

YS67FXXXX(X)

数据手册（预览版）（版本 V1.3）

1. 器件概述

8-Bit Touch Flash MCU

- ◆ 基于 8051 指令流水线结构的 8 位单片机。
- Flash ROM: 8K 字节, 100,000 次可擦除
- Firmware 可访问, 用户数据保护, 支持 16bit CRC。

4COM-23SEG

5COM-22SEG

6COM-21SEG

- ◆ **RAM:IRAM** :256 字节,
- XRAM:829 字节,其中 General XRAM 占用 608 字节,CS XRAM 占用 160 字节,CS CTRLXRAM 占用 32 字节,LCDXRAM 占用 24 字节, LCDRAM 24 字节
- SFR:128 字节

8COM-19SEG, 可选

LED 驱动: 8COM-8SEG 共阴极/共阳极设置

- ◆ **工作电压:** 1.8V~5.5V

- ◆ **振荡器:**

Fosc=12MHZ \pm 1% (内部/外部振荡)

Losc=32KHZ (内部/外部振荡)

- ◆ 4 个通用 16 位定时器/计数器: T0、T1、T2、T3
- T1,T2,T3 具有 PWM 功能
- T2,T3 具有 CAPTURE 功能

- ◆ **运行模式**

- 正常模式
- 待机模式
- 睡眠
- 深度睡眠

- ◆ **最多 41 GPIO**

- ◆ UART

- ◆ **Touch:**内部集成 8 个触摸通道

- ◆ SPI

- ◆ **支持 LCD/LED 功能**

- ◆ I2C

LCD 驱动: LCD 1/3 BIAS

- ◆ 10 通道 10bit ADC

支持 3COM-24SEG

- ◆ 支持在线仿真功能

器件	VDD	ROM	RAM			I/O	Touch (CH)	Timer	Interface	LCD	Package
		FLASH (word)	IRAM (word)	XRAM (word)	SFR (word)						
YS67F9808(B)	1.8V~5.5V	8K	256	829	128	41	8	4*16bit	SPI/I2C/UART	8COM-19SEG	LQFP48

目 录

1. 器件概述	1
1.1. 系统结构图	5
1.2. 封装脚位图	6
1.3. 引脚说明	7
2. CPU 内核	10
3. 数据存储空间	11
4. SFR 空间	12
4.1. FLASH 程序存储器	12
5. FLASH 安全管理（软件加密）	15
5.1. 特性	16
5.2. 安全保护等级	16
5.3. LOCKBYTE 值和加密块的对应关系	17
5.4. 编程步骤	19
6. 功耗模式	21
6.1. 正常模式	21
6.2. 待机模式	21
6.3. 睡眠模式	22
6.4. 深度睡眠模式	22
6.5. PCON 寄存器	22
6.6. 唤醒时间	23
7. 复位源	23
7.1. 外部复位、上电复位	23
7.2. 低电压侦测复位	23
7.3. 看门狗溢出复位	24
7.4. FEDR 复位	24
7.5. 软复位	24
7.6. 过度电应力复位	24
7.7. 各复位事件对系统的影响	24
7.8. 复位寄存器	24
8. 输入/输出端口	26
8.1. IO 等效电路及说明	26
8.2. 寄存器说明	26
9. 中断系统	39
9.1. 中断源	39
9.2. 关于 IO 电平变化中断	40
9.3. 缺失时钟中断（MCD）	41
9.4. 中断相关寄存器	42
10. 系统时钟	46

10.1. CLKCF 寄存器	46
10.2. 时钟切换流程	47
10.3. 快慢切换	47
10.4. 由外部时钟切换到内部时钟	47
10.5. 切换到外部时钟注意	48
10.6. 外设的时钟门控	48
11. RTC.....	50
11.1. 概述	50
11.2. 功能列表	50
11.3. 功能框图	50
11.4. 功能描述	50
11.5. SFR 描述	50
12. 定时器	51
12.1. 定时器 1	51
12.2. 定时器 2	55
12.3. 定时器 3	60
12.4. 定时器 4	66
13. 看门狗定时器	71
13.1. WDTCF 寄存器	71
14. UART	72
14.1. 工作模式	72
15. 增强型串行外设接口 (SPI)	75
15.1. 引脚说明	75
15.2. SPI 主方式	75
15.3. SPI 从方式	76
15.4. SPI 中断源	77
15.5. SFR	78
16. I2C.....	80
16.1. I2C 操作步骤	82
16.2. I2C 寄存器	82
17. 循环冗余检查单元 (CRC)	85
17.1. CRC 寄存器	85
17.2. 操作步骤	88
18. LCD/LED	89
18.1. 概述	89
18.2. 功能列表	89
18.3. 功能框图	89
18.4. 模块端口定义	90
18.5. 功能详述	90
18.6. 寄存器说明	96

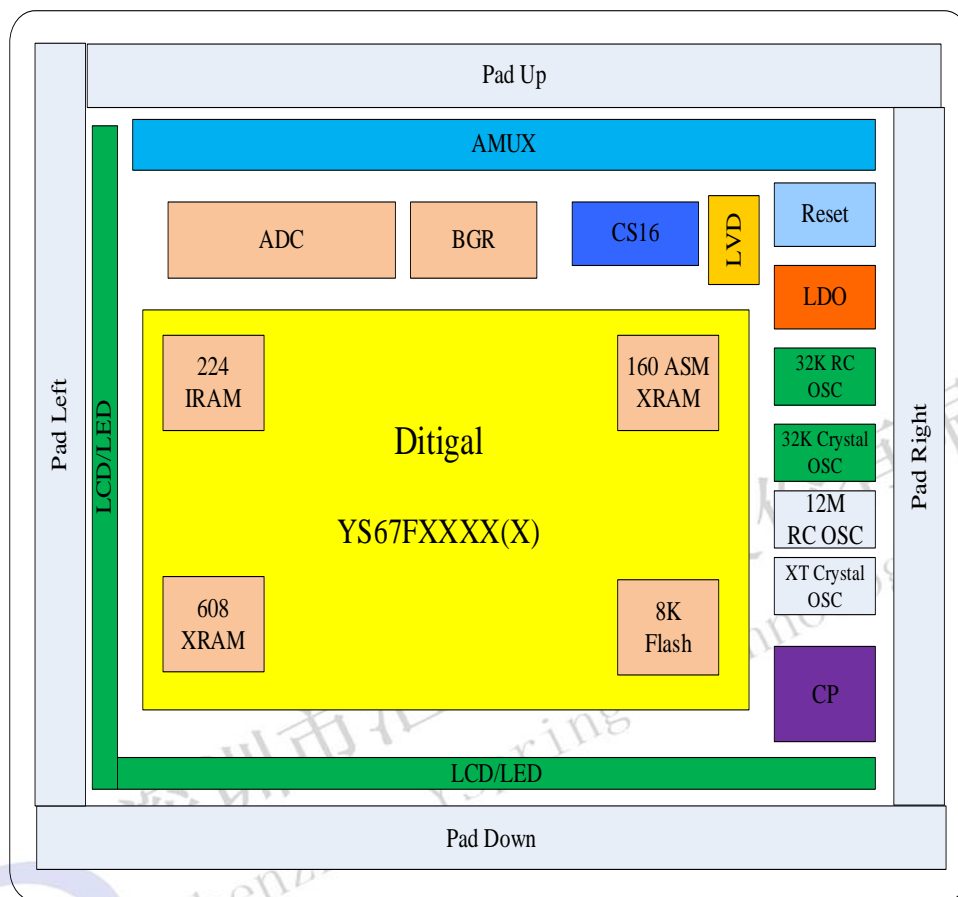
19.	电容感应(CS16)	98
19.1.	概述	98
19.2.	功能列表	98
20.	ADC	99
20.1.	引脚描述	99
20.2.	低功耗模式 (LOW POWER MODE)	100
20.3.	ADC0 模拟多路选择器	100
20.4.	寄存器说明	100
21.	电气特性	105
22.	封装信息	106
22.1.	LQFP48	106
23.	汇春知识产权政策	107
23.1.	专利权	107
23.2.	著作权	107
24.	版本编号	107



深圳市汇春科技股份有限公司
Shenzhen Yspring Technology Co., Ltd.

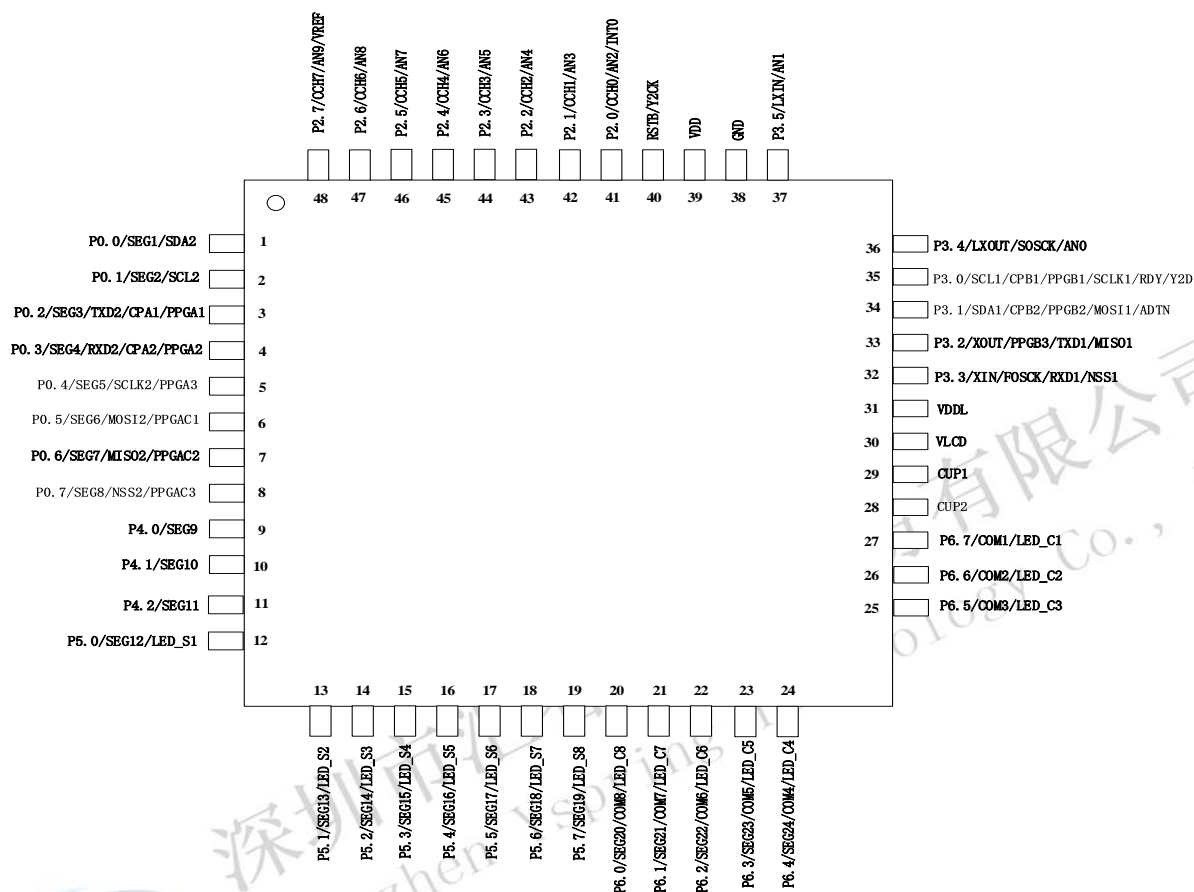
1.1. 系统结构图

YS67FXXXX(X)为一款基于 8051 内核的单片机，大部分指令周期为 1T 或 2T，具有电容触摸检测、LCD 显示、ADC、PPG、串口等功能，工作电压为 1.8~5.5V。主要应用于同时带 LCD 和触摸输入的市场。



1.2. 封装脚位图

1.2.1 LQFP48 封装引脚图



1.3. 引脚说明

Name	Description		
	48		
P0.0/SEG1/SDA2	1	B	P0.0 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 1 功能转移后的 I2C 之数据端
P0.1/SEG2/SCL2	2	B	P0.1 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 2 功能转移后的 I2C 之时钟端
P0.2/SEG3/TXD2/CPA1/PPGA1	3	B	P0.2 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 3 功能转移后的 UART 之发射端 功能转移前的 CAPTURE 输入 1, 配合 timer3 使用 功能转移前的 PWM 输出 1, 配合 timer3 使用
P0.3/SEG4/RXD2/CPA2/PPGA2	4	B	P0.3 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 4 功能转移后的 UART 之接收端 功能转移前的 CAPTURE 输入 2, 配合 timer4 使用 功能转移前的 PWM 输出 2, 配合 timer4 使用
P0.4/SEG5/SCLK2/PPGA3	5	B	P0.4 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 5 功能转移后的 SPI 之时钟端 功能转移前的 PWM 输出 3, 配合 timer2 使用
P0.5/SEG6/MOSI2	6	B	P0.5 可编程 IO, 可设置端口变化产生产生中断 LCD Segment 6 功能转移后的 SPI 数据端, Master 模式输出, Slave 模式输入
P0.6/SEG7/MISO2	7	B	P0.6 可编程 IO, 可设置端口变化产生产生中断 LCD Segment7 功能转移后的 SPI 数据端, Master 模式输入, Slave 模式输出
P0.7/SEG8/NSS2	8	B	P0.7 可编程 IO, 可设置端口变化产生产生中断 LCD Segment8 功能转移后的 SPI 选择端口
P4.0/SEG9	9	B	P4.0 可编程 IO LCD Segment 9
P4.1/SEG10	10	B	P4.1 可编程 IO LCD Segment 10
P4.2/SEG11	11	B	P4.2 可编程 IO LCD Segment 11
P5.0/SEG16/ LED_S1	12	B	P5.0 可编程 IO LCD Segment 12
P5.1/SEG17/LED_S2	13	B	P5.1 可编程 IO

			LCD Segment 13 LED Segment 2
P5.2/SEG18/LED_S3	14	B	P5.2 可编程 IO LCD Segment 14 LED Segment 3
P5.3/SEG19/LED_S4	15	B	P5.3 可编程 IO LCD Segment 15 LED Segment 4
P5.4/SEG20/LED_S5	16	B	P5.4 可编程 IO LCD Segment 16 LED Segment 5
P5.5/SEG21/LED_S6	17	B	P5.5 可编程 IO LCD Segment 17 LED Segment 6
P5.6/SEG22/LED_S7	18	B	P5.6 可编程 IO LCD Segment 18 LED Segment 7
P5.7/SEG23/LED_S8	19	B	P5.7 可编程 IO LCD Segment 19 LED Segment 8
P6.0/SEG20/COM8/LED_C8	20	B	P6.0 可编程 IO LCD Segment 20 LCD COM 8 LED COM 8
P6.1/SEG21/COM7/LED_C7	21	B	P6.1 可编程 IO LCD Segment 21 LCD COM 7 LED COM 7
P6.2/SEG22/COM6/LED_C6	22	B	P6.2 可编程 IO LCD Segment 22 LCD COM 6 LED COM 6
P6.3/SEG23/COM5/LED_C5	23	B	P6.3 可编程 IO LCD Segment 23 LCD COM 5 LED COM 5
P6.4/SEG24/COM4/LED_C4	24	B	P6.4 可编程 IO LCD Segment 24 LCD COM 4 LED COM 4
P6.5/COM3/LED_C3	25	B	P6.5 可编程 IO LCD COM 3 LED COM 3
P6.6/COM2/LED_C2	26	B	P6.6 可编程 IO LCD COM 2

			LED COM 2
P6.7/COM1/LED_C1	27	B	P6.7 可编程 IO LCD COM 1 LED COM 1
CUP2	28	O	Charge Pump 输出 2
CUP1	29	O	Charge Pump 输出 1 与 CUP2 之间接 0.1uF 电容
VLCD	30	O	LCD 电源, 接 1uF 电容到地
VDDL	31	O	1.8V 数字电源, 内建 LDO, 外接 1uF 电容到地
P3.3/XIN/FOSCK/RXD1/NSS1	32	B	P3.3 可编程 IO 快时钟晶振输入 外灌快时钟输入 UART 数据接收端 SPI 选择端口
P3.2/XOUT/PPGB3/TXD1/MISO1	33	B	P3.2 可编程 IO 快时钟晶振输出 功能转移后 PWM 输出 3, 配合 timer2 使用 UART 数据发射端 SPI 数据端, Master 模式输入, Slave 模式输出
P3.1/SDA1/CPB2/PPGB2/MOSI1/AD TN	34	B	P3.1 可编程 IO, 支持 Open Drain 输出 功能转移前 I2C 之数据端 功能转移后 Capture 输入 2, 配合 timer4 使用 功能转移后 PWM 输出 2, 配合 timer4 使用 功能转移前 SPI 数据端, Master 模式输出, Slave 模式输入 FLASH 测试输入脚
P3.0/SCL1/CPB1/PPGB1/SCLK1/RD Y/Y2D	35	B	P3.0 可编程 IO, 支持 Open Drain 输出 功能转移前 I2C 之时钟端 功能转移后 Capture 输入 1, 配合 timer3 使用 功能转移后 PWM 输出 1, 配合 timer3 使用 功能转移前 SPI 时钟端 FLASH 测试输出脚 Y2D 端口
P3.4/LXOUT/SOSCK/AN0	36	B	P3.4 可编程 IO 低频晶振输出, 外接 32768 晶振 外灌慢时钟输入 ADC 输入通道 0
P3.5/LXIN/AN1	37	B	P3.5 可编程 IO 低频晶振输入, 外接 32768 晶振 ADC 输入通道 1
GND	38	I	电源地
VDD	39	I	电源输入, 1.8~5.5V
RSTB/Y2CK	40	I	片外 Reset 脚, 低电平复位, 内置上拉 Y2CK 端口
P2.0 /CCH0/AN2INT0	41	B	P2.0 可编程 IO 电容触摸传感输入通道 0

			ADC 输入通道 2 中断 0 输入
P2.1 /CCH1/AN3	42	B	P2.1 可编程 IO 电容触摸传感输入通道 1 ADC 输入通道 3
P2.2 /CCH2/AN4	43	B	P2.2 可编程 IO 电容触摸传感输入通道 2 ADC 输入通道 4
P2.3 /CCH3/AN5	44	B	P2.3 可编程 IO 电容触摸传感输入通道 3 ADC 输入通道 5
P2.4 /CCH4/AN6	45	B	P2.4 可编程 IO 电容触摸传感输入通道 4 ADC 输入通道 6
P2.5 /CCH5/AN7	46	B	P2.5 可编程 IO 电容触摸传感输入通道 5 ADC 输入通道 7
P2.6 /CCH6/AN8	47	B	P2.6 可编程 IO 电容触摸传感输入通道 6 ADC 输入通道 8
P2.7 /CCH7/AN9/VREF	48	B	P2.7 可编程 IO 电容触摸传感输入通道 7 ADC 输入通道 9 ADC 参考电压输入或者输出

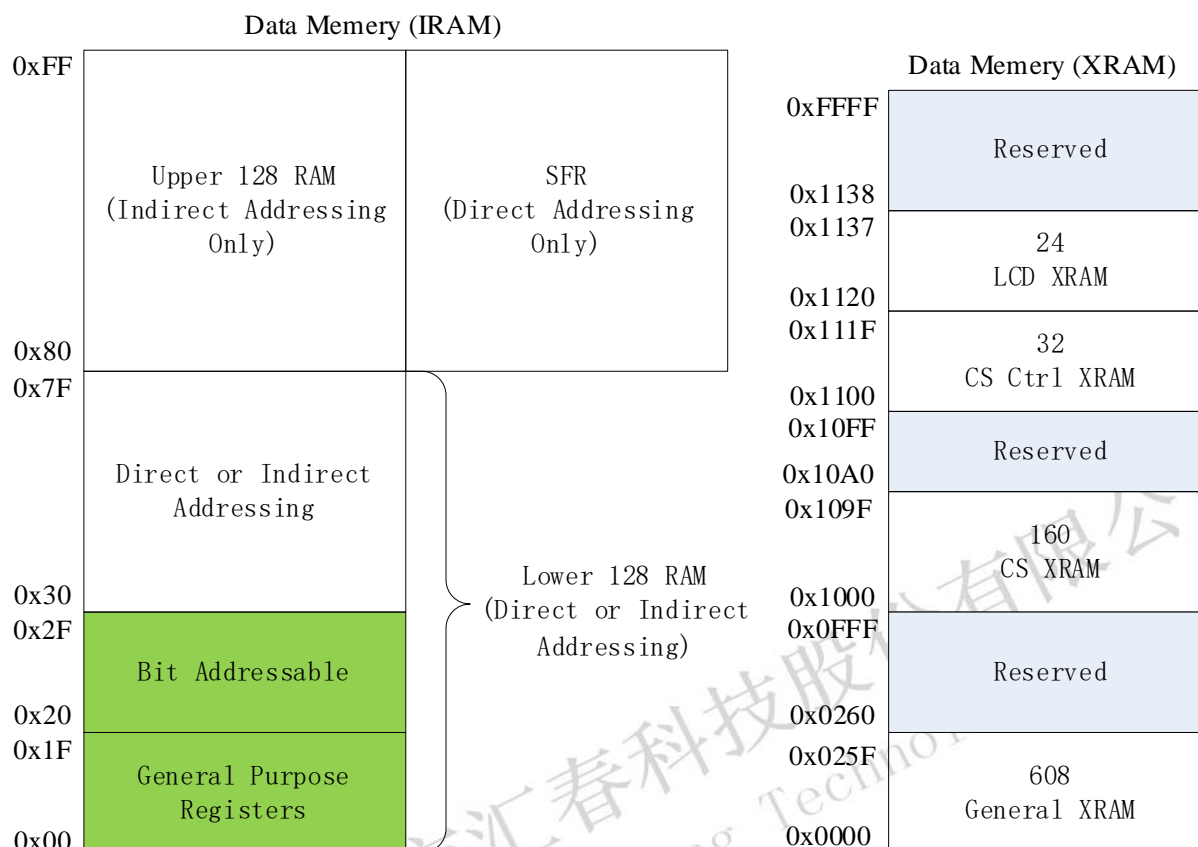
2.CPU 内核

YS67FXXXX(X)的 MCU 核是流水线设计的 8051 核，指令集和标准 8051 全兼容，其流水线设计使得 70% 的指令可以在 1 到 2 个系统时钟内完成，最慢的除法指令也只需要 8 个系统时钟。YS67FXXXX(X) 只支持高速 12MHz 和慢速 32KHz 内部振荡，在出厂时已经被校准为 12MHz，其精度在整个温度和电压范围内为 $\pm 1\%$ 。

YS67FXXXX(X)共有 111 条指令，下表列出了各种指令执行时间（指令执行时所需的系统时钟周期数）所对应的指令条数：

执行周期数	1	2	3	4	5	8
指令数	26	55	23	4	2	1

3. 数据存储空间



- FLASH

FLASH 地址范围是 0x0000~0x1FFF，共 8KB。

- XRAM

一共有 4 块独立的 XRAM，其中低 608 字节是通用 XRAM，供软件使用；0x1000~0x109F 是 CS XRAM（触摸章节有详细介绍）；0x1100~111F 是 CS CTRL XRAM；0x1120~0x1137 是 LCD XRAM

内部 RAM 大小为 256 字节，其中低 128 字节可以直接或间接寻址；高 128 字节的地址 0x80~0xFF 和 SFR 空间重叠，使用直接寻址指令将访问 SFR，间接寻址访问 IRAM 的高 128 字节。

4.SFR 空间

从 0x80 到 0xFF 的直接寻址存储器空间为特殊功能寄存器 (SFR),地址以 0 或者 8 结尾的 SFR 既可以按字节寻址也可以按位寻址,所有其它 SFR 只能按字节寻址。SFR 空间中未使用的地址保留为将来使用,访问这些地址会产生不确定的结果,应避免。下表列出的每个寄存器的详细说明请参见本数据表的相关部分。

地址	0(8)	1(9)	2(A)	3(B)	4(C)	5(D)	6(E)	7(F)
0xF8	SPICTL					DISP_CFG	DISP_EN	CFGKEY
0xF0	B	TMR4RLL	TMR4RLH	TMR4BLL	TMR4BLH		PER_EN	TEST
0xE8	EIE1	RSTEN(CCFG0)	CLKCF(CCFG1)	CCFG2	CCFG3			RSTSRC
0xE0	ACC	TMR3RLL	TMR3RLH	TMR3BLL	TMR3BLH	TMR2DZ	TMR3DZ	TMR4DZ
0xD8	EIP1					P4AN	P4PU	P2PU
0xD0	PSW	P0DIR		P2DIR	P3DIR	P0PU	P5PU	PXC
0xC8	TMR2CN	P4	P5	P6	P3AN	P0AN	P5AN	P2AN
0xC0	IRQ0	P4DIR	P5DIR	P6DIR		P6AN	P6PU	MCDST
0xB8	IP	FCTR	I2C_DIN	I2C_DOUT	I2C_SLAD	I2C_STAT	RTCL	RTCH
0xB0	P3	TMR3CN	TMR4CN					FLKEY
0xA8	IE	ADC0L	ADC0H					ANA_CTL
0xA0	P2	SPICFG	SPISCR	SPIDAT		ADC0PWR	ADC0TK	ADC0MX
0x98	SCON0	SBUF0	CRC0CN	CRC0IN	CRC0DAT	CRC0ST	CRC0CNT	REF0CN
0x90		TMR2RLL	TMR2RLH	TMR2BLL	TMR2BLH	ADC0CN	ADC0CF	ADC0AC
0x88	TCON	TMOD		TL1		TH1	CKCON	PSCTL
0x80	P0	SP	DPL	DPH	P0SEL	WKSRC	WDTCF	PCON

4.1. FLASH 程序存储器

YS67FXXXX(X)有 8K 字节的 Flash,一共有 128 个扇区,每个扇区 64 字节。其中主程序区是 128 扇区控制器支持内部软件编程和 Y2 接口编程,对编程时系统时钟的频率没有要求。本 FLASH 可支持一次对一个扇区的编程。本节讲解软件更新 Flash 的方法。

软件读 Flash 用 MOVC 指令,软件擦除、编程用的是 MOVX 指令。在使用 MOVX 指令对 Flash 写入之前,必须将程序写允许位 FLA_WEN (PSCTL.0)置 1,以允许写操作,它告诉了 MCU 之后的 MOVX 指令指向的是 Flash 而不是 XRAM。

强烈要求 Flash 写期间禁止中断,以避免出错

Flash 扇区擦除后,数据是 0x00 而不是常见的 0xFF! 写 Flash 可以把数据位置 1,但不能使数据位清 0。如果某一字节已经编程过了,要对它重写就必须先擦除其所在扇区,再执行写入操作。

写和擦除均由硬件自动定时,以确保操作正确,无须进行数据查询来操作的完成与否。

编程步骤:

1 禁止中断

- 2 置 FLA_SEC (PSCTL.1) 和 FLA_WEN (PSCTL.0) 为 1, 以允许软件扇区擦除
- 3 顺序往 FLKEY 写 0xA5 和 0xF1, 进入开锁状态
- 4 加入五条 nop 指令
- 5 用 MOVX 指令向待擦除扇区的某一地址写任意值
- 6 向 FLA_ACT (PSCTL.2) 写 1, 使 Flash 进入擦除状态
- 7 清除 FLA_SEC 禁止擦除操作
- 8 顺序往 FLKEY 写 0xA5 和 0xF1, 重新进入开锁状态
- 9 加入五条 nop 指令
- 10 用 MOVX 指令向刚擦除的扇区的目标地址写入数据, 可以重复此步骤直到写满所需字节 (最大支持 64B, 不支持跨页编程!)
- 11 向 FLA_ACT (PSCTL.2) 写 1, 使 Flash 进入编程状态
- 12 清除 FLA_WEN 禁止 Flash 写
- 13 重新允许中断

注意:

- 1、为避免软件跑飞误对 Flash 擦除, 只允许软件对块和扇区的擦除, 而不允许软件对整片 Flash 擦除。
- 2、要开锁一定要先写 0xA5, 后写 0xF1, 写 0xA5 和 0xF1 中间允许 MCU 其它操作。“MOVX 编程 Flash”功能在没冻结的情况下, 任何往 PSCTL 的写操作将使此功能重新上锁; 若不按照这个写顺序或者数据不对, 则此功能将冻结, 直到下一次系统复位!
- 3、程序过程出现跨页时, 硬件会记录在 PSCTL.3。当出现这种情况时, FLASH 控制器将放弃这一操作, 故用户程序应检查这一位。
- 4、在已经开锁的状态不要往 FLKEY 写数据, 否则会进入冻结状态。如果要重新上锁, 则需要往执行一次 PSCTL 写操作, 可以是任何数值。

4.1.1 编程控制寄存器 PSCTL (SFR 地址 0x8F)

位	名字	复位值	功 能
7:4	NA	NA	保留位, 写无效。读=0x00
3	FLA_VIO	0	编程出错标志, 可读写 1: 在对 FLASH 编程时, 连续的 MOVX 指令目标地址出现跨页。可由软件清 0, 或者在进行一次成功的扇区编程操作后由硬件清 0 0: 在对 FLASH 编程时, 连续的 MOVX 指令目标地址落在同一扇区
2	FLA_ACT	0	空寄存器位, 写 1 表示开始 Flash 操作, 如编程, 擦除 写 0 无效, 读永远是 0
1	FLA_SEC	0	扇区擦除使能, 高有效
0	FLA_WEN	0	编程使能, 高有效 只有在 FLA_WEN 为高时, FLA_SEC 才起作用

4.1.2 FLASH 编程开锁寄存器 FLKEY (SFR 地址 0xB7)

位	名字	复位值	功 能
7:0	FLKEY	NA	FLKEY 并没有实际的寄存器映射, 它负责解释来自软件写内容, 进而控制“软件编程 FLASH”的功能

		<p>写： 按先后顺序往 FLKEY 写 0xA5、0xF1 将开启“软件编程 FLASH”功能。若顺序不对或者写其他值将使此功能冻结，直到下一次系统复位！</p>
		<p>读： 最低 2 位反映的是内部状态，高 6 位返回的是 0x00： 00：上锁 01：0xA5 已经写入，等待 0xF1 写入 10：开锁 11：冻结</p>

表 4-1 FLKEY 描述

4.1.3 扇区和地址的关系

0	0x0000 – 0x003f	64	0x1000 – 0x103f
1	0x0040 – 0x007f	65	0x1040 – 0x107f
2	0x0080 – 0x00bf	66	0x1080 – 0x10bf
3	0x00c0 – 0x00ff	67	0x10c0 – 0x10ff
4	0x0100 – 0x013f	68	0x1100 – 0x113f
5	0x0140 – 0x017f	69	0x1140 – 0x117f
6	0x0180 – 0x01bf	70	0x1180 – 0x11bf
7	0x01c0 – 0x01ff	71	0x11c0 – 0x11ff
8	0x0200 – 0x023f	72	0x1200 – 0x123f
9	0x0240 – 0x027f	73	0x1240 – 0x127f
10	0x0280 – 0x02bf	74	0x1280 – 0x12bf
11	0x02c0 – 0x02ff	75	0x12c0 – 0x12ff
12	0x0300 – 0x033f	76	0x1300 – 0x133f
13	0x0340 – 0x037f	77	0x1340 – 0x137f
14	0x0380 – 0x03bf	78	0x1380 – 0x13bf
15	0x03c0 – 0x03ff	79	0x13c0 – 0x13ff
16	0x0400 – 0x043f	80	0x1400 – 0x143f
17	0x0440 – 0x047f	81	0x1440 – 0x147f
18	0x0480 – 0x04bf	82	0x1480 – 0x14bf
19	0x04c0 – 0x04ff	83	0x14c0 – 0x14ff
20	0x0500 – 0x053f	84	0x1500 – 0x153f
21	0x0540 – 0x057f	85	0x1540 – 0x157f
22	0x0580 – 0x05bf	86	0x1580 – 0x15bf
23	0x05c0 – 0x05ff	87	0x15c0 – 0x15ff
24	0x0600 – 0x063f	88	0x1600 – 0x163f
25	0x0640 – 0x067f	89	0x1640 – 0x167f
26	0x0680 – 0x06bf	90	0x1680 – 0x16bf
27	0x06c0 – 0x06ff	91	0x16c0 – 0x16ff
28	0x0700 – 0x073f	92	0x1700 – 0x173f
29	0x0740 – 0x077f	93	0x1740 – 0x177f
30	0x0780 – 0x07bf	94	0x1780 – 0x17bf

31	0x07c0 – 0x07ff	95	0x17c0 – 0x17ff
32	0x0800 – 0x083f	96	0x1800 – 0x183f
33	0x0840 – 0x087f	97	0x1840 – 0x187f
34	0x0880 – 0x08bf	98	0x1880 – 0x18bf
35	0x08c0 – 0x08ff	99	0x18c0 – 0x18ff
36	0x0900 – 0x093f	100	0x1900 – 0x193f
37	0x0940 – 0x097f	101	0x1940 – 0x197f
38	0x0980 – 0x09bf	102	0x1980 – 0x19bf
39	0x09c0 – 0x09ff	103	0x19c0 – 0x19ff
40	0x0a00 – 0x0a3f	104	0x1a00 – 0x1a3f
41	0x0a40 – 0x0a7f	105	0x1a40 – 0x1a7f
42	0x0a80 – 0x0abf	106	0x1a80 – 0x1abf
43	0x0ac0 – 0x0aff	107	0x1ac0 – 0x1aff
44	0x0b00 – 0x0b3f	108	0x1b00 – 0x1b3f
45	0x0b40 – 0x0b7f	109	0x1b40 – 0x1b7f
46	0x0b80 – 0x0bbf	110	0x1b80 – 0x1bbf
47	0x0bc0 – 0x0bff	111	0x1bc0 – 0x1bff
48	0x0c00 – 0x0c3f	112	0x1c00 – 0x1c3f
49	0x0c40 – 0x0c7f	113	0x1c40 – 0x1c7f
50	0x0c80 – 0x0cbf	114	0x1c80 – 0x1cbf
51	0x0cc0 – 0x0cff	115	0x1cc0 – 0x1cff
52	0x0d00 – 0x0d3f	116	0x1d00 – 0x1d3f
53	0x0d40 – 0x0d7f	117	0x1d40 – 0x1d7f
54	0x0d80 – 0x0dbf	118	0x1d80 – 0x1dbf
55	0x0dc0 – 0x0dff	119	0x1dc0 – 0x1dff
56	0x0e00 – 0x0e3f	120	0x1e00 – 0x1e3f
57	0x0e40 – 0x0e7f	121	0x1e40 – 0x1e7f
58	0x0e80 – 0x0ebf	122	0x1e80 – 0x1ebf
59	0x0ec0 – 0x0eff	123	0x1ec0 – 0x1eff
60	0x0f00 – 0x0f3f	124	0x1f00 – 0x1f3f
61	0x0f40 – 0x0f7f	125	0x1f40 – 0x1f7f
62	0x0f80 – 0x0fbf	126	0x1f80 – 0x1fbf
63	0x0fc0 – 0x0fff	127	0x1fc0 – 0x1fff

5. FLASH 安全管理（软件加密）

位于 NVR 中地址 0x3d 的一个字节是安全锁定字节，器件擦除后对其编程可以实现对代码的保护。例如存放在 lock byte 单元的值为 0x01，则表示 block0 (0x0000~0x00ff) 将被保护，一个 block 为 256 字节，后面有个 lock byte 与加密区域表可参考。

在上电过程 LockByte 值被映射到一个影子寄存器（SFR 地址 0x84，特殊访问方式），它决定了 Flash 的读写、擦除操作的合法性，在执行完一次器件擦除，或者 LockByte 的编程操作后，这个影子寄存器将被更新，无

需重复上电过程。

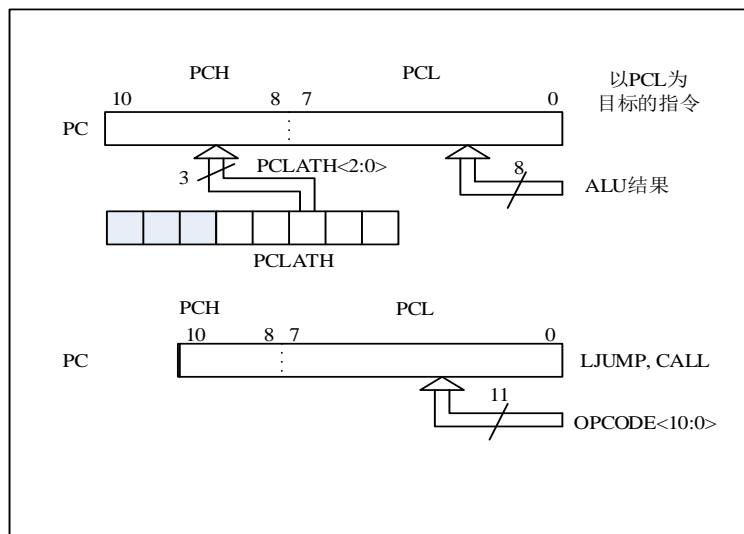


图 5-1 FLASH 地址映射

5.1. 特性

- 代码保护以 256B 大小的块进行管理
- 软件的非法读写、擦除操作将产生 FEDR 复位
- 忽略 Y2 接口的非法读写、擦除操作

5.2. 安全保护等级

访问 Flash 可能的方式有以下 6 种：

- 通过 Y2 接口访问加密块
- 通过 Y2 接口访问非加密块
- 从加密块访问加密块
- 从加密块访问非加密块
- 从非加密块访问非加密块
- 从非加密块访问加密块

以上六种操作可能产生的影响如下表所示:

序号	行 为	Y2 调试接口	用户固件执行位置	
			非加密块	加密块
1	读、写或擦除非加密块 (没有加密块的情况)	允许	允许	允许
2	读、写或擦除加密块 (有加密块的情况)	不允许	不允许 产生 FEDR 复位	允许
3	读取 LockByte(NVR)的值 (没有加密块的情况)	允许	不允许	不允许
4	读取 LockByte(NVR)的值 (有加密块的情况)	不允许	不允许	不允许
5	增加加密块(NVR) (改变 LockByte 的值)	不允许	不允许	不允许
6	减少加密块(NVR) (改变 LockByte 的值)	不允许	不允许	不允许

表 5-1 操作权限关系表

总结:

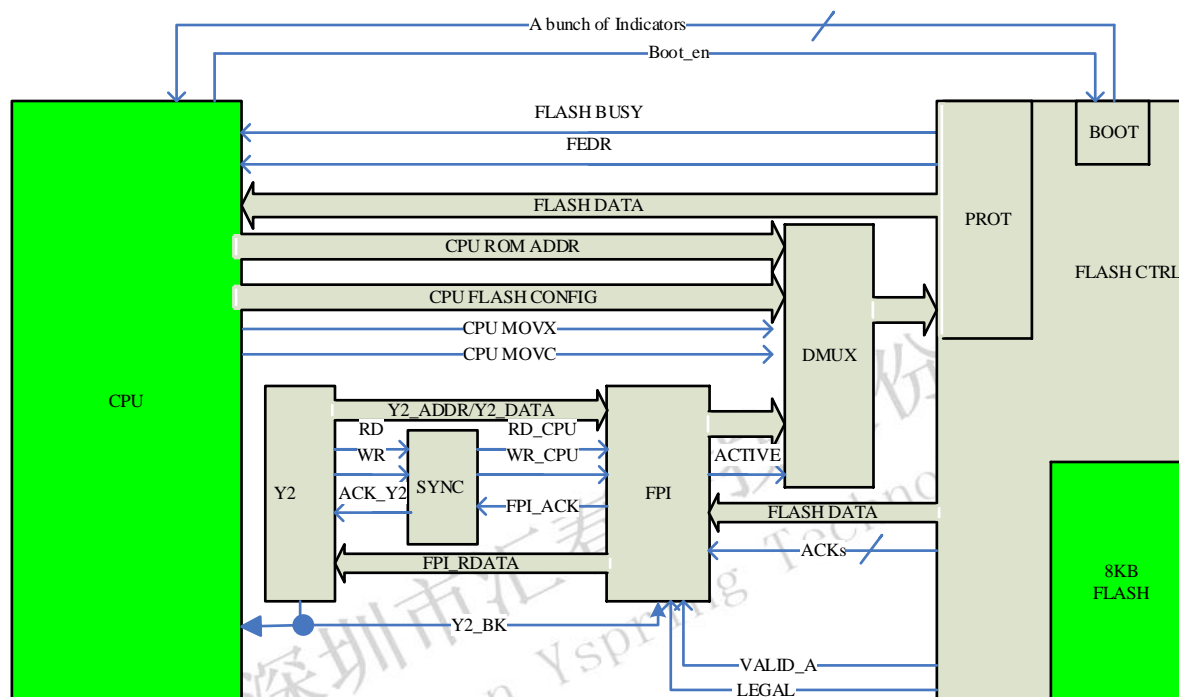
1. LockByte 一旦被写入, 涉及对其的编程、擦除操作都被禁止, 除非执行一次 Y2 所有扇区擦除;
2. 不管加密情况如何, 位于加密块的程序总是可以读取 LockByte 的值;
3. 只要某一块被加密保护, 奇数扇区擦除、奇数扇区编程以及所有扇区编程被禁止;
4. 在 NVR 扇区被选中、或者没有加密块的情况下, 允许偶数扇区擦除和偶数扇区编程;
5. 总是允许所有扇区擦除、所有扇区写。

5.3. LockByte 值和加密块的对应关系

LockByte[7:0]/block	加密地址	LockByte[7:0]/block	加密地址
0x00/NA	0x0000 – 0x000	0x11/16	0x0000 – 0x10ff
0x01/0	0x0000 – 0x00ff	0x12/17	0x0000 – 0x11ff
0x02/1	0x0000 – 0x01ff	0x13/18	0x0000 – 0x12ff
0x03/2	0x0000 – 0x02ff	0x14/19	0x0000 – 0x13ff
0x04/3	0x0000 – 0x03ff	0x15/20	0x0000 – 0x14ff
0x05/4	0x0000 – 0x04ff	0x16/21	0x0000 – 0x15ff
0x06/5	0x0000 – 0x05ff	0x17/22	0x0000 – 0x16ff
0x07/6	0x0000 – 0x06ff	0x18/23	0x0000 – 0x17ff
0x08/7	0x0000 – 0x07ff	0x19/24	0x0000 – 0x18ff
0x09/8	0x0000 – 0x08ff	0x1a/25	0x0000 – 0x19ff
0x0a/9	0x0000 – 0x09ff	0x1b/26	0x0000 – 0x1aff
0x0b/10	0x0000 – 0x0aff	0x1c/27	0x0000 – 0x1bff
0x0c/11	0x0000 – 0x0bff	0x1d/28	0x0000 – 0x1cff

0x0d/12	0x0000 – 0x0cff	0x1e/29	0x0000 – 0x1dff
0x0e/13	0x0000 – 0x0dff	0x1f/30	0x0000 – 0x1eff
0x0f/14	0x0000 – 0x0eff	0x20~0xff/31	0x0000 – 0x1fff
0x10/15	0x0000 – 0x0fff		

YS67FXXXX(X) FLASH CONTROL



图

5-2 Flash 控制器结构框图

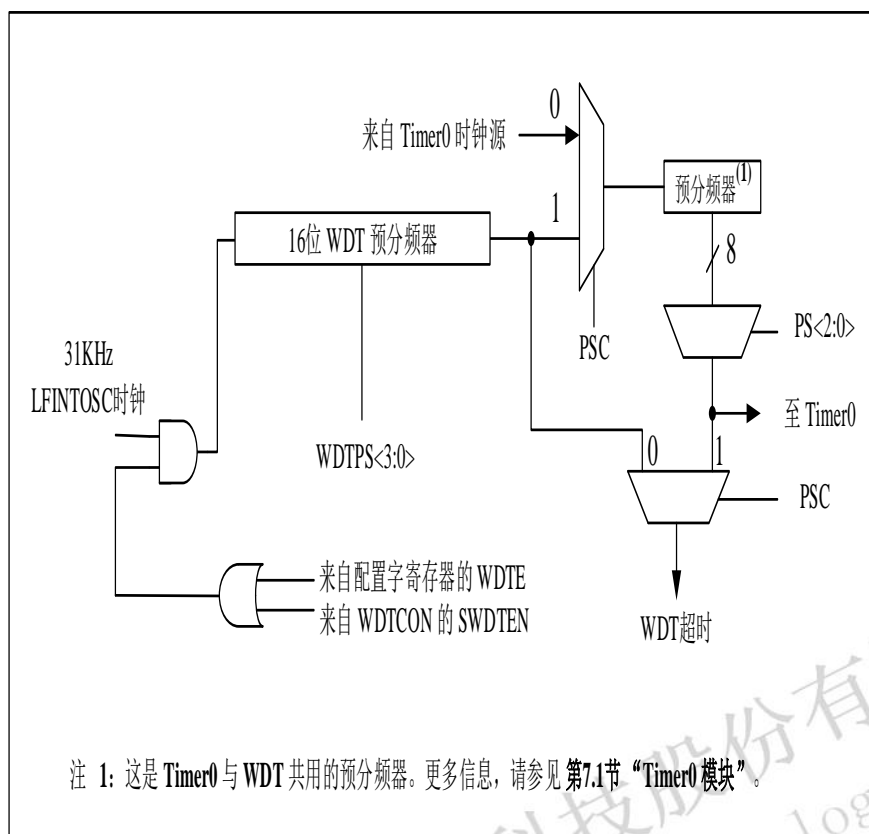


图 5-3 Flash 控制器状态转换图

5.4. 编程步骤

- 禁止中断
- 置 FLA_SEC (PSCTL.1) 和 FLA_WEN (PSCTL.0) 为 1, 以允许软件扇区擦除
- 顺序往 FLKEY 写 0xA5 和 0xF1, 进入开锁状态
- 加入五条 nop 指令
- 用 MOVX 指令向待擦除扇区的某一地址写任意值
- 向 FLA_ACT (PSCTL.2) 写 1, 使 Flash 进入擦除状态
- 清除 FLA_SEC 禁止擦除操作
- 顺序往 FLKEY 写 0xA5 和 0xF1, 重新进入开锁状态
- 加入五条 nop 指令
- 用 MOVX 指令向刚擦除的扇区的目标地址写入数据, 可以重复此步骤直到写满所需字节(最大支持 64B, 不支持跨页编程!)
- 向 FLA_ACT (PSCTL.2) 写 1, 使 Flash 进入编程状态
- 清除 FLA_WEN 禁止 Flash 写
- 重新允许中断

注意:

1、擦除、写操作期间 CPU 停止执行指令, 期间发生的中断也被挂起; 为避免软件跑飞误对 Flash 擦除, 只允许软件对块和扇区的擦除, 而不允许软件对整片 Flash 擦除。

2、要开锁一定要先写 0xA5, 后写 0xF1, 写 0xA5 和 0xF1 中间允许 MCU 其它操作。 “MOVX 编程 Flash” 功能在没冻结的情况下, 任何往 PSCTL 的写操作将使此功能重新上锁; 若不按照这个写顺序或者数据不对, 则此功能将冻结, 直到下一次系统复位!

3、程序过程出现跨页时，硬件会记录在 PSCTL.3。当出现这种情况时，FLASH 控制器将放弃这一操作，故用户程序应检查这一位。

4、在已经开锁的状态不要往 FLKEY 写数据，否则会进入冻结状态。如果要重新上锁，则需要往执行一次 PSCTL 写操作，可以是任何数值。

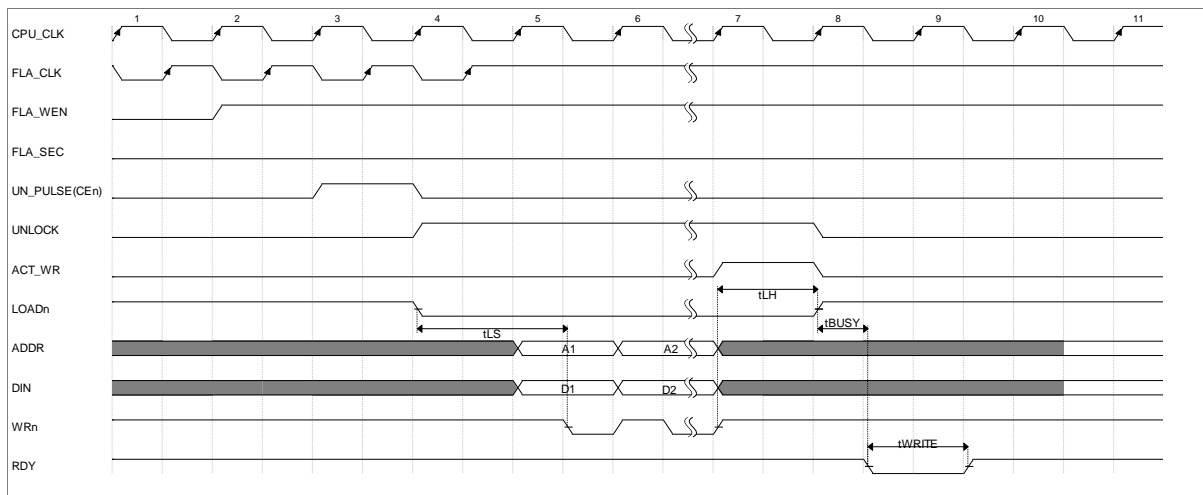


图 5-4 软件编程时序

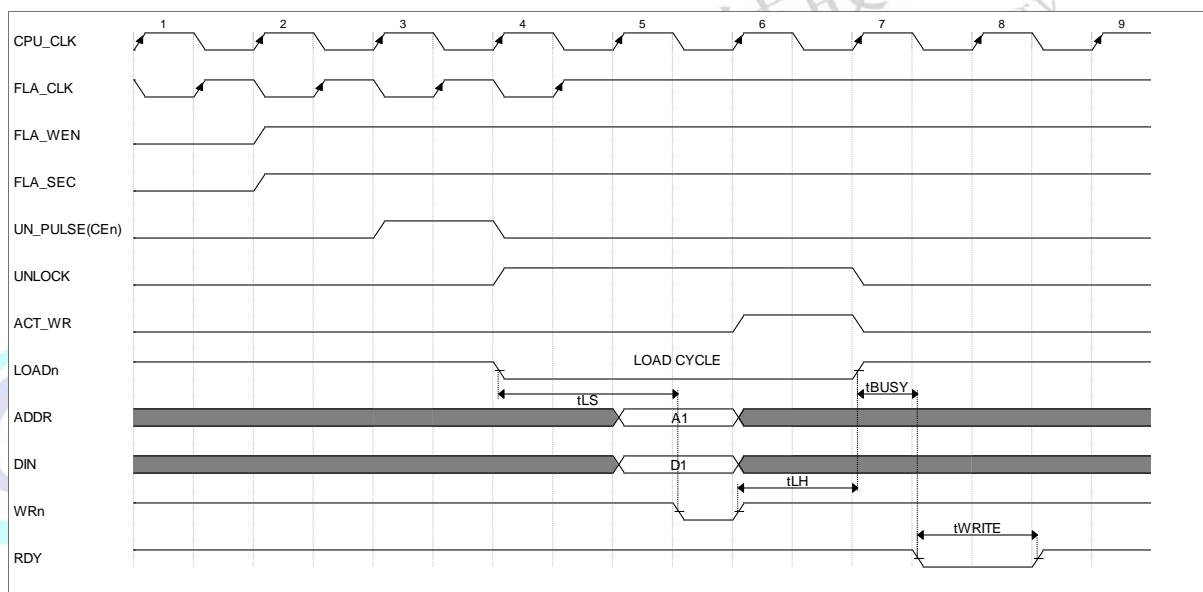


图 5-5 软件擦除时序

6. 功耗模式

有 4 种功耗模式，分别是正常、待机、睡眠和深度睡眠。各种功耗模式下的模块工作情况总结如下：

模式	描 述	唤醒源	功耗性能
正常	除去被关掉的外设，其他模块全速工作，快时钟源可以选择关闭	NA	功耗较高，性能最好。最高达 12MIPS 速度
待机	CPU 停止，其他功能模块关闭或工作，快慢时钟源均打开，Flash 待机	任何中断 看门狗溢出复位 外部/软件复位	功耗低 性能灵活
睡眠	CPU 停止，Flash 处于关机模式，其他功能模块关闭或工作，快时钟关闭（但若当前自容模块在工作时，快时钟不会立即关闭，待自容扫描结束，片内快时钟才会关闭），慢时钟打开	外部/IO 中断 RTC 溢出中断 看门狗溢出复位 触摸相关中断 I2C 中断 外部/软件复位	功耗很低 性能灵活
深度睡眠	快慢时钟关闭（但若当前自容模块或 ADC 在工作时，快慢时钟不会立即关闭，待自容扫描或 ADC 转换结束，片内快慢时钟才会关闭）	I2C 中断 外部/IO 中断 外部/软件复位	功耗最低

表 6-1 功耗模式描述

小结：

- 睡眠模式下，如果选择快时钟作为系统时钟源，Timer1/2/3/4 都将停止工作，所以不能使用这些中断作为唤醒源；如果选择慢时钟作为系统时钟，则以上几个中断将可作为唤醒源；
- 如果想要使用 UART/SPI 通信模块，必须选择快时钟为系统时钟，同时 MCU 处于正常工作或者待机模式。
- 关于中断唤醒，通过配置 WKSRC，外部中断可以在没有使能总中断允许位 IE.7 和相关的独立中断使能位的情况下，唤醒 CPU。其它中断源则必须通过中断才能唤醒。

6.1. 正常模式

正常工作模式下所有功能模块均可处于工作状态下，外设可以通过配置相关 SFR 来禁止、开启，也可以把不用的模块时钟门控掉，节省部分功耗开销。此模式下性能最强，快时钟下指令速度可达到 12MIPS。

6.2. 待机模式

此模式的“优先级”最低，即如果同时往 PCON.4/PCON.1/PCON.0 写 1，MCU 将进入深度睡眠模式，而不是 IDLE 模式。

通过置位 PCON 的最低位（IDLE 位）可使芯片进入此模式，CPU 工作时钟将被门控，其他功能模块的时钟不受此位的影响，它们可以工作或者关闭，取决于它们各自的配置 SFR。

内部振荡器不受影响，它们将保持进入待机模式之前的状态。

任何中断标志的置起均可结束待机模式，中断发生后，PCON.0 由硬件清 0，CPU 从中断向量处取指令执行代码。RETI 返回后执行的是设置 PCON.0 为 1 的下一条指令。

如果进入待机之前使能了看门狗定时器，看门狗计数溢出后复位也会唤醒 CPU，这样可以避免因对 PCON.0 的错误写入而一直处于待机模式。

6.3. 睡眠模式

标准的 80C51 里面，置位 PCON.1 位可使 MCU 进入一个 STOP 模式，该模式下 CPU 和所有的外设如 TIMERS, WDT, UART 等功能模块都将停止工作，而且只能通过外部复位唤醒。

YS67FXXXX(X)的睡眠模式的进入方式跟标准的 80C51 兼容，但外设的工作状态、唤醒方式又有所不同，WDT, RTC 可以工作。如果系统时钟选择了慢时钟，Timers 也可以工作。

外部中断/I2C, RTC 或者是看门狗的溢出复位可以唤醒。

Flash 处于关机模式，典型情况下消耗电流约 1uA。

6.4. 深度睡眠模式

通过置位 PCON.4 可以使 MCU 进入此模式，该模式下所有数字外设的时钟均被门控；内部快慢时钟源关闭，晶振电路不工作。

该模式可以通过外部中断或 I2C 唤醒，唤醒后 PCON.4 被硬件清 0。

Flash 处于关机模式，典型情况下消耗电流约 1uA。

6.5. PCON 寄存器

PCON 地址为 0x87，此寄存器的第 5、4、第 1 和第 0 位用作功耗管理，由于唤醒后硬件自动清 0 的特性，故这 4 位的读出来的值永远是 0。

位	名字	复位值	功 能	
7	NA	NA	保留，只读 0	
6	GF3	0	通用标志位 3	
5	GF2	0	通用标志位 2	
4	DEEPPD	0	写 1 使芯片进入深度睡眠模式，唤醒后由硬件清 0	
3	GF1	0	通用标志位 1	
2	GF0	0	通用标志位 0	
1	STOP	0	写 1 使芯片进入睡眠模式，唤醒后由硬件清 0	
0	IDLE	0	写 1 使芯片进入待机模式，唤醒后由硬件清 0	
{PCON 4, PCON 1:0}			功耗模式	
	1	x	x	深睡眠
	0	1	x	浅睡眠
	0	0	1	待机
	0	0	0	正常

表 6-2 PCON 定义

注意：如果通过调试接口进行低功耗模式的切换，例如由正常模式切换到待机、睡眠或深睡眠的一种，CPU 将不能被中断唤醒，故上位机不应支持对 PCON 的写操作。

6.6. 唤醒时间

从待机模式返回到正常模式，大约需要 400ns；

从睡眠、深度睡眠模式返回到正常模式，大约需要 125us(12MHz 作为系统时钟，若慢时钟为系统时钟，则需要 48ms 左右)，这是因为 IP 需要重新上电。

7. 复位源

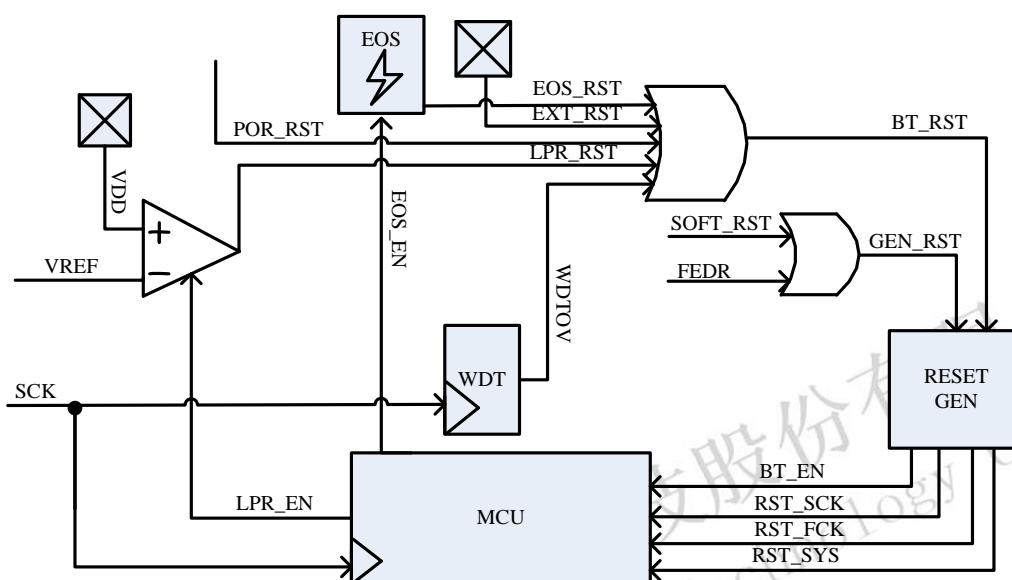


图 7-1 复位源框图

YS67FXXXX(X)有七个复位源，其中外部复位、上电复位、低电压复位、EOS 复位会启动 boot 过程，其他复位则不会，但看门狗复位可配置产生 boot 或者不产生 boot，配置位为 **CCFG3** 寄存器的第 2 位。

复位标志可查询，记录在 0xEF 寄存器里面。最近一次的复位会把相关的位置 1，把其他各位清 0。模拟复位源（外部/上电、低电压、EOS）标志互斥，数字的复位源（WDT、FEDR、SOFT_RST）标志也是互斥的。

假设此前已有模拟复位标志位（外部、上电、EOS 或低电压）被置起，且没有被清 0，数字的复位事件（没有使能 BOOT 的 WDT、FEDR、SOFT_RST）不会把模拟复位标志位清掉。软件可以通过对 RSTEN 的写访问来把模拟复位标志清除。

相反，假设此前有数字复位事件发生，标志位被记录。之后发生了某次模拟复位事件，那么所有数字复位事件标志位会被清 0。

当使能了看门狗溢出引发 BOOT，看门狗复位将被看成模拟复位事件，所以所有模拟复位标志将被清 0，事件记录在 0xEF.3，WDTR。

7.1. 外部复位、上电复位

当 RST 管脚为低超过 20us 时，模拟侦测电路认为这是一次复位事件，把复位信号置为有效，MCU 将启动复位和 boot 过程。

同样，芯片在上电过程中模拟电路也会向把 POR 置起，启动复位和 boot 过程。

7.2. 低电压侦测复位

内置 2 路低压侦测复位电路。一路固定使能，当 VDD 低至理论电压阈值 1.65V 时，监测电路发出 PWR_FAIL，

此时复位产生模块将发出复位信号；另一路在 LVD_PD=0 有效后使能，低压复位电压变为可配，通过 LVD_CAL 可配为 1.8/2.1/2.3/3.8v，同理当 VDD 低至理论电压阈值时，监测电路发出 PWR_FAIL，此时复位产生模块将发出复位信号。

注：当 LVD_PD 使能后模块同时还有低压预警功能，比上面可配电压高 0.2V，即 2.0/2.3/2.5/4，预警信号仅做标志位。

7.3. 看门狗溢出复位

使能看门狗定时器后，如果在其计数溢出之前没有喂狗，计数器溢出之后将会引发系统复位。这个复位源可以避免程序跑飞或者由于错误写入 PCON 值而造成睡眠。看门狗溢出后复位模块将产生系统复位，同时如果看门狗溢出引发 BOOT，溢出事件将通知模拟电路以产生 boot 过程。

7.4. FEDR 复位

Flash 控制器提供了“MOVX 编程、擦除 Flash”功能，如果用该功能访问加密的扇区，或者是保留区，控制器都将发出 Flash 错误操作复位。FEDR 复位源一直使能，不能禁止，详情请参阅“FLASH 安全管理”一节。

7.5. 软复位

软件复位是供片上调试模式使用的，当上位机往芯片写 Soft reset 命令时，经过同步电路处理后将产生系统复位。复位后 CPU 暂停运行，PC 值为 0x0000 地址。

7.6. 过度电应力复位

EOS 是指芯片的电压、电流超出了芯片能承受的范围，当这种情况发生的时候而且 EOS 被使能（RSTEN.1 为 1），EOS 侦测电路会发出复位信号，启动系统复位和 boot 过程。

7.7. 各复位事件对系统的影响

复位事件	受影响的电路
上电复位	数字电路所有模块 清除所有数字复位标志位 引发 BOOT 过程
低电压复位	
外部复位	
EOS 复位	
看门狗复位	除配置寄存器（CFGxx）外所有模块 可配置是否产生 BOOT 过程
软复位	除断点、配置寄存器（CFGxx）、FPI、Y2 及 EMU 外所有模块 不引发 BOOT 过程
FEDR	

表 7-1 RSTSRC 定义

7.8. 复位寄存器

7.8.1 RSTSRC 复位状态寄存器

RSTSRC 地址为 0xEF，只读。通过此寄存器可以了解芯片发生了哪些复位，其各位都是高电平有效。

RSTSRC 地址为 0xEF，页地址为 0x0，只读。通过此寄存器可以了解芯片发生了哪些复位，其各位都是高电平有效。

向第 0 位、第 1 位、第 3 位(使能 BOOT 时)、第 6 位、第 7 位中任何一位写 0 时，会将这 5 位全部清 0。

如果没有使能 BOOT，向第 3 位写 0 时，只能清除第 3 位。此时向第 0、1、6、7 位写 0 时，不会将第 3 位清除。

位	名 字	复位值	功 能
7	EXTR	NA	只读，1 表示发生过外部复位
6	EOSR	NA	只读：1 表示发生了电过度应力事件，EOS_RST_EN=1 时可以指示发生了电过度应力复位
5	VDD_STAT	NA	VDD 状态位
4	FEDR	NA	只读，1 表示此前有 MOVX 的非法操作，引发了复位
3	WDTR	NA	只读：1 表示看门狗溢出引发了复位
2	SOFTTR	NA	只读，1 表示发生过软件复位
1	POR	NA	只读，1 表示发生过上电复位
0	PWR_FAIL	NA	只读：1 表示曾经掉电，引发了复位

表 7-2 RSTSRC 定义



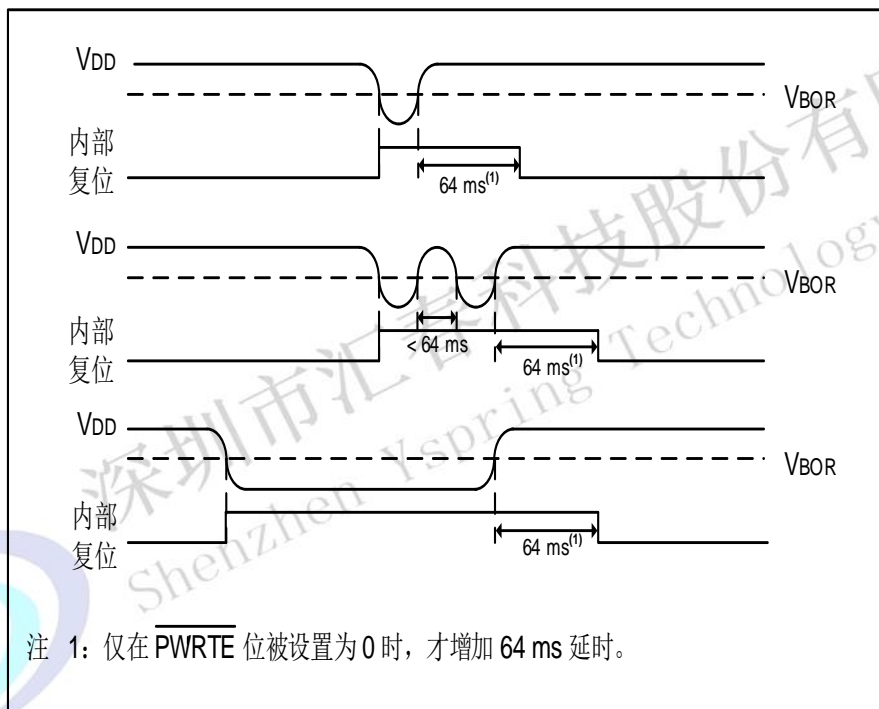
深圳市汇泰科技股份有限公司
Shenzhen Yspring Technology Co., Ltd.

8. 输入/输出端口

- 提供 41 个双向 GPIO
- IO 端口可与其他功能复用
- YS6105 IO 控制模块提供 41 个 GPIO，端口的数据在 PxDR 中,每个 IO 都有内部上拉电阻
- 端口方向控制寄存器 PxDIR 可以设置每个 IO 口为输入/输出。
- 所有的 IO 口都可以把输出模式的电流设置为正常型/增强型
- P3[1:0]可以设置为推挽/OPENDRAIN 模式
- 部分 IO 具有复用功能,并且复用功能可以转移

8.1. IO 等效电路及说明

双向模拟 IO 等效电路图



双向模拟 IO 可用于模拟 IO，也可以用于数字 IO。用做数字 IO 时，可以选择上拉，上拉只在输入状态下有效，输入具有施密特特性。

适用对象：P0[7:0]、P2[7:0]、P4[2:0]、P5[7:0]、P6[7:0]、P3[5:2]

注：芯片全局复位期间，P0、P4、P5、P6 为输出态，输出全为 0。

8.2. 寄存器说明

IO 控制模块设置了以下寄存器：

寄存器名	偏移地址	有效位宽	复位值	说 明
FCTR	0XB9		0X00	功能控制寄存器
P0PU	0XD5		0X00	P0 端口上拉控制寄存器

P2PU	0XDF		0X00	P2 端口上拉控制寄存器
P4PU	0XDE		0X00	P4 端口上拉控制寄存器
P5PU	0XD6		0X00	P5 端口上拉控制寄存器
P6PU	0XC6		0X00	P6 端口上拉控制寄存器
PXC	0XD7		0X00	P3[1:0]开漏控制; 电流驱动模式控制; P3 口上拉控制寄存器
P0AN	0XCD		0X00	P0 端口数字/模拟模式控制寄存器
P2AN	0XCF		0X00	P2 端口数字/模拟模式控制寄存器
P3AN	0XCC		0X00	P3 端口数字/模拟模式控制寄存器
P4AN	0XDD		0X00	P4 端口数字/模拟模式控制寄存器
P5AN	0XCE		0X00	P5 端口数字/模拟模式控制寄存器
P6AN	0XC5		0X00	P6 端口数字/模拟模式控制寄存器
P0DIR	0XD1		0XFF	P0 端口方向控制寄存器
P2DIR	0XD3		0XFF	P2 端口方向控制寄存器
P3DIR	0XD4		0X3F	P3 端口方向控制寄存器
P4DIR	0XC1		0XFF	P4 端口方向控制寄存器
P5DIR	0XC2		0XFF	P5 端口方向控制寄存器
P6DIR	0XC3		0XFF	P6 端口方向控制寄存器
P0DR	0X80		0XFF	P0 数据寄存器
P2DR	0XA0		0XFF	P2 数据寄存器
P3DR	0XB0		0X3F	P3 数据寄存器
P4DR	0XC9		0XFF	P4 数据寄存器
P5DR	0XCA		0XFF	P5 数据寄存器
P6DR	0XCB		0XFF	P6 数据寄存器

8.2.1 功能控制寄存器 FCTR

Bit	7	6	5	4	3	2	1	0
Name	Reserved	UARTOIR	TIMER4FTR	TIMER3FTR	TIMER2FTR	UARTFTR	SPIFTR	I2CFTR
Type	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址：0XB9

Bit	Name	Function
[7]	(Reserved)	保留位，只读，读为 0。
[6]	UARTOIR	用于控制 Uart 在发送完一帧数据后，是否将数据发送端口(TX)自动设置为输入类型。此功能是由于多处理器通讯的情况，如果某个 Uart 从机的 TX 始终发送高电平，则其它从机无法发送数据。 0: 不自动转换端口类型 1: 自动转换端口类型
[5]	TIMER4FTR	控制 TIMER4FTR 功能是否转移。TIMER4 功能默认在 PAD4/P0.3 上，功能转移以后，在 PAD34/P3.1。 0: TIMER4 功能不转移 1: TIMER4 功能转移
[4]	TIMER3FTR	控制 TIMER3FTR 功能是否转移。TIMER3 功能默认在 PAD3/P30.2 上，功能转移以后，在 PAD35/P3.0。 0: TIMER3 功能不转移 1: TIMER3 功能转移
[3]	TIMER2FTR	控制 TIMER2FTR 功能是否转移。TIMER2 功能默认在 PAD5/P0.4 上，功能转移以后，在 PAD33/P3.2。 0: TIMER2 功能不转移 1: TIMER2 功能转移
[2]	UARTFTR	控制 Uart 功能是否转移。Uart 功能默认在 PAD42/P3.2 和 PAD41/P3.3 上，功能转移以后，分别在 PAD4/P0.2 和 PAD5/P0.3 上。 0: Uart 功能不转移 1: Uart 功能转移
[1]	SPIFTR	控制 SPI 功能是否转移。SPI 功能默认在 PAD44/P3.0、PAD43/P3.1、PAD42/P3.2、PAD41/P3.3 上，功能转移以后，分别在 PAD6/P0.4、PAD7/P0.5、PAD8/P0.6、PAD9/P0.7 上。 0: SPI 功能不转移 1: SPI 功能转移
[0]	I2CFTR	控制 I2C 功能是否转移。I2C 功能默认在 PAD43/P3.1、PAD44/P3.0 上，功能转移以后，分别在 PAD2/P0.0 和 PAD3/P0.1 上。 0: I2C 功能不转移 1: I2C 功能转移

8.2.2 P0 端口上拉控制寄存器 P0PU

Bit	7	6	5	4	3	2	1	0
Name	P0PU[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XD5

Bit	Name	Function
[7:0]	P0PU	<p>P0 端口上拉使能位, P0PU[7:0]分别对应 P0[7:0]的上拉使能。</p> <p>0: 上拉禁止</p> <p>1: 上拉使能</p> <p>注: 当 P0 被配置为输入口, 并且相应的上拉控制位为 1 时, 上拉才有效 (即 P0DIR.x & P0PU.x=1), 否则内部上拉不起作用。</p>

8.2.3 P2 端口上拉控制寄存器 P2PU

Bit	7	6	5	4	3	2	1	0
Name	P2PU[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XDF

Bit	Name	Function
[7:0]	P2PU	<p>P2 端口上拉使能位, P2PU[7:0]分别对应 P2[7:0]的上拉使能。</p> <p>0: 上拉禁止</p> <p>1: 上拉使能</p> <p>注: 当 P2 被配置为输入口, 并且相应的上拉控制位为 1 时, 上拉才有效 (即 P2DIR.x & P2PU.x=1), 否则内部上拉不起作用。</p>

8.2.4 P4 端口上拉控制寄存器 P4PU

Bit	7	6	5	4	3	2	1	0
Name	Reserved					P4PU[2:0]		
Type	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XDE

Bit	Name	Function
[7:3]	Reserved	保留位，只读，读为 0。
[2:0]	P4PU	P4 端口上拉使能位，P4PU[2:0]分别对应 P4[2:0]的上拉使能。 0：上拉禁止 1：上拉使能 注：当 P4 被配置为输入口，并且相应的上拉控制位为 1 时，上拉才有效（即 P4DIR.x & P4PU.x=1）， 否则内部上拉不起作用。

8.2.5 P5 端口上拉控制寄存器 P5PU

Bit	7	6	5	4	3	2	1	0
Name	P5PU[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址：0XD6

Bit	Name	Function
[7:0]	P5PU	P5 端口上拉使能位，P5PU[7:0]分别对应 P5[7:0]的上拉使能。 0：上拉禁止 1：上拉使能 注：当 P5 被配置为输入口，并且相应的上拉控制位为 1 时，上拉才有效（即 P5DIR.x & P5PU.x=1）， 否则内部上拉不起作用。

8.2.6 P6 端口上拉控制寄存器 P6PU

Bit	7	6	5	4	3	2	1	0
Name	P6PU[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址：0XC6

Bit	Name	Function
[7:0]	P6PU	P6 端口上拉使能位，P6PU[7:0]分别对应 P6[7:0]的上拉使能。 0：上拉禁止 1：上拉使能 注：当 P6 被配置为输入口，并且相应的上拉控制位为 1 时，上拉才有效（即 P6DIR.x & P6PU.x=1）， 否则内部上拉不起作用。

8.2.7 Px 端口控制寄存器 PXC

Bit	7	6	5	4	3	2	1	0
Name	OD_DRV	HDRV	P3PU[5:0]					
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XD7

Bit	Name	Function
[7]	OD_DRV	P3[1:0]的 Open Drain 模式控制 0: 推挽模式, 此模式下 IO 的输出电流能力由 HDRV 决定 1: Open Drain 模式, 此模式下 IO 将会自动使用增强电流输出模式
[6]	HDRV	IO 输出电流驱动模式, 适用于所有 IO 0: 正常电流输出模式 1: 增强电流输出模式
[5:0]	P3PU [5:0]	P3[5:0]端口上拉使能位, P3PU[5:0]分别对应 P3[5:0]的上拉使能。 0: 上拉禁止 1: 上拉使

8.2.8 P0 模式控制寄存器 P0AN

Bit	7	6	5	4	3	2	1	0
Name	P0AN[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XCD

Bit	Name	Function
[7:0]	P0AN	P0 端口模拟模式控制, P0AN[7:0]分别对应 P0[7:0]的模拟模式控制。 0: 数字模式 1: 模拟模式 注: P0[7:0]端口配置成模拟模式后, 只能用于 LCD 输出。

8.2.9 P2 模式控制寄存器 P2AN

Bit	7	6	5	4	3	2	1	0
-----	---	---	---	---	---	---	---	---

Name	P2AN[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XCF

Bit	Name	Function
[7:0]	P2AN	P2 端口模拟模式控制, P2AN[7:0]分别对应 P2[7:0]的模拟模式控制。 0: 数字模式 1: 模拟模式 注: P2[7:0]配成模拟模式后, 如果其触摸通道的 MASK 为 1 则为触摸通道, 否则为 ADC 的通道。

8.2.10 P3 模式控制寄存器 P3AN

Bit	7	6	5	4	3	2	1	0
Name	Reserved		P3AN[5:4]		Reserved			
Type	R	R	R/W	R/W	R	R	R	R
Reset	0	0	0	0	0	0	0	0

地址: 0XCC

Bit	Name	Function
[7:6]	(Reserved)	保留位, 只读, 读为 0。
[5:4]	P3AN[5:4]	P3 端口模拟模式控制, P3AN[5:4]分别对应 P3[5:4]的模拟模式控制。 0: 数字模式 1: 模拟模式
[3:0]	(Reserved)	保留位, 只读, 读为 0。

8.2.11 P4 模式控制寄存器 P4AN

Bit	7	6	5	4	3	2	1	0
Name	Reserved					P4AN[2:0]		
Type	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址: 0XDD

Bit	Name	Function
-----	------	----------

[7:3]	Reserved	保留位，只读，读为 0。
[2:0]	P4AN	<p>P4 端口模拟模式控制，P4AN[2:0]分别对应 P4[2:0]的模拟模式控制。</p> <p>0: 数字模式</p> <p>1: 模拟模式</p> <p>注：</p> <p>P4[2:0]端口配置成模拟模式后，只能用于 LCD 输出。</p>

8.2.12 P5 模式控制寄存器 P5AN

Bit	7	6	5	4	3	2	1	0
Name	P5AN[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址：0XCCE

Bit	Name	Function
[7:0]	P5AN	<p>P5 端口模拟模式控制，P5AN[7:0]分别对应 P5[7:0]的模拟模式控制。</p> <p>0: 数字模式</p> <p>1: 模拟模式</p> <p>注：P5[7:0]端口配置成模拟模式后，只能用于 LCD 输出。</p>

8.2.13 P6 模式控制寄存器 P6AN

Bit	7	6	5	4	3	2	1	0
Name	P6AN[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

地址：0XC5

Bit	Name	Function
[7:0]	P6AN	<p>P6 端口模拟模式控制，P6AN[7:0]分别对应 P6[7:0]的模拟模式控制。</p> <p>0: 数字模式</p> <p>1: 模拟模式</p> <p>注：P6[7:0]端口配置成模拟模式后，只能用于 LCD 输出。</p>

8.2.14 P0 端口方向控制寄存器 P0DIR

Bit	7	6	5	4	3	2	1	0
-----	---	---	---	---	---	---	---	---

Name	P0DIR[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

地址: 0XD1

Bit	Name	Function
[7:0]	P0DIR	P0 端口数字输入、输出方向控制位, P0DIR[7:0]分别对应 P0[7:0]的方向控制。 0: 输出 1: 输入

8.2.15 P2 端口方向控制寄存器 P2DIR

Bit	7	6	5	4	3	2	1	0
Name	P2DIR[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

地址: 0XD3

Bit	Name	Function
[7:0]	P2DIR	P2 端口数字输入、输出方向控制位, P2DIR[7:0]分别对应 P2[7:0]的方向控制。 0: 输出 1: 输入

8.2.16 P3 端口方向控制寄存器 P3DIR

Bit	7	6	5	4	3	2	1	0
Name	Reserved		P3DIR[5:0]					
Type	只读, 读为 0。		R/W	R/W	R/W	R/W	R/W	R/W
Reset	0		1	1	1	1	1	1

地址: 0XD4

Bit	Name	Function
[7:6]	(Reserved)	保留位, 只读, 读为 0。
[5:0]	P3DIR	P3 端口数字输入、输出方向控制位, P3DIR[5:0]分别对应 P3[5:0]的方向控制。 0: 输出 1: 输入

8.2.17 P4 端口方向控制寄存器 P4DIR

Bit	7	6	5	4	3	2	1	0
Name	Reserved					P4DIR[2:0]		
Type	R	R	R	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	1	1	1

地址: 0XC1

Bit	Name	Function
[7:3]	Reserved	保留位, 只读, 读为 0。
[2:0]	P4DIR	P4 端口数字输入、输出方向控制位, P4DIR[2:0]分别对应 P4[2:0]的方向控制。 0: 输出 1: 输入

8.2.18 P5 端口方向控制寄存器 P5DIR

Bit	7	6	5	4	3	2	1	0
Name	P5DIR[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

地址: 0XC2

Bit	Name	Function
[7:0]	P5DIR	P5 端口数字输入、输出方向控制位, P5DIR[7:0]分别对应 P5[7:0]的方向控制。 0: 输出 1: 输入

8.2.19 P6 端口方向控制寄存器 P6DIR

Bit	7	6	5	4	3	2	1	0
Name	P6DIR[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

地址: 0XC3

Bit	Name	Function
-----	------	----------

[7:0]	P6DIR	P6 端口数字输入、输出方向控制位，P6DIR[7:0]分别对应 P6[7:0]的方向控制。 0: 输出 1: 输入
-------	-------	--

8.2.20 P0 数据寄存器 P0DR

Bit	7	6	5	4	3	2	1	0
Name	P0DR[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

地址: 0x80

Bit	Name	Function
[7:0]	P0DR	端口 0 数据寄存器。 将端口 0 设置为通用输入端口时，此寄存器保存端口 P0[7:0]上的输入数据；将端口 0 设置为通用输出端口时，向此寄存器写入数据，则数据会从 P0[7:0]输出。

8.2.21 P2 数据寄存器 P2DR

Bit	7	6	5	4	3	2	1	0
Name	P2DR[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

地址: 0xA0

Bit	Name	Function
[7:0]	P2DR	端口 2 数据寄存器。 将端口 2 设置为通用输入端口时，此寄存器保存端口 P2[7:0]上的输入数据；将端口 2 设置为通用输出端口时，向此寄存器写入数据，则数据会从 P2[7:0]输出。

8.2.22 P3 数据寄存器 P3DR

Bit	7	6	5	4	3	2	1	0
Name	Reserved		P3DR[5:0]					
Type	只读，读为 0。		R/W					
Reset	0		1	1	1	1	1	1

地址: 0xB0

Bit	Name	Function
[7:6]	(Reserved)	保留位, 只读, 读为 0。
[5:0]	P3DR	端口 3 数据寄存器。 将端口 3 设置为通用输入端口时, 此寄存器保存端口 P3[5:0]上的输入数据; 将端口 3 设置为通用输出端口时, 向此寄存器写入数据, 则数据会从 P3[5:0]输出。

8.2.23 P4 数据寄存器 P4DR

Bit	7	6	5	4	3	2	1	0
Name	Reserved					P4DR[2:0]		
Type	R					R/W		
Reset	0	0	0	0	0	1	1	1

地址: 0XC9

Bit	Name	Function
[7:3]	Reserved	保留位, 只读, 读为 0。
[2:0]	P4DR	端口 4 数据寄存器。 将端口 4 设置为通用输入端口时, 此寄存器保存端口 P4[2:0]上的输入数据; 将端口 4 设置为通用输出端口时, 向此寄存器写入数据, 则数据会从 P4[2:0]输出。

8.2.24 P5 数据寄存器 P5DR

Bit	7	6	5	4	3	2	1	0
Name	P5DR[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

地址: 0XCA

Bit	Name	Function
[7:0]	P5DR	端口 5 数据寄存器。 将端口 5 设置为通用输入端口时, 此寄存器保存端口 P5[7:0]上的输入数据; 将端口 5 设置为通用输出端口时, 向此寄存器写入数据, 则数据会从 P5[7:0]输出。

8.2.25 P6 数据寄存器 P6DR

Bit	7	6	5	4	3	2	1	0
Name	P6DR[7:0]							
Type	R/W							
Reset	1	1	1	1	1	1	1	1

地址：0XCB

Bit	Name	Function
[7:0]	P6DR	端口 6 数据寄存器。 将端口 6 设置为通用输入端口时，此寄存器保存端口 P6[7:0]上的输入数据；将端口 6 设置为通用输出端口时，向此寄存器写入数据，则数据会从 P6[7:0]输出。



深圳市汇春科技股份有限公司
Shenzhen Yspring Technology Co., Ltd.

9. 中断系统

YS67FXXXX(X)在标准 80C51 基础上对中断源进行了扩展，支持多个中断源和两个优先级。其中高优先级中断可以打断正在执行的低优先级中断，实现了中断嵌套；同级别的中断不能相互打断。

所有中断均拥有相应的标志位，当标志位为 1 时并且对应的中断位被允许以及中断总开关 EA 为 1 时，CPU 在执行完当前指令后产生一条 LCALL 跑到中断向量处取指，同时当前 PC 入栈，PSW、ACC 的值要靠 PUSH 指令压栈。每一个中断处理程序都必须以 RETI 结束。

要注意的是退出中断处理程序前必须确保中断标志位已经被清 0，否则在 CPU 执行完返回后的第一条指令后又再次进入该中断。

硬件对中断标志位的清除不尽相同，有的中断由硬件自动清 0，有的则依靠用户清 0，具体内容请看“中断源”一节。

9.1. 中断源

一共有 13 个中断源，优先级和向量地址关系如下表：

中断源	向量地址	默认优先级	标志位	是否硬件清标志位	中断允许控制	优先级控制
复位	0x0000	最高	NA	NA	一直允许	最高
外部中断 Int0	0x0003	0	IE0(TCON.1)	是	IE.0	IP.0
Reserved	0x000b	1	NA	NA	NA	NA
IO 变化中断	0x0013	2	PI(TCON.3)	否	IE.2	IP.2
Timer1 溢出	0x001b	3	TF1(TMOD.1)	是	IE.3	IP.3
UART	0x0023	4	RI(SCON.0) TI(SCON.1)	否	IE.4	IP.4
SPI	0x002b	5	SPI_CNTL.7 SPI_CNTL.6 SPI_CNTL.5 SPI_CNTL.4	否	IE.5	IP.5
ADC	0x0033	6	ADC0CN.5	否	IE.6	IP.6
Timer2 溢出	0x003b	7	TMR2CN.7 TMR2CN.6	否	EIE1.0	EIP1.0
Reserved	0x0043	8	NA	NA	NA	NA
Timer3 溢出	0x004b	9	TMR3CN.7 TMR3CN.6	否	EIE1.2	EIP1.2
RTC	0x0053	10	IRQ0.3	否	EIE1.3	EIP1.3
Timer4 溢出	0x005b	11	TMR4CN.7 TMR4CN.6	否	EIE1.4	EIP1.4
CS	0x0063	12	IRQ0.5	否	EIE1.5	EIP1.5
I2C	0x006b	13	IRQ0.6	否	EIE1.6	EIP1.6
MCD	0x0073	14	IRQ0.7	否	EIE1.7	EIP1.7

表 10-1 中断源及其向量地址

注意:

1. 除了外部中断、定时器 1 中断标志可以由硬件清 0 外, 其它 11 个中断标志位必须由软件清 0, 方法是往相应标志位写 0。
2. 除了 INT0 脚中断和 P0 端口变化中断外, 软件向相关中断标志位置 1 时可引发中断(IE.7=1 和相应的中断允许情况下)。

9.2. 关于 IO 电平变化中断

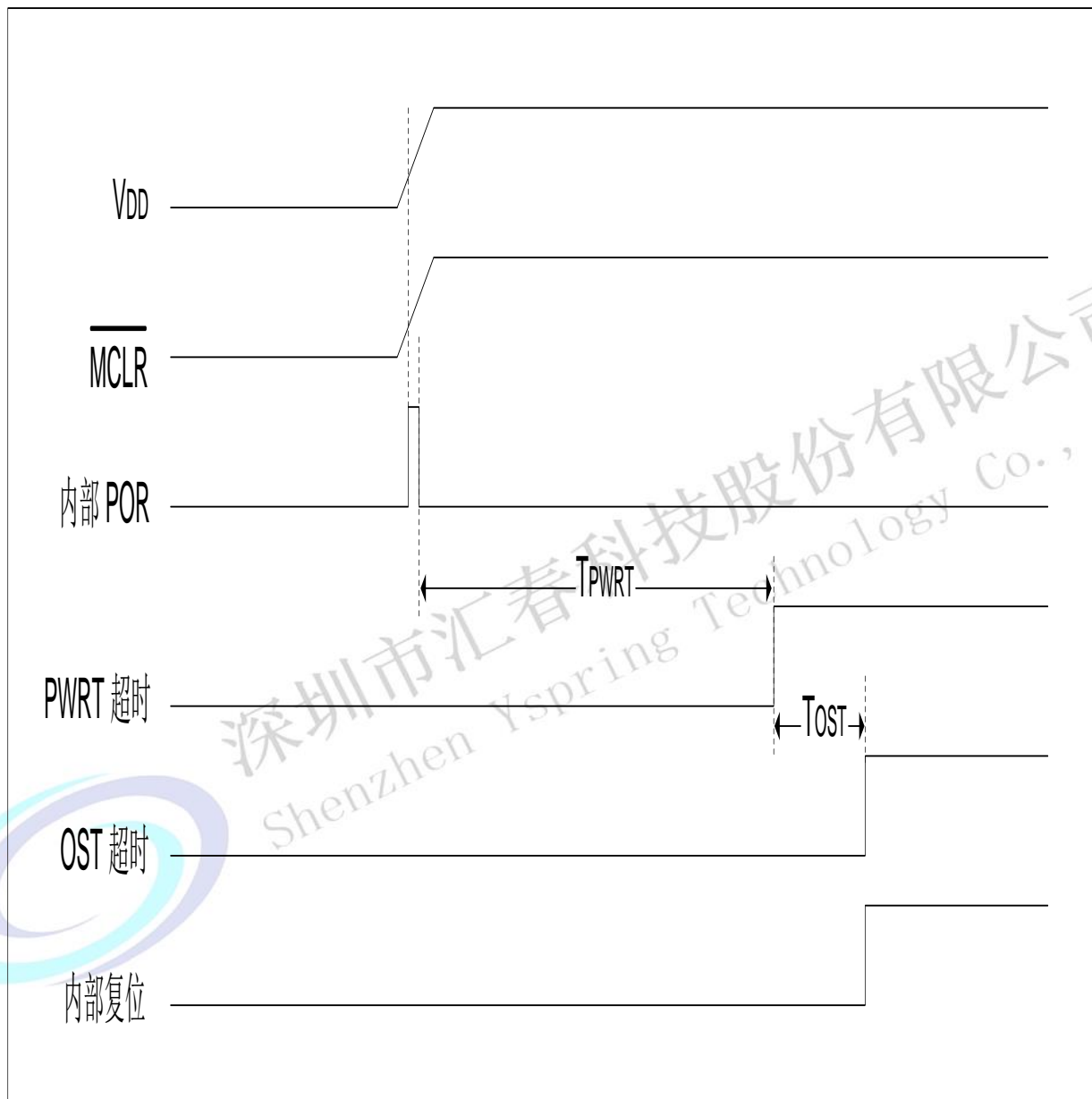


图 10-3 IO 电平变化中断电路原理 (P0.0~P0.3)

YS67FXXXX(X)的 P0.0~P0.7 一共 8 个管脚口具有电平变化中断功能, 由上述原理图可知, 使用它的条件是:

- P0 相关管脚处于数字输入状态;
- 软件读取 P0 口, 把值锁存到后级寄存器;
- 置 POSEL 的相关位;

同样可以看到, 满足以下条件之一才能清端口变化中断标志:

- 外部端口恢复原来的电平;

- 软件重新读取一下端口，刷新寄存器值；
- 满足以上条件之一后，向 TCON.3 写 0；

9.3. 缺失时钟中断（MCD）

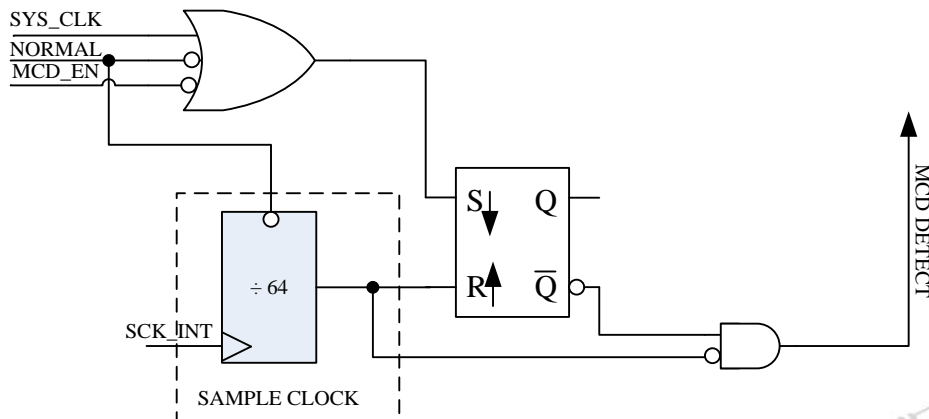


图 10-4 MCD 原理框图

CLKCF.2 (MCD_EN) 被置为 1 后将使能缺失时钟侦测，如果芯片工作在正常模式 (NORMAL) 下，并且选择为系统时钟的外部时钟缺失超过 1~2ms (这个时间取决于时钟缺失下降沿与采样时钟上升沿的位置)，侦测电路就发出 MCD 有效信号，故障保护条件有效，启动内部时钟（快或者慢取决 CLKCF 的 CKSEL 位），但不更新 CKSEL 位和 SYS_IND 位。如果相应的中断允许和 IE.7 打开，CPU 将执行 MCD 中断程序。

注意：

1、MCD 模块只监测外部时钟，所以只有当系统时钟选择为外部时钟时它才工作。它的采样时钟是内部慢时钟，64 分频后周期大约是 2ms。

3、当 MCD 事件发生后，故障保护条件可以由以下事件清除：

- 产生了系统复位；
- 软件改变了 CLKCF 的第 4、5 位，具体是：
 - 当 CKSEL 为 1 时，CLKCF 第 4 位由高变低；
 - 当 CKSEL 为 0 时，CLKCF 第 5 位由高变低；

4、软件清除 IRQ0.7 的前提是故障保护条件清除，然后由软件往 IRQ0.7 写 0。

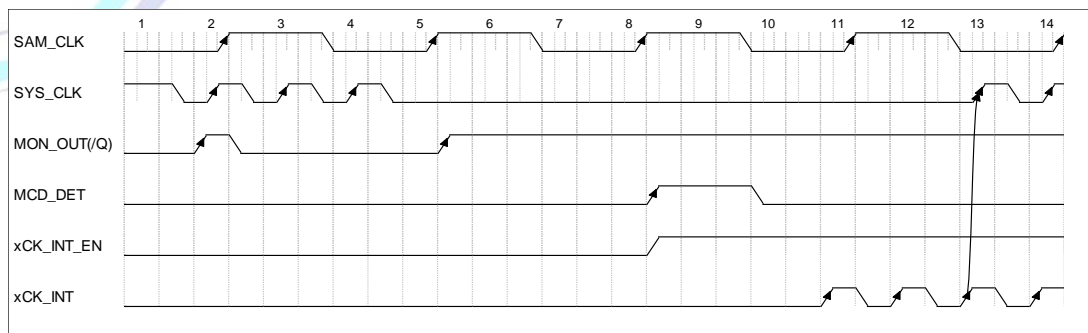


图 10-5 MCD 时序图

寄存器 MCDST 用来指示是否发生了时钟缺失，地址为 0xC7：

位	名字	复位值	功 能
7:1	Reserved	NA	保留位，读 0，写无效
0	MCDST	0	时钟缺失状态 1：发生时钟缺失

			0: 未发生时钟缺失 注: 只有当时钟缺失条件消失后, 软件才能把它清 0
--	--	--	---

9.4. 中断相关寄存器

9.4.1 寄存器 IE (SFR 地址 0xA8)

位	名字	复位值	功 能
7	EA	0	中断总开关。 1: 打开 0: 禁止所有中断
6	EADC	0	ADC 中断允许
5	ESPI	0	SPI 中断允许
4	ES0	0	UART 中断允许
3	ET1	0	Timer1 中断允许
2	EP0	0	P0 端口中断允许
1	Reserved	NA	保留位, 读 0, 写无效
0	EX0	0	外部 (Int0) 中断允许

9.4.2 扩展中断允许寄存器 EIE1 (SFR 地址 0xE8)

位	名字	复位值	功 能
7	EMCD	0	MCD 中断允许
6	EI2C	0	I2C 中断允许
5	ECS	0	自容中断允许
4	ET4	0	T4 中断允许
3	ERTC	0	RTC 中断允许
2	ET3	0	定时器 3 中断允许
1	Reserved	NA	保留位, 读 0, 写无效
0	ET2	0	定时器 2 中断允许

9.4.3 中断优先级控制 IP (SFR 地址 0xB8)

通过此寄存器可以改变中断的优先级关系。某一位设置为 1, 表示该位对应的中断拥有高优先级, 否则按默认优先级。

位	名字	复位值	功 能
7:6	NA	NA	保留位, 读=0, 写忽略
5	PSPI	0	SPI 优先级
4	PS0	0	UART 中断优先级
3	PT1	0	Timer1 中断优先级
2	PP0	0	P0 端口中断优先级
1	Reserved	NA	保留位, 读 0, 写无效
0	PX0	0	外部 (Int0) 中断优先级

9.4.4 扩展中断优先级控制寄存器 EIP1 (SFR 地址 0xD8)

位	名字	复位值	功 能
7	PMCD	0	MCD 优先级
6	PI2C	0	I2C 优先级
5	PCS	0	自容中断优先级
4	PT4	0	定时器 4 中断优先级
3	PRTC	0	RTC 中断优先级
2	PT3	0	定时器 3 中断优先级
1	Reserved	NA	保留位，读 0，写无效
0	PT2	0	定时器 2 中断优先级

表 10-2 EIP1 寄存器说明

关于中断优先级：

1. 如果没有手动设置优先级，中断嵌套不会发生；默认的中断优先级（Priority Within Level，“同级优先级”）是当多个中断来时处理的顺序，中断处理过程中它们都被视作同一级，所以就不存在中断被打断、挂起的情况。
2. 当设置了中断优先级（IP/EIP），多个中断一齐到来时，优先级高的中断会先得到处理；高优先级的中断可以打断当前中断。

9.4.5 中断标志位寄存器 IRQ0 (SFR 地址 0xC0)

位	名字	复位值	功 能
7	MCD_IRQ	0	MCD 中断标志 只有当时钟缺失条件消失后，软件才能把它清 0
6	I2C_IRQ	0	I2C 中断标志。 只能由软件清 0，往此位写 0 即可
5	CS_IRQ	0	自容触摸中断标志。 只能由软件清 0，往此位写 0 即可
4	NA	0	读 0，写无效
3	RTC_IRQ	0	RTC 中断标志。 只能由软件清 0，往此位写 0 即可
2	NA	0	读 0，写无效
1	NA	0	读 0，写无效
0	NA	0	读 0，写无效

9.4.6 P0 沿中断引脚选择寄存器 P0SEL (SFR 地址 0x84)

位	名字	复位值	功 能
7	P0.7_SEL	0	1: 选择 P0.7 为 IO 沿中断触发源 0: 禁止 P0.7 作为 IO 沿中断触发源

6	P0.6_SEL	0	1: 选择 P0.6 为 IO 沿中断触发源 0: 禁止 P0.6 作为 IO 沿中断触发源
5	P0.5_SEL	0	1: 选择 P0.5 为 IO 沿中断触发源 0: 禁止 P0.5 作为 IO 沿中断触发源
4	P0.4_SEL	0	1: 选择 P0.4 为 IO 沿中断触发源 0: 禁止 P0.4 作为 IO 沿中断触发源
3	P0.3_SEL	0	1: 选择 P0.3 为 IO 沿中断触发源 0: 禁止 P0.3 作为 IO 沿中断触发源
2	P0.2_SEL	0	1: 选择 P0.2 为 IO 沿中断触发源 0: 禁止 P0.2 作为 IO 沿中断触发源
1	P0.1_SEL	0	1: 选择 P0.1 为 IO 沿中断触发源 0: 禁止 P0.1 作为 IO 沿中断触发源
0	P0.0_SEL	0	1: 选择 P0.0 为 IO 沿中断触发源 0: 禁止 P0.0 作为 IO 沿中断触发源

YS67FXXXX(X)的 P0.0~P0.7 一共 8 个管脚口具有电平变化中断功能，使用它的条件是：

- P0 相关管脚处于数字输入状态；
- 软件读取 P0 口，把值锁存到后级寄存器；
- 置 P0SEL 的相关位；

同样可以看到，满足以下条件之一才能清端口变化中断标志：

- 外部端口恢复原来的电平；
- 软件重新读取一下端口，刷新寄存器值；
- 满足以上条件之一后，向 TCON.3 写 0；

9.4.7 定时器控制寄存器 TCON (SFR 地址 0x88)

位	名字	复位值	功 能
7:4	NA	NA	保留位，只读 0
3	IE1	0	P0 端口变化中断标志 硬件检测到 P0 口有电平变化事件时，此位被置起。只能用软件清 0
2	NA	NA	保留位，只读 0
1	IE0	0	引脚 Int0 中断标志 硬件检测到 Int0 引脚有由 IT0 设置的沿事件时，此位被置起。 CPU 处理中断时硬件自动清 0，也可以用软件清 0
0	IT0	0	Int0 引脚的上升下降沿检测选择，为 1 是选择上升沿，为 0 时选择下降沿

9.4.8 唤醒源配置寄存器 WKSRC (SFR 地址 0x85)

位	名字	复位值	功 能
7:2	NA	NA	保留位，只读 0
1	INT1WK	0	读写 0: 只有在打开相关中断位的情况下，P0 端口变化中断源才能唤醒 MCU 1: 在没有使能相关中断位的情况下，P0 端口变化中断源可以唤醒 MCU
0	INT0WK	0	读写 0: 只有在打开相关中断位的情况下，INT0 沿中断源才能唤醒 MCU 1: 在没有使能相关中断位的情况下，INT0 沿中断源可以唤醒 MCU

注意:

如果要使用非中断唤醒 CPU 的低功耗模式 (包括 IDLE/STOP/DEPD)，软件必须在写完 PCON 的指令后面紧跟着三条 NOP 指令，否则程序会跑飞。



深圳市汇春科技股份有限公司
Shenzhen Yspring Technology Co., Ltd.

10. 系统时钟

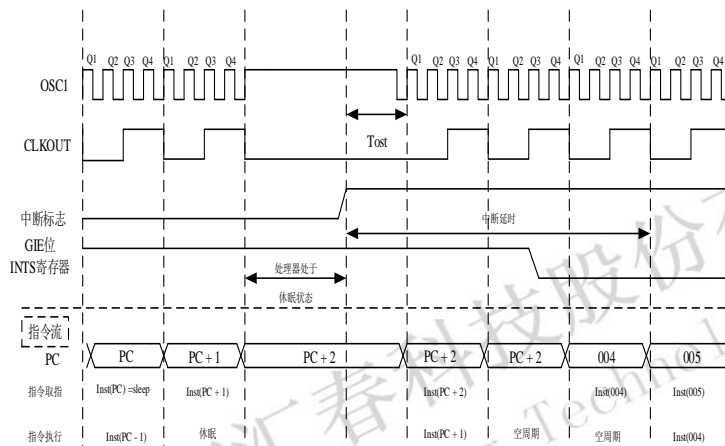
YS67FXXXX(X) CLKC 模块主要用于实现对系统时钟的控制，包括内部外部时钟切换，快慢时钟切换，晶体震荡模式下的时钟缺失检测模式，时钟门控等功能。

YS67FXXXX(X)内部有两个时钟源，一个 12MHz 可编程精密快时钟振荡器和一个 32.768KHz 慢时钟振荡器，系统时钟可以是内部时钟，也可以是外部时钟。通过写 CLKCF.0，快、慢时钟的切换，写 CLKCF[5:4]可实现内外时钟的切换。系统复位后默认选择内部快时钟，外部快慢时钟分为晶体振荡模式和外灌模式，

YS67FXXXX(X)对系统时钟具有时钟缺失检测模式，用于检测当系统工作于外部晶振模式下发生时钟缺失故障时候，产生中断。

时钟源框图如图 11-1 示，系统时钟可以是内部时钟，也可以是外部时钟。系统复位后默认选择内部快时钟，boot 完成后由 CLKCF 决定，它可以由软件改写。

振荡电路还支持外部快、慢时钟的直接灌入，ESCK_EC，EFCK_EC 模式。



注：中断延时在GIE=1处理器唤醒后，将调用004H的ISR程序，如果GIE=0时，程序将继续执行。

图 11-1 时钟源

10.1. CLKCF 寄存器

CLKCF 地址为 0xEA。通过它可实现内外时钟和系统时钟的选择。

位	名字	复位值	功 能
7	NA	NA	保留位，读 0
6	CK12M	0	内部快时钟使能，低有效 0：启用内部快时钟 1：禁用内部快时钟，在正常或待机模式下，如果选择了内部快时钟作为系统时钟（即 CKSEL=0 并 FCK_EXT_EN=0），内部快时钟将强制开启，这样做是为了避免用户在切换系统时钟时忘记清零 CK12M 而导致的死机
5	FCK_EXT_EN	0	12MHz 快时钟源选择 1：选择外部快时钟 0：选择内部快时钟

4	SCK_EXT_EN	0	32KHz 慢时钟源选择 1: 选择外部慢时钟 0: 选择内部慢时钟
3			
2	MCD_EN	0	MCD 模块控制 1 使能 MCD 模块, MCD 事件发生后发出中断请求 0 禁止
1	SYS_IND	0	只读, 系统时钟指示位 内、外时钟切换完成与否的标志 0 表示系统时钟是内部时钟 1 表示系统时钟是外部时钟
0	CKSEL	0	系统时钟选择 读写: 0 选择快时钟, 1 选择慢时钟

表 11-1 CLKCF 定义

10.2. 时钟切换流程

时钟切换可以划分为两大类：一是快时钟和慢时钟的切换，二是内部时钟和外部时钟的切换。支持各种快慢、内外时钟之间的切换，内部设计保证了这个操作的安全性，不会导致意外死机。

下面举例从内部快时钟切换到外部慢时钟：

```

...           ;外部管脚为时钟输入
...           ;外部灌入慢时钟
MOV A, CLKCF  ;暂存 CLKCF
ORL A, 0x10    ;选择外部慢时钟
ORL A, 0x01    ;把 CKSEL 置 1
MOV CLKCF, A  ;进行内部快时钟到外部慢时钟的切换

```

10.3. 快慢切换

由快切换到慢：

1. 使能慢时钟（如果使用外部慢时钟，则置相关引脚为时钟输入）；
2. 切换到慢时钟（CKSEL=1）；
3. 根据需要，可关闭快时钟，以节省功耗；（如果当前使用内部快时钟，置 CK12M=1；如果使用外部快时钟，则停止时钟输入）。

由慢切换到快：

1. 使能快时钟（如果使用内部快时钟，置 CK12M=0；如果使用外部快时钟，则置相关引脚为时钟输入）；
2. 切换到快时钟（CKSEL=0）。

10.4. 由外部时钟切换到内部时钟

1. 使能内部时钟，CK12M=0；
2. 切换到内部时钟，FCK_EXT_EN/SCK_EXT_EN=0；

- 过了两个外部时钟下降沿之后，系统时钟切换到内部时钟；
 - 停止外部时钟输入；（可选）
- 由内部时钟切换到外部时钟

- 设置外部引脚为时钟输入模式，灌入相应频率的时钟；
- 切换外部时钟，FCK_EXT_EN/SCK_EXT_EN=1；
- 关闭内部快时钟，CK12M=1；（可选）

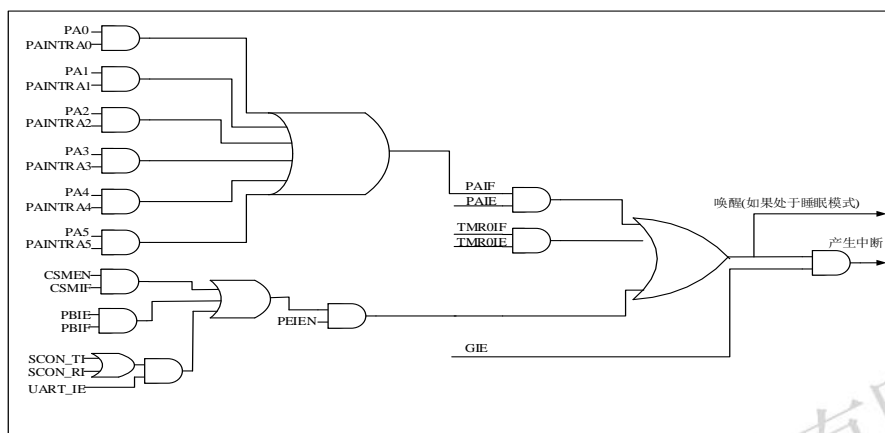


图 11-3 切换到外部时钟检测

10.5. 切换到外部时钟注意

如果需要确保程序段一定要工作在外部时钟，可以通过判断 SYS_IND (CLKCF.1 位)，它为 1 时表示已经切换到外部时钟。

时钟分配和控制由模块 CLKCF 实现，主要涉及到 CPU、中断控制器及外设自身。所有外设由 CLKCF 统一分配。它有以下特点：

当 CPU 处于非正常模式，CPU 时钟就会被停止；

其它外设则取决于低功耗模式和该外设本身的配置、工作状态；

处于在线仿真的暂停状态时，所有 timer 的时钟停止，RTC 时钟停止；LCD/LED、UART（由于 TIMER1 的停止，UART 通信受影响）、ADC、SPI 不受影响。

10.6. 外设的时钟门控

时钟分配和控制由模块 CLKC 实现，主要涉及到 CPU、中断控制器及外设自身。所有外设由 CLKC 统一分配。它有以下特点：

当 CPU 处于非正常模式，CPU 时钟就会被停止；

其它外设则取决于低功耗模式和该外设本身的配置、工作状态；

处于在线仿真的暂停状态时，所有 timer 的时钟停止，RTC 时钟停止；LCD/LED、UART（由于 TIMER1 的停止，UART 通信受影响）、ADC、SPI 不受影响。

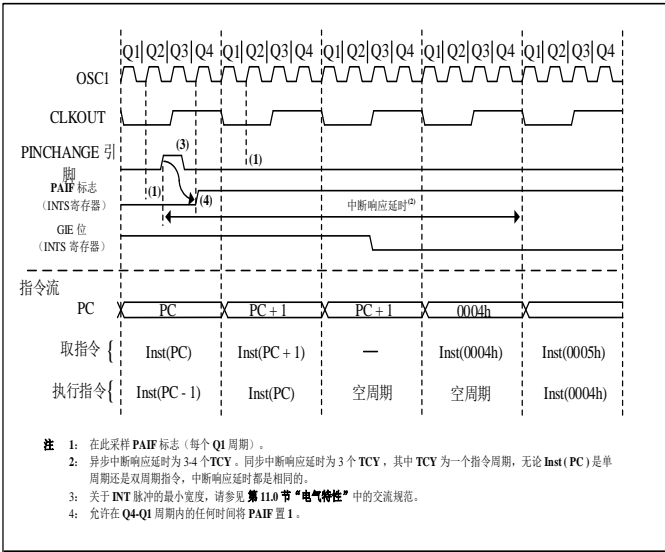


图 11-4 时钟控制功能框图

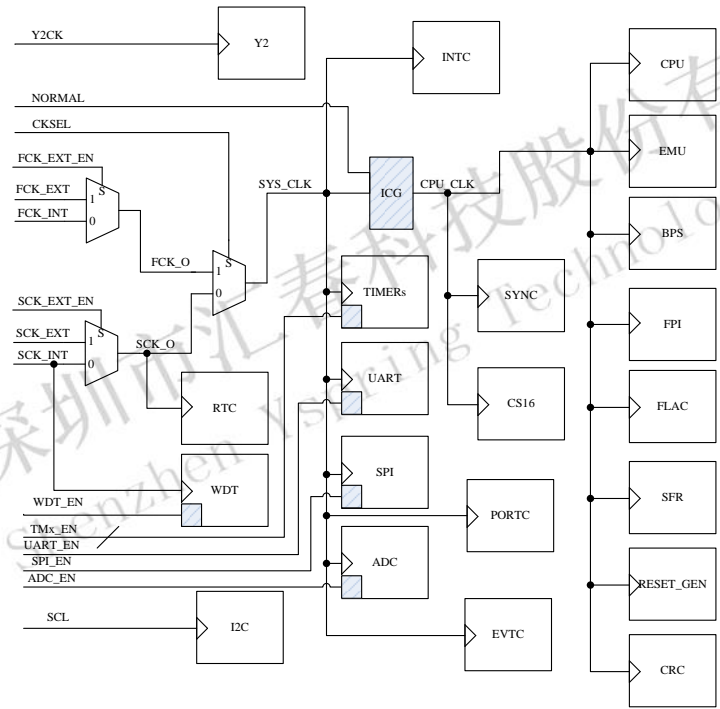


图 11-5 时钟分配框图

11. RTC

11.1. 概述

RTC 模块是基于一个被 32768Hz 时钟驱动的 16bit 减计数器，用户配置好初始时间寄存器，使能后 RTC 开始计时，计时溢出后上报中断，并重新装载继续计时。

11.2. 功能列表

- 16 位减计数器，32768 时钟驱动，可实现最长 2 秒钟精确计时。
- 可使能中断，上报中断至 MCU
- 计数过程中可读取动态计数值
- MCU 休眠（深睡除外）条件下，RTC 仍可正常工作

11.3. 功能框图

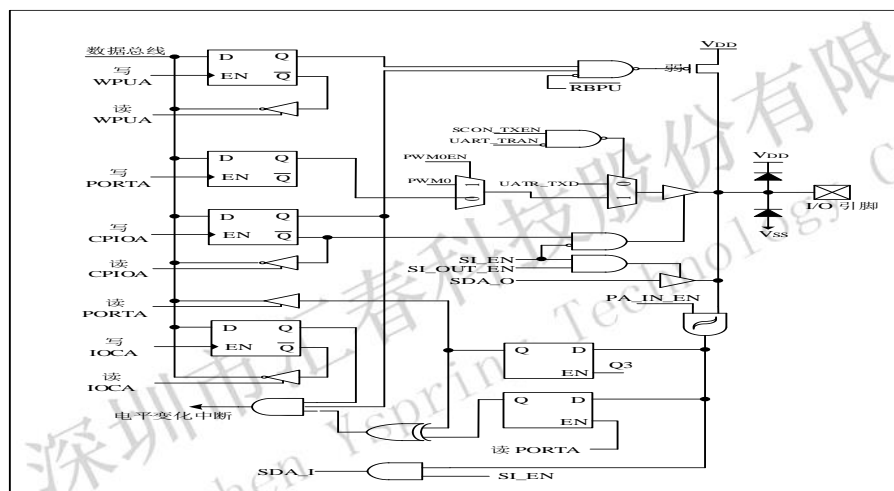


图 12-1 RTC 功能框图

11.4. 功能描述

RTC 模块是以一个 32768Hz 慢时钟为基础的计数器，MCU 在 SFR 中可设置外部慢时钟或内部慢时钟。RTC 可在 MCU 休眠时独立工作，可设置 RTC 中断以唤醒 MCU。

设置 RTC_TMR 进行 RTC 定时，定时溢出标志位 RTC_OV(一般为一个 32KHz 的高电平)，MCU 端进行同步并产生中断。

11.5. SFR 描述

11.5.1 RTC 低字节寄存器 RTCL

Bit	7	6	5	4	3	2	1	0
Name	RTCL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

地址: 0xbe

Bit	Name	Function
[7:0]	RTCL[7:0]	16 位 RTC 的低字节值。

11.5.2 RTC 高字节寄存器 RTCH

Bit	7	6	5	4	3	2	1	0
Name	RTCH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

地址: 0xbf

Bit	Name	Function
[7:0]	RTCH[7:0]	16 位 RTC 的高字节值。

12. 定时器

功能转移调整: YS67FXXXX(X)功能转移前为 P0.2---PPGA1 P0.3---PPGA2 P0.4---PPGA3 功能转移后为 P3.0---PPGB1 P3.1---PPGB2 P3.2---PPGB3
PWM 复制口选择: P0.5---PPGAC1 P0.6---PPGAC2 P0.7---PPGAC3

YS67FXXXX(X)有 4 个 16 位的定时/计数器, 它们分别是 Timer1、Timer2、Timer3 和 Timer4, 每个定时/计数器都是由两个 8bit 的特殊功能寄存器构成。时钟源可以选择快时钟或慢时钟, 只有 Timer1 带有预分频器, Timer2、Timer3 和 Timer4 只能选择固定的分频比。根据设置的不同, 它们可工作在 16bit 模式、8bit 模式、PPG 模式以及 Capture 模式。

12.1. 定时器 1

Timer1 由高 8 位 TH1 和低 8 位 TL1 组成, 可通过字节传送指令为它们分别设置初值, 以便它们可以定时为不同的时间并获得所需的计数值, 它可以工作于 16 bit 模式和 8 bit 自动重载模式。Timer1 分为 8 位自动重载功能和 16 位计数功能, 重载值为高位的寄存器。16bit 的模式下有一对预装载寄存器进行重载。

12.1.1 16 bit 自动重载模式

在此方式下, timer1 是按 16 位加 1 计数器工作的, 该计数器是由高 8 位 TH 和低 8 位 TL 组成, 同时拥有一对预装载寄存器 (不可见, 图中虚线框)。在 timer1 启动工作前, CPU 先要为其装入方式控制字, 以设定其工作方式和时钟来源, 在事件计数方式下, 时钟来源是 P0.7。然后再为其预装定时器/计数器初值, 并通过指令启动其工作。16 位计数器按加 1 计数器计数, 当计数从 0xFFFF 溢出到 0x0000 时自动向 CPU 发溢出中断请求, 并再次按照预装载寄存器值计数。发生重载主要有两种方式: TR1 第一次使能立即装载、计数器发生溢出重载。

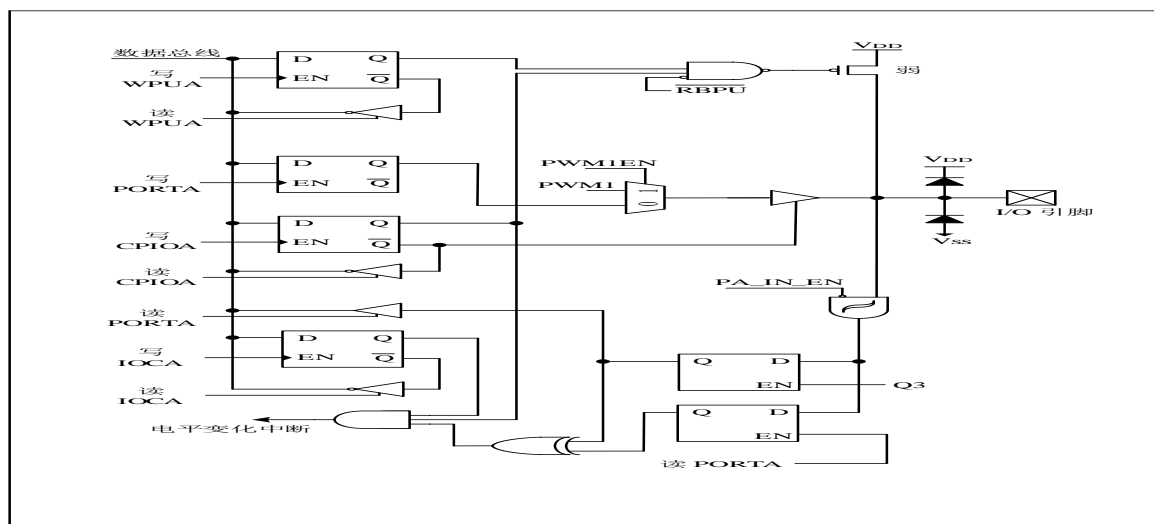


图 13-1 16b 功能框图

12.1.2 8 bit 自动重载模式

在此方式下，timer1 被拆成一个 8 位重载寄存器 TH 和一个 8 位计数器 TL，CPU 对它们初始化时必须送相同的定时时间常数初值/计数器初值。当定时器/计数器启动后，TL 按 8 位加 1 计数器计数，每当它计满回零时，一方面向 CPU 发出溢出中断请求，另一方面从 TH 中重载获得时间常数初值并启动计数。显然，定时器/计数器在此方式下工作时 TL 回零能自动重载 TH 中的初值，但计数器长度仅有 8 位，最大计数值只有 256。在此模式下，如果 Timer 在溢出产生中断的同时，CPU 写 Timer 的值，则以 CPU 写入的值为最高优先级。

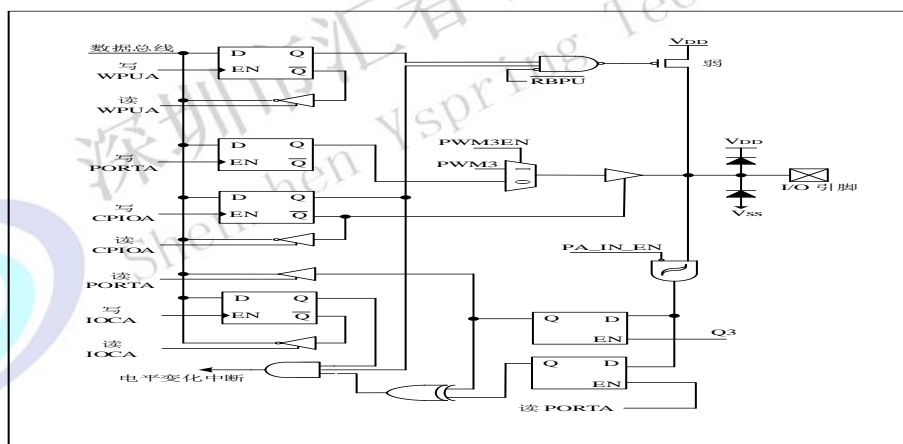


图 13-2 8b 功能框图

12.1.3 外部计数模式

Timer1 外部计数模式时系统时钟 TS 对外部引脚（P0.7）上的电平进行检测，当检测到下降沿时递增 1。

Timer1 的使能信号除 TR1 外，还可以选择根据外部 pin（P0.4）的输入来控制其使能，Timer1 根据寄存器设置可以选择外部计数模式或定时器模式，所有模式都通过控制定时器计数的 en 端来实现。

12.1.4 SFR 描述

12.1.4.1 Timer 时钟控制寄存器 CKCON: 0x8e

Bit	7	6	5	4	3	2	1	0
Name	T4MH	T4ML	T3MH	T3ML	T2MH	T2ML	SCA[1:0]	
Type	W/R	W/R	W/R	W/R	W/R	W/R	R/W	
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7]	T4MH	Timer4 高字节时钟选择 选择提供给 Timer4 的计数时钟。具体来说，是在 8 位分离计数模式下，选择高 8 位计数器的计数时钟。 0: Timer4 高字节使用 TMR4CN 寄存器中的 T4XCLK 定义的时钟来计数； 1: Timer4 高字节使用系统时钟来计数。						
[6]	T4ML	Timer4 低字节时钟选择 选择提供给 Timer4 的计数时钟。具体来说，是在 8 位分离计数模式下，选择低 8 位计数器的计数时钟。 0: Timer4 低字节使用 TMR4CN 寄存器中的 T4XCLK 定义的时钟来计数； 1: Timer4 低字节使用系统时钟来计数。						
[5]	T3MH	Timer3 高字节时钟选择 选择提供给 Timer3 的计数时钟。具体来说，是在 8 位分离计数模式下，选择高 8 位计数器的计数时钟。 0: Timer3 高字节使用 TMR3CN 寄存器中的 T3XCLK 定义的时钟来计数； 1: Timer3 高字节使用系统时钟来计数。						
[4]	T3ML	Timer3 低字节时钟选择 选择提供给 Timer3 的计数时钟。具体来说，是在 8 位分离计数模式下，选择低 8 位计数器的计数时钟。 0: Timer3 低字节使用 TMR3CN 寄存器中的 T3XCLK 定义的时钟来计数； 1: Timer3 低字节使用系统时钟来计数。						
[3]	T2MH	Timer2 高字节时钟选择 选择提供给 Timer2 的计数时钟。具体来说，是在 8 位分离计数模式下，选择高 8 位计数器的计数时钟。 0: Timer2 高字节使用 TMR2CN 寄存器中的 T2XCLK 定义的时钟来计数； 1: Timer2 高字节使用系统时钟来计数。						
[2]	T2ML	Timer2 低字节时钟选择 选择提供给 Timer2 的计数时钟。具体来说，是在 8 位分离计数模式下，选择低 8 位计数器的计数时钟。 0: Timer2 低字节使用 TMR2CN 寄存器中的 T2XCLK 定义的时钟来计数； 1: Timer2 低字节使用系统时钟来计数。						
[1:0]	SCA[1:0]	Timer1 分频比选择。 00: 系统时钟 4 分频 01: 系统时钟 12 分频 10: 系统时钟 48 分频 11: 系统慢时钟 8 分频						

12.1.4.2 Timer1 模式寄存器 TMOD: 0x89

Bit	7	6	5	4	3	2	1	0
Name	GATE1	IN1PL	C/T1	T1M		T1CM	TF1	TR1
Type	R/W	R/W	R/W	R/W	(Reserved)	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7]	GATE1	timer1 Gate Control。 0: timer1 使能只受控于 TR1 1: timer1 使能不止受控于 TR1, 同时受控于 IN1PL 位						
[6]	IN1PL	用于决定外部引脚 (P0.4) 上的电平控制 timer1 使能。 0: 外部引脚低电平使能 1: 外部引脚高电平使能						
[5]	C/T1	counter1/timer1 选择。 0: timer 计数模式: 按照 SCA[1:0]选择的时钟递增 1。 1: 外部计数模式: 系统时钟 TS 对外部引脚 (P0.7) 上的电平进行检测, 当检测到下降沿时递增 1。 注: SCA[1:0]在寄存器 CKCON[1:0]里面。						
[4]	T1M	timer1/counter1 模式选择。 0: 模式 1, 即 16 位自动重载 counter/timer 1: 模式 2, 即 8 位自动重载 counter/timer 注: 在 8 位自动重载模式下, 第一个计数周期并不是从重载值开始计数, 而是从初始值开始计数。 初始值可以用软件直接设置。						
[3]	Reserved	保留位						
[2]	T1CM	Timer1 时钟选择 选择 Timer1 的计数时钟源。当 C/T1 为 1 时, 则忽略此设置。 0: 定时器 1 使用分频配置位 SCA[1:0]选择的时钟计数; 1: 定时器 1 使用系统时钟计数。						
[1]	TF1	timer1/counter1 溢出标志。 当 timer1/counter1 上溢 (由计数最大值变为 0) 时, 硬件自动把这一位置 1。此标志位可用软件来清除, 但是当 CPU 进入相应的中断向量后, 会自动清除此标志, 所以在中断处理程序中去读取这一位时, 始终返回 0。						
[0]	TR1	timer1/counter1 运行控制。 设置为 1 时, timer1/counter1 开始运行; 设置为 0 时暂停运行。						

12.1.4.3 Timer1 低字节寄存器 TL1: 0x8b

Bit	7	6	5	4	3	2	1	0
Name	TL1[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						

[7:0]	TL1	16 位 Timer1 的低字节值。 注意：写此寄存器为写预装载寄存器，读为当前计数值
-------	-----	--

12.1.4.4 Timer1 高字节寄存器 TH1: 0x8d

Bit	7	6	5	4	3	2	1	0
Name	T1H[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7:0]	TH1	16 位 Timer1 的高字节值。 注意：写此寄存器为写预装载寄存器，读为当前计数值 8 位模式时用于保存 TIMER1 的自动重载值。 注：配置 8 位重载值需要先配置 8 位模式						

12.2. 定时器 2

Timer2 是一个 16 bit 定时器，由两个 8 bit 的特殊功能寄存器组成，它们分别是 TMR2L 和 TMR2H。Timer2 可以工作在三种模式：16 bit 自动重载模式、8 bit 自动重载模式、以及 PPG 模式，另外它还可以用于 I2C 超时监控功能。

12.2.1 16 bit 自动重载模式

在此模式下，Timer2 是按 16 位加 1 计数器工作的，当计数发生溢出时（从 0xffff 到 0x0000），定时器溢出中断标志 TF2H 被置位，重载寄存器 TMR2RLL 和 TMR2RLH 的值被重新载入到 Timer2 的寄存器中。在中断使能的情况下，CPU 会进入中断向量。如果 TF2LEN 被置一，则使能 timer2 的低字节中断，当 timer2 的低字节溢出时，也会产生 CPU 中断。

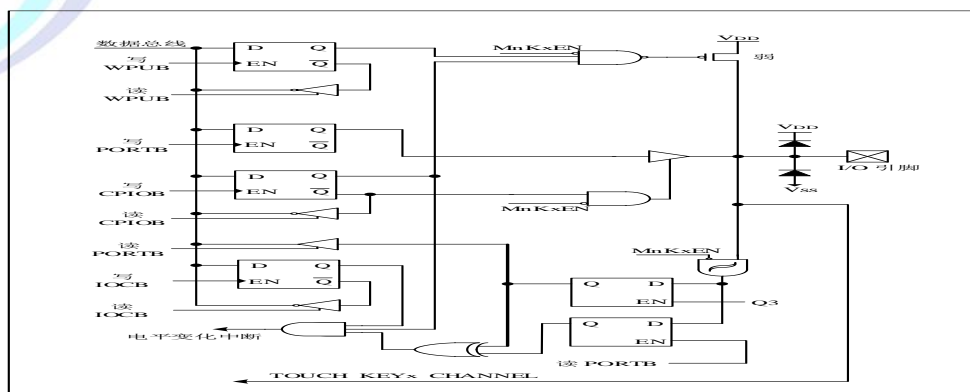


图 13-3 16b 功能框图

12.2.2 8 bit 自动重载模式

将寄存器 TMR2CN[1:0] 设置成[00]，则 Timer2 工作在 8 bit 自动重载模式。此时 Timer2 被分成两个独立的 8 bit 的定时/计数器，计数值分别在 TMR2L 与 TMR2H 之中。

当 timer2 的高字节从 0xFF 溢出到 0x00 时，重载寄存器 TMR2RLH 的值被重新载入到 TMR2H 之中，同时 TF2H 由硬件自动置 1。如果使能了 timer2 的中断，CPU 会进入中断向量。

当 timer2 的低字节从 0xFF 溢出到 0x00 时，重载寄存器 TMR2RLL 的值自动载入到 TMR2L 之中，由硬件自动置一 TF2L 位。中断允许，则 CPU 会进入中断向量。注意，在此模式下，只有高字节需要使能信号才开始计数，低字节不需要使能信号就开始计数。

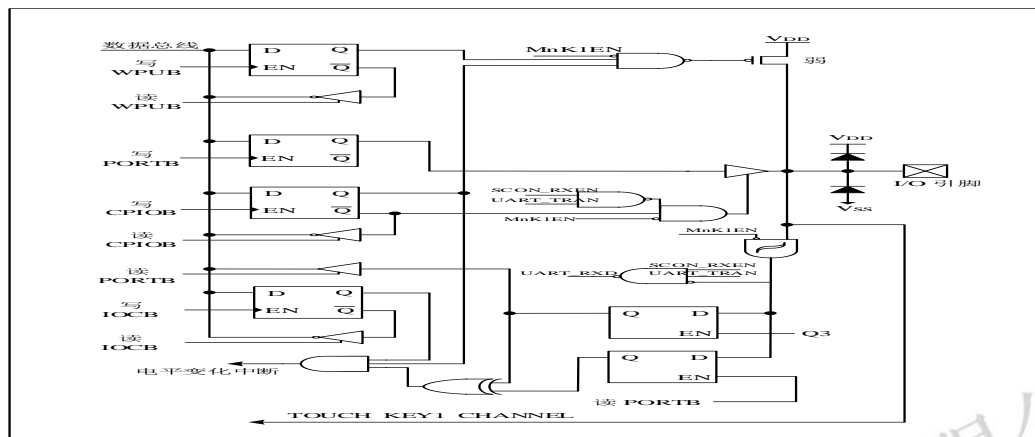


图 13-4 8b 功能框图

12.2.3 PPG 模式

Timer2 在 PPG 模式下，有三组 16 bit 寄存器，分别是 PWM 输出周期寄存器 TMR2RLL 和 TMR2RLH（A 寄存器）、占空比设置寄存器 TMR2BLL 和 TMR2BLH（B 寄存器）、计数寄存器 TMR2L 和 TMR2H。功能转移之前 PWM3（PPGA3）的输出是 P0.4，功能转移之后 PWM3（PPGB3）的输出是 P3.2。

当往 TMR2_EN 位写 1 时，就开始了 PPG 的一个周期，这个动作会清零计数寄存器 TMR2L 和 TMR2H，并设置 PWM 输出为高。当计数值与 A 寄存器发生匹配时，将清零计数器的值，并置 PWM 的输出为高；计数值与 B 寄存器发生匹配时，PWM 输出发生翻转。如果 B 寄存器的值大于 A 寄存器，则 PWM 输出为高，B 寄存器的值为零时，PWM 输出为低。PWM3CO（PPGAC3）为 PWM3 死区复制口，详细内容在死区有相关介绍。

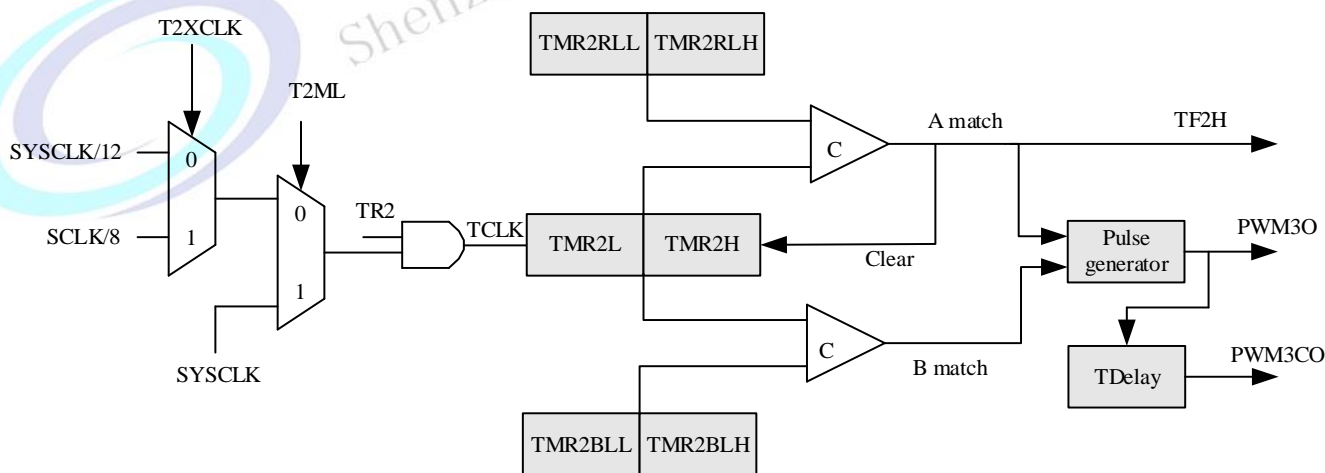


图 13-5 PPG 功能框图

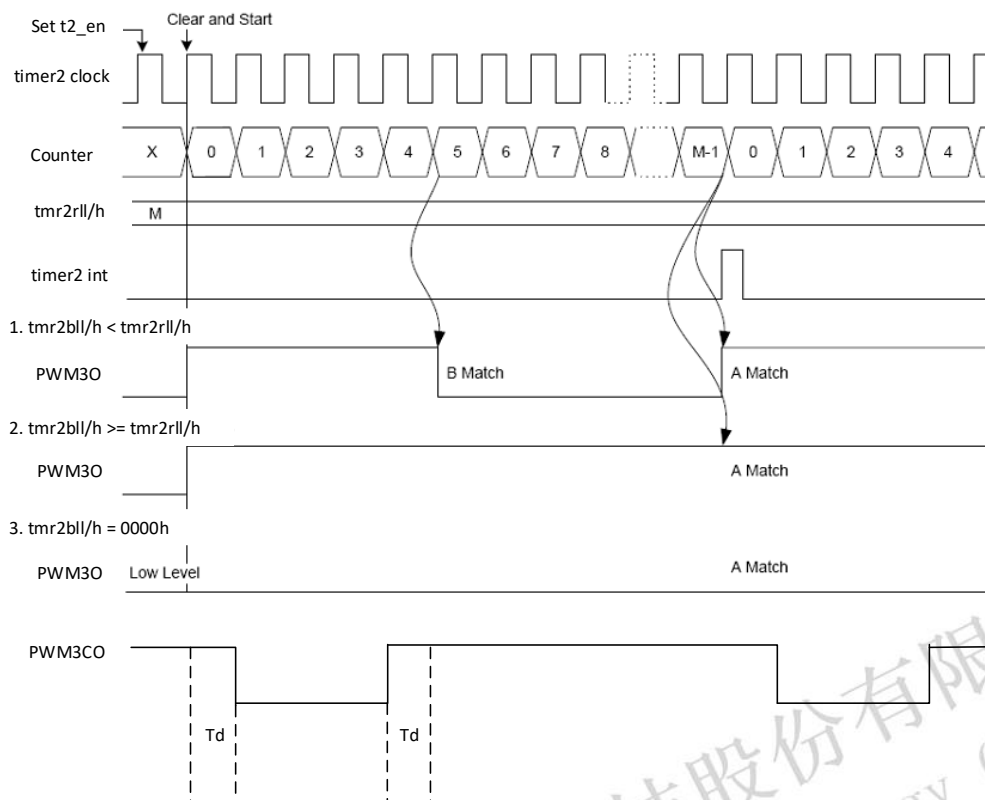


图 13-6 ppg 输出波形

12.2.4 死区延时

在 PPG 模式下可以设置 PWM 的死区延时，死区延时由一组专用的寄存器（TMR2DZ）来进行配置，注意死区时间设置必须小于二分之一的占空比设置，占空比设置过低（小于 5）或过高（大于 FFFA）都会导致死区功能异常。同时打开 PPG 模式使能和死区使能，PWM 复制口会直接输出波形。

死区使能前需要提前配置好 PWM 周期和占空比以及死区时间，死区使能后 PWM 复制口正常工作。调整 PWM 口极性不会导致 PWM 复制口极性发生改变，可根据需要单独对 PWM 口和 PWM 复制口进行极性选择。当死区时间按照系统时钟 12M 时，可以做到约 5us 左右。死区时间 $T_d = (DZTM + 1) * \text{Timer clock}$ 。

12.2.5 I2C 超时检测功能描述

timer2 的超时监控功能必须用于 timer2 工作于 16bit 模式，并且打开 timer2。在 I2C 进行工作的时候打开边沿监测，检测到 I2C 的 start 信号以后，会在时钟 SCL 电平发生跳变的时候，对 TIMER2 的计数值进行自动重载；如果 SCL 电平不变化，则 Timer2 一直递增计数，最终产生溢出中断。当 I2C 通信结束（收到 End 信号）以后，timer2 停止监测，不再有重载操作，并一直递增计数，直到溢出后再发生重载操作。

注：用于 I2C 超时监控的 timer2 计数时钟只能选择系统时钟，即 CKCON 寄存器的 bit2 必须为 1（T2ML=1）。

12.2.6 SFR 描述

12.2.6.1 Timer2 控制寄存器 TMR2CN : 0xc8

Bit	7	6	5	4	3	2	1	0
Name	TF2H	TF2L	TF2LEN		T2XCLK	T2POL	T2MOD[1:0]	
Type	R/W	R/W	R/W	R	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	1
Bit	Name	Function						
[7]	TF2H	<p>timer2 高字节溢出标志。</p> <p>当 timer2 的高字节从 0xFF 溢出到 0x00 时，硬件自动置 1。在 16 位模式下，当 timer2 从 0xFFFF 溢出到 0x0000 时，硬件自动置 1。如果使能了 timer2 的中断，则在此位置 1 时，CPU 会进入中断向量。硬件不会自动清除此位，此标志位只能由硬件产生，不能由软件写入。</p> <p>对于 PPG 模式：</p> <p>在调试模式下（单步时或 CPU 暂停时），PWM 照常输出，但不会产生中断标志位。PWM 的占空比为进入调试模式之前的占空比，如果在单步时修改了占空比寄存器，则也会实时反应到当前输出的 PWM 波形上。</p> <p>在全速运行时，会在每个 PWM 输出周期产生一次中断标志。</p>						
[6]	TF2L	<p>timer2 低字节溢出标志。</p> <p>在 8 位计数模式或 16 位计数模式下，当 timer2 的低字节从 0xFF 溢出到 0x00 时，硬件自动置 1。硬件不会自动清除此位，此标志位只能由硬件产生，不能由软件写入。</p> <p>在 PPG 下不会产生溢出标志，也不会产生相应的中断。</p>						
[5]	TF2LEN	<p>timer2 低字节中断使能。</p> <p>如果设置为 1，则使能 timer2 的低字节中断。如果同时使能了 timer2 的中断，则当 timer2 的低字节溢出时，就会产生 CPU 中断。</p>						
[4]	Reserved	保留位						
[3]	T2XCLK	<p>timer2 外部时钟选择。</p> <p>如果 timer2 为 8 位模式，则同时为高字节和低字节计数器选择外部时钟。timer2 时钟选择位（CKCON 中的 T2MH 和 T2ML）可以进一步为各字节选择外部时钟或系统时钟。注：外部时钟源均同步到系统时钟源。</p> <p>0：计数时钟为系统时钟 12 分频</p> <p>1：计数时钟为慢时钟 8 分频（慢时钟由 CLKCF [4] 决定）</p> <p>注：I2C 超时监测功能使能时，必须选择系统时钟为计数时钟，即 CKCON 的 T2ML=1。</p> <p>当系统时钟为慢时钟时，计数时钟禁止选慢时钟 8 分频，即 T2XCLK=1。</p>						
[2]	T2POL	<p>timer2 PPG PWNOUT 极性选择。</p> <p>0: start high。</p> <p>1: start low。</p>						
[1:0]	T2MOD[1:0]	<p>timer2 模式选择。</p> <p>00: timer2 工作在两个单独的 8 位计数器模式。</p> <p>01: timer2 工作在一个 16 位计数器模式。</p> <p>1x: timer2 工作与 PPG 模式(16bit)</p>						

12.2.6.2 Timer2 重载寄存器低字节 TMR2RLL: 0x91

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7:0]	TMR2RLL	Timer2 重载寄存器低字节。 TMR2RLL 保存 timer2 的重载值的低字节。 在 PPG 模式下, 用于设置 PWM 输出的周期, 即 A 寄存器						

12.2.6.3 Timer2 重载寄存器高字节 TMR2RLH: 0x92

Bit	7	6	5	4	3	2	1	0
Name	TMR2RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7:0]	TMR2RLH	Timer2 重载寄存器高字节。 TMR2RLH 保存 timer2 的重载值的高字节。 在 PPG 模式下, 用于设置 PWM 输出的周期, 即 A 寄存器						

12.2.6.4 Timer2 PPG 模式占空比设置值低字节 TMR2BLL: 0x93

Bit	7	6	5	4	3	2	1	0
Name	TMR2BLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
[7:0]	TMR2BLL	Timer2 PPG 模式下设置占空比的低字节。 在 8B,16B 模式下, 读此位将读出 Timer2 的值						

12.2.6.5 Timer2 PPG 模式占空比设置值高字节 TMR2BLH: 0x94

Bit	7	6	5	4	3	2	1	0
Name	TMR2BLH[7:0]							

Type	R/W						
Reset	0	0	0	0	0	0	0
Bit	Name	Function					
[7:0]	TMR2BLH	Timer2 PPG 模式下设置占空比的高字节。 在 8B,16B 模式下，读此位将读出 Timer2 的值					

12.2.6.6 Timer2 PPG 模式死区延时控制寄存器 TMR2DZ: 0xE5

Bit	7	6	5	4	3	2	1	0
Name	TMR2DZ[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
7	DZEN	死区延时使能 (PWM 复制口使能)						
6	DZPOL	PWM 复制口极性选择 0: start low 1: start high						
5:0	DZTM	死区时间值						

12.3. 定时器 3

Timer3 是一个 16 bit 定时器，由两个 8 bit 的特殊功能寄存器组成，它们分别是 TMR3L 和 TMR3H。Timer3 可以工作在四种模式：16 bit 自动重载模式、8 bit 自动重载模式、PPG 模式、以及外部事件捕捉模式。Timer3 与 Timer2 的不同之处在于 Timer3 具有 Capture 模式，而 Timer2 没有，但 Timer2 具有 i2c 时钟超时检测而 Timer3 没有。

12.3.1 16 bit 自动重载模式

请参考 timer2 16 bit 自动重载模式的章节。

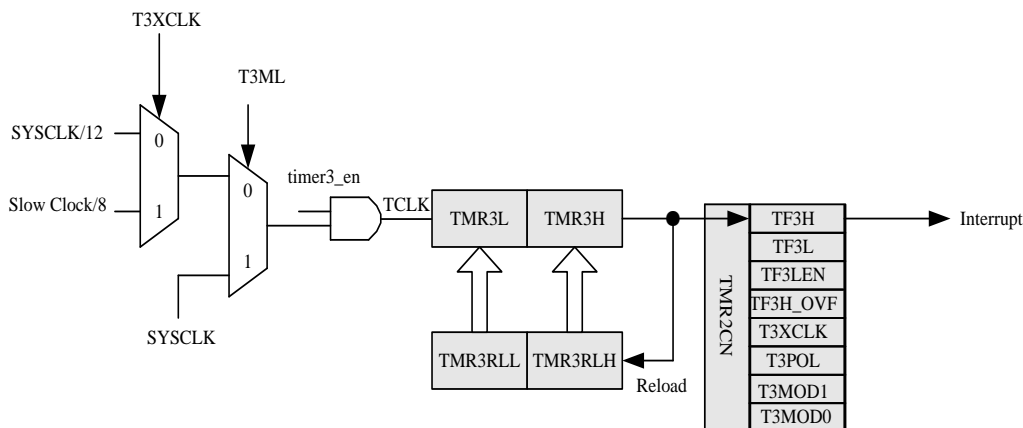


图 13-7 16b 功能框图

12.3.2 8 bit 自动重载模式

请参考 timer2 8 bit 自动重载模式的章节。

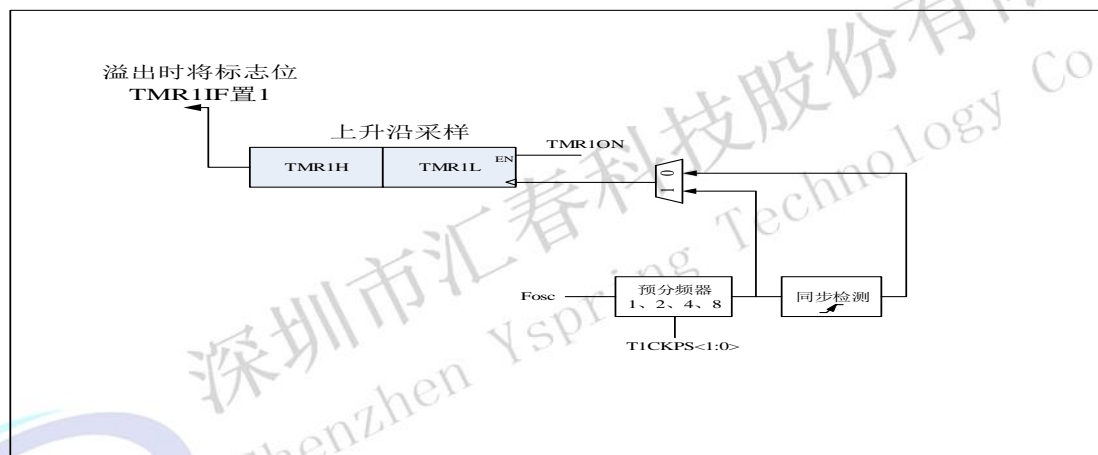


图 13-8 8b 功能框图

12.3.3 PPG 模式

Timer3 在此模式下，功能转移之前 PWM1（PPGA1）的输出是 P0.2，功能转移之后 PWM1（PPGB1）的输出是 P3.0。其他功能请参考 timer2 PPG 模式的章节。PWM1CO（PPGAC1）为 PWM1 死区复制口。

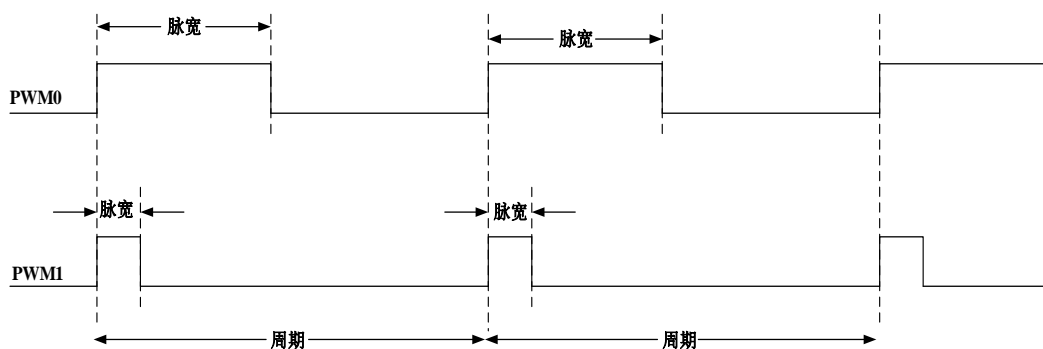


图 13-9 PPG 功能框图

12.3.4 死区延时

请参考 Timer2 模块。

12.3.5 捕捉模式

Timer3 工作在此模式下，可以对外部事件进行计数，功能转移之前外部事件引脚是 P0.2，功能转移后外部事件引脚为 P3.0。根据软件配置的不同，还可以对慢时钟的频率进行测量。当外部事件引脚的电平发生从 0 到 1 的变化，或者处于慢时钟的上升沿时，捕捉事件就发生。此时硬件会把计数器 TMR3L 和 TMR3H 的值捕捉到 TMR3RLL 和 TMR3RLH 寄存器，与此同时会对 TMR3L 和 TMR3H 清零，并置 TF3H 标志位，如果中断允许，则 CPU 进入中断向量。

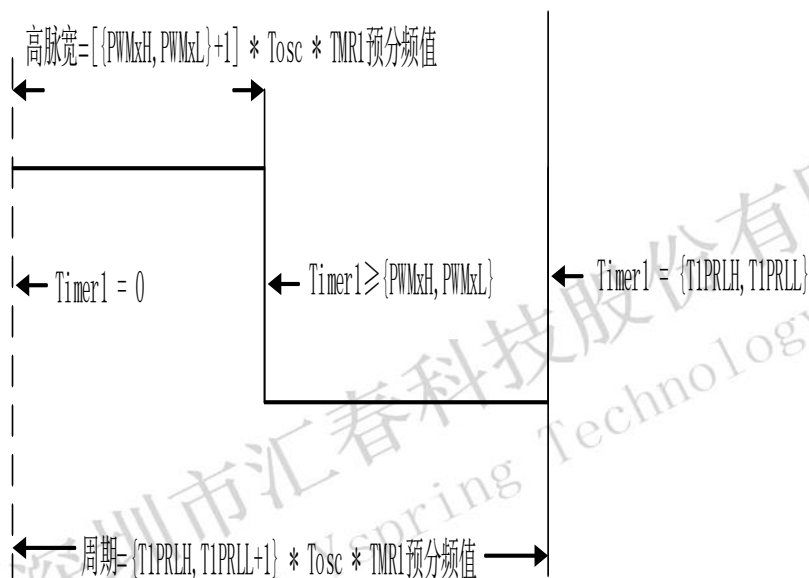


图 13-10 capture 功能框图

12.3.6 SFR 描述

13.3.6.1 Timer3 控制寄存器 TMR3CN: 0xb1

Bit	7	6	5	4	3	2	1	0
Name	TF3H	TF3L	TF3LEN	TF3H_OVF	T3XCLK	T3POL	T3MOD[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	1

Bit	Name	Function
[7]	TF3H	<p>Timer3 高字节溢出标志。</p> <p>当 timer3 的高字节从 0xFF 溢出到 0x00 时，硬件自动置 1。在 16 位模式下，当 timer3 从 0xFFFF 溢出到 0x0000 时，硬件自动置 1。如果使能了 timer3 的中断，则在此位置 1 时，CPU 会进入中断向量。硬件不会自动清除此位。</p> <p>注：抓捕模式同时也是 16 位模式，但仅在抓捕时产生中断，溢出不产生中断。</p> <p>此标志位只能由硬件产生，不能由软件写入。</p> <p>在抓捕模式下，当抓捕动作发生时，会产生此标志位。</p> <p>对于 PPG 模式：</p> <p>在调试模式下（单步时或 CPU 暂停时），PWM 照常输出，但不会产生中断标志位。PWM 的占空比为进入调试模式之前的占空比，如果在单步时修改了占空比寄存器，则也会实时反应到当前输出的 PWM 波形上。</p> <p>在全速运行时，会在每个 PWM 输出周期产生一次中断标志。</p>
[6]	TF3L	<p>Timer3 低字节溢出标志。</p> <p>在 8 位或 16 位模式下，当 timer3 的低字节从 0xFF 溢出到 0x00 时，硬件自动置 1。硬件不会自动清除此位。</p> <p>此标志位只能由硬件产生，不能由软件写入。</p> <p>在 PPG 或 Capture mode 下不会产生溢出标志，也不会产生相应的中断。</p>
[5]	TF3LEN	<p>Timer3 低字节中断使能。</p> <p>如果设置为 1，则使能 timer3 的低字节中断。如果同时使能了 timer3 的中断，则当 timer3 的低字节溢出时，就会产生 CPU 中断。</p>
[4]	TF3H_OVF	<p>Timer3 高字节溢出标志位。</p> <p>当 timer3 的高字节从 0xFF 溢出到 0x00 时，硬件自动置 1。软件清零。</p> <p>只在 capture mode 时产生此标志，但此标志不产生中断。</p>
[3]	T3XCLK	<p>Timer3 外部时钟选择。</p> <p>选择“外部”和“抓捕”时钟信号。如果 timer3 为 8 位模式，则同时为高字节和低字节计数器选择“外部”时钟。Timer3 时钟选择位（CKCON 中的 T3MH 和 T3ML）可以进一步为各字节选择“外部”时钟或系统时钟。</p> <p>注：外部时钟源均同步到系统时钟源。</p> <p>0：计数时钟为系统时钟 12 分频，抓捕信号为慢时钟 8 分频；</p> <p>1：计数时钟为慢时钟 8 分频，抓捕信号为外部 pin 输入信号</p> <p>注：当系统时钟为慢时钟时，计数时钟禁止选慢时钟 8 分频，即 T3XCLK=1。</p>
[2]	T3POL	<p>Timer3 PPG PWNOUT 极性选择。</p> <p>0：start high(T20/PWMO is low level at disable)。</p> <p>1：start low(T20/PWMO is high level at disable)。</p>

[1:0]	T3MOD[1:0]	Timer3 模式选择。 00: timer3 工作在两个单独的 8 位计数器模式。 01: timer3 工作在一个 16 位计数器模式。 10: timer3 工作于抓捕模式(16bit) 11: timer3 工作与 PPG 模式(16bit)
-------	------------	---

13.3.6.2 Timer3 重载寄存器低字节 TMR3RLL: 0xe1

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR3RLL[7:0]	Timer3 重载寄存器低字节。Cap 模式下，将 T3 的值存入此寄存器。 TMR3RLL 保存 timer3 的重载值的低字节。 在 PPG 模式下，用于设置 PWM 输出的周期，即 A 寄存器

13.3.6.3 Timer3 重载寄存器高字节 TMR3RLH: 0xe2

Bit	7	6	5	4	3	2	1	0
Name	TMR3RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR3RLH[7:0]	Timer3 重载寄存器高字节。Cap 模式下，将 T3 的值存入此寄存器 TMR3RLH 保存 timer3 的重载值的高字节。 在 PPG 模式下，用于设置 PWM 输出的周期，即 A 寄存器

13.3.6.4 Timer3 PPG 模式占空比设置值低字节 TMR3BLL: 0xe3

Bit	7	6	5	4	3	2	1	0
Name	TMR3BLL[7:0]							
Type	R/W							

Reset	0	0	0	0	0	0	0	0
-------	---	---	---	---	---	---	---	---

Bit	Name	Function
[7:0]	TMR3BLL[7:0]	Timer3 PPG 模式下设置占空比的低字节。 在 8B,16B,CAP 模式下，读此位将读出 Timer3 的值

13.3.6.5 Timer3 PPG 模式占空比设置值高字节 TMR3BLH: 0xe4

Bit	7	6	5	4	3	2	1	0
Name	TMR3BLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR3BLH[7:0]	Timer3 PPG 模式下设置占空比的高字节。 在 8B,16B,CAP 模式下，读此位将读出 Timer3 的值

13.3.6.6 Timer3 PPG 模式死区延时控制寄存器 TMR3DZ: 0xE6

Bit	7	6	5	4	3	2	1	0
Name	TMR3DZ[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
7	DZEN	死区延时使能（PWM 复制口使能）						
6	DZPOL	PWM 复制口极性选择 0: 反向 1: 正向						
5:0	DZTM	死区时间值						

12.4. 定时器 4

Timer4 是一个 16 bit 定时器，由两个 8 bit 的特殊功能寄存器组成，它们分别是 TMR4L 和 TMR4H。Timer4 可以工作在四种模式：16 bit 自动重载模式、8 bit 自动重载模式、PPG 模式、以及外部事件捕捉模式。

12.4.1 16 bit 自动重载模式

请参考 timer2 16 bit 自动重载模式的章节。

$$\text{PWM分辨率} = \frac{\text{Log}[(\text{T1PRLH}, \text{T1PRLH}) + 1]}{\text{Log}[2]} \text{ 位}$$

图 13-11 16b 功能框图

12.4.2 8 bit 自动重载模式

请参考 timer2 8 bit 自动重载模式的章节。

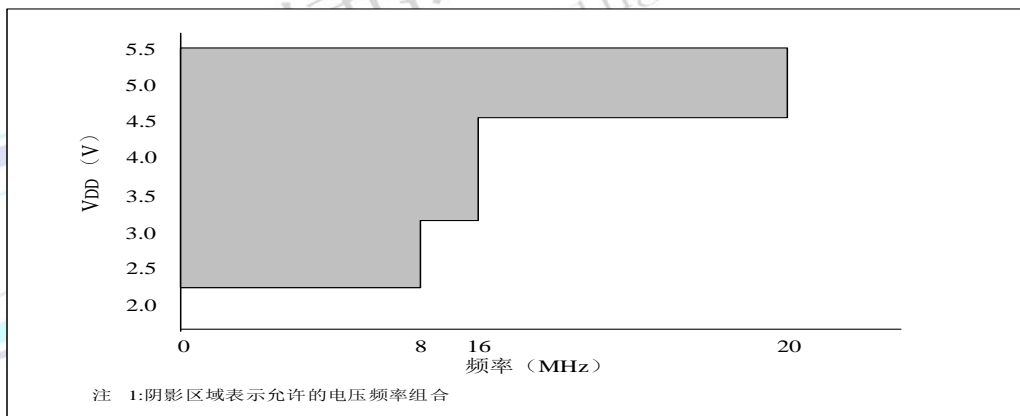


图 13-12 8b 功能框图

12.4.3 PPG 模式

Timer4 在此模式下，功能转移之前 PWM2（PPGA2）的输出是 P0.3，功能转移之后 PWM2（PPGB2）的输出是 P3.1。其他功能请参考 timer2 PPG 模式的章节。PWM1CO（PPGAC2）为 PWM2 死区复制口。

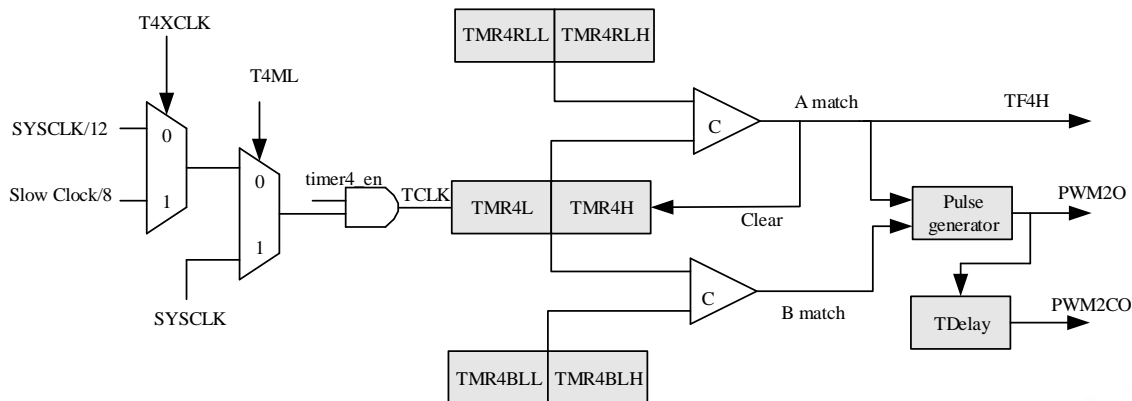


图 13-13 PPG 功能框图

12.4.4 死区延时

请参考 Timer2 模块。

12.4.5 捕捉模式

Timer4 工作在此模式下，可以对外部事件进行计数，功能转移之前外部事件引脚是 P0.3，功能转移后外部事件引脚为 P3.1。其他功能请参考 Timer3 捕捉模式的章节。

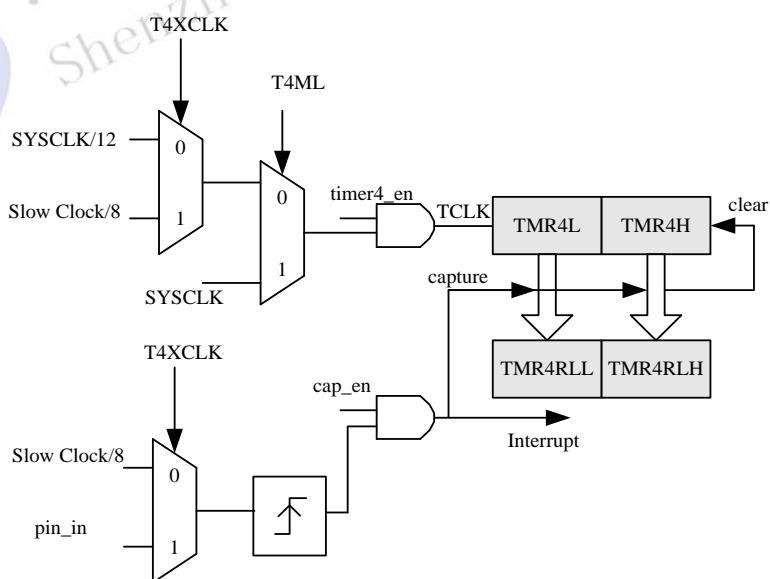


图 13-14 capture 功能框图

12.4.6 SFR 描述

13.4.6.1 Timer4 控制寄存器 TMR4CN: 0xb2

Bit	7	6	5	4	4	2	1	0
Name	TF4H	TF4L	TF4LEN	TF4H_OVF	T4XCLK	T4POL	T4MOD[1:0]	
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Reset	0	0	0	0	0	0	0	1

Bit	Name	Function
[7]	TF4H	<p>Timer4 高字节溢出标志。</p> <p>当 timer4 的高字节从 0xFF 溢出到 0x00 时，硬件自动置 1。在 16 位模式下，当 timer4 从 0xFFFF 溢出到 0x0000 时，硬件自动置 1。如果使能了 timer4 的中断，则在此位置 1 时，CPU 会进入中断向量。硬件不会自动清除此位。</p> <p>注：抓捕模式同时也是 16 位模式，但仅在抓捕时产生中断，溢出不产生中断。</p> <p>此标志位只能由硬件产生，不能由软件写入。</p> <p>在抓捕模式下，当抓捕动作发生时，会产生此标志位。</p> <p>对于 PPG 模式：</p> <p>在调试模式下（单步时或 CPU 暂停时），PWM 照常输出，但不会产生中断标志位。PWM 的占空比为进入调试模式之前的占空比，如果在单步时修改了占空比寄存器，则也会实时反应到当前输出的 PWM 波形上。</p> <p>在全速运行时，会在每个 PWM 输出周期产生一次中断标志。</p>
[6]	TF4L	<p>Timer4 低字节溢出标志。</p> <p>在 8 位或 16 位模式下，当 timer4 的低字节从 0xFF 溢出到 0x00 时，硬件自动置 1。硬件不会自动清除此位。</p> <p>此标志位只能由硬件产生，不能由软件写入。</p> <p>在 PPG 或 Capture mode 下不会产生溢出标志，也不会产生相应的中断。</p>
[5]	TF4LEN	<p>Timer4 低字节中断使能。</p> <p>如果设置为 1，则使能 timer4 的低字节中断。如果同时使能了 timer4 的中断，则当 timer4 的低字节溢出时，就会产生 CPU 中断。</p>
[4]	TF4H_OVF	<p>Timer4 高字节溢出标志位。</p> <p>当 timer4 的高字节从 0xFF 溢出到 0x00 时，硬件自动置 1。软件清零。</p> <p>只在 capture mode 时产生此标志，但此标志不产生中断。</p>
[3]	T4XCLK	<p>Timer4 外部时钟选择。</p> <p>选择“外部”和“抓捕”时钟信号。如果 timer4 为 8 位模式，则同时为高字节和低字节计数器选择“外部”时钟。Timer4 时钟选择位（CKCON 中的 T4MH 和 T4ML）可以进一步为各字节选择“外部”时钟或系统时钟。</p> <p>注：外部时钟源均同步到系统时钟源。</p> <p>0：计数时钟为系统时钟 12 分频，抓捕信号为慢时钟 8 分频；</p> <p>1：计数时钟为慢时钟 8 分频，抓捕信号为外部 pin 输入信号</p> <p>注：当系统时钟为慢时钟时，计数时钟禁止选慢时钟 8 分频，即 T4XCLK=1。</p>
[2]	T4POL	<p>Timer4 PPG PWNOUT 极性选择。</p> <p>0：start high(T20/PWMO is low level at disable)。</p> <p>1：start low(T20/PWMO is high level at disable)。</p>

[1:0]	T4MOD[1:0]	Timer4 模式选择。 00: timer4 工作在两个单独的 8 位计数器模式。 01: timer4 工作在一个 16 位计数器模式。 10: timer4 工作于抓捕模式(16bit) 11: timer4 工作与 PPG 模式(16bit)
-------	------------	---

13.4.6.2 Timer4 重载寄存器低字节 TMR4RLL: 0xf1

Bit	7	6	5	4	4	2	1	0
Name	TMR4RLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR4RLL[7:0]	Timer4 重载寄存器低字节。Cap 模式下，将 T4 的计数值低字节存入此寄存器。 TMR4RLL 保存 timer4 的重载值的低字节。 在 PPG 模式下，用于设置 PWM 输出的周期，即 A 寄存器

13.4.6.3 Timer4 重载寄存器高字节 TMR4RLH: 0xf2

Bit	7	6	5	4	4	2	1	0
Name	TMR4RLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR4RLH[7:0]	Timer4 重载寄存器高字节。Cap 模式下，将 T4 的计数值高字节存入此寄存器。 TMR4RLH 保存 timer4 的重载值的高字节。 在 PPG 模式下，用于设置 PWM 输出的周期，即 A 寄存器

13.4.6.4 Timer4 PPG 模式占空比设置值低字节 TMR4BLL: 0xf3

Bit	7	6	5	4	4	2	1	0
Name	TMR4BLL[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR4BLL[7:0]	Timer4 PPG 模式下设置占空比的低字节。 在 8B,16B,CAP 模式下, 读此寄存器将读出 Timer4 的计数值低字节。 注: 如果 timer4 一直在计数, 则读出的计数值并不能实时反映出其当前真实值。

13.4.6.5 Timer4 PPG 模式占空比设置值高字节 TMR4BLH: 0xf4

Bit	7	6	5	4	4	2	1	0
Name	TMR4BLH[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	TMR4BLH[7:0]	Timer4 PPG 模式下设置占空比的高字节。 在 8B,16B,CAP 模式下, 读此寄存器将读出 Timer4 的计数值高字节。 注: 如果 timer4 一直在计数, 则读出的计数值并不能实时反映出其当前真实值。

13.4.6.6 Timer4 PPG 模式死区延时控制寄存器 TMR4DZ: 0xE7

Bit	7	6	5	4	3	2	1	0
Name	TMR4DZ[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0
Bit	Name	Function						
7	DZEN	死区延时使能（PWM 复制口使能）						
6	DZPOL	PWM 复制口极性选择 0: 反向 1: 正向						
5:0	DZTM	死区时间值						

13. 看门狗定时器

内置一个 16 位的看门狗计数器，其中 8 位拆分为预分频计数，另外 8 位是主计数器，由行波计数器实现进一步降低功耗。当它使能时，主计数器从 0xFF 计数到 0x00 时产生溢出，并复位系统，通过寄存器 RSTEN 可配置此复位是否产生 boot。

注意：WDT 固定使用内部慢时钟。

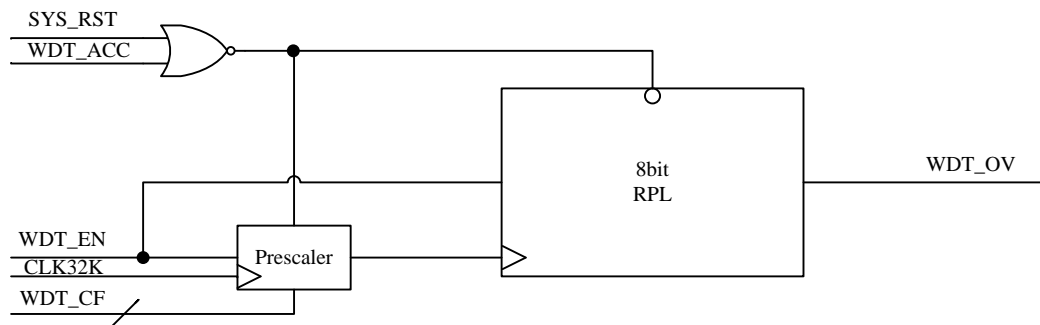


图 14-1 WDT 功能框图

13.1. WDTCF 寄存器

地址 0x86。注意，对 WDTCF 的读或写均可清除看门狗预分频器和计数器。

位	名字	复位值	功 能
7:3	NA	NA	保留位，写无效。读=0x00。
2:0	WDTCF	3'b111	分频比设置，读写 3'b000: 不分频 3'b001: 4 分频 3'b010: 8 分频 3'b011: 16 分频 3'b100: 32 分频 3'b101: 64 分频 3'b110: 128 分频 3'b111: 256 分频

注：

1. 当软件向该寄存器执行读/写操作时，就会产生一个清零信号，将看门狗复位；
2. 在 WDT 正常运行过程中，在其溢出之前，如果禁用，则内部计数器清 0，并停止计数；
3. WDT 的使能位 WDT_OPT 由寄存器 CCFG3[0]控制（读写 CCFG3 寄存器需要先解锁）。

14. UART

YS67FXXXX(X)的UART支持8位数据位和9位数据位2种工作模式。UART模块由发射子模块TX module和接收子模块RX module组成，共用一个系统时钟，但它们是两个独立的子模块，任何一个子模块均可单独工作，不影响另一个子模块。

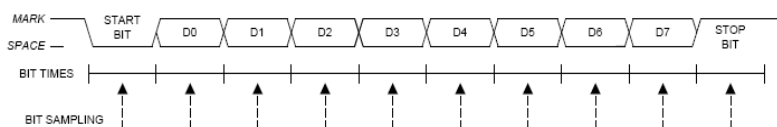
UART有两个相关的特殊功能寄存器：串行控制寄存器（SCON0）和串行数据缓冲器（SBUF0）。用同一个SBUF0地址可以访问发送寄存器和接收寄存器。写SBUF0时自动访问发送寄存器；读SBUF0时自动访问接收寄存器，不可能从发送数据寄存器中读数据。

如果UART中断被允许，则每次发送完成（SCON0中的TI0位被置‘1’）或接收到数据字节（SCON0中的RI0位被置‘1’）时将产生中断。当CPU转向中断服务程序时硬件不清除UART0中断标志。中断标志必须用软件清除，这就允许软件查询UART0中断的原因（发送完成或接收完成）。

14.1. 工作模式

14.1.1 8 位 UART

在8位UART模式下，传送每个字节数据共需要10位：1个起始位，8个数据位（先传低位）以及1个停止位。数据按照从低位到高位顺序，从TXD引脚发送出去，并从RXD引脚接收。在接收数据时，8个数据位存入SBUF0寄存器，停止位进入SCON0寄存器的第2位（RB80）。



当软件向SBUF0寄存器写入一个字节时开始数据发送。在发送结束时（停止位开始）发送中断标志TI0（SCON0.1）被置‘1’。在接收允许位REN0（SCON0.4）被置‘1’后，数据接收可以在任何时刻开始。

注：此时只是开始对接收信号进行检测，只有当接收信号电平和时序满足UART协议要求以后，才真正开始接收数据。

收到停止位后，如果满足下述条件：

- 1: RI0必须为逻辑‘0’；
- 2: 如果MCE0为逻辑‘1’，则停止位必须为‘1’。

则数据字节将被装入到接收寄存器SBUF0。

在发生接收数据溢出的情况下，先接收到的8位数据被锁存到SBUF0，而后面的溢出数据被丢弃。

如果这些条件满足，则8位数据被存入SBUF0，停止位被存入RB80，RI0标志被置位。如果这些条件不满足，则不装入SBUF0和RB80，RI0标志也不会被置‘1’。如果中断被允许，在TI0或RI0置位时将产生一个中断。

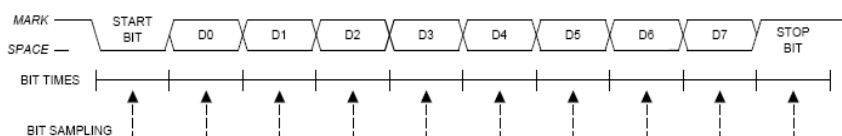


图 15-2 8 位 UART 时序图

14.1.2 9 位 UART

在9位UART方式，每个数据字节共使用11位：一个起始位、8个数据位（LSB在先）、一个可编程的第九位和一个停止位。

在发送状态下第九位数据由TB80（SCON0.3）中的值决定，由用户软件赋值。在接收状态下，第九数据位进入RB80（SCON0.2），停止位被忽略。

当执行一条向SBUF0寄存器写一个数据字节的指令时开始数据发送。在发送结束时（停止位开始）发送中断标志TI0被置‘1’。

在接收允许位REN0（SCON0.4）被置‘1’后，数据接收可以在任何时刻开始。

收到停止位后如果满足下述条件则数据字节将被装入到接收寄存器SBUF0：

1:RI0为逻辑‘0’；

2:如果MCE0为逻辑‘1’，则第九位必须为逻辑‘1’（当MCE0为逻辑‘0’时，第九位数据的状态并不重要）。如果这些条件满足，则8位数据被存入SBUF0，第九位被存入RB80，RI0标志被置位。如果这些条件不满足，则不装入SBUF0和RB80，RI0标志也不会被置‘1’。如果中断被允许，在TI0或RI0置位时将产生一个中断。

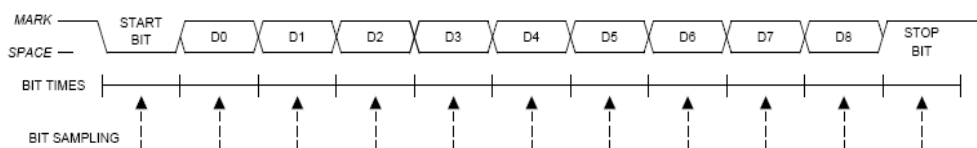


图15-3 9位UART时序图

14.1.3 SFR

15.2.3.1 UART 控制寄存器：SCON0

地址：0x98

Bit	7	6	5	4	3	2	1	0
Name	S0MODE	TX_EN	MCE0	RX_EN	TB80	RB80	TI0	RI0
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7]	S0MODE	UART 的操作模式选择。 0: 8 位 UART 模式 1: 9 位 UART 模式
[6]	TX_EN	UART 模块使能。 0: 发送功能和接收功能均禁用。 1: 发送功能启用，要启用接收功能，还必须将 SCON0[4]（RINT0）写 1。

[5]	MCE0	多处理器通讯使能。 对于 8 位 UART 模式：用于检查合法的停止位。 0：忽略停止位 1：当停止位为 1 时，RI0 有效 对于 9 位 UART 模式：多处理器通讯使能 0：忽略第九位数据。不管第九位数据是 0 还是 1，都会正常接收，并产生中断； 1：当第九位数据为 1 时，将 RI0 置 1，并产生中断。如果第九位数据是 0，则会丢弃当前已接收到的字节数据，并重新等待下一字节的开始。
[4]	RX_EN	接收使能。 0：UART0 不能接收数据。 1：UART0 可以接收数据。 注：没有寄存器对发送使能进行控制。因为发送使能要结合发送数据进行同步控制。当要发送的数据准备好以后，才给出发送使能。
[3]	TB80	第 9 个数据发送位的值。此值在 9 位 UART 模式下，自动传送到第 9 个数据位上向外发送。在 8 位 UART 模式下，这一位没有用处。
[2]	RB80	第 9 个数据接收位的值。在 9 位 UART 模式下，这一位存储第 9 个接收数据位的值。在 8 位 UART 模式下，此位置 1。
[1]	TI0	发送中断标志。 在发送完一个字节的的数据以后，此位由硬件自动置 1。在开始发送停止位时就被视为发送完毕。如果使能了串口中断，则当这一位置 1 时，会导致 CPU 进入中断向量。进入中断服务程序后，这一位必须由软件来清除。
[0]	RI0	接收中断标志。 在接收完一个字节的的数据以后，此位由硬件自动置 1。在对停止位采样时就视为接收完了一个字节的数据。如果使能了串口中断，则当这一位置 1 时，会导致 CPU 进入中断向量。进入中断服务程序后，这一位必须由软件来清除。

15.2.3.2 数据缓存寄存器 SBUF0

地址：0x99

Bit	7	6	5	4	3	2	1	0
Name	SBUF0[7:0]							
Type	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bit	Name	Function
[7:0]	SBUF0	串口数据缓冲位 7:0 (MSB-LSB) 通过此寄存器可以访问 2 个 9 寄存器：1 个发送移位寄存器和 1 个接收锁存寄存器。当向 SBUF0 写入数据时，数据进入发送移位寄存器用于串行发送。向 SBUF0 写入一个字节会初始化发送操作。读取 SBUF0 会返回接收锁存器的值。

15. 增强型串行外设接口（SPI）

YS67FXXXX(X)的 SPI 可以作为主器件或从器件工作，可以使用 3 线或 4 线方式，并可在同一 SPI 总线上支持多个主器件和从器件。从选择信号（NSS）可被配置为输入以选择工作在从方式的 SPI_{in}，或在多主环境中禁止主方式操作，以避免两个以上主器件试图同时进行数据传输时发生 SPI 总线冲突。NSS 可以被配置为片选输出（在主方式），或在 3 线操作时被禁止。在主方式，可以用其他通用端口 I/O 引脚选择多个从器件。

15.1. 引脚说明

SPI 接口有 4 个信号：SCK/P3.0、MOSI/P3.1、MISO/P3.2、NSS/P3.3，或 SCK/P0.4、MOSI/P0.5、MISO/P0.6、NSS/P0.7，2 组只能选择其中一组使用。

15.1.1 主输出、从输入（MOSI）

主出从入（MOSI）信号是主器件的输出和从器件的输入，用于从主器件到从器件的串行数据传输。当 SPI 作为主器件时，该信号是输出；当 SPI 作为从器件时，该信号是输入。数据传输时最高位在先。当被配置为主器件时，MOSI 由移位寄存器的 MSB 驱动。

15.1.2 主输入、从输出（MISO）

主入从出（MISO）信号是从器件的输出和主器件的输入，用于从从器件到主器件的串行数据传输。当 SPI 作为主器件时，该信号是输入；当 SPI 作为从器件时，该信号是输出。数据传输时最高位在先。当 SPI 被禁止或工作在 4 线从方式而未被选中时，MISO 引脚被置于高阻态。当作为从器件工作在 3 线方式时，MISO 总是由移位寄存器的 MSB 驱动。

15.1.3 串行时钟（SCK）

串行时钟（SCK）信号是主器件的输出和从器件的输入，用于同步主器件和从器件之间在 MOSI 和 MISO 线上的串行数据传输。当 SPI 作为主器件时产生该信号。在 4 线从方式，当从器件未被选中时（NSS=1），SCK 信号被忽略。

15.1.4 从选择（NSS）

从选择（NSS）信号的功能取决于 SPICN 寄存器中 NSSMD1 和 NSSMD0 位的设置。

有 3 种可能的方式：

- NSSMD[1:0] = 00：3 线主方式或从方式：SPI 工作在 3 线方式，NSS 被禁止。当作为从器件工作在 3 线方式时，SPI 总是被选择。由于没有选择信号，SPI 工作在 3 线方式时必须是总线唯一的从器件。这种情况用于一个主器件和一个从器件之间点对点通信。
- NSSMD[1:0] = 01：4 线从方式或多主方式：SPI 工作在 4 线方式，NSS 被使能为输入。当作为从器件时，NSS 选择 SPI_{in} 器件。当作为主器件时，NSS 信号的负跳变禁止 SPI 的主器件功能，以便可以在同一个 SPI 总线上使用多个主器件。
- NSSMD[1:0] = 1x：4 线主方式：SPI 工作在 4 线方式，NSS 被使能为输出。NSSMD 的设置值决定 NSS 引脚的输出逻辑电平。这种配置只能在 SPI 作为主器件时使用。

15.2. SPI 主方式

SPI 总线上的所有数据传输都由 SPI 主器件启动。通过将主允许标志（MSTEN，SPI_CFG.6）置 1 将 SPI 置于主方式。当处于主方式时，向 SPI 数据寄存器（SPI_DAT）写入一个数据字节时是写发送缓冲器。如果 SPI

移位寄存器为空，发送缓冲器中的数据字节被传送到移位寄存器，数据传输开始。SPI 主器件立即在 MOSI 线上串行移出数据，同时在 SCK 上提供串行时钟。在传输结束后 SPIF (SPI_CNTL.7) 标志被置为逻辑 1。如果中断被允许，在 SPIF 标志置位时将产生一个中断请求。在全双工操作中，当 SPI 主器件在 MOSI 线向从器件发送数据时，被寻址的 SPI 从器件可以同时在 MISO 线上向主器件发送其移位寄存器中的内容。因此，SPIF 标志既作为发送完成标志又作为接收数据准备好标志。从从器件接收的数据字节以 MSB 在先的形式传送到主器件的移位寄存器。当一个数据字节被完全移入移位寄存器时，便被传送到接收缓冲器，处理器通过读 SPI_DAT 来读该缓冲器。

当被配置为主器件时，SPI 可以工作在下面的三种方式之一：多主方式、3 线单主方式或 4 线单主方式。当 NSSMD1 (SPI_CNTL.3) = 0 且 NSSMD0 (SPI_CNTL.2) = 1 时，是默认的多主方式。在该方式，NSS 是器件的输入，用于禁止主 SPI，以允许另一主器件访问总线。在该方式，当 NSS 被拉为低电平时，MSTEN (SPI_CNTL.6) 和 SPIEN (SPI_CNTL.0) 位被清 0，以禁止 SPI 主器件，且方式错误标志 (MODF, SPI_CNTL.5) 被置 1。如果中断被允许，将产生方式错误中断。在这种情况下，必须用软件重新使能 SPI。在多主系统中，当器件不作为系统主器件使用时，一般被默认为从器件。在多主方式，可以用通用 I/O 引脚对从器件单独寻址（如果需要）。图 1 给出了两个主器件在多主方式下的连接图。

当 NSSMD1 (SPI_CNTL.3) = 0 且 NSSMD0 (SPI_CNTL.2) = 0 时，SPI 工作在 3 线单主方式。在该方式，NSS 未被使用，也不被交叉开关映射到外部端口引脚。在该方式，应使用通用 I/O 引脚选择要寻址的任何从器件。图 2 给出了一个 3 线主方式主器件和一个从器件的连接图。

当 NSSMD1 (SPI_CNTL.3) = 1 时，SPI 工作在 4 线单主方式。在该方式，NSS 被配置为输出引脚，可被用作从选择信号去选中一个 SPI 器件。在该方式，NSS 的输出值由 NSSMD0 (SPI_CNTL.2) 控制（用软件）。可以用通用 I/O 引脚寻址另外的从器件。图 3 给出了一个 4 线主方式主器件和两个从器件的连接图。

15.3. SPI 从方式

当 SPI_{In} 被使能而未被配置为主器件时，它将作为 SPI 从器件工作。作为从器件，由主器件控制串行时钟 (SCK)，从 MOSI 移入数据，从 MISO 引脚移出数据。SPI 逻辑中的位计数器对 SCK 边沿计数。当 8 位数据经过移位寄存器后，SPIF 标志被置为逻辑 1，接收到的字节被复制到接收缓冲器。通过读 SPI_{In}DAT 来读取接收缓冲器中的数据。从器件不能启动数据传送。通过写 SPIDAT 来预装要发送给主器件的数据到移位寄存器。写往 SPI_DAT 的数据是双缓冲的，首先被放在发送缓冲器。如果移位寄存器为空，发送缓冲器中的数据会立即被传送到移位寄存器。当移位寄存器中已经有数据时，SPI 将在下一次（或当前）SPI 传输的最后一个 SCK 边沿过去后再将发送缓冲器的内容装入移位寄存器。

当被配置为从器件时，SPI 可以工作 4 线或 3 线方式。当 NSSMD1 (SPI_CNTL.3) = 0 且 NSSMD0 (SPI_CNTL.2) = 1 时，是默认的 4 线从方式。在 4 线方式，NSS 被分配到一个端口引脚并被配置为数字输入。当 NSS 为逻辑 0 时，SPI 被使能；当 NSS 为逻辑 1 时，SPI 被禁止。在 NSS 的下降沿，位计数器被复位。注意，对应每次字节传输，在第一个有效 SCK 边沿到来之前，NSS 信号必须被驱动到低电平至少两个系统时钟周期。图 3 给出了两个 4 线方式从器件和一个主器件的连接图。

当 NSSMD1 (SPI_CNTL.3) = 0 且 NSSMD0 (SPI_CNTL.2) = 0 时，SPI 工作在 3 线从方式。在该方式，NSS 未被使用，也不被交叉开关映射到外部端口引脚。由于在 3 线从方式无法唯一地寻址从器件，所以 SPI 必须是总线上唯一的从器件。需要注意的是，在 3 线从方式，没有外部手段对位计数器复位以判断是否收到一个完整的字节。只能通过用 SPIEN 位禁止并重新使能 SPI 来复位位计数器。图 2 给出了一个 3 线从器件和一个主器件的连接图。

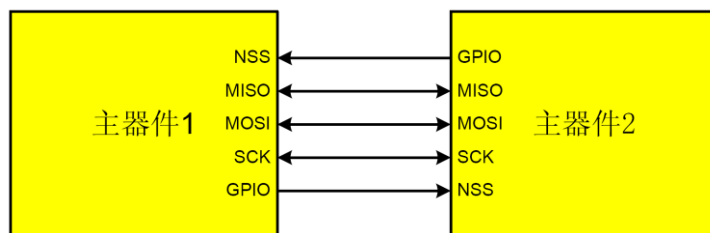


图16-1 多主方式连接图

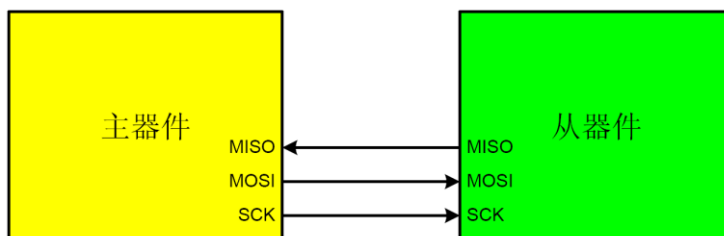


图16-2 线单主方式和3线单从方式

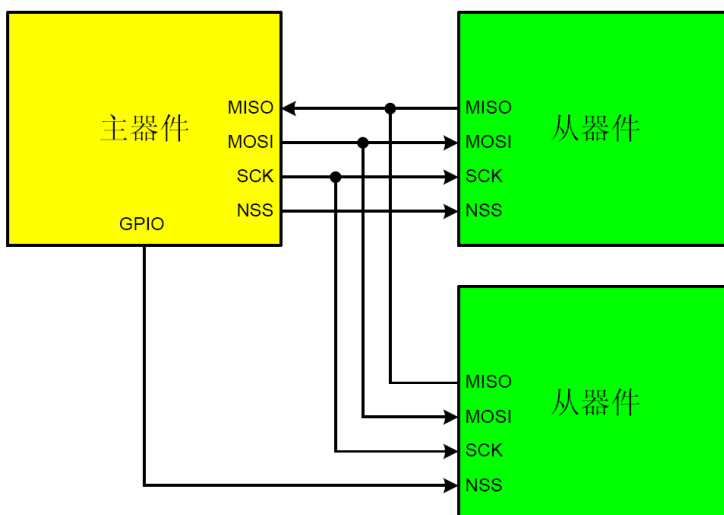


图16-3 线单主方式和4线从方式连接图

15.4. SPI 中断源

如果 SPI 中断被允许，在下述 4 个标志位被置 1 时将产生中断。

注意：这 4 个标志位都必须用软件清 0。

- 在每次字节传输结束时，SPI 中断标志 SPIF (SPI_CNTL.7) 被置 1。该标志适用于所有 SPI 方式。
- 如果在发送缓冲器中的数据尚未被传送到 SPI 移位寄存器时写 SPI_DAT，写冲突标志 WCOL (SPI_CNTL.6) 被置 1。发生这种情况时，写 SPIDAT 的操作被忽略，不会对发送缓冲器写入。该标志适用于所有 SPI 方式。
- 当 SPI 被配置为工作于多主方式的主器件而 NSS 被拉为低电平时，方式错误标志 MODF (SPI_CNTL.5) 被置 1。当发生方式错误时，SPI_CNTL 中的 MSTEN 和 SPIE 位被清 0，以禁止 SPI 并允许另一个主器件访问总线。
- 当 SPI 被配置为从器件并且一次传输结束，而接收缓冲器中还保持着上一次传输的数据未被读取时，接收溢出标志 RXOVRN (SPI_CNTL.4) 被置 1。新接收的字节将不被传送到接收缓冲器，允许前面接收的字节被读取。引起溢出的数据字节丢失。

15.5. SFR

15.5.1 SPICFG 寄存器

SPI 配置寄存器，地址 0XA1

位	7	6	5	4	3	2	1	0
名称	SPIBSY	MSTINT	CKPHA	CKPOL	SLVSEL	NSSIN	SRMT	RXBMT
类型	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	1	1	1

位	名称	功 能
7	SPIBSY	SPI 工作中 此位在 SPI 工作时，会设置为 1
6	MSTINT	主机模式使能位 0: 不使能主机模式，工作于从机模式 1: 使能主机模式，工作于主机模式
5	CHPHA	SPI 时钟相位选择 0: 数据采样点位于 SCK 周期的第一个时钟延 1: 数据采样点位于 SCK 周期的第二个时钟延
4	CKPOL	SPI 时钟极性选择 0: sck 线在不工作时处于低电平 1: sck 线在不工作时处于高电平
3	SLVSEL	从机选择标志位 当 NSS pin 输入为低时，表示从机是 SPI 主机选择的通讯对象，此位硬件置 1；当 NSS pin 输入为高（从机没有被选择）时，会被清为 0。
2	NSSIN	NSS 实时数据输入
1	SRMT	移位寄存器空标志(只在从机模式时有效)
0	RXBMT	接收暂存器空标志(只在从机模式时有效)

15.5.2 SPICTL 寄存器

SPI 控制寄存器，地址 0XF8

位	7	6	5	4	3	2	1	0
名称	SPIF	WCOL	MODF	RXOVRN	NSSMID[1:0]		TXBMT	SPIEN
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	1	1	0

位	名称	功 能
7	SPIF	SPI 中断标志位 当每次传输完一个数据（8bit）之后，这位将由硬件拉高。此位必须由软件清 0
6	WCOL	写冲突标志位 当 TXBMT 为 0 时，写入 SPIDAT 则将此位拉高，表明写冲突。 此位必须由软件清 0

5	MODF	模式错误标志位 当检测到主机模式冲突的时候将此位置为 1T (NSS is low, MSTINT = 1 and NSSMD[1:0]=01). 此位必须由软件清 0
4	RXOVRN	接收 overrun 标志(在主、从机模式下均有效)
3:2	NSSMD	从机选择模式 00: 3 线从方式或3 线主方式。NSS 信号不连到端口引脚。 01: 4 线从方式或多主方式（默认值）。NSS 是器件的输入。 1x: 4 线单主方式。NSS 信号被分配一个输出引脚并输出NSSMD0 的值
1	TXBMT	发送暂存器空标志 当新数据被写入发送缓冲器时，该位被清 0。当发送缓冲器中的数据被传送到 SPI 移位寄存器时，该位被置 1，表示可以安全地向发送缓冲器写新字节。
0	SPIEN	SPI 使能位 0: 禁止SPI 1: 使能SPI

15.5.3 SPISCR 寄存器

SPI 时钟速率寄存器，地址 0XA2

位	7	6	5	4	3	2	1	0
名称	SCR[7:0]							
类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值								

位	名称	功 能
7:0	SCR[7:0]	<p>SPI 时钟频率</p> <p>当SPI 模块被配置为工作于主方式时，这些位决定SCK 输出的频率。SCK 时钟频率是从系统时钟分频得到的，由下面的方程给出，其中：SYSCLK 是系统时钟频率，SPICKR 是spi_scr 寄存器中的8 位值。</p> $f_{SCK} = \text{SYSCLK} / 2 * (\text{SPICKR} + 1)$ $(2 \leq \text{spi_scr} \leq 255)$ <p>例如：如果SYSCLK = 2MHz，SPICKR = 0x04，则</p> $f_{SCK} = 2000000 / 2 * (4 + 1)$ $f_{SCK} = 200 \text{ kHz}$

15.5.4 SPIDAT 寄存器

SPI 数据寄存器，地址 0XA3

位	7	6	5	4	3	2	1	0
名称	SPIDAT							
类型	R	R	R/W	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	0	0	0	0	0

位	名 称	功 能
7:0	SPIDAT	SPI 发送和接收数据寄存器。 SPI_DAT 寄存器用于发送和接收SPI 数据。在主方式下，向SPI_DAT 写入数据时，数据被放到发送缓冲器并启动发送。读SPI_DAT 返回接收缓冲器的内容。

16. I2C

YS67FXXXX(X)的 I2C 接口是一个双线，双向的串行接口。本设计与 I2C 协议定义 3.0 相匹配。它可以运行于高速模式，最高可达到 3.4M。CPU 可以通过寄存器控制 I2C 模块读写数据。本接口同时支持 clock stretching，可以允许 CPU 实时控制发送和接收的每个 Byte 的数据。本 I2C 是自时钟的，可以在 sleep 模式下工作，并且当接收的地址与本机相匹配时可以唤醒 CPU。

注意：YS67FXXXX(X)的 I2C 接口只支持从机模式。

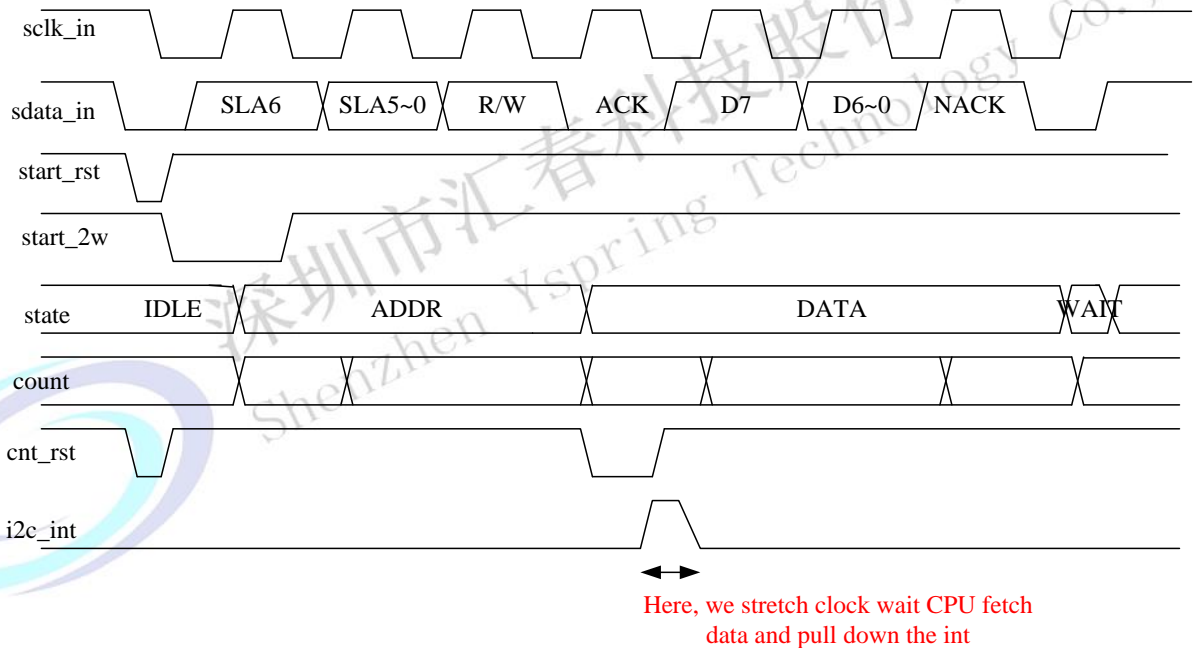


图17-1 I2C接收波形图

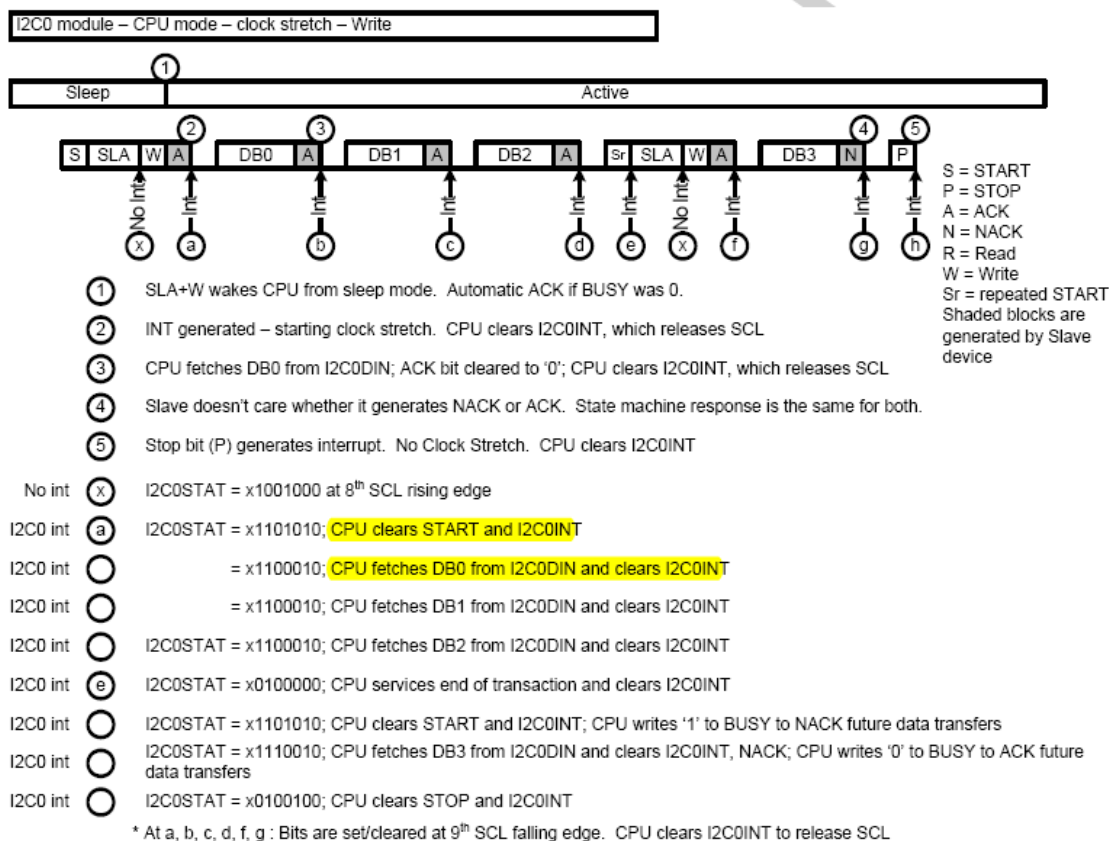


图 17-2 typical i2c write sequence in CPU Mode

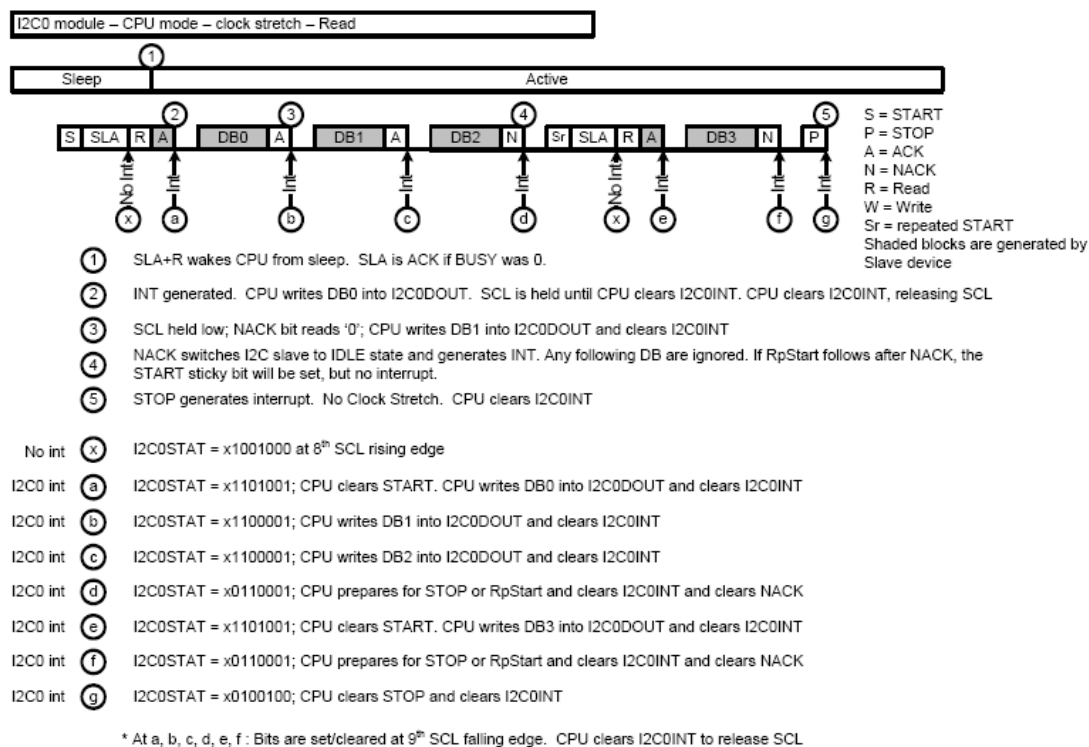
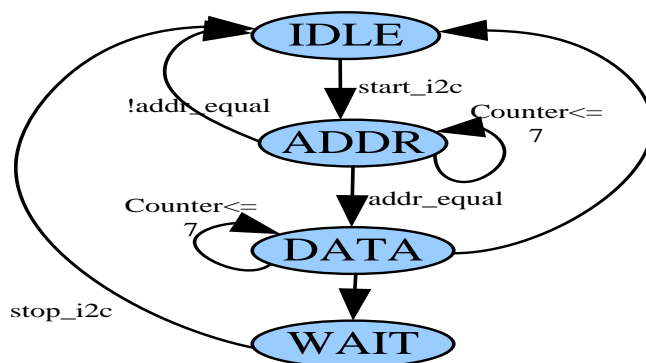


图 17-3 typical i2c read sequence in CPU Mode

从上图我们可以看出，i2c 协议自动进行 clock stretching 在每个发送或接收的 ACK 或 NACKbit 的 scl 时钟下降沿。此时，我们将相应的状态位拉高以告知 CPU 取数据或发送数据。CPU 在进行完上述动作后会将相应状态位拉低以结束 clock stretching，进入下一个步骤。

I2C 外设接口在每次上电后必须要先经过配置相应的寄存器才能使用 I2C。当接收到开始的标志后，将 start_i2c 拉高，进入到 ADDR 状态，在这个状态下接收主机端发送的地址，如果地址匹配，则进入下一个状态。如果地址不匹配，则 I2C 不动作，退回到 IDLE 状态。在 ADDR 状态，我们还同时接收读写标志位，决定是发送还是接收数据。每接收完一个 byte 的数据后，可根据是否接收到结束的标志决定是停留在 DATA 状态还是返回到 IDLE 状态。

注： 为了更稳定工作，外部SCL和SDA引脚上最好加上拉电阻。



I2C 外设接口不能再 power on 后直接使用，我们必须要先配置相应的寄存器来使能 I2C。

16.1. I2C 操作步骤

1. 设置 I2C_EN 为高，使能 I2C；
2. CPU 侦测 I2C 状态位，如果发现 I2C 中断拉高，CPU 将延长时钟并且接收或发送数据；
3. 当 CPU 完成上述工作，将复位 I2C 中断位，结束时钟延长；
4. 回到 步骤 2。

16.2. I2C 寄存器

I2C 总线有 6 个相关寄存器，分别是 FCTR,PER_EN,I2C_DIN,I2C_DOUT,I2C_SLAD,I2C_STAT，其中 FCTR 可设置 I2C 功能转移，PER_EN 可设置 I2C 使能及 I2C 通讯超时监控等功能，请查阅 IO 输入/输出端口寄存器说明。

16.2.1 I2C_DIN 寄存器

I2C 接受数据寄存器，地址 0XBA

位	7	6	5	4	3	2	1	0
名称	I2C_DIN[7:0]							
类型	R	R	R	R	R	R	R	R
复位值								

位	名称	功能
7:0	I2C_DIN[7:0]	I2C 接口从 master 端接收的数据

16.2.2 I2C_DOUT 寄存器

I2C 发送数据寄存器，地址 0XBB

位	7	6	5	4	3	2	1	0
名称	I2C_DOUT[7:0]							

类型	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值								

位	名称	功能
7:0	I2C_DOUT[7:0]	I2C 发送数据到 master 端

16.2.3 I2C_SLAD 寄存器

I2C 从机地址寄存器，地址 0XBC

位	7	6	5	4	3	2	1	0
名称	I2C_SLAD[6:0]							
类型	---	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位值								

位	名称	功能
6:0	I2C_SLAD[6:0]	I2C 从机地址.

16.2.4 I2C_STAT 寄存器

I2C 状态寄存器，地址 0XBD

位	7	6	5	4	3	2	1	0
名称	DMOD	ACTIVE	I2CINT	NACK	START	STOP	DATA	STRETCH
类型	R	R	R	R/W	R/W	R/W	R/W	R/W
复位值	0	0	0	x	0	0	0	0

位	名称	功 能
Bit 7	DMOD	用来指示当前 I2C 是读模式还是写模式； 0: 写模式 1: 读模式
Bit 6	ACTIVE	当前数据的地址匹配时，硬件自动置 1； 当 I2C 检测到 NACK 或 STOP 或者地址不匹配的 START 时，硬件清 0； I2C 没有被使能时，ACTIVE 恒为 0。
Bit 5	I2CINT	当 Bit[3:1]任何一位为高时，I2CINT 为高； 当 Bit[3:1]全为 0 时，I2CINT 为 0； I2C 没有被使能时，I2CINT 恒为 0。
Bit 4	NACK	用来指示当前发送/接收的字节的响应位； 0 表示 ACK； 1 表示 NACK； 当收到正确的地址后，硬件自动清 0； 写模式时，软件往此位写 1 将此位置 1，表示接下来的字节响应 NACK； 读模式时，此位反应的是主机发来的响应位。 CPU 确保 NACK 位写 0 无动作。
Bit 3	START	当收到匹配的地址字节后，硬件置 1； 软件往此位写 0 清除此位，I2C 没有被使能时，START 恒为 0。

Bit 2	STOP	当前数据的地址匹配时，收到 STOP，硬件置 1； 软件往此位写 0 清除此位，I2C 没有被使能时，STOP 恒为 0。
Bit 1	DATA	当前数据的地址匹配时，接收/发送完数据字节，硬件置 1； 软件往此位写 0 清除此位，I2C 没有被使能时，DATA 恒为 0。
Bit 0	STRETCH	在地址匹配的情况下，I2C 处理完每个字节的第 9 位时，硬件自动置 1，SCL 被硬件 Stretch；软件往此位写 0 清除此位，释放 SCL 的 Stretch。

i2c_stat 寄存器在 I2C 时钟域；

i2c_stat 寄存器 Bit[3:0]为异步复位，MCU 需保证复位信号无毛刺；

往 i2c_stat 寄存器的 Bit[4]写 1 将其置 1，异步置位，MCU 需保证置位信号无毛刺。

STRETCH: 在地址匹配的情况下，I2C 处理完每个字节的第 9 位时，硬件自动置 1，SCL 被硬件 Stretch；软件往此位写 0 清除此位，释放 SCL 的 Stretch。

DATA: 当前数据的地址匹配时，接收/发送完每个数据字节，硬件置 1；软件往此位写 0 清除此位，I2C 没有被使能时，DATA 恒为 0。

STOP: 当前数据的地址匹配时，收到 STOP，硬件置 1；软件往此位写 0 清除此位，I2C 没有被使能时，STOP 恒为 0。

START: 当收到匹配的地址字节后，硬件置 1；软件往此位写 0 清除此位，I2C 没有被使能时，START 恒为 0。

NACK: 用来指示当前发送/接收的字节的响应位，0 表示 ACK，1 表示 NACK；当收到正确的地址后，硬件自动清 0；

写模式时，软件往此位写 1 将此位置 1，表示接下来的字节响应 NACK（仅在写模式时起作用。在读模式时，NACK/ACK 由主机发送）；

读模式时，此位反应的是主机发来的响应位。

CPU 确保 NACK 位写 0 无动作。

I2CINT: 当 Bit[3:1]任何一位为高时，I2CINT 为高，当 Bit[3:1]全为 0 时，I2CINT 为 0；I2C 没有被使能时，I2CINT 恒为 0。

ACTIVE: 当前数据的地址匹配时，硬件自动置 1；当 I2C 检测到 NACK 或 STOP 或者地址不匹配的 START 时，硬件清 0；I2C 没有被使能时，ACTIVE 恒为 0。（当第 1 串地址匹配的数据不发 NACK 或 STOP 接着来一串地址不匹配的数据，硬件不会产生干扰）

DMOD: 用来指示当前 I2C 是读模式还是写模式，只读；0：写模式；1：读模式（被其它 I2C 设备读）。

17. 循环冗余检查单元 (CRC)

YS67FXXXX(X)的 CRC 采用 8 位并行算法, 与传统的 CRC 电路串行工作相比, 速度可提高到原来的 8 倍。并行算法是通过分析串行算法在连续移位 8 次的过程, 得出结果中每个 bit 与原始值的每个 bit 的关系, 从而直接用组合逻辑来表示结果。CRC 能使用 16 位或 32 位多项式执行 CRC。CRC 接收写到 CRC0IN 寄存器的 8 位数据流, 向一个内部寄存器输出 16 位或 32 位的结果。内部结果寄存器可以用 CRC0PNT 位和 CRC0DAT 寄存器间接访问。

17.1. CRC 寄存器

17.1.1 控制寄存器: CRC0CN

Bit	7	6	5	4	3	2	1	0
Name	(Reserved)			CRCDONE	CRC0INI	CRC0VAL	AUTOINT	CRC0PNT
Type	R	R	R	R	R/W	R/W	R/W	R/W
Reset	0	0	0	1	0	0	0	0

地址: 0x9A

Bit	Name	Function
[7:5]	Reserved	保留位, 只读, 读为 0。
[4]	CRCDONE	自动 CRC 计算完成标志。 在自动 CRC 计算模式过程中, 硬件自动将这一位写 0, 并且软件代码也会停止执行; 在其它情况下, 硬件自动将这一位置为 1, 所以, 软件读取这一位始终返回 1。
[3]	CRC0INI	CRC 结果初始化使能 0: 初始化无效 1: 初始化有效; 当软件向这一位写 1 时, 硬件并没有真正将 1 写入此位, 而是同步产生一个时钟周期的高电平脉冲, 送到 CRC 引擎, 作为 CRC 结果初始化的条件。所以, 不管软件向这一位写入什么值, 读取时总是返回 0。
[2]	CRC0VAL	CRC 结果初始化选择位。 0: 将 CRC 结果初始化为 0x0000 1: 将 CRC 结果初始化为 0xFFFF
[1]	AUTOINT	CRC 自动计算使能。 当向此位写 1 时, 会自动对 Flash 的某片连续的块中的数据进行 CRC 计算。计算的起始块为 CRC0ST, 共计算 CRC0CNT 个块。 注: 在启用自动 CRC 计算功能之前, 应先将其它位配置好, 再将这一位写 1。换句话说, 这一位不能与其它位同时配置。(不需要?)
[0]	CRC0PNT	CRC 结果指针。 0: 读取 CRC0DATA 寄存器时, 访问的是 16 位 CRC 结果的低字节 1: 读取 CRC0DATA 寄存器时, 访问的是 16 位 CRC 结果的高字节

注：在应用时，如果要判断读取的是 CRC 结果的低字节还是高字节，应该要先读取 CRC 结果，再判断这一位是 0 还是 1。如果是 0，则说明刚才读取的是低字节；如果是 1，则说明刚才读取的是高字节。即遵循“先读结果，再根据指针判断”的原则；而不是先看指针，再读 CRC 结果。

另外，在上位机单步运行调试时，由于上位机本身会读，如果软件再读，则会发现读出来的永远是高字节或低字节。

注：由于 CRC 计算过程分为两大类，一类是单个字节的 CRC 计算，一类是 ROM 数据批量 CRC 自动计算。向控制寄存器 CRC0CN 的 bit[1]写入 1，会立即启动 CRC 自动计算过程。如果要计算软件写入 CRC0IN 寄存器中的单个字节的 CRC 值，则 CRC0CN 寄存器的 bit[1]不能为 1。

17.1.2 输入数据寄存器：CRC0IN

Bit	7	6	5	4	3	2	1	0
Name	CRC0IN[7:0]							
Type	W							
Reset	0	0	0	0	0	0	0	0

地址：0x9B

Bit	Name	Function
[7:0]	CRC0IN	CRC 模块输入数据。 每次向此寄存器写入一个数据时，CRC 模块就自动在现有 CRC 结果的基础上，根据输入数据计算出新的 CRC 结果，并覆盖原 CRC 结果。 注：此寄存器是一个虚拟寄存器，写入的数据并不保存。读取此地址时返回 0x00。

17.1.3 结果输出寄存器：CRC0DAT

Bit	7	6	5	4	3	2	1	0
Name	CRC0DAT[7:0]							
Type	R/W							
Reset	0	0	0	0	0	0	0	0

地址：0x9C

Bit	Name	Function
[7:0]	CRC0DAT	CRC 结果输出。 每次读、写此寄存器时，会根据控制寄存器 CRC0CN 中的结果指针 CRC0PNT 来决定访问的是 CRC 结果的高字节还是低字节。需要注意的是，每次读或写此寄存器，都会导致指针 CRC0PNT 变化一次。

注：由于此寄存器的值除了直接由软件决定以外，还可由其它信号导致发生变化，所以直接放在 CRC 模块内部，

而不放在寄存器专用模块里。

17.1.4 自动计算起点寄存器：CRC0ST

Bit	7	6	5	4	3	2	1	0
Name	Reserved	CRC0ST[6:0]						
Type	R/W							
Reset		0	0	0	0	0	0	0

地址：0x9D

Bit		Name	Function
[7]	Reserved	保留位	
[6:0]		CRC0ST	<p>自动计算 CRC 的 ROM 起始 sector。</p> <p>计算的起始地址是：$CRC0ST \times \text{Sector Size}$</p> <p>例如：如果 CRC0ST[6:0]的值是 1，每个 Sector size 是 64 个字节，则自动 CRC 计算的起始地址是：$1 \times 64 = 64$，实际上是从第二个 sector 的第一个字节开始。</p>

17.1.5 扇区控制寄存器：CRC0CNT

Bit	7	6	5	4	3	2	1	0
Name	Reserved	CRC0CNT[6:0]						
Type		R/W						
Reset	NA	0	0	0	0	0	0	0

地址：0x9E

Bit	Name	Function
[7]	Reserved	保留位
[6:0]	CRC0CNT	<p>自动 CRC 计算扇区(Sector)总数。</p> <p>此值定义了需要计算 CRC 值的 ROM 中扇区总数，例如：0x00 表示 1 个扇区；0x7F 表示 128 个扇区；需要计算 CRC 的最后一个扇区的起始地址是：</p> $(CRC0ST + CRC0CNT) \times \text{SectorSize}$ <p>其中，每个扇区(Sector)为 64 个字节。</p>

17.2. 操作步骤

17.2.1 计算单个字节的 CRC

要计算单个字节的 CRC 值，请按以下步骤进行：

- (1)、向控制寄存器 CRC0CN 的 bit[1]写 0，使 CRC 模块工作在单字节计算模式下；
- (2)、根据需要，向控制寄存器 CRC0CN 写入适当的值。例如，写入 0xFD，则 bit[7:5]由于是保留位，故不起作用；bit[0]为 1，使得 CRC 结果指针指向 16 位结果的高字节；bit[2]为 1，使得 CRC 结果初始化为 0xFFFF；bit[3]为 1 使得 CRC 结果初始化有效；
- (3)、向输入数据寄存器 CRC0IN 写入一个数据，例如 0x63，则在下一个时钟周期内，CRC 结果马上就被计算出来，且结果为 0xBD35；
- (4)、读取 CRC 结果：软件读取结果输出寄存器 CRC0DAT，得到高字节数据 0xBD；再读一次，得到低字节数据 0x35；合并起来就是正确的 CRC 结果。
- (5)、如果在计算 CRC 之前，不想让初始化结果是 0x0000 或 0xFFFF，也可以先设定好 CRC 结果寄存器访问指针，然后向结果寄存器中写入自定义的初始化值。例如，先将指针定位在低字节，然后向 CRC0DAT 寄存器写入数据 0xCF，再次写入 0xD4，则 CRC 结果被初始化成 0xD4CF；向 CRC0IN 寄存器写入 0xD1，则在下一个时钟周期内，计算出 0xD1 的 CRC 值为 0x9FA5。

17.2.2 批量计算 ROM 数据 CRC

要计算 ROM 中某片连续区域数据的 CRC 值，请按以下步骤进行：

- (1)、向控制寄存器 CRC0CN 的 bit[1]写 0，停止自动计算模式；
- (2)、向控制寄存器 CRC0CN 的其它位(保持 bit[1]为 0)写入适当值，设置结果访问指针、结果初始化值，并使能初始化；
- (3)、向自动控制寄存器 CRC0ST 写入适当值，设置起始地址；
- (4)、向扇区控制寄存器 CRC0CNT 写入适当值，设置需要计算 CRC 值的扇区总数；
- (5)、向控制寄存器 CRC0CN 的 bit[1]写 1，保持其它位不变，会启动自动计算过程。

下图是进行 ROM 数据 CRC 计算时，访问的分区图。

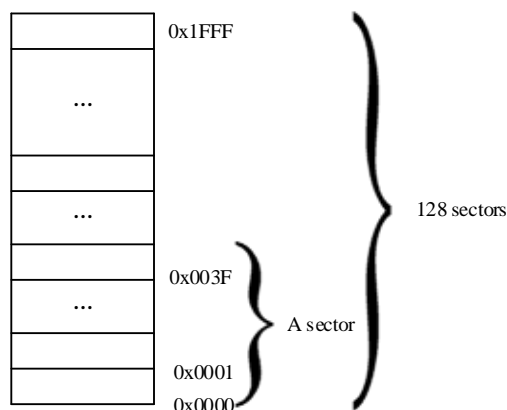


图 18-10 ROM 访问分区图

整个 ROM 共 8K 字节，分成 128 个 sector，编号从 sector0 到 sector127，每个 sector 包含 64 个字节。在进行 CRC 批量计算时，起始 sector 的值 CRC0ST 可以是 0x00~0x7F 之间的任何值，包括 0x00 和 0x7F；需要计算的 sector 总数的数值 CRC0CNT 可以是 0x00~0xFF，包括 0x00 和 0xFF。

需要注意的是，随着 CRC0ST 的值的增大，CRC0CNT 的值应该相应减小。例如，如果 CRC0ST 的值为 0xFF，则 CRC0CNT 的值只能是 0x00，即只能计算最后一个 sector 中数据的 CRC 值。此时，如果不小心将 CRC0CNT 的值设置为 0x01 或更大的值，则 CRC 控制器硬件会自动限制计算的字节数，使 CRC 引擎只计算最后一个 sector 中数据的 CRC 值。

18. LCD/LED

18.1. 概述

LCD/LED 功能包括一个显示控制器、显示存储阵列及对应 IO 输出引脚，以达到段式 LCD/LED 显示目的。增加了全部显示和关闭显示的功能。

18.2. 功能列表

- 支持 LCD/LED 功能，可选 LCD 或 LED
- 支持 LCD 1/3 BIAS
- 支持 LCD 3COM-24SEG, 4COM-23SEG, 5COM-22SEG, 6COM-21SEG, 8COM-19SEG，可选
- 支持 LCD/LED 显示频率预分频 1/2、1/4、1/8 或 1/16
- 支持 LED 8COM-8SEG
- 支持 LED 共阴极/共阳极设置
- LCD/LED 存储阵列，MCU 以 XRAM 方式访问。
- 支持 LCD/LED 全部显示和关闭显示。

18.3. 功能框图

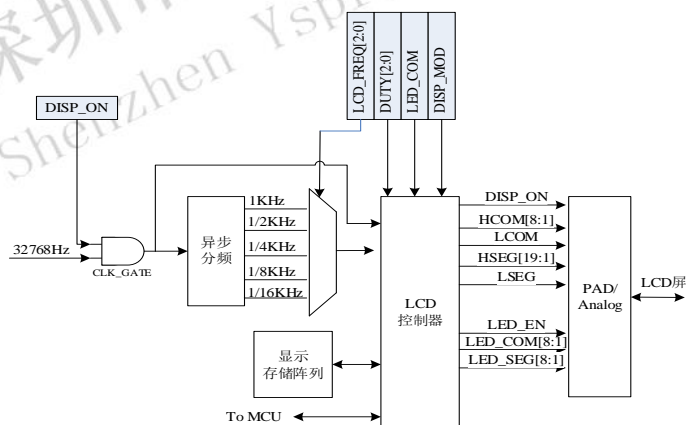
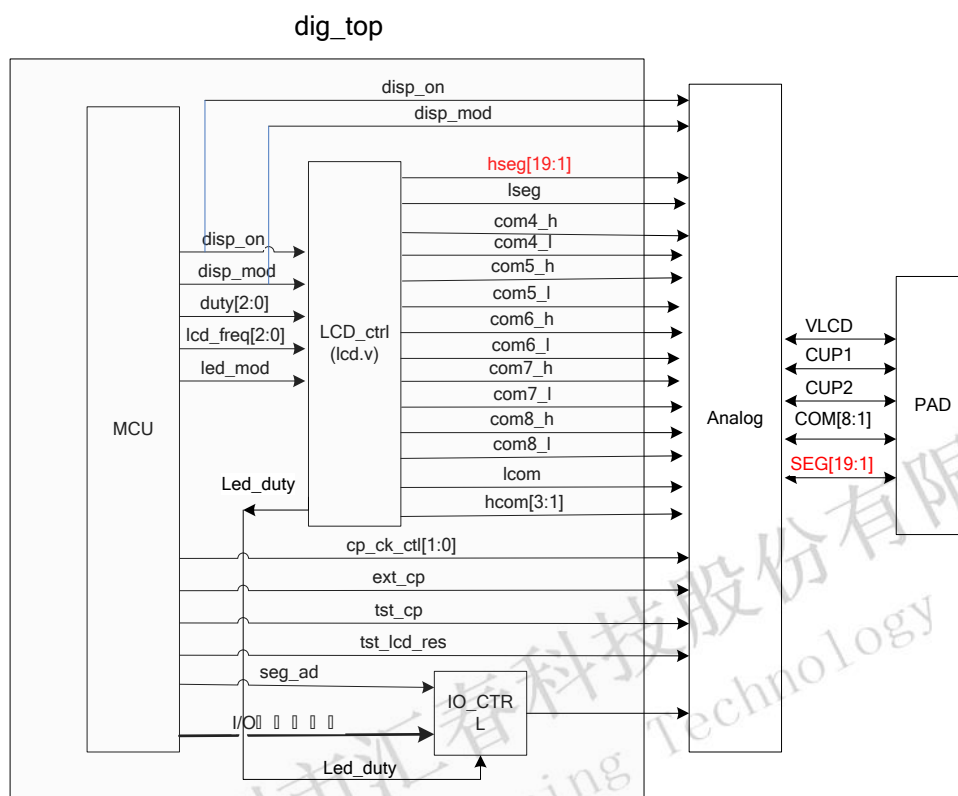


图 19-1 LCD 功能框图

18.4. 模块端口定义

18.4.1 LCD 模块 IO 接口简图:



18.5. 功能详述

18.5.1 LCD 直流分压电平的选择关系表

直流分压电平			1/3 Bias	HCOM,LCOM	HSEG,LSEG
前半扫描 周期	COM driver	选电平	VLCD	2'b11	/
		非选电平	1/3 VLCD	2'b01	/
	SEG driver	选电平	GND	/	2'b00
		非选电平	2/3 VLCD	/	2'b10
后半扫描 周期	COM driver	选电平	GND	2'b00	/
		非选电平	2/3 VLCD	2'b10	/
	SEG driver	选电平	VLCD	/	2'b11
		非选电平	1/3 VLCD	/	2'b01

18.5.2 LCD 波形图

LCD 波形(1/4 DUTY, 1/3 BIAS)

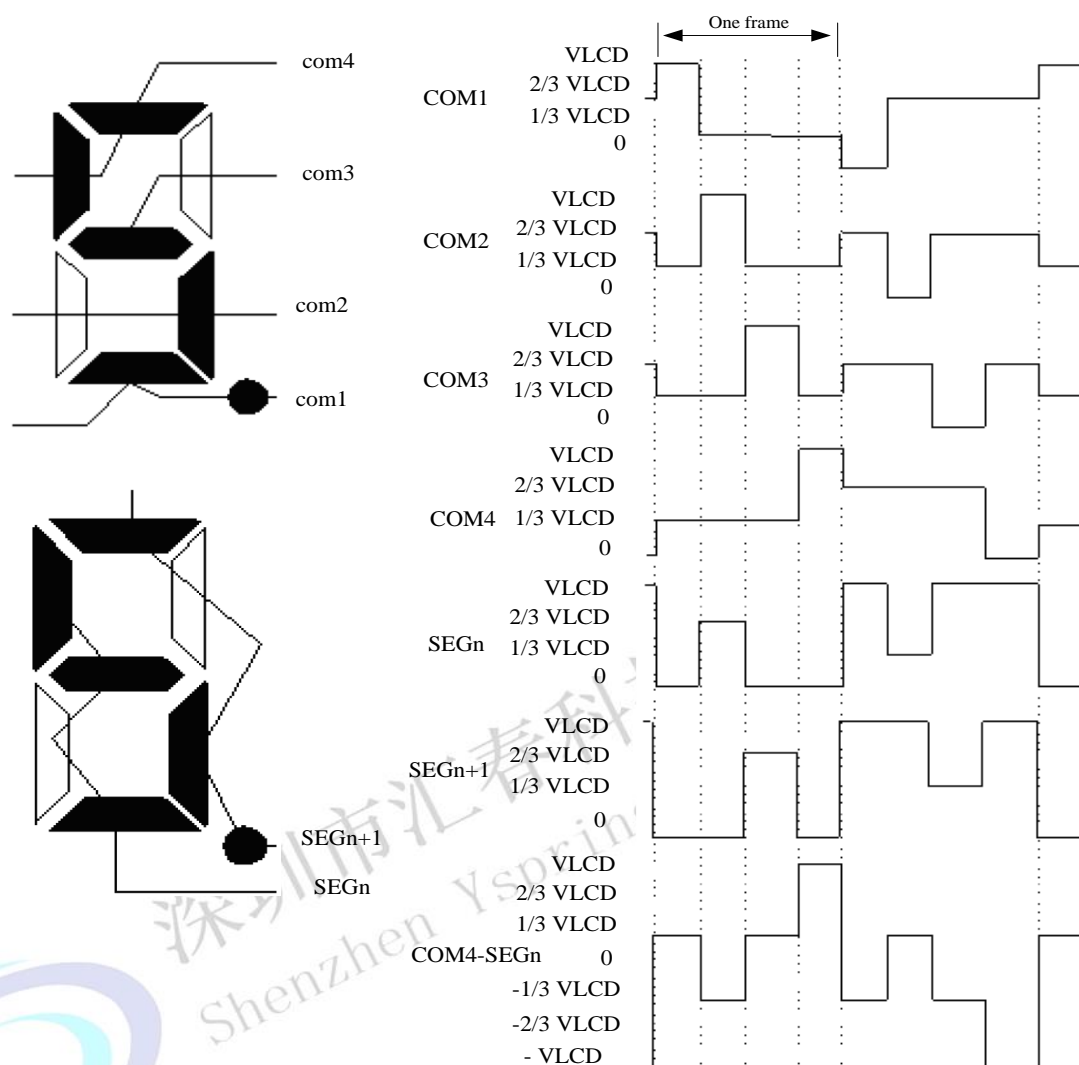


图 19-2 LCD 波形图

18.5.3 LCD 显示存储结构

LCD 1/3 Duty, 1/3Bias(COM1-3, SEG1-24)

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	-	-	-	-	-	SEG1	SEG1	SEG1
0x1121	-	-	-	-	-	SEG2	SEG2	SEG2
0x1122	-	-	-	-	-	SEG3	SEG3	SEG3
0x1123	-	-	-	-	-	SEG4	SEG4	SEG4
0x1124	-	-	-	-	-	SEG5	SEG5	SEG5
0x1125	-	-	-	-	-	SEG6	SEG6	SEG6
0x1126	-	-	-	-	-	SEG7	SEG7	SEG7
0x1127	-	-	-	-	-	SEG8	SEG8	SEG8
0x1128	-	-	-	-	-	SEG9	SEG9	SEG9

0x1129	-	-	-	-	-	SEG10	SEG10	SEG10
0x112A	-	-	-	-	-	SEG11	SEG11	SEG11
0x112B	-	-	-	-	-	SEG12	SEG12	SEG12
0x112C	-	-	-	-	-	SEG13	SEG13	SEG13
0x112D	-	-	-	-	-	SEG14	SEG14	SEG14
0x112E	-	-	-	-	-	SEG15	SEG15	SEG15
0x112F	-	-	-	-	-	SEG16	SEG16	SEG16
0x1130	-	-	-	-	-	SEG17	SEG17	SEG17
0x1131	-	-	-	-	-	SEG18	SEG18	SEG18
0x1132	-	-	-	-	-	SEG19	SEG19	SEG19
0x1133	-	-	-	-	-	SEG20	SEG20	SEG20
0x1134	-	-	-	-	-	SEG21	SEG21	SEG21
0x1135	-	-	-	-	-	SEG22	SEG22	SEG22
0x1136	-	-	-	-	-	SEG23	SEG23	SEG23
0x1137	-	-	-	-	-	SEG24	SEG24	SEG24

LCD 1/4 Duty, 1/3Bias(COM1-4, SEG1-23)

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	-	-	-	-	SEG1	SEG1	SEG1	SEG1
0x1121	-	-	-	-	SEG2	SEG2	SEG2	SEG2
0x1122	-	-	-	-	SEG3	SEG3	SEG3	SEG3
0x1123	-	-	-	-	SEG4	SEG4	SEG4	SEG4
0x1124	-	-	-	-	SEG5	SEG5	SEG5	SEG5
0x1125	-	-	-	-	SEG6	SEG6	SEG6	SEG6
0x1126	-	-	-	-	SEG7	SEG7	SEG7	SEG7
0x1127	-	-	-	-	SEG8	SEG8	SEG8	SEG8
0x1128	-	-	-	-	SEG9	SEG9	SEG9	SEG9
0x1129	-	-	-	-	SEG10	SEG10	SEG10	SEG10
0x112A	-	-	-	-	SEG11	SEG11	SEG11	SEG11
0x112B	-	-	-	-	SEG12	SEG12	SEG12	SEG12
0x112C	-	-	-	-	SEG13	SEG13	SEG13	SEG13
0x112D	-	-	-	-	SEG14	SEG14	SEG14	SEG14
0x112E	-	-	-	-	SEG15	SEG15	SEG15	SEG15
0x112F	-	-	-	-	SEG16	SEG16	SEG16	SEG16
0x1130	-	-	-	-	SEG17	SEG17	SEG17	SEG17
0x1131	-	-	-	-	SEG18	SEG18	SEG18	SEG18
0x1132	-	-	-	-	SEG19	SEG19	SEG19	SEG19
0x1133	-	-	-	-	SEG20	SEG20	SEG20	SEG20
0x1134	-	-	-	-	SEG21	SEG21	SEG21	SEG21
0x1135	-	-	-	-	SEG22	SEG22	SEG22	SEG22
0x1136	-	-	-	-	SEG23	SEG23	SEG23	SEG23
0x1137	-	-	-	-	-	-	-	-

LCD 1/5 Duty, 1/3Bias(COM1-5, SEG1-22)

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	-	-	-	SEG1	SEG1	SEG1	SEG1	SEG1
0x1121	-	-	-	SEG2	SEG2	SEG2	SEG2	SEG2
0x1122	-	-	-	SEG3	SEG3	SEG3	SEG3	SEG3
0x1123	-	-	-	SEG4	SEG4	SEG4	SEG4	SEG4
0x1124	-	-	-	SEG5	SEG5	SEG5	SEG5	SEG5
0x1125	-	-	-	SEG6	SEG6	SEG6	SEG6	SEG6
0x1126	-	-	-	SEG7	SEG7	SEG7	SEG7	SEG7
0x1127	-	-	-	SEG8	SEG8	SEG8	SEG8	SEG8
0x1128	-	-	-	SEG9	SEG9	SEG9	SEG9	SEG9
0x1129	-	-	-	SEG10	SEG10	SEG10	SEG10	SEG10
0x112A	-	-	-	SEG11	SEG11	SEG11	SEG11	SEG11
0x112B	-	-	-	SEG12	SEG12	SEG12	SEG12	SEG12
0x112C	-	-	-	SEG13	SEG13	SEG13	SEG13	SEG13
0x112D	-	-	-	SEG14	SEG14	SEG14	SEG14	SEG14
0x112E	-	-	-	SEG15	SEG15	SEG15	SEG15	SEG15
0x112F	-	-	-	SEG16	SEG16	SEG16	SEG16	SEG16
0x1130	-	-	-	SEG17	SEG17	SEG17	SEG17	SEG17
0x1131	-	-	-	SEG18	SEG18	SEG18	SEG18	SEG18
0x1132	-	-	-	SEG19	SEG19	SEG19	SEG19	SEG19
0x1133	-	-	-	SEG20	SEG20	SEG20	SEG20	SEG20
0x1134	-	-	-	SEG21	SEG21	SEG21	SEG21	SEG21
0x1135	-	-	-	SEG22	SEG22	SEG22	SEG22	SEG22
0x1136	-	-	-	-	-	-	-	-
0x1137	-	-	-	-	-	-	-	-

LCD 1/6 Duty, 1/3Bias(COM1-6, SEG1-21)

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	-	-	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
0x1121	-	-	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
0x1122	-	-	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
0x1123	-	-	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
0x1124	-	-	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5
0x1125	-	-	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
0x1126	-	-	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
0x1127	-	-	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
0x1128	-	-	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
0x1129	-	-	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
0x112A	-	-	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
0x112B	-	-	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
0x112C	-	-	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
0x112D	-	-	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
0x112E	-	-	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15

0x112F	-	-	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
0x1130	-	-	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
0x1131	-	-	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
0x1132	-	-	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19
0x1133	-	-	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20
0x1134	-	-	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21
0x1135	-	-	-	-	-	-	-	-
0x1136	-	-	-	-	-	-	-	-
0x1137	-	-	-	-	-	-	-	-

LCD 1/8 Duty, 1/3Bias(COM1-8, SEG1-19)

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
0x1121	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
0x1122	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
0x1123	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
0x1124	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5
0x1125	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
0x1126	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
0x1127	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
0x1128	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
0x1129	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
0x112A	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
0x112B	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
0x112C	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
0x112D	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
0x112E	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15
0x112F	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
0x1130	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
0x1131	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
0x1132	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19
0x1133	-	-	-	-	-	-	-	-
0x1134	-	-	-	-	-	-	-	-
0x1135	-	-	-	-	-	-	-	-
0x1136	-	-	-	-	-	-	-	-
0x1137	-	-	-	-	-	-	-	-

LCD 1/3、1/4、1/5、1/6、1/8 Duty 依次对应的 24SEG、23SEG、22SEG、21SEG 及 19SEG

XRAM	7	6	5	4	3	2	1	0
Addr.	COM8	COM7	COM6	COM5	COM4	COM3	COM2	COM1
0x1120	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
0x1121	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
0x1122	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
0x1123	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
0x1124	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5

0x1125	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
0x1126	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
0x1127	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
0x1128	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
0x1129	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
0x112A	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
0x112B	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
0x112C	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
0x112D	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
0x112E	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15
0x112F	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
0x1130	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
0x1131	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
0x1132	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19
0x1133	-	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20
0x1134	-	-	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21
0x1135	-	-	-	SEG22	SEG22	SEG22	SEG22	SEG22
0x1136	-	-	-	-	SEG23	SEG23	SEG23	SEG23
0x1137	-	-	-	-	-	SEG24	SEG24	SEG24

18.5.4 LED 功能及波形图

LED 共阴/共阳可选择

LED_MOD=0, 共阴极, COM=0, SEG=1 时 LED 导通

LED_MOD=1, 共阳极, COM=1, SEG=0 时 LED 导通

LED 波形图(LED_MOD=0, 1/4 duty)

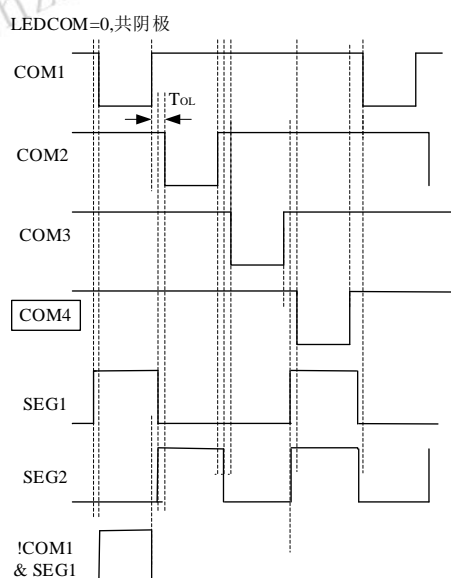


图 19-3 LED 波形图

注: TOL 为 LED COMMON 信号间的重叠时间, 取值范围: 20us-40us。

LED 1/4 占空比(LED_C1-4, LED_S1-8)

XRAM Addr.		7	6	5	4	3	2	1	0
0x1120	COM1	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1121	COM2	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1122	COM3	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1123	COM4	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1

LED 1/8 占空比(LED_C1-8,LED_S1-8)

XRAM Addr.		7	6	5	4	3	2	1	0
0x1120	COM1	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1121	COM2	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1122	COM3	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1123	COM4	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1124	COM5	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1125	COM6	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1126	COM7	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1
0x1127	COM8	SEG8	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1

18.6. 寄存器说明

寄存器	地址	说明
disp_mod	0xFD[1]	LCD/LED Select control 0: Enable LCD display driver, disable LED display driver. (default) 1: Enable LED display driver, disable LCD display driver
led_mod	0xFD[0]	LED COMMON mode 0: 共阴极 (default) 1: 共阳极
ext_cp	0xAF[2]	VLCD 电压供电模式 0: VLCD 电压由芯片内部 Charge Pump 提供, 默认值 1: VLCD 电压由片外提供
cpck_ctl[1:0]	0xAF[1:0]	LCD Charge Pump 分频输入.
tst_cp	0xf7[1]	Charge Pump Test Enable 0: Charge Pump 处于正常模式, 默认值 1: Charge Pump 进入测试模式, 如果此时 EXT_CP=0, 则 Charge Pump 使能
tst_lcd_res	0xf7[0]	LCD 偏置电阻测试使能 0: 正常模式, 默认值 1: 测试 LCD 偏置电阻

SFR.	DISP_CFG: 显示配置寄存器				
Bit	7:4	3	2	1	0
Name	/	all_on	all_off	disp_mod	led_mod
Type	/	W/R	W/R	W/R	W/R
Reset	4'h00	1'b0	1'b0	1'b0	1'b0
SFR Addr. = 0xFD					
Bit	Name	Description			

[7:2]	Reverse	读 0, 写无效
[3]	all_on	0: 在 disp_on=1 时, LCD/LED 正常显示; 1: 在 disp_on=1 时, LCD/LED 全部显示;
[2]	all_off	0: 在 disp_on=1 时, LCD/LED 正常显示; 1: 在 disp_on=1 时, LCD/LED 关闭显示; 注: all_on 和 all_off 同时为 1 时, 只有 all_on 有效。
[1]	disp_mod	LCD/LED Select control 0: Enable LCD display driver, disable LED display driver. (default) 1: Enable LED display driver, disable LCD display driver
[0]	led_mod	LED COMMON mode 0: 共阴极 (default) 1: 共阳极

SFR.	DISP_EN: display enable			
Bit	7	6	5:3	2:0
Name	disp_on	/	duty[2:0]	lcd_freq[2:0]
Type	W/R	/	W/R	W/R
Reset	0	0	3'b100	3'b001
SFR Addr. = 0xFE				
Bit	Name	Description		
[7]	disp_on	LCD/LED enable 0: Disable LCD/LED display driver 1: Enable LCD/LED display driver		
[6]	Reverse			
[5:3]	duty[2:0]	LCD duty configuration: 000: 1/3 duty, 1/3 bias, support LCD 24 SEG 001: 1/4 duty, 1/3 bias, support LCD 23 SEG 010: 1/5 duty, 1/3 bias, support LCD 22SEG 011: 1/6 duty, 1/3 bias, support LCD 21 SEG 100: 1/8 duty, 1/3 bias, support LCD 19 SEG (default) Other: 1/8 duty, 1/3 bias, support LCD 19SEG LED duty configuration: xx0: 1/4 duty xx1: 1/8 duty		

[2:0]

lcd_

freq[2:0]

Frame frequency setting

LCD/LED driver clock pre-scale is 32768Hz/32

Frame Frequency = Duty * DIV * 32768 Hz / 32

3'b000: Frame frequency is : duty * 1 * 32768Hz / 32

3'b001: Frame frequency is : duty * 1/2 * 32768Hz / 32 (default)

3'b010: Frame frequency is : duty * 1/4 * 32768Hz / 32

3'b011: Frame frequency is : duty * 1/8 * 32768Hz / 32

3'b100: Frame frequency is : duty * 1/16 * 32768Hz / 32

FREQ[2:0]	DIV	Frame Frequency (Hz)				
		1/3 duty	1/4 duty	1/5 duty	1/6 duty	1/8 duty
000	1	341.3	256	204.8	170.7	128
001	1/2	170.7	128	102.4	85.3	64
010	1/4	85.3	64	51.2	42.7	32
011	1/8	42.7	32	25.6	21.3	16
100	1/16	21.3	16	12.8	10.7	8

19. 电容感应(CS16)

19.1. 概述

电容感应模块(Capacitive Sense of 16Bit, 以下称 CS16)采用自电容感应原理, 实现最多 8 个通道 16bit ADC 触控感应功能, 集成自适应阈值控制, 配合 MCU 使用, 能够满足不同场合的单品触控要求。

19.2. 功能列表

- 8 个可选择感应通道, 非选通道可设置为 GPIO
- 12/13/14/16bit SAR ADC
- 支持 1 次、4 次、8 次、16 次、32 次、64 次采样平均, 以提高信噪比
- 自适应阈值更新功能, 支持 WAKE-ON-TOUCH 功能
- 可设置门限、可设置门限差值, 支持可调节感应强度功能。
- 全通道感应功能
- 电容感应值自动正向比较、反向比较, 支持 WAKE-ON-TOUCH 功能
- 虚拟感应功能, 以提高信噪比
- PIN Monitor 用于监控 I/O 变化, 以降低干扰, 提高信噪比

20. ADC

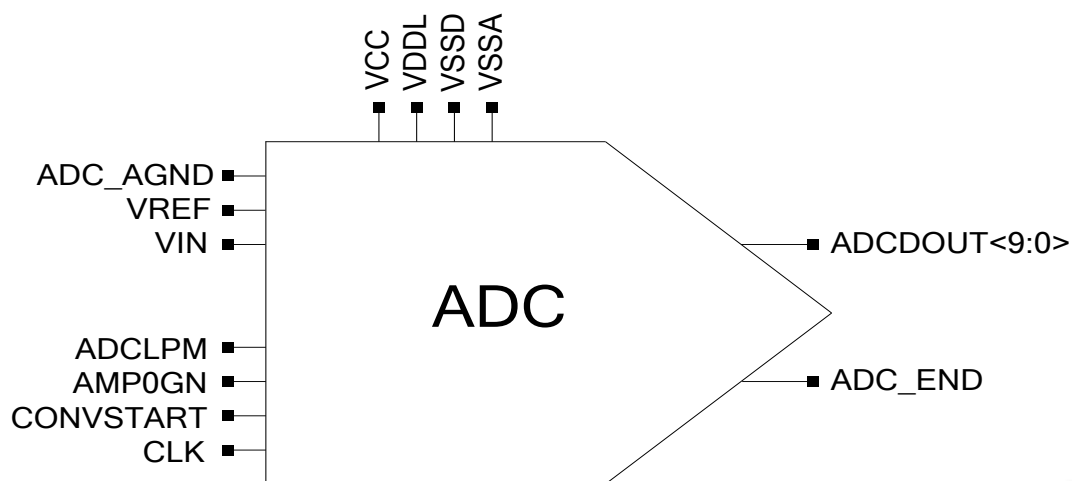


图 21-1

20.1. 引脚描述

引脚名称	类型	方向	描述
VCC	P	I/O	Power of the chip.(2.5~5.5)
VDDL	P	I/O	Power of the digital.(1.62~2.0)
VSSD	P	I/O	Ground of the digital.
VSSA	P	I/O	Ground of the analog.
ADC_AGND	P	I/O	Ground of the ADC.
VREF	A	I/O	Reference of the ADC.(2V 3V 4V)
VIN	A	I/O	Input of the ADC for sampling voltage.
NIB_ADC_COMP	A	I/O	the reference current of the buffer (OP) for common-mode voltage
PIB1_ADC_COMP	A	I/O	The reference current of the comparator of the ADC
PIB2_ADC_COMP	A	I/O	The reference current of the comparator of the ADC
DEM_EN	D	I	Enabel signal of the temperature decoder for the ADC.
ADJ_TEM_DEC<1:0>	D	I	Temperature decoder bits for ADC

20.2. 低功耗模式（Low Power Mode）

当 ADC 工作在低速 SAR 时钟时，SAR 转换提供了低功耗模式减少了电流消耗。配置 AD0LPM=1，使能低功耗模式。一般来说，当 SAR 转换时钟频率低于 4MHz 时，推荐使用低功耗模式。

20.3. ADC0 模拟多路选择器

AMUX0 选择 ADC 的输入通道。可参考寄存器 ADC0MX。

20.4. 寄存器说明

SAR ADC 模块设置了以下寄存器：

寄存器名	地址	有效位宽	复位值	说 明
ADC0CN	0X95	6	0X00	ADC0 控制寄存器
ADC0CF	0X96	8	0XF8	ADC0 配置寄存器
ADC0AC	0X97	8	0X00	ADC0 累加器配置寄存器
ADC0PWR	0XA5	6	0X0F	ADC0 突发模式上电时间寄存器
ADC0TK	0XA6	6	0X1E	ADC0 突发模式跟踪时间寄存器
ADC0H	0XAA	8	0X00	ADC0 数据字高字节寄存器
ADC0L	0XA9	8	0X00	ADC0 数据字低字节寄存器
ADC0MX	0XA7	4	0X0F	ADC0 输入通道选择寄存器
REF0CN	0X9F	7	0X40	电压参考控制寄存器

20.4.1 控制寄存器 ADC0CN（SFR 地址： 0x95）

Bit	名称	属性	功能说明	复位值
7	AD0EN	R/W	ADC0 使能 1'b0: ADC0 不使能（低功耗关断） 1'b1: ADC0 使能（使能且准备开始数据转换）	1'h0
6	BURSTEN	R/W	ADC0 Burst Mode Enable 1'b0: ADC0 突发模式不使能 1'b1: ADC0 突发模式使能	1'h0
5	AD0INT	R/W	ADC0 转换结束中断标志 AD 转换完成（BURSTEN=0/1）后由硬件置 1。可触发中断，需软件清零	1'h0
4	AD0BUSY	R/W	ADC0 Busy 当 ADC0CM[1:0]=2'b00，将 ADC0BUSY 写 1，启动 AD 转换； 写 0，ADC0 不动作 读该 bit，返回 ADC 当前状态：	1'h0

			1'b1: ADC0 BUSY, 1'b0: ADC0 IDLE 注: 当 ADC0CM[1:0]=2'b00, ADC 关闭下, 启用 ADC 时 AD0EN 位和 AD0BUSY 位不能同时配置, 先配置使能再启动。	
3:2	-	R	Reserved	2'h0
1:0	AD0CM	R/W	ADC0 启动转换模式选择, 指定 ADC 启动转换源 2'b00: ADC0BUSY 写 1 2'b01: Timer1 溢出 2'b10: Timer2 溢出 2'b11: Timer3 溢出	2'h0

注: 先使能 ADC, 转换启动前需要写 5 次相关 SFR; 后使能 ADC, 转换启动前需要写 3 次相关 SFR。

ADC 转换进行时, 禁止关闭 ADC 使能, 需等待当前转换结束。

20.4.2 配置寄存器 ADC0CF (SFR 地址: 0x96)

Bit	名称	属性	功能说明	复位值
7:3	AD0SC	R/W	ADC0 SAR 转换时钟分频系数 BURSTEN=0: FCLK 是当前的系统时钟 BURSTEN=1: FCLK 是 12MHz 低功耗振荡时钟 (内部快时钟), 独立于系统时钟 $CLK_{SAR}=FCLK/AD0SC$ (AD0SC=1~31) $CLK_{SAR}=FCLK/32$ (AD0SC=0)	5'h1F
2	-	R/W	可读写, 但无实际意义	1'h0
1	AD0TM	R/W	ADC0 跟踪模式 写 1 或 0, 均为延迟跟踪模式: 当 ADC0 使能, 在转换启动信号有效 3 个 SAR 时钟才开始 AD 转换。在延迟的时间里, ADC 允许进行跟踪	1'h0
0	AMP0GN	R/W	ADC0 增益控制 1'b0: 片上 PGA 增益为 1 1'b1: 片上 PGA 增益为 0.5	1'h0

20.4.3 累加器配置寄存器 ADC0AC (SFR 地址: 0x97)

Bit	名称	属性	功能说明	复位值
7	-	R/W	可读写, 但无实际意义	1'h0
6	AD0AE	R/W	正常模式 ADC0 累加使能 当禁能突发模式, 即正常模式下, 使能此位以实现多次转换累加。 1'b0: ADC0H:ADC0L 为最新转换的结果 1'b1: ADC0H:ADC0L 为累加转换的结果; 软件需写 0x0000 到 ADC0H:ADC0L 以清除累加结果	1'b0

5:3	AD0SJST	R/W	ADC0 累加器移位和对齐方式 指示 ADC0H:ADC0L 读回的数据格式 3'b000: 右对齐; 没有移位 3'b001: 右对齐; 右移 1 位 3'b010: 右对齐; 右移 2 位 3'b011: 右对齐; 右移 3 位 3'b100: 左对齐; 没有移位 其他: 右对齐; 没有移位 注: 当重复次数大于 1, 必须右对齐。	3'h0
2:0	AD0RPT	R/W	突发模式 ADC0 累加次数 选择突发模式的转换和累加次数; 如果突发模式不使能, AD0RPT 必须配置为 3'b000/3'b110/3'b111 3'b000: 1 次 3'b001: 4 次 3'b010: 8 次 3'b011: 16 次 3'b100: 32 次 3'b101: 64 次 其他: 1 次	1'h0

20.4.4 突发模式上电时间寄存器 ADC0PWR (SFR 地址: 0xA5)

Bit	名称	属性	功能说明	复位值
7	AD0LPM	R/W	ADC0 低功耗模式使能 1'b0: 低功耗模式不使能 1'b1: 低功耗模式使能	1'h0
6:5	-	R	Reserved	2'h0
4:0	AD0PWR	R/W	ADC0 突发模式上电时间 设置 ADC0 从低功耗状态到上电的延时 1. 当 BURSTEN=0: ADC0 电源状态由 ADC0EN 控制 2. 当 BURSTEN=1&ADC0EN=1: ADC0 保持使能并且在转换完成后不进入低功耗模式; 在转换启动信号有效后, 启动立即开始 3. 当 BURSTEN=1&ADC0EN=0: ADC0 在转换完成后进入低功耗模式; 在转换启动信号有效后, 需经历 可编程的延时 后才启动转换; 延时的计算公式如下, 参考图 10 时序: $T_{startup} = (AD0PWR + 1) * 8 * 84ns$	5'h0F
注: 1. YS67FXXXX(X)的 burst_clk 频率为 12MHz, 周期约为 84ns				

20.4.5 突发模式跟踪时间寄存器 ADC0TK (SFR 地址: 0xA6)

Bit	名称	属性	功能说明	复位值
7:6	-	R	Reserved	2'h0
5:0	AD0TK	R/W	AD0 突发模式跟踪时间 设置 ADC0 突发模式下, 连续转换之间的延时, 参考图 10 ADC0 突发模式跟踪时间 T _{track} : T _{track} =(64-AD0TK)* 84ns	6'h1E
注:				
1. 在开始转换前额外插入 3 个 SAR 时钟周期				
2. T _{track} 没有插入到第一次转换, 对第一次转换来说, 跟踪时间需由突发模式上电时间 (T _{startup}) 满足				
3. YS67FXXXX(X)的 burst_clk 频率为 12MHz, 周期约为 84ns				

20.4.6 数据字高字节寄存器 ADC0H (SFR 地址: 0xAA)

Bit	名称	属性	功能说明	复位值
7:0	ADC0H	R/W	ADC0 数据字高字节 读: 16bit 的 ADC0 累加器数据格式由 AD0SJST[2:0]设置 写: 设置 16bit 的 ADC0 累加器的高字节	8'h00

20.4.7 数据字低字节寄存器 ADC0L (SFR 地址: 0xA9)

Bit	名称	属性	功能说明	复位值
7:0	ADC0L	R/W	ADC0 数据字低字节 读: 16bit 的 ADC0 累加器数据格式由 AD0SJST[2:0]设置 写: 设置 16bit 的 ADC0 累加器的低字节	8'h00

20.4.8 输入通道选择寄存器 ADC0MX (SFR 地址: 0xA7)

Bit	名称	属性	功能说明	复位值
7:4	-	R	Reserved	4'h0
3:0	AD0MX	R/W	AMUX0 (模拟源选择器选择端) 4'b0000: P3.4 ADC 输入通道 0 4'b0001: P3.5 ADC 输入通道 1 4'b0010: P2.0 ADC 输入通道 2 4'b0011: P2.1 ADC 输入通道 3 4'b0100: P2.2 ADC 输入通道 4 4'b0101: P2.3 ADC 输入通道 5 4'b0110: P2.4 ADC 输入通道 6 4'b0111: P2.5 ADC 输入通道 7 4'b1000: P2.6 ADC 输入通道 8 4'b1001: P2.7 ADC 输入通道 9	4'hF

			4'b101x: VDD 4'b1100: VDDL 4'b1101: Ground 注： 因 ADC 输入通道 0~9 与其他模拟功能复用 PAD，当其他模拟功能打开时，对应的 ADC 输入通道屏蔽。 复用关系如下： ✓ ADC 输入通道 0~1 复用外部慢时钟输入 ✓ ADC 输入通道 2~8 复用电容触摸通道 0~6 ✓ ADC 输入通道 9 复用电容触摸通道 7 或 VREF 外部参考电压	
--	--	--	--	--

20.4.9 电压参考控制寄存器 REF0CN (SFR 地址: 0x9F)

Bit	名称	属性	功能说明	复位值
7	-	R	Reserved	1'h0
6	DEM_EN	R/W	ADC 电容阵列温度译码使能控制 1'b0: 不使能 1'b1: 使能	1'h1
5	TESTIREF	R/W	选择内部电压参考输出到 PAD 用于测试 1'b0: 内部电压参考不输出到 PAD 1'b1: 内部电压参考输出到 PAD 注: 内部参考电压输出到 PAD 需要 TESTIREF=1 且 REFSL=2'b11。	1'h0
4	REFGND	R/W	模拟地参考 选择 ADC0 地参考 1'b0: ADC0 地参考是数字 GND 1'b1: ADC0 地参考是模拟 GND	1'h0
3:2	REFSL	R/W	电压参考选择 选择 ADC0 电压参考 2'b00: ADC0 电压参考是 P2.7/VREF pin 2'b01: ADC0 电压参考是 VDD pin 2'b10: ADC0 电压参考是内部 1.8V 数字供电电压 2'b11: ADC0 电压参考是内部电压参考 (2V/3V/4V) 注： 因 VREF pin 与 CCH7、AN9 复用 P2.7： ✓ REFSL=2'b00 时，当电容触摸通道 7 功能打开，将屏蔽 AN9，电压参考选择 VDD pin ✓ REFSL=2'b00 时，当电容触摸通道 7 功能不打开，将屏蔽 AN9，电压参考选择 VREF pin ✓ 需要使用 AN9 功能时，将 REFSL 不等于 2'b00。	2'h0
1:0	IREFSL	R/W	当 REFSL=2'b11 时，内部电压参考选择 (2V/3V/4V) 2'b00: 2V 2'b01: 3V 2'b1x: 4V	2'h0

21. 电气特性

偏置电压下的环境温度.....	-40°C 至+85°C
储存温度.....	-65°C 至+150°C
VDD 引脚相对于 VSS 的电压.....	-0.3V 至+6.5V
MCLR 引脚相对于 Vss 的电压.....	-0.3V 至+13.5V
所有其他引脚相对于 VSS 的电压.....	-0.3V 至(VDD+0.3V)

注意： 如果运行条件超过了上述“绝对极限参数值”，即可能对器件造成永久性损坏。上述值仅为运行条件的极大值，我们不建议器件运行在该规范范围以外。器件长时间工作在绝对极限参数条件下，其稳定性可能受到影响。

直流电器特性（VDD=1.8V-5.5V，GND=0V，TA=+25°C，除非另有说明）

直流特性		标准工作条件 工作温度 -40°C ≤ Ta ≤ +85°C				
符号	特性	最小值	典型值 ⁽¹⁾	最大值	单位	条件
VDD	电源电压	1.8		5.5	V	
VDR	RAM 数据保持电压 ⁽²⁾	—	0.8*	—	V	器件处于休眠模式
VPOR	Vdd 起始电压确保能够产生上电复位信号	—	Vss	—	V	
SVDD	Vdd 上升速率确保能够产生上电复位信号	0.05*	—	—	V/ms	
IDD 工作电流 ⁽³⁾		—	2.0	—	mA	电压 5V，系统内部时钟为 12MHZ，所有 IO 输出为低，外设默认
IPD	掉电流 ⁽⁴⁾	—	5	—	uA	WDT Disable VDD=5V
ΔIWDT	WDT 电流 ⁽⁴⁾	—	1	—	uA	VDD=5V
VIL	输入低电平电压	VSS	1.2	—	V	3V SCHMITT
		VSS	1.76	—		5V SCHMITT
VIH	输入高电平电压	—	2.1	VDD	V	3V SCHMITT
		—	2.76	VDD		5V SCHMITT
IOL	正常模式输出灌电流	—	9	—	mA	VOL=0.7V 3V
		—	16	—		VOL=0.7V 5V
	增强模式输出灌电流	—	28	—		VOL=0.7V 3V
		—	45	—		VOL=0.7V 5V
IOH	正常模式输出拉电流	—	—	—	mA	VOH=1.7V 3V
		—	—	—		VOH=3.6V 5V
		—	—	—		
		—	—	—		
		—	—	—		
		—	—	—		
	增强模式输出拉电流	—	—	—		VOH=2.1V 3V
		—	—	—		VOH=3.6V 5V
		—	—	—		
		—	—	—		
Vlvr	低电压复位电压	1.7 -20%	1.7	1.7 +20%	V	

		2.1 -20%	2.1	2.1 +20%		
		2.3 -20%	2.3	2.3 +20%		
		2.1 -20%	3.8	3.8 +20%		
Rpu	上拉电阻	—	30	—	K	3V
		—	16	—		5V

注: “—”表示没有, 或待定。

(1) 典型栏中数据均为 25℃ 条件下值, 此部分数据仅供参考。

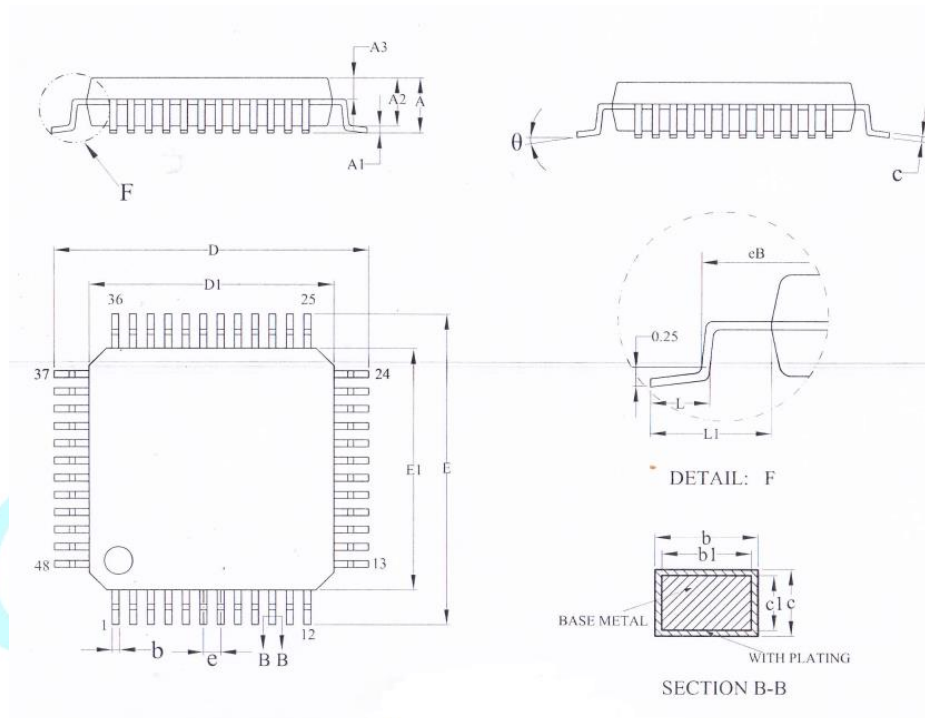
(2) 该电压是保证不丢失 RAM 数据的最小 VDD。

(3) 工作电流主要随工作电压和频率而变化。其它因素, 如总线负载、总线速率、内部代码执行模式和温度也会影响电流消耗。

(4) 掉电电流是在器件休眠时, 所有 I/O 引脚都处于高阻态并且连接到 Vdd 或 Vss 时测得。

22. 封装信息

22.1. LQFP48



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	—	—	1.60
A1	0.05	—	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	—	0.26
b1	0.17	0.20	0.23
c	0.13	—	0.17
c1	0.12	0.13	0.14
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
eB	8.10	—	8.25
e	0.50BSC		
L	0.45	—	0.75
L1	1.00REF		
θ	0	—	7°

23. 汇春知识产权政策

23.1. 专利权

汇春公司在全球各地区已核准和申请中之专利权至少有 160 件以上，享有绝对之合法权益。与汇春公司 MCU 或其它产品有关的专利权并未被同意授权使用，任何经由不当手段侵害汇春公司专利权之公司、组织或个人，汇春将采取一切可能的法律行动，遏止侵权者不当的侵权行为，并追讨汇春公司因侵权行为所受之损失、或侵权者所得之不法利益。

23.2. 著作权

Copyright 2021 by INC.

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

<http://www.yspringtech.com>

24. 版本编号

版本	日期	更新内容	作者
V1.0	2022-04-18	原始版本	Conrad
V1.1	2022-04-27	修订封装、命名信息	Conrad
V1.2	2022-04-27	删减部分内容	Conrad
V1.3	2022-04-28	修改错误	Conrad