中的 HLCLK 位和 CKS2~CKS0 位段进行选择。单片机使用高速振荡器分频作为系统时钟可减少工作电流。

### 低速模式

此模式的系统时钟虽为较低速时钟源，但单片机仍能正常工作。该低速时钟源可来自低速振荡器 LIRC。单片机在此模式中运行所耗工作电流较低。在低速模式中，f<sub>H</sub> 关闭。

### 休眠模式

在 HALT 指令执行后且 SMOD 寄存器中的 IDLEN 位为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行。但 f<sub>LIRC</sub> 时钟将继续运行且看门狗定时器功能使能。

### 空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中的 IDLEN 位为高，SMOD1 寄存器中的 FSYS0 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，系统振荡器停止，但看门狗定时器和定时器模块等一些外围功能将保持运行。

### 空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中的 IDLEN 位为高，SMOD1 寄存器中的 FSYS0 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但高速或低速振荡器将继续运行以驱动如看门狗定时器和定时器模块等一些外围功能，看门狗定时器时钟 f<sub>LIRC</sub> 继续运行。

## 控制寄存器

SMOD 和 SMOD1 寄存器用于控制单片机的内部时钟。

### SMOD 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	LTO	HTO	IDLEN	HLCLK
R/W	R/W	R/W	R/W	—	R	R	R/W	R/W
POR	0	0	0	—	0	0	1	1

Bit 7~5      **CKS2~CKS0:** HLCLK 为 “0” 时系统时钟选择

- 000:  $f_L(f_{LIRC})$
- 001:  $f_L(f_{LIRC})$
- 010:  $f_H/64$
- 011:  $f_H/32$
- 100:  $f_H/16$
- 101:  $f_H/8$
- 110:  $f_H/4$
- 111:  $f_H/2$

这三位用于选择系统时钟源。除了 LIRC 可作为系统时钟源外，高速系统振荡器的分频比也可作为系统时钟源。

Bit 4      未定义，读为 “0”

Bit 3      **LTO:** 低速振荡器就绪标志位

- 0: 未就绪
- 1: 就绪

此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或唤醒后何时稳定下来。

Bit 2      **HTO:** 高速振荡器就绪标志位

- 0: 未就绪
- 1: 就绪

此位为高速系统振荡器就绪标志位，用于表明唤醒后高速系统振荡器何时稳定下来。此标志在系统上电后经硬件清零，高速系统振荡器稳定后变为高电平。因此，此位在单片机上电后由应用程序读取的值为 “1”。

Bit 1      **IDLEN:** 空闲模式控制位

- 0: 除能
- 1: 使能

此位为空闲模式控制位，用于决定 HALT 指令执行后发生动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYS0 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYS0 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。

Bit 0      **HLCLK:** 系统时钟选择位

- 0:  $f_H/2-f_H/64$  或  $f_L$
- 1:  $f_H$

此位用于选择  $f_H$  或  $f_H/2-f_H/64$  或  $f_L$  作为系统时钟。该位为高时选择  $f_H$  作为系统时钟，为低时则选择  $f_H/2-f_H/64$  或  $f_L$  作为系统时钟。当系统时钟由  $f_H$  时钟向  $f_L$  时钟转换时， $f_H$  将自动关闭以降低功耗。

### SMOD1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	—	WRF
R/W	R/W	—	—	—	—	R/W	—	R/W
POR	0	—	—	—	—	x	—	0

“x”未知

Bit 7      **FSYSON:** 空闲模式下 fsys 控制位

- 0: 除能
- 1: 使能

Bit 6~3    未定义, 读为 “0”

Bit 2       **LVRF:** LVR 复位标志位

详见其它章节

Bit 1       未定义, 读为 “0”

Bit 0       **WRF:** WDTC 控制寄存器软件复位标志位

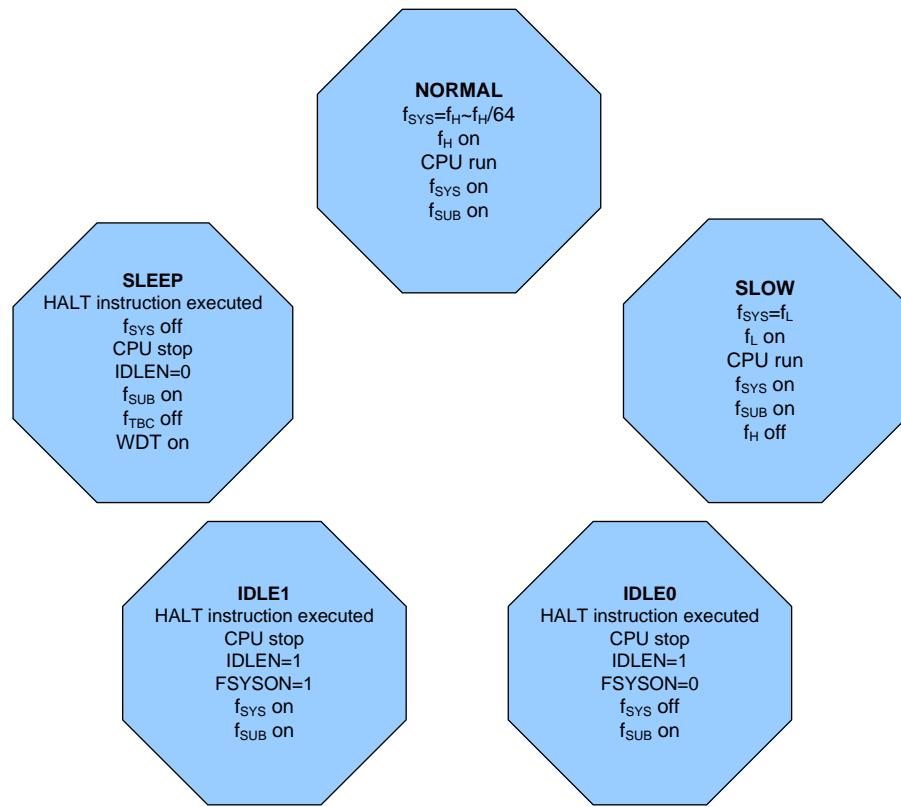
详见其它章节

### 工作模式切换

该系列单片机可在各个工作模式间自由切换, 使得用户可根据所需选择最佳的性能 / 功耗比。用此方式, 对单片机工作的性能要求不高的情况下, 可使用较低频时钟以减少工作电流, 在便携式应用上延长电池的使用寿命。

简单来说, 正常模式和低速模式间的切换仅需设置 SMOD 中的 HLCLK 位及 CKS2~CKS0 位段即可实现, 而正常模式 / 低速模式与休眠模式 / 空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后, 单片机是否进入空闲模式或休眠模式由 SMOD 寄存器中的 IDLEN 位和 SMOD1 寄存器中的 FSYSON 位决定的。

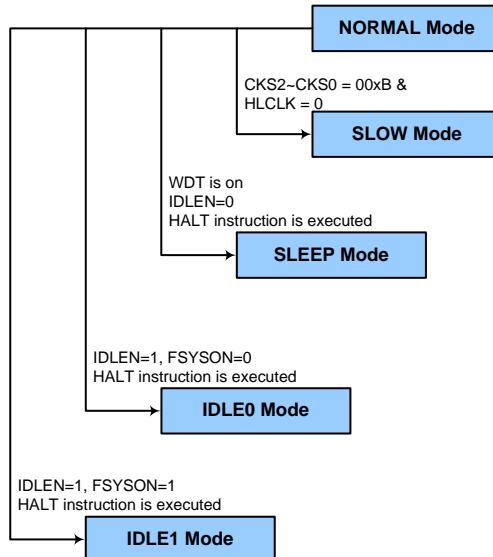
当 HLCLK 位变为低电平时, 时钟源将由高速时钟源  $f_H$  转换成时钟源  $f_H/2 \sim f_H/64$  或  $f_L$ 。若时钟源来自  $f_L$ , 高速时钟源将停止运行以节省耗电。此时必须注意内部时钟源  $f_H/16 \sim f_H/64$  也将停止运行, 这可能会影响到其它内部功能如 TM 的操作。



### 正常模式切换到低速模式

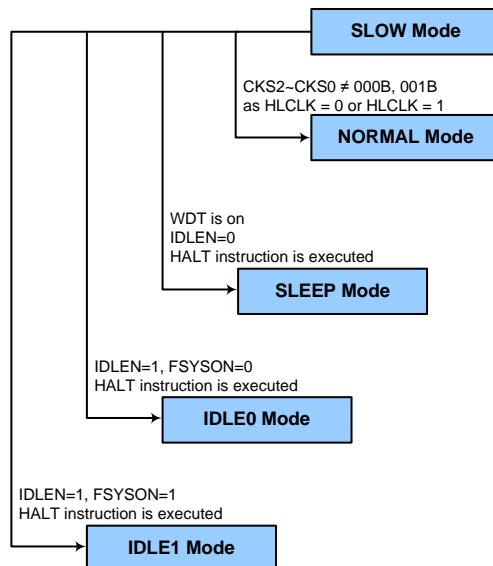
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位段为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 LIRC 振荡器，因此要求该振荡器在所有模式切换动作发生前稳定下来。该动作由 SMOD 寄存器中 LTO 位监测。



### 低速模式切换到正常模式

在低速模式系统使用 LIRC 低速振荡器。切换回使用高速系统时钟振荡器的正常模式需设置 HLCLK 位为“1”，也可设置 HLCLK 位为“0”但 CKS2~CKS0 位段需设为“010”、“011”、“100”、“101”、“110”或“111”。高频时钟需要一定的稳定时间，通过检测 HTO 位的状态可进行判断。高速振荡器稳定需要多少的延时取决于所选的高速振荡器类型。



### 进入休眠模式

进入休眠模式的方法仅有一种——应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处，而 WDT 将继续运行，其时钟源来自  $f_{LIRC}$ 。
- 数据存储器中的内容和寄存器将保持当前值。
- WDT 将被清零并在使能后重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

### 进入空闲模式 0

进入空闲模式 0 的方法仅有一种——应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 SMOD1 寄存器中的 FSYSON 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处，时基时钟将继续运行。
- 数据存储器中的内容和寄存器将保持当前值。
- WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

### 进入空闲模式 1

进入空闲模式 1 的方法仅有一种——应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 SMOD1 寄存器中的 FSYSON 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟和时基时钟开启，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- WDT 将被清零并在使能后重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

### 待机电流注意事项

由于单片机进入休眠或空闲模式的主要原因是将单片机的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式 1 除外），所以如果要将电路的电流降到最低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入 / 输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这也应用于有不同封装的单片机，因为它们可能含有未引出的引脚，这些引脚也必须设为输出或带有上拉电阻的输入。

另外还需注意单片机设为输出的 I/O 引脚上的负载。应将它们设置在有最小拉电流的状态或将它们和其它的 CMOS 输入一样接到没有拉电流的外部电路上。在空闲模式 1 中，系统时钟开启。若系统时钟来自高速系统振荡器，额外的待机电流也可能会有几百微安。

## 唤醒

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 系统中断
- WDT 溢出

若由 WDT 溢出唤醒，则会发生看门狗定时器复位。可以通过检测状态寄存器中 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，会清零 PDF；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，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

PA 口中的每个引脚都可以通过 PAWU 寄存器使能下降沿唤醒功能。PA 端口唤醒后，程序将在“HALT”指令后继续执行。如果系统是通过中断唤醒，则有两种可能发生。第一种情况是：相关中断除能或是中断使能且堆栈已满，则程序会在“HALT”指令之后继续执行。这种情况下，唤醒系统的中断会等到相关中断使能或有堆栈层可以使用之后才执行。第二种情况是：相关中断使能且堆栈未满，则中断可以马上执行。如果在进入休眠或空闲模式之前中断标志位已经被设置为“1”，则相关中断的唤醒功能将无效。

## 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。

### 看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟  $f_{LIRC}$ 。 $f_{LIRC}$  的时钟源由 LIRC 振荡器提供。看门狗定时器的时钟源可分频为  $2^8\sim2^{15}$  以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。LIRC 内部振荡器在 5V 供电下有一个约为 32kHz 的频率，应注意此指定的内部时钟可随  $V_{DD}$ 、温度以及制成工艺的不同而变化。

### 看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。当发生 WDTC 寄存器复位时，SMOD1 寄存器中的 WRF 软件复位标志位将被置高。

#### WDTC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	WE4	WE3	WE2	WE1	WE0	WS2	WS1	WS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	0	1	1

Bit 7~3   **WE4~WE0:** WDT 使能 / 除能控制

01010/10101：使能

其它值：复位 MCU

当此位段受环境噪声影响改变为其它值复位单片机时，复位动作将在 2~3 个 LIRC 时钟周期后执行且 WRF 位将置高表明复位源。

Bit 2~0      **WS2~WS0:** WDT 溢出周期选择

- 000:  $2^8/f_{LIRC}$
- 001:  $2^9/f_{LIRC}$
- 010:  $2^{10}/f_{LIRC}$
- 011:  $2^{11}/f_{LIRC}$  (默认值)
- 100:  $2^{12}/f_{LIRC}$
- 101:  $2^{13}/f_{LIRC}$
- 110:  $2^{14}/f_{LIRC}$
- 111:  $2^{15}/f_{LIRC}$

这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

### SMOD1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	—	WRF
R/W	R/W	—	—	—	—	R/W	—	R/W
POR	0	—	—	—	—	x	—	0

“x”未知

Bit7      **FSYSON:** 空闲模式下  $f_{SYS}$  控制位

详见其它章节

Bit 6~3      未定义，读为“0”

Bit 2      **LVRF:** LVR 复位标志位

详见其它章节

Bit 1      未定义，读为“0”

Bit 0      **WRF:** WDTC 控制寄存器软件复位标志位

0: 未发生

1: 发生

当 WDTC 控制寄存器发生复位时，此位置为“1”，通过应用程序清零。应注意的是该位只能通过应用程序清零。

### 看门狗定时器操作

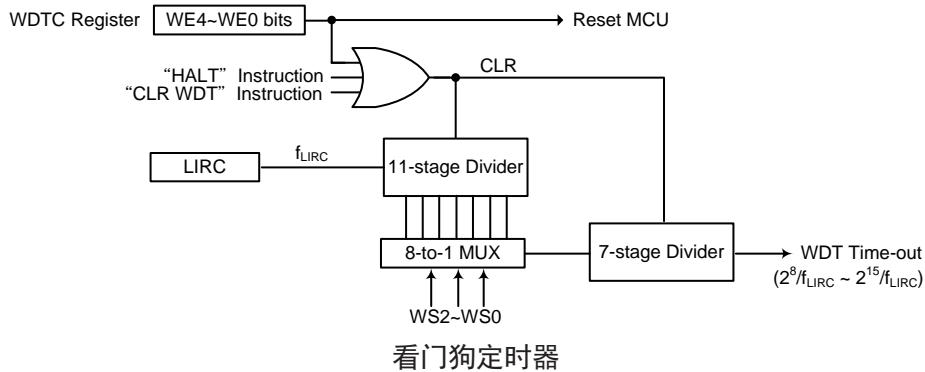
当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这些清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。WDT 始终使能，但看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位段可用于看门狗定时器操作控制。

WE4~WE0 位	WDT 功能
01010B 或 10101B	使能
其它值	复位单片机

### 看门狗定时器使能 / 复位控制

程序正常运行时，WDT 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO 被置位，程序计数器 PC 和堆栈指针 SP 将被复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDT 复位，即写入除 01010B 和 10101B 外的任何值到 WE4~WE0 位段，而第二种是通过 WDT 软件清除指令，第三种是通过“HALT”指令。只有一种软件指令用于清除看门狗定时器。因此只要执行“CLR WDT”便清除 WDT。

当设置分频比为  $2^{15}$  时，溢出周期最大。例如，时钟源为 32kHz LIRC 振荡器，分频比为  $2^{15}$  时最大溢出周期约 1s，分频比为  $2^8$  时最小溢出周期约 7.8ms。



## 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。

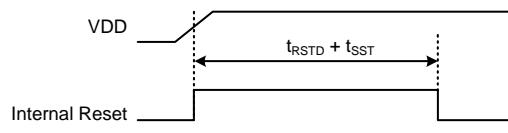
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。此外还有一种复位为低电压复位即 LVR 复位，在电源供应电压低于一定的阈值时，系统会产生完全复位。

### 复位功能

通过内部事件触发，此系列单片机有以下几种复位方式：

#### 上电复位

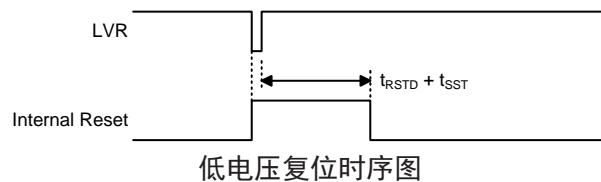
这是最基本且不可避免的复位，发生在单片机初次上电后。除了保证程序存储器从开始地址执行，上电复位也使得其它寄存器被设定在预设条件。所有的输入 / 输出端口控制寄存器在上电复位时会保持高电平，以确保上电后所有引脚被设定为输入状态。



上电复位时序图

#### 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。若电压值低于某个预定的电平则产生 MCU 复位动作。在正常和低速模式下 LVR 始终使能，其指定电压为  $V_{LVR}$ 。例如在更换电池的情况下，单片机供应的电压可能会落在  $0.9V \sim V_{LVR}$  的范围内，这时 LVR 将会自动复位单片机且 SMOD1 寄存器的 LVRF 标志位会被置位。一个有效的 LVR 信号，即在  $0.9V \sim V_{LVR}$  的低电压状态的时间，必须超过 LVR 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。 $V_{LVR}$  参数值为 2.1V，单片机将在 2~3 个 LIRC 时钟周期后复位。应注意当单片机进入休眠 / 空闲模式时 LVR 功能将自动除能。



低电压复位时序图

#### • SMOD1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	—	LVRF	—	WRF
R/W	R/W	—	—	—	—	R/W	—	R/W
POR	0	—	—	—	—	x	—	0

“x” 未知

Bit 7      **FSYSON:** 空闲模式下 fsys 控制位

详见其它章节

Bit 6~3

未定义, 读为 “0”

Bit 2

**LVRF:** LVR 复位标志位

0: 未发生

1: 发生

此位可被清零, 但不能置 “1”。

Bit 1

未定义, 读为 “0”

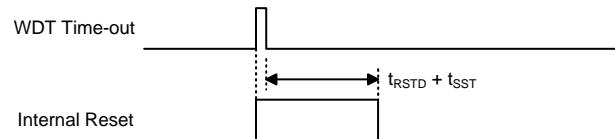
Bit 0

**WRF:** WDTC 控制寄存器复位标志位

详见其它章节

#### 正常运行时的看门狗溢出复位

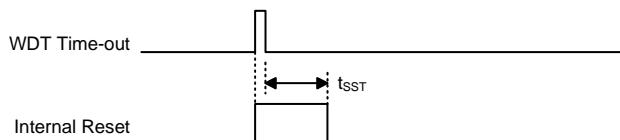
除了看门狗溢出标志位 TO 将被置高外, 正常运行时的看门狗溢出复位与 LVR 复位相同。



正常运行时的 WDT 溢出复位

#### 休眠或空闲模式时看门狗溢出复位

休眠或空闲时看门狗溢出复位和其它类型的复位有些不同。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 tsst 的详细说明请参考交流电气特性。



休眠或空闲模式时 WDT 溢出时序复位

## 复位初始条件

不同的复位形式以不同的方式影响复位标志位。这些标志位，即 PDF 和 TO 位存放在状态寄存器中，由休眠或空闲模式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：

TO	PDF	复位条件
0	0	上电复位
u	u	正常模式或低速模式时的 LVR 复位
1	u	正常模式或低速模式时的 WDT 溢出复位
1	1	空闲或休眠模式时的 WDT 溢出复位

注：“u”表示不改变

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	复位后清零，WDT 重新开始计数
定时器模块	定时器模块停止
输入 / 输出口	I/O 口设为输入模式
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。注意对于有多种封装类型的单片机，该表反应的是较大封装类型的情况。

## HT45F3420

寄存器	上电复位	正常运行时的 WDT 溢出复位	正常运行时的 LVR 复位	WDT 溢出 (HALT)*
MP0	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
MP1	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
BP	---- --0	---- --0	---- --0	---- --u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	000- 0011	uuu- uuuu
INTEG	---- --00	---- --00	---- --00	---- --uu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	00-0 00-0	00-0 00-0	00-0 00-0	uu-u uu-u
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
MFI	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	正常运行时的 WDT 溢出复位	正常运行时的 LVR 复位	WDT 溢出 (HALT)*
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	---- -000	---- -000	---- -000	---- -uuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
SMOD1	0--- -x-0	0--- -x-0	0--- -x-0	u--- -u-u
EEA	--0 0000	--0 0000	--0 0000	---u uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOL (ADRFS=0)	xxxx ----	xxxx ----	xxxx ----	uuuu ----
SADOL (ADRFS=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=0)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=1)	---- xxxx	---- xxxx	---- xxxx	---- uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0--0 0000	0--0 0000	0--0 0000	0--0 0000
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 00--	0000 00--	0000 00--	uuuu uu--
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
PWMP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMD	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL	0000 ---0	0000 ---0	0000 ---0	uuuu ---u
CPR	1000 0000	1000 0000	1000 0000	uuuu uuuu
CPOR	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPC0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCPC1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCPOCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPCCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPC0	--00 0000	--00 0000	--00 0000	--uu uuuu
OVPC1	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu

注：“\*”表示暖复位

“-”表示未定义

“u”表示不改变

“x”表示未知

### HT45F3430

寄存器	上电复位	正常运行时的 WDT 溢出复位	正常运行时的 LVR 复位	WDT 溢出 (HALT)*
MP0	XXXX XXXX	XXXX XXXX	XXXX XXXX	uuuu uuuu
MP1	XXXX XXXX	XXXX XXXX	XXXX XXXX	uuuu uuuu
BP	- - - - 0	- - - - 0	- - - - 0	- - - - - u
ACC	XXXX XXXX	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	XXXX XXXX	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-XXX XXXX	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	000- 0011	uuu- uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	00-0 00-0	00-0 00-0	00-0 00-0	uu-u uu-u
INTC2	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	--00 0000	--00 0000	--00 0000	--uu uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
SMOD1	0--- -x-0	0--- -x-0	0--- -x-0	u--- -u-u
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOL (ADRFS=0)	XXXX ----	XXXX ----	XXXX ----	uuuu ----
SADOL (ADRFS=1)	XXXX XXXX	XXXX XXXX	XXXX XXXX	uuuu uuuu
SADOH (ADRFS=0)	XXXX XXXX	XXXX XXXX	XXXX XXXX	uuuu uuuu
SADOH (ADRFS=1)	---- XXXX	---- XXXX	---- XXXX	---- uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0--0 0000	0--0 0000	0--0 0000	u--u uuuu
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- -00	---- -00	---- -00	---- -u-u
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- -00	---- -00	---- -00	---- -u-u

寄存器	上电复位	正常运行时的 WDT 溢出复位	正常运行时的 LVR 复位	WDT 溢出 (HALT)*
PWMP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMD	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL	0000 ---0	0000 ---0	0000 ---0	uuuu ---u
CPR	1000 0000	1000 0000	1000 0000	uuuu uuuu
CPOR	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPC0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCPC1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCPOCAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCPCCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPC0	--00 0000	--00 0000	--00 0000	--uu uuuu
OVPC1	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPDA	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PBS0	-----0	-----0	-----0	----- -u
PCS0	--00 0000	--00 0000	--00 0000	--uu uuuu
ASCR	---- 0000	---- 0000	---- 0000	---- uuuu
SCOMC	-000 ----	-000 ----	-000 ----	-uuu ----
SLEDC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC1	-----00	-----00	-----00	-----00

注：“\*”表示暖复位

“-”表示未定义

“u”表示不改变

“x”表示未知

## 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。大部分引脚可在用户程序控制下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，这些特性也使得此系列单片机在广泛应用上都能符合开发的需求。

该系列单片机提供 PA~PC 双向输入 / 输出口。这些 I/O 端口映射到 RAM 数据存储器中的特定地址上，如特殊功能数据存储器表所示。所有 I/O 口可用于输入 / 输出操作。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]”时 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。

单片机 型号	寄存器 名称	位							
		7	6	5	4	3	2	1	0
HT45F3420	PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
	PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
	PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
	PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
	PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
	PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
	PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
	PC	—	—	PC5	PC4	PC3	PC2	PC1	PC0
HT45F3430	PCC	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
	PCPU	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0

输入 / 输出端口控制寄存器列表

## 上拉电阻

许多产品应用在端口处于输入状态时，通常需要外加一个上拉电阻来实现上拉的功能。为了避免外部上拉电阻，当引脚配置为输入时，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~PCPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

应注意只有在 I/O 端口被设置为数字输入或 NMOS 输出时，可通过相关的上拉控制寄存器使能内部上拉功能。其它情况下，内部上拉功能除能。

### PAPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
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 7~0 PA bit 7~bit 0 上拉控制位

- 0: 除能
- 1: 使能

### PBPU 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1
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 7~0 PB bit 7~bit 0 上拉控制位

- 0: 除能
- 1: 使能

### PCPU 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义, 读为“0”

Bit 5~0 PC bit 7~bit 0 上拉控制位

- 0: 除能
- 1: 使能

### PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入休眠或空闲模式, 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 PA 口的其中一个引脚从高电平转为低电平。这个功能特别适合于通过外部开关来唤醒的应用。PA 口的每个引脚可以通过设置 PAWU 寄存器来单独选择是否具有唤醒功能。应注意只有在 PA 引脚被配置为普通 I/O 口且单片机处于 HALT 状态下, PA 端口唤醒功能可通过 PAWU 寄存器中的相关位来使能。在其它情况下, 唤醒功能除能。

### PAWU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
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 7~0 PA bit 7~bit 0 唤醒控制位

- 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, 即 PAC~PCC, 用来控制输入 / 输出配置。从而每个 I/O 引脚都可以在软件控制下, 动态地设置为 CMOS 输出或输入。I/O 端口的每个引脚都映射到其相关端口控制寄存器中的一个位。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 CMOS 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。但应注意, 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。

### PAC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
R/W								
POR	1	1	1	1	1	1	1	1

Bit 7~0 PA bit 7~bit 0 输入 / 输出控制位

0: 输出  
1: 输入

### PBC 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
R/W								
POR	1	1	1	1	1	1	1	1

Bit 7~0 PB bit 7~bit 0 输入 / 输出控制位

0: 输出  
1: 输入

### PCC 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	1	1	1	1	1	1

Bit 7~6 未定义, 读为“0”

Bit 5~0 PC bit 7~bit 0 输入 / 输出类型选择位  
0: 输出  
1: 输入

### 引脚共用功能

引脚的多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。此外, 这些引脚功能可以通过应用程序进行设定。

### 引脚共用选择寄存器

封装中有限的引脚个数会对单片机某些功能造成影响。然而, 引脚功能共用功能通过引脚功能选择, 使得小封装单片机具有更多不同的功能。该系列单片机包含 PA~PC 端口功能选择寄存器 PxS0 和 PAS1, 用于选择多功能共用引脚上的所需功能, 此外另有一个寄存器 IFS0 用于选择输入功能源引脚。

需要注意的一点是要确保所需引脚共用功能被正确地选中和取消。要正确地选择所需引脚共用功能, 需通过对相应的引脚共用控制寄存器正确配置来实现。接着配置相应的外围功能设定从而使能这些外围功能。要正确地取消选择的引脚共用功能, 应先除能外围功能, 接着修改相应的引脚共用功能控制寄存器以选择其它引脚共用功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	—	—
IFS0	—	—	—	—	—	IFS02	IFS01	IFS00

引脚共用功能选择寄存器列表 – HT45F3420

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PBS0	—	—	—	—	—	—	—	PBS00
PCS0	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
IFS0	—	—	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00

引脚共用功能选择寄存器列表 – HT45F3430

#### • PAS0 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
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 7~6      **PAS07~PAS06:** PA3 引脚共用功能选择

- 00: PA3
- 01: PA3
- 10: VREF
- 11: VREF/AN3

Bit 5~4      **PAS05~PAS04:** PA2 引脚共用功能选择

- 00: PA2
- 01: PA2
- 10: PA2
- 11: VBAT/AN2

Bit 3~2      **PAS03~PAS02:** PA1 引脚共用功能选择

- 00: PA1
- 01: PA1
- 10: PA1
- 11: OVP/AN1

Bit 1~0      **PAS01~PAS00:** PA0 引脚共用功能选择

- 00: PA0
- 01: PA0
- 10: PA0
- 11: OCP/AN0

● PAS0 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
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 7~6      **PAS07~PAS06:** PA3 引脚共用功能选择

- 00: PA3
- 01: PA3
- 10: PA3
- 11: OVP/AN3

Bit 5~4      **PAS05~PAS04:** PA2 引脚共用功能选择

- 00: PA2/STPI\_0
- 01: PA2/STPI\_0
- 10: STP
- 11: AN2

Bit 3~2      **PAS03~PAS02:** PA1 引脚共用功能选择

- 00: PA1
- 01: PA1
- 10: PA1
- 11: OCP\_0/AN1

Bit 1~0      **PAS01~PAS00:** PA0 引脚共用功能选择

- 00: PA0
- 01: PA0
- 10: PA0
- 11: AN0

● PAS1 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—
POR	0	0	0	0	0	0	—	—

Bit 7~6      **PAS17~PAS16:** PA7 引脚共用功能选择

- 00: PA7/STPI
- 01: PA7/STPI
- 10: PA7/STPI
- 11: STP

Bit 5~4      **PAS15~PAS14:** PA6 引脚共用功能选择

- 00: PA6
- 01: PA6
- 10: PA6
- 11: PWMH

Bit 3~2      **PAS13~PAS12:** PA5 引脚共用功能选择

- 00: PA5
- 01: PA5
- 10: PA5
- 11: PWML

Bit 1~0      未定义, 读为“0”

● PAS1 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
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 7~6      **PAS17~PAS16:** PA7 引脚共用功能选择

- 00: PA7
- 01: PA7
- 10: VREF
- 11: VREFI/AN7

Bit 5~4      **PAS15~PAS14:** PA6 引脚共用功能选择

- 00: PA6
- 01: PA6
- 10: PA6
- 11: OCP\_1/AN6

Bit 3~2      **PAS13~PAS12:** PA5 引脚共用功能选择

- 00: PA5
- 01: PA5
- 10: PA5
- 11: AN5

Bit 1~0      **PAS11~PAS10:** PA4 引脚共用功能选择

- 00: PA4
- 01: PA4
- 10: PA4
- 11: VBAT/AN4

● PBS0 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	PBS00
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1      未定义, 读为“0”

Bit 0      **PBS00:** PB0 引脚共用功能选择位

- 0: PB0/STPI\_1
- 1: STP

● PCS0 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6      未定义, 读为“0”

Bit 5      **PCS05:** PC5 引脚共用功能选择位

- 0: PC5
- 1: PWMH

Bit 4      **PCS04:** PC4 引脚共用功能选择位

- 0: PC4
- 1: PWML

Bit 3	<b>PCS03:</b> PC3 引脚共用功能选择位 0: PC3 1: SCOM3
Bit 2	<b>PCS02:</b> PC2 引脚共用功能选择位 0: PC2 1: SCOM2
Bit 1	<b>PCS01:</b> PC1 引脚共用功能选择位 0: PC1 1: SCOM1
Bit 0	<b>PCS00:</b> PC0 引脚共用功能选择位 0: PC0 1: SCOM0

• IFS0 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	IFS02	IFS01	IFS00
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3	未定义, 读为“0”
Bit 2	<b>IFS02:</b> OVP 输入源选择位 0: OVP_IN1 1: OVP_IN2
	此位用于选择 OVP 输入信号。OVP_IN1 信号直接来自外部 OVP 引脚而 OVP_IN2 信号来自 OVP 引脚上所集成的电压分压器电路。详情参考“集成电压分压器电路”章节。
Bit 1	<b>IFS01:</b> STCK 源选择位 0: PA7 1: PA4
Bit 0	<b>IFS00:</b> INT0 源选择位 0: PA7 1: PA4

• IFS0 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6	未定义, 读为“0”
Bit 5	<b>IFS05:</b> STPI 输入源选择位 0: PA2 1: PB0
Bit 4	<b>IFS04:</b> INT1 输入源选择位 0: PA6 1: PB3
Bit 3	<b>IFS03:</b> OCP 输入源选择位 0: PA1 1: PA6

Bit 2	<b>IFS02:</b> OVP 输入源选择位 0: OVP_IN1 1: OVP_IN2
	此位用于选择 OVP 输入信号。OVP_IN1 信号直接来自外部 OVP 引脚而 OVP_IN2 信号则来自 OVP 引脚集成电压分压器电路。详情参考“集成电压分压器电路”章节。
Bit 1	<b>IFS01:</b> STCK 源选择位 0: PA5 1: PB1
Bit 0	<b>IFS00:</b> INT0 输入源选择位 0: PA5 1: PB2

### 输入 / 输出端口源电流控制 – 仅用于 HT45F3430

HT45F3430 对于每个输入 / 输出端口支持不同的源电流驱动能力。通过相应的选择寄存器 SLEDC0 和 SLEDC1，每个输入 / 输出端口可支持 4 个 level 的源电流驱动能力。这可用于 LED 驱动应用。用户可参考直流电气特性章节针对不同应用选择所需源电流。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SLEDC0	PBPS3	PBPS2	PBPS1	PBPS0	PAPS3	PAPS2	PAPS1	PAPS0
SLEDC1	—	—	—	—	—	—	PCPS1	PCPS0

输入 / 输出端口源电流控制寄存器列表

#### SLEDC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PBPS3	PBPS2	PBPS1	PBPS0	PAPS3	PAPS2	PAPS1	PAPS0
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 7~6 **PBPS3~PBPS2:** PB7~PB4 源电流选择

- 00: 源电流 = Level 0 (min.)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (max.)

##### Bit 5~4 **PBPS1~PBPS0:** PB3~PB0 源电流选择

- 00: 源电流 = Level 0 (min.)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (max.)

##### Bit 3~2 **PAPS3~PAPS2:** PA7~PA4 源电流选择 (PMOS 调整)

- 00: 源电流 = Level 0 (min.)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (max.)

##### Bit 1~0 **PAPS1~PAPS0:** PA3~PA0 源电流选择 (PMOS 调整)

- 00: 源电流 = Level 0 (min.)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (max.)

注：用户需参考直流电气特性章节针对不同应用选取确定的值。

### SLEDC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PCPS1	PCPS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 保留位，可读写

Bit 1~0 **PCPS1~PCPS0:** PC3~PC0 源电流选择 (PMOS 调整)

00: 源电流 = Level 0 (min.)

01: 源电流 = Level 1

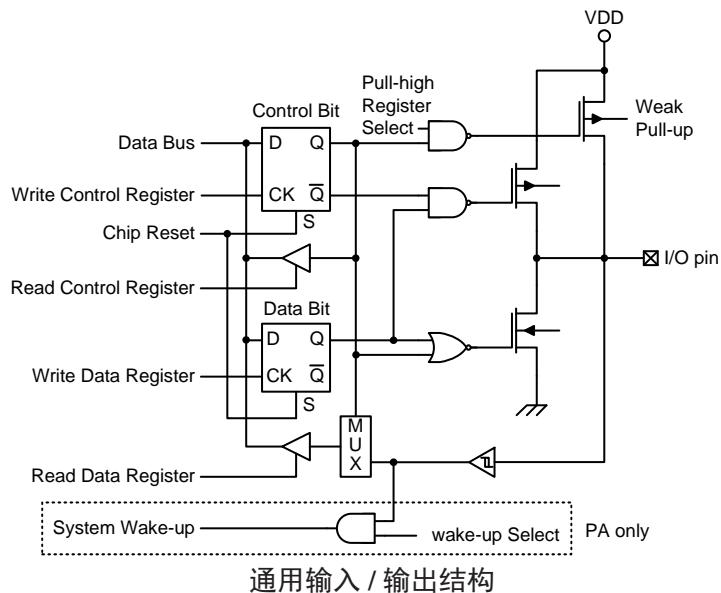
10: 源电流 = Level 2

11: 源电流 = Level 3 (max.)

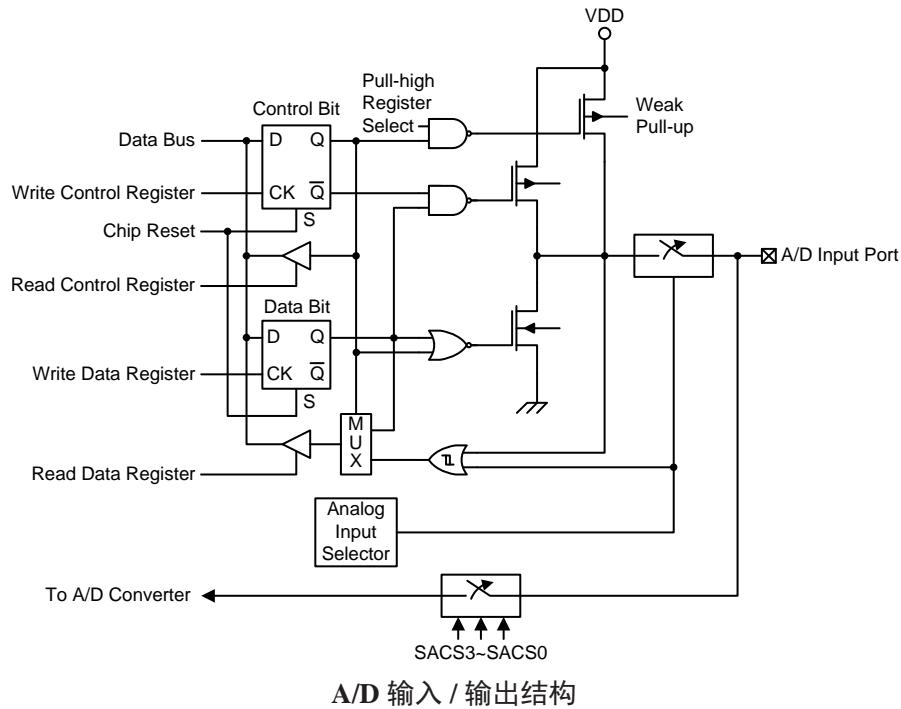
注：用户需参考直流电气特性章节针对不同应用选取确定的值。

### 输入 / 输出引脚结构

下图为某些一般类型输入 / 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 I/O 引脚功能的理解提供的一个参考。图中的引脚共用结构并非针对所有单片机。



通用输入 / 输出结构



### 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。也就是说所有输入 / 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。此时若对端口控制寄存器 PAC~PCC 编程以将一些引脚设定为输出状态，除非相关的端口数据寄存器 PA~PC 程序中预先被设定，否则这些引脚会有初始高电平输出。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。

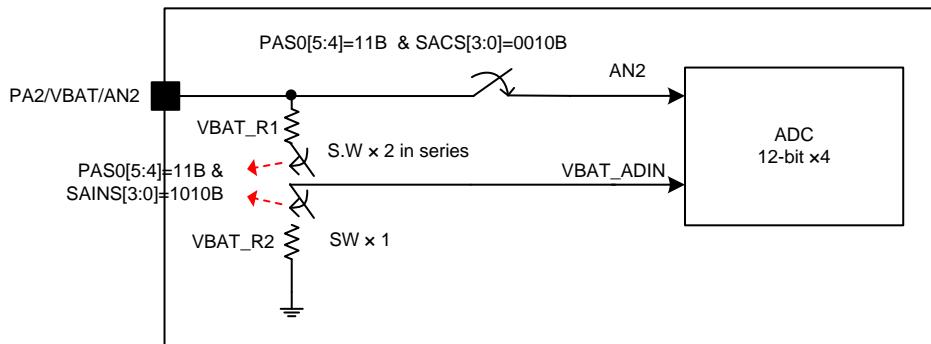
PA 口的每个引脚另带唤醒功能。单片机处于休眠或空闲模式时，有很多方法可以唤醒单片机，其中之一就是通过 PA 任一引脚电平从高到低转换的方式，可以设置 PA 口一个或多个引脚使其具有唤醒功能。

## 集成电压分压器电路

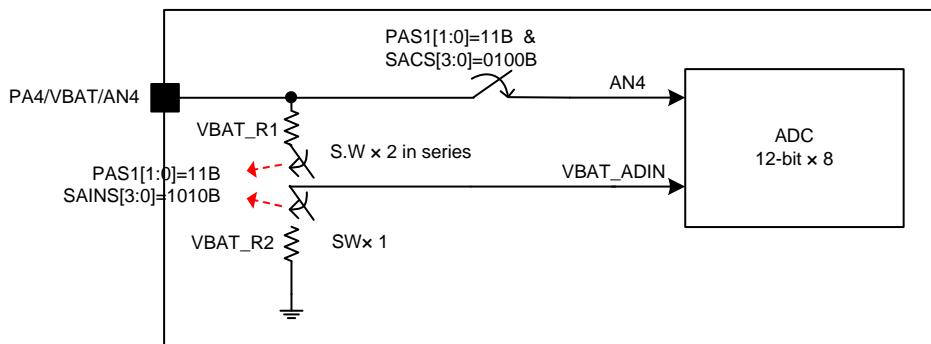
该系列单片机包含了高度集成的电阻分压器电路，用于测量外部电池电压，同时减少对外部元器件的需求。

### 功能性描述

这些电阻分压器的内部电阻可通过内部模拟开关控制其连接状态。

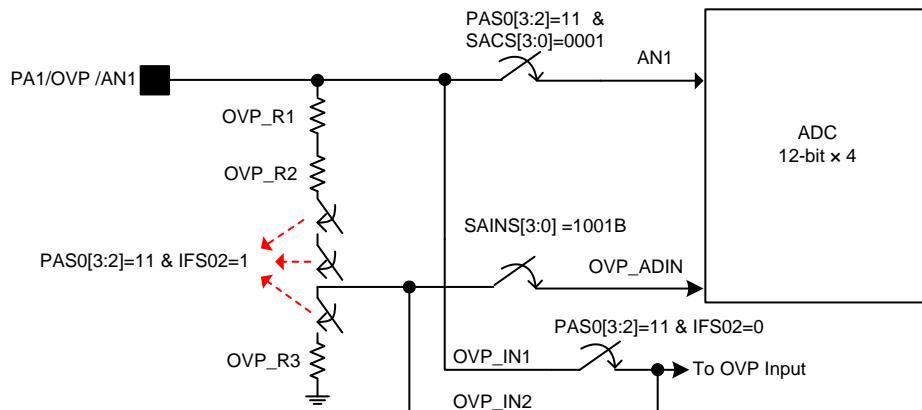
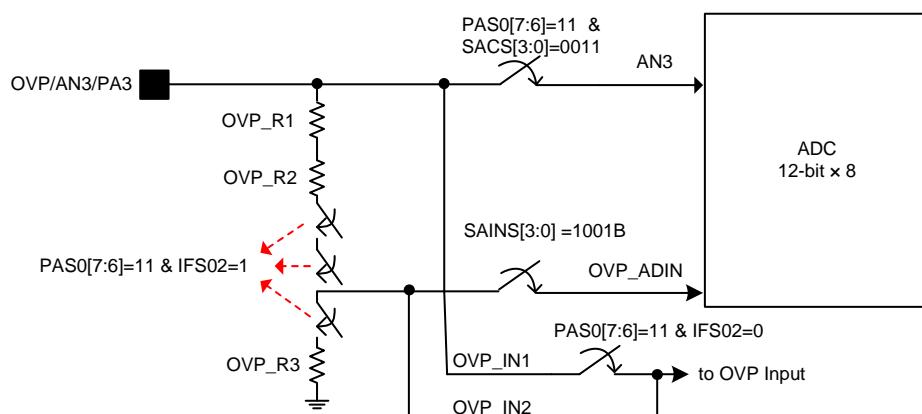


VBAT 引脚电压分压器电路 – HT45F3420



VBAT 引脚电压分压器电路 – HT45F3430

电池电压输入引脚即 VBAT 引脚连接至内部电阻分压器电路。VBAT\_R1 的值为 2K，VBAT\_R2 的值为 1K，从而可得分压比为 2:1，其精度必须等于 1%。在 VBAT 引脚上输入 4.2V 电压，通过这个电压分压器将产生一个 1.4V 的电压。当 SADC1 寄存器中的  $SAINS[3:0]=1010B$  时，VBAT\_R1 和 VBAT\_R2 电阻之间的开关将全部闭合。此时 VBAT\_R1 和 VBAT\_R2 直接相连，分压后的电压直接提供给 VBAT\_ADIN。当  $SAINS[3:0] \neq 1010B$ ，则 VBAT\_R1 和 VBAT\_R2 电阻之间的开关将全部断开。


**OVP 引脚电压分压器电路 – HT45F3420**

**OVP 引脚电压分压器电路 – HT45F3430**

在过电压保护功能中，OVP 引脚连接至内部电压分压器。这由电阻 OVP\_R1+OVP\_R2 (总电阻值为 2K) 和 OVP\_R3 (电阻值为 1K) 组成，由此可得其分压比为 2:1，其精度必须等于 1%。OVP\_R1+OVP\_R2+ OVP\_R3 的总的电阻值应为 3K，但该值的容差可达 50%。

若 IFS0 寄存器的位 IFS02=1，则 OVP\_R2 和 OVP\_R3 电阻之间的开关将全部闭合。此时 OVP\_R2 和 OVP\_R3 电阻将相连，所得分压后的电压将提供给 OVP\_IN2。当 IFS02=0，OVP\_R2 和 OVP\_R3 电阻之间的开关将全部断开。

## 定时器模块 – TM

控制和测量时间在任何单片机中都是一个很重要的部分。该系列单片机提供一个标准型定时器模块(简称 STM)，来实现和时间有关的功能。STM 是包括多种操作的定时单元，提供的操作有：定时 / 事件计数器，捕捉输入，比较匹配输出，单脉冲输出以及 PWM 输出等功能。定时器模块有两个独立中断，其外加的输入 / 输出引脚，扩大了定时器的灵活性，便于用户使用。

这里将描述标准型 TM 的一般特征，更多详情请参考标准型 TM 章节。

### 简介

该系列单片机包含一个 TM，即 STM。本节将描述标准型 TM 的一般特征，详细操作将在相应的章节中讲述。STM 的主要特性如下表所示。

TM 功能	STM
定时 / 计数器	√
捕捉输入	√
比较匹配输出	√
PWM 通道数	1
单脉冲输出	1
PWM 对齐方式	边沿对齐
PWM 调节周期 & 占空比	占空比或周期

TM 功能概要

### TM 操作

标准型 TM 提供从简单的定时操作到 PWM 信号产生等多种功能。理解 TM 操作的关键是比较 TM 内独立运行的计数器的值与内部比较器的预置值。当计数器的值与比较器的预置值相同时，则比较匹配，TM 中断信号产生，清零计数器并改变 TM 输出引脚的状态。用户选择内部时钟或外部时钟来驱动内部 TM 计数器。

### TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 STMC0 控制寄存器的 STCK2~STCK0 位段可选择所需的时钟源。该时钟源来自系统时钟  $f_{SYS}$  或内部高速时钟  $f_H$  或  $f_{TBC}$  时钟源或外部 STCK 引脚。STCK 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。

### TM 中断

标准型 TM 各有两个内部中断，即内部比较器 A 和比较器 P，当发生比较匹配时比较器将产生一个 TM 中断。中断产生时将清除计数器，并改变 TM 输出引脚的状态。

## TM 外部引脚

TM 有两个 TM 输入引脚，分别是 STCK 和 STPI。STCK 作为 TM 的时钟源输入脚，通过设置 STMC0 寄存器中的 STCK2~STCK0 位段进行选择。外部时钟源可通过该引脚来驱动内部 TM。TM 输入引脚可以选择上升沿或下降沿有效，通过 STMC1 寄存器中的 STIO1~STIO0 位段选择有效边沿转换类型。

TM 有一个输出引脚 STP。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 STP 输出引脚也被 TM 用来产生 PWM 输出波形。

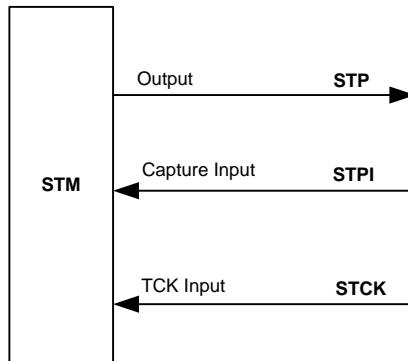
由于 STM 输入或输出引脚与其它功能共用，TM 输出功能需要通过寄存器先被设置。寄存器中的单个位决定了其相关引脚是用作外部 TM 输出引脚还是其它功能。

单片机型号	STM	
	输入	输出
HT45F3420/HT45F3430	STCK/STPI	STP

TM 外部引脚

## TM 输入 / 输出引脚控制寄存器

选择作为 TM 输入 / 输出或其它共用功能可通过配置一个寄存器中与 TM 输入 / 输出引脚相对应的单个位实现。适当配置选择位则相应引脚功能可设为 TM 输入 / 输出。详情参考引脚共用功能章节中的引脚共用功能选择。

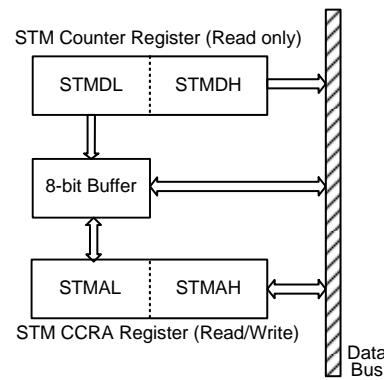


STM 功能引脚控制方框图

## 编程注意事项

TM 计数器寄存器和捕捉 / 比较 CCRA 寄存器，含有低字节和高字节结构。高字节可直接访问，低字节则仅能通过一个内部 8-bit 的缓存器进行访问。读写这些成对的寄存器需通过特殊的方式。值得注意的是 8-bit 缓存器的存取数据及相关低字节的读写操作仅在其相应的高字节读取操作执行时发生。

由于 CCRA 寄存器访问方式如下图所示，读写这些成对的寄存器需通过特殊的方式。建议使用“MOV”指令，通过以下步骤访问 CCRA 低字节寄存器。否则将导致不可预期的结果。

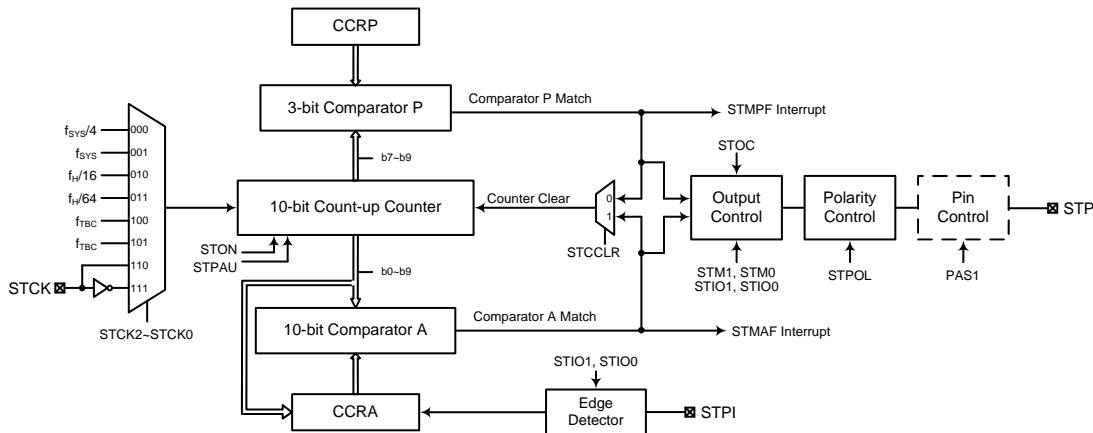


读写流程如下步骤所示：

- 写数据至 CCRA
  - ◆ 步骤 1. 写数据至低字节寄存器 STMAL
    - 注意，此时数据仅写入 8-bit 缓存器。
  - ◆ 步骤 2. 写数据至高字节寄存器 STMAH
    - 注意，此时数据直接写入高字节寄存器，同时锁存在 8-bit 缓存器中的数据写入低字节寄存器。
- 由计数器寄存器和 CCRA 中读取数据
  - ◆ 步骤 1. 从高字节寄存器 STMDH、STMAH 中读取数据
    - 注意，此时高字节寄存器中的数据直接读取，同时由低字节寄存器读取的数据锁存至 8-bit 缓存器中。
  - ◆ 步骤 2. 从低字节寄存器 STMDL、STMAL 中读取数据
    - 注意，此时读取 8-bit 缓存器中的数据。

## 标准型 TM – STM

标准型 TM 包括 5 种工作模式，即比较匹配输出，定时 / 事件计数器，捕捉输入，单脉冲输出和 PWM 输出模式。标准型 TM 可由两个外部输入脚控制并驱动一个外部输出脚。



注：对于 HT45F3430，其 STCK 和 STPI 输入源引脚可通过 IFS0 寄存器的相应位进行选择。

标准型 TM 方框图

## 标准型 TM 操作

标准型 TM 的核心是一个由用户选择的内部时钟源驱动的 10-bit 向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 3 位宽度，与计数器的高 3 位比较；而 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10-bit 计数器值的唯一方法是使 STON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 STM 中断信号。标准型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

## 标准型 TM 寄存器

标准型 TM 的所有操作由一系列寄存器控制。一对只读寄存器用来存放 10-bit 计数器的值，一对读 / 写寄存器存放 10-bit CCRA 的值，剩下两个控制寄存器设置不同的操作和控制模式以及 3 位 CCRP 的值。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
STMC0	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDGX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	—	—	—	—	—	—	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	—	—	—	—	—	—	D9	D8

10-bit 标准型 TM 寄存器列表

### STMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
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      **STPAU:** STM 计数器暂停控制位

- 0: 运行
- 1: 暂停

通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，STM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4      **STCK2~STCK0:** STM 计数器时钟选择

- 000: f<sub>SYS</sub>/4
- 001: f<sub>SYS</sub>
- 010: f<sub>H</sub>/16
- 011: f<sub>H</sub>/64
- 100: f<sub>TBC</sub>
- 101: f<sub>TBC</sub>
- 110: STCK 上升沿时钟
- 111: STCK 下降沿时钟

此三位用于选择 STM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f<sub>SYS</sub> 是系统时钟，f<sub>H</sub> 和 f<sub>TBC</sub> 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3	<b>STON:</b> STM 计数器 On/Off 控制位 0: Off 1: On
	此位控制 STM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 STM。清零此位将停止计数器并关闭 STM 以减少耗电。当此位经由低到高转换时，内部计数器将复位清零，当此位由高到低转换时，内部计数器将保持其剩余值直到此位再次变高。若 STM 处于比较匹配输出模式或 PWM 输出模式或单脉冲输出模式时，当 STON 位经由低到高转换时，STM 输出脚将复位至 STOC 位指定的初始值。
Bit 2~0	<b>STRP2~STRP0:</b> STM CCRP 3-bit 寄存器，与 STM 计数器 bit 9~bit 7 比较 比较器 P 匹配周期 000: 1024 个 STM 时钟 001: 128 个 STM 时钟 010: 256 个 STM 时钟 011: 384 个 STM 时钟 100: 512 个 STM 时钟 101: 640 个 STM 时钟 110: 768 个 STM 时钟 111: 896 个 STM 时钟  此三位设定内部 CCRP 3-bit 寄存器的值，然后与内部计数器的高三位进行比较。如果 STCCLR 位设定为 0 时，选中该比较结果清除内部计数器。STCCLR 位设为 0，内部计数器在比较器 P 比较匹配发生时被重置；由于 CCRP 只与计数器高三位比较，比较结果是 128 时钟周期的倍数。CCRP 被清零时，会使得计数器在最大值溢出。

### STMC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDGX	STCCLR
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 7~6	<b>STM1~STM0:</b> 选择 STM 工作模式 00: 比较匹配输出模式 01: 捕捉输入模式 10: PWM 输出模式或单脉冲输出模式 11: 定时 / 计数器模式
	这两位设置 STM 需要的工作模式。为了确保操作可靠，STM 应在 STM1 和 STM0 位有任何改变前先关掉。在定时 / 计数器模式，STM 输出脚状态未定义。
Bit 5~4	<b>STIO1~STIO0:</b> 选择 STM 功能 比较匹配输出模式 00: 无变化 01: 输出低 10: 输出高 11: 输出翻转 PWM 输出模式 / 单脉冲输出模式 00: PWM 输出无效状态 01: PWM 输出有效状态 10: PWM 输出 11: 单脉冲输出 捕捉输入模式 00: 在 STPI 上升沿输入捕捉 01: 在 STPI 下降沿输入捕捉 10: 在 STPI 双沿输入捕捉 11: 输入捕捉除能

## 定时 / 计数器模式

未使用

此两位用于决定在一定条件达到时 STM 输出脚如何改变状态。这两位值的选择取决于 STM 运行在哪种模式下。

在比较匹配输出模式下，STIO1~STIO0 位段决定了当比较器 A 比较匹配发生时 STM 输出脚如何改变状态。当比较器 A 比较匹配发生时 STM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。STM 输出脚的初始值通过 STOC 位设置取得。注意，由 STIO1~STIO0 位段得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STM 输出脚将不会发生变化。在 STM 输出脚改变状态后，通过 STON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，STIO1~STIO0 位段用于决定比较匹配条件发生时怎样改变 STM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 STM 关闭时改变 STIO1~STIO0 位段的值是很有必要的。若在 STM 运行时改变 STIO1~STIO0 位段的值，PWM 输出的值是无法预料的。

Bit 3

**STOC:** STM 输出控制位

## 比较匹配输出模式

0: 初始低

1: 初始高

## PWM 输出模式 / 单脉冲输出模式

0: 低有效

1: 高有效

这是 STM 输出脚输出控制位。它取决于 STM 此时正运行于比较匹配输出模式还是 PWM 模式 / 单脉冲输出模式。若 STM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 STM 输出脚的逻辑电平值。在 PWM 输出模式时，其决定 PWM 信号是高有效还是低有效。在单脉冲输出模式时，其决定当 STON 位由低转高时 STM 输出脚的逻辑电平值。

Bit 2

**STPOL:** STM 输出极性控制位

0: 同相

1: 反相

此位控制 STM 输出脚的极性。此位为高时 STM 输出脚反相，为低时 STM 输出脚同相。若 STM 处于定时 / 计数器模式时其不受影响。

Bit 1

**STDPX:** STM PWM 周期 / 占空比控制位

0: CCRP - 周期; CCRA - 占空比

1: CCRP - 占空比; CCRA - 周期

此位决定 CCRA 与 CCRP 寄存器哪个被用于 PWM 波形的周期和占空比控制。

Bit 0

**STCCLR:** 选择 STM 计数器清零条件位

0: STM 比较器 P 匹配

1: STM 比较器 A 匹配

此位用于选择清除计数器的方法。标准型 TM 包括两个比较器，即比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。STCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。STCCLR 位在 PWM 输出、单脉冲或输入捕捉模式时未使用。

### STMDL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 STM 计数器低字节寄存器 bit 7~bit 0  
STM 10-bit 计数器 bit 7~bit 0

### STMDH 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”  
Bit 1~0 STM 计数器高字节寄存器 bit 1~bit 0  
STM 10-bit 计数器 bit 9~bit 8

### STMAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0 STM CCRA 低字节寄存器 bit 7~bit 0  
STM 10-bit CCRA bit 7~bit 0

### STMAH 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”  
Bit 1~0 STM CCRA 高字节寄存器 bit 1~bit 0  
STM 10-bit CCRA bit 9~bit 8

## 标准型 TM 工作模式

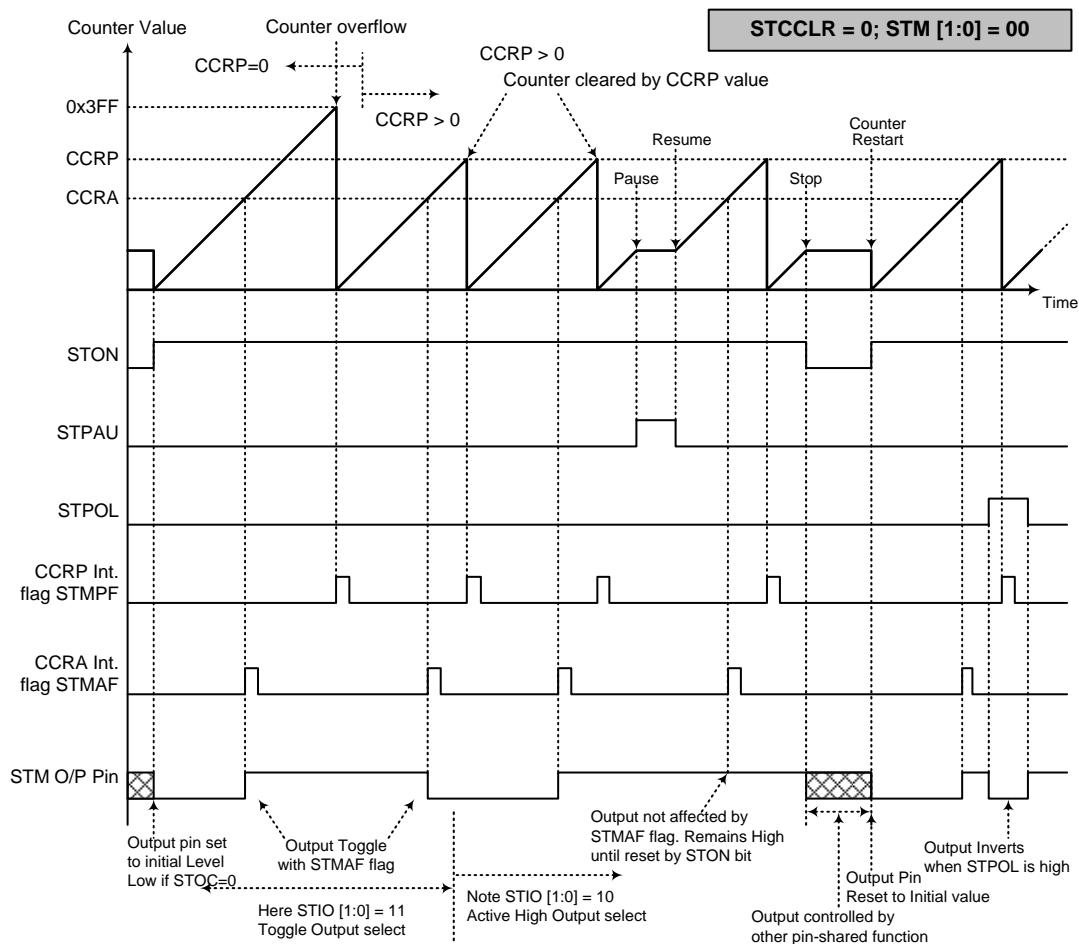
标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式或定时 / 计数器模式。通过设置 STMC1 寄存器的 STM1 和 STM0 位选择任意模式。

### 比较匹配输出模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 STCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 STMAF 和 STMPF 将分别置位。

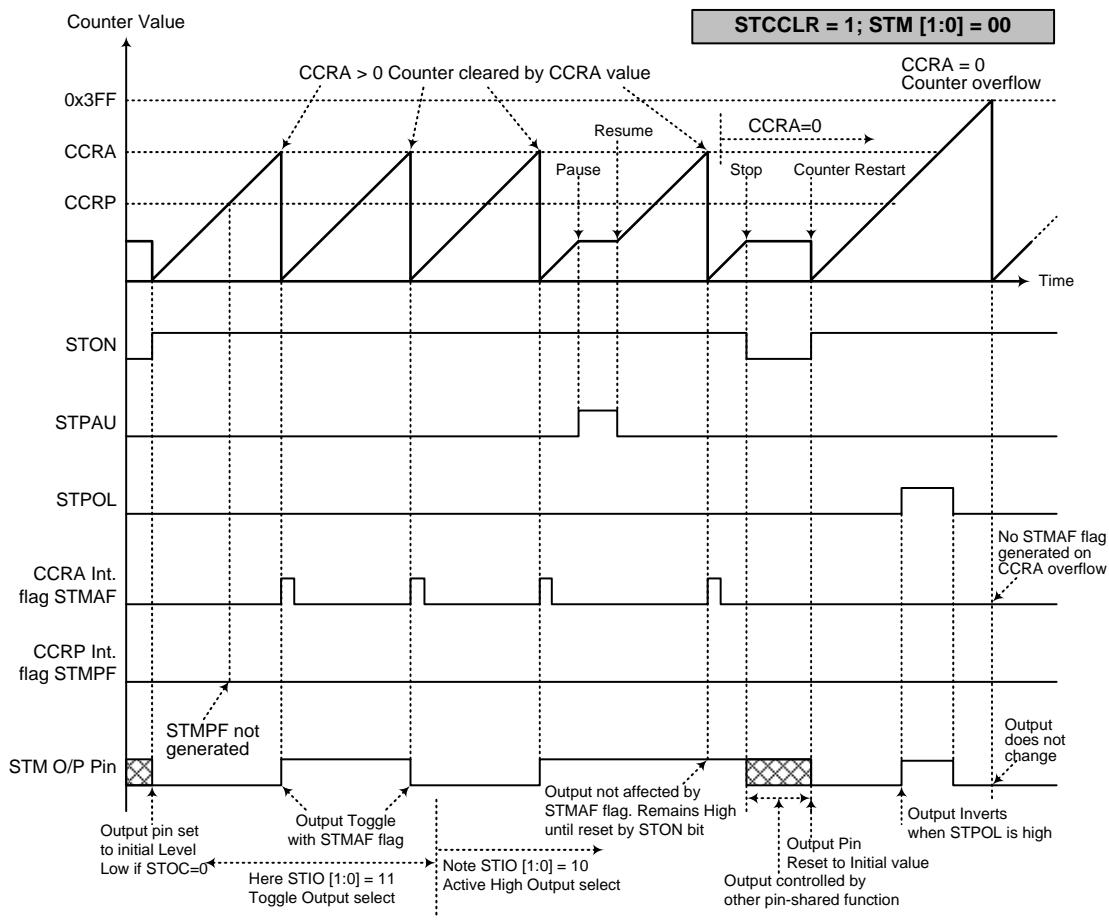
如果 STMC1 寄存器的 STCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMAF 中断请求标志。所以当 STCCLR 为高时，不会产生 STMPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 STMAF 请求标志。

正如该模式名所言，当比较匹配发生后，STM 输出脚状态改变。当比较器 A 比较匹配发生后 STMAF 标志产生时，STM 输出脚状态改变。比较器 P 比较匹配发生时产生的 STMPF 标志不影响 STM 输出脚。STM 输出脚状态改变方式由 STMC1 寄存器中 STIO1 和 STIO0 位决定。当比较器 A 比较匹配发生时，STIO1 和 STIO0 位决定 STM 输出脚输出高，低或翻转当前状态。STM 输出脚初始值，在 STON 位由低到高电平的变化后通过 STOC 位设置。注意，若 STIO1 和 STIO0 位同时为 0 时，引脚输出不变。



### 比较匹配输出模式 – STCCLR=0

- 注:
1. STCCLR=0, 比较器 P 匹配将清除计数器
  2. TM 输出引脚仅由 STMAF 标志位控制
  3. 输出引脚在 STON 上升沿复位至初始值



### 比较匹配输出模式 – STCCLR=1

- 注：
1. STCCLR=1，比较器 A 匹配将清除计数器
  2. TM 输出引脚仅由 STMAF 标志位控制
  3. 输出引脚在 STON 上升沿复位至初始值
  4. 当 STCCLR=1 时，不产生 STMPF 标志位

### 定时 / 计数器模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 应设置为 11。定时 / 计数器模式操作方式与比较匹配输出模式相同，产生相同的中断标志位。唯一不同的就是在定时 / 计数器模式中未使用 STM 输出引脚。因此，比较匹配输出模式中的描述和时序图可以帮助理解此功能。该模式中未使用的 STM 输出脚用作普通 I/O 脚或其它功能。

### PWM 输出模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，且 STIO1 和 STIO0 位也需要设置为“10”。STM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 STM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就极其灵活。在 PWM 模式中，STCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 STMC1 寄存器的 STDPX 位。因此，PWM 波形的频率和占空比受 CCRA 和 CCRP 寄存器中的值控制。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。STMC1 寄存器中的 STOC 位决定 PWM 波形的极性，STIO1 和 STIO0 位使能 PWM 输出或将 STM 输出脚置为逻辑高或逻辑低。STPOL 位对 PWM 输出波形的极性取反。

#### • 10-bit STM, PWM 输出模式，边沿对齐模式，STDPX=0

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	128	256	384	512	640	768	896	1024
Duty	CCRA							

若  $f_{SYS}=4MHz$ , TM 时钟源为  $f_{SYS}/4$ , CCRP=100b 且 CCRA=128,

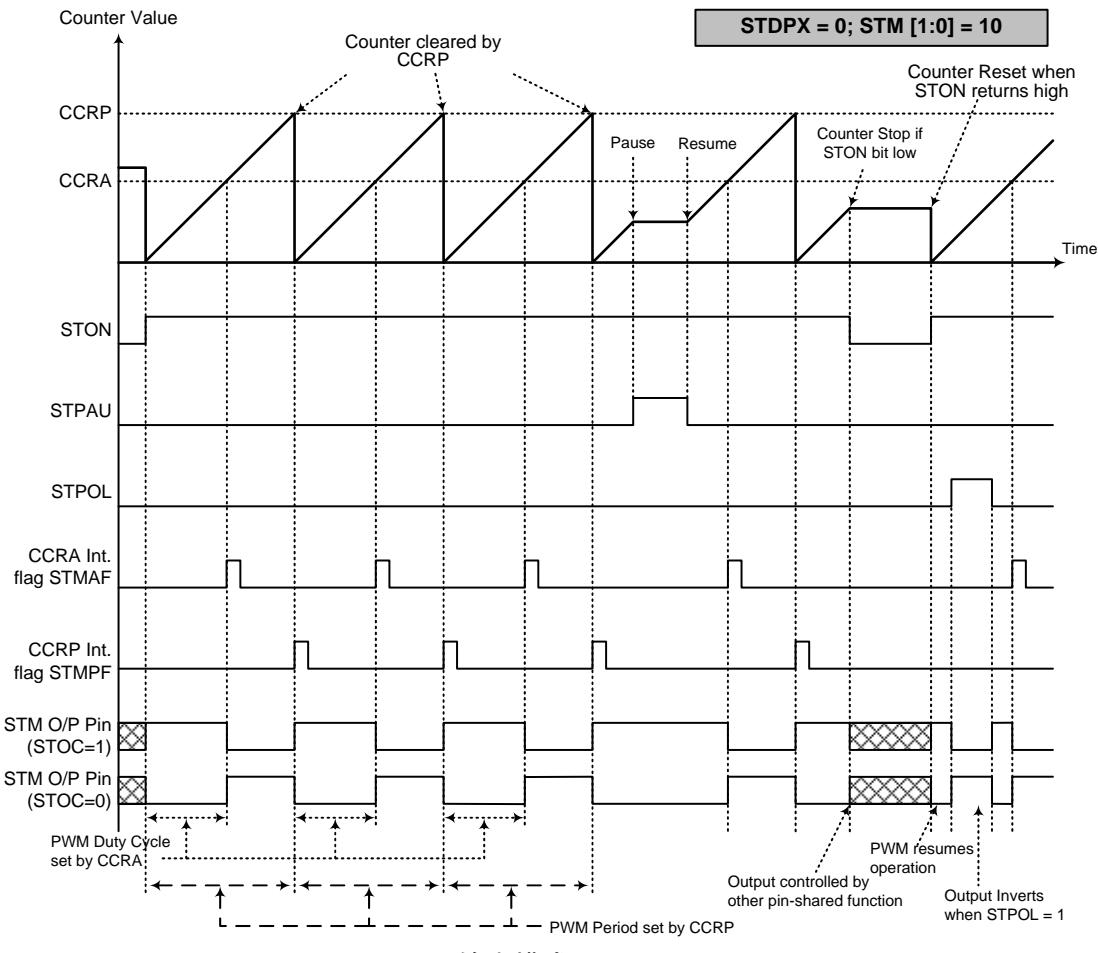
STM PWM 输出频率 =  $(f_{SYS}/4)/512=f_{SYS}/2048=1.9531kHz$ , 周期 =  $128/512=25\%$ 。

若 CCRA 所定义的 Duty 值等于或大于 Period 值，则 PWM 输出占空比为 100%。

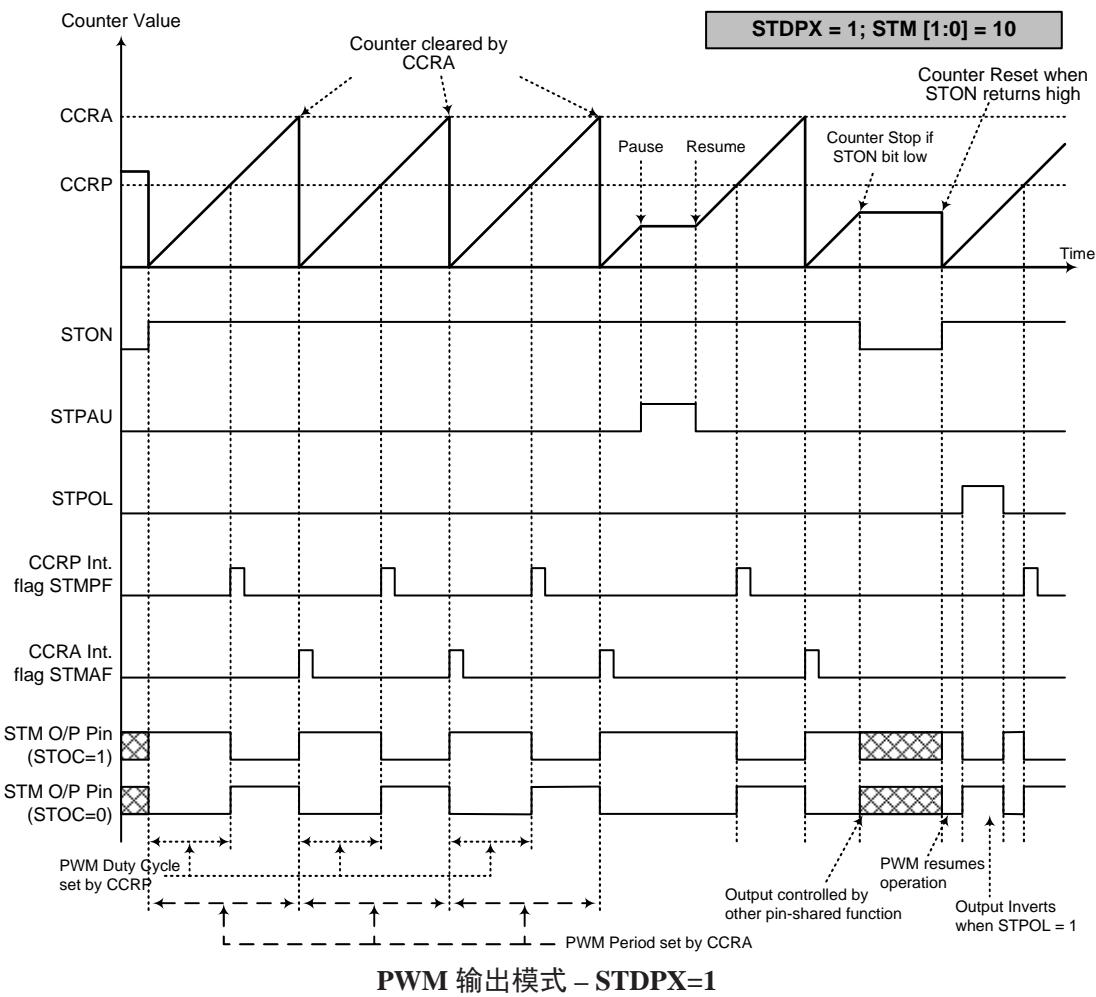
#### • 10-bit STM, PWM 输出模式，边沿对齐模式，STDPX=1

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

PWM 的输出周期由 CCRA 寄存器的值与 STM 的时钟共同决定，PWM 的占空比由 CCRP 的值决定。



- 注：
1. STDPX=0 – 计数器由 CCRP 清除
  2. 计数器清除设置 PWM 周期
  3. 即使当 STIO[1:0]=00 或 01 时，内部 PWM 功能继续运行
  4. STCCLR 对 PWM 操作无影响

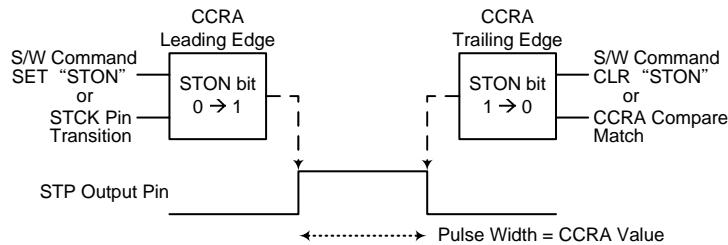


- 注:
1. STDPX=1 – 计数器由 CCRA 清除
  2. 计数器清除设置 PWM 周期
  3. 即使当 STIO[1:0]=00 或 01 时, 内部 PWM 功能继续运行
  4. STCCLR 对 PWM 操作无影响

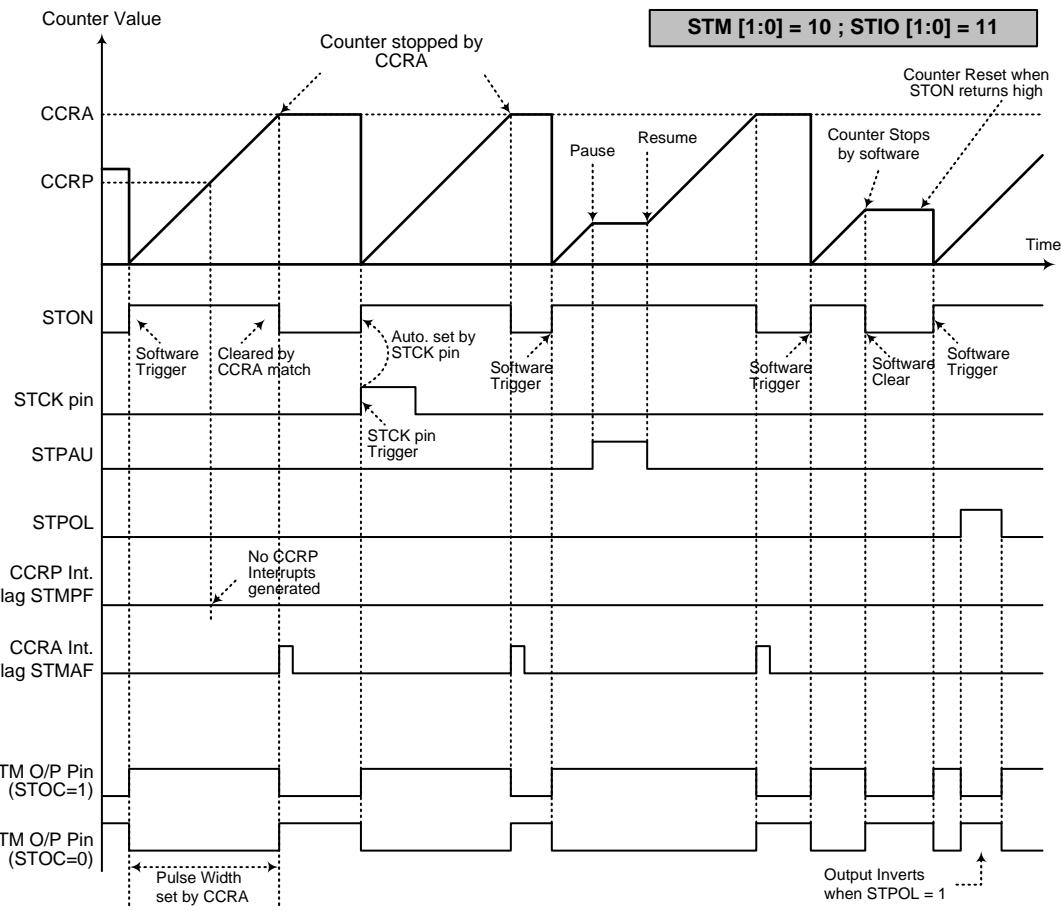
### 单脉冲模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STM 输出脚将产生一个单脉冲输出。

脉冲输出可以通过应用程序控制 STON 位由低到高的转变来触发。而处于单脉冲模式时，STON 位可利用外部 STCK 脚自动由低转变为高，进而开始单脉冲输出。当 STON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 STON 位保持高电平。通过应用程序使 STON 位清零或比较器 A 比较匹配发生时，产生脉冲下降沿。



单脉冲产生示意图



单脉冲模式

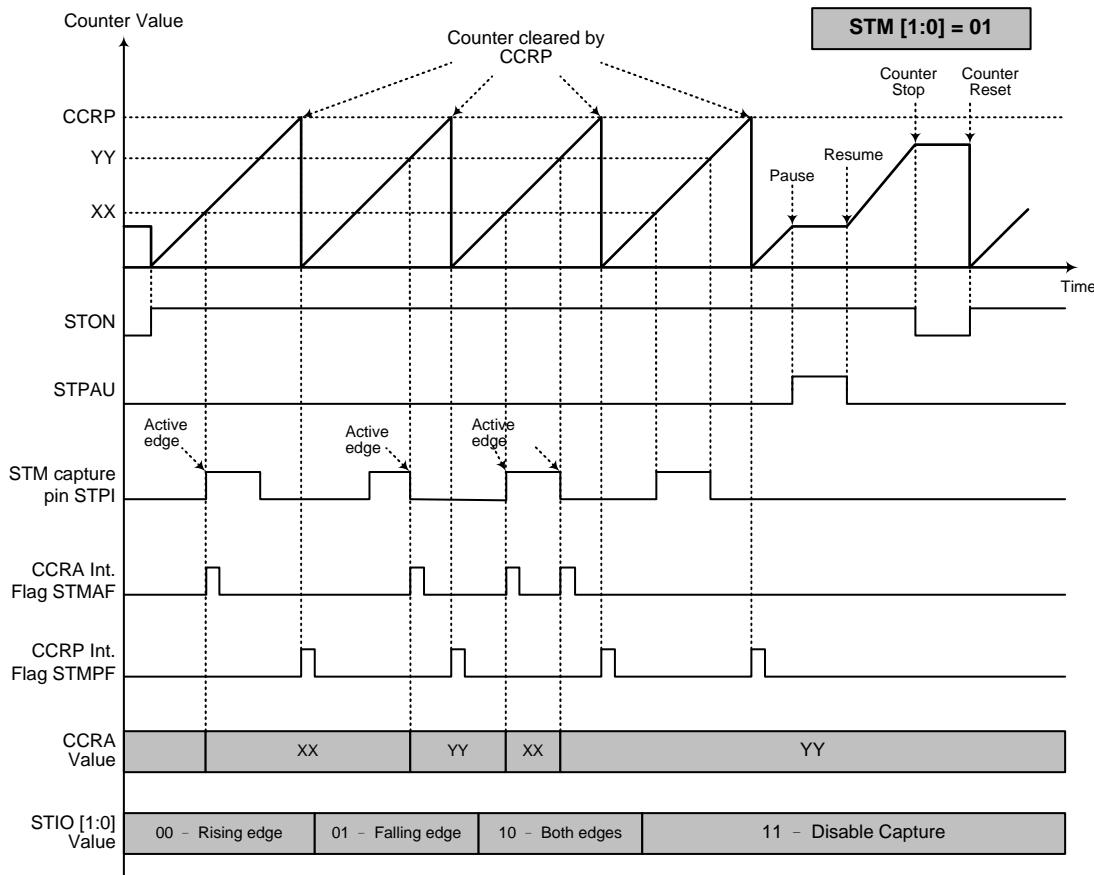
- 注:
1. 通过 CCRA 匹配停止计数器
  2. CCRP 未使用
  3. 通过设 STON 为高触发电脉冲
  4. 在单脉冲模式下, STIO[1:0] 必须设为 “11” 且不能被改变。

但比较器 A 的比较匹配也会自动清除 STON 位, 从而产生单脉冲输出下降沿。此时 CCRA 的值可用于控制脉冲宽度。比较器 A 的比较匹配也能产生一个 STM 中断信号。当计数器重新启动, STON 位从低到高转换时, 计数器将被复位回 0。在单脉冲模式下, CCRP 寄存器, STCCLR 位和 STDPX 位未使用。

### 捕捉输入模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。STPI 脚上的外部信号，通过设置 STMC1 寄存器的 STIO1~STIO0 位段选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将 STON 位由低到高转变时，计数器启动。

当 STPI 脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 STM 中断。无论 STPI 引脚发生何种事件，计数器将继续工作直到 STON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 STM 中断。记录 CCRP 溢出中断信号的值可以测量脉宽。通过设置 STIO1~STIO0 位段选择 STPI 引脚为上升沿，下降沿或双沿有效。如果 STIO1~STIO0 位段都设置为高，无论 STPI 引脚发生哪种边沿转换都不会产生捕捉操作，但应注意计数器将会继续运行。STCCLR 和 STDPX 位在此模式中未使用。



### 捕捉输入模式

- 注：
1. STM[1:0]=01，有效边沿通过 STIO[1:0] 位段设置
  2. TM 捕捉输入引脚有效边沿将计数器的值传到 CCRA 中
  3. STCCLR 和 STDPX 位未使用
  4. 无输出功能 – STOC 和 STPOL 位未使用
  5. CCRP 决定计数器的值，且当 CCRP 等于 0 时计数器有一个最大计数值

## A/D 转换器

对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

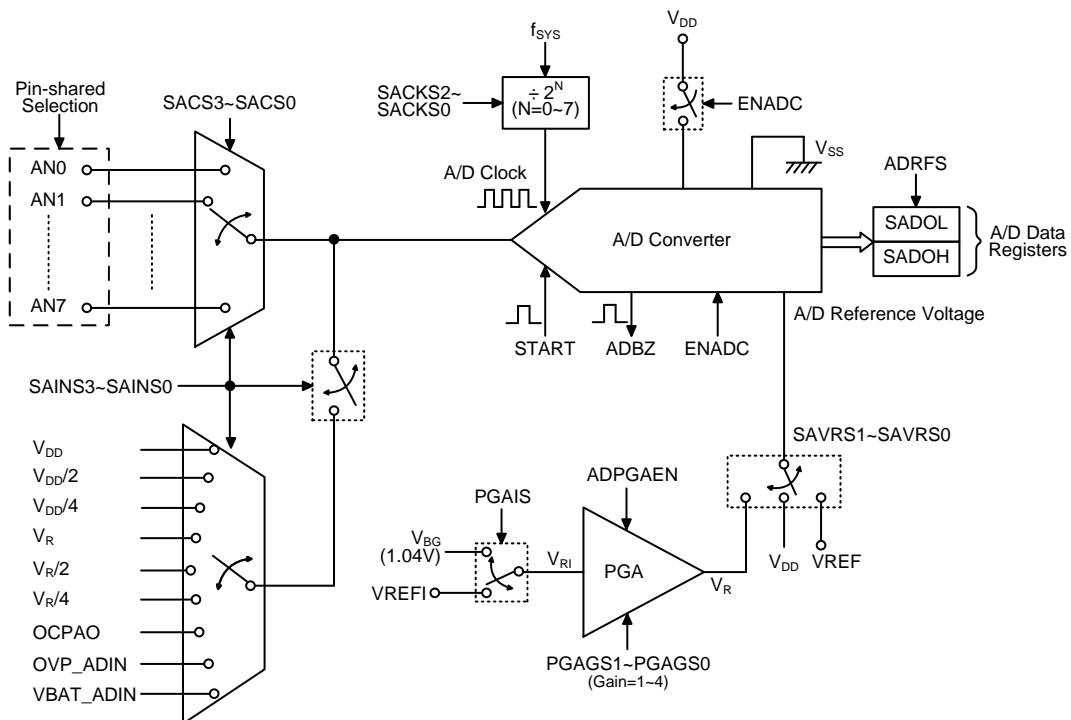
### A/D 简介

该系列每款单片机各包含一个多通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）并将这些信号直接转换成 12-bit 数字量。选择转换外部或内部模拟信号由 SAINS3~SAINS0 和 SACS3~SACS0 位段控制。注意若选择内部模拟信号，则所选中的外部输入通道将自动断开以避免出错。

关于 A/D 输入信号的更多详细信息请分别参见“A/D 转换器控制寄存器”和“A/D 转换器输入引脚”章节。

单片机型号	外部输入通道	内部输入信号	A/D 输入选择位
HT45F3420	AN0~AN3	V <sub>DD</sub> , V <sub>DD</sub> /2, V <sub>DD</sub> /4 V <sub>R</sub> , V <sub>R</sub> /2, V <sub>R</sub> /4	SAINS[3:0] SACS[3:0]
HT45F3430	AN0~AN7	OCPAO, OVP_AIN, VBAT_AIN	

下图显示了 A/D 转换器内部结构和相关的寄存器。



注：对于 HT45F3420，外部输入通道为 AN0~AN3

对于 HT45F3430，外部输入通道为 AN0~AN7

A/D 转换器结构

## A/D 转换器寄存器

A/D 转换器的所有工作由 5 个寄存器控制。一对只读寄存器用于存放 12-bit 的 A/D 转换值。其余三个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ENADC	ADRFS	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
SADC2	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0

**A/D 转换器寄存器列表**

### A/D 转换器数据寄存器 – SADOL, SADOH

对于具有 12-bit A/D 转换器的单片机，需要两个数据寄存器存放转换结果，一个高字节寄存器 SADOH 和一个低字节寄存器 SADOL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 12 位，其数据存储格式由 SADC0 寄存器的 ADRFS 位控制，如下表所示。D0~D11 是 A/D 换转数据结果位。未使用的位读为“0”。需注意的是，当 A/D 转换器除能时，数据寄存器的值将保持不变。

ADRFS	SADOH								SADOL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
1	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

**A/D 数据寄存器**

### A/D 转换器控制寄存器 – SADC0, SADC1, SADC2

寄存器 SADC0、SADC1 和 SADC2 用来控制 A/D 转换器的功能和操作。这些 8-bit 寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，控制并监测 A/D 转换器的开始和转换忙状态等功能。寄存器 SADC0 的 SACS3~SACS0 位定义 A/D 转换器外部输入通道编号。寄存器 SADC1 的 SAINS3~SAINS0 位用来决定模拟信号是来自内部模拟信号或外部模拟输入信号。

引脚共用控制寄存器 – PAS0 和 PAS1 中包含了相应的引脚共用选择位，用于决定 PA 端口上可用作 A/D 转换器输入的引脚。当引脚被选中作为 A/D 输入时，其原始功能需移除。此外，若引脚被选中作为 A/D 输入，引脚上所连接的上拉电阻将自动移除。

● SADC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ENADC	ADRFS	SACS3	SACS2	SACS1	SACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

- Bit 7      **START:** 启动 A/D 转换位  
               0 → 1 → 0 : 启动 A/D 转换  
               此位用于启动 A/D 转换过程。通常此位为低，但如果设为高再被清零，将启动 A/D 转换过程。
- Bit 6      **ADBZ:** A/D 转换忙标志位  
               0: A/D 转换结束或未开始转换  
               1: A/D 转换正在进行  
               该只读标志位用于表明 A/D 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已启动。A/D 转换结束后，此位被清零。
- Bit 5      **ENADC:** A/D 转换器功能使能控制位  
               0: A/D 转换器除能  
               1: A/D 转换器使能  
               此位控制 A/D 内部功能。该位被置高将使能 A/D 转换器。如果该位设为低将关闭 A/D 转换器以降低功耗。当 A/D 转换器功能除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将被不变。
- Bit 4      **ADRFS:** A/D 转换器数据格式选择位  
               0: A/D 转换器数据格式 → SADOH=D[11:4]; SADOL=D[3:0]  
               1: A/D 转换器数据格式 → SADOH=D[11:8]; SADOL=D[7:0]  
               此位控制存放在两个 A/D 数据寄存器中的 12-bit A/D 转换结果的格式。详情请参考 A/D 转换器数据寄存器章节。
- Bit 3~0     **SACS3~SACS0:** A/D 转换器输入通道选择  
               HT45F3420:  
               0000: A/D 转换器输入通道来自 AN0  
               0001: A/D 转换器输入通道来自 AN1  
               0010: A/D 转换器输入通道来自 AN2  
               0011: A/D 转换器输入通道来自 AN3  
               其它值: A/D 转换器输入浮空  
               HT45F3430:  
               0000: A/D 转换器输入通道来自 AN0  
               0001: A/D 转换器输入通道来自 AN1  
               0010: A/D 转换器输入通道来自 AN2  
               0011: A/D 转换器输入通道来自 AN3  
               0100: A/D 转换器输入通道来自 AN4  
               0101: A/D 转换器输入通道来自 AN5  
               0110: A/D 转换器输入通道来自 AN6  
               0111: A/D 转换器输入通道来自 AN7  
               其它值: A/D 转换器输入浮空

● SADC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
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 7~4      **SAINS3~SAINS0:** 内部 A/D 转换器输入信号选择

0000: A/D 转换器输入来自外部引脚

0001: A/D 转换器输入来自  $V_{DD}$

0010: A/D 转换器输入来自  $V_{DD}/2$

0011: A/D 转换器输入来自  $V_{DD}/4$

0100: A/D 转换器输入来自外部引脚

0101: A/D 转换器输入来自  $V_R$

0110: A/D 转换器输入来自  $V_R/2$

0111: A/D 转换器输入来自  $V_R/4$

1000: A/D 转换器输入来自 OCPAO

1001: A/D 转换器输入来自 OVP\_ADIN

1010: A/D 转换器输入来自 VBAT\_ADIN

1011: A/D 转换器输入接地 (未使用)

其它值: A/D 转换器输入来自外部引脚

注:  $V_R$  为 PGA 输出电压。OCPAO 为 OCP 电路的 OPA 输出。VBAT\_ADIN 和 OVP\_ADIN 来自集成电压分压器电路, 详情可参考“集成电压分压器电路”章节。

Bit 3      未定义, 读为 “0”

Bit 2~0      **SACKS2~SACKS0:** A/D 转换器时钟源选择

000:  $f_{SYS}$

001:  $f_{SYS}/2$

010:  $f_{SYS}/4$

011:  $f_{SYS}/8$

100:  $f_{SYS}/16$

101:  $f_{SYS}/32$

110:  $f_{SYS}/64$

111:  $f_{SYS}/128$

### ● SADC2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0
R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	—	—	0	0	0	0	0

Bit 7      **ADPGAEN:** PGA 使能 / 除能控制位

- 0: 除能
- 1: 使能

注: 当 PGA 输出  $V_R$  被选为 A/D 转换器输入或 A/D 转换器参考电压, 需通过设置此位为高使能 PGA。否则 PGA 需通过清除 ADPGAEN 位除能以节省功耗。由于 PGA 输出也可用作 OCP 和 OVP DAC 的参考输入, 若被以上功能选中 PGA 必须预先使能。

Bit 6~5      未定义, 读为“0”

Bit 4      **PGAIS:** PGA 输入 ( $V_{RI}$ ) 选择位

- 0:  $V_{RI}$  只来自 VREFI 引脚
- 1:  $V_{RI}$  只来自  $V_{BG}$

Bit 3~2      **SAVRS1~SAVRS0:** A/D 转换器参考电压选择

- 00: A/D 转换器参考电压来自  $V_{DD}$
- 01: A/D 转换器参考电压来自 VREF 引脚
- 1x: A/D 转换器参考电压来自  $V_R$ (PGA 输出)

Bit 1~0      **PGAGS1-PGAGS0:** PGA 增益选择

- 00: Gain=1
- 01: Gain=2
- 10: Gain=3
- 11: Gain=4

### A/D 转换器操作

START 位用于 A/D 转换器的启动与复位。当单片机设定此位从逻辑低到逻辑高, 然后再到逻辑低, 就会开始一个模数转换周期。当 START 位由低转高后未被再次置低, SADC0 寄存器中的 ADBZ 位将被清零, A/D 转换器复位。START 位用于控制内部 A/D 转换器的所有启动操作。

SADC0 寄存器中的 ADBZ 位用于表明模数转换过程是否完成。当设置 START 位由低转高, A/D 转换器将复位, 此时 ADBZ 将清零。A/D 转换成功启动后, ADBZ 位被自动置为“1”。在转换周期结束后, ADBZ 位会被清零。此外, 中断控制寄存器内相应的 A/D 转换器中断请求标志位也会置位, 如果中断使能, 就会产生对应的内部中断信号。A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口。如果 A/D 内部中断被禁止, 可以让单片机轮询 SADC0 寄存器中的 ADBZ 位, 检测此位是否被清除, 以作为另一种侦测 A/D 转换周期结束的方法。

虽然 A/D 时钟源是由系统时钟  $f_{SYS}$  和 SACKS2~SACKS0 位段决定, 但可选择的最大 A/D 时钟源还有一些限制。由于允许的 A/D 时钟周期  $t_{ADCK}$  的范围为  $0.5\mu s \sim 10\mu s$ , 所以选择系统时钟速度时就必须小心。如果系统时钟速度为 4MHz 时, SACKS2~SACKS0 位段不能设为“000”或“11x”。否则所得的 A/D 时钟周期将小于最小时钟周期或大于最大时钟周期, 导致所得 A/D 转换结果不准确。

f <sub>sys</sub>	A/D 时钟周期 (t <sub>ADCK</sub> )							
	SACKS [2:0]=000 (f <sub>sys</sub> )	SACKS [2:0]=001 (f <sub>sys</sub> /2)	SACKS [2:0]=010 (f <sub>sys</sub> /4)	SACKS [2:0]=011 (f <sub>sys</sub> /8)	SACKS [2:0]=100 (f <sub>sys</sub> /16)	SACKS [2:0]=101 (f <sub>sys</sub> /32)	SACKS [2:0]=110 (f <sub>sys</sub> /64)	SACKS [2:0]=111 (f <sub>sys</sub> /128)
1MHz	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *	128μs *
2MHz	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *
4MHz	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *
8MHz	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *

### A/D 时钟周期范例

SADC0 寄存器的 ENADC 位用于控制 A/D 转换电路电源的开启 / 关闭。该位必须置高以开启 A/D 转换器电源。当设置 ENADC 位为高开启 A/D 转换器内部电路时，在 A/D 转换成功初始化前需一段延时。如时序图所示。即使通过设置相关引脚共用控制位，选择无引脚作为 A/D 输入，如果 ENADC 设为“1”，那么仍然会产生功耗。因此当未使用 A/D 转换器功能时，在功耗敏感的应用中建议设置 ENADC 为低以减少功耗。

A/D 转换器的参考电压可以来自正电源引脚 VDD 或来自 VREF 引脚上的外部参考源或是 V<sub>BG</sub> 电压，通过配置 SAVRS1~SAVRS0 位段进行选择。由于 VREF 引脚与其它功能共用，当 VREF 引脚被选择参考电压提供引脚时，VREF 引脚共用功能控制位应适当配置以除能其它引脚功能。当 V<sub>R</sub> 被选作 A/D 转换器参考电压，应将 ADPGAEN 位置高使能 PGA。

### A/D 转换器输入信号

A/D 转换器的所有模拟输入引脚与 PA 端口的 I/O 引脚及其它功能共用。PAS0 和 PAS1 寄存器中每个引脚相应的引脚共用功能选择位决定了输入引脚是设置为 A/D 转换器模拟通道输入还是其它功能。如果相应引脚设置为 A/D 转换器模拟通道输入那么该引脚原引脚功能除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器编程设置的所有上拉电阻会自动断开。请注意，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当引脚共用控制位使能 A/D 输入时，端口控制寄存器的状态将被重置。

通过配置 SAINS3~SAINS0 位，可选择几个内部模拟信号 (Bandgap 参考电压 V<sub>BG</sub>、V<sub>DD</sub>、PGA 输出 V<sub>R</sub>、OCP 输出 OCPAO 或是集成电压分压电路所产生的 VBAT\_ANIN 和 OVP\_ANIN 信号) 连接到 A/D 转换器。若 SAINS3~SAINS0 位段设为“0000”，则所要转换信号为外部模拟通道信号，SACS3~SACS0 位段可决定选择哪个外部通道的信号进行转换。若 SAINS3~SAINS0 位段设置为“0001~0011”，则选择 V<sub>DD</sub> 电压进行转换。若 SAINS3~SAINS0 设置为“0101~0111”，则选择 PGA 输出电压进行转换。

A/D 转换器有自己的参考电压引脚，VREF，但参考电压也可来自电源供电引脚，这可通过 SADC2 寄存器中的 SAVRS[1:0] 位段进行选择。模拟输入值必须小于所选的 A/D 参考电压。A/D 转换器还有一个 VREFI 引脚，该引脚为 PGA 的其中一个输入，可作为 A/D 转换器参考电压。要选择此 PGA 输入信号，必须将 PGAIIS 位清零且适当配置相关的引脚共用控制位。

注意，当程序同时选中外部信号 (HT45F3420 的 AN0~AN3, HT45F3430 的 AN0~AN7) 和内部信号 ( $V_{DD}$ 、 $V_{DD}/2$ 、 $V_{DD}/4$ 、 $V_R$ 、 $V_R/2$ 、 $V_R/4$ 、OVPAO、VBAT\_ADIN 或 OVP\_ADIN) 作为 A/D 转换器输入信号，此时硬件只选择内部信号作为 A/D 转换器输入。此外。若程序选择外部参考信号  $V_{REF}$  和内部参考信号  $V_{DD}$  或  $V_R$  作为 A/D 参考电压，此时硬件将自动选择内部参考电压作为 A/D 参考电压。同样的硬件控制方法也可应用于 PGA 输入选择。若程序同时选择外部电压  $V_{REFI}$  和内部电压  $V_{BG}$  作为 PGA 输入，此时硬件将选择内部电压  $V_{BG}$  作为 PGA 输入。

SAINS[3:0]	SACS[3:0]	输入信号	描述
0000,0100, 1100~1111	0000~0011	AN0~AN3	外部引脚模拟输入
	0000~0111	AN0~AN7	
0001	xxxx	$V_{DD}$	A/D 转换器电源供电电压
0010	xxxx	$V_{DD}/2$	A/D 转换器电源供电电压 /2
0011	xxxx	$V_{DD}/4$	A/D 转换器电源供电电压 /4
0101	xxxx	$V_R$	内部参考电压
0110	xxxx	$V_R/2$	内部参考电压 /2
0111	xxxx	$V_R/4$	内部参考电压 /4
1000	xxxx	OCPAO	OCP 电路 OPA 输出
1001	xxxx	OVP_ADIN	来自集成电压分压电路
1010	xxxx	VBAT_ADIN	来自集成电压分压电路
1011	xxxx	Ground	未使用

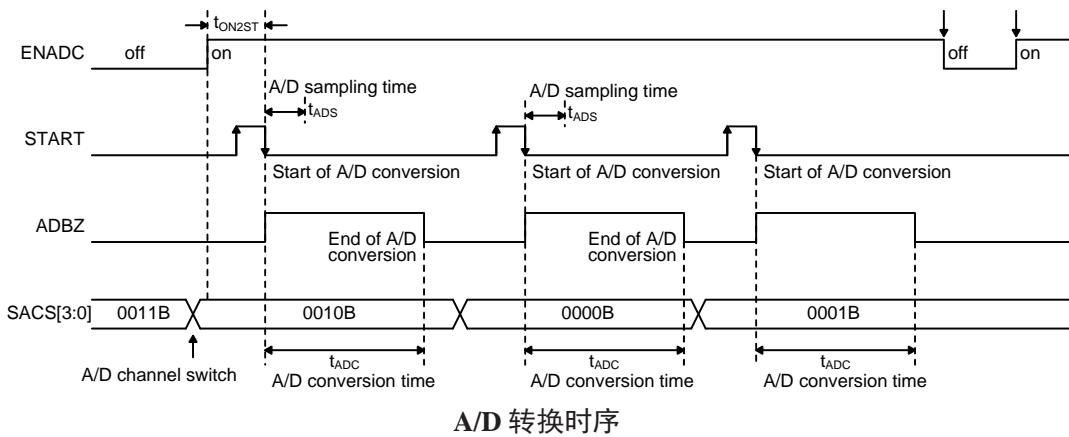
#### A/D 转换器输入信号选择

#### A/D 转换率与时序图

一个完整的 A/D 转换包含两部分，数据采样和数据转换。数据采样时间定义为  $t_{ADS}$ ，需 4 个 A/D 时钟周期，而数据转换需 12 个 A/D 时钟周期。所以一个完整的 A/D 转换时间  $t_{ADC}$  一共需要 16 个 A/D 时钟周期。

$$\text{最大 A/D 转换率} = \text{A/D 时钟周期} / 16$$

但在当前转换结束后，下个 A/D 转换开始前有一个使用限制。当前 A/D 转换结束，转换后的数字信号将存储在 A/D 数据寄存器对中并在半个 A/D 时钟周期后锁存。若 START 位在 A/D 转换结束的半个 A/D 时钟周期内置高，存入 A/D 数据寄存器对中的转换值将被改变。因此，建议在当前 A/D 转换结束后等待时间需大于半个 A/D 时钟周期才可以开始下一轮转换。



A/D 转换时序

### A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1  
通过 SACKS2~SACKS0 位选择所需的 A/D 转换器转换时钟。
- 步骤 2  
将 ENADC 位置高以使能 A/D 转换器。
- 步骤 3  
选择作为 A/D 转换器模拟输入的引脚。
- 步骤 4  
若输入来自 I/O 端口，设置 SAINS[3:0]=0000 且设置 SACS[3:0] 选中相应的 PAD 输入  
若输入来自内部输入，设置 SAINS[3:0] 选中相应的内部输入源
- 步骤 5  
设置 SAVRS[1:0] 位选择参考电压源为外部  $V_{REF}$  或来自  $V_{DD}$  或  $V_{BG}$ 。
- 步骤 6  
设置 ADRFS 选择 A/D 转换器输出数据格式。
- 步骤 7  
若使用 A/D 转换器中断，则中断控制寄存器必须正确配置以确保 A/D 转换器中断功能有效。总中断控制位 EMI 和 A/D 转换器中断位 ADE 都应预先置高。
- 步骤 8  
现在可以通过设定 START 位从低到高再到低，开始 A/D 转换的过程。
- 步骤 9  
若 A/D 转换正在进行，ADBZ 标志将置高。A/D 转换结束后，ADBZ 标志置低，并可从 SADOH 和 SADOL 寄存器读取输出数据。若 A/D 转换器中断使能且中断未满，可通过中断服务程序获取数据。另外可通过轮询 ADBZ 标志位以获取 A/D 转换器输出数据。

### 编程注意事项

在单片机工作时，如果 A/D 转换器未使用，通过设置 SADC0 寄存器中的 ENADC 为低，关闭 A/D 内部电路以减少电源功耗。此时，不考虑输入脚的模拟电压，内部 A/D 转换器电路不产生功耗。如果 A/D 转换器输入脚用作普通 I/O 脚，必须特别注意，输入电压为无效逻辑电平也可能增加功耗。

## A/D 转换功能

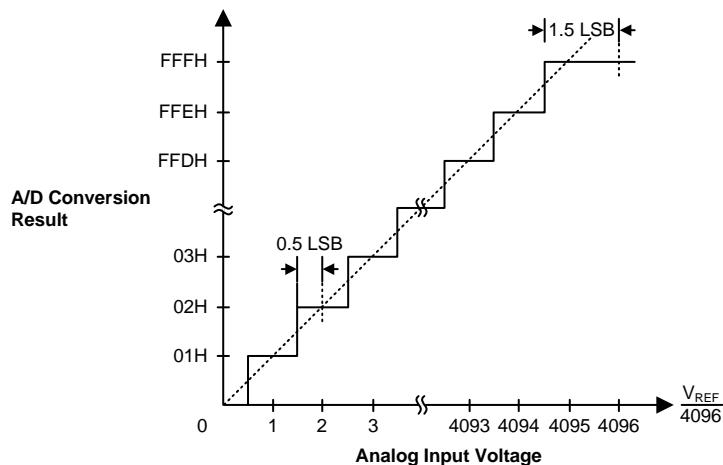
该系列单片机均含有一个 12-bit A/D 转换器，其转换的最大值可达 FFFF。由于模拟输入最大值等于实际 A/D 转换器参考电压  $V_{REF}$  (可选择来自  $V_{DD}$ 、 $V_{REF}$  引脚电压或  $V_R$ )，因此每一位可表示为  $V_{REF}/4096$  的模拟输入值。

$$1 \text{ LSB} = V_{REF}/4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

$$\text{A/D 输入电压} = \text{A/D 数字输出值} \times (V_{REF}/4096)$$

下图显示 A/D 转换器模拟输入值和数字输出值之间理想的转换功能。除了数字化数值 0，其后的数字化数值会在精确点之前的 0.5 LSB 处改变，而数字化数值的最大值将在  $V_{REF}$  之前的 1.5 LSB 处改变。



理想的 A/D 转换功能

## A/D 转换程序范例

下面两个范例程序用来说明怎样设置和实现 A/D 转换。第一个范例是轮询 SADC0 寄存器中的 ADBZ 位来判断 A/D 转换何时完成，而第二个范例则使用中断的方式判断。

**范例 1：使用 EOCB 轮询方式来检测转换结束**

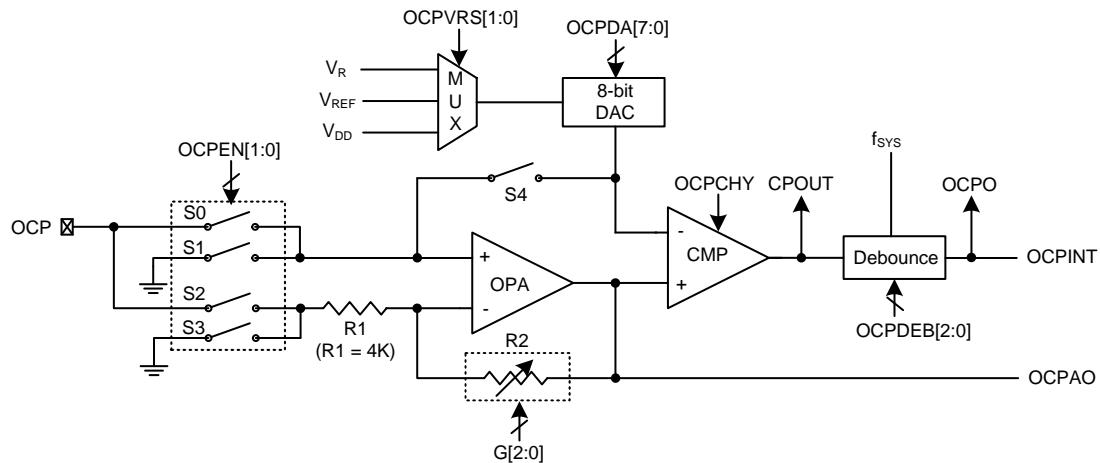
```
clr ADE           ; disable ADC interrupt
mov a, 03H
mov SADC1, a      ; select fSYS/8 as A/D clock
set ENADC
mov a, 03H
            ; setup PAS0 to configure pin AN0
mov PAS0, a
mov a, 20H
mov SADC0, a      ; enable and connect AN0 channel to the A/D
            ; converter
:
start_conversion:
clr START          ; high pulse on start bit to initiate conversion
set START          ; reset A/D
clr START          ; start A/D
:
polling_EOC:
sz ADBZ           ; poll the SADC0 register ADBZ bit to detect end
            ; of A/D conversion
jmp polling_EOC
mov a, SADOL
mov SADOL_buffer, a
mov a, SADOH
mov SADOH_buffer, a
:
jmp start_conversion ; start next A/D conversion
```

**范例 2：使用中断的方式来检测转换结束**

```
clr ADE           ; disable ADC interrupt
mov a, 03H
mov SADC1, a      ; select fSYS/8 as A/D clock
set ENADC
mov a, 03H
mov PAS0,a        ; setup PAS0 to configure pin AN0
mov a, 20H
mov SADC0, a      ; enable and connect AN0 to A/D convertor
:
start_conversion:
clr START          ; high pulse on START bit to initiate conversion
set START          ; reset A/D
clr START          ; start A/D
clr ADF           ; clear ADC interrupt request flag
set ADE           ; enable ADC interrupt
set EMI           ; enable global interrupt
:
:
ADC_ISR:
mov acc_stack, a  ; save ACC to user defined memory
mov a, STATUS      ; save STATUS to user defined memory
:
mov a, SADOL      ; read low byte conversion result value
mov SADOL_buffer, a ; save result to user defined register
mov a, SADOH      ; read high byte conversion result value
mov SADOH_buffer, a ; save result to user defined register
:
EXIT_INT_ISR:
mov a, status_stack ; restore STATUS from user defined memory
mov STATUS, a       ; restore ACC from user defined memory
reti
```

## 过电流保护 – OCP

该系列单片机内建了过电流保护功能，为应用提供了一个过电流保护机制。为预防电池充电或负载电流超过指定值，使用 OCP 运算放大器将 OCP 引脚上的电流根据电流值转换为相关的电压电平值，接着与一个 8-bit D/A 转换器产生的参考电压进行比较。OCPCHY 引脚上的电压与 8-bit D/A 转换器产生的参考电压相比较。当出现过电流事件时，若相应的中断控制使能，将产生一个 OCP 中断。



注：1.  $V_R$  为 ADC PGA 输出信号

2. 对于 HT45F3430 单片机，OCP 输入源引脚可为 PA1 或 PA6，由 IFS0 寄存器中的 IFS03 位确定。

### 过电流保护电路

#### OCP 操作

OCP 电路用于保护输入电流使其不超过参考值。OCP 引脚上的电流被转换为电压，并由 OCP 运算放大器放大，可由 OCPC1 寄存器中的 G2~G0 位选择 1~50 的增益。这便是所谓的可编程增益运算放大器 PGA。此 PGA 也可通过配置工作在同相，反相或输入失调校准模式，这由 OCPC0 寄存器中的 OCPEN1 和 OCPEN0 位段所决定。电流经转换并放大为指定电压电平值后，将与 8-bit D/A 转换器提供的参考电压相比较。8-bit D/A 转换器的电源可以由  $V_{DD}$ 、 $V_{REF}$  或  $V_R$  提供，这由 OCPC0 寄存器中的 OCPVRS[1:0] 位段决定。比较器输出 CPOUT 将先经过一定去抖动时间周期的滤波，去抖动时钟周期由 OCPC1 寄存器中的 OCPDEB2~OCPDEB0 位段选择。所获得的滤波后的 OCP 数字比较器输出 OCPO 表明了是否出现过电流情况。若出现过电流情况则 OCPO 将被置高，否则此位为零。一旦出现过电压事件，即 OCP 输入电流转换得到的电压大于参考电压，若相关中断控制位使能，将产生相应的中断。

#### OCP 控制寄存器

OCP 功能的所有操作由一系列的寄存器控制。一个寄存器用于为 OCP 电路提供参考电压。两个寄存器用于运算放大器和比较器输入失调校准。剩下的两个控制寄存器用于控制 OCP 功能，D/A 转换器参考电压选择，PGA 增益选择，比较器去抖动时间以及迟滞功能。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
OCPC0	OCOPEN1	OCOPEN0	OCPVRS1	OCPVRS0	OCPCHY	—	—	OCPO
OCPC1	—	—	G2	G1	G0	OCPDEB2	OCPDEB1	OCPDEB0
OCPDA	D7	D6	D5	D4	D3	D2	D1	D0
OCPOCAL	OOFM	ORSP	OOF5	OOF4	OOF3	OOF2	OOF1	OOF0
OCPCCAL	CPOUT	COFM	CRSP	COF4	COF3	COF2	COF1	COF0

### OCP 控制寄存器列表

#### OCPC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCOPEN1	OCOPEN0	OCPVRS1	OCPVRS0	OCPCHY	—	—	OCPO
R/W	R/W	R/W	R/W	R/W	R/W	—	—	R
POR	0	0	0	0	0	—	—	0

Bit 7~6

#### OCOPEN1~OCOPEN0: OCP 工作模式选择

- 00: OCP 除能, S1 和 S3 on, S0 和 S2 off
- 01: 同相模式, S0 和 S3 on, S1 和 S2 off
- 10: 反相模式, S1 和 S2 on, S0 和 S3 off
- 11: 校准模式, S1 和 S3 on, S0 和 S2 off

Bit 5~4

#### OCPVRS1~OCPVRS0: OCP DAC 参考电压选择

- 00: 来自 V<sub>DD</sub>
- 01: 来自 V<sub>REF</sub>
- 1x: 来自 V<sub>R</sub> (ADC PGA 输出)

Bit 3

#### OCPCHY: OCP 比较器迟滞功能控制位

- 0: 除能
- 1: 使能

Bit 2~1

未定义, 读为“0”

Bit 0

#### OCPO: OCP 数字输出位

- 0: 监测电流源未结束
- 1: 监测电流源结束

#### OCPC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	G2	G1	G0	OCPDEB2	OCPDEB1	OCPDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6

未定义, 读为“0”

Bit 5~3

#### G2~G0: PGA R2/R1 比例选择

- 000: 单位增益缓冲器 (同相模式) 或 R2/R1=1 (反相模式)
- 001: R2/R1=5
- 010: R2/R1=10
- 011: R2/R1=15
- 100: R2/R1=20
- 101: R2/R1=30
- 110: R2/R1=40
- 111: R2/R1=50

这三位用于选择 R2/R1 比, 为反相和同相模式选择多种增益。反相和同相模式中的 PGA 增益计算公式请参见“输入电压范围”章节。

Bit 2~0      **OCPDEB2~OCPDEB0:** OCP 输出滤波去抖动时间选择

- 000: 旁路, 无去抖动
- 001:  $(1\sim2) \times t_{DEB}$
- 010:  $(3\sim4) \times t_{DEB}$
- 011:  $(7\sim8) \times t_{DEB}$
- 100:  $(15\sim16) \times t_{DEB}$
- 101:  $(31\sim32) \times t_{DEB}$
- 110:  $(63\sim64) \times t_{DEB}$
- 111:  $(127\sim128) \times t_{DEB}$

注:  $t_{DEB}=1/f_{SYS}$

### OCPDA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0      OCP DAC 输出电压控制位

OCP DAC 输出 =  $(DAC \text{ 参考电压} / 256) \times D[7:0]$

### OCPOCAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OOFM	ORSP	OOF5	OOF4	OOF3	OOF2	OOF1	OOF0
R/W								
POR	0	0	1	0	0	0	0	0

Bit 7      **OOFM:** OCP 运算放大器输入失调校准模式使能控制位

- 0: 输入失调校准模式除能
- 1: 输入失调校准模式使能

此位用于控制 OCP 运算放大器输入偏置校准功能。OCPEN1 和 OCPEN0 位必须设置为“11”且 OOFM 位设为“1”随后 COFM 位设置为“0”才能使能运算放大器输入失调校准模式。关于失调校准过程的更多详情请参见“运算放大器输入失调校准”章节。

Bit 6      **ORSP:** OCP 运算放大器输入失调电压校准参考选择位

- 0: 选择负输入作为参考输入
- 1: 选择正输入作为参考输入

Bit 5~0      **OOF5~OOF0:** OCP 运算放大器输入失调电压校准值

此 6 位段用于执行运算放大器输入失调校准操作，操作的值可以重新存储到这个位段。更多详情请参见“运算放大器输入失调校准”章节。

### OCPCCAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CPOUT	COFM	CRSP	COF4	COF3	COF2	COF1	COF0
R/W	R	R/W						
POR	0	0	0	1	0	0	0	0

Bit 7      **CPOUT:** OCP 比较器输出, 正逻辑 (只读)

- 0: 正输入电压 < 负输入电压
- 1: 正输入电压 > 负输入电压

此位用于在 OCP 工作于输入失调校准模式时显示正输入电压是否大于负输入电压。若 CPOUT 设为“1”，表示正输入电压大于负输入电压，否则正输入电压小于负输入电压。

Bit 6	<b>COFM:</b> OCP 比较器输入失调校准模式使能控制位 0: 输入失调校准模式除能 1: 输入失调校准模式使能 此位用于控制 OCP 比较器输入失调校准功能。OCPEN1 和 OCPEN0 的值应设为“11”且 COFM 位必须设为“1”随即 OOFM 位设为“0”，此时比较器输入失调校准模式使能。关于失调校准过程的更多详情请参见“比较器输入失调校准”章节。
Bit 5	<b>CRSP:</b> OCP 比较器输入失调校准参考输入选择位 0: 选择负输入作为参考输入 1: 选择正输入作为参考输入
Bit 4~0	<b>COF4~COF0:</b> OCP 比较器输入失调校准值 此 5 位位段用于执行比较器输入失调校准操作。操作的值可以重新存入该位段。更多详情请参见“比较器输入失调校准”章节。

## 输入电压范围

为了操作的灵活性，在不同的 PGA 操作模式下，OCP 引脚上的输入电压可以为正或者为负。正输入或负输入电压的 PGA 输出基于不同的公式计算如下：

- 输入电压  $V_{IN} > 0$  时，同相模式下的 PGA 和 PGA 输出可由以下公式获得：

$$V_{OUT} = (1 + R2/R1) \times V_{IN}$$

- 将 OCPEN[1:0] 的值设为“01”使 PGA 工作在同相模式，并设置 G[2:0] 的值为“000”以选择单位增益，此时 PGA 将作为一个单位增益缓存器，其输出与  $V_{IN}$  相等。

$$V_{OUT} = V_{IN}$$

- 输入电压  $0 > V_{IN} > -0.4V$  时，工作在反相模式下的 PGA 以及 PGA 输出可由以下公式获得。注意，若输入电压为负，其值不可小于  $-0.4V$ ，否则会漏电流。

$$V_{OUT} = -(R2/R1) \times V_{IN}$$

## 失调校准

OCP 电路通过 OCPEN[1:0] 位段控制四种操作模式，其中一种就是校准模式。在此模式下，运算放大器和比较器失调可以得到校准。

### 运算放大器输入失调校准

步骤 1. 设置 OCPEN[1:0]=11, OOFM=1 且 COFM=0, ORSP=1, OCP 将工作在运算放大器输入失调校准模式。

步骤 2. 设置 OOF[5:0]=000000，此时开始读 CPOUT 位。

步骤 3. OOF[5:0] 的值加一，读 CPOUT 位。  
若 CPOUT 位状态不变，则重复步骤 3 直到 CPOUT 位状态改变。

若 CPOUT 位状态改变，则记录下 OOF 值为 VOOS1，前往步骤 4。

步骤 4. 设置 OOF[5:0]=111111，此时开始读 CPOUT 位。

步骤 5. OOF[5:0] 的值减一，读 CPOUT 位。  
若 CPOUT 位状态未改变，重复步骤 5 直到 CPOUT 位状态改变。

若 CPOUT 位状态改变，则记录下 OOF 值为 VOOS2，前往步骤 6。

步骤 6. 将运算放大器输入失调校准值 VOOS，重新存入 OOF[5:0] 位段。此时失调校准步骤完成。  
这里的 VOOS=(VOOS1 + VOOS2) / 2

注：S4 为 OFF，在此模式下，OPAMP 输出为 CPOUT，直接 bypass 比较器。

### 比较器输入失调校准

步骤 1. 设置 OCPEN[1:0]=11, COFM=1 且 OOFM=0, OCP 将工作在比较器输入失调校准模式, S4 为 ON。

步骤 2. 设置 COF[4:0]=00000, 此时开始读 CPOUT 位。

步骤 3. COF[4:0] 的值加一, 读 CPOUT 位。

若 CPOUT 位状态不变, 则重复步骤 3 直到 CPOUT 位状态改变。

若 CPOUT 位状态改变, 则记录下 COF 值为 VCOS1, 前往步骤 4。

步骤 4. 设置 COF[4:0]=11111, 此时开始读 CPOUT 位。

步骤 5. COF[4:0] 的值减一, 读 CPOUT 位。

若 CPOUT 位状态未改变, 重复步骤 5 直到 CPOUT 位状态改变。

若 CPOUT 位状态改变, 则记录下 COF 值为 VCOS2, 前往步骤 6.

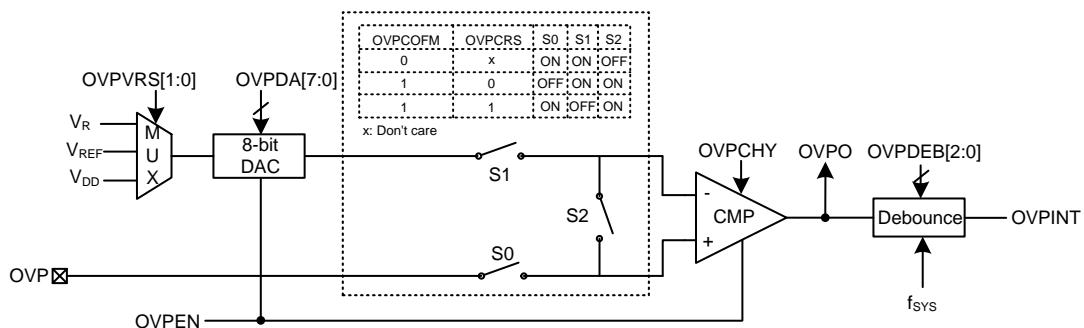
步骤 6. 将运算放大器输入失调校准值 VCOS, 重新存入 COF[4:0] 位段。此时失调校准步骤完成。

这里的  $VCOS=(VCOS1 + VCOS2) / 2$

注: S4 为 on 且 D/A 转换器关闭这种情况只适用于比较器校准步骤。在正常操作中, S4 为 off。

## 过电压保护 – OVP

该系列单片机具有过电压保护功能, 为应用提供了一个保护机制。将 OVP 引脚上的电压与一个 8-bit D/A 转换器产生的参考电压相比较, 以避免工作电压超出指定电压值。当出现过电压的情况, 在相应的中断控制使能的情况下将产生 OVP 中断。



注: 1.  $V_R$  为 ADC PGA 输出信号。

2. OVP 引脚信号可为 OVP\_IN1 或 OVP\_IN2, 由 IFS0 寄存器中的 IFS02 位确定。

过电压保护电路

### OVP 操作

电压源供给 OVP 引脚, 并连接至比较器的其中一个输入。D/A 转换器用于产生一个参考电压。比较器将参考电压和输入电压进行比较产生 OVPO 信号。

### OVP 控制寄存器

过电压保护的所有操作是通过一系列寄存器来控制的。一个寄存器用于为过电压保护电路提供参考电压。剩下的两个控制寄存器用于控制 OVP 功能、D/A 转换器参考电压选择、比较器去抖时间、比较器迟滞功能和比较器输入失调校准等功能。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
OVPC0	—	—	OVOPEN	OVPCHY	OVPVRS1	OVPVRS0	OVPDEB1	OVPDEB0
OVPC1	OVPO	OVPCOFM	OVPCRS	OVPCOF4	OVPCOF3	OVPCOF2	OVPCOF1	OVPCOF0
OVPDA	D7	D6	D5	D4	D3	D2	D1	D0

### OVP 控制寄存器列表

#### OVPC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	OVOPEN	OVPCHY	OVPVRS1	OVPVRS0	OVPDEB1	OVPDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **OVOPEN:** OVP 功能控制位

0: 除能

1: 使能

若 OVOPEN 位清零，过电压保护功能将除能以节省功耗。此时 OVP 中的比较器和 D/A 转换器也全部关闭。

Bit 4 **OVPCHY:** OVP 比较器迟滞功能控制位

0: 除能

1: 使能

Bit 3~2 **OVPVRS1~OVPVRS0:** OVP DAC 参考电压选择

00: DAC 参考电压来自 V<sub>DD</sub>

01: DAC 参考电压来自 V<sub>REF</sub>

1x: DAC 参考电压来自 V<sub>R</sub> (ADC PGA 输出)

Bit 1~0 **OVPDEB1~OVPDEB0:** OVP 比较器去抖动时间控制

00: 无去抖动时间

01: (7~8)×1/f<sub>SYS</sub>

10: (15~16)×1/f<sub>SYS</sub>

11: (31~32)×1/f<sub>SYS</sub>

#### OVPC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OVPO	OVPCOFM	OVPCRS	OVPCOF4	OVPCOF3	OVPCOF2	OVPCOF1	OVPCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **OVPO:** OVP 比较器输出位

0: 正输入电压 < 负输入电压

1: 正输入电压 > 负输入电压

Bit 6 **OVPCOFM:** OVP 比较器正常工作模式或输入失调电压校准模式选择位

0: 正常工作模式

1: 输入失调电压校准模式

Bit 5 **OVPCRS:** OVP 比较器输入失调校准参考输入选择位

0: 选择负输入作为参考输入

1: 选择正输入作为参考输入

Bit 4~0 **OVPCOF4~OVPCOF0:** OVP 比较器输入失调电压校准控制

### OVPDA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0 OVP DAC 输出电压控制位  
 $DAC\ V_{OUT} = (DAC\ 参考电压 / 256) \times D[7:0]$

### 失调校准

OVPC1 寄存器中的 OVPCOFM 位用于选择 OVP 比较器工作模式，正常工作模式或失调校准模式。若设置此位为高，比较器将进入失调电压校准模式。需注意的是在失调校准之前，需设 OVPCHY=0 使得迟滞电压为“0”，且由于 OVP 引脚与 I/O 口共用，需注意将引脚预置为比较器输入。

### 比较器校准步骤

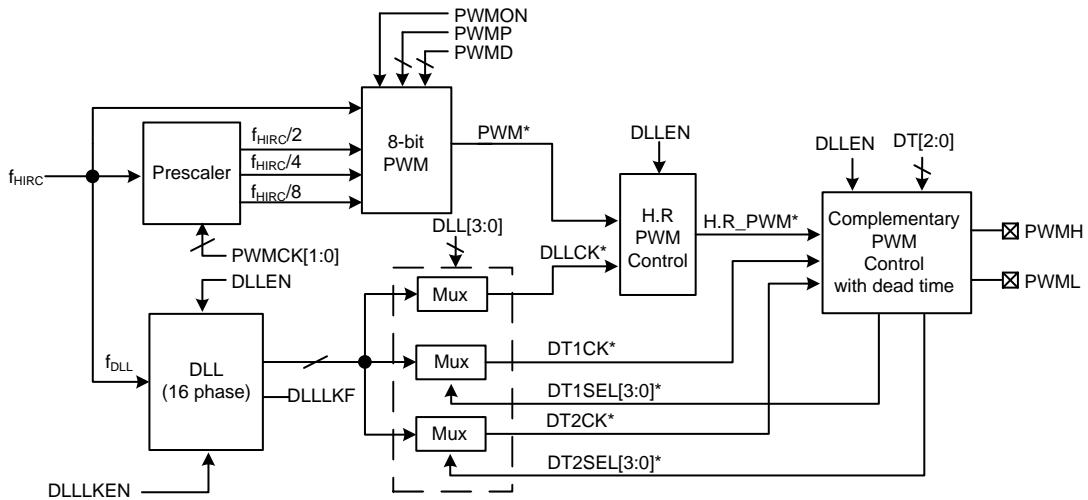
- 步骤 1. 设置 OVPCOFM=1, OVPCRS=1, OVP 将工作在比较器输入失调校准模式，S0 和 S2 闭合。为保证  $V_{os}$  在校准后尽可能小，校准中的输入参考电压应与正常工作模式时的输入直流工作电压相同。
- 步骤 2. 设置 OVPCOF[4:0]=00000，此时开始读 OVPO 位。
- 步骤 3. OVPCOF[4:0]=OVPCOF[4:0]+1，读 OVPO 位。若 OVPO 位状态改变，则记录下 OVPCOF[4:0] 为  $V_{os1}$ 。
- 步骤 4. 设置 OVPCOF[4:0]=11111，此时开始读 OVPO 位。
- 步骤 5. OVPCOF[4:0]=OVPCOF[4:0]-1，读 OVPO 位。若 OVPO 位状态改变，则记录下 OVPCOF[4:0] 为  $V_{os2}$ 。
- 步骤 6. 将  $V_{os}=(V_{os1}+V_{os2})/2$  重新存入 OVPCOF[4:0] 位段。此时失调校准步骤完成。  
 若  $(V_{os1}+V_{os2})/2$  不是整数则忽略小数，保留  $V_{os}=V_{OUT} - V_{IN}$

### 高精度互补输出 PWM 发生器

该系列单片机包含一个具有互补输出的多功能高度集成 PWM 发生器，极大地增加了应用的灵活性。

### 功能性描述

高精度 8-bit PWM 发生器电路包括一个延迟锁相环电路和带死区时间插入的 PWM 互补输出。



注：1. “\*”为内部信号名而非特殊功能寄存器位。

DT1SEL[3:0] 和 DT2SEL[3:0] 基于 DT[2:0] 自动进行计算与选择

DT1CK 为 PWMH DT 参考信号

DT2CK 为 PWML DT 参考信号

DLLCK 为 H.R\_PWM 参考信号

2. H.R = 高精度

### H.R PWM 输出方框图

#### 高精度 PWM 寄存器

高精度 PWM 的所有操作由五个寄存器控制。一个 PWM 周期寄存器 PWMP 用于存储所需 8-bit PWM 周期值。PWM 占空比值存于一个 8-bit PWMD 寄存器中。 PWM 功能控制、PWM 计数器时钟选择、DLL 电路与死区持续时间由 CPR 寄存器决定。寄存器 DLL 用于 DLL 电路相位选择。剩下的寄存器 CPOR 用于 PWM 设置、互补输出控制、保护及反相控制。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PWMP	D7	D6	D5	D4	D3	D2	D1	D0
PWMD	D7	D6	D5	D4	D3	D2	D1	D0
DLL	DLL3	DLL2	DLL1	DLL0	—	—	—	DLLKF
CPR	DLLKEN	DLEN	PWMCK1	PWMCK0	PWMON	DT2	DT1	DT0
CPOR	PWMACT	CNVTYP	INVH	INVL	OCPHEN	OCPLEN	OVPHEN	OVPLEN

#### 高精度 PWM 发生器寄存器列表

##### PWMP 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0      D7~D0: 8-bit PWM 周期寄存器

PWM 周期 = PWMP[7:0] +1

### PWMD 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
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 7~0

**D7~D0:** 8-bit PWM 占空比寄存器

PWMP 与 PWMD 两个寄存器用于 8-bit PWM 周期与占空比控制。在设置过程中应注意下面几点：

1. PWMD 的值应符合此条件:  $1 \leq \text{PWMD} \leq (\text{PWMP} - 1)$
2. 当  $\text{DLL}[3:0]=0000\text{B}$ ,  $\text{DT}[2:0]=111\text{B}$ , PWM 占空比 (Min.)= $1 + \text{DLL}[3:0] - \text{DT}[2:0]$
3. 当  $\text{DLL}[3:0]=1111\text{B}$ ,  $\text{DT}[2:0]=000\text{B}$ , PWM 占空比 (Max.)= $\text{PWMP}-1 + \text{DLL}[3:0] - \text{DT}[2:0]$

### DLL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DLL3	DLL2	DLL1	DLL0	—	—	—	DLLLKF
R/W	R/W	R/W	R/W	R/W	—	—	—	R/W
POR	0	0	0	0	—	—	—	0

Bit 7~4

**DLL3~DLL0:** DLL 相位选择

0000: 将 H.R\_PWM 占空比下降沿微调至 DLL 相位 #0 上升沿

0001: 将 H.R\_PWM 占空比下降沿微调至 DLL 相位 #1 上升沿

0010: 将 H.R\_PWM 占空比下降沿微调至 DLL 相位 #2 上升沿

...

1110: 将 H.R\_PWM 占空比下降沿微调至 DLL 相位 #14 上升沿

1111: 将 H.R\_PWM 占空比下降沿微调至 DLL 相位 #15 上升沿

Bit 3~1

未定义, 读为 “0”

Bit 0

**DLLLKF:** DLL 电路失锁标志位

0: 未发生失锁现象

1: 发生失锁现象

此位可由软件清零, 但无法被置高。

注: 当 DLLLKEN=1 且 DLL 功能使能, 若无失锁现象发生, DLLLKF 位为 “0”, 若发生失锁现象, 此位置高且可由软件清零。若 DLLLKEN=0, DLLLKF 位始终为零。

### CPR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DLLLKEN	DLLEN	PWMCK1	PWMCK0	PWMON	DT2	DT1	DT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	0	0	0	0	0

Bit 7

**DLLLKEN:** DLL 电路锁定锁保护功能控制位

0: 除能

1: 使能

注: 当 DLLLKEN=1 且 DLL 功能使能, 若发生失锁现象, DLLLKF 位置高且 DLL 将自动排除失锁现象。而当 DLLLKEN=0 时, 即使发生失锁现象, DLLLKF 位始终为零, 此时 DLL 将不排除失锁现象。

Bit 6	<b>DLLEN:</b> DLL 与死区时间功能控制位 0: DLL 除能且未插入死区时间 1: DLL 使能且插入死区时间 (由 DT[2:0] 所决定) 若此位为零则 PWMCK[1:0] 位段可通过软件设置为“00~11”且 H.R_PWM=PWM，无死区时间插入。若此位置高，硬件将 PWMCK[1:0] 设为“00”且不可改变，DLL 将微调 PWM，此时得到带死区时间插入的高精度 PWM 输出。
Bit 5~4	<b>PWMCK1~PWMCK0:</b> PWM 计数器时钟源选择 00: f <sub>HIRC</sub> 01: f <sub>HIRC</sub> /2 10: f <sub>HIRC</sub> /4 11: f <sub>HIRC</sub> /8
Bit 3	<b>PWMON:</b> PWM 功能控制位 0: 除能, PWM 计数器 = 0 1: 使能
Bit 2~0	<b>DT2~DT0:</b> 死区时间选择 000: 死区时间 = t <sub>DLL</sub> × 0 ~ t <sub>DLL</sub> × 1 001: 死区时间 = t <sub>DLL</sub> × 2 ~ t <sub>DLL</sub> × 3 010: 死区时间 = t <sub>DLL</sub> × 4 ~ t <sub>DLL</sub> × 5 011: 死区时间 = t <sub>DLL</sub> × 6 ~ t <sub>DLL</sub> × 7 100: 死区时间 = t <sub>DLL</sub> × 8 ~ t <sub>DLL</sub> × 9 101: 死区时间 = t <sub>DLL</sub> × 10 ~ t <sub>DLL</sub> × 11 110: 死区时间 = t <sub>DLL</sub> × 12 ~ t <sub>DLL</sub> × 13 111: 死区时间 = t <sub>DLL</sub> × 14 ~ t <sub>DLL</sub> × 15 注: t <sub>DLL</sub> =1/(f <sub>HIRC</sub> ×16)

### CPOR 寄存器

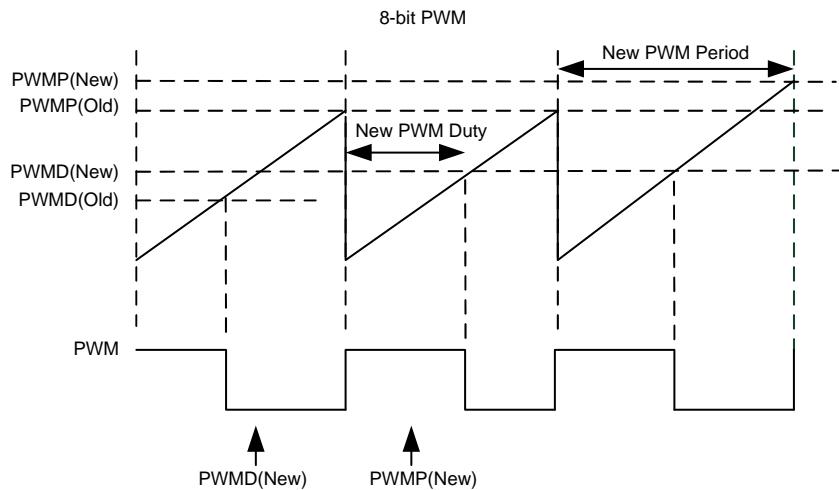
Bit	7	6	5	4	3	2	1	0
Name	PWMACT	CNVTYP	INVH	INVL	OCPHEN	OCPLEN	OVPHEN	OVPLEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7	<b>PWMACT:</b> PWMH/PWML 输出有效控制位 0: 强制无效 1: 有效
Bit 6	<b>CNVTYP:</b> 转换器类型选择位 0: 降压 (占空比 @ PWMH) 1: 升压 (占空比 @ PWML) CNVTYP 位用于选择降压或升压转换器类型且确定 PWM 输出为 PWMH 或 PWML。若使用降压型，则占空比控制输出到 PWMH，当使用升压型则占空比控制输出到 PWML。
Bit 5	<b>INVH:</b> 送入 PWMH_HV 前的反相控制 0: 同相 1: 反相
Bit 4	<b>INVL:</b> 送入 PWML_HV 前的反相控制 0: 同相 1: 反相
Bit 3	<b>OCPHEN:</b> 对 PWMH 过电流保护使能控制位 0: 除能 1: 使能
Bit 2	<b>OCPLEN:</b> 对 PWMH 过电流保护使能控制位 0: 除能 1: 使能

- |       |  |
|-------|--|
| Bit 1 | <b>OVPHEN:</b> 对 PWMH 过电压保护使能控制位<br>0: 除能<br>1: 使能 |
| Bit 0 | <b>OVPLEN:</b> 对 PWML 过电压保护使能控制位<br>0: 除能<br>1: 使能 |

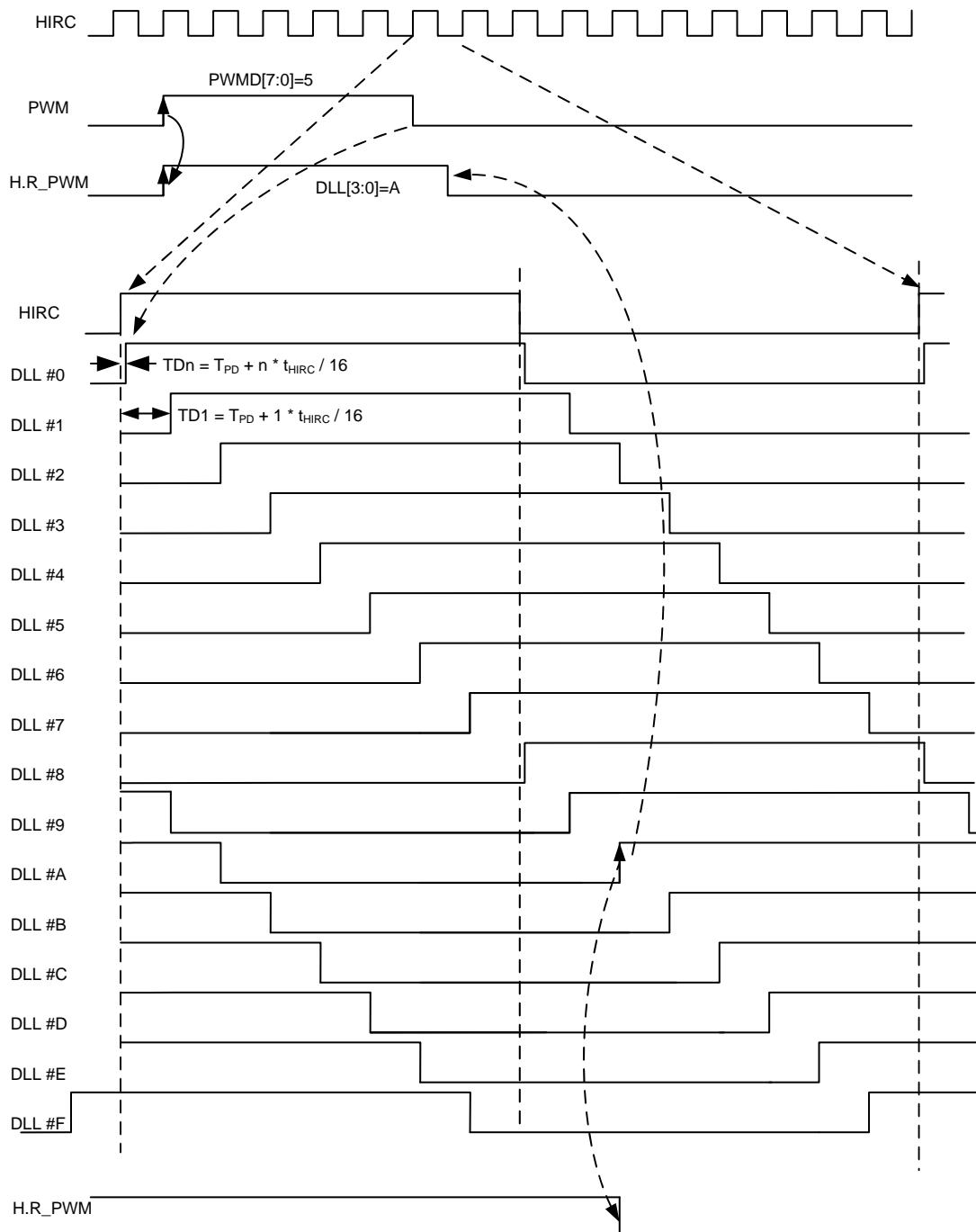
## PWM 发生器

PWM 由 HIRC 时钟驱动并产生一个 PWM 信号，通过配置 8-bit PWMP 和 PWMD 寄存器其占空比和周期可变。PWM 信号周期取决于 PWM 计数器时钟源，由 CPR 寄存器中的 PWMCK[1:0] 设置且取决于 PWMP 寄存器。PWM 信号占空比由 PWMD 寄存器所确定。



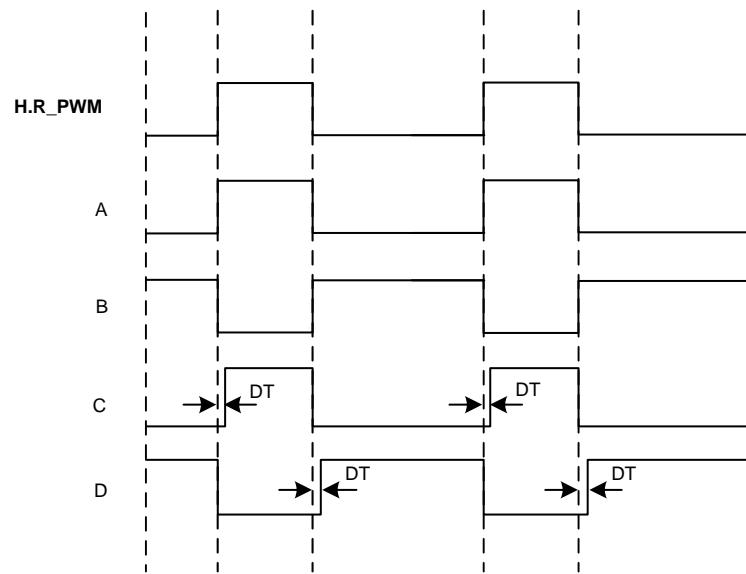
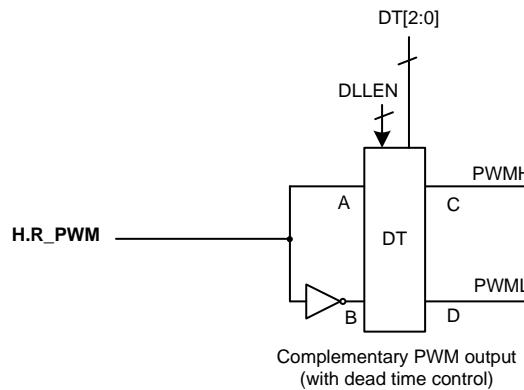
## 延迟锁相环

DLL 为延迟锁相环 (Delay Lock Loop) 的简写。DLL 可在一个 HIRC 时钟周期内产生 16 个相位输出。此 16 个相位输出用于微调 PWM 信号输出。若 PWM 时钟 =  $f_{HIRC}$ ，这表示 PWM 输出占空比精度为  $1/f_{HIRC}$ 。PWM 信号通过 DLL 相位选择与 H.R\_PWM 控制 (由 DLL 寄存器中的 DLL[3:0] 控制) 电路后输出微调过的 PWM 信号，且 H.R\_PWM 的占空比精度提升了 4-bit。



## 死区时间插入

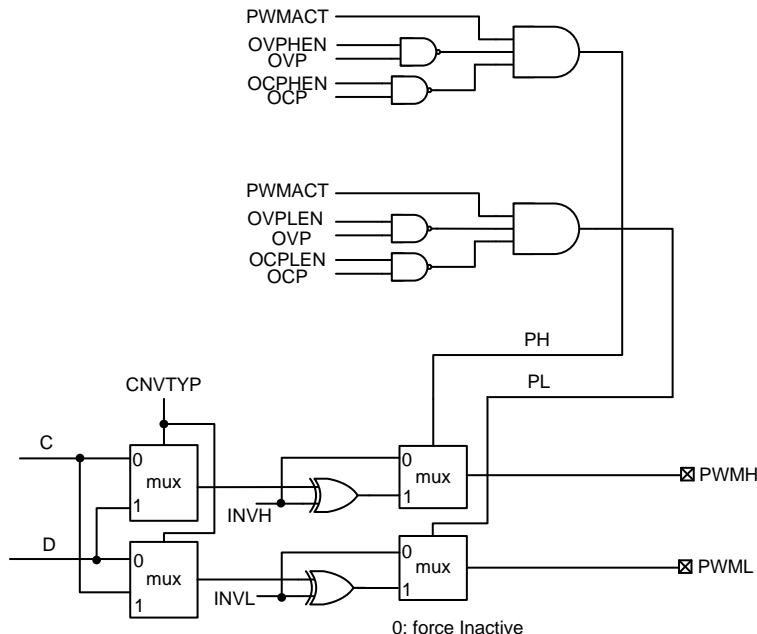
该系列单片机提供信号的互补输出对，可作为 PWM 驱动器信号。该信号源自高精度 PWM 输出信号，带 DLL 电路的 8-bit PWM。PWM 输出为高有效信号。通过 DLLEN 位可使能死区时间发生器，对 CPR 寄存器中的 DT[2:0] 进行编程配置将插入一个死区时间以预防过大的直流电流。死区时间发生器输入信号每出现一个上升沿便插入一个死区时间。



注：C 和 D 为带死区时间电路的互补 PWM 控制的输出信号。

## 保护与反相控制

尽管已在 H.R\_PWM 互补信号对中插入了一个死区时间以预防过大直流电流，这两个信号也可能由于出错或电噪声等不可预期的原因而处于无效状态。该系列单片机提供了一个保护功能，在 PWMH 或 PWML 信号为无效状态时强制使这两个信号输出反相信号。反相控制电路通过 CPOR 寄存器中的 INVH 或 INVL 决定了是否将信号反相或不使用相应的反相控制位。该系列单片机还包含了过电流保护与过电压保护功能，分别在 OCP 与 OVP 章节中进行描述。通过适当配置 OCPHEN、OCPLEN、OVPHEN 和 OVPLEN 等位将使能输出信号的保护电路。



## 编程注意事项

下列步骤说明了读写的过程：

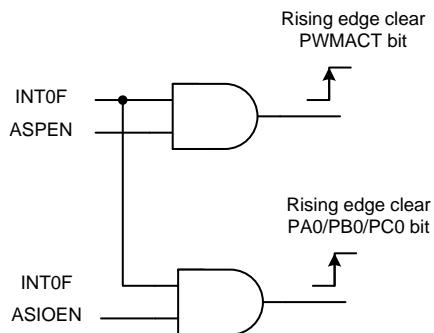
- 写数据到 DLL/PWMD
  - ◆ 步骤 1. 写数据到 DLL
    - 注意这里数据仅写入 4-bit 缓冲器
  - ◆ 步骤 2. 写数据到 PWMD
    - 这里数据直接写入 PWMD，同时数据从 4-bit 缓冲器锁存至 DLL 寄存器
- 从 DLL/PWMD 中读取数据
  - ◆ 步骤 1. 从 PWMD 中读取数据
    - 这里数据从 PWMD 寄存器中直接被读取，同时数据从 DLL 寄存器锁存至 4-bit 缓冲器
  - ◆ 步骤 2. 从 DLL 中读取数据
    - 此步骤从 4-bit 缓冲器读取数据

## 自动关闭控制 – 仅用于 HT45F3430

HT45F3430 单片机提供了一个自动关闭功能以保护内部电路不被外部恶劣情况损坏。

### 功能性描述

在 HT45F3430 单片机中，PWM 与 PA0/PB0/PC0 端口有自动关闭功能。当某些外部状况发生(如USB电源插入)时，硬件将自动关闭 PWM 或指定的 I/O 端口，此功能常用于控制负载的 On/Off。



### ASCR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	ASPEN	ASIOEN	ASIOS1	ASIOS0
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 5 **ASPEN:** PWM 自动关闭使能控制位

- 0: 除能
- 1: 使能

若 ASPEN=1，出现 INT0 中断时 PWMACT 将被清零。

Bit 2 **ASIOEN:** I/O 端口自动关闭使能控制位

- 0: 除能
- 1: 使能

若 ASIOEN=1，出现 INT0 中断时 PA0/PB0/PC0 位将被清零。PA0、PB0 或 PC0 由 ASIOS[1:0] 位段选择。

Bit 1~0 **ASIOS1~ASIOS0:** 自动关闭 I/O 端口选择

- 00: PA0
- 01: PB0
- 1x: PC0

注：当 ASPEN 或 ASIOEN 位为“1”，若用软件将 INT0F 位置高，将不清除 PWMACT 或 PA0/PB0/PC0 位。只有当 INT0F 标志位由外部 INT0 引脚触发时才会执行此功能。

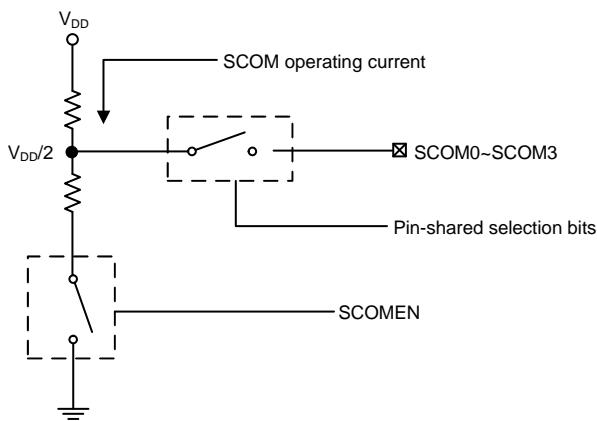
## LCD SCOM 功能 – 仅用于 HT45F3430

HT45F3430 单片机具有驱动外部 LCD 面板的能力。LCD 驱动的 COM 脚 SCOM0~SCOM3 与 I/O 口的某些引脚共用。LCD 信号由应用程序产生。

### LCD 操作

单片机通过设置相关 I/O 口为 COM 引脚驱动外部的晶体面板。LCD 驱动功能是由 SCOMC 寄存器控制的，另外，SCOMC 寄存器还可设置开启 / 关闭功能以及控制偏压设置功能，这使得 LCD 的 COM 驱动器为 LCD 1/2 偏压操作产生其所必需的  $V_{DD}/2$  电平。

SCOMC 寄存器中的 SCOMEN 位是 LCD 驱动的主控制位。通过相应的引脚共用功能选择位选择 LCD SCOMn 引脚用于 LCD 驱动。应注意端口控制寄存器无需预先将这些引脚设置为输出以使能 LCD 驱动器操作。



**LCD COM 偏压**

### LCD 偏压电流控制

LCD COM 驱动器可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 ISEL0 位和 ISEL1 位可以配置不同的偏压电阻。

#### SCOMC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	ISEL1	ISEL0	SCOMEN	—	—	—	—
R/W	—	R/W	R/W	R/W	—	—	—	—
POR	—	0	0	0	—	—	—	—

Bit 7 未定义，读为“0”

Bit 6~5 **ISEL1~ISEL0:** 针对 R 型 LCD 偏压电流选择电阻 ( $V_{DD}=5V$ )

00:  $2 \times 100k\Omega$  (1/2 Bias),  $I_{BIAS}=25\mu A$

01:  $2 \times 50k\Omega$  (1/2 Bias),  $I_{BIAS}=50\mu A$

10:  $2 \times 25k\Omega$  (1/2 Bias),  $I_{BIAS}=100\mu A$

11:  $2 \times 12.5k\Omega$  (1/2 Bias),  $I_{BIAS}=200\mu A$

Bit 4 **SCOMEN:** LCD 功能使能控制位

0: 除能

1: 使能

SCOMEN 置高将开启电阻的直流通道，产生 1/2  $V_{DD}$  偏压。

Bit 3~0 未定义，读为“0”

## 中断

中断是单片机一个重要功能。当外部事件或内部功能如定时器模块或 A/D 转换器等需要单片机注意时，其相应的中断将使系统暂时中止当前的程序而转到执行相对应的中断服务程序。此系列单片机提供了外部中断和内部中断功能，外部中断由 INTn 引脚产生，而内部中断由各种内部功能，如定时器模块、时基、EEPROM、OCP、OVP 和 A/D 转换器等产生。

### 中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于特殊功能数据存储器中的一系列寄存器控制的。这些寄存器分为三类。第一类是 INTC0~INTC2 寄存器，用于设置基本的中断；第二类是 MFI 寄存器，用于设置多功能中断；第三类是 INTEG 寄存器，用于设置外部中断边沿触发类型。

寄存器中含有中断控制位和中断请求标志位。中断控制位用于使能或除能各种中断，中断请求标志位用于存放当前中断请求的状态。它们都按照特定的模式命名，前面表示中断类型的缩写，紧接着是中断号（可选），最后的字母“E”代表使能/除能位，“F”代表请求标志位。

功能	使能位	请求标志位	注释
总中断	EMI	—	—
OCP 中断	OCPE	OCPF	—
OVP 中断	OVPE	OVPF	—
INTn 引脚中断	INTnE	INTnF	仅用于 HT45F3420, n=0 仅用于 HT45F3430, n=0 或 1
多功能中断	MFE	MFF	—
A/D 转换器中断	ADE	ADF	—
时基中断	TBnE	TBnF	n=0 或 1
EEPROM 中断	DEE	DEF	—
TM 中断	STMAE	STMAF	—
	STMPE	STMPF	—

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	—	—	INT0S1	INT0S0
INTC0	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	EMI
INTC1	DEF	ADF	—	MFF	DEE	ADE	—	MFE
INTC2	—	—	TB1F	TB0F	—	—	TB1E	TB0E
MFI	—	—	STMAF	STMPF	—	—	STMAE	STMPE

中断寄存器列表 – HT45F3420

寄存器 名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	EMI
INTC1	DEF	ADF	—	MFF	DEE	ADE	—	MFE
INTC2	—	INT1F	TB1F	TB0F	—	INT1E	TB1E	TB0E
MFI	—	—	STMAF	STMPF	—	—	STMAE	STMPE

### 中断寄存器标志位 – HT45F3430

#### INTEG 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	INT0S1	INT0S0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”

Bit 1~0 **INT0S1~INT0S0:** INT0 引脚中断有效边沿选择

- 00: 除能中断
- 01: 上升沿中断
- 10: 下降沿中断
- 11: 双沿中断

#### INTEG 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
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 **INT1S1~INT1S0:** INT1 引脚中断有效边沿选择

- 00: 除能中断
- 01: 上升沿中断
- 10: 下降沿中断
- 11: 双沿中断

Bit 1~0 **INT0S1~INT0S0:** INT0 引脚中断有效边沿选择

- 00: 除能中断
- 01: 上升沿中断
- 10: 下降沿中断
- 11: 双沿中断

### INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	INT0F	OVPF	OCPF	INT0E	OVPE	OCPE	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 **INT0F:** 外部中断 0 请求标志位

0: 无请求

1: 中断请求

Bit 5 **OVPF:** OVP 中断请求标志位

0: 无请求

1: 中断请求

Bit 4 **OCPF:** OCP 中断请求标志位

0: 无请求

1: 中断请求

Bit 3 **INT0E:** 外部中断 0 控制位

0: 除能

1: 使能

Bit 2 **OVPE:** OVP 中断控制位

0: 除能

1: 使能

Bit 1 **OCPE:** OCP 中断控制位

0: 除能

1: 使能

Bit 0 **EMI:** 总中断控制位

0: 除能

1: 使能

### INTC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DEF	ADF	—	MFF	DEE	ADE	—	MFE
R/W	R/W	R/W	—	R/W	R/W	R/W	—	R/W
POR	0	0	—	0	0	0	—	0

Bit 7      **DEF:** 数据 EEPROM 中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 6      **ADF:** A/D 转换器中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 5      未定义, 读为“0”

Bit 4      **MFF:** 多功能中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 3      **DEE:** 数据 EEPROM 中断控制位

- 0: 除能
- 1: 使能

Bit 2      **ADE:** A/D 转换器中断控制位

- 0: 除能
- 1: 使能

Bit 1      未定义, 读为“0”

Bit 0      **MFE:** 多功能中断控制位

- 0: 除能
- 1: 使能

### INTC2 寄存器 – HT45F3420

Bit	7	6	5	4	3	2	1	0
Name	—	—	TB1F	TB0F	—	—	TB1E	TB0E
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6      未定义, 读为“0”

Bit 5      **TB1F:** 时基 1 中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 4      **TB0F:** 时基 0 中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 3~2      未定义, 读为“0”

Bit 1      **TB1E:** 时基 1 中断控制位

- 0: 除能
- 1: 使能

Bit 0      **TB0E:** 时基 0 中断控制位

- 0: 除能
- 1: 使能

### INTC2 寄存器 – HT45F3430

Bit	7	6	5	4	3	2	1	0
Name	—	INT1F	TB1F	TB0F	—	INT1E	TB1E	TB0E
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **INT1F:** 外部中断 1 请求标志位  
0: 无请求  
1: 中断请求

Bit 5 **TB1F:** 时基 1 中断请求标志位  
0: 无请求  
1: 中断请求

Bit 4 **TB0F:** 时基 0 中断请求标志位  
0: 无请求  
1: 中断请求

Bit 3 未定义，读为“0”

Bit 2 **INT1E:** 外部中断 1 控制位  
0: 除能  
1: 使能

Bit 1 **TB1E:** 时基 1 中断控制位  
0: 除能  
1: 使能

Bit 0 **TB0E:** 时基 0 中断控制位  
0: 除能  
1: 使能

### MFI 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	STMAF	STMPF	—	—	STMAE	STMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **STMAF:** STM 比较器 A 中断请求标志位  
0: 无请求  
1: 中断请求

Bit 4 **STMPF:** STM 比较器 P 中断请求标志位  
0: 无请求  
1: 中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **STMAE:** STM 比较器 A 中断控制位  
0: 除能  
1: 使能

Bit 0 **STMPE:** STM 比较器 P 中断控制位  
0: 除能  
1: 使能

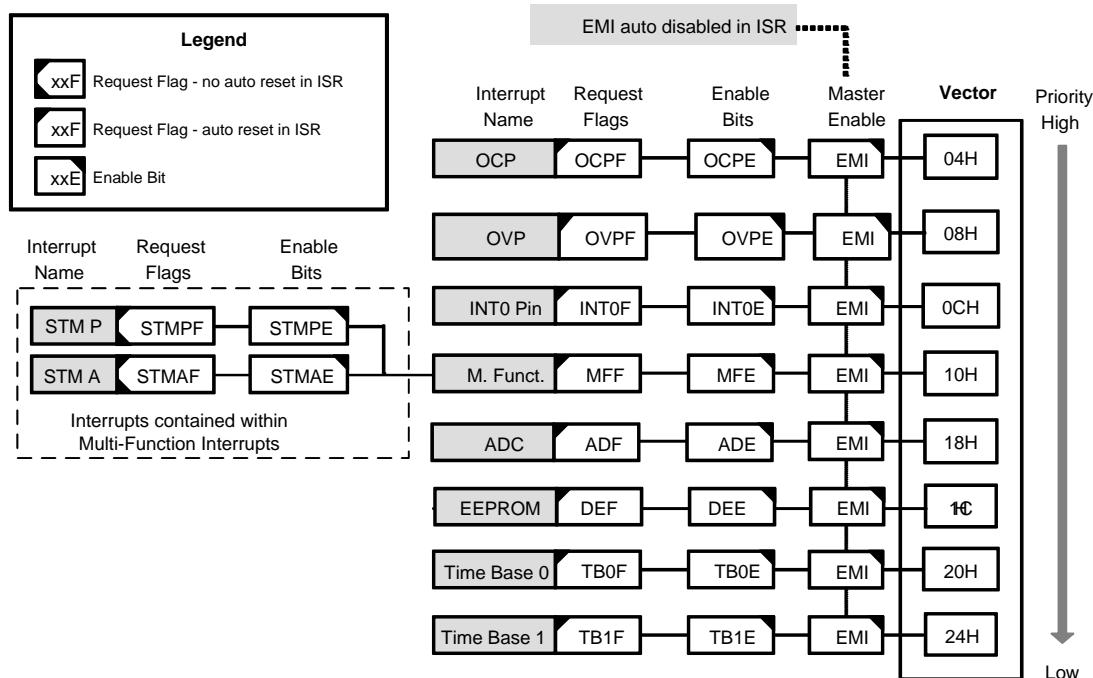
## 中断操作

若中断事件条件产生，如一个 TM 比较器 P、比较器 A 匹配或 A/D 转换结束等，相关中断请求标志将置起。中断标志产生后程序是否会跳转至相关中断向量执行是由中断使能位的条件决定的。若使能位为“1”，程序将跳至相关中断向量中执行；若使能位为“0”，即使中断请求标志置起中断也不会发生，程序也不会跳转至相关中断向量执行。若总中断使能位为“0”，所有中断都将除能。

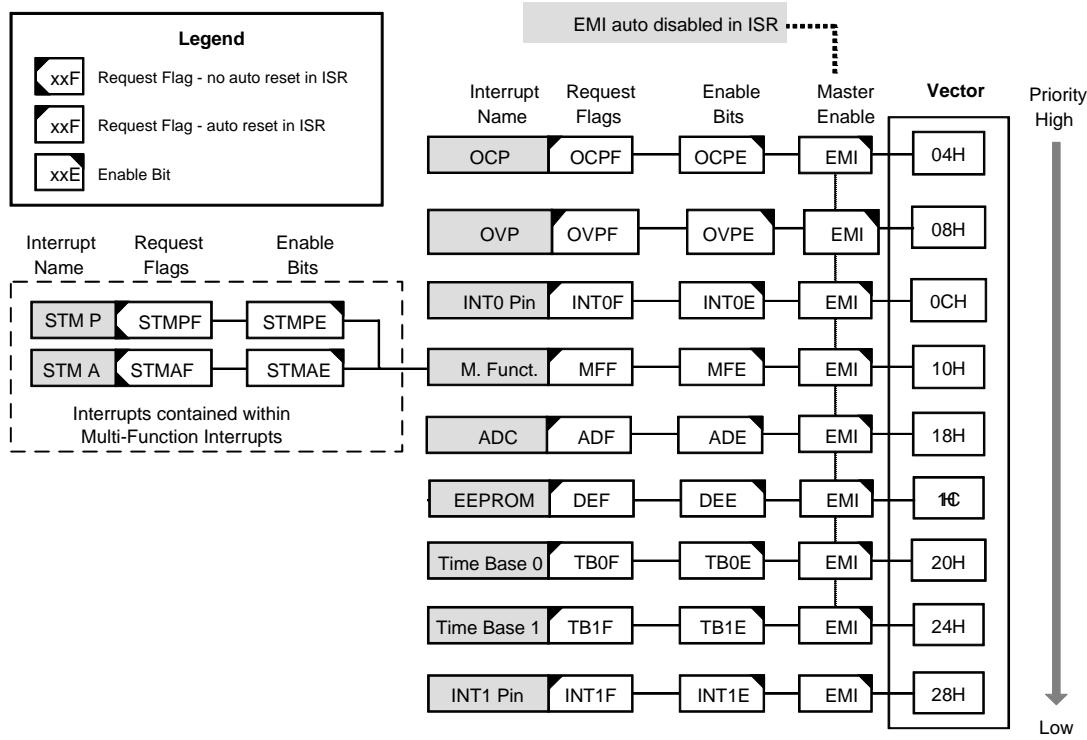
当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为跳转指令，以跳转到相应的中断服务程序。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

各个中断使能位以及相应的请求标志位，以优先级的次序显示在下图。一些中断源有自己的向量，但是有些中断却共用多功能中断向量。一旦中断子程序被响应，系统将自动清除 EMI 位，所有其它的中断将被屏蔽，这个方式可以防止任何进一步的中断嵌套。其它中断请求可能发生在此期间，虽然中断不会立即响应，但是中断请求标志位会被记录。

如果某个中断服务子程序正在执行时，有另一个中断要求立即响应，那么 EMI 位应在程序进入中断子程序后置位，以允许此中断嵌套。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。如果要求立刻动作，则堆栈必须避免成为储满状态。请求同时发生时，执行优先级如下流程图所示。所有被置起的中断请求标志都可把单片机从休眠或空闲模式中唤醒，若要防止唤醒动作发生，在单片机进入休眠或空闲模式前应将相应的标志置起。



中断结构 – HT45F3420



中断结构 – HT45F3430

## 外部中断

通过 INT0 或 INT1 引脚上的信号变化可控制外部中断。对于 HT45F3430 单片机，INT0 和 INT1 输入源可通过设置 IFS0 寄存器的位来进行选择。当触发沿选择位设置好触发类型，外部引脚的状态发生变化，外部中断请求标志 INTnF 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INTnE 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INTnF 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其上拉电阻选择仍保持有效。

寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

## OCP 中断

通过检测 OCP 输入电流控制 OCP 中断。当检测到大电流时，其中断请求标志位 OCPF 置位，OCP 中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和 OCP 中断使能位 OCPE 需先被设置。当中断使能，堆栈未满且发生过电流情况时，将调用 OCP 中断向量子程序。当响应 OCP 中断服务子程序时，OCP 中断请求标志位 OCPF 将自动清除，EMI 也将自动清除以除能其它中断。

## OVP 中断

通过检测输入电压控制 OVP 中断。当检测到高电压时，其中断请求标志位 OVPF 置位，OVP 中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和 OVP 中断使能位 OVPE 需先被设置。当中断使能，堆栈未满且检测到高电压时，将调用 OVP 中断向量子程序。当响应 OVP 中断服务子程序时，OVP 中断请求标志位 OVPF 将自动清除，EMI 也将自动清除以除能其它中断。

## 多功能中断

该系列单片机中只有 1 种多功能中断，与其它中断不同，这些没有独立源，但由其它现有的中断源构成，即 STM 的 CCRA 与 CCRP 中断。

当多功能中断中任何一种中断请求标志 MFF 被置位，多功能中断请求产生。当所包含的任一功能产生中断请求标志，多功能中断标志将置位。若要跳转到相应的中断向量地址，当多功能中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

但必须注意的是，在中断响应时，虽然多功能中断标志会自动复位，但多功能中断源的请求标志位，即 STM 中断的请求标志位不会自动复位，必须由应用程序清零。

## A/D 转换器中断

A/D 转换动作的结束控制 A/D 转换器中断。当 A/D 转换器中断请求标志 ADF 被置位，即 A/D 转换过程完成时，中断请求发生。若要跳转到相应中断向量地址，总中断控制位 EMI、A/D 转换器中断使能位 ADE 需先被置位。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用 A/D 转换器中断向量子程序。当 A/D 转换器中断响应时，ADF 标志将自动清除，EMI 将被自动清零以除能其它中断。

## 时基中断

时基中断提供一个固定周期的中断信号，由各自的定时器功能产生溢出信号控制。当出现这种情况时其各自的中断请求标志 TB0F 或 TB1F 被置位，中断请求发生。若要跳转到其相应的中断向量地址，总中断使能位 EMI 和时基使能位 TB0E 或 TB1E 需先被置位。当中断使能，堆栈未满且时基溢出时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 TB0F 或 TB1F 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断的目的是提供一个固定周期的中断信号。其时钟源来自内部时钟源  $f_{TB}$ 。 $f_{TB}$  输入时钟通过一个分频器，其分频比可由程序配置 TBC 寄存器中的相关位获取合适的分频值以提供更长的时基中断周期。产生  $f_{TB}$  的时钟源还控制时基中断周期，并可来自多种不同的时钟源，可通过 TBC 寄存器的 TBCK 位选择。

**TBC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB11	TB10	—	TB02	TB01	TB00
R/W	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	0	0	1	1	—	1	1	1

Bit 7      **TBON:** TB0 和 TB1 控制位

- 0: 除能
- 1: 使能

Bit 6      **TBCK:**  $f_{TB}$  时钟源选择位

- 0:  $f_{TBC}$
- 1:  $f_{SYS}/4$

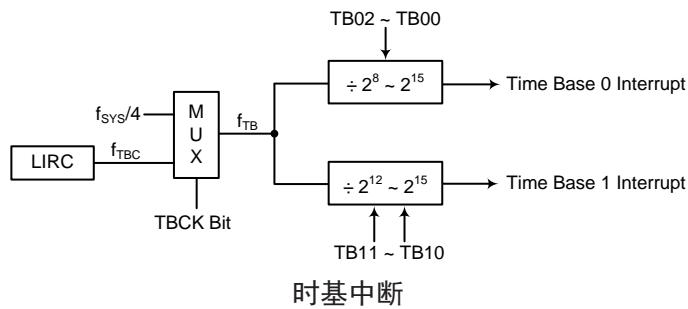
Bit 5~4    **TB11~TB10:** 时基 1 溢出周期选择

- 00:  $2^{12}/f_{TB}$
- 01:  $2^{13}/f_{TB}$
- 10:  $2^{14}/f_{TB}$
- 11:  $2^{15}/f_{TB}$

Bit 3       未定义, 读为“0”

Bit 2~0     **TB02~TB00:** 时基 0 溢出周期选择

- 000:  $2^8/f_{TB}$
- 001:  $2^9/f_{TB}$
- 010:  $2^{10}/f_{TB}$
- 011:  $2^{11}/f_{TB}$
- 100:  $2^{12}/f_{TB}$
- 101:  $2^{13}/f_{TB}$
- 110:  $2^{14}/f_{TB}$
- 111:  $2^{15}/f_{TB}$


**EEPROM 中断**

当 EEPROM 写周期结束, EEPROM 中断请求标志位 DEF 将置位, EEPROM 中断请求产生。若要程序跳转到相应中断向量地址, 总中断控制位 EMI 和 EEPROM 中断使能位 DEE 需先被置位。当中断使能, 堆栈未满且一个 EEPROM 写周期结束时, 将调用相应的 EEPROM 中断向量子程序。当相应 EEPROM 中断服务时, EMI 位将自动清零以除能其它中断, DEF 标志位也将自动清零。

## TM 中断

STM 有两个中断，都属于多功能中断。STM 有两个中断请求标志位 STMPF 和 STMAF 及两个使能位 STMPE 和 STMAE。当 STM 比较器 P 或比较器 A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

若要程序跳转到相应中断向量地址，总中断控制位 EMI、相应 TM 中断使能位和相关多功能中断使能位 MFE 需先被置位。当中断使能，堆栈未满且 TM 比较器匹配情况发生时，可跳转至相关多功能中断向量子程序中执行。当 TM 中断响应，EMI 将被自动清零以除能其它中断，相关 MFF 标志也可自动清除，但 TM 中断请求标志需在应用程序中手动清除。

## 中断唤醒功能

每个中断都具有将处于休眠或空闲模式的单片机唤醒的能力。当中断请求标志由低到高转换时唤醒动作产生，其与中断是否使能无关。因此，尽管单片机处于休眠或空闲模式且系统振荡器停止工作，如有外部中断脚上产生外部边沿跳变、低电压都可能导致其相应的中断标志被置位，由此产生中断，因此必须注意避免伪唤醒情况的发生。若中断唤醒功能被除能，单片机进入休眠或空闲模式前相应中断请求标志应被置起。中断唤醒功能不受中断使能位的影响。

## 编程注意事项

通过禁止相关中断使能位，可以屏蔽中断请求，然而，一旦中断请求标志位被设定，它们会被保留在中断控制寄存器内，直到相应的中断服务子程序执行或请求标志位被软件指令清除。

多功能中断中所含中断相应程序执行时，多功能中断请求标志 MFF 可以自动清零，但各自的请求标志需在应用程序中手动清除。

建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

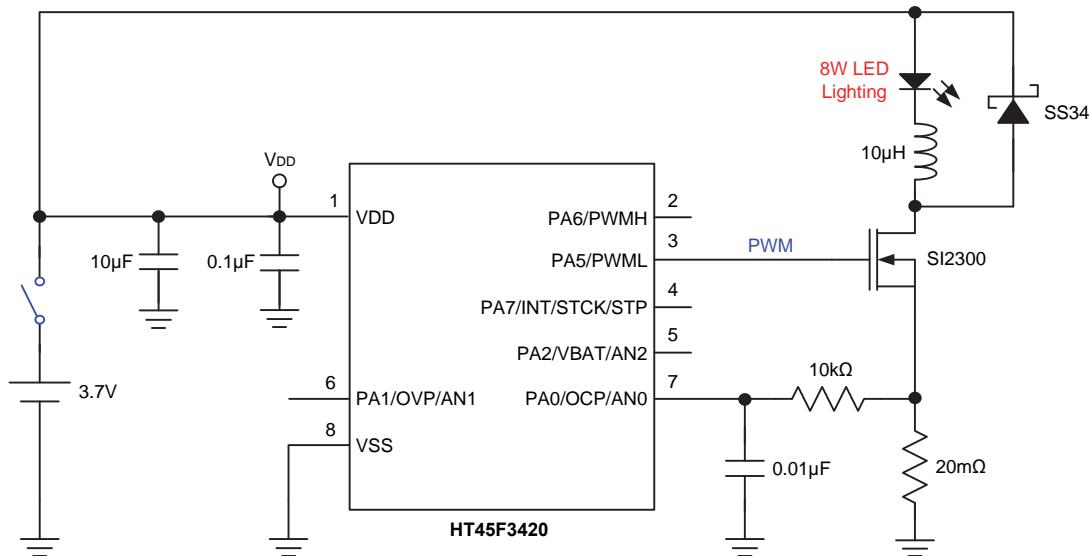
所有中断在休眠或空闲模式下都具有唤醒功能，当中断请求标志发生由低到高的转变时都可产生唤醒功能。若要避免相应中断产生唤醒动作，在单片机进入休眠或空闲模式前需先将相应请求标志置为高。

当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，如果中断服务程序会改变状态寄存器或其它的寄存器的内容而破坏控制流程，应事先将这些数据保存起来。

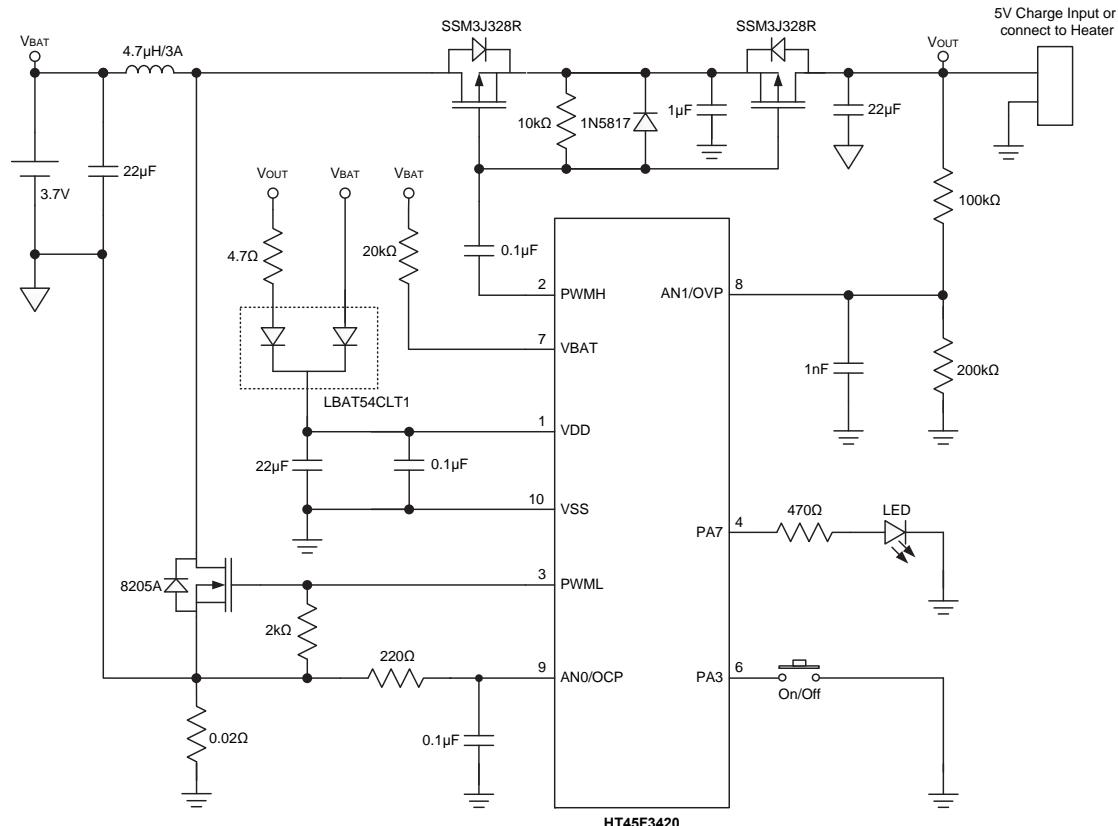
若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清除 EMI 位，除能进一步中断。

## 应用电路

### LED 手电筒



### 电子烟



## 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 HOLTEK 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它们的应用。

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 0.5μs 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”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

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	指令周期	影响标志位
<b>算术运算</b>			
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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	C
<b>逻辑运算</b>			
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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	Z
CPLA [m]	对数据存储器取反，结果放入 ACC	1	Z
<b>递增和递减</b>			
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 <sup>注</sup>	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 <sup>注</sup>	Z

助记符	说明	指令周期	影响标志位
<b>移位</b>			
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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	C
<b>数据传送</b>			
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 <sup>注</sup>	无
MOV A, x	将立即数送至 ACC	1	无
<b>位运算</b>			
CLR [m].i	清除数据存储器的位	1 <sup>注</sup>	无
SET [m].i	置位数据存储器的位	1 <sup>注</sup>	无
<b>转移</b>			
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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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 <sup>注</sup>	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
<b>查表</b>			
TABRD [m]	读取特定页的 ROM 内容, 并送至数据存储器和 TBLH	2 <sup>注</sup>	无
TABRDC [m]	读取当前页的 ROM 内容, 并送至数据存储器和 TBLH	2 <sup>注</sup>	无
TABRDL [m]	读取最后页的 ROM 内容, 并送至数据存储器和 TBLH	2 <sup>注</sup>	无
<b>其它指令</b>			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 <sup>注</sup>	无
SET [m]	置位数据存储器	1 <sup>注</sup>	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

助记符	说明	指令周期	影响标志位
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 <sup>注</sup>	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, 否则 TO 和 PDF 标志位保持不变

## 指令定义

### **ADC A, [m]**

指令说明

Add Data Memory to ACC with Carry

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。

功能表示

$ACC \leftarrow ACC + [m] + C$

影响标志位

OV、Z、AC、C

### **ADCM A, [m]**

指令说明

Add ACC to Data Memory with Carry

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。

功能表示

$[m] \leftarrow ACC + [m] + C$

影响标志位

OV、Z、AC、C

### **ADD A, [m]**

指令说明

Add Data Memory to ACC

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。

功能表示

$ACC \leftarrow ACC + [m]$

影响标志位

OV、Z、AC、C

### **ADD A, x**

指令说明

Add immediate data to ACC

功能表示

$ACC \leftarrow ACC + x$

影响标志位

OV、Z、AC、C

### **ADDM A, [m]**

指令说明

Add ACC to Data Memory

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。

功能表示

$[m] \leftarrow ACC + [m]$

影响标志位

OV、Z、AC、C

### **AND A, [m]**

指令说明

Logical AND Data Memory to ACC

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。

功能表示

$ACC \leftarrow ACC \text{ ``AND'' } [m]$

影响标志位

Z

<b>AND A, x</b>	Logical AND immediate data to ACC 将累加器中的数据和立即数做逻辑与，结果存放到累加器。 $ACC \leftarrow ACC \text{ ``AND'' } x$ Z
<b>ANDM A, [m]</b>	Logical AND ACC to Data Memory 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。 $[m] \leftarrow ACC \text{ ``AND'' } [m]$ Z
<b>CALL addr</b>	Subroutine call 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。 $Stack \leftarrow Program Counter + 1$ $Program Counter \leftarrow addr$ 无
<b>CLR [m]</b>	Clear Data Memory 将指定数据存储器的内容清零。 $[m] \leftarrow 00H$ 无
<b>CLR [m].i</b>	Clear bit of Data Memory 将指定数据存储器的 i 位内容清零。 $[m].i \leftarrow 0$ 无
<b>CLR WDT</b>	Clear Watchdog Timer WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。 WDT cleared $TO \& PDF \leftarrow 0$ TO、PDF

<b>CLR WDT1</b>	Preclear Watchdog Timer
指令说明	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。
功能表示	$\text{WDT} \leftarrow 00\text{H}$
影响标志位	TO、PDF
<b>CLR WDT2</b>	Preclear Watchdog Timer
指令说明	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。
功能表示	$\text{WDT} \leftarrow 00\text{H}$
影响标志位	TO、PDF
<b>CPL [m]</b>	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	$[m] \leftarrow [\bar{m}]$
影响标志位	Z
<b>CPLA [m]</b>	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	$\text{ACC} \leftarrow [\bar{m}]$
影响标志位	Z

<b>DAA [m]</b>	Decimal-Adjust ACC for addition with result in Data Memory 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，结果存放到数据存储器。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。
指令说明	
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
<b>DEC [m]</b>	Decrement Data Memory 将指定数据存储器内容减1。
指令说明	
功能表示	[m] ← [m] - 1
影响标志位	Z
<b>DECA [m]</b>	Decrement Data Memory with result in ACC 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。
指令说明	
功能表示	ACC ← [m] - 1
影响标志位	Z
<b>HALT</b>	Enter power down mode 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。
指令说明	
功能表示	TO ← 0 PDF ← 1
影响标志位	TO、PDF
<b>INC [m]</b>	Increment Data Memory 将指定数据存储器的内容加1。
指令说明	
功能表示	[m] ← [m] + 1
影响标志位	Z

<b>INCA [m]</b>	Increment Data Memory with result in ACC
指令说明	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。
功能表示	$ACC \leftarrow [m] + 1$
影响标志位	Z
<b>JMP addr</b>	Jump unconditionally
指令说明	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。
功能表示	$Program\ Counter \leftarrow addr$
影响标志位	无
<b>MOV A, [m]</b>	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。
功能表示	$ACC \leftarrow [m]$
影响标志位	无
<b>MOV A, x</b>	Move immediate data to ACC
指令说明	将 8 位立即数载入累加器。
功能表示	$ACC \leftarrow x$
影响标志位	无
<b>MOV [m], A</b>	Move ACC to Data Memory
指令说明	将累加器的内容复制到指定的数据存储器。
功能表示	$[m] \leftarrow ACC$
影响标志位	无
<b>NOP</b>	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	$PC \leftarrow PC+1$
影响标志位	无
<b>OR A, [m]</b>	Logical OR Data Memory to ACC
指令说明	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。
功能表示	$ACC \leftarrow ACC \text{ "OR" } [m]$
影响标志位	Z

<b>OR A, x</b>	Logical OR immediate data to ACC 将累加器中的数据和立即数逻辑或，结果存放到累加器。 $ACC \leftarrow ACC \text{ "OR" } x$ Z
<b>ORM A, [m]</b>	Logical OR ACC to Data Memory 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。 $[m] \leftarrow ACC \text{ "OR" } [m]$ Z
<b>RET</b>	Return from subroutine 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。 $\text{Program Counter} \leftarrow \text{Stack}$ 无
<b>RET A, x</b>	Return from subroutine and load immediate data to ACC 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。 $\text{Program Counter} \leftarrow \text{Stack}$ $ACC \leftarrow x$ 无
<b>RETI</b>	Return from interrupt 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。 $\text{Program Counter} \leftarrow \text{Stack}$ $EMI \leftarrow 1$ 无
<b>RL [m]</b>	Rotate Data Memory left 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。 $[m].(i+1) \leftarrow [m].i \ (i=0\sim6)$ $[m].0 \leftarrow [m].7$ 无

<b>RLA [m]</b>	Rotate Data Memory left with result in ACC
指令说明	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。
功能表示	$ACC.(i+1) \leftarrow [m].i \ (i=0\sim6)$
	$ACC.0 \leftarrow [m].7$
影响标志位	无
 <b>RLC [m]</b>	 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。
功能表示	$[m].(i+1) \leftarrow [m].i \ (i=0\sim6)$
	$[m].0 \leftarrow C$
	$C \leftarrow [m].7$
影响标志位	C
 <b>RLC A [m]</b>	 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。
功能表示	$ACC.(i+1) \leftarrow [m].i \ (i=0\sim6)$
	$ACC.0 \leftarrow C$
	$C \leftarrow [m].7$
影响标志位	C
 <b>RR [m]</b>	 Rotate Data Memory right
指令说明	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。
功能表示	$[m].i \leftarrow [m].(i+1) \ (i=0\sim6)$
	$[m].7 \leftarrow [m].0$
影响标志位	无
 <b>RRA [m]</b>	 Rotate Data Memory right with result in ACC
指令说明	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。
功能表示	$ACC.i \leftarrow [m].(i+1) \ (i=0\sim6)$
	$ACC.7 \leftarrow [m].0$
影响标志位	无

<b>RRC [m]</b>	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。
功能表示	$[m].i \leftarrow [m].(i+1) \ (i=0\sim6)$ $[m].7 \leftarrow C$ $C \leftarrow [m].0$
影响标志位	C
<b>RRCA [m]</b>	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。
功能表示	$ACC.i \leftarrow [m].(i+1) \ (i=0\sim6)$ $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
影响标志位	C
<b>SBC A, [m]</b>	Subtract Data Memory from ACC with Carry
指令说明	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	$ACC \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ
<b>SBCM A, [m]</b>	Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明	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	$[m] \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ
<b>SDZ [m]</b>	Skip if Decrement Data Memory is 0
指令说明	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。
功能表示	$[m] \leftarrow [m] - 1$ , 如果 $[m]=0$ 跳过下一条指令执行
影响标志位	无

<b>SDZA [m]</b>	Decrement data memory and place result in ACC, skip if 0
指令说明	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。
功能表示	$ACC \leftarrow [m]-1$ , 如果 $ACC=0$ 跳过下一条指令执行
影响标志位	无
<b>SET [m]</b>	Set Data Memory
指令说明	将指定数据存储器的每一位设置为 1。
功能表示	$[m] \leftarrow FFH$
影响标志位	无
<b>SET [m].i</b>	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	$[m].i \leftarrow 1$
影响标志位	无
<b>SIZ [m]</b>	Skip if increment Data Memory is 0
指令说明	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。
功能表示	$[m] \leftarrow [m]+1$ , 如果 $[m]=0$ 跳过下一条指令执行
影响标志位	无
<b>SIZA [m]</b>	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。
功能表示	$ACC \leftarrow [m]+1$ , 如果 $ACC=0$ 跳过下一条指令执行
影响标志位	无

<b>SNZ [m].i</b>	Skip if bit i of Data Memory is not 0
指令说明	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。
功能表示	如果 $[m].i \neq 0$ , 跳过下一条指令执行
影响标志位	无
<b>SUB A, [m]</b>	Subtract Data Memory from ACC
指令说明	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。
功能表示	$ACC \leftarrow ACC - [m]$
影响标志位	OV、Z、AC、C、SC、CZ
<b>SUBM A, [m]</b>	Subtract Data Memory from ACC with result in Data Memory
指令说明	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。
功能表示	$[m] \leftarrow ACC - [m]$
影响标志位	OV、Z、AC、C、SC、CZ
<b>SUB A, x</b>	Subtract immediate Data from ACC
指令说明	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。
功能表示	$ACC \leftarrow ACC - x$
影响标志位	OV、Z、AC、C、SC、CZ
<b>SWAP [m]</b>	Swap nibbles of Data Memory
指令说明	将指定数据存储器的低 4 位和高 4 位互相交换。
功能表示	$[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$
影响标志位	无
<b>SWAPA [m]</b>	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，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。
功能表示	$ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$ $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
影响标志位	无

<b>SZ [m]</b>	Skip if Data Memory is 0
指令说明	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。
功能表示	如果 $[m]=0$ , 跳过下一条指令执行
影响标志位	无
<b>SZA [m]</b>	Skip if Data Memory is 0 with data movement to ACC
指令说明	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。
功能表示	$ACC \leftarrow [m]$ , 如果 $[m]=0$ , 跳过下一条指令执行
影响标志位	无
<b>SZ [m].i</b>	Skip if bit i of Data Memory is 0
指令说明	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。
功能表示	如果 $[m].i=0$ , 跳过下一条指令执行
影响标志位	无
<b>TABRD [m]</b>	Read table (specific page) to TBLH and Data Memory
指令说明	将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	$[m] \leftarrow$ 程序代码 (低字节)
影响标志位	$TBLH \leftarrow$ 程序代码 (高字节) 无
<b>TABRDC [m]</b>	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) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	$[m] \leftarrow$ 程序代码 (低字节)
影响标志位	$TBLH \leftarrow$ 程序代码 (高字节) 无

<b>TABRDL [m]</b>	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	$[m] \leftarrow \text{程序代码 ( 低字节 )}$
	$\text{TBLH} \leftarrow \text{程序代码 ( 高字节 )}$
影响标志位	无
 <b>XOR A, [m]</b>	 Logical XOR Data Memory to ACC
指令说明	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。
功能表示	$\text{ACC} \leftarrow \text{ACC "XOR" } [m]$
影响标志位	Z
 <b>XORM A, [m]</b>	 Logical XOR ACC to Data Memory
指令说明	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。
功能表示	$[m] \leftarrow \text{ACC "XOR" } [m]$
影响标志位	Z
 <b>XOR A, x</b>	 Logical XOR immediate data to ACC
指令说明	将累加器的数据与立即数逻辑异或，结果存放到累加器。
功能表示	$\text{ACC} \leftarrow \text{ACC "XOR" } x$
影响标志位	Z

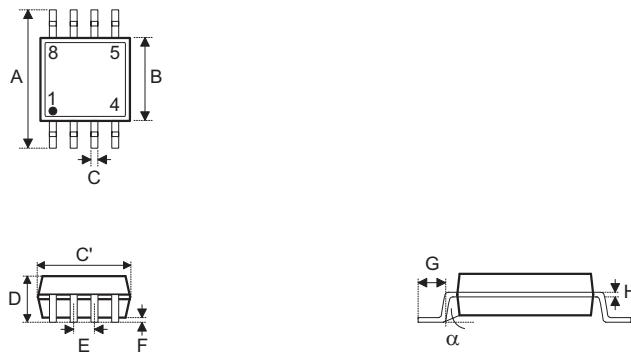
## 封装信息

请注意，这里提供的封装信息仅作为参考。由于这个信息经常更新，提醒用户咨询 [Holtek 网站](#) 以获取最新版本的封装信息。

封装信息的相关内容如下所示，点击可链接至 Holtek 网站相关信息页面。

- 封装信息 (包括外形尺寸、包装带和卷轴规格 )
- 封装材料信息
- 纸箱信息

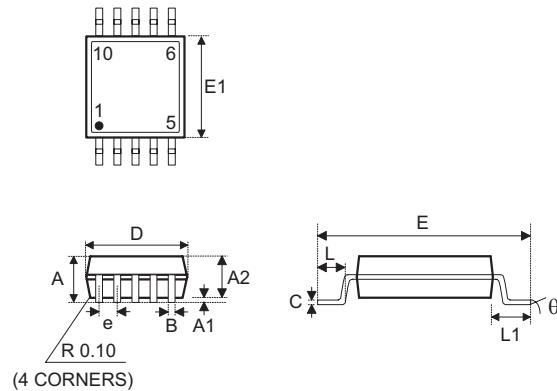
### 8-pin SOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小值	典型值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C'	—	0.193 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
$\alpha$	0°	—	8°

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C'	—	4.90 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
$\alpha$	0°	—	8°

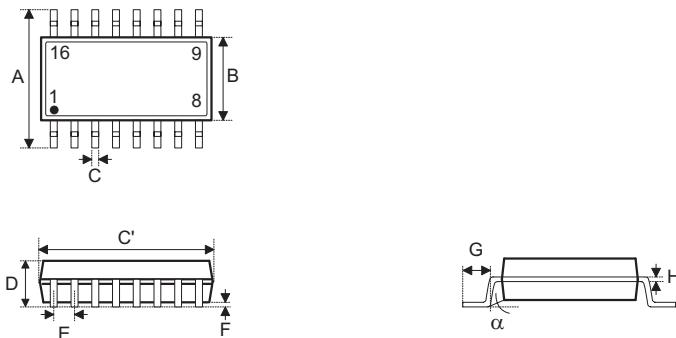
## 10-pin MSOP 外形尺寸



符号	尺寸(单位: inch)		
	最小值	典型值	最大值
A	—	—	0.043
A1	0.000	—	0.006
A2	0.030	0.033	0.037
B	0.007	—	0.013
C	0.003	—	0.009
D	—	0.118 BSC	—
E	—	0.193 BSC	—
E1	—	0.118 BSC	—
e	—	0.020 BSC	—
L	0.016	0.024	0.031
L1	—	0.037 BSC	—
y	—	0.004	—
θ	0°	—	8°

符号	尺寸(单位: mm)		
	最小值	典型值	最大值
A	—	—	1.10
A1	0.00	—	0.15
A2	0.75	0.85	0.95
B	0.17	—	0.33
C	0.08	—	0.23
D	—	3.00 BSC	—
E	—	4.90 BSC	—
E1	—	3.00 BSC	—
e	—	0.50 BSC	—
L	0.40	0.60	0.80
L1	—	0.95 BSC	—
y	—	0.10	—
θ	0°	—	8°

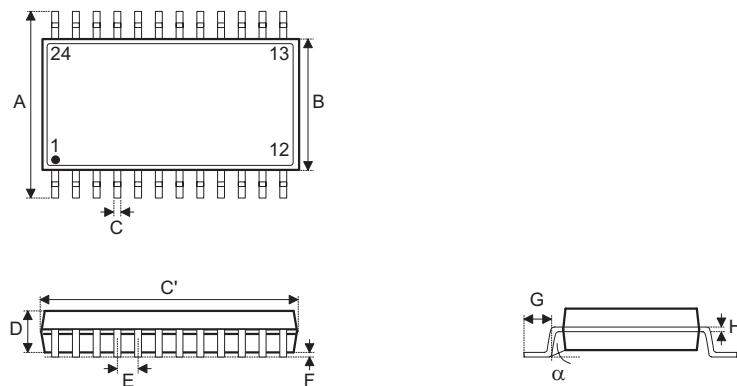
### 16-pin NSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小值	典型值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C'	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
$\alpha$	0°	—	8°

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.31	—	0.51
C'	—	9.9 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
$\alpha$	0°	—	8°

### 24-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小值	典型值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C'	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
$\alpha$	0°	—	8°

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.20	—	0.30
C'	—	8.66 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
$\alpha$	0°	—	8°

Copyright<sup>®</sup> 2016 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看说明，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、维生从机或系统中做为关键从机。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/zh/>.