Panchip Microelectronics Co., Ltd.

PAN262x

用户手册

2.4GHz 高速无线收发 SOC 芯片

当前版本： 1.6

发布日期： 2023.11

上海磐启微电子有限公司

地址：上海张江高科技园区盛夏路 666 号 D 栋 302 室

联系电话：021-50802371

网址：http://www.panchip.com

文档说明

由于版本升级或存在其他原因，本文档内容会不定期进行更新。除非另有约定，本文档内容仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

商标

磐启是磐启微电子有限公司的商标。本文档中提及的其他名称是其各自所有者的商标/注册商标。

免责声明

本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，磐启微电子有限公司对本文档内容不做任何明示或暗示的声明或保证。

修订历史

版本	修订时间	更新内容
V1.0	2023.01	第一版
V1.1	2023.05	更新 ADC 章节，更新 16.1.3 中 RF 收发机的工作模式（增加图），更新 12.3.1，统一 USB2.0 全速模式描述，更新图 1.3，增加 15.4 章节，简化产品简介，SYS_CLK_DIV 更正为默认不分频，删除 RCH 校正一节。
V1.2	2023.06	更正 15.6.5 和 15.6.8 寄存器描述，增加 MSOP10 封装。
V1.3	2023.06	删除 LVRL。
V1.4	2023.06	更正引脚说明中电压笔误。
V1.5	2023.09	更正一些 RCL 时钟一致性的错误描述。
V1.6	2023.11	工作电压更新为 1.9V~3.6V。

此版本为内部版本，仅供参考。

目录

第1章 产品简介

1.1 概述

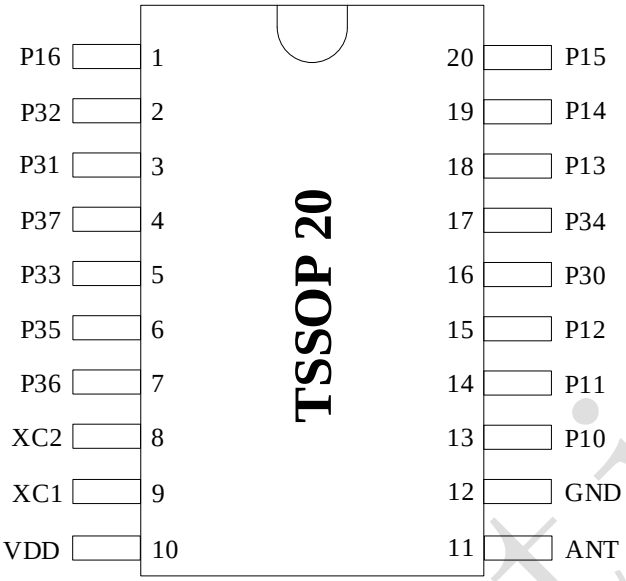
PAN262x 是一款集成高性能 8051 内核和 2.4G 高速无线收发器 SOC 芯片。具有低成本，低功耗的特点，为超低功耗无线应用提供单芯片解决方案。

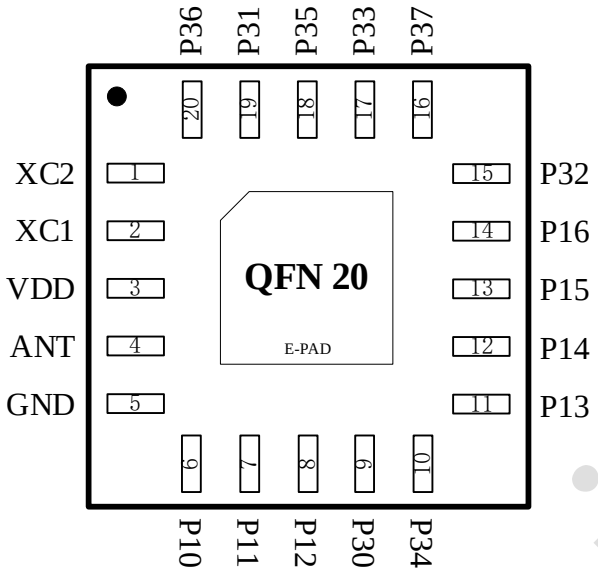
该收发器适用于 2.400~2.483GHz 全球 ISM 频段。它集成了射频（RF）发射器和接收器，频率合成器，晶体振荡器，基带 GFSK 调制解调器等。PAN262x 支持普通型模式、增强型模式、一对多网络通讯模式以及广播蓝牙帧格式。发射功率、通讯频率和数据速率都可配置。

PAN262x 最高运行频率可达 16 MHz，支持 1.9V ~ 3.6V 的宽工作电压范围，工作温度-40℃ ~ 85℃。其嵌入式程序 Flash 大小高达 32KB，256B IRAM 和 2KB XRAM。

PAN262x 具有许多高性能外设功能，例如最多可支持 15 个 GPIO 引脚，3 个定时器，1 个 UART，1 个 SPI 主从接口，1 个 I2C 主机接口，6 路 PWM 发生器，1 个 8 通道 12 位 ADC，1 个 USB，1 个看门狗定时器，1 个上电复位和 1 个掉电复位、1 个低压复位。为了减少元件数量、节约电路板空间和系统成本，PAN262x 将天线的匹配电路，LDO 滤波电容和晶体的外部电容都集成到了芯片里面。

1.2 引脚说明





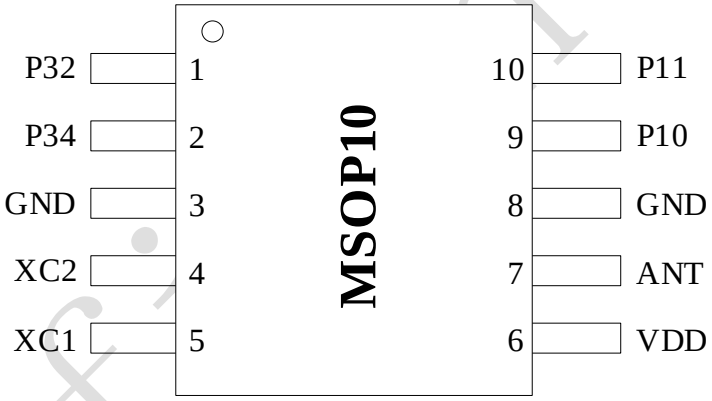


图 1.2 PAN262x 引脚说明

封装			引脚名称	引脚类型	描述
TSSOP	QFN	MSOP			
1	14	-	P16	IO	通用输入输出 IO
2	15	1	C2CK	I	C2 时钟
			P32	IO	通用输入输出 IO
			INT0	I	IO 外部中断 0 输入
			PWM5	O	PWM5 输出
			ADC_CH2	AIO	ADC 通道 2
3	19	-	P31	IO	通用输入输出 IO
			SSN/SPSSN[0]	IO	SPI 从机模式片选/SPI 主机模式，从机选择 0
			TXD0	O	UART 输出

			PWM4	O	PWM4 输出
			ADC_TRG	I	ADC 触发
			SDA	IO	I2C 数据
			ADC_CH1	AIO	ADC 通道 1
4	16	-	P37	IO	通用输入输出 IO
			CC3	I	Timer2 比较器/捕获器 3 输入
			EN_PA	O	外挂 PA 控制
			SDA	IO	I2C 数据
			ADC_TRG	I	ADC 触发
			TXD0	O	UART 输出
			ADC_CH7	AIO	ADC 通道 7
5	17	-	P33	IO	通用输入输出 IO
			INT1	I	IO 外部中断 1 输入
			MISO	IO	SPI 主机输入从机输出
			ADC_TRG	I	ADC 触发
			ADC_CH3	AIO	ADC 通道 3
6	18	-	P35	IO	通用输入输出 IO
			MOSI	IO	SPI 主机输出从机输入
			T1	I	Timer1 计数器模式触发
			SPSSN[4]	O	SPI 主机模式从机选择 4
			ADC_CH5	AIO	ADC 通道 5
7	20	-	PAD_NRST	I	PAD 复位，低有效
			P36	IO	通用输入输出 IO
			CC2	I	Timer2 比较器/捕获器 2 输入
			SPSSN[5]	O	SPI 主机模式从机选择 5
			SCL	IO	I2C 时钟
			ADC_TRG	I	ADC 触发
			ADC_CH6	AIO	ADC 通道 6
			RXD0	I	UART 接收
8	1	4	XC2	AO	晶振引脚 2
9	2	5	XC1	AI	晶振引脚 1
10	3	6	VDD	P	1.9V-3.6V 供电电源 外接 1uF 电容
11	4	7	ANT	AIO	天线
12	5	8	GND	P	地
13	6	9	P10	IO	通用输入输出 IO
			PWM0	O	PWM0 输出

			T2	I	Timer2 计数器模式触发
			CC0	I	Timer2 比较器/捕获器 0 输入
			RXD0	I	UART 接收
			USB_DP	AIO	USB DP
14	7	10	P11	IO	通用输入输出 IO
			PWM1	O	PWM1 输出
			T2EX	I	Timer2 重载模式触发
			SPSSN[2]	O	SPI 主机模式从机选择 2
			TXD0	O	UART 输出
			USB_DM	AIO	USB DM
15	8	-	P12	IO	通用输入输出 IO
			SCK	IO	SPI 时钟
			CC1	I	Timer2 比较器/捕获器 1 输入
			PWM2	O	PWM2 输出
16	9	-	P30	IO	通用输入输出 IO
			RXD0	I	UART 接收
			SPSSN[1]	O	SPI 主机模式从机选择 1
			PWM3	O	PWM3 输出
			SCL	IO	I2C 时钟
			ADC_CH0	AIO	ADC 通道 0
17	10	2	C2DAT	IO	C2 数据
			P34	IO	通用输入输出 IO
			T0	I	Timer0 计数器模式触发
			SPSSN[3]	O	SPI 主机模式从机选择 3
			ADC_CH4	AIO	ADC 通道 4
18	11	-	P13	IO	通用输入输出 IO
19	12	-	P14	IO	通用输入输出 IO
20	13	-	P15	IO	通用输入输出 IO
-	-	3	GND	P	地

表 1.1 PAN262x PAD 功能

1.3 地址映射

1.3.1 存储结构说明

存储结构和标准的 8051 类似，分为三部分：程序存储区（CODE），外部数据存储区（XDATA），内部数据存储区（DATA，也分为 DATA 和 IDATA）。

CODE：0x0000~0x7FFF，共 32K Bytes FLASH，通过 MOVC 访问；

XDATA：0x0000~0x07FF，共 2K Bytes SRAM，通过 MOVX 访问；

DATA：0x00~0x7F 区域，通过 MOV 直接寻址。其中 0x00~0x1F 为 4 组通用寄存器 R0~R7，由 PSW.4 和 PSW.3 来选择。0x20~0x2F 可以位寻址，其他字节寻址。

IDATA：包括 DATA 的 0x00~0xFF，其中低 128 Bytes 同 DATA，高 128 Bytes 作为 RAM 时 MOV 间接寻址，作为 SFR 时直接寻址。

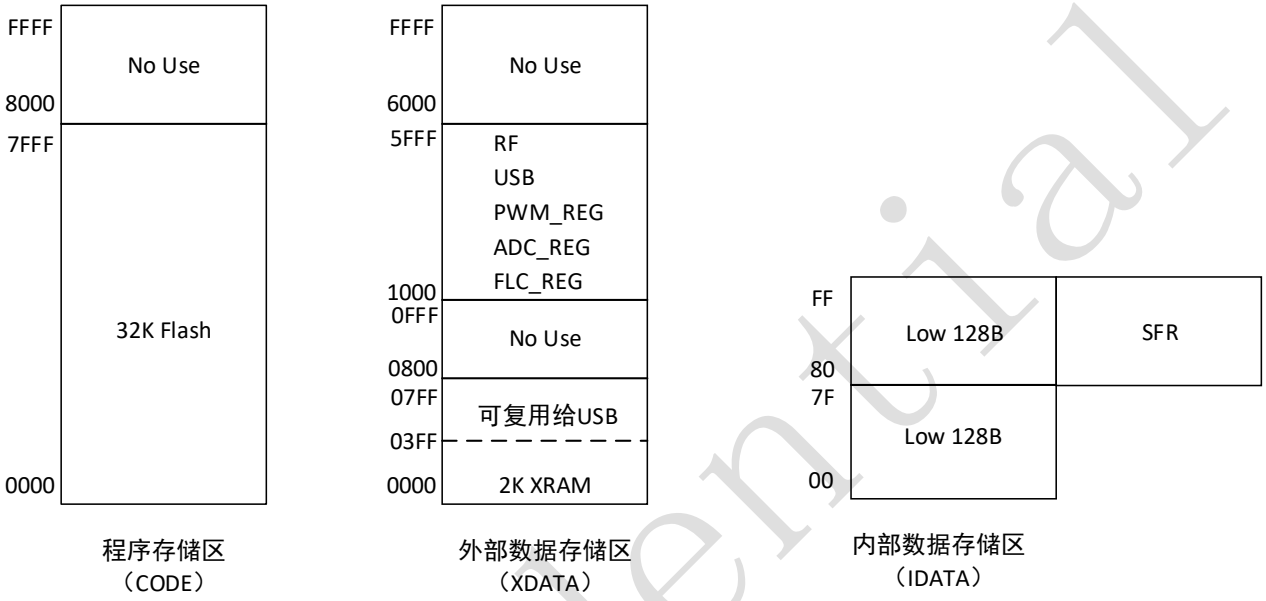


图 1.3　PAN262x 地址映射

1.3.2 特殊功能寄存器（SFR）

下图是特殊功能寄存器映射图。

offset +7	offset +6	offset +5	offset +4	offset +3	offset +2	offset +1	offset + 0	Base Addr
		rcc_dat	rcc_sel	ana_dat	ana_sel	sys_dat	sys_sel	**0xF8**
srst							b	**0xF0**
		flc_trg	flc_cmd	flc_dat	flc_adrh	flc_adrl	flc_ctrl	**0xE8**
			spssn	spdat	spcon	spsta	acc	**0xE0**
		i2csta	i2ccon	i2cadr	i2cdat		adcon	**0xD8**
							psw	**0xD0**
		th2	tl2	crch	crcl		t2con	**0xC8**
cch3	ccl3	cch2	ccl2	cch1	ccl1	ccen	ircon	**0xC0**
					srelh	ip1	ien1	**0xB8**
							p3	**0xB0**
					srell	ip0	ien0	**0xA8**
							p2	**0xA0**
						sbuf	scon	**0x98**

				dpc	dps		p1	**0x90**
	ckcon	th1	th0	tl1	tl0	tmod	tcon	**0x88**
pcon	wdtrel	dph1	dpl1	dph	dpl	sp	p0	**0x80**

说明：各寄存器具体说明，参考各功能模块。

1.3.3 位寻址寄存器

SFR 的 16 个地址既可以字节寻址又可以位寻址。这些地址为后 3 个 bit 为 0 的地址（0x80,0x88…0xF8）。 DATA 空间的 0x20~0x2F 共 128 个 BIT 同样可以位寻址。具体映射如下表：

offset +7	offset +6	offset +5	offset +4	offset +3	offset +2	offset +1	offset + 0	Base Addr
								0xF8
b.7	b.6	b.5	b.4	b.3	b.2	b.1	b.0	**0xF0**
								0xE8
acc.7	acc.6	acc.5	acc.4	acc.3	acc.2	acc.1	acc.0	**0xE0**
adcon.7	adcon.6	adcon.5	adcon.4	adcon.3	adcon.2	adcon.1	adcon.0	**0xD8**
psw.7	psw.6	psw.5	psw.4	psw.3	psw.2	psw.1	psw.0	**0xD0**
t2con.7	t2con.6	t2con.5	t2con.4	t2con.3	t2con.2	t2con.1	t2con.0	**0xC8**
ircon.7	ircon.6	ircon.5	ircon.4	ircon.3	ircon.2	ircon.1	ircon.0	**0xC0**
ien1.7	ien1.6	ien1.5	ien1.4	ien1.3	ien1.2	ien1.1	ien1.0	**0xB8**
p3.7	p3.6	p3.5	p3.4	p3.3	p3.2	p3.1	p3.0	**0xB0**
ien0.7	ien0.6	ien0.5	ien0.4	ien0.3	ien0.2	ien0.1	ien0.0	**0xA8**
p2.7	p2.6	p2.5	p2.4	p2.3	p2.2	p2.1	p2.0	**0xA0**
scon.7	scon.6	scon.5	scon.4	scon.3	scon.2	scon.1	scon.0	**0x98**
p1.7	p1.6	p1.5	p1.4	p1.3	p1.2	p1.1	p1.0	**0x90**
tcon.7	tcon.6	tcon.5	tcon.4	tcon.3	tcon.2	tcon.1	tcon.0	**0x88**
p0.7	p0.6	p0.5	p0.4	p0.3	p0.2	p0.1	p0.0	**0x80**
0x2F.7	0x2F.6	0x2F.5	0x2F.4	0x2F.3	0x2F.2	0x2F.1	0x2F.0	**0x78**
							0x2E	**0x70**
							0x2D	**0x68**
							0x2C	**0x60**
							0x2B	**0x58**
							0x2A	**0x50**
							0x29	**0x48**
							0x28	**0x40**
							0x27	**0x38**
							0x26	**0x30**
							0x25	**0x28**
							0x24	**0x20**
							0x23	**0x18**
							0x22	**0x10**
0x21.7	0x21.6	0x21.5	0x21.4	0x21.3	0x21.2	0x21.1	0x21.0	**0x08**
0x20.7	0x20.6	0x20.5	0x20.4	0x20.3	0x20.2	0x20.1	0x20.0	**0x00**

1.3.4 XDATA

XDATA 的 0x000~0x7FF 为 2K Bytes 的外部数据存储，其中高 1K 的空间和 USB 的 FIFO 共用，当使用 USB 时该空间不能使用。

XDATA 的 0x1000~0x5FFF 空间，分配给了外设 Flash Controller、ADC、PWM、USB 和 RF 收发机使用，具体映射表如下：

地址范围	外设
0x5000-0x5FFF	RF 收发机
0x4000-0x4FFF	USB
0x3000-0x3FFF	PWM
0x2000-0x2FFF	ADC
0x1000-0x1FFF	Flash Controller

第2章 CPU

2.1 CPU 特性

PAN262x 的 51 内核使用 ASM51 指令集，有两条片内总线：Memory 总线和 SFR 总线。Memory 总线用于在片内扩展程序和数据存储器，如扩展片内 ROM、Flash、XRAM 等。SFR(SpecialFunction Register) 总线用于和片内的外设寄存器接口，除了工作寄存器 R0～R7、程序计数器（PC）和指令寄存器（IR）外，所有控制、配置和状态寄存器都映射到 SFR 空间，8051 可通过直接寻址的方式访问这些寄存器，控制系统工作。PAN262x 支持两线烧录、在线调试功能。

2.2 指令集

指令集和标准工业 8051 指令集兼容，这种兼容性表现在指令的操作码，功能以及指令运行对相关标志位的影响相同。下列表格列出了具体的指令集，包括指令的助记符、指令功能、指令的字节数以及相应的机器周期等。

指令集列表符号说明	
符号	功能
A	累加器
(A)	累加器内容
Rn	工作寄存器 R0-R7
(Rn)	工作寄存器的内容
Ri	i=0， 1，数据指针 R0 或 R1
(Ri)	R0 或 R1 的内容
((Ri))	R0 或 R1 指出的单元内容
@Ri	R0 或者 R1 指针指向的内部寄存器（除了 MOVX 指令）
(X)	某一寄存器的内容
X	某一寄存器
((X))	某一寄存器指出的单元内容
direct	直接地址单元
(direct)	直接地址指出的单元内容
rel	相对偏移量，带符号的（2 的补码） 8 位偏移字节
bit	位地址
#data	8 位立即数
#data 16	16 位立即数
addr 16	16 位绝对地址
addr 11	页面地址
←	数据传送方向
∧	逻辑与
∨	逻辑或
⊕	逻辑异或

2.2.1 算术运算指令

助记符	功能描述	机器码	字节数	机器周期
ADD A,Rn	A (A)+(Rn)	0x28-0x2F	1	1
ADD A,direct	A (A)+(direct)	0x25	2	2
ADD A,@Ri	A (A)+((Ri))	0x26-0x27	1	2
ADD A,#data	A (A)+data	0x24	2	2
ADDC A,Rn	A (A)+(Rn)+(CY)	0x38-0x3F	1	1
ADDC A,direct	A (A)+(direct)+ (CY)	0x35	2	2
ADDC A,@Ri	A (A)+((Ri)) + (CY)	0x36-0x37	1	2
ADDC A,#data	A (A)+data+ (CY)	0x34	2	2
SUBB A,Rn	A (A)-(Rn)-(CY)	0x98-0x9F	1	1
SUBB A,direct	A (A)-(direct)- (CY)	0x95	2	2
SUBB A,@Ri	A (A)-((Ri)) - (CY)	0x96-0x97	1	2
SUBB A,#data	A (A)-data- (CY)	0x94	2	2
INC A	A (A)+1	0x04	1	1
INC Rn	Rn Rn+1	0x08-0x0F	1	1
INC direct	direct direct+1	0x05	2	2
INC @Ri	(Ri) ((Ri)) + 1	0x06-0x07	1	2
INC DPTR	DPTR (DPTR)+1	0xA3	1	1
DEC A	A (A)-1	0x14	1	1
DEC Rn	A (Rn)-1	0x18-0x1F	1	1
DEC direct	A (direct)-1	0x15	2	2
DEC @Ri	A ((Ri)) - 1	0x16-0x17	1	2
MUL AB	AB (A).(B)	0xA4	1	4
DIV	AB (A)/(B)	0x84	1	4
DA A	对 A 进行十进制调整	0xD4	1	1

2.2.2 逻辑运算指令

助记符	功能描述	机器码	字节数	机器周期
ANL A,Rn	A (A)∧(Rn)	0x58-0x5F	1	1
ANL A,direct	A (A)∧(direct)	0x55	2	2
ANL A,@Ri	A (A)∧((Rn))	0x56-0x57	1	2
ANL A,#data	A (A)∧data	0x54	2	2
ANL direct,A	direct (direct) ∧A	0x52	2	2
ANL direct,#data	direct (direct) ∧data	0x53	3	3
ORL A,Rn	A (A) ∨(Rn)	0x48-0x4F	1	1
ORL A,direct	A (A) ∨(direct)	0x45	2	2
ORL A,@Ri	A (A) ∨((Ri))	0x46-0x47	1	2
ORL A,#data	A (A) ∨data	0x44	2	2
ORL direct,A	direct (direct) ∨A	0x42	2	2
ORL direct,#data	direct (direct) ∨data	0x43	3	3
XRL A,Rn	A (A) ⊕ (Rn)	0x68-0x6F	1	1
XRL A,direct	A (A) ⊕ (direct)	0x65	2	2
XRL A,@Ri	A (A) ⊕ ((Ri))	0x66-0x67	1	2
XRL A,#data	A (A) ⊕data	0x64	2	2
XRL direct,A	direct (direct) ⊕ A	0x62	2	2
XRL direct,#data	direct (direct) ⊕ data	0x63	3	3
CLR A	A 0	0xE4	1	1
CPL A	A	0xF4	1	1
RL A	A 循环左移一位	0x23	1	1
RLC A	A 带进位循环左移一位	0x33	1	1
RR A	A 循环右移一位	0x03	1	1
RRC A	A 带进位循环右移一位	0x13	1	1
SWAP A	A 半字节交换	0xC4	1	1

2.2.3 数据传输指令

助记符	功能描述	机器码	字节数	机器周期
MOV A,Rn	A (Rn)	0xE8-0xEF	1	1
MOV A,direct	A （direct）	0xE5	2	2
MOV A,@Ri	A ((Ri))	0xE6-0xE7	1	2
MOV A,#data	A data	0x74	2	2
MOV Rn,A	Rn (A)	0xF8-0xFF	1	1
MOV Rn,direct	Rn (direct)	0xA8-0xAF	2	2
MOV Rn,#data	Rn data	0x78-0x7F	2	2
MOV direct,A	direct (A)	0xF5	2	2
MOV direct,Rn	direct (Rn)	0x88-0x8F	2	2
MOV direct1,direct2	direct (direct)	0x85	3	3
MOV direct,@Ri	direct ((Ri))	0x86-0x87	2	2
MOV direct,#data	direct data	0x75	3	3
MOV @Ri,A	((Ri)) (A)	0xF6-0xF7	1	1
MOV @Ri,direct	(Ri) (direct)	0xA6-0xA7	2	2
MOV @Ri,#data	(Ri) data	0x76-0x77	2	2
MOV DPTR,#data16	DPTR data	0x90	3	3
MOVC A,@A+DPTR	A ((A)+(DPTR))	0x93	1	3
MOVC A,@A+PC	A ((A)+(PC))	0x83	1	3
MOVX A,@Ri	A ((Ri+P2))	0xE2-0xE3	1	3~10
MOVX A,@DPTR	A ((DPTR))	0xE0	1	3~10
MOVX @Ri,A	((Ri)+P2) A	0xF2-0xF3	1	3~12
MOVX @DPTR,A	A ((DPTR))	0xF0	1	3~12
PUSH direct	SP SP+1， (SP) (direct)	0xC0	2	2
POP direct	direct ((SP))， SP (SP)-1	0xD0	2	2
XCH A,Rn	(A)←→(Rn)	0xC8-0xCF	1	1
XCH A,direct	(A)←→(direct)	0xC5	2	2
XCH A,@Ri	(A)←→((Ri))	0xC6-0xC7	1	2
XCHD A,@Ri	$(A)_{0\sim3}$←→$((Ri))_{0\sim3}$	0xD6-0xD7	1	2

2.2.4 跳转指令

助记符	功能描述	机器码	字节数	机器周期
ACALL addr11	PC (PC)+2， SP (SP)+1， (SP) (PC)L， SP (SP)+1， (SP) (PC)H， PC10~0 addr11	xxx10001b	2	2
LCALL addr16	PC (PC)+2， SP (SP)+1， (SP) (PC)L， SP (SP)+1， (SP) (PC)H， PC10~0 addr16	0x12	3	3
RET	(PC)H ((SP))， SP (SP)-1， (PC)L ((SP))， SP (SP)-1	0x22	1	4
RETI	(PC)H ((SP))， SP (SP)-1， (PC)L ((SP))， SP (SP)-1， 从中断中返回	0x32	1	4
AJMP addr11	PC10~0 addr11	xxx00001b	2	2
LJMP addr16	PC addr16	0x02	3	3
SJMP rel	PC PC+rel	0x80	2	2
JMP @A+DPTR	PC PC+2，若 CY=1，则 PC PC+rel	0x73	1	3
JZ rel	PC PC+2，若 CY=0，则 PC PC+rel	0x60	2	3
JNZ rel	PC PC+3，若(bit)=1，则 PC PC+rel	0x70	2	3
JC rel	PC PC+3，若(bit)=0，则 PC PC+rel	0x40	2	3
JNC	PC PC+3，若(bit)=1，则 bit 0，PC (PC)+rel	0x50	2	3
JB bit,rel	PC (A)+(DPTR)	0x20	3	4
JNB bit,rel	PC PC+2，若(A)=0，PC (PC)+rel	0x30	3	4
JBC bit,rel	PC PC+2，若(A)不等于 0，PC (PC)+rel	0x10	3	4
CJNE A,direct,rel	PC PC+3，若(A)不等于（direct），则 PC (PC)+rel	0xB5	3	4
CJNE A,#data,rel	PC PC+3，若(A)不等于 data，则 PC (PC)+rel	0xB4	3	4
CJNE Rn,#data,rel	PC PC+3，若(Rn)不等于 data，则 PC (PC)+rel	0xB8-0xBF	3	4
CJNE @Ri,#data,rel	PC PC+3，若((Ri))不等于 d，则 PC (PC)+rel	B6-B7	3	5
DJNZ Rn,rel	PC PC+2， Rn =(Rn)-1， 若(Rn)不等于 0， 则 PC (PC)+rel	D8-DF	2	3
DJNZ direct,rel	PC PC+2， direct =(direct)-1，若(direct)不等于 0， 则 PC (PC)+rel	D5	3	4
NOP	No operation	0	1	1

2.2.5 布尔运算指令

助记符	功能描述	机器码	字节数	机器周期
CLR C	C_Y 0	0xC3	1	1
CLR bit	bit 0	0xC2	2	2
SETB C	C_Y 1	0xD3	1	1
SETB bit	bit 1	0xD2	2	2
CPL C	C_Y	0xB3	1	2
CPL bit	bit	0xB2	2	2
ANL C,bit	C_Y (C_Y)∧(bit)	0x82	2	2
ANL C,/bit	C_Y (C_Y)∧	0xB0	2	2
ORL C,bit	C_Y (C_Y)∨(bit)	0x72	2	2
ORL C,/bit	C_Y (C_Y)∨	0xA0	2	2
MOV C,bit	C_Y (bit)	0xA2	2	2
MOV bit,C	bit (C_Y)	0x92	2	2

2.3 CPU 相关寄存器

2.3.1 ACC

累加器是一个最常用的专用寄存器。大部分单操作数指令的操作取自累加器。很多双操作数指令的一个操作数取自累加器。加、减、乘、除算术运算指令的运算结果都存放在累加器 A 或 AB 寄存器中。指令系统中用 A 作为累加器的助记符。

2.3.2 B

在乘除指令中用到 B 寄存器。乘法指令的两个操作数分别取自 A 和 B，其结果存放在 AB 寄存器中。除法指令中，被除数取自 A，除数取自 B，商数存放于 A，余数存放于 B。在其他指令中，B 寄存器可作为 RAM 中的一个单元来使用。

2.3.3 PSW

程序状态字 PSW 是一个 8 位寄存器，它包含了程序状态信息。此寄存器的含义参见下表：

Register	Address	R/W	Description	Reset Value
PSW	0xD0	R/W	程序状态寄存器	0x00

Bits	Description	
7	CY	进位标志。在执行某些算术和逻辑指令时，可以被硬件或软件复位或清零。在布尔处理机中它被认为是位累加器，其重要性相当于一般中央处理机中的累加器 A。
6	AC	辅助进位标志。在进行加法或减法操作而产生低 4 位数（十进制的一个数字）向高 4 位数进位或借位时，AC 将被置位，否则被清零。AC 被用于 DAA 指令的十进制调整。
5	F0	标志 0。是用户定义的一个状态标记，可用软件置位或清零。
4	RS1	工作寄存器区选择控制位 1，和 RS0 一起用以选择工作寄存器区。
3	RS0	工作寄存器区选择控制位 0 RS1 | RS0 | Bank 选择 0 | 0 | 区 0（00H-07H） 0 | 1 | 区 1 （08H-0FH） 1 | 0 | 区 2 （10H-17H） 1 | 1 | 区 3 （18H-1FH）
2	OV	溢出标志。当加法产生进位，减法产生借位，乘除产生溢出时置 1，否则为 0。
1	F1	标志 1，是用户定义的一个状态标记，可用软件置位或清零。
0	P	奇偶校验。每个指令周期都由硬件来置位或清零，以表示累加器 A 中 1 的位数的奇偶数。若 1 的位数为奇数，则 P 置位，否则清 0。

2.3.4 SP

栈指针 SP 是一个 8 位专用寄存器。它指示出堆栈顶部在内部数据存储中的位置。系统复位后，SP 初始化为 07H，使得堆栈事实上由 08H 单元开始。考虑到 08H～1FH 单元分属于工作寄存器 1～3，若程序设计要用到这些区，则把 SP 的值改置更大的值。SP 的初值越小，堆栈深度就越深。堆栈指针的值可由软件改变，因此堆栈在内部数据存储中的位置比较灵活。

除用软件改变 SP 值外，在执行 PUSH/POP、各种子程序调用、中断响应、子程序返回（RET）和中断返回（RETI）等指令时，SP 值将自动增加或减少。

2.3.5 DPTR

为加速数据的块搬移操作，PAN262x 的 CPU 使用双数据指针。标准 8051 的数据指针是一个 16 位专用寄存器，其高位字节寄存器用 DPH 表示，低位字节用 DPL 表示，DPTR 主要用来存放 16 位地址，当对外部数据存储器空间寻址时，可作为间接寄存器用。

PAN262x 的 CPU 包括一个同标准 8051 相同的数据指针 DPTR0，它位于 SFR 82H（DPL0）和 83H（DPH0），默认情况下数据指针使用 DPTR0。除此之外 PAN262x 增加了第二个数据指针 DPTR1， DPTR1 位于 SFR 84H(DPL1)和 85H(DPH1)。DPS 寄存器(SFR 86H)的 SEL 位用来选择当前数据指针使用 DPTR0 还是 DPTR1，当 SEL=0 使用 DPTR 的指令的数据指针用 DPL0 和 DPH0 作为数据指针；当 SEL=1，使用 DPTR 的指令的数据指针用 DPL1 和 DPH1。SEL 是 DPS 的第 0 位，DPS 的其他位无用。

所有和 DPTR 相关的指令使用 DPS 选择的数据指针。SEL 取反将导致数据指针切换，切换最快的方法是使用 INC DPS 指令，仅需要一条指令就可使数据指针由源地址指向目的地址，当进行块数据搬移时，这样做节省了保存源地址和目的地址的代码和时间。当搬移大批量数据时，使用双数据指针的机制显著地提高了代码的效率。

2.3.6 DPS

PAN262x 的 CPU 使用双数据指针，DPS 的 BIT0 用于指示当前工作的 DPTR，0 选择 DPTR0，1 选择 DPTR1，DPS 的其他 BIT 为保留位。

2.3.7 DPC

PAN262x 的 CPU 包含一个 DPTR 相关的算术单元。该算术单元可以让当前工作的 DPTR 在两个 DPTR 之间自动切换。这个功能有 DPC 寄存器控制。每个 DPTR 都有独立的 DPC 寄存器，以提供数据传输的高度灵活性。0x93 指向的 DPC 寄存器，实际是 DPC 寄存器的入口，由 DPS 来选择相应的 DPTR，和 DPTR 的选择一致。

Register	Address	R/W	Description	Reset Value
DPC	0x93	R/W	DP 控制寄存器	0x00/0x08

Bits	Description	
7:4	Reserved	Reserved
3	Next Data Pointer Selection	在每个 MOVX @DPTR 指令之后，这个字段的内容被加载到 DPS 寄存器。因为这个特性总是启用的，因此，对于每个 DPC 寄存器的这个字段必须包含指向自身的值，以便自动切换不会发生在默认值情况下。
2	Auto-modification size	当为 0 时，且 DPCbit0=1 时，在 MOVX @DPTR 指令之后，当前的 DPTR 自动增加或者减少 1。 当为 1 时，且 DPCbit0=1 时，在 MOVX @DPTR 指令之后，当前的 DPTR 自动增加或者减少 2。
1	Auto-modification direction	当为 0 时，且 DPCbit0=1 时，在 MOVX @DPTR 指令之后，当前的 DPTR 自动增加。 当为 1 时，且 DPCbit0=1 时，在 MOVX @DPTR 指令之后，当前的 DPTR 自动减少。
0	Auto-modification enable	当为 1 时，使能自动修改的功能。

2.3.8 PCON

PAN262x CPU 内部的 PCON 寄存器，用于控制进入低功耗模式，具体功能参考下表。

Register	Address	R/W	Description	Reset Value
PCON	0x87	R/W	电源控制寄存器	0x08

Bits	Description	
7	SMOD	用于串口波特率选择，具体用法请参考串口部分
6:5	Reserved	
4	PMW	0：MOVX 正常读写 XDATA 1：MOVX 读写 CODE 区域
3：2	Reserved	
1	STOP	写 1，系统进入深度睡眠状态，该 BIT 读出为 0
0	IDLE	写 1，系统进入睡眠状态，CPU 时钟关闭，该 BIT 读出为 0

第3章 中断系统

3.1 概述

PAN262x 中断系统共支持 18 个中断源，其中 4 个通用中断：Timer0、Timer1、Timer2 以及 UART。

其他 14 个中断源分别为:外部中断 INT0 和 INT1、低功耗唤醒中断 WK_CNT 和 GPIO_WK、I2C、SPI、ADC、PWM、USB、RF 以及 TIMER2 比较器 0/1/2/3 中断。其中部分中断源共用一个中断入口，由相应的寄存器控制。

3.2 中断列表

中断号为 C 程序中断服务程序中对应的数字，示例：void I2C_INT(void) interrupt 8{;}

WAKEUP_INT_CTRL 寄存器参考 5.2.2.6 章节，CCEN、TCON、T2CON 寄存器参考 TIMER 章节，S0CON 寄存器参考 UART 章节，其他寄存器在本章节后续有描述。

中断源	入口地址	中断号	中断使能(IEN0.7)	中断标志位	中断源选择	中断触发模式选择
RF/COM3	0x6B	13	IEN1.5	IRCON.5(COM3)	CCEN.7 & CCEN.6	上升沿
USB/COM2	0x63	12	IEN1.4	IRCON.4(COM2)	CCEN.5 & CCEN.4	上升沿
PWM/COM1	0x5B	11	IEN1.3	IRCON.3(COM1)	CCEN.3 & CCEN.2	上升沿
ADC/COM0	0x53	10	IEN1.2	IRCON.2(COM0)	CCEN.1 & CCEN.0	T2CON.6
SPI	0x4B	9	IEN1.1	IRCON.1	无	无
I2C	0x43	8	IEN1.0	IRCON.0	无	无
Timer2	0x2B	5	IEN0.5	IRCON.6/7	无	无
Uart	0x23	4	IEN0.4	S0CON.0/1	无	无
Timer1	0x1B	3	IEN0.3	TCON.7	无	无
INT1/GPIO_WK	0x13	2	IEN0.2	TCON.3(INT1)	WAKEUP_INT_CTRL.1	TCON.2
Timer0	0x0B	1	IEN0.1	TCON.5	无	无
INT0/WK_CNT	0x03	0	IEN0.0	TCON.1(INT0)	WAKEUP_INT_CTRL.0	TCON.0

关于中断的一些说明：

1. 中断源选择为 INT0 和 INT1 时，可以通过配置 TCON.0 和 TCON.2 来选择低电平触发或者下降沿触发，当选择为 WK_CNT 或 GPIO_WK 时，不用配置。
2. 中断源选择为 COM0 时，可以通过配置 T2CON.6 来选择上升沿触发或者下降沿触发。
3. 当{CCEN.7,CCEN.6}，{CCEN.5,CCEN.4}，{CCEN.3,CCEN.2}，{CCEN.1,CCEN.0}配置为 2’b10 时，中断源选择为 COM0/1/2/3，对应的中断标志位参见上述表格。

4. 当选择 WK_CNT、GPIO_WK、ADC、PWM、USB 和 RF 为中断源时，中断标志位参见各自对应的章节。

3.3 中断优先级

PAN262x 可设定 4 个中断优先级，可对各个中断组的优先级进行调整，不支持对单个中断源的优先级进行调整。一共有 6 个中断组，优先级的调整通过寄存器 IP0、IP1 来设置。下表为中断组优先级控制表：

中断组	组成员（左边比右边优先级高）		组优先级控制位
Group 0	INT0/WK_CNT	I2C	IP1.0&IP0.0
Group 1	Timer0	SPI	IP1.1&IP0.1
Group 2	INT1/GPIO_WK	ADC/COM0	IP1.2&IP0.2
Group 3	Timer1	PWM/COM1	IP1.3&IP0.3
Group 4	UART	USB/COM2	IP1.4&IP0.4
Group 5		RF/COM3	IP1.5&IP0.5

四个中断优先级的设置如下表所示：

级别	优先级	IP1.X	IP0.X
Level 0	Lowest	0	0
Level 1	Low	0	1
Level 2	High	1	0
Level 3	Highest	1	1

3.4 中断相关寄存器

地址	名称	复位值	功能描述
0x88	TCON	0x00	Timer/Counter 控制寄存器
0xC8	T2CON	0x00	COM0 边沿触发模式选择
0xA8	IEN0	0x00	中断使能寄存器 0
0xB8	IEN1	0x00	中断使能寄存器 1
0xA9	IP0	0x00	中断优先级寄存器 0
0xB9	IP1	0x00	中断优先级寄存器 1
0xC0	IRCON	0x00	中断标志寄存器

3.4.1 TCON（BIT3~BIT0）

Register	Address	R/W	Description	Reset Value
TCON	0x88	R/W	Timer/Counter 控制寄存器	0x00

Bits	Description	
7	TF1	TIMER1 溢出标志。当 TIMER1 溢出时由硬件置位。当进入中断处理程序后由硬件自动清除，也可以由软件清除。
6	TR1	TIMER1 运行控制。如果清除，TIMER1 停止工作。
5	TF0	TIMER0 溢出标志。当 TIMER0 溢出时由硬件置位。当进入中断处理程序后由硬件自动清除，也可以由软件清除。
4	TR0	TIMER0 运行控制。如果清除，TIMER0 停止工作。
3	IE1	外部中断 INT1 标志位。INT1（边沿/电平）触发置位，进入中断后自动清零。
2	IT1	外部中断 INT1 触发模式选择，0：低电平触发，1：下降沿触发
1	IE0	外部中断 INT0 标志位。INT0（边沿/电平）触发置位，进入中断后自动清零。
0	IT0	外部中断 INT0 触发模式选择，0：低电平触发，1：下降沿触发

3.4.2 T2CON（BIT6）

Register	Address	R/W	Description	Reset Value
T2CON	0xC8	R/W	COM0 边沿触发模式选择	0x00

Bits	Description
7	用于 TIMER2，参考 TIMER 模块
6	Reserved
4：0	用于 TIMER2，参考 TIMER 模块

3.4.3 IEN0

Register	Address	R/W	Description	Reset Value
IEN0	0xA8	R/W	中断使能寄存器 0	0x00

Bits	Description	
7	EAL	全体中断使能位： 0：关闭所有中断 1：全体中断使能，若要打开某一个中断，还需要打开它对应的中断使能位
6	WDT	用于 WDT，参考 WDT 模块
5	ET2	TIMER2 中断使能： 0：TIMER2 中断关闭 1：并且 EAL=1，TIMER2 中断使能
4	ES0	UART 中断使能： 0：UART 中断关闭 1：并且 EAL=1，UART 中断使能
3	ET1	TIMER1 溢出中断使能： 0：TIMER1 溢出中断关闭 1：并且 EAL=1，TIMER1 溢出中断使能
2	EX1	外部中断 INT1 使能： 0：外部中断 INT1 关闭 1：并且 EAL=1，外部中断 INT1 使能
1	ET0	TIMER0 溢出中断使能： 0：TIMER0 溢出中断关闭 1：并且 EAL=1，TIMER0 溢出中断使能
0	EX0	外部中断 INT0 使能： 0：外部中断 INT0 关闭 1：并且 EAL=1，外部中断 INT0 使能

3.4.4 IEN1

Register	Address	R/W	Description	Reset Value
IE1	0xB8	R/W	中断使能寄存器 1	0x00

Bits	Description	
7	EXEN2	用于 TIMER2，参考 TIMER 模块
6	SWDT	用于 WDT，参考 WDT 模块
5	RF_IE /COM3_IE	RF/COM3（Timer2 的比较器 3）中断使能： 0：RF/COM3 中断关闭 1：并且 EAL=1，COM3 中断使能，RF 还需要配置额外寄存器 CONFIG3/4/5
4	USB_IE /COM2_IE	USB/COM2（Timer2 的比较器 2）中断使能： 0：USB/COM2 中断关闭 1：并且 EAL=1，COM2 中断使能，USB 还需要配置额外寄存器 IntrIn1E、IntrOut1E、IntrUSB
3	PWM_IE /COM1_IE	PWM/COM1（Timer2 的比较器 1）中断使能： 0：PWM/COM1 中断关闭 1：并且 EAL=1，COM1 中断使能，PWM 还需要配置额外寄存器 PWM_INTEN
2	ADC_IE /COM0_IE	ADC/COM0（Timer2 的比较器 0）中断使能： 0：ADC/COM0 中断关闭 1：并且 EAL=1，COM0 中断使能，ADC 还需要配置额外寄存器 ADC_CTL0.1
1	SPI_IE	SPI 中断使能： 0：SPI 中断关闭 1：并且 EAL=1，SPI 中断使能
0	I2C_IE	I2C 中断使能： 0：I2C 中断关闭 1：并且 EAL=1，I2C 中断使能

3.4.5 IRCON

Register	Address	R/W	Description	Reset Value
IRCON	0xC0	R/W	中断请求控制寄存器	0x00

Bits	Description	
7	EXF2	Timer2 重载中断标志位： 0：表示未发生 Timer2 重载中断 1：表示已发生 Timer2 重载中断（写 0 清零）
6	TF2	Timer2 溢出中断标志位： 0：表示未发生 Timer2 溢出中断 1：表示已发生 Timer2 溢出中断（写 0 清零），读此寄存器中断标志自动清零
5	RF/COM3	RF/COM3（Timer2 的比较器 3）中断标志位： 0：表示未发生中断 1：表示已发生中断（写 0 清零），需要查询 STATUS.4/5/6 确定具体中断类型，读此寄存器中断标志自动清零
4	USB/COM2	USB/COM2（Timer2 的比较器 2）中断标志位： 0：表示未发生中断 1：表示已发生中断（写 0 清零），需要查询 IntrIn1.0/1 和 IntrOut1.1 确定具体中断类型，读此寄存器中断标志自动清零
3	PWM/COM1	PWM/COM1（Timer2 的比较器 1）中断标志位： 0：表示未发生中断 1：表示已发生中断（写 0 清零），需要查询 PWM_INTSTS.0/1/2/3/4/5 确定具体中断类型，读此寄存器中断标志自动清零
2	ADC/COM0	ADC/COM0（Timer2 的比较器 0）中断标志位： 0：表示未发生中断 1：表示已发生中断（写 0 清零），需要查询 ADC_STATUS0.0/1/2 确定具体中断类型，读此寄存器中断标志自动清零
1	中断 2	Reserved
0	中断 7	Reserved

3.4.6 IP0，IP1

Register	Address	R/W	Description	Reset Value
IP0	0xA9	R/W	中断优先级寄存器 0	0x00
IP1	0xB9	R/W	中断优先级寄存器 1	0x00

Bits	Description	
IP0.7	Reserved	Reserved
IP0.6	WDTS	用于 WDT，参考 WDT 模块。
IP0.5		每个位与来自 IP1 寄存器的相应位一起指定了各自中断优先级组的优先级。
IP0.4		
IP0.3		
IP0.2		
IP0.1		
IP0.0		

Bits	Description	
IP1.7/6	Reserved	Reserved
IP1.5		每个位与来自 IP0 寄存器的相应位一起指定了各自中断优先级组的优先级。
IP1.4		
IP1.3		
IP1.2		
IP1.1		
IP1.0		

3.5 中断处理

中断系统遵循以下两条基本规则：

1. 低优先级中断源可被高优先级中断源所中断，而高优先级中断源不能被任何中断源所中断；
2. 一种中断源不管是高优先级或低优先级，一旦得到响应，与它同级的中断源不能再中断它。

当同时收到几个同一优先级的中断时，响应哪一个中断源取决于内部查询顺序。其优先级排列见 3.3 章节中同级中断优先顺序。

ADC 中断、PWM 中断、USB 中断、RF 中断等都包含了若干个中断源。以 PWM 中断为例，先要判断是 COM1 中断还是 PWM 中断，PWM 中断标志寄存器也包含 6 个通道的中断，用户可以在 ISR 中通过软件查询的方式判断具体是哪个中断源，然后在 ISR 中清除相应的中断标志位。

第4章 复位和时钟

4.1 复位

模拟端给出的复位源有：

1. 上电复位：NRESET_POR
2. PAD 复位：NRESET_PAD，与 GPIO P3.6 复用，由软件决定是否用作复位（默认为复位功能）
3. 低压复位：LVRH_OUT

数字系统复位源包括：

1. 全局软件复位：srst
2. Watch dog 复位：wdtrst
3. OCDS 调试口复位：debugrst
4. 各个外设软件复位：xxx_rsts

注：其中全局软件复位 srst 通过连续两次将“1”值写入“srst”寄存器（0F7h）中的“srstreq(srst 的最低位)”位后，生成软件复位。

4.1.1 系统复位框图

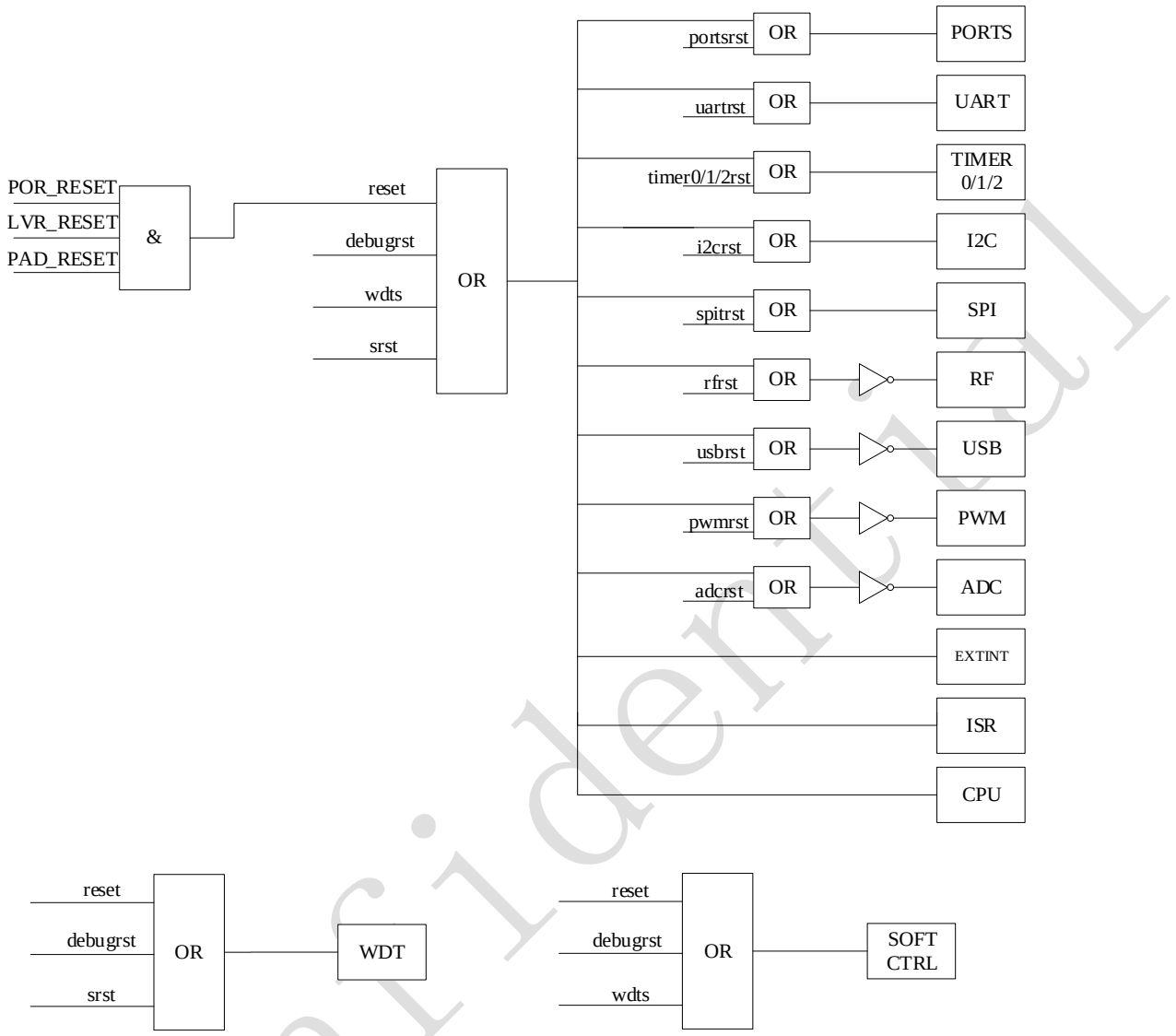


图 4.1　PAN262x 复位系统框图

说明：

RF、PWM、USB、ADC 模块使用异步复位，低电平有效，其他模块为同步复位，高电平有效。

4.1.1.1 上电复位 POR

PAN262x 内置 V_{DD} 电压上升检测电路，当系统电压开始上升时，会产生一个上电复位高电平信号，该复位拉高后 C2_WINDOW 延时计数器计数 2ms，溢出后复位撤离，MCU 开始执行指令。

4.1.1.2 LVRH 低压复位

PAN262x 芯片内部集成低压复位功能，低压复位可以份不同档位使用（由寄存器 LVRH_SEL 设置）。当系统工作电压 VDD 低于 LVR 设定的复位阈值电压(VLVR)时，LVRH 检测到 VDD<VLVR 即输出低电平并且系统延迟 15*Tcpu 后复位系统。这将确保 PAN262x 在 VDD<VLVR 时不再继续执行程序并且保持低压复位状

态，直至电源电压上升到工作电压跳变点以上。

LVR 类型	电压跳变点	时间阈值
LVRH	由寄存器 LVRH_SEL 选择	15*Tmcu

4.1.1.3 外部 PAD 复位

外部 PAD 复位信号 PAD_NRST 的传输路径中有一个 PAD_NRST 噪声滤波器，用来检测和滤除小的尖脉冲信号。PAD_NRST 的最小脉冲宽度是 SYS_RSTDBC[7:0]*512*Tcpu，当 P36 脚上出现大于 SYS_RSTDBC[7:0]*512*Tcpu 的低电平脉冲时，发生外部复位。发生外部复位时，系统复位。寄存器 SYS_RSTDBC 不能配为 0。

在 DEEPSLEEP 模式下，复位滤波器功能自动失效。退出 STOP 模式后，复位滤波器功能自动开启。

4.1.2 复位标志

复位	标志位	说明
NRESET_POR	pors / PERRST0[7]	上电复位POR状态指示寄存器，1：POR复位，写0清除
LVRH_OUT	lvrhs / PERRST1[6]	LVRH 复位状态指示寄存器，1：LVRH 复位，写 0 清除
NRESET_PAD	padrsts / PERRST1[7]	P36 PAD 复位状态指示寄存器，1：PAD 复位，写 0 清除
srst	srstflag / srst[0] / SFR(0xF7)	软复位状态指示寄存器，1：软复位，NRESET_POR / LVRH_OUT / NRESET_PAD / debugrst / wdts 复位清除
wdts	wdts / IP0[6]	WDT 复位状态指示寄存器，1：WDT 复位，写 0 清除
debugrst	OCDSCTRL[5]	debugrst = rst(OCDSCTRL[5]) & ocdsen(OCDSCTRL[4]);

4.1.3 各模块复位源

本章节按照模块罗列相关寄存器的复位信号源，同一个模块中复位源不同的寄存器单独罗列，同一个模块中复位源一致的统一罗列。

4.1.3.1 8051 CPU

寄存器	复位源
CPU 内核寄存器	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst

4.1.3.2 RCC

寄存器	复位源	备注
LVRH_CTRL、LVRH_SEL	NRESET_POR	
PERRST0、PERRST1、CLK_TOP_CTRL、RCL_CTRL0、RCL_CTRL1、RCH_CTRL0、XTH_CTRL、DPLL_CTRL、PERCLKEN0、PERCLKEN1、RCHCAL_1H、RCHCAL_1L、RCHCAL_2H、RCHCAL_2L、XTH_CTRL1、RCHCAL_EN	LVRH_OUT、NRESET_POR、NRESET_PAD、debugrst、wdts、srst	
RCHCAL_RO_1H、RCHCAL_RO_1L、RCHCAL_RO_2	/	只读

4.1.3.3 Timer

模块	寄存器	复位源
Timer0	TCON、TMOD、TL0、TH0	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[2]
Timer1	TCON、TMOD、TL1、TH1、ADCON	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[3]
Timer2	IRCON、TL2、TH2、CRCL、CRCH、T2CON、CCEN、CCL1、CCH1、CCL2、CCH2、CCL3、CCH3、TCAPCON、TCAPSTA、IEN1	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[4]

4.1.3.4 电源管理

寄存器	复位源
RCH_SYNC_TRG[4](P36_NRST_EN)	NRESET_POR、LVRH_OUT
P1_WAKEUP_EN、P3_WAKEUP_EN、P1_WAKEUP_SEL、P3_WAKEUP_SEL、WAKEUP_CNT_CTRL、WAKEUP_INT_CTRL、ANA_PWR_CTRL0、ANA_PWR_CTRL1、ANA_PWR_CTRL2、RCH_DLY_CTRL、RCH_SYNC_TRG[7:5]（FLASH_SIZE、USB_EN、ADC_EN)、RCH_SYNC_TRG[0]	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst

4.1.3.5 中断系统

寄存器	复位源
IEN0[7]、IEN0[5:0]、IEN1[5:0]、IEN2[5:0]、IP0[5:0]、IP1[5:0]、PCON[5]	NRESET_POR 、 LVRH_OUT 、 NRESET_PAD 、 debugrst、wdts、srst

4.1.3.6 GPIO

寄存器	复位源	备注
SYS_RSTDBC	NRESET_POR、LVRH_OUT	
C2_CONNECT_R	NRESET_POR、LVRH_OUT、NRESET_PAD 、 debugrst 、 wdts、srst	
SYS_P1_MODE0 、 SYS_P1_MODE1 、 SYS_P3_MODE0 、 SYS_P3_MODE1、SYS_P1_DIEN、SYS_P3_DIEN、SYS_P1_PUEN、SYS_P3_PUEN、SYS_P1_PDEN、SYS_P3_PDEN、SYS_P1_OE、SYS_P3_OE、SYS_P1_MFP0、SYS_P1_MFP1、SYS_P1_MFP2、SYS_P3_MFP0、SYS_P3_MFP1、SYS_P3_MFP2、SYS_USB_CTRL、SYS_USB_CTRL1、SYS_USB_CTRL2、EINT0_DBC、EINT1_DBC	NRESET_POR、LVRH_OUT、NRESET_PAD 、 debugrst 、 wdts、srst	
SYS_READ	/	只读

4.1.3.7 OCDS

寄存器	复位源
C2_COM： C2_REQ、C2_CONNECT、C2_PERMISSION	NRESET_POR、LVRH_OUT
C2_DEBUGPORT：all OCDS_UNIT：all	C2_WINDOW_END、NRESET_POR、LVRH_OUT、NRESET_PAD

4.1.3.8 其他模块

模块	复位源
UART	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[1]
I2C	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[5]
SPI	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST0[6]
RF	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST1[0]
ADC	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST1[1]、ADC_EN (RCH_SYNC_TRG[5])

PWM	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST1[2]
WDT	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、srst
USB	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST1[3]、USB_EN (RCH_SYNC_TRG[6])
RF 小数字 Dig2	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst、PERRST1[4]
Flash 控制器	NRESET_POR、LVRH_OUT、NRESET_PAD、debugrst、wdts、srst

注：ADC、USB 模块未使能时，相关寄存器一直处于复位状态，需要先使能再修改寄存器。

4.2 时钟

4.2.1 模拟时钟

系统时钟源有：

1，RCH：15MHz 内部高速 RC；

2，XTH：16M Hz 外部高速晶振；

3，RCL：60KHz（WDT）、15KHz（Wakeup Counter）内部低速 RC；

4，DPLL_RF：96M Hz DPLL 时钟，RF 收发机解调器使用；

5，DPLL_USB：48M Hz DPLL 时钟，USB 使用；

6，DPLL_16M：16M Hz DPLL 时钟，可用作系统时钟。

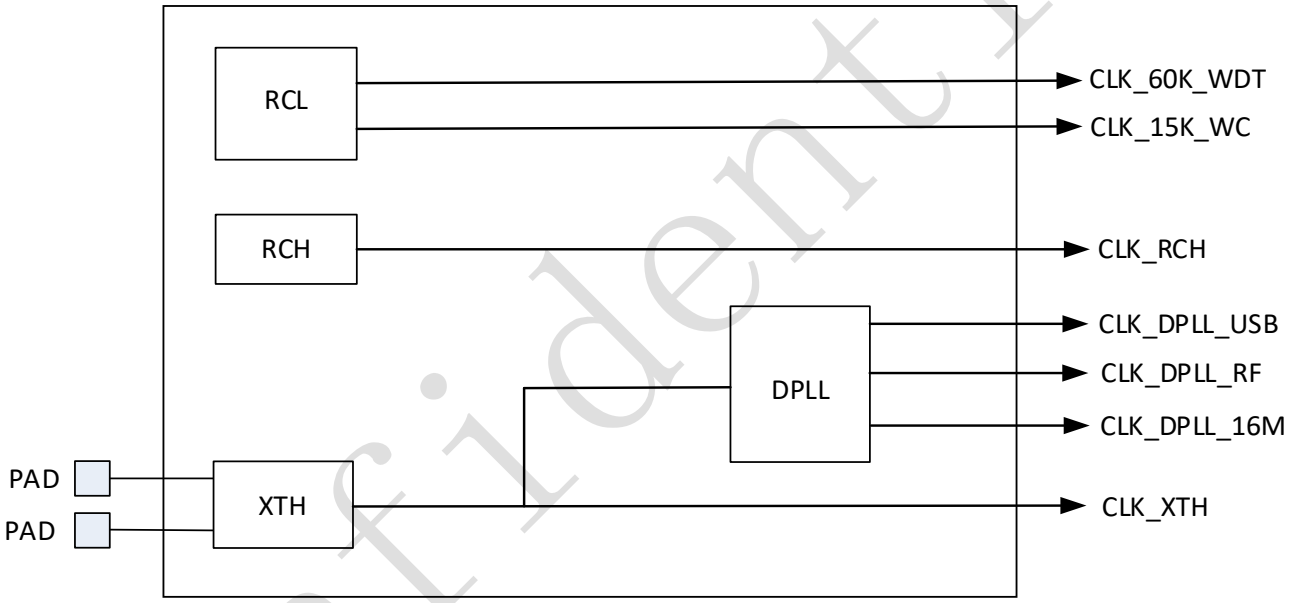


图 4.2　PAN262x 时钟系统框图

4.2.2 模拟时钟示意图

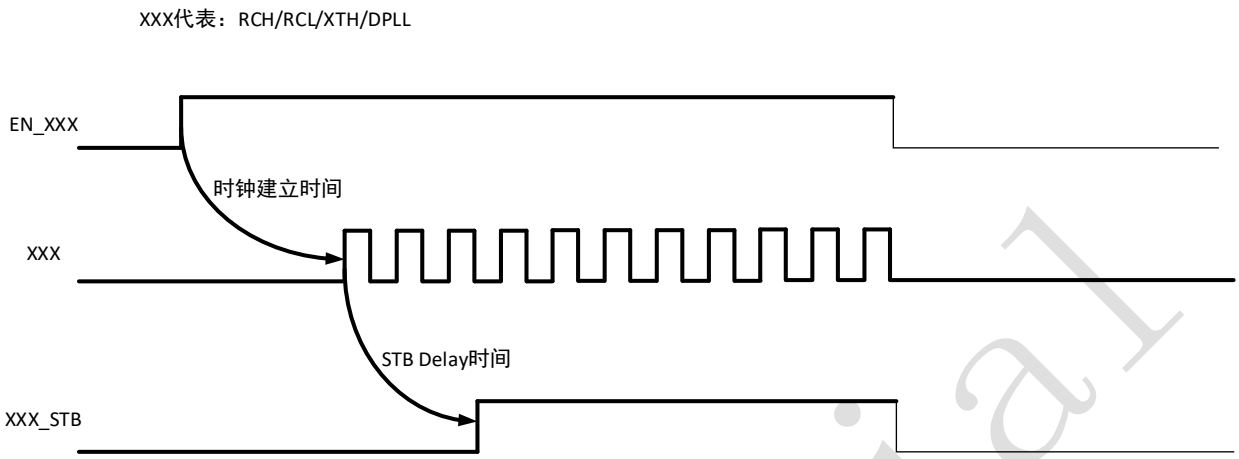


图 4.3　PAN262x 模拟时钟示意图

EN_XXX 为相应时钟的使能信号，使能信号拉高后，经过一个时钟建立时间，输出时钟给数字。

STB Delay 时间由两个 bit 控制，用于控制 XXX_STB 拉高的时间，具体参考寄存器定义。

4.2.3 数字时钟

4.2.3.1 系统时钟结构图

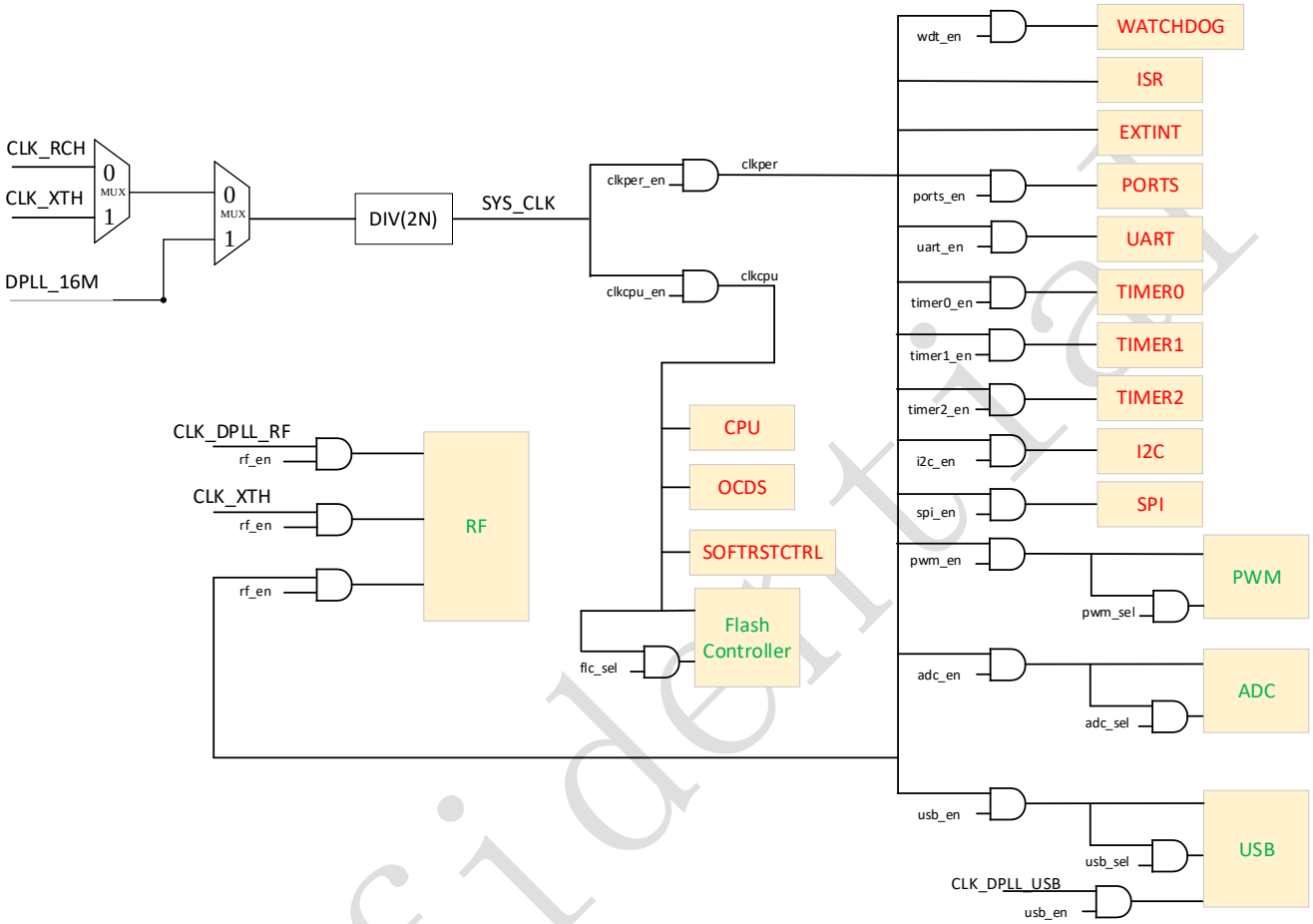


图 4.4　PAN262x 系统时钟示意图

说明：

1，WatchDog 计数器时钟使用的是 60kHz 的 RCL 时钟；

2，PWM、ADC、USB 和 FlashController 的时钟分为寄存器读写时钟和工作时钟。pwm_sel、adc_sel、usb_sel、flc_sel 由内部总线译码而来，只有在读写寄存器时使能。

4.3 寄存器

4.3.1 寄存器列表

时钟、复位寄存器映射，SFR 接口：

Register	Offset	R/W	Description	Reset Value
RCC_SEL	0xFC	R/W	RCC 寄存器地址选择	0x00
RCC_DAT	0xFD	R/W	RCC 寄存器数据读写寄存器	0x00

时钟、复位寄存器列表：

Register	RCC_SEL	R/W	Description	Reset Value
PERRST0	0x0	R/W	外设复位控制	0x80
PERRST1	0x1	R/W	外设复位控制	0x00
CLK_TOP_CTRL	0x3	R/W	系统时钟全局控制寄存器	0x13
RCL_CTRL0	0x4	R/W	RCL 控制寄存器	0x3F
RCL_CTRL1	0x5	R/W	RCL 控制寄存器	0x08
RCH_CTRL0	0x6	RS/W	RCH 控制寄存器	0x3B
LVRH_CTRL	0x7	RS/W	LVRH 相关控制	0x05
XTH_CTRL	0x8	RS/W	XTH 控制寄存器	0x17
DPLL_CTRL	0xA	RS/W	DPLL 控制寄存器	0x0B
PERCLKEN0	0xC	R/W	外设时钟使能控制	0xFF
PERCLKEN1	0xD	R/W	外设时钟使能控制	0x00
RCHCAL_1H	0xE	R/W	RCH 校准控制寄存器 1 高字节	0x04
RCHCAL_1L	0xF	R/W	RCH 校准控制寄存器 1 低字节	0x00
RCHCAL_2H	0x10	R/W	RCH 校准控制寄存器 2 高字节	0x3E
RCHCAL_2L	0x11	R/W	RCH 校准控制寄存器 2 低字节	0x00
RCHCAL_RO_1H	0x12	RO	RCH 校准只读寄存器 1 高字节	0x00
RCHCAL_RO_1L	0x13	RO	RCH 校准只读寄存器 1 低字节	0x00
RCHCAL_RO_2	0x14	RO	RCH 校准只读寄存器 2	0x1F
RCHCAL_EN	0x15	WO	RCH 校准使能寄存器，自动清零	0x00
LVRH_SEL	0x16	R/W	LVRH 复位点选择	0x01
XTH_CTRL1	0x17	R/W	XTH 控制寄存器 1	0x04

4.3.2 寄存器描述

4.3.2.1 PERRST0

Register	RCC_SEL	R/W	Description	Reset Value
PERRST0	0x0	R/W	外设复位控制寄存器 0	0x80

Bits	Description	
[7]	pors	上电复位 POR 状态指示寄存器，1：POR 复位，写 0 清除
[6]	spirst	SPI 复位控制，1：复位；0：释放
[5]	i2crst	I2C 复位控制，1：复位；0：释放
[4]	timer2rst	TIMER2 复位控制，1：复位；0：释放
[3]	timer1rst	TIMER1 复位控制，1：复位；0：释放
[2]	timer0rst	TIMER0 复位控制，1：复位；0：释放
[1]	uartrst	UART 复位控制，1：复位；0：释放
[0]	portsrst	PORT 复位控制，1：复位；0：释放

4.3.2.2 PERRST1

Register	RCC_SEL	R/W	Description	Reset Value
PERRST1	0x1	R/W	外设复位控制寄存器 1	0x00

Bits	Description	
[7]	PADRSTS	P36 PAD 复位状态指示寄存器，1：PAD 复位，写 0 清除
[6]	LVRHS	Lvrh 复位状态指示寄存器，1：lvrh 复位，写 0 清除
[5]	Reserved	
[4]	DIG2NRST	RF PLL 小数字复位，1：复位；0：释放
[3]	USBRST	USB 复位，1：复位；0：释放
[2]	PWMRST	PWM 复位，1：复位；0：释放
[1]	ADCRST	ADC 复位，1：复位；0：释放
[0]	RFRST	RF 大数字复位，1：复位；0：释放

4.3.2.3 CLK_TOP_CTRL

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
CLK_TOP_CTRL	0x3	RW	系统时钟全局控制	0x13	0x8F

Bits	Description		
[7:6]	SYS_CLK_SEL	RW	系统时钟选择，默认 RCH 00：RCH 01：不支持 10：XTH，建议使用 XTH 作为系统时钟 11：DPLL_16M
[5:4]	SYS_CLK_DIV	RW	系统时钟分频控制，默认不分频 00：不分频 01：2 分频 10：4 分频 11：6 分频
[3]	EN_DPLL	RW	DPLL 使能信号，默认关闭 0：关闭 1：打开 接收和发射数据需要打开 DPLL
[2]	XTH_EN	RW	控制 PWR_UP 以及 CE_INT，晶振 16M 使能控制，默认关闭 0：关闭 1：打开
[1]	EN_RCH	RW	内部 15M 高速 RC 时钟使能，默认打开 0：关闭 1：打开 当使用看门狗时，推荐打开 RCH
[0]	EN_RCL	RW	内部 60K/15K 低速 RC 时钟使能，默认打开 0：关闭 1：打开

4.3.2.4 RCL_CTRL0

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCL_CTRL0	0x4	R/W	RCL 校正控制寄存器	0x3F	0x79

Bits	Description		
[7]	Reserved	1’b0	
[6:4]	RCL_CAL_PER_SEL	3’b011	RCL 校准周期控制 000 ： 32ms

			001 ： 64ms 010 ： 128ms 011 ： 256ms 100 ： 512ms 101 ： 1024ms 110 ： 2048ms 111 ： 4096ms
[3]	RCL_CAL_EN	1’b1	RCL 校准模块使能控制，1：打开，0：关闭
[2:0]	RCL_CAL_H_SEL	3’b111	RCL 校准时间控制 000 ： 62.5us 001 ： 93.75us 010 ： 156.25us 011 ： 281.25us 100 ： 531.25us 101 ： 1031.25us 110 ： 2031.25us 111 ： 4031.25us

4.3.2.5 RCL_CTRL1

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCL_CTRL1	0x5	R/W	RCL 控制寄存器	0x08	默认值

Bits	Description		
[7:6]	Reserved	2’b00	
[5]	RCL_CAL_TEST_MODE	1’b0	RCL 校准模块测试模式控制，测试模式下送给模拟的 EN_RCLCAL = RCL_CAL_TEST_MODE_DAT 1：打开，0：关闭
[4]	RCL_CAL_TEST_MODE_DAT	1’b0	RCL 校准模块测试模式数据，测试模式下 EN_RCLCAL = RCL_CAL_TEST_MODE_DAT
[3:1]	RCL_ICP	3’b100	RCL 校准 charge pump 充电速度调节, 0.25uA~2uA，0.25uA/step
[0]	EN_RCLCPCAL	1’b0	RCL 频率 CP 粗调，1：打开，0：关闭

4.3.2.6 RCH_CTRL0

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCH_CTRL0	0x6	RS/W	RCH 控制寄存器	0x3B	0x2B

Bits	Description		
[7]	RCH_RDY	RO	指示 RCH 已经稳定

[6:4]	RCH_CAP_TRIM	3'b011	电容控制位，改变 RCH 频率。000 --> 111 频率降低 应用端需要读取 CP 校准值，手动模式推荐 010。
[3:2]	RCH_TEMP_TRIM	2'b10	RCH 温度系数控制位
[1:0]	RCH_DELAY	2'b11	RCH 稳定时间控制，01:2us，10:4us，11:8us

4.3.2.7 LVRH_CTRL

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
LVRH_CTRL	0x7	RS/W	LVRH 控制寄存器	0x05	默认值

Bits	Register	Description	Reset
[7:3]	Reserved		00000
[2]	LVRH_FLAG	只读寄存器，指示 LVRH_OUT 输出状态	1
[1]	EN_LVRH	高压 LVR 复位电路使能控制，1：打开；0：关闭	0
[0]	LVRHRST_CLOSE	LVRH 复位功能屏蔽，1：屏蔽；0：不屏蔽	1

4.3.2.8 XTH_CTRL

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
XTH_CTRL	0x8	RS/W	XTH 控制寄存器	0x17	0x1F

Bits	Description		
[7]	Reserved		
[6]	XTH_RDY	RO	晶振稳定指示信号，1: 频率稳定，0: 不稳定
[5]	CHIRPFLAG	RO	晶振快速启动结束信号，1：快速启动阶段结束，0：未结束
[4]	EN_XTAL_RDY	1'b1	晶振快速起振 RDY 使能控制，1: 打开，0: 关闭
[3]	FAST_TRIM	1'b0	快速启动电路控制挡位，改变输出频率范围。
[2]	EN_FAST	1'b1	晶振快速起振使能控制，1: 打开，0: 关闭
[1]	XTAL_BC	1'b1	16M 晶振电流控制档位
[0]	XTAL_RDY_DE-LAY	1'b1	晶振稳定时间控制，0: 160us，1: 320us

4.3.2.9 DPLL_CTRL

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
DPLL_CTRL	0xA	RS/W	DPLL 控制寄存器	0x0B	默认值

Bits	Description		
[7:6]	Reserved		

[5]	DPLL_RDY	RO	表示 DPLL 已经稳定
[4]	DPLL_TEST_EN	1'b0	使能 DPLL 8 分频测试输出，同时需要配置 IOMUX 到 P12
[3:2]	DPLL_CP_BC	2'b10	DPLL charge pump 电流调节，00:3.5uA，01:7uA，10:10.5uA，11:14uA
[1:0]	DPLL_RDY_DE-LAY	2'b11	DPLL 稳定时间控制，00:7us，01:12us，10:22us，11:44us

4.3.2.10 PERCLKEN0

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
PERCLKEN0	0xC	R/W	外设时钟使能控制寄存器 0	0xFF	TBD

Bits	Description	
[7]	wdgclken	Watchdog 时钟使能控制，1：打开，0：关闭
[6]	spiclken	SPI 时钟使能控制，1：打开，0：关闭
[5]	i2cclken	I2C 时钟使能控制，1：打开，0：关闭
[4]	timer2clken	TIMER2 时钟使能控制，1：打开，0：关闭
[3]	timer1clken	TIMER1 时钟使能控制，1：打开，0：关闭
[2]	timer0clken	TIMER0 时钟使能控制，1：打开，0：关闭
[1]	uartclken	UART 时钟使能控制，1：打开，0：关闭
[0]	portsclken	PORTS 时钟使能控制，1：打开，0：关闭

4.3.2.11 PERCLKEN1

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
PERCLKEN1	0xD	R/W	外设时钟使能控制寄存器 1	0x00	0x01

Bits	Description	
[7:4]	Reserved	
[3]	usbclken	USB 时钟使能控制，1：打开，0：关闭
[2]	pwmclken	PWM 时钟使能控制，1：打开，0：关闭
[1]	adcclken	ADC 时钟使能控制，1：打开，0：关闭
[0]	rfclken	RF 数字时钟使能控制，1：打开，0：关闭 射频寄存器读写，需要 rfclken 打开

4.3.2.12 RCHCAL_1H

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_1H	0xE	R/W	RCH 校正控制寄存器 1H	0x04	默认值

Bits	Description	
[7:3]	Reserved	
[2:0]	rchcal_ref_count[10:8]	一个待校准时钟周期内的基准 count 值高 3bit

4.3.2.13 RCHCAL_1L

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_1L	0xF	R/W	RCH 校正控制寄存器 1L	0x00	默认值

Bits	Description	
[7:0]	rchcal_ref_count[7:0]	一个待校准时钟周期内的基准 count 值低 8 bit

4.3.2.14 RCHCAL_2H

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_2H	0x10	R/W	RCH 校正控制寄存器 2H	0x3E	默认值

Bits	Description		Reset
[7]	rchcal_code_cfg_en	RCH 手动校正使能，1：使能，0：关闭，使能后 rch_trim= rchcal_code_cfg	0

[6:1]	rchcal_code_cfg	RCH 手动校正 code，手动值推荐 101000	011111
[0]	rchcal_wait_test_cnt[8]	一次校准结束后距离下次校准的时间间隔第 8 bit	0

4.3.2.15 RCHCAL_2L

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_2L	0x11	R/W	RCH 校正控制寄存器 2L	0x00	默认值

Bits	Description		Reset
[7:0]	rchcal_wait_test_cnt[7:0]	一次校准结束后距离下次校准的时间间隔低 8 bit	0x00

4.3.2.16 RCHCAL_RO_1H

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_RO_1H	0x12	RO	RCH 校正只读寄存器 1H	0x00	默认值

Bits	Description		Reset
[7:3]	Reserved		
[2:0]	rchcal_scounter[10:8]	校准结束后输出的待校准时钟周期的 count 数高 3 bit，即最终的校准频率 F=16M/scounter	000

4.3.2.17 RCHCAL_RO_1L

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_RO_1L	0x13	RO	RCH 校正只读寄存器 1L	0x00	默认值

Bits	Description		Reset
[7:0]	rchcal_scounter[7:0]	校准结束后输出的待校准时钟周期的 count 数低 8bit，即最终的校准频率 F=16M/scounter	0x00

4.3.2.18 RCHCAL_RO_2

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_RO_2	0x14	RO	RCH 校正只读寄存器 2	0x1F	默认值

Bits	Description		Reset
[7:6]	Reserved		00
[5:0]	rch_trim[5:0]	RCH 校正结果	011111

4.3.2.19 RCHCAL_EN

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
RCHCAL_EN	0x15	WO	RCH 自动校正触发寄存器	0x00	0x01

Bits	Description		Reset
[7:1]	Reserved		0000000
[0]	rchcal_en	RCH 自动校正触发寄存器，自动清零。 脉冲宽度应大于 1/16MHz，什么时候校正完全由软件控制	0

4.3.2.20 LVRH_SEL

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
LVRH_SEL	0x16	R/W	LVRH 复位点选择寄存器	0x01	默认值

Bits	Description		Reset
[7]	Reserved		
[6:0]	LVRH_SEL	LVRH 复位点选择： 0000001：1.85V 0000010：2V 0000100：2.2V 0001000：2.35V 0010000：2.5V 0100000：2.65V 1000000：2.8V	0000001

4.3.2.21 XTH_CTRL1

Register	RCC_SEL	R/W	Description	Reset Value	推荐值
XTH_CTRL1	0x17	R/W	XTH 控制寄存器 1	0x04	默认值

Bits	Description		Reset
[7:3]	Reserved		
[2:0]	XTAL_CAP	16M 晶振电容选择，000~111 电容变化范围为 14pF~21pF，step 为 1pF	100

第5章 电源管理

PAN262x 提供两种低功耗模式，IDLE 和 STOP。IDLE 模式下 CPU 时钟停止，外设时钟运行。STOP 模式下，CPU 和外设时钟全部停止。IDLE 又称作 SLEEP 模式，STOP 又称作 DEEP_SLEEP 模式。

PAN262x 为多电压阈设计，内部逻辑分为高压区逻辑以及低压区逻辑。其中高压区逻辑为 always on 逻辑，而低压区逻辑在 DEEP_SLEEP 模式下会保电，但不能运行。

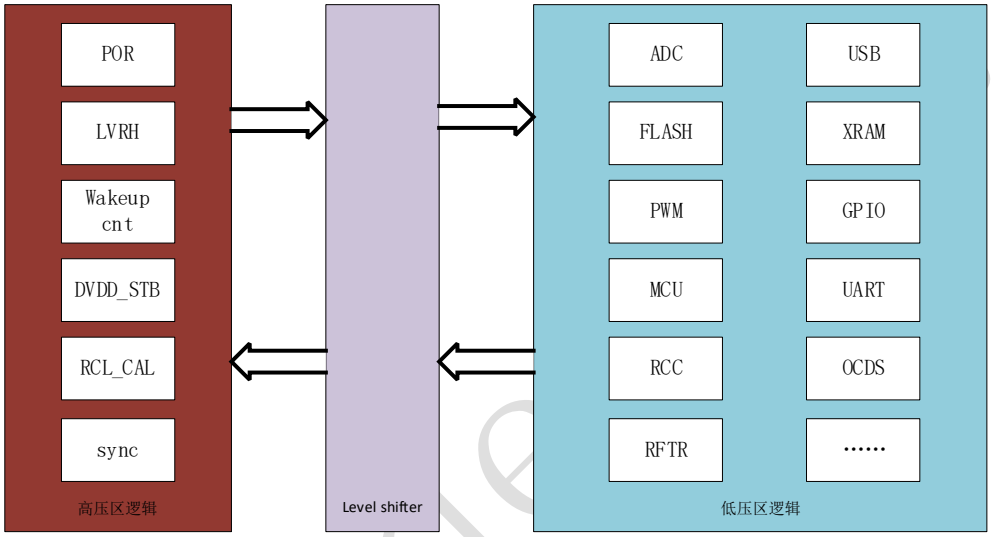


图 5.1 PAN262x 电压阈设计

5.1 低功耗模式

默认情况下，系统上电复位或内部其他复位（LVR 复位，WDT 复位，Soft 复位）后，系统进入正常运行模式。正常运行模式，系统默认使用 RCH 时钟。为了降低功耗，提供以下几种低功耗模式：

低功耗模式	描述	进入方式	唤醒	供电	功耗
DeepSleep	系统高速时钟、RCL、HPLDO 都关闭，RF 相关 LDO 关闭	PCON.1 (stop)写 1	GPIO	LPLDO	700 nA
DeepSleep1	系统高速时钟、HPLDO 都关闭，RF 相关 LDO 关闭	PCON.1 (stop)写 1	GPIO Wakeup Counter	LPLDO	1 uA
Sleep	仅 CPU 时钟关闭	PCON.0 (idle)写 1	GPIO Wakeup Counter WDT Timer0/1/2 UART INT0/1	HPLDO	850 uA

			I2C SPI ADC PWM USB RFTR		

5.1.1 深度睡眠

深度睡眠 DeepSleep 模式下低压区 DVDD 区域由弱 LDO LPLDO 供电，强 LDO HPLDO 和 BandGap 关闭，高压区 VDD 区域维持供电，RCL 打开，其他高速时钟（RCH，XTH 和 DPLL）全部关闭。

进入方式：软件向寄存器 PCON 的 BIT1(STOP) 写 1。

唤醒条件：所有 GPIO 可以选择高电平或者低电平唤醒，Wakeup Counter 唤醒。

5.1.2 睡眠

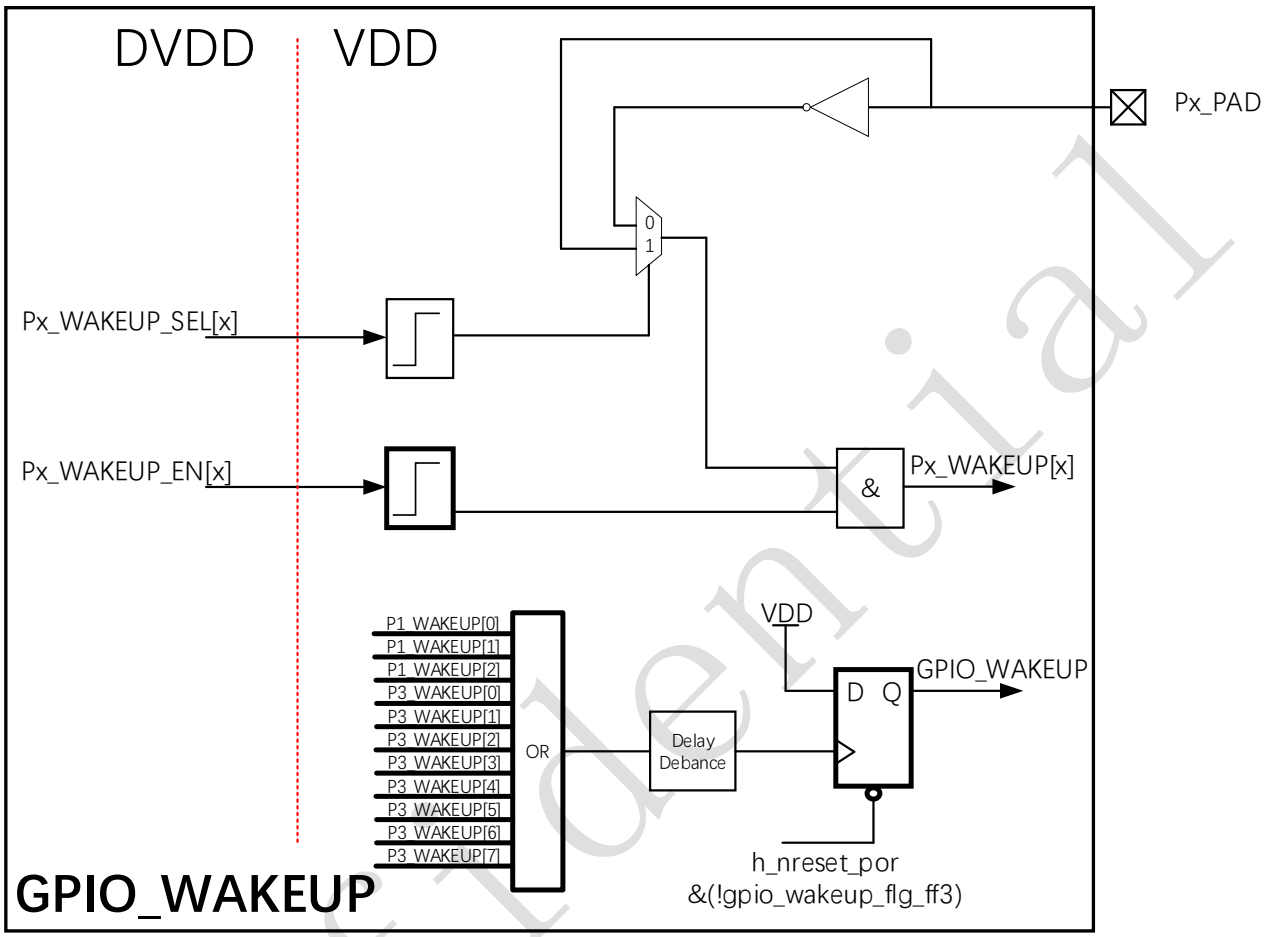
睡眠 Sleep 模式下低压区 DVDD 区域由强 LDO HPLDO 供电，高压区 VDD 区域正常供电，系统时钟保持进入睡眠前的状态。CPU 时钟停止，其他外设时钟正常运行。为了降低功耗，可以在进入睡眠模式之前关闭外设时钟。

进入方式：软件向寄存器 PCON 的 BIT0(IDLE)写 1。

唤醒条件：任意中断源均可唤醒。

5.2 唤醒电路

5.2.1 GPIO 唤醒



说明：

1， GPIO 唤醒可选择高电平唤醒，也可以选择低电平唤醒；

2， 模拟 PAD 分两路给到数字，一路 VDD 电压域用于唤醒，一路 DVDD 电压域用于状态锁存；

3， 低电平唤醒时，GPIO 输入经过取反，内部变为高；Px_WAKEUP[x]信号‘或’之后，经过模拟的 DelayDebance 模块，通过 D 触发器锁存把低转高锁存为高电平；

4， 如果外部 GPIO 唤醒电平时间过短，一种情况是被 DelayDebance 滤掉。如果没有滤掉，经过 DFF 锁存，产生 GPIO_WAKEUP 唤醒信号，唤醒系统。软件查看被使能的 GPIO 状态，如果不是唤醒电平，软件再次触发进入 DeepSleep 模式。

5.2.2 WakeupCounter 唤醒

说明：

1，Wakeup Counter 使用 RippleCounter，定时周期如下：

定时器 BIT	15K 定时周期（us）
0	62.5
1	125
2	250
3	500
4	1000
5	2000
6	4000
7	8000
8	16000
9	32000
10	64000
11	128000
12	256000
13	512000
14	1024000
15	2048000
16	4096000
17	8192000
18	16384000

2，最长定时周期 16.384s，如果需要更长的定时，可以在唤醒后由软件计时。这样做的好处是降低硬件面积和功耗，换算平均功耗，不会使系统功耗有很大的增加。

3，有两种方式启用 WakeupCounter，一种是软件触发，将 lh_wakeup_cnt_trg 置 1，该 bit 自动清零。这种方法触发后 WakeupCounter 立即计数。另一种是将 lh_wakeup_cnt_sleep_en 置 1，系统进入 DeepSleep 后自动开始计数。

4，两种启用方式都需要软件先设置 lh_wakeup_cnt_sel，再配置对应的 lh_wakeup_cnt_trg 或者 lh_wakeup_cnt_sleep_en。目的是为了保证 lh_wakeup_cnt_sel 能够从 DVDD 电压域同步到 VDD 电压域。

5，Wakeup count 唤醒只适用于系统时钟为 RCH 的情况，并且在使用 wakeup counter 唤醒的时候寄存器 rch_dly_sel 必须配置为 0x01 或 0x02.

5.3 低功耗流程

5.3.1 深度睡眠进入和退出流程

进入流程：

1，使能用于唤醒的中断，中断 0（WakeupCounter）、中断 2（GPIO）；

2，设置 WAKEUP_INT_CTRL 的 INT0_SEL 或者 INT1_SEL 为 1，INT0 对应 Wakeup Counter 唤醒，INT1 对应 GPIO 唤醒；

3，使用 WakeupCounter 唤醒请设置 WAKEUP_CNT_CTRL 的 Wakeup_cnt_sleep_en 为 1，并设置 Wakeup_cnt_sel 选择唤醒时间；

4，使用 GPIO 唤醒先选择对应的唤醒方式 P1_WAKEUP_SEL 和 P3_WAKEUP_SEL，再使能对应的 GPIO 的输入模式，再打开对应的 GPIO 唤醒使能 P1_WAKEUP_EN 和 P3_WAKEUP_EN（顺序不能变）；

5，关闭 XTH 和 DPLL，关闭 RF 相关的 LDO。如果系统时钟使用 XTH，先切换为 RCH 再关闭 XTH（可以不开放 XTH 作为系统时钟的选项）；

6，PCON 的 bit 1 写 1。

退出流程：

1，系统唤醒后，进入相应的中断服务程序；

2，INT0 中断为 WakeupCounter 唤醒，软件读取 WAKEUP_CNT_CTRL 的 BIT2，判断是从 DEEPSLEEP 唤醒还是 SLEEP 唤醒或者是普通中断，清除中断标志位；

3，INT1 中断为 GPIO 唤醒，软件读取 WAKEUP_CNT_CTRL 的 BIT2 判断是从 DEEP SLEEP 唤醒还是 SLEEP 唤醒或者普通中断，软件根据设置的 GPIO 唤醒使能（P1/3_WAKEUP_EN）查询管脚的状态，判断具体的唤醒 GPIO。

5.3.2 睡眠模式进入和退出

进入流程：

1，配置需要唤醒的中断，包括普通中断和 GPIO 唤醒，WakeupCounter 唤醒；

2，关闭睡眠中不会使用的模块的时钟，使能中断；

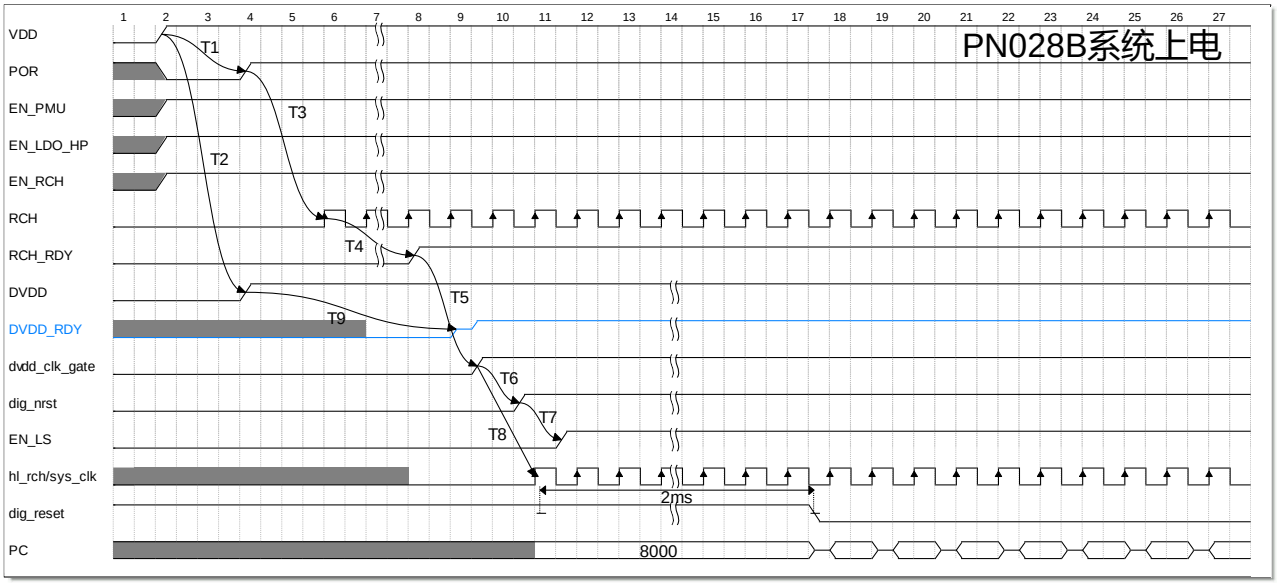
3，PCON 的 BIT0 写 1。

退出流程：

1，系统唤醒后进入相应的中断服务程序；

2，处理相应中断程序，然后退出。

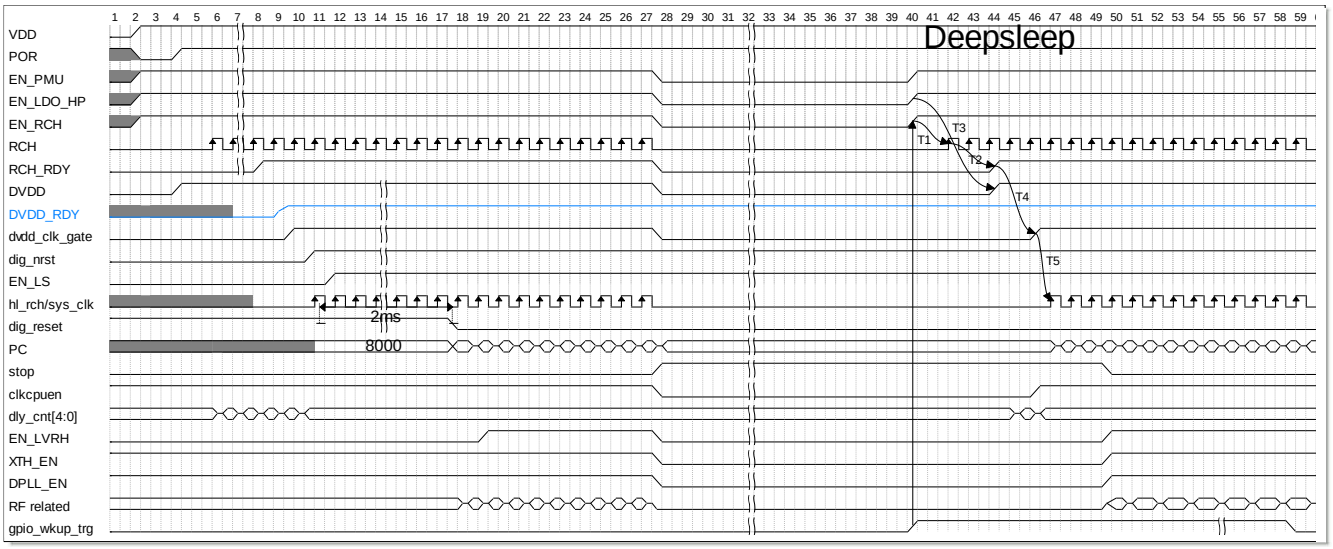
5.4 系统上电及低功耗进入退出时序



PAN262x 上电时序图

上电过程说明：

1. VDD 拉高之前，POR、EN_PMU、EN_LDO_HP、EN_RCH 的状态是不确定的；
2. 从 028 测试情况来看，T1、T2 哪个长哪个短不确定。如果 VDD 上电时间较长或者外挂大电容的时候，T1 可能会小于 T2，这样是有问题的，可能会导致低压区 POR 复位失败；
3. T3 大概 1us 左右，由模拟电路决定；
4. T4 大概 4us 左右，由模拟电路决定；
5. T5 大概 1.8ms 左右，由 RCH_3V 来计数，不可配置；
6. T6 大概 3 个 RCH 时钟周期，大概 180ns 左右；
7. T7 大概 3 个 RCH 时钟周期，大概 180ns 左右；
8. T8 大概 1 个 RCH 时钟周期，大概 60ns 左右；
9. T9 由模拟电路决定；
10. 从 sys_clk 有效到 MCU 开始运行有 2ms 的等待时间，由 sys_clk/8M 计数，做死的。该 2ms 是留给 C2 的窗口期。



deepsleep 进入以及 IO 唤醒时序图

说明：

1. MCU 执行寄存器写操作 PCON = 0x0A 之后进入 deepsleep 模式，上图为 stop 信号拉高；
2. Deepsleep 模式应该关掉的模块有：强 LDO、BandGap、RCH、XTH、DPLL、LVRH、RF 相关的模块；
3. DVDD_RDY 信号在 deepsleep 阶段一直为高，与其命名不符；
4. 执行 PCON = 0x0A 之后，相关模块在 600ns 之内全部关闭；
5. IO 唤醒的时候，相关 IO 会触发 gpio_wkup_trg 信号，该信号将 EN_PMU、EN_LDO_HP、EN_RCH 立马拉高；
6. T1 大概为 1~2us，之后 RCH 有输出；
7. T2 为 4~5us，之后 RCH_RDY 拉高；
8. T3 最大 10us，之后 DVDD 才真正 ready；
9. T4 时间寄存器可配，默认为 1F 约等于 1.8ms（用于上电），之后可配小（用于快速唤醒），该值 0~1.8ms 可配，step 60us；
10. T5 为 5 个时钟周期，大概 300ns；
11. MCU 在之前的基础上继续运行，那些被 deepsleep 关掉的模块在 stop 拉低之后会再次打开，相当于做了个 AND。

DeepSleep 进入退出说明：

1，软件触发进入 DEEPSLEEP 模式后，sleep_status 拉高。sleep_status 拉高后，表明 DVDD 电压域数字电路时钟可以关闭，直接拉低 dvdd_clk_gate 信号。由于使用 ICG，输出给 DVDD 电压域的时钟不会有毛刺产生。

2，DVDD 电压域时钟关闭后，关闭 HPLDO。（这之间需要多长时间间隔需要确认。VDD 电压域给到 DVDD 电压域的时钟，由于存在 ICG 和 Level Shift，会有比较大的延时。要保证时钟彻底关闭后再关电源）。

3，HPLDO 关闭后再关闭 BandGap 和 RCH。

4，Gpio_wakeup 和 wakeup_cnt_clr 信号唤醒系统后，sleep_status 拉低。具体过程和上电类似，不同之处在于 EN_LS 和给到 DVDD 电压域的 dig_nrst 一直为高。

5，进入低功耗模式前需要保证 Flash 进入 DeepStandby 模式，SRAM 的 CEN 要拉高。

5.5 寄存器

5.5.1 寄存器映射

Register	Offset	R/W	Description	Reset Value
ANA_SEL	0xFA	R/W	ANA 寄存器地址选择	0x00
ANA_DAT	0xFB	R/W	ANA 寄存器数据读写寄存器	0x00

电源管理寄存器列表：

Register	ANA_SEL	R/W	Description	Reset Value
P1_WAKEUP_EN	0x00	R/W	P1 唤醒使能控制	0x00
P3_WAKEUP_EN	0x01	R/W	P3 唤醒使能控制	0x00
P1_WAKEUP_SEL	0x02	R/W	P1 唤醒方式选择	0x00
P3_WAKEUP_SEL	0x03	R/W	P3 唤醒方式选择	0x00
WAKEUP_CNT_CTRL	0x04	RS/W	Wakeup counter 唤醒控制	0x00
WAKEUP_INT_CTRL	0x05	RS/W	GPIO、Wakeup Counter 唤醒中断控制及状态	0x00
ANA_PWR_CTRL0	0x10	R/W	电源相关控制，L2H 给到高压区再同步、逻辑给到模拟	0x4F
ANA_PWR_CTRL1	0x11	R/W	电源相关控制，直接给到模拟	0x90
ANA_PWR_CTRL2	0x12	R/W	电源相关控制，直接给到模拟	0x28
RCH_DLY_CTRL	0x15	R/W	控制 DVDD 时钟域时钟启动时间	0x1F
RCH_SYNC_TRG	0x16	RS/W	用于高压区控制信号同步	0xF0

5.5.2 寄存器描述

5.5.2.1 P1_WAKEUP_EN

Register	ANA_SEL	R/W	Description	Reset Value
P1_WAKEUP_EN	0x00	R/W	P1 唤醒使能控制	0x00

Bits	Description	
[7]	Reserved	
[6]	P16_wk_en	P16 唤醒使能，1：使能，0：关闭
[5]	P15_wk_en	P15 唤醒使能，1：使能，0：关闭
[4]	P14_wk_en	P14 唤醒使能，1：使能，0：关闭
[3]	P13_wk_en	P13 唤醒使能，1：使能，0：关闭
[2]	P12_wk_en	P12 唤醒使能，1：使能，0：关闭
[1]	P11_wk_en	P11 唤醒使能，1：使能，0：关闭
[0]	P10_wk_en	P10 唤醒使能，1：使能，0：关闭

5.5.2.2 P3_WAKEUP_EN

Register	ANA_SEL	R/W	Description	Reset Value
P3_WAKEUP_EN	0x01	R/W	P3 唤醒使能控制	0x00

Bits	Description	
[7]	P37_wk_en	P37 唤醒使能，1：使能，0：关闭
[6]	P36_wk_en	P36 唤醒使能，1：使能，0：关闭
[5]	P35_wk_en	P35 唤醒使能，1：使能，0：关闭
[4]	P34_wk_en	P34 唤醒使能，1：使能，0：关闭
[3]	P33_wk_en	P33 唤醒使能，1：使能，0：关闭
[2]	P32_wk_en	P32 唤醒使能，1：使能，0：关闭
[1]	P31_wk_en	P31 唤醒使能，1：使能，0：关闭
[0]	P30_wk_en	P30 唤醒使能，1：使能，0：关闭

5.5.2.3 P1_WAKEUP_SEL

Register	ANA_SEL	R/W	Description	Reset Value
P1_WAKEUP_SEL	0x02	R/W	P1 唤醒方式选择	0x00

Bits	Description	
[7]	Reserved	
[6]	P16_wk_sel	P16 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[5]	P15_wk_sel	P15 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[4]	P14_wk_sel	P14 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[3]	P13_wk_sel	P13 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[2]	P12_wk_sel	P12 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[1]	P11_wk_sel	P11 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[0]	P10_wk_sel	P10 唤醒方式选择，1：低电平唤醒；0：高电平唤醒

5.5.2.4 P3_WAKEUP_SEL

Register	ANA_SEL	R/W	Description	Reset Value
P3_WAKEUP_SEL	0x03	R/W	P3 唤醒方式选择	0x00

Bits	Description	
[7]	P37_wk_sel	P37 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[6]	P36_wk_sel	P36 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[5]	P35_wk_sel	P35 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[4]	P34_wk_sel	P34 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[3]	P33_wk_sel	P33 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[2]	P32_wk_sel	P32 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[1]	P31_wk_sel	P31 唤醒方式选择，1：低电平唤醒；0：高电平唤醒
[0]	P30_wk_sel	P30 唤醒方式选择，1：低电平唤醒；0：高电平唤醒

5.5.2.5 WAKEUP_CNT_CTRL

Register	ANA_SEL	R/W	Description	Reset Value
WAKEUP_CNT_CTRL	0x4	RS/W	Wakeup counter 唤醒控制	0x00

Bits	Description	
[7:6]	Reserved	
[5]	Wakeup_cnt_trg	软件控制 wakeup counter 使能，设为 1 后，wakeup counter 开始计时，该 BIT 自动清零
[4]	Wakeup_cnt_sleep_en	为 1 时，CPU 发起进入 DeepSleep 后，wakeup counter 自动计时； 为 0 时，wakeup counter 由软件控制
[3:0]	Wakeup_cnt_sel	Wakeup_cnt 唤醒时间选择： 时间为 2 的(N+3)次方乘以 62.5uS。N 为 wakeup_cnt_sel 的值，范围从 0.5ms~16s

5.5.2.6 WAKEUP_INT_CTRL

Register	ANA_SEL	R/W	Description	Reset Value
WAKEUP_INT_CTRL	0x5	RS/W	GPIO、Wakeup Counter 唤醒中断控制及状态	0x00

Bits	Description	
[7:5]	Reserved	
[4]	Gpio_wakeup_flg	GPIO 唤醒中断标志位，写 1 清零
[3]	Wakeup_cnt_flg	WakeupCounter 中断标志位，写 1 清零
[2]	Int_status	0：中断之前，MCU 处于 Sleep 或者正常工作模式 1：中断之前，MCU 处于 DeepSleep 模式 说明：硬件进入 DeepSleep 前自动置 1，需软件写 1 清零
[1]	Int1_sel	0：INT1 来源于外部中断 EXINT1 1：INT1 来源于 gpio 唤醒中断
[0]	Int0_sel	0：INT0 来源于外部中断 EXINT0 1：INT0 来源于 wakeup counter 中断

5.5.2.7 ANA_PWR_CTRL0

Register	ANA_SEL	R/W	Description	Reset Value	推荐值
ANA_PWR_CTRL0	0x10	R/W	电源相关控制，L2H 给到高压区再同步、逻辑给到模拟	0x4F	默认值

Bits	Description	
[7]	Reserved	
[6:4]	LP_LDO_TRIM	lp ldo 输出 TRIM，默认 100
[3]	HP_LDO_MODE	强 LDO 模式选择：1，PUMP 模式；0，NORMAL 模式。默认 1
[2]	Reserved	
[1]	EN_LDO_HP	数字强 LDO 使能控制：1，打开；0，关掉。默认 1
[0]	EN_PMU	电源管理 BandGap 使能控制：1，打开；0，关掉。默认 1

5.5.2.8 ANA_PWR_CTRL1

Register	ANA_SEL	R/W	Description	Reset Value	推荐值
ANA_PWR_CTRL1	0x11	R/W	电源相关控制	0x90	默认值

Bits	Description	
[7:5]	LP_BIAS_TRIM	LP BIAS 电流校准，默认 100
[4:0]	VBG_TRIM	VBG 输出 TRIM，默认 10000

5.5.2.9 ANA_PWR_CTRL2

Register	ANA_SEL	R/W	Description	Reset Value	推荐值
ANA_PWR_CTRL2	0x12	R/W	电源相关控制	0x28	默认值

Bits	Description	
[7:6]	Reserved	
[5:4]	HP_LDO_CAPSEL	hp ldo 电容选择，默认 10
[3:0]	HP_LDO_TRIM	hp ldo 输出 TRIM，默认 1000

5.5.2.10 RCH_DLY_CTRL

Register	ANA_SEL	R/W	Description	Reset Value	推荐值
RCH_DLY_CTRL	0x15	R/W	用于控制 DVDD 时钟域时钟启动时间	0x1F	默认值

Bits	Description	
[7:5]	Reserved	
[4:0]	RCH_DLY_SEL	用于选择高压区 RCH delay 时间，单位是 RCH/1024 约为 64us 的周期 如果使用 wakeup counter 唤醒，进入 deepsleep 之前必须配置为 0x01/0x02

5.5.2.11 RCH_SYNC_TRG

Register	ANA_SEL	R/W	Description	Reset Value	推荐值
RCH_SYNC_TRG	0x16	RS/W	用于同步 RCH_DLY_SEL 的值到 VDD 电压域	0xF0	默认值

Bits	Description	
[7]	FLASH_SIZE	Root 模式 FLASH 大小配置，1：32K，0：16K，默认 1
[6]	USB_EN	Root 模式 USB 使能配置，1：使能，0：不使能，默认 1
[5]	ADC_EN	Root 模式 ADC 使能配置，1：使能，0：不使能，默认 1
[4]	P36_NRST_EN	P36 复用为复位脚，1：使能，0：不使能，默认 1，如果要将 P36 用作 I2C 等功能，需要先把该 bit 置 0
[3:1]	Reserved	
[0]	RCH_SYNC_TRG	用于同步下列 12 个寄存器的值到 VDD 电压域，下列寄存器配置后不能立即生效，必须对 rch_sync_trg 写 1 触发一下，该 bit 自动清零 rcl_test_en rcl_cal_h_sel[2:0] rcl_cal_per_sel[2:0] en_rch en_pmu en_ldo_hp en_ls（非寄存器控制，硬件自动控制） en_rcl(CLK_TOP_CTRL.bit0) en_lvrh(LVRH_CTRL.bit1) hp_ldo_mode(ANA_PWR_CTRL0.bit3) lp_ldo_trim[2:0](ANA_PWR_CTRL0.bit[6:4])

第6章 Flash控制器

PAN262x 包含一个用户可编程的 flash，可重新编程闪存用于程序代码和非易失性数据存储。flash 可以通过 C2 接口或使用 FMC 指令的软件在系统中编程，每次一个字节。一旦清除为逻辑 0，必须擦除闪存位以将其设置回逻辑 1。闪存字节通常在重新编程之前被擦除（设置为 0xFF）。写入和擦除操作由硬件自动计时，以便正确执行；数据轮询以确定不需要写入/擦除操作的结束。代码执行在闪存写入/擦除操作期间暂停。

6.1 特征

- 支持CRC32校验
- 支持代码保护机制
- 支持ISP在线编程
- 支持sector擦除
- 支持chip擦除

6.2 模块框图

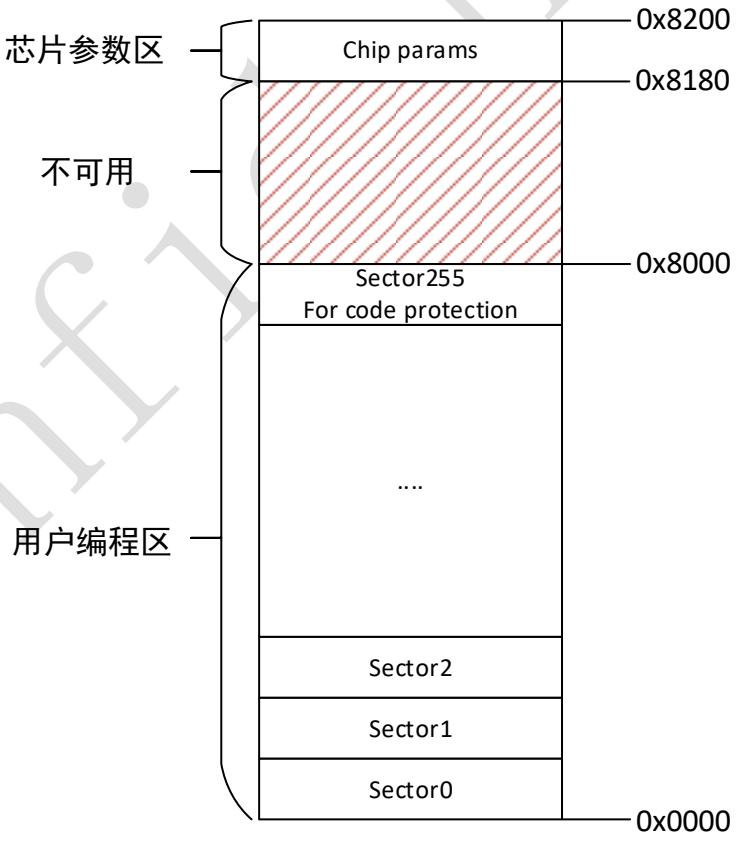


图 6-1　32K flash 地址划分

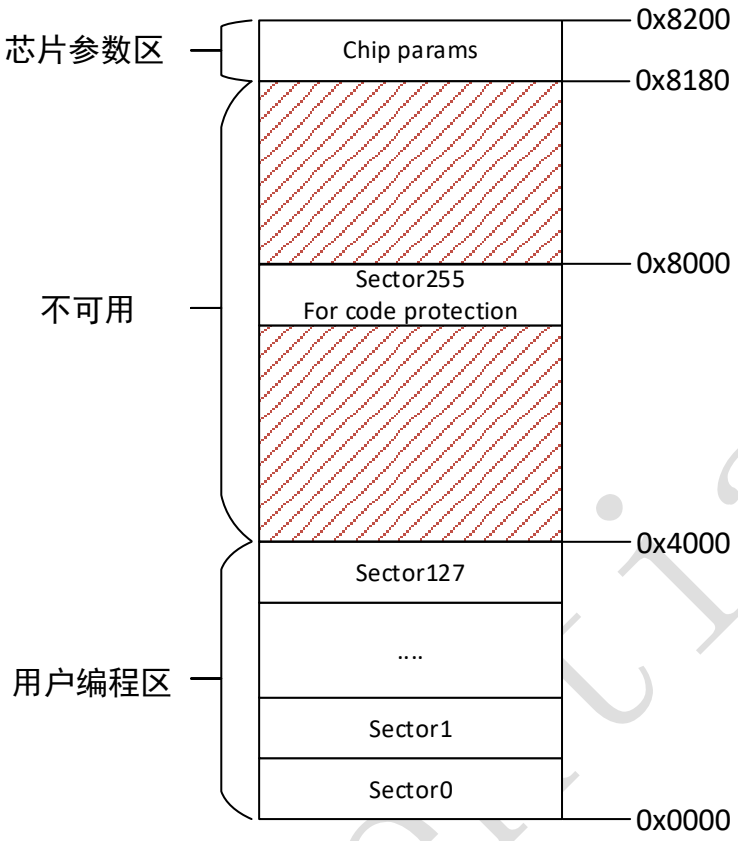


图 6-2　16K flash 地址划分

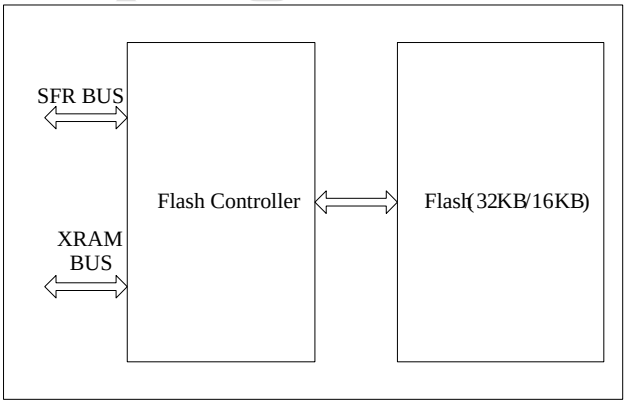


图 6-3　FMC 模块框图

6.3 功能描述

6.3.1 CRC32 校验

FMC 支持硬件 CRC32 功能，软件需要配置起始地址与大小，硬件将自动计算该段 code 的 CRC32，并反馈给寄存器中。具体流程参考 6.6.5 小节。

其中，CRC32 算法多项式为 $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$。

硬件每次读取 4 个 Byte，将数据进行反序异或，LSB first，按照上述的多项式进行计算。

6.3.2 代码保护机制

FMC 支持代码保护机制，根据烧录在 main 区域的最后一个 sector（sector 255）中的最后一个 byte 来决定是否开启保护。除去 0xFF 外，其他数值均代表开启保护模式。

注意 16KB 模式也是由 sector 255 的最后一个 byte 进行保护。

6.4 寄存器列表

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	0xE8	R/W	FMC ISP 控制寄存器	0x00
FMC_ISPADDR0	0xE9	R/W	FMC ISP 地址寄存器 0	0x00
FMC_ISPADDR1	0xEA	R/W	FMC ISP 地址寄存器 1	0xd9
FMC_ISPDATA	0xEB	R/W	FMC ISP 数据寄存器	0x03
FMC_ISPCMD	0xEC	R/W	FMC ISP 命令寄存器	0x00
FMC_ISPTRG	0xED	R/W	FMC ISP 触发寄存器	0x00
FMC_DLYCRL2	0x09(0x1000)	R/W	FMC ISP 时序寄存器	0x00
FMC_LPCNT	0x0E(0x1000)	R/W	FMC ISP 低功耗唤醒寄存器	0x2a
FMC_DOWN-LOAD_KEY1	0x1A(0x1000)	R/W	FMC ISP 下载寄存器 1	0x55
FMC_DOWN-LOAD_KEY2	0x1B(0x1000)	R/W	FMC ISP 下载寄存器 2	0xaa
FMC_DOWN-LOAD_KEY3	0x1C(0x1000)	R/W	FMC ISP 下载寄存器 3	0x5a
FMC_JUMP_CNT	0x1D(0x1000)	R/W	FMC 跳转指令周期寄存器	0x02
FMC_REG_BYTE	0x1E(0x1000)	R/W	FMC 保护寄存器	0x00
FMC_CRC_SIZE0	0x20(0x1000)	R/W	FMC CRC SIZE 配置寄存器 0	0x00
FMC_CRC_SIZE1	0x21(0x1000)	R/W	FMC CRC SIZE 配置寄存器 1	0x00
FMC_CRC_DATA0	0x22(0x1000)	R/W	FMC CRC 数据寄存器 0	0x00
FMC_CRC_DATA1	0x23(0x1000)	R/W	FMC CRC 数据寄存器 1	0x00
FMC_CRC_DATA2	0x24(0x1000)	R/W	FMC CRC 数据寄存器 2	0x00
FMC_CRC_DATA3	0x25(0x1000)	R/W	FMC CRC 数据寄存器 3	0x00

6.5 寄存器描述

6.5.1 FMC_ISPCTL

Register	**Offset**		**R/W**	**Description**	**Reset Value**
FMC_ISPCTL	0xE8 (SFR)		R/W	ISP Control Register	0x0000_000X
Bits	**Descriptions**				
[7:1]	Reserved	Reserved.			
[0]	ISPEN	ISP Enable Bit (Write Protect) Set this bit to enable the ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.			

6.5.2 FMC_ISPADDR0

Register	Offset	R/W	Description	Reset Value
FMC_ISPADDR0	0xE9(SFR)	R/W	ISP Address Register0	0x0000_0000
Bits	**Descriptions**			
[7:0]	ISPADDR0	FLASH_ADDR[7:0]		

6.5.3 FMC_ISPADDR1

Register	Offset	R/W	Description	Reset Value
FMC_ISPADDR1	0xEA(SFR)	R/W	ISP Address Register1	0x0000_0000
Bits	**Descriptions**			
[7:0]	ISPADDR1	FLASH_ADDR[15:8] 32K flash 使用[6:0],16K flash 使用[5:0]		

6.5.4 FMC_ISPDATA

Register	Offset	R/W	Description	Reset Value
FMC_ISPDAT	0xEB(SFR)	R/W	ISP DataRegister	0x0000_0000
Bits	**Descriptions**			
[7:0]	ISPDAT	ISP Data Write data to this register before ISP program operation. Read data from this register after ISP read operation.		

6.5.5 FMC_ISPCMD

Register	Offset	R/W	Description	Reset Value
FMC_ISPCMD	0xEC(SFR)	R/W	ISP Command Register	0x0000_0000
Bits	**Descriptions**			
[7]	Reserved	Reserved		
[6:0]	CMD	ISP CMD ISP command table is shown below: 0x00 = FLASH Read. 0x21 = FLASH 8-bit Program. 0x22 = FLASH Page Erase. 0x23 = FLASH whole chip Erase (ROM mode). The other commands are invalid.		

6.5.6 FMC_ISPTRG

Register	Offset	R/W	Description	Reset Value
FMC_ISPTRG	0xED(SFR)	R/W	ISP Trigger Control Register	0x0000_0000

Bits	Descriptions	
[7:1]	Reserved	Reserved
[0]	ISPGO	ISP Start Trigger (Write Protect) Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation has finished. 0 = ISP operation has finished. 1 = ISP is progressed.

6.5.7 FMC_DLYCRL2

Register	Offset	R/W	Description	Reset Value
FMC_DLYCRL1	0x09(xdata fmc base 0x1000)	R/W	Erase / Write Timing Control 1	0x00
Bits	**Descriptions**			
[7:2]	Reserved	Reserved		
[1:0]	cnt_lp_ctl	2'b00：cnt_lp 配置为 8Mhz 时序， 2'b01：cnt_lp 配置为 16Mhz 时序 2'b11: 自定义参数， other :8M 时序		

6.5.8 FMC_LPCNT

Register	Offset	R/W	Description	Reset Value
FMC_LPCNT	0x0E(xdata fmc base 0x1000)	R/W	LP timing Control Register	0x2a
Bits	**Descriptions**			
[7:0]	cnt_lp_timing_set	自定义设置唤醒时间参数 注 ：根据当前系统时钟，推算最小 cnt 值 ，不能小于 2us cnt 时间，如 8M 时钟下，2000/125=16，设置要大于 0x10。		

6.5.9 FMC_DOWNLOAD_KEY1

Register	Offset	R/W	Description	Reset Value
FMC_DOWN- LOAD_KEY1	0x1A(xdata fmc base 0x1000)	W	DOWNLOAD KEY 1 register	0x55
Bits	**Descriptions**			
[7:0]	Download_key1	上电复位后默认 0x55，在第二段 reset 程序跳转 main 区域之前被配置 0x00，**激活值 0x55**		

6.5.10 FMC_DOWNLOAD_KEY2

Register	Offset	R/W	Description	Reset Value
FMC_DOWN-LOAD_KEY2	0x1B(xdata fmc base 0x1000)	W	DOWNLOAD KEY 2 register	0xAA

Bits	Descriptions	
[7:0]	Download_key2	上电复位后默认 0xAA，在第二段 reset 程序跳转 main 区域之前,Download_key1 被配置 0x00 后，再清零 Download_key2 除清零之外的写入操作，改变 Download_key2 需要 Download_key1 为 0x55,

6.5.11 FMC_DOWNLOAD_KEY3

Register	Offset	R/W	Description	Reset Value
FMC_DOWN-LOAD_KEY3	0x1C(xdata fmc base 0x1000)	W/R	DOWNLOAD KEY 3 register	0x5A

Bits	Descriptions	
[7:0]	Download_key3	W: 上电复位后默认 0x5A，在第二段 reset 程序跳转 main 区域之前,Download_key1&Download_key2 依次被配置 0x00 后，再清零 Download_key3 除清零之外的写入操作，改变 Download_key3 需要 Download_key1 为 0x55 且 Download_key2 为 0xAA。 R : 1: info 程序未执行（或被中断） 0：info 程序运行结束 flag

6.5.12 FMC_JUMP_CNT

Register	Offset	R/W	Description	Reset Value
FMC_JUMP_CNT	0x1D(xdata fmc base 0x1000)	W/R	Jump instruction cycle register	0x02

Bits	Descriptions	
[7:2]	Reserved	Reserved
[1:0]	Jump instruction cycle	跳转指令占用 4 个周期，配置 2’b11 跳转指令占用 3 个周期，配置 2’b10 跳转指令占用 2 个周期，配置 2’b01 跳转指令占用 1 个周期，配置 2’b00

6.5.13 FMC_REG_BYTE

Register	Offset	R/W	Description	Reset Value

FMC_REG_BYTE	0x1E(xdata fmc base 0x1000)	W/R	Security byte register	0x0

Bits	Descriptions	
[7:0]	Reg_byte	更新保护 byte 寄存器时间： 1）每次上电 INFO 程序软件 load 更新 2）在烧录模式下，软件主动更新（为了防客户误触引入 prot_wr_byte 寄存器，硬件保证不可写入 FF） 3）chip erase 完成对此寄存器写入 0xff 关闭保护 R: 1: 保护状态 ，0：非保护状态

6.5.14 FMC_CRC_SIZE0

Register	Offset	R/W	Description	Reset Value
FMC_CRC_SIZE0	0x20(xdata fmc base 0x1000)	W/R	CRC SIZE register	0x0

Bits	Descriptions	
[7:0]	W: crc_size_low	crc size 低八位，单位是 byte

6.5.15 FMC_CRC_SIZE1

Register	Offset	R/W	Description	Reset Value
FMC_CRC_SIZE1	0x21(xdata fmc base 0x1000)	W/R	CRC SIZE register	0x0

Bits	Descriptions	
[7:0]	W: crc_size_high	crc size 高八位，单位是 byte

6.5.16 FMC_CRC_DATA0

Register	Offset	R/W	Description	Reset Value
FMC_CRC_DATA0	0x22(xdata fmc base 0x1000)	W/R	CRC DATA register	0x0

Bits	Descriptions	
[7:0]	W: crc_data	{crc_data3,crc_data2,crc_data1,crc_data0}

6.5.17 FMC_CRC_DATA1

Register	Offset	R/W	Description	Reset Value

FMC_CRC_DATA1	0x23(xdata fmc base 0x1000)	W/R	CRC DATA register	0x0

Bits	Descriptions
[7:0]	W: crc_data {crc_data3,crc_data2,crc_data1,crc_data0}

6.5.18 FMC_CRC_DATA2

Register	Offset	R/W	Description	Reset Value
FMC_CRC_DATA2	0x24(xdata fmc base 0x1000)	W/R	CRC DATA register	0x0

Bits	Descriptions
[7:0]	W: crc_data {crc_data3,crc_data2,crc_data1,crc_data0}

6.5.19 FMC_CRC_DATA3

Register	Offset	R/W	Description	Reset Value
FMC_CRC_DATA3	0x25(xdata fmc base 0x1000)	W/R	CRC DATA register	0x0

Bits	Descriptions
[7:0]	W: crc_data {crc_data3,crc_data2,crc_data1,crc_data0}

6.6 软件使用流程

6.6.1 flash 读操作

单位：byte

方法一 通过 SFR 寄存器配置

1）配置 ISP 操作使能，FMC_ISPCTL= 0x1;//ispctl

2）配置 ISP 操作命令，FMC_ISPCMD = 0x0;//cmd

3）配置 ISP 操作 flash 低 8 位地址，FMC_ISPADDR0 = flash 低 8 位地址;

4）配置 ISP 操作 flash 高 8 位地址，FMC_ISPADDR1 = flash 高 7 位地址;

5）配置 ISP 操作 trg 信号开始读操作，FMC_ISPTRG = 0x01;

6）等待 FMC_ISPTRG trg 信号拉低，去读 FMC_ISPDAT 寄存器

方法二 通过 MOVX ACC, @DPTR 访问 flash

6.6.2 flash 写操作

单位：byte

方法一 通过 SFR 寄存器配置

1）配置 ISP 操作使能，FMC_ISPCTL= 0x1;//ispctl

2）配置 ISP 操作命令，FMC_ISPCMD = 0x21;//cmd

3）配置 ISP 操作的 flash 低 8 位地址，FMC_ISPADDR0 = flash 低 8 位地址;

4）配置 ISP 操作的 flash 高 8 位地址，FMC_ISPADDR1 = flash 高 7 位地址;

5）将待写数据配置到 FMC_ISPDATA, FMC_ISPDATA = 待写入数据

5）配置 ISP 操作 trg 信号开始写操作，FMC_ISPTRG=0x01;

6）等待 FMC_ISPTRG trg 信号拉低，才可以下一次操作

方法二 通过 MOVX@DPTR,A 烧录

6.6.3 flash 页擦操作

单位：128 byte

寄存器配置

1）配置 ISP 操作使能，FMC_ISPCTL= 0x1

2）配置 ISP 操作命令，FMC_ISPCMD = 0x22

3）配置 ISP 操作的 flash 低 8 位地址，FMC_ISPADDR0 = flash 低 8 位地址;

4）配置 ISP 操作的 flash 高 8 位地址，FMC_ISPADDR1 = flash 高 7 位地址;

5）配置 ISP 操作 trg 信号开始页擦除操作，FMC_ISPTRG=0x01;

6）等待 FMC_ISPTRG trg 信号拉低，才可以下一次操作

6.6.4 flash 片擦操作

单位：main 区域（32KB/16KB）

寄存器配置

1）配置 ISP 操作使能，FMC_ISPCTL = 0x1

2）配置 ISP 操作命令，FMC_ISPCMD = 0x23

3）配置 ISP 操作 FMC_ISPTRG 信号开始片擦除操作，FMC_ISPTRG=0x01;

注：

1）flash 写操作某个区域之前需要先擦除操作

2）擦除操作时注意不要把自身程序擦除，否则程序跑飞

6.6.5 flash crc32 校验操作

1）配置 fmc 使能，FMC_ISPCTL = 0x1

2）配置 crc32 命令，FMC_ISPCMD = 0x24

3）配置 crc32 flash 地址，FMC_ISPADD0/ADD1

4）配置 crc32 flash size，FMC_ISPSIZE0/SIZE1

5）配置 fmc trg 信号开始片擦除操作，FMC_ISPTRG=0x01;

6）硬件 crc32 期间 FMC_ISPTRG 一直为 0x01，转换完成为 0x0

第7章 GPIO

7.1 IO 概述

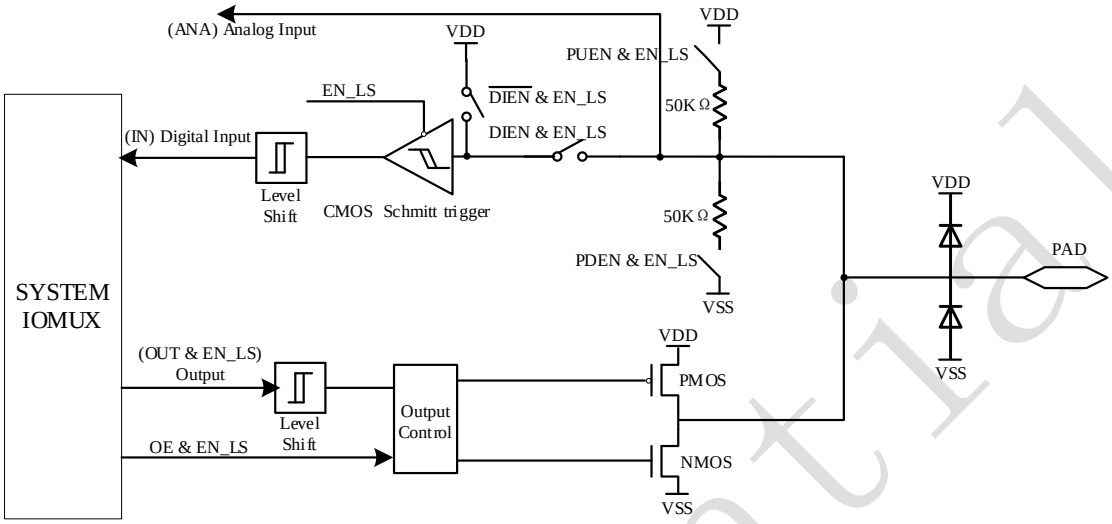


图7.1 GPIO电路结构图

PAN262x提供最多15个双向I/O端口，分成2组P1和P3。所有I/O引脚可以被软件独立配置成六种I/O模式中的一种。这六种模式是模拟输入、数字输入、上拉输入、下拉输入、推挽输出、开漏输出。

PAN262x上电复位后，C2CK（P3.2）、C2DAT（P3.4）端口为上拉输入状态，其他端口为模拟输入状态。

P3.6 默认为复位脚，其端口是上拉输入状态，该引脚可复用为 GPIO。下表中的 OUT 与寄存器 p1.x/p3.x 相对应，由软件控制。X 代表不关心当前值。

GPIO 模式	DIEN	OE	PUEN	PDEN	IN	OUT	PAD
模拟输入	0	0	0	0	0	X	PAD
数字输入	1	0	0	0	PAD	X	PAD
上拉输入	1	0	1	0	PAD	X	PAD
下拉输入	1	0	0	1	PAD	X	PAD
推挽输出	0	1	0	0	0	OUT	OUT
开漏输出	1	OUT	0	0	0	0	OUT

表 7.1 GPIO 模式真值表

7.1.1 模拟输入模式

模拟输入模式提供真实的高阻输入路径。模拟输入模式的好处是减少在逻辑 0 时电流的消耗。用户需要注意的是，模拟输入模式应该由外部设备或电阻提供一个确定的电平。悬浮的引脚在掉电状态下会引起漏电。

配置方式：Px_MFPx=000、Px_MODEx=00、Px_DIENx=0、Px_PUENx=0、Px_PDENx=0

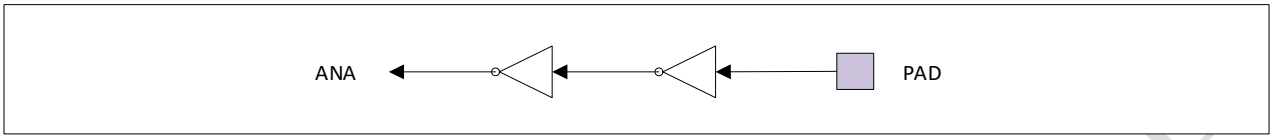


图 7.2 模拟输入模式结构图

7.1.2 数字输入模式

数字输入模式下端口 IN 直接由外部设备提供一个确定的高或低电平，否则会引起漏电。

配置方式：Px_MFPx=000、Px_MODEx=00、Px_DIENx=1、Px_PUENx=0、Px_PDENx=0

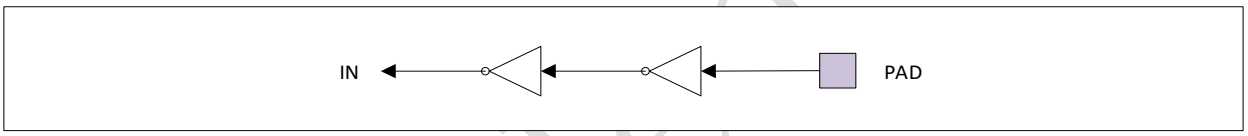


图 7.3 数字输入模式结构图

7.1.3 上拉输入模式

上拉输入模式下端口 IN 默认为高电平，外部设备或电阻不提供确定的高低电平也不会漏电。

配置方式：Px_MFPx=000、Px_MODEx=00、Px_DIENx=1、Px_PUENx=1、Px_PDENx=0

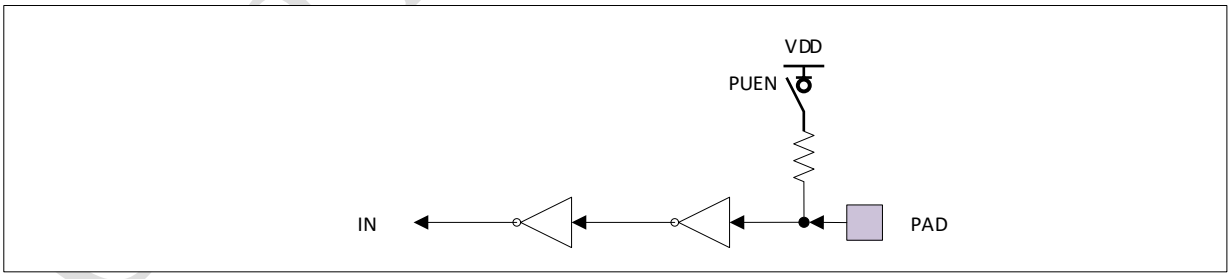


图 7.4 上拉输入模式结构图

7.1.4 下拉输入模式

下拉输入模式下端口 IN 默认为低电平，外部设备或电阻不提供确定的高低电平也不会漏电。

配置方式：Px_MFPx=000、Px_MODEx=00、Px_DIENx=1、Px_PUENx=0、Px_PDENx=1

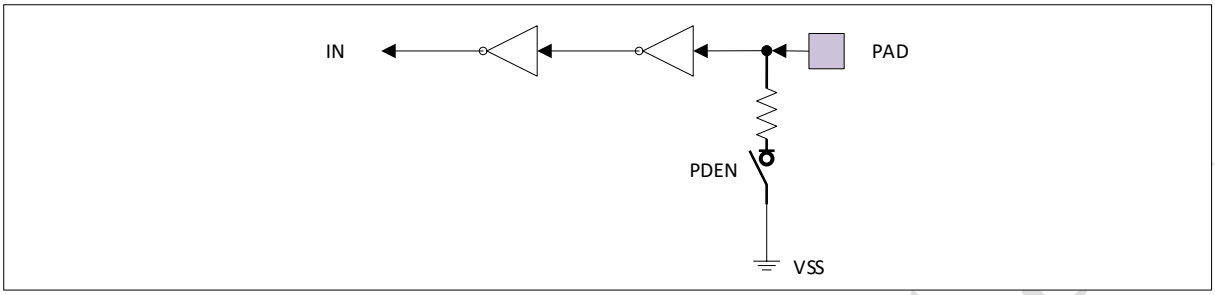


图 7.5 下拉输入模式结构图

7.1.5 推挽输出模式

推挽输出模式当端口锁定为 1 时，提供持续的强上拉。推挽输出模式用于需要从端口输出大电流时的应用。

配置方式：Px_MFPx=000、Px_MODEx=01、Px_DIENx=0、Px_PUENx=0、Px_PDENx=0

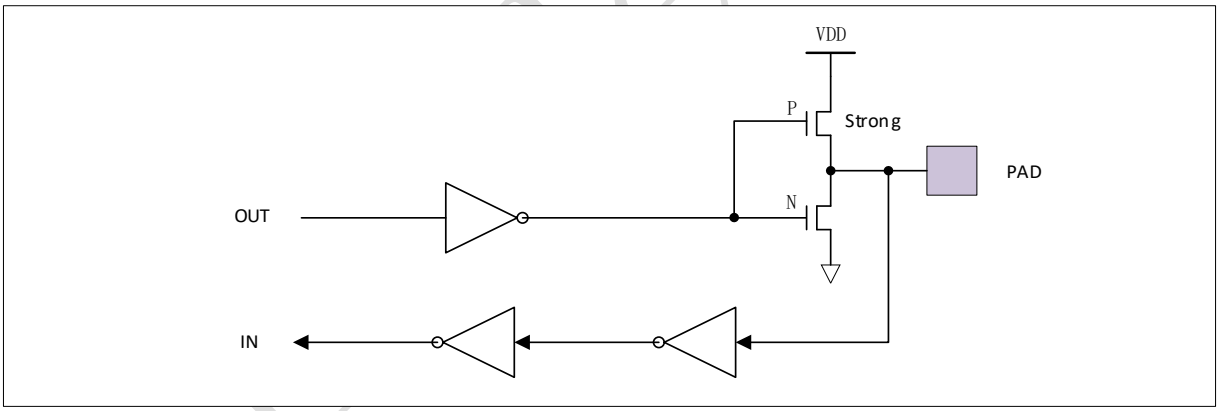


图 7.6 推挽输出模式结构图

7.1.6 开漏输出模式

开漏输出配置关闭所有内部上拉，当端口锁定为逻辑 0 时，仅打开驱动端口的下拉晶体管。当端口锁存为逻辑 1 时，它就和输入模式一样。通常用于 I2C 输出线上，开漏引脚需要加一个外部上拉电阻，典型连一个电阻到 V_{DD}。用户需要注意的是，开漏模式输出逻辑 1 的时候，应该由外部设备或电阻提供一个确定的电平。悬浮的引脚在掉电状态下会引起漏电。

配置方式：Px_MFPx=000、Px_MODEx=10、Px_DIENx=1、Px_PUENx=0、Px_PDENx=0

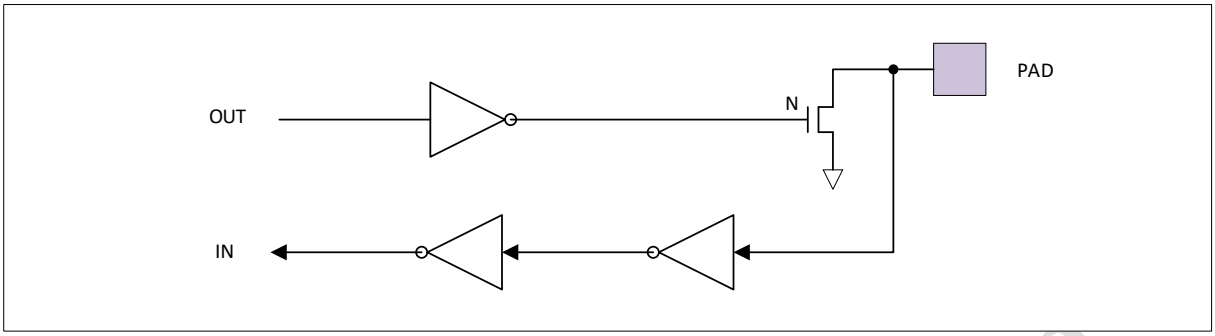


图 7.7 开漏输出模式结构图

7.2 IOMUX 分配方案

MFP	3'b000	3'b001	3'b010	3'b011	3'b101	3'b110	3'b111	Default
P1	P1_0	t2	cc0	PWM0	rxd0	USB_DP	-	ANA
	P1_1	t2ex	spssn[2]	PWM1	txd0	USB_DM	-	ANA
	P1_2	cc1	sck	PWM2	-	-	-	ANA
	P1_3	-	-	-	-	-	-	ANA
	P1_4	-	-	-	-	-	-	ANA
	P1_5	-	-	-	-	-	-	ANA
	P1_6	-	-	-	-	-	-	ANA
P3	P3_0	rxd0	spssn[1]	PWM3	Scl	ADC_CH0	-	ANA
	P3_1	txd0	ssn/spssn[0]	PWM4	Sda	ADC_CH1	-	ANA
	P3_2	int0	-	PWM5	C2CK	ADC_CH2	-	C2CK
	P3_3	int1	miso	-	-	ADC_CH3	-	ANA
	P3_4	t0	-	spssn[3]	C2DAT	ADC_CH4	-	C2DAT
	P3_5	t1	mosi	spssn[4]	-	ADC_CH5	-	ANA
	P3_6	scl	cc2	spssn[5]	rxd0	ADC_CH6	-	PAD_NRST
	P3_7	sda	cc3	spssn[6]	txd0	ADC_CH7	EN_PA	ANA

说明：

- 红色为模拟信号，对应 GPIO 配置为模拟输入输出态（DIEN=0、OE=0、PUEN=0、PDEN=0）；
- MFP 以及 MODE 共同决定 p1/3_oe[x]以及 p1/3_out[x]的值，当 MFP= 000 即 GPIO 模式时，相应的 OE、OUT 由{MODE[1]，MODE[0]}控制。当 MFP≠000 即非 GPIO 模式时，相应的 OE、OUT 由 MFP 来选择；
- SYS_P1/3_OE 寄存器仅在非 GPIO 模式下生效，GPIO 模式下的 OE 由 MODE 控制；
- P1 口的 DIEN、PUEN、PDEN 完全由寄存器直接控制；
- P3 口的 DIEN、PUEN、PDEN 部分受其他选择信号控制。

7.3 寄存器说明

7.3.1 寄存器映射

Register	Offset	R/W	Description	Reset Value
SYS_SEL	0xF8	R/W	System 寄存器地址选择	0x00
SYS_DAT	0xF9	R/W	System 寄存器数据读写寄存器	0x00

Register	SYS_SEL	R/W	Description	Reset Value
SYS_P1_MODE0	0x0	R/W	P1 GPIO mode select bit0，只影响 p1_oe 以及 p1_out	0x00
SYS_P1_MODE1	0x1	R/W	P1 GPIO mode select bit1，只影响 p1_oe 以及 p1_out	0x00
SYS_P3_MODE0	0x2	R/W	P3 GPIO mode select bit0，只影响 p3_oe 以及 p3_out	0x00
SYS_P3_MODE1	0x3	R/W	P3 GPIO mode select bit1，只影响 p3_oe 以及 p3_out	0x00
SYS_P1_DIEN	0x4	R/W	P1 input enable control	0x00
SYS_P3_DIEN	0x5	R/W	P3 input enable control	0x54
SYS_P1_PUEN	0x6	R/W	P1 pull up enable control	0x00
SYS_P3_PUEN	0x7	R/W	P3 pull up enable control	0x54
SYS_P1_PDEN	0x8	R/W	P1 pull down enable control	0x00
SYS_P3_PDEN	0x9	R/W	P3 pull down enable control	0x00
SYS_P1_OE	0xA	R/W	P1 output enable control	0x00
SYS_P3_OE	0xB	R/W	P3 output enable control	0x00
SYS_P1_MFP0	0xC	R/W	P1 multi function port select bit0	0x00
SYS_P1_MFP1	0xD	R/W	P1 multi function port select bit1	0x00
SYS_P1_MFP2	0xE	R/W	P1 multi function port select bit2	0x00
SYS_P3_MFP0	0xF	R/W	P3 multi function port select bit0	0x14
SYS_P3_MFP1	0x10	R/W	P3 multi function port select bit1	0x00
SYS_P3_MFP2	0x11	R/W	P3 multi function port select bit2	0x14
SYS_USB_CTRL	0x12	R/W	USB control register	0x00
SYS_RSTDBC	0x13	R/W	Pad reset de-bounce control register	0xFF
SYS_USB_CTRL1	0x14	R/W	USB control register 1	0x00
SYS_USB_CTRL2	0x15	R/W	USB control register 2	0x04
EINT0_DBC	0x16	R/W	Ext int0 de-bounce control register	0x40
EINT1_DBC	0x17	R/W	Ext int1 de-bounce control register	0x40
SYS_READ	0x18	RO	System read only register	0x00

7.3.2 寄存器描述

7.3.2.1 P1_MODE0& P1_MODE1

Register	SYS_SEL	R/W	Description	Reset Value
P1_MODE0	0x0	R/W	P1 GPIO mode select bit0，只影响 p1_oe 以及 p1_out	0x00
P1_MODE1	0x1	R/W	P1 GPIO mode select bit1，只影响 p1_oe 以及 p1_out	0x00

bit	Description	Reset Value
6	{P1_MODE1[0]，P1_MODE0[0]}用于控制 p1_oe[0]，p1_out[0]	{0,0}
5	{P1_MODE1[1]，P1_MODE0[1]}用于控制 p1_oe[1]，p1_out[1]	{0,0}
4	{P1_MODE1[2]，P1_MODE0[2]}用于控制 p1_oe[2]，p1_out[2]	{0,0}
3	{P1_MODE1[3]，P1_MODE0[3]}用于控制 p1_oe[3]，p1_out[3]	{0,0}
2	{P1_MODE1[4]，P1_MODE0[4]}用于控制 p1_oe[4]，p1_out[4]	{0,0}
1	{P1_MODE1[5]，P1_MODE0[5]}用于控制 p1_oe[5]，p1_out[5]	{0,0}
0	{P1_MODE1[6]，P1_MODE0[6]}用于控制 p1_oe[6]，p1_out[6]	{0,0}

P1_MODE1	P1_MODE0	Function
0	0	Input Mode，p1_oe=0，p1_out=port1_o
0	1	Push-pull Output Mode，p1_oe=1，p1_out=port1_o
1	0	Open-drain Output Mode，p1_oe= !port1_o，p1_out=0
1	1	Open-drain Output Mode，p1_oe= !port1_o，p1_out=0

7.3.2.2 P3_MODE0 & P3_MODE1

Register	SYS_SEL	R/W	Description	Reset Value
P3_MODE0	0x2	R/W	P3 GPIO mode select bit0，只影响 p3_oe 以及 p3_out	0x00
P3_MODE1	0x3	R/W	P3 GPIO mode select bit1，只影响 p3_oe 以及 p3_out	0x00

<table>
<tr><th>P1 bit</th><th>Description</th><th>Reset Value</th></tr>
<tr><td>7</td><td rowspan="9">{P3_MODE1[0]，P3_MODE0[0]}用于控制 p3_oe[0]，p3_out[0]
{P3_MODE1[1]，P3_MODE0[1]}用于控制 p3_oe[1]，p3_out[1]
{P3_MODE1[2]，P3_MODE0[2]}用于控制 p3_oe[2]，p3_out[2]
… …
{P3_MODE1[7]，P3_MODE0[7]}用于控制 p3_oe[7]，p3_out[7]
<table><tr><th>P3_MODE1</th><th>P3_MODE0</th><th>Function</th></tr><tr><td>0</td><td>0</td><td>Input Mode，p3_oe=0，p3_out=port3_o</td></tr><tr><td>0</td><td>1</td><td>Push-pull Output Mode，p3_oe=1，p3_out=port3_o</td></tr><tr><td>1</td><td>0</td><td>Open-drain Output Mode，p3_oe= !port3_o，p3_out=0</td></tr><tr><td>1</td><td>1</td><td>Open-drain Output Mode，p3_oe= !port3_o，p3_out=0</td></tr></table></td><td>{0,0}</td></tr>
<tr><td>6</td><td>{0,0}</td></tr>
<tr><td>5</td><td>{0,0}</td></tr>
<tr><td>4</td><td>{0,0}</td></tr>
<tr><td>3</td><td>{0,0}</td></tr>
<tr><td>2</td><td>{0,0}</td></tr>
<tr><td>1</td><td>{0,0}</td></tr>
<tr><td>0</td><td>{0,0}</td></tr>
</table>

7.3.2.3 P1_DIEN

Register	SYS_SEL	R/W	Description	Reset Value
P1_DIEN	0x4	R/W	P1 input enable	0x00

Bits	Description
[6:0]	Each of these bits is used to control if the digital input path of corresponding P1.n pinis disabled. If input is analog signal, users can disable P1.n digital input path to avoid input currentleakage. 1 = P1.n digital input path enabled. 0 = P1.n digital input path disabled (digital input tied to low).

7.3.2.4 P3_DIEN

Register	SYS_SEL	R/W	Description	Reset Value
P3_DIEN	0x5	R/W	P3 input enable	0x54

Bits	Description
[7:0]	Each of these bits is used to control if the digital input path of corresponding P3.n pinis disabled. If input is analog signal, users can disable P3.n digital input path to avoid input currentleakage. 1 = P3.n digital input path enabled. 0 = P3.n digital input path disabled (digital input tied to low).

7.3.2.5 P1_PUEN

Register	SYS_SEL	R/W	Description	Reset Value
P1_PUEN	0x6	R/W	P1 pullup enable	0x00

Bits	Description
[6:0]	Each of these bits is used to control if the digital pull up path of corresponding P1.npin is enabled. 0 = P1.n digital pull up path disabled. 1 = P1.n digital pull up path enabled.

7.3.2.6 P3_PUEN

Register	SYS_SEL	R/W	Description	Reset Value
P3_PUEN	0x7	R/W	P1 pullup enable	0x54

Bits	Description
[7:0]	Each of these bits is used to control if the digital pull up path of corresponding P1.npin is enabled. 0 = P1.n digital pull up path disabled. 1 = P1.n digital pull up path enabled.

7.3.2.7 P1_PDEN

Register	SYS_SEL	R/W	Description	Reset Value
P1_PDEN	0x8	R/W	P1 pull down enable	0x00

Bits	Description
[6:0]	Each of these bits is used to control if the digital pull down path of corresponding P1.npin is enabled. 0 = P1.n digital pull down path disabled. 1 = P1.n digital pull down path enabled.

7.3.2.8 P3_PDEN

Register	SYS_SEL	R/W	Description	Reset Value
P3_PDEN	0x9	R/W	P1 pull down enable	0x00

Bits	Description
[7:0]	Each of these bits is used to control if the digital pull down path of corresponding P1.npin is enabled. 0 = P1.n digital pull down path disabled. 1 = P1.n digital pull down path enabled.

7.3.2.9 P1_OE

Register	SYS_SEL	R/W	Description	Reset Value
P1_OE	0xA	R/W	P1 output enable	0x00

Bits	Description
[6:0]	Each of these bits is used to control if the digital output path of corresponding P1.n pinis disabled 1 = P1.n digital output enabled 0 = P1.n digital output disabled

7.3.2.10 P3_OE

Register	SYS_SEL	R/W	Description	Reset Value
P3_OE	0xB	R/W	P3 output enable	0x00

Bits	Description
[7:0]	Each of these bits is used to control if the digital output path of corresponding P1.n pinis disabled 1 = P1.n digital output enabled. 0 = P1.n digital output disabled .

7.3.2.11 P1_MFP0& P1_MFP1 & P1_MFP2

Register	SYS_SEL	R/W	Description	Reset Value
P1_MFP0	0xC	R/W	P1 multi function port select bit0	0x00
P1_MFP1	0xD	R/W	P1 multi function port select bit1	0x00
P1_MFP2	0xE	R/W	P1 multi function port select bit2	0x00

bit	Description	Reset Value
6	Reference 7.2	{0,0}
5		{0,0}
4		{0,0}
3		{0,0}
2		{0,0}
1		{0,0}
0		{0,0}

7.3.2.12 P3_MFP0& P3_MFP1 & P3_MFP2

Register	SYS_SEL	R/W	Description	Reset Value
P3_MFP0	0x0F	R/W	P3 multi function port select bit0	0x14
P3_MFP1	0x10	R/W	P3 multi function port select bit1	0x00
P3_MFP2	0x11	R/W	P3 multi function port select bit2	0x14

bit	Description	Reset Value
7	Reference 2.1	{0,0,0}
6		{0,0,0}
5		{0,0,0}
4		{1,0,1}
3		{0,0,0}
2		{1,0,1}
1		{0,0,0}
0		{0,0,0}

7.3.2.13 SYS_USB_CTRL

Register	SYS_SEL	R/W	Description	Reset Value
SYS_USB_CTRL	0x12	R/W	控制 USB 的使能和上拉控制	0x00

bit	Description	Reset Value
[7]	MUX_RCL，1：将 RCL_15K mux 到 P10，0：无操作 还有个寄存器 RCL_TEST_EN 要配置一下，注意是个需要触发的信号	0
[6]	MUX_RCH，1：将 RCH mux 到 P11，0：无操作	0
[5]	MUX_DPLL48_DIV8，1：将 DPLL48_DIV8 mux 到 P12，0：无操作 还有个寄存器 DPLL_CTRL[4]要打开	0
[4]	MUX_SYS_CLK_DIV16，1：将 sys_clk_div16 mux 到 P30，0：无操作	0
[3]	MUX_SYS_CLK，1：将 sys_clk mux 到 P31，0：无操作	0
[2]	USB_PU2，对应数模接口 USB_PUEN_150K，用于控制 DM 的上拉电阻是否打开 1：打开 DM 的 150K 上拉电阻 0：关闭	0
[1]	USB_PU，对应数模接口 USB_PUEN_1P5K，用于控制 DP 的上拉电阻是否打开 1：打开 DP 的 1.5K 上拉电阻 0：关闭	0
[0]	EN_USB，控制数模接口 EN_USB 以及 XRAM.usb_ram_en 1：使能模拟信号 EN_USB 以及 XRAM.usb_ram_en 0：关闭	0

注意：

由于 DM 与 P11 连在一起，DP 与 P10 连在一起，所以 USB 功能只能通过 P10、P11 来测试，此时需要把 P10、P11 配置为模拟输入模式（ DIEN=0 、 OE=0 、 PUEN=0 、 PDEN=0）。

7.3.2.14 SYS_RSTDBC

Register	SYS_SEL	R/W	Description	Reset Value
SYS_RSTDBC	0x13	R/W	PAD reset 消抖控制位，消抖时间等于 SYS_RSTDBC[7:0] * 512 * Tcpu	0xFF

bit	Description	Reset Value
[7:0]	PAD reset 消抖控制位，消抖时间等于 SYS_RSTDBC[7:0] * 512 * Tcpu	0xFF

注意：掉电模式下自动关闭外部 RST 消抖功能，退出掉电模式又自动打开。

7.3.2.15 SYS_USB_CTRL1

Register	SYS_SEL	R/W	Description	Reset Value
SYS_USB_CTRL1	0x14	R/W	控制 USB 的插入中断 debounce	0x00

bit	Description	Reset Value
[7:0]	usb_valid_removal_cycles[15:8]，For usb insert detection usage Usb deivce 检测插入事件产生的条件是：首先处于拔出状态，然后监测到 DIP&DIM=0，在检测拔出状态上，DIP=DIM=1 必须维持指定 48Mhz 周期个数，才认定是有效的 removal 状态	0x00

7.3.2.16 SYS_USB_CTRL2

Register	SYS_SEL	R/W	Description	Reset Value
SYS_USB_CTRL2	0x15	R/W	控制 USB 的插入中断 debounce	0x04

bit	Description	Reset Value
[7:0]	usb_valid_removal_cycles[7:0]，For usb insert detection usage Usb deivce 检测插入事件产生的条件是：首先处于拔出状态，然后监测到 DIP&DIM=0，在检测拔出状态上，DIP=DIM=1 必须维持指定 48Mhz 周期个数，才认定是有效的 removal 状态	0x04

7.3.2.17 EINT0_DBC

Register	SYS_SEL	R/W	Description	Reset Value
EINT0_DBC	0x16	R/W	控制外部中断 int0 的去抖功能	0x40

bit	Signal	Description	Reset Value
[7]	int0dbc_en	消抖使能选择：1，使能；0，关闭	0
[6:0]	int0dbc_t	消抖计数时钟个数，消抖时间是指端口输入时，其对应端口低电平所需要维持的时间。消抖时间等于 int0dbc_t [6:0] * 128us，不支持 0	0x40

7.3.2.18 EINT1_DBC

Register	SYS_SEL	R/W	Description	Reset Value
EINT1_DBC	0x17	R/W	控制外部中断 int1 的去抖功能	0x40

bit	Signal	Description	Reset Value
[7]	int1dbc_en	消抖使能选择：1，使能；0，关闭	0
[6:0]	int1dbc_t	消抖计数时钟个数，消抖时间是指端口输入时，其对应端口低电平所需要维持的时间。消抖时间等于 int1dbc_t [6:0] * 128us，不支持 0	0x40

7.3.2.19 SYS_READ

Register	SYS_SEL	R/W	Description	Reset Value
SYS_READ	0x18	RO	一个只读寄存器	0x00

bit	Signal	Description	Reset Value
[7:1]	Reserved		0000000
[0]	c2_connect_r	c2_connect_r 信号只读寄存器，1：C2 强制连接打开，0：未打开	0

第8章 UART

PAN262x 包含一个可编程的全双工 UART。其主要由数据缓冲寄存器、发送控制器、接收控制器、输入移位寄存器和若干控制电路组成。

8.1 特征

- 全双工模式，发送/接收，其可配成三种异步模式
- 数据位宽度为8bit
- 波特率可配置，最大支持115200
- 奇偶校验可配置，其校验bit需要由软件更新获得
- 发送/接收中断，中断向量号为4

8.2 模块框图

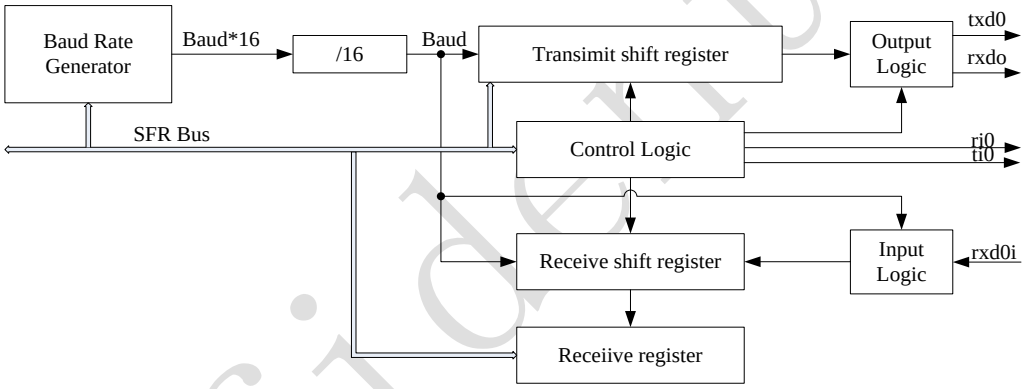


图 8-1　UART 模块框图

8.3 功能描述

8.3.1 波特率

（1）UART 被配置成模式 1 与模式 3 时，其波特率是可配置的。波特率的来源由两个选择，一个是 UART 内部的波特率发生器；另一个是由定时器 timer1 溢出率来确定。这由 ADCON 寄存器的第 7bit bd 选择。具体如下：

$$当 bd(adcon.7)=0:$$

$$baud\ rate=\frac{2^{SMOD}*fclk}{32}*(timer1\ over\ rate)$$

$$当 bd(adcon.7)=1:$$

$$baud\ rate=\frac{2^{SMOD}*fclk}{64*(2^{10}-s0rel)}$$

其中，bd 为 adcon 寄存器第 7bit；smod 为 PCON 的第 7 bit；s0rel 为寄存器 s0relh 与 s0rell。

（2）UART 在模式 2 时，其波特率是系统时钟的 32 分频或是 64 分频，其由 PCON 的第 7bit smod 来选择。

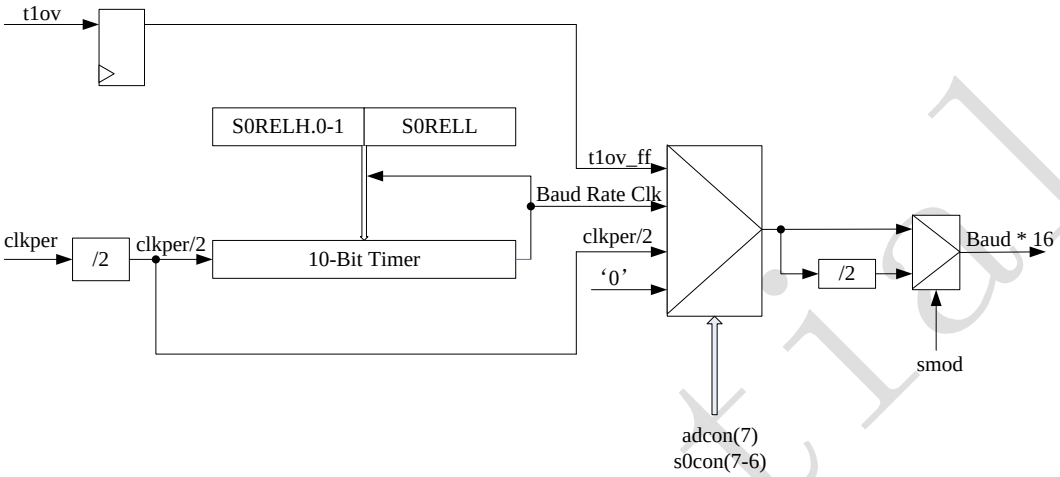


图 8-2　UART 波特率产生框图

8.3.2 工作模式 1

在模式 1 中，UART 作为具有 8 个数据位的异步发送器/接收器，和可编程波特率。波特率的配置如上一小节所述。

当写入 s0buf 寄存器开始传输。“txd0”引脚输出数据。传输的第一位是起始位（总是 0），然后 8 位数据继续，然后传输停止位（总是 1）。

当接收开始时，在引脚“rxd0i”处检测到下降沿，即开始接收，接收完成后输入数据保存在 s0buf 寄存器中，停止位的值在接收完成之后保存在 s0con 寄存器中的 rb80。

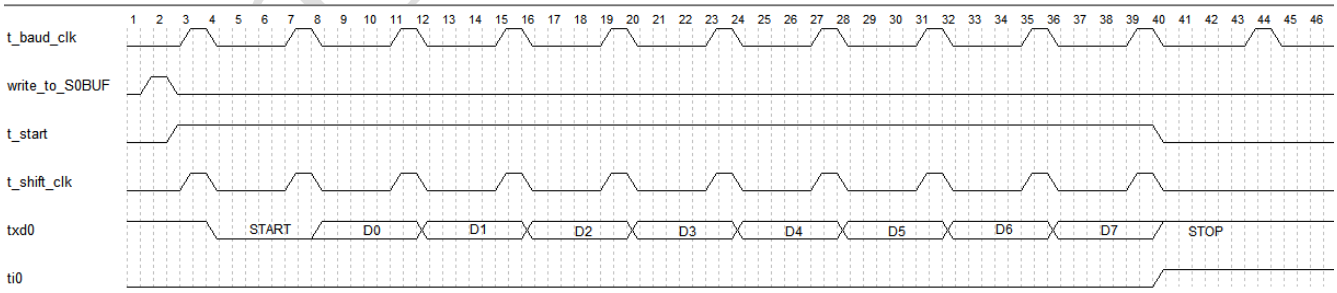


图 8-3 模式 1 的 UART 发送

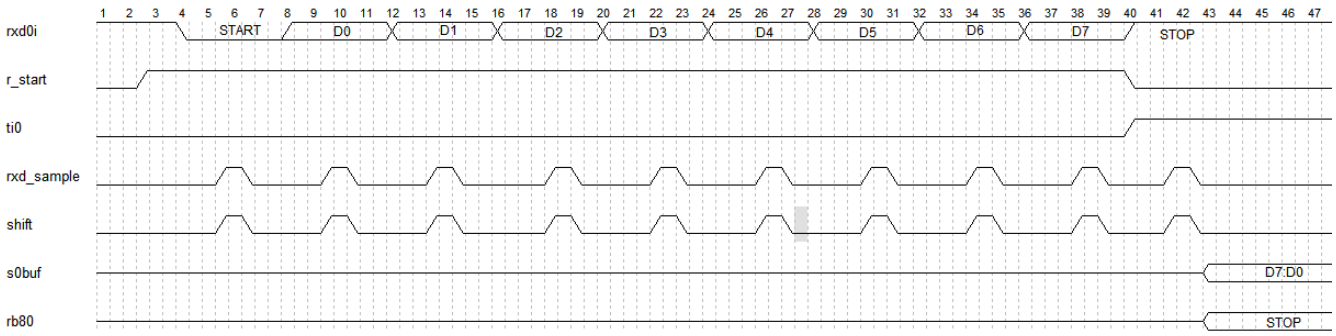


图 8-4 模式 1 的 UART 接收

8.3.3 工作模式 2

在模式 2 中，UART 作为具有 8 个数据位，1 个奇偶校验位的异步发送器/接收器，波特率是系统时钟的 32 分频或是 64 分频。波特率的配置如上一小节所述。

当写入 s0buf 寄存器时，UART 开始传输。txd0 引脚输出数据。这个传输的第一位是起始位（始终为 0），然后 9 位的数据由 s0con 寄存器的位 tb80 控制，然后传输停止位（始终为 1）。

当接收数据时，UART 在引脚“rxd0”处检测到下降沿，接收完成后输入数据保存在 s0buf 寄存器中，第 9 位在传输完成后，保存在 s0con 寄存器中的位 rb80。

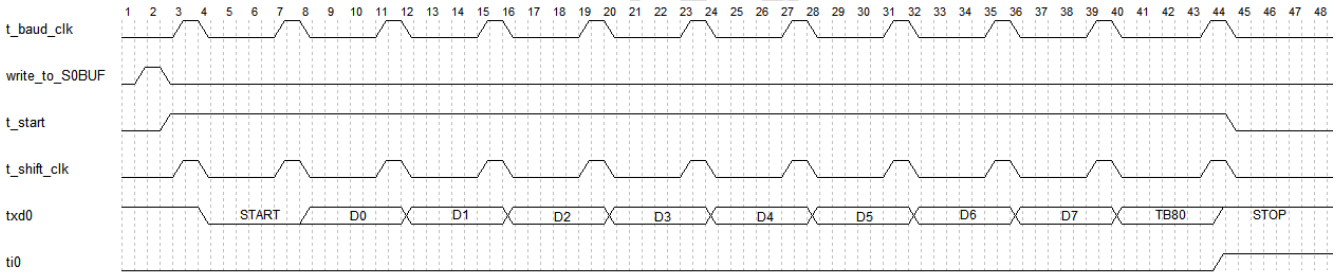


图 8-5 模式 2 的 UART 发送

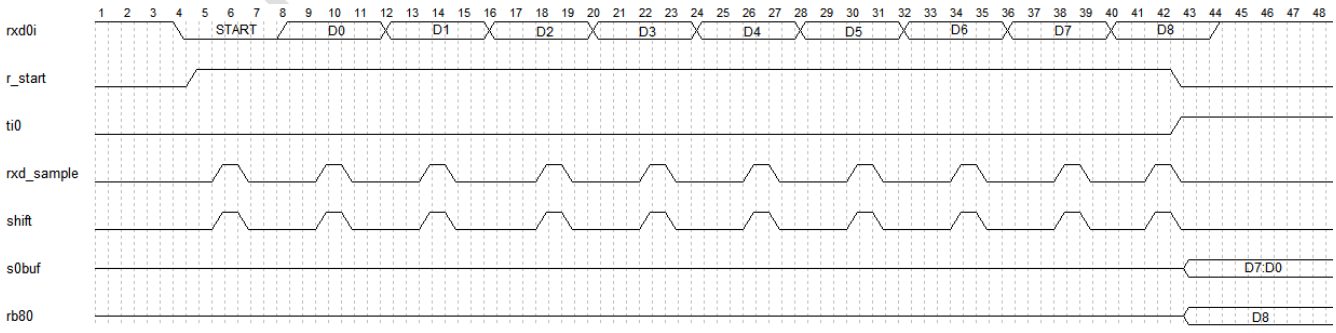


图 8-6 模式 2 的 UART 接收

8.3.4 工作模式 3

在模式 3 中，UART 作为具有 8 个数据位，1 个奇偶校验位的异步发送器/接收器，波特率是

可配置的。波特率的配置如 1.3.1 小节所述。其与模式 2 的差别在与，其波特率可由内部波特率产生器或是定时器 1 产生。

当写入 s0buf 寄存器时，UART 开始传输。txd0 引脚输出数据。这个传输的第一位是起始位（始终为 0），然后 9 位的数据由 s0con 寄存器的位 tb80 控制，然后传输停止位（始终为 1）。

当接收数据时，UART 在引脚“rxd0”处检测到下降沿，接收完成后输入数据保存在 s0buf 寄存器中，第 9 位在传输完成后，保存在 s0con 寄存器中的位 rb80。

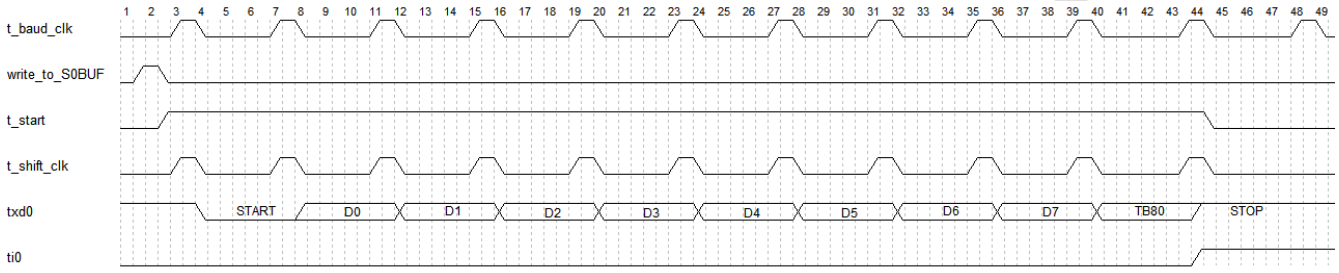


图 8-7 模式 3 的 UART 发送

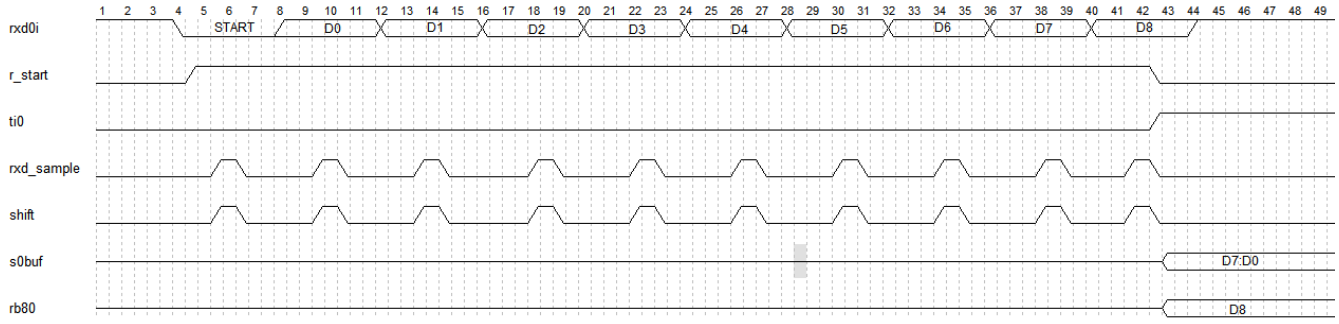


图 8-8 模式 3 的 UART 接收

8.4 寄存器列表

Register	Offset	R/W	Description	Reset Value
S0CON	0x98	R/W	UART 控制寄存器	0x00
S0BUF	0x99	R/W	UART 数据寄存器	0x00
S0RELL	0xAA	R/W	UART 重载寄存器（低八位）	0xd9
S0RELH	0xBA	R/W	UART 重载寄存器（高两位）	0x03
IEN0	0xA8	R/W	中断使能寄存器	0x00
ADCON	0xD8	R/W	UART 波特率选择寄存器	0x00

8.5 寄存器描述

8.5.1 Serial Port0 Control Register (S0CON)

Register	offset	R/W	Description	Reset Value
S0CON	0x98	R/W	UART 控制寄存器	0x00

Bits	Symbol	Type	Description
7	sm0	R/W	01：模式 1，10：模式 2，11：模式 3
6	sm1	R/W	
5	sm20	R/W	多机通讯使能寄存器，0: 不使能，1：使能
4	ren0	R/W	UART 接收使能，0：不使能，1：使能
3	tb80	R/W	UART 模式 2 和模式 3 中的 9th 发送位控制，奇偶校验位或是多机通讯控制位
2	rb80	R/W	UART 模式 1 中，若是使能多机通讯，该 Bit 在接收时是停止位； 模式 2 和模式 3 中，其反应接收的 9th 位
1	ti0	R/W	发送中断标志位，其表示一次发送的完成，其必须由软件进行清除
0	ri0	R/W	接收中断标志位，其表示一次接收的完成，其必须由软件进行清除

UART 模式与波特率配置：

sm0	sm1	Mode	Description	Baud Rate
0	1	Mode1	8-bit UART	可配置的波特率（请参考 8.3.1 小节）
1	0	Mode2	9-bit UART	固定波特率（请参考 8.3.1 小节）
1	1	Mode3	9-bit UART	可配置的波特率（请参考 8.3.1 小节）

8.5.2 Serial Port0 Data Buffer (S0BUF)

Register	offset	R/W	Description	Reset Value
S0BUF	0x99	R/W	UART 发送接收数据寄存器	0x00

Bits	Symbol	Type	Description
7:0	S0BUF	R/W	UART 发送接收数据寄存器

8.5.3 Serial Port0 Reload Low Register(S0RELL)

Register	offset	R/W	Description	Reset Value
S0RELL	0xAA	R/W	UART 波特率生成寄存器	0xD9

Bits	Symbol	Type	Description
7:0	S0RELL	R/W	UART 波特率生成寄存器，低八位

8.5.4 Serial Port0 Reload High Register(SORELH)

Register	offset	R/W	Description	Reset Value
S0RELH	0xBA	R/W	UART 波特率生成寄存器	0xFF

Bits	Symbol	Type	Description
1:0	S0RELH	R/W	UART 波特率生成寄存器，高两位

8.5.5 Interrupt Enable0 Register(IEN0)

Register	offset	R/W	Description	Reset Value
IEN0	0xA8	R/W	中断使能寄存器	0x00

Bits	Symbol	Type	Description
7	eal	R/W	中断总使能 0：所有中断不使能 1：使能中断总开关
6:5	reserved	R/W	other function
4	es0	R/W	UART 中断使能 0：UART 中断不使能 1：UART 中断使能，且 eal 置 1
3:0	reserved	R/W	other function

8.5.6 Serial Port0 Baud Rate Select register (ADCON)

Register	offset	R/W	Description	Reset Value
ADCON	**0xD8**	**R/W**	**Serial Port0 Baud Select**	**0x00**

Bits	**Symbol**	**Type**	**Description**
7	bd	R/W	UART 波特率选择（模式 1 与模式 3）： 0：timer1 溢出 1：内部波特率产生器
6	timer1_div	R/W	timer1 分频选择 0：12 分频 1：3 分频，此时支持波特率 115200
5:0	Reserved	Re-served	

8.5.7 Power Control Register(PCON)

Register	offset	R/W	Description	Reset Value
PCON	**0x87**	**R/W**	**Serial Port0 Baud Select**	**0x00**

Bits	Symbol	Type	Description
7	smod	R/W	UART 波特率选择，具体参考 8.3.1 小节
6:0	Reserved		

第9章 I2C

9.1 特征

- 两种使用模式，Master发送、Master接收
- 时钟分频，速度可达266Kb/s
- 7-bit地址格式
- 读写缓存共用，位宽为8bit，深度为1
- SDA, SCL输入信号过滤，可过滤掉电平长度小于3个时钟的脉冲
- 收发中断

9.2 模块框图

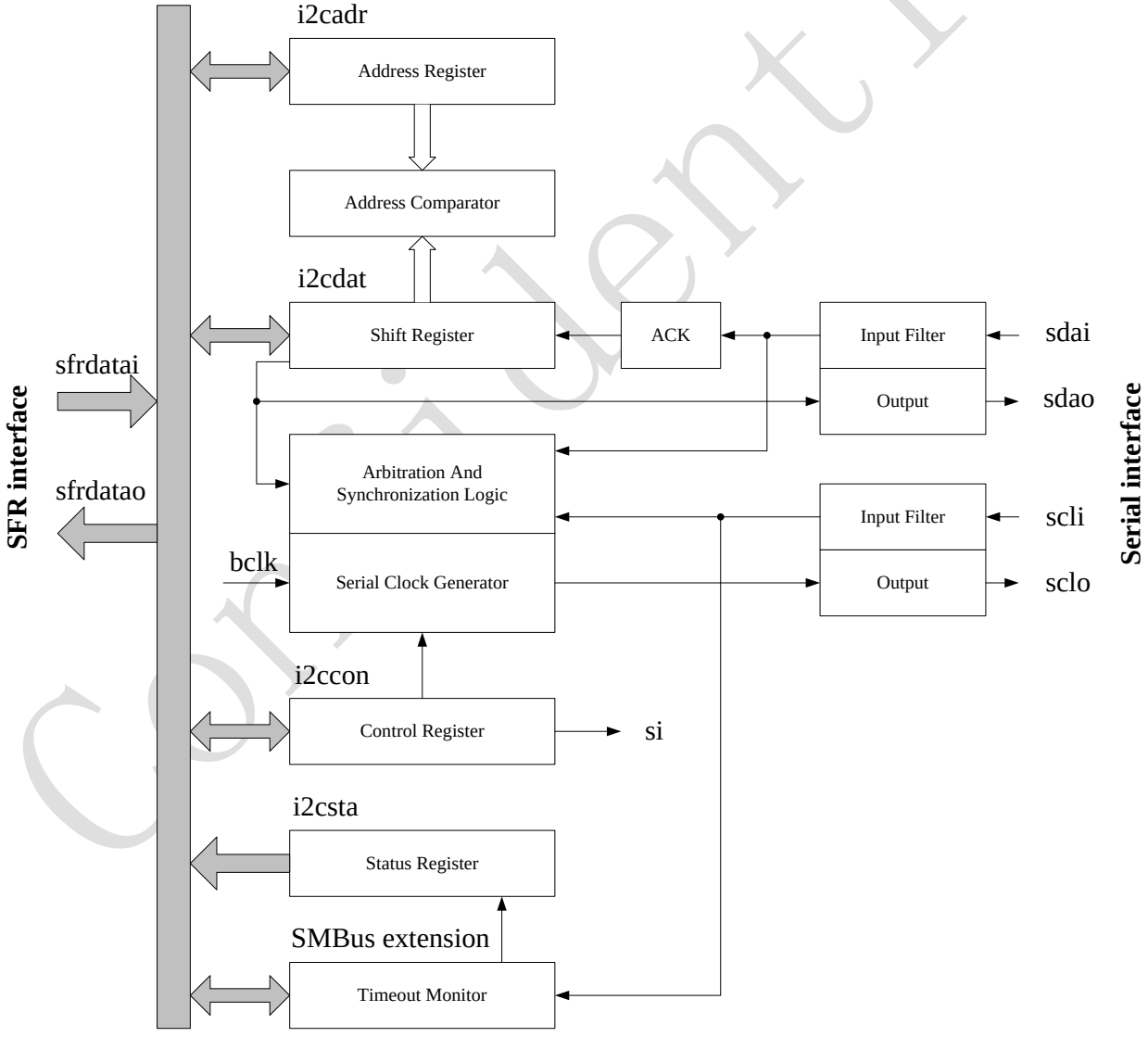


图 9-1　I2C 整体框图

9.3 功能描述

I2C 在连接设备时需要两根线进行传输：“scl”（串行时钟线）和“sda”（串行数据线）。因为 I2C 是双向端口，因此其需要使用外部开漏缓冲。连接到总线的每个设备都可以通过一个唯一的地址进行软件寻址。I2C 是一个真正的多主总线，在两个或多个主同时启动数据传输时，其具有包括冲突检测和仲裁的功能，以防止数据损坏。I2C 并且具有滤波功能，可以过滤掉毛刺，从而保证数据的完整性与正确性。

I2C 传输速率最高可以达到 266 Kbit/s，可以在主机发送与主机接收两种模式下运行。

9.3.1 主机发送模式

在主机发送模式下，串行数据通过端口“sda”输出，串行时钟通过“scl”输出。具体时序如下图所示。

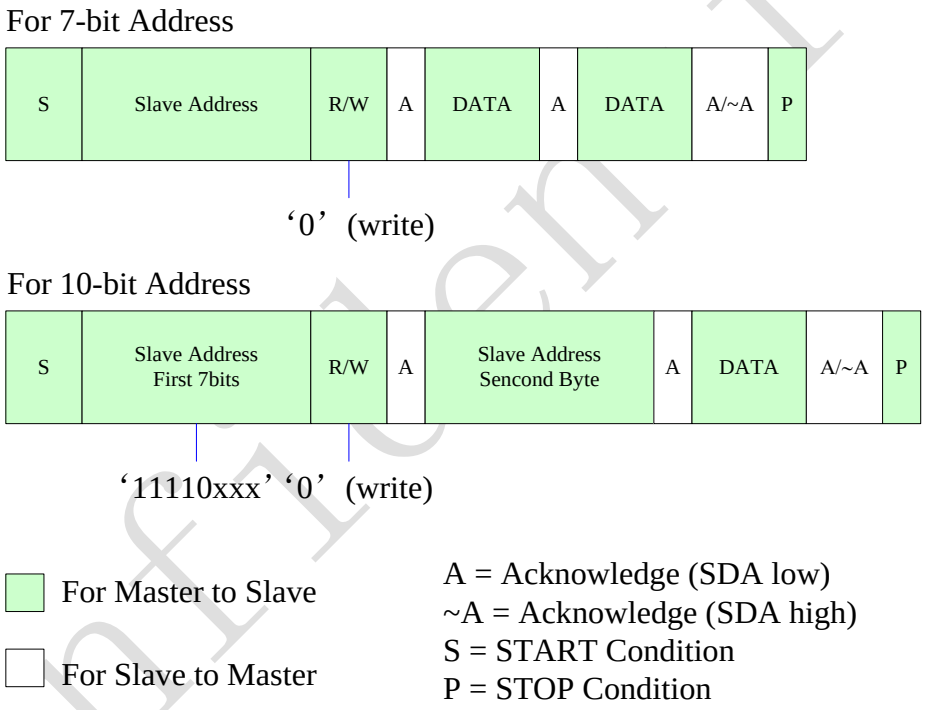


图 9-2 主机发送示意图

9.3.2 主机接收模式

在主机接收模式下，串行数据通过端口“sda”输入，串行时钟通过“scl”输出。具体时序如下图所示。

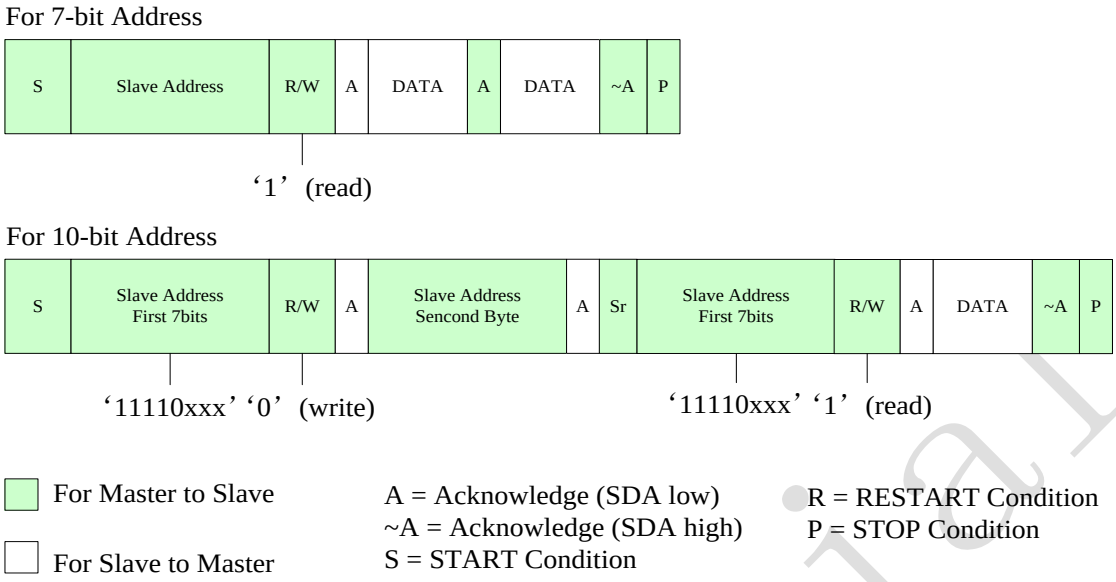


图 9-3 主机发送示意图

9.3.3 开启与停止信号

在数据传输过程中的 start 与 stop 信号，如下图所示。

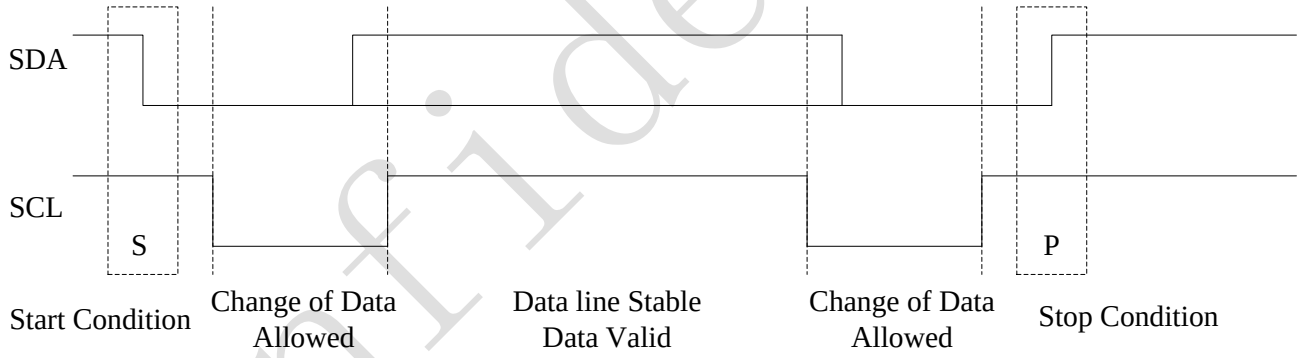


图 9-4 开启停止示意图

9.3.4 地址格式

下图为传输过程中，7 bit 地址时序图。

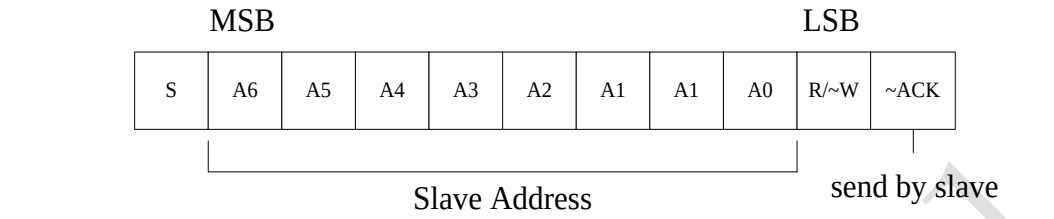


图 9-5 7bit 地址时序图

9.3.5 时钟分频

在主机模式下，scl 时钟由系统时钟分频得到。时钟发生器的分频系数由“i2ccon”的位“cr0”、“cr1”和“cr2”控制。下表显示了在主机模式下“clko”的速率。表中引用的“bclk”输入连接到定时器 1 溢出频率。

表 9-1 时钟分频

cr2	cr1	cr0	Bit Frequency Kbps	Clk Divided by
0	0	0	63	256
0	0	1	71	224
0	1	0	83	192
0	1	1	100	160
1	0	0	17	960
1	0	1	133	120
1	1	0	266	60
1	1	1	“bclk” input divided by 8	

9.3.6 输入滤波

输入信号与时钟（“clk”）同步，毛刺在小于三个时钟周期会被过滤掉。每个滤波器由三个触发器组成。第一个用于锁定输入，另外两个组成移位寄存器。当第二和第三触发器的状态为“11”或“00”，其为置位或是复位状态。、

9.3.7 中断产生

I2C 共有 15 个状态，当进入其中中的 14 个状态时，“i2ccon”寄存器的“si”标志将被硬件置位，并产生中断。唯一不产生中断，不被置位的状态是是 F8h，其表示没有检测到 I2C 时序。

在中断处理函数中，“i2ccon”寄存器的“si”标志需要被软件写“0”清除，写“1”无效。

其中，I2C 中断向量号是 8。

9.3.8 轮询状态

MCU 通过四个功能寄存器“i2ccon”（控制寄存器）、“i2csta”（状态寄存器）、“i2cdat”（数据寄存器）来控制 I2C。“i2ccon”寄存器包含全局 I2C 使能位“ens1”，时钟速率设置位“cr0”，“cr1”、“cr2”。其还提供开始位或停止位的配置（“sta”、“sto”位），以及控制 I2C 中 ACK 状态的“aa”。最后，“i2ccon”还提供中断请求标志“si”，在中断中需要检测 FSM，并及时清除。

“i2cdat”寄存器包含通过 I2C 总线传输的一个字节或通过 I2C 总线接收的一个字节。MCU 应该只在 I2C 中断发生时读取它。

“i2csta”寄存器反映 I2C 组件的 FSM 状态。此寄存器的三个最低有效位始终为零，有 15 个可能的状态。当其中的 14 个状态发生时，中断产生。唯一不产生中断的状态是 F8h 状态。其记录了 I2C 运行状态，软件读取此状态，并执行相应的操作。

具体状态含义如下表所示“SLA”表示从机地址；“R”表示 R/W 位=1，读取从机地址；“W”表示 R/W 位=0，写从机地址。

表 9-2 主机发送模式下的 I2C 状态

状态	状态含义	软件的响应					I2C 硬件下一步行为
		To/from i2cdat	To i2ccon				
			sta	sto	si	aa	
08H	开始信号已经发起	加载 SLA+W	X	0	0	X	SLA+W 将被发送 ACK 将被接收
10H	开始信号再次发起	加载 SLA+W	X	0	0	X	SLA+W 将被发送 ACK 将被接收
		加载 SLA+R	X	0	0	X	SLA+R 将被发送 I2C 将切换到 MST/REC mode
18H	SLA+W 已经被发送； ACK 已经被接收	加载 data	0	0	0	X	数据将被发送；ACK 将被接收
		没有行为	1	0	0	X	开始信号将再次发起
		没有行为	0	1	0	X	停止信号将被发起；STO 标志将被复位
		没有行为	1	1	0	X	停止信号将被发起，其紧接着开始信号，STO 标志将被复位
20H	SLA+W 已经被发送； 但是没有收到 ACK	加载 data	0	0	0	X	数据将被发送；ACK 将被接收
		没有行为	1	0	0	X	开始信号将再次被发起
		没有行为	0	1	0	X	停止信号将被发起；STO 标志将被复位
		没有行为	1	1	0	X	停止信号将被发起，其后紧接着开始信号，STO 标志将被复位
28H		加载 data	0	0	0	X	数据将被发送；ACK 将被接收
		没有行为	1	0	0	X	开始信号将再次被发起

	数据已经被发送，ACK 已经被接收	没有行为	0	1	0	X	停止信号将被发起；STO 标志将被复位
		没有行为	1	1	0	X	停止信号将被发起，其后紧接着开始信号，STO 标志将被复位
30H	数据已经被发送，ACK 没有被接收到	加载 data	0	0	0	X	数据将被发送；ACK 将被接收
		没有行为	1	0	0	X	开始信号将再次被发起
		没有行为	0	1	0	X	停止信号将被发起；STO 标志将被复位
		没有行为	1	1	0	X	停止信号将被发起，其后紧接着开始信号，STO 标志将被复位

表 9-3 主机接收模式下的 I2C 状态

Sta-tus Code	Status of Code	软件的响应					I2C 硬件下一步行为
		To/from i2cdat	To i2ccon				
			sta	sto	si	aa	
08H	开始信号已经被发起	加载 SLA+R	X	0	0	X	SLA+R 将被发送；ACK 将被接收
10H	开始信号已经被再次发起	加载 SLA+R	X	0	0	X	SLA+R 将被发送；ACK 将被接收
		加载 SLA+W	X	0	0	X	SLA+W 将被发送；I2C 将切换到 MST/TRX 模式
40H	SLA+R 已经被发送；ACK 已经被接收	没有行为	0	0	0	0	数据将被接收；ACK 将没有被返回
		没有行为	0	0	0	1	数据将被接收；ACK 将被返回
48H	SLA+R 已经被发送；ACK 没有被接收到	没有行为	1	0	0	X	将再次发起开始信号
		没有行为	0	1	0	X	停止信号将被发送；STO 标志将被复位
		没有行为	1	1	0	X	停止信号将被发起，其后紧接着开始信号，STO 标志将被复位
50H	数据已经被接收；ACK 已经被返回	读数据	0	0	0	0	数据将被接收；ACK 将不被返回
		读数据	0	0	0	1	数据将被接收；ACK 将被返回
58H	数据已经被接收；ACK 没有被返回	读数据	1	0	0	X	将再次发起开始信号
		读数据	0	1	0	X	停止信号将被发起；STO 标志将被复位
		读数据	1	1	0	X	停止信号将被发起，其后紧接着开始信号，STO 标志将被复位

表 9.4 I2C 的其他状态

状态	状态含义	软件的响应					I2C 硬件下一步行为
		To/from i2cdat	To i2ccon				
			sta	sto	si	aa	
38H	仲裁失败	没有行为	0	0	0	X	I2C 将被释放； 开始信号将被发起
		没有行为	1	0	0	X	当处于空闲状态时（进入主机模式）
F8H	没有相关有效信息； si=0	没有行为	没有行为				等待或是继续等待当前传输
00H	在 MST 期间总线错误	没有行为	0	1	0	X	在 MST 模式下，内部硬件会被影响。在所有情况下，总线将被释放。STO 标志将被复位。

9.4 寄存器列表

Register	offset	R/W	Description	Reset Value
I2C2DAT	0xDA	R/W	I2C 数据寄存器	0x00
I2C2CON	0xDC	R/W	I2C 控制寄存器	0x00
I2C2STA	0xDD	R	I2C 状态寄存器	0xF8

9.5 寄存器描述

9.5.1 I2C Data Register(I2CDAT)

I2C2DAT 寄存器为读写缓存，用来保存将要发送到 I2C 总线上的数据或从总线上接收到的数据。CPU 可读取或写入这个寄存器。

Register	offset	R/W	Description	Reset Value
I2C2DAT		R/W	I2C data register	0x00
Bits	**Description**			

[7:0]	dat	读写缓存

9.5.2 I2C Control Register(I2CCON)

I2CCON 寄存器用来操作 I2C 模块。CPU 可读取或写入该寄存器。其中有两个 bit 会被 I2C 硬件电路影响：当中断触发时，“si”位会被置 1。当 STOP 命令发送完毕后，“sto”位会被清除。

Register	R/W	Description	Reset Value
I2C2CON	R/W	I2C control register	0x00
Bits	Description		
[7]	cr2	时钟控制位	
[6]	ens1	I2C 使能位。 当 ens1=0 时，sda 和 scl 信号线为高。IO 口表现为高阻态； 当 ens1=1 时，模块使能	
[5]	sta	启动标志位 当 sta=1 时，I2C 检查总线状态，如果空闲将会启动 START 命令	
[4]	sto	结束标志位 当 sto=1 时，I2C 发送 STOP 命令到总线上。命令发送完毕后，硬件自动清零。	
[3]	si	中断标志位 I2C 进入到任何状态中（见 2.3.4 节，除去 F8 状态），都会触发中断，并拉高此中断标志位。软件写 0 清零，写 1 无影响。	
[2]	aa	ACK 标志位 当 aa=1 时，I2C 接收到数据后在总线上返回 ACK 信号 当 aa=0 时，I2C 接收到数据后在总线上返回 NACK 信号	
[1:0]	cr1,cr0	时钟控制位	

9.5.3 I2C Status Register(I2CSTA)

I2CSTA 寄存器用来保存 I2C 当前状态。CPU 只能读取该寄存器的值。

Register	R/W	Description	Reset Value
I2CSTA	R/W	I2C status register	0xF8
Bits	Description		

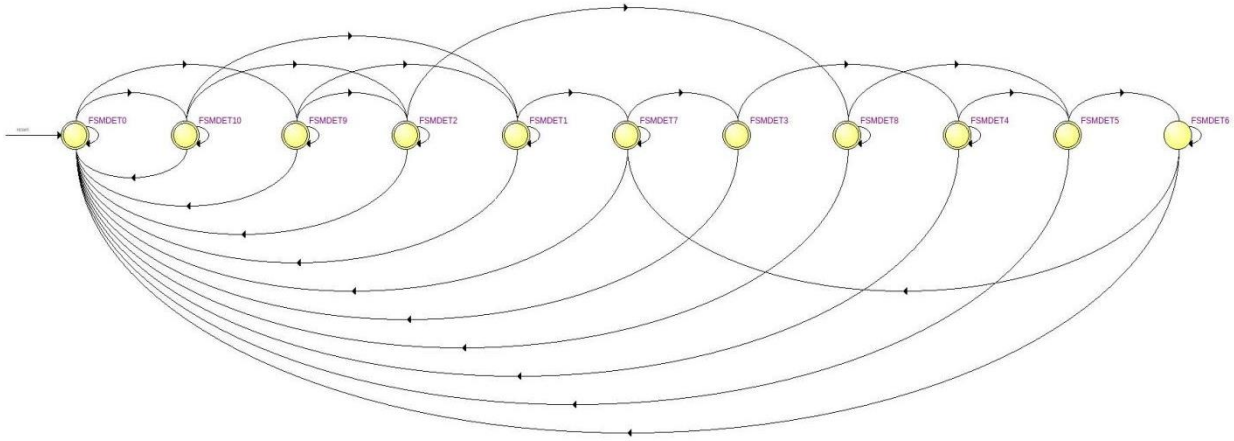
[7:3]	status	I2C 状态标志
[2:0]	reserved	

9.6 硬件状态机

I2C 模块内部有四个状态机：

- FSMDET 状态机检测 START/STOP 命令
- FSMSYNC 状态机用于同步时钟
- FSMMOD 状态机检测工作模式
- FSMSTA 状态机控制 I2C 运行

9.6.1 FSMDET



FSMDET 状态机用于检测 START/STOP 命令。检测到 START/STOP 命令后，状态机会等待 300ns。检测到 SCL 上升沿后，状态机会等待 200ns。在 8051XC2 系统设计中，时钟最高为 24MHz，控制等待时间的参数分别为 SDA_SAMPLE_COUNT=5，SDA_SWITCH_COUNT=8。

FSMDET0：SCL =0, SDA 不定。SCL 为低时，状态机会锁住在此状态。

FSMDET1：总线空闲，SCL=1,SDA=1。检测 START 命令；

FSMDET2：SCL=1,SDA=0。检测 STOP 命令；

FSMDET3：SCL=1,SDA=0，检测到 START 命令。

FSMDET4：SCL=1, SDA=0，检测 STOP 命令；

FSMDET5：SCL=1, SDA=1，检测到 STOP 命令，等待进入总线空闲；

FSMDET6：总线空闲；

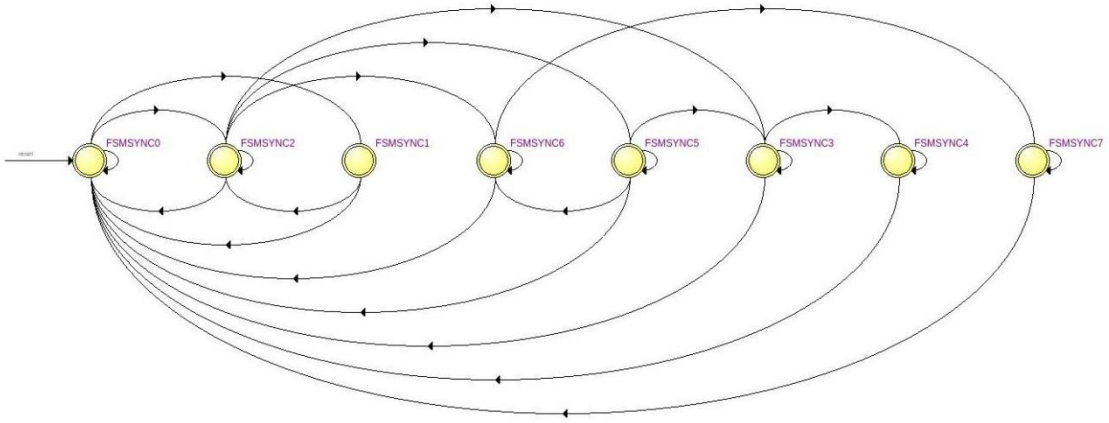
FSMDET7：SCL=1, SDA=0, 检测到 START 命令，等待 300ns；

FSMDET8：SCL=1, SDA=1, 检测到 STOP 命令，等待 300ns；

FSMDET9：SCL=1, SDA=1, 检测到 SCL 拉高进入此状态，并等待至少 200ns；

FSMDET10：SCL=1, SDA=0, 检测到 SCL 拉高后进入此状态，并等待至少 200ns。

9.6.2 FSMSYNC



FSMSYNC 状态机用于控制 SCL 产生以及时钟同步。

FSMSYNC0：复位状态；SCL 为高电平，等待高电平计数完成。

FSMSYNC1：SCL 为低电平，开启低电平计数；

FSMSYNC2：SCL 为低电平，等待低电平计数完成；

FSMSYNC3：确认 SCL 是否为高电平。

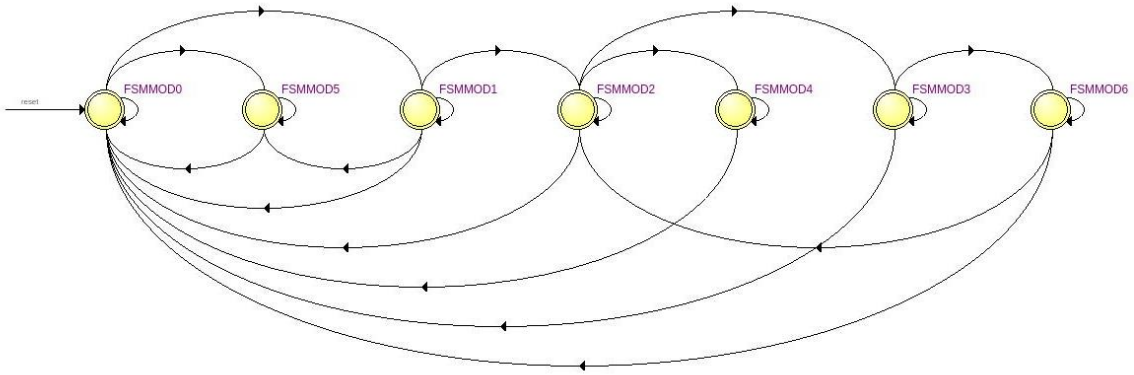
FSMSYNC4：延展 SCL 低电平；

FSMSYNC5：SCL 为低电平，等待中断处理完成；

FSMSYNC6：SCL 为低电平，等待 SDA 为低电平；

FSMSYNC7：等待下一次传输

9.6.3 FSMMOD



FSMMOD 状态机用于检测 I2C 工作模式

FSMMOD0：默认状态。等待软件启动；

FSMMOD1：I2C 发送 START 命令，并等待 SCL 拉低；

FSMMOD2：I2C 发送地址或数据，等待 STOP 命令或 RESTART 命令，并检测 BUS ERROR；

FSMMOD3：等 RESTART 命令发送完成；

FSMMOD4：等 STOP 命令发送完成；

FSMMOD5：发送两个 SCL 时钟，用于解挂死（SDA 被从机拉低，主机无法发送 START 命令）；

FSMMOD6：等 RESTART 命令发送完成

第10章 SPI

10.1 特征

- Master和Slave模式，四线全双工（CS,SCK,MOSI,MISO）
- Master模式下，波特率有7种配置
- Slave模式下，最大支持SCK频率为系统时钟的1/8
- 支持SPI四种模式（极性，相位配置）
- Master模式下最大支持连接6个Slave
- 发送/接收字节比特顺序可配置
- SPI总线出错标志

10.2 模块框图

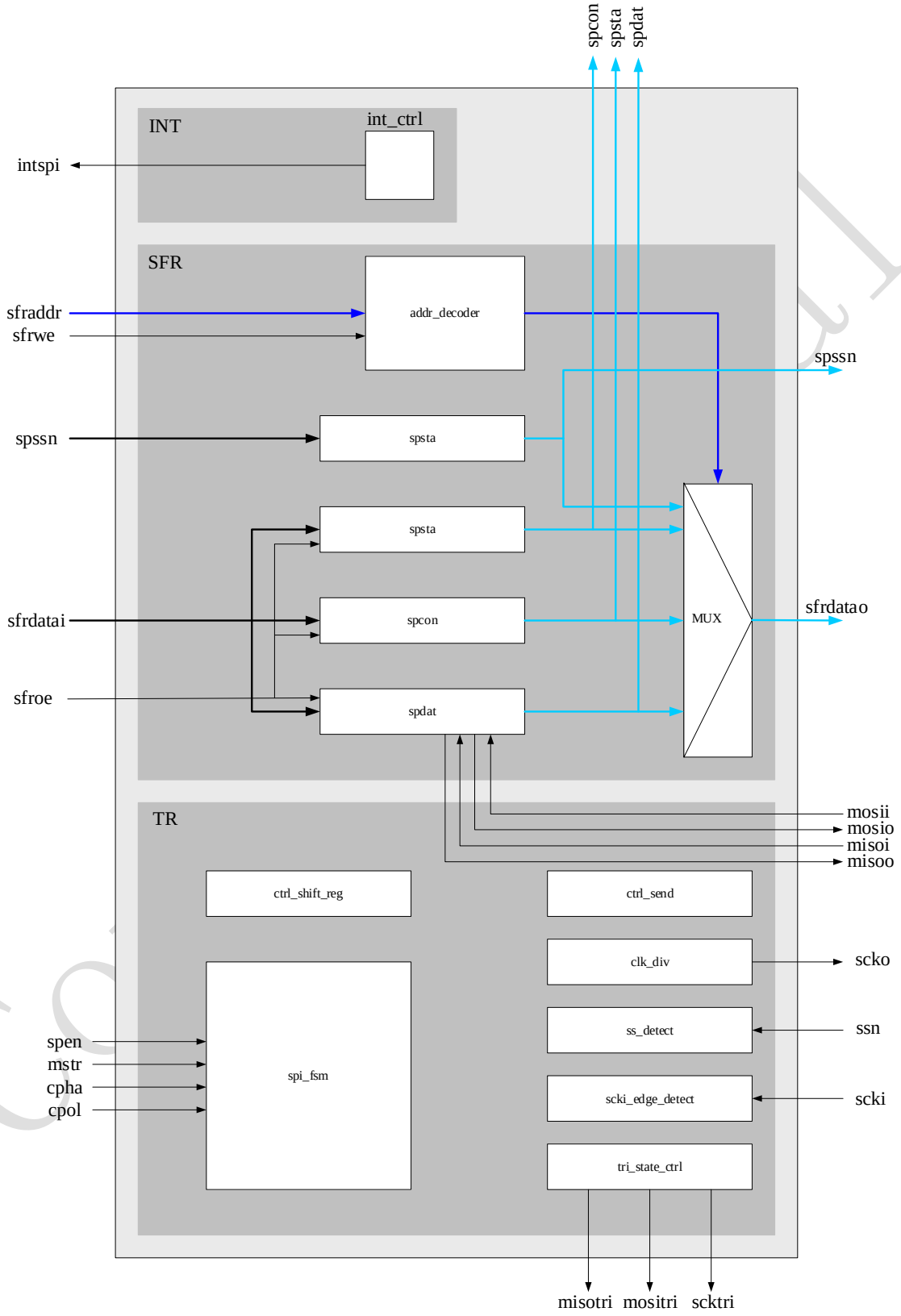


图 10-1　SPI 模块框图

10.3 功能描述

10.3.1 发送/接收接口

数据传输的一般格式如图 10-2 所示。根据配置，如果 cpol ^ cpha = 1, scko 上升沿驱动，下降沿采样；如果 cpol ^ cpha = 0, scko 下降沿驱动，上升沿采样。

如果“cpha”位被置位，第一位（MSB）将在“scko”的第一个有效时钟边沿，通过“mosio”/“misoo”发送出去。

如果“cpha”位被清零，第一位（MSB）将在“scko”的第一个有效时钟边沿之前的半个周期发送出去。此外，数据输入（“misoi”代表主机，“mosii”代表从机）在传输的每一位的一半进行采样，在时钟的相反边沿，数据从“mosio”输出。

这适用于主机或是从机的发送/接收，其中“scko”是传输的主时钟。

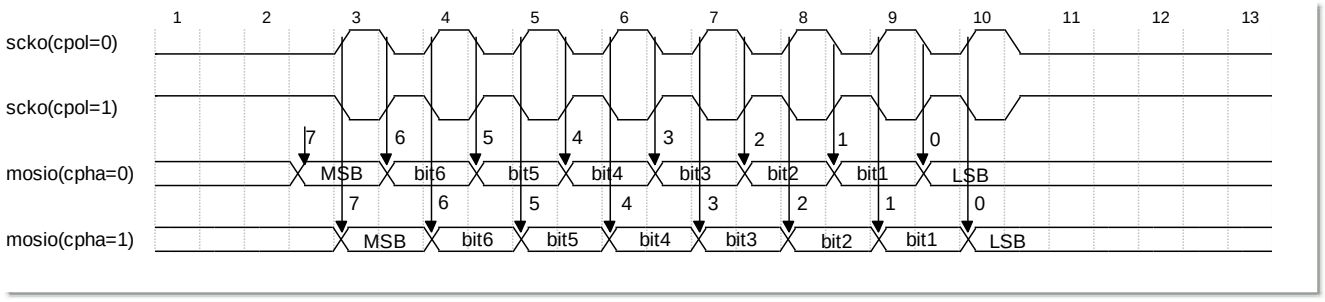


图 10-2　SPI 的发送帧格式

其中，发送/接收接口包含四个输入端口和六个输出端口：

输入：scki-串行时钟；ssn-从机片选；misoi-主机输入从机输出；mosii-主机输出从机输入

输出：scko-串行时钟；scktri-sck 端口三态 BUF 使能；misoo-主机输入从机输出；misotri-miso 端口的三态 BUF 使能；mosio-主机输出从机输入；mositri-mosi 端口的三态 BUF 使能

10.3.2 主机模式

在主机模式下（通过配置“spcon”寄存器的“mstr”位），SPI 等待对“spdat”寄存器的写操作。如果对“spdat”寄存器的写操作完成，则开始数据传输。数据“mosio”在时钟端口“scko”（发送边沿）上依次移出。同时，数据的接收从“misoi”引脚（捕获边沿）上依次移入（来自于从机）。

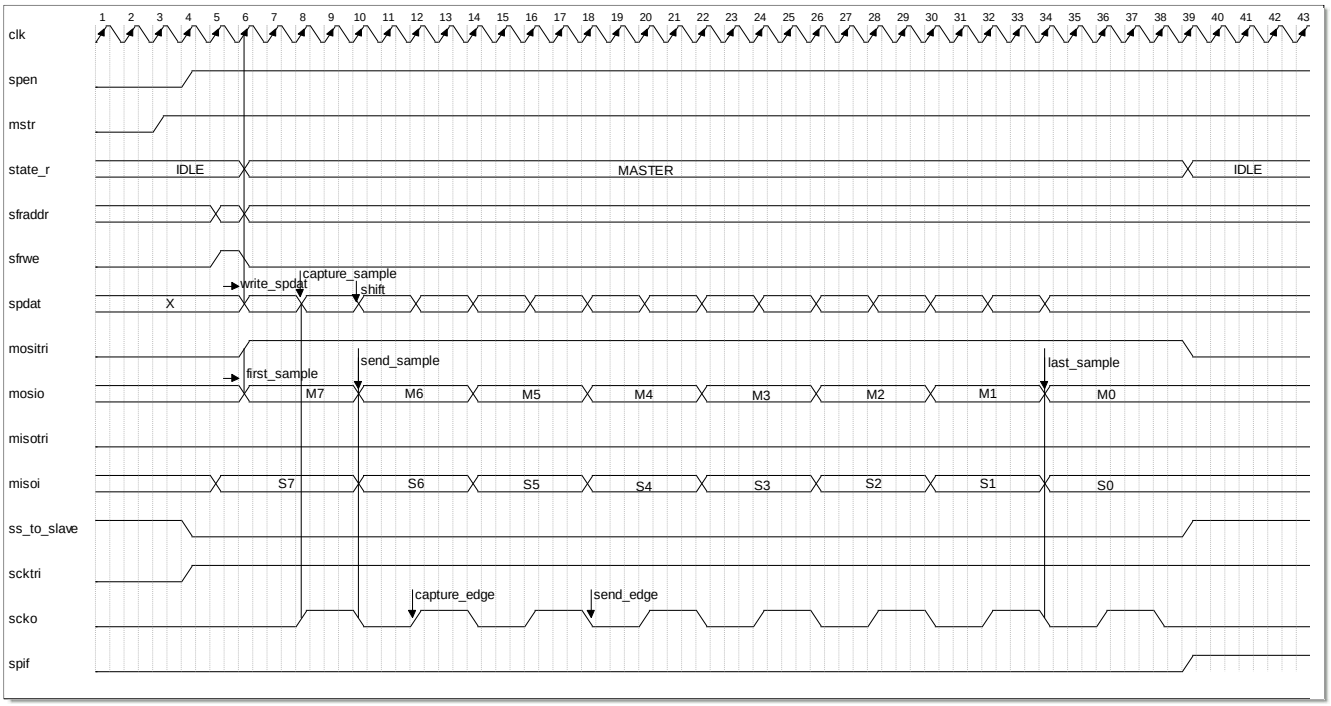


图 10-3 主机模式下数据传输格式（cpha='0',cpol='0'）

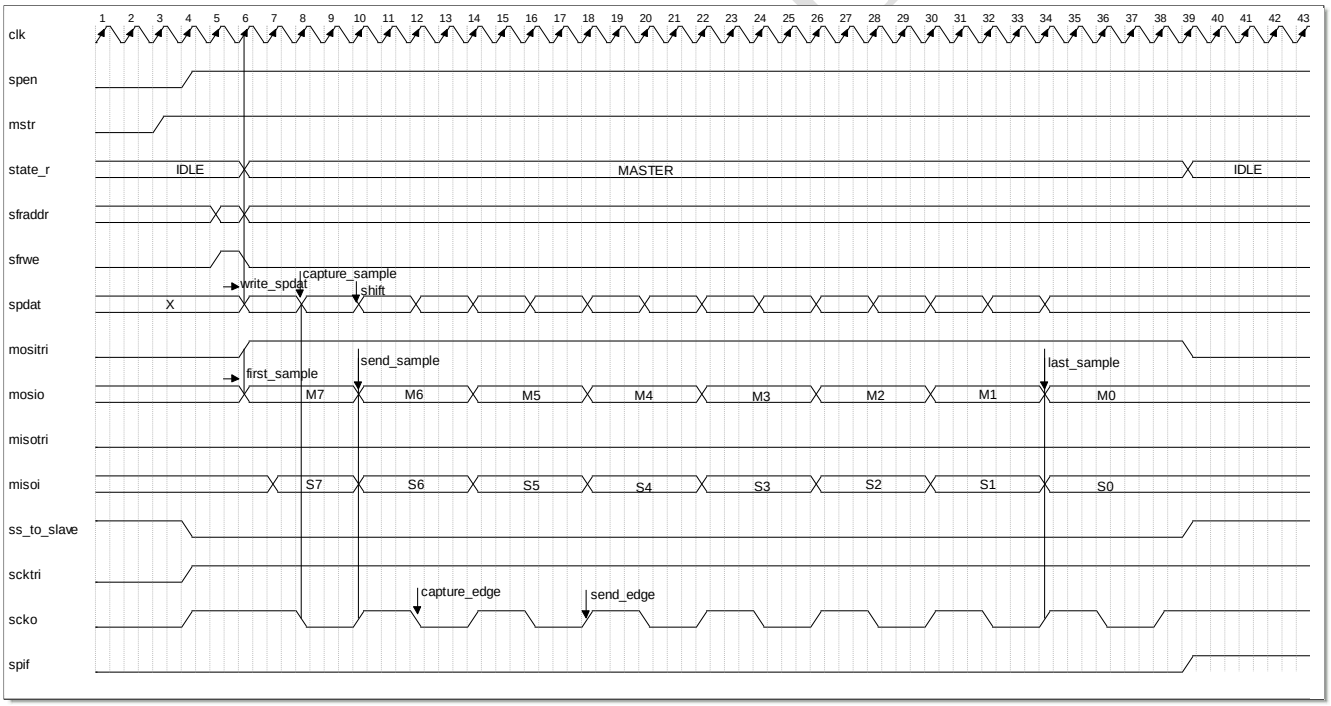


图 10-4 主机模式下数据传输格式（cpha='0',cpol='1'）

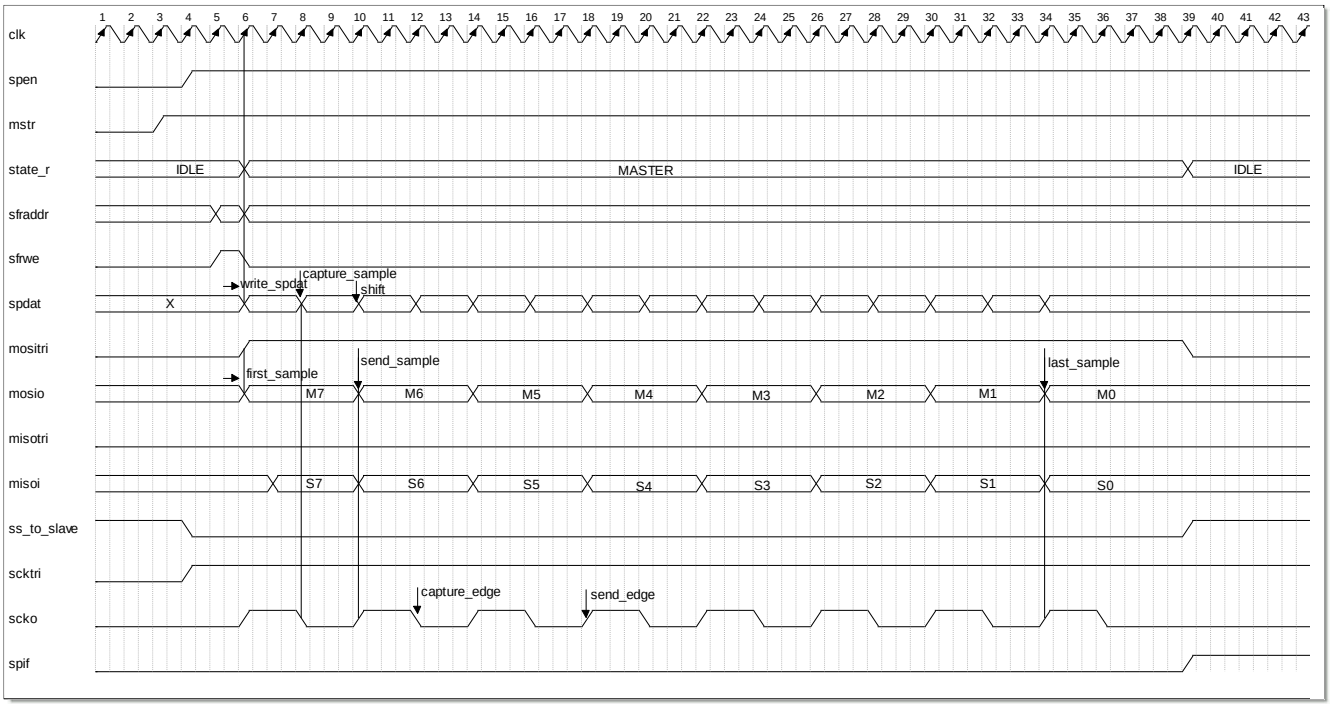


图 10-5 主机模式下数据传输格式（cpha=’1’,cpol=’0’）

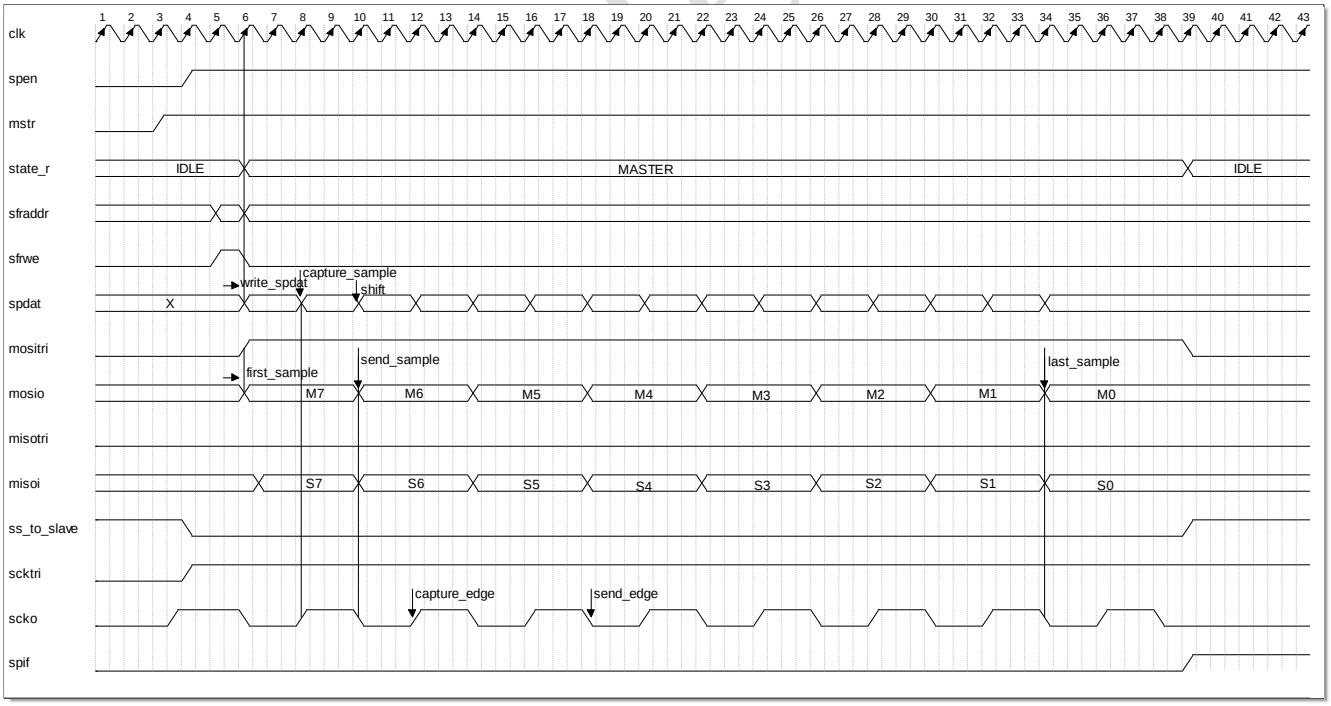


图 10-6 主机模式下数据传输格式（cpha=’1’,cpol=’1’）

10.3.3 从机模式

首先，必须通过在“spcon”寄存器中写入“mstr”=0，将 SPI 配置为从机。然后必须通过设置“spen”=1 来使能。图 10-7 显示了从模式下的数据传输过程。其配置如下：“cpha”=“1”，“cpol”=“1”，波特率为 f/4（位 spcon(7)、spcon(1)、spcon(0)在此模式下不起作用）。

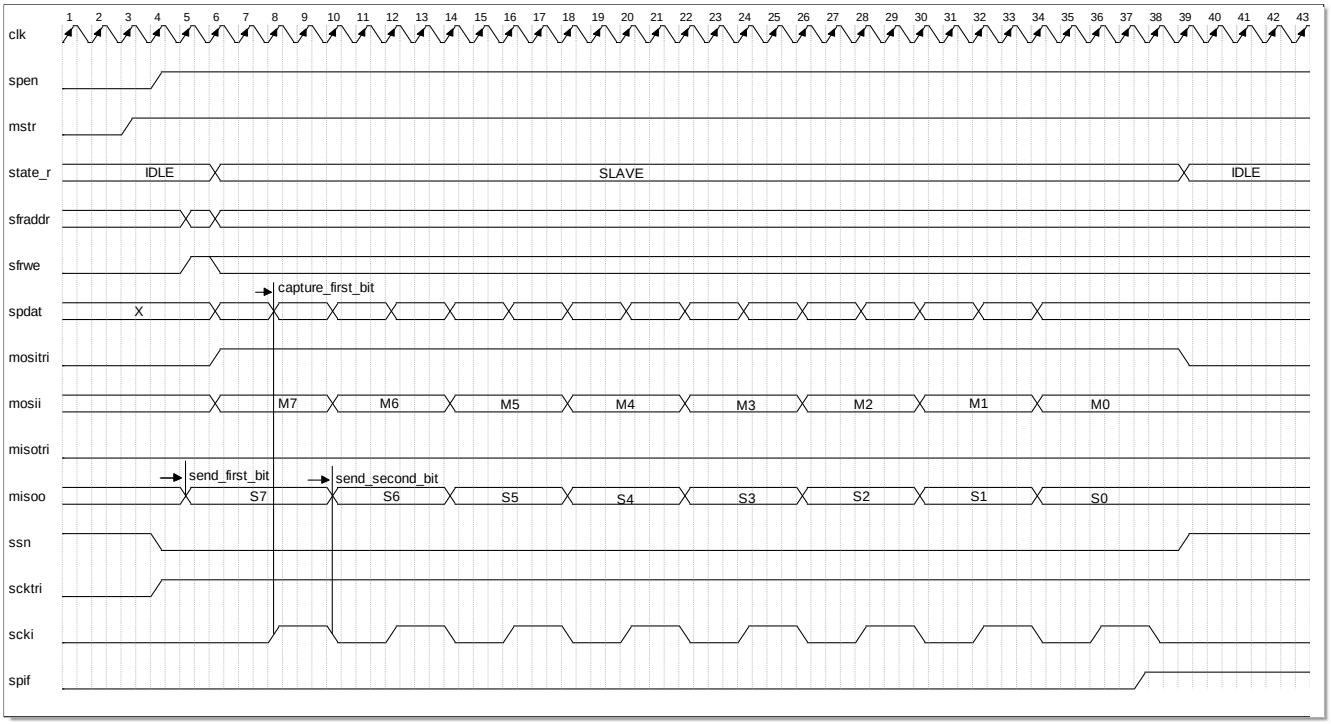


图 10-7 从机模式下数据传输格式

在从机模式下，SPI 等待“ssn”输入低电平。在传输完成之前，“ssn”输入必须保持为低。传输的开始取决于 SPCON 寄存器的“cpha”位的状态（图 10-7、图 10-8）。

当“cpha”被清零时，从机必须在第一个“scki”时钟边沿之前就需要准备好数据，输入片选信号“ssn”的下降边沿用于触发数据传输。当“cpha”置位时，从机使用“scki”的第一个时钟边沿用于触发数据传输。

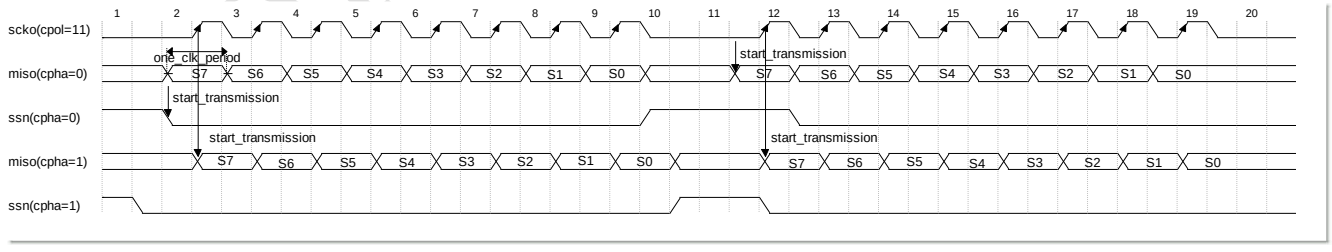


图 10-8 从机模式下数据传输格式（cpha=’0’,cpol=’1’）

10.3.4 中断

SPI 具有中断端口“intspi”，其在两种情况下会产生中断，如表 10-1 所示：

表 10-1 SPI 中断标志

名称	描述
“spif”	当数据传输完成，该位将由硬件置位
“modf”	当“ssn”输入与当前模式冲突时，该位将被置位（当在主机模式时，设置“ssdis”=“0”）

图 10-9 显示了由“spif”标志引起的中断请求。当传输结束时，“spif”标志由硬件自动置位。

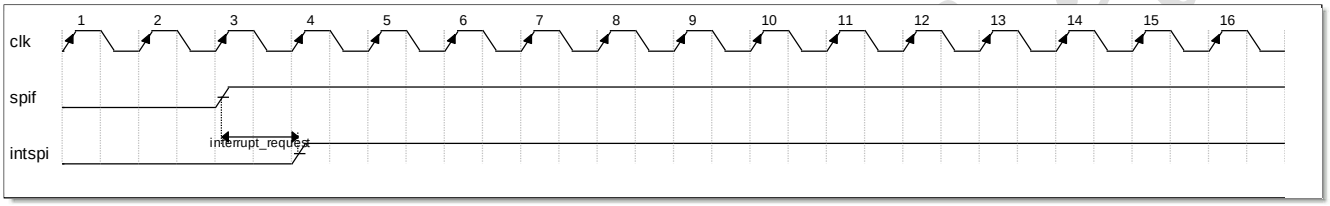


图 10-9 由 spif 触发的中断请求

图 10-10 显示了由“modf”标志引起的中断请求。当“ssn”输入的电平与所选操作模式不一致时（当 SPI 模块配置为主机，在“ssn”输入处检测到低电平），硬件自动置位“modf”标志，“ssdis”标志被清零。

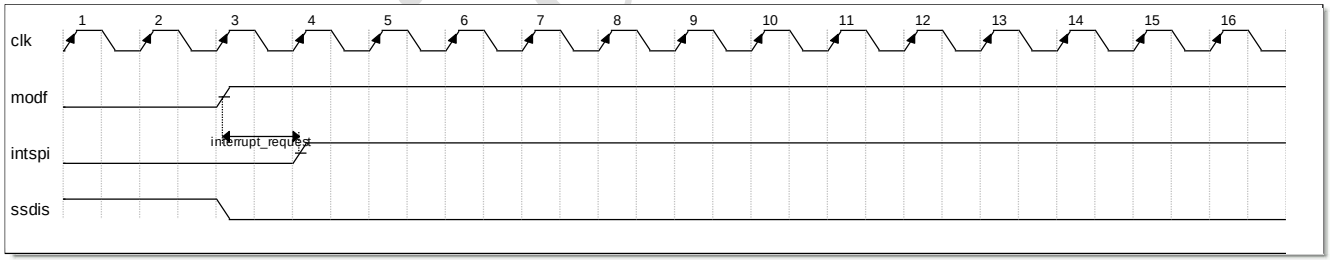


图 10-10 由 modf 触发的中断请求

图 10-11 显示了在“ssdis”置位时，即使检测到“modf”为高时，也不会产生中断。

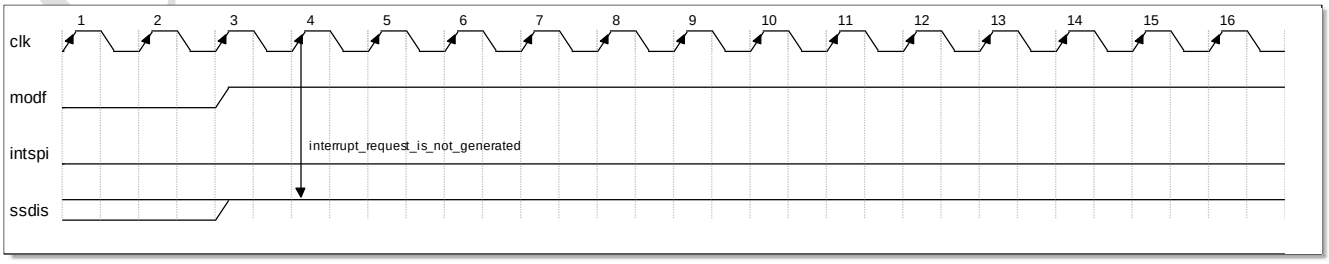


图 10-11 不会由 modf 触发的中断请求

当“spif”和“modf”均为零时，没有中断产生。为了确定 SPI 中断的实际来源，在中断服务程序中需要查询“modf”和“spif”标志。

10.3.5 scki 同步

输入时钟“scki”由两个在不同时钟边沿触发的 D 触发器进行同步。这些触发器之间的连接（在边沿检测逻辑之前）如下图所示。

之所以使用负边沿 D 触发器（“scki_u fall_u FF”），是因为从“scki”输入到“misoo”输出的时间对于保持协议的高速运行至关重要。在两个 D 触发器同步的情况下，从机模式下的“sck”最大速率为 Fclkper/8。

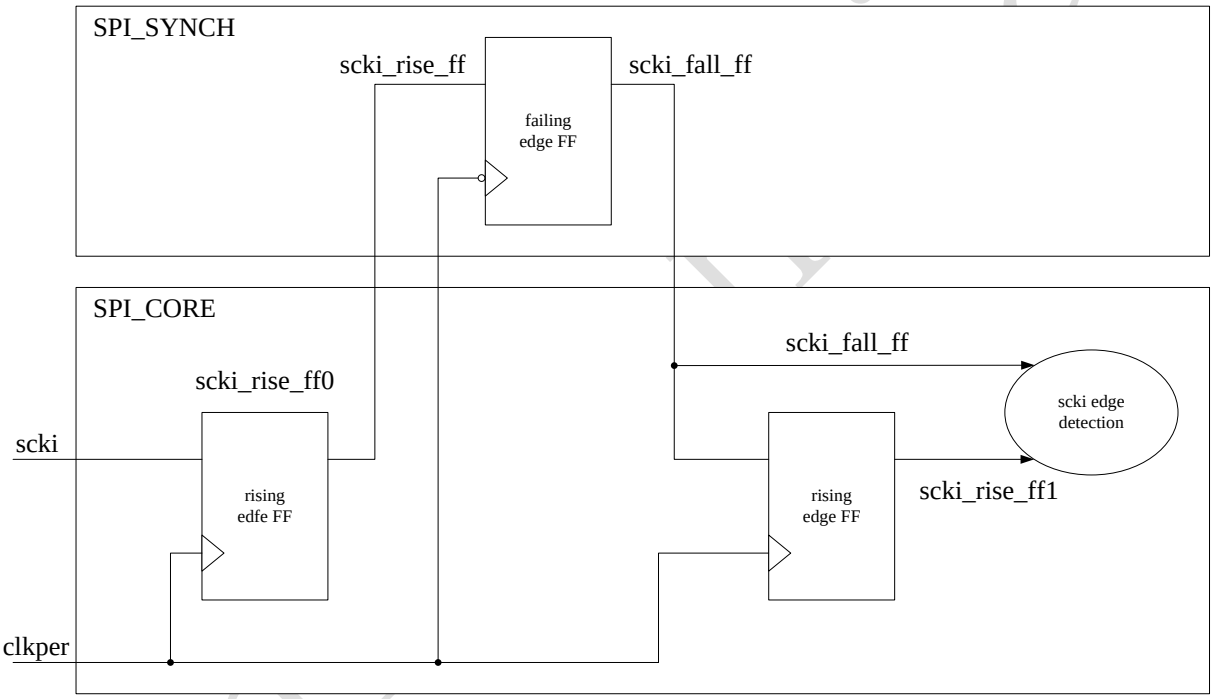


图 10-12“scki”同步寄存器

10.4 软件接口

10.5 寄存器列表

Register	offset	R/W	Description	Reset Value
SPSTA	0xE1	R	SPI 状态寄存器	0x00
SPCON	0xE2	R/W	SPI 控制寄存器	0x14
SPDAT	0xE3	R/W	SPI 数据寄存器	0x00
SPSSN	0xE4	R/W	SPI 从机选择寄存器	0xFF
SPCON1	0xE5	R/W	SPI 控制寄存器 1	0x00

10.6 寄存器描述

10.6.1 SPI Serial Peripheral Status Register(SPSTA)

Register	offset	R/W	Description	Reset Value
SPSTA	0x1	R	SPI 状态寄存器	0x00

Bits	Symbol	Type	Description
7	spif	R	数据传输标志。数据传输完成后该位置 1。软件读 SPSTA 寄存器会清除此标志位。 该位置 1 后会触发中断
6	wcol	R	写冲突标志。当 SPI 总线上正在传输数据，SPDAT 寄存器又被写入新的数据时，该位置 1。软件读 SPSTA 寄存器，并读 SPDAT 寄存器会清除此标志位。 该位置 1 后不会触发中断
5	sserr	R	从机错误标志。当 SPI 作为 slave 模式正在接收数据时，输入 SS 线拉高，该位置 1。通过禁用 SPI 使能信号清除此标志位。若 ssdis 置 1，该位不起作用。 该位置 1 后不会触发中断
4	modf	R	模式错误标志。当 SPI 作为 master 模式，输入 CS 线拉低，该位置 1。该位置 1 后，SPEN， MSTR 寄存器自动清零。软件读 SPSTA 寄存器，并写一次 SPCON 寄存器清除此标志位。 该位置 1 后，会触发中断；若 ssdis 置 1，modf 不会置 1，且不会触发中断。
[3:0]			reserved

10.6.2 SPI Serial Peripheral Control Register(SPCON)

Register	offset	R/W	Description	Reset Value
SPCON	0x2	R/W	SPI 控制寄存器。该寄存器用来选择 Master 时钟，选择 Master/Slave 模式，选择 Master 时钟极性和相位，使能/禁用 SPI。	0x14

Bits	Symbol	Type	Description
7	spr2	R/W	与 spr1/0 配合使用
6	spen	R/W	SPI 模块使能信号。 1:模块使能 0：模块禁用
5	ssdis	R/W	输入 SS 信号控制 1：SS 输入信号通道关闭（master 或 slave 模式） 0：SS 输入信号通道打开（master 或 slave 模式） 在 slave 模式下，如果 cpha=0，该位不起作用。 如果该位置 1，modf 位将不会被置 1，也不会触发中断
4	mstr	R/W	模式控制。 1：SPI 配置成 master 模式 0：SPI 配置成 slave 模式 modf 置 1 后，该位会被清掉
3	cpol	R/W	时钟极性控制。 1：默认情况下，时钟为高电平； 0：默认情况下，时钟为低电平
2	cpha	R/W	时钟相位控制 1：TX 在第一个有效边沿发送数据，RX 在第二个有效边沿接收数据； 0：TX 在第一个有效边沿之前发送数据，RX 在第一个有效边沿接收数据。
[1:0]	spr1/0	R/W	SPI 做 Master 时。时钟分频配置。见下图

表 10-2 Master 时钟分频

spr2:0	波特率
000	2 分频
001	4 分频
010	8 分频
011	16 分频
100	32 分频
101	64 分频
110	128 分频
111	无时钟产生（sck 端口极性取决 cpol 的配置）

10.6.3 SPI Serial Peripheral Data Register(SPDAT)

Register	offset	R/W	Description	Reset Value
SPDAT	0x3	R/W	SPI 读写缓存	0x00

Bits	Symbol	Type	Description
[7:0]	spdat	R/W	向该寄存器写入数据后，数据会直接传输给移位寄存器，输出到数据线上。 读该寄存器会返回读缓存中的数据。

10.6.4 SPI Serial Peripheral Slave Register(SPSSN)

Register	offset	R/W	Description	Reset Value
SPSSN	0x4	R/W	SPI 外设从机选择	0xFF

Bits	Symbol	Type	Description
[7:0]	spssn	R/W	控制 SS[7:0]端口的输出，用于选择从机，最大支持 8 个从机，由于 GPIO 分配，仅支持使用 spssn[6:5]和 sspn[3:0]

10.6.5 SPI Serial Peripheral Control Register1(SPCON1)

Register	offset	R/W	Description	Reset Value
SPCON1	0x5	R/W	控制发送数据顺序	0x00

Bits	Symbol	Type	Description
[7:1]			reserved
[0]	endian	R/W	控制发送数据顺序。 1：LSB 先，MSB 后； 0：MSB 先，LSB 后

第11章 TIMER

PAN262x 中包含 3 个 Timer，分别为 Timer0、Timer1、Timer2。每个 Timer 均可配置为定时器方式，又可用作计数器方式使用。每个 Timer 均具有独立定时/计数寄存器，分别为 Timer0 的 TL0 与 TH0；Timer1 的 TL1 与 TH1；Timer2 的 TL2 与 TH2。

11.1 特征

- 13位/16位的计数器0或是定时器0
- 自动重载的8位定时器0或是计数器0
- 双8位计数器0或是定时器0
- 13位/16位的计数器1或是定时器1
- 自动重载的8位定时器1或是计数器1
- 16位的计数器2或是定时器2
- 16位门控定时器2；
- 定时器2/计数器2具有自动重载模式
- 计数器2具有捕获计数模式
- 均具有溢出中断

11.2 功能描述

11.2.1 Timer0

当 Timer0 配置成定时器模式，Timer0 每隔 12 个时钟周期递增一次。

当 Timer0 配置成计数器模式，当检测到“t0”pin 脚的下降沿时，计数器将加 1。由于其需要两个时钟周期去检测边沿跳变，因此输入的最大计数频率为系统时钟的一半。对于占空比没有要求，但是为了确保能够正常检测到 0 或是 1 状态，其高/低电平至少要保持 1 个时钟周期。

（1）模式 0 与模式 1

在模式 0 中，Timer0 可以被配置成 13 位的定时器/计数器（“tl0”=5 bits,“th0”=8bits）。“tl0”的高 3 位将不会被使用，并且其中的数值将会保持不变。

当“tmod[1:0]”=“00”时，即配置成模式 0。在该模式下，其计数的时钟源头不是内部时钟（定时器）就是外部 pin 脚（计数器）。且在 tmod[2]置 1，其将配置成计数器模式，否则是定时器模式。定时器/计数器分为两个 8 位寄存器，低字节和高字节。在这种模式下，低字节被另外分成两部分，由低 5 位和高 3 位组成（只有低 5 位是计数器的一部分，高 3 位未被使用）。Timer0 将每 12 个时钟周期，或是外部 pin 脚下降沿时，进行递增。当定时器 0/计数器 0 溢出时，tcon[5]将被置位，并产生中断（硬件自动清中断）。在计数模式下，该可配置成门控计数器（tmod[3]需要开启门控使能），在“int0”为高时，计数使能。

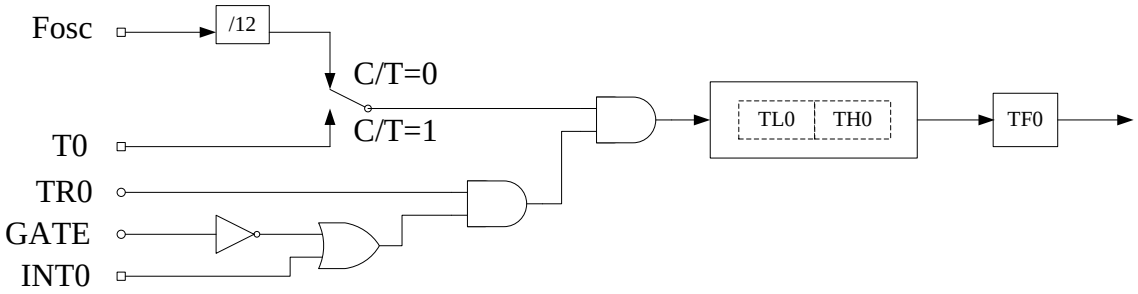


图 11-1 Timer0 模式 0 与模式 1

当“tmod[1:0]”=“01”时，即配置成模式 1。在该模式下，其将作为 16 位的定时器/计数器。

（2）模式 2

在模式 2，Timer0 具有自动重载 8 位定时器/计数器。

当“tmod[1:0]”=“10”时，即配置成模式 2。在该模式下，其计数的时钟源头不是内部时钟（定时器）就是外部 pin 脚（计数器）。且在 tmod[2]置 1，其将配置成计数器模式，否则是定时器模式。Timer0 将每 12 个时钟周期，或是外部 pin 脚下降沿时，进行递增。在该模式下，其是自动重载的定时器/计数器，当 TL0 溢出时，并产生中断（硬件自动清中断），寄存器 TH0 的数值会自动加载到 TL0 寄存器中。在计数模式下，该可配置成门控计数器（tmod[3]需要开启门控使能），在“int0”为高时，计数使能。

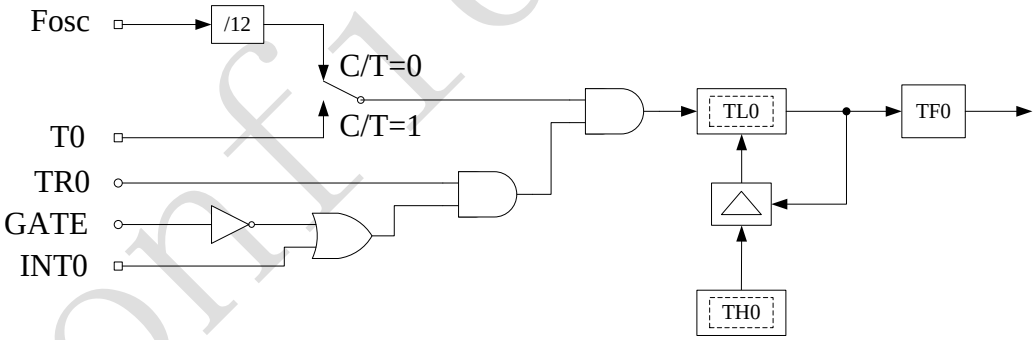


图 11-2 Timer0 模式 2

（3）模式 3

在模式 3，Timer0 可以被配置成一个 8 位定时器/计数器和一个定时器。当 Timer0 工作在模式 3，Timer1 依然可以被用做 Uart 的波特率发生器，或是不需要使用中断的功能。

当“tmod[1:0]”=“11”时，即配置成模式 3。在该模式下，TL0 其计数的时钟源头不是内部时钟（定时器）就是外部 pin 脚（计数器）；TL0 的计数源头只能是外部 pin 脚（计数器）。且在 tmod[2]置 1，其将配置成计数器模式，否则是定时器模式。当 TL0 溢出时，tcon[5]将被置位，并产生中断（硬件自动清中断）。当 TH0 溢出时，tcon[7]将被置位，并产生中断（硬件自动清中

断）。TL0 在计数模式下，该可配置成门控计数器（tmod[3]需要开启门控使能），在“int0”为高时，计数使能。

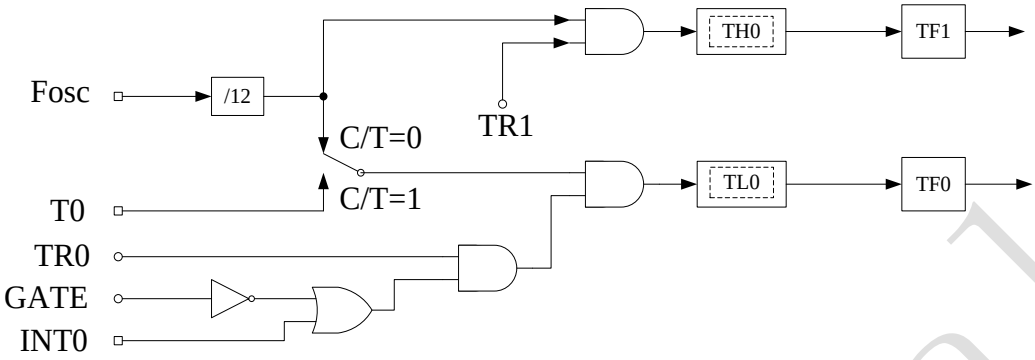


图 11-3 Timer0 模式 3

11.2.2 Timer1

Timer1 可配置成 13 位或是 16 位的定时器或是计数器，以及自动重载的 8 位定时器或是计数器。

当 Timer1 配置成定时器模式，Timer1 每隔 12 个时钟周期递增一次。

当 Timer1 配置成计数器模式，当检测到“t1”pin 脚的下降沿时，计数器将加 1。由于其需要两个时钟周期去检测边沿跳变，因此输入的最大计数频率为系统时钟的一半。对于占空比没有要求，但是为了确保能够正常检测到 0 或是 1 状态，其高/低电平至少要保持 1 个时钟周期。

（1）模式 0 与模式 1

在模式 0 中，Timer1 是 13 位的定时器或计数器（“tl1”=5bits“th1”=8bits），“tl1”高 3 位没有被使用，并保持不变。

在模式 0 中，Timer1 可以被配置成 13 为的定时器/计数器（“tl1”=5 bits,“th1”=8bits）。“tl1”的高 3bit 将不会被使用，并且其中的数值将会保持不变。

当“tmod[5:4]”=“00”时，即配置成模式 0。在该模式下，其计数的时钟源头不是内部时钟（定时器）就是外部 pin 脚（计数器）。且在 tmod[6]置 1，其将配置成计数器模式，否则是定时器模式。定时器/计数器分为两个 8 位寄存器，低字节和高字节。在这种模式下，低字节被另外分成两部分，由低 5 位和高 3 位组成（只有低 5 位是计数器的一部分，高 3 位未被使用）。Timer1 将每 12 个时钟周期，或是外部 pin 脚下降沿时，进行递增。当定时器 1/计数器 1 溢出时，tcon[7]将被置位，并产生中断（硬件自动清中断）。在计数模式下，该可配置成门控计数器（tmod[7]需要开启门控使能），在“int1”为高时，计数开始工作。

在模式 1 中，Timer1 是 16 位的定时器或是计数器。当“tmod[5:4]”=“01”时，其配置成模式 1。

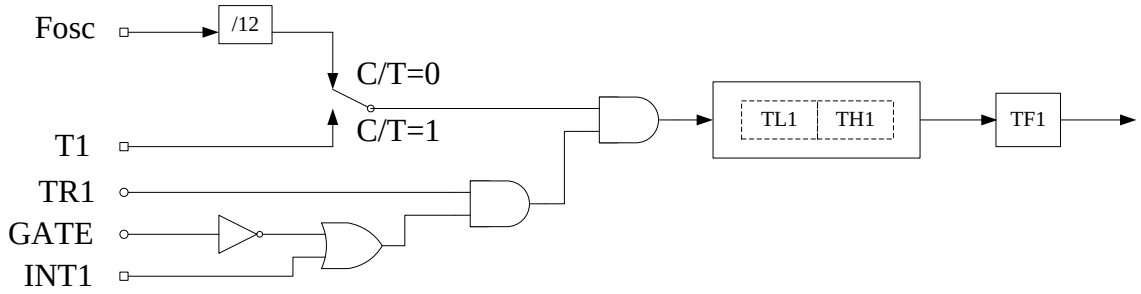


图 11-4 Timer1 模式 0 与模式 1

（2）模式 2

在模式 2 中的，Timer1 是自动重载的 8 位定时器或是计数器。

当“tmod[1:0]”=“10”时，即配置成模式 2。在该模式下，其计数的时钟源头不是内部时钟（定时器）就是外部 pin 脚（计数器）。且在 tmod[6]置 1，其将配置成计数器模式，否则是定时器模式。Timer1 将每 12 个时钟周期，或是外部 pin 脚下降沿时，进行递增。在该模式下，其是自动重载的定时器/计数器，当 TL1 溢出时，tcon[7]将被置位，并产生中断（硬件自动清中断），寄存器 TH1 的数值会自动加载到 TL1 寄存器中。在计数模式下，该可配置成门控计数器（tmod[3]需要开启门控使能），在“int0”为高时，计数使能。置位 tcon[6]，将使能 Timer1。

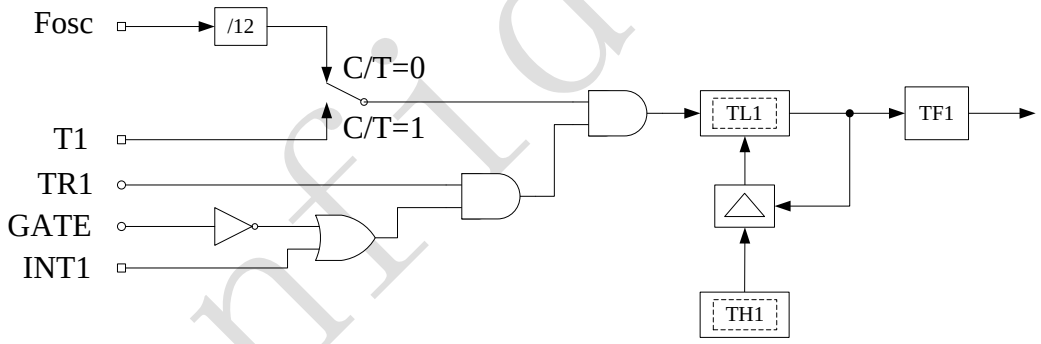


图 11-5 Timer1 模式 2

11.2.3 Timer2

Timer2 可以配置成定时器、计数器，并具有比较与捕获功能。

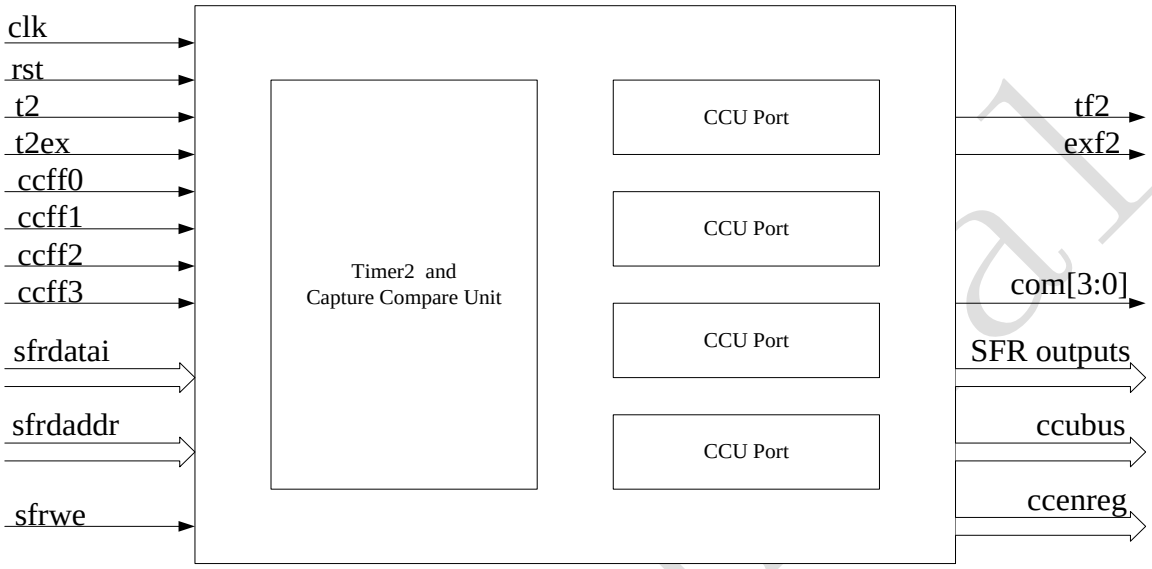


图 11-6 Timer2 框图 0

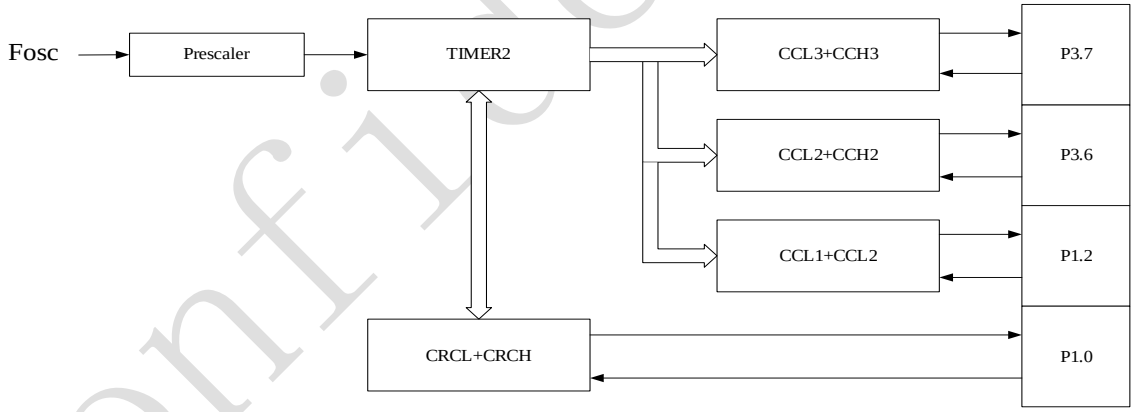


图 11-7 Timer2 框图 1

11.2.3.1 定时计数模式

Time2 可以配置成定时器、计数器、与门控定时器。

（1）定时器模式

当寄存器 t2con 的“t2i0=1”“t2i1=0”时，Timer2 进入定时器模式。定时器以每 12 个或是 24 的时钟周期进行递增。当 t2con 的“t2ps=0”时，递增是以 12 个时钟周期，否则“t2ps=1”是以 24 个时钟周期进行递增。

（2）计数器模式

当寄存器 t2con 的“t2i0=0”“t2i1=1”时，Timer2 进入计数器模式。当检测到“t2”pin 脚的下降沿时，计数器将加 1。由于其需要两个时钟周期去检测边沿跳变，因此输入的最大计数频率为系统时钟（16M）的一半。

（3）门控定时器模式

当寄存器 t2con 的“t2i0=1”“t2i1=1”时，Timer2 进入定时器模式。定时器以每 12 个或是 24 的时钟周期进行递增（取决于 t2ps 的配置）。此外，其有外部外部 pin 脚“t2”门控，当“t2=0”定时器停止工作，否则开启工作。

（4）Reload 模式

16 位“crc”寄存器重载有两种模式，一种是在 Timer2 溢出的时候，进行重载；另一种，由外部 pin 脚“t2ex”的下降沿触发重载。

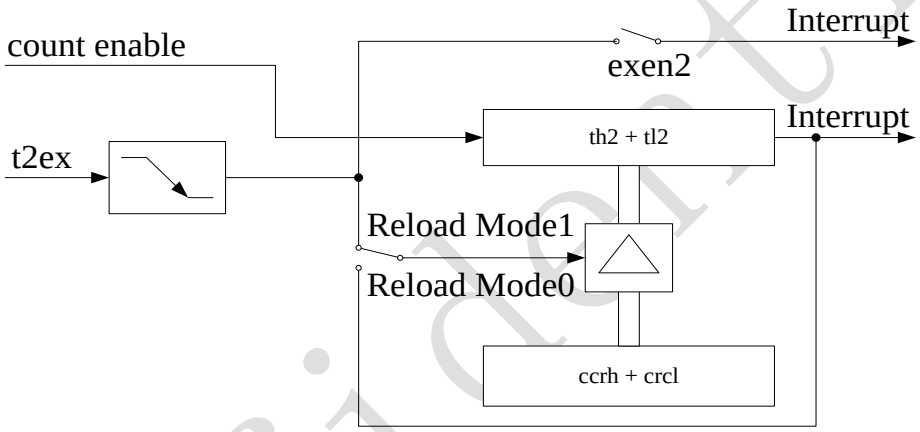


图 11-8 Timer2 的 reload 模式

11.2.3.2 比较模式

比较/捕获单元由四个寄存器组成：“cc1”、“cc2”、“cc3”和“crc”。每个寄存器都可以配置为在比较器模式下工作。在这个模式将寄存器中存储的值与定时器 2 的内容进行比较。该比较器驱动“ccubus”输出，且相应的输出端口为“p1.0”“p1.2”“p3.6”“p3.7”：

①“p1.0”是与寄存器“crc”（“ccubus.0”）相关联的比较器的输出

②“p1.2”是与寄存器“cc1”（“ccubus.1”）相关联的比较器的输出

③“p3.6”是与寄存器“cc2”（“ccubus.2”）相关联的比较器的输出

④“p3.7”是与寄存器“cc3”（“ccubus.3”）相关联的比较器的输出

“t2con”寄存器中的位“t2cm”选择了两种比较模式。

（1）比较模式 0

当“t2con”寄存器的位“t2cm”=0，即进入比较模式 0。在模式 0 中，当定时器 2 中的值等于比

较寄存器的值时，比较器输出从低变高。它在定时器 2 溢出时变低。在这种模式下，写入端口 1（“p1”）对“ccubus”没有影响，因为来自内部总线的输入线和写入寄存器线断开。下图说明了比较模式 0 的功能。

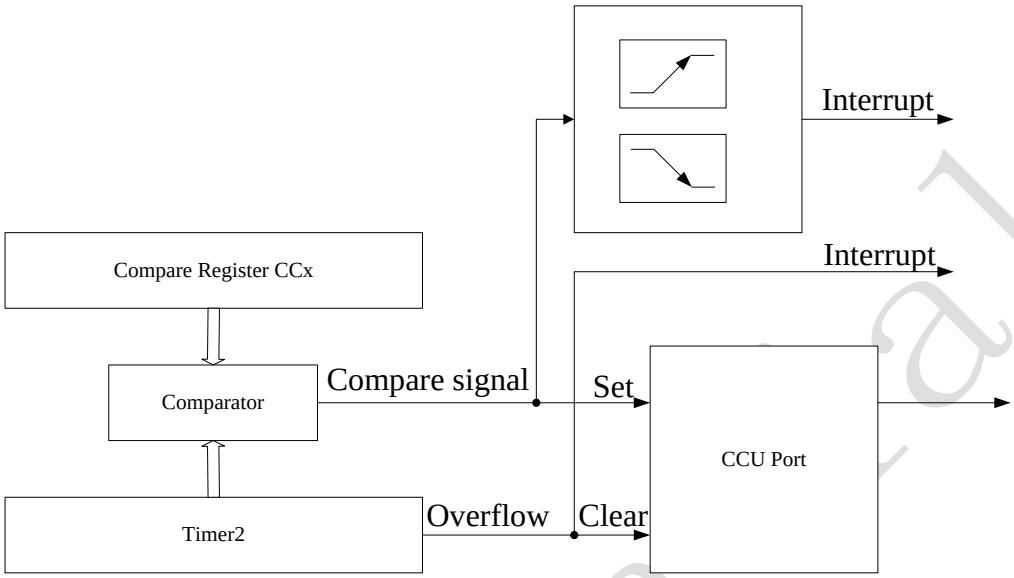


图 11-9 Timer2 的比较模式 0

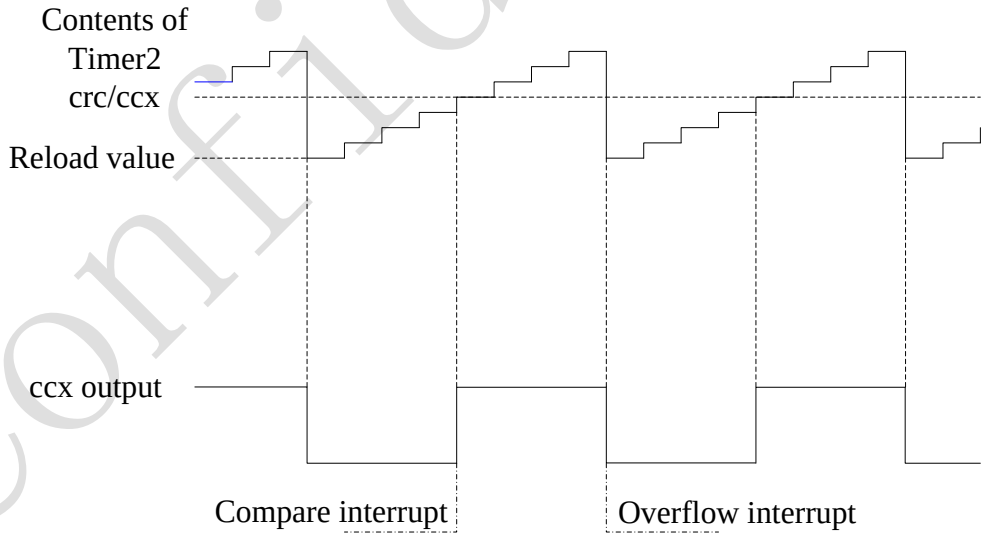


图 11-10 Timer2 比较模式 0 的输出

（3）比较模式 1

当“t2con”寄存器的位“t2cm”=1，即进入比较模式 1。在比较模式 1 中，输出信号的转换

可由软件确定。定时器 2 溢出不会导致输出变化。在这种模式下，输出信号的两种转换都可以被控制。下图显示了比较模式 1 的寄存器，以及端口输出的功能图。

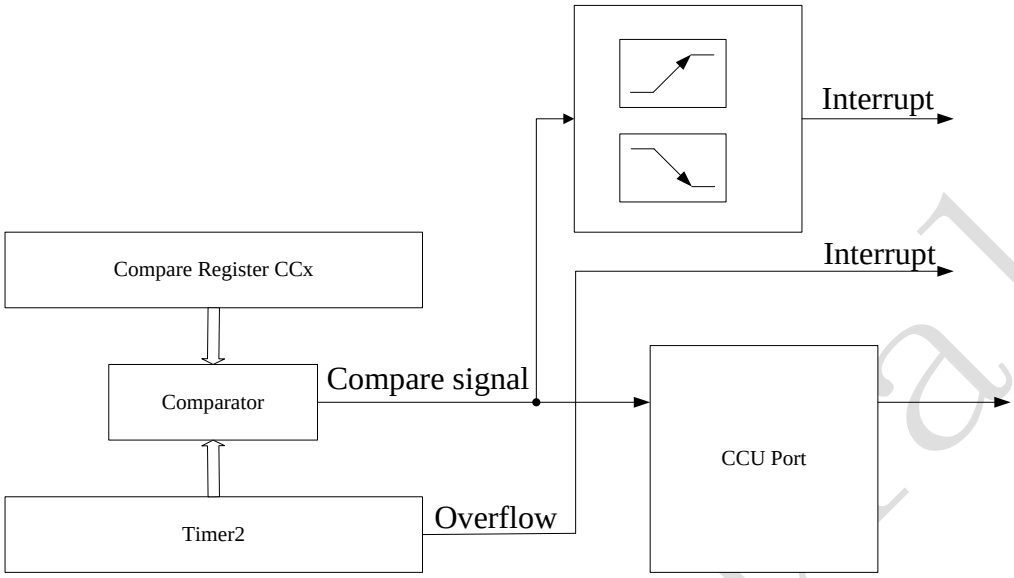


图 11-11 Timer2 的比较模式 1

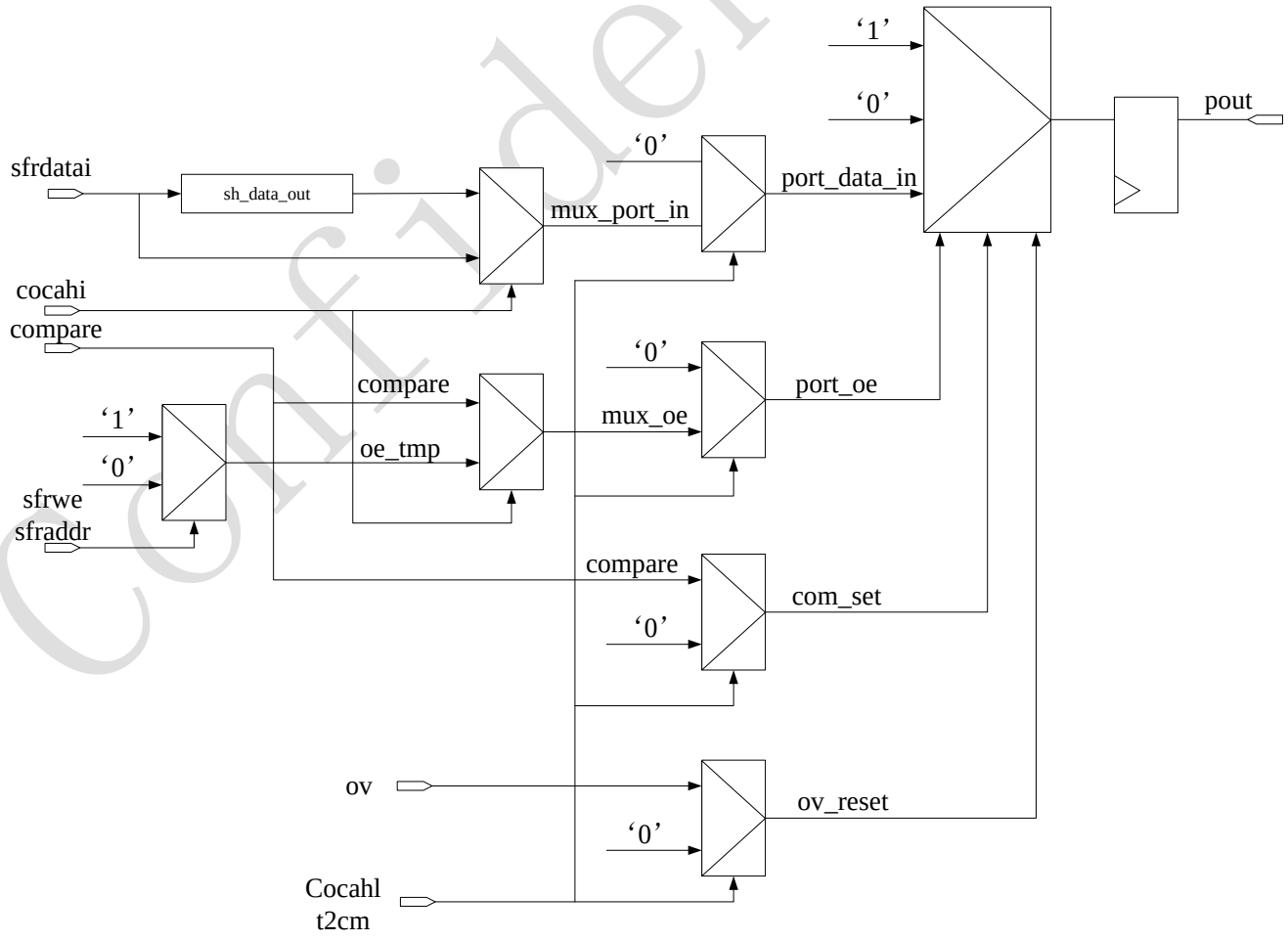


图 11-12 比较单元的端口框图

11.2.3.3 捕获模式

四个 16 位 CCU 寄存器中的每一个都可以配置为在捕获模式下工作。在这种模式下，实际的定时器/计数器内容被保存到外部时钟上的 CCU 寄存器中事件（模式 0）或软件写入操作（模式 1）。

（1）捕获模式 0

在模式 0 中，在以下情况下执行定时器 2 内容的捕获：

①在输入“cc0”上检测到上升沿（或是下降沿或是 both，可配置）；

②在输入“cc1”上检测到上升沿（或是下降沿或是 both，可配置）；

③在输入“cc2”上检测到上升沿（或是下降沿或是 both，可配置）；

④在输入“cc3”上检测到上升沿（或是下降沿或是 both，可配置）。

定时器 2 将上述内容锁存到捕获寄存器时，将产生捕获中断，其与溢出中断共用同一个中断向量，通过状态寄存器为区分。

（2）捕获模式 1

在模式 1 中，定时器 2 的捕获是由写入专用捕获寄存器的低位字节引起的。其写入捕获寄存器的值，之间不存在联系。定时器 2 的内容将被锁存到适当的捕获寄存器中。在此模式下，不具有捕获中断，只具有溢出中断，

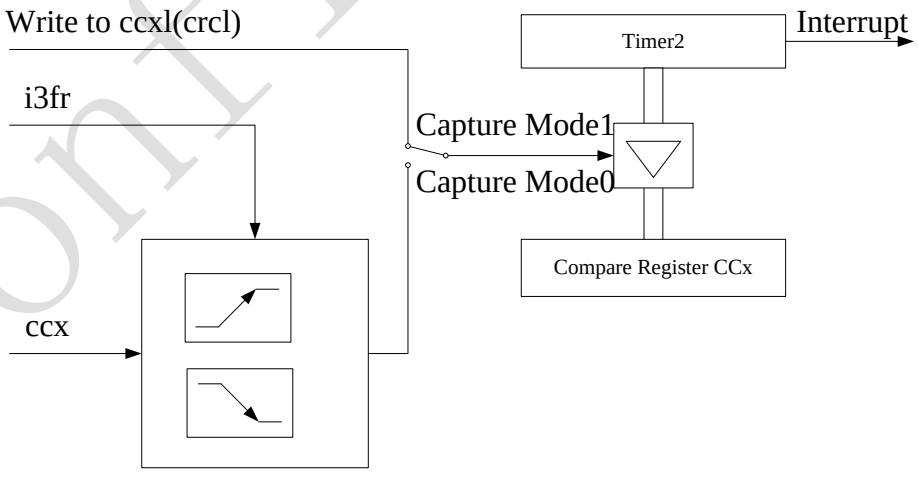


图 11-13 比较单元的端口框图

11.3 软件接口

11.4 寄存器列表

Register	offset	R/W	Description	Reset Value
TCON	0x88	R/W	定时器/计数器控制寄存器	0x00
TMOD	0x89	R/W	定时器模式寄存器	0x00
TL0	0x8a	R/W	定时器 0，低字节	0x00
TL1	0x8b	R/W	定时器 1，低字节	0x00
TH0	0x8c	R/W	定时器 0，高字节	0x00
TH1	0x8d	R/W	定时器 1，高字节	0x00
CCEN	0xc1	R/W	比较/捕获使能寄存器	0x00
CCL1	0xc2	R/W	比较/捕获寄存器 1，低字节	0x00
CCH1	0xc3	R/W	比较/捕获寄存器 1，高字节	0x00
CCL2	0xc4	R/W	比较/捕获寄存器 2，低字节	0x00
CCH2	0xc5	R/W	比较/捕获寄存器 2，高字节	0x00
CCL3	0xc6	R/W	比较/捕获寄存器 3，低字节	0x00
CCH3	0xc7	R/W	比较/捕获寄存器 3，高字节	0x00
T2CON	0xc8	R/W	定时器 2 控制寄存器	0x00
CRCL	0xca	R/W	比较/重载/捕获寄存器，低字节	0x00
CRCH	0xcb	R/W	比较/重载/捕获寄存器，高字节	0x00
TL2	0xcc	R/W	定时器 2，低字节	0x00
TH2	0xcd	R/W	定时器 2，高字节	0x00
ADCON	0xd8	R/W	串口波特率选择	0x00
TCAPCON	0xab	R/W	捕获配置寄存器	0x00
TCAPSTA	0xac	R/W	捕获状态寄存器	0x00

11.5 寄存器描述

11.5.1 Timer/Counter Control Register(TCON)

Register	offset	R/W	Description	Reset Value
TCON	0x88	R/W	定时器/计数器控制寄存器	0x00

Bits	Symbol	Type	Description
7	tf1	R/W	定时器 1 的溢出标志，该标志可通过软件清除，以及可以在中断函数中自动清除
6	tr1	R/W	定时器 1 的控制寄存器 0：不使能 1：使能

5	tf0	R/W	定时器 0 的溢出标志，该标志可通过软件清除，以及可以在中断函数中自动清除
4	tr0	R/W	定时器 0 的控制寄存器 0：不使能 1：使能
3	ie1	R/W	外部中断 1 标志，该标志通过硬件自动清除
2	it1	R/W	外部中断 1 控制 0：外部中断 1 低电平触发 1：外部中断 1 下降沿触发
1	ie0	R/W	外部中断 0 标志，该标志通过硬件自动清除
0	it0	R/W	外部中断 0 控制 0：外部中断 0 低电平触发 1：外部中断 0 下降沿触发

11.5.2 Timer Mode Register(TMOD)

Register	offset	R/W	Description	Reset Value
TMOD	0x89	R/W	定时器模式寄存器	0x00

Bits	Symbol	Type	Description
7	gate	R/W	timer1 门控使能 0：不使能 1：使能 当 int1 为高时，当 tr1 为高，计数器 1 在“t1”下降沿进行计数
6	c/t	R/W	timer1 计数器/定时器选择 0：定时器 1：计数器
5	m1	R/W	timer1 模式选择 00：模式 0，13 位计数器/定时器 01：模式 1，16 位技术器/定时器 10：模式 2，8 位自动重载计数器/定时器 11：模式 3，停止
4	m0	R/W	
3	gate	R/W	timer0 门控使能 0：不使能 1：使能 当 int0 为高时，当 tr0 为高，计数器 0 在“t0”下降沿进行计数
2	c/t	R/W	timer0 计数器/定时器选择 0：定时器

			1：计数器
1	m1	R/W	timer0 模式选择
0	m0	R/W	00：模式 0，13 位计数器/定时器 01：模式 1，16 位技术器/定时器 10：模式 2，8 位自动重载计数器/定时器 11：模式 3，双计数器/定时器

11.5.3 Timer0 Low Data Register (TL0)

Register	offset	R/W	Description	Reset Value
TL0	0x8a	R/W	Timer0，low byte	0x00
Bits	**Symbol**	**Type**	**Description**	
7:0	tl0	R/W	Timer0 低 8 位	

11.5.4 Timer1 Low Data Register(TL1)

Register	offset	R/W	Description	Reset Value
TL1	0x8b	R/W	Timer1，low byte	0x00
Bits	**Symbol**	**Type**	**Description**	
7:0	tl1	R/W	Timer1 低 8 位	

11.5.5 Timer0 High Data Register(TH0)

Register	offset	R/W	Description	Reset Value
TH0	0x8c	R/W	Timer0，high byte	0x00
Bits	**Symbol**	**Type**	**Description**	
7:0	th0	R/W	Timer0 高 8 位	

11.5.6 Timer1 High Data Register(TH1)

Register	offset	R/W	Description	Reset Value
TH1	0x8d	R/W	Timer1，high byte	0x00

Bits	Symbol	Type	Description
7:0	th1	R/W	Timer1 高 8 位

11.5.7 Interrupt Request Control Register(IRCON)

Register	offset	R/W	Description	Reset Value
IRCON	0xc0	R/W	Interrupt Request Control Register	0x00

Bits	Symbol	Type	Description
7	other function	R/W	
6	tf2	R/W	Timer2 interrupt flag，写 0 进行清除
5:0	other function		

11.5.8 Compare/Capture Enable Register(CCEN)

Register	offset	R/W	Description	Reset Value
CCEN	0xc1	R/W	比较/捕获寄存器	**0x00**

Bits	Symbol	Type	Description
7	cocah3	R/W	比较/捕获模式 00：比较/捕获不使能 01：pin cc0 边沿捕获使能 10：比较使能 11：在写入 cc3 时，进行捕获
6	cacal3	R/W	
5	cocah2	R/W	比较/捕获模式 00：比较/捕获不使能 01：pin cc1 边沿捕获使能 10：比较使能 11：在写入 ccl2 时，进行捕获
4	cacal2	R/W	
3	cocah1	R/W	比较/捕获模式 00：比较/捕获不使能 01：pin cc2 边沿捕获使能 10：比较使能 11：在写入 cc1 时，进行捕获
2	cacal1	R/W	
1	cocah0	R/W	比较/捕获模式 00：比较/捕获不使能 01：pin cc3 边沿捕获使能 10：比较使能 11：在写入 crcl 时，进行捕获
0	cacal0	R/W	

11.5.9 Compare/Capture Low Data Register1 (CCL1)

Register	offset	R/W	Description	Reset Value
CCL1	0xc2	R/W	CC1 比较捕获，low byte	0x00

Bits	Symbol	Type	Description
7:0	ccl1	R/W	CC1 比较捕获低 8 位

11.5.10 Compare/Capture High Data Register1(CCH1)

Register	offset	R/W	Description	Reset Value
CCH1	0xc3	R/W	CC1 比较捕获，high byte	0x00

Bits	Symbol	Type	Description
7:0	cch1	R/W	CC1 比较捕获高 8 位

11.5.11 Compare/Capture Low Data Register2(CCL2)

Register	offset	R/W	Description	Reset Value
CCL2	0xc4	R/W	CC2 比较捕获，low byte	0x00

Bits	Symbol	Type	Description
7:0	ccl2	R/W	CC2 比较捕获低 8 位

11.5.12 Compare/Capture High Data Register2(CCH2)

Register	offset	R/W	Description	Reset Value
CCH2	0xc5	R/W	CC2 比较捕获，high byte	0x00

Bits	Symbol	Type	Description
7:0	cch2	R/W	CC2 比较捕获高 8 位

11.5.13 Compare/Capture Low Data Register3(CCL3)

Register	offset	R/W	Description	Reset Value
CCL3	0xc6	R/W	CC3 比较捕获，low byte	0x00

Bits	Symbol	Type	Description
7:0	ccl3	R/W	CC3 比较捕获低 8 位

11.5.14 Compare/Capture High Data Register3(CCH3)

Register	offset	R/W	Description	Reset Value
CCH3	0xc7	R/W	CC3 比较捕获，high byte	0x00

Bits	Symbol	Type	Description
7:0	cch3	R/W	CC3 比较捕获高 8 位

11.5.15 Timer2 Control Register(T2CON)

Register	offset	R/W	Description	Reset Value
T2CON	0xc8	R/W	Timer2 控制寄存器	0x00

Bits	Symbol	Type	Description
7	t2ps	R/W	time2 时钟选择，0：1/12 晶振频率，1：1/24 晶振频率
6	Reserved	R/W	
5	Reserved	R/W	
4	t2r1	R/W	Timer2 重载模式 0x:不使能 10：模式 0 11：模式 1
3	t2r0	R/W	
2	t2cm	R/W	timer2 比较模式选择 0：模式 0 1：模式 1
1	t2i1	R/W	timer2 输入选择 00：timer2 stop 01：输入频率 1/12 或 1/24 10：time2 在 pin 脚 t2 下降沿递增 11：输入频率由外部 pin 脚 t2 门控
0	t2i0	R/W	

11.5.16 Compare/Reload/Capture Low Data Register(CRCL)

Register	offset	R/W	Description	Reset Value
CRCL	0xca	R/W	比较/重载/捕获寄存器 低 8 位	0x00

Bits	Symbol	Type	Description
7:0	crcl	R/W	比较/重载/捕获寄存器 低 8 位

11.5.17 Compare/Reload/Capture High Data Register(CRCH)

Register	offset	R/W	Description	Reset Value
CRCH	0xcb	R/W	比较/重载/捕获寄存器 高 8 位	0x00

Bits	Symbol	Type	Description
7:0	crch	R/W	比较/重载/捕获寄存器 高 8 位

11.5.18 Timer2 Low Data Register(TL2)

Register	offset	R/W	Description	Reset Value
TL2	0xcc	R/W	Timer2，低 8 位	0x00

Bits	Symbol	Type	Description
7:0	tl2	R/W	Timer2 低 8 位

11.5.19 Timer2 High Data Register(TH2)

Register	offset	R/W	Description	Reset Value
TH2	0xcd	R/W	Timer2，高 8 位	0x00

Bits	Symbol	Type	Description
7:0	th2	R/W	Timer2 高 8 位

11.5.20 Serial Port 0 Baud Rate Select register(ADCON)

Register	offset	R/W	Description	Reset Value
ADCON	0xd8	R/W	Serial Port0 Baud Select	0x00

Bits	Symbol	Type	Description
7	bd	R/W	Serial Port 0 baud rate select (in modes 1 and 3) When 1, additional internal baud rate generator is used,

			otherwise Timer 1 overflow is used,
6	timer1_div	R/W	timer1 frequency division select 0: 12 frequency division(default) 1: 3 frequency division, this is support uart 115200 baud rate
5:0	Reserved	Reserved	

11.5.21 Timer2 caputure configure register(TCAPCON)

Register	offset	R/W	Description	Reset Value
TCAPCON	0xab	R/W	Serial Port0 Baud Select	0x00

Bits	Symbol	Type	Description
7:6	cc3_cap_con	R/W	00：上升沿 01：下降沿 10：both 11：reserved
5:4	cc2_cap_con	R/W	00：上升沿 01：下降沿 10：both 11：reserved
3:2	cc1_cap_con	R/W	00：上升沿 01：下降沿 10：both 11：reserved
1:0	cc0_cap_con	R/W	00：上升沿 01：下降沿 10：both 11：reserved

11.5.22 Timer2 caputure status register(TCAPSTA)

Register	offset	R/W	Description	Reset Value
TCAPSTA	0xac	R/W	Serial Port0 Baud Select	0x00

Bits	Symbol	Type	Description
7	cc3_cap_sta	R/W	置 1 表示捕获中断，写 0 清除
6	cc2_cap_sta	R/W	置 1 表示捕获中断，写 0 清除
5	cc1_cap_sta	R/W	置 1 表示捕获中断，写 0 清除
4	cc0_cap_sta	R/W	置 1 表示捕获中断，写 0 清除
3	timer2 overflow	R/W	置 1 表示溢出中断，写 0 清除

2:0	Reserved	Re-served	

第12章 WDT

看门狗定时器由 15 位计数器（不能作为 SFR 访问）、重载寄存器“wdtrel”、预分频器以及控制逻辑组成。

12.1 特征

- 60KHz计数时钟以及可配置的时钟分频，包括不分频、2分频、4分频
- 可配置的溢出周期时间间隔，时间覆盖约为4ms～2s
- 软件启动

12.2 模块框图

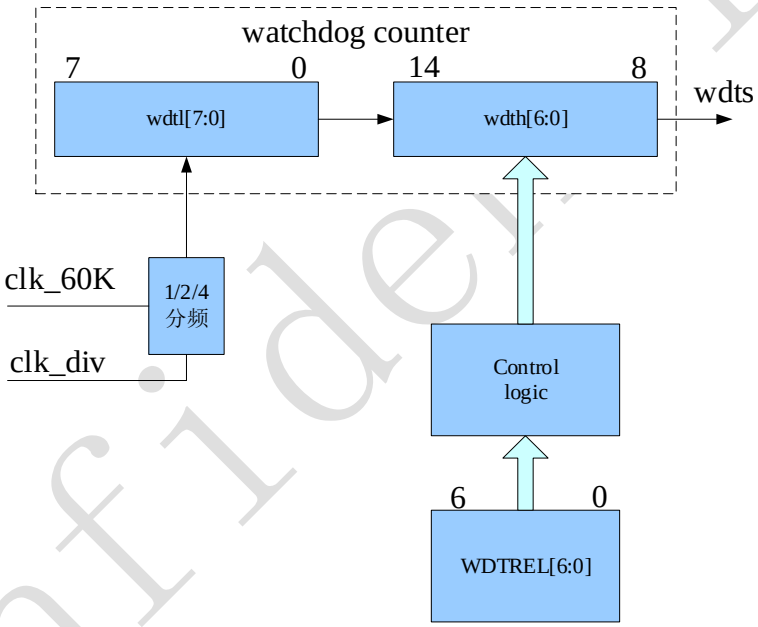


图 12.1 WDT 模块框图

12.3 功能描述

看门狗定时器的计数频率为 60KHz，通过寄存器“WDTCLK”可配置为不分频（WDTCLK 配置 00）、2 分频（WDTCLK 配置 01）、4 分频（WDTCLK 配置 10）。当不分频的时候，WDT 总的计数周期为 256*128=32768 时钟周期。当 2 分频的时候，WDT 总的计数周期为 2*256*128=65536 时钟周期。当 4 分频的时候，WDT 总的计数周期为 4*256*128=131072 时钟周期。

12.3.1 启动过程

启动看门狗定时器的方法是通过软件配置。看门狗定时器通过设置“ien1”寄存器的“swdt”

来启动。仅通过设置“swdt”标志来启动看门狗定时器，该操作不会重新加载看门狗定时器。

一旦启动，看门狗定时器将无法关闭，除非芯片进行复位操作或者通过寄存器 wdgclken 将看门狗的时钟关闭。当看门狗计数器达到 7FFCh 时，“wdts”输出为高电平，产生内部复位。看门狗定时器复位之后，“ip0”寄存器的“wdts”标志位被置位，清除该标志位的方法是复位或是软件的写“0”清除。

“wdts”信号不会复位看门狗模块，此时看门狗仍在运行。当看门狗计数器从 7FFFh 溢出到 0000h 时，“wdts”输出为低电平（不具有复位功能），而“ip0”寄存器的“wdts”标志位依然保持为 1，其让软件辨认该复位是由外部输入引起或是由看门狗超时引起的。

12.3.2 刷新 WDT

看门狗定时器必须定期刷新以防止 WDT 复位信号“wdts”被触发。这个需要软件连续发出两条指令。第一条指令将“ien0”寄存器的“wdt”位置 1，第二条指令将“ien1”寄存器的“swdt”位置 1。设置“wdt”和“swdt”之间允许的最大延迟为 1 个指令周期（这意味着设置这两个标志的指令不能被任何其他指令分开）。设置完成之后，从而避免出现 WDT 复位，此时“wdtrel”寄存器的内容（看门狗定时器的高 7 位）将重新加载到定时器中。

其中寄存器“wdtrel”中的值越大，刷新看门狗定时器所需的时间越短。

12.4 软件接口

12.5 寄存器列表

Register	SFR Address	R/W	Description	Reset Value
WDTREL	0x86	R/W	WDT 重载寄存器	0x00
PCON	0x87	R/W	功率控制寄存器	0x00
IEN0	0xa8	R/W	中断使能 0 寄存器	0x00
IEN1	0xb8	R/W	中断使能 1 寄存器	0x00
IP0	0xa9	R/W	中断优先 0 寄存器	0x00

12.6 寄存器描述

12.6.1 Watchdog Timer Reload Register(WDTREL)

Register	SFR Address	R/W	Description	Reset Value
WDTREL	0x86	R/W	WDT 重新加载寄存器	0x00

Bits	Symbol	Type	Description
7	reserved	reserved	
6:0	reload_value	R/W	wdt 计数器高 7 位

12.6.2 Power Control Register(PCON)

Register	SFR Address	R/W	Description	Reset Value
PCON	0x87	R/W	功率控制寄存器	0x00

Bits	Symbol	Type	Description
7	other function	R/W	其他功能寄存器，具体功能请参考其他模块
6	reserved		
5:0	other function	R/W	其他功能寄存器，具体功能请参考其他模块

12.6.3 WDT CLK Register(WDTCLK)

Register	SFR Address	R/W	Description	Reset Value
WDTCLK	0x8f	R/W	功率控制寄存器	0x00

Bits	Symbol	Type	Description
7:2	reserved	R	not used
1:0	clk_div	R/W	wdt 60K 时钟分频 00：不分频 01：2 分频 10：4 分频

12.6.4 Interrupt Enable 0 Register (IEN0)

Register	SFR Address	R/W	Description	Reset Value
IEN0	0xa8	R/W	中断使能 0 寄存器	0x00

Bits	Symbol	Type	Description
7	eal	R/W	中断总使能 0：不使能 1：使能
6	wdt	R/W	wdt 刷新控制位 其必须在 swdt(ien1.6)之前进行置位，才可以刷新 wdt 计数器
5	et2	R/W	timer2 中断使能 0：不使能 1：使能
4	es0	R/W	UART 中断使能

			0：不使能 1：使能
3	et1	R/W	timer1 溢出中断使能 0：不使能 1：使能
2	ex1	R/W	外部中断 1 使能 0：不使能 1：使能
1	et0	R/W	timer0 溢出中断使能 0：不使能 1：使能
0	ex0	R/W	外部中断 0 使能 0：不使能 1：使能

12.6.5 Interrupt Enable 1 Register(IEN1)

Register	SFR Address	R/W	Description	Reset Value
IEN1	0xb8	R/W	中断使能 1 寄存器	0x00

Bits	Symbol	Type	Description
7	exen2	R/W	timer2 外部重新加载中断使能 0：不使能 1：使能
6	swdt	R/W	wdt 使能/刷新控制位 刷新时，需要在 wdt（ien0.6）之后置位
5	ex6	R/W	外部中断 6 使能 0：不使能 1：使能
4	ex5	R/W	外部中断 5 使能 0：不使能 1：使能
3	ex4	R/W	外部中断 4 使能 0：不使能 1：使能
2	ex3	R/W	外部中断 3 使能 0：不使能 1：使能
1	ex2	R/W	外部中断 2 使能 0：不使能

			1：使能
0	ex7	R/W	外部中断 7 使能 0：不使能 1：使能

12.6.6 Interrupt Priority 0 Register(IP0)

Register	SFR Address	R/W	Description	Reset Value
IP0	0xa9	R/W	中断优先 0 寄存器	0x00

Bits	Symbol	Type	Description
7	reserved	R	not used
6	wdts	R/W	wdt 状态标志位，当 wdt 复位发生时，其由硬件置位
5:0	other function	R/W	other function

第13章 ADC

仅支持PAN2628系列。

PAN2628包含一个具有13个输入通道的12位逐次逼近模数转换器（SAR ADC）。该ADC可以被软件或是外部pin脚P31/P33/P36/P37触发。

13.1 特征

- 一个输入的电压参考档位，0~VDD
- 12位分辨率与10位精度
- 8个单端模拟输入通道、1个band-gap输入通道、1个VRSSI输入通道、1个VDD/2输入通道、1个GND输入通道、1个Temp输入通道
- ADC可以通过软件触发或是外部pin脚触发
- ADC每次转换完成会存入数据寄存器中，并通过中断或是状态寄存器指示出来
- ADC转换结果可以用来与特定的数值进行比较，当转换结果与特定的数值相符时，可以通过相应中断响应
- 温度传感器使用，需要内部温度传感器电压转换code与VDD电压(或者与内部vbg转换code)折算处理

13.2 输入阻抗

$$Rin < \frac{Ts}{Csample \times \ln(2^{N+1})} - R_{adc}$$

有效位	**系统频率（MHz）**	**采样时间**	**ADC_CLK 分频**	**周期数 (ADC_CLK)**	**Radc (Ω)**	**采样电容（pF）**	**PCLK（us）**	**ADC_CLK(us)**	**Ts(us)**	**Rin 最大 (kΩ)**
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	1	1	2	300	12	0.0625	0.125	0.25	2.012
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	4	4	5	300	12	0.0625	0.3125	1.5625	14.15
12	16	15	7	16	300	12	0.0625	0.5	8	73.684
12	16	15	7	16	300	12	0.0625	0.5	8	73.684

12	16	15	7	16	300	12	0.0625	0.5	8	73.684
12	16	15	7	16	300	12	0.0625	0.5	8	73.684
12	16	15	7	16	300	12	0.0625	0.5	8	73.684
12	16	15	7	16	300	12	0.0625	0.5	8	73.684
12	16	31	7	32	300	12	0.0625	0.5	16	147.668
12	16	31	7	32	300	12	0.0625	0.5	16	147.668
12	16	31	7	32	300	12	0.0625	0.5	16	147.668
12	16	31	7	32	300	12	0.0625	0.5	16	147.668
12	16	31	7	32	300	12	0.0625	0.5	16	147.668
12	16	31	7	32	300	12	0.0625	0.5	16	147.668

13.3 模块框图

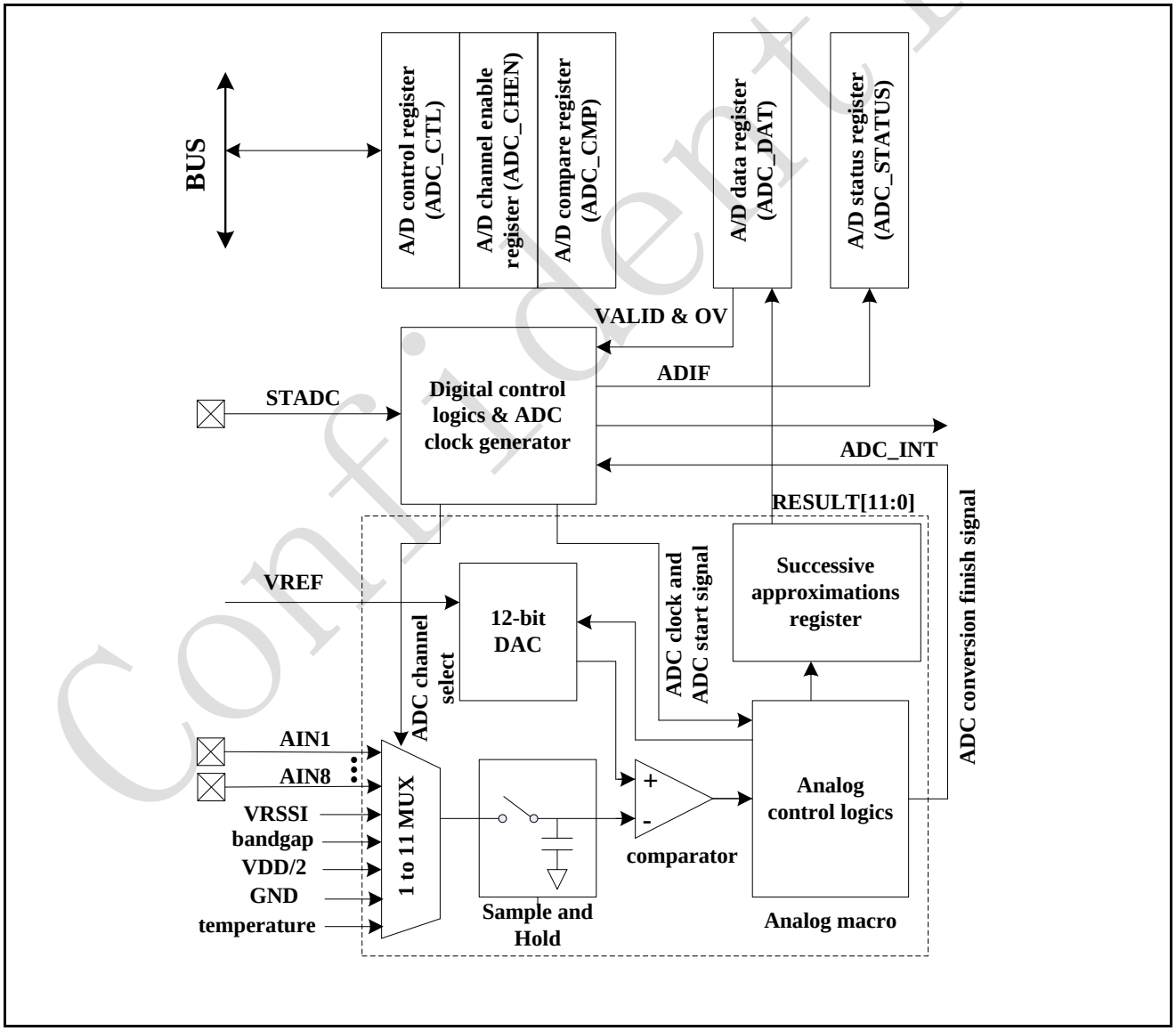


图 13.1 ADC 模块框图

13.4 功能描述

ADC模拟电压pin脚分布在P30/P31/P32/P33/P34/P35/P36/P37。

ADC外部触发pin脚分布在P31/P33/P36/P37。

13.4.1 ADC 模式

13.4.1.1 ADC 转换模式

ADC 在特定的单端通道下，执行模数转换操作一次。具体操作如下：

(1). 当ADC_CTL的SWTRG位被置1或是有外部的pin脚的触发时，ADC开始转换工作。

(2). 当ADC完成转换操作，其结果被保存在ADC的数据寄存器中。

(3). 状态寄存器ADC_STATUS的ADIF位将会被置1。如果ADC_CTL的ADCIE位被置1，ADC中断将会被触发。

(4). 在ADC转换过程中，SWTRG位需要一直保持为1，在ADC转换完成时，其被自动清0，并且ADC会进入空闲状态。

下图所示，为 ADC 一次转换的时序图，其中 N 为采样时间，单位为 ADC_CLK，转换时间为 14 个 ADC_CLK。

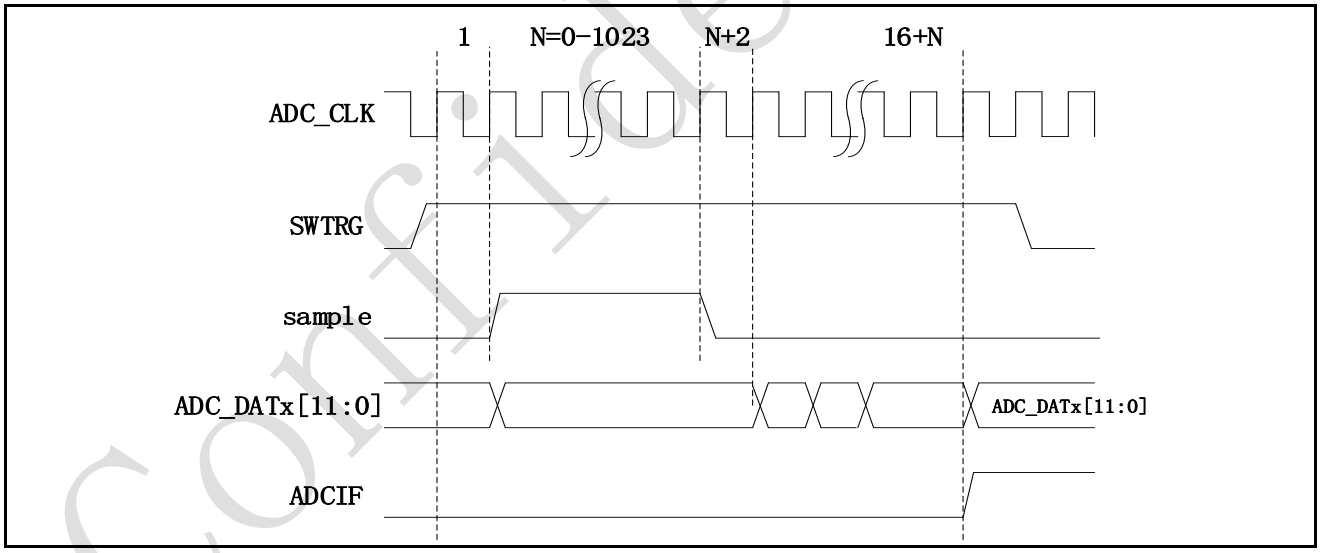


图 13.2 ADC 转换时序图

13.4.1.2 ADC 校准模式

ADC 添加校准模式，其主要是通过模拟反馈给数字信号，确定寄生电容校准值，以给正常转换时，给模拟电路使用。

13.4.2 外部触发

当HWTRGSEL设置为00，并且HWTRGEN被置1，使能外部pin脚触发ADC，其上升沿或是下降沿通过HWTRGCOND来设置选择。被设置成外部触发后，ADC内部有个去抖逻辑，当边沿触发被选择之后，其相应的高低电平必须持续保持4个系统时钟，如果小于该时间，该将被忽略。

13.4.3 ADC 比较功能

PAN2628 ADC转换器包含两个比较寄存器，ADC_CMP0与ADC_CMP1，去检测特定的两个通道。软件可以通过配置CMPCH来选择要监视的通道。CMPCOND是用来设置比较条件，当其设置为0，当转换结果小于CMPDAT中指定的值时，内部计数器会被加一；如果CMPCOND设置成1，当转换结果大于或等于CMPDAT中指定的值时，内部计数器会被加一。

当CMPCH指定的通道转换完成时，比较动作会自动触发一次。当比较结果满足设置时，比较计数器将增加1，否则，比较计数器将清除为0。当计数器达到(CMPMCNT+1) 设置时，ADCMPIF位将被设置为1，如果ADCMPIE被置位，则生成ADC_INT中断。

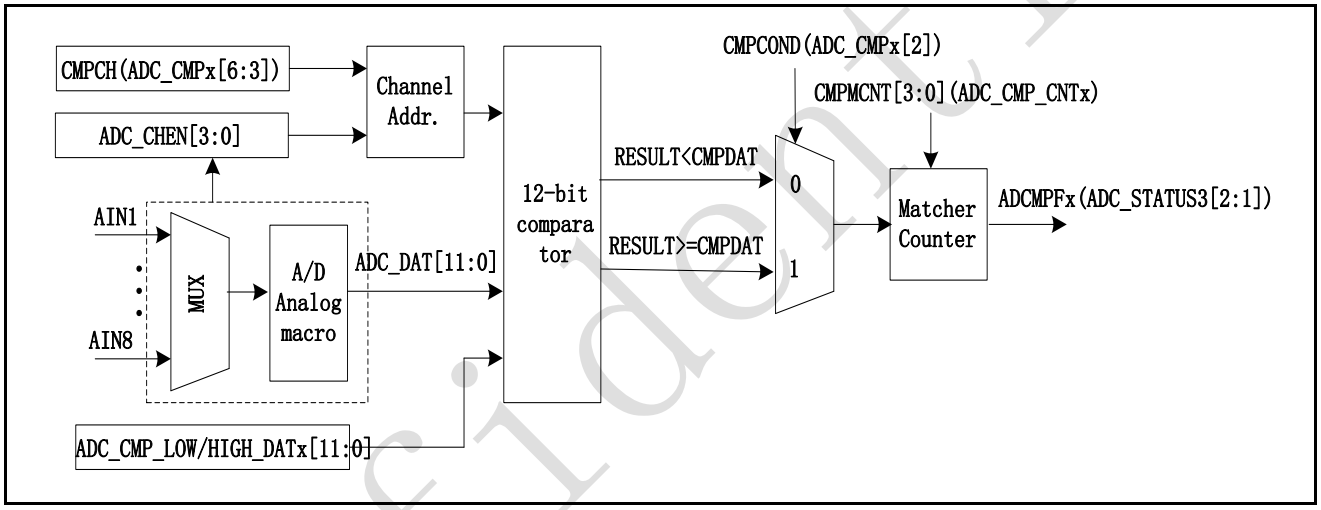


图 13.3 ADC 比较功能

13.4.4 ADC 中断源

ADC控制器有三个中断源。当转换模式或是校准模式完成时，ADC的结束标志ADIF置位。ADCMPIF0与ADCMPIF1是比较的中断标志，当满足结果时，相应的标志将置1。当中断使能，相应的将触发中断。

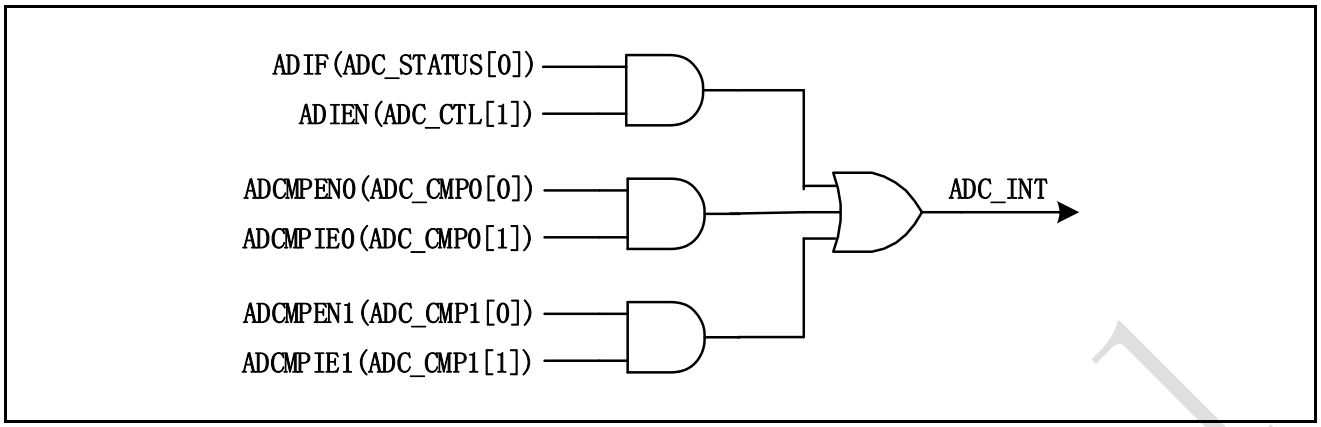


图 13.4 ADC 中断源

13.5 ADC 寄存器列表

R: read only, **W**: write only, **R/W**: both read and write

ADC_BA= 0x2000

Register	Offset	R/W	Description	Reset Value
ADC_DAT0	ADC_BA+0x00	R	ADC 数据寄存器 0	0x00
ADC_DAT1	ADC_BA+0x01	R	ADC 数据寄存器 1	0x00
ADC_DAT_STATUS	ADC_BA+0x02	R	ADC 数据状态寄存器	0x00
ADC_CTL0	ADC_BA+0x20	R/W	ADC 控制寄存器 0	0x00
ADC_CTL1	ADC_BA+0x21	R/W	ADC 控制寄存器 1	0x00
ADC_CH	ADC_BA+0x24	R/W	ADC 通道选择寄存器	0x00
ADC_CMP0/1	ADC_BA+0x28/2c	R/W	ADC 比较寄存器 0/1	0x00
ADC_CMP_CNT0/1	ADC_BA+0x29/2d	R/W	ADC 比较 CNT 寄存器 0/1	0x00
ADC_CMP_LOW_DAT0	ADC_BA+0x2a	R/W	ADC 低阈值比较数据寄存器 0	0x00
ADC_CMP_LOW_DAT1	ADC_BA+0x2b	R/W	ADC 低阈值比较数据寄存器 1	0x00
ADC_CMP_HIGH_DAT0	ADC_BA+0x2e	R/W	ADC 高阈值比较数据寄存器 0	0x00
ADC_CMP_HIGH_DAT1	ADC_BA+0x2f	R/W	ADC 高阈值比较数据寄存器 1	0x00
ADC_STATUS0	ADC_BA+0x30	R	ADC 状态寄存器 0	0x00
ADC_STATUS1	ADC_BA+0x31	R	ADC 状态寄存器 1	0x00
ADC_STATUS2	ADC_BA+0x32	R	ADC 状态寄存器 2	0x00
ADC_STATUS3	ADC_BA+0x33	R	ADC 状态寄存器 3	0x00
ADC_EXTSMPT0	ADC_BA+0x48	R/W	ADC 采样时间寄存器 0	0x00
ADC_EXTSMPT1	ADC_BA+0x49	R/W	ADC 采样时间寄存器 1	0x02
ADC_CAL_CTL	ADC_BA+0x58	R/W	ADC 校准控制寄存器	0x02
ADC_CLK_CTL	ADC_BA+0x59	R/W	ADC 时钟控制寄存器	0x0a
ADC_ANA_CTL0	ADC_BA+0x5a	R/W	ADC 模拟寄存器 0	0x00
ADC_ANA_CTL1	ADC_BA+0x5c	R/W	ADC 模拟寄存器 1	0x00

ADC_ANA_CTL2	ADC_BA+0x5d	R/W	ADC 模拟寄存器 2	0x10
ADC_CAL_DAT	ADC_BA+0x5e	R	ADC 校准数据寄存器	0x00
ADC_CAL_SOFT_DAT	ADC_BA+0x5f	R/W	ADC 软件校准数据寄存器	0x00

13.6 ADC 寄存器描述

13.6.1 ADC Data Register (ADC_DAT0)

Register	Offset	R/W	Description	Reset Value
ADC_DAT0	ADC_BA+0x00	R	ADC 数据寄存器 0	0x00

Bits	Description	
[7:0]	RESULT	A/D 转换结果的低 8 位，共 12 位

13.6.2 ADC Data Register (ADC_DAT1)

Register	Offset	R/W	Description	Reset Value
ADC_DAT1	ADC_BA+0x01	R	ADC 数据寄存器 1	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[3:0]	RESULT	A/D 转换结果的高 4 位，共 12 位

13.6.3 ADC Data Status Register (ADC_DAT_STATUS)

Register	Offset	R/W	Description	Reset Value
ADC_DAT_STA	ADC_BA+0x02	R	ADC 数据寄存器 2	0x00

Bits	Description	
[7:2]	Reserved	Reserved.
[1]	VALID	有效标志位，该位在转换完成时被置 1，并且在数据寄存器 ADC_DAT0 被读出时自动清 0。 0 :数据寄存器 ADC_DAT0 与 ADC_DAT1 无效 1 :数据寄存器 ADC_DAT0 与 ADC_DAT1 有效
[0]	OV	溢出标志，如果转换结果没有在新的转换结束之前被读出，该 OV 将被置位，在数据寄存器被读出时，其将被清 0。 0 :数据寄存器 ADC_DAT0 与 ADC_DAT1 为当前转换的结果。 1 :数据寄存器 ADC_DAT0 与 ADC_DAT1 被覆盖。

13.6.4 ADC Control Register (ADC_CTL0)

Register	Offset	R/W	Description	Reset Value
ADC_CTL0	ADC_BA+0x20	R/W	ADC 控制寄存器 0	0x00

Bits	Description	
[7]	Reserved	Reserved.

[6]	HWTRGCOND	硬件外部触发条件选择 该位决定外部 pin 脚 STADC 的触发条件是上升沿还是下降沿。相应的状态需要保持至少 4 个系统时钟时间。 0 :下降沿触发 1 :上升沿触发.
[5:4]	HWTRGSEL	硬件触发选择 00 ： AD 转换由外部 pin 脚触发 其他：保留. 注意：软件需要在改变 HWTRGSEL 之前，需要禁止使能 HWTRGEN 和 SWTRG
[3:2]	Reserved	Reserved.
[1]	ADCIEN	A/D 中断使能控制位 AD 转换完成，并且该位被置 1 时，会产生中断 0 : A/D 中断禁止使能 1 :A/D 中断使能
[0]	EN_SAR_ADC	SAR_ADC 使能控制 0：SAR_ADC 不使能 1：SAR_ADC 使能 注意：在 A/D 转换之前，该位需要被置 1。为了节省功耗，不使用时需要清 0，以降低功耗。

13.6.5 ADC Control Register (ADC_CTL1)

Register	Offset	R/W	Description	Reset Value
ADC_CTL1	ADC_BA+0x21	R/W	ADC 控制寄存器 1	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[3]	SWTRG	A/D 的软件触发使能 SWTRG 位被置 1，有两个来源，一个是软件，另一个是外部 pin 脚。当转完成后，该位被硬件自动清 0。 0 :A/D 停止工作，进入空闲状态 1 :A/D 转换开始
[2:1]	Reserved	Reserved.
[0]	HWTRGEN	硬件外部触发使能 通过外部 pin 脚，使能或是禁止使能 A/D 转换。如果使能，SWTRG 位被相应的外部触发源置位。 0 :硬件外部不使能 1 :硬件外部使能

13.6.6 ADC Channel Enable Register (ADC_CH)

Register	Offset	R/W	Description	Reset Value

ADC_CH	ADC_BA+0x24	R/W	ADC 通道选择寄存器	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[3:0]	ADC_CH	模拟通道选择 0000: 通道 1 选择 ... 0111: 通道 8 选择 1000: VRSSI 1001: 使能测试 VBG 1010: 使能测试 VDD/2 1011: 使能测试 GND 1100: 温度检测 注：在 VRSSI 通道选择时，ADC 的触发工作，由硬件触发控制，ADC 转换完成时，将产生 ADC 中断，软件根据中断读取 ADC DATA

13.6.7 ADC Compare Register 0/1 (ADC_CMP0/1)

Register	Offset	R/W	Description	Reset Value
ADC_CMP0	ADC_BA+0x28	R/W	ADC 比较寄存器 0	0x00
ADC_CMP1	ADC_BA+0x2C	R/W	ADC 比较寄存器 1	0x00

Bits	Description	
[7]	Reserved	Reserved.
[6:3]	CMPCH	通道的选择 选择哪个通道被用来做比较 注意：该通道有效数值被设置为 0~7
[2]	CMPCOND	比较条件设置 0 :当 ADC 转换结果小于 CMPDAT 时，内部计数器将增 1 1 :当 ADC 转换结果大于或等于 CMPDAT 时，内部计数器将增 1 注意：当内部计数器达到 CMPMCNT+1 时， ADCMPFx 将被置位
[1]	ADCMPIE	A/D 比较中断使能位 如果比较功能使能，当比较的条件达到 CMPCOND 与 CMPMCNT 设置的条件时，并且 ADCMPIE 被置位，将产生比较中断。 0 :比较功能中断不使能 1 :比较功能中断使能
[0]	ADCMPEN	A/D 比较功能使能位 使能的 CMPDAT 与 A/D 转换结果进行比较 0 :比较功能不使能 1 :比较功能使能.

13.6.8 ADC Compare CNT Register 0/1 (ADC_CMP_CNT0/1)

Register	Offset	R/W	Description	Reset Value
ADC_CMP_CNT0	ADC_BA+0x29	R/W	ADC 比较 CNT 计数器 0	0x00
ADC_CMP_CNT1	ADC_BA+0x2D	R/W	ADC 比较 CNT 计数器 1	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[3:0]	CMPMCNT	比较计数器 当指定的 A/D 转换结果达到由 CMPCOND 设置的条件时，内部计数器将被加 1。当内部计数器达到 CMPMCNT+1，ADCMPF 将被置位。

13.6.9 ADC Compare Low DAT Register 0 (ADC_CMP_LOW_DAT0)

Register	Offset	R/W	Description	Reset Value
ADC_CMP_LOW_DAT0	ADC_BA+0x2A	R/W	ADC 低阈值比较数据寄存器 0	0x00

Bits	Description	
[7:0]	CMP_LOW_DAT	低阈值比较数值，低 8 位，共 12 位

13.6.10 ADC Compare Low DAT Register 1 (ADC_CMP_LOW_DAT1)

Register	Offset	R/W	Description	Reset Value
ADC_CMP_LOW_DAT1	ADC_BA+0x2B	R/W	ADC 低阈值比较数据寄存器 1	0x00

Bits	Description	
[7:0]	CMP_LOW_DAT	低阈值比较数值，高 4 位，共 12 位

13.6.11 ADC Compare High DAT Register 0 (ADC_CMP_HIGH_DAT0)

Register	Offset	R/W	Description	Reset Value
ADC_CMP_HIGH_DAT0	ADC_BA+0x2E	R/W	ADC 高阈值比较数据寄存器 0	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[7:0]	CMP_HIGH_DAT	高阈值比较数值，低 8 位，共 12 位

13.6.12 ADC Compare High DAT Register 1 (ADC_CMP_HIGH_DAT1)

Register	Offset	R/W	Description	Reset Value
ADC_CMP_HIGH_DAT1	ADC_BA+0x2F	R/W	ADC 高阈值比较数据寄存器 1	0x00

Bits	Description	
[7:4]	Reserved	Reserved.
[3:0]	CMP_HIGH_DAT	高阈值比较数值，高 4 位，共 12 位

13.6.13 ADC Status0 Register (ADC_STATUS0)

Register	Offset	R/W	Description	Reset Value
ADC_STATUS0	ADC_BA+0x30	R/W	ADC 状态寄存器 0	0x00

Bits	Description	
[7:4]	CHANNEL	当前通道选择(只读) 当 BUSY=1 时，该反应了当前的转换通道；当 BUSY=0 时，其反应了下次转换通道 。
[3]	BUSY	忙/空闲(只读) 该位是 ADC_CTL 中 SWTRG 的镜像 0 :A/D 转换进入空闲状态 1 :A/D 转换进行忙状态
[2]	ADCMPIF1	A/D 比较中断标志位 当 A/D 转换结果达到设置的比较条件时，该将被置位。 0 :A/D 转换结果没有达到 ADC_CMP1 的转换条件 1 :A/D 转换结果达到 ADC_CMP1 的转换条件 注意：该位可以通过软件写 1 进行清 0
[1]	ADCMPIF0	A/D 比较中断标志位 当 A/D 转换结果达到设置的比较条件时，该将被置位。 0 :A/D 转换结果没有达到 ADC_CMP0 的转换条件 1 :A/D 转换结果达到 ADC_CMP0 的转换条件 注意：该位可以通过软件写 1 进行清 0
[0]	ADIF	A/D 转换结束中断标志位 当置位时，其反应 A/D 转换结果的完成。 注意：该位可以通过软件写 1 进行清 0

13.6.14 ADC Status1 Register (ADC_STATUS1)

Register	Offset	R/W	Description	Reset Value
ADC_STATUS1	ADC_BA+0x31	R/W	ADC 状态寄存器 1	0x00

Bits	Description	
[7:1]	Reserved	Reserved.
[0]	VALID	数据有效标志位（只读） 其是 ADC_DAT 中 VALID 的镜像

13.6.15 ADC Status2 Register (ADC_STATUS2)

Register	Offset	R/W	Description	Reset Value
ADC_STA-TUS2	ADC_BA+0x32	R/W	ADC 状态寄存器 2	0x00

Bits	Description	
[7:1]	Reserved	Reserved.
[0]	OV	溢出标志位（只读） 其是 ADC_DAT 中 OV 的镜像

13.6.16 ADC Status3 Register (ADC_STATUS3)

Register	Offset	R/W	Description	Reset Value
ADC_STA-TUS3	ADC_BA+0x33	R/W	ADC 状态寄存器 3	0x00

Bits	Description	
[7:3]	Reserved	Reserved.
[2]	ADCMPF1	A/D 比较标志位 1 当选择的通道 A/D 转换结果达到 ADC_CMP1 设置的条件时，该位将被置 1。 0: A/D 转换结果没有达到 ADC_CMP1 设置的条件 1: A/D 转换结果达到 ADC_CMP1 设置的条件 注意：该位可以通过软件写 1 进行清 0
[1]	ADCMPF0	A/D 比较标志位 0 当选择的通道 A/D 转换结果达到 ADC_CMP0 设置的条件时，该位将被置 1。 0:A/D 转换结果没有达到 ADC_CMP0 设置的条件 1:A/D 转换结果达到 ADC_CMP0 设置的条件 注意：该位可以通过软件写 1 进行清 0
[0]	ADCF	A/D 转换结束标志位 当置位时，其反应 A/D 转换结果的完成。 注意：该位可以通过软件写 1 进行清 0

13.6.17 ADC Sampling Register0 (ADC_EXTSMPT0)

Register	Offset	R/W	Description	Reset Value
ADC_EXTSMPT0	ADC_BA+0x48	R/W	ADC 低字节采样时间寄存器	0x00

Bits	Description	
[7:0]	ADC_EXTSMPT0	A/D 采样时间 A/D 转换采样时间的低 8 位，单位是 ADC_CLK；ADC_EXTSMPT0 与 ADC_EXTSMPT1

		均为0将不使能ADC;采样时间为{ADC_EXTSMP1,ADC_EXTSMP0}+1个ADC_CLK

13.6.18 ADC Sampling Register1 (ADC_EXTSMPT1)

Register	Offset	R/W	Description	Reset Value
ADC_EXTSMPT1	ADC_BA+0x49	R/W	ADC 高字节采样时间寄存器	0x02

Bits	Description	
[7:2]	Reserved	Reserved.
[1:0]	ADC_EXTSMPT1	A/D 采样时间 A/D 转换采样是时间的高 2 位，单位是 ADC_CLK；ADC_EXTSMPT0 与 ADC_EXTSMPT1 均为 0 将不使能 ADC；采样时间为{ADC_EXTSMP1,ADC_EXTSMP0}+1 个 ADC_CLK

13.6.19 ADC CAL Control Register (ADC_CAL_CTL)

Register	Offset	R/W	Description	Reset Value
ADC_CAL_CTL	ADC_BA+0x58	R/W	ADC 校准控制寄存器	0x02

Bits	Description	
[7:4]	Reserved	Reserved.
[3]	ADC_CAL_EN	A/D 校准模式使能 0: 不使能 A/D 校准 1: 使能 A/D 校准
[2]	Reserved	Reserved
[1]	ADC_SEL_SH	额外的采样时间设置 0: 添加 HLOD2 到采样时间 1: 不添加额外的采样时间
[0]	TEST_MODE	A/D 测试使能 0：不使能 A/D 测试模式 1：使能 A/D 测试模式

13.6.20 ADC CLK Control Register (ADC_CLK_CTL)

Register	Offset	R/W	Description	Reset Value
ADC_CLK_CTL	ADC_BA+0x59	R/W	ADC 时钟寄存器	0x0A

Bits	Description	
[7:6]	Reserved	Reserved.
[5:3]	CLK_DIV_DUTY	ADC 时钟分频占空比，默认为 1，CLK_DIV 值需要大于 CLK_DIV_DUTY； 其 ADC_CLK 时钟的高低电平时间占空比为(CLK_DIV_DUTY+1) ：(CLK_DIV - CLK_DIV_DUTY)

[2:0]	CLK_DIV	ADC 时钟分频系数，不能被设置为 0，默认为 2，CLK_DIV 值需要大于 CLK_DIV_DUTY ADC_CLK = 系统时钟/(CLK_DIV+1)，其中系统时钟为 16M

13.6.21 ADC ANA Control Register 0(ADC_ANA_CTL0)

Register	Offset	R/W	Description	Reset Value
ADC_ANA_CTL0	ADC_BA+0x5A	R/W	A/D 模拟控制寄存器 0	0x00

Bits	Description	
[7:1]	Reserved	Reserved.
[0]	ADC_EN_BUFTST	参考电平测试使能： 0：使能关断 1：使能输出

13.6.22 ADC ANA Control Register 1(ADC_ANA_CTL1)

Register	Offset	R/W	Description	Reset Value
ADC_ANA_CTL1	ADC_BA+0x5C	R/W	A/D 模拟控制寄存器 1	0x00

Bits	Description	
[7:4]	Reserved	Reserved
[3]	ADC_VBG_RESARR_FT	8 个输入通道过 BUF
[2]	ADC_VCM_BC	VCM 参考电压驱动电流控制位 0：2uA 1：3uA
[1]	ADC_CMP_BC	比较器的电流控制位 0：1uA 1：1.5uA
[0]	TST_ADCIN	ADC 内部测试 使能信号 0：无效 1：有效

13.6.23 ADC ANA Control Register 2 (ADC_ANA_CTL2)

Register	Offset	R/W	Description	Reset Value
ADC_ANA_CTL2	ADC_BA+0x5D	R/W	A/D 模拟控制寄存器 2	0x10

Bits	Description	
[7:5]	Reserved	Reserved .
[4:0]	ADC_VBG_RESARR_CT	偏置电压电阻阵列控制位，1.105V~1.31V，6.6mV/step

13.6.24 ADC Cal Data Register (ADC_CAL_DAT)

Register	Offset	R/W	Description	Reset Value
ADC_CAL_DAT	ADC_BA+0x5E	R	A/D 校准数据寄存器	0x00

Bits	Description	
[7:5]	Reserved	Reserved
[4:0]	ADC_CAL_CTLH	ADC 硬件校准值(只读)

13.6.25 ADC Cal Soft Data Register (ADC_CAL_SOFT_DAT)

Register	Offset	R/W	Description	Reset Value
ADC_CAL_SOFT_DAT	ADC_BA+0x5F	R/W	A/D 软件校准数据寄存器	0x00

Bits	Description	
[7:5]	Reserved	Reserved
[4:0]	ADC_SOFT_CAL_CTLH	ADC 软件校准值；SAR_ADC 在使能之后，才可更新该校准值寄存器。

第14章 PWM

14.1 特征

- PWM单元支持6通道独立输出PWM0_CH0~PWM0_CH5
- PWM单元支持3组互补输出（PWM0_CH0~PWM0_CH1）/（PWM0_CH2~PWM0_CH3）/（PWM0_CH4~PWM0_CH5），并支持上升沿/下降沿死区区间设置
- 周期计数器period、比较寄存器cmp都是12bit，保证分辨率六个通道的频率保持一致性
- 分频系数 1、1/2、1/4、1/8，即4个时钟源，且6通道共用
- 每个通道拥有单独反向权限
- 组模式下每个通道共用2个比较寄存器，单独模式下两个比较寄存器分别作用0/1通道，且此时死区时间设定无效
- 保证IO模式和PWM模式切换同步，即保证一个PWM周期波形的稳定性，例如从PWM模式退出到IO其他模式，保证一个PWM周期波形完整输出，从IO其他模式进入PWM模式也是需要稳定后才能切到PWM输出（不止涉及PWM）
- 中断类型仅支持周期完成时触发中断，零中断触发

14.2 功能框图

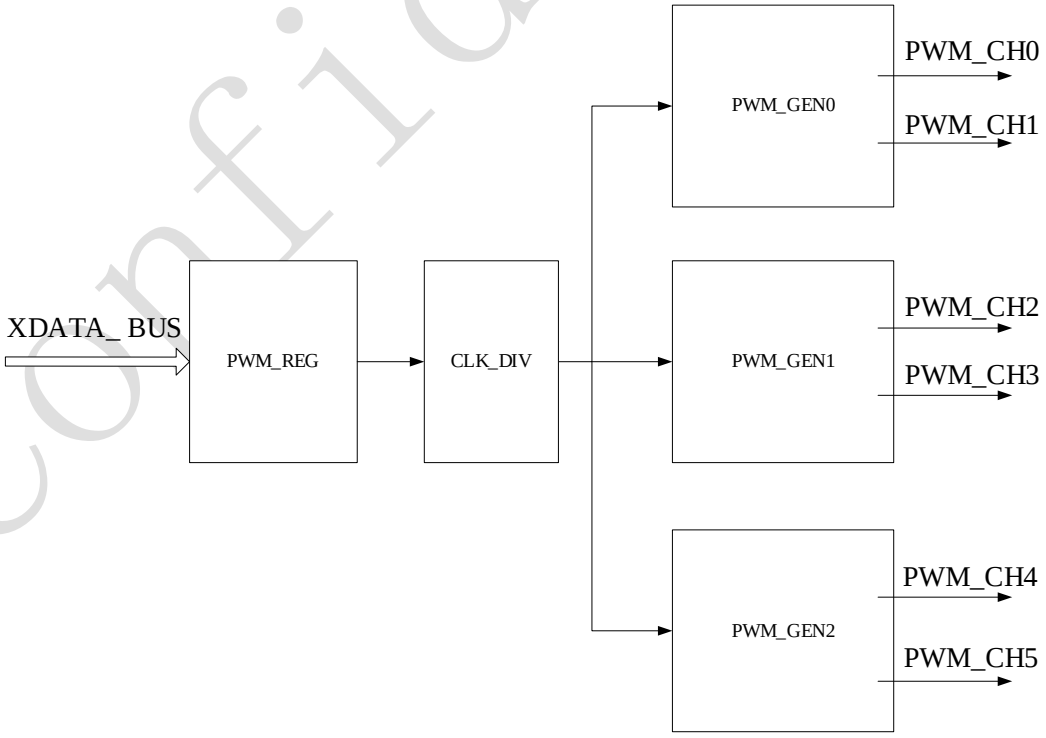


图 14-1 PWM 模块框图

14.3 功能描述

14.3.1 波形频率

PWM 模块存在两种模式，独立模式与组模式，不同模式下所形成的 PWM 波形频率存在不同，如下公式所示。

①独立模式

$$F_{pwm} = \frac{f_{osc}}{PWM\_DIV * (PWM\_PERIOD[11:0]+1)}$$

②组模式

$$F_{pwm} = \frac{f_{osc}}{PWM\_DIV * 2 * (PWM\_PERIOD[11:0]+1)}$$

14.3.2 独立模式

在独立模式下，PWM 每个通道均可作为独立的通道进行输出波形。其中高低电平比=(CMPn+1)/(PERIODn-CMPn)。软件在配置时，PERIODn 的数值需要大于等于 CMPn 的数值。

①当 PERIOD = 0 时，PWM 输出一直为低。

②当 PERIOD!=0, PWM 输出如下所示，CMPn ≥ count，PWM 输出为高，其宽度为(CMP+1)；CMPn < count， PWM 输出为低，其宽度(PERIODn-CMPn)。

③当 CMPn=0，PWM 输出一直为低。

④当 CMPn=PERIODn ，PWM 输出一直为高。

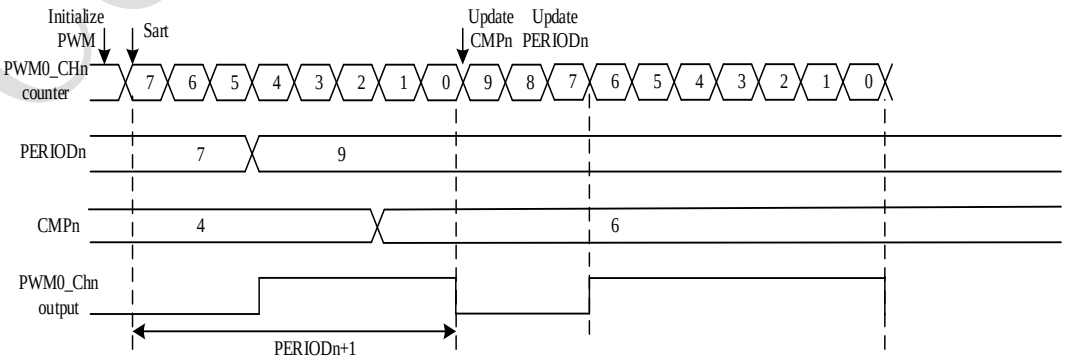


图 14-2 独立模式下的 PWM 输出示意图

14.3.3 组模式

在组模式下，PWM 每两个通道为一组，分别为 PWM0_CH0 与 PWM0_CH1，PWM0_CH2 与 PWM0_CH3，PWM0_CH4 与 PWM0_CH5，可以作为同步输出，或是互补输出。其中高低电平比=（2*PERIODn-CMPn（上升沿）- CMPn（下降沿））/（CMPn（上升沿）+ CMPn（下降沿）+2）。软件在配置时，PERIODn 的数值需要大于等于 CMPn 的数值。

①PERIOD =0 时，PWM 输出一直为低。

②PERIOD !=0 时，CMP 上升沿＞ count || CMP 下降沿≥ count，PWM 输出为低，其输出宽度为 CMPn（上升沿）+ CMPn（下降沿）+2 ；count ≥ CMP 上升沿 || count ＞CMP 下降沿，PWM 输出为高，其输出宽度为 2*PERIODn - CMPn（上升沿）- CMPn（下降沿）。

③CMPn（上升沿）=CMPn（下降沿）= 0，PWM 输出一直为高。

④CMPn（上升沿）=CMPn（下降沿）= PERIODn，PWM 输出一直为低。

注：互补模式下不应该使用 PWM_PINV 寄存器去反转输出通道极性。

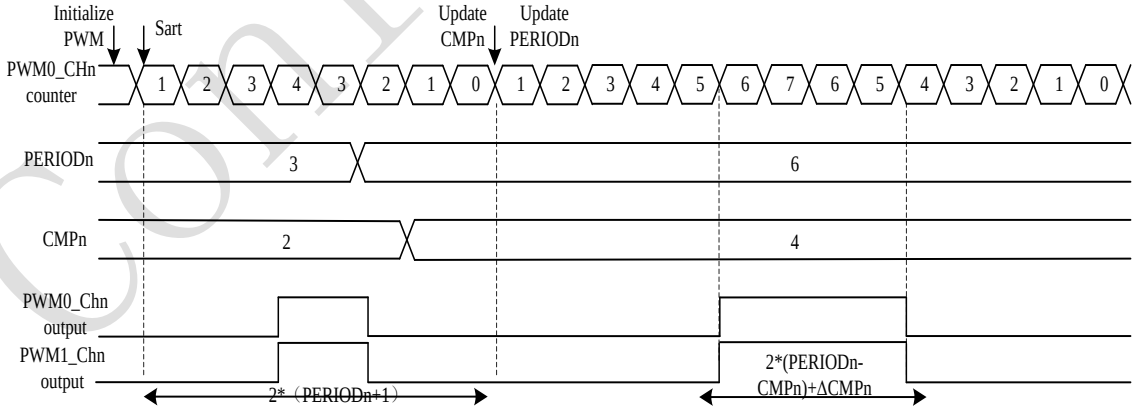


图 14-3 组模式下同步输出的 PWM 输出示意图（此时上升沿 cmp 与下降沿 cmp 相同）

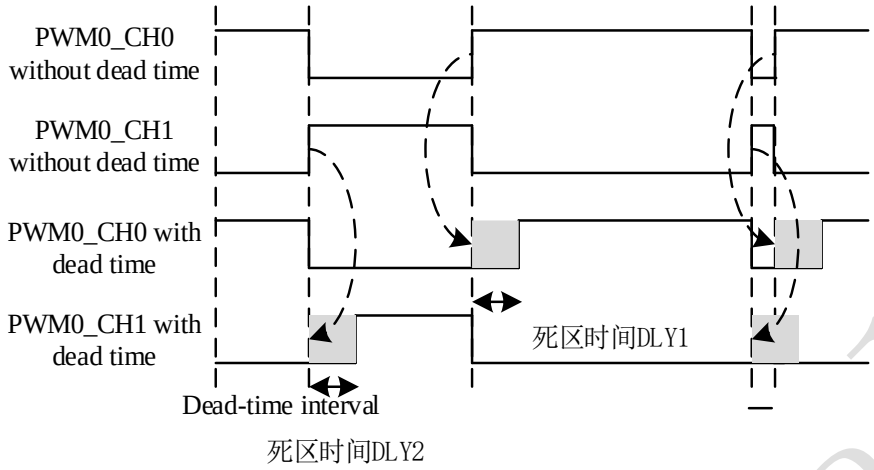


图 14-4 组模式下互补输出的 PWM 输出示意图（上升沿 cmp 与下降沿 cmp 相同）

14.4 寄存器列表

Register	Offset	R/W	Description	Reset Value
PWM_BASE ： 0x3000				
PWM_CTL	0x00	R/W	PWM 控制寄存器	0x00
PWM_DIV	0x01	R/W	PWM 时钟源选择寄存器	0x00
PWM_CMPn	0x04~0x0f	R/W	PWM 比较寄存器	0x00
PWM_DLY1	0x10	R/W	PWM 下降沿死区寄存器	0x00
PWM_DLY2	0x11	R/W	PWM 上升沿死区寄存器	0x00
PWM_INV	0x12	R/W	PWM 极性控制寄存器	0x00
PWM_INTEN	0x13	R/W	PWM 中断使能寄存器	0x00
PWM_INTSTS	0x14	R/W	PWM 中断状态寄存器	0x00
PWM_POEN	0x15	R/W	PWM 通道输出使能寄存器	0x00
PWM_PERIODn	0x16~0x21	R/W	PWM 周期寄存器	0x00

14.5 寄存器描述

14.5.1 PWM Control Register (PWM_CTL)

Register	Offset	R/W	Description	Reset Value
PWM_CTL	0x00	R/W	PWM 控制寄存器	0x00
Bits	**Descriptions**			
[7]	MODE	PWM 操作模式选择，0 = 独立模式.，1 = 组模式.		
[6]	COMEN	0 = 同步模式，1 = 互补模式 PWM0_CH0&PWM0_CH1、PWM0_CH2 &PWM0_CH3、 PWM0_CH4 &PWM0_CH5 （在组模式下才有效）		
[5]	CNTEN5	PWM 通道 5 开始使能 0 = 相关通道计数停止使能		

		1 = 相关通道计数开始使能.
[4]	CNTEN4	PWM 通道 4 开始使能 0 = 相关通道计数停止使能 1 = 相关通道计数开始使能.
[3]	CNTEN3	PWM 通道 3 开始使能 0 = 相关通道计数停止使能 1 = 相关通道计数开始使能.
[2]	CNTEN2	PWM 通道 2 开始使能 0 = 相关通道计数停止使能 1 = 相关通道计数开始使能.
[1]	CNTEN1	PWM 通道 1 开始使能 0 = 相关通道计数停止使能 1 = 相关通道计数开始使能.
[0]	CNTEN0	PWM 通道 0 开始使能 0 = 相关通道计数停止使能 1 = 相关通道计数开始使能.

14.5.2 PWM Clk Divide Register(PWM_DIV)

Register	Offset	R/W	Description	Reset Value
PWM_DIV	0x01	R/W	PWM 时钟源选择寄存器	0x00
Bits	**Descriptions**			
[7:3]	Reserved	Reserved		
[2:0]	CLK_DIV	000 = 时钟输入 *.clock divider:1* 001 = 时钟输入 / 2.*clock divider:2* 010 = 时钟输入 / 4.*clock divider:4* 100 = 时钟输入 / 8 *clock divider:8* Others = 时钟输入，即不分频		

14.5.3 PWM Compare Register(PWM_CMP)

Register	Offset	R/W	Description	Reset Value
PWM_CMP01	0x04	R/W	PWM0 比较寄存器（低 8 位）	0x00
PWM_CMP02	0x05	R/W	PWM0 比较寄存器（高 4 位）	0x00
PWM_CMP11	0x06	R/W	PWM1 比较寄存器（低 8 位）	0x00
PWM_CMP12	0x07	R/W	PWM1 比较寄存器（高 4 位）	0x00
PWM_CMP21	0x08	R/W	PWM2 比较寄存器（低 8 位）	0x00
PWM_CMP22	0x09	R/W	PWM2 比较寄存器（高 4 位）	0x00
PWM_CMP31	0x0A	R/W	PWM3 比较寄存器（低 8 位）	0x00
PWM_CMP32	0x0B	R/W	PWM3 比较寄存器（高 4 位）	0x00
PWM_CMP41	0x0C	R/W	PWM4 比较寄存器（低 8 位）	0x00

PWM_CMP42	0x0D	R/W	PWM4 比较寄存器（高 4 位）	0x00
PWM_CMP51	0x0E	R/W	PWM5 比较寄存器（低 8 位）	0x00
PWM_CMP52	0x0F	R/W	PWM5 比较寄存器（高 4 位）	0x00

注：软件使用上不允许 cmp 配成 0 或者大于等于 period，造成波形异常

Register	Offset	R/W	Description	Reset Value
PWM_CMP01	PWM_BA+0x04	R/W	PWM01 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP01	(1) (0,1)通道上升沿比较寄存器（低 8 位，组模式） (2) 0 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP02	PWM_BA+0x05	R/W	PWM02 比较寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[3:0]	PWM_CMP02	（1）通道（0,1） 上升沿比较寄存器（高 4 位，组模式） （2）0 通道比较寄存器（高 4 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP11	PWM_BA+0x06	R/W	PWM11 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP11	（1）通道（0,1） 下降沿比较寄存器（低 8 位，组模式） （2）1 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP12	PWM_BA+0x07	R/W	PWM12 比较寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[3:0]	PWM_CMP12	（1）通道（0,1）下降沿比较寄存器（高 4 位，组模式） （2）1 通道比较寄存器（高 4 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP21	PWM_BA+0x08	R/W	PWM21 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP21	(3) (2,3)通道上升沿比较寄存器（低 8 位，组模式） (4) 2 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP22	PWM_BA+0x09	R/W	PWM22 比较寄存器（高 4 位）	0x00

Bits	Descriptions	
[3:0]	PWM_CMP22	（1）通道（2,3） 上升沿比较寄存器（高 4 位，组模式） （2）2 通道比较寄存器（高 4 位，单独模式）

Register	Offset	R/W	Description	Reset Value
PWM_CMP31	PWM_BA+0x0a	R/W	PWM31 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP31	（1）通道（2,3） 下降沿比较寄存器（低 8 位，组模式） （2）3 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP32	PWM_BA+0x0b	R/W	PWM32 比较寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[3:0]	PWM_CMP32	（1）通道（2,3）下降沿比较寄存器（高 4 位，组模式） （2）3 通道比较寄存器（高 4 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP41	PWM_BA+0x0c	R/W	PWM41 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP41	（1）(4,5)通道上升沿比较寄存器（低 8 位，组模式） （2）4 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP42	PWM_BA+0x0d	R/W	PWM42 比较寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[3:0]	PWM_CMP42	（1）通道（4,5） 上升沿比较寄存器（高 4 位，组模式） （2）4 通道比较寄存器（高 4 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP51	PWM_BA+0x0e	R/W	PWM51 比较寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_CMP51	(1) 通道（4,5） 下降沿比较寄存器（低 8 位，组模式） (2) 5 通道比较寄存器（低 8 位，单独模式）		

Register	Offset	R/W	Description	Reset Value
PWM_CMP52	PWM_BA+0x0f	R/W	PWM52 比较寄存器（高 4 位）	0x00

Bits	Descriptions	
[3:0]	PWM_CMP52	（1）通道（4,5）下降沿比较寄存器（高 4 位，组模式） （2）5 通道比较寄存器（高 4 位，单独模式）

14.5.4 PWM Delay Register1(PWM_DLY1)

Register	Offset	R/W	Description	Reset Value
PWM_DLY1	0x10	R/W	PWM 下降沿死区计数器	0x00
Bits	**Descriptions**			
[7]	Reserved	Reserved		
[6]	PWM_DLY1_EN	PWM 0/2/4 通道计数器使能控制位		
[5:0]	PWM_DLY1	PWM 0/2/4 通道死区计数器，其区间为 PWM_DLY1+1		

14.5.5 PWM Delay Register2(PWM_DLY2）

Register	Offset	R/W	Description	Reset Value
PWM_DLY2	0x11	R/W	PWM 上升沿死区计数器	0x00
Bits	**Descriptions**			
[7]	Reserved	Reserved		
[6]	PWM_DLY2_EN	PWM 1/3/5 通道计数器使能控制位		
[5:0]	PWM_DLY2	PWM 1/3/5 通道死区计数器，其区间为 PWM_DLY2+1		

14.5.6 PWM Invert Register(PWM_INV)

Register	Offset	R/W	Description	Reset Value
PWM_INV	0x12	R/W	PWM 极性控制寄存器	0x00
Bits	**Descriptions**			
[7:6]	Reserved	Reserved		
[5]	PINV5	PWM 通道 5 反向使能 0 = 输出极性翻转不使能 1 = 输出极性翻转使能.		
[4]	PINV4	PWM 通道 4 反向使能 0 = 输出极性翻转不使能 1 = 输出极性翻转使能.		
[3]	PINV3	PWM 通道 3 反向使能 0 = 输出极性翻转不使能 1 = 输出极性翻转使能.		
[2]	PINV2	PWM 通道 2 反向使能 0 = 输出极性翻转不使能		

		1 = 输出极性翻转使能.
[1]	PINV1	PWM 通道 1 反向使能 0 = 输出极性翻转不使能 1 = 输出极性翻转使能.
[0]	PINV0	PWM 通道 0 反向使能 0 = 输出极性翻转不使能 1 = 输出极性翻转使能.

14.5.7 PWM Interrupt Enable Register(PWM_INTEN)

Register	Offset	R/W	Description	Reset Value
PWM_INTEN	0x13	R/W	PWM 中断使能寄存器	0x00
Bits	**Descriptions**			
[7:6]	Reserved	Reserved		
[5]	PWM_INTEN5	通道 5 周期中断使能位		
[4]	PWM_INTEN4	通道 4 周期中断使能位		
[3]	PWM_INTEN3	通道 3 周期中断使能位		
[2]	PWM_INTEN2	通道 2 周期中断使能位		
[1]	PWM_INTEN1	通道 1 周期中断使能位		
[0]	PWM_INTEN0	通道 0 周期中断使能位		

14.5.8 PWM Interrupt Status Register(PWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
PWM_INTSTS	0x14	R/W	PWM 中断状态寄存器	0x00
Bits	**Descriptions**			
[7:6]	Reserved	Reserved		
[5]	PWM_INTSTS5	通道 6 PWM 零点中断标志信号 标志信号将在 PWM5_CHn 的计数器计数到零点时硬件拉起. Note: 此 bit 可以被软件写 1 清 0 .		
[4]	PWM_INTSTS4	通道 5 PWM 零点中断标志信号 标志信号将在 PWM4_CHn 的计数器计数到零点时硬件拉起. Note: 此 bit 可以被软件写 1 清 0 .		
[3]	PWM_INTSTS3	PWM 周期中断标志信号 标志信号将在 PWM3_CHn 的计数器计数到零点时硬件拉起. Note: 此 bit 可以被软件写 1 清 0 .		
[2]	PWM_INTSTS2	PWM 周期中断标志信号 标志信号将在 PWM2_CHn 的计数器计数到零点时硬件拉起.		

		Note: 此 bit 可以被软件写 1 清 0 .
[1]	PWM_INTSTS1	PWM 周期中断标志信号 标志信号将在 PWM1_CHn 的计数器计数到零点时硬件拉起. Note: 此 bit 可以被软件写 1 清 0 .
[0]	PWM_INTSTS0	PWM 周期中断标志信号 标志信号将在 PWM0_CHn 的计数器计数到零点时硬件拉起. Note: 此 bit 可以被软件写 1 清 0 .

14.5.9 PWM Output Enable Register(PWM_OEN)

Register	Offset	R/W	Description	Reset Value
PWM_POEN	0x15	R/W	PWM 通道输出使能寄存器	0x00
Bits	**Descriptions**			
[7:6]	Reserved	Reserved		
[5:0] n=0,1..5	POENn	PWM 通道输出使能 0 = PWM 通道禁止使能输出到相应 pin. 1 = PWM 通道使能输出到相应 pin.		

14.5.10 PWM Period Register(PWM_PERIOD)

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD01	0x16	R/W	PWM0 周期寄存器（低 8 位）	0x00
PWM_PERIOD02	0x17	R/W	PWM0 周期寄存器（高 4 位）	0x00
PWM_PERIOD11	0x18	R/W	PWM1 周期寄存器（低 8 位）	0x00
PWM_PERIOD12	0x19	R/W	PWM1 周期寄存器（高 4 位）	0x00
PWM_PERIOD21	0x1A	R/W	PWM2 周期寄存器（低 8 位）	0x00
PWM_PERIOD22	0x1B	R/W	PWM2 周期寄存器（高 4 位）	0x00
PWM_PERIOD31	0x1C	R/W	PWM3 周期寄存器（低 8 位）	0x00
PWM_PERIOD32	0x1D	R/W	PWM3 周期寄存器（高 4 位）	0x00
PWM_PERIOD41	0x1E	R/W	PWM4 周期寄存器（低 8 位）	0x00
PWM_PERIOD42	0x1F	R/W	PWM4 周期寄存器（高 4 位）	0x00
PWM_PERIOD51	0x20	R/W	PWM5 周期寄存器（低 8 位）	0x00
PWM_PERIOD52	0x21	R/W	PWM5 周期寄存器（高 4 位）	0x00

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD01	0x16	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD01	独立模式的通道 0(或组模式的通道 0/1)period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时		

		间，否则会导致上一次中断未处理完成，产生新的中断请求

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD02	0x17	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD02	独立模式的通道 0 (或组模式的通道 0/1)period 寄存器（高 4 位）		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD11	0x18	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD11	独立模式的通道 1 period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时间，否则会导致上一次中断未处理完成，产生新的中断请求		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD12	0x19	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD12	独立模式的通道 1 period 寄存器（高 4 位）		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD21	0x1A	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD21	独立模式的通道 2 (或组模式的通道 2/3) period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时间，否则会导致上一次中断未处理完成，产生新的中断请求		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD22	0x1B	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD22	独立模式的通道 2(或组模式的通道 2/3) period 寄存器（高 4 位）		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD31	0x1C	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD31	独立模式的通道 3 period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时		

		间，否则会导致上一次中断未处理完成，产生新的中断请求

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD32	0x1D	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD32	独立模式的通道 3 period 寄存器（高 4 位）		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD41	0x1E	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD41	独立模式的通道 4 (或组模式的通道 4/5)period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时间，否则会导致上一次中断未处理完成，产生新的中断请求		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD42	0x1F	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD42	独立模式的通道 4(或组模式的通道 4/5) period 寄存器（高 4 位）		

Register	Offset	R/W	Description	Reset Value
PWM_PERIOD51	0x20	R/W	PWM 周期寄存器（低 8 位）	0x00
Bits	**Descriptions**			
[7:0]	PWM_PERIOD51	独立模式的通道 5 period 寄存器（低 8 位） 注：开启多通道 PWM 时，要注意 PWM 周期大于软件处理中断的时间，否则会导致上一次中断未处理完成，产生新的中断请求		

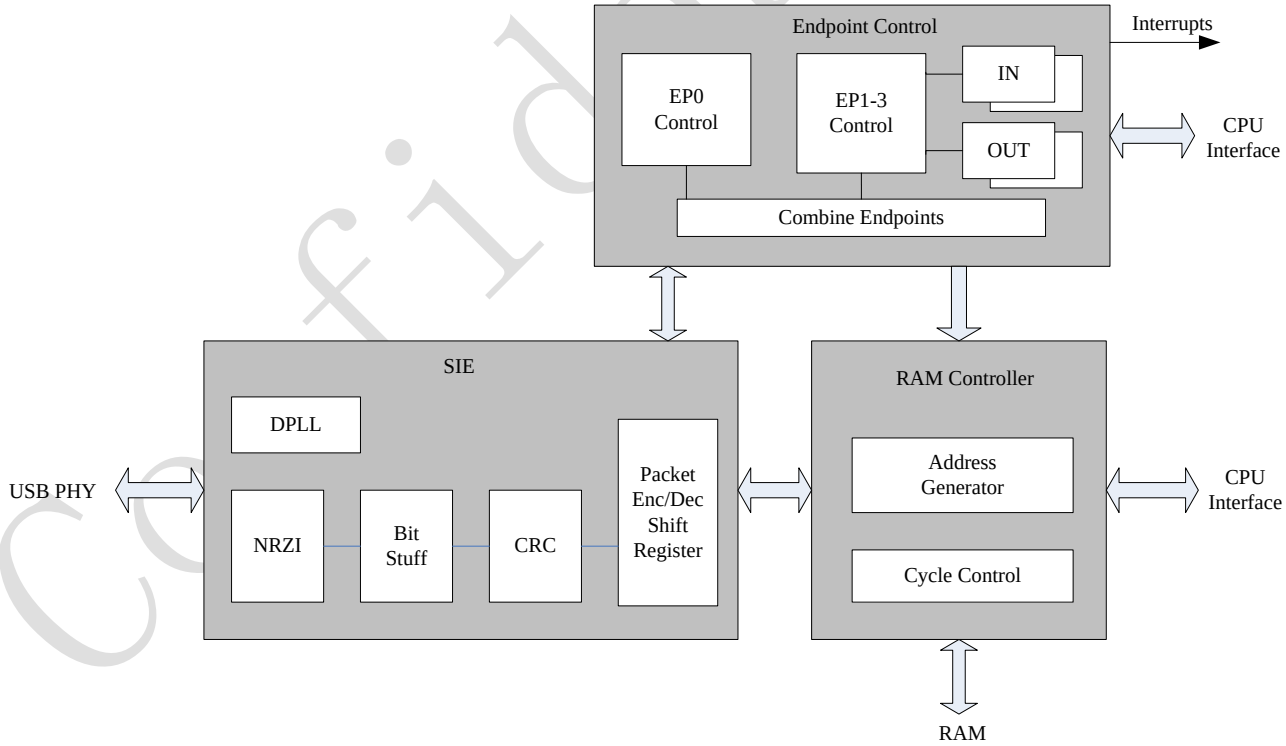
Register	Offset	R/W	Description	Reset Value
PWM_PERIOD52	0x21	R/W	PWM 周期寄存器（高 4 位）	0x00
Bits	**Descriptions**			
[7:4]	Reserved	Reserved		
[3:0]	PWM_PERIOD52	独立模式的通道 5 period 寄存器（高 4 位）		

第15章 USB

15.1 特征

- 支持USB2.0全速模式，Full-speed 12Mbps
- 支持EndPoint个数为4个，EndPoint0+EndPoint1/2/3。
- Endpoint0的PHY层TX/RX FIFO各32byte
- Endpoint1-3采用Bulk传输方式，PHY层TX/RX FIFO各32byte
- EndPoint3采用ISO传输，PHY层TX/RX FIFO最大32byte
- 支持SOF、Reset、Resume、Suspend、EndPointx(x=0-3)、拔插中断
- 支持NRZI编解码，以及bit位填充
- 支持挂起与恢复信号

15.2 模块框图

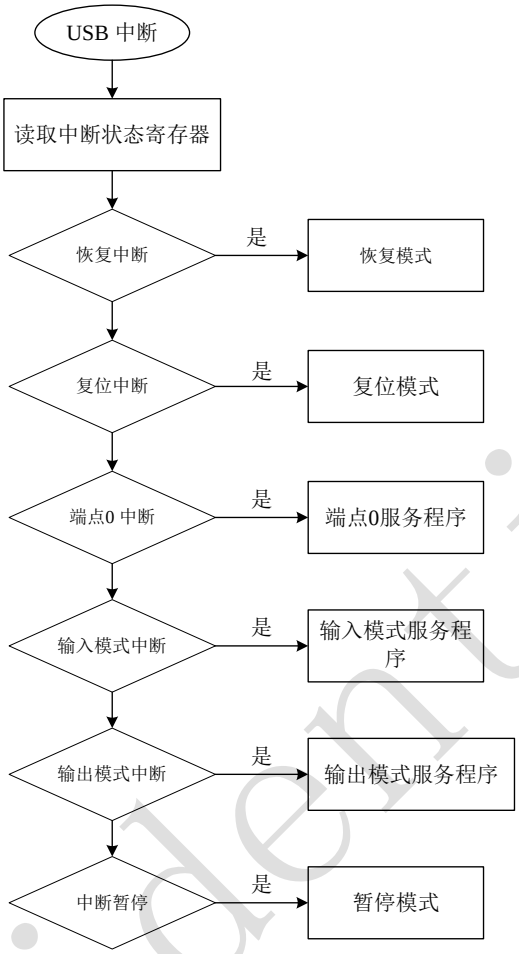


15.3 功能描述

15.3.1 USB 中断处理

当USB发起中断时，CPU需要读取中断状态寄存器，以确定哪个端点导致中断并跳到适当的

中断服务程序。如果多个端点导致中断，则应先服务端点0，其次是其他端点。



15.3.2 USB 复位

当在 USB 上检测到复位时，PAN262x 执行如下操作：

- 将地址设置成0
- 将索引设置为0
- 清空所有端点的fifo
- 清空所有控制以及状态寄存器
- 使能所有中断，除了暂停功能.
- 产生复位中断

当软件接收到复位中断，其需要关掉所有打开的管道并等待总线枚举开始。

15.3.3 暂停/恢复

当 USB 没有活动持续时间达到 3ms，其将产生暂停中断请求（如果暂停中断使能）。将由软件决定在 USB 进入暂停模式时，或是关掉 USB。

15.3.4 暂停期间激活

USB 通过在总线上发送恢复信号来退出暂停模式。如果 USB 处于暂停模式时，USB 将处于挂起状态，并一直监控 USB 重新发起恢复信号。当恢复信号发起，USB 将产生恢复启动中断。

15.3.5 暂停期间不活动

当接收到上述暂停中断时，软件可以通过停止其时钟来禁用USB（须通过时钟模块来完成）。但是，由于 USB 被禁用，因此它将无法检测 USB 上的恢复信号。因此，需要一些外部硬件来检测恢复信号（通过监测 DIM 和 DIP 信号），以便可以重新启动 USB。

15.3.6 远程唤醒

如果 USB 处于挂起模式，并且软件希望启动远程唤醒，则应写入电源寄存器，以将恢复位置为 1（如果 USB 的时钟已停止，则需要重新启动，然后才能进行此写入）。软件应将该位保留约 10 毫秒（最少 2 毫秒，最多 15 毫秒），然后将其重置为 0。此时 hub 应已接管 USB 上的驱动恢复信号。

注意：当软件启动远程唤醒时，不会产生恢复中断。

15.3.7 端点 0 处理

标准设备请求可分为三类：

- 零数据请求，所有信息都包含在命令中
- 写请求，命令后面会有额外的数据
- 读取请求，需要设备将数据发送回主机

15.3.8 零数据请求

零数据请求的所有信息都包含在 8 字节的命令中，不需要传输额外的数据。

15.3.9 写请求

写请求会在 8 字节命令之后从主机发送的额外数据包。

15.3.10 读请求

读取请求，需要设备将数据发送回主机。

15.4 USB 模块使用 RAM 空间

片内为 2K Bytes 的外部数据存储，其中高 1K 的空间和 USB 的 FIFO 共用，当使用 USB 时该空间不能使用。即 USB 模块使能后，0x400~0x7FF 范围内 RAM 由 USB 模块控制，USB_RAM_BASE 为 0x400。各个 EndPoint 使用内存空间为：

Endpoint	方向	地址范围
0	/	USB_RAM_BASE+0x000~ USB_RAM_BASE+0x03F
1	IN	USB_RAM_BASE+0x040~ USB_RAM_BASE+0x0BF
	OUT	USB_RAM_BASE+0x0C0~ USB_RAM_BASE+0x13F
2	IN	USB_RAM_BASE+0x140~ USB_RAM_BASE+0x1BF
	OUT	USB_RAM_BASE+0x1C0~ USB_RAM_BASE+0x23F
3	IN	USB_RAM_BASE+0x240~ USB_RAM_BASE+0x2BF
	OUT	USB_RAM_BASE+0x2C0~ USB_RAM_BASE+0x33F

对于 Endpoint1-3，当设置 InMaxP/OutMaxP 寄存器小于等于最大值（16）的一半，即将该寄存器设置为小于等于 8 时，硬件会开启双缓冲，能够存储两个 USB 数据包。相应的地址区域会划分为两个区域，用于分别存储数据包。开启双缓冲后，各个数据包使用 RAM 区域如下：

Endpoint	方向	数据包	地址范围
1	IN	1	USB_RAM_BASE+0x040~ USB_RAM_BASE+0x07F
		2	USB_RAM_BASE+0x080~ USB_RAM_BASE+0x0BF
	OUT	1	USB_RAM_BASE+0x0C0~ USB_RAM_BASE+0x0FF
		2	USB_RAM_BASE+0x100~ USB_RAM_BASE+0x13F
2	IN	1	USB_RAM_BASE+0x140~ USB_RAM_BASE+0x17F
		2	USB_RAM_BASE+0x180~ USB_RAM_BASE+0x1BF
	OUT	1	USB_RAM_BASE+0x1C0~ USB_RAM_BASE+0x1FF
		2	USB_RAM_BASE+0x200~ USB_RAM_BASE+0x23F
3	IN	1	USB_RAM_BASE+0x240~ USB_RAM_BASE+0x27F
		2	USB_RAM_BASE+0x280~ USB_RAM_BASE+0x2BF
	OUT	1	USB_RAM_BASE+0x2C0~ USB_RAM_BASE+0x2FF
		2	USB_RAM_BASE+0x300~ USB_RAM_BASE+0x33F

15.5 寄存器列表

Register	Offset	R/W	Description	Reset Value
USB Base Address: **USB_BA = 0x4000**				
FADDR	USB_BA+0x00	R	功能地址寄存器	8'h00
POWER	USB_BA+0x01	R/W	电源管理寄存器	8'h00
IntrIn1	USB_BA+0x02	R/W	端点 0 与输入端点中断寄存器	8'h00
IntrOut1	USB_BA+0x04	R/W	输出端点中断寄存器	8'h00
IntrUSB	USB_BA+0x06	R/W	通用 USB 中断寄存器	8'h00
IntrIn1E	USB_BA+0x07	R	输入端点中断使能寄存器	8'hFF
In-trOut1E	USB_BA+0x09	R/W	输出端点中断使能寄存器	8'hFE

In-trUSBE	USB_BA+0x0B	R	通用 USB 中断使能寄存器	8'h06
Frame1	USB_BA+0x0C	R/W	帧数寄存器 0 到 7	8'b00000000
Frame2	USB_BA+0x0D	R/W	帧数寄存器 8 到 15	3'b000
Index	USB_BA+0x0E	R	索引寄存器	4'b0000
InMaxP	USB_BA+0x10	R	输入端点的 pack size 寄存器	8'h00
CSR0	USB_BA+0x11	R/W	端点 0 控制与状态寄存器	8'h00
InCSR1	USB_BA+0x11	R/W	输入端点的控制与状态寄存器 1	8'h00
InCSR2	USB_BA+0x12	R/W	输入端点的控制与状态寄存器 2	8'h20
Out-MaxP	USB_BA+0x13	R	输出端点的 pack size 寄存器	8'h00
Out-CSR1	USB_BA+0x14	R/W	输出端点的控制与状态寄存器 1	8'h00
Out-CSR2	USB_BA+0x15	R	输出端点的控制与状态寄存器 2	8'h00
Count0	USB_BA+0x16	W	端点 0 接收数据的数量寄存器	7'h00
Out-Count1	USB_BA+0x16	W	输出端点数据的数量寄存器（低字节）	8'b00000000
Out-Count2	USB_BA+0x17	W	输出端点数据的数量寄存器（高字节）	3'b000
FIFOx	USB_BA+0x20/0x24/0x28/0x2C		端点 0 到 3 fifo 地址寄存器	

15.6 寄存器描述

15.6.1 FADDR

Register	Offset	R/W	Description	Reset Value
FADDR	USB_BA+0x00	R	功能地址寄存器	8'h00

Bits	Description	
[7]	Update	更新当前地址，当新地址有效时（当前传输的最后阶段），该位自动清 0
[6:0]	Func Addr	地址

15.6.2 POWER

Register	Offset	R/W	Description	Reset Value
POWER	USB_BA+0x01	R/W	电源管理寄存器	8'h00

Bits	Description	
[7]	ISO Update	由 CPU 设置时，在发送数据包之前等待从 InPktRdy 开始的 SOF 令牌。如果在 SOF 令牌之前接收到 IN 令牌，则将发送零长度数据包。此位仅由执行同步传输的端点使用。

[6:4]	Reserved	Reserved
[3]	Reset	当总线上出现复位信号时，该位将被置 1，并且仅只读
[2]	Resume	由 CPU 设置，其在暂停模式时生成恢复信号。置 1 之后，CPU 应在 10 ms（最多 15 ms）后清除此位，以结束恢复信号
[1]	Suspend Mode	当进入暂停模式时，其会被 USB 置 1。当 CPU 读取中断寄存器或设置该寄存器的“Resume”位时清除。
[0]	Enable Suspend	使能暂停模式，当总线上接收到暂停信号时，允许进入暂停模式

15.6.3 INTRIN1

Register	Offset	R/W	Description	Reset Value
INTRIN1	USB_BA+0x02	R/W	端点 0 与输入端点中断寄存器	8’h00

Bits	Description	
[7:4]	Reserved	Reserved
[3]	EP3	输入端点 3 中断
[2]	EP2	输入端点 2 中断
[1]	EP1	输入端点 1 中断
[0]	EP0	端点 0 中断

15.6.4 INTROUT1

Register	Offset	R/W	Description	Reset Value
INTROUT1	USB_BA+0x04	R/W	输出端点中断寄存器	8’h00

Bits	Description	
[7:4]	Reserved	Reserved
[3]	EP3	输出端点 3 中断
[2]	EP2	输出端点 2 中断
[1]	EP1	输出端点 1 中断
[0]	Reserved	Reserved

15.6.5 INTRUSB

Register	Offset	R/W	Description	Reset Value
INTRUSB	USB_BA+0x06	R/W	通用 USB 中断寄存器	8’h00

Bits	Description	
[7]	DPSTA	实时体现 D+状态，读取不清零

[6]	DMSTA	实时体现 D-状态，读取不清零
[5]	Reserved	Reserved
[4]	PUPIF	PUPIF 中断到来时，读取此 bit，若为 1，则表示发生了 device 插拔事件，此 bit 读取清零
[3]	SOF	在每帧开始时，该位需要被置 1
[2]	Reset	当复位信号出现时，该位将被置 1
[1]	Resume	在暂停模式下，当重新发起信号出现时，该位将被置 1
[0]	Suspend	当暂停信号出现时，该位将被置 1

15.6.6 INTRIN1E

Register	Offset	R/W	Description	Reset Value
INTRIN1E	USB_BA+0x07	R	输入端点中断使能寄存器	8'hFF

Bits	Description	
[7:4]	Reserved	Reserved
[3]	EP3_EN	输入端点 3 中断使能
[2]	EP2_EN	输入端点 2 中断使能
[1]	EP1_EN	输入端点 1 中断使能
[0]	EP0_EN	端点 0 中断使能

15.6.7 INTROUT1E

Register	Offset	R/W	Description	Reset Value
INTROUT1E	USB_BA+0x09	R/W	输出端点中断使能寄存器	8'hFE

Bits	Description	
[7:4]	Reserved	Reserved
[3]	EP3_EN	输出端点 3 中断使能
[2]	EP2_EN	输出端点 2 中断使能
[1]	EP1_EN	输出端点 1 中断使能
[0]	Reserved	Reserved

15.6.8 INTRUSBE

Register	Offset	R/W	Description	Reset Value
INTRUSBE	USB_BA+0x0B	R	通用 USB 中断使能寄存器	8'h06

Bits	Description

[7:5]	Reserved	Reserved
[4]	PUPIF_EN	插拔中断使能位
[3]	SOF_EN	每帧中断使能位
[2]	Reset_EN	复位中断使能位
[1]	Resume_EN	重新发起中断使能位
[0]	Suspend_EN	暂停中断使能位

15.6.9 FRAME1

Register	Offset	R/W	Description	Reset Value
FRAME1	USB_BA+0x0C	R/W	帧数寄存器 0 到 7	8'b00000000

Bits	Description	
[7:0]	Frame1	帧数的低字节

15.6.10 FRAME2

Register	Offset	R/W	Description	Reset Value
FRAME2	USB_BA+0x0D	R/W	帧数寄存器 8 到 15	3'b000

Bits	Description	
[2:0]	Frame2	帧数的高字节

15.6.11 INDEX

Register	Offset	R/W	Description	Reset Value
INDEX	USB_BA+0x0E	R	索引寄存器	4’b0000

Bits	Description	
[3:0]	Index	确定在寄存器地址 10h 到 17h 访问哪些端点控制/状态寄存器。 每个输入端点和每个输出端点都有自己的一组控制/状态寄存器。一次映射中只显示一组 IN-control/status 和一组 OUT-control/status 寄存器。在访问端点的控制/状态寄存器之前，应该将端点编号写入索引寄存器，以确保在地址映射中显示正确的控制/状态寄存器。

15.6.12 INMAXP

Register	Offset	R/W	Description	Reset Value
INMAXP	USB_BA+0x10	R	输入端点的 pack size 寄存器	8’h00

Bits	Description

[7:0]	InMaxP	最大数据包大小/事务。 它保存通过当前选定的端点的事务的最大数据包大小（以 8 字节为单位），当值设置成 128 时，其将最大数据包大小设置为 1023（全速事务中传输的同步数据包的最大大小）而不是 1024。 如果该值大于为端点配置的 FIFO 的大小，该值将自动更新为 FIFO 的大小。如果写入该寄存器的值小于或等于 FIFO 大小的一半，则可以缓冲为两个 IN 数据包。如果在端点发送数据包后更改此寄存器，则在将新值写入此寄存器后，应完全刷新端点中的 IN FIFO。

15.6.13 CSR0

Register	Offset	R/W	Description	Reset Value
CSR0	USB_BA+0x11	R/W	端点 0 控制与状态寄存器	8'h00

Bits	Description	
[7]	ServicedSetupEnd	CPU 向该位写入 1 以清除 SetupEnd 位。该位自动清除。
[6]	ServicedOutPktRdy	CPU 向该位写入 1 以清除 OutPktRdy 位。该位自动清除。
[5]	SendStall	CPU 向该位写入 1 以终止当前事务。摆摊握手将被传输，然后该位将被自动清除。
[4]	SetupEnd	当控制事务在 DataEnd 置位之前结束时，该位将被置位，此时将生成中断并刷新 FIFO。当 CPU 向 ServicedSetupEnd 位写入 1 时，该位将被清除。
[3]	DataEnd	CPU 将该位置位， 1.为最后一个数据包设置 InPktRdy 时。 2.在卸载最后一个数据包后清除 OutPktRdy 时。 3.为零长度数据包设置 InPktRdy 时。 该位硬件自动清零
[2]	SentStall	此位在 STALL 握手时被置位，该位需要由 CPU 清除。
[1]	InPktRdy	CPU 在将数据包装入 FIFO 后，该位将被置位。当数据包被传输时，该位将被自动清除。当该位清除时，将产生中断。
[0]	OutPktRdy	该位在接收到数据包时被置位，并产生中断。CPU 通过设置 ServicedOutPktRdy 位来清除该位。

15.6.14 INCSR1

Register	Offset	R/W	Description	Reset Value
INCSR1	USB_BA+0x11	R/W	输入端点的控制与状态寄存器 1	8'h00

Bits	Description	
[7]	Reserved	Reserved
[6]	ClrDataTog	CPU 将 1 写入该位，以将数据端点中的输入数据重置到零。
[5]	SentStall	该位在传输 STALL handshake 时被置位。FIFO 被刷新，InPktRdy 位被清零。CPU 需要清除该位。

[4]	SendStall	CPU 将 1 写入该位，以向 IN token 发出 STALL handshake。CPU 清除此位以终止 stall condition。如果输入端点处于 ISO 模式，则此位无效。
[3]	FlushFIFO	CPU 将 1 写入该位，以刷新 FIFO 中要从端点传输的下一个数据包。FIFO 指针复位，InPktRdy 位被清零。 注意：如果 FIFO 包含两个数据包，则需要对 FlushFIFO 置位两次以完全清除 FIFO。
[2]	UnderRun	在 ISO 模式下，在接收到 IN token 并且 InPktRdy 没有被置位，零长度数据发之后，该位将被置位。在 Bulk/Interrupt 模式下，NAK 应答 IN token 之后，该位将被置位。该位的清零，需要由 CPU 进行清除。
[1]	FIFONotEmpty	当 in-FIFO 中至少有 1 个数据包时，该位将被置位。
[0]	InPktRdy	CPU 在将数据包装入 FIFO 后设置该位。当数据包被传输时，它被自动清除。当该位被清除时，将生成一个中断（如果中断使能）。

15.6.15 INCSR2

Register	Offset	R/W	Description	Reset Value
INCSR2	USB_BA+0x12	R/W	输入端点的控制与状态寄存器 2	8'h20

Bits	Description	
[7]	AutoSet	如果 CPU 设置此位，当最大数据包（InMaxP 中的值）的数据加载到 in-FIFO 时，InPktRdy 将自动置位。如果加载的数据包小于最大数据包大小，则必须手动设置 InPktRdy。
[6]	ISO	将该位置 1，启用输入端点进行同步传输（ISO 模式）；写 0，以将输入端点作为批量/中断传输。
[5]	Mode	写 1，将端点设置为输入；写 0 将端点设置为输出。其仅在使用相同的端点 FIFO 作为输入输出传输。
[4]	Reserved	Reserved
[3]	FrcDataTog	CPU 将此位置位，以强制端点在发送每个数据包之后进行切换，而不管是否接收到 ACK。这可被输入端点的中断所使用，作为同步端点的速率反馈使用。
[2:0]	Reserved	Reserved

15.6.16 OUTMAXP

Register	Offset	R/W	Description	Reset Value
OUTMAXP	USB_BA+0x13	R	输出端点的 pack size 寄存器	8'h00

Bits	Description

[7:0]	OutMaxP	输出最大数据包大小/事务。 其为设置通过当前选定的输出端点的事务的最大数据包大小（以 8 字节为单位），数值 128 表示将最大数据包大小设置为 1023（同步数据包的最大大小），而不是 1024。 写入该寄存器的值数据总量不得超过输出端点的 FIFO 大小，如果需要双缓冲，则不应超过 FIFO 大小的一半。如果一个大于输出 FIFO 大小的值被写入该寄存器，该值将自动更改为输出 FIFO 大小。如果写入该寄存器的值小于或等于输出 FIFO 大小的一半，则可以缓冲两个输出数据包。

15.6.17 OUTCSR1

Register	Offset	R/W	Description	Reset Value
OutCSR1	USB_BA+0x14	R/W	输出端点的控制与状态寄存器 1	8'h00

Bits	Description	
[7]	ClrDataTog	CPU 将 1 写入该位，以将数据端点中的输入数据重置到零。
[6]	SentStall	该位在传输 STALL handshake 时被置位。CPU 需要清除该位。
[5]	SendStall	写 1，以发起 STALL handshake。CPU 清除此位以终止 stall condition。如果输入端点处于 ISO 模式，则此位无效。
[4]	FlushFIFO	CPU 将 1 写入该位，以刷新从输出端点 FIFO 中的下一个数据包。 注意：如果 FIFO 包含两个数据包，则需要对 FlushFIFO 置位两次以完全清除 FIFO。
[3]	DataError	当 OutPktRdy 置位时，如果数据包有 CRC 或位填充错误，该位将被置位。当 OutPktRdy 被清除时，该位将被清除。该位仅在 ISO 模式下有效。
[2]	OverRun	如果输出数据包没有加载到输出 FIFO 中，该位将被置位。该位需要 CPU 清零，该位尽在 ISO 模式有效。
[1]	FIFOFull	若没有更多的数据包加载至输出 FIFO 中，该位将被置位。
[0]	OutPktRdy	该位在接收到数据包时将被置位。当数据包没有加载到输出 FIFO 中时，该位需要 CPU 进行清零。当该位被置位时，将触发中断。

15.6.18 OUTCSR2

Register	Offset	R/W	Description	Reset Value
OUTCSR2	USB_BA+0x15	R	输出端点的控制与状态寄存器 2	8'h00

Bits	Description	
[7]	AutoClear	如果设置该位，当 OutMaxP 字节的数据包从 OUT FIFO 取出时，OutPktRdy 位将自动清除。当取出的数据量小于最大数据包大小时，必须手动清除 OutPktRdy 位。
[6]	ISO	写 1，将使能输出端点的同步传输；写 0，使能输出端点批量/中断传输。
[5:0]	Reserved	Reserved

15.6.19 COUNT0

Register	Offset	R/W	Description	Reset Value
COUNT0	USB_BA+0x16	W	端点 0 接收数据的数量寄存器	7'h00

Bits	Description	
[6:0]	Count0	表示端点 0 FIFO 中接收的数据字节数。其值在 OutPktRdy（CSR0.D0）置位时有效。

15.6.20 OUTCOUNT1

Register	Offset	R/W	Description	Reset Value
OUTCOUNT1	USB_BA+0x16	W	输出端点数据的数量寄存器（低字节）	8'b00000000

Bits	Description	
[7:0]	OutCount1	当前选中的输出端点的接收 FIFO 数量值（低字节），其在 OutktrDy（OutCSR1.D0）置位时有效。

15.6.21 OUTCOUNT2

Register	Offset	R/W	Description	Reset Value
OUTCOUNT2	USB_BA+0x17	W	输出端点数据的数量寄存器（高字节）	3'b000

Bits	Description	
[7:0]	OutCount2	当前选中的输出端点的接收 FIFO 数量值（高字节，高 3 位），其在 OutktrDy（OutCSR1.D0）置位时有效。

15.6.22 FIFOx

每个端点对应 16 个供 CPU 访问的 FIFO 地址。写 FIFO 地址，即将数据写入 FIFO；读 FIFO 地址，即从 FIFO 中读取数据。

其中，端点 0 对应 FIFO 地址为 20h，端点 1 对应 FIFO 地址为 24h，端点 2 对应 FIFO 地址为 28h，端点 3 对应 FIFO 地址为 2Ch。

第16章 RF收发机

16.1 概述

PAN262x 2.4GHz GFSK RF 收发机工作在世界通用 ISM 频段 2.400~2.4835 GHz。作为 SOC 的一个外设，通信上兼容 BLE 广播包（4.2 短包）、XN297L 以及 24L01。

PAN262x 所有的 RFTR 寄存器挂在 xram bus 上，基地址 RF_ADD_BASE = 16`h5000，此文档只描述相对地址。表中未定义的寄存器，读取结果为“0”。

16.1.1 特性

PAN262x RF 收发机特性包括：

- 常规
  - 通信频段：2.400GHz ~2.483GHz
  - 调制方式：GFSK
  - 数据率：250 Kbps、1 Mbps、2 Mbps
  - 天线口：收发共用
- 射频综合器
  - 完全集成频率合成器
  - 1Mbps/2Mbps 模式（晶振精度±60ppm）
  - 250kbps 模式（晶振精度±20ppm）
- 发射机
  - 输出功率：13~-35dBm
  - 18mA@0dBm
- 接收机
  - -101 dBm @ 250 Kbps
  - -93 dBm @ 1 Mbps
  - -89 dBm @ 2 Mbps
  - 12.4mA@1Mbps
- 协议
  - 支持 1 到 64 字节数据长度可配
  - 支持自动应答及自动重传

- 6 个接收通道构成 1:6 的星状网络
- 支持 BLE 4.2 广播包
- 兼容 297L 帧结构
- 兼容 24L01 帧结构

16.1.2 框图

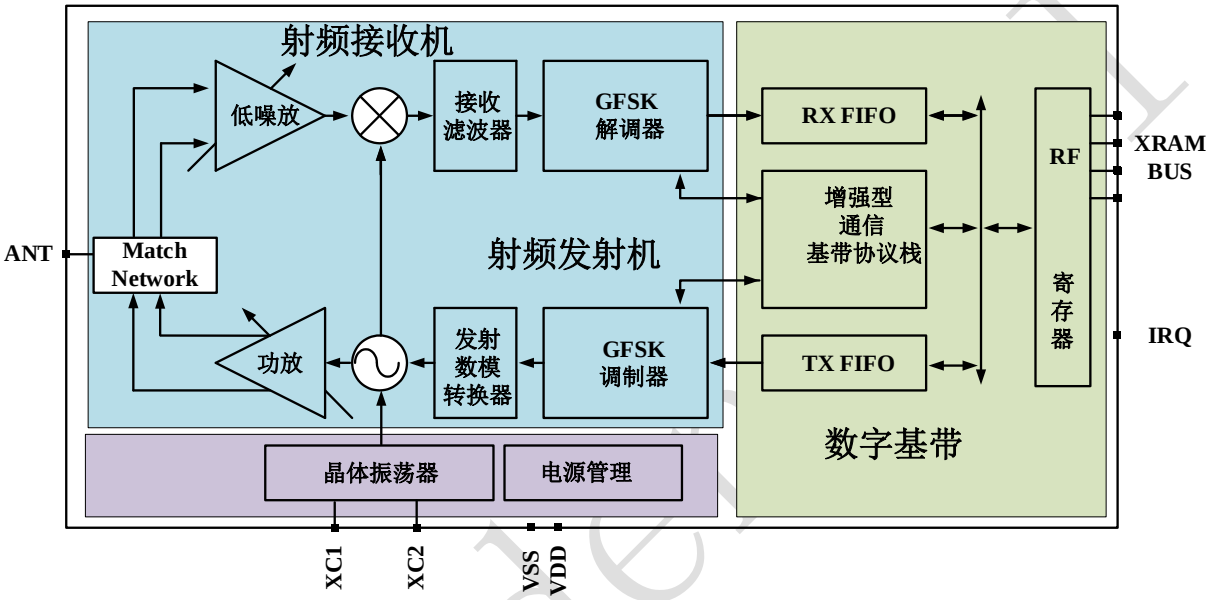


图 16.1　RF 收发机框图

16.1.3 功能描述

本小节描述 RF 收发机的不同工作模式以及其他一些功能点。

16.1.3.1 工作模式

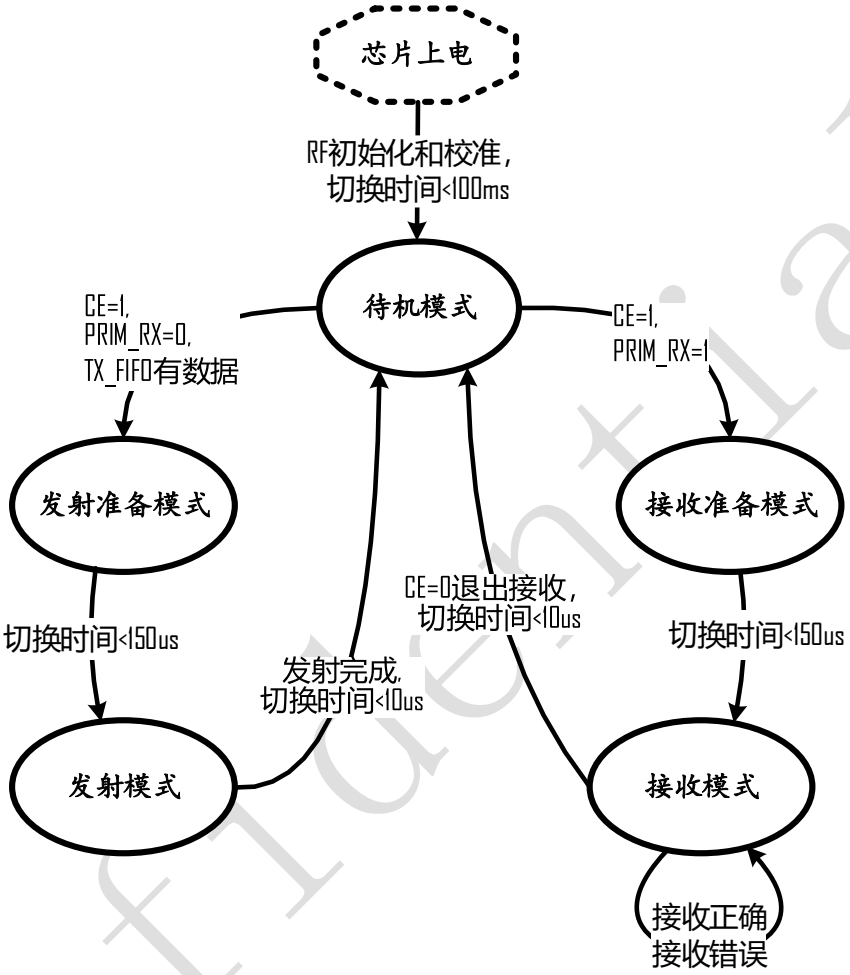
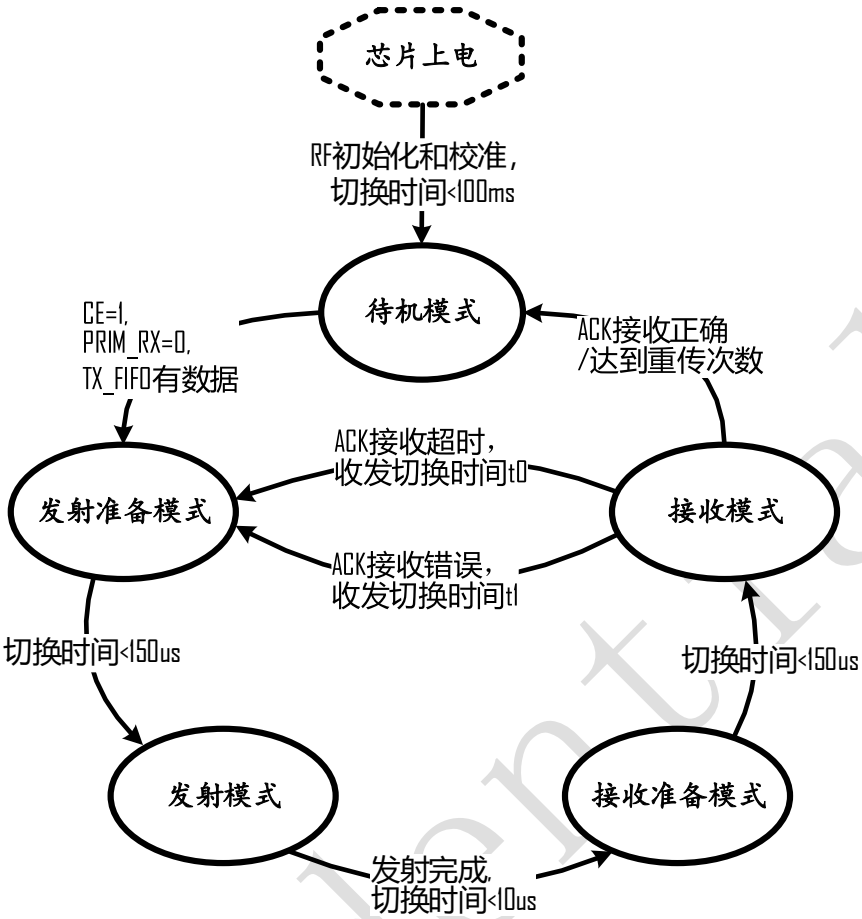


图 16.2　RF 收发机普通模式状态转移图



注：
① 接收超时时间t0由RX_ACK_TIME决定
② 收发切换时间t1由自动传输延时SETUP_RETR.ARD决定

图 16.3　RF 收发机增强型发射模式状态转移图

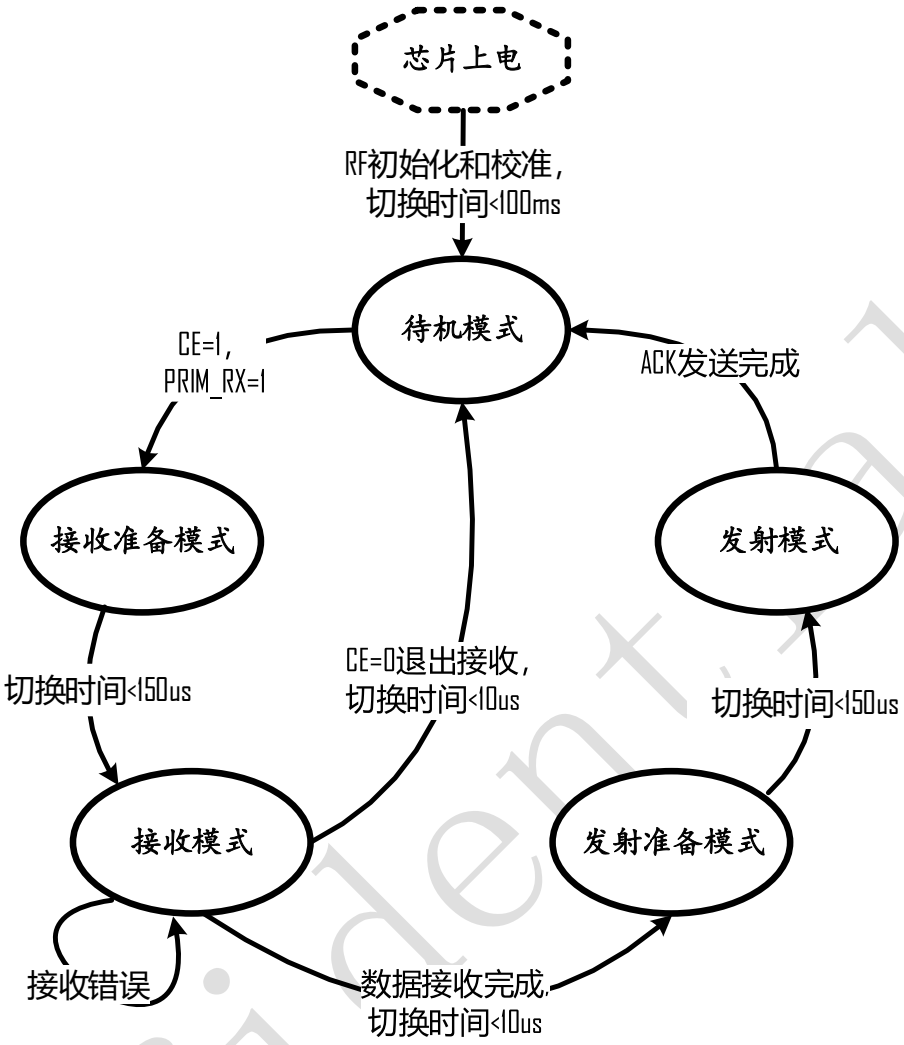


图 16.4　RF 收发机增强型接收模式状态转移图

RF 收发机有以下五种工作模式：普通型发射模式、普通型接收模式、增强型发射模式、增强型接收模式、待机模式。

- 普通型发射模式：收发机处于 Burst 发射状态，在 MCU 的控制下将 FIFO 中的数据发射出去并上报中断给 MCU
- 普通型接收模式：收发机一直处于接收状态，在收到有效数据后上报中断给 MCU
- 增强型发射模式：收发机一开始处于发射状态，发射完成后自动进入接收状态，等待 ACK 或者 ACK + PAYLOAD
- 增强型接收模式：收发机一开始处于接收状态，接收到有效数据后上报中断给 MCU 并且自动进入发射状态，发送 ACK 或者 ACK + PAYLOAD
- 待机模式：收发机处于待机状态，等待进一步的 MCU 指令

寄存器 / 模式	EN_DPLL	EN_XTH	RFCLKEN	PRIM_RX	RF_CE	ARC	ENAA_P0	FIFO
普通型发射	1	1	1	0	1	0000	0	TX FIFO
普通型接收	1	1	1	1	1	-	0	RX FIFO
增强型发射	1	1	1	0	1	> 0	0	TX/RX

								FIFO
增强型接收	1	1	1	1	1	-	1	RX/TX FIFO
待机模式	1	1	0	-	0	-	-	-

表 16.1 PAN262x RF 收发机工作模式

链路层如数据组帧、校验、地址判断、数据白化的扰码、数据重传和 ACK 响应等处理是由收发机内部完成的，无需 MCU 参与。

收发机可配置为两个不同的 RX FIFO 寄存器（32 字节）或者一个 RX FIFO 寄存器（64 字节）（6 个接收通道共享）、二个不同的 TX FIFO 寄存器（32 字节）或者一个 TX FIFO 寄存器（64 字节）。在 RF STB 模式和 PWR_DN 模式下，MCU 可以访问 FIFO 寄存器。

PAN262x 主要有两种数据通信模式：

■ 不带自动重传不带 ACK 的通信模式（后简称为普通模式），发射端可以使用命令有 W_TX_PAYLOAD，REUSE_TX_PL 等；

■ 带自动重传带 ACK 的通信模式（后简称为增强模式），发射端可以使用命令有 W_TX_PAYLOAD，W_TX_PAYLOAD_NOACK，REUSE_TX_PL 等。接收端可以使用命令有 W_ACK_PAYLOAD 等。

表 16.2 普通模式

通信名称	**普通模式**	
通信方	PTX	PRX
特点	单向发送	单向接收
发送数据的组帧方式	*发送数据组帧方式 I*	无
开启 REUSE_TX_PL 命令	重复发送前一包数据	无

表 16.3 增强模式

通信名称	**增强模式**	
通信方	PTX	PRX
特点	发送数据后，等待接收 ACK	接收数据后，回发送 ACK
发送数据的组帧方式	发送数据组帧方式 II	回发送 ACK 组帧方式 III
PTX 使用 REUSE_TX_PL 命令	重复发送前一包数据	每收到一包，回发送 ACK
PTX 使用 W_TX_PAYLOAD 命令 PRX 使用 W_ACK_PAYLOAD 命令	发送数据后，等待接收 ACK PAYLOAD	接收数据后，回发送 ACK PAYLOAD，组帧方式 II
PTX 使用 W_TX_PAYLOAD_NO	发送一次数据，不等 ACK，组帧方	接收数据，不回 ACK

ACK 命令	式 II	

16.1.3.1.1. 普通模式

普通模式下，发送端从 TX FIFO 寄存器中取出数据并且发送，发送完成后上报中断（中断需要清除），同时 TX FIFO 寄存器清除该数据（TX FIFO 需要清空）；接收端接收到有效的地址和数据时上报中断通知 MCU，随后 MCU 可将该数据从 RX FIFO 寄存器中读出（TX FIFO 和 RX FIFO 需要清空，中断需要清除）。

普通模式，（0x01）EN_AA 寄存器置 0x00，（0x06）SETUP_RETR 寄存器置 0x00，（0x3C）DYNPD 寄存器置 0x00，（0x3D）FEATURE 寄存器的低 3 bit 置 000。

16.1.3.1.2. 增强模式

增强模式下，把主动发起通信的一方称为 PTX（主发端），把接收数据并响应的一方称为 PRX（主收端）。PTX 发出数据后等待应答信号，PRX 接收到有效数据后回应答信号。PTX 规定时间内未收到应答信号，自动重新发送数据。自动重传和自动应答功能为收发机自带，无需 MCU 参与。

PTX 在发送数据后自动转到接收模式等待应答信号。如果没有在规定时间内收到正确的应答信号，PTX 将重发相同的数据包，直到收到应答信号，或传输次数超过 ARC 的值（SETUP_RETR 寄存器）产生 MAX_RT 中断。PTX 收到应答信号，即认为数据已经发送成功（PRX 收到有效数据），清除 TX FIFO 中的数据并产生 TX_DS 中断（TX FIFO 和 RX FIFO 需要清空，中断需要清除）。

PRX 每次收到一包有效数据都会回 ACK 应答信号，该数据如果为新数据（PID 值与上一包数据不同）保存到 RX FIFO，否则就丢弃。

增强模式，需要保证 PTX 的 TX 地址（TX_ADDR）、通道 0 的 RX 地址（如 RX_ADDR_P0），以及 PRX 的 RX 地址（如 RX_ADDR_P5）三者相同。例：在图 16.4 中，PTX5 对应 PRX 的数据通道 5，地址设置如下：

- PTX5：TX_ADDR=0xC2C3C4C5C1
- PTX5：RX_ADDR_P0=0xC2C3C4C5C1
- RX： RX_ADDR_P5=0xC2C3C4C5C1

增强模式有如下特征：

- 减少 MCU 的控制，简化软件操作；
- 抗干扰能力强，减少无线传输中因瞬间同频干扰造成的丢包，更易开发跳频算法；
- 重传过程中，减少 MCU 每次写入待发送数据的操作时间。

16.1.3.1.3. 增强发送模式

1、 CE 置 0，CONFIG 寄存器的 PRIM_RX 位先置 0。

2、 当发送数据时，发送地址（TX_ADDR）和有效数据（TX_PLD）按字节写入地址寄存器和 TX FIFO。

3、 CE 从 0 置 1，启动发射（CE 至少持续置 1 在 30us 以上，该操作生效）。

4、 自动应答模式下（SETUP_RETR 寄存器置非 0，ENAA_P0=1），PTX 发送完数据后立即自动将通道 0 切换到接收模式等待应答信号。如果在有效应答时间范围内收到 ACK 应答信号，则认为数据发送成功，状态寄存器的 TX_DS 位置 1 并自动清除 TX FIFO 中的数据。如果在设定时间范围内没有接收到应答信号，则自动重传数据。

5、 如果自动传输计数器（ARC_CNT）溢出（超过了设定值），则状态寄存器的 MAX_RT 位置 1，不清除 TX FIFO 中的数据。当 MAX_RT 或 TX_DS 为 1 时，IRQ 引脚产生低电平中断（需要使能相应中断）。中断可以通过写状态寄存器来复位。

6、 数据包丢失计数器（PLOS_CNT）在每次产生 MAX_RT 中断后加一。自动传输计数器 ARC_CNT 统计重发数据包的次数；数据包丢失计数器 PLOS_CNT 统计在达到最大允许传输次数时仍没有发送成功的数据包个数。

7、 产生 MAX_RT 或 TX_DS 中断后，RF 进入 STB 模式。

16.1.3.1.4. 增强接收模式

1、 CE 置 0，CONFIG 寄存器的 PRIM_RX 位先置 1。准备接收数据的通道必须被使能（EN_RXADDR 寄存器），所有工作在增强型通信模式下的数据通道的自动应答功能是由 EN_AA 寄存器来使能的，有效数据宽度是由 RX_PW_PX 寄存器来设置的。

2、 接收模式由设置 CE 为 1 启动。

3、 预设的等待时间后，PRX 开始检测无线信号。

4、 接收到有效的数据包后，数据存储在 RX_FIFO 中，同时 RX_DR 位置 1，产生中断。状态寄存器中 RX_P_NO 位显示数据是由哪个通道接收到的。

5、 自动发送 ACK 应答信号。

6、 如果 CE 保持为 1，继续进入接收模式；如果 CE 置为 0，则 RF 进入 STB 模式；

7、 MCU 将数据读出。

16.1.3.1.5. 增强模式下的数据包识别

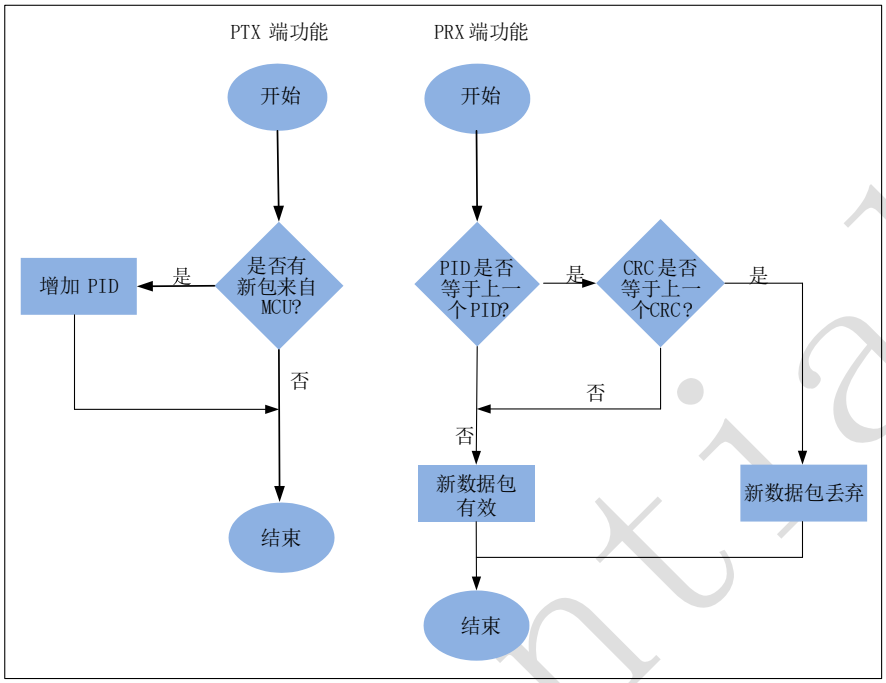


图 16.2 PID 生成和检测

每一包数据都包括两位的 PID（数据包标志位），用来帮助接收端识别该数据是新数据包还是重发的数据包，防止多次存入相同的数据包，PID 的生成和检测如上图所示。发送端从 MCU 取得一包新数据后 PID 值加一。

16.1.3.1.6. 增强模式下的 PTX 和 PRX 时序图

如图 16.3 所示的是一次 PTX 和 PRX 通信的芯片内部时序图，使得通信成功必须满足以下两个条件：

■ 条件 1、PTX 发射的锁相环稳定+功放使能+功放 RAMP UP 的三段时间之和，大于 PRX 接收的锁相环稳定时间，这样可以保证 PTX 发射数据的时间段落在 PRX 接收数据的时间段内，即：

PTX.T1 + PTX.T2 + PTX.T3 + PTX.T4 + PTX.T5 > PRX.T1 + PRX.T2

■ 条件 2、PRX 发送 ACK 的锁相环稳定+功放使能+功放 RAMP UP 的三段时间之和，大于 PTX 接收的锁相环稳定时间，保证 PRX 回复 ACK 的时间端落在 PTX 等待 ACK 的时间段内，即：

PRX.T5 + PRX.T6 + PRX.T7 + PRX.T8 + PRX.T9 > PTX.T12 + PTX.T13

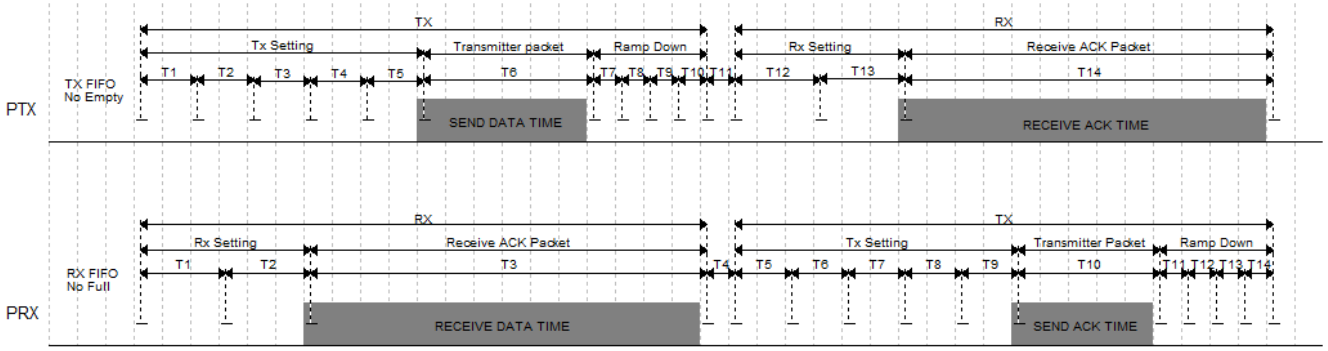


图 16.3 增强模式的时序图

图 16.3 中的各段时间描述见下表：

PTX 时间	PRX 时间	描述	决定因素	最小值	最大值	默认值	备注
T1	T5	LDO 使能到 TX 通道使能	CE_JUST_TIME + LDO_WAIT_TIME	0 0	12us 56us	4us 32us	两个相加
T2	T6	TX 通道使能到 EN_TX_PA2	PA_DLY_TIME_UP[4:0]	0	256us	64us	
T3	T7	EN_TX_PA2 到 EN_RAMP	RAMP_DLY_TIME_UP[3:0]	0	128us	32us	
T4	T8	EN_RAMP 到 TX_PA2ST_RAMP=PA2ST_RAMP_MAX	PA2ST_RAMP_MAX[2:0] * RAMP_STEP_UP[2:0]	0 0.5us	7 8us	7 2us	两个相乘
T5	T9	TX_PA2ST_RAMP=MAX 到开始发射	（EX_PA_TIME 或者 TX_SETUP_TIME）-（T2+T3+T4）+ TRX_TIME	0 0	1008us-（T2+T3+T4） 120us	320us-（T2+T3+T4） 0us	以时间长的为准
T6	T10	数据发射的时间	由 payload 的长度决定				
T7	T11	数据发射结束到 TX_PA2ST_RAMP 开始 ramp down	WAIT_CONCLUDE_TIME[1:0] + WAIT_RAMP_DN_TIME[3:0]	1us 0	4us 7.5us	4us 2us	两个相加
T8	T12	TX_PA2ST_RAMP=MAX 到 EN_RAMP 拉低	PA2ST_RAMP_MAX[2:0] * RAMP_STEP_DN[2:0]	0 0.5us	7 8us	7 2us	两个相乘
T9	T13	EN_RAMP 拉低到 EN_TX_PA2 拉低	RAMP_DLY_TIME_DN[3:0]	0	128us	32us	
T10	T14	EN_TX_PA2 拉低到 TX 通道关闭	PA_DLY_TIME_DN[4:0]	0	256us	64us	
T11	T4	TX/RX 切换时间	固定值	8us	8us	8us	
T12	T1	LDO 使能到 PLL 使能	LDO_WAIT_TIME	0	56us	32us	
T13	T2	PLL 使能到 RX 通道使能	RX_SETUP_TIME	0	496us	320us	
T14	T3	接收窗口	TX 数据包	0	+∞	-	

16.1.3.1.7. 增强模式下的接收端一对多通信

PAN262x作为发射端，对于一对多通信，可以采用不同的地址与多个接收端进行通信。

PAN262x作为接收端，可以接收6路不同地址、相同频率的发送端数据。每个数据通道拥有自己的地址。

使能哪些数据通道是通过寄存器EN_RXADDR来设置的。每个数据通道的地址是通过寄存器RX_ADDR_PX来配置的。通常情况下不允许不同的数据通道设置完全相同的地址。如下表表给出了一例多接收通道地址配置的示例。

表 16.4 多通道地址设置

	Byte 4	Byte 3	Byte 2	Byte 1	Byte 0
Data pipe 0(RX_ADDR_P0)	0xF1	0xD2	0xE6	0xA2	0x33
Data pipe 1(RX_ADDR_P1)	0xD3	0xD3	0xD3	0xD3	0xD3
Data pipe 2(RX_ADDR_P2)	0xD3	0xD3	0xD3	0xD3	0xD4
Data pipe 3(RX_ADDR_P3)	0xD3	0xD3	0xD3	0xD3	0xD5
Data pipe 4(RX_ADDR_P4)	0xD3	0xD3	0xD3	0xD3	0xD6
Data pipe 5(RX_ADDR_P5)	0xD3	0xD3	0xD3	0xD3	0xD7

从上表可以看出数据通道0的5byte总共40位的地址都是可配的；数据通道1~5的地址配置为32位共用地址（与数据通道1共用）+8位各自的地址（最低字节）。

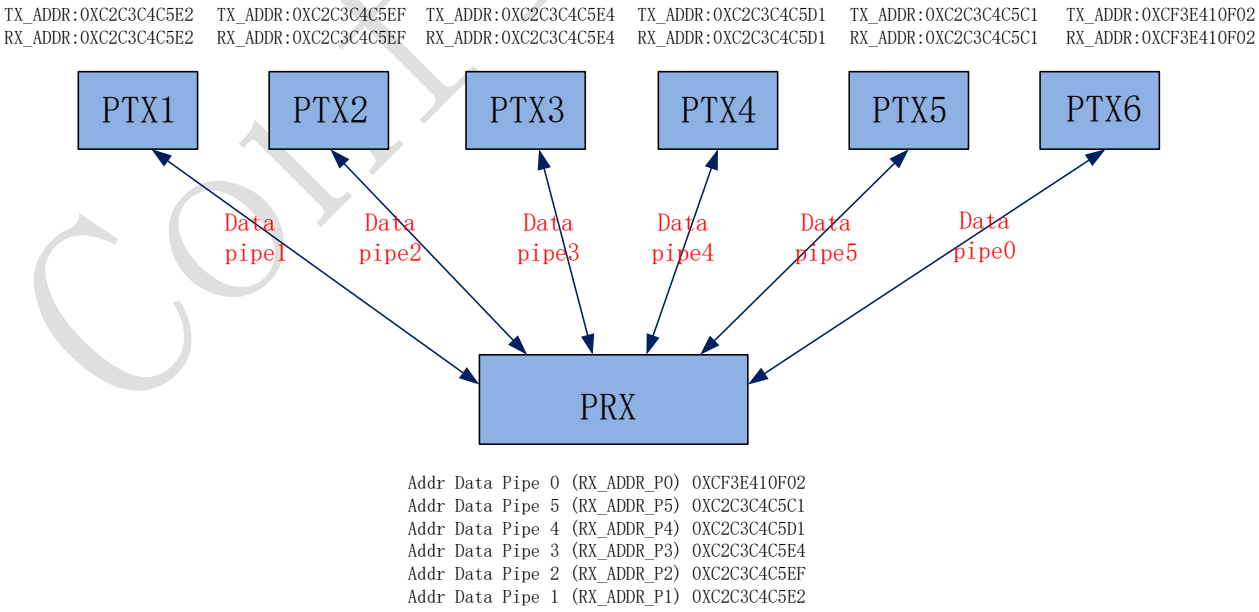


图 16.4 星状网络下的数据管道寻址示例

PAN262x 在接收模式下可以与最多 6 路不同通道通信，如上图所示。每一个数据通道使用不同的地址，共用相同的频道。所有的发射端和接收端设置为增强模式。

PRX 在接收到有效数据后记录 PTX 的 TX 地址，并以此地址为目标地址发送应答信号。PTX 数据通道 0 被用做接收应答信号时，数据通道 0 的 RX 地址要与 TX 地址相等以确保接收到正确的应答信号。上图给出了 PTX 和 PRX 地址如何配置的例子。

16.1.3.1.8. DATA FIFO

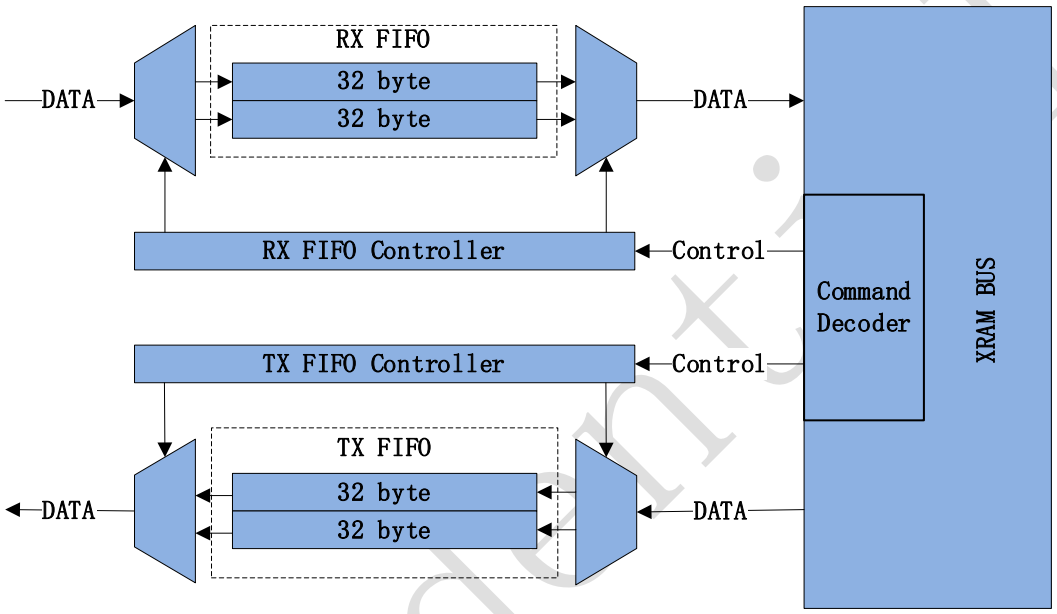


图 16.5 FIFO 框图

PAN262x 包含发 TX_FIFO，RX_FIFO。通过 XRAM BUS 可以读写 FIFO。在发送模式下通过 W_TX_PAYLOAD 和 W_TX_PAYLOAD_NO_ACK 指令来写 TX_FIFO。如果产生 MAX_RT 中断，在 TX_FIFO 中的数据不会被清除。在接收模式下通过 R_RX_PAYLOAD 指令读取 RX_FIFO 中的 payload，R_RX_PL_WID 指令读取 payload 的长度。FIFO_STATUS 寄存器指示 FIFO 的状态。

16.1.3.2 RF 频道设置

RF 频道有两种设置方法，一种方法是通过寄存器 RF_CH 来设置，该方法由下边的公式计算 F_{VCO}：

$$F_{VCO} = 2336 + RF\_CH + CH\_OFFSET[MHz]$$

另一种方法是通过手动修改分频值来设置，相关寄存器为fsynsdnint_sel（分频值整数部分手动改写使能），frequency_offset_ovrdwd_sel（分频值小数部分手动改写使能），fsynsdnint_ovrd[5:0] （分频值整数部分），frequency_offset_ovrdwd[22:0] （分频值小数部分），该设置方法由下边的公式来计算中心频率：

$$F_{VCO} = (fsynsdnint\_ovrd[5:0] + 32 + frequency\_offset\_ovrdwd[22:0]/2^{23}) * 32M\ [MHz]$$

寄存器	配置	模式
fsynsdnint_sel、frequency_offset_ovrdwd_sel	0、0	通过RF_CH/CH_OFFSET设置频点
fsynsdnint_sel、frequency_offset_ovrdwd_sel	0、1	不支持
fsynsdnint_sel、frequency_offset_ovrdwd_sel	1、0	不支持
fsynsdnint_sel、frequency_offset_ovrdwd_sel	1、1	通过配置分频值设置频点

表 16.5 PAN262x RF 频点配置模式

16.1.3.3 PA 控制

PAN262x 的 PA 是两级串联结构，分别为：PRE_AMP BUF 级预放大、POWER_AMP 功率级放大器。其中 PA 输入由 VCO 直接驱动 PA1 第一级，第一级有 3 比特电流控制位，第二级分 6 比特电流档位控制和 7 比特功率档位调节。PA1 和 PA2 由独立开关控制其打开。

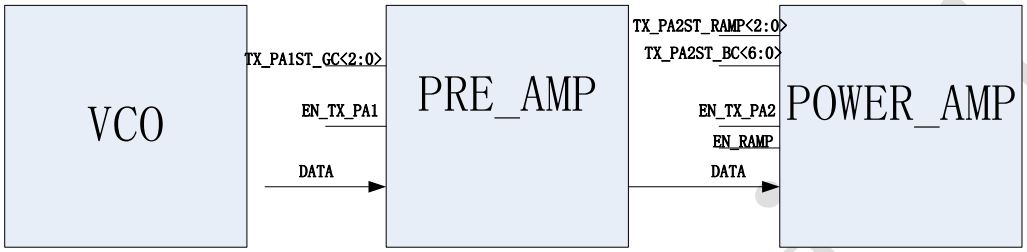


图 16.6 PAN262x PA 结构图

PA RAMP 开启顺序：

PLL 和 TX 通道除了 PA 同时打开，等待 PA_DLY_TIME_UP（注：4 到 128uS 可配置，默认 64uS，调节间隔 4uS）时间后，打开 PA2，然后等待 RAMP_DLY_TIME_UP（注：4 到 64uS 可配置，默认 32uS，调节间隔 4uS），等待 PLL 重新锁定，接着打开 EN_RAMP，开始 RAMP UP 过程，RAMP 由 3BIT 寄存器控制，模拟内部通过 3-8 译码进行转换，TX_PA2ST_RAMP<2:0>依次从 0 增加到 PA2ST_RAMP_MAX，步长 RAMP_STEP_UP（注：0.5uS，1uS，2uS，4uS，8uS 可配置，默认 2uS），最后发送数据，RAMP DOWN 过程基本对称，只是 step 和 DLY 时间需要可配置，要求和 RAMP UP 一样。

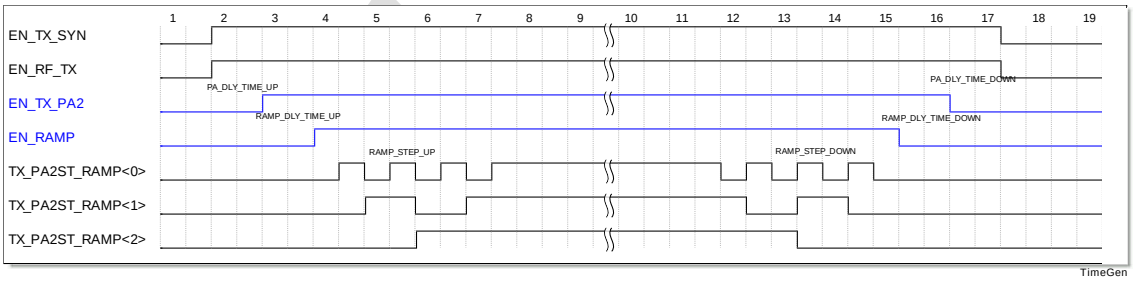


图 16.7 PAN262x PA RAMP 时序图

注：PA_DLY_TIME_UP 为变量，1 到 128uS 可配置，默认 64uS，调节间隔由寄存器 up1_step_mode 配置

PA_DLY_TIME_DOWN 为变量，1 到 128uS 可配置，默认 64uS，调节间隔由寄存器 dn1_step_mode 配置

RAMP_DLY_TIME_UP 为变量，1 到 64uS 可配置，默认 32uS，调节间隔由寄存器 up2_step_mode 配置

RAMP_DLY_TIME_DOWN 为变量，1 到 64uS 可配置，默认 32uS，调节间隔由寄存器 dn2_step_mode 配置

RAMP_STEP_UP 为变量，0.5uS，1uS，2uS，4uS，8uS 可配置，默认 2uS

RAMP_STEP_DOWN 为变量，0.5uS，1uS，2uS，4uS，8uS 可配置，默认 2uS

接口信号	控制方式	说明	位宽	默认值
RX_LNA_CTM<2:0>	寄存器	PA1ST peak point 调节	3	100
TX_PA2ST_CTM<2:0>	寄存器	PA2ST peak point 调节，保留位	3	111
EN_TX_PA1	寄存器	Enable 控制，PA_BUF 打开	1	TX 模式为 1，RX 模式为 0
EN_TX_PA2	RAMP 时序/寄存器	Enable 控制，PA2ST 开	1	TX 模式为 1，RX 模式为 0
TX_PA1ST_GC<2:0>	寄存器	PA1ST 的偏置电流控制	3	100
TX_PA2ST_BC<6:0>	寄存器	PA2ST 的偏置电流控制目标值	6	0111111
TX_PA2ST_RAMP<2:0>	RAMP 时序/寄存器	PA2ST 的功率档位控制目标值	3	111
EN_TXPA_LP	寄存器	PA 低功率模式使能，仅 B 版本生效	1	0
EN_RAMP	RAMP 时序/寄存器	RAMP 开启使能	1	1
TX_PA_RAMP_MODE	寄存器	Ramp 模式选择，1 数模结合，0 纯模拟	1	1
TX_RAMP_RC_TRIM<4:0>	寄存器	RAMP 时间常数调节	5	00011

表 16.6　PAN262x PA 数模接口

另外 PAN262x 可外挂 PA，外置 PA 的控制信号从 P37 引出，注意 IO MUX 设置。

16.1.3.4 校正

16.1.3.4.1.　接收滤波器校正

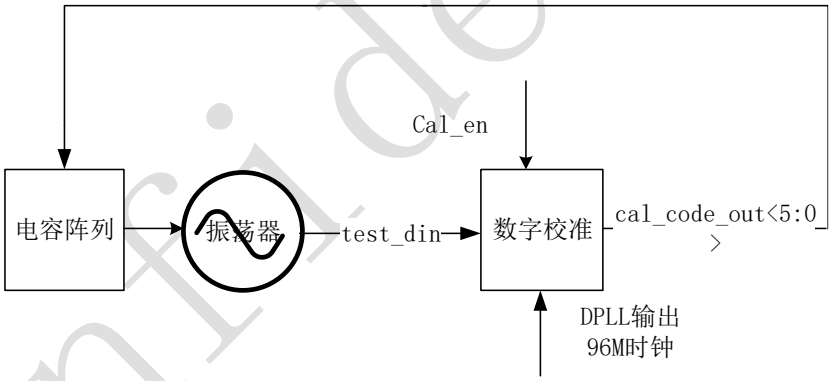


图 16.8　PAN262x 接收滤波器校正系统框图

工作原理：cal_en=1 开始校准，模拟振荡器输出 test_din（LIMIT_I）给到数字校准模块。数字校准用 96M 时钟计算 test_din 频率，数字收到 test_din 后数上升沿,并计算 test_din 频率（每次校准反馈后从 test_din 第 2~8 个上升沿后开始计数，此功能可以通过配置来实现 delay 时间）。为增加分辨率，本次设计采用计数两个被测频率周期，当 test_din 频率高于基准频率（2MHz）的计数后增加电容阵列的值，当 test_din 频率低于基准频率（2MHz）的计数后降低电容阵列的值，电容阵列的校准值反馈到模拟调整振荡器频率。

本设计采用二分法校准法，大致原理为：首先设置 cal_code_out 为 6'b100000 的中位数，然后开始计数两个被测周期内的 counter 值，如果测试 counter 值大于标准配置的值，如计数 counter 为 100，标准配置值为 96，则校准时钟太慢需要减少电容阵列来增加频率，故 cal_code_out 调节为 6'b010000，否则保存最高位的 1，cal_code_out 调节为 6'b110000，开始校准次高位，以此类推总共校准六次。

IO	位宽	端口名称	说明

input	1	clk	校准基准 96M 时钟
input	1	nrst	复位，低有效
input	1	test_din	校准时钟输入，本模块为 2M 左右的中频输入 LIMIT_I
input	8	ref_count	两个校准时钟周期内的基准 count 值，默认为 96M/2M *2，寄存器可配
input	3	wait_test_cnt	大周期内循环校准一次后等待模拟反馈的时间（一次大周期有六次循环校正），默认等待一个校准时钟周期，最终等待时间为配置值加 1，默认 1，寄存器可配
input	5	if_cal_wait	中频滤波器校正之前的等待周期，等待该周期个 LIMIT 信号之后才开始校正，留给模拟电路一些稳定时间
input	1	cal_en	状态机校准模块使能，一个大于 96M 的脉冲，该信号由主状态机给出，第一次从 RX_SET 进入 RX 之前会先进入 IF_CAL 状态，该状态会触发一次 cal_en
input	1	cal_busy	状态机校准模式下的 ICG EN 信号，由状态机控制
input	1	cal_en_m	软件触发模式触发信号，由寄存器 IF_CAL_EN_M 控制，该 bit 自动清零
input	1	cal_en_m_sel	软件触发模式选择，有寄存器 IF_CAL_EN_M_SEL 控制
input	1	cal_code_cfg_en	手动配置使能，使能后 cal_code_out 输出为 cal_code_cfg 的值
input	6	cal_code_cfg	手动配置 code 值
output	8	scounter	校准结束输出的校准时钟周期的 count 数，最终的校准频率 IF=2M*scounter/96，寄存器可读
output	6	cal_code_out	校准 code 输出，控制电容阵列数模接口 RCCAL_IN，寄存器 RCCAL_IN 可读
output	1	cal_done	校准结束标志，cal_en 置一后会拉低

表 16.7　PAN262x 接收滤波器校正模块端口信号描述

16.1.3.4.2.　VCO 校正

校正原因：由于 VCO 的频率线性范围有限，VCO 需要通过调整 VCO code 来实现覆盖整个频段。这里 VCO code 是指 VCO 内部的电容阵列的配置，VCO 通过调整 LC 电路中电容值来保证 VCO 振荡在不同的频率下。一般而言，VCO 的 LC 电路中会存在一个固定的 C，这样 VCO 会振荡在一个基本的频率上，而后通过调整额外的电容阵列，实现不同频率的振荡。

本设计采用二分法校正，校正 1~4 次后，会产生一个最合适的 vco_code。

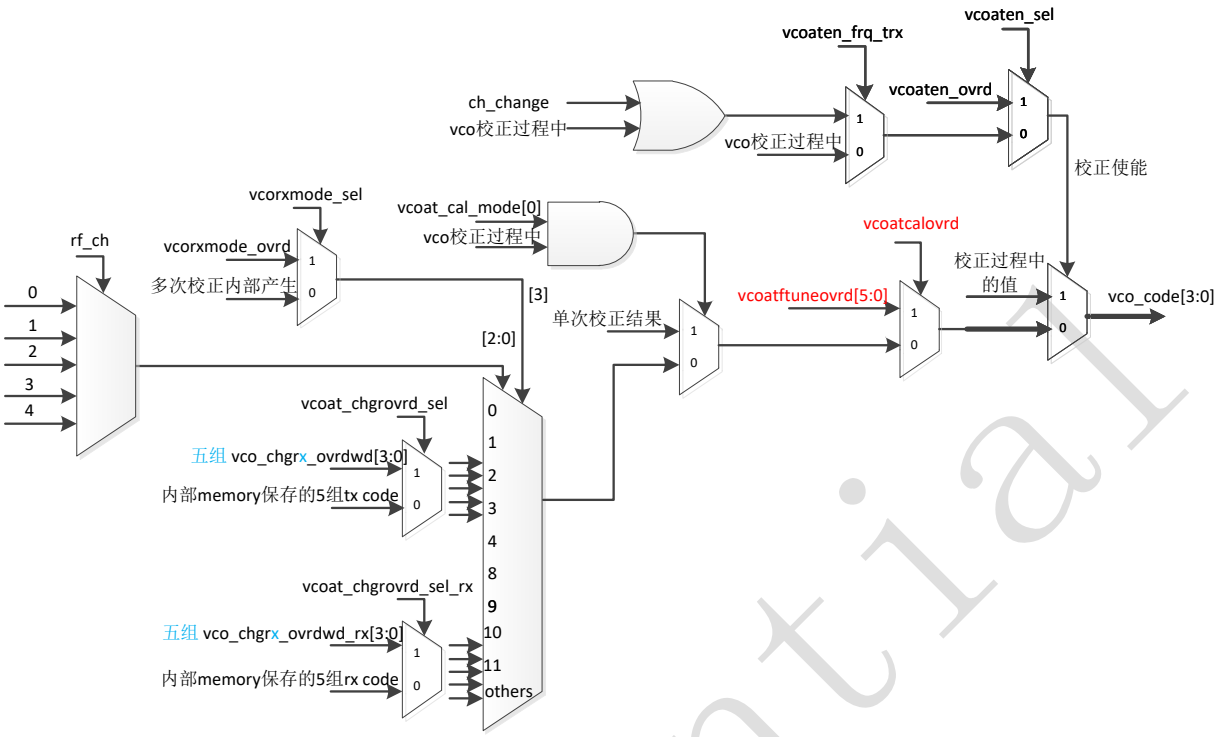


图 16.9 VCO CODE 产生原理图

vco 校正模块输入接口：

fsynvcoatstartmchcal：开始校正使能，电平使能，为 1 开始校正，校正结束拉低；

fsynvcoat_cal_mode [2:0]：其中[1:0]为 vco 校正模式选择。00 : curve based mode，多次校正，10 个频道；01 :channel based 比较 4 次，单次校正；11 :channel based 比较 3 次，单次校正。[2]用来配置[1:0]=11 时 code 变化方式。

cntref_mem_din[11:0]：多次校正模式计数参考值 ram 的写入值，对应多次校正的目标频率；

cntref_mem_wre：多次校正模式计数参考值 ram 的写使能信号；

cntref_mem_addr[3:0]：多次校正模式计数参考值 ram 的写地址，可配 0~9；

rf_ch [5:0]：频道选择；

ch_offset[7:0]：起始频道选择，tx 起始频道为 2336+ch_offset，rx 起始频道为 2338+ch_offset；

fsynvcoatmxcnt [8:0]：vco 校正参考时钟计数个数选择，决定了校正时长，计数时长计算公式为：62.5ns*fsynvcoatmxcnt；

fsyncaldivcalclk [1:0]：vco 输出几分频后送给数字选择信号，配置 00 为 8 分频（约 300M），配置 01 为 16 分频。配置 10 为 32 分频；

mc_vcorx_mode_sel： vco 校正是否手动改写为 rx 模式选择，用于单次 vco 校正，默认为 0，rx 模式下配 1；

mc_vcorx_mode_ovrd：vco 校正手动改为 rx 模式改写值，用于单次 vco 校正，默认为 0，rx 模式下配 1；

error_limit[3:0]：校正过程中，如果计数值减去参考值大于 error_limit，则不用等计数完规定窗口直接跳到下一个 code，可加快校正速度，仅在校正结果小于 32 时生效；

vcocal_freq_cover_tx[3:0]：tx vco 多次校正时每个 code 可以覆盖的频段选择。

vcocal_freq_cover_tx[3:0]：rx vco 多次校正时每个 code 可以覆盖的频段选择。

vco_dly_sel[1:0]：VCO 电容配置充电稳定时间，配 0~3 分别对应 3/6/9/12us；

rf_vcocalen：pll 校正相关时钟开关，校正开始前配 1，校正结束配 0；

fsynvcoat_chgrovrd_sel_rx：保存的 5 组 rx vco_code 是否手动改写选择，配 1 手动改写；

chgr0_ovrdwd_rx[3:0]：保存的第一组 rx vco code 手写值；

chgr1ovrdwd_rx[3:0]：保存的第二 vco code 手写值；

chgr2ovrdwd_rx[3:0]：保存的第三 rx vco code 手写值；

chgr3ovrdwd_rx[3:0]：保存的第四 rx vco code 手写值；

chgr4vrdwd_rx[3:0]：保存的第五 rx vco code 手写值；

fsynvcoat_chgrovrd_sel：保存的 5 组 tx vco_code 是否手动改写选择，配 1 手动改写；

chgr0_ovrdwd_tx[3:0]：保存的第一组 tx vco code 手写值；

chgr1ovrdwd_tx[3:0]：保存的第二组 tx vco code 手写值；

chgr2ovrdwd_tx[3:0]：保存的第三组 tx vco code 手写值；

chgr3ovrdwd_tx[3:0]：保存的第四组 tx vco code 手写值；

chgr4ovrdwd_tx[3:0]：保存的第五组 tx vco code 手写值；

vco_ready_sel[1:0]：等待模拟电路准备的时间长度配置

输出接口：

vcoat_mchcaldone：vco 校正结束标志；

PLL_VCO_CHSEL[3:0]：vco 校正结果，只读寄存器；

rf_fsynvcoat_grp0ftune[3:0]：多次校正的第一组 TX_code，只读寄存器；

rf_fsynvcoat_grp4ftune[3:0]：多次校正的第五组 TX_code，只读寄存器；

rf_fsynvcoat_grp0ftune_rx[3:0]：多次校正的第一组 RX_code，只读寄存器；

rf_fsynvcoat_grp4ftune_rx[3:0]：多次校正的第五组 RX_code，只读寄存器；

vco_ready：只读寄存器，标志着模拟电路准备好，可以开始校正。

校正流程：

Vco 校正分多次校正和单次校正，单次校正又分 TX 校正和 RX 校正。触发多次校正时会先做 5 次 tx 校正，然后做 5 次 rx 校正，使用时会自动从 10 个保存的 code 里面选一个最合适的。

1.手动配置单次 code

把寄存器 fsynvcoatcalovrd 配 1，然后配置寄存器 fsynvcoatftuneovrd[3:0]。

2.手动配置多次校正模式的 code

配置寄存器 fsynvcoat_chgrovrd_sel 和 fsynvcoat_chgrovrd_sel_rx 为 1，然后配置 10 组手写值，其中 5 组 TX（对应寄存器为 chgr0_ovrdwd_tx[3:0]、chgr1_ovrdwd_ tx [3:0]、chgr2_ovrdwd_ tx [3:0]、chgr3_ovrdwd_ tx [3:0]、chgr4_ovrdwd_ tx [3:0]），5 组 RX（对应寄存器为 chgr0_ovrdwd_rx[3:0]、chgr1_ovrdwd_rx[3:0]、chgr2_ovrdwd_rx[3:0]、chgr3_ovrdwd_rx[3:0]、chgr4_ovrdwd_rx[3:0]）。

3.多次校正：

1）RF 初始化配置；

2）寄存器 vcocalen=1 打开 PLL，此时小数字送给模拟的 caldiven 拉高，lfdacen 拉高，EN_TX_SYN 拉高。（注意：caldiven 为 VCO 频率 2.4G 8 分频到 300M 使能，lfdacen 为 PLL 开环使能，EN_TX_SYN 控制 Tx VCO 使能）；

3）等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

读到寄存器 vco_ready 为高，配置 vcoatstartmchcal=1，开始校正；

读取寄存器 vcoat_mchcaldone=1，则认为多次校正结束。数字内部会保存 5 组 tx_code 和 5 组 rx_code。

校正结束后，把寄存器 vcoatstartmchcal 和 vcocalen 拉低；

改变寄存器 RF_CH 的值，则硬件自动从保存的 10 个 code 中选择合适的值送给模拟电路使用；

8）只要弱 LDO 不关闭以及数字复位没有被拉低，这 10 个 code 会一直保存。

4.单次 TX 校正

1）RF 初始化配置；

2）vcocalen=1，打开 PLL，此时小数字送给模拟的 caldiven 拉高，lfdacen 拉高，EN_TX_SYN 拉高。

3）配置 vcoat_cal_mode 为 01 或 11，选择单次校正模式；

4）配置寄存器 fsynvcorxmode_sel=0，fsynvcorxmode_ovrd=0；

5）通过寄存器 RF_CH 或者手动模式，配置需要校正的频点。

6）等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

7）读到寄存器 vco_ready 为高，配置 vcoatstartmchcal=1，开始校正；

8）读取寄存器 vcoat_mchcaldone=1，则认为校正结束。此时 vco 校正结果会保存在只读寄存器 PLL_VCO_CHSEL_RO 里边，只要不再进行 vco 校正、弱 LDO 不关闭、数字复位不拉低，这个 code 会一直保存。

5. 单次 RX 校正

1）RF 初始化配置；

2）vcocalen=1，打开 PLL，此时小数字送给模拟的 caldiven 拉高，lfdacen 拉高，EN_RX_SYN 拉高。

3）配置 vcoat_cal_mode 为 01 或 11，选择单次校正模式；

4）配置寄存器 fsynvcorxmode_sel=1，fsynvcorxmode_ovrd=1；

5）通过 RF_CH 或者手动模式，配置需要校正的频点；

6）等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

7）读到 vco_ready 为高，配置 vcoatstartmchcal=1，高电平开始校正；

8）读取寄存器 vcoat_mchcaldone=1，则认为校正结束。此时 vco 校正结果会保存在只读寄存器 PLL_VCO_CHSEL_RO 里边，只要不再进行 vco 校正、弱 LDO 不关闭、数字复位不拉低，这个 code 会一直保存。

16.1.3.4.3. 两点式校正

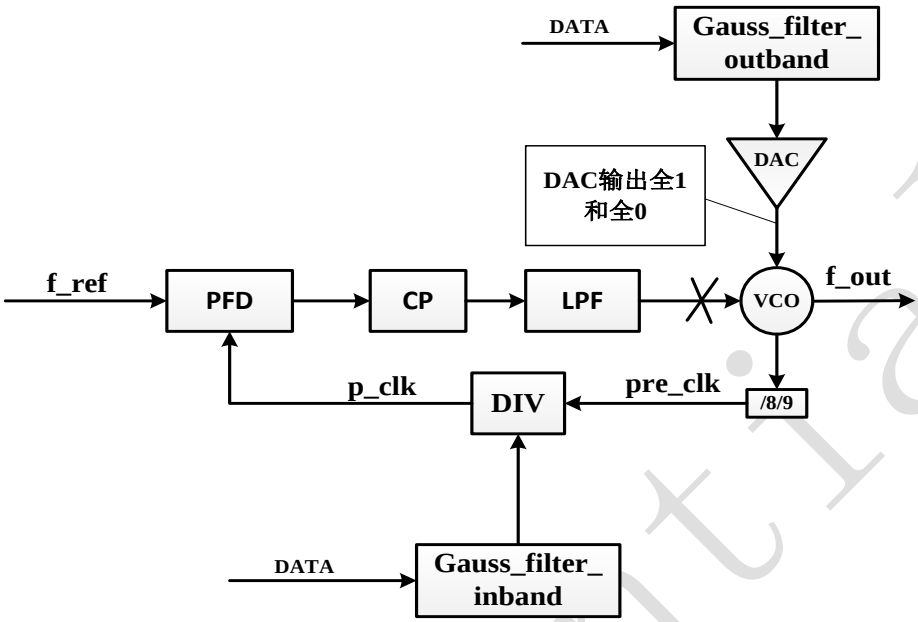


图 16.10 两点式校正原理图

每次上电时做一次两点式 calibration，以使得带内带外增益保持一致，校正时间长度可配。两点式校验在开环的情况下进行。

两点式校正时 2.4G 的 fvco 与 dac_out 的计算关系为

fvco= f0 + Vctrl*Kvco1 + vco_code*M1 + dac_out *Kvco

其中，f0 为 vco 固定频率。Vctrl 为 LPF 的输出电压，Kvco1 为 Vctrl 对 vco 的影响系数。vco_code 为调整电容阵列的输入，M1 为 vco_code 的影响系数。dac_out 为 DAC 的输出，由带外高斯滤波器决定大小，Kvco 为 dac_out 的影响系数。

做两点式校正时，Vctrl 和 vco_code 都保持不变，因此频率变化只由 dac_out 变化引起。

使用晶振 16M 计数产生 T=4.096ms 的时钟窗口，在这个时钟窗口内对 fvco 8 分频产生的 pre_clk(约 300M)计数，第一个窗口内高斯输入全 1，vco 频率设为约 2.4G 的 fvco1;第二个窗口内高斯输入全 0，设为约 2.4G 的 fvco2。

两次计数差为

$$\text{actual\_diff} = \text{pre\_cnt1-pre\_cnt2}$$

$$= 2^{12}us * (f_{\text{pre\_clk1}} - f_{pre\_clk2})$$

$$=2^{9} * (f\text{vco1} - fvco2)$$

$$=2^{9} * (f_{up} - f_{dn})$$

如果校正结束，fup – fdn = 500K，actual_diff = 256。因此 actual_diff 每变化 1，对应 VCO 处变化 250k/256=0.9765625KHZ。

带内通过配置寄存器算出 deltaf2pcal[15:0]，参考计数差为

$$ref\_diff = \text{delta}f2pcal3 - deltaf2pcal4$$

由于 deltaf2pcal 是通过 fvco 除以 16M 时钟计算出来的，把 deltaf2pcal 与 fvco 关系式带入后得

$$\text{ref}\_diff = \frac{2^9 * 2^5}{2^{23}} \ (\text{delta}f2pcal3 - deltaf2pcal4) * 2^9$$

$$= 2^9 * 16 * 2 * \left(\frac{\text{delta}\_f3 - delta\_f4}{2^{23}}\right)$$

$$= 2^9 * \left(\frac{fvco3}{16M*2} - \frac{f\text{vco}4}{16\text{M}*2}\right) * 16\text{M}*2$$

$$= 2^9 * (f\text{vco}3 - fvco4)$$

可见，带内带外 deviation 相等时，ref_diff 应该等于 actual_diff。

校正开始前，先配置好带内的 ref_diff（配置 deltaf2pcal），通过比较 actual_diff 和 ref_diff，不断通过校正改变 actual_diff，最多比较 5 次后得到一个最接近 ref_diff 的 actual_diff。

配置接口：

en_two_point_cal：两点式校正自动 code 使能，若为 0，使用手动配置 code，若为 1，使用校正的 code；

tp_code_group_sel：手动模式下使用几组 code 选择，00 为 1 组，01 为 2 组，10 为 3 组，11 为 4 组；

two_point_manul_code_in0：spi 配置的第一组手动 code；

two_point_manul_code_in1：spi 配置的第二组手动 code；

two_point_manul_code_in2：spi 配置的第三组手动 code；

two_point_manul_code_in3：spi 配置的第四组手动 code；

two_point_spi_trig：触发信号，下降沿触发 two_point_cal；

two_point_auto_code_in：数字内部保存的上次校正的 code；

code_offset：两点式校正偏置（有符号），偏置后的 code 最终输出给带外 gauss_filter；

dac_gain_sel[1:0]：模拟 DAC 放大倍数选择；

window_int_sel[1:0]：方案 2 间隔时间选择；

tp_cal_mode_sel[1:0]：两点式校正模式选择；

tp_done_sel[2:0]：actual_diff 与 ref_diff 差多少时即可结束校正选择。

two_point_cal_clk_en：打开数字内部两点式校正需要的 clk。拉高后会把 caldiven 拉高，lfdacen 拉高，EN_TX_SYN 拉高。

caldivcalclk[1:0]：vco 分频器的分频值选择，同等校正精度下，其值越大，校正时间越长。

校正流程：

本设计分三种方案，方案 0 最基本，耗时最长。

方案 0

1.RF 初始化配置，配置寄存器 gauss_scale 和 df_sel；

2.令寄存器 en_two_point_cal=1，使用自动 code 配置频道，PLL_LPF_VSEL[2]配 0，把 vctrl 接地；

3.Two_point_clk_en=1，打开相关时钟；

4. 等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

5. 读到 vco_ready 为高，配置 Two_point_spi_trig 先为 1 再为 0，下降沿触发一次校正；

6.等待 1ms，读取寄存器 two_point_cal_done 为高，校正结束；

7.等待 5us，Two_point_clk_en 拉低。

如果使用刚校正的 code，en_two_point_cal 仍配置为 1。如果 en_two_point_cal 配置为 0，自动选择使用手动 code two_point_manul_code_in0～two_point_manul_code_in3。

方案 1

1.RF 初始化配置，配置 DAC_GAIN_SEL[1:0]，配置 gauss_scale 和 df_sel；

2.令寄存器 en_two_point_cal=1，使用自动 code 配置频道，PLL_LPF_VSEL[2]配 0，把 vctrl 接地；

3.Two_point_clk_en=1，打开相关时钟；

4.等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

5. 读到 vco_ready 为高，配置 Two_point_spi_trig 先为 1 再为 0，下降沿触发一次校正；

6. 等待 1ms，读取寄存器 two_point_cal_done 为高，校正结束；

7. 等待 5us，Two_point_clk_en 拉低；

8. 把 DAC_GAIN_SEL 配置为 00，把带内 deviation 重新配置为 62.5K/250K/500K。

如果使用刚校正的 code，en_two_point_cal 仍配置为 1。如果 en_two_point_cal 配置为 0，自动选择使用手动 code two_point_manul_code_in0~ two_point_manul_code_in3。

方案 2

1.RF 初始化配置，配置 gauss_scale 和 df_sel；

2.令寄存器 en_two_point_cal=1，使用自动 code 配置频道，PLL_LPF_VSEL[2]配 0，把 vctrl 接地；

3.Two_point_clk_en=1，打开相关时钟；

4. 等待模拟电路准备好，通过 vco_dly_sel 寄存器配置 VCO 校正电容充电的等待时间，通过配置寄存器 vco_ready_sel 来等待 vco_ready 从低到高的时间；

5. 读到 vco_ready 为高，配置 Two_point_spi_trig 先为 1 再为 0，下降沿触发一次校正；

6. 等待 1ms，读取寄存器 two_point_cal_done 为高，校正结束；

7. 等待 5us，Two_point_clk_en 拉低。

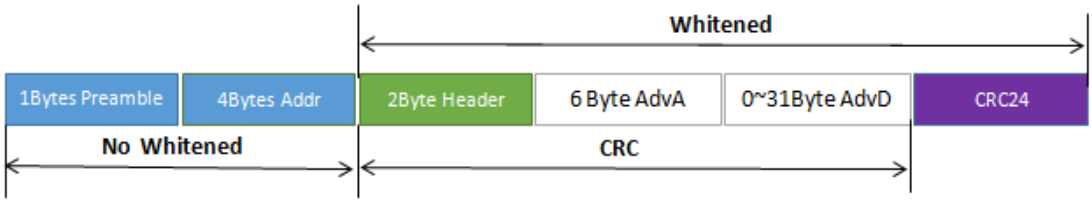
如果使用刚校正的 code，en_two_point_cal 仍配置为 1。如果 en_two_point_cal 配置为 0，自动选择使用手动 code two_point_manul_code_in0~ two_point_manul_code_in3。

16.1.3.5 数据包格式描述

PAN262x RF 收发机支持以下几种帧结构：

16.1.3.5.1. BLE 4.2 广播包

兼容 BLE 4.2 广播包格式，注意接收端会把 2 bytes 的 header 存到 FIFO 里边。



16.1.3.5.2. 297L 普通型数据包

普通模式的数据包格式如表16.8所示，组帧方式 I。

前导码（3字节）	地址（3~5字节）	数据（1~32/64字节）	CRC校验（1/2字节）

表16.8 普通模式的数据包形式

表 16.8 中地址和数据部分可以选择扰码方式，使能/关闭扰码配置位。

16.1.3.5.3. 297L 增强型数据包

增强模式的数据包格式如表 16.9 所示，组帧方式 II。

前导码（3字节）	地址（3~5字节）	标识（10bit）			数据（0~32/64字节）	CRC校验（1/2字节）
		数据长度标识（7bit）	PID标识（2bit）	NO_ACK标识（1bit）		

表16.9 增强模式的数据包形式

表 16.9 中地址、标识和数据部分可以选择扰码方式，使能/关闭扰码配置位。

16.1.3.5.4. 297L 增强型 ACK 包

增强模式的 ACK 包格式如表 16.10 所示，组帧方式 III。

前导码	地址	标识（10bit）	CRC校验

（3字节）	（3~5字节）	数据长度标识（7bit）	PID标识（2bit）	NO_ACK标识（1bit）	（1/2字节）

表16.10 增强模式的数据包形式

表 16.10 中地址和标识部分需要选择与 PTX 相同的使能/关闭扰码方式。

16.1.3.5.5. 24L01 普通型数据包

24L01普通模式的数据包格式如表16.11所示，组帧方式 I。

前导码（1字节）	地址（3~5字节）	数据（1~32字节）	CRC校验（1/2字节）

表16.11 24L01普通模式的数据包形式

表 16.11 中地址和数据部分可以选择 CRC8 或者 CRC16，无扰码配置。

16.1.3.5.6. 24L01 增强型数据包

24L01 增强模式的数据包格式如表 16.12 所示，组帧方式 II。

前导码（1字节）	地址（3~5字节）	标识（9bit）			数据（0~32字节）	CRC校验（1/2字节）
		数据长度标识（6bit）	PID标识（2bit）	NO_ACK标识（1bit）		

表16.12 增强模式的数据包形式

表 16.12 中地址、标识和数据部分可以选择 CRC8 或者 CRC16，无扰码配置。

16.1.3.5.7. 24L01 增强型 ACK 包

24L01 增强模式的 ACK 包格式如表 16.13 所示，组帧方式 III。

前导码（1字节）	地址（3~5字节）	标识（9bit）			CRC校验（1/2字节）
		数据长度标识（6bit）	PID标识（2bit）	NO_ACK标识（1bit）	

表16.13 增强模式的数据包形式

16.1.3.5.8. 数据包组件描述

前导码

297L 中的前导码长度为 3 个字节。 前导码的比特序列是固定的，它不能由寄存器配置。 前导码用于使接收器解调器与输入比特流同步。

地址

寄存器将地址宽度配置为 3,4 或 5 个字节。接收器对来自发送器的数据包进行解调，如果数据包的地址与接收器的地址相同，它会把后续有效负载保存在 RX FIFO 中，否则丢弃后续有效负载并恢复同步。

标识

分组控制字段由 10 比特组成，其包含 7 比特的数据长度标识字段，2 比特 PID（数据包标识）字段和 1 比特 NO_ACK 标识。仅存在于增强模式中。

数据长度标识

数据长度标识由 7 比特组成。有效载荷的长度可以是 0 到 64 个字节。

编码：0000000 = 0 字节（仅用于空 ACK 数据包），1000000 = 64 字节。

仅当启用动态有效载荷长度功能时才使用此字段。

NO_ACK 标识

NO_ACK 标识仅在 297L 增强模式时使用。使用此功能时，PTX 在发送数据包后直接进入待机模式。PRX 在收到数据包时不发送 ACK 数据包。

有效载荷

有效载荷可以是 0 到 64 字节，其中包含用户定义的信息。

在普通模式下，发射机和接收机具有相同的静态长度。不同于普通模式，增强模式具有动态有效载荷长度，静态有效载荷长度除外。

利用静态有效载荷长度，发射器和接收器之间的所有数据包都具有相同的长度。静态有效载荷长度由接收器侧的 RX_PW_Px 寄存器设置。在发送器上，有效载荷长度由时钟输入 TX_FIFO 的字节数设置，并且必须等于接收器侧 RX_PW_Px 寄存器中的值。

DPL 启用动态有效负载长度。这意味着发送器将具有可变有效载荷长度的数据包发送到接收器。MCU 可以使用 R_RX_PL_WID 命令而不是使用 RX_PW_Px 寄存器来读取接收到的有效负载的长度。为了使能 DPL，必须先使能 FEATURE 寄存器中的 EN_DPL 位。在 RX 模式下，必须设置 DYNPD 寄存器。如果要将 PTX 发送给一个使能 DPL 的 PRX，这个 PTX 必须设置 DYNPD 的 DPL_P0 位。

CRC

CRC 是数据包中的强制性错误检测机制。它在 CONFIG 寄存器中设置使能位和字节数。 如果启用了 CRC，接收器将检查接收到的数据包的 CRC。如果它们不相同，接收方将丢弃数据而不产生中断。如果 CRC 被禁用，当地址相同时，接收器将把数据保存在 FIFO 中。

16.1.3.6 中断引脚

作为外设，RF 的中断引脚 IRQ 为高电平触发，IRQ 引脚初始状态为低电平，当状态寄存器中 TX_DS、RX_DR 或 MAX_RT 为 1，以及相应的中断上报使能位为 0 时，IRQ 引脚的中断触发。MCU 给相应中断源写‘1’时，清除中断。IRQ 引脚的中断触发可以被屏蔽或者使能，通过设置中断上报使能位为 1，禁止 IRQ 引脚的中断触发。

16.1.3.7 BLE 白名单

在 BLE 接收模式下，支持两种白名单过滤方式，一种是基于 Header.Length 的过滤方式，另一种是基于 payload 的过滤方式，两种方式可以单独使用也可以结合在一起使用，都有单独的寄存器可以控制。

基于 Header.Length 的过滤方式由寄存器 LEN_MATCH_MODE 控制：

LEN_MATCH_MODE[1:0]	过滤方式
00	不过滤 length
01	只有收到包的 length 等于 rx_pw_p0 才继续收包
10	只有收到包的 length 大于 rx_pw_p0 才继续收包
11	只有收到包的 length 小于 rx_pw_p0 才继续收包

基于 payload 的过滤方式由寄存器 WL_MATCH_MODE[2:0]、WL_ADDR[47:0]、PLD_START_BYTES[5:0]共同决定：

WL_MATCH_MODE[2:0]	描述
000	不过滤 payload，全部上报
001	只需匹配上 WL_ADDR[47:40]即上报
010	只需匹配上 WL_ADDR[47:32]即上报
011	只需匹配上 WL_ADDR[47:24]即上报
100	只需匹配上 WL_ADDR[47:16]即上报
101	只需匹配上 WL_ADDR[47:8]即上报
110	只需匹配上 WL_ADDR[47:0]即上报
111	不过滤 payload，全部上报，同 000
WL_ADDR[47:0]	基于 payload 过滤机制的白名单设置，其值应该为 028.payload 当中的某一段
PLD_START_BYTES[5:0]	设置与 WL_ADDR[47:0]进行比较的 payload 的起始字节数，PAN262x 广播包 payload 范围 9~39，前 8 个 byte 为 Header(2)+AdvA(6) 配置为 1~2 ： 从 Header 开始过滤 配置为 3~8 ： 从 AdvA 开始过滤 配置为 9~39 ： 从 payload 开始过滤

16.2 RFTR 寄存器列表

Register	Offset（0x）	R/W	Description	Reset Value
CONFIG	00	R/W	工作模式配置寄存器	0x06
EN_AA	01	R/W	接收通道自动应答使能	0x00
AW_RXADDR	02	R/W	接收通道使能	0xC1
SETUP_RX	03	R/W	接收通道设置	0x20
SETUP_RX_1	04	R/W	接收通道设置	0xDC
SETUP_RX_2	05	R/W	接收通道设置	0x04
SETUP_RETR	06	R/W	发射端自动传输设置	0x03
RF_CH	07	R/W	通信频道设置	0x00
PLL_CTL0	08	R/W	PLL 控制寄存器 0	0x61
SETUP_RXTX	09	R/W	通信参数配置	0x00
SETUP_RXTX_1	0A	R/W	通信参数配置	0xFF
SETUP_RXTX_2	0B	R/W	通信参数配置	0x1B
STATUS	0C	RS/W	状态寄存器	0x0E
OBSERVE_TX	0D	RO	传输状态寄存器	0x00
PA_RAMP_CTL4	0E	R/W	PA RAMP 控制寄存器	0x07
PA_RAMP_CTL5	0F	R/W	PA RAMP 控制寄存器	0xAA
RX_ADDR_P0	10~14	R/W	PIPE 0 接收地址，最长 5 字节，具体长度由 AW 定义	0xE7E7E7E7E7
RX_ADDR_P1	15~19	R/W	PIPE 1 接收地址，最长 5 字节，具体长度由 AW 定义	0xC2C2C2C2C2
RX_ADDR_P3	1A	R/W	PIPE 3 接收地址低 8 位，高位等于 RX_ADDR_P1[39:8]	0xC4
RX_ADDR_P2	1B	R/W	PIPE 2 接收地址低 8 位，高位等于 RX_ADDR_P1[39:8]	0xC3
RX_ADDR_P5	1C	R/W	PIPE 5 接收地址低 8 位，高位等于 RX_ADDR_P1[39:8]	0xC6
RX_ADDR_P4	1D	R/W	PIPE 4 接收地址低 8 位，高位等于 RX_ADDR_P1[39:8]	0xC5
SETUP_RF	1E	R/W	RF 控制寄存器	0x7F
SETUP_RF_1	1F	R/W	RF 控制寄存器	0x3F
RXWIN_CTRL	20	R/W	RX 开窗时间控制寄存器	0x68
RXWIN_CTRL_1	21	R/W	RX 开窗时间控制寄存器	0x89
RXWIN_CTRL_2	22	R/W	RX 开窗时间控制寄存器	0x09
Reserved	23			
DAC_CTL1	24	R/W	DAC 控制寄存器	0x28
DAC_CTL2	25	R/W	DAC 控制寄存器	0x08
TX_ADDR	26~2A	R/W	发射端地址	0xE7E7E7E7E7
Register	Offset（0x）	R/W	Description	Reset Value

RX_PW_P0	2B	R/W	Data pipe 0 中的 RX payload 数据长度	0x00
RX_PW_P1	2C	R/W	Data pipe 1 中的 RX payload 数据长度	0x80
RX_PW_P2	2D	R/W	Data pipe 2 中的 RX payload 数据长度	0x80
RX_PW_P3	2E	R/W	Data pipe 3 中的 RX payload 数据长度	0x00
RX_PW_P4	2F	R/W	Data pipe 4 中的 RX payload 数据长度	0x00
RX_PW_P5	30	R/W	Data pipe 5 中的 RX payload 数据长度	0x00
FIFO_STATUS	31	RO	FIFO 状态寄存器	0x11
SETUP_TXRX	32	R/W	接收通道配置	0x24
SETUP_TXRX_1	33	R/W	接收通道配置	0xC4
SETUP_TXRX_2	34	R/W	接收通道配置	0x00
DEMOD_CAL	35	R/W	调制解调参数配置	0x0F
PMU	36	R/W	电源管理配置	0x33
PMU_1	37	R/W	电源管理配置	0x33
PMU_2	38	R/W	电源管理配置	0x43
DEM_CTL	39	R/W	解调器参数配置	0xC5
DEM_CTL_2	3A	R/W	解调器参数配置	0x6E
DEM_CTL_3	3B	R/W	解调器参数配置	0x09
DYNPD	3C	R/W	动态长度 payload 使能	0x00
FEATURE	3D	R/W	特征寄存器	0x80
PLL_CTL1	3E	R/W	PLL 控制寄存器 1	0x52
PLL_CTL2	3F	R/W	PLL 控制寄存器 2	0xA9
BB_CAL	40	R/W	基带参数配置	0xCA
BB_CAL_2	41	R/W	基带参数配置	0x94
BB_CAL_3	42	R/W	基带参数配置	0x54
BB_CAL_4	43	R/W	基带参数配置	0x87
BB_CAL_5	44	R/W	基带参数配置	0x04
WL_ADDR	45~4A	R/W	BLE RX 模式白名单设置	0x000000000000
WL_MODE	4B	R/W	白名单过滤模式选择	0x10
ACCESS_ADD	4C~4F	R/W	BLE 模式 Access Address 设置	0x8E89BED6
DEV_COM_P	50~51	R/W	整体定向频偏向上调节	0x0000
DEV_COM_N	52~53	R/W	整体定向频偏向下调节	0x0000
PA_RAMP_CTL0	54	R/W	PA RAMP 控制寄存器 0	0x4F
PA_RAMP_CTL1	55	R/W	PA RAMP 控制寄存器 1	0x4F
Register	Offset（0x）	R/W	Description	Reset Value
PA_RAMP_CTL2	56	R/W	PA RAMP 控制寄存器 2	0x77

PA_RAMP_CTL3	57	R/W	PA RAMP 控制寄存器 3	0x7F
PLL_CTL3	58	R/W	PLL 控制寄存器 3	0x28
PLL_CTL4	59	R/W	PLL 控制寄存器 4	0xE8
PLL_CTL5	5A	R/W	PLL 控制寄存器 5	0xF6
DIG2_REG0	5B	R/W	PLL 小数字控制寄存器 0	0xD2
DIG2_REG0_2	5C	R/W	PLL 小数字控制寄存器 0	0x05
DIG2_REG0_3	5D	R/W	PLL 小数字控制寄存器 0	0x9E
DIG2_REG0_4	5E	R/W	PLL 小数字控制寄存器 0	0x80
FIR_CAL1	5F	R/W	中频滤波器校正寄存器 1	0x60
RF_CE	60	R/W	RF 收发机片选使能	0x00
RF_CMD	61	R/W	RF 特殊 CMD 寄存器	0x00
RF_FIFO	62	R/W	读写 FIFO 入口寄存器	0x00
RF_PL_WIDTH	63	RO	RX payload 长度读取	0x00
DIG2_REG1	64	R/W	PLL 小数字寄存器 1	0x88
DIG2_REG1_2	65	R/W	PLL 小数字寄存器 1_2	0x88
DIG2_REG1_3	66	R/W	PLL 小数字寄存器 1_3	0x08
DIG2_REG2	67	R/W	PLL 小数字寄存器 2	0x88
DIG2_REG2_2	68	R/W	PLL 小数字寄存器 2_2	0x88
DIG2_REG2_3	69	R/W	PLL 小数字寄存器 2_3	0x08
DIG2_REGC_1	6A	R/W	PLL 小数字寄存器 C_1	0x10
DIG2_REGC_2	6B	R/W	PLL 小数字寄存器 C_2	0x10
DIG2_REGC_3	6C	R/W	PLL 小数字寄存器 C_3	0x10
DIG2_REGC_4	6D	R/W	PLL 小数字寄存器 C_4	0x10
DIG2_REG3	6E	R/W	PLL 小数字寄存器 3	0x40
DIG2_REG3_2	6F	R/W	PLL 小数字寄存器 3	0x79
DIG2_REG3_3	70	R/W	PLL 小数字寄存器 3	0x84
DIG2_REG3_4	71	R/W	PLL 小数字寄存器 3	0x00
DIG2_REG4	72	R/W	PLL 小数字寄存器 4	0x60
DIG2_REG4_2	73	R/W	PLL 小数字寄存器 4	0x60
DIG2_REG4_3	74	R/W	PLL 小数字寄存器 4	0x04
DIG2_REG4_4	75	R/W	PLL 小数字寄存器 4	0x6A
DIG2_REG4_5	76	R/W	PLL 小数字寄存器 4	0xF8
DIG2_REG5	77	R/W	PLL 小数字寄存器 5	0x00
DIG2_REG5_2	78	R/W	PLL 小数字寄存器 5	0x00
DIG2_REG5_3	79	R/W	PLL 小数字寄存器 5	0x00
DIG2_REG5_4	7A	R/W	PLL 小数字寄存器 5	0x00
DIG2_REG5_5	7B	R/W	PLL 小数字寄存器 5	0x00
DIG2_REG5_6	7C	R/W	PLL 小数字寄存器 5	0x78
Register	Offset（0x）	R/W	Description	Reset Value
DIG2_REG5_7	7D	R/W	PLL 小数字寄存器 5	0xD2
DIG2_REG5_8	7E	R/W	PLL 小数字寄存器 5	0x05

DIG2_REG6	7F	R/W	PLL 小数字寄存器 6	0x1B
DIG2_REG6_2	80	R/W	PLL 小数字寄存器 6	0xA0
DIG2_REG6_3	81	R/W	PLL 小数字寄存器 6	0xA0
DIG2_REG6_4	82	R/W	PLL 小数字寄存器 6	0x00
DIG2_REG7	83	R/W	PLL 小数字寄存器 7	0x00
DIG2_REG7_2	84	R/W	PLL 小数字寄存器 7	0x00
DIG2_REG7_3	85	R/W	PLL 小数字寄存器 7	0x40
DIG2_REG7_4	86	R/W	PLL 小数字寄存器 7	0x80
DIG2_REG8	87	R/W	PLL 小数字寄存器 8	0x1F
DIG2_REG8_2	88	R/W	PLL 小数字寄存器 8	0x5F
DIG2_REG8_3	89	R/W	PLL 小数字寄存器 8	0x00
Reserved	8A~9C			
DIG2_REG9	9D	R/W	PLL 小数字寄存器 9	0x00
DIG2_RO0	9E	RO	PLL 小数字只读寄存器 0	0x88
DIG2_RO0_2	9F	RO	PLL 小数字只读寄存器 0	0x88
DIG2_RO0_3	A0	RO	PLL 小数字只读寄存器 0	0x08
DIG2_RO1	A1	RO	PLL 小数字只读寄存器 1	0x00
DIG2_RO1_2	A2	RO	PLL 小数字只读寄存器 1	0x00
Reserved	A3~A4			
DIG2_RO3	A5	RO	PLL 小数字只读寄存器 3	0x30
DIG2_RO4	A6	RO	PLL 小数字只读寄存器 4	0x00
Reserved	A7			
DIG2_RO6	A8	RO	PLL 小数字只读寄存器 6	0x08
FIR_CAL_RO	A9	RO	中频滤波器校正结果	0x00
FIR_CAL_RO_1	AA	RO	中频滤波器校正结果	0x20
FIR_CAL_M	AB	R/W	中频滤波器校正手动控制寄存器	0x00
DIG2_RO7	AC	RO	PLL 小数字只读寄存器 7	0x88
DIG2_RO7_2	AD	RO	PLL 小数字只读寄存器 7	0x88
DIG2_RO7_3	AE	RO	PLL 小数字只读寄存器 7	0x08
Reserved	AF~B0			
BLE_WHIT_TEST	B1	R/W	BLE 加扰测试	0x33
BLE_RX_CTRL	B2	R/W	BLE RX 模式过滤配置	0x09

注意：对于本章表格，RO：表示只读；WO：表示只写；R/W：表示可读可写，读出的值为最近一次写入的值；RS/W：表示写入的值用来某些控制，读出的值反应某些状态信息，即读出的值与写入的值不一样。

16.3 RFTR 寄存器详细描述

地址（hex）	寄存器	BIT	默认值	推荐值	读写	说明
00	CONFIG	7:0	0x06	0x06	R/W	工作模式配置寄存器
	WORK_MODE	7:6	00	00	R/W	工作模式选择 00：297L 模式 01：BLE 模式 10：24L01 模式 11：未定义
	MASK_RX_DR	5	0	0	R/W	接收数据成功的中断上报使能位 1: 中断不反映到 IRQ 引脚 0: RX_DR 中断反映到 IRQ 引脚
	MASK_TX_DS	4	0	0	R/W	发送数据成功的中断上报使能位 1: 中断不反映到 IRQ 引脚 0: TX_DS 中断反映到 IRQ 引脚
	MASK_MAX_RT	3	0	0	R/W	发送失败并达到最大传输次数的中断上报使能位 1: 中断不反映到 IRQ 引脚 0: MAX_RT 中断反映到 IRQ 引脚
	CRC16_SEL	2	1	1	R/W	1：CRC16 0：CRC8
	CRC_EN	1	1	1	R/W	CRC 使能位 1: CRC 使能 0: CRC 不使能
	PRIM_RX	0	0	0	R/W	RX/TX 控制位 1: PRX 0: PTX
01	EN_AA	7:0	0x00	0x00	R/W	接收通道的自动应答使能
	Reserved	7:6	00	00		
	ENAA_P5	5	0	0	R/W	使能 pipe5 自动应答
	ENAA_P4	4	0	0	R/W	使能 pipe4 自动应答
	ENAA_P3	3	0	0	R/W	使能 pipe3 自动应答
	ENAA_P2	2	0	0	R/W	使能 pipe2 自动应答
	ENAA_P1	1	0	0	R/W	使能 pipe1 自动应答
	ENAA_P0	0	0	0	R/W	使能 pipe0 自动应答
02	AW_RXADDR	7:0	0xC1	0xC1	R/W	接收通道使能控制
	AW	7:6	11	11	R/W	RX/TX 地址宽度,如果地址宽度设置低于 5 字节，地址使用低字节 00: 无效 01: 3 字节 10: 4 字节

						11: 5 字节
	ERX_P5	5	0	0	R/W	使能 data pipe 5
	ERX_P4	4	0	0	R/W	使能 data pipe 4
	ERX_P3	3	0	0	R/W	使能 data pipe 3
	ERX_P2	2	0	0	R/W	使能 data pipe 2
	ERX_P1	1	0	0	R/W	使能 data pipe 1
	ERX_P0	0	1	1	R/W	使能 data pipe 0
03	SETUP_RX	7:0	0x20	0x20	R/W	接收通道设置寄存器
	RX_MIXL_BC[1:0]	7:6	00	00	R/W	接收端 MIXL 电流控制：00，254uA；01，318uA；10，382uA；11，444uA
	IF_CAL_CODE_CFG[5:0]	5:0	100000	100000	R/W	中频校正手动 code：cal_code_cfg[5:0] 需要与手动使能信号 cal_code_cfg_en 共同使用 中频滤波器使用软件模式触发校准，不使用手动 CODE.
04	SETUP_RX_1	7:0	0xDC	0xE0	R/W	接收通道设置寄存器 1
	RX_LNA_GC[1:0]	7:6	11	11	R/W	接收机 LNA 的增益控制：00，5.9dB；01，11.7dB；10，17.4dB；11，23.8dB；
	RX_LNA_BC[1:0]	5:4	01	10	R/W	接收端 LNA 电流控制：00，2.44mA；01，3.5mA；10，4.1mA；11，5.2mA
	RX_MIXL_GC[1:0]	3:2	11	11	R/W	接收机 MIXER 的增益控制：00,MIN;11,MAX
	RX_MIXH_BC[1:0]	1:0	00	00	R/W	接收端 MIXH 电流控制：00，0.963mA；01，1.196mA；10，1.425mA；11，1.649mA
05	SETUP_RX_2	7:0	0x04	0x97	R/W	接收通道设置寄存器 2
	RX_MIX_IQS	7	0	1	R/W	接收端 I/Q 取反开关配置
	RX_MIXH_IBLEEDING	6	0	0	R/W	接收端 MIXH Switch 管抽电流控制
	RX_MIXL_IBLEEDING	5	0	0	R/W	接收端 MIXL Switch 管抽电流控制
	RX_RCCAL_MODE	4	0	1	R/W	接收中频滤波器校准状态选择： 1：校准模式； 0：正常模式； 在中频滤波器校准过程中置 1，校准结束后置 0
	RX_RSSI_SEL_BPF	3	0	0	R/W	接收机中频测试、RSSI 输入端口选择： 1：选择滤波器； 0：不选择；
	RX_LNA_CTM[2:0]	2:0	100	111	R/W	LNA 和 PA buf 的负载电容控制：000~111，step=1 电容线性增大；RX 配置为 111，TX 保持默认配置 100
06	SETUP_RETR	7:0	0x03	0x03	R/W	发射端自动传输设置
	ARD	7:4	0000	0000	R/W	自动传输延时设置 0000 :250μs 0001 :500μs 0010 :750μs

						…… 1111: 4000μs
	ARC	3:0	0011	0011	R/W	自动传输次数设置，0000 为基本型，其他为增强型 0000: 不带自动重传不带 ACK 的通信模式 0001～1111: 带自动重传的通信模式 0001: 带 ACK 的 1 次传输 0002: 带自动重传带 ACK 的 2 次传输 …… 1111: 带自动重传带 ACK 的 15 次传输
07	RF_CH	7:0	0x00	0x00	R/W	通信频道设置
	RF_CH[7:0]	7:0	0000_0000	0000_0000	R/W	设置频道 Channel = 2336 + RF_CH + CH_OFFSET（默认 64）
08	PLL_CTL0	7:0	0x61	0x64	R/W	PLL 控制寄存器 0
	PLL_LPF_CP[1:0]	7:6	01	01	R/W	调节 Cp 电阻的大小 00 ： 3.56p 01 ： 7.12p 10 ： 10.68p 11 ： 14.24p
	PLL_PFD_DE-LAY[1:0]	5:4	10	10	R/W	调节鉴相器的复位时间 00 ： 545.4ps 01 ： 833ps 10 ： 1065ps 11 ： 1280ps
	PLL_PFD_PAL	3	0	0	R/W	PFD 的输出信号 UP DN 互换 0 ： 不互换 1 ： 互换
	PLL_LPF_RZ[2:0]	2:0	001	100	R/W	调节 Rz 电阻的大小 000：20K，001：30K，010：40K，011：50K
09	SETUP_RXTX	7:0	0x00	0x00	R/W	RXTX 通信参数配置
	TX_RAMP_RC_TRIM	7:3	00000	00000	R/W	RAMP 时间常数和步长控制
	EN_TXPA_LP	2	0	0	R/W	PA 低功率模式使能位，仅在差分版本生效
	DIG_DR[1:0]	1:0	00	00	R/W	数字 data rate 设置，与模拟滤波器 RX_BPF_BW 分开配置 00 : 1Mbps 01 : 2Mbps 10 : 125Kbps（模拟不支持） 11 : 250Kbps
0A	SETUP_RXTX_1	7:0	0xFF	0xFF	R/W	RXTX 通信参数配置 1
	TX_PA_RAMP_MODE	7	1	1	R/W	PA RAMP 模式选择 0 ： 纯模拟 1 ： 数模结合

	TX_PA2ST_BC	6:0	1111111	1111111	R/W	PA 电流档位调节，控制增益大小 000 ： 20uA 111 ： 65uA
0B	SETUP_RXTX_2	7:0	0x1B	0x1B	R/W	RXTX 通信参数配置 2
	fsm_rf_cal5	7	0	0	R/W	主状态机内部的一个控制信号，控制 ldo_en 在 ARD_WAIT 状态的值 0 ： ldo_en 取决于其他条件 1 ： ldo_en=0
	cal_code_cfg_en	6	0	0	R/W	中频滤波器校正手动 code 使能，使能后 cal_code_ out 输出为 cal_code_cfg 的值 0 ： 手动模式关闭 1 ： 手动模式使能 使用软件触发 BPF 校准模式
	TX_PA1ST_GC	5:3	011	011	R/W	PA BUF 电流档位调节，控制增益大小
	TX_PA2ST_CTM	2:0	011	011	R/W	保留位，内部无连接
0C	STATUS	7:0	0x0E	0x0E	R/W	状态寄存器
	RX_WIN_TIMEOUT	7	0	0	R/W	RX WINDOW 超时中断位，写 1 清中断
	RX_DR	6	0	0	R/W	RX FIFO 接收数据中断位，在新数据被接收并到达 RX FIFO 时产生中断，写 1 清中断
	TX_DS	5	0	0	R/W	TX FIFO 发送数据成功中断位，在不带自动重传模式下，数据发送完成后产生中断；在带自动重传模式下，仅在发送端收到 ACK 信号后才会将该位置高，写 1 清中断
	MAX_RT	4	0	0	R/W	发送达到最大传输次数未成功中断位，产生该中断后，继续进行通信必须先清该中断，写 1 清中断
	RX_P_NO	3:1	111	111	RO	可从 RX_FIFO 读取的 pipe 号 000~101 : pipe 号 110 : Not Used 111 : RX_FIFO 为空
	TX_FULL	0	0	0	RO	TX FIFO 满标志 1 : TX FIFO 满 0 : TX FIFO 未满可用
0D	OBSERVE_TX	7:0	0x00	0x00	R/W	TX 传输状态寄存器
	PLOS_CNT	7:4	0000	0000	RO	丢包计数器，该计数器达到最大值 15 时将停止计数，该计数器在写 RF_CH 时被复位，未复位该值时可以继续进行通信
	ARC_CNT	3:0	0000	0000	RO	自动重传的传输次数计数器，自动重传增加一次，ARC_CNT 加一；在 ARC_CNT 达到 ARC 限定值时，视为丢包，并将 PLOS_CNT 加一； 当新数据写入 TX FIFO 时该计数器复位
0E	PA_RAMP_CTL4	7:0	0x07	0x07	R/W	
	Reserved	7:4	0000	0000		
	dain_test	3	0	0	R/W	DA_IN[5:0]测试模式控制

						0：正常发射模式，1：测试模式，DA_IN 来自于 PAD
	PA2ST_RAMP_MAX	2:0	111	111	R/W	设置 PA RAMP 信号 TX_PA2ST_RAMP[2:0]的最大值
0F	PA_RAMP_CTL5	7:0	0xAA	0xAA	R/W	
	up1_step_mode[1:0]	7:6	10	10	R/W	PA_DLY_TIME_UP 对应计数器的 step 选择,0：1us,1：2us,2：4us,3：8us
	up2_step_mode[1:0]	5:4	10	10	R/W	RAMP_DLY_TIME_UP 对应计数器的 step 选择,0：1us,1：2us,2：4us,3：8us
	dn2_step_mode[1:0]	3:2	10	10	R/W	RAMP_DLY_TIME_DOWN 对应计数器的 step 选择,0:1us,1:2us,2：4us,3：8us
	dn1_step_mode[1:0]	1:0	10	10	R/W	PA_DLY_TIME_DOWN 对应计数器的 step 选择,0：1us,1：2us,2：4us,3：8us
10~bit[7:0] 11~bit[15:8] 12~bit[23:16] 13~bit[31:24] 14~bit[39:32]	RX_ADDR_P0	39:0	0xE7E7E7E7E7	0xCCCCCCCCCC	R/W	data pipe 0 的接收地址，最长 5 字节，地址长度由 AW 定义
15~bit[7:0] 16~bit[15:8] 17~bit[23:16] 18~bit[31:24] 19~bit[39:32]	RX_ADDR_P1	39:0	0xC2C2C2C2C2	0xC2C2C2C2C2	R/W	data pipe 1 的接收地址，最长 5 字节，地址长度由 AW 定义
1A	RX_ADDR_P3	7:0	0xC4	0xC4	R/W	data pipe 3 的接收地址
	RX_ADDR_P3	7:0	0xC4	0xC4	R/W	data pipe 3 的接收地址的低 8 位，高位等于 RX_ADDR_P1[39:8]
1B	RX_ADDR_P2	7:0	0xC3	0xC3	R/W	data pipe 2 的接收地址
	RX_ADDR_P2	7:0	0xC3	0xC3	R/W	data pipe 2 的接收地址的低 8 位，高位等于 RX_ADDR_P1[39:8]
1C	RX_ADDR_P5	7:0	0xC6	0xC6	R/W	data pipe 5 的接收地址
	RX_ADDR_P5	7:0	0xC6	0xC6	R/W	data pipe 5 的接收地址的低 8 位，高位等于 RX_ADDR_P1[39:8]
1D	RX_ADDR_P4	7:0	0xC5	0xC5	R/W	data pipe 4 的接收地址
	RX_ADDR_P4	7:0	0xC5	0xC5	R/W	data pipe 4 的接收地址的低 8 位，高位等于 RX_ADDR_P1[39:8]
1E	SETUP_RF	7:0	0x7F	0x7F	R/W	RF 设置寄存器
	Reserved	7	0	0		
	RX_BPF_GC	6:0	1111111	1111111	R/W	RX_BPF_GC<2:1>接收机中频滤波器增益控制： 000，MIN；111，MAX； RX_BPF_GC<6:3>做 BQB 认证调试用; RX_BPF_GC<6>、RX_BPF_GC<4>可改变 2M 带宽增益； RX_BPF_GC<5>、RX_BPF_GC<3>可改变 1M 带宽增益；
1F	SETUP_RF_1	7:0	0x3F	0x3F	R/W	RF 设置寄存器 1

	TXSYN_AL-WAYS_ON	7	0	0	R/W	1 ： EN_TX_SYN 常高 0 ： EN_TX_SYN 受状态机控制
	RXSYN_AL-WAYS_ON	6	0	0	R/W	1 ： EN_RX_SYN 常高 0 ： EN_RX_SYN 受状态机控制
	EN_RX_LNA	5	1	1	R/W	接收 LNA 使能控制 1 ： 打开 0 ： 关闭
	EN_RX_MIX	4	1	1	R/W	接收混频器使能控制 1 ： 打开 0 ： 关闭
	EN_RX_BPF	3	1	1	R/W	接收滤波器使能控制 1 ： 打开 0 ： 关闭
	EN_RX_LIMITER	2	1	1	R/W	接收 LIMITER 使能控制 1 ： 打开 0 ： 关闭
	EN_BPFOP_CLAMP	1	1	1	R/W	接收机滤波器运放 Clamp 使能控制 1 ： 打开 0 ： 关闭 （备注：常温下配成 1，当使用 250Kbps 速率且在 125℃高温使用时，需要配成 0）
	en_tx_sel	0	1	1	R/W	EN_TX 模式选择 1：与 EN_TX_SYN 同时打开 0：比 EN_TX_SYN 打开延迟 TX_SETUP_TIME
20	RXWIN_CTRL	7:0	0x68	0x68	R/W	RX window 设置寄存器
	rxwin_reg[7:0]	7:0	0x68	0x68	R/W	RX window 超时设置寄存器，如果在该时间内 RF RX 收不到数据，那么将触发 RX window 超时计数器溢出，进而产生 rf_rxwin_timeout_irq 中断以及自动关闭 RF 模块。RX window = rxwin_reg[20:0] * 8 us，默认 5S，最大约 16.8S
21	RXWIN_CTRL_1	7:0	0x89	0x89	R/W	RX window 设置寄存器 1
	rxwin_reg[15:8]	7:0	0x89	0x89	R/W	RX window 超时设置寄存器，如果在该时间内 RF RX 收不到数据，那么将触发 RX window 超时计数器溢出，进而产生 rf_rxwin_timeout_irq 中断以及自动关闭 RF 模块。RX window = rxwin_reg[20:0] * 8 us，默认 5S，最大约 16.8S
22	RXWIN_CTRL_2	7:0	0x09	0x09	R/W	RX window 设置寄存器 2
	Reserved	7	0	0		
	mask_rxwin_irq	6	0	0	R/W	接收窗口超时中断上报使能位 1: 中断不反映到 IRQ 0: RX_WIN 中断反映到 IRQ
	rxwin_en	5	0	0	R/W	RX window 超时计数器使能控制，1：使能，0：关闭 下次要开启 RF RX 怎么个操作 flow？把 rxwin_en 关闭或者关

						闭再打开，相当于把溢出信号给复位掉
	rxwin_reg[20:16]	4:0	01001	01001	R/W	RX window 超时设置寄存器，如果在该时间内 RF RX 收不到数据，那么将触发 RX window 超时计数器溢出，进而产生 rf_rxwin_timeout_irq 中断以及自动关闭 RF 模块。RX window = rxwin_reg[20:0] * 8 us，默认 5S，最大约 16.8S
24	DAC_CTL1	7:0	0x28	0x24	R/W	DAC 控制寄存器 1
	DAC_ISET[2:0]	7:5	001	010	R/W	DAC 输出幅度控制（细调） 000~111 幅度范围约为 24mV~138mV 1M 模式：001 2M 模式：101 1250Kbps 模式：000
	DAC_VREF_SEL[2:0]	4:2	010	001	R/W	DAC 输出共模电压控制 000~111：897mV~414mV 1M 模式：001 2M 模式：001 1250Kbps 模式：001
	DAC_GAIN_SEL[1:0]	1:0	00	00	R/W	DAC 输出摆幅调节（粗调） 00: 1X；01:4/3X；10:2X；11:4X 1M 模式：00 2M 模式：00 1250Kbps 模式：00
25	DAC_CTL2	7:0	0x08	0x08	R/W	DAC 控制寄存器 2
	Reserved	7:4				
	EN_DAC	3	1	1	R/W	DAC 使能，1: 打开； 0:关闭
	TST_DAC_OUT	2	0	0	R/W	DAC 输出测试使能，1: 打开； 0: 关闭
	TX_DAC_CLKINV	1	0	0	R/W	DAC 输入时钟相位选择 1：DAC 内部时钟与输入时钟反相 0：DAC 内部时钟与输入时钟同相
	DAC_SEL_DR	0	0	0	R/W	DAC 低通滤波器带宽选择 0: 1M 模式；1: 2M 模式
26~bit[7:0] 27~bit[15:8] 28~bit[23:16] 29~bit[31:24] 2A~bit[39:32]	TX_ADDR	39:0	0xE7E7E7E7E7	0xCCCCCCCCCC	R/W	发送端地址，只能在配置为 PTX 模式的芯片中使用，需要设置 RX_ADDR_P0 等于该地址以便接收 ACK 自动应答
2B	RX_PW_P0	7:0	0x00	0x00	R/W	data pipe 0 中的 RX payload 的数据长度
	Reserved	7	0	0		
	RX_PW_P0	6:0	0000000	0000000	R/W	data pipe 0 中的 RX payload 的数据长度（1 到 32/64 字节） 0：该 Pipe 未用 1 ：1 byte

						… 32/64 ： 32/64bytes
2C	RX_PW_P1	7:0	0x80	0x80	R/W	data pipe 1 中的 RX payload 的数据长度
	rx_sync_sel3	7	1	1	R/W	RX_BLOCK 内部 sync_flag 逻辑控制： 0：选择 rx_sync_out_reg[1] 1：选择 rx_sync_out_reg[3]
	RX_PW_P1	6:0	0000000	0000000	R/W	data pipe 1 中的 RX payload 的数据长度（1 到 32/64 字节） 0 ： 该 Pipe 未用 1 ： 1 byte … 32/64 ： 32/64 bytes
2D	RX_PW_P2	7:0	0x80	0x80	R/W	data pipe 2 中的 RX payload 的数据长度
	zc_fir_bypass	7	1	1	R/W	Zero cross 模块新增滤波器 bypass 控制 0：使用该新增的 fir 滤波器 1：bypass 该新增的 fir 滤波器
	RX_PW_P2	6:0	0000000	0000000	R/W	data pipe 2 中的 RX payload 的数据长度（1 到 32/64 字节） 0 ： 该 Pipe 未用 1 : 1 byte … 32/64 : 32/64 bytes
2E	RX_PW_P3	7:0	0x00	0x00	R/W	data pipe 3 中的 RX payload 的数据长度
	Reserved	7	0	0		
	RX_PW_P3	6:0	0000000	0000000	R/W	data pipe 3 中的 RX payload 的数据长度（1 到 32/64 字节） 0 ： 该 Pipe 未用 1 : 1 byte … 32/64 : 32/64 bytes
2F	RX_PW_P4	7:0	0x00	0x00	R/W	data pipe 4 中的 RX payload 的数据长度
	Reserved	7	0	0		
	RX_PW_P4	6:0	0000000	0000000	R/W	data pipe 4 中的 RX payload 的数据长度（1 到 32/64 字节） 0 ： 该 Pipe 未用 1 : 1 byte … 32/64 : 32/64 bytes
30	RX_PW_P5	7:0	0x00	0x00	R/W	data pipe 5 中的 RX payload 的数据长度
	RX_DATA_JUST	7	0	0	R/W	对 FSM 中 rx_data_mark 的产生有影响，297L 是常低
	RX_PW_P5	6:0	0000000	0000000	R/W	data pipe 5 中的 RX payload 的数据长度（1 到 32/64 字节） 0 ： 该 Pipe 未用 1 : 1 byte … 32/64 : 32/64 bytes

31	FIFO_STATUS	7:0	0x11	0x11	RO	FIFO 状态寄存器
	Reserved	7	0	0		
	TX_REUSE	6	0	0	RO	调用上一帧数据发送的指示位，在使用 REUSE_TX_PL 命令后，该位为 1，重传上一次发送中最后一帧数据。该位可以由命令 W_TX_PAYLOAD、W_TX_PAYLOAD_ NOACK、DEACTIVATE、FLUSH TX 进行复位操作
	TX_FULL	5	0	0	RO	TX FIFO 满标志位 1 : TX FIFO 满 0 : TX FIFO 可用
	TX_EMPTY	4	1	1	RO	TX FIFO 空标志位 1 : TX FIFO 空 0 : TX FIFO 有数据
	Reserved	3:2	00	00		
	RX_FULL	1	0	0	RO	RX FIFO 满标志位 1 : RX FIFO 满 0 : RX FIFO 可用
	RX_EMPTY	0	1	1	RO	RX FIFO 空标志位 1 : RX FIFO 空 0 : RX FIFO 有数据
32	SETUP_TXRX	7:0	0x24	0x24	R/W	TXRX 相关设置寄存器
	RX_BPF_CTM[1:0]	7:6	00	00	R/W	接收机中频滤波器 RC 电容粗调控制 00 ： 不增加 01/10 ： 增加 0.33pF 11 ： 增加 0.66pF
	RX_BPF_BUF_VSEL[2:0]	5:3	100	100	R/W	接收机中频滤波器 BUF DC 工作电压控制 000 ： 1.25V 001 ： 0.85V 010 ： 0.9V 011 ： 0.95V 100 ： 1.0V 101 ： 1.05V 110 ： 1.1V 111 ： 1.15V
	RX_BPF_VCM_VSEL[2:0]	2:0	100	100	R/W	接收机中频滤波器 BUF DC 工作电压控制 000 ： 1.15V 001 ： 0.75V 010 ： 0.8V 011 ： 0.85V 100 ： 0.9V 101 ： 0.95V 110 ： 1.0V 111 ： 1.05V
33	SETUP_TXRX_1	7:0	0xC4	0xC4	R/W	TXRX 相关设置寄存器 1

	RX_BPF_BW[3:0]	7:4	1100	1100	R/W	RX_BPF_BW<3:2>是 1M 模式小带宽挡位： 00，-6dB 带宽是 1.3M， 01，改变中心频率左边带宽； 10，改变中心频率右边带宽； RX_BPF_BW<1:0>接收机中频带宽模式选择： 00，1M 模式； 01，2M 模式； 11，250K 模式；
	RX_RSSI_SEL_MIX	3	0	0	R/W	接收机中频测试、RSSI 输入端口选择 1 ： 选择混频器 0 ： 不选择
	BALUN_CTM[2:0]	2:0	100	100	R/W	BALUN 次级线圈谐振电容选择 000 ： Min 111 ： Max
34	SETUP_TXRX_2	7:0	0x00	0x01	R/W	TXRX 相关设置寄存器 2
	Reserved	7:2	000000	000000		
	RX_BPF_BC	1	0	0	R/W	接收机中频滤波器电流控制 0 ： 小电流 1 ： 大电流
	RX_BPF_BUF_BC	0	0	1	R/W	接收机中频滤波器电流控制 0 ： 小电流； 1 ： 大电流；
35	DEMOD_CAL	7:0	0x0F	0x0F	R/W	调制解调参数寄存器
	CHIP_MODE	7	0	0	R/W	设置芯片是否进入测试模式 1: 进入测试模式 0: 退出测试模式
	CARR_MOSI	6	0	0	R/W	单载波测试使能，只影响 RX 1: CHIP=1 并且 CE=1 并且 en_rx_mode=0，EN_RF_RX=1 0：CHIP=1 并且 CE=1 并且 en_rx_mode=0，EN_RF_RX=0
	CARR_CSK	5	0	0	R/W	单载波测试使能，影响 TX 跟 RX 对 TX 的影响： 1：CHIP=1 并且 CE=0，EN_RF_TX=1 0：CHIP=1 并且 CE=0，EN_RF_TX=0 对 RX 的影响： 1：CHIP=1 并且 CE=1 并且 en_rx_mode=1，EN_RF_RX=1 0：CHIP=1 并且 CE=1 并且 en_rx_mode=1，EN_RF_RX=0
	Reserved	4	0	0		
	REG_EN_CLK_BUF	3	1	1	R/W	chip_mode=1 时，rf_xth_en= reg_en_clk_buf
	REG_EN_TX_SYN	2	1	1	R/W	txsyn_always_on=1 时，EN_TX_SYN= reg_en_tx_syn
	REG_EN_RX_SYN	1	1	1	R/W	rxsyn_always_on=1 时，EN_RX_SYN= reg_en_rx_syn
	SCR_EN	0	1	1	R/W	扰码功能是否使能

						1：使能 0：关闭
36	PMU	7:0	0x33	0xb3	R/W	电源管理寄存器
	EN_VBG_IPOLY	7	0	1	R/W	RF 电流源使能控制，使用 RF 该位必须置 1 1 ： 打开 0 ： 关闭
	VDDVCO_LDO_SEL[2:0]	6:4	011	011	R/W	模拟 LDO 输出控制 000 ： 1.306V 001 ： 1.375V 010 ： 1.445V 011 ： 1.513V 100 ： 1.588V 101 ： 1.605V 110 ： 1.724V 111 ： 1.791V
	EN_RSSI	3	0	0	R/W	接收中频信号强度检查使能控制 1 ： 打开校准 0 ： 关闭校准
	VDDDPLL_LDO_SEL[2:0]	2:0	011	011	R/W	模拟 LDO 输出控制 000 ： 1.306V 001 ： 1.375V 010 ： 1.445V 011 ： 1.513V 100 ： 1.588V 101 ： 1.605V 110 ： 1.724V 111 ： 1.791V
37	PMU_1	7:0	0x33	0x3b	R/W	电源管理寄存器 1
	Reserved	7	0	0		
	VDDIF_LDO_SEL[2:0]	6:4	011	011	R/W	模拟 LDO 输出控制 000 ： 1.306V 001 ： 1.375V 010 ： 1.445V 011 ： 1.513V 100 ： 1.588V 101 ： 1.605V 110 ： 1.724V 111 ： 1.791V
	EN_RCCAL	3	0	1	R/W	接收中频滤波器校准使能 1 ： 打开 0 ： 关闭

						在中频滤波器校准过程中置 1，校准结束后置 0
	VDDLO_LDO_SEL[2:0]	2:0	011	011	R/W	模拟 LDO 输出控制 000 ： 1.306V 001 ： 1.375V 010 ： 1.445V 011 ： 1.513V 100 ： 1.588V 101 ： 1.605V 110 ： 1.724V 111 ： 1.791V
38	PMU_2	7:0	0x43	0x43	R/W	电源管理寄存器 2
	IF_CAL_WAIT[4:0]	7:3	01000	01000	R/W	中频滤波器校正之前的等待周期，等待该周期个 LIMIT 信号之后才开始校正，留给模拟电路一些稳定时间
	VDDRX_LDO_SEL[2:0]	2:0	011	011	R/W	模拟 LDO 输出控制 000 ： 1.306V 001 ： 1.375V 010 ： 1.445V 011 ： 1.513V 100 ： 1.588V 101 ： 1.605V 110 ： 1.724V 111 ： 1.791V
39	DEM_CTL	7:0	0xC5	0xC5	R/W	解调参数寄存器
	DECOD_INV	7	1	1	R/W	前导码是否按位取反，使能该功能需要收发两端同时进行，一般置 1 1：不按位取反 0：按位取反
	AGGRESSIVE	6	1	1	R/W	解调器的码率同步单元的速度选择 1：大步长调整，速度快 0：小步长调整，速度慢
	GAIN2	5:0	000101	000101	R/W	解调器的数据中心值调整环路的根据基准波形的调整速度，置 000101
3A	DEM_CTL_2	7:0	0x6E	0x6E	R/W	解调参数寄存器 2
	PTH	7:4	0110	0110	R/W	接收机数字解调器前导码相关阈值设置，24 位前导码的相关阈值=PTH+16 0000 ： 16 位 0001 ： 17 位 …… 0110 ： 22 位 1000 ： 24 位
	GAIN1	3:0	1110	1110	R/W	解调器的数据中心值调整环路的基准波形的幅度，置 1110

3B	DEM_CTL_3	7:0	0x09	0x0b	R/W	解调参数寄存器 3
	PIN_MODE	7:5	000	000	R/W	设置 demod_en 的控制逻辑 000（且 CHIP_MODE 为 1）为常 1 否则 demod_en 受 RX 状态机控制
	EN_RX_MODE	4	0	0	R/W	接收通道是否与锁相环同时开启 1: 同时打开 0: 分时打开
	REG_EN_LDO_1P8	3	1	1	R/W	SIGNAL_TH1 等于 1 时，EN_LDO_ANA 等于 REG_EN_LDO_1P8
	DELAY0	2	0	0	R/W	解调器是否叠加收报的初始偏移量，解调器不叠加初始偏移量可作为接收灵敏度测试 1: 不叠加初始偏移量 0: 叠加初始频偏，接收状态下可以抵消由于中心频偏引起的误码
	SIGNAL_TH1	1	0	1	R/W	EN_LDO_ANA 是否使能，在测试模式下，测试发射单载波和接收灵敏度时该位置 1 1 : EN_LDO_ANA 等于 REG_EN_LDO_1P8 0 : EN_LDO_ANA 受状态机控制
	SYNC_SEL	0	1	1	R/W	接收机数字解调器的 4 倍采样，取几点相关上计算该位数据正确 1 : 3bit 0 : 2bit
3C	DYNPD	7:0	0x00	0x00	R/W	动态长度 PAYLOAD 使能控制
	Reserved	7	0	0		
	noIRQ_noWORK	6	0	0	R/W	控制主状态机、TX、RX 状态机 1 ： 不清 IRQ 数字所有状态机不工作 0 ： 不清 IRQ 数字状态机也工作
	DPL_P5	5	0	0	R/W	使能 PIPE 5 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P5)
	DPL_P4	4	0	0	R/W	使能 PIPE 4 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P4)
	DPL_P3	3	0	0	R/W	使能 PIPE 3 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P3)
	DPL_P2	2	0	0	R/W	使能 PIPE 2 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P2)
	DPL_P1	1	0	0	R/W	使能 PIPE 1 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P1)
	DPL_P0	0	0	0	R/W	使能 PIPE 0 动态 PAYLOAD 长度 (需要 EN_DPL 和 ENAA_P0)
3D	FEATURE	7:0	0x80	0x80	R/W	特征寄存器
	RX_BYTE_REV	7	1	1	R/W	读取 RX fifo 时字节序取反

						1：取反 0：不取反
	RX_BIT_REV	6	0	0	R/W	RX 数据写入 fifo 时每个 byte 的 bit 序取反 1：取反 0：不取反
	TX_BIT_REV	5	0	0	R/W	TX 数据写入 fifo 时每个 byte 的 bit 序取反 1：取反 0：不取反
	DATA_LEN_SEL	4:3	00	00	R/W	数据长度选择 11 : 64byte（512bit）模式 00 : 32byte（256bit）模式
	EN_DPL	2	0	0	R/W	RX 端动态 PAYLOAD 长度使能控制 1 ： 打开 0 ： 关闭
	EN_ACK_PAY	1	0	0	R/W	使能 ACK 带 PAYLOAD 功能 1 ： 打开 0 ： 关闭
	EN_DYN_ACK	0	0	0	R/W	使能 W_TX_PAYLOAD_NOACK 命令 1 ： 打开 0 ： 关闭
3E	PLL_CTL1	7:0	0x52	0xb2	R/W	PLL 控制寄存器 1
	PLL_LPF_R3[1:0]	7:6	01	10	R/W	调节 R3 电阻大小 00 ： 5K 01 ： 10K 10 ： 15K 11 ： 20K
	PLL_LPF_C3[1:0]	5:4	01	11	R/W	调节 C3 电容大小 00 ： 3.56p 01 ： 7.12p 10 ： 10.68p 11 ： 14.24p
	PLL_BYP_FT	3	0	0	R/W	PLL 模块中引脚 CP_TEST 与引脚 VC_TEST 测试使能控制 1 ： 使能 0 ： 关闭
	PLL_LPF_VSEL[2:0]	2:0	010	010	R/W	VCO Calibration 电压输出选择 00 ： 496mV 01 ： 597mV 10 ： 691mV 11 ： 781mV VCO 2 点式校准中，可以选择 1XX （VC 为 GND 0V），同时也可以选择 0XX 不同的 VC 电压

						VCO 两点式校准电压输出 00 496mV 01 597mV 10 691mV 11 781mV
3F	PLL_CTL2	7:0	0xA9	0xA9	R/W	PLL 控制寄存器 2
	PLL_VCO-BIAS_ISEL[1:0]	7:6	10	10	R/W	控制 VCO 中与温度无关的电流，步进 2.5uA 00 ： 0uA 01 ： 2.5uA 10 ： 5uA 11 ： 7.5uA
	PLL_VCOP-TAT_ISEL[1:0]	5:4	10	10	R/W	控制 VCO 中与温度相关的电流，步进 2.5uA 00 ： 0uA 01 ： 2.5uA 10 ： 5uA 11 ： 7.5uA
	PLL_VCO_CT	3	1	1	R/W	两点式小 kvco 调节 1 ： 30MHz/V 0 ： 15MHz/V
	PLL_DIV2_ISEL[2:0]	2:0	001	001	R/W	PLL 除 2 偏置电流调节 000 ： 7.5uA …… 111 ： 17.5uA
40	BB_CAL	7:0	0xCA	0xCA	R/W	数字基带参数寄存器
	WAIT_COCLUDE_TIME[1:0]	7:6	11	11	R/W	TX 状态机 TX_CONCLUDE 状态的等待时间，计算公式为 (WAIT_CONCLUDE_TIME+1)x 1/data_rate，单位 us 2M 模式 ： 1/data_rate = 0.5 1M 模式 ： 1/data_rate = 1 250K 模式 ： 1/data_rate = 4
	RX_ACK_TIME[5:0]	5:0	001010	001010	R/W	PTX 转为接收模式后等待 ACK 的最长时间，超出该时间则认为本次传输失败 2Mbps 模式下的时间长度计算： (RX_ACK_TIME[5:0]×16)+16，单位为 us 1Mbps 模式下的时间长度计算： (RX_ACK_TIME[5:0]×32)+32，单位为 us 250kbps 模式下的时间计算： (RX_ACK_TIME[5:0]×128)+128，单位为 us
41	BB_CAL_2	7:0	0x94	0x00	R/W	数字基带参数寄存器 2
	LDO_WAIT_TIME[2:0]	7:5	100	000	R/W	LDO 稳定时间，之前是 30us 做死的，298 改为可配置，时间长度计算： LDO_WAIT_TIME[2:0] x 8，单位 us 范围 0~56us，默认 32us
	RX_SETUP_TIME[4:0]	4:0	10100	00000	R/W	RX 射频通路锁相环稳定时间， 时间长度计算：

						RX_SETUP_TIME[4:0]×16，单位为 us 范围 0~496us，默认 320us
42	BB_CAL_3	7:0	0x54	0x44	R/W	数字基带参数寄存器 3
	CE_JUST_TIME[1:0]	7:6	01	01	R/W	主状态机从 STANDBY 进入 TX_SET 的等待时间，时间长度计算： CE_JUST_TIME[1:0]x4，单位 us 范围 0~12us，默认 4us
	TX_SETUP_TIME[5:0]	5:0	010100	000100	R/W	发射锁相环使能到开始发数据的时间间隔，时间长度计算： TX_SETUP_TIME[5:0]×16，单位为 us 范围 0~1008us，默认 320us
43	BB_CAL_4	7:0	0x87	0x87	R/W	数字基带参数寄存器 4
	INVERTER	7	1	1	R/W	进入 RX_block 前是否取反 RX 通路数据 1: 取反 0: 保持不变
	dac_mode	6	0	0	R/W	发射 DA_IN 是否倒序 1：倒序 0：不倒序
	EX_PA_TIME[5:0]	5:0	000111	000111	R/W	发射锁相环使能到 PA 使能的时间间隔，时间长度计算： EX_PA_TIME[5:0]×16，单位为 us 范围 0~1008us，默认 112us
44	BB_CAL_5	7:0	0x04	0x04	R/W	数字基带参数寄存器 5
	TRX_TIME[3:0]	7:4	0000	0000	R/W	锁相环开环/PA 使能（以后开者为基准）到开始发射数据的时间间隔，时间长度计算： TRX_TIME[3:0]×8，单位为 us 范围 0~120us，默认 0us
	WAIT_RAMP_DN_TIME[3:0]	3:0	0100	0100	R/W	TX 端发完数据到开始 ramp down 的等待时间，包含在 TX_WAITSTOP 状态之内 WAIT_RAMP_DN_TIME[3:0]x0.5，单位 us 范围 0~7.5us，默认 2us
45~bit[7:0] 46~bit[15:8] 47~bit[23:16] 48~bit[31:24] 49~bit39:32] 4A~bit[47:40]	WL_ADDR	47:0	0x000000000000	0x000000000000	R/W	BLE RX 模式白名单设置，该白名单应该为 028.payload 当中的某一段（可配置），跟寄存器 PLD_START_BYTES 以及 WL_MATCH_ MODE 配合使用
4B	WL_MODE	7:0	0x10	0x10	R/W	白名单匹配模式选择
	IF_CAL_EN_M_SEL	7	0	0	R/W	中频滤波器校正触发方式选择： 1：通过软件手动触发，0：通过状态机自动触发
	wait_test_cnt[2:0]	6:4	001	001	R/W	IF 滤波器校正模块大周期内循环校准一次后等待模拟反馈的时间（一次大周期有六次循环校正），默认等待一个校准时钟周

						期，最终等待时间为配置值加 1。整个校准时间是由等待时间 wait_test_cnt 的长短来确定的，整个校准时间约为（wait_test_cnt+3）x 校准频率周期（1/2e6）x 6 次
	FIR_CUT_MODE	3	0	0	R/W	解调器内部 fir_filter 模式选择 0 ： 297L 模式 1 ： 新增滤波器溢出保护模式
	WL_MATCH_MODE[2:0]	2:0	000	000	R/W	白名单过滤模式选择 000：不过滤，全部上报 001：只需匹配上 WL_ADDR[47:40]即上报 010：只需匹配上 WL_ADDR[47:32]即上报 011：只需匹配上 WL_ADDR[47:24]即上报 100：只需匹配上 WL_ADDR[47:16]即上报 101：只需匹配上 WL_ADDR[47:8]即上报 110：需要 WL_ADDR[47:0]全部匹配即上报 111：同 000，不过滤全部上报
4C~bit[7:0] 4D~bit[15:8] 4E~bit[23:16] 4F~bit[31:24]	ACCESS_ADD	31:0	0x8E89BED6	0x8E89BED6	R/W	BLE 模式 Access Address 设置 广播包 ： 0x8E89BED6 数据包 ： 可配置
50~bit[7:0] 51~bit[15:8]	DEV_COM_P	15:0	0x0000	0x0000	R/W	整体定向频偏向上调节，步进值为 2M 模式 ： 步进值约 4KHz 1M 模式 ： 步进值约 514Hz 250K 模式 ： 步进值约 8Hz
52~bit[7:0] 53~bit[15:8]	DEV_COM_N	15:0	0x0000	0x0000	R/W	整体定向频偏向下调节，步进值为 2M 模式 ： 步进值约 4KHz 1M 模式 ： 步进值约 514Hz 250K 模式 ： 步进值约 8Hz
54	PA_RAMP_CTL0	7:0	0x4F	0x4a	R/W	PA RAMP 控制寄存器 0
	RAMP_STEP_UP[2:0]	7:5	010	010	R/W	TX_PA2ST_RAMP UP 步长设置 000 ： 0.5us 001 ： 1us 010 ： 2us 011 ： 4us Others : 8us
	PA_DLY_TIME_UP[4:0]	4:0	01111	01010	R/W	EN_TX_SYN/EN_TX 到 EN_TX_PA2 的时间 (PA_DLY_TIME_UP[4:0]+1)*($2^{up1\_step\_mode}$)us 0~256us 可配，默认 64us
55	PA_RAMP_CTL1	7:0	0x4F	0x40	R/W	PA RAMP 控制寄存器 1

	RAMP_STEP_DN[2:0]	7:5	010	010	R/W	TX_PA2ST_RAMP DOWN 步长设置 000 ： 0.5us 001 ： 1us 010 ： 2us 011 ： 4us Others : 8us
	PA_DLY_TIME_DN[4:0]	4:0	01111	00000	R/W	EN_TX_PA2 拉低到 EN_TX_SYN/EN_TX 拉低的时间， (PA_DLY_TIME_DN[4:0]+1)*($2^{dn1\_step\_mode}$)us 0~256us 可配，默认 64us
56	PA_RAMP_CTL2	7:0	0x77	0x22	R/W	PA RAMP 控制寄存器 2
	RAMP_DLY_TIME_DN[3:0]	7:4	0111	0010	R/W	EN_RAMP 拉低到 EN_TX_PA2 拉低的时间 (RAMP_DLY_TIME_DN[3:0]+1)*($2^{dn2\_step\_mode}$) us 0~128us 可配，默认 32us
	RAMP_DLY_TIME_UP[3:0]	3:0	0111	0010	R/W	EN_TX_PA2 拉高到 EN_RAMP 拉高的时间 (RAMP_DLY_TIME_UP[3:0]+1)*($2^{up2\_step\_mode}$) us 0~128us 可配，默认 32us
57	PA_RAMP_CTL3	7:0	0x7F	0x7F	R/W	PA RAMP 控制寄存器 3
	Reserved	7	0	0		
	EN_PLL_RXDFF	6	1	1	R/W	RX 通路 DFF 除 2 模块使能控制 1 ： 打开 0 ： 关闭
	REG_EN_TX_PA1	5	1	1	R/W	EN_TX_PA1 控制信号 1 ： 打开 0 ： 关闭
	REG_EN_TX_PA2	4	1	1	R/W	测试模式下 EN_TX_PA2 由该信号控制，否则由状态机控制 1 ： 打开 0 ： 关闭
	REG_EN_RAMP	3	1	1	R/W	测试模式下 EN_RAMP 由该信号控制，否则由状态机控制 1 ： 打开 0 ： 关闭
	REG_TX_PA2ST_RAMP[2:0]	2:0	111	111	R/W	测试模式下 TX_PA2ST_RAMP 由该信号控制，否则由状态机控制
58	PLL_CTL3	7:0	0x28	0x30	R/W	PLL 控制寄存器 3
	EN_PLL_FBDIVTST	7	0	0	R/W	PLL 分频器 PCLK 信号测试使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_CALDIV-TST	6	0	0	R/W	VCO 校准与带宽校准时钟信号测试使能 1 ： 打开 0 ： 关闭
	EN_PLL_BC	5	1	1	R/W	PLL 模块偏置使能控制 1 ： 打开

						0 ： 关闭
	PLL_RX_VCO_BC[4:0]	4:0	01000	10000	R/W	控制 RX VCO 的电流，0000：550uA，1111：2681.25uA， 步进 68.75uA
59	PLL_CTL4	7:0	0xE8	0xf0	R/W	PLL 控制寄存器 4
	EN_PLL_PFD	7	1	1	R/W	PFD 模块使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_CP	6	1	1	R/W	CP 模块使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_CP_SHIFT	5	1	1	R/W	CP_SHIFT 电流补偿 1 ： 打开 0 ： 关闭
	PLL_TX_VCO_BC[4:0]	4:0	01000	10000	R/W	控制 TX VCO 的电流，0000：550uA，1111：2681.25uA， 步进 68.75uA
5A	PLL_CTL5	7:0	0xF6	0xF6	R/W	PLL 控制寄存器 5
	EN_PLL_CP_CMP	7	1	1	R/W	CP 模块中运放使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_LPF	6	1	1	R/W	LPF 模块使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_VCO	5	1	1	R/W	VCO 模块使能控制 1 ： 打开 0 ： 关闭
	EN_PLL_DIV2	4	1	1	R/W	PLL 除 2 模块使能控制 1 ： 打开 0 ： 关闭
	PLL_DIV2_VB[1:0]	3:2	01	01	R/W	PLL 除 2 电路 DC 电压调节 00 ： 940Mv 11 ： 1100Mv
	PLL_VCO_CAPSEL[1:0]	1:0	10	10	R/W	PLL VCO 电容选择档位，11 最大，00 最小
5B	DIG2_REG0	7:0	0xD2	0xD2	R/W	PLL 小数字控制寄存器 0
	fsynvcoatcntre-fovrdwd[7:0]	7:0	8’hD2	8’hD2	R/W	单次 vco 校正模式的参考计数值手动改写值，与目标频率的换算公式为：cntrefovrdwd= vcomxcnt*0.0625us*freq/(2^(3+caldivclk))
5C	DIG2_REG0_2	7:0	0x05	0x05	R/W	PLL 小数字控制寄存器 0_2
	fsynvcoatcalovrd	7	0	0	R/W	最终使用的 vco_code 是否手动改写 1 ： 手动改写 0 ： 非手动改写

	fsynvcorxmode_ovrd	6	0	0	R/W	vco 校正 RX 模式的手动改写值，在单次校正时使用，优先级低于最终使用的 code（fsynvcoatftune ovrd[5:0]） 1 ： 使能 0 ： 关闭
	fsynvcorxmode_sel	5	0	0	R/W	vco 校正 RX 模式是否手动改写 1 ： 手动改写 0 ： 非手动改写
	fsynvcoat_chgrovrd_sel_rx	4	0	0	R/W	RX 模式 5 组保存的 vco code 是否手动改写选择，1 ： 改写 0 ： 不改写
	fsynvcoatcntre-fovrdwd[11:8]	3:0	0101	0101	R/W	单次 vco 校正模式的参考计数值手动改写值，与目标频率的换算公式为：cntrefovrdwd= vcomxcnt*0.0625us*freq/(2^(3+caldivclk))
5D	DIG2_REG0_3	7:0	0x9E	0x9E	R/W	PLL 小数字控制寄存器 0_3
	fsynvcoatmxcnt[7:0]	7:0	10011110	10011110	R/W	vco 校正参考时钟计数个数选择，决定了校正时长,时长计算公式为：62.5ns*fsynvcoatmxcnt。默认配置下计数时长为 9.875us
5E	DIG2_REG0_4	7:0	0x80	0x80	R/W	PLL 小数字控制寄存器 0_4
	fsynvcoatftune-ovrd[3:0]	7:4	1000	1000	R/W	最终使用的 vco code 手动改写值
	fsynvcoatstartmchcal	3	0	0	R/W	VCO 校正使能 1 ： 使能校正 0 ： 关闭校正
	Reserved	2:1	00	00		
	fsynvcoatmxcnt[8]	0	0	0	R/W	vco 校正参考时钟计数个数选择，决定了校正时长,时长计算公式为：62.5ns*fsynvcoatmxcnt。默认配置下计数时长为 9.875us
5F	FIR_CAL1	7:0	0x60	0x60	R/W	中频滤波器校正控制
	ref_count[7:0]	7:0	0x60	0x60	R/W	两个校准时钟周期内的基准 count 值，本设计为 96M/2M *2 1Mbps 模式：0x60; 2Mbps 模式：0x60; 250Kbps 模式：0x62;（备注：高温 125 摄氏度需要配成 66，配合 SETUP_RF_1=3d）
60	RF_CE	7:0	0x00	0x00	R/W	RF 收发机片选使能控制
	Reserved	7:1	0000000	0000000		
	CE	0	0	0	R/W	RF 收发机片选使能控制 1 ： 使能 0 ： 关闭
61	RF_CMD	7:0	0x00	0x00	R/W	特殊 CMD 寄存器，实现特殊的功能
	RF_CMD	7:0	0x00	0x00	R/W	8`h61 : R_RX_PAYLOAD 8`hA0 : W_TX_PAYLOAD 8`hE1 : FLUSH_TX 8`hE2 : FLUSH_RX 8`hE3 : REUSE_TX_PL

						8`h73 : ACTIVE 8`h8C : DEACTIVE 8`b10101ppp : W_ACK_PAYLOAD 8`hB0 : W_TX_PAYLOAD_NOACK 8`h5A : RST_HOLD 8`hA5 : RST_RELS 8`hFF : RF_NOP
62	RF_FIFO	7:0	0x00	0x00	R/W	读/写 FIFO 入口寄存器
	RF_FIFO	7:0	0x00	0x00	R/W	TX 模式 : TX FIFO 入口寄存器，1~64 bytes RX 模式 : RX FIFO 入口寄存器，1~64 bytes
63	RF_PL_WIDTH	7:0	0x00	0x00	RO	只读 register，RX Payload 长度信息
64	DIG2_REG1	7:0	0x88	0x88	R/W	PLL 小数字控制寄存器 1
	Chgr1_ov-rdwd_rx[3:0]	7:4	1000	1000	R/W	使用的 rx 模式第 1 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel_rx 配 1 时生效
	Chgr0_ov-rdwd_rx[3:0]	3:0	1000	1000	R/W	使用的 rx 模式第 0 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel_rx 配 1 时生效
65	DIG2_REG1_2	7:0	0x88	0x88	R/W	PLL 小数字控制寄存器 1_2
	Chgr3_ov-rdwd_rx[3:0]	7:4	1000	1000	R/W	使用的 rx 模式第 3 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel_rx 配 1 时生效
	Chgr2_ov-rdwd_rx[3:0]	3:0	1000	1000	R/W	使用的 rx 模式第 2 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel_rx 配 1 时生效
66	DIG2_REG1_3	7:0	0x08	0x08	R/W	PLL 小数字控制寄存器 1_3
	Reserved	7:4	0000	0000		
	Chgr4_ov-rdwd_rx[3:0]	3:0	1000	1000	R/W	使用的 rx 模式第 4 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel_rx 配 1 时生效
67	DIG2_REG2	7:0	0x88	0x88	R/W	PLL 小数字控制寄存器 2
	Chgr1_ov-rdwd_tx[3:0]	7:4	1000	1000	R/W	使用的 tx 模式第 1 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel 配 1 时生效
	Chgr0_ov-rdwd_tx[3:0]	3:0	1000	1000	R/W	使用的 tx 模式第 0 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel 配 1 时生效
68	DIG2_REG2_2	7:0	0x88	0x88	R/W	PLL 小数字控制寄存器 2_2
	Chgr3_ov-rdwd_tx[3:0]	7:4	1000	1000	R/W	使用的 tx 模式第 3 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel 配 1 时生效
	Chgr2_ov-rdwd_tx[3:0]	3:0	1000	1000	R/W	使用的 tx 模式第 2 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel 配 1 时生效
69	DIG2_REG2_3	7:0	0x08	0x08	R/W	PLL 小数字控制寄存器 2_3
	Reserved	7:4	0000	0000		
	Chgr4_ov-rdwd_tx[3:0]	3:0	1000	1000	R/W	使用的 tx 模式第 4 组 vco_code 手动改写值，fsynvcoat_chgrovrd_sel 配 1 时生效
6A	DIG2_REGC_1	7:0	0x10	0x10	R/W	PLL 小数字控制寄存器 C_1

	Reserved	7:5	000	000		
	Two_point_manul_code_in3	4:0	10000	10000	R/W	两点式校正第三组手动 code。默认值 5'd16. 两点式自动校准 CODE 软件操作将校准后的 code-1
6B	DIG2_REGC_2	7:0	0x10	0x10	R/W	PLL 小数字控制寄存器 C_2
	Reserved	7:5	000	000		
	Two_point_manul_code_in2	4:0	10000	10000	R/W	两点式校正第二组手动 code。默认值 5'd16. 两点式自动校准 CODE 软件操作将校准后的 code-1
6C	DIG2_REGC_3	7:0	0x10	0x10	R/W	PLL 小数字控制寄存器 C_3
	Reserved	7:5	000	000		
	Two_point_manul_code_in1	4:0	10000	10000	R/W	两点式校正第一组手动 code。默认值 5'd16. 两点式自动校准 CODE 软件操作将校准后的 code-1
6D	DIG2_REGC_4	7:0	0x10	0x10	R/W	PLL 小数字控制寄存器 C_4
	Reserved	7:5	000	000		
	Two_point_manul_code_in0	4:0	10000	10000	R/W	两点式校正第零组手动 code。默认值 5'd16. 两点式自动校准 CODE 软件操作将校准后的 code-1
6E	DIG2_REG3	7:0	0x40	0x40	R/W	PLL 小数字控制寄存器 3
	Ch_offset[7:0]	7:0	0100_0000	0100_0000	R/W	配置频点起始值，默认值为 8'd64，这样 tx 频率起始 2400M，rx 频率起始 2402M
6F	DIG2_REG3_2	7:0	0x79	0x57	R/W	PLL 小数字控制寄存器 3_2
	Vco_freq_cover_tx[3:0]	7:4	0111	0101	R/W	vco 多次校正 tx 模式每个 code 覆盖范围选择，默认值为 4'd7，此配置下每个 code 覆盖范围 ch-14～ch+14.
	Vco_freq_cover_rx[3:0]	3:0	1001	0111	R/W	vco 多次校正 rx 模式每个 code 覆盖范围选择，默认值为 4'd9，此配置下每个 code 覆盖范围 ch-18～ch+18.
70	DIG2_REG3_3	7:0	0x84	0x84	R/W	PLL 小数字控制寄存器 3_3
	error_limit[3:0]	7:4	1000	1000	R/W	用于加快 vco 校正的配置。校正过程中，如果计数值减去参考值大于 error_limit，则不用等计数完规定窗口直接跳到下一个 code，可加快校正速度，来自 spi。仅在校正结果小于 32 时生效。
	vcocalen	3	0	0	R/W	vco_cal 相关时钟使能，1 为使能
	Reserved	2:0	100	100	R/W	
71	DIG2_REG3_4	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 3_4
	fsynvcoat_chgrovrd_sel	7	0	0	R/W	TX 模式保存的 5 组 vco code 是否手动改写选择，优先级低于最终使用的 code（fsynvcoatftune ovrd[5:0]） 1 ： 改写 0 ： 不改写
	fsynvcoaten_ovrd	6	0	0	R/W	vco_cal 使能手动改写值，可应用于单次校正
	fsynvcoaten_sel	5	0	0	R/W	vco_cal 使能是否手动改写，配 1 为改写

	Reserved	4	0	0	R/W	
	fsynvcoatmchcalovrd	3	0	0	R/W	单次校正计数参考值是否手动改写选择，1 为手动改写
	fsynvcoat_cal_mode[2:0]	2:0	000	000	R/W	vco_cal 模式选择，配置 [1:0]为 00 : curve based mode，多次校正，10 个频道 01 :channel based 比较 6 次，单次校正 11 :channel based 比较 3 次，单次校正
72	DIG2_REG4	7:0	0x60	0x60	R/W	PLL 小数字控制寄存器 4
	fsynsden	7	0	0	R/W	SDM 手动使能，1 为手动使能
	Tp_code_group_sel[1:0]	6:5	11	11	R/W	两点式校正使用几组手动 code 选择，默认值 2'd3，为使用 4 组 code。配置 0/1/2/3 分别对应使用 1/2/3/4 组 code.
	En_two_point_cal	4	0	0	R/W	两点式校正 code 使用自动还是手动选择, 1:自动模式，0:手动模式。默认手动，后续应用也应该使用手动模式。
	cntref_mem_addr[3:0]	3:0	0000	0000	R/W	curve_based vco_cal（多次校正）参考计数值手动改写地址，可配 0~9，0~4 对应 TX，5~9 对应 RX
73	DIG2_REG4_2	7:0	0x60	0x60	R/W	PLL 小数字控制寄存器 4_2
	Tp_done_sel[2:0]	7:5	011	011	R/W	两点式校正计数值与参考计数值差多少即结束校正选择，默认 3'd3.该值越大，理论上校正精度越差。
	Inband_delay[4:0]	4:0	00000	00000	R/W	带内发射延时选择，默认 0
74	DIG2_REG4_3	7:0	0x04	0x04	R/W	PLL 小数字控制寄存器 4_3
	Code_offset[2:0]	7:5	000	000	R/W	两点式校正自动模式的 code 加的 offset 值，有符号数，默认 0。自动模式下，校准出来的 code 会加上本值再使用。
	Phase_adj	4	0	0	R/W	调整 gauss_filter 内 16M 时钟对发射数据的采样相位
	fsyncaldivcalclk[1:0]	3:2	01	01	R/W	vco_cal/bw_cal 使用的时钟为 vco 几分频选择， 00 为 8 分频（约 300M），该版本不支持。 01 为 16 分频 10 为 32 分频
	Window_int_sel[1:0]	1:0	00	00	R/W	两点式校正模式 2 取窗口间隔选择，默认 2'd0，间隔 1us
75	DIG2_REG4_4	7:0	0x6A	0x6A	R/W	PLL 小数字控制寄存器 4_4
	Ctl_dither_lsb[2:0]	7:5	011	011	R/W	DSM 加抖系数选择，默认 3'd3
	Tp_code_cover[4:0]	4:0	01010	01010	R/W	两点式校正每个 code 覆盖范围选择，默认 5'd10，此配置下每个 code 覆盖范围 ch-10～ch+10，共 21M。4 个 code 共覆盖 84M
76	DIG2_REG4_5	7:0	0xF8	0xF8	R/W	PLL 小数字控制寄存器 4_5
	Ctl_dither_shape	7	1	1	R/W	DSM 加抖系数使能，默认 1
	Reserved	6:5	11	11	R/W	
	Gauss_scale[4:0]	4:0	11000	11000	R/W	带内高斯的 scale 值，默认值为 5'd24 df_peak: 1M:11000 2M:11111 250Kbps：01111
77	DIG2_REG5	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 5

	frequency_offset_ov-rdwd[7:0]	7:0	0x00	0x00	R/W	分频值小数部分手动改写值
78	DIG2_REG5_2	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 5_2
	frequency_offset_ov-rdwd[15:8]	7:0	0x00	0x00	R/W	分频值小数部分手动改写值
79	DIG2_REG5_3	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 5_3
	Shift_offset	7	0	0	R/W	锁相环 DELTA-SIGMA 调制器移位偏置使能设置，默认 0
	frequency_offset_ov-rdwd[22:16]	6:0	000_0000	000_0000	R/W	分频值小数部分手动改写值
7A	DIG2_REG5_4	7:0	0x00	0x3a	R/W	PLL 小数字控制寄存器 5_4
	frequency_correc-tion_rx[7:0]	7:0	0x00	0x3a	R/W	rx 模式分频值小数部分修正值，最终的小数部分= fre-quency_correction_rx[17:0]+小数初始值
7B	DIG2_REG5_5	7:0	0x00	0xff	R/W	PLL 小数字控制寄存器 5_5
	frequency_correc-tion_rx[15:8]	7:0	0x00	0xff	R/W	rx 模式分频值小数部分修正值，最终的小数部分= fre-quency_correction_rx[17:0]+小数初始值
7C	DIG2_REG5_6	7:0	0x78	0x7b	R/W	PLL 小数字控制寄存器 5_6
	fsynsdnint_ovrd[5:0]	7:2	011110	011110	R/W	分频值整数部分手动改写值
	frequency_correc-tion_rx[17:16]	1:0	00	11	R/W	rx 模式分频值小数部分修正值，最终的小数部分= fre-quency_correction_rx[17:0]+小数初始值
7D	DIG2_REG5_7	7:0	0xD2	0xD2	R/W	PLL 小数字控制寄存器 5_7
	cntref_mem_din[7:0]	7:0	1101_0010	1101_0010	R/W	curve_based vco_cal（多次校正）参考计数值手动改写值. cntref = vcomxcnt*0.0625us*freq/(2^(3+caldivclk))
7E	DIG2_REG5_8	7:0	0x05	0x05	R/W	PLL 小数字控制寄存器 5_8
	cntref_mem_wre	7	0	0	R/W	curve_based vco_cal（多次校正）参考计数值手动改写使能，使用时先配置地址 cntref_mem_addr[3:0] 和写入值 cntref_mem_din[11:0]，然后该 wre 配一个高脉冲信号（先配 1 再配 0），即可把当前值写入当前地址，总共 10 组
	frequency_offset_ov-rdwd_sel	6	0	0	R/W	分频值小数部分手动改写选择，1 为手动改写
	fsynsdnint_sel	5	0	0	R/W	分频值整数部分手动改写选择，1 为改写
	Reserved	4	0	0	R/W	
	cntref_mem_din[11:8]	3:0	0101	0101	R/W	curve_based vco_cal（多次校正）参考计数值手动改写值. cntref = vcomxcnt*0.0625us*freq/(2^(3+caldivclk))
7F	DIG2_REG6	7:0	0x1B	0x19	R/W	PLL 小数字控制寄存器 6
	Two_point_clk_en	7	0	0	R/W	两点式校正时钟开关，做校正前打开，会把送给模拟的 caldiven 拉高，lfdacen 拉高，EN_TX_SYN 拉高。默认 0。做完校正需要手动拉低。
	Two_point_spi_trig	6	0	0	R/W	两点式校正触发信号，下降沿触发。默认 0
	Df_sel[5:0]	5:0	011011	011001	R/W	与 Gauss_scale 配合使用以调整带内 deviation，默认值 6'd27 df_sel: 1M：011001

						2M：110010 250Kbps：011001
80	DIG2_REG6_2	7:0	0xA0	0xA0	R/W	PLL 小数字控制寄存器 6_2
	Tp_cal_mode_sel[1:0]	7:6	10	10	R/W	两点式校正模式选择，默认 2，为模式 2，校正时间较短。配置 0 为模式 0，校正时间最长。配置 1 为模式 1，需要同时配置 dac_gain_sel 来校正，校正时间较短。
	Dac_basal_out-band[5:0]	5:0	100000	100000	R/W	带外高斯滤波器的输出中间值，默认 6'd32
81	DIG2_REG6_3	7:0	0xA0	0xA0	R/W	PLL 小数字控制寄存器 6_3
	Clk_en	7	1	1	R/W	锁相环 DELTA-SIGMA 调制器时钟使能设置，默认 1
	Div2_en	6	0	0	R/W	锁相环 DELTA-SIGMA 调制器除 2 模块使能设置，默认 0
	Dac_basal_in-band[5:0]	5:0	100000	100000	R/W	带内高斯滤波器的输出中间值，默认 6'd32
82	DIG2_REG6_4	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 6_4
	Ds_shift	7	0	0	R/W	锁相环 DELTA-SIGMA 调制器移位使能设置，默认 0
	Inv_clk_en	6	0	0	R/W	锁相环 DELTA-SIGMA 调制器反向工作模式使能设置，默认 0
	Mash2_mode	5	0	0	R/W	锁相环 DELTA-SIGMA 调制器 mash2 模式使能设置，默认 0
	Outband_delay[4:0]	4:0	00000	00000	R/W	带外发射延时选择，默认 0
83	DIG2_REG7	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 7
	frequency_correc-tion_tx[7:0]	7:0	0x00	0x00	R/W	tx 模式分频值小数部分修正值
84	DIG2_REG7_2	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 7_2
	frequency_correc-tion_tx[15:8]	7:0	0x00	0x00	R/W	tx 模式分频值小数部分修正值
85	DIG2_REG7_3	7:0	0x40	0x40	R/W	PLL 小数字控制寄存器 7_3
	Reserved	7	0	0		
	fsynsdtxin_pol	6	1	1	R/W	调制输出送到 SDM 时输出样式选择，0 为晶振时钟下降沿输出，1 为晶振时钟上升沿输出
	vco_ready_sel[1:0]	5:4	00	00	R/W	用来配置只读寄存器 vco_ready 的等待时间
	fsyncaldiven	3	0	0	R/W	caldiven 手动使能，配 1 为使能
	sd_ndiv_ovrd_sel	2	0	0	R/W	给模拟的分频值手动改写选择，配 1 时手动改写
	frequency_correc-tion_tx[17:16]	1:0	00	00	R/W	tx 模式分频值小数部分修正值
86	DIG2_REG7_4	7:0	0x80	0x80	R/W	PLL 小数字控制寄存器 7_4
	vco_dly_sel[1:0]	7:6	10	10	R/W	vco 校正 trx 开始充电时间选择，00 for 3us，01 for 6us，10 for 9us， 11 for 12us
	sd_ndiv_ovrd[5:0]	5:0	000000	000000	R/W	给模拟的分频值手动改写值，第三种配分频值的方式，只能配部分整数频点
87	DIG2_REG8	7:0	0x1F	0x0a	R/W	PLL 小数字控制寄存器 8
	PLL_PFD_BWUN[1:0]	7:6	00	00	R/W	带宽校准时，小数字可直接控制 PFD 的输出信号 该寄存器仅在 EN_PLL_BWCAL=1 时有效，PLL_PFD_BWUN<1:0>=00,电

						荷泵不充电也不放电，PLL_PFD_BWUN<1:0>=01，电荷泵放电，PLL_PFD_BWUN<1:0>=10，电荷泵充电，PLL_PFD_BWUN<1:0>=11，电荷泵同时充电放电
	PLL_CP_NISEL[5:0]	5:0	011111	001010	R/W	电荷泵 NMOS 电流调节 000000:10uA, 111111:88.75uA，步进 1.25uA
88	DIG2_REG8_2	7:0	0x5F	0x4a	R/W	PLL 小数字控制寄存器 8_2
	fbrstn_dly_sel[1:0]	7:6	01	01	R/W	fbdiv_rst_n 延时选择 00 ： fbdiv_rst_n 延时为 2us 01 ： fbdiv_rst_n 延时为 4us 10 ： fbdiv_rst_n 延时为 6us 11 ： fbdiv_rst_n 延时为 10us
	PLL_CP_PISEL[5:0]	5:0	011111	001010	R/W	电荷泵 PMOS 电流调节 000000:10uA, 111111:88.75uA，步进 1.25uA
89	DIG2_REG8_3	7:0	0x00	0x02	R/W	PLL 小数字控制寄存器 8_3
	Reserved	7:5				
	EN_PLL_BWCAL	4	0	0	R/W	带宽校准使能控制：1，打开；0，关掉
	PLL_CP_SHIFT_IP_SEL[3:0]	3:0	0000	0010	R/W	电荷泵 PMOS 电流失配补偿 0000 不补偿失配 0001 补偿 1/40 的 PMOS 电流 1111 补偿(1/40+1/20+1/10+1/5)的 PMOS 电流
9D	DIG2_REG9	7:0	0x00	0x00	R/W	PLL 小数字控制寄存器 9
	Int_mode_en	7	0	0	R/W	整数模式选择，0：小数模式，1：整数模式
	Gauss_ctrl[1:0]	6:5	00	00	R/W	发射方式选择，11：开环带外，10：带内，01：闭环带外，00 两点式
	fsynfbdiven	4	0	0	R/W	fbdiven 使能，配 1 为使能
	fsynvcoen	3	0	0	R/W	vcoen 使能，配 1 为使能
	fsynfbdiv_rst_n_sel	2	0	0	R/W	fbdiv_rst_n 手动改写选择，配 1 为改写
	fsynfbdiv_rst_n_ovrd	1	0	0	R/W	fbdiv_rst_n 手动改写值
	fsynlfdacen	0	0	0	R/W	vco 开环使能，配 1 为使能
9E	DIG2_RO0	7:0	0x88	0x88	R/W	PLL 小数字只读寄存器 0
	Grp1ftune[3:0]	7:4	1000	1000	RO	VCO 多次校正 TX code 1
	Grp0ftune[3:0]	3:0	1000	1000	RO	VCO 多次校正 TX code 0
9F	DIG2_RO0_2	7:0	0x88	0x88	R/W	PLL 小数字只读寄存器 0_2
	Grp3ftune[3:0]	7:4	1000	1000	RO	VCO 多次校正 TX code 3
	Grp2ftune[3:0]	3:0	1000	1000	RO	VCO 多次校正 TX code 2
A0	DIG2_RO0_3	7:0	0x08	0x08	R/W	PLL 小数字只读寄存器 0_3
	Reserved	7:4	0000	0000	RO	
	Grp4ftune[3:0]	3:0	1000	1000	RO	VCO 多次校正 TX code 4
A1	DIG2_RO1	7:0	0x00	0x00	R/W	PLL 小数字只读寄存器 1
	vcoat_ncnt[7:0]	7:0	0x00	0x00	RO	Vco 校正对 vco 分频时钟计数个数[7:0]
A2	DIG2_RO1_2	7:0	0x00	0x00	R/W	PLL 小数字只读寄存器 1_2
	Reserved	7	0	0		

	vcoat_caldone	6	0	0	RO	Vco 单次校正结束标志
	vco_ready	5	0	0	RO	vco 校正或两点式校正时，先打开 pll，此时会把数字送给模拟的 caldiven、lfdacen、EN_TX_SYN 拉高，但此时 pll 模拟部分并没有准备好，需要等待至少十几 us 才能开始校正。数字内部从 lfdacen 拉高开始有一个计数器开始计数，等待可配置的一段时间后，把只读寄存器 vco_ready 拉高，软件读到 vco_ready 为高，即认为模拟电路已准备好，配置 calstart 信号，真正开始校正
	vcoat_mchcaldone	4	0	0	RO	VCO 校正结束标志，1：结束，0：未结束
	vcoat_ncnt[11:8]	3:0	0x000	0x000	RO	VCO 校正对 VCO 分频时钟计数个数[11:8]
A5	DIG2_RO3	7:0	0x30	0x30	R/W	PLL 小数字只读寄存器 3
	Reserved	7:6	00	00	RO	
	Two_point_cal_done	5	1	1	RO	两点式校正结束标志
	Two_point_code_reg[4:0]	4:0	10000	10000	RO	两点式校正 code 值
A6	DIG2_RO4	7:0	0x00	0x00	R/W	PLL 小数字只读寄存器 4
	Reserved	7:5	000	000		
	vcoat_calerr[4:0]	4:0	00000	00000	RO	Vco 校正过程误差值
A8	DIG2_RO6	7:0	0x08	0x08	R/W	PLL 小数字只读寄存器 6
	Reserved	7:4	0000	0000		
	PLL_VCO_CHSEL_RO[3:0]	3:0	1000	1000	RO	VCO 校正结果只读寄存器 0000：VCO 频率最低，1111：VCO 频率最高
A9	FIR_CAL_RO	7:0	0x00	0x00	R/W	中频滤波器校正只读寄存器
	scounter	7:0	0x00	0x00	RO	校准结束后输出的校准时钟周期的 count 数，最终的校准频率 IF=2M*scounter/96
AA	FIR_CAL_RO_1	7:0	0x20	0x20	R/W	中频滤波器校正只读寄存器
	if_cal_done	7	0	0	RO	中频滤波器校正结束指示位，1：结束，0：未结束
	Reserved	6	0	0		
	RCCAL_IN	5:0	100000	100000	RO	中频滤波器校正结果，只读寄存器
AB	FIR_CAL_M	7:0	0x00	0x01	R/W	中频滤波器校正手动控制寄存器
	Reserved	7:1	0000000	0000000		
	IF_CAL_EN_M	0	0	1	WO	软件触发，自动清零，软件写该 bit 为 1 即可触发中频滤波器校正
AC	DIG2_RO7	7:0	0x88	0x88	R/W	PLL 小数字只读寄存器 7
	Grp1ftune_rx [3:0]	7:4	1000	1000	RO	VCO 多次校正 RX code 1
	Grp0ftune_rx [3:0]	3:0	1000	1000	RO	VCO 多次校正 RX code 0
AD	DIG2_RO7_2	7:0	0x88	0x88	R/W	PLL 小数字只读寄存器 7_2
	Grp3ftune_rx [3:0]	7:4	1000	1000	RO	VCO 多次校正 RX code 3
	Grp2ftune_rx [3:0]	3:0	1000	1000	RO	VCO 多次校正 RX code 2
AE	DIG2_RO7_3	7:0	0x08	0x08	R/W	PLL 小数字只读寄存器 7_3

	Reserved	7:4	0000	0000	RO	
	Grp4ftune_rx[3:0]	3:0	1000	1000	RO	VCO 多次校正 RX code 4
B1	BLE_WHIT_TEST	7:0	0x33	0x33	R/W	BLE 加扰测试寄存器
	BLE_WHIT_INI_TEST	7	0	0	R/W	BLE 扰码初始相位测试模式使能 1 ： 测试模式 0 ： 自动模式
	BLE_WHIT_INI_REG[6:0]	6:0	011_0011	011_0011	R/W	测试模式下 BLE 扰码初始相位配置，当 BLE_ WHIT_INI_TEST 等于 1 时生效
B2	BLE_RX_CTRL	7:0	0x09	0x09	R/W	BLE RX 控制寄存器
	LEN_MATCH_MODE[1:0]	7:6	00	00	R/W	BLE RX 模式基于 Header.Length 的过滤机制 00：不过滤 length 01：只有收到的 length 等于 rx_pw_p0 才继续收包 10：只有收到的 length 大于 rx_pw_p0 才继续收包 11：只有收到的 length 小于 rx_pw_p0 才继续收包
	PLD_START_BYTES[5:0]	5:0	001001	001001	R/W	BLE RX 模式基于 BLE.payload 的过滤机制， 配置白名单过滤的起始字节，广播包 payload 范围 9~39，前 8 个 byte 为 Header(2)+ AdvA(6)，与寄存器 WL_MATCH_MODE 以及 WL_ADDR 一起使用 配置为 1~2 ： 从 Header 开始过滤 配置为 3~8 ： 从 AdvA 开始过滤 配置为 9~39 ： 从 payload 开始过滤