目 录

插图目录

表格目录

1 芯片简介

1.1 主要特性

1.1.1 基本特性

1.1.1.1.1 计算能力特性

Hi3660 计算能力核心规格如下：

- ARM Cortex-A73 MPCore 高性能重核（4 核）（频率 2.4 GHz）和 ARM Cortex-A53 MPCore 高能效比轻核（4 核）处理器（频率 1.8 GHz）的 8 核 CPU。
- 支持 OPENGL ES3.2、OPENCL 1.2、OPENCL 2.0、DirectX 11、Renderscrip 高性能 3D 加速技术
- 支持全芯片通路的 MMU 管理，减少预留内存开销
- 支持 1866MHz 四通道 LPDDR4，支持 DDR 最高 8GB 空间

1.1.1.1.2 多媒体特性

Hi3660 多媒体核心规格如下：

- 内置视频硬件解码器（H.265/H.264 4096%2160@60fps）
- 内置视频硬件编码器（H.265/H.264 4096%2160@30fps）
- 内置独立 GPU，Mali G71 MP8@1 GHz
- 支持 960MPixel/s 的吞吐率，最大支持 23M pixel@30fps，2 个 pipe 独立可以支持 16Mpixel/pipe @30fps
- 支持独立的 JPEG 编码器和人脸检测加速模块

- 支持最大 3840%2400@60Hz 双 MIPI DSI 接口，支持多种 IFBC
- 通过 MIPI DSI 和 SPDIF 外扩 external 显示接口
- 通过外扩芯片支持 TypeC 接口

1.1.2 多媒体特性

Hi3660 多媒体特性如表 1-1 所示。

表1-1 Hi3660 多媒体特性

媒体特性	特性描述
GPU	
3D 加速	■支持 OpenGL 3.2/open VG1.1 ■支持 OpenGL ES 1.1/2.0/OpenGL ES 3.0/ OpenGL ES 3.1/ OpenGL ES 3.2 ■支持 OpenCL 1.1/1.2/2.0 ■DirectX 11.1 Specification ■Renderscript
显示像素深度	RGB888，RGB565
LCD 显示分辨率	最高至 3840%2400，刷新率 60Hz
LCD 接口	支持两路 MIPI-DSI LCD 接口，通过双 MIPI 支持 1/3 或者 1/2 接口压缩支持 3840%2400 显示
TV 接口	■通过 MIPI DSI 和 SPDIF 外扩 external 显示接口 ■通过外扩芯片支持 TypeC 接口
WFD	支持 WiFi display 1080P@60fps
视频编码	编码格式：H.265/H264 高清拍摄： 3840x2400@30fps 4 路高清同时编码：4 x 1080p@30fps 支持视频 720p 240 帧，快录慢放
视频解码	■支持 H.265、HEVC MP/High Tier、Main 10/High Tier、H.264 BP/MP/HP、MPEG1/2/4、VC-1、VP6/8、RV8/9/10， SVC、DIVX、MVC ■最高到 H.265 4K@60fps 和 H2644K@30fps
音频接口	支持 Slimbus（Master/Slave 模式）、I2S（Master/Slave 模式），PCM（Master/Slave 模式）、DSD 接口
音频采样率	8kHz、16kHz、32kHz、48kHz、96kHz
ADC/DAC	模拟 Codec 支持 4 路数字 DAC 通道，其中 2 路支持超声波发射，另外 2 路支持高清播放，模拟支持 5 路 DAC 独立通道； ADC SNR 100dB (A-Weighted），THD -80dB，48kHz 采样率

媒体特性	特性描述
数据精度	支持 24 bits 数据精度，音频通路支持 SNR
音频 DSP 处理器	Hi3660 侧和 Hi6403 侧均使用 HiFI3 DSP 做独立 DSP 处理 最高频率 533Mhz
音频	支持 MP3 播放
MIC 输入	支持 4 MIC 输入
音效	支持多种音效处理
录音	支持双声道录音
通信语音	支持 AVS、VOLTE 等高性能语音特性

1.1.3 接口特性

Hi3660 外设接口特性如表 1-2 所示。

表1-2 Hi3660 外设接口特性

接口名称	接口描述
USB 接口	1、支持 USB3.0 OTG 协议、USB2.0 OTG 协议 2、支持 BC1.2 充电
PCIE 接口	支持 PCIE1.1/PCIE2.0 协议
Micro SD 卡接口	1、支持 SD 卡 SD3.0/SD2.0 2、支持高速 SD 卡 3、支持热插拔
BT\WiFi 空口	1、WiFi：WLAN、支持便携式热点、WIFI-DIRECT，支持 IEEE802.11 b/g/n/ac 协议 2、蓝牙：BT 4.1/BT 4.0/BT Class1.5/ BT Profile（通话、A2DP、FTP）
FM 空口	1、支持耳机收听 2、支持 FM Speaker 收听 3、支持 FM 录音
移动电视	ISDB-T/ CMBB
人机接口	支持多个按键，若支持键盘矩阵，通过 I2C 扩展。支持触摸屏；支持多点触摸
MIC 接口	支持语音通话输入：Mic 输入、耳机 Mic 输入、蓝牙耳机 Mic 输入

接口名称	接口描述
I2C 接口	支持高速 I2C 7 组，最高支持 3M 速率
SPI 接口	支持 SPI 5 组 master，最高支持到 30M 速率
UART 接口	支持 9 组高速 UART，最高频率 9M Bauds
GPIO	Hi3660 和 Hi6403 均支持丰富的 GPIO 接口
耳机接口	1、支持 3.5mm 耳机输出 2、支持耳机 MIC 输入 3、支持 MIC 线控（4 键）
Speaker 接口	1、支持外扩 Stereo speaker 功放 2、支持两个 Lineout 输出接口
Receiver 接口	支持一个听筒
键盘	支持键盘背景灯

Hi3660 memory 接口特性如表 1-3 所示。

表1-3 Hi3660 memory 接口特性

接口名称	接口描述
eMMC Flash	支持 eMMC5.1，提供非易失存储区域用于存储启动代码、数据等信息 总线速率 eMMC 支持 HS400
UFS Flash	支持 UFS2.1，提供非易失存储区域用于存储启动代码、数据等信息，支持 Inlien 加密
LPDDR4 SDRAM	提供动态存储区域用于系统运行，支持四通道，每个通道支持最多两个片选 支持 366BALL LPDDR4：1866MHz 支持 HBPOP 封装形式，最大容量 8GB
Micro SD card	支持 SD3.0 协议 提供用户数据存放，用户软件安装等功能，作为主存储器用户空间扩展。

Hi3660 调试接口特性如表 1-4 所示。

表1-4 Hi3660 调试接口特性

接口名称	接口描述
Coresight 接口	支持 Coresight 调试接口，支持 SWD 调试模式
JTAG 接口	符合 *IEEE std 1149.1* 标准。芯片提供 JTAG 调试接口

1.1.4 低功耗特性

Hi3660 低功耗特性如表 1-5 所示。

表1-5 Hi3660 低功耗特性

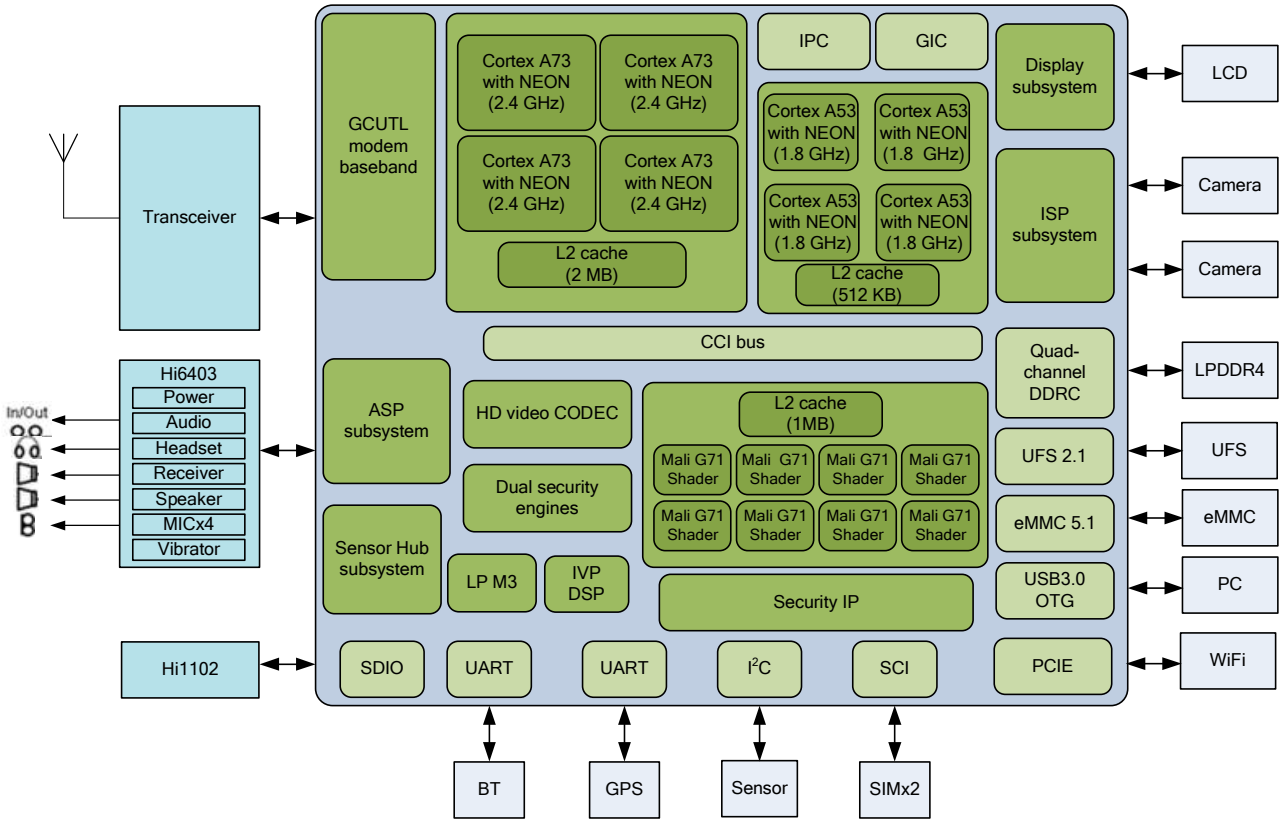
功耗控制	接口描述
DVFS	4 核 Cortex A73 重核和 4 核 Cortex A53 轻核，GPU 均支持独立的 DVFS 功能
Idle 态功耗管理	支持 idle 态超低功耗设计，维持最小系统低功耗

1.2 芯片架构

Hi3660 芯片采用 TSMC 16nm Finfet 工艺，具有多核、多模、高性能、高集成度的特点。单芯片承载高速移动计算能力，集成了丰富的多媒体处理功能和高规格的通信处理功能，并且运用业界领先的低功耗技术，是高端智能机市场的完整产品解决方案 Kirin960 的核心处理 SoC 芯片。

Hi3660 架构框图如图 1-1 所示。

图1-1 Hi3660 架构框图



Hi3660 芯片包含以下主要功能模块和子系统：

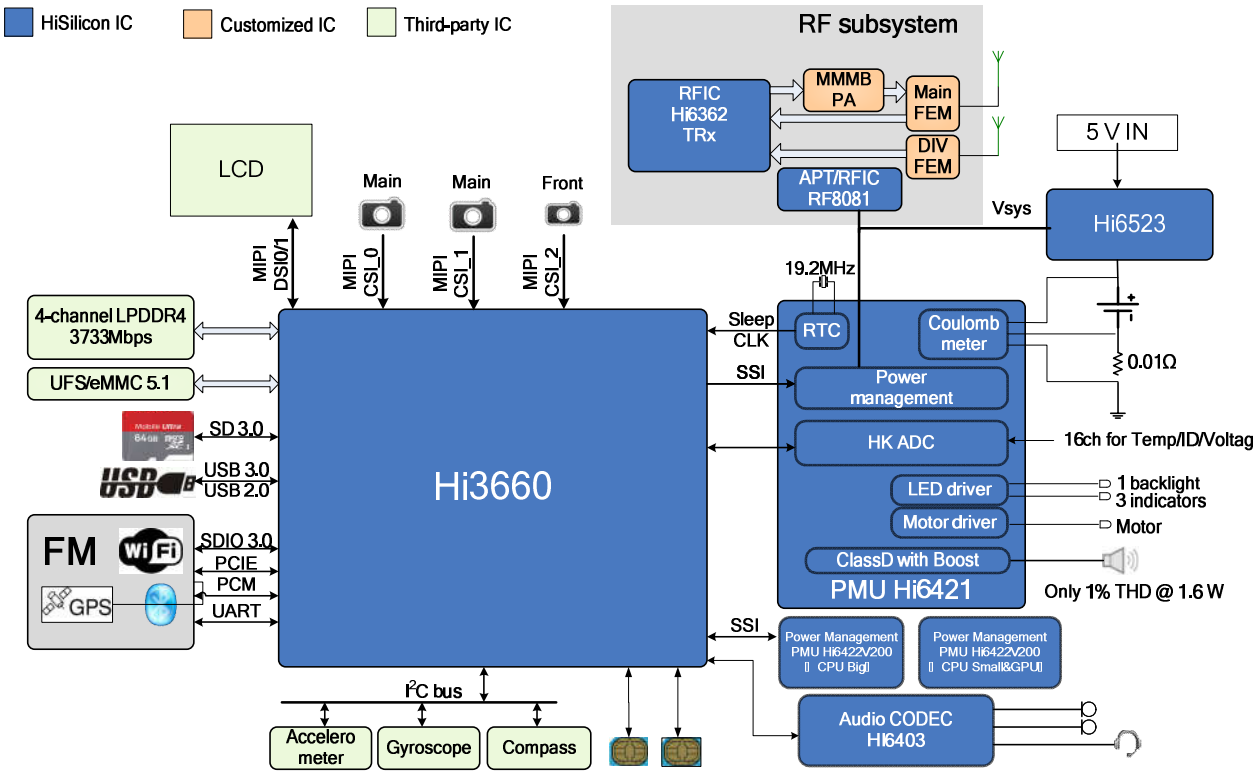
- ARM Cortex-A73 MPCore 高性能重核（4 核）和 ARM Cortex-A53 MPCore 高能效比轻核（4 核）处理器的 big.LITTLE 8 核 CPU 子系统
- ARM Mali G71 MP8 3D GPU
- 独立 ISP 处理器
- 3840%2400 高清视频硬件解码器和 3840%2400 高清视频硬件编码器
- 显示和图形加速子系统
- 1 x Tensilica HiFi3.0 DSP 的音频子系统
- DMA 控制器，PCIE 控制器，USB3.0 控制器，SD 卡控制器等输入输出设备控制器的外设子系统
- DDR 控制器，UFS 控制器
- Security IP 提供增强的加解密算法
- 双安全引擎和安全 SRAM 的安全子系统

1.3 典型应用

Hi3660 可支持的典型产品形态：智能手机、平板电脑等。

智能机应用场景如图 1-2 所示。

图1-2 Hi3660 典型应用（智能机）



2 整体描述

2.1 芯片结构

2.1.1 SoC 系统

表2-1 系统结构模块说明

模块缩写	模块说明	模块缩写	模块说明
A73/A53	应用处理器 Cluster，big.LITTLE 架构	IVP	图像视频处理模块
ASP	音频处理子系统	LPMCU	低功耗处理子系统
BLPWM	背光脉宽调制模块	MMC	多媒体卡控制模块
BOOTROM	片上 ROM	NANDC	FLASH 控制器
CODEC_SSI	与 CODEC 通信的 SSI 接口模块	PCTRL	外设控制器
CRG	时钟复位模块	PMCTRL	DFS/DVFS 等功耗控制模块
CSSYS	处理器联合调试模块	PMU_I2C	与 PMU 通信的 I2C 接口模块
DDRC	DDR 控制器	PMU_SSI	与 PMU 通信的 SSI 接口模块
DJTAG	JTAG 口调测模块	PWM	脉宽调制模块
DMAC	DMA 控制器模块	RTC	实时时钟计数器
DSS	显示模块	SCI	SIM 卡控制器
EFUSEC	EFUSE 控制模块	SCTRL	系统控制器

模块缩写	模块说明	模块缩写	模块说明
GIC	处理器中断处理模块	SEC_P/SEC_S	安全处理模块
GNSPWM	通用 PWM 模块	SPI	SPI 控制器
GPIO	通用 I/O 接口模块	SYS_CNT	处理器专用计数器模块
GPU	媒体业务处理器	TIMER	计时、计数模块
HKADC_SSI	与外置 PMU 的 HKADC 的 SSI 接口转换模块	TSENSORC	TSENSOR 控制器
I2C	I2C 控制器	TZPC	安全信号分配模块
IOC	IO 控制模块	UART	通用串口模块
IOMCU	Sensor-Hub 处理子系统	USB3OTG	USB3.0 控制器模块
IPC	核间通信模块	VENC/VDEC	视频加解密处理模块
ISP	图像处理模块	WD	看门狗计数器
EMMC5.1	EMMC5.1 控制器模块	UFS	Unified File system 控制器模块
SDIO	SDIO 控制器模块	SD	SD 卡控制器模块
PCIe	PCIe 控制器模块		

SoC 系统采用了多层总线架构，每层支持独立并行访问，集成了芯片外设、接口外设和其他内置功能模块。

SoC 系统总线架构具有以下特性：

- 通过一致性总线互连，支持硬件一致性。
- 采用独立多层总线互连，提升芯片总线带宽，提供良好的可扩展性。
- 高速数据总线是 AXI，位宽 128bit 或者 64bit。
- 低速数据总线及配置总线是 32bitAXI。

2.1.2 媒体子系统

媒体子系统包括图像采集、图像显示输出、图像编解码加速、3D 图形加速、音频处理加速等，提供强大的多媒体处理加速功能。

媒体子系统支持多项高层应用，比如：数码拍照、数码摄像、录音、播放本地音/视频、浏览本地图片、播放流媒体音/视频、多媒体编辑器、视频电话、UI 和视频硬件加速、游戏硬件加速等。

说明

关于媒体子系统的详细描述，请参见“6 媒体处理”。

2.1.3 存储子系统

Hi3660 支持多种形式的外部存储接口，为系统提供灵活的存储器方案。支持 UFS/EMMC/SD/SDIO 静态存储卡接口，LPDDR4 动态 Memory 接口，以及 NAND FLASH 接口，能够满足不同产品形态的需求。

说明

关于存储子系统的详细描述，请参见“7 存储控制”。

2.2 时钟

2.2.1 功能描述

Hi3660 芯片接收外部时钟输入，利用内部锁相环（PLL：Phase-Locked Loop）及时钟电路产生芯片内部需要的各种工作时钟，并可为其他芯片提供多路时钟。

Hi3660 芯片内部提供 11 个 PLL，用于产生芯片各模块所需各种工作时钟。

2.2.2 时钟输入

Hi3660 芯片外部输入时钟是 32kHz 时钟、19.2MHz 时钟和外部备份时钟（100MHz）。

2.3 复位

2.3.1 功能描述

Hi3660 芯片接收外部提供的复位输入，在上电开机过程中按照解复位序列完成芯片内部各模块的上电复位、解复位操作。

芯片内置 POR （Power on reset）模块，在 AO 区上电之后，输出低电平信号，复位芯片，并在 12~48ms 左右拉高输出电平。可以保证在 IO （外部输入复位）处于不定态时整个芯片处于复位态。

除上电复位之外，芯片还支持多种全局复位：Watchdog 复位、温度传感器复位和芯片软复位。以上任何一种复位有效，都会引起芯片的全局复位。各复位之间没有优先级之分。

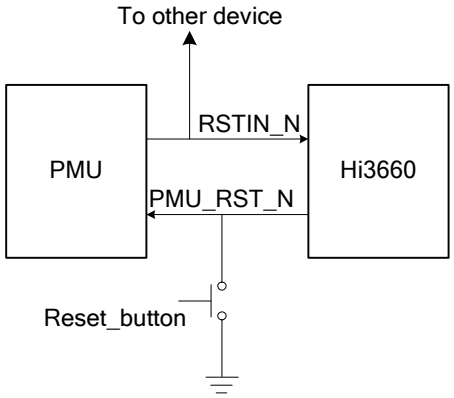
2.3.2 套片复位方案

Hi3660 上电过程由 PMU 提供复位输入（RSTIN_N），在开机过程中自行完成内部各模块的上电复位操作。

当 Hi3660 内部任何一种全局复位有效时，芯片输出 PMU 复位信号（PMU_RST_N），用于复位 PMU，PMU 复位之后，会拉低 RSTIN_N，从而实现芯片的全局复位。

外部复位关系图如图 2-1 所示。

图2-1 外部复位关系图



2.3.3 复位结构

Hi3660 芯片支持多种全局复位：上电复位、Watchdog 复位、温度传感器复位和芯片软复位。

复位模式分为临终遗言复位和非临终遗言复位两种方式：配置 SCPEREN1[31]为 1 可使能临终遗言复位模式；配置 SCPERDIS1[31]为 1，可关闭临终遗言复位模式；

- 临终遗言复位：wd 复位请求、过温复位请求有效时，首先发出 ddr 进入自出刷新请求，等待 DDR 进入自刷新模式（软件收到中断后保护现场，保存当前关键系统信息到外部非易失存储器或者 SDRAM，然后将 SDRAM 置为自刷新模式）或者超时 1ms 后，发出复位请求信号，拉低 PMU 复位信号，复位整个系统；
- 非临终遗言流程：wd 复位请求、过温复位请求、软件复位请求有效时，直接发出复位请求信号，拉低 PMU 复位信号，复位整个系统，整个过程不保留现场。

2.3.3.1.1 上电复位

上电全局复位由两个复位信号相与而成：芯片外部 RSTIN_N 复位和 POR 输出复位。芯片内部 POR 在 AO 区上电之后，输出低电平，复位整个芯片，并在 12~48ms 之间解除复位；外部复位 RSTIN_N 信号由 PMU 产生，在 IO 上电之后，会保持芯片的复位状态，等待 PMU 解除复位之后，可以解除芯片全局复位。

2.3.3.1.2 Watchdog 复位

Watchdog 包括：wd0、wd1、lpmcu 子系统的 watchdog、iomcu 子系统的 watchdog、modem 子系统的 watchdog、asp 子系统的 watchdog、ivp 子系统的 watchdog、ispa7 子系统的 watchdog、uce 的 watchdog、ocbc 的 watchdog、gpu 的 watchdog、小核的 watchdog、大核的 watchdog。

Watchdog 模块用于监控系统的运行状态。正常情况下系统需要定时设置计数初值。如果没有及时设置计数初值，表明软件运行异常，Watchdog 将采取以下动作：

- 发出异常中断，同时载入计数器初值，重新开始计数。
- 如果上述异常中断没有被处理，则计数器计到零后发出复位信号。

2.3.3.1.3 温度传感器复位

芯片 A53、A73、G3D 区域均内置一个温度传感器，当芯片温度达到预设的门限时，会发出复位请求信号将芯片内部 A53、A73 复位。

2.3.3.1.4 芯片软复位

软件可以在必要时进行芯片软复位。软件写寄存器 SCSYSSTAT，会引起芯片的全局软复位。

注意

软复位只能在非临终遗言复位模式下使用，否则写寄存器 SCSYSSTAT，不会引起芯片的全局软复位。

2.3.3.1.5 模块软复位

Hi3660 芯片的各主要模块都可以由软件独立控制复位，如 RTC、Timer、GPIO、USB、DMAC、VENC、VDEC、DSS、ISP、ASP、DDRC、MMC、PWM、UART、SPI、G3D 等。具体内容请参见本文的各模块描述。

2.4 中断

2.4.1 功能描述

ACPU 使用 GIC 模块进行中断处理及中断控制，其他微控制器、媒体及通信处理器内部都有各自相应的中断处理逻辑。本节简述 GIC 的中断处理的基本功能，其他处理器的中断处理原理请参见“3 移动处理模块”。

GIC 具有如下基本特性：

- 支持中断嵌套；
- 管理多核中断分发；
- 支持安全扩展；
- 支持中断源状态可查询；
- 每个中断有唯一的中断号；
- 中断可配置为高电平或边沿沿触发；
- 每个中断可配置优先级；
- 可产生软件中断；

GIC 支持如下三种类型的中断：

- 软中断 SGI（Software Generated Interrupt）

支持 16 个软中断，中断号为 0~15，通过写寄存器实现。

- 私有外设中断 PPI（Private Peripheral Interrupt）

  对每个处理器，有 7 个 PPI 与其相对应。

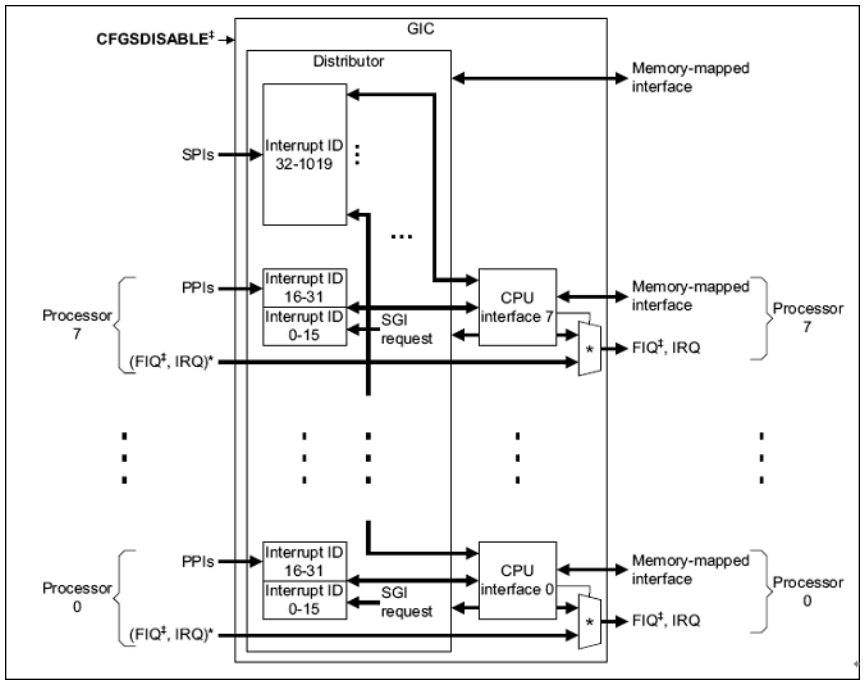
- 共享外设中断（Shared Peripheral Interrupt）

  支持 352 个外设中断。

2.4.2 中断结构

GIC 的内部结构如图 2-2 所示，主要分为 Distributor 和 CPU interface，其中 Distributor 主要完成所有中断的集中管理，CPU 接口主要实现中断和 CPU 的交互。一个 GIC 只有一个 Distributor，但一个 GIC 有多个 CPU 接口，Hi3660 集成的 GIC-400 有 8 个 CPU interface，分别连接 4 核 A73 和 4 核 A53。

图2-2 GIC 结构图



GIC 支持 TrustZone，每个中断源均可配置为安全中断源或者非安全中断源。

GIC 支持每个中断的优先级设置，中断优先级数值越小，优先级越高。当两个中断具有同级优先级，以中断 ID 号小者优先。

2.4.3 中断映射

Hi3660 的 GIC 共支持 384 个中断，其中外设中断 352 个，GIC 各中断源及其中断信号如表 2-2 所示。

表2-2 GIC 中断分配表

中断源	GIC 中断号	中断源	GIC 中断号
A73_interr	32	PMC-AVS-IDLE-G3D	205
A73_exterr	33	M3_LP_wd	206
A73_pmu0	34	~CCI400_err	207
A73_pmu1	35	~&CCI400_overflow[6:0]	208
A73_pmu2	36	~CCI400_overflow[7]	209
A73_pmu3	37	IPC_S_int0	210
A73_cti0	38	IPC_S_int1	211
A73_cti1	39	IPC_S_int4	212
A73_cti2	40	IPC_S_mbx0	213
A73_cti3	41	IPC_S_mbx1	214
A73_COMMRX0	42	IPC_S_mbx2	215
A73_COMMRX1	43	IPC_S_mbx3	216
A73_COMMRX2	44	IPC_S_mbx4	217
A73_COMMRX3	45	IPC_S_mbx5	218
A73_COMMTX0	46	IPC_S_mbx6	219
A73_COMMTX1	47	IPC_S_mbx7	220
A73_COMMTX2	48	IPC_S_mbx8	221
A73_COMMTX3	49	IPC_S_mbx9	222
A73_COMMIRQ0	50	IPC_S_mbx18	223
A73_COMMIRQ1	51	IPC_NS_int0	224
A73_COMMIRQ2	52	IPC_NS_int1	225
A73_COMMIRQ3	53	IPC_NS_int4	226
A53_interr	54	IPC_NS_int5	227
A53_exterr	55	IPC_NS_int6	228
A53_pmu0	56	IPC_NS_mbx0	229
A53_pmu1	57	IPC_NS_mbx1	230
A53_pmu2	58	IPC_NS_mbx2	231
A53_pmu3	59	IPC_NS_mbx3	232
A53_cti0	60	IPC_NS_mbx4	233
A53_cti1	61	IPC_NS_mbx5	234

中断源	GIC 中断号	中断源	GIC 中断号
A53_cti2	62	IPC_NS_mbx6	235
A53_cti3	63	IPC_NS_mbx7	236
A53_COMMRX0	64	IPC_NS_mbx8	237
A53_COMMRX1	65	IPC_NS_mbx9	238
A53_COMMRX2	66	IPC_NS_mbx18	239
A53_COMMRX3	67	mdm_aximon_intr	240
A53_COMMTX0	68	MDM_WDOG_intr	241
A53_COMMTX1	69	ASP-IPC-ARM	242
A53_COMMTX2	70	ASP-IPC-MCPU	243
A53_COMMTX3	71	ASP-IPC-BBE16	244
A53_COMMIRQ0	72	ASP_WD	245
A53_COMMIRQ1	73	ASP_AXI_DLOCK	246
A53_COMMIRQ2	74	ASP_DMA_SECURE	247
A53_COMMIRQ3	75	ASP_DMA_SECURE_N	248
WatchDog0	76	SCI0	249
WatchDog1	77	SCI1	250
RTC0	78	SOCP0	251
RTC1	79	SOCP1	252
TIME00	80	MDM_IPF_intr0	253
TIME01	81	MDM_IPF_intr1	254
TIME10	82	|ddrc_fatal_int[3:0]	255
TIME11	83	mdm_axi_dlock_int	256
TIME20	84	mdm_wdt1_intr(CDSP)	257
TIME21	85	~GIC_IRQ_OUT[0]	258
TIME30	86	~GIC_IRQ_OUT[1]	259
TIME31	87	~GIC_IRQ_OUT[2]	260
TIME40	88	~GIC_IRQ_OUT[3]	261
TIME41	89	~GIC_IRQ_OUT[4]	262
TIME50	90	~GIC_IRQ_OUT[5]	263
TIME51	91	~GIC_IRQ_OUT[6]	264
TIME60	92	~GIC_IRQ_OUT[7]	265

中断源	GIC 中断号	中断源	GIC 中断号
TIME61	93	~GIC_FIQ_OUT[0]	266
TIME70	94	~GIC_FIQ_OUT[1]	267
TIME71	95	~GIC_FIQ_OUT[2]	268
TIME80	96	~GIC_FIQ_OUT[3]	269
TIME81	97	~GIC_FIQ_OUT[4]	270
TIME90	98	~GIC_FIQ_OUT[5]	271
TIME91	99	~GIC_FIQ_OUT[6]	272
TIME100	100	~GIC_FIQ_OUT[7]	273
TIME101	101	NANDC	274
TIME110	102	CoreSight_ETR_Full	275
TIME111	103	CoreSight_ETF_Full	276
TIME120	104	DSS-pdp	277
TIME121	105	DSS-sdp	278
UART0	106	DSS-offline	279
UART1	107	DSS_mcu_pdp	280
UART2	108	DSS_mcu_sdp	281
UART4	109	DSS_mcu_offline	282
UART5	110	DSS_dsi0	283
UART6	111	DSS_dsi1	284
SPI1	112	IVP32_SMMU_irpt_s	285
I2C3	113	IVP32_SMMU_irpt_ns	286
I2C4	114	IVP32_WATCH_DOG	287
I2C5(PMU_I2C)	115	ATGC	288
GPIO0_INTR1	116	G3D_IRQEVENT	289
GPIO1_INTR1	117	G3D_JOB	290
GPIO2_INTR1	118	G3D_MMU	291
GPIO3_INTR1	119	G3D_GPU	292
GPIO4_INTR1	120	isp_irq[0]	293
GPIO5_INTR1	121	isp_irq[1]	294
GPIO6_INTR1	122	isp_irq[2]	295
GPIO7_INTR1	123	isp_irq[3]	296

中断源	GIC 中断号	中断源	GIC 中断号
GPIO8_INTR1	124	isp_irq[4]	297
GPIO9_INTR1	125	isp_irq[5]	298
GPIO10_INTR1	126	isp_irq[6]	299
GPIO11_INTR1	127	isp_irq[7]	300
GPIO12_INTR1	128	isp_a7_to_gic_mbx_int[0]	301
GPIO13_INTR1	129	isp_a7_to_gic_mbx_int[1]	302
GPIO14_INTR1	130	isp_a7_to_gic_ipc_int	303
GPIO15_INTR1	131	isp_a7_watchdog_int	304
GPIO16_INTR1	132	isp_axi_dlcok	305
GPIO17_INTR1	133	isp_a7_irq_out	306
GPIO18_INTR1	134	ivp32_dwaxi_dlock_irq	307
GPIO19_INTR1	135	mmbuf_asc0	308
GPIO20_INTR1	136	mmbuf_asc1	309
GPIO21_INTR1	137	UFS	310
GPIO22_INTR1	138	pcie_link_down_int	311
GPIO23_INTR1	139	pcie_edma_int	312
GPIO24_INTR1	140	pcie_pm_int	313
GPIO25_INTR1	141	pcie_radm_inta	314
GPIO26_INTR1	142	pcie_radm_intb	315
GPIO27_INTR1	143	pcie_radm_intc	316
IOMCU_WD	144	pcie_radm_intd	317
IOMCU_SPI	145	psam_intr[0]	318
IOMCU_UART3	146	psam_intr[1]	319
IOMCU_UART8	147	ocbc_pe_npint[0]	320
IOMCU_SPI2	148	intr_wdog_ocbc	321
IOMCU_I2C3	149	intr_vdec_mfde_norm	322
IOMCU_I2C0	150	intr_vdec_scd_norm	323
IOMCU_I2C1	151	intr_vdec_bpd_norm	324
IOMCU_I2C2	152	intr_vdec_mmu_norm	325
IOMCU_GPIO0_INT1	153	intr_vdec_mfde_safe	326
IOMCU_GPIO1_INT1	154	intr_vdec_scd_safe	327

中断源	GIC 中断号	中断源	GIC 中断号
IOMCU_GPIO2_INT1	155	intr_vdec_bpd_safe	328
IOMCU_GPIO3_INT1	156	intr_vdec_mmu_safe	329
IOMCU_DMAC_INT0	157	intr_venc_vedu_norm	330
IOMCU_DMAC_NS_INT0	158	intr_venc_mmu_norm	331
PERF_STAT	159	intr_venc_vedu_safe	332
IOMCU_COMB	160	intr_venc_mmu_safe	333
IOMCU_BLPWM	161	intr_qosbuf0	334
NOC-comb	162	intr_qosbuf1	335
intr_dmss	163	intr_ddrc2_err	336
intr_ddrc0_err	164	intr_ddrc3_err	337
intr_ddrc1_err	165	intr_ddrphy[0]	338
PMCTRL	166	intr_ddrphy[1]	339
SECENG_P	167	intr_ddrphy[2]	340
SECENG_S	168	intr_ddrphy[3]	341
EMMC51	169	intr0_mdm_ipc_gic_s	342
ASP_IPC_MODEM_CBBE	170	intr1_mdm_ipc_gic_s	343
SD3	171	SPI3	344
SDIO	172	SPI4(Finger/墨水屏)	345
GPIO28_INTR1	173	I2C7	346
PERI_DMAC_int0	174	intr_uce0_wdog	347
PERI_DMAC_NS_int0	175	intr_uce1_wdog	348
CLK_MONITOR(SCTRL 内)	176	intr_uce2_wdog	349
TSENSOR_A73	177	intr_uce3_wdog	350
TSENSOR_A53	178	intr_exmbist	351
TSENSOR_G3D	179	intr_hisee_wdog	352
TSENSOR_Modem	180	intr_hisee_ipc_mbx_gic[0]	353
ASP_ARM_SECURE (asp_hmdi 安全中断、src_up 安全中断)	181	intr_hisee_ipc_mbx_gic[1]	354

中断源	GIC 中断号	中断源	GIC 中断号
ASP_ARM (asp_hmdi 非安全中断、src_up 非安全中断、slimbus 的组合中断)	182	intr_hisee_ipc_mbx_gic[2]	355
VDM_INT2	183	intr_hisee_ipc_mbx_gic[3]	356
VDM_INT0	184	intr_hisee_ipc_mbx_gic[4]	357
VDM_INT1	185	intr_hisee_ipc_mbx_gic[5]	358
|{MODEM_IPC0[0],MDM_IPC_APPCPU_intr0}	186	intr_hisee_ipc_mbx_gic[6]	359
|{MODEM_IPC1[0],MDM_IPC_APPCPU_intr1}	187	intr_hisee_ipc_mbx_gic[7]	360
MDM_bus_err	188	intr_hisee_alarm[0]	361
Reserved	189	intr_hisee_alarm[1]	362
MDM_EDMAC0_INTR_NS[0]	190	Reserved	363
USB3	191	Reserved	364
Reserved	192	intr_hisee_eh2h_slv	365
USB3_OTG	193	intr_hisee_as2ap_irq	366
USB3_BC	194	intr_hisee_ds2ap_irq	367
GPIO1_SE_INTR1	195	intr_hisee_senc2ap_irq	368
GPIO0_SE_INTR1	196	GPIO0_EMMC	369
PMC-DVFS-A73	197	GPIO1_EMMC	370
PMC-DVFS-A53	198	AONOC_TIMEOUT	371
PMC-DVFS-G3D	199	intr_hisee_tsensor[0]	372
PMC-AVS-A73	200	intr_hisee_tsensor[1]	373
PMC-AVS-A53	201	intr_hisee_lockup	374
PMC-AVS-G3D	202	intr_hisee_dma	375
PMC-AVS-IDLE-A73	203	Reserved	376~383
PMC-AVS-IDLE-A53	204		

2.5 芯片工作模式与控制

Hi3660 芯片系统有以下 4 种工作模式，由寄存器 SCCTRL[modectrl]（LPMCU 访问：0xFFF0_A000；CPU 访问：0x4020_A000）控制。

- 000：表示系统切换到 SLEEP 状态；
- 001：表示系统切换到 DOZE 状态；
- 01X：表示系统切换到 SLOW 状态；
- 1XX：表示系统切换到 NORMAL 状态。

上电复位后，状态机默认处于 SLOW 模式。软件可通过配置寄存器 SCCTRL[modectrl]（LPMCU 访问：0xFFF0_A000；CPU 访问：0x4020_A000）控制系统状态迁移，具体的迁移流程详见系统模式切换章节。

全局复位（包括全局软复位、Watchdog 复位、Tsensor 过温复位）会导致系统状态机恢复到 SLOW 模式。

2.6 启动机制

2.6.1 整体流程

Hi3660 芯片支持 USB 加载和存储器启动两种方式，其中存储器启动中包含，eMMC 启动和 UFS 启动两种方式，以上方式都是通过芯片内部的 BOOTROM 引导，再进入对应的启动流程。

除去以上的常规启动方式外，芯片同样支持测试模式下的 NAND 启动 UFS 启动

2.6.1.1.1 管脚配置

BOOTROM 引导 UFS 启动的管脚配置如下：

- TEST_MODE：0
- BOOT_MODE：1
- BOOT_UFS：1

2.6.1.1.2 基本流程

步骤 1 开机，上电复位。

步骤 2 启动模式判断。

步骤 3 执行 BOOTROM 代码，首先读取 boot_mode 寄存器，判断启动模式。若判断到 BOOT_MODE 管脚值为 1，则进入存储器启动分支。进入存储器启动分支后，进而读取 boot_ufs 寄存器，判断到 BOOT_UFS 管脚值为 1，则进入 UFS 启动流程。

步骤 4 UFS 时钟及 IP 初始化。

步骤 5 UFS Link startup 流程。

步骤 6 UFS 参数及器件初始化。

步骤 7 UFS 启动镜像搬运，将 UFS device 里 boot lun 中的镜像 RAM 中。

步骤 8 安全校验。

步骤 9 开始执行 bootloader。

步骤 10 进行 DDR 初始化，并且将 UFS device 存储的 fastboot 镜像拷贝到 DDR 中，初始化 ACPU 并且解复位后，ACPU 侧开始执行 fastboot，进行后续启动流程。

----结束

2.6.2 eMMC 启动

2.6.2.1.1 管脚配置

BOOTROM 引导 eMMC 启动的管脚配置如下：

- TEST_MODE：0
- BOOT_MODE：1
- BOOT_UFS：0

2.6.2.1.2 基本流程

步骤 1 开机，上电复位。

步骤 2 启动模式判断。

执行 BOOTROM 代码，首先读取 boot_mode 寄存器，判断启动模式。若判断到 BOOT_MODE 管脚值为 1，则进入存储器启动分支。进入存储器启动分支后，进而读取 boot_ufs 寄存器，判断到 BOOT_UFS 管脚值为 0，则进入 eMMC 启动流程。

步骤 3 eMMC 时钟及 IP 初始化

步骤 4 eMMC 启动镜像搬运，将 eMMC device 里的镜像搬运到 eMMC 内部的 buffer 中，再将数据搬运到 RAM 中。

步骤 5 安全校验。

步骤 6 开始执行 bootloader。

步骤 7 进行 DDR 初始化，并且将 UFS device 存储的 fastboot 镜像拷贝到 DDR 中，初始化 ACPU 并且解复位后，ACPU 侧开始执行 fastboot，进行后续启动流程。

----结束

2.7 调试方式

Hi3660 系统调试的相关特性如下：

- JTAG 调试

芯片提供符合 *IEEE1149.1* 标准的 JTAG 接口，内部实现 JTAG 链路和 TAP 控制器，支持对 LPM3/IOM7/SC300/OCBC_M3/DDRA-B-C-D uC engine M3/A53/A73/MODEM A9/ISP A7 和 BBP DSP/HiFi DSP / IVP DSP 的调试。

2.7.1 JTAG 调试

Hi3660 提供符合 *IEEE1149.1* 标准的 JTAG 接口：

- 支持 DSP 仿真器对内部 4 个 DSP 进行调试；
- 支持 PC 连接 JTAG 仿真器对 ARM 处理器单独调试。

JTAG MUX 用于将外部 JTAG 管脚的信号连接到芯片中各个 core 上，具体详见“8.14 JTAG”。

调试步骤如下：

步骤 1 芯片上电复位。

步骤 2 配置 JTAG_SEL1 和 JTAG_SEL0 为 2’b01，将 CPU JTAG 复用到 JTAG 管脚上。

步骤 3 配置 JTAG_SEL1 和 JTAG_SEL0 为 2’b00，进入寄存器选择模式；配置系统控制寄存器 JTAGSYS_SW_SEL [7:0]，切换到选择的调试接口进行调试工作。

步骤 4 连接对应仿真器，打开对应调试软件，开始调试。

----结束

2.7.2 Coresight 调试

芯片包括了一个功能强大的 Debug 系统，集成了一套 ARM Coresight 系统，包含以下特性：

- Coresight 系统分为两个层次：顶层 coresight 和各 cluster 内部的 local coresight。
  具体包括：A73 coresight、A53 coresight。
- 支持侵入式调试（debug）和非侵入式调试（trace）
  其中，A73 /A53 同时支持 debug 和 trace。
- 支持软件 debug 和传统 JTAG debug 两种方式。

2.7.2.1 应用说明

2.8 可维可测

Hi3660 芯片提供如下可维可测手段：

- JTAG 调试

2.9 Memory Map

Hi3660 要求支持 8GByte/6GByte/4GByte DDR 存储解决方案，对接不同容量 DDR 器件，以及从不同处理器视角看到的系统地址空间不尽相同，总体规则如下：

- 当对接 4GByte 器件时，ACPU、IVP、GPU、VENC、VDEC、DSS、ISP 视角所看到的整芯片系统统一编址中的“0~3.5GB”和“4~4.5GB”地址空间规定为可访问到的 DRAM 的 4GByte 空间，而这些 Master 视角所看到的“3.5~4GB”地址空间为配置口空间，只可访问 0~3.5GB 的 DRAM 空间及 3.5~4GB 的外设空间。
- 当对接 8GByte 器件时，A53、A73、IVP、GPU、VENC、VDEC、DSS、ISP 视角所看到的整芯片系统统一编址中的“0~3.5GB”和“4~8.5GB”地址空间规定为可访问到的 DRAM 的 8GByte 空间，而这些 Master 视角所看到的“3.5~4GB”地址空间为配置口空间。MODEM 及外设子系统（其中包括 IOMCU、LPMCU、ASP、DMAC，以及 USB3OTG、SECENG、MMC 等外设）只可访问 0~3.5GB 的 DRAM 空间及 3.5~4GB 的外设空间。
- 当对接 6GByte 等其他 DDR 容量器件时，类似于对接 8GByte 器件时的地址映射方案，3.5~4GB 空间仍作为外设的访问空间。

各处理器由于自身的设计约束，其能够访问的指令、数据空间各不相同。将 SOC 主干总线（SYTEM BUS 及 CONFIG BUS）设计采用的 SLAVE 地址空间划分定义为这些设备的绝对地址。系统中的处理器或微控制器由于自身的设计约束，可能需要采用一定的映射规则才能访问到系统中的全部或部分设备，具体映射原则请参见“3 移动处理模块”。

2.9.1 ACPU 视角的地址空间分配

注意

表中的 Reserved 区域禁止访问，如访问 Reserved 区域会发生不可预知的结果。

表 2-3 列出了 Hi3660 芯片中 ACPU 所见的所有寄存器组和 Memory 的地址范围。

表2-3 寄存器组和 Memory 地址范围（ACPU）

Start addr	End addr	Size（byte）	Module
0xFFF38000	0xFFF38FFF	4K	PMU_SSI2
0xFFF36000	0xFFF36FFF	4K	PMU_SSI1
0xFFF35000	0xFFF35FFF	4K	PERI_CRG
0xFFF34000	0xFFF34FFF	4K	PMU_SSI0
0xFFF33000	0xFFF33FFF	4K	PMU_I2C
0xFFF32000	0xFFF32FFF	4K	UART6
0xFFF31000	0xFFF31FFF	4K	PMCTRL

Start addr	End addr	Size（byte）	Module
0xFFF30000	0xFFF30FFF	4K	TSENSORC
0xFFF20000	0xFFF2FFFF	64K	Reserved
0xFFF1F000	0xFFF1FFFF	4K	Reserved
0xFFF1D000	0xFFF1DFFF	4K	GPIO28
0xFFF1C000	0xFFF1CFFF	4K	TIMER8
0xFFF1B000	0xFFF1BFFF	4K	TIMER7
0xFFF1A000	0xFFF1AFFF	4K	TIMER6
0xFFF19000	0xFFF19FFF	4K	TIMER5
0xFFF18000	0xFFF18FFF	4K	TIMER4
0xFFF17000	0xFFF17FFF	4K	TIMER3
0xFFF16000	0xFFF16FFF	4K	TIMER2
0xFFF15000	0xFFF15FFF	4K	TIMER1
0xFFF14000	0xFFF14FFF	4K	TIMER0
0xFFF11000	0xFFF11FFF	4K	AO_IOC
0xFFF10000	0xFFF10FFF	4K	GPIO27
0xFFF0F000	0xFFF0FFFF	4K	GPIO26
0xFFF0E000	0xFFF0EFFF	4K	GPIO25
0xFFF0D000	0xFFF0DFFF	4K	GPIO24
0xFFF0C000	0xFFF0CFFF	4K	GPIO23
0xFFF0B000	0xFFF0BFFF	4K	GPIO22
0xFFF0A000	0xFFF0AFFF	4K	SCTRL
0xFFF08000	0xFFF09FFF	8K	SYS_CNT
0xFFF05000	0xFFF05FFF	4K	RTC1
0xFFF04000	0xFFF04FFF	4K	RTC0
0xFFD00000	0xFFD7FFFF	512K	IOMCU
0xFF400000	0xFFCFFFFF	9M	Reserved
0xFF3FF000	0xFF3FFFFF	4K	SDIO0
0xFF3FE000	0xFF3FEFFF	4K	PCIE_APB_CFG
0xFF3FD000	0xFF3FDFFF	4K	IOC_MMC1
0xFF3FC000	0xFF3FCFFF	4K	Reserved
0xFF3FB000	0xFF3FBFFF	4K	EMMC

Start addr	End addr	Size（byte）	Module
0xFF3E2000	0xFF3FAFFF	100K	Reserved
0xFF3E1000	0xFF3E1FFF	4K	GPIO1_MMC1
0xFF3E0000	0xFF3E0FFF	4K	GPIO0_MMC1
0xFF3B8000	0xFF3DFFFF	160K	Reserved
0xFF3B7000	0xFF3B7FFF	4K	Reserved
0xFF3B6000	0xFF3B6FFF	4K	IOC_FIX
0xFF3B5000	0xFF3B5FFF	4K	GPIO19
0xFF3B4000	0xFF3B4FFF	4K	GPIO18
0xFF3B3000	0xFF3B3FFF	4K	SPI3
0xFF3B2000	0xFF3B2FFF	4K	Reserved
0xFF3B1000	0xFF3B1FFF	4K	UFS_SYS_CTRL
0xFF3B0000	0xFF3B0FFF	4K	UFS_CFG
0xFF3A0000	0xFF3AFFFF	64K	Reserved
0xFF390000	0xFF39FFFF	64K	Reserved
0xFF380000	0xFF38FFFF	64K	Reserved
0xFF37F000	0xFF37FFFF	4K	SD3
0xFF37E000	0xFF37EFFF	4K	IOC_MMC0
0xFF37D000	0xFF37DFFF	4K	Reserved
0xFF300000	0xFF37CFFF	500K	Reserved
0xFF201000	0xFF2FFFFF	1020K	Reserved
0xFF200000	0xFF200FFF	4K	USB3OTG_BC
0xFF100000	0xFF1FFFFF	1M	USB3OTG
0xFF050000	0xFF0FFFFF	704K	Reserved
0xFF032000	0xFF03FFFF	56K	Reserved
0xFF013000	0xFF02FFFF	116K	Reserved
0xFF012000	0xFF012FFF	4K	Reserved
0xFF011000	0xFF011FFF	4K	IPC_MDM_NS
0xFF010000	0xFF010FFF	4K	IPC_MDM_S
0xFF00F000	0xFF00FFFF	4K	Reserved
0xFF000000	0xFF00EFFF	60K	Reserved
0xFDF31000	0xFDFFFFFF	828K	Reserved

Start addr	End addr	Size（byte）	Module
0xFDF30000	0xFDF30FFF	4K	PERI_DMAC
0xFDF20000	0xFDF2FFFF	64K	Reserved
0xFDF16000	0xFDF1FFFF	40K	Reserved
0xFDF15000	0xFDF15FFF	4K	Reserved
0xFDF14000	0xFDF14FFF	4K	Reserved
0xFDF13000	0xFDF13FFF	4K	Reserved
0xFDF12000	0xFDF12FFF	4K	Reserved
0xFDF11000	0xFDF11FFF	4K	Reserved
0xFDF10000	0xFDF10FFF	4K	PERF_STAT
0xFDF0D000	0xFDF0DFFF	4K	I2C4
0xFDF0C000	0xFDF0CFFF	4K	I2C3
0xFDF0B000	0xFDF0BFFF	4K	I2C7
0xFDF09000	0xFDF0AFFF	8K	Reserved
0xFDF08000	0xFDF08FFF	4K	SPI1
0xFDF07000	0xFDF07FFF	4K	Reserved
0xFDF06000	0xFDF06FFF	4K	SPI4
0xFDF05000	0xFDF05FFF	4K	UART5
0xFDF04000	0xFDF04FFF	4K	Reserved
0xFDF03000	0xFDF03FFF	4K	UART2
0xFDF02000	0xFDF02FFF	4K	UART0
0xFDF01000	0xFDF01FFF	4K	UART4
0xFDF00000	0xFDF00FFF	4K	UART1
0xFC000000	0xFDEFFFFF	31M	Reserved
0xF4000000	0xFBFFFFFF	128M	PCIECtrl
0xF3F40000	0xF3FFFFFF	768K	Reserved
0xF3F00000	0xF3F3FFFF	256K	PCIEPHY
0xF1300000	0xF3EFFFFF	44M	Reserved
0xF12F0000	0xF12FFFFF	64K	Reserved
0xF1110000	0xF12EFFFF	1920K	Reserved
0xF0E00000	0xF0E1FFFF	128K	Reserved
0xF0C00000	0xF0DFFFFF	2M	Reserved

Start addr	End addr	Size（byte）	Module
0xF0000000	0xF0BFFFFF	12M	IOMCU_TCM
0xED800000	0xEFFFFFFF	40M	Reserved
0xEC000000	0xED7FFFFF	24M	CSSYS_APB
0xE9890000	0xE989FFFF	64K	MMC0_NOC_Service_Target
0xE9880000	0xE988FFFF	64K	MMC1_NOC_Service_Target
0xE9870000	0xE987FFFF	64K	AOBUS_Service_Target
0xE9860000	0xE986FFFF	64K	DMA_NOC_Service_Target
0xE9810000	0xE981FFFF	64K	UFSBUS_Service_Target
0xE9800000	0xE980FFFF	64K	CFGBUS_Service_Target
0xE8E00000	0xE97FFFFF	10M	Reserved
0xE8DD0000	0xE8DFFFFF	192K	Reserved
0xE8A23000	0xE8AFFFFF	884K	Reserved
0xE8A20000	0xE8A20FFF	4K	GPIO21
0xE8A1F000	0xE8A1FFFF	4K	GPIO20
0xE8A1E000	0xE8A1EFFF	4K	Reserved
0xE8A1D000	0xE8A1DFFF	4K	Reserved
0xE8A1C000	0xE8A1CFFF	4K	GPIO17
0xE8A1B000	0xE8A1BFFF	4K	GPIO16
0xE8A1A000	0xE8A1AFFF	4K	GPIO15
0xE8A19000	0xE8A19FFF	4K	GPIO14
0xE8A18000	0xE8A18FFF	4K	GPIO13
0xE8A17000	0xE8A17FFF	4K	GPIO12
0xE8A16000	0xE8A16FFF	4K	GPIO11
0xE8A15000	0xE8A15FFF	4K	GPIO10
0xE8A14000	0xE8A14FFF	4K	GPIO9
0xE8A13000	0xE8A13FFF	4K	GPIO8
0xE8A12000	0xE8A12FFF	4K	GPIO7
0xE8A11000	0xE8A11FFF	4K	GPIO6
0xE8A10000	0xE8A10FFF	4K	GPIO5
0xE8A0F000	0xE8A0FFFF	4K	GPIO4
0xE8A0E000	0xE8A0EFFF	4K	GPIO3

Start addr	End addr	Size（byte）	Module
0xE8A0D000	0xE8A0DFFF	4K	GPIO2
0xE8A0C000	0xE8A0CFFF	4K	GPIO1
0xE8A0B000	0xE8A0BFFF	4K	GPIO0
0xE8A0A000	0xE8A0AFFF	4K	GPIO0_SE
0xE8A09000	0xE8A09FFF	4K	PCTRL
0xE8A07000	0xE8A07FFF	4K	WD1
0xE8A06000	0xE8A06FFF	4K	WD0
0xE8A04000	0xE8A04FFF	4K	PWM
0xE8A03000	0xE8A03FFF	4K	TIMER12
0xE8A02000	0xE8A02FFF	4K	TIMER11
0xE8A01000	0xE8A01FFF	4K	TIMER10
0xE8A00000	0xE8A00FFF	4K	TIMER9
0xE8971000	0xE89FFFFF	572K	Reserved
0xE896E000	0xE8970FFF	12K	Reserved
0xE896D000	0xE896DFFF	4K	Reserved
0xE896C000	0xE896CFFF	4K	IOC
0xE896B000	0xE896BFFF	4K	IPC_NS
0xE896A000	0xE896AFFF	4K	IPC
0xE8969800	0xE8969FFF	2K	Reserved
0xE8961800	0xE89697FF	32K	Reserved
0xE8961400	0xE89617FF	1K	Reserved
0xE8961000	0xE89613FF	1K	Reserved
0xE8960000	0xE8960FFF	4K	Reserved
0xE8950000	0xE895FFFF	64K	Reserved
0xE8680000	0xE86BFFFF	256K	Reserved
0xE8600000	0xE8600FFF	4K	Reserved
0xE8300000	0xE83FFFFF	1M	Reserved
0xE82C4000	0xE82FFFFF	240K	Reserved
0xE82C0000	0xE82C3FFF	16K	G3D
0xE82BA000	0xE82BFFFF	24K	Reserved
0xE82B9000	0xE82B9FFF	4K	CODEC_SSI

Start addr	End addr	Size（byte）	Module
0xE82B8000	0xE82B8FFF	4K	HKADC_SSI
0xE82B0000	0xE82B7FFF	32K	GIC400
0xE82A0000	0xE82AFFFF	64K	Reserved
0xE8200000	0xE829FFFF	640K	Reserved
0xE8100000	0xE81FFFFF	1M	CCI_CFG
0x00000000	0xDFFFFFFF	3584M	DRAM

说明

- 上表中描述的是 4GB 空间内的设备地址分配，DRAM 的空间是 0~3.5G，当对接 8GByte/6GByte/4GByte DDR 时，DRAM 会占据 4GB 空间外的地址，具体规则参见前文描述。

3 移动处理模块

3.1 CPU

3.1.1 概述

主处理器采用由 Cortex-A73 MP 和 Cortex-A53 MP 构成的大小核 big.LITTLE 异构 CPU 子系统来实现。Cortex-A73 MP 和 Cortex-A53 MP 都基于 ARMv8-A 架构。

Cortex-A73 MP 的基本特性如下：

- 处理器性能 3.66DMIPS/MHz。
- 超标量、变长、乱序执行流水线。
- 动态分支预测配置跳转目标地址缓存、全局历史地址缓存、返回地址栈、非直接跳转预测器。
- L1 指令 TLB 为全相连结构，包括 32 个表项，支持 4KB/16KB/64KB/1MB 大小的页表项；
- L1 数据 TLB 为全相连结构，包括 48 个表项，支持 4KB/16KB /64KB/1MB 大小的页表项；
- L2 TLB 为 4 路组相连结构，包括 1024 个表项；
- 固定 L1 指令 Cache 为 64KB，L1 数据 Cache 为 64KB；
- 数据和指令共享 L2 Cache 为 2MB。
- ACE 总线接口。
- 支持 ETM Trace 调试。
- 支持 CTI 多核联合 Debug 调试。
- 配备性能计数单元，符合 PMUv3 架构。
- 支持 VFP 和 NEON 单元。
- 支持基于 ARMv8 的 Cryptography 扩展指令
- 使用外部 GIC。
- 每个 CPU 配置内部 64-bit 通用计数器。
- 支持 CPU 核单独掉电。

Cortex-A53 MP 的基本特性如下：

- 处理器性能 2.3DMIPS/MHz。
- 按序流水线，支持双指令执行功能。
- 支持直接和非直接分支预测。
- 针对指令，数据读写配备 2 个独立的全相关 L1 TLB。每个 TLB 具有 10 个表项。
- L2 TLB 为 2 路组相连结构，具有 256 个表项。
- L1 数据和指令 Cache 都为 32KB。
- 数据和指令共享 L2 Cache 为 512KB。
- ACE 总线接口。
- 支持 ETM Trace 调试。
- 支持 CTI 多核联合 Debug 调试。
- 配备性能计数单元，符合 PMUv3 架构。
- 支持 VFP 和 NEON 单元。
- 支持基于 ARMv8 的 Cryptography 扩展指令
- 使用外部 GIC。
- 每个 CPU 配置内部 64-bit 通用计数器。
- 支持 CPU 核单独掉电。

A73 MP 和 A53MP 之间使用 CCI-550 实现了 Cache 数据一致性，利用 GIC-400 实现了中断虚拟化，用 System Counter 实现了 Timer 虚拟化，使用 Event 接口进行事件交互。

3.1.2 工作模式

3.1.2.1 工作状态

由 ARMv8-A 架构决定， A73 MP 和 Cortex-A53 MP 的工作状态分为 4 个维度：

- 架构状态。支持 AArch32 和 AArch64。
- 指令集状态。由指令集支持决定，支持 A32、T32 和 A64 几种状态。
- 异常等级状态。分为 4 种，在后文详细描述。
- 安全状态。非安全和安全两种。

3.1.2.2 异常等级

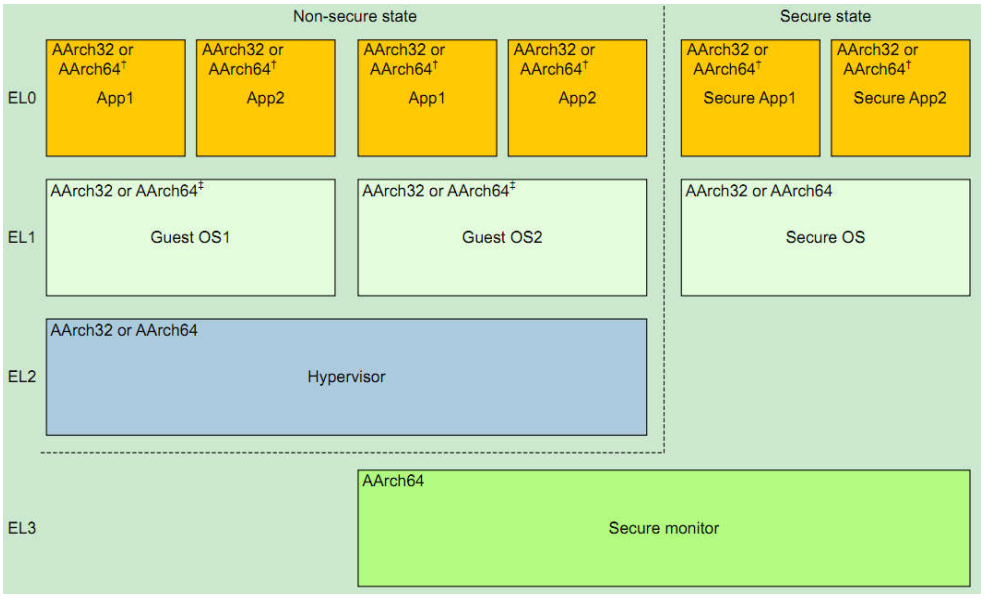
ARMv8-A 异常等级如表 3-1 所示。

表3-1 ARMv8-A 的异常等级

处理器异常等级	描述
EL0	用户程序执行模式，非特权级，安全/非安全两套
EL1	操作系统运行模式，特权级，安全/非安全两套
EL2	用于虚拟化扩展功能，仅存在于非安全
EL3	用于切换安全和非安全世界

以上各种运行模式之间的对应关系如图 3-1 所示。

图3-1 运行模式关系图



3.1.3 一致性总线

Big.LITTLE 架构中大小核之间的数据一致性通过 ARM 的一致性总线，CCI-550 来实现，其特性包括：

- 支持 AMBA FULL-ACE 协议和 AMBA ACE-Lite 协议，可实现大小核，以及其他一致性 Master 之间的数据一致性。
- Crossbar 结构的总线互联结构，可实现上游 Master 对内存和配置空间的访问。
- 实现了 snoop filter 来提高一致性 Master 之间 snoop 的性能。
- 支持大小核之间的 DVM 消息广播。
- 支持基于带宽的 QoS 流量管理机制。
- 支持静态交织粒度和交织方式配置。
- 支持通过 PMU 计数器对 master/slave/global 等多种事件进行性能统计。
- 支持通过 Q-channel 握手实现 CCI 自动门控机制。
- 支持通过 P-channel 握手实现 CCI 内部 RAM 的 dynamic retention 功能。
- 通过 APB slave 配置接口来控制一致性及总线功能。

4 系统控制

4.1 SC

4.1.1 CRG

CRG 的用途主要是给各 IP 提供时钟、复位。这里只介绍 AO CRG、PERIPH CRG 的功能，其余 CRG 的功能在各自模块的时钟复位章节描述。

4.1.1.1 AOCRG

4.1.1.1.1 功能描述

AO CRG 主要支持以下功能：

- 为系统区各 IP 提供时钟复位；
- 系统区各 IP 时钟复位的软件控制和状态查询以及时钟分频比配置都在 SC 中控制。
- 支持 AONOC 自动降频功能。

4.1.1.2 PERICRG

4.1.1.2.1 功能描述

PERIPH CRG 主要支持以下功能：

- 支持 APB 接口访问寄存器；
- 支持安全与分安全访问寄存器区域隔离；
- 为外设区的各 IP 以及 NOC 总线提供时钟复位；
- 支持外设区各 IP 以及 NOC 总线时钟复位的软件控制和状态查询；
- 支持外设区各 IP 以及 NOC 总线的时钟分频比配置；
- 为 A53_B 、A53_L 、LPM3、CCI500、ADB、CORESIGHT 等模块提供软件配置接口以及状态查询接口；
- 支持 APB、AHB 总线接口防挂死功能；

- 支持外设区 Timer 与 WatchDog 时钟使能控制：
  - 时钟使能可以使计数频率独立于系统时钟频率，即使系统时钟发生改变，计数器仍然会保持固定的计数频率；
  - 支持对输入的计数时钟进行采样，生成时钟使能信号，输出给外设区的 Timer4、Timer5、Timer6、Timer7 模块；
  - 支持对外设区 Timer 的计数时钟源进行选择，可选择 32kHz 计数或 4.8MHz 计数；
  - 外设区 Timer 可由软件配置使能强制拉高，此时 Timer 的工作时钟是总线时钟。
  - 外设区 WatchDog 可由软件配置使能强制拉高，此时 WatchDog 的工作时钟是总线时钟；

    如果外设区 WatchDog 没有强制拉高，外设区 WatchDog 的计数时钟为 32kHz。
- 支持 A53_B、A53_L core 自动上下电状态机控制：
  - 支持软件配置寄存器独立使能、不使能状态机上下电 A53_B、A53_L core 功能；
  - 支持上下电状态机使能时，A53_B、A53_L 各个 core 上下电完成是否发送中断，单独软件可配置；同时上下电完成中断状态可单独查询；
  - 支持上下电状态机使能时，响应 GIC 中断，上电 A53_B、A53_L 对应的 core；同时是否响应 GIC 中断，A53_B、A53_L 每个 core 均单独可配置；
  - 支持上下电状态机使能时，软件写寄存器，独立上电 A53_B、A53_L 各个 core；
  - 支持上下电状态机使能时，软件写寄存器，独立下电 A53_B、A53_L 各个 core；
  - 支持上下电状态机使能时，实时查询 A53_B、A53_L 每个 core 上下电状态机的当前状态；
  - 支持独立配置 A53_B、A53_L core 上下电过程中的复位时间、解复位时间、MTCMOS 上电等待时间、ISO 等待时间、解 ISO 等待时间、DBG 配置等待时间。
- 支持 A53_B ocldo 状态机控制：
  - 支持状态机响应大核 4 个 core 的独立的 ocldo 请求信号启动状态机
  - 支持状态机控制大核 4 个 core 的 mtcmos 控制信号
  - 支持状态机控制大核 4 个 core 的 iso 钳位信号，该功能可以静态 bypass
  - 支持状态机控制大核 4 个 core 的 ocldo 使能信号
  - 支持软件查询当前状态机的状态
- 支持 DDR 时钟自动门控/降频。

4.1.2 SCTRL

4.1.2.1 功能描述

SC（System Controller）是系统控制器，为用户提供系统控制接口，由 APB 接口模块、系统主控制模块、中断响应模式请求模块、上下电控制模块、通用计数器模块、晶振控制模块、LPRAM 的低功耗控制、EFUSE 控制模块等组成。

系统控制器的功能特点有：

- 总线接口：
  - 支持 AMBA2.0 APB Slave 接口规范，数据位宽 32bit，地址位宽 12bit。
- 复位：
  - 系统控制器接受 preset_n 复位。
  - 支持设置全局软复位，全局软复位请求送到 CRG。

4.1.3 PCTRL

4.1.3.1 功能描述

PCTRL（Peripheral Control）的用途主要是给部分外部 IP 提供配置接口。PCTRL 只在系统进入 normal 模式后起作用，因此在使用 PCTRL 的任何功能之前，需要确认系统已经处于 normal 模式。

PCTRL 主要支持以下功能：

- 支持 3D LCD 光栅控制
- 为部分外设 IP 提供配置接口
- 支持资源锁

4.2 RTC

4.2.1 功能描述

RTC（Real Time Clock）是实时时钟，实现时间显示和定时报警功能。本芯片中含有三个 RTC 分别为 RTC0、RTC1、 RTC2。

RTC 的运行基于 1 个 32bit 加法计数器，计数初值由寄存器 RTCLR 提供。计数器的值在每个计数时钟的上升沿加 1。当计数值递加到寄存器 RTCDR 与寄存器 RTCMR 的值相等时，RTC 产生一个中断，然后在下一个计数时钟上升沿，计数器继续递加计数。

RTC 的计数时钟采用 1Hz 时钟，便于通过计数值转换为具体的年、月、日、时、分、秒。

RTC 的功能特点有：

- 计数初值可配置
- 计数比较值可配置
- 支持超时中断产生
- 支持软复位

4.2.2 寄存器描述

详见 ARM Timer IP SP804 手册网址（http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0270b/index.html）

4.3 WatchDog

4.3.1 功能描述

Watchdog 用于系统异常情况下，在一定时间内发出复位信号，以复位整个系统。本芯片中含有两个 WD，分别为 WD0、WD1。

Watchdog 的功能特点有：

- 一个 32bit 减法计数器，计数时钟可配置。
- 支持超时时间间隔（即计数初值）可配置。
- 支持寄存器锁定，防止寄存器被误改。
- 支持超时中断产生。
- 支持复位信号产生。
- 支持调试模式。

Watchdog 的运行基于 1 个 32bit 减法计数器，计数初值由寄存器 WdogLoad 载入。在 Watchdog 时钟使能情况下，计数器的值在每个计数时钟的上升沿减 1。当计数值递减到 0，Watchdog 将产生一个中断。然后在下一个计数时钟上升沿，计数器又从寄存器 WdogLoad 中重新载入计数初值，开始递减计数。

如果计数器第 2 次计数递减到 0 时，CPU 还没有清除 Watchdog 中断，则 Watchdog 将发出复位信号，计数器停止计数。

配置寄存器 WdogControl，使能或者禁止 Watchdog 产生中断和复位信号。

- 当禁止产生中断时，计数器将停止计数。
- 当重新使能中断时，Watchdog 将从寄存器 WdogLoad 的设定值开始计数，而不是从计数器上次停止时的计数值开始计数。在中断到来之前，可以重新载入初值。

Watchdog 的计数时钟可以选择睡眠时钟或者总线时钟，便于选择不同的计数时间范围。

通过配置寄存器 WdogLock，可以禁止对 Watchdog 寄存器的写操作。

- 向 WdogLock 写入 0x1ACC_E551，打开所有 Watchdog 寄存器的写权限。
- 向 WdogLock 写入其他任何值，关闭所有 Watchdog 寄存器（寄存器 WdogLock 除外）的写权限。

该特性保护 Watchdog 的寄存器不被软件错误地修改，从而使得在异常情况下，Watchdog 不致被软件错误地中止操作。

ARM 在调试模式时，Watchdog 自动关闭，以防止干扰正常的调试操作。

注意

在系统进入 SLEEP 模式之前，必须关闭 Watchdog。具体操作请参见“错误！未找到引用源。”。

4.3.2 寄存器描述

详见 ARM Watchdog IP SP805x 手册网址
（http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0270b/index.html）

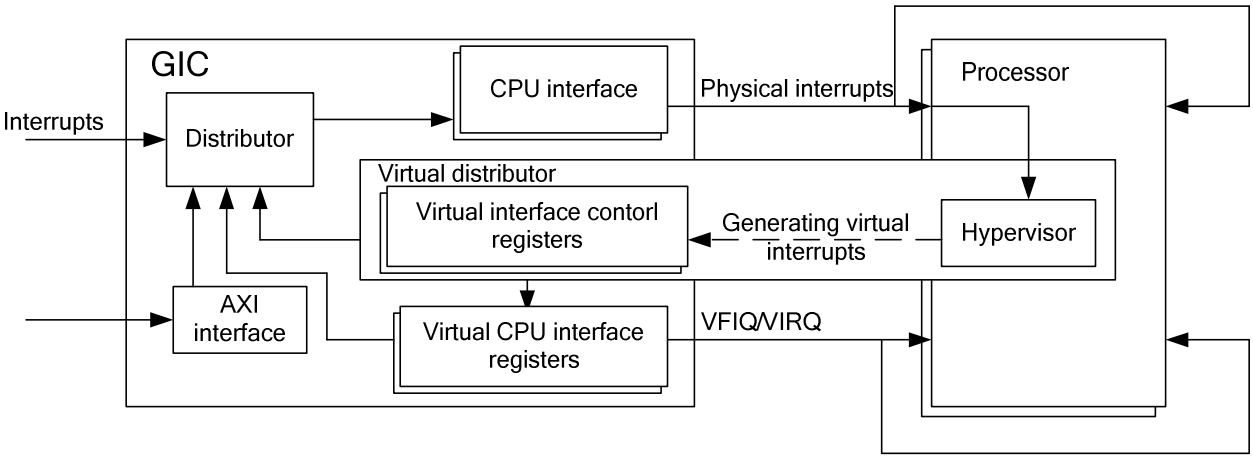
4.4 GIC

4.4.1 功能描述

GIC 用于实现外部中断、内部中断及软件中断的检测、管理和分配。GIC 支持安全扩展和虚拟化扩展。

GIC 模块的结构框图如图 4-1 所示。

图4-1 GIC 结构框图



GIC 的功能特点有：

- 支持三种中断：软中断 SGIs、私有外设中断 PPIs 和共享外设中断 SPIs。
- 支持电平敏感和边沿触发方式中断，每个中断源的触发方式可单独配置。
- 为每个 CPU Core 提供一个物理中断 FIQ 和 IRQ 接口。
- 支持虚拟化扩展，只支持物理 IRQ 中断虚拟化。通过配置产生虚拟中断 Virtual FIQ 和 Virtual IRQ，同时为每个 CPU Core 增加一个虚拟中断 VFIQ 和 VIRQ 接口。
- 支持中断路由，物理中断查询目标处理器寄存器路由至相应 CPU Core；虚拟中断的路由目标由 Hyperisor 根据 VMID 等信息配置目标处理器寄存器，路由至相应 CPU Core。
- 支持中断单独使能。
- 支持中断状态查询。
- 支持中断屏蔽，设置 Mask Priority，中断的优先级大于 Mask Priority 的才能上报。

- 支持中断嵌套，对物理中断，中断嵌套的深度由 Group Priority 决定，只有 Group Priority 较大中断才能引发中断嵌套，Group Priority 的级数可配；对虚拟中断，其中断嵌套深度为 32。
- 为每个 CPU Core 提供六个内部器件的 PPI、一个内部虚拟 maintenance PPI 中断、一个 BypassFIQ 和一个 BypassIRQ 输入，各核之间 PPI 中断相互独立。
- 支持 TrustZone 扩展，输入中断可按照 Secure/Non-Secure 分组。
- 对于物理中断，所有 Secure 中断的优先级高于 Non-secure 中断。在 Secure 模式下，支持最少 5bits 中断优先级域，最多可扩展到 8bit；在 Non-secure 模式下，支持最少 4bits 中断优先级域，最多可扩展到 7bits；中断优先级的值越小说明该中断优先级越大。对于虚拟中断，其优先级等级固定为 32 等级。
- 支持 SPIs 中断锁定，支持 32 个 LSPIs，LSPI 只能是安全中断，锁定后其相关配置寄存器不能被改写。

  支持中断唤醒事件，预留 nFIQOUT，nIRQOUT 直接输出端口，在 GIC 不使能的情况下也能将中断作为唤醒事件上报。

4.4.2 寄存器描述

详见 ARM GIC-400 手册网址
（http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0270b/index.html）

4.5 DMAC

4.5.1 概述

AXI DMAC 是基于 AMBA AXI 协议，面向 SoC 系统芯片应用而开发的低成本高性能 IP。直接内存访问（DMA）方式是一种完全由硬件执行数据搬移的工作方式，在这种方式中，DMA 控制器通过一个独立于处理器的总线主控制器执行数据搬移操作，不需要经过 CPU，数据被直接在存储器之间，存储器与外设之间进行搬移。DMA 方式一般用于高速传送成组或者成块的数据。DMA 控制器收到 DMA 传输请求后根据 CPU 的配置启动总线主控制器，向内存或外设发出地址和读写控制信号，并对传送的次数进行计数，一般以中断方式向 CPU 报告传送操作的完成。

Hi3660 DMAC 的特点：

- 16 个逻辑通道（Channel）；
- 逻辑通道优先级可编程；
- 1 个 AXI Master 接口；
- 1 个 APB2.0 配置接口；
- 支持 32 套外设请求（每套请求包括 Single、Burst 两种请求）；
- 支持存储器到存储器、存储器到外设、外设到存储器传输；
- 数据在 memory 和外设之间传输时，支持 DMAC 流控方式；
- 支持链表传输；
- 支持逻辑通道链接传输；
- 支持 1 维、2 维、3 维传输模式；

- 支持 4 套中断；
- 支持一致性与非一致性操作；
- 支持安全与非安全搬运及对应中断上报。

4.5.2 应用背景

DMA 方式是一种完全由硬件执行 I/O 交换的工作方式。在这种方式中，DMAC 直接在存储器和外设、存储器和存储器之间进行数据传输，避免处理器干涉并减少了处理器中断处理开销。

DMA 方式一般用于高速传输成组的数据。DMAC 在收到 DMA 传输请求后根据 CPU 对通道的配置启动总线主控制器，向存储器和外设发出地址和控制信号，对传输数据的个数计数，并且以中断方式向 CPU 报告传输操作的结束或错误。

DMAC 对系统性能的改进表现在以下两个方面：

- DMAC 完成了数据的搬移操作，减轻了 CPU 的负担；
- 通过编程 DMAC，给 I/O 外设分配通道从而实现大数据量的传输，减少了中断的频率。

4.5.3 功能描述

DMAC 的功能框图如**错误！未找到引用源。**所示，主要包括以下接口：

- DMAC 硬件请求接口
- 16 个通道
- 仲裁器
- AXI Master 接口
- APB Slave 接口

对一个传输来说，必须首先触发 DMAC 的一个逻辑通道（Channel），Master 接口从源地址读取数据并将其写入目标地址中。

DMAC Hardware Request I/F 用于外设请求 DMAC 搬运数据。

以一个软件触发的数据搬移为例：

处理器首先通过 APB slave 接口配置一个逻辑通道，明确搬移的源地址、目的地址和搬移方式。然后触发搬移使能。经过 Arbiter 对通道进行仲裁以后，DMAC 根据配置的信息，从 AXI master 接口读取源地址的数据，放到目的地址上，完成产生中断。

4.5.4 应用说明

4.5.4.1 初始化配置

系统初始化时 DMAC 按以下步骤进行配置。

步骤 1 配置通道的参数寄存器，根据所需要的传输配置 CX_AXI_SPECIAL、CX_SRC_ADDR、CX_DES_ADDR、CX_LLI，CX_CNT0、CX_CNT1、CX_BINDX 和 CX_CINDX。如

果是一维非链表传输，且无需发出 AXI CACHE 等操作只需要配置 CX_SRC_ADDR、CX_DES_ADDR 和 CX_CNT0 寄存器，可以缩短寄存器配置的时间。

说明：如果是一维非链表传输只需要配置 CX_SRC_ADDR、CX_DES_ADDR、CX_CNT0 和 CX_CONFIG 寄存器，可以缩短寄存器配置的时间，但需要保证其它寄存器为默认值。

步骤 2 配置 CX_CONFIG 寄存器，确定传输方式，对应外设号，总线的操作参数等。

步骤 3 配置 CX_CONFIG bit[0]为 1，使能通道，DMAC 开始传输。

----结束

上面步骤 1、2 顺序可以变化，但是需要保证最后配置 CX_CONFIG bit[0]为 1，启动传输。

4.5.4.2 典型配置流程

4.5.4.2.1 Memory 传输

这里以 memory 一维 incr 传输，burst size 32bit，burst length 8 做为实例

步骤 1 配置时钟使能，解复位。

步骤 2 配置中断屏蔽寄存器，打开所有中断。

步骤 3 确定当前传输使用 DMA 哪个逻辑通道，下面的配置即配置对应通道的这些寄存器。

步骤 4 配置通道的参数寄存器，根据所需要的传输配置 CX_AXI_SPECIAL、CX_SRC_ADDR、CX_DES_ADDR、CX_LLI、CX_CNT0。如果是一维非链表传输，且无需发出 AXI CACHE 等操作只需要配置 CX_SRC_ADDR、CX_DES_ADDR 和 CX_CNT0 寄存器，可以缩短寄存器配置的时间。即传输源地址，传输目的地址，搬运整个数据量。

如果对于安全属性有要求时需要配置 CX_AXI_SPECIAL，否则默认为安全通道。详细的安全属性介绍请见“**错误！未找到引用源。错误！未找到引用源。**”。

说明：如果是一维非链表传输只需要配置 CX_SRC_ADDR、CX_DES_ADDR、CX_CNT0 和 CX_CONFIG 寄存器，可以缩短寄存器配置的时间，但需要保证其它寄存器为默认值。

步骤 5 配置 CX_CONFIG 寄存器，确定传输方式，总线的操作参数等。此时是 memory 传输不需要配置外设号，流控方式这里只能选择“00：存储器和存储器之间传输，DMAC 流控”。

步骤 6 配置 CX_CONFIG bit[31] bit[30]均为 1，控制 DMA 搬运源地址目的地址为 INCR 操作。

配置 CX_CONFIG bit[27:24] bit[23:20]均为 0111，控制 DMA 源传输目的传输 burst 长度均为 8。

配置 CX_CONFIG bit[18:16] bit[14:12]均为 010，控制 DMA 源传输目的传输数据宽度均为 32bit。

配置 CX_CONFIG bit[3:2]为 00，控制 DMA 传输流控方式为 memory 传输的 DMA 流控。

步骤 7　配置 CX_CONFIG bit[0]为 1，使能通道，DMAC 开始传输。

----结束

4.5.4.2.2 外设传输

这里以 memory 到外设一维传输，DMA 流控，burst size 32bit，burst length 8 做为实例。

步骤 1　这里重复上述 memory 传输的前二步步骤。

步骤 2　确定当前传输使用 DMA 哪个逻辑通道，下面的配置即配置对应通道的这些寄存器。

步骤 3　配置通道的参数寄存器，根据所需要的传输配置 CX_AXI_SPECIAL、CX_SRC_ADDR、CX_DES_ADDR、CX_LLI、CX_CNT0。如果是一维非链表传输，且无需发出 AXI CACHE，一致性等操作只需要配置 CX_SRC_ADDR、CX_DES_ADDR 和 CX_CNT0 寄存器，可以缩短寄存器配置的时间。即传输源地址，传输目的地址，搬运整个数据量。　如果对于安全属性有要求时需要配置 CX_AXI_SPECIAL，否则默认为安全通道。详细的安全属性介绍请见“**错误！未找到引用源。错误！未找到引用源。**”。

说明：如果是一维非链表传输只需要配置 CX_SRC_ADDR、CX_DES_ADDR、CX_CNT0 和 CX_CONFIG 寄存器，可以缩短寄存器配置的时间，但需要保证其它寄存器为默认值。

步骤 4　配置 CX_CONFIG 寄存器，确定传输方式，外设号，总线的操作参数等。

步骤 5　配置 CX_CONFIG bit[31] 为 1，bit[30]为 0，控制 DMA 搬运源地址为 INCR 操作，目的地址由于对接外设所以是 FIX 操作。

配置 CX_CONFIG bit[27:24] bit[23:20]均为 0111，控制 DMA 源传输目的传输 burst 长度均为 8。

配置 CX_CONFIG bit[18:16] bit[14:12]均为 010，控制 DMA 源传输目的传输数据宽度均为 32bit。

配置 CX_CONFIG bit[9:4]，确定使用外设的外设号，这个外设号是每个外设特有的在设计中已经分配好的。详细见下面说明。

配置 CX_CONFIG bit[3:2]为 01，控制 DMA 传输流控方式为外设传输的 DMA 流控。

步骤 6　配置 CX_CONFIG bit[0]为 1，使能通道，DMAC 开始传输。

----结束

4.5.5 寄存器描述

The base address for AP DMAC registers is 0xFDF30000

下表为 describes DMAC registers.

Offset	Register	Register Description
0x0000+0x40*in	INT_STAT	处理器 X 的中断状态寄存器。

0x0004+0x40*in	INT_TC1	处理器 X 的通道传输完成中断状态寄存器。
0x0008+0x40*in	INT_TC2	处理器 X 的链表节点传输完成中断状态寄存器。
0x000C+0x40*in	INT_ERR1	处理器 X 的配置错误中断状态寄存器。
0x0010+0x40*in	INT_ERR2	处理器 X 的数据传输错误中断状态寄存器。
0x0014+0x40*in	INT_ERR3	处理器 X 的读链表错误中断状态寄存器。
0x0018+0x40*in	INT_TC1_MASK	处理器 X 的通道传输完成中断屏蔽寄存器。
0x001C+0x40*in	INT_TC2_MASK	处理器 X 的链表节点传输完成中断屏蔽寄存器。
0x0020+0x40*in	INT_ERR1_MASK	处理器 X 的配置错误中断屏蔽寄存器。
0x0024+0x40*in	INT_ERR2_MASK	处理器 X 的数据传输错误中断屏蔽寄存器。
0x0028+0x40*in	INT_ERR3_MASK	处理器 X 的链表读取错误中断屏蔽寄存器。
0x0600	INT_TC1_RAW	原始通道传输完成中断状态寄存器。
0x0608	INT_TC2_RAW	原始链表节点传输完成中断状态寄存器。
0x0610	INT_ERR1_RAW	原始配置错误中断状态寄存器。
0x0618	INT_ERR2_RAW	原始数据传输错误中断状态寄存器。
0x0620	INT_ERR3_RAW	原始链表读取错误中断状态寄存器。
0x660	SREQ	单传输请求寄存器。
0x664	LSREQ	末次单传输请求寄存器。
0x668	BREQ	突发传输请求寄存器。
0x66C	LBREQ	末次突发传输请求寄存器。
0x670	FREQ	批量传输请求寄存器。
0x674	LFREQ	末次批量传输请求寄存器。
0x688	CH_PRI	优先级控制寄存器。
0x690	CH_STAT	全局 DMA 状态寄存器。
0x0694	SEC_CTRL	DMA 全局安全控制寄存器。
0x0698	DMA_CTRL	DMA 全局控制寄存器。
0x0700+0x10*cn	CX_CURR_CNT1	通道 X 的三维传输剩余 size 状态寄存器。

0x0704+0x10*cn	CX_CURR_CNT0	通道 X 的一、二维传输剩余 size 状态寄存器。
0x0708+0x10*cn	CX_CURR_SRC_ADDR	通道 X 的源地址寄存器。
0x070C+0x10*cn	CX_CURR_DES_ADDR	通道 X 的目的地址寄存器。
0x0800+0x40*cn	CX_LLI	通道 X 的链表地址寄存器。
0x0804+0x40*cn	CX_BINDX	通道 X 的二维地址偏移量配置寄存器。
0x0808+0x40*cn	CX_CINDX	通道 X 的三维地址偏移量配置寄存器。
0x080C+0x40*cn	CX_CNT1	通道 X 的传输长度 1 配置寄存器。
0x0810+0x40*cn	CX_CNT0	通道 X 的传输长度配置寄存器。
0x0814+0x40*cn	CX_SRC_ADDR	通道 X 的源地址寄存器。
0x0818+0x40*cn	CX_DES_ADDR	通道 X 的目的地址寄存器。
0x081C+0x40*cn	CX_CONFIG	通道 X 的配置寄存器。
0x0820+0x40*cn	CX_AXI_CONF	通道 X 的 AXI 特殊操作配置寄存器。

5 媒体处理

5.1 概述

Hi3660 集成了强大的多媒体处理子系统，用于图像采集和处理、LCD 显示控制、视频编解码加速、2D/3D 图形加速、音频采集/输出处理等应用。

5.2 VENC

5.2.1 功能描述

5.2.1.1 概述

Hi3660 集成了 H.264/H265/JPEG 硬件编码器，支持 H.264/H265/JPEG 编码标准。

VENC 的功能特性：

- 含有一个 JPEG 编码器
  - 支持 ITU-T T.81 baseline DCT 即扫描顺序。
  - 输入图像数据格式有 420PL111YCbCr8、420PL12YCbCr8、420PL12YCrCb8、422PL111YCbCr8、422PL12YCbCr8、422PL12YCrCb8、422IL3YCbYCr8、422IL3YCrYCb8、422IL3CbYCrY8、422IL3CrYCbY8。
  - 输出数据格式为 JFIF 1.02 和 Non-progressive JPEG。
  - 支持最大像素 16K x 16K。水平和垂直步长为 16 像素。
- 含有一个 H264 编码器。
  - H.264 Baseline Profile 使用的工具：

    I 和 P slice。

    帧内支持 4x4/16x16 的划分，16x16 支持 DC/V/H 预测模式，4x4 支持 DC/V/H 预测模式

    帧间支持/8x8/16x16 的划分

    支持 MB 级码率控制

    支持 1/2 和 1/4 像素的帧间预测。

余弦变换支持 Hadamard 变换。

支持 deblocking。

- H.264 Main Profile 增加使用的工具：

  支持 CABAC。

- H.264 High Profile 增加的工具：

  除 4x4 变换外增加 8x8 变换。

  支持帧内 8x8 预测。

- 输入图像数据格式有

  planar YUV 4:2:0

  planar YUV 4:2:2

  semi-planar YUV 4:2:0

  semi-planar YVU 4:2:0

  package UYVY4:2:2，VYUY4:2:2

  package YUYV4:2:2，YVYU4:2:2

  ARGB/BGRA8888

  ABGR/RGBA8888

- 输出数据格式为各格式的原始码流。
- 支持最大像素 4K x 2K。像素水平和垂直步长增量为 2。
- 最大帧率为 720P@240。
- 最大码率为 80Mbps。
- 性能，单 pipe 4K2K@30f。
- 支持帧存储格式：Linear
- 支持最小尺寸 176x144。
- 支持缩放，最大水平垂直各 1/4
- 支持 ROI(感兴趣区域编码)
- 多路编码：支持 4 路 H264 编码

- 含有一个 H265 编码器。
  - H.265 MainProfile 使用的工具：

    I 和 P slice。

    帧内支持 4x4/8x8/16x16/32x32 的划分，4x4/8x8/16x16 划分都支持 35 种预测模式，32x32 支持 DC/Planar 预测模式；

    帧间支持 8x8/16x16/32x32/64x64 的划分

    支持 CU 级码率控制

    支持水平正负 512，垂直正负 144 的整数搜索

    支持 1/2 和 1/4 像素的帧间预测。

    支持 DCT4/8/16/32，支持 DST4。

    支持 Merge/MergeSkip

    支持 TMV

    支持 deblocking/SAO。

- 输入图像数据格式有
  planar YUV 4:2:0
  planar YUV 4:2:2
  semi-planar YUV 4:2:0
  semi-planar YVU 4:2:0
  package UYVY4:2:2，VYUY4:2:2
  package YUYV4:2:2，YVYU4:2:2
  ARGB/BGRA8888
  ABGR/RGBA8888
- 输出数据格式为各格式的原始码流。
- 支持最大像素 4K x 2K。像素水平和垂直步长增量为 2。
- 最大帧率为 720P@240。
- 最大码率为 60Mbps。
- 性能，单 pipe 4K2K@30f。
- 支持帧存储格式：Linear
- 支持最小尺寸 176x144。
- 支持缩放，最大水平垂直各 1/4
- 支持 ROI(感兴趣区域编码)
- 多路编码：支持 4 路 H265 编码

5.3 VDEC

5.3.1 功能描述

5.3.1.1 概述

Hi3660 集成了一个支持 H.265、H.264、MPEG1、MPEG2、MPEG4、VC1（包括 WMV9，以下简称 VC1）、VP6、VP8 协议的视频解码器 VDEC。

VDEC 由运行于 ARM 处理器的 VFMW（Video Firmware）和内嵌的硬件视频解码引擎构成，VFMW 从上层软件获得码流，对码流进行解析并调用硬件视频解码引擎，产生解码图像序列。解码图像序列在上层软件的控制下，由后级模块输出到显示器或其它设备。

5.4 DSS

5.4.1 功能描述

DSS 对 Base、Video、Graphic 多个图层进行叠加处理和 3D 合成，把叠加合成后的像素送往 DSI 或者 HDMI 显示。

5.5 ASP

5.5.1 概述

ASP（Audio Signal Processing）是用于管理和处理各种音频、语音数据应用的子系统。ASP 支持音频播放、录音、数字 FM 播放、蓝牙语音拨号（录音）、通话上下行、混音、语音唤醒等应用场景。

5.5.2 DSP

5.5.2.1 功能描述

5.5.2.1.1 功能

- 音频 DSP 采用 tensilica 的 HiFi3 处理器，主要用于完成高清视频伴音解码（如 DTS）及后处理、专用播放器普通音频解码（如 MP3）等场景。

5.5.2.1.2 规格

5.5.3 SIO

5.5.3.1 功能描述

音频输入输出接口 SIO（Sonic Input/Output），用于和片外 Audio CODEC 芯片连接，完成音乐（语音）的播放及录制。其主要作用就是传输符合 I2S/PCM 协议的数字数据。ASP 内部集成 2 个 SIO,SIO0、SIO2。SIO0 为 SIO_AUDIO，用于音乐播放和录音功能，支持 I2S 及 PCM 格式的输入输出。SIO2 为 SIO_BT，用于进行蓝牙通话。

- SIO 支持主从两种模式的运行，同时支持播放 TX，录音 RX。
- SIO 模块支持 I2S 和 PCM 两种接口时序，这两种模式的接口信号线完全复用。
- 该 SIO 模块只支持时钟和同步信号的输入，如果需要作为 I2S 主模式，需要与 CRG 模块配合，由 CRG 模块对外送出时钟和同步信号。

I2S 模式特性如下：

- I2S 接口支持 16，18，20，24，32 位传输工作模式；
- 支持 8Ksps～192Ksps 采样率。
- 扩展模块支持 2/4/8/16 路接收，传输工作模式为 8、16 位。
- I2S 接收通道和发送通道具有独立的 FIFO，并且，每个通道的左声道和右声道均有独立的 FIFO，其 FIFO 深度为 16，宽度为 32bit；
- I2S 支持 FIFO disable 功能，在 FIFO disable 状态下，接收和发送数据均不经过 FIFO，直接在一个 buffer 中缓存；
- I2S 支持 TX 和 RX 通道单独使能，如果某个通道不使能，则该通道的控制单元及数据存储单元都不会翻转，以节省功耗；

- 对于 I2S16bit 位宽传输模式，支持左右声道接收数据合并成一个 32bit 数据在接收 FIFO 中存储，支持左右声道发送数据合并成一个 32bit 数据写入发送 FIFO，从而提高 FIFO 的缓冲能力，扩展模块不支持该合并功能。
- 接收通道支持高位符号扩展。
- 支持发送和接收左右声道选择信号共用同一个信号，方便与 4 线 codec 的连接。

PCM 模式特性如下：

- 支持 8，16 位传输工作模式；
- 扩展模块支持 2/4/8/16 路接收，传输工作模式为 8、16 位。
- 只支持短帧同步模式，支持标准时序模式和自定义时序模式。
- 接收通道和发送通道具有独立的 FIFO，深度为 16，宽度为 32bit；
- 支持 FIFO disable 功能，在 FIFO disable 状态下，接收和发送数据均不经过 FIFO，直接在一个 buffer 中缓存；
- 支持 TX 和 RX 通道单独使能，如果某个通道不使能，则该通道的控制单元及数据存储单元都不会翻转，以节省功耗；
- 接收通道支持高位符号扩展。

SIO 模块还提供了 CPU/DSP 访问接口，其特性如下：

- CPU/DSP 可以通过 SIO 模块提供的 AHB Slave（AMBA 2.0）接口对 SIO 进行访问；
- SIO AHB 接口只支持 32bit 操作；
- SIO AHB 接口只支持 OK 响应，不支持 ERROR、Retry、Split 响应。
- SIO AHB 接口支持各种 burst 操作；
- SIO 支持 Burst 模式的 DMA 操作；
- SIO 支持左右声道发送数据共用同一个发送地址；支持左右声道接收数据共用同一个接收地址。

说明

- SIO 这里所说的主从模式，主要是时钟 BCLK 以及采样率 ADWS 的来源不同。主模式是 HI3630V100 的 AP 侧产生时钟与采样率；从模式是 AUDIO_CODEC 侧产生时钟与采样率。这个概念和 TX，RX 并没有必然关系。
- PCM 模式下如果数据宽度依然是 24bit 或是 32bit，但是 SIO 的配置只能最大配置为 16bit。因此 SIO 的传输也是截取高 16bit 传输。

6 存储控制

6.1 概述

Hi3660 芯片的主要存储控制模块如下：

- UFS 控制器

  UFS 控制器用于对接 UFS 器件。协议版本支持 UFS2.1，Unipro 1.6，M-PHY 3.1. 支持 2Lane TX+RX 通道；支持 PWM G1~G4、HS G1~3 RateA/B 的速率模式。支持从 UFS 启动，以及将 UFS 用作主要的非易失性存储的需求。

- eMMC 控制器

  eMMC 控制器支持 eMMC5.1 协议，可以控制 8bit eMMC5.1 器件。与外接的 eMMC 器件一起支持系统启动和系统主要的非易失性存储需求。

- SD 卡控制器

  SD 卡控制器支持 SD 3.0 协议，可向下兼容 SD 2.0 协议，支持外接符合 Default-Speed、High-Speed、UHS-I 规范的 SD 卡。用于扩展系统非易失性存储需求。

  DDR 控制器 DDR 控制器支持 4 通道 LPDDR4 器件，每个通道最高支持两个 Rank。作为系统主要动态存储器可提供最高 4GByte 的动态存储空间和最高 21.3GByte/s 的理论访问带宽。

6.2 存储方案

Hi3660 芯片仅支持 2 种的存储方案，如表 6-1 所示。

表6-1 存储方案

序号	方案描述
1	四通道 LPDDR4+ UFS +SD 卡
2	四通道 LPDDR4 + eMMC + SD 卡

LPDDR4 支持对称四通道 LPDDR4 器件，每个通道数据位宽 16bit，每个通道最高支持 2 个片选，器件接口最高工作频率 DDR 1866MHz

UFS 支持 2Lane TX+RX 通道；支持 PWM G1~G4、HS G1~3 RateA/B 的速率模式。

eMMC 支持 8bit 数据位宽，最高工作频率 DDR 200MHz，向下可兼容支持 SDR/DDR 多种频率。

SD 卡支持 4bit 位宽，最高支持 SDR 200MHz 频率，向下可兼容支持 SDR 模式多种频率。

6.3 eMMC

6.3.1 功能描述

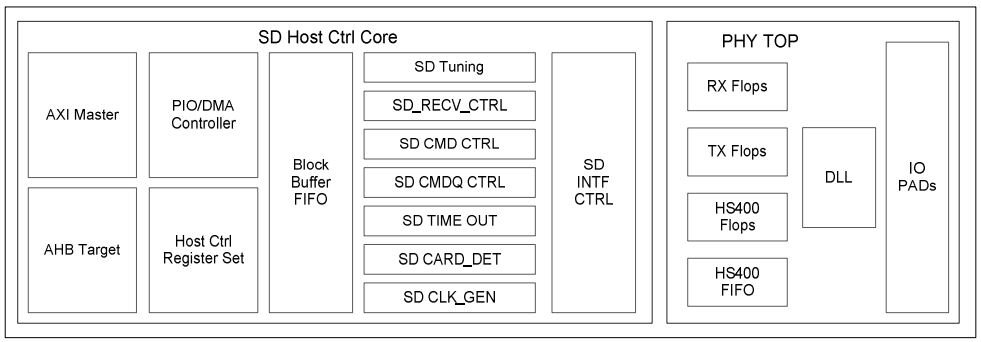
eMMC (Embedded Multi-Media Card)控制器用于处理对 eMMC 器件的命令的收发、数据读写等操作。

eMMC 控制器模块支持以下功能特性：

- JEDEC eMMC5.1 协议
- Enhanced strobe
- Command Queue
- Auto-tuning 功能
- SDMA/ADMA 方式的 DMA 传输
- 命令、数据的 CRC 校验

eMMC 控制器模块功能框图如图 6-1 所示。

图6-1 eMMC 控制器模块功能框图



eMMC 控制器模块主要由以下几个单元功能如下：

- Host Interface:

  包括 AHB target 和 AXI Master。AHB target 用于对控制器寄存器的配置，以及对 FIFO 直接读取和写入数据。AXI Master 用于 DMA 的数据读写。
- Host Controller Register Set

  用于对寄存器的配置，支持 Byte 操作访问。也用于 PIO 方式对 FIFO 数据的读写。
- PIO/DMA Controller

DMA Controller 实现 ADMA 和 SDMA 的功能，用于在读写 eMMC 器件时使用 DMA 方式搬运数据，以减少读写过程中 CPU 的干预，也可通过 PIO 方式直接从 FIFO 读写数据。

- CQ Controller

  实现 Command Queue 功能，用于对任务的发送、查询和执行等的任务管理。

- Block Buffer

  一个双口的数据读写接口，实现总线时钟域和卡时钟域的数据传输中转。

- Clock Generator

  用于时钟的分频等。

- SD Tuning Control

  实现 eMMC 自动的 tuning 流程，实现 tuning 命令的发送、数据校验以及相位的选择。

- PHY TOP

  包括接口数字电路、模拟 DLL 和 IO。模拟 DLL 包括 TX_DLL、RX_DLL 和 STROBE_DLL，分别实现输出时钟 tx_clk、采样时钟 rx_clk 和 strobe_90 时钟。

6.3.2 寄存器描述

eMMC IP 手册详见(https://arasan.com/products/emmc51.html)

6.4 SD/SDIO

6.4.1 功能描述

6.4.1.1.1 功能框图

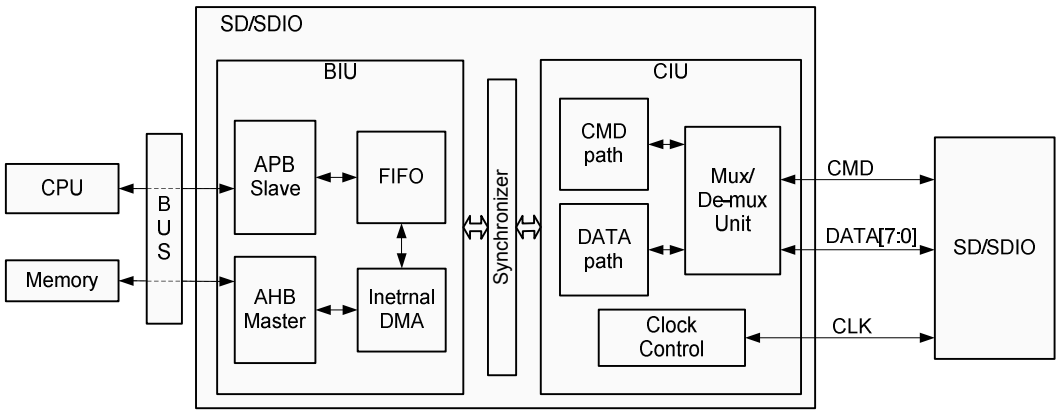
SD/SDIO 控制器用于处理对 SD 卡的读写等操作，并通过 SDIO 协议实现对扩展外设（如 WiFi）的支持。Hi3660 提供 SD 和 SDIO0 等 2 个 SD/SDIO 控制器，分别用于控制 SD 卡、使用 SDIO 接口的 WiFi 等。

SD/SDIO 控制符合以下协议的设备：

- Secure Digital memory (SD mem-version 3.0，兼容 2.0、1.1)；
- Secure Digital I/O (SDIO - version 3.0，兼容 2.0)。

SD/SDIO 的功能框图如图 6-2 所示。

图6-2 SD/SDIO 功能框图



SD/SDIO 控制器通过内部总线与系统连接，由以下单元构成：

- 总线接口模块：提供 AMBA AHB Master 和 APB Slave 接口，通过 APB Slave 接口配置寄存器和读写 FIFO，也可通过 AHB Master 接口，由内部 DMA 控制读写 FIFO 中的数据。
- 卡接口模块：处理协议相关内容和时钟的处理。
  - 命令通道：完成指令的发送与响应的接收；
  - 数据通道：配合命令通道完成数据读写操作；
  - 编码解码单元：对输入输出数据根据协议进行编码译码操作；
  - 接口时钟控制单元：控制接口时钟的关闭与开启，也可根据需要将 cclk_in 时钟分频输出，作为器件的工作时钟。

SD/SDIO 控制器的功能特点有：

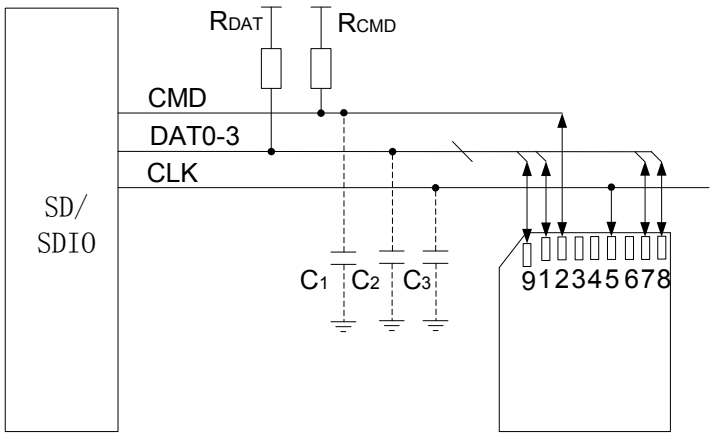
- 支持内部 DMA 数据传输方式。
- FIFO 深度为 256，宽度 32bit，支持 FIFO 阈值可配，DMA 传输时 Burst 大小可配。
- 支持 FIFO 上溢出与下溢出中断告警，防止数据传输错误。
- 支持命令与数据的 CRC 生成与校验。
- 接口时钟频率可编配置。
- 支持低功耗模式，关闭 SD/SDIO 控制器时钟和接口时钟。
- 支持位宽为 1bit 和 4bit 的数据传输和 SDIO 中断检测
- 支持 1byte～512byte 的块数据读写操作。
- 支持 SDIO 的 suspend 操作、resume 操作和 read wait 操作。
- 不支持 DDR50 模式。

6.4.1.1.2 典型应用

以 SD 卡为例，SD/SDIO 控制器的典型应用电路如图 6-3 所示。SD/SDIO 控制器典型应用电路图

SD/SDIO 控制器通过 1 根时钟信号线、1 根双向指令信号线和 4 根双向数据信号线与 SD 卡设备对接来完成命令与数据的交互。指令信号、数据信号均工作在上拉模式。

图6-3 SD/SDIO 控制器典型应用电路图



注意

除图 6-3 中信号线外，卡槽一般还提供机械写保护信号和卡检测信号。系统可借助 GPIO 检测这 2 根信号线上的电平，完成机械写保护和热插拔时的卡检测功能。

6.4.2 寄存器描述

SD/SDIO IP 手册详见(https://www.synopsys.com/dw/ipdir.php?ds=dwc_sd_emmc_host_controller)，其中名称 dwc_mobile_storage 版本 2.70a （最新的是 2.90a，2.70a 已经 unavailable）

6.5 UFS

6.5.1 功能描述

UFS（Universal Flash Storage）控制器用于处理对 UFS 大容量存储器件的命令的收发，数据读写等操作。UFS 是一种简单的，高性能的，支持大存储介质的串行接口，主要用于手机系统。

UFS 控制器模块支持以下功能特性：

- JEDEC UFS2.0 协议
  - Higher speed up to HS-G3 (High-Speed Gear 3)
  - Symmetric 2RX-2TX lanes, support two lanes
  - Auto-hibernate entry and exit sequence
- JEDEC UFSHCI2.1 协议
  - Support up to 32 task requests

  - Support up to 8 task management requests
  - Support to pre-fetch more than one PRD entry (up to 16 PRD entries)
  - Clock gating ready design
  - Inline Encryption (IE)
- MIPI UniPro1.6 以及 M-PHY3.1 协议
  - SKIP symbol insertion
  - Scrambling for EMI mitigation
  - HS-Gear3 adaption
  - Advanced granularity support

6.5.2 寄存器描述

SD/SDIO IP 手册详见(https://www.synopsys.com/dw/ipdir.php?ds=ufs)，名称 dwc_ufs_crypto_host_controller 版本 1.00a

7 接口控制

7.1 USB3OTG

7.1.1 功能描述

功能特性

Hi3660 的 USB 3.0 OTG 包括 USB3.0 控制器和 USB3.0 femtoPHY，支持 Host 和 Device 功能。作为 Device，用于对接 PC 或者其它 USB Host。作为 Host，用于对接 U 盘或者其它 USB device。

USB 模块具有以下功能特点：

- 完全兼容 USB 3.0 协议
- 集成 USB 控制器和 USB3 femtoPHY
- Host 模式支持 Super-Speed、High-speed、Full-speed 和 low speed
- Device 模式支持 Super-Speed、High-speed 和 Full-speed
- 支持 LPM（Link Power Management）协议
- 作为 Device 最大支持 16 个 IN 端点和 16 个 OUT 端点
- 端点 0 为控制端点，端点 1~端点 15 类型可配置，可配置为 Bulk、Isochronous、Interrupt 3 种端点类型
- 每个端点有独立的发送 FIFO，FIFO 大小动态可配
- 作为 Host 完全兼容 xHCI 1.0 协议
- 内置 DMA，支持 scatter/gather
- 工作在 Super-Speed 模式时，控制器和 PHY 之间的接口为 PIPE，时钟频率为 125MHz
- 工作在 High-speed 和 Full-speed 模式时，控制器和 PHY 之间的接口为 UTMI，时钟频率为 60 MHz
- 支持 BC1.2 协议（ACA 除外）

7.1.1 寄存器描述

详见 synopsys　IP 手册(https://www.synopsys.com/cn/IP/InterfaceIP/USB/Pages/default.aspx)

7.2 UART

7.2.1 功能描述

7.2.1.1.1 功能特性

Hi3660 的 USB 3.0 OTG 包括 USB3.0 控制器和 USB3.0 femtoPHY，支持 Host 和 Device 功能。作为 Device，用于对接 PC 或者其它 USB Host。作为 Host，用于对接 U 盘或者其它 USB device。

USB 模块具有以下功能特点：

- 完全兼容 USB 3.0 协议
- 集成 USB 控制器和 USB3 femtoPHY
- Host 模式支持 Super-Speed、High-speed、Full-speed 和 low speed
- Device 模式支持 Super-Speed、High-speed 和 Full-speed
- 支持 LPM（Link Power Management）协议
- 作为 Device 最大支持 16 个 IN 端点和 16 个 OUT 端点
- 端点 0 为控制端点，端点 1~端点 15 类型可配置，可配置为 Bulk、Isochronous、Interrupt 3 种端点类型
- 每个端点有独立的发送 FIFO，FIFO 大小动态可配
- 作为 Host 完全兼容 xHCI 1.0 协议
- 内置 DMA，支持 scatter/gather
- 工作在 Super-Speed 模式时，控制器和 PHY 之间的接口为 PIPE，时钟频率为 125MHz
- 工作在 High-speed 和 Full-speed 模式时，控制器和 PHY 之间的接口为 UTMI，时钟频率为 60 MHz
- 支持 BC1.2 协议（ACA 除外）

7.2.2 寄存器描述

详见 synopsys　IP 手册(https://www.synopsys.com/cn/IP/InterfaceIP/USB/Pages/default.aspx)

7.3 UART

7.3.1 功能描述

7.3.1.1.1 功能特性

UART 是 Universal Asynchronous Receiver/Transmitter 的缩写，即通用异步接收发送器。UART 完成接收数据的串并转换和发送数据的并串转换。

Hi3660 集成了 9 个 UART，对外提供 9 个 UART 接口。除了 UART7，其余的都支持流控。这里的 UART 不包含 modem 调试用 UART。

其中 UART6 最高支持 1.2M 波特率，其余的 UART 最高支持 9M 波特率。

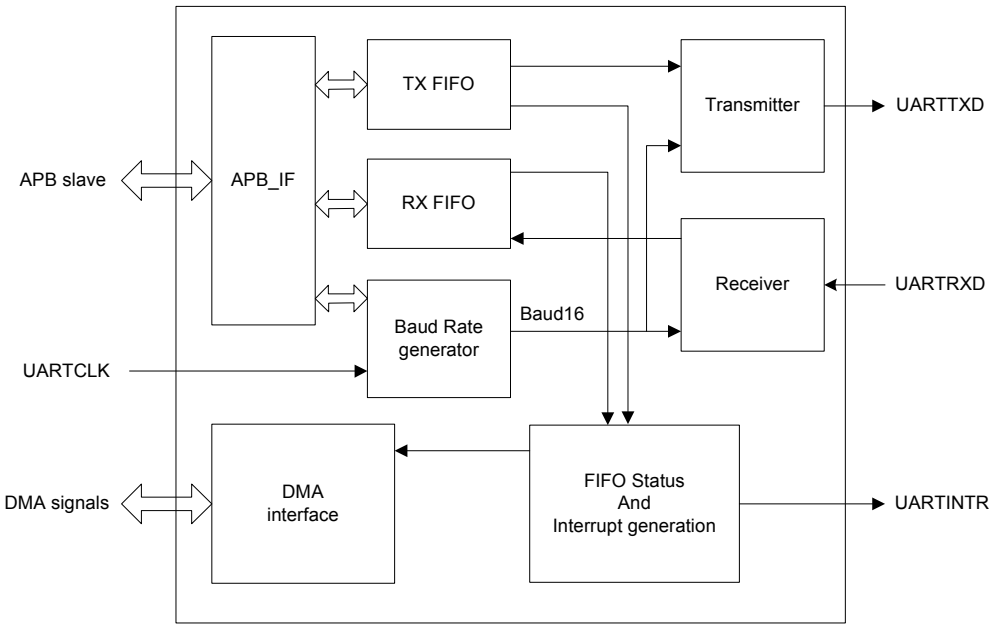
UART 模块支持以下功能特性：

- 数据位和停止位位宽可配。
- 数据位可配置为 5/6/7/8bit
- 停止位可配置为 1/2bit
- 支持奇、偶校验方式或者无校验位。
- 传输速率编程可配，最大支持 9Mbit/s。
- 支持 DMA 数据搬运方式。（UART7 不支持 DMA）
- 支持接收 FIFO 中断、发送 FIFO 中断、接收超时中断和错误中断。
- UART 发送 FIFO 深度为 64bit，宽度为 8bit；接收 FIFO 深度为 64bit，宽度为 12bit。（UART7/8 发送/接收 FIFO 深度为 16bit，其余的为 64bit）。
- 支持 IrDA 串行红外模式。

7.3.1.1.2 逻辑框图

模块逻辑框图如图 7-1 所示。

图7-1 UART 模块逻辑图



UART 模块主要有 8 个单元组成。各单元功能/工作原理如下：

- APB Interface：APB slave 接口，处理总线的读写数据，配置寄存器等；
- TX FIFO：发送 FIFO，用于存储发送的数据；
- RX FIFO：接收 FIFO，用于存储接收的数据；

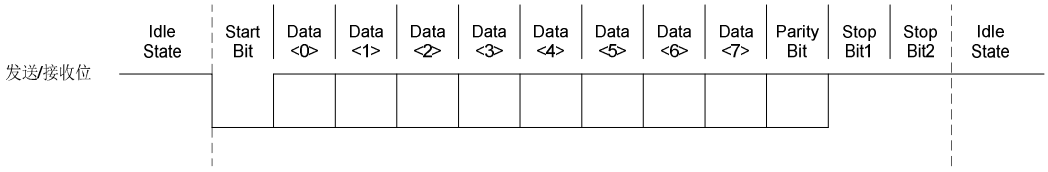
- Baud Rate generator：波特率产生器，用于产生配置的 baud16 时钟；
- Transmitter：将数据进行并串转换并输出，支持将数据译码成红外格式输出；
- Reciever：对接收数据进行串并转换，支持对红外格式的数据进行解码；
- DMA Interface：负责 DMA 接口处理；
- FIFO Status And Interrupt generation：对 FIFO 状态进行监控，并根据配置的中断 FIFO level 产生中断。

7.3.1.1.3 帧格式

UART 的一次帧传输包括起始位、数据位、校验位和结束位，如图 7-2 所示。

发送或接收的数据为 NRZ 码。数据帧从某一 UART 的 TXD 端输出，由 UART 的 RXD 端输入。

图7-2 UART 帧格式



帧格式中各域含义如下：

- 起始位（start bit）

  一个数据帧开始的标志。UART 协议规定发送信号出现一个低电平就表示一个数据帧的开始。在 UART 不传输数据时，应该保持为“1”。

- 数据位（data bit）

  数据位宽根据应用要求进行调整，可以配置成 5/6/7/8bit。

- 校验位（parity bit）

  校验位是 1bit 纠错信号。UART 的校验位有奇校验、偶校验和固定校验位，同时支持校验位的使能和禁止。

- 结束位（stop bit）

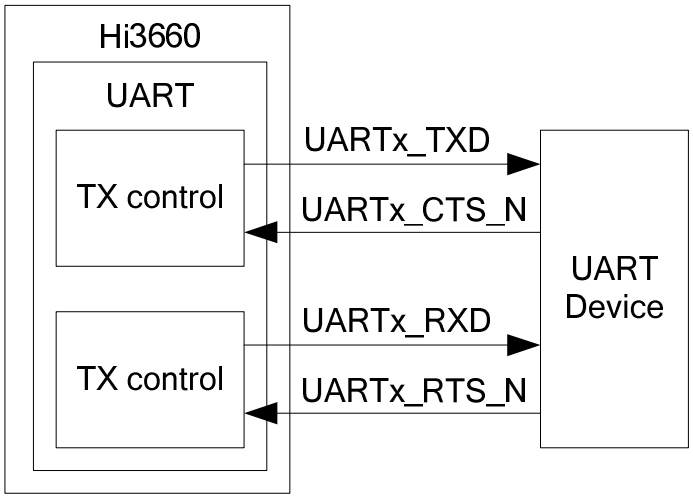
  结束信号即数据帧的停止位，支持 1bit 和 2bit 的停止位。数据帧的结束信号就是把发送信号拉成“1”。

7.3.1.1.4 典型应用场景

UART 用于和芯片外部支持 UART 协议的器件或接口之间收发数据，完成接收数据的串并转换和发送数据的并串转换。也可以发符合 IrDA 协议的红外数据。

UART 模块典型的应用场景如图 7-3 所示。

图7-3 UART 应用场景



UART 支持如下工作模式：

- UART：支持 UART 协议的数据收发，支持流控；
- 红外模式：支持 IrDA 协议的串行红外收发数据。

7.3.2 信号描述

UART 信号框图如图 7-4 所示。

图7-4 UARTx 信号框图

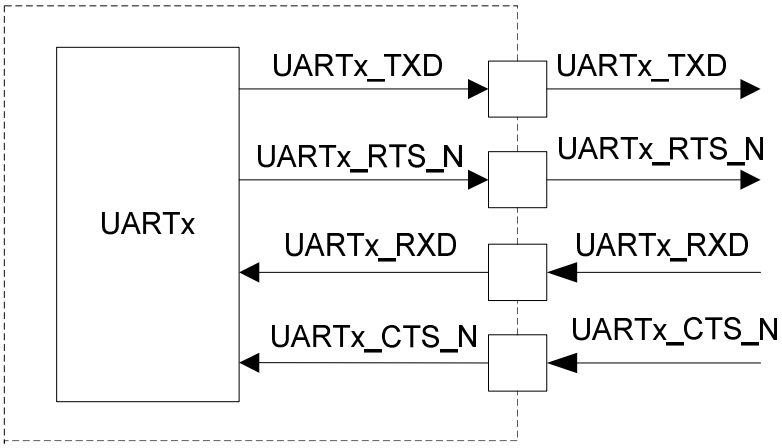


表7-1 UART 接口信号表

信号名称	方向	描述
UARTx_TXD	O	UARTx 发送数据信号。
UARTx_RXD	I	UARTx 接收数据信号。

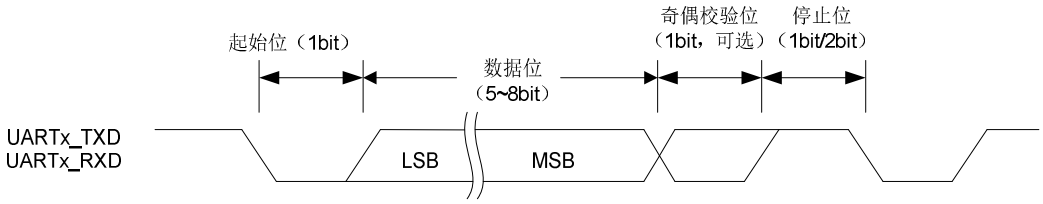
信号名称	方向	描述
UARTx_RTS_N	O	UARTx request to send 信号。
UARTx_CTS_N	I	UARTx clear to send 信号。

7.3.3 时序

7.3.3.1.1 UART 时序

UART 时序图如图 7-5 所示。

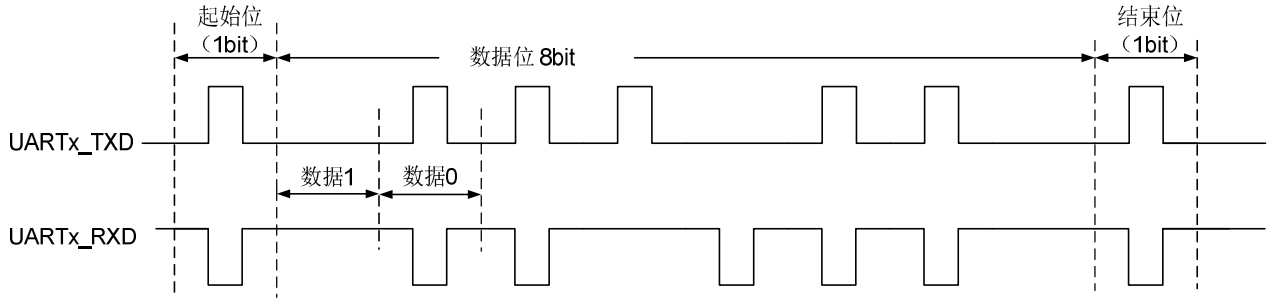
图7-5 UART 时序图



7.3.3.1.2 红外模式时序

UART 红外模式下的时序图如图 7-6 所示。

图7-6 UART 红外模式时序图



7.3.4 寄存器描述

详见 ARM 公版 pl011 IP 手册
(http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0183g/index.html)

7.4 SPI

7.4.1 功能描述

7.4.1.1.1 功能特性

SPI 实现数据的串并联传输，在 Hi3660 中 SPI 作为 Master 与外部设备进行同步串行通信，不作为 slave 用。

Hi3660 芯片中提供五个 SPI 接口：SPI0、SPI1、SPI2、SPI3、SPI4。其中，SPI0 暂时保留，SPI1 有一个片选，主要用于连接 ESE；SPI2 有 4 个片选：SPI2_CS0 连接指纹传感器，SPI2_CS1/2/3 保留；SPI3 有 4 个片选：SPI3_CS0 连接 MINI_ISP，SPI3_CS1 连接墨水屏，SPI3_CS3/4 保留；SPI4 有 4 个片选：SPI4_CS0 连接指纹传感器，SPI4_CS1/2/3 保留。

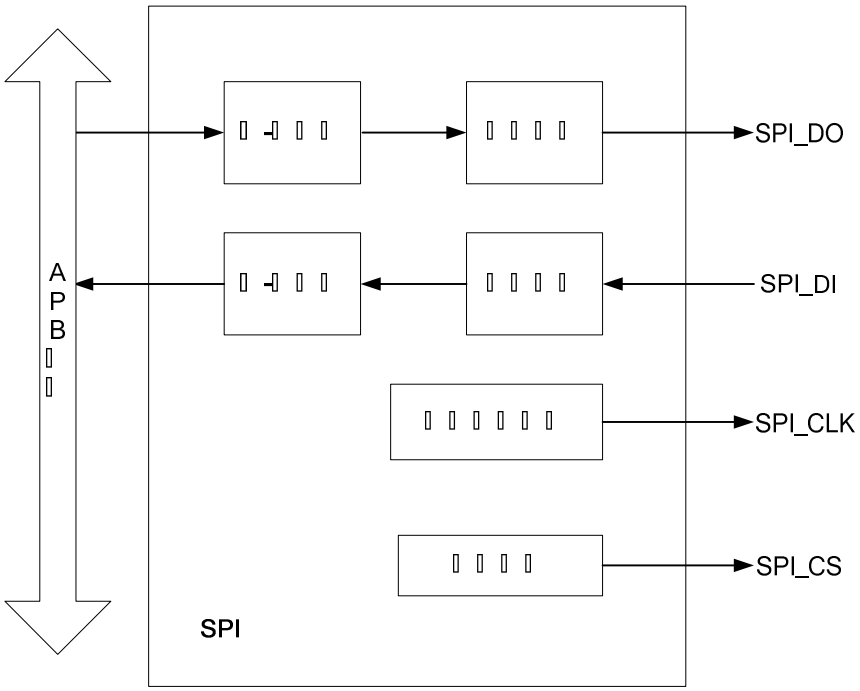
SPI 模块支持以下特性：

- 支持接口时钟频率可编程。
- 收/发分开的宽度 16bit、深度 256bit 的 FIFO（发送和接收 FIFO 各 1 个）。
- 支持 SPI 帧格式。
- 串行数据帧长度可编程：4bit~16bit。
- 支持接收、发送 FIFO 请求中断水线可编程。
- 支持接收、发送 FIFO 请求 DMA 进行 Burst 传输的水线可编程。
- 支持发送 FIFO 中断、接收 FIFO 中断、接收超时中断和接收 FIFO 溢出中断独立屏蔽。
- 内部提供回环测试。
- 支持 DMA 操作。

7.4.1.1.2 逻辑框图

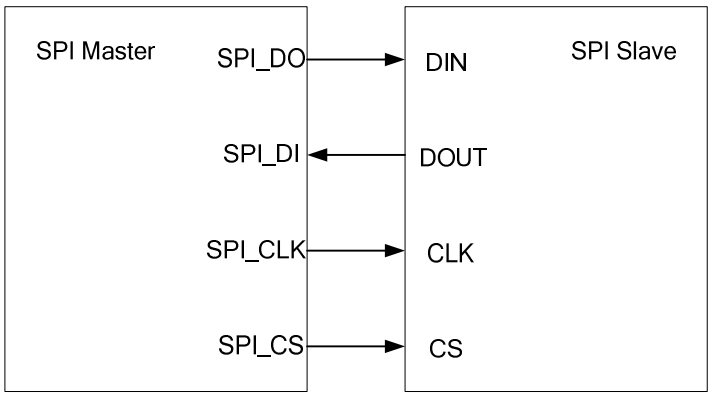
SPI 模块功能框图如图 7-7 所示。

图7-7 SPI 模块功能框图



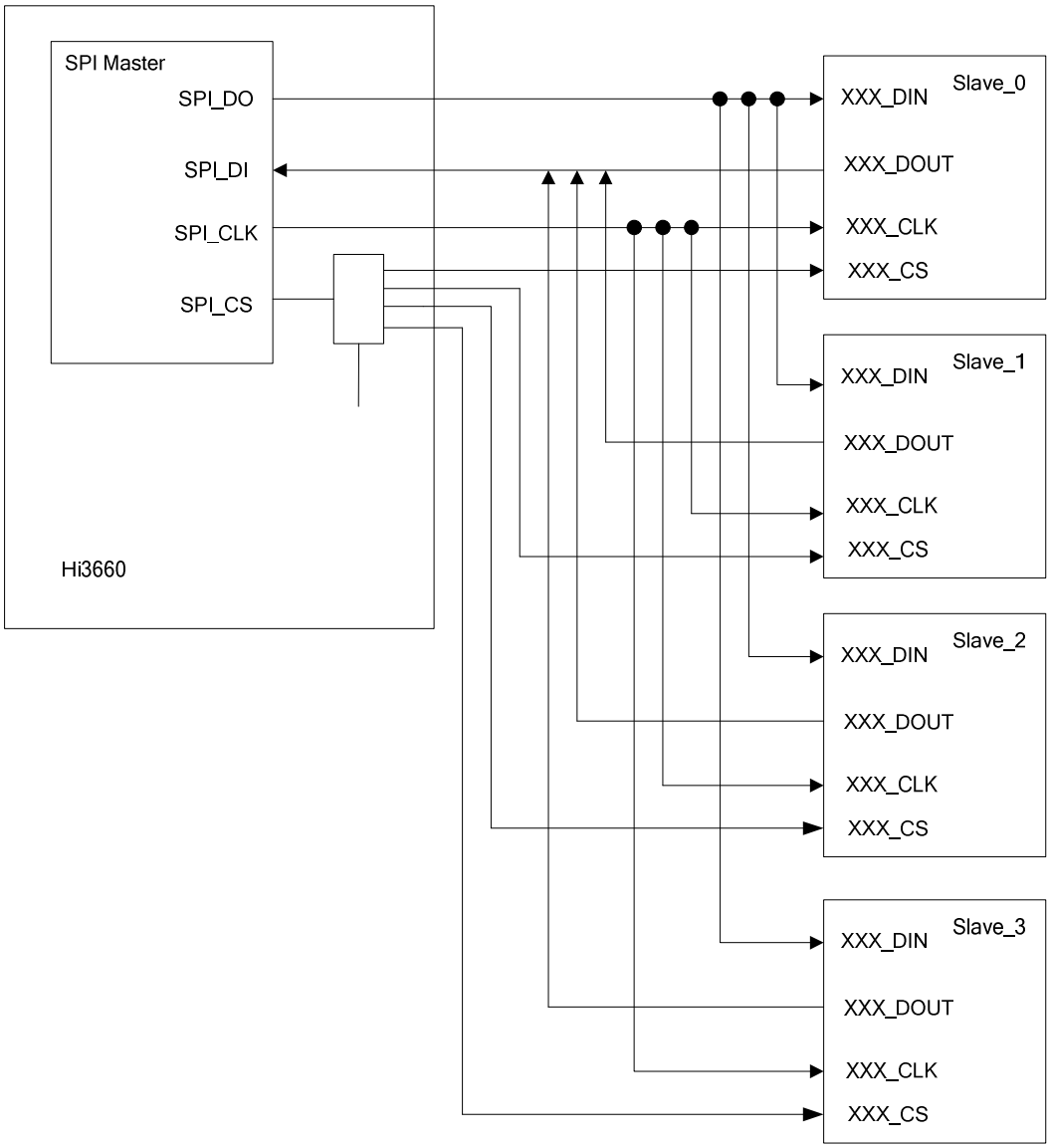
SPI0/1/2/3/4 都可以接单 slave，对应的应用框图如图 7-8 所示。

图7-8 SPI 接单 slave 的应用



SPI0/2/3/4 可以支持接多个 Slave，对应的应用框图如图 7-9 所示。

图7-9 SPI 接多 slave 的应用



7.4.1.1.3 典型应用场景

SPI 的工作模式分为中断或查询方式下的数据传输和 DMA 方式下的数据传输。

7.4.2 中断处理

SPI 有 5 个中断，其中前 4 个是独立中断源、可屏蔽、高电平有效。

- SPIRXINTR
  接收 FIFO 中断请求。当接收 FIFO 中有 4 个或更多的有效数据时，该中断置位。
- SPITXINTR
  发送 FIFO 中断请求。当发送 FIFO 中有 4 个或更少的有效数据时，该中断置位。
- SPIRORINTR

接收 overflow 中断请求。当 FIFO 已满，且又有新的数据需要写入 FIFO 时，会引起 FIFO overflow，该中断置位。此时数据被写入接收移位寄存器，而不是 FIFO。

- SPIRTINTR

  接收 time out 中断请求。当接收 FIFO 非空，且 SPI 处于 idle 态超过一个固定的 32bit 周期，该中断置位。

  此时表明接收 FIFO 中仍有数据需要传输。如果接收 FIFO 被读空或者当有新的数据被接收到 SPIRXD 中，该中断解除置位。也可以通过写寄存器 SPIICR[RTIC]清除该中断。

- SPIINTR

  组合中断，为以上 4 个中断经过“或”运算后的结果。如果上述 4 个独立中断中任意一个置位且使能，该中断置位。

SPI0/SPI1/SPI2/SPI3/SPI4 到 CPU 的中断号分别对应 145、112、148、344、345，不支持到 LPM3 的中断注册。SPI1/SPI3/SPI4 不支持到 IOM7 的中断注册，SPI0/SPI2 到 IOM7 的中断号分别是 1 和 26。

7.4.3 信号描述

SPI0 接口信号如表 7-2 所示。

表7-2 SPI0 接口信号表

信号名称	方向	含义
SPI0_CLK	Output	SPI 时钟，Master mode 输出。
SPI0_CS0_N	Output	Master mode SPI 片选 0。
SPI0_CS1_N	Output	Master mode SPI 片选 1。
SPI0_CS2_N	Output	Master mode SPI 片选 2。
SPI0_CS3_N	Output	Master mode SPI 片选 3。
SPI0_DI	Input	SPI 接收数据输入。
SPI0_DO	Output	SPI 发送数据输出。

SPI1 接口信号如表 7-3 所示。

表7-3 SPI1 接口信号表

信号名称	方向	含义
SPI1_CLK	Output	SPI 时钟，Master mode 输出。
SPI1_CS_N	Output	Master mode SPI 片选 0。
SPI1_DI	Input	SPI 接收数据输入。
SPI1_DO	Output	SPI 发送数据输出。

SPI0 接口信号如表 7-4 所示。

表7-4 SPI2 接口信号表

信号名称	方向	含义
SPI2_CLK	Output	SPI 时钟，Master mode 输出。
SPI2_CS0_N	Output	Master mode SPI 片选 0。
SPI2_CS1_N	Output	Master mode SPI 片选 1。
SPI2_CS2_N	Output	Master mode SPI 片选 2。
SPI2_CS3_N	Output	Master mode SPI 片选 3。
SPI2_DI	Input	SPI 接收数据输入。
SPI2_DO	Output	SPI 发送数据输出。

SPI0 接口信号如表 7-5 所示。

表7-5 SPI3 接口信号表

信号名称	方向	含义
SPI3_CLK	Output	SPI 时钟，Master mode 输出。
SPI3_CS0_N	Output	Master mode SPI 片选 0。
SPI3_CS1_N	Output	Master mode SPI 片选 1。
SPI3_CS2_N	Output	Master mode SPI 片选 2。
SPI3_CS3_N	Output	Master mode SPI 片选 3。
SPI3_DI	Input	SPI 接收数据输入。
SPI3_DO	Output	SPI 发送数据输出。

SPI4 接口信号如表 7-6 所示。

表7-6 SPI4 接口信号表

信号名称	方向	含义
SPI4_CLK	Output	SPI 时钟，Master mode 输出。
SPI4_CS0_N	Output	Master mode SPI 片选 0。
SPI4_CS1_N	Output	Master mode SPI 片选 1。

SPI4_CS2_N	Output	Master mode SPI 片选 2。
SPI4_CS3_N	Output	Master mode SPI 片选 3。
SPI4_DI	Input	SPI 接收数据输入。
SPI4_DO	Output	SPI 发送数据输出。

7.4.4 时序与参数

图 7-10～图 7-17 中的缩略语含义为：

- MSB：Most Significant Bit
- LSB：Least Significant Bit
- Q：Q is an undefined signal

7.4.4.1.1 SPI 帧格式

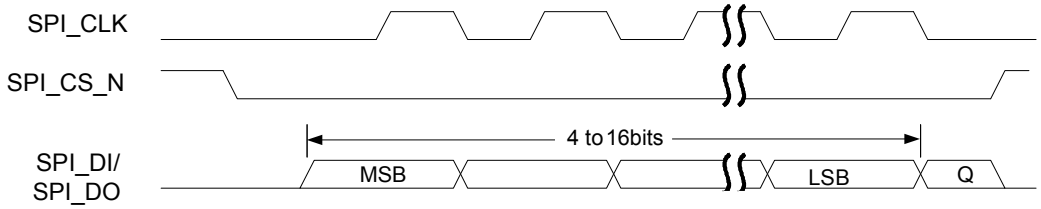
说明

SPO 表示 SPICLKOUT 极性，SPH 表示 SPICLKOUT 相位。它们是寄存器 SPICR0 bit[7:6]。

（1）SPO=0、SPH=0

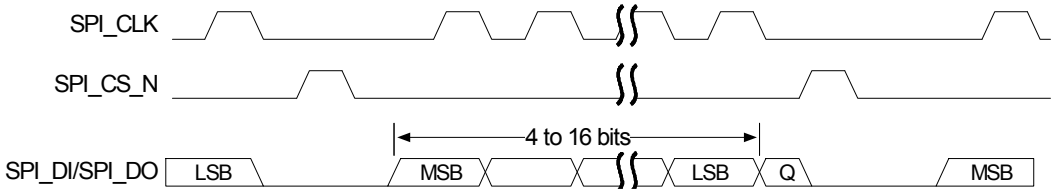
SPI 单帧帧格式如图 7-10 所示。

图7-10 SPI 单帧帧格式（SPO=0、SPH=0）



SPI 连续帧帧格式如图 7-11 所示。

图7-11 SPI 连续帧帧格式（SPO=0、SPH=0）



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为低
- SPI_CS_N 信号设置为高
- 发送数据线 SPI_DO 强制为低

当 SPI 处于使能状态，而且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低表示开始传输数据。此时使能 Slave 的数据放入 Master 的输入 SPI_DI。半个 SPI_CLK 时钟周期之后，有效的 Master 数据传输到 SPI_DO。此时 Master 和 Slave 数据都已经有效，SPI_CLK 管脚在接下来的半个 SPI_CLK 时钟周期之后变为高电平。数据在 SPI_CLK 时钟的上升沿被捕获，在时钟的下降沿被传送。

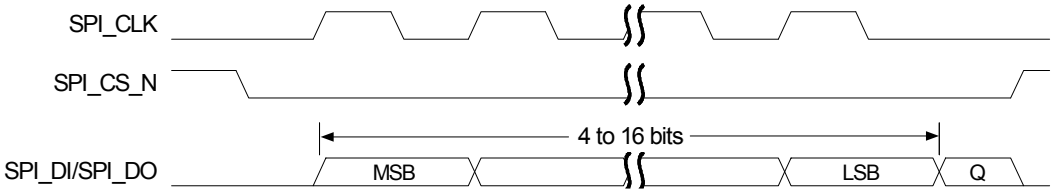
如果传输单个 word，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

如果是连续的传输，SPI_CS_N 信号在每个 word 传输之间必须将 SPI_CLK 时钟拉高一个时钟周期。这是因为 SPH 为 0 时，Salve 选择管脚会固定其内部串行设备寄存器的数据，使它不会变化。因此在连续传输时，主设备必须在每个 word 传输之间将 SPI_CS_N 信号拉高。连续传输结束时，SPI_CS_N 在捕获到最后 1bit 之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

（2）SPO=0、SPH=1

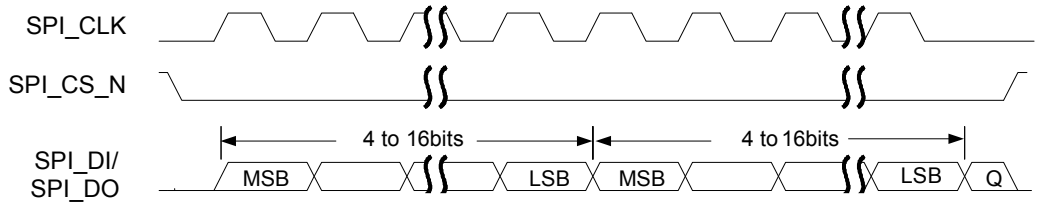
SPI 单帧帧格式如图 7-12 所示。

图7-12 SPI 单帧帧格式（SPO=0、SPH=1）



SPI 连续帧帧格式如图 7-13 所示。

图7-13 SPI 连续帧帧格式（SPO=0、SPH=1）



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为低
- SPI_CS_N 设置为高
- 发送数据线 SPI_DO 强制为低

当 SPI 为使能状态，而且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低表示开始传输数据。半个 SPI_CLK 时钟周期之后，Master 和 Slave 的有效数据分别在各自的传输线上有效。同时，SPI_CLK 从第一个上升沿开始有效。数据在 SPI_CLK 时钟的下降沿被捕获，在时钟的上升沿被传送。

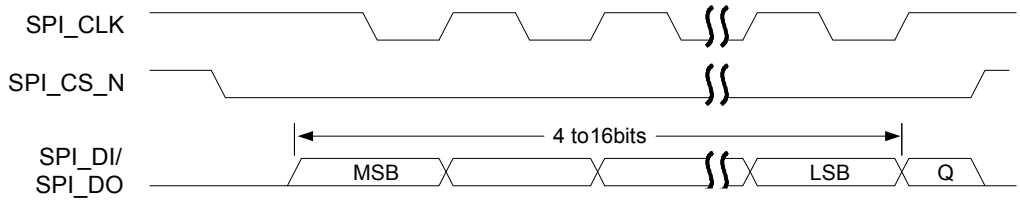
如果传输单个 word，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟之后恢复为高电平。

当连续传输时，在传输数据 word 之间 SPI_CS_N 保持为低。连续传输结束时，SP_CS_N 在最后 1bit 捕获之后的 1 个 SPI_CLK 时钟之后恢复为高电平。

（3）SPO=1、SPH=0

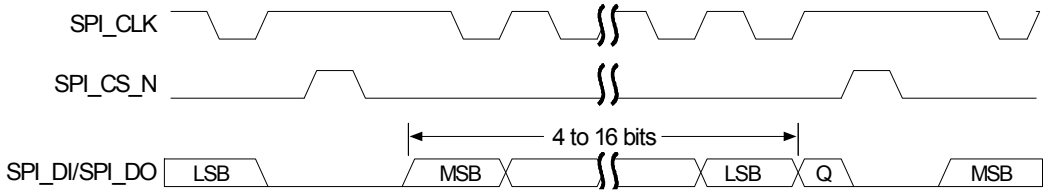
SPI 单帧帧格式如图 7-14 所示。

图7-14 SPI 单帧帧格式（SPO=1、SPH=0）



SPI 连续帧帧格式如图 7-15 所示。

图7-15 SPI 连续帧帧格式（SPO=1、SPH=0）



在该配置下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为高
- SPI_CS_N 信号设置为高
- 发送数据线 SPI_DO 强制为低

当 SPI 为使能状态，而且发送 FIFO 内有有效数据时，设置 SPI_CS_N 信号为低表示开始传输数据。此时 Slave 的数据立刻发送到 Master 的接收数据线 SPI_DI。半个 SPI_CLK 周期之后，Master 的有效数据传送到 SPI_DO。再过半个 SPI_CLK 时钟周期之后，SPI_CLK Master 管脚设置为低。这表示数据在 SPI_CLK 时钟的下降沿被捕获，在 SPI_CLK 时钟的上升沿被传送。

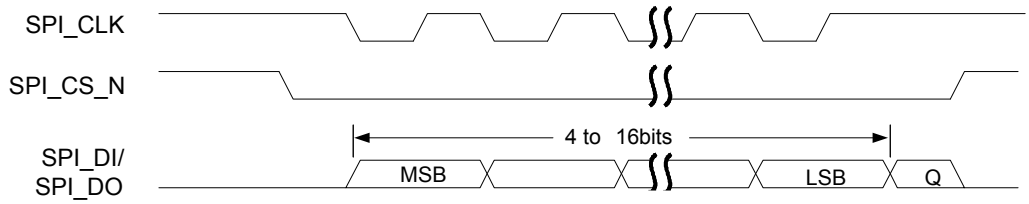
如果传输单个 word，当捕捉到最后 1bit 数据时，SPI_CS_N 在接下来的 1 个 SPI_CLK 时钟之后恢复为高电平。

如果是连续的传输，SPI_CS_N 信号在每个 word 传输之间必须拉高。这是因为当 SPH 为 0 时，Salve 选择管脚固定其内部串行设备寄存器的数据，使它不会变化。SPI_CS_N 在捕获到最后 1bit 数据之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

（4）SPO=1、SPH=1

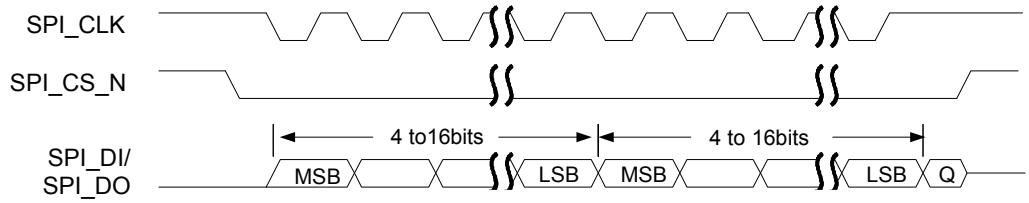
SPI 单帧帧格式如图 7-16 所示。

图7-16 SPI 单帧帧格式（SPO=1、SPH=1）



SPI 连续帧帧格式如图 7-17 所示。

图7-17 SPI 连续帧帧格式（SPO=1、SPH=1）



在该模式下，当 SPI 处于空闲状态时：

- SPI_CLK 信号设置为高
- SPI_CS_N 信号设置为高
- 发送数据线 SPI_DO 强制为低

当 SPI 为使能状态，而且发送 FIFO 内有有效数据时，设置 SPI_CS_N Master 信号为低表示开始传输数据。半个 SPI_CLK 时钟周期后，Master 和 Slave 数据在各自的传输线上有效。同时，时钟 SPI_CLK 从 1 个下降沿开始有效。数据在 SPI_CLK 时钟的上升沿被捕获，在时钟的下降沿被传送。

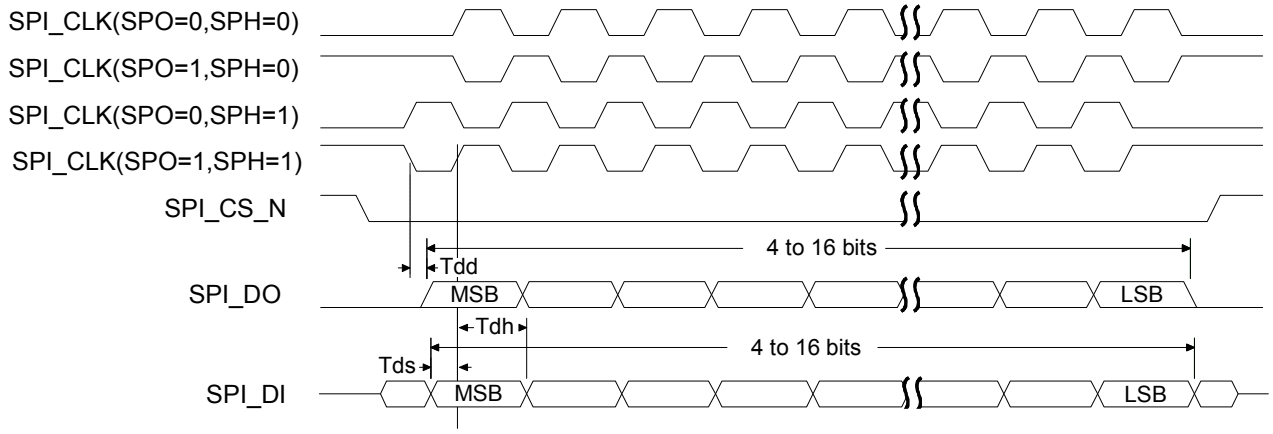
当传输单个 word 时，SPI_CS_N 在传输的最后 1bit 捕获之后的 1 个 SPI_CLK 时钟周期之后恢复为高电平。

如果是连续传输，SPI_CS_N 信号始终保持为低。SPI_CS_N 在捕获到最后 1bit 之后的 1 个 SPI_CLK 时钟周期之后恢复到高状态。对于连续传输来说，SPI_CS_N 在传输过程中一直保持为低，结束方式与单个传输方式相同。

7.4.4.1.2 时序

SPI 接口时序图如图 7-18 所示。

图7-18 SPI 接口时序图



SPI 接口输入时序参数如表 7-7 所示。

表7-7 SPI 接口输入时序参数

参数	符号	最小值	最大值	单位
DI Setup Time	Tds	10	-	ns
DI Hold Time	Tdh	10	-	ns

7.4.5 寄存器描述

详见 ARM 公版 pl022 IP 手册
(http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0183g/index.html)

7.5 I2C

7.5.1 功能描述

7.5.1.1.1 功能框图

I^2C 控制器利用两条线（SCL、SDA）和片外具有 I^2C 接口的设备进行通讯。I^2C 接口遵守 I^2C 2.1 版本规范，可完成对 I^2C 总线上从设备的数据发送和接收。在 Hi3660 中，I^2C 控制器只能做主设备。

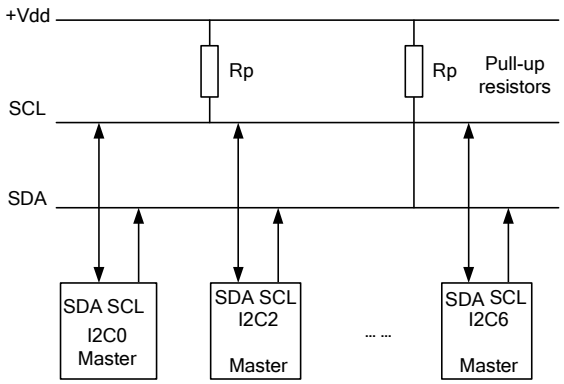
Hi3660 集成了 8 个 I^2C 模块：

- I^2C0：用于连接加速传感器、陀螺仪、气压计等 Sensor。
- I^2C1：I^2C0 备份。

- I²C2：用于连接电容 TP。
- I²C3：用于连接普通 GPIO 。
- I²C4：用于充电、NFC（Near Field Communication）、外置闪光灯驱动、外置 Speaker PA 。
- I²C5：用于连接 A53 大小核 外设 BUCK 电源芯片（位于 PMUI2C 内部）。
- I²C6：保留，备选方案使用。
- I²C7：保留。

I²C 典型应用电路如图 7-19 所示。

图7-19 I²C 典型应用图



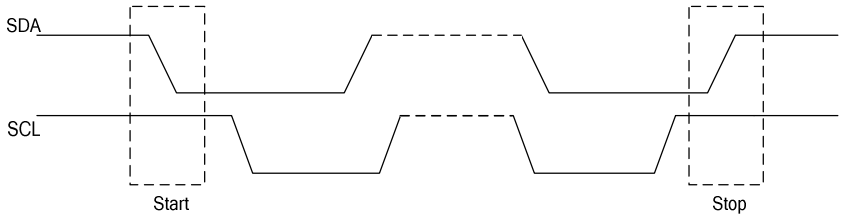
I²C 控制器的功能特点有：

- 支持 2.1 版本的 I²C 总线协议。
- 在 I²C 总线上只作为 Master。
- 在 I²C 总线上作为发送器，主设备向从设备发送数据。
- 作为主设备时支持的从设备地址：标准地址（7bit）和扩展地址（10bit）。
- 支持标准模式 100Kbit/s、快速模式 400Kbit/s 和高速模式 3.4Mbit/s 的数据传输速率。
- 提供发送 FIFO、接收 FIFO，支持 DMA 数据搬运方式。
- 支持中断上报和初始中断状态、屏蔽后中断状态查询。
- 支持 Clock stretching 功能，数据发送期间，当发送 FIFO 数据为空时，拉低 SCL 等待 FIFO 再次填充数据；数据接收期间，当接收 FIFO 数据为空时，拉低 SCL 等待 FIFO 再次填充数据。
- 支持 SDA 数据 Restart 发送，数据总线不释放，在同一通道继续传输。
- 支持 SDA setup/hold time 可寄存器配置。

7.5.1.1.2 Start/Stop 条件

I²C Start/Stop 时序图如图 7-20 所示。

图7-20 I²C Start/Stop 时序图



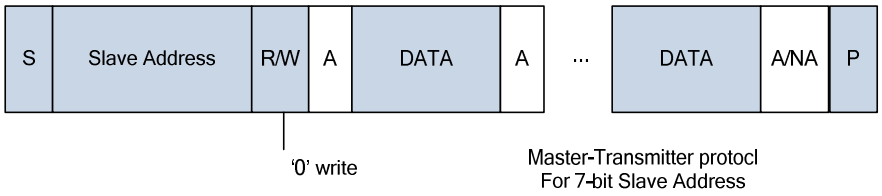
Start：在 SCL 为高电平，SDA 从高电平向低电平转换。

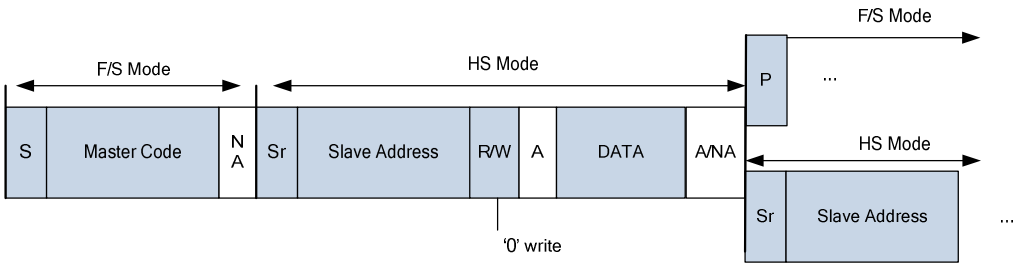
Stop：在 SCL 为高电平，SDA 从低电平向高电平转换。

7.5.1.1.3 I²C 帧格式

I²C 数据传输帧格式图如图 7-21 所示。

图7-21 I²C 数据传输各模式下帧格式图





S：Start Condition　　A：Acknowledge（SDA low）

P：Stop Condition　　NA：No Acknowledge（SDA high）

I²C 标准协议规定的起始信号（Start），由总线上的主设备发出，用来唤醒所有从设备并指示数据传送开始的特殊信号。

主设备工作在 F/S 模式时，在发出起始信号后所发出的第一个数据包为从设备的地址（Slave address+W/R），包括 7bit 从地址（如果是 10bit 从地址，则需发送两个数据包）

和 1bit 读/写数据。当主设备工作在 HS 模式时，主设备要先发送 Master Code，之后帧格式与 F/S 模式相同，区别仅在于速度不同。各从设备依靠 Slave 地址信息来确认是否需要传送或接收数据。当从地址被成功接收后，相应从设备会在第 9 个时钟周期（SCL）将 SDA 拉低，从而反馈给主设备一个应答信号（ACK），然后就可以开始进行后续的数据传输。

数据传送（DATA）指主设备在成功接收到从地址响应信号后按照读、写命令进行相应的数据接收或发送。在数据传输过程中 SDA 只能在 SCL 为低电平时变化，而在 SCL 高电平时保持。同时从设备在每接收到 1byte 后需要给发送设备发出应答信号（ACK）。如果主设备没有成功接收到应答信号，就会终止数据传输或者重新开始发送。

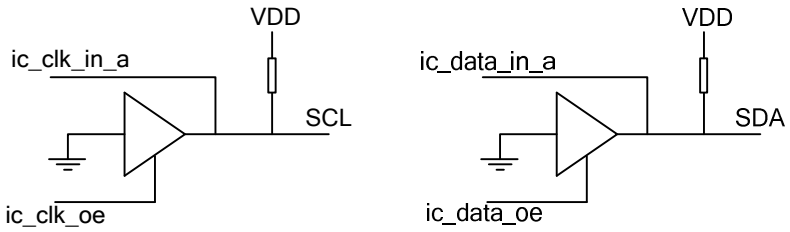
结束信号（Stop）是主设备在当前数据传输完成并且没有新的数据传输需要发起时，所发出的 I²C 标准协议规定的表示数据传输结束的特殊信号。当主设备发出结束信号后从设备必须释放总线。

7.5.2 信号描述

7.5.2.1 接口 OD 门

I²C 接口为 OD 门，如图 7-22 所示。

图7-22 SCL 以及 SDA OD 门输入输出



7.5.2.2 接口信号

I²C 接口信号表如下所示。表 7-8 中的信号复用请参见 IOC 相关文档。

表7-8 I²C 接口信号表

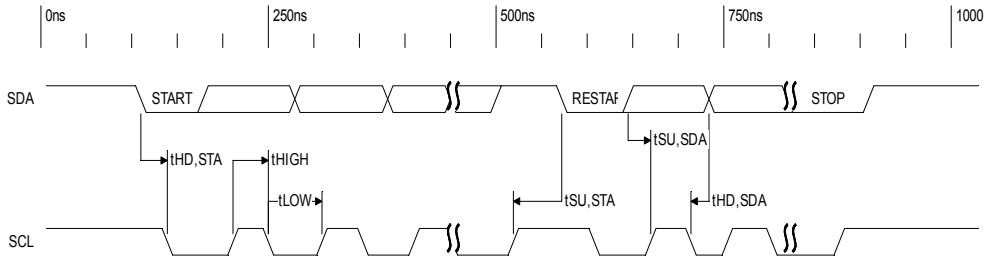
信号名称	方向	含义
I2C0_SDA	Input/Output	I²C0 数据信号。
I2C0_SCL	Output	I²C0 时钟信号。
I2C2_SDA	Input/Output	I²C2 数据信号。
I2C2_SCL	Output	I²C2 时钟信号。
I2C3_SDA	Input/Output	I²C3 数据信号。
I2C3_SCL	Output	I²C3 时钟信号。
I2C4_SDA	Input/Output	I²C4 数据信号。

信号名称	方向	含义
I2C4_SCL	Output	I²C4 时钟信号。
I2C5_SDA	Input/Output	I²C5 数据信号。
I2C5_SCL	Output	I²C5 时钟信号。
I2C6_SDA	Input/Output	I²C6 数据信号。
I2C6_SCL	Output	I²C6 时钟信号。
I2C7_SDA	Input/Output	I²C7 数据信号。
I2C7_SCL	Output	I²C7 时钟信号。

7.5.3 时序与参数

I²C 时序图如图 7-23 所示。

图7-23 I²C 时序图



I²C 时序参数如表 7-9 和表 7-10 所示。

表7-9 I²C 时序参数（F/S 模式）

参数	描述	标准模式		快速模式		单位
		最小值	最大值	最小值	最大值	
fSCL	SCL 时钟频率	0	100	0	400	kHz
tHD;STA	START 条件的保持时间	4.0	-	0.6	-	µs
tLOW	SCL 低电平宽度	4.7	-	1.3	-	µs
tHIGH	SCL 高电平宽度	4.0	-	0.6	-	µs
tSU;STA	START 条件的建立时间	4.7	-	0.6	-	µs
tHD;DAT	数据保持时间 [a]	0	3.45	0	0.9	µs

参数	描述	标准模式		快速模式		单位
		最小值	最大值	最小值	最大值	
tSU;DAT	数据建立时间 [a]	250	-	100	-	ns
tR	SDA 和 SCL 的信号上升时间	-	1000	20+0.1Cb	300	ns
tF	SDA 和 SCL 的信号下降时间	-	300	20+0.1Cb	300	ns
tSU;STO	STOP 条件的建立时间	4.0	-	0.6	-	μs
tBUF	STOP 与 START 之间的总线空闲时间	4.7	-	1.3	-	μs
Cb	每条总线线路的电容负载	-	400	-	400	pF

a：Hi3660 中此处寄存器可配。

表7-10 I^2C 时序参数（HS 模式）

参数	描述	Cb 最大=100pF		Cb = 400pF		单位
		最小值	最大值	最小值	最大值	
fSCL	SCL 时钟频率	0	3.4	0	1.7	MHz
tSU;STA	START/RESTART 条件的建立时间	160	-	160	-	ns
tHD;STA	START/RESTART 条件的保持时间	160	-	160	-	ns
tLOW	SCL 低电平宽度	160	-	320	-	ns
tHIGH	SCL 高电平宽度	60	-	120	-	ns
tSU;DAT	数据建立时间 [a]	10	-	10	-	ns
tHD;DAT	数据保持时间 [a]	0	70	0	150	ns
trCL	SCL 的信号上升时间	10	40	20	80	ns
trCL1	RESTART 和响应位后 SCL 的信号上升时间	10	80	20	160	ns
tF	SCL 的信号下降时间	10	40	20	80	ns
trDA	SDA 信号上升时间	10	80	20	160	ns
tfDA	SD 信号下降时间	10	80	20	160	ns
tSU;STO	STOP 条件的建立时间	160	-	160	-	ns

参数	描述	Cb 最大=100pF		Cb = 400pF		单位
		最小值	最大值	最小值	最大值	
Cb	SCL 与 SDA 之间的电容负载	-	100	-	400	pF

a：Hi3660 中此处寄存器可配。

注意

I2C 达到高速 3.4M 的条件：

要求 tHD:DAT 这个参数最大值不能超过 70ns。而且要求最小数据窗口时间是 230ns，hold 最大时间是 70ns。

7.5.4 寄存器描述

详见 synopsys　I2C IP 手册(https://www.synopsys.com/dw/ipdir.php?c=DW_apb_i2c)

7.6 GPIO

7.6.1 功能描述

7.6.1.1.1 功能介绍

Hi3660 共有 27 个普通 GPIO 模块（GPIO0~17，GPIO20~21，GPIO22~28）。外设区 20 个（GPIO0~17，GPIO20~21）；常开区 AON_SUBSYS 7 个（GPIO22~28）。

Hi3660 还有 2 个安全 GPIO 模块，外设区 1 个（GPIO0_SE）；常开区 AON_SUBSYS 区 1 个（GPIO1_SE）。

Hi3660 还有独立子系统内部的 GPIO 模块，如 SensorHub 的 4 组 GPIO（GPIO0_SH~GPIO3_SH）；UFS_PERI_SUBSYS 区的 2 组 GPIO（GPIO18~19）；MMC1_PERI_SUBSYS 区的 2 组 GPIO（GPIO0_EMMC ~ GPIO1_EMMC）。

每一个 GPIO 模块对应一组（共 8 个）GPIO 接口，且每组 GPIO 可产生三个中断，分别送到 GIC，LPMCU，CCPU 中的三个（另外 GPIO0SH~ GPIO3SH 的中断分别送到 GIC，IOMCU 中的两个）。这三个索引中断的源中断共享组内的 8 个 GPIO，但带有独立的屏蔽位。

相关 GPIO 的详细信息汇总如表 7-11 所示。

表7-11 GPIO 分组信息表

Instance	有效 GPIO 管脚数	中断注册支持
GPIO22~27	GPIO_176 ~ GPIO_190， GPIO_192 ~ GPIO_222	GIC
GPIO28	GPIO28 管脚接死没用	GIC
GPIO1_SE	GPIO_008_SE ~ GPIO_015_SE	GIC
GPIO0~17，	GPIO_001 ~ GPIO_098， GPIO_103~ GPIO_127	GIC
GPIO20	GPIO_160~ GPIO_165，	GIC
GPIO21	GPIO_168~ GPIO_173	GIC
GPIO0_SE	GPIO_000_SE ~ GPIO_007_SE	GIC
GPIO0_SH~ GPIO3_SH	GPIO_000_SH ~ GPIO_027_SH	GIC
	GPIO_028_SH ~ GPIO_031_SH，	GIC
GPIO18~19	GPIO_144 ~ GPIO_155	GIC
GPIO0_EMMC ~ GPIO1_EMMC	GPIO_00_EMMC ~ GPIO_09_EMMC	GIC
相关内容不在此章节详述。		

每组 GPIO 提供 8 个可编程的输入输出管脚，用于生成特定应用的输出信号或采集特定应用的输入信号。上表中未在有效 GPIO 管脚范围内的 GPIO 接口，虽 GPIO 模块内部软件可配，但没有从 PAD 复用出，忽略其功能即可。

使用时，按 GPIO 管脚序号模 8 取整，即可得该 GPIO 管脚所在组号，余数为该管脚在组中的序号。组中序号从 0 到 7（0 为最低位，7 为最高位）。

组号=int(GPIO 管脚序号 / 8)

组内序号=mod(GPIO 管脚序号，8)

例如 GPIO_017，其管脚号为 17，模 8 取整为 2 余数为 1，则该管脚处于 GPIO2 这一组中第 1 个。而 GPIO_016 则处于 GPIO2 这一组中第 0 个，GPIO_023 处于 GPIO2 这一组中第 7 个。

具体的 GPIO 的复用情况请参见“11.4.17 GPIO 信号”。

GPIO 的功能特点有：

- 每个 GPIO 管脚可配置为输入、输出或漏极开路输出。
  - 作为输入管脚时，可作为中断源，每个 GPIO 管脚都具有独立的中断控制。
  - 作为输出管脚时，每个 GPIO 管脚都可以独立地清零或置“1”。
  - 作为漏极开路输出时，单板上需要进行上拉控制，内部通过寄存器 GPIODIR 控制输出使能，从而实现板级的“线与”功能。
- 支持初始中断状态查询和屏蔽后中断状态查询。
- 支持中断唤醒系统：系统进入睡眠之前，配置 GPIO 使能中断。当系统进入睡眠之后，外设输入变化，GPIO 产生中断信号，唤醒系统。
- 不提供单独的软复位。

7.6.2 信号描述

GPIO 接口信号如表 7-12 所示。

说明

GPIO_000 为芯片内部信号，用户不可见。

表7-12 GPIO 接口信号表

信号名称	方向	含义
GPIO_001	Input/Output	标准输入输出端口，管脚复用请参见“10.5.15 GPIO 信号”。
GPIO_002	Input/Output	
……		
GPIO_174	Input/Output	
GPIO_222	Input/Output	

7.6.3 寄存器描述

详见 ARM 公版 pl061 IP 手册
(http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0183g/index.html)