广东新岸线计算机系统芯片有限公司

Guangdong Nufront CSC Co., Ltd

NL6621-NuAgent SDK

设计概念

张汇楼

2015 年 12 月 21 日

Change Log

Date	Version	Types (New/Delete/ Modify)	Editor	Description
2015-07-03	0.01.01	New	张汇楼	完成文档基本框架

目录

1. 引言

1.1 概述

本文描述 NL6621 NuAgent SDK 的 BSP 接口使用说明。BSP 编程风格与 STM32 库大部分一致，NuAgent SDK 提供(gpio,exti,usart,spi,wdg,spi,timer,i2c,dma,i2s,sdio)接口与相对应的 Demo，各个外设的 Demo 相关源代码文件 BspDemo.c、BspDemo.h。

客户根据自己的需求可以对相应功能的裁剪和实现。这里描述的接口只为了客户能够更好的理解 NL6621 的一些资源和接口的使用。本文档以“NL6621 SDK 用户使用手册.pdf”为基准，添加通用编程接口。相关接口和代码的变动以“NL6621 SDK 用户使用手册.pdf”为准。具体寄存器操作请参考《NL6621M_Product_Datasheet_Ch_V1.2(对外)》。

2. GPIO 库函数

函数名	描述
GPIO_Init	根据 GPIO_InitStruct 中指定的参数初始化外设 GPIOx 寄存器
GPIO_StructInit	把 GPIO_InitStruct 中的每一个参数按缺省值填入
GPIO_ReadInputDataBit	读取指定端口管脚的输入
GPIO_ReadInputData	读取 GPIO 全部端口的输入
GPIO_SetBits	设置指定的数据端口位（高电平）
GPIO_ResetBits	清除指定的数据端口位（低电平）
BSP_GPIOSetDir	设置 GPIO 端口输出输入模式
BSP_GPIOSetValue	设置 GPIO 端口输出高电平/低电平
GPIO_EXTILineConfig	选择 GPIO 管脚用作外部中断线路

2.1 函数 GPIO_Init

函数名	GPIO_Init
函数原形	void GPIO_Init(GPIO_InitTypeDef* GPIO_InitStruct)
功能描述	根据 GPIO_InitStruct 中指定的参数初始化外设 GPIOx 寄存器
输入参数 1	GPIO_InitStruct：指向结构 GPIO_InitTypeDef 的指针，包含了外设 GPIO 的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* GPIO5=设置输出高电平，低电平  */
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_5;       //管脚 PIN51
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out;   //设置输出
GPIO_Init(&GPIO_InitStructure);
GPIO_SetBits(GPIO_Pin_5);                       //输出高电平
GPIO_ResetBits(GPIO_Pin_5);                     //输出低电平
```

2.2 函数 GPIO_StructInit

函数名	GPIO_StructInit
函数原形	void GPIO_StructInit(GPIO_InitTypeDef* GPIO_InitStruct)
功能描述	把 GPIO_InitStruct 中的每一个参数按缺省值填入
输入参数	无

输出参数	无
返回值	无
先决条件	无
被调用函数	无

2.3 函数 GPIO_ReadInputDataBit

函数名	GPIO_ReadInputDataBit
函数原形	uint8_t GPIO_ReadInputDataBit(uint32_t GPIO_Pin)
功能描述	读取指定端口管脚的输入
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合
输出参数	无
返回值	输入端口管脚值
先决条件	无
被调用函数	无

例：/* 读取 GPIO7 输入值，赋值到 ReadValue */

```
u8 ReadValue;
ReadValue = GPIO_ReadInputDataBit(GPIO_Pin_7);
```

2.4 函数 GPIO_ReadInputData

函数名	GPIO_ReadInputData
函数原形	uint8_t GPIO_ReadInputData(uint32_t GPIO_Pin)
功能描述	读取 GPIO 全部端口的输入
输入参数	无
输出参数	无
返回值	读取 GPIO 全部端口的输入
先决条件	无
被调用函数	无

例：/* 读取 GPIO 全部输入值，赋值到 ReadValue */

```
u8 ReadValue;
ReadValue = GPIO_ReadInputData();
```

2.5 函数 GPIO_SetBits

函数名	GPIO_SetBits
函数原形	void GPIO_SetBits(uint32_t GPIO_Pin)
功能描述	设置指定的数据端口位（高电平）
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合

输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：/* 设置 GPIO10 、 GPIO17 输出高电平 */
```
GPIO_SetBits(GPIOA, GPIO_Pin_10 | GPIO_Pin_17);
```

2.6 函数 GPIO_ResetBits

函数名	GPIO_ResetBits
函数原形	void GPIO_ResetBits(uint32_t GPIO_Pin)
功能描述	清除指定的数据端口位（低电平）
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：/* 设置 GPIO10 、 GPIO17 输出低电平 */
```
GPIO_ResetBits(GPIOA, GPIO_Pin_10 | GPIO_Pin_17);
```

2.7 函数 BSP_GPIOSetDir

函数名	BSP_GPIOSetDir
函数原形	void BSP_GPIOSetDir(uint32_t GPIO_Pin, unsigned char dir)
功能描述	设置 GPIO 端口输出输入模式
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合
输入参数 2	Dir：输出/输入 - 0: Input. - 1: Output.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：/* 设置 GPIO10、GPIO17 输入模式 */
```
BSP_GPIOSetDir(GPIO_Pin_10 | GPIO_Pin_17, 0);
```

2.8 函数 BSP_GPIOSetDir

函数名	BSP_GPIOSetValue

函数原形	void BSP_GPIOSetValue(uint32_t GPIO_Pin, unsigned char bitValue)
功能描述	设置 GPIO 端口输出高低电平
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合
输入参数 2	bitValue：输出/输入 - 0: 低电平. - 1: 高电平.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：/* 设置 GPIO10、GPIO17 输出高电平 */
BSP_GPIOSetValue(GPIO_Pin_10 | GPIO_Pin_17, 0);

2.9 函数 GPIO_EXTILineConfig

函数名	GPIO_EXTILineConfig
函数原形	void GPIO_EXTILineConfig(uint32_t GPIO_Pin, FunctionalState Cmd)
功能描述	选择 GPIO 管脚用作外部中断线路
输入参数 1	GPIO_Pin：待设置的端口位 该参数可以取 GPIO_Pin_x(x 可以是 3-12 or 17-31)的任意组合
输入参数 2	Cmd：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：/* 设置 GPIO17 外部中断，首先先调用 GPIO_EXTILineConfig */
GPIO_EXTILineConfig(GPIO_Pin_17,IRQ_ENABLE);

3. EXTI 库函数

函数名	描述
EXTI_DeInit	将外设 EXTI 寄存器重设为缺省值
EXTI_Init	根据 EXTI_InitStruct 中指定的参数初始化外设 EXTI 寄存器
EXTI_StructInit	把 EXTI_InitStruct 中的每一个参数按缺省值填入
EXTI_GetITStatus	检查指定的 EXTI 线路触发请求发生与否
EXTI_ClearITPendingBit	清除 EXTI 线路挂起位
EXTI_GetIRQ	得到 GPIO NVIC 中断编号
EXTI_GPIO_Cmd	是否使能 GPIO NVIC 中断

3.1 函数 EXTI_DeInit

函数名	EXTI_DeInit
函数原形	void EXTI_DeInit(void)
功能描述	将外设 EXTI 寄存器重设为缺省值
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 将外设 EXTI 寄存器设置缺省值 */
EXTI_DeInit();
```

3.2 函数 EXTI_Init

函数名	EXTI_Init
函数原形	void EXTI_Init(EXTI_InitTypeDef* EXTI_InitStruct)
功能描述	根据 EXTI_InitStruct 中指定的参数初始化外设 EXTI 寄存器
输入参数 1	EXTI_InitStruct：指向结构 EXTI_InitTypeDef 的指针，包含了外设 EXTI 的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 设置 GPIO10、GPIO17 分别低电平中断和上升沿中断 */
GPIO_EXTILineConfig(GPIO_Pin_10,IRQ_ENABLE);
/* 初始化 GPIO5 低电平触发 */
EXTI_InitStructure.EXTI_Line = EXTI_Line10;
EXTI_InitStructure.EXTI_Mode = EXTI_LEVEL_SENSITIVE;
```

```
EXTI_InitStructure.EXTI_Trigger = EXTI_ACTIVE_LOW;
EXTI_InitStructure.EXTI_LineCmd = IRQ_ENABLE;
EXTI_Init(&EXTI_InitStructure);

GPIO_EXTILineConfig(GPIO_Pin_17,IRQ_ENABLE);
/* 初始化 GPIO7 上升沿触发 */
EXTI_InitStructure.EXTI_Line = EXTI_Line17;
EXTI_InitStructure.EXTI_Mode = EXTI_EDGE_SENSITIVE;
EXTI_InitStructure.EXTI_Trigger = EXTI_ACTIVE_HIGH;
EXTI_InitStructure.EXTI_LineCmd = IRQ_ENABLE;
EXTI_Init(&EXTI_InitStructure);
```

3.3 函数 EXTI_StructInit

函数名	EXTI_StructInit
函数原形	void EXTI_StructInit(EXTI_InitTypeDef* EXTI_InitStruct)
功能描述	把 EXTI_InitStruct 中的每一个参数按缺省值填入
输入参数 1	EXTI_InitStruct：指向结构 EXTI_InitTypeDef 的指针，待初始化
输出参数	无
返回值	无
先决条件	无
被调用函数	无

3.4 函数 EXTI_GetITStatus

函数名	EXTI_GetITStatus
函数原形	ITStatus EXTI_GetITStatus(uint32_t EXTI_Line)
功能描述	检查指定的 EXTI 线路触发请求发生与否
输入参数 1	EXTI_Line：待检查 EXTI 线路的挂起位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：

```
/* Get the status of EXTI line 8 */
ITStatus EXTIStatus;
EXTIStatus = EXTI_GetITStatus(EXTI_Line8);
```

3.5 函数 EXTI_ClearITPendingBit

函数名	EXTI_ClearITPendingBit
函数原形	void EXTI_ClearITPendingBit(uint32_t EXTI_Line)

功能描述	清除 EXTI 线路挂起位
输入参数 1	EXTI_Line：待清除 EXTI 线路的挂起位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Clears the EXTI line 2 interrupt pending bit */
EXTI_ClearITpendingBit(EXTI_Line2);
```

3.6 函数 EXTI_GetIRQ

函数名	EXTI_GetIRQ
函数原形	int EXTI_GetIRQ(uint32_t EXTI_Line)
功能描述	得到 GPIO NVIC 中断编号
输入参数 1	EXTI_Line：待查询 EXTI 线路的中断编号
输出参数	无
返回值	返回 NVIC 中断编号
先决条件	无
被调用函数	无

例：
```
/* 查询 EXTI_Line5 中断编号且使能中断 */
  NVIC_EnableIRQ(EXTI_GetIRQ(EXTI_Line5));
```

3.7 函数 EXTI_GPIO_Cmd

函数名	EXTI_GPIO_Cmd
函数原形	void EXTI_GPIO_Cmd(uint32_t EXTI_Line, FunctionalState NewState)
功能描述	是否使能 GPIO NVIC 中断
输入参数 1	EXTI_Line：待使能 EXTI 线路
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE：使能. - IRQ_DISABLE：不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 使能 EXTI_Line5 中断 */
  EXTI_GPIO_Cmd(EXTI_Line5,IRQ_ENABLE);
```

4. USART 库函数

函数名	描述
USART_Init	根据 USART_InitStruct 中指定的参数初始化外设 USART1 寄存器
USART_StructInit	把 USART_InitStruct 中的每一个参数按缺省值填入
USART_Cmd	使能或者失能 USART 外设
USART_ITConfig	使能或者失能指定的 USART 中断
USART_SendData	通过外设 USART 发送单个数据
USART_ReceiveData	返回 USART 最近接收到的数据
USART_GetFlagStatus	检查指定的 USART 标志位设置与否
USART_ClearFlag	清除 USART 的待处理标志位
USART_GetITStatus	检查指定的 USART 中断发生与否

4.1 函数 USART_Init

函数名	USART_Init
函数原形	void USART_Init(USART_InitTypeDef* USART_InitStruct)
功能描述	根据 USART_InitStruct 中指定的参数初始化外设 USART 寄存器
输入参数 1	USART_InitStruct：指向结构 USART_InitTypeDef 的指针，包含了外设 USART 的配置信息。
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：

```
/* 初始化串口，设置字长 8bit，停止位 1，无校验，FIFO 接收为 14 byte */
   USART_InitTypeDef USART_InitStructure;
   USART_InitStructure.USART_BaudRate = bound;
   USART_InitStructure.USART_WordLength = USART_WordLength_8b;
   USART_InitStructure.USART_StopBits = USART_StopBits_1;
   USART_InitStructure.USART_Parity = USART_Parity_No;
   //fifo RX 14byte
   USART_InitStructure.USART_FIFOMode=FCR_RCVR_TRIG_14|
FCR_FIFO_ON;
   USART_Init(&USART_InitStructure);
```

4.2 函数 USART_StructInit

函数名	USART_StructInit
函数原形	void USART_StructInit(USART_InitTypeDef* USART_InitStruct)

功能描述	把 USART_InitStruct 中的每一个参数按缺省值填入
输入参数 1	USART_InitStruct：指向结构 USART_InitTypeDef 的指针，待初始化
输出参数	无
返回值	无
先决条件	无
被调用函数	无

4.3 函数 USART_Cmd

函数名	USART_Cmd
函数原形	void USART_Cmd(FunctionalState NewState)
功能描述	使能或者失能 USART 外设
输入参数 1	NewState：使能/不使能 - IRQ_ENABLE：使能. - IRQ_DISABLE：不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Enable the USART1 */
USART_Cmd(IRQ_ENABLE);
```

4.4 函数 USART_ITConfig

函数名	USART_ITConfig
函数原形	void USART_ITConfig(uint16_t USART_IT, FunctionalState NewState)
功能描述	使能或者失能指定的 USART 中断
输入参数 1	USART_IT：待使能或者失能的 USART 中断源
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE：使能. - IRQ_DISABLE：不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 使能串口超时接收中断和 FIFO 接收中断 */
USART_ITConfig(IER_ERBFI_MASK, IRQ_ENABLE);
```

4.5 函数 USART_SendData

函数名	USART_SendData
函数原形	void USART_SendData(uint16_t Data)
功能描述	通过外设 USART 发送单个数据
输入参数 1	Data：待发送的数据
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Send one HalfWord on USART */
USART_SendData(0x26);
```

4.6 函数 USART_ReceiveData

函数名	USART_ReceiveData
函数原形	void USART_SendData(uint16_t Data)
功能描述	返回 USART 最近接收到的数据
输入参数	无
输出参数	无
返回值	接收到的字
先决条件	无
被调用函数	无

例：
```
/* Receive one halfword on USART */
u16 RxData;
RxData = USART_ReceiveData();
```

4.7 函数 USART_GetFlagStatus

函数名	USART_GetFlagStatus
函数原形	FlagStatus USART_GetFlagStatus(uint16_t USART_FLAG)
功能描述	检查指定的 USART 标志位设置与否
输入参数 1	USART_FLAG：待检查的 USART 标志位
输出参数	无
返回值	USART_FLAG 的新状态（ SET 或者 RESET）
先决条件	无
被调用函数	无

例：
```
/* 检查 USART 接收是否有数据标记 */
while (USART_GetFlagStatus(USART_FLAG_RXNE) == RESET);
```

4.8 函数 USART_ClearFlag

函数名	USART_ClearFlag
函数原形	uint32_t USART_ClearFlag(void)
功能描述	返回并清除 USARTx 的待处理标志位
输入参数	无
输出参数	无
返回值	返回 USART 状态寄存器
先决条件	无
被调用函数	无

4.9 函数 USART_GetITStatus

函数名	USART_GetITStatus
函数原形	ITStatus USART_GetITStatus(uint16_t USART_IT)
功能描述	检查指定的 USART 中断发生与否并且清除
输入参数	无
输出参数	无
返回值	返回 USART 中断状态寄存器
先决条件	无
被调用函数	无

5. TIMER 库函数

函数名	描述
TIM_Cmd	使能或者失能 TIMx 外设
TIM_ClearITPendingBit	清除 TIMx 的中断待处理位
TIM_ITConfig	使能或者失能指定的 TIM 中断
TIM_GetITStatus	检查指定的 TIMx 中断发生与否
TIM_TimeBaseInit	根据 TIM_TimeBaseInitStruct 中指定的参数初始化 TIMx 的时间基数单位

5.1 函数 TIM_Cmd

函数名	TIM_Cmd
函数原形	void TIM_Cmd(TIM_TypeDef TIMx, FunctionalState NewState)
功能描述	使能或者失能 TIMx 外设
输入参数 1	TIMx： TIMER1 or TIMER0
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Enables the TIM1 counter */
TIM_Cmd(TIMER1, IRQ_ENABLE);
```

5.2 函数 TIM_ClearITPendingBit

函数名	TIM_ClearITPendingBit
函数原形	void TIM_ClearITPendingBit(TIM_TypeDef TIMx)
功能描述	清除 TIMx 的中断待处理位
输入参数 1	TIMx： TIMER1 or TIMER0
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Clear the TIM1 interrupt pending bit */
TIM_ClearITPendingBit(TIMER1);
```

5.3 函数 TIM_ITConfig

函数名	TIM_ITConfig
函数原形	void TIM_ITConfig(TIM_TypeDef TIMx, FunctionalState NewState)
功能描述	使能或者失能指定的 TIM 中断
输入参数 1	TIMx： TIMER1 or TIMER0
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Enables the TIM1 Capture Compare channel 1 Interrupt source */
  TIM_ITConfig(TIMER1, IRQ_ENABLE);
```

5.4 函数 TIM_GetITStatus

函数名	TIM_GetITStatus
函数原形	ITStatus TIM_GetITStatus(TIM_TypeDef TIMx)
功能描述	检查指定的 TIMx 中断发生与否
输入参数 1	TIMx： TIMER1 or TIMER0
输出参数	无
返回值	定时器中断是否发送： SET or RESET
先决条件	无
被调用函数	无

例：
```
/* Check if the TIM2 Capture Compare 1 interrupt has occured or not
*/if(TIM_GetITStatus(TIMER1) == SET)
```

5.5 函数 TIM_TimeBaseInit

函数名	TIM_TimeBaseInit
函数原形	void TIM_TimeBaseInit(TIM_TypeDef TIMx, TIM_TimeBaseInitTypeDef* TIM_TimeBaseInitStruct)
功能描述	检查指定的 TIMx 中断发生与否
输入参数 1	TIMx： TIMER1 or TIMER0
输入参数 2	TIMTimeBase_InitStruct：指向结构 TIM_TimeBaseInitTypeDef 的指针，包含

	了 TIMx 时间基数单位的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：设置定时器 1,1ms 溢出中断。

```
    TIM_TimeBaseInitTypeDef  TIM_TimeBaseStructure;
    TIM_TimeBaseStructure.TIM_Period = 1000;
    TIM_TimeBaseStructure.TIM_CounterMode=
TIM_CounterMode_UserDown; /* TIM_CounterMode_FreeRun 不支持中断功能 */
    TIM_TimeBaseInit(TIMER1, &TIM_TimeBaseStructure);
```

6. WATCH 库函数

函数名	描述
IWDG_WriteAccessCmd	设置可写操作权限
IWDG_SetReload	设置 IWDG 重装载值
IWDG_ReloadCounter	按照 IWDG 重装载寄存器的值重装载 IWDG 计数器（喂狗）
IWDG_Enable	使能 IWDG

6.1 函数 IWDG_WriteAccessCmd

函数名	IWDG_WriteAccessCmd
函数原形	void IWDG_WriteAccessCmd(void)
功能描述	设置可写操作权限
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 设置可写操作权限 */
IWDG_WriteAccessCmd();
```

6.2 函数 IWDG_SetReload

函数名	IWDG_SetReload
函数原形	void IWDG_SetReload(uint8_t Reload)
功能描述	设置 IWDG 重装载值
输入参数 1	Reload：看门狗溢出时间公式：Tout = 2^(Reload+16)/40000 ms * 大概 100ms： Reload=6 * 大概 200ms： Reload=7 * 大概 400ms： Reload=8
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 设置看门狗溢出时间为 200ms */
IWDG_SetReload(7);
```

6.3 函数 IWDG_ReloadCounter

函数名	IWDG_ReloadCounter
函数原形	void IWDG_ReloadCounter(void)
功能描述	按照 IWDG 重装载寄存器的值重装载 IWDG 计数器
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 重装寄存器值（喂狗） */
IWDG_ReloadCounter();
```

6.4 函数 IWDG_Enable

函数名	IWDG_Enable
函数原形	void IWDG_Enable(void)
功能描述	使能 IWDG
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* 使能看门狗，一旦使能就无法关闭 */
IWDG_Enable();
```

7. QSPI 库函数

函数名	描述
QSPI_Init	根据 SPI_InitStruct 中指定的参数初始化外设 QSPI 寄存器
QSPI_Cmd	使能或者失能 QSPI 外设
QSPI_GetFlagStatus	检查指定的 QSPI 标志位设置与否
QSPI_ReceiveData	返回通过 QSPI 最近接收的数据
QSPI_SendData	通过外设 QSPI 发送一个数据
QSPI_GetRxFIFOLevel	获取 QSPI 当前 RXFIFO 内的数据的个数
QSPI_GetTxFIFOLevel	获取 QSPI 当前 TXFIFO 内的数据的个数
QSPI_SetTxFIFOLevel	设置 QSPI 的 TXFIFO 达到多少数据，产生中断。
QSPI_SetRxFIFOLevel	设置 QSPI 的 RXFIFO 达到多少数据，产生中断。
QSPI_Clear_Interrupt	清除 QSPI 中断
QSPI_ITConfig	使能或者失能指定的 QSPI 中断
QSPI_GetITFlagStatus	检查指定的 QSPI 中断发生与否
BSP_QSpiWriteByte	QSPI 外设写一个字节
BSP_QSpiRead	QSPI 外设读 N 数据
BSP_QSpiWait	等待 QSPI 一个读写过程完成

7.1 函数 QSPI_Init

函数名	QSPI_Init
函数原形	void QSPI_Init(QSPI_InitTypeDef* QSPI_InitStruct)
功能描述	根据 SPI_InitStruct 中指定的参数初始化外设 QSPI 寄存器
输入参数 1	QSPI_InitStruct：指向结构 QSPI_InitTypeDef 的指针，包含了外设 QSPI 的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：

```
        QSPI_InitTypeDef QSPI_InitStructure;
        QSPI_InitStructure.QSPI_Direction        =        QSPI_TMOD_TR;
//设置 SPI 单向或者双向的数据模式:SPI 设置为双线双向全双工
        QSPI_InitStructure.QSPI_Mode = QSPI_FRF_SPI;
//设置 SPI 模式: Motorola SPI
        QSPI_InitStructure.QSPI_DataSize       =       QSPI_DataSize_8;
//设置 SPI 的数据大小:SPI 发送接收 8 位帧结构
        QSPI_InitStructure.QSPI_CPOL = QSPI_CPOL_High;
//选择了串行时钟的稳态:时钟悬空高
        QSPI_InitStructure.QSPI_CPHA = QSPI_CPHA_2Edge;
```

```
//数据捕获于第二个时钟沿
        QSPI_InitStructure.QSPI_BaudRatePrescaler = 0x018;//20Mhz
        QSPI_Init(&QSPI_InitStructure);
//根据 SPI_InitStruct 中指定的参数初始化外设 SPIx 寄存器
```

7.2 函数 QSPI_Cmd

函数名	QSPI_Cmd
函数原形	void QSPI_Cmd(void)
功能描述	使能或者失能 QSPI 外设
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Enable QSPI */
QSPI_Cmd();
```

7.3 函数 QSPI_GetFlagStatus

函数名	QSPI_GetFlagStatus
函数原形	FlagStatus QSPI_GetFlagStatus(uint16_t QSPI_FLAG)
功能描述	检查指定的 QSPI 标志位设置与否
输入参数 1	SPI_FLAG：待检查的 QSPI 标志位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* QSPI TXFIFO 是否为满 */
while (QSPI_GetFlagStatus(QSPI_SR_TF_NOT_FULL) == RESET);
```

7.4 函数 QSPI_ReceiveData

函数名	QSPI_ReceiveData
函数原形	uint16_t QSPI_ReceiveData(void)
功能描述	返回通过 QSPI 最近接收的数据
输入参数	无
输出参数	无
返回值	接收到的字
先决条件	无
被调用函数	无

例：
```
/* Read the most recent data received by the QSPI peripheral */
u16 ReceivedData;
ReceivedData = QSPI_ReceiveData();
```

7.5 函数 QSPI_SendData

函数名	QSPI_SendData
函数原形	void QSPI_SendData(uint16_t Data)
功能描述	通过外设 QSPI 发送一个数据
输入参数 1	Data: 待发送的数据
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Send 0xA5 through the QSPI peripheral */
QSPI_SendData(0xA5);
```

7.6 函数 QSPI_GetRxFIFOLevel

函数名	QSPI_GetRxFIFOLevel
函数原形	uint16_t QSPI_GetTxFIFOLevel(void)
功能描述	获取 QSPI 当前 RXFIFO 内的数据的个数
输入参数	无
输出参数	无
返回值	返回 RXFIFO 当前的数据
先决条件	无
被调用函数	无

例：
```
u16 DataCount;
DataCount = QSPI_GetRxFIFOLevel();
```

7.7 函数 QSPI_GetTxFIFOLevel

函数名	QSPI_GetTxFIFOLevel
函数原形	uint16_t QSPI_GetTxFIFOLevel(void)
功能描述	获取 QSPI 当前 TXFIFO 内的数据的个数
输入参数	无
输出参数	无
返回值	返回 TXFIFO 当前的数据
先决条件	无
被调用函数	无

例：

```
u16 DataCount;
DataCount = QSPI_GetTxFIFOLevel();
```

7.8 函数 QSPI_SetTxFIFOLevel

函数名	QSPI_SetTxFIFOLevel
函数原形	uint16_t QSPI_SetTxFIFOLevel(uint16_t tx)
功能描述	设置 QSPI 的 TXFIFO 达到多少数据，产生中断。
输入参数 1	rx: 设置数据个数
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：TXFIFO 有 4 个或以上为数据时，产生中断

```
QSPI_SetTxFIFOLevel(0X03);
```

7.9 函数 QSPI_SetRxFIFOLevel

函数名	QSPI_SetRxFIFOLevel
函数原形	uint16_t QSPI_SetRxFIFOLevel(uint16_t rx)
功能描述	设置 QSPI 的 RXFIFO 达到多少数据，产生中断。
输入参数 1	rx: 设置数据个数
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：RXFIFO 有 4 个或以上为数据时，产生中断

```
QSPI_SetRxFIFOLevel(0X03);
```

7.10 函数 QSPI_Clear_Interrupt

函数名	QSPI_Clear_Interrupt
函数原形	void QSPI_Clear_Interrupt(uint8_t QSPI_IT)
功能描述	清除 QSPI 中断。
输入参数 1	QSPI_IT: 待清除的 QSPI 中断标志位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：清除 QSPI 发送溢出中断

```
QSPI_Clear_Interrupt(QSPI_TXOICR_FLAG);
```

7.11 函数 QSPI_ITConfig

函数名	QSPI_ITConfig
函数原形	void QSPI_ITConfig(uint16_t QSPI_IT, FunctionalState NewState)
功能描述	使能或者失能指定的 QSPI 中断
输入参数 1	QSPI_IT: 待清除的 QSPI 中断标志位
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：使能 QSPI 发送溢出中断

```
QSPI_ITConfig(QSPI_IMR_TXOIM,IRQ_ENABLE);
```

7.12 函数 QSPI_GetITFlagStatus

函数名	QSPI_GetITFlagStatus
函数原形	FlagStatus QSPI_GetITFlagStatus(uint16_t QSPI_FLAG)
功能描述	检查指定的 QSPI 中断发生与否
输入参数 1	QSPI_FLAG：待检查的 SPI 标志位
输出参数	无
返回值	QSPI_FLAG 的新状态（ SET 或者 RESET）
先决条件	无
被调用函数	无

例：检查 QSPI 发送溢出中断是否发生

```
if(QSPI_GetITFlagStatus(QSPI_IMR_TXOIM) == SET);
```

7.13 函数 BSP_QSpiWriteByte

函数名	BSP_QSpiWriteByte
函数原形	void BSP_QSpiWriteByte(uint8_t Byte)
功能描述	QSPI 外设写一个字节
输入参数 1	Byte：写入数据
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：QSPI 写入 0xAF

```
BSP_QSpiWriteByte(0xAF);
```

7.14 函数 BSP_QSpiRead

函数名	BSP_QSpiRead
函数原形	void BSP_QSpiRead(uint32_t RdCnt, uint8_t* pBuf)
功能描述	QSPI 外设读 N 数据
输入参数 1	Byte：读数据个数
输出参数	pBuf: 读出数据
返回值	无
先决条件	无
被调用函数	无

例：QSPI 连续读取 10 个字节，保存在 pRxData。

```
Unsigned char pRxData[10];
BSP_QSpiRead(10, pRxData);
```

7.15 函数 BSP_QSpiWait

函数名	BSP_QSpiWait
函数原形	void BSP_QSpiWait(void)
功能描述	等待 QSPI 一个读写过程完成
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

8. SPI 库函数

函数名	描述
SPI_Init	根据 SPI_InitStruct 中指定的参数初始化外设 SPI 寄存器
SPI_Cmd	使能或者失能 SPI 外设
SPI_GetFlagStatus	检查指定的 SPI 标志位设置与否
SPI_ReceiveData	返回通过 SPI 最近接收的数据
SPI_SendData	通过外设 SPI 发送一个数据
SPI_GetRxFIFOLevel	获取 SPI 当前 RXFIFO 内的数据的个数
SPI_GetTxFIFOLevel	获取 SPI 当前 TXFIFO 内的数据的个数
SPI_SetTxFIFOLevel	设置 SPI 的 TXFIFO 达到多少数据，产生中断。
SPI_SetRxFIFOLevel	设置 SPI 的 RXFIFO 达到多少数据，产生中断。
SPI_Clear_Interrupt	清除 SPI 中断
SPI_ITConfig	使能或者失能指定的 SPI 中断
SPI_GetITFlagStatus	检查指定的 SPI 中断发生与否
SPI_DMA_Cmd	使能 SPI DMA 传输模式

8.1 函数 SPI_Init

函数名	SPI_Init
函数原形	void SPI_Init(SPI_InitTypeDef* SPI_InitStruct)
功能描述	根据 SPI_InitStruct 中指定的参数初始化外设 SPI 寄存器
输入参数 1	SPI_InitStruct：指向结构 SPI_InitTypeDef 的指针，包含了外设 SPI 的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：

```
        SPI_InitTypeDef SPI_InitStructure;
        SPI_InitStructure.SPI_Direction=SPI_TMOD_TR;
//设置 SPI 单向或者双向的数据模式:SPI 设置为双线双向全双工
        SPI_InitStructure.QSPI_Mode = SPI_FRF_SPI;
//设置 SPI 模式: Motorola SPI
        SPI_InitStructure.SPI_DataSize= PI_DataSize_8;
 //设置 SPI 的数据大小:SPI 发送接收 8 位帧结构
        SPI_InitStructure.SPI_CPOL = SPI_CPOL_High;
//选择了串行时钟的稳态:时钟悬空高
        SPI_InitStructure.SPI_CPHA = SPI_CPHA_2Edge;
//数据捕获于第二个时钟沿
        SPI_InitStructure.QSPI_BaudRatePrescaler=
```

```
SPI_BaudRatePrescaler_2;//20Mhz
        SPI_Init(&QSPI_InitStructure);
//根据 SPI_InitStruct 中指定的参数初始化外设 SPIx 寄存器
```

8.2 函数 SPI_Cmd

函数名	SPI_Cmd
函数原形	void SPI_Cmd(void)
功能描述	使能或者失能 SPI 外设
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Enable SPI */
    SPI_Cmd();
```

8.3 函数 SPI_GetFlagStatus

函数名	SPI_GetFlagStatus
函数原形	FlagStatus SPI_GetFlagStatus(uint16_t SPI_FLAG)
功能描述	检查指定的 SPI 标志位设置与否
输入参数 1	SPI_FLAG：待检查的 SPI 标志位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* SPI TXFIFO 是否为满 */
    while (SPI_GetFlagStatus(SR_TF_NOT_FULL) == RESET);
```

8.4 函数 SPI_ReceiveData

函数名	SPI_ReceiveData
函数原形	uint16_t SPI_ReceiveData(void)
功能描述	返回通过 SPI 最近接收的数据
输入参数	无
输出参数	无
返回值	接收到的字
先决条件	无
被调用函数	无

例：
```
/* Read the most recent data received by the SPI peripheral */
   u16 ReceivedData;
```

```
ReceivedData = SPI_ReceiveData();
```

8.5 函数 SPI_SendData

函数名	SPI_SendData
函数原形	void SPI_SendData(uint16_t Data)
功能描述	通过外设 SPI 发送一个数据
输入参数 1	Data: 待发送的数据
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Send 0xA5 through the SPI peripheral */
SPI_SendData(0xA5);
```

8.6 函数 SPI_GetRxFIFOLevel

函数名	SPI_GetRxFIFOLevel
函数原形	uint16_t SPI_GetTxFIFOLevel(void)
功能描述	获取 SPI 当前 RXFIFO 内的数据的个数
输入参数	无
输出参数	无
返回值	返回 RXFIFO 当前的数据
先决条件	无
被调用函数	无

例：
```
u16 DataCount;
DataCount = SPI_GetRxFIFOLevel();
```

8.7 函数 QSPI_GetTxFIFOLevel

函数名	SPI_GetTxFIFOLevel
函数原形	uint16_t SPI_GetTxFIFOLevel(void)
功能描述	获取 SPI 当前 TXFIFO 内的数据的个数
输入参数	无
输出参数	无
返回值	返回 TXFIFO 当前的数据
先决条件	无
被调用函数	无

例：
```
u16 DataCount;
```

```
DataCount = SPI_GetTxFIFOLevel();
```

8.8 函数 SPI_SetTxFIFOLevel

函数名	SPI_SetTxFIFOLevel
函数原形	uint16_t SPI_SetTxFIFOLevel(uint16_t tx)
功能描述	设置 SPI 的 TXFIFO 达到多少数据，产生中断。
输入参数 1	rx: 设置数据个数
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：TXFIFO 有 4 个或以上为数据时，产生中断

```
SPI_SetTxFIFOLevel(0X03);
```

8.9 函数 SPI_SetRxFIFOLevel

函数名	SPI_SetRxFIFOLevel
函数原形	uint16_t SPI_SetRxFIFOLevel(uint16_t rx)
功能描述	设置 SPI 的 RXFIFO 达到多少数据，产生中断。
输入参数 1	rx: 设置数据个数
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：RXFIFO 有 4 个或以上为数据时，产生中断

```
SPI_SetRxFIFOLevel(0X03);
```

8.10 函数 SPI_Clear_Interrupt

函数名	SPI_Clear_Interrupt
函数原形	void SPI_Clear_Interrupt(uint8_t SPI_IT)
功能描述	清除 SPI 中断。
输入参数 1	SPI_IT: 待清除的 SPI 中断标志位
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：清除 SPI 发送溢出中断

```
SPI_Clear_Interrupt(TXOICR_FLAG);
```

8.11 函数 SPI_ITConfig

函数名	SPI_ITConfig
函数原形	void SPI_ITConfig(uint16_t SPI_IT, FunctionalState NewState)
功能描述	使能或者失能指定的 SPI 中断
输入参数 1	SPI_IT: 待清除的 SPI 中断标志位
输入参数 2	NewState：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：使能 SPI 发送溢出中断

```
SPI_ITConfig(IMR_TXOIM,IRQ_ENABLE);
```

8.12 函数 SPI_GetITFlagStatus

函数名	SPI_GetITFlagStatus
函数原形	FlagStatus SPI_GetITFlagStatus(uint16_t SPI_FLAG)
功能描述	检查指定的 SPI 中断发生与否
输入参数 1	SPI_FLAG：待检查的 SPI 标志位
输出参数	无
返回值	SPI_FLAG 的新状态（ SET 或者 RESET）
先决条件	无
被调用函数	无

例：检查 QSPI 发送溢出中断是否发生

```
if(SPI_GetITFlagStatus(IMR_TXOIM) == SET);
```

9. I2C 库函数

函数名	描述
SPI_Init	根据 SPI_InitStruct 中指定的参数初始化外设 SPI 寄存器
SPI_Cmd	使能或者失能 SPI 外设
SPI_GetFlagStatus	检查指定的 SPI 标志位设置与否
SPI_ReceiveData	返回通过 SPI 最近接收的数据
SPI_SendData	通过外设 SPI 发送一个数据
SPI_GetRxFIFOLevel	获取 SPI 当前 RXFIFO 内的数据的个数
SPI_GetTxFIFOLevel	获取 SPI 当前 TXFIFO 内的数据的个数
SPI_SetTxFIFOLevel	设置 SPI 的 TXFIFO 达到多少数据，产生中断。
SPI_SetRxFIFOLevel	设置 SPI 的 RXFIFO 达到多少数据，产生中断。
SPI_Clear_Interrupt	清除 SPI 中断
SPI_ITConfig	使能或者失能指定的 SPI 中断
SPI_GetITFlagStatus	检查指定的 SPI 中断发生与否
SPI_DMA_Cmd	使能 SPI DMA 传输模式

9.1 函数 SPI_GetITFlagStatus

函数名	SPI_GetITFlagStatus
函数原形	FlagStatus SPI_GetITFlagStatus(uint16_t SPI_FLAG)
功能描述	检查指定的 SPI 中断发生与否
输入参数 1	SPI_FLAG：待检查的 SPI 标志位
输出参数	无
返回值	SPI_FLAG 的新状态（ SET 或者 RESET）
先决条件	无
被调用函数	无

例：检查 QSPI 发送溢出中断是否发生

```
if(SPI_GetITFlagStatus(IMR_TXOIM) == SET);
```

10. I2C 库函数

函数名	描述
I2C_Init	根据 I2C_InitStruct 中指定的参数初始化外设 I2C 寄存器
I2C_Cmd	使能或者失能 I2C 外设
I2C_WaitForBus	等待 I2C 发送数据完毕且总线空闲
I2C_XferFinish	等待总线停止信号，用于 I2C 读数据
BSP_I2CSetAdr	设置访问 I2C 设备访问地址
BSP_I2CRead	连续读 I2C 总线数据
BSP_I2CWrite	连续写 I2C 总线数据
BSP_I2CSeqRead	连续读某地址 I2C 设备数据
BSP_I2CSeqWrite	连续写某地址 I2C 设备数据

10.1 函数 I2C_Init

函数名	I2C_Init
函数原形	void I2C_Init(I2C_InitTypeDef* I2C_InitStruct)
功能描述	根据 I2C_InitStruct 中指定的参数初始化外设 I2C 寄存器
输入参数 1	I2C_InitStruct：指向结构 I2C_InitTypeDef 的指针，包含了外设 GPIO 的配置信息
输出参数	无
返回值	无
先决条件	无
被调用函数	无

例：
```
/* Initialize the I2C1 according to the I2C InitStructure members */
    I2C InitTypeDef  I2C InitStruct;

    I2C InitStruct.I2C Mode = FAST MODE;

    I2C InitStruct.I2C ClockLow = 200;// I2C 频率 40000/(200+200) Khz =
100Khz

    I2C InitStruct.I2C ClockHihg = 200;

    I2C InitStruct.I2C OwnAddress1 = EEPROM ADDR;

    I2C Init(&I2C InitStruct);
```

10.2 函数 I2C_Cmd

函数名	I2C_Init
函数原形	void I2C_Cmd(FunctionalState NewState)

功能描述	使能或者失能 I2C 外设
输入参数 1	NewState：使能/不使能 - IRQ_ENABLE: 使能. - IRQ_DISABLE: 不使能.
输出参数	无
返回值	无
先决条件	无
被调用函数	无

10.3 函数 I2C_WaitForBus

函数名	WaitForBus
函数原形	void I2C_WaitForBus(void)
功能描述	等待 I2C 发送数据完毕且总线空闲
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

10.4 函数 I2C_XferFinish

函数名	I2C_XferFinish
函数原形	void I2C_XferFinish(void)
功能描述	等待总线停止信号，用于 I2C 读数据
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

10.5 函数 BSP_I2CSetAdr

函数名	BSP_I2CSetAdr
函数原形	void BSP_I2CSetAdr(uint32_t Adr,uint32_t time)
功能描述	设置访问 I2C 设备访问地址
输入参数 1	Adr: 设置访问地址
输入参数 2	time: 设置等待应答信号超时时间
输出参数	无
返回值	无
先决条件	无

被调用函数	无

10.6 函数 BSP_I2CRead

函数名	BSP_I2CRead
函数原形	BOOL_T BSP_I2CRead(UINT8* pBuf, UINT32 Len,uint32_t time)
功能描述	连续读 I2C 总线数据
输入参数 1	pBuf: 数据缓存
输入参数 2	Len: 数据长度
输入参数 3	time: 设置等待应答信号超时时间
输出参数	无
返回值	暂时没有用
先决条件	无
被调用函数	无

10.7 函数 BSP_I2CWrite

函数名	BSP_I2CWrite
函数原形	BOOL_T BSP_I2CWrite(uint8_t* pBuf, uint32_t Len)
功能描述	连续写 I2C 总线数据
输入参数 1	pBuf: 数据缓存
输入参数 2	Len: 数据长度
输出参数	无
返回值	暂时没有用
先决条件	无
被调用函数	无

10.8 函数 BSP_I2CSeqRead

函数名	BSP_I2CSeqRead
函数原形	BOOL_T BSP_I2CSeqRead(uint32_t Adr, uint8_t *pBuf, uint32_t Len)
功能描述	连续读某地址 I2C 设备数据
输入参数 1	Adr: I2C 设备内存访问地址
输入参数 2	pBuf: 数据缓存
输入参数 3	Len: 数据长度
输出参数	无
返回值	暂时没有用
先决条件	无
被调用函数	无

10.9 函数 BSP_I2CSeqWrite

函数名	BSP_I2CSeqWrite
函数原形	BOOL_T BSP_I2CSeqWrite(uint32_t Adr, uint8_t *pBuf, uint32_t Len)
功能描述	连续写某地址 I2C 设备数据
输入参数 1	Adr: I2C 设备内存访问地址
输入参数 2	pBuf: 数据缓存
输入参数 3	Len: 数据长度
输出参数	无
返回值	暂时没有用
先决条件	无
被调用函数	无

11. I2S 库函数

函数名	描述
BSP_I2SInit	初始化 I2S 外设功能
BSP_I2SStart	启动 I2S 传输
BSP_I2SHalt	暂停 I2S 传输
BSP_I2SDeinit	初始化 I2S 寄存器默认状态
BSP_I2SMute	进入静音（数据不足使用）
BSP_I2SSound	取消静音
BSP_I2SWrite	I2S 写数据
BSP_I2SDMAWrite	I2S 通过 DMA 传输数据
BSP_I2SIntISR	I2S 中断函数

12. SDIO 库函数

函数名	描述
BSP_SDIOInit	初始化 SDIO Slave 功能，CMD53 命令传输
BSP_HostIntISR	SDIO 中断函数

12.1 函数 BSP_SDIOInit

函数名	BSP_SDIOInit
函数原形	void BSP_SDIOInit(void)
功能描述	初始化 SDIO Slave 功能，CMD53 命令传输
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

12.2 函数 BSP_HostIntISR

函数名	BSP_HostIntISR
函数原形	void BSP_HostIntISR(void)
功能描述	SDIO 中断函数，处理数据发送和接收
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无

13. DMA 库函数

函数名	描述
BSP_DmaInit	选择 DMA 通道初始化寄存器
BSP_DmaMoveMemInit	选择 DMA 通道初始化寄存器（内存搬运用）
BSP_DmaStart	选择 DMA 通道启动 DMA 传输
BSP_DmaHalt	暂停 DMA 传输数据
BSP_DmaDeinit	初始化 DMA 寄存器默认状态
BSP_DmaIntISR	DMA 中断函数

13.1 函数 BSP_DmaInit

函数名	BSP_DmaInit
函数原形	void BSP_DmaInit (uint8_t channel_num)
功能描述	选择 DMA 通道初始化寄存器
输入参数 1	channel_num： DMA_CHANNEL_0 DMA_CHANNEL_1 DMA_CHANNEL_2 DMA_CHANNEL_3
输出参数	无
返回值	无
先决条件	无
被调用函数	无

13.2 函数 BSP_DmaMoveMemInit

函数名	BSP_DmaMoveMemInit
函数原形	void BSP_DmaMoveMemInit (uint8_t channel_num)
功能描述	选择 DMA 通道初始化寄存器（内存搬运用）
输入参数 1	channel_num： DMA_CHANNEL_0 DMA_CHANNEL_1 DMA_CHANNEL_2 DMA_CHANNEL_3
输出参数	无
返回值	无
先决条件	无
被调用函数	无

13.3 函数 BSP_DmaStart

函数名	BSP_DmaStart
函数原形	void BSP_DmaStart(uint32_t* pSrc, uint32_t* pDest, uint8_t channel_num)
功能描述	选择 DMA 通道启动 DMA 传输
输入参数 1	pSrc： 源地址
输入参数 2	pDest： 目标地址
输入参数 3	channel_num： DMA_CHANNEL_0 DMA_CHANNEL_1 DMA_CHANNEL_2 DMA_CHANNEL_3
输出参数	无
返回值	无
先决条件	无
被调用函数	无

13.4 函数 BSP_DmaHalt

函数名	BSP_DmaHalt
函数原形	void BSP_DmaHalt(uint8_t channel_num)
功能描述	暂停 DMA 传输数据
输入参数 1	channel_num： DMA_CHANNEL_0 DMA_CHANNEL_1 DMA_CHANNEL_2 DMA_CHANNEL_3
输出参数	无
返回值	无
先决条件	无
被调用函数	无

13.5 函数 BSP_DmaDeinit

函数名	BSP_DmaDeinit
函数原形	void BSP_DmaDeinit (void)
功能描述	初始化 DMA 寄存器默认状态
输入参数	无

输出参数	无
返回值	无
先决条件	无
被调用函数	无

13.6 函数 BSP_DmaIntISR

函数名	BSP_DmaIntISR
函数原形	void BSP_DmaIntISR(void)
功能描述	DMA 中断函数
输入参数	无
输出参数	无
返回值	无
先决条件	无
被调用函数	无