Built - in 12 Bit ADC / PWM / Touch Key / 1T 8051 16K Flash MCU

CA51F1 系列 MCU
中文用户手册

REV1.3

深圳市锦锐科技股份有限公司

电话：0755-83949938

传真：0755-83949977

http://www.cachip.com.cn

地址：中国广东省深圳市南山区沙河西路深圳湾科技生态园一区 2 栋 B 座 5 层

目录

1 概述

CA51F1 系列芯片是基于 1T 8051 内核的 8 位微控制器，通常情况下，运行速度比传统的 8051 芯片快 10 倍，性能更加优越。内置 16K Flash 程序存储器，可多次重复编程的特性，给用户开发带来了极大的方便。不仅保留了传统 8051 芯片的基本特性，还集成了 12 bit ADC、Touch Key、16 Bit PWM、UART、I^2C、RGB_LED 级联控制器以及低电压检测(LVD)等功能模块。支持 IDLE、STOP 和低速运行三种省电模式以适应不同功耗要求的应用。

2 基本特性

- **内核**
  - CPU：1T 8051, 最高速度比传统 8051 快 10 倍
  - 兼容 8051 指令集, 双 DPTR 工作模式
- **存储器**
  - Flash: 16K 字节，支持多次重复擦写
  - Flash 可划分为程序空间和数据空间，数据空间功能类似 EEPROM，可用于存储掉电需要保存的数据
  - RAM:256 字节内部 RAM
- **工作电压**
  - CPU 时钟为 8MHz 时， 工作电压范围：2.2 ~ 5.5V
  - CPU 时钟为 16MHz 时，工作电压范围：2.7 ~ 5.5V
- **时钟系统**
  - 内置低速 RC 振荡器：100KHz，精度±25%
  - 内置高速 RC 振荡器：16MHz，精度为±1.5%
- **TMC 功能**
  - 时钟源为内置低速 RC 振荡器，中断时间最小单位为 512 个低速 RC 振荡器时钟周期。
  - 可配置中断时间为 1-256 个最小单位时间。
- **中断系统**
  - 7 个有效中断源
  - 两级中断优先级，支持中断嵌套
  - 2 个外部中断源
- **定时器**
  - 2 个 16 位通用定时器: 定时器 0，定时器 1
- **通用输入输出口（GPIO）**
  - 最多支持 6 个 GPIO 口，支持推挽、开漏、上拉和下拉、高阻模式
- **模/数转换器（ADC）**

- 支持 6 通道 12 位 SAR ADC
- 支持 2 种基准电压源：VDD 和内部基准(1.5V)
- 选择内部电压为基准可直接测量 VDD 电压

◆ **触摸按键（Touch Key）**
- 内置触摸感应控制器
- 可设置内部充电和内部基准
- 最大支持 5 个触摸通道
- 内置触摸跳频功能，可显著提升抗电压脉冲注入(CS)性能
- 高抗干扰性能，符合 EMC(CS)标准
- 支持触摸省电模式

◆ **PWM**
- 支持 3 通道 PWM，在 16 位范围内可任意配置周期和占空比
- 支持可直接输出内部时钟功能
- 支持 1 路级联 LED 驱动，扫描频率大于 400Hz/S，数据发送速度 800Kbps
- 支持直接控制 WS2812 或类似的驱动芯片，符合单色或七彩 LED 灯带产品的需求

◆ **低电压检测（LVD）**
- 检测电压可设置为 2.2V，2.7V，3.7V 和 4.2V。
- 可设置低电压复位或中断

◆ **复位模式**
- 芯片支持多种复位源：硬复位，软复位，看门狗复位，低电压检测复位，上电/掉电复位

◆ **看门狗**
- 27 位看门狗定时器，16 位调节精度，可配置看门狗复位或中断

◆ **通用串行接口（UART）**
- 支持 1 个 UART 接口
- 支持 1 字节接收缓存

◆ **I^2C 接口**
- 内置 1 路 I^2C 接口，支持主从模式，支持标准/快速/高速模式

◆ **程序下载和仿真**
- 支持 ISP 和 IAP
- 支持在线仿真功能

◆ **低功耗**
- STOP 模式，电流<5uA
- IDLE 模式，电流<20uA
- 低速运行模式，电流<40uA

◆ **封装类型：** SOP8

3 芯片型号功能介绍

表 3-1 CA51F1 系列具体型号功能特点

芯片型号	Flash 容量[BYTE]	内部 Ram[BYTE]	内部高速 RC 振荡器	内部低速 RC 振荡器	GPIO 数量	UART 数量	I²C	16 bit PWM 通道数量	触摸按键数量	12 位 ADC	级联 LED 驱动	通用 16 位定时器数量	ISP	片上仿真功能	工作电压	封装形式
CA51F152S1	16K	256	√	√	6	1	√	3	5	6	1	2	√	√	2.2-5.5	SOP8

4 系统框图

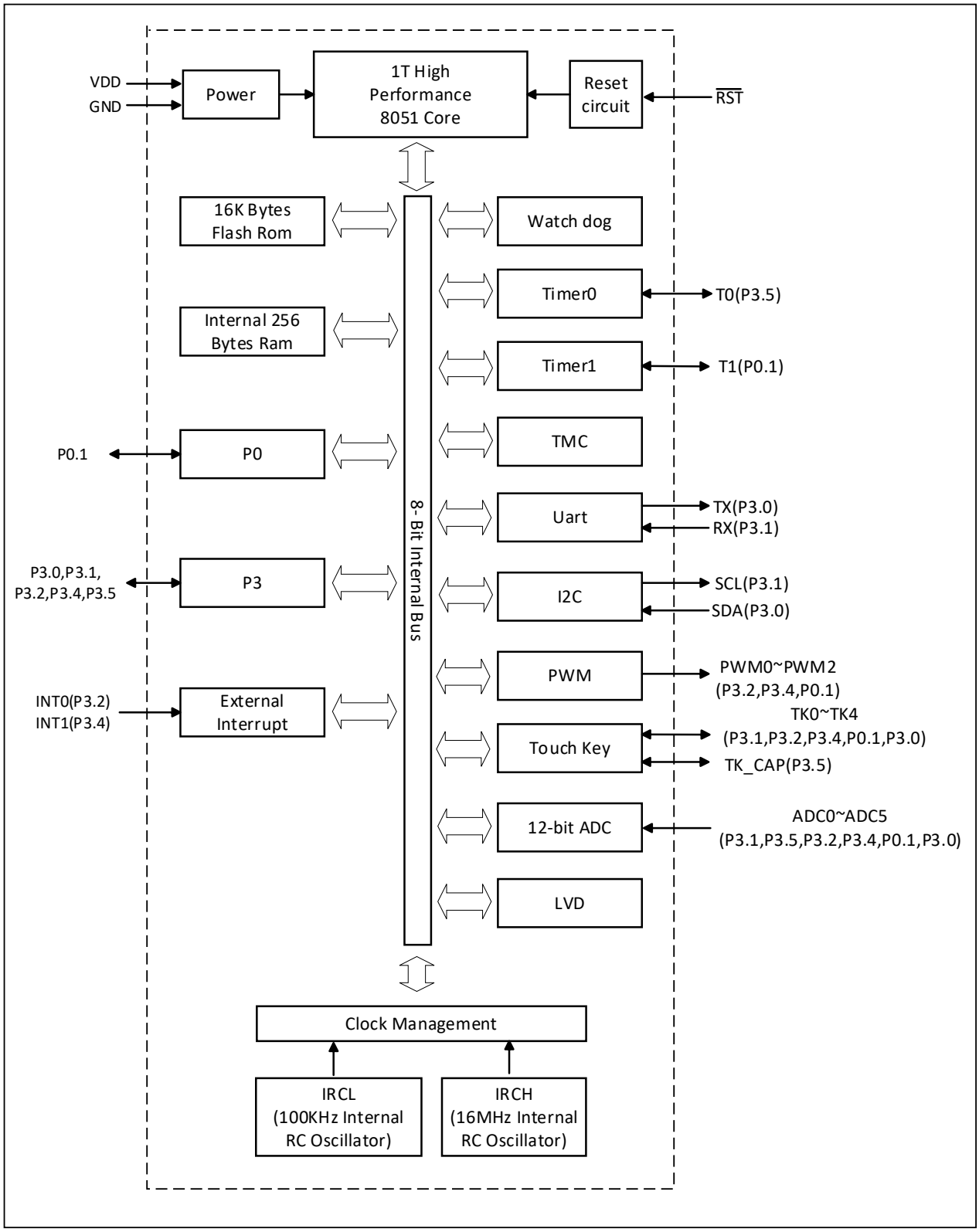


图 4-1-1 芯片框图

5 引脚封装及其描述

5.1 封装定义

型号：CA51F152S1

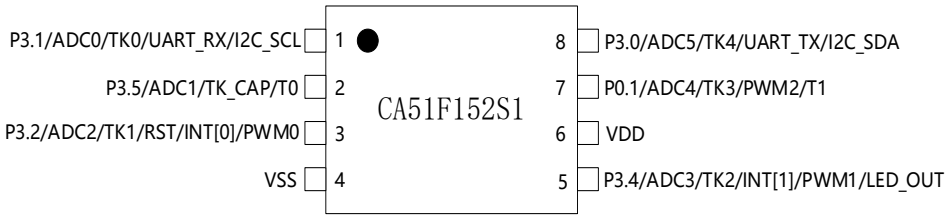


图 5-1-1 SOP8 封装图

5.2 引脚描述

表 5-2-1 引脚描述

引脚序号 SOP8	管脚名称	管脚功能	默认功能
1	P3.1/ADC0/TK0/UART_RX/I2C_SCL	通用双向 I/O 口 UART0_RX 传输口 ADC 模拟通道输入 触摸按键模拟通道输入 I²C 时钟传输口	I²C 时钟传输口
2	P3.5/ADC1/TK_CAP/T0	通用双向 I/O 口 ADC 模拟通道输入 触摸外部电容输入口	通用双向 I/O 口
3	P3.2/ADC2/TK1/RST/INT[0]/PWM0	通用双向 I/O 口 ADC 模拟通道输入 触摸按键模拟通道输入 硬件复位引脚 PWM 信号输出	硬件复位引脚
4	VSS	电源地管脚	电源地引脚
5	P3.4/ADC3/TK2/INT[1]/PWM1/LED_OUT	通用双向 I/O 口 外部复位管脚 触摸按键模拟通道输入 ADC 模拟通道输入 PWM 信号输出/级联 LED 驱动输出	通用双向 I/O 口
6	VDD	芯片供电管脚	芯片供电管脚
7	P0.1/ADC4/TK3/PWM2/T1	通用双向 I/O 口 ADC 模拟通道输入 触摸按键模拟通道输入 PWM 信号输出	通用双向 IO 口
8	P3.0/ADC5/TK4/UART_TX/I2C_SDA	通用双向 I/O 口 UART0_TX 传输口 ADC 模拟通道输入 I²C 数据传输口	I²C 数据传输口

6 中央处理器（CPU）

6.1 CPU 简介

CA51F1 系列芯片采用单周期 8051 CPU，与原来的 MCS-51 指令集完全兼容。CPU 采用流水线结构，通常情况下，单周期 8051 CPU 的运行速度比标准 8051 处理器快 10 倍。

CPU 有以下特性：

- 1T 8051 CPU
- 兼容 8051 指令集，见指令集附录
- 双 DPTR，可用于数据快速搬移

6.2 寄存器描述

程序计数器 PC

程序计数器 PC 寄存器为 16 位，是专门用来控制指令执行顺序的寄存器，它没有寄存器地址。单片机上电或复位后，PC 值为 0，单片机从零地址开始执行程序。

累加器 ACC

累加器 ACC 是一个常用的专用寄存器，指令系统中采用 A 作为累加器的助记符，常用于存放算术或逻辑运算的操作数及运算结果。

通用寄存器 B

B 在乘除法运算中需要和 ACC 配合使用。MUL AB 指令把 ACC 和 B 中 8 位无符号数相乘，所得的 16 位乘积的低字节存放在 A 中，高字节存放在 B 中。DIV AB 指令用 B 除以 A，整数商存放在 A 中，余数存放在 B 中。寄存器 B 还可以用作通用暂存寄存器。

堆栈指针 SP

堆栈指针 SP 是一个 8 位专用寄存器。它指示出堆栈顶部在内部 RAM 块中的位置。系统复位后，SP 初始化位 07H，使得堆栈事实上由 08H 单元开始，考虑 08H~1FH 单元分别 属于工作寄存器组 1~3，若在程序设计中用到这些区，则最好 SP 改变为 80H 或更大的为宜。

数据指针 DPTR

数据指针 DPTR0/DPTR1 是两个 16 位专用寄存器，它们的高位字节寄存器用 DP0H/DP1H 表示，低位字节寄存器用 DP0L/DP1L 表示，通过 DPS(PSW.1)可选择使用 DPTR0/DPTR1。每个 DPTR 既可以作为一个 16 位寄存器来处理，也可以作为 2 个独立的 8 位寄存器 DP0H/DP1H 和 DP0L/DP1L 来处理。

状态寄存器 PSW

状态寄存器 PSW 是 CPU 的状态寄存器。在 CPU 做算术运算或者逻辑运算时，对应的 PSW 状态位会发生改变。

表 6-2-1 累加器 ACC

E0H	7	6	5	4	3	2	1	0
ACC	ACC[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-2 通用寄存器 B

F0H	7	6	5	4	3	2	1	0
B	B[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-3 堆栈指针 SP

81H	7	6	5	4	3	2	1	0
SP	SP[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	1	1	1

表 6-2-4 数据指针 DP0L

82H	7	6	5	4	3	2	1	0
DP0L	DP0L[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-5 数据指针 DP0H

83H	7	6	5	4	3	2	1	0
DP0H	DP0H[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-6 数据指针 DP1L

84H	7	6	5	4	3	2	1	0
DP1L	DP1L[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-7 数据指针 DP1H

85H	7	6	5	4	3	2	1	0
DP1H	DP1H[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

表 6-2-8 状态寄存器 PSW

D0H	7	6	5	4	3	2	1	0
PSW	CY	AC	F0	RS[1:0]		OV	DPS	P
R/W	R/W	R/W	R/W	R/W		R/W	R	R
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	CY	进位标志位 0：算术或逻辑运算中，没有进位或借位发生 1：算术或逻辑运算中，有进位或借位发生
6	AC	辅助进位标志位 0：算术或逻辑运算中，没有辅助进位或借位发生 1：算术或逻辑运算中，有辅助进位或借位发生
5	F0	F0 标志位 用户自定义标志位
4~3	RS	R0~R7 寄存器页选择位 00：页 0（映射到 00H-07H) 01：页 1（映射到 08H-0FH) 10：页 2（映射到 10H-17H) 11：页 3（映射到 18H-1FH)
2	OV	溢出标志位 0：没有溢出发生 1：有溢出发生
1	DPS	DPTR 选择寄存器，0 为选择 DPTR0，1 为选择 DPTR1
0	P	奇偶校验位 0：累加器 A 值为 1 的位数为偶数 1：累加器 A 值为 1 的位数为奇数

表 6-2-9 寄存器 SPMAX

F3H	7	6	5	4	3	2	1	0
SPMAX	SPMAX[7:0]							
R/W	R	R	R	R	R	R	R	R
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	SPMAX	寄存器 SPMAX 用于记录 SP 的最大值，用户在应用程序中可查看此寄存器来判断堆栈有没有溢出风险

7 存储器系统

7.1 随机数据存储器（RAM）

CA51F1 系列芯片提供了 256 字节内部 RAM，存储器地址分配如下：

- 低位 128 字节的内部 RAM（地址：00H ~ 7FH）可直接寻址或间接寻址。
- 高位 128 字节的内部 RAM（地址：80H ~ FFH）只能间接寻址。

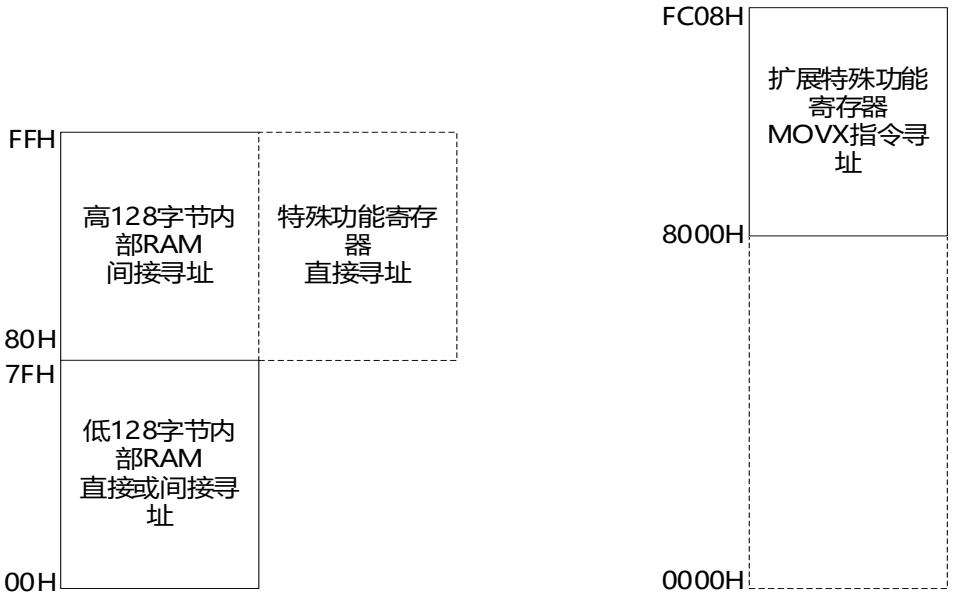


图 7-1-1　RAM 组织结构图

7.2 特殊功能寄存器（SFR）

CA51F1 系列芯片提供了兼容传统 8051 的 SFR 分布，SFR 和高 128 字节内部 RAM 共用地址 80H ~ FFH，只能直接寻址，SFR 映射如表 7-2-1 所示。

表 7-2-1　特殊功能寄存器（SFR）映射表

	可位寻址	不可位寻址						
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8H	TKCON	TKCFG	TKMTS	TKIF	-	-	-	-
F0H	B	-	-	SPMAX	-	-	-	-
E8H	LVDCON	-	-	-	-	-	-	IDCODE
E0H	ACC	TK0MSL	TK0MSH	TK1MSL	TK1MSH	TK2MSL	TK2MSH	TK3MSL
D8H	UDCKS	TK3MSH	TK4MSL	TK4MSH	TK5MSL	TK5MSH	TKCKS	TKPWC
D0H	PSW	-	-	-	-	TMCON	TMSNU	I2CIOS
C8H	CKCON	CKDIV	IHCFG	-	-	-	-	TPCTL

C0H	I2CCON	I2CADR	I2CADM	I2CCR	I2CDAT	I2CSTA	I2CFLG	PWMEN
B8H	IP	PWM0CON	PWM1CON	PWM2CON	-	-	-	-
B0H	P3	PWM0CKD	PWM1CKD	PWM2CKD	-	-	-	-
A8H	IE	PWM0DIVL	PWM0DIVH	PWM1DIVL	PWM1DIVH	PWM2DIVL	PWM2DIVH	-
A0H	WDCON	WDFLG	WDVTHL	WDVTHH	-	-	-	-
98H	SCON	SBUF	-	PWM0DUTL	PWM0DUTH	PWM1DUTL	PWM1DUTH	PWM2DUTL
90H	-	PWM2DUTH	LEDUTL	LEDUTH	LEDWTML	LEDWTMH	LEDAT0	LEFLG
88H	TCON	TMOD	TL0	TL1	TH0	TH1	IDLST	STPST
80H	P0	SP	DP0L	DP0H	DP1L	DP1H	PWCON	PCON

由于 SFR 地址空间有限，CA51F1 系列芯片在外部 RAM 地址空间增加了扩展特殊功能寄存器，扩展特殊功能寄存器映射如图表 7-2-2 所示。

表 7-2-2 扩展特殊功能寄存器映射表

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
8000H	-	P01F	-	-	-	-	-	-
8008H	-	-	-	-	-	-	-	-
8018H	P30F	P31F	P32F	-	P34F	P35F	-	-
8060H	ADCON	ADCFGL	ADCDL	ADCDH	ADCALL	ADCALH	-	-
8068H	SRELL	SRELH						
8120H	-	P01C	-	-	-	-	-	-
8128H	-	-	-	-	-	-	-	-
8138H	P30C	P31C	P32C	-	P34C	P35C	-	-
FC00H	MECON	FSCMD	FSDAT	LOCK	PADRD	PTSL	PTSH	-
FC08H	-	-	-	-	-	-	-	-

7.3 Flash 存储器

7.3.1 功能简介

Flash 存储器包含 16K 字节 Flash 主数据区，Flash 存储器可重复擦写。Flash 存储器由一组特定的寄存器控制，用户可用这些寄存器进行读写擦、设置写保护等操作。

7.3.2 Flash 存储器组织结构

- Flash 由若干个页组成，页是进行擦除和写操作的最小单位，每个页为 64 字节。
- Flash 写操作以页为单位进行，必须一次性写入 64 字节，不支持单字节写入。
- Flash 可以按功能划分为程序区和数据区，划分单位为 128 字节，程序区用于存储用户的程序，数据区是用于存储一些掉电需要保存的数据，功能类似 EEPROM。

图 7-3-2-1　16K Flash 存储器结构

7.3.3 Flash 寄存器描述

表 7-3-3-1 寄存器 MECON

FC00H	7	6	5	4	3	2	1	0
MECON	-	DPSTB	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
初始值	-	0	-	-	-	-	-	-

位编号	位符号	说明
7	-	-
6	DPSTB	IDLE/STOP 模式下 Flash 进入睡眠模式控制位 0：IDLE/STOP 模式下，Flash 处于正常工作模式 1：IDLE/STOP 模式下，Flash 进入睡眠模式 *备注：如果 DPSTB=1，当芯片进入 IDLE/STOP 模式，Flash 也同时进入睡眠模式，当芯片退出 IDLE/STOP 模式，Flash 也同时退出睡眠模式。*
5~0	-	-

表 7-3-3-2 寄存器 FSCMD

FC01H	7	6	5	4	3	2	1	0
FSCMD	IFEN	-	-	-	CLRPL	CMD[2:0]		
R/W	R/W	-	-	-	0	R/W		
初始值	0	-	-	-	0	0	0	0

位编号	位符号	说明
7	IFEN	信息区访问使能位，访问时需将此位置位
6~4	-	-
3	CLRPL	清除 Flash 锁存器中数据
2~0	CMD	命令寄存器 000：无操作 100：Flash 扇区擦除 001：读 Flash 数据区 010：写 Flash 数据区 011：擦除 Flash 数据区一个页 101：读 Flash 程序区 110：写 Flash 程序区 111：擦除 Flash 程序区一个页 *备注：* *1. 擦除和写命令执行后 CMD 自动清零。* *2. 读命令写入后 CMD 保持不变然后通过读写 FSDAT 完成。*

表 7-3-3-3 寄存器 FSDAT

FC02H	7	6	5	4	3	2	1	0
FSDAT	FSDAT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	FSDAT	Flash 数据寄存器

表 7-3-3-4 寄存器 LOCK

FC03H	7	6	5	4	3	2	1	0
LOCK								
R	-	REPE	-	-	FLKF	PLKF	DLKF	ILKF
W	LOCK[7:0]							
初始值	-	0	-	-	0	0	0	0

位编号	位符号	说明
写操作		
7~0	LOCK	28H：对 Flash 可编程区解锁 29H：对 Flash 程序区解锁 2AH：对 Flash 数据区解锁 AAH：Flash 加锁，不能进行写擦操作
读操作		
7~4	-	
3	FLKF	可编程区解锁标志，1 表示已解锁
2	PLKF	程序区解锁标志，1 表示已解锁
1	DLKF	数据区解锁标志，1 表示已解锁
0	-	-

表 7-3-3-5 寄存器 PADRD

FC04H	7	6	5	4	3	2	1	0
PADRD	PADRD[7:0]							
R/W	R/W							
初始值	1	0	0	0	0	0	0	0

位编号	位符号	说明
7	-	-
6~0	PADRD	程序区和数据区划分配置寄存器 程序区和数据区以 128 字节为单位进行划分，当 PADRD>0 时， 程序区的地址空间为: 0 ~ (PADRD×128 - 1), 数据区的地址空间为: (PADRD×128) ~ 3FFFH. *备注：* *1. 当 PADRD=0 时，整个 Flash 空间都是数据空间。* *2. PADRD 的最大值为 80H，PADRD 的设置值不能超过最大值。*

表 7-3-3-6 寄存器 PTS

FC05H	7	6	5	4	3	2	1	0
PTSL	PTS[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
FC06H	7	6	5	4	3	2	1	0
PTSH	-	-	PTS[13:8]					
R/W	-	-	R/W					
初始值	-	-	0	0	0	0	0	0

位编号	位符号	说明

15~14	-	-
13~0	PTS	目标地址指针寄存器，写 FSDAT 操作时，数据会写入 PTS[5:0]相关的 flash 锁存器中暂存，PTS[5:0]对应实际写操作的低 6 位；发送写命令时，需要设置相关的 page 地址 PTS[13:6]。每次读、写、擦操作时最好重新配置一次 PTS 地址，连续读取操作时，可以只设置连续读取操作的首地址操作即可。

7.3.4 Flash 控制例程

◆ **Flash 划分程序区和数据区**

例如，16K 的 Flash 空间划分最后 128 字节为数据空间，其余为程序空间，程序如下：

```
--------------------------------------------------------------------------------
PADRD = 127; //程序区空间地址为：0~0x3F7F,数据区空间地址为：0x3F80~0x3FFF
--------------------------------------------------------------------------------
```

备注：以上设置数据区在 FLASH 中的物理地址是 0x3F80~0x3FFF，但是逻辑地址是 0x0000~0x007F，读写数据区时应填写逻辑地址。

◆ **数据空间页擦除**

例如，需要擦除数据空间页 n，程序如下：

```
--------------------------------------------------------------------------------
FSCMD = 0; //设置 CMD 为 0
LOCK = 0x2A; //数据空间解锁
FSCMD = 8;   //设置擦除 latch
PTSH = (unsigned char)((n*0x40)>>8); //设置扇区高位地址
PTSL = (unsigned char)(n*0x40);      //设置扇区低位地址
FSCMD = 3;   //设置数据区擦除命令
LOCK = 0xAA; //FLASH 加锁
--------------------------------------------------------------------------------
```

备注：页序号 n=0、1、2.....。

◆ **数据空间页写入数据**

例如，往数据空间地址为 n~(n+63)写入数据 0xAA，程序如下：

```
--------------------------------------------------------------------------------
unsigned char i;
FSCMD = 0; //设置 CMD 为 0
LOCK = 0x2A; //数据空间解锁
PTSH = 0;     //设置 page latch 起始地址
PTSL = 0;     //设置 page latch 起始地址
```

```
FSCMD = 8;   //设置擦除 latch
for(i=0;i<64;i++)
{
    FSDAT = 0xAA; //连续写入 1 page 的数据
}
PTSH = (unsigned char)(n>>8); //设置数据首地址高 8 位
PTSL = (unsigned char)n;      //设置数据首地址低 8 位
FSCMD = 2;   //设置写命令
LOCK = 0xAA; //FLASH 加锁
```

备注：

1. 当连续写入数据时，只需设置首地址，每次写 FSDAT 后，数据指针寄存器 PTS 会自动累加。

2. 读写数据区时，设置的地址是数据区的逻辑地址，而不是 FLASH 的物理地址，逻辑地址是从 0 开始的。

3. 数据写入只能以页为单位，每次必须写入 64 字节。

◆ **数据空间读出数据**

例如，从数据空间地址为 n~(n+63)读出数据到指针 dataBuf，程序如下：

```
unsigned int i,dataBuf[64];
FSCMD =       0;
LOCK = CMD_DATA_AREA_UNLOCK;              //数据区解锁
PTSH = (unsigned char)(Address>>8);       //填写高位地址
PTSL = (unsigned char)Address;            //填写低位地址
FSCMD = CMD_DATA_AREA_READ;               //执行读操作
for(i = 0; i < Length; i++)
{
    dataBuf[i]= FSDAT;
}
FSCMD = 0;
LOCK = CMD_FLASH_LOCK;                    //对 FLASH 加锁
```

备注：

1. 当连续读出数据时，只需设置首地址，每次读 FSDAT 后，数据指针寄存器 PTS 会自动累加。

2. 数据读出不需要以页为单位，可以连续读取任意数量字节。

◆ **程序空间扇区擦除**

例如，需要擦除程序空间扇区 n，程序如下：

```
FSCMD = 0; //设置 CMD 为 0
LOCK = 0x29; //程序空间解锁
FSCMD = 8;   //设置擦除 latch
PTSH = (unsigned char)((n*0x40)>>8); //设置扇区高位地址
PTSL = (unsigned char)(n*0x40);      //设置扇区低位地址
FSCMD = 7;   //设置擦除命令
LOCK = 0xAA; //FLASH 加锁
```

备注：扇区序号 n=0、1、2.....。

◆ **程序空间写入数据**

例如，往程序空间地址为 n~(n+63)写入数据 0xAA，程序如下：

```
unsigned char i;
FSCMD = 0; //设置 CMD 为 0
LOCK = 0x29; //程序空间解锁
FSCMD = 8;   //设置擦除 latch
PTSH = 0;    //设置 page latch 起始地址
PTSL = 0;    //设置 page latch 起始地址
for(i=0;i<64;i++)
{
    FSDAT = 0xAA; //连续写入 1page 数据
}
PTSH = (unsigned char)(n>>8);   //设置数据首地址高 8 位
PTSL = (unsigned char)n;        //设置数据首地址低 8 位
FSCMD = 6;   //设置写命令
LOCK = 0xAA; //FLASH 加锁
```

备注：

1. 当连续写入数据时，只需设置首地址，每次写 FSDAT 后，数据指针寄存器 PTS 会自动累加。

2. 数据写入只能以页为单位，每次必须写入 64 字节。

◆ **程序空间读出数据**

例如，从程序空间地址为 n~(n+63)读出数据到指针 dataBuf，程序如下：

```
unsigned char i, dataBuf[64];
FSCMD = 0; //设置 CMD 为 0
LOCK = 0x29; //程序空间解锁
```

```
PTSH = (unsigned char)(n>>8); //设置数据首地址高 8 位
PTSL = (unsigned char)n;      //设置数据首地址低 8 位
FSCMD = 5;   //设置读命令
for(i=0;i<64;i++)
{
    dataBuf[i] = FSDAT ; //连续写入数据
}
FSCMD   =     0;
LOCK = 0xAA; //FLASH 加锁
```

备注：　1. 当连续读出数据时，只需设置首地址，每次读 FSDAT 后，数据指针寄存器 PTS 会自动累加。

2.数据读出不需要以页为单位，可以连续读取任意数量字节。

8 中断系统

8.1 功能简介

CA51F1 系列芯片有一个增强的中断控制系统，共有 7 个中断入口，每个中断入口有若干中断源，每个中断源有 2 级中断优先级。每个中断源都有独立的中断向量、优先级设置位、中断使能位、中断标志。CPU 在响应中断后，进入该中断对应的中断服务程序，接到 RETI 指令后将返回中断前状态。如果同时有多个有效中断产生中断请求，CPU 将根据设置的中断优先级依次响应；如果优先级相同，则根据它们的自然优先级（中断入口地址从低到高）依次响应。

8.2 中断逻辑

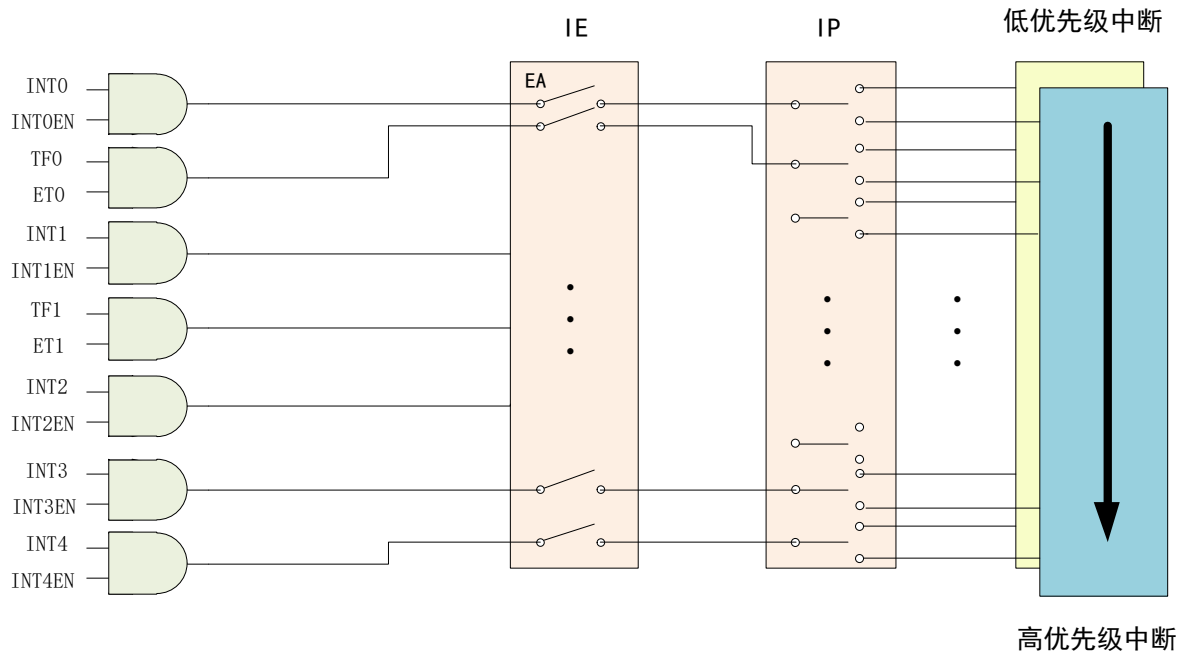


表 8-2-1 中断逻辑图

8.3 中断向量表

表 8-3-1 中断向量表

中断	中断源	向量	默认优先级
INT0	INT0	03H	0
TF0	定时器 0	0BH	1
INT1	INT1	13H	2
TF1	定时器 1	1BH	3
INT2	UART	23H	4
INT3	触摸按键中断/TMC 中断	2BH	5
INT4	WDT 中断/I2C 中断/LVD 中断	33H	6

8.4 中断控制寄存器

表 8-4-1 寄存器 IE

A8H	7	6	5	4	3	2	1	0
IE	EA	INT4EN	INT3EN	INT2EN	ET1	INT1EN	ET0	INT0EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	EA	全局中断使能控制位 0：关闭 1：打开
6	INT4EN	中断 4 使能控制位（中断 4 用于 WDT/I2C/LVD/外部中断 4） 0：关闭 1：打开
5	INT3EN	中断 3 使能控制位（中断 3 用于 TMC/TK） 0：关闭 1：打开
4	INT2EN	中断 2 使能控制位（中断 2 用于 UART） 0：关闭 1：打开
3	ET1	定时器 1 中断使能控制位 0：关闭 1：打开
2	EX1	中断 1 使能控制位（中断 1 用于外部中断 1） 0：关闭

		1：打开
1	ET0	定时器 0 中断使能控制位 0：关闭 1：打开
0	EX0	中断 0 使能控制位（中断 0 用于外部中断 0） 0：关闭 1：打开

备注：IE 的使能控制位是对应中断向量的，各中断源的中断开关也要另外打开。

表 8-4-2 寄存器 IP

B8H	7	6	5	4	3	2	1	0
IP	-	PS1	PT2	PS0	PT1	PX1	PT0	PX0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	-	-
6	PS1	中断 INT4 优先级控制位 0：低优先级 1：高优先级
5	PT2	中断 INT3 优先级控制位 0：低优先级 1：高优先级
4	PS0	中断 INT2 优先级控制位 0：低优先级 1：高优先级
3	PT1	定时器 1 优先级控制位 0：低优先级 1：高优先级
2	PX1	外部中断 1 优先级控制位 0：低优先级 1：高优先级
1	PT0	定时器 0 优先级控制位 0：低优先级 1：高优先级
0	PX0	外部中断 0 优先级控制位 0：低优先级 1：高优先级

8.5 外部中断

8.5.1 外部中断介绍

INT0/INT1 的中断引脚分别为 P3.2/P3.4，功能基本兼容标准 8051。INT0 和 INT1 可选择上升沿或下降沿触发，选择位分别为 IT0 和 IT1，详见寄存器 TCON 相关描述。INT0 和 INT1 可以用于 STOP 模式唤醒。

8.5.2 外部中断控制例程

◆ **外部中断 0/1 控制例程**

例如，使能外部中断 0，程序如下:

```
---------------------------------------------------------------------------------
void INT0_init(void)
{
    P32F = 1;       //外部中断 0 的中断引脚为 P32，设置 P32 为输入功能
    EX0 = 1;        //INT0 中断使能
    IE0 = 1;        //外部中断 0 使能
    IT0 = 1;        //设置为下降沿中断
    PX0 = 1;        //设置 INT0 为高优先级
    EA = 1;         //总中断使能
}
void INT0_ISR (void) interrupt 0
{
    //外部中断 0 中断服务程序
}
---------------------------------------------------------------------------------
```

例如，使能外部中断 1，程序如下:

```
---------------------------------------------------------------------------------
void INT1_init(void)
{
    P34F = 1;       //外部中断 1 的中断引脚为 P34，设置 P34 为输入功能
    EX1 = 1;        //INT1 中断使能
    IE1 = 1;        //外部中断 1 使能
    IT1 = 1;        //设置为下降沿中断
    PX1 = 1;        //设置 INT1 为高优先级
    EA = 1;         //总中断使能
}
void INT1_ISR (void) interrupt 2
{
    //外部中断 1 中断服务程序
}

---------------------------------------------------------------------------------
```

9 时钟系统

9.1 时钟系统介绍

CA51F1 系列芯片共支持以下时钟源：

- 内置 16MHz RC 振荡器
- 内置 100KHz RC 振荡器

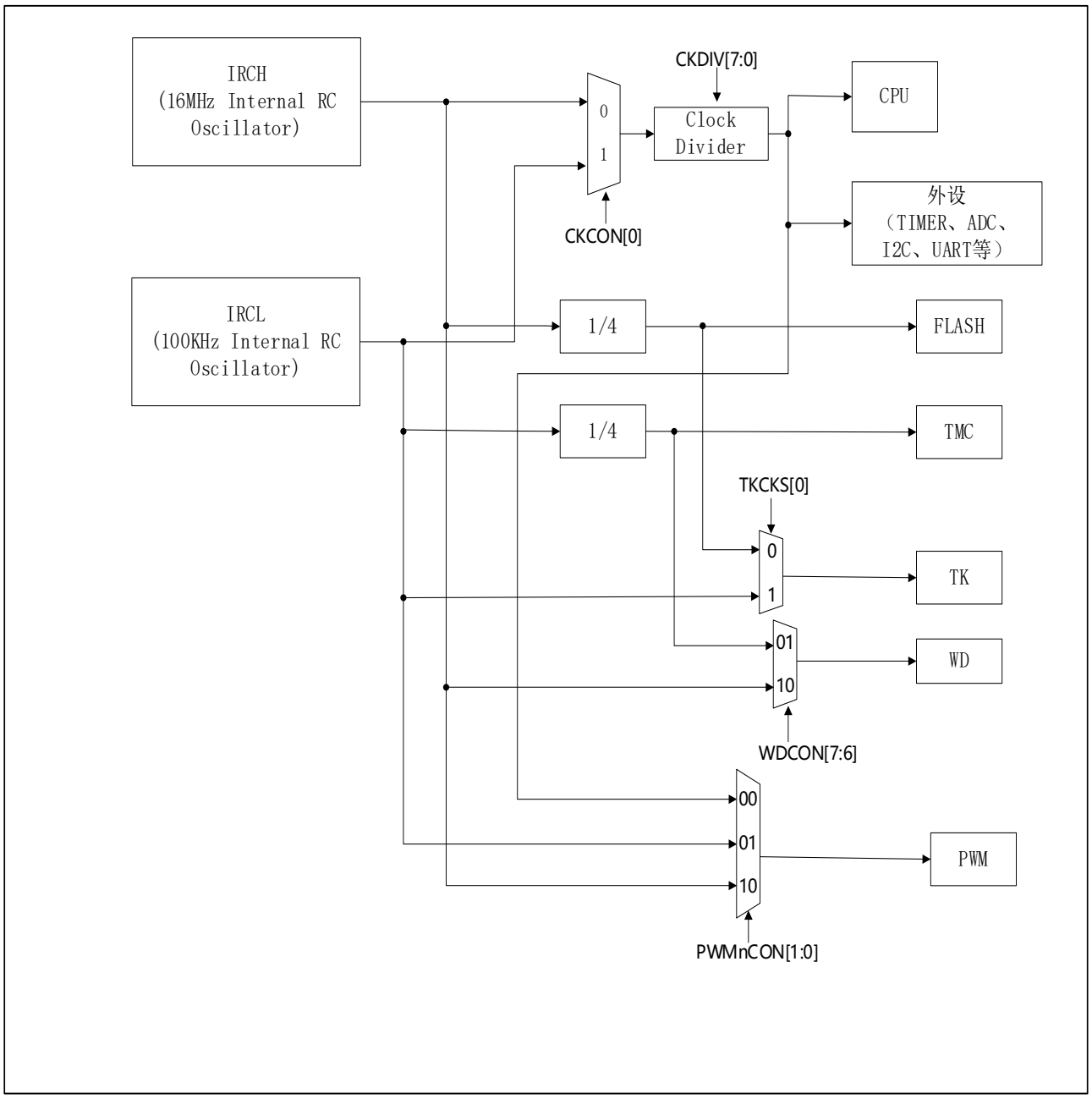


图 9-1-1 时钟结构图

用户可独立的管理各个时钟源，每个时钟源都可以单独打开或关闭，从而可以灵活控制功耗。

9.1.1 时钟专用名称定义

名称缩写	描述
IRCH	内置 16MHz RC 振荡器
IRCL	内置 100KHz RC 振荡器

9.1.2 内置 16MHz RC 振荡器（IRCH）

IRCH 是芯片上电后默认的系统时钟，可通过寄存器 CKCON 的 IHCKE 位打开或关闭。芯片出厂后，IRCH 的频率校正为 16MHz@3.3V/25℃，时钟精度为±1%。

9.1.3 内置 100 KHz RC 振荡器（IRCL）

IRCL 可通过寄存器 CKCON 的 ILCKE 位打开或关闭。IRCL 设为系统时钟可实现系统低功耗。IRCL 的时钟精度为±25%@3.3V/25℃。

9.1.4 时钟控制寄存器描述

表 9-2-2-6 寄存器 IHCFG

CAH	7	6	5	4	3	2	1	0
IHCFG	IHCFG[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7~0	IHCFG	IRCH 频率调整寄存器						

9.2 系统时钟

系统时钟控制由寄存器 CKCON、CKDIV 完成。通过这些寄存器组，可以单独设置各时钟源的开关、系统时钟的切换和分频等操作。

系统时钟有两个时钟可选：IRCH 和 IRCL，上电后，默认的系统时钟是 IRCH，并且 CKDIV 值为 1，即系统时钟上电默认为 IRCH 的二分频，如 CPU 需运行于更高频率，软件可设置 CKDIV 为 0。

9.2.1 系统时钟结构图

系统时钟结构图见图 9-2-1。

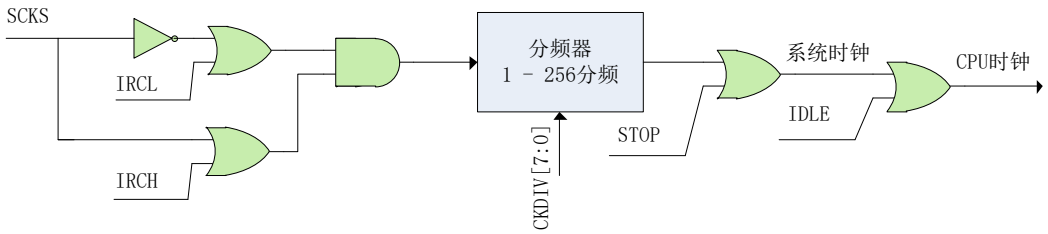


图 9-2-1 系统时钟结构图

9.2.2 系统时钟控制寄存器描述

表 9-3-1 寄存器 CKCON

C8H	7	6	5	4	3	2	1	0
CKCON	IHCKE	ILCKE	-	-	-	-	-	SCKS
R/W	R/W	R/W	-	-	-	-	-	R/W
初始值	0	0	-	-	-	-	-	0

位编号	位符号	说明
7	IHCKE	IRCH 使能控制位 1：打开 0：关闭 *备注：* *该位为 1 时，时钟模块打开，但是该位为 0 时，如果系统或者其他模块选择了该时钟源，该时钟仍然会被打开。*
6	ILCKE	IRCL 使能控制位 1：打开 0：关闭

		备注： *该位为1 时，时钟模块打开，但是该位为0 时，如果系统或者其他模块选择了该时钟源，该时钟仍然会被打开。*
-	-	-
1-0	SCKS	系统时钟选择位 0：选择 IRCH 1：选择 IRCL

表 9-3-2 寄存器 CKDIV

C9H	7	6	5	4	3	2	1	0
CKDIV	CKDIV[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	1
位编号	位符号	说明						
7~0	CKDIV	系统时钟分频： 00H：不分频 01H：2 分频 02H：3 分频 03H：4 分频 …… FFH：256 分频						

9.3.3 系统时钟控制方法及例程

◆ **设置系统时钟为 IRCH**

设置系统时钟为 IRCH，程序如下：

```
---------------------------------------------------------------------------------
#define IHCKE               (1<<7)
#define CKSEL_IRCH     0
void Sys_Clk_Set_IRCH(void)
{
      CKCON |= IHCKE;                                  //打开 IRCH 时钟
      CKCON = (CKCON&0xFE) | CKSEL_IRCH;       //设置系统时钟为 IRCH
}
---------------------------------------------------------------------------------
```

◆ **设置系统时钟为 IRCL**

设置系统时钟为 IRCL，程序如下：

```
---------------------------------------------------------------------------------
#define ILCKE        (1<<6)
#define CKSEL_IRCL     1
void Sys_Clk_Set_IRCL(void)
{
      CKCON |= ILCKE;                                  //打开 IRCL 时钟
      Delay_ms(1); //延时 1ms，等待 IRCL 时钟稳定
      CKCON = (CKCON&0xFE) | CKSEL_IRCL;       //设置系统时钟为 IRCL
}
---------------------------------------------------------------------------------
```

10 供电和复位系统

10.1 供电系统

在 CA51F1 系列芯片 VDD 和 VSS 引脚间接入 2.2V - 5.5V 的电源，此电源可直接给芯片内部数字及模拟系统供电。需要注意的是，不同的供电电压条件下，芯片支持运行的最高频率和功耗并不相同，具体请查看电气特性章节。

芯片内部还设计了 BANDGAP 基准电压，作为 ADC 内部参考电压、LVD 电压、触摸内部运放等的基准电压源，此基准电压源在出厂经校准后，精度为±30mV。

10.1.2 内部基准电压控制寄存器

表 10-1-2-1 寄存器 PWCON

86H	7	6	5	4	3	2	1	0
PWCON	FLEVEL[3:0]				**VREFS**	-	-	-
R/W	R/W				**R/W**	-	-	-
初始值	0	1	1	1	**0**	-	-	-

位编号	位符号	说明
7~4	FLEVEL	内部基准电压（Bandgap）输出调整位域 0000：0.825V 0001：0.850V 0010：0.875V 0011：0.900V 0100：0.925V 0101：0.950V 0110：0.975V 0111：1.000V 1000：1.025V 1001：1.050V 1010：1.075V 1011：1.100V 1100：1.125V 1101：1.150V 1110：1.175V

		1111：1.200V *备注：此基准电压出厂经校准后精度为±30mV，校准值上电自动加载，用户不允许更改。*
3	VREFS	参考电压的驱动选择： 0： 0.8uA 的驱动能力， default 1：运放驱动
2-0	-	-

10.2 复位系统

CA51F1 系列芯片有多个内部和外部复位源，如图 10-2-1 所示。

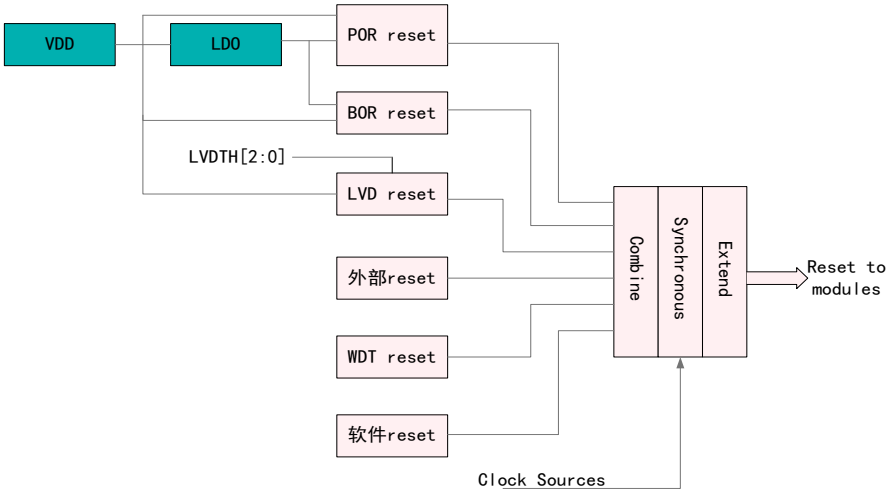
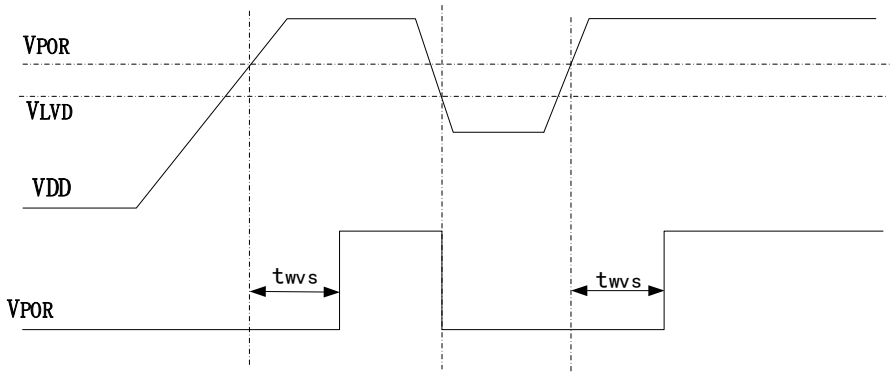


图 10-2-1 复位系统结构图

- **上电复位（POR）**

系统上电呈现逐渐上升的曲线形式，需要一定时间才能达到正常的工作电压。上电复位是基于电源电压 VDD，当电压低于检测阈值时，上电复位信号有效。

上电复位电路能够保证芯片在上电过程中处于复位状态，芯片上电后能够从一个已知的稳定的状态开始运行。上电复位信号也会被芯片内部的计数器展宽，以保证上电后内部的各种模拟模块能够进入稳定的工作状态。



twvs：等待电压稳定时间

图 10-2-2 上电复位电路示例及上电过程

- **掉电复位（BOR）**

利用掉电复位，可以为芯片提供电源跌落(例如受到干扰或者负载变化)的预警信号。一旦发现电源电压 VDD 或内部 LDO 的输出电压下降到某一个阈值时，就使芯片及时复位以免系统工作状态不正常或者程序执行错误。

- **低电压复位**

低电压检测（LVD）可以在多种工作模式下持续监控电源电压 VDD。当 VDD 低于 LVD 设定的域值电压超过 20us 就可以产生复位信号（前提是 LVD 设置为复位模式）。

- **外部复位**

通过拉低复位引脚(RESET)，可以从外部源复位器件。在正常工作情况下，RESET 可以复位整个芯片，在 STOP 状态，硬复位会唤醒芯片后再复位。一般情况下，RESET 被内部上拉拉高，不会影响内部的复位电路。

- **看门狗复位**

看门狗定时器负责监控处理器执行指令的情况，通过合适的配置，如果看门狗定时器在特定时间段内未被刷新，则可以产生复位信号。上电复位后，看门狗定时器是关闭的，用户需要时，再配置开启。

- **软复位**

芯片可以在程序控制下执行软复位。通过对 PCON 寄存器中的 SWRST 位写 1，CPU 可以发出复位指令。

上电掉电复位及外部硬复位将复位所有的电路，LVD 和 WDT 的复位不能复位其本身电路，但可以复位其他电路（例如：WDT 复位产生后，WDT 模块电路没有复位，WDT 寄存器还保持复位之前的状态，但 WDT 之外的电路已经复位了）。LVD/WDT 和软复位都不能复位存储控制电路。所有复位产生之后，PC 都将指向地址 0。

11 功耗管理

CA51F1 系列芯片有三种不同的低功耗模式: IDLE 模式、STOP 模式、低速运行模式。IDLE 模式时系统电流小于 20uA，STOP 模式时系统电流小于 5uA，低速运行时电流小于 40uA。

11.1 IDLE 模式

在 IDLE 模式下，CPU 将停止工作。进入 IDLE 模式前，除了主时钟，其他的时钟源根据需要都可选择关闭，以便节省功耗。同样地，进入 IDLE 模式前，可根据需要设定芯片某些外设的开关。打开的外设在 IDLE 状态下仍然可以正常工作。

设置进入 IDLE 模式前，需要先查看一下寄存器 IDLST（IDLSTH 和 IDLSTL），如果所有位都为 0，则设置进入 IDLE 模式后，CPU 将正常进入 IDLE 模式。如果 IDLST 的位不全为 0，即使有设置进入 IDLE 模式的操作，CPU 也不会进入 IDLE 模式，而是继续停留在正常工作模式。此时用户需先把 IDLST 对应位的中断处理完成，再重新设置进入 IDLE 模式的动作。

所有复位事件和任何中断事件都将唤醒芯片。中断唤醒 CPU 后，芯片首先将恢复时钟，然后响应该中断，进入该中断的服务程序。退出中断服务程序后，芯片将执行置位 IDLE 指令后面的指令。退出 IDLE 模式时，IDLE 位将自动清零。

需要注意的是，在置位 IDLE 的指令后面需要紧接两条 nop 指令，防止程序出错。

11.2 STOP 模式

STOP 模式是比 IDLE 更深层次的低功耗模式。STOP 模式可以停止所有时钟（包括主时钟）和时钟产生电路。如果 WDT 和 RTC 处于打开状态，则它们使用的时钟模块将处于工作状态，可以有选择地关闭 WDT 和 RTC 以节省功耗。

类似于 IDLE 模式，进入 STOP 模式前，需要先查看 STPST（STPSTH 和 STPSTL）寄存器，若有置 1 的位存在，需要先行处理，以确保能顺利进入 STOP 模式。

STOP 模式可以通过外部中断、LVD 中断或复位、硬复位、RTC 中断、WDT 中断或复位、时钟监控中断、触摸中断来唤醒。如果是中断唤醒，那么唤醒 MCU 后，芯片首先将恢复时钟，然后响应该中断，进入该中断的服务程序。退出中断服务程序后，芯片将执行置位 STOP 指令后面的指令。退出 STOP 模式时，STOP 位将自动清零。

为了更好的唤醒芯片，推荐在进入 STOP 模式前切换系统时钟到内部时钟，因为唤醒时，外部时钟需要更多时间去等待稳定。

在进入 STOP 模式时，最后一个时钟沿将关闭系统时钟，然后芯片完全进入 STOP 模式。需要注意的是，在置位 STOP 的指令后面需要紧接三条 nop 指令，防止程序出错。

11.3 低速运行模式

由于芯片的功耗与运行速度直接相关，所以把主时钟切换到低速时钟运行也可以显著降低功耗。系统设为IRCL（频率为 100KHz）时的电流小于 40uA。

11.4 低功耗相关寄存器描述

表 11-4-1 寄存器 PCON

87H	7	6	5	4	3	2	1	0
PCON	-	-	SWRST	-	-	TSMODE	STOP	IDLE
R/W	-	-	W	-	-	R	W	W
初始值	-	-	0	-	-	0	0	0

位编号	位符号	说明
7~6	-	-
5	SWRST	软复位控制位，1 有效 设置 SWRST=1 产生软复位，复位产生后自动清 0。
4~3	-	-
2	TSMODE	在线仿真模式标志位，为 1 表示芯片正工作于在线仿真模式
1	STOP	STOP 模式控制位，1 有效 当设置 STOP=1 且 STPST 为 0 时，芯片进入 STOP 模式，退出 STOP 模式后自动清 0
0	IDLE	IDLE 模式控制位，1 有效 当设置 IDLE=1 且 IDLST 为 0 时，芯片进入 IDLE 模式，退出 IDLE 模式后自动清 0

表 11-4-2 寄存器 IDLST

8EH	7	6	5	4	3	2	1	0
IDLST	-	IDLSTL[6:0]						
R/W	-	R						
初始值	-	0	0	0	0	0	0	0

位编号	位符号	说明
7	-	-
6	I2CINT/WDIF/LVDINT/EPIF[2]	IDLE 模式时，I²C/WDT/LVD/外部中断 4 的中断状态
5	TKINT/TMINT/EPIF[1]	IDLE 模式时，触摸按键/TMC/外部中断 3 的中断状态
4	UART/EPIF[0]/ADC	IDLE 模式时，UART/外部中断 2 的中断状态/ADC
3	TF1	IDLE 模式时，定时器 1 的中断状态
2	PIF[1]	IDLE 模式时，外部中断 1 的中断状态
1	TF0	IDLE 模式时，定时器 0 的中断状态
0	PIF[0]	IDLE 模式时，外部中断 0 的中断状态

表 11-4-2 寄存器 STPST

8FH	7	6	5	4	3	2	1	0
STPST	-	STPSTL [6:0]						
R/W	-	R						
初始值	-	0	0	0	0	0	0	0

位编号	位符号	说明
7	-	-
6	WDTWKF/LVDWKF/I2CWKF	STOP 模式时，WDT/LVD/I²C 的中断状态
5	TKWKF/TMWKF	STOP 模式时，触摸按键/TMC 的中断状态
4	EPWKF[2]	STOP 模式时，外部中断 4 的中断状态
3	EPWKF[1]	STOP 模式时，外部中断 3 的中断状态
2	EPWKF[0]	STOP 模式时，外部中断 2 的中断状态
1	PWKF[1]	STOP 模式时，外部中断 1 的中断状态
0	PWKF[0]	STOP 模式时，外部中断 0 的中断状态

11.5 低功耗模式控制例程

◆ **STOP 模式例程**

STOP 模式程序如下：

```
---------------------------------------------------------------------------------------
void Stop(void)
{
    I2CCON = 0;  //关闭 I2C 功能，因为 I2C 默认是使能的，如果 I2C 不关闭将无法关闭 IRCH 时钟
    CKCON = 0;  //关闭所有时钟
    MECON |= (1<<6); //设置 FLASH 进入深度睡眠状态
    PCON |= 0x02;       //进入 STOP 模式
    _nop_();
    _nop_();
    _nop_();
}
---------------------------------------------------------------------------------------
```

◆ **IDLE 模式例程**

IDLE 模式程序如下：

```
---------------------------------------------------------------------------------------
#define IHCKE           (1<<7)
#define ILCKE           (1<<6)

#define CKSEL_IRCH    0
#define CKSEL_IRCL    1
```

```
void Idle(void)
{
    CKCON |= ILCKE;                          //IRCL 时钟使能
    Delay_ms(1);                             //延时 1ms，等待 IRCL 时钟稳定
    CKCON = (CKCON&0xFE) | CKSEL_IRCL; //系统时钟切换到 IRCL
    I2CCON = 0;       //关闭 I2C 模块，因为 I2C 默认是使能的，如果 I2C 不关闭将无法关闭 IRCH 时钟
    CKCON&=~IHCKE;                           //关闭 IRCH 时钟
    MECON |= (1<<6);                         //设置 FLASH 进入深度睡眠状态
    while(IDLST&0x7F);                       //如果有中断未响应,等待中断被响应
    PCON |= 0x01;                            //进入 IDLE 模式
    _nop_();
    _nop_();
}
```

备注：由于进入 IDLE 后，主时钟仍是打开的，如果进入 IDLE 前主时钟是高速时钟，进入 IDLE 模式后功耗仍会很大，所以进入 IDLE 之前需要把主时钟切换到低速时钟。

◆ **低速运行模式例程**

低速运行模式程序如下：

```
#define IHCKE           (1<<7)
#define ILCKE           (1<<6)

#define CKSEL_IRCH    0
#define CKSEL_IRCL    1

void LowSpeedMode(void)
{
    I2CCON = 0;           //关闭 I2C 模块，因为 I2C 默认是使能的，如果 I2C 不关闭将无法关闭 IRCH 时钟
    CKCON |= ILCKE;                                          //IRCL 时钟使能
    Delay_ms(1); //延时 1ms，等待 IRCL 时钟稳定
    CKCON = (CKCON&0xFE) | CKSEL_IRCL;     //系统时钟切换到 IRCL
    CKCON &= ~IHCKE;       //关闭 IRCH 时钟
}
```

---

12 通用定时器（定时器 0, 定时器 1）

12.1 定时器 0

12.1.1 定时器 0 介绍

定时器或计数器功能通过 CT0 位（TMOD[2]）来选择，CT0=0 选择为定时器，CT0=1 选择为计数器。作为定时器时，时钟是系统时钟的 12 分频。作为计数器时，时钟是 T0 的输入时钟。由于检测 T0 输入边沿变化需要 2 个时钟周期，所以作为计数器时最大的输入波特率是内部系统时钟频率的 1/2。T0 输入信号在占空比上没有限制，然而为了完全识别 0 或 1 的状态，信号至少需要保持 1 个内部系统时钟周期时间。定时器 0 有 4 个工作模式，通过 T0M0、T0M1 位(TMOD[1:0])来选择。

- **模式 0**

在此模式下，定时器 0 作为 13 位定时器/计数器，TH0 存放 13 位定时器/计数器的高 8 位，TL0[4:0]存放低 5 位，而 TL0[7:5]是无效的，在读取时应被忽略。当定时器 0 溢出，中断标志位 TF0（TCON[5]）会被置 1。中断被响应后，TF0 位会自动清 0。当 GATE0（TCON[3]）=0 时，定时器/计数器由 TR0（TCON[4]）位使能计数，当 GATE0=1 时，定时器/计数器由引脚 INT0 控制使能，INT0 为高电平时计数，INT0 为低电平则停止计数。

- **模式 1**

此模式下，定时器 0 作为 16 位定时器/计数器，除此之外，功能与模式 0 完全相同。

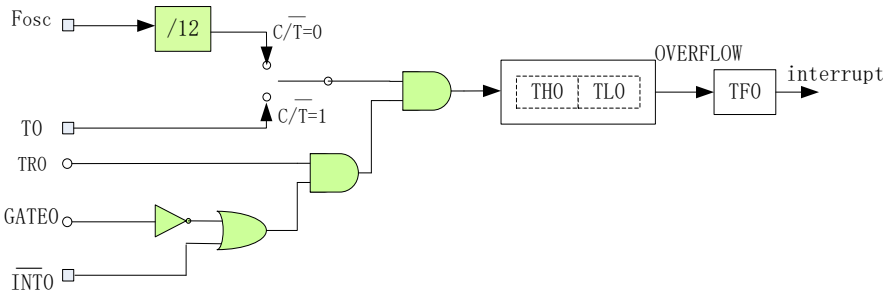


图 12-1-1-1 定时器 0 的模式 0 与 1

- **模式 2**

在此模式中，定时器 0 作为 8 位自动重载定时器/计数器，只有 TL0 自动累加。当 TL0 计数溢出时，不但产生中断标志 TF0，而且从 TH0 中自动装载计数初始值到 TL0。其他设置方法和模式 0、1 相同。

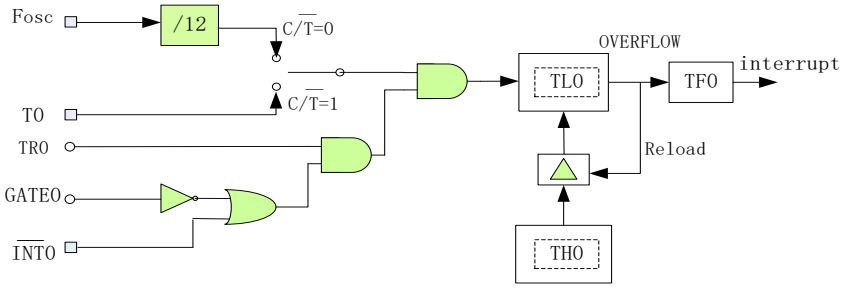


图 12-1-1-2 定时器 0 的模式 2

- **模式 3**

在此模式中，TL0 和 TH0 作为两个独立的 8 位定时器/计数器。TL0 可以作为定时器或计数器，而 TH0 只能作为定时器。其中 TL0 占用定时器 0 的控制位 CT0、GATE0、TR0、TF0、INT0，而 TH0 只能占用定时器 1 的控制位 TR1、TF1。其他控制方法和模式 0、1 相同。当定时器 0 工作于模式 3 时，定时器 1 和 TH0 共用控制位 TR1，但定时器 1 由于 TF1 已被 TH0 占用，所以只能工作于不需要产生中断的场合，例如作为 UART 的波特率产生器。

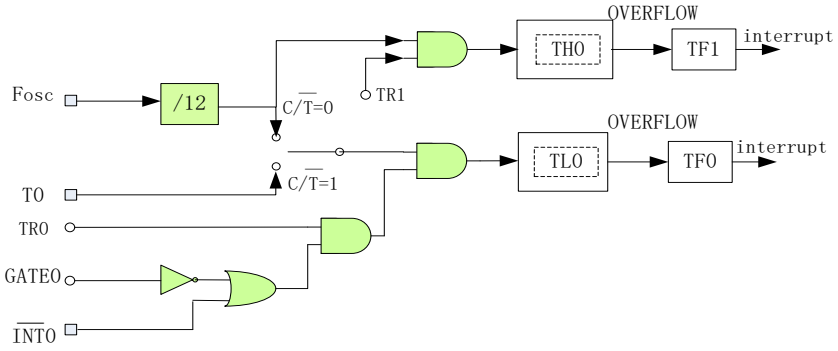


图 12-1-1-3 定时器 0 的模式 3

12.1.2 定时器 0 寄存器描述

表 12-1-2-1 寄存器 TCON

88H	7	6	5	4	3	2	1	0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	TF1	定时器 0 模式 3 的 TH0 溢出/定时器 1 溢出标志位，中断被响应后自动清 0.
6	TR1	定时器 1 运行控制位，1 有效
5	TF0	定时器 0 溢出标志位，中断被响应后自动清 0.
4	TR0	定时器 0 运行控制位，1 有效
3	IE1	外部中断 1 使能位，1 有效
2	IT1	外部中断 1 触发类型控制位 0：外部中断 1 在输入管脚上升沿时触发 1：外部中断 1 在输入管脚下降沿时触发
1	IE0	外部中断 0 使能位，1 有效
0	IT0	外部中断 0 触发类型控制位 0：外部中断 0 在输入管脚上升沿时触发 1：外部中断 0 在输入管脚下降沿时触发

表 12-1-2-2 寄存器 TMOD

89H	7	6	5	4	3	2	1	0
TMOD	GATE1	CT1	T1M1	T1M0	GATE0	CT0	T0M1	T0M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	GATE1	定时器 1 门控控制位，1 有效。有效时定时器 1 由 INT1 控制开关
6	CT1	定时器 1 计数器/定时器选择位 0：定时器，时钟为系统时钟 12 分频 1：计数器，时钟为 T1 输入时钟
5	T1M1	[T1M1,T1M0]为定时器 1 模式选择位 00：模式 0，TL1 和 TH1 组成 13 位定时器/计数器 01：模式 1，TL1 和 TH1 组成 16 位定时器/计数器 10：模式 2，TL1 作为 8 位定时器/计数器，TH1 作为自动重载寄存器 11：模式 3，此模式会锁住 TH1/TL1，等效于 TR1=0
4	T1M0	
3	GATE0	定时器 0 门控控制位，1 有效。有效时定时器 0 由 INT0 控制开关
2	CT0	定时器 0 计数器/定时器选择位 0：定时器，时钟为系统时钟 12 分频 1：计数器，时钟为 T0 输入时钟
1	T0M1	[T0M1,T0M0]为定时器 0 模式选择位 00：模式 0，TL0 和 TH0 组成 13 位定时器/计数器 01：模式 1，TL0 和 TH0 组成 16 位定时器/计数器 10：模式 2，TL0 作为 8 位定时器/计数器，TH0 作为自动重载寄存器 11：模式 3，TL0 和 TH0 作为两个完全独立的 8 位定时器/计数器
0	T0M0	

表 12-1-2-3 寄存器 TL0

8AH	7	6	5	4	3	2	1	0
TL0	TL0							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	TL0	定时器 0 模式 0/1 计数值的低字节，模式 2/3 计数值

表 12-1-2-4 寄存器 TH0

8CH	7	6	5	4	3	2	1	0
TH0	TH0							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	TH0	定时器 0 模式 0/1 计数值的高字节，模式 2 重载值，模式 3 计数值

12.2 定时器 1

12.2.1 定时器 1 介绍

定时器或计数器功能通过 CT1 位（TMOD[6]）来选择，CT1=0 选择为定时器，CT1=1 选择为计数器。作为定时器时，时钟是系统时钟的 12 分频。作为计数器时，时钟是 T1 的输入时钟。由于检测 T1 输入边沿变化需要 2 个时钟周期，所以作为计数器时最大的输入波特率是内部系统时钟频率的 1/2。T1 输入信号在占空比上没有限制，然而为了完全识别 0 或 1 的状态，信号至少需要保持 1 个内部系统时钟周期时间。定时器 1 有 4 个工作模式，通过 T1M0、T1M1 位(TMOD[5:4])来选择。

- **模式 0**

  在此模式下，定时器 1 作为 13 位定时器/计数器，TH1 存放 13 位定时器/计数器的高 8 位，TL1[4:0]存放低 5 位，而 TL1[7:5]是无效的，在读取时应被忽略。当定时器 1 溢出，中断标志位 TF1（TCON[7]）会被置 1。中断被响应后，TF1 位会自动清 0。当 GATE1（TCON[7]）=0 时，定时器/计数器由 TR1（TCON[6]）位使能计数，当 GATE1=1 时，定时器/计数器由引脚 INT1 控制使能，INT1 为高电平时计数，INT1 为低电平则停止计数。

- **模式 1**

  在此模式下，定时器 1 作为 16 位定时器/计数器，TH1 存放 16 位定时器/计数器的高 8 位，TL1 存放低 8 位。当定时器 1 溢出，中断标志位 TF1（TCON[7]）会被置 1。中断被响应后，TF1 位会自动清 0。当 GATE1（TCON[7]）=0 时，定时器/计数器由 TR1（TCON[6]）位使能计数，当 GATE1=1 时，定时器/计数器由引脚 INT1 控制使能，INT1 为高电平时计数，INT1 为低电平则停止计数。

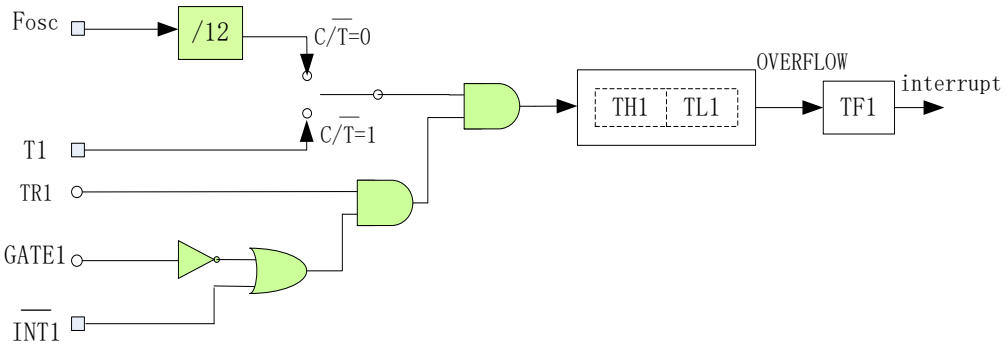


图 12-2-1 定时器 1 的模式 0 和 1

- **模式 2**

  在此模式中，定时器 1 作为 8 位自动重载定时器/计数器，只有 TL1 自动累加。当 TL1 计数溢出时，不但产生中断标志 TF1，而且从 TH1 中自动装载计数初始值到 TL1。其他设置方法和模式 0、1 相同。

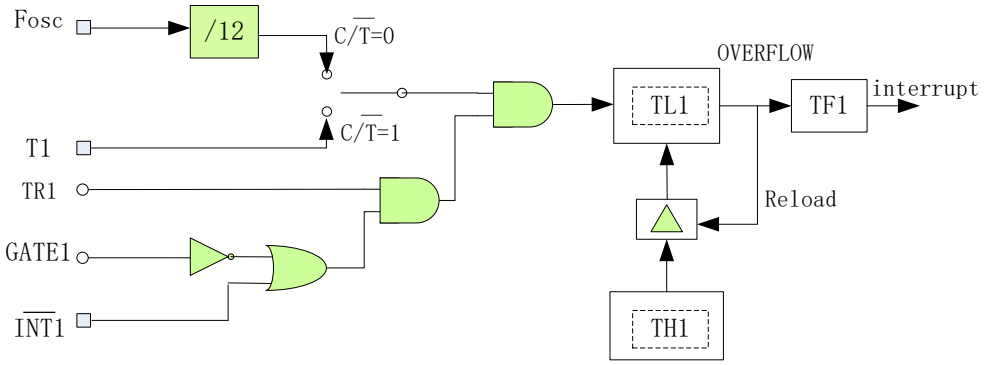


图 12-2-2 定时器 1 的模式 2

- **模式 3**
  此模式下，TH1、TL1 会被锁住，等效于 TR1=0。

12.2.2 定时器 1 寄存器描述

寄存器 TCON 和 TMOD 见表 12-2-2-1 和表 12-2-2-2。

表 12-2-2-1 寄存器 TL1

8BH	7	6	5	4	3	2	1	0
TL1	TL1							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7~0	TL1	定时器 1 模式 0/1 计数值的低字节，模式 2/3 计数值						

表 12-2-2-2 寄存器 TH1

8DH	7	6	5	4	3	2	1	0
TH1	TH1							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7~0	TH1	定时器 1 模式 0/1 计数值的高字节，模式 2 重载值，模式 3 计数值						

13 看门狗定时器（WDT）

13.1 看门狗定时器(WDT)功能简介

看门狗定时器是一个可选时钟源的 27 位减法计数器，时钟为 16MHz 下计数时间范围为 0.128ms – 4.096s，有 16 位调节精度。看门狗主要用于监控系统，避免 CPU 因为外界干扰出现死机。如果软件不能在溢出前刷新看门狗定时器，看门狗将产生内部复位或者中断。写 A5H 到寄存器 WDFLG 将刷新看门狗，读 WDFLG 可得到看门狗状态。在 STOP 模式下，如果看门狗处于使能状态，则看门狗所选的时钟源正常工作，此时如果看门狗设为中断，看门狗中断可唤醒 CPU。

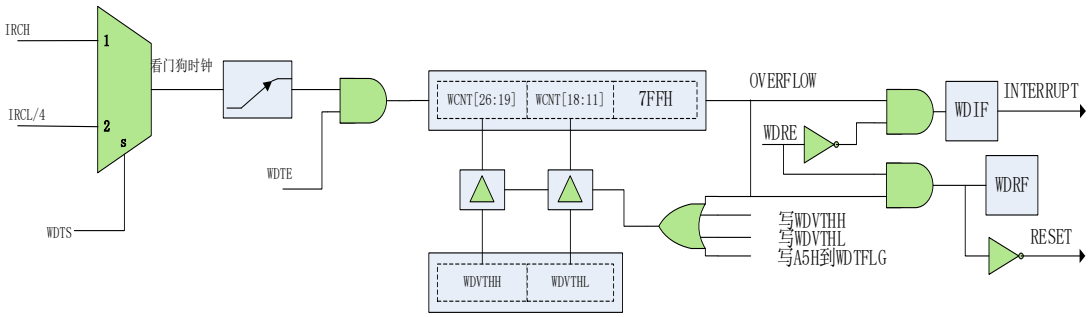


图 13-1-1 看门狗模块结构图

13.2 看门狗定时器(WDT)寄存器描述

表 13-2-1 寄存器 WDCON

A0H	7	6	5	4	3	2	1	0
WDCON	WDTS[1:0]			-	-	-	-	WDRE
R/W	R/W			-	-	-	-	R/W
初始值	0	0		-	-	-	-	0

位编号	位符号	说明
7~6	WDTS	WDT 时钟选择位 01：选择 IRCH 10：选择 IRCL 四分频 其他：WDT 关闭
5~1	-	
0	WDRE	WDT 功能选择位 0：WDT 溢出后产生中断 1：WDT 溢出后产生复位

表 13-2-2 寄存器 WDFLG

A1H	7	6	5	4	3	2	1	0
WDFLG							WDIF	WDRF
R/W	-						R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~2	-	-
1	WDIF	WDT 中断标志，写 A5H 时将清除该标志
0	WDRF	WDT 复位标志，写 A5H 时将清除该标志

表 13-2-3 寄存器 WDVTHL、WDVTHH

A2H	7	6	5	4	3	2	1	0
WDVTHL	WDVTH[7:0]							
R/W	R/W							
初始值	1	1	1	1	1	1	1	1
A3H	7	6	5	4	3	2	1	0
WDVTHH	WDVTH[15:8]							
R/W	R/W							
初始值	1	1	1	1	1	1	1	1

位编号	位符号	说明
15~0	WDVTH	WDT 阈值设置寄存器，计算公式如下： WDT 触发时间＝ (WDVTH * 800H+7FFH) * clock cycle

13.3 看门狗定时器控制例程

◆ **看门狗中断模式例程**

例如，看门狗时钟设置为 IRCH，IRCH 的频率为 16MHz，看门狗设置为中断模式，溢出时间为 1 秒，程序如下：

```
---------------------------------------------------------------------------------
#define WDTS_IRCH           (1<<6)
#define WDTS_IRCL           (2<<6)

#define WDRE_reset          (1<<0)
#define WDRE_int            (0<<0)
void WDT_init(void)
{
    WDCON = WDTS_IRCH | WDRE_int;     //设置看门时钟为 IRCH，看门狗中断模式
    WDVTHH = 0x1E;                    //设置看门狗时间为 1 秒
    WDVTHL = 0x83;
    WDFLG = 0xA5;                     //刷新看门狗
    INT4EN = 1;                       //开启看门狗中断
    EA = 1;                           //开启总中断
}
void WDT_ISR (void) interrupt 6
{
    if(WDFLG & 0x02)
    {
        //看门狗中断服务程序
        WDFLG = 0xA5;          //刷新看门狗
    }
}
---------------------------------------------------------------------------------
```

◆ **看门狗复位模式例程**

例如，看门狗时钟设置为 IRCH，IRCH 的频率为 16MHz，看门狗设置为复位模式，溢出时间为 1 秒，程序如下：

```
---------------------------------------------------------------------------------
#define WDTS_IRCH           (1<<6)
#define WDTS_IRCL           (2<<6)

#define WDRE_reset          (1<<0)
#define WDRE_int            (0<<0)
void WDT_init(void)
{
    WDCON = WDTS_IRCH | WDRE_reset;          //设置看门时钟为 IRCH，看门狗复位模式
    WDVTHH = 0x1e;                           //设置看门狗时间为 1 秒
```

```
    WDVTHL = 0x83;
    WDFLG = 0xA5;                                   //刷新看门狗
}
--------------------------------------------------------------------------------
```

14 TMC 定时器

14.1 TMC 功能简介

TMC 定时器的时钟源为 IRCL，中断的最小单位为 512 个 IRCL 时钟周期，可配置中断时间为 1~256 个最小单位时间。在 STOP/IDLE 模式下，TMC 中断可唤醒 CPU。

14.2 TMC 寄存器描述

表 14-2-1 寄存器 TMCON

D5H	7	6	5	4	3	2	1	0
TMCON	TME	-	-	-	-	-	-	TMF
R/W	R/W	-	-	-	-	-	-	R
初始值	0	-	-	-	-	-	-	0
位编号	位符号	说明						
7	RTCE	TME 模块使能，1 有效						
6~1	-	-						
0	TMF	TMC 中断标志，1 有效，写 1 清 0						

表 14-2-2 寄存器 TMSNU

D6H	7	6	5	4	3	2	1	0
TMSNU	TMSNU[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	-
位编号	位符号	说明						
7~0	TMSNU	TMC 中断时间配置寄存器，TMC 的中断时间为（TMSNU+1）×512×Tircl *注：Tircl 为 IRCL 时钟一个周期时间。*						

14.3 TMC 控制例程

设置 TMC 为最小单位时间中断（即 512 个 IRCL 时钟周期），程序如下：

```
--------------------------------------------------------------------------------
#define TME(N)          (N<<7)    //N=0-1
#define TMF             (1<<0)

#define IHCKE           (1<<7)
#define ILCKE           (1<<6)

void INT3_ISR (void) interrupt 5
{
    if(TMCON & TMF)             //判断 TMC 中断标志
    {
        TMCON |= TMF;           //清除 TMC 中断标志

    }
}

void TMC_init(void)
{
    CKCON |= ILCKE;    //打开 IRCL 时钟
    TMCON = TME(1);  //TMC 使能
    TMSNU = 0;         //设置 1 个最小单位时间（即 512 个 IRCL 时钟周期）产生中断
    INT3EN =1;         //开启 TMC 中断
    EA = 1;            //开启总中断
}
--------------------------------------------------------------------------------
```

15 通用输入输出口（GPIO）及复用定义

15.1 功能简介

CA51F1 系列芯片最大封装有 6 个 I/O 引脚，每个引脚都是复用功能引脚，不仅能独立编程为输入/输出口，而且还能设置为其他功能引脚。每个引脚都分配了一个功能设置寄存器 PnxF（分别对应引脚 Pnx，其中 n=0、3，代表 P0、P3，x=0~7，代表 Pn.0~Pn.7），用户可通过寄存器 PnxF 配置引脚的主功能和其他选项。详见寄存器部分介绍。

GPIO 的主要特性如下：

- 可配置为高阻模式
- I/O 结构可独立设置上拉下拉电阻
- 输出模式可选开漏输出或推挽输出
- 数据输出锁存支持读-修改-写
- 支持 2.2~5.5V 宽电压范围

GPIO 推挽模式结构图如图 15-1-1 所示。

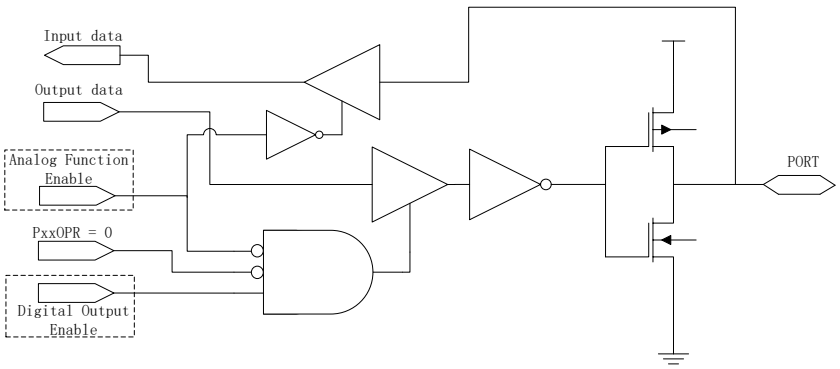


图 15-1-1 I/O 推挽模式结构示意图

GPIO 开漏模式结构图如图 15-1-2 所示。

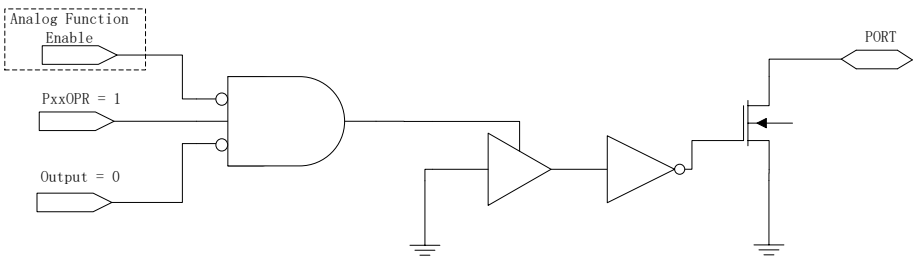


图 15-1-2 I/O 开漏模式结构示意图

GPIO 下拉结构图如图 15-1-3 所示。

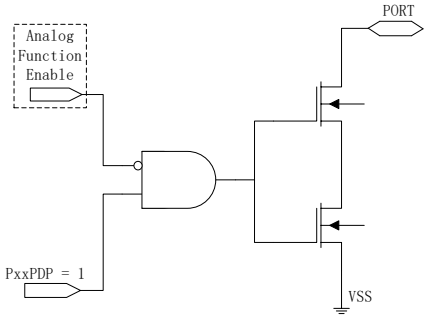


图 15-1-3 I/O 下拉模式结构示意图

GPIO 上拉结构图如图 15-1-4 所示。

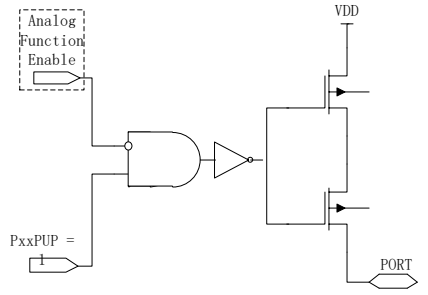


图 15-1-4 I/O 上拉模式结构示意图

15.2 引脚寄存器描述

表 15-2-1 寄存器 P0

80H	7	6	5	4	3	2	1	0
P0	-	-	-	-	-	-	P01	-
R/W	-	-	-	-	-	-	R/W	-
初始值	-	-	-	-	-	-	0	-

位编号	位符号	说明
7~2	-	-
1	P01	引脚 P01 的数据寄存器，管脚功能设置为 GPIO 时有效 0：设为输入时 P01 电平为低，设为输出时 P01 输出低电平

		1：设为输入时 P01 电平为高，设为输出时 P01 输出高电平
0	-	-

表 15-2-2 寄存器 P3

B0H	7	6	5	4	3	2	1	0
P3	-	-	P35	P34	-	P32	P31	P30
R/W	-	-	R/W	R/W	-	R/W	R/W	R/W
初始值	-	-	0	0	-	0	0	0

位编号	位符号	说明
7~6	-	-
5~0	P3x	引脚 P3x 的数据寄存器，管脚功能设置为 GPIO 时有效 0：设为输入时 P3x 电平为低，设为输出时 P3x 输出低电平 1：设为输入时 P3x 电平为高，设为输出时 P3x 输出高电平

表 15-2-3 引脚功能控制寄存器

8001H	7	6	5	4	3	2	1	0
P01F	P01PUP	P01PDP	P01OPR	-	-	P01S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	0	0
8018H	7	6	5	4	3	2	1	0
P30F	P30PUP	P30PDP	P30OPR	-	-	P30S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	1	1
8019H	7	6	5	4	3	2	1	0
P31F	P31PUP	P31PDP	P31OPR	-	-	P31S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	1	1
800DH	7	6	5	4	3	2	1	0
P32F	P32PUP	P32PDP	P32OPR	-	-	P32S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	0	0
800EH	7	6	5	4	3	2	1	0
P34F	P34PUP	P34PDP	P34OPR	-	-	P34S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	0	0
800FH	7	6	5	4	3	2	1	0
P35F	P35PUP	P35PDP	P35OPR	-	-	P35S		
R/W	R/W	R/W	R/W	-	-	R/W		
初始值	0	0	0	-	-	0	1	1

位编号	位符号	说明
7	PnxPUP	上拉电阻使能控制位 0: 上拉电阻关闭 1: 上拉电阻打开 *备注：上拉电阻为30K*
6	PnxPDP	下拉电阻使能控制位 0: 下拉电阻关闭 1: 下拉电阻打开 *备注：下拉电阻为30K*
5	PnxOPR	开漏使能控制位，引脚设为数字输出时才有效 0: 开漏关闭 1: 开漏打开

表 15-2-5 寄存器 PnxC

8121H	7	6	5	4	3	2	1	0
P01C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	1	1
8138H	7	6	5	4	3	2	1	0
P30C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	0	0
8139H	7	6	5	4	3	2	1	0
P31C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	0	0
8126H	7	6	5	4	3	2	1	0
P32C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	0	0
8127H	7	6	5	4	3	2	1	0
P34C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	0	0
8138H	7	6	5	4	3	2	1	0
P35C	-	SMIT_EN	-	-	-	-	DRV	SR
R/W	-	R/W	-	-	-	-	R/W	R/W
初始值	-	1	-	-	-	-	0	0

位编号	位符号	说明
7	-	-
6	SMIT_EN	为 1 输入的 SMIT 使能，为 0 输入是反相器使能
5~2	-	-
1	DRV	输出强度选择 0：弱驱动 1：强驱动
0	SR	输出斜率控制 0：最慢斜率控制 1：最快斜率控制

备注：Pnx → n=0、3，代表 P0、P3
x=0~7，代表 Pn.0~Pn.7

表 15-2-6 引脚复用功能映射表

取值 / 名称	0	1	2	3	4	5	6	7
P01S	高阻	数字输入	数字输出	ADC[4]	TK[3]	T1	PWM2	高阻
P30S	高阻	数字输入	数字输出	ADC[5]	TK[4]	UART_TX	I2C_SDA	高阻
P31S	高阻	数字输入	数字输出	ADC[0]	TK[0]	UART_RX	I2C_SCL	高阻
P32S	高阻	数字输入	数字输出	ADC[2]	TK[1]	高阻	PWM0	RESET
P34S	高阻	数字输入	数字输出	ADC[3]	TK[2]	高阻	PWM1	高阻
P35S	高阻	数字输入	数字输出	ADC[1]	TK_CAP	T0	高阻	高阻

15.3 引脚控制例程

◆ **引脚功能设置**

例如，P01 设置为推挽输出，程序如下：

```
P01F = 2;
```

P01 设置为开漏输出，程序如下：

```
P01F = (1<<5)|2;
```

P01 设置为开漏输出，并且打开上拉，程序如下：

```
P01F = (1<<7)｜(1<<5)｜2;
```

P01 设置为输入功能，并且打开上拉，程序如下：

```
P01F = (1<<7)｜1;
```

16 通用串行接口（UART）

16.1 功能简介

UART 是一个全双工异步串行数据收发器，UART 有一字节的接收缓存。UART 有两种不同的工作模式，如表 16-1-1-1 所示。

SM	模式	描述	波特率
0	A	9 位异步模式	CPUCLK/(32*(1024-SREL))
1	B	8 位异步模式	CPUCLK/(32*(1024-SREL))

表 16-1-1-1 UART 工作模式

UART 设计了专门的波特率发生器，波特率通过寄存器 SRELL、SRELH 来配置。

图 16-1-1-1 是 UART 的原理示意图。

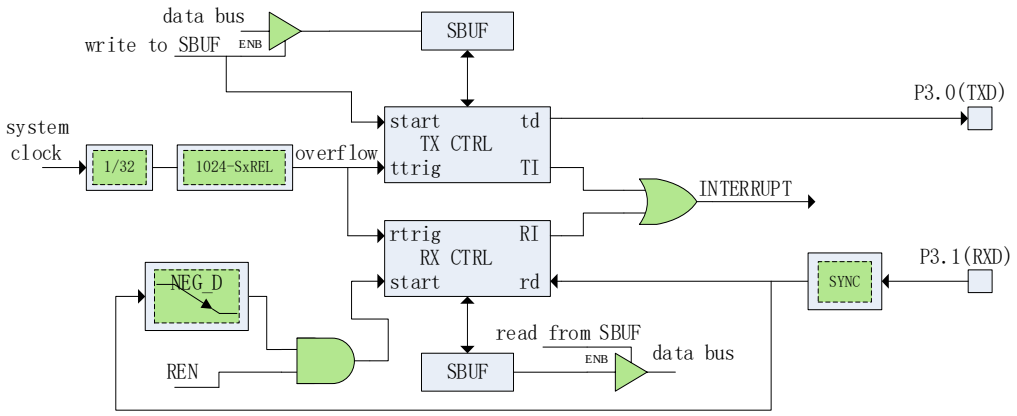


图 16-1-1-1 UART 工作原理示意图

- **模式 A**

在模式 A，UART 可异步同时收发 9 位数据。写入数据到寄存器 SBUF 会启动 UART 数据发送。第一个传送的位是开始位（为 0），然后是 9 位数据（低位先发），第 9 位数据是寄存器 SCON 的 TB81 位，最后传送的是停止位（为 1）。在接收状态，UART 通过检测引脚 RX 的下降沿来同步。传送过程完成后，低 8 位数据存放在寄存器 SBUF，第 9 位数据存放在 RB8 位。

- **模式 B**

模式 B 和模式 A 不同的是，模式 B 是 8 位数据传输，停止位存放的是有效停止位。其他功能和模式 A 一致。

- **UART 多机通信**

在 UART 模式 A 中有一个专门适用于多机通信的机制。当寄存器 SCON 的 SM21 位置 1，只有接收到第 9 位数据为 1（RB8=1）的从机才会产生接收中断，利用这个功能可进行多机通信，从机将它们的 SM21 位都置为 1，主机传送从机的地址时将第 9 位数据设为 1，这样所有的从机都会产生接收中断；从机的软件用它们自己的地址和接收的地址进行比较，如果一致，被寻址的从机设置 SM21=0，然后主机继续传送后面的数据时设置第 9 位为 0，因为其他的从机 SM21 仍然设为 1，这样就只有被寻址的从机才会产生接收中断。

16.2 寄存器描述

表 16-2-1 寄存器 SCON

98H	7	6	5	4	3	2	1	0
SCON	SM	UIE	SM2	REN	TB8	RB8	TI	RI
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	SM	UART 模式选择位，详见表 17-2-1-1
6	UIE	UART 中断使能位，1 有效
5	SM2	多机通信使能位，1 有效
4	REN	串行接收使能位，1 有效
3	TB8	发送数据的第 9 位 在模式 A，这个位用于 UART 传送数据，对应传送数据的第 9 位 （例如奇偶校验或多主机通信），由软件控制
2	RB8	接收数据的第 9 位 在模式 A，这个位用于 UART 接收数据，对应接收数据的第 9 位； 在模式 B，这个位是接收到的停止位
1	TI	传送中断标志位，1 有效，写 1 清 0
0	RI	接收中断标志位，1 有效，写 1 清 0

表 16-2-2 寄存器 SBUF

99H	7	6	5	4	3	2	1	0
SBUF	SBUF[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	SBUF	发送接收缓冲器

		写 SBUF 将启动发送所写的数据 读 SBUF 将读取已经接收的数据

表 16-2-3 寄存器 UDCKS

D8H	7	6	5	4	3	2	1	0
UDCKS	UDE	-	-	DNUM[4:0]				
R/W	R/W	-	-	R/W				
初始值	0	-	-	0	0	0	0	0

位编号	位符号	说明
7	UDE	快速波特率配置使能控制位，1 有效 ***备注：*** *UDE=0 时，UART 波特率按照原来的配置，UDE=1，UART 波特率由 DNUM 来配置。* 快速波特率配置使能控制位，1 有效
6~5	-	-
4~0	DNUM	快速波特率配置寄存器，仅在 UDE=1 时有效 发送时，须满足 DNUM>=0；接收时，DNUM>=6 BR = F_{sys}/((DNUM+1)*(1024-SREL))

表 16-2-4 寄存器 SRELL、SRELH

8068H	7	6	5	4	3	2	1	0
SRELL	SREL[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
8069H	7	6	5	4	3	2	1	0
SRELH	-	-	-	-	-	-	SREL[9:8]	
R/W	-	-	-	-	-	-	R/W	
初始值	-	-	-	-	-	-	0	0

位编号	位符号	说明
9~0	SREL	波特率配置寄存器 UDE=0 时， BR = Fsys/(32 * (1024 – SREL)) UDE=1 时， BR = F_{sys}/((DNUM+1)*(1024-SREL))

17 I²C 接口

17.1 功能简介

I²C 模块支持芯片与外围 I²C 器件以标准 I²C 协议进行串行数据传输，可设置为主机或从机模式，通过合理配置可使 I²C 支持标准/快速/高速模式。

17.2 I²C 主要特点

- 简单且强大而灵活的通讯接口，双向两线总线
- 可设置为主机或从机模式
- 可以工作于发送器模式或接收器模式
- 7 位从机地址
- 支持多主机仲裁
- 支持广播功能

17.3 I²C 功能描述

I²C 模块支持 I²C 标准总线协议。I²C 总线用 2 根线在设备间传输数据，分别为 SCL（串行时钟线）和 SDA（串行数据线），如图 19-3-1 所示。由于 I²C 端口是开漏结构，所以 I²C 总线上必须有上拉电阻，上拉电阻可以外接也可以在芯片内部打开。每个连接在总线上的设备都有一个唯一的 7 位地址。

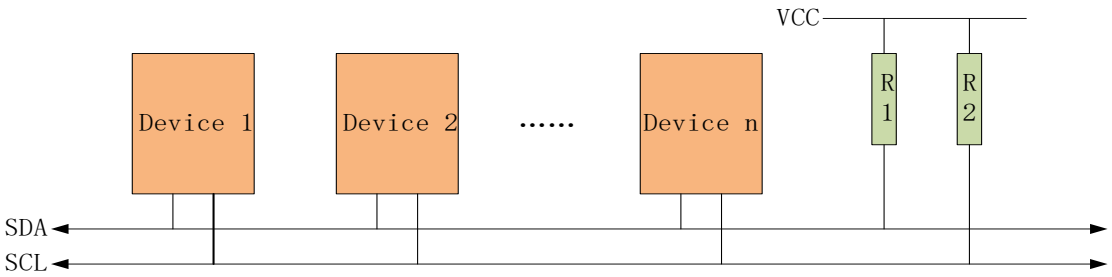


图 17-3-1 I2C 总线互连图

I²C 模块原理示意图如图图 17-3-2 所示。

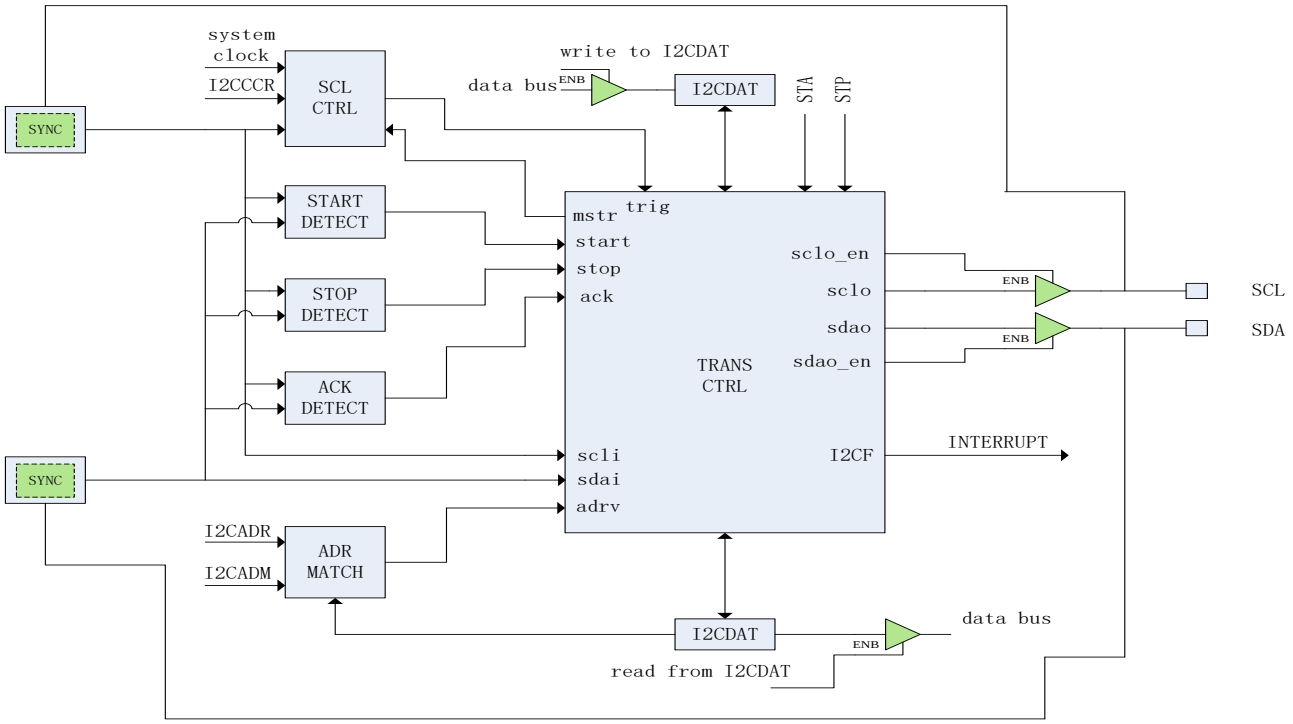


图 17-3-2　I²C 模块原理示意图

- **I²C 模式选择**

I²C 可以在以下 4 种模式中的一种运行：从机发送模式、从机接收模式、主机发送模式、主机接收模式。默认情况下，I²C 处于从机模式。I²C 在产生开始信号后自动从从机模式切换到主机模式，当仲裁失败或产生 STOP 信号后又自动切回从机模式。

- **I²C 总线数据传输格式**

一般情况下，标准的 I²C 通信由四部分组成：开始信号、从机地址传输、数据传输和结束信号。I²C 总线上传送的数据均为 8 位，高位先发，每发送一个字节后都必须跟随一个应答位，每次通信的数据字节数没有限制；在全部数据传送结束后，由主机发送停止信号，结束通信。

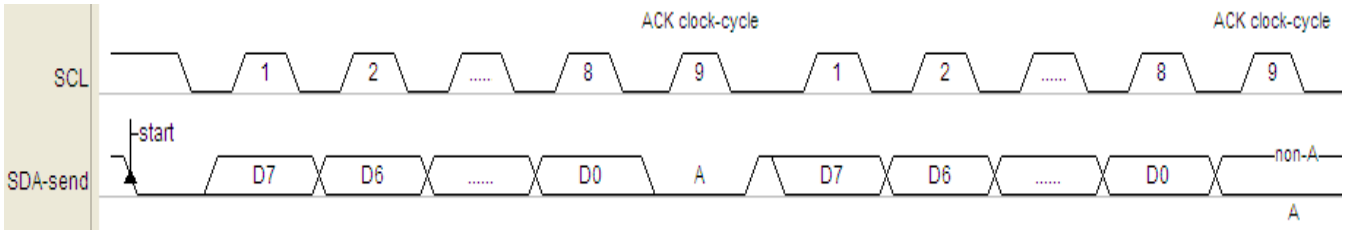


图 17-1-3　I2C 总线数据传输格式

- **通信过程**

在主机模式下，I²C 接口启动数据传输并产生时钟信号。串行数据传输总是以 START 信号开始，以 STOP 信号结束。START 信号和 STOP 信号都是在主机模式下通过软件控制产生的，START 信号通过设置 STA=1 产生，而 STOP 信号通过设置 STP=1 产生。

在从机模式下，I²C 接口能识别自身地址（7 位地址）和广播地址。软件能通过 GCE 位使能或禁止广播地址的识别。

地址和数据以字节为单位进行传输，地址会跟在 START 信号之后由主机发送。在一个字节传输的 8 个时钟

后的第 9 个时钟周期内，接收器必须回送一个应答位给发送器。应答位通过 AAK 位设置，设置应答位必须在一个字节传输完之前设置，接收器完成一个字节接收时，应答信号自动产生。数据传输过程中，数据发送/接收完一字节、仲裁失败等事件都会产生中断标志 I2CF，而事件的状态则由寄存器 I2CSTA 指示（详细请参考寄存器 I2CSTA 介绍），软件应在产生中断标志后根据事件的状态设置数据传输的下一步操作，清除中断标志 I2CF 将启动下一步操作。通信结束后主机产生 STOP 信号也会在从机端产生中断标志 I2CSTP，指示通信过程的完成。当中断标志 I2CF 产生时，如果 SHD=1，在没有清除 I2CF 之前，SCL 会被从机拉低，主机检测到 SCL 被释放后才会进行下一步操作；如果 SHD=0，从机不会拉低 SCL，这样设计是为了兼容主机是软件模拟 I2C 的应用，此时，主机的软件必须等待足够长的时间让从机响应每字节数据传输的处理。

当 I²C 接口作为从机时，SCL 的时钟由主机输入，和从机的时钟配置无关。作为从机时，需要保证 SCL 为低电平的宽度最少为 6.5 个系统时钟，而高电平最少为 2.5 个系统时钟。所以，外部主机发送的 SCL 频率最高为系统时钟频率的 1/9。

17.4 I²C 通信引脚的映射

为了方便硬件设计，I²C 通信引脚可以有不同的映射，由寄存器 I2CIOS 设置不同的值来选择。详细见寄存器 I2CIOS 的描述。

17.5 寄存器描述

表 17-5-1 寄存器 I2CCON

C0H	7	6	5	4	3	2	1	0
I2CCON	I2CE	I2CIE	STA	STP	SHD	AAK	CBSE	STFE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	1	0	0	0	0	1	0	0

位编号	位符号	说明
7	I2CE	I²C 模块使能位，1 有效
6	I2CIE	I²C 中断使能位，1 有效
5	STA	I²C 发送 START 信号控制位，1 有效，检测到 START 信号后将自动清 0
4	STP	I²C 发送 STOP 信号控制位，1 有效，检测到 STOP 信号后将自动清 0
3	SHD	为 1 时，如果 I2CF 为 1，那么当 SCL 变低之后，I2CF 将会使 SCL 保持在低的状态
2	AAK	I²C 发送 ACK 信号控制位，1 有效 *备注：* *当 I²C 接口配置为从机模式时，这一位须预先置 1，否则即使地址匹配也不会回复 ACK，从而无法被寻址。*
1	CBSE	CBUS 兼容使能位 当这一位设置为 1 时，将会使传输忽略 ACK 位的状态判断，以兼容 CBUS 总线。
0	STFE	为 1 时，I²C 模块检测到 START 信号时将置位 I2CF

表 17-5-2 寄存器 I2CADR

C1H	7	6	5	4	3	2	1	0
I2CADR	GCE	I2CADRL[6:0]						
R/W	R/W	R/W						
初始值	1	0	0	0	0	0	0	0

位编号	位符号	说明
7	GCE	识别广播地址（00H）使能位，1 有效
6~0	I2CADRL	I²C 从机地址，作为从机时有效 *备注：* *（在 AAK 为 1 的前提下）7 位地址模式时，接收的第一个地址字节高 7 位和 I2CADR 匹配，则回复 ACK，进入从机模式。*

表 17-5-3 寄存器 I2CADM

C2H	7	6	5	4	3	2	1	0
I2CADM	SPFE	I2CADML[6:0]						
R/W	R/W	R/W						
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	SPFE	为 1 时，I²C 模块检测到 STOP 信号时将置位 I2CF
6~0	I2CADML	I²C 从地址按位屏蔽寄存器，为从机时有效 当 I2CADM[n](n=0~6)=1 时，对应的地址位 I2CADR[n]将不比对（即认为无论收到 1 还是 0 都算匹配）。

表 17-5-4 寄存器 I2CCR

C3H	7	6	5	4	3	2	1	0
I2CCR	I2CCR[7:0]							
R/W	R/W							
初始值	0	0	1	0	0	0	0	0

位编号	位符号	说明
7~0	I2CCR	I²C 时钟配置寄存器 采样频率为 I²C 工作时钟的 $2^{I2CCR[7:5]}$分频，当 I2CCR[7:5]等于 000：$F_{sample}=F_{i2cclk}$ 001：$F_{sample}=F_{i2cclk}/2$ 010：$F_{sample}=F_{i2cclk}/4$ … 111：$F_{sample}=F_{i2cclk}/128$

		输出频率为采样频率的(I2CCCR[4:0]+1)分频， $F_{scl}=F_{i2cclk}/(2^{I2CCCR[7:5]}*(I2CCCR[4:0]+1))$ 例如 I2CCCR[4:0]=9 时，当 I2CCR[7:5]等于 000：$F_{scl}=F_{i2cclk}/(1*10)$ 001：$F_{scl}=F_{i2cclk}/(2*10)$ 010：$F_{scl}=F_{i2cclk}/(4*10)$ … 111：$F_{scl}=F_{i2cclk}/(128*10)$ ***备注：*** *1. 当 I2CCCR[7:5] = 0 时，如果对 I2CCR[4:0]写小于 9 的值，将自动按 9 的值计算。* *2. 当 I2CCCR[7:5] > 0 时，如果对 I2CCR[4:0]写小于 7 的值，将自动按 7 的值计算。*

表 17-5-5 寄存器 I2CDAT

C4H	7	6	5	4	3	2	1	0
I2CDAT	I2CDAT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7~0	I2CDAT	发送和接收数据缓存 *备注：* *当 I2CF 为 1 时，建议改写/读取 I2CDAT 时，让 I2CF 保持在 1，等处理完成之后再清除 I2CF，以继续传输，这样可以避免总线发生不必要的错误。*

表 17-5-6 寄存器 I2CSTA

C5H	7	6	5	4	3	2	1	0
I2CSTA	I2CSTA[7:0]							
R/W	R							
初始值	1	1	1	1	1	0	0	0

位编号	位符号	说明
7~0	I2CSTA	I^2C 状态寄存器 00H: （主/从）总线错误 08H: （主/从）检测到 START 信号（只在 STFE=1 时才有效） 18H: （主）已发送地址+写位，已接收到应答信号 20H: （主）已发送地址+写位，无接收到应答信号 28H: （主）已发送/接收一字节数据，已检测到应答信号 30H: （主）已发送/接收一字节数据，无检测到应答信号

		38H: （主）失去仲裁（主机失去仲裁后会变为从机） 40H: （主）已发送地址+读位，已接收到应答信号 48H: （主）已发送地址+读位，无接收到应答信号 60H: （从）已接收地址+写位，已发送出应答信号 70H: （主/从）已接收广播地址，已发送出应答信号（主机或从机都会变为从机） 80H: （从）已发送/接收一字节数据，已检测到应答信号 88H: （从）已发送/接收一字节数据，无检测到应答信号 A0H: （主/从）检测到 STOP 信号（只在 SPFE=1 时才有效） A8H: （从）已接收地址+读位，已发送出应答信号 F8H: （主/从）总线空闲

表 17-5-7 寄存器 I2CFLG

C6H	7	6	5	4	3	2	1	0
I2CFLG	-	-	-	-	-	-	-	I2CF
R/W	-	-	-	-	-	-	-	R
初始值	-	-	-	-	-	-	-	0

位编号	位符号	说明
7~1	-	-
0	I2CF	I^2C 中断标志，1 有效，写 1 清 0 *备注：* *1. 每字节地址或数据传输完成后（收到/发送完 ACK/NAK），将置位 I2CF。* *2. 总线出错时，将置位 I2CF。* *3. 当 STFE=0 时，检测到 START 信号，I2CF 不会置 1。* *4. 当 SPFE=0 时，检测到 STOP 信号，I2CF 不会置 1。*

18 PWM

18.1 PWM 功能描述

CA51F1 系列芯片有 3 通道 PWM 输出，PWM 周期和占空比可在 16 位范围内任意配置。

每路 PWM 通道都有一个专门的 16 位计数器，PWM 的周期通过寄存器 PWMnDIV 来设置，而寄存器 PWMnDUT 则对应 PWM 的占空比。PWM 通过寄存器 PWMEN 使能，寄存器 PWMEN 的每一位对应 PWM 的一个通道。PWM 可通过 PWMnTOG 位设置 PWM 引脚输出反相。PWM 有多种时钟源可以选择，每路时钟源都是单独进行设置的，对应的控制寄存器为 PWMnCON 的 PWMnCKS。另外，每路 PWM 的时钟分频可通过 PWMnCKD 独立设置。

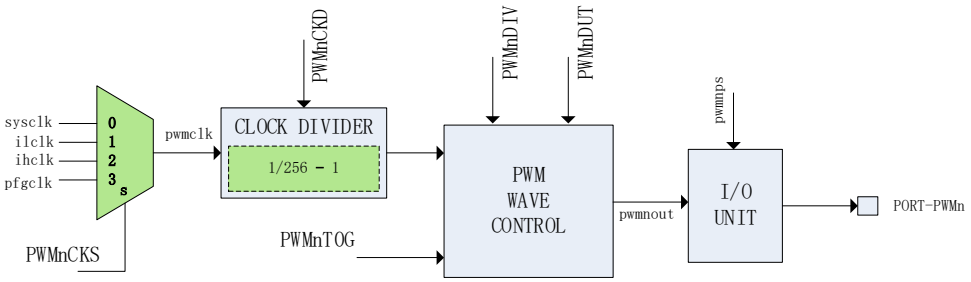


图 18-1-1 PWM 原理示意图

备注：

PWMnDIV，PWMnDUT 等带“n”的寄存器名称，其中“n”表示 0/1/2，分别代表 PWM 通道 0/1/2 这 3 个通道的控制或者配置寄存器。

- **PWM 输出波形**

PWM 使能后，PWM 计数器开始累加计数，当计数值不大于 PWMnDUT 时，PWM 引脚输出高电平（PWMnTOG=0），当计数值大于 PWMnDUT 时，PWM 引脚输出低电平（PWMnTOG=0）。当计数值与 PWMnDIV 相等时，一个 PWM 周期完成，PWM 计数器重置并开始下一周期计数，此时将产生 PWM 中断。

当 PWM 波形满足条件 PWMnDIV>PWMnDUT>0 时，PWM 波形如图所示。

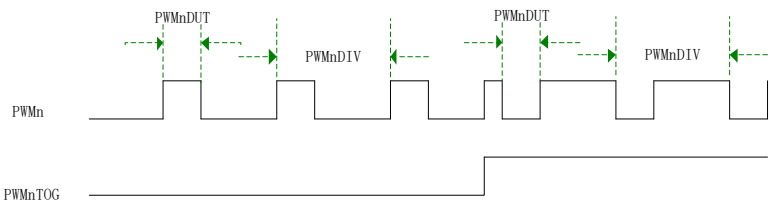


图 18-1-2 PWM 输出波形

值得注意的是，当 PWMnDIV=0 时，PWM 引脚直接输出 PWM 时钟，如果 PWMnCKD=0，PWM 引脚输出的是所选的时钟源的时钟信号；如果 PWMnCKD！=0，PWM 引脚输出的是所选的时钟源的 1/（PWMnCKD+1）频率的时钟信号；当 PWMnDIV 不为 0，而 PWMnDUT=0 时，PWM 引脚输出低/高电平（PWMnTOG＝0/1）；当 PWMnDUT>=PWMnDIV>0 时，PWM 引脚输出高/低电平（PWMnTOG＝0/1）。

- **单线级联 LED 驱动**

PWM1 通道支持单线级联 LED 驱动，级联 LED 的典型驱动时序图如图 18-1-3 所示。

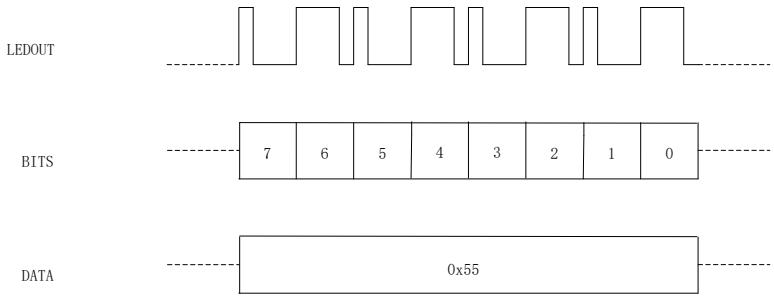


图 18-1-3 级联 LED 时序图

位码示意图如图 18-1-4 所示。

图 18-1-4 位码示意图

在级联 LED 时序图中，位码 0 的高电平时间宽度由 PWM1DUT 配置，位码 1 的高电平时间宽度由 LEDUTH 配置，而位周期时间由 PWM1DIV 配置。当 PWM1MOD 不为 0 时，级联 LED 驱动使能，LEDAT 为 LED 的数据寄存器，当 LEF 为 0 时，可以向 LEDAT 写入 LED 数据。写入 LEDAT 即启动 LED 驱动数据发送，当 LED 发送器正处于发送状态时，LEBSY 置 1，当发送器处于空闲状态时，LEBSY 变为 0。LED 发送器有一字节的发送缓存，当数据寄存器和缓存寄存器都有数据时，LEF 位置 1，当缓存寄存器的数据发送完后，会自动从数据寄存器中加载，同时 LEF 位置 0，LEF=0 表示可以重新向 LEDAT 装载数据。当 PWM1MOD 不为 0 时，PWM1MOD 也同时表示发送完 PWM1MOD 个字节后插入等待时间，等待时间由 LEWTM 来配置。

当 PWM1POL 时，LEDAT 的数据反相，即：例如写入 01010101B，实际发送出来的是 10101010B。

18.2 PWM 寄存器描述

表 18-2-1 寄存器 PWMEN

C7H	7	6	5	4	3	2	1	0
PWMEN	-	-	-	-	-	PWM2EN	PWM1EN	PWM0EN
R/W	-	-	-	-	-	R/W	R/W	R/W
初始值	-	-	-	-	-	0	0	0

位编号	位符号	说明
7~3	-	-
2	PWM2EN	PWM2 使能控制位，1 有效
1	PWM1EN	PWM1 使能控制位，1 有效
0	PWM0EN	PWM0 使能控制位，1 有效

表 18-2-2 寄存器 PWMCON

B9H	7	6	5	4	3	2	1	0
PWM0CON	-	PWM0TOG	-	-	-	-	PWM0CKS[1:0]	
R/W	-	R/W	-	-	-	-	R/W	
初始值	-	0	-	-	-	-	0	0

BAH	7	6	5	4	3	2	1	0

PWM1CON	-	PWM1TOG	PWM1MOD[2:0]			PWM1POL	PWM1CKS[1:0]	
R/W	-	R/W	R/W			R/W	R/W	
初始值	-	0	0	0	0	0	0	0
BBH	7	6	5	4	3	2	1	0
PWM2CON	-	PWM2TOG	-	-	-	-	PWM2CKS[1:0]	
R/W	-	R/W	-	-	-	-	R/W	
初始值	-	0	-	-	-	-	0	0

备注：以下 n=0，1，2.

位编号	位符号	说明
7	-	-
6	PWMnTOG	PWMn 输出取反使能控制位， 1 有效
5~3	PWM1MOD	PWM1 作为 LED 驱动时，连续发送字节数配置寄存器，0 表示 PWM1 不作为 LED 驱动使用，1~7 表示 PWM1 每发送 1~7 字节数据就暂停 1 次 ***备注：*** 1. *只有 PWM1CON 才有此配置项。* *2. 详细使用参考 LEWTM。*
2	PWM1POL	PWM1 作为 LED 驱动时，发送数据取反使能控制位，1 有效 ***备注：*** ***1.****只有 PWM1CON 才有此配置项。* *2. 当 PWM1MOD！=0 时，对应的 PWM1POL 的值才有意义；* *3. 当 PWM1POL=1 时，如果对应的 LEDAT=01010101B，那么实际上发送的将会是 10101010B。*
1~0	PWMnCKS	PWM 时钟源选择位 00：系统时钟 01：IRCL 10：IRCH

表 18-2-3 寄存器 PWMCKD

B1H	7	6	5	4	3	2	1	0
PWM0CKD	PWM0CKD[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
B2H	7	6	5	4	3	2	1	0
PWM1CKD	PWM1CKD[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
B3H	7	6	5	4	3	2	1	0

PWM2CKD	PWM2CKD[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
备注：以下 n=0，1，2.								
位编号	位符号	说明						
7~0	PWMnCKD	PWM 工作时钟预分频配置寄存器 00H：不分频 01H：2 分频 02H：3 分频 …… FEH：255 分频 FFH：256 分频						

表 18-2-4 寄存器 PWMDIVL、PWMDIVH

A9H	7	6	5	4	3	2	1	0
PWM0DIVL	PWM0DIV[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
AAH	7	6	5	4	3	2	1	0
PWM0DIVH	PWM0DIV[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
ABH	7	6	5	4	3	2	1	0
PWM1DIVL	PWM1DIV[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
ACH	7	6	5	4	3	2	1	0
PWM1DIVH	PWM1DIV[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
ADH	7	6	5	4	3	2	1	0
PWM2DIVL	PWM2DIV[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
AEH	7	6	5	4	3	2	1	0
PWM2DIVH	PWM2DIV[15:8]							
R/W	R/W							

初始值	0	0	0	0	0	0	0	0
备注：以下 n=0，1，2.								
位编号	位符号	说明						
15~0	PWMnDIV	PWMn 周期配置寄存器						

表 18-2-5 寄存器 PWMDUTL、PWMDUTH

9BH	7	6	5	4	3	2	1	0
PWM0DUTL	PWM0DUT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
9CH	7	6	5	4	3	2	1	0
PWM0DUTH	PWM0DUT[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
9DH	7	6	5	4	3	2	1	0
PWM1DUTL	PWM1DUT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
9EH	7	6	5	4	3	2	1	0
PWM1DUTH	PWM1DUT[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
9FH	7	6	5	4	3	2	1	0
PWM2DUTL	PWM2DUT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
91H	7	6	5	4	3	2	1	0
PWM2DUTH	PWM2DUT[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
15~0	PWMnDUT	PWMn 占空比配置寄存器						

表 18-2-6 寄存器 LEDAT

D7H	7	6	5	4	3	2	1	0
LEDAT	LEDAT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7~0	LEDAT	LED 驱动数据 ***备注：*** *LEDAT 的数据按照从 MSB 到 LSB 的顺序发送。*						

表 18-2-7 寄存器 LEDUTL、LEDUTH

8060H	7	6	5	4	3	2	1	0
LEDUTL	LEDUT[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
8061H	7	6	5	4	3	2	1	0
LEDUTH	LEDUT[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
15~0	LEDUT	LED 发送数据“1”占空比配置寄存器 ***备注：*** *1. 级联 LED 的驱动波形中，每 1 位数据的周期都由对应的 PWM1DIV 决定，而数据“1”的占空比由 LEDUT 决定，数据“0”的占空比由 PWM1DUT 决定；* *2. 如果 LEDAT=01010101B，同时对应的 PWMPOL=1，那么实际发送的数据按照 BIT7-BIT6-BIT5-BIT4-BIT3-BIT2-BIT1-BIT0 顺序就是 1-0-1-0-1-0-1-0，而且 BIT7/BIT5/BIT3/BIT1 的占空比由 LEDUT 决定，BIT6/BIT4/BIT2/BIT0 的占空比由对应的 PWMDUT 决定，即 LEDUT 的起效在 PWMPOL 之后；*							

表 18-2-8 寄存器 LEWTML、LEWTMH

CEH	7	6	5	4	3	2	1	0
LEWTML	LEWTM[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

CFH	7	6	5	4	3	2	1	0
LEWTMH	LEWTM[15:8]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
15~0	LEWTM	LED 暂停时间配置寄存器，结合 PWM1MOD 配置寄存器使用 ***备注：*** *每发送 PWM1MOD 字节数据之后，暂停（LEWTM+1）个 PWM 的工作时钟后进入下一次传输。*

表 18-2-9 寄存器 LEFLG

BFH	7	6	5	4	3	2	1	0
LEFLG	-	-	-	-	-	LEBSY	-	-
R/W	-	-	-	-	-	R	-	-
初始值	-	-	-	-	-	0	-	-

位编号	位符号	说明
7~3	-	
2	LEBSY	LEDAT 数据发送忙标志，1 表示此时 LEDAT 的数据缓存中的数据还没有全部发送完成，0 表示全部发送完成
1~0	-	-

18.3 PWM 功能控制例程

◆ PWM 单路输出例程

以 PWM0 为例，PWM 时钟源为 IRCH（IRCH 频率为 16MHz），输出频率为 30K 的时钟，占空比为 30%，程序如下：

```
-------------------------------------------------------------------------------
//PWMxCON
#define TOG(n)              (n<<6)
#define PWM_CKS_IH          (2<<0)
void PWM_init(void)
{
    PWM0CON = TOG(1)|PWM_CKS_IH;   //设置 PWM 时钟，PWM 是否反向输出
    PWM0CKD = 0;            //设置预分频系数，设置为 0 表示不分频
    PWM0DIVH = 0x02;     //设置 DIV 值，16000000/30000=0x215
    PWM0DIVL = 0x15;
    PWM0DUTH = 0x00;   //设置 DUT 值，占空比为 30%
    PWM0DUTL = 0xA0;
    P32F = 6;                  //设置 P32 为 PWM 引脚功能
    PWMEN |= (1<<0);       //PWM0 使能
}
-------------------------------------------------------------------------------
```

◆ 单线级联 LED 驱动灯带控制例程

以 PWM1 为例，驱动 8 级 RGB LED ，使 RGB LED 循环变色，程序如下：

```
-------------------------------------------------------------------------------
//PWMxCON
#define TOG(n)              (n<<6)
#define PWM_CKS_SYS     (0<<0)
#define PWM_CKS_IL          (1<<0)
#define PWM_CKS_IH          (2<<0)

#define PWMMOD(N)           (N<<3)      //N=0-7
#define PWMPOL(N)           (N<<2)      //N=0-1

void PWM_init(void)
{
    PWM1CON = TOG(0)| PWMMOD(3) | PWMPOL(0)   | PWM_CKS_IH;      //设置 IRCH 为 PWM 时钟源，发送 3 个字节后插入暂停时间
    PWM1CKD = 0;                //设置预分频系数，设置为 0 表示不分频
    PWM1DIVH  = 0;              //设置位周期时间
    PWM1DIVL  = 20;
    PWM1DUTH =    0;            //设置位码 0 时间
    PWM1DUTL  = 6;
    LEDUTH= 0;                  //设置位码 1 时间
    LEDUTL = 13;
```

```
    LEDWTMH   = 0;        //设置暂停时间
    LEDWTML   = 50;
    P34F = 6;             //设置 P34 为 PWM 引脚功能
    PWMEN |= (1<<1);      //PWM1  使能
}
void main(void)
{
     PWM_init(void);
     while(1)
     {
          unsigned char i;
          static unsigned char color_index = 0;
          code unsigned char LED_DAT[][3] =
          {
               {0xff,0x00,0x00},
               {0xff,0xff,0x00},
               {0x00,0xff,0x00},
               {0x00,0xff,0xff},
               {0x00,0x00,0xff},
               {0xff,0x00,0xff},
          };
          for(i=0;i<24;i++)
          {
               while(LEFLG & LEF0);
               LEDAT0 = LED_DAT[color_index][i%3];
          }
          color_index++;
          if(color_index>=6)
          color_index=0;
          Delay_ms(500);
     }
}
```

---

19 模/数字转换器（ADC）

19.1 功能简介

模拟/数字转换器是 12 位逐次逼近寄存器型（SAR）ADC，最多提供多达 6 个输入通道。ADC 时钟源是系统时钟，可设置时钟预分频。ADC 有多种参考电压源可选，其中选择内部电压为参考电压时可用于检测芯片供电电压。ADC 选择内部电压为参考电压时有自动校正功能，避免芯片一致性问题。

19.2 主要特性

- 12 位的分辨率
- 最多提供多达 6 个输入通道
- 支持 ADC 中断
- 可设置 ADC 时钟预分频
- 多种参考电压可选：内部参考电压、VDD。
- 支持 VDD 和参考地电压的测量
- 选择内部参考电压时，支持自动数据校正功能
- 输入电压范围：VSS<=VIN<=VDD。

19.3 结构框图

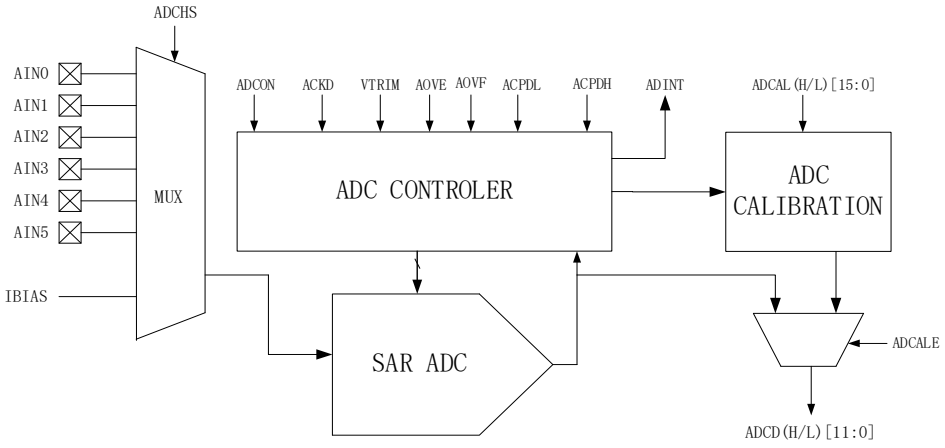


图 20-3-1 ADC 结构示意图

19.4 功能描述

ADC 的启动通过 AST 位使能，设置 AST=1 后，ADC 模块对 ADCHS 选择的输入电压源进行模/数转换。ADC 可通过 ACKD 设置时钟预分频，由系统时钟预分频后的时钟作为 ADC 转换时钟。在 ADC 时钟不变的条件

下，ADC 的单次转换时间是由 HTME 设置的，转换时间为(13+2^HTME)个 ADC 时钟周期。当转换结束后，12 位的 A/D 值会被加载到寄存器 ADCDH、ADCDL，转换完后的 2.5 个时钟周期，AST 位自动清 0，同时中断标志 ADIF 位会置 1，如果 ADC 中断使能，会产生 ADC 中断。图 22-4-1 为 ADC 的转换时序图。

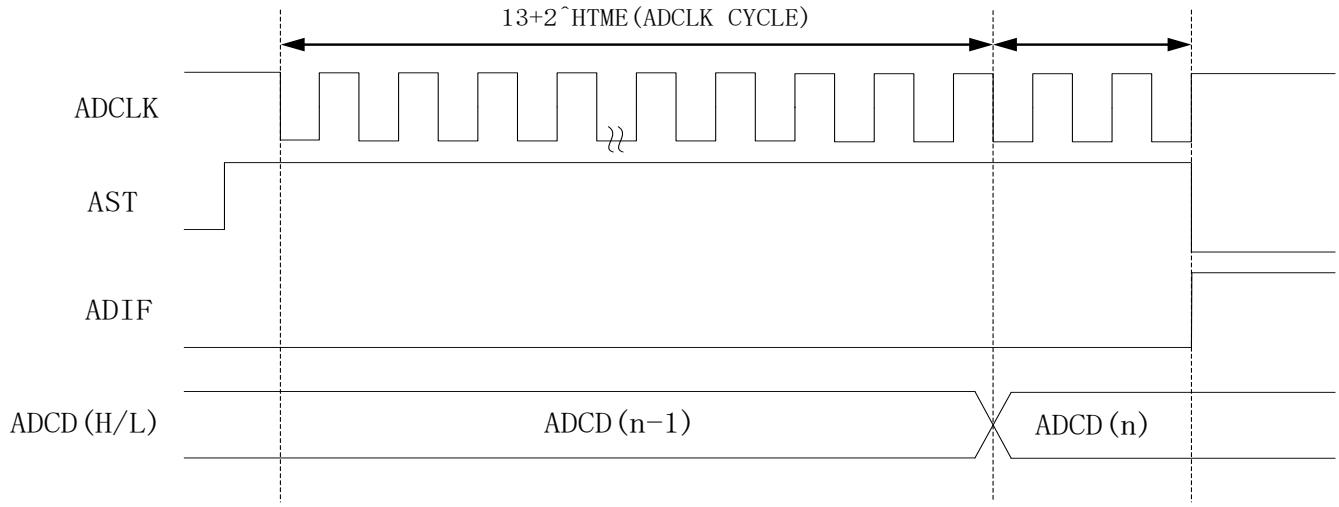


图 20-4-1 ADC 时序示意图

- **ADC 数据校正**

当选择内部 1.5V 作为参考电压时，由于芯片的离散性，每个芯片的内部电压不一定完全相同，导致每个芯片的 ADC 转换结果也有一定的偏差，所以在 ADC 转换完后，有必要对 AD 值进行校正。芯片在出厂时，会对每个芯片的内部电压进行测试，得出与内部电压成反比例的校正值，在芯片上电启动时，自动将此校正值加载到寄存器 ADCALL、ADCALH，当 ADC 转换完成后自动将 AD 值根据校正寄存器 ADCALL、ADCALH 的值进行等比例换算，得出准确的 AD 值，最终的 AD 值也是存放在寄存器 ADCD 中。此功能通过 ADCALE 使能，对于用户来说，在应用时只需要设置 ADCALE=1 即可，校正过程是自动完成的。

19.5 寄存器描述

表 19-5-1 寄存器 ADCON

8060H	7	6	5	4	3	2	1	0
ADCON	AST	ADIE	ADCIF	HTME			-	VSEL
R/W	R/W	R/W	R/W	R/W			-	R/W
初始值	0	0	0	0	1	0	0	0

位编号	位符号	说明
7	AST	ADC 转换开始控制位，写 1 启动转换，转换后硬件自动清 0
6	ADIE	ADC 中断使能位，1 有效
5	ADCIF	ADC 中断标志位，写 1 清 0
4~2	HTME	采样保持周期数为 2 的 HTME 次幂
1	-	-
0	VSEL	ADC 参考电压选择位 0: 内部 1.5V(INNER_VREF)作为参考电压

		1:电源作为参考电压

表 19-5-2 寄存器 ADCFGL

8061H	7	6	5	4	3	2	1	0
ADCFGL	ACKD			ADCALE	ADCHS			
R/W	R/W			R/W	R/W			
初始值	0	0	0	1	0	0	0	0

位编号	位符号	说明
7~5	ACKD	ADC 时钟分频设置 000：不分频 001：2 分频 010：4 分频 … 111：14 分频
4	ADCALE	ADC 校准使能位，1 有效 此位只有选择参考电压为内部 1.5V 时才有效，当 ADCALE=1，ADC 的转换结果将根据 ADCAL 寄存器的数值进行校准。具体参考寄存器 ADCAL 说明。
3~0	ADCHS	ADC 通道使能选择位域 0000：通道关闭 0001：通道 AD_CH[0](P31)使能 0010：通道 AD_CH[1](P35)使能 0011：通道 AD_CH[2](P32)使能 0100：通道 AD_CH[3](P34)使能 0101：通道 AD_CH[4](P01)使能 0110：通道 AD_CH[5](P30)使能 0111：检测 VDD 的 1/4 使能 其他：通道关闭

表 19-5-4 寄存器 ADCAL

8065H	7	6	5	4	3	2	1	0
ADCALL	ADCAL[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
8065H	7	6	5	4	3	2	1	0
ADCALH	ADCAL[15:8]							
R/W	R/W							
初始值	0	0	0	1	0	0	0	0

位编号	位符号	说明

15~0	ADCAL	ADC 校准寄存器，只有 ADCALE=1 并且选择参考电压为内部 1.5V 才有效。有效时，ADC 的输出按照如下公式： ADCDL=（ADC 转换结果*ADCAL）/32768

表 19-5-5 寄存器 ADCD

8062H	7	6	5	4	3	2	1	0
ADCDL	ADCDL[3:0]				-			
R/W	R				-			
初始值	0	0	0	0	-	-	-	-
8063H	7	6	5	4	3	2	1	0
ADCDH	ADCDH[11:4]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
11~0	ADCD	ADC 转换值						

19.6 ADC 控制例程

例如，设置 ADC 参考电压为外部 VDD，采集 ADC 通道 0，程序如下：

```
---------------------------------------------------------------------------------
void main(void)
{
  P31F = P31_ADC0_SETTING;
  ADCON = AST(0) | ADIE(0) | HTME(7) | VSEL(ADC_REF_VDD);       //设置 ADC 参考电压为 VDD
  ADCFGL = ACKD(7) | ADCALE(1) | ADCHS(ADC_CH0);     //选择 ADC0 通道
  while(1)
  {
      unsigned int AD_Value;
      ADCON |= AST(1);                                    //启动 ADC 转换
      while(!(ADCON & ADIF));                             //等待 ADC 转换完成
      ADCON |= ADIF;                                      //清除 ADC 中断标志
      AD_Value = ADCDH*256 + ADCDL;                       //读取 AD 值
      AD_Value >>= 4;
  }
}
---------------------------------------------------------------------------------
```

20 电容式触摸按键（Touch Key）

20.1 功能简介

CA51F1 系列芯片的触摸功能模块具有优越的抗干扰性能，可通过 EFT、CS 等测试。触摸模块最大可支持 5 个通道。针对有低功耗需求的应用，还设计了芯片在 STOP 模式时仍能正常工作和唤醒的机制，以达到产品省电功能。

20.2 主要特性

- 高抗干扰性能，符合 EMC(CS)标准
- 最大支持 5 个通道
- 支持低功耗模式
- 支持触摸中断
- 支持充放电时钟预分频
- 支持手动和自动启动模式
- 比较器阈值有多级可选
- STOP 模式下可设自动唤醒阈值

20.3 结构图

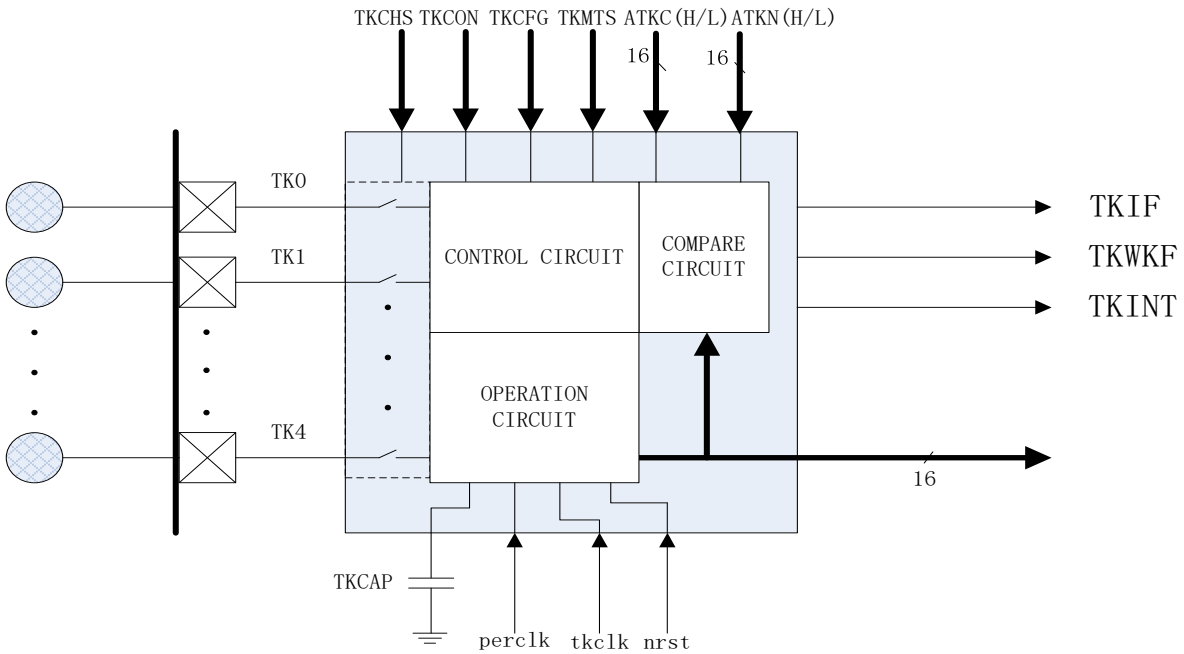


图 20-3-1 触摸模块结构图

20.4 功能描述

20.4.1 触摸通道的使能

触摸模块共支持 5 个外部通道和 1 个内部通道，当触摸外部通道对应的引脚功能寄存器设置为 4 时，触摸通道使能，例如：通道 0 对应的引脚为 P31，当 P31F 设置为 4 时，触摸通道 0 使能。对于内部参考通道，设置 TCAPSEL 为 1 即可使能。

20.4.2 手动模式和自动模式

设置 TMEN=0 选择为手动模式。在手动模式下，触摸数据采集通过 TKST 位启动。当设置 TKST=1 后，触摸模块开始相应触摸通道的数据采集。当数据采集完成后，TKST 位自动清 0，相应通道的中断标志位 TKIF 置 1，采集的触摸数据保存于寄存器 TKnMS（TK0~TK4 对应 TK0MS~TK4MS，内部参考通道对应 TK5MS）。

设置 TMEN=1 选择为自动模式。和手动模式不同的是，自动模式的触摸数据采集是由定时器定时启动的，定时器的时钟源是 IRCL，定时时间由寄存器 TKMTS 设置。

备注：
TKnCHS 等寄存器中的“n”表示 0/1/2/3/4/5。

20.4.3 触摸时钟预分频

触摸控制器对触摸电极充放电的时钟源是 IRCH 的 4 分频或 IRCL，充放电的时钟频率对触摸的性能至关重要，当充放电频率太高时，有可能造成对触摸电极的充电不充分从而导致手指触摸时触摸数据变化量变小。触摸时钟预分频通过 TKDIV 进行设置，通过设置合理的值可以使触摸的性能更优。注意，触摸跳频功能使能后，TKDIV 的设置是无效的。

20.4.4 低功耗模式

为了实现触摸功能的低功耗应用，触摸模块设计了相应的省电机制。在 STOP 模式下，只要触摸的充放电时钟源（IRCH 或 IRCL，由于 IRCH 的静态功耗偏高，触摸省电模式一般选择 IRCL 作为触摸充放电时钟）和低速时钟（IRCL）处于开启状态，触摸模块就可以保持正常的充放电和计数。当触摸采集完成后，触摸采集完成中断会唤醒 CPU，软件在 CPU 唤醒之后可以读取触摸数据，然后再次进入 STOP 模式。

20.4.5 触摸跳频功能

为提升触摸抗电压脉冲注入性能，芯片设计了触摸跳频功能。

触摸跳频功能由 FAEN 位使能，使能后，触摸的充放电时钟在触摸充放电源时钟(即 IRCH 的四分频或 IRCL)的基础上每次充放电按顺序以 2、3、4、5 分频的顺序变化，可有效降低电压脉冲注入时某频段注入干扰信号对触摸的干扰幅度。

20.6 寄存器描述

表 20-6-1 寄存器 TKCON

F8H	7	6	5	4	3	2	1	0
TKCON	TKST	TKIE	TMEN	FAEN	TCAPSEL	VRS[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W		
初始值	0	0	0	0	0	1	0	0

位编号	位符号	说明
7	TKST	数据采集启动使能位，1 有效，采集完后自动清 0
6	TKIE	TK 中断使能控制位，1 有效
5	TMEN	启动方式选择位 0:通过 TKST 位控制启动 1:定时器控制启动
4	FAEN	0：不使能频率调节 1：频率调节使能，触摸时钟每周期切换一次，频率有基准频率的 1-2-3-4 分频轮流切换，每周期切换一次；
3	TCAPSEL	内置参考通道选择位，1 有效
2~0	VRS	比较器阈值电压基准选择位（阈值电压与 VDD 电压成正比例） 0：阈值电压最高 … 7：阈值电压最低

表 20-6-2 寄存器 TKPWC

DFH	7	6	5	4	3	2	1	0
TKPWC	TKPC		VDS		VIRS		TKPWS	TKCVS
R/W	R/W		R/W		R/W		R/W	R/W
初始值	0		0	0	0	0	0	0

位编号	位符号	说明
7~6	TKPC	触摸按键未采样通道输出控制 00：悬浮 01：输出低 10：输出补偿（即防水补偿模式）
5~4	VDS	内部运放输出电压选择 00：2V 01：2.5V 10：3V

		11：4V
3~2	VIRS	内部电压基准选择 00：1.5V 01：2.0V 10：2.5V 11：3.0V
1	TKPWS	充电电源选择 0：选择外部电源 1：选择内部运放输出
0	TKCVS	充电基准电压选择 0：选择外部电压基准 1：选择内部电压基准

表 20-6-3 寄存器 TKCKS

DEH	7	6	5	4	3	2	1	0
TKCKS	-	-	-	-	-	-	-	TKSIL
R/W	-	-	-	-	-	-	-	R/W
初始值	-	-	-	-	-	-	-	0

位编号	位符号	说明
7~1	-	-
0	TKSIL	触摸按键采样时钟选择 0：选择 16M 的四分频(4M)时钟 1：选择慢速(100K)时钟

表 20-6-4 寄存器 TKCFG

F9H	7	6	5	4	3	2	1	0
TKCFG	TKDIV			TKTMS				
R/W	R/W			R/W				
初始值	0	0	0	1	1	1	1	1

位编号	位符号	说明
7~5	TKDIV	触摸时钟分频选择 000: 不分频 001：2 分频 010：3 分频 … 111：8 分频
4~0	TKTMS	外挂调制电容放电时间设置 放电时间 = TKTMS x 128 x 充放电时钟周期 在 TKDIV=0 的条件下，放电时间范围是：32us – 992us

		备注：TKTMS 不能设置为 0。

表 20-6-5 寄存器 TKMTS

FAH	7	6	5	4	3	2	1	0
TKMTS	TKMTS[7:0]							
R/W	R/W							
初始值	0	0	0	0	0	0	0	0
位编号	位符号	说明						
7~0	TKMTS	定时模式的启动时间选择寄存器 启动时间=（TKMTS+1）x 128 x IRCL 时钟周期。因为 IRCL 时钟频率为 100KHz，所以时间范围是 1.28ms–328ms。						

表 20-6-9 寄存器 TKMS

E1H	7	6	5	4	3	2	1	0
TK0MSL	TK0MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E2H	7	6	5	4	3	2	1	0
TK0MSH	TK0MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E3H	7	6	5	4	3	2	1	0
TK1MSL	TK1MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E4H	7	6	5	4	3	2	1	0
TK1MSH	TK1MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E5H	7	6	5	4	3	2	1	0
TK2MSL	TK2MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E6H	7	6	5	4	3	2	1	0
TK2MSH	TK2MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
E7H	7	6	5	4	3	2	1	0

TK3MSL	TK3MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
D9H	7	6	5	4	3	2	1	0
TK3MSH	TK3MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
DAH	7	6	5	4	3	2	1	0
TK4MSL	TK4MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
DBH	7	6	5	4	3	2	1	0
TK4MSH	TK4MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
DCH	7	6	5	4	3	2	1	0
TK5MSL	TK5MS[7:0]							
R/W	R							
初始值	0	0	0	0	0	0	0	0
DDH	7	6	5	4	3	2	1	0
TK5MSH	TK5MS[15:8]							
R/W	R							
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
15~0	TKnMS	触摸采样数据寄存器

表 20-6-10 寄存器 TKIF

FBH	7	6	5	4	3	2	1	0
TKIF	-	-	TKIF5	TKIF4	TKIF3	TKIF2	TKIF1	TKIF0
R/W	-	-	R	R	R	R	R	R
初始值	-	-	0	0	0	0	0	0

位编号	位符号	说明
7~6	-	-
5	TKIF5	内部参考通道触摸采集中断标志位，写 1 清 0
4	TKIF4	外部触摸通道 4(TK4)触摸采集中断标志位，写 1 清 0
3	TKIF3	外部触摸通道 3(TK3)触摸采集中断标志位，写 1 清 0
2	TKIF2	外部触摸通道 2(TK2)触摸采集中断标志位，写 1 清 0
1	TKIF1	外部触摸通道 1(TK1)触摸采集中断标志位，写 1 清 0
0	TKIF0	外部触摸通道 0(TK0)触摸采集中断标志位，写 1 清 0

21 低电压检测（LVD）

21.1 功能简介

低电压检测（LVD）用于监控芯片自身的供电 VDD，可设置检测电压范围为 2.2V~4.2V（四档可选）。当 VDD 小于所设定的电压值时，可设置触发中断或复位。

备注：*由于生产工艺的影响，芯片之间 LVD 触发电压存在一定的差异。*

LVD 结构图如图 21-1-1 所示。

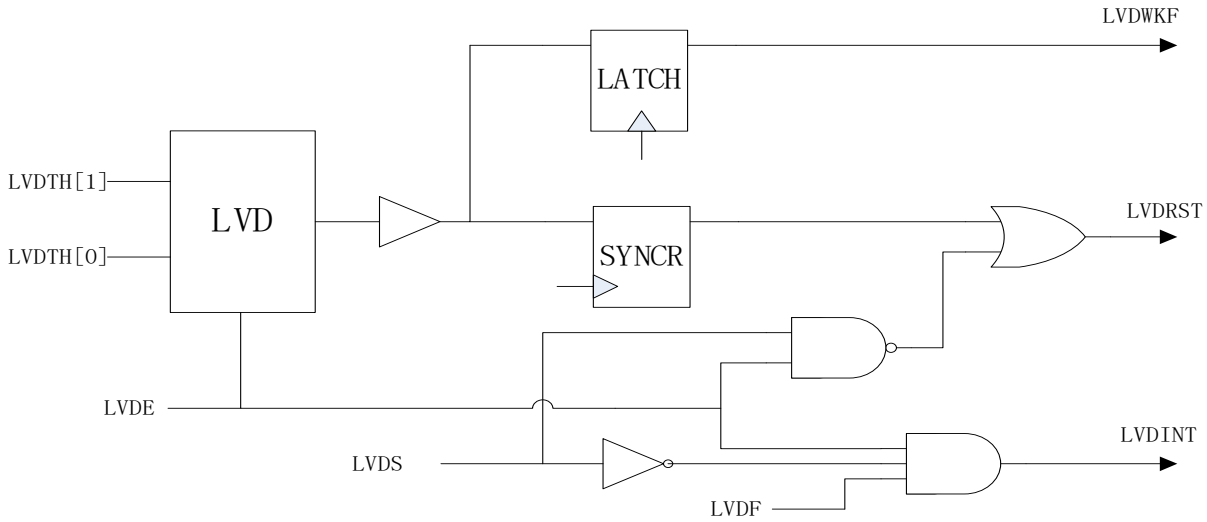


图 21-1-1 LVD 模块示意图

21.2 功能描述

LVD 功能通过 LVDE 位使能，而检测的电压则通过 LVDTH 位域设置。当芯片 VDD 小于所设置的电压时，LVD 功能产生的标志 LVDF 位将置 1，如果 LVDS=0，会产生 LVD 中断，如果 LVDS=1，会产生复位。要注意的是，LVD 复位产生之后，LVD 自身的电路并不会复位，寄存器 LVDCON 还会保持之前的状态，所以，当 LVD 复位产生之后，如果 VDD 持续低于所设定的电压，芯片将会一直处于复位状态。同样地，当 LVD 中断产生后，如果 VDD 持续低于所设定的电压，LVD 中断也会重复地产生。

21.3 寄存器描述

表 21-3-1 寄存器 LVDCON

EFH	7	6	5	4	3	2	1	0

LVDCON	LVDE	LVDS	LVDF				LVDTH[1:0]	
R/W	R/W	R/W	R/W				R/W	
初始值	0	0	0	0	0	0	0	0

位编号	位符号	说明
7	LVDE	LVD 使能位，1 有效
6	LVDS	LVD 功能选择位 0：中断 1：复位
5	LVDF	LVD 产生标志位，写 1 清 0
1~0	LVDTH	LVD 触发电平选择位域 00：2.2V 01：2.7V 10：3.7V 11：4.2V

21.4 LVD 控制例程

LVD 中断例程

例如，设置 LVD 为中断模式，检测电压为 2.2V，程序如下：

```
---------------------------------------------------------------------------------------
#define LVDE(N)                 (N<<7)          //N=0~1
#define LVDS_reset      (1<<6)
#define LVDS_int        (0<<6)
#define LVDF                    (1<<5)

#define LVDTH_2p2V          0
void LVD_init(void)
{
    LVDCON = LVDE(1) | LVDF | LVDS_int | LVDTH_2p2V;  //设置 LVD 使能,设置 LVD 为中断模式,检测电压为 2.2V
    INT4EN = 1;   //INT4 中断使能
    EA = 1;       //开启总中断
}

void LVD_ISR(void) interrupt 6          //LVD 中断服务程序
{
    if(LVDCON & LVDF)
    {
        LVDCON |= LVDF;                 //清除 LVD 中断标志
        //LVD 中断服务程序
```

```
    }
}
```

LVD 复位例程

例如，设置 LVD 为复位模式，检测电压为 2.2V，程序如下：

```
#define LVDE(N)          (N<<7)          //N=0~1
#define LVDS_reset       (1<<6)
#define LVDS_int         (0<<6)
#define LVDF             (1<<5)

#define LVDTH_2p2V          0
void LVD_init(void)
{
    LVDCON = LVDE(1) | LVDF | LVDS_reset | LVDTH_2p2V;//设置 LVD 使能,设置 LVD 为复位模式,检测电压为 2.2V

}
```

---

22 程序下载和仿真

22.1 程序下载

CA51F1 系列芯片主要采用 ISP 方式下载程序，芯片通过 I2C 接口与下载工具相连接，升级接口为 P30(I2C SDA),P31(I2C SCL)。

更多关于程序下载步骤的细节请参考“CACHIP 开发下载工具使用说明”。

22.2 在线仿真

CA51F1 系列芯片支持在线仿真，芯片与仿真器之间通过 IIC 接口进行通信，I2C 接口是 P30(I2C SDA)和 P31(I2C SCL）。要注意的是，由于芯片与仿真器间通过 IIC 通信，所以与仿真器连接的 I2C 接口引脚不能设置为其他功能，并且应用程序里不能使用 IIC 功能，否则将无法进入仿真模式。另外，由于 I2C 的通信速度是由主时钟决定，所以应用程序里不能将主时钟设置为低速时钟，也不能进入省电模式，否则都会影响芯片与仿真器间的通信。

当 TSME=0（PCON[3]）时，芯片禁止进入仿真模式。当芯片进入仿真模式后，TSMODE 位（PCON[2]）置 1，应用程序可通过判断此位状态来决定是否切换至低速时钟或进入省电模式。

更多关于仿真功能的细节可参考仿真器的相关文档介绍。

23 电气特性

23.1 极限参数

参数	最小值	最大值	单位
直流供电电压	-0.3	6	V
I/O 引脚输入电压	-0.3	VDD+0.3	V
工作环境温度	-40	85	℃
储存温度	-45	125	℃

*备注：超过“**极限参数**”范围有可能对芯片造成损坏，无法预期芯片在上述范围外的工作状态，若长期在标示范围外工作，可能会影响芯片的可靠性。*

23.2 直流电气特性

芯片参数	符号	工作电压	最小值	典型值	最大值	单位	测试条件
工作电流	Iop1	VDD=2.5V		2.26		mA	系统时钟为 IRCH(16MHz)，其他时钟关闭，所有输出引脚无负载，所有数字输入引脚不浮动，所有外设关闭，CPU 执行 NOP 指令
		VDD=3.3V		2.90			
		VDD=5V		4.43			
	Iop3	VDD=2.5V		24.7		uA	系统时钟为 IRCL(100kHZ)，其他时钟关闭，所有输出引脚无负载，所有数字输入引脚不浮动，所有外设关闭，CPU 执行 NOP 指令
		VDD=3.3V		28.9			
		VDD=5V		38.6			
STOP 模式电流	Istp	VDD=2.5V		2.0		uA	所有时钟关闭，所有输出引脚无负载，所有数字输入引脚不浮动，所有外设关闭，Flash 进入睡眠模式，CPU 进入 STOP 模式。
		VDD=3.3V		2.1			
		VDD=5V		2.3			
IDLE 模式电流	Iidl1	VDD=2.5V		0.96		mA	系统时钟设为 IRCH（16MHz），其他时钟关闭，所有输出引脚无负载，所有数字输入引脚不浮动，所有外设关闭，Flash 进入睡眠模式，CPU 进入 IDLE 模式。
		VDD=3.3V		1.26			
		VDD=5V		2.05			
	Iidl2	VDD=2.5V		9.0		uA	系统时钟设为 IRCL（100KHz），其他时钟关闭，所有输出引脚无负载，所有数
		VDD=3.3V		11.2			

		VDD=5V		16.5			字输入引脚不浮动，所有外设关闭，CPU 进入 IDLE 模式。
IO 端口输入高电压（斯密特模式开启）	Vhi1	VDD=2.5V	1.20	-	2.5	V	-
		VDD=3.3V	1.60		3.3		
		VDD=5V	2.30		5		
IO 端口输入高电压（斯密特模式关闭）	Vhi2	VDD=2.5V		0.5*VDD	VDD	V	-
		VDD=3.3V					
		VDD=5V					
IO 端口输入低电压（斯密特模式开启）	Vlo1	VDD=2.5V	0	-	1.00	V	-
		VDD=3.3V	0	-	1.30		
		VDD=5V	0	-	1.90		
IO 端口输入低电压（斯密特模式关闭）	Vlo2	VDD=2.5V	0	0.5*VDD		V	-
		VDD=3.3V					
		VDD=5V					
普通 GPIO 推电流	Ipu	VDD=3.3V	-	3.27	-	mA	IO 设为推挽输出模式，驱动能力设为最大，Vol=VDD-0.3V
		VDD=5V	-	4.12	-		
IO 端口灌电流	Iol	VDD=3.3V	-	5.63	-	mA	IO 设为推挽输出模式，驱动能力设为最大，Vol=GND+0.3V
		VDD=5V	-	7.35	-		
IO 端口下拉电阻	Rd1	VDD=2.5~5.5V		30		KΩ	-
IO 端口上拉电阻	Ru1	VDD=2.5~5.5V	-	30	-	KΩ	-

23.3 交流电气特性

交流电气特性（VDD=2.2-5.5V, TA=25℃，除非其它说明）

芯片参数	符号	最小值	典型值	最大值	单位	条件
内部低速时钟（IRCL）起振时间	Trc1	-	50	-	us	IRCL 频率为 100K
内部高速时钟（IRCH）起振时间	Trc2	-	10	-	us	IRCH 频率为 16MHz
复位脉冲时间	Trst	-	0.5	-	us	

备注：VDD=3.3V,TA=25 ℃,内部高速时钟出厂频率为16MHz，精度为±1%.

23.4 最低工作电压

CPU 频率 (单位：MHz)	**最低工作电压（单位:V）**
8	2.2
16	2.7

23.5 内部 RC 时钟温度特性

◆ IRCH 温度特性

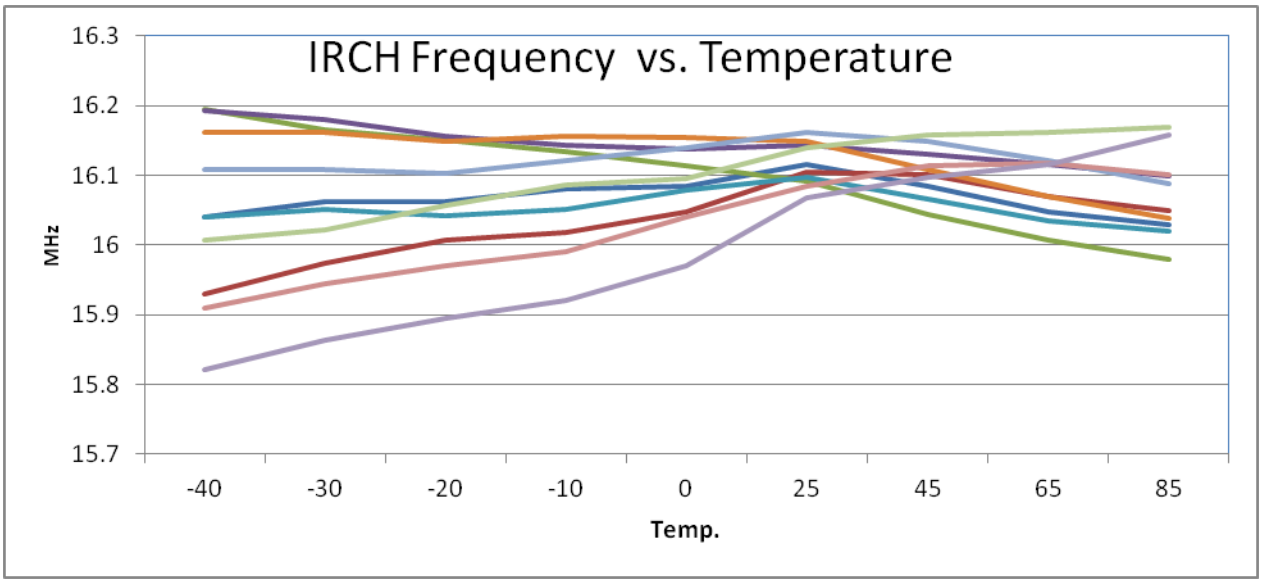


图 23-5-1 IRCH 温度特性曲线图

说明：以上图形数据为随机抽取的 10 片芯片实测数据，数据仅供参考。

◆ IRCL 温度特性

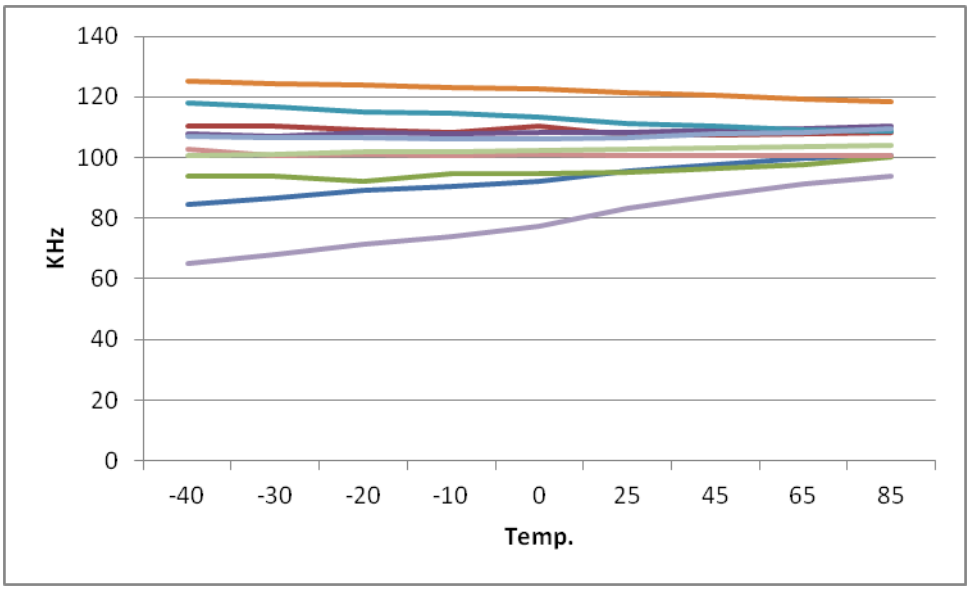
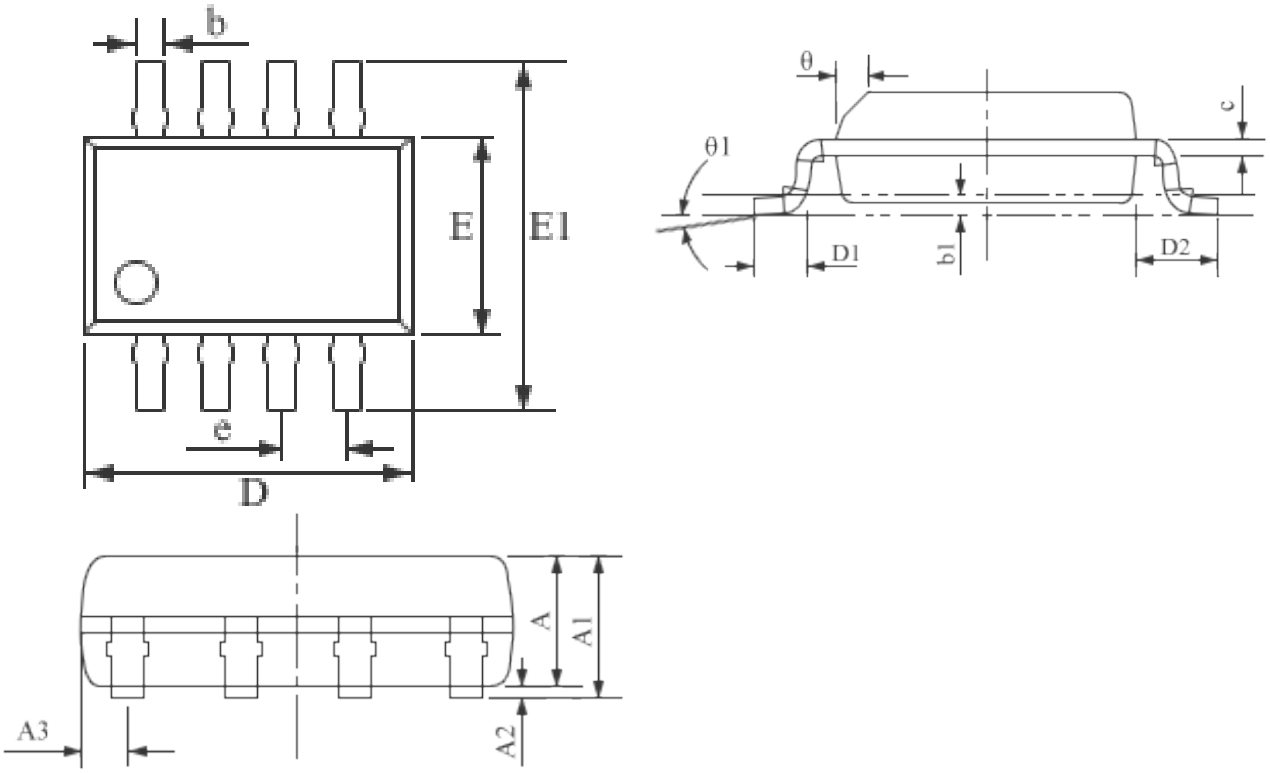


图 23-5-2 IRCL 温度特性曲线图

说明：以上图形数据为随机抽取的 10 片芯片实测数据，数据仅供参考。

24 封装类型

封装形式（一）(SOP8)



序号	最小值(mm)	标准值(mm)	最大值(mm)
A	1.40	1.45	1.50
A1	1.55	1.60	1.65
A2	0.10	0.15	0.20
A3	0.50	0.535	0.540
b	0.354	0.406	0.504
b1	0.155	0.150	0.175
c	0.20	0.203	0.210
D	4.830	4.880	4.910
D1	0.610	0.660	0.710
D2	1.045	1.050	1.0505
e	———	1.270	———
E	3.810	3.910	3.96
E1	5.900	6.000	6.10

25 附录

附录 1 指令集速查表

指令	描述	说明	周期
数据传送指令			
MOV A,Rn	寄存器内容送入累加器	(A) ← (Rn)	1
MOV A,direct	直接地址单元中的数据送入累加器	(A) ← (direct)	1
MOV A,@Ri	间接 RAM 中的数据送入累加器	(A) ← ((Ri))	1
MOV A,#data8	8 位立即数送入累加器	(A) ← #data	1
MOV Rn,A	累加器内容送入寄存器	(Rn) ← (A)	1
MOV Rn,direct	直接地址单元中的数据送入寄存器	(Rn) ← (direct)	2
MOV Rn,#data8	8 位立即数送入寄存器	(Rn) ← #data	1
MOV direct,A	累加器内容送入直接地址单元	(direct) ← (A)	1
MOV direct,Rn	寄存器内容送入直接地址单元	(direct) ← (Rn)	2
MOV direct,direct	直接地址单元中的数据送入直接地址单元	(direct) ← (direct)	2
MOV direct,@Ri	间接 RAM 中的数据送入直接地址单元	(direct) ← ((Ri))	2
MOV direct,#data8	8 位立即数送入直接地址单元	(direct) ← #data	2
MOV @Ri,A	累加器内容送入间接 RAM 单元	((Ri)) ← (A)	1
MOV @Ri,direct	直接地址单元中的数据送入间接 RAM 单元	((Ri)) ← (direct)	2
MOV @Ri,#data8	8 位立即数送入间接 RAM 单元	((Ri)) ← #data	1
MOV DPTR,#data16	16 位立即数地址送入地址寄存器	(DPTR) ← #data116	2
MOV A,@A+DPTR	以 DPTR 为基地址变址寻址单元中的数据送入累加器	(A) ← ((A)) + (DPTR)	2
MOV A,@A+PC	以 PC 为基地址变址寻址单元中的数据送入累加器	(PC) ← (PC) + 1 (A) ← ((A) + (PC))	2
MOVX A,@Ri	外部 RAM(8 位地址)送入累加器	(A) ← ((Ri))	2
MOVX A,@DPTR	外部 RAM(16 位地址)送入累加器	(A) ← ((DPTR))	2
MOVX @Ri,A	累加器送入外部 RAM(8 位地址)	((Ri)) ← (A)	2
MOVX @DPTR,A	累加器送入外部 RAM(16 位地址)	(DPTR) ← (A)	2
PUSH direct	直接地址单元中的数据压入堆栈	(SP) ← (SP) + 1 ((SP)) ← (direct)	2
POP DIRECT	堆栈中的数据弹出到直接地址单元	(direct) ← ((SP)) (SP) ← (SP) - 1	2
XCH A,Rn	寄存器与累加器交换	(A) ↔ (Rn)	1
XCH A,direct	直接地址单元与累加器交换	(A) ↔ (direct)	1
XCH A,@Ri	间接 RAM 与累加器交换	(A) ↔ ((Ri))	1

XCHD A,@Ri	间接 RAM 与累加器进行低半字节交换	(A.3,...,A.0) ↔ ((Ri).3,...,(Ri).0)	1
SWAP A	累加器半字节交换	(A.3,...,A.0) ↔ (A.7,...,A.4)	1
算术操作类指令			
ADD A, Rn	寄存器内容加到累加器	(A) ← (A) + (Rn)	1
ADD A, direct	直接地址单元加到累加器	(A) ← (A) + (direct)	1
ADD A, @Ri	间接 RAM 内容加到累加器	(A) ← (A) + ((Ri))	1
ADD A, #data8	8 位立即数加到累加器	(A) ← (A) + #data	1
ADDC A, Rn	寄存器内容带进位加到累加器	(A) ← (A) + (C) + (Rn)	1
ADDC A, direct	直接地址单元带进位加到累加器	(A) ← (A) + (C) + (direct)	1
ADDC A, @Ri	间接 RAM 内容带进位加到累加器	(A) ← (A) + (C) + ((Ri))	1
ADDC A, #data8	8 位立即数带进位加到累加器	(A) ← (A) + (C) + #data	1
SUBB A, Rn	累加器带借位减寄存器内容	(A) ← (A) - (C) - (Rn)	1
SUBB A, direct	累加器带借位减直接地址单元	(A) ← (A) - (C) - (direct)	1
SUBB A, @Ri	累加器带借位减间接 RAM 内容	(A) ← (A) - (C) - ((Ri))	1
SUBB A, #data8	累加器带借位减 8 位立即数	(A) ← (A) - (C) - #data	1
INC A	累加器加 1	(A) ← (A) + 1	1
INC Rn	寄存器加 1	(Rn) ← (Rn) + 1	1
INC direct	直接地址单元内容加 1	(direct) ← (direct) + 1	1
INC @Ri	间接 RAM 内容加 1	((Ri)) ← ((Ri)) + 1	1
INC DPTR	DPTR 加 1	(DPTR) ← (DPTR) + 1	2
DEC A	累加器减 1	(A) ← (A) – 1	1
DEC Rn	寄存器减 1	(Rn) ← (Rn) - 1	1
DEC direct	直接地址单元内容减 1	(direct) ← (direct) - 1	1
DEC @Ri	间接 RAM 内容减 1	((Ri)) ← ((Ri)) - 1	1
MUL AB	A 乘以 B	temp16 ← (A) X (B) (A)←(temp.7,temp.6,...,temp.0) (B)←(temp.15,temp.14,...,temp.8)	4

DIV AB	A 除以 B	QUO ← (A) / (B) .....REM (A) ← QUO (B) ← REM	4

DA A	累加器进行十进制转换	IF (A.3,...,A.0) > 9 || AC = 1
		THEN
		temp16 ← (A) + 0x06
		(A) ← (temp.7,...,temp.0)
		
		IF (temp16) > 0xFF
		THEN
		CY ← 1
		
		IF (A.7,...,A.4) > 9 || CY = 1
		THEN
		temp16 ← (A) + 0x60
		(A) ← (temp.7,...,temp.0)
		
		IF (temp16) > 0xFF
		THEN
		CY ← 1
			1

逻辑操作类指令			
ANL A, Rn	累加器与寄存器相“与”	(A) ← (A) & (Rn)	1
ANL A, direct	累加器与直接地址单元相“与”	(A) ← (A) & (direct)	1
ANL A, @Ri	累加器与间接 RAM 内容相“与”	(A) ← (A) & ((Ri))	1
ANL A, #data8	累加器与 8 位立即数相“与”	(A) ← (A) & #data	1
ANL direct, A	直接地址单元与累加器相“与	(direct) ← (direct) & (A)	1
ANL direct, #data8	直接地址单元与 8 位立即数相“与”	(direct) ← (direct) & #data	2
ORL A, Rn	累加器与寄存器相“或”	(A) ← (A) | (Rn)	1
ORL A, direct	累加器与直接地址单元相“或”	(A) ← (A) | (direct)	1
ORL A, @Ri	累加器与间接 RAM 内容相“或”	(A) ← (A) | ((Ri))	1
ORL A, #data8	累加器与 8 位立即数相“或”	(A) ← (A) | #data	1
ORL direct, A	直接地址单元与累加器相“或”	(direct) ← (direct) |	1

		(A)	
ORL direct, #data8	直接地址单元与 8 位立即数相“或”	(direct) ← (direct) | #data	2
XRL A, Rn	累加器与寄存器相“异或”	(A) ← (A) ^ (Rn)	1
XRL A, direct	累加器与直接地址单元相“异或”	(A) ← (A) ^ (direct)	1
XRL A, @Ri	累加器与间接 RAM 内容相“异或”	(A) ← (A) ^ ((Ri))	1
XRL A, #data8	累加器与 8 位立即数相“异或”	(A) ← (A) ^ #data	1
XRL direct, A	直接地址单元与累加器相“异或”	(direct) ← (direct) ^ (A)	1
XRL direct, #data8	直接地址单元与 8 位立即数相“异或”	(direct) ← (direct) ^ #data	2
CLR A	累加器清 0	(A) ← 0	1
CPL A	累加器求反	(A) ← /(A)	1
RL A	累加器循环左移	(A) ← (A.6,A.5,...,A.0,A.7)	1
RLC A	累加器带进位循环左移	C ← A.7 (A) ← (A.6,A.5,...,A.0,C)	1
RR A	累加器循环右移	(A) ← (A.0,A.7,...,A.2,A.1)	1
RRC A	累加器带进位循环右移	C ← A.0 (A) ← (C,A.7,...,A.2,A.1)	1
控制转移类指令			
ACALL addr11	绝对短调用子程序	(PC) ← (PC) + 2 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) (SP) ← (SP) + 1 ((SP)) ← (PC15-8) (PC10-0) ← page address	2
LACLL addr16	长调用子程序	(PC) ← (PC) + 3 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) ((SP)) ← (PC15-8) (PC) ←addr15-0	2
RET	子程序返回	(PC15-8) ← ((SP)) (SP) ← (SP) - 1 (PC7-0) ← ((SP)) (SP) ← (SP) - 1	2
RETI	中断返回	(PC15-8) ← ((SP)) (SP) ← (SP) - 1	2

		(PC7-0) ← ((SP)) (SP) ← (SP) - 1	
AJMP addr11	绝对短转移	(PC) ← (PC) + 2 (PC10-0) ← page address	2
LJMP addr16	长转移	(PC) ← (PC) + 3 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) (SP) ← (SP) + 1 ((SP)) ← (PC15-8) (PC10-0) ←addr15-0	2
SJMP rel	相对转移	(PC) ← (PC) + 2 (PC) ← (PC) + rel	2
JMP @A+DPTR	相对于 DPTR 的间接转移	(PC) ← (A) + (DPTR)	2
JZ rel	累加器为零转移	(PC) ← (PC) + 2 IF (A) = 0 THEN (PC) ← (PC) + rel	2
JNZ rel	累加器非零转移	(PC) ← (PC) + 2 IF (A) <> 0 THEN (PC) ← (PC) + rel	2

CJNE A, direct, rel	累加器与直接地址单元比较，不等则转移	(PC) ← (PC) + 3
		IF (A) <> (direct)
		THEN
		(PC) ← (PC) + relative offset
		IF (A) < (direct)
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

CJNE A, #data8, rel	累加器与 8 位立即数比较，不等则转移	(PC) ← (PC) + 3
		IF (A) <> data
		THEN
		(PC) ← (PC) + relative offset
		IF (A) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

CJNE Rn, #data8, rel	寄存器与 8 位立即数比较，不等则转移	(PC) ← (PC) + 3
		IF (Rn) <> data
		THEN
		(PC) ← (PC) +
		relative offset
		IF (Rn) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

CJNE @Ri, #data8, rel	间接 RAM 单元，不等则转移	(PC) ← (PC) + 3
		IF ((Ri)) <> data
		THEN
		(PC) ← (PC) +
		relative offset
		IF ((Ri)) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

DJNZ Rn, rel	寄存器减 1，非零转移	(PC) ← (PC) + 2 (Rn) ← (Rn) - 1 IF (Rn) <> 0 THEN (PC) ← (PC) + rel	2
DJNZ direct, rel	直接地址单元减 1，非零转移	(PC) ← (PC) + 2 (direct) ← (direct) - 1 IF (direct) <> 0 THEN (PC) ← (PC) + rel	2
NOP	空操作	(PC) ← (PC) + 1	1
布尔变量操作类指令			
CLR C	清进位位	(C) ← 0	1
CLR bit	清直接地址位	(bit) ← 0	1
SETB C	置进位位	(C) ← 1	1
SETB bit	置直接地址位	(bit) ← 1	1
CPL C	进位位求反	(C) ← /(C)	1
CPL bit	直接地址位求反	(bit) ← /(bit)	1
ANL C, bit	进位位和直接地址位相“与”	(C) ← (C) & (bit)	2
ANL C, /bit	进位位和直接地址位的反码相“与”	(C) ← (C) & /(bit)	2
ORL C, bit	进位位和直接地址位相“或”	(C) ← (C) | (bit)	2
ORL C, /bit	进位位和直接地址位的反码相“或”	(C) ← (C) | /(bit)	2
MOV C, bit	直接地址位送入进位位	(C) ← (bit)	1

MOV bit, C	进位位送入直接地址位	(bit) ← (C)	2
JC rel	进位位为 1 则转移(CY=O 不转移，=1 转移)	(PC) ← (PC) + 2 IF (C) = 1 THEN (PC) ← (PC) + rel	2
JNC rel	进位位为 0 则转移	(PC) ← (PC) + 2 IF (C) = 0 THEN (PC) ← (PC) + rel	2
JB bit, rel	直接地址位为 1 则转移	(PC) ← (PC) + 3 IF (bit) = 1 THEN (PC) ← (PC) + rel	2
JNB bit, rel	直接地址位为 0 则转移	(PC) ← (PC) + 3 IF (bit) = 0 THEN (PC) ← (PC) + rel	2
JBC bit, rel	直接地址位为 1 则转移，该位清零	(PC) ← (PC) + 3 IF (bit) = 1 THEN (bit) ← 0 (PC) ← (PC) + rel	2
伪指令			
ORG	设置程序起始地址		
END	标志源代码结束		
EQU	定义常数		
SET	定义整型数		
DATA	给数据地址定值		
BYTE	给字节类型符号定值		
WROD	给字类型符号定值		
BIT	给位地址取名		
ALTNAME	用自定义名取代保留字		
DB	给一块连续的存储区装载字节型数据		
DW	给一块连续的存储区装载字型数据		
DS	预留一个连续的存储区或装入指定字节		
INCLUDE	将一个源文件插入程序中		
TITLE	列表文件中加入标题行		
NOLIST	汇编时不产生列表文件		
NOCODE	条件汇编时，条件为假的不产生清单		