CJC2100

音频编解码 SOC For UAC

Edition	Author	Date	Description
V1.0	Yanlan liu	2016.12	音频编解码 SOC For UAC

1. overview

CJC2100 is a Cortex-M0+ based MCU, designed for USB headphone appliances.

It integrates one 32-bit RISC CPU with 8KB SRAM, USB, UART, IIC, audio codec, GPIO, TIMER, WDT, PWM, SPI, IIS, SPDIF, PDM, SARADC, PLL, LDO etc.

CJC2100 can boot from external flash through SPI interface. After powered on, the program is read from external flash into internal SRAM for execution.

The CJC2100 can run up to 48MHz, and it is designed with special care to minimize the power consumption while allowing for the flexibility to reach for high performance. It includes the clock gating for individual IP, and CJC2100 can be further operated under different power-saving modes: Normal, Idle, Standby, Power-down, different mode have different clock and power strategy.

1.1 Features

- **Cortex-M0+ Like**
- **LDO**
  - Built-in LDO for wide operating voltage range:3.3V/1.8V
- **Memory**
  - Support program memory up to 16KB
  - RAM:8KB SRAM
- **In-system programming & In-Circuit programming by USB/UART**
- **Clock control**
  - Programmable system clock source
  - 12MHz internal RC-oscillator(1% accuracy at 25℃)
  - Support external crystal oscillator
  - 10KHz internal low-power RC-oscillator for watchdog and idle wake-up
- **USB Compliance**
  - USB Spec.V2.0 high speed/full speed mode compatible
  - USB Audio Class V1.0/V2.0 compatible
  - USB Human Interface Device V1.1 compatible
  - Support USB suspend/resume/reset function
  - Support control, interrupt, bulk and isochronous data transfer
- **Audio codec**
  - Default sample rate:192k/176.4k/96k/88.2k/48k/44.1k(192k/176.4k are available only in USB audio class V2.0/High-speed mode)
  - Support bit length:16/24/32bit
- **I/O port**
  - Up to 32 general purpose I/O(GPIO)
- **TIMER**
  - 3 internal timers
  - Internal or external clock source selection
  - Interrupt can be issued upon overflow and time-up

- Each timer has two match registers
- Supports the incrementing and decrementing models

- **Watchdog Timer**
- During the timeout, the outputs are one or a combination of the following signals
  -----System reset
  -----System interrupt
  -----External interrupt
- 32-bit down counter
- Internal or external clock source selection
- A variable time-out period of reset
- Access protection

- **PWM**
- One 16-bit timers PWM channel
- Programmable duty control of output waveform
- Auto reload mode or one-shot pulse mode
- Capture and compare function

- **UART**
- Programmable baud rates ,Baud rate up to 921.6K
- Support 38KHZ house IR transceiver

- **SPI**
- One specified SPI interface as AHB device for boot loader and APB device for write back
- One master/slave programmable SPI interface as APB device
- speed up to 40MHz

- **I2C**
- compatible with Philips IIC standard

- **I2S/SPDIF**
- support the Sony/Philips Digital Interface Format(SPDIF) transmitter
- support master/slave mode and 16/24/32bit data width

- **PDM(For digital microphone)**
- One PDM interface with 1bit DSM data from digital microphone

- **SARADC**
- 6bit SARADC, 4bit accuracy is guaranteed

- **Brown out reset**
- Programmable 3 threshold levels: 3.8v/2.7v/2.0v(default 2.0v)
- Optional BOD interrupt or reset

- **operating temperature:-20~+85 Degree**
- **Package: minimum SSOP 16pin for 2 dies package**

1.2 System diagram

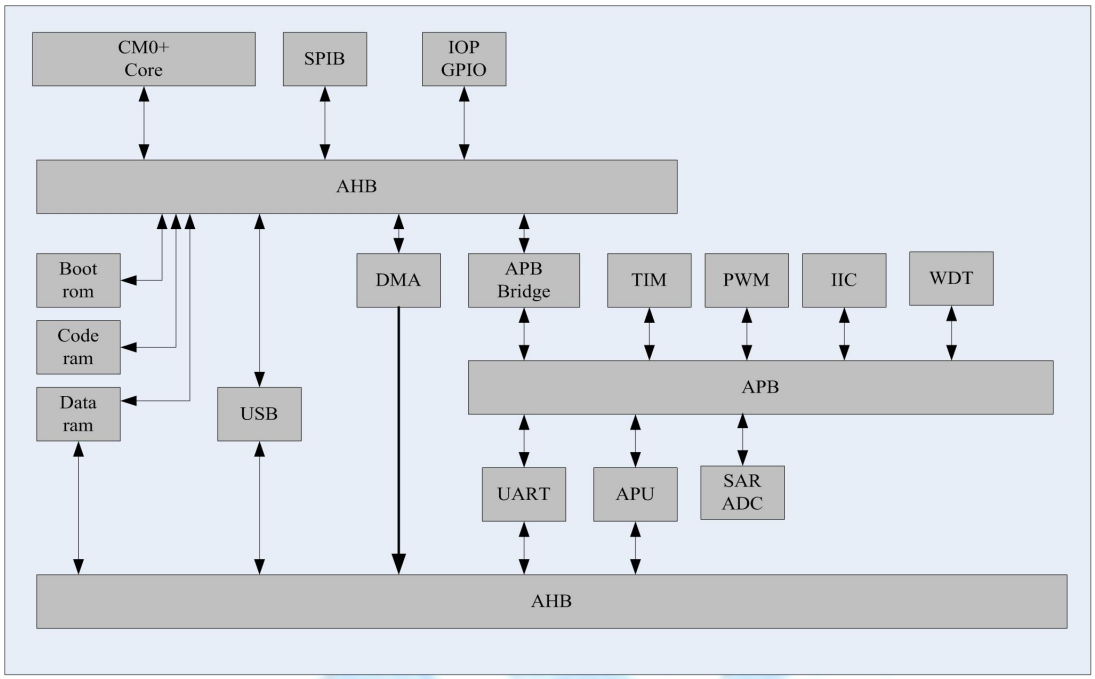

Figure 1. CJC2100 chip block diagram

2. PIN diagram

Figure 2. CJC2100 PIN diagram

3. PIN description

Table 1

36 Pin No.	PIN_NAME	IO_TYPE	COMMENT
DIGITAL PIN			
1	IIS_SDO	IO	IIS data out(SPDIF data out)
2	IIS_SDI	IO	IIS data in(SPDIF data in)
3	IIS_LRCK	IO	IIS channel select(SPDIF channel select)
4	IIS_SCLK	IO	IIS clock(SPDIF clock)
5	IIC_SDA	IO	IIC data(GPIO4)
6	IIC_SCL	IO	IIC clock(GPIO5)
26	SPIB_CLK	IO	SPI CLOCK for boot flash
25	SPIB_DI	IO	SPI data in for boot flash
23	SPIB_CS	IO	SPI chip select for boot flash
22	SPIB_DO	IO	SPI data out for boot flash
36	PWM0	IO	PWM channel 0(GPIO0/SPI_SCLK)
35	GPIO1	IO	GPIO1(SPI_CS)
34	GPIO2	IO	GPIO2(SPI_DI)
33	GPIO3	IO	GPIO3(SPI_DO)
27	UART_TX	IO	UART transmitter(GPIO6)
28	UART_RX	IO	UART receiver(GPIO7)
ANALOG PIN			
8	MIC/AGND	IO	Ground or MIC
9	AGND/MIC	IO	Ground or MIC
10	MIC_BIAS	IO	Mic Bias
11	MIC_SEL	IO	Select result of Ground or MIC
12	MIC_IN	IO	Mic input
18	HP_L	IO	left HP
20	HP_R	IO	right HP
19	HP_VCOM	IO	HP Voltage Reference
30	USB_DP	IO	USB DP
29	USB_DM	IO	USB DM
13	ADC_IN	IO	ADC input
POWER			
7	AVDD	POWER	power for analog except HP
15	AVSS	GROUND	ground for analog except HP
16	AVSS_HP	GROUND	ground for HP

17	AVDD_HP	POWER	power for HP
31	VDDIO0	GROUND	power for IO
21	VDDIO1	POWER	power for IO
32	DVDD18	POWER	Power for core
24	VDD33	GROUND	power for IO
RESET			
14	RESETB	I	Chip reset enable (low active)

4. Function description

4.1 CJC2100 address map

Figure 3 shows the address map of the practical peripherals.

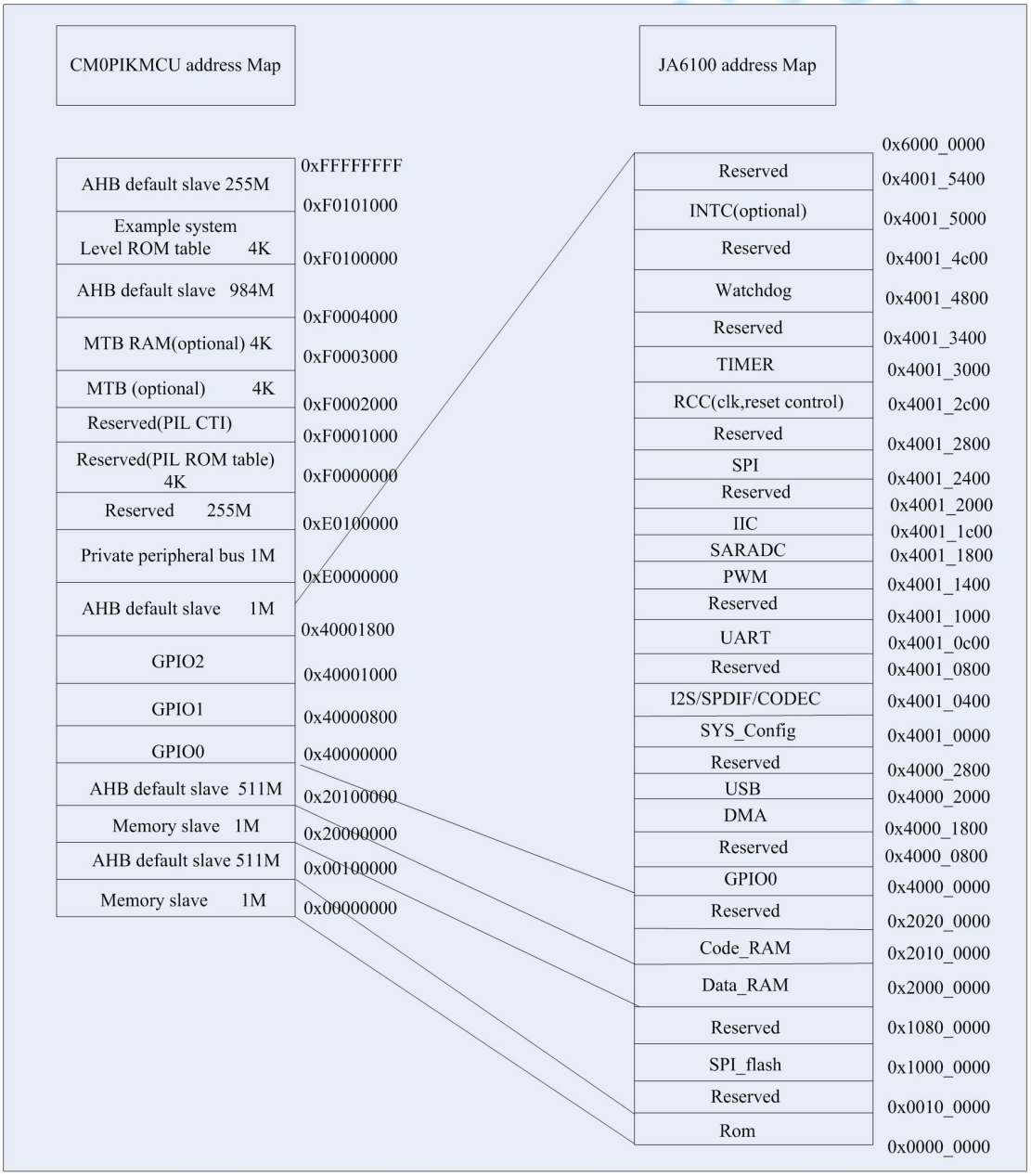

Figure 3 CJC2100 address map

4.2 BUS interface unit

CJC2100 chip integrate 2 AHB bus and 1 APB (AMBA protocol compatible). CPU core operates as AHB master in one AHB bus, and DMA controller operates as AHB master on other AHB bus. One AHB2APB Bridge is used for peripherals configuration.(see Figure1).

4.3 ROM

CJC2100 integrate 1KB boot ROM . When ISP is available, CPU Boots from internal boot ROM, Receives program code from UART bus and Stores in external flash. If normal mode is enabled, CPU Boots from internal boot ROM, Fetch program code from external SPI flash and Stores in internal SRAM, then, re-mapping memory configuration, boots from internal SRAM.

4.4 SRAM

The embedded high-speed SRAM is designed for both program code and scratchpad RAM. CJC2100 integrates one 16KB SRAM as the system program memory, 8KB SRAM as the data memory.

4.5 PLL and clock generation

PLL module generates system and block level clock from the internal 12MHZ RC-oscillator or an external 12MHZ crystal. CJC2100 chip contains two clock domains: one is system PLL clock source domain and another is Audio processor clock source domain. System PLL clock source domain includes CPU clock, AHB clock, APB clock based on the 12MHZ clock source; Audio processor clock source domain offer clock source for audio processor according to the audio sampling rate, if sampling rate is 8KHz, 16KHz, 32KHz, 48KHz and so on, the APU clock is 24.576MHZ and if sampling rate is 22KHZ, 44.1KHZ and so on, the APU clock is 22.5792MHZ. The APU PLL output can be configured via registers.

CJC2100 chip has one internal low-power oscillator to generate 10KHZ output. Figure 4 is CJC2100 clock diagram.

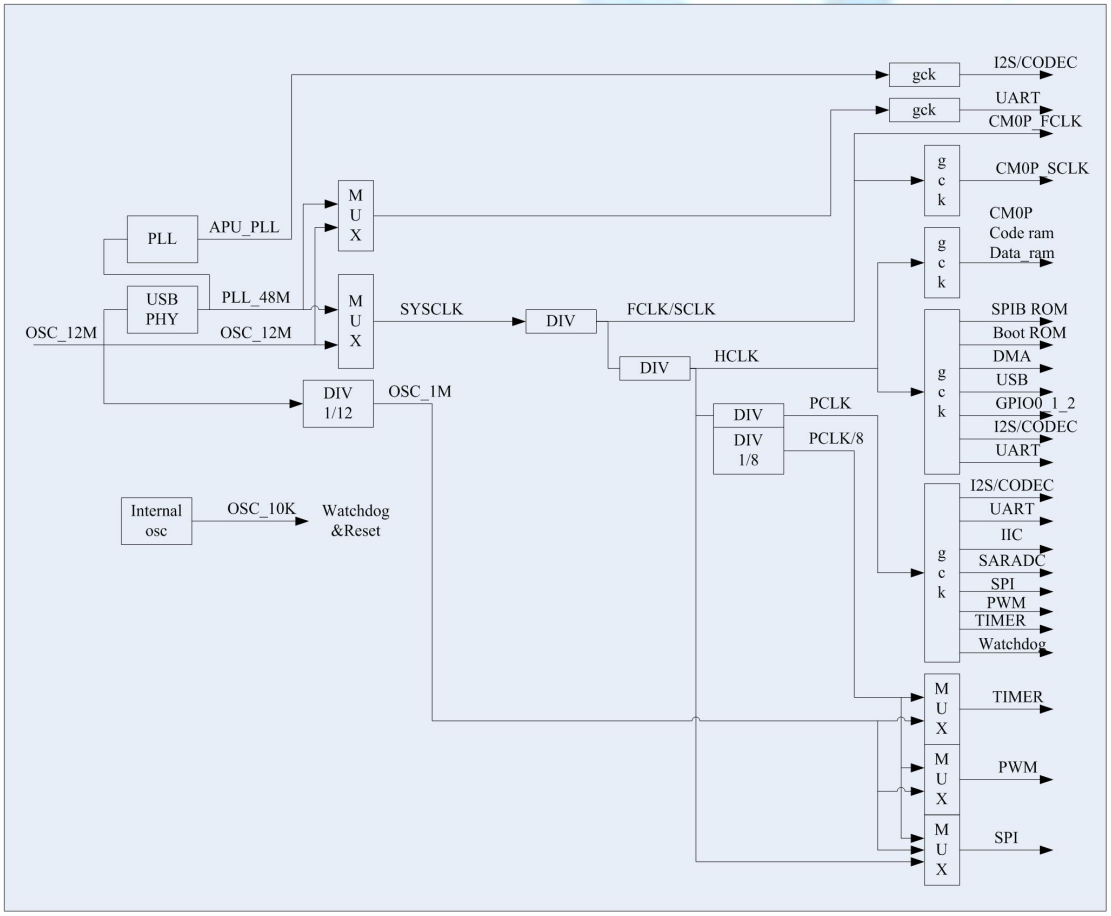

Figure 4 CJC2100 clock diagram

Table 2 shows system control register list (BaseAddr = 0x4001_0000).

Addr.	Name	Type	Default	Bit	Default
0x10	R4	R/W	0x0	[31:16]	Reserved
				[15:14]	uart_sel: uart clock select 0 fref 12MHz 1 pll 48MHz 2 codec clk Other reserved
				[13:12]	Wdt_sel: watchdog extclk select 0 pll fref 12MHz 1 10K osc Other reserved
				[11:9]	pclk_div: ratio of pclk/hclk 0: 1/1 1:1/2 2:1/4 3:1/8 4:1/16 other:reserved
				[8:6]	Hclk_div:ratio of hclk/core_clk 0:1/1 1:1/2 2:1/4 3:1/8 4:1/16 other:reserved
				[5:3]	Sys_div:Ratio of core_clk/system clock 0:1/1 1:1/2 2:1/4 3:1/8 4:1/16 other:reserved
				[2:0]	Sysclk_sel: ‘b00 12MHz osc ‘b01 48MHz sys PLL Other: reserved
0x14	R5	R/W	0x031270	[31:23]	Reserved
				[22:21]	Pll_n_sel
				[20:5]	Pll_sin16_in
				[4:0]	Pll_div_num
0x18	R6	R/W	0x0	[31:5]	Reserved
				[4:3]	Pll_icp_trim
				[2:1]	Pll_vco_trim
				[0]	Pll_vco_test

4.6 DMA

DMA is designed to enhance the system performance and reduce the processor-interrupt generation. The system efficiency is improved by employing the high-speed data transfers between the system and the device. The DMA controller provides up to 16 configurable channels (Figure 5) for memory-to-memory, memory-to-peripheral, peripheral-to-peripheral, and peripheral-to-memory transfers with a shared buffer. Figure 4 shows the DMA controller module block diagram.

DMA consists of 5 main blocks: AHB master interfaces, AHB slave interface, FIFO buffer, and DMA core. AHB master interface transfer data between the system and the DMA FIFO, system can configure the DMA controller through AHB slave interface, FIFO buffer provides the buffer between the source and the destination, and DMA core is configurable up to an 16-channel DMA engine, both source and destination are on AHB Bus, Each channel can be assigned with a group priority level, and the same group priority is serviced in the round-robin fashion. Figure 6 shows signal interface of DMA controller module.

DMA controller uses the 4-group priority and the round-robin scheme to select which channel to serve. Arbitration is based on the priority level of the channels. If the channels have the same priority level, the arbitration will then be based on the round robin scheme. Each channel has a 2-bit priority value associated with it. A value of 3 indicates the highest priority level and a value of 0 indicates the lowest priority.

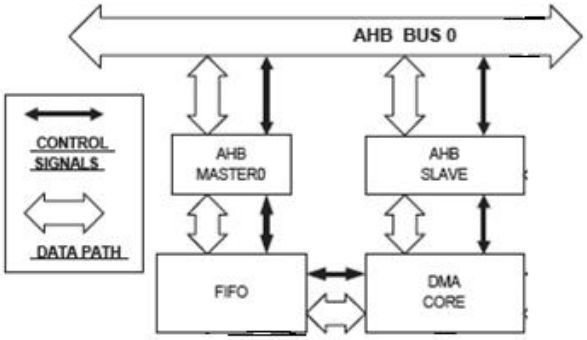

Figure 5 DMA controller module block diagram

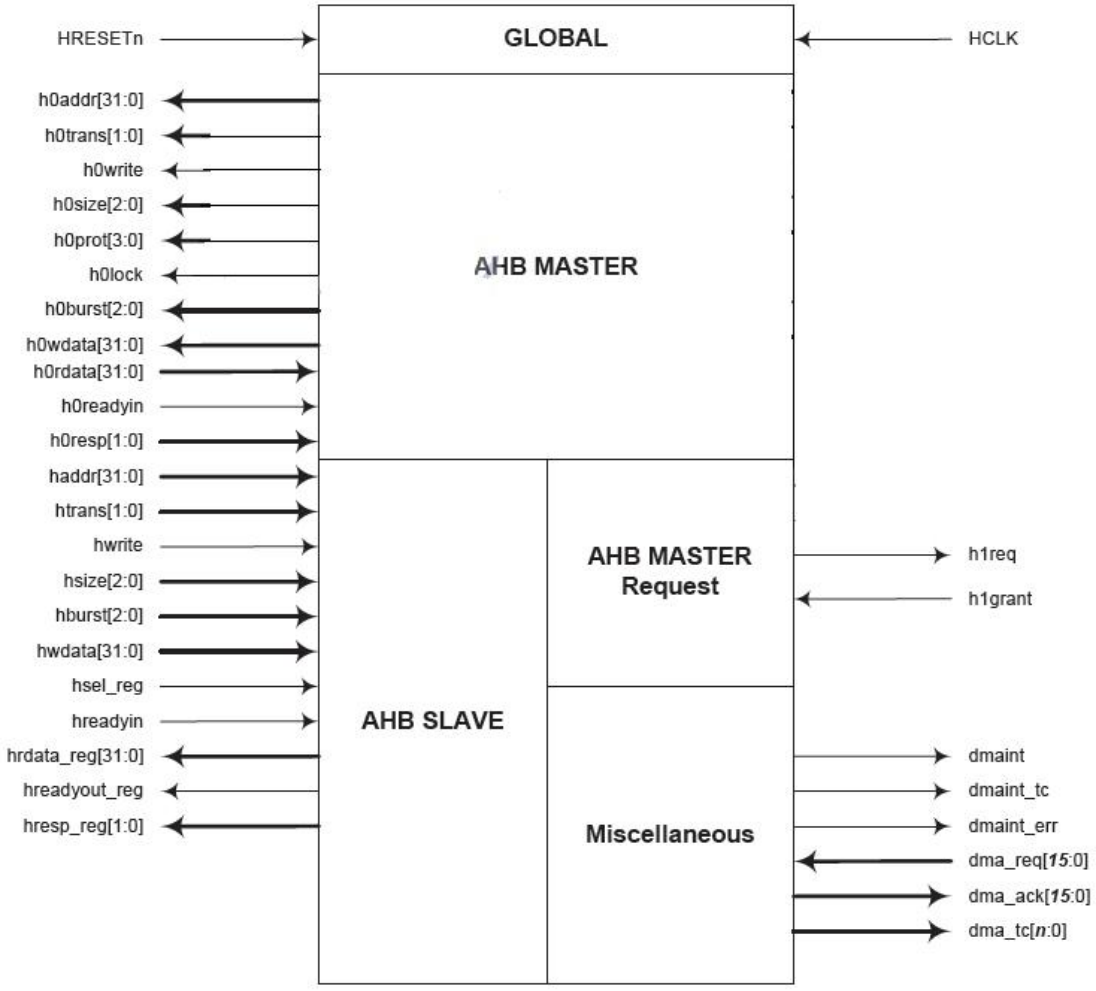

Figure 6 DMA controller module signal interface

DMA controller has one type work mode: hardware handshake mode.

Hardware handshake mode: when the channel wins the arbitration, the DMA controller will wait for the external DMA request to be asserted before starting the DMA transfer. Each time the DMA request is asserted, the controller transfers units equal to SRC_BURST_SIZE. When SRC_BURST_SIZE transfer is completed, the DMA controller asserts the acknowledge and then re-arbitrates among all DMA requests. After detecting the assertion of acknowledge, the external device should de-assert the DMA request to let the DMA controller de-assert acknowledge. After TOT_SIZE transfers have been done, the DMA controller asserts TC[0] (bit 0 of Terminal Count Status Register (TC)), dma_tc[0] and both dmaint_tc and dmaint interrupts (if not masked).

During the transfer, if the source or destination slave returns an ERROR response, the DMA will set the ERR bit and terminate the DMA transfer at once.

During the transfer, if the software sets the abort bit, after finishing SRC_BURST_SIZE transfers or TOT_SIZE transfers, the DMA controller will set the ABT bit and terminate the DMA transfer at once.

Figure 7 show the DMA hardware handshake mode protocol.

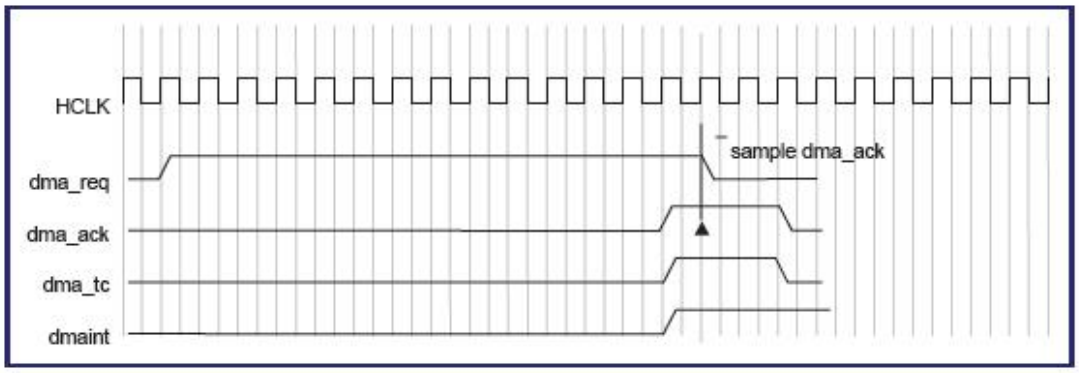

Figure 7 DMA hardware handshake mode protocol

Table 3 shows DMA control register list.

Name	Addr.	Width	Access	Description
Global registers				
INT	+0	8	RO	Interrupt status register
INT_TC	+4	8	RO	Interrupt for terminal count status register
INT_TC_CLR	+8	8	WO	Interrupt for terminal count clear register
INT_ERR/ABT	+c	32	RO	Interrupt for Error/Abort status register
INT_ERR/ABT_CLR	+10	32	WO	Interrupt for Error/Abort clear register
TC	+14	8	RO	Terminal count status register
ERR/ABT	+18	32	RO	Error/Abort status register
CH_EN	+1c	8	RO	Channel enable status register
CH_BUSY	+20	8	RO	Channel busy register status register
CSR	+24	8	RW	Main configuration status register
SYNC	+28	8	RW	Sync register
DMAC_REVISION	+30	32	RO	DMAC revision register
DMAC_FEATURE	+34	32	RO	DMAC feature register
Channel n registers				
Cn_CSR	+100+20*(n-1)	32	RW	Channel o control register
Cn_CFG	+104+20*(n-1)	32	RW	Channel o configuration register
Cn_SrcAddr	+108+20*(n-1)	32	RW	Channel o source register
Cn_DstAddr	+10C+20*(n-1)	32	RW	Channel o destination register
Cn_LLP	+110+20*(n-1)	32	RW	Channel o linked list pointer register
Cn_SIZE	+114+20*(n-1)	32	RW	Channel o transfer size register

Table 4 shows the DMA channel selection.

				Description
	DMA	Channel_0	UART TX	DMA req/ack for UART
		Channel_1	RX	
		Channel_2	I2S/SPDIF ADC_RX	DMA req/ack for codec
		Channel_3	/CODEC DAC_TX	
		Channel_4	IIS_RX	DMA req/ack for I2S/SPDIF
		Channel_5	IIS_TX	
		Channel_6	USB TX_1	DMA req/ack for USB
		Channel_7	TX_2	
		Channel_8	TX_3	
		Channel_9	TX_4	
		Channel_10	RX_1	
		Channel_11	RX_2	
		Channel_12	RX_3	
		Channel_13	RX_4	
		Channel_14	Reserved	
		Channel_15		

Table 4 shows the DMA channel distribution, 14 channels DMA channel are used. UART_TX and UART_RX will occupy two dedicated DMA channel, ADC_RX, DAC_TX, IIS_RX and IIS_TX will occupy four dedicated DMA channel, USB will occupy eight dedicated DMA channel.

4.7 Interrupt controller

Interrupt controller module has NVIC mode to communicate with CPU. It supports 32 NVIC priority level interrupt inputs. Provides 0(max.)~7(min.) configurable priority levels for each NVIC interrupt input.

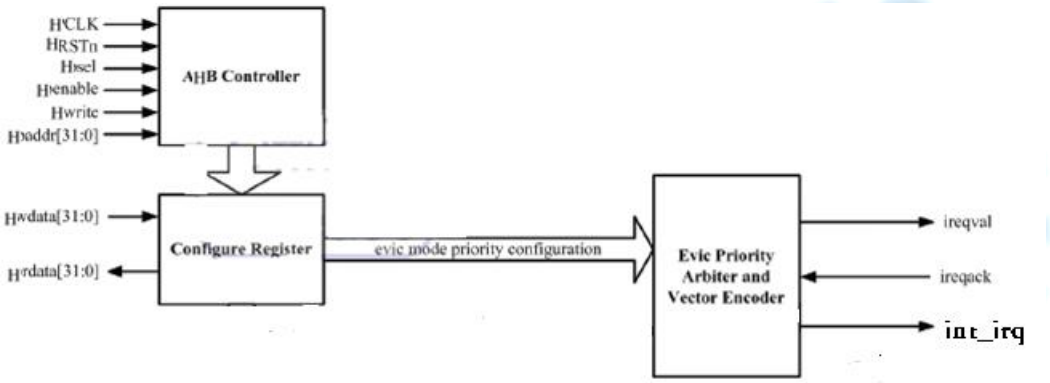

Figure 8 NVIC module block diagram

Figure 8 is NVIC module block diagram, it include AHB interface, control register, priority arbiter and vector encoder.

Figure 9 shows this block interface.

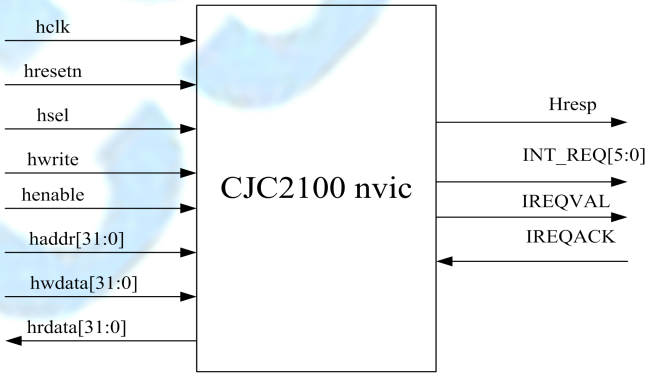

Figure 9 interrupt contrller module interface

Table 5 Summary of NVIC controller registers.

Name	Addr.	Width	Access	Description	default
Global registers					
NVIC_ISER	0xE000E100	32	RW	SETENA[31:0] Enables/disables the interrupt request lines:	0x00

				0=interrupt disabled 1=interrupt enabled. Enables interrupt request to processor.	
NVIC_ICER	0xE000E180	32	WO	CLRENA[31:0] 1=clear the corresponding bit in the interrupt register. 0=no effect	0x00
NVIC_ISPR	0xE000E200	32		SETPEND[31:0]	
NVIC_ICPR	0xE000E280	32		CLRPEND[31:0]	
NVIC_IPR0	0xE000E400	32		[31:24] VNIC3 priority; [23:16] VNIC2 priority; [15:8] VNIC1 priority; [7:0] VNIC0 priority	
NVIC_IPR1	0xE000E42F	32		[31:24] VNIC7 priority; [23:16] VNIC6 priority; [15:8] VNIC5 priority; [7:0] VNIC4 priority	
NVIC_IPR2	0xE000E44F	32		[31:24] VNIC11 priority; [23:16] VNIC10 priority; [15:8] VNIC9 priority; [7:0] VNIC8 priority	
NVIC_IPR3	0xE000E46F	32		[31:24] VNIC15 priority; [23:16] VNIC14 priority; [15:8] VNIC13 priority; [7:0] VNIC12 priority	
NVIC_IPR4	0xE000E48F	32		[31:24] VNIC19 priority; [23:16] VNIC18 priority; [15:8] VNIC17 priority; [7:0] VNIC16 priority	
NVIC_IPR5	0xE000E4AF	32		[31:24] VNIC23 priority; [23:16] VNIC22 priority; [15:8] VNIC21 priority; [7:0] VNIC20 priority	
NVIC_IPR6	0xE000E4CF	32		[31:24] VNIC27 priority; [23:16] VNIC26 priority; [15:8] VNIC25 priority; [7:0] VNIC24 priority	
NVIC_IPR7	0xE000E4EF	32		[31:24] VNIC31 priority; [23:16] VNIC30 priority; [15:8] VNIC29 priority; [7:0] VNIC28 priority	

Table 6 show CJC2100 chip interrupt distribution.

			Signal	
		IRQ_0	Sys_gpio0_int	Interrupt for GPIO
		IRQ_1	Sys_tm0_int	Interrupt for timer0
		IRQ_2	Sys_tm1_int	Interrupt for timer1
		IRQ_3	Sys_wdt_int	Interrupt for watchdog
		IRQ_4	Sys_uart_int	Interrupt for uart
	NVIC	IRQ_5	Sys_spi_int	Interrupt for spi

	ID	IRQ_6	Sys_iic_int	Interrupt for iic
		IRQ_7	Sys_dmac_int	Interrupt for dmac
		IRQ_8	Sys_saradc_int	Interrupt for saradc
		IRQ_9	Sys_usb_mc_nint	Interrupt for usb
		IRQ_10	Sys_usb_sof_int	Interrupt for usb frame sync pulse
		IRQ_11	Sys_dma_nint_usb_w	Interrupt for usb_dma
		IRQ_12		
		IRQ_13		
		IRQ_14		
		IRQ_15		
		IRQ_16		
		IRQ_17		
		IRQ_18		
		IRQ_19		
		IRQ_20		
		IRQ_21		
		IRQ_22		
		IRQ_23		
		IRQ_24		
		IRQ_25		
		IRQ_26		
		IRQ_27		
		IRQ_28		
		IRQ_29		
		IRQ_30		
		IRQ_31		
		IRQ_32		

Figure 10 show the interrupt REQ/ACK timing sequence diagram.

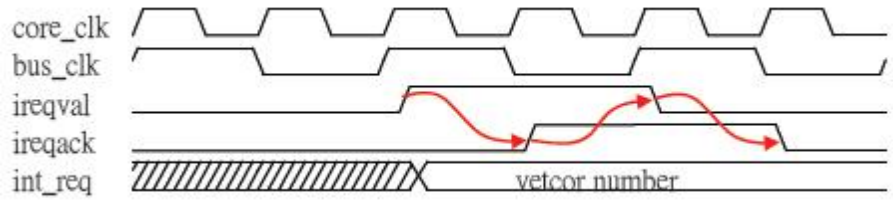

Figure 10 the interrupt REQ/ACK timing sequence diagram.

4.8 GPIO

GPIO controller is an AHB bus device communicates with CM0+ core. Each GPIO can be programmed as an input or output. It is used to input/output data from the system and device.

This GPIO can also be an interrupt input.

The GPIO provides up to 32 programmable I/O ports and each port can be independently programmed.

Table 7 summary of general purpose I/O registers (BaseAddr = 0x4000_0000)

Addr.	Name	Type	Default	Description
0x000~0x3FF	GPIODATA	R/W	0x0	Reads the value of the GPIOIN pins, or sets the value driven onto GPIOUT pins.
0x400	GPIODIR	R/W	0x0	GPIO direction register 0:Input 1:Output
0x410	GPIOIE	R/W	0x0	GPIO interrupt enable register 0:Pin interrupt is disabled 1:Pin interrupt is enabled

When GPIOIE enable, interrupt is open and low to high or high to low of GPIO can arose interrupt. Interrupt time has one hclk clock.

Figure 11 shows GPIO module block diagram.

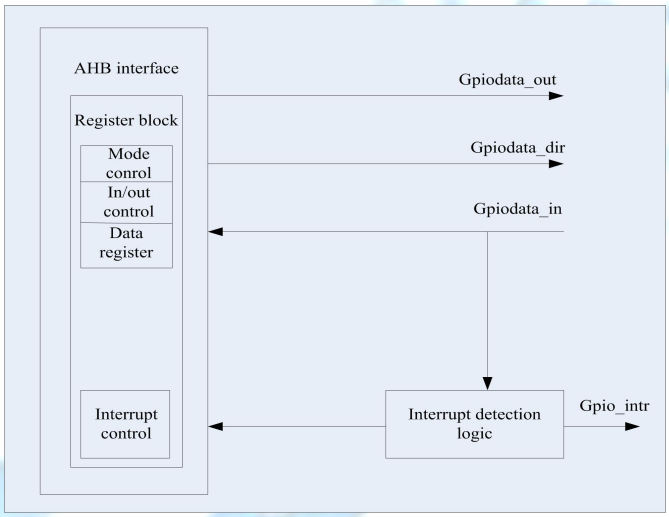

Figure 11 GPIO module block diagram

4.9 SARADC

SARADC is accessible by CM0+ core via APB bus. This peripheral is used sampling the external sensor, voltage signal, transfer them to digital data by SARADC block, update the status register and interrupt signal, exchange data with CJC2100 processor.

The SARADC supports four external analog input signal come from sensor, mechanism key etc, the sample time is about 400HZ and sample sequence is one by one, the SARADC transfer result is store in internal register. After finishing one round, interrupt signal will generate, processor will respond this interrupt and enter into ISR.

SARADC module block diagram is shown as Figure 12, it include SARADC unit, decimation filter and ADC control unit. SARADC unit is a 3.3V power supply analog module, co-work with decimation filter to implement the analog-to-digital transfer. ADC control include module timing generation, register control, interrupt generation and APB bus wrapper.

Figure 13 is SARADC module interface and table 8 show this module register list.

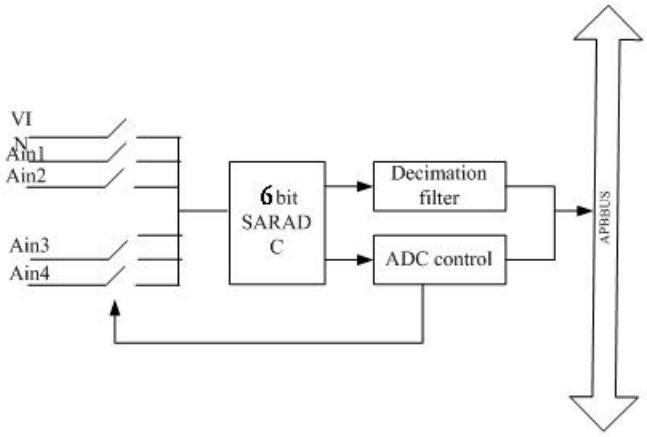

Figure 12 SARADC module block diagram

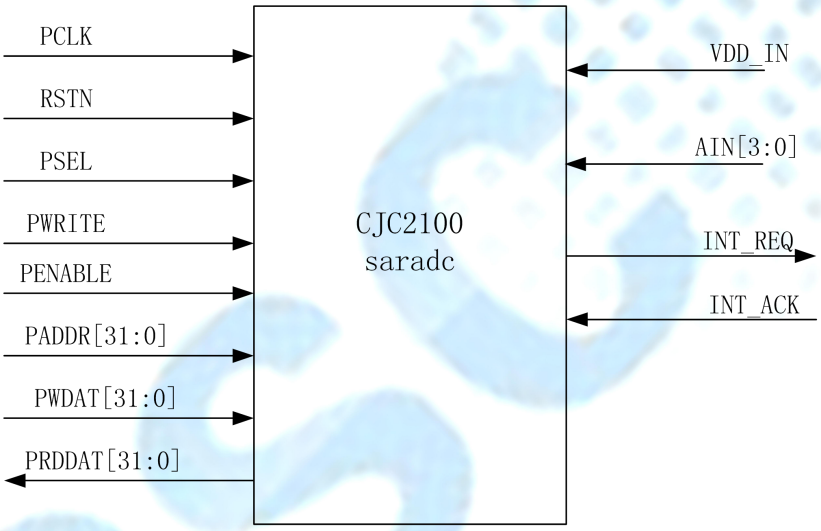

Figure 13 SARADC module interface diagram

Table 8 SARADC module register list (BaseAddr = 0x4001_1800).

Offset	Type	Name	Bit	description	Default
0X00	R/W	ADC_CTRL	[31:18]	TRIG_DELAY: delay from the end of one sample to the start of the next sample. One ADC sample cycle include	0x0

				0: 1 saradc_clk delay N:N+1 saradc_clk delay So, if TRIG_DELAY is set to N, then one ADC sample cycle includes (N+16) sarade_clk cycles	
			[17:14]	Reserved	
			[13:8]	PCLK_DIV: SARADC clock is 4*(pclk_div+1) division of hclk. (Make sure the division of hclk is less than the highest frequency of SARADC workable clock)	
			[7:2]	Reserved	
			[1]	INTP_EN: channel interrupt enable 1: enable 0:disable	
			[0]	SARADC_EN: channel enable 1:enable 0:disable	
0X04	R/W	DAT_STATUS	[31:1]	Reserved	0x0
			[0]	ADC_READY: ADC is ready 1:ready 0: not ready Write 1 to this bit will clear the according bits. And at the same time, the interrupt and status will be cleared.	
0X08	R/W	DAT_RESULT	[31:8]	Reserved	0x0
			[7:0]	SARADC result	

4.10 CODEC/IIS/SPDIF

CJC2100 audio processor can be accessed via AHB bus or APB bus. CM0+ configures audio codec register by Ahb2apb Bridge and DAM translates data with codec.

CODEC receives data from IIS/SPDIF interface and DMA fifo data and sends to DAC module. ADC module disposes analogy data (3bit dsm data) and digital mic data(1bit PDM data) and sends digital data to IIS/SPDIF interface output and DMA fifo. Figure 14 shows the block diagram. IIS interface and SPDIF interface are Selected by configure FFMT register and they don't occur at one time.

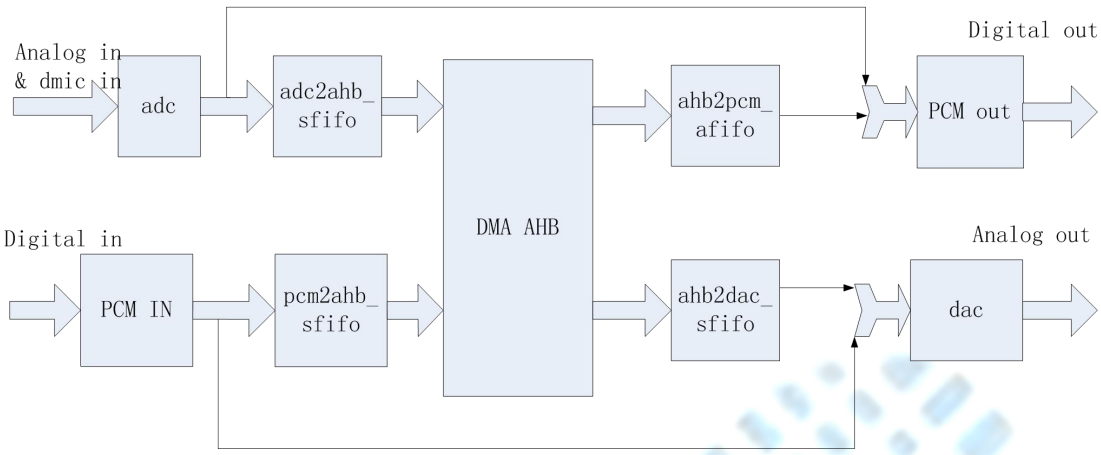

Figure 14 APU block diagram

Table 9 lists this module interface pin.

Signal Name	***Width***	***Dir***	***Description***
AHB interface part			
hclk	1	I	AHB interface module
hreset_n	1	I	
haddr	[31:0]	I	
hburst	[2:0]	I	
hlock	1	I	
hprot	[3:0]	I	
hwdata	[31:0]	I	
hsize	[2:0]	I	
htrans	[1:0]	I	
hsel	1	I	
hready_in	1	I	
hwrite	1	I	
hready_out	1	O	
hresp	1	O	
hrdata	[31:0]	O	
APB interface part			
pclk	1	I	APB interface module
paddr	[31:0]	I	
psel	1	I	
penable	1	I	
pwrite	1	I	
pwdata	[31:0]	I	
prdata	[31:0]	O	
codec_clk	1	I	主时钟输入
bclk_in	1	I	I2S 的位时钟输入

lrc_in	1	I	I2S 的左右通道选择输入
dacdat_in	1	I	I2S/SPDIF 数据输入
codec_dlout	[2:0]	I	3bit 的 dsm 数据输入
mic_data_in	1	I	数字 mic 的数据输入
bclk_out	1	O	I2S 的位时钟输出
bclk_oen	1	O	I2S 的位时钟输出使能
lrc_oen	1	O	I2S 的左右通道选择使能
lrc_out	1	O	I2S 的左右通道选择输出
adcdac_out	1	O	I2S/SPDIF 数据输出
mic_clk	1	O	数字 mic 时钟输出

Table10 list APU module register. (BaseAddr=0x4001_0400)

Register	Bit	Name	Description	Default value
0x00 Input volume	[31]	dacdiv2	DAC 6dB attenuate enable 0 = disabled (0dB) 1= -6dB enabled	0x40000757
	[30]	dacmu	Digital soft mute 1=mute 0= no mute (signal active)	
	[29:28]	deemph	De-emphasis control 11 = 48kHz sample rate 10 = 44.1kHz sample rate 01 = 32kHz sample rate 00 = No De-emphasis	
	[27]	ana_adcdiv2	ADC 6dB attenuate enable 0 = disable(0dB) 1 = -6dB enabled	
	[26]	dmic_adcdiv2	dmic 6dB attenuate enable 0 = disable(0dB) 1 = -6dB enabled	
	[25]	hpor_ana	ADC channel store dc offset when high-pass filter disabled 1 = store offset 0 = clear offset	
	[24]	hpor_dmic	dmic channel store dc offset when high-pass filter disabled 1 = store offset 0 = clear offset	
	[23]		***Reserved***	
	[22]	adcpol_ana	0 = adc polarity not inverted	

		1 = adc polarity invert	
[21:20]	adcpol_dmic	00 = dmic polarity not inverted 01 = dmic1 polarity invert 10 = dmic2 polarity invert 11 = dmic1 & dmic2 polarity inverted	
[19:18]		***Reserved***	
[17]	adchpd_ana	Adchpd_ana determine high-pass filter behavior 1'b0 = ADC hpf on 1'b1 = ADC hpf off	
[16]	adchpd_dmic	Adchpd_dmic and hpflren_dmic together determine high-pass filter behavior [Hpflren_dmic ,Adchpd_dmic] 2'b00 = dmic1 & dmic2 hpf on 2'b01 = dmic1 & dmic2 hpf off 2'b10 = dmic1 hpf on & dmic2 hpf off 2'b11 = dmic1 hpf off & dmic2 hpf on	
[15:14]		***Reserved***	
[13:12]	lmicboost	Microphone Gain Boost 00 = Boost off (bypassed) 01 = 13dB boost 10 = 20dB boost 11 = 29dB boost	
[11]		***Reserved***	
[10:8]	maxgain	Set Maximum Gain of PGA 111 = +12dB 110 = +6dB ….(-6dB steps) 001=-24dB 000=-30dB	
[7]	livu	Volume update 1=Update gain 0=Store LINVOL in intermediate latch(no gain change)	
[6]	linmute	Left channel analog input mute. 1 = Enable Mute 0 = Disable Mute Note: LIVU must be set to un-mute.	
[5]	lzcen	Left channel zero cross detector. 1 = change gain on zero cross only 0 = change gain immediately	
[4:0]	linvol	Left channel input volume control 11111 = +12dB	

			11110 = +10.5dB … 10111=0dB ..1.5dB steps down to 00000 = -34.5dB	
0x04 LOUT1 and ROUT1 volume control	[31:21]		***Reserved***	0x79079
	[20]	ro1vu	Right volume update 0=store rout1vol in intermediate latch(no gain change) 1=update left and right channel gains (left= intermediate latch, right = rout1vol)	
	[19]	ro1zc	Right zero cross enable 1 = Change gain on zero cross only 0 = Change gain immediately	
	[18:12]	rout1vol	ROUT1 Volume 1111111 = +6dB …(80 steps) 0110000 = -67dB 0111111 to 0000000 = Analogue mute	
	[11:9]		***Reserved***	
	[8]	lo1vu	Left volume update 0=store LOUT1VOL in intermediate latch(no gain change) 1=update left and right channel gains(left=LOUT1VOL,right=intermediate latch)	
	[7]	lo1zc	Left zero cross enable 1 = Change gain on zero cross only 0 = Change gain immediately	
	[6:0]	lout1vol	LOUT1 Volume 1111111 = +6dB …(80 steps) 0110000 = -67dB 0111111 to 0000000 = Analogue mute	
0x08 ADC & DAC control	[31]	ffmt	Pcm mode 1=spdif 0=IIS	0x1000a
	[30]	align_24bit_iis	24bit USB data input 1= IIS data from USB 0= IIS data from AHB bus or adc	
	[29]	align_24bit_dac	24bit USB data input 1= dac data from USB 0= dac data from AHB bus or IIS	

	[28:27]		***Reserved***
	[26:24]	sr_adc	ADC sample rate control 3'b000 = 8k 3'b001 = 8.0182k 3'b010 = 12k 3'b110 = 11.025k 3'b011 = 16k
	[23:19]		***Reserved***
	[18:17]	bcm_dac	DAC BCLK frequency 00=BCM function disabled 01=MCLK/4 10=MCLK/8 11=MCLK/16
	[16]	clkdiv2_dac	DAC master clock divide by 2 1=MCLK is divided by 2 0=MCLK is not divided
	[15:12]	sr_dac	DAC sample rate control MCLK = 12.288M MCLK=11.2896M 4'b0011 = 8k 4'b1011=8.0182k 4'b0100 = 12k 4'b1100=11.025k 4'b0101 = 16k 4'b1101=22.05k 4'b1110 = 24k 4'b1000=44.1k 4'b0110 = 32k 4'b1111=88.2k 4'b0000 = 48k 4'b1010=176.4k 4'b0111 = 96k 4'b0010=192k
	[11]	right_in	Valid when adc2pcm =1 & dmic_in=0 1 = ana data output from i2s right channel 0 = ana data output from i2s left channel
	[10]	dmic_in	Valid when adc2pcm =1 IIS/SPDIF data source from adc data/dmic data 1= dmic data 0= ana data
	[9]	adc2pcm	IIS/SPDIF data source 1 = adc data 0 = fifo data
	[8]	pcm2dac	DAC data source 1 = IIS/SPDIF data 0 = fifo data
	[7]	bclkinv	BCLK invert bit(for master and slave modes) 0 = BCLK not inverted 1 = BCLK inverted

	[6]	ms	Master/Slave mode control 1 = Enable Master mode 0 = Enable slave mode	
	[5]	lrswap	Left/Right channel swap 1 = swap left and right DAC data in audio interface 0 = output left and right data as normal	
	[4]	lrp	Right, left and i2s modes – LRCLK polarity 1 = invert LRCLK polarity 0 = normal LRCLK polarity	
	[3:2]	wl	Audio Data Word Length 11 = 32 bits 10 = 24 bits 01 = 20 bits 00 = 16 bits	
	[1:0]	format	Audio Data Format Select 11 = DSP Mode 10 = I2S Format 01 = Left justified 00 = reserved(do not use this setting)	
0x0c DAC volume	[31:16]		***Reserved***	0xffff
	[15:8]	RDACVOL	Right DAC Digital volume control similar to LDACVOL	
	[7:0]	LDACVOL	Left DAC Digital Volume Control 0000 0000 = Digital mute 0000 0001 = -127dB 0000 0010 = -126.5dB … 0.5dB steps up to 1111 1111 = 0dB	
0x10 Additional control	[31]	vroi	VREF to analogue output resistance 0:1.5k 1:40k	0xc3
	[30:29]	dmonomix	DAC mono mix 00:stereo 01:mono(L+R)/2into DACL,'0'into DACR 10:mono(L+R)/2into DACR,'0'into DACL 11: mono(L+R)/2into DACL and DACR	
	[28]	dacinv	DAC phase invert 0:non-inverted 1:inverted	
	[27]	toen	Timeout enable 0:timeout disable 1:timeout enable	

	[26]	hpflren_dmic	Adchpd_dmic and hpflren_dmic together determine high-pass filter behaviour	
	[25]	reg_tri	Tristates ADCDATA and switches LRC and BCLK to inputs 0=adcdat is an output,lrc and bclk are inputs or outputs 1=adcdat is tristated, lrc and bclk are inputs	
	[24]		***Reserved***	
	[23:22]	codec_mic	Mic signal to dac mixer volume 00=-6dB 01=-9dB 10=-12dB 11=-15dB	
	[21]	dacosr	DAC oversample rate select 1=64x(lowest power) 0=128x(best SNR)	
	[20]	micvu	MIC volume update 0 = store micvol in intermediate latch(no gain change) 1 = update left and right channel gains	
	[19:12]	micvol	Dig mic digital volume control 00000000=digital mute 00000001=-97dB 00000010=-96.5dB …0.5dB steps up to 11111111=+30dB	
	[11:9]		***Reserved***	
	[8]	avu	ADC volume update 0 = store adcvol in intermediate latch(no gain change) 1 = update left and right channel gains	
	[7:0]	adcvol	ADC digital volume control 00000000=digital mute 00000001=-97dB 00000010=-96.5dB …0.5dB steps up to 11111111=+30dB	
0x14 Power management	[31]	codec_rstn_scf	DAC 模块 SCF 的 reset 信号 0=reset 1=正常工作	0x0
	[30]	codec_en_vmid	VMID enable 0=power down 1=power up	
	[29]	codec_en_ibias	VREF 0=power down 1=power up	
	[28]	codec_en_micb	Micbias buffer 0=power down 1=power up	
	[27]	codec_en_linein	Linein buffer	

			0=power down 1=power up	
	[26]	codec_en_vref	HP buffer 0=power down 1=power up	
	[25]	codec_mic2o	Mic to dac mixer enable 0=disable 1=enable	
	[24]	codec_en_rhp	R_Channel HP enable 0=disable 1=enable	
	[23:13]		***reserved***	
	[12]	codec_en_lhp	L_Channel HP enable 0=disable 1=enable	
	[11]	codec_en_hp_vmid	Headphone vmid enable 0=power down 1=power up	
	[10]	dmic2_en	dmic2 enable 0=power down 1=power up	
	[9]	dmic1_en	dmic1 enable 0=power down 1=power up	
	[8]	digenb	Master clock disable 0=master clock enabled 1=master clock disabled	
	[7]		***reserved***	
	[6]	iis_clr_reg	I2S reset 0=reset 1=正常工作	
	[5]	adc_clr_reg	ADC reset 0=reset 1=正常工作	
	[4]	dac_clr_reg	DAC reset 0=reset 1=正常工作	
	[3]	codec_en_pga	PGA enable 0=power down 1=power up	
	[2]	dacr_en	DAC right enable 0=power down 1=power up	
	[1]	dacl_en	DAC left enable 0=power down 1=power up	
	[0]	ana_en	ADC enable 0=power down 1=power up	
0x18 Fifo depth control	[31:30]		***Reserved***	0x3370b
	[29:25]	codec_iplus	0 = normal ibias current 1 = adding ibias current codec_iplus[5] HP ibias current codec_iplus[4] R-dac channel ibias current codec_iplus[3] L-dac channel ibias current codec_iplus[2] adc ibias current(else op1st) codec_iplus[1] adc op1st ibias current codec_iplus[0] pga ibias current	

	[24:19]		***Reserved***	
	[18:16]	iis_tx_trig	IIS tx fifo depth，值只能设置为 0~6	
	[15]		***reserved***	
	[14:12]	iis_rx_trig	IIS rx fifo depth，值只能设置为 0~6	
	[11:8]	dac_tx_trig	DAC tx fifo depth,值只能设置为 0~14	
	[7:5]		***reserved***	
	[4:0]	adc_rx_trig	ADC rx fifo depth,值只能设置为 0~23	
0x1c Test mode	[31:9]		***Reserved***	0x00
	[8]	adc_o_mu	Disable adc mute 0=no mute 1=mute	
	[7]	bclk_gate	Enable BCLK clock gating	
	[6]	chnl_sel	Left/right channel select 0=left 1=right	
	[5]	clk_sel	Output adc_clk, dac_clk to external	
	[4]	dmic_ana_sel	adc_dsm output select from dmic ana ana	
	[3]	dac_dsm_oen	Output dac_dsm to external	
	[2]	dac_dsm_ien	Input dac_dsm from external 0=正常工作 1=测试模式	
	[1]	adc_dsm_oen	Output adc_dsm to external 0=正常模式 1=测试模式	
	[0]	adc_dsm_ien	Input adc_dsm from external 0=正常模式 1=测试模式	
0x20	[31:0]	adc_rx_fifo	DMA 读数据，只读寄存器	0x00
0x24	[31:0]	dac_tx_fifo	DMA 写数据，只写寄存器	0x00
0x28	[31:0]	iis_rx_fifo	DMA 读数据，只读寄存器	0x00
0x2c	[31:0]	iis_tx_fifo	DMA 写数据，只写寄存器	0x00

4.10.1 Digital microphone input

Digital microphone mainly consists of a ∑-Δ ADC modulator and allows for the pulse density modulated (PDM) output of two microphones to be time multiplexed on a single data line using a single clock.

Figure 15 shows the block diagram of digital microphone.

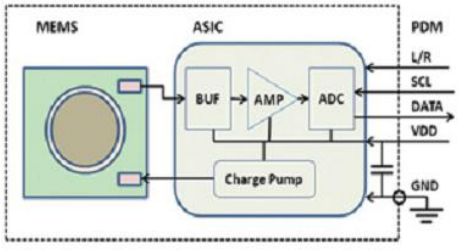

Figure15 digital microphone block diagram

Digital microphone has five pin, see Table 10.

Table 10 Pin function diagram

Pin No.	Mnemonic	Description
1	CLK	Clock Input to Microphone.
2	L/R SELECT	Left Channel or Right Channel Select. DATA1 (Right): L/R SELECT tied to GND. DATA2 (Left): L/R SELECT pulled to V_{DD}.
3	GND	Ground.
4	V_{DD}	Power Supply.
5	DATA	Digital Output Signal (DATA1, DATA2).

Digital microphone must a VDD power supply and external clock supply. The frequency of CLK ranges from 1.024MHz to 3.074MHz. When VDD and CLK are supplied, it's state from idle to work and output PDM data. Singleness Digital microphone output right channel data by select L/R select to GND, and it can output left channel data by select L/R SELECT to VDD. Figure 16 shows the timing diagram. DATA1 and DATA2 are singleness Digital microphone output. DATA is two digital microphone output.

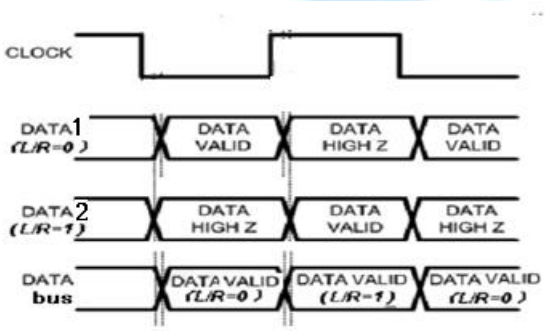

Figure 16 The timing diagram of Digital microphone

4.10.2 IIS interface

The IIS interface supports ADCDAT output, DMA FIFO output, DMA FIFO input and DACDAT input. It supports several data format such as IIS, Left_justified, DSP, Right_justified, and it supports 16bit, 20bit, 24bit, 32bit word length. Figure17 to figure 20 show the timing sequence example for different format.

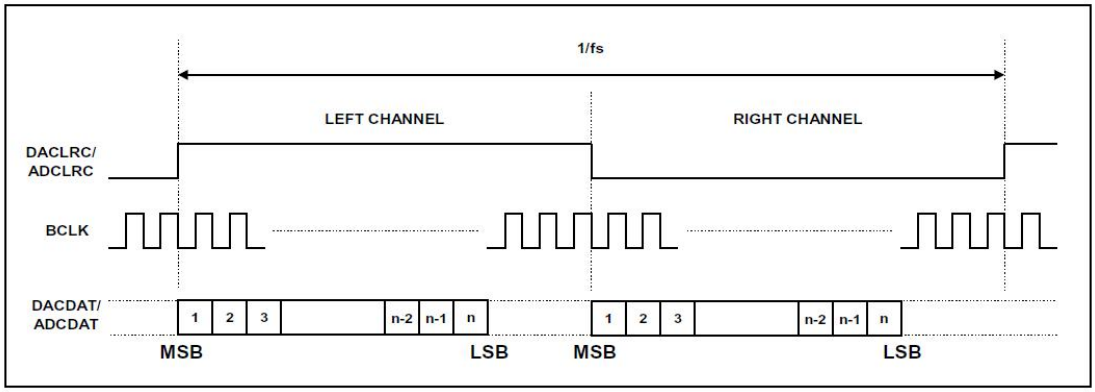

Figure 17 Left Justified Audio Interface (assuming n-bit word length)

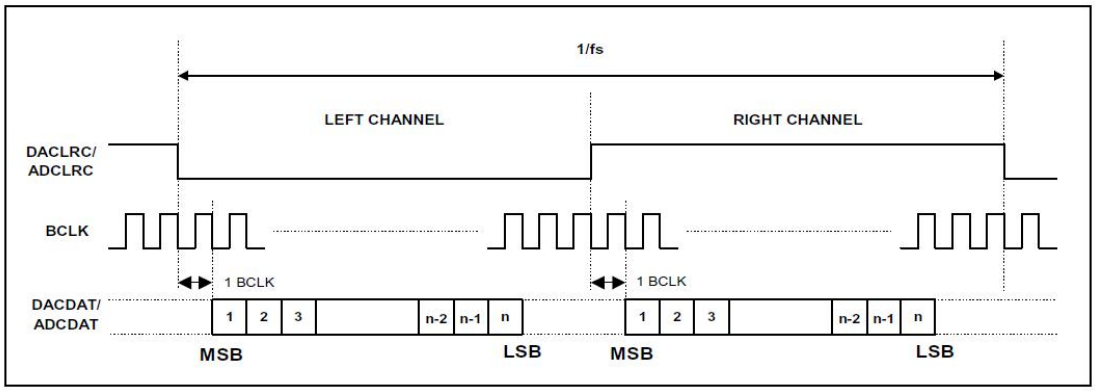

Figure 18 I2S Justified Audio Interface (assuming n-bit word length)

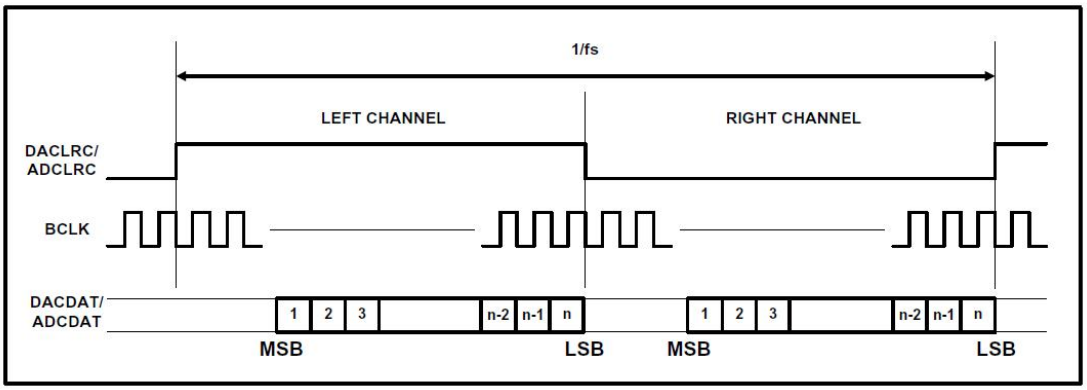

Figure 19 Right Justified Audio Interface (assuming n-bit word length)

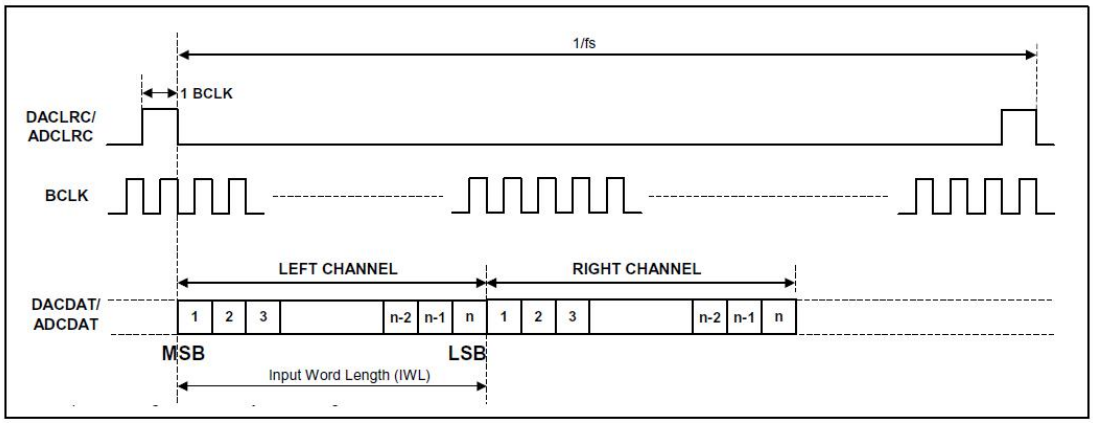

Figure 20 DSP Audio Interface (assuming n-bit word length)

4.10.3 SPDIF interface

The SPDIF is a point-to-point protocol for serial transmission of digital audio through a single transmission line. It provides two channels for audio data, a method for communicating control information and some error detection capabilities. The control information is transmitted as one bit per sample and accumulates in a block structure. The data is bi-phase encoded, which enables the receiver to extract a clock from the data. Coding violations, defined as preambles, are used to identify sample and block boundaries.

The SPDIF format is designed to transmit audio data. Each sample of audio data is packetized into a 32-bit sub-frame (see Figure21) that includes additional information such as parity, validity, and user-definable bits. A frame is composed of two

sub-frames, a block consists of 192 frames (see Figure 22). The first sub-frame normally starts with preamble X. However the preamble changes to preamble Z once every 192 frmaes. This defines the block structure used to organize the channel status information. The second sub-frame always starts with preamble Y. Figure 23 shows preamble X, preamble Y and preamble Z.

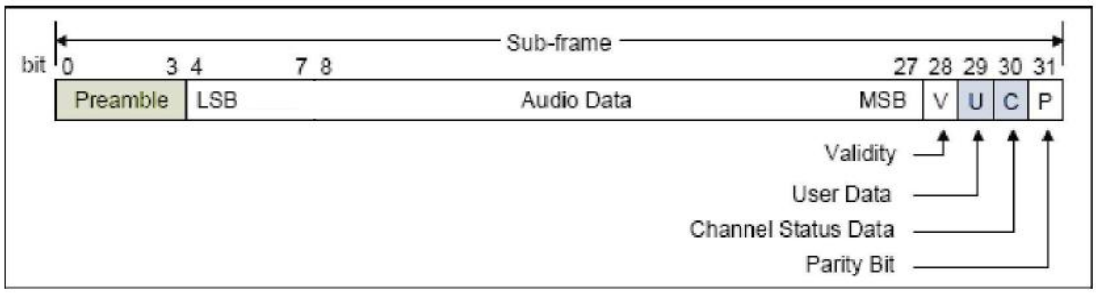

Figure 21 sub-frame structure

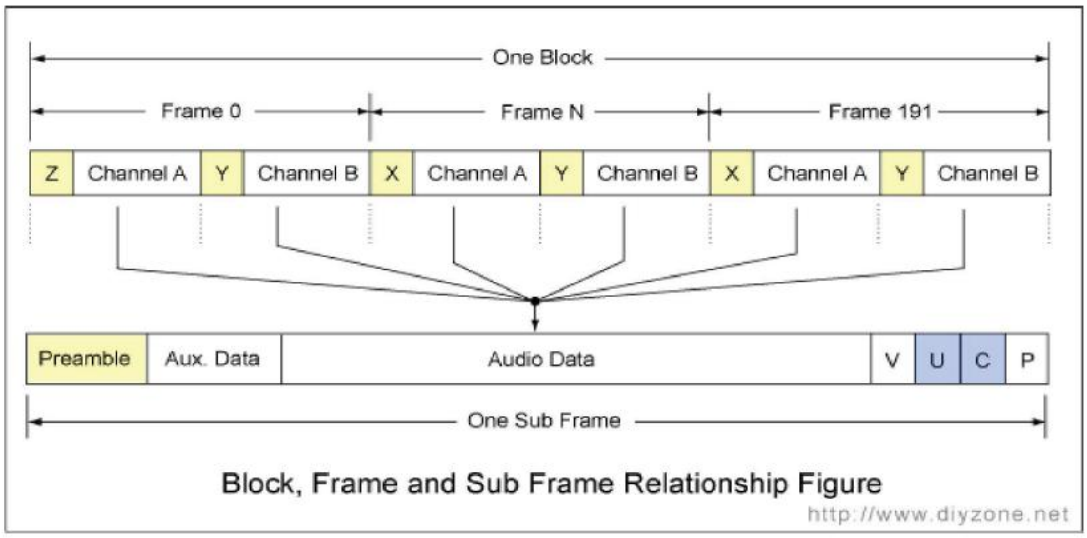

Figure 22 SPDIF block structure

	Biphase Pattern Prev.'s Last Cell = 0	Biphase Pattern Prev.'s Last Cell = 0	Channel
X	11100010	00011101	Ch. A (left)
Y	11100100	00011011	Ch. B (right)
Z	11101000	00010111	Ch. A (Block Start)

Preambles Configuration Figure

Figure 23 Preamble configuration

The preamble X, Y, Z is not the same with the data encode. The data is bi-phase encode (see Figure 24). Coding violation, defined as preambles, are used to identify sample and block boundaries.

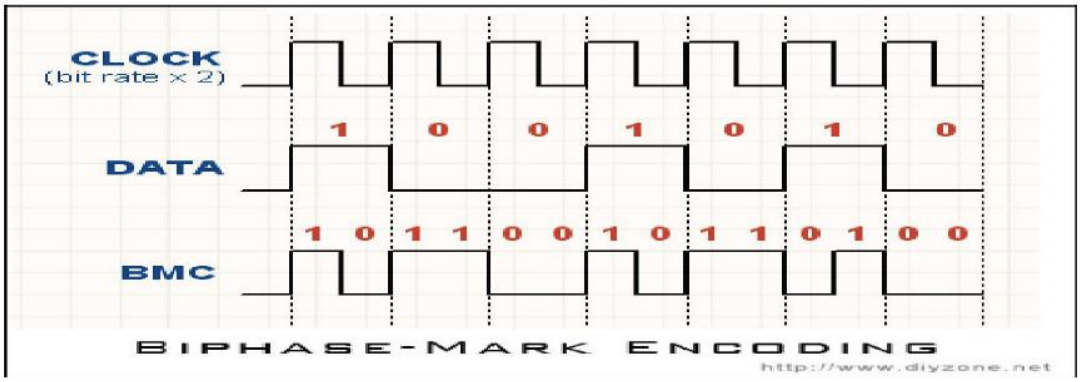

Figure 24 Bi-phase encode

4.11 IIC

IIC bus interface controller is an APB device, it allows the host processor to serve as a master or slave in the IIC bus. Data are transmitted to and received from the IIC bus via a buffered interface.

It Supports the stand and fast modes by programming the clock division register, Supports the 7-bit, 10-bit, and general-call addressing modes. It has glitch suppression capability through the debounce circuit. The salve address is Programmable, It supports the master-transmit, master-receive, slave-transmit, and slave-receive modes, and supports the multi-master mode also.

Figure 25 shows the IIC controller module block diagram, figure 26 shows this module interface and table 12 lists this module configuration register.

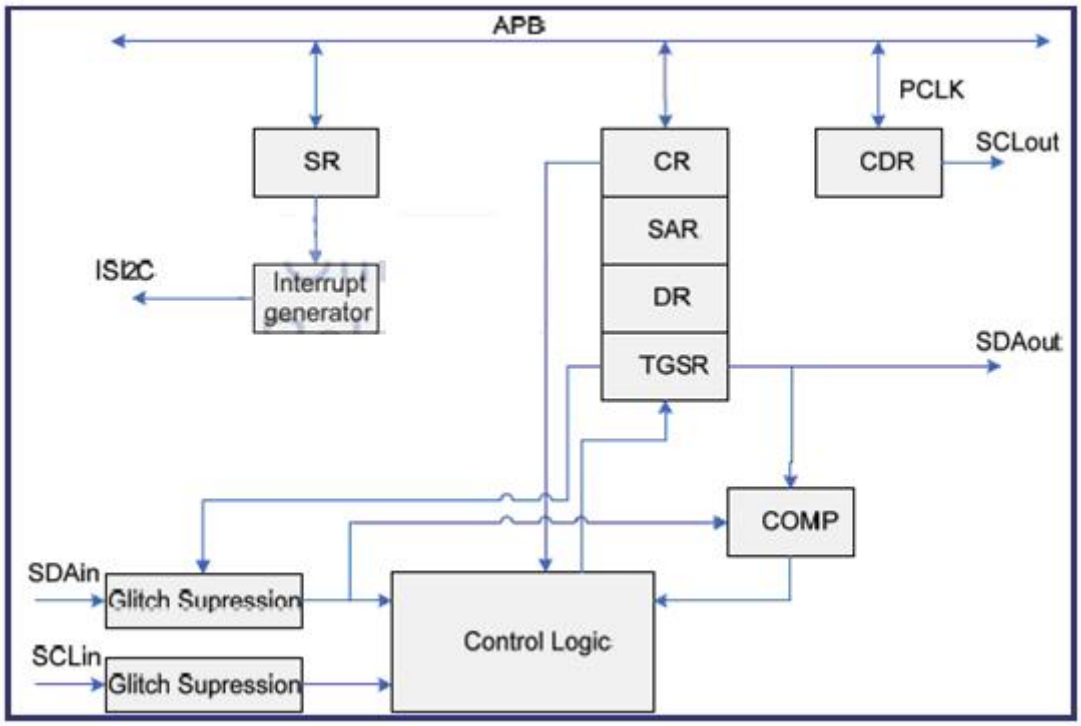

Figure 25 IIC controller module block diagram

This module includes register files, control logic, SCLout generator, and debounce circuit.

The register files, which contain the control register, slave address register, clock divider, status, data, setup/hold time, and glitch suppression, bus monitor registers, maximum timeout register, minimum timeout register, master extend time register, and slave extend time register are read/write, or read-only, or read/clear from the host processor throughout APB bus protocol.

The control logic, which detects the SCLin and SDAin in the I2C bus, decides the status on the bus.

The SCLout generator accepts the APB bus clock (PCLK) and divides its value in the clock division register, and multiplies 2 to generate the SCLout on the I2C bus.

The debounce circuit is used as the glitch suppression logic. Glitches are suppressed according to GSR* internal bus clock period, where GSR is bit 10 ~ bit 12 of register TGSR at offset 0x14.

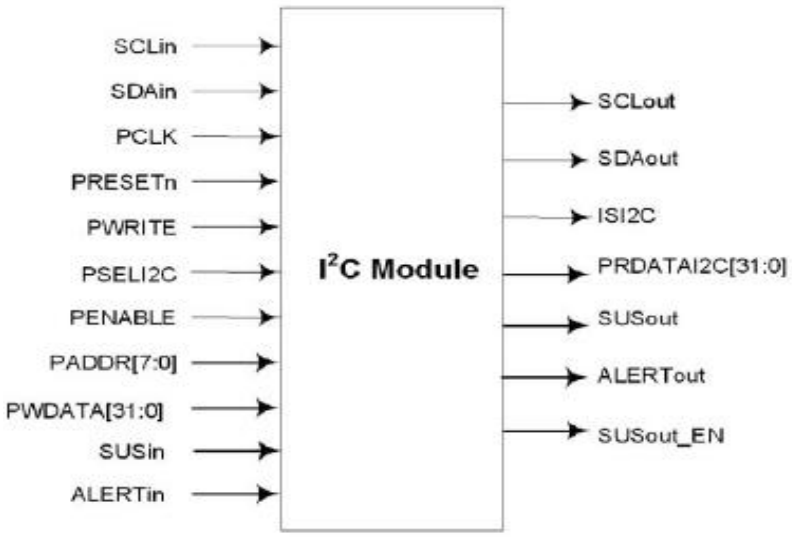

Figure 26 IIC controller module interface diagram

Table11 IIC controller module register list (BaseAddr=0x4001_1c00)

Offset Address	Type	Description	Reset Value
0x00	R/W	I²C Control Register (CR)	0x0000_0000
0x04	R/RC	I²C Status Register (SR)	0x0000_0000
0x08	R/W	I²C Clock Division Register (CDR)	0x0000_0000
0x0C	R/W	I²C Data Register (DR)	0x0000_0000
0x10	R/W	I²C Slave Address Register (SAR)	0x0000_0000
0x14	R/W	I²C Setup/Hold Time and Glitch Suppression Setting Register (TGSR)	0x0000_0401
0x18	R	I²C Bus Monitor Register (BMR)	0x0000_0003
0x30	R	I²C Revision Register	-

4.12 UART

UART links two bus. CM0+ configures UART register and transfers data with UART, DAM translates data with UART.

The system assigns two dedicated DMA channel to the UART_TX and UART_RX data transfer. UART have a programmable interrupt to the system.

UART controller is a serial communication element that implements the most common infrared communication protocols. It also support IRDA1.3 SIR protocol which is used in household electrical device IR transmitter and receiver (38KHZ).

UART support two work mode: UART mode , SIR mode.

The UART mode is default enabled after power up or system reset. This mode uses a wired interface for serial communication with a remote device or a modem. It can operate in a full-duplex mode, data transmission and reception can take place simultaneously. It works as a regular serial asynchronous communication controller that converts the parallel data received from the CPU or the DMA controller into serial data. It also converts the serial data received on the serial input terminal into parallel data. The format of the serial data stream is shown in figure 27. A data character contains 5 to 8 data bits. It is preceded by a start bit and is followed by an optional parity bit and a stop bit. Data is transferred in little-endian order (Least significant bit first). The clock for both transmit and receive channels is provided by an internal baud generator that divides the pre-scaled clock by any divisor value from 1 to 216 - 1. The output clock frequency of the baud generator must be programmed to be sixteen times the baud rate value. The baud generator input clock is derived from io_irda_uclk clock through a programmable prescaler. Both the communications format and baud rate must be programmed properly before operation.

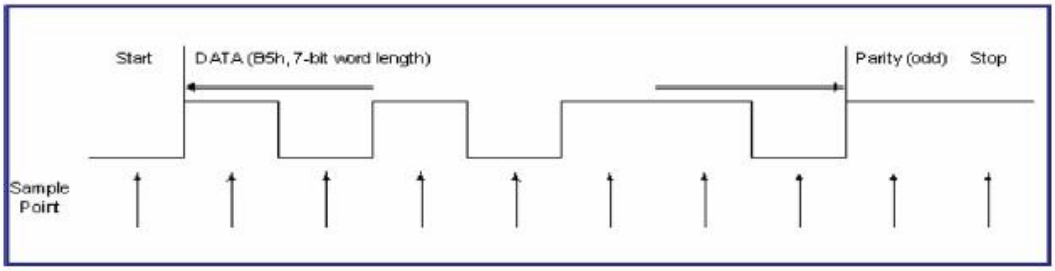

Figure 27 UART data representation and sampling

SIR (serial IR) mode supports bi-directional data communication with a remote device using the infrared radiation as the transmission medium. IrDA 1.3 SIR allows serial communication at baud rates of up to 115.2 kbps. The format of the serial data is similar to the UART data format. Each data word is sent serially beginning with a zero value start bit, followed by 8 data bits, and ending with one stop bit with a binary value of one. Sending a single infrared pulse signals a zero. A one is signaled by not sending any pulse. The width of each pulse can be either 1.6 µs or 3/16 of a single bit time. (1.6 µs equals 3/16 of a bit time at 115.2 kbps). This way, each word begins with a pulse for the start bit. The device operation in the IrDA SIR mode is similar to the operation in UART mode. The main differences are that, those data transfer operations are normally **performed in half-duplex fashion.**

Each data byte starts with a start bit (0), 1 byte of data, and then ends with at least a stop bit (1). Each serial data bit is encoded before transmission and decoded after reception. A 1 is decoded with no IR pulse and a 0 is decoded by sending 3/16ths of one

bit time IR pulse. Similarly, the received serial pulse is decoded as a 0 and the absence of an IR pulse is decoded as a 1, Please refer to Figure 28.

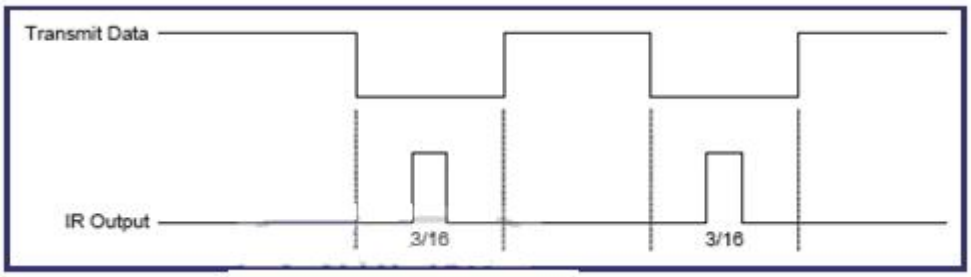

Figure 28 SIR encoding

Table 12 Uart module register list (BaseAddr=0x4001_0c00)

Offset	Type	Width	Name	Description	Reset Value
UART/Infrared SIR Mode					
+0x00	R	8	RBR	Receiver Buffer Register	0x00
	W	8	THR	Transmitter Holding Register	0x00
+0x04	R/W	4	IER	Interrupt Enable Register	0x00
+0x08	R	8	IIR	Interrupt Identification Register	0x01
	W		FCR	FIFO Control Register	0x00
+0x0C	R/W	8	LCR	Line Control Register	0x00
+0x10	R/W	7	MCR	Modem Control Register	0x00
+0x14	R	8	LSR	Line Status Register	0x60
	W		TST	Testing Register	0x00
+0x18	R	8	MSR	Modem Status Register	0x00
+0x1C	R/W	8	SPR	Scratch Pad Register	0x00
Registers accessible when DLAB = 1					
+0x00	R/W	8	DLL	Baud Rate Divisor Latch Least Significant Byte	0x01
+0x04	R/W	8	DLM	Baud Rate Divisor Latch Most Significant Byte	0x00
+0x08	R/W	5	PSR	Prescaler Register	0x01

4.13 PWM

CJC2100 integrates one channel PWM as APB device. The PWM output signals are based on the pwm_clk pin and must be a minimum of 2 clock cycles wide. Various configurations can be programmed to adjust the period and the waveform of the output signals. Figure 29 shows the block diagram of PWM.

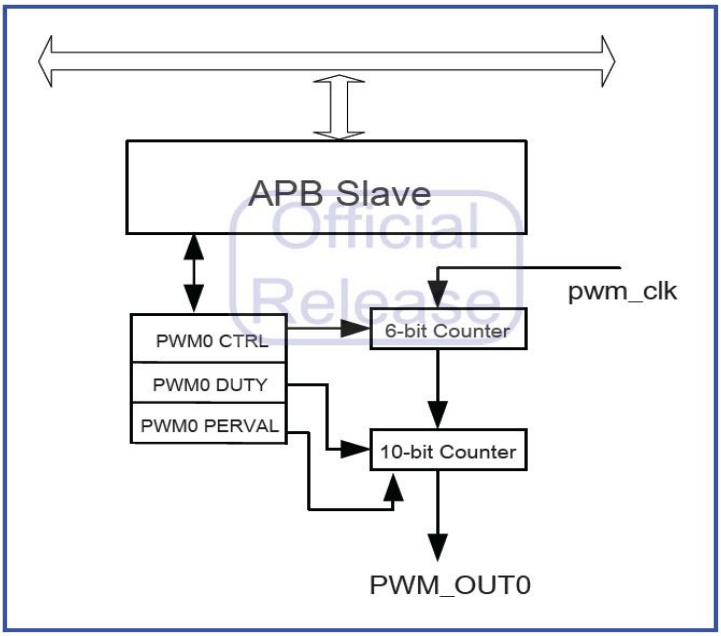

Figure 29 Block diagram of PWM

Table 13 PWM module register list (Baseaddr = 0x4001_1400)

Addr	Type	Name	Bit	Description	Default
0x00	R/W	CTRL	[31:6]	Reserved	0x0
			[5:0]	PRESCALE Determines the frequency of the PWM module clock PSCLK_PWM = pwm_clk/(CTRL+1)	
0x04	R/W	DUTY	[31:11]	Reserved	0x0
			[10]	FDCYCLE PWM full duty cycle 0=PWM_OUT0 duty cycle is determined by DCYCLE field 1=PWM_OUT0 is set high and does not toggle	
			[9:0]	DCYCLE PWM duty cycle Duty cycle of PWM_OUT.	
0x08	R/W	PERVAL	[31:10]	Reserved	0x0
			[9:0]	PERVAL PWM period control The number of PSCLK_PWM cycle that comprise one PWM_OUT cycle.	

Figure 30 shows the timing diagram of PWM.

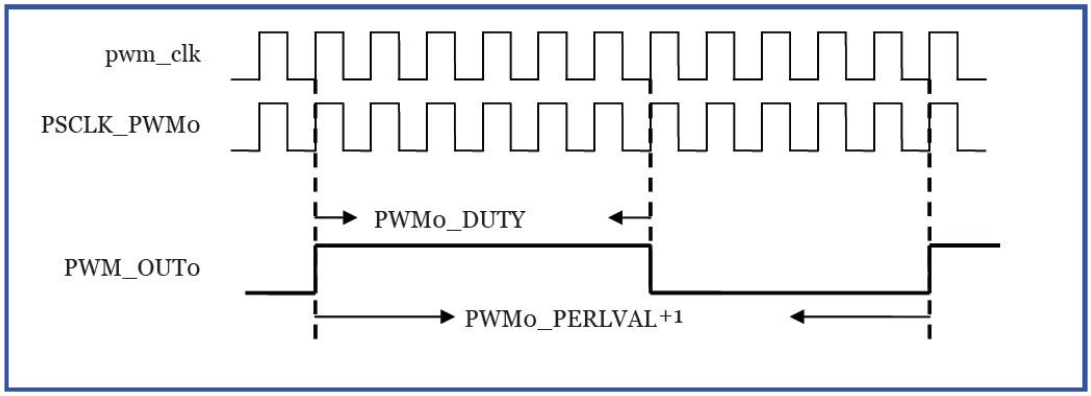

Figure 30 timing diagram of PWM

4.14 TIMER

Timer module is an APB device, it provides three independent sets of 32-bit sub-timers, and the first sub-timer is the default sub-timer. Each sub-timer can use either internal system clock (PCLK) or external clock (EXTCLK) to increase or decrease the counting. Two match registers are provided for each sub-timer. Whenever the value of the match registers equals to any one of the sub-timers, the timer interrupt is triggered immediately. The issuance of the timer interrupt can be decided by the register setting when an overflow occurs. CJC2100 assigns 3 interrupt for timer.

Figure 31 is timer module block diagram. It is composed of a timer register, a timer counter, an overflow detector, and a match comparator.

The timer register block includes a counter register, two match registers, an autoload register, and a control register. The timer counter block can be configured to either decrease or increase. When the counter register is reset, its value is set to the value of the register TmLoad. The counter register will then hold the TmLoad value until the timer is enabled. Once the timer is enabled, the value of the counter register will be increased (Decreased if Tm(1 ~ 3)UpDown is set to count down) by PCLK or EXTCLK. The programmer can read or write the counter register at any time.

Whether the TmOFEnable bit in the TmCR register is enabled or disabled, the value of the TmLoad register will be automatically copied into the counter register when the counter overflows. The programmer can use TmLoad to set the period between two counter overflows and use TmOFEnable to determine whether or not tm_intr is triggered when the counter overflows. The signal tm_intr will be triggered when the TmEnable bit is set and the value of the counter register equals the value in the TmMatch1 or TmMatch2 register. The tm_intr signal will be triggered alternatively where the counter overflows and when the TmEnable and TmOFEnable bits in TmCR are set.

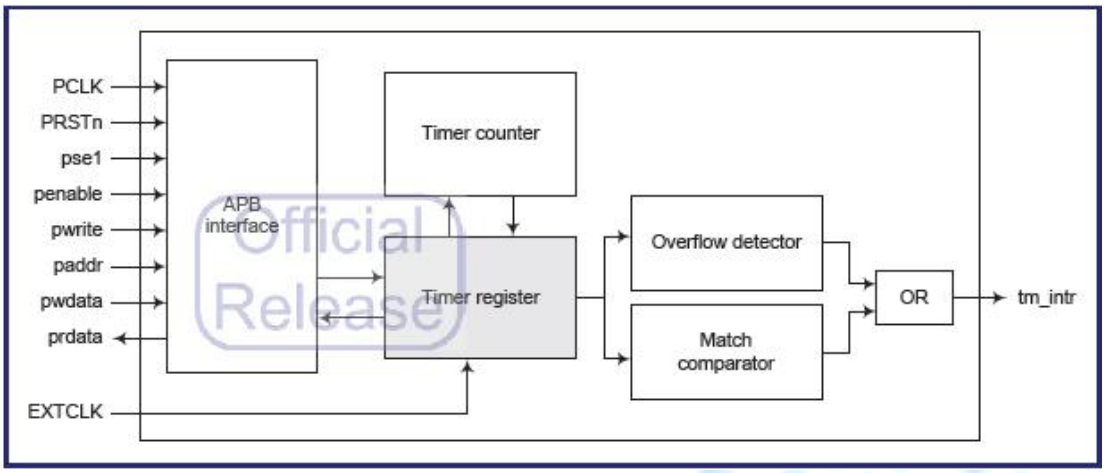

Figure 31 timer module block diagram

Figure 32 is this module interface pin and table 15 is register list.

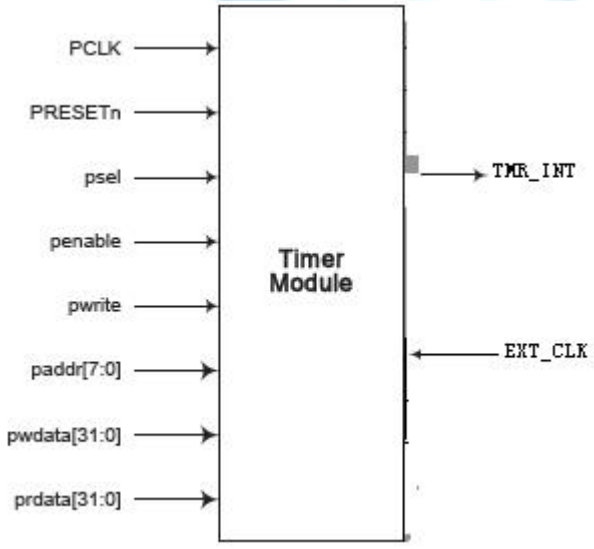

Figure 32 timer module interface diagram

Table 14 Timer register list (BaseAddr = 0x4001_3000)

Offset	Type	Width	Reset	Name	Configuration	Description
0x00	R/W	32	0x--	Tm1Counter	None	Timer1 counter
0x04	R/W	32	0x--	Tm1Load	None	Timer1 auto reload value
0x08	R/W	32	0x--	Tm1Match1	None	Timer1 match value
0x0C	R/W	32	0x--	Tm1Match2	None	Timer1 match value
0x10	R/W	32	0x--	Tm2Counter	TM2	Timer2 counter
0x14	R/W	32	0x--	Tm2Load	TM2	Timer2 auto reload value
0x18	R/W	32	0x--	Tm2Match1	TM2	Timer2 match value
0x1C	R/W	32	0x--	Tm2Match2	TM2	Timer2 match value
0x20	R/W	32	0x--	Tm3Counter	TM3	Timer3 counter
0x24	R/W	32	0x--	Tm3Load	TM3	Timer3 auto reload value
0x28	R/W	32	0x--	Tm3Match1	TM3	Timer3 match value
0x2C	R/W	32	0x--	Tm3Match2	TM3	Timer3 match value
0x30	R/W	12	0x0	TmCR	None	Timer1, Timer2, Timger3 control register
0x34	R/W	9	0x0	IntrState	None	Interrupt State of ATFTMR010
0x38	R/W	9	0x0	IntrMask	None	Interrupt Mask of ATFTMR010
0x3C	R	32	0x--	TmRevision	None	ATFTMR010 revision number

4.15 Watchdog

Watchdog module is an APB bus device. It is used to prevent the system from the infinite loop if the software gets trapped in the deadlock. In the normal operation, the user restarts the WDT at the regular intervals before the counter counts down to 0. If the counter does reach 0, the WDT generates one or a combination of the signals, system reset, system interrupt, or external interrupt to reset the system, interrupt the system, or interrupt an external device correspondingly.

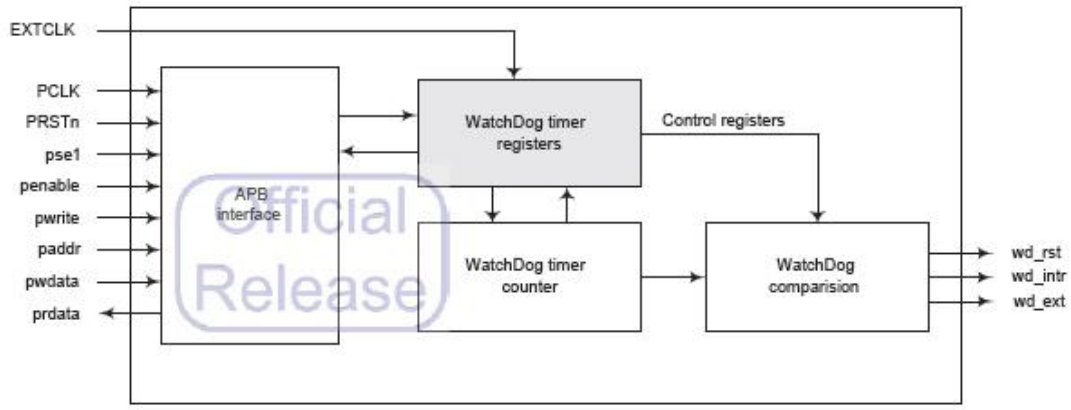

Figure 33 watchdog module block diagram

Figure 33 shows the watchdog module block diagram, The APB interface can receive the signals from APB bus. When reset, the WDT registers can reset the values. After the programmer turns on the enable bit of the WDT in the WatchDog timer control register, the WDT counter starts to reduce the counting. If the Watch Dog timer reaches 0, the wd_rst, wd_intr, or wd_ext signal is triggered. As long as the signal is set in the WdInstlen register, the assertion of this signal depends on the WdRst, WdIntr, WdExt bits in the WatchDog timer control register, the signal will keep asserted for a period of time. The WdStatus is used to check if the counter reaches 0 or not. The programmer can clear the WdStatus bit by writing a 1 to the WdClear bit.

To prevent the unexpected reset, the programmer needs to write 0x5AB9 to the WdRestart register as the password to activate the down counting function. If the WdRestart register equals 0x5AB9, the value of the WdLoad register will be loaded into the counter of the WatchDog timer. The default reset value of the WdRestart register is 0, and the WdRestart register will automatically return to 0 after each write. The default value of the WdLoad register is set to 0x3EF1480. The programmer can write this register to customize the operation of the WatchDog timer reset.

Figure 34 shows the watchdog module interface and table 16 shows this module register list.

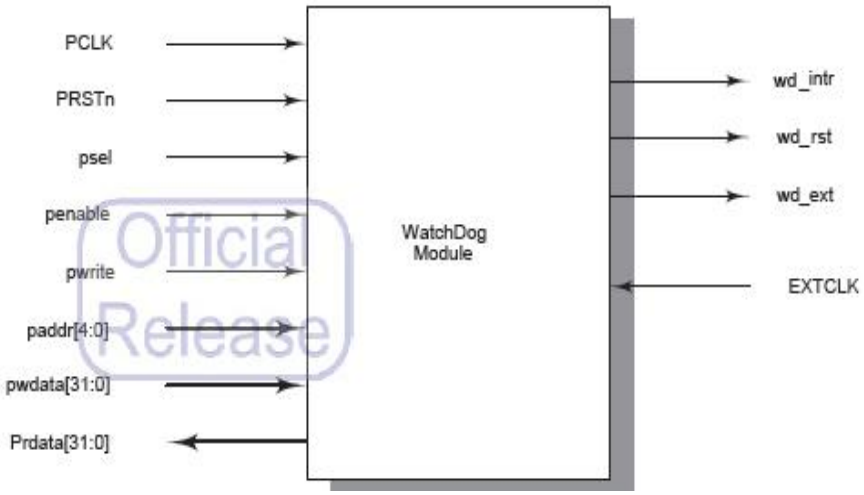

Figure 34 watchdog module interface diagram

Table 15 Watchdog module register list (BaseAddr = 0x4001_4800)

Offset	Type	Width	Reset Value	Name	Config.	Description
0x00	R	32	0x3EF1480	WdCounter	None	The WatchDog timer counter register
0x04	R/W	32	0x3EF1480	WdLoad	None	The WatchDog timer counter auto reload register The auto-reload register is set to 0x3EF1480 as the default.
0x08	W	16	0x0000	WdRestart	None	The WatchDog timer counter restart register If writing 0x5AB9 to this register, the WatchDog timer will automatically reload the WdLoad to Wdcounter and restart the counting.
0x0C	R/W	5	0x0	WdCR	None	The WatchDog timer control register
0x10	R	1	0x0	WdStatus	None	The WatchDog timer status register This bit is set when the counter reaches 0. 0: Does not reach 0 1: Reache 0
0x14	W	1	0x0	WdClear	None	The WatchDog timer is cleared. Writing 1 or 0 to this register will clear the WdStatus.
0x18	R/W	8	0xFF	WdIntrlen	None	The WatchDog timer interrupt length This register controls the length of wd_rst, wd_intr, and wd_ext. The default value is 0xFF.
0x1C	R	32	0x--	WdRevision	None	The revision number of ATFWDT010

4.16 SPI

CJC2100 integrates 2 SPI interfaces. SPI is a kind of synchronous serial port interface that allows the host processor to serve as a master or a slave. It can connect to various devices by using serial protocol. It supports several kind of synchronous serial port such as the Synchronous Serial Port (SSP) from Texas Instruments, the Serial Peripheral Interface (SPI) from Motorola, MICROWIRE from National Semiconductor, I2S from Philips, AC-link from Intel, and SPDIF. And the serial data formats may range from 4 bits to 32 bits in length.

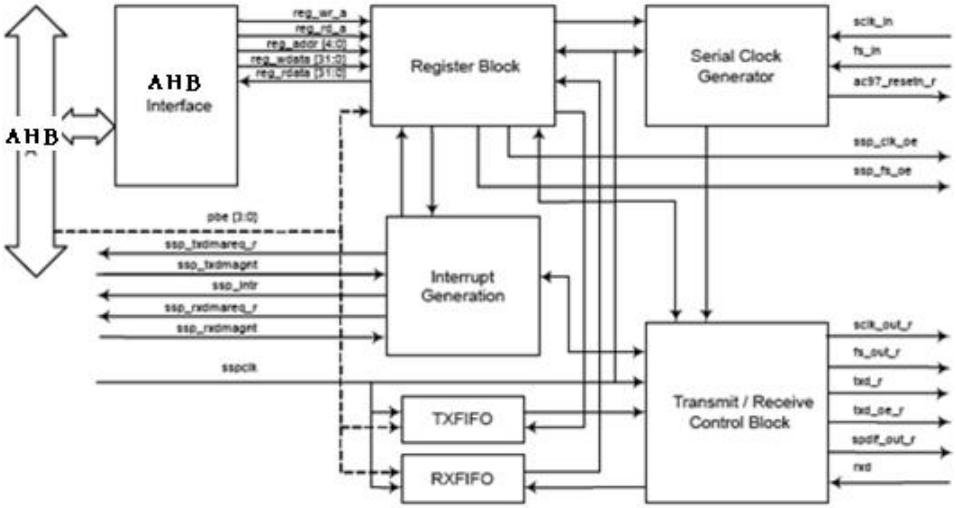

Figure 35 SPI module block diagram

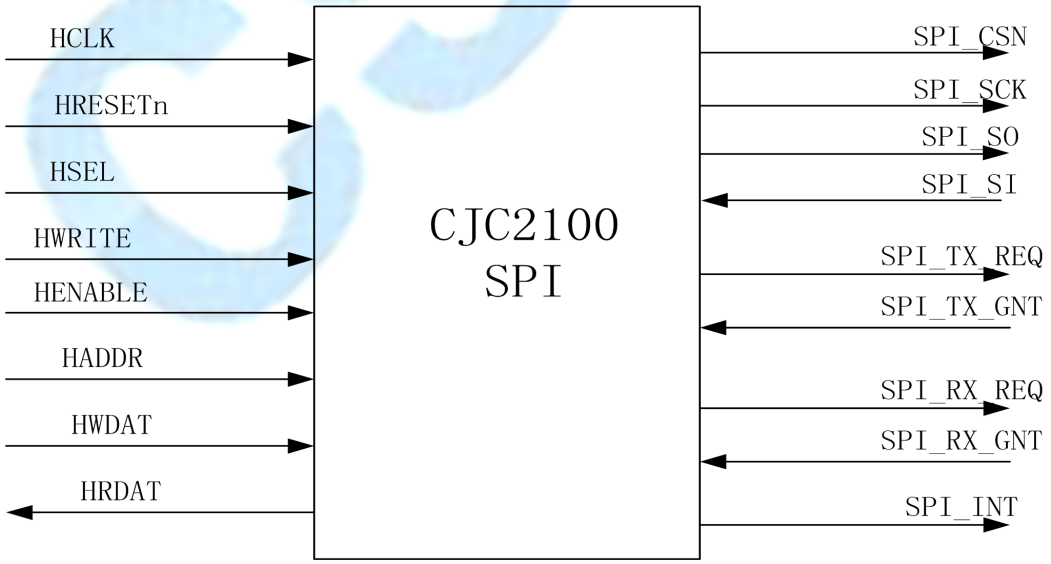

Figure 36 SPI module interface

Figure 35 is SPI module block diagram and figure 36 is SPI module interface. The module includes AHB wrapper, register

block, interrupt generation control, clock generator and transmit/receive control block.

The register block accepts the register read/write from the AHB interface. This register block also provides the property information to other blocks of the SSP controller.

The interrupt generation block collects the information (FIFO full/empty, transmit/receive busy, etc.) from the transmit/receive control block and provides the value to the register block. If the interrupt conditions match, the signal ssp_intr will be asserted.

The clock generation block generates a serial clock for the communication. The format of the serial clock is defined through the control register 0. Generally, the clock will not start if the SSP controller is not enabled. Once the SSP controller is enabled and the clock running condition is matched (For example, when TI's SSP is specified and the transmit FIFO is not empty, or when I2S is specified), the serial clock will start running. The operating frequency of the serial clock depends on the setting of the SCLKDIV register. This block also gives the serial clock and the frame/sync. Information to the data transmit/receive control block.

The main function of transmit/receive control module is to perform the parallel-to-serial transmission or handle the serial-to-parallel reception from the external devices. If the **master mode** is specified and the transmit FIFO is not empty, the data in the transmit FIFO will be read and shifted out via the txd_r pin. If a whole word is completely shifted out and the transmit FIFO still contains valid data, the next data will be read and shifted out. The received data can be shifted in via the rxd pin. Once the reception is complete, the received word will be written into the receive FIFO, and the receive control logic will continue to receive the next word. If the half-duplex protocol is specified, the receive control logic will not start until the transmission has been completed. If the **slave mode** is specified, the SSP controller will start to transmit/receive data when frame/sync. is activated. The transmission or reception will continue until the SSP controller is disabled or the frame/sync. is deactivated. If the half-duplex protocol is specified, the transmit control logic will not start until the reception has been completed. The transmission and reception can be activated simultaneously when the full-duplex protocol is specified.

SPI module interface inlcude AHB bus interface, SPI external interface, TX/RX FIFO signal and interrupt signal.

Table 16 SPI module register list (BaseAddr = 0x1000_0000)

Address	Type	Description	Reset Value
+0x00	R/W	SSP control register 0	0x0000_010C
+0x04	R/W	SSP control register 1	0x0007_8000
+0x08	R/W	SSP control register 2	0x0000_0002
+0x0C	R	SSP status register	0x0000_0002
+0x10	R/W	SSP interrupt control register	0x0000_2200
+0x14	RC	SSP interrupt status register	0x0000_0008
+0x18	R/W	SSP data register	N/A
+0x60	R	SSP revision register	N/A
+0x64	R	SSP feature register	N/A

4.17 USB controller

USB controller is an AHB device, the main function is to implement the data transfer between CJC2100 system and external USB master device or USB slave device.

USB controller module support USB OTG, it can work as a host to access the external USB device, it also can work as USB device being accessed by external USB master such as PC.

USB controller is compliant with USB specification revision 2.0, it is Compliant with On-The-Go supplement to USB 2.0 specification revision 1.0,it Supports UTMI+ level 2 compliant transceiver and compliant with EHCI(Enhanced Host Controller Interface Specification for USB) 1.0,it support OTG SRP(OTG Session Request Protocol) and HNP(OTG Host Negotiation Protocol) .it Supports point-to-point communications with one HS/FS/LS device, endpoint in this module is can be hardware configured as HS/FS device. Both host and device support isochronous, interrupt, control, bulk transfers. it support DMA access to internal FIFO, and support suspend mode, remote wake-up and resume.

USB controller is mainly composed of a UTM synchronization, packet encode/decode, RAM controller endpoint control and CPU interface, as shown in Figure 37. From the top, the ATFOTG200 system bus can be a PVCI bus or an AHB bus, the actual implementation is dependent on the system platform. Basically, the DMA controller is a bus master on a system bus that conveys data between the shared system memory to or from the ATFOTG200. The microprocessor interface controller is a bus slave on the system bus that provides the interface for the microprocessor to access the configuration register files of the ATFOTG200. The host controller, device controller, and OTG controller; each has its corresponding register files.

The system bus and USB are running in different clock frequencies. Because of this, the synchronization block provides a synchronizing mechanism for these 2 clock domains. The acceptable system bus frequencies of the ATFOTG200 are in the range from 15 MHz to 133 MHz. The USB clock frequency must be 30 MHz with a 16-bit interface as defined in the UTMI specification.

For the host controller, being compatible with EHCI means the USB 2.0 uses the same skeleton as the EHCI, but without the additional USB 1.1 controller for FS/LS. Instead, the USB 1.1 controller is integrated into the host controller of the USB 2.0 EHCI. Thus, for HS, FS, and LS, the same EHCI interface host controller is used. The device controller is a HS and FS USB 2.0 device with a built-in DMA controller to provide the intellectual properties. Each endpoint, except the endpoint 0, can be programmed as an endpoint for the isochronous, interrupt, or bulk transfer. Additionally, the OTG bus monitor controller provides a dual-role capability for dynamic switching between the host and device. It supports 2 OTG protocols, the Session Request Protocol (SRP) and Host Negotiation Protocol (HNP). It also controls the power management and speed emulation for the host and device.

The FIFO controller and PIE are both used by the host and device. The otg_cur_role (Current role is decided by the OTG controller) pin decides which one can take control of them. The FIFO size is a 2 kB SRAM that is divided into 4 512-byte blocks and can provide the ping-pong mechanism when acting as a device. There is also a small 64-bit SRAM dedicated to the endpoint 0 device for the control transfers. The PIE operates in the 16-bit parallel data streams with a UTMI+ interface and runs on a 30 MHz clock provided by the transceiver. The major function as a host is to generate the tokens and data packets. The major function as a device is to decode the tokens and data packets. The host results are delivered to the device and the device results are delivered to the host.

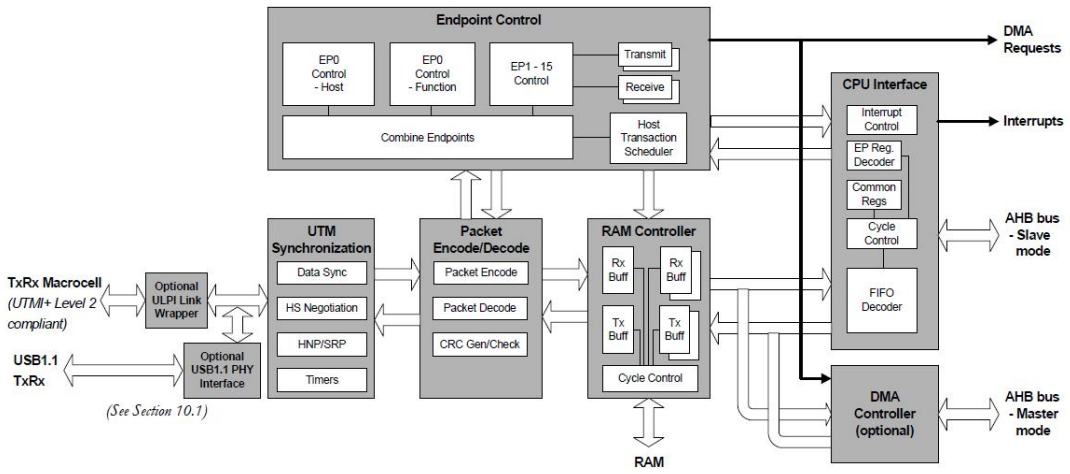

Figure 37 USB controller module block diagram

The following part will describe each block function more detailed.

UTM Synchronization

The role of the UTM synchronization block is to resynchronize between the transceiver macrocell 30/60MHz clock domain and the dual role controller's system clock CLK, which drives the remainder of the core up to and including the CPU interface. This allows the rest of the USB driver controller to run at the CPU bus speed without requiring any further synchronization. The block also performs High-speed detection handshaking and handles HNP and SRP in point-to-point communications with another USB OTG device.

Where the core has been configured for an 8-bit(60MHz) transceiver, the block first converts the data to 16-bit – allowing the

core to be driven by a system clock running at a little over 30MHz. (You might expect that the core could be run from a similar clock when used with a 16-bit(30MHz) transceiver but in practice over 48MHz is necessary to guarantee the required bus turnaround times with a 16-bit transceiver.

Packet Encoding/Decoding

The Packet Encode/Decode block generates headers for packets to be transmitted and decodes the headers on received packets. It also generates the CRC for packets to be transmitted and checks the CRC on received packets.

Endpoint Controllers

Two controller state machines are used: one for control transfers over Endpoint 0, and one for Bulk/Interrupt/Isochronous transactions over Endpoints 1 to 15.

Data BUS Interface

The Data bus Interface allows access to the control/status registers and the FIFOs for each endpoint. It also generates interrupts to the CPU when packets are successfully transmitted or received, and when the core enters Suspend mode or resumes from Suspend mode.

The interface provided by the USB driver controller is a 32-bit synchronous interface that follows the design specified for interfaces to an AMBA AHB bus. Interface to other bus standards may be achieved through the addition of an appropriate wrapper to the core.

RAM controller

The RAM controller provides an interface to a single block of synchronous single-port RAM, which is used to buffer packets between the CPU and USB. It takes the FIFO pointers from the endpoint controllers, converts them to address pointers within the RAM block and generates the RAM access control signals.

DMA Controller

If required, the USB driver controller may include a multi-channel DMA controller for efficient loading/unloading of the endpoint FIFOs. This DMA controller is configurable for up to 8 channels. The DMA controller has its own block of control registers and its own interrupt controller. It supports two DMA modes, referred to as DMA modes 0 and 1 and it can handle packet sizes up to 8k.

When operating in DMA Mode 0, the DMA controller can be only programmed to load/unload one packet, so processor intervention is required for each packet transferred over the USB. This mode can be used with any endpoint, whether it uses Control, Bulk, Isochronous, or Interrupt transactions.

When operating in DMA Mode 1, the DMA controller can be programmed to load/unload a complete bulk transfer (which can be many packets). Once set up, the DMA controller will load/unload all packets of the transfer, interrupting the processor only when the transfer has completed. DMA Mode 1 can only be used with endpoints that use Bulk transactions.

Each channel can be independently programmed for operating mode.

The USB driver controller core has a maximum of 438 external signals (see Figure 38), 182 inputs and 256 outputs.

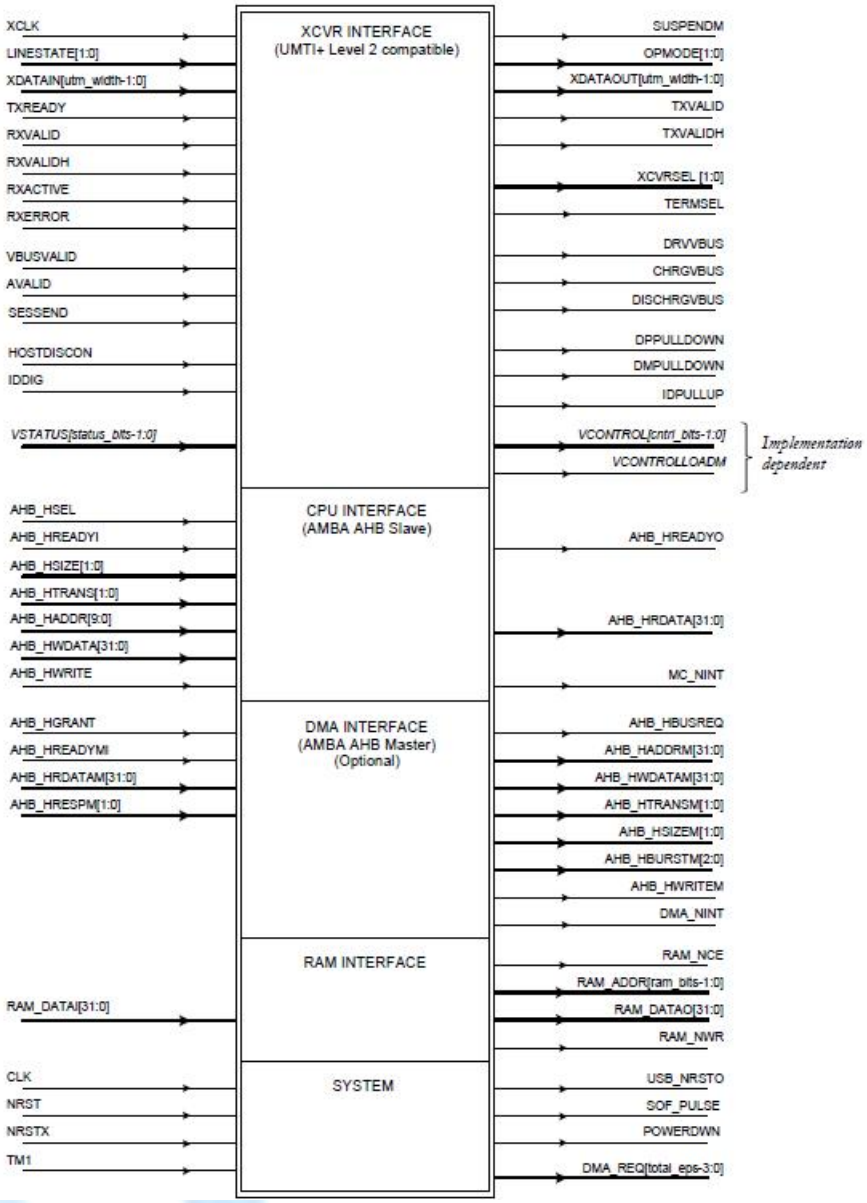

Figure 38 USB interface signal

The USB driver controller register map is split into the following sections (BaseAddr=0x4000_2000):

Common USB registers (00h–0Fh) – These registers provide control and status for the complete core.

Indexed Endpoint Control/Status registers (10h–1Fh) – These registers provide control and status for the endpoints. The registers mapped into this section depend on whether the core is in Peripheral mode (DevCtl.D2=0) or in Host mode (DevCtl.D2=1) and on the value of the Index register.

FIFOs (20h–5Fh) – This address range provides access to the endpoint FIFOs.

Additional Control and Configuration registers (60h–7Fh) – These registers provide additional device status and control.

Non-Indexed Endpoint Control/Status registers (100h and above) – The registers available at 10h–1Fh, accessible independently of the setting of the Index register. 100h–10Fh EP0 registers; 110h–11Fh EP1 registers; 120h–12Fh EP2; and so

on.

DMA Control Registers (200h and above) – These registers only appear if the design is synthesized to include optional DMA controller.

RqPktCount Registers (302h – 31Eh) – These registers are used in Host mode in conjunction with AutoReq.

Note: Any further registers associated with any bridge provided for use with the USB driver controller core or any changes to the following registers that result from using this bridge will be described in the separate specification for the bridge included in the **musbhdrc/docs** directory.

Figure 39 shows USB memory map.

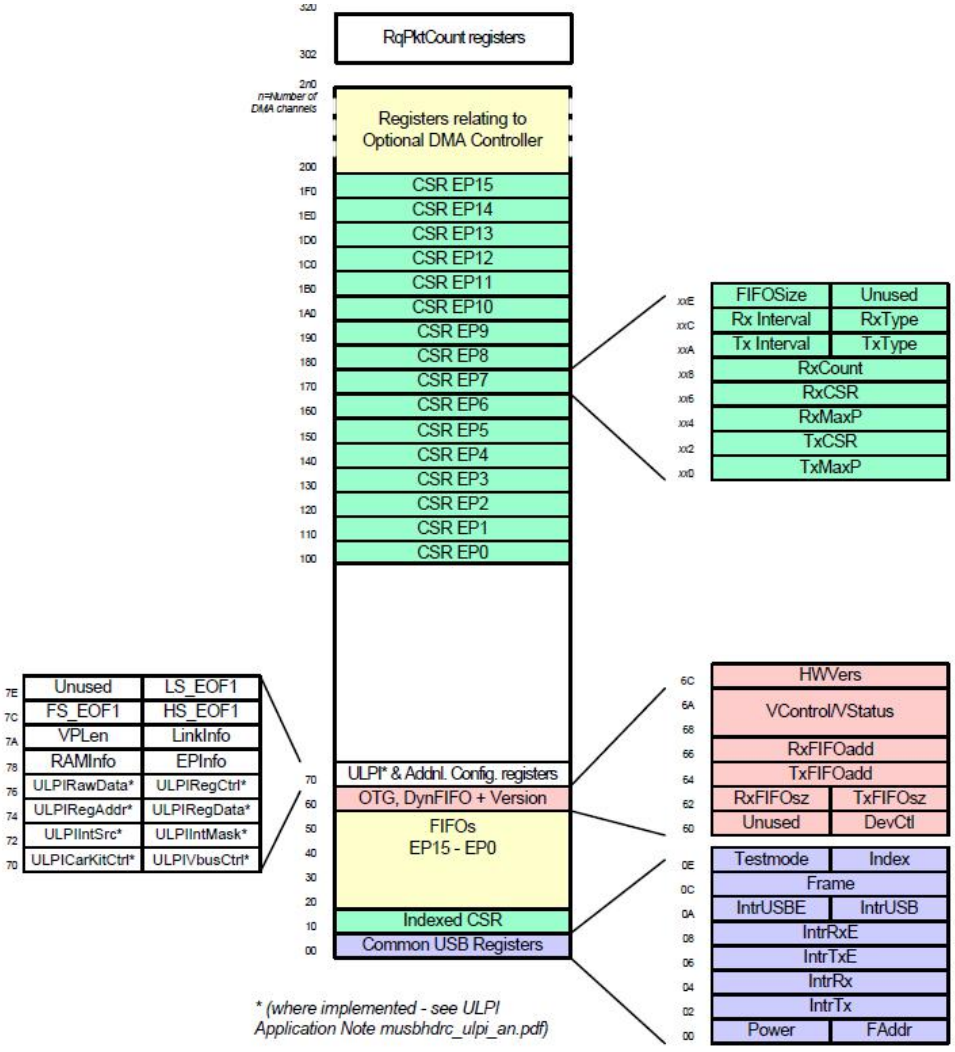

Figure 39 USB memory map

Table 17 USB register list(BaseAddr=0x4000_2000)

USB driver controller REGISTER MAP: Common USB registers				
ADDR	NAME	TYPE	DESCRIPTION	DEFAULT
00	FAddr		Function address register	
01	Power		Power management register	
02,03	IntrTx		Interrupt register for endpoint 0 plus Tx Endpoints 1 to 15	
04,05	IntrRx		Interrupt register for Rx Endpoints 1 to 15	

06,07	IntrTxE		Interrupt enable register for IntrTx	
08,09	IntrRxE		Interrupt enable register for IntrRx	
0A	IntrUSB		Interrupt register for common USB interrupts	
0B	IntrUSBE		Interrupt enable register for IntrUSB	
0C,0D	Frame		Frame number	
0E	Index		Index register for selecting the endpoint status and control registers	
0F	Testmode		Enables the USB 2.0 test modes	
USB driver controller REGISTER MAP: Indexed registers – Peripheral mode (Control Status registers for endpoint selected by the Index register when DevCtl.D2 = 0)				
10,11	TxMaxP		Maximum packet size for peripheral Tx endpoint. (Index register set to select Endpoints 1 – 15 only)	
12,13	CSR0		Control Status register for Endpoint 0. (Index register set to select Endpoint 0)	
	Tx CSR		Control Status register for peripheral Tx endpoint. (Index register set to select Endpoints 1 – 15)	
14,15	RxMaxP		Maximum packet size for peripheral Rx endpoint. (Index register set to select Endpoints 1 – 15 only)	
16,17	RxCSR		Control Status register for peripheral Rx endpoint. (Index register set to select Endpoints 1 – 15)	
18,19	Count0		Control Status register for Endpoint 0. (Index register set to select Endpoint 0)	
	RxCount		Control Status register for peripheral Tx endpoint. (Index register set to select Endpoints 1 – 15)	
1A,1B	-		*Reserved.* Value returned affected by use in Host mode	
1C,1E	-		*Unused,* always return 0	
1F	ConfigData		Returns details of core configuration. (Index register set to select Endpoint 0.)	
	FIFOSize		Returns the configured size of the selected Rx FIFO and Tx FIFOs (Endpoints 1 – 15 only).	
USB driver controller REGISTER MAP: Indexed registers – Host mode (Control Status registers for endpoint selected by the Index register when DevCtl.D2 = 1)				
10,11	TxMaxP		Maximum packet size for host Tx endpoint. (Index register set to select Endpoints 1 – 15 only)	
12,13	CSR0		Control Status register for Endpoint 0. (Index register set to select Endpoint 0)	
	Tx CSR		Control Status register for host Tx endpoint. (Index register set to select Endpoints 1 – 15)	
14,15	RxMaxP		Maximum packet size for host Rx endpoint. (Index register set to select Endpoints 1 – 15 only)	
16,17	RxCSR		Control Status register for host Rx endpoint. (Index	

			register set to select Endpoints 1 – 15)	
18,19	Count0		Number of received bytes in Endpoint 0 FIFO. (Index register set to select Endpoint 0)	
	RxCount		Number of bytes in host Rx endpoint FIFO. (Index register set to select Endpoints 1 – 15)	
1A	TxType		Sets the transaction protocol and peripheral endpoint number for the host Tx endpoint. (Index register set to select Endpoints 1 – 15 only)	
1B	NAKLimit0		Sets the NAK response timeout on endpoint 0. (Index register set to select Endpoint 0)	
	TxInterval		Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host Tx endpoint. (Index register set to select Endpoints 1 – 15 only)	
1C	RxType		Sets the transaction protocol and peripheral endpoint number for the host Rx endpoint. (Index register set to select Endpoints 1 – 15 only)	
1D	RxInterval		Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host Rx endpoint. (Index register set to select Endpoints 1 – 15 only)	
1E	-		*Unused,* always return 0	
1F	ConfigData		Returns details of core configuration. (Index register set to select Endpoint 0).	
	FIFOSize		Returns the configured size of the selected Rx FIFO and Tx FIFOs (Endpoints 1 – 15 only).	
USB driver controller REGISTER MAP: FIFOs				
20-5F	FIFOx		FIFOs for endpoints 0-15	
60	DevCtl		OTG device control register	
61	-		Unused	
62	TxFIFOsz		Tx Endpoint FIFO size	Only used if Dynamic FIFO sizing option is selected otherwise return 0.
63	RxFIFOsz		Rx Endpoint FIFO size	
64,65	TxFIFOadd		Tx Endpoint FIFO address	
66,67	RxFIFOadd		Rx Endpoint FIFO address	
68-6B	VControl/VStatus		UTMI+PHY Vendor registers	
6C,6D	HWVers		Hardware Version number register	
6E,6F	-		Unused	
70-77	-		ULPI Registers, only implemented where ULPI Link Wrapper is used.	

78	EPInfo		Information about numbers of Tx and Rx endpoints.	
79	RAMInfo		Information about the width of the RAM and the number of DMA channels.	
7A	LinkInfo		Information about delays to be applied	
7B	VPLen		Duration of the VBus pulsing charge	
7C	HS_EOF1		Time buffer available on High-Speed transactions	
7D	FS_EOF1		Time buffer available on Full-Speed transactions	
7E	LS_EOF1		Time buffer available on Low-Speed transactions	
7F	-		Unused	
USB driver controller REGISTER MAP: RqPktCount Registers (302h – 31Eh)				
300+2*n	RqPktCount		Number of requested packets for receive endpoint n (endpoints 1-15 only)	

4.18 POR

Power on reset module is a system asynchronous reset signal generation module, it detect the power status and generate the reset signal when power is supply. Figure 40 is CJC2100 power on reset circuit block diagram and figure 41 is the POR signal timing sequence.

TBD

Figure 40 CJC2100 POR circuit block diagram

TBD

Figure 41 CJC2100 POR timing

4.19 Power control unit

4.19.1 Power supply

Figure 42 is the CJC2100 chip power pad and power supply diagram. The power supply include three part: 3.3V supply for CJC2100 analog circuit,3.3V power supply for I/O and 1.8V power pad from internal LDO. 3.3V power supply for analog circuit have 3 pairs power/ground pin, one pair is for USB PHY, another is for PLL which need stable and “clear” power supply to improve jitter and accurate performance, the other is for other analog module in CJC2100 chip such as LDO, APU, SARADC, LVR controller analog circuit. 3.3V power supply for I/O includes 1 pairs. CJC2100 have a internal LDO, which transfer 3.3V power to 1.8V power, LDO output power supply for CJC2100 digital logic and USB PHY digital logic also, a 1.8V pin is output to connect capacitor for decoupling.

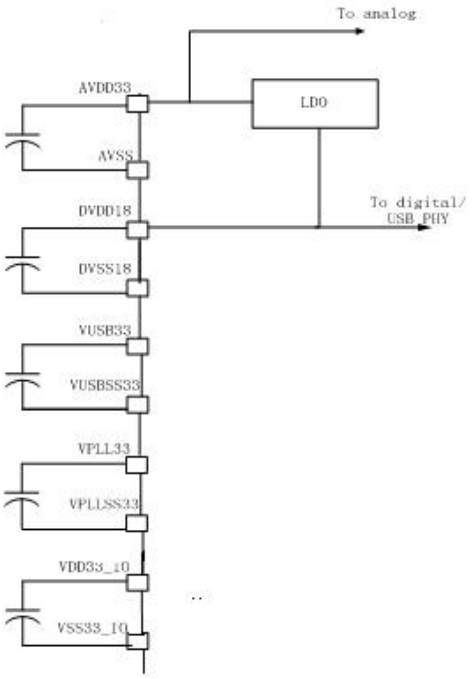

Figure 42 CJC2100 power supply diagram

CJC2100 has a programmable 3 threshold levels by configurated registers. The threshold levels have 3.8V, 2.7V and 2.0V (see Table 19).

4.19.2 LVR

CJC2100 has a low voltage reset generation circuit(LVR). The main circuit of LVR is comparator, it comparator the supply voltage with the configured threshold. If the supply is lower than the threshold, reset will be generated and send to all others module, then CJC2100 will enter into reset state.

4.19.3 Register control

Table 18 Power unit register list (BaseAddr = 0x4001_0000)

Addr	Type	Name	Bit	Description	Default
0x8	R/W	R2	[31:8]	Reserved	0x0
			[7:4]	Clk_io_mux Select clk to output	
			[3]	Control sys pll bypass	
			[2]	Lvr_en Enable low voltage reset 1: enable 0:disable	
			[1]	Reserved	
			[0]	Configure the threshold voltage reset 00: 2.0v 01: 2.4v 10:2.7v 11:3v	