AC690X 用户手册

珠海市杰理科技有限公司
Zhuhai Jieli Technologyco.,LTD

2016 年 10 月

修改记录

版本	更新日期	描述
V1.0	2016/10/21	初始版本
V1.1	2016/10/27	添加第 12 章液晶显示控制器 LCDC
V1.2	2016/12/9	更改第一章总体介绍，蓝牙部分 1.支持蓝牙协议 V4.2 2.支持蓝牙双模，蓝牙经典+EDR+蓝牙低功耗

目 录

第 1 章 总体介绍

CPU 内核：

- 精简指令集 CPU
- 支持 32 位 DSP 指令
- 最高工作频率 160MHz
- 64 个向量中断
- 4 个优先级的中断系统

数字 I/O：

- 最多 49 个通用 IO
- 每个 IO 可独立设置为输入或输出
- 每个 IO 可独立设置为上拉或下拉
- CMOS/TTL 电平施密特输入
- 所有 IO 支持外部中断或唤醒

数字模块：

- 4 个多功能 16 位计数器，支持捕获和 PWM 功能
- 一个 16 位主动并行端口
- 一个全双工基本型 UART 异步串行口
- 两个全双工增强型 UART 异步串行口
- 两个支持主从机模式的 SPI 同步串行口
- 两个 SDIO 控制器
- 一个音频设备接口，支持 IIS，左对齐，右对齐和 DSP 模式
- 一个全速 USB 2.0 OTG 控制器
- 看门狗

模拟模块：

- 三个晶体振荡器
- 全速 USB 2.0 PHY
- 160MHz PLL 时钟合成器
- 16 位立体声音频 DAC，SNR>93dB
- 4 通道 16 位音频 ADC，SNR>90dB，带有一路 mic 前置放大器，SNR>72dB
- 内置的耳机功率放大器
- 3 路立体声音频选择器
- 16 通道 10 位通用 ADC
- 2 通道 4 级低电压检测器
- 内置电容触摸按键控制器
- 上电复位
- 两个低压差稳压器：
  5V 至 1.8V，5V 至 3.3V

蓝牙：

- 全内置射频，调制解调和基带
- 支持蓝牙协议版本 V4.2
- 支持蓝牙双模，蓝牙经典+EDR+蓝牙低功耗
- 支持蓝牙网络
- 符合 class2 和 class3 发射功率要求
- 支持+2dB 发射功率
- 接收敏度-85dB

FM 接收机：

- 支持世界频段 87.5-108MHz
- 立体声解码器
- 自动电台搜索
- 数字自动增益控制
- 数字自适应噪声衰减
- 可配置去加重特性（50/75us）
- 接受信号场强指示

工作范围：

- LDOIN 为 3.3 ~5.0V，VDDIO 为 3~3.6V

温度：

- 工作温度：-40℃~85℃
- 保存温度：-65℃~150℃

第 2 章 输入/输出（I/O）

2.1 概述

PAP/SD/SPI/IIC/UART 等外设的端口可支持 remapping，即可配置为使用不同的 IO 端口。IOMC0、IOMC1、IOMC2 和 IOMC3 寄存器用于设置此类的 IO remapping。

2.1.1 各引脚功能表

IO mapping 如下表

NO.							
1	PA0	SEG0					
2	PA1	SEG1	AMUX0L	Touch8	PWM0	UART1TXC	
3	PA2	SEG2	AMUX0R	Touch9	CAP3	UART1RXC	
4	PA3(下拉)	SEG3	AMUX1L	Touch10	ADC0	UART2TXA	
5	PA4	SEG4	AMUX1R	Touch11	ADC1	UART2RXA	
6	PA5	SEG5	UART0TXA	Touch12	ADC2	IIC_SCL_D	
7	PA6	SEG6	UART0RXA	Touch13	ADC3	IIC_SDA_D	
8	PA7	SEG7		Touch14		TMR0	PAPD0
9	PA8	SEG8	SD0DAT3A	Touch15			PAPD1
10	PA9	SEG9	SD0DAT2A	UART2TXB	ADC4	ALNK_SCLKA	PAPD2
11	PA10	SEG10	SD0DAT1A	UART2RXB	ADC5	ALNK_LRCKA	PAPD3
12	PA11	SEG11	SD0DAT0A			ALNK_DAT0A	
13	PA12	SEG12	SD0CDMA			ALNK_DAT1A	
14	PA13	SEG13	SD0CLKA			ALNK_DAT2A	
15	PA14	SEG14	TMR1	UART0TXD	PA_EN	ALNK_DAT3A	IIC_SCL_C
16	PA15	SEG15	CAP2	UART0RXD	LNA_EN	ALNK_MCLKA	IIC_SDA_C
17	PB0	CLKOUT0	UART1TXA	SPI2CLKA	SD1DAT0B	ALNK_SCLKB	Touch0
18	PB1	TMR2	UART1RXA	SPI2D0A	SD1CMDB	ALNK_LRCKB	Touch1
19	PB2			SPI2DIA	SD1CLKB	ALNK_DAT0B	Touch2
20	PB3	PWM2	spi0_DAT3AB(3)	SFC_DAT3AB(3)	SD1DAT1B	ALNK_DAT1B	Touch3
21	PB4(上拉)	PWM3	spi0_DAT3AB(2)	SFC_DAT2AB(2)	SD1DAT2B	ALNK_DAT2B	Touch4
22	PB5	SFC_DIB(1)	spi0_DIB(1)		SD1DAT3B	ALNK_DAT3B	
23	PB6(上拉)	SFC_CSB	spi0_CSB	UART0TXB		ALNK_MCLKB	
24	PB7	SFCDOB(0)	spi0_DOB(0)	UART0RXB	ADC6	TMR3	SD0DAT3B
25	PB8	SFC_CLKB	spi0_CLKB		ADC7		SD0DAT2B
26	PB9	UART2TXC	NFC_CRR_TEST		ADC8	CLKOUT1	SD0DAT1B
27	PB10	UART2RXC	NFC_AMP_TEST	Touch5	ADC9	SPI1DIA	SD0DAT0B

28	PB11	AMUX2L	NFCTX	Touch6	ADC10	SPI1CLKA	SD0CMDB
29	PB12	AMUX2R	NFCRX	Touch7	ADC11	SPI1D0A	SD0CLKB
30	PB13		MIC				
31	PC0	SEG16/COM5	SD1DAT3A	PAPD4	SPI2CLKB	UART1TXB	
32	PC1	SEG17/COM4	SD1DAT2A	PAPD5	SPI2D0B	UART1RXB	
33	PC2	SEG18/COM3	SD1DAT1A	PAPD6	SPI2DIB	UART0TXC	CAP1
34	PC3	SEG19/COM2	SD1DAT0A	PAPD7	SPI1DIB	UART0RXC	
35	PC4	SEG20/COM1	SD1CMDA	PAP_RD	SPI1CLKB	UART2TXD	IIC_SCL_B
36	PC5	SEG21/COM0	SD1CLKA	PAP_WR	SPI1D0B	UART2RXD	IIC_SDA_B
37	PD0			spi0_CLKA		SFC_CLKA	
38	PD1			spi0_D0A(0)		SFC_D0A(0)	
39	PD2			spi0_D1A(1)		SFC_D1A(1)	
40	PD3(上拉)			spi0_CSA		SFC_CSA	
41	OSCI			OSCI_32K(双脚振)			
42	PR0	长按 Reset0		OSCO_32K(双脚振)			
43	PR1(输出 0)	长按 Reset1			ADC12		
44	PR2	长按 Reset2			ADC13		
45	PR3	长按 Reset3			OSCO_12(双脚振)		
46	HOSCI				OSCI_12(双脚振)		
47	USBDP(下拉)	UART1TXD					IIC_SCL_A
48	UDBDM(下拉)	UART1RXD					IIC_SDA_A

2.2 寄存器说明

PAP/SD/SPI/IIC/UART 等外设的端口可支持 remapping，即可配置为使用不同的 IO 端口。IOMC0、IOMC1、IOMC2 和 IOMC3 寄存器用于设置此类的 IO remapping。

2.2.1 IO 功能配置寄存器

1) IOMC0: IO mapping control register0 (16bit addressing)

SD0IOS	PAPDEN	PAPREN	PAPWEN	ALNKIOS	IRFLTOS2	IRFLTOS1	IRFLTOS0
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

UT0IOS1	UT0IOS0	SPI0IOS	SD1IOS	SD1DTEN	SD1CKEN	SD0DTEN	SD0CKEN
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

SD0IOS: SD0 IO 选择

0:选择 PA12/PA13/{PA11，PA10，PA9，PA8}作为 SD0 的 CMD/CLK/DAT；

1:选择 PA2/PA3/{PA1，PA0，PA5，PA4}作为 SD0 的 CMD/CLK/DAT；

PAPDEN: PAP 接口数据信号引脚使能

0: PAx 引脚不被 PAP 占用；

1: 当 PAP 接口使能时，PAx 引脚作为 PAP 接口的数据信号；

PAPREN:PAP 接口读信号引脚使能

0: PB0 引脚不被 PAP 占用；

1: 当 PAP 接口使能时，PB0 引脚作为 PAP 接口的读使能信号；

PAPWEN: PAP 接口写信号引脚使能

0: PB4 引脚不被 PAP 占用；

1: 当 PAP 接口使能时，PB4 引脚作为 PAP 接口的写使能信号；

ALNKIOS: ALNK IO 选择

0: 选择 PA9/PA10/PA15/{PA11，PA12，PA13，PA14}作为 ALNK 的 SCLK/LRCK/MCLK/DAT；

1: 选择 PB0/PB1/PB6/{ PB2，PB3，PB4，PB5}作为 ALNK 的 SCLK/LRCK/MCLK/DAT；

IRFLTOS2-0: 与 IRFLT_EN 一起组成 TIMR 捕获端选择

0xx：IRFLT 不输出；

100：IRFLT 输出至 timer0 的捕获端；

101：IRFLT 输出至 timer1 的捕获端；

110：IRFLT 输出至 timer2 的捕获端；

111：IRFLT 输出至 timer3 的捕获端；

UT0IOS1-0: UART0 模块 IO re-mapping 设置

UT0IOS	TX	RX
0	PA5	PA6
1	PB6	PB7
2	PC2	PC3
3	PA14	PA15

SPI0IOS: SPI0 模块 IO 选择

0: 选择 PD0~PD2，PB3，PB4 作为 SPI0 的数据线；

1: 选择 PB3~PB8 作为 SPI0 的数据线；

SD1IOS: SD1 模块 IO 选择

0: 选择 PC4/PC5/{PC3，PC2，PC1，PC0}作为 SD1 的 CMD/CLK/DAT；

1: 选择 PB1/PB2/{PB0，PB3，PB4，PB5}作为 SD1 的 CMD/CLK/DAT；

SD1DTEN: 允许 SD1 CMD/DAT 占用 IO

0: 无论 SD1 有无使能，其 CMD/DAT 不占用相应 IO；

1: 当 SD1 使能时，其 CMD/DAT 占用相应 IO；

SD1CKEN: 允许 SD1 CLK 占用 IO

0: 无论 SD1 有无使能，其 CLK 不占用相应 IO；

1: 当 SD1 使能时，其 CLK 占用相应 IO；

SD0DTEN: 允许 SD0 CMD/DAT 占用 IO

0: 无论 SD0 有无使能，其 CMD/DAT 不占用相应 IO；

1: 当 SD0 使能时，其 CMD/DAT 占用相应 IO；

SD0CKEN: 允许 SD0 CLK 占用 IO

0: 无论 SD0 有无使能，其 CLK 不占用相应 IO；

1: 当 SD0 使能时，其 CLK 占用相应 IO；

2) IOMC1: IO mapping control register1 (16bit addressing)

Reserved	Reserved	WLUIOS		IICIOS		Reserved	SPI2IOS
BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

UT2IOS		OUT_CH1S			OUT_CH0S		
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

Reserved	CAPEDS	SFCIOS	SPI1IOS	UT1IOS		SPI0 DIDO MIX	Reserved
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

WLUIOS: WLU 模块 IO re-mapping 设置

WLUIOS	TX	RX
0	PA3	PA4
1	PA9	PA10
2	PB9	PB10
3	PC4	PC5

IICIOS: IIC 模块 IO 选择

IICIOS	CLK	DAT
0	USBDP	USBDM
1	PC4	PC5
2	PA14	PA15
3	PA5	PA6

SFCIOS: SFC 模块 IO 选择

0: 选择 PA13~PA15 作为 SFC 的数据线；

1: 选择 PB5~PB7 作为 SFC 的数据线；

UT2IOS: UART2 模块 IO re-mapping 设置

UT2IOS	TX	RX
0	PA3	PA4
1	PA9	PA10
2	PB9	PB10
3	PC4	PC5

OUT_CH1S: 选择输出到 IO 的信号

000: 选择 UT0_TX；

001: 选择 UT1_TX；

010: 选择 TMR2_PWM_OUT；

011: 选择 TMR3_PWM_OUT；

100: 选择 IRTCX32K；

101: 选择 OSC_CLK；

110: 选择 WLU_TX；

111: 选择 PWM4_OUT；

OUT_CH0S: 选择输出到 IO 的信号

000: 选择 UT0_TX；

001: 选择 UT1_TX；

010: 选择 TMR0_PWM_OUT；

011: 选择 TMR1_PWM_OUT；

100: 选择 RTC_OSCH；

101: 选择 BTOSC_CLK；

110: 选择 PLL_12M；

111: 选择 UT2_TX；

CAPEDS: CAP_MUX_EDGE_SEL，捕获端边沿选择

0: 上升沿；

1: 下降沿；

SFCIOS: SFC 模块 IO 选择

0: 选择 PD0~PD2，PB3 作为 SFC 的数据线；

1: 选择 PB3~PB8 作为 SFC 的数据线；

SPI1IOS: SPI1 模块 IO 选择

0: 选择 PB10~PB12 作为 SPI1 的数据线；

1: 选择 PC2~PC5 作为 SPI1 的数据线；

UT1IOS: UART1 模块 IO re-mapping 设置

UT1IOS	TX	RX
0	PB0	PB1
1	PC0	PC1
2	PA1	PA2
3	USBP	USBM

SPI0 DIDO MIX: spi0 输入结果产生选项

0: spi0 内部输入为 di；

1: spi0 内部输入为 di & do；

3) IOMC2: IO mapping control register2 (32bit addressing)

Reserved	Reserved	UTS					
BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

Reserved	Reserved	CAPS					
BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

Reserved	Reserved	IRFLTS					
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

Reserved	Reserved	WKUPS					
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

UTS: 串口功能脚选择

0x00: PA0；

0x01: PA1；

……

0x36: PD6；

0x37: PD7；

0x3e: USBDP；

0x3f: USBDM；

CAPS: CAPTURE 功能脚选择

0x00: PA0；

0x01: PA1；

……

0x36: PD6；

0x37: PD7；

0x3e: USBDP；

0x3f: USBDM；

IRFLTS: IRFLT 功能脚选择

0x00: PA0；

0x01: PA1；

……

0x36: PD6；

0x37: PD7；

0x3e: USBDP；

0x3f: USBDM；

WAKUPS: 唤醒功能脚选择

0x00: PA0；

0x01: PA1；

……

0x36: PD6；

0x37: PD7；

0x3e: USBDP；

0x3f: USBDM；

4) IOMC3: IO mapping control register3 (16bit addressing)

Reserved	Reserved	Reserved	Reserved	UT2IOEN	UT2MXS		
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	1	0	0	0
UT1IOEN	UT1MXS			UT0IOEN	UT0MXS		
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
1	0	0	0	1	0	0	0

UT2IOEN: 允许 UART2 占用 IO
1: 占用相应 IO；
0: 不占用 IO，该 IO 用于其它功能；

UT2MXS: UART2 输入选择
0XX: 选择普通 IO 作为输入（UT0IOS）；
100: 选择由 IOMC2[5:0]选择的 IO 作为输入；
101: 选择由 IOMC2[13:8]选择的 IO 作为输入；
110: 选择由 IOMC2[21:16]选择的 IO 作为输入；
111: 选择由 IOMC2[29:24]选择的 IO 作为输入；

UT1IOEN: 允许 UART1 占用 IO
1: 占用相应 IO；
0: 不占用 IO，该 IO 用于其它功能；

UT1MXS: UART1 输入选择
0XX: 选择普通 IO 作为输入（UT1IOS）；
100: 选择由 IOMC2[5:0]选择的 IO 作为输入；
101: 选择由 IOMC2[13:8]选择的 IO 作为输入；
110: 选择由 IOMC2[21:16]选择的 IO 作为输入；
111: 选择由 IOMC2[29:24]选择的 IO 作为输入；

UT0IOEN: 允许 UART0 占用 IO
1: 占用相应 IO；
0: 不占用 IO，该 IO 用于其它功能；

UT0MXS: UART0 输入选择
0XX: 选择普通 IO 作为输入（UT2IOS）；
100: 选择由 IOMC2[5:0]选择的 IO 作为输入；
101: 选择由 IOMC2[13:8]选择的 IO 作为输入；
110: 选择由 IOMC2[21:16]选择的 IO 作为输入；
111: 选择由 IOMC2[29:24]选择的 IO 作为输入；

第 3 章时钟系统（Clock_System）

3.1 概述

3.1.1 AC690X 具备如下 4 个原生时钟源：

1) rc_clk: 来自 RC 振荡器，振荡频率约 250KHz，随电压和温度不同会有较大变化。来自 RC 振荡器，振荡频率约 250KHz，随电压和温度不同会有较大变化。

2) rt_osl: 来自 PMU 内置 RTC 模块的振荡器，外部需在 PR0（OSCO）/ OSCI 间挂载 32.768KHz 晶振（谐振电容需外置）。

3) 3.rt_osh: 来自 PMU 内置 RTC 模块的振荡器，外部需在 PR3（HOSCO）/ HOSCI 间挂载 12-26MHz 晶振（谐振电容需外置）。

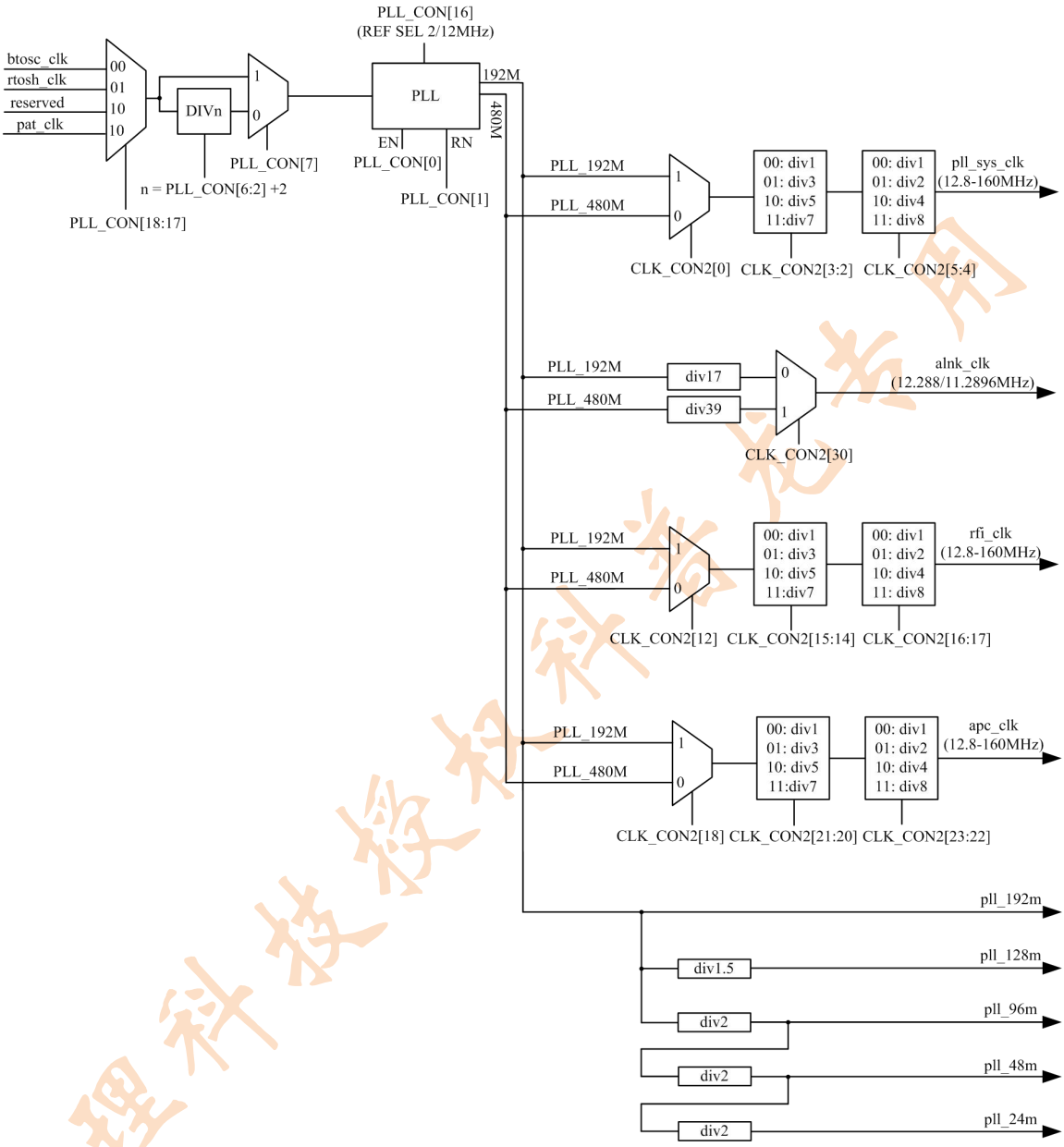
4) bt_osc: 来自蓝牙模块的振荡器，外部需在 BTXOSCO 和 BTXOSCI 引脚挂载 12-26MHz 晶振（内部已有谐振电容，也可外置）。

3.1.2 AC690X 的衍生时钟源

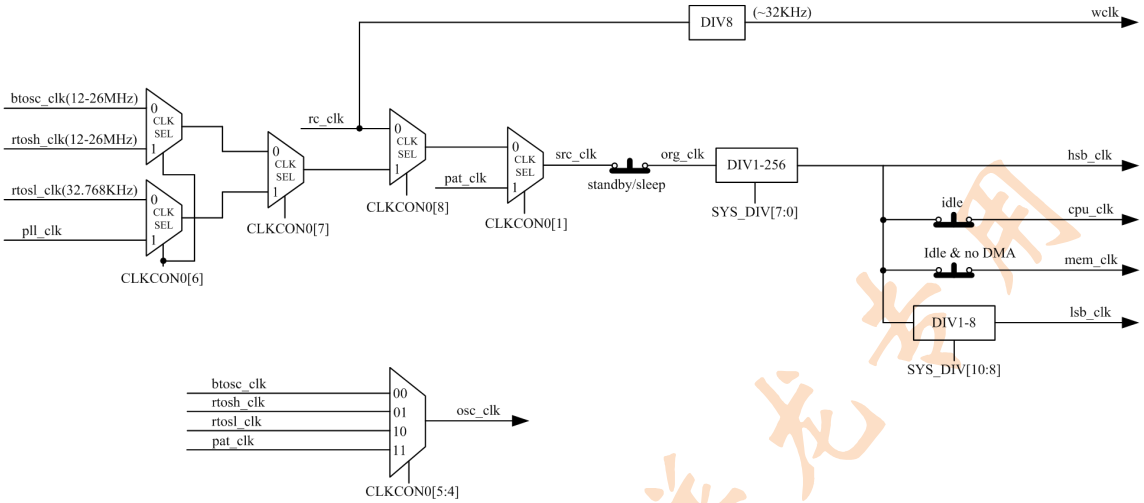
1) pll_clk: 来自片内 SYS_PLL 的输出，正常状态下同时输出 480MHz 和 192MHz 的信号。

3.2 结构框图

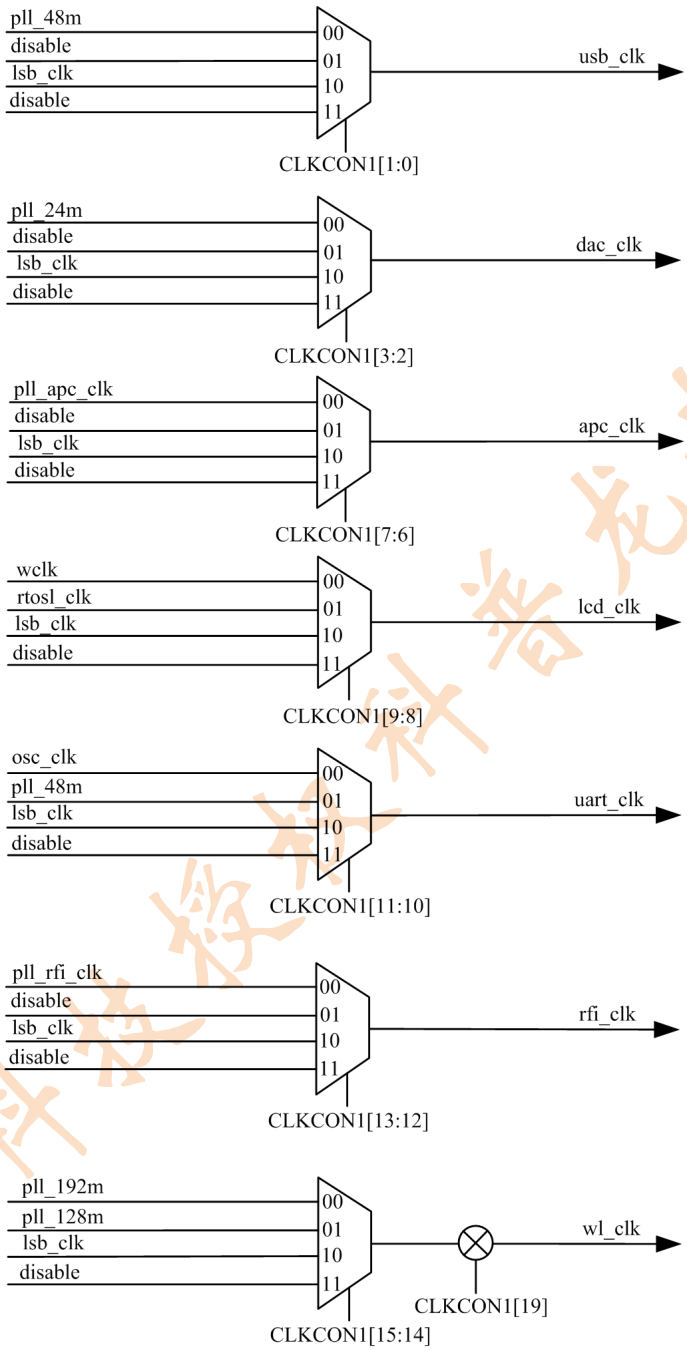
1）1.PLL 的输入参考时钟可由 btosc_clk, rtosh_clk 或 pat_clk 提供。PLL 部分电路框图如下。



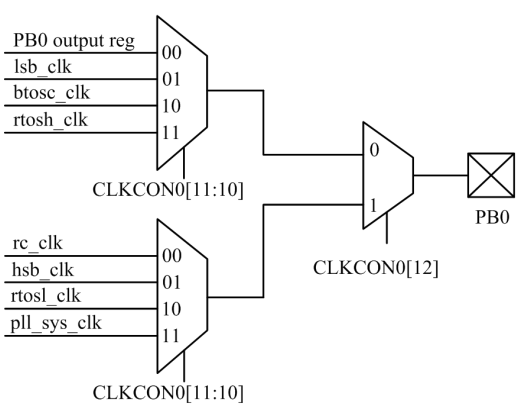
2）AC690X 系统可运行于上述 4 个原生时钟源或 1 个衍生时钟源，hsb_clk 最高运行频率为 160MHz，lsb_clk（低速总线时钟）最高运行频率为 80MHz，可随时更改 lsb_clk 的分频值，但需确保在任何时刻，lsb_clk 都不会超过其允许的最高运行频率。系统时钟部分电路框图如下。

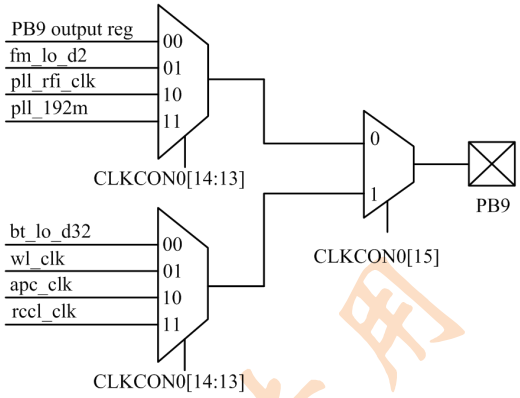


3）与系统异步的部分外设具有单独的时钟切换电路，该部分框图如下。



4）AC690X 的 PB0/PB9 引脚为复用测试引脚，该引脚除了普通的 PB0/PB9 的 IO 功能之外，还可以将内部的时钟信号驱动至片外，用于测试或特殊用途。





3.3 寄存器说明

1）CLK_CON0: clock control register 0

15	14	13	12	11	10	9	8
PB9_SEL			PB0_SEL			SFR_CKMD	CKSEL2
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
CKSEL1	CKSEL0	OSC_SEL		reserved	reserved	TSSEL	RC_EN
rw	rw	rw	rw	r	r	rw	rw
0	0	0	0	0	0	0	1

PB9_SEL: 外部参考时钟源选择

000: 选择 PB9 输出寄存器；

001: 选择 FM_LO_D2；

010: 选择 PLL_RFI_CLK；

011: 选择 PLL_192M；

100: 选择 BT_LO_D32；

101: 选择 WL_CLK；

110: 选择 APC_CLK；

111: 选择 RCCL_CLK；

PA4_SEL: 外部参考时钟源选择

000: 选择 PA4 输出寄存器；

001: 选择 LSB_CLK；

010: 选择 BTOSC_CLK；

011: 选择 RTOSH_CLK；

100: 选择 RC_CLK；

101: 选择 HSB_CLK；

110: 选择 RTOSL_CLK；

111: 选择 PLL_SYS_CLK；

SFR_CKMD: SFR 时钟模式选择

0: 使用 CPU 时钟作为 SFR 时钟；

1: 不写 SFR 的时候，SFR 时钟自动关闭；

CKSEL2-0: 主时钟切换，详见第一张第二小节框图；

OSC_SEL:片内 OSC_CLK 来源选择

0: 选择 BTOSC_CLK 作为其时钟来源；

1: 选择 RTOSH_CLK 作为其时钟来源；

2: 选择 RTOSL_CLK 作为其时钟来源；

3: 选择 TEST_MODE input clock 作为其时钟来源；

TSSEL 系统时钟源测试选择

0: 选择正常时钟作为系统时钟源；

1: 选择 TEST_MODE input clock 作为系统时钟源；

RC_EN: 250KHz 片内 RC 振荡器使能

0: 关闭片内 RC 振荡器；

1: 打开片内 RC 振荡器；

2）**CLK_CON1: clock control register 1**

31	30	29	28	27	26	25	24
reserved	reserved	SFC_DSEL		reserved	reserved	reserved	reserved
r	r	rw	rw	r	r	r	r
0	0	0	0	0	0	0	0

23	22	21	20	19	18	17	16
reserved	reserved	reserved	reserved	WL_CKINV	ROM_PD	TMSEL	MEM_SCKE
r	r	r	r	r	rw	rw	rw
0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8
BT_CKSEL		RFI_CKSEL		UART_CKSEL		LCD_CKSEL	
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
APC_CKSEL		reserved	reserved	DAC_CKSEL		USB_CKSEL	
rw	rw	r	r	rw	rw	rw	rw
0	0	0	0	0	0	0	0

SFC_DSEL: SFC 延时时间选择

00: 延时最小；

01: ……；

10: ……；

11: 延时最大；

WL_CKINV： wireless 时钟反向使能

0: 正向；

1: 反向；

ROM_PD： ROM power down 使能

0: ROM 正常工作；

1: ROM 进入 power down 状态，节省功耗；

TMSEL： test mode 时钟选项

0: test mode 下选择 TEST_MODE input clock 作为系统时钟；

1: test mode 下强制不选择 TEST_MODE input clock 作为系统时钟；

MEM_SCKE： MEM_CLK 软件使能

0: IDLE 状态下且无 DMA 请求时，MEM_CLK 关闭；

1: MEM_CLK 一直打开；

BT_CKSEL: BT 模块时钟选择

00: 选择 PLL_WL_CLK 作为 BT 模块时钟；

01: disable；

10: 选择 LSB_CLK 作为 BT 模块时钟；

11: disable；

RFI_CKSEL: RFI 模块时钟选择

00: 选择 PLL_RFI_CLK 作为 RFI 模块时钟；

01: disable；

10: 选择 LSB_CLK 作为 RFI 模块时钟；

11: disable；

UART_CKSEL: UART 模块时钟选择

00: 选择 OSC_CLK 作为 UART 模块时钟；

01: 选择 PLL_48M 作为 UART 模块时钟；

10: 选择 LSB_CLK 作为 UART 模块时钟；

11: disable；

LCD_CKSEL: LCD 模块时钟选择

00: 选择 WCLK 作为 LCD 模块时钟；

01: 选择 RTOSL_CLK 作为 LCD 模块时钟；

10: 选择 LSB_CLK 作为 LCD 模块时钟；

11: disable；

APC_CKSEL: APC 模块时钟选择

00: 选择 PLL_APC_CLK 作为 APC 模块时钟；

01: disable；

10: 选择 LSB_CLK 作为 APC 模块时钟；

11: disable；

DAC_CKSEL: DAC 模块时钟选择

00: 选择 PLL_24M 作为 DAC 模块时钟；

01: disable；

10: 选择 LSB_CLK 作为 DAC 模块时钟；

11: disable；

USB_CKSEL: USB 模块时钟选择

00: 选择 PLL_48M 作为 USB 模块时钟；

01: disable；

10: 选择 LSB_CLK 作为 USB 模块时钟；

11: disable；

3）CLK_CON2: clock control register 2

31	30	29	28	27	26	25	24
reserved	PLL_ALNK_SEL	reserved	reserved	reserved	reserved	reserved	reserved
r	rw	r	r	r	r	r	r
0	0	0	0	0	0	0	0

23	22	21	20	19	18	17	16
PLL_APC_DIV1		PLL_APC_DIV0		PLL_APC_SEL		PLL_RFI_DIV1	
rw	rw	rw	rw	rw	rw	rw	rw
0	1	0	1	0	1	0	1

15	14	13	12	11	10	9	8
PLL_RFI_DIV0		PLL_RFI_SEL		reserved	reserved	reserved	reserved
rw	rw	rw	rw	r	r	r	r
0	1	0	1	0	0	0	0

7	6	5	4	3	2	1	0
reserved	reserved	PLL_SYS_DIV1		PLL_SYS_DIV0		PLL_SYS_SEL	
r	r	rw	rw	rw	rw	rw	rw
0	0	0	1	0	1	0	1

PLL_ALNK_SEL: ALNK 时钟选择

0: 选择 PLL_192M 分频至 11.2896MHz 的时钟；

1: 选择 PLL_480M 分频至 12.288MHz 的时钟；

PLL_APC_DIV1/0: APC_CLK 分频选择，请参看时钟框图

PLL_APC_SEL: APC_CLK 时钟源选择

00: 选择 PLL_192M；

01: 选择 PLL_480M；

10: disable；

11: disable；

PLL_RFI_DIV1/0: RFI_CLK 分频选择，请参看时钟框图

PLL_RFI_SEL: RFI_CLK 时钟源选择

00: 选择 PLL_192M；

01: 选择 PLL_480M；

10: disable；

11: disable；

PLL_FM_DIV1/0: FM_CLK 分频选择，请参看时钟框图

PLL_FM_SEL: FM_CLK 时钟源选择

00: 选择 PLL_192M；

01: 选择 PLL_480M；

10: disable；

11: disable；

PLL_SYS_DIV1/0: PLL_SYS_CLK 分频选择，请参看时钟框图

PLL_SYS_SEL: PLL_SYS_CLK 时钟源选择

00: 选择 PLL_192M；

01: 选择 PLL_480M；

10: disable；

11: disable；

4）PLL_CON: pll control register

23	22	21	20	19	18	17	16
reserved	reserved	reserved	PLL_TEST	reserved	PLL REF SEL		PLL_RSEL12
r	r	r	rw	r	rw	rw	rw
0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8
PLL_TSEL				PLL_DSMS	PLL_DIVS	PLL_RSEL	
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
PLL_REFDE	PLL_REFDS					PLL RST	PLL EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

PLL_TEST： PLL 测试使能，请设置为‘0’；

PLL REF SEL: PLL 模块参考时钟选择

00: 选择 BT_OSC；

01: 选择 RT_OSH；

10: 保留；

11: PAT_CLK；

PLL_RSEL12： PLL 模块鉴相器时钟选择

0: 2MHz 鉴相器时钟；

1: 12MHZ 鉴相器时钟；

PLL_TSEL： PLL 小数分频调制器设置

PLL_DSMS： PLL 小数分频调制器选择

0: 使用 CIFF 调制器；

1: 使用 MASH 调制器；

PLL_DIVS： PLL 分频器选择

0： 选择整数分频器，此时 PLL 工作于整数模式；

1： 选择小数分频器，此时 PLL 工作于小数模式；

PLL_RSEL： PLL 参考时钟选择

0: 选择 BT_OSC 差分时钟；

1: 选择 RT_OSL 差分时钟；

2: 选择 PLL REF SEL 指定的来源；

3: 保留；

PLL_REFDE： PLL 参考时钟分频使能

0: PLL 内参考时钟分频器关闭（DIV1）；

1: PLL 内参考时钟分频器打开（DIV2-33）；

PLL_REFDS： PLL 参考时钟分频值，R = PLL_REFDS + 2 （2-33）；

PLL RST： PLL 模块复位，需在 PLL EN 使能 10uS 之后才能释放；

0: 复位；

1: 释放；

PLL EN： PLL 模块使能

0: 关闭；

1: 打开；

5）PLL_INTF: pll fractional divider setting

PLL 小数分频器设置

[31:24]为分频器整数设置值，[23:0]为分频器小数设置值；

6）PLL_DMAX: pll ssc max setting

PLL 频谱扩展上限值，24bit

7）PLL_DMIN: pll ssc min setting

PLL 频谱扩展下限值，24bit

8）**PLL_DSTP: pll ssc step setting**

PLL 频谱扩展步进值，24bit

第 4 章 16 位定时器(TIMER 16)

4.1 概述

Timer16 是一个集合了定时/计数/捕获功能于一体的多动能 16 位定时器。它的驱动源可以选择片内时钟或片外信号。它带有一个可配置的最高达 64 的异步预分频器，用于扩展定时时间或片外信号的最高频率。它还具有上升沿/下降沿捕获功能，可以方便的对片外信号的高电平/低电平宽度进行测量。

4.2 控制寄存器

寄存器列表	TIMER0	TIMER1	TIMER2	TIMER2
Tx_CON	T0_CON	T1_CON	T2_CON	T3_CON
Tx_CNT	T0_CNT	T1_CNT	T2_CNT	T3_CNT
Tx_PRD	T0_PRD	T1_PRD	T2_PRD	T3_PRD
Tx_PWM	T0_PWM	T1_PWM	T2_PWM	T3_PWM

1) Tx_CON： timer x control register

PND	PCLR	Reserved	Reserved	Reserved	Reserved	PWM_INV	PWM_EN
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
r	w	r	r	r	r	rw	rw
0	0	0	0	0	0	0	0

PSET3	PSET2	PSET1	PSET0	SSEL1	SSEL0	MODE1	MODE0
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

PND： 中断请求标志，当 timer 溢出或产生捕获动作时会被硬件置 1，需要由软件清 0。

PCLR： 软件在此位写入‘1’将清除 PND 中断请求标志。

PWM_INV： PWM 信号输出反向。

PWM_EN： PWM 信号输出使能。此位置 1 后，相应 IO 口的功能将会被 PWM 信号输出替代。

PSET3-0： 预分频选择位

0000：预分频 1；

0001：预分频 4；

0010：预分频 16；

0011：预分频 64；

0100：预分频 1*2；

0101：预分频 4*2；

0110：预分频 16*2；

0111：预分频 64*2；

1000：预分频 1*256；

1001：预分频 4*256；

1010：预分频 16*256；

1011：预分频 64*256；

1100：预分频 1*2*256；

1101：预分频 4*2*256；

1110：预分频 16*2*256；

1111：预分频 64*2*256；

SSEL1-0： timer 驱动源选择

00：使用系统时钟作为 timer 的驱动源；

01：使用 IO 口信号作为 timer 的驱动源；

10：使用 OSC 时钟作为 timer 的驱动源；

11：使用 RC 时钟作为 timer 的驱动源；

MODE1-0：工作模式选择

00： timer 关闭；

01： 定时/计数模式；

10： IO 口上升沿捕获模式（当 IO 上升沿到来时，把 TxCNT 的值捕捉到 TxPR 中）；

11： IO 口下降沿捕获模式（当 IO 下降沿到来时，把 TxCNT 的值捕捉到 TxPR 中）；

2) **Tx_CNT：timer x counter register**

Tx_CNT							
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
1	1	0	0	1	1	0	1

Tx_CNT							
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
1	1	0	0	1	1	0	1

timer16 的计数寄存器

3) Tx_PR： timer x period register

Tx_PR							
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

Tx_PR							
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

timer16 的周期寄存器

在定时/计数模式下，当 Tx_CNT == Tx_PR 时，Tx_CNT 会被清 0。

在上升沿/下降沿捕获模式下，TxPR 是作为捕获寄存器使用的，当捕获发生时，Tx_CNT 的值会被复制到 Tx_PR 中。而此时 Tx_CNT 自由的由 0-65535-0 计数，不会和 Tx_PR 进行比较清 0。

4) **Tx_PWM： timer x PWM register**

Tx_PWM							
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
rw	rw	rw	rw	rw	rw	rw	rw
x	x	x	x	x	x	x	x

Tx_PWM							
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
x	x	x	x	x	x	x	x

Timer16 的 PWM 设置寄存器

在 PWM 模式下，此寄存器的值决定 PWM 输出的占空比。占空比 N 的计算公式如下：

$$N = (Tx\_PWM / Tx\_PR) * 100\%$$

此寄存器不带有缓冲，写此寄存器的动作将可能导致不同步状态产生的 PWM 波形占空比瞬间过大或过小的问题。

第 5 章 串行通信（UART）

5.1 概述

UART0、UART1 支持接收带循环 Buffer 的 DMA 模式和普通模式

UART2 只支持普通模式，不支持 DMA 模式。

UART0 和 UART1 在 DMA 接收的时候有一个循环 Buffer，UTx_RXSADR 表示它的起始，UTx_RXEADR 表示它的结束。同时，在接收过程中，会有一个超时计数器（UTx_OTCNT），如果在指定的时间里没有收到任何数据，则超时中断就会产生。超时计数器是在收到数据的同时自动清空。

5.2 控制寄存器

寄存器列表	UART0	UART1	UART2
UTx_CON	UT0_CON	UT1_CON	UT2_CON
UTx_BAUD	UT0_BAUD	UT1_BAUD	UT2_BAUD
UTx_BUF	UT0_BUF	UT1_BUF	UT2_BUF
UTx_TXADR	UT0_TXADR	UT1_TXADR	
UTx_TXCNT	UT0_TXCNT	UT1_TXCNT	
UTx_RXADR			
UTx_RXCNT	UT0_RXCNT	UT1_RXCNT	
UTx_RXSADR	UT0_RXSADR	UT1_RXSADR	
UTx_RXEADR	UT0_RXEADR	UT1_RXEADR	
UTx_HRXCNT	UT0_HRXCNT	UT1_HRXCNT	
UTx_OTCNT	UT0_OTCNT	UT1_OTCNT	

1) UTx_CON: uart x control register (16bit addressing).

TPND	RPND	CLRTPND	CLRRPND	OTPND	CLR_OTPND	TB8	RB8
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
r	r	r	r	r	r	rw	r
1	0	0	0	0	0	x	x

RDC	RX_MODE	OT_IE	DIVS	RXIE	TXIE	M9EN	UTEN
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
w	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

TPND: TX Pending

RPND: RX Pending& Dma_Wr_Buf_Empty，数据接收不完 Pending 不会为 1

CLRTPND： 清空 TX Pending

CLRRPND： 清空 RX Pending

OTPND : OverTime Pending

CLR_OTPND: 清空 OTPND

TB8: 9Bit 模式时，TX 发送的第 9 位

RB8: 9Bit 模式时。RX 接收到的第 9 位

RXMODE（*）： 读模式选择

0: 普通模式，不用 DMA；

1: DMA 模式；

RDC（*）： 写 1 时，将已经收到的数目写到 UTx_HRXCNT，已收到的数目清零

写 0 无效

OTIE（*）： OT 中断允许

0： 不允许；

1: 允许；

（*）：只有 UART0 和 UART1 支持

DIVS: 前 3 分频选择，0 为 4 分频，1 为 3 分频

RXIE: RX 中断允许

当 RX Pending 为 1，而且 RX 中断允许为 1，则会产生中断

TXIE: TX 中断允许

当 TX Pending 为 1，而且 TX 中断允许为 1，则会产生中断

M9EN: 9bit 模式使能

UTEN: UART 模块使能

2) UTx_BAUD: uart x baudrate register (16bit addressing, Write Only).

UTx_BAUD							
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

uart 的波特率寄存器

当 DIVS=0 时，

Baudrate = Freq_sys / ((UTx_BAUD+1) *4)

当 DIVS=1 时，

Baudrate = Freq_sys / ((UTx_BAUD+1) *3)

(Freq_sys 是 apb_clk,指慢速设备总线的时钟，非系统时钟)

3) UTx_BUF: uart x data buffer register (8bit addressing).

UT0BUF							
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
rw	rw	rw	rw	rw	rw	rw	rw
x	x	x	x	x	x	x	x

uart 的收发数据寄存器

写 UTx_BUF 可启动一次发送；

读 UTx_BUF 可获得已接收到的数据。

4) UTx_TXADR：uart x TX DMA address(17bit addressing, Write Only).

UTx_TXADR							
							BIT16
							x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

DMA 发送数据的起始地址

5) UTx_TXCNT：uart x TX DMA count (16bit addressing, Write Only).

UTx_TXCNT							
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

写 UTx_TXCNT，控制器产生一次 DMA 的操作，同时清空中断，当 uart 需要发送的数据达到 UTx_TXCNT+1 的值，控制器会停止发送数据的操作，同时产生中断（UTx_CON[15]）。

6) UTx_RXCNT：uart x receive DMA count(32bit addressing, Write Only).

UTx_RXCNT							
BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24
x	x	x	x	x	x	x	x
BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
x	x	x	x	x	x	x	x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

写 UTx_RXCNT，控制器产生一次 DMA 的操作，同时清空中断，当 uart 需要接收的数据达到 UTx_RXCNT 的值，控制器会停止接收数据的操作，同时产生中断（UTx_CON[14]）。

7) UTx_RXSADR：uart x receive DMA address(17bit addressing, Write Only).

UTx_RXSADR							
							BIT16
							x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

DMA 接收数据时，循环 buffer 的起始地址

8) UTx_RXEADR：uart x receive DMA end address(17bit addressing, Write Only).

UTx_RXEADR							
							BIT16
							x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

DMA 接收数据时，循环 buffer 的结束地址

9) UTx_HRXCNT：uart x have receive DMA count(32bit addressing, Read Only).

UTx_HRXCNT							
BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24
x	x	x	x	x	x	x	x
BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
x	x	x	x	x	x	x	x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

当设这 RDC（UTx_CON[7]）=1 时，串口设备会将当前总共收到的字节数记录到 UTx_HRXCNT 里。

10) UTx_OTCNT：uart x OverTime count(32bit addressing, Write Only).

UTx_OTCNT							
BIT31	BIT30	BIT29	BIT28	BIT27	BIT26	BIT25	BIT24
x	x	x	x	x	x	x	x
BIT23	BIT22	BIT21	BIT20	BIT19	BIT18	BIT17	BIT16
x	x	x	x	x	x	x	x
BIT15	BIT14	BIT13	BIT12	BIT11	BIT10	BIT9	BIT8
x	x	x	x	x	x	x	x
BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
x	x	x	x	x	x	x	x

设置串口设备在等待多久 RX 下降沿的时间，如果在所设置的时间里没收到 RX 的下降沿，则产生 OT 中断（OT_PND）

Time(ot)= Time(uart_clk) * UTx_OTCNT；

例如：波特率时间为 100ns，UTx_OTCNT = 10,那么，OT 的时间就为 1000ns

第 6 章串行外设接口（SPI0&SPI1）

6.1 概述

SPI 接口是一个标准的遵守 SPI 协议的串行通讯接口，在上面传输的数据以 Byte（8bit）为最小单位，且永远是 MSB 在前。

SPI 接口支持主机和从机两种模式

主机：SPI 接口时钟由本机产生，提供给片外 SPI 设备使用

从机：SPI 接口时钟由片外 SPI 设备产生，提供给本机使用

工作于主机模式时，SPI 接口的驱动时钟可配置，范围为 系统时钟~系统时钟/256

工作于从机模式时，SPI 接口的驱动时钟频率无特殊要求，但数据速率需要进行限制，否则易出现接收缓冲覆盖错误。

SPI 接口可独立地选择在 SPI 时钟的上升沿或下降沿更新数据，在 SPI 时钟的上升沿或下降沿采样数据。

SPI 接口支持单向（Unidirection）和双向（Bidirection）模式

单向模式：使用 SPICK 和 SPIDAT 两组连线，其中 SPIDAT 为双向信号线，同一时刻数据只能单方向传输。

双向模式：使用 SPICK，SPIDI 和 SPIDO 三组连线，同一时刻数据双向传输。但 DMA 不支持双向数据传输，当在本模式下使能 DMA 时，也只有一个方向的数据能通过 DMA 和系统进行传输。

SPI 单向模式支持 1bit data、2bit data 和 4bit data 模式，即：

1bit data 模式：串行数据通过一根 DAT 线传输，一个字节数据需 8 个 SPI 时钟。

2bit data 模式：串行数据通过两根 DAT 线传输，一个字节数据需 4 个 SPI 时钟。

4bit data 模式：串行数据通过四根 DAT 线传输，一个字节数据需 2 个 SPI 时钟。

（注：SPI0 支持 1bit，2bit 和 4bit 模式，SPI1 只支持 1bit 模式）

SPI 双向模式只支持 1bit data 模式，即：

1bit data 模式：串行数据通过一根 DAT 线传输，一个字节数据需 8 个 SPI 时钟。

SPI 接口在发送方向上为单缓冲，在上一次传输未完成之前，不可开始下一次传输。在接收方向上为双缓冲，如果在下一次传输完成时 CPU 还未取走本次的接收数据，那么本次的接收数据将会丢失。

SPI 接口的发送寄存器和接收寄存器在物理上是分开的，但在逻辑上它们一起称为 SPIBUF 寄存器，使用相同的 SFR 地址。当写这个 SFR 地址时，写入至发送寄存器。当读这个 SFR 地址时，从接收寄存器读出。

SPI 传输支持由 CPU 直接驱动，写 SPIBUF 的动作将启动一次 Byte 传输。

SPI 传输也支持 DMA 操作，但 DMA 操作永远是单方向的，即一次 DMA 要么是发送一包数据，要么是接收一包数据，不能同时发送并且接收一包数据，即使在双向模式下也是这样。每次 DMA 操作支持的数据量为 1-65535Byte。写 SPIADR 的动作将启动一次 DMA 传输。

6.2 控制寄存器

寄存器列表	SPI0	SPI1	SPI2
SPIx_CON	SPI0_CON	SPI1_CON	SPI2_CON
SPIx_BUF	SPI0_BUF	SPI1_BUF	SPI2_BUF
SPIx_BAUD	SPI0_BAUD	SPI1_BAUD	SPI2_BAUD
SPIx_ADR	SPI0_ADR	SPI1_ADR	SPI2_ADR
SPIx_CNT	SPI0_CNT	SPI1_CNT	SPI2_CNT

1) SPIx_CON: SPIx control register (16bit addressing)

15	14	13	12	11	10	9	8
PND	PCLR	IE	DIR	DATW		reserved	reserved
r	w	rw	rw	rw	rw	r	r
1	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
CSID	CKID	UE	SE	BIDIR	CSE	SLAVE	SPIE
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

PND: 中断请求标志，当 1Byte 传输完成或 DMA 传输完成时会被硬件置 1。

有 3 种方法清除此标志

1， 向 PCLR 写入‘1’；

2， 写 SPIBUF 寄存器来启动一次传输；

3， 写 SPICNT 寄存器来启动一次 DMA；

PCLR: 软件在此位写入‘1’将清除 PND 中断请求标志。

IE: SPI 中断使能

0: 禁止 SPI 中断；

1: 允许中断允许；

DIR: 在单向模式或 DMA 操作时设置传输的方向

0: 发送数据；

1: 接收数据；

DATW: SPI 数据宽度设置

00: 1bit 数据宽度；

01: 2bit 数据宽度；

10: 4bit 数据宽度；

11：NA，不可设置为此项；

（注：SPI0 支持 1bit，2bit 和 4bit 模式，SPI1 /SPI2 只支持 1bit 模式）

CSID: SPICS 信号极性选择

0: SPICS 空闲时为 0 电平；

1: SPICS 空闲时为 1 电平；

CKID: SPICK 信号极性选择

0: SPICK 空闲时为 0 电平；

1: SPICK 空闲时为 1 电平；

UE: 更新数据边沿选择

0: 在 SPICK 的上升沿更新数据；

1: 在 SPICK 的下降沿更新数据；

SE: 采样数据边沿选择

0: 在 SPICK 的上升沿采样数据；

1: 在 SPICK 的下降沿采样数据；

BIDIR: 单向/双向模式选择

0: 单向模式，数据单向传输，同一时刻只能发送或者接收数据。

数据传输方向因收发而改变，所以由硬件控制，不受写 IO 口 DIR 影响。

1: 双向模式，数据双向传输，同时收发数据，但 DMA 只支持一个方向的数据传输。

数据传输方向设置后不改变，所以由软件控制，通过写 IO 口 DIR 控制。

CSE: SPICS 信号使能

0: 不使用 SPICS 信号；

1: 使用 SPICS 信号；

SPIE: SPI 接口使能

0: 关闭 SPI 接口；

1: 打开 SPI 接口；

2) SPIx_BAUD: SPI baudrate setting register (8bit addressing, write only)

SPI 主机时钟设置寄存器

SPICK = system clock / (SPIBAUD + 1)

3) SPI_BUF: SPI buffer register (8bit addressing)

发送寄存器和接收寄存器共用此 SFR 地址。写入至发送寄存器，从接收寄存器读出。

4) SPI_ADR: SPI DMA start address register (32bit addressing, write only)

SPI DMA 起始地址寄存器，只写，读出为不确定值。

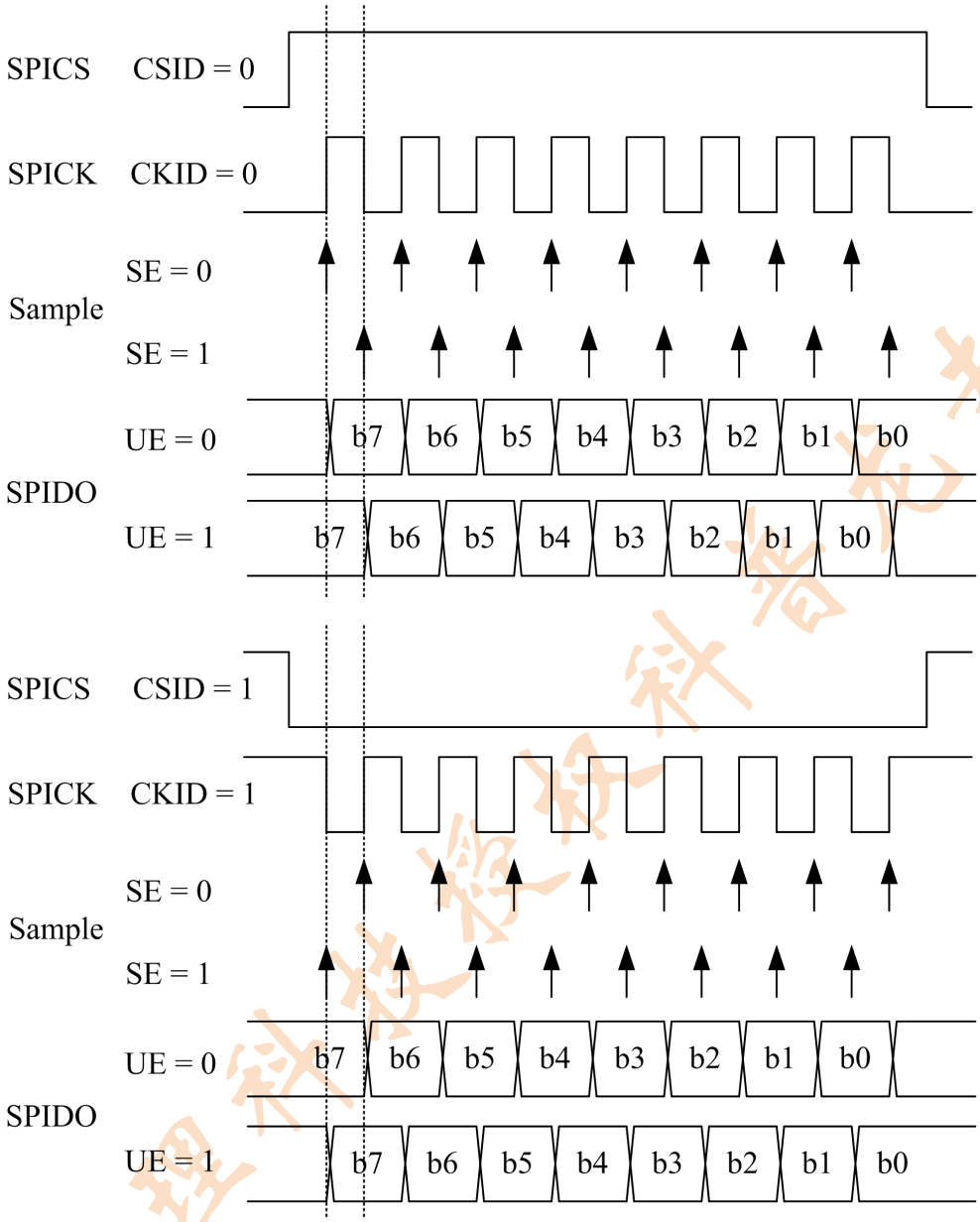
5) SPI_CNT: SPI DMA counter register (16bit addressing, write only)

SPI DMA 计数寄存器，只写，读出为不确定值。

此寄存器用于设置 DMA 操作的数目（按 Byte 计）并启动 DMA 传输。

如，需启动一次 512Byte 的 DMA 传输，写入 0x0200，此写入动作将启动本次传输。

6.3 传输波形



第 7 章 PAP

7.1 概述

PAP 接口（Parallel Active Port）是一个并行主动数据接口，它工作于主动模式，支持 8bit 或 16bit 数据宽度，具有读/写使能信号，并可支持 DMA 方式发送/接收数据。

PAP 传输支持由 CPU 直接驱动，写 PAPBUF 的动作将启动一次传输。

PAP 传输也支持 DMA 操作，每次 DMA 操作支持的数据量为 1-65535Byte，写 PAPCNT 的动作将启动一次传输。

PAP 在 DMA 发送数据（写）时支持普通模式和数据扩展模式。

普通模式是指，PAP 接口直接将原始数据发送出去。

数据扩展模式是指，PAP 接口可预设两个 16 位的数值 DAT0 和 DAT1，在发送过程中，硬件依次检查原始数据（1Byte）的每一 bit（从 LSB 到 MSB，或从 MSB 到 LSB），若该 bit 为 1，则发送预设值 DAT1，否则发送预设值 DAT0。此状态下，若设置为 16bit 数据宽度，则 DAT0/DAT1 一次性写出，若设置为 8bit 数据宽度，则 DAT0/DAT1 先后分两次写出。

PAP 在接收数据（读）时只支持普通模式，请勿在读状态下使能数据扩展模式，否则会导致不可预计的错误。

7.2 控制寄存器

1) PAPCON: PAP control register 0(32bit addressing)

23	22	21	20	19	18	17	16
reserved	reserved	reserved	reserved	reserved	PAPIE	EXTMSB	EXTE
r	r	r	r	r	rw	rw	rw
0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8
TS1	TS0	TH1	TH0	TW3	TW2	TW1	TW0
rw	rw	rw	rw	rw	rw	rw	rw
x	x	x	x	x	x	x	x

7	6	5	4	3	2	1	0
PND	PCLR	DW16ED	DW16EN	PRE	PWE	DIR	PAPE
r	w	rw	rw	rw	rw	rw	rw
1	0	0	0	0	0	0	0

PAPIE: 中断使能

0: 关闭，不允许 PAP 引发 CPU 中断；

1: 打开，允许 PAP 引发 CPU 中断；

EXTMSB: 数据扩展模式下，数据扩展顺序设置

0: 从 LSB 到 MSB 逐位检查原始数据；

1: 从 MSB 到 LSB 逐位检查原始数据；

EXTE: 数据扩展模式使能

0: 普通模式；

1: 数据扩展模式，此模式只支持数据发送（写），勿在数据接收（读）时设置此位；

TS1-0: 数据建立时间设置

0x0: 数据建立时间为 0；

0x1: 数据建立时间为 1 个系统时钟的宽度；

0x2: 数据建立时间为 2 个系统时钟的宽度；

0x3: 数据建立时间为 3 个系统时钟的宽度；

TH1-0: 数据保持时间设置

0x0: 数据保持时间不小于 0 ；

0x1: 数据保持时间不小于 1 个系统时钟的宽度；

0x2: 数据保持时间不小于 2 个系统时钟的宽度；

0x3: 数据保持时间不小于 3 个系统时钟的宽度；

TW3-0: 读/写使能信号宽度设置

0x0: 读/写使能信号宽度为 16 个系统时钟的宽度；

0x1: 读/写使能信号宽度为 1 个系统时钟的宽度；

0x2: 读/写使能信号宽度为 2 个系统时钟的宽度；

0x3: 读/写使能信号宽度为 3 个系统时钟的宽度；

0x4: 读/写使能信号宽度为 4 个系统时钟的宽度；

0x5: 读/写使能信号宽度为 5 个系统时钟的宽度；

0x6: 读/写使能信号宽度为 6 个系统时钟的宽度；

0x7: 读/写使能信号宽度为 7 个系统时钟的宽度；

0x8: 读/写使能信号宽度为 8 个系统时钟的宽度；

0x9: 读/写使能信号宽度为 9 个系统时钟的宽度；

0xA: 读/写使能信号宽度为 10 个系统时钟的宽度；

0xB: 读/写使能信号宽度为 11 个系统时钟的宽度；

0xC: 读/写使能信号宽度为 12 个系统时钟的宽度；

0xD: 读/写使能信号宽度为 13 个系统时钟的宽度；

0xE: 读/写使能信号宽度为 14 个系统时钟的宽度；

0xF: 读/写使能信号宽度为 15 个系统时钟的宽度；

PND: 中断请求标志，当一次传输完成或 dma 传输完成时会被硬件置 1。

有 3 种方法清除此标志

1，向 PCLR 写入‘1’；

2，写 PAPBUFL 寄存器来启动下一次传输；

3，写 PAPCNT 寄存器来启动下一次 dma；

PCLR: 软件在此位写入‘1’将清除 PND 中断请求标志，写入‘0’无效

DWED: 数据大小端选择

0: 16bit 模式下，低地址数据至端口低位。

8bit 模式下，数据至端口低位。

1: 16bit 模式下，低地址数据至端口高位。

8bit 模式下，数据至端口高位。

DW16EN: 8bit/16bit 模式选择

0: 8bit 模式；

1: 16bit 模式；

PRE: 读使能信号极性选择

0: 读使能信号空闲时为 0 电平，有效时为 1 电平；

1: 读使能信号空闲时为 1 电平，有效时为 0 电平；

PWE: 写使能信号极性选择

0: 写使能信号空闲时为 0 电平，有效时为 1 电平；

1: 写使能信号空闲时为 1 电平，有效时为 0 电平；

DIR: 传输方向设置

0: 发送数据；

1: 接收数据；

PAPE: PAP 接口使能

0: 关闭 PAP 接口；

1: 打开 PAP 接口；

2) PAPBUF: PAP buffer register (16bit addressing)

发送寄存器和接收寄存器共用此 SFR 地址。写入至发送寄存器，从接收寄存器读出。写此寄存器将启动一次发送或接收操作。

3) PAPDAT0: PAP data register 0 (16bit addressing)

用于数据扩展模式的 DAT0 寄存器，只写，读出为不确定值

4) PAPDAT1: PAP data register 1 (16bit addressing)

用于数据扩展模式的 DAT1 寄存器，只写，读出为不确定值

5) PAPADR: PAP dma start address register (32bit addressing)

PAP dma 起始地址寄存器，只写，读出为不确定值

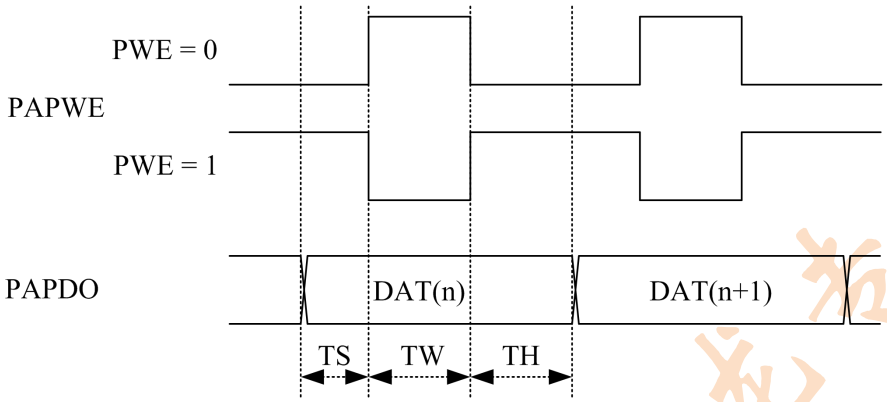
6) PAPCNT: PAP dma counter register (16bit addressing)

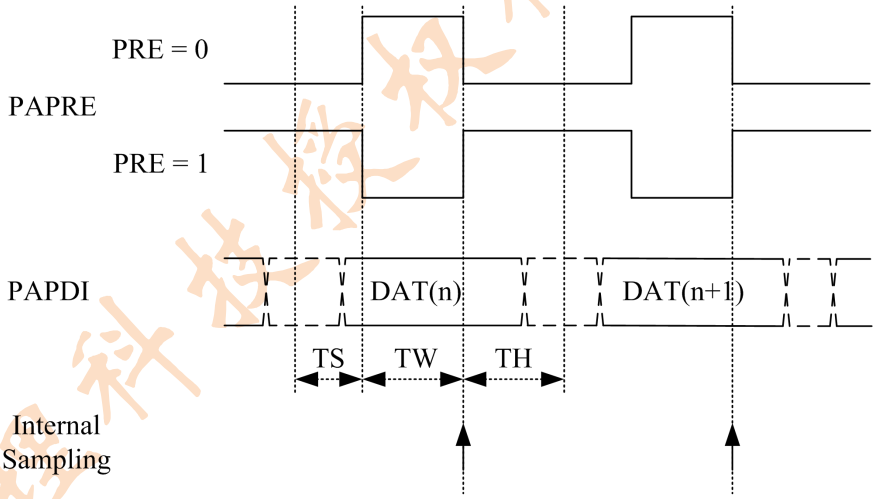
PAP dma 计数寄存器，只写，读出为不确定值

此寄存器用于设置 dma 操作的数目（按 Byte 计）并启动 dma 传输。

如，需启动一次 512Byte 的 dma 传输，写入 0x0200，此写入动作将启动本次传输。

7.3 传输波形





第 8 章 音频模块（AUDIO_ONCHIP）

8.1 概述

AC690X 内置音频模块包含了播放音乐的 Delta-Sigma DAC 和录制音乐的 Delta-Sigma ADC 这两个部分。DAC 为双通道，支持独立的左声道和右声道，两个通道同时使能或关闭。ADC 为三通道，支持独立的左声道，右声道和话筒（MIC）通道，三个通道可分别使能或关闭。

DAC/ADC 在结构上相互独立，可独立地开关，可工作于各自不同的采样率下。

与此部分相关的 SFR 如下：

数字部分：

DAC_CON DAC 控制寄存器

DAC_ADR DAC 起始地址寄存器

DAC_LEN DAC 采样点数寄存器

DAC_TRML DAC 左声道偏置微调寄存器

DAC_TRMR DAC 右声道偏置微调寄存器

ADC_CON ADC 控制寄存器

ADC_ADR ADC 起始地址寄存器

ADC_LEN ADC 采样点数寄存器

模拟模块控制部分：

DAA_CON0 DAC 模拟模块控制寄存器 0

DAA_CON1 DAC 模拟模块控制寄存器 1

DAA_CON2 DAC 模拟模块控制寄存器 2

DAA_CON3 DAC 模拟模块控制寄存器 3

DAA_CON4 DAC 模拟模块控制寄存器 4

处理能力需求：（48KHz 采样率下，其他采样率按比例换算）

功能	条件	apc_clk 频率需求 (MHz)
DAC	打开 DCC，双声道	35
	关闭 DCC，双声道	33
ADC	打开 DCC，1 通道	37
	打开 DCC，2 通道	40
	打开 DCC，3 通道	43
	关闭 DCC，3 通道	40

8.2 控制寄存器

1) DAC_CON: dac control register (16bit addressing)

15	14	13	12	11	10	9	8
DCCS3	DCCS2	DCCS1	DCCS0	reserved	reserved	reserved	BUFF
rw	rw	rw	rw	r	r	r	r
0	0	0	0	0	0	0	x

7	6	5	4	3	2	1	0
PND	CPND	DACIE	DACEN	DACSR3	DACSR2	DACSR1	DACSR0
r	w	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

DCCS3-0: dc cancelling filter setting，去直流滤波器配置

0000：去直流滤波器关闭；

0001-1111：去直流滤波器打开，数值越大，其高通转折点越低，建议设置为 14；

BUFF: dma buffer flag

0: 当前正在使用 buf0，buf1 可被读写；

1: 当前正在使用 buf1，buf0 可被读写；

PND: Pending，当 dma buf0 或 buf1 被使用完毕后，此位被硬件置 1

CPND: 写 1 清除 Pending，写 0 无效

DACIE: 中断允许

0：不允许 DAC PND 引发中断；

1：允许 DAC PND 引发中断；

DACEN: DAC 数字模块使能

0：DAC 数字模块关闭 ；

1：DAC 数字模块打开；

DACSR3-0:DAC 采样率设置

0000：44.1KHz；

0001：48KHz；

001x：32KHz；

0100：22.05KHz；

0101：24KHz；

011x：16KHz；

1x00：11.025KHz；

1x01：12KHz；

1x1x：8KHz；

2) ADC_CON: adc control register (16bit addressing)

15	14	13	12	11	10	9	8
DCCS3	DCCS2	DCCS1	DCCS0	CHE2	CHE1	CHE0	BUFF
rw	rw	rw	rw	rw	rw	rw	r
0	0	0	0	1	0	0	x

7	6	5	4	3	2	1	0
PND	CPND	ADCIE	ADCEN	ADCSR3	ADCSR2	ADCSR1	ADCSR0
r	w	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

CCS3-0:dc cancelling filter setting，去直流滤波器配置

0000：去直流滤波器关闭；

0001-1111：去直流滤波器打开，数值越大，其高通转折点越低，建议设置为 14；

CHE2-0: ADC 通道使能（CH0：left CH1：right CH2：mic）

0：不使用该通道；

1：使用该通道；

BUFF: dma buffer flag

0: 当前正在使用 buf0，buf1 可被读写；

1: 当前正在使用 buf1，buf0 可被读写；

PND: Pending，当 dma buf0 或 buf1 被使用完毕后，此位被硬件置 1

CPND: 写 1 清除 Pending，写 0 无效

ADCIE: 中断允许

0：不允许 ADC PND 引发中断；

1：允许 ADC PND 引发中断；

ADCEN: ADC 数字模块使能

0：ADC 数字模块关闭；

1：ADC 数字模块打开；

ADCSR3-0:ADC 采样率设置

0000：44.1KHz；

0001：48KHz；

001x：32KHz；

0100：22.05KHz；

0101：24KHz；

011x：16KHz；

1x00：11.025KHz；

1x01：12KHz；

1x1x：8KHz；

3) DAC_TRML: dac left channel trimming control register (8bit addressing)

7	6	5	4	3	2	1	0
DAC_TRML							
w	w	w	w	w	w	w	w
x	x	x	x	x	x	x	x

DAC 左声道偏置微调寄存器，应写入有符号数。复位为不确定值，在使用 DAC 之前，必须由软件初始化。

4) DAC_TRMR: dac right channel trimming control register (8bit addressing)

7	6	5	4	3	2	1	0
DAC_TRMR							
w	w	w	w	w	w	w	w
x	x	x	x	x	x	x	x

DAC 右声道偏置微调寄存器，应写入有符号数。复位为不确定值，在使用 DAC 之前，必须由软件初始化。

5) DAC_ADR: dac dma start address register (26bit addressing)

DAC DMA 操作起始地址寄存器，在使用 DAC 之前，必须由软件初始化对齐至 4Byte。

6) DAC_LEN: dac dma sample length register (16bit addressing)

DAC DMA 样点长度寄存器，在使用 DAC 之前，必须由软件初始化为 2 的倍数，允许写入值为 2-32768，超出此范围的设置值可能导致不可预料的错误。

例如，当 DAC_LEN 设置为 100 时，则 DAC 每次 DMA 过程消耗（左/右各）100 个样点。此时 DMA buffer 占用的空间为：100 * 2(CH) * 2(Byte) * 2(dual buffer) = 800 Byte

7) ADC_ADR: ladc dma start address register (26bit addressing)

ADC DMA 操作起始地址寄存器，在使用 ADC 之前，必须由软件初始化对齐至 4Byte。

8) ADC_LEN: ladc dma sample length register (16bit addressing)

ADC DMA 样点长度寄存器，在使用 ADC 之前，必须由软件初始化为 2 的倍数，允许写入值为 2-32768，超出此范围的设置值可能导致不可预料的错误。

例如，当 ADC_LEN 设置为 100 时，则 ADC 每条通道每次 DMA 过程消耗 100 个样点。此时 DMA buffer 占用的空间为：100 * 3(CH) * 2(Byte) * 2(dual buffer) =1200 Byte

8.3 DAC 模拟部分

1) DAA_CON0: dac analog module control register 0(16 bit addressing)

15	14	13	12	11	10	9	8
TRIM_SW	TRIM_SEL	TRIM_EN	Reserved	Reserved	Reserved	Reserved	PNS10k_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
MUTE	PNS_EN	HP_R_EN	HP_L_EN	LDO2_EN	LDO1_EN	DAC_DTSEL	DAC_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

TRIM_SW: TRIM 开关。

TRIM_SEL: TRIM 声道输出。

TRIM_EN: TRIM 使能控制位，高有效。

PNS10k_EN: DAC 输出下拉 10K 电阻控制位，高时为下拉 10K 电阻，低时为没有下拉 10K 电阻。

MUTE: DAC 和 AMUX 静音控制位，高有效。

PNS_EN: DAC 输出下拉 50K 电阻控制位，高时为下拉 50K 电阻，低时为没有下拉 50K 电阻。

HP_R_EN: DAC 右声道输出控制位，高有效；设为低时，VOUTR 输出高阻。

HP_L_EN: DAC 左声道输出控制位，高有效；设为低时，VOUTL 输出高阻。

LDO2_EN: DAC 内部 LDO2 使能控制信号，高有效；使用 LDO2 时，DACVDD 不需要外接电容。

LDO1_EN: DAC 内部 LDO1 使能控制信号，高有效。

DAC_DTSEL:DAC 时钟死区时间的设置。

DAC_EN: DAC 模拟部分使能控制信号，高有效。

2) DAA_CON1: dac analog module control register 1 (16bit addressing)

15	14	13	12	11	10	9	8
MIC_2_R	MIC_2_L	VCM_RSEL	RG_SEL4	RG_SEL3	RG_SEL2	RG_SEL1	RG_SEL0
rw	rw	rw	rw	rw	Rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
LR_2_R	LR_2_L	Reserve	LG_SEL4	LG_SEL3	LG_SEL2	LG_SEL1	LG_SEL0
rw	rw	rw	rw	rw	Rw	rw	rw
0	0	0	0	0	0	0	0

MIC_2_R: 在卡拉 OK 模式下，把 MIC 放大后的信号在右声道输出，高有效。

MIC_2_L: 在卡拉 OK 模式下，把 MIC 放大后的信号在左声道输出，高有效。

VCM_RSEL： VCM 偏置电压电阻选择，高时缩短 DAC 稳定时间。

RG_SEL<4:0>: 右声道输出模拟音量控制器。

LR_2_R: Lin-in 输入的左右声道信号合成后在右声道输出，高有效。

LR_2_L: Lin-in 输入的左右声道信号合成后在左声道输出，高有效。

LG_SEL<4:0>: 左声道输出模拟音量控制器。

3) DAA_CON2: dac analog module control register 2 (16bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	AMUX_MUTE	AMUX_BIAS_EN	VCM_OUT_PD	VCM_OUT_EN	VCM_EN	VCM_DET_EN
rw	rw	rw	rw	rw	Rw	rw	rw
0	0	0	0	0	0	0	1

7	6	5	4	3	2	1	0
AMUX_EN	AMUX_G	LIN2R_EN	LIN2L_EN	LIN1R_EN	LIN1L_EN	LIN0R_EN	LIN0L_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

AMUX_MUTE: AMUX 静音使能控制位，高有效。

AMUX_BIAS_EN: AMUX 偏置电压使能控制位，高有效。

VCM_OUT_PD: VCMO 下拉 1K 电阻使能控制位，高有效。

VCM_OUT_EN: VCM 使能控制位，高有效。

VCM_EN: VCM 使能控制位，高有效。

VCM_DET_EN: VCM 复位使能控制位，高有效。

AMUX_EN: AMUX BYPASS 使能控制位，高有效。

AMUX_G: AMUX 增益控制器，高有效。

LIN2R_EN: PB12Enable.

LIN2L_EN: PB11 Enable.

LIN1R_EN: PA4 Enable.

LIN1L_EN: PA3 Enable.

LIN0R_EN: PA2 Enable.

LIN0L_EN: PA1 Enable.

4) DAA_CON3: dac analog module control register 3(16bit addressing)

15	14	13	12	11	10	9	8
TRIM_OUT	MIC_EN	LIN2R_BIAS_EN	LIN2L_BIAS_EN	LIN1R_BIAS_EN	LIN1L_BIAS_EN	LIN0R_BIAS_EN	LIN0L_BIAS_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
MIC_MUTE	MIC_GX2	MIC_G5	MIC_G4	MIC_G3	MIC_G2	MIC_G1	MIC_G0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

MIC_EN: MIC 使能控制位，高有效。

LIN2R_BIAS_EN: PB12 Bias Enable.

LIN2L_BIAS_EN: PB11 Bias Enable.

LIN1R_BIAS_EN: PA4 Bias Enable.

LIN1L_BIAS_EN: PA3 Bias Enable.

LIN0R_BIAS_EN: PA2 Bias Enable.

LIN0L_BIAS_EN: PA1 Bias Enable.

MIC_MUTE: MIC 静音控制位，高有效。

MIC_GX2: MIC 增益倍增控制器，一般情况下不建议使用，高有效。

MIC_G<5:0>: MIC 信号放大器增益控制器。

5) DAA_CON4: dac analog module control register 4(16bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
r	r	r	r	r	r	r	r
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	MIC_NEG12	DAC_ISEL_HALF	DAC_ISEL_THIRD	DAC_ISEL5U
r	r	r	r	rw	rw	rw	rw
0	0	0	0	0	0	0	0

DAC_ISEL5U: DAC 全功耗电流选择，LDO1 为 4mA，LDO2 为 5mA。

DAC_ISEL_THIRD: DAC 半功耗电流选择，LDO1 为 2mA，LDO2 为 2.5mA。

DAC_ISEL_HALF: DAC1/3 功耗电流选择，LDO1 为 1.2mA，LDO2 为 1.25mA。

MIC_NEG12: MIC -12dB 增益放大控制位。

6) DAA_CON5: dac analog module control register 5(16bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
r	r	r	r	r	r	r	r
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
Reserved	Reserved	ADC_DIT	ADC_DOE	ADC_COE	DAC_EXT	DATEN	CKEN
r	r	r	r	rw	rw	rw	rw
0	0	0	0	0	0	0	0

CLKEN: DAC 时钟输出使能，高有效，当为 1 时，PA4 输出 DAC 时钟

DATEN: DAC 数据使出使能，高有效，当为 1 时，PA6（右）、PA5（左）输出 DAC 数据信号

DAC_EXT: DAC 外部控制测试使能，高有效，当 DAC_EXT 为 1 时，DAC 会接到外部的 PA4、PA5、PA6，其中 PA4 为时钟输入，PA6 为右输入，PA5 为左输入。

ADC_COE: ADC 时钟输出使能，高有效，时钟输出口 PB8，通道 0 数据输入口 PB3、PB2，通道 1 数据输入口 PB5、PB4，通道 2 数据输入口 PB7、PB6。

ADC_DOE: ADC 数据输出使能，高有效，时钟输入口 PB8，通道 0 数据输出口 PB3、PB2，通道 1 数据输出口 PB5、PB4，通道 2 数据输出口 PB7、PB6。

ADC_DIT: ADC dither 控制位 0：打开 1：关闭。

8.4 ADC 模拟部分

1) ADC_CON0: adc analog module control register 0(16 bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	ADC_ISEL	ADC0_PGA_EN	ADC0_PGA_G3	ADC0_PGA_G2	ADC0_PGA_G1	ADC0_PGA_G0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
ADC0_DITHER_CFG1	ADC0_DITHER_CFG0	ADC0_S1_ISEL1	ADC0_S1_ISEL0	ADC0_TEST	ADC0_FF_EN	ADC0_CHANNEL_EN	ADC0_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

ADC0_EN: ADC0 模拟模块使能。

ADC0_CHANNEL_EN:ADC0 通道使能。

ADC0_FF_EN: ADC0 FF 模式使能位，高有效。

ADC0_TEST: ADC0 测试模式使能位，高有效。

ADC0_S1_ISEL1-0: ADC0 OTA 偏置电流选择。

ADC0_DITHER_CFG1-0: ADC0 dither GFG 配置。

ADC0_PGA_G3-0: ADC0 PGA 增益控制选择。

ADC0_PGA_EN: ADC0 PGA 使能控制位

ADC_ISEL: ADC 电流选择位。

2) ADC_CON1: adc analog module control register 1(16 bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	ADC1_PGA_EN	ADC1_PGA_G3	ADC1_PGA_G2	ADC1_PGA_G1	ADC1_PGA_G0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
ADC1_DITHER_CFG1	ADC1_DITHER_CFG0	ADC1_S1_ISEL1	ADC1_S1_ISEL0	ADC1_TEST	ADC1_FF_EN	ADC1_CHANNEL_EN	ADC1_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

ADC1_EN: ADC1 模拟模块使能。

ADC1_CHANNEL_EN:ADC1 通道使能。

ADC1_FF_EN: ADC1 FF 模式使能位，高有效。

ADC1_TEST: ADC1 测试模式使能位，高有效。

ADC1_S1_ISEL1-0:ADC1 OTA 偏置电流选择。

ADC1_DITHER_CFG1-0:ADC1 dither GFG 配置。

ADC1_PGA_G3-0:ADC1 PGA 增益控制选择。

ADC1_PGA_EN:ADC1 PGA 使能控制位。

3) ADC_CON2: adc analog module control register 2(16 bit addressing)

15	14	13	12	11	10	9	8
Reserved	Reserved	Reserved	ADC2_PGA_EN	ADC2_PGA_G3	ADC2_PGA_G2	ADC2_PGA_G1	ADC2_PGA_G0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
ADC2_DITHER_CFG1	ADC2_DITHER_CFG0	ADC2_S1_ISEL1	ADC2_S1_ISEL0	ADC2_TEST	ADC2_FF_EN	ADC2_CHANNEL_EN	ADC2_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

ADC2_EN: ADC2 模拟模块使能。

ADC2_CHANNEL_EN:ADC2 通道使能。

ADC2_FF_EN: ADC2 FF 模式使能位，高有效。

ADC2_TEST: ADC2 测试模式使能位，高有效。

ADC2_S1_ISEL1-0:ADC2 OTA 偏置电流选择。

ADC2_DITHER_CFG1-0:ADC2 dither GFG 配置。

ADC2_PGA_G3-0:ADC2 PGA 增益控制选择。

ADC2_PGA_EN:ADC2 PGA 使能控制位。

8.5 数据通路

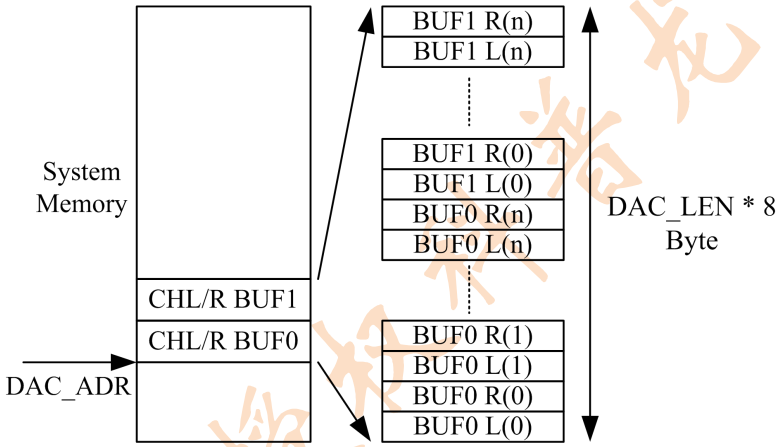
DAC/ADC 的数据都是通过 DMA 的方式与片内系统连接的，使用了 dual-buffer（乒乓缓冲）的方式，每个样点的精度均为 16bit(2Byte)。

DAC 的 buf0/buf1 容纳样点数由 DAC_LEN 寄存器指定，总 buffer 需求为：

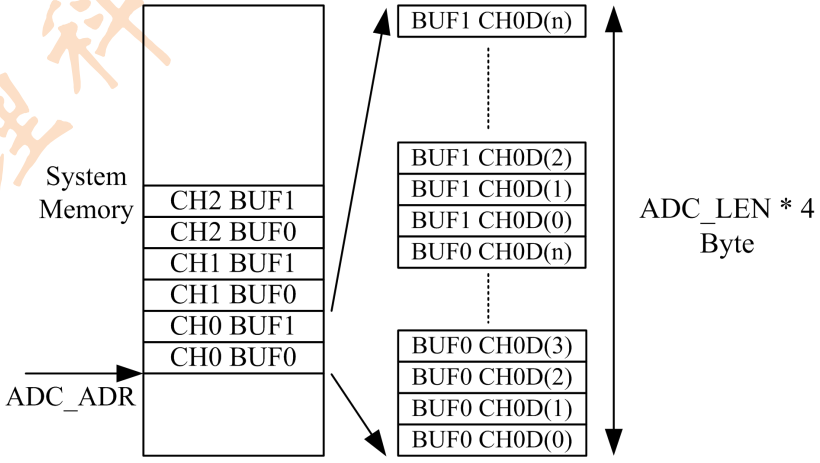
DAC_LEN * 2(CH) * 2(Byte) * 2(dual buffer) = DAC_LEN * 8 Byte

ADC 每声道的 buf0/buf1 容纳样点数由 ADC_LEN 寄存器指定，总 buffer 需求为：

ADC_LEN * 3(CH) * 2(Byte) * 2(dual buffer) = ADC_LEN * 12 Byte



DAC 数据组织示意图



ADC 数据组织示意图

第 9 章 AUDIO_LINK

9.1 概述

Audio Link 接口(以下简称 ALNK)是一个通用的双声道音频接口，用于连接片外的 DAC 或 ADC，连接信号有 MCLK，SCLK，LRCK，DATA。支持 IIS/左对齐/右对齐/DSP0/DSP1 共 5 种模式，原生支持 16/24bit 数据位宽，对 18/20/32bit 位宽的设备可提供兼容支持。ALNK 通过 DMA 的方式与片内系统进行数据连接，不论输入或输出，每条通道占用 buffer 的大小可由软件配置。

MCLK 是音频接口的主时钟，Delta-Sigma 类型的 DAC/ADC 都是需要此信号的。MCLK 可由 ALNK 提供给 DAC/ADC，也可由 DAC/ADC 提供给 ALNK。

ALNK 可配置为主机模式或从机模式。主机模式是指 SCLK 和 LRCK 由本模块提供，此时需从外部提供相应采样率参考时钟。有 3 种方式可供选择：

1) 挂接 22.5792MHz 晶振以支持 44.1KHz 组别的采样率。

2) 挂接 24.576MHz 晶振以支持 48/32KHz 组别的采样率。

3) 从 PA15/PB6 端口输入相应频率的 MCLK

从机模式是指 SCLK 和 LRCK 由外部 DAC/ADC 提供，此时采样率由外部提供的 LRCK 确定，因此芯片不需外接晶振来提供采样率参考时钟。

ALNK 具备 4 条独立的通道，可相互独立地工作，每条通道都可独立配置为输入或输出，也可配置为不同的连接模式。需注意的是它们共用了 MCLK/SCLK/LRCK 信号，因此使用上有一定的限制。根据 LRCK 信号的不同，将 IIS/左对齐/右对齐定义为基本模式，DSP0/DSP1 定义为扩展模式。有：

1) 4 条通道只能同时工作于基本模式或扩展模式。

不可一些通道工作于基本模式，另一些通道工作于扩展模式。

2) 基本模式下每条通道都只支持双声道。

扩展模式下每条通道均可支持单声道或立体声，这由硬件自动适应，当每帧的 SCLK 时钟个数大于等于立体声所需的时钟个数时，该通道工作于立体声状态，否则工作于单声道状态（只有左声道）。

下表为 ALNK 支持的采样率设置列表

SR(KHz)	256fs	384fs	512fs
8		3.072MHz，推荐	
11.025	2.8224MHz，可用		5.6448MHz，推荐
12	3.072MHz，可用		6.144MHz，推荐
16		6.144MHz，推荐	
22.05	5.6448MHz，可用		11.2896MHz，推荐
24	6.144MHz，可用		12.288MHz，推荐
32		12.288MHz，推荐	
44.1	11.2896MHz，可用		22.5792MHz，推荐
48	12.288MHz，可用		24.576MHz，推荐
64		24.576MHz，推荐	
88.2	22.5792MHz，推荐		
96	24.576MHz，推荐		

下表为 ALNK 使用的 IO 端口列表

信号名称	占用 IO 端口
MCLK	PA15/PB6
SCLK	PA9/PB0
LRCK	PA10/PB1
CH0DAT	PA11/PB2
CH1DAT	PA12/PB3
CH2DAT	PA13/PB4
CH3DAT	PA14/PB5

下表为 ALNK 使用的 SFR 列表

SFR 名称	描述
ALNK_CON0	ALNK 控制寄存器 0
ALNK_CON1	ALNK 控制寄存器 1
ALNK_CON2	ALNK 控制寄存器 2
ALNK_ADR0	通道 0 DMA 起始地址寄存器
ALNK_ADR1	通道 1 DMA 起始地址寄存器
ALNK_ADR2	通道 2 DMA 起始地址寄存器
ALNK_ADR3	通道 3 DMA 起始地址寄存器
ALNK_LEN	通道 0-3 DMA 数据长度寄存器

9.2 控制寄存器

1) ALNK_CON0: control register 0 (16bit addressing)

15	14	13	12	11	10	9	8
FLAG3	FLAG2	FLAG1	FLAG0	ALNKE	SCKINV	F32E	MOE
r	r	r	r	rw	rw	rw	rw
x	x	x	x	0	0	0	0

7	6	5	4	3	2	1	0
SOE	DSPEN	LRDIV1	LRDIV0	MDIV1	MDIV0	MSRC1	MSRC0
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

FLAGx: 通道 x dma buffer flag

0: 当前正在使用 buf0，buf1 可被读写；

1: 当前正在使用 buf1，buf0 可被读写；

ALNKE： ALNK 模块使能

0：模块关闭；

1：模块打开；

SCKINV： SCLK 边沿选择

0：SCLK 的下降沿更新数据，上升沿采样数据；

1：SCLK 的上升沿更新数据，下降沿采样数据；

F32E： 主机模式下，每帧数据的 SCLK 个数

0：64 SCLKs per-frame；

1：32 SCLKs per-frame；

MOE： 0：不输出 MCLK 至对应 IO 端口；

1：输出 MCLK 至对应 IO 端口；

SOE： 0：不输出 SCLK/LRCK 至对应 IO 端口；

1：输出 SCLK/LRCK 至对应 IO 端口；

DSPE： 0：ALNK 工作于基本模式（IIS/左对齐/右对齐）；

1：ALNK 工作于扩展模式（DSP0/DSP1）；

LRDIV: ALNK 采样率设置

00：从外部输入 LRCK（即从机模式）；

01：LRCK 为 MCKD 的 1/256，即 256fs；

10：LRCK 为 MCKD 的 1/384，即 384fs；

11：LRCK 为 MCKD 的 1/512，即 512fs；

MDIV: MCLK 前置分频器设置

00：MCKD 为 MCLK 的 1 分频；

01：MCKD 为 MCLK 的 2 分频；

10：MCKD 为 MCLK 的 4 分频；

11：MCKD 为 MCLK 的 8 分频；

MSRC: 设置 MCLK 的来源

00：从外部输入 MCLK；

01：系统时钟；

10：OSC CLK；

11：PLL ALNK CLK；

注意：OSC CLK 和 PLL ALNK CLK 参考 Clock System spec 的说明。

2) ALNK_CON1: control register 1 (16bit addressing)

15	14	13	12	11	10	9	8
T3DIR	T3LEN	T3MOD		T2DIR	T2LEN	T2MOD	
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
T1DIR	T1LEN	T1MOD		T0DIR	T0LEN	T0MOD	
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

TxDIR： 通道 x 方向设置

0：发送；

1：接收；

TxLEN： 通道 x 数据位宽设置

0：16bit；

1：24bit；

TxMOD： 通道 x 工作模式设置，与 DSPEN 一起决定该通道的工作模式

DSPEN	TxMOD	
x	0	不使用通道 x
0	1	IIS（数据延后 1bit）
	2	左对齐
	3	右对齐
1	1	DSP0（数据延后 1bit）
	2	DSP1
others		保留，不可设置为此模式

3) ALNK_CON2: control register 2 (8bit addressing)

7	6	5	4	3	2	1	0
PND3	PND2	PND1	PND0	CPND3	CPND2	CPND1	CPND0
r	r	r	r	w	w	w	w
0	0	0	0	0	0	0	0

PNDx: 通道 x Pending，当 dma buf0 或 buf1 被使用完毕后，此位被硬件置 1

CPNDx: 写 1 清除 Pending，写 0 无效

4) ALNK_ADR: alnk dma start address register (26bit addressing)

ALNK DMA 操作起始地址寄存器，在使用 ALNK 之前，必须由软件初始化对齐至 4Byte。

5) ALNK_LEN: alnk dma sample length register (16bit addressing)

ALNK DMA 样点长度寄存器，在使用 ALNK 之前，必须由软件初始化为 2 的倍数，允许写入值为 2-32768，超出此范围的设置值可能导致不可预料的错误。

例如，当 ALNK_LEN 设置为 100 时，则 ALNK 每条通道每次 DMA 过程消耗 100 个样点。此时每通道 DMA buffer 占用的空间为：

16bit data: 100 * 2(CH) * 2(Byte) * 2(dual buffer) = 800 Byte

24bit data: 50 * 2(CH) * 4(Byte) * 2(dual buffer) = 800 Byte

9.3 数据组织结构

ALNK 的数据都是通过 DMA 的方式与片内系统连接的，使用了 dual-buffer（乒乓缓冲）的方式，每条通道的 buf0/buf1 容纳样点数由 ALNK_LEN 寄存器指定，总 buffer 需求为：

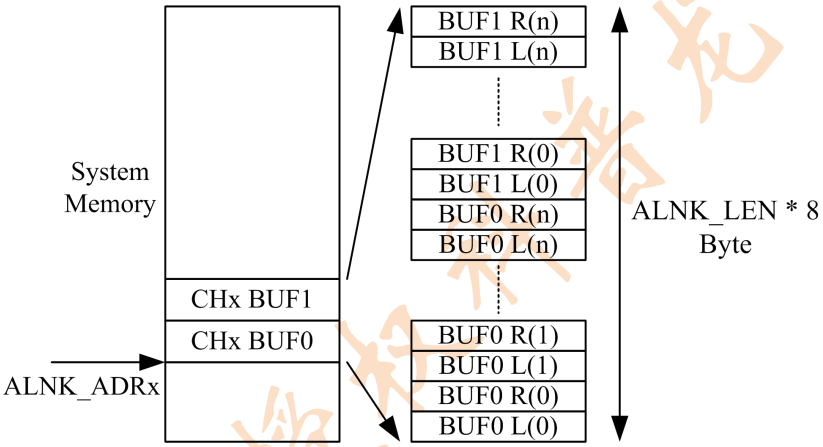
16bit data: ALNK_LEN * 2(CH) * 2(Byte) * 2(dual buffer) = ALNK_LEN * 8 Byte

24bit data: ALNK_LEN / 2 * 2(CH) * 4(Byte) * 2(dual buffer) = ALNK_LEN * 8 Byte

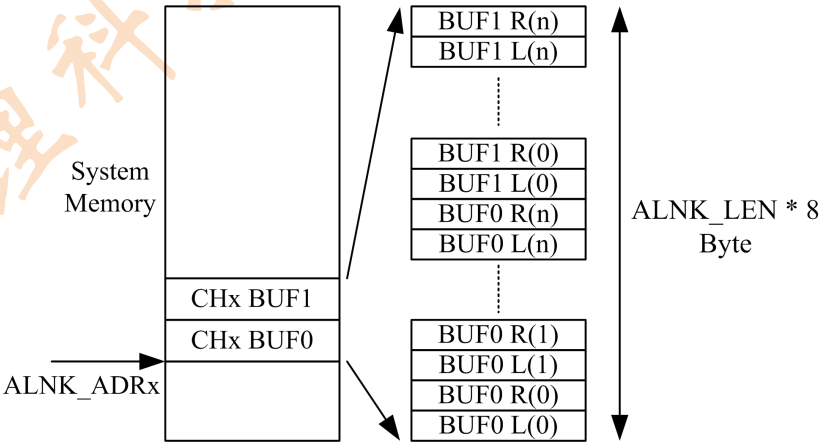
当四条通道一起使用时，共需要 4 倍大小的 buffer。

特别的，在单声道状态下，只有左声道会被使用到，右声道将被停用。

16bit 数据宽度时，数据组织如下图。



24bit 数据宽度时，数据组织如下图。



第 10 章 低功耗模式（LOWPOWER）

10.1概述

Low Power Mode 具体可分为 Idle/Standby/Sleep 三种类型的低功耗模式，用于在不同需求下让系统进入低功耗的非工作状态，以尽可能的节省电能。在指定的唤醒事件发生时，系统会退出当前的低功耗模式，进入正常工作状态。

唤醒事件分为两大类：

1，任一外设的中断请求（pending）出现，且当时该模块中断使能打开时，发生第一类唤醒事件。

2，当某 WAKEUP 功能被使能（请参考 Wakeup 部分文档）且发生了相应的唤醒事件，或 RTC 的 2mS/500mS/闹钟唤醒被使能，或 LVD 被使能且检测到低电压时，发生第二类唤醒事件。

Idle 状态下，CPU 停止工作，其他被使能的设备则正常工作。当发生第一类或第二类唤醒事件时，CPU 将从 Idle 状态被唤醒。若总中断有使能，则进入相应中断入口（如果由第二类唤醒，则不会进中断），执行该中断服务程序。中断返回后，CPU 返回进入 Idle 的后一条地址处继续执行指令。

Standby 状态下，所有数字设备停止工作，但被使能的模拟设备仍然正常工作。当发生第二类唤醒事件时，CPU 将从 Standby 状态被唤醒，继续从进入 Standby 的后一条地址处继续执行指令，同时所有的其他数字外设也回到正常工作状态。

Sleep 状态下，所有数字和时钟相关的模拟设备（包括 PLL，XOSC，EOSC0，EOSC1，HTC，RC）均停止工作。当发生第二类唤醒事件时，CPU 将从 Sleep 状态被唤醒，继续从进入 Sleep 的后一条地址处继续执行指令，同时所有的其他数字和模拟设备也回到正常工作状态。

10.2相关寄存器

1) PCON: power control register (8bit addressing)

7	6	5	4	3	2	1	0
RSRC2	RSRC2	RSRC2	SRST	SSMODE	SLEEP	STANDBY	IDLE
r	r	r	w	rw	w	w	w
x	x	x	0	0	0	0	0

RSRC2-0: 系统复位源指示

000: 上电复位；

001: VCM 复位；

010: PB1 或 PD1 的 4 秒‘0’电平复位；

011: LVD 低电压复位；

100: WDT 溢出复位；

101: SRST 软件复位；

others: 保留；

SRST: 只写，软件复位控制位，写入‘1’将导致系统复位

SSMODE: sleep/standby mode，睡眠/待机模式选择

0: Sleep/standby 状态下，当指定唤醒事件发生后，系统先进行 8mS 的延时，然后才真正唤醒整个系统。

1: Sleep/standby 状态下，当指定唤醒事件发生后，系统立即被唤醒。

SLEEP: 只写，写入‘1’将进入 Sleep 模式

STANDBY: 只写，写入‘1’将进入 Standby 模式

IDLE: 只写，写入‘1’将进入 Idle 模式

第 11 章 红外过滤（IRFLT）

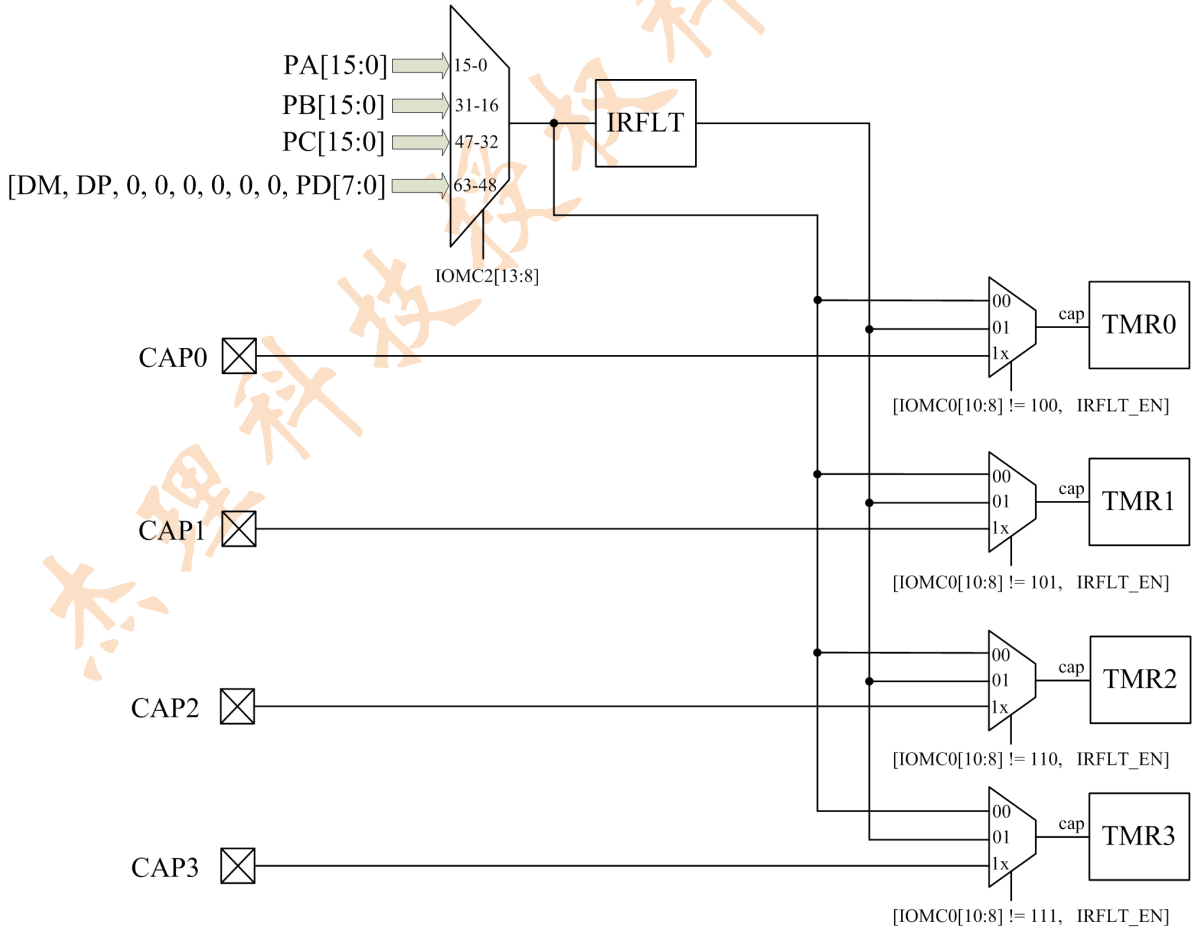
11.1 概述

IRFLT 是一个专用的硬件模块，用于去除掉红外接收头信号上的窄脉冲信号，提升红外接收解码的质量。

IRFLT 使用一个固定的时基对红外信号进行采样，必须连续 4 次采样均为‘1’时，输出信号才会变为‘1’，必须连续 4 次采样均为‘0’时，输出信号才会变为‘0’。换言之，脉宽小于 3 倍时基的窄脉冲将被滤除。改变该时基的产生可兼容不同的系统工作状态，也可在一定范围内调整对红外信号的过滤效果。

通过对 IOMC（IO re-mapping）寄存器的配置，可以将 IRFLT 插入到系统 4 个 timer 中某一个的捕获引脚之前。例如通过 IOMC 寄存器选择了 IRFLT 对 timer1 有效，并且 IRFLT_EN 被使能之后，则 IO 口的信号会先经过 IRFLT 进行滤波，然后再送至 timer1 中进行边沿捕获。

具体连接情况请参见下述示意图。



11.2 控制寄存器

1) IRFLT_CON: irda filter control register (8bit addressing)

PSEL3	PSEL2	PSEL1	PSEL0	TSRC1	TSRC0	reserved	IRFLT_EN
rw	rw	rw	rw	rw	rw	r	rw
x	x	x	x	x	x	0	0

PSEL3-0 时基发生器分频选择

0000: 分频倍数为 1；

0001: 分频倍数为 2；

0010: 分频倍数为 4；

0011: 分频倍数为 8；

0100: 分频倍数为 16；

0101: 分频倍数为 32；

0110: 分频倍数为 64；

0111: 分频倍数为 128；

1000: 分频倍数为 256；

1001: 分频倍数为 512；

1010: 分频倍数为 1024；

1011: 分频倍数为 2048；

1100: 分频倍数为 4096；

1101: 分频倍数为 8192；

1110: 分频倍数为 16384；

1111: 分频倍数为 32768；

TSRC1-0: 时基发生器驱动源选择

00: 选择 LSB_CLK 来驱动时基发生器；

01: 选择 RC 时钟来驱动时基发生器；

10: 选择 OSC _CLK 时钟来驱动时基发生器；

11: 选择 PLL_48M 时钟来驱动时基发生器；

IRFLT_EN: IRFLT 使能

0: 关闭 IRFLT；

1: 打开 IRFLT；

11.3 时基选择

PSEL 选定的分频倍数 N 和 TSRC 选定的驱动时钟的周期 Tc 共同决定了 IRFLT 用于采样红外接收信号的时基 Ts

Ts = Tc * N

例如，当选择 32KHz 的 OSC 时钟，并且分频倍数为 1 时，Ts = 30.5uS。根据 IRFLT 的工作规则，所有小于(30.5*3=91.5uS)的窄脉冲信号，均会被滤除。

又如，当选择 48MHz 的系统时钟，并且分频倍数为 1024 时，Ts = 21.3uS。根据 IRFLT 的工作规则，所有小于(21.3*3=63.9uS)的窄脉冲信号，均会被滤除。

第 12 章 液晶显示控制器（LCDC）

12.1概述

LCD 模块控制器主要负责控制推 LCD 屏幕。在 AC690X 中，最大可以控制推 6 COM & 16 SEG 的屏幕。LCD 模块可以使用 RTC 32KHz 晶振时钟或内部 RC(约 250KHz/8)作为模块时钟。

12.2控制寄存器

1) LCDC_CON0：LCDC control register 0(16bit addressing).

15	14	13	12	11	10	9	8
COM_TOTAL_CNT		CHARGE_MODE		CHARGE_DUTY_SEL			
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

7	6	5	4	3	2	1	0
FF	LCDC_VDD_SEL			BIAS_MODE		DOT_EN	LCDC_EN
rw	rw	rw	rw	rw	rw	rw	rw
0	0	0	0	0	0	0	0

COM_TOTAL_CNT：选择 COM 的数目(Duty)；

00：3COM；

01：4COM；

10：5COM；

11：6COM；

➢ COM 数与 Duty 的关系：1/Duty = 1/(COMCNT + 3)；

CHARGE_MODE ： 充电模式控制

00：一直用弱充电模式；

01：一直用强充电模式；

10：交替充电模式 A；

11：交替充电模式 B；

➢ 交替充电模式 A 在状态切换时开始转强驱动，经过 CHGDUTY+1 个 32KHz 时钟后撤销强驱动；

➢ 交替充电模式 B 在状态切换前半个周期（32KHz 时钟）开始转强驱动，经过 CHGDUTY+1.5 个 32KHz 时钟后撤销强驱动；

CHARGE_DUTY_SEL：交替充电模式下强充电占的 Cycle 数（按 32KHz 时钟）

0000：1/128；

0001：2/128；

0010：3/128；

0011：4/128；

……

1111：16/128；

FF：Frame Frequency，帧频率控制

0：FLCD = 32KHz/128；

1：FLCD = 32KHz/64；

- FLCD 为模块状态切换时钟频率

LCDC_VDD_SEL：VLCD 电压控制

000：2.6V；

001：2.7V；

……

110：3.2V；

111：3.3V；

BIAS_MODE：BIAS 选择

00：模拟模块工作禁止；

01： 1/2 bias；

10： 1/3 bias；

11： 1/4 bias；

DOT_EN：“跳秒”信号使能，(COM0/ SEG31)输出一个 1Hz 的信号，可作为“跳秒”脚

0：不使能；

1：使能；

LCDC_EN：LCDC 模块时能

0：不使能；

1：使能；

LCDC 时钟选择说明

2) CLK_CON1：clock system control register1(16bit addressing).

15	14	13	12	11	10	9	8
						LCD_CKSEL	
						rw	rw
						0	0

LCD_CKSEL：LCD 时钟选择

00：选择 wclk 作为 LCD 时钟；

01：选择 rtosl_clk 作为 LCD 时钟；

10：选择 lsb_clk 作为 LCD 时钟；

11：选择 1'b1 作为 LCD 时钟；

使用说明

a. 首先选择 LCD 模块工作时钟，如果系统有 32K 晶振，则选 32KHz 晶振时钟，否则选内部 RC 时钟；

b. 其次设定 LCD 电压（VLCDS）、偏置（BIAS）、充电模式（CHGMOD、CHGDUTY）等；

c. 再次选择 COM 和 SEG 的位置。例如选择需要 4 个 COM，选择 COM0(PC5), COM1(PC4), COM2(PC3), COM3(PC2), 然后选择 SEG0~SEG7(PA0~PA7)，组成 4COMX8SEG。那么控制器选择如下：

```
COM_TOTAL_CNT =  01; //选择 COM
SEG_IOEN0 = 0xff; //选择 SEG
```

d. 然后打开 LCD 模块

```
LCDCCON0 |= 0x01;      //打开 LCD
```

e. 填写数据：共有 6 个 COM，每个 SEG 需要 22bit，填写数据分别是：

```
SEG0_DAT:{ portc_die[5:0],  porta_die};
SEG1_DAT:{ portc_hd[5:0],  porta_hd};
SEG2_DAT:{ portc_pd[5:0],  porta_pd};
SEG3_DAT:{ portc_pu [5:0],  porta_pu};
SEG4_DAT:{ portc_out[5:0],  porta_out};
SEG5_DAT:{ portc_dir[5:0],  porta_dir};
```

注意：AC690X 中 SEG16-SEG21 和 COM5-COM0 分别为同个 IO，在使用中 SEG 的优先级大于 COM 的，如 PC0 若被选用做 SEG，则对应于该 IO 的 COM5 无效；（具体可查看 IO-mapping）。