NUFRONT ®

NL6621M Datasheet

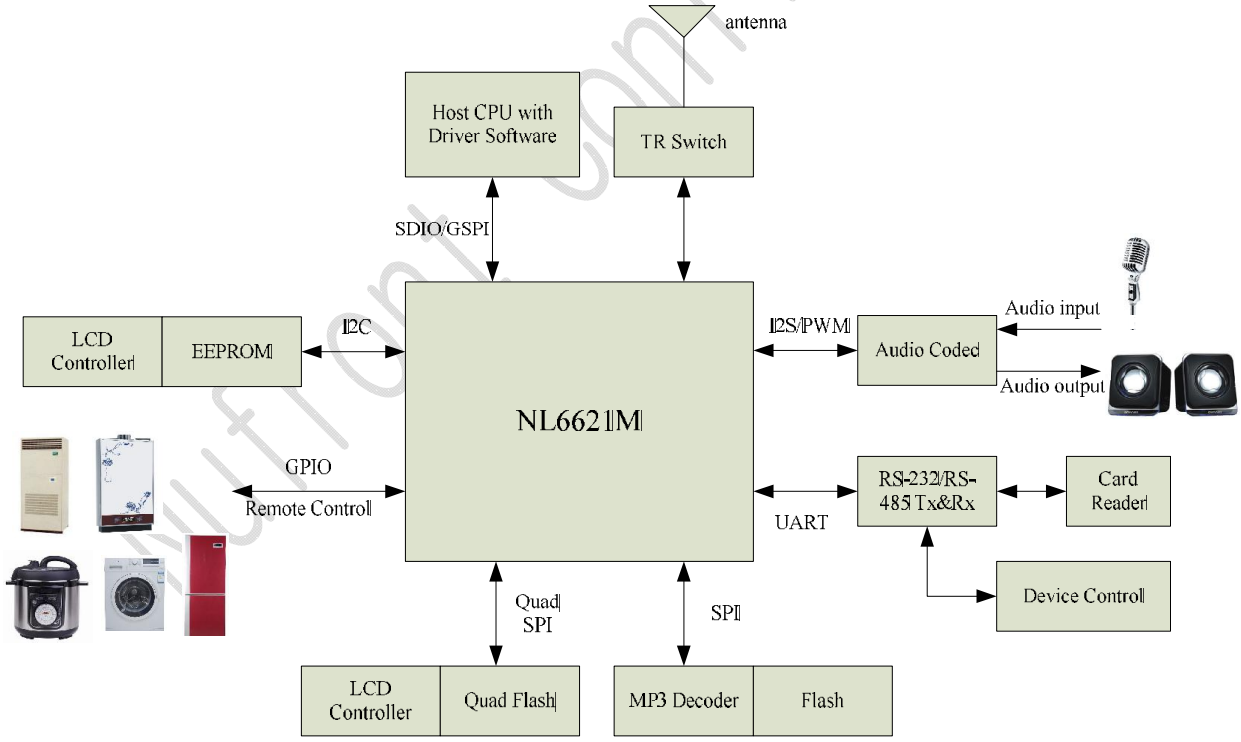
版本	1.2
日期	2014.8.28
作者	周朝显
修改	刘健生

Table of Contents

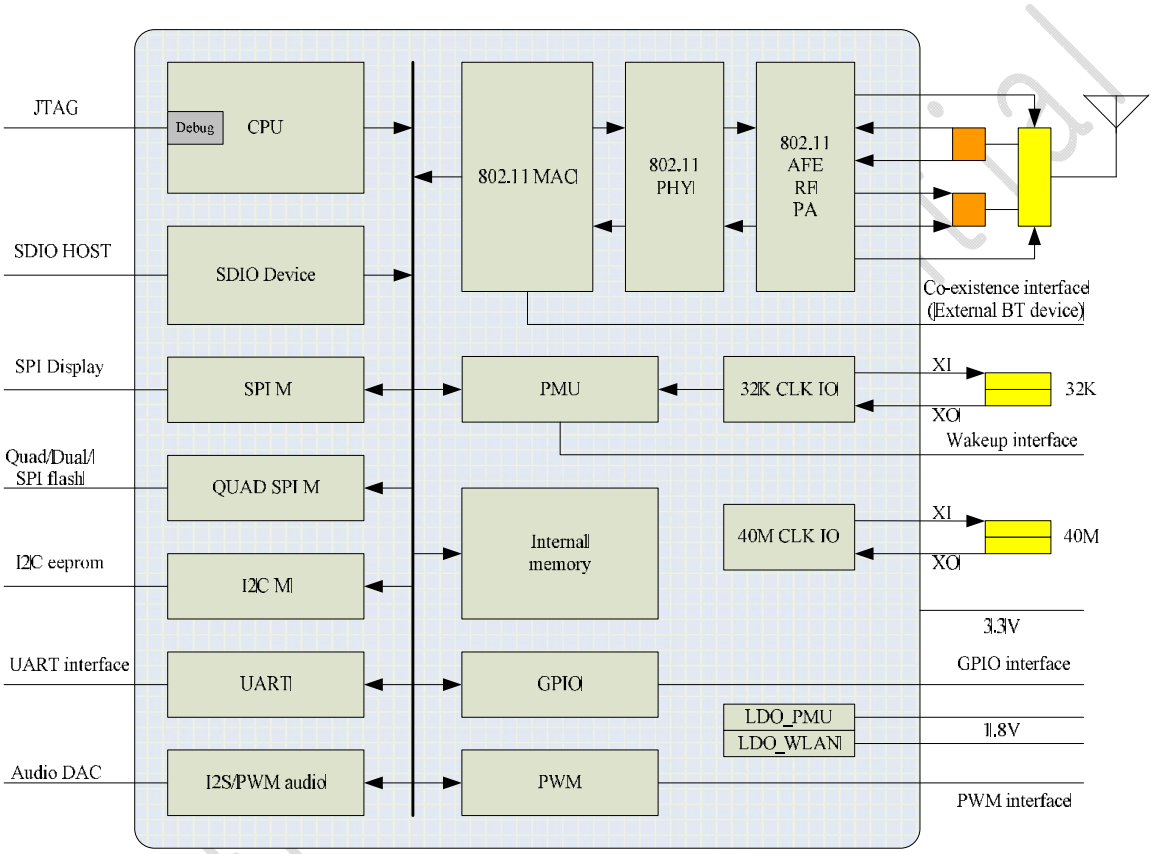
1. 概述

新岸线NL6621M是一款高集成度SOC，特别为高数据吞吐率低成本的无线局域网产品而设计。它集成了MCU，MAC，1T1R基带和带功放RF收发机于一颗芯片上。NL6621M支持802.11b/g/n和Wi-Fi direct（SDIO网卡模式），BSS STA，软AP，WiFi保护设置。它还支持WMM-PS和WPA/WPA2安全协议。

NL6621M既可作受主处理器控制的从设备也可作不需要任何其他处理器的独立WLAN设备。它支持的通用接口包括QuadSPI，标准SPI，I2C，UART，GPIO，I2S音频，PWM音频。NL6621M可与如音箱，摄像头，闪存，液晶显示器等丰富的周边设备直接相连。TCP/IP，UPD，HTTP等丰富的互联网协议已被集成于NL6621M。高性能的无线传输，丰富的接口和协议支持使NL6621M成为无线音频，无线视频，无线智慧家庭和无线医疗等单芯片解决方案的最佳选择。



2. 框图



3. 特性

3.1.芯片集成

WLAN 单芯片 支持 802.11 b/g/n, 它集成了：

1， Radio/ LNA /AFE/MAC/PHY
2， 高效 PA，最大输出功率：17dBm(11b), 16dBm(11n)
3， LDOs，将 1.8V电压转化成1.2V供电电压
4， 集成片上CPU和程序/数据SRAM，支持串行在线调试口。.
5， 集成Quad SPI，标准SPI，UART，I2S音频口，PWM音频口，I2C，PWM，GPIO，硬Timer，硬看门狗Timer。

3.2.电源管理

自适应功耗管理和低功耗设计实现优异的功耗指标:

1， 深睡眠时，内核消耗电流：10uA
2， 浅睡眠时，内核消耗电流：3.7mA
3， 帧发送时，典型芯片功耗为：400mW
4， 帧接收时，典型芯片功耗为：175mW
5， 自适应发送功率和发送速率调整，提高实际数据吞吐率，降低运行功耗
6， 结合节能协议进行芯片功耗管理，灵活进行睡眠和唤醒，降低运行功耗.
7， 和主机之间的互相唤醒机制

3.3.高级802.11特性

1， HT20 (2.4GHz) 单流
2， 全/半保护间隔
3， A-MPDU 帧聚合
4， STBC
5， HT混合和HT绿地格式

3.4.QoS 机制支持

1， WMM，WMM-PS
2， 802.11e

3.5.节能协议支持

1， BSS Ps-poll
2， Normal Power Save
3， U-APSD
4， WMM-PS

3.6.安全模式硬件支持

1， WEP
2， WPA/WPA2（TKIP，CCMP）

3.7.支持速率

1， 802.11b：1，2，5.5，11Mbps
2， 802.11g：6，9，12，18，24，36，48，54Mbps
3， 802.11n：6.5Mbps-65Mbps, 7.2Mbps-72.2Mb

3.8.接受灵敏度

1， 11n MCS7: -75dBm
2， 11g 54M: -76dBm; 11g 6M: -93dBm
3， 11b 1Mbps: -97dBm

3.9.支持信道

2.4GHz, 1-14 信道

3.10.蓝牙共存

暂未支持

3.11.主机接口形式

支持接口有如下形式
1， SDIO 2.0，全速和高速
2， UART

3.12.网络工作模式

Wi-Fi 网络工作模式：
1， BSS STA
2， Soft BSS AP
3， WIFI direct（SDIO网卡模式）
4， Ad hoc
5， DirectConfig一键快捷配置

3.13.WPS支持

1， PBC
2， PIN

3.14.片上CPU

1， 主频可以灵活配置为： 160M， 120M，80M，40M；
2， 其次448KB RAM（其中CODE RAM为192K），以供WLAN和客户定制的应用程序使用。

3.15.Quad SPI

1， 支持Quad/Dual/Standard SPI 模式
2， 接口时钟可配置，最高可到120MHz
3， 通过QuadSPI外接高速Flash存储芯片，支持CPU直接寻址.
4， 支持CPU从此Flash进行Boot.

3.16.I2S 音频接口

1， 支持5.1 声道, I2S 音频 输入/输出接口.
（注：无法支持同时输入输出并发）
2， 支持Master 模式
3， 支持Resolution：16-32 bits
4， 数字音量控制：0 ~ 127dB
5， 支持的音频数据模式有：
   a) I2S Philips Standard Mode
   b) I2S Right justified Mode
   c) I2S Left justified Mode

d) I2S DSP Mode
e) I2S User mode

6，支持的音频采样率:

a) 32 KHz
b) 22.05 KHz
c) 44.1 KHz
d) 24 KHz
e) 48 KHz
f) 96 KHz
g) 88.2 KHz

3.17.其余应用接口

其余应用接口:

1，UART，可作为高速数据或控制接口
2，I2C，可外接EEPROM存储芯片
3，集成标准SPI接口，可以外接SPI接口的显示模块
4，集成最多32个GPIO接口
5，集成2路脉冲宽度调制接口
6，支持在线调试接口
7，1个硬定时器和1个硬看门狗

3.18.供电需求

1，1.8V 电源
2，3.3V 电源
注：IO 电压支持范围：1.8V~3.3V

3.19.时钟需求

1，40MHz Crystal 或时钟源，±20PPM
2，32768Hz Crystal或时钟源
注：本芯片支持不提供32768Hz时钟的工作模式

3.20.系统复位

1，系统复位管脚
2，片内集成上电复位

3.21.GPIO 管脚

几乎所有外设接口管脚均可被配置为GPIO s

3.22.封装形式

NL6621M，QFN60，7x7mm^2，Pitch 0.4mm

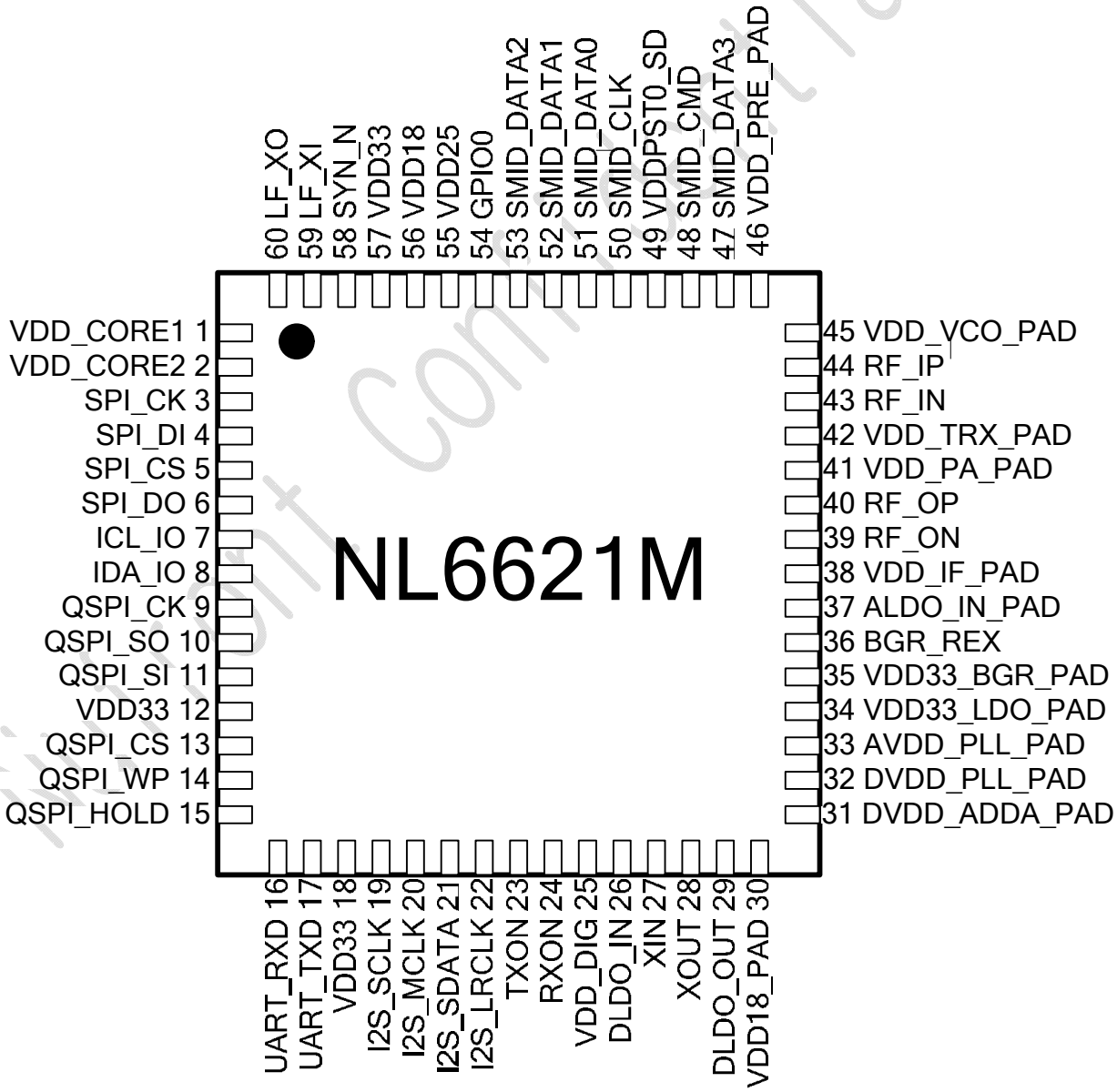
3.23.NL6621M

1，QFN60 封装形式，7x7mm^2
2，本封装形式，有如下特征：

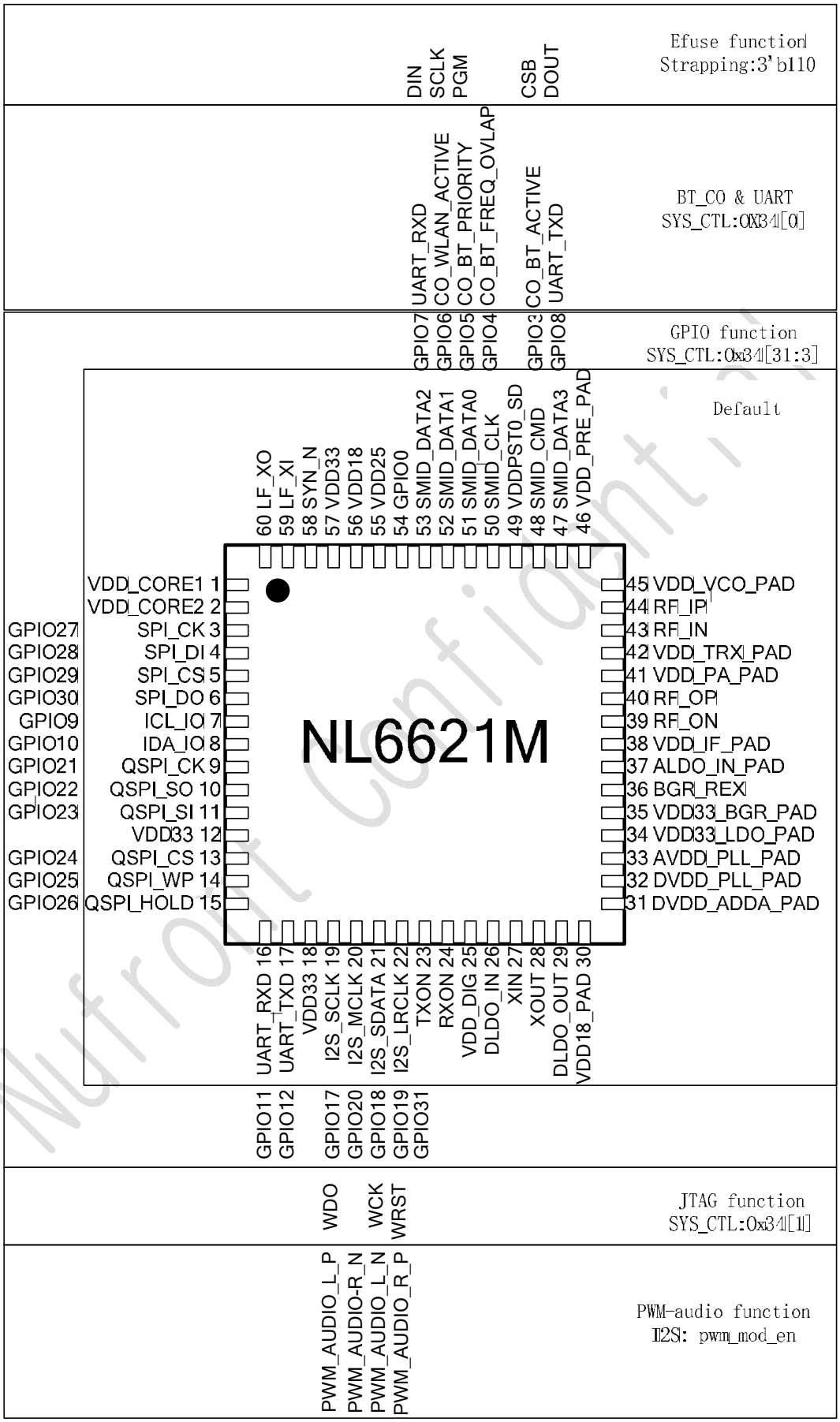
a) 无主机接口形式存在
b) GPIO唤醒主机
c) Quad SPI接口
d) I2S左右两声道/PWM左右两声道接口
e) I2C 接口

4. 封装描述

4.1.封装管脚示意图



4.2.封装管脚复用示意图



5. 管脚描述

5.1.管脚类型

管脚类型	描述
IPU	内部上拉
IPD	内部下拉
I	输入
O	输出
IO	双向
P	电源
G	地
A	模拟
NC	未连接

5.2.管脚描述

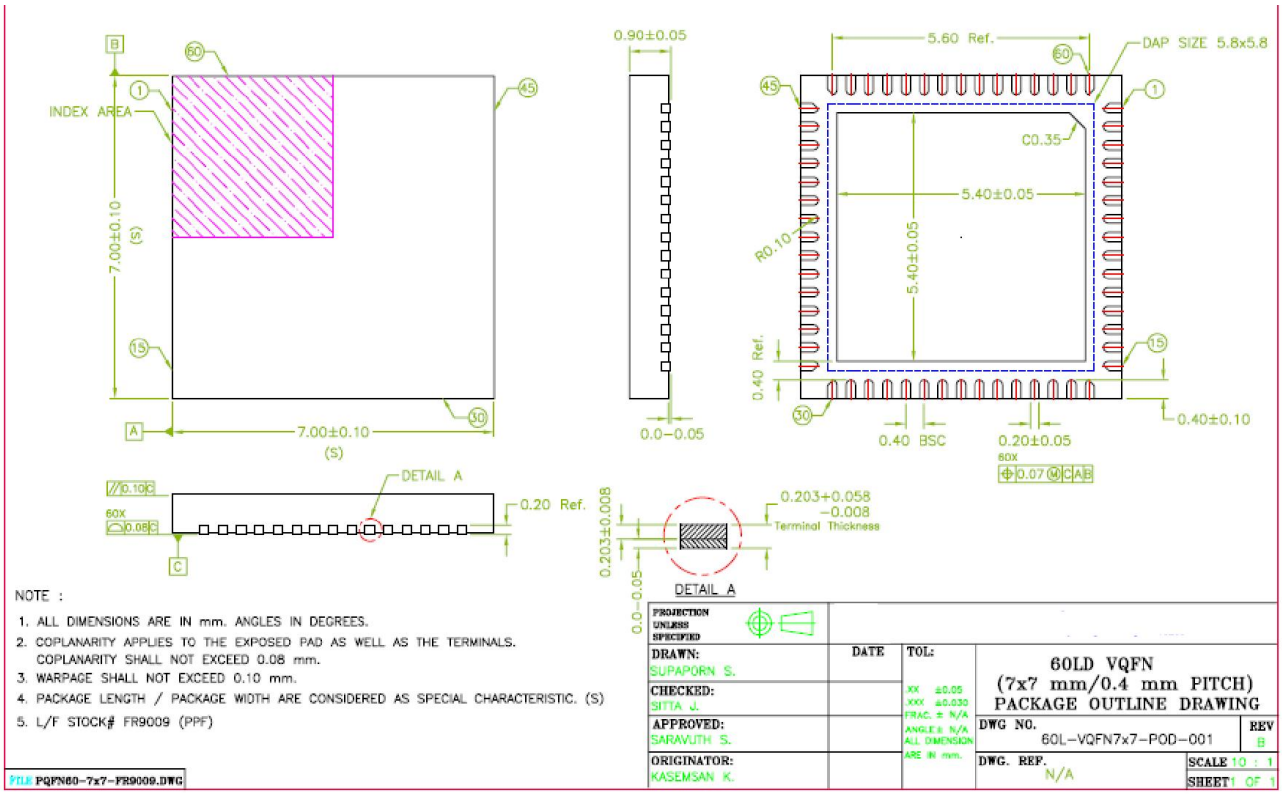
管脚	编号.	IO 类型	描述
VDD_CORE1	1	O	1.2V LDO output , Bypass with a capacitor as close to the pin as possible.
VDD_CORE2	2	O	1.2V LDO output, Bypass with a capacitor as close to the pin as possible.
SPI_CK	3	I/O	SPI clock when NL6621M as SPI master, can also be configured as GPIO
SPI_DI	4	I/O	SPI data in when NL6621M as SPI master, can also be configured as GPIO
SPI_CS	5	I/O	SPI chip select when NL6621M as SPI master, can also be configured as GPIO
SPI_DO	6	I/O	SPI data out when NL6621M as SPI master, can also be configured as GPIO
ICL	7	I/O	I2C clock when NL6621M as I2C master, pull up outside, can also be configured as GPIO
IDA	8	I/O	I2C data when NL6621M as I2C master, pull up outside, can also be configured as GPIO
QSPI_CK	9	I/O	QSPI clock when NL6621M as QSPI master, can also be configured as GPIO

QSPI_SO	10	I/O	QSPI data out when NL6621M as QSPI master, can also be co nfigured as GPIO, strapping pin, see blow description
QSPI_SI	11	I/O	QSPI data input when NL6621M as QSPI master, can also be c onfigured as GPIO
VDD33	12	P	3.3V power supply for IO interface
QSPI_CS	13	I/O	QSPI chip select when NL6621M as QSPI master, can also be configured as GPIO
QSPI_WP	14	I/O	QSPI write protect when NL6621M as QSPI master, can also be configured as GPIO, strapping, see below description
QSPI_HOLD	15	I/O	QSPI hold when NL6621M as QSPI master, can also be configu red as GPIO, strapping with QSPI_DO,QSPI_WP as: {QSPI_HOLD, QSPI_WP, QSPI_DO} 000: load firmware from SDIO/SPI 001, load firmware from i2c EEPROM 010: load firmware from QSPI Flash 011: load firmware from UART 100: SW Debug Mode only when QFN88 Package 101: QSPI flash execution directly 110: efuse burning mode 111: reserved
UART_RX	16	I/O	UART RXD, can also be configured as GPIO
UART_TX	17	I/O	UART TXD, can also be configured as GPIO
VDD33	18	P	3.3V power supply for IO interface
I2S_SCLK	19	I/O	I2S SCLK, can also be configured as GPIO
I2S_MCLK	20	I/O	I2S master clock, can also be configured as GPIO
I2S_SDATA	21	I/O	I2S DATA, can also be configured as GPIO
I2S_RLCLK	22	I/O	I2S R/L clock, can also be configured as GPIO
TXON	23	O	TX mode enable digital input ,set high to enable TX
RXON	24	O	RX mode enable digital input,set high to enable RX
VDD_DIG	25	P	3.3V power supply for Digital IO post-drive voltage
DLDO_IN	26	P	1.8V supply for AFE-LDO's
XIN	27	Crystal Input	40 MHz crystal oscillator input or external clock input
XOUT	28	Crystal Output	40 MHz crystal oscillator output
DLDO_OUT	29	O	IF-LDO's 1.2V output, Bypass with a capacitor as close to the pi n as possible.
VDD18_PAD	30	P	1.8V power supply for RF-LDOs'
DVDD_ADDA_PAD	31	O	LDO's output. Bypass with a capacitor as close to the pin as po ssible.
DVDD_PLL_PAD	32	O	LDO's output. Bypass with a capacitor as close to the pin as po

			ssible.
AVDD_PLL_PAD	33	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
VDD33_LDO_PAD	34	P	3.3V power supply
VDD33_BGR_PAD	35	P	3.3V power supply
BGR_REX	36	O	24Kom resister
ALDO_IN_PAD	37	P	1.8V power supply
VDD_IF_PAD	38	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
RF_ON	39	O	PA's negative output
RF_OP	40	O	PA's positive output
VDD_PA_PAD	41	P	3.3V power supply
VDD_TRX_PAD	42	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
RF_IN	43	I	LNA's negative port input
RF_IP	44	I	LNA's positive port input
VDD_VCO_PAD	45	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
VDD_PRE_PAD	46	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
SD_DATA3	47	I/O	SDIO data0 pin, can also be configured as GPIO
SD_CMD	48	I/O	SDIO CMD pin, can also be configured as GPIO
VDDPST_SD	49	P	3.3V power supply for SDIO interface
SD_CLK	50	I	SDIO CLK pin, can also be configured as GPIO
SD_DATA0	51	I/O	SDIO data0 pin, can also be configured as GPIO
SD_DATA1	52	I/O	SDIO data1 pin, can also be configured as GPIO
SD_DATA2	53	I/O	SDIO data2 pin, can also be configured as GPIO
GPIO0	54	I/O	Multi-function multiplexed pin. **Funciont 1:** GPIO0 for general purpose IO usage after system reset or power on reset (Default). It can be used as input wake up signal from host to make the chip recovers from sleeping state. **Function 2:** Input pin for strapping register **NO_32K_MODE.** The input value (Usually by external pull up/down) before power on reset or system reset released is latched to register **NO_32K_MODE** at the time when the reset is releasing. After reset released, the latched value in register **NO_32K_MODE** will keep unchanged until the next reset happens and the pin is ready for use in function 1 or function 2. The register **NO_32K_MODE** is defined as:

			1'b1: No external 32.768 KHz crystal/ oscillator mode. The 32.768 Khz clock for active clock in sleeping state is generated by divided clock from 40MHz in this mode. It's the lowest system cost design by saving a 32.768 Khz crystal/ oscillator. **1'b0:** External 32.768 KHz crystal/ oscillator mode. The 32.768 Khz clock for active clock in sleeping state is from external 32.768 Khz crystal/ oscillator. 40 MHz oscillator can be powered off to achieve the lowest power consumption in sleeping state.
VDD25	55	P	2.5V power supply for EFUSE write； Normal Condition，this pin is floating
VDD18	56	P	1.8V power supply for LDO
AVDD33	57	P	3.3V power supply for IO
RSTN	58	I	Chip reset input pin. Tie this pin HIGH if only use on chip power on reset. Connect this pin to system reset if you want to reset the chip from other components in the system.
LF_XIN	59	Crystal Input	32.768 KHz crystal input or external clock input
LF_XOUT	60	Crystal Output	32.768 KHz crystal output

6. 封装尺寸



7. 供电电源

7.1.供电管脚

供电电源电压为需求3.3V和1.8V两种

QFN60封装时，3.3V供电管脚为Pin 12，18，25，34，35，41，49，57共8个管脚；

QFN60封装时，1.8V供电管脚为Pin 26，30，37，56共4个管脚；

其次Pin 55为 2.5V供电管脚，仅仅在对写Efuse时，才进行2.5V供电，此电源管脚默认需加滤波电容。

VDD33	12	P	3.3V power supply for IO interface
VDD33	18	P	3.3V power supply for Digital IO interface
VDD_DIG	25	P	3.3V power supply for Digital IO post-drive voltage
DLDO_IN	26	P	1.8V supply for AFE-LDO's
VDD18_PAD	30	P	1.8V power supply for RF-LDOs'
VDD33_LDO_PAD	34	P	3.3V power supply
VDD33_BGR_PAD	35	P	3.3V power supply
ALDO_IN_PAD	37	P	1.8V power supply
VDD_PA_PAD	41	P	3.3V power supply
VDDPST_SD	49	P	3.3V power supply for SDIO interface
VDD25	55	P	2.5V power supply for efuse write，
VDD18	56	P	1.8V power supply for LDO
AVDD33	57	P	3.3V power supply for IO

7.2.LDO输出滤波管脚

共10个LDP输出的滤波管脚；

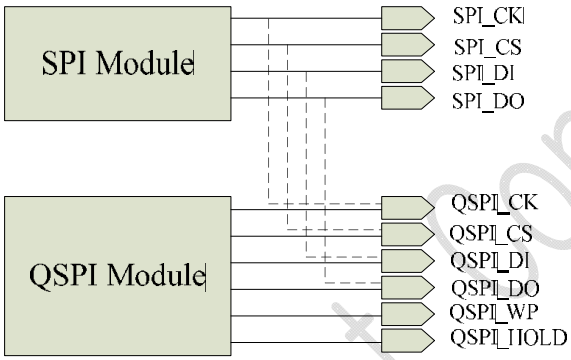
VDD_CORE1	1	O	1.2V LDO output , Bypass with a capacitor as close to the pin as possible.
VDD_CORE2	2	O	1.2V LDO output, Bypass with a capacitor as close to the pin as possible.
DLDO_OUT	29	O	IF-LDO's 1.2V output, Bypass with a capacitor as close to the pin as possible.
DVDD_ADDA_PAD	31	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
DVDD_PLL_PAD	32	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
AVDD_PLL_PAD	33	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
VDD_IF_PAD	38	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
VDD_TRX_PAD	42	O	LDO's output. Bypass with a capacitor as close to the pin as possible.
VDD_VCO_PAD	45	O	LDO's output. Bypass with a capacitor as close to the pin as possible.

VDD_PRE_PAD	46	O	LDO's output. Bypass with a capacitor as close to the pin as possible.

8. 功能描述

8.1.SPI Master接口

共4个管脚，SPI_CK最高频率为20MHz，最低可以到KHz级；
SPI module的管脚可以map到QSPI的管脚，进行访问QSPI管脚上连接的外部设备。示意图如下。



8.2.I2C Master接口

支持：

1，*standard mode (0 to 100 kbit/s)*

2，*fast mode (400 kbit/s)*

3，*high speed mode (3.4 Mbit/s)*

8.3.QSPI Master接口

QSPI模块支持standard SPI mode，Dual SPI mode，Quad SPI mode;
其次QSPI的时钟频率，可以配成如下频率：

MHz	120	60	30	15	7.5	3.75	1.875	0.9375
MHz	80	40	20	10	5	2.5	1.25	0.625

8.4.UART接口

NL6621M仅仅支持两线的UART，不支持流量控制；
支持的最大传输波特率为 40MHz/16=2.5MHz 。

8.5.I2S Audio接口

本芯片内部不集成小数分频PLL，所以I2S Audio接口无法提供MCLK输出，仅可接收MCLK输入。

8.6.PWM Audio接口

PWM Audio接口，和I2S Audio共用管脚

8.7.SDIO Device接口

支持全速模式，最高25MHz时钟频率
支持高速模式，最高50MHz时钟频率
支持1bit模式和4bit模式，切记在1bit模式，需要将不用的SDIO管脚做上拉；
支持SDIO中断申请
支持Suspend和Resume
支持SPI Device模式

9. Strapping Pins

9.1.固件加载模式

固件加载模式取决于三个复用为strapping pin的管脚pin15，pin14，pin10，即{QSPI_Hold, QSPI_WP, QSPI_SO}，这三个管脚可以在板子上做好电平的上下拉，芯片内部逻辑将在芯片复位期间，锁存上三个管脚的输入电平，根据其值，现在固件的加载方式，具体映射表如下：

Strapping Pins Value {Pin15，Pin14，Pin10} ={QSPI_Hold,QSPI_WP,QSPI_SO}	描述
000	从SDIO（SDIO也可工作在SPI Slave模式下）加载固件
001	从I2C EEPROM加载固件
010	从QSPI Flash加载固件
011	从UART加载固件
100	SW Debug模式下的Remap（特定存储空间地址映射），可以使用Jlink进行固件下载和调试
101	从QSPI Flash直接执行的Remap（特定存储空间地址映射）
110	EFUSE Program 模式
111	保留

9.2.是否提供32K时钟的模式

Pin54，即GPIO0，也是一个Strapping Pin，在芯片复位时，芯片内部逻辑锁存此管脚输入值，决定是否需要外部提供32.768KHz时钟。

Strapping Pin Value Pin 54=GPIO0	描述
1	不需要外部提供32.768KHz时钟；即使在芯片休眠时，40MHz时钟，也不再关断，由其分频提供内部所需的32.768KHz时钟
0	需要外部提供32.768KHz时钟；在芯片时，40MHz时钟即使在休眠时，将被关断。

10. 地址空间映射

10.1.AHB地址空间

决定固件加载模式的三根Strapping Pin为 {QSPI_HOLD, QSPI_WP, QSPI_DO}

基地址映射表

BLOCK	Base Address			Size
	Strapping Pins 100/101	Strapping Pins= 100（SW Debug 模式）	Strapping Pins= 101 （QSPI Flash直接执行模式）	
CODE ROM	0x0000_0000	0x0007_0000	0x0100_0000	64KB
CODE SRAM	0x0001_0000	0x0000_0000	0x0101_0000	192KB
CODE SRAM0	0x0008_0000		0x0108_0000	96KB
External QSPI Flash	0x0100_0000		0x0000_0000	Flash Size
BUF SRAM1	0x2000_0000			96KB
BUF SRAM2	0x2004_0000			64KB
APB bridge 0	0x4000_0000			256KB
APB bridge 1	0x4004_0000			256KB
MAC AHB Slave0	0x4008_0000			128KB
MAC AHB Slave1	0x400a_0000			128KB
SDIO AHB Slave	0x400c_0000			256KB
PMU AHB Slave	0x4010_0000			156KB
QSPI reg	0x4014_0000			256KB
GDMA reg	0x4018_0000			256KB

10.2.APB地址空间

Block	Base address	Size

APB Bridge 0		
SPI	0x4000_0000	1KB
Timers	0x4000_1000	1KB
PHY APB	0x4000_2000	8KB
Watch dog 0	0x4000_4000	1KB
I2C	0x4000_5000	1KB
Iq_calitration	0x4000_6000	1KB
Rf_spi	0x4000_7000	1KB
PWM	0x4000_9000	1KB
APB Bridge 1		
GPIO	0x4004_0000	1KB
UART	0x4004_1000	1KB
Effuse	0x4004_2000	1KB
I2S（I2S & PWM audio）	0x4004_3000	1KB

11. SDIO寄存器（0x400C_0000）

11.1 SDIO寄存器地址偏移量

Signal	Registers
0x00	Control Register
0x04	Command Register
0x08	Argument Register
0x0C	Block Count Register
0x10	DMA1 Address Register
0x14	DMA1 Control Register
0x18	DMA2 Address Register
0x1C	DMA2 Control Register
0x20	Erase Write Block Start Register
0x24	Erase Write Block End Register
0x28	Password Length Register
0x2C	Secure Block Count Register
0x30	Reserved for future use

0x34	Reserved for future use
0x38	Reserved for future use
0x3C	Interrupt Status Register
0x40	Interrupt Status Enable Register
0x44	Interrupt Signal Enable Register
0x48	Card Address Register
0x4C	Card Data Register
0x50	IOREADY Register
0x54	Function1 Control Register
0x58	Function2 Control Register
0x5C	SDIO CCCR Control Register
0x60-0x7C	SDIO FBRx Control Registers
0x80	Card Size Register
0x84	Card OCR Register
0x88	Control2 Register
0x90	Function3 Control Register
0x94	Function4 Control Register
0x98	Function5 Control Register
0x9C	Interrupt Status2 Register
0xA0	Interrupt Status Enable2 Register
0xA4	Interrupt Signal Enable2 Register
0x8C-0xFF	Reserved for future use
0x100	CARD_REVISION_REG
0x104	CARD_FW_STATUS0_REG/CARD_FW_STATUS1_REG
0x108	CARD_STATUS_REG
0x10C	HOST_F1_RD_BASE_0/ HOST_F1_RD_BASE_1
0x110	WR_BITMAP
0x114	HOST_PWR_CTRL
0x118	RD_LEN_P1
0x11C	RD_LEN_P2
0x120	RD_LEN_P3
0x124~	Reserved for future use

11.2SDIO寄存器说明

11.2.1 Control Register

Control register(0x00)				
Register Field	Offset	Access	Default Value	Description

Program Done	0	RWAC	1’b 0	The processor sets this bit after completion ofProgramming / Erase Operation / CMD43 / CMD20Interrupt.
Reserved	1	Rsvd	1’b 0	**Reserved for future.**
Card Init Done	2	RWAC	1’b 0	On Program Start Interrupt, the processor will start programming the CSR and SD/MMC/SDIO card registers. Processor set this bit to 1, once it is done with the Initialization. 1 – Card is ready to Operate 0 – Card is busy This bit reflects in bit31 of CMD1 and ACMD41 response. Note Device Controller clear this bit on soft reset.
Address Out of Range	3	RWAC	1’b 0	This bit reflects in Card Status Register. 1 - A multiple block or stream read/write operation is (although started in a valid address) attempting to read or write beyond the card capacity 0 – No Error Note: The Processor set this bit, only for infinite or stream transfers. Address out of range for R type is handled inside the controller.
Address Misalign	4	RWAC	1’b 0	This bit reflects in Card Status Register. 1 - A multiple block read/write operation (although started with a valid address/blocklength combination) is attemptin g to read or write a data block which does not align with the physical blocks of the card. 0 – No Error Note: Address Misalign for R type is handled inside the controller. For SD Configuration, the alignment is always done based on 512 byte blocks, regardless of the CSD values, whereas for MMC, Read Operation, the alignment is based on READ_BL_LEN value and similarly for MMC write operation, the

				alignment is based on WRITE_BL_LEN value.
rpmb_dis_en	5	RW	1’b 0	1 – Set this bit when CMD6 (switch) is received by the firmware set to PARTITION_ACCESS 0 - Clear this bit when CMD6 (switch) is received by the firmware to clear the PARTITION_ACCESS The SDMMC SDIODevice Controller will consider only class0/2/4 commands and the rest of the commands are treated as invalid
Erase Param	6	RWAC	1’b 0	This bit reflects in Card Status Register. . 1 - An invalid selection of erase groups for erase occurred. 0 – No Error
Card ECC Failed	7	RWAC	1’b 0	This bit reflects in Card Status Register. 1 - Card internal ECC was applied but failed to correct the data 0 – No Error
CC Error	8	RWAC	1’b 0	This bit reflects in Card Status Register. 1 - A card error occurred, which is not related to the host command. 0 – No Error
Error	9	RWAC	1’b 0	This bit reflects in Card Status Register. A generic card error related to the (and detected during) execution of the last host command (e.g. read or write failures). 0 – No Error
MMC_IRQ_Trigger	10	RWAC	1’b 0	This bit is used in MMC Interrupt Mode. Whenever this bit is set, the PE-SMID Device Controller will send its response to the host. . Note: The device controller will send the response only in Interrupt mode. The device controller will ignore the M

				MC_IRQ_Trigger events in other states.
CMD_Data_Output_Edge	11	RW	1’b 0	Command and Data Output Edge: The SD/MMC/SDIO Device controller drive the data and cmd lines based on this bit. 0 – Drive the Command and Data line at the falling edge of SD/MMC/SDIO clock 1 – Drive the Command and Data line at the Rising edge of SD/MMC/SDIO clock
CMD32_CMD33_Enable	12	RW	1’b 0	0 – Disabled. The SD/MMC/SDIO Device Controller consider cmd32/cmd33 as illegal command. 1 – Enabled. The SD/MMC/SDIO Device Controller accept the Erase sequence with cmd32 and cmd33. In other words, Host can issue both Erase sequences 1. CMD35, CMD36, CMD38 2. CMD32, CMD33, CMD38. Note: This field is applicable only to MMC mode. This is mainly to compatible with MMC3.31 spec.
Boot Sequence Support	13	RW	1’b 0	0 – Boot Sequence is not supported. 1 – Boot Sequence is supported.
Switch Error	14	RWAC	1’b 0	0 – No Switch Error 1 – Switch Error This bit reflects in card status register. The response type is “X”. The Processor has to set this bit, whenever there is a switch error, including data width setting
Boot_ACK	15	RW	1’b 0	0 – No Boot Ack 1 – Send Boot Ack
WP_Violation	16	RWAC	1’b 0	0 – No WP Violation 1 – WP Violation
WP_Erase_Skip	17	RWAC	1’b 0	0 – No WP Erase Skip 1 – WP Erase Skip
CID_CSD_Overwrite	18	RWAC	1’b 0	0 – No CID/CSD Overwrite Error 1 – CID/CSD Overwrite Error

AKE_Seq_ Error	19	RWAC	1’b 0	0 – No AKE Sequence Error 1 – AKE Sequence Error
Card_ECC _Disabled	20	RW	1’b 0	0 – ECC Enabled 1 – ECC Disabled
Stream Threshold Size	23:21	RW	3’b 100	000 – 32Bytes 001 – 64Bytes 010 – 128Bytes 011 – 256Bytes 100 - 512Bytes 101 – 1KBytes 110 – 2KBytes 111 – Reserved for Future Use The Internal DMA engine uses this threshold value to do Stream write or Stream read operations with the system memory. Instead of waiting for a block size amount of space or data (Read or Write operation), the internal dma engine wait for stream threshold amount of space or data or end of transaction for Stream read or write operations.
Permanent Write Protect	24	RW	1’b 0	0 – The card is not permanently Write protected 1 – The card is permanently write protected This field is required for internal lock unlock logic.
Temporary Write Protect	25	RW	1’b 0	0 – The card is not Temporary Write protected 1 – The card is Temporary write protected This field is required for internal lock unlock logic
WP Commands Enabled	26	RW	1’b 0	0 – WP Commands are Disabled. 1 – WP Commands are Enabled
ALLOW_AKE	27	RW	1’b 0	This bit determines whether to allow AKE commands or not. The Processor set this bit after both GET_MKB and GET_MID were executed. The Controller clear this bit after power on reset or cmd0 soft reset. 1- Allow AKE commands (ACMD45-48) 0 - Ignore (ACMD45 – 48) command

				s and treat as illegal commands.
SECURED_MODE	28	RW	1’b 0	This bit determines whether to allow protected area access commands (ACMD18, ACMD25, ACMD26 ACMD38, and ACMD49) or not. Processor set this bit after successful AKE sequence. The Controller clear this bit after ACMD45, ACMD46, ACMD47, ACMD18, ACMD25, ACMD26, ACMD38, or ACMD49 1 – Allow protected area access commands(ACMD18, ACMD25, ACMD26, ACMD38, ACMD49) 0 – Ignore protected area access commands and treat as Illegal commands.
AKE_SEQ_OK	29	RW	1’b 0	The Processor set this bit after AKE was successful upto ACMD47. The controller clears this bit, when SECURED_MODE is cleared. 1 – AKE sequence is ok 0 – AKE sequence is not ok. The controller set AKE_SEQ_ERROR in the R1 response of the next ACMD48 if AKE_SEQ_OK is cleared. If AKE_SEQ_OK is set, then the Controller clear AKE_SEQ_ERROR in the R1 response of the next ACMD48.
assd_dis_en	30	RW	1’b 0	0 – ASSD Commands are Disabled. 1 – ASSD Commands are Enabled
boot_data_rdy	31	RW	1’b 0	0-Boot data is not ready from Firmware 1-Boot data is ready from Firmware

11.2.2 Command Register

Command register (0x04)				
Register Field	Offset	Acces	Default value	Description
Application	0	ROC	1’b 0	1 – Current command is an Application Command 0 – Not an Application Command.
Block Size	12:1	ROC	12’b 0	This field denotes the size of the data

				block. 12’d 0 – Reserved 12’d 1 – 1 Byte 12’d 2 - 2 Bytes ...
Command Index	18:13	ROC	6’b 0	This field denotes the Index of the Current command
Current Bus Width	20:19	ROC	2’b 0	Denotes the current Bus Width 00 - 1 Bit 01 - 4 Bits 10 - 8 Bits 11 – Reserved
Current Speed	23:21	ROC	3’b 0	Defines the Speed Class Control Bits 0000b: Start Recording 0001b: Create DIR 0010b: Reserved for Future Use 0011b: Reserved for Future Use 0100b: Update CI Others: Reserved
Card state	27:24	ROC	4’b 0	Defines the current state of the Controller. 0 = Idle 1 = Ready 2 = Ident 3 = Stby 4 = Tran 5 = Data 6 = Rcv 7 = Prg 8 = Dis 9 = Btst (Applicable only for MMC Card) 10 = Slp (Applicable only for MMC Card) 11 – 15 = reserved
Erase Sequence	28	ROC	1’b 0	Erase Sequence This bit reflects the Host Erase sequence 0 – Erase Sequence with CMD32, CMD33, CMD38 has occurred 1 – Erase Sequence with CMD35, CMD36, CMD38 has occurred.
Reserved	31:29	Rsvd	4’b 0	Reserved for Future Use

11.2.3 Argument Register

Argument register (0x08)				
Register Field	Offset	Access	Default value	Description
Argument	31:0	ROC	32’b 0	This field denotes the 32bit argument of SD/MMC/SDIO command

11.2.4 Block Count Register

Block Count Register (0x0C)				
Register Field	Offset	Access	Default Value	Description
Block Count	31:0	ROC	32’b 0	This regiser should be accessed only when no transaction is executing. During data transfer, read operations on this register may return an invalid value and write operations are ignored. 32’d 0 – Block count is 0 32’d 1 – Block Count is 1. 32’d 2 – Block Count is 2 .. .. Incase of Infinite transfer, the block count is initialized to 32’h FFFF_FFFF. So with posede SMID controller, 32’h FFFF_FFFF blocks can be transferrred in single write or read command. Note : When SMID_SD_VER_SEL is set to SD2.0 and on read by the FIRMWARE will get 32'hFFFF_FFFF only.

Note:

Read Operation [Host Device]:

On receiving the read start interrupt, the processor has to read this Block Count register to find

the number of blocks host is going to read from system memory. In case of Data crc error / Data end bit error), Host may send an abort command in the middle of a data tranasction. The

PESMID controller will assert Func_crc_end error interrupt in cases of error transactions. On

receiving this func_crc_end error interrupt, the processor willl read the block count regis ter, to find the exact number of blocks read from system memory. Incase of infinite tra nsfers, i.e. Block count is initialized with 32'h FFFF_FFFF, the processor wait for transa ction complete interrupt and then it will read the block count register to find the exact number of blocks read from system memory.

Write Operation [Host Device]:

On receiving the write start interrupt, the processor can read the Block Count register t o find the number of blocks host intend to write. The processor can use this informatio n for memory

allocation. In order to know the actual number of blocks got transferred to system mem ory, the processor will read this block count register at the end of the transaction. i.e. wait till transfer complete interrupt for normal or infinite transfers or fun_crc_end error in terrupt for error transactions.

11.2.5 DMA1 Address Register

DMA1 Address Register (0x10)				
Register Field	Offset	Access	Default Value	Description
DMA1 Address Register	31:0	RW	32'b 0	32bit DMA1 Address Register

11.2.6 DMA1 Control Register

DMA1 Control Register (0x14)				
Register Field	Offset	Access	Default Value	Description
DMA1 Address Valid	0	RWAC	1'b 0	1 – DMA1 Address register and DMA 1 Buffer Size is valid 0 – DMA1 Address register and DMA 1 Buffer Size is not valid
DMA1 Buffer Size	3:1	RW	3'b 0	DMA1 Buffer Size 000b 4K bytes (Detects A11 carry out) 001b 8K bytes (Detects A12 carry out) 010b 16K Bytes (Detects A13 carry ou t) 011b 32K Bytes (Detects A14 carry ou t) 100b 64K bytes (Detects A15 carry ou t)

				101b 128K Bytes (Detects A16 carry out) 110b 256K Bytes (Detects A17 carry out) 111b 512K Bytes (Detects A18 carry out)
Reserved	31:4	Rsvd	28’b 0	Reserved for Future Use

11.2.7 DMA2 Address Register

DMA2 Address Register (0x18)				
Register Field	Offset	Access	Default Value	Description
DMA2 Address Register	31:0	RW	32’b 0	32bit DMA2 Address Register

11.2.8 DMA2 Control Register

Control Register (0x1C)				
Register Field	Offset	Access	Default Value	Description
DMA2 Address Valid	0	RWAC	1’b 0	1 – DMA2 Address register and DMA 2 Buffer Size is valid 0 – DMA2 Address register and DMA 2 Buffer Size is not valid
DMA2 Buffer Size	3:1	RW	3’b 0	DMA2 Buffer Size 000b 4K bytes (Detects A11 carry out) 001b 8K bytes (Detects A12 carry out) 010b 16K Bytes (Detects A13 carry out) 011b 32K Bytes (Detects A14 carry out) 100b 64K bytes (Detects A15 carry out) 101b 128K Bytes (Detects A16 carry out) 110b 256K Bytes (Detects A17 carry o

				ut) 111b 512K Bytes (Detects A18 carry out)
Reserved	31:4	Rsvd	28’b 0	Reserved for Future Use

Note: The current PE-SMID supports only DMA1 address register and DMA2 register is kept for future purpose.

11.2.9 Erase Write Block Start Register

Erase Write Block Start Register (0x20)				
Register Field	Offset	Access	Default Value	Description
Erase Write Block Start	31:0	ROC	32’b 0	This field denotes the Starting Write Block Address for Erase Operation.

11.2.10 Erase Write Block End Register

Erase Write Block End Register (0x24)				
Register Field	Offset	Access	Default Value	Description
Erase Write Block End	31:0	ROC	32’b 0	This field denotes the End Write Block Address for Erase Operation.

11.2.11 Password Length Register

Password Length Register (0x28)				
Register Field	Offset	Offset Access	Default Value	Description
PWDS_LEN	7:0	RW	8’b 0	**PWDS_LEN:** This field denotes the length of the password in Bytes 0 – No password 1 – Password is 1Byte length 2 – Password is 2Bytes length .. .. The processor should program this field only during Card initialization. The c

				ontroller lock state is determined by th is field. The controller will be in locke d state, if this field has a non-zero value. Similarly the controller will be in unlocked state, if the processor progr ams a zero value to this field. The Processor should first program th e Password fields in Register RAM th en the PWD_LEN during card initilizati on to determine the lock state of the controller.
Disable Lock Unlock	8	RW	1’b 0	0 – Lock Unlock Feature is Disabled. 1 – Lock Unlock Feature is Enabled. Note: CMD42 will be treated as illegal command, if the lock unlock feature is disabled.
Reserved	31:9	Rsvd	23’b 0	Reserved for Future Use

11.2.12 Secure Block Count Register

Secure Block Count Register (0x2C)				
Register Field	Offset	Access	Default Value	Description
Secure Block Count	7:0	RW	8’b 0	**Secure Block count:** This field denotes the block count valu e for 1.ACMD18 2.ACMD25 3.ACMD26. 4.CMD50 5.CMD57 On receiving the Write or Read Start i nterrupt, the processor has to read th e Command Register and confirm the above mentioned commands. Then the processor will program the block cou nt in this field and clear the write / re ad start interrupt. The PE-SMID Devic e controller will use this block count v alue for the above mentioned comman ds. Note: The processor will get this block

				count info from ACMD45 / CMD35. ACMD18/ACMD25/ACMD26 use ACMD 45 and similarly CMD50 and CMD57 uses CMD35 for the block count info.
Reserved	31:8	Rsvd	24’b 0	Reserved for Future Use

11.2.13 Interrupt Status Register

Interrupt Status Register (0x3C)				
Register Field	Offset	Access	Default Value	Descrioption
Transfer Complete Interrupt	0	RW1C	1’b 0	For Write operation, the SDMMCSDIO Device Controller asserts the interrupt after sending last byte of the last data block in system bus. For Read Operation, the SDMMCSDIODevice Controller asserts this interrupt, after sending last byte of the last data block in SD/MMC/SDIO bus. The PE-SMID Device controller will determine the last block based on one of these conditions. 1.Whenever the local block count register value reaches the value zero 2.On receiving the abort command **Busy after last block for Write Operations:** The Busy in data0 line behavior differs, based on wr_last_blk_busy bit in control register. If wr_last_blk_busy is set to 1, then the SDMMCSDIODevice Controller will pull the data0 line low after the last data block. The data0 line will be pulled high only when processor sets program_done bit to 1 in control register. In other case, if wr_last_blk_busy is set to 0, then the SDMMCSDIODevice Controller will pull the data0 line low after the last data block. The data0 line will be pulled high immediately after transferring the last byte of

				last data block in system bus.
DMA1 Interrupt	1	RW1C	1’b 0	Asserted high for both Write and Read operation on every page buffer boundary for DMA1.
SLEEP / AWAKE Interrupt	2	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host sets the SLEEP/AWAKE bit for CMD5 in MMC mode. SDMMCSDIODevice Controller Drives busy on DATA0 line and will deassert the busy when the Firmware set the Program Done bit in Control Register SDMMCSDIODevice Controller will ignore the argument in CMD5 when busy is driven in SLEEP state
Write Start Interrupt	3	RW1C	1’b 0	Asserted high, whenever there is a new write command with data transfer from SD/MMC/SDIO Bus.
Read Start Interrupt	4	RW1C	1’b 0	Asserted high, whenever there is a new read command with data transfer from SD/MMC/SDIO Bus.
Password Set Interrupt	5	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host sets the password for the card.
Password Reset Interrupt	6	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host resets the password for the card.
Lock Card Interrupt	7	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host locks the card.
Unlock Card Interrupt	8	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host unlocks the card.
Force Erase Interrupt	9	RW1C	1’b 0	SDMMCSDIODevice Controller will assert this interrupt, whenever host initiates a Force Erase sequence. On receiving the interrupt, the Processor has to clear the temporary write protect (if it is already set) in CSD register, through card address and card data register.

				The device controller will locally clear the password contents.
Erase Interrupt	10	RW1C	1’b 0	This interrupt will get asserted, whenever Host sends an Erase Command. On receiving this Interrupt, the processor has to get the Erase Start Address register and Ease End Address Register from SDMMCSDIOCSR registers and then do an Erase operation for the specified block address. Once Erase operation is done, the processor has to set a bit called “program done” in Control register. The PE-SMID Device Controller will pull the data0 line low (Busy) after asserting the Erase Interrupt. The data0 line will be pulled high, only when processor sets the program done bit in control register.
CMD11 Interrupt	11	RW1C	1’b 0	Command 11 Voltage switch Interrupt. Device Controlller assert this interrupt, whenever host issues cmd11 in SD and SDIO mode. On receiving this interrupt the processor has to switch on the system clock and wait for cmd11 clock stop interrupt. On receiving this Interrupt, the card will start the voltage switching process. Host keeps SDCLK low for 5ms and within this period, the card has to complete the voltage switching process. After 5ms, Host starts providing SDCLK at 1.8V. The device controller detects the SD clock and assert cmd11 clock start interrupt to the processor.
CMD0/CMD52 Soft Reset	12	RW1C	1’b 0	Soft Reset. Asserted, whenever SDMMCSDIOcommand receives a CMD0/CMD52 Soft reset in SD/MMC/SDIO bus.
CMD6-Check Done Interrupt	13	RW1C	1’b 0	PE-SMID Controller will assert this interrupt for SD, eSD and MMC CMD6. On receiving this interrupt, the processor will read the Argument register to find CMD6 Argument from the Host.

CMD6-Switch Done Interrupt	14	RW1C	1’b 0	PE-SMID Controller will assert this interrupt for SD, eSD CMD6 Switch Function and MMC CMD6. On receiving this interrupt, the processor will read the Argument register to find CMD6 Argument from the Host. The CMD6 behaviour slightly differs between SD/eSD and MMC **SD/eSD:** CMD6 is a data transaction command. Apart from this Switch Interrupt, the PE-SMID device controller will also assert read start interrupt to the Processor, inorder to read a block of data from system memory. This Interrupt may be a redundant one for SD/eSD CMD6. PE-SMID device controller handle the following by decoding the SD/eSD CMD6 Argument 1.DDR Mode 2.UHS Speed Note: Through ACMD6, Host will change the data width. **MMC:** CMD6 is an addressed command. Based on the Argument value, the processor has to update the extended CSD via Card Address and Card Data Registers. PE-SMID device controller handle the following by decoding the MMC CMD6 Argument 1.Data Width Switching (1/4/8) 2.DDR Mode
Program CSD Interrupt	15	RW1C	1’b 0	Asserted whenever Host updates the programmable bits in the CSD. On receiving this Interrupt, the processor will read the CSD contents through Card Address and Card Data Registers (Indirect addressing)
ACMD23 Interrupt	16	RW1C	1’b 0	This interrupt will get asserted, whenever Host sends ACMD23. On receiving this Interrupt, the processor has to read the Argument register, to find number of blocks to be pre-erased before

				writing.
CMD20 Interrupt	17	RW1C	1’b 0	This interrupt will get asserted, whenever Host sends CMD20 (SD Speed Class Control). On receiving this Interrupt, the processor has to read the Argument register, to find the Speed Class Control info. The PE-SMID Device Controller will pull the data0 line low (Busy) after asserting the CMD20 Interrupt. The data0 line will be pulled high, only when processor sets the program done bit in control register. Note: The controller will not assert CMD20 interrupt in MMC mode.
Reserved	18	Rsvd	1’b0	Reserved for Future Use
CMD4 Interrupt	19	RW1C	1’b 0	Asserted whenever Host updates the DSR register. On receiving this Interrupt, the processor will read the DSR contents through Card Address and Card Data Registers (Indirect addressing)
Boot Start	20	RW1C	1’b 0	This Interrupt is Asserted based on two conditions. 11Host pulls the cmd line low for greater than or equal to 74 Clock cycles 11CMD0 with Arg 0xFFFF_FFFA On receiving the Interrupt, the Processor has to read the Extended CSD register to find the Boot Partition Area, BOOT_SIZE_MULT etc.
Function1 Reset	21	RW1C	1’b 0	Function1 Reset Interrupt: This bit is set, if SDIO Host performs the following sequence for IOE1 The host can also use IOEx as a per function reset for error recovery. The host sequence for a per function reset is to reset IOEx to 0, wait until IORx becomes 0 and then set IOEx to 1 again. If the error is not recovered by this sequence, SDIO reset should be used noting that the operation of all functions will be aborted.
Function2 Reset	22	RW1C	1’b 0	Function2 Reset Interrupt: This bit is set, if SDIO Host performs the following sequence for IOE2 The

				host can also use IOEx as a per function reset for error recovery. The host sequence for a per function reset is to reset IOEx to 0, wait until IORx becomes 0 and then set IOEx to 1 again. If the error is not recovered by this sequence, SDIO reset should be used noting that the operation of all functions will be aborted.
CMD11_CLK_STOP	23	RW1C	1’b 0	CMD11 Clock Stop Interrupt. Asserted, whenever SD Host stops the SD clock during CMD11 transaction. Card will Start switching voltage at this point. System Clock (AHB) will be used to find the SD clock stoppint.
CMD11_CLK_START	24	RW1C	1’b 0	CMD11 Clock Start Interrupt. Asserted, whenever SD Host starts the SD clock during CMD11 transaction. On receiving this interrupt, the processor has to clear the cmd11 switch interrupt, so that the device controller will start driving cmd and data lines high.
PROGRAM_START	25	RW1C	1’b 0	Asserted high for the following conditions. 1. CMD1 or ACMD41 or CMD5 with a valid voltage range has been received, 2. the device has switched to SPI mode, 3. one of the e•MMC boot modes has been entered. 4. Soft Reset 5. SMID when configured as a Combo card, Program start interrupt will be asserted either for SDIO or SD Memory only. Program start interrupt once asserted for IO will not be asserted for Memory and vice versa. This is to notify the processor, that the operating Processor can now be loaded from the flash memory. For eSD, the processor will know, whether the FAST Boot mode set to 1 or not in ACMD41 argument.
CMD40 Interrupt	26	RW1C	1’b 0	Asserted high, whenever host sends cmd40. The device controller enters Int

				errupt mode and wait for the start bit on SD bus. The device controller will send the response in SD bus, whenever the processor sets the MMC_IRQ_Trigger bit to 1 in control register. Note: The device controller will send the response only in Interrupt mode. The device controller will ignore the MMC_IRQ_Trigger events in other states.
CMD R1b Interrupt	27	RW1C	1’b 0	Asserted high, whenever host sends the following commands. 1.cmd43. 2.cmd37 3.Acmd38 4.Acmd49 5.CMD28 6.CMD29 The PE-SMID Device Controller will pull the data0 line low (Busy) after asserting this CMD R1b Interrupt. The data 0 line will be pulled high, only when processor sets the program done bit in control register. Note: CMD28/CMD29 In case of SD, this interrupt will be asserted only when "WP commands enabled" is set to 1 in control register, whereas for MMC, both "WP commands enabled" and "Boot part enable" should bet set to 1.
FunctionX CRC End Error Interrupt	28	RW1C	1’b 0	FunctionX CRC / End bit Error Interrupt: This bit is set, whenever there is a crc or end bit error on sd bus.
FunctionX Abort Interrupt	29	RW1C	1’b 0	Abort Interrupt: This bit is set, whenever SD/SDIO/MMC Host aborts the FunctionX Write/Read Operation
LRST Interrupt	30	RW1C	1’b 0	LRST Interrupt Used only in eSD mode. This bit is set whenever LRST input goes low. On receiving this interrupt, the processor has to reset the volatil

				e, sticky-write and sticky-read protectio n states.
BOOT COMPLETE Interrupt	31	RW1C	1’b 0	Boot complete Interrupt This bit is set when the Boot operatio n is completed by the PE-SMID.

11.2.14 Interrupt Status Enable Register

Interrupt Status Enable Register (0x40)				
Register Field	Offset	Access	Default Value	Descrioption
Transfer Complete Interrupt	0	RW	1’b 1	1 – Enabled 0 - Masked
DMA2 Interrupt	1	RW	1’b 1	1 – Enabled 0 - Masked
SLEEP / AWAKE Interrupt	2	RW	1’b 1	1 – Enabled 0 - Masked
Write Start Interrupt	3	RW	1’b 1	1 – Enabled 0 - Masked
Read Start Interrupt	4	RW	1’b 1	1 – Enabled 0 - Masked
Password Set Interrupt	5	RW	1’b 1	1 – Enabled 0 - Masked
Password Reset Interrupt	6	RW	1’b 1	1 – Enabled 0 - Masked
Lock Card Interrupt	7	RW	1’b 1	1 – Enabled 0 - Masked
Unlock Card Interrupt	8	RW	1’b 1	1 – Enabled 0 - Masked
Force Erase Interrupt	9	RW	1’b 1	1 – Enabled 0 - Masked
Erase Interrupt	10	RW	1’b 1	1 – Enabled 0 – Masked
CMD11 Interrupt	11	RW	1’b 1	1 – Enabled 0 – Masked
CMD0/CMD52 Soft Reset	12	RW	1’b 1	1 – Enabled 0 – Masked
CMD6-Check Done	13	RW	1’b 1	1 – Enabled 0 – Masked

Interrupt				
CMD6-Switch Done Interrupt	14	RW	1'b 1	1 – Enabled 0 – Masked
Program CSD Interrupt	15	RW	1'b 1	1 – Enabled 0 – Masked
ACMD23 Interrupt	16	RW	1'b 1	1 – Enabled 0 – Masked
CMD20 Interrupt	17	RW	1'b 1	1 – Enabled 0 – Masked
Reserved	18	Rsvd	1'b 1	Reserved for Future Use
CMD4 Interrupt	19	RW	1'b 1	1 – Enabled 0 – Masked
Boot START Interrupt	20	RW	1'b 1	1 – Enabled 0 – Masked
Function1 Reset	21	RW	1'b 1	1 – Enabled 0 – Masked
Function2 Reset	22	RW	1'b 1	1 – Enabled 0 – Masked
CMD11_CLK _START	23	RW	1'b 1	1 – Enabled 0 – Masked
CMD11_CLK _STOP	24	RW	1'b 1	1 – Enabled 0 – Masked
Program_Star t	25	RW	1'b 1	1 – Enabled 0 – Masked
CMD40 Interrupt	26	RW	1'b 1	1 – Enabled 0 – Masked
CMD R1b Interrupt	27	RW	1'b 1	1 – Enabled 0 – Masked
FunctionX CRC / End Error Interrupt	28	RW	1'b 1	1 – Enabled 0 – Masked
FunctionX Abort Interrupt	29	RW	1'b 1	1 – Enabled 0 – Masked
LRST Interrupt	30	RW	1'b 1	1 – Enabled 0 – Masked
BOOT COMPLETE Interrupt	31	RW	1'b 1	1 – Enabled 0 – Masked

Setting 1 to Interrupt Status Enable register, enables the Interrupt Status.

11.2.15 Interrupt Signal Enable Register

Interrupt Signal Enable Register (0x44)				
Register Field	Offset	Access	Default Value	Descrioption
Transfer Complete Interrupt	0	RW	1’b 1	1 – Enabled 0 – Masked
DMA1 Interrupt	1	RW	1’b 1	1 – Enabled 0 – Masked
SLEEP / AWAKE Interrupt	2	RW	1’b 1	1 – Enabled 0 – Masked
Write Start Interrupt	3	RW	1’b 1	1 – Enabled 0 – Masked
Read Start Interrupt	4	RW	1’b 1	1 – Enabled 0 – Masked
Password Set Interrupt	5	RW	1’b 1	1 – Enabled 0 – Masked
Password Reset Interrupt	6	RW	1’b 1	1 – Enabled 0 – Masked
Lock Card Interrupt	7	RW	1’b 1	1 – Enabled 0 – Masked
Unlock Card Interrupt	8	RW	1’b 1	1 – Enabled 0 – Masked
Force Erase Interrupt	9	RW	1’b 1	1 – Enabled 0 – Masked
Erase Interrupt	10	RW	1’b 1	1 – Enabled 0 – Masked
CMD11 Interrupt	11	RW	1’b 1	1 – Enabled 0 – Masked
CMD0/CMD52 Soft Reset	12	RW	1’b 1	1 – Enabled 0 – Masked
CMD6-Check Done Interrupt	13	RW	1’b 1	1 – Enabled 0 – Masked
CMD6-Switch Done Interrupt	14	RW	1’b 1	1 – Enabled 0 – Masked
Program CSD Interrupt	15	RW	1’b 1	1 – Enabled 0 – Masked
ACMD23 Interrupt	16	RW	1’b 1	1 – Enabled 0 – Masked

CMD20 Interrupt	17	RW	1’b 1	1 – Enabled 0 – Masked
Reserved	18	Rsvd	1’b 1	Reserved for Future Use
CMD4 Interrupt	19	RW	1’b 1	1 – Enabled 0 – Masked
Boot Start Interrupt	20	RW	1’b 1	1 – Enabled 0 – Masked
Function1 Reset	21	RW	1’b 1	1 – Enabled 0 – Masked
Function2 Reset	22	RW	1’b 1	1 – Enabled 0 – Masked
CMD11_CLK _START	23	RW	1’b 1	1 – Enabled 0 – Masked
CMD11_CLK _STOP	24	RW	1’b 1	1 – Enabled 0 – Masked
Program_Star t	25	RW	1’b 1	1 – Enabled 0 – Masked
CMD40 Interrupt	26	RW	1’b 1	1 – Enabled 0 – Masked
CMD R1b Interrupt	27	RW	1’b 1	1 – Enabled 0 – Masked
FunctionX CRC / End Error Interrupt	28	RW	1’b 1	1 – Enabled 0 – Masked
FunctionX Abort Interrupt	29	RW	1’b 1	1 – Enabled 0 – Masked
LRST Interrupt	30	RW	1’b 1	1 – Enabled 0 – Masked
BOOT COMPLETE Interrupt	31	RW	1’b 1	1 – Enabled 0 – Masked

11.2.16 Interrupt Status Enable2 Register

Interrupt Status Enable2 Register (0xa0)

Register Field	Offset	Access	Default Value	Descrioption
Function3 Reset	0	RW	1’b 1	1 – Enabled 0 – Masked
Function4 Reset	1	RW	1’b 1	1 – Enabled 0 – Masked

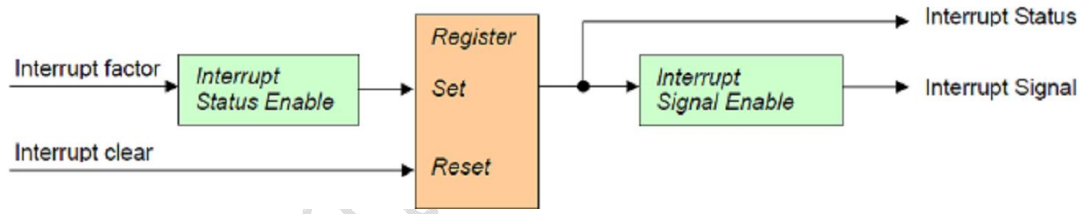
Function5 Reset	2	RW	1’b 1	1 – Enabled 0 – Masked
Reserved	29	Rsvd	1’b 0	Reserved for Future Use0 – Masked

11.2.17 Interrupt Signal2 Enable Register

Interrupt Signal Enable2 Register (0xa4)

Register Field	Offset	Access	Default Value	Descrioption
Function3 Reset	0	RW	1’b 1	1 – Enabled 0 – Masked
Function4 Reset	1	RW	1’b 1	1 – Enabled 0 – Masked
Function5 Reset	2	RW	1’b 1	1 – Enabled 0 – Masked
Reserved	29	RW	1’b 0	Reserved for Future Use

The Interrupt Signal Enable register is used to select which Interrupt status is indicated to the Processor as the Interrupt. These status bits all share the same 1 bit Interrupt line. Setting any of these bits to 1 enables Interrupt generation.



11.2.18 Interrupt Status2 Register

Interrupt Status2 Register (0x9C)

Register Field	Offset	Access	Default Value	Descrioption
Function3 Reset	0	RW1C	1’b 0	Function3 Reset Interrupt: This bit is set, if SDIO Host performs the following sequence for IOE1 The host can also use IOEx as a per function reset for error recovery. The host sequence for a per function reset is to reset IOEx to 0, wait until IORx becomes 0 and then set IOEx to 1 again. If the error is not recovered by this sequence, SDIO reset should be used noting that the operation of a

				ll functions will be aborted.
Function4 Reset	1	RW1C	1’b 0	Function4 Reset Interrupt: This bit is set, if SDIO Host performs the following sequence for IOE2 The host can also use IOEx as a per function reset for error recovery. The host sequence for a per function reset is to reset IOEx to 0, wait until IORx becomes 0 and then set IOEx to 1 again. If the error is not recovered by this sequence, SDIO reset should be used noting that the operation of all functions will be aborted.
Function5 Reset	2	RW1C	1’b 1	Function5 Reset Interrupt: This bit is set, if SDIO Host performs the following sequence for IOE2 The host can also use IOEx as a per function reset for error recovery. The host sequence for a per function reset is to reset IOEx to 0, wait until IORx becomes 0 and then set IOEx to 1 again. If the error is not recovered by this sequence, SDIO reset should be used noting that the operation of all functions will be aborted.
Reserved	31:3	RW	29’b 0	Reserved for Future Use

11.2.19 Card Address Register

Card Address Register (0x48)

Register Field	Offset	Access	Default Value	Descrioption
Starting Address	9:0	RW	10’b 0	For every write or read access, the card address will increment"
Reserved	31:10	Rsvd	14’b 0	Reserved for Future Use

11.2.20 Card Data Register

Card Data Register (0x4C)

Register Field	Offset	Access	Default	Descrioption

			Value	
Card Data Register	31:0	RW	32’b 0	This register is used for Indirect Accessing. The processor will access the Card Registers through this register. Please refer Data flow section for this register usage.

11.2.21 IOREADY Register

IOREADY Register (0x50)

Register Field	Offset	Access	Default Value	Descrioption
Reserved	0	Rsvd	1’b 0	Reserved for Future Use
Function1 Ready	1	RW	1’b 0	Function1 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Function2 Ready	2	RW	1’b 0	Function2 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Function3 Ready	3	RW	1’b 0	Function3 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Function4 Ready	4	RW	1’b 0	Function4 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Function5 Ready	5	RW	1’b 0	Function5 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Function2 Ready	2	RW	1’b 0	Function2 Ready. This bit reflects in CMD5-R4 response (Card Ready).
Reserved	7:6	R	2’b 0	Reserved for Future Use
Reserved	31:8	RW	24’b 0	Reserved for Future Use

11.2.22 Function1 Control Register

Function1 Control Register (0x54)

Register Field	Offset	Access	Default Value	Descrioption
Function1 Read Count	15:0	RW	32’b 0	Function1 Read Count: This field denotes the number of bytes to read from Function1.There are two types of Read Transactions 1. Host Initiated Read Transaction 2. Device Initiated Read Transaction This register field is used for Device Initiated Read Transaction. The PE-SMID IP will assert an Interrupt, whenever processor writes

				a nonzero value to this register. On receving the Interrupt the SD Host has to read the function1 Read Control register to find the actual number of bytes to read from Function1 Area. SD Host will access this read count field of read control register using cmd52 with address offset 0 and function number 1. Based on the read count, the SD host will initiate single or multiple cmd53 read transactions. The PE-SMID will clear this register, whenever SD Host reads the Functionx read count register. The processor should not program the new transfer count value, till this register gets cleared by PE-SMID.
Reserved	31:16	R	16’b 0	Reserved for Future Use

11.2.23 SDIO CCCR Control Register

SDIO CCCR Control Register (0x5C)

Register Field	**Offset**	**Access**	**Default Value**	**Descrioption**
CCCR Revision	3:0	RW	4’h 3	CCCR Format Version Number. Value CCCR/FBR Format Version 00h CCCR/FBR defined in SDIO Version 1.00 01h CCCR/FBR defined in SDIO Version 1.10 02h CCCR/FBR defined in SDIO Version 2.00 1.20 03h CCCR/FBR defined in SDIO Version 3.00 04h-0Fh Reserved for Future Use
SDIO Specification Revision	7:4	RW	4’h 4	SDIO Specification Revision Number Value SDIO Specification 00h SDIO Specification Version 1.00 01h SDIO Specification Version 1.10 02h SDIO Specification Version 1.20 (unreleased) 03h SDIO Specification Version 2.00 04h SDIO Specification Version 3.00

				05h-0Fh Reserved for Future Use
SDx	11:8	RW	4’h 3	SD Format Version Number. Value Physical Layer Specification 00h Physical Layer Specification Version 1.01 (March 2000) 01h Physical Layer Specification Version 1.10 (October 2004) 02h Physical Layer Specification Version 2.00 (May 2006) 03h Physical Layer Specification Version 3.0x 04h-0Fh Reserved for Future Use
S8B	12	RW	1’b 1	Support 8it bus mode. 0 – 8bit bus mode is not supported 1 – 8bit bus mode is supported
SCSI	13	RW	1’b 1	Support Continuous Support of Interrupt 1 – Continuous SPI Interrupt is supported 0 – Continuous SPI Interrupt is not supported.
SDC	14	RW	1’b 1	Support Direct Command 0 – Direct Command (CMD52) is not supported 1 – Direct Command (CMD52) is supported
SMB	15	RW	1’b 1	Support Multiple block 0 – Multiple Block Transfer is not supported 1 – Multiple Block Transfer is supported
SRW	16	RW	1’b 1	Support Read Wait 0 – Read Wait is not supported 1 – Read Wait is supported
SBS	17	RW	1’b 1	Support Bus Control 0 – Suspend/Resume is not supported 1 – Suspend/Resume is supported
S4MI	18	RW	1’b 0	Support Block Gap Interrupt 0 – Block Gap Interrupt is not supported 1 – Block Gap Interrupt is supported
LSC	19	RW	1’b 0	0 – Not a Low Speed Card

				1 – Low Speed Card.
4BLS	20	RW	1’b 1	0 – 4bit mode for low speed card is not supported 1 – 4bit mode for low speed card is supported
SMPC	21	RW	1’b 1	Support for Master Power Control
SHS	22	RW	1’b 1	0 – High Speed is not supported 1 – High Speed is supported
SDR50	23	RW	1’b 1	0 – SDR50 is not supported 1 – SDR50 is supported.
SDR104	24	RW	1’b 1	0 – SDR104 is not supported 1 – SDR104 is supported.
DDR50	25	RW	1’b 1	0 – DDR50 is not supported 1 – DDR50 is supported.
SDTA	26	RW	1’b 0	0 – Driver Type A is not supported 1 – Driver Type A is supported.
SDTC	27	RW	1’b 0	0 – Driver Type C is not supported 1 – Driver Type C is supported.
SDTD	28	RW	1’b 0	0 – Driver Type D is not supported 1 – Driver Type D is supported.
SAI	29	RW	1’b 1	0 – Asynchronous Interrupt is not supported 1 – Asynchronous Interrupt is supported.
Reserved	31:30	Rsvd	2’b 0	Reserved for Future Use.

11.2.24 SDIO FBRX Control Register

This register is optional. We can configure through defines too

0x60 – SDIO FBR1 Control Register

0x64 – SDIO FBR2 Control Register

0x7C – SDIO FBR7 Control Register

Register Field	Offset	Access	Default Value	Descrioption
Function X Standard Interface code	3:0	RW	4’h 0	SDIO Standard Function Code. 0h: No SDIO standard interface supported by this function 1h: This function supports the SDIO Standard UART 2h: This function supports the SDIO Bluetooth Type-A standard interface

				3h: This function supports the SDIO Bluetooth Type-B standard interface 4h: This function supports the SDIO GPS standard interface 5h: This function supports the SDIO Camera standard interface 6h: This function supports the SDIO PHS standard interface 7h: This function supports the SDIO WLAN interface 8h: This function supports the Embedded SDIOATA standard interface (Embedded SDIO-ATA shall be implemented only on devices following the "Embedded SDIO Specification"). 9h: This function supports the SDIO Bluetooth Type-A AMP standard interface (AMP: Alternate MAC PHY) 10h-Eh: Not assigned, reserved for future use Fh: This function supports an SDIO standard interface number greater than Eh. In this case, the value in byte 101h identifies the standard SDIO interfaces type.
Function X Extended Standard SDIO Function Interface Code.	11:4	RW	8’b 0	Function X Extended Standard SDIO Function Interface Code.
FunctionX supports CSA	15	RW	1’b 0	0 – CSA is not supported 1 – CSA is supported
15 RW 1’b 0 0 – CSA is not supported 1 – CSA is supported SPS	16	RW	1’b 0	0 – Power Selection is supported 1 – Power Selection is not supported
Reserved	31:17	Rsvd	15’b 0	Reserved for Future Use

11.2.25 Card Size Register

Card Size Register (0x80)

Register Field	Offset	Access	Default Value	Descrioption
Card Size	31:0	RW	32’h 0	Card Size Register. Unit is 512Bytes ‘h 0 – Reserved ‘h 1 – 512Bytes ‘h 2 – 1K Bytes .. ‘h 400000 – 2GB .. ‘h 8000000 – 64GB The Controller user this register value for “ADDRESS_OUT_OF_RANGE” error detection.

11.2.26 Card OCR Register

Card OCR Register (0x84)

Register Field	Offset	Access	Default Value	Descrioption
Card OCR	23:0	RW	24’h 0	Card OCR – Operational Condition Register. For Multiple OCRs, the processor has to program the logical AND of the voltage range(s) of all the IO functions (including SD memory). The SMID Controller will send the preset value of this field for ACMD41/CMD5/CMD1 query commands. 14:0 - Reserved 15 - 2.7 to 2.8 16 – 2.8 to 2.9 17 – 2.9 to 3.0 18 – 3.0 to 3.1 19 – 3.1 to 3.2 20 – 3.2 to 3.3 21 – 3.3 to 3.4 22 – 3.4 to 3.5 23 – 3.5 to 3.6.
Switching to	24	RW	1’b 1	1- Switching to 1.8V accepted

1.8V				0 – Switching to 1.8V not accepted
SDIO or Combo Card	30:25	Rsvd	7’b 0	In case of SDIO or Combo card [30:25] 26:25 Reserverd for Future Use in SDIO or Combo 27th Used only in SDIO or Combo (SDIO + SDMEM) mode Bit 27 -1 => Memory present Bit 27 -0 => Memory not present 30:28 Number of I/O Functions
Reserved	28:25	Rsvd	4’b 0	Reserved for Future use
Access Mode	29	RW	1’b 0	Used only in MMC mode – tied to zero always Based on bit30 only the modes are differentiated (byte mode Vs sector mode) Bit30 – 0 => byte mode Bit30 – 1 => Sector mode
Access Mode / CCS	30	RW	1’b 0	MMC Mode – Access mode SD Mode - Card Capacity Status
Reserved	31	Rsvd	1’b 0	Reserved for Future use This bit is not used, instead bit2 of control register (card_init_done) is used to drive bit31 of OCR register.

11.2.27 Control2 Register

Control2 Register (0x88)

Register Field	Offset	Access	Default Value	Descrioption
CMD60_busy_en_dis	0	RW	1’b 0	User Defined Command 0 – R1 Response The Firmware when set this bit to Zero will receive the CMD R1b interrupt “bit 27”. The Firmware should not set program done bit in Control register. 1 - R1b Response The Firmware should set Wr_last_blk_busy “bit1” in Control register to 1 when this bit is set. to 1. SMID will drive busy in Data0 line and asserts CMD R1b “bit 27” interrupt to Firmware. Busy will get deasserted only when Program done bit0 in is set to 1 in Control register.
rd_busy_en	1	RW	1’b 0	0 - Assert Busy after cmd12 for a read

_dis				operation 1 - Do not assert busy after cmd12 for a read operation
Reserved	31:2	Rsvd	30’b0	Reserved for Future use

11.2.28 Custom Design Registers

偏移地址	位宽	读写属性	寄存器名字
0x100	8	WR	CARD_REVISION_REG
0x104	16	WR	CARD_FW_STATUS0_REG/CARD_FW_STATUS1_REG
0x108	8	WR	CARD_STATUS_REG
0x10C	16	WR	HOST_F1_RD_BASE_0/ HOST_F1_RD_BASE_1
0x110	8	WR	WR_BITMAP
0x114	8	RO	HOST_PWR_CTRL
0x118	16	WR	RD_LEN_P1
0x11C	16	WR	RD_LEN_P2
0x120	16	WR	RD_LEN_P3
0x124~	/	/	Reserved

其中所有的属性为WR特点的寄存器在hardware层次上没有特别，仅仅为firmware向host进行信息的传递，这些寄存器传递到host可以访问的对应地址时，仅仅为只读。

其次HOST_PWR_CTRL寄存器为HOST可以读写，但firmware仅仅为只读，其内容如下：

Bit	说明
0	Sdio_power_off 当电源管理处于NON_ASSO_SLEEP状态时，本bit从0变1，将触发进入WLAN_OFF状态。注：为1的时间足够为32768Hz时钟能够检测到。
1	Sdio_power_on 当电源管理处于NON_ASSO_SLEEP时，本bit从0变1，将触发进入WLAN_OFF状态。 当电源管理处于ASSO_SLEEP/NON_ASSO_SLEEP/WLAN_OFF，本bit从0变1，将触发进入wakeup状态。
7:2	Unused bits

12. PMU寄存器(0x4010_0000)

寄存器偏移地址	寄存器名字	属性	位宽	说明	默认值
30	SPI_PIN_MUX_CTRL	WR	1	Bit 0, 为0, QSPI管脚（QSPI_SCK, QSPI_CS, QSPI_DI, QSPI_DO）就是QSPI模块使用； 为1, QSPI管脚（QSPI_SCK, QSPI_CS, QSPI_DI, QSPI_DO）就是SPI（sysnopsy）模块使用；	0
34	GPIO_PIN_MUX_CTRL	WR	31:0	Bit [31:3]： 29个GPIO的管脚复用的GPIO使能信号。bit为1控制相应的管脚为GPIO。 Bit[2]: PAON 使能信号。0：GPIO2;1：PAON。 Bit[1]: I2S 和JTAG的MUX 选择。0：I2S接口；1：JTAG接口。 Bit[0]：SDIO mux 成UART和蓝牙共存接口。0：SDIO 接口；1：UART和蓝牙共存接口。	32'h0004

13. SPI寄存器(0x4000_0000)

13.1 SPI寄存器MAP

Name	Address Offset	Width	Description	Reset Value
CTRLR0	0x0	16	*Control Register 0*	*0x7*
CTRLR1	0x04	16	*Control Register 1*	0x0
SSIENR	0x08	1	*SSI Enable Register*	0x0
MWCR	0x0c	3	*Microwire Control Register*	0x0
SER	0x10	*2*	*Slave Enable Register*	0x0
BAUDR	0x14	16	*Baud Rate Select*	0x0
TXFTLR	0x18	3	*Transmit FIFO Threshold Level*	0x0
RXFTLR	0x1C	3	*Receive FIFO Threshold Level*	0x0
TXFLR	0x20	4	*Transmit FIFO Level Register*	0x0
RXFLR	0x24	4	*Receive FIFO Level Register*	0x0
SR	0x28	7	Status Register	0x6
IMR	0x2C	*6*	*Interrupt Mask Register*	*0x3F*
ISR	0x30	6	*Interrupt Status Register*	0x0
RISR	0x34	6	*Raw Interrupt Status Register*	0x0
TXOICR	0x38	1	*Transmit FIFO Overflow Interrupt Clear Register*	0x0
RXOICR	0x3c	1	*Receive FIFO Overflow Interrupt Clear Register*	0x0

RXUICR	0x40	1	*Receive FIFO Underflow Interrupt Clear Register*	0x0
MSTICR	0x44	1	*Multi-Master Interrupt Clear Register*	0x0
ICR	0x48	1	*Interrupt Clear Register*	0x0
DMACR	0x4C	2	*DMA Control Register*	0x0
DMATDLR	0x50	3	*DMA Transmit Data Level*	0x0
DMARDLR	0x54	3	*DMA Receive Data Level*	0x0
IDR	0x58	32	Identification Register	0xffffffff
SSI_COMP_VERSION	0x5C	32	*coreKit version ID register*	0x3332312a
DR	0x60~0xec	16	Data Register	0x0
RX_SAMPLE_DLY	0xf0	8	*RXD Sample Delay Register*	0x0
RSVD_0	0xf4			
RSVD_1	0xf8			
RSVD_2	0xfc			

13.2SPI寄存器说明

13.2.1 CTRLR0

Bits	Name	R/W	Description
15:12	*CFS*	*R/W*	*Control Frame Size. Selects the length of the control word for the Microwire frame format.*
11	*SRL*	*R/W*	*Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. Can be used in both serialslave and serial-master modes.* *0 – Normal Mode Operation* *1 – Test Mode Operation*
10	Reserved		
9:8	*TMOD*	*R/W*	*Transfer Mode. Selects the mode of transfer for serial communication. This field does not affect the transfer duplicity. Only indicates whether the receive or transmit data are valid. In transmit-only mode, data received*

			from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both transmit and receive data are valid. The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be accessed by the host processor.In eeprom-read mode, receive data is not valid while control data is being transmitted. When all control data is sent to the EEPROM, receive data becomes valid and transmit data becomes invalid. All data in the transmit FIFO is considered control data in this mode. This transfer mode is only valid when the DW_apb_ssi is configured as a master device. *00 — Transmit & Receive* *01 — Transmit Only* *10 — Receive Only* *11 — EEPROM Read*
7	*SCPOL*	*R/W*	*Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. Used to select the polarity of the inactive serial clock, which is held inactive when the DW_apb_ssi master is not actively transferring data on the serial bus.* *0 – Inactive state of serial clock is low* *1 – Inactive state of serial clock is high* **Reset Value:** *0*
6	*SCPH*	*R/W*	*Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock.* *0: Serial clock toggles in middle of first data bit* *1: Serial clock toggles at start of first data bit* **Reset Value:** *0*
5:4	*FRF*	*R/W*	*Frame Format. Selects which serial protocol transfers the data.* *00 — Motorola SPI* *01 — Texas Instruments SSP* *10 — National Semiconductors Microwire* *11 — Reserved* **Reset Value:** *0x0*
3:0	*DFS*	*R/W*	*Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by*

			the receive logic, with the upper bits of the receive FIFO zero-padded. You must right-justify transmit data before writing into the transmit FIFO. The transmit *logic ignores the upper unused bits when transmitting the data.* **Reset Value:** *0x7*

DFS Decode

DFS Value	**Description**
0000	*Reserved – undefined operation*
0001	*Reserved – undefined operation*
0010	*Reserved – undefined operation*
0011	*4-bit serial data transfer*
0100	*5-bit serial data transfer*
0101	*6-bit serial data transfer*
0110	*7-bit serial data transfer*
0111	*8-bit serial data transfer*
1000	*9-bit serial data transfer*
1001	*10-bit serial data transfer*
1010	*11-bit serial data transfer*
1011	*12-bit serial data transfer*
*110*0	*13-bit serial data transfer*
1101	*14-bit serial data transfer*
1110	*15-bit serial data transfer*
1111	*16-bit serial data transfer*

CFS Decode

CFSValue	**Description**
0000	*1-bit control word*
0001	*2-bit control word*
0010	3*-bit control word*
0011	4*-bit control word*
0100	5*-bit control word*
0101	6*-bit control word*
0110	7*-bit control word*
0111	8*-bit control word*
1000	9*-bit control word*
1001	10*-bit control word*
1010	*11-bit control word*
1011	12*-bit control word*
*110*0	13*-bit control word*
1101	14*-bit control word*
1110	15*-bit control word*
1111	16*-bit control word*

13.2.2 CTRLR1

Bits	Name	R/W	Description
15:0	*NDF*	*R/W*	*Number of Data Frames. When TMOD = 10 or TMOD = 11, this register field sets the number of data frames to be continuously received by the DW_apb_ssi. The DW_apb_ssi continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer.* **Reset Value:** *0x0*

13.2.3 SSIENR

Bits	Name	R/W	Description
0	*SSI_EN*	*R/W*	*SSI Enable. Enables and disables all DW_apb_ssi operations. When disabled, all serial transfers are halted immediately. Transmit and receive FIFO buffers are cleared when the device is disabled. It is impossible to program some of the DW_apb_ssi control registers when enabled. When disabled, the ssi_sleep output is set (after delay) to inform the system that it is safe to remove the ssi_clk, thus saving power consumption in the system.* **Reset Value:** *0x0*

13.2.4 MWCR

Bits	Name	R/W	Description
2	*MHS*	*R*	*Microwire Handshaking. Relevant only when the DW_apb_ssi is configured as a serial-master device. Used to enable and disable the "busy/ready" handshaking interface for the Microwire protocol. When enabled, the DW_apb_ssi checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the SR register.* *0: handshaking interface is disabled* *1: handshaking interface is enabled* **Reset Value:** *0x0*
1	*MDD*	*R*	*Microwire Control. Defines the direction of the data word when the Microwire serial protocol is used. When this bit is set to 0, the data word is received by the DW_apb_ssi MacroCell from the external serial device. When this bit is set to 1, the data word is transmitted from the DW_apb_ssi MacroCell to the external serial device.* **Reset Value:** *0x0*
0	*MWMOD*	*R*	*Microwire Transfer Mode. Defines whether the Microwire transfer is*

			sequential or *non-sequential. When sequential mode is used, only one control word is needed to* *transmit or receive a block of data words. When non-sequential mode is used, there* *must be a control word for each data word that is transmitted or received.* *0 – non-sequential transfer* *1 – sequential transfer* **Reset Value:** *0x0*

13.2.5 SER

Bits	Name	R/W	Description
31:SSI_NUM_SLAVES	*Reserved*	*N/A*	*Reserved*
SSI_NUM_SLAVES-1:0	*SER*	*R/W*	*Slave Select Enable Flag. Each bit in this register corresponds to a slave select line (ss_**X**_n]) from the DW_apb_ssi master. When a bit in this register is set (1), the corresponding slave select line from the master is activated when a serial transfer begins. It should be noted that setting or clearing bits in this register have no effect on the corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable the bit in this register that corresponds to the slave device with which the master wants to communicate.* *When not operating in broadcast mode, only one bit in this field should be set.* *1: Selected* *0: Not Selected* **Reset Value:** *0x0*

13.2.6 BAUDR

Bits	Name	R/W	Description
15:0	*SCKDV*	*R/W*	*SSI Clock Divider. The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation:* `Fsclk_out = Fssi_clk/SCKDV` `Where Fssi_clk` *= apb_clk, SCKDV is any even value between 2 and 65534. For example:* `for Fssi_clk = 3.6864MHz and SCKDV =2` `Fsclk_out = 3.6864/2 = 1.8432MHz` **Reset Value:** *0x0*

13.2.7 TXFTLR

Bits	Name	R/W	Description
31:TX_ABW	*Reserved*	*N/A*	*Reserved*
TX_ABW-1:0	*TFT*	*R/W*	*Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is configurable in the range 2-256; this register is sized to the number of address bits needed to access the FIFO.* *If you attempt to set bits [7:0] of this register to a value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered. For field decode, refer to Table below* **Reset Value:** *0x0*

TFT Decode

TFT Value	Description
0000_0000	*ssi_txe_intr is asserted when 0 data entries are present in transmit FIFO*
0000_0001	*ssi_txe_intr is asserted when 1 or less data entry is present in transmit FIFO*
0000_0010	*ssi_txe_intr is asserted when 2 or less data entries are present in transmit FIFO*
0000_0011	*ssi_txe_intr is asserted when 3 or less data entries are present in transmit FIFO*
……	……
1111_1100	*ssi_txe_intr is asserted when 252 or less data entries are present in transmit FIFO*
1111_1101	*ssi_txe_intr is asserted when 253 or less data entries are present in transmit FIFO*
1111_1110	*ssi_txe_intr is asserted when 254 or less data entries are present in transmit FIFO*
1111_1111	*ssi_txe_intr is asserted when 255 or less data entries are present in transmit FIFO*

13.2.8 RXFTLR

Bits	Name	R/W	Description
31:RX_ABW	*Reserved*	*N/A*	*Reserved*
RX_ABW-1:0	*RFT*	*R/W*	*Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is*

			configurable in the range 2-256. This register is sized to the number of address bits needed to access the FIFO. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered. For field decode, refer to Table below. **Reset Value:** *0x0*

TFT Decode

TFT Value	Description
0000_0000	*ssi_rxf_intr is asserted when 1 or more data entry is present in receive FIFO*
0000_0001	*ssi_rxf_intr is asserted when 2 or more data entries are present in receive FIFO*
0000_0010	*ssi_rxf_intr is asserted when 3 or more data entries are present in receive FIFO*
0000_0011	*ssi_rxf_intr is asserted when 4 or more data entries are present in receive FIFO*
……	……
1111_1100	*ssi_rxf_intr is asserted when 253 or more data entries are present in receive FIFO*
1111_1101	*ssi_rxf_intr is asserted when 254 or more data entries are present in receive FIFO*
1111_1110	*ssi_rxf_intr is asserted when 255 or more data entries are present in receive FIFO*
1111_1111	*ssi_rxf_intr is asserted when 256 data entries are present in receive FIFO*

13.2.9 TXFLR

Bits	Name	R/W	Description
31:TX_ABW+1	*Reserved*	*N/A*	*Reserved*
TX_ABW:0	*TXTFL*	*R*	*Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.* **Reset Value:** *0x0*

13.2.10 RXFLR

Bits	Name	R/W	Description
31:RX_ABW+1	*Reserved*	*N/A*	*Reserved*

RX_ABW:0	*RXTFL*	*R*	*Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.* **Reset Value:** *0x0*

13.2.11 SR

Bits	Name	R/W	Description
6	*DCOL*	*R*	*Data Collision Error. Relevant only when the DW_apb_ssi is configured as a master device. This bit is set if the DW_apb_ssi master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was halted before completion. This bit is cleared when read.* *0 – No error* *1 – Transmit data collision error* **Reset Value:** *0x0*
5	*TXE*	*R*	*Transmission Error. Set if the transmit FIFO is empty when a transfer is started. This bit can be set only when the DW_apb_ssi is configured as a slave device. Data from the previous transmission is resent on the txd line. This bit is cleared when read.* *0 – No error* *1 – Transmission error* **Reset Value:** *0x0*
4	*RFF*	*R*	*Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared.* *0 – Receive FIFO is not full* *1 – Receive FIFO is full* **Reset Value:** *0x0*
3	*RFNE*	*R*	*Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO.* *0 – Receive FIFO is empty* *1 – Receive FIFO is not empty* **Reset Value:** *0x0*
2	*TFE*	*R*	*Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt.* *0 – Transmit FIFO is not empty* *1 – Transmit FIFO is empty* **Reset Value:** *0x1*
1	*TFNF*	*R*	*Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full.* *0 – Transmit FIFO is full*

			1 – Transmit FIFO is not full **Reset Value:** *0x1*
0	*BUSY*	*R*	*SSI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the DW_apb_ssi is idle or disabled.* *0 – DW_apb_ssi is idle or disabled* *1 – DW_apb_ssi is actively transferring data* **Reset Value:** *0x0*

13.2.12 IMR

Bits	Name	R/W	Description
5	*MSTIM*	*R/W*	*Multi-Master Contention Interrupt Mask. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.* *0 – ssi_mst_intr interrupt is masked* *1 – ssi_mst_intr interrupt is not masked* **Reset Value:** *0x1*
4	*RXFIM*	*R/W*	*Receive FIFO Full Interrupt Mask* *0 – ssi_rxf_intr interrupt is masked* *1 – ssi_rxf_intr interrupt is not masked* **Reset Value:** *0x1*
3	*RXOIM*	*R/W*	*Receive FIFO Overflow Interrupt Mask* *0 – ssi_rxo_intr interrupt is masked* *1 – ssi_rxo_intr interrupt is not masked* **Reset Value:** *0x1*
2	*RXUIM*	*R/W*	*Receive FIFO Underflow Interrupt Mask* *0 – ssi_rxu_intr interrupt is masked* *1 – ssi_rxu_intr interrupt is not masked* **Reset Value:** *0x1*
1	*TXOIM*	*R/W*	*Transmit FIFO Overflow Interrupt Mask* *0 – ssi_txo_intr interrupt is masked* *1 – ssi_txo_intr interrupt is not masked* **Reset Value:** *0x1*
0	*TXEIM*	*R/W*	*Transmit FIFO Empty Interrupt Mask* *0 – ssi_txe_intr interrupt is masked* *1 – ssi_txe_intr interrupt is not masked* **Reset Value:** *0x1*

13.2.13 ISR

Bits	Name	R/W	Description
5	*MSTIM*	*R*	*Multi-Master Contention Interrupt Status. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.*

			0 = ssi_mst_intr interrupt not active after masking *1 = ssi_mst_intr interrupt is active after masking* **Reset Value:** *0x0*
4	*RXFIS*	*R*	*Receive FIFO Full Interrupt Status* *0 = ssi_rxf_intr interrupt is not active after masking* *1 = ssi_rxf_intr interrupt is full after masking* **Reset Value:** *0x0*
3	*RXOIS*	*R*	*Receive FIFO Overflow Interrupt Status* *0 = ssi_rxo_intr interrupt is not active after masking* *1 = ssi_rxo_intr interrupt is active after masking* **Reset Value:** *0x0*
2	*RXUIS*	*R*	*Receive FIFO Underflow Interrupt Status* *0 = ssi_rxu_intr interrupt is not active after masking* *1 = ssi_rxu_intr interrupt is active after masking* **Reset Value:** *0x0*
1	*TXOIS*	*R*	*Transmit FIFO Overflow Interrupt Status* *0 = ssi_txo_intr interrupt is not active after masking* *1 = ssi_txo_intr interrupt is active after masking* **Reset Value:** *0x0*
0	*TXEIS*	*R*	*Transmit FIFO Empty Interrupt Status* *0 = ssi_txe_intr interrupt is not active after masking* *1 = ssi_txe_intr interrupt is active after masking* **Reset Value:** *0x0*

13.2.14 RISR

Bits	Name	R/W	Description
5	*MSTIR*	*R*	*Multi-Master Contention Raw Interrupt Status. This bit field is not present if the DW_apb_ssi is configured as a serial-slave device.* *0 = ssi_mst_intr interrupt is not active prior to masking* *1 = ssi_mst_intr interrupt is active prior masking* **Reset Value:** *0x0*
4	*RXFIR*	*R*	*Receive FIFO Full Raw Interrupt Status* *0 = ssi_rxf_intr interrupt is not active prior to masking* *1 = ssi_rxf_intr interrupt is active prior to masking* **Reset Value:** *0x0*
3	*RXOIR*	*R*	*Receive FIFO Overflow Raw Interrupt Status* *0 = ssi_rxo_intr interrupt is not active prior to masking* *1 = ssi_rxo_intr interrupt is active prior masking* **Reset Value:** *0x0*
2	*RXUIR*	*R*	*Receive FIFO Underflow Raw Interrupt Status* *0 = ssi_rxu_intr interrupt is not active prior to masking* *1 = ssi_rxu_intr interrupt is active prior to masking*

			Reset Value: *0x0*
1	*TXOIR*	*R*	*Transmit FIFO Overflow Raw Interrupt Status* *0 = ssi_txo_intr interrupt is not active prior to masking* *1 = ssi_txo_intr interrupt is active prior masking* **Reset Value:** *0x0*
0	*TXEIR*	*R*	*Transmit FIFO Empty Raw Interrupt Status* *0 = ssi_txe_intr interrupt is not active prior to masking* *1 = ssi_txe_intr interrupt is active prior masking* **Reset Value:** *0x0*

13.2.15 TXOICR

Bits	Name	R/W	Description
0	*TXOICR*	*R*	*Clear Transmit FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_txo_intr interrupt; writing has no effect.* **Reset Value:** *0x0*

13.2.16 RXOICR

Bits	Name	R/W	Description
0	*RXOICR*	*R*	*Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt.A read from this register clears the ssi_rxo_intr interrupt; writing has no effect.* **Reset Value:** *0x0*

13.2.17 RXUICR

Bits	Name	R/W	Description
0	*RXUICR*	*R*	*Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_rxu_intr interrupt; writing has no effect.* **Reset Value:** *0x0*

13.2.18 MSTICR

Bits	Name	R/W	Description
0	*MSTICR*	*R*	*Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the ssi_mst_intr interrupt; writing has no effect.* **Reset Value:** *0x0*

13.2.19 ICR

Bits	Name	R/W	Description
0	*ICR*	*R*	*Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the ssi_txo_intr, ssi_rxu_intr, ssi_rxo_intr, and the ssi_mst_intr interrupts. Writing to this register has no effect.* **Reset Value:** *0x0*

13.2.20 DMACR

Bits	Name	R/W	Description
1	*TDMAE*	*R/W*	*Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel.* *0 = Transmit DMA disabled* *1 = Transmit DMA enabled* **Reset Value:** *0x0*
0	*RDMAE*	*R/W*	*Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel* *0 = Receive DMA disabled* *1 = Receive DMA enabled* **Reset Value:** *0x0*

13.2.21 DMATDLR

Bits	Name	R/W	Description
TX_ABW-1:0	*DMATDL*	*R/W*	*Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1. Refer to Table below for the field decode.* **Reset Value:** *0x0*

Table 6-6 DMATDL Decode

DMATDL Value	Description
0000_0000	*dma_tx_req is asserted when 0 data entries are present in the transmit FIFO*
0000_0001	*dma_tx_req is asserted when 1 or less data entry is present in the transmit FIFO*
0000_0010	*dma_tx_req is asserted when 2 or less data entries are present in the transmit FIFO*
0000_0011	*dma_tx_req is asserted when 3 or less data entries are pres*

	ent in the transmit FIFO
……	……
1111_1100	*dma_tx_req is asserted when 252 or less data entries are p resent in the transmit FIFO*
1111_1101	*dma_tx_req is asserted when 253 or less data entries are p resent in the transmit FIFO*
1111_1110	*dma_tx_req is asserted when 254 or less data entries are p resent in the transmit FIFO*
1111_1111	*dma_tx_req is asserted when 255 or less data entries are p resent in the transmit FIFO*

13.2.22 DMARDLR

Bits	Name	R/W	Description
RX_ABW-1:0	*DMARDL*	*R/W*	*Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1. Refer to Table below for the field decode.* **Reset Value:** *0x0*

14 **Table 6-7 DMARDLDecode**

DMARDL Value	Description
0000_0000	*dma_rx_req is asserted when 1 or more data entries are pr esent in the receive FIFO*
0000_0001	*dma_rx_req is asserted when 2 or more data entries are pr esent in the receive FIFO*
0000_0010	*dma_rx_req is asserted when 3 or more data entries are pr esent in the receive FIFO*
0000_0011	*dma_rx_req is asserted when 4 or more valid data entries a re present in the receive FIFO*
……	……
1111_1100	*dma_rx_req is asserted when 253 or more data entries are present in the receive FIFO*
1111_1101	*dma_rx_req is asserted when 254 or more data entries are present in the receive FIFO*
1111_1110	*dma_rx_req is asserted when 255 or more data entries are present in the receive FIFO*
1111_1111	*dma_rx_req is asserted when 256 data entries are present in the receive FIFO*

13.2.23 IDR

Bits	Name	R/W	Description
31:0	*IDCODE*	*R*	*Identification Code. This register contains the peripherals identification code, which is written into the register at configuration time using coreConsultant.* **Reset Value:** *N/A*

13.2.24 SSI_COMP_VERSION

Bits	Name	R/W	Description
31:0	*SSI_COMP_ VERSION*	*R*	*Contains the hex representation of the Synopsys component version. Consists of ASCII value for each number in the version, followed by *. For example 32_30_31_2A represents the version 2.01*.* **Reset Value:** *See the releases table in the AMBA 2 release notes*

13.2.25 DR

Bits	Name	R/W	Description
15:0	*DR*	*R/W*	*Data Register. When writing to this register, you must right-justify the data. Read* *data are automatically right-justified.* *Read = Receive FIFO buffer* *Write = Transmit FIFO buffer* **Reset Value:** *0x0*

13.2.26 RX_SAMPLE_DLY

Bits	Name	R/W	Description
7:0	*RSD*	*R/W*	*Receive Data (rxd) Sample Delay. This register is used to delay the sample of the rxd input signal. Each value represents a single ssi_clk delay on the sample of the rxd signal.* *NOTE: If this register is programmed with a value that exceeds the depth of the internal shift registers (4), a zero (0) delay will be applied to the rxd sample.* **Reset Value:** *0x0*

14. Timers寄存器(0x4000_1000)

Register Name	Addr offset	Width	R / W	default	Description
Timer0LoadCount	0x00	32	R / W	*0*	*Value to be loaded into Timer0. This is the value from which counting commences. Any value written to this register is loaded into the associated timer*
Timer0CurrentValue	0x04	32	R	*0*	*Current Value of Timer0.*
Timer0ControlReg	0x08	4	R / W	*0*	
	[2]	*Timer Interrupt Mask*			*Timer interrupt mask for Timer0* *0 – not masked* *1 – masked*
	[1]	*Timer Mode*			*Timer mode for Timer0* *0 – free-running mode* *1 – user-defined count mode*
	[0]	*Timer Enable*			*Timer enable bit for Timer0* *0 – disable* *1 – enable*
Timer0EOI	0x0C	1	R	*0*	*Reading from this register returns all zeroes (0) and clears the interrupt from Timer0*
Timer0IntStatus	0x10	1	R	*0*	*Contains the interrupt status for Timer0*
Timer1LoadCount	0x14	32	R / W	*0*	*Value to be loaded into Timer1. This is the value from which counting commences. Any value written to this register is loaded into the associated timer*
Timer1CurrentValue	0x18	32	R	*0*	*Current Value of Timer1*
Timer1ControlReg	0x1c	4	R / W		*Reading from this register returns all zeroes (0) and clears the interrupt from Timer0*
	[2]	*Timer Interrupt Mask*			*Timer interrupt mask for Timer1* *0 – not masked* *1 – masked*
	[1]	*Timer Mode*			*Timer mode for Timer1* *0 – free-running mode* *1 – user-defined count mode*

	[0]	*Timer Enable*			*Timer enable bit for Timer1* *0 – disable* *1 – enable*
Timer1EOI	0x20	1	R		*Reading from this register returns all zeroes (0) and clears the interrupt from Timer1*
Timer1IntStatus	0x24	1	R		*Contains the interrupt status for Timer1*
TimersIntStatus	0xa0	2	R		*Contains the interrupt status of all timers in the component. If a bit of this register is 0, then the corresponding timer interrupt is not active –and the corresponding interrupt could be on either the timer_intr bus or the timer_intr_n bus, depending on the interrupt polarity you have chosen. Similarly, if a bit of this register is 1, then the corresponding interrupt bit has been set in the relevant interrupt bus. In both cases, the status reported is the status after the interrupt mask has been applied. Reading from this register does not clear any active interrupts:* *0 – either timer_intr or timer_intr_n is not active after masking* *1 – either timer_intr or timer_intr_n is active after masking*
TimersEOI	0xa4	2	R		*Reading this register returns all zeroes (0) and clears all active interrupts.*
TimersRawIntStatus	*0xa8*	2			*The register contains the unmasked interrupt status of all timers in the component.* *0 – either timer_intr or timer_intr_n is not active prior to masking* *1 – either timer_intr or timer_intr_n is active prior to masking*
TIMERS_COMP_VERSION	*0xac*	32	*R*		*Current revision number of the DW_apb_timers component.* *Reset Value:*

15. Watch Dog寄存器(0x4000_4000)

Register Name	Addr offset	Width	R/W	Default	Description
WDT_CR	0x00	5	RW	0x0	
	[4:2]	*RPL*			*This is used to select the number of pclk cycles for which the system reset stays asserted. The range of values available is 2 to 256 pclk cycles.* *000 – 2 pclk cycles* *001 – 4 pclk cycles* *010 – 8 pclk cycles* *011 – 16 pclk cycles* *100 – 32 pclk cycles* *101 – 64 pclk cycles* *110 – 128 pclk cycles* *111 – 256 pclk cycles*
	[1]	*RMOD*			*Response mode.* *Selects the output response generated to a timeout.* *0 = Generate a system reset.* *1 = First generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset*
	[0]	*WDT_EN*			*WDT enable.* *This bit is used to enable and disable the DW_apb_wdt. When disabled, the counter does not decrement. Thus, no interrupts or system resets are generated. Once this bit has been enabled, it can be cleared only by a system reset.* *0 = WDT disabled.* *1 = WDT enabled.*
WDT_TORR	0X04	8	RW	0	
	[7:4]	*TOP_INIT*			*Timeout period for initialization. Used to select the timeout period that the watchdog counter restarts from for the first counter restart (kick). This register should be written after reset and before the WDT is enabled. A change of the TOP_INIT is seen only once the WDT has been enabled, and any c*

					hange after the first kick is not seen as subsequen t kicks use the period specified by the TOP bits. The range of values is limited by the WDT_CNT_ WIDTH. If TOP_INIT is programmed to select a r ange that is greater than the counter width, the ti meout period is truncated to fit to the counter wid th. This affects only the non-user specified values as users are limited to these boundaries during co nfiguration. *The range of values available for a 32-bit watchdo g counter are:* *Where i = TOP_INIT and* *t = timeout period* *For i = 0 to 15* *if WDT_USE_FIX_TOP==1* *t = 2(16 + i)* *else* *t = WDT_USER_TOP_INIT_(i)*
	[3:0]	*TOP*			*Timeout period.* *This field is used to select the timeout period from which the watchdog counter restarts. A change of the timeout period takes effect only after the next counter restart (kick). The range of values is limi ted by the WDT_CNT_WIDTH. If TOP is program med to select a range that is greater than the cou nter width, the timeout period is truncated to fit to the counter width. This affects only the non-user specified values as users are limited to these boun daries during configuration.* *The range of values available for a 32-bit watchdo g counter are:* *Where i = TOP and* *t = timeout period* *For i = 0 to 15* *if WDT_USE_FIX_TOP==1* *t = 2(16 + i)* *else* *t = WDT_USER_TOP_(i)*
WDT_CCVR	0x08	32	R	0xffff	*This register, when read, is the current value of t he internal counter. This value is read coherently when ever it is read, which is relevant when the 32 is less than the counter width.*
WDT_CRR	0x0c`	8	W	0x0	*This register is used to restart the WDT counter. As a safety feature to prevent accidental restarts, t*

					he value 0x76 must be written. A restart also clea rs the WDT interrupt. Reading this register returns zero.
WDT_STAT	0x10	1	R	0	*This register shows the interrupt status of the WD T.* *1 = Interrupt is active regardless of polarity.* *0 = Interrupt is inactive.*
WDT_EOI	0x14	1	R	0	*Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter.*
WDT_COMP_ PARAMS_5	0xe4	32	R		*Test only*
WDT_COMP_ PARAMS_4	0xe8	32	R		*Test only*
WDT_COMP_ PARAMS_3	0xec	32	R		*Test only*
WDT_COMP_ PARAMS_2	0xf0	32	R	0xffff	*Test only*
WDT_COMP_ PARAMS_1	0xf4	32	R		*Test only*
WDT_COMP_ VERSION	0xf8	32	R	0x3130372a	*ASCII value for each number in the version, follow ed by *. For example* *32_30_31_2A represents the version 2.01**
WDT_COMP_ TYPE	0xfc	32	R	0x44570120	*Designware Component Type number = 0x44_57_01 _20. This assigned unique hex value is constant, a nd is derived from the two ASCII letters "DW" fo llowed by a 16-bit unsigned number.*

16. I2C寄存器(0x4000_5000)

Register Name	Addr offset	Width	R/W	default	Description
IC_CON	0x00	7	RW	0x7d	
	[6]	*IC_SLAVE_DISABLE*			*This bit controls whether I2C has its slave disabled You have the choice of having the slave enabled or disabled after reset is ap*

			plied, which means software does not have to configure the slave. By default, the slave is always enabled (in reset state as well). If you need to disable it after reset, set t his bit to 1. If this bit is set (slave is dis abled), i2c functions only as a master and does not perform any action that requires a slave. *0: slave is enabled* *1: slave is disabled*
	[5]	*IC_RESTART_EN*	*Determines whether RESTART conditions ma y be sent when acting as a master. Some older slaves do not support handling RESTA RT conditions; however, RESTART conditions are used in several i2c operations.* *0: disable* *1: enable* *When the RESTART is disabled, the i2c ma ster is incapable of performing the following functions:* *• Sending a START BYTE* *• Performing any high-speed mode operation* *• Performing direction changes in combined format mode* *• Performing a read operation with a 10-bit address* *By replacing RESTART condition followed b y a STOP and a subsequent START conditi on, split operations are broken down into m ultiple i2c transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the IC_RAW_INT R_STAT register.*
	[4]	*IC_10BITADDR_MASTER*	*This bit controls whether the i2c starts its t ransfers in 7- or 10-bit addressing mode w hen acting as a master.* *0: 7-bit addressing* *1: 10-bit addressing*
	[3]	*IC_10BITADDR_SLAVE*	*When acting as a slave, this bit controls w hether the i2c responds to 7- or 10-bit add resses.* *0: 7-bit addressing. The i2c ignores transact ions that involve 10-bit addressing; for 7-bit addressing, only the lower 7 bits of the I*

			C_SAR register are compared. *1: 10-bit addressing. The i2c responds to only 10-bit addressing transfers that match the full 10 bits of the IC_SAR register.*
	[2:1]	*SPEED*	*These bits control at which speed the i2c operates; its setting is relevant only if one is operating the i2c in master mode. Hardware protects against illegal values being programmed by software. This register should be programmed only with a value in the range of 1 to 2; otherwise, hardware updates this register with the value of 2.* *1: standard mode (0 to 100 kbit/s)* *2: fast mode (≤ 400 kbit/s)* *3: high speed mode (≤ 3.4 Mbit/s)*
	[0]	*MASTER_MODE*	*This bit controls whether the i2c master is enabled.* *0: master disabled* *1: master enabled* **NOTE**: *Software should ensure that if this bit is written with '1,' then bit 6 should also be written with a '1'.*
IC_TAR	0x04	12 *RW* *0x1055*	
	[11]	*SPECIAL*	*This bit indicates whether software performs a General Call or START BYTE command.* *0: ignore bit 10 GC_OR_START and use IC_TAR normally* *1: perform special I2C command as specified in GC_OR_START bit* **Reset value:** *0x0*
	[10]	*GC_OR_START*	*If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the i2c.* *0: General Call Address – after issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the IC_RAW_INTR_STAT register. The i2c remains in General Call mode until the SPECIAL bit value (bit 11) is cleared.* *1: START BYTE* **Reset value:** *0x0*
	[9:0]	*IC_TAR*	*This is the target address for any master tr*

					ansaction. When transmitting a General Cal l, these bits are ignored. To generate a ST ART BYTE, the CPU needs to write only o nce into these bits. **Reset value: 0x055** *If the IC_TAR and IC_SAR are the sa me, loopback exists but the FIFOs are shar ed between master and slave, so full loopba ck is not feasible. Only one direction loopb ack mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.*
IC_SAR	0x08	10	*RW*	*0x055*	*The IC_SAR holds the slave address when t he I2C is operating as a slave. For 7-bit a ddressing, only IC_SAR[6:0] is used. This r egister can be written only when the I2C in terface is disabled, which corresponds to the IC_ENABLE register being set to 0. W rites at other times have no effect.* **Note** *The default values cannot be any of the reserved address locations: that is, 0x00 to 0x07, or 0x78 to 0x7f. The correct operation of the device is not guaranteed if you program the IC_SAR or IC_TAR to a reserved value.* **Reset value:** *IC_DEFAULT_SLAVE_ADD R configuration parameter*
IC_HS_MADD R	0x0c	3	*RW*	*0x1*	*This bit field holds the value of the I2C H S mode master code. HS-mode master codes are reserved 8-bit codes (00001xxx) that a re not used for slave addressing or other p urposes. Each master has its unique master code; up to eight highspeed mode masters can be present on the same I2C bus syste m. Valid values are from 0 to 7.* *This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set t o 0. Writes at other times have no effect.*
IC_DATA_CM D	0x10	11	*RW*	*0x0*	
	[8]	*CMD*			*This bit controls whether a read or a write is performed. This bit does not control the direction when the i2c acts as a slave. It controls only the*

					direction when it acts as a master. ■ *1 = Read* ■ *0 = Write* *When a command is entered in the TX FIFO, this bit distinguishes the write and read commands. In slave-receiver mode, this bit is a "don't care" because writes to this register are not required. In slave-transmitter mode, a "0" indicates that the data in IC_DATA_CMD is to be transmitted.* *When programming this bit, you should remember the following: attempting to perform a read operation after a General Call command has been sent results in a TX_ABRT interrupt (bit 6 of the IC_RAW_INTR_STAT register), unless bit 11 (SPECIAL) in the IC_TAR register has been cleared.* *If a "1" is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt occurs.* **Reset value:** *0x0*
	[7:0]	*DAT*			*This register contains the data to be transmitted or received on the I2C bus. If you are writing to this register and want to perform a read, bits 7:0 (DAT) are ignored by the i2c. However, when you read this register, these bits return the value of data received on the i2c interface.*
IC_SS_SCL_HCNT	0x14	16	*RW*	*0x190*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed.* *This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. For designs with APB_DATA_WIDTH = 8, the order of programming is important to ensure the correct operation of the DW_apb_i2c. The lower byte must be programmed first. Then the upper byte is programmed. When the configuration parameter*

					NOTE: *This register must not be programmed to a value higher than 65525, because i2c uses a 16-bit counter to flag an I2C bus idle condition when this counter reaches a value of IC_SS_SCL_HCNT + 10.*
IC_SS_SCL_LCNT	0x18	16	*RW*	*0x1d6*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. This register can be written only when the I2C interface is disabled which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.* *The minimum valid value is 8; hardware prevents values less than this being written, and if attempted, results in 8 being set. For designs with APB_DATA_WIDTH = 8, the order of programming is important to ensure the correct operation of DW_apb_i2c. The lower byte must be programmed first, and then the upper byte is programmed. When the configuration parameter*
IC_FS_SCL_HCNT	0x1c	16	*RW*	*0x3c*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.*
IC_FS_SCL_LCNT	0x20	16	*RW*	*0x82*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being se*

					t to 0. Writes at other times have no effect. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. If the value is less than 8 then the count value gets changed to 8.
IC_HS_SCL_HCNT	0x24	16	*RW*	*0x6*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high period count for high speed. The SCL High time depends on the loading of the bus. For 100pF loading, the SCL High time is 60ns; for 400pF loading, the SCL High time is 120ns. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set.*
IC_HS_SCL_LCNT	0x28	16	*RW*	*0x10*	*This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for high speed. The SCL low time depends on the loading of the bus. For 100pF loading, the SCL low time is 160ns; for 400pF loading, the SCL low time is 320ns.This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect.* *The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set.*
IC_INTR_STAT	0x2c	12	*R*	*0x0*	Each bit in this register has a corresponding mask bit in the IC_INTR_MASK register. These bits are cleared by reading the matching interrupt clear register. The unmasked raw versions of these bits are available in the *IC_RAW_INTR_STAT* register.

	[11]	*R_GEN_CALL*	*Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling i2c or when the CPU reads bit 0 of the IC_CLR_GEN_CALL register. i2c stores the received data in the Rx buffer.*
	[10]	*R_START_DET*	*Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether i2c is operating in slave or master mode.*
	[9]	*R_STOP_DET*	*Indicates whether a STOP condition has occurred on the I2C interface regardless of whether i2c is operating in slave or master mode.*
	[8]	*R_ACTIVITY*	*This bit captures i2c activity and stays set until it is cleared. There are four ways to clear it:* ■ *Disabling the i2c* ■ *Reading the IC_CLR_ACTIVITY register* ■ *Reading the IC_CLR_INTR register* ■ *System reset* *Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the i2c module is idle, this bit remains set until cleared, indicating that there was activity on the bus.*
	[7]	*R_RX_DONE*	*When the i2c is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.*
	[6]	*R_TX_ABRT*	*This bit indicates if i2c, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a "transmit abort". When this bit is set to 1, the IC_TX_ABRT_SOURCE register indicates the reason why the transmit abort takes places.* **NOTE:** *i2c flushes/resets/empties the TX FIFO whenever this bit is* *set. The TX FIFO remains in this flushed state until the register IC_CLR_TX_ABRT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.*

	[5]	*R_RD_REQ*	*This bit is set to 1 when i2c is acting as a slave and another I2C master is attempting to read data from i2c. The i2c holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the IC_DATA_CMD register. This bit is set to 0 just after the processor reads the IC_CLR_RD_REQ register.*
	[4]	*R_TX_EMPTY*	*This bit is set to 1 when the transmit buffer is at or below the threshold value set in the IC_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the IC_ENABLE bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines.When there is no longer activity, then with ic_en=0, this bit is set to 0.*
	[3]	*R_TX_OVER*	*Set during transmit if the transmit buffer is filled to 8 and the processor attempts to issue another I2C command by writing to the IC_DATA_CMD register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.*
	[2]	*R_RX_FULL*	*Set when the receive buffer reaches or goes above the RX_TL threshold in the IC_RX_TL register. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (IC_ENABLE[0]=0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the IC_ENABLE bit 0 is programmed with a 0, regardless of the activity that continues.*
	[1]	*R_RX_OVER*	*Set if the receive buffer is completely filled to 8 and an additional byte is received from an external I2C device. The i2c acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled*

					(IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.
	[0]	*R_RX_UNDER*			*Set if the processor attempts to read the receive buffer when it is empty by reading from the IC_DATA_CMD register. If the module is disabled (IC_ENABLE[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when ic_en goes to 0, this interrupt is cleared.*
IC_INTR_MASK	0x30	12	*RW*	*0x8ff*	These bits mask their corresponding interrupt status bits. This register is active low; a value of 0 masks the interrupt, whereas a value of 1 unmasks the interrupt.
	[11]	*M_GEN_CALL*			*These bits mask their corresponding interrupt status bits in the IC_INTR_STAT register.*
	[10]	*M_START_DET*			
	[9]	*M_STOP_DET*			
	[8]	*M_ACTIVITY*			
	[7]	*M_RX_DONE*			
	[6]	*M_TX_ABRT*			
	[5]	*M_RD_REQ*			
	[4]	*M_TX_EMPTY*			
	[3]	*M_TX_OVER*			
	[2]	*M_RX_FULL*			
	[1]	*M_RX_OVER*			
	[0]	*M_RX_UNDER*			
IC_RAW_INTR_STAT	0x34	12	*R*	*0x0*	Unlike the *IC_INTR_STAT* register, these bits are not masked so they always show the true status of the DW_apb_i2c
	[11]	*GEN_CALL*			*These bits mask their corresponding interrupt status bits in the IC_INTR_STAT register.*
	[10]	*START_DET*			
	[9]	*STOP_DET*			
	[8]	*ACTIVITY*			
	[7]	*RX_DONE*			
	[6]	*TX_ABRT*			
	[5]	*RD_REQ*			
	[4]	*TX_EMPTY*			

	[3]	*TX_OVER*			
	[2]	*RX_FULL*			
	[1]	*RX_OVER*			
	[0]	*RX_UNDER*			
IC_RX_TL	0x38	8	*RW*	*0x0*	*Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt (bit 2 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries.*
IC_TX_TL	0x3c	8	*RW*	*0x0*	*Transmit FIFO Threshold Level Controls the levels of entries (or below) that trigger the TX_EMPTY interrupt (bit 4 in IC_RAW_INTR_STAT register). The valid range is 0-255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries.*
IC_CLR_INTR	0x40	1	*R*	*0x0*	*Read this register to clear the combined interrupt, all individual interrupts, and the IC_TX_ABRT_SOURCE register. This bit does not clear hardware clearable interrupts but software clearable interrupts. Refer to Bit 9 of the IC_TX_ABRT_SOURCE register for an exception to clearing IC_TX_ABRT_SOURCE.*
IC_CLR_RX_UNDER	0x44	1	*R*	*0x0*	*Read this register to clear the RX_UNDER interrupt (bit 0) of the IC_RAW_INTR_STAT register.*
IC_CLR_RX_OVER	0x48	1	*R*	*0x0*	*Read this register to clear the RX_OVER interrupt (bit 1) of the IC_RAW_INTR_STAT register.*
IC_CLR_TX_	0x4c	1	*R*	*0x0*	*Read this register to clear the TX_OVE*

OVER					*R interrupt (bit 3) of the IC_RAW_INTR_STAT register*
IC_CLR_RD_REQ	0x50	1	*R*	*0x0*	*Read this register to clear the RD_REQ interrupt (bit 5) of the IC_RAW_INTR_STAT register.*
IC_CLR_TX_ABRT	0x54	1	*R*	*0x0*	*Read this register to clear the TX_ABRT interrupt (bit 6) of the IC_RAW_INTR_STAT register, and the IC_TX_ABRT_SOURCE register. This also releases th e TX FIFO from the flushed/reset state, allo wing more writes to the TX FIFO. Refer to Bit 9 of the IC_TX_ABRT_SOURC E register for an exception to clearing IC _TX_ABRT_SOURCE*
IC_CLR_RX_DONE	0X58	1	*R*	*0x0*	*Read this register to clear the RX_DON E interrupt (bit 7) of the IC_RAW_INT R_STAT register.*
IC_CLR_ACTIVITY	0x5c	1	*R*	*0x0*	*Reading this register clears the ACTIVIT Y interrupt if the I2C is not active anymor e. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by h ardware if the module is disabled and if th ere is no further activity on the bus. The v alue read from this register to get status of the ACTIVITY interrupt (bit 8) of the I C_RAW_INTR_STAT register.*
IC_CLR_STOP_DET	0X60	1	*R*	*0x0*	*Read this register to clear the STOP_D ET interrupt (bit 9) of the IC_RAW_IN TR_STAT register.*
IC_CLR_START_DET	0x64	1	*R*	*0x0*	*Read this register to clear the START_D ET interrupt (bit 10) of the IC_RAW_I NTR_STAT register.*
IC_CLR_GEN_CALL	0X68	1	*R*	*0x0*	*Read this register to clear the GEN_CA LL interrupt (bit 11) of IC_RAW_INTR _STAT register.*
IC_ENABLE	0X6C	1`	*Rw*	*0x0*	*Controls whether the DW_apb_i2c is enable d.* *0: Disables DW_apb_i2c (TX and RX FIFO s are held in an erased state)* *1: Enables DW_apb_i2c* *Software can disable i2c while it is active. However, it is important that care be taken to ensure that i2c is disabled properly. A*

					recommended procedure is described in "Disabling DW_apb_i2c" on page 51. *When i2c is disabled, the following occurs:* *•The TX FIFO and RX FIFO get flushed.* *•Status bits in the IC_INTR_STAT register are still active until i2c goes into IDLE state.* *If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete.* *If the module is receiving, the i2c stops the current transfer at the end of the current byte and does not acknowledge the transfer.* *In systems with asynchronous pclk and ic_clk when IC_CLK_TYPE parameter set to asynchronous (1), there is a two ic_clk delay when enabling or disabling the DW_apb_i2c.*
IC_STATUS	0X70	7	*RW*	*0x6*	This is a read-only register used to indicate the current transfer status and FIFO status. The status register may be read at any time. None of the bits in this register request an interrupt. When the I2C is disabled by writing 0 in bit 0 of the *IC_ENABLE* register: ❖ Bits 1 and 2 are set to 1 ❖ Bits 3 and 4 are set to 0
	[6]	*SLV_ACTIVITY*			*Slave FSM Activity Status. When the Slave Finite State Machine (FSM) is not in the IDLE state, this bit is set.* *0: Slave FSM is in IDLE state so the Slave part of DW_apb_i2c is not Active* *1: Slave FSM is not in IDLE state so the Slave part of DW_apb_i2c is Active*
	[5]	*MST_ACTIVITY*			*Master FSM Activity Status. When the Master Finite State Machine (FSM) is not in the IDLE state, this bit is set.* *0: Master FSM is in IDLE state so the Master part of DW_apb_i2c is not Active* *1: Master FSM is not in IDLE state so the Master part of DW_apb_i2c is Active*
	[4]	*RFF*			*Receive FIFO Completely Full. When the re*

					ceive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. *0: Receive FIFO is not full* *1: Receive FIFO is full*
	[3]	*RFNE*			*Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty.* *0: Receive FIFO is empty* *1: Receive FIFO is not empty*
	[2]	*TFE*			*Transmit FIFO Completely Empty. When the transmit FIFO is completely empty,this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt.* *0: Transmit FIFO is not empty* *1: Transmit FIFO is empty*
	[1]	*TFNF*			*Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full.* *0: Transmit FIFO is full* *1: Transmit FIFO is not full*
	[0]	*ACTIVITY*			*I2C Activity Status.*
IC_TXFLR	0x74		*R*	*0x0*	**Transmit FIFO Level**. *Contains the number of valid data entries in the transmit FIFO.*
IC_RXFLR	0X78		*R*	*0x0*	**Receive FIFO Level**. *Contains the number of valid data entries in the receive FIFO.*
IC_SDA_HOLD	0X7C	16	*RW*	*0x1*	*Sets the required SDA hold time in units of ic_clk period.*
IC_TX_ABRT_SOURCE	0x80	16	*R*	*0x0*	This register has 16 bits that indicate the source of the *TX_ABRT* bit. Except for Bit 9, this register is cleared whenever the *IC_CLR_TX_ABRT* register or the *IC_CLR_INTR* register is read. To clear Bit 9, the source of the *ABRT_SBYTE_NORSTRT* must be fixed first; RESTART must be enabled (*IC_CON*[5]=1), the SPECIAL bit

					must be cleared (*IC_TAR*[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the *ABRT_SBYTE_NORSTRT* is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the *ABRT_SBYTE_NORSTRT* is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.
	[15]	*ABRT_SLVRD_INTX*			*1: When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of IC_DATA_CMD register.*
	[14]	*ABRT_SLV_ARBLOST*			*1: Slave lost the bus while transmitting data to a remote master. IC_TX_ABRT_SOURCE[12] is set at the same time.* **Note:** *Even though the slave never "owns" the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then i2c no longer own the bus.*
	[13]	*ABRT_SLVFLUSH_TXFIFO*			*1: Slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO.*
	[12]	*ARB_LOST*			*1: Master has lost arbitration, or if IC_TX_ABRT_SOURCE[14] is also set, then the slave transmitter has lost arbitration. Note: I2C can be both master and slave at the same time.*
	[11]	*ABRT_MASTER_DIS*			*1: User tries to initiate a Master operation with the Master mode disabled.*
	[10]	*ABRT_10B_RD_NORSTRT*			*1: The restart is disabled (IC_RESTART_EN bit (IC_CON[5]) = 0) and the master sends a read command in 10-bit addres*

			sing mode.
	[9]	*ABRT_SBYTE_NORSTRT*	*To clear Bit 9, the source of the* *ABRT_SBYTE_NORSTRT* *must be fixed first; restart must be enabled (IC_CON[5]=1), the SPECIAL bit must be cleared (IC_TAR[11]), or the GC_OR_START bit must be cleared (IC_TAR[10]). Once the source of the* *ABRT_SBYTE_NORSTRT* *is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the* *ABRT_SBYTE_NORSTRT* *is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets re-asserted.1: The restart is disabled (*IC_RESTART_EN *bit (IC_CON[5]) = 0) and the user is trying to send a START Byte.*
	[8]	*ABRT_HS_NORSTRT*	*1: The restart is disabled (*IC_RESTART_EN *bit (IC_CON[5]) = 0) and the user is trying to use the master to transfer data in High Speed mode.*
	[7]	*ABRT_SBYTE_ACKDET*	*1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior).*
	[6]	*ABRT_HS_ACKDET*	*1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior).*
	[5]	*ABRT_GCALL_READ*	*1: DW_apb_i2c in master mode sent a General Call but the user programmed the byte following the General Call to be a read from the bus (*IC_DATA_CMD*[9] is set to 1).*
	[4]	*ABRT_GCALL_NOACK*	*1: DW_apb_i2c in master mode sent a General Call and no slave on the bus acknowledged the General Call.*
	[3]	*ABRT_TXDATA_NOACK*	*1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s).*
	[2]	*ABRT_10ADDR2_NOACK*	*1: Master is in 10-bit address mode and the second address byte of the 10-bit address*

					was not acknowledged by any slave
	[1]	*ABRT_10ADDR1_NOACK*			*1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.*
	[0]	*ABRT_7B_ADDR_NOACK*			*1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave.*
IC_SLV_DATA_NACK_ONLY	0x84	1	*RW*	*0x0*	*Generate NACK. This NACK generation only occurs when i2c is a slavereceiver.If this register is set to a value of 1, it can only generate a NACK after a data byte is received; hence, the data transfer is aborted and the data received is not pushed to the receive buffer. When the register is set to a value of 0, it generates NACK/ACK, depending on normal criteria.* *1 = generate NACK after data byte received* *0 = generate NACK/ACK normally*
IC_DMA_CR	0x88	2	*RW*	*0x0*	The register is used to enable the DMA Controller interface operation. There is a separate bit for transmit and receive. This can be programmed regardless of the state of IC_ENABLE
	[1]	*TDMAE*			**Transmit DMA Enable**. *This bit enables/disables the transmit FIFO DMA channel.* *0 = Transmit DMA disabled* *1 = Transmit DMA enabled*
	[0]	*RDMAE*			**Receive DMA Enable**. *This bit enables/disables the receive FIFO DMA channel.* *0 = Receive DMA disabled* *1 = Receive DMA enabled*
IC_DMA_TDLR	0x8c	2	*RW*	*0x0*	*DMATDL* **Transmit Data Level**. *This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field* *value, and TDMAE = 1.*
IC_DMA_RDL	0x90	2	*RW*	*0x0*	*DMARDL*

R					**Receive Data Level**. *This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or more than this field value + 1, and RDMAE =1. For instance, when DMARDL is 0, then dma_rx_req is asserted when 1 or more data entries are present in the receive FIFO.*
IC_SDA_SETUP	0x94	8	*RW*	*0x64*	**SDA Setup**. *It is recommended that if the required delay is 1000ns, then for an ic_clk frequency of 10 MHz, IC_SDA_SETUP should be programmed to a value of 11. IC_SDA_SETUP must be programmed with a minimum value of 2.* **Default Reset value:** *0x64*
IC_ACK_GENERAL_CALL	0x98	1	*RW*	*0x1*	**ACK General Call**. *When set to 1, DW_apb_i2c responds with a ACK (by asserting ic_data_oe) when it receives a General Call. When set to 0, the DW_apb_i2c does not generate General Call interrupts.* **Default Reset value:** *0x1*
IC_ENABLE_STATUS	0x9c	3	*R*	*0x0*	
	[2]	*SLV_RX_DATA_LOST*			**Slave Received Data Lost.** *This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting of IC_ENABLE from 1 to 0. When read as 1, DW_apb_i2c is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK.* **NOTE**: *If the remote I2C master terminates the transfer with a STOP condition before the i2c has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit is also set to 1. When read as 0, i2c is deemed to have been disabled without being actively involved in the data phase*

					of a Slave-Receiver transfer. **NOTE:** *The CPU can safely read this bit when IC_EN (bit 0) is read as 0.*
	[1]	*SLV_DISABLED_WHILE_BUSY*			**Slave Disabled While Busy (Transmit, Receive)**. *This bit indicates if a potential or active Slave operation has been aborted due to the setting of the IC_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the IC_ENABLE register while: (a) i2c is receiving the address byte of the Slave-Transmitter operation from a remote master; OR, (b) address and data bytes of the Slave-Receiver operation from a remote master.* *When read as 1, i2c is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in DW_apb_i2c (IC_SAR register) OR if the transfer is completed before IC_ENABLE is set to 0 but has not taken effect.* **NOTE**: *If the remote I2C master terminates the transfer with a STOP condition before the i2c has a chance to NACK a transfer, and IC_ENABLE has been set to 0, then this bit will also be set to 1. When read as 0, i2c is deemed to have been disabled when there is master activity, or when the I2C bus is idle.* **NOTE:** *The CPU can safely read this bit when IC_EN (bit 0) is read as 0.*
	[0]	*IC_EN*			**ic_en Status**. *This bit always reflects the value driven on the output port ic_en. When read as 1, i2c is deemed to be in an enabled state. When read as 0, i2c is deemed completely inactive.* **NOTE:** *The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_RX_DATA_LOST (bit 2) and SLV_DISABLED_WHILE_BUSY (bit 1).*
IC_FS_SPKLEN	0xA0	8	*RW*	*0x5*	*This register must be set before any I2C bus transaction can take place to ensure stabl*

					e operation. This register sets the duration, measured in ic_clk cycles, of the longest spike in the SCL or SDA lines that are filtered out by the spike suppression logic;This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. The minimum valid value is 1; hardware prevents values less than this being written, and if attempted results in 2 being set.

17. PWM寄存器(0x4000_9000)

Register Name:	PWM0_MODE_CTRL			
Address Offset	0x00			
Default Value:	0x0			
Description	PWM0 mode control register			
Bits	Field Name	Description	Type	Default
31:26	RESERVED		RW	0x0
25	Pwm0_fast_mode	Pwm0 fast mode: 0: low speed mode 1: high speed mode	RW	0x0
24	Pwm0_pol	Pwm0 polarity 0: low when duty is 100% 1: high when duty is 100%	RW	0x0
23:21	RESERVED		RW	0x0
20	Pwm0_duty_en	Load pwm0 duty value	RW	0x0
19:17	RESERVED		RW	0x0
16:12	Pwm0_duty[4:0]	Pwm0 duty, MAX 30, total 30 steps. 100% * duty[4:0]/30	RW	0x0
11:9	RESERVED		RW	0x0
8	Pwm0_freq_en	Load pwm0 frequency value	RW	0x0
7	RESERVED		RW	0x0
6:0	Pwm0_freq[6:0]	Pwm0 frequency: 13k/freq[6:0] when low speed mode;	RW	0x0

		1300K/(freq[6:0]+1) when fast speed mode.		
Register Name:		PWM1_MODE_CTRL		
Address Offset		0x10		
Default Value:		0x0		
Description		PWM1 mode control register		
Bits	Field Name	Description	Type	Default
31:26	RESERVED		RW	0x0
25	Pwm1_fast_mode	Pwm1 fast mode: 0: low speed mode 1: high speed mode	RW	0x0
24	Pwm1_pol	Pwm1 polarity 0: low when duty is 100% 1: high when duty is 100%	RW	0x0
23:21	RESERVED		RW	0x0
20	Pwm1_duty_en	Load pwm1 duty value	RW	0x0
19:17	RESERVED		RW	0x0
16:12	Pwm1_duty[4:0]	Pwm1 duty, MAX 30, total 30 steps. 100% * duty[4:0]/30	RW	0x0
11:9	RESERVED		RW	0x0
8	Pwm1_freq_en	Load pwm1 frequency value	RW	0x0
7	RESERVED		RW	0x0
6:0	Pwm1_freq[6:0]	Pwm1 frequency: 13k/freq[6:0] when low speed mode; 1300K/(freq[6:0]+1) when fast speed mode.	RW	0x0

18. GPIO寄存器(0x4004_0000)

偏移地址	寄存器名称	位宽	RW	Description	Default
0x00	**gpio_swporta_dr**	32	RW	*Values written to this register are output on the I/O signals for GPIO Port if the corresponding data direction bits are set to Output mode and the corresponding con*	0

				trol bit is set to Software mode. The valu e read back is equal to the last value w ritten to this register.	
0x04	**gpio_sw porta_dd r**	32	RW	*Values written to this register independentl y control the direction of the correspondin g data bit.* *0 – Input (default)* *1 – Output*	0
0x30	**gpio_int en**	32	RW	*Allows each bit of Port to be configured for interrupts. By default the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it config ures the corresponding bit on Port to bec ome an interrupt; otherwise,Port operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port if the corresponding data direction regist er is set to Output or if mode is set to Hardware.* *0 – Configure bit as normal GPIO signal (default)* *1 – Configure bit as interrupt*	0
0x34	**gpio_int mask**	32	RW	*Controls whether an interrupt on Port can create an interrupt for the interrupt cont roller by not masking it. By default, all i nterrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability f or this signal; otherwise interrupts are all owed through. The unmasked status can b e read as well as the resultant status aft er masking.* *0 – Interrupt bits are unmasked (default)* *1 – Mask interrupt*	0
0x 38	**gpio_intt ype_leve l**	32	RW	*Controls the type of interrupt that can oc cur on Port Whenever a 0 is written to a bit of this register, it configures the int errupt type to be level-sensitive; otherwise, it is edge-sensitive.* *0 – Level-sensitive (default)* *1 – Edge-sensitive*	0
0x3c	**gpio_int _polarity**	32	RW	*Controls the polarity of edge or level sen sitivity that can occur on input of Port. Whenever a 0 is written to a bit of this*	0

				register, it configures the interrupt type to falling-edge or active-low sensitive; other wise, it is rising-edge or active-high sensit ive. *0 – Active-low (default)* *1 – Active-high*	
0x40	**gpio_int status**	32	R	*Interrupt status of GPIO port*	0
0x44	**gpio_ra w_intstat us**	32	R	*Raw interrupt of status of GPIO port (pr emasking bits)*	0
0x4c	**gpio_por ta_eoi**	32	W	*Controls the clearing of edge type interru pts from Port .When a 1 is written into a corresponding bit of this register, the i nterrupt is cleared. All interrupts are clea red when Port is not configured for inte rrupts.* *0 – No interrupt clear (default)* *1 – Clear interrupt*	0
0x50	**gpio_ext _porta**	32	R	*When Port is configured as Input, then reading this location reads the values on the signal. When the data direction of Po rt is set as Output, reading this location reads the data register for Port.* **Reset Value:** *0x0*	0
0x6c	**gpio_ver _id_code**	32	R	*ASCII value for each number in the versi on, followed by *. For example 32_30_31 _2A represents the version 2.01**	

19. UART寄存器(0x4004_1000)

寄存器描述

Address Offset	Name	Width	R/ W	Description	Reset Value
0x00	RBR (*LCR[7] bi t = 0)*	*31:8*	R	*Reserved and read as zero*	**0x0**
		7:0	R	*Receive Buffer Register*	**0x0**

				Data byte received on the serial input port (sin) in UART mode, or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line Status Register (LSR) is set. If in FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an over-run error. *If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO is preserved, but any incoming data are lost and an over-run error occurs.*	
	THR	*31:8*	R	*Reverved*	0x0
	(*LCR[7] bit = 0*)	*7:0*	W	*Transmit Holding Register* *IData to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set. If FIFOs are disabled (FCR[0] = 0) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten.* *If FIFOs are enabled (FCR[0] = 1) and THRE is set, 16 number of characters of data may be written to the THR before the FIFO is full. Any attempt to write data when the FIFO is full results in the write data being lost.*	0x0
	DLL	31:8	R	Reserved	0x0
	(*LCR[7] bit = 1*)	*7:0*	*R / W*	*Lower 8 bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit (LCR[7]) is set and the UART is not busy (USR[0] is zero);* *The output baud rate is equal to the serial clock (pclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor). Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLL is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to p*	*0x0*

				ass before transmitting or receiving data.	
0x4	DLH (*LCR[7] bit = 1*)	*31:8*	*R*	*Reserved*	*0x0*
		7:0	*R / W*	*Upper 8-bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit (LCR[7]) is set and the UART is not busy (USR[0] is zero);.* *The output baud rate is equal to the serial clock (pclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor). Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest DW_apb_uart clock should be allowed to pass before transmitting or receiving data*	*0x0*
	IER(*LCR[7] bit = 0*)	*31:8*	*R*	*Reserved*	*0x0*
		7	*R*	*PTIME* *This is used to enable/disable the generation of THRE Interrupt. This bit is tied to be 0.* *• 0 – disabled* *• 1 – enabled*	*0x0*
		6:4	*R*	*Reserved*	*0x0*
		3	*R / W*	*EDSSI* *Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt.* *• 0 – disabled* *• 1 – enabled*	*0x0*
		2	*R / W*	*ELSI* *Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt.* *• 0 – disabled* *• 1 – enabled*	*0x0*
		1	*R / W*	*ETBEI* *Enable Transmit Holding Register Empty Interrupt. This is used to enable/disable the generation of Transmitter Holding Register Empty Interrupt. This is the third highest priority interrupt.* *• 0 – disabled* *• 1 – enabled*	*0x0*
		0	*R / W*	*ERBFI* *generation of Received Data Available Interrupt and*	*0x0*

				the Character Timeout Interrupt (if FIFOs enabled). *These are the second highest priority interrupts.* *• 0 – disabled* *• 1 – enabled*	
0x08	IIR	*31:8*	*R*	*Reserved*	*0x0*
		7:6	*R*	*FIFOs Enabled (FIFOSE)* *This is used to indicate whether the FIFOs are enabled or Disabled.* *• 00 – disabled* *• 11 – enabled*	*0x0*
		5:4	*R*	*Reserved*	*0x0*
		3:0	*R*	*Interrupt ID (IID)* *This indicates the highest priority pending interrupt which can be one of the following types:* *• 0000 – modem status* *• 0001 – no interrupt pending* *• 0010 – THR empty* *• 0100 – received data available* *• 0110 – receiver line status* *• 0111 – busy detect* *• 1100 – character timeout* *Bit 3 indicates an interrupt can only occur when the FIFOs are used to distinguish a Character Timeout condition interrupt.* *The interrupt priorities are split into several levels that are detailed in* **Table 6-2**	*0x1*
	FCR	*31:8*	*R*	*Reserved*	*0x0*
		7:6	*W*	*RCVR Trigger (RT).* *This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. It also determines when the dma_rx_req_n signal is asserted in certain modes of operation. The following trigger levels are supported:* *• 00 – 1 character in the FIFO* *• 01 – FIFO ¼ full* *• 10 – FIFO ½ full* *• 11 – FIFO 2 less than full*	*0x0*
		5:4	*R*	*TX Empty Trigger (or TET)* *Not writable.* *It determines when the dma_tx_req_n signal is asserted when in certain modes of operation.* *The following trigger levels are supported:* *• 00 – FIFO empty* *• 01 – 2 characters in the FIFO*	*0x0*

				• 10 – FIFO ¼ full • 11 – FIFO ½ full	
		3	W	DMA Mode (or DMAM) This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals. • 0 – mode 0 • 1 – mode 1	0x0
		2	W	XMIT FIFO Reset (or XFIFOR) This resets the control portion of the transmit FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	0x0
		1	W	RCVR FIFO Reset (or RFIFOR) This resets the control portion of the receive FIFO and treats the FIFO as empty. Note that this bit is 'self-clearing'. It is not necessary to clear this bit.	0x0
		0	W	FIFO Enable. This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFOs is reset.	0x0
0xc	**LCR**	31:8	R	Reserved	0x0
		7	R / W	DLAB Divisor Latch Access Bit. Divisor Latch Access Bit. Writeable only when UART is not busy (USR[0] is zero); This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.	0x0
		6	R / W	Break (or BC) Break Control Bit.This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low.	0x0
		5	R / W	Stick Parity Writeable only when UART is not busy (USR[0] is 0); This bit is used to force parity value. When PEN, EPS, and Stick Parity are set to 1, the parity bit is	0x0

				transmitted and checked as logic 0. If PEN and Stick Parity are set to 1 and EPS is a logic 0, then parity bit is transmitted and checked as a logic 1. If this bit is set to 0, Stick Parity is disabled.	
		4	*R / W*	*EPS* *Even Parity Select. writeable only when UART is not busy (USR[0] is 0); This is used to select between even and odd parity, when parity is enabled (PEN set to 1). If set to 1, an even number of logic 1s is transmitted or checked. If set to 0, an odd number of logic 1s is transmitted or checked*	*0x0*
		3	*R / W*	*PEN* *Parity Enable. Writeable only when UART is not busy (USR[0] is 0); This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.* ■ *0 – parity disabled* ■ *1 – parity enabled*	*0x0*
		2	*R / W*	*STOP* *Number of stop bits. Writeable only when UART is not busy (USR[0] is 0); This is used to select the number of stop bits per character that the peripheral transmits and receives. If set to 0, one stop bit is transmitted in the serial data.* *If set to 1 and the data bits are set to 5 (LCR[1:0] set to 0) one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver checks only the first stop bit.* ■ *0 – 1 stop bit* ■ *1 – 1.5 stop bits when DLS (LCR[1:0]) is 0, else 2 stop bit* *NOTE: The STOP bit duration implemented by DW_apb_uart may appear longer due to idle time inserted between characters for some configurations and baud clock divisor values in the transmit direction; for details on idle time between transmitted transfers*	*0x0*
		1:0	*R / W*	*DLS (or CLS, as used in legacy)* *Data Length Select. Writeable only when UART is not busy (USR[0] is 0); This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows:* ■ *00 – 5 bits*	*0x0*

				■ *01 – 6 bits* ■ *10 – 7 bits* ■ *11 – 8 bits*	
0x10	**MCR**	*31:7*	*R*	*Reserved*	*0x0*
		6	*R*	*SIRE* *SIR Mode Enable. Not writeable as this version of UART does't support SIR Mode. This is used to enable/disable the IrDA SIR Mode features.* ■ *0 – IrDA SIR Mode disabled* ■ *1 – IrDA SIR Mode enabled*	*0x0*
		5	*R*	*AFCE* *Auto Flow Control Enable.Read only as this version of UART does't support auto flow control.* ■ *0 – Auto Flow Control Mode disabled* ■ *1 – Auto Flow Control Mode enabled*	*0x0*
		4	*R / W*	*LoopBack (or LB)* *LoopBack Bit. This is used to put the UART into a diagnostic mode for test purposes. data on the sout line is held high, while serial data output is looped back to the sin line, internally. In this mode all the interrupts are fully functional. Also, in loopback mode, the modem control inputs (dsr_n, cts_n, ri_n, dcd_n) are disconnected and the modem control outputs (dtr_n, rts_n, out1_n, out2_n) are looped back to the inputs, internally.*	*0x0*
		3	*R / W*	*OUT2* *This is used to directly control the user-designated Output2 (out2_n) output (not available for user).The value written to this location is inverted and driven out on out2_n, that is:* ■ *0 – out2_n de-asserted (logic 1)* ■ *1 – out2_n asserted (logic 0)* *Note that in Loopback mode (MCR[4] set to 1), the out2_n output is held inactive high while the value of this location is internally looped back to an input.*	*0x0*
		2	*R / W*	*OUT1 (not connected)* *OUT1. This is used to directly control the user-designated Output1 (out1_n) output(not available for user).. The value written to this location is inverted and driven out on out1_n, that is:* ■ *0 – out1_n de-asserted (logic 1)* ■ *1 – out1_n asserted (logic 0)* *Note that in Loopback mode (MCR[4] set to 1), the out1_n output is held inactive high*	*0x0*

				while the value of this location is internally looped back to an input.	
		1	*R / W*	*RTS* *Request to Send. This is used to directly control the Request to Send (rts_n) output(not available for user).. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data.The rts_n signal is set low by programming MCR[1] (RTS) to a high. The rts_n signal is de-asserted when MCR[1] is set low. Note that in Loopback mode (MCR[4] set to 1), the rts_n output is held inactive high while the value of this location is internally looped back to an input.*	*0x0*
		0	*R / W*	*DTR* *Data Terminal Ready. This is used to directly control the Data Terminal Ready (dtr_n) output(not available for user).. The value written to this location is inverted and driven out on dtr_n, that is:* ■ *0 – dtr_n de-asserted (logic 1)* ■ *1 – dtr_n asserted (logic 0)* *The Data Terminal Ready output is used to inform the modem or data set that the UART is ready to establish communications.* *Note that in Loopback mode (MCR[4] set to 1), the dtr_n output is held inactive high while the value of this location is internally looped back to an input.*	*0x0*
0x14	**LSR**	*31:8*	*R*	*Reserved*	*0x0*
		7	*R*	*RFE* *Receiver FIFO Error bit. This bit is only relevant when FIFOs are enabled (FCR[0] set to 1). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO.* ■ *0 – no error in RX FIFO* ■ *1 – error in RX FIFO* *This bit is cleared when the LSR is read and the character with the error is at the top of the receiver FIFO and there are no subsequent errors in the FIFO.*	*0x0*
		6	*R*	*TEMT* *Transmitter Empty bit. If FIFOs enabled (FCR[0] set to 1), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If FIFOs are disabled, this bit is set whenever the Transmitter Holding Register and the Transmitter Shift Register are both empty.*	*0x1*

		5	*R*	*THRE* *Transmit Holding Register Empty bit. This bit indicates that the THR or TX FIFO is empty.This bit is set whenever data is transferred from the THR or TX FIFO to the transmitter shift register and no new data has been written to the THR or TX FIFO. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled.*	*0x1*
		4	*R*	*BI* *Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data.It is set whenever the serial input, sin, is held in a logic '0' state for longer than the sum of start time + data bits + parity + stop bits. A break condition on serial input causes one and only one character, consisting of all 0s, to be received by the UART. In FIFO mode, the character associated with the break condition is carried through the FIFO and is revealed when the character is at the top of the FIFO. Reading the LSR clears the BI bit. In non-FIFO mode, the BI indication occurs immediately and persists until the LSR is read.* **NOTE:** *If a FIFO is full when a break condition is received, a FIFO overrun occurs. The break condition and all the information associated with it—parity and framing errors—is discarded; any information that a break character was received is lost.*	*0x0*
		3	*R*	*FE* *Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data. In the FIFO mode, since the framing error is associated with a character received, it is revealed when the character with the framing error is at the top of the FIFO. When a framing error occurs, the DW_apb_uart tries to resynchronize. It does this by assuming that the error was due to the start bit of the next character and then continues receiving the other bit; that is, data, and/or parity and stop. It should be noted that the Framing Error (FE) bit (LSR[3]) is set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). This happens because the break character implicitly generates a framing error by*	*0x0*

				holding the sin input to logic 0 for longer than the duration of a character. ■ *0 – no framing error* ■ *1 – framing error* *Reading the LSR clears the FE bit.*	
		2	*R*	*PE* *Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set. In the FIFO mode, since the parity error is associated with a character received, it is revealed when the character with the parity error arrives at the top of the FIFO. It should be noted that the Parity Error (PE) bit (LSR[2]) can be set if a break interrupt has occurred, as indicated by Break Interrupt (BI) bit (LSR[4]). In this situation, the Parity Error bit is set if parity generation and detection is enabled (LCR[3]=1) and the parity is set to odd (LCR[4]=0).* ■ *0 – no parity error* ■ *1 – parity error* *Reading the LSR clears the PE bit.*	*0x0*
		1	*R*	*OE* *Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read. In the non-FIFO mode, the OE bit is set when a new character arrives in the receiver before the previous character was read from the RBR. When this happens, the data in the RBR is overwritten. In the FIFO mode, an overrun error occurs when the FIFO is full and a new character arrives at the receiver. The data in the FIFO is retained and the data in the receive shift register is lost.* ■ *0 – no overrun error* ■ *1 – overrun error* *Reading the LSR clears the OE bit.*	*0x0*
		0	*R*	*DR* *Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO.* ■ *0 – no data ready* ■ *1 – data ready* *This bit is cleared when the RBR is read in non-FIFO mode, or when the receiver FIFO is empty, in FIFO mode.*	*0x0*

0x18	**MSR**	*31:8*	*R*	*Reserved*	*0x0*
		7	*R*	*DCD* *Data Carrier Detect. This is used to indicate the current state of the modem control line dcd_n. This bit is the complement of dcd_n. When the Data Carrier Detect input (dcd_n) is asserted it is an indication that the carrier has been detected by the modem or data set.* ■ *0 – dcd_n input is de-asserted (logic 1)* ■ *1 – dcd_n input is asserted (logic 0) In Loopback Mode (MCR[4] set to 1), DCD is the same as MCR[3] (Out2).*	*0x0*
		6	*R*	*RI* *Ring Indicator. This is used to indicate the current state of the modem control line ri_n.This bit is the complement of ri_n. When the Ring Indicator input (ri_n) is asserted it is an indication that a telephone ringing signal has been received by the modem or data set.* ■ *0 – ri_n input is de-asserted (logic 1)* ■ *1 – ri_n input is asserted (logic 0)* *In Loopback Mode (MCR[4] set to 1), RI is the same as MCR[2] (Out1).*	*0x0*
		5	*R*	*DSR* *Data Set Ready. This is used to indicate the current state of the modem control line dsr_n.This bit is the complement of dsr_n. When the Data Set Ready input (dsr_n) is asserted it is an indication that the modem or data set is ready to establish communications with the DW_apb_uart.* ■ *0 – dsr_n input is de-asserted (logic 1)* ■ *1 – dsr_n input is asserted (logic 0)* *In Loopback Mode (MCR[4] set to 1), DSR is the same as MCR[0] (DTR).*	*0x0*
		4	*R*	*CTS* *Clear to Send. This is used to indicate the current state of the modem control line cts_n.This bit is the complement of cts_n. When the Clear to Send input (cts_n) is asserted it is an indication that the modem or data set is ready to exchange data with the DW_apb_uart.* ■ *0 – cts_n input is de-asserted (logic 1)* ■ *1 – cts_n input is asserted (logic 0)* *In Loopback Mode (MCR[4] = 1), CTS is the same as MCR[1] (RTS).*	*0x0*

		3	*R*	*DDCD* *Delta Data Carrier Detect. This is used to indicate that the modem control line dcd_n has changed since the last time the MSR was read.* ■ *0 – no change on dcd_n since last read of MSR* ■ *1 – change on dcd_n since last read of MSR* *Reading the MSR clears the DDCD bit. In Loopback Mode (MCR[4] = 1), DDCD reflects changes on MCR[3] (Out2).Note, if the DDCD bit is not set and the dcd_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDCD bit is set when the reset is removed if the dcd_n signal remains asserted.*	*0x0*
		2	*R*	*TERI* *Trailing Edge of Ring Indicator. This is used to indicate that a change on the input ri_n (from an active-low to an inactive-high state) has occurred since the last time the MSR was read.* ■ *0 – no change on ri_n since last read of MSR* ■ *1 – change on ri_n since last read of MSR* *Reading the MSR clears the TERI bit. In Loopback Mode (MCR[4] = 1), TERI reflects when MCR[2] (Out1) has changed state from a high to a low.*	*0x0*
		1	*R*	*DDSR* *Delta Data Set Ready. This is used to indicate that the modem control line dsr_n has changed since the last time the MSR was read.* ■ *0 – no change on dsr_n since last read of MSR* ■ *1 – change on dsr_n since last read of MSR* *Reading the MSR clears the DDSR bit. In Loopback Mode (MCR[4] = 1), DDSR reflects changes on MCR[0] (DTR). Note, if the DDSR bit is not set and the dsr_n signal is asserted (low) and a reset occurs (software or otherwise), then the DDSR bit is set when the reset is removed if the dsr_n signal remains asserted.*	*0x0*
		0	*R*	*DCTS* *Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read.* ■ *0 – no change on cts_n since last read of MSR* ■ *1 – change on cts_n since last read of MSR* *Reading the MSR clears the DCTS bit. In Loopback Mode (MCR[4] = 1), DCTS reflects changes on*	*0x0*

				MCR[1] (RTS). *Note, if the DCTS bit is not set and the cts_n signal is asserted (low) and a reset occurs (software or otherwise), then the DCTS bit is set when the reset is removed if the cts_n signal remains asserted.*	
0x1C	**SCR**	*31:8*	*R*	*Reserved*	*0x0*
		7:0	*R / W*	*Scratchpad Register* *This register is for programmers to use as a temporary storage space. It has no defined purpose in the DW_apb_uart.*	*0x0*
0x70	**FAR**	*31:1*	*R*	*Reserved*	*0x0*
		0	*R*	*FIFO Access Register* *This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFOs are implemented and enabled. When FIFOs are not implemented or not enabled it allows the RBR to be written by the master and the THR to be read by the master.* ■ *0 – FIFO access mode disabled* ■ *1 – FIFO access mode enabled*	*0x0*
0x7C	**USR**	*31:5*	*R*	*Reserved*	*0x0*
		4	*R*	*RFF* *Receive FIFO Full. This bit is not valid in current version.*	*0x0*
		3	*R*	*RFNE* *Receive FIFO Not Empty. This bit is not valid in current version.*	*0x0*
		2	*R*	*TFE* *Transmit FIFO Empty. This bit is not valid in current version.*	*0x0*
		1	*R*	*TFNF* *Transmit FIFO Not Full. This bit is not valid in current version.*	*0x0*
		0	*R*	*BUSY* *UART Busy. This bit indicates that a serial transfer is in progress; when cleared, indicates that the DW_apb_uart is idle or inactive.* ■ *0 –uart is idle or inactive* ■ *1 –uart is busy (actively transferring data)* *This bit will be set to 1 (busy) under any of the following conditions:* *1. Transmission in progress on serial interface*	*0x0*

				2. Transmit data present in THR, when FIFO access mode is not being used (FAR = 0) and the baud divisor is non-zero ({DLH,DLL} does not equal 0) when the divisor latch access bit is 0 (LCR.DLAB = 0) *3. Reception in progress on the interface* *4. Receive data present in RBR, when FIFO access mode is not being used (FAR = 0)* **NOTE:** *It is possible for the UART Busy bit to be cleared even though a new character may have been sent from another device. That is, if the uart has no data in THR and RBR and there is no transmission in progress and a start bit of a new character has just reached the uart. This is due to the fact that a valid start is not seen until the middle of the bit period and this duration is dependent on the baud divisor that has been programmed.*	
0xA4	HTX	*31:1*	*R*	*Reserved*	*0x0*
		0		*Halt TX* *This register is use to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled.* ■ *0 – Halt TX disabled* ■ *1 – Halt TX enabled* *Note, if FIFOs are not enabled, the setting of the halt TX register has no effect on operation.*	*0x0*
0xA8	**DMASA**	*31:1*	*R*	*Reserved*	*0x0*
			W	*DMA Software Acknowledge* *Not supported in current version*	

Table 6-2 Interrupt Control Functions

Interrupt Set and Reset Functions				
ID	**Priority Level**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
0001	–	*None*	*None*	–
0110	*Highest*	*Receiver line status*	*Overrun/parity/ framing errors or break interrupt*	*Reading the line status register*
0100	*Second*	*Received data available*	*Receiver data available (FIFOs disabled) or RCVR FIFO trigger level reached (FIFOs enabled)*	*Reading the receiver buffer register (FIFOs disabled) or the FIFO drops below the trigger level (FIFOs enabled)*
1100	*Second*	*Character timeout indication*	*No characters in or out of the RCVR FIFO during the last 4 character times and there is at*	*Reading the receiver buffer register*

			least 1 character in it during this time	
0010	*Third*	*Transmit holding register empty*	*Transmitter holding register empty*	*Reading the IIR register (if source of interrupt); or, writing into THR (FIFOs)*
0000	*Fourth*	*Modem status*	*Clear to send or data set ready or ring indicator or data carrier detects.*	*Reading the line status register*
0111	*Busy detect indication*	*Busy detect indication*	*Master has tried to write to the Line Control Register while the DW_apb_uart is busy (USR[0] is set to one).*	*Reading the UART status register*

20. I2S & PWM Audio寄存器(0x4004_3000)

20.1 PWM模式说明

Pwm 模式：

i2s 设置成slaveTX模式，ws 和sck由pwm mod模块提供，i2s 将音频数据传给pwm mod模块，调制成pwm波输出到功放。

I2s模块增加控制寄存器pwm mod。

20.2 寄存器说明

Register Name:	I2S_CTRL			
Address Offset	0x00			
Default Value:	0x01900000			
Description	I2S Control Register			
Bits	Field Name	Description	Type	Default
31:29	RESERVED		R	0x0
28	rsync_loop_back	Loop-back configuration bit, for receiver synchronization unit. When 0 (normal mode), the scki and wsi inputs of the I2S module (configured to be receiver synchronization unit) are connected to the external inputs rclki and rwsi. When 1 (loop-back mode) the scki and wsi inputs of the I2S module (configured to be receiver synch	RW	0

		ronization unit) are connected to the transmitte r synchronization unit outputs tscko and twso. Sampled on the rising edge of the clock.		
27	tsync_loop_back	Loop-back configuration bit, for transmitter synchronization unit. When 0 (normal mode), th e scki and wsi inputs of the I2S module (confi gured to be transmitter synchronization unit) ar e connected to the external inputs tclki and tw si. When 1 (loop-back mode) the scki and wsi inputs of the I2S module (configured to be tra nsmitter synchronization unit) are connected to the receiver synchronization unit outputs rscko and rwso. Sampled on the rising edge of the clock.	RW	0
26	rsync_rst	Reset for receiver synchronizing unit. Active LO W. Sampled on the rising edge of the clock.	RW	0
25	tsync_rst	Reset for transmitter synchronizing unit. Active LOW. Sampled on the rising edge of the clock	RW	0
24	rfifo_rst	Receive FIFO reset. When '0', receive FIFO poi nters are reset to zero. Threshold level for this FIFO is unchanged. This bit is automatically set to '1' a fter one clock cycle. Sampled on the rising edg e of the clock.	RW	1
23	tfifo_rst	Transmit FIFO reset. When '0', transmit FIFO p ointers are reset to zero. Threshold level for thi s FIFO is unchanged. This bit is automatically s et to '1' after one clock cycle. Sampled on the rising edge of the clock.	RW	1
22	r_ms	Master (value '1') or slave (value '0') configurat ion bit for unit synchronizing all receivers with I2S bus. Sampled on the rising edge of the clock.	RW	0

21	t_ms	Master (value '1') or slave (value '0') configuration bit for unit synchronizing all transmitters with I2S bus. Sampled on the rising edge of the clock.	RW	0
20	sfr_rst	SFR block synchronous reset. When '0', all bits in SFR registers are reset to default values. This bit is automatically set to '1' after one clock cycle. Sampled on the rising edge of the clock.	RW	1
19	loop_back_6_7	channels 6 and 7 into the loop-back mode. In this mode channels 6 and 7 can work in both directions depending on configuration bits. Default value '0' causes normal operation without loop-back. Sampled on the rising edge of the clock.	RW	0
18	loop_back_4_5	channels 4 and 5 into the loop-back mode. In this mode channels 4 and 5 can work in both directions depending on configuration bits. Default value '0' causes normal operation without loop-back. Sampled on the rising edge of the clock.	RW	0
17	loop_back_2_3	channels 2 and 3 into the loop-back mode. In this mode channels 2 and 3 can work in both directions depending on configuration bits. Default value '0' causes normal operation without loop-back. Sampled on the rising edge of the clock.	RW	0
16	loop_back_0_1	channels 0 and 1 into the loop-back mode. In this mode channels 0 and 1 can work in both directions depending on configuration bits. Default value '0' causes normal operation without loop-back. Sampled on the rising edge of the clock.	RW	0
15	tr_cfg_7	Transmitter (value '1') or receiver (value '0')	RW	0x0

		configuration bit for I2S channel 7. Sampled on the rising edge of the clock.		
14	tr_cfg_6	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 6. Sampled on the rising edge of the clock.	RW	0x0
13	tr_cfg_5	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 5. Sampled on the rising edge of the clock.	RW	0x0
12	tr_cfg_4	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 4. Sampled on the rising edge of the clock.	RW	0x0
11	tr_cfg_3	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 3. Sampled on the rising edge of the clock.	RW	0x0
10	tr_cfg_2	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 2. Sampled on the rising edge of the clock.	RW	0x0
9	tr_cfg_1	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 1. Sampled on the rising edge of the clock.	RW	0x0
8	tr_cfg_0	Transmitter (value '1') or receiver (value '0') configuration bit for I2S channel 0. Sampled on the rising edge of the clock.	RW	0x0
7	i2s_en_7	Enable bit for I2S channel 7. Value '0' causes r eset signal for this channel (i2s_rst_7), configuration	RW	0x0

		SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.		
6	i2s_en_6	Enable bit for I2S channel 6. Value '0' causes reset signal for this channel (i2s_rst_6), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
5	i2s_en_5	Enable bit for I2S channel 5. Value '0' causes reset signal for this channel (i2s_rst_5), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
4	i2s_en_4	Enable bit for I2S channel 4. Value '0' causes reset signal for this channel (i2s_rst_4), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
1	i2s_en_3	Enable bit for I2S channel 3. Value '0' causes reset signal for this channel (i2s_rst_3), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
1	i2s_en_2	Enable bit for I2S channel 2. Value '0' causes reset signal for this channel (i2s_rst_2), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
1	i2s_en_1	Enable bit for I2S channel 1. Value '0' causes r	RW	0x0

		eset signal for this channel (i2s_rst_1), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.		
0	i2s_en_0	Enable bit for I2S channel 0. Value '0' causes reset signal for this channel (i2s_rst_0), configuration SFR bits for this channel are unchanged. Value '1' enables channel. Sampled on the rising edge of the clock.	RW	0x0
Register Name:		I2S_STAT		
Address Offset		0x04		
Default Value:		0x00001100		
Description		I2S Status Register		
Bits	Field Name	Description	Type	Default
31:16	RESERVED		R	0x0
15	rfifo_afull	Receive FIFO almost full flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0
14	rfifo_full	Receive FIFO full flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0
13	rfifo_aempty	Receive FIFO almost empty flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0
12	rfifo_empty	Receive FIFO empty flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x1
11	tfifo_afull	Transmit FIFO almost full flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0

10	tfifo_full	Transmit FIFO full flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0
9	tfifo_aempty	Transmit FIFO almost empty flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x0
8	tfifo_empty	Transmit FIFO empty flag. Active HIGH. Updated on the rising edge of the clock. This bit can trigger the interrupt.	RW	0x1
[7:5]	ovrerr_code	Code of the receiver that caused overrun error. Updated on the rising edge of the clock. The code is a binary notation of the channel's number.	RO	0x0
4	rdata_ovrerr	Indicates receiver data overrun error, active HIGH. Sampled and updated on the rising edge of the clock. This bit can trigger the interrupt. Writing a LOW value to this bit resets all receivers and the receive FIFO. The receiver configuration is preserved.	RW	0x0
[3:1]	underr_code	Code of the transmitter that caused underrun error. Updated on the rising edge of the clock. The code is a binary notation of the channel's number.	RO	0x0
0	tdata_underr	Indicates transmitter data underrun error, active HIGH. Sampled and updated on the rising edge of the clock. This bit can trigger the interrupt. Writing a LOW value to this bit resets all transmitters. The transmit FIFO contents and pointers are preserved. The transmitter configuration is preserved.	RW	0x0
Register Name:		I2S_SRR		
Address Offset		0x08		

Default Value:		0x0		
Description		I2S Channels Sample Rate & Resolution Configuration Register		
Bits	Field Name	Description	Type	Default
[31:27]	rresolution	Receiver resolution (0 to 31). Sampled on the rising edge of the clock. It simply should be assigned the value equal to the number of valid bits minus one.	RW	0x0
[26:23]	reserved		RO	0x0
[22:16]	rsample_rate	Receiver sample rate. Sampled on the rising edge of the clock.	RW	0x0
[15:11]	tresolution	Transmitter resolution (0 to 31). Sampled on the rising edge of the clock. It simply should be assigned the value equal to the number of valid bits minus one.	RW	0x0
[10:7]	reserved		RO	0x0
[6:0]	tsample_rate	Transmitter sample rate. Sampled on the rising edge of the clock.	RW	0x0
Register Name:		CID_CTRL		
Address Offset		0x0C		
Default Value:		0x0		
Description		Clock, Interrupt and DMA Control Register		
Bits	Field Name	Description	Type	Default
31	rfifo_afull_mask	Bit masking interrupt request generation after receive FIFO becomes almost full. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0

30	rfifo_full_mask	Bit masking interrupt request generation after receive FIFO becomes full. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
29	rfifo_aempty_mask	Bit masking interrupt request generation after receive FIFO becomes almost empty. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
28	rfifo_empty_mask	Bit masking interrupt request generation after receive FIFO becomes empty. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
27	tfifo_afull_mask	Bit masking interrupt request generation after transmit FIFO becomes almost full. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
26	tfifo_full_mask	Bit masking interrupt request generation after transmit FIFO becomes full. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
25	tfifo_aempty_mask	Bit masking interrupt request generation after transmit FIFO becomes almost empty. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
24	tfifo_empty_mask	Bit masking interrupt request generation after transmit FIFO becomes empty. When LOW, masks generation of interrupt request. Sampled on the rising edge of the clock.	RW	0x0
[23:21]	Reserved		RO	0x0
20	i2s_mask_4	Bit masking interrupt request generation after underrun / overrun condition occurrence in I2S c	RW	0x0

		hannel 4. When LOW, masks generation of interrupt request caused by the channel 4. Sampled on the rising edge of the clock.		
19	i2s_mask_3	Bit masking interrupt request generation after underrun / overrun condition occurrence in I2S channel 3. When LOW, masks generation of interrupt request caused by the channel 3. Sampled on the rising edge of the clock.	RW	0x0
18	i2s_mask_2	Bit masking interrupt request generation after underrun / overrun condition occurrence in I2S channel 2. When LOW, masks generation of interrupt request caused by the channel 2. Sampled on the rising edge of the clock.	RW	0x0
17	i2s_mask_1	Bit masking interrupt request generation after underrun / overrun condition occurrence in I2S channel 1. When LOW, masks generation of interrupt request caused by the channel 1. Sampled on the rising edge of the clock.	RW	0x0
16	i2s_mask_0	Bit masking interrupt request generation after underrun / overrun condition occurrence in I2S channel 0. When LOW, masks generation of interrupt request caused by the channel 0. Sampled on the rising edge of the clock.	RW	0x0
15	intreq_mask	Bit masking all interrupt requests. When '0' all interrupts are masked, when '1' interrupts use individual masks. Sampled on the rising edge of the clock.	RW	0x0
[14:10]	Reserved		RO	0x0
9	strobe_rs	Clock enable for the unit synchronizing receivers. When high the clk_rs clock is blocked, else it is enabled.	RW	0x0

		Sampled on the rising edge of the clock.		
8	strobe_ts	Clock enable for the unit synchronizing transmitters. When high the clk_ts clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0
[7:5]	Reserved		RO	0x0
4	i2s_strobe_4	Clock enable, channel 4. When high the clk_4 clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0
3	i2s_strobe_3	Clock enable, channel 3. When high the clk_3 clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0
2	i2s_strobe_2	Clock enable, channel 2. When high the clk_2 clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0
1	i2s_strobe_1	Clock enable, channel 1. When high the clk_1 clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0
0	i2s_strobe_0	Clock enable, channel 0. When high the clk_0 clock is blocked, else it is enabled. Sampled on the rising edge of the clock.	RW	0x0

Register Name:	TFIFO_STAT
Address Offset	0x10
Default Value:	0x0
Description	Transmit FIFO Status Register

Bits	Field Name	Description	Type	Default
31:6	RESERVED		R	0x0
[5:0]	tlevel	Indicates transmit FIFO level. Updated on the rising edge of the clock.	RO	0x0

Register Name:	RFIFO_STAT
Address Offset	0x14

Default Value:		0x0		
Description		Receive FIFO Status Register		
Bits	Field Name	Description	Type	Default
31:8	RESERVED		R	0x0
rlevel	[5:0]	Indicates receive FIFO level. Updated on the rising edge of the clock.	RO	0x0
Register Name:		TFIFO_CTRL		
Address Offset		0x18		
Default Value:		0x000F0000		
Description		Transmit FIFO Control Register		
Bits	Field Name	Description	Type	Default
31:8	RESERVED		R	0x0
[20:16]	tafull_threshold	Determines threshold for almost full flag in the transmit FIFO. Sampled on the rising edge of the clock.	RW	0xF
[15:5]	Reserved		RO	0x0
[4:0]	taempty_threshold	Determines threshold for almost empty flag in the transmit FIFO. Sampled on the rising edge of the clock.	RW	0x0
Register Name:		RFIFO_CTR		
Address Offset		0x1C		
Default Value:		0x000F0000		
Description		Receive FIFO Control Register		
Bits	Field Name	Description	Type	Default
31:8	RESERVED		R	0x0
[20:16]	rafull_threshold	Determines threshold for almost full flag in the receive FIFO. Sampled on the rising edge of the clock	RW	0x0
[15:	Reserved		RO	0x0

5]				
[4:0]	raempty_threshold	Determines threshold for almost empty flag in the receive FIFO. Sampled on the rising edge of the clock.	RW	0x0
Register Name:		DEV_CONF		
Address Offset		0x20		
Default Value:		0x00000208		
Description		Device Configuration Register		
Bits	Field Name	Description	Type	Default
31:8	RESERVED		R	0x0
7:5	Pwm_src_sel	Select which i2s channel to pwm modulator 0：i2s channel 0 1: i2s channel 1 2: i2s channel 2 3: i2s channel 3 4: i2s channel 4 Others:i2s channel 0	RW	0x0
4	Pwm_mute	Pwm mute funtion 0: no mute 1: mute	RW	0x0
3	Pwm_mode_en	Pwm modulator enable	RW	0x0
2:0	Pwm_samp_rate	Samping rate selection 0-> 22.05KHz 1-> 24KHz 2-> 32KHz 3-> 44.1KHz 4-> 48KHz Others-> reserved	RW	0x0
Register Name:		PWM_CONF		
Address Offset		0x24		
Default Value:		0x0		

Description		PWM modulator configuration		
Bits	Field Name	Description	Type	Default
31:8	RESERVED		R	0x0
7:5	Pwm_src_sel	Select which i2s channel to pwm modulator 0：i2s channel 0 1: i2s channel 1 2: i2s channel 2 3: i2s channel 3 4: i2s channel 4 Others:i2s channel 0	RW	0x0
4	Pwm_mute	Pwm mute funtion 0: no mute 1: mute	RW	0x0
3	Pwm_mode_en	Pwm modulator enable	RW	0x0
2:0	Pwm_samp_rate	Samping rate selection 0-> 22.05KHz 1-> 24KHz 2-> 32KHz 3-> 44.1KHz 4-> 48KHz Others-> reserved	RW	0x0

21. QSPI寄存器(0x4014_0000)

21.1 QSPI寄存器地址映射

地址偏移量	属性	位宽	名称
00	RW	9:0	CTRL_R0
04	RW	15:0	CTRL_R1
08	RW	1:0	SSI_ENR
0C	RW	9:0	BAUD
10	RW	4:0	TXF_TLR
14	RW	3:0	RXF_TLR
18	R	4:0	TXF_LR
1C	R	4:0	RX_LR
20	R	4:0	SR
24	RW	5:0	IMR
28	RW	5:0	ISR
2C	RW	5:0	RISR
30	R	0	TXOICR
34	R	0	RXOICR
38	R	0	RXUICR
3C	R	0	AHBICR
40	R	0	ICR
44	RW	1:0	HOLD_WP
48	RW	19:0	READ_CMD
4C	RW	7:0	PGM_CMD
50	R	0	CACHE_FLUSH
54	RW	0	CACHE_DIS_UPDATE
58	R	0	TXFIFO_FLUSH
5C	R	0	RXFIFO_FLUSH

60	RW	1:0	DMA_CTRL
64	RW	4:0	DMA_TDLR
68	RW	3:0	DMA_RDLR
6C~FC	RW	15:0	DR

21.2寄存器说明

21.2.1 CTRLR0

在SSI enable时，本寄存器不可写。

Bits	Name	R/W	Description
3:0	DFS	R/W	Data Frame Size. 访问字长，4-16bits有效，复位为0x7，代表每个地址为8bits 在transmit/receive FIFO内，都是右对齐。 0000/0001/0010/0011四个值保留
5:4	FRF	R/W	Frame Format 00, Motorala SPI 01,Texas Instruments SSP 10,National Semiconductors Microwire 11,Reserved 这个对应的时序不同，因为clock极性等有专门的控制位，是否需要这个县在还不清楚。 有可能为只读
6	SCPH	R/W	Serial clock Phase 0， SCK在第一个data bit的中间翻转 1， SCK在第一个data bit的起始翻转 Reset value:1
7	SCPOL	R/W	Serial clock polarity 0, Inactive state of serial clock is low 1,Inactive state of serial clock is high Reset value: 1
9:8	TMOD	R/W	Transfer Mode

			00------ Transmit and Receive 01------ Transmit only 10------ receive only 11------ EEPROM Read Reset Value: 11
12:10	E2PRMODE	R/W	REGISTER ACCESS SPI READ MODE 仅仅在TMODE ==11有效 000: standard SPI read, 001: standard SPI fast read 010: dual output fast read 011: quad output fast read 100: dual I/O fast read 101: quad I/O fast read 110: quad I/O word read 111：invalid Reset: 000
13	CONTINOUS	R/W	仅仅在为E2PRMODE为100,101,110时有效 为1，表示为连续模式，不需要发CMD； 为0，表示为首次模式，需要发CMD。 注，dummy不需要发 Reset ： 0

21.2.2 CTRLR1

在SSI enable时，本寄存器不可写。

Bits	Name	R/W	Description
15:0	NDF	R/W	Number of Data Frames 仅仅在TMOD=10/11，作为连续接收的数据个数。接收的个数为本寄存器值加1。 Reset 0

21.2.3 SSIENR

Bits	Name	R/W	Description
0	SSI_EN	R/W	SSI Enable 当disable时，所有传输立即停止。收发FIFO清空。 当enable时，有些寄存器不可以修改。 当disable时，ssi_sleep被输出（有delay）通知系统可以关闭ssi_clk，从而节省系统功耗。 其次与SSI_AHB_EN不同时有效 Reset 0
1	SSI_AHB_EN	R/W	SSI AHB enable 当disable时，不可以接受类似ram的直接访问 其次与SSI_EN不同时有效 Reset 1

21.2.4 BAUDR

在SSI enable时，本寄存器不可写。

Bits	Name	R/W	Description
9:0	SCKDIV	R/W	用来产生SCK，使用模块外部的pll_clk（480MHz）来产生 最高为120M，至少支持如下档位： 120M，60M，30M，15M，7.5M，3.75M，1.875M，0.9375M 80M，40M，20M，10M，5M，2.5M，1.25M，0.625M 共16档,建议仅仅采用如下配置，reset value为40M 00_0000_0100，120M 00_0000_1000，60M 00_0001_0000，30M 00_0010_0000，15M 00_0100_0000，7.5M 00_1000_0000，3.75M 01_0000_0000，1.875M 10_0000_0000，0.9375M

			00_0000_0110, 80M 00_0000_1100, 40M 00_0001_1000, 20M 00_0011_0000, 10M 00_0110_0000, 5M 00_1100_0000, 2.5M 01_1000_0000, 1.25M 11_0000_0000, 0.625M

21.2.5 TXFTLR

在SSI enable时，本寄存器不可写。

Transmit FIFO Threshold寄存器

当TXFIFO超出此深度时，产生中断。

宽度为TX_ABW

当写的值超过FIFO深度时，维持原来的值。

（主要是提醒cpu表示txfifo数据水位低于警戒线，必须往其内部写新的数据了，以满足spi的连续操作）

Bits	Name	R/W	Description
4:0	TFT	R/W	当TXFIFO达到这种情况时，产生中断。 0_0000, TXFIFO有0个为数据（fifo空）时，产生中断 0_0001, TXFIFO有1个或以下为数据时，产生中断 0_0010, TXFIFO有2个或以下为数据时，产生中断 …. Reset 0

21.2.6 RXFTLR

在SSI enable时，本寄存器不可写。

Receive FIFO Threshold寄存器

当RXFIFO超出此深度时，产生中断。

宽度为RX_ABW

当写的值超过FIFO深度时，维持原来的值。

（表示rxfifo的数据个数已经超出了警戒线，必须马上读走。）

Bits	Name	R/W	Description
3:0	RFT	R/W	当RXFIFO收满到这种情况时，产生中断。 0000, RXFIFO有1个或以上为数据时，产生中断 0001, RXFIFO有2个或以上为数据时，产生中断 0010,RXFIFO有3个或以上为数据时，产生中断 …. Reset 0

21.2.7 TXFLR

Transmit FIFO Level Register

Bits	Name	R/W	Description
TX_ABW-1:0	TXTFL	R	当TXFIFO内的数据的个数 Reset 0

21.2.8 RXFLR

Receive FIFO Level Register

Bits	Name	R/W	Description
RX_ABW-1:0	RXTFL	R	当RXFIFO内的数据的个数 Reset 0

21.2.9 SR

Status register

Bits	Name	R/W	Description
0	BUSY	R	SSI Busy Flag. When set, indicates that a serial transfer is in progress; wh en cleared indicates that the DW_apb_ssi is idle or disable d 0 – idle or disabled 1- Actively transferring data Reset 0
1	TFNF	R	TXFIFO not Full. 0- TX FIFO is full 1- TX FIFO is not full Reset 1
2	TFE	R	TXFIFO Empty 0- TXFIFO is not empty 1- TXFIFO is empty Reset 1
3	RFNE	R	RXFIFO not Empty 0- RXFIFO is empty 1- RXFIFO is not empty Reset 0
4	RFF	R	RXFIFO full 0- RXFIFO is not full 1- RXFIFO is full Reset 0

21.2.10 IMR

Interrupt mask register

Bits	Name	R/W	Description
0	TXEIM	RW	Transmit FIFO Empty Interrupt Mask 0- Ssi_txe_intr interrupt is masked 1- Ssi_txe_intr interrupt is not masked Reset : 1

1	TXOIM	RW	Transmit FIFO Overflow interrupt mask Ssi_txo_intr Reset : 1
2	RXUIM	RW	Receive FIFO underflow interrupt mask Ssi_rxu_intr Reset : 1
3	RXOIM	RW	Receive FIFO Overflow interrupt mask Ssi_rxo_intr Reset : 1
4	RXFIM	RW	Receive FIFO Full interrupt mask Ssi_rxf_instr Reset : 1
5	AHBIM	RW	AHB非法访问中断的评比 Reset : 1

21.2.11 ISR

Interrupt status register

被屏蔽之后的中断状态。

Bits	Name	R/W	Description
0	TXEIS	R	Transmit FIFO Empty(threshold) Interrupt status Ssi_txe_intr 往TXFIFO写数据将清除此中断 Reset 0
1	TXOIS	R	Transmit FIFO Overflow interrupt status Ssi_txo_intr 读TXOICR/ICR清除此中断 Reset 0
2	RXUIS	R	Receive FIFO underflow interrupt status Ssi_rxu_intr 读RXUICR/ICR清除此中断 Reset 0
3	RXOIS	R	Receive FIFO Overflow interrupt status Ssi_rxo_intr

			读RXOICR/ICR清除此中断 Reset 0
4	RXFIS	R	Receive FIFO Full interrupt(threshold) status Ssi_rxf_instr 从RXFIFO读数据将清除此中断 Reset 0
5	AHB_IS	R	在ssi_ena=1或者ssi_ahb_ena=0时，访问了flash空间，将触发此中断 读AHBICR/ICR清除此中断 Reset 0

21.2.12 RISR

Raw Interrupt status register

在屏蔽之前的中断状态。

Bits	Name	R/W	Description
0	TXEIS	R	Transmit FIFO Empty(threshold) Interrupt status Ssi_txe_intr Reset 0
1	TXOIS	R	Transmit FIFO Overflow interrupt status Ssi_txo_intr Reset 0
2	RXUIS	R	Receive FIFO underflow interrupt status Ssi_rxu_intr Reset 0
3	RXOIS	R	Receive FIFO Overflow interrupt status Ssi_rxo_intr Reset 0
4	RXFIS	R	Receive FIFO Full (threshold) interrupt status Ssi_rxf_instr Reset 0
5	AHB_IS	R	在ssi_ena=1或者ssi_ahb_ena=0时，访问了flash空间，将触发此中断 读AHBICR/ICR清除此中断

			Reset 0

21.2.13 TXOICR

TX FIFO overflow interrupt clear register

读本寄存器清除 ssi_txo_intr
写操作无效。

21.2.14 RXOICR

RXFIFO overflow interrupt clear register
读本寄存器清除 ssi_rxo_intr
写操作无效。

21.2.15 RXUICR

RXFIFO underflow interrupt clear register
读本寄存器清除 ssi_rxu_intr
写操作无效。

21.2.16 AHBICR

读本寄存器清除ssi_ahb_intr
写操作无效

21.2.17 ICR

interrupt clear register
读本寄存器清除 ssi_rxu_intr，ssi_rxo_intr, ssi_txo_intr, ssi_ahb_intr

写操作无效

21.2.18 HOLD_WP

RW，在非quad_spi情况下，WP#，HOLD#的输出

Bit 0, WP#输出

Bit1，HOLD#输出

Reset value: 2'b11

21.2.19 READ_CMD

Ssi_ahb_ena=1时不可写

Bits	Name	R/W	Description
7:0	SPI_READ_CMD	R/W	When SPI READ MODE, the SPI READ CMD 000: 03H 001: 0BH 010: 3BH 011: 6BH 100: BBH 101: EBH 110: E7H Others: 03H Reset value: 03H
15:8	Flash M	R/W	When Dual I/O read, Quad I/O read, Quad Word Read, use flash M. So When reset, the CMD dual I/O read , Quad I/O read, Quad Word Read should not be used because flash M still have not been gotten。 Reset value:00H
18 : 16	DIRECT_SPI_READ_MODE	R/W	DIRECT SPI READ MODE 000: standard SPI read, 001: standard SPI fast read

			010: dual output fast read 011: quad output fast read 100: dual I/O fast read 101: quad I/O fast read 110: quad I/O word read Others: rsvd Reset value: 000
19	SPI_READ_PREFETCH	R/W	SPI READ AHB Prefetch Enable

21.2.20 PGM_CMD

Ssi_ahb_ena=1时不可写

Bits	Name	R/W	Description
7:0	SPI PAGE PROGRAM	R/W	Flash program command Reset value：02H
9:8	Flash Page Size	R/W	00: page size为256 bytes，即地址100整数倍 01: page size 为512 bytes，即地址100整数倍 10: page size 为1024 bytes，即地址100整数倍 11: page size为2048 bytes，即地址100整数倍 Reset value: 00,256 bytes 注，当外部flash器件page size大于2048 bytes时，可以配置成2048 bytes，不会对功能造成什么影响，仅仅是program性能略有降低。

21.2.21 CACHE_FLUSH

Bits	Name	R/W	Description
0	Flush	R	Read this register to flush cache Write has no effect

21.2.22 CACHE_DIS_UPDATE

Ssi_ahb_ena=1时不可写

Bits	Name	R/W	Description
0	Update_DISABLE	R/W	Set 1, cache will not be updated again Set 0, cache will be update automatically Reset value : 0

21.2.23 DR

RW，发送数据/接收数据

共40个地址入口，以方便于AHB burst操作。

21.2.24 TXFIFO_FLUSH

Ssi_ena=0时禁止读该寄存器

Bits	Name	R/W	Description
0	Flush	R	Read this register to flush txfifo Write has no effect

21.2.25 RXFIFO_FLUSH

Ssi_ena=0时禁止读该寄存器

Bits	Name	R/W	Description
0	Flush	R	Read this register to flush rxfifo Write has no effect

21.2.26 DMA_CTRL

Bits	Name	R/W	Description

1	TDMAE	R/W	Transmit DMA enable This bit enables/disables the transmit FIFO DMA 0 = transmit DMA disabled 1 = transmit DMA enabled Reset value : 0x0
0	RDMAE	R/W	Receive DMA enable This bit enables/disables the receive FIFO DMA 0 = receive DMA disabled 1 = receive DMA enabled Reset value : 0x0

21.2.27 DMA_TDLR

产生送给DMA engine的握手信号，必须往其内部写新的数据了，以满足spi的连续操作

Bits	Name	R/W	Description
4:0	DMA_TDLR	R/W	当TXFIFO达到这种情况时，产生dma_tx_req信号 0_0000, TXFIFO有0个为数据（fifo空）时，产生dma_tx_req 0_0001, TXFIFO有1个或以下为数据时，产生dma_tx_req 0_0010, TXFIFO有2个或以下为数据时，产生dma_tx_req …. Reset 0

21.2.28 DMA_RDLR

产生送给DMA engine的握手信号，表示rxfifo的数据个数已经超出了警戒线，必须马上读走。

Bits	Name	R/W	Description
3:0	RFT	R/W	当RXFIFO收满到这种情况时，产生dma_rx_req 0000, RXFIFO有1个或以上为数据时，产生dma_rx_req

			0001, RXFIFO有2个或以上为数据时，产生dma_rx_req 0010,RXFIFO有3个或以上为数据时，产生dma_rx_req …. Reset 0

22. DMA寄存器(0x4018_0000)

22.1 DMA 寄存器MAP

地址偏移量	名称	位宽	属性	说明
0x 000	SAR0	64	RW	Channel 0 source address register
0x008	DAR0	64	RW	Channel 0 destination address register
0x 010	LLP0	64	RW	Channel 0 linked list pointer register
0x018	CTL0	64	RW	Channel 0 control register
0x 020	SSTAT0	64	RW	Channel 0 source status register
0x028	DSTAT0	64	RW	Channel 0 destination status register
0x 030	SSTATAR0	64	RW	Channel 0 source status address register
0x038	DSTATAR0	64	RW	Channel 0 destination status address register
0x 040	CFG0	64	RW	Channel 0 configuration register
0x048	SGR0	64	RW	Channel 0 source gather register
0x 050	DSR0	64	RW	Channel 0 destination scatter register
0x058	SAR1	64	RW	Channel 1 source address register
0x 060	DAR1	64	RW	Channel 1 destination address regist

				er
0x068	LLP1	64	RW	Channel 1 linked list pointer register
0x 070	CTL1	64	RW	Channel 1 control register
0x078	SSTAT1	64	RW	Channel 1 source status register
0x 080	DSTAT1	64	RW	Channel 1 destination status register
0x088	SSTATAR1	64	RW	Channel 1 source status address reg ister
0x 090	DSTATAR1	64	RW	Channel 1 destination status address register
0x098	CFG1	64	RW	Channel 1 configuration register
0x 0A0	SGR1	64	RW	Channel 1 source gather register
0x0A8	DSR1	64	RW	Channel 1 destination scatter register
0x 0B0	SAR2	64	RW	Channel 2 source address register
0x0B8	DAR2	64	RW	Channel 2 destination address regist er
0x 0C0	LLP2	64	RW	Channel 2 linked list pointer register
0x0C8	CTL2	64	RW	Channel 2 control register
0x 0D0	SSTAT2	64	RW	Channel 2 source status register
0x0D8	DSTAT2	64	RW	Channel 2 destination status register
0x 0E0	SSTATAR2	64	RW	Channel 2 source status address reg ister
0x0E8	DSTATAR2	64	RW	Channel 2 destination status address register
0x 0F0	CFG2	64	RW	Channel 2 configuration register
0x0F8	SGR2	64	RW	Channel 2 source gather register
0x 100	DSR2	64	RW	Channel 2 destination scatter register
0x108	SAR3	64	RW	Channel 3 source address register
0x 110	DAR3	64	RW	Channel 3 destination address regist er
0x118	LLP3	64	RW	Channel 3 linked list pointer register
0x 120	CTL3	64	RW	Channel 3 control register
0x128	SSTAT3	64	RW	Channel 3 source status register
0x 130	DSTAT3	64	RW	Channel 3 destination status register

0x138	SSTATAR3	64	RW	Channel 3 source status address register
0x 140	DSTATAR3	64	RW	Channel 3 destination status address register
0x148	CFG3	64	RW	Channel 3 configuration register
0x 150	SGR3	64	RW	Channel 3 source gather register
0x158	DSR3	64	RW	Channel 3 destination scatter register
0x 2C0	RawTfr	64	R	Raw status for intTfr interrupt
0x2C8	RawBlock	64	R	Raw status for intBlock interrupt
0x 2D0	RawSrcTran	64	R	Raw status for intSrcTran interrupt
0x2D8	RawDstTran	64	R	Raw status for intDstTran interrupt
0x 2E0	RawErr	64	R	Raw status for intErr intterrupt
0x2E8	StatusTfr	64	R	Status for intTfr interrupt
0x 2F0	StatusBlock	64	R	Status for intBlock interrupt
0x2F8	StatusSrcTran	64	R	Status for intSrcTran interrupt
0x 300	StatusDstTran	64	R	Status for intDstTran interrupt
0x308	StatusErr	64	R	Status for intErr intterrupt
0x 310	MaskTft	64	RW	Mask for intTfr interrupt
0x318	MaskBlock	64	RW	Mask for intBlock interrupt
0x 320	MaskSrcTran	64	RW	Mask for intSrcTran interrupt
0x328	MaskDstTran	64	RW	Mask for intDstTran interrupt
0x 330	MaskErr	64	RW	Mask for intErr intterrupt
0x338	ClearTfr	64	W	Clear for intTfr interrupt
0x 340	ClearBlock	64	W	Clear for intBlock interrupt
0x348	ClearSrcTran	64	W	Clear for intSrcTran interrupt
0x 350	ClearDstTran	64	W	Clear for intDstTran interrupt
0x358	ClearErr	64	W	Clear for intErr intterrupt
0x 360	StatusInt	64	W	Status for each interrupt type
0x368	ReqSrcReg	64	RW	Source software transaction request register
0x 370	ReqDstReg	64	RW	Destionation software transaction request register
0x378	SglReqSrcReg	64	RW	Single source transcation request reg

				ister
0x 380	SglReqDstReg	64	RW	Single destination transaction request register
0x388	LstSrcReg	64	RW	Last source transaction request register
0x 390	LstDstReg	64	RW	Last destination transaction request register
0x398	DmaCfgReg	64	RW	DMA channel configuration register
0x 3A0	ChEnReg	64	RW	DMA channel enable register
0x3A8	DmaIdReg	64	R	DMA ID register
0x 3B0	DmaTestReg	64	RW	DMA Test register
0x3C8	DMA_COMP_PARAMS_6	64	R	Test only
0x 3D0	DMA_COMP_PARAMS_5	64	R	Test only
0x3D8	DMA_COMP_PARAMS_4	64	R	Test only
0x 3E0	DMA_COMP_PARAMS_3	64	R	Test only
0x3E8	DMA_COMP_PARAMS_2	64	R	Test only
0x 3F0	DMA_COMP_PARAMS_1	64	R	Test only
0x3F8	DAM Component ID Register	64	R	0x3231372a44571110

22.2Configuration and channel enable registers

22.2.1 DmaCfgReg

Bits	Name	R/W	Reset	Description
63:1	Undefined	N/A	0x0	Reserved

0	DMA_EN	R/W	0x0	**DW_ahb_dmac Enable bit.** 0 = DW_ahb_dmac Disabled 1 = DW_ahb_dmac Enabled.

22.2.2 ChEnReg

Bits	Name	R/W	Reset	Description
63:12	Undefined	N/A	0x0	Reserved
11:8	CH_EN_WE	W	0x0	Channel enable write enable.
7:4	Undefined	N/A	0x0	Reserved
3:0	CH_EN	R/W	0x0	Enables/Disables the channel. Setting this bit enables a channel; clearing this bit disables the channel. 0 = Disable the Channel 1 = Enable the Channel The ChEnReg.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.

22.3Channel Registers

22.3.1 SARx

Bits	Name	R/W	Reset	Description

63:32	Undefined	N/A	0x0	Reserved
31:0	*SAR*	*R/W*	0x0	**Current Source Address of DMA transfer**. Updated after each source transfer. The SINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every source transfer throughout the block transfer.

22.3.2 DARx

Bits	Name	R/W	Reset	Description
63:32	Undefined fined	N/A	0x0	Reserved
31:0	DAR	*R/W*	0x0	**Current Destination address of DMA transfer**. Updated after each destination transfer. The DINC field in the CTLx register determines whether the address increments, decrements, or is left unchanged on every destination transfer throughout the block transfer

22.3.3 LLPx

Bits	Name	R/W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:2	LOC	*R/W*	0x0	**Starting Address In Memory** of next LLI if block chaining is enabled. Note that the two LSBs of the starting address are not stored because the address is assumed to be aligned to a 32-bit boundary. LLI accesses are always 32-bit accesses (Hsize = 2) aligned

				to 32-bit boundaries and cannot be changed or programmed to anything other than 32- bit.
1:0	LMS	R/W	0x0	**Reserved**

22.3.4 CTLx

Bits	Name	R/W	Description
63:45	Undefined	N/A	Reserved
44	DONE	*R/W*	Done bit If status write-back is enabled, the upper word of the control register, CTL*X*[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set. Software can poll the LLI CTL*X*.DONE bit to see when a block transfer is complete. The LLI CTL*X*.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel. LLI accesses are always 32-bit accesses (Hsize = 2) aligned to 32-bit boundaries and cannot be changed or programmed to anything other than 32-bit. Reset Value: 0x0
b:32 (See description)	BLOCK_TS	R/W	**Block Transfer Size**. When the DW_ahb_dmac is the flow controller, the user writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat. **Width:** The width of the single transaction is determined by CTL*X*.SRC_TR_WIDTH. For further information on setting this field,Once the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of what is the flow controller. When the source or destination peripheral is assigned as the flow controller, then the maximum block size that can be read back saturates at 127, but the actual block size can be greater. **Reset Value:** 0x2
31:29	Undefined	N/A	Reserved
28	LLP_SRC	R/W	Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLP*x*.LOC is non-zero; Reset Value: 0x0

	_EN		
27	LLP_DST_EN	R/W	Block chaining is enabled on the destination side only if the LLP_DST_EN field is high and LLP*x*.LOC is non-zero. Reset Value: 0x0
26:25	SMS	R/W	**reserved**
24:23	DMS	R/W	**reserved**
22:20	TT_FC	R/W	**Transfer Type and Flow Control**. The following transfer types are supported. • Memory to Memory • Memory to Peripheral • Peripheral to Memory • Peripheral to Peripheral Flow Control can be assigned to the DW_ahb_dmac, the source peripheral, or the destination peripheral. **Reset Value: 0x3**
19	Undefined	N/A	Reserved
18	DST_SCATTER_EN	R/W	Destination scatter enable bit: 0 = Scatter disabled 1 = Scatter enabled Scatter on the destination side is applicable only when the CTL*x*.DINC bit indicates an incrementing or decrementing address control. Reset Value: 0x0
17	SRC_GATHER_EN	R/W	Source gather enable bit: 0 = Gather disabled 1 = Gather enabled Gather on the source side is applicable only when the CTL*x*.SINC bit indicates an incrementing or decrementing address control. Reset Value: 0x0
16:14	SRC_MSIZE	R/W	**Source Burst Transaction Length**. Number of data items, each of width CTLx.SRC_TR_WIDTH, to be read from the source every time a source burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Reset Value:** 0x1
13:11	DEST_MSIZE	R/W	**Destination Burst Transaction Length**. Number of data items, each of width CTLx.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. **NOTE:** This value is not related to the AHB bus master HBURST bus. **Reset Value:** 0x1
10:9	SINC	R/W	**Source Address Increment**. Indicates whether to increment or dec

			rement the source address on every source transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to "No change." 00 = Increment 01 = Decrement 1x = No change **NOTE**: Incrementing or decrementing is done for alignment to the next CTLx.SRC_TR_WIDTH boundary. **Reset Value:** 0x0
8:7	DINC	R/W	**Destination Address Increment**. Indicates whether to increment or decrement the destination address on every destination transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to "No change." 00 = Increment 01 = Decrement 1x = No change **NOTE:** Incrementing or decrementing is done for alignment to the next CTLx.DST_TR_WIDTH boundary. **Reset Value:** 0x0
6:4	SRC_TR_WIDTH	R/W	Hardcoded to 32(WORD)
3:1	DST_TR_WIDTH	R/W	**Hardcoded to 32 (WORD)**
0	INT_EN	R/W	**Interrupt Enable Bit**. If set, then all interrupt-generating sources are enabled. Functions as a global mask bit for all interrupts for the channel; raw* interrupt registers still assert if CTLx.INT_EN = 0. **Reset Value:** 0x1

22.3.5 SSTATx

Bits	Name	R / W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:0	SSTAT	*R/W*	0x0	Source status information retrieved by hardware from the address pointed to by the contents of the SSTATARx register.

22.3.6 DSTATx

Bits	Name	R / W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:0	DSTAT	*R/W*	0x0	which is registered in the DSTATx register and written out to the DSTAT*x* register location of the LLI before the start of the next block.

22.3.7 SSTATARx

Bits	Name	R / W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31：1:0	SSTATAR	*R/W*	0x0	Pointer from where hardware can fetch the source information, which is registered in the SSTATx register and written out to the SSTATx register location of the LLIbefore the start of the next block.

22.3.8 DSTATARx

Bits	Name	R / W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31：1:0	DSTATAR	*R/W*	0x0	Pointer from where hardware can fetch the destination status information,which is registered in the DSTATx register and written out to the DSTAT*x* register location of the LLI before the start of the next block.

22.3.9 CFGx

Bits	Name	R/W	Reset	Description
63:47	Undefined	N/A	0x0	Reserved
46:43 (see notes)	DEST_PER	*R/W*	0x0	Assigns a hardware handshaking interface (0 - DMAH_NUM_HS_INT-1) to the destination of channel *x* if the CFG*x*.HS_SEL_DST field is 0; otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. NOTE1: For correct DMA operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.
42:39 (see notes)	SRC_PER	R/W	0x0	Assigns a hardware handshaking interface (0 - DMAH_NUM_HS_INT-1) to the source of channel *x* if the CFG*x*.HS_SEL_SRC field is 0; otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. NOTE1: For correct DW_ahb_dmac operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.
38	SS_UPD_EN	R/W	0x0	**Source Status Update Enable**. Source status information is fetched only from the location pointed to by the SSTATARx register, stored in the SSTATx register and written out to the SSTAT*x* location of the LLI if *SS_UPD_EN* is high.
37	DS_UPD_EN	R/W	0x0	**Destination Status Update Enable**. Destination status information is fetched only from the location pointed to by the DSTATARx register, stored in the DSTATx register and written out to the DSTAT*x* location of the LLI if DS_UPD_EN is high.
36:34	PROTCTL	R/W	0x1	**Protection Control** bits used to drive the AHB HPROT[3:1] bus. The *AMBA Specification* recommends that the default value of HPROT indicates a non-cached, non-buffered, privileged data access. The reset value is used to indicate such an access. HPROT[0] is tied high because all transfers are data accesses, as there are no opcode fetches. There is a one-to-one mapping of these register bits to the HPROT[3:1] master interface signals.
33	FIFO_MODE	R/W	0x0	**FIFO Mode Select**. Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. 0 = Space/data available for single AHB transfer of the s

				pecified transfer width. 1 = Data available is greater than or equal to half the FIFO depth for destination transfers and space available is greater than half the fifo depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer
32	FCMODE	R/W	0x0	**Flow Control Mode**. Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. 0 = Source transaction requests are serviced when they occur. Data pre-fetching is enabled. 1 = Source transaction requests are not serviced until a destination transaction request occurs. In this mode, the amount of data transferred from the source is limited so that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled.
31	RELOAD_DST	R/W	0x0	**Automatic Destination Reload**. The DARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. For conditions under which this occurs.
30	RELOAD_SRC	R/W	0x0	**Automatic Source Reload**. The SARx register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated. For conditions under which this occurs.
29:20				reserved
19	SRC_HS_PO	R/W	0x0	**Source Handshaking Interface Polarity**. 0 = Active high 1 = Active low For information on this.
18	DST_HS_POL	R/W	0x0	**Destination Handshaking Interface Polarity**. 0 = Active high 1 = Active low For information on this.
17:12	LOCK_B_L	R/W	0x0	reserved
11	HS_SEL	R/W	0x1	**Source Software or Hardware Handshaking Select**. This register selects which of the handshakin

	_SRC			g interfaces – hardware or software – is active for source requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware-initiated transaction requests are ignored. If the source peripheral is memory, then this bit is ignored.
10	HS_SEL_DST	R/W	0x1	**Destination Software or Hardware Handshaking Select**. This register selects which of the handshaking interfaces – hardware or software – is active for destination requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware- initiated transaction requests are ignored. If the destination peripheral is memory, then this bit is ignored.
9	FIFO_EMPTY	R/	0x1	Indicates if there is data left in the channel FIFO. Can be used in conjunction with CFG*x*.CH_SUSP to cleanly disable a channel. For more information. 1 = Channel FIFO empty 0 = Channel FIFO not empty
8	CH_SUSP	R/W	0x1	**Channel Suspend**. Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also be used in conjunction with CFG*x*.FIFO_EMPTY to cleanly disable a channel without losing any data. 0 = Not suspended. 1 = Suspend DMA transfer from the source.
7:5	CH_PRIOR	R/W	Chan0=0 Chan1=1 Chan2=2 Chan3=3	**Channel priority**. A priority of 7 is the highest priority, and 0 is the lowest. This field must be programmed within the following range: 0: (DMAH_NUM_CHANNELS – 1) A programmed value outside this range will cause erroneous behavior.
4:0	Undefined	N/A	0x0	Reserved

22.3.10 SGRx

Bits	Name	R/W	Reset	Description
63:27	Undefined	N/A	0x0	Reserved
26:20 S	SGC	*R/W*	0x0	Source gather count. Source contiguous transfer count bet

ee description				ween successive gather boundaries. b = log2 (DMAH_CHx_MAX_BLK_SIZE + 1) + 19
19:0	SGI	*R/W*	0x0	Source gather interval.

22.3.11 DSRx

Bits	Name	R/W	Reset	Description
63:27	Undefined	N/A	0x0	Reserved
26:20 See description	DSC	*R/W*	0x0	Destination scatter count. Destination contiguous transfer count between successive scatter boundaries. b = log2 (DMAH_CHx_MAX_BLK_SIZE + 1) + 19
19:0	*DSI*	*R/W*	0x0	Destination scatter interval.

22.4Interrupt Registers

22.4.1 RawBlock,RawDstTran,RawErr, RawSrcTran, RawTfr

Bits	Name	R/W	Reset	Description
63:DMAH_NUM_CHANNELS	Undefined	N/A	0x0	Reserved
DMAH_NUM_CHANNELS–1:0	RAW	*R/W*	0x0	Raw interrupt status.

22.4.2 StatusBlock, StatusDstTran, StatusErr, StatusSrcTran, StatusTfr

Bits	Name	R/W	Reset	Description
63:4	Undefined	N/A	0x0	Reserved
3:0	STATUS	R	0x0	Interrupt status.

22.4.2 MaskBlock, MaskDstTran, MaskErr, MaskSrcTran, MaskTfr

Bits	Name	R/W	Description
63:12	Undefined	N/A	Reserved dnc = DMAH_NUM_CHANNELS Reset Value: 0x0
11:8	INT_MASK_WE	W	Interrupt Mask Write Enable 0 = write disabled 1 = write enabled dnc = DMAH_NUM_CHANNELS Reset Value: 0x0
7:4	Undefined	N/A	Reserved dnc = DMAH_NUM_CHANNELS If dnc = 8, then this field is not present. Reset Value: 0x0
3:0	INT_MASK	R/W	Interrupt Mask 0 = masked 1 = unmasked dnc = DMAH_NUM_CHANNELS Reset Value: 0x0

22.4.4 ClearBlock, ClearDstTran, ClearErr, ClearSrcTran, ClearTfr

Bits	Name	R/W	Reset	Description
63:4	Undefined	N/A	0x0	Reserved
3:0	CLEAR	W	0x0	Interrupt clear. 0 = no effect 1 = clear interrupt

22.4.5 StatusInt

Bits	Name	R/W	Reset	Description
63:5	Undefined	N/A	0x0	Reserved
4	ERR	R	0x0	OR of the contents of StatusErr register
3	DSTT	R	0x0	OR of the contents of StatusDst register.
2	SRCT	R	0x0	OR of the contents of StatusSrcTran register
1	BLOCK	R	0x0	OR of the contents of StatusBlock register
0	TFR	R	0x0	OR of the contents of StatusTfr register.

22.5Software Handshaking Registers

22.5.1 ReqSrcReg

Bits	Name	R/W	Reset	Description
63:12	Undefined	N/A	0x0	Reserved
11:8	SRC_REQ_WE	W	0x0	Source request write enable 0 = write disabled 1 = write enabled
7:4	Undefined	N/A	0x0	Reserved
3:0	SRC_REQ	R/W	0x0	Source request

22.5.2 ReqDstReg

Bits	Name	R/W	Reset	Description
63:12	Undefined	N/A	0x0	Reserved
11:8	DST_REQ_WE	W	0x0	Destination request write enable 0 = write disabled 1 = write enabled

7:4	Undefined	N/A	0x0	Reserved
3:0	DST_REQ	R/W	0x0	Destination request

22.5.3 SglReqSrcReg

Bits	Name	R/W	Reset	Description
63:12	Undefined	N/A	0x0	Reserved
11:8	SRC_SGLREQ_WE	W	0x0	Single write enable 0 = write disabled 1 = write enabled
7:4	Undefined	N/A	0x0	Reserved
3:0	SRC_SGLREQ	R/W	0x0	Destination single or burst request

22.5.4 LstSrcReg

Bits	Name	R/W	Reset	Description
63:12	Undefined	N/A	0x0	Reserved
11:8	LSTSRC_WE	W	0x0	Source last transaction request write enable 0 = write disabled 1 = write enabled
7:4	Undefined	N/A	0x0	Reserved
3:0	LSTSRC	R/W	0x0	Source last transaction request 0 = Not last transaction in current block 1 = Last transaction in current block

22.5.5 LstDstReg

Bits	Name	R/W	Reset	Description

63:12	Undefined	N/A	0x0	Reserved
11:8	LSTSRC_WE	W	0x0	Destination last transaction request write enable 0 = write disabled 1 = write enabled
7:4	Undefined	N/A	0x0	Reserved
3:0	LSTSRC	R/W	0x0	Destination last transaction request 0 = Not last transaction in current block 1 = Last transaction in current block

22.6Miscellaneous DW_ahb_dmac registers

22.6.1 DmaIdReg

Bits	Name	R/W	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:0	DMA_ID	R	DMAH_ID_NUM	Hardcoded DW_ahb_dmac Peripheral ID

22.6.2 DmaTestReg

Bits	Name	R/W	Reset	Description
63:1	Undefined	N/A	0x0	Reserved
0	TEST_SLV_IF	R/W	0x0	Puts the AHB slave interface into test mode. In this mode, the readback value of the writable registers always matches the value written. This bit does not allow writing to read-only registers. 0 = Normal mode 1 = Test mode

23. 版本历史

版本	日期	作者	描述
1.0	2013-5-24	周朝显	新版
1.1	2014-2-11	周朝显	增加封装管脚复用示意图，并修改管脚描述 增加供电电源描述

24. 联系信息

北京新岸线移动多媒体技术有限公司

地址：中国北京市海淀区中关村东路1号清华科技园科技大厦A、D座16、17层，B座6层，100084

电话：（+8610）82150688-8670

传真：（+8610）82150699

Web：www.nufront.com

E-mail: NL6621@nufront.com