



内置双运算放大器 A/D 型 Flash 单片机

HT66F4530/HT66F4540

HT66F4550/HT66F4560

版本 : V1.40 日期 : 2020-07-07

www.holtek.com

目 录

特性	7
CPU 特性	7
周边特性	7
概述	8
选型表	8
方框图	9
引脚图	9
引脚说明	12
极限参数	27
直流电气特性	28
工作电压特性	28
待机电流特性	29
工作电流特性	30
交流电气特性	31
内部高速振荡器 – HIRC – 频率精确度	31
内部低速振荡器电气特性 – LIRC	32
工作频率电气特性曲线图	32
系统上电时间电气特性	33
输入 / 输出电气特性	33
存储器电气特性	35
LVD&LVR 电气特性	35
A/D 转换器电气特性	36
内部参考电压特性	36
运算放大器电气特性	37
比较器电气特性	38
8-Bit D/A 转换器电气特性	39
16-Bit 语音 D/A 转换器电气特性	39
LCD 电气特性	39
上电复位特性	40
系统结构	40
时序和流水线结构	40
程序计数器	41
堆栈	42
算术逻辑单元 – ALU	42
Flash 程序存储器	43
结构	43
特殊向量	43
查表	44

查表范例	44
在线烧录 – ICP	45
片上调试 – OCDS	46
数据存储器	47
结构	47
数据存储器寻址	48
通用数据存储器	48
特殊数据存储器	48
特殊功能寄存器	53
间接寻址寄存器 – IAR0, IAR1, IAR2	53
存储器指针 – MP0, MP1L, MP1H, MP2L, MP2H	53
程序存储区指针 – PBP	54
累加器 – ACC	55
程序计数器低字节寄存器 – PCL	55
表格寄存器 – TBLP, TBHP, TBLH	55
状态寄存器 – STATUS	55
EEPROM 数据存储器	57
EEPROM 数据存储器结构	57
EEPROM 寄存器	57
从 EEPROM 中读取数据	59
写数据到 EEPROM	59
写保护	59
EEPROM 中断	59
编程注意事项	60
振荡器	61
振荡器概述	61
系统时钟配置	61
外部晶体 / 陶瓷振荡器 – HXT	62
内部 RC 振荡器 – HIRC	62
外部 32.768kHz 晶体振荡器 – LXT	62
内部 32kHz 振荡器 – LIRC	63
工作模式和系统时钟	64
系统时钟	64
系统工作模式	65
控制寄存器	66
工作模式切换	68
待机电流注意事项	72
唤醒	72
看门狗定时器	73
看门狗定时器时钟源	73
看门狗定时器控制寄存器	73
看门狗定时器操作	74

复位和初始化	75
复位功能	75
复位初始状态	79
输入 / 输出端口	84
上拉电阻	86
PA 口唤醒	86
输入 / 输出端口控制寄存器	87
I/O 口灌电流控制	87
I/O 口源电流控制 – HT66F4560	88
引脚共用功能	89
输入 / 输出引脚结构	103
编程注意事项	103
定时器模块 – TM	104
简介	104
TM 操作	104
TM 时钟源	104
TM 中断	105
TM 外部引脚	105
TM 输入 / 输出引脚选择	105
编程注意事项	106
标准型 TM – STM	107
标准型 TM 操作	107
标准型 TM 寄存器介绍	108
标准型 TM 工作模式	112
周期型 TM – PTM	122
周期型 TM 操作	122
周期型 TM 寄存器介绍	123
周期型 TM 工作模式	127
A/D 转换器	136
A/D 转换器简介	136
A/D 转换寄存器介绍	137
A/D 转换器操作	140
A/D 转换器参考电压	141
A/D 转换器输入信号	141
A/D 转换率及时序图	142
A/D 转换步骤概述	143
编程注意事项	144
A/D 转换功能	144
A/D 转换应用范例	145
串行接口模块 – SIM	147
SPI 接口	147
I ² C 接口	154

UART 串行接口 – HT66F4540/HT66F4550/HT66F4560	164
UART 外部引脚.....	164
UART 数据传输方案.....	165
UART 状态和控制寄存器.....	165
波特率发生器	170
UART 模块的设置与控制.....	170
UART 发送器.....	171
UART 接收器.....	172
接收错误处理	174
UART 模块中断结构.....	175
UART 模块暂停和唤醒.....	176
运算放大器与比较器	177
模拟功能操作	177
控制寄存器	177
失调校准程序	185
带 SCOM 功能的 LCD – 仅适用于 HT66F4540/HT66F4550/HT66F4560	187
LCD 操作	187
LCD 偏压电流控制	187
语音 D/A 转换器 – HT66F4550/HT66F4560	188
语音 D/A 转换器寄存器	188
低电压检测 – LVD	189
LVD 寄存器	189
LVD 操作	190
中断	191
中断寄存器	191
中断操作	199
外部中断	202
SD 比较器中断	203
A/D 转换器中断	203
时基中断	203
SIM 中断	205
LVD 中断	205
EEPROM 中断	205
UART 中断.....	205
多功能中断	205
TM 中断	206
中断唤醒功能	206
编程注意事项	206
配置选项	207
应用电路	207
指令集	208
简介	208
指令周期	208

数据的传送	208
算术运算	208
逻辑和移位运算	208
分支和控制转换	209
位运算	209
查表运算	209
其它运算	209
指令集概要	210
惯例	210
扩展指令集	213
指令定义	215
扩展指令定义	227
封装信息	237
16-pin NSOP (150mil) 外形尺寸	238
20-pin SSOP (150mil) 外形尺寸	239
24-pin SSOP (150mil) 外形尺寸	240
28-pin SSOP (150mil) 外形尺寸	241
48-pin LQFP (7mm × 7mm) 外形尺寸	242

特性

CPU 特性

- 工作电压：
 - ◆ $f_{\text{SYS}}=4\text{MHz}$: 2.2V~5.5V
 - ◆ $f_{\text{SYS}}=8\text{MHz}$: 2.2V~5.5V
 - ◆ $f_{\text{SYS}}=12\text{MHz}$: 2.2V~5.5V
- $V_{\text{DD}}=5\text{V}$, 系统时钟为 12MHz 时, 指令周期为 $0.33\mu\text{s}$
- 提供暂停和唤醒功能, 以降低功耗
- 振荡器类型
 - ◆ 外部高速晶振 – HXT
 - ◆ 内部高速 RC – HIRC
 - ◆ 外部低速 32.768kHz 晶振 – LXT
 - ◆ 内部低速 32kHz RC – LIRC
- 多种工作模式: 快速模式、低速模式、空闲模式和休眠模式
- 内部集成 2MHz、4MHz 和 8MHz 振荡器, 无需外接元件
- 所有指令都可在 1~3 个指令周期内完成
- 查表指令
- 115 条功能强大的指令系统
- 多达 16 层堆栈
- 位操作指令

周边特性

- 程序存储器: $2\text{K}\times 16\sim 16\text{K}\times 16$
- 数据存储器: $128\times 8\sim 512\times 8$
- True EEPROM 存储器: $32\times 8\sim 128\times 8$
- 看门狗定时器功能
- 多达 46 个双向 I/O 口
- 2 个与 I/O 口复用的外部中断输入
- 两个运算放大器, 两个比较器和三个 8-bit D/A 转换器
- 软件控制 4-SCOM 1/2 bias LCD 驱动 – 仅适用于 HT66F4540/HT66F4550/HT66F4560
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、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
- 串行接口模块 – 包含 SPI 和 I²C 接口
- 全双工异步通信接口 UART – 仅适用于 HT66F4540/HT66F4550/HT66F4560
- 高达 8 个外部通道 12-bit 分辨精度的 A/D 转换器
- 16-bit 高性能的语音 D/A 转换器 – 仅适用于 HT66F4550/HT66F4560
- 低电压复位功能
- 低电压检测功能
- 支持多种封装类型

概述

该系列单片机是一款 Flash 型具有 8 位高性能精简指令集的单片机，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列号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，该系列单片机包含一个多通道 12 位 A/D 转换器、一个 16 位语音 D/A 转换器、两个比较器、两个运算放大器和三个多通道 8 位 D/A 转换器。关于内部定时器，该系列单片机带有多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生等功能。内建完整的 SPI、I²C 和 UART 功能，为设计者提供了一个易与外部硬件通信的接口。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，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。

该系列单片机提供丰富的内部和外部高速、低速振荡器功能选项，且内建完整的系统振荡器，无需外围元器件。其不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

外加 I/O 使用灵活、一个完全集成的 LCD 驱动器和时基功能等其它特性，使该系列单片机可以广泛应用于各种产品中，例如烟雾探测器、电子测量仪器、环境监测、手持式测量工具、家庭应用、电子控制工具、马达控制等方面。

选型表

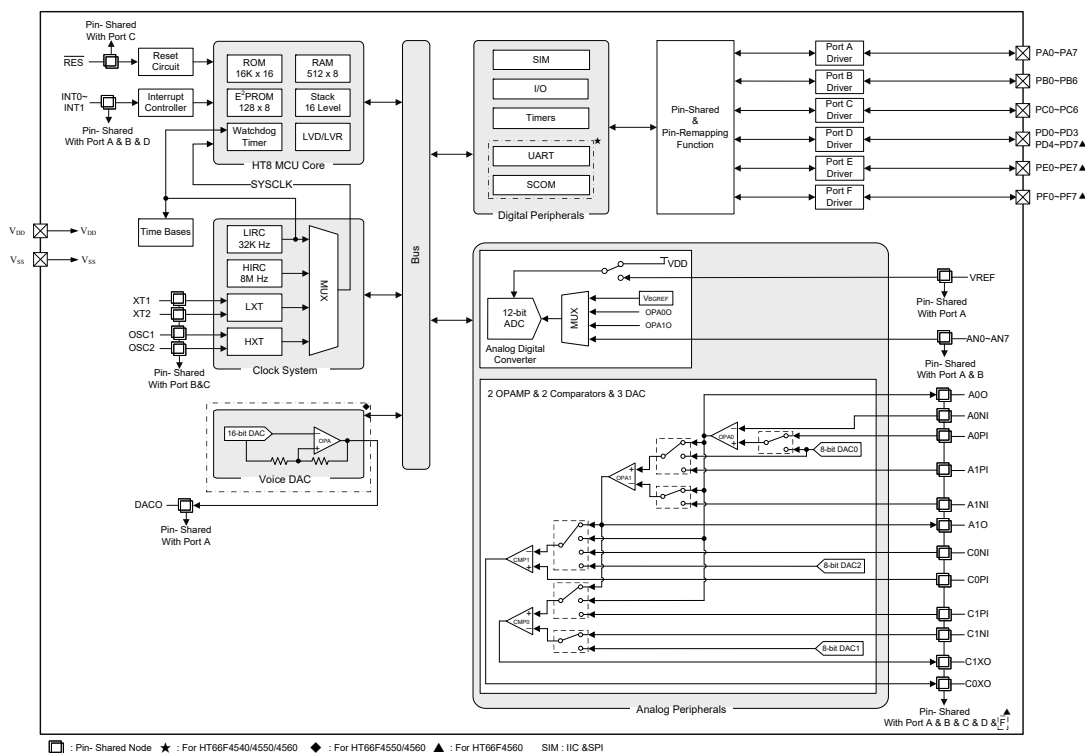
对于该系列的单片机而言，大多数的特性参数都是一样的。主要差异在于内存容量、I/O 数目、A/D 转换器输入通道数目，定时器模块的特点、SCOM、UART、语音 D/A 转换器、堆栈和封装类型。下表列出了各单片机的主要特性。

产品型号	程序存储器	数据存储器	数据 EEPROM	I/O	外部中断	A/D	定时器模块	时基
HT66F4530	2K×16	128×8	32×8	18	2	12-bit×5	10-bit STM×1 10-bit PTM×1	2
HT66F4540	4K×16	256×8	64×8	26	2	12-bit×8	10-bit STM×1 10-bit PTM×2	2
HT66F4550	8K×16	384×8	64×8	26	2	12-bit×8	10-bit STM×2 10-bit PTM×2	2
HT66F4560	16K×16	512×8	128×8	46	2	12-bit×8	10-bit STM×2 10-bit PTM×2	2

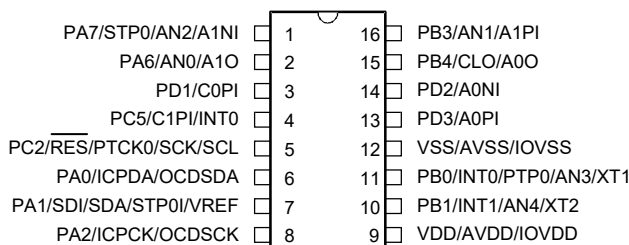
产品型号	SCOM	SIM	UART	比较器	运算放大器	D/A	语音 D/A	堆栈	封装
HT66F4530	×	√	×	2	2	8-bit×3	×	6	16NSOP 20SSOP
HT66F4540	4	√	√	2	2	8-bit×3	×	8	24/28SSOP
HT66F4550	4	√	√	2	2	8-bit×3	√	8	24/28SSOP
HT66F4560	4	√	√	2	2	8-bit×3	√	16	28SSOP 48LQFP

注：由于单片机存在多种封装形式，该表反映了最多引脚封装的情况。

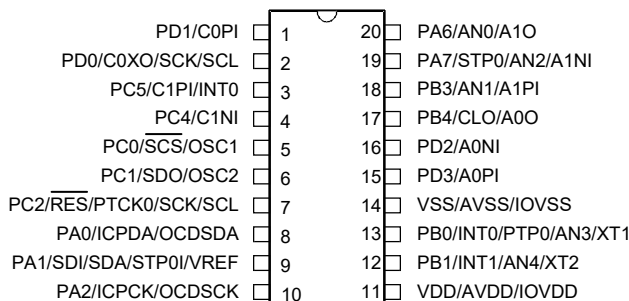
方框图



引脚图



HT66F4530/HT66F4540
16 NSOP-A



HT66F4530/HT66F4540
20 SSOP-A

PD1/C0PI	1	24	PA6/PTCK1/AN0/A1O
PC6/C0NI	2	23	PA7/STP0/AN2/A1NI
PC5/C1PI	3	22	PB3/AN1/A1PI
PC4/C1NI	4	21	PB4/CLO/A0O
PC0/SCS/OSC1	5	20	PD2/A0NI
PC1/SDO/OSC2	6	19	PD3/A0PI
PC2/RES/PTP1	7	18	VSS/AVSS/IOVSS
PA0/RX/ICPDA/OCSDA	8	17	PB0/PTP0/AN3/XT1
PA1/SDI/SDA/STP0I	9	16	PB1/INT1/AN4/XT2
PA2/TX/ICPCK/OCDSCK	10	15	VDD/AVDD/IOVDD
PA3/SCOM0/SCS	11	14	PA4/INT0/SCOM3/SCK/SCL/VREF
PB6/SCOM1/SDI/SDA/RX	12	13	PB5/SCOM2/TX/AN7

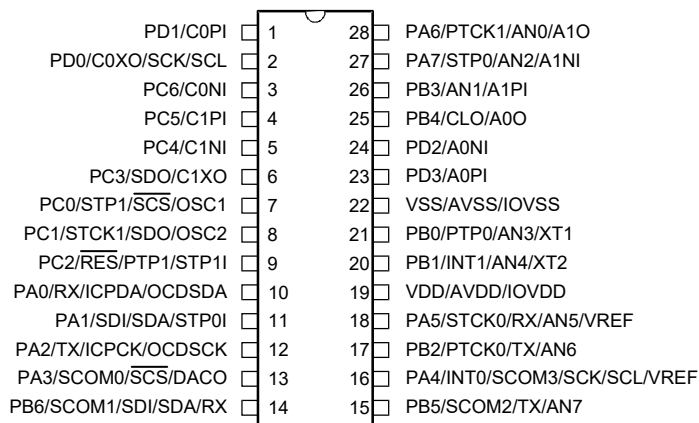
HT66F4540/HT66V4540
24 SSOP-A

PD1/C0PI	1	28	PA6/PTCK1/AN0/A1O
PD0/C0XO/SCK/SCL	2	27	PA7/STP0/AN2/A1NI
PC6/C0NI	3	26	PB3/AN1/A1PI
PC5/C1PI	4	25	PB4/CLO/A0O
PC4/C1NI	5	24	PD2/A0NI
PC3/SDO/C1XO	6	23	PD3/A0PI
PC0/SCS/OSC1	7	22	VSS/AVSS/IOVSS
PC1/SDO/OSC2	8	21	PB0/PTP0/AN3/XT1
PC2/RES/PTP1	9	20	PB1/INT1/AN4/XT2
PA0/RX/ICPDA/OCSDA	10	19	VDD/AVDD/IOVDD
PA1/SDI/SDA/STP0I	11	18	PA5/STCK0/RX/AN5/VREF
PA2/TX/ICPCK/OCDSCK	12	17	PB2/PTCK0/TX/AN6
PA3/SCOM0/SCS	13	16	PA4/INT0/SCOM3/SCK/SCL/VREF
PB6/SCOM1/SDI/SDA/RX	14	15	PB5/SCOM2/TX/AN7

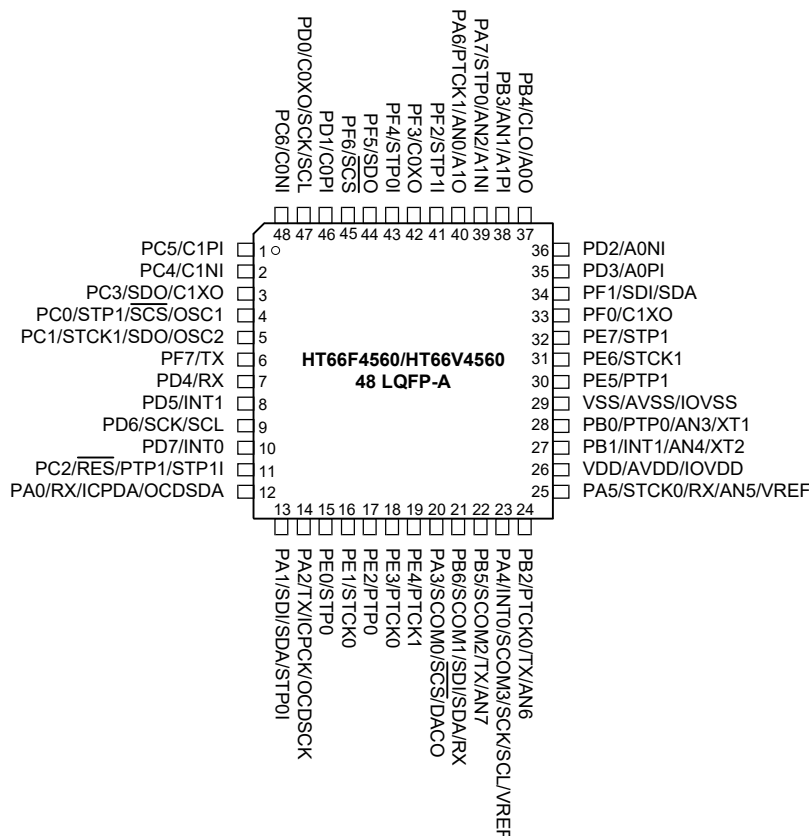
HT66F4540/HT66V4540
28 SSOP-A

PD1/C0PI	1	24	PA6/PTCK1/AN0/A1O
PC6/C0NI	2	23	PA7/STP0/AN2/A1NI
PC5/C1PI	3	22	PB3/AN1/A1PI
PC4/C1NI	4	21	PB4/CLO/A0O
PC0/STP1/SCS/OSC1	5	20	PD2/A0NI
PC1/STCK1/SDO/OSC2	6	19	PD3/A0PI
PC2/RES/PTP1/STP1I	7	18	VSS/AVSS/IOVSS
PA0/RX/ICPDA/OCSDA	8	17	PB0/PTP0/AN3/XT1
PA1/SDI/SDA/STP0I	9	16	PB1/INT1/AN4/XT2
PA2/TX/ICPCK/OCDSCK	10	15	VDD/AVDD/IOVDD
PA3/SCOM0/SCS/DACO	11	14	PA4/INT0/SCOM3/SCK/SCL/VREF
PB6/SCOM1/SDI/SDA/RX	12	13	PB5/SCOM2/TX/AN7

HT66F4550/HT66V4550
24 SSOP-A



HT66F4550/HT66V4550
HT66F4560/HT66V4560
28 SSOP-A



HT66F4560/HT66V4560
48 LQFP-A

- 注：1. 若共用脚同时有多种输出，所需引脚共用功能通过相应的软件控制位决定。
2. OCSDA 和 OCDSCK 引脚为片上调试功能专用引脚，仅存在于 HT66F45x0 的 OCDS EV 芯片 HT66V45x0。
3. 在较小封装中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入/输出端口”章节。

引脚说明

除了电源引脚外，此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。

引脚说明表格所列情况是针对最多引脚的封装，并非所有引脚都适用于小封装。

HT66F4530

引脚名称	功能	OPT	I/T	O/T	说明
PA0/ICPDA/ OCDSDA	PA0	PAPU PAWU 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
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/SDI/SDA/ STP0I/VREF	PA1	PAPU PAWU 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
	SDI	PAS0 IFS0	ST	—	SPI 串行数据输入
	SDA	PAS0 IFS0	ST	NMOS	I ² C 数据线
	STP0I	PAS0	ST	—	STM0 捕捉输入
	VREF	PAS0	AN	—	A/D 转换器参考电压输入
PA2/ICPCK/ OCDSCK	PA2	PAPU PAWU 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
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA6/AN0/A1O	PA6	PAPU PAWU 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
	AN0	PAS1	AN	—	A/D 转换器输入通道 0
	A1O	PAS1	—	AN	运算放大器 1 输出
PA7/STP0/ AN2/A1NI	PA7	PAPU PAWU 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
	STP0	PAS1	—	CMOS	STM0 输出
	AN2	PAS1	AN	—	A/D 转换器输入通道 2
	A1NI	PAS1	AN	—	运算放大器 1 反相输入引脚

引脚名称	功能	OPT	I/T	O/T	说明
PB0/INT0/ PTP0/AN3/XT1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT0	PBS0 IFS0	ST	—	外部中断 0
	PTP0	PBS0	—	CMOS	PTM0 输出
	AN3	PBS0	AN	—	A/D 转换器输入通道 3
	XT1	PBS0	LXT	—	LXT 引脚
PB1/INT1/ AN4/XT2	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS0	ST	—	外部中断 1
	AN4	PBS0	AN	—	A/D 转换器输入通道 4
	XT2	PBS0	—	LXT	LXT 引脚
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器输入通道 1
	A1PI	PBS0	AN	—	运算放大器 1 同相输入引脚
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CLO	PBS1	—	CMOS	系统时钟输出
	A0O	PBS1	—	AN	运算放大器 0 输出
PC0/ $\overline{\text{SCS}}$ /OSC1	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{SCS}}$	PCS0 IFS0	ST	CMOS	SPI 从机选择引脚
	OSC1	PCS0	HXT	—	HXT 引脚
PC1/SDO/ OSC2	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	OSC2	PCS0	—	HXT	HXT 引脚
PC2/ $\overline{\text{RES}}$ / PTCK0/ SCK/SCL	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{RES}}$	PCS0	ST	—	外部复位输入
	PTCK0	PCS0	ST	—	PTM0 时钟输入
	SCK	PCS0 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PCS0 IFS0	ST	NMOS	I ² C 时钟线
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1NI	PCS1	AN	—	比较器 1 反相输入引脚

引脚名称	功能	OPT	I/T	O/T	说明
PC5/C1PI/INT0	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1PI	PCS1	AN	—	比较器 1 同相输入引脚
	INT0	PCS1 IFS0	ST	—	外部中断 0
PD0/C0XO/ SCK/SCL	PD0	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0XO	PDS0	—	AN	比较器 0 输出
	SCK	PDS0 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PDS0 IFS0	ST	NMOS	I ² C 时钟线
PD1/C0PI	PD1	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0PI	PDS0	AN	—	比较器 0 同相输入引脚
PD2/A0NI	PD2	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0NI	PDS0	AN	—	运算放大器 0 反相输入引脚
PD3/A0PI	PD3	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0PI	PDS0	AN	—	运算放大器 0 同相输入引脚
VDD/AVDD/ IOVDD*	VDD	—	PWR	—	电源电压
	AVDD	—	PWR	—	A/D 转换器电源电压
	IOVDD	—	PWR	—	I/O 端口电源电压
VSS/AVSS/ IOVSS**	VSS	—	PWR	—	地
	AVSS	—	PWR	—	A/D 转换器地
	IOVSS	—	PWR	—	I/O 端口地

HT66F4540

引脚名称	功能	OPT	I/T	O/T	说明
PA0/RX/ ICPDA/ OCSDSA	PA0	PAPU PAWU 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
	RX	PAS0 IFS0	ST	—	UART 接收引脚
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCSDSA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/SDI/SDA/ STP0I	PA1	PAPU PAWU 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
	SDI	PAS0 IFS0	ST	—	SPI 串行数据输入
	SDA	PAS0 IFS0	ST	NMOS	I ² C 数据线
	STP0I	PAS0	ST	—	STM0 捕捉输入
PA2/TX/ ICPCK/ OCDSCK	PA2	PAPU PAWU 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
	TX	PAS0	—	CMOS	UART 发送引脚
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/SCOM0/ SCS	PA3	PAPU PAWU 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
	SCOM0	PAS0	—	AN	软件控制 1/2 bias LCD COM
	SCS	PAS0 IFS0	ST	CMOS	SPI 从机选择引脚
PA4/INT0/ SCOM3/ SCK/SCL/ VREF	PA4	PAPU PAWU 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	PAS1	ST	—	外部中断 0
	SCOM3	PAS1	—	AN	软件控制 1/2 bias LCD COM
	SCK	PAS1 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PAS1 IFS0	ST	NMOS	I ² C 时钟线
	VREF	PAS1	AN	—	A/D 转换器参考电压输入

引脚名称	功能	OPT	I/T	O/T	说明
PA5/STCK0/ RX/AN5/VREF	PA5	PAPU PAWU 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
	STCK0	PAS1	ST	—	STM0 时钟输入
	RX	PAS1 IFS0	ST	—	UART 接收引脚
	AN5	PAS1	AN	—	A/D 转换器输入通道 5
	VREF	PAS1	AN	—	A/D 转换器参考电压输入
PA6/PTCK1/ AN0/A1O	PA6	PAPU PAWU 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
	PTCK1	PAS1	ST	—	PTM1 时钟输入
	AN0	PAS1	AN	—	A/D 转换器输入通道 0
	A1O	PAS1	—	AN	运算放大器 1 输出
PA7/STP0/ AN2/A1NI	PA7	PAPU PAWU 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
	STP0	PAS1	—	CMOS	STM0 输出
	AN2	PAS1	AN	—	A/D 转换器输入通道 2
	A1NI	PAS1	AN	—	运算放大器 1 反相输入引脚
PB0/PTP0/ AN3/XT1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP0	PBS0	—	CMOS	PTM0 输出
	AN3	PBS0	AN	—	A/D 转换器输入通道 3
	XT1	PBS0	LXT	—	LXT 引脚
PB1/INT1/ AN4/XT2	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS0	ST	—	外部中断 1
	AN4	PBS0	AN	—	A/D 转换器输入通道 4
	XT2	PBS0	—	LXT	LXT 引脚
PB2/PTCK0/ TX/AN6	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK0	PBS0	ST	—	PTM0 时钟输入
	TX	PBS0	—	CMOS	UART 发送引脚
	AN6	PBS0	AN	—	A/D 转换器输入通道 6
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器输入通道 1
	A1PI	PBS0	AN	—	运算放大器 1 同相输入引脚
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CLO	PBS1	—	CMOS	系统时钟输出
	A0O	PBS1	—	AN	运算放大器 0 输出

引脚名称	功能	OPT	I/T	O/T	说明
PB5/SCOM2/ TX/AN7	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM2	PBS1	—	AN	软件控制 1/2 bias LCD COM
	TX	PBS1	—	CMOS	UART 发送引脚
	AN7	PBS1	AN	—	A/D 转换器输入通道 7
PB6/SCOM1/ SDI/SDA/RX	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM1	PBS1	—	AN	软件控制 1/2 bias LCD COM
	SDI	PBS1 IFS0	ST	—	SPI 串行数据输入
	SDA	PBS1 IFS0	ST	NMOS	I ² C 数据线
	RX	PBS1 IFS0	ST	—	UART 接收引脚
PC0/ $\overline{\text{SCS}}$ /OSC1	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{SCS}}$	PCS0 IFS0	ST	CMOS	SPI 从机选择引脚
	OSC1	PCS0	HXT	—	HXT 引脚
PC1/SDO/ OSC2	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	OSC2	PCS0	—	HXT	HXT 引脚
PC2/ $\overline{\text{RES}}$ /PTP1	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{RES}}$	PCS0	ST	—	外部复位引脚
	PTP1	PCS0	—	CMOS	PTM1 输出
PC3/SDO/ C1XO	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	C1XO	PCS0	—	AN	比较器 1 输出
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1NI	PCS1	AN	—	比较器 1 反相输入引脚
PC5/C1PI	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1PI	PCS1	AN	—	比较器 1 同相输入引脚
PC6/C0NI	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0NI	PCS1	AN	—	比较器 0 反相输入引脚

引脚名称	功能	OPT	I/T	O/T	说明
PD0/C0XO/ SCK/SCL	PD0	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0XO	PDS0	—	AN	比较器 0 输出
	SCK	PDS0 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PDS0 IFS0	ST	NMOS	I ² C 时钟线
PD1/C0PI	PD1	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0PI	PDS0	AN	—	比较器 0 同相输入引脚
PD2/A0NI	PD2	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0NI	PDS0	AN	—	运算放大器 0 反相输入引脚
PD3/A0PI	PD3	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0PI	PDS0	AN	—	运算放大器 0 同相输入引脚
VDD/AVDD/ IOVDD*	VDD	—	PWR	—	电源电压
	AVDD	—	PWR	—	A/D 转换器电源电压
	IOVDD	—	PWR	—	I/O 端口电源电压
VSS/AVSS/ IOVSS**	VSS	—	PWR	—	地
	AVSS	—	PWR	—	A/D 转换器地
	IOVSS	—	PWR	—	I/O 端口地

HT66F4550

引脚名称	功能	OPT	I/T	O/T	说明
PA0/RX/ ICPDA/ OCDSDA	PA0	PAP PAW 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
	RX	PAS0 IFS0	ST	—	UART 接收引脚
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/SDI/SDA/ STP0I	PA1	PAP PAW 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
	SDI	PAS0 IFS0	ST	—	SPI 串行数据输入
	SDA	PAS0 IFS0	ST	NMOS	I ² C 数据线
	STP0I	PAS0	ST	—	STM0 捕捉输入

引脚名称	功能	OPT	I/T	O/T	说明
PA2/TX/ ICPCK/ OCDSCK	PA2	PAPU PAWU 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
	TX	PAS0	—	CMOS	UART 发送引脚
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/SCOM0/ SCS/DACO	PA3	PAPU PAWU 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
	SCOM0	PAS0	—	AN	软件控制 1/2 bias LCD COM
	SCS	PAS0 IFS0	ST	CMOS	SPI 从机选择引脚
	DACO	PAS0	—	AN	语音 D/A 转换器输出
PA4/INT0/ SCOM3/ SCK/SCL/ VREF	PA4	PAPU PAWU 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	PAS1	ST	—	外部中断 0
	SCOM3	PAS1	—	AN	软件控制 1/2 bias LCD COM
	SCK	PAS1 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PAS1 IFS0	ST	NMOS	I ² C 时钟线
	VREF	PAS1	AN	—	A/D 转换器参考电压输入
PA5/STCK0/ RX/AN5/VREF	PA5	PAPU PAWU 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
	STCK0	PAS1	ST	—	STM0 时钟输入
	RX	PAS1 IFS0	ST	—	UART 接收引脚
	AN5	PAS1	AN	—	A/D 转换器输入通道 5
	VREF	PAS1	AN	—	A/D 转换器参考电压输入
PA6/PTCK1/ AN0/A1O	PA6	PAPU PAWU 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
	PTCK1	PAS1	ST	—	PTM1 时钟输入
	AN0	PAS1	AN	—	A/D 转换器输入通道 0
	A1O	PAS1	—	AN	运算放大器 1 输出
PA7/STP0/ AN2/A1NI	PA7	PAPU PAWU 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
	STP0	PAS1	—	CMOS	STM0 输出
	AN2	PAS1	AN	—	A/D 转换器输入通道 2
	A1NI	PAS1	AN	—	运算放大器 1 反相输入引脚

引脚名称	功能	OPT	I/T	O/T	说明
PB0/PTP0/ AN3/XT1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP0	PBS0	—	CMOS	PTM0 输出
	AN3	PBS0	AN	—	A/D 转换器输入通道 3
	XT1	PBS0	LXT	—	LXT 引脚
PB1/INT1/ AN4/XT2	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS0	ST	—	外部中断 1
	AN4	PBS0	AN	—	A/D 转换器输入通道 4
	XT2	PBS0	—	LXT	LXT 引脚
PB2/PTCK0/ TX/AN6	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK0	PBS0	ST	—	PTM0 时钟输入
	TX	PBS0	—	CMOS	UART 发送引脚
	AN6	PBS0	AN	—	A/D 转换器输入通道 6
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器输入通道 1
	A1PI	PBS0	AN	—	运算放大器 1 同相输入引脚
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CLO	PBS1	—	CMOS	系统时钟输出
	A0O	PBS1	—	AN	运算放大器 0 输出
PB5/SCOM2/ TX/AN7	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM2	PBS1	—	AN	软件控制 1/2 bias LCD COM
	TX	PBS1	—	CMOS	UART 发送引脚
	AN7	PBS1	AN	—	A/D 转换器输入通道 7
PB6/SCOM1/ SDI/SDA/RX	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM1	PBS1	—	AN	软件控制 1/2 bias LCD COM
	SDI	PBS1 IFS0	ST	—	SPI 串行数据输入
	SDA	PBS1 IFS0	ST	NMOS	I ² C 数据线
	RX	PBS1 IFS0	ST	—	UART 接收引脚
PC0/STP1/ SCS/OSC1	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP1	PCS0	—	CMOS	STM1 输出
	SCS	PCS0 IFS0	ST	CMOS	SPI 从机选择引脚
	OSC1	PCS0	HXT	—	HXT 引脚

引脚名称	功能	OPT	I/T	O/T	说明
PC1/STCK1/ SDO/OSC2	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK1	PCS0	ST	—	STM1 时钟输入
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	OSC2	PCS0	—	HXT	HXT 引脚
PC2/ $\overline{\text{RES}}$ / PTP1/STP1I	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{RES}}$	PCS0	ST	—	外部复位引脚
	PTP1	PCS0	—	CMOS	PTM1 输出
	STP1I	PCS0	ST	—	STM1 捕捉输入
PC3/SDO/ C1XO	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	C1XO	PCS0	—	AN	比较器 1 输出
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1NI	PCS1	AN	—	比较器 1 反相输入引脚
PC5/C1PI	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1PI	PCS1	AN	—	比较器 1 同相输入引脚
PC6/C0NI	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0NI	PCS1	AN	—	比较器 0 反相输入引脚
PD0/C0XO/ SCK/SCL	PD0	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0XO	PDS0	—	AN	比较器 0 输出
	SCK	PDS0 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PDS0 IFS0	ST	NMOS	I ² C 时钟线
PD1/C0PI	PD1	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0PI	PDS0	AN	—	比较器 0 同相输入引脚
PD2/A0NI	PD2	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0NI	PDS0	AN	—	运算放大器 0 反相输入引脚
PD3/A0PI	PD3	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0PI	PDS0	AN	—	运算放大器 0 同相输入引脚
VDD/AVDD/ IOVDD*	VDD	—	PWR	—	电源电压
	AVDD	—	PWR	—	A/D 转换器电源电压
	IOVDD	—	PWR	—	I/O 端口电源电压

引脚名称	功能	OPT	I/T	O/T	说明
VSS/AVSS/ IOVSS**	VSS	—	PWR	—	地
	AVSS	—	PWR	—	A/D 转换器地
	IOVSS	—	PWR	—	I/O 端口地

HT66F4560

引脚名称	功能	OPT	I/T	O/T	说明
PA0/RX/ ICPDA/ OCSDA	PA0	PAPU PAWU 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
	RX	PAS0 IFS0	ST	—	UART 接收引脚
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/SDI/SDA/ STP0I	PA1	PAPU PAWU 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
	SDI	PAS0 IFS0	ST	—	SPI 串行数据输入
	SDA	PAS0 IFS0	ST	NMOS	I ² C 数据线
	STP0I	PAS0 IFS2	ST	—	STM0 捕捉输入
PA2/TX/ ICPCK/ OCDSCK	PA2	PAPU PAWU 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
	TX	PAS0	—	CMOS	UART 发送引脚
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/SCOM0/ SCS/DACO	PA3	PAPU PAWU 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
	SCOM0	PAS0	—	AN	软件控制 1/2 bias LCD COM
	SCS	PAS0 IFS0	ST	CMOS	SPI 从机选择引脚
	DACO	PAS0	—	AN	语音 D/A 转换器输出

引脚名称	功能	OPT	I/T	O/T	说明
PA4/INT0/ SCOM3/ SCK/SCL/ VREF	PA4	PAPU PAWU 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	PAS1 IFS1	ST	—	外部中断 0
	SCOM3	PAS1	—	AN	软件控制 1/2 bias LCD COM
	SCK	PAS1 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PAS1 IFS0	ST	NMOS	I ² C 时钟线
	VREF	PAS1	AN	—	A/D 转换器参考电压输入
PA5/STCK0/ RX/AN5/VREF	PA5	PAPU PAWU 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
	STCK0	PAS1 IFS2	ST	—	STM0 时钟输入
	RX	PAS1 IFS0	ST	—	UART 接收引脚
	AN5	PAS1	AN	—	A/D 转换器输入通道 5
	VREF	PAS1	AN	—	A/D 转换器参考电压输入
PA6/PTCK1/ AN0/A1O	PA6	PAPU PAWU 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
	PTCK1	PAS1 IFS1	ST	—	PTM1 时钟输入
	AN0	PAS1	AN	—	A/D 转换器输入通道 0
	A1O	PAS1	—	AN	运算放大器 1 输出
PA7/STP0/ AN2/A1NI	PA7	PAPU PAWU 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
	STP0	PAS1	—	CMOS	STM0 输出
	AN2	PAS1	AN	—	A/D 转换器输入通道 2
	A1NI	PAS1	AN	—	运算放大器 1 反相输入引脚
PB0/PTP0/ AN3/XT1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP0	PBS0	—	CMOS	PTM0 输出
	AN3	PBS0	AN	—	A/D 转换器输入通道 3
	XT1	PBS0	LXT	—	LXT 引脚
PB1/INT1/ AN4/XT2	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS0 IFS1	ST	—	外部中断 1
	AN4	PBS0	AN	—	A/D 转换器输入通道 4
	XT2	PBS0	—	LXT	LXT 引脚

引脚名称	功能	OPT	I/T	O/T	说明
PB2/PTCK0/ TX/AN6	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK0	PBS0 IFS1	ST	—	PTM0 时钟输入
	TX	PBS0	—	CMOS	UART 发送引脚
	AN6	PBS0	AN	—	A/D 转换器输入通道 6
PB3/AN1/A1PI	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器输入通道 1
	A1PI	PBS0	AN	—	运算放大器 1 同相输入引脚
PB4/CLO/A0O	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CLO	PBS1	—	CMOS	系统时钟输出
	A0O	PBS1	—	AN	运算放大器 0 输出
PB5/SCOM2/ TX/AN7	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM2	PBS1	—	AN	软件控制 1/2 bias LCD COM
	TX	PBS1	—	CMOS	UART 发送引脚
	AN7	PBS1	AN	—	A/D 转换器输入通道 7
PB6/SCOM1/ SDI/SDA/RX	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM1	PBS1	—	AN	软件控制 1/2 bias LCD COM
	SDI	PBS1 IFS0	ST	—	SPI 串行数据输入
	SDA	PBS1 IFS0	ST	NMOS	I ² C 数据线
	RX	PBS1 IFS0	ST	—	UART 接收引脚
PC0/STP1/ SCS/OSC1	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP1	PCS0	—	CMOS	STM1 输出
	SCS	PCS0 IFS0	ST	CMOS	SPI 从机选择引脚
	OSC1	PCS0	HXT	—	HXT 引脚
PC1/STCK1/ SDO/OSC2	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK1	PCS0 IFS2	ST	—	STM1 时钟输入
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	OSC2	PCS0	—	HXT	HXT 引脚

引脚名称	功能	OPT	I/T	O/T	说明
PC2/ $\overline{\text{RES}}$ / PTP1/STP1I	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{RES}}$	PCS0	ST	—	外部复位引脚
	PTP1	PCS0	—	CMOS	PTM1 输出
	STP1I	PCS0 IFS2	ST	—	STM1 捕捉输入
PC3/SDO/ C1XO	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PCS0	—	CMOS	SPI 串行数据输出
	C1XO	PCS0	—	AN	比较器 1 输出
PC4/C1NI	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1NI	PCS1	AN	—	比较器 1 反相输入引脚
PC5/C1PI	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1PI	PCS1	AN	—	比较器 1 同相输入引脚
PC6/C0NI	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0NI	PCS1	AN	—	比较器 0 反相输入引脚
PD0/C0XO/ SCK/SCL	PD0	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0XO	PDS0	—	AN	比较器 0 输出
	SCK	PDS0 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PDS0 IFS0	ST	NMOS	I ² C 时钟线
PD1/C0PI	PD1	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0PI	PDS0	AN	—	比较器 0 同相输入引脚
PD2/A0NI	PD2	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0NI	PDS0	AN	—	运算放大器 0 反相输入引脚
PD3/A0PI	PD3	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	A0PI	PDS0	AN	—	运算放大器 0 同相输入引脚
PD4/RX	PD4	PDP PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	RX	PDS1 IFS0	ST	—	UART 接收引脚
PD5/INT1	PD5	PDP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	IFS1	ST	—	外部中断 1

引脚名称	功能	OPT	I/T	O/T	说明
PD6/SCK/SCL	PD6	PDP PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCK	PDS1 IFS0	ST	CMOS	SPI 串行时钟
	SCL	PDS1 IFS0	ST	NMOS	I ² C 时钟线
PD7/INT0	PD7	PDP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT0	IFS1	ST	—	外部中断 0
PE0/STP0	PE0	PEP PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP0	PES0	—	CMOS	STM0 输出
PE1/STCK0	PE1	PEP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK0	IFS2	ST	—	STM0 时钟输入
PE2/PTP0	PE2	PEP PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP0	PES0	—	CMOS	PTM0 输出
PE3/PTCK0	PE3	PEP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK0	IFS1	ST	—	PTM0 时钟输入
PE4/PTCK1	PE4	PEP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK1	IFS1	ST	—	PTM1 时钟输入
PE5/PTP1	PE5	PEP PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1	PES1	—	CMOS	PTM1 输出
PE6/STCK1	PE6	PEP	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK1	IFS2	ST	—	STM1 时钟输入
PE7/STP1	PE7	PEP PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP1	PES1	—	CMOS	STM1 输出
PF0/C1X0	PF0	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C1X0	PFS0	—	AN	比较器 1 输出
PF1/SDI/SDA	PF1	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PFS0 IFS0	ST	—	SPI 串行数据输入
	SDA	PFS0 IFS0	ST	NMOS	I ² C 数据线
PF2/STP1I	PF2	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP1I	IFS2	ST	—	STM1 捕捉输入
PF3/C0X0	PF3	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	C0X0	PFS0	—	AN	比较器 0 输出

引脚名称	功能	OPT	I/T	O/T	说明
PF4/STP0I	PF4	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STP0I	IFS2	ST	—	STM0 捕捉输入
PF5/SDO	PF5	PFPU PFS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PFS1	—	CMOS	SPI 串行数据输出
PF6/ $\overline{\text{SCS}}$	PF6	PFPU PFS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	$\overline{\text{SCS}}$	PFS1 IFS0	ST	CMOS	SPI 从机选择引脚
PF7/TX	PF7	PFPU PFS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	TX	PFS1	—	CMOS	UART 发送引脚
VDD/AVDD/ IOVDD*	VDD	—	PWR	—	电源电压
	AVDD	—	PWR	—	A/D 转换器电源电压
	IOVDD	—	PWR	—	I/O 端口电源电压
VSS/AVSS/ IOVSS**	VSS	—	PWR	—	地
	AVSS	—	PWR	—	A/D 转换器地
	IOVSS	—	PWR	—	I/O 端口地

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；

ST：施密特触发输入；

CMOS：CMOS 输出；

NMOS：NMOS 输出

AN：模拟信号；

HXT：高频晶体振荡器；

LXT：低频晶体振荡器；

*：AVDD、IOVDD 引脚和 VDD 在内部是同一个引脚。

**：AVSS、IOVSS 引脚和 VSS 在内部是同一个引脚。

极限参数

电源供应电压	$V_{SS}-0.3V \sim V_{SS}+6.0V$
端口输入电压	$V_{SS}-0.3V \sim V_{DD}+0.3V$
储存温度	$-50^{\circ}\text{C} \sim 125^{\circ}\text{C}$
工作温度	$-40^{\circ}\text{C} \sim 85^{\circ}\text{C}$
I_{OL} 总电流	80mA
I_{OH} 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

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作温度、工作频率、引脚负载状况、温度和程序指令等等。

工作电压特性

$T_a = -40^{\circ}\text{C} \sim 85^{\circ}\text{C}$

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压 – HXT	—	$f_{SYS} = f_{HXT} = 4\text{MHz}$	2.2	—	5.5	V
			$f_{SYS} = f_{HXT} = 8\text{MHz}$	2.2	—	5.5	
			$f_{SYS} = f_{HXT} = 12\text{MHz}$	2.7	—	5.5	
	工作电压 – HIRC	—	$f_{SYS} = f_{HIRC} = 2\text{MHz}$	2.2	—	5.5	V
			$f_{SYS} = f_{HIRC} = 4\text{MHz}$	2.2	—	5.5	
			$f_{SYS} = f_{HIRC} = 8\text{MHz}$	2.2	—	5.5	
	工作电压 – LXT	—	$f_{SYS} = f_{LXT} = 32.768\text{kHz}$	2.2	—	5.5	V
	工作电压 – LIRC	—	$f_{SYS} = f_{LIRC} = 32\text{kHz}$	2.2	—	5.5	V

待机电流特性

Ta = 25°C

符号	待机模式	测试条件		最小	典型	最大	最大	单位
		V _{DD}	条件				85°C	
I _{STB}	休眠模式	2.2V	WDT off	—	0.2	0.6	0.7	μA
		3V		—	0.2	0.8	1	
		5V		—	0.5	1	1.2	
		2.2V	WDT on	—	1.2	2.4	2.9	μA
		3V		—	1.5	3	3.6	
		5V		—	3	5	6	
	空闲模式 0 – LIRC	2.2V	f _{SUB} on	—	2.4	4	4.8	μA
		3V		—	3	5	6	
		5V		—	5	10	12	
	空闲模式 0 – LXT	2.2V	f _{SUB} on	—	2.4	4	4.8	μA
		3V		—	3	5	6	
		5V		—	5	10	12	
	空闲模式 1 – HIRC	2.2V	f _{SUB} on, f _{SYS} = 2MHz	—	60	90	90	μA
		3V		—	90	135	135	
		5V		—	180	270	270	
		2.2V	f _{SUB} on, f _{SYS} = 4MHz	—	144	200	240	μA
		3V		—	180	250	300	
		5V		—	400	600	720	
		2.2V	f _{SUB} on, f _{SYS} = 8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	
	空闲模式 1 – HXT	2.2V	f _{SUB} on, f _{SYS} = 4MHz	—	144	200	240	μA
		3V		—	180	250	300	
		5V		—	400	600	720	
		2.2V	f _{SUB} on, f _{SYS} = 8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	
		2.7V	f _{SUB} on, f _{SYS} = 12MHz	—	432	600	720	μA
		3V		—	540	750	900	
		5V		—	800	1200	1440	

注：当使用该表格电气特性数据时，以下几点需注意：

- 任何数字输入都设置为非浮空的状态。
- 所有测量都在无负载且所有外围功能关闭的条件下进行。
- 无直流电流路径。
- 所有待机电流数值都是在 HALT 指令执行后测得，因此 HALT 后停止执行所有指令。

工作电流特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{DD}	低速模式 – LIRC	2.2V	f _{sys} = 32kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	低速模式 – LXT	2.2V	f _{sys} = 32.768kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	快速模式 – HIRC	2.2V	f _{sys} = 2MHz	—	0.14	0.21	mA
		3V		—	0.2	0.3	
		5V		—	0.4	0.6	
		2.2V	f _{sys} = 4MHz	—	0.3	0.5	mA
		3V		—	0.4	0.6	
		5V		—	0.8	1.2	
		2.2V	f _{sys} = 8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	
	快速模式 – HXT	2.2V	f _{sys} = 4MHz	—	0.4	0.6	mA
		3V		—	0.5	0.75	
		5V		—	1	1.5	
		2.2V	f _{sys} = 8MHz	—	0.8	1.2	mA
		3V		—	1	1.5	
		5V		—	2	3	
		2.7V	f _{sys} = 12MHz	—	1.2	2.2	mA
		3V		—	1.5	2.75	
		5V		—	3	4.5	

注：当使用该表格电气特性数据时，以下几点需注意：

- 任何数字输入都设置为非浮空的状态。
- 所有测量都在无负载且所有外围功能关闭的条件下进行。
- 无直流电流路径。
- 所有工作电流数值都是通过连续的 NOP 指令循环测得。

交流电气特性

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作电压、工作频率、和温度等等。

内部高速振荡器 – HIRC – 频率精确度

程序烧录时，烧录器可调整 HIRC 振荡器使其工作在用户选择的 HIRC 频率和工作电压（3V 或 5V）条件下。

3V Trim

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{HIRC}	通过烧录器调整后的 2MHz HIRC 频率	3V	Ta = 25°C	-1%	2	+1%	MHz
			Ta = -40°C~85°C	-2%	2	+2%	
		2.2~5.5V	Ta = 25°C	-4%	2	+4%	
			Ta = -40°C~85°C	-6%	2	+6%	
	通过烧录器调整后的 4MHz HIRC 频率	3V	Ta = 25°C	-1%	4	+1%	MHz
			Ta = -40°C~85°C	-2%	4	+2%	
		2.2~5.5V	Ta = 25°C	-3%	4	+3%	
			Ta = -40°C~85°C	-4%	4	+4%	
	通过烧录器调整后的 8MHz HIRC 频率	3V	Ta = 25°C	-1%	8	+1%	MHz
			Ta = -40°C~85°C	-2%	8	+2%	
		2.2~5.5V	Ta = 25°C	-2.5%	8	+2.5%	
			Ta = -40°C~85°C	-3%	8	+3%	

5V Trim

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{HIRC}	通过烧录器调整后的 2MHz HIRC 频率	5V	Ta = 25°C	-1%	2	+1%	MHz
			Ta = -40°C~85°C	-2%	2	+2%	
		2.2~5.5V	Ta = 25°C	-7%	2	+7%	
			Ta = -40°C~85°C	-8%	2	+8%	
	通过烧录器调整后的 4MHz HIRC 频率	5V	Ta = 25°C	-1%	4	+1%	MHz
			Ta = -40°C~85°C	-2%	4	+2%	
		2.2~5.5V	Ta = 25°C	-5%	4	+5%	
			Ta = -40°C~85°C	-6%	4	+6%	
	通过烧录器调整后的 8MHz HIRC 频率	5V	Ta = 25°C	-1%	8	+1%	MHz
			Ta = -40°C~85°C	-2%	8	+2%	
		2.2~5.5V	Ta = 25°C	-2.5%	8	+2.5%	
			Ta = -40°C~85°C	-4%	8	+4%	

注：1. 烧录器可在 3V/5V 这两个可选的固定电压下对 HIRC 频率进行调整，在此提供 3V Trim 及 5V Trim 时的参数值。

2. 表格中提供的最小和最大误差值仅在对应的烧录器调整频率下有效。当烧录器已将 HIRC 调整为某一固定频率，此后再通过程序中振荡器控制位将其频率改为其它值时，频率误差范围将增加到 ±20%。

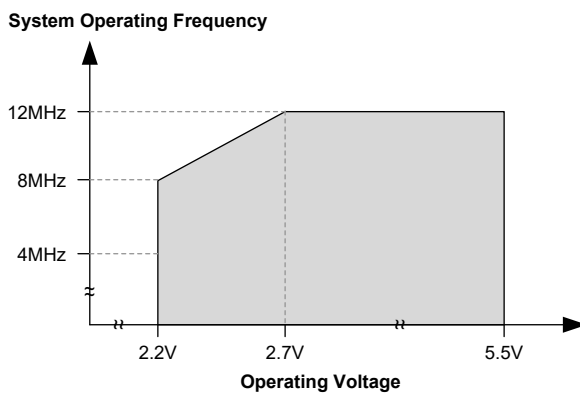
3. 应用于 2.2~5.5V 下，建议使用 3V Trim HIRC 频率误差范围比 5V Trim HIRC 小。

内部低速振荡器电气特性 – LIRC

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{LIRC}	LIRC 振荡器频率	2.2V~5.5V	T _a = 25°C	-5%	32	+5%	kHz
			T _a = -40°C~85°C	-10%	32	+10%	
t _{START}	LIRC 启动时间	—	—	—	—	100	μs

注：在 V_{DD}=3V 条件下调整这些频率，且仅在 V_{DD}=3V 条件下验证这些参数。

工作频率电气特性曲线图



系统上电时间电气特性

Ta = -40°C ~ 85°C

符号	参数	测试条件	最小	典型	最大	单位
t _{SST}	系统启动时间 (从 f _{sys} off 的状态下唤醒)	f _{sys} = f _H ~ f _H /64, f _H = f _{HXT}	—	128	—	t _{HXT}
		f _{sys} = f _H ~ f _H /64, f _H = f _{HIRC}	—	16	—	t _{HIRC}
		f _{sys} = f _{SUB} = f _{LXT}	—	1024	—	t _{LXT}
		f _{sys} = f _{SUB} = f _{LIRC}	—	2	—	t _{LIRC}
	系统启动时间 (从 f _{sys} on 的状态下唤醒)	f _{sys} = f _H ~ f _H /64, f _H = f _{HXT} 或 f _{HIRC}	—	2	—	t _H
		f _{sys} = f _{SUB} = f _{LXT} 或 f _{LIRC}	—	2	—	t _{SUB}
	系统速度切换时间 (快速模式 → 低速模式或 低速模式 → 快速模式)	f _{HXT} off → on	—	1024	—	t _{HXT}
		f _{HIRC} off → on	—	16	—	t _{HIRC}
		f _{LXT} off → on	—	1024	—	t _{LXT}
t _{RSTD}	系统复位延迟时间 (上电复位或 LVR 硬件复位)	RR _{POR} = 5V/ms	42	48	54	ms
	系统复位延迟时间 (LVRC/WDTC/RSTC 软件复位)	—				
	系统复位延迟时间 (WDT 溢出或 RES 引脚复位)	—	14	16	18	ms
t _{SRESET}	软件复位最小延迟脉宽	—	45	90	120	μs

- 注：1. 系统启动时间里提到的 f_{sys} on/off 状态取决于工作模式类型以及所选的系统时钟振荡器。更多相关细节请参考系统工作模式章节。
2. t_{HXT}、t_{HIRC} 等符号所表示的时间单位，是对应频率值的倒数，相关频率值在前面表格有说明。例如，t_{HIRC} = 1/f_{HIRC}，t_{sys} = 1/f_{sys} 等等。
3. 若 LIRC 被选择作为系统时钟源且在休眠模式下 LIRC 关闭，则上面表格中对应 t_{SST} 数值还需加上 LIRC 频率表格里提供的 LIRC 启动时间 t_{START}。
4. 系统速度切换时间实际上是指新使能的振荡器的启动时间。

输入 / 输出电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{IL}	除 RES 引脚外，I/O 口低电平输入电压	5V	—	0	—	1.5	V
		—		0	—	0.2V _{DD}	
	RES 引脚低电平输入电压	—	—	0	—	0.4V _{DD}	V
V _{IH}	除 RES 引脚外，I/O 口高电平输入电压	5V	—	3.5	—	5.0	V
		—		0.8V _{DD}	—	V _{DD}	
	RES 引脚高电平输入电压	—	—	0.9V _{DD}	—	V _{DD}	V

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OL}	除 PB0/PB1 引脚外, I/O 口灌电流	3V	V _{OL} = 0.1V _{DD}	16	32	—	mA
		5V		32	65	—	
	PB0/PB1 引脚灌电流	3V	V _{OL} = 0.3V _{DD} , SKLEDC[m+1, m] = 00B (m = 0 或 2 或 4 或 6)	37.5	75	—	mA
		5V		75	150	—	
		3V	V _{OL} = 0.3V _{DD} , SKLEDC[m+1, m] = 01B (m = 0 或 2 或 4 或 6)	50	100	—	mA
		5V		100	200	—	
		3V	V _{OL} = 0.3V _{DD} , SKLEDC[m+1, m] = 10B (m = 0 或 2 或 4 或 6)	62.5	125	—	mA
		5V		125	250	—	
		3V	V _{OL} = 0.3V _{DD} , SKLEDC[m+1, m] = 11B (m = 0 或 2 或 4 或 6)	75	150	—	mA
		5V		150	300	—	
I _{OH}	除 PD4~PD7, PE, PF 引脚外 I/O 口源电流	3V	V _{OH} = 0.9V _{DD}	-4	-8	—	mA
		5V		-8	-16	—	
	PD4~PD7, PE, PF 引脚源电流	3V	V _{OH} = 0.9V _{DD} , SLEDCn[m+1, m]=00B (n=0, 1..., m=0 或 2 或 4 或 6)	-0.7	-1.5	—	mA
		5V		-1.5	-2.9	—	
		3V	V _{OH} = 0.9V _{DD} , SLEDCn[m+1, m]=01B (n=0, 1..., m=0 或 2 或 4 或 6)	-1.3	-2.5	—	mA
		5V		-2.5	-5.1	—	
		3V	V _{OH} = 0.9V _{DD} , SLEDCn[m+1, m]=10B (n=0, 1..., m=0 或 2 或 4 或 6)	-1.8	-3.6	—	mA
		5V		-3.6	-7.3	—	
		3V	V _{OH} = 0.9V _{DD} , SLEDCn[m+1, m]=11B (n=0, 1..., m=0 或 2 或 4 或 6)	-4	-8	—	mA
		5V		-8	-16	—	
R _{PH}	I/O 口上拉电阻 (注)	3V	—	20	60	100	kΩ
		5V		10	30	50	
I _{LEAK}	输入漏电流	5V	V _{IN} = V _{DD} 或 V _{IN} = V _{SS}	—	—	±1	μA
t _{TPI}	TM 捕捉输入引脚最小脉宽	—	—	0.3	—	—	μs
t _{TCK}	TM 时钟输入引脚最小脉宽	—	—	0.3	—	—	μs
t _{INT}	外部中断输入引脚最小脉宽	—	—	10	—	—	μs
t _{RES}	外部复位最小脉宽	—	—	10	—	—	μs

注：R_{PH} 内部上拉电阻值的计算方法是：将其接地并使能输入引脚的上拉电阻选项，然后在特定电源电压下测量输入灌电流，在此基础上测量上拉电阻上的分压从而得到此上拉电阻值 R_{PH}。

存储器电气特性

Ta = -40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{RW}	读 / 写工作电压	—	—	V _{DDmin}	—	V _{DDmax}	V
程序 Flash / 数据 EEPROM 存储器							
t _{DEW}	擦除 / 写周期时间 – Flash 程序存储器	—	—	—	2	3	ms
	写周期时间 – 数据 EEPROM 存储器	—	—	—	4	6	ms
I _{DDPGM}	V _{DD} 电压下烧录 / 擦除电流	—	—	—	—	5.0	mA
E _P	电容耐久性 – Flash 程序存储器	—	—	10K	—	—	E/W
	电容耐久性 – 数据 EEPROM 存储器	—	—	100K	—	—	
t _{RETD}	ROM 数据保存时间	—	Ta = 25°C	—	40	—	Year
RAM 数据存储器							
V _{DR}	RAM 数据保存电压	—	单片机处于休眠模式	1.0	—	—	V

LVD&LVR 电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{LVR}	低电压复位电压	—	LVR 使能, 电压选择 2.1V	-5%	2.1	+5%	V
			LVR 使能, 电压选择 2.55V		2.55		
			LVR 使能, 电压选择 3.15V		3.15		
			LVR 使能, 电压选择 3.8V		3.8		
V _{LVD}	低电压检测电压	—	LVD 使能, 电压选择 2.0V	-5%	2.0	+5%	V
			LVD 使能, 电压选择 2.2V		2.2		
			LVD 使能, 电压选择 2.4V		2.4		
			LVD 使能, 电压选择 2.7V		2.7		
			LVD 使能, 电压选择 3.0V		3.0		
			LVD 使能, 电压选择 3.3V		3.3		
			LVD 使能, 电压选择 3.6V		3.6		
I _{LVRLVDBG}	工作电流	3V	LVD 使能, LVR 使能,	—	—	20	μA
		5V	VBGEN = 0	—	20	25	
		3V	LVD 使能, LVR 使能,	—	—	25	μA
		5V	VBGEN = 1	—	25	30	
t _{LVDS}	LVDO 稳定时间	—	LVR 使能时, VBGEN = 0, LVD off → on	—	—	15	μs
			LVR 除能时, VBGEN = 0, LVD off → on	—	—	150	

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
t _{LVR}	低电压复位最小低电压脉宽	—	—	120	240	480	μs
t _{LVD}	低电压中断最小低电压脉宽	—	—	60	120	240	μs
I _{LVR}	使能 LVR 的额外电流	—	LVD 除能, VBGEN = 0	—	—	24	μA
I _{LVD}	使能 LVD 的额外电流	—	LVR 除能, VBGEN = 0	—	—	24	μA

A/D 转换器电气特性

T_a = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	A/D 转换器工作电压	—	—	2.2	—	5.5	V
V _{ADI}	A/D 转换器输入电压	—	—	0	—	V _{REF}	V
V _{REF}	A/D 转换器参考电压	—	—	2	—	V _{DD}	V
DNL	A/D 非线性微分误差	—	V _{REF} = V _{DD} , t _{ADCK} = 0.5μs T _a = -40°C ~ 85°C	-3	—	+3	LSB
INL	A/D 非线性积分误差	—	V _{REF} = V _{DD} , t _{ADCK} = 0.5μs T _a = -40°C ~ 85°C	-4	—	+4	LSB
I _{ADC}	A/D 转换器使能的额外电流	3V	无负载, t _{ADCK} = 0.5μs	—	340	500	μA
		5V		—	500	700	
t _{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t _{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t _{ADC}	A/D 转换时间 (包括采样和保持时间)	—	—	—	16	—	t _{ADCK}

内部参考电压特性

T_a = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	2.2	—	5.5	V
V _{BGREF}	Bandgap 参考电压	2.2V~5.5V	T _a = -40°C~85°C	-1.0%	1.2	+1.0%	V
I _{BGREF}	工作电流	5.5V	T _a = -40°C~85°C	—	25	40	μA
I _{SD}	关机电流	—	VBGREN = 0	—	—	0.1	μA
t _{START}	启动时间	2.2V~5.5V	—	—	—	400	μs

注：1. 0.1μF 陶瓷电容器应连接在 VDD 和 GND 之间。

2. V_{BGREF} 电压作为 A/D 转换器的内部信号输入。

运算放大器电气特性

Ta = 25°C

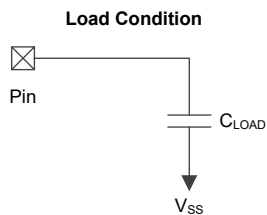
符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	2.2	—	5.5	V
I _{OPA}	运算放大器使能的 额外电流	—	无负载, SDAmBW[1:0] = 00B (m = 0, 1)	—	1.2	2.4	μA
			无负载, SDAmBW[1:0] = 01B (m = 0, 1)	—	10	16	
			无负载, SDAmBW[1:0] = 10B (m = 0, 1)	—	80	128	
			无负载, SDAmBW[1:0] = 11B (m = 0, 1)	—	200	320	
V _{OS}	输入失调电压	5V	未校准 (SDAmOF[5:0]=100000B, m= 0, 1)	-15	—	+15	mV
			校准后	-2	—	+2	mV
V _{CM}	共模电压范围	—	—	V _{SS}	—	V _{DD} -1.4	V
PSRR	电源电压抑制比	5V	—	58	70	—	dB
CMRR	共模抑制比	5V	—	58	80	—	dB
A _{OL}	开环增益	—	—	60	80	—	dB
SR	转换速率 +, 转换 速率 -	5V	R _L = 1MΩ, C _L = 60pF, SDAmBW[1:0] = 00B (m = 0, 1)	0.5	1.5	—	V/ms
			R _L = 1MΩ, C _L = 60pF, SDAmBW[1:0] = 01B (m = 0, 1)	5	15	—	
			R _L = 1MΩ, C _L = 60pF, SDAmBW[1:0] = 10B (m = 0, 1)	180	500	—	
			R _L = 1MΩ, C _L = 60pF, SDAmBW[1:0] = 11B (m = 0, 1)	600	1800	—	
GBW	运算放大器增益带宽 (每个运算放大器)	5V	R _L = 1MΩ, C _L = 100pF, SDAmBW[1:0] = 00B (m = 0, 1)	2.5	5	—	kHz
			R _L = 1MΩ, C _L = 100pF, SDAmBW[1:0] = 01B (m = 0, 1)	20	40	—	
			R _L = 1MΩ, C _L = 100pF, SDAmBW[1:0] = 10B (m = 0, 1)	400	600	—	
			R _L = 1MΩ, C _L = 100pF, SDAmBW[1:0] = 11B (m = 0, 1)	1300	2000	—	

比较器电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DDC}	比较器工作电压	—	—	2.2	—	5.5	V
I _{DD}	比较器工作电流 (每个比较器)	—	无负载, SDC1IS[1:0] = 00 或 SDC0IS[1:0] = 00	—	1.7	2.7	μA
			无负载, SDC1IS[1:0] = 01 或 SDC0IS[1:0] = 01	—	14	22	
			无负载, SDC1IS[1:0] = 10 或 SDC0IS[1:0] = 10	—	36	57	
			无负载, SDC1IS[1:0] = 11 或 SDC0IS[1:0] = 11	—	58	92	
V _{OS}	比较器输入失调电压	5V	未校准 SDCmOF[4:0]=10000B (m=0, 1)	-10	—	+10	mV
			校准后	-4	—	+4	mV
V _{CM}	比较器共模电压范围	—	—	V _{SS}	—	V _{DD} -1.4V	V
t _{RP}	响应时间	3V	10mV 过载, (注)	—	—	35	μs
		5V	SDCmIS[1:0]=00B (m=0, 1)	—	—	35	μs
		3V	10mV 过载, (注)	—	—	2.5	μs
		5V	SDCmIS[1:0]=01B (m=0, 1)	—	—	2.5	μs
		3V	10mV 过载, (注)	—	—	1	μs
		5V	SDCmIS[1:0]=10B (m=0, 1)	—	—	1	μs
		3V	10mV 过载, (注)	—	—	0.7	μs
		5V	SDCmIS[1:0]=11B (m=0, 1)	—	—	0.7	μs
V _{HYS}	迟滞	3V	SDCmHYS[1:0]=00,	0	0	5	mV
		5V	SDCmIS[1:0]=00 (m=0, 1)	0	0	5	mV
		3V	SDnCmHYS[1:0]=01,	20	40	60	mV
		5V	SDCmIS[1:0]=01 (m=0, 1)	20	40	60	mV
		3V	SDCmHYS[1:0]=10,	50	100	150	mV
		5V	SDCmIS[1:0]=10 (m=0, 1)	50	100	150	mV
		3V	SDCmHYS[1:0]=11,	80	160	240	mV
		5V	SDCmIS[1:0]=11 (m=0, 1)	80	160	240	mV

注：负载条件：C_{LOAD} = 50pF



8-Bit D/A 转换器电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	2.2	—	5.5	V
V _{DACO}	输出电压范围	—	—	V _{SS}	—	V _{REF}	V
V _{REF}	参考电压	—	—	2	—	V _{DD}	V
I _{DAC}	D/A 转换器使能的额外电流 (每个 D/A 转换器)	3V	—	—	—	200	μA
		5V		—	—	280	μA
t _{ST}	建立时间	3V	C _{LOAD} = 50pF	—	—	10	μs
		5V		—	—	10	μs
DNL	非线性微分误差	3V	V _{REF} = V _{DD}	—	—	±1	LSB
		5V		—	—	±1	LSB
INL	非线性积分误差	3V	V _{REF} = V _{DD}	—	—	±1.5	LSB
		5V		—	—	±1.5	LSB

16-Bit 语音 D/A 转换器电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	2.2	5.0	5.5	V
I _{DAC}	具有缓冲器的 D/A 转换器使能的额外电流	3V	—	—	—	3	mA
		5V		—	—	3	
I _{STB (DAC)}	待机电流	5V	DACEN = 0	—	—	1	μA
THD+N	总谐波失真 + 噪声 (注)	3V	10kΩ 负载	—	-55	—	dB
V _{OUT}	输出电压范围	5V	无负载	0.01	—	0.99	V _{DD}
t _{DACS}	D/A 转换器电路的稳定时间	5V	—	—	—	1	ms

注：正弦波输入 @ 1kHz, -6dBFS。

LCD 电气特性

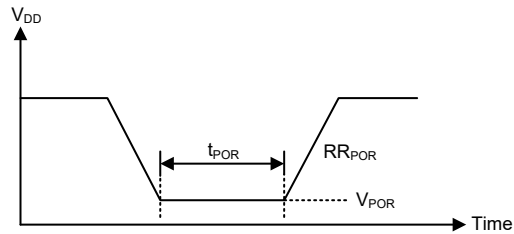
Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{BIAS}	LCD V _{DD} /2 Bias 电流	3V	ISEL[1:0] = 00B	10.5	15	19.5	μA
		5V		17.5	25	32.5	
		3V	ISEL[1:0] = 01B	21	30	39	μA
		5V		35	50	65	
		3V	ISEL[1:0] = 10B	42	60	78	μA
		5V		70	100	130	
		3V	ISEL[1:0] = 11B	82.6	118	153.4	μA
		5V		140	200	260	
V _{SCOM}	LCD COM 口 V _{DD} /2 电压	2.2V~5.5V	无负载	0.475V _{DD}	0.5V _{DD}	0.525V _{DD}	V

上电复位特性

Ta = 25°C

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

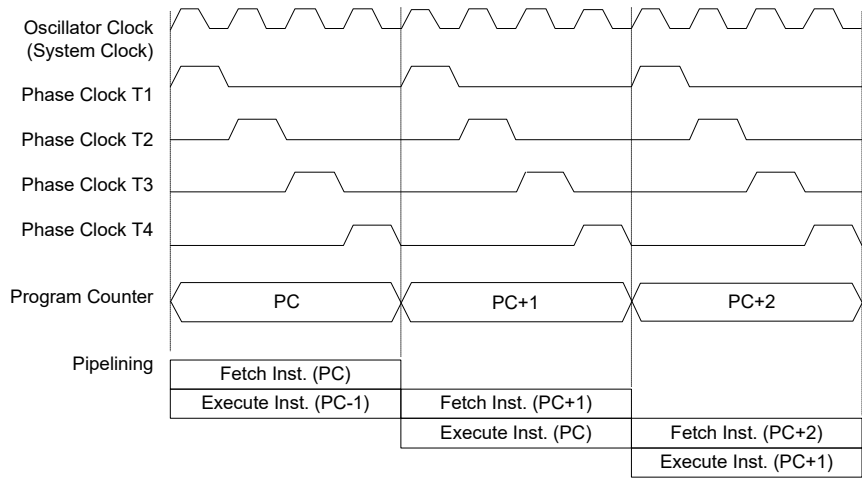


系统结构

内部系统结构是 Holtek 单片机具有良好性能的主要因素。由于采用 RISC 结构，此系列单片机具有高运算速度和高性能的特点。通过流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令外，其它指令都能在一个指令周期内完成。8 位 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器和 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。使得该系列单片机适用于低成本和批量生产的控制应用。

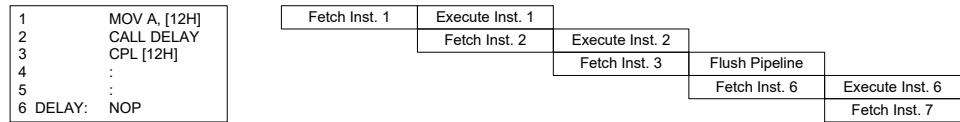
时序和流水线结构

主系统时钟由 HXT、LXT、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。



系统时序和流水线

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



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。对于存储器容量大于 8K 个字的单片机，程序存储器地址可能位于某一程序存储区，可通过程序存储区指针的 PBP0 位来选择。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地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
HT66F4530	PC10~PC8	PCL7~PCL0
HT66F4540	PC11~PC8	PCL7~PCL0
HT66F4550	PC12~PC8	PCL7~PCL0
HT66F4560	PBP0, PC12~PC8	PCL7~PCL0

程序计数器

程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前

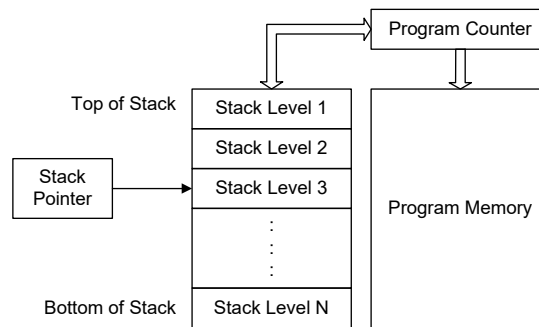
页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。PCL 的使用可能引起程序跳转，因此需要一个额外的指令周期。

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。堆栈具有多层且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。



注：HT66F4530（N=6），HT66F4540/HT66F4550（N=8），HT66F4560（N=16）

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,
 - LADD, LADDM, LADC, LADCM, LSUB, LSUBM, LSBC, LSBCM,
 - LDAA
- 逻辑运算：
 - AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA,
 - LAND, LOR, LXOR, LANDM, LORM, LXORM, LCPL, LCPLA
- 移位运算：
 - RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC,
 - LRR, LRRR, LRRCA, LRR, LRLA, LRL, LRLCA, LRLC
- 递增和递减：
 - INCA, INC, DECA, DEC,
 - LINCA, LINC, LDECA, LDEC

- 分支判断：
 JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI,
 LSZ, LSZA, LSNZ, LSIZ, LSDZ, LSIZA, LSDZA

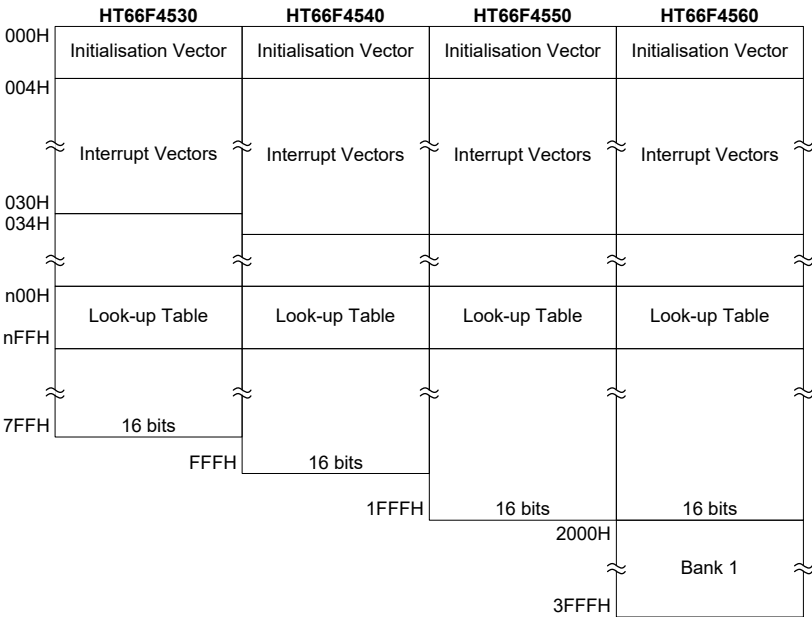
Flash 程序存储器

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。

单片机型号	容量
HT66F4530	2K×16
HT66F4540	4K×16
HT66F4550	8K×16
HT66F4560	16K×16

结构

程序存储器的容量为 2K×16 ~ 16K×16 位，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。



程序存储器结构

特殊向量

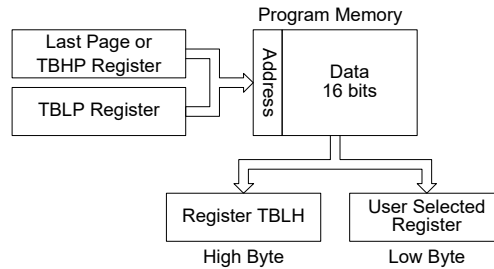
程序存储器内部某些地址保留用作诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

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 和 TBHP 中。这些寄存器定义总的表格地址。

在设定完表格指针后，当数据存储器 [m] 位于 Sector 0，表格数据可以使用如“TABRD [m]”或“TABRDL [m]”等指令分别从程序存储器查表读取。如果存储器 [m] 位于其它 Sector，表格数据可以使用如“LTABRD [m]”或“LTABRDL [m]”等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。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。HT66F4530 单片机 ORG 指令的值“0700H”指向的地址是 2K 程序存储器中最后一页的起始地址。表格指针低字节寄存器的初始值设为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址“0706H”，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”或“LTABRD [m]”指令被执行，则表格指针指向 TBLP 和 TBHP 指定的页。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”或“LTABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

TBLH 寄存器为可读 / 写寄存器，且能重新储存，若主程序和中断服务程序都使用表格读取指令，应该注意它的保护。使用表格读取指令，中断服务程序可能会改变 TBLH 的值，若随后在主程序中再次使用这个值，则会发生错误，因此建议避免同时使用表格读取指令。然而在某些情况下，如果同时使用表格读取指令是不可避免的，则在执行任何主程序的表格读取指令前，中断应该先除能，另外要注意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 page that tbhp pointed
mov a,07h           ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer
                    ; data at program
                    ; memory address "0706H" transferred to tempreg1 and TBLH
dec tblp            ; reduce value of table pointer by one
tabrd tempreg2      ; transfers value in table referenced by table pointer
                    ; data at program memory address "0705H" transferred to
                    ; tempreg2 and TBLH in this example the data "1AH" is
                    ; transferred to tempreg1 and data "0FH" to register tempreg2
:
:
org 0700h           ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

在线烧录 – ICP

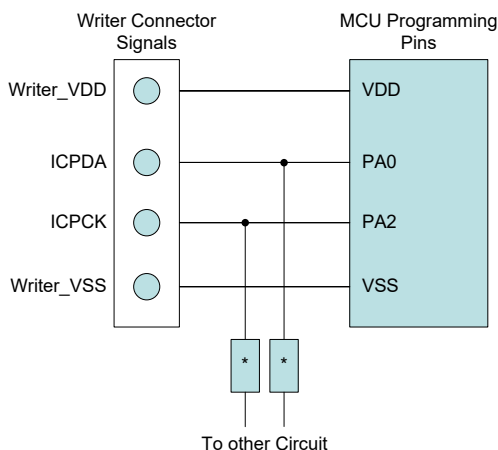
Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧录，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

Holtek Flash MCU 与烧录器引脚对应表如下：

Holtek 烧录器引脚名称	MCU 在线烧录引脚名称	功能
ICPDA	PA0	串行数据 / 地址烧录
ICPCK	PA2	时钟烧录
VDD	VDD	电源
VSS	VSS	地

芯片内部程序存储器可以通过 4 线的接口在线进行烧录。其中一条用于数据串行下载或上传、一条用于串行时钟、另外两条用于提供电源。芯片在线烧写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。

在烧录过程中，烧录器会控制 ICPDA 和 ICPCK 脚进行数据和时钟烧录，用户必须确保这两个引脚没有连接至其它输出脚。



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

片上调试 – OCDS

EV 芯片 HT66V45x0 用于单片机 HT66F45x0 的仿真。此 EV 芯片提供片上调试功能（OCDS）用于开发过程中的单片机调试。除了片上调试功能方面，EV 芯片和实际 MCU 在功能上几乎是兼容的。用户可将 OCSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV 芯片对实际 IC 的仿真。OCSDA 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，实际单片机 OCSDA 和 OCDSCK 引脚上的其它共用功能无效。由于这两个 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，请参考“Holtek e-Link for 8-bit MCU OCDS 使用手册”文件。

Holtek e-Link 引脚名称	EV 芯片引脚名称	功能
OCSDA	OCSDA	片上调试串行数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
VSS	VSS	地

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。

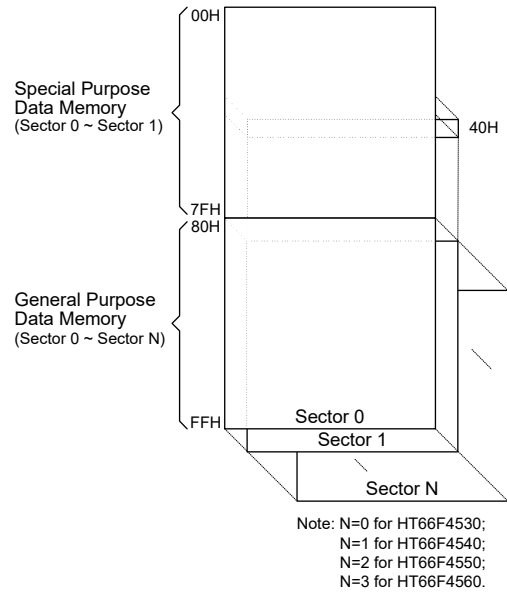
数据存储器分为两个部分，第一部分是特殊功能数据存储器。这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部分数据存储器是做一般用途使用，都可在程序控制下进行读取和写入。

切换不同的数据存储器 Sector 可通过设置正确的间接寻址指针值实现。

结构

数据存储器被分为多个 Sector，都是 8 位存储器。每个数据存储器 Sector 分为两类，特殊功能数据存储器和通用数据存储器。特殊功能数据存储器地址范围为 00H~7FH，而通用数据存储器地址范围为 80H~FFH。

单片机型号	特殊功能数据存储器	通用数据存储器	
	位于 Sector	容量	Sector: 地址
HT66F4530	0: 00H~7FH 1: 40H (EEC)	128×8	0: 80H~FFH
HT66F4540	0: 00H~7FH 1: 40H (EEC)	256×8	0: 80H~FFH 1: 80H~FFH
HT66F4550	0: 00H~7FH 1: 40H (EEC)	384×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH
HT66F4560	0: 00H~7FH 1: 40H (EEC)	512×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH 3: 80H~FFH



数据存储器结构

数据存储器寻址

此系列单片机支持扩展指令架构，它们并没有可用于数据存储器的存储器指针。对于数据存储器，使用间接寻址访问方式所需的 Sector 是通过 MP1H 或 MP2H 寄存器指定，而所选 Sector 的某一数据存储器地址是通过 MP1L 或 MP2L 寄存器指定。

直接寻址可用于所有 Sector，通过扩展指令可以寻址所有可用的数据存储器空间。当所访问的数据存储器位于除 Sector 0 外的任何数据存储器 Sector 时，扩展指令可代替间接寻址方式用来访问数据存储器。标准指令和扩展指令的主要区别在于扩展指令中的数据存储器地址“m”有 10 个有效位，高字节表示 Sector，低字节表示指定的地址。

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。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。

特殊数据存储器

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

	Sector 0	Sector 1		Sector 0	Sector 1
00H	IAR0		40H		EEC
01H	MP0		41H	EEA	
02H	IAR1		42H	EED	
03H	MP1L		43H		
04H	MP1H		44H		
05H	ACC		45H	SIMC0	
06H	PCL		46H	SIMC1	
07H	TBLP		47H	SIMD	
08H	TBLH		48H	SIMA/SIMC2	
09H	TBHP		49H	SIMTOC	
0AH	STATUS		4AH	INTEG	
0BH			4BH	INTC0	
0CH	IAR2		4CH	INTC1	
0DH	MP2L		4DH	INTC2	
0EH	MP2H		4EH	INTC3	
0FH	RSTFC		4FH	MFIO	
10H	SCC		50H	MF11	
11H	HIRCC		51H	IFS0	
12H	HXTC		52H	PD	
13H	LXTC		53H	PDC	
14H	PA		54H	PDCU	
15H	PAC		55H		
16H	PAPU		56H		
17H	PAWU		57H	PC	
18H	PSCR		58H	PCC	
19H	LVRC		59H	PCPU	
1AH	WDTC		5AH		
1BH	TB0C		5BH		
1CH	TB1C		5CH		
1DH	LVDC		5DH		
1EH			5EH		
1FH	RSTC		5FH		
20H	SADOL		60H	PAS0	
21H	SADOH		61H	PAS1	
22H	SADC0		62H	PBS0	
23H	SADC1		63H	PBS1	
24H	VBGRC		64H	PCS0	
25H	PB		65H	PCS1	
26H	PBC		66H	PDS0	
27H	PBPU		67H		
28H	PTM0C0		68H	SKLEDC	
29H	PTM0C1		69H	SDSW0	
2AH	PTM0DL		6AH	SDSW1	
2BH	PTM0DH		6BH	SDDAC0C	
2CH	PTM0AL		6CH	SDDA0L	
2DH	PTM0AH		6DH	SDDAC1C	
2EH	PTM0RPL		6EH	SDDA1L	
2FH	PTM0RPH		6FH	SDDAC2C	
30H	STM0C0		70H	SDDA2L	
31H	STM0C1		71H	SDPGAC	
32H	STM0DL		72H	SDA0C	
33H	STM0DH		73H	SDA0VOS	
34H	STM0AL		74H	SDA1C	
35H	STM0AH		75H	SDA1VOS	
36H			76H	SDC0C	
37H			77H	SDC0VOS	
38H			78H	SDC1C	
39H			79H	SDC1VOS	
3AH			7AH	SDCHYC	
3BH			7BH		
3CH			7CH		
3DH			7DH		
3EH			7EH		
3FH			7FH		

□ : Unused, read as 00H


特殊数据存储结构 – HT66F4530

	Sector 0	Sector 1		Sector 0	Sector 1
00H	IAR0		40H		EEC
01H	MP0		41H	EEA	
02H	IAR1		42H	EED	
03H	MP1L		43H		
04H	MP1H		44H		
05H	ACC		45H	SIMC0	
06H	PCL		46H	SIMC1	
07H	TBLP		47H	SIMD	
08H	TBLH		48H	SIMA/SIMC2	
09H	TBHP		49H	SIMTOC	
0AH	STATUS		4AH	INTEG	
0BH			4BH	INTC0	
0CH	IAR2		4CH	INTC1	
0DH	MP2L		4DH	INTC2	
0EH	MP2H		4EH	INTC3	
0FH	RSTFC		4FH	MFIO	
10H	SCC		50H	MF11	
11H	HIRCC		51H	IFS0	
12H	HXTC		52H	PD	
13H	LXTC		53H	PDC	
14H	PA		54H	PDPU	
15H	PAC		55H		
16H	PAPU		56H		
17H	PAWU		57H	PC	
18H	PSCR		58H	PCC	
19H	LVRC		59H	PCPU	
1AH	WDTC		5AH		
1BH	TB0C		5BH		
1CH	TB1C		5CH		
1DH	LVDC		5DH		
1EH	SCOMC		5EH		
1FH	RSTC		5FH		
20H	SADOL		60H	PAS0	
21H	SADOH		61H	PAS1	
22H	SADC0		62H	PBS0	
23H	SADC1		63H	PBS1	
24H	VBGRC		64H	PCS0	
25H	PB		65H	PCS1	
26H	PBC		66H	PDS0	
27H	PBPU		67H		
28H	PTM0C0		68H	SKLEDC	
29H	PTM0C1		69H	SDSW0	
2AH	PTM0DL		6AH	SDSW1	
2BH	PTM0DH		6BH	SDDAC0C	
2CH	PTM0AL		6CH	SDDA0L	
2DH	PTM0AH		6DH	SDDAC1C	
2EH	PTM0RPL		6EH	SDDA1L	
2FH	PTM0RPH		6FH	SDDAC2C	
30H	STM0C0		70H	SDDA2L	
31H	STM0C1		71H	SDPGAC	
32H	STM0DL		72H	SDA0C	
33H	STM0DH		73H	SDA0VOS	
34H	STM0AL		74H	SDA1C	
35H	STM0AH		75H	SDA1VOS	
36H			76H	SDC0C	
37H	PTM1C0		77H	SDC0VOS	
38H	PTM1C1		78H	SDC1C	
39H	PTM1DL		79H	SDC1VOS	
3AH	PTM1DH		7AH	SDCHYC	
3BH	PTM1AL		7BH	USR	
3CH	PTM1AH		7CH	UCR1	
3DH	PTM1RPL		7DH	UCR2	
3EH	PTM1RPH		7EH	TXR_RXR	
3FH			7FH	BRG	

□ : Unused, read as 00H

特殊数据存储器结构 – HT66F4540

Sector 0		Sector 1	Sector 0		Sector 1	
00H	IAR0		40H		EEC	
01H	MP0		41H	EEA		
02H	IAR1		42H	EED		
03H	MP1L		43H	DACC		
04H	MP1H		44H			
05H	ACC		45H	SIMC0		
06H	PCL		46H	SIMC1		
07H	TBLP		47H	SIMD		
08H	TBLH		48H	SIMA/SIMC2		
09H	TBHP		49H	SIMTOC		
0AH	STATUS		4AH	INTEG		
0BH			4BH	INTC0		
0CH	IAR2		4CH	INTC1		
0DH	MP2L		4DH	INTC2		
0EH	MP2H		4EH	INTC3		
0FH	RSTFC		4FH	MFIO		
10H	SCC		50H	MF1I		
11H	HIRCC		51H	IFS0		
12H	HXTC		52H	PD		
13H	LXTC		53H	PDC		
14H	PA		54H	PDPU		
15H	PAC		55H	DAL		
16H	PAPU		56H	DAH		
17H	PAWU		57H	PC		
18H	PSCR		58H	PCC		
19H	LVRD		59H	PCPU		
1AH	WDTC		5AH	STM1C0		
1BH	TB0C		5BH	STM1C1		
1CH	TB1C		5CH	STM1DL		
1DH	LVDC		5DH	STM1DH		
1EH	SCOMC		5EH	STM1AL		
1FH	RSTC		5FH	STM1AH		
20H	SADOL		60H	PAS0		
21H	SADOH		61H	PAS1		
22H	SADC0		62H	PBS0		
23H	SADC1		63H	PBS1		
24H	VBGRC		64H	PCS0		
25H	PB		65H	PCS1		
26H	PBC		66H	PDS0		
27H	PBPU		67H			
28H	PTM0C0		68H	SKLEDC		
29H	PTM0C1		69H	SDSW0		
2AH	PTM0DL		6AH	SDSW1		
2BH	PTM0DH		6BH	SDDAC0C		
2CH	PTM0AL		6CH	SDDA0L		
2DH	PTM0AH		6DH	SDDAC1C		
2EH	PTM0RPL		6EH	SDDA1L		
2FH	PTM0RPH		6FH	SDDAC2C		
30H	STM0C0		70H	SDDA2L		
31H	STM0C1		71H	SDPGAC		
32H	STM0DL		72H	SDA0C		
33H	STM0DH		73H	SDA0VOS		
34H	STM0AL		74H	SDA1C		
35H	STM0AH		75H	SDA1VOS		
36H			76H	SDC0C		
37H	PTM1C0		77H	SDC0VOS		
38H	PTM1C1		78H	SDC1C		
39H	PTM1DL		79H	SDC1VOS		
3AH	PTM1DH		7AH	SDCHYC		
3BH	PTM1AL		7BH	USR		
3CH	PTM1AH		7CH	UCR1		
3DH	PTM1RPL		7DH	UCR2		
3EH	PTM1RPH		7EH	TXR_RXR		
3FH		7FH	BRG			

 : Unused, read as 00H

特殊数据存储器结构 – HT66F4550

Sector 0		Sector 1		Sector 0		Sector 1		
00H	IAR0			40H		EEC		
01H	MP0			41H	EEA			
02H	IAR1			42H	EED			
03H	MP1L			43H	DACC			
04H	MP1H			44H				
05H	ACC			45H	SIMC0			
06H	PCL			46H	SIMC1			
07H	TBLP			47H	SIMD			
08H	TBLH			48H	SIMA/SIMC2			
09H	TBHP			49H	SIMTOC			
0AH	STATUS			4AH	INTEG			
0BH	PBP			4BH	INTC0			
0CH	IAR2			4CH	INTC1			
0DH	MP2L			4DH	INTC2			
0EH	MP2H			4EH	INTC3			
0FH	RSTFC			4FH	MF10			
10H	SCC			50H	MF11			
11H	HIRCC			51H	IFS0			
12H	HXTC			52H	PD			
13H	LXTC			53H	PDC			
14H	PA			54H	PDPU			
15H	PAC			55H	DAL			
16H	PAPU			56H	DAH			
17H	PAWU			57H	PC			
18H	PSCR			58H	PCC			
19H	LVRC			59H	PCPU			
1AH	WDTC			5AH	STM1C0			
1BH	TB0C			5BH	STM1C1			
1CH	TB1C			5CH	STM1DL			
1DH	LVDC			5DH	STM1DH			
1EH	SCOMC			5EH	STM1AL			
1FH	RSTC			5FH	STM1AH			
20H	SADOL			60H	PAS0		PES0	
21H	SADOH			61H	PAS1		PES1	
22H	SADC0			62H	PBS0		PFS0	
23H	SADC1			63H	PBS1		PFS1	
24H	VBGRC			64H	PCS0		IFS1	
25H	PB			65H	PCS1		IFS2	
26H	PBC			66H	PDS0		SLEDC0	
27H	PBPU			67H	PDS1		SLEDC1	
28H	PTM0C0			68H	SKLEDC		PE	
29H	PTM0C1			69H	SDSW0		PEC	
2AH	PTM0DL			6AH	SDSW1		PEPU	
2BH	PTM0DH			6BH	SDDAC0C		PF	
2CH	PTM0AL			6CH	SDDA0L		PFC	
2DH	PTM0AH			6DH	SDDAC1C		PFPU	
2EH	PTM0RPL			6EH	SDDA1L			
2FH	PTM0RPH			6FH	SDDAC2C			
30H	STM0C0			70H	SDDA2L			
31H	STM0C1			71H	SDPGAC			
32H	STM0DL			72H	SDA0C			
33H	STM0DH			73H	SDA0VOS			
34H	STM0AL			74H	SDA1C			
35H	STM0AH			75H	SDA1VOS			
36H				76H	SDC0C			
37H	PTM1C0			77H	SDC0VOS			
38H	PTM1C1			78H	SDC1C			
39H	PTM1DL			79H	SDC1VOS			
3AH	PTM1DH			7AH	SDCHYC			
3BH	PTM1AL			7BH	USR			
3CH	PTM1AH			7CH	UCR1			
3DH	PTM1RPL			7DH	UCR2			
3EH	PTM1RPH			7EH	TXR_RXR			
3FH				7FH	BRG			

□ : Unused, read as 00H

特殊数据存储结构 – HT66F4560

特殊功能寄存器

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, IAR2

间接寻址寄存器 IAR0、IAR1 和 IAR2 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、IAR1 和 IAR2 上的任何动作，将对间接寻址指针 MP0、MP1L/MP1H 或 MP2L/MP2H 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 Sector 0，而 IAR1 和 MP1L/MP1H、IAR2 和 MP2L/MP2H 可以访问任何 Sector。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, MP1L, MP1H, MP2L, MP2H

该系列单片机提供五个存储器指针，即 MP0、MP1L、MP1H、MP2L 和 MP2H。由于这些指针在数据存储区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存储器指针所指定的地址。MP0、IAR0 用于访问 Sector 0，而 MP1L/MP1H 和 IAR1、MP2L/MP2H 和 IAR2 可根据 MP1H 或 MP2H 寄存器访问所有的 Sector。使用相关指令可对所有的数据 Sector 进行直接寻址。

以下例子说明如何清除一个具有 4 RAM 地址的区块，它们已事先定义成地址 adres1 到 adres4。

间接寻址程序举例 1

```
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:
```

间接寻址程序举例 2

```
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, 01h                ; setup the memory sector
    mov mplh, a
    mov a, offset adres1      ; Accumulator loaded with first RAM address
    mov mp1l, a               ; setup memory pointer with first RAM address
loop:
    clr IAR1                  ; clear the data at address defined by MP1L
    inc mp1l                  ; increment memory pointer MP1L
    sdz block                  ; check if last memory location has been cleared
    jmp loop
continue:
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

使用扩展指令直接寻址程序举例

```
data .section 'data'

temp db ?
code .section at 0 'code'
org 00h
start:
    lmov a, [m]                ; move [m] data to acc
    lsub a, [m+1]              ; compare [m] and [m+1] data
    snz c                      ; [m]>[m+1]?
    jmp continue               ; no
    lmov a, [m]                ; yes, exchange [m] and [m+1] data
    mov temp, a
    lmov a, [m+1]
    lmov [m], a
    mov a, temp
    lmov [m+1], a
continue:
```

注：“m”是位于任何数据存储器 Sector 的某一地址。例如，m=1F0H 表示 Sector 1 中的地址 0F0H

程序存储区指针—PBP

HT66F4560 单片机程序存储器被分为几个 Bank，可以通过设置程序存储区指针 PBP 来访问不同的程序存储区。PBP 寄存器应在单片机使用“JMP”或“CALL”指令执行“分支”操作前正确地配置。在指令执行后会跳转到由程序存储区指针所选 Bank 的一个非连续的程序存储器地址。

● PBP 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	PBP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 未定义，读为“0”

Bit 0 **PBP0**: 选择程序存储区
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。

表格寄存器 – TBLP, TBHP, TBLH

这三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 和 TBHP 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

这 8 位的状态寄存器由 SC 标志位、CZ 标志位、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 PDF 和 TO 标志外，状态寄存器中的位像其它大部分寄存器一样可以通过指令改变。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、C、SC 和 CZ 标志位通常反映最近运算的状态。

- 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，则 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。
- **CZ**: 不同指令不同标志位的操作结果。详细资料请参考寄存器定义部分。
- **SC**: 当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果。

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。应注意, STATUS 寄存器的 0~3 位是可以读取和写入的。

● STATUS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SC	CZ	TO	PDF	OV	Z	AC	C
R/W	R	R	R	R	R/W	R/W	R/W	R/W
POR	x	x	0	0	x	x	x	x

“x”: 未知

- Bit 7 **SC**: 当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果
- Bit 6 **CZ**: 不同指令不同标志位的操作结果。
对于 SUB/SUBM/LSUB/LSUBM 指令, CZ 等于 Z 标志位。
对于 SBC/SBCM/LSBC/LSBCM 指令, CZ 等于上一个 CZ 标志位与当前零标志位执行“AND”所得结果。
对于其它指令, CZ 标志位无影响。
- 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”进位标志位也受循环移位指令的影响。

EEPROM 数据存储

此系列单片机的一个特性是内建 EEPROM 数据存储。 “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。这种存储器扩展了 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。

单片机型号	容量	地址
HT66F4530	32×8	00H~1FH
HT66F4540	64×8	00H~3FH
HT66F4550	64×8	00H~3FH
HT66F4560	128×8	00H~7FH

EEPROM 数据存储结构

此系列单片机 EEPROM 数据存储容量为 32×8~128×8 位。由于映射方式与程序存储器和数据存储器不同，因此不能像其它类型的存储器一样寻址。使用 Sector 0 中的一个地址和数据寄存器以及 Sector 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储器的操作，地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 位于 Sector 0 中，它们能像其它特殊功能寄存器一样直接被访问。EEC 位于 Sector 1 中，可以通过 MP1L/MP1H 和 IAR1 或 MP2L/MP2H 和 IAR2 进行间接读取或写入。由于 EEC 控制寄存器位于 Sector 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，MP1L 或 MP2L 必须先设为“40H”，MP1H 或 MP2H 被设为“01H”。

寄存器名称	位							
	7	6	5	4	3	2	1	0
EEA(HT66F4530)	—	—	—	EEA4	EEA3	EEA2	EEA1	EEA0
EEA(HT66F4540/ HT66F4550)	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EEA(HT66F4560)	—	EEA6	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM 寄存器列表

• EEA 寄存器 – HT66F4530

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	EEA4	EEA3	EEA2	EEA1	EEA0
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 **EEA4~EEA0**: 数据 EEPROM 地址
数据 EEPROM 地址 bit 4 ~ bit 0

● EEA 寄存器 – HT66F4540/HT66F4550

Bit	7	6	5	4	3	2	1	0
Name	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
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 **EEA5~EEA0**: 数据 EEPROM 地址
数据 EEPROM 地址 bit 5 ~ bit 0

● EEA 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	—	EEA6	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
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 6~0 **EEA6~EEA0**: 数据 EEPROM 地址
数据 EEPROM 地址 bit 6 ~ 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 **D7~D0**: 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

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: 写周期有效

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: 读周期有效
 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 RD 位已置为高而 RDEN 位还未被设置则不能开始读操作。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。应用程序可轮询 RD 位以确定数据可以有效地被读取。

写数据到 EEPROM

写数据至 EEPROM, EEPROM 中写入数据的地址要先放入 EEA 寄存器中。写入的数据要存入 EED 寄存器中。写数据至 EEPROM, EEC 寄存器中的写使能位 WREN 先置为高以使能写功能。之后将 WR 位置为高, 初始化一个写周期。这两个指令必须连续执行。在执行任何写操作之前, 总中断位 EMI 要先清零, 写周期开始后, 再将 EMI 置为高。需要注意的是若 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 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。因此, 应用程序可轮询 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。上电后存储器指针高字节寄存器 MP1H 或 MP2H 将重置为“0”, 这意味着数据存储器 Sector 0 被选中。由于 EEPROM 控制寄存器位于 Sector 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 写中断。当 EEPROM 写周期结束, DEF 请求标志位将被置位。若 EEPROM 中断使能且堆栈未满的情况下将跳转到相应的 EEPROM 中断向量中执行。当中断被响应, 中断请求标志位 DEF 会被复位且 EMI 位会被清零以除能其它中断。

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增强保护功能。存储器指针高字节寄存器 MP1H 或 MP2H 也可以正常清零以阻止进入 EEPROM 控制寄存器所在的 Sector 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。应注意，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。

程序举例

● 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 MP1L
MOV MP1L, A              ; MP1L points to EEC register
MOV A, 01H               ; setup memory pointer MP1H
MOV MP1H, 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 read/write
CLR MP1H
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 MP1L
MOV MP1L, A              ; MP1L points to EEC register
MOV A, 01H               ; setup memory pointer MP1H
MOV MP1H, A
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 read/write
CLR MP1H
```

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。振荡器选择和操作是通过相关控制寄存器完成的。

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。外部振荡器需要一些外围器件，而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。所有振荡器选择通过寄存器选择。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。

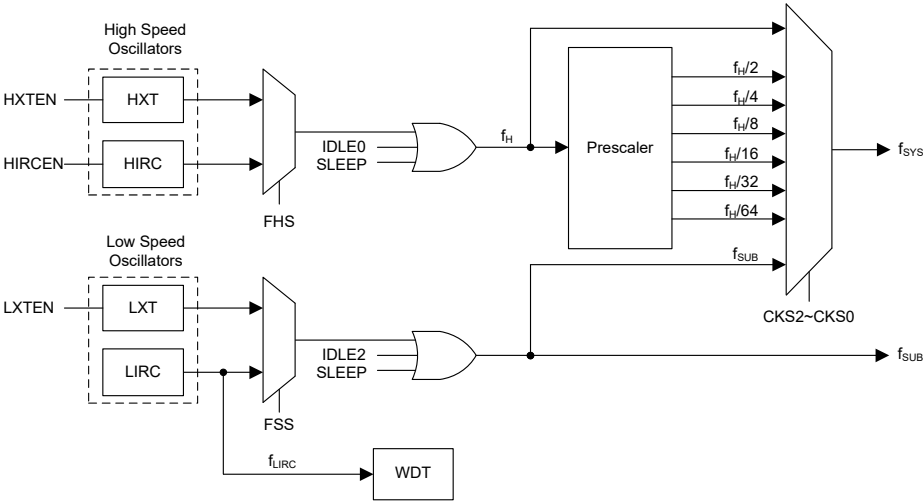
类型	名称	频率	引脚
外部高速晶振	HXT	400kHz~12MHz	OSC1/OSC2
内部高速 RC	HIRC	2/4/8MHz	—
外部低速晶振	LXT	32.768kHz	XT1/XT2
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

该系列单片机有四个系统振荡器，包括两个高速振荡器和两个低速振荡器。高速振荡器有外部晶体 / 陶瓷振荡器 HXT 和内部 2/4/8MHz 高速振荡器 HIRC，低速振荡器有内部 32kHz 低速振荡器 LIRC 和外部 32.768kHz 晶振 LXT。使用高速或低速振荡器作为系统时钟的选择是通过设置 SCC 寄存器中的 CKS2~CKS0 位决定的，系统时钟可动态选择。

低速振荡器的实际时钟源由 SCC 寄存器的 FSS 位选择，高速振荡器的实际时钟源由 SCC 寄存器的 FHS 位选择。低速或高速系统时钟频率由 SCC 寄存器的 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。不可选择一个无振荡器选择的高或低转速振荡器。

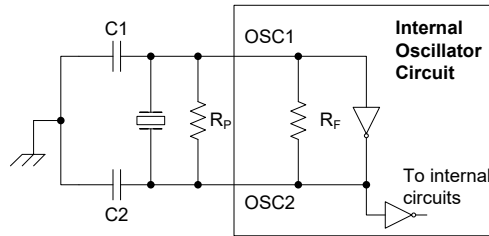


系统时钟配置

外部晶体 / 陶瓷振荡器 – HXT

外部高频晶体 / 陶瓷振荡器是一个高频振荡器。对于晶体振荡器，只要简单地 将晶体连接至 OSC1 和 OSC2，则会产生振荡所需的相移及反馈，而不需其它 外部电容。为保证某些低频率的晶体振荡和陶瓷谐振器的振荡频率更精准，建 议连接两个小容量电容 C1 和 C2 到 VSS，具体数值与客户选择的晶体 / 陶瓷晶 振有关。

为了确保振荡器的稳定性及减少噪声和串扰的影响，晶体振荡器及其相关的电 阻和电容以及它们之间的连线都应尽可能的接近单片机。



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

晶体 / 陶瓷振荡器 – HXT

晶体振荡器 C1 和 C2 值		
晶体频率	C1	C2
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
注：C1 和 C2 数值仅作参考		

晶体振荡器电容推荐值

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡 器具有三种固定的频率：2MHz，4MHz，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。如果选择了该内部时钟，无需额外的引脚；PC0 和 PC1 可以 作为通用 I/O 口使用。

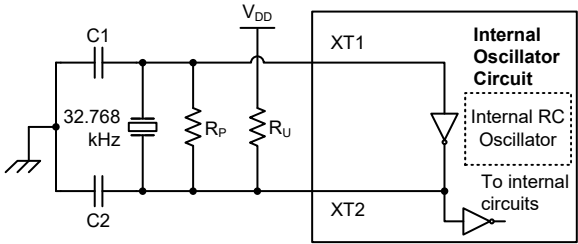
外部 32.768kHz 晶体振荡器 – LXT

外部 32.768kHz 晶体振荡器是一个低频振荡器，由 FSS 控制位选择。时钟频率 固定为 32.768kHz，此时 XT1 和 XT2 间引脚必须连接 32.768kHz 的晶体振荡器。 需要外部电阻和电容连接到 32.768kHz 晶振以帮助起振。对于那些要求精确频 率的场合中，可能需要这些元件来对由制程产生的误差提供频率补偿。LXTEN 位置高使能 LXT 振荡器后，LXT 振荡器启动需要一定的延时。

然而，对于一些晶体，为了保证系统频率的启动与精度要求，需要外接两个小 容量电容 C1 和 C2，具体数值与客户选择的晶体规格有关。外部并联的反馈电 阻 R_P 和接一个上拉电阻 R_U 是必需的。

引脚共用的软件控制位决定 XT1/XT2 脚是用于 LXT 还是作为普通 I/O 口或其它共用功能使用。

- 若 LXT 振荡器未被用于任何时钟源，XT1/XT2 脚能被用作一般 I/O 口或其它共用功能使用。
- 若 LXT 振荡器被用于一些时钟源，32.768kHz 晶体应被连接至 XT1/XT2 脚。为了确保振荡器的稳定性及减少噪声和串扰的影响，晶体振荡器及其相关的电阻和电容以及它们之间的连线都应尽可能的接近单片机。



Note: 1. R_P , R_U , $C1$ and $C2$ are required.
2. Although not shown XT1/XT2 pins have a parasitic capacitance of around 7pF.

外部 LXT 振荡器

LXT 振荡器 C1 和 C2 值		
晶体频率	C1	C2
32.768kHz	10pF	10pF
注：1、C1 和 C2 数值仅作参考用 2、 R_P 的建议值为 5M Ω ~10M Ω 3、 R_U 的建议值为 10M Ω		

32.768kHz 振荡器电容推荐值

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个低频振荡器，由 FSS 控制位选择。它是一个完全集成 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。

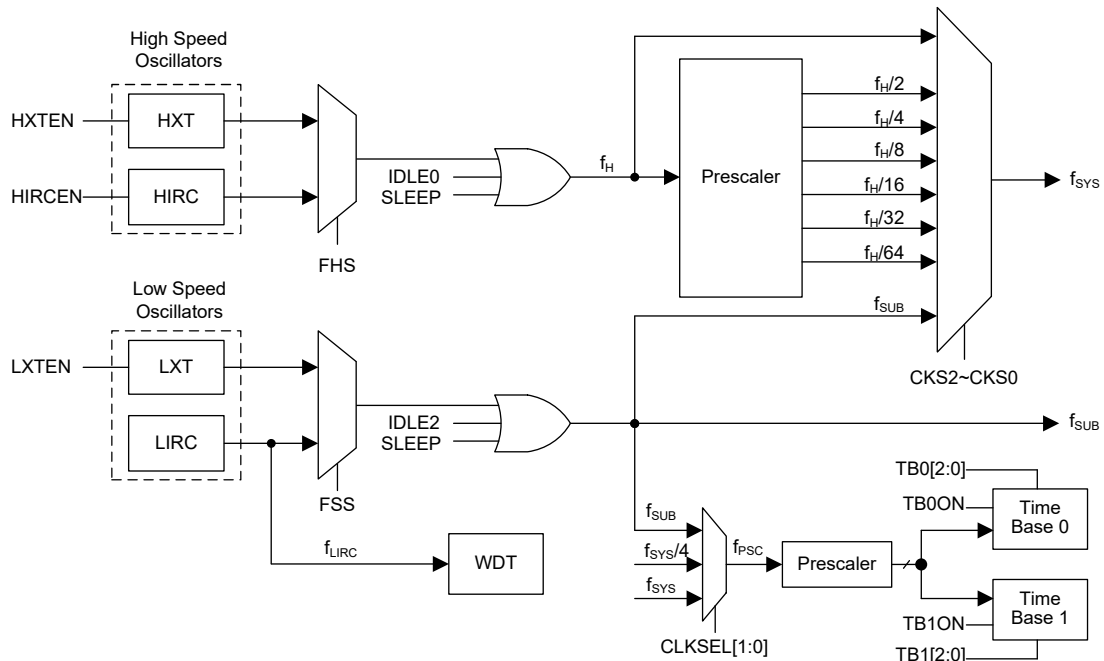
工作模式和系统时钟

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。此系列单片机提供高、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

此系列单片机为 CPU 和外围功能操作提供了多种不同的时钟源。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_{SUB} ，通过 SCC 寄存器中的 CKS2~CKS0 位进行选择。高频时钟来自 HXT 或 HIRC 振荡器，可通过 SCC 寄存器中的 FHS 位选择。低频系统时钟源来自 f_{SUB} ，若 f_{SUB} 被选择，低频时钟来自 LXT 或 LIRC 振荡器，可通过 SCC 寄存器中的 FSS 位选择。其它系统时钟还有高速系统振荡器的分频 $f_H/2 \sim f_H/64$ 。



单片机时钟配置

注：当系统时钟源 f_{SYS} 由 f_H 到 f_{SUB} 转换时，可以通过设置相应的高速振荡器使能控制位，选择停止以节省耗电，或者继续振荡，为外围电路提供 $f_H \sim f_H/64$ 频率的时钟源。

系统工作模式

单片机有 6 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：快速模式和低速模式。剩余的 4 种工作模式：休眠模式、空闲模式 0、空闲模式 1 和空闲模式 2 用于单片机 CPU 关闭时以节省耗电。

工作模式	CPU	寄存器设置			f _{sys}	f _H	f _{SUB}	f _{LIRC}
		FHIDEN	FSIDEN	CKS2~CKS0				
快速模式	On	x	x	000~110	f _H ~f _H /64	On	On	On
低速模式	On	x	x	111	f _{SUB}	On/Off ⁽¹⁾	On	On
空闲模式 0	Off	0	1	000~110	Off	Off	On	On
				111	On			
空闲模式 1	Off	1	1	xxx	On	On	On	On
空闲模式 2	Off	1	0	000~110	On	On	Off	On
				111	Off			
休眠模式	Off	0	0	xxx	Off	Off	Off	On/Off ⁽²⁾

“x”：无关

注：1. 在低速模式中，f_H 开启或关闭由相应的振荡器使能位控制。
2. 在休眠模式中，f_{LIRC} 时钟开启或关闭由 WDT 功能使能或禁能决定。

快速模式

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 HXT 或 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 SCC 寄存器中的 CKS2~CKS0 位选择。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 f_{SUB}，而 f_{SUB} 可来自于 LIRC 或 LXT 振荡器，通过 SCC 寄存器的 FSS 位选择。

休眠模式

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行，f_{SUB} 停止为外围功能提供时钟。如果看门狗定时器功能使能，f_{LIRC} 继续运行。

空闲模式 0

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为低、FSIDEN 位为高时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但低速振荡器会开启以驱动一些外围功能。

空闲模式 1

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但高速和低速振荡器都会开启以确保一些外围功能继续工作。

空闲模式 2

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为高、FSIDEN 位为低时，系统进入空闲模式 2。在空闲模式 2 中，CPU 停止，但高速振荡器会开启以确保一些外围功能继续工作。

控制寄存器

寄存器 SCC、HIRCC、HXTC 和 LXTC 用于控制系统时钟和相应的振荡器配置。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SCC	CKS2	CKS1	CKS0	—	FHS	FSS	FHIDEN	FSIDEN
HIRCC	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
HXTC	—	—	—	—	—	HXTM	HXTF	HXTEN
LXTC	—	—	—	—	—	—	LXTF	LXTEN

系统工作模式控制寄存器列表

● SCC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	FHS	FSS	FHIDEN	FSIDEN
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 7~5 **CKS2~CKS0**: 系统时钟选择位

000: f_H
001: $f_H/2$
010: $f_H/4$
011: $f_H/8$
100: $f_H/16$
101: $f_H/32$
110: $f_H/64$
111: f_{SUB}

这三位用于选择系统时钟源。除了 f_H 或 f_{SUB} 提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟。

Bit 4 未定义，读为“0”

Bit 3 **FHS**: 高频时钟选择位

0: HIRC
1: HXT

Bit 2 **FSS**: 低频时钟选择位

0: LIRC
1: LXT

Bit 1 **FHIDEN**: CPU 关闭时高频振荡器控制位

0: 除能
1: 使能

此位用来控制在 CPU 执行 HALT 指令关闭后高速振荡器是被激活还是停止。

Bit 0 **FSIDEN**: CPU 关闭时低频振荡器控制位

0: 除能
1: 使能

此位用来控制在 CPU 执行 HALT 指令关闭后低速振荡器是被激活还是停止。LIRC 振荡器是由该位与 WDT 功能使能控制位共同控制的。如果该位被清零，但 WDT 功能使能， f_{LIRC} 振荡器也将使能。

• HIRCC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
R/W	—	—	—	—	R/W	R/W	R	R/W
POR	—	—	—	—	0	0	0	1

Bit 7~4 未定义，读为“0”

Bit 3~2 **HIRC1~HIRC0**: HIRC 频率选择位

00: 2MHz

01: 4MHz

10: 8MHz

11: 2MHz

当 HIRC 振荡器使能或通过应用程序改变 HIRC 频率选择位时，在 HIRCF 标志位置高后时钟频率会自动改变。

Bit 1 **HIRCF**: HIRC 振荡器稳定标志位

0: HIRC 未稳定

1: HIRC 稳定

此位用于表明 HIRC 振荡器是否稳定。HIRCEN 位置高使能 HIRC 振荡器或通过应用程序改变 HIRC 振荡器频率选择位，HIRCF 位会先被清零，在 HIRC 稳定后会被置高。

Bit 0 **HIRCEN**: HIRC 振荡器使能控制位

0: 除能

1: 使能

• HXTC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	HXTM	HXTF	HXTEN
R/W	—	—	—	—	—	R/W	R	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”

Bit 2 **HXTM**: HXT 模式选择位

0: HXT 频率 ≤10MHz

1: HXT 频率 >10MHz

此位用于选择 HXT 振荡器的工作模式。注意，此位必须在 HXT 使能前正确地配置，在 HXTEN 位置高使能 HXT 振荡器后再改变此位的值将无效。

Bit 1 **HXTF**: HXT 振荡器稳定标志位

0: HXT 未稳定

1: HXT 稳定

此位用于表明 HXT 振荡器是否稳定。HXTEN 位置高使能 HXT 振荡器后，HXTF 位会先被清零，在 HXT 稳定后会被置高。

Bit 0 **HXTEN**: HXT 振荡器使能控制位

0: 除能

1: 使能

• LXTC 寄存器

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

Bit 7~2 未定义，读为“0”

Bit 1 **LXTF**: LXT 振荡器稳定标志位

0: LXT 未稳定

1: LXT 稳定

此位用于表明 LXT 振荡器是否稳定。LXTEN 位置高使能 LXT 振荡器后，LXTF 位会先被清零，在 LXT 稳定后会被置高。

Bit 0 **LXTEN**: LXT 振荡器使能控制位

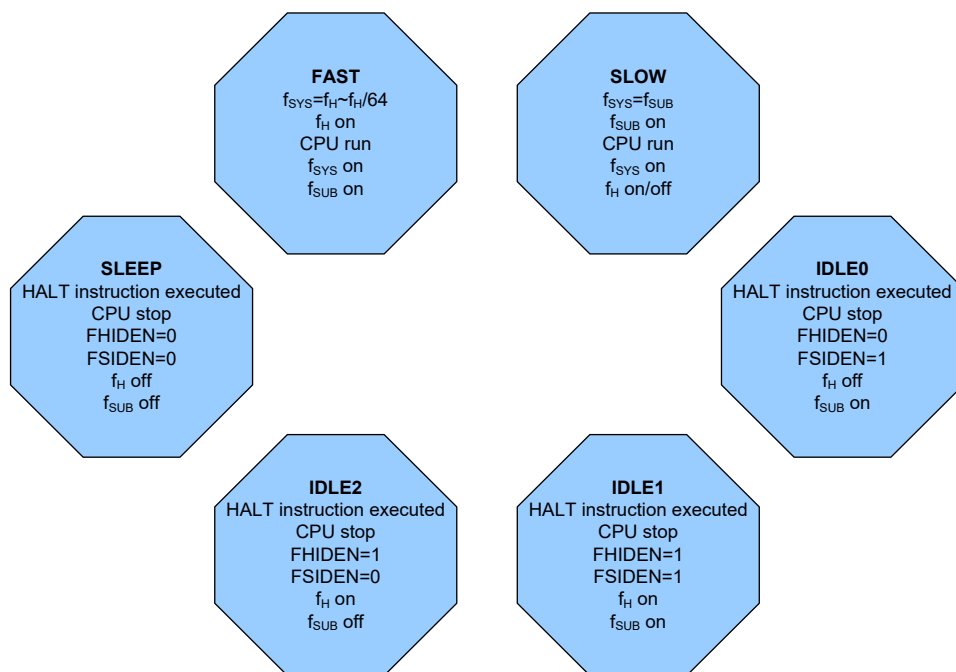
0: 除能

1: 使能

工作模式切换

此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。

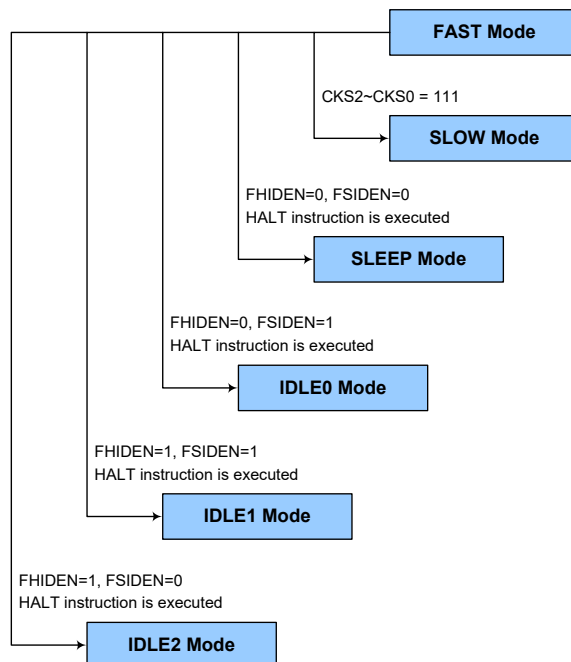
简单来说，快速模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现，而快速模式 / 低速模式与休眠模式 / 空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入空闲模式或休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位决定的。



快速模式切换到低速模式

系统运行在快速模式时使用高速系统振荡器，因此较为耗电。可通过设置 SCC 寄存器中的 CKS2~CKS0 位为“111”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。

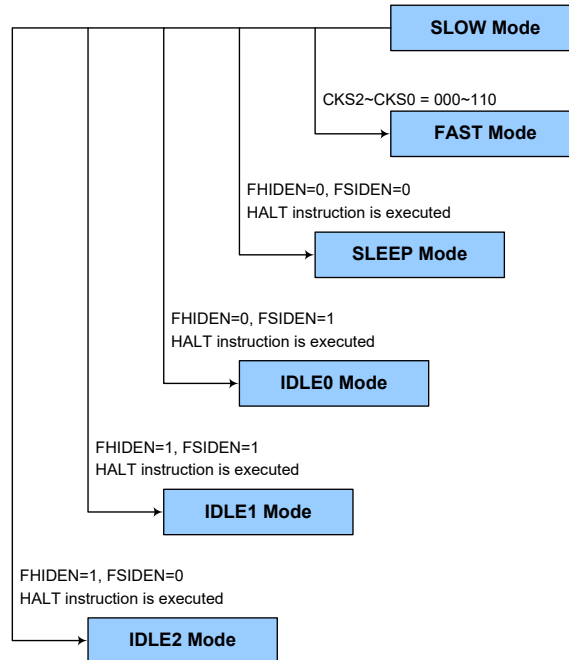
低速模式的时钟源来自 LXT 或 LIRC 振荡器，由 SCC 寄存器中的 FSS 位确定，因此要求这些振荡器在所有模式切换动作发生前稳定下来。



低速模式切换到快速模式

在低速模式时系统时钟来自 f_{SUB} 。切换回快速模式时，需设置 CKS2~CKS0 位为“000”~“110”使系统时钟从 f_{SUB} 切换到 $f_H \sim f_H/64$ 。

然而，如果在低速模式下 f_H 因未使用而关闭，那么从低速模式切换到快速模式时，它需要一定的时间来重新起振和稳定，可通过检测 HXTC 寄存器中的 HXTF 位或 HIRCC 寄存器中的 HIRCF 位进行判断，所需的高速系统振荡器稳定时间在系统上电时间电气特性中有说明。



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 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“0”。在这种模式下，除了 WDT 以外的所有时钟和功能都将关闭。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。
- 数据存储器和寄存器将保持当前值。
- 输入 / 输出将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止运行。

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 SCC 寄存器中的 FHIDEN 位为“0”且 FSIDEN 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 时钟停止运行，应用程序停止在“HALT”指令处，但 f_{SUB} 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。
- 输入 / 输出将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止运行。

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 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“1”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 和 f_{SUB} 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。
- 输入 / 输出将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止运行。

进入空闲模式 2

进入空闲模式 2 的方法仅有一种——应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 位为“1”且 FSIDEN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 时钟开启， f_{SUB} 时钟关闭，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。
- 输入 / 输出将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止运行。

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 和空闲模式 2 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。还应注意的，如果选择 LXT 或 LIRC 振荡器，会导致耗电增加。

在空闲模式 1 和空闲模式 2 中高速振荡器开启。若外围功能时钟源来自高速振荡器，额外的待机电流也可能会有几百微安。

唤醒

单片机进入休眠模式或空闲模式后，系统时钟将停止以降低功耗。然而单片机再次唤醒，原来的系统时钟重新起振、稳定且恢复正常工作需要一定的时间。

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：

- 外部 $\overline{\text{RES}}$ 引脚复位
- PA 口下降沿
- 系统中断
- WDT 溢出

单片机由外部 $\overline{\text{RES}}$ 引脚唤醒，系统会完全复位；若由 WDT 溢出唤醒，则会发生看门狗定时器复位。这两种唤醒方式都会使系统复位，可通过状态寄存器中的 TO 和 PDF 位来判断唤醒源。系统上电或执行清除看门狗的指令，PDF 将被清零；执行 HALT 指令，PDF 将被置位。看门狗定时器溢出会将 TO 置位并唤醒系统，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 定时器时钟源由内部低速振荡器 LIRC 提供。内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期会随 V_{DD} 、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 $2^8 \sim 2^{18}$ 以提供更大的溢出周期，分频比由 WDT 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDT 寄存器用于选择溢出周期、控制 WDT 功能的使能 / 除能。

• WDT 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 软件控制位
10101: 除能
01010: 使能
其它值: MCU 复位
若因外部环境噪声或软件设置使单片机复位，复位动作发生在一段延迟时间 t_{SRESET} 后，复位后 RSTFC 寄存器中的 WRF 标志位会被置位。

Bit 2~0 **WS2~WS0**: WDT 溢出周期选择位
000: $2^8/f_{LIRC}$
001: $2^{10}/f_{LIRC}$
010: $2^{12}/f_{LIRC}$
011: $2^{14}/f_{LIRC}$
100: $2^{15}/f_{LIRC}$
101: $2^{16}/f_{LIRC}$
110: $2^{17}/f_{LIRC}$
111: $2^{18}/f_{LIRC}$
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。

• RSTFC 寄存器

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

“x”：未知

Bit 7~4 未定义，读为 “0”
Bit 3 **RSTF**: 复位控制寄存器软件复位标志位
详见其他章节。
Bit 2 **LVRF**: LVR 复位标志位
详见其他章节。

- Bit 1 **LRF**: LVR 控制寄存器软件复位标志位
详见其他章节。
- Bit 0 **WRF**: WDT 控制寄存器软件复位标志位
0: 未发生
1: 发生
当 WDT 控制寄存器软件复位发生时, 此位被置为“1”, 且只能通过应用程序清零。

看门狗定时器操作

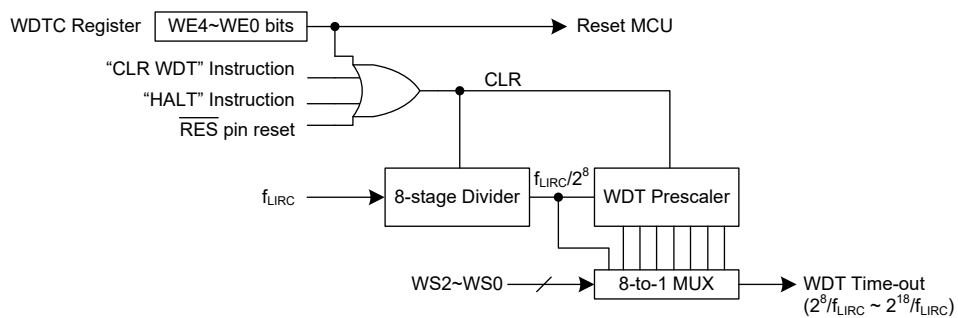
当 WDT 溢出时, 它产生一个单片机复位的动作。这也就意味着正常工作期间, 用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位, 可使用清除看门狗指令实现。无论什么原因, 程序失常跳转到一个未知的地址或进入一个死循环, 这些清除指令都不能被正确执行, 此种情况下, 看门狗将溢出以使单片机复位。看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位可提供使能 / 除能控制以及控制看门狗定时器复位单片机。如果 WE4~WE0 为 10101B, 则 WDT 除能; 如果 WE4~WE0 为 01010B, 则 WDT 使能; 而当设置为“01010B”或“10101B”以外的值时, 单片机将在一段延迟时间 t_{SRESET} 后复位。上电后这些位初始化为“01010B”。

WE4~WE0 位	WDT 功能
10101B	除能
01010B	使能
其它值	单片机复位

看门狗定时器使能 / 除能控制

程序正常运行时, WDT 溢出将导致单片机复位, 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 应置位, 仅 PC 和堆栈指针复位。有四种方法可以用来清除 WDT 的内容。第一种是 WDT 复位, 即将 WE4~WE0 位设置成除了“01010B”和“10101B”外的任意值; 第二种是通过软件清除指令, 而第三种是通过“HALT”指令。最后一种是外部硬件复位, 即外部 RES 引脚低电平。

该系列单片机只使用软件指令清看门狗。只要执行“CLR WDT”便清除 WDT。当设置分频比为 2^{18} 时, 溢出周期最大。例如, 时钟源为 32kHz LIRC 振荡器, 分频比为 2^{18} 时最大溢出周期约 8s, 分频比为 2^8 时最小溢出周期约 8ms。



看门狗定时器

复位和初始化

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

除了上电复位外，即使单片机处于正常工作状态，有些情况的发生也会迫使单片机复位。譬如当单片机上电后已经开始执行程序， $\overline{\text{RES}}$ 引脚被强制拉为低电平。这种复位为正常操作复位，单片机中只有一些寄存器受影响，而大部分寄存器不会改变，在复位引脚恢复至高电平后，单片机可以正常运行。

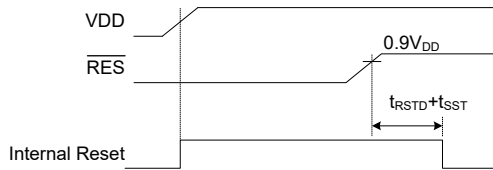
另一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设置值时，系统会产生 LVR 复位。这种复位与 $\overline{\text{RES}}$ 引脚拉低复位方式相似。

复位功能

单片机的几种内部和外部复位方式将在此处做具体介绍。

上电复位

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



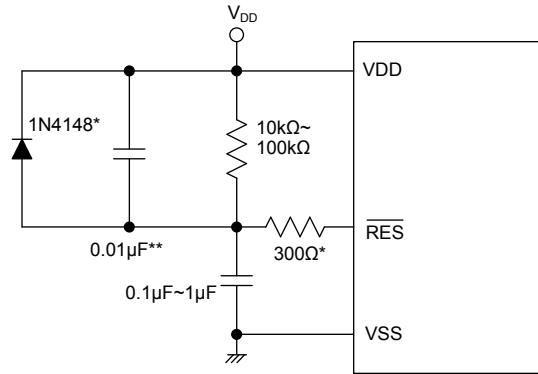
上电复位时序图

$\overline{\text{RES}}$ 引脚复位

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

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

当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。



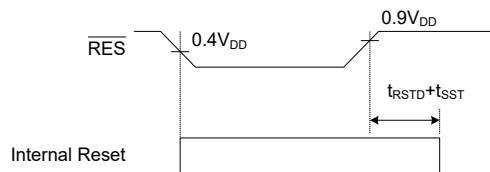
注：“*”表示建议加上此元件以加强静电保护。

“**”表示建议在电源有较强干扰场合加上此元件。

外部 RES 电路

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

RES 引脚通过外部硬件强迫拉至低电平时，此种复位形式即会发生。这种复位方式和其它的复位方式一样，程序计数器会被清除为零且程序从头开始执行。



RES 复位时序图

● RSTC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	RSTC7	RSTC6	RSTC5	RSTC4	RSTC3	RSTC2	RSTC1	RSTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7~0 **RSTC7~RSTC0**: PC2/RES 选择位

01010101: 配置为 PC2 或其它引脚共用功能

10101010: 配置为 RES 引脚

其它值: MCU 复位

如果由于不利的环境因素使这些位发生改变，单片机将复位。复位动作发生在一段延迟时间 t_{SRESET} 后，且 RSTFC 寄存器的 RSTF 位将置为“1”。

除了 WDT 溢出是硬件热复位外所有的复位将该寄存器复位为 POR 值。注意，当通过设置 RSTC 寄存器为“10101010B”选择 RES 引脚功能后，比其它引脚共用功能，RES 引脚功能具有较高优先级。

● RSTFC 寄存器

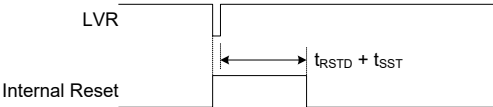
Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	RSTF	LVRF	LRF	WRF
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	x	0	0

“x”：未知

- Bit 7~4 未定义，读为“0”
- Bit 3 **RSTF**：复位控制寄存器软件复位标志位
0：未发生
1：发生
当 RSTC 控制寄存器软件复位发生时，此位被置为“1”，且只能通过应用程序清零。
- Bit 2 **LVRF**：LVR 复位标志位
详见其他章节。
- Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
详见其他章节。
- Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
详见其他章节。

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和电压低于设定值单片机复位。低电压复位功能总是使能于特定的电压值， V_{LVR} 。例如在更换电池的情况下，单片机供应的电压可能会在 $0.9V \sim V_{LVR}$ 之间，这时 LVR 将会自动复位单片机且 RSTFC 寄存器中的 LVRF 标志位置位。所谓有效的 LVR 信号，即在 $0.9V \sim V_{LVR}$ 的低电压状态的时间，必须超过 LVD&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} 参数值可通过 LVRC 寄存器中的 LVS7~LVS0 位设置。若由于受到干扰 LVS7~LVS0 变为其它值时，需经过一段延迟时间 t_{SRESET} 后才响应复位。此时 RSTFC 寄存器的 LRF 位被置位。上电后寄存器的值为 01010101B。注意的是，LVR 会于休眠或空闲时自动除能关闭。



低电压复位时序图

● LVRC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	LVS7	LVS6	LVS5	LVS4	LVS3	LVS2	LVS1	LVS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	1	0	1

- Bit 7~0 **LVS7~LVS0**：LVR 电压选择
01010101：2.1V
00110011：2.55V
10011001：3.15V
10101010：3.8V
其它值：单片机复位一寄存器复位为 POR 值
若低电压情况发生且满足以上定义的低电压复位值，则单片机复位。此时复位后的寄存器内容保持不变。
除了以上定义的低电压复位值外，其它值也能导致单片机复位。需要经过一段延迟时间 t_{SRESET} 才响应复位。但此时寄存器内容将复位为 POR 值。

• RSTFC 寄存器

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

“x”：未知

Bit 7~4 未定义，读为“0”

Bit 3 **RSTF**：复位控制寄存器软件复位标志位
详见其他章节。

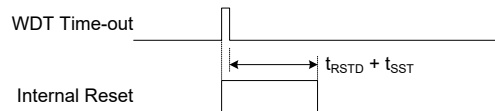
Bit 2 **LVRF**：LVR 功能复位标志位
0：未发生
1：发生
当特定的低电压复位条件发生时，此位被置为“1”，且只能通过应用程序清零。

Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
0：未发生
1：发生
如果 LVRC 寄存器包含任何非定义的 LVR 电压值，此位被置为“1”，这类似于软件复位功能，且只能通过应用程序清零。

Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
详见其他章节。

正常运行时看门狗溢出复位

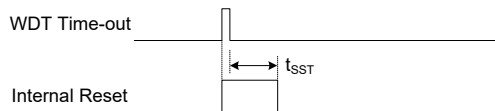
除了看门狗溢出标志位 TO 将被设为“1”之外，在快速模式或低速模式时看门狗溢出复位和 RES 引脚复位相同。



正常运行时看门狗溢出复位时序图

休眠或空闲时看门狗溢出复位

休眠或空闲时看门狗溢出复位和其它种类的复位有些不同，除了程序计数器与堆栈指针将被清零及 TO 位被设为“1”外，绝大部份的条件保持不变。图中 t_{SST} 的详细说明请参考系统上电时间电气特性。



休眠或空闲时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位，即 PDF 和 TO 位存放在状态寄存器中，由休眠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	快速模式或低速模式时的 $\overline{\text{RES}}$ 复位或 LVR 复位
1	u	快速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

“u”：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，时基	都清除，且 WDT 重新计数
定时器模块	所有定时器模块停止
输入 / 输出	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。注意的是，此系列单片机支持多种封装类型，该表将反映较大封装类型的情况。

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	$\overline{\text{RES}}$ 复位 (正常运行)	$\overline{\text{RES}}$ 复位 (空闲 / 休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲 / 休眠)
IAR0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	●	—	—	—	---- -xxx	---- -uuu	---- -uuu	---- -uuu	---- -uuu
	—	●	—	—	---- xxxx	---- uuuu	---- uuuu	---- uuuu	---- uuuu
	—	—	●	—	---x xxxx	---u uuuu	---u uuuu	---u uuuu	---u uuuu
	—	—	—	●	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	●	●	●	●	xx00 xxxx	uuuu uuuu	uu01 uuuu	xx1u uuuu	uu11 uuuu
PBP	—	—	—	●	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
IAR2	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	RES 复位 (正常运行)	RES 复位 (空闲/休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
MP2L	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
RSTFC	●	●	●	●	---- 0x00	---- uuuu	---- uuuu	---- uuuu	---- uuuu
SCC	●	●	●	●	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
HIRCC	●	●	●	●	---- 0001	---- 0001	---- 0001	---- 0001	---- uuuu
HXTC	●	●	●	●	---- -000	---- -000	---- -000	---- -000	---- -uuu
LXTC	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PA	●	—	—	—	11-- -111	11-- -111	11-- -111	11-- -111	uu-- -uuu
	—	●	●	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	●	—	—	—	11-- -111	11-- -111	11-- -111	11-- -111	uu-- -uuu
	—	●	●	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	●	—	—	—	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	●	—	—	—	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PSCR	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
LVRC	●	●	●	●	0101 0101	0101 0101	0101 0101	0101 0101	uuuu uuuu
WDTC	●	●	●	●	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
TB0C	●	●	●	●	0--- -000	0--- -000	0--- -000	0--- -000	u--- -uuu
TB1C	●	●	●	●	0--- -000	0--- -000	0--- -000	0--- -000	u--- -uuu
LVDC	●	●	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
SCOMC	—	●	●	●	-000 ----	-000 ----	-000 ----	-000 ----	-uuu ----
RSTC	●	●	●	●	01010101	01010101	01010101	01010101	uuuuuuuu
SADOL	●	●	●	●	xxxx ----	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
	●	●	●	●					uuuu uuuu (ADRFS=1)
SADOH	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
	●	●	●	●					---- uuuu (ADRFS=1)
SADC0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
VBGRC	●	●	●	●	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
PB	●	—	—	—	---1 1-11	---1 1-11	---1 1-11	---1 1-11	---u u-uu
	—	●	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PBC	●	—	—	—	---1 1-11	---1 1-11	---1 1-11	---1 1-11	---u u-uu
	—	●	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	RES 复位 (正常运行)	RES 复位 (空闲/休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
PBPUR	●	—	—	—	---0 0-00	---0 0-00	---0 0-00	---0 0-00	---u u-uu
	—	●	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
PTM0C0	●	●	●	●	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTM0C1	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DH	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM0AL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0AH	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM0RPL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPH	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM0C0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0C1	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DH	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM0AL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0AH	●	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1C0	—	●	●	●	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTM1C1	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	—	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1AL	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	—	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1RPL	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	—	●	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
EEC	●	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
EEA	●	—	—	—	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
	—	●	●	—	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
	—	—	—	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
EED	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
DACC	—	—	●	●	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
SIMC0	●	●	●	●	111- 0000	111- 0000	111- 0000	111- 0000	uuu- uuuu
SIMC1	●	●	●	●	1000 0001	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA	●	●	●	●	0000 000-	0000 000-	0000 000-	0000 000-	uuuu uu-
SIMC2	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMTOC	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTEG	●	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
INTC0	●	●	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	RES 复位 (正常运行)	RES 复位 (空闲/休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
INTC2	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC3	●	—	—	—	---0 ---0	---0 ---0	---0 ---0	---0 ---0	---u ---u
	—	●	●	●	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
MFI0	●	—	—	—	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	●	●	—	—	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	●	●	●	—	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	●	●	●	—	---- 1111	---- 1111	---- 1111	---- 1111	---- uuuu
	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	●	●	●	—	---- 0000	--- 0000	--- 0000	--- 0000	---- uuuu
	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
DAL	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
DAH	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	●	—	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCC	●	—	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	●	-111 1111	-111 1111	-111 1111	-111 1111	-uuu uuuu
PCPU	●	—	—	—	--11 -111	--11 -111	--11 -111	--11 -111	--uu -uuu
	—	●	●	●	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
STM1C0	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1C1	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1DL	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1DH	—	—	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
STM1AL	—	—	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM1AH	—	—	●	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PAS0	●	—	—	—	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	●	—	—	—	0000 ----	0000 ----	0000 ----	0000 ----	uuuu ----
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS0	●	—	—	—	00-- 0000	00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS1	●	—	—	—	---- --00	---- --00	----- 00	---- --00	---- --uu
	—	●	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
PCS0	●	—	—	—	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	RES 复位 (正常运行)	RES 复位 (空闲/休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
PCS1	●	—	—	—	---- 0000	---- 0000	----0000	---- 0000	---- uuuu
	—	●	●	●	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
PDS0	●	—	—	—	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS1	—	—	—	●	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
SKLEDC	●	●	●	●	---- --00	---- --00	-----00	---- --00	---- --uu
SDSW0	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDSW1	●	●	●	●	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
SDDAC0C	●	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA0L	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDDAC1C	●	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA1L	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDDAC2C	●	●	●	●	0--- ----	0--- ----	0--- ----	0--- ----	u--- ----
SDDA2L	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SDPGAC	●	●	●	●	00-- -000	00-- -000	00-- -000	00-- -000	uu-- -uuu
SDA0C	●	●	●	●	-00- --00	-00- --00	-00- --00	-00- --00	-uu- --uu
SDA0VOS	●	●	●	●	0010 0000	0010 0000	0010 0000	0010 0000	uuuu uuuu
SDA1C	●	●	●	●	-00- --00	-00- --00	-00- --00	-00- --00	-uu---uu
SDA1VOS	●	●	●	●	0010 0000	0010 0000	0010 0000	0010 0000	uuuu uuuu
SDC0C	●	●	●	●	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
SDC0VOS	●	●	●	●	-001 0000	-001 0000	-001 0000	-001 0000	-uuu uuuu
SDC1C	●	●	●	●	000- 0000	000- 0000	000- 0000	000- 0000	uuu- uuuu
SDC1VOS	●	●	●	●	-001 0000	-001 0000	-001 0000	-001 0000	-uuu uuuu
SDCHYC	●	●	●	●	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu
USR	—	●	●	●	0000 1011	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	—	●	●	●	0000 00x0	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	—	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR_RXR	—	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	—	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PES0	—	—	—	●	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
PES1	—	—	—	●	00-- 00--	00-- 00--	00-- 00--	00-- 00--	uu-- uu--
PFS0	—	—	—	●	00-- 0000	00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
PFS1	—	—	—	●	0000 00--	0000 00--	0000 00--	0000 00--	uuuu uu--
IFS1	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS2	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC0	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC1	—	—	—	●	---- --00	---- --00	---- --00	---- --00	---- --uu
PE	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

寄存器	HT66F4530	HT66F4540	HT66F4550	HT66F4560	上电复位	RES 复位 (正常运行)	RES 复位 (空闲/休眠)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
PEPU	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFC	—	—	—	●	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PFPU	—	—	—	●	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu

注：“u”：不改变。

“x”：未知。

“-”：未定义。

输入 / 输出端口

Holtek 单片机的输入 / 输出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，这些特性也使得此类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~PF 双向输入 / 输出口。这些寄存器在数据存储器有特定的地址。所有 I/O 口用于输入输出操作。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]”，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。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	—	—	—	PA2	PA1	PA0
PAC	PAC7	PAC6	—	—	—	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	—	—	—	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	—	—	—	PAWU2	PAWU1	PAWU0
PB	—	—	—	PB4	PB3	—	PB1	PB0
PBC	—	—	—	PBC4	PBC3	—	PBC1	PBC0
PBPU	—	—	—	PBPU4	PBPU3	—	PBPU1	PBPU0
PC	—	—	PC5	PC4	—	PC2	PC1	PC0
PCC	—	—	PCC5	PCC4	—	PCC2	PCC1	PCC0
PCPU	—	—	PCPU5	PCPU4	—	PCPU2	PCPU1	PCPU0
PD	—	—	—	—	PD3	PD2	PD1	PD0
PDC	—	—	—	—	PDC3	PDC2	PDC1	PDC0
PDPU	—	—	—	—	PDPU3	PDPU2	PDPU1	PDPU0

I/O 逻辑功能寄存器列表 – HT66F4530

寄存器名称	位							
	7	6	5	4	3	2	1	0
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	—	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	—	—	—	—	PD3	PD2	PD1	PD0
PDC	—	—	—	—	PDC3	PDC2	PDC1	PDC0
PDPU	—	—	—	—	PDPU3	PDPU2	PDPU1	PDPU0

I/O 逻辑功能寄存器列表 – HT66F4540/HT66F4550

寄存器名称	位							
	7	6	5	4	3	2	1	0
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	—	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PF	PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0
PFC	PFC7	PFC6	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PFPU	PFPU7	PFPU6	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0

I/O 逻辑功能寄存器列表 – HT66F4560

上拉电阻

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

应注意只有在引脚共用功能用脚配置为输入或 NMOS 输出时，可通过相关上拉控制寄存器控制上拉电阻，否则，上拉电阻无法被使能。

● P_xPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	P _x PU7	P _x PU6	P _x PU5	P _x PU4	P _x PU3	P _x PU2	P _x PU1	P _x PU0
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

P_xPU_n: 输入 / 输出端口 x 引脚上拉功能控制

0: 除能

1: 使能

P_xPU_n 位用于控制引脚的上拉功能。此处“x”可以是 A~F。然而，每个实际使用端口的位可以是不同的。

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。

应注意只有在引脚共用功能选择为通用输入 / 输出口且单片机进入暂停模式时，此功能可由唤醒控制寄存器控制。

● 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 **PAWU7~PAWU0:** PA7~PA0 唤醒功能控制位

0: 除能

1: 使能

输入 / 输出端口控制寄存器

每一个输入 / 输出端口都具有各自的控制寄存器，即 PAC~PFC，用来控制输入 / 输出状态。从而每个 I/O 引脚都可以通过软件控制，动态的设置为 CMOS 输出或输入。所有的 I/O 端口的引脚都各自对应于 I/O 端口控制的某一位。若 I/O 引脚要实现输入功能，则对应的控制寄存器的位需要设置为“1”。这时程序指令可以直接读取输入脚的逻辑状态。若控制寄存器相应的位被设定为“0”，则此引脚被设置为 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。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。

● PxC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PxC7	PxC6	PxC5	PxC4	PxC3	PxC2	PxC1	PxC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

PxCn: 输入 / 输出端口 x 引脚类型选择位

0: 输出

1: 输入

PxCn 位用于引脚类型选择。此处“x”可以是 A~F。然而，每个实际使用端口的位可以是不一样的。

I/O 口灌电流控制

此系列单片机 PB0 和 PB1 引脚支持不同的灌电流驱动能力。通过相关选择寄存器 SKLEDC，这些引脚支持四种灌电流驱动能力。用户参考输入 / 输出电气特性章节给不同应用设计灌电流驱动能力。

● SKLEDC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	SKLEDC1	SKLEDC0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **SKLEDC1~SKLEDC0:** PB1~PB0 灌电流选择位

00: 灌电流 = Level 0 (最小)

01: 灌电流 = Level 1

10: 灌电流 = Level 2

11: 灌电流 = Level 3 (最大)

注：对于不同的应用，用户需参考输入 / 输出电气特性章节获得精确值。

I/O 口源电流控制 – HT66F4560

HT66F4560 单片机 PD4~PD7、PE、PF 引脚支持不同的源电流驱动能力。通过相关选择寄存器 SLEDC0~SLEDC1，这些引脚支持四种源电流驱动能力。用户参考输入 / 输出电气特性章节给不同应用设计源电流驱动能力。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SLEDC0	SLEDC07	SLEDC06	SLEDC05	SLEDC04	SLEDC03	SLEDC02	SLEDC01	SLEDC00
SLEDC1	—	—	—	—	—	—	SLEDC11	SLEDC10

I/O 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	SLEDC07	SLEDC06	SLEDC05	SLEDC04	SLEDC03	SLEDC02	SLEDC01	SLEDC00
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 **SLEDC07~SLEDC06**: PF3~PF0 源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 5~4 **SLEDC07~SLEDC06**: PE7~PE4 源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 3~2 **SLEDC07~SLEDC06**: PE3~PE0 源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 1~0 **SLEDC07~SLEDC06**: PD7~PD4 源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

• SLEDC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	SLEDC11	SLEDC10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **SLEDC11~SLEDC10**: PF7~PF4 源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

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一系列寄存器进行设定。

引脚共用功能选择寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能共用和引脚功能选择，使得小封装单片机具有更多不同的功能。单片机包含端口“x”输出功能选择寄存器“n”，记为 PxSn，和输入功能选择寄存器“i”，记为 IFSi，用于选择多功能共用引脚上的所需功能。

应注意的一点是，确保所需的引脚共用功能被正确地选择和取消。对于多数引脚共用功能，要选择所需的引脚共用功能，首先应通过相应的引脚共用控制寄存器正确地选择该功能，然后再配置相应的功能设置最后再使能此功能。但是，在设置相关引脚控制字段时，一些数字输入引脚如 INTn、xTCKn、xTPnI 等，与对应的通用 I/O 口共用同一个引脚共用设置选项。要选择这些引脚功能，除了上述的必要的引脚共用控制和外围功能设置外，还必须将其对应的端口控制寄存器位设置为输入。要正确地取消引脚共用功能，首先应除能该功能，然后再修改相应的引脚共用控制寄存器以选择其它的引脚共用功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	—	—	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	—	—	—	—
PBS0	PBS07	PBS06	—	—	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	—	—	—	—	PBS11	PBS10
PCS0	—	—	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	—	—	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
IFS0	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00

引脚共用功能选择寄存器列表 – HT66F4530

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	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
IFS0	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00

引脚共用功能选择寄存器列表 – HT66F4540/HT66F4550

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	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	—	—	PDS15	PDS14	—	—	PDS11	PDS10
PES0	—	—	PES05	PES04	—	—	PES01	PES00
PES1	PES17	PES16	—	—	PES13	PES12	—	—
PFS0	PFS07	PFS06	—	—	PFS03	PFS02	PFS01	PFS00
PFS1	PFS17	PFS16	PFS15	PFS14	PFS13	PFS12	—	—
IFS0	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
IFS1	IFS17	IFS16	IFS15	IFS14	IFS13	IFS12	IFS11	IFS10
IFS2	IFS27	IFS26	IFS25	IFS24	IFS23	IFS22	IFS21	IFS20

引脚共用功能选择寄存器列表 – HT66F4560

● PAS0 寄存器 – HT66F4530

Bit	7	6	5	4	3	2	1	0
Name	—	—	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
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 5~4 **PAS05~PAS04**: PA2 引脚共用功能选择

00: PA2

01: PA2

10: PA2

11: PA2

Bit 3~2 **PAS03~PAS02**: PA1 引脚共用功能选择

00: PA1/STP0I

01: SDI/SDA

10: VREF

11: PA1/STP0I

Bit 1~0 **PAS01~PAS00**: PA0 引脚共用功能选择

00: PA0

01: PA0

10: PA0

11: PA0

● PAS0 寄存器 – HT66F4540

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: SCOM0
10: SCS
11: PA3

Bit 5~4 **PAS05~PAS04:** PA2 引脚共用功能选择

00: PA2
01: TX
10: PA2
11: PA2

Bit 3~2 **PAS03~PAS02:** PA1 引脚共用功能选择

00: PA1/STP0I
01: SDI/SDA
10: PA1/STP0I
11: PA1/STP0I

Bit 1~0 **PAS01~PAS00:** PA0 引脚共用功能选择

00: PA0
01: RX
10: PA0
11: PA0

● PAS0 寄存器 – HT66F4550/HT66F4560

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: SCOM0
10: SCS
11: DACO

Bit 5~4 **PAS05~PAS04:** PA2 引脚共用功能选择

00: PA2
01: TX
10: PA2
11: PA2

Bit 3~2 **PAS03~PAS02:** PA1 引脚共用功能选择

00: PA1/STP0I
01: SDI/SDA
10: PA1/STP0I
11: PA1/STP0I

Bit 1~0 **PAS01~PAS00**: PA0 引脚共用功能选择
 00: PA0
 01: RX
 10: PA0
 11: PA0

● **PAS1 寄存器 – HT66F4530**

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

Bit 7~6 **PAS17~PAS16**: PA7 引脚共用功能选择
 00: PA7
 01: STP0
 10: AN2
 11: A1NI

Bit 5~4 **PAS15~PAS14**: PA6 引脚共用功能选择
 00: PA6
 01: AN0
 10: A1O
 11: PA6

Bit 3~0 未定义，读为“0”

● **PAS1 寄存器 – HT66F4540/HT66F4550/HT66F4560**

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: STP0
 10: AN2
 11: A1NI

Bit 5~4 **PAS15~PAS14**: PA6 引脚共用功能选择
 00: PA6/PTCK1
 01: AN0
 10: A1O
 11: PA6/PTCK1

Bit 3~2 **PAS13~PAS12**: PA5 引脚共用功能选择
 00: PA5/STCK0
 01: RX
 10: AN5
 11: VREF

Bit 1~0 **PAS11~PAS10**: PA4 引脚共用功能选择
 00: PA4/INT0
 01: SCK/SCL
 10: VREF
 11: SCOM3

● **PBS0 寄存器 – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	—	—	PBS03	PBS02	PBS01	PBS00
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7~6 **PBS07~PBS06:** PB3 引脚共用功能选择

00: PB3
01: AN1
10: A1PI
11: PB3

Bit 5~4 未定义，读为“0”

Bit 3~2 **PBS03~PBS02:** PB1 引脚共用功能选择

00: PB1/INT1
01: AN4
10: PB1/INT1
11: XT2

Bit 1~0 **PBS01~PBS00:** PB0 引脚共用功能选择

00: PB0/INT0
01: PTP0
10: AN3
11: XT1

● **PBS0 寄存器 – HT66F4540/HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
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 **PBS07~PBS06:** PB3 引脚共用功能选择

00: PB3
01: AN1
10: A1PI
11: PB3

Bit 5~4 **PBS05~PBS04:** PB2 引脚共用功能选择

00: PB2/PTCK0
01: AN6
10: TX
11: PB2/PTCK0

Bit 3~2 **PBS03~PBS02:** PB1 引脚共用功能选择

00: PB1/INT1
01: AN4
10: PB1/INT1
11: XT2

Bit 1~0 **PBS01~PBS00:** PB0 引脚共用功能选择

00: PB0
01: PTP0
10: AN3
11: XT1

● **PBS1 寄存器 – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBS11	PBS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **PBS11~PBS10**: PB4 引脚共用功能选择

00: PB4

01: CLO (系统时钟输出: 当系统时钟除能, CLO 强制为高)

10: A00

11: PB4

● **PBS1 寄存器 – HT66F4540/HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
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 5~4 **PBS15~PBS14**: PB6 引脚共用功能选择

00: PB6

01: SCOM1

10: SDI/SDA

11: RX

Bit 3~2 **PBS13~PBS12**: PB5 引脚共用功能选择

00: PB5

01: SCOM2

10: TX

11: AN7

Bit 1~0 **PBS11~PBS10**: PB4 引脚共用功能选择

00: PB4

01: CLO (系统时钟输出: 当系统时钟除能, CLO 强制为高)

10: A00

11: PB4

● **PCS0 寄存器 – HT66F4530**

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 5~4 **PCS05~PCS04**: PC2 引脚共用功能选择

00: PC2/RES/PTCK0

01: SCK/SCL

10: PC2/RES/PTCK0

11: PC2/RES/PTCK0

Bit 3~2 **PCS03~PCS0:** PC1 引脚共用功能选择
 00: PC1
 01: SDO
 10: OSC2
 11: PC1

Bit 1~0 **PCS01~PCS00:** PC0 引脚共用功能选择
 00: $\overline{\text{PC0}}$
 01: $\overline{\text{SCS}}$
 10: OSC1
 11: PC0

● **PCS0 寄存器 – HT66F4540**

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS07~PCS06:** PC3 引脚共用功能选择
 00: PC3
 01: PC3
 10: SDO
 11: C1XO

Bit 5~4 **PCS05~PCS04:** PC2 引脚共用功能选择
 00: PC2/ $\overline{\text{RES}}$
 01: PTP1
 10: PC2/ $\overline{\text{RES}}$
 11: PC2/ $\overline{\text{RES}}$

Bit 3~2 **PCS03~PCS02:** PC1 引脚共用功能选择
 00: PC1
 01: SDO
 10: OSC2
 11: PC1

Bit 1~0 **PCS01~PCS00:** PC0 引脚共用功能选择
 00: $\overline{\text{PC0}}$
 01: $\overline{\text{SCS}}$
 10: OSC1
 11: PC0

● **PCS0 寄存器 – HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS07~PCS06:** PC3 引脚共用功能选择
 00: PC3
 01: PC3
 10: SDO
 11: C1XO

- Bit 5~4 **PCS05~PCS04:** PC2 引脚共用功能选择
 00: PC2/RES/STP1I
 01: PTP1
 10: PC2/RES/STP1I
 11: PC2/RES/STP1I
- Bit 3~2 **PCS03~PCS02:** PC1 引脚共用功能选择
 00: PC1/STCK1
 01: SDO
 10: OSC2
 11: PC1/STCK1
- Bit 1~0 **PCS01~PCS00:** PC0 引脚共用功能选择
 00: PC0
 01: SCS
 10: OSC1
 11: STP1

● **PCS1 寄存器 – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PCS13	PCS12	PCS11	PCS10
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 **PCS13~PCS12:** PC5 引脚共用功能选择
 00: PC5/INT0
 01: C1PI
 10: PC5/INT0
 11: PC5/INT0
- Bit 1~0 **PCS11~PCS10:** PC4 引脚共用功能选择
 00: PC4
 01: C1NI
 10: PC4
 11: PC4

● **PCS1 寄存器 – HT66F4540/HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
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 5~4 **PCS15~PCS14:** PC6 引脚共用功能选择
 00: PC6
 01: C0NI
 10: PC6
 11: PC6
- Bit 3~2 **PCS13~PCS12:** PC5 引脚共用功能选择
 00: PC5
 01: C1PI
 10: PC5
 11: PC5

Bit 1~0 **PCS11~PCS10:** PC4 引脚共用功能选择
 00: PC4
 01: C1NI
 10: PC4
 11: PC4

● **PDS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PDS07	PDS05	PDS04	PDS03	PDS02	PDS02	PDS01	PDS00
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 **PDS07~PDS06:** PD3 引脚共用功能选择
 00: PD3
 01: A0PI
 10: PD3
 11: PD3

Bit 5~4 **PDS05~PDS04:** PD2 引脚共用功能选择
 00: PD2
 01: A0NI
 10: PD2
 11: PD2

Bit 3~2 **PDS03~PDS02:** PD1 引脚共用功能选择
 00: PD1
 01: C0PI
 10: PD1
 11: PD1

Bit 1~0 **PDS01~PDS00:** PD0 引脚共用功能选择
 00: PD0
 01: C0XO
 10: SCK/SCL
 11: PD0

● **PDS1 寄存器 – HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PDS14	PDS13	—	—	PDS11	PDS10
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 5~4 **PDS14~PDS13:** PD6 引脚共用功能选择
 00: PD6
 01: SCK/SCL
 10: PD6
 11: PD6

Bit 3~2 未定义，读为“0”

Bit 1~0 **PDS11~PDS10:** PD4 引脚共用功能选择
 00: PD4
 01: RX
 10: PD4
 11: PD4

● PES0 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	—	—	PES05	PES04	—	—	PES01	PES00
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 5~4 **PES05~PES04**: PE2 引脚共用功能选择

00: PE2
01: PTP0
10: PE2
11: PE2

Bit 3~2 未定义，读为“0”

Bit 1~0 **PES01~PES00**: PE0 引脚共用功能选择

00: PE0
01: STP0
10: PE0
11: PE0

● PES1 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	PES17	PES16	—	—	PES13	PES12	—	—
R/W	R/W	R/W	—	—	R/W	R/W	—	—
POR	0	0	—	—	0	0	—	—

Bit 7~6 **PES17~PES16**: PE7 引脚共用功能选择

00: PE7
01: STP1
10: PE7
11: PE7

Bit 5~4 未定义，读为“0”

Bit 3~2 **PES13~PES12**: PE5 引脚共用功能选择

00: PE5
01: PTP1
10: PE5
11: PE5

Bit 1~0 未定义，读为“0”

● PFS0 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	PFS07	PFS06	—	—	PFS03	PFS02	PFS01	PFS00
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7~6 **PFS07~PFS06**: PF3 引脚共用功能选择

00: PF3
01: C0XO
10: PF3
11: PF3

- Bit 5~4 未定义，读为“0”
- Bit 3~2 **PFS03~PFS02**: PF1 引脚共用功能选择
00: PF1
01: SDI/SDA
10: PF1
11: PF1
- Bit 1~0 **PFS01~PFS00**: PF0 引脚共用功能选择
00: PF0
01: C1XO
10: PF0
11: PF0

● **PFS1 寄存器 – HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	PFS17	PFS16	PFS15	PFS14	PFS13	PFS12	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—
POR	0	0	0	0	0	0	—	—

- Bit 7~6 **PFS17~PFS16**: PF7 引脚共用功能选择
00: PF7
01: TX
10: PF7
11: PF7
- Bit 5~4 **PFS15~PFS14**: PF6 引脚共用功能选择
00: PF6
01: SCS
10: PF6
11: PF6
- Bit 3~2 **PFS13~PFS12**: PF5 引脚共用功能选择
00: PF5
01: SDO
10: PF5
11: PF5
- Bit 1~0 未定义，读为“0”

● **IFS0 寄存器 – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
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 **IFS07~IFS06**: SDI/SDA 输入源引脚选择位
00: PA1
01: PA1
10: PA1
11: PA1
- Bit 5~4 **IFS05~IFS04**: SCS 输入源引脚选择位
00: PC0
01: PC0
10: PC0
11: PC0

Bit 3~2 **IFS03~IFS02**: SCK/SCL 输入源引脚选择位

00: PD0

01: PD0

10: PC2

11: PC2

Bit 1~0 **IFS01~IFS00**: INT0 输入源引脚选择位

00: PB0

01: PB0

10: PC5

11: PC5

注：在 SPI 主机模式时，SIMEN=1 和 CSEN=1，不论 IFS0[5:4] 位段为何值，PC0 能作为 SCS 引脚，当 SIMEN=1，不论 IFS0[3:2] 位段为何值，PC2 和 PD0 引脚能作为 SCK 引脚。

● IFS0 寄存器 – HT66F4540/HT66F4550

Bit	7	6	5	4	3	2	1	0
Name	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
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 **IFS07~IFS06**: SDI/SDA 输入源引脚选择位

00: PA1

01: PA1

10: PB6

11: PB6

Bit 5~4 **IFS05~IFS04**: $\overline{\text{SCS}}$ 输入源引脚选择位

00: PA3

01: PA3

10: PC0

11: PC0

Bit 3~2 **IFS03~IFS02**: SCK/SCL 输入源引脚选择位

00: PD0

01: PD0

10: PA4

11: PA4

Bit 1~0 **IFS01~IFS00**: RX 输入源引脚选择位

00: PB6

01: PA0

10: PA5

11: PB6

注：在 SPI 主机模式时，SIMEN=1 和 CSEN=1，不论 IFS0[5:4] 位段为何值，PA3 和 PC0 能作为 SCS 引脚，当 SIMEN=1，不论 IFS0[3:2] 位段为何值，PA4 和 PD0 引脚能作为 SCK 引脚。

● IFS0 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	IFS07	IFS06	IFS05	IFS04	IFS03	IFS02	IFS01	IFS00
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 **IFS07~IFS06**: SDI/SDA 输入源引脚选择位

00: PA1
01: PA1
10: PB6
11: PF1

Bit 5~4 **IFS05~IFS04**: $\overline{\text{SCS}}$ 输入源引脚选择位

00: PA3
01: PA3
10: PC0
11: PF6

Bit 3~2 **IFS03~IFS02**: SCK/SCL 输入源引脚选择位

00: PD0
01: PD0
10: PA4
11: PD6

Bit 1~0 **IFS01~IFS00**: RX 输入源引脚选择位

00: PB6
01: PA0
10: PA5
11: PD4

注: 在 SPI 主机模式时, $\text{SIMEN}=1$ 和 $\text{CSEN}=1$, 不论 IFS0[5:4] 位段为何值, PA3、PC0 和 PF6 能作为 SCS 引脚, 当 $\text{SIMEN}=1$, 不论 IFS0[3:2] 位段为何值, PA4、PD0 和 PD6 引脚能作为 SCK 引脚。

● IFS1 寄存器 – HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	IFS17	IFS16	IFS15	IFS14	IFS13	IFS12	IFS11	IFS10
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 **IFS17~IFS16**: INT0 输入源引脚选择位

00: PA4
01: PA4
10: PD7
11: PD7

Bit 5~4 **IFS15~IFS14**: INT1 输入源引脚选择位

00: PB1
01: PB1
10: PD5
11: PD5

Bit 3~2 **IFS13~IFS12**: PTCK0 输入源引脚选择位

00: PB2
01: PB2
10: PE3
11: PE3

Bit 1~0 **IFS11~IFS10:** PTCK1 输入源引脚选择位
 00: PA6
 01: PA6
 10: PE4
 11: PE4

● **IFS2 寄存器 – HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	IFS27	IFS26	IFS25	IFS24	IFS23	IFS22	IFS21	IFS20
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 **IFS27~IFS26:** STPOI 输入源引脚选择位
 00: PA1
 01: PA1
 10: PF4
 11: PF4

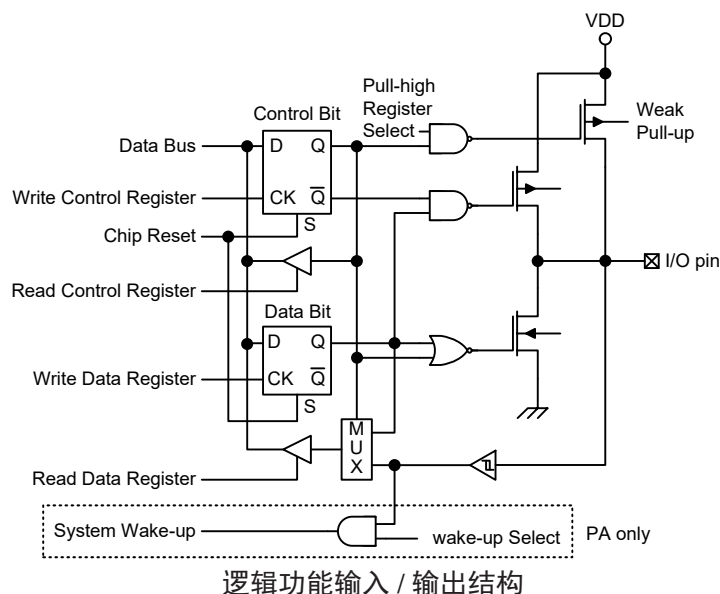
Bit 5~4 **IFS25~IFS24:** STPII 输入源引脚选择位
 00: PC2
 01: PC2
 10: PF2
 11: PF2

Bit 3~2 **IFS23~IFS22:** STCK0 输入源引脚选择位
 00: PA5
 01: PA5
 10: PE1
 11: PE1

Bit 1~0 **IFS21~IFS20:** STCK1 输入源引脚选择位
 00: PC1
 01: PC1
 10: PE6
 11: PE6

输入 / 输出引脚结构

下图为输入 / 输出引脚逻辑功能的内部结构图。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 引脚逻辑功能的理解提供一个参考。由于存在诸多的引脚共用结构，在此不方便提供所有类型引脚功能结构图。



编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器将某些引脚设定为输出状态，这些输出引脚会有初始高电平输出，除非端口数据寄存器在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到对应的端口控制寄存器，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 口的每个引脚都带唤醒功能。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 口一个或多个引脚具有唤醒功能。

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。该系列单片机提供该几个定时器模块（简称 TM），来实现和时间有关的功能。定时器模块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。每个定时器模块有两个独立中断。每个 TM 外加的输入引脚，扩大了定时器的灵活性，便于用户使用。

这里只介绍各种 TM 的共性，更多详细资料请参考标准型和周期型定时器章节。

简介

该系列单片机多达四个 TM，每个 TM 可被划分为一个特定的类型，即标准型 TM 或周期型 TM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍标准型和周期型 TM 的共性，更多详细资料分别见后面各章。两种类型 TM 的特性和区别见下表。

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

TM 功能概要

单片机型号	STM	PTM
HT66F4530	10-bit STM0	10-bit PTM0
HT66F4540	10-bit STM0	10-bit PTM0 10-bit PTM1
HT66F4550/ HT66F4560	10-bit STM0 10-bit STM1	10-bit PTM0 10-bit PTM1

TM 名称 / 类型参考

TM 操作

不同类型的 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 xTMn 控制寄存器的 xTnCK2~xTnCK0 位，选择所需的时钟源，其中 x 代表 S 或 P 类型，n 代表具体 TM 编号。该时钟源来自系统时钟 f_{sys} 的分频比或内部高速时钟 f_H 或 f_{SUB} 时钟源或外部 xTCKn 引脚。xTCKn 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

标准型和周期型 TM 拥有两个内部中断，分别是内部比较器 A 或比较器 P，当比较匹配发生时产生 TM 中断。当 TM 中断产生时，计数器清零并改变 TM 输出状态。

TM 外部引脚

无论哪种类型的 TM，都有一个或两个 TM 输入引脚，分别为 xTCKn 和 xTPnI。xTMn 输入引脚 xTCKn 作为 xTMn 时钟源输入脚，通过设置 xTMnC0 寄存器中的 xTnCK2~xTnCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。xTCKn 输入引脚可选择上升沿有效或下降沿有效。xTCKn 引脚还可分别用作 xTMn 单脉冲输出模式的外部触发引脚。

另一种 xTMn 输入引脚 xTPnI 作为捕捉输入脚，其有效边沿有上升沿、下降沿和双沿，通过设置 xTMnC1 寄存器中的 xTnIO1~xTnIO0 位来选择有效边沿类型。此外，PTCKn 引脚也可用作 PTMn 捕捉输入模式的外部触发引脚。

每个 TM 都有一个输出引脚 xTPn。该 TM 输出引脚可以选择使用引脚共用功能部分描述的相应的引脚共用功能选择位。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 xTPn 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 输出引脚与其它功能共用时，TM 输出功能需要事先通过相关引脚共用功能选择寄存器先被设置。

单片机型号	STM		PTM	
	输入	输出	输入	输出
HT66F4530	STP0I	STP0	PTCK0	PTP0
HT66F4540	STCK0, STP0I	STP0	PTCK0; PTCK1	PTP0; PTP1;
HT66F4550/ HT66F4560	STCK0, STP0I; STCK1, STP1I	STP0; STP1	PTCK0; PTCK1	PTP0; PTP1;

TM 外部引脚

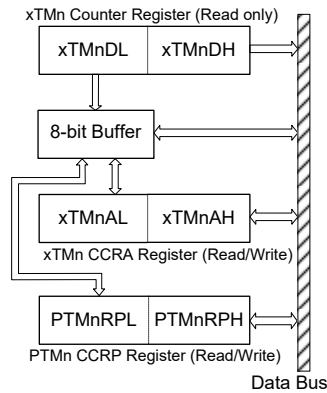
TM 输入 / 输出引脚选择

通过设置与 TM 输入 / 输出引脚相关的引脚共用功能选择寄存器的位，选择作为 TM 输入 / 输出功能或其它共用功能。正确设置选择位将相应的引脚用作 TM 输入 / 输出。更多引脚共用功能选择详见引脚共用功能章节。

编程注意事项

TM 计数寄存器和捕捉/比较寄存器 CCRA 和 CCRP，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。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。

CCRA 和 CCRP 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上述的特殊方式。建议使用“MOV”指令按照以下步骤访问 CCRA 或 CCRP 低字节寄存器，即 xTMnAL 和 PTMnRPL，否则可能导致无法预期的结果。



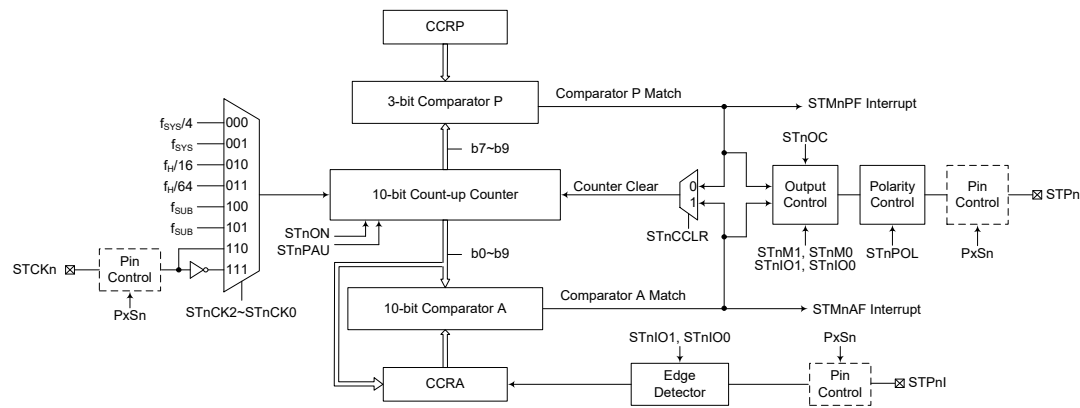
读写流程如下步骤所示：

- 写数据至 CCRA 或 CCRP
 - ◆ 步骤 1. 写数据至低字节寄存器 xTMnAL 或 PTMnRPL
 - 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 xTMnAH 或 PTMnRPH
 - 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 或 CCRP 中读取数据
 - ◆ 步骤 1. 由高字节寄存器 xTMnDH、xTMnAH 或 PTMnRPH 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. 由低字节寄存器 xTMnDL、xTMnAL 或 PTMnRPL 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 可由一个或两个外部输入脚控制并驱动一个外部输出脚。

单片机型号	STM 核	STM 输入引脚	STM 输出引脚
HT66F4530	10-bit STM(STM0)	STP0I	STP0
HT66F4540	10-bit STM(STM0)	STCK0, STP0I	STP0
HT66F4550/ HT66F4560	10-bit STM (STM0, STM1)	STCK0, STP0I; STCK1, STP1I	STP0; STP1



注：1. HT66F4530/HT66F4540 单片机 n=0，HT66F4550/HT66F4560 单片机 n=0~1。
2. STCKn 不可用于 HT66F4530。

标准型 TM 方框图

标准型 TM 操作

标准型 TM 是 10-bit 宽度。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-bit 宽度，与计数器的高 8 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 STnON 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 STMn 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 寄存器介绍

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-bit CCRP 的值。

寄存器名称	位							
	7	6	5	4	3	2	1	0
STMnC0	STnPAU	STnCK2	STnCK1	STnCK0	STnON	STnRP2	STnRP1	STnRP0
STMnC1	STnM1	STnM0	STnIO1	STnIO0	STnOC	STnPOL	STnDPX	STnCCLR
STMnDL	D7	D6	D5	D4	D3	D2	D1	D0
STMnDH	—	—	—	—	—	—	D9	D8
STMnAL	D7	D6	D5	D4	D3	D2	D1	D0
STMnAH	—	—	—	—	—	—	D9	D8

10-bit 标准型 TM 寄存器列表 (n=0~1)

• STMnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	STnPAU	STnCK2	STnCK1	STnCK0	STnON	STnRP2	STnRP1	STnRP0
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 **STnPAU**: STMn 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，STMn 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 **STnCK2~STnCK0**: STMn 计数器时钟选择

- 000: $f_{SYS}/4$
- 001: f_{SYS}
- 010: $f_{H}/16$
- 011: $f_{H}/64$
- 100: f_{SUB}
- 101: f_{SUB}
- 110: STCKn 上升沿时钟 (不可用于 HT66F4530)
- 111: STCKn 下降沿时钟 (不可用于 HT66F4530)

此三位用于选择 STMn 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_{SYS} 是系统时钟， f_{H} 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **STnON**: STMn 计数器 On/Off 控制位

- 0: Off
- 1: On

此位控制 STMn 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 STMn。清零此位将停止计数器并关闭 STMn 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 STMn 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 STnON 位经由低到高转换时，STMn 输出脚将复位至 STnOC 位指定的初始值。

Bit 2~0 **STnRP2~STnRP0**: STMn CCRP 3-bit 寄存器, 与 STMn 计数器 bit 9~bit 7 比较
比较器 P 匹配周期 =

000: 1024 个 STMn 时钟
001: 128 个 STMn 时钟
010: 256 个 STMn 时钟
011: 384 个 STMn 时钟
100: 512 个 STMn 时钟
101: 640 个 STMn 时钟
110: 768 个 STMn 时钟
111: 896 个 STMn 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 STnCCLR 位设定为 0 时, 选中该比较结果清除内部计数器。STnCCLR 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。

● STMnC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	STnM1	STnM0	STnIO1	STnIO0	STnOC	STnPOL	STnDPX	STnCCLR
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 **STnM1~STnM0**: 选择 STMn 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 STMn 需要的工作模式。为了确保操作可靠, STMn 应在 STnM1 和 STnM0 位有任何改变前先关掉。在定时 / 计数器模式, STMn 输出引脚控制除能。

Bit 5~4 **STnIO1~STnIO0**: 选择 STMn 外部引脚 STPn 或 STPnI 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 STPnI 上升沿输入捕捉
01: 在 STPnI 下降沿输入捕捉
10: 在 STPnI 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 STMn 输出脚如何改变状态。这两位值的选择取决于 STMn 运行在何种模式下。

在比较匹配输出模式下，STnIO1~STnIO0 位段决定了当比较器 A 比较匹配发生时 STMn 输出脚如何改变状态。当比较器 A 比较匹配发生时 STMn 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。STMn 输出脚的初始值通过 STnOC 位设置取得。注意，由 STnIO1~STnIO0 位段得到的输出电平必须与通过 STnOC 位设置的初始值不同，否则当比较匹配发生时，STMn 输出脚将不会发生变化。在 STMn 输出脚改变状态后，通过 STnON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，STnIO1~STnIO0 位段用于决定比较匹配条件发生时怎样改变 STMn 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 STMn 关闭时改变 STnIO1~STnIO0 位段的值是很有必要的。若在 STMn 运行时改变 STnIO1~STnIO0 位段的值，PWM 输出的值是无法预料的。

Bit 3 **STnOC**: STMn STPn 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 STMn 输出脚输出控制位。它取决于 STMn 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 STMn 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，其决定比较匹配发生前 STMn 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 STnON 位由低转高时 STMn 输出脚的逻辑电平值。

Bit 2 **STnPOL**: STMn STPn 输出极性控制位

0: 同相

1: 反相

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

Bit 1 **STnDPX**: STMn 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 **STnCCLR**: STMn 计数器清零条件选择位

0: 比较器 P 匹配

1: 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。STnCCLR 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。STnCCLR 位在 PWM 输出、单脉冲输出或捕捉输入模式时未使用。

● STMnDL 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 **D7~D0:** STMn 计数器低字节寄存器 bit 7 ~ bit 0
STMn 10-bit 计数器 bit 7 ~ bit 0

● STMnDH 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 **D9~D8:** STMn 计数器高字节寄存器 bit 1~bit 0
STMn 10-bit 计数器 bit 9~bit 8

● STMnAL 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:** STMn CCRA 低字节寄存器 bit 7 ~ bit 0
STMn 10-bit CCRA bit 7 ~ bit 0

● STMnAH 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 **D9~D8:** STMn CCRA 高字节寄存器 bit 1 ~ bit 0
STMn 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 STMnC1 寄存器的 STnM1 和 STnM0 位选择任意模式。

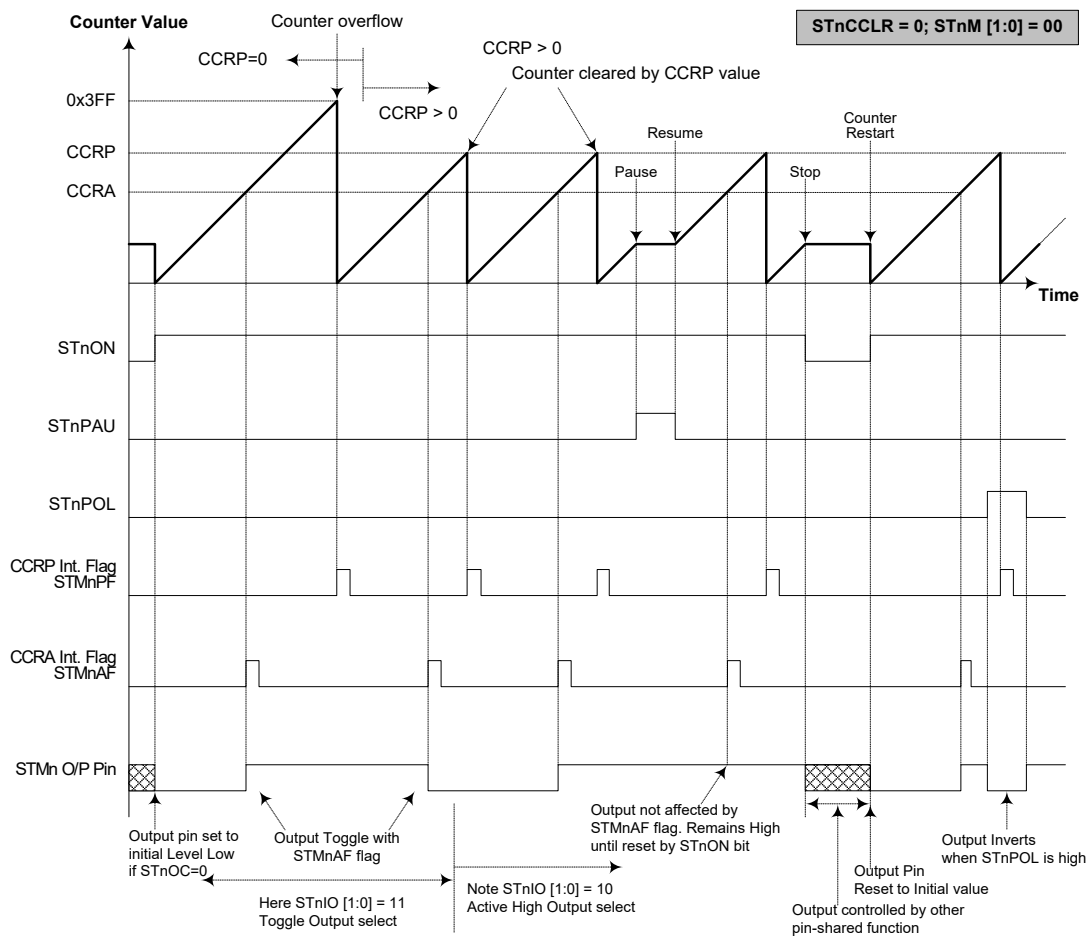
比较匹配输出模式

要工作在此模式，STMnC1 寄存器中的 STnM1 和 STnM0 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 STnCCLR 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 STMnAF 和 STMnPF 将分别置位。

如果 STMnC1 寄存器的 STnCCLR 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 STMnAF 中断请求标志。所以当 STnCCLR 为高时，不会产生 STMnPF 中断请求标志。在比较匹配输出模式，CCRA 的值不能设置为“0”。

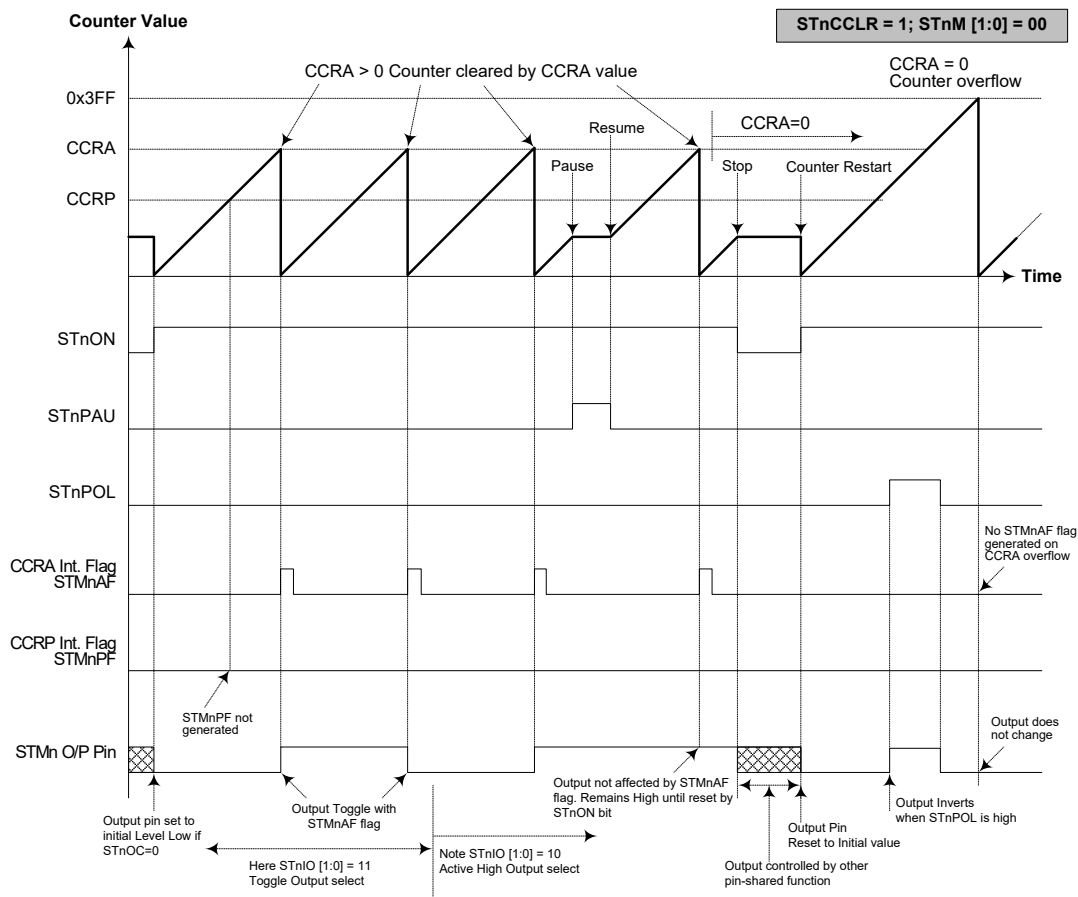
如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 STMnAF 请求标志。

正如该模式名所言，当比较匹配发生后，STMn 输出脚状态改变。当比较器 A 比较匹配发生后 STMnAF 标志产生时，STMn 输出脚状态改变。比较器 P 比较匹配发生时产生的 STMnPF 标志不影响 STMn 输出脚。STMn 输出脚状态改变方式由 STMnC1 寄存器中 STnIO1 和 STnIO0 位决定。当比较器 A 比较匹配发生时，STnIO1 和 STnIO0 位决定 STMn 输出脚输出高，低或翻转当前状态。STMn 输出脚初始值，在 STnON 位由低到高电平的变化后通过 STnOC 位设置。注意，若 STnIO1 和 STnIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – STnCCLR=0 (n=0~1)

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



比较匹配输出模式 – STnCCLR=1 (n=0~1)

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

定时 / 计数器模式

要工作在此模式，STnMC1 寄存器中的 STnM1 和 STnM0 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 STMn 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 STMn 输出脚可通过设置相关引脚共用寄存器用作普通 I/O 脚或其它功能。

PWM 输出模式

要工作在此模式，STMnC1 寄存器中的 STnM1 和 STnM0 位需要设置为“10”，且 STnIO1 和 STnIO0 位也需要设置为“10”。STMn 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 STMn 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，STnCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来控制占空比，另一个用来清除内部计数器并控制 PWM 波形的频率。哪个寄存器控制频率或占空比取决于 STMnC1 寄存器的 STnDPX 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。STnMC1 寄存器中的 STnOC 位决定 PWM 波形的极性，STnIO1 和 STnIO0 位使能 PWM 输出或将 STMn 输出脚置为逻辑高或逻辑低。STnPOL 位对 PWM 输出波形的极性取反。

• 10-bit STMn，PWM 输出模式，边沿对齐模式 – STnDPX=0

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

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

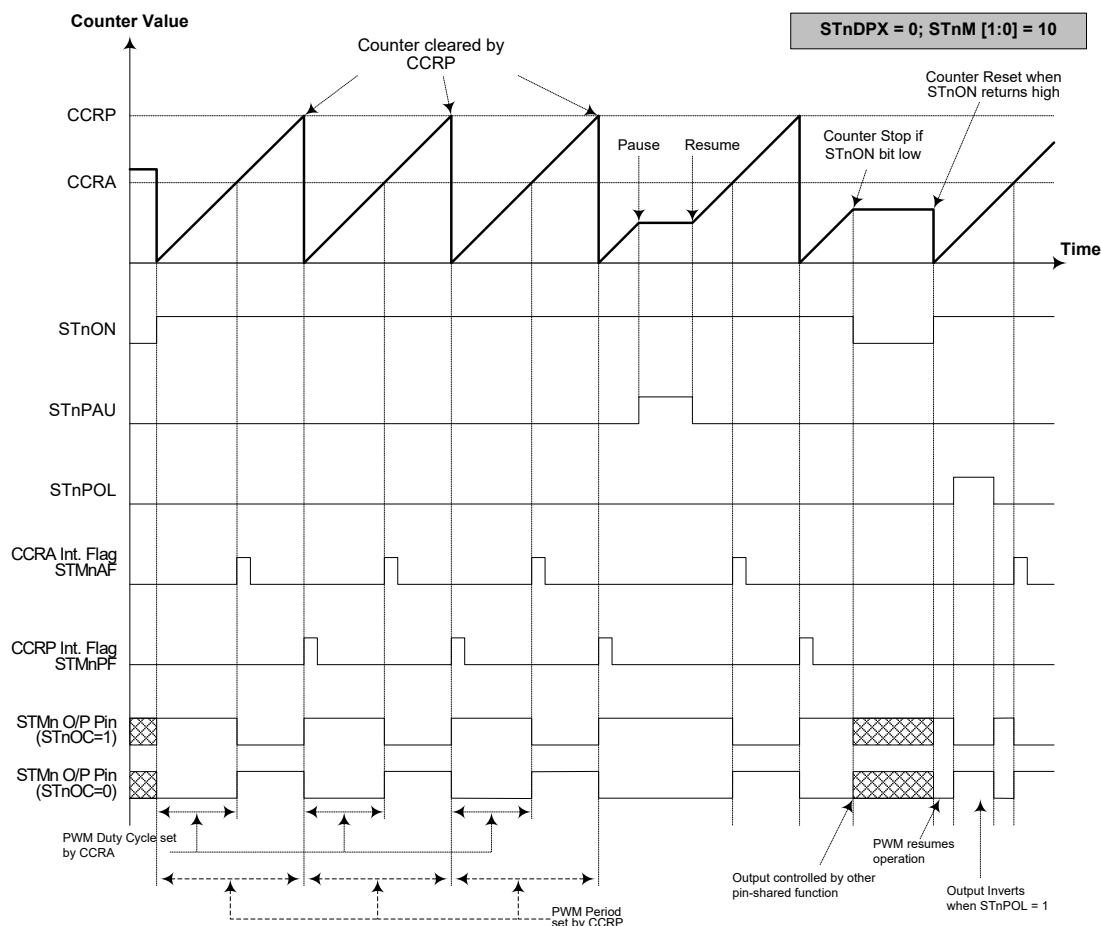
STMn PWM 输出频率 = $(f_{SYS}/4)/512 = f_{SYS}/2048 = 1.9531\text{kHz}$ ，Duty = $128/512 = 25\%$ 。

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 STMn，PWM 输出模式，边沿对齐模式 – STnDPX=1

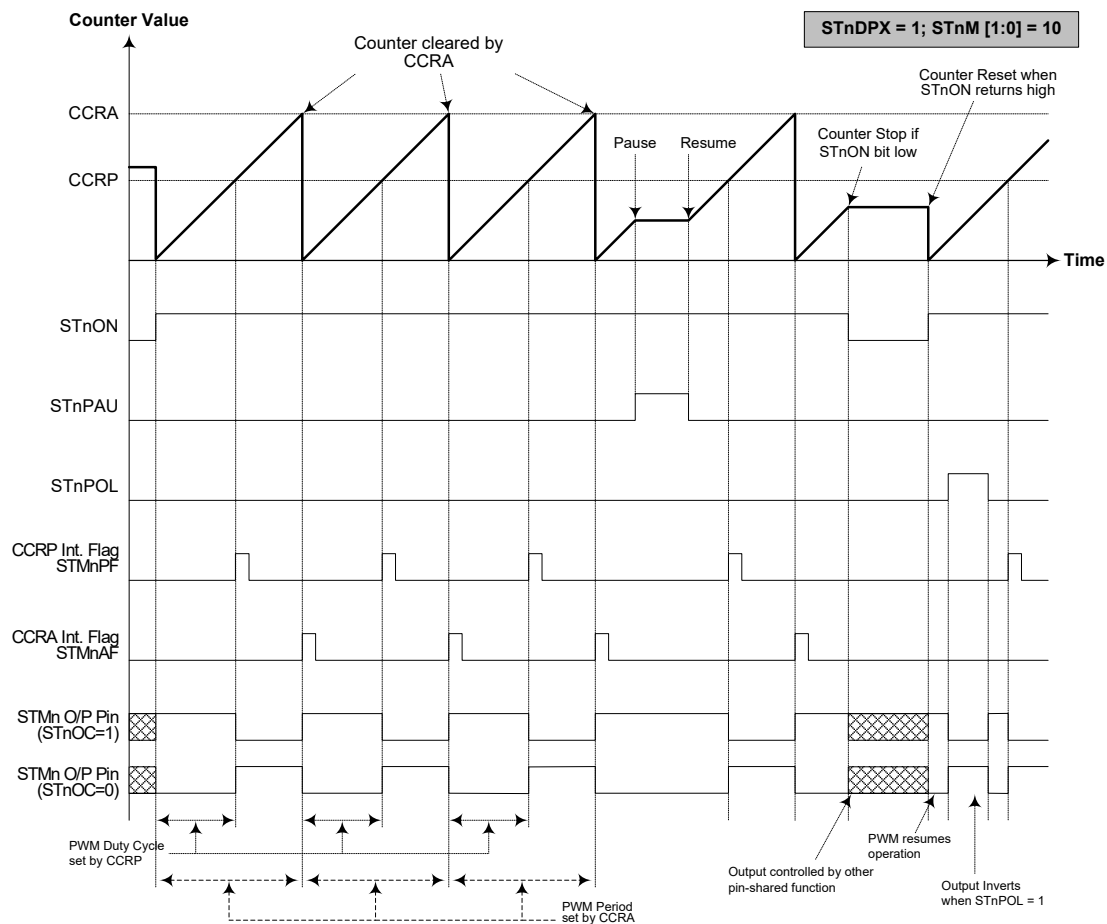
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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



PWM 输出模式 – STnDPX=0 (n=0~1)

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



PWM 输出模式 – STnDPX=1 (n=0~1)

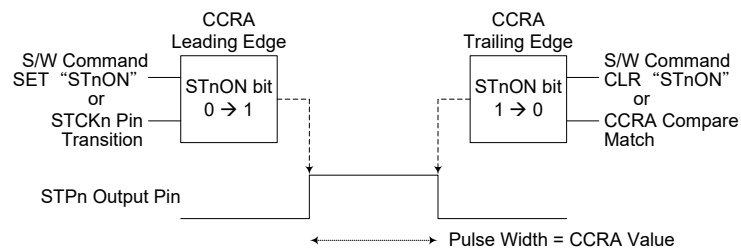
- 注：1. STnDPX=1 – 计数器由 CCRA 清除。
2. 计数器清除设置 PWM 周期。
3. 即使当 STnIO[1:0]=00 或 01 时，内部 PWM 功能继续运行。
4. STnCCLR 对 PWM 操作无影响。

单脉冲输出模式

要工作在此模式，STnMC1 寄存器中的 STnM1 和 STnM0 位需要设置为“10”，同时 STnIO1 和 STnIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STMn 输出脚将产生一个单脉冲输出。

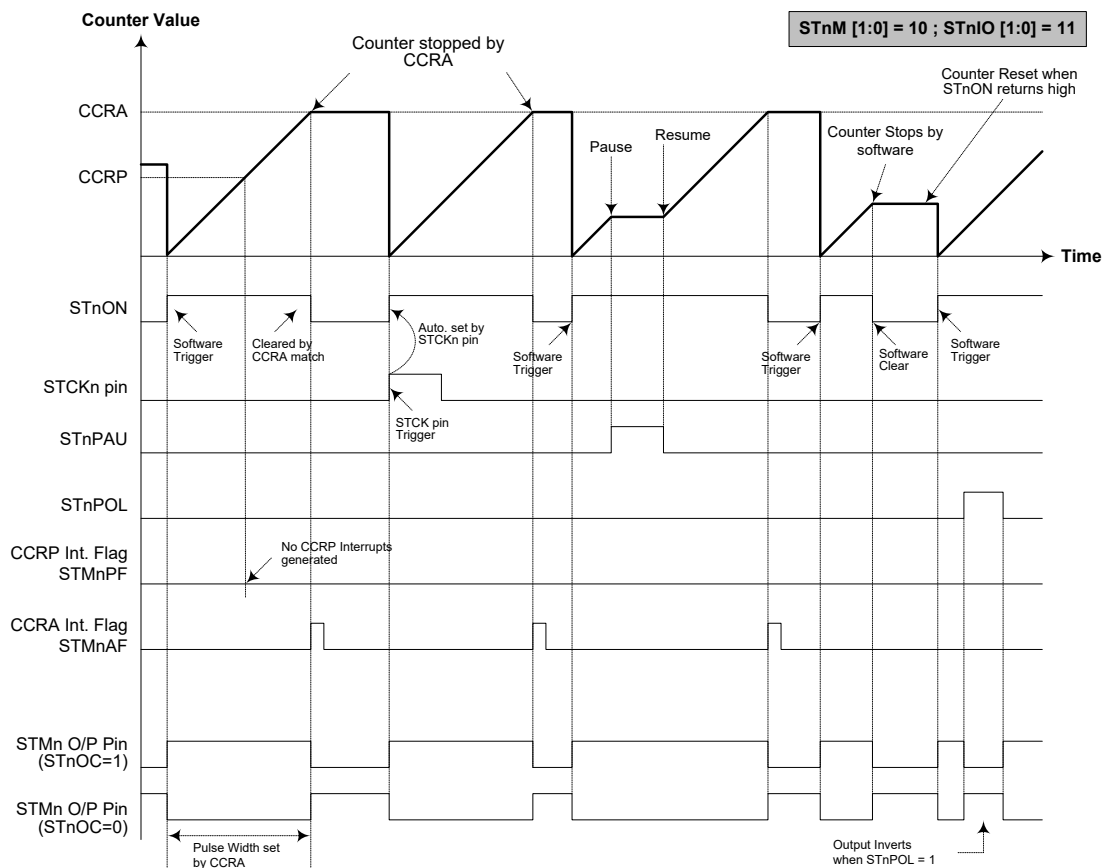
脉冲输出可以通过应用程序控制 STnON 位由低到高的转变来触发。而处于单脉冲模式时，STnON 位可在 STCKn 脚发生有效边沿跳转时自动由低转变为高，进而开始单脉冲输出。当 STnON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 STnON 位保持高电平。通过应用程序使 STnON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

比较器 A 的比较匹配发生时，会自动清除 STnON 位并产生单脉冲输出后沿。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 STMn 中断信号。当计数器重新启动，STnON 位从低到高转换时，计数器将被复位至零。在单脉冲模式下，CCRP 寄存器，STCCLR 位和 STnDPX 位未使用。



注：STCKn 不可用于 HT66F4530。

单脉冲产生示意图 (n=0~1)



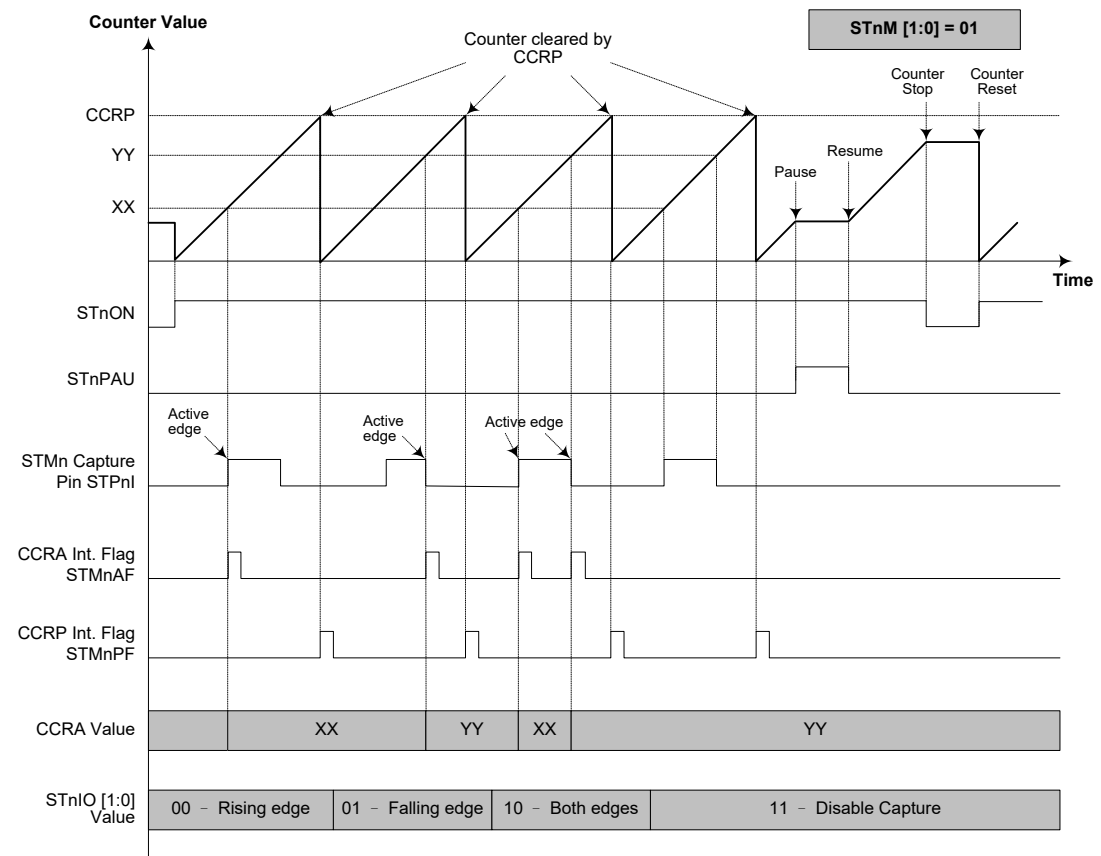
单脉冲输出模式 (n=0~1)

- 注：1. 通过 CCRA 匹配停止计数器。
2. CCRP 未使用。
3. 通过 STCKn 引脚或设置 STnON 为高触发脉冲。
4. STCKn 脚有效沿会自动置位 STnON。
5. 在单脉冲模式下，STnIO[1:0] 必须设为“11”且不能被改变。
6. STCKn 不可用于 HT66F4530。

捕捉输入模式

要工作在此模式，STMnC1 寄存器中的 STnM1 和 STnM0 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。STPnI 脚上的外部信号，通过设置 STMnC1 寄存器的 STnIO1~STnIO0 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 STnON 位由低到高转变时，计数器启动。

当 STPnI 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 STMn 中断。无论 STPnI 引脚发生何种事件，计数器将继续工作直到 STnON 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 STMn 中断。记录 CCRP 溢出中断信号的值可以测量脉宽。通过设置 STnIO1~STnIO0 位段选择 STPnI 引脚为上升沿，下降沿或双沿有效。如果 STnIO1~STnIO0 位段都设置为高，无论 STPnI 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。STnCCLR 和 STnDPX 位在此模式中未使用。



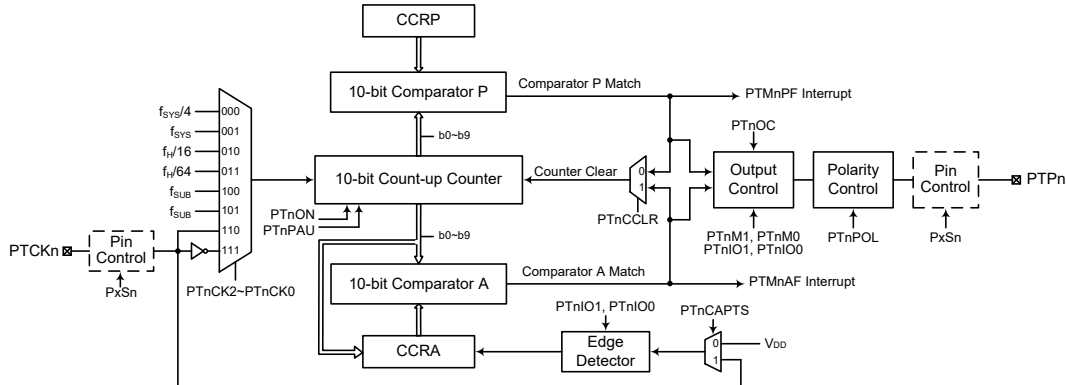
捕捉输入模式 (n=0~1)

- 注：1. STnM [1:0]=01，有效边沿通过 STnIO[1:0] 位段设置。
2. STMn 捕捉输入引脚有效边沿将计数器的值传到 CCRA 中。
3. STnCCLR 位未使用。
4. 无输出功能 - STnOC 和 STnPOL 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。

周期型 TM – PTM

周期型 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。周期型 TM 由一个外部输入脚控制并驱动一个外部输出脚。

单片机型号	PTM 核	PTM 输入引脚	PTM 输出引脚
HT66F4530	10-bit PTM (PTM0)	PTCK0	PTP0
HT66F4540/ HT66F4550/ HT66F4560	10-bit PTM (PTM0, PTM1)	PTCK0; PTCK1	PTP0; PTP1



注：1. HT66F4530 单片机 n=0，HT66F4540/HT66F4550/HT66F4560 单片机 n=0~1。

2. HT66F45x0 系列单片机捕捉输入仅来自 PTCKn 引脚。

周期型 TM 方框图

周期型 TM 操作

周期型 TM 是 10 位宽度。周期型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 10 位向上计数器，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 CCRA 和 CCRP 寄存器中的值进行比较。CCRP 是 10 位的。

通过应用程序改变 10 位计数器值的唯一方法是使 PTnON 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 PTMn 中断信号。周期型 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。

周期型 TM 寄存器介绍

周期型 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 位计数器的值，两对读 / 写寄存器存放 10 位 CCRA 和 CCRP 的值。剩下两个控制寄存器用来设置不同的操作和控制模式。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PTMnC0	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
PTMnC1	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCPTS	PTnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	—	—	—	—	—	—	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	—	—	—	—	—	—	D9	D8
PTMnRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnRPH	—	—	—	—	—	—	D9	D8

10-bit 周期型 TM 寄存器列表 (n=0~1)

● PTMnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTnPAU**: PTMn 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，PTMn 保持上电状态并继续耗电。当此位由低到高转变时，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 **PTnCK2~PTnCK0**: 选择 PTMn 计数时钟位

000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: PTCKn 上升沿
111: PTCKn 下降沿

此三位用于选择 PTMn 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_{SYS} 是系统时钟， f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **PTnON**: PTMn 计数器 On/Off 控制位

0: Off
1: On

此位控制 PTMn 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 PTMn。清零此位将停止计数器并关闭 PTMn 减少耗电。当此位经由低到高转变时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。

若 PTMn 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式时，当 PTnON 位经由低到高转换时，PTMn 输出脚将复位至 PTnOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

• PTMnC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
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 **PTnM1~PTnM0**: 选择 PTMn 工作模式位

- 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 PTMn 需要的工作模式。为了确保操作可靠，PTMn 应在 PTnM1 和 PTnM0 位有任何改变前先关掉。在定时 / 计数器模式，PTMn 输出脚控制必须除能。

Bit 5~4 **PTnIO1~PTnIO0**: 选择 PTMn 外部引脚 PTPn 或 PTCKn 功能位

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: 强制无效状态
- 01: 强制有效状态
- 10: PWM 输出
- 11: 单脉冲输出

捕捉输入模式

- 00: 在 PTCKn 上升沿输入捕捉
- 01: 在 PTCKn 下降沿输入捕捉
- 10: 在 PTCKn 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 PTMn 输出脚如何改变状态。这两位值的选择决定 PTMn 运行在何种模式下。

在比较匹配输出模式下，PTnIO1 和 PTnIO0 位决定当从比较器 A 比较匹配输出发生时 PTMn 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTMn 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。PTMn 输出脚的初始值通过 PTMnC1 寄存器的 PTnOC 位设置取得。注意，由 PTnIO1 和 PTnIO0 位得到的输出电平必须与通过 PTnOC 位设置的初始值不同，否则当比较匹配发生时，PTMn 输出脚将不会发生变化。在 PTMn 输出脚改变状态后，通过 PTnON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，PTnIO1 和 PTnIO0 用于决定比较匹配条件发生时怎样改变 PTMn 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 PTMn 关闭时改变 PTnIO1 和 PTnIO0 位的值是很有必要的。若在 PTMn 运行时改变 PTnIO1 和 PTnIO0 的值，PWM 输出的值是无法预料的。

- Bit 3 **PTnOC**: PTMn PTPn 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 PTMn 输出脚输出控制位。它取决于 PTMn 此时正运行于比较匹配输出模式还是 PWM 输出模式 / 单脉冲输出模式。若 PTMn 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, 其决定比较匹配发生前 PTMn 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 PTnON 位由低变高时 PTMn 输出脚的逻辑电平。
- Bit 2 **PTnPOL**: PTMn PTPn 输出极性控制位
0: 同相
1: 反相
此位控制 PTPn 输出脚的极性。此位为高时 PTMn 输出脚反相, 为低时 PTMn 输出脚同相。若 PTMn 处于定时 / 计数器模式时其不受影响。
- Bit 1 **PTnCAPTS**: 选择 PTMn 捕捉触发源
0: 未定义
1: 来自 PTCKn 引脚
注: 当 PTMn 用于捕捉输入模式, 该位必须设置为高。
- Bit 0 **PTnCCLR**: 选择 PTMn 计数器清零条件位
0: PTMn 比较器 P 匹配
1: PTMn 比较器 A 匹配
此位用于选择清除计数器的方法。周期型 TM 包括两个比较器 -- 比较器 A 和比较器 P, 两者都可以用作清除内部计数器。PTnCCLR 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。PTnCCLR 位在 PWM 输出模式、单脉冲输出模式或输入捕捉模式时未使用。

● PTMnDL 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 **D7~D0**: PTMn 计数器低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit 计数器 bit 7 ~ bit 0

● PTMnDH 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 **D9~D8**: PTMn 计数器高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit 计数器 bit 9 ~ bit 8

• PTMnAL 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:** PTMn CCRA 低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit CCRA bit 7 ~ bit 0

• PTMnAH 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 **D9~D8:** PTMn CCRA 高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit CCRA bit 9 ~ bit 8

• PTMnRPL 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:** PTMn CCRP 低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit CCRP bit 7 ~ bit 0

• PTMnRPH 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 **D9~D8:** PTMn CCRP 高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit CCRP bit 9 ~ bit 8

周期型 TM 工作模式

周期型 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 PTMnC1 寄存器的 PTnM1 和 PTnM0 位选择任意模式。

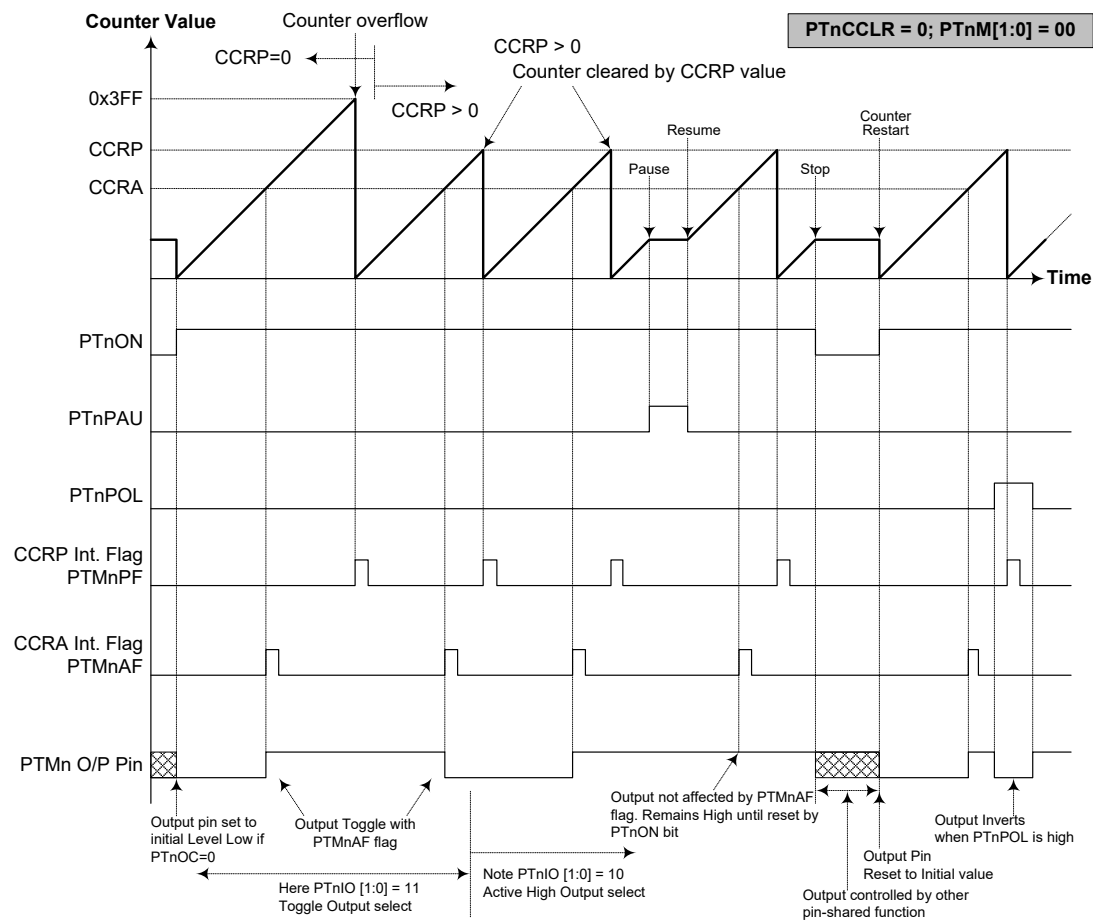
比较匹配输出模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 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 PTnCCLR 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 PTMnAF 和 PTMnPF 将分别置起。

如果 PTMnC1 寄存器的 PTnCCLR 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 PTMnAF 中断请求标志产生。所以当 PTnCCLR 为高时，不会产生 PTMnPF 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。

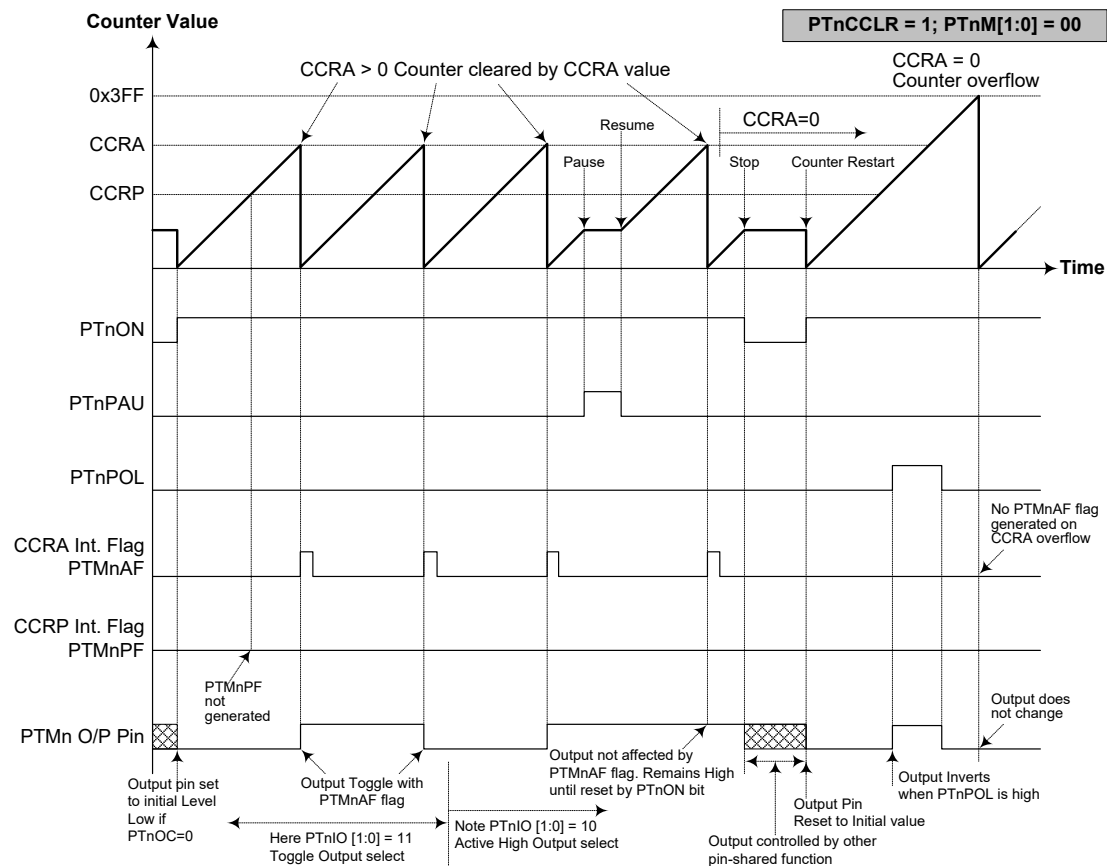
如果 CCRA 位都清除为零，当计数器的值达到 10 位最大值 3FFH 时将溢出，但此时不会产生 PTMnAF 中断请求标志。

正如该模式名所言，当比较匹配发生后，PTMn 输出脚状态改变。当比较器 A 比较匹配发生后 PTMnAF 中断请求标志产生时，PTMn 输出脚状态改变。比较器 P 比较匹配发生时产生的 PTMnPF 标志不影响 PTMn 输出脚。PTMn 输出脚状态改变方式由 PTMnC1 寄存器中 PTnIO1 和 PTnIO0 位决定。当比较器 A 比较匹配发生时，PTnIO1 和 PTnIO0 位决定 PTMn 输出脚输出高，低或翻转当前状态。PTMn 输出脚初始值，在 PTnON 位由低到高电平的变化后通过 PTnOC 位设置。注意，若 PTnIO1 和 PTnIO0 位同时为 0 时，引脚输出不变。



比较器匹配输出模式 - PTnCCLR = 0 (n=0~1)

- 注：1. PTnCCLR=0，比较器 P 匹配将清除计数器
2. PTMn 输出脚仅由 PTMnAF 标志位控制
3. 在 PTnON 上升沿 PTMn 输出脚复位至初始值



比较器匹配输出模式 – PTnCCLR = 1 (n=0~1)

- 注：1. PTnCCLR=1，比较器 A 匹配将清除计数器
2. PTMn 输出脚仅由 PTMnAF 标志位控制
3. 在 PTnON 上升沿 PTMn 输出脚复位至初始值
4. 当 PTnCCLR=1，不产生 PTMnPF 标志位

定时 / 计数器模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 PTMn 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 PTMn 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 位需要设置为“10”，且 PTnIO1 和 PTnIO0 位也需要设置为“10”。PTMn 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 PTMn 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。

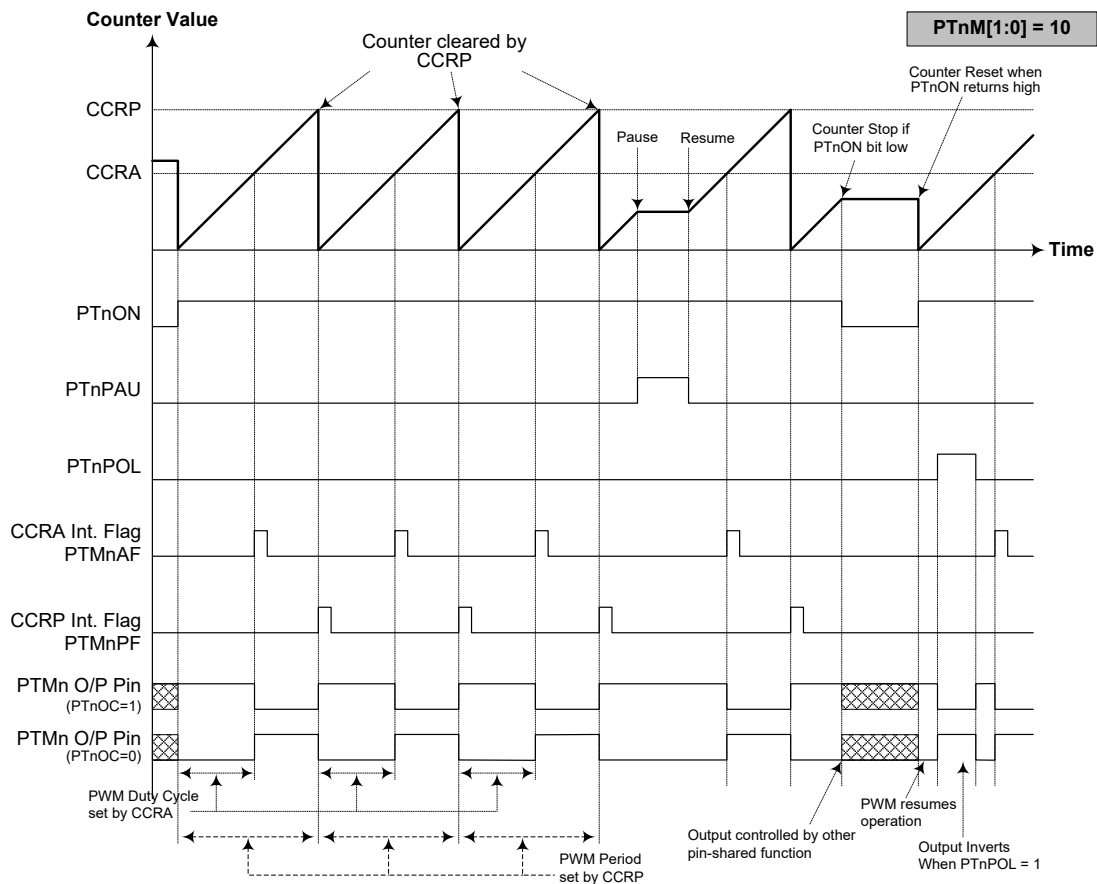
由于 PWM 波形的周期和占空比可调，其波形的选择就极其灵活。在 PWM 输出模式中，PTnCCLR 位对 PWM 周期无影响。CCRP 和 CCRA 寄存器都用于控制 PWM 方波。CCRP 寄存器通过清除内部计数从而控制 PWM 周期，CCRA 寄存器设置 PWM 的占空比。PWM 波形的周期和占空比由 CCRP 和 CCRA 寄存器的值控制。

当比较器 A 或比较器 P 比较匹配发生时，CCRA 和 CCRP 中断标志位分别产生。PTMnC1 寄存器的 PTnOC 位选择 PWM 波形的极性，PTnIO1 和 PTnIO0 位使能 PWM 输出或强制 PTMn 输出脚为高电平或低电平。PTnPOL 位用于 PWM 输出波形的极性反相控制。

● 10-bit PTMn，PWM 输出模式，边沿对齐模式

CCRP	1~1023	0
Period	1~1023	1024
Duty	CCRA	

若 $f_{SYS}=12\text{MHz}$ ，PTMn 时钟源选择 $f_{SYS}/4$ ，CCRP=512 且 CCRA=128，
PTMn PWM 输出频率 $= (f_{SYS}/4) / 512 = f_{SYS}/2048 = 5.8594\text{kHz}$ ，Duty=128/512=25%，
若由 CCRA 寄存器定义的 Duty 值等于或大于 Period 值，PWM 输出占空比为 100%。



PWM 输出模式 (n=0~1)

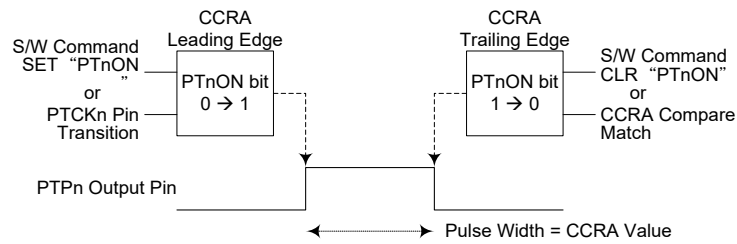
- 注：1. CCRP 清除计数器
2. 计数器清除并决定 PWM 周期
3. 当 PTnIO[1:0]=00 或 01，PWM 功能不变
4. PTnCCLR 位对 PWM 功能无影响

单脉冲输出模式

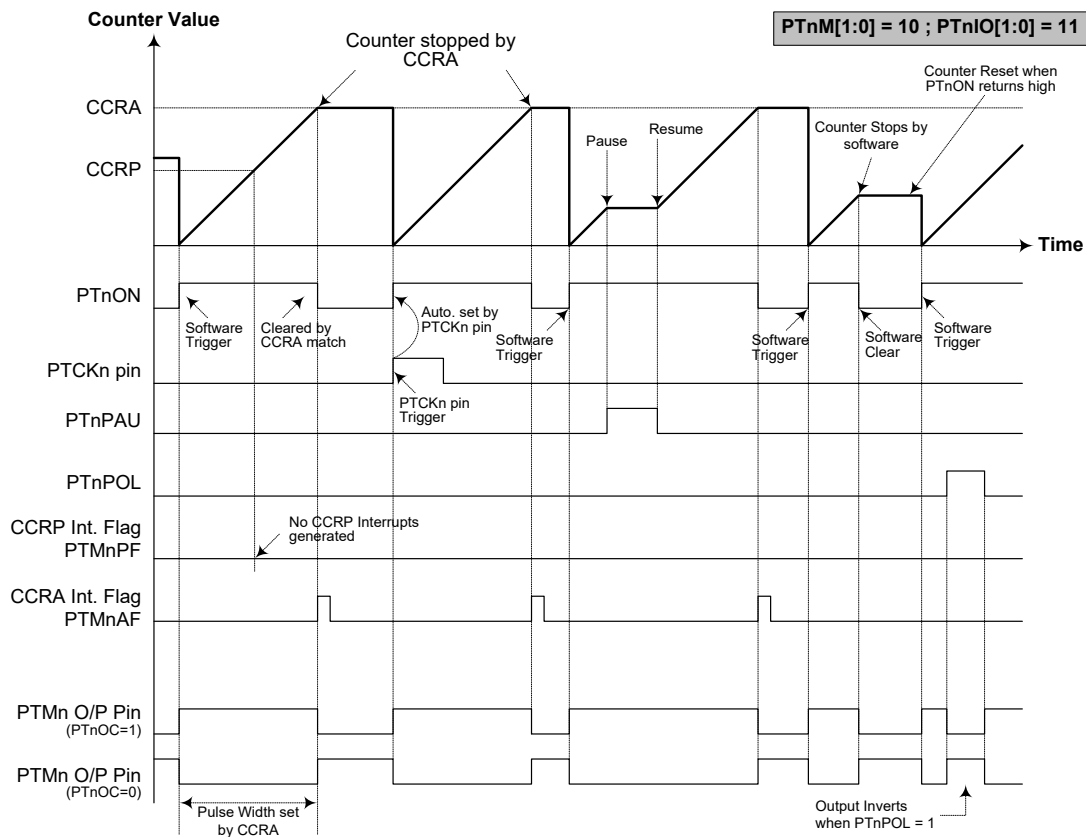
为使 PTMn 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“10”，并且相应的 PTnIO1 和 PTnIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTMn 输出脚将产生一个脉冲输出。

通过应用程序控制 PTnON 位由低到高的转变来触发脉冲前沿输出。而处于单脉冲输出模式时，PTnON 位可在 PTCKn 脚发生有效边沿跳转时自动由低转变为高，进而开始单脉冲输出。当 PTnON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。通过应用程序使 PTnON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

而比较器 A 比较匹配发生时，会自动清除 PTnON 位并产生单脉冲输出后沿。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 PTMn 中断。PTnON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲输出模式中，CCRP 寄存器和 PTnCCCLR 位未使用。



单脉冲产生示意图 (n=0~1)



单脉输出冲模式 (n=0~1)

- 注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCKn 脚或设置 PTnON 位为高来触发脉冲
4. PTCKn 脚有效沿会自动置位 PTnON
5. 在单脉冲输出模式，PTnIO[1:0] 需设置为“11”且不可改变

捕捉输入模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 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。PTCKn 引脚上的外部信号，通过设置 PTMnC1 寄存器的 PTnCAPTS 位选择。可通过设置 PTMnC1 寄存器的 PTnIO1 和 PTnIO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将 PTnON 位由低变为高时，计数器启动。

当 PTCKn 引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 PTMn 中断。无论 PTCKn 引脚发生哪种边沿转换，计数器将继续工作直到 PTnON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；通过这种方式 CCRP 的值可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 PTMn 中断。记录 CCRP 溢出中断信号的值可以测量长脉宽。通过设置 PTnIO1 和 PTnIO0 位选择 PTCKn 引脚为上升沿，下降沿或双沿有效。如果 PTnIO1 和 PTnIO0 位都设置为高，无论 PTCKn 引脚发生哪种边沿转换都不会产生捕捉操作，但计数器仍会继续运行。

当 PTCKn 引脚与其它功能共用，PTMn 工作在输入捕捉模式时需多加注意。这是因为如果引脚被设为输出，那么该引脚上的任何电平转变都可能执行输入捕捉操作。PTnCCLR，PTnOC 和 PTnPOL 位在此模式中未使用。

A/D 转换器

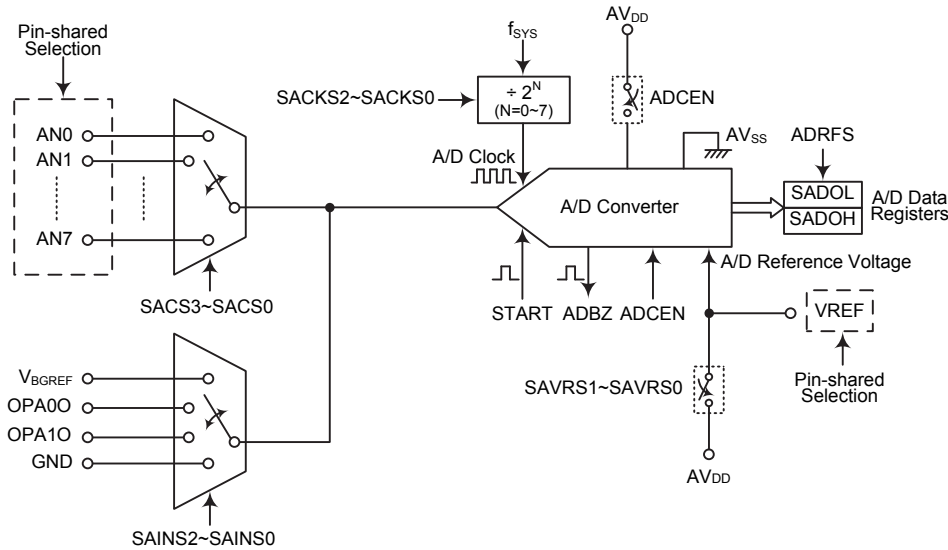
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 转换器简介

此系列单片机包含一个多通道的 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）或内部模拟信号（高性能的 bandgap 参考电压 V_{BGREF} 、SD 运算放大器 0 的输出信号 OPA0O 和 SD 运算放大器 1 的输出信号 OPA1O）并直接将这些信号转换成 12 位的数字量。选择转换外部或内部模拟信号由 SAINS2~SAINS0 位和 SACS3~SACS0 位共同控制。关于 A/D 输入信号的详细描述请参考“A/D 转换器控制寄存器”和“A/D 转换器输入信号”两节内容。

单片机型号	外部输入通道数	内部信号	通道选择位
HT66F4530	5: AN0~AN4	3: V_{BGREF} , OPA0O, OPA1O	SAINS2~SAINS0, SACS3~SACS0
HT66F4540/ HT66F4550/ HT66F4560	8: AN0~AN7		

下图显示了 A/D 转换器整体的内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由一系列寄存器控制。一对只读寄存器来存放 12 位 A/D 转换数据的值。剩下两个控制寄存器设置 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 (ADRF5=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRF5=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRF5=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRF5=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ADCEN	ADRF5	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS2	SAINS1	SAINS0	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0

A/D 转换寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

对于具有 12 位 A/D 转换器的芯片，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 转换数据结果位。未使用的位读为“0”。应注意，当 A/D 转换器除能时，数据寄存器的内容不变。

ADRF5	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

寄存器 SADC0 和 SADC1 用来控制 A/D 转换器的功能和操作。这些 8 位的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，并控制和监视 A/D 转换器的忙碌状态。由于每个单片机只包含一个实际的模数转换电路，因此这些外部和内部模拟信号中的每一个都需要分别被发送到转换器。SADC0 寄存器中的 SACS3~SACS0 位用于选择哪个外部模拟输入通道被连接到内部 A/D 转换器。SADC1 寄存器中的 SAINS2~SAINS0 位用于选择外部模拟输入通道或内部模拟信号被连接到内部 A/D 转换器。

引脚共用功能选择寄存器的相关位用来定义 I/O 端口中的哪些引脚为 A/D 转换器的模拟输入，哪些引脚不作为 A/D 转换输入。当引脚作为 A/D 输入时，其原来的 I/O 或其它引脚共用功能消失，此外，其内部上拉电阻也将自动断开。

• SADC0 寄存器 – HT66F4530

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	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
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。
- Bit 6 **ADBZ**: A/D 转换忙碌标志位
0: A/D 转换结束或未开始转换
1: A/D 转换中
此位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 转换已初始化。A/D 转换结束后，此位被清零。
- Bit 5 **ADCEN**: A/D 转换器使能 / 除能控制位
0: 除能
1: 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 转换器除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将保持不变。
- Bit 4 **ADRFS**: A/D 转换数据格式选择位
0: A/D 转换数据格式 → SADOH=D[11:4]; SADOL=D[3:0]
1: A/D 转换数据格式 → 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 位 A/D 转换结果的格式。细节方面请参考 A/D 数据寄存器章节。
- Bit 3~0 **SACS3~SACS0**: A/D 外部模拟通道输入选择位
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100: AN4
0101~1111: 未定义，输入浮空

• SADC0 寄存器 – HT66F4540/HT66F4550/HT66F4560

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	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
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。
- Bit 6 **ADBZ**: A/D 转换忙碌标志位
0: A/D 转换结束或未开始转换
1: A/D 转换中
此位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 转换已初始化。A/D 转换结束后，此位被清零。

- Bit 5 **ADCEN:** A/D 转换器使能 / 除能控制位
 0: 除能
 1: 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 转换器除能时, A/D 数据寄存器 SADOH 和 SADOL 的内容将保持不变。
- Bit 4 **ADRF5:** A/D 转换数据格式选择位
 0: A/D 转换数据格式 → SADOH=D[11:4]; SADOL=D[3:0]
 1: A/D 转换数据格式 → 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 位 A/D 转换结果的格式。细节方面请参考 A/D 数据寄存器章节。
- Bit 3~0 **SACS3~SACS0:** A/D 外部模拟通道输入选择位
 0000: AN0
 0001: AN1
 0010: AN2
 0011: AN3
 0100: AN4
 0101: AN5
 0110: AN6
 0111: AN7
 1000~1111: 未定义, 输入浮空

• SADC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SAINS2	SAINS1	SAINS0	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
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 7~5 **SAINS2~SAINS0:** A/D 输入信号选择位
 000: 外部信号 – 外部模拟通道输入
 001: 内部信号 – 内部高性能 bandgap 参考电压 V_{BGREF}
 010: 内部信号 – 内部 SD 运算放大器 0 输出信号 OPA00
 011: 内部信号 – 内部 SD 运算放大器 1 输出信号 OPA10
 100: 内部信号 – 未使用, 连接到地
 101~111: 外部信号 – 外部模拟通道输入
 当 SAINS2~SAINS0 被设为 “001” ~ “100” 选择转换内部模拟信号时, 需特别注意。当选择内部模拟信号时, 应将 SACS3~SACS0 位正确的设置, 避免外部通道输入作为 A/D 输入信号。否则, 已外部输入通道会和内部模拟信号一起连接至内部 A/D 转换器, 这将导致无法预期的损害。
- Bit 4~3 **SAVRS1~SAVRS0:** A/D 转换器参考电压选择位
 00: VREF 引脚
 01: 内部 A/D 转换器电源, AV_{DD}
 1x: VREF 引脚
 这两位用于选择 A/D 转换器参考电压。当 SAVRS1~SAVRS0 为被设为 “01” 选择内部参考电压时需特别注意。当选择 A/D 转换器电源作为 A/D 转换器参考电压时, 外部 VREF 引脚需通过引脚共用控制位选择作为其它共用的引脚功能。否则, 外部 VREF 输入电压会和内部参考电压一起连接至内部 A/D 转换器。

Bit 2~0 **SACKS2~SACKS0**: A/D 时钟源选择位

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: $f_{SYS}/128$

这三位用于选择 A/D 转换器的时钟源。

A/D 转换器操作

SADC0 寄存器中的 START 位，用于开启 A/D 转换器。当单片机设置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。

SADC0 寄存器中的 ADBZ 位用于表明模数转换过程是否正在进行。A/D 转换成功启动后，ADBZ 位会被单片机自动置为“1”。在转换周期结束后，ADBZ 位会自动置为“0”。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，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跳转到相应的 A/D 内部中断地址。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，检查此位是否被清除，作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{SYS} 或其分频，而分频系数由 SADC1 寄存器中的 SACKS2~SACKS0 位决定。虽然 A/D 时钟源是由系统时钟 f_{SYS} 和 SACKS2~SACKS0 位决定，但可选择的最大 A/D 时钟源则有一些限制。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。例如，如果系统时钟速度为 8MHz 时，SACKS2~SACKS0 位不能设为“000”、“001”或“111”。，这样 A/D 转换周期小于 A/D 时钟周期的最小值将会产生不准确的 A/D 转换值。参考如下表，其中用星号 * 标示的值取决于单片机，必须特别注意。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或大于时钟周期的最大值。

f_{SYS}	A/D 时钟周期 (t_{ADCK})							
	SACKS [2:0]=000 (f_{SYS})	SACKS [2:0]=001 ($f_{SYS}/2$)	SACKS [2:0]=010 ($f_{SYS}/4$)	SACKS [2:0]=011 ($f_{SYS}/8$)	SACKS [2:0]=100 ($f_{SYS}/16$)	SACKS [2:0]=101 ($f_{SYS}/32$)	SACKS [2:0]=110 ($f_{SYS}/64$)	SACKS [2:0]=111 ($f_{SYS}/128$)
1MHz	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *	64 μs *	128 μs *
2MHz	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *	64 μs *
4MHz	250ns *	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *
8MHz	125ns *	250ns *	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *
12MHz	83ns *	167ns *	333ns *	667ns	1.33 μs	2.67 μs	5.33 μs	10.67 μs *

A/D 时钟周期范例

使用内部通道 OPA0O/OPA1O 输入 A/D 转换器时，须注意 SD 运算放大器的频带宽度，不同的频带宽度能使用的 A/D 时钟源也不同。

频带宽度控制位	A/D 时钟频率 (kHz)				
	125	250	500	1000	2000
SDAnBW[1:0] = 00	√	—	—	—	—
SDAnBW[1:0] = 01	√	—	—	—	—
SDAnBW[1:0] = 10	√	√	√	√	√
SDAnBW[1:0] = 11	√	√	√	√	√

内部 OPA 输出使用 A/D 时钟表

注：A/D 转换第一笔数据须舍去，不能使用。

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

A/D 转换器参考电压

A/D 转换器参考电压来自正电源电压 AV_{DD} 或外部参考源引脚 VREF，通过 SAVRS1 和 SAVRS0 位选择。当 SAVRS1~SAVRS0 位为“01”时，A/D 转换器参考电压来自 AV_{DD} 引脚。当 SAVRS1~SAVRS0 位为“01”以外的任意值时，A/D 转换器参考电压来自 VREF 引脚。由于 VREF 引脚与其它功能共用，当选择 VREF 引脚作为参考电压源时，VREF 引脚控制位 VREFS 需预先设置为高以使能 VREF 引脚功能且自动除能其它共用引脚功能。然而，当内部 A/D 转换器电源被选作参考电压源时，相关的引脚共用控制位不可选择 VREF 参考电压输入功能，避免 VREF 引脚电压跟内部参考电压一起接入 A/D 转换器。模拟输入值一定不能超过所选的参考电压值。

A/D 转换器输入信号

所有的 A/D 模拟输入引脚都与 I/O 口及其它功能共用。使用 PxS0 和 PxS1 寄存器中的相应位，可以将它们设置为 A/D 转换器模拟输入脚或具有其它功能。如果对应的引脚作为 A/D 转换输入，那么它原来的引脚功能将除能。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 A/D 输入功能选择位使能 A/D 输入时，端口控制寄存器的状态将被重置。

通过配置 SAINS2~SAINS0 位有四个内部模拟信号来自高性能的 bandgap 参考电压 V_{BREF} 、SD 运算放大器 0 的输出信号 OPA0O 和 SD 运算放大器 1 的输出信号 OPA1O 可连接到 A/D 转换器的模拟输入信号。若 SAINS2~SAINS0 位为“000”或“101~111”，则选择转换外部模拟输入信号，具体通道编号由 SACS3~SACS0 位决定。若选择内部模拟信号时，应将 SACS3~SACS0 位设置为一个适当的值，以关闭外部模拟通道输入。否则，外部通道输入会与内部模拟信号一起接入从而导致错误。

V_{BREF} 具有驱动能力的高性能 bandgap 参考电压。当输入电源电压 V_{DD} 变化或温度变化时，它将输出准确的参考电压。而且，该 bandgap 将在低压下启动。因此，该参考电压具有高电源抑制比 (PSRR) 用于低压差线性稳压器 (LDO)。

SAINS[2:0]	SACS[3:0]	输入信号	说明
000, 101~111	0000~0100	AN0~AN4	外部模拟通道输入
	0101~1111	—	未选择外部通道，输入浮空
001	0101~1111	V _{BGREF}	内部高性能 bandgap 参考电压
010	0101~1111	OPA0O	内部 SD 运算放大器 0 输出信号
011	0101~1111	OPA1O	内部 SD 运算放大器 1 输出信号
100	0101~1111	GND	未使用，连接到地

A/D 转换器输入信号选择 – HT66F4530

SAINS[2:0]	SACS[3:0]	输入信号	说明
000, 101~111	0000~0111	AN0~AN7	外部模拟通道输入
	1000~1111	—	未选择外部通道，输入浮空
001	1000~1111	V _{BGREF}	内部高性能 bandgap 参考电压
010	1000~1111	OPA0O	内部 SD 运算放大器 0 输出信号
011	1000~1111	OPA1O	内部 SD 运算放大器 1 输出信号
100	1000~1111	GND	未使用，连接到地

A/D 转换器输入信号选择 – HT66F4540/HT66F4550/HT66F4560

● VBGRC 寄存器

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

Bit 7~1 未定义，读为“0”

Bit 0 **VBGREN**: Bandgap 使能 / 除能控制位

0: 除能

1: 使能

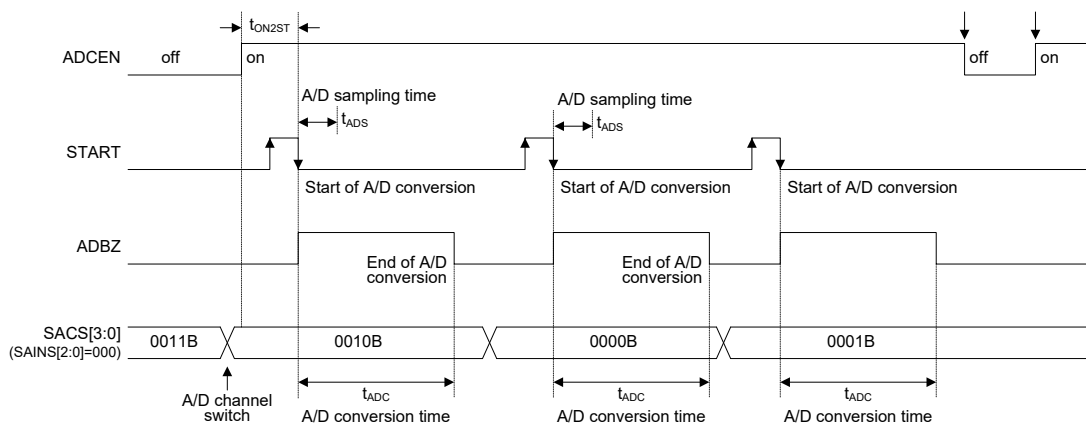
当 VBGREN 位清零，V_{BGREF} 处于高阻态。

A/D 转换率及时序图

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} ，需要 4 个 A/D 时钟周期，而数据转换需要 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。

最大 A/D 转换率 = A/D 时钟周期 ÷ 16

下列时序图表示模数转换过程中不同阶段的图形与时序。由应用程序控制开始 A/D 转换过程后，单片机的内部硬件就会开始进行转换，在这个过程中，程序可以继续其它功能。A/D 转换时间为 $16t_{ADCK}$ ， t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序图

A/D 转换步骤概述

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
通过 SADC1 寄存器中的 SACKS2~SACKS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
将 SADC0 寄存器中的 ADCEN 位置高使能 A/D 转换器。
- 步骤 3
通过配置 SAINS2~SAINS0 位，选择连接至内部 A/D 转换器的信号。
若选择外部通道输入，接着执行步骤 4。
若选择内部模拟信号，接着执行步骤 5。
- 步骤 4
若已通过 SAINS2~SAINS0 位选择 A/D 输入信号来自外部通道输入，接着应设置相关的引脚共用控制位将该引脚规划为 A/D 输入引脚。通过设置 SACS3~SACS0 位选择哪个外部通道接至 A/D 转换器。接着执行步骤 6。
- 步骤 5
选择内部模拟信号前，应将 SACS3~SACS0 设置为为一个适当的值，以关闭外部模拟通道输入。设置 SAINS2~SAINS0 位选择 A/D 输入信号来自内部模拟信号。接着执行步骤 6。
- 步骤 6
通过 SADC1 寄存器中的 SAVRS1~SAVRS0 位选择参考电压。若选择内部参考电压，必须通过合理设置相关引脚共用控制位将外部参考电压输入引脚功能除能。
- 步骤 7
设置 SADC0 寄存器中的 ADRFS 位选择 A/D 转换器输出数据格式。
- 步骤 8
如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。
- 步骤 9
现在可以通过设置 SADC0 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。

● 步骤 10

如果 A/D 转换正在进行中，ADBZ 位会被置为逻辑高。A/D 转换完成后，ADBZ 位会被置为逻辑低，并可从 SADOH 和 SADOL 寄存器中读取输出数据。

注：若使用轮询 SADC0 寄存器中 ADBZ 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

编程注意事项

在编程时，如果 A/D 转换器未使用，通过设置 SADC0 寄存器中的 ADCEN 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

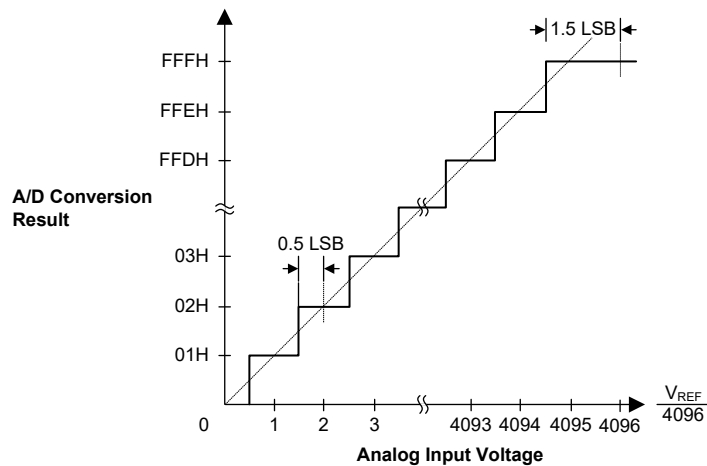
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于实际 A/D 转换器参考电压值， V_{REF} ，因此每一位可表示 $V_{REF}/4096$ 的模拟输入值。

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

通过下面的等式可估算 A/D 转换器输入电压值：

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

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。注意，这里的 V_{REF} 电压指代的是通过 SAVRS 位选择的实际 A/D 转换器参考电压。



理想的 A/D 转换功能

A/D 转换应用范例

下面两个范例程序用来说明怎样使用 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 转换是否完成；第二个范例则使用中断的方式判断。

范例：使用查询 ADBZ 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 ADCEN
mov a,10h                              ; setup PAS1 register to configure pin AN0
mov PAS1,a
mov a,20h
mov SADC0,a                            ; enable and connect AN0 channel to 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                        ; continue polling
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
:
:
jmp start_conversion                    ; start next A/D conversion
```

范例：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 ADCEN
mov a,10h              ; setup PAS1 register to configure pin AN0
mov PAS1,a
mov a,20h
mov SADC0,a            ; enable and connect AN0 channel to 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
clr ADF                ; clear ADC interrupt request flag
set ADE                ; enable ADC interrupt
set EMI                ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a        ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a     ; 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
mov STATUS,a           ; restore STATUS from user defined memory
mov a,acc_stack        ; restore ACC from user defined memory
reti

```

串行接口模块 – SIM

该系列单片机内有一个串行接口模块，包括两种易与外部设备通信的串行接口：四线 SPI 或两线 I²C 接口。这两种接口具有相当简单的通信协议，单片机可以通过这些接口与传感器、闪存或 EEPROM 内存等硬件设备通信。SIM 接口的引脚与其他的 I/O 引脚共用，所以要使用 SIM 功能前应设置相关引脚共用功能选择位。因为这两种接口共用引脚和寄存器，所以要通过 SIM 控制位即 SIMC0 寄存器中的 SIM2~SIM0 位来选择哪一种通信接口。如果 SIM 功能使能和相关引脚作为 SIM 输入引脚，可通过上拉电阻控制寄存器选择与输入 / 输出共用的 SIM 脚的上拉电阻。

SPI 接口

SPI 接口常用于与外部设备如传感器、闪存或 EEPROM 内存等通信。四线 SPI 接口最初是由摩托罗拉公司研制，是一个有相当简单的通信协议的串行数据接口，这个协议可以简化与外部硬件的编程要求。

SPI 通信模式为全双工模式，且能以主 / 从模式的工作方式进行通信，单片机既可以做为主机，也可以做为从机。虽然 SPI 接口理论上允许一个主机控制多个从机，但此处的 SPI 中只有一个片选信号引脚。若主机需要控制多个从机，可使用输入 / 输出引脚选择从机。

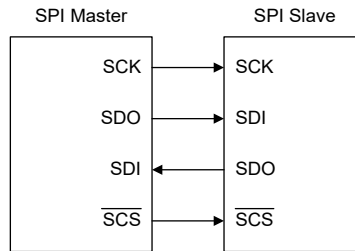
SPI 接口操作

SPI 接口是一个全双工串行数据传输器。SPI 接口的四线为：SDI、SDO、SCK 和 $\overline{\text{SCS}}$ 。SDI 和 SDO 是数据的输入和输出线。SCK 是串行时钟线， $\overline{\text{SCS}}$ 是从机的选择线。SPI 的接口引脚与普通 I/O 口和 I²C 的功能脚共用。通过设定 SIMC0 和 SIMC2 寄存器的对应位，来使能 SPI 接口。连接到 SPI 接口的单片机以从主 / 从模式进行通信，且主机完成所有的数据传输初始化，并控制时钟信号。由于单片机只有一个 $\overline{\text{SCS}}$ 引脚，所以只能拥有一个从机。可通过软件控制 $\overline{\text{SCS}}$ 引脚使能与除能，设置 CSEN 位为“1”使能 $\overline{\text{SCS}}$ 功能，设置 CSEN 位为“0”， $\overline{\text{SCS}}$ 引脚将处于浮空状态。

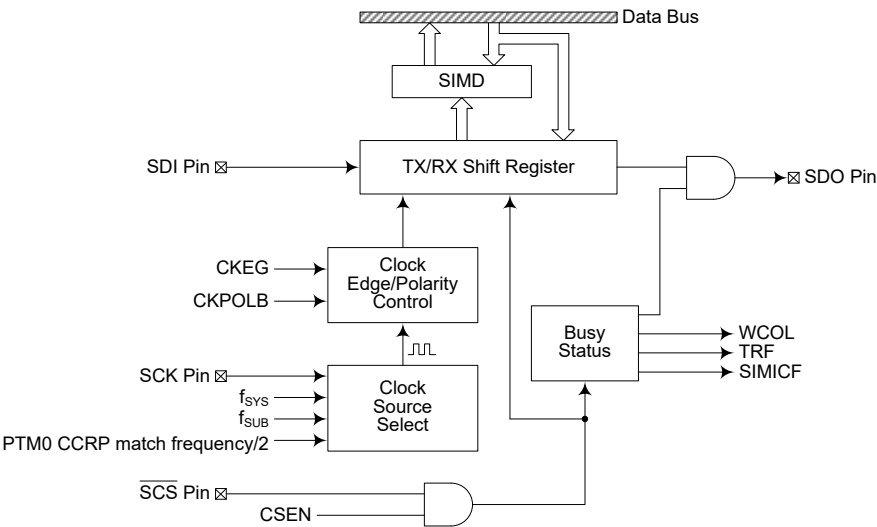
该系列单片机的 SPI 功能具有以下特点：

- 全双工同步数据传输
- 主从模式
- 最低有效位先传或最高有效位先传的数据传输模式
- 传输完成标志位
- 时钟源上升沿或下降沿有效

SPI 接口状态受很多因素的影响，如单片机处于主机或从机的工作模式和 CSEN，SIMEN 位的状态。



SPI 主 / 从机连接方式



SPI 方框图

SPI 寄存器

有三个内部寄存器用于控制 SPI 接口的所有操作，其中有一个数据寄存器 SIMD 和两个控制寄存器 SIMC0, SIMC2。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
SIMD	D7	D6	D5	D4	D3	D2	D1	D0

SPI 寄存器列表

SPI 数据寄存器

SIMD 用于存储发送和接收的数据。这个寄存器由 SPI 和 I²C 功能所共用。在单片机尚未将数据写入到 SPI 总线中时，要传输的数据应先存在 SIMD 中。SPI 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 SPI 传输或接收的数据都必须通过 SIMD 实现。

• SIMD 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	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 D7~D0: SIM 数据寄存器 bit 7~bit 0

SPI 控制寄存器

单片机中也有两个控制 SPI 接口功能的寄存器，SIMC0 和 SIMC2。应注意的是 SIMC2 与 I²C 接口功能中的寄存器 SIMA 是同一个寄存器。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC2 用于其他的控制功能如 LSB/MSB 选择，写冲突标志位等。

• SIMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5 **SIM2~SIM0**: SIM 工作模式控制位

- 000: SPI 主机模式; SPI 时钟为 $f_{\text{SYS}}/4$
- 001: SPI 主机模式; SPI 时钟为 $f_{\text{SYS}}/16$
- 010: SPI 主机模式; SPI 时钟为 $f_{\text{SYS}}/64$
- 011: SPI 主机模式; SPI 时钟为 f_{SUB}
- 100: SPI 主机模式, SPI 时钟为 PTM0 CCRP 匹配频率 / 2
- 101: SPI 从机模式
- 110: I²C 从机模式
- 111: 未使用模式

这几位用于设置 SIM 功能的工作模式, 用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 f_{SUB} 或 PTM0。若选择的是作为 SPI 从机, 则其时钟源从外部主机而得。

Bit 4 未定义, 读为 “0”

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C 去抖动时间选择

此位段在仅用于 I²C 模式中, 该位段描述在 I²C 寄存器章节。

Bit 1 **SIMEN**: SIM 使能控制位

- 0: 除能
- 1: 使能

此位为 SIM 接口的 on/off 控制位。此位为 “0” 时, SIM 接口除能, SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚用于输入 / 输出功能, SIM 工作电流减小到最小值。此位为 “1” 时, SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口, 当 SIMEN 位由低到高转变时, SPI 控制寄存器中的设置不会发生变化, 其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口, 当 SIMEN 位由低到高转变时, I²C 控制寄存器中的设置, 如 HTX 和 TXAK, 将不会发生变化, 其首先应在应用程序中初始化, 此时相关 I²C 标志, 如 HCF、HAAS、HBB、SRW 和 RXAK, 将被设置为其默认状态。

Bit 0 **SIMICF**: SIM SPI 未完成标志位

- 0: 未发生 SIM SPI 未完成现象
- 1: 发生 SIM SPI 未完成现象

此位仅在 SIM 工作在 SPI 从机模式时有效。若 SPI 工作在从机模式且 SIMEN 和 CSEN 都置高, 但 SCS 在 SPI 数据传输结束前由外部主机拉高, 则 SIMICF 与 TRF 都将被置位。此时若中断功能使能则 will 发生中断。但若此位由应用程序软件置位, TRF 位将不会被置 “1”。

● SIMC2 寄存器

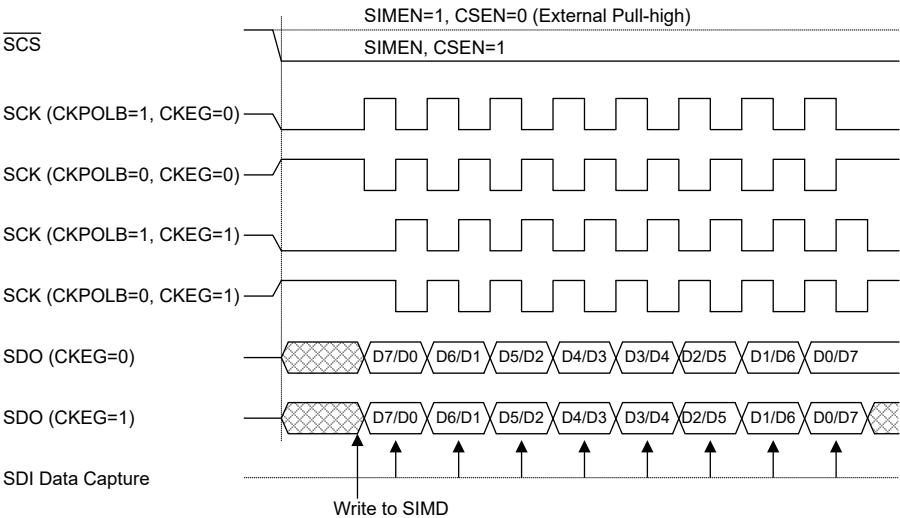
Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

- Bit 7~6 **D7~D6:** 未定义位
用户可通过应用程序对这两位进行读写。
- Bit 5 **CKPOLB:** SPI 时钟线的基础状态位
0: 当时钟无效时, SCK 口为高电平
1: 当时钟无效时, SCK 口为低电平
此位决定了时钟线的基础状态, 当时钟无效时, 若此位为高, SCK 为低电平, 若此位为低, SCK 为高电平。
- Bit 4 **CKEG:** SPI 的 SCK 有效时钟边沿类型位
CKPOLB=0
0: SCK 为高电平且在 SCK 上升沿抓取数据
1: SCK 为高电平且在 SCK 下降沿抓取数据
CKPOLB=1
0: SCK 为低电平且在 SCK 下降沿抓取数据
1: SCK 为低电平且在 SCK 上升沿抓取数据
CKEG 和 CKPOLB 位用于设置 SPI 总线上时钟信号输入和输出方式。在执行数据传输前, 这两位必须被设置, 否则将产生错误的时钟边沿信号。CKPOLB 位决定时钟线的基本状态, 若时钟无效且此位为高, 则 SCK 为低电平, 若时钟无效且此位为低, 则 SCK 为高电平。CKEG 位决定有效时钟边沿类型, 取决于 CKPOLB 的状态。
- Bit 3 **MLS:** SPI 数据移位命令位
0: LSB 优先传输
1: MSB 优先传输
数据移位选择位, 用于选择数据传输时高位优先传输还是低位优先传输。此位设置为高时高位优先传输, 为低时低位优先传输。
- Bit 2 **CSEN:** SPI \overline{SCS} 引脚控制位
0: 除能
1: 使能
CSEN 位用于 \overline{SCS} 引脚的使能 / 除能控制。此位为低时, \overline{SCS} 除能并处于浮空状态。此位为高时, SCS 作为选择脚。
- Bit 1 **WCOL:** SPI 写冲突标志位
0: 无冲突
1: 冲突
WCOL 标志位用于监测数据冲突的发生。此位为高时, 数据在传输时被写入 SIMD 寄存器。若数据正在被传输时, 此操作无效。此位可被应用程序清零。
- Bit 0 **TRF:** SPI 发送 / 接收结束标志位
0: 数据正在发送
1: 数据发送结束
TRF 位为发送 / 接收结束标志位, 当 SPI 数据传输结束时, 此位自动置高, 但须通过应用程序设置为“0”。此位也可用于产生中断。

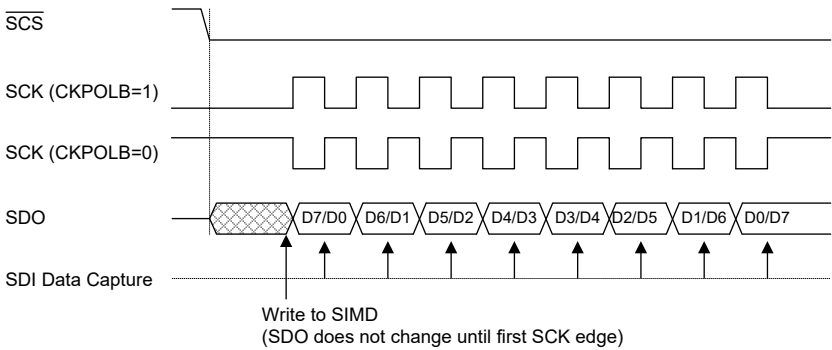
SPI 通信

将 SIMEN 设置为高，使能 SPI 功能之后，单片机处于主机模式，当数据写入到寄存器 SIMD 的同时传输 / 接收开始进行。数据传输完成时，TRF 位将自动被置位但清除只能通过应用程序完成。单片机处于从机模式时，收到主机发来的时钟信号之后，会传输 SIMD 中的数据，而且在 SDI 引脚上的数据也会被移位到 SIMD 寄存器中。主机应在输出时钟信号之前先输出一个 $\overline{\text{SCS}}$ 信号以使能从机，从机的数据传输功能也应在与 SCS 信号相关的适当时候准备就绪，这由 CKPOLB 和 CKEG 位决定。所附时序图表明了 CKPOLB 和 CKEG 位各种设置情况下从机数据与 SCS 信号的关系。

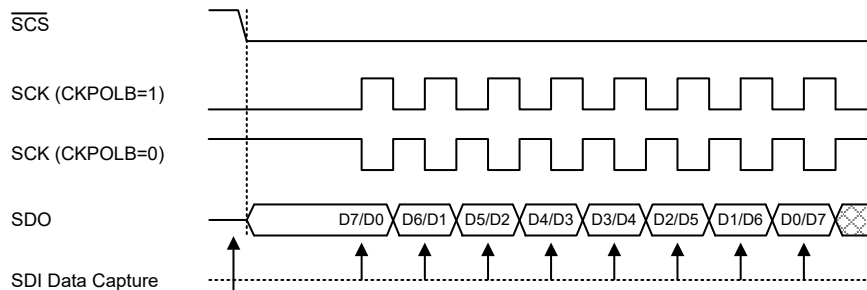
即使在单片机处于空闲模式，SPI 功能仍将继续执行。



SPI 主机模式时序



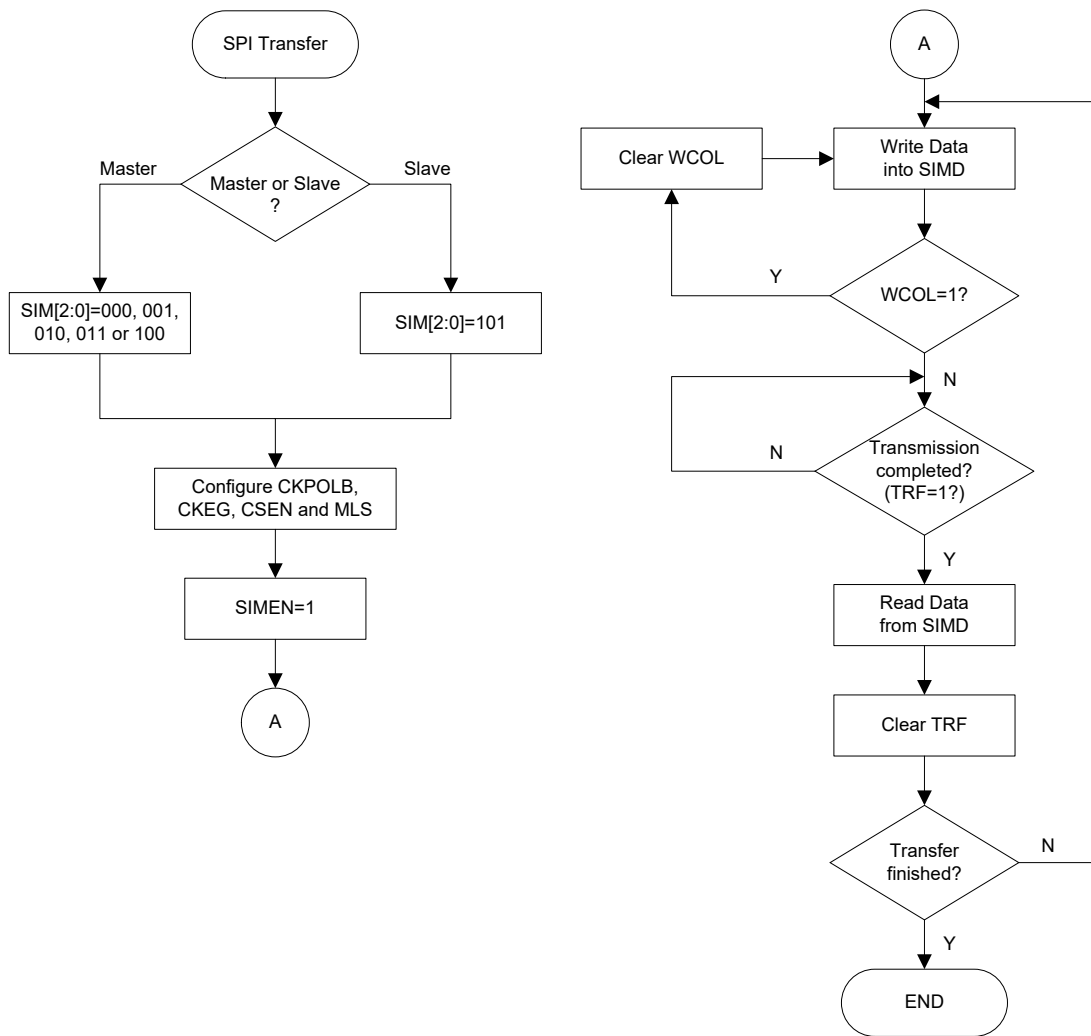
SPI 从机模式时序 – CKEG=0



Write to SIMD
(SDO changes as soon as writing occurs; SDO is floating if $\overline{SCS}=1$)

Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the SCS level.

SPI 从机模式时序 – CKEG=1



SPI 传输控制流程图

SPI 总线使能 / 除能

使能 SPI 总线，设置 CSEN=1 和 $\overline{\text{SCS}}=0$ ，然后等待数据写入 SIMD (TXRX 缓存器) 寄存器。对于主机模式，数据写入 SIMD (TXRX 缓存器) 寄存器后，将自动开始数据发送和接收操作。当所有数据都传输完成，TRF 位将置位。对于从机模式，当 SCK 线接收到时钟脉冲时，TXRX 缓存器内的数据将移出或 SDI 脚数据移入。

除能 SPI 总线，SCK，SDI，SDO 和 $\overline{\text{SCS}}$ 通过相应的引脚共用控制位设置为将为 I/O 口或其他功能。

SPI 操作

SPI 通过 4 线接口完成主机或从机通信。

SIMC2 寄存器内的 CSEN 位为 SPI 接口通信总的控制位。将该位设为“1”时， $\overline{\text{SCS}}$ 线有效，且用于控制 SPI 接口。如果 CSEN 位为低，SPI 接口将除能，且 $\overline{\text{SCS}}$ 线处于浮空状态，此时不能用于控制 SPI 接口。如果 SIMC0 寄存器的 CSEN 和 SIMEN 位为高，SDI 将浮空且 SDO 线为高。如果处于主机模式，SCK 线的高低取决于 SIMC2 寄存器的 CKPOLB 位设置的时钟的极性。如果处于从机模式，SCK 将浮空。如果 SIMEN 位为低，总线将除能且 $\overline{\text{SCS}}$ ，SDI，SDO 和 SCK 都为 I/O 引脚或其他功能。处于主机模式，时钟信号总是由主机生成。当数据写入 SIMD 寄存器后，主机完成所有的数据传输初始化并控制时钟信号。处于从机模式，时钟信号将来自于外部主机用于数据的接收和发送。下面介绍主 / 从模式数据传输步骤。

● 主机模式

- 步骤 1
设置 SIMC0 控制寄存器中的 SIM2~SIM0 位，选择时钟源和主机模式。
- 步骤 2
设置 CSEN 和 MLS 位，选择高位或低位数据优先传送，这必须与从机一致。
- 步骤 3
设置 SIMC0 控制寄存器中的 SIMEN 位，使能 SPI 接口功能。
- 步骤 4
对于写操作：写数据到 SIMD 寄存器，实际上，数据被存储在 TXRX 缓存器中。再使用 SCK 和 $\overline{\text{SCS}}$ 信号线将数据输出。跳至步骤 5。
对于读操作：使用 SDI 信号线将 TXRX 缓存器中的数据移出，并全部锁存至 SIMD 寄存器内。
- 步骤 5
检查 WCOL 位，若此位为高，则发生数据冲突并跳回至步骤 4；若为低，则继续执行下面的步骤。
- 步骤 6
检查 TRF 位或等待 SPI 串行总线中断发生。
- 步骤 7
从 SIMD 寄存器读数据。
- 步骤 8
清除 TRF。
- 步骤 9
跳回至步骤 4。

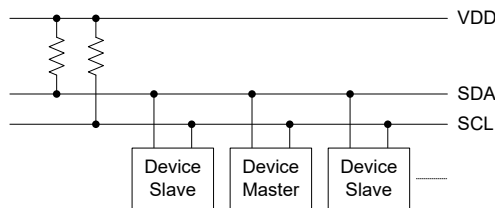
- 从机模式
 - 步骤 1
设置 SIMC0 控制寄存器中的 SIM2~SIM0 位，选择从机模式。
 - 步骤 2
设置 CSEN 和 MLS 位，选择高位或低位数据优先传送，者必须与主机一致。
 - 步骤 3
设置 SIMC0 控制寄存器中的 SIMEN 位，使能 SPI 接口功能。
 - 步骤 4
对于写操作：写数据到 SIMD 寄存器，实际上，数据被存储在 TXRX 缓存器中。等待主机时钟信号 SCK 和 \overline{SCS} 信号。跳至步骤 5。
对于读操作：使用 SDI 信号线将 TXRX 缓存器中的数据移出，并全部锁存至 SIMD 寄存器内。
 - 步骤 5
检查 WCOL 位，若此位为高，则发生数据冲突并跳回至步骤 4；若为低，则继续执行下面的步骤。
 - 步骤 6
检查 TRF 位或等待 SPI 串行总线中断发生。
 - 步骤 7
从 SIMD 寄存器读数据。
 - 步骤 8
清除 TRF。
 - 步骤 9
跳回至步骤 4。

错误检测

SIMC2 寄存器中的 WCOL 位用来数据传输期间监测数据冲突的发生。此位由 SPI 串行接口设置为高但必须由应用程序清零。在数据传输期间，如果写数据到 SIMD 寄存器，此时数据冲突发生且不允许数据继续被写入。

I²C 接口

I²C 可以和传感器，EEPROM 内存等外部硬件接口进行通信。最初是由飞利浦公司研制，是适用于同步串行数据传输的双线式低速串行接口。I²C 接口具有两线通信，非常简单的通信协议和在同一总线上和多个设备进行通信的能力的优点，使之在很多的场合中大受欢迎。

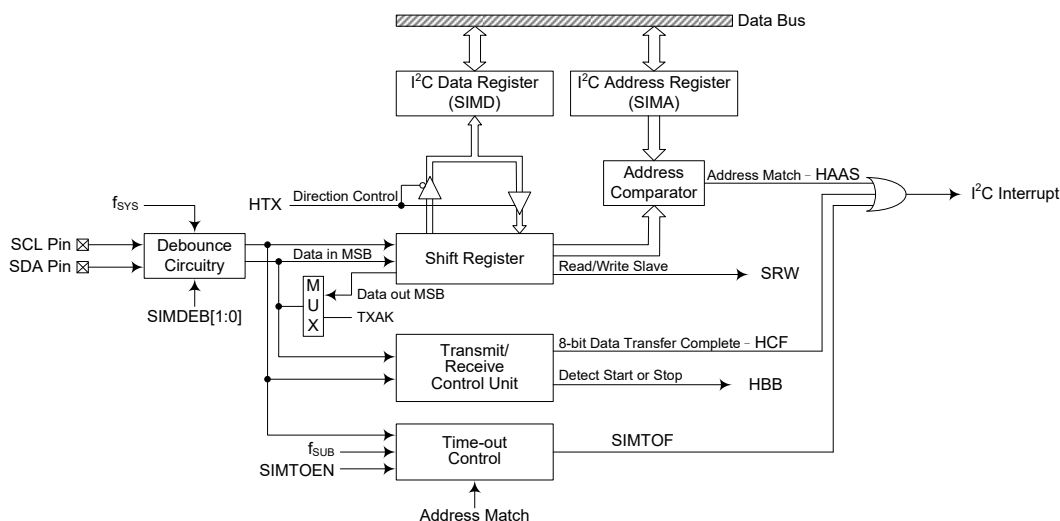


I²C 主从总线连接图

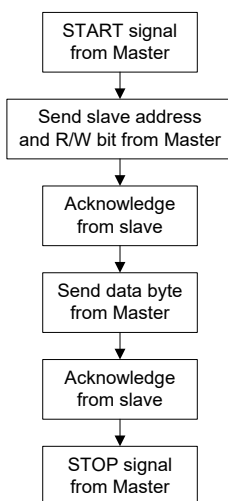
I²C 接口操作

I²C 串行接口是一个双线的接口，有一条串行数据线 SDA 和一条串行时钟线 SCL。由于可能有多个设备在同一条总线上相互连接，所以这些设备的输出都是开漏型输出。因此应在这些输出上都应加上拉电阻。应注意的是：I²C 总线上的每个设备都没有选择线，但分别与唯一的地址匹配，用于 I²C 通信。

如果有两个设备通过双向的 I²C 总线进行通信，那么就存在一个主机和一个从机。主机和从机都可以用于传输和接收数据，但只有主机才可以控制总线动作。那些处于从机模式的设备，要在 I²C 总线上传输数据只有两种方式，一是从机发送模式，二是从机接收模式。



I²C 方框图



SIMDEB1 和 SIMDEB0 位决定 I²C 接口的去抖时间。这个功能可以使用内部时钟在外部时钟上增加一个去抖间隔，会减小时钟线上毛刺发生的可能性，以避免单片机发生错误动作。如果选择了这个功能，去抖时间可以选择 2 个或 4 个系统时钟。为了达到需要的 I²C 数据传输速度，系统时钟 f_{SYS} 和 I²C 去抖时间之间存在一定的关系。I²C 标准模式或快速模式下，用户需注意所选的系统时钟频率与标准匹配去抖时间的设置，其具体关系如下表所示。

I ² C 去抖时间选择	I ² C 标准模式 (100kHz)	I ² C 快速模式 (400kHz)
无去抖时间	$f_{SYS} > 2 \text{ MHz}$	$f_{SYS} > 5 \text{ MHz}$
2 个系统时钟去抖时间	$f_{SYS} > 4 \text{ MHz}$	$f_{SYS} > 10 \text{ MHz}$
4 个系统时钟去抖时间	$f_{SYS} > 8 \text{ MHz}$	$f_{SYS} > 20 \text{ MHz}$

I²C 最小 f_{SYS} 频率要求

I²C 寄存器

I²C 总线的三个控制寄存器是 SIMC0, SIMC1 和 SIMTOC，一个地址寄存器 SIMA 及一个数据寄存器 SIMD。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	A6	A5	A4	A3	A2	A1	A0	D0
SIMTOC	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0

I²C 寄存器列表

I²C 数据寄存器

SIMD 用于存储发送和接收的数据。这个寄存器由 SPI 和 I²C 功能所共用。在单片机尚未将数据写入到 I²C 总线中时，要传输的数据应存在 SIMD 中。I²C 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 SPI 传输或接收的数据都必须通过 SIMD 实现。

• SIMD 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	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 D7~D0: SIM 数据寄存器 bit 7~bit 0

I²C 地址寄存器

SIMA 寄存器也在 SPI 接口功能中使用，但其名称改为 SIMC2。SIMA 寄存器用于存放 7 位从机地址，寄存器 SIMA 中的第 7~1 位是单片机的从机地址，位 0 未定义。如果接至 I²C 的主机发送处的地址和寄存器 SIMA 中存储的地址相符，那么就选中了这个从机。应注意的是寄存器 SIMA 和 SPI 接口使用的寄存器 SIMC2 是同一个寄存器。

• SIMA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	A6	A5	A4	A3	A2	A1	A0	—
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 7~1 **A6~A0**: I²C 从机地址位
A6~A0 是从机地址对应的 bit 6~bit 0。

Bit 0 未定义，读为“0”

I²C 控制寄存器

I²C 总线有三个控制寄存器 SIMC0、SIMC1 和 SIMTOC。SIMC0 寄存器用于控制使能 / 除能功能以及设置数据传输时钟频率。SIMC1 寄存器包含用于显示 I²C 通信状态的相关标志位。另一个寄存器 SIMTOC 用于控制 I²C 超时功能，在相关章节中进行描述。

• SIMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5 **SIM2~SIM0**: SIM 工作模式控制位
 000: SPI 主机模式; SPI 时钟为 $f_{SYS}/4$
 001: SPI 主机模式; SPI 时钟为 $f_{SYS}/16$
 010: SPI 主机模式; SPI 时钟为 $f_{SYS}/64$
 011: SPI 主机模式; SPI 时钟为 f_{SUB}
 100: SPI 主机模式; SPI 时钟为 PTM0 CCRP 匹配频率 / 2
 101: SPI 从机模式
 110: I²C 从机模式
 111: 未使用

这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 PTM0 和 f_{SUB} 。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 未定义，读为“0”

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C 去抖时间选择位

00: 未定义
 01: 2 个系统时钟去抖时间
 1x: 4 个系统时钟去抖时间

通过设置 SIM2 ~SIM0 位为“110”，SIM 被配置为 I²C 接口功能。这些位用于选择 I²C 去抖时间。

- Bit 1 **SIMEN**: SIM 使能控制位
0: 除能
1: 使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时, SIM 接口除能, SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I²C 功能, SIM 工作电流减小到最小值。此位为“1”时, SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口, 当 SIMEN 位由低到高转变时, SPI 控制寄存器中的设置不会发生变化, 其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口, 当 SIMEN 位由低到高转变时, I²C 控制寄存器中的设置, 如 HTX 和 TXAK, 将不会发生变化, 其首先应在应用程序中初始化, 此时相关 I²C 标志, 如 HCF、HAAS、HBB、SRW 和 RXAK, 将被设置为其默认状态。
- Bit 0 **SIMICF**: SIM 未完成标志位
此位在仅用于 SPI 模式中, 该位在 SPI 寄存器章节有描述。

● SIMC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

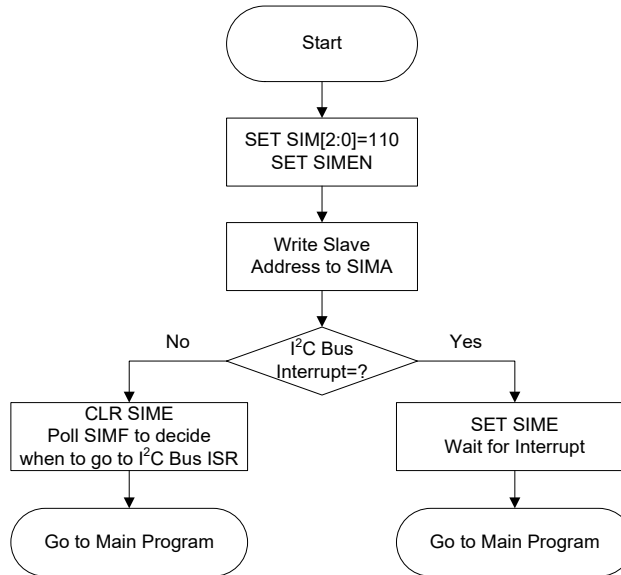
- Bit 7 **HCF**: I²C 总线数据传输完成标志位
0: 数据正在被传输
1: 8 位数据传输完成
HCF 是数据传输标志位。数据正在传输时该位为低。当 8 位数据传输完成, 此位为高并产生一个中断。
- Bit 6 **HAAS**: I²C 总线地址匹配标志位
0: 地址不匹配
1: 地址匹配
HAAS 标志位为地址匹配标志位。此标志位用于决定从机地址是否与主机发送地址相同。若地址匹配此位为高, 否则此位为低。
- Bit 5 **HBB**: I²C 总线忙标志位
0: I²C 总线闲
1: I²C 总线忙
HBB 标志位为 I²C 忙标志位。当检测到 START 信号时 I²C 忙, 此位变为高电平。当检测到 STOP 信号时 I²C 总线停止, 该位变为低电平。
- Bit 4 **HTX**: I²C 从机处于发送或接收模式标志位
0: 从机处于接收模式
1: 从机处于发送模式
- Bit 3 **TXAK**: I²C 总线发送确认标志位
0: 从机发送确认标志
1: 从机没有发送确认标志
TXAK 位为发送确认标志位。从机接收 8 位数据之后会将该位在第九个时钟传到总线上。如果从机想要接收更多的数据, 则应在接收数据之前将此位设置为“0”。
- Bit 2 **SRW**: I²C 从机读 / 写标志位
0: 从机应处于接收模式
1: 从机应处于发送模式
SRW 位是 I²C 从机读写标志位。决定主机是否希望传输或接收来自 I²C 总线的的数据。当传输地址和从机的地址相匹配时, HAAS 位会被设置为高, 从机将检测 SRW 位来决定进入发送模式还是接收模式。如果 SRW 位为高时, 主机将请求从总线上读数据, 此时从机处于传输模式。当 SRW 位为“0”时, 主机往总线上写数据, 从机处于接收模式以读取该数据。

Bit 1	IAMWU: I ² C 地址匹配唤醒控制位 0: 除能 1: 使能 此位应设置为“1”使能 I ² C 地址匹配以使系统从休眠或空闲模式中唤醒。若进入休眠或空闲模式前 IAMWU 已经设置以使能 I ² C 地址匹配唤醒功能，在系统唤醒后须软件清除此位以确保单片机正确地运行。
Bit 0	RXAK: I ² C 总线接收确认标志位 0: 从机接收到确认标志 1: 从机没有接收到确认标志 RXAK 位是接收确认标志位。如果 RXAK 位被设为“0”即 8 位数据传输之后，从机在第九个时钟有接收到一个正确的确认位。如果从机处于发送状态，则从机将会检查 RXAK 位来判断主机接收方是否愿意继续接收下一个字节。因此直到 RXAK 为“1”时，从机传输方停止发送数据。这时，传输方将释放 SDA 线，主机发出停止信号释放 I ² C 总线。

I²C 总线通信

I²C 总线上的通信需要四步完成，一个起始信号，一个从机地址发送，一个数据传输，还有一个停止信号。当起始信号被写入 I²C 总线时，总线上的所有从机都会接收到这个起始信号并且被通知总线上会即将有数据到达。数据的前 7 位是从机地址，高位在前，低位在后。如果发出的地址和从机地址匹配，SIMC1 寄存器的 HAAS 位会被置位，同时产生 I²C 中断。进入中断服务程序后，系统要检测 HAAS 和 SIMTOF 位，以判断 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传递完毕，还是来自 I²C 超时发生。在数据传递中，注意的是，在 7 位从机地址被发送后，接下来的一位，即第 8 位，是读 / 写控制位，该位的值会反映到 SRW 位中。从机通过检测 SRW 位以确定主控制器是要进入发送模式还是接收模式。在 I²C 总线开始传送数据前，需要先初始化 I²C 总线，初始化 I²C 总线步骤如下：

- 步骤 1
设置 SIMC0 寄存器中 SIM2~SIM0 为“110”和 SIMEN 位为“1”，以使能 I²C 总线。
- 步骤 2
向 I²C 总线地址寄存器 SIMA 写入从机地址。
- 步骤 3
设置 SIME 位中断使能位，以使能 SIM 中断。



I²C 总线初始化流程图

I²C 总线起始信号

起始信号只能由连接 I²C 总线主机产生，而不是由从机产生。总线上的所有从机都可以侦测到起始信号。如果有从机侦测到起始信号，则表明 I²C 总线处于忙碌状态，并会置位 HBB。起始信号是指在 SCL 为高电平时，SDA 线上发生从高到低的电平变化。

I²C 从机地址

总线上的所有从机都会侦测由主机发出的起始信号。发送起始信号后，紧接着主机将发送从机地址以选择要进行数据传输的从机。所有在 I²C 总线上的从机接收到 7 位地址数据后，都会将其与各自内部的地址进行比较。如果从机从主机上接收到的地址与自身内部的地址相匹配，则从机会产生一个 I²C 总线中断信号。地址位接下来的一位为读 / 写状态位（即第 8 位），将被保存到 SIMC1 寄存器的 SRW 位，随后发出一个低电平应答信号（即第 9 位）。当单片机从机的地址匹配时，会将状态标志位 HAAS 置位。

I²C 总线有三个中断源，当程序运行至中断服务子程序时，通过检测 HAAS 和 SIMTOF 位以确定 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传输完毕，还是来自 I²C 超时发生。当是从机地址匹配发生中断时，则从机或是用于发送模式并将数据写进 SIMD 寄存器，或是用于接收模式并从 SIMD 寄存器中无效读取以释放 SCL 线。

I²C 总线读 / 写信号

SIMC1 寄存器的 SRW 位用来定义主机是要从 I²C 总线上读取数据还是要将数据写到 I²C 总线上。从机则通过检测该位以确定自己是作为发送方还是接收方。当 SRW 置“1”，表示主机要从 I²C 总线上读取数据，从机则作为发送方，将数据写到 I²C 总线；当 SRW 清“0”，表示主机要写数据到 I²C 总线上，从机则做为接收方，从 I²C 总线上读取数据。

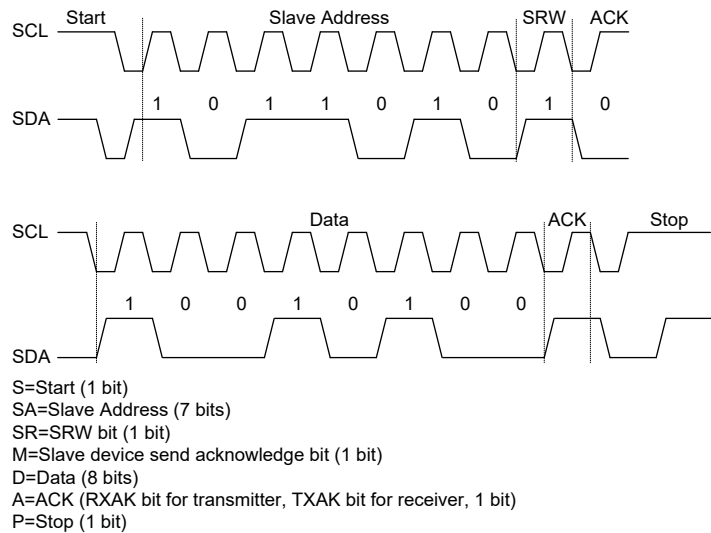
I²C 总线从机地址和确认信号

主机发送呼叫地址后，当 I²C 总线上的任何从机内部地址与其匹配时，会发送一个应答信号。此应答信号会通知主机有从机已经接收到了呼叫地址。如果主机没有收到应答信号，则主机必须发送停止 (STOP) 信号以结束通信。当 HAAS 为高时，表示从机接收到的地址与自己内部地址匹配，则从机需检查 SRW 位，以确定自己是作为发送方还是作为接收方。如果 SRW 位为高，从机须设置成发送方，这样会置位 SIMC1 寄存器的 HTX 位。如果 SRW 位为低，从机须设置成接收方，这样会清零 SIMC1 寄存器的 HTX 位。

I²C 总线数据和确认信号

在从机确认接收到地址后，会进行 8 位宽度的数据传输。这个数据传输顺序是的高位在前，低位在后。接收方在接收到 8 位数据后必须发出一个应答信号 (“0”) 以继续接收下一个数据。如果从机发送方没接收到主机接收方的应答信号，从机发送方将释放 SDA 线，同时，主机将发出 STOP 信号以释放 I²C 总线。所传送的数据存储在 SIMD 寄存器中。如果设置成发送方，从机必须先将欲传输的数据写到 SIMD 寄存器中；如果设置成接收方，从机必须从 SIMD 寄存器读取数据。

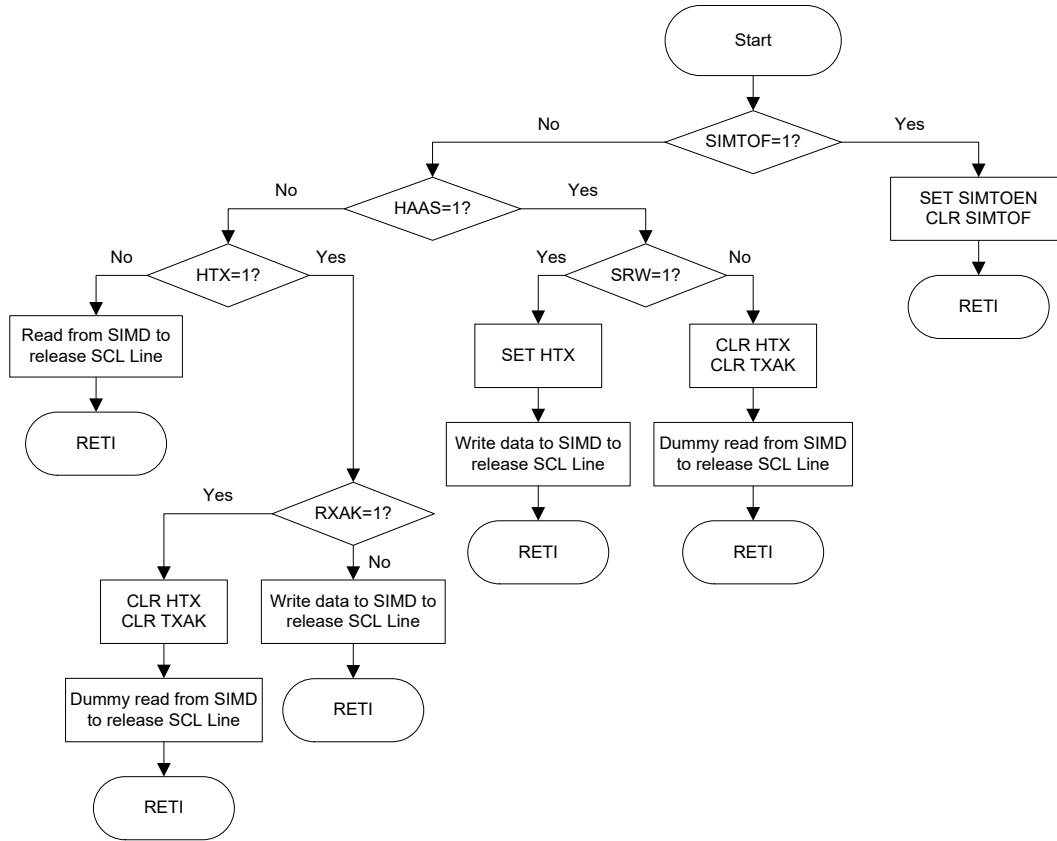
当从机接收方想要继续接收下一个数据时，必须在第 9 个时钟发出应答信号 (TXAK)。被设为发送方的从机将检测寄存器 SIMC1 中的 RXAK 位以判断是否传输下一个字节的数据，如果单片机不传输下一个字节，那么它将释放 SDA 线并等待接收主机的停止信号。



S	SA	SR	M	D	A	D	A	S	SA	SR	M	D	A	D	A	P
---	----	----	---	---	---	---	---	-------	---	----	----	---	---	---	---	---	-------	---

I²C 通信时序图

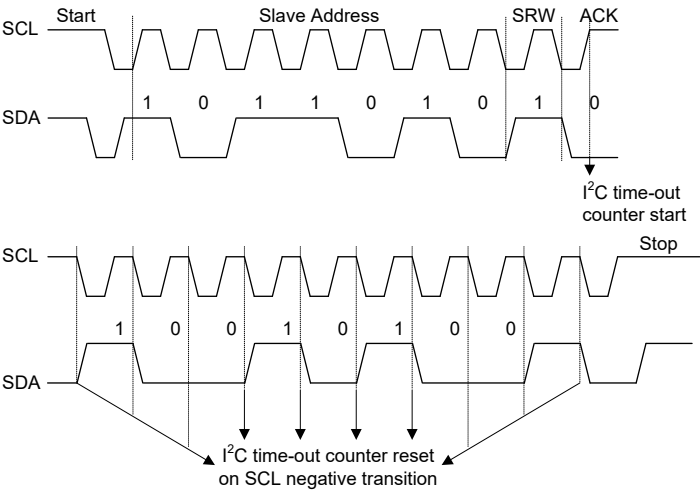
注：* 当从机地址匹配时，从机必须选择设置为发送模式还是接收模式。若设置为发送模式，需写数据至 SIMD 寄存器；若设置为接收模式，需立即从 SIMD 寄存器中无效读数据以释放 SCL 线。



I²C 总线 ISR 流程图

I²C 超时控制

超时功能可减少 I²C 接收错误的时钟源而引起的锁死问题。如果连接到 I²C 总线的时钟源经过一段时间还未接收到，则在一定的超时周期后，I²C 电路和寄存器将复位。超时计数器在 I²C 总线“START”和“地址匹配”条件下开始计数，在 SCL 下降沿清零。在下一个 SCL 下降沿到来之前，如果超时时间大于 SIMTOC 寄存器指定的超时周期，则超时发生。I²C “STOP”条件发生时超时功能停止。



I²C 超时

当 I²C 超时计数器溢出时，计数器将停止计数，SIMTOEN 位被清零，且 SIMTOF 位被置高以表明超时计数器中断发生。超时中断使用的也是 I²C 中断向量。当 I²C 超时发生时，I²C 内部电路会被复位，寄存器也将发生如下复位情况。

寄存器	I ² C 超时发生后
SIMD, SIMA, SIMC0	无变化
SIMC1	复位到 POR 状态

超时后的 I²C 寄存器状态

SIMTOF 标志位由应用程序清零。共有 64 个溢出周期，可通过 SIMTOC 寄存器的 SIMTOS 位段进行选择。溢出周期可通过公式计算： $((1\sim64)\times32)/f_{SUB}$ 。由此可得溢出周期范围为 1ms~64ms。

● SIMTOC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0
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 **SIMTOEN**: SIM I²C 超时控制位
0: 除能
1: 使能

Bit 6 **SIMTOF**: SIM I²C 超时标志位
0: 超时未发生
1: 超时发生

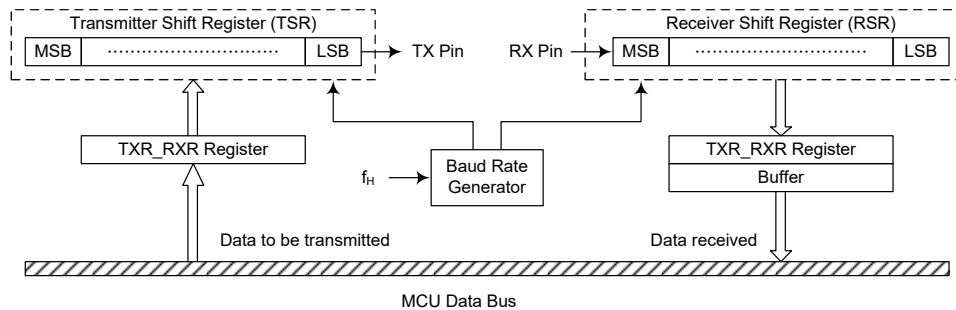
Bit 5~0 **SIMTOS5~SIMTOS0**: SIM I²C 超时时间选择位
I²C 超时时钟源是 $f_{SUB}/32$ 。
I²C 超时时间计算公式: $(SIMTOS[5:0]+1)\times(32/f_{SUB})$ 。

UART 串行接口 – HT66F4540/HT66F4550/HT66F4560

该系列单片机具有一个全双工的异步串行通信接口——UART，可以很方便的与其它具有串行口的芯片通信。UART 具有许多功能特性，发送或接收串行数据时，将数据组成一个 8 位或 9 位的数据帧，连同数据特征位一并传输。具有检测数据覆盖或帧错误等功能。UART 功能占用一个内部中断向量，当接收到数据或数据发送结束，触发 UART 中断。

内置的 UART 功能包含以下特性：

- 全双工通用异步接收器 / 发送器
- 8 位或 9 位传输格式
- 奇校验、偶校验或无校验
- 1 位或 2 位停止位
- 8 位预分频的波特率发生器
- 奇偶、帧、噪声和溢出检测
- 支持地址匹配中断（最后一位 =1）
- 独立的发送和接收使能
- 2-byte Deep FIFO 接收缓冲器
- RX 引脚唤醒功能
- 发送和接收中断
- 中断可由下列条件初始化：
 - ◆ 发送器为空
 - ◆ 发送器空闲
 - ◆ 接收完成
 - ◆ 接收器溢出
 - ◆ 地址模式检测
 - ◆ RX 引脚唤醒功能



UART 数据传输方框图

UART 外部引脚

内部 UART 有两个外部引脚 TX 和 RX，可与外部串行接口进行通信。TX 和 RX 分别为 UART 发送脚和接收脚。在使用 UART 功能前，应先通过相应的引脚共用功能选择寄存器，选择 TX 和 RX 引脚功能。当 UARTEN 和 TXEN/RXEN 位置高时，将自动设置这些 I/O 脚或其它共用功能脚作为 TX 输出和 RX 输入，并且除能 TX 和 RX 引脚上的上拉电阻功能。当 UARTEN、TXEN 或 RXEN 位清零除能 TX 或 RX 引脚功能后，TX 或 RX 引脚将处于浮空状态。这时 TX 或 RX 引脚是否连接内部上拉电阻是由相应的 I/O 上拉电阻控制位决定的。

UART 数据传输方案

前面方框图显示了 UART 的整体结构。需要发送的数据首先写入 TXR_RXR 寄存器，接着此数据被传输到发送移位寄存器 TSR 中，然后在波特率发生器的控制下将 TSR 寄存器中数据一位位地移到 TX 引脚上，低位在前。TXR_RXR 寄存器被映射到单片机的数据存储器中，而发送移位寄存器没有实际地址，所以发送移位寄存器不可直接操作。

数据在波特率发生器的控制下，低位在前高位在后，从外部引脚 RX 进入接收移位寄存器 RSR。当数据接收完成，数据从接收移位寄存器移入可被用户程序操作的 TXR_RXR 寄存器中。TXR_RXR 寄存器被映射到单片机数据存储器中，而接收移位寄存器没有实际地址，所以接收移位寄存器不可直接操作。

需要注意的是，发送和接收都是共用同一个数据存储器地址的数据寄存器，即 TXR_RXR 寄存器。

UART 状态和控制寄存器

与 UART 功能相关的有五个寄存器——控制 UART 模块整体功能的 USR、UCR1 和 UCR2 寄存器，控制波特率的 BRG 寄存器，管理发送和接收数据的数据寄存器 TXR_RXR。

寄存器名称	位							
	7	6	5	4	3	2	1	0
USR	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
UCR1	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
UCR2	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	THIE	TEIE
TXR_RXR	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
BRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0

UART 寄存器列表

● USR 寄存器

寄存器 USR 是 UART 的状态寄存器，可以通过程序读取以确定 UART 的当前状态。所有 USR 位是只读的。详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

- Bit 7

PERR: 奇偶校验出错标志位

0: 奇偶校验正确

1: 奇偶校验出错

PERR 是奇偶校验出错标志位。若只读标志位 PERR=0，奇偶校验正确；若 PERR=1，接收到的数据奇偶校验出错。只有使能了奇偶校验（奇校验或偶校验）此位才有效。可使用软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器来清除此位。
- Bit 6

NF: 噪声干扰标志位

0: 没有受到噪声干扰

1: 受到噪声干扰

NF 是噪声干扰标志位。若只读标志位 NF=0，没有受到噪声干扰；若 NF=1，UART 接收数据时受到噪声干扰。它与 RXIF 在同周期内置位，但不会与溢出标志位同时置位。可使用软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器将清除此标志位。

Bit 5	<p>FERR: 帧错误标志位</p> <p>0: 无帧错误发生</p> <p>1: 有帧错误发生</p> <p>FREE 是帧错误标志位。若只读标志位 FREE=0, 没有帧错误发生; 若 FREE=1, 当前的数据发生了帧错误。可使用软件清除该标志位, 即先读取 USR 寄存器再读 TXR_RXR 寄存器来清除此位。</p>
Bit 4	<p>OERR: 溢出错误标志位</p> <p>0: 无溢出错误发生</p> <p>1: 有溢出错误发生</p> <p>OERR 是溢出错误标志位, 表示接收缓冲器是否溢出。若只读标志位 OERR=0, 没有溢出错误; 若 OERR=1, 发生了溢出错误, 它将禁止下一组数据的接收。可通过软件清除该标志位, 即先读取 USR 寄存器再读 TXR_RXR 寄存器将清除此标志位。</p>
Bit 3	<p>RIDLE: 接收状态标志位</p> <p>0: 正在接收数据</p> <p>1: 接收器空闲</p> <p>RIDLE 是接收状态标志位。若只读标志位 RIDLE=0, 正在接收数据; 若 RIDLE=1, 接收器空闲。在接收到停止位和下一个数据的起始位之间, RIDLE 被置位, 表明 UART 空闲, RX 脚处于逻辑高状态。</p>
Bit 2	<p>RXIF: 接收器 TXR_RXR 寄存器状态标志位</p> <p>0: TXR_RXR 寄存器为空</p> <p>1: TXR_RXR 寄存器含有有效数据</p> <p>RXIF 是接收寄存器状态标志位。当只读标志位 RXIF=0, TXR_RXR 寄存器为空; 当 RXIF=1, TXR_RXR 寄存器接收到新数据。当数据从移位寄存器加载到 TXR_RXR 寄存器中, 如果 UCR2 寄存器中的 RIE=1, 则会触发中断。当接收数据时检测到一个或多个错误时, 相应的标志位 NF、FERR 或 PERR 会在同一周期内置位。读取 USR 寄存器再读 TXR_RXR 寄存器, 如果 TXR_RXR 寄存器中没有新的数据, 那么将清除 RXIF 标志。</p>
Bit 1	<p>TIDLE: 数据发送完成标志位</p> <p>0: 数据传输中</p> <p>1: 无数据传输</p> <p>TIDLE 是数据发送完成标志位。若只读标志位 TIDLE=0, 数据传输中。当 TXIF=1 且数据发送完毕或者暂停字被发送时, TIDLE 置位。TIDLE=1, TX 引脚空闲且处于逻辑高状态。读取 USR 寄存器再写 TXR_RXR 寄存器将清除 TIDLE 位。数据字符或暂停字就绪时, 不会产生该标志位。</p>
Bit 0	<p>TXIF: 发送数据寄存器 TXR_RXR 状态位</p> <p>0: 数据还没有从缓冲器加载到移位寄存器中</p> <p>1: 数据已从缓冲器加载到移位寄存器中 (TXR_RXR 数据寄存器为空)</p> <p>TXIF 是发送数据寄存器为空标志位。若只读标志位 TXIF=0, 数据还没有从缓冲器加载到移位寄存器中; 若 TXIF=1, 数据已从缓冲器中加载到移位寄存器中。读取 USR 寄存器再写 TXR_RXR 寄存器将清除 TXIF。当 TXEN 被置位, 由于发送缓冲器未满, TXIF 也会被置位。</p>

● UCR1 寄存器

UCR1 和 UCR2 是 UART 的两个控制寄存器，用来定义各种 UART 功能，例如 UART 的使能与除能、奇偶校验控制和传输数据的长度等等。详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	W
POR	0	0	0	0	0	0	x	0

“x”：未知

Bit 7 **UARTEN**: UART 功能使能位

0: UART 除能, TX 和 RX 脚作为 I/O 或其它引脚共用功能

1: UART 使能, TX 和 RX 脚作为 UART 功能引脚

此位为 UART 的使能位。UARTEN=0, UART 除能, UART 的 RX 和 TX 引脚除能作为通用 I/O 或其它引脚共用功能; UARTEN=1, UART 使能, TX 和 RX 将分别由 TXEN 和 RXEN 控制。

当 UART 被除能将清除缓冲器, 所有缓冲器中的数据将被忽略, 另外波特率计数器、错误和状态标志位被复位, TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零, 而 TIDLE、TXIF 和 RIDLE 置位, UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零, 所有发送和接收将停止, 模块也将复位成上述状态。当 UART 再次使能时, 它将在上次配置下重新工作。

Bit 6 **BNO**: 发送数据位数选择位

0: 8-bit 传输数据

1: 9-bit 传输数据

BNO 是发送数据位数选择位。BNO=1, 传输数据为 9 位; BNO=0, 传输数据为 8 位。若选择了 9 位数据传输格式, RX8 和 TX8 将分别存储接收和发送数据的第 9 位。

Bit 5 **PREN**: 奇偶校验使能位

0: 奇偶校验除能

1: 奇偶校验使能

此位为奇偶校验使能位。PREN=1, 使能奇偶校验; PREN=0, 除能奇偶校验。最高有效位置换成一个奇偶校验位。

Bit 4 **PRT**: 奇偶校验选择位

0: 偶校验

1: 奇校验

奇偶校验选择位。PRT=1, 奇校验; PRT=0, 偶校验。

Bit 3 **STOPS**: 停止位的长度选择位

0: 有一位停止位

1: 有两位停止位

此位用来设置停止位的长度。STOP=1, 有两位停止位; STOP=0, 只有一位停止位。

Bit 2 **TXBRK**: 暂停字发送控制位

0: 没有暂停字要发送

1: 发送暂停字

TXBRK 是暂停字发送控制位。TXBRK=0, 没有暂停字要发送, TX 引脚正常操作; TXBRK=1, 将会发送暂停字, 发送器将发送逻辑“0”。若 TXBRK 为高, 缓冲器中数据发送完毕后, 发送器输出将至少保持 13 位宽的低电平直至 TXBRK 复位。

Bit 1 **RX8**: 接收 9-bit 数据传输格式中的 bit 8(只读)

此位只有在传输数据为 9 位的格式中有效, RX8 用来存储接收数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

Bit 0 **TX8**: 发送 9-bit 数据传输格式中的 bit 8 (只写)
此位只有在传输数据为 9 位的格式中有效, TX8 用来存储发送数据的第 9 位。
BNO 是用来控制传输位数是 8 位还是 9 位。

● UCR2 寄存器

UCR2 是 UART 的第二个控制寄存器, 它的主要功能是控制发送器、接收器以及各种 UART 中断源的使能或除能。它也可用来控制波特率, 使能接收唤醒和地址检测。详细解释如下:

Bit	7	6	5	4	3	2	1	0
Name	TXEN	RXEN	BRGH	ADDEN	WAKE	R1E	TI1E	TE1E
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 **TXEN**: UART 发送使能位
0: UART 发送除能
1: UART 发送使能
此位为发送使能位。TXEN=0, 发送将被除能, 发送器立刻停止工作。另外发送缓冲器将被复位, 此时 TX 引脚将作为 I/O 或其它引脚共用功能。
若 TXEN=1 且 UARTEN=1, 则发送将被使能, TX 引脚将由 UART 来控制。在数据传输时清除 TXEN 将中止数据发送且复位发送器, 此时 TX 引脚将作为 I/O 或其它引脚共用功能。

Bit 6 **RXEN**: UART 接收使能位
0: UART 接收除能
1: UART 接收使能
此位为接收使能位。RXEN=0, 接收将被除能, 接收器立刻停止工作。另外接收缓冲器将被复位, 此时 RX 引脚将作为 I/O 或其它引脚共用功能。若 RXEN=1 且 UARTEN=1, 则接收将被使能, RX 引脚将由 UART 来控制。在数据传输时清除 RXEN 将中止数据接收且复位接收器, 此时 RX 引脚将作为 I/O 或其它引脚共用功能。

Bit 5 **BRGH**: 波特率发生器高低速选择位
0: 低速波特率
1: 高速波特率
此位为波特率发生器高低速选择位, 它和 BRG 寄存器一起控制 UART 的波特率。BRGH=1, 为高速模式; BRGH=0, 为低速模式。

Bit 4 **ADDEN**: 地址检测使能位
0: 地址检测除能
1: 地址检测使能
此位为地址检测使能 / 除能位。ADDEN=1, 地址检测使能, 此时数据的第 8 位 (BNO=0) 或第 9 位 (BNO=1) 为高, 那么接到的是地址而非数据。若相应的中断使能且接收到的值最高位为 1, 那么中断请求标志将会被置位, 若地址检测功能使能且最高位为 0, 那么将不会产生中断且收到的数据也会被忽略。

Bit 3 **WAKE**: RX 脚下降沿唤醒 UART 功能使能位
0: RX 脚下降沿唤醒 UART 功能除能
1: RX 脚下降沿唤醒 UART 功能使能
此位用于控制 RX 引脚下降沿时是否唤醒 UART 功能。此位仅当 UART 时钟源 f_{H} 关闭时有效。若 UART 时钟源 f_{H} 还开启, 则无 RX 引脚唤醒 UART 功能无效。若此位置高且 UART 时钟 f_{H} 关闭, 当 RX 引脚发生下降沿时会产生 UART 唤醒请求。若相应的中断使能, 将产生 RX 引脚唤醒 UART 的中断, 以告知单片机使其通过应用程序开启 UART 时钟源 f_{H} , 从而唤醒 UART 功能。否则, 若此位为低, 即使 RX 引脚发生下降沿也无法恢复 UART 功能。

- Bit 2 **RIE**: 接收中断使能位
 0: 接收中断除能
 1: 接收中断使能
此位为接收中断使能或除能位。若 RIE=1, 当 OERR 或 RXIF 置位时, UART 的中断请求标志置位; 若 RIE=0, UART 中断请求标志不受 OERR 和 RXIF 影响。
- Bit 1 **TIIE**: 发送器为空中断使能位
 0: 发送器空闲中断除能
 1: 发送器空闲中断使能
此位为发送器空闲中断的使能或除能位。若 TIIE=1, 当发送器空闲触发 TIDLE 置位时, UART 的中断请求标志置位; 若 TIIE=0, UART 中断请求标志不受 TIDLE 的影响。
- Bit 0 **TEIE**: 发送寄存器为空中断使能位
 0: 发送寄存器为空中断除能
 1: 发送寄存器为空中断使能
此位为发送寄存器为空中断的使能或除能位。若 TEIE=1, 当发送器为空触发 TXIF 置位时, UART 的中断请求标志置位; 若 TEIE=0, UART 中断请求标志不受 TXIF 的影响。

● **TXR_RXR 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

“x”: 未知

Bit 7~0 **TXRX7~TXRX0**: UART 发送 / 接收数据位 Bit 7~Bit 0

● **BRG 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

“x”: 未知

Bit 7~0 **BRG7~BRG0**: 波特率值
通过应用程序设置 UCR2 寄存器的 BRGH 位 (设置波特率发生器的速度) 和 BRG 寄存器 (设置波特率的值), 一起控制 UART 的波特率。
注: 若 BRGH=0, 波特率 = $f_{\text{H}}/[64 \times (N+1)]$;
 若 BRGH=1, 波特率 = $f_{\text{H}}/[16 \times (N+1)]$ 。

波特率发生器

UART 自身具有一个波特率发生器，通过它可以设定数据传输速率。波特率是由一个独立的内部 8 位计数器产生，它由 BRG 寄存器和 UCR2 寄存器的 BRGH 位来控制。BRGH 是决定波特率发生器处于高速模式还是低速模式，从而决定计算公式的选用。BRG 寄存器的值 N 可根据下表中的公式计算，N 的范围是 0 到 255。

UCR2 的 BRGH 位	0	1
波特率 (BR)	$f_H / [64 (N+1)]$	$f_H / [16 (N+1)]$

为得到相应的波特率，首先需要设置 BRGH 来选择相应的计算公式从而算出 BRG 的值。应注意，由于 BRG 的值不连续，所以实际波特率和理论值之间有一个偏差。下面举例怎样计算 BRG 寄存器中的值 N 和误差。

波特率和误差的计算

若选用 4MHz 时钟频率且 BRGH=0，若期望的波特率为 4800，计算它的 BRG 寄存器的值 N，实际波特率和误差。

根据上表，波特率 $BR = f_H / [64 (N+1)]$

转换后的公式 $N = [f_H / (BR \times 64)] - 1$

带入参数 $N = [4000000 / (4800 \times 64)] - 1 = 12.0208$

取最接近的值，十进制 12 写入 BRG 寄存器，实际波特率如下

$BR = 4000000 / [64 \times (12+1)] = 4808$

因此，误差 = $(4808 - 4800) / 4800 = 0.16\%$

UART 模块的设置与控制

UART 采用标准的不归零码传输数据，这种方法通常被称为 NRZ 法。它由 1 位起始位，8 位或 9 位数据位和 1 位或者两位停止位组成。奇偶校验是由硬件自动完成的，可设置成奇校验、偶校验和无校验三种格式。常用的数据传输格式由 8 位数据位，1 位停止位，无校验组成，用 8、N、1 表示，它是系统上电的默认格式。数据位数、停止位数和奇偶校验由 UCR1 寄存器的 BNO、PRT、PREN 和 STOPS 设定。用于数据发送和接收的波特率由一个内部的 8 位波特率发送器产生，数据传输时低位在前高位在后。尽管 UART 发送器和接收器在功能上相互独立，但它们使用相同的数据传输格式和波特率，在任何情况下，停止位是必须的。

UART 的使能 / 除能

UART 是由 UCR1 寄存器的 UARTEN 位来使能 / 除能的。若 UARTEN、TXEN 和 RXEN 都为高，则 TX 和 RX 分别为 UART 的发送端口和接收端口。若没有数据发送，TX 引脚默认状态为高电平。

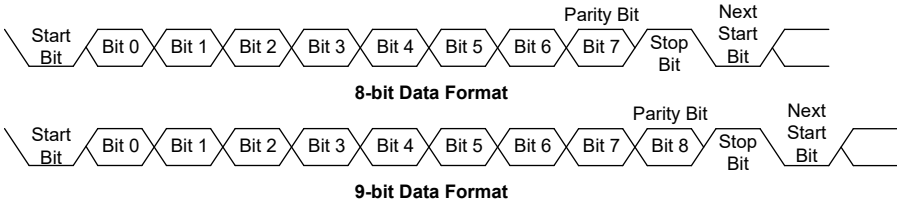
UARTEN 清零将除能 TX 和 RX，通过设置相关引脚共用控制位，这两个引脚可用作普通 I/O 口或其它引脚共用功能。当 UART 被除能时将清空缓冲器，所有缓冲器中的数据将被忽略，另外一些使能控制、错误标志和状态标志将被复位，如 TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零，而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

数据位、奇偶校验以及停止位位数的选择

数据传输格式由数据长度、是否校验、校验类型、地址位以及停止位长度组成。它们都是由 UCR1 寄存器的各个位控制的。BNO 决定数据传输是 8 位还是 9 位；PRT 决定校验类型；PREN 决定是否选择奇偶校验；而 STOPS 决定选用 1 位还是 2 位停止位。下表列出了各种数据传输格式。若地址检测功能使能，地址位，即数据字节的最高位，用来确定此帧是地址还是数据。停止位的长度和数据位的长度无关，且只有发送器需设置停止位长度。接收器只接收一个停止位。

起始位	数据位	地址位	校验位	停止位
8 位数据位				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
9 位数据位				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

下图是传输 8 位和 9 位数据的波形。



UART 发送器

UCR1 寄存器的 BNO 位是控制数据传输的长度。BNO=1 其长度为 9 位，第 9 位 MSB 存储在 UCR1 寄存器的 TX8 中。发送器的核心是发送移位寄存器 TSR，它的数据由发送寄存器 TXR_RXR 提供，应用程序只须将发送数据写入 TXR_RXR 寄存器。上组数据的停止位发出前，TSR 寄存器禁止写入。如果还有新的数据要发送，一旦停止位发出，待发数据将会从 TXR_RXR 寄存器加载到 TSR 寄存器。TSR 不像其它寄存器一样映射到数据存储器，所以应用程序不能对其进行读写操作。TXEN=1，发送使能，但若 TXR_RXR 寄存器没有数据或者波特率没有设置，发送器将不会工作。先写 TXR_RXR 寄存器再置高 TXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 TXR_RXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，TXEN 清零，发送器将立刻停止工作并且复位，此时通过设置相关引脚共用控制位，TX 引脚用作普通 I/O 口或其它引脚共用功能。

发送数据

当 UART 发送数据时，数据从移位寄存器中移到 TX 引脚上，其低位在前高位在后。在发送模式中，TXR_RXR 寄存器在内部总线和发送移位寄存器间形成一个缓冲。如果选择 9 位数据传输格式，最高位 MSB 取自 UCR1 寄存器的 TX8。发送器初始化可由如下步骤完成：

- 正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 BRG 寄存器，选择期望的波特率。
- 置高 TXEN，使能 UART 发送器且使 TX 作为 UART 的发送端。
- 读取 USR 寄存器，然后将待发数据写入 TXR_RXR 寄存器。注意，此步骤会清除 TXIF 标志位。

如果要发送多个数据只需重复上一步骤。

当 TXIF=0 时，数据将禁止写入 TXR_RXR 寄存器。可以通过以下步骤来清除 TXIF：

1. 读取 USR 寄存器
2. 写 TXR_RXR 寄存器

只读标志位 TXIF 由 UART 硬件置位。若 TXIF=1，TXR_RXR 寄存器为空，其它数据可以写入而不会覆盖之前的数据。若 TEIE=1，TXIF 标志位会产生中断。

在数据传输时，写 TXR_RXR 指令会将待发数据暂存在 TXR_RXR 寄存器中，当前数据发送完毕后，待发数据被加载到发送移位寄存器中。当发送器空闲时，写 TXR_RXR 指令会将数据直接加载到 TSR 寄存器中，数据传输立刻开始且 TXIF 置位。当发送完停止位或暂停帧后，表示一帧数据已发送完毕，此时 TIDLE 位将被置位。可以通过以下步骤来清除 TIDLE：

1. 读取 USR 寄存器
2. 写 TXR_RXR 寄存器

清除 TXIF 和 TIDLE 软件执行次序相同。

发送暂停字

若 TXBRK=1，下一帧将会发送暂停字。它是由一个起始位、13×N（N=1，2……）位逻辑 0 和暂停位组成。通过应用程序置位 TXBRK 将会发送暂停字，而清除 TXBRK 将产生停止位，传输暂停字不会产生中断。需要注意的是，暂停字至少 13 位宽。若 TXBRK 持续为高，那么发送器会一直发送暂停字；当应用程序将 TXBRK 清零后，发送器结束最后一帧暂停字的发送后接着发送一位或两位停止位。最后一帧暂停字的结尾自动为高电平，以确保下一帧数据起始位的检测。

UART 接收器

UART 接收器支持 8 位或者 9 位数据接收。若 BNO=1，数据长度为 9 位，而最高位 MSB 存放在 UCR1 寄存器的 RX8 中。接收器的核心是串行移位寄存器 RSR。RX 引脚上的数据送入数据恢复器中，它在 16 倍波特率的频率下工作，而串行移位器工作在正常波特率下。当在 RX 引脚上检测到停止位，若 TXR_RXR 寄存器为空，数据从 RSR 寄存器中加载到 TXR_RXR 寄存器。RX 引脚上的每一位数据会被采样三次以判断其逻辑状态。RSR 不像其它寄存器一样映射在数据存储区，所以应用程序不能对其进行读写操作。

接收数据

当 UART 接收数据时，数据低位在前高位在后，连续地从 RX 引脚进入移位寄存器。TXR_RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。TXR_RXR 寄存器是一个两层深度的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 TXR_RXR 寄存器，否则忽略第三帧数据并且发生溢出错误 OERR。接收器的初始化可由如下步骤完成：

- 正确地设置 BNO、PRT 和 PREN 位以确定数据长度和校验类型。
- 设置 BRG 寄存器，选择期望的波特率。
- 置高 RXEN，使能 UART 发送器且使 RX 作为 UART 的接收端。

此时接收器被使能并检测起始位。

接收数据将会发生如下事件：

- 当 TXR_RXR 寄存器中包含有效数据时，USR 寄存器中的 RXIF 位将会置位，溢出错误发生之前至多还有一帧数据可读。
- 若 RIE=1，数据从 RSR 寄存器加载到 TXR_RXR 寄存器中将产生中断。
- 若接收器检测到帧错误、噪声干扰错误、奇偶出错或溢出错误，那么相应的错误标志位置位。

可以通过如下步骤来清除 RXIF：

1. 读取 USR 寄存器
2. 读取 TXR_RXR 寄存器

接收暂停字

UART 接收任何暂停字都会当作帧错误处理。接收器只根据 BNO 位的设置外加一个停止位来确定一帧数据的长度。若暂停字位数远大于 13 位，接收完 BNO 位指定的长度外加一个停止位后，接收器认为接收已完毕，RXIF 和 FERR 置位，TXR_RXR 寄存器清 0，若相应的中断允许且 RIDLE 为高将会产生中断。暂停字只会被认为包含信息 0 且会置位 FERR 标志位。如果检测到较长的暂停信号，接收器会将此信号视为包含一个起始位、数据位和无效的停止位的数据帧并且置位 FERR 标志位。在下个开始位到来之前，接收器必须等待一个有效的停止位。接收器不会假定线上的暂停信号是下一个开始位。暂停字将会加载到缓冲器中，在接收到停止位前不会再接收数据，没有检测到停止位也会置位只读标志位 RIDLE。UART 接收到暂停字会产生以下事件：

- 帧错误标志位 FERR 置位。
- TXR_RXR 寄存器清零。
- OERR、NF、PERR、RIDLE 或 RXIF 可能会置位。

空闲状态

当 UART 接收数据时，即在起初位和停止位之间，USR 寄存器的接收状态标志位 RIDLE 清零。在停止位和下一帧数据的起始位之间，RIDLE 被置位，表示接收器空闲。

接收中断

USR 寄存器的只读标志位 RXIF 由接收器的边沿触发置位。若 RIE=1，数据从移位寄存器 RSR 加载到 TXR_RXR 寄存器时产生中断，同样地，溢出也会产生中断。

接收错误处理

UART 会产生几种接收错误，下面部分将描述各错误以及怎样处理。

溢出错误 – OERR 标志

TXR_RXR 寄存器是一个两层深度的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 TXR_RXR 寄存器，否则发生溢出错误。

产生溢出错误时将会发生以下事件：

- USR 寄存器中 OERR 被置位。
- TXR_RXR 寄存器中数据不会丢失。
- RSR 寄存器数据将会被覆盖。
- 若 RIE=1，将会产生中断。

先读取 USR 寄存器再读取 TXR_RXR 寄存器可将 OERR 清零。

噪声干扰 – NF 标志

数据恢复时多次采样可以有效的鉴别出噪声干扰。当检测到数据受到噪声干扰时将会发生以下事件：

- 在 RXIF 上升沿，USR 寄存器中只读标志位 NF 置位。
- 数据从 RSR 寄存器加载到 TXR_RXR 寄存器中。
- 不产生中断，但此位置位发生在 RXIF 置位产生中断的同周期内。

应注意，先读取 USR 寄存器再读取 TXR_RXR 寄存器可将 NF 清零。

帧错误 – FERR 标志

若在停止位上检测到 0，USR 寄存器中只读标志 FERR 置位。若选择两位停止位，此两位都必须为高，否则将置位 FERR。此标志位同接收的数据分别记录在 USR 寄存器和 TXR_RXR 寄存器中，此标志位可被任何复位清零。

奇偶校验错误 – PERR 标志

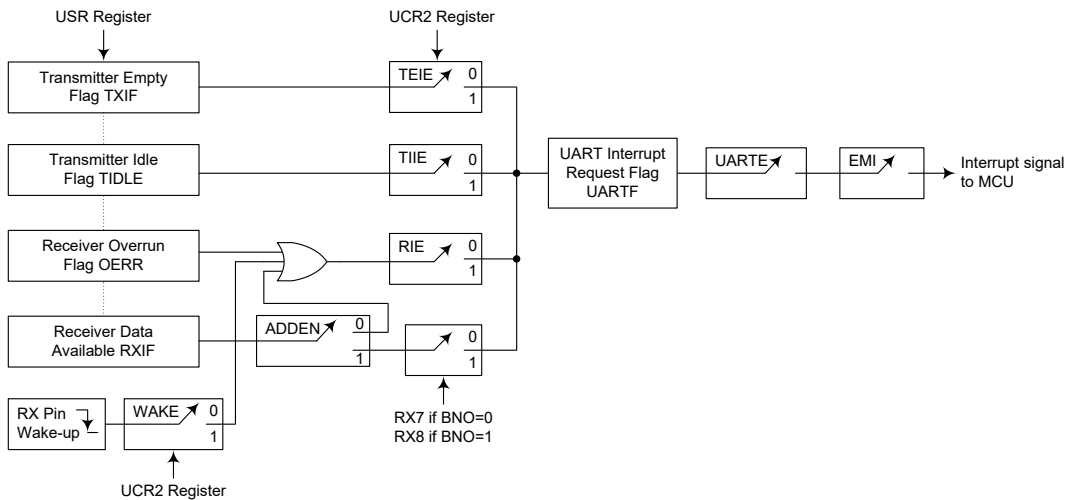
若接收到数据出现奇偶校验错误，USR 寄存器中只读标志 PERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。此标志位同接收的数据分别记录在 USR 寄存器和 TXR_RXR 寄存器中，此标志位可被任何复位清零。注意，在读取相应的数据之前必须先访问 USR 寄存器中的 FERR 和 PERR 错误标志位。

UART 模块中断结构

几个独立的 UART 条件可以产生一个 UART 中断。当条件满足时，会产生一个低脉冲信号以引起单片机注意。发送寄存器为空、发送器空闲、接收器数据有效、溢出和地址检测和 RX 引脚唤醒都会产生中断。当其中任何一种情况发生时，若其对应的中断控制位使能、整体 UART 中断允许且堆栈未滿，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。其中四种情况，若其 UCR2 寄存器中相应中断使能位被置位，则 USR 寄存器中对应中断标志位将产生 UART 中断。发送器相关的两个中断情况有各自对应的中断使能位，而接收器相关的两个中断情况共用一个中断使能位。这些使能位可用于禁止个别的 UART 中断源。

地址检测也是 UART 的中断源，它没有相应的标志位，若 UCR2 寄存器中 ADDEN=1，当检测到地址将会产生 UART 中断。RX 引脚唤醒也可以产生 UART 中断，它没有相应的标志位，当 UART 时钟源 f_{HCLK} 关闭且 UCR2 中的 WAKE 和 RIE 位被置位，RX 引脚上有下降沿时会产生 UART 中断。

应注意，USR 寄存器标志位为只读状态，软件不能对其进行设置，和其它一些中断一样，在进入相应中断服务程序时也不能清除这些标志位。这些标志位仅在 UART 特定动作发生时才会自动被清除，详细解释见 UART 寄存器章节。整体 UART 中断的使能或除能可由中断控制寄存器中的相关中断使能控制位控制，其中断请求由 UART 模块决定。



UART 中断结构

地址检测模式

置位 UCR2 寄存器中的 ADDEN 将启动地址检测模式。若此位为“1”，可产生接收数据有效中断，其请求标志位为 RXIF。若 ADDEN 有效，只有在接收到数据最高位为 1 才会产生中断，中断使能位 UARTE 和 EMI 也要使能才会产生中断。地址的最高位为第 9 位 (BNO=1) 或第 8 位 (BNO=0)，若此位为高，则接收到的是地址而非数据。只有接收的数据的最后一位为高才会产生中断。若 ADDEN 除能，每接收到一个有效数据便会置位 RXIF，而不用考虑数据的最后一位。地址检测和奇偶校验在功能上相互排斥，若地址检测模式使能，为了确保操作正确，必须将奇偶校验使能位清零以除能奇偶校验。

ADDEN	Bit 9 (BNO=1) Bit 8 (BNO=0)	产生 UART 中断
0	0	√
	1	√
1	0	×
	1	√

ADDEN 位功能

UART 模块暂停和唤醒

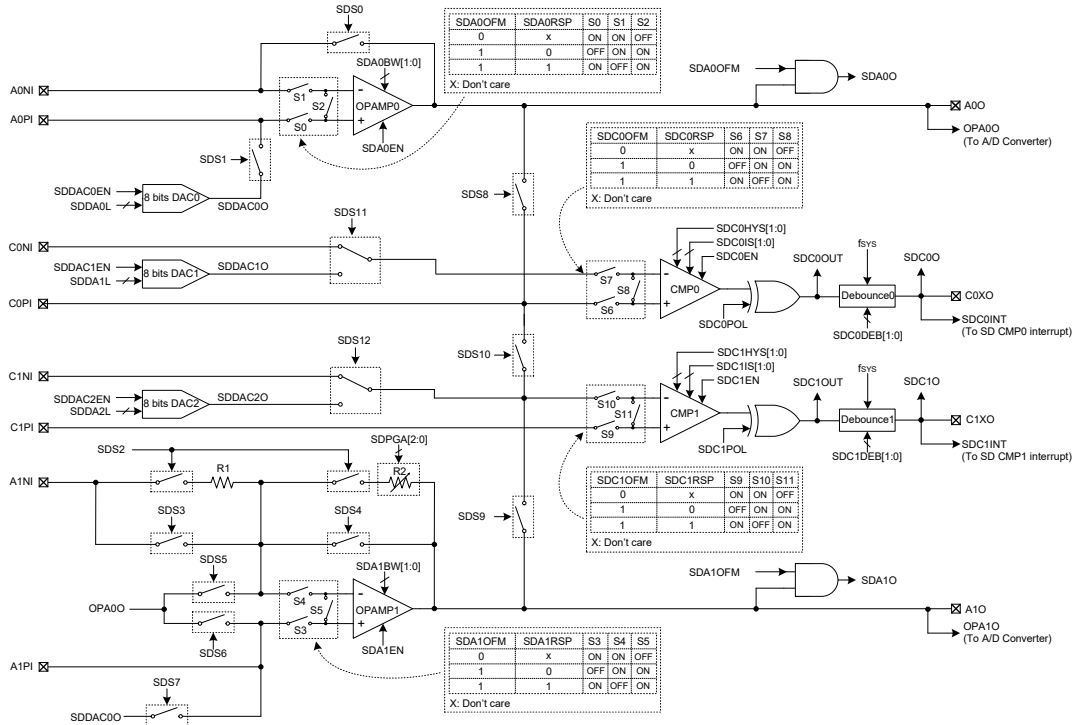
UART 时钟 f_{H} 关闭后 UART 模块将停止运行。当传送数据时 UART 时钟 f_{H} 关闭，发送将停止直到 UART 模块时钟再次使能。同样地，当接收数据时单片机进入空闲或休眠模式，数据接收也会停止。当单片机进入空闲或休眠模式，USR、UCR1、UCR2、接收 / 发送寄存器以及 BRG 寄存器都不会受到影响。建议在单片机进入空闲或休眠模式前先确保数据发送或接收已完成。

UART 功能中包括了 RX 引脚的唤醒功能，由 UCR2 寄存器中 WAKE 位控制。当单片机进入空闲或休眠模式且 UART 时钟 f_{H} 关闭时，若 WAKE 位与 UART 使能位 UARTE、接收器使能位 RXEN 和接收器中断使能位 RIE 都被置位，则 RX 引脚的下降沿可触发产生 RX 引脚唤醒 UART 的中断。注意，唤醒后系统需延时一段时间才能正常工作，在此期间，RX 引脚上的任何数据将被忽略。

若要唤醒并产生 UART 中断，除了唤醒使能控制位和接收中断使能控制位需置位外，全局中断使能位 EMI 和 UART 中断使能控制位 UARTE 也必须置位；若这两控制位没有被置位，那么，单片机将可以被唤醒但不会产生中断。同样唤醒后系统需一定的延时才能正常工作，然后才会产生 UART 中断。

运算放大器与比较器

该系列单片机内置两个运算放大器、两个比较器和三个 8 位 D/A 转换器，可用于需要外部传感器接口的应用程序。下面的方框图说明了这些内部模拟功能。



内部模拟功能方框图

模拟功能操作

该运算放大器可根据用户的特殊要求将信号放大。8 位 D/A 转换器可以给运算放大器同相输入和比较器反相输入提供一个准确的参考电压。比较器是用来比较两个模拟电压，并提供它们的差分输出，若对应的中断控制功能使能将产生一个中断。

该运算放大器具有多个开关和输入路径选择、多种参考电压的选择、多种内部软件增益控制、去抖时间控制、迟滞、偏移参考电压校准功能和其他功能，这些功能增强了该电路的灵活性使其可以广泛的应用于各种应用中。

控制寄存器

模拟电路整体的操作使用一系列寄存器控制。SDSW0~SDSW1 寄存器用于控制开关和 SD 比较器 n 反相输入端的连接。SDDAC0C~SDDAC2C 寄存器用于控制 SD DACn 的使能 / 除能。SDDA0L~SDDA2L 寄存器用于控制 SD DACn 输出电压。SDPGAC 寄存器用于控制 PGA R2/R1 比率和 SD 比较器 n 输出极性。SDA0C~SDA1C 寄存器用于控制 SD OPAn 的使能 / 除能和带宽功能来表明其当前状态。SDA0VOS~SDA1VOS 寄存器用于控制 SD OPAn 输入失调电压校准功能。SDC0C~SDC1C 寄存器用于控制 SD 比较器 n 的使能 / 除能和去抖时间来表明其当前状态。SDC0VOS~SDC1VOS 寄存器用来控制 SD 比较器 n 输入失调电压校准功能。SDCHYC 寄存器用于控制 SD 比较器 n 的迟滞功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SDSW0	SDS7	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
SDSW1	—	—	—	SDS12	SDS11	SDS10	SDS9	SDS8
SDDAC0C	SDDAC0EN	—	—	—	—	—	—	—
SDDAC1C	SDDAC1EN	—	—	—	—	—	—	—
SDDAC2C	SDDAC2EN	—	—	—	—	—	—	—
SDDA0L	D7	D6	D5	D4	D3	D2	D1	D0
SDDA1L	D7	D6	D5	D4	D3	D2	D1	D0
SDDA2L	D7	D6	D5	D4	D3	D2	D1	D0
SDPGAC	SDC0POL	SDC1POL	—	—	—	SDPGA2	SDPGA1	SDPGA0
SDA0C	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
SDA1C	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
SDA0VOS	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
SDA1VOS	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0
SDC0C	SDC0OUT	SDC0EN	SDC0O	—	SDC0DEB1	SDC0DEB0	SDC0IS1	SDC0IS0
SDC1C	SDC1OUT	SDC1EN	SDC1O	—	SDC1DEB1	SDC1DEB0	SDC1IS1	SDC1IS0
SDC0VOS	—	SDC0OFM	SDC0RSP	SDC0OF4	SDC0OF3	SDC0OF2	SDC0OF1	SDC0OF0
SDC1VOS	—	SDC1OFM	SDC1RSP	SDC1OF4	SDC1OF3	SDC1OF2	SDC1OF1	SDC1OF0
SDCHYC	—	—	—	—	SDC1HYS1	SDC1HYS0	SDC0HYS1	SDC0HYS0

模拟功能控制寄存器列表

● SDSW0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SDS7	SDS6	SDS5	SDS4	SDS3	SDS2	SDS1	SDS0
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 **SDS7:** on/off 控制位
0: Off
1: On
- Bit 6 **SDS6:** on/off 控制位
0: Off
1: On
- Bit 5 **SDS5:** on/off 控制位
0: Off
1: On
- Bit 4 **SDS4:** on/off 控制位
0: Off
1: On
- Bit 3 **SDS3:** on/off 控制位
0: Off
1: On
- Bit 2 **SDS2:** on/off 控制位
0: Off
1: On
- Bit 1 **SDS1:** on/off 控制位
0: Off
1: On
- Bit 0 **SDS0:** on/off 控制位
0: Off
1: On

● **SDSW1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	SDS12	SDS11	SDS10	SDS9	SDS8
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 **SDS12**: SD 比较器 1 反相端连接选择位
0: 连接至 SD DAC2
1: 连接至 C1NI

Bit 3 **SDS11**: SD 比较器 0 反相端连接选择位
0: 连接至 SD DAC1
1: 连接至 C0NI

Bit 2 **SDS10**: on/off 控制位
0: Off
1: On

Bit 1 **SDS9**: on/off 控制位
0: Off
1: On

Bit 0 **SDS8**: on/off 控制位
0: Off
1: On

● **SDDAC0C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SDDAC0EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC0EN**: SD DAC0 使能或除能控制位
0: 除能
1: 使能

Bit 6~0 未定义，读为“0”

● **SDDAC1C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SDDAC1EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC1EN**: SD DAC1 使能或除能控制位
0: 除能
1: 使能

Bit 6~0 未定义，读为“0”

● **SDDAC2C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SDDAC2EN	—	—	—	—	—	—	—
R/W	R/W	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **SDDAC2EN**: SD DAC2 使能或除能控制位

0: 除能

1: 使能

Bit 6~0 未定义, 读为 “0”

● **SDDA0L 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**: SD DAC0 输出控制码

$$SDDAC0O = (DAC AV_{DD}/2^8) \times D[7:0]$$

● **SDDA1L 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**: SD DAC1 输出控制码

$$SDDAC1O = (DAC AV_{DD}/2^8) \times D[7:0]$$

● **SDDA2L 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**: SD DAC2 输出控制码

$$SDDAC2O = (DAC AV_{DD}/2^8) \times D[7:0]$$

● **SDPGAC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SDC0POL	SDC1POL	—	—	—	SDPGA2	SDPGA1	SDPGA0
R/W	R/W	R/W	—	—	—	R/W	R/W	R/W
POR	0	0	—	—	—	0	0	0

Bit 7 **SDC0POL**: SD 比较器 0 输出极性控制位

0: 输出同相

1: 输出反相

该位是 SD 比较器 0 输出极性控制位。SDC0POL=0, SDC0OUT 位将反映比较器同相输出条件, SDC0POL=1, SDC0OUT 位将反映比较器反相输出条件。

- Bit 6 **SDC1POL**: SD 比较器 1 输出极性控制位
 0: 输出同相
 1: 输出反相
 该位是 SD 比较器 1 输出极性控制位。SDC1POL=0, SDC1OUT 位将反映比较器同相输出条件, SDC1POL=1, SDC1OUT 位将反映比较器反相输出条件。
- Bit 5~3 未定义, 读为 “0”
- Bit 2~0 **SDPGA2~SDPGA0**: PGA R2/R1 比率控制位
 000: ×1
 001: ×2
 010: ×4
 011: ×8
 100: ×16
 101: ×32
 110: ×56
 111: ×1

• SDA0C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	SDA0EN	SDA0O	—	—	—	SDA0BW1	SDA0BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

- Bit 7 未定义, 读为 “0”
- Bit 6 **SDA0EN**: SD OPA0 使能或除能控制位
 0: 除能
 1: 使能
- Bit 5 **SDA0O**: SD OPA0 输出状态 (正逻辑)
 该位是只读位。
 当 SDA0OFM 设置为 “1”, SDA0O 定义 SD OPA0 的输出状态, 详细的偏移校准程序参考 “运算放大器输入失调校准” 章节。
 当 SDA0OFM 位清零, 该位将固定为低电平。
- Bit 4~2 未定义, 读为 “0”
- Bit 1~0 **SDA0BW1~SDA0BW0**: SD OPA0 带宽控制位
 00: 5kHz
 01: 40kHz
 10: 600kHz
 11: 2MHz
 详细内容参考 “运算放大器电气特性”。

• SDA1C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	SDA1EN	SDA1O	—	—	—	SDA1BW1	SDA1BW0
R/W	—	R/W	R	—	—	—	R/W	R/W
POR	—	0	0	—	—	—	0	0

- Bit 7 未定义, 读为 “0”
- Bit 6 **SDA1EN**: SD OPA1 使能或除能控制位
 0: 除能
 1: 使能

- Bit 5 **SDA1O**: SD OPA1 输出状态（正逻辑）
该位是只读位。
当 SDA1OFM 为“1”，SDA1O 定义 SD OPA1 定义输出状态。详细的偏移校准程序参考“运算放大器输入失调校准”章节。
当 SDA1OFM 位清零，该位将固定为低电平。
- Bit 4~2 未定义，读为“0”
- Bit 1~0 **SDA1BW1~SDA1BW0**: SD OPA1 带宽控制位
00: 5kHz
01: 40kHz
10: 600kHz
11: 2MHz
详细内容参考“运算放大器电气特性”。

● SDA0VOS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SDA0OFM	SDA0RSP	SDA0OF5	SDA0OF4	SDA0OF3	SDA0OF2	SDA0OF1	SDA0OF0
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 **SDA0OFM**: SD OPA0 正常操作或输入失调电压校准模式选择位
0: 正常操作
1: 输入失调电压校准模式
- Bit 6 **SDA0RSP**: SD OPA0 输入失调电压参考电压选择位
0: 选择 A0NI 作为参考电压输入
1: 选择 A0PI 作为参考电压输入
- Bit 5~0 **SDA0OF5~SDA0OF0**: SD OPA0 输入失调电压参考电压控制位
SDA0OF5~SDA0OF0 位用于执行运算放大器输入失调电压的操作和 SD OPA0 输入失调电压值存回该位段，更多详细信息的描述在“运算放大器输入失调电压校准”章节。

● SDA1VOS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SDA1OFM	SDA1RSP	SDA1OF5	SDA1OF4	SDA1OF3	SDA1OF2	SDA1OF1	SDA1OF0
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 **SDA1OFM**: SD OPA1 正常操作或输入失调电压校准模式选择位
0: 正常操作
1: 输入失调电压校准模式
- Bit 6 **SDA1RSP**: SD OPA1 输入失调电压参考电压选择位
0: 选择 A1NI 作为参考电压输入
1: 选择 A1PI 作为参考电压输入
- Bit 5~0 **SDA1OF5~SDA1OF0**: SD OPA1 输入失调电压参考电压控制位
SDA1OF5~SDA1OF0 位用于执行运算放大器输入失调电压的操作和 SD OPA1 输入失调电压值存回该位段，更多详细信息的描述在“运算放大器输入失调电压校准”章节。

● SDC0C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SDC0OUT	SDC0EN	SDC0O	—	SDC0DEB1	SDC0DEB0	SDC0IS1	SDC0IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

- Bit 7 **SDC0OUT**: SD 比较器 0 输出位
若 SDC0POL = 0,
0: C0NI > C0PI
1: C0PI > C0NI
若 SDC0POL = 1,
0: C0NI < C0PI
1: C0PI < C0NI
该位存储 SD 比较器 0 输出位。该位的极性通过 SD 比较器 0 输入和 SDC0POL 位的条件决定。
- Bit 6 **SDC0EN**: SD 比较器 0 使能或除能控制位
0: 比较器除能
1: 比较器使能
该位是 SD 比较器 0 开关控制位。如果该位清零, SD 比较器 0 输出低。否则, 当 SDC0POL=0 时设置 SDC0OUT=0 或当 SDC0POL=1 设置 SDC0OUT=1。
- Bit 5 **SDC0O**: SD 比较器 0 去抖输出
SDC0O 是 SDC0OUT 去抖后的输出
若 SDC0POL=0, 仅当 SDC0OUT 当前及之前 N 个采样为“1” SDC0O 输出“1”。
若 SDC0POL=1, 仅当 SDC0OUT 当前及之前 N 个采样为“0” SDC0O 输出“0”。
N 采样取决于 SDC0DEB[1:0] 位的配置。
- Bit 4 未定义, 读为“0”
- Bit 3~2 **SDC0DEB1~SDC0DEB0**: SD 比较器 0 去抖时间控制
00: 无去抖功能
01: $(31\sim32) \times 1/f_{SYS}$
10: $(63\sim64) \times 1/f_{SYS}$
11: $(126\sim127) \times 1/f_{SYS}$
- Bit 1~0 **SDC0IS1~SDC0IS0**: SD 比较器 0 电流控制
详细内容参考“比较器电气特性”。

● SDC1C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SDC1OUT	SDC1EN	SDC1O	—	SDC1DEB1	SDC1DEB0	SDC1IS1	SDC1IS0
R/W	R	R/W	R	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

- Bit 7 **SDC1OUT**: SD 比较器 1 输出位
若 SDC1POL = 0,
0: C1NI > C1PI
1: C1PI > C1NI
若 SDC1POL = 1,
0: C1NI < C1PI
1: C1PI < C1NI
该位存储 SD 比较器 1 输出位。该位的极性通过 SD 比较器 1 输入和 SDC1POL 位的条件决定。

- Bit 6 **SDC1EN**: SD 比较器 1 使能或除能控制位
0: 比较器除能
1: 比较器使能
该位是 SD 比较器 1 开关控制位。如果该位清零, SD 比较器 1 输出低。否则, 当 SDC1POL=0 时设置 SDC1OUT=0 或当 SDC1POL=1 设置 SDC1OUT=1。
- Bit 5 **SDC1O**: SD 比较器 1 去抖输出
SDC1O 是 SDC1OUT 去抖后的输出
若 SDC1POL=0, 仅当 SDC1OUT 当前及之前 N 个采样为“1” SDC1O 输出“1”。
若 SDC1POL=1, 仅当 SDC1OUT 当前及之前 N 个采样为“0” SDC1O 输出“0”。
N 采样取决于 SDC1DEB[1:0] 位的配置。
- Bit 4 未定义, 读为“0”
- Bit 3~2 **SDC1DEB1~SDC1DEB0**: SD 比较器 1 去抖时间控制
00: 无去抖功能
01: $(31\sim32) \times 1/f_{SYS}$
10: $(63\sim64) \times 1/f_{SYS}$
11: $(126\sim127) \times 1/f_{SYS}$
- Bit 1~0 **SDC1IS1~SDC1IS0**: SD 比较器 1 电流控制
详细内容参考“比较器电气特性”

• SDC0VOS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	SDC0OFM	SDC0RSP	SDC0OF4	SDC0OF3	SDC0OF2	SDC0OF1	SDC0OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

- Bit 7 未定义, 读为“0”
- Bit 6 **SDC0OFM**: SD 比较器 0 正常操作或输入失调电压模式选择位
0: 正常操作
1: 输入失调电压模式
- Bit 5 **SDC0RSP**: SD 比较器 0 输入失调电压参考电压选择位
0: 选择 C0NI 作为参考输入
1: 选择 C0PI 作为参考输入
- Bit 4~0 **SDC0OF4~SDC0OF0**: SD 比较器 0 输入失调电压参考电压控制位
SDC0OF4~SDC0OF0 位用于执行比较器器输入失调电压的操作和 SD 比较器 0 输入失调电压值存回该位段, 更多详细信息的描述在“比较器输入失调电压校准”章节。

• SDC1VOS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	SDC1OFM	SDC1RSP	SDC1OF4	SDC1OF3	SDC1OF2	SDC1OF1	SDC1OF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **SDC1OFM**: SD 比较器 1 正常操作或输入失调电压模式选择位
0: 正常操作
1: 输入失调电压模式

Bit 5 **SDC1RSP**: SD 比较器 1 输入失调电压参考电压选择位
0: 选择 C1NI 作为参考输入
1: 选择 C1PI 作为参考输入

Bit 4~0 **SDC1OF4~SDC1OF0**: SD 比较器 1 输入失调电压参考电压控制位
SDC1OF4~SDC1OF0 位用于执行比较器输入失调电压的操作和 SD 比较器 1 输入失调电压值存回该位段，更多详细信息的描述在“比较器输入失调电压校准”章节。

• SDCHYC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	SDC1HYS1	SDC1HYS0	SDC0HYS1	SDC0HYS0
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 **SDC1HYS1~SDC1HYS0**: 比较器 1 迟滞电压窗口控制
更多详细信息的参考“比较器输入失调电压校准”章节。

Bit 1~0 **SDC0HYS1~SDC0HYS0**: 比较器 0 迟滞电压窗口控制
更多详细信息的参考“比较器输入失调电压校准”章节。

失调校准程序

SDAnOFM 或 SDCnOFM 位先设置为“1”使 SD 运算放大器 n 或 SD 比较器 n 处于输入失调校准模式，然后通过配置 SDAnRSP 或 SDCnRSP 位选择输入失调电压参考电压。应注意，若 SD 运算放大器 n 或 SD 比较器 n 输入引脚与 I/O 引脚共用，应先将引脚配置为 SD 运算放大器 n 或 SD 比较器 n 输入引脚。

运算放大器输入失调校准

步骤 1: 设置 SDAnOFM=1 且 SDAnRSP=1，使 SD 运算放大器 n 工作于失调校准模式，开关 S0 和 S2 都 ON 或 S3 和 S5 都 ON。为了确保校准后的 V_{AnOS} 尽可能小，校准模式下的输入参考电压应该跟正常模式下的输入直流工作电压相同。

步骤 2: 设置 SDAnOF[5:0]=000000，请延迟一段时间后再读取 SDAnO 位。

步骤 3: 使 SDAnOF[5:0] = SDAnOF[5:0] + 1，延迟一段时间后再读取 SDAnO 位。如果 SDAnO 位状态不改变，然后重复步骤 3，直到 SDAnO 位状态改变。如果 SDAnO 位状态改变，记录此时的 SDAnOF[5:0] 值为 V_{AnOS1} 然后转到步骤 4。

步骤 4: 设置 SDAnOF[5:0]=111111，请延迟一段时间后再读取 SDAnO 位。

步骤 5: 使 $SDAnOF[5:0]=SDAnOF[5:0] - 1$, 延迟一段时间后再读取 $SDAnO$ 位。如果 $SDAnO$ 位状态不改变, 然后重复步骤 5, 直到 $SDAnO$ 位状态改变。如果 $SDAnO$ 位状态改变, 记录此时的 $SDAnOF[5:0]$ 值为 V_{AnOS2} 然后转到步骤 6。

步骤 6: 将 SD 运算放大器 n 输入失调校准值 V_{AnOS} 存入 $SDAnOF[5:0]$ 位中, 校准结束。

$$V_{AnOS} = (V_{AnOS1} + V_{AnOS2}) / 2。如果 (V_{AnOS1} + V_{AnOS2}) / 2 不是整数, 舍弃小数。$$

注: 1. 当 $SDAnOF[5:0]=000000$ 时, $SDAnO = 0$; 反之 $SDAnOF[5:0]=111111$ 时, $SDAnO = 1$ 。

2. 于失调校准模式中, 当 $SDAnOF[5:0]=000000$ 时, $SDAnO=0$, 当 $SDAnOF[5:0]$ 往上调至 111111 时, $SDAnO=1$

3. 依频带宽度的长短来决定延迟时间的大小, 当频带宽度越小所需的延迟时间越大; 反之频带宽度越大所需的延迟时间越小。

4. 每更换一次频带宽度就要做一次失调校准。

比较器输入失调校准

步骤 1: 设置 $SDCnOFM=1$ 且 $SDCnRSP=1$, 使 SD 比较器 n 工作于失调校准模式, 开关 S6 和 S8 都 ON 或 S9 和 S11 都 ON。为了确保校准后的 V_{CnOS} 尽可能小, 校准模式下的输入参考电压应该跟正常模式下的输入直流工作电压相同。

步骤 2: 设置 $SDCnOF[4:0]=00000$, 请延迟一段时间后再读取 $SDCnOUT$ 位。

步骤 3: 使 $SDCnOF[4:0]=SDCnOF[4:0] + 1$, 延迟一段时间后再读取 $SDCnOUT$ 位。如果 $SDCnOUT$ 位状态不改变, 然后重复步骤 3, 直到 $SDCnOUT$ 位状态改变。如果 $SDCnOUT$ 位状态改变, 记录此时的 $SDCnOF[4:0]$ 值为 V_{CnOS1} 然后转到步骤 4。

步骤 4: 设置 $SDCnOF[4:0]=11111$, 请延迟一段时间后再读取 $SDCnOUT$ 位。

步骤 5: 使 $SDCnOF[4:0]=SDCnOF[4:0] - 1$, 延迟一段时间后再读取 $SDCnOUT$ 位。如果 $SDCnOUT$ 位状态不改变, 然后重复步骤 5, 直到 $SDCnOUT$ 位状态改变。如果 $SDCnOUT$ 位状态改变, 记录此时的 $SDCnOF[4:0]$ 值为 V_{CnOS2} 然后转到步骤 6。

步骤 6: 将 SD 比较器 n 输入失调校准值 V_{CnOS} 存入 $SDCnOF[4:0]$ 位中, 校准结束。

$$V_{CnOS} = (V_{CnOS1} + V_{CnOS2}) / 2。如果 (V_{CnOS1} + V_{CnOS2}) / 2 不是整数, 舍弃小数。$$

注: 1. 当 $SDCnOF[4:0]=00000$ 时, $SDCnOUT=1$; 反之 $SDCnOF[4:0]=11111$ 时, $SDCnOUT=0$ 。

2. 于失调校准模式中, 当 $SDCnOF[4:0]=00000$ 时, $SDCnOUT = 1$, 当 $SDCnOF[4:0]$ 往上调至 11111 时, $SDCnOUT = 0$

3. 依响应时间的长短来决定延迟时间的大小, 当响应时间越长所需的延迟时间越大; 反之响应时间越短所需的延迟时间越小。

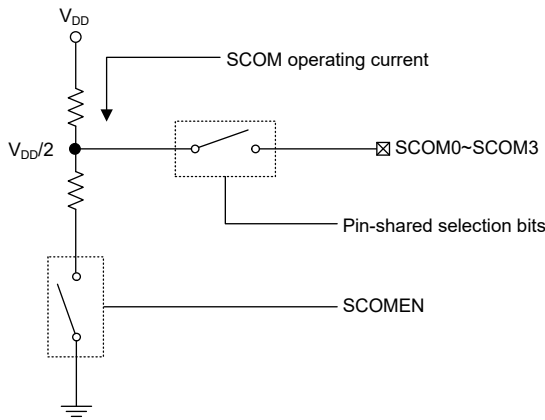
带 SCOM 功能的 LCD – 仅适用于 HT66F4540/HT66F4550/HT66F4560

此系列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 口共用。LCD 信号（COM 和 SEG）由软件编程实现。

LCD 操作

单片机通过设置输入 / 输出口作为 COM 和 SEG 引脚以驱动外部的液晶面板。LCD 驱动功能是由 SCOMC 寄存器来控制，该寄存器除了可设置 LCD 的开启和关闭外还可控制输出偏压值等功能，使得 COM 口输出 $V_{DD}/2$ 的电压，从而实现 1/2 bias LCD 的显示。

SCOMC 寄存器中的 SCOMEN 位是 LCD 驱动的主控制位。LCD SCOM_n 引脚可通过相应的引脚共用功能选择位来选择哪些端口用于 LCD 驱动。应注意的是，端口控制寄存器不需要先设置为输出以使能 LCD 驱动操作。



LCD COM 偏压

LCD 偏压电流控制

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

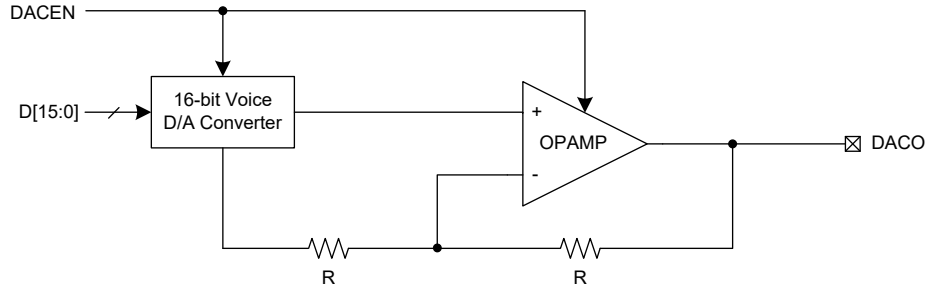
● SCOMC 寄存器

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**: 选择 R 型 LCD 偏压电流 ($V_{DD}=5V$)
- 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
- 当 SCOMEN 位为高时，所选择的直流电阻将被开启以产生 1/2 V_{DD} bias 的电压。
- Bit 3~0 未定义，读为“0”

语音 D/A 转换器 – HT66F4550/HT66F4560

HT66F4550/HT66F4560 单片机内置一个具有缓冲输出的 16-bit 语音 D/A 转换器电路和运算放大器，可用于音频应用。语音 D/A 转换器的参考源只能是模拟电源电压，能够降低功耗。16 位 D/A 转换器用于语音或音频应用。虽然该 D/A 转换器不是一对一的数字模拟转换器，但它给大小声源提供了良好和相同的音质。最后，通过运算放大器将输出电压放大，然后缓冲输出。



语音 D/A 转换器方框图

语音 D/A 转换器寄存器

语音 D/A 转换器整体操作由几个寄存器控制。DACC 寄存器用于 D/A 转换器的使能 / 除能控制。

DAH 和 DAL 寄存器对的 16 位值用于存储 D/A 转换器语音数据。

• DAH 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8
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 **D15~D8:** 16-bit 语音 D/A 转换器数据高字节

16 位 D/A 转换器的语音数据的低字节寄存器是 DAL，数据先写入 DAL 寄存器，再写入 DAH 寄存器。当每一次 DAH 寄存器被写入，16 位的数据将被加载到 D/A 转换器且一个转换周期初始化。应注意，在数据更新之前，必须先使能 D/A 转换器。

• DAL 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:** 16-bit 语音 D/A 转换器数据低字节

写该寄存器将只把数据写到影子缓冲器，然后写 DAH 寄存器的同时将影子缓冲器的数据复制到 DAL 寄存器。应注意，在数据更新之前，必须先使能 D/A 转换器。

• DACC 寄存器

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

Bit 7~1 未定义，读为“0”

Bit 0 **DACEN**: 16-bit 语音 D/A 转换器使能或除能控制位

0: 除能

1: 使能

如果 D/A 转换器使能，用户必须等待 t_{DACS} 以保证 D/A 转换电路是稳定的。一段时间 t_{DACS} 后 D/A 转换器电路的稳定。在该电路是稳定后，16 位语音 / 数据转换器的数据寄存器更新。

低电压检测 – LVD

此系列单片机都具有低电压检测功能，即 LVD。该功能使能用于监测电源电压 V_{DD} ，若电源电压低于一定值可提供一个警告信号。此功能在电池类产品中非常有用，在电池电压较低时产生警告信号。低电压检测也可产生中断信号。

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择八个固定电压参考点的其中一个参考点。LVDO 位被置位时低电压情况发生，若 LVDO 位为低表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一定的功耗，在不使用时可考虑关闭此功能，此举在功耗要求严格的电池供电应用中值得考虑。

• LVDC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	VBGEN	VLVD2	VLVD1	VLVD0
R/W	—	—	R	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 **LVDO**: LVD 输出标志位

0: 未检测到低电压

1: 检测到低电压

Bit 4 **LVDEN**: 低电压检测控制位

0: 除能

1: 使能

Bit 3 **VBGEN**: Bandgap 缓冲器控制

0: 除能

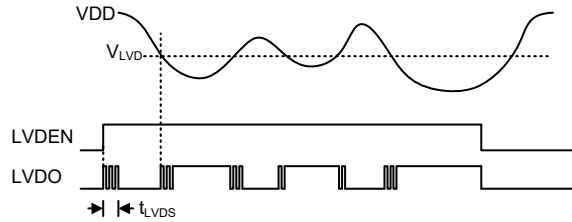
1: 使能

应注意，当 LVD 或 LVR 功能使能或此位置位时，Bandgap 电路使能。

Bit 2~0	VLVD2~VLVD0: 选择 LVD 电压
	000: 2.0V
	001: 2.2V
	010: 2.4V
	011: 2.7V
	100: 3.0V
	101: 3.3V
	110: 3.6V
	111: 4.0V

LVD 操作

通过比较电源电压 V_{DD} 与存储在 LVDC 寄存器中的预置电压值的结果，低电压检测功能工作。其设置的值范围为 2.0V~4.0V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置高，表明低电压产生。低电压检测功能由一个自动使能的参考电压提供。当单片机处于休眠模式时，即使 LVDEN 位为高，低电压检测器除能。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDS} 。注意， V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

低电压检测器也有自己的中断功能，它是除了轮询 LVDO 位之外的另一种检测低电压的方法。中断条件产生置位 LVDO 并延时 t_{LVD} 后，中断产生。此种情况下，若 V_{DD} 降至小于 LVD 预置电压值时，中断请求标志位 LVF 将被置位，中断产生，单片机将被从休眠或空闲模式中唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

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 转换器有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此系列单片机提供多个外部中断和内部中断功能，外部中断由 INT0~INT1 引脚动作产生，而内部中断由各种内部功能，如定时器模块、时基、LVD、EEPROM 和 A/D 转换器产生。

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专用数据存储器中的一系列寄存器控制的。寄存器总的分为三类。第一类是 INTC0~INTC3 寄存器，用于设置基本的中断；第二类是 MFI0~MFI1 寄存器，用于设置多功能中断；最后一种为 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	注释
总中断	EMI	—	—
INTn 脚	INTnE	INTnF	n=0~1
SD 比较器	SDCnE	SDCnF	n=0~1
多功能中断	MFnE	MFnF	n=0~1
A/D 转换器	ADE	ADF	—
时基	TBnE	TBnF	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM 写操作	DEE	DEF	—
STMn	STMnPE	STMnPF	n=0
	STMnAE	STMnAF	
PTMn	PTMnPE	PTMnPF	n=0
	PTMnAE	PTMnAF	

中断寄存器位命名模式 – HT66F4530

功能	使能位	请求标志	注释
总中断	EMI	—	—
INT _n 引脚	INT _n E	INT _n F	n=0~1
SD 比较器	SDC _n E	SDC _n F	n=0~1
多功能中断	MF _n E	MF _n F	n=0~1
A/D 转换器	ADE	ADF	—
时基	TB _n E	TB _n F	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
UART	UARTE	UARTF	—
STM _n	STM _n PE	STM _n PF	n=0
	STM _n AE	STM _n AF	
PTM _n	PTM _n PE	PTM _n PF	n=0~1
	PTM _n AE	PTM _n AF	

中断寄存器位命名模式 – HT66F4540

功能	使能位	请求标志	注释
总中断	EMI	—	—
INT _n 引脚	INT _n E	INT _n F	n=0~1
SD 比较器	SDC _n E	SDC _n F	n=0~1
多功能中断	MF _n E	MF _n F	n=0~1
A/D 转换器	ADE	ADF	—
时基	TB _n E	TB _n F	n=0~1
SIM	SIME	SIMF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
UART	UARTE	UARTF	—
STM _n	STM _n PE	STM _n PF	n=0~1
	STM _n AE	STM _n AF	
PTM _n	PTM _n PE	PTM _n PF	n=0~1
	PTM _n AE	PTM _n AF	

中断寄存器位命名模式 – HT66F4550/HT66F4560

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	—	DEF	—	—	—	DEE
MF10	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
MF11	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE

中断寄存器列表 – HT66F4530

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	UARTF	DEF	—	—	UARTE	DEE
MF10	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MF11	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE

中断寄存器列表 – HT66F4540

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	EMI
INTC1	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
INTC2	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
INTC3	—	—	UARTF	DEF	—	—	UARTE	DEE
MF10	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
MF11	STM1AF	STM1PF	STM0AF	STM0PF	STM1AE	STM1PE	STM0AE	STM0PE

中断寄存器列表 – HT66F4550/HT66F4560

● INTEG 寄存器

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 脚中断边沿控制位

- 00: 除能
- 01: 上升沿
- 10: 下降沿
- 11: 双沿

Bit 1~0 **INT0S1~INT0S0**: INT0 脚中断边沿控制位
 00: 除能
 01: 上升沿
 10: 下降沿
 11: 双沿

• INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	MF0F	SDC0F	INT0F	MF0E	SDC0E	INT0E	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 **MF0F**: 多功能中断 0 请求标志位
 0: 无请求
 1: 中断请求
 Bit 5 **SDC0F**: SD 比较器 0 请求标志位
 0: 无请求
 1: 中断请求
 Bit 4 **INT0F**: INT0 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 3 **MF0E**: 多功能中断 0 控制位
 0: 除能
 1: 使能
 Bit 2 **SDC0E**: SD 比较器 0 控制位
 0: 除能
 1: 使能
 Bit 1 **INT0E**: INT0 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	TB1F	TB0F	ADF	MF1F	TB1E	TB0E	ADE	MF1E
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 **TB1F**: 时基 1 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 6 **TB0F**: 时基 0 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 5 **ADF**: A/D 转换器中断请求标志位
 0: 无请求
 1: 中断请求

- Bit 4 **MF1F**: 多功能中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **TB1E**: 时基 1 中断控制位
0: 除能
1: 使能
- Bit 2 **TB0E**: 时基 0 中断控制位
0: 除能
1: 使能
- Bit 1 **ADE**: A/D 转换器中断控制位
0: 除能
1: 使能
- Bit 0 **MF1E**: 多功能中断控制位
0: 除能
1: 使能

● **INTC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	LVF	SDC1F	SIMF	INT1F	LVE	SDC1E	SIME	INT1E
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 **LVF**: LVD 中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **SDC1F**: SD 比较器 1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **SIMF**: SIM 中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **INT1F**: INT1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **LVE**: LVD 中断控制位
0: 除能
1: 使能
- Bit 2 **SDC1E**: SD 比较器 1 中断控制位
0: 除能
1: 使能
- Bit 1 **SIME**: SIM 中断控制位
0: 除能
1: 使能
- Bit 0 **INT1E**: INT1 中断控制位
0: 除能
1: 使能

● **INTC3 寄存器 – HT66F4530**

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

Bit 7~5 未定义，读为“0”

Bit 4 **DEF**: 数据 EEPROM 中断请求标志位
0: 无请求
1: 中断请求

Bit 3~1 未定义，读为“0”

Bit 0 **DEE**: 数据 EEPROM 中断控制位
0: 除能
1: 使能

● **INTC3 寄存器 – HT66F4540/HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	—	—	UARTF	DEF	—	—	UARTE	DEE
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 **UARTF**: UART 中断请求标志位
0: 无请求
1: 中断请求

Bit 4 **DEF**: 数据 EEPROM 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 **UARTE**: UART 中断控制位
0: 除能
1: 使能

Bit 0 **DEE**: 数据 EEPROM 中断控制位
0: 除能
1: 使能

● **MF10 寄存器 – HT66F4530**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
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 **PTM0AF**: PTM0 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求

Bit 4 **PTM0PF**: PTM0 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求

- Bit 3~2 未定义，读为“0”
- Bit 1 **PTM0AE**: PTM0 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **PTM0PE**: PTM0 比较器 P 匹配中断控制位
0: 除能
1: 使能

● **MF10 寄存器 – HT66F4540/HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	PTM1AF	PTM1PF	PTM0AF	PTM0PF	PTM1AE	PTM1PE	PTM0AE	PTM0PE
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 **PTM1AF**: PTM1 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **PTM1PF**: PTM1 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **PTM0AF**: PTM0 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **PTM0PF**: PTM0 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **PTM1AE**: PTM1 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 2 **PTM1PE**: PTM1 比较器 P 匹配中断控制位
0: 除能
1: 使能
- Bit 1 **PTM0AE**: PTM0 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **PTM0PE**: PTM0 比较器 P 匹配中断控制位
0: 除能
1: 使能

● **MF11 寄存器 – HT66F4530/HT66F4540**

Bit	7	6	5	4	3	2	1	0
Name	—	—	STM0AF	STM0PF	—	—	STM0AE	STM0PE
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 **STM0AF**: STM0 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求

- Bit 4 **STM0PF**: STM0 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未定义, 读为 “0”
- Bit 1 **STM0AE**: STM0 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **STM0PE**: STM0 比较器 P 匹配中断控制位
0: 除能
1: 使能

● **MF11 寄存器 – HT66F4550/HT66F4560**

Bit	7	6	5	4	3	2	1	0
Name	STM1AF	STM1PF	STM0AF	STM0PF	STM1AE	STM1PE	STM0AE	STM0PE
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 **STM1AF**: STM1 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **STM1PF**: STM1 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **STM0AF**: STM0 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **STM0PF**: STM0 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **STM1AE**: STM1 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 2 **STM1PE**: STM1 比较器 P 匹配中断控制位
0: 除能
1: 使能
- Bit 1 **STM0AE**: STM0 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **STM0PE**: STM0 比较器 P 匹配中断控制位
0: 除能
1: 使能

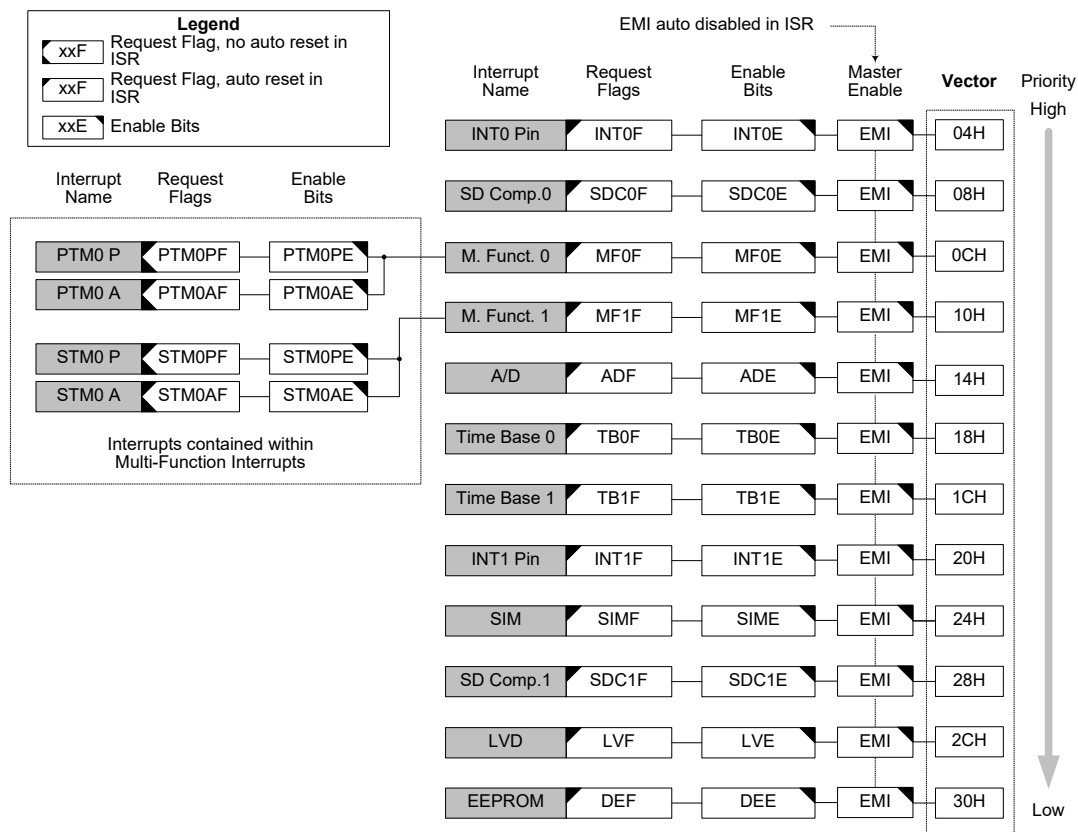
中断操作

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 转换完成等等，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。

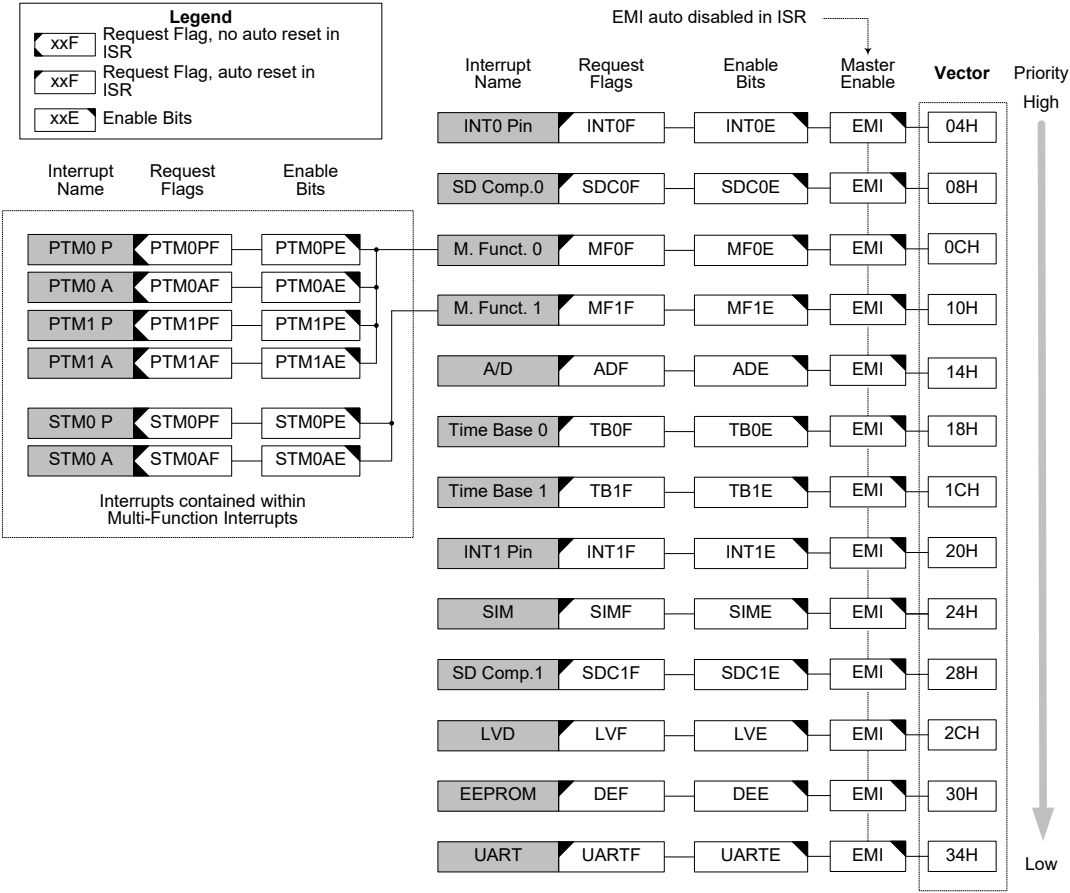
当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为跳转指令“JMP”，以跳转到相应的中断服务程序。这里的代码控制适当的中断。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

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。然而，如果其它中断请求发生在此期间，虽然中断不会立即响应，但是中断请求标志位会被记录。

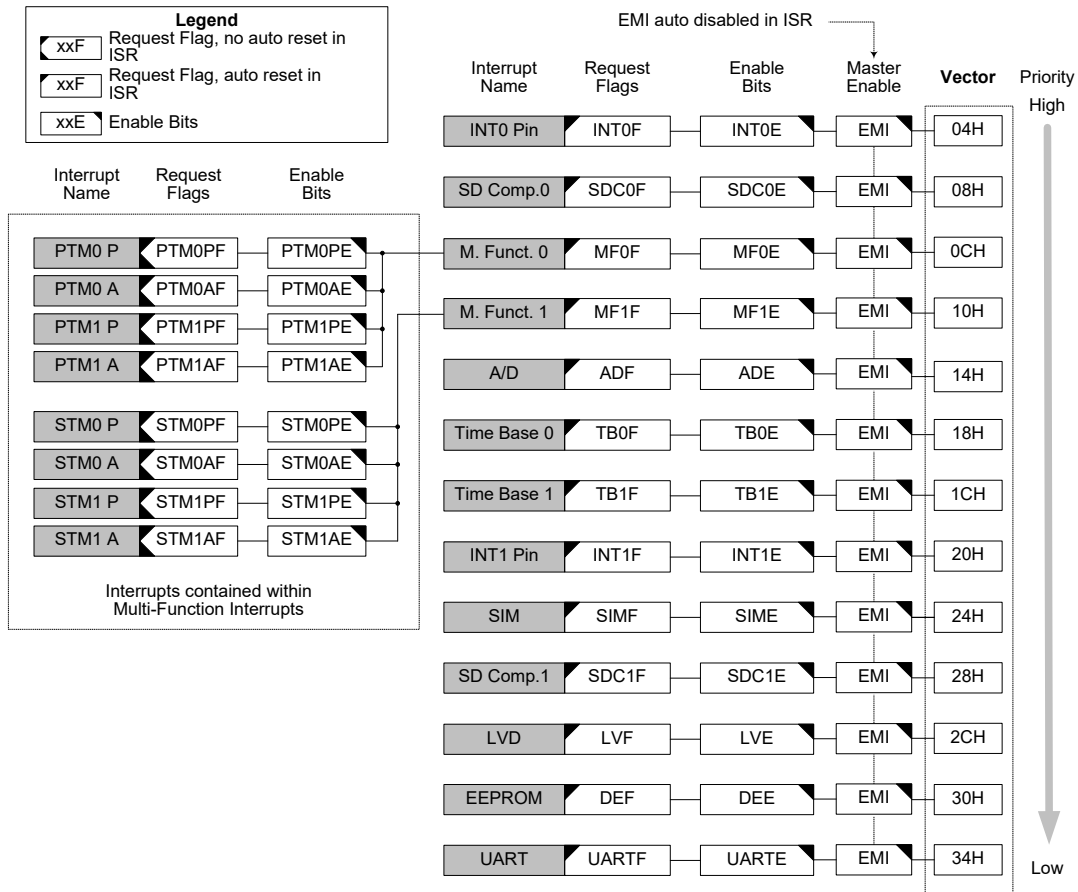
如果某个中断服务子程序正在执行时，有另一个中断要求立即响应，那么 EMI 位应在程序进入中断子程序后置位，以允许此中断嵌套。如果堆栈已满，即使此中断使能，中断请求也不会被响应，直到 SP 减少为止。如果要求立刻动作，则堆栈必须避免成为储满状态。请求同时发生时，执行优先级如下流程图所示。所有被置起的中断请求标志都可把单片机从休眠或空闲模式中唤醒，若要防止唤醒动作发生，在单片机进入休眠或空闲模式前应将相应的标志置起。



中断结构 – HT66F4530



中断结构 – HT66F4540



中断结构 – HT66F4550/HT66F4560

外部中断

通过 INT0~INT1 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT1 引脚的状态发生变化，外部中断请求标志 INT0F~INT1F 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 INT0E~INT1E 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和通过相关引脚共用选择位选择此引脚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 INT0F~INT1F 会自动复位且 EMI 位会被清零以除能其他中断。注意，即使此引脚被用作外部中断输入，其配置选项中的上拉电阻仍保持有效。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。

SD 比较器中断

SD 比较器中断由两个内部比较器控制。当 SD 比较器输出位状态改变，SD 比较器中断请求标志 SDCnF 被置位，SD 比较器中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和 SD 比较器中断使能位 SDCnE 需先被置位。当中断使能，堆栈未满并且 SD 比较器输入产生一个比较器输出位变化时，将调用 SD 比较器中断向量子程序。当响应中断服务子程序时，SD 比较器中断请求标志位 SDCnF 会自动复位且 EMI 位会被清零以除能其它中断。

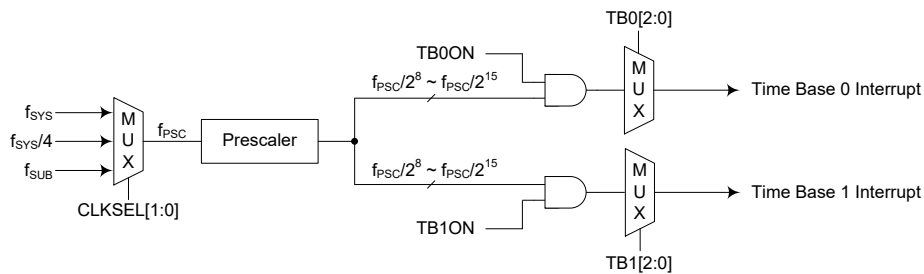
A/D 转换器中断

A/D 转换器中断由 A/D 转换动作的结束来控制。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 和 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。当响应中断服务子程序时，A/D 中断请求标志位 ADF 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断

时基中断是以中断的形式提供规律的时间信号。时基中断由各自的定时器功能产生溢出信号控制。当各自的中断请求标志 TB0F 或 TB1F 被置位时，中断请求发生。当总中断使能位 EMI 和时基中断使能位 TB0E 或 TB1E 被置位，允许程序跳转到对应的时基中断向量地址。当中断使能，堆栈未满且时基溢出时，将调用相应的时基中断向量子程序。当时基中断响应，相应的时基中断请求标志位 TB0F 或 TB1F 会自动复位且 EMI 将被自动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 f_{PSC} 来自内部时钟源 f_{SYS} 、 $f_{SYS}/4$ 或 f_{SUB} 。 f_{PSC} 输入时钟首先经过分频器，分频率由程序设置 TB0C 和 TB1C 寄存器相关位获取合适的分频值以提供更长的时基中断周期。相应的控制时基中断周期的时钟源可通过 PSCR 寄存器的 CLKSEL1~CLKSEL0 位选择。



时基中断

● PSCR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL1	CLKSEL0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **CLKSEL1~CLKSEL0**: 预分频时钟源选择

00: f_{SYS}

01: $f_{SYS}/4$

1x: f_{SUB}

● TB0C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TB0ON	—	—	—	—	TB02	TB01	TB00
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB0ON**: 时基 0 控制位

0: 除能

1: 使能

Bit 6~3 未定义，读为“0”

Bit 2~0 **TB02~TB00**: 选择时基 0 溢出周期

000: $2^8/f_{PSC}$

001: $2^9/f_{PSC}$

010: $2^{10}/f_{PSC}$

011: $2^{11}/f_{PSC}$

100: $2^{12}/f_{PSC}$

101: $2^{13}/f_{PSC}$

110: $2^{14}/f_{PSC}$

111: $2^{15}/f_{PSC}$

● TB1C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TB1ON	—	—	—	—	TB12	TB11	TB10
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB1ON**: 时基 1 控制位

0: 除能

1: 使能

Bit 6~3 未定义，读为“0”

Bit 2~0 **TB12~TB10**: 选择时基 1 溢出周期

000: $2^8/f_{PSC}$

001: $2^9/f_{PSC}$

010: $2^{10}/f_{PSC}$

011: $2^{11}/f_{PSC}$

100: $2^{12}/f_{PSC}$

101: $2^{13}/f_{PSC}$

110: $2^{14}/f_{PSC}$

111: $2^{15}/f_{PSC}$

SIM 中断

串行接口模块中断，即 SIM 中断。当一个字节数据已由 SIM 接口接收或发送完或 I²C 从机地址匹配或 I²C 超时发生时，中断请求标志 SIMF 被置位，SIM 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 和串行接口中断使能位 SIME 需先被置位。当中断使能，堆栈未满且上述任一情况发生时，可跳转至 SIM 中断向量子程序中执行。当串行接口中断响应，EMI 将被自动清零以除能其他中断，SIMF 请求标志也可自动清除。

LVD 中断

当低电压检测功能检测到一个低电压时，LVD 中断请求标志 LVF 置位，LVD 中断请求产生。若要程序跳转到 LVD 中断向量地址，总中断控制位 EMI、低电压中断使能位 LVE 和相关的多功能中断使能位需先被置位。当中断使能，堆栈未满且低电压条件发生时，可跳转至 LVD 中断向量子程序中执行。当低电压中断响应，EMI 将被自动清零以除能其他中断，LVF 请求标志也可自动清除。

EEPROM 中断

当写周期结束，EEPROM 中断请求标志 DEF 被置位，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、EEPROM 中断使能位 DEE 需先被置位。当中断使能，堆栈未满且 EEPROM 写周期结束时，可跳转至相应的 EEPROM 中断向量子程序中执行。当 EEPROM 中断响应，相应中断请求标志位 DEF 会自动清零且 EMI 位也会被清零以除能其它中断。

UART 中断

由几种 UART 条件来控制 UART 中断发生。当发送器为空、发送器空闲、接收器数据有效、接收器溢出、地址检测和 RX 引脚唤醒，UART 中断请求标志 UARTF 被置位，UART 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、UART 中断使能位 UARTE 需先被置位。当中断使能，堆栈未满且以上任何一种情况发生时，将调用 UART 中断向量子程序。当响应中断服务子程序时，EMI 位会被清零以除能其他中断，UARTF 标志位也自动清除。但是 USR 寄存器标志位只能通过 UART 的某些动作清楚，其中的细节在 UART 部分。

多功能中断

此系列单片机中多达两个多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，即 TM 中断。

当 MFnF 多功能中断请求标志位被置位，即包含在多功能中断内的任意一个中断发生请求时，多功能中断请求产生。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 MFnF 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位会自动复位，但触发此多功能中断的初始中断源的请求标志位不会自动复位，必须由应用程序清零。

TM 中断

标准型和周期型 TM 各有两个中断，分别来自比较器 P 和比较器 A 匹配，都属于多功能中断。每个 TM 各有两个中断请求标志位 xTMnPF 和 xTMnAF 及两个使能位 xTMnPE 和 xTMnAE。当 TM 比较器 P、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、相应的 TM 中断使能位和相关多功能中断使能位 MFnE 需先被置位。当中断使能，堆栈未满且 TM 比较器 P 或比较器 A 匹配情况发生时，可跳转至相关多功能中断向量程序执行。当 TM 中断响应，EMI 将被自动清零以除能其它中断，相关 MFnF 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零。

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或低电源电压供电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。若要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高。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 MFnF 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。

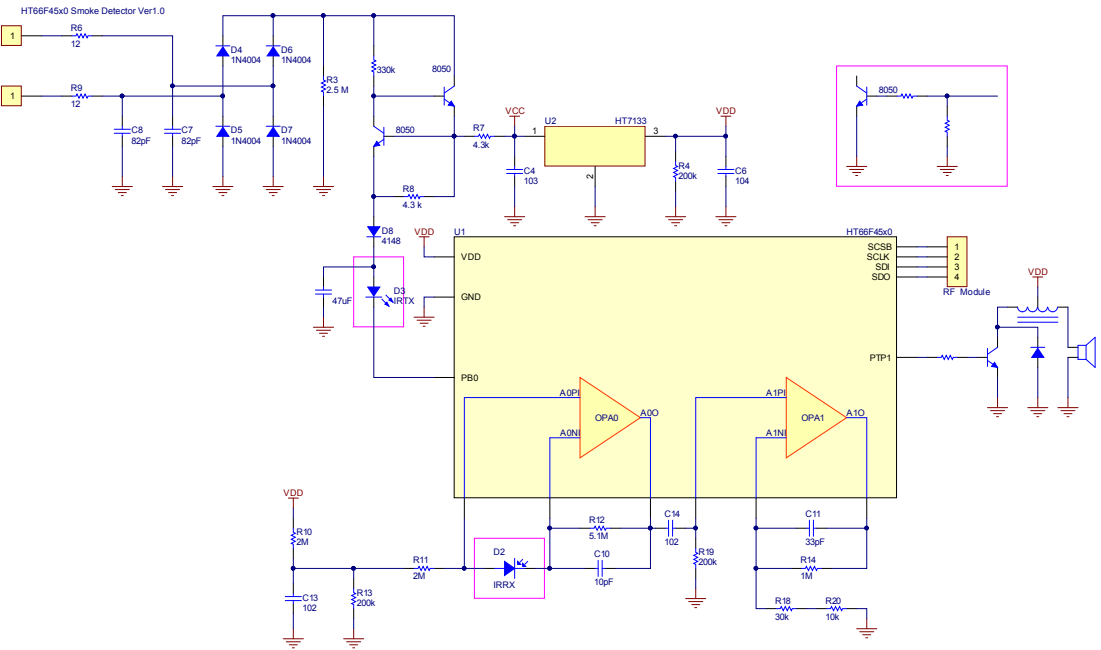
当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，因为累加器、状态寄存器或其它的寄存器的内容会被中断服务程序更改，应事先将这些数据应在中断服务程序执行前保存起来。若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清除 EMI 位，除能进一步中断。

配置选项

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

No.	Options
1	高速 RC 振荡器频率选择： 1. 2MHz 2. 4MHz 3. 8MHz

应用电路



指令集

简介

任何单片机成功运作的核心在于它的指令集，此指令集为一组程序指令码，用来指导单片机如何去执行指定的工作。在 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他们的应用。

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用几种 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 Holtek 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 Holtek 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 Holtek 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入口的 8 位数据，处理这些数据，然后再输出正确的新数据。这种读入 - 修改 - 写出的过程现在则被位运算指令所取代。

查表运算

数据的储存通常由寄存器完成，然而当处理大量固定的数据时，它的存储量常常造成对个别存储器的不便。为了改善此问题，Holtek 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

当要操作的数据存储器位于数据存储器 Sector 0 时，下表说明了与数据存储器存取有关的指令。

惯例

x: 立即数
m: 数据存储器地址
A: 累加器
i: 第 0~7 位
addr: 程序存储器地址

助记符	说明	指令周期	影响标志位
算术运算			
ADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	1	Z, C, AC, OV, SC
ADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	1 注	Z, C, AC, OV, SC
ADD A, x	ACC 与立即数相加，结果放入 ACC	1	Z, C, AC, OV, SC
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, SC
ADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	1 注	Z, C, AC, OV, SC
SUB A, x	ACC 与立即数相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
SUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
SUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	1 注	Z, C, AC, OV, SC, CZ
SBC A, x	ACC 与立即数、进位标志相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
SBC A,[m]	ACC 与数据存储器、进位标志相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
SBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	1 注	Z, C, AC, OV, SC, CZ
DAA [m]	将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器	1 注	C
逻辑运算			
AND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	1	Z
OR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	1	Z
XOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	1	Z
ANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	1 注	Z
ORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	1 注	Z
XORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	1 注	Z
AND A, x	ACC 与立即数做“与”运算，结果放入 ACC	1	Z
OR A, x	ACC 与立即数做“或”运算，结果放入 ACC	1	Z
XOR A, x	ACC 与立即数做“异或”运算，结果放入 ACC	1	Z
CPL [m]	对数据存储器取反，结果放入数据存储器	1 注	Z
CPLA [m]	对数据存储器取反，结果放入 ACC	1	Z
递增和递减			
INCA [m]	递增数据存储器，结果放入 ACC	1	Z
INC [m]	递增数据存储器，结果放入数据存储器	1 注	Z
DECA [m]	递减数据存储器，结果放入 ACC	1	Z
DEC [m]	递减数据存储器，结果放入数据存储器	1 注	Z

助记符	说明	指令周期	影响标志位
移位			
RRA [m]	数据存储器右移一位，结果放入 ACC	1	无
RR [m]	数据存储器右移一位，结果放入数据存储器	1 ^注	无
RRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	1	C
RRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	1 ^注	C
RLA [m]	数据存储器左移一位，结果放入 ACC	1	无
RL [m]	数据存储器左移一位，结果放入数据存储器	1 ^注	无
RLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	1	C
RLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	1 ^注	C
数据传送			
MOV A,[m]	将数据存储器送至 ACC	1	无
MOV [m],A	将 ACC 送至数据存储器	1 ^注	无
MOV A, x	将立即数送至 ACC	1	无
位运算			
CLR [m].i	清除数据存储器的位	1 ^注	无
SET [m].i	置位数据存储器的位	1 ^注	无
转移			
JMP addr	无条件跳转	2	无
SZ [m]	如果数据存储器为零，则跳过下一条指令	1 ^注	无
SZA [m]	数据存储器送至 ACC，如果内容为零，则跳过下一条指令	1 ^注	无
SNZ [m]	如果数据存储器不为零，则跳过下一条指令	1 ^注	无
SZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	1 ^注	无
SNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	1 ^注	无
SIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	1 ^注	无
SDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	1 ^注	无
SIZA [m]	递增数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	1 ^注	无
SDZA [m]	递减数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	1 ^注	无
CALL addr	子程序调用	2	无
RET	从子程序返回	2	无
RET A, x	从子程序返回，并将立即数放入 ACC	2	无
RETI	从中断返回	2	无
查表			
TABRD [m]	读取特定页的 ROM 内容，并送至数据存储器 and TBLH	2 ^注	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器 and TBLH	2 ^注	无
ITABRD [m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器 and TBLH	2 ^注	无
ITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器 and TBLH	2 ^注	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 ^注	无
SET [m]	置位数据存储器	1 ^注	无
CLR WDT	清除看门狗定时器	1	TO, PDF

助记符	说明	指令周期	影响标志位
SWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	1 注	无
SWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	1	无
HALT	进入暂停模式	1	TO, PDF

注: 1. 对跳转指令而言，如果比较的结果牵涉到跳转即需 2 个周期，如果没有发生跳转，则只需一个周期。

2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

扩展指令集

扩展指令用来提供更大范围的数据存储器寻址。当被存取的数据存储器位于 Sector 0 之外的任何数据存储器 Sector，扩展指令可直接存取数据存储器而无需使用间接寻址，此举不仅可节省 Flash 存储器空间的使用，同时可提高 CPU 执行效率。

助记符	说明	指令周期	影响标志位
算术运算			
LADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	2 注	Z, C, AC, OV, SC
LADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	2 注	Z, C, AC, OV, SC
LSUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	2 注	Z, C, AC, OV, SC, CZ
LSBC A,[m]	ACC 与数据存储器、进位标志相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	2 注	Z, C, AC, OV, SC, CZ
LDAA [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	2 注	C
逻辑运算			
LAND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	2	Z
LOR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	2	Z
LXOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	2	Z
LANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	2 注	Z
LORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	2 注	Z
LXORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	2 注	Z
LCPL [m]	对数据存储器取反，结果放入数据存储器	2 注	Z
LCPLA [m]	对数据存储器取反，结果放入 ACC	2	Z
递增和递减			
LINCA [m]	递增数据存储器，结果放入 ACC	2	Z
LINC [m]	递增数据存储器，结果放入数据存储器	2 注	Z
LDECA [m]	递减数据存储器，结果放入 ACC	2	Z
LDEC [m]	递减数据存储器，结果放入数据存储器	2 注	Z
移位			
LRRA [m]	数据存储器右移一位，结果放入 ACC	2	无
LRR [m]	数据存储器右移一位，结果放入数据存储器	2 注	无
LRRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	2	C
LRRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	2 注	C
LRLA [m]	数据存储器左移一位，结果放入 ACC	2	无
LRL [m]	数据存储器左移一位，结果放入数据存储器	2 注	无
LRLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	2	C
LRLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	2 注	C
数据传送			
LMOV A,[m]	将数据存储器送至 ACC	2	无
LMOV [m],A	将 ACC 送至数据存储器	2 注	无

助记符	说明	指令周期	影响标志位
位运算			
LCLR [m].i	清除数据存储器的位	2 注	无
LSET [m].i	置位数据存储器的位	2 注	无
转移			
LSZ [m]	如果数据存储为零，则跳过下一条指令	2 注	无
LSZA [m]	数据存储器送至 ACC，如果内容为零，则跳过下一条指令	2 注	无
LSNZ [m]	如果数据存储器不为零，则跳过下一条指令	2 注	无
LSZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	2 注	无
LSNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	2 注	无
LSIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	2 注	无
LSDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	2 注	无
LSIZA [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	2 注	无
LSDZA [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	2 注	无
查表			
LTABRD [m]	读取特定页的 ROM 内容，并送至数据存储器 and TBLH	3 注	无
LTABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器 and TBLH	3 注	无
LITABRD [m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器 and TBLH	3 注	无
LITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器 and TBLH	3 注	无
其它指令			
LCLR [m]	清除数据存储器	2 注	无
LSET [m]	置位数据存储器	2 注	无
LSWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	2 注	无
LSWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	2	无

注：1. 对扩展跳转指令而言，如果比较的结果牵涉到跳转即需 3 个周期，如果没有发生跳转，则只需两个周期。
2. 任何扩展指令若要改变 PCL 的内容将需要 3 个周期来执行。

指令定义

ADC A, [m]	Add Data Memory to ACC with Carry
指令说明	将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
ADCM A, [m]	Add ACC to Data Memory with Carry
指令说明	将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
ADD A, [m]	Add Data Memory to ACC
指令说明	将指定的数据存储器和累加器内容相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
ADD A, x	Add immediate data to ACC
指令说明	将累加器和立即数相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + x$
影响标志位	OV、Z、AC、C、SC
ADDM A, [m]	Add ACC to Data Memory
指令说明	将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
AND A, [m]	Logical AND Data Memory to ACC
指令说明	将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "AND" } [m]$
影响标志位	Z

AND A, x	Logical AND immediate data to ACC
指令说明	将累加器中的数据和立即数做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "AND" } x$
影响标志位	Z
ANDM A, [m]	Logical AND ACC to Data Memory
指令说明	将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。
功能表示	$[m] \leftarrow ACC \text{ "AND" } [m]$
影响标志位	Z
CALL addr	Subroutine call
指令说明	无条件地调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定地址并从新地址继续执行程序，由于此指令需要额外的运算，所以为一个 2 周期的指令。
功能表示	$Stack \leftarrow Program\ Counter + 1$ $Program\ Counter \leftarrow addr$
影响标志位	无
CLR [m]	Clear Data Memory
指令说明	将指定数据存储器的内容清零。
功能表示	$[m] \leftarrow 00H$
影响标志位	无
CLR [m].i	Clear bit of Data Memory
指令说明	将指定数据存储器的第 i 位内容清零。
功能表示	$[m].i \leftarrow 0$
影响标志位	无
CLR WDT	Clear Watchdog Timer
指令说明	WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。
功能表示	WDT cleared $TO \ \& \ PDF \leftarrow 0$
影响标志位	TO、PDF

CPL [m]	Complement Data Memory
指令说明	将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1。
功能表示	$[m] \leftarrow \overline{[m]}$
影响标志位	Z
CPLA [m]	Complement Data Memory with result in ACC
指令说明	将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。
功能表示	$ACC \leftarrow \overline{[m]}$
影响标志位	Z
DAA [m]	Decimal-Adjust ACC for addition with result in Data Memory
指令说明	将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。
功能表示	$[m] \leftarrow ACC + 00H$ 或 $[m] \leftarrow ACC + 06H$ 或 $[m] \leftarrow ACC + 60H$ 或 $[m] \leftarrow ACC + 66H$
影响标志位	C
DEC [m]	Decrement Data Memory
指令说明	将指定数据存储器内容减 1。
功能表示	$[m] \leftarrow [m] - 1$
影响标志位	Z
DECA [m]	Decrement Data Memory with result in ACC
指令说明	将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。
功能表示	$ACC \leftarrow [m] - 1$
影响标志位	Z

HALT	Enter power down mode
指令说明	此指令终止程序执行并关掉系统时钟，RAM 和寄存器的内容保持原状态，WDT 计数器和分频器被清“0”，暂停标志位 PDF 被置位 1，WDT 溢出标志位 TO 被清 0。
功能表示	$TO \leftarrow 0$ $PDF \leftarrow 1$
影响标志位	TO、PDF
INC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 1。
功能表示	$[m] \leftarrow [m] + 1$
影响标志位	Z
INCA [m]	Increment Data Memory with result in ACC
指令说明	将指定数据存储器的内容加 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
JMP addr	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
MOV A, [m]	Move Data Memory to ACC
指令说明	将指定数据存储器的内容复制到累加器。
功能表示	$ACC \leftarrow [m]$
影响标志位	无
MOV A, x	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

MOV [m], A	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	$[m] \leftarrow \text{ACC}$
影响标志位	无
NOP	No operation
指令说明	空操作，接下来顺序执行下一条指令。
功能表示	$\text{PC} \leftarrow \text{PC} + 1$
影响标志位	无
OR A, [m]	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	$\text{ACC} \leftarrow \text{ACC} \text{ "OR" } [m]$
影响标志位	Z
OR A, x	Logical OR immediate data to ACC
指令说明	将累加器中的数据和立即数逻辑或，结果存放到累加器。
功能表示	$\text{ACC} \leftarrow \text{ACC} \text{ "OR" } x$
影响标志位	Z
ORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器中的数据和累加器逻辑或，结果放到数据存储器。
功能表示	$[m] \leftarrow \text{ACC} \text{ "OR" } [m]$
影响标志位	Z
RET	Return from subroutine
指令说明	将堆栈寄存器中的程序计数器值恢复，程序由取回的地址继续执行。
功能表示	$\text{Program Counter} \leftarrow \text{Stack}$
影响标志位	无
RET A, x	Return from subroutine and load immediate data to ACC
指令说明	将堆栈寄存器中的程序计数器值恢复且累加器载入指定的立即数，程序由取回的地址继续执行。
功能表示	$\text{Program Counter} \leftarrow \text{Stack}$ $\text{ACC} \leftarrow x$
影响标志位	无

RETI	Return from interrupt
指令说明	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被响应，则这个中断将在返回主程序之前被响应。
功能表示	Program Counter \leftarrow Stack EMI \leftarrow 1
影响标志位	无
RL [m]	Rotate Data Memory left
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。
功能表示	[m].(i+1) \leftarrow [m].i (i=0~6) [m].0 \leftarrow [m].7
影响标志位	无
RLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 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=0~6) ACC.0 \leftarrow [m].7
影响标志位	无
RLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 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=0~6) [m].0 \leftarrow C C \leftarrow [m].7
影响标志位	C
RLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 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=0~6) ACC.0 \leftarrow C C \leftarrow [m].7
影响标志位	C

RR [m]	Rotate Data Memory right
指令说明	将指定数据存储器的内容循环右移 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
RRA [m]	Rotate Data Memory right with result in ACC
指令说明	将指定数据存储器的内容循环右移 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
RRC [m]	Rotate Data Memory right through Carry
指令说明	将指定数据存储器的内容连同进位标志右移 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
RRCA [m]	Rotate Data Memory right through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志右移 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
SBC A, [m]	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

SBC A, x	Subtract immediate data from ACC with Carry
指令说明	将累加器减去立即数以及进位标志的反，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
SBCM A, [m]	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
SDZ [m]	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
SDZA [m]	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
SET [m]	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

SET [m].i	Set bit of Data Memory
指令说明	将指定数据存储器的第 i 位置位为 1。
功能表示	$[m].i \leftarrow 1$
影响标志位	无
SIZ [m]	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
SIZA [m]	Skip if increment Data Memory is zero with result in ACC
指令说明	将指定数据存储器的内容加 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
SNZ [m].i	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
SNZ [m]	Skip if Data Memory is not 0
指令说明	判断指定存储器，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 $[m] \neq 0$ ，跳过下一条指令执行
影响标志位	无

SUB A, [m]	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
SUBM A, [m]	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
SUB A, x	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
SWAP [m]	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
SWAPA [m]	Swap nibbles of Data Memory with result in ACC
指令说明	将指定数据存储器的低 4 位与高 4 位互相交换，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
SZ [m]	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

SZA [m] 指令说明	Skip if Data Memory is 0 with data movement to ACC 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 ← [m]，如果 [m]=0，跳过下一条指令执行
影响标志位	无
SZ [m].i 指令说明	Skip if bit i of Data Memory is 0 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
TABRD [m] 指令说明	Read table (specific page) to TBLH and Data Memory 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] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
TABRDL [m] 指令说明	Read table (last page) to TBLH and Data Memory 将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
ITABRD [m] 指令说明	Increment table pointer low byte first and read table (specific page) to TBLH and data memory 自加表格指针低字节 TBLP，将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

ITABRDL [m]	Increment table pointer low byte first and read table (last page) to TBLH and data memory
指令说明	自加表格指针低字节 TBLP，将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	$[m] \leftarrow \text{程序代码 (低字节)}$ $TBLH \leftarrow \text{程序代码 (高字节)}$
影响标志位	无
XOR A, [m]	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	$ACC \leftarrow ACC \text{ "XOR" } [m]$
影响标志位	Z
XORM A, [m]	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	$[m] \leftarrow ACC \text{ "XOR" } [m]$
影响标志位	Z
XOR A, x	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	$ACC \leftarrow ACC \text{ "XOR" } x$
影响标志位	Z

扩展指令定义

扩展指令被用来直接存取存储在任何数据存储器 Sector 中的数据。

LADC A, [m]	Add Data Memory to ACC with Carry
指令说明	将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
LADCM A, [m]	Add ACC to Data Memory with Carry
指令说明	将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
LADD A, [m]	Add Data Memory to ACC
指令说明	将指定的数据存储器和累加器内容相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
LADDM A, [m]	Add ACC to Data Memory
指令说明	将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
LAND A, [m]	Logical AND Data Memory to ACC
指令说明	将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ “AND” } [m]$
影响标志位	Z
LANDM A, [m]	Logical AND ACC to Data Memory
指令说明	将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。
功能表示	$[m] \leftarrow ACC \text{ “AND” } [m]$
影响标志位	Z

LCLR [m]	Clear Data Memory
指令说明	将指定数据存储器的内容清零。
功能表示	$[m] \leftarrow 00H$
影响标志位	无
LCLR [m].i	Clear bit of Data Memory
指令说明	将指定数据存储器的第 i 位内容清零。
功能表示	$[m].i \leftarrow 0$
影响标志位	无
LCPL [m]	Complement Data Memory
指令说明	将指定数据存储器中的每一位取逻辑反， 相当于从 1 变 0 或 0 变 1。
功能表示	$[m] \leftarrow \overline{[m]}$
影响标志位	Z
LCPLA [m]	Complement Data Memory with result in ACC
指令说明	将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，结果被存放回累加器且数据寄存器的内容保持 不变。
功能表示	$ACC \leftarrow \overline{[m]}$
影响标志位	Z
LDAA [m]	Decimal-Adjust ACC for addition with result in Data Memory
指令说明	将累加器中的内容转换为 BCD (二进制转成十进制) 码。 如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执 行对低四位加“6”，否则低四位保持不变；如果高四位的 值大于“9”或 C=1，那么 BCD 调整就执行对高四位加“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。
功能表示	$[m] \leftarrow ACC + 00H$ 或 $[m] \leftarrow ACC + 06H$ 或 $[m] \leftarrow ACC + 60H$ 或 $[m] \leftarrow ACC + 66H$
影响标志位	C

LDEC [m]	Decrement Data Memory
指令说明	将指定数据存储器的内容减 1。
功能表示	$[m] \leftarrow [m] - 1$
影响标志位	Z
LDECA [m]	Decrement Data Memory with result in ACC
指令说明	将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。
功能表示	$ACC \leftarrow [m] - 1$
影响标志位	Z
LINC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 1。
功能表示	$[m] \leftarrow [m] + 1$
影响标志位	Z
LINCA [m]	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
LMOV A, [m]	Move Data Memory to ACC
指令说明	将指定数据存储器的内容复制到累加器中。
功能表示	$ACC \leftarrow [m]$
影响标志位	无
LMOV [m], A	Move ACC to Data Memory
指令说明	将累加器的内容复制到指定数据存储器。
功能表示	$[m] \leftarrow ACC$
影响标志位	无
LOR A, [m]	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

LORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器中的数据 and 累加器逻辑或，结果放到数据存储器。
功能表示	$[m] \leftarrow \text{ACC} \text{ "OR" } [m]$
影响标志位	Z
LRL [m]	Rotate Data Memory left
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。
功能表示	$[m].(i+1) \leftarrow [m].i \ (i=0\sim6)$ $[m].0 \leftarrow [m].7$
影响标志位	无
LRLA [m]	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	$\text{ACC}.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $\text{ACC}.0 \leftarrow [m].7$
影响标志位	无
LRLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 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
LRLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 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	$\text{ACC}.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $\text{ACC}.0 \leftarrow C$ $C \leftarrow [m].7$
影响标志位	C

LRR [m]	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
LRRA [m]	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
LRRC [m]	Rotate Data Memory right through Carry
指令说明	将指定数据存储器的内容连同进位标志右移 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
LRRCA [m]	Rotate Data Memory right through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志右移 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
LSBC A, [m]	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

LSBCM A, [m]	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
LSDZ [m]	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 3 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
LSDZA [m]	Skip if decrement Data Memory is zero with result in ACC
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 3 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
LSET [m]	Set Data Memory
指令说明	将指定数据存储器的每一个位置位为 1。
功能表示	$[m] \leftarrow FFH$
影响标志位	无
LSET [m].i	Set bit of Data Memory
指令说明	将指定数据存储器的第 i 位置位为 1。
功能表示	$[m].i \leftarrow 1$
影响标志位	无

LSIZ [m] 指令说明	Skip if increment Data Memory is 0 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 3 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
LSIZA [m] 指令说明	Skip if increment Data Memory is zero with result in ACC 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 3 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
LSNZ [m].i 指令说明	Skip if bit i of Data Memory is not 0 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 3 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
LSNZ [m] 指令说明	Skip if Data Memory is not 0 判断指定数据存储器，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 3 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 $[m] \neq 0$ ，跳过下一条指令执行
影响标志位	无
LSUB A, [m] 指令说明	Subtract Data Memory from ACC 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

LSUBM A, [m]	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
LSWAP [m]	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
LSWAPA [m]	Swap nibbles of Data Memory with result in ACC
指令说明	将指定数据存储器的低 4 位和高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。
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
LSZ [m]	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 3 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
LSZA [m]	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 3 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

LSZ [m].i	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 3 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
LTABRD [m]	Move the ROM code (specific page) to TBLH and data memory
指令说明	将表格针对 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] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LTABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRD [m]	Increment table pointer low byte first and read table (specific page) to TBLH and data memory
指令说明	自加表格指针低字节 TBLP，将表格针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRDL [m]	Increment table pointer low byte first and read table (last page) to TBLH and data memory
指令说明	自加表格指针低字节 TBLP，将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

LXOR A, [m]	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	$ACC \leftarrow ACC \text{ "XOR" } [m]$
影响标志位	Z
 LXORM A, [m]	 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	$[m] \leftarrow ACC \text{ "XOR" } [m]$
影响标志位	Z

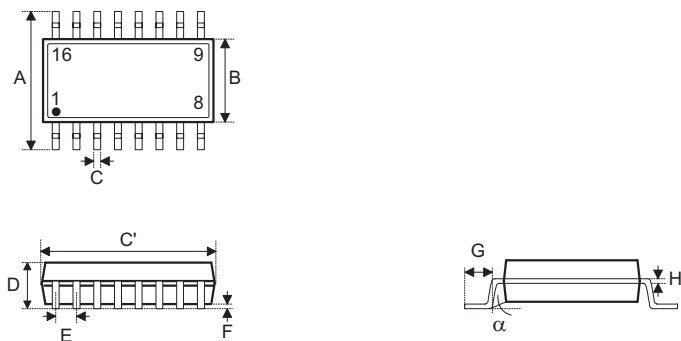
封装信息

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

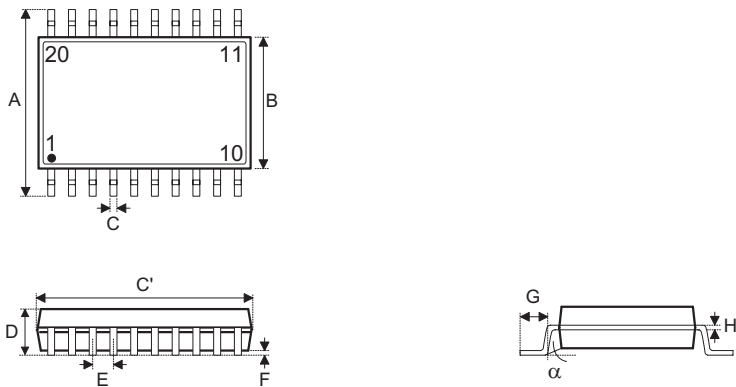
16-pin NSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小	正常	最大
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
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C'	—	9.90 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

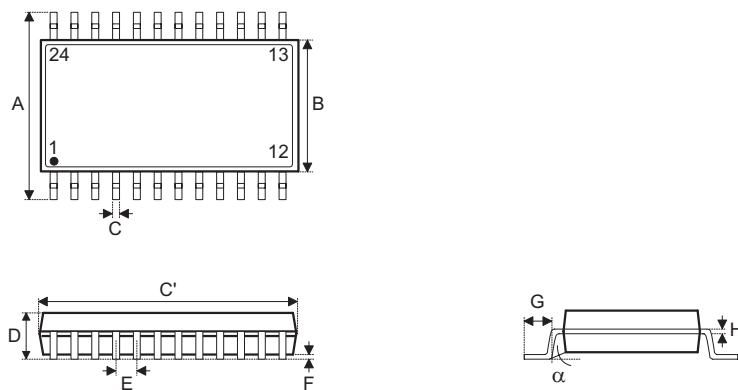
20-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小	正常	最大
A	—	0.236 BSC	—
B	—	0.155 BSC	—
C	0.008	—	0.012
C'	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.05
H	0.004	—	0.01
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.20	—	0.30
C'	—	8.660 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

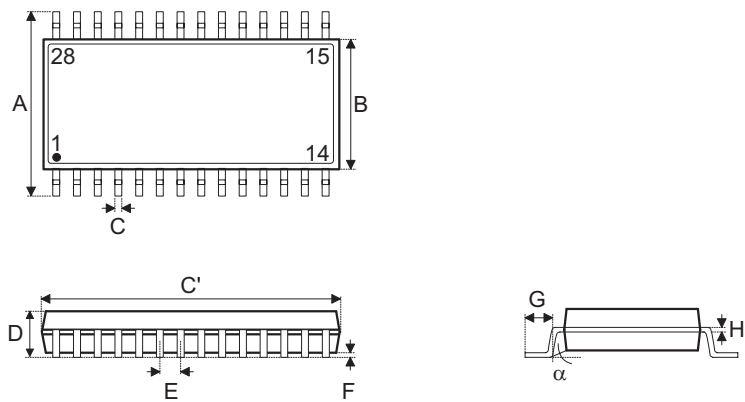
24-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小	正常	最大
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
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.20	—	0.30
C'	—	8.660 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

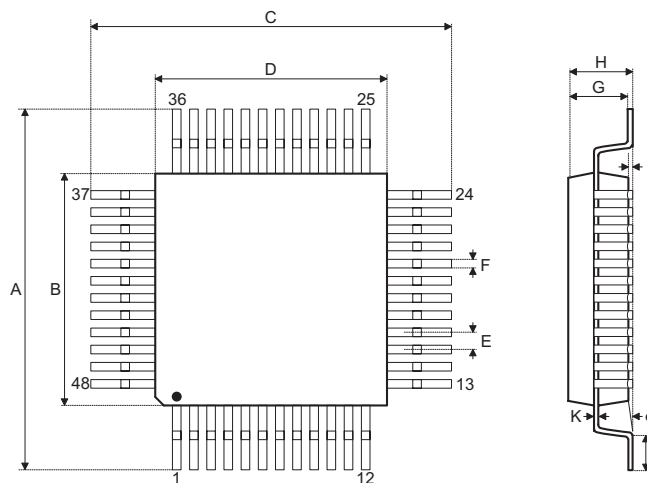
28-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小	正常	最大
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C'	—	0.390 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	6.000 BSC	—
B	—	3.900 BSC	—
C	0.20	—	0.30
C'	—	9.900 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

48-pin LQFP (7mm × 7mm) 外形尺寸



符号	尺寸 (单位: inch)		
	最小	正常	最大
A	—	0.354 BSC	—
B	—	0.276 BSC	—
C	—	0.354 BSC	—
D	—	0.276 BSC	—
E	—	0.020 BSC	—
F	0.007	0.009	0.011
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	9.000 BSC	—
B	—	7.000 BSC	—
C	—	9.000 BSC	—
D	—	7.000 BSC	—
E	—	0.500 BSC	—
F	0.170	0.220	0.270
G	1.350	1.400	1.450
H	—	—	1.600
I	0.050	—	0.150
J	0.450	0.600	0.750
K	0.090	—	0.200
α	0°	—	7°

Copyright© 2020 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 Holtek 对于说明书的使用不负任何责任。文中提到的应用目的仅仅是用来做说明，Holtek 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。Holtek 产品不授权使用于救生、维生从机或系统中做为关键从机。Holtek 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/>.