带 OCDS 的 Sub-1GHz RF 发射器 Flash 单片机

BC68F2130/BC68F2140/BC68F2150

版本 : V1.60　日期 : 2022-08-02

www.holtek.com

目 录

特性 5
CPU 特性 5
周边特性 5
RF 发射器特性 5
概述 6
方框图 7
选型表 7
引脚图 8
引脚说明 9
极限参数 12
直流电气特性 12
交流电气特性 13
LVD & LVR 电气特性 14
RF 发射器电气特性 15
上电复位特性 16
系统结构 16
时序和流水线结构 16
程序计数器 17
堆栈 18
算术逻辑单元 – ALU 18
Flash 程序存储器 19
结构 19
特殊向量 19
查表 19
查表范例 20
在线烧录 – ICP 21
片上调试 – OCDS 21
在应用烧录 – IAP 22
数据存储器 32
结构 32
数据存储器寻址 33
通用数据存储器 33
特殊功能数据存储器 33
特殊功能寄存器 35
间接寻址寄存器 – IAR0, IAR1, IAR2 35
存储器指针 – MP0, MP1H/MP1L, MP2H/MP2L 35
累加器 – ACC 36
程序计数器低字节寄存器 – PCL 37
查表寄存器 – TBLP, TBHP, TBLH 37
状态寄存器 – STATUS 37

振荡器 39
振荡器概述 39
系统时钟配置 39
外部晶体 / 陶瓷振荡器 – HXT 40
内部高速 RC 振荡器 – HIRC 40
内部 32kHz 振荡器 – LIRC 40
工作模式和系统时钟 41
系统时钟 41
系统工作模式 42
控制寄存器 43
工作模式切换 46
待机电流的注意事项 50
唤醒 50
看门狗定时器 52
看门狗定时器时钟源 52
看门狗定时器控制寄存器 52
看门狗定时器操作 53
复位和初始化 54
复位功能 54
复位初始状态 57
输入 / 输出端口 61
上拉电阻 61
PA 口唤醒 62
输入 / 输出端口控制寄存器 62
引脚共用功能 63
输入 / 输出引脚结构 64
编程注意事项 65
定时器模块 – TM 66
简介 66
TM 操作 66
TM 时钟源 66
TM 中断 66
TM 外部引脚 67
TM 输入 / 输出引脚选择 67
编程注意事项 68
简易型 TM – CTM 69
简易型 TM 操作 69
简易型 TM 寄存器介绍 69
简易型 TM 工作模式 73
周期型 TM – PTM 79
周期型 TM 操作 79
周期型 TM 寄存器介绍 79
周期型 TM 工作模式 83

RF 发射器 91
RF 发射器缩写注意事项 91
RF 发射器控制寄存器 91
调制模式和工作模式选择 98
突发模式下的 TX FIFO 模式 100
RF 通道设置 102
软件编程指南 104
中断 106
中断寄存器 106
中断操作 110
外部中断 111
时基中断 111
多功能中断 113
RF TX FIFO 长度阈值检测中断 113
RF 突发模式传送完成中断 114
LVD 中断 114
TM 中断 114
中断唤醒功能 114
编程注意事项 115
低电压检测 – LVD 115
LVD 寄存器 115
LVD 操作 116
配置选项 116
应用电路 117
指令集 118
简介 118
指令周期 118
数据的传送 118
算术运算 118
逻辑和移位运算 118
分支和控制转换 119
位运算 119
查表运算 119
其它运算 119
指令集概要 120
惯例 120
扩展指令集 123
指令定义 125
扩展指令定义 137
封装信息 147
16-pin NSOP-EP (150mil) 外形尺寸 148
24-pin SSOP-EP (150mil) 外形尺寸 149
SAW Type 16-pin QFN (4mm×4mm×0.75mm) 外形尺寸 150
SAW Type 24-pin QFN (4mm×4mm×0.75mm) 外形尺寸 151

特性

CPU 特性

- 工作电压
  - ♦ f_{SYS}=16MHz：2.0V~3.6V
- V_{DD}=5V，系统时钟为 16MHz 时，指令周期为 0.25μs
- 提供省电和唤醒功能，以降低功耗
- 振荡器类型
  - ♦ RF 外部高速晶振 – HXT
  - ♦ 内部高速 RC – HIRC
  - ♦ 内部低速 32kHz RC – LIRC
- 多种工作模式：正常、低速、空闲、休眠和深度休眠
- 内部集成的振荡器，无需外接元件
- 所有指令都可在 1 到 3 个指令周期内完成
- 查表指令
- 115 条指令
- 多达 8 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：2K×16 ~ 8K×16
- RAM 数据存储器：256×8
- 支持在应用烧录功能 – IAP
- 看门狗定时器功能
- 多达 14 个双向 I/O 口
- 2 个与 I/O 口复用的外部中断输入
- 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
  - ♦ 1 个 10-bit 简易型 TM – CTM
  - ♦ 1 个 10-bit 周期性 TM – PTM
- 双时基功能，用于产生固定时间的中断信号
- 内建 1.5V LDO
- 低电压复位功能
- 低电压检测功能
- Flash 程序存储器可重复烧录达 100,000 次
- Flash 程序存储器数据可保存 10 年以上
- 封装类型：16-pin NSOP-EP/QFN，24-pin SSOP-EP/QFN

RF 发射器特性

- 带传送相位低噪声的完整超高频 OOK/FSK 发射器
- 支持 315/433/868/915MHz 频带
- 小于 2kHz 分辨率的可编程通道设置

- OOK 符号传送速率 0.5Ksps~25Ksps，FSK 数据传送速率 0.5Kbps~100Kbps
- 输出功率高达 13dBm (可软件控制输出功率：0，+10dBm，+13dBm)
- RF 外部晶振：16MHz

概述

该系列 Sub-1GHz RF 发射器单片机提供一个功能齐全的单片机和 RF 发射器功能，使其灵活地应用于一系列无线输入 / 输出控制如工业控制、消费类产品和子系统控制器等。

该系列单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了较大的方便。存储器方面，除了 Flash 程序存储器，还包含了一个可用于存储序列号、校准数据等非易失性数据的 RAM 数据存储器。通过使用 Holtek 在线应用编程功能，用户可方便直接地将测量数据存储在 Flash 存储器中，且方便对应用程序进行更新。

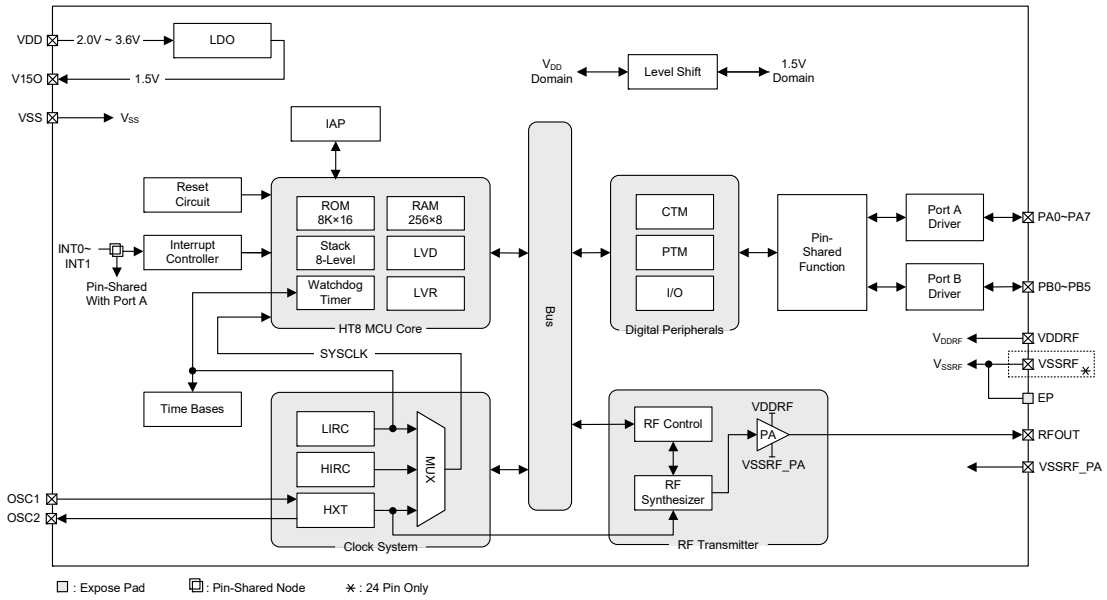
该单片机还带有多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生功能。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

该单片机提供了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。其在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。

集成的 RF 发射器可工作在 315MHz、433MHz、868MHz 和 915MHz 频带。创建一个完整通用的 RF 发射器系统仅需一个额外的晶振和少量的外部元器件。该系列单片机还提供了内部功率放大器，可为 50Ω 负载提供高达 +13dBm 的功率。这样的功率水平可使一个小型的发射器在接近最大传输规限条件下仍可工作。该系列单片机接收类型有 OOK—On-Off Keying (开关键控) 和 FSK—Frequency Shift Keying (频移键控) 两种。FSK 数据传送速率高达 100Kbps，允许单片机支持更复杂的控制协议。

外加 I/O 使用灵活和时基功能等其它特性，确保该系列单片机拥有足够能力提供低成本，高效益的单片机 Sub-1GHz RF 发射器方案应用于无线远程。

方框图



注：该图为设备最大型号的方框图，选型表中提供了设备型号间的功能差异。

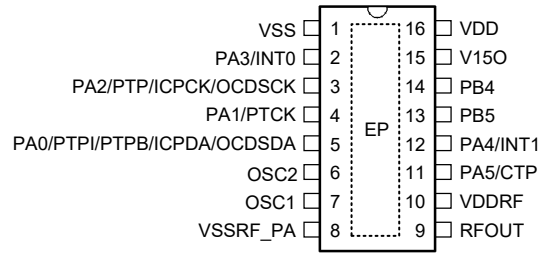
选型表

对此系列的芯片而言，大多数的特性参数都是一样的。主要差异在于程序存储器的容量、I/O 口数目和封装类型。下表列出了各单片机的主要特性。

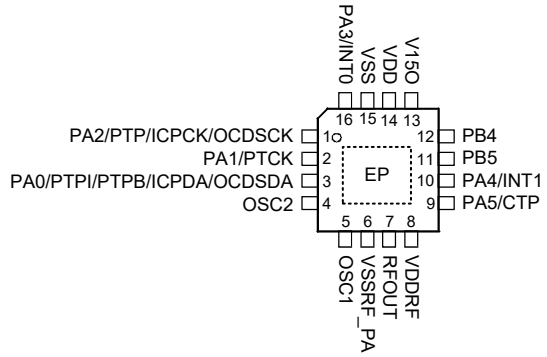
型号	V_{DD}	ROM	RAM	I/O	外部中断	LDO	TM 模块
BC68F2130	2.0V~3.6V	2K×16	256×8	8	2	√	10-bit CTM×1 10-bit PTM×1
BC68F2140	2.0V~3.6V	4K×16	256×8	14	2	√	10-bit CTM×1 10-bit PTM×1
BC68F2150	2.0V~3.6V	8K×16	256×8	14	2	√	10-bit CTM×1 10-bit PTM×1

型号	集成 RF		时基	堆栈	封装
	频带	类型			
BC68F2130	315~915MHz	OOK/FSK TX	2	8	16NSOP-EP/QFN
BC68F2140	315~915MHz	OOK/FSK TX	2	8	24SSOP-EP/QFN
BC68F2150	315~915MHz	OOK/FSK TX	2	8	24SSOP-EP/QFN

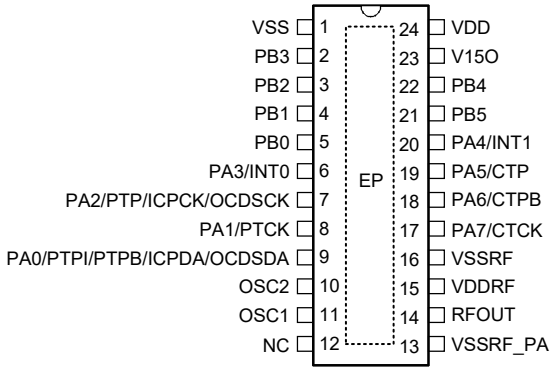
引脚图



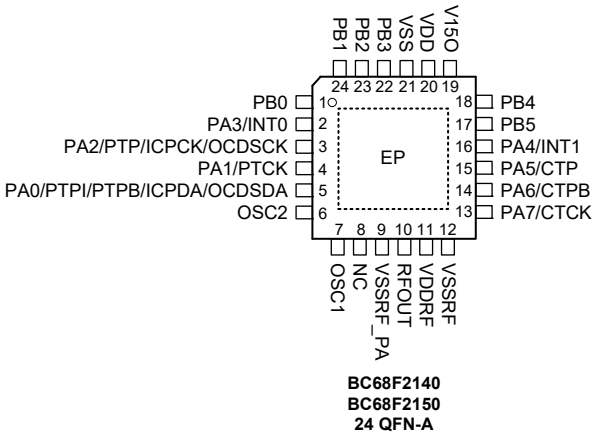
BC68F2130
16 NSOP-EP-A



BC68F2130
16 QFN-A



BC68F2140/BC68F2150
24 SSOP-EP-A



注：1. 若共用脚同时有多种输出，所需引脚共用功能通过引脚共用寄存器中相应的软件控制位控制。

2. OCDSDA 和 OCDSCK 引脚为 OCDS 专用引脚。

引脚说明

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。

BC68F2130

引脚名称	功能	OPT	I/T	O/T	说明
PA0/PTPI/PTPB/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
	PTPI	PAS0	ST	—	PTM 捕捉输入
	PTPB	PAS0	—	CMOS	PTM 反相输出
	ICPDA	—	ST	CMOS	ICP 地址 / 数据
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址
PA1/PTCK	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
	PTCK	PAS0	ST	—	PTM 时钟输入
PA2/PTP/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
	PTP	PAS0	—	CMOS	PTM 输出
	ICPCK	—	ST	—	ICP 时钟
	OCDSCK	—	ST	—	OCDS 时钟
PA3/INT0	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
	INT0	PAS0 INTEG INTC0	ST	—	外部中断 0 输入

引脚名称	功能	OPT	I/T	O/T	说明
PA4/INT1	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
	INT1	PAS1 INTEG INTC0	ST	—	外部中断 1 输入
PA5/CTP	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
	CTP	PAS1	—	CMOS	CTM 输出
PB4~PB5	PBn	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
OSC1	OSC1	—	HXT	—	RF 振荡器输入
OSC2	OSC2	—	—	HXT	RF 振荡器输出
RFOUT	RFOUT	—	—	AN	RF 功率放大器输出
VDD	VDD	—	PWR	—	3.3V 正电源
V15O	V15O	—	PWR	—	1.5V 内部 LDO 输出
VSS	VSS	—	PWR	—	数字负电源
VDDRF	VDDRF	—	PWR	—	RF 正电源
VSSRF_PA	VSSRF_ PA	—	PWR	—	RF 功率放大器负电源
EP	VSS	—	PWR	—	裸露焊盘，必须接地

注：I/T：输入类型；　O/T：输出类型；　OPT：通过寄存器选择；
　　PWR：电源；　ST：施密特触发输入；　CMOS：CMOS 输出；
　　AN：模拟信号；　HXT：高频晶体振荡器

BC68F2140/BC68F2150

引脚名称	功能	OPT	I/T	O/T	说明
PA0/PTPI/ PTPB/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
	PTPI	PAS0	ST	—	PTM 捕捉输入
	PTPB	PAS0	—	CMOS	PTM 反相输出
	ICPDA	—	ST	CMOS	ICP 地址 / 数据
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址
PA1/PTCK	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
	PTCK	PAS0	ST	—	PTM 时钟输入
PA2/PTP/ 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
	PTP	PAS0	—	CMOS	PTM 输出
	ICPCK	—	ST	—	ICP 时钟
	OCDSCK	—	ST	—	OCDS 时钟

引脚名称	功能	OPT	I/T	O/T	说明
PA3/INT0	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
	INT0	PAS0 INTEG INTC0	ST	—	外部中断 0 输入
PA4/INT1	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
	INT1	PAS1 INTEG INTC0	ST	—	外部中断 1 输入
PA5/CTP	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
	CTP	PAS1	—	CMOS	CTM 输出
PA6/CTPB	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
	CTPB	PAS1	—	CMOS	CTM 反相输出
PA7/CTCK	PA7	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	CTCK	—	ST	—	CTM 时钟输入
PB0~PB5	PBn	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
OSC1	OSC1	—	HXT	—	RF 振荡器输入
OSC2	OSC2	—	—	HXT	RF 振荡器输出
RFOUT	RFOUT	—	—	AN	RF 功率放大器输出
VDD	VDD	—	PWR	—	3.3V 正电源
V15O	V15O	—	PWR	—	1.5V 内部 LDO 输出
VSS	VSS	—	PWR	—	数字负电源
VDDRF	VDDRF	—	PWR	—	RF 正电源
VSSRF	VSSRF	—	PWR	—	RF 负电源
VSSRF_PA	VSSRF_ PA	—	PWR	—	RF 功率放大器负电源
NC	NC	—	—	—	未连接
EP	VSS	—	PWR	—	裸露焊盘，必须接地

注：I/T：输入类型；　O/T：输出类型；　OPT：通过寄存器选择；
　　PWR：电源；　ST：施密特触发输入；　CMOS：CMOS 输出；
　　AN：模拟信号；　HXT：高频晶体振荡器

极限参数

电源供应电压 ..... V_{SS}-0.3V ~ V_{SS}+3.6V
端口输入电压 ..... V_{SS}-0.3V ~ V_{DD}+0.3V
储存温度 ..... -60°C ~ 150°C
工作温度 ..... -40°C ~ 85°C
I_{OH}总电流 ..... -80mA
I_{OL}总电流 80mA
总功耗 500mW
ESD HBM ..... ±2kV
ESD MM ..... ±400V

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。

该系列单片机为 ESD 感应。人体模式 HBM (Human Body Mode) 基于标准 MIL-STD-883H Method 3015.8。机器模式 MM (Machine Mode) 基于 JEDEC EIA/JESD22-A115。

直流电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	MCU 工作电压 – HXT (RF 外部晶振)	—	$f_{SYS}=f_{HXT}$=16MHz	2.0	—	3.6	V
	MCU 工作电压 – HIRC	—	$f_{SYS}=f_{HIRC}$=16MHz	2.0	—	3.6	V
	MCU 工作电压 – LIRC	—	$f_{SYS}=f_{LIRC}$=32kHz	2.0	—	3.6	V
V_{15O}	LDO 工作电压	2.0V ~3.6V	—	-10%	1.5	+10%	V
I_{DD}	工作电流 – LIRC (LCMD=1)	3V	无负载，所有外设关闭，$f_{SYS}=f_{LIRC}$=32kHz，RF TX 电源关闭	—	30	50	μA
	工作电流 – LIRC (LCMD=0)	3V	无负载，所有外设关闭，$f_{SYS}=f_{LIRC}$=32kHz，RF TX 电源关闭	—	42	70	μA
	工作电流 – HIRC	3V	无负载, 所有外设关闭，$f_{SYS}=f_{HIRC}$=16MHz，RF TX 电源关闭	—	0.76	1.50	mA
	工作电流 – HXT	3V	无负载, 所有外设关闭，$f_{SYS}=f_{HXT}$=16MHz，RF TX 电源关闭	—	1.1	1.6	mA

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{STB}	待机电流 (深度休眠模式，LDO off)	3V	无负载，所有外设关闭，WDT off	—	0.4	1.0	μA
	待机电流 (深度休眠模式，LDO off)	3V	无负载，所有外设关闭，WDT on	—	1.2	3.0	μA
	待机电流 (空闲模式 1，LDO on)	3V	无负载，所有外设关闭，f_{SUB} on，$f_{SYS}=f_{HXT}$=16MHz	—	600	900	μA
V_{IL}	I/O 口低电平输入电压	3V	—	0	—	0.9	V
		—	—	0	—	$0.3V_{DD}$	
V_{IH}	I/O 口高电平输入电压	3V	—	2.1	—	3.0	V
		—	—	$0.7V_{DD}$	—	V_{DD}	
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	10	20	—	mA
I_{OH}	I/O 口源电流	3V	$V_{OH}=0.9V_{DD}$	-5	-10	—	mA
V_{OL}	I/O 口低电平输出电压	3V	I_{OL}=10mA	—	—	0.3	V
V_{OH}	I/O 口高电平输出电压	3V	I_{OH}=-5mA	2.7	—	—	V
R_{PH}	I/O 口上拉电阻	3V	—	20	30	60	kΩ

交流电气特性

Ta=25°C，除非另有说明

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS}	系统时钟 – HXT (RF 外部晶振)	2.0V~3.6V	$f_{SYS}=f_{HXT}$=16MHz	—	16	—	MHz
	MCU 系统时钟 – HIRC	2.0V~3.6V	$f_{SYS}=f_{HIRC}$=16MHz	—	16	—	MHz
	系统时钟 – LIRC	2.0V~3.6V	$f_{SYS}=f_{LIRC}$=32kHz	—	32	—	kHz
f_{HIRC}	内部高速 RC 振荡器时钟	3V	Ta=25°C	-2%	16	+2%	MHz
		3V±0.1V	Ta=0°C~70°C	-5%	16	+5%	
		2.0V~3.6V	Ta=0°C~70°C	-7%	16	+7%	
		2.0V~3.6V	Ta=-40°C~85°C	-10%	16	+10%	
f_{LIRC}	内部低速 RC 振荡器时钟	3V	Ta=25°C	-10%	32	+10%	kHz
t_{INT}	外部中断最小脉宽	—	—	10	—	—	μs
t_{TCK}	xTM xTCK 输入引脚最小脉宽	—	—	0.3	—	—	μs
t_{TPI}	PTM PTPI 输入引脚最小脉宽	—	—	0.3	—	—	μs

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
t_{SST}	系统启动时间 (从 f_{SYS} off 的 HALT 状态下唤醒)	—	$f_{SYS}=f_{HXT} \sim f_{HXT}/64$	128	—	—	t_{HXT}
		—	$f_{SYS}=f_{HIRC} \sim f_{HIRC}/64$	16	—	—	t_{HIRC}
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{SUB}
	系统启动时间 (从 f_{SYS} on 的 HALT 状态下唤醒)	—	$f_{SYS}=f_H \sim f_H/64$, $f_H=f_{HXT}$ 或 f_{HIRC}	2	—	—	t_H
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{SUB}
	系统启动时间 (低速模式 → 正常模式或 $f_H=f_{HIRC}$ → f_{HXT})	—	f_{HXT} off→on (HXTF=1)	1024	—	—	t_{HXT}
		—	f_{HIRC} off→on (HIRCF=1)	16	—	—	t_{HIRC}
	系统启动时间 (WDT 溢出复位)	—	—	0	—	—	t_H
t_{RSTD}	系统复位延迟时间 (上电复位，LVR 硬件复位，LVRC/WDTC/RSTC 软件复位)	—	—	25	50	100	ms
	系统复位延迟时间 (WDT 硬件复位)	—	—	8.3	16.7	33.3	ms
t_{SRESET}	软件复位最小延迟脉宽	—	—	45	90	120	μs

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 1.9V	Typ. -0.1	1.9	Typ. +0.1	V
V_{LVD}	低电压检测电压	—	LVD 使能，电压选择 1.9V	Typ. -0.1	1.9	Typ. +0.1	V
		—	LVD 使能，电压选择 2.0V	-5%	2.0	+5%	
		—	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 2.8V		2.8		
		—	LVD 使能，电压选择 2.9V		2.9		
		—	LVD 使能，电压选择 3.0V		3.0		
		—	LVD 使能，电压选择 3.3V		3.3		
I_{OP}	工作电流	3V	LVD 使能，LVR 使能	—	25	35	μA
t_{LVDS}	LVDO 稳定时间	—	LVR 使能，LVD off → on	—	—	15	μs
		—	LVR 除能，LVD off → on	—	—	150	μs
t_{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	120	240	480	μs
t_{LVD}	产生 LVD 中断的低电压最短保持时间	—	—	60	120	240	μs

RF 发射器电气特性

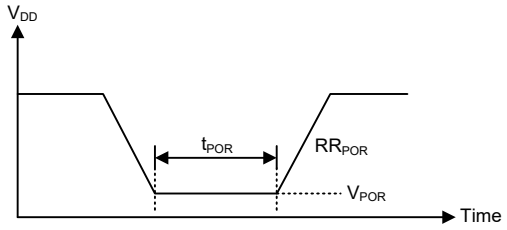
Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DDRF}	RF 工作电压	—	—	2.0	3.0	3.6	V
I_{STBRF}	RF 暂停电流	—	—	—	—	1	μA
f_{XTAL}	RF 工作晶振	—	—	—	16	—	MHz
f_{XTALV}	RF 工作晶振变化	—	—	-20	—	+20	ppm
f_{OP}	RF 工作频率	—	—	315	—	915	MHz
f_{FSK}	FSK 频率偏差	—	—	10	—	150	kHz
f_{STEP}	RF 频率步进	—	f_{OP}=315/433MHz	—	—	50	Hz
		—	f_{OP}=868/915MHz	—	—	100	
R_{FSK}	无线数据传输速率 (FSK)	—	—	0.5	—	100	Kbps
R_{OOK}	无线符号传输速率 (OOK)	—	—	0.5	—	25	Ksps
P_{RF}	输出功率	3.0V	—	0	10	13	dBm
P_{RFV}	输出功率变化	2.0V~3.3V	P_{RF}=10dBm (f_{OP}=433MHz)	—	—	6	dBm
		2.5V~3.3V	P_{RF}=10dBm (f_{OP}=433MHz)	—	—	3	dBm
I_{DDTX}	发射器工作电流	3.0V	P_{RF}=0dBm (f_{OP}=315/433MHz)	—	12	—	mA
		3.0V	P_{RF}=10dBm (f_{OP}=315/433MHz)	—	18	—	mA
		3.0V	P_{RF}=13dBm (f_{OP}=315/433MHz)	—	25	—	mA
		3.0V	P_{RF}=0dBm (f_{OP}=868/915MHz)	—	13	—	mA
		3.0V	P_{RF}=10dBm (f_{OP}=868/915MHz)	—	23	—	mA
		3.0V	P_{RF}=13dBm (f_{OP}=868/915MHz)	—	28	—	mA
PN_{TX}	发射器相位噪声	—	P_{RF}=10dBm，100kHz 来自载波	—	—	-80	dBc/Hz
		—	P_{RF}=10dBm，1MHz 来自载波	—	—	-100	dBc/Hz
SE_{TX}	发射器杂散发射 (P_{RF}=10dBm，f_{OP}=433MHz)	—	f < 1GHz	—	—	-36	dBm
		—	47MHz < f < 74MHz 87.5MHz < f < 118MHz 174MHz < f < 230MHz 470MHz < f < 790MHz	—	—	-54	dBm
		—	二次谐波	—	—	-30	dBm
		—	三次谐波	—	—	-30	dBm
		—	二次谐波 (其它频率)	—	—	-30	dBm
		—	三次谐波 (其它频率)	—	—	-30	dBm
f_{LO}	RF 频率覆盖范围	—	315MHz 频带	240	—	340	MHz
		—	433MHz 频带	410	—	510	
		—	868/915MHz 频带	800	—	1000	

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



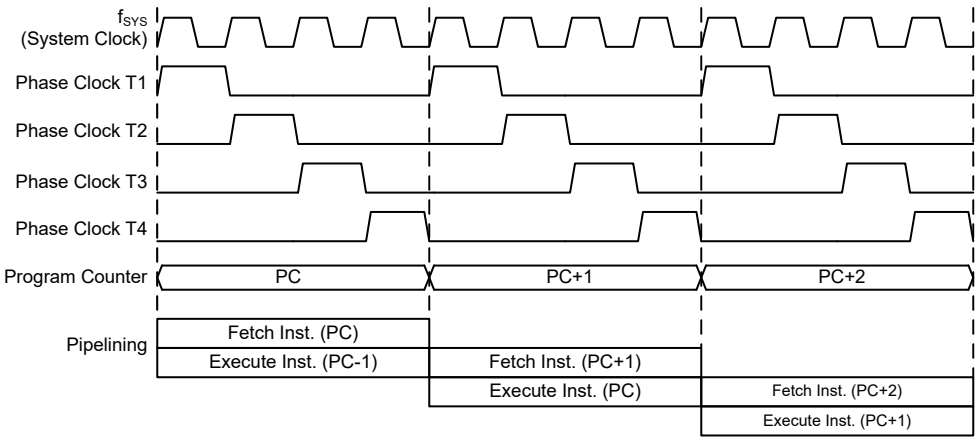
系统结构

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，该系列单片机具有高运算速度和高性能的特点。通过流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令需多一个指令周期外，其它大部分标准指令或扩展指令都能分别在一个或两个指令周期内完成。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，确保了在提供具有较大可靠度和灵活性的 I/O 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大量生产的控制应用。

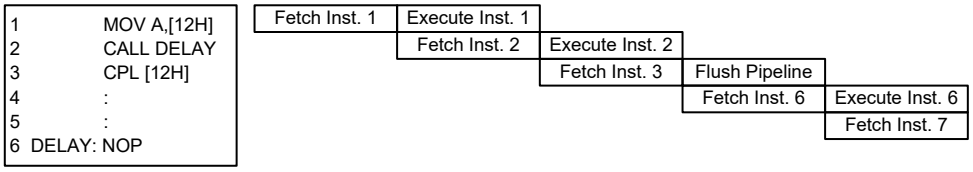
时序和流水线结构

主系统时钟由 HXT、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。

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。



系统时序和流水线



指令捕捉

程序计数器

在程序执行期间，程序计数器用来指向下一个要执行的指令地址。除了“JMP”和“CALL”指令需要跳转到一个非连续的程序存储器地址之外，它会在每条指令执行完成以后自动加一。只有较低的 8 位，即所谓的程序计数器低字节寄存器 PCL，可以被用户直接读写。

当执行的指令要求跳转到不连续的地址时，如跳转指令、子程序调用、中断或复位等，单片机通过加载所需要的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**
BC68F2130	PC10~PC8	PCL7~PCL0
BC68F2140	PC11~PC8	PCL7~PCL0
BC68F2150	PC12~PC8	PCL7~PCL0

程序计数器

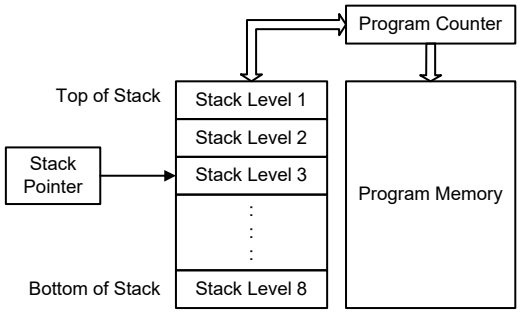
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

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

如果堆栈已满，且有非屏蔽的中断发生，中断请求标志会被置位，但中断响应将被禁止。当堆栈指针减少 (执行 RET 或 RETI)，中断将被响应。这个特性提供程序设计者简单的方法来预防堆栈溢出。然而即使堆栈已满，CALL 指令仍然可以被执行，而造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这可能导致不可预期的程序分支指令执行错误。

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



算术逻辑单元 – ALU

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

- 算术运算：
  ADD，ADDM，ADC，ADCM，SUB，SUBM，SBC，SBCM，DAA，
  LADD，LADDM，LADC，LADCM，LSUB，LSUBM，LSBC，LSBCM，
  LDAA
- 逻辑运算：
  AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA，
  LAND，LANDM，LOR，LORM，LXOR，LXORM，LCPL，LCPLA
- 移位运算：
  RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC，
  LRR，LRRA，LRRCA，LRRC，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，
  LSNZ，LSZ，LSZA，LSIZ，LSIZA，LSDZ，LSDZA

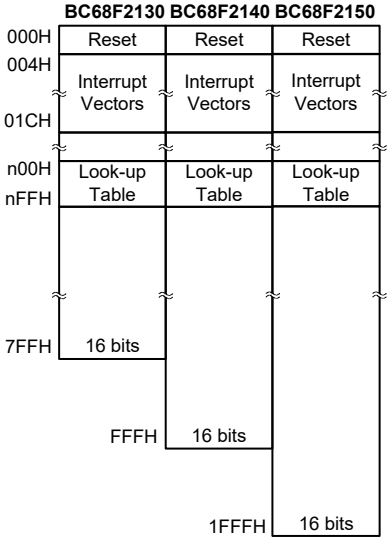
Flash 程序存储器

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。

结构

程序存储器的容量为 2K×16~8K×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，由表格指针寄存器来寻址。

单片机型号	容量
BC68F2130	2K×16
BC68F2140	4K×16
BC68F2150	8K×16



程序存储器结构

特殊向量

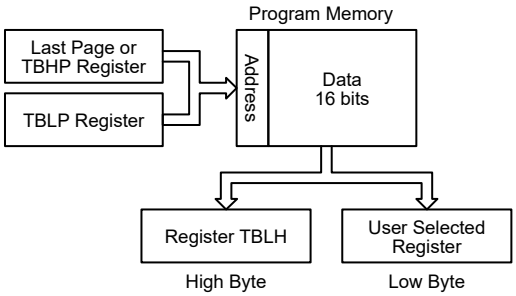
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

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。这些寄存器定义表格总的地址。

在设定完表格指针后，若数据存储器 [m] 位于 Sector 0，表格数据可以使用例如“TABRD [m]”或“TABRDL [m]”指令分别从程序存储器查表读取。如果存储器 [m] 位于数据存储器其它 Sector，表格数据可以使用例如“LTABRD [m]”或“LTABRDL [m]”指令分别从程序存储器查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到用户指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊功能寄存器。

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。ORG 指令的值“700H”指向的地址是 BC68F2130 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 706H，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]”指令被使用，则表格指针指向 TBLP 和 TBHP 指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [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，可重复储存，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

```
ds .section 'data'
tempreg1 db?        ; temporary register #1
tempreg2 db?        ; temporary register #2
code0 .section 'code'
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          ; it is not necessary to set tbhp if executing tabrdl
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer
                    ; data at program memory address "706H" transferred to
                    ; tempreg1 and TBLH
dec tblp            ; reduce value of table pointer by one
tabrd tempreg2      ; transfers value in table referenced by table pointer
                    ; data at program memory address "705H" transferred to
                    ; tempreg2 and TBLH
                    ; in this example the data "1AH" is transferred to
                    ; tempreg1 and data "0FH" to tempreg2
                    ; the value "00H" will be transferred to the high byte
                    ; register TBLH
:
:
```

```
org 700h            ; sets initial address of last page
dc 00Ah,00Bh,00Ch,00Dh,00Eh,00Fh,01Ah,01Bh
```

在线烧录 – ICP

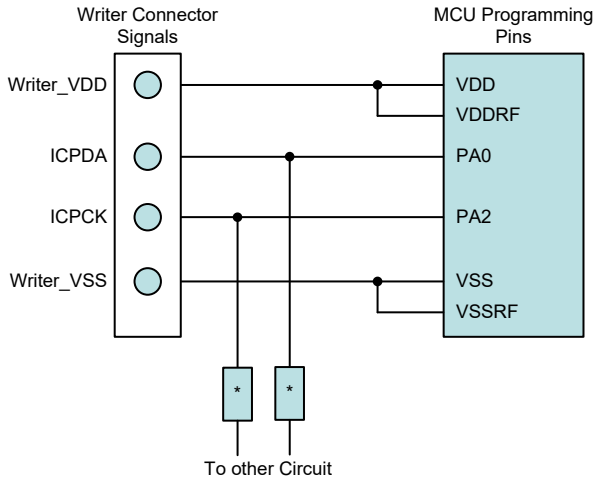
Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。

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 烧录器引脚名称	MCU 在线烧录引脚名称	引脚说明
ICPDA	PA0	串行数据 / 地址烧录
ICPCK	PA2	时钟烧录
VDD	VDD&VDDRF	电源 (3V)
VSS	VSS&VSSRF	地

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。

烧录过程中，用户必须确保 ICPDA 和 ICPCK 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 1kΩ，若为电容则其必须小于 1nF。

片上调试 – OCDS

该系列单片机还提供片上调试功能 (OCDS) 用于开发过程中的 MCU 调试。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 OCDS 功能进行调试时，单片机 OCDSDA 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 用户手册”文件。

Holtek e-Link 引脚名称	MCU 芯片引脚名称	引脚说明
OCDSDA	OCDSDA	片上调试数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
VSS	VSS	地

在应用烧录 – IAP

该系列单片机提供 IAP 功能来对 Flash ROM 进行数据和程序更新。用户可自行定义 IAP ROM 地址，但是用户在使用 IAP 功能时必须注意几个特点。

BC68F2130 IAP 配置	
块擦除	256 个字 / 块
写	4 个字 / 次
读	1 个字 / 次

BC68F2140 IAP 配置	
块擦除	256 个字 / 块
写	4 个字 / 次
读	1 个字 / 次

BC68F2150 IAP 配置	
块擦除	256 个字 / 块
写	4 个字 / 次
读	1 个字 / 次

IAP 控制寄存器

位于数据存储器 Sector 0 的地址寄存器 FARL/FARH 和数据寄存器 FD0L/FD0H、FD1L/FD1H、FD2L/FD2H 和 FD3L/FD3H 以及控制寄存器 FC0 和 FC1，都是与 IAP 相关的 Flash 存取寄存器。所有与这些寄存器相关的读写操作可以使用间接寻址存取方式来执行。

寄存器名称	位							
	7	6	5	4	3	2	1	0
FC0	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
FC1	D7	D6	D5	D4	D3	D2	D1	D0
FARL	A7	A6	A5	A4	A3	A2	A1	A0
FARH (BC68F2130)	—	—	—	—	—	A10	A9	A8
FARH (BC68F2140)	—	—	—	—	A11	A10	A9	A8
FARH (BC68F2150)	—	—	—	A12	A11	A10	A9	A8
FD0L	D7	D6	D5	D4	D3	D2	D1	D0
FD0H	D15	D14	D13	D12	D11	D10	D9	D8
FD1L	D7	D6	D5	D4	D3	D2	D1	D0
FD1H	D15	D14	D13	D12	D11	D10	D9	D8
FD2L	D7	D6	D5	D4	D3	D2	D1	D0
FD2H	D15	D14	D13	D12	D11	D10	D9	D8
FD3L	D7	D6	D5	D4	D3	D2	D1	D0
FD3H	D15	D14	D13	D12	D11	D10	D9	D8

IAP 寄存器列表

- **FC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
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 **CFWEN**：Flash 存储器擦 / 写使能控制位
0：Flash 存储器擦 / 写功能除能
1：Flash 存储器擦 / 写功能已成功使能
当此位由应用程序清零后，Flash 存储器擦 / 写功能除能。注意，对此位直接写“1”不会使能擦 / 写功能。此位只是用来指示 Flash 存储器擦 / 写功能状态，即当此位由硬件置为“1”时，表明 Flash 存储器擦 / 写功能已经成功使能，若为“0”，该功能除能。

Bit 6~4 **FMOD2~FMOD0**：模式选择
000：写程序存储器
001：块擦除程序存储器
010：保留位
011：读程序存储器
100：保留位
101：保留位
110：Flash 存储器擦 / 写功能使能模式
111：保留位

Bit 3 **FWPEN**：Flash 存储器擦 / 写功能使能步骤控制
0：除能
1：使能
当此位置为“1”且 FMOD2~FMOD0 为“110”时，IAP 控制器将执行“Flash 存储器擦 / 写功能使能”步骤。一旦 Flash 存储器擦 / 写功能成功使能，无需再设置 FWPEN 位。

Bit 2 **FWT**：Flash ROM 写开始控制位
0：未开始 Flash 存储器写或 Flash 存储器写过程已完成
1：开始 Flash 存储器写过程
此位由软件置“1”，当 Flash 存储器写过程完成，由硬件清零。

Bit 1 **FRDEN**：Flash 存储器读使能位
0：Flash 存储器读除能
1：Flash 存储器读使能

Bit 0 **FRD**：Flash 存储器读开始控制位
0：未开始 Flash 存储器读或 Flash 存储器读过程已完成
1：开始 Flash 存储器读过程
此位由软件置“1”，当 Flash 存储器读过程完成，由硬件清零。

注：1. 在同一条指令中 FWT、FRDEN 和 FRD 位不可同时设置为“1”。
2. 确保 f_{SUB} 时钟在执行擦或写动作前已稳定。
3. 当读、擦或写动作成功启动后，CPU 相关操作将停止。
4. 确保读、擦或写动作成功完成后才可执行其它操作。

- **FC1 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**：整个芯片复位控制位
当用户写特定值“55H”到该寄存器，将产生一个复位信号使整个单片机复位。

- **FARL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	A7	A6	A5	A4	A3	A2	A1	A0
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 **A7~A0**：Flash 存储器地址 [7:0]

- **FARH 寄存器 – BC68F2130**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	A10	A9	A8
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”

Bit 2~0 **A10~A8**：Flash 存储器地址 [10:8]

- **FARH 寄存器 – BC68F2140**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	A11	A10	A9	A8
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 3~0 **A11~A8**：Flash 存储器地址 [11:8]

- **FARH 寄存器 – BC68F2150**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	A12	A11	A10	A9	A8
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 **A12~A8**：Flash 存储器地址 [12:8]

- **FD0L 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**：第一个 Flash 存储器数据 [7:0]

- **FD0H 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**：第一个 Flash 存储器数据 [15:8]

- **FD1L 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**：第二个 Flash 存储器数据 [7:0]

- **FD1H 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**：第二个 Flash 存储器数据 [15:8]

- **FD2L 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**：第三个 Flash 存储器数据 [7:0]

- **FD2H 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**：第三个 Flash 存储器数据 [15:8]

- **FD3L 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**：第四个 Flash 存储器数据 [7:0]

- **FD3H 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**：第四个 Flash 存储器数据 [15:8]

Flash 存储器擦 / 写功能使能步骤

为使用户可以通过 IAP 控制寄存器来更改 Flash 存储器数据，用户必须先使能 Flash 存储器擦 / 写操作，步骤如下：

步骤 1. 写“110”到 FMOD2~FMOD0 位，选择 Flash 存储器擦 / 写功能使能模式。

步骤 2. FWPEN 置为“1”。步骤 1 和步骤 2 可同时执行。

步骤 3. 数据序列 00H、04H、0DH、09H、C3H 和 40H 必须分别写入寄存器 FD1L、FD1H、FD2L、FD2H、FD3L 和 FD3H。

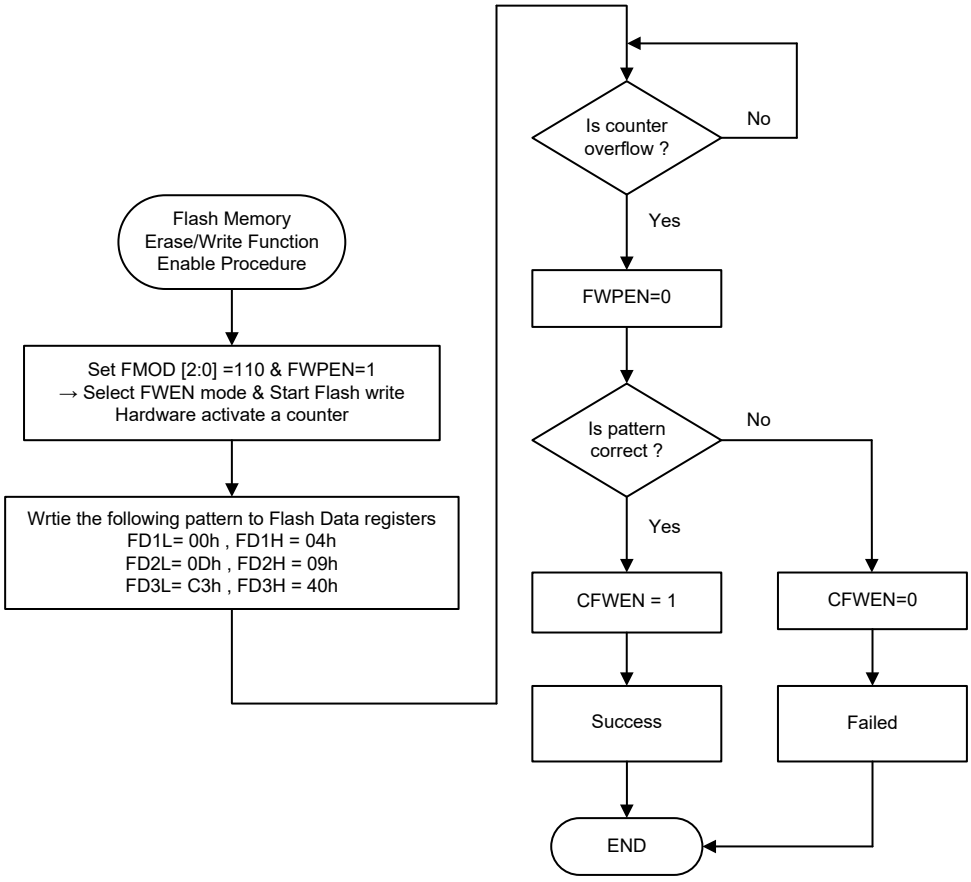
步骤 4. 溢出周期为 300μs 的计数器将进行有效计时，此时要求用户尽快将正确的数据序列写入 FD1L/FD1H~FD3L/FD3H 寄存器对。计数器时钟来自 f_{SUB} 时钟。

步骤 5. 如果计数器溢出或模式数据不正确，Flash 存储器擦 / 写操作不被使能，用户必须再次重复以上步骤。FWPEN 位将自动由硬件清零。

步骤 6. 如果计数器溢出前模式数据正确，Flash 存储器擦 / 写操作将使能且 FWPEN 位将自动由硬件清零。CFWEN 位也由硬件置为“1”，表明 Flash 存储器擦 / 写操作成功使能。

步骤 7. 一旦 Flash 存储器擦 / 写操作使能，用户可通过 Flash 控制寄存器更改 Flash ROM 数据。

步骤 8. 用户可以清零 CFWEN 位来除能 Flash 存储器擦 / 写操作。



Flash 存储器擦 / 写功能使能步骤

Flash 存储器读 / 写步骤

通过前面的 IAP 步骤成功使能 Flash 存储器擦 / 写功能后，用户必须先擦除相应的 Flash 存储块，然后再开始 Flash 存储器写操作。对于该系列单片机，块擦除操作的数量是每块 256 个字，有效的块擦除地址只能由 FARH 寄存器指定，FARL 寄存器不用来指定块擦除地址。

块擦除	FARH [2:0]	FARL [7:0]
0	0000 0000	xxxx xxxx
1	0000 0001	xxxx xxxx
2	0000 0010	xxxx xxxx
3	0000 0011	xxxx xxxx
4	0000 0100	xxxx xxxx
5	0000 0101	xxxx xxxx
6	0000 0110	xxxx xxxx
7	0000 0111	xxxx xxxx

“x”：无关

BC68F2130 块擦除数量和选择

块擦除	FARH [3:0]	FARL [7:0]
0	0000 0000	xxxx xxxx
1	0000 0001	xxxx xxxx
2	0000 0010	xxxx xxxx
3	0000 0011	xxxx xxxx
4	0000 0100	xxxx xxxx
5	0000 0101	xxxx xxxx
6	0000 0110	xxxx xxxx
7	0000 0111	xxxx xxxx
8	0000 1000	xxxx xxxx
9	0000 1001	xxxx xxxx
10	0000 1010	xxxx xxxx
11	0000 1011	xxxx xxxx
12	0000 1100	xxxx xxxx
13	0000 1101	xxxx xxxx
14	0000 1110	xxxx xxxx
15	0000 1111	xxxx xxxx

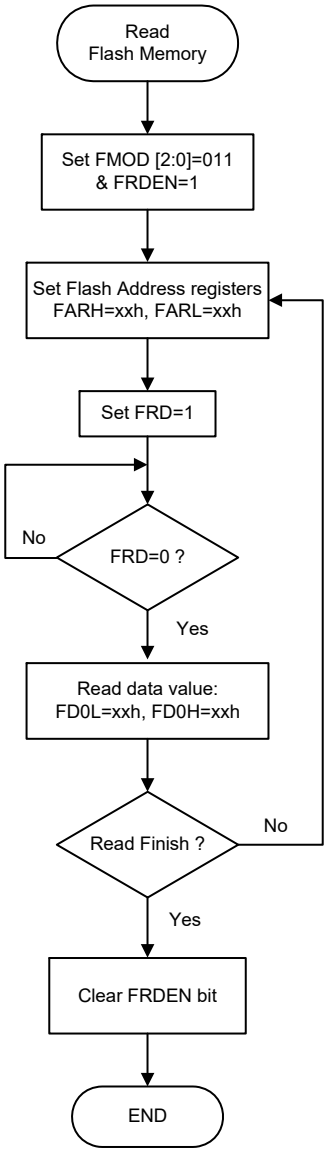
“x”：无关

BC68F2140 块擦除数量和选择

块擦除	FARH [4:0]	FARL [7:0]
0	0000 0000	xxxx xxxx
1	0000 0001	xxxx xxxx
2	0000 0010	xxxx xxxx
3	0000 0011	xxxx xxxx
4	0000 0100	xxxx xxxx
5	0000 0101	xxxx xxxx
6	0000 0110	xxxx xxxx
7	0000 0111	xxxx xxxx
8	0000 1000	xxxx xxxx
9	0000 1001	xxxx xxxx
10	0000 1010	xxxx xxxx
11	0000 1011	xxxx xxxx
12	0000 1100	xxxx xxxx
13	0000 1101	xxxx xxxx
14	0000 1110	xxxx xxxx
15	0000 1111	xxxx xxxx
16	0001 0000	xxxx xxxx
17	0001 0001	xxxx xxxx
18	0001 0010	xxxx xxxx
19	0001 0011	xxxx xxxx
20	0001 0100	xxxx xxxx
21	0001 0101	xxxx xxxx
22	0001 0110	xxxx xxxx
23	0001 0111	xxxx xxxx
24	0001 1000	xxxx xxxx
25	0001 1001	xxxx xxxx
26	0001 1010	xxxx xxxx
27	0001 1011	xxxx xxxx
28	0001 1100	xxxx xxxx
29	0001 1101	xxxx xxxx
30	0001 1110	xxxx xxxx
31	0001 1111	xxxx xxxx

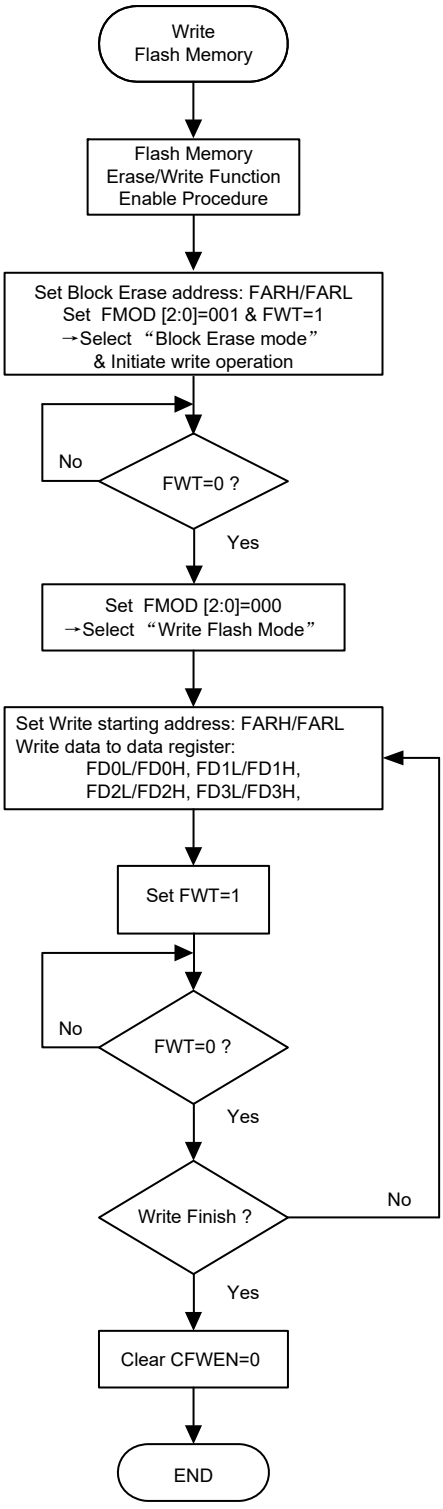
“x”：无关

BC68F2150 块擦除数量和选择



读 Flash 存储器步骤

注：1. 当读动作成功启动后，所有 CPU 相关操作将暂停。
2. FRD 位由高变低所需时间为 3 个指令周期 (典型值)。



写 Flash 存储器步骤

注：1. 当擦或写动作成功启动后，所有 CPU 相关操作将暂停。
2. FWT 位由高变低所需时间为 2.2ms (典型值)。
3. 当 FARH/FARL 的地址组合并非 4 的倍数时，数据无法正常写入依序的 4 笔地址中。

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。

数据存储器分为两种类型，第一种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。第二种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。

当数据存储器使用间接寻址时，切换不同的数据存储器 Sector 可通过设置正确的存储器指针实现。

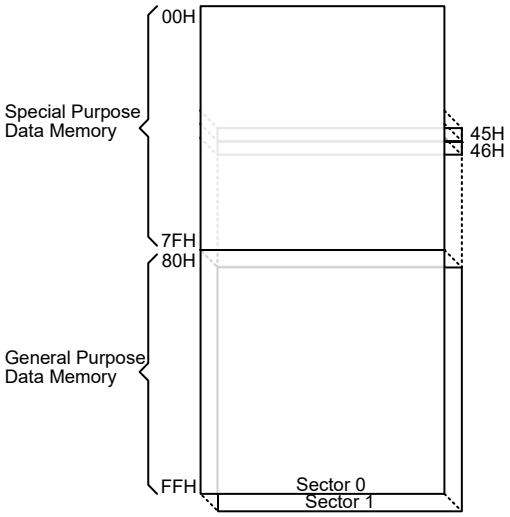
结构

数据存储器被分为多个 Sector，都位于 8 位存储器中。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	
	分布 Sectors	容量	Sector：地址
BC68F2130 BC68F2140 BC68F2150	0, 1	256×8	0：80H~FFH 1：80H~FFH

数据存储器概要



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，它并没有可用于数据存储器的存储区指针。对于数据存储器，使用间接寻址访问方式时所需的 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。

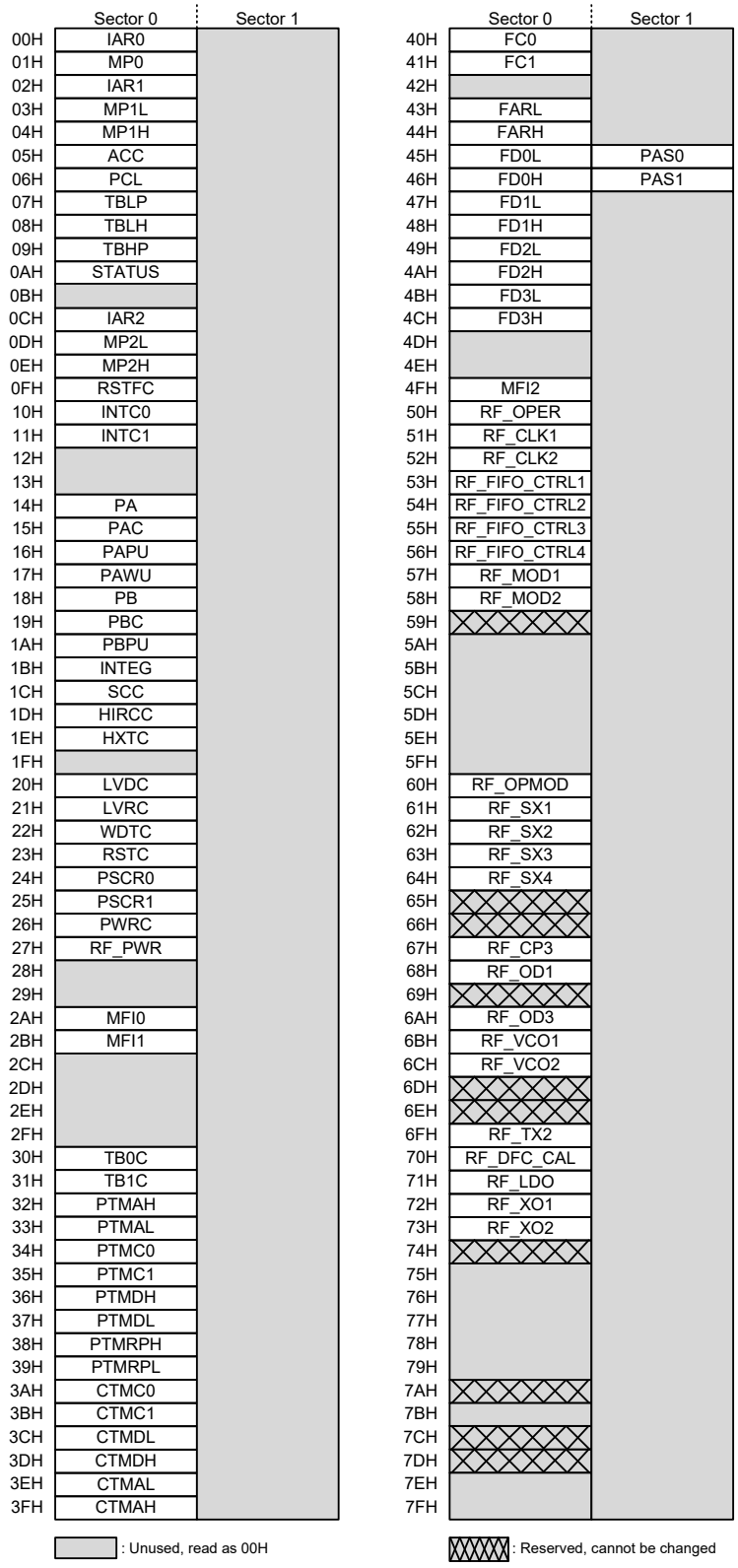
直接寻址可用于所有 Sector，通过扩展指令可以寻址所有可用的数据存储器空间。当所访问的数据存储器位于除 Sector 0 外的任何数据存储器 Sector 时，扩展指令可代替间接寻址方式用来访问数据存储器。标准指令和扩展指令的主要区别在于扩展指令中的数据存储器地址“m”有 9 个有效位，高字节表示 Sector，低字节表示指定的地址。

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，较大地方便了用户在数据存储器内进行位操作。

特殊功能数据存储器

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



特殊功能数据存储器结构

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，但不同于正常寄存器，它们没有实际的物理地址。与定义实际存储器地址的直接存储器寻址不同，间接寻址是使用间接寻址寄存器和存储器指针来执行存储器数据操作。在间接寻址寄存器 IAR0、IAR1 和 IAR2 上的任何动作，将存储器指针 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 只可以访问 Sector 0，而 IAR1 和 MP1L/MP1H、IAR2 和 MP2L/MP2H 可以访问所有 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, MP1H/MP1L, MP2H/MP2L

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器中能像普通的寄存器一般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由存储器指针所指定的地址。MP0、IAR0 仅可用于访问 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。直接寻址通过相关的数据存储器寻址指令来访问所有的数据 Sector。

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                    ; increase 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 mp1h,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                   ; increase 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”是位于数据存储器任意 Sector 的某一地址。例如，m=1F0H 表示 Sector 1 中的地址 0F0H。

累加器 – ACC

对任何单片机来说，累加器是相当重要的，且与 ALU 所完成的运算有密切关系，所有 ALU 得到的运算结果都会暂时存在 ACC 累加器里。若没有累加器，ALU 必须在每次进行如加法、减法和移位的运算时，将结果写入到数据存储器，这样会造成程序编写和时间的负担。另外数据传送也常常牵涉到累加器的临时储存功能，例如在使用者定义的一个寄存器和另一个寄存器之间传送数据时，由于两寄存器之间不能直接传送数据，因此必须通过累加器来传送数据。

程序计数器低字节寄存器 – PCL

为了提供额外的程序控制功能，程序计数器低字节设置在数据存储器的特殊功能区域内，程序员可对此寄存器进行操作，很容易的直接跳转到其它程序地址。直接给 PCL 寄存器赋值将导致程序直接跳转到程序存储器的某一地址，然而由于寄存器只有 8 位长度，因此只允许在本页的程序存储器范围内进行跳转，而当使用这种运算时，要注意会插入一个空指令周期。

查表寄存器 – TBLP, 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被改变。任何数据写入到状态寄存器将不会改变 TO 或 PDF 标志位。另外，执行不同的指令后，与状态寄存器有关的运算可能会得到不同的结果。TO 标志位只会受系统上电、看门狗溢出或执行“CLR WDT”或“HALT”指令影响。PDF 标志位只会受执行“HALT”或“CLR WDT”指令或系统上电影响。

Z、OV、AC、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。

- **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/W	R/W	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 也受循环移位指令的影响。

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较佳的优化。振荡器选择和操作是通过相关的控制寄存器完成的。

振荡器概述

振荡器除了作为系统时钟源，还作为 RF 电路、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部器件。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。

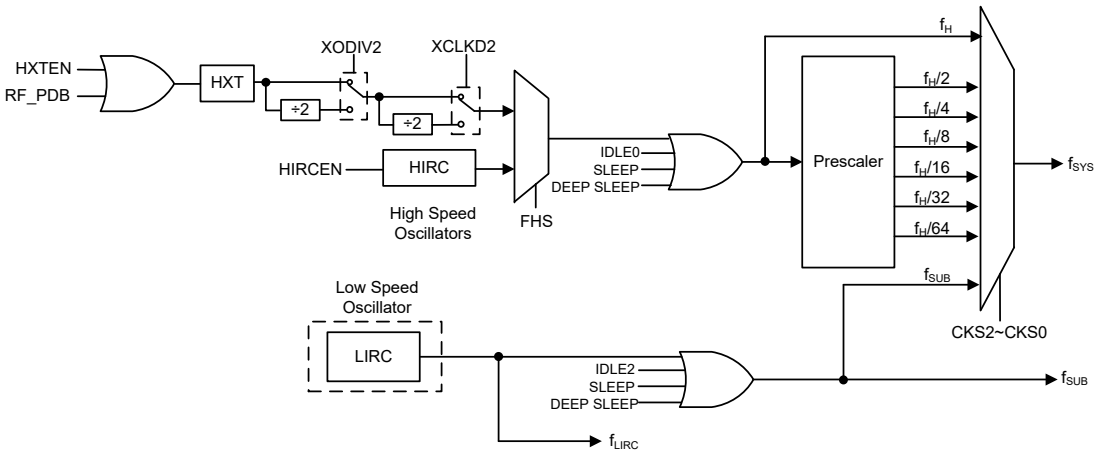
类型	名称	频率	引脚
外部高速晶振	HXT	16MHz	OSC1/OSC2
内部高速 RC	HIRC	16MHz	—
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

该系列单片机有三个系统振荡器，包括两个高速振荡器和一个低速振荡器。高速振荡器有外部晶体 / 陶瓷振荡器 HXT 和内部 16MHz 高速振荡器 HIRC，低速振荡器有内部 32kHz 低速振荡器 LIRC。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。

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。

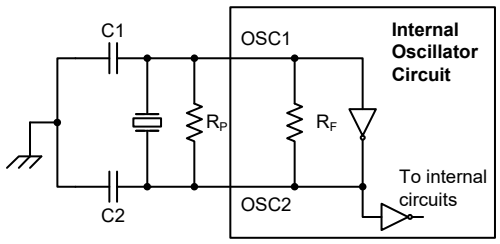


系统时钟配置

外部晶体 / 陶瓷振荡器 – HXT

外部晶体 / 陶瓷振荡器是高频振荡器之一，也为 RF 电路的时钟源。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. The capacitor load is internally integrated and determined by the RF_XO1 register.

晶体 / 陶瓷振荡器

晶体振荡器 C1 和 C2 值		
晶振频率	C1	C2
16MHz	0pF	0pF
注：C1 和 C2 数值仅作参考用		

晶体振荡器电容推荐值

内部高速 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 振荡器具有固定的频率 16MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因电源电压、温度以及芯片制成工艺不同的影响较大程度地降低。如果选择了该内部时钟，无需额外的外部引脚。

内部 32kHz 振荡器 – LIRC

内部 32kHz 振荡器是低频振荡器。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较大程度地降低。

工作模式和系统时钟

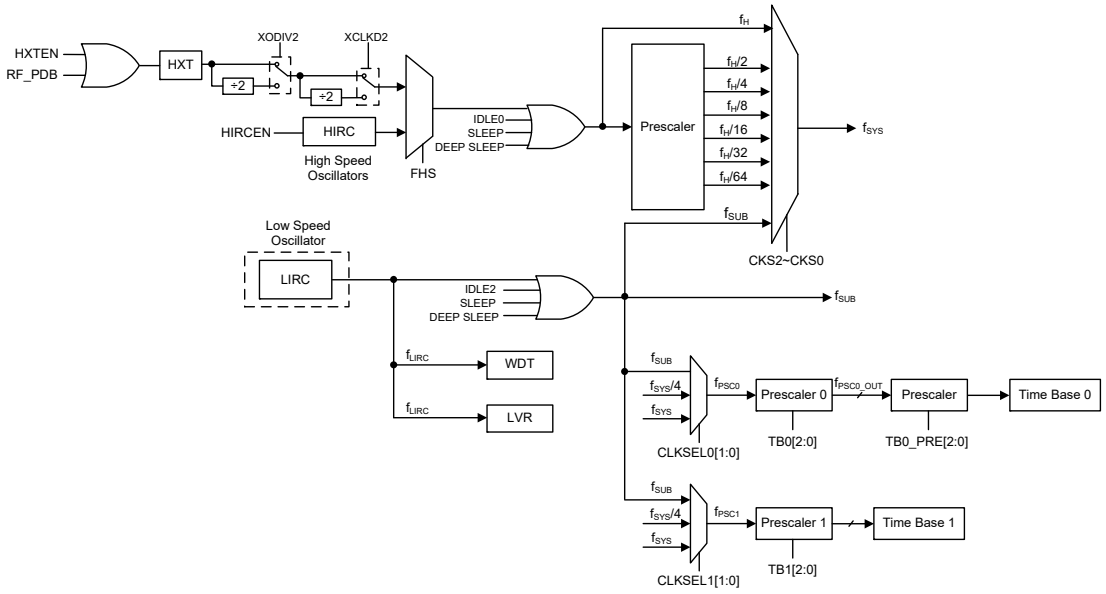
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。此系列单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

系统时钟

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较大的应用性能。

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。通过 HXTEN 和 RF_PDB 位使能 HXT 振荡器，然后通过 XODIV2 和 XCLKD2 进行二或四分频。RF_PDB、XODIV2 和 XCLKD2 这三个位的相关描述详见 RF 章节。

低频系统时钟源来自内部时钟 f_{SUB}，若 f_{SUB} 被选择，低频时钟来自 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。



设备时钟配置

注：1. 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~$f_H/64$ 频率的时钟源。

2. 当系统进入深度休眠模式时，时基 0 时钟源为 f_{LIRC}，此时钟源的开关取决于看门狗定时器的开关状态。

系统工作模式

单片机有 7 种不同的工作模式，每种有它自身的特性，根据应用中不同的性能和功耗要求可选择不同的工作模式。单片机正常工作有两种模式：正常模式和低速模式。剩余的 5 种工作模式：深度休眠模式、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	LDO	CPU	寄存器设置				f_{SYS}	f_H	f_{SUB}	f_{LIRC}
			PWDN	FHIDEN	FSIDEN	CKS2~CKS0				
正常模式	On	On	0	x	x	000~110	f_H~f_H/64	On	On	On
低速模式	On	On	0	x	x	111	f_{SUB}	On/Off (1)	On	On
空闲模式 0	On	Off	0	0	1	000~110	Off	Off	On	On
						111	On			
空间模式 1	On	Off	0	1	1	xxx	On	On	On	On
空闲模式 2	On	Off	0	1	0	000~110	On	On	Off	On
						111	Off			
休眠模式	On	Off	0	0	0	xxx	Off	Off	Off	On/Off (2)
深度休眠模式	Off	Off	1	x	x	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. 在休眠或深度休眠模式中，f_{LIRC} 开启或关闭由 WDT 功能使能或除能控制。

正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。将 PWRC 寄存器中的 PWDN 位设为低可开启 1.5V LDO。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。将寄存器 PWRC 中的 PWDN 位设为低可开启 1.5V LDO。该低速时钟源可来自 f_{SUB}，而 f_{SUB} 可来自 LIRC 振荡器。

休眠模式

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为低，寄存器 PWRC 中的 PWDN 位为低时，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。若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，寄存器 PWRC 中的 PWDN 位为低时，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，寄存器 PWRC 中的 PWDN 位为低时，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保持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，寄存器 PWRC 中的 PWDN 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保持一些外围功能继续工作。

深度休眠模式

执行 HALT 指令后且寄存器 PWRC 中的 PWDN 位为高时，系统进入深度休眠模式。在深度休眠模式中，CPU 停止，f_{SUB} 停止为外围功能提供时钟。若看门狗定时器功能使能，f_{LIRC} 继续运行。

控制寄存器

寄存器 SCC、HIRCC、HXTC 和 PWRC 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	—	FHIDEN	FSIDEN
HIRCC	—	—	—	—	—	—	HIRCF	HIRCEN
HXTC	—	—	—	—	—	—	HXTF	HXTEN
PWRC	PA_WAKE	—	—	TB0_WAKE	POF33V	LCMD	IO_ISO_EN	PWDN

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	—	FHIDEN	FSIDEN
R/W	R/W	R/W	R/W	—	R/W	—	R/W	R/W
POR	0	0	1	—	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 – 内部高频振荡器
1：HXT – 外部晶振
当该位置高时，即选择外部 HXT 振荡器作为系统时钟源，单片机实际时钟输入由连接到 OSC1 和 OSC2 引脚的晶体振荡器，以及寄存器 RF_XO2 中的 XODIV2 位和寄存器 RF_CLK1 中的 XCLKD2 位共同决定的。例如，如果

f_{HXT}=16MHz，XODIV2=0，XCLKD2=0，则 f_H=16MHz。

Bit 2　未定义，读为“0”

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。无论是 LIRC 被选择作为低速振荡器时钟源还是 WDT 功能使能，LIRC 振荡器都是由该位与 WDT 功能控制位共同控制的。如果该位被清零，但 WDT 功能使能，LIRC 振荡器也将使能。

- **HIRCC 寄存器**

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

Bit 7~2　未定义，读为“0”

Bit 1　**HIRCF**：HIRC 振荡器稳定标志位
0：HIRC 不稳定
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，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	—	—	—	—	—	—	HXTF	HXTEN
R/W	—	—	—	—	—	—	R	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2　未定义，读为“0”

Bit 1　**HXTF**：HXT 振荡器稳定标志位
0：HXT 不稳定
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

- **PWRC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PA_WAKE	—	—	TB0_WAKE	POF33V	LCMD	IO_ISO_EN	PWDN
R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	—	—	0	1	0	0	0

Bit 7 **PA_WAKE**：端口 A 使单片机从深度休眠模式中唤醒标志位
0：未发生端口 A 使单片机从深度休眠模式中唤醒
1：发生端口 A 使单片机从深度休眠模式中唤醒
该位表明单片机是否通过端口 A 从深度休眠模式中唤醒。在单片机进入深度休眠模式前，应正确配置 PAWU 寄存器。

Bit 6~5 未定义，读为“0”

Bit 4 **TB0_WAKE**：时基 0 使单片机从深度休眠模式中唤醒标志位
0：未发生时基 0 使单片机从深度休眠模式中唤醒
1：发生时基 0 使单片机从深度休眠模式中唤醒
该位表明单片机是否通过时基 0 中断从深度休眠模式中唤醒。在单片机进入深度休眠模式前，应正确配置 TB0C 寄存器和 TB0F 中断标志位。

Bit 3 **POF33V**：3V 电源域发生上电复位标志位
0：3V 电源域未发生上电复位
1：3V 电源域发生上电复位
当 3V 电源域发生上电复位时，该位由硬件置高。可通过应用程序或执行“CLR WDT”指令将此位清零。

Bit 2 **LCMD**：1.5V LDO 小电流模式选择
0：1.5V LDO 正常模式
1：1.5V LDO 小电流模式

Bit 1 **IO_ISO_EN**：I/O 隔离模式选择
0：I/O 处于正常工作模式
1：I/O 处于隔离模式 (I/O 状态保持不变)
在系统进入深度休眠模式前，该位必须置高来锁存输入 / 输出端口。所以当系统进入深度休眠模式时，输入 / 输出端口状态保持不变。当单片机从深度休眠模式中唤醒后，必须通过应用程序将该位清零来将输入 / 输出端口解除锁存。

Bit 0 **PWDN**：深度休眠模式控制
0：单片机正常工作
1：执行 HALT 指令后，单片机进入深度休眠模式 (1.5V LDO 由硬件关闭)
如果该位置高，且执行 HALT 指令后，单片机进入深度休眠模式，这时 1.5V LDO 将由硬件关闭。在单片机进入深度休眠模式前，用户应预先通过应用程序将 1.5V 域系统设定储存到数据存储器，并锁存住 I/O。

有几点注意事项需在此强调：

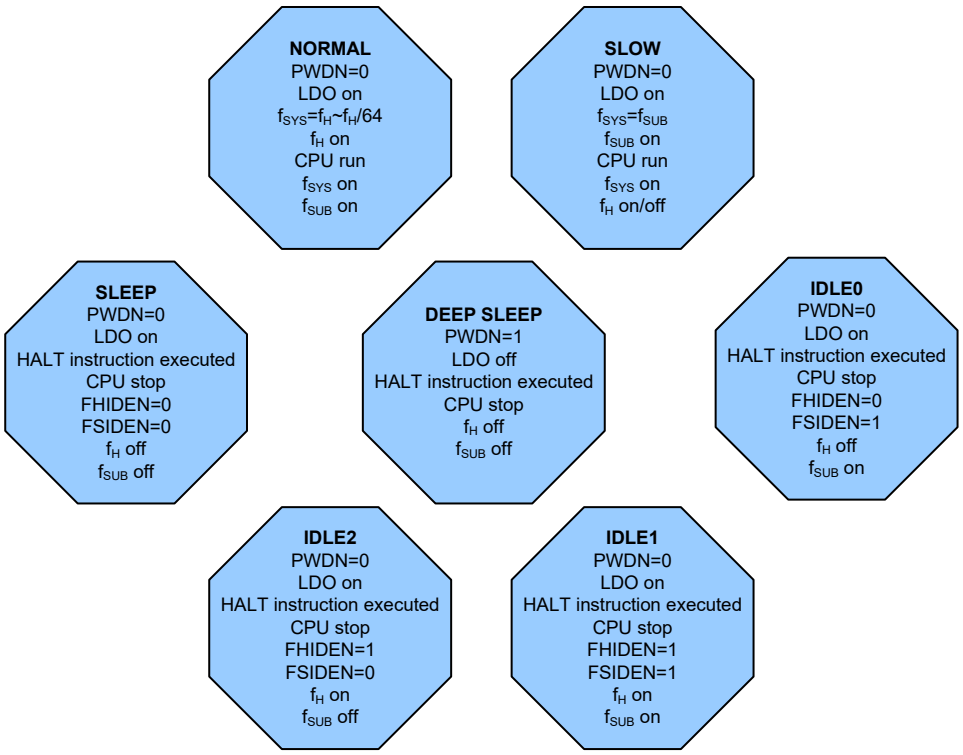
1. 在单片机进入深度休眠模式之前，需通过应用程序将系统设置预先备份。
2. 当 PWDN=1 且执行 HALT 指令后，单片机将进入深度休眠模式。
3. 当单片机进入深度休眠模式后，数据存储器将保持最新值。
4. 当单片机进入深度休眠模式后，RF_PWR 寄存器的 RF_PDB 位将由硬件自动清零。
5. 可通过以下情况将单片机从深度休眠模式中唤醒，唤醒后程序将从 0000h 开始执行。
   - PA 口下降沿
   - 时基 0 中断
6. 要想通过 PA 口下降沿将单片机从深度休眠模式中唤醒，则在单片机进入深度休眠模式之前应合理设置 PAWU 寄存器，并且将 PA_WAKE 位置高。

7. 要想通过 TB0 中断将单片机从深度休眠模式中唤醒，则在单片机进入深度休眠模式之前应先将 TB0F 中标标志位清零。注意，TB0 中断唤醒功能与该中断功能是否使能无关。
8. 当单片机进入深度休眠模式时，无论分频器 0 时钟源选择如何设置，TB0 时钟源将固定来自 f_{LIRC}。在深度休眠模式下，TB0 溢出周期将会是其预设值的一半。而在非深度休眠模式下，TB0 和 TB1 的第一次溢出周期将会是各自预设值的一半。
9. 若单片机是通过 TB0 中断从深度休眠模式中唤醒的，总的唤醒时间将是 (t_{RSTD}+(TB0 溢出周期预设值 / 2))。
10. 单片机从深度休眠模式唤醒后，所有寄存器都将被重置。因此需通过应用程序恢复系统设置。

工作模式切换

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较佳的性能 / 功耗比。用此方式，对单片机工作的性能要求不高的情况下，可使用较低频时钟以减少工作电流，在便携式应用上延长电池的使用寿命。

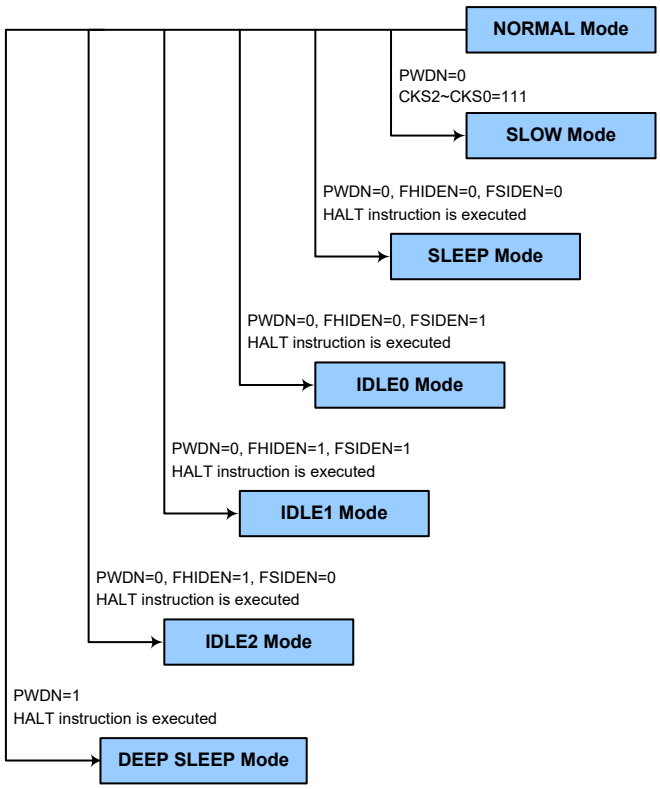
简单来说，正常模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现，而正常模式 / 低速模式与休眠模式 / 空闲模式或深度休眠模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入深度休眠模式、空闲模式或休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位以及寄存器 PWRC 中的 PWDN 位决定的。



正常模式切换到低速模式

系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 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。

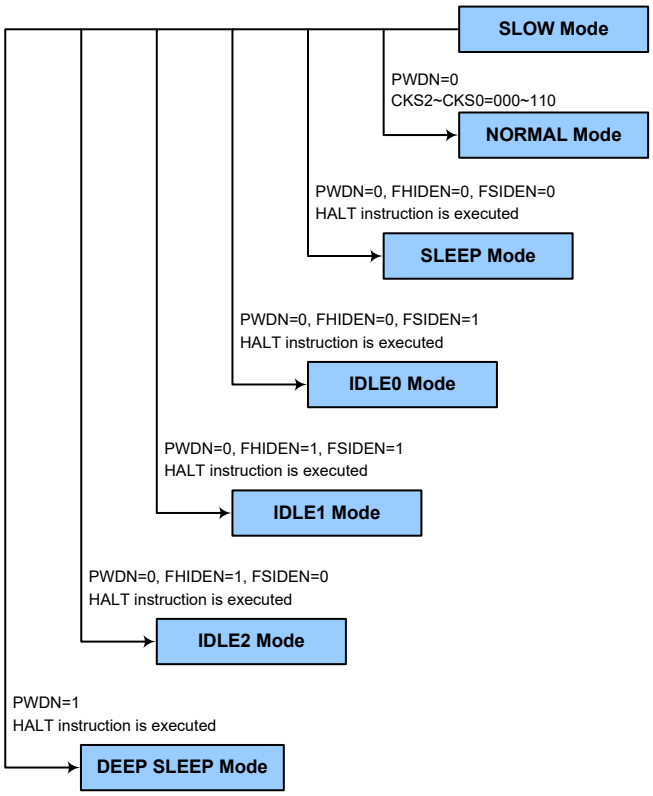
低速模式的时钟源来自 LIRC 振荡器，因此要求这些振荡器在所有模式切换动作发生前稳定下来。



低速模式切换到正常模式

在低速模式时系统时钟来自 f_{SUB}。切换回正常模式时，需设置 CKS2~CKS0 位为“000”~“110”使系统时钟从 f_{SUB} 切换到 f_H~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正常模式时，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指定在交流电气特性中。



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”，且寄存器 PWRC 中的 PWDN 位为“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。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 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 将被清零并停止计数。

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”，且寄存器 PWRC 中的 PWDN 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 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 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 将被清零并停止计数。

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”，且寄存器 PWRC 中的 PWDN 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 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 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 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 将被清零并停止计数。

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”，且寄存器 PWRC 中的 PWDN 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 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 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 将被清零并停止计数。

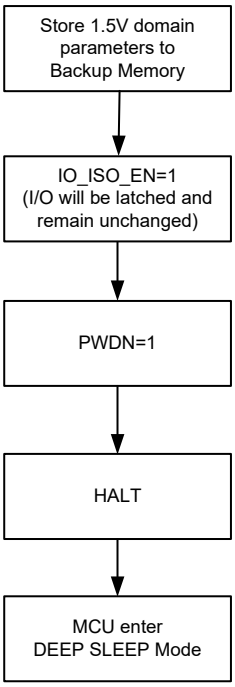
进入深度休眠模式

进入深度休眠模式的方法仅有一种——应用程序中执行“HALT”指令前需寄存器 PWRC 中的 PWDN 位为“1”。 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。
- 数据存储器中的内容和寄存器将保持当前值。建议预先储存重要的 1.5V 域系统设置到数据存储器中。
- 如果寄存器 PWRC 中的 IO_ISO_EN 位置高，输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

- 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 将被清零并停止计数。

以下流程图显示了单片机进入深度休眠模式后的程序步骤。



待机电流的注意事项

由于单片机进入深度休眠、休眠或空闲模式的主要原因是将 MCU 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进一步降低，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。还应注意的是，如果选择 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

单片机进入深度休眠模式、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：
  - ♦ PA 口下降沿
  - ♦ 系统中断
  - ♦ WDT 溢出

- 系统进入深度休眠模式之后，可以通过以下两种方式唤醒：
  - ♦ PA 口下降沿
  - ♦ 时基 0 中断

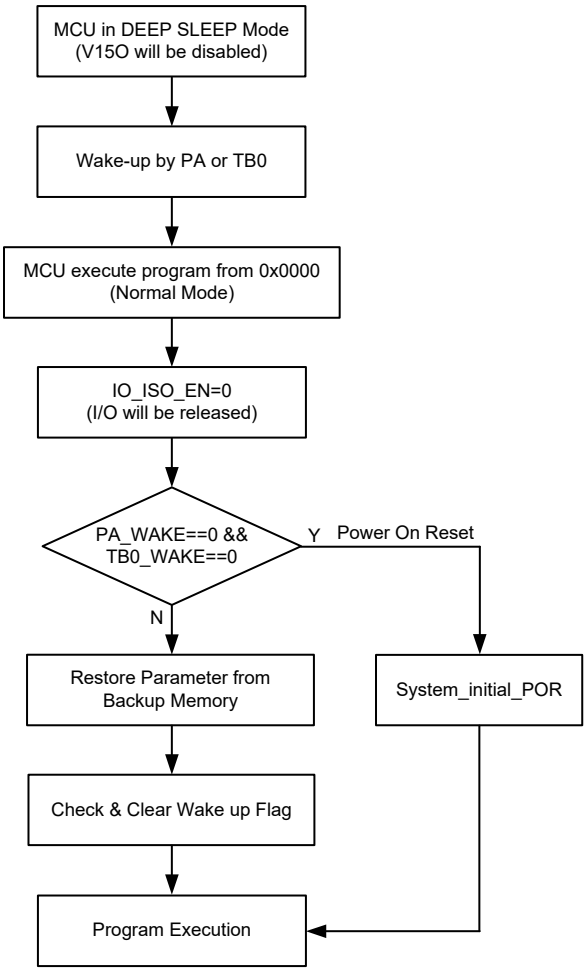
单片机执行 HALT 指令，PDF 将被置位；系统上电或执行清除看门狗的指令，PDF 将被清零。看门狗计数器溢出将会置位 TO 标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

PA 口中的每个引脚都可以通过 PAWU 寄存器使能下降沿唤醒功能。PA 端口使单片机系统从休眠或空闲模式唤醒后，程序将在“HALT”指令后继续执行。然而，当PA端口使单片机系统从深度休眠模式唤醒后，程序将从0000h继续执行。

如果系统是通过中断从休眠或空闲模式唤醒，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。如果系统是通过时基 0 中断从深度休眠模式唤醒，程序将从 0000h 继续执行。

如果在进入深度休眠、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。

以下流程图显示了单片机从深度休眠模式唤醒后的程序步骤。



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

看门狗定时器时钟源由内部低速 RC 振荡器 LIRC 提供。电压为 5V 时内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期随 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 2^8~2^{18} 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDTC 寄存器用于控制看门狗定时器功能的使能 / 除能及选择溢出周期。

- **WDTC 寄存器**

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

Bit 7~3 **WE4~WE0**：WDT 软件控制

如果配置选项中 WDT 设置为“始终使能”
10101：使能
01010：使能
其它值：单片机复位

如果配置选项中 WDT 设置为“由 WDT 控制寄存器设置”
10101：除能
01010：使能
其它值：单片机复位

通过选择 WDT 配置选项决定 WE4~WE0 位如何设置。

如果由于不利的环境因素使这些位变为除 10101B 和 01010B 的其它值，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 WRF 位将置为“1”。

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
具体描述见内部复位控制章节。

Bit 2 **LVRF**：LVR 复位标志位
具体描述见低电压复位章节。

Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
具体描述见低电压复位章节。

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

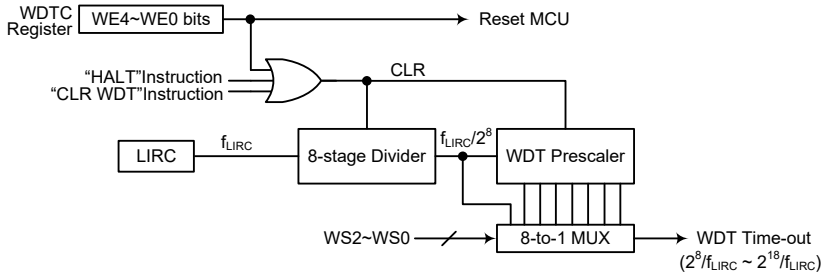
当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位可提供看门狗定时器使能 / 除能控制以及单片机复位控制。WDT 功能可由相关的配置选项以及下表中 WE4~WE0 位共同决定。如果 WE4~WE0 设置为除“01010B”和“10101B”以外的值时，单片机将在一段延迟时间 t_{SRESET} 后复位。上电后这些位初始化为“01010B”。

WDT 配置选项	WE4~WE0 位	WDT 功能
WDT 始终使能	01010B 或 10101B	使能
	其它值	单片机复位
WDT 控制寄存器控制	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 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于深度休眠、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。有三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 WDT 软件清除指令，而第三种是通过“HALT”指令。该系列单片机只使用一条清看门狗指令“CLR WDT”。因此只要执行“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。

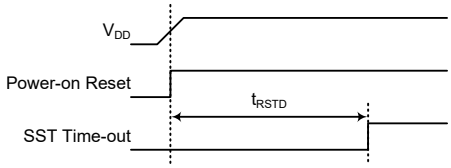
除了上电复位外，另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。另一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。

复位功能

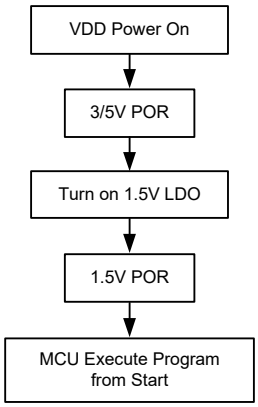
单片机的几种内部复位方式将在此处做具体介绍。

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



内部复位控制

内部复位控制寄存器 RSTC 用于为单片机在受到环境噪声干扰而异常工作时提供复位。如果 RSTC 寄存器的值被设置为除 01010101B 或 10101010B 以外的任何值，单片机会在一段延迟时间 t_{SRESET} 后发生复位。上电后寄存器的值为 01010101B。

RSTC7~RSTC0 位	复位功能
01010101B	无操作
10101010B	无操作
其它值	MCU 复位

内部复位功能控制

- **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**：复位功能控制位
01010101：无操作
10101010：无操作
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”。

- **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
具体描述见低电压复位章节。

Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
具体描述见低电压复位章节。

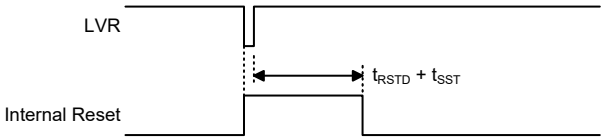
Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
具体描述见看门狗定时器控制寄存器章节。

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。当电压低于一个预先定义的特定电压时，单片机会复位。

LVR 功能的使能 / 除能控制取决于相关的配置选项，且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~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。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过 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。实际 V_{LVR} 参数值固定为 1.9V。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。正常执行时 LVR 会于深度休眠、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
01010101：1.9V
00110011：1.9V
10011001：1.9V
10101010：1.9V
其它值：MCU 复位—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。当低电压状况保持时间大于 t_{LVR} 后，响应复位。此时复位后的寄存器内容保持不变。
除了以上定义的四个低电压复位值外，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
具体描述见内部复位控制章节。

Bit 2 **LVRF**：LVR 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。

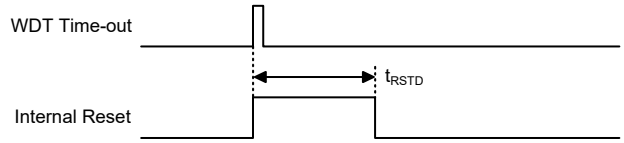
Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
具体描述见看门狗定时器控制寄存器章节。

IAP 复位

通过往 FC1 寄存器写数据 55H 可产生 IAP 复位。

正常运行时看门狗溢出复位

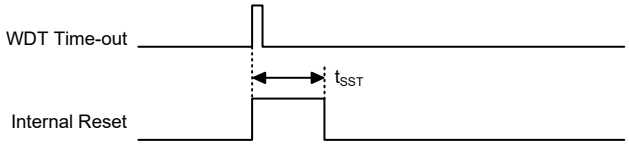
除了看门狗溢出标志位 TO 将被设为“1”之外，正常运行时看门狗溢出复位和 LVR 复位相同。



正常运行时看门狗溢出复位时序图

深度休眠 / 休眠 / 空闲时看门狗溢出复位

深度休眠、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 0 及 TO 位被设为 1 外，绝大部份的条件保持不变。图中 t_{SST} 的详细说明请参考交流电气特性。



深度休眠 / 休眠 / 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定时器等几种控制器操作控制。复位标志位如下所示：

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

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器，时基	复位后清零，看门狗定时器重新计数
定时器模块	所有定时器模块停止
输入 / 输出口	I/O 口设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

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

寄存器	BC68F2130	BC68F2140	BC68F2150	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (HALT)
IAR0	●	●	●	0000 0000	0000 0000	uuuu uuuu
MP0	●	●	●	0000 0000	0000 0000	uuuu uuuu
IAR1	●	●	●	0000 0000	0000 0000	uuuu uuuu
MP1L	●	●	●	0000 0000	0000 0000	uuuu uuuu
MP1H	●	●	●	0000 0000	0000 0000	uuuu uuuu
ACC	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	●	●	●	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBHP	●			---- -xxx	---- -uuu	---- -uuu
		●		---- xxxx	---- uuuu	---- uuuu
			●	---x xxxx	---u uuuu	---u uuuu
STATUS	●	●	●	xx00 xxxx	xx1u uuuu	uu11 uuuu
IAR2	●	●	●	0000 0000	0000 0000	uuuu uuuu
MP2L	●	●	●	0000 0000	0000 0000	uuuu uuuu
MP2H	●	●	●	0000 0000	0000 0000	uuuu uuuu
RSTFC	●	●	●	---- 0x00	---- uuuu	---- uuuu
INTC0	●	●	●	-000 0000	-000 0000	-uuu uuuu
INTC1	●	●	●	0000 0000	0000 0000	uuuu uuuu
PA	●			--11 1111	--11 1111	--uu uuuu
		●	●	1111 1111	1111 1111	uuuu uuuu
PAC	●			--11 1111	--11 1111	--uu uuuu
		●	●	1111 1111	1111 1111	uuuu uuuu
PAPU	●			--00 0000	--00 0000	--uu uuuu
		●	●	0000 0000	0000 0000	uuuu uuuu
PAWU	●			--00 0000	--00 0000	--uu uuuu
		●	●	0000 0000	0000 0000	uuuu uuuu
PB	●			--11 ----	--11 ----	--uu ----
		●	●	--11 1111	--11 1111	--uu uuuu
PBC	●			--11 ----	--11 ----	--uu ----
		●	●	--11 1111	--11 1111	--uu uuuu
PBPU	●			--00 ----	--00 ----	--uu ----
		●	●	--00 0000	--00 0000	--uu uuuu
INTEG	●	●	●	---- 0000	---- 0000	---- uuuu
SCC	●	●	●	001- 0-00	001- 0-00	uuu- u-uu
HIRCC	●	●	●	---- --01	---- --01	---- --uu

寄存器	BC68F2130	BC68F2140	BC68F2150	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (HALT)
HXTC	●	●	●	---- --00	---- --00	---- --uu
LVDC	●	●	●	--00 -000	--00 -000	--uu -uuu
LVRC	●	●	●	0101 0101	0101 0101	uuuu uuuu
WDTC	●	●	●	0101 0011	0101 0011	uuuu uuuu
RSTC	●	●	●	0101 0101	0101 0101	uuuu uuuu
PSCR0	●	●	●	---- --00	---- --00	---- --uu
PSCR1	●	●	●	---- --00	---- --00	---- --uu
PWRC	●	●	●	0--0 1000	u--u uuuu	u--u uuuu
RF_PWR	●	●	●	---- ---0	---- ---0	---- ---u
MFI0	●	●	●	--00 --00	--00 --00	--uu --uu
MFI1	●	●	●	--00 --00	--00 --00	--uu --uu
TB0C	●	●	●	0000 -000	0000 -000	uuuu -uuu
TB1C	●	●	●	0--- -000	0--- -000	u--- -uuu
PTMAH	●	●	●	---- --00	---- --00	---- --uu
PTMAL	●	●	●	0000 0000	0000 0000	uuuu uuuu
PTMC0	●	●	●	0000 0---	0000 0---	uuuu u---
PTMC1	●	●	●	0000 0000	0000 0000	uuuu uuuu
PTMDH	●	●	●	---- --00	---- --00	---- --uu
PTMDL	●	●	●	0000 0000	0000 0000	uuuu uuuu
PTMRPH	●	●	●	---- --00	---- --00	---- --uu
PTMRPL	●	●	●	0000 0000	0000 0000	uuuu uuuu
CTMC0	●	●	●	0000 0000	0000 0000	uuuu uuuu
CTMC1	●	●	●	0000 0000	0000 0000	uuuu uuuu
CTMDL	●	●	●	0000 0000	0000 0000	uuuu uuuu
CTMDH	●	●	●	---- --00	---- --00	---- --uu
CTMAL	●	●	●	0000 0000	0000 0000	uuuu uuuu
CTMAH	●	●	●	---- --00	---- --00	---- --uu
FC0	●	●	●	0000 0000	0000 0000	uuuu uuuu
FC1	●	●	●	0000 0000	0000 0000	uuuu uuuu
FARL	●	●	●	0000 0000	0000 0000	uuuu uuuu
FARH	●			---- -000	---- -000	---- -uuu
		●		---- 0000	---- 0000	---- uuuu
			●	---0 0000	---0 0000	---u uuuu
FD0L	●	●	●	--00 0000	--00 0000	--uu uuuu
FD0H	●	●	●	0000 0000	0000 0000	uuuu uuuu
FD1L	●	●	●	0000 0000	0000 0000	uuuu uuuu
FD1H	●	●	●	0000 0000	0000 0000	uuuu uuuu
FD2L	●	●	●	0000 0000	0000 0000	uuuu uuuu

寄存器	BC68F2130	BC68F2140	BC68F2150	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (HALT)
FD2H	●	●	●	0000 0000	0000 0000	uuuu uuuu
FD3L	●	●	●	0000 0000	0000 0000	uuuu uuuu
FD3H	●	●	●	0000 0000	0000 0000	uuuu uuuu
MFI2	●	●	●	-000 -000	-000 -000	-uuu -uuu
RF_OPER	●	●	●	--00 ---0	--00 ---0	--uu ---u
RF_CLK1	●	●	●	10-0 ---0	10-0 ---0	uu-u ---u
RF_CLK2	●	●	●	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL1	●	●	●	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL2	●	●	●	0011 1111	0011 1111	uuuu uuuu
RF_FIFO_CTRL3	●	●	●	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL4	●	●	●	0--- 0001	0--- 0001	u--- uuuu
RF_MOD1	●	●	●	1001 0000	1001 0000	uuuu uuuu
RF_MOD2	●	●	●	---- -001	---- -001	---- -uuu
RF_OPMOD	●	●	●	---- -000	---- -000	---- -uuu
RF_SX1	●	●	●	-011 0110	-011 0110	-uuu uuuu
RF_SX2	●	●	●	0000 1010	0000 1010	uuuu uuuu
RF_SX3	●	●	●	1101 0111	1101 0111	uuuu uuuu
RF_SX4	●	●	●	---- 0011	---- 0011	---- uuuu
RF_CP3	●	●	●	1100 1010	1100 1010	uuuu uuuu
RF_OD1	●	●	●	0000 0100	0000 0100	uuuu uuuu
RF_OD3	●	●	●	--01 -100	--01 -100	--uu -uuu
RF_VCO1	●	●	●	0001 0000	0001 0000	uuuu uuuu
RF_VCO2	●	●	●	0-10 0000	0-10 0000	u-uu uuuu
RF_TX2	●	●	●	1101 -000	1101 -000	uuuu -uuu
RF_DFC_CAL	●	●	●	00-0 0000	00-0 0000	uu-u uuuu
RF_LDO	●	●	●	00011000	00011000	uuuu uuuu
RF_XO1	●	●	●	10-1 0101	10-1 0101	uu-u uuuu
RF_XO2	●	●	●	-01- 0011	-01- 0011	-uu- uuuu
PAS0	●	●	●	0000 0000	0000 0000	uuuu uuuu
PAS1	●			---- 0000	---- 0000	---- uuuu
		●	●	--00 0000	--00 0000	--uu uuuu

注：“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

Holtek 单片机的输入 / 输出口控制具有很大的灵活性。大部分引脚可在用户程序控制下被设定为输入或输出。所有引脚的上拉电阻设置以及指定引脚的唤醒设置也都由软件控制，这些特性也使得此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~PB 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	—	—	PA5	PA4	PA3	PA2	PA1	PA0
PAC	—	—	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	—	—	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	—	—	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	PB5	PB4	—	—	—	—
PBC	—	—	PBC5	PBC4	—	—	—	—
PBPU	—	—	PBPU5	PBPU4	—	—	—	—

“—”：未定义，读为“0”

I/O 逻辑功能寄存器列表 – BC68F2130

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	—	—	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0

“—”：未定义，读为“0”

I/O 逻辑功能寄存器列表 – BC68F2140/BC68F2150

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，当引脚规划为数字输入时，可由内部连接到一个上拉电阻。这些上拉电阻可通过相应的上拉控制寄存器 PAPU~PBPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

需要注意的是，当 I/O 引脚设为数字输入或 NMOS 输出时，上拉功能才会受 PxPU 控制开启，其它状态下上拉功能不可用。

- **PxPU 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PxPU7	PxPU6	PxPU5	PxPU4	PxPU3	PxPU2	PxPU1	PxPU0
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

PxPUn：I/O Px 口上拉电阻控制位
0：除能
1：使能

PxPUn 位用于控制上拉电阻功能。这里的 x 可以是端口 A 和 B。但是，每个单片机的每个 I/O 端口实际有效位可能不同。

PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入深度休眠、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。

需要注意的是，只有当引脚选作通用输入类型且单片机处于深度休眠、休眠或空闲模式时，唤醒功能才会受 PAWU 控制开启，其它状态下此唤醒功能不可用。

- **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

PAWUn：PA 口引脚唤醒功能控制位
0：除能
1：使能

PAWUn 用来控制端口 A 引脚唤醒功能。但是，每个单片机实际有效位可能不同。

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~PBC，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：I/O Px 口类型选择位
0：输出
1：输入
PxCn 位用于控制引脚类型。这里的 x 可以是端口 A 和 B。但是，每个单片机的每个 I/O 端口实际有效位可能不同。

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，这些寄存器可以用来选择共用引脚的特定功能。

要注意的最重要一点是，确保所需的引脚共用功能被正确地选择和取消。对于大多数引脚共用功能来说，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，然后再配置相应的外围功能设置以使能外围功能。但是，当设置相关引脚共用控制字段时，应特别注意一些与通用 I/O 功能共用相同的引脚共用控制配置的数字输入引脚，例如 INTn、xTCKn、xTPnI 等。要选择这些引脚功能，除了必要的引脚共用控制和前面提到的外围功能设置，还必须设置输入 / 输出端口控制寄存器的相应位，将这些引脚设置为输入。要正确地取消引脚共用功能，首先应除能外围功能，然后再修改相应的引脚共用控制寄存器以选择其它的共用功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1 (BC68F2130)	—	—	—	—	PAS13	PAS12	PAS11	PAS10
PAS1 (BC68F2140/ BC68F2150)	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10

引脚共用功能选择寄存器列表

- **PAS0 寄存器**

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
00/01/10/11：PA3/INT0

Bit 5~4 **PAS05~PAS04**：PA2 引脚共用功能选择
00/10/11：PA2
01：PTP

Bit 3~2 **PAS03~PAS02**：PA1 引脚共用功能选择
00/01/10/11：PA1/PTCK

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

- **PAS1 寄存器 – BC68F2130**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PAS13	PAS12	PAS11	PAS10
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 **PAS13~PAS12**：PA5 引脚共用功能选择
00/10/11：PA5
01：CTP

Bit 1~0 **PAS11~PAS10**：PA4 引脚共用功能选择
00/01/10/11：PA4/INT1

- **PAS1 寄存器 – BC68F2140/BC68F2150**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
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”

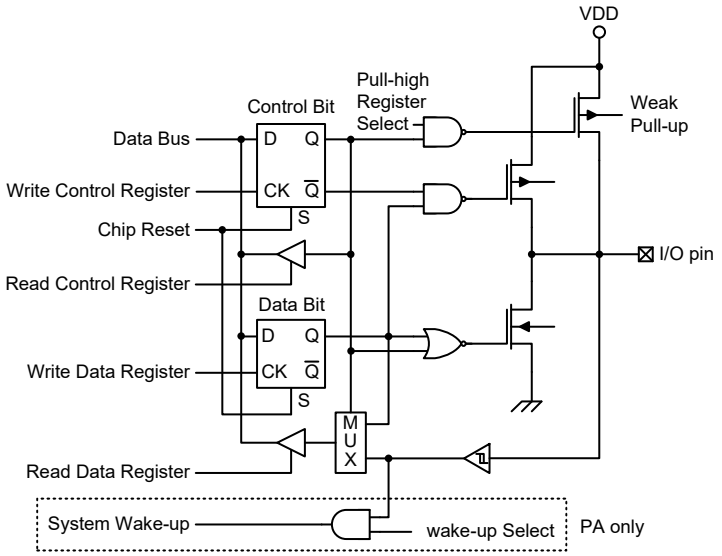
Bit 5~4 **PAS15~PAS14**：PA6 引脚共用功能选择
00/10/11：PA6
01：CTPB

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

Bit 1~0 **PAS11~PAS10**：PA4 引脚共用功能选择
00/01/10/11：PA4/INT1

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。



逻辑功能输入 / 输出结构

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。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。单片机处于深度休眠、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 外加的输入输出引脚，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简易型和周期型定时器章节。

简介

该系列单片机包含两个 TM，每个 TM 可被划分为一个特定的类型，即简易型 TM 或和周期型 TM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍简易型和周期型 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	CTM	PTM
定时 / 计数器	√	√
捕捉输入	—	√
比较匹配输出	√	√
PWM 输出	√	√
单脉冲输出	—	√
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

CTM	PTM
10-bit CTM	10-bit PTM

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。用户选择内部时钟或外部时钟来驱动内部 TM 计数器。

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTM 控制寄存器的 xTCK2~xTCK0 位，选择所需的时钟源，其中 x 代表 C 或 P。时钟源可以选择来自系统时钟 f_{SYS} 的分频比或内部高速时钟 f_H 或 f_{SUB} 时钟源或外部 xTCK 引脚。xTCK 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

简易型和周期型 TM 都有两个内部中断，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 输出引脚的状态。

TM 外部引脚

无论哪种类型的 TM，都有一个 TM 输入引脚 xTCK。xTM 输入引脚 xTCK 作为 xTM 时钟源输入脚，通过设置 xTMC0 寄存器中的 xTCK2~xTCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。xTCK 输入引脚可选择上升沿有效或下降沿有效。PTCK 引脚还可分别用作 PTM 单脉冲模式的外部触发引脚。

另一种 PTM 输入引脚 PTPI 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 PTMC1 寄存器中的 PTIO1~PTIO0 位来选择有效边沿类型。除了 PTPI 引脚外，PTCK 引脚也可用作 PTM 捕捉输入模式的外部触发引脚。

每个 TM 都有两个输出引脚 xTP 和 xTPB。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 xTP 和 xTPB 输出引脚也被 TM 用来产生 PWM 输出波形。

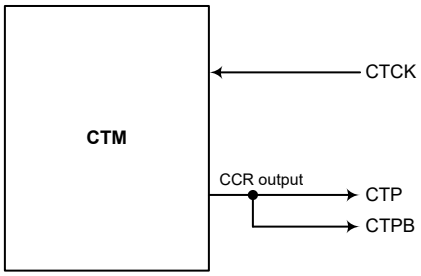
因 TM 输入和输出引脚与其它功能共用，故在使用 TM 输入和输出功能时需要先行设置相应的引脚共用功能选择寄存器，具体见引脚共用功能章节描述。

CTM		PTM	
输入	输出	输入	输出
CTCK	CTP, CTPB	PTCK, PTPI	PTP, PTPB

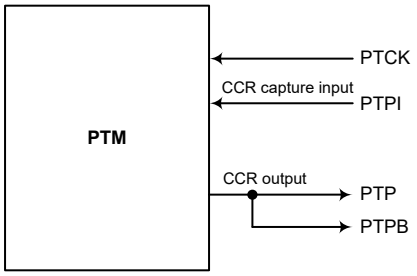
TM 外部引脚

TM 输入 / 输出引脚选择

选择作为 TM 输入 / 输出引脚还是其它共用引脚功能是通过设置相关引脚共用选择寄存器来实现的。每个 TM 输入 / 输出引脚都有对应的引脚共用选择位。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。



CTM 功能引脚框图

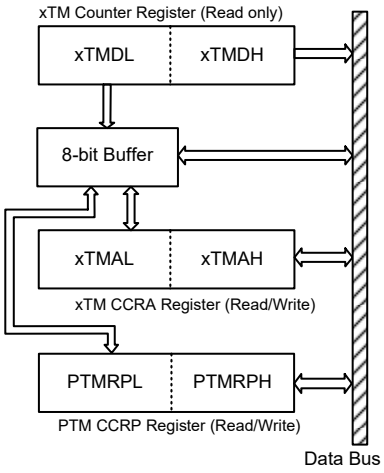


PTM 功能引脚框图

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特殊的方式。建议使用“MOV”指令按照以下步骤访问 CCRA 和 CCRP 低字节寄存器，即 xTMAL 和 PTMRPL，否则可能导致无法预期的结果。

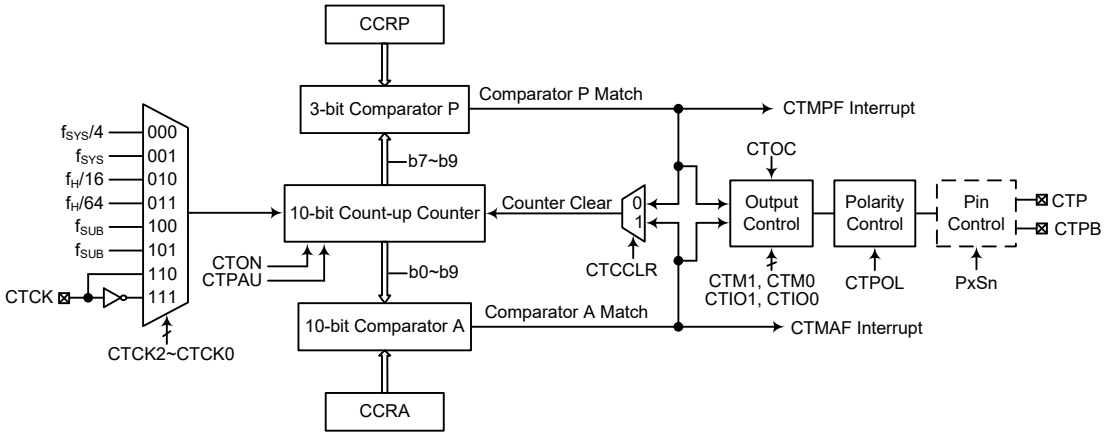


读写流程如下步骤所示：

- 写数据至 CCRA 或 CCRP
  - ♦ 步骤 1. 写数据至低字节寄存器 xTMAL 或 PTMRPL
    – 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 xTMAH 或 PTMRPH
    – 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 xTMDH、xTMAH 或 PTMRPH 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 xTMDL、xTMnAL 或 PTMRPL 读取数据
    – 注意，此时读取 8-bit 缓存器中的数据。

简易型 TM – CTM

简易型 TM 包括三种工作模式，即比较匹配输出，定时 / 事件计数器和 PWM 输出模式。简易型 TM 也由一个外部输入脚控制并驱动两个外部输出脚。



简易型 TM 方框图

简易型 TM 操作

简易型 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 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 3 位的，与计数器的高 3 位比较；而 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10 位计数器值的唯一方法是使 CTON 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，通常情况下会产生 TM 中断信号。简易型 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，也可以控制两个输出脚。所有工作模式的设定都是通过设置相关内部寄存器来实现的。

简易型 TM 寄存器介绍

简易型 TM 的所有操作由一系列寄存器控制。包含一对只读寄存器用来存放 10 位计数器的值，一对读 / 写寄存器存放 10 位 CCRA 的值，剩下两个控制寄存器设置不同的操作和控制模式以及 3 位的 CCRP 值。

寄存器名称	位							
	7	6	5	4	3	2	1	0
CTMC0	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
CTMC1	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
CTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMDH	—	—	—	—	—	—	D9	D8
CTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMAH	—	—	—	—	—	—	D9	D8

10-bit 简易型 TM 寄存器列表

- **CTMC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
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 **CTPAU**：CTM 计数器暂停控制位
0：运行
1：暂停
通过设置此位为高可暂停计数器。此位清零可恢复正常计数器操作。在暂停情况下 CTM 将保持上电并继续产生功耗。当此位由低到高转换，计数器将保留其剩余值，当其电平再变为低时从此位开始继续计数。

Bit 6~4 **CTCK2~CTCK0**：CTM 计数器时钟选择位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：CTCK 上升沿时钟
111：CTCK 下降沿时钟
此三位用于选择 CTM 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 **CTON**：CTM 计数器开 / 关控制位
0：关闭
1：开启
此位控制 CTM 的总开关功能。设置此位为高可使能计数器使其运行，清零此位则除能 CTM。清零此位将停止计数器并关闭 CTM 减少功耗。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转变为高。
若 CTM 处于比较匹配输出模式或 PWM 输出模式，当 CTON 位经由低到高转换时， CTM 输出脚将复位至 CTOC 位指定的初始值。

Bit 2~0 **CTRP2~CTRP0**：CTM CCRP 3-bit 寄存器，与 CTM 计数器 bit 9~bit 7 比较
比较器 P 匹配周期 =
000：1024 个 CTM 时钟
001：128 个 CTM 时钟
010：256 个 CTM 时钟
011：384 个 CTM 时钟
100：512 个 CTM 时钟
101：640 个 CTM 时钟
110：768 个 CTM 时钟
111：896 个 CTM 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 CTCCLR 位设定为 0 时，选中该比较结果清除内部计数器。CTCCLR 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。

- **CTMC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
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 **CTM1~CTM0**：CTM 工作模式选择位
00：比较匹配输出模式
01：未定义
10：PWM 输出模式
11：定时 / 计数器模式
这两位设置 CTM 需要的工作模式。为保证操作的可靠性，CTM 应在 CTM1 和 CTM0 位有任何改变前先关掉。在定时 / 计数器模式下，CTM 输出引脚状态未定义。

Bit 5~4 **CTIO1~CTIO0**：CTM 外部引脚功能选择位
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
00：PWM 输出无效状态
01：PWM 输出有效状态
10：PWM 输出
11：未定义
定时 / 计数器模式
未使用
这两位用于确定在达到某些条件时 CTM 输出引脚状态应如何改变。这些位的功能选择取决于当下 CTM 的运行模式。
在比较匹配输出模式下，CTIO1 和 CTIO0 位决定当比较器 A 比较匹配输出发生时 CTM 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 CTM 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，输出将不会改变。CTM 输出脚的初始值通过 CTOC 位设置取得。注意，由 CTIO1 和 CTIO0 位得到的输出电平必须与通过 CTOC 位设置的初始值不同，否则当比较匹配发生时，CTM 输出脚将不会发生变化。在 CTM 输出脚改变状态后，通过 CTON 位由低到高电平的转换可复位至初始值。
在 PWM 输出模式，CTIO1 和 CTIO0 用于决定比较匹配条件发生时怎样改变 CTM 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 CTM 关闭时改变 CTIO1 和 CTIO0 位的值是很有必要的。若在 CTM 运行时改变 CTIO1 和 CTIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **CTOC**：CTP 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
0：低有效
1：高有效
此位为 CTM 输出脚的输出控制位。它取决于 CTM 此时正运行于比较匹配输出模式还是 PWM 输出模式。若 CTM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 CTM 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。

Bit2 **CTPOL**：CTP 输出极性控制位
0：同相
1：反相
此位控制 CTM 输出脚的极性。此位为高时 CTM 输出脚反相，为低时 CTM 输出脚同相。若 CTM 处于定时 / 计数器模式时其不受影响。

Bit 1 **CTDPX**：CTM 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 **CTCCLR**：选择 CTM 计数器清零条件位
0：CTM 比较器 P 匹配
1：CTM 比较器 A 匹配
此位用于选择清除计数器的方法。简易型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用于清除内部计数器。CTCCLR 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 的位都被清除为 0 时才能生效。CTCCLR 位在 PWM 输出模式时未使用。

- **CTMDL 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**：CTM 计数器低字节寄存器 bit 7~bit 0
CTM 10-bit 计数器 bit 7~bit 0

- **CTMDH 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**：CTM 计数器高字节寄存器 bit 1~bit 0
CTM 10-bit 计数器 bit 9~bit 8

- **CTMAL 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**：CTM CCRA 低字节寄存器 bit 7~bit 0
CTM 10-bit CCRA bit 7~bit 0

- **CTMAH 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**：CTM CCRA 高字节寄存器 bit 1~bit 0
CTM 10-bit CCRA bit 9~bit 8

简易型 TM 工作模式

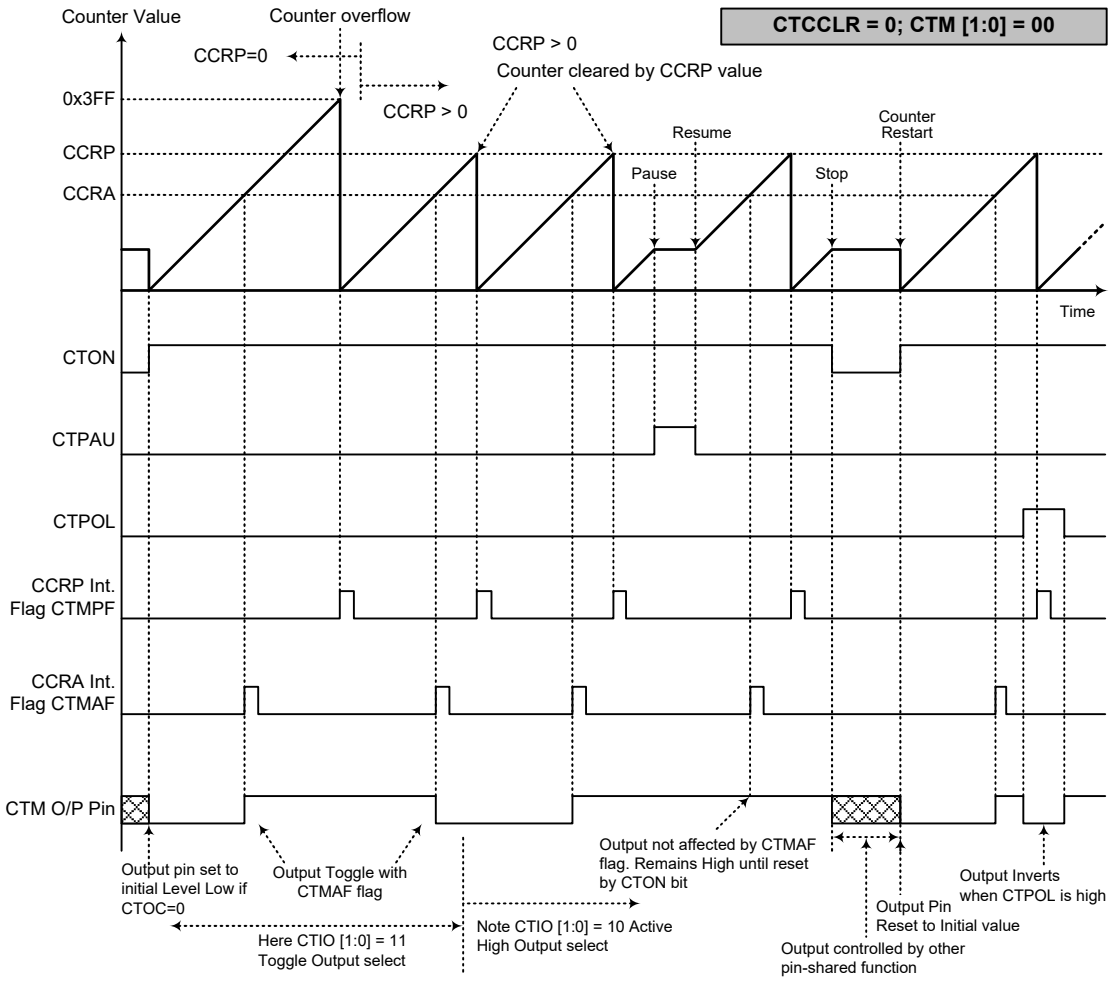
简易型 TM 有三种工作模式，即比较匹配输出模式，PWM 输出模式或定时 / 计数器模式。通过设置 CTMC1 寄存器的 CTM1 和 CTM0 位选择任意工作模式。

比较匹配输出模式

要工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 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 CTCCLR 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 CTMAF 和 CTMPF 将分别置起。

如果 CTMC1 寄存器的 CTCCLR 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 CTMAF 中断请求标志产生。所以当 CTCCLR 为高时，不产生 CTMPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 CTMAF 请求标志。

正如该模式名所言，当比较匹配发生后，CTM 输出脚状态改变。当比较器 A 比较匹配发生后 CTMAF 标志产生时，CTM 输出脚状态改变。比较器 P 比较匹配发生时产生的 CTMPF 标志不影响 CTM 输出脚。CTM 输出脚状态改变方式由 CTMC1 寄存器中 CTIO1 和 CTIO0 位决定。当比较器 A 比较匹配发生时，CTIO1 和 CTIO0 位决定 CTM 输出脚输出高，低或翻转当前状态。在 CTON 位由低到高电平的变化后，CTM 输出脚初始状态为 CTOC 位所指定的电平。注意，若 CTIO1 和 CTIO0 位同时为 0 时，引脚输出不变。

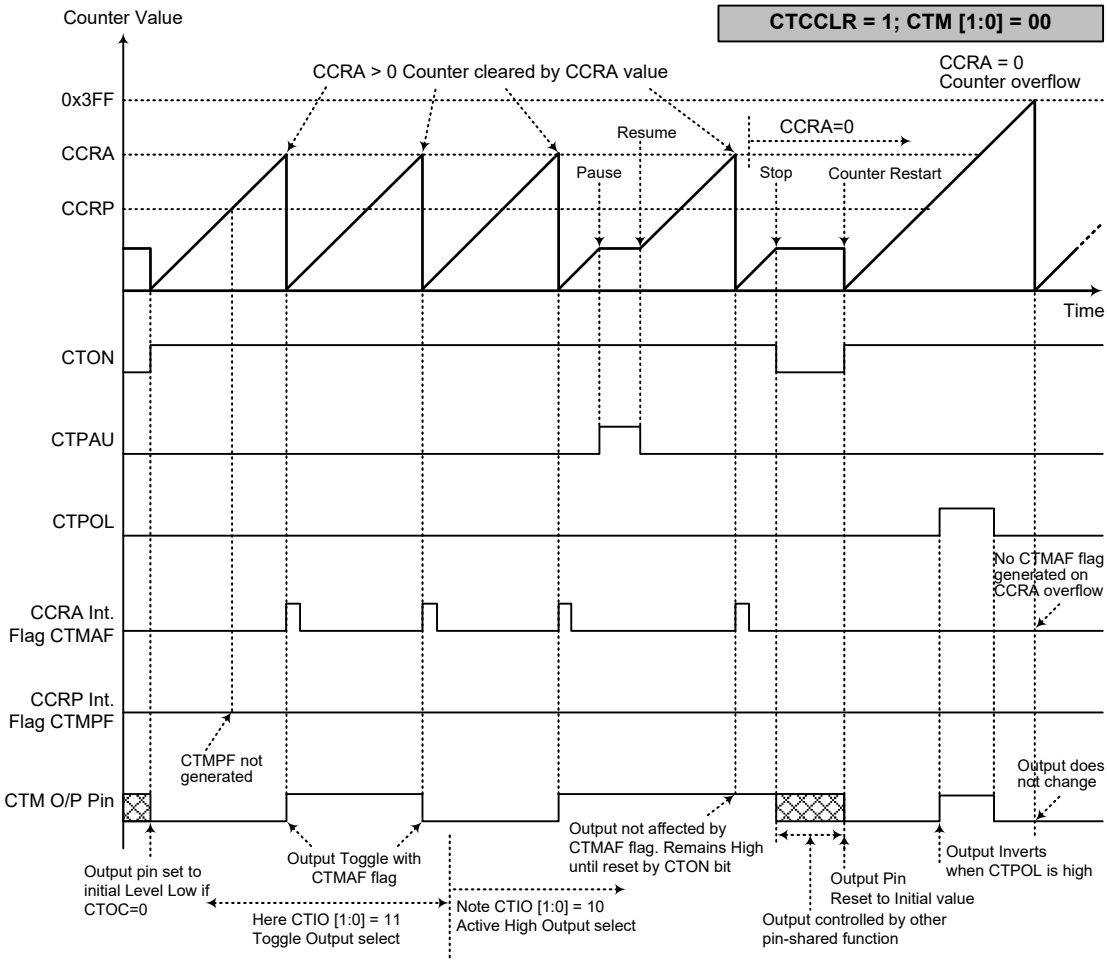


比较匹配输出模式 – CTCCLR=0

注：1. CTCCLR=0 时，比较器 P 匹配将清除计数器。

2. CTM 输出引脚仅由 CTMAF 标志位控制

3. 输出引脚通过 CTON 位上升沿复位为初始值



比较匹配输出模式 – CTCCLR=1

注：1. CTCCLR=1 时，比较器 A 匹配将清除计数器
2. CTM 输出引脚控制仅由 CTMAF 标志位控制
3. 输出引脚通过 CTON 上升沿复位至初始值
4. 当 CTCCLR=1 时不产生 CTMPF 标志位

定时 / 计数器模式

要工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 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 CTM 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帮助理解此功能。该模式中未使用的 CTM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

要工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 位需要设置为“10”。CTM 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 CTM 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控，其波形的选择就较为灵活。在 PWM 输出模式中，CTCCLR 位不影响 PWM 操作。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 CTMC1 寄存器的 CTDPX 位。所以 PWM 波形频率和占空比由 CCRA 和 CCRP 寄存器共同决定。

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。CTMC1 寄存器中的 CTOC 位决定 PWM 波形的极性，CTIO1 和 CTIO0 位使能 PWM 输出或将 CTM 输出脚置为逻辑高或逻辑低。CTPOL 位对 PWM 输出波形的极性取反。

- **CTM，PWM 输出模式，边沿对齐模式，CTDPX=0**

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

若 f_{SYS}=8MHz，CTM 时钟源选择 f_{SYS}/4，CCRP=100b，CCRA=128，

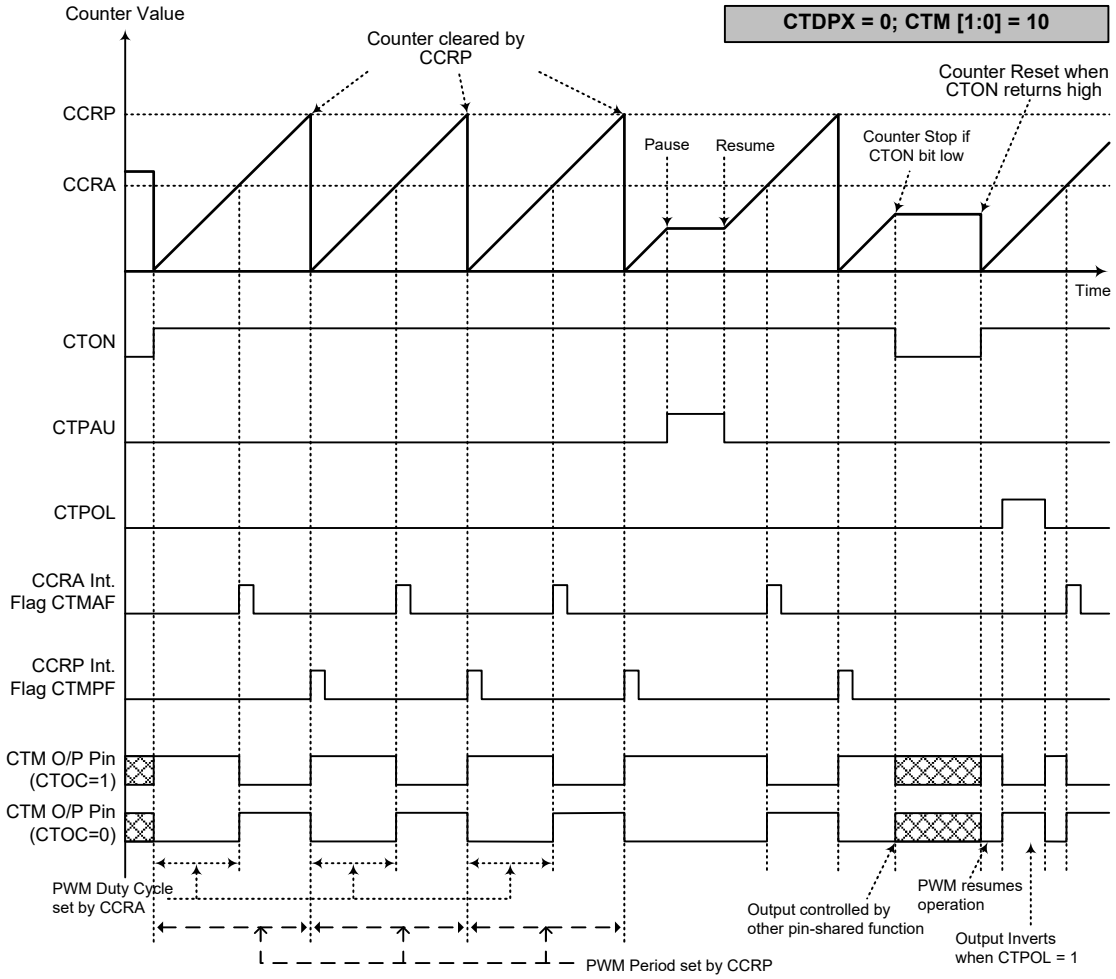
CTM PWM 输出频率 = (f_{SYS}/4) /512=f_{SYS}/2048=3.9063kHz，duty=128 /512=25%。

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%。

- **CTM，PWM 输出模式，边沿对齐模式，CTDPX=1**

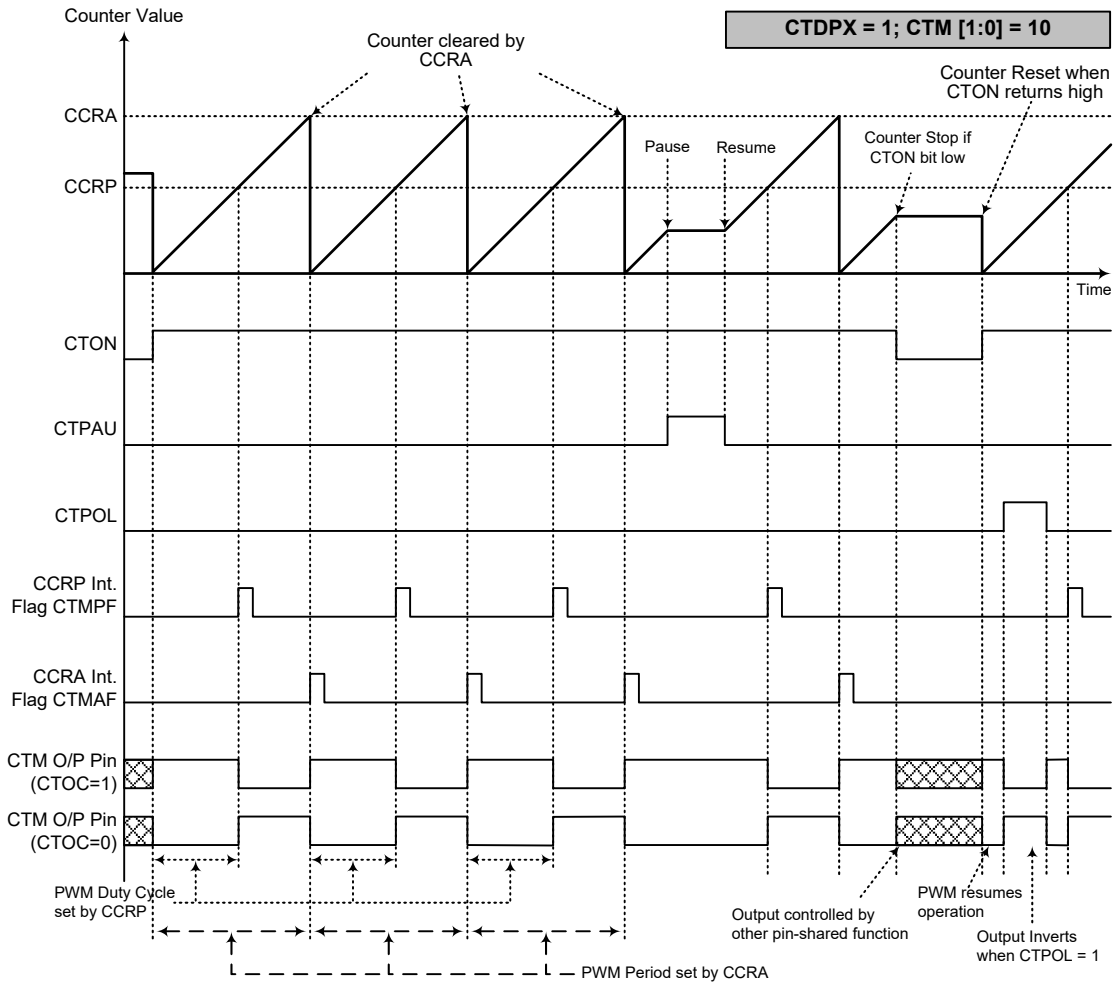
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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



PWM 输出模式 – CTDPX=0

注：1. 这里的 CTDPX=0 – 计数器由 CCRP 清除
2. 计数器清除设置 PWM 周期
3. 即使在 CTIO[1:0]=00 或 01 时，内部 PWM 功能继续运行
4. CTCCLR 位对 PWM 操作没有影响



PWM 输出模式 – CTDPX=1

注：1. 这里的 CTDPX=1 – 计数器由 CCRA 清除

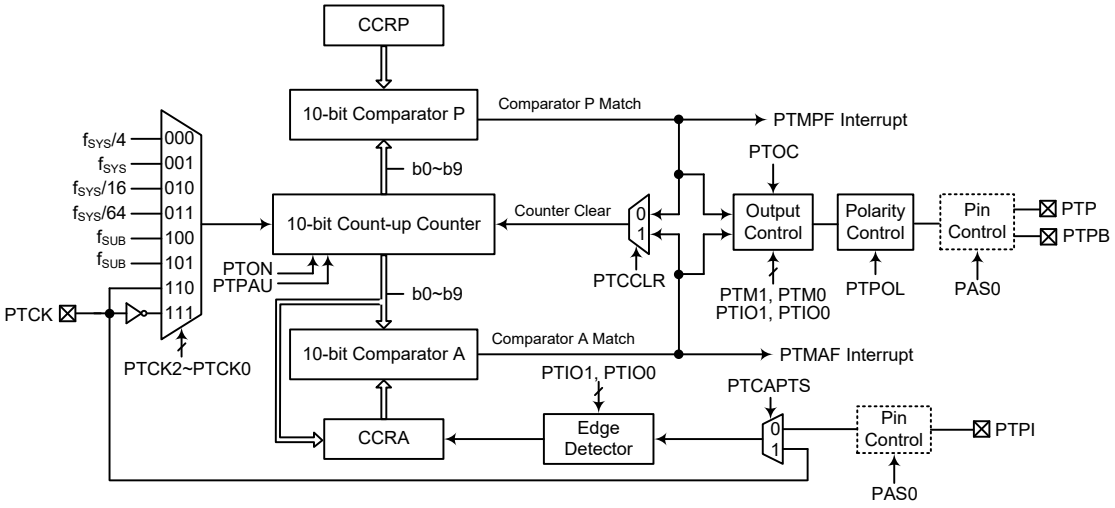
2. 计数器清除设置 PWM 周期

3. 即使在 CTIO[1:0]=00 或 01 时，内部 PWM 功能继续运行

4. CTCCLR 位对 PWM 操作无影响

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 由两个外部输入脚控制并驱动两个外部输出脚。



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 和 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10 位计数器值的唯一方法是使 PTON 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 PTM 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
PTMC0	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
PTMC1	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
PTMDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMDH	—	—	—	—	—	—	D9	D8
PTMAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMAH	—	—	—	—	—	—	D9	D8
PTMRPL	PTRP7	PTRP6	PTRP5	PTRP4	PTRP3	PTRP2	PTRP1	PTRP0

寄存器名称	位							
	7	6	5	4	3	2	1	0
PTMRPH	—	—	—	—	—	—	PTRP9	PTRP8

10-bit 周期型 TM 寄存器列表

- **PTMC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTPAU**：PTM 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，PTM 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 **PTCK2~PTCK0**：选择 PTM 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：PTCK 上升沿
111：PTCK 下降沿
此三位用于选择 PTM 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 **PTON**：PTM 计数器开 / 关控制位
0：关闭
1：开启
此位控制 PTM 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 PTM。清零此位将停止计数器并关闭 PTM 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 PTM 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式，当 PTON 位经由低到高转换时，PTM 输出脚将复位至 PTOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

- **PTMC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
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 **PTM1~PTM0**：选择 PTM 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 PTM 需要的工作模式。为了确保操作可靠，PTM 应在 PTM1 和

PTM0 位有任何改变前先关掉。在定时 / 计数器模式，PTM 输出引脚状态为未定义。

Bit 5~4 **PTIO1~PTIO0**：选择 PTM 外部引脚功能位

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 PTPI 或 PTCK 上升沿输入捕捉
- 01：在 PTPI 或 PTCK 下降沿输入捕捉
- 10：在 PTPI 或 PTCK 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 PTM 输出脚如何改变状态。这两位值的选择决定 PTM 运行在哪种模式下。

在比较匹配输出模式下，PTIO1 和 PTIO0 位决定当从比较器 A 比较匹配输出发生时 PTM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM 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。PTM 输出脚的初始值通过 PTMC1 寄存器的 PTOC 位设置取得。注意，由 PTIO1 和 PTIO0 位得到的输出电平必须与通过 PTOC 位设置的初始值不同，否则当比较匹配发生时，PTM 输出脚将不会发生变化。在 PTM 输出脚改变状态后，通过 PTON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，PTIO1 和 PTIO0 用于决定比较匹配条件发生时怎样改变 PTM 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 PTM 关闭时改变 PTIO1 和 PTIO0 位的值是很有必要的。若在 PTM 运行时改变 PTIO1 和 PTIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **PTOC**：PTM PTP 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 PTM 输出脚输出控制位。它取决于 PTM 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 PTM 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 PTM 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 PTON 位由低变为高时 PTM 输出脚的逻辑电平。

Bit 2 **PTPOL**：PTM PTP 输出极性控制位
- 0：同相
- 1：反相

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

Bit 1 **PTCAPTS**：选择 PTM 捕捉触发源
- 0：来自 PTPI 引脚
- 1：来自 PTCK 引脚

Bit 0 **PTCCLR**：选择 PTM 计数器清零条件位
0：PTM 比较器 P 匹配
1：PTM 比较器 A 匹配
此位用于选择清除计数器的方法。周期型 PTM 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。PTCCLR 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。PTCCLR 位在 PWM 输出模式、单脉冲输出模式或捕捉输入模式时未使用。

- **PTMDL 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**：PTM 计数器低字节寄存器 bit 7 ~ bit 0
PTM 10-bit 计数器 bit 7 ~ bit 0

- **PTMDH 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**：PTM 计数器高字节寄存器 bit 1 ~ bit 0
PTM 10-bit 计数器 bit 9 ~ bit 8

- **PTMAL 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**：PTM CCRA 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRA bit 7 ~ bit 0

- **PTMAH 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**：PTM CCRA 高字节寄存器 bit 1 ~ bit 0
PTM 10-bit CCRA bit 9 ~ bit 8

- **PTMRPL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTRP7	PTRP6	PTRP5	PTRP4	PTRP3	PTRP2	PTRP1	PTRP0
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 **PTRP7~PTRP0**：PTM CCRP 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

- **PTMRPH 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PTRP9	PTRP8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **PTRP9~PTRP8**：PTM CCRP 高字节寄存器 bit 1 ~ bit 0
PTM 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 PTMC1 寄存器的 PTM1 和 PTM0 位选择任意模式。

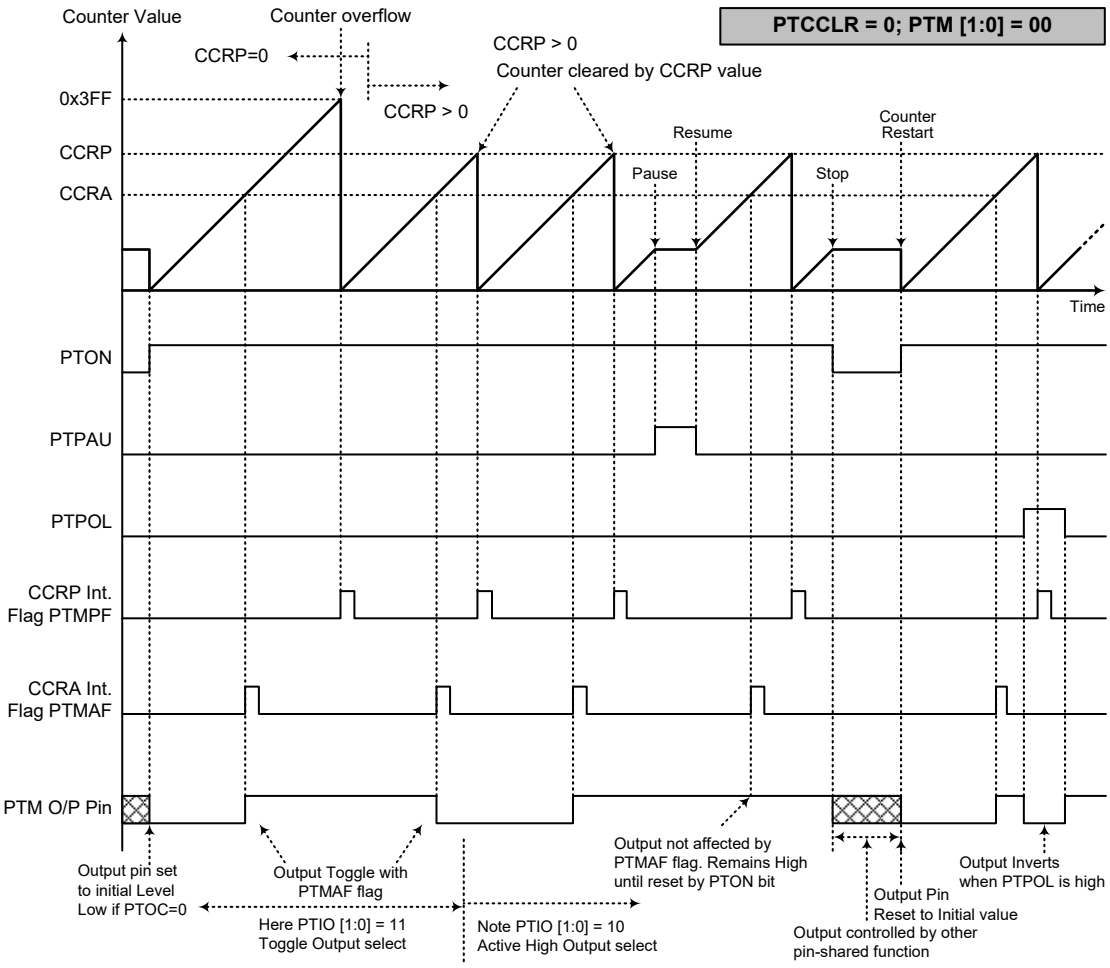
比较匹配输出模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 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 PTCCLR 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 PTMAF 和 PTMPF 将分别置起。

如果 PTMC1 寄存器的 PTCCLR 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 PTMAF 中断请求标志产生。所以当 PTCCLR 为高时，不会产生 PTMPF 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”。

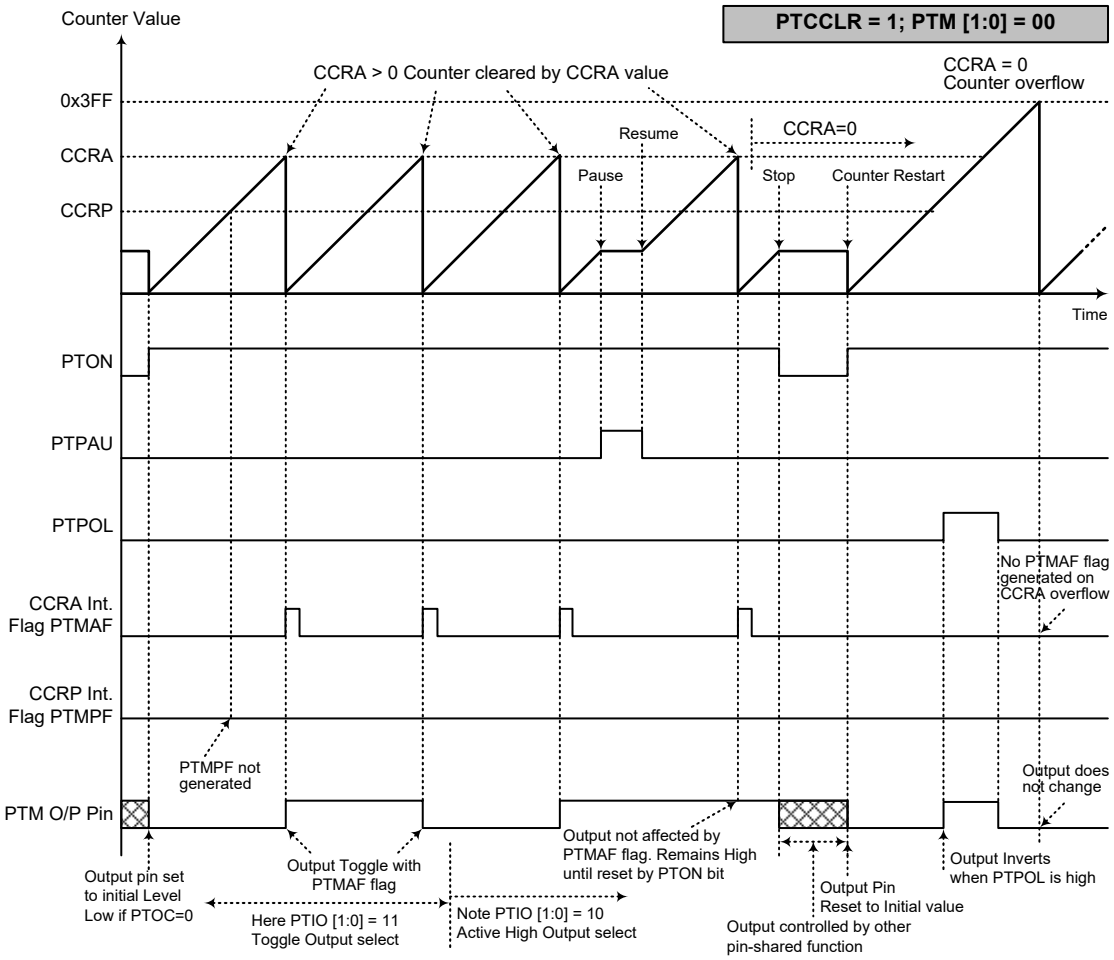
如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 PTMAF 请求标志。

正如该模式名所言，当比较匹配发生后，PTM 输出脚状态改变。当比较器 A 比较匹配发生后 PTMAF 中断请求标志产生时，PTM 输出脚状态改变。比较器 P 比较匹配发生时产生的 PTMPF 标志不影响 PTM 输出脚。PTM 输出脚状态改变方式由 PTMC1 寄存器中 PTIO1 和 PTIO0 位决定。当比较器 A 比较匹配发生时，PTIO1 和 PTIO0 位决定 PTM 输出脚输出高，低或翻转当前状态。在 PTON 位由低到高电平的变化后，PTM 输出脚初始状态为 PTOC 位所指定的电平。注意，若 PTIO1 和 PTIO0 位同时为 0 时，引脚输出不变。



比较器匹配输出模式 – PTCCLR=0

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



比较器匹配输出模式 – **PTCCLR=1**

注：1. PTCCLR=1，比较器 A 匹配将清除计数器
2. PTM 输出脚仅由 PTMAF 标志位控制
3. 在 PTON 上升沿 PTM 输出脚复位至初始值
4. 当 PTCCLR=1 时，不会产生 PTMPF 标志

定时 / 计数器模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 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 PTM 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 PTM 输出脚可用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“10”，且 PTIO1 和 PTIO0 位也需要设置为“10”。PTM 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 PTM 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，其波形的选择就较为灵活。在 PWM 输出模式中，PTCCLR 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。PTMC1 寄存器的 PTOC 位选择 PWM 波形的极性，PTIO1 和 PTIO0 位使能 PWM 输出或强制 PTM 输出脚为高电平或低电平。PTPOL 位用于 PWM 输出波形的极性反相控制。

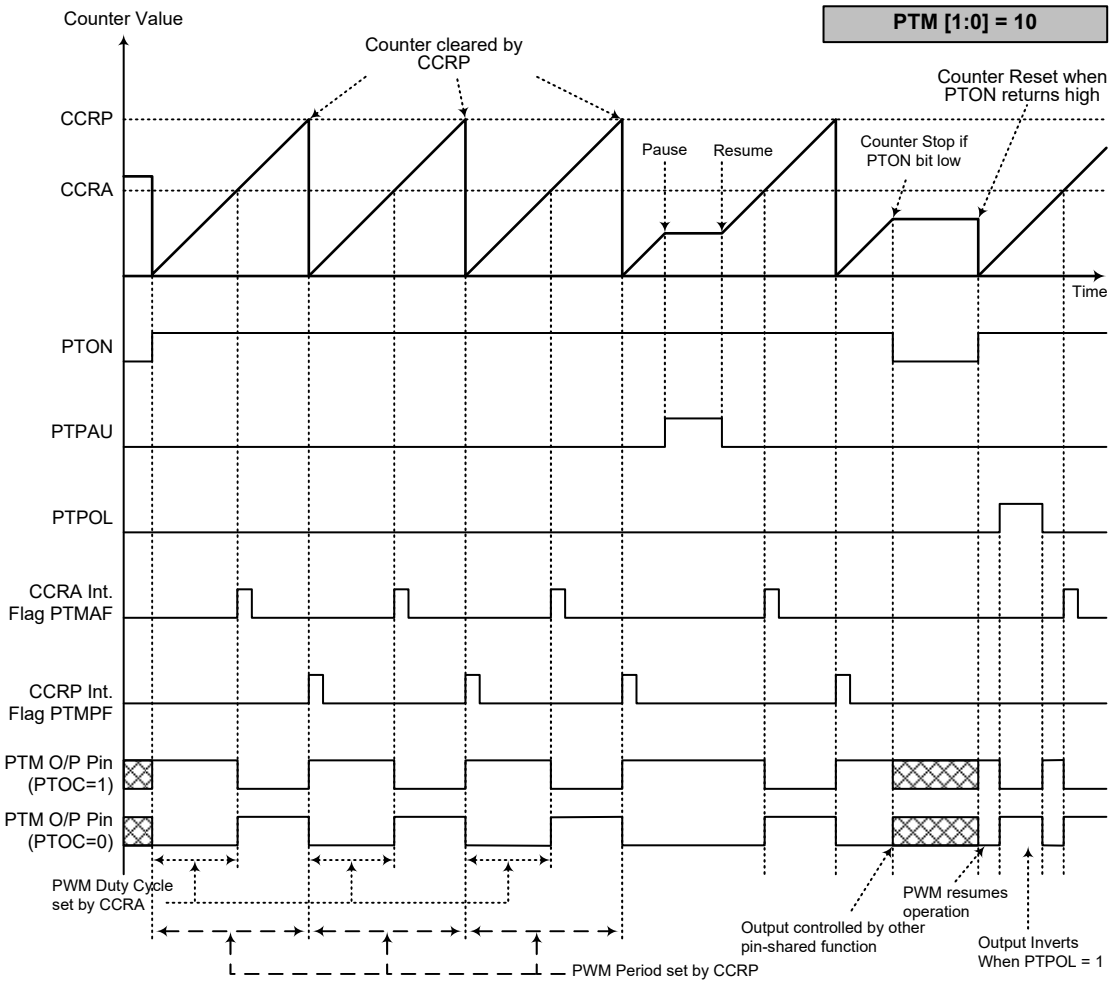
- **10-bit PTM，PWM 输出模式，边沿对齐模式**

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

若 f_{SYS}=16MHz，PTM 时钟源选择 $f_{SYS}/4$，CCRP=512 且 CCRA=128，

PTM PWM 输出频率 = ($f_{SYS}/4$)/512=$f_{SYS}/2048$=7.8125kHz，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

注：1. 计数器由 CCRP 清除

2. 计数器清除设置 PWM 周期

3. 即使在 PTIO[1:0]=00 或 01 时，内部 PWM 功能继续运行

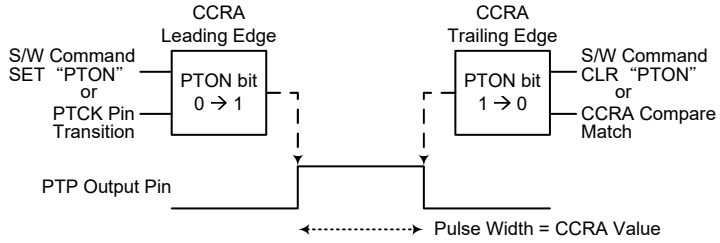
4. PTCCLR 位对 PWM 操作无影响

单脉冲输出模式

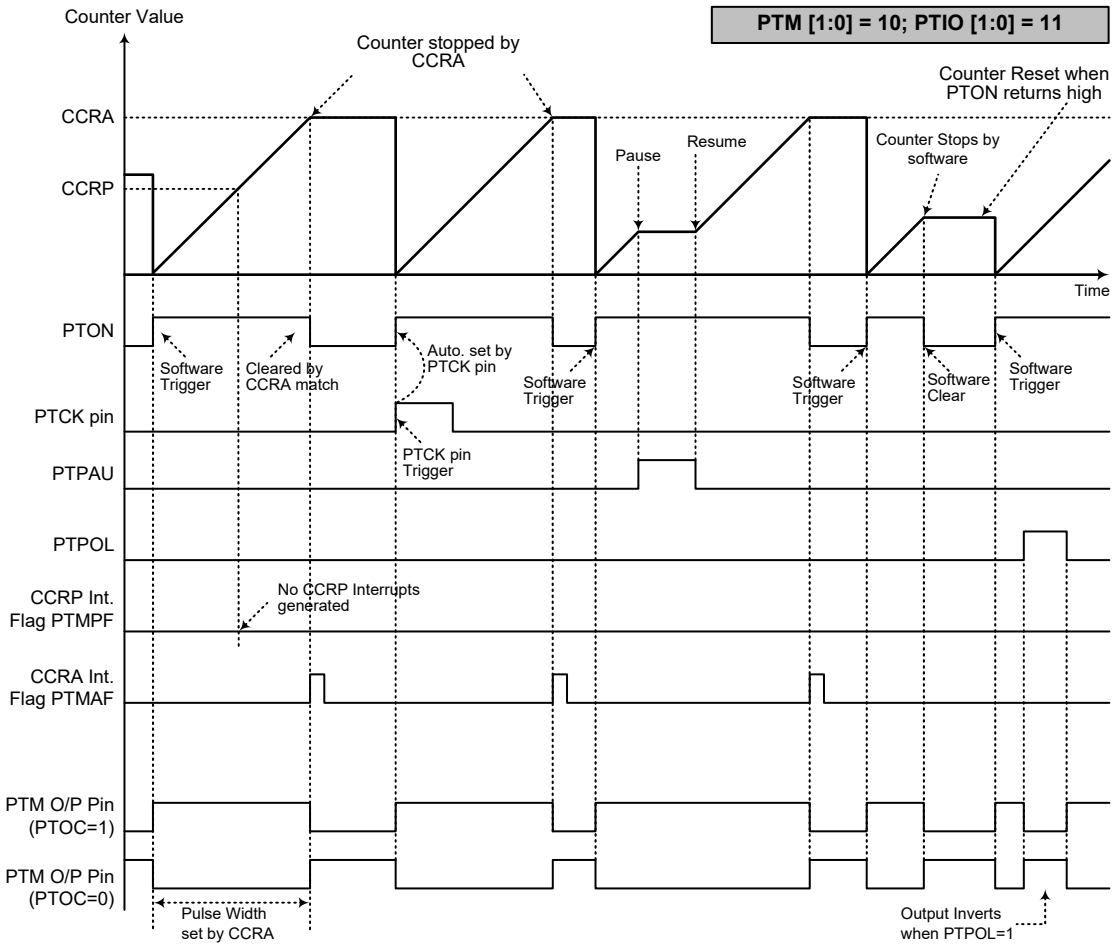
为使 PTM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，并且相应的 PTIO1 和 PTIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

通过应用程序控制 PTON 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，PTON 位可在 PTCK 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 PTON 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 PTON 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 PTON 位并产生单脉冲输出边沿跳转。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 PTM 中断。PTON 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 PTCCLR 位未使用。



单脉冲产生示意图



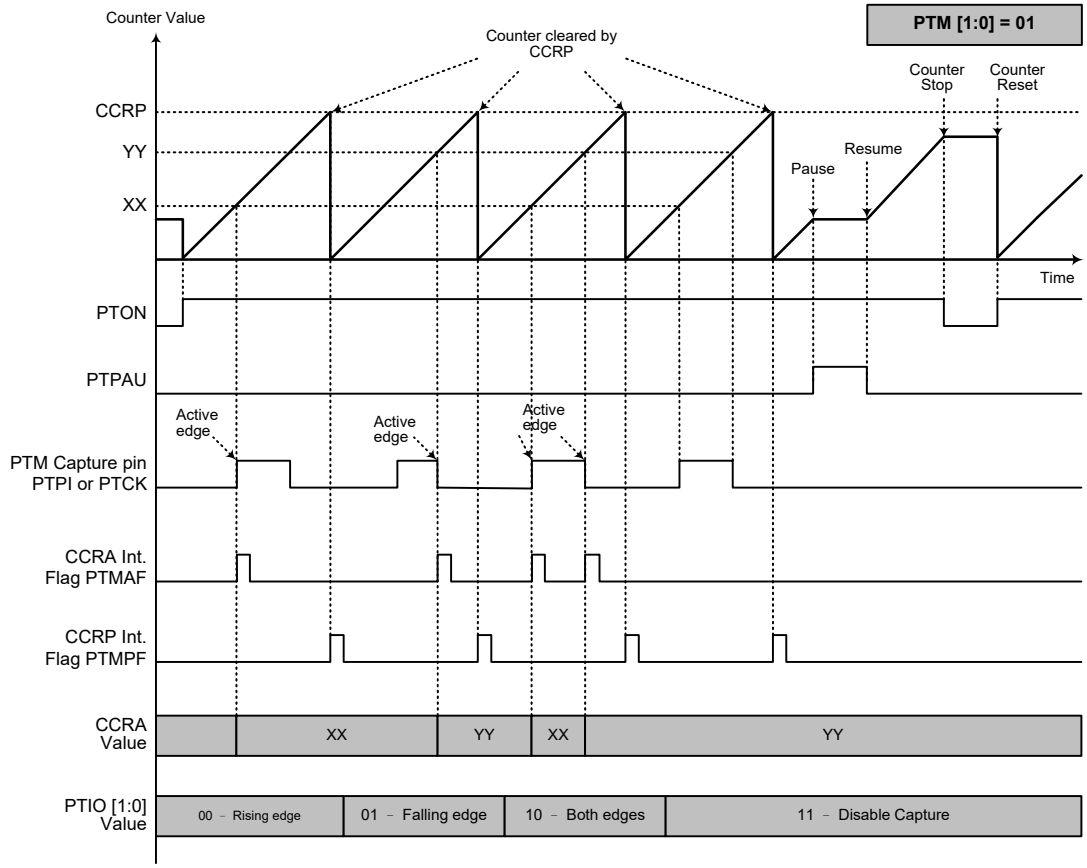
单脉冲输出模式

注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCK 脚或设置 PTON 位为高来触发脉冲
4. PTCK 脚有效沿会自动置位 PTON
5. 单脉冲输出模式中，PTIO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

为使PTM工作在此模式，PTMC1寄存器的PTM1和PTM0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。通过设置 PTMC1 寄存器的 PTCAPTS 位选择 PTPI 或 PTCK 引脚上的外部信号。可通过设置 PTMC1 寄存器的 PTIO1 和 PTIO0 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 PTON 位由低到高转变时，计数器启动。

当 PTPI 或 PTCK 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 PTM 中断。无论 PTPI 或 PTCK 引脚发生哪种边沿转换，计数器将继续工作直到 PTON 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 PTM 中断。记录 CCRP 溢出中断信号的值可以测量长脉宽。通过设置 PTIO1 和 PTIO0 位选择 PTPI 或 PTCK 引脚为上升沿，下降沿或双沿有效。如果 PTIO1 和 PTIO0 位都设置为高，无论 PTPI 或 PTCK 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。PTCCLR，PTOC 和 PTPOL 位在此模式中未使用。



捕捉输入模式

注：1. PTM[1:0]=01 并通过 PTIO[1:0] 位设置有效边沿
2. PTM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. PTCCLR 位未使用
4. 无输出功能 – PTOC 和 PTPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

RF 发射器

RF 发射器是完全集成的发射器，能够使用频移键控 FSK 和开关键控 OOK 两种调制方式进行数据串流。它有两种工作模式，即突发模式和直接模式。RF 发射器工作在 315/433/868/915MHz 频段。

RF 发射器缩写注意事项

CP：充电泵

DFC：数字频率中心

FIFO：TX FIFO

MMD：多模分频器

PAD：功率放大驱动器

SX：合成器

TXD：来自 FIFO 或 DTXD 位的数据，取决于 DIR_EN 位

VCO：压控振荡器

XO：外部晶振输出

XCLK：RF 电路主要时钟。通过配置 XO_SEL[2:0]，XODIV2 和 XCLKD2 位控制，通过将 XCLK_EN 位置高来使能。

XCLK_MCU：单片机系统时钟。通过配置 XO_SEL[2:0], XODIV2 和 XCLKD2 位控制，通过将寄存器 HXTC 的 HXTEN 位置高来使能。

RF 发射器控制寄存器

RF 发射器的所有操作都由一系列寄存器控制。这些寄存器控制 RF 所有功能，包括暂停控制、工作模式选择、时钟分频选择、FIFO 数据配置、调制器控制、小数 N 分频合成器控制、充电泵 (CP) 控制、多模分频器 (MMD) 控制、压控振荡器 (VCO) 控制、TX 功率微调、VCO DFC 校准、RF LDO 控制和 RF 外部晶振输出等。

寄存器名称	位							
	7	6	5	4	3	2	1	0
RF_PWR	—	—	—	—	—	—	—	RF_PDB
RF_OPER	—	—	FSK_EN	DIR_EN	—	—	—	TX_STROBE
RF_CLK1	XCLK_EN	XCLKINV	—	XCLKD2	—	—	—	RST_RF
RF_CLK2	DTR7	DTR6	DTR5	DTR4	DTR3	DTR2	DTR1	DTR0
RF_FIFO_CTRL1	FFDATA7	FFDATA6	FFDATA5	FFDATA4	FFDATA3	FFDATA2	FFDATA1	FFDATA0
RF_FIFO_CTRL2	FFLEN7	FFLEN6	FFLEN5	FFLEN4	FFLEN3	FFLEN2	FFLEN1	FFLEN0
RF_FIFO_CTRL3	RST_TX_FF	FFSA6	FFSA5	FFSA4	FFSA3	FFSA2	FFSA1	FFSA0
RF_FIFO_CTRL4	DTXD	—	—	—	TXFFLT	FFMG_EN	FFMG1	FFMG0
RF_MOD1	FDEV7	FDEV6	FDEV5	FDEV4	FDEV3	FDEV2	FDEV1	FDEV0
RF_MOD2	—	—	—	—	—	FDEV10	FDEV9	FDEV8
RF_OPMOD	—	—	—	—	—	ACAL_EN	TX_EN	SX_EN
RF_SX1	—	DN6	DN5	DN4	DN3	DN2	DN1	DN0

寄存器名称	位							
	7	6	5	4	3	2	1	0
RF_SX2	DK7	DK6	DK5	DK4	DK3	DK2	DK1	DK0
RF_SX3	DK15	DK14	DK13	DK12	DK11	DK10	DK9	DK8
RF_SX4	—	—	—	—	DK19	DK18	DK17	DK16
RF_CP3	DLY_SYN2	DLY_SYN1	DLY_SYN0	D4	D3	D2	D1	D0
RF_OD1	D7	D6	D5	D4	D3	D2	D1	D0
RF_OD3	—	—	DLY_SXPD1	DLY_SXPD0	—	D2	D1	D0
RF_VCO1	VCO_SWHB	D6	D5	D4	D3	D2	D1	D0
RF_VCO2	DFCSF	—	VCO_DFC4	VCO_DFC3	VCO_DFC2	VCO_DFC1	VCO_DFC0	DFC_OW
RF_TX2	CT_PAD3	CT_PAD2	CT_PAD1	CT_PAD0	—	CT_TXLDO1	CT_TXLDO0	D0
RF_DFC_CAL	CT_MMDLDO1	CT_MMDLDO0	—	D4	D3	D2	D1	D0
RF_LDO	D7	D6	D5	D4	D3	D2	D1	D0
RF_XO1	XSHIFT1	XSHIFT0	—	XO_TRIM4	XO_TRIM3	XO_TRIM2	XO_TRIM1	XO_TRIM0
RF_XO2	—	D6	D5	—	XODIV2	XO_SEL2	XO_SEL1	XO_SEL0

RF 发射器控制寄存器列表

大多数 RF 发射器控制寄存器都在此章节描述，但是一些各别寄存器分别在其它章节中描述。

- **RF_PWR 寄存器 – RF 暂停控制寄存器**

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

Bit 7~1　未定义，读为“0”

Bit 0　**RF_PDB**：RF 暂停控制
0：RF 暂停模式
1：RF 有效模式
该位决定 RF 电路处于有效模式还是暂停模式。当单片机发生上电复位、LVR 复位或正常运行情况下发生 WDT 溢出，又或者通过将 PWRC 寄存器的 PWDN 位置 1 且执行 HALT 指令使单片机进入深度休眠模式时，此位都将被清零。

- **RF_CLK1 寄存器 – RF 时钟控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	XCLK_EN	XCLKINV	—	XCLKD2	—	—	—	RST_RF
R/W	R/W	R/W	—	R/W	—	—	—	R/W
POR	1	0	—	0	—	—	—	0

Bit 7 **XCLK_EN**：时钟自动选通用于 RF 内部数字电路
0：除能
1：使能
写数据到 FIFO 需要先使能 XCLK。XCLK_EN 位应该先清零，才能通过将 RF_PDB 位清零使 RF 电路进入暂停模式。这能避免 XCLK 时钟发生毛刺，从而保证不会因为时钟丢失而影响 XCLK 域寄存器。

Bit 6 **XCLKINV**：XCLK 时钟反相控制
0：XCLK 时钟同相
1：XCLK 时钟反相

Bit 5 未定义，读为“0”

Bit 4 **XCLKD2**：XCLK 时钟二分频控制
0：XCLK 时钟未二分频
1：XCLK 时钟二分频
建议将此位清零。

Bit 3~1 未定义，读为“0”

Bit 0 **RST_RF**：RF 复位控制寄存器
0：RF 控制寄存器已完成自动清零
1：RF 控制寄存器复位
当该位置 1 时，除了 RF_PWR 寄存器，其它所有 RF 相关的寄存器都将复位。这需要一个指令周期来完成自动清零使能动作。不要在两个连续的指令中设置 RST_RF 位为 1。

- **RF_CLK2 寄存器 – RF 时钟控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	DTR7	DTR6	DTR5	DTR4	DTR3	DTR2	DTR1	DTR0
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 **DTR7~DTR0**：RF 数据传输速率设置
RF 数据传输速率 = 100kHz/(DTR[7:0]+1)
注：寄存器 RF_XO2 中的 XO_SEL[2:0] 位和 XODIV2 位应先设置以获得正确的参考时钟。

- **RF_CP3 寄存器 – RF CP 控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	DLY_SYN2	DLY_SYN1	DLY_SYN0	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	0	0	1	0	1	0

Bit 7~5 **DLY_SYN2~DLY_SYN0**：合成器延迟就绪时间
000：16μs
001：20μs
010：24μs
011：28μs
100：32μs
101：36μs

110：40μs
111：100μs

Bit 4~0 **D4~D0**：保留位

注：建议该寄存器内容应保持为 POR 值。

- **RF_OD1 寄存器 – RF MMD 和 OD 控制寄存器 1**

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	1	0	0

Bit 7~0 **D7~D0**：多模分频器和输出分频

该寄存器用来控制 RF 电路的多模分频器和输出分频。需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_OD1 设置值	08H	04H	00H

- **RF_OD3 寄存器 – RF MMD 和 OD 控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	—	—	DLY_SXPD1	DLY_SXPD0	—	D2	D1	D0
R/W	—	—	R/W	R/W	—	R/W	R/W	R/W
POR	—	—	0	1	—	1	0	0

Bit 7~6 未定义，读为“0”

Bit 5~4 **DLY_SXPD1~DLY_SXPD0**：用于选择 SX_EN 位由 1 切换到 0 延迟时间
00：25μs
01：50μs
10：75μs
11：100μs

Bit 3 未定义，读为“0”

Bit 2~0 **D2~D0**：保留位

注：建议该寄存器内容应保持为 POR 值。

- **RF_VCO1 寄存器 – RF VCO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	VCO_SWHB	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	1	0	0	0	0

Bit 7 **VCO_SWHB**：VCO 2.5GHz 频带切换控制
0：其它频带
1：315MHz 频带
该位用来选择 VCO 315MHz 频带。如果该位置 1，则选择 VCO 315MHz。当该位清 0 时，则选择除了 315MHz 外的其它 VCO 频带。

Bit 6~0 **D6~D0**：频带控制

需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_VCO1 设置值	90H	10H	10H

- **RF_VCO2 寄存器 – RF VCO 控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	DFCSF	—	VCO_DFC4	VCO_DFC3	VCO_DFC2	VCO_DFC1	VCO_DFC0	DFC_OW
R/W	R	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	—	1	0	0	0	0	0

Bit 7 **DFCSF**：DFC 选择错误标志位 – 只读位
当 VCO 完成校准，该位的状态会改变。如果校准结果超出 DFC 曲线设计范围，该位将置高，表明在当前所选择的通道里无 DFC。若 DFC 选择错误，意味着 VCO 不能工作在所需频率或可能存在其它异常。

Bit 6 未定义，读为“0”

Bit 5~1 **VCO_DFC4~VCO_DFC0**：VCO DFC 曲线 trim 值
00000：最小频率
↓ ： ↓
11111：最大频率
这些位用来设置不同频段对应的各种电容阵列值，以增加 VCO 频率覆盖范围。设置的字段值越大，获得的 VCO 频率越高。这个位字段中的 VCO DFC trim 值可通过 DFC_OW 位选择手动软件设置或硬件自动校准。

Bit 0 **DFC_OW**：VCO DFC 曲线 trim 值来源选择
0：VCO DFC 曲线 trim 值取决于自动校准结果
1：VCO DFC 曲线 trim 值取决于手动设置
该位用来选择 VCO DFC 曲线 trim 值来源。当该位设为 0 时，无论 VCO_DFC 字段为何值，VCO DFC 曲线 trim 值来自自动校准结果。但是，当该位置 1 时，VCO DFC 曲线 trim 值取决于 VCO_DFC 字段而不是自动校准结果。当该位设为 0 时，若读取 VCO_DFC 字段，所返回的 VCO DFC 曲线 trim 值来自自动校准结果。

注：建议该寄存器内容应保持为 POR 值。

下面将介绍两个 DFC 校准范例。

- RF 每次上电后，进行 DFC 校准
  步骤 1：系统上电复位，默认将 DFC_OW 位和 RF_PDB 位都设为 0
  步骤 2：通过将 ACAL_EN 位置 1 来使能 DFC 自动校准，等待，直至 ACAL_EN 位清零
  步骤 3：由于 RF_PDB 位为 0，则 RF 进入暂停模式
  步骤 4：将 RF_PDB 位置 1 启动 RF 电路，并保持 DFC_OW 位为 0
  步骤 5：通过将 ACAL_EN 位置 1 来使能 DFC 自动校准，等待，直至 ACAL_EN 位清零
- 仅当系统上电复位后，进行 DFC 校准
  步骤 1：系统上电复位，默认将 DFC_OW 位和 RF_PDB 位都设为 0
  步骤 2：通过将 RF_PDB 置 1 启动 RF 电路
  步骤 3：通过将 ACAL_EN 位置 1 来使能 DFC 自动校准，等待，直至 ACAL_EN 位清零
  步骤 4：将 VCO_DFC[4:0] 备份至 RAM

步骤 5：通过将 RF_PDB 位设为 0，RF 进入暂停模式
步骤 6：将 RF_PDB 位置 1 启动 RF 电路，并保持 DFC_OW 位为 0
步骤 7：从 RAM 中还原 VCO_DFC[4:0]
步骤 8：将 DFC_OW 位置 1

- **RF_TX2 寄存器 – RF TX 控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	CT_PAD3	CT_PAD2	CT_PAD1	CT_PAD0	—	CT_TXLDO1	CT_TXLDO0	D0
R/W	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	1	1	0	1	—	0	0	0

Bit 7~4 **CT_PAD3~CT_PAD0**：TX PAD 线性功率控制

CT_PAD[3:0]	输出功率微调
1100	0dBm_L3
1000	0dBm_L2
0100	0dBm_L1
0000	0dBm_L0
1101	10dBm_L3
1001	10dBm_L2
0101	10dBm_L1
0001	10dBm_L0
111x	13dBm_L3
101x	13dBm_L2
011x	13dBm_L1
001x	13dBm_L0

注：1. xdBm_L0~xdBm_L3 用于输出功率微调。
2. 功耗大小：xdBm_L0 > xdBm_L1 > xdBm_L2 > xdBm_L3。

Bit 3 未定义，读为“0”

Bit 2~1 **CT_TXLDO1~CT_TXLDO0**：XO LDO 电压设置
00：1.35V
01：1.5V
10：1.65V
11：1.8V

Bit 0 **D0**：保留位

RF Band	CT_TXLDO[1:0]		D0
315MHz	0	1	0
433MHz	0	1	0
868MHz	1	0	0
915MHz	1	0	0

- **RF_DFC_CAL 寄存器 – RF VCO DFC 校准控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CT_MMDLDO1	CT_MMDLDO0	—	D4	D3	D2	D1	D0
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 7~6 **CT_MMDLDO1~CT_MMDLDO0**：MMD LDO 电压设置
00：1.35V
01：1.5V
10：1.65V
11：1.8V

Bit 5 未定义，读为“0”

Bit 4~0 **D4~D0**：保留位

注：建议该寄存器内容应保持为 POR 值。

- **RF_LDO 寄存器 – RF LDO 控制寄存器**

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	1	1	0	0	0

Bit 7~0 **D7~D0**：RF 合成器和 VCO LDO 功能控制

该寄存器用来控制 RF 合成器和 VCO 电路的 LDO 整体功能。需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_LDO 设置值	68H	68H	64H

- **RF_XO1 寄存器 – RF XO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	XSHIFT1	XSHIFT0	—	XO_TRIM4	XO_TRIM3	XO_TRIM2	XO_TRIM1	XO_TRIM0
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	1	0	—	1	0	1	0	1

Bit 7~6 **XSHIFT[1:0]**：晶体振荡器内部电容负载粗调整
每阶电容变化 = 4.5pF，XSHIFT 字段值 = 0 ~ 3
总电容负载 ≈ 7 + XSHIFT[1:0] × 4.5 + XO_TRIM[4:0] × 0.2，单位：pF

Bit 5 未定义，读为“0”

Bit 4~0 **XO_TRIM[5:0]**：晶体振荡器内部电容负载微调整
每阶电容变化 = 0.2pF，XO_TRIM 字段值 = 0 ~ 31
总电容负载 ≈ 7 + XSHIFT[1:0] × 4.5 + XO_TRIM[4:0] × 0.2，单位：pF

- **RF_XO2 寄存器 – RF XO 控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	—	D6	D5	—	XODIV2	XO_SEL2	XO_SEL1	XO_SEL0
R/W	—	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	—	0	1	—	0	0	1	1

Bit 7　未定义，读为“0”

Bit 6~5　**D6~D5**：建议的设定值概括为下表所示

Bit 4　未定义，读为“0”

Bit 3　**XODIV2**：XO 输出二分频选择
0：无任何分频
1：二分频
建议将此位清零。

Bit 2~0　**XO_SEL2~XO_SEL0**：XO 输出选择
000：保留
001：保留
010：保留
011：16MHz
100：保留
101/110/111：保留
应通过 XODIV2 位和 XO_SEL 字段正确选择外部晶振。需注意的是，针对不同的外部晶振选择和 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带 / 外部晶振选择	315MHz	433MHz	868/915MHz
16MHz	0010 0011	0010 x011	0100 x011

“x”：0 或 1

调制模式和工作模式选择

调制模式

该系列单片机有两种 RF 调制模式，即 FSK 模式和 OOK 模式，可通过 FSK_EN 位选择。

在 OOK 调制模式中，RFOUT 引脚将输出 RF 载波信号，其频率 f_C 可通过通道代码 DN[6:0] 和 DK[19:0] 选择。RF 载波信号的开启和关闭由数据速率及传输的数据来控制。

在 FSK 调制模式中，RFOUT 引脚将输出 RF 信号，数据“1”的频率为 (f_C+f_{DEV})，而数据“0”的频率为 (f_C-f_{DEV})。RF 载波信号的开启和关闭由数据速率及传输的数据来控制。

工作模式

该系列单片机有两种 RF 工作模式，即突发模式和直接模式，可通过 DIR_EN 位选择。

在突发模式中，被传送的数据来自 FIFO，且通过状态机控制 RF 模块。用户只需将数据写入 FIFO，且通过将 TX_STROBE 置高来启动传送，直至传送完成。这是一个易用模式。

在直接模式中，被传送的数据由寄存器 RF_FIFO_CTRL4 中的 DTXD 位配置。RF 功能模块中时序的启动和关闭可通过特定的程序序列控制，下文将一一介绍。该模式拥有较大的灵活性，因此要求用户注意模块控制和被传送的数据。

- **RF_OPER 寄存器 – RF 工作控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	FSK_EN	DIR_EN	—	—	—	TX_STROBE
R/W	—	—	R/W	R/W	—	—	—	R/W
POR	—	—	0	0	—	—	—	0

Bit 7~6　未定义，读为“0”

Bit 5　**FSK_EN**：RF 输出调制模式选择
0：OOK 模式，PAD 使能通过 TXD 直接控制。
1：FSK 模式，PAD 强制使能，且 TXD 调制 TX 频率。

Bit 4　**DIR_EN**：RF 输出工作模式选择
0：突发模式 (Burst Mode)，可从 FIFO 读取 TX 数据，通过状态机控制 RF。若 FIFO 传送完成，RF TX 将进入待机模式。
1：直接模式 (Direct Mode)，TX 数据来自寄存器 RF_FIFO_CTRL4 的 DTXD 位，可通过 SX_EN 位 (第一层使能) 和 TX_EN 位 (第二层使能) 控制 RF 状态。

Bit 3~1　未定义，读为“0”

Bit 0　**TX_STROBE**：TX Strobe 启动 RF 突发模式数据传输
0：TX 数据包已经发送完毕
1：启动 TX 传送
若该位置高，则自动启动 TX 传送步骤，此时所有寄存器设置都不能被改变。当一个 TX 数据包发送完毕时，该位将自动清零，此时会产生 BMTCF 中断请求。若该位恢复为低，用户可启动下一个传送。通过对这个位写入一个 0，可迫使用户将 TX 传送结束。

- **RF_MOD1 寄存器 – RF 调制器控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	FDEV7	FDEV6	FDEV5	FDEV4	FDEV3	FDEV2	FDEV1	FDEV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	1	0	0	0	0

Bit 7~0　**FDEV7~FDEV0**：设置 FSK 模式下的频率偏差
这些位将和寄存器 RF_MOD2 的 FDEV10~FDEV9 位一起用来设置 FSK 模式下的频率偏差。

- **RF_MOD2 寄存器 – RF 调制器控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	FDEV10	FDEV9	FDEV8
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	1

Bit 7~3　未定义，读为“0”

Bit 2~0　**FDEV10~FDEV8**：设置 FSK 模式下的频率偏差
这些位将和寄存器 RF_MOD1 的 FDEV7~FDEV0 位一起用来设置 FSK 模式下的频率偏差。
FDEV[10:0]=DEC2HEX (INT($(2^{17}-1)/f_{XTAL}$)×f_D)
例如，若 f_{XTAL}=16MHz，XODIV2=0，f_D=50kHz，则 FDEV[10:0]=19AH。

- **RF_OPMOD 寄存器 – RF 工作模式控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	ACAL_EN	TX_EN	SX_EN
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3　未定义，读为“0”

Bit 2　**ACAL_EN**：自动校准使能控制
0：除能
1：使能
通过应用程序置位或复位此位来激活或终止自动校准功能。当完成自动校准时，该位会被硬件自动清零。

Bit 1　**TX_EN**：发射器控制 (仅在直接模式下有效)
0：除能
1：使能
该位仅用于直接模式，不可用于突发模式。

Bit 0　**SX_EN**：合成器 PLL 开启控制 (仅在直接模式下有效)
0：除能
1：使能
该位仅用于直接模式，不可用于突发模式。

突发模式下的 TX FIFO 模式

在突发模式中，要传送的数据来自 FIFO，由单片机预先写入。有三种 FIFO 模式，分别为简易型 FIFO 模式、模块型 FIFO 模式和扩展型 FIFO 模式，可用于支援不同应用。在突发模式下使用 FIFO，可通过设置寄存器 RF_FIFO_CTRL3 的 RST_TX_FF 位为 1 来复位 FIFO 指针和缓冲器。此后，FIFO 处于像上电复位时的初始状态。

简易型 FIFO 模式

此 FIFO 模式用于普通应用。数据长度不应超过 128 字节。为了采用简易型 FIFO 模式，单片机必须通过访问寄存器 RF_FIFO_CTRL1 的 FFDATA[7:0] 位将要传送的数据写入 FIFO。传送到发射器的传输序列须先入先出，且每个字节都是最高位先出。用户应先确定传输数据包格式，例如前导码和识别码，数据包编码分别有 FEC (前向纠错)，CRC (循环冗余校验) 和数据白化等。当 FIFO 被完全写满后，将寄存器 RF_FIFO_CTRL3 的 FFSA[6:0] 位设为 0，并通过设置寄存器 RF_FIFO_CTRL2 的 FFLEN[7:0] 位配置所需的传输长度 (单位：字节)。然后通过配置 FSK_EN、DIR_EN 和 TX_STROBE 位来启动传送。完成当前的传送后，数据将被保存在 FIFO 以等待下一次传送。

模块型 FIFO 模式

模块型 FIFO 模式用于支援多键编码应用。用户应预先将所有按键编码写入 FIFO。当按下按键时，单片机将检测到按键，并通过设置寄存器 RF_FIFO_CTRL3 的 FFSA[6:0] 位为目标按键码起始地址，然后通过设置寄存器 RF_FIFO_CTRL2 的 FFLEN[7:0] 位来表明按键码长度，最后通过设置寄存器 RF_OPER 的 TX_STROBE 位来启动传送。最大 FIFO 长度也应限制为 128 字节。

扩展型 FIFO 模式

扩展型 FIFO 模式用于长度多达 256 字节的大数据包。FIFO 的物理长度为 128 字节。为了在一个数据包里扩展可用的传输长度，则在单片机和 FIFO 之间需要一个握手机制。

通过设置寄存器 RF_FIFO_CTRL4 的 FFMG[1:0] 位来决定 FIFO 数据长度阈值，通过将 FFMG_EN 位置 1 使能阈值检测功能以提醒单片机。当收到提示信号时，单片机应尽可能快地将数据写入 FIFO，以避免 FIFO 数据长度为 0 从而导致传输终止。

- **RF_FIFO_CTRL1 寄存器 – RF FIFO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	FFDATA7	FFDATA6	FFDATA5	FFDATA4	FFDATA3	FFDATA2	FFDATA1	FFDATA0
R/W	W	W	W	W	W	W	W	W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **FFDATA7~FFDATA0**：FIFO 数据端口
写数据到该端口，即将数据写入 FIFO。从该端口读取数据，即从 FIFO 顶部读取数据。

- **RF_FIFO_CTRL2 寄存器 – RF FIFO 控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	FFLEN7	FFLEN6	FFLEN5	FFLEN4	FFLEN3	FFLEN2	FFLEN1	FFLEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	1	1	1	1

Bit 7~0 **FFLEN7~FFLEN0**：TX 数据长度 (仅用于突发模式)
被传送的数据字节数为 FFLEN[7:0]+1。在简易型 FIFO 模式和模块型 FIFO 模式中，FFLEN[7:0] 应限制在 7Fh。用户设置寄存器时不可大于该值。

- **RF_FIFO_CTRL3 寄存器 – RF FIFO 控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	RST_TX_FF	FFSA6	FFSA5	FFSA4	FFSA3	FFSA2	FFSA1	FFSA0
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 **RST_TX_FF**：复位 TX FIFO
0：TX FIFO 未复位到初始状态或复位动作完成
1：TX FIFO 复位到初始状态
该位置 1 时，TX FIFO 复位到初始状态。复位动作完成后，该位将自动清零。

Bit 6~0 **FFSA6~FFSA0**：FIFO 发送数据的起始地址 — 仅用于模块型 FIFO 模式

- **RF_FIFO_CTRL4 寄存器 – RF FIFO 控制寄存器 4**

Bit	7	6	5	4	3	2	1	0
Name	DTXD	—	—	—	TXFFLT	FFMG_EN	FFMG1	FFMG0
R/W	R/W	—	—	—	R	R/W	R/W	R/W
POR	0	—	—	—	0	0	0	1

Bit 7 **DTXD**：直接模式 TX 数据设置

Bit 6~4 未定义，读为“0”

Bit 3 **TXFFLT**：TX FIFO 长度小于 FFMG[1:0] 指定的长度阈值 (只读)
0：TX FIFO 长度大于 FFMG[1:0] 指定的长度阈值
1：TX FIFO 长度低于或等于 FFMG[1:0] 指定的长度阈值
若 FFMG_EN 位为 0，TXFFLT 位将始终保持为 0。若 FFMG_EN 位为 1，TXFFLT 位将表明 TX FIFO 的长度状态。当该位被硬件置高时，表明 TX FIFO 长度低于或等于 FFMG[1:0] 指定的长度阈值，此时，将产生一个 FFMGF 中断请求。

Bit 2 **FFMG_EN**：TX FIFO 长度阈值检测功能使能控制
0：除能
1：使能

Bit 1~0 **FFMG1~FFMG0**：TX FIFO 长度阈值 – 表示 FIFO 中剩余的数据字节数
00：4 个字节
01：8 个字节
10：16 个字节
11：32 个字节

RF 通道设置

RF 通道可通过四个寄存器 RF_SX1 ~ RF_SX4 设置。

- **RF_SX1 寄存器 – RF 小数 N 分频合成器控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	—	DN6	DN5	DN4	DN3	DN2	DN1	DN0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	1	1	0	1	1	0

Bit 7 未定义，读为“0”

Bit 6~0 **DN6~DN0**：被除数的整数部分用于 MMD，支持 32~127
由于 MMD 数据来自 delta-sigma 调制器，DN 支持范围将受影响。实际范围将为 (32+3) ~ (127-4)，即 35~123。
设置初始值，即 XO=16MHz，TX 频带 = 433.92MHz：
例如，若晶振 = XO=16MHz，XODIV2=0，且 RF_OD1=04H；
VCO 频率 = TX 频率 × 4=433.92MHz×4=1735.68MHz；
$XO\times(DN+DK/2^{20})=16MHz\times(DN+DK/2^{20})$=VCO 频率 / 2=1735.68MHz/2=867.84MHz；
$(DN+DK/2^{20})$=54.24，因此 DN[6:0]=36H=54，DK[19:0]=3D70AH=251658。

- **RF_SX2 寄存器 – RF 小数 N 分频合成器控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	DK7	DK6	DK5	DK4	DK3	DK2	DK1	DK0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	0	1	0

Bit 7~0 **DK7~DK0**：被除数的 20-bit 小数部分的低字节用于 MMD
设置初始值，即 XO=16MHz，且 TX 频带 = 433.92MHz。

- **RF_SX3 寄存器 – RF 小数 N 分频合成器控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	DK15	DK14	DK13	DK12	DK11	DK10	DK9	DK8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	0	1	0	1	1	1

Bit 7~0 **DK15~DK8**：被除数的 20-bit 小数部分的中间字节用于 MMD
设置初始值，即 XO=16MHz，且 TX 频带 = 433.92MHz。

- **RF_SX4 寄存器 – RF 小数 N 分频合成器控制寄存器 4**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	DK19	DK18	DK17	DK16
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	1	1

Bit 7~4　未定义，读为“0”

Bit 3~0　**DK19~DK16**：被除数的 20-bit 小数部分的高字节用于 MMD 设置初始值，即 XO=16MHz，且 TX 频带 = 433.92MHz。

RF 通道设置说明 (VCO 工作频率：1.7GHz~1.9GHz，2.5GHz)

XODIV2 位和寄存器 RF_OD1 必须按照下面两个规则配置：

1. VCO 工作频率是否在规定范围内。
2. DN 和 DK 计算值在寄存器 RF_SX1~RF_SX4 的可用范围内。

当晶振频率 = XO=16MHz 时，以下为五个频率配置范例：

- 315MHz

  XODIV2=0，RF_OD1=08H
  VCO 频率 = TX 频率 × 8=315MHz×8=2520MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=2520MHz/2=1260MHz
  (DN+DK/2^{20})=78.75，因此 DN[6:0]=4EH=78，DK[19:0]=C0000H=786432
  注：当 TX 频率为 315MHz 时，应通过应用程序将 VCO_SWHB 位置高，将 VCO 转换到 2.5GHz 频带。

- 433MHz

  XODIV2=0，RF_OD1=04H
  VCO 频率 = TX 频率 × 4=433MHz×4=1732MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1732MHz/2=866MHz
  (DN+DK/2^{20})=54.125，因此 DN[6:0]=36H=54，DK[19:0]=20000H=131072

- 433.92MHz (默认设置)

  XODIV2=0，RF_OD1=04H
  VCO 频率 = TX 频率 × 4=433.92MHz×4=1735.68MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1735.68MHz/2=867.84MHz
  (DN+DK/2^{20})=54.24，因此 DN[6:0]=36H=54，DK[19:0]=3D70AH=251658

- 868MHz

  XODIV2=0，RF_OD1=00H
  VCO 频率 = TX 频率 × 2=868MHz×2=1736MHz
  XO×(DN+DK/(2^{20}))=16MHz×(DN+DK/(2^{20}))=VCO 频率 / 2=1736MHz/2=868MHz
  (DN+DK/(2^{20}))=54.25，因此 DN[6:0]=36H=54，DK[19:0]=40000H=262144

- 915MHz

  XODIV2=0，RF_OD1=00H
  VCO 频率 = TX 频率 × 2=915MHz×2=1830MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1830MHz/2=915MHz
  (DN+DK/2^{20})=57.1875，因此 DN[6:0]=39H=57，DK[19:0]=30000H=196608

软件编程指南

为了帮助用户实现所需的传送，此章节将介绍突发模式和直接模式下的参考设置步骤。这两个模式的差异在于被传送数据的产生、RF 模块开启和关闭控制，更多细节详见下文描述。

突发模式 (Burst Mode)

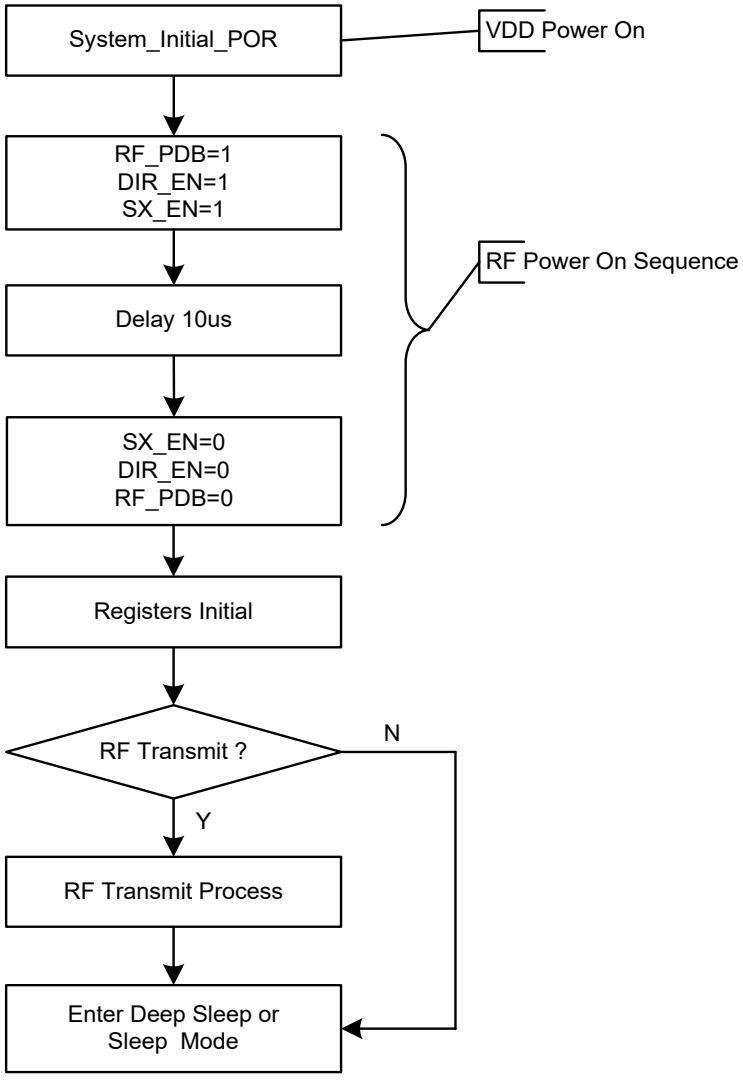
1. 根据所选的晶体振荡器正确设置 XO_SEL[2:0]、XODIV2 和 XCLKD2 位。
2. 由于 XCLK_EN 位默认为 1，通过将 RF_PDB 位置 1，RF 电路将采用 XCLK 时钟开始工作。
3. 若用户选择 HXT 作为系统时钟，请检查 HXTF 位。若此位为 1，表明单片机时钟已经准备完毕，单片机可采用 XCLK_MCU 时钟作为它的主要时钟。
4. 设置 RF 相关配置寄存器。
5. 若第一次上电或工作环境变化很大，通过设置寄存器 RF_OPMOD 的 ACAL_EN 位来启动自动校准进程，轮询该位直至校准结束。读取寄存器 RF_VCO2 的 VCO_DFC[4:0] 位的值，并将其备份到 RAM。当单片机或 RF 电路进入暂停模式时，该动作可以节省校准时间。通过对该字段写入访问，该数据将被重新存储到 VCO_DFC[4:0]。然后通过设置寄存器 RF_VCO2 的 DFC_OW 位为 1，迫使 DFC 选择该曲线。
6. 对 FIFO 写入数据。通过 FFLEN[7:0] 位来设置所需的传输字节长度。通过 DTR[7:0] 位来设置数据传输速率。通过 FDEV[10:0] 位设置 FSK 模式下的频率偏差。
7. 通过设置 DIR_EN 位为 0 来选择突发模式，设置 FSK_EN 位为 0，则选择 OOK 调制方式，而设置 FSK_EN 位为 1，则选择 FSK 调制方式，设置 TX_STROBE 位为 1，启动传输。然后 FIFO 的数据将通过 FSK 或 OOK 调制方式自动转移到 RFOUT 引脚。
8. 轮询 TX_STROBE 位直至该位被硬件清零。当 TX_STROBE 位为 1 时，不允许对 RF 配置寄存器进行任何写入访问。
9. 若用户想要再发送先前的数据，只需检查 TX_STROBE 位。当该位为低，再设置为 1 将启动下一次传输。
10. 若 TX_STROBE 位为高，用户想强制终止此次传输，只需将 TX_STROBE 位设为低，而等待至少 1μs 后，状态机将关闭功率放大驱动器。用户可再次设置 X_STROBE 位来启动新的传输。
11. 通过将 XCLK_EN 设为 0，然后将 RF_PDB 清零即可关闭 RF 电路。

直接模式 (Direct Mode)

1. 根据所选的晶体振荡器正确设置 XO_SEL[2:0]、XODIV2 和 XCLKD2 位。
2. 由于 XCLK_EN 位默认为 1，通过将 RF_PDB 位置 1，RF 电路将采用 XCLK 时钟开始工作。
3. 若用户选择 HXT 作为系统时钟，请检查 HXTF 位。若此位为 1，表明单片机时钟已经准备完毕，单片机可采用 XCLK_MCU 时钟作为它的主要时钟。
4. 设置 RF 相关配置寄存器。
5. 若第一次上电或工作环境变化很大，通过设置寄存器 RF_OPMOD 的 ACAL_EN 位来启动自动校准进程，轮询该位直至校准结束。读取寄存器 RF_VCO2 的 VCO_DFC[4:0] 位的值，并将其备份到 RAM。当单片机或 RF 电路进入暂停模式时，该动作可以节省校准时间。通过对该字段写入访问，该数据将被重新存储到 VCO_DFC[4:0]。然后通过设置寄存器 RF_VCO2 的 DFC_OW 位为 1，迫使 DFC 选择该曲线。

6. 写入要传送的第一个数据位至 DTXD 位。通过 FDEV[10:0] 位来设置 FSK 模式下的频率偏差。
7. 通过将 DIR_EN 位设为 1 来选择直接模式，设置 FSK_EN 位为 0，则选择 OOK 调制方式，而设置 FSK_EN 位为 1，则选择 FSK 调制方式。
8. 通过将 SX_EN 置位来使能所有合成器模块功能，执行一段延迟时间以等待合成器稳定。
9. 通过将 TX_EN 置位来使能功率放大器模块功能，然后开始通过 RFOUT 引脚传输数据。
10. 按照所需数据传输速率持续更新 DTXD 位直至所有数据传送完成。
11. 将 TX_EN 位设为 0，延迟 1μs，然后也将 SX_EN 位为设 0。
12. 将 XCLK_EN 位设为 0，然后将 RF_PDB 位清零来关闭 RF 电路。

注：在直接模式中，当完成数据传送时，为了使 RF 电路进入暂停模式以减少功耗，用户必须用正确的顺序配置相关位，如步骤 11 和步骤 12 所描述的，否则会产生不必要的待机电流。



RF 发射器流程

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 和 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 和时基等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于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~INTC1 寄存器，用于设置基本的中断；第二类是 MFI0~MFI2 寄存器，用于设置多功能中断；最后一种有 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 或 1
时基	TBnE	TBnF	n=0 或 1
多功能	MFInE	MFInF	n=0~2
LVD	LVDE	LVDF	—
RF TX FIFO 长度阈值检测	FFMGE	FFMGF	—
RF 突发模式传送完成	BMTCE	BMTCF	—
CTM	CTMPE	CTMPF	—
	CTMAE	CTMAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MFI2F	MFI1F	MFI0F	TB1F	MFI2E	MFI1E	MFI0E	TB1E
MFI0	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
MFI2	—	LVDF	FFMGF	BMTCF	—	LVDE	FFMGE	BMTCE

中断寄存器列表

- **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	—	TB0F	INT1F	INT0F	TB0E	INT1E	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 **TB0F**：时基 0 中断请求标志位
0：无请求
1：中断请求

Bit 5 **INT1F**：INT1 中断请求标志位
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 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 2 **INT1E**：INT1 中断控制位
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	MFI2F	MFI1F	MFI0F	TB1F	MFI2E	MFI1E	MFI0E	TB1E
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 **MFI2F**：多功能中断 2 请求标志位
0：无请求
1：中断请求

Bit 6 **MFI1F**：多功能中断 1 请求标志位
0：无请求
1：中断请求

Bit 5 **MFI0F**：多功能中断 0 请求标志位
0：无请求
1：中断请求

Bit 4 **TB1F**：时基 1 中断请求标志位
0：无请求
1：中断请求

Bit 3 **MFI2E**：多功能中断 2 控制位
0：除能
1：使能

Bit 2 **MFI1E**：多功能中断 1 控制位
0：除能
1：使能

Bit 1 **MFI0E**：多功能中断 0 控制位
0：除能
1：使能

Bit 0 **TB1E**：时基 1 中断控制位
0：除能
1：使能

- **MFI0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	CTMAF	CTMPF	—	—	CTMAE	CTMPE
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 **CTMAF**：CTM 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **CTMPF**：CTM 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 **CTMAE**：CTM 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **CTMPE**：CTM 比较器 P 匹配中断控制位
0：除能
1：使能

- **MFI1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
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 **PTMAF**：PTM 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PTMPF**：PTM 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 **PTMAE**：PTM 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **PTMPE**：PTM 比较器 P 匹配中断控制位
0：除能
1：使能

- **MFI2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	LVDF	FFMGF	BMTCF	—	LVDE	FFMGE	BMTCE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **LVDF**：LVD 中断请求标志位
0：无请求
1：中断请求

Bit 5 **FFMGF**：RF TX FIFO 长度阈值检测中断请求标志位
0：无请求
1：中断请求

Bit 4 **BMTCF**：RF 突发模式传送完成中断请求标志位
0：无请求
1：中断请求

Bit 3 未定义，读为“0”

Bit 2 **LVDE**：LVD 中断控制位
0：除能
1：使能

Bit 1 **FFMGE**：RF TX FIFO 长度阈值检测中断控制位
0：除能
1：使能

Bit 0 **BMTCE**：RF 突发模式传送完成中断控制位
0：除能
1：使能

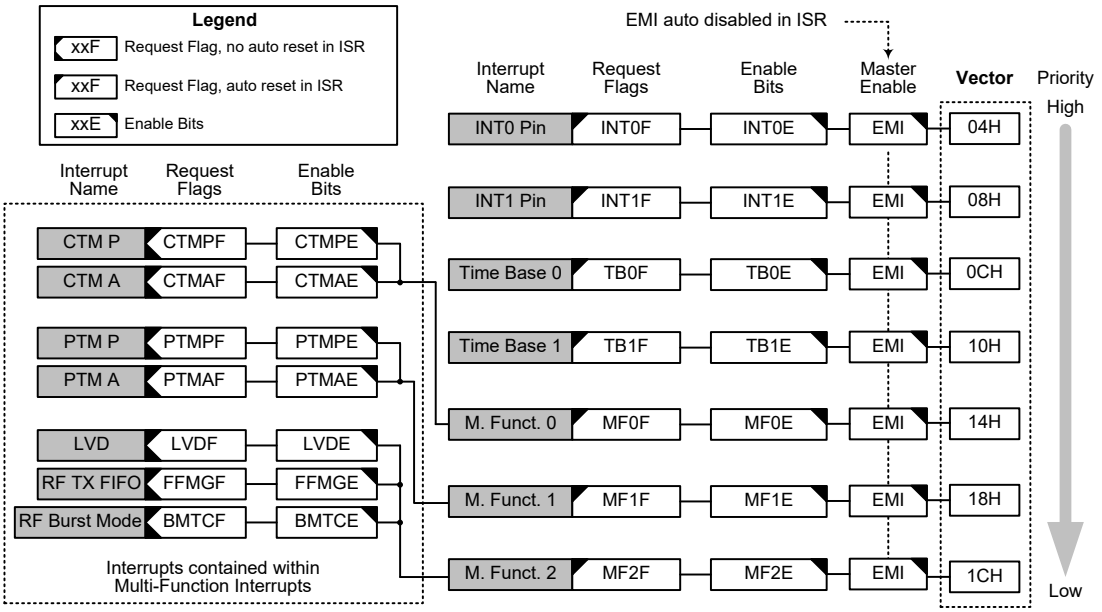
中断操作

若中断事件条件产生，如一个 TM 比较器 P 或比较器 A 匹配等等，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“JMP”指令，以跳转到相应的中断服务程序。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

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

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



中断结构

外部中断

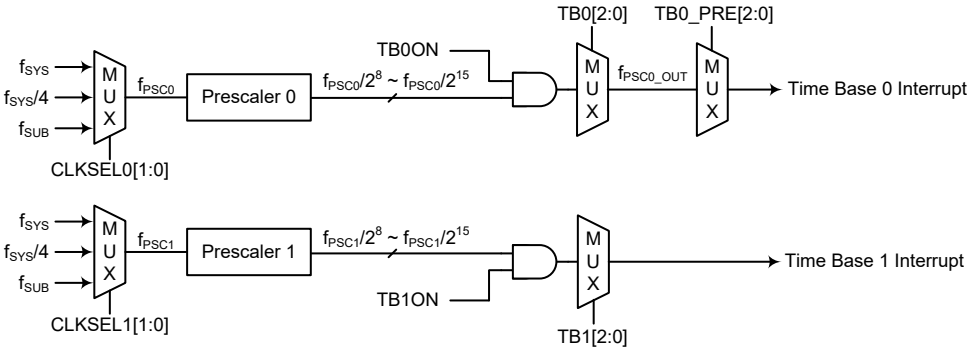
通过 INT0~INT1 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT1 引脚的状态发生变化，外部中断请求标志 INT0F~INT1F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INT0E~INT1E 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，并且通过引脚共用寄存器选择外部中断脚，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断引脚发生了所选择的边沿跳转，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它中断。注意，即使此引脚被用作外部中断输入，其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。

时基中断

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

时基的时钟源 f_{PSC0} 或 f_{PSC1} 来自内部时钟源 f_{SYS}、$f_{SYS}/4$ 或 f_{SUB}。f_{PSC0} 或 f_{PSC1} 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。时基中断周期控制时钟 (f_{PSC0} 或 f_{PSC1}) 的时钟源，可通过 PSCR0 或 PSCR1 寄存器的 CLKSEL1[1:0] 和 CLKSEL0[1:0] 位选择。



时基中断

- **PSCR0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL01	CLKSEL00
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 CLKSEL01~CLKSEL00：分频器 0 时钟源 f_{PSC0} 选择
00：f_{SYS}
01：$f_{SYS}/4$
1x：f_{SUB}
需要注意的是当单片机进入深度休眠模式时，无论分频器 0 时钟源如何选择，时基 0 时钟源来自 f_{LIRC}。

- **PSCR1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL11	CLKSEL10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **CLKSEL11~CLKSEL10**：分频器 1 时钟源 f_{PSC1} 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	TB0_PRE2	TB0_PRE1	TB0_PRE0	—	TB02	TB01	TB00
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 **TB0ON**：时基 0 控制位
0：除能
1：使能

Bit 6~4 **TB0_PRE2~TB0_PRE0**：选择来自 f_{PSC0_OUT} 的时基 0 溢出周期
000：$2^0/f_{PSC0\_OUT}$
001：$2^1/f_{PSC0\_OUT}$
010：$2^2/f_{PSC0\_OUT}$
011：$2^3/f_{PSC0\_OUT}$
100：$2^4/f_{PSC0\_OUT}$
101：$2^5/f_{PSC0\_OUT}$
110：$2^6/f_{PSC0\_OUT}$
111：$2^7/f_{PSC0\_OUT}$
在深度休眠模式下，TB0 溢出周期将会是其预设值的一半。而在非深度休眠模式下，TB0 的第一次溢出周期将会是其预设值的一半。

Bit 3 未定义，读为“0”

Bit 2~0 **TB02~TB00**：选择来自 f_{PSC0} 的时基 0 溢出周期位
000：$2^8/f_{PSC0}$
001：$2^9/f_{PSC0}$
010：$2^{10}/f_{PSC0}$
011：$2^{11}/f_{PSC0}$
100：$2^{12}/f_{PSC0}$
101：$2^{13}/f_{PSC0}$
110：$2^{14}/f_{PSC0}$
111：$2^{15}/f_{PSC0}$

- **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 使能 / 除能控制位
0：除能
1：使能

Bit 6~3 未定义，读为“0”

Bit 2~0 **TB12~TB10**：选择来自 f_{PSC1} 的时基 1 溢出周期位
000：$2^8/f_{PSC1}$
001：$2^9/f_{PSC1}$
010：$2^{10}/f_{PSC1}$
011：$2^{11}/f_{PSC1}$
100：$2^{12}/f_{PSC1}$
101：$2^{13}/f_{PSC1}$
110：$2^{14}/f_{PSC1}$
111：$2^{15}/f_{PSC1}$
在非深度休眠模式下，TB1 的第一次溢出周期将会是其预设值的一半。

多功能中断

此系列单片机中有多达 4 种多功能中断，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、LVD 中断、RF FFMG 长度阈值中断和 RF 突发模式传送完成中断。

当多功能中断中任何一种中断请求标志 MFInF 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

但必须注意的是，在中断响应时，虽然多功能中断标志会自动复位，但多功能中断源的请求标志位，即 TM 中断、LVD 中断、RF FFMG 长度阈值中断和 RF 突发模式传送完成中断的请求标志位不会自动复位，必须由应用程序清零。

RF TX FIFO 长度阈值检测中断

RF TX FIFO 长度阈值检测中断属于多功能中断。当 TX FIFO 长度小于阈值长度时，TX FIFO 长度阈值检测中断请求标志位 FFMGF 被置位，TX FIFO 长度阈值检测中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、TX FIFO 长度阈值检测中断使能位 FFMGE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满且前面提到的条件发生时，可跳转至相关多功能中断向量子程序中执行。当中断响应，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 FFMGF 标志需在应用程序中手动清除。

RF 突发模式传送完成中断

RF 突发模式传送完成中断属于多功能中断。当突发模式下的 TX 数据包已经完全发送出去时，RF 突发模式传送完成中断请求标志位 BMTCF 被置位，RF 突发模式传送完成中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、RF 突发模式传送完成中断使能位 BMTCE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满且前面提到的条件发生时，可跳转至相关多功能中断向量子程序中执行。当中断响应，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 BMTCF 标志需在应用程序中手动清除。

LVD 中断

LVD 中断属于多功能中断。当低电压检测功能检测到一个低电压时，LVD 中断请求标志 LVDF 被置位，LVD 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、低电压中断使能位 LVDE 和相应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，可跳转至相关多功能中断向量子程序中执行。当LVD中断响应，EMI将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 LVDF 标志需在应用程序中手动清除。

TM 中断

简易型和周期型 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。所有类型的 TM 都有两个中断请求标志位及两个使能位。当 TM 比较器 P 或比较器 A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

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

中断唤醒功能

每个中断都具有将处于休眠或空闲模式的单片机唤醒的能力。当中断请求标志由低到高转换时唤醒动作产生，其与中断是否使能无关。因此，尽管单片机处于休眠或空闲模式且系统振荡器停止工作，如有外部中断脚上产生外部边沿跳变、低电压都可能导致其相应的中断标志被置位，由此产生中断。

当系统进入深度休眠模式时，时基 0 中断能够唤醒单片机，在深度休眠模式下时基 0 时钟源来自 f_{LIRC}。

因此必须注意避免伪唤醒情况的发生。若中断唤醒功能被除能，单片机进入深度休眠、休眠或空闲模式前相应中断请求标志应被置起。中断唤醒功能不受中断使能位的影响。

编程注意事项

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

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 MFInF 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，时基 0 中断在深度休眠模式下具有唤醒功能，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，在单片机进入深度休眠、休眠或空闲模式前需先将相应请求标志置为高。

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

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

低电压检测 – LVD

该系列单片机具有低电压检测功能，即 LVD。该功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 8 个固定电压中的一个参考点。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。ENLVD 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	ENLVD	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	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

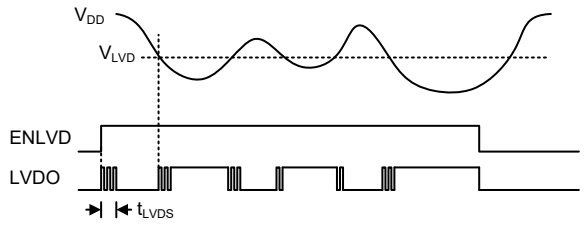
Bit 4　**ENLVD**：低电压检测控制位
0：除能
1：使能

Bit 3　　未定义，读为“0”

Bit 2~0　　**VLVD2~VLVD0**：选择 LVD 电压位
000：1.9V
001：2.0V
010：2.2V
011：2.4V
100：2.8V
101：2.9V
110：3.0V
111：3.3V

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。其设置的范围为 1.9V~3.3V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。若 ENLVD 位为高，当单片机处于空闲模式时低电压检测器保持有效状态。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

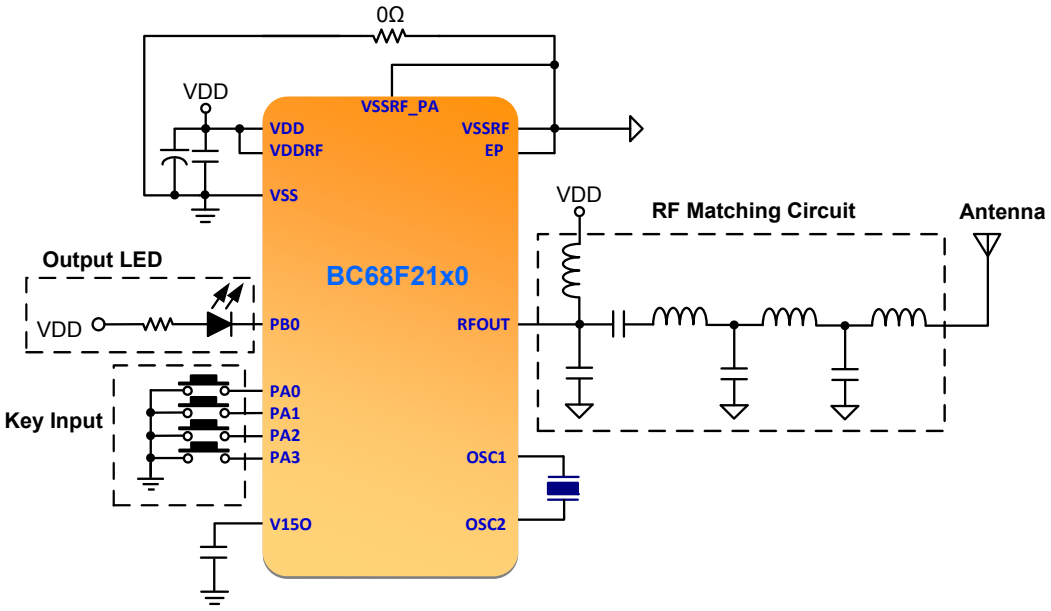
低电压检测器也有自己的中断功能，也是属于多功能中断的一种，它是除了轮询 LVDO 位之外的另一种检测低电压的方法。中断条件产生置位 LVDO 并延时 t_{LVD} 后，中断产生。此种情况下，若 V_{DD} 降至小于 LVD 预置电压值时，中断请求标志位 LVDF 将被置位，中断产生，单片机将从空闲模式中被唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVDF 标志置为高。

配置选项

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

序号	选项
看门狗定时器选项	
1	看门狗定时器选项 1. 始终使能 2. 由寄存器 WDTC 控制
LVR 选项	
2	LVR 功能 1. 除能 2. 使能

应用电路


0Ω
VDD
VSSRF_PA
VDD
VDDRF
VSS
VSSRF
EP
VDD
RF Matching Circuit
Antenna
Output LED
BC68F21x0
VDD
PB0
RFOUT
Key Input
PA0
PA1
PA2
PA3
OSC1
V15O
OSC2

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 0.5μs 中执行完成，而分支或调用操作则将在 1μs 中执行完成。虽然需要两个指令周期的指令通常指的是 JMP、CALL、RET、RETI 和查表指令，但如果牵涉到程序计数器低字节寄存器 PCL 也将多花费一个周期去加以执行。即指令改变 PCL 的内容进而导致直接跳转至新地址时，需要多一个周期去执行，例如“CLR PCL”或“MOV PCL，A”指令。对于跳转指令必须注意的是，如果比较的结果牵涉到跳转动作将多花费一个周期，如果没有则需一个周期即可。

数据的传送

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

查表运算

数据的储存通常由寄存器完成，然而当处理大量固定的数据时，它的存储量常常造成对个别存储器的不便。为了改善此问题，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，并送至数据存储器和 TBLH	2[注]	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2[注]	无
ITABRD [m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器和 TBLH	2[注]	无
ITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 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]	如果数据存储器为零，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，并送至数据存储器和 TBLH	3 注	无
LTABRDL	[m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	3 注	无
LITABRD	[m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器和 TBLH	3 注	无
LITABRDL	[m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 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 ← 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] ←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 ← 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 ← 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] ←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 ← ACC“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 ← ACC“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] ← ACC“AND”[m]
影响标志位	Z

CALL addr	Subroutine call
指令说明	无条件地调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定地址并从新地址继续执行程序，由于此指令需要额外的运算，所以为一个 2 周期的指令。
功能表示	Stack ← Program Counter + 1 Program Counter ← addr
影响标志位	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] ← 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 ← 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 ← 0
影响标志位	TO、PDF

CPL [m] Complement Data Memory
指令说明 将指定数据存储器中的每一位取逻辑反，
相当于从 1 变 0 或 0 变 1。
功能表示 [m] ← $[\overline{m}]$
影响标志位 Z

CPLA [m] Complement Data Memory with result in ACC
指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。
功能表示 ACC←$[\overline{m}]$
影响标志位 Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory
指令说明 将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] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← ACC + 66H
影响标志位 C

DEC [m] Decrement Data Memory
指令说明 将指定数据存储器内容减 1。
功能表示 [m] ← [m] – 1
影响标志位 Z

DECA [m] Decrement Data Memory with result in ACC
指令说明 将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 ← [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 ← 0 PDF ← 1
影响标志位	TO、PDF

INC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 1。
功能表示	[m] ← [m] + 1
影响标志位	Z

INCA [m]	Increment Data Memory with result in ACC
指令说明	将指定数据存储器的内容加 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 ← [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 ← 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← [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 ← 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] ← 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	无操作
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	ACC ← ACC“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	ACC ← ACC“OR”x
影响标志位	Z

ORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器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] ← ACC“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	Program Counter←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	Program Counter ← Stack ACC←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 ←Stack
EMI ← 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) ← [m].i (i=0~6)
[m].0 ← [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) ← [m].i (i=0~6)
ACC.0 ←[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) ← [m].i (i=0~6)
[m].0 ← C
C ← [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) ← [m].i (i=0~6)
ACC.0 ← C
C ← [m].7
影响标志位 C

RR [m] Rotate Data Memory right
指令说明 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。
功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7 ← [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 ← [m].(i+1) (i=0~6)
ACC.7 ← [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 ← [m].(i+1) (i=0~6)
[m].7← C
C ← [m].0
影响标志位 C

RRCA [m] Rotate Data Memory right through Carry with result in ACC
指令说明 将指定数据存储器的内容连同进位标志右移 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 ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [m].0
影响标志位 C

SBC A, [m] Subtract Data Memory from ACC with Carry
指令说明 将累加器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 ← ACC – [m] – $\overline{C}$
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 ← ACC – [m] – $\overline{C}$
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] ← ACC – [m] – $\overline{C}$
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] ← [m] – 1，如果 [m]=0 跳过下一条指令执行
影响标志位 无

SDZA [m] Skip if decrement Data Memory is zero with result in ACC
指令说明 将指定数据存储器内容减 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 ← [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] ← 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 ← 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] ←[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 ←[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≠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 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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]≠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 ← 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] ← 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 ← 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~[m].0 ↔ [m].7~[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~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0
影响标志位 无

SZ [m]	Skip if Data Memory is 0
指令说明	指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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]	Skip if Data Memory is 0 with data movement to ACC
指令说明	将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	ACC ←[m]，如果 [m]=0，跳过下一条指令执行
影响标志位	无

SZ [m].i	Skip if bit i of Data Memory is 0
指令说明	判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i=0，跳过下一条指令执行
影响标志位	无

TABRD [m]	Read table (specific page) to TBLH and Data Memory
指令说明	将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

TABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

ITABRD [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] ← 程序代码 (低字节)
TBLH ← 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 ← ACC“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] ← ACC“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 ← ACC“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 ← 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] ←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 ← 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] ←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 ← ACC“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] ← ACC“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] ← 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 ← 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] ← $[\overline{m}]$
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←$[\overline{m}]$
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] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← 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] ← [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 ← [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] ← [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 ← [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← [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] ← 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 ← ACC“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和累加器逻辑或，结果放到数据存储器。
功能表示 [m] ← ACC "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) ← [m].i (i=0~6)
[m].0 ← [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 ACC.(i+1) ← [m].i (i=0~6)
ACC.0 ←[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) ← [m].i (i=0~6)
[m].0 ← C
C ← [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 ACC.(i+1) ← [m].i (i=0~6)
ACC.0 ← C
C ← [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 ← [m].(i+1) (i=0~6)
[m].7 ← [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 ← [m].(i+1) (i=0~6)
ACC.7 ← [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 ← [m].(i+1) (i=0~6)
[m].7← C
C ← [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 ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [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 ← ACC – [m] – $\overline{C}$

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] ← ACC – [m] – $\overline{C}$
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] ← [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 ← [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] ← 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 ← 1
影响标志位 无

LSIZ [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 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] ←[m] + 1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

LSIZA [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 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 ←[m] + 1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

LSNZ [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 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≠0，跳过下一条指令执行

影响标志位 无

LSNZ [m] Skip if Data Memory is not 0

指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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]≠0，跳过下一条指令执行

影响标志位 无

LSUB 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 ← 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] ← 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~[m].0 ↔ [m].7~[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~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

影响标志位 无

LSZ [m] Skip if Data Memory is 0

指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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 ← [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

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

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 ← ACC“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] ← ACC“XOR”[m]
影响标志位	Z

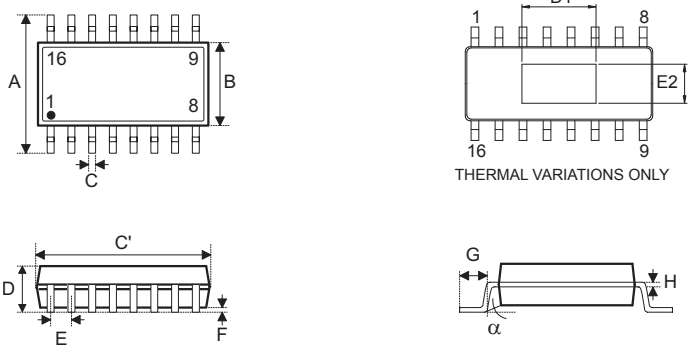
封装信息

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-EP (150mil) 外形尺寸

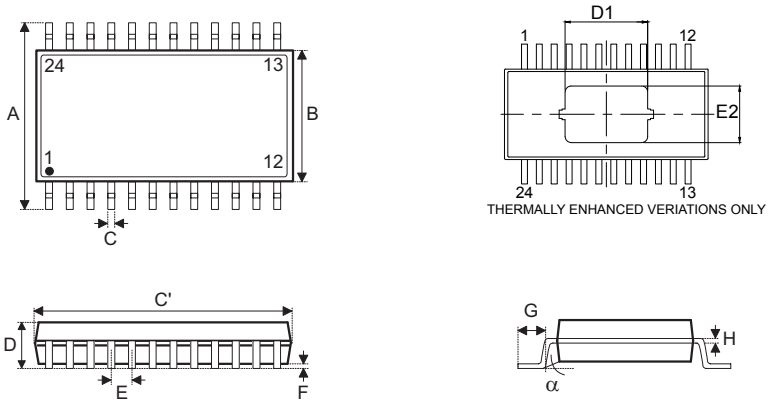


符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C’	—	0.390 BSC	—
D	—	—	0.069
D1	0.152	—	0.186
E	—	0.050 BSC	—
E2	0.066	—	0.101
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C’	—	9.90 BSC	—
D	—	—	1.75
D1	3.86	—	4.72
E	—	1.27 BSC	—
E2	1.68	—	2.56
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

注：针对此封装类型，请参考此处提供的封装信息，Holtek 网站不会对此再做更新。

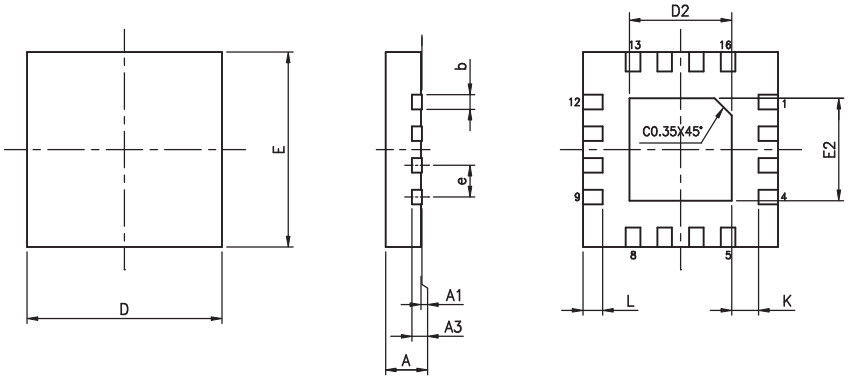
24-pin SSOP-EP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C’	—	0.341 BSC	—
D	—	—	0.069
D1	—	0.140	—
E	—	0.025 BSC	—
E2	—	0.096	—
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)		
	最小值	正常值	最大值
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.20	—	0.30
C’	—	8.66 BSC	—
D	—	—	1.75
D1	—	3.56	—
E	—	0.635 BSC	—
E2	—	2.44	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

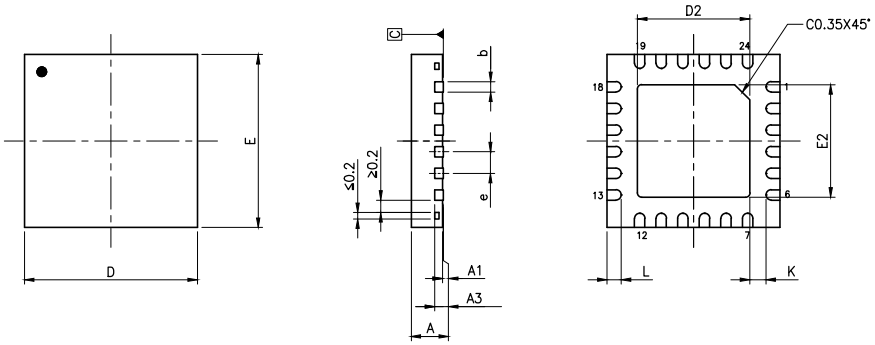
SAW Type 16-pin QFN (4mm×4mm×0.75mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.010	0.012	0.014
D	—	0.157 BSC	—
E	—	0.157 BSC	—
e	—	0.026 BSC	—
D2	0.079	0.083	0.085
E2	0.079	0.083	0.085
L	0.014	0.016	0.018
K	0.008	—	—

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3	—	0.203 BSC	—
b	0.25	0.30	0.35
D	—	4.00 BSC	—
E	—	4.00 BSC	—
e	—	0.65 BSC	—
D2	2.00	2.10	2.15
E2	2.00	2.10	2.15
L	0.35	0.40	0.45
K	0.20	—	—

SAW Type 24-pin QFN (4mm×4mm×0.75mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.007	0.010	0.012
D	—	0.157 BSC	—
E	—	0.157 BSC	—
e	—	0.020 BSC	—
D2	0.104	0.106	0.108
E2	0.104	0.106	0.108
L	0.014	0.016	0.018

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3	—	0.203 BSC	—
b	0.18	0.25	0.30
D	—	4.00 BSC	—
E	—	4.00 BSC	—
e	—	0.50 BSC	—
D2	2.65	2.70	2.75
E2	2.65	2.70	2.75
L	0.35	0.40	0.45