版本	*V1.5*
日期	*2022.10.21*

APT32F110x 系列

CSI API 手册

相关文档：

APT32F110x系列芯片使用手册

APT32F1101芯片数据手册

APT32F1102芯片数据手册

APT32F1103芯片数据手册

APT32F1104芯片数据手册

QuickStart_APT32F110x系列_CSI

Revision History

版本	日期	描述	作者
V1.0	2022.1.11	新建	APT AE Group
V1.1	2022.1.24	升级接口函数，修改部分BUG	APT AE Group
V1.2	2022.2.18	统一优化部分接口函数	APT AE Group
V1.3	2022.4.1	优化部分接口函数、修改UART和DMA驱动中BUG	APT AE Group
V1.4	2022.7.15	升级IWDT的OPEN函数、修改DMA驱动； UART/USART/SPI相关DMA操作函数更新； 支持动态内存申请，添加中断处理开关。	APT AE Group
V1.5	2022.10.21	增加touch处理相关代码、升级PIN中断处理，更新GPTA/GPTB相位使能，CMP部分规范代码，系统频率配置函数增加参数，IFC中增加page erase和page PGM操作，删除系统中long long类型除法运算等	APT AE Group

1 概述

APT32F110x 驱动软件遵循 APT CSI（APT chip standard interface）接口规范，帮助客户快速形成产品方案。本文档旨在对该接口规范下的 API 函数进行说明。

几个注意点：

1. APT32F110x 系列内包含多款 MCU，如果使用的 MCU 不包含某一外设，这部分外设相关的 api 函数无效。
2. 所有外设的 api 函数中都不包含管脚功能配置，使用前需要调用 csi_pin_set_mux()函数进行配置。
3. 有关工程结构、系统初始化、调试打印、中断相关的使用请参考 QuickStart_APT32F110x系列_CSI。

2 时钟

2.1 API列表

Table 2-1 时钟CSI接口函数

API	说明	函数位置
csi_sysclk_config	配置系统时钟SCLK。	sys_clk.c
csi_clo_config	设置管脚的CLO输出。	
csi_get_sclk_freq	获得当前SCLK频率值。	
csi_get_pclk_freq	获得当前PCLK频率值。	
csi_clk_enable	使能模块时钟（SYSCON端）	clk.c
csi_clk_disable	禁止模块时钟（SYSCON端）	
csi_emosc_enable	打开EMOSC。	
csi_emosc_disable	关闭EMOSC。	
csi_esosc_enable	打开ESOSC。	
csi_esosc_disable	关闭ESOSC。	
csi_imosc_enable	打开IMOSC。	
csi_imosc_disable	关闭IMOSC。	
csi_hfosc_enable	打开HFOSC。	
csi_hfosc_disable	关闭HFOSC。	
csi_isosc_enable	打开ISOSC。	
csi_isosc_disable	关闭ISOSC。	
csi_emosc_filter_enable	使能/禁止 EMOSC滤波	
csi_esosc_filter_enable	使能/禁止 ESOSC施密特滤波	
csi_emosc_set_gain	设置EMOSC增益控制值	
csi_esosc_set_gain	设置ESOSC增益控制值	
csi_clk_calib	内部IMCLK/HFCLK校准	

2.2 API详细说明

2.2.1 csi_sysclk_config

ccsi_error_t csi_sysclk_config(csi_clk_config_t tClkCfg)

2.2.1.1 功能描述

用于配置系统时钟 SCLK。调用此函数，需要传入 csi_clk_config_t 结构体类型参数，系统默认传入 tClkConfig，具体如下介绍

1. 需要设置时钟参数 tClkConfig （位于board_config.c）。

```
/// system clock configuration parameters to define source, source freq(if selectable), sdiv and pdiv
csi_clk_config_t tClkConfig =
    //{SRC_HFOSC, HFOSC_48M_VALUE, SCLK_DIV3, PCLK_DIV1, 5556000, 5556000};
    //{SRC_EMOSC, 20000000, SCLK_DIV1, PCLK_DIV2, 5556000, 5556000};
    {SRC_IMOSC, IMOSC_5M_VALUE, SCLK_DIV1, PCLK_DIV2,5556000, 5556000};
    //{SRC_HFOSC, HFOSC_48M_VALUE, SCLK_DIV2, PCLK_DIV1,5556000, 5556000};
    //{SRC_IMOSC, IMOSC_4M_VALUE, SCLK_DIV1, PCLK_DIV1,5556000, 5556000};
```

Figure 2-1 时钟参数结构体

2. 如果时钟源为EMOSC，需要预先设置XIN，XOUT管脚AF功能。

函数执行完毕后，会更新tClkConfig中的wSclk和wPclk。

2.2.1.2 参数/返回值说明

1. 参数：无。

2. 返回值

   如果系统时钟选择IMOSC或HFOSC，但tClkConfig的第二次参数不是可选值时，会返回CSI_ERROR。否则返回CSI_OK。

3. 参数说明表

变量	说明	变量位置

tClkConfig	全局变量，结构体。
	`typedef struct {`
	`  cclk_src_e  eClkSrc;  //clock source`
	`  uint32_t  wFreq;  //clock frequency`
	`  hclk_div_e  eSdiv;  //SDIV`
	`  pclk_div_e  ePdiv;  //PDIV`
	`  uint32_t  wSclk;  //SCLK`
	`  uint32_t  wPclk;`
	`}csi_clk_config_t;`
		在board_config.c中定义
		在board_config.h中extern。如需使用，#include board_config.h

2.2.2 csi_clo_config

csi_error_t csi_clo_config(clo_src_e eCloSrc, clo_div_e eCloDiv)

2.2.2.1 功能描述

用于设置管脚的 CLO 输出。该函数常用于调试阶段，查看当前各种时钟信号的实际频率。

2.2.2.2 参数/返回值说明

1. 参数

   eCloSrc：CLO管脚上输出的时钟对象，枚举定义详见clo_src_e。

   eCloDiv：CLO管脚上输出的时钟对象分频值，枚举定义详见clo_div_e。

2. 返回值

   CSI_OK：初始化成功。

   CSI_ERROR：初始化失败。

3. 参数说明表

参数/返回值	说明	枚举变量位置
eCloSrc	clo_src_e枚举值。 typedef enum{ CLO_ISCLK = 0, CLO_IMCLK, CLO_EMCLK = 3, CLO_HFCLK, CLO_RTCCLK = 6, CLO_PCLK, CLO_HCLK, CLO_IWDTCLK, CLO_SYSCLK = 0xd }clo_src_e;	csp_syscon,h
eCloDiv	clo_div_e枚举值。 typedef enum{ CLO_DIV1 = 1, CLO_DIV2, CLO_DIV4, CLO_DIV8, CLO_DIV16 }clo_div_e;	
csi_error_t	csi_error_t 中定义值	common.h

2.2.3 csi_get_sclk_freq

uint32_t csi_get_sclk_freq(void)

2.2.3.1 功能描述

获得当前SCLK频率值。函数执行完毕后，会更新tClkConfig中的wSclk。

2.2.3.2 参数/返回值说明

1. 参数：无。
2. 返回值：返回SCLK的频率值（单位：Hz）
3. 参数说明表

返回值类型	说明
uint32_t	返回系统时钟(SCLK)频率

2.2.4 csi_get_pclk_freq

uint32_t csi_get_pclk_freq(void)

2.2.4.1 功能描述

获得当前PCLK频率值。函数执行完毕后，会更新tClkConfig中的wPclk。我们的csi驱动中，如UART在配置时只需要传入波特率的数值就可以，是因为内部会调用这个函数，自动计算出在当前PCLK的频率下要取得用户设置的波特率时，需要对相关寄存器写入的值。很多通信外设和定时外设也都是这样。

2.2.4.2 参数/返回值说明

1. 参数：无。
2. 返回值：返回PCLK的频率值（单位：Hz）
3. 参数说明表

返回值类型	说明
uint32_t	返回外设时钟(PCLK)频率

2.2.5 csi_clk_enable

*void csi_clk_enable(void *pIpBase)*

2.2.5.1 功能描述

处于功耗的考虑，大部分外设的时钟都有开关。这些开关都集中位于 SYSCON_PCER0/1。所以要开启某一个外设，需要先打开这个开关。这个函数通常已经集成在外设的初始化函数中，不需要用户调用。

2.2.5.2 参数/返回值说明

1. 参数

   pIpBase：指向外设基地址的结构体指针，使用时直接传入对应结构体指针，类型无需强制转换。

2. 返回值：无

3. 参数说明表

参数	说明	位置

pIpBase	csp_uart_t *UART0 = (csp_uart_t *)(APB_UART0_BASE);
	csp_uart_t *UART1 = (csp_uart_t *)(APB_UART1_BASE);
	csp_uart_t *UART2 = (csp_uart_t *)(APB_UART2_BASE);
	csp_usart_t *USART0 = (csp_usart_t *)(APB_USART0_BASE);
	csp_spi_t *SPI0 = (csp_spi_t *)(APB_SPI0_BASE);
	csp_sio_t *SIO0 = (csp_sio_t *)(APB_SIO0_BASE);
	csp_i2c_t *I2C0 = (csp_i2c_t *)(APB_I2C0_BASE);
	csp_cnta_t *CA0 = (csp_cnta_t *)(APB_CNTA_BASE);
	csp_gpta_t *GPTA0 = (csp_gpta_t *)(APB_GPTA0_BASE);
	csp_gpta_t *GPTA1 = (csp_gpta_t *)(APB_GPTA1_BASE);
	csp_gptb_t *GPTB0 = (csp_gptb_t *)(APB_GPTB0_BASE);
	csp_gptb_t *GPTB1 = (csp_gptb_t *)(APB_GPTB1_BASE);
	csp_ept_t *EPT0 = (csp_ept_t *)(APB_EPT0_BASE);
	csp_lpt_t *LPT = (csp_lpt_t *)(APB_LPT_BASE) ;
	csp_wwdt_t *WWDT = (csp_wwdt_t *)(APB_WWDT_BASE);
	csp_bt_t *BT0 = (csp_bt_t *)(APB_BT0_BASE) ;
	csp_bt_t *BT1 = (csp_bt_t *)(APB_BT1_BASE) ;
	csp_crc_t *CRC = (csp_crc_t *)(AHB_CRC_BASE);
		devices.c中定义

2.2.6 csi_clk_disable

*void csi_clk_disable(void *pIpBase)*

2.2.6.1 功能描述

关闭外设时钟，功能和 csi_clk_enable 函数相反。

2.2.6.2 参数/返回值说明

同 csi_clk_enable 函数。

2.2.7 csi_emosc_enable

csi_error_t csi_emosc_enable(uint32_t wFreq)

2.2.7.1 功能描述

使能外部主晶振。

2.2.7.2 参数/返回值说明

1. 参数

   wFreq：外部主振荡器频率值。

2. 返回值：

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明表

参数/返回值	说明	枚举变量位置
wFreq	uint32_t 类型数值	
csi_error_t	csi_error_t 中定义值	common.h

2.2.8 csi_emosc_disable

csi_error_t csi_emosc_disable(void)

2.2.8.1 功能描述

关闭外部主晶振。

2.2.8.2 参数/返回值说明

1. 参数：无

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.9 csi_esosc_enable

csi_error_t csi_esosc_enable(uint32_t wFreq)

2.2.9.1 功能描述

使能外部副晶振。

2.2.9.2 参数/返回值说明

1. 参数

   wFreq：外部副振荡器频率值。

2. 返回值：

   CSI_OK：初始化成功。

   CSI_ERROR：初始化失败。

3. 参数/返回值说明表

返回值	说明	枚举变量位置
wFreq	uint32_t 类型数值	
csi_error_t	csi_error_t 中定义值	common.h

2.2.10 csi_esosc_disable

2.2.10.1 功能描述

关闭外部副副晶振。

2.2.10.2 参数/返回值说明

1. 参数：无

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.11 csi_imosc_enable

csi_error_t csi_imosc_enable(uint8_t byFre)

2.2.11.1 功能描述

使能内部低速振荡器。

2.2.11.2 参数/返回值说明

1. 参数

   byFre：振荡器可选择 4 中不同频率值，0~3 对应频率为：5.5M/4.2M/2.09M/131K

2. 返回值：

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明表

返回值	说明	枚举变量位置
byFre	uint8_t 类型数值	
csi_error_t	csi_error_t 中定义值	common.h

2.2.12 csi_imosc_disable

csi_error_t csi_imosc_disable(void)

2.2.12.1 功能描述

关闭内部低速振荡器。

2.2.12.2 参数/返回值说明

1. 参数：无

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.13 csi_hfosc_enable

csi_error_t csi_hfosc_enable(uint8_t byFre)

2.2.13.1 功能描述

使能内部高速振荡器。

2.2.13.2 参数/返回值说明

1. 参数

   byFre：振荡器可选择 4 中不同频率值，0~3 对应频率为：48M/24M/12M/6M

2. 返回值：

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明表

返回值	说明	枚举变量位置
byFre	uint8_t 类型数值	
csi_error_t	csi_error_t 中定义值	common.h

2.2.14 csi_hfosc_disable

csi_error_t csi_hfosc_disable(void)

2.2.14.1 功能描述

关闭内部高速振荡器。

2.2.14.2 参数/返回值说明

1. 参数：无

2. 返回值

   CSI_OK：成功。

CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.15 csi_isosc_enable

csi_error_t csi_isosc_enable(void)

2.2.15.1 功能描述

使能内部超低速振荡器，频率值为 27K。

2.2.15.2 参数/返回值说明

1. 参数：无
2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.16 csi_isosc_disable

csi_error_t csi_isosc_disable(void)

2.2.16.1 功能描述

关闭内部超低速振荡器。

2.2.16.2 参数/返回值说明

1. 参数：无
2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

2.2.17 csi_emosc_filter_enable

void csi_emosc_filter_enable(csi_emfilt_e eFiltSel, bool bEnable)

2.2.17.1 功能描述

EMOSC 滤波使能/禁止

2.2.17.2 参数/返回值说明

1. 参数

eFiltSel：滤波范围选择，枚举定义详见csi_emfilt_e。

bEnable：使能或者禁止滤波功能，ENABLE/DISABLE。

2. 返回值：无

3. 参数说明

参数	说明	概述及其结构体定义位置

eFiltSel	typedef enum {
	EM_FILT_5NS = 0, //Pulse filtering < 5ns interval
	EM_FILT_10NS, //Pulse filtering < 10ns interval
	EM_FILT_15NS, //Pulse filtering < 15ns interval
	EM_FILT_20NS //Pulse filtering < 20ns interval
	} csi_emfilt_e;
		共有有四档选择
		在clk.h中定义

bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能滤波 DISABLE：禁止滤波 系统默认禁止 在common.h中定义

2.2.18 csi_esosc_filter_enable

void csi_esosc_filter_enable(bool bEnable)

2.2.18.1 功能描述

ESOSC 施密特滤波使能/禁止

2.2.18.2 参数/返回值说明

1. 参数

   bEnable：使能或者禁止滤波功能，ENABLE/DISABLE。

2. 返回值：无

3. 参数说明

参数	说明	概述及其结构体定义位置
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能滤波 DISABLE：禁止滤波 系统默认使能 在common.h中定义

2.2.19 csi_emosc_set_gain

void csi_emosc_set_gain(uint8_t byEmGain)

2.2.19.1 功能描述

设置 EMOSC 增益值

2.2.19.2 参数/返回值说明

1. 参数

   byEmGain：emosc 增益控制值

2. 返回值：无

3. 参数说明

参数	说明	概述及其结构体定义位置
byEmGain	Uint8_t 类型数值, byEmGain < 32;	

2.2.20 csi_esosc_set_gain

void csi_esosc_set_gain(uint8_t byEsGain)

2.2.20.1 功能描述

设置 ESOSC 增益值

2.2.20.2 参数/返回值说明

4. 参数

   byEsGain：emosc 增益控制值

5. 返回值：无

6. 参数说明

参数	说明	概述及其结构体定义位置
byEsGain	Uint8_t 类型数值, byEsGain < 16;	

2.2.21 csi_clk_calib

csi_error_t csi_clk_calib(void)

2.2.21.1 功能描述

系 时钟选择 IMOSC 或者 HFOSC 时，校准 IMCLK/HFCLK。

2.2.21.2 参数/返回值说明

1. 参数：无

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 返回值说明表

返回值	说明	枚举变量位置
csi_error_t	csi_error_t 中定义值	common.h

3 CORET

3.1 概述

系统定时器（CORET），24位循环递减计数器，用于系统计时，提供延时函数等。系统启动时默认打开，定时器默认配置为100Hz，即定时时间是10ms； APT CSI接口提供了TICK的相关配置和操作。

3.2 API列表

Table 3-1 CORET CSI接口函数

API	说明	函数位置
csi_tick_init	初始化tick	tickc
csi_tick_get	获取系统tick计数值	
csi_tick_get_ms	获取系统tick计时时间，单位：ms	
csi_tick_get_us	获取系统tick计时时间，单位：us	
csi_tick_attach_callback	注册tick中断处理回调函数	
mdelay	毫秒延时函数	
udelay	微秒延时函数	

3.3 API详细说明

3.3.1 csi_tick_init

csi_error_t csi_tick_init(void)

3.3.1.1 功能描述

初始化系统tick

3.3.1.2 参数/返回值说明

1. 参数

无参数，要修改tick定时时间，去修改宏定义CONFIG_SYSTICK_HZ的值，在tick.h中，默认100，单位Hz。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 返回值说明

返回值	说明	概述及其枚举定义位置
csi_error_t 中	csi_error_t 中定义值	在common.h中定义

3.3.2 csi_tick_get

uint32_t csi_tick_get(void)

3.3.2.1 功能描述

获取系统tick计数值

3.3.2.2 参数/返回值说明

1. 参数

无参数。

2. 返回值

tick计数值，即每到达tick的定时值，计数值加1。如，tick定时值为10ms，tick计数值每10ms加1。

3. 返回值说明

返回值	说明	概述
reruen value	uint32_t 类型数值	系统tick计数值

3.3.3 csi_tick_get_ms

uint32_t csi_tick_get_ms(void)

3.3.3.1 功能描述

获取系统tick运行时间

3.3.3.2 参数/返回值说明

1. 参数

无参数。

2. 返回值

系统tick运行时间，即系统tick总的运行时间，单位：ms

3. 返回值说明

返回值	说明	概述
reruen value	uint32_t 类型数值，系统tick运行时间	tick运行时间，单位：ms

3.3.4 csi_tick_get_us

uint64_t csi_tick_get_us(void)

3.3.4.1 功能描述

获取系统tick运行时间

3.3.4.2 参数/返回值说明

1. 参数

无参数

2. 返回值

系统tick运行时间，即系统tick总的运行时间，单位：us

3. 返回值说明

返回值	说明	概述
reruen value	uint64_t 类型数值，系统tick运行时间	tick运行时间，单位：us

3.3.5 csi_tick_attach_callback

*void csi_tick_attach_callback(void *callback)*

3.3.5.1 功能描述

注册tick中断处理回调函数，用户可注册一个函数，被注册的函数在tick中断中调用。

3.3.5.2 参数返回值说明

1. 参数

callback：回调函数指针，指向要注册的函数，回调函数详见结构体定义 csi_tick_t。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述

callback	`typedef struct {`
	`  void  (*callback)(void *pArg);`
	`} csi_tick_t;`
		回调函数参数为tick全局计数值，每个tick中断计数一次，给用户使用。
		tick.h中定义

3.3.6 mdelay

void mdelay(uint32_t ms)

3.3.6.1 功能描述

毫秒延时函数

3.3.6.2 参数/返回值说明

4. 参数

ms：延时时间，单位：ms

5. 返回值

无返回值。

6. 参数说明

参数	说明	概述
ms	uint32_t 类型数值，延时值	延时时间，单位：ms

3.3.7 udelay

void udelay(uint32_t us)

3.3.7.1 功能描述

微秒延时函数

3.3.7.2 参数/返回值说明

1. 参数

us：延时时间，单位：us；us应大于10us。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述
us	uint32_t 类型数值，延时值	延时时间，单位：us

4 系统可靠性

4.1 LVD

LVD模块包括LVD和LVR，分别是低电压检测和低电压复位的意思，这里的电压指的是MCU供电电压。LVD通常用于电平下降或上升时产生中断事件。LVR一旦使能，则会在电压下降到某个电平值后，使芯片进入复位状态，直到电压重新回到设定值以上。无论是LVR还是LVD，存在一个迟滞电压。具体参数请参考芯片的数据手册。

LVD和LVR可以设置为不同的电平值。LVD和LVR可以同时工作。

4.1.1 API列表

Table 4-1 系统可靠性 CSI接口函数

API	说明	函数位置
csi_lvd_int_enable	配置低电压检测的电平值，并使能中断	reliability.c
csi_lvd_disable	关闭低电压检测功能	
csi_get_lvdlevel	读取当前低电压检测的电平值	
csi_lvr_enable	配置低电压复位的电平值	
csi_lvr_disable	关闭低电压复位功能	
csi_get_lvrlevel	读取当前低电压复位的电平值	

4.1.2 API详细说明

4.1.2.1 csi_lvd_int_enable

void csi_lvd_int_enable(csi_lvd_pol_e ePol, csi_lvd_level_e eLvl)

4.1.2.1.1 功能描述

该函数实现了3个功能：配置低电压检测的电平值、设置中断事件发生极性（下降到预设值或上升到预设值），并使能中断。

4.1.2.1.2 参数/返回值说明

参数	说明	枚举量位置
ePol	可选值为csi_lvd_pol_e枚举值中的一个。 typedef enum {	reliability.h

	LVD_INTF = 1, //电压下降到预设值时产生事件
	LVD_INTR, //电压上升到预设值时产生事件
	LVD_INTFR //电压下降或上升到预设值时产生事件
	}clvd_pol_e;	

eLvl	可选值为csi_lvd_level_e枚举值中的一个。 typedef enum{ LVD_24 = 0, LVD_21, LVD_27, LVD_30, LVD_33, LVD_36, LVD_39, }clvd_level_e;	

4.1.2.2 csi_lvd_disable

void csi_lvd_disable(void)

4.1.2.2.1 功能描述

关闭芯片的低电压检测功能。注意，因为LVD和LVR实际是一个模块，所以关闭LVD的同时LVR功能也会被关闭。

4.1.2.2.2 参数/返回值说明

无。

4.1.2.3 csi_lvr_enable

void csi_lvr_enable(csi_lvr_level_e eLvl)

4.1.2.3.1 功能描述

该函数用于配置触发复位的低电压值。

4.1.2.3.2 参数/返回值说明

1. 参数：

eLvl：低电压复位的门限电平值。

2. 返回值：无。

3. 参数说明表

参数	说明	位置
eLvl	可选值为csi_lvr_level_e枚举值中的一个。 typedef enum { LVR_19 = 0, LVR_22, LVR_25, LVR_28, LVR_31, LVR_34, LVR_37, LVR_40 }clvr_level_e;	reliability.h

4.2 EMOSC 时钟监测

EMOSC时钟监测模块一旦使能，会持续监控外部EMOSC的状态。如果检测到振荡异常，可以有两种动作：复位，自动切换系统时钟到IMOSC，同时可以配置产生中断。

4.2.1 API列表

API	说明	函数位置
csi_emcm_2_imosc_int	使能EMCM功能，当监测到EMOSC异常时，将系统时钟切换到IMOSC，且触发中断。	reliability.c
csi_emcm_rst	使能EMCM功能，当监测到EMOSC异常时，系统复位。	
csi_emcm_disable	关闭EMCM功能。	

4.2.2 API详细说明

略。见API列表中的说明。

4.3 内存检验

内存校验模块一旦使能，可以对SRAM和Flash的内容进行实时的检测。可以设置允许的检验错误次数上限。如果SRAM内容错误次数超过上限，可以有两种动作：复位和中断事件。如果Flash内容错误次数超过上限，芯片会直接复位。

4.3.1 API列表

API	说明	函数位置

csi_sramcheck_set_times	设置SRAM校验允许的错误次数上限。复位值为0xffff。	reliability.c
csi_sramcheck_rst	设置SRAM校验错误次数超过设置的上限时，复位芯片。	
csi_sramcheck_int	设置SRAM校验错误次数超过设置的上限时，产生中断。	
csi_sramcheck_disable	禁止SRAM校验功能。	
csi_flashcheck_set_times	设置Flash校验允许的错误次数上限。复位值为0xffffff。	
csi_flashcheck_rst	设置Flash校验错误次数超过设置的上限时，复位芯片。	

4.3.2 API详细说明

略。见API列表中的说明。

4.4 复位源

芯片在每次复位的时候都会对复位的原因进行记录。

4.4.1 API列表

API	说明	函数位置
csi_get_rst_reason	获取上次复位的原因。	reliability.c
csi_clr_rst_reason	清除复位信息	

4.4.2 API详细说明

4.4.2.1 csi_get_rst_reason

uint16_t csi_get_rst_reason(void)

4.4.2.1.1 功能描述

返回上次复位的原因。该功能可用于定位导致异常复位的原因，也可以是系统可靠运行的一部分：不同的复位条件转向不同的复位流程。

4.4.2.1.2 参数/返回值说明

1. 参数：无
2. 返回值：

   返回chip复位原因，复位信息的MASK值，BIT0~14，复位源详情请参考csi_rsr_src_e枚举中定义。

3. 返回值说明表

返回值	说明	概述

复位信息	uint16_t 类型数值，复位信息的 MASK 值；具体复位信息，请参考 csi_rsr_src_e 中具体定义
	`typedef enum {`
	`  RST_SRC_POR = (0x01ul << 0),`
	`  RST_SRC_LVD = (0x01ul << 1),`
	`  RST_SRC_EXT = (0x01ul << 2),`
	`  RST_SRC_SHD_WKUP = (0x01ul << 3),`
	`  RST_SRC_IWDT = (0x01ul << 4),`
	`  RST_SRC_ESCM = (0x01ul << 5),`
	`  RST_SRC_EMCM = (0x01ul << 6),`
	`  RST_SRC_CPU = (0x01ul << 7),`
	`  RST_SRC_SW = (0x01ul << 8),`
	`  RST_SRC_CPUFAULT = (0x01ul << 9),`
	`  RST_SRC_RAMERR = (0x01ul << 11),`
	`  RST_SRC_EFLERR = (0x01ul << 12),`
	`  RST_SRC_WWDT = (0x01ul << 13),`
	`  RST_SRC_SNOOZE_WKUP = (0x01ul << 14)`
	`} csi_rsr_src_e;`	

4.4.2.2 csi_clr_rst_reason

void csi_clr_rst_reason(uint16_t hwRstSrc)

4.4.2.2.1 功能描述

清除 chip 复位信息。

4.4.2.2.2 参数/返回值说明

1. 参数

   hwRstSrc：复位信息的 MASK 值，BIT0~14，复位源详情请参考 csi_rsr_src_e 枚举中定义。

2. 返回值：无

3. 参数说明

参数	说明	概述
复位信息	uint16_t 类型数值，复位信息的 MASK 值	

4.5 CHIP软件复位

4.5.1 API列表

API	说明	函数位置
csi_sys_swrst	Chip软件复位	reliability.c

4.5.2 API详细说明

4.5.2.1 csi_sys_swrst

void csi_sys_swrst(void)

4.5.2.1.1 功能描述

软件复位 mcu。

4.5.2.1.2 参数/返回值说明

1. 参数：无
2. 返回值：无

4.6 UREG操作

4.6.1 API列表

API	说明	函数位置
csi_ureg_write	写用户寄存器	reliability.c
csi_ureg_read	读取用户寄存器	

4.6.2 API详细说明

4.6.2.1 csi_ureg_write

csi_error_t csi_ureg_write(csi_user_reg_e eUreg, uint32_t wValue)

4.6.2.1.1 功能描述

写用户寄存器，除 POR 复位外，其余原因导致的复位，寄存器值保持不变。

4.6.2.1.2 参数/返回值说明

1. 参数

   eUreg：用户寄存器选项，详见枚举定义 csi_user_reg_e

   wValue：用户写入寄存器值，eUreg = USER_REG0/ USER_REG1,为 32 位值，USER_REG2 时，为 16 位值。

2. 返回值：

   CSI_OK：初始化成功。

   CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	枚举量位置
eUreg	typedef enum{ USER_REG0 USER_REG1, USER_REG2, }csi_user_reg_e;	reliability.h
wValue	写入寄存器值，USER_REG0/ USER_REG1 为 32 位，其余为 16 位	
csi_error_t	csi_error_t 中定义值	在common.h中定义

4.6.2.2 csi_ureg_read

uint32_t csi_ureg_read(csi_user_reg_e eUreg)

4.6.2.2.1 功能描述

读取用 存寄储器除 POR 复位外，其余原因导致的复位，寄存器值保持不变。

4.6.2.2.2 参数/返回值说明

1. 参数

   eUreg：用户寄存器选项，详见枚举定义 csi_user_reg_e

2. 返回值：寄存器中存储值

3. 参数/返回值说明

参数/返回值	说明	枚举量位置

eUreg	typedef enum{ USER_REG0 USER_REG1, USER_REG2, }csi_user_reg_e;	reliability.h
返回值	寄存器存储值，USER_REG0/ USER_REG1 为32位，其余为16位	

5 PM

APT32F110x 支持四种低功耗模式，SLEEP、DEEPSLEEP、SNOOZE 和 SHUTDOWN。在 csi 驱动中，有一个全局的宏 CONFIG_USER_PM。一旦开启，就必须定义进出低功耗模式前后的自定义处理函数。

5.1 API列表

Table 5-1 PM CSI接口函数

API	说明	函数位置
csi_pm_attach_callback	设置进出某个低功耗模式前后的用户自定义处理函数。	pm.c
csi_pm_config_wakeup_source	设置唤醒源。	
csi_pm_enter_sleep	进入低功耗模式。	
csi_pm_snooze_power_enable	SNOOZE模式下使能LCD/HOUCH模块	
csi_pm_clk_enable	睡眠模式下使能/关闭时钟	
csi_pm_get_wkint	SHUTDOWN模式下获取外部IO的唤醒信息	
csi_pm_get_wkint	SHUTDOWN模式下清除外部IO的唤醒信息	

5.2 API详细说明

5.2.1 csi_pm_attach_callback

*void csi_pm_attach_callback(csi_pm_mode_e eMd, void *pBeforeSlp, void *pWkup)*

5.2.1.1 功能描述

这个函数受一个宏（CONFIG_USER_PM）控制。只有打开这个宏，函数才会加入编译。可以考虑在以下应用场景时打开这个宏：希望在系统进入低功耗模式前做一些状态和数据保存的操作，在系统从低功耗模式中恢复时恢复一些状态。建议在工程设置 compiler tab 下的 Define 中加入：CONFIG_USER_PM=1。

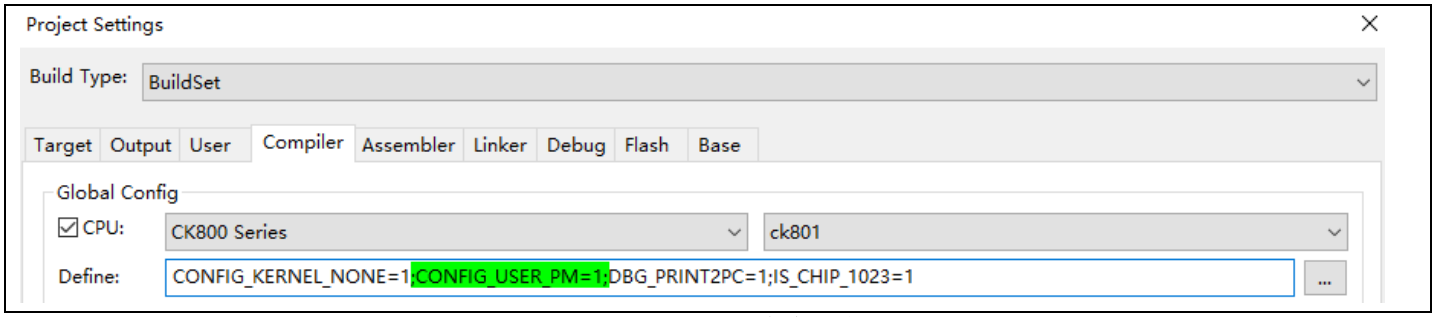


Figure 5-1 开启全局宏

5.2.1.2 参数/返回值说明

1. 参数：

   eMd：低功耗模式。

   pfBeforeSlp：进入低功耗模式前需调用的函数名。

   pfWkup：从低功耗模式出来后需调用的函数名。

2. 返回值：无。

3. 参数说明表

参数	说明	位置
eMd	可选值为csi_pm_mode_t枚举值中的一个。 typedef enum { PM_MODE_LPRUN = 0 PM_MODE_SLEEP, PM_MODE_DEEPSLEEP, PM_MODE_SNOOZE, PM_MODE_SHUTDOWN } csi_pm_mode_t;	pm.c
pfBeforeSlp	(void *)空指针。传入进入低功耗模式前需要调用函数（用户自定义）的名字。	
pfWkup	(void *)空指针。从低功耗模式退出后调用函数（用户自定义）的名字。	用户自定义函数位置

5.2.2 csi_pm_enter_sleep

csi_error_t csi_pm_enter_sleep(csi_pm_mode_e eMode)

5.2.2.1 功能描述

调用函数即进入指定的低功耗模式。如果工程打开了宏（CONFIG_USER_PM），如Figure 5-1所示，会在进出低

功耗模式前分别调用用户自定义函数。此时必须用csi_pm_attach_callback() 指定自定义函数，否则芯片会出现异常。如果不需要自定义函数，则需要关闭全局宏，即Figure 5-1中删除CONFIG_USER_PM=1。

5.2.2.2 参数/返回值说明

1. 参数：

   eMd：低功耗模式。

2. 返回值：

   csi_error_t型数据。当传入低功耗模式不在枚举变量范围时，会返回CSI_ERROR。

3. 参数说明表

参数	说明	位置
eMode	可选值为csi_pm_mode_e枚举值中的一个。 typedef enum { PM_MODE_LPRUN = 0 PM_MODE_SLEEP, PM_MODE_DEEPSLEEP, PM_MODE_SNOOZE, PM_MODE_SHUTDOWN } csi_pm_mode_e;	pm.h

5.2.3 csi_pm_config_wakeup_source

csi_error_t csi_pm_config_wakeup_source(wakeupn_type_e eWkupSrc, bool bEnable)

5.2.3.1 功能描述

设置将系统从 DEEP-SLEEP 模式中唤醒的源。

注：任何使能的中断都可以将系统从 SLEEP 模式中唤醒。无需在这里配置。

5.2.3.2 参数/返回值说明

1. 参数：

   eWkupSrc：选择唤醒源。

   bEnable：使能/禁止。

2. 返回值：

   csi_error_t型数据。当传入唤醒源不在枚举变量范围时，会返回CSI_ERROR。

3. 参数说明表

参数/返回值	说明	位置
eWkupSrc	可选值为wakeupn_type_e枚举值中的一个。 typedef enum { WKUP_ALV0 = 0, WKUP_ALV1, WKUP_ALV2, WKUP_ALV3, WKUP_IWDT = 8, WKUP_RTC, WKUP_LPT, WKUP_LVD, WKUP_TCH, WKUP_CMP } wakeupn_type_e;	pm.h
bEnable	ENABLE/DISABLE	
csi_error_t	csi_error_t 中定义值	csi_error_t

5.2.4 csi_pm_snooze_power_enable

csi_error_t csi_pm_snooze_power_enable(csi_snooze_power_e ePower, bool bEnable)

5.2.4.1 功能描述

SNOOZE 模式下，使能/关闭 LCD/TOUCH 模块。

5.2.4.2 参数/返回值说明

1. 参数

   1. ePower：选择 TOUCH 或者 LCD 模块，枚举定义详见 csi_snooze_power_e。
   2. bEnable：使能/禁止（ENABLE/DISABLE）

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明表

参数/返回值	说明	位置
ePower	可选值为csi_snooze_power_e枚举值中的一个。	pm.h

	typedef enum { SNOOZE_TOUCH_POWER = 0, SNOOZE_LCD_POWER } csi_snooze_power_e;	
bEnable	Bool 类型数值，ENBALE/DISABLE	common.h
csi_error_t	csi_error_t 中定义值	

5.2.5 csi_pm_clk_enable

void csi_pm_clk_enable(csi_pm_clk_e eOsc, bool bEnable)

5.2.5.1 功能描述

睡眠模式下使能/禁止某时钟。

5.2.5.2 参数/返回值说明

1. 参数

   3. eOsc：选择某个时钟，枚举定义详见 csi_pm_clk_e。

   4. bEnable：使能/禁止（ENABLE/DISABLE）

2. 返回值：无。

3. 参数/返回值说明表

参数	说明	位置

ePower	可选值为csi_snooze_power_e枚举值中的一个。
	typedef enum {
	SP_IDLE_PCLK = (0x01ul << 8), ///< IDLE_PCLK for sleep
	SP_IDLE_HCLK = (0x01ul << 9), ///< DLE_HCLK for sleep
	DP_ISOSC = (0x01ul << 12), ///< ISOSC for deepslee
	DP_IMOSC = (0x01ul << 13), ///< ISOSC for deepsleep
	DP_ESOSC = (0x01ul << 14), ///< ISOSC for deepsleep
	DP_EMOSC = (0x01ul << 15) ///< ISOSCfordeepsleep
	} csi_pm_clk_e;
		pm.h

bEnable	Bool 类型数值，ENBALE/DISABLE	common.h

5.2.6 csi_pm_get_wkint

uint8_t csi_pm_get_wkint(void)

5.2.6.1 功能描述

SHUTDOWN 模式下，获取外部 IO 唤醒源信息。

SHUTDOWN 模式下支持 4 个特殊 IO 口唤醒，PA00/PB00/PA12/PB011，唤醒后可获取唤醒源，即哪个 IO 唤醒。

5.2.6.2 参数/返回值说明

1. 参数：无。
2. 返回值

   外部唤醒源信息，BIT0~3 的 MASK 值；BIT0~3 表示对应的 4 个唤醒源。

3. 返回值说明表

返回值	说明	位置
唤醒源	uint8_t 类型的 MASK 值，BIT0~3 表示对应的 4 个唤醒源	pm.h

5.2.7 csi_pm_clr_wkint

void csi_pm_clr_wkint(uint8_t byWkInt)

5.2.7.1 功能描述

SHUTDOWN 模式下，外部 IO 唤醒时，需清除对应的唤醒状态信息。

5.2.7.2 参数/返回值说明

1. 参数

   byWkInt：唤醒源状态，BIT0~3 的 MASK 值，BIT0~3 表示对应的 4 个唤醒源。

2. 返回值：无。
3. 参数说明表

参数	说明	位置
唤醒源	uint8_t 类型的 MASK 值，BIT0~3 表示对应的 4 个唤醒源	pm.h

6 CRC

ATP CSI接口CRC的设计中，提供CRC模块的初始化、CRC-CCITT、CRC-16、CRC-32计算函数接口。

6.1 API列表

Table6-1CRCCSI接口函数

API	说明	函数位置
csi_crc_init	CRC模块初始化	crc.c
csi_crc_rst	CRC模块复位	
csi_crc16	CRC-16转换函数	
csi_crc16_ccitt	CRC-16/CCITT转换函数	
csi_crc16_itu	CRC-16 XMODEM转换函数	
csi_crc32_be	CRC-32转换函数	

6.2 API详细说明

6.2.1 csi_crc_init

void csi_crc_init(void)

6.2.1.1 功能描述

CRC模块初始化。

6.2.1.2 参数/返回值说明

1. 参数：无。
2. 返回值：无。

6.2.2 csi_crc_rst

void csi_crc_rst(void)

6.2.2.1 功能描述

软件复位CRC模块。

6.2.2.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：无返回值。

6.2.3 csi_crc16

uint16_t csi_crc16(uint16_t hwCrcSeed, uint8_t pbyData, uint32_t wSize)*

6.2.3.1 功能描述

CRC-16模式计算。

6.2.3.2 参数/返回值说明

1. 参数

   hwCrcSeed：CRC计算种子值。

   pbyData：需要计算的数据指针。

   wSize：需要计算的数据长度。

2. 返回值

   CRC计算结果值。

6.2.4 csi_crc16_ccitt

*uint16_t csi_crc16_ccitt(uint16_t hwCrcSeed, uint8_t *pbyData, uint32_t wSize)*

6.2.4.1 功能描述

CRC-16/CCITT模式计算。

6.2.4.2 参数/返回值说明

1. 参数

hwCrcSeed：CRC计算种子值。

pbyData：需要计算的数据指针。

wSize：需要计算的数据长度。

2. 返回值

CRC计算结果值。

6.2.5 csi_crc16_itu

uint16_t csi_crc16_itu(uint16_t hwCrcSeed, uint8_t pbyData, uint32_t wSize)*

6.2.5.1 功能描述

CRC-16 XMODEM模式计算。

6.2.5.2 参数/返回值说明

1. 参数

hwCrcSeed：CRC计算种子值。

pbyData：需要计算的数据指针。

wSize：需要计算的数据长度。

2. 返回值

CRC计算结果值。

6.2.6 csi_crc32_be

uint32_t csi_crc32_be(uint32_t wCrcSeed, uint8_t pbyData, uint32_t wSize)*

6.2.6.1 功能描述

CRC-32模式计算

6.2.6.2 参数/返回值说明

1. 参数

   hwCrcSeed：CRC计算种子值。

   pbyData：需要计算的数据指针。

   wSize：需要计算的数据长度。

2. 返回值

7 HWDIV

除法器的使用实际上包含两种，一种是系统自带的除法器，另外一种是硬件除法器。

命名规则：

- s – 单（精度，字？）四个字节整数（int）/花车（float）
- d – 双（精度）八个字节整数（long 或 long long）/浮（double）
- t – 10 个字节为整数（long long）/浮标（long double）

此命名用于所有算术算法，如__divsi3,__divdi3,__divti3 或__mulsi3,__muldi3,__multi3 ...（和所有 u – 无符号变量）

驱动里，我们对库函数实现了重写，当使用"/"时，编译器会使用我们写的__udivsi3 之类的函数。

1）如果不重写，除法是库函数用软件的方式实现的，此时进出中断会 push 和 pop，现场会被保护住

2）重写后，这种保护最简单的实现方法就是：除法函数里屏蔽中断

API	说明	函数位置
__divsi3	32位带符号除法器，计算商(可选，系统自带或硬件除法器)	hwdiv.c
__modsi3	32位带符号除法器，计算余数(可选，系统自带或硬件除法器)	
__udivsi3	32位无符号除法器，计算商(可选，系统自带或硬件除法器)	
__umodsi3	32位无符号除法器，计算余数(可选，系统自带或硬件除法器)	
__divdi3	64位带符号除法器，计算商(目前为系统自带的除法器实现)	
__udivdi3	64位无符号除法器，计算商(目前为系统自带的除法器实现)	
__moddi3	64位带符号除法器，计算余数(目前为系统自带的除法器实现)	
__umoddi3	64位无符号除法器，计算余数(目前为系统自带的除法器实现)	

7.1 API详细说明

7.1.1 __divsi3

int __divsi3(int wDividend, int wDivisor)

7.1.1.1 功能描述

带符号除法器，计算商。

7.1.1.2 参数/返回值说明

1. 参数

wDividend：32位带符号的被除数。

wDivisor：32位带符号的除数。

2. 返回值：商值。

3. 参数/返回值说明

参数	说明	概述及其枚举定义位置
wDividend	int 类型的被除数	
wDivisor	int 类型的除数	
load value	返回计数器加载值，int类型	

7.1.2 __ modsi3

int __modsi3 (int wDividend, int wDivisor)

7.1.2.1 功能描述

带符号除法器，计算余数。

7.1.2.2 参数/返回值说明

1. 参数

wDividend：32位带符号的被除数。

wDivisor：32位带符号的除数。

2. 返回值：余数。

3. 参数/返回值说明

参数	说明	概述及其枚举定义位置
wDividend	int 类型的被除数	
wDivisor	int 类型的除数	
load value	返回计数器加载值，int类型	

7.1.3 __udivsi3

unsigned int __udivsi3(unsigned int wDividend, unsigned int wDivisor)

7.1.3.1 功能描述

无符号除法器，计算商。

7.1.3.2 参数/返回值说明

1. 参数

wDividend：32位无符号的被除数。

wDivisor：32位无符号的除数。

2. 返回值：商值。

3. 参数/返回值说明

参数	说明	概述及其枚举定义位置
wDividend	unsigned int 类型的被除数	
wDivisor	unsigned int 类型的除数	
load value	返回计数器加载值，unsigned int类型	

7.1.4 __umodsi3

unsigned int __umodsi3(unsigned int wDividend, unsigned int wDivisor)

7.1.4.1 功能描述

无符号除法器，计算余数。

7.1.4.2 参数/返回值说明

1. 参数

   wDividend：32位无符号的被除数。

   wDivisor：32位无符号的除数。

2. 返回值：余数。

3. 参数/返回值说明

参数	说明	概述及其枚举定义位置
wDividend	unsigned int 类型的被除数	
wDivisor	unsigned int 类型的除数	
load value	返回计数器加载值，unsigned int类型	

7.2 注意事项

64 位除法器占空空间较大，以__udivdi3 除法函数举例，CK802 内核中，该函数占用空间大概 950 个字节，E902 内核，该函数占用空间大概 1150 个字节，在条件允许的情况下，尽量使用 32 位的除法器。

8 IFC

APT32F102x 的内部 flash 包括两个区域，一个是 64K/32K 的 PFLASH，用于程序代码的存储，一个是 4K 的 DFLASH，用于用户数据的存储。这两个区域最大的区别是扇区大小不同。

无论是 DFLASH 还是 PFLASH，一次写操作之前必需有一次擦除操作（以扇区为单位）。从 FLASH 寿命角度考虑，擦和写的次数需要对等。正是出于这个考虑，驱动不再支持单独的擦除操作，写操作函数中实现先擦除再写。

8.1 API列表

Table 8-1 IFC CSI接口函数

API	说明	函数位置
csi_ifc_read	读取flash区域的内容，支持PFLASH和DFLASH	ifc.c
csi_ifc_program	往flash区域写入内容，支持PFLASH和DFLASH	
csi_ifc_get_status	获得flash状态（error, busy）	
csi_ifc_erase	Flash页擦除，支持PFLASH和DFLASH	

8.2 API详细说明

8.2.1 csi_ifc_read

*csi_error_t csi_ifc_read(csp_ifc_t *ptIfcBase, uint32_t wAddr, uint32_t *wData, uint32_t wDataNum)*

8.2.1.1 功能描述

读取 flash 区域的内容，支持 PFLASH 和 DFLASH。发生以下情况返回错误：

1. 传入地址没有字对齐。
2. 地址不在系统 flash（PFLASH+DFLASH）空间

8.2.1.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h

wAddr	读取Flash数据的首址	
wData	指向目标数据首址的指针	
wDataNum	一次读取数据的长度（单位：byte）	

8.2.1.3 返回值说明

返回值类型	说明	位置
csi_error_t	CSI_ERROR/CSI_OK	common.h

8.2.2 csi_ifc_program

*csi_error_t csi_ifc_program(csp_ifc_t *ptIfcBase, uint32_t wAddr, uint32_t *pwData, uint32_t wDataNum)*

8.2.2.1 功能描述

往 flash 区域写入内容，带校验功能。支持 PFLASH 和 DFLASH。发生以下情况返回错误：

- 传入地址没有字对齐
- 地址不在系统 flash（PFLASH+DFLASH）空间
- 校验失败

8.2.2.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h
wAddr	操作Flash的目标首址	
wData	指向源数据首址的指针	
wDataNum	一次写入的数据的长度（单位：byte）	

8.2.2.3 返回值说明

返回值类型	说明	位置
csi_error_t	CSI_ERROR/CSI_OK	common.h

8.2.3 csi_ifc_get_status

*csi_ifc_status_t csi_ifc_get_status(csp_ifc_t *ptIfcBase)*

8.2.3.1 功能描述

读取 Flash 的状态。

8.2.3.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h

8.2.3.3 返回值说明

返回值类型	说明	位置

csi_ifc_status_t	typedef struct {
	uint32_t busy : 1; // busy 状态
	uint32_t error : 1; //错误状态（IFC_RISR[xxx_ERR]）标志
	} csi_ifc_status_t;
		ifc.h

8.2.4 csi_ifc_page_erase

*csi_error_t csi_ifc_page_erase(csp_ifc_t *ptIfcBase, uint32_t wPageStAddr)*

8.2.4.1 功能描述

Flash 擦除，按页擦除，支持 PFLASH 和 DFLASH；发生以下情况返回错误：

- 传入地址不在 PFASH 或者 DFLASH 范围
- 传入地址不是要擦除页(起始页)的起始地址

8.2.4.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h

wPageStAddr	要擦除页的起始地址	

8.2.4.3 返回值说明

返回值类型	说明	位置
csi_error_t	CSI_ERROR/CSI_OK	common.h

8.2.5 csi_ifc_pflash_page_program

*csi_error_t csi_ifc_pflash_page_program(csp_ifc_t *ptIfcBase, uint32_t wAddr, uint32_t *pwData, uint32_t wDataWordNum)*

8.2.5.1 功能描述

往 pflash 区域写入内容，写入格式为 word（4byte）

- 传入地址不在 PFASH 范围
- 传入 num 字节数不是 word（4byte）的整数倍

8.2.5.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h
wAddr	写入的地址	
pwData	要写入数据的缓存首地址	
wDataWordNum	要写入的word数	

8.2.5.3 返回值说明

返回值类型	说明	位置
csi_error_t	CSI_ERROR/CSI_OK	common.h

8.2.6 csi_ifc_dflash_page_program

*csi_error_t csi_ifc_dflash_page_program(csp_ifc_t *ptIfcBase, uint32_t wAddr, uint32_t *pwData, uint32_t wDataWordNum)*

8.2.6.1 功能描述

往 dflash 区域写入内容，写入格式为 word（4byte）

- 传入地址不在 DFASH 范围
- 传入 num 字节数不是 word（4byte）的整数倍

8.2.6.2 参数说明

参数	说明	位置
ptIfcBase	指向IFC控制寄存器结构体的指针，用于进行寄存器操作。	csp_ifc.h
wAddr	写入的地址	
pwData	要写入数据的缓存首地址	
wDataWordNum	要写入的word数	

8.2.6.3 返回值说明

返回值类型	说明	位置
csi_error_t	CSI_ERROR/CSI_OK	common.h

9 IWDT

9.1 概述

嵌入式系统中，为了使系统在异常情况下能自动复位，保证系统健壮性，一般都需要引入看门狗。IWDT是一个独立看门狗，它的时钟源是ISOSC。上电时IWDT是否使能由User Option的高16位决定，可以通过读取SYSCON_OPT0[IWDTEN]获得User Option中对应的缺省状态。

IWDT运行后计数器开始计数，计数器溢出前没有被复位，计数器溢出会对MCU产生复位信号使系统复位。系统正常运行时，需要在计数器溢出前对计数器清零(喂狗)，不让复位产生。如系统正常运行，喂狗正常。一旦程序异常，不能正常喂狗，则系统复位。

在默认情况下，IWDT是使能的，如果没有喂狗的操作，会在8s后复位芯片。

当IWDT工作时，ISOSC缺省使能。任何尝试关闭ISOSC的操作都会触发命令错误中断。

APT CSI接口提供了IWDT相关配置和操作。

9.2 API列表

Table 9-1 IWDT CSI接口函数

API	说明	函数位置
csi_iwdt_init	初始化看门狗	iwdt.c
csi_iwdt_open	打开看门狗(开始工作)	
csi_iwdt_close	关闭看门狗(停止工作)	
csi_iwdt_feed	看门狗喂狗	
csi_iwdt_irq_enable	使能看门狗中断	
csi_iwdt_is_running	检测看门狗工作状态	
csi_iwdt_get_remaining_time	获取看门狗复位剩余时间	
csi_iwdt_debug_enable	看门狗debug模式使能	

9.3 API详细说明

9.3.1 csi_iwdt_init

csi_error_t csi_iwdt_init(csi_iwdt_to_e eTimeOut)

9.3.1.1 功能描述

初始化看门狗。

9.3.1.2 参数/返回值说明

1. 参数

eTimeOut：看门狗溢出时间，枚举定义详见csi_iwdt_to_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
eTimeOut	typedef enum{ IWDT_TO_128 = 0, //128ms IWDT_TO_256, //256ms IWDT_TO_512, //512ms IWDT_TO_1024, //1024ms IWDT_TO_2048, //2048ms IWDT_TO_3072, //3072ms IWDT_TO_4096, //4096ms IWDT_TO_8192 //8192ms }csi_iwdt_to_e;	看门狗溢出(系统复位)时间，有8档选择，在iwdt.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

9.3.2 csi_iwdt_open

void csi_iwdt_open(void)

9.3.2.1 功能描述

打开看门狗，即启动看门狗。

9.3.2.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：无返回值。

9.3.3 csi_iwdt_close

void csi_iwdt_close(void)

9.3.3.1 功能描述

关闭看门狗，即停止看门狗

9.3.3.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：无返回值。

9.3.4 csi_iwdt_feed

void csi_iwdt_feed(void)

9.3.4.1 功能描述

看门狗喂狗函数

9.3.4.2 参数/返回值说明

1. 参数：无参数
2. 返回值：无返回值

9.3.5 csi_iwdt_irq_enable

void csi_iwdt_irq_enable(csi_iwdt_alarm_e eAlarmTo, bool bEnable)

9.3.5.1 功能描述

使能看门狗报警中断。

9.3.5.2 参数/返回值说明

1. 参数

eAlarmTo：报警中断时间，枚举定义详见csi_iwdt_alarm_e。

bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
eAlarmTo	typedef enum{ IWDT_ALARMTO_1_8 = 0, IWDT_ALARMTO_2_8, IWDT_ALARMTO_3_8, IWDT_ALARMTO_4_8, IWDT_ALARMTO_5_8, IWDT_ALARMTO_6_8, IWDT_ALARMTO_7_8 }csi_iwdt_alarm_e;	看门狗报警中断时间有7档选择，即占总溢出时间的n/8，n的范围：1~7 在iwdt.h中定义
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

9.3.6 csi_iwdt_is_running

bool csi_iwdt_is_running(void)

9.3.6.1 功能描述

检测看门狗工作状态。

9.3.6.2 参数/返回值说明

1. 参数：无参数。
2. 返回值

ture：正在工作。

false：停止工作。

3. 返回值说明

返回值类型	说明	概述
bool	Ture：false(1/0)	ture：数值为1(真) false：数值为0(假)

9.3.7 csi_iwdt_get_remaining_time

uint32_t csi_iwdt_get_remaining_time(void)

9.3.7.1 功能描述

获取看门狗复位剩余时间

9.3.7.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：

复位剩余时间，单位ms。

3. 返回值说明

返回值类型	说明	概述
return value	uint32_t 类型数值	复位剩余时间，单位ms

9.3.8 csi_iwdt_debug_enable

void csi_iwdt_debug_enable(bool bEnable)

9.3.8.1 功能描述

使能看门狗debug模式

9.3.8.2 参数/返回值说明

1. 参数

bEnable：使能/禁止debug模式，默认禁止，ENABLE/DISABLE。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其定义位置
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义

10 WWDT

10.1 概述

窗口型看门狗，作为可靠性保护逻辑，用于监测当前程序运行状况。当外部干扰或不可预见的逻辑错误发生时，造成当前程序运行错误，看门狗逻辑可以在预设时间周期结束时产生系统复位信号。看门狗计数器通过软件刷新防止溢出而产生复位，刷新必须在预设的时间窗口没进行才有效，否则刷新事件也会触发系统复位信号。看门狗一旦开启，不能被关闭，除非系统复位，系统复位时看门狗默认关闭。APT CSI接口提供了WWDT相关配置和操作。

Table 10-1 WWDT CSI接口函数

API	说明	函数位置
csi_wwdt_init	初始化看门狗	iwdt.c
csi_wwdt_set_window_time	设置窗口时间	
csi_iwdt_open	打开看门狗(开始工作)	
csi_wwdt_feed	看门狗喂狗	
csi_wwdt_irq_enable	使能看门狗中断	
csi_wwdt_is_running	检测看门狗工作状态	
csi_wwdt_get_remaining_time	获取看门狗复位剩余时间	
csi_wwdt_debug_enable	看门狗debug模式使能	

10.2 API详细说明

10.2.1 csi_wwdt_init

csi_error_t csi_wwdt_init(uint32_t wTimeOut)

10.2.1.1 功能描述

初始化看门狗

10.2.1.2 参数/返回值说明

1. 参数

wTimeOut：看门狗溢出时间，单位ms。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
wTimeOut	uint32_t 类型数值，单位：ms	看门狗溢出(系统复位)时间，单位ms
csi_error_t	csi_error_t 中定义值	在common.h中定义

10.2.2 csi_wwdt_set_window_time

csi_error_t csi_wwdt_set_window_time(uint32_t wTimeOut)

10.2.2.1 功能描述

设置看门狗窗口时间

10.2.2.2 参数/返回值

1. 参数

wTimeOut：看门狗窗口时间，单位ms。

2. 返回值

CSI_OK：设置成功。

CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
wTimeOut	uint32_t 类型数值，单位：ms	看门狗窗口时间，单位ms
csi_error_t	csi_error_t中定义值	在common.h中定义

10.2.3 csi_wwdt_open

void csi_wwdt_open(void)

10.2.3.1 功能描述

打开看门狗，即启动看门狗

10.2.3.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：无返回值。

10.2.4 csi_wwdt_feed

void csi_wwdt_feed(void)

10.2.4.1 功能描述

看门狗喂狗函数

10.2.4.2 参数/返回值说明

1. 参数：无参数。
2. 返回值：无返回值。

10.2.5 csi_wwdt_irq_enable

void csi_wwdt_irq_enable(bool bEnable)

10.2.5.1 功能描述

使能看门狗报警中断。

10.2.5.2 参数/返回值说明

1. 参数

   bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其定义位置
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

10.2.6 csi_wwdt_is_running

bool csi_wwdt_is_running(void)

10.2.6.1 功能描述

检测看门狗工作状态

10.2.6.2 参数/返回值说明

1. 参数：无参数。
2. 返回值

   ture：正在工作。

   false：停止工作。

3. 返回值说明

返回值	说明	概述
return value	Bool 类型数值，ture/false(1/0)	ture：数值为1(真) false：数值为0(假)

10.2.7 csi_wwdt_get_remaining_time

uint32_t csi_wwdt_get_remaining_time(void)

10.2.7.1 功能描述

获取看门狗复位剩余时间

10.2.7.2 参数/返回值说明

1. 参数：无参数。

2. 返回值

复位剩余时间，单位ms。

3. 返回值说明

返回值	说明	概述
return value	uint32_t 类型数值，单位：ms	复位剩余时间，单位ms

10.2.8 csi_wwdt_debug_enable

void csi_wwdt_debug_enable(bool bEnable)

10.2.8.1 功能描述

使能看门狗debug模式

10.2.8.2 参数/返回值说明

1. 参数

bEnable：使能/禁止debug模式，默认禁止，ENABLE/DISABLE。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其定义位置
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义

11 ETCB

11.1 概述

ETCB的功能是实现一个IP触发另一个IP，相当于在两个不同的外设之间架起了一座桥。这种硬件的桥接，大大增加了系统的反应速度，同时减少了CPU处理中断的压力。

API CSI接口函数提供了相关配置和操作。

11.2 API列表

Table 11-1 ETCB CSI接口函数

API	说明	函数位置
csi_etb_init	初始化ETB	etb.c
csi_etb_ch_alloc	申请一个ETB通道	
csi_etb_ch_free	释放一个ETB通道	
csi_etb_ch_config	配置ETB通道	
csi_etb_ch_swtrig	软件触发ETB通道	
csi_etb_ch_start	启动(使能)ETB通道	
csi_etb_ch_stop	关闭(禁止)ETB通道	

11.3 API详细说明

11.3.1 csi_etb_init

void csi_etb_init(void)

11.3.1.1 功能描述

初始化ETB模块(使能模块)

11.3.1.2 参数/返回值说明

1. 参数：无参数。

2. 返回值：无返回值。

3. 参数/返回值说明

11.3.2 csi_etb_ch_alloc

int32_t csi_etb_ch_alloc(csi_etb_ch_type_e eChType)

11.3.2.1 功能描述

申请一个ETB通道

11.3.2.2 参数/返回值说明

1. 参数

eChType：ETB通道类型，枚举详见csi_etb_ch_type_e。

2. 返回值

通道号/错误信息。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举定义位置

eChType	typedef enum {
	ETB_ONE_TRG_ONE = 0, //one device trig one deivce
	ETB_ONE_TRG_MORE, //one device trig two for more device
	ETB_MORE_TRG_ONE //two or more device trig one deivce
	} csi_etb_ch_type_e;
		有三种通道类型：
		单个源触发单个目标(通道3~7)
		单个源触发多个目标(通道1~2)
		多个源触发单个目标(通道0)
		在etb.h中定义

return value	int32_t 类型数值，范围：-1 ~ 7	-1：获取通道失败 0~7：为通道号，申请成功

11.3.3 csi_etb_ch_free

void csi_etb_ch_free(csi_etb_chid_e eChId)

11.3.3.1 功能描述

释放一个ETB通道

11.3.3.2 参数/返回值说明

1. 参数

eChId：ETB通道ID号，枚举详见csi_etb_chid_e。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
eChId	typedef enum { ETB_CH0_ID = 0, ETB_CH1_ID, ETB_CH2_ID, ETB_CH3_ID, ETB_CH4_ID, ETB_CH5_ID, ETB_CH6_ID, ETB_CH7_ID } csi_etb_chid_e;	ETB有8个通道： ETB_CH0_ID ~ ETB_CH7_ID 数值：0~7 在etb.h中定义

11.3.4 csi_etb_ch_config

*csi_error_t csi_etb_ch_config(csi_etb_chid_e eChId, csi_etb_config_t *ptConfig)*

11.3.4.1 功能描述

配置ETB通道。

11.3.4.2 参数/返回值说明

1. 参数

eChId：ETB通道ID号，枚举详见csi_etb_chid_e。

ptConfig：ETB通道配置参数结构体指针，结构体详见csi_etb_config_t。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
eChId	typedef enum { ETB_CH0_ID = 0, ETB_CH1_ID, ETB_CH2_ID, ETB_CH3_ID, ETB_CH4_ID, ETB_CH5_ID, ETB_CH6_ID, ETB_CH7_ID } csi_etb_chid_e;	ETB有8个通道： ETB_CH0_ID ~ ETB_CH7_ID 数值：0~7 在etb.h中定义

ptConfig	typedef struct { uint8_t bySrcIp; uint8_t bySrcIp1; uint8_t bySrcIp2; uint8_t byDstIp; uint8_t byDstIp1; uint8_t byDstIp2; uint8_t byTrgMode; uint8_t byChType; } csi_etb_config_t;	8个配置参数： bySrcIp：触发源0事件 bySrcIp1：触发源1事件 bySrcIp2：触发源2事件 byDstIp：触发目标0事件 byDstIp1：触发目标1事件 byDstIp2：触发目标2事件 byTrgMode：触发模式 byChType：通道类型 触发模式：硬件、软件两种模式；详见枚举csi_etb_trg_mode_e 通道类型：参阅7.3.2.2参数说明
csi_error_t	csi_error_t 中定义值	在common.h中定义

11.3.5 csi_etb_ch_swtrig

void csi_etb_ch_swtrig(csi_etb_chid_e eChId)

11.3.5.1 功能描述

软件触发ETB通道

11.3.5.2 参数/返回值说明

1. 参数

   eChId：ETB通道ID号，枚举详见csi_etb_chid_e。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
eChId	typedef enum { ETB_CH0_ID = 0, ETB_CH1_ID, ETB_CH2_ID, ETB_CH3_ID, ETB_CH4_ID, ETB_CH5_ID, ETB_CH6_ID, ETB_CH7_ID } csi_etb_chid_e;	ETB有8个通道： ETB_CH0_ID ~ ETB_CH7_ID 数值：0~7 在etb.h中定义

11.3.6 csi_etb_ch_start

void csi_etb_ch_start(csi_etb_chid_e eChId)

11.3.6.1 功能描述

启动(使能)ETB通道

11.3.6.2 参数/返回值说明

1. 参数

   eChId：ETB通道ID号，枚举详见csi_etb_chid_e。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
eChId	typedef enum { ETB_CH0_ID = 0, ETB_CH1_ID, ETB_CH2_ID, ETB_CH3_ID, ETB_CH4_ID, ETB_CH5_ID, ETB_CH6_ID, ETB_CH7_ID } csi_etb_chid_e;	ETB有8个通道： ETB_CH0_ID ~ ETB_CH7_ID 数值：0~7 在etb.h中定义

11.3.7 csi_etb_ch_stop

void csi_etb_ch_stop(csi_etb_chid_e eChId)

11.3.7.1 功能描述

关闭(禁止)ETB通道

11.3.7.2 参数/返回值说明

1. 参数

   p eChId：ETB通道ID号，枚举详见csi_etb_chid_e。

2. 返回值：无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
eChId	typedef enum { ETB_CH0_ID = 0, ETB_CH1_ID, ETB_CH2_ID, ETB_CH3_ID, ETB_CH4_ID, ETB_CH5_ID, ETB_CH6_ID, ETB_CH7_ID } csi_etb_chid_e;	ETB有8个通道： ETB_CH0_ID ~ ETB_CH7_ID 数值：0~7 在etb.h中定义

12 ADC

12.1 概述

ADC是Analog-to-Digital Converter的缩写。指模/数转换器或者模拟/数字转换器，是指将连续变量的模拟信号转换为离散的数字信号的器件，改ADC位12位ADC。APT CSI接口提供了ADC包括轮询、中断模式，单通道以及多通道采样相关配置和操作。

12.2 API列表

Table 12-1 时钟CSI接口函数

API	说明	函数位置
csi_adc_init	初始化ADC	adc.c
csi_adc_set_seqx	配置ADC转换序列	
csi_adc_set_buffer	配置ADC采样数据缓存(buffer)	
csi_adc_start	启动AD转换	
csi_adc_stop	停止AD转换	
csi_adc_conv_mode	配置ADC转换模式	
csi_adc_conv_pri	配置ADC序列转换优先级	
csi_adc_read_channel	获取ADC转换序列某通道数据	
csi_adc_read_seqx	获取ADC转换序列所有通道数据	
csi_adc_set_vref	配置ADC参考电压	
csi_adc_freq_div	配置ADC采样分频系数	
csi_adc_get_freq	获取ADC采样频率	
csi_adc_int_enable	使能ADC中断	
csi_adc_set_cmp0	配置ADC采样比较寄存器0比较功能	
csi_adc_set_cmp1	配置ADC采样比较寄存器1比较功能	
csi_adc_get_status	获取ADC数据转换状态	
csi_adc_clr_status	清除ADC数据转换状态	
csi_adc_set_sync	配置ADC外部同步触发输入	
csi_adc_rearm_sync	设置同步触发模式自动REARM	

csi_adc_set_evtrg	配置ADC事件触发输出	
csi_adc_fvrout_enable	使能ADC固定参考电压源	
csi_adc_bufout_enable	使能ADC 内部电压输出（1v或者温度传感器电压）	

12.3 API详细说明

12.3.1 csi_adc_init

*csi_error_t csi_adc_init(csp_adc_t *ptAdcBase, csi_adc_config_t *ptAdcCfg)*

12.3.1.1 功能描述

初始化ADC

12.3.1.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

ptAdcCfg：ADC配置结构体指针，结构体定义详见csi_adc_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置

ptAdcBase	参数：ADC0
	csp_adc_t *ADC0 = (csp_adc_t *)(APB_ADC0_BASE);
	typedef struct
	{
	__OM uint32_t ECR;
	__OM uint32_t DCR;
	__IM uint32_t PMSR;
	__IM uint32_t RSVD0;
	__OM uint32_t CR;
	__IOM uint32_t MR;
	__IOM uint32_t SHR;
	__OM uint32_t CSR;
	__IM uint32_t SR;
	__OM uint32_t IER;
	__OM uint32_t IDR;
	__IM uint32_t IMR;
	__IOM uint32_t SEQ[16];
		系统有一个ADC，即ADC(0)，定义了对应的结构体指针ADC0，指向系统ADC基地址。
		ADC0的指针定义在devices.c,指针类型定义csp_adc.h

	__IOM uint32_t PRI; __IOM uint32_t TDL0; __IOM uint32_t TDL1; __IOM uint32_t SYNCR; __IOM uint32_t TRGFCR; __IOM uint32_t TRGFWR; __IOM uint32_t EVTRG; __IOM uint32_t EVPS; __IOM uint32_t EVSWF; __IOM uint32_t RSVD2[27]; __IM uint32_t DR[16]; __IOM uint32_t CMP0; __IOM uint32_t CMP1; __IOM uint32_t DRMASK; } csp_adc_t;	
ptAdcCfg	typedef struct { uint8_t byClkDiv; uint8_t bySampHold; uint8_t byConvMode; uint8_t byVrefSrc; uint32_t wInt; csi_adc_seq_t *ptSeqCfg; } csi_adc_config_t;	初始化配置参数： byClkDiv：分频系数 bySampHold：采样保持时间 byConvMode：转换模式(连续/单次) byVrefSrc：参考电压源 wInter：中断选择 ptSeqCfg：csi_adc_seq_t类型指针 在adc.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.2 csi_adc_set_seqx

*csi_error_t csi_adc_set_seqx(csp_adc_t *ptAdcBase, csi_adc_seq_t *ptSeqx, uint8_t byChNum)*

12.3.2.1 功能描述

配置ADC转换序列

12.3.2.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

ptSeqx：ADC转换序列结构体指针，结构体定义详见csi_adc_seq_t。

byChNum：转换序列总得通道数。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	

ptSeqx	typedef struct {
	uint8_t byInChnl; //adc input channel
	uint8_t byRepCnt; //continuous repeat sample count
	uint8_t byAvgCof; //average coefficient
	uint8_t byTrgSrc; //trigger source
	} csi_adc_seq_t;
		转换序列参数：
		byInChnl：输入通道
		byRepCnt：连续重复采样次数
		byAvgCof：平均系数
		byTrgSrc：同步输入触发源
		在adc.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.3 csi_adc_set_buffer

*csi_error_t csi_adc_set_buffer(uint16_t *phwData, uint16_t hwRdLen)*

12.3.3.1 功能描述

配置ADC采样数据缓存(用户定义数据缓存(buffer和采样深度)，通过此API函数传入)

12.3.3.2 参数/返回值说明

1. 参数

   phwData：ADC转换序列采样值缓存指针，指向采样buffer首地址；采样数据存放在该指针指向的buffer中。

   hwRdLen：ADC转换序列通道采样值深度，即每通道采样数值次数。

2. 返回值

   CSI_OK：配置成功。

   CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
phwData	uint16_t 类型指针，指向用户定义的采样数值buffer	指针，指向采样数据缓存首地址，数据缓存由用户定义，缓存数组可以是一维、或者二维，取决于采样深度(通道采样次数)。
hwRdLen	uint16_t 类型数据，通道采样深度	每通道采样次数 1：数据缓存为一维数组 大于1：数据缓存为二维数组
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.4 csi_adc_start

*csi_error_t csi_adc_start(csp_adc_t *ptAdcBase)*

12.3.4.1 功能描述

启动AD转换

12.3.4.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

   CSI_OK：启动成功。

   CSI_ERROR：启动失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.5 csi_adc_stop

*csi_error_t csi_adc_stop(csp_adc_t *ptAdcBase)*

12.3.5.1 功能描述

停止AD转换(连续转换模式有效，单次转换模式不可关闭，转换序列结束时停止)

12.3.5.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

   CSI_OK：设置成功。

   CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.6 csi_adc_conv_mode

*void csi_adc_conv_mode(csp_adc_t *ptAdcBase, csi_adc_conv_mode_e eConvMode)*

12.3.6.1 功能描述

配置ADC转换模式

12.3.6.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

   eConvMode：转换模式，枚举定义详见csi_adc_conv_mode_e。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eConvMode	typedef enum { ADC_CONV_ONESHOT= 0, ADC_CONV_CONTINU= 1 }csi_adc_conv_mode_e;	两种模式：单次转换、连续转换 在adc.h中定义。

12.3.7 csi_adc_conv_pri

*void csi_adc_conv_pri(csp_adc_t *ptAdcBase, uint8_t byPri)*

12.3.7.1 功能描述

配置ADC转换序列优先级

12.3.7.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

byPri：优先级数值，小于该值的转换序列只有被触发时才会转换，大于等于该值的序列正常转换。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
byPri	uint8_t 类型数值，优先级控制数值	用户AD采样序列配置为如下：SEQ0~SEQ5，byPri = 2，那么AD启动时采样从SEQ2启动，即SEQ2~SEQ5正常转换；SEQ0~SEQ1只有在被触发的时候才会转换，否则不会转换。
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.8 csi_adc_read_channel

uint8_t csi_pin_get_num(pin_name_e ePinName)

12.3.8.1 功能描述

获取ADC转换序列指定通道采样数值

12.3.8.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

   byChIdx：ADC转换序列中的通道ID号，即具体哪个通道，用户选择。

2. 返回值

   ADC转换序列中某一通道采样值(用户指定获取)。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t类型指针，ADC0；请参阅12.3.1.2参数说明	
byChIdx	uint8_t 类型数值，采样通道号：0~15	转换序列中任一通道号
return value	uint16_t 类型数值，数值范围：0~7FF	ADC某通道采样数值

12.3.9 csi_adc_read_seqx

*csi_error_t csi_adc_read_seqx(csp_adc_t *ptAdcBase)*

12.3.9.1 功能描述

获取ADC采样序列所有通道数据

12.3.9.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

   CSI_OK：获取成功，采样数值存放于用户定义缓存中，请参阅8.3.3 csi_adc_set_buffer部分。

   CSI_ERROR：获取失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.10 csi_adc_set_vref

*void csi_adc_set_vref(csp_adc_t *ptAdcBase, csi_adc_vref_e eVrefSrc)*

12.3.10.1 功能描述

配置ADC参考电压

12.3.10.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eVrefSrc：参考电压源，枚举定义详见csi_adc_vref_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eVrefSrc	typedef enum{ ADCVERF_VDD_VSS = (0x00ul), ADCVREF_VREFP_VSS = (0x01ul), ADCVREF_FVR2048_VSS = (0x02ul), ADCVREF_FVR4096_VSS = (0x03ul), ADCVREF_INTVREF_VSS = (0x04ul), ADCVREF_VDD_VREFN = (0x08ul), ADCVREF_VREFP_VREFN = (0x09ul), ADCVREF_FVR2048_VREFN = (0x0aul), ADCVREF_FVR4096_VREFN = (0x0bul), ADCVREF_INTVREF_VREFN = (0x0cul) }csi_adc_vref_e;	外部参考源有10中选择，默认选择VDD_VSS。 在adc.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.11 csi_adc_freq_div

*uint32_t csi_adc_freq_div(csp_adc_t *ptAdcBase, uint8_t byDiv)*

12.3.11.1 功能描述

配置ADC采样分频系数

12.3.11.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

   byDiv：分频系数。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
byDiv	uint8_t 类型数值，范围：0~62	byDiv = 0和byDiv=1一样，分频系数为1

12.3.12 csi_adc_get_freq

*uint32_t csi_adc_get_freq(csp_adc_t *ptAdcBase)*

12.3.12.1 功能描述

获取ADC采样频率

12.3.12.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

   ADC采样频率，即ADC工作频率。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
return value	uint32_t 类型数值，单位：Hz	ADC工作频率

12.3.13 csi_adc_int_enable

*void csi_adc_int_enable(csp_adc_t *ptAdcBase, csi_adc_intsrc_e eIntSrc, bool bEnable)*

12.3.13.1 功能描述

使能ADC中断

12.3.13.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eIntSrc：BT中断源，枚举定义详见csi_adc_intsrc_e。

bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	

eIntSrc	typedef enum{
	ADC_INTSRC_NONE = (0x00uL << 0), //no interrupt
	ADC_INTSRC_EOC = (0x01uL << 0),
	ADC_INTSRC_READY= (0x01uL << 1),
	ADC_INTSRC_OVR = (0x01uL << 2),
	ADC_INTSRC_CMP0H= (0x01uL << 4),
	ADC_INTSRC_CMP0L= (0x01uL << 5),
	ADC_INTSRC_CMP1H= (0x01uL << 6),
	ADC_INTSRC_CMP1L= (0x01uL << 7),
	//SEQX0-15
	ADC_INTSRC_SEQ0 = (0x01uL<<16),
	ADC_INTSRC_SEQ1 = (0x01uL << 17),
	ADC_INTSRC_SEQ2 = (0x01uL << 18),
		有23个中断源，包含16个ADC序列转换中断(序列号0~15)和其余7个中断，ADC转换采样中断模式，建议采样序列转换中断。
		在adc.h中定义

	ADC_INTSRC_SEQ3 = (0x01uL << 19), ADC_INTSRC_SEQ4 = (0x01uL << 20), ADC_INTSRC_SEQ5 = (0x01uL << 21), ADC_INTSRC_SEQ6 = (0x01uL << 22), ADC_INTSRC_SEQ7 = (0x01uL << 23), ADC_INTSRC_SEQ8 = (0x01uL << 24), ADC_INTSRC_SEQ9 = (0x01uL << 25), ADC_INTSRC_SEQ10= (0x01uL << 26), ADC_INTSRC_SEQ11= (0x01uL << 27), ADC_INTSRC_SEQ12= (0x01uL << 28), ADC_INTSRC_SEQ13= (0x01uL << 29), ADC_INTSRC_SEQ14= (0x01uL << 30), ADC_INTSRC_SEQ15= (0x01uL << 31) }csi_adc_intsrc_e;	
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

12.3.14 csi_adc_set_cmp0

*csi_error_t csi_adc_set_cmp0(csp_adc_t *ptAdcBase, uint8_t byCmpChnl, uint32_t wCmpData, csi_adc_cmp_dir_e eDir)*

12.3.14.1 功能描述

配置ADC采样比较寄存器0比较功能

12.3.14.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

byCmpChnl：要比较的通道号，转换序列中的某一通道，用户选择。

wCmpData：要比较的参考值，即指定的采样通道AD数值要和这个值进行比较。

eDir：比较方向选择，大于/小于比较参考值，枚举详见csi_adc_cmp_dir_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
byCmpChnl	uint8_t 类型数值，被比较的通道，用户选择。	转换序列中通道号，0~15
wCmpData	uint32_t 类型数值，比较的参考值	ADC采样范围中任意数值
eDir	typedef enum { ADC_CMP_H = 0, ADC_CMP_L, }csi_adc_cmp_dir_e;	两种选择：大于wCmpData、小于wCmpData 在adc.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

12.3.15 csi_adc_set_cmp1

*csi_error_t csi_adc_set_cmp1(csp_adc_t *ptAdcBase, uint8_t byCmpChnl, uint32_t wCmpData, csi_adc_cmp_dir_e eDir)*

12.3.15.1 功能描述

配置ADC采样比较寄存器1比较功能

12.3.15.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

byCmpChnl：要比较的通道号，转换序列中的某一通道，用户选择。

wCmpData：要比较的参考值，即指定的采样通道AD数值要和这个值进行比较。

eDir：比较方向选择，大于/小于比较参考值(wCmpData)，枚举详见csi_adc_cmp_dir_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
byCmpChnl	uint8_t 类型数值，被比较的通道，用户选择。	转换序列中通道号，0~15
wCmpData	uint32_t 类型数值，比较的参考值	ADC采样范围中任意数值
eDir	typedef enum { ADC_CMP_H = 0, ADC_CMP_L, }csi_adc_cmp_dir_e;	两种选择：大于wCmpData、小于wCmpData 在adc.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.16 csi_adc_get_status

*csi_adc_state_e csi_adc_get_status(csp_adc_t *ptAdcBase)*

12.3.16.1 功能描述

获取ADC转换状态

12.3.16.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

ADC转换状态，枚举详见csi_adc_state_e。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	

return value	typedef enum{
	ADC_STATE_IDLE = 0, //idle
	ADC_STATE_DOING, //working
	ADC_STATE_DONE //complete
	}csi_adc_state_e;
		三种状态：空闲、工作中、转换完成
		在adc.h中定义

12.3.17 csi_adc_clr_status

*void csi_adc_clr_status(csp_adc_t *ptAdcBase)*

12.3.17.1 功能描述

清除ADC工作状态(设置ADC为空闲状态)

12.3.17.2 参数/功能描述

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	

12.3.18 csi_adc_set_sync

*csi_error_t csi_adc_set_sync(csp_adc_t *ptAdcBase, csi_adc_trgin_e eTrgIn, csi_adc_trgmode_e eTrgMode, uint8_t byDelay)*

12.3.18.1 功能描述

配置ADC外部同步触发输入

12.3.18.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eTrgin：同步触发输入，枚举定义详见csi_adc_trgin_e。

eTrgMode：同步触发输入模式，枚举定义详见csi_adc_trgmode_e。

byDelay：触发延时，即触发时延时一段时间才开始ADC转换。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eTrgin	typedef enum{ ADC_TRG_SYNCEN0 = 0, ADC_TRG _SYNCEN1, ADC_TRG _SYNCEN2, ADC_TRG _SYNCEN3, ADC_TRG _SYNCEN4, ADC_TRG _SYNCEN5 }csi_adc_trgin_e;	ADC同步触发输入有6个端口：SYNCIN0~SYNCIN5 在adc.h中定义

eTrgMode	typedef enum{
	ADC_TRG_CONTINU= 0, //continuous trG mode
	ADC_TRG_ONCE //once trgmode
		两种触发模式：连续触发、一次性触发

	}csi_adc_trgmode_e;	在adc.h中定义
byDelay	uint8_t 类型数值，范围：0~0xff	触发延时启动，即触发时，延时一段时间才开始ADC转换
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.19 csi_adc_rearm_sync

*void csi_adc_rearm_sync(csp_adc_t *ptAdcBase, csi_adc_trgin_e eTrgIn)*

12.3.19.1 功能描述

使能硬件自动REARM

12.3.19.2 参数/返回值说明

1. 参数

   ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

   eTrgin：同步触发输入，枚举定义详见csi_adc_trgin_e

2. 返回值、

   无返回值。

3. 参数说明

参数	说明	枚举定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eTrgin	typedef enum{ ADC_SYNCEN0 = 0, ADC_SYNCEN1, ADC_SYNCEN2, ADC_SYNCEN3, ADC_SYNCEN4, ADC_SYNCEN5 }csi_adc_trgin_e;	ADC同步触发输入有6个端口： SYNCIN0~SYNCIN5在adc.h中定义

12.3.20 csi_adc_set_evtrg

*csi_error_t csi_adc_set_evtrg(csp_adc_t *ptAdcBase, csi_adc_trgout_e eTrgOut, csi_adc_trgsrc_e eTrgSrc)*

12.3.20.1 功能描述

配置ADC事件触发输出

12.3.20.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eTrgOut：输出端口选择，共有两个端口(数值：0~1)

eTrgSrc：ADC触发源，枚举定义详见csi_adc_trgsrc_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eTrgOut	uint8_t 类型数值，有效数值：0~1 typedef enum{ ADC_TRGOUT0 = 0, ADC_TRGOUT1 }csi_adc_trgout_e;	两个触发输出端口(0/1)
eTrgSrc	typedef enum{ ADC_TRGSRC_NONE = 0, ADC_TRGSRC_EOC, ADC_TRGSRC_READY, ADC_TRGSRC_OVR, ADC_TRGSRC_CMP0H, ADC_TRGSRC_CMP0L, ADC_TRGSRC_CMP1H,	ADC事件触发源有23个：包含16个序列通道采样结束事件(序列通道号0~15)和其余7个事件 在adc.h中定义。

	ADC_TRGSRC_CMP1L, ADC_TRGSRC_SEQEND0, ADC_TRGSRC_SEQEND1, ADC_TRGSRC_SEQEND2, ADC_TRGSRC_SEQEND3, ADC_TRGSRC_SEQEND4, ADC_TRGSRC_SEQEND5, ADC_TRGSRC_SEQEND6, ADC_TRGSRC_SEQEND7, ADC_TRGSRC_SEQEND8, ADC_TRGSRC_SEQEND9, ADC_TRGSRC_SEQEND10, ADC_TRGSRC_SEQEND11, ADC_TRGSRC_SEQEND12, ADC_TRGSRC_SEQEND13, ADC_TRGSRC_SEQEND14, ADC_TRGSRC_SEQEND15 }csi_adc_trgsrc_e;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

12.3.21 csi_adc_fvrout_enable

*void csi_adc_fvrout_enable(csp_adc_t *ptAdcBase, csi_adc_fvrsel_e eLvl, bool bEnable)*

12.3.21.1 功能描述

使能ADC FVROUT

12.3.21.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eLvl：FVR(固定参考电压源)电平选择，枚举详见csi_adc_fvrsel_e。

bEnable：使能/禁止，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	
eLvl	typedef enum{ ADC_FVR2048 = 0, ADC_FVR4096 }csi_adc_fvrsel_e;	电压值有两种选择：2.048V、4.096V 在adc.h中定义
bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义

12.3.22 csi_adc_bufout_enable

*void csi_adc_bufout_enable(csp_adc_t *ptAdcBase , csi_adc_bufsel_e eBufSel, bool bEnable)*

12.3.22.1 功能描述

使能ADC BUFFER输出,有两种选择，一个是内部1v电压，另一个是温度传感器的电压

12.3.22.2 参数/返回值说明

1. 参数

ptAdcBase：ADC寄存器结构体指针，指向ADC基地址，结构体定义详见csp_adc_t。

eBufSel：内部输出电压选择，详见枚举类型csi_adc_bufsel_e

bEnable：使能/禁止输出，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptAdcBase	csp_adc_t 类型指针，ADC0；请参阅12.3.1.2参数说明	

eBufSel	typedef enum{
	ADCIN_INTERIOR_1V0 = 2, //interior 1V0
	ADCIN_INTERIOR_TEMP //interior temp
	}csi_adc_bufsel_e;
		ADCIN_INTERIOR_1V0：内部1v电压。
		ADCIN_INTERIOR_TEMP：内部温度传感器电压

bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能输出 DISABLE：禁止输出 在common.h中定义

13 PINCTRL

13.1 概述

ATP CSI接口PINCTRL的设计中，提供IO丰富的配置及其操作。配置方面包括IO复用功能选择，上拉/下拉，输入/输出模式，速度以及驱动力配置。操作方面提供单个IO翻转，输出高低电平等。

13.2 API列表

Table 2-1 时钟CSI接口函数

API	说明	函数位置
csi_pin_set_mux	设置pin复用功能	pinctrl.c
csi_pin_set_iomap	设置pin的IO重定义	
csi_pin_get_mux	获取pin复用功能	
csi_pin_pull_mode	设置pin上拉/下拉模式	
csi_pin_input_filter	使能/禁止pin输入滤波功能	
csi_pin_output_mode	设置pin输出模式	
csi_pin_speed	设置pin速度	
csi_pin_drive	设置pin驱动能力	
csi_pin_get_num	通过pin name获取pin number	
csi_pin_read	通过pin name读取pin输入电平状态	
csi_pin_irq_mode	设置pin中断模式	
csi_pin_irq_enable	使能pin外部中断	
csi_pin_toggle	翻转pin输出电平状态	
csi_pin_set_high	设置pin输出电平状态为高	
csi_pin_set_low	设置pin输出电平状态为低	
csi_exi_set_evtrg	配置EXI(外部中断)事件触发	
csi_exi_soft_evtrg	EXI(外部中断)事件软件触发	
csi_exi_flt_enable	EXI通道的数字滤波器使能	

13.3 API详细说明

13.3.1 csi_pin_set_mux

void csi_pin_set_mux(pin_name_e ePinName, pin_func_e ePinFunc)

13.3.1.1 功能描述

设置pin复用功能。

13.3.1.2 参数/返回值说明

1. 参数

   ePinName：每一款MCU都定义了对应的pin name，枚举定义详见pin_name_e。

   ePinFunc：每一款MCU对应的pin脚都定义了自己的复用编号，枚举定义详见pin_func_e。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
ePinName	typedef enum { PA00 = 0U, PA01 = 1U, PA02 = 2U, PA03 = 3U, PA04 = 4U, PA05 = 5U, PA06 = 6U, PA07 = 7U, PA08 = 8U, PA09 = 9U, PA010 = 10U, PA011 = 11U, PA012 = 12U, PA013 = 13U, PA014 = 14U, PA015 = 15U, PB00 = 16U, PB01 = 17U, PB02 = 18U, PB03 = 19U, PB04 = 20U, PB05 = 21U, } pin_name_e;	gpio全部的引脚都定义了自己的pin name，pin name都包含在枚举类型pin_name_e中； 在soc.h中定义

ePinFunc	PA015_GPD = 0U, PA015_INPUT = 1U, PA015_OUTPUT = 2U, PA015_OUTPUT_MONI = 3U, PA015_EPT_CHAX = 4U, PA015_BTO_OUT = 5U, PA015_I2C_SCL = 6U, PA015_UART1_RX = 7U, PA015_SPI_MISO = 8U, IOMAP = 10U } pin_func_e;	pin脚所有的功能都包含在枚举类型pin_func_e中；这里只列举出PA015所有功能定义，其它的pin脚功能定义类似 在soc.h中定义

13.3.2 csi_pin_set_iomap

13.3.2.1 功能描述

配置pin的IOMAP功能

csi_error_t csi_pin_set_iomap(pin_name_e ePinName, csi_gpio_iomap_e eIoMap)

13.3.2.2 参数/返回值说明

1. 参数

   ePinName：每一款MCU都定义了对应的pin name，枚举定义详见pin_name_e。

   eIoMap：IOMAP 共有两组 group0/1；没有 group 支持 8 个 pin 脚，每个 pin 脚支持 8 个预设功能，枚举定义详见 csi_gpio_iomap_e。

2. 返回值

   CSI_OK：设置成功。

   CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

eIoMap	typedef enum {
	IOMAP_I2C_SCL =0U, //IOMAP GROUP0
	IOMAP_I2C_SDA,
	IOMAP_GPT_CHA,
	IOMAP_GPT_CHB,
	IOMAP_SPI_MOSI,
	IOMAP_SPI_MISO,
	IOMAP_SPI_SCK,
	IOMAP_SPI_NSS,
	IOMAP_UART0_RX, //IOMAP GROUP1
	IOMAP_UART0_TX,
	IOMAP_EPT_CHAX,
	IOMAP_EPT_CHBX,
	IOMAP_EPT_CHCX,
	IOMAP_EPT_CHAY,
	IOMAP_EPT_CHBY,
	IOMAP_EPT_CHCY
	} csi_gpio_iomap_e;
		系列mcu支持两组IOMAP，每组对应8个预设功能，每组支持8个不同的pin脚(IO口)。
		比如：PA00支持group0，PA00可以配置为IOMAP GROUP0中的8个预设功能中任一个。IOMAP功能详情，在用户手册syscon章节的IO重定义有详细介绍。
		枚举在gpio.h 中定义。

return value	pin_func_e 中枚举值，请参阅13.3.1.2参数说明	

13.3.3 csi_pin_get_mux

pin_func_e csi_pin_get_mux(pin_name_e ePinName)

13.3.3.1 功能描述

获取pin复用功能。

13.3.3.2 参数/返回值说明

1. 参数

   ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

   返回pin脚复用功能编号，类型pin_func_e，具体定义详见pin_func_e。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义
return value	pin_func_e 中枚举值，请参阅13.3.1.2参数说明	

13.3.4 csi_pin_pull_mode

csi_error_t csi_pin_pull_mode(pin_name_e ePinName, csi_gpio_pull_mode_e ePullMode)

13.3.4.1 功能描述

设置pin上拉/下拉模式

13.3.4.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

ePullMode：上/下拉模式，枚举定义详见csi_gpio_pull_mode_e。

2. 返回值

CSI_OK：设置成功。

CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

ePullMode	typedef enum {
	GPIO_PULLNONE = 0, //Pull none
	GPIO_PULLUP, //Pull up
	GPIO_PULLDOWN, //Pull down
	} csi_gpio_pull_mode_e;
		三种模式：上拉、下拉和无上下拉；默认无上下拉。
		在gpio.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.5 csi_pin_input_filter

void csi_pin_input_filter(pin_name_e ePinName, bool bEnable)

13.3.5.1 功能描述

使能/禁止PIN输入滤波功能

13.3.5.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

bEnable：使能/禁止，ENABLE/DISABLE。

2. 返回值：无。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义

13.3.6 csi_pin_output_mode

csi_error_t csi_pin_output_mode(pin_name_e ePinName, csi_gpio_output_mode_e eOutMode)

13.3.6.1 功能描述

设置pin输出模式

13.3.6.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

eOutMode：输出模式，枚举定义详见csi_gpio_output_mode_e。

2. 返回值

CSI_OK：设置成功。

CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

eOutMode	typedef enum {
	GPIO_PUSH_PULL = 0, //push-pull
	GPIO_OPEN_DRAIN, //open drain
	} csi_gpio_output_mode_e;
		两种模式：推挽输出、开漏输出；默认为推挽输出
		在gpio.h中定义。

csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.7 csi_pin_speed

void csi_pin_speed(pin_name_e ePinName, csi_gpio_speed_e eSpeed)

13.3.7.1 功能描述

设置pin作为输出时的速度模式

13.3.7.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

eSpeed：速度模式，枚举定义详见csi_gpio_speed_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

eSpeed	typedef enum {
	GPIO_SPEED_LV0 = 0U, //normal
	GPIO_SPEED_LV1, //fast
	} csi_gpio_speed_e;
		两种模式：慢速(普通)、快速；默认为慢速
		在gpio.h中定义。

13.3.8 csi_pin_drive

void csi_pin_drive(pin_name_e ePinName, csi_gpio_drive_e eDrive)

13.3.8.1 功能描述

设置pin作为输出时的驱动能力

13.3.8.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

eDrive：驱动能力，枚举定义详见csi_gpio_drive_e。

2. 返回值：无

3. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

eDrive	typedef enum {
	GPIO_DRIVE_LV0 = 0U, //normal
	GPIO_DRIVE_LV1, //strong
	} csi_gpio_drive_e;
		两种模式：弱驱(普通)、强驱模式；默认为弱驱模式
		在gpio.h中定义。

13.3.9 csi_pin_get_num

uint8_t csi_pin_get_num(pin_name_e ePinName)

13.3.9.1 功能描述

通过pin name获取pin number

13.3.9.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

pin number(如：(PA00，返回0)；(PA010，返回10)；(PB01，返回1))。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明表	在soc.h中定义
return value	uint8_t 类型数值，数值范围：0~15	pin number：0~15

13.3.10 csi_pin_read

uint32_t csi_pin_read(pin_name_e ePinName)

13.3.10.1 功能描述

读取pin输入电平状态

13.3.10.2 参数/返回值说明

1. 参数

   ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

   电平状态：高低(1/0)；返回值格式：(0 或 1 << (pin number))。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举定义位置
ePinName	pin_name_e中枚举值，请参阅13.3.1.2参数说明表	在soc.h中定义
return value	uint32_t 类型数值，数值范围：0~(1 << 15)	电平状态：高低(1/0)

13.3.11 csi_pin_irq_mode

csi_error_t csi_pin_irq_mode(pin_name_e ePinName, csi_exi_grp_e eExiGrp, csi_gpio_irq_mode_e eTrgEdge)

13.3.11.1 功能描述

配置pin中断模式

13.3.11.2 参数/返回值说明

1. 参数

   ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

   eExiGrp：外部中断组，枚举定义详见csi_exi_grp_e。

   eTrgEdge：中断触发边沿模式，枚举定义详见csi_gpio_irq_mode_e。

2. 返回值

   CSI_OK：配置成功

   CSI_ERROR：配置失败

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

eExiGrp	typedef enum { EXI_GRP0 = 0, EXI_GRP1, EXI_GRP2, EXI_GRP3, EXI_GRP4, EXI_GRP5, EXI_GRP6, EXI_GRP7, EXI_GRP8, EXI_GRP9, EXI_GRP10, EXI_GRP11, EXI_GRP12, EXI_GRP13, EXI_GRP14, EXI_GRP15, EXI_GRP16, EXI_GRP17, EXI_GRP18, EXI_GRP19, }csi_exi_grp_e;	系统有16(0~15)个中断组和4(16~19)个扩展中断组，共20个。配置外部中断时，默认建议用0~15中断组（PA00:GRP0->PA015:GRP15），PB时和PA配置类似。资源不够时（不同端口的相同pin number都需要配置中断，如PA00和PB00都需要中断），再使用扩展中断组 在gpio.h中定义。

eTrgEdge	typedef enum {
	GPIO_IRQ_RISING_EDGE = 0, //rising edge
	GPIO_IRQ_FALLING_EDGE, //falling edge
	GPIO_IRQ_BOTH_EDGE, //both edge
	} csi_gpio_irq_mode_e;
		外部中断触发边沿有三种模式：上升沿、下降沿和双边(上升/下降)沿

csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.12 csi_pin_vic_irq_enable

csi_error_t csi_pin_vic_irq_enable(csi_exi_grp_e eExiGrp, bool bEnable)

13.3.12.1 功能描述

使能 pin 的 VIC 中断

13.3.12.2 参数/返回值说明

1. 参数

   eExiGrp：外部中断组，枚举定义详见csi_exi_grp_e。

   bEnable：使能/禁止中断（ENABLE/DISABLE）

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
eExiGrp	csi_exi_grp_e 中枚举值	在gpio.h中定义。
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.13 csi_pin_irq_enable

void csi_pin_irq_enable(pin_name_e ePinName, bool bEnable)

13.3.13.1 功能描述

使能pin中断

13.3.13.2 参数/返回值说明

1. 参数

   ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

   bEnable：使能/禁止中断（ENABLE/DISABLE）

2. 返回值

   CSI_OK：成功。

   CSI_ERROR：失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明表	在soc.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.14 csi_pin_toggle

void csi_pin_toggle(pin_name_e ePinName)

13.3.14.1 功能描述

翻转pin输出电平状态

13.3.14.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

13.3.15 csi_pin_set_high

void csi_pin_set_high(pin_name_e ePinName)

13.3.15.1 功能描述

设置pin输出电平状态为高

13.3.15.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

13.3.16 csi_pin_set_low

void csi_pin_set_low(pin_name_e ePinName)

13.3.16.1 功能描述

设置设置pin输出电平状态为低

13.3.16.2 参数/返回值说明

1. 参数

ePinName：pin脚名字，即pin name，枚举定义详见pin_name_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	枚举定义位置
ePinName	pin_name_e 中枚举值，请参阅13.3.1.2参数说明	在soc.h中定义

13.3.17 csi_exi_set_evtrg

csi_error_t csi_exi_set_evtrg(csi_exi_trgout_e eTrgOut, csi_exi_trgsrc_e eExiTrgSrc, uint8_t byTrgPrd)

13.3.17.1 功能描述

配置EXI(外部中断)事件触发输出

13.3.17.2 参数/返回值说明

1. 参数

eTrgOut：输出通道选择，枚举定义详见csi_exi_trgout_e。

eExiTrgSrc：EXI触发源，枚举定义详见csi_exi_trgsrc_e。

byTrgPrd：EXI事件触发计数周期

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举定义位置
byTrgOut	typedef enum { EXI_TRGOUT0 = 0, EXI_TRGOUT1, EXI_TRGOUT2, EXI_TRGOUT3, EXI_TRGOUT4, EXI_TRGOUT5, }csi_exi_trgout_e;	共有6个(EXI_TRGOUT0~5)触发输出通道 在gpio.h中定义。
eExiTrgSrc	typedef enum { TRGSRC_EXI0 = 0, TRGSRC_EXI1, TRGSRC_EXI2, TRGSRC_EXI3, TRGSRC_EXI4, TRGSRC_EXI5, TRGSRC_EXI6, TRGSRC_EXI7, TRGSRC_EXI8, TRGSRC_EXI9, TRGSRC_EXI10, TRGSRC_EXI11, TRGSRC_EXI12, TRGSRC_EXI13, TRGSRC_EXI14, TRGSRC_EXI15, TRGSRC_EXI16, TRGSRC_EXI17, TRGSRC_EXI18, TRGSRC_EXI19, }csi_exi_trgsrc_e;	EXI事件触发源共20个(EXI0~EXI19)，和外部中断组相对应。 在gpio.h中定义。
byTrgPrd	uint8_t 类型数值，数值范围：0~15	EXI触发次数大于该设置值时产生触发输出(该值为1是，需要2次触发才会产生一次触发输出)； 该值默认为0
csi_error_t	csi_error_t 中定义值	在common.h中定义

13.3.18 csi_exi_soft_evtrg

void csi_exi_soft_evtrg(csi_exi_trgout_e eTrgOut)

13.3.18.1 功能描述

EXI(外部中断)事件软件触发

13.3.18.2 参数/返回值说明

1. 参数

eTrgOut：输出通道选择，枚举定义详见csi_exi_trgout_e。

2. 返回值：无。

3. 参数说明

参数	说明	概述及其枚举定义位置
byTrgOut	typedef enum { EXI_TRGOUT0 = 0, EXI_TRGOUT1, EXI_TRGOUT2, EXI_TRGOUT3, EXI_TRGOUT4, EXI_TRGOUT5, }csi_exi_trgout_e;	共有6个(EXI_TRGOUT0~5)触发输出通道 在gpio.h中定义。

13.3.19 csi_exi_flt_enable

void csi_exi_flt_enable(csi_exi_flt_ckdiv_e eCkDiv, bool bEnable)

13.3.19.1 功能描述

EXI 通道的数字滤波器使能

13.3.19.2 参数/返回值说明

1. 参数

eCkDiv：数字滤波clk分频，枚举定义详见csi_exi_flt_ckdiv_e。滤波时间和IMO_FSEL选择有关， IMO_FSEL有4种频率值，具体的配置API接口请参阅时钟章节csi_error_t csi_imosc_enable(uint8_t byFre)接口函数。

bEnable：使能/禁止滤波（ENABLE/DISABLE）

2. 返回值：无

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
eCkDiv	数字滤波clk分频	在gpio.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

14 GPIO PORT

14.1 概述

ATP32F110x有两个GPIO端口，GPIOA0(Port A0) 、GPIOA1(Port A1) 、GPIOB0(Port B0)和GPIOC0(Port C0)；CSI接口GPIO的PORT设计中，提供IO输入和输出的相关配置。配置方面包括I/O方向(输入/输出)，输入/输出模式，上拉/下拉和中断等。操作方面提供GIO翻转，输出高低电平等。API接口函数对Port A0/A1/B0/C0进行操作（可以是整个Port或者Port中的某几个引脚）。

14.2 API列表

Table 14-1 GPIO PORT CSI接口函数

API	说明	函数位置
csi_gpio_port_dir	设置gpio port 方向(输入/输出)	gpio.c
csi_gpio_port_pull_mode	设置gpio port 上拉/下拉	
csi_gpio_port_input_filter	使能/禁止gpio port 输入滤波功能	
csi_gpio_port_output_mode	设置gpio port 输出模式	
csi_gpio_port_write	设置gpio port 引脚输出电平状态	
csi_gpio_port_read	读取gpio port 引脚输入电平状态	
csi_gpio_port_irq_mode	设置gpio port 中断模式	
csi_gpio_port_irq_enable	使能gpio port 引脚中断	
csi_gpio_port_toggle	翻转gpio port 引脚输出电平状态	
csi_gpio_port_set_high	设置gpio port 引脚输出电平状态为高	
csi_gpio_port_set_low	设置gpio port 引脚输出电平状态为低	

14.3 API详细说明

14.3.1 csi_gpio_port_dir

*csi_error_t csi_gpio_port_dir(csp_gpio_t *ptGpioBase, uint32_t wPinMask, csi_gpio_dir_e eDir)*

14.3.1.1 功能描述

配置gpio port方向(输入/输出模式)

14.3.1.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/A1/B0/C0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

eDir：方向，枚举定义详见csi_gpio_dir_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针：GPIOA0/ GPIOA1/GPIOB0/ GPIOC0，指向对应外设基地址	四个Port，定义两个对应的结构体指针GPIOA0/ GPIOA1/GPIOB0/ GPIOC0，分别指向Port A0/A1/B0/C0基地址。 GPIOA0/ GPIOA1/GPIOB0/ GPIOC0在devices.c中定义 csp_gpio_t在csp_gpio.h中定义
wPinMask	uint32_t（unsigned int）类型，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

eDir	typedef enum { GPIO_DIR_GPD = 0, GPIO_DIR_INPUT, //GPIO as input GPIO_DIR_OUTPUT //GPIO as output } csi_gpio_dir_e;	三种模式：高阻、输入和输出； 默认为高阻态 在gpio.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

14.3.2 csi_gpio_port_pull_mode

*csi_error_t csi_gpio_port_pull_mode(csp_gpio_t *ptGpioBase, uint32_t wPinMask, csi_gpio_pull_mode_e eMode)*

14.3.2.1 功能描述

配置gpio port上拉/下拉模式

14.3.2.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

ePullMode：上/下拉模式，枚举定义详见csi_gpio_pull_mode_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t（unsigned int）类型，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

ePullMode	typedef enum {
	GPIO_PULLNONE = 0, //Pull none
	GPIO_PULLUP, //Pull up
	GPIO_PULLDOWN, //Pull down
	} csi_gpio_pull_mode_e;
		三种模式：上拉、下拉和禁止上下拉；默认禁止上下拉。
		在gpio.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

14.3.3 csi_gpio_port_input_filter

*void csi_gpio_port_input_filter(csp_gpio_t *ptGpioBase, uint32_t wPinMask, bool bEnable)*

14.3.3.1 功能描述

使能/禁止gpio port 输入滤波功能

14.3.3.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

bEnable：使能/禁止，ENABLE/DISABLE。

2. 返回值：无。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3
bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

14.3.4 csi_gpio_port_output_mode

*csi_error_t csi_gpio_port_output_mode(csp_gpio_t *ptGpioBase, uint32_t wPinMask, csi_gpio_output_mode_e eOutMode)*

14.3.4.1 功能描述

设置gpio port输出模式。

14.3.4.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

eOutMode：输出模式，枚举定义详见csi_gpio_output_mode_e。

2. 返回值

CSI_OK：设置成功。

CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

eOutMode	typedef enum {
	GPIO_PUSH_PULL = 0, //push-pull
	GPIO_OPEN_DRAIN, //open drain
	} csi_gpio_output_mode_e;
		两种模式：推挽输出、开漏输出；默认为推挽输出。
		在gpio.h中定义。

csi_error_t	csi_error_t 中定义值	在common.h中定义

14.3.5 csi_gpio_port_write

*void csi_gpio_port_write(csp_gpio_t *ptGpioBase, uint32_t wPinMask, csi_gpio_pin_state_e ePinVal)*

14.3.5.1 功能描述

设置gpio port引脚输出电平状态

14.3.5.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

ePinVal：电平状态，枚举定义详见csi_gpio_pin_state_e

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

ePinVal	typedef enum {
	GPIO_PIN_LOW = 0, //GPIO low level
	GPIO_PIN_HIGH, //GPIO high level
	} csi_gpio_pin_state_e;
		两种状态：高电平、低电平
		在gpio.h中定义。

14.3.6 csi_gpio_port_read

*uint32_t csi_gpio_port_read(csp_gpio_t *ptGpioBase, uint32_t wPinMask)*

14.3.6.1 功能描述

读取gpio port指定引脚输入电平状态

14.3.6.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

2. 返回值

根据指定bit位掩码，得到对应的引脚状态。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3

		如：0x0009：设置pin0和pin3
return value	uint32_t 类型数值，范围：0~0xffff	引脚电平状态：高、低(1/0)

14.3.7 csi_gpio_port_irq_mode

*csi_error_t csi_gpio_port_irq_mode(csp_gpio_t *ptGpioBase, uint32_t wPinMask, csi_gpio_irq_mode_e eTrgEdge)*

14.3.7.1 功能描述

配置gpio port中断模式。

14.3.7.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

eTrgEdge：中断触发边沿模式，枚举定义详见csi_gpio_irq_mode_e。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

eTrgEdge	typedef enum {
	GPIO_IRQ_RISING_EDGE = 0, //rising edge
	GPIO_IRQ_FALLING_EDGE, //falling edge
	GPIO_IRQ_BOTH_EDGE, //both edge
	} csi_gpio_irq_mode_e;
		中断触发边沿有三种模式：上升沿、下降沿、双边(上升/下降)沿

return value	csi_error_t 中定义值	在common.h中定义

14.3.8 csi_gpio_port_irq_enable

*void csi_gpio_port_irq_enable(csp_gpio_t *ptGpioBase, uint32_t wPinMask, bool bEnable)*

14.3.8.1 功能描述

使能gpio port引脚中断

14.3.8.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位(pin)。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3
bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

14.3.9 csi_gpio_port_toggle

*void csi_gpio_port_toggle(csp_gpio_t *ptGpioBase, uint32_t wPinMask)*

14.3.9.1 功能描述

翻转gpio port引脚电平状态

14.3.9.2 参数/返回值说明

1. 参数

   ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

   wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

14.3.10 csi_gpio_port_set_high

*void csi_gpio_port_set_high(csp_gpio_t *ptGpioBase, uint32_t wPinMask)*

14.3.10.1 功能描述

设置gpio port引脚输出电平状态为高

14.3.10.2 参数/返回值说明

1. 参数

   ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

   wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

14.3.11 csi_gpio_port_set_low

*void csi_gpio_port_set_low(csp_gpio_t *ptGpioBase, uint32_t wPinMask)*

14.3.11.1 功能描述

设置gpio port引脚输出电平状态为低

14.3.11.2 参数/返回值说明

1. 参数

ptGpioBase：gpio port寄存器结构体指针，指向具体端口(Port A0/B0)基地址，结构体定义详见csp_gpio_t。

wPinMask：bit位掩码，指定需要设置的bit位，如：0x00ff代表，设置pin0~pin7。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptGpioBase	csp_gpio_t 类型指针，请参阅14.3.1.2参数说明	在devices.c中定义
wPinMask	uint32_t 类型数值，范围：1 ~0xffff	指定需要设置的bit位。 如：0x000f：设置pin0~pin3 如：0x0009：设置pin0和pin3

15 BT

15.1 概述

基本型定时器(Basic Timer)，是一个16位递增计数器。支持自动重载功能。可提供定时、计数和简单PWM波形输出。APT CSI接口提供了BT的定时和PWM输出相关配置和操作。

15.2 API列表

Table 15-1 时钟CSI接口函数

API	说明	函数位置
csi_bt_timer_init	定时功能初始化	pin.c
csi_bt_start	启动BT	
csi_bt_stop	停止BT	
csi_bt_pwm_init	PWM输出功能初始化	
csi_bt_get_remaining_value	获取BT剩余计数值(距离定时中断)	
csi_bt_get_load_value	获取BT的Load寄存器值	
csi_bt_is_running	检测BT是否正在工作	
csi_bt_count_mode	设置BT计数器工作模式	
csi_bt_int_enable	使能BT中断	
csi_bt_pwm_duty_cycle_updata	更新BT的PWM输出占空比	
csi_bt_pwm_updata	更新BT的PWM输出周期和占空比	
csi_bt_prdr_cmp_updata	更新BT的PRDR和CMP寄存器值	
csi_bt_set_sync	配置BT外部同步触发输入	
csi_bt_rearm_sync	设置同步触发模式自动REARM	
csi_bt_set_evtrg	配置BT事件触发输出	
csi_bt_soft_evtrg	BT软件产生一次事件触发输出	

15.3 API详细说明

15.3.1 csi_bt_timer_init

*csi_error_t csi_bt_timer_init(csp_bt_t *ptBtBase, uint32_t wTimeOut)*

15.3.1.1 功能描述

定时功能初始化，默认使用计数器周期结束中断(PEND)

15.3.1.2 参数/返回值说明

1. 参数

   ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

   wTimeOut：计数器溢出时间，即定时时间，单位ms。

2. 返回值

   CSI_OK：初始化成功。

   CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1，指向对应BT的基地址	两个BT，定义两个对应的结构体指针BT0、BT1 分别指向BT0~1对应的基地址。 BT0/BT1在devices.c中定义 csp_bt_t在csp_bt.h中定义
wTimeOut	uint32_t 类型数值，单位：us	定时时间，单位us
csi_error_t	csi_error_t 中定义值	在common.h中定义

15.3.2 csi_bt_start

*void csi_bt_start(csp_bt_t *ptBtBase)*

15.3.2.1 功能描述

启动BT

15.3.2.2 参数/返回值说明

1. 参数

   ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	枚举定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

15.3.3 csi_bt_stop

*void csi_bt_stop(csp_bt_t *ptBtBase)*

15.3.3.1 功能描述

停止BT

15.3.3.2 参数/返回值说明

1. 参数

   ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	枚举定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2中参数说明	在devices.c中定义

15.3.4 csi_bt_pwm_init

*csi_error_t csi_bt_pwm_init(csp_bt_t *ptBtBase, csi_bt_pwm_config_t *ptBtPwmCfg)*

15.3.4.1 功能描述

PWM输出初始化

15.3.4.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

ptBtPwmCfg：PWM输出配置结构体指针，结构体定义详见csi_bt_pwm_config_t。

2. 返回值

CSI_OK：设置成功。

CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

ptBtPwmCfg	typedef struct {
	uint8_t byIdleLevel; //PWM idel level
	uint8_t byStartLevel; //PWM start Level
	uint8_t byInt; //PWM interrupt source
	uint8_t byDutyCycle; //PWM duty cycle
	uint32_t wFreq; //PWM frequency
	} csi_bt_pwm_config_t;
		初始化配置中包含PWM输出空闲电平、起始电平、中断源、占空比和频率
		在bt.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

15.3.5 csi_bt_get_remaining_value

*uint32_t csi_bt_get_remaining_value(csp_bt_t *ptBtBase)*

15.3.5.1 功能描述

获取BT定时剩余计数值

15.3.5.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

2. 返回值

定时剩余计数值，若要得到剩余时间需要换算(t = 1/Tclk * 剩余计数值)。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
return value	uint32_t 类型数值，count的剩余计数值	count的剩余计数值

15.3.6 csi_bt_get_load_value

*uint32_t csi_bt_get_load_value(csp_bt_t *ptBtBase)*

15.3.6.1 功能描述

获取BT的Load寄存器值(PRDR寄存器值)。

15.3.6.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

2. 返回值

计数器加载值，若要得到加载时间(定时时间)需换算(t = 1/Tclk * 加载值)。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
return value	uint32_t 类型数值，count的加载值	count的加载值

15.3.7 csi_bt_is_running

*bool csi_bt_is_running(csp_bt_t *ptBtBase)*

15.3.7.1 功能描述

检测BT工作状态

15.3.7.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

2. 返回值

ture：正在工作。

false：停止工作。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
return value	Bool 类型数值，ture/false(1/0)	ture：数值为1(真) false：数值为0(假)

15.3.8 csi_bt_count_mode

*void csi_bt_count_mode(csp_bt_t *ptBtBase, csi_bt_cntmode_e eCntMode)*

15.3.8.1 功能描述

设置BT计数工作模式

15.3.8.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

eCntMode：计数工作模式，枚举定义详见csi_bt_cntmode_e。

2. 返回值

无返回值

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

eCntMode	typedef enum{
	BT_CNT_CONTINU = 0, //continuous
	BT_CNT_ONCE //once
	}csi_bt_cntmode_e;
		计数工作模式有两种，连续计数模式、单次触发模式
		默认为连续计数模式

15.3.9 csi_bt_int_enable

*void csi_bt_int_enable(csp_bt_t *ptBtBase, csi_bt_intsrc_e eIntSrc, bool bEnable)*

15.3.9.1 功能描述

使能BT中断

15.3.9.2 参数/返回值说明

1. 参数

ptBtBase：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

eIntSrc：BT中断源，枚举定义详见csi_bt_intsrc_e。

bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

eIntSrc	typedef enum
	{
	BT_INTSRC_NONE = (0x00ul << 0), //NONE interrupt
	BT_INTSRC_PEND = (0x01ul << 0), //PEND interrupt
	BT_INTSRC_CMP = (0x01ul << 1), //CMP interrupt
	BT_INTSRC_OVF = (0x01ul << 2), //OVF interrupt
	BT_INTSRC_EVTRG = (0x01ul << 3) //EVTRG interrupt
	}csi_bt_intsrc_e;
		有4个中断源：周期结束、比较匹配、计数溢出和触发输出事件
		在bt.h中定义

bEnable	Bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

15.3.10 csi_bt_pwm_duty_cycle_updata

*void csi_bt_pwm_duty_cycle_updata(csp_bt_t *ptBtBase, uint8_t byDutyCycle)*

15.3.10.1 功能描述

更新PWM输出占空比

15.3.10.2 参数/返回值说明

1. 参数

   ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

   byDutyCycle：PWM输出占空比(0 < byDutyCycle < 100)。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
byDutyCycle	uint8_t 类型数值，范围：(>0 && < 100)	PWM占空比，(>0 && < 100)

15.3.11 csi_bt_pwm_updata

*void csi_bt_pwm_updata(csp_bt_t *ptBtBase, uint32_t wFreq, uint8_t byDutyCycle)*

15.3.11.1 功能描述

更新PWM输出频率和占空比

15.3.11.2 参数/返回值说明

1. 参数

   ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

   wFreq：PWM输出频率，单位Hz

   byDutyCycle：PWM输出占空比(0 < byDutyCycle < 100)。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
wFreq	uint32_t 类型数值，单位Hz	PWM频率，单位：Hz
byDutyCycle	uint8_t 类型数值，范围：(>0 && < 100)	PWM占空比，(>0 && < 100)

15.3.12 csi_bt_prdr_cmp_updata

*void csi_bt_prdr_cmp_updata(csp_bt_t *ptBtBase, uint16_t hwPrdr, uint16_t hwCmp)*

15.3.12.1 功能描述

更新PRDR和CMP寄存器值

15.3.12.2 参数/返回值说明

1. 参数

ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

hwPrdr：加载到PRDR寄存器值，即周期结束值。

hwCmp：加载到CMP寄存器值，即比较匹配值。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
hwPrdr	uint16_t 类型数值，范围：0~0xffff	PRDR加载值，周期结束寄存器
hwCmp	uint16_t 类型数值，范围：0~0xffff	CMP加载值，比较匹配寄存器

15.3.13 csi_bt_set_sync

*csi_error_t csi_bt_set_sync(csp_bt_t *ptBtBase,csi_bt_trgin_e eTrgin, csi_bt_trgmode_e eTrgMode, bool bAutoRearm)*

15.3.13.1 功能描述

配置BT外部同步触发输入

15.3.13.2 参数/返回值说明

1. 参数

ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

eTrgin：同步触发输入，枚举定义详见csi_bt_trgin_e。

eTrgMode：同步触发输入模式，枚举定义详见csi_bt_trgmode_e。

bAutoRearm：自动REARM禁止，禁止/允许后续触发。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
eTrgin	typedef enum{ BT_TRG_SYNCIN0 = 0, BT_TRG_SYNCIN1, BT_TRG_SYNCIN2 }csi_bt_trgin_e;	BT同步触发输入有两个端口： SYNCIN0触发BT的启动， SYNCIN1触发BT的计数值增加一拍 SYNCIN2触发BT的停止 在bt.h中定义

eTrgMode	typedef enum{
	BT_TRG_CONTINU = 0, //continuous trg mode
	BT_TRG_ONCE //once trg mode
	}csi_bt_trgmode_e;
		两种触发模式：连续触发、一次性触发
		在bt.h中定义

bAutoRearm	bool类型数值，ENABLE/DISABLE	一次性触发模式下有效 ENABLE：清除当初通道状态，并允许新的触发 DISABLE：无效 在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

15.3.14 csi_bt_rearm_sync

*void csi_bt_rearm_sync(csp_bt_t *ptBtBase,csi_bt_trgin_e eTrgin)*

15.3.14.1 功能描述

使能硬件自动REARM；计数周期结束时，自动REARM

15.3.14.2 参数/返回值说明

1. 参数

   ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

   eTrgin：同步触发输入，枚举定义详见csi_bt_trgin_e

2. 返回值

   无返回值。

3. 参数说明

参数	说明	枚举定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

eTrgin	typedef enum{
	BT_TRGIN_SYNCEN0 = 0, //sync evtrr input0
	BT_TRGIN_SYNCEN1 //sync evtrg input1
	}csi_bt_trgin_e;
		BT同步触发输入有两个端口： SYNCIN0触发BT的启动，SYNCIN1触发BT的计数值增加一拍
		在bt.h中定义

15.3.15 csi_bt_set_evtrg

*csi_error_t csi_bt_set_evtrg(csp_bt_t *ptBtBase, csi_bt_trgout_e eTrgOut, csi_bt_trgsrc_e eTrgSrc)*

15.3.15.1 功能描述

配置BT事件触发输出

15.3.15.2 参数/返回值说明

1. 参数

   ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

   eTrgOut：输出通道选择，枚举定义详见csi_bt_trgout_e。

   eTrgSrc：BT触发源，枚举定义详见csi_bt_trgsrc_e。

2. 返回值

   CSI_OK：配置成功。

   CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义
eTrgOut	typedef enum { BT_TRGOUT = 0, }csi_bt_trgout_e;	只有一个触发输出通道 BT_TRGOUT

eExiTrgSrc	typedef enum
	{
	BT_TRGSRC_DIS = 0, //none trigger
	BT_TRGSRC_PEND, //PEND as trigger event
	BT_TRGSRC_CMP, //CMP as trigger event
	BT_TRGSRC_OVF //CMP as trigger event
	csi_bt_trgsrc_e;
		BT事件触发源有三个：
		PEND，周期结束
		CMP，比较匹配、
		OVF，计数器计数溢出
		在bt.h中定义。

csi_error_t	csi_error_t 中定义值	在common.h中定义

15.3.16 csi_bt_soft_evtrg

*void csi_bt_soft_evtrg(csp_bt_t *ptBtBase)*

15.3.16.1 功能描述

软件产生一次事件触发输出

15.3.16.2 参数/返回值说明

4. 参数

ptBtBase ：BT寄存器结构体指针，指向BT基地址，结构体定义详见csp_bt_t。

5. 返回值： 无。

6. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ptBtBase	csp_bt_t 类型指针，BT0/BT1；请参阅15.3.1.2参数说明	在devices.c中定义

16 CNTA

16.1 概述

计数器A（CNT A）是一个16位计数器，可用于生成占空比可配的载波。同时CNTA可以和BT0联动，由BT0的MATCH和PEND事件控制输出包络。

APT CSI接口CNTA的设计中，提供CNTA丰富的配置及其操作。

16.2 API列表

Table 16-1 计数器A CSI接口函数

API	说明	函数位置
csi_cnta_start	启动计数器	cnta.c
csi_cnta_stop	停止计数器	
csi_cnta_get_datal_value	获取DATAL控制计数器A输出的低电平宽度	
csi_cnta_get_datah_value	获取DATAH控制计数器A输出的高电平宽度	
csi_cnta_pwm_para_updata	更新DATAH，DATAL的数据	
csi_cnta_count_mode	计数器的工作模式，单次还是重复	
csi_cnta_bt0_sync	硬件触发控制载波的打开和关闭，是否使能计数值寄存器的硬件自动更新功能	
csi_cnta_timer_init	计数器当做定时器来用时的初始化	
csi_cnta_pwm_init	计数器当做PWM来用时的初始化	

16.3 API详细说明

16.3.1 csi_cnta_start

*void csi_cnta_start(csp_cnta_t *ptCntaBase)*

16.3.1.1 功能描述

启动计数器。

16.3.1.2 参数/返回值说明

1. 参数

   ptCntaBase：指向cnta外设寄存器结构体的指针。

2. 返回值：无返回值。

3. 参数说明表

参数	说明	概述及其枚举/结构体定义位置

ptCntaBase	参数：CNTA
	csp_cnta_t *CNTA = (csp_cnta_t *)(APB_CNTA_BASE);
	typedef struct
	{
	__IOM uint32_t CADATAH;
	__IOM uint32_t CADATAL;
	__IOM uint32_t CACON;
	__IOM uint32_t INTMASK;
	} csp_cnta_t ;
		该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h

16.3.2 csi_cnta_stop

*void csi_cnta_stop(csp_cnta_t *ptCntaBase)*

16.3.2.1 功能描述

停止计数器。

16.3.2.2 参数/返回值说明

1. 参数

   ptCntaBase：指向cnta外设寄存器结构体的指针。

2. 返回值：无返回值。

3. 参数说明表

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h

16.3.3 csi_cnta_get_datal_value

*uint32_t csi_cnta_get_datal_value(csp_cnta_t *ptCntaBase)*

16.3.3.1 功能描述

获取DATAL寄存器的数据

16.3.3.2 参数/返回值说明

1. 参数

ptCntaBase：指向cnta外设寄存器结构体的指针。

2. 返回值

DATAL寄存器的值

3. 参数/返回值说明表

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h
uint32_t 类型变量	CADATAL	该类型为uint32_t

16.3.4 csi_cnta_get_datah_value

*uint32_t csi_cnta_get_datah_value(csp_cnta_t *ptCntaBase)*

16.3.4.1 功能描述

获取DATAH寄存器的数据

16.3.4.2 参数/返回值说明

1. 参数

   ptCntaBase：指向cnta外设寄存器结构体的指针。

2. 返回值

   DATAH寄存器的值。

3. 参数/返回值说明表

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h
uint32_t 类型变量	CADATAH	该类型为uint32_t

16.3.5 csi_cnta_pwm_para_updata

*void csi_cnta_pwm_para_updata (csp_cnta_t *ptCntaBase, uint16_t hwDatah, uint16_t hwDatal, csi_cnta_sw_updata_e eUpdata)*

16.3.5.1 功能描述

更新DATAH，DATAL的数据。即改变高低电平的时间。

16.3.5.2 参数/返回值说明

4. 参数

ptCntaBase：指向cnta外设寄存器结构体的指针。

hwDatah：设置高电平计数值

hwDatal：设置低电平计数值

eUpdata：软件立即更新设置的值，详见csi_cnta_sw_updata_e

5. 返回值：无返回值。

6. 参数/返回值说明表

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h
hwDatah	设置高电平计数值	该类型为uint16_t
hwDatal	设置低电平计数值	该类型为uint16_t
eUpdata	typedef enum { CNTA_SW_EN = (0x01ul) }csi_cnta_sw_updata_t;	软件立即更新设置的值

16.3.6 csi_cnta_count_mode

*csi_cnta_count_mode (csp_cnta_t *ptCntaBase, csi_cnta_cntmode_e eCntMode)*

16.3.6.1 功能描述

设置计数器的工作模式，是单次还是连续

16.3.6.2 参数/返回值说明

7. 参数

ptCntaBase：指向cnta外设寄存器结构体的指针。

eCntMode：计数器工作模式，详见csi_cnta_cntmode_e

8. 返回值：无返回值。

9. 参数/返回值说明表

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h

eUpdata	typedef enum
	{
	CNTA_CNT_ONCE = 0, // once
	CNTA_CNT_CONTINU //continuous
	} csi_cnta_cntmode_e ;
		0：单次
		1：重复

16.3.7 csi_cnta_bt0_sync

*csi_error_t csi_cnta_bt0_sync(csp_cnta_t *ptCntaBase, csi_cnta_tcpend_e tcpend_rem, csi_cnta_tcmatch_e tcmatch_rem,csi_cnta_hw_updata_e hw_updata)*

16.3.7.1 功能描述

结合bt0模块，实现硬件触发控制载波的打开和关闭，以及是否使能计数值寄存器的硬件自动更新功能；（受bt0脉冲匹配中断和bt0周期结束中断控制,原始事件也可以，不一定要开bt0中断）

16.3.7.2 参数/返回值说明

10. 参数

ptCntaBase：指向cnta外设寄存器结构体的指针。

tcpend_rem：枚举类型，详见csi_cnta_tcpend_e。

tcmatch_rem：枚举类型，详见csi_cnta_tcmatch_e

hw_updata:枚举类型，详见csi_cnta_hw_updata_e

11. 返回值

CSI_OK：设置成功 CSI_ERROR：设置失败

12. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在devices.c,指针类型定义在csp_cnta.h
tcpend_rem	typedef enum { CNTA_PEND_CARRIERON_DIS = 0, CNTA_PEND_CARRIERON_CLR, CNTA_PEND_CARRIERON_SET }csi_cnta_tcpend_e;	当TC周期结束中断发生时， 硬件触发控制载波的打开和关闭，00/11是禁止。01是关闭，10是打开。定义在cnta.h
tcmatch_rem	typedef enum {	当TC脉冲匹配中断发生时， 硬件触发控制载波的打开和关

	CNTA_MATCH_CARRIERON_DIS = 0, CNTA_MATCH_CARRIERON_CLR, CNTA_MATCH_CARRIERON_SET }csi_cnta_tcmatch_e;	闭,00/11是禁止。01是关闭，10是打开。定义在cnta.h
hw_updata	typedef enum { CNTA_HW_DIS = (0x00ul), CNTA_HW_TCMATCH = (0x01ul), CNTA_HW_TCPEND = (0x02ul) }csi_cnta_hw_updata_e;	X1: 当TC脉冲匹配中断发生时，计数值会自动载入计数器 1X: 当TC周期结束中断发生时，计数值会自动载入计数器,定义在cnta.h
csi_error_t	csi_error_t中定义值	在common.h中定义

16.3.8 csi_cnta_timer_init

*csi_error_t csi_cnta_timer_init (csp_cnta_t *ptCntaBase,uint32_t wTimeOut)*

16.3.8.1 功能描述

计数器当做定时器来用时的初始化,初始化中主要设置定时周期

16.3.8.2 参数/返回值说明

1. 参数

   ptCntaBase：指向cnta外设寄存器结构体的指针。

   wTimeOut：时器（us）。

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向 csp_cnta_t 结构体的指针，请参阅 16.3.1.2 中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在csp.c,指针类型定义在csp_cnta.h
wTimeOut	定时参数配置	
csi_error_t	csi_error_t中定义值	在common.h中定义

16.3.9 csi_cnta_pwm_init

*csi_error_t csi_cnta_pwm_init (csp_cnta_t *ptCntaBase,csi_cnta_pwm_config_t *ptCntaPwmCfg)*

16.3.9.1 功能描述

计数器当做PWM来用时的初始化

16.3.9.2 参数/返回值说明

13. 参数

ptCntaBase：指向cnta外设寄存器结构体的指针。

ptCntaPwmCfg：pwm的配置参数。详见csi_cnta_pwm_config_t

14. 返回值

CSI_OK：设置成功; CSI_ERROR：设置失败

15. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptCntaBase	参数：CNTA 指向csp_cnta_t结构体的指针，请参阅16.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为CNTA。CNTA的指针定义在csp.c,指针类型定义在csp_cnta.h

ptCntaPwmCfg	typedef struct {
	uint8_t byStartLevel; //计数器 pwm 初始电平
	uint8_t byStopLevel; //计数器 pwm 结束电平
	uint8_t byInt; //int source
	uint8_t byDutyCycle; //占空比(0-100)
	uint32_t wFreq; //频率（hz）
	} csi_cnta_pwm_config_t;
		Pwm配置参数结构体
		定义在cnta.h。占空比请配置在0-100之间

csi_error_t	csi_error_t中定义值	在common.h中定义

17 GPTA

17.1 概述

通用定时器（General Purpose Timer）作为MCU的关键外设，可以提供多种时基计数和波形产生功能。通过灵活的PWM输出，可以适用于各种复杂多变的应用。GPTA内部包含一个16位的定时/计数模块，支持2种工作模式(捕捉模式和波形发生器模式)。

17.2 API列表

Table 16-1 GPTA CSI接口函数

API	说明	函数位置
gpta0_initen_irqhandler	定时器0中断回调函数	
gpta1_initen_irqhandler	定时器1中断回调函数	
csi_gpta_capture_init	定时器捕获模式参数设置	
csi_gpta_wave_init	定时器波形模式参数基础设置（计数模式，周期，占空比等）	
csi_gpta_channel_cmpload_config	定时器CMPA.CMPB载入时机设置	
csi_gpta_channel_config	设置通道PWM1、PWM2的波形	
csi_gpta_channel_aqload_config	通道PWM1、PWM2的波形载入时机设置	
csi_gpta_global_config	全局载入控制配置	
csi_gpta_gldcfg	全局载入对象使能或禁止	
csi_gpta_global_sw	软件产生一次全局载入	
csi_gpta_global_rearm	单次模式下全局载入重使能	
csi_gpta_start	定时器开始计数	
csi_gpta_stop	定时器停止计数	
csi_gpta_set_start_mode	定时器开始位模式选择（普通开始/开始的同时触发同步事件0）	
csi_gpta_set_os_mode	定时器模式选择（单次模式或连续模式）	
csi_gpta_set_stop_st	波形输出被停止时，输出端口的缺省状态	
csi_gpta_get_prdr	读取定时器周期寄存器	
csi_gpta_change_ch_duty	改变比较值寄存器	
csi_gpta_debug_enable	调试使能控制	
csi_gpta_evtrg_enable	使能或禁止事件	

csi_gpta_onetimesoftware_output	一次性软件强制输出控制	
csi_gpta_aqcsfload_config	AQCSF寄存器载入时机设置	
csi_gpta_continuous_software_waveform	连续软件强制输出控制	
csi_gpta_int_enable	中断使能控制器	
csi_gpta_set_sync	同步事件配置	
csi_gpta_set_extsync_chnl	同步事件用于事件触发输出	
csi_gpta_set_sync_filter	同步事件滤波器控制	
csi_gpta_rearm_sync	单次模式下同步事件重使能	
csi_gpta_set_evtrg	事件输出功能配置	
csi_gpta_set_evcntinit	事件输出计数功能配置	
csi_gpta_reglk_config	链接寄存功能设置	
csi_gpta_burst_enable	群脉冲模式配置	
csi_gpta_timer_init	定时功能初始化	
csi_gpta_count_mode	设置GPTA计数器工作模式	
csi_gpta_set_phsr	相位设置	

17.3 API详细说明

17.3.1 gpta0_irqhandler_pro

*__attribute__((weak)) void gpta0_irqhandler_pro(csp_gpta_t *ptGptaBase)*

17.3.1.1 功能描述

定时器0中断回调函数。

17.3.1.2 参数/返回值说明

1. 参数

   ptGptaBase： 每个功能模块都有对应的ptGptaBase，枚举定义详见csp_gpta_t

2. 返回值

   无

3. 参数说明表

参数	说明	概述及其变量位置

ptGptaBase	csp_gpta_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_gpta.h中定义

17.3.2 gpta1_irqhandler_pro

```
__attribute__((weak)) void gpta1_irqhandler_pro(csp_gpta_t *ptGptaBase)
```

17.3.2.1 功能描述

定时器1中断回调函数。

17.3.2.2 参数/返回值说明

4. 参数

   ptGptaBase： 每个功能模块都有对应的ptGptaBase，枚举定义详见csp_gpta_t

5. 返回值

   无

6. 参数说明表

参数	说明	概述及其变量位置
ptGptaBase	csp_gpta_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_gpta.h中定义

17.3.3 gpta_capture_init

*csi_error_t csi_gpta_capture_init(csp_gpta_t *ptGptaBase, csi_gpta_captureconfig_t *ptGptaPwmCfg)*

17.3.3.1 功能描述

定时器工作在捕获模式下的参数定义。

17.3.3.2 参数/返回值说明

1. 参数

   ptGptaBase: 定义详见csp_gpta_t

   ptGptaPwmCfg: 定义详见csi_gpta_captureconfig_t

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	枚举变量位置
ptGptaBase	csp_gpta_t中定义	在csp_gpta.h中定义

ptGptaPwmCfg	typedef struct csi_gpta_captureconfig csi_gpta_captureconfig_t;
	struct csi_gpta_captureconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_gpta_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint8_t byCaptureCapLden;
	uint8_t byCaptureRearm;
	uint8_t byCaptureCapmd;
	uint8_t byCaptureStopWrap;
	uint8_t byCaptureLdaret;
	uint8_t byCaptureLdbret;
	uint8_t byCaptureLdaaret;
	uint8_t byCaptureLdbaret;
	uint32_t wInt;
	};
		在gpta.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.4 csi_gpta_wave_init

*csi_error_t csi_gpta_wave_init(csp_gpta_t *ptGptaBase, csi_gpta_pwmconfig_t *ptGptaPwmCfg)*

17.3.4.1 功能描述

定时器工作在波形模式下的定义

17.3.4.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

   ptGptaPwmCfg：定义详见csi_gpta_pwmconfig_t。

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

ptGptaPwmCfg	typedef struct csi_gpta_pwmconfig csi_gpta_pwmconfig_t;
	struct csi_gpta_pwmconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_gpta_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint32_t wFreq; //TIMER PWM OUTPUT frequency
	uint32_t wInt;
	uint8_t byCks;
	};
		在gpta.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.5 csi_gpta_channel_cmpload_config

*csi_error_t csi_gpta_channel_cmpload_config(csp_gpta_t *ptGptaBase, csp_gpta_cmpdata_ldmd_e tld, csp_gpta_ldamd_e tldamd ,csi_gpta_camp_e channel)*

17.3.5.1 功能描述

定时器CMPA.CMPB载入时机设置

17.3.5.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   tld： 定义详见 csp_gpta_cmpdata_ldmd_e

   tldamd： 定义详见 csp_gpta_ldamd_e

channel： 定义详见 csi_gpta_camp_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
tld	typedef enum { GPTA_CMPLD_SHDW = 0, GPTA_CMPLD_IMM }csp_gpta_cmpdata_ldmd_e;	在csp_gpta.h中定义
tldamd	typedef enum{ GPTA_LDCMP_NEVER = 0, GPTA_LDCMP_ZRO, GPTA_LDCMP_PRD, GPTA_LDCMP_LD_SYNC = 4, }csp_gpta_ldamd_e;	在csp_gpta.h中定义

channel	typedef enum{
	GPTA_CAMPA=1,
	GPTA_CAMPB,
	// GPTA_CAMPC,
	// GPTA_CAMPD
	}csi_gpta_camp_e;
		在gpta.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.6 csi_gpta_channel_config

*csi_error_t csi_gpta_channel_config(csp_gpta_t *ptGptaBase, csi_gpta_pwmchannel_config_t *tPwmCfg, csi_gpta_channel_e channel)*

17.3.6.1 功能描述

设置通道参数

17.3.6.2 参数/返回值说明

4. 参数

ptGptaBase：定义详见csp_gpta_t

tPwmCfg：定义详见 csi_gpta_pwmchannel_config_t

channel：定义详见csi_gpta_channel_e

5. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

6. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

tPwmCfg	typedef struct csi_gpta_pwmchannel_config csi_gpta_pwmchannel_config_t;
	struct csi_gpta_pwmchannel_config {
	uint8_t byActionZro; //
	uint8_t byActionFrd; //
	uint8_t byActionCau; //
	uint8_t byActionCad; //
	uint8_t byActionCbu; //
	uint8_t byActionCbd; //
	uint8_t byActionT1u; //
	uint8_t byActionT1d; //
	uint8_t byActionT2u; //
	uint8_t byActionT2d; //
	uint8_t byChoiceCasel;
	uint8_t byChoiceCbsel;
	};
		在gpta.h中定义

channel	typedef enum{ GPTA_CHANNEL_1=1, GPTA_CHANNEL_2 }csi_gpta_channel_e;	在gpta.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.7 csi_gpta_channel_aqload_config

*csi_error_t csi_gpta_channel_aqload_config(csp_gpta_t *ptGptaBase, csp_gpta_ld_e tld, csp_gpta_ldamd_e tldamd ,csi_gpta_channel_e channel)*

17.3.7.1 功能描述

通道PWM1、PWM2的波形载入时机设置

17.3.7.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见csp_gpta_t

tld： 定义详见csp_gpta_ld_e

tldamd： 定义详见csp_gpta_ldamd_e

channel： 定义详见csi_gpta_channel_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义
tld	typedef enum{ GPTA_LD_IMM = 0, GPTA_LD_SHDW }csp_gpta_ld_e;	在csp_gpta.h中定义
tldamd	typedef enum{ GPTA_LDCMP_NEVER = 0, GPTA_LDCMP_ZRO, GPTA_LDCMP_PRD, GPTA_LDCMP_LD_SYNC = 4, }csp_gpta_ldamd_e;	
channel	typedef enum{ GPTA_CHANNEL_1=1, GPTA_CHANNEL_2 }csi_gpta_channel_e;	在gpta.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.8 csi_gpta_global_config

*csi_error_t csi_gpta_global_config(csp_gpta_t *ptGptaBase,csi_gpta_Global_load_control_config_t *Global)*

17.3.8.1 功能描述

设置全局载入控制器参数

17.3.8.2 参数/返回值说明

1. 参数

ptGptaBase：定义详见csp_gpta_t

Global：定义详见 csi_gpta_Global_load_control_config_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

Global	typedef struct csi_gpta_Global_load_control_config csi_gpta_Global_load_control_config_t;
	struct csi_gpta_Global_load_control_config{
	bool bGlden;
	bool bOstmd;
	uint8_t bGldprd;
	uint8_t byGldmd;
	};
		在gpta.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.9 csi_gpta_gldcfg

*csi_error_t csi_gpta_gldcfg(csp_gpta_t *ptGptaBase ,csi_gpta_Global_load_gldcfg Glo,bool bENABLE)*

17.3.9.1 功能描述

全局载入对象使能或禁止

17.3.9.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

Glo： 定义详见 csi_gpta_Global_load_gldcfg

bENABLE: ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义
Global	typedef enum { byprdr_A=0, bycmpa_A, bycmpb_A, byaqcra_A=8, byaqcrb_A, byaqcsf_A=12, }csi_gpta_Global_load_gldcfg;	

csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.10 csi_gpta_global_sw

*void csi_gpta_global_sw(csp_gpta_t *ptGptaBase)*

17.3.10.1 功能描述

软件触发全局载入

17.3.10.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

2. 返回值：无

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

17.3.11 csi_gpta_global_rearm

*void csi_gpta_global_rearm(csp_gpta_t *ptGptaBase)*

17.3.11.1 功能描述

单次模式下全局载入重使能

17.3.11.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

17.3.12 csi_gpta_start

*void csi_gpta_start(csp_gpta_t *ptgptaBase)*

17.3.12.1 功能描述

定时器开始计数

17.3.12.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

17.3.13 csi_gpta_stop

*void csi_gpta_stop(csp_gpta_t *ptgptaBase)*

17.3.13.1 功能描述

定时器停止计数

17.3.13.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

2. 返回值：无

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

17.3.14 csi_gpta_set_start_mode

*void csi_gpta_set_start_mode(csp_gpta_t *ptgptaBase, csi_gpta_stmd_e eMode)*

17.3.14.1 功能描述

定时器开始位模式选择（普通开始/开始的同时触发同步事件0）

17.3.14.2 参数/返回值说明

1. 参数

   ptGptaBase：定义详见csp_gpta_t

   eMode： 定义详见 csi_gpta_stmd_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

eMode	`typedef enum {`
	`  GPTA_SW = 0,`
	`  GPTA_SYNC`
	`}csi_gpta_stmd_e;`
		在gpta.h中定义

17.3.15 csi_gpta_set_os_mode

*void csi_gpta_set_os_mode(csp_gpta_t *ptgptaBase, csi_gpta_opmd_e eMode)*

17.3.15.1 功能描述

定时器工作模式设定（单次或连续）

17.3.15.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   eMode： 定义详见 csi_gpta_opmd_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
eMode	typedef enum { GPTA_OP_CONT = 0, GPTA_OP_OT, } csi_gpta_opmd_e;	在gpta.h中定义

17.3.16 csi_gpta_set_stop_st

*void csi_gpta_set_stop_st(csp_gpta_t *ptgptaBase, csp_gpta_stpst_e eSt)*

17.3.16.1 功能描述

波形输出被停止时，输出端口的缺省状态

17.3.16.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   eSt： 定义详见 csp_gpta_stpst_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
eSt	typedef enum { GPTA_STPST_HZ = 0, GPTA_STPST_LOW } csp_gpta_stpst_e;	在gpta.h中定义

17.3.17 csi_gpta_get_prdr

*uint16_t csi_gpta_get_prdr(csp_gpta_t *ptgptaBase)*

17.3.17.1 功能描述

获得周期寄存器值

17.3.17.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

2. 返回值

   返回周期寄存器值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
return	返回周期寄存器值	

17.3.18 csi_gpta_change_ch_duty

*csi_error_t csi_gpta_change_ch_duty(csp_gpta_t *ptGptaBase, csi_gpta_chtype_e eCh, uint32_t wActiveTime)*

17.3.18.1 功能描述

改变比较值寄存器

17.3.18.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   eCh： 定义详见 csi_gpta_chtype_e

   wActiveTime：周期寄存器的x%

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

eCh	typedef enum{
	GPTA_CAMPA=1,
	GPTA_CAMPB,
	// GPTA_CAMPC,
	// GPTA_CAMPD
	}csi_gpta_camp_e;
		在gpta.h中定义

wActiveTime	周期寄存器的x%	

17.3.19 csi_gpta_debug_enable

*void csi_gpta_debug_enable(csp_gpta_t *ptGptaBase, bool bEnable)*

17.3.19.1 功能描述

调试使能控制

17.3.19.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

bEnable： 使能或禁止

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
bEnable	ENABLE/DISABLE	在common.h中定义

17.3.20 csi_gpta_evtrg_enable

*csi_error_t csi_gpta_evtrg_enable(csp_gpta_t *ptGptaBase, csi_gpta_trgout_e byCh, bool bEnable)*

17.3.20.1 功能描述

事件输出使能或禁止

17.3.20.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   byCh： 定义详见csi_gpta_trgout_e

   bEnable： ENABLE/DISABLE

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

byCh	*/
	typedef enum{
	GPTA_TRG_OUT0 = 0, //trigger out0
	GPTA_TRG_OUT1, //trigger out1
	// GPTA_TRG_OUT2, //trigger out2
	// GPTA_TRG_OUT3 //trigger out3
	}csi_gpta_trgout_e;
		在gpta.h中定义

bEnable	ENABLE/DISABLE	在common.h中定义
csi_error_t	csi_error_t中定义	在common.h中定义

17.3.21 csi_gpta_onetimesoftware_output

*csi_error_t csi_gpta_onetimesoftware_output(csp_gpta_t *ptGptaBase, uint16_t byCh,csp_gpta_action_e bEnable)*

17.3.21.1 功能描述

一次性软件强制输出控制

17.3.21.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   byCh： GPTA_OSTSFA/GPTA_OSTSFB

   bEnable： 定义详见csp_gpta_action_e

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
byCh	///AQOSF #define GPTA_OSTSFA_POS (0) #define GPTA_OSTSFB_POS (4) #define GPTA_OSTSFA (1) #define GPTA_ACTA_POS (1) #define GPTA_ACTA_MSK (0x3 << GPTA_ACTA_POS) #define GPTA_OSTSFB (0x1 << 4) #define GPTA_ACTB_POS (5) #define GPTA_ACTB_MSK (0x3 << GPTA_ACTB_POS) #define GPTA_AQCSF_LDTIME_POS (16) #define GPTA_AQCSF_LDTIME_MSK (0x3 << GPTA_AQCSF_LDTIME_POS)	在csp_gpta.h中定义
bEnable	typedef enum { GPTA_NA = 0, GPTA_LO, GPTA_HI, GPTA_TG } csp_gpta_action_e;	在csp_gpta.h中定义
csi_error_t	csi_error_t中定义	在common.h中定义

17.3.22 csi_gpta_aqcsfload_config

*void csi_gpta_aqcsfload_config (csp_gpta_t *ptGptaBase, csp_gpta_aqosf_e bEnable)*

17.3.22.1 功能描述

AQCSF寄存器从Shadow载入到Active的控制

17.3.22.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   csp_gpta_aqosf_e：定义详见csp_gpta_aqosf_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
csp_gpta_aqosf_e	typedef enum { GPTA_AQCSF_NOW=0, GPTA_AQCSF_ZRO, GPTA_AQCSF_PRD, GPTA_AQCSF_ZRO_PRD }csp_gpta_aqosf_e;	在gpta.h中定义

17.3.23 csi_gpta_continuous_software_waveform

*csi_error_t csi_gpta_continuous_software_waveform(csp_gpta_t *ptGptaBase, csi_gpta_channel_e byCh, csp_gpta_aqcsf_e bEnable)*

17.3.23.1 功能描述

连续软件强制输出控制

17.3.23.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   byCh： 定义详见csi_gpta_channel_e

   bEnable 定义详见csp_gpta_aqcsf_e

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义
csi_gpta_channel_e	typedef enum{ GPTA_CHANNEL_1=1, GPTA_CHANNEL_2 }csi_gpta_channel_e; typedef enum{	在gpta.h中定义

csp_gpta_aqcsf_e	typedef enum {
	GPTA_AQCSF_NONE=0,
	GPTA_AQCSF_L,
	GPTA_AQCSF_H,
	GPTA_AQCSF_NONE1
	}csp_gpta_aqcsf_e;
	typedef enum{
		在gpta.h中定义

csi_error_t	csi_error_t中定义	在common.h中定义

17.3.24 csi_gpta_int_enable

*void csi_gpta_int_enable(csp_gpta_t *ptGptaBase, csp_gpta_int_e eInt, bool bEnable)*

17.3.24.1 功能描述

中断使能

17.3.24.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

eInt： 定义详见csp_gpta_int_e

bEnable ENABLE/DISABLE

2. 返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_gpta.h中定义

csp_gpta_int_e	typedef enum{
	GPTA_INT_TRGEV0 = 0x1,
	GPTA_INT_TRGEV1 = 0x2,
	// GPTA_INT_TRGEV2 = 0x4,
	// GPTA_INT_TRGEV3 = 0x8,
	GPTA_INT_CAPLD0 = 0x1 << 4,
	GPTA_INT_CAPLD1 = 0x1 << 5,
	// GPTA_INT_CAPLD2 = 0x1 << 6,
	// GPTA_INT_CAPLD3 = 0x1 << 7,
	GPTA_INT_CAU = 0x1 << 8,
	GPTA_INT_CAD = 0x1 << 9,
	GPTA_INT_CBU = 0x1 << 10,
	GPTA_INT_CBD = 0x1 << 11,
	GPTA_INT_PEND = 0x1 << 16
	}csp_gpta_int_e;	

bEnable	ENABLE/DISABLE	在common.h中定义

17.3.25 csi_gpta_set_sync

*void csi_gpta_set_sync(csp_gpta_t *ptGptaBase, csi_gpta_trgin_e eTrgIn, csi_gpta_trgmode_e eTrgMode, csi_gpta_arearm_e eAutoRearm)*

17.3.25.1 功能描述

同步事件配置

17.3.25.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

eTrgIn： 定义详见csi_gpta_trgin_e

eTrgMode： 定义详见csi_gpta_trgmode_e

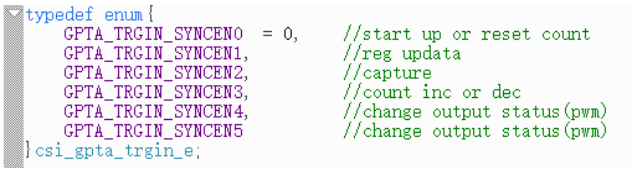
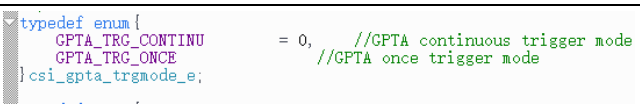
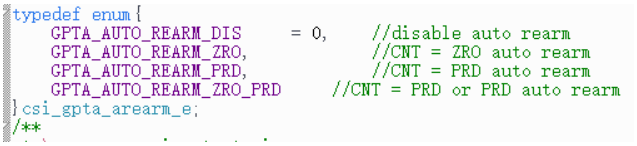
eAutoRearm： 定义详见csi_gpta_arearm_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义

eTrgIn		
eTrgMode		
eAutoRearm		

17.3.26 csi_gpta_set_extsync_chnl

*csi_error_t csi_ept_set_sync2evtrg (csp_gpta_t *ptGptaBase, csi_gpta_trgin_e eTrgIn, uint8_t byTrgChx)*

17.3.26.1 功能描述

事件输出选择（同步事件中选择）

17.3.26.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   eTrgIn： 定义详见csi_gpta_trgin_e

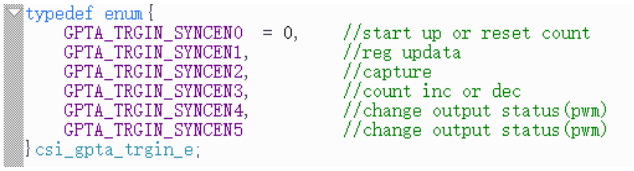
   byTrgChx： 0/1

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义
eTrgIn		

byTrgChx	0/1	
csi_error_t	csi_error_t中定义	在common.h中定义

17.3.27 csi_gpta_set_sync_filter

*csi_error_t csi_gpta_set_sync_filter(csp_gpta_t *ptGptaBase, csi_gpta_filter_config_t *ptFilter)*

17.3.27.1 功能描述

同步事件计数器配置

17.3.27.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   ptFilter： 定义详见csi_gpta_filter_config_t

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义

ptFilter	`typedef struct {`
	`  uint8_t byFiltSrc; //filter input signal source`
	`  uint8_t byWinInv; //window inversion`
	`  uint8_t byWinAlign; //window alignment`
	`  uint8_t byWinCross; //window cross`
	`  uint16_t byWinOffset; //window offset`
	`  uint16_t byWinWidth; //window width`
	`} csi_gpta_filter_config_t;`	

csi_error_t	csi_error_t中定义	在common.h中定义

17.3.28 csi_gpta_rearm_sync

*void csi_gpta_rearm_sync(csp_gpta_t *ptGptaBase,csi_gpta_trgin_e eTrgin)*

17.3.28.1 功能描述

单次模式下同步事件重使能

17.3.28.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

eTrgin： 定义详见csi_gpta_trgin_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义

eTrgin	typedef enum{
	GPTA_TRGIN_SYNCEN0 = 0, //start up or reset count
	GPTA_TRGIN_SYNCEN1, //reg updata
	GPTA_TRGIN_SYNCEN2, //capture
	GPTA_TRGIN_SYNCEN3, //count inc or dec
	GPTA_TRGIN_SYNCEN4, //change output status(pwm)
	GPTA_TRGIN_SYNCEN5 //change output status(pwm)
	}csi_gpta_trgin_e;	

17.3.29 csi_gpta_set_evtrg

*csi_error_t csi_gpta_set_evtrg(csp_gpta_t *ptGptaBase, csi_gpta_trgout_e byTrgOut, csp_gpta_trgsrc0_e eTrgSrc)*

17.3.29.1 功能描述

事件输出功能配置

17.3.29.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

byTrgOut： 定义详见csi_gpta_trgout_e

eTrgSrc： 定义详见csp_gpta_trgsrc0_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义

byTrgOut	typedef enum{
	GPTA_TRG_OUT0 = 0, //trigger out0
	GPTA_TRG_OUT1, //trigger out1
	// GPTA_TRG_OUT2, //trigger out2
	// GPTA_TRG_OUT3 //trigger out3
	}csi_gpta_trgout_e;	

eTrgSrc	typedef enum{ GPTA_TRG01_DIS = 0, GPTA_TRG01_ZRO, GPTA_TRG01_PRD, GPTA_TRG01_ZRO_PRD, GPTA_TRG01_CMPA_R, GPTA_TRG01_CMPA_F, GPTA_TRG01_CMPB_R, GPTA_TRG01_CMPB_F, GPTA_TRG01_SYNC = 0xc, GPTA_TRG01_PE0, GPTA_TRG01_PE1, GPTA_TRG01_PE2 }csp_gpta_trgsrc0_e;	在csp_gpta.h中定义
csi_error_t	csi_error_t中定义	在common.h中定义

17.3.30 csi_gpta_set_evcntinit

*csi_error_t csi_gpta_set_evcntinit(csp_gpta_t *ptGptaBase, uint8_t byCntChx, uint8_t byCntVal, uint8_t byCntInitVal)*

17.3.30.1 功能描述

事件输出计数功能配置

17.3.30.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

byCntChx： 事件通道选择0~1

byCntVal： 事件周期值

byCntInitVal： 事件周期初始

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在gpta.h中定义
byCntChx	事件通道选择（0~1）	
byCntVal	事件周期值	
byCntInitVal	事件周期初始化值	

17.3.31 csi_gpta_reglk_config

*csi_error_t csi_gpta_reglk_config(csp_gpta_t *ptGptaBase,csi_gpta_feglk_config_t *Global)*

17.3.31.1 功能描述

链接寄存功能设置

17.3.31.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   Global： 定义详见 csi_gpta_feglk_config_t

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_ gpta.h中定义

Global	typedef struct { uint8_t byPrdr; uint8_t byCmpa; uint8_t byCmpb; uint8_t byGld2; uint8_t byRssr; uint8_t byEmslclr; uint8_t byEmhlclr; uint8_t byEmicr; uint8_t byEmfrcr; uint8_t byAqosf; uint8_t byAqcsf; } csi_gpta_feglk_config_t;	在gpta.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.32 csi_gpta_burst_enable

*csi_error_t csi_gpta_burst_enable(csp_gpta_t *ptGptaBase,uint8_t byCgsrc,uint8_t byCgflt, bool bEnable)*

17.3.32.1 功能描述

群脉冲模式配置

17.3.32.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

byCgsrc：群脉冲门控输入源选择（0：GPTA_CHA 1:GPTA_CHB）

byCgflt：门控输入数字滤波控制(0~7)

bEnable：ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_ gpta.h中定义
byCgsrc	8位参数，0：GPTA_CHA 1:GPTA_CHB	
byCgflt	8位参数，范围为0~7，详解如下：	在csp_gpta.h中定义

	typedef enum { GPTA_CGFLT_BP = 0, GPTA_CGFLT_2, GPTA_CGFLT_3, GPTA_CGFLT_4, GPTA_CGFLT_6, GPTA_CGFLT_8, GPTA_CGFLT_16, GPTA_CGFLT_32 }csp_gpta_cnflt_e;	
bEnable	ENABLE/DISABLE	在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.33 csi_gpta_timer_init

*csi_error_t csi_gpta_timer_init(csp_gpta_t *ptGptaBase, uint32_t wTimeOut)*

17.3.33.1 功能描述

定时功能初始化

17.3.33.2 参数/返回值说明

1. 参数

   ptGptaBase： 定义详见 csp_gpta_t

   wTimeOut：计数器溢出时间，即定时时间，单位us

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_ gpta.h中定义
wTimeOut	uint32_t 类型数值，单位：us	定时时间，单位us
csi_error_t	csi_error_t 中定义值	在common.h中定义

17.3.34 csi_gpta_count_mode

*void csi_gpta_count_mode(csp_gpta_t *ptGptaBase, csi_gpta_opmd_e eCntMode)*

17.3.34.1 功能描述

设置GPTA计数器工作模式

17.3.34.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

eCntMode：计数工作模式，枚举定义详见csi_gpta_opmd_e。

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_ gpta.h中定义
eCntMode	typedef enum { GPTA_OP_CONT = 0, GPTA_OP_OT, } csi_gpta_opmd_e;	计数工作模式有两种，连续计数模式、单次触发模式 默认为连续计数模式

17.3.35 csi_gpta_set_phsr

*void csi_gpta_set_phsr(csp_gpta_t *ptGptaBase, uint32_t wPhsr,bool bEnable)*

17.3.35.1 功能描述

相位设置

17.3.35.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gpta_t

wPhsr：相位计数值设置

bEnable：ENABLE/DISABLE

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptaBase	csp_gpta_t中说明	在csp_ gpta.h中定义
wPhsr	uint32_t 类型数值	相位计数值
bEnable	ENABLE/DISABLE	在common.h中定义

18 GPTB

18.1 概述

通用定时器（General Purpose Timer）作为 MCU 的关键外设，可以在各种功率控制应用中发挥关键控制作用。通过灵活的 PWM 输出，可以适用于各种复杂多变的应用，这些应用包括数字马达控制、开关电源控制、不间断电源控制、变频功率转换控制等。GPTB 内部包含两个 16 位的定时/计数模块，分别支持 2 种工作模式(捕捉模式和波形发生器模式)。

18.2 API列表

Table 18-1 GPTB CSI接口函数

API	说明	函数位置
gptb0_irqhandler_pro	定时器0中断回调函数	
gptb1_irqhandler_pro	定时器1中断回调函数	
csi_gptb_capture_init	定时器捕获模式参数设置	
csi_gptb_wave_init	定时器波形模式参数基础设置（计数模式，周期，占空比等）	
csi_gptb_channel_cmpload_config	定时器CMPA.CMPB载入时机设置	
csi_gptb_channel_config	设置通道PWM1、PWM2的波形	
csi_gptb_channel_aqload_config	通道PWM1、PWM2的波形载入时机设置	
csi_gptb_dz_config	死区功能基本配置	
csi_gptb_dbload_config	死区配置载入控制	
csi_gptb_channelmode_config	死区通道配置	
csi_gptb_emergency_cfg	紧急状态输入基本参数设置	
csi_gptb_emergency_pinxout	设置紧急事件到来时，CHAX/Y, CHBX处的状态	
csi_gptb_gldcfg	全局载入对象使能或禁止	
csi_gptb_global_config	全局载入控制配置	
csi_gptb_global_sw	软件产生一次全局载入	
csi_gptb_global_rearm	单次模式下全局载入重使能	
csi_gptb_start	定时器开始计数	
csi_gptb_stop	定时器停止计数	

csi_gptb_set_start_mode	定时器开始位模式选择（普通开始/开始的同时触发同步事件0）	
csi_gptb_set_os_mode	定时器模式选择（单次模式或连续模式）	
csi_gptb_set_stop_st	波形输出被停止时，输出端口的缺省状态	
csi_gptb_get_prdr	读取定时器周期寄存器	
csi_gptb_change_ch_duty	改变比较值寄存器	
csi_gptb_force_em	紧急状态软件触发	
csi_gptb_get_hdlck_st	获取紧急硬锁止状态寄存器	
csi_gptb_clr_hdlck	紧急硬锁止状态清除寄存器	
csi_gptb_get_sftlck_st	获取紧急软锁止状态寄存器	
csp_gptb_clr_sftlck	紧急软锁止状态清除寄存器	
csi_gptb_debug_enable	使能或禁止debug模式	
csi_gptb_emergency_int_enable	紧急状态中断使能或禁止	
csi_gptb_evtrg_enable	使能或禁止事件	
csi_gptb_onetimesoftware_output	一次性软件强制输出控制	
csi_gptb_aqcsfload_config	AQCSF寄存器载入时机设置	
csi_gptb_continuous_software_waveform	连续软件强制输出控制	
csi_gptb_int_enable	中断使能控制器	
csi_gptb_set_sync	同步事件配置	
csi_gptb_set_extsync_chnl	同步事件用于事件触发输出	
csi_gptb_set_sync_filter	同步事件滤波器控制	
csi_gptb_rearm_sync	单次模式下同步事件重使能	
csi_gptb_set_evtrg	事件输出功能配置	
csi_gptb_set_evcntinit	事件输出计数功能配置	
csi_gptb_reglk_config	链接寄存功能设置	
csi_gptb_burst_enable	群脉冲模式配置	
csi_gptb_timer_init	定时功能初始化	
csi_gptb_count_mode	设置GPTB计数器工作模式	
csi_gptb_set_phsr	相位设置	

18.3 API详细说明

gptb0_irqhandler_pro

*__attribute__((weak)) void gptb0_irqhandler_pro(csp_gptb_t *ptGptbBase)*

18.3.1.1 功能描述

定时器0中断回调函数。

18.3.1.2 参数/返回值说明

1. 参数

   ptGptbBase： 每个功能模块都有对应的ptGptbBase，枚举定义详见csp_gptb_t

2. 返回值

   无

3. 参数说明表

参数	说明	概述及其变量位置
ptGptbBase	csp_gptb_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_gptb.h中定义

gptb1_irqhandler_pro

18.3.2.1 功能描述

定时器 1 中断回调函数。

18.3.2.2 参数/返回值说明

1. 参数

   ptGptbBase： 每个功能模块都有对应的ptGptbBase，枚举定义详见csp_gptb_t

2. 返回值

   无

3. 参数说明表

参数	说明	概述及其变量位置
ptGptbBase	csp_gptb_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_gptb.h中定义

csi_gptb_capture_init

*csi_error_t csi_gptb_capture_init(csp_gptb_t *ptGptbBase, csi_gptb_captureconfig_t *ptgptbPwmCfg)*

18.3.3.1 功能描述

捕获模式下定时器基本参数设定

18.3.3.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见csp_gptb_t

   ptgptbPwmCfg： 定义详见csi_gptb_captureconfig_t

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

ptgptbPwmCfg	typedef struct csi_gptb_captureconfig csi_gptb_captureconfig_t;
	struct csi_gptb_captureconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_gptb_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint8_t byCaptureCapLden;
	uint8_t byCapSrcMode; //CAPMODE_SEL
	uint8_t byCaptureRearm;
	uint8_t byCaptureCapmd;
	uint8_t byCaptureStopWrap;
	uint8_t byCaptureLdaret;
	uint8_t byCaptureLdbret;
	uint8_t byCaptureLdaaret;
	uint8_t byCaptureLdbaret;
	uint32_t wInt;
	};	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_wave_init

*csi_error_t csi_gptb_wave_init(csp_gptb_t *ptGptbBase, csi_gptb_pwmconfig_t *ptgptbPwmCfg)*

18.3.4.1 功能描述

定时器波形模式参数基础设置（计数模式，周期，占空比等）

18.3.4.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见csp_gptb_t

   ptgptbPwmCfg： 定义详见csi_gptb_pwmconfig_t

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

ptgptbPwmCfg	typedef struct csi_gptb_pwmconfig csi_gptb_pwmconfig_t;
	struct csi_gptb_pwmconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_gptb_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint32_t wFreq; //TIMER PWM OUTPUT frequency
	uint32_t wInt;
	};	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_channel_cmpload_config

*csi_error_t csi_gptb_channel_cmpload_config(csp_gptb_t *ptGptbBase, csp_gptb_cmpdata_ldmd_e tld, csp_gptb_ldamd_e tldamd ,csi_gptb_camp_e channel)*

18.3.5.1 功能描述

定时器 CMPA.CMPB 载入时机设置

18.3.5.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见csp_gptb_t

   tld： 定义详见csp_gptb_cmpdata_ldmd_e

   tldamd： 定义详见csp_gptb_ldamd_e

   channel： 定义详见csi_gptb_camp_e

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义
tld	typedef enum { GPTB_CMPLD_SHDW = 0, GPTB_CMPLD_IMM }csp_gptb_cmpdata_ldmd_e;	在csp_gptb.h中定义
tldamd	typedef enum{ GPTB_LDCMP_NEVER= 0, GPTB_LDCMP_ZRO , GPTB_LDCMP_PRD, GPTB_LDCMP_LD_SYNC=4, }csp_gptb_ldamd_e;	
channel	typedef enum{ GPTB_CAMPA=1, GPTB_CAMPB }csi_gptb_camp_e;	在gptb.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_channel_config

*csi_error_t csi_gptb_channel_config(csp_gptb_t *ptGptbBase, csi_gptb_pwmchannel_config_t *tPwmCfg, csi_gptb_channel_e channel)*

18.3.6.1 功能描述

设置通道PWM1、PWM2的波形

18.3.6.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见csp_gptb_t

   tPwmCfg： 定义详见csi_gptb_pwmchannel_config_t

   channel： 定义详见csi_gptb_channel_e

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

tPwmCfg	`typedef struct csi_gptb_pwmchannel_config csi_gptb_pwmchannel_config_t;`
	`struct csi_gptb_pwmchannel_config {`
	`uint8_t byActionZro; //`
	`uint8_t byActionPrd; //`
	`uint8_t byActionC1u; //`
	`uint8_t byActionC1d; //`
	`uint8_t byActionC2u; //`
	`uint8_t byActionC2d; //`
	`uint8_t byActionT1u; //`
	`uint8_t byActionT1d; //`
	`uint8_t byActionT2u; //`
	`uint8_t byActionT2d; //`
	`uint8_t byChoiceC1sel;`
	`uint8_t byChoiceC2sel;`
	`};`	

channel	`typedef enum{`
	`GPTB_CHANNEL_1=1,`
	`GPTB_CHANNEL_2`
	`}csi_gptb_channel_e;`	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_channel_aqload_config

*csi_error_t csi_gptb_channel_aqload_config(csp_gptb_t *ptGptbBase, csp_gptb_ld_e tld, csp_gptb_ldamd_e tldamd ,csi_gptb_channel_e channel)*

18.3.7.1 功能描述

通道PWM1、PWM2的波形载入时机设置

18.3.7.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见csp_gptb_t

tld： 定义详见csp_gptb_ld_e

tldamd： 定义详见csp_gptb_ldamd_e

channel： 定义详见csi_gptb_channel_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义
tld	typedef enum{ GPTB_LD_IMM = 0, GPTB_LD_SHDW }csp_gptb_ld_e;	在csp_gptb.h中定义
tldamd	typedef enum{ GPTB_LDCMP_NEVER= 0, GPTB_LDCMP_ZRO , GPTB_LDCMP_PRD, GPTB_LDCMP_LD_SYNC=4, }csp_gptb_ldamd_e;	
channel	typedef enum{ GPTB_CHANNEL_1=1, GPTB_CHANNEL_2 }csi_gptb_channel_e;	在gptb.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_dz_config

*csi_error_t csi_gptb_dz_config(csp_gptb_t *ptGptbBase, csi_gptb_deadzone_config_t *tCfg)*

18.3.8.1 功能描述

死区功能基本配置

18.3.8.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t csp_ept_t

tCfg： 定义详见 csi_gptb_deadzone_config_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_ept_t中说明	在ept.h中定义

tCfg	typedef struct csi_gptb_deadzone_config csi_gptb_deadzone_config_t;
	struct csi_gptb_deadzone_config
	{
	uint8_t byChxOuselS1S0 ; //
	uint8_t byChxPolarityS3S2 ; //
	uint8_t byChxInselS5S4 ; //
	uint8_t byChxOutSwapS8S7 ; //
	uint8_t byDcksel;
	uint8_t byChaDedb;
	uint8_t byChbDedb;
	uint8_t byChcDedb;
	uint16_t hwDpsc; //
	uint16_t hwRisingEdgereGister ; //
	uint16_t hwFallingEdgereGister; //
	};	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_dbload_config

*csi_error_t csi_gptb_dbload_config(csp_gptb_t *ptGptbBase, csi_gptb_dbdldr_e byVal,csp_gptb_shdw_e byWod,csp_gptb_lddb_e byWod2)*

18.3.9.1 功能描述

死区配置载入控制

18.3.9.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byVal： 定义详见 csi_gptb_dbdldr_e

byWod： 定义详见 csp_gptb_shdw_e

byWod2： 定义详见 csp_gptb_lddb_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_gptb_t中说明	在gptb.h中定义
byVal	typedef enum{ GPTB_DBCR =0, GPTB_DBDTR, GPTB_DBDTF, GPTB_DCKPSC }csi_gptb_dbdldr_e;	
byWod	typedef enum{ GPTB_SHDW_IMMEDIATE =0, GPTB_SHDW_SHADOW }csp_gptb_shdw_e;	在csp_ept.h中定义
byWod2	typedef enum{ GPTB_LD_NEVER = 0, GPTB_LD_ZRO, GPTB_LD_PRD, GPTB_LD_ZRO_PRD }csp_gptb_lddb_e;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_channelmode_config

*csi_error_t csi_gptb_channelmode_config(csp_gptb_t *ptGptbBase,csi_gptb_deadzone_config_t *tCfg,csi_gptb_channel_e byCh)*

18.3.10.1 功能描述

死区通道配置

18.3.10.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

tCfg： 定义详见 csi_gptb_deadzone_config_t

byCh： 定义详见 csi_gptb_channel_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_gptb_t中说明	在gptb.h中定义

tCfg	typedef struct csi_gptb_deadzone_config csi_gptb_deadzone_config_t;
	struct csi_gptb_deadzone_config
	{
	uint8_t byChxOuselS1S0 ; //
	uint8_t byChxPolarityS3S2 ; //
	uint8_t byChxInselS5S4 ; //
	uint8_t byChxOutSwapS8S7 ; //
	uint8_t byDcksel;
	uint8_t byChaDedb;
	uint8_t byChbDedb;
	uint8_t byChcDedb;
	uint16_t hwDpsc; //
	uint16_t hwRisingEdgereGister ; //
	uint16_t hwFallingEdgereGister; //
	};	

byCh	typedef enum{ GPTB_CHANNEL_1=1, GPTB_CHANNEL_2 }csi_gptb_channel_e; 此参数只能配置为GPTB_CHANNEL_1	
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_emergency_cfg

*csi_error_t csi_gptb_emergency_cfg(csp_gptb_t *ptGptbBase, csi_gptb_emergency_config_t *tCfg)*

18.3.11.1 功能描述

紧急状态输入基本参数设置

18.3.11.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   tCfg： 定义详见 csi_gptb_emergency_config_t

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_gptb_t中说明	在gptb.h中定义

tCfg	typedef struct csi_gptb_emergency_ csi_gptb_emergency_config_t;
	struct csi_gptb_emergency_
	{
	uint8_t byEpxInt ;
	uint8_t byPolEbix;
	uint8_t byEpx;
	uint8_t byEpxLckmd;
	uint8_t byFltpace0;
	uint8_t byFltpace1;
	uint8_t byOrl0;
	uint8_t byOrl1;
	uint8_t byOsrshdw;
	};	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_emergency_pinxout

*csi_error_t csi_gptb_emergency_pinxout(csp_gptb_t *ptGptbBase,csi_gptb_osrchx_e byCh ,csp_gptb_emout_e byCh2)*

18.3.12.1 功能描述

设置紧急事件到来时，CHAX/Y, CHBX处的状态

18.3.12.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byCh： 定义详见 csi_gptb_osrchx_e

byCh2： 定义详见 csp_gptb_emout_e

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_gptb_t中说明	在gptb.h中定义
byCh	typedef enum{ GPTB_EMCOAX =0, GPTB_EMCOBX, GPTB_EMCOAY }csi_gptb_osrchx_e;	
byCh2	typedef enum { B_EM_OUT_HZ, B_EM_OUT_H, B_EM_OUT_L, B_EM_OUT_NONE }csp_gptb_emout_e;	

csi_error_t	csi_error_t 中定义值	common.h

csi_gptb_gldcfg

*csi_error_t csi_gptb_gldcfg(csp_gptb_t *ptGptbBase ,csi_gptb_Global_load_gldcfg Glo,bool bENABLE)*

18.3.13.1 功能描述

全局载入对象使能或禁止

18.3.13.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   Glo： 定义详见 csi_gptb_Global_load_gldcfg

   bENABLE: ENABLE/DISABLE

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义
Global	typedef enum { byprdr_B=0, bycmpa_B, bycmpb_B, byaqcra_B, byaqcrb_B, byaqcsf_B, }csi_gptb_Global_load_gldcfg;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_global_config

*void csi_gptb_global_config(csp_gptb_t *ptGptbBase,csi_gptb_Global_load_control_config_t *Global)*

18.3.14.1 功能描述

全局载入控制配置

18.3.14.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

Global： 定义详见 csi_gptb_Global_load_control_config_t

2. 返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

Global	typedef struct csi_gptb_Global_load_control_config csi_gptb_Global_load_control_config_t;
	struct csi_gptb_Global_load_control_config{
	bool bGlden;
	bool bOstmd;
	uint8_t bGldprd;
	uint8_t byGldmd;
	uint8_t byGldcnt;
	};	

csi_gptb_gload_sw

*void csi_gptb_global_sw(csp_gptb_t *ptGptbBase)*

18.3.15.1 功能描述

软件产生一次全局载入

18.3.15.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置

ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

csi_gptb_gload_rearm

*void csi_gptb_global_rearm(csp_gptb_t *ptGptbBase)*

18.3.16.1 功能描述

单次模式下全局载入重使能

18.3.16.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在gptb.h中定义

csi_gptb_start

*void csi_gptb_start(csp_gptb_t *ptgptbBase)*

18.3.17.1 功能描述

定时器开始计数

18.3.17.2 参数/返回值说明

1. 参数

   ptgptbBase： 定义详见 csp_gptb_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptgptbBase	csp_gptb_t中说明	在gptb.h中定义

csi_gptb_stop

*void csi_gptb_sotp(csp_gptb_t *ptgptbBase)*

18.3.18.1 功能描述

定时器停止计数

18.3.18.2 参数/返回值说明

1. 参数

   ptgptbBase： 定义详见 csp_gptb_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptgptbBase	csp_gptb_t中说明	在gptb.h中定义

csi_gptb_set_start_mode

*void csi_gptb_set_start_mode(csp_gptb_t *ptgptbBase, csi_gptb_stmd_e eMode)*

18.3.19.1 功能描述

定时器开始位模式选择（普通开始/开始的同时触发同步事件0）

18.3.19.2 参数/返回值说明

1. 参数

   ptgptbBase： 定义详见 csp_gptb_t

   eMode： 定义详见 csi_gptb_stmd_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptgptbBase	csp_gptb_t中说明	在gptb.h中定义

eMode	typedef enum { GPTB_SW = 0, GPTB_SYNC }csi_gptb_stmd_e;	

csi_gptb_set_os_mode

*void csi_gptb_set_os_mode(csp_gptb_t *ptgptbBase, csi_gptb_opmd_e eMode)*

18.3.20.1 功能描述

定时器模式选择（单次模式或连续模式）

18.3.20.2 参数/返回值说明

1. 参数

ptgptbBase： 定义详见 csp_gptb_t

eMode： 定义详见 csi_gptb_opmd_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptgptbBase	csp_gptb_t中说明	在gptb.h中定义
eMode	typedef enum{ GPTB_OP_CONT = 0, GPTB_OP_OT }csi_gptb_opmd_e;	

csi_gptb_set_stop_st

*void csi_gptb_set_stop_st(csp_gptb_t *ptgptbBase, csp_gptb_stpst_e eSt)*

18.3.21.1 功能描述

波形输出被停止时，输出端口的缺省状态

18.3.21.2 参数/返回值说明

1. 参数

ptgptbBase： 定义详见 csp_gptb_t

eSt： 定义详见 csp_gptb_stpst_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptgptbBase	csp_ gptb _t中说明	在csp_ gptb.h中定义
eSt	typedef enum{ GPTB_STPST_HZ = 0, GPTB_STPST_LOW }csp_gptb_stpst_e;	

csi_gptb_get_prdr

*uint16_t csi_gptb_get_prdr(csp_gptb_t *ptGptbBase)*

18.3.22.1 功能描述

读取定时器周期寄存器

18.3.22.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

2. 返回值

返回周期值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
uint16_t	返回周期值	

csi_gptb_change_ch_duty

*csi_error_t csi_gptb_change_ch_duty(csp_gptb_t *ptGptbBase, csi_gptb_camp_e eCh, uint32_t wActiveTime)*

18.3.23.1 功能描述

改变比较值寄存器

18.3.23.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

eCh： 定义详见 csi_gptb_camp_e

wActiveTime： 周期值的X%

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
eCh	typedef enum{ GPTB_CAMPA=1, GPTB_CAMPB }csi_gptb_camp_e;	在gptb ept.h中定义
wActiveTime	周期值的X%	
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_force_em

*void csi_gptb_force_em(csp_gptb_t *ptGptbBase, csp_gptb_ep_e byEbi)*

18.3.24.1 功能描述

紧急状态软件触发

18.3.24.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   byEbi： 定义详见 csp_gptb_ep_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

byEbi	typedef enum{
	B_EP0 = 0,
	B_EP1,
	B_EP2,
	B_EP3,
	// EP4,
	// EP5,
	// EP6,
	// EP7,
	}csp_gptb_ep_e;	

csi_gptb_get_hdlck_st

*uint8_t csi_gptb_get_hdlck_st(csp_gptb_t *ptGptbBase)*

18.3.25.1 功能描述

获取紧急硬锁止状态寄存器

18.3.25.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

2. 返回值

   返回紧急硬锁止状态寄存器值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
uint8_t	返回紧急硬锁止状态寄存器值	

csi_gptb_clr_hdlck

*void csi_gptb_clr_hdlck(csp_gptb_t *ptGptbBase, csp_gptb_ep_e eEbi)*

18.3.26.1 功能描述

紧急硬锁止状态清除寄存器

18.3.26.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   eEbi 定义详见 csp_gptb_ep_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

eEbi	typedef enum{
	B_EP0 = 0,
	B_EP1,
	B_EP2,
	B_EP3,
	// EP4,
	// EP5,
	// EP6,
	// EP7,
	}csp_gptb_ep_e;	

csi_gptb_get_sftlck_st

*uint8_t csi_gptb_get_sftlck_st(csp_gptb_t *ptGptbBase)*

18.3.27.1 功能描述

获取紧急软锁止状态寄存器

18.3.27.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

2. 返回值

返回紧急软锁止状态值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
uint8_t	返回紧急软锁止状态值	

csp_gptb_clr_sftlck

*void csp_gptb_clr_sftlck(csp_gptb_t *ptGptbBase, csp_gptb_ep_e eEpi)*

18.3.28.1 功能描述

紧急软锁止状态清除寄存器

18.3.28.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

eEpi： 定义详见 csp_gptb_ep_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_ gptb _t中说明	在csp_ gptb.h中定义

eEpi	typedef enum{
	B_EP0 = 0,
	B_EP1,
	B_EP2,
	B_EP3,
	// EP4,
	// EP5,
	// EP6,
	// EP7,
	}csp_gptb_ep_e;	

csi_gptb_debug_enable

*void csi_gptb_debug_enable(csp_gptb_t *ptGptbBase, bool bEnable)*

18.3.29.1 功能描述

使能或禁止debug模式

18.3.29.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   bEnable： ENABLE/DISABLE

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
bEnable	ENABLE/DISABLE	在common.h中定义

csi_gptb_emergency_int_enable

*void csi_gptb_emergency_int_enable(csp_gptb_t *ptGptbBase, csp_gptb_emint_e eEbi)*

18.3.30.1 功能描述

紧急状态中断使能或禁止

18.3.30.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   eEbi： 定义详见 csp_gptb_emint_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

eEbi	typedef enum {
	B_EM_INT_EP0 = 0x1,
	B_EM_INT_EP1 = 0x1 << 1,
	B_EM_INT_EP2 = 0x1 << 2,
	B_EM_INT_EP3 = 0x1 << 3,
	// EM_INT_EP4 = 0x1 << 4,
	// EM_INT_EP5 = 0x1 << 5,
	// EM_INT_EP6 = 0x1 << 6,
	// EM_INT_EP7 = 0x1 << 7,
	B_EM_INT_CPUF= 0x1 << 8,
	B_EM_INT_MEMF= 0x1 << 9,
	B_EM_INT_EOMF= 0x1 << 10
	}csp_gptb_emint_e;	

csi_gptb_evtrg_enable

*csi_error_t csi_gptb_evtrg_enable(csp_gptb_t *ptGptbBase, uint8_t byCh, bool bEnable)*

18.3.31.1 功能描述

使能或禁止事件

18.3.31.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byCh： 0/1

bEnable ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_gptb_t.h中定义
byCh	0/1	
bEnable	ENABLE/DISABLE	

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_onetime_software_output

*csi_error_t csi_gptb_onetimesoftware_output(csp_gptb_t *ptGptbBase, uint16_t byCh, csp_gptb_action_e bEnable)*

18.3.32.1 功能描述

一次性软件波形控制设置

18.3.32.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byCh： GPTB_OSTSFA/ GPTB_OSTSFB

bEnable ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
byCh	GPTB_OSTSFA/ GPTB_OSTSFB	
bEnable	typedef enum { B_NA = 0, B_LO, B_HI, B_TG }csp_gptb_action_e;	在csp_ gptb.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_aqcsfload_config

*void csi_gptb_aqcsfload_config (csp_gptb_t *ptGptbBase, csp_gptb_aqosf_e bEnable)*

18.3.33.1 功能描述

AQCSF寄存器载入时机设置

18.3.33.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

bEnable 定义详见 csp_gptb_aqosf_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
bEnable	typedef enum { GPTB_AQCSF_NOW=0, GPTB_AQCSF_ZRO, GPTB_AQCSF_PRD, GPTB_AQCSF_ZRO_PRD }csp_gptb_aqosf_e;	在csp_gptb.h中定义

csi_gptb_continuous_software_waveform

*csi_error_t csi_gptb_continuous_software_waveform(csp_gptb_t *ptGptbBase, csi_gptb_channel_e byCh, csp_gptb_aqcsf_e bEnable)*

18.3.34.1 功能描述

持续性软件波形控制设置

18.3.34.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byCh 定义详见 csi_gptb_channel_e

bEnable 定义详见 csp_gptb_aqcsf_e

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
byCh	typedef enum{ GPTB_CHANNEL_1=1, GPTB_CHANNEL_2 }csi_gptb_channel_e;	在gptb.h中定义
bEnable	typedef enum { EM_AQCSF_NONE=0, EM_AQCSF_L, EM_AQCSF_H, EM_AQCSF_NONE1 }csp_gptb_aqcsf_e;	在csp_ gptb.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_int_enable

*void csi_gptb_int_enable(csp_gptb_t *ptGptbBase, csp_gptb_int_e eInt, bool bEnable)*

18.3.35.1 功能描述

中断使能控制器

18.3.35.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

eInt： 定义详见 csp_gptb_int_e

bEnable ENABLE/DISABLE

2. 返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定

eInt	typedef enum{
	GPTB_INT_TRGEV0 = 1,
	GPTB_INT_TRGEV1 = 2,
	// GPTB_INT_TRGEV2 = 4,
	// GPTB_INT_TRGEV3 = 8,
	GPTB_INT_CAPLD0 = 1 << 4,
	GPTB_INT_CAPLD1 = 1 << 5,
	GPTB_INT_CAPLD2 = 1 << 6,
	GPTB_INT_CAPLD3 = 1 << 7,
	GPTB_INT_CAU = 1 << 8,
	GPTB_INT_CAD = 1 << 9,
	GPTB_INT_CBU = 1 << 10,
	GPTB_INT_CBD = 1 << 11,
	GPTB_INT_PEND = 1 << 16
	}csp_gptb_int_e;
		义

bEnable	ENABLE/DISABLE	

csi_gptb_set_sync

*void csi_gptb_set_sync(csp_gptb_t *ptGptbBase, csi_gptb_trgin_e eTrgIn, csi_gptb_trgmode_e eTrgMode, csi_gptb_arearm_e eAutoRearm)*

18.3.36.1 功能描述

同步事件使能

18.3.36.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

eTrgIn： 定义详见 csi_gptb_trgin_e

eTrgMode： 定义详见 csi_gptb_trgmode_e

eAutoRearm： 定义详见 csi_gptb_arearm_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ gptb_t中说明	在csp_ gptb.h中定义

eTrgIn	typedef enum{
	GPTB_TRG_SYNCEN0 = 0, //start up or reset count
	GPTB_TRG_SYNCEN1, //reg updata
	GPTB_TRG_SYNCEN2, //capture
	GPTB_TRG_SYNCEN3, //
	GPTB_TRG_SYNCEN4, //
	GPTB_TRG_SYNCEN5, //
	}csi_gptb_trgin_e;
		在gptb.h中定义

eTrgMode	typedef enum{ GPTB_TRG_CONTINU = 0, GPTB_TRG_ONCE }csi_gptb_trgmode_e;	

eAutoRearm	typedef enum{
	GPTB_AUTO_REARM_DIS = 0, //disable auto rearm
	GPTB_AUTO_REARM_ZRO, //CNT = ZRO auto rearm
	GPTB_AUTO_REARM_PRD, //CNT = PRD auto rearm
	GPTB_AUTO_REARM_ZRO_PRD //CNT = PRD or PRD auto rearm
	}csi_gptb_arearm_e;	

csi_gptb_set_extsync_chnl

*csi_error_t csi_gptb_set_extsync_chnl(csp_gptb_t *ptGptbBase, csi_gptb_trgin_e eTrgIn, csi_gptb_syncrout_e byTrgChx)*

18.3.37.1 功能描述

同步事件用于事件触发输出

18.3.37.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   eTrgIn： 选择用于事件的同步源

   byTrgChx： 选择通道号

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

eTrgIn	typedef enum{
	GPTB_TRGIN_SYNCEN0 = 0, //start up or reset count
	GPTB_TRGIN_SYNCEN1, //reg updata
	GPTB_TRGIN_SYNCEN2, //capture
	GPTB_TRGIN_SYNCEN3, //
	GPTB_TRGIN_SYNCEN4, //
	GPTB_TRGIN_SYNCEN5, //
	// GPTB_TRGIN_SYNCEN6
	}csi_gptb_trgin_e;
		在gptb.h中定义

byTrgChx	0/1	

csi_gptb_set_sync_filter

*csi_error_t csi_gptb_set_sync_filter(csp_gptb_t *ptGptbBase, csi_gptb_filter_config_t *ptFilter)*

18.3.38.1 功能描述

同步事件滤波器控制

18.3.38.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

ptFilter： 定义详见 csi_gptb_filter_config_t

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

ptFilter	/// \struct csi_gptb_filter_config_t
	/// \brief gptb sync trigger filter parameter configuration, open to users
	typedef struct {
	uint8_t byFiltSrc; //filter input signal source
	uint8_t byWinInv; //window inversion
	uint8_t byWinAlign; //window alignment
	uint8_t byWinCross; //window cross
	uint16_t byWinOffset; //window offset
	uint16_t byWinWidth; //window width
	} csi_gptb_filter_config_t;
		在gptb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_rearm_sync

*void csi_gptb_rearm_sync(csp_gptb_t *ptGptbBase,csi_gptb_trgin_e eTrgin)*

18.3.39.1 功能描述

在单次模式下同步事件重新使能

18.3.39.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   eTrgin： 定义详见 csi_gptb_trgin_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义

eTrgin	typedef enum{
	GPTB_TRGIN_SYNCEN0 = 0, //start up or reset count
	GPTB_TRGIN_SYNCEN1, //reg updata
	GPTB_TRGIN_SYNCEN2, //capture
	GPTB_TRGIN_SYNCEN3, //
	GPTB_TRGIN_SYNCEN4, //
	GPTB_TRGIN_SYNCEN5, //
	// GPTB_TRGIN_SYNCEN6
	}csi_gptb_trgin_e;
		在gptb.h中定义

csi_gptb_set_evtrg

*csi_error_t csi_gptb_set_evtrg(csp_gptb_t *ptGptbBase, csi_gptb_trgout_e byTrgOut, csi_gptb_trgsrc_e eTrgSrc)*

18.3.40.1 功能描述

事件触发控制设置

18.3.40.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   eTrgOut： 定义详见 csi_gptb_trgout_e

   eTrgSrc 定义详见 csi_gptb_trgsrc_e

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
eTrgOut	typedef enum{ GPTB_TRGOUT0 = 0, GPTB_TRGOUT1, }csi_gptb_trgout_e;	

eTrgSrc	typedef enum {
	GPTB_TRGSRC_DIS = 0,
	GPTB_TRGSRC_ZRO,
	GPTB_TRGSRC_PRD,
	GPTB_TRGSRC_ZRO_PRD,
	GPTB_TRGSRC_CAU,
	GPTB_TRGSRC_CAD,
	GPTB_TRGSRC_CBU,
	GPTB_TRGSRC_CBD,
	// GPTB_TRGSRC_CCU,
	// GPTB_TRGSRC_CCD,
	// GPTB_TRGSRC_CDU,
	// GPTB_TRGSRC_CDD,
	GPTB_TRGSRC_EX =0x0c,
	GPTB_TRGSRC_EP0,
	GPTB_TRGSRC_EP1,
	GPTB_TRGSRC_EP2,
	}csi_gptb_trgsrc_e;
		在gptb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_set_evcntinit

*csi_error_t csi_gptb_set_evcntinit(csp_gptb_t *ptGptbBase, uint8_t byCntChx, uint8_t byCntVal, uint8_t byCntInitVal)*

18.3.41.1 功能描述

事件输出计数功能配置

18.3.41.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

byCntChx： 事件通道选择0~1

byCntVal 事件周期值

byCntInitVal 事件周期初始

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
byCntChx	typedef enum{ GPTB_TRG_OUT0 = 0, GPTB_TRG_OUT1, }csi_gptb_trgout_e;	在gptb.h中定义
byCntVal	事件周期值	
byCntInitVal	事件周期初始	
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_reglk_config

*csi_error_t csi_gptb_reglk_config(csp_gptb_t *ptGptbBase,csi_gptb_feglk_config_t *Global)*

18.3.42.1 功能描述

链接寄存功能设置

18.3.42.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

Global： 定义详见 csi_gptb_feglk_config_t

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
Global	typedef struct { uint8_t byPrdr; uint8_t byCmpa; uint8_t byCmpb; uint8_t byGld2; uint8_t byRssr; uint8_t byEmslclr; uint8_t byEmhlclr; uint8_t byEmicr; uint8_t byEmfrcr; uint8_t byAqosf; uint8_t byAqcsf; } csi_gptb_feglk_config_t;	在gptb.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_burst_enable

*csi_error_t csi_gptb_burst_enable(csp_gptb_t *ptGptbBase,uint8_t byCgsrc,uint8_t byCgflt, bool bEnable)*

18.3.43.1 功能描述

群脉冲模式配置

18.3.43.2 参数/返回值说明

1. 参数

   ptGptbBase： 定义详见 csp_gptb_t

   byCgsrc：群脉冲门控输入源选择（0：GPTA_CHA 1:GPTA_CHB）

   byCgflt：门控输入数字滤波控制(0~7)

   bEnable：ENABLE/DISABLE

2. 返回值

   CSI_OK： 设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
byCgsrc	8位参数，0：GPTA_CHA 1:GPTA_CHB	

byCgflt	8位参数，范围为0~7，详解如下：
	`typedef enum {`
	`GPTA_CGFLT_BP = 0,`
	`GPTA_CGFLT_2,`
	`GPTA_CGFLT_3,`
	`GPTA_CGFLT_4,`
	`GPTA_CGFLT_6,`
	`GPTA_CGFLT_8,`
	`GPTA_CGFLT_16,`
	`GPTA_CGFLT_32`
	`}csp_gpta_cnflt_e;`
		在csp_gptb.h中定义

bEnable	ENABLE/DISABLE	在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_timer_init

*csi_error_t csi_gptb_timer_init(csp_gptb_t *ptGptbBase, uint32_t wTimeOut)*

18.3.45.1 功能描述

定时功能初始化

18.3.45.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

wTimeOut：计数器溢出时间，即定时时间，单位us

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
wTimeOut	uint32_t 类型数值，单位：us	定时时间，单位us
csi_error_t	csi_error_t 中定义值	在common.h中定义

csi_gptb_count_mode

*void csi_gptb_count_mode(csp_gptb_t *ptGptbBase, csi_gptb_opmd_e eCntMode)*

18.3.46.1 功能描述

设置GPTB计数器工作模式

18.3.46.2 参数/返回值说明

1. 参数

ptGptaBase： 定义详见 csp_gptb_t

eCntMode：计数工作模式，枚举定义详见csi_gptb_opmd_e。

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gpta_t中说明	在csp_ gptb.h中定义
eCntMode	typedef enum{ GPTB_OP_CONT = 0, GPTB_OP_OT }csi_gptb_opmd_e;	计数工作模式有两种，连续计数模式、单次触发模式 默认为连续计数模式

csi_gptb_set_phsr

*void csi_gptb_set_phsr(csp_gptb_t *ptGptbBase, uint16_t hwPhsr,bool bEnable)*

18.3.47.1 功能描述

相位设置

18.3.47.2 参数/返回值说明

1. 参数

ptGptbBase： 定义详见 csp_gptb_t

hwPhsr：相位计数值设置

bEnable：ENABLE/DISABLE

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptGptbBase	csp_gptb_t中说明	在csp_ gptb.h中定义
hwPhsr	uint16_t 类型数值	相位计数值
bEnable	ENABLE/DISABLE	在common.h中定义

19 EPT

19.1 概述

增强型通用定时器（Enhanced Purpose Timer）作为MCU的关键外设，可以在各种功率控制应用中发挥关键控制作用。通过灵活的PWM输出，可以适用于各种复杂多变的应用，这些应用包括数字马达控制、开关电源控制、不间断电源控制、变频功率转换控制等。EPT内部包含一个16位的定时/计数模块，支持2种工作模式(捕捉模式和波形发生器模式)。

下图为EPT框图的一部分，用于配合api函数说明。

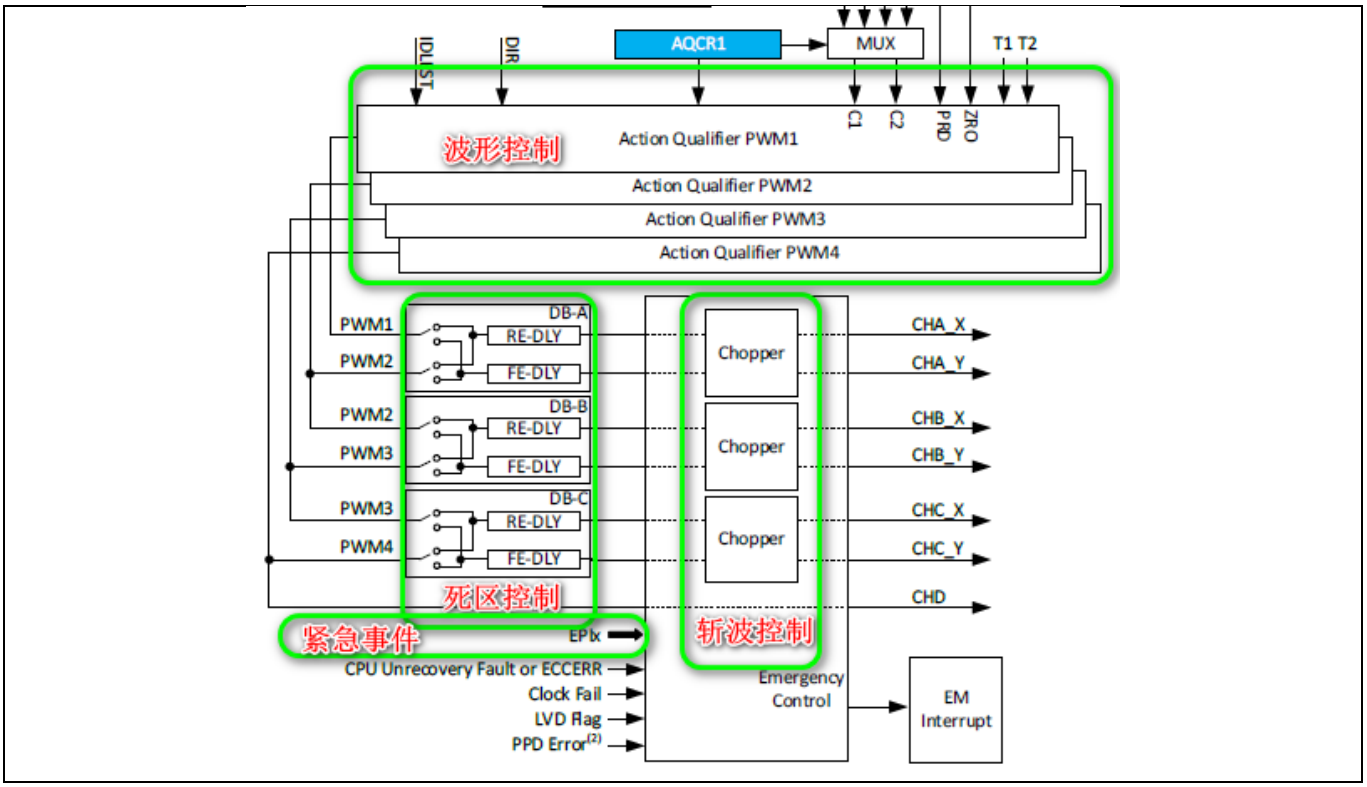


Figure 17-1 波形控制

19.2 API列表

Table 17-1 EPT CSI接口函数

API	说明	函数位置
ept_initen_irqhandler	定时器中断回调函数	
csi_ept_capture_init	定时器捕获模式参数设置	

csi_ept_wave_init	定时器波形模式参数基础设置（计数模式，周期，占空比等）	
csi_ept_channel_config	设置通道PWM1、PWM2、PWM3、PWM4的波形	
csi_ept_channel_cmpload_config	定时器CMPA.CMPB.CMPC.CMPD载入时机设置	
csi_ept_channel_aqload_config	通道PWM1、PWM2、PWM3、PWM4的波形载入时机设置	
csi_ept_dbload_config	死区配置载入控制	
csi_ept_dz_config	死区功能基本配置	
csi_ept_channelmode_config	死区通道配置（CH_A, CH_B, CH_C）	
csi_ept_chopper_config	斩波输出控制	
csi_ept_chopper_enable	通道斩波输出使能	
csi_ept_emergency_config	紧急状态输入基本参数设置	
csi_ept_emergency_pinout	设置紧急事件到来时，CHA/B/C/D_X/Y处的状态。	
csi_ept_ gload_config	全局载入控制配置	
csi_ept_gldcfg	全局载入对象使能或禁止	
csi_ept_ gload_sw	软件产生一次全局载入	
csi_ept_ gload_rearm	单次模式下全局载入重使能	
csi_ept_start	定时器开始计数	
csi_ept_stop	定时器停止计数	
csi_ept_set_start_mode	定时器开始位模式选择（普通开始/开始的同时触发同步事件0）	
csi_ept_set_os_mode	定时器模式选择（单次模式或连续模式）	
csi_ept_set_stop_st	波形输出被停止时，输出端口的缺省状态	
csi_ept_get_prdr	读取定时器周期寄存器	
csi_ept_change_ch_duty	改变比较值寄存器	
csi_ept_force_em	紧急状态软件触发	
csi_ept_get_hdlck_st	获取紧急硬锁止状态寄存器	
csi_ept_clr_hdlck	清除紧急硬锁止状态寄存器	
csi_ept_get_sftlck_st	获取紧急软锁止状态寄存器	
csp_ept_clr_sftlck	紧急软锁止状态清除寄存器	
csi_ept_debug_enable	使能或禁止debug模式	
csi_ept_emergency_int_enable	紧急状态中断使能或禁止	
csi_ept_evtrg_enable	使能或禁止事件	
csi_ept_onetime_software_output	一次性软件波形控制设置	
csi_ept_aqcsfload_config	AQCSF寄存器载入时机设置	
csi_ept_continuous_software_output	持续性软件波形控制设置	
csi_ept_int_enable	中断使能控制器	
csi_ept_set_sync	同步事件使能	
csi_ept_set_sync2evtrg	同步事件用于事件触发输出	
csi_ept_set_sync_filter	同步事件滤波器控制	

csi_ept_rearm_sync	在单次模式下同步事件重新使能	
csi_ept_set_evtrg	事件触发控制设置	
csi_ept_set_evcntinit	事件输出计数功能配置	
csi_ept_reglk_config	链接寄存功能设置	
csi_ept_burst_enable	群脉冲模式配置	

19.3 API详细说明

19.3.1 ept_initen_irqhandler

*__attribute__((weak)) void ept_initen_irqhandler(csp_ept_t *ptEptBase)*

19.3.1.1 功能描述

定时器中断回调函数。

19.3.1.2 参数/返回值说明

1. 参数

ptEptBase： 每个功能模块都有对应的ptEptBase，枚举定义详见csp_ept_t

2. 返回值

无

3. 参数说明表

参数	说明	概述及其变量位置
ptEptBase	csp_ept_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_ ept.h中定义

19.3.2 csi_ept_config_init

*csi_error_t csi_ept_config_init(csp_ept_t *ptEptBase, csi_ept_config_t *pteptPwmCfg)*

19.3.2.1 功能描述

定时器常规寄存器定义。

19.3.2.2 参数/返回值说明

4. 参数

ptEptBase：每个功能模块都有对应的ptEptBase，枚举定义详见csp_ept_t

pteptPwmCfg：每个功能模块都有常规参数定义，枚举定义详见csi_ept_config_t

5. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

6. 参数说明表

参数	说明	概述及其变量位置
ptEptBase	csp_ept_t结构体，用于进行寄存器操作。	定时器模块所有寄存器的定义 在csp_ept.h中定义

pteptPwmCfg	typedef struct csi_ept_config csi_ept_config_t;
	struct csi_ept_config {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_ept_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint8_t byCaptureCapLden;
	uint8_t byCaptureRearm;
	uint8_t byCaptureCapmd;
	uint8_t byCaptureStopWrap;
	uint8_t byCaptureLdaret;
	uint8_t byCaptureLdbret;
	uint8_t byCaptureLdcret;
	uint8_t byCaptureLddret;
	uint8_t byBurst;
	uint8_t byCgsrc;
	uint8_t byCgflt;
	uint32_t wFreq; //TIMER PWM OUTPUT frequency
	uint32_t wInt;
	};
		对定时器工作参数的定义
		在gpta.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.3 csi_ept_capture_init

*csi_error_t csi_ept_capture_init(csp_ept_t *ptEptBase, csi_ept_captureconfig_t *pteptPwmCfg)*

19.3.3.1 功能描述

捕获模式下定时器基本参数设定

19.3.3.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见csp_ept_t

pteptPwmCfg： 定义详见csi_ept_captureconfig_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

pteptPwmCfg	typedef struct csi_ept_captureconfig csi_ept_captureconfig_t;
	struct csi_ept_captureconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_ept_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint8_t byCaptureCapLden;
	uint8_t byCaptureRearm;
	uint8_t byCaptureCapmd;
	uint8_t byCaptureStopWrap;
	uint8_t byCaptureLdaret;
	uint8_t byCaptureLdbret;
	uint8_t byCaptureLdcret;
	uint8_t byCaptureLddret;
	uint32_t wInt;
	};
		在ept.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.4 csi_ept_wave_init

*csi_error_t csi_ept_wave_init(csp_ept_t *ptEptBase, csi_ept_pwmconfig_t *pteptPwmCfg)*

19.3.4.1 功能描述

定时器波形模式参数基础设置（计数模式，周期，占空比等）

19.3.4.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见csp_ept_t

pteptPwmCfg： 定义详见csi_ept_pwmconfig_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

pteptPwmCfg	typedef struct csi_ept_pwmconfig csi_ept_pwmconfig_t;
	struct csi_ept_pwmconfig {
	uint8_t byWorkmod; //Count or capture
	uint8_t byCountingMode; //csi_ept_cntmd_e
	uint8_t byOneshotMode; //Single or continuous
	uint8_t byStartSrc ;
	uint8_t byPscld;
	uint8_t byDutyCycle; //TIMER PWM OUTPUT duty cycle
	uint32_t wFreq; //TIMER PWM OUTPUT frequency
	uint32_t wInt;
	};
		在ept.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.5 csi_ept_channel_config

*csi_error_t csi_ept_channel_config(csp_ept_t *ptEptBase, csi_ept_pwmchannel_config_t *tPwmCfg, csi_ept_channel_e channel)*

19.3.5.1 功能描述

设置通道PWM1、PWM2、PWM3、PWM4的波形

19.3.5.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见csp_ept_t

tPwmCfg： 定义详见csi_ept_pwmchannel_config_t

channel： 定义详见 csi_ept_channel_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

tPwmCfg	typedef struct csi_ept_pwmchannel_config csi_ept_pwmchannel_config_t;
	struct csi_ept_pwmchannel_config {
	uint8_t byActionZro; //
	uint8_t byActionPrd; //
	uint8_t byActionCau; //
	uint8_t byActionCad; //
	uint8_t byActionCbu; //
	uint8_t byActionCbd; //
	uint8_t byActionT1u; //
	uint8_t byActionT1d; //
	uint8_t byActionT2u; //
	uint8_t byActionT2d; //
	uint8_t byChoiceCasel;
	uint8_t byChoiceCbsel;
	};	

channel	typedef enum { EPT_CHANNEL_A=1, EPT_CHANNEL_B, EPT_CHANNEL_C, EPT_CHANNEL_D }csi_ept_channel_e;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.6 csi_ept_channel_cmpload_config

*csi_error_t csi_ept_channel_cmpload_config(csp_ept_t *ptEptBase, csp_ept_cmpdata_ldmd_e tld, csp_ept_ldamd_e tldamd ,csi_ept_camp_e channel)*

19.3.5.3 功能描述

定时器CMPA.CMPB.CMPC.CMPD载入时机设置

19.3.5.4 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

tld： 定义详见 csp_ept_cmpdata_ldmd_e

tldamd： 定义详见 csp_ept_ldamd_e

channel： 定义详见 csi_ept_camp_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
tld	typedef enum { EPT_CMPLD_SHDW = 0, EPT_CMPLD_IMM }csp_ept_cmpdata_ldmd_e;	在csp_ept.h中定义
tldamd	typedef enum{ EPT_LDCMP_NEVER=0, EPT_LDCMP_ZRO, EPT_LDCMP_PRD, EPT_LDCMP_LD_SYNC=4, }csp_ept_ldamd_e;	
channel	typedef enum{ EPT_CAMPA=1, EPT_CAMPB, EPT_CAMPC, EPT_CAMPD }csi_ept_camp_e;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.7 csi_ept_channel_aqload_config

*csi_error_t csi_ept_channel_aqload_config(csp_ept_t *ptEptBase, csp_ept_ld_e tld, csp_ept_ldamd_e tldamd ,csi_ept_channel_e channel)*

19.3.7.1 功能描述

通道PWM1、PWM2、PWM3、PWM4的波形载入时机设置

19.3.7.2 参数/返回值说明

4. 参数

ptEptBase： 定义详见csp_ept_t

tld： 定义详见csp_ept_ld_e

tldamd： 定义详见csp_ept_ldamd_e

channel： 定义详见csi_ept_channel_e

5. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

6. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
tld	typedef enum{ EPT_LD_IMM = 0, EPT_LD_SHDW }csp_ept_ld_e;	在csp_ ept.h中定义
tldamd	typedef enum{ EPT_LDCMP_NEVER=0, EPT_LDCMP_ZRO, EPT_LDCMP_PRD, EPT_LDCMP_LD_SYNC=4, }csp_ept_ldamd_e;	
channel	typedef enum{ EPT_CHANNEL_1=1, EPT_CHANNEL_2, EPT_CHANNEL_3, EPT_CHANNEL_4 }csi_ept_channel_e;	在ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.8 csi_ept_dbload_config

*csi_error_t csi_ept_dbload_config(csp_ept_t *ptEptBase, csi_ept_dbdldr_e byVal,csp_ept_shdw_e byWod,csp_ept_lddb_e byWod2)*

19.3.8.1 功能描述

死区配置载入控制

19.3.8.2 参数/返回值说明

4. 参数

ptEptBase： 定义详见 csp_ept_t

byVal： 定义详见 csi_ept_dbdldr_e

byWod： 定义详见 csp_ept_shdw_e

byWod2： 定义详见 csp_ept_lddb_e

5. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

6. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byVal	typedef enum{ DBCR =0, DBDTR, DBDTF, DCKPSC, }csi_ept_dbdldr_e;	
byWod	typedef enum{ EPT_SHDW_IMMEDIATE =0, EPT_SHDW_SHADOW }csp_ept_shdw_e;	在csp_ept.h中定义
byWod2	typedef enum{ EPT_LD_NEVER = 0, EPT_LD_ZRO, EPT_LD_PRD, EPT_LD_ZRO_PRD }csp_ept_lddb_e;	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.9 csi_ept_dz_config

*csi_error_t csi_ept_dz_config(csp_ept_t *ptEptBase, csi_ept_deadzone_config_t *tCfg)*

19.3.9.1 功能描述

死区功能基本配置

19.3.9.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

tCfg： 定义详见 csi_ept_deadzone_config_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

tCfg	typedef struct csi_ept_deadzone_config csi_ept_deadzone_config_t;
	struct csi_ept_deadzone_config
	{
	uint8_t byChxOuselS1S0 ; //
	uint8_t byChxPolarityS3S2 ; //
	uint8_t byChxInselS5S4 ; //
	uint8_t byChxOutSwapS8S7 ; //
	uint8_t byDcksel;
	uint8_t byChaDedb;
	uint8_t byChbDedb;
	uint8_t byChcDedb;
	uint16_t hwDpsc; //
	uint16_t hwRisingEdgereGister ; //
	uint16_t hwFallingEdgereGister; //
	};	

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.10 csi_ept_channelmode_config

*csi_error_t csi_ept_channelmode_config(csp_ept_t *ptEptBase,csi_ept_deadzone_config_t *tCfg,csi_ept_channel_e byCh)*

19.3.10.1 功能描述

死区通道配置（CHANNEL_A, CHANNEL_B, CHANNEL_C）

19.3.10.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

tCfg： 定义详见 csi_ept_deadzone_config_t

byCh： 定义详见csi_ept_channel_e

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

tCfg	typedef struct csi_ept_deadzone_config csi_ept_deadzone_config_t;
	struct csi_ept_deadzone_config
	{
	uint8_t byChxOuselS1S0 ; //
	uint8_t byChxPolarityS3S2 ; //
	uint8_t byChxInselS5S4 ; //
	uint8_t byChxOutSwapS8S7 ; //
	uint8_t byDcksel;
	uint8_t byChaDedb;
	uint8_t byChbDedb;
	uint8_t byChcDedb;
	uint16_t hwDpsc; //
	uint16_t hwRisingEdgereGister ; //
	uint16_t hwFallingEdgereGister; //
	};
		在ept.h中定义

byCh	typedef enum{ EPT_CHANNEL_A=1, EPT_CHANNEL_B, EPT_CHANNEL_C, EPT_CHANNEL_D }csi_ept_channel_e;	在ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.11 csi_ept_chopper_config

*void csi_ept_chopper_config(csp_ept_t *ptEptBase, csi_ept_Chopper_config_t *tCfg)*

19.3.11.1 功能描述

斩波输出控制

19.3.11.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   tCfg： 定义详见 csi_ept_Chopper_config_t

2. 返回值：无

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

tCfg	typedef struct csi_ept_Chopper_config csi_ept_Chopper_config_t;
	struct csi_ept_Chopper_config
	{
	uint8_t byOspwth;
	uint8_t byCdiv;
	uint8_t byCduty;
	uint8_t byCasel;
	uint8_t chen;
	};
		在ept.h中定义

19.3.12 csi_ept_chopper_enable

*csi_error_t csi_ept_chopper_enable(csp_ept_t *ptEptBase, csi_ept_chx_e byCh, bool bEn)*

19.3.12.1 功能描述

通道斩波输出使能

19.3.12.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   byCh： 定义详见 csi_ept_chx_e

   bEn： ENABLE/DISABLE

2. 返回值

   CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
tCfg	typedef enum{ EPTCHAX = 0x1, EPTCHAY, EPTCHBX, EPTCHBY, EPTCHCX, EPTCHCY }csi_ept_chx_e;	在ept.h中定义
bEn	ENABLE/DISABLE	在csp_common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.13 csi_ept_emergency_config

*csi_error_t csi_ept_emergency_config (csp_ept_t *ptEptBase, csi_ept_emergency_config_t *tCfg)*

19.3.13.1 功能描述

紧急状态输入基本参数设置

19.3.13.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

tCfg： 定义详见 csi_ept_emergency_config_t

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

tCfg	typedef struct csi_ept_emergency_config csi_ept_emergency_config_t;
	struct csi_ept_emergency_config
	{
	uint8_t byEpxInt ;
	uint8_t byPolEbix;
	uint8_t byEpx;
	uint8_t byEpxLckmd;
	uint8_t byFltpace0;
	uint8_t byFltpace1;
	uint8_t byOrl0;
	uint8_t byOrl1;
	};
		在ept.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.14 csi_ept_emergency_pinout

*csi_error_t csi_ept_emergency_pinout(csp_ept_t *ptEptBase,csi_ept_osrchx_e byCh ,csp_ept_emout_e byCh2)*

19.3.14.1 功能描述

设置紧急事件到来时，CHA/B/C/D_X/Y处的状态

19.3.14.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byCh： 定义详见 csi_ept_osrchx_e

byCh2： 定义详见 csp_et_emout_e

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

byCh	typedef enum{ EMCOAX =0, EMCOBX, EMCOCX, EMCOD, EMCOAY, EMCOBY, EMCOCY }csi_ept_osrchx_e;	在ept.h中定义
byCh2	typedef enum { EM_OUT_HZ, EM_OUT_H, EM_OUT_L, EM_OUT_NONE }csp_ept_emout_e;	在ept.h中定义
csi_error_t	csi_error_t 中定义值	common.h

19.3.15 csi_ept_gload_config

*void csi_ept_ gload _config(csp_ept_t *ptEptBase,csi_ept_Global_load_control_config_t *Global)*

19.3.15.1 功能描述

全局载入控制配置

19.3.15.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   Global： 定义详见 csi_ept_Global_load_control_config_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

Global	typedef struct csi_ept_Global_load_control_config csi_ept_Global_load_control_config_t;
	struct csi_ept_Global_load_control_config{
	bool bGlden;
	bool bOstmd;
	uint8_t bGldprd;
	uint8_t byGldmd;
	};
		在ept.h中定义

19.3.16 csi_ept_gldcfg

*csi_error_t csi_ept_gldcfg(csp_ept_t *ptEptBase ,csi_ept_Global_load_gldcfg Glo,bool bENABLE)*

19.3.16.1 功能描述

全局载入对象使能或禁止

19.3.16.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

Glo： 定义详见 csi_ept_Global_load_gldcfg

bENABLE: ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
Global	typedef enum { byprdr_B=0, bycmpa_B, bycmpb_B, byaqcra_B, byaqcrb_B, byaqcsf_B, }csi_gptb_Global_load_gldcfg;	在ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.17 csi_ept_gload_sw

*void csi_ept_gload_sw(csp_ept_t *ptEptBase)*

19.3.17.1 功能描述

软件产生一次全局载入

19.3.17.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

19.3.18 csi_ept_gload_rearm

*void csi_ept_gload_rearm(csp_ept_t *ptEptBase)*

19.3.18.1 功能描述

单次模式下全局载入重使能

19.3.18.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

19.3.19 csi_ept_start

*void csi_ept_start(csp_ept_t *pteptBase)*

19.3.19.1 功能描述

定时器开始计数

19.3.19.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

2. 返回值：无

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

19.3.20 csi_ept_stop

*void csi_ept_stop(csp_ept_t * ptEptBase)*

19.3.20.1 功能描述

定时器停止计数

19.3.20.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置

ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

19.3.21 csi_ept_set_start_mode

*void csi_ept_set_start_mode(csp_ept_t * ptEptBase, csi_ept_stmd_e eMode)*

19.3.21.1 功能描述

定时器开始位模式选择（普通开始/开始的同时触发同步事件0）

19.3.21.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   eMode： 定义详见 csi_ept_stmd_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eMode	`typedef enum {`
	`  EPT_SW = 0,`
	`  EPT_SYNC`
	`}csi_ept_stmd_e;`	

19.3.22 csi_ept_set_os_mode

*void csi_ept_set_os_mode(csp_ept_t * ptEptBase, csi_ept_opmd_e eMode)*

19.3.22.1 功能描述

定时器模式选择（单次模式或连续模式）

19.3.22.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   eMode： 定义详见 csi_ept_opmd_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eMode	typedef enum { EPT_OP_CONT = 0, EPT_OP_OT, }csi_ept_opmd_e;	在ept.h中定义

19.3.23 csi_ept_set_stop_st

*void csi_ept_set_stop_st(csp_ept_t * ptEptBase, csp_ept_stpst_e eSt)*

19.3.23.1 功能描述

波形输出被停止时，输出端口的缺省状态

19.3.23.2 参数/返回值说明

1. 参数

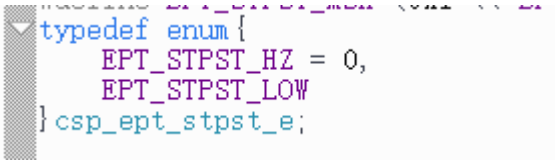
   ptEptBase： 定义详见 csp_ept_t

   eSt： 定义详见 csp_ept_stpst_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eSt		

19.3.24 csi_ept_get_prdr

*uint16_t csi_ept_get_prdr(csp_ept_t * ptEptBase)*

19.3.24.1 功能描述

读取定时器周期寄存器

19.3.24.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

2. 返回值

   返回周期值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
uint16_t	返回周期值	

19.3.25 csi_ept_change_ch_duty

*csi_error_t csi_ept_change_ch_duty(csp_ept_t *ptEptBase, csi_ept_chtype_e eCh, uint32_t wActiveTime)*

19.3.25.1 功能描述

改变比较值寄存器

19.3.25.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eCh： 定义详见 csi_ept_chtype_e

wActiveTime： 周期值的X%

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eCh	typedef enum { EPT_CH_A = 0, EPT_CH_B, EPT_CH_C, EPT_CH_D }csi_ept_chtype_e;	在ept.h中定义
wActiveTime	周期值的X%	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.26 csi_ept_force_em

*void csi_ept_force_em(csp_ept_t *ptEptBase, csp_ept_ep_e byEbi)*

19.3.26.1 功能描述

紧急状态软件触发

19.3.26.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byEbi： 定义详见 csp_ept_ep_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byEbi	typedef enum{ EP0 = 0, EP1, EP2, EP3, EP4, EP5, EP6, EP7, }csp_ept_ep_e;	在csp_ept.h中定义

19.3.27 csi_ept_get_hdlck_st

*uint8_t csi_ept_get_hdlck_st(csp_ept_t *ptEptBase)*

19.3.27.1 功能描述

获取紧急硬锁止状态寄存器

19.3.27.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

2. 返回值

返回紧急硬锁止状态寄存器值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置

ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
uint8_t	返回紧急硬锁止状态寄存器值	

19.3.28 csi_ept_clr_hdlck

*void csi_ept_clr_hdlck(csp_ept_t *ptEptBase, csp_ept_ep_e eEbi)*

19.3.28.1 功能描述

清除紧急硬锁止状态寄存器

19.3.28.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   eEbi 定义详见 csp_ept_ep_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eEbi	typedef enum { EP0 = 0, EP1, EP2, EP3, EP4, EP5, EP6, EP7, } csp_ept_ep_e;	在csp_ept.h中定义

19.3.29 csi_ept_get_sftlck_st

*uint8_t csi_ept_get_sftlck_st(csp_ept_t *ptEptBase)*

19.3.29.1 功能描述

获取紧急软锁止状态寄存器

19.3.29.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

2. 返回值

返回紧急软锁止状态值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
uint8_t	返回紧急软锁止状态值	

19.3.30 csp_ept_clr_sftlck

*void csp_ept_clr_sftlck(csp_ept_t *ptEptBase, csp_ept_ep_e eEpi)*

19.3.30.1 功能描述

紧急软锁止状态清除寄存器

19.3.30.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eEpi： 定义详见 csp_ept_ep_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置

ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eEpi	typedef enum { EP0 = 0, EP1, EP2, EP3, EP4, EP5, EP6, EP7, } csp_ept_ep_e;	在csp_ept.h中定义

19.3.31 csi_ept_debug_enable

*void csi_ept_debug_enable(csp_ept_t *ptEptBase, bool bEnable)*

19.3.31.1 功能描述

使能或禁止debug模式

19.3.31.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

bEnable： ENABLE/DISABLE

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
bEnable	ENABLE/DISABLE	在common.h中定义

19.3.32 csi_ept_emergency_int_enable

*void csi_ept_emergency_int_enable(csp_ept_t *ptEptBase, csp_ept_emint_e eEbi)*

19.3.32.1 功能描述

紧急状态中断使能或禁止

19.3.32.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eEbi： 定义详见 csp_ept_emint_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
eEbi	typedef enum { EPT_INT_EP0 = 0x1, EPT_INT_EP1 = 0x1 << 1, EPT_INT_EP2 = 0x1 << 2, EPT_INT_EP3 = 0x1 << 3, EPT_INT_EP4 = 0x1 << 4, EPT_INT_EP5 = 0x1 << 5, EPT_INT_EP6 = 0x1 << 6, EPT_INT_EP7 = 0x1 << 7, EPT_INT_CPUF= 0x1 << 8, EPT_INT_MEMF= 0x1 << 9, EPT_INT_EOMF= 0x1 << 10 }csp_ept_emint_e;	在csp_ept.h中定义

19.3.33 csi_ept_evtrg_enable

*csi_error_t csi_ept_evtrg_enable(csp_ept_t *ptEptBase, uint8_t byCh, bool bEnable)*

19.3.33.1 功能描述

使能或禁止事件

19.3.33.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byCh： 0/1/2/3

bEnable ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byCh	0/1/2/3	
bEnable	ENABLE/DISABLE	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.34 csi_ept_onetime_software_output

*csi_error_t csi_ept_onetime_software_output(csp_ept_t *ptEptBase, uint16_t byCh, csp_ept_action_e bEnable)*

19.3.34.1 功能描述

一次性软件波形控制设置

19.3.34.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byCh： EPT_OSTSFA/EPT_OSTSFB/EPT_OSTSFC/EPT_OSTSFD

bEnable ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败。

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byCh	EPT_OSTSFA/EPT_OSTSFB/EPT_OSTSFC/EPT_OSTSFD	
bEnable	typedef enum { NA = 0, LO, HI, TG }csp_ept_action_e;	在csp_ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.35 csi_ept_aqcsfload_config

*void csi_ept_aqcsfload_config (csp_ept_t *ptEptBase, csp_ept_aqosf_e bEnable)*

19.3.35.1 功能描述

AQCSF寄存器载入时机设置

19.3.35.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

bEnable 定义详见 csp_ept_aqosf_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

bEnable	typedef enum { EPT_AQCSF_NOW=0, EPT_AQCSF_ZRO, EPT_AQCSF_PRD, EPT_AQCSF_ZRO_PRD }csp_ept_aqosf_e;	在csp_ept.h中定义

19.3.36 csi_ept_continuous_software_output

*csi_error_t csi_ept_continuous_software_output(csp_ept_t *ptEptBase, csi_ept_channel_e byCh, csp_ept_aqcsf_e bEnable)*

19.3.36.1 功能描述

持续性软件波形控制设置

19.3.36.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byCh 定义详见 csi_ept_channel_e

bEnable 定义详见 csp_ept_aqcsf_e

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byCh	typedef enum{ EPT_CHANNEL_A=1, EPT_CHANNEL_B, EPT_CHANNEL_C, EPT_CHANNEL_D }csi_ept_channel_e;	在ept.h中定义

bEnable	typedef enum { EM_AQCSF_NONE=0, EM_AQCSF_L, EM_AQCSF_H, EM_AQCSF_NONE1 }csp_ept_aqcsf_e;	在csp_ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.37 csi_ept_int_enable

*csi_error_t csi_ept_int_enable(csp_ept_t *ptEptBase, csp_ept_int_e eInt, bool bEnable)*

19.3.37.1 功能描述

中断使能控制器

19.3.37.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eInt： 定义详见 csp_ept_int_e

bEnable ENABLE/DISABLE

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eInt	typedef enum{ EPT_INT_TRGEV0 = 0x1, EPT_INT_TRGEV1 = 0x2, EPT_INT_TRGEV2 = 0x4, EPT_INT_TRGEV3 = 0x8, EPT_INT_CAPLD0 = 0x1 << 4, EPT_INT_CAPLD1 = 0x1 << 5, EPT_INT_CAPLD2 = 0x1 << 6, EPT_INT_CAPLD3 = 0x1 << 7, EPT_INT_CAU = 0x1 << 8, EPT_INT_CAD = 0x1 << 9, EPT_INT_CBU = 0x1 << 10, EPT_INT_CBD = 0x1 << 11, EPT_INT_CCU = 0x1 << 12, EPT_INT_CCD = 0x1 << 13, EPT_INT_CDU = 0x1 << 14, EPT_INT_CDD = 0x1 << 15, EPT_INT_PEND = 0x1 << 16 }csp_ept_int_e;	在csp_ept.h中定义
bEnable	ENABLE/DISABLE	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.38 csi_ept_set_sync

*void csi_ept_set_sync(csp_ept_t *ptEptBase, csi_ept_trgin_e eTrgIn, csi_ept_trgmode_e eTrgMode, csi_ept_arearm_e eAutoRearm)*

19.3.38.1 功能描述

同步事件使能

19.3.38.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eTrgIn： 定义详见 csi_ept_trgin_e

eTrgMode： 定义详见 csi_ept_trgmode_e

eAutoRearm： 定义详见 csi_ept_arearm_e

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eTrgIn	typedef enum{
	EPT_TRG_SYNCEN0 = 0, //start up or reset count
	EPT_TRG_SYNCEN1, //reg updata
	EPT_TRG_SYNCEN2, //capture
	EPT_TRG_SYNCEN3, //count inc or dec
	EPT_TRG_SYNCEN4, //change output status(pwm)
	EPT_TRG_SYNCEN5 //change output status(pwm)
	}csi_ept_trgin_e;
		在ept.h中定义

eTrgMode	typedef enum{
	EPT_TRG_CONTINU = 0, //EPT continuous trigger mode
	EPT_TRG_ONCE //EPT once trigger mode
	}csi_ept_trgmode_e;	

eAutoRearm	typedef enum{
	EPT_AUTO_REARM_DIS = 0, //disable auto rearm
	EPT_AUTO_REARM_ZRO, //CNT = ZRO auto rearm
	EPT_AUTO_REARM_PRD, //CNT = PRD auto rearm
	EPT_AUTO_REARM_ZRO_PRD //CNT = PRD or PRD auto rearm
	}csi_ept_arearm_e;	

19.3.39 csi_ept_set_sync2evtrg

*csi_error_t csi_ept_set_sync2evtrg(csp_ept_t *ptEptBase, csi_ept_trgin_e eTrgIn, uint8_t byTrgChx)*

19.3.39.1 功能描述

同步事件用于事件触发输出

19.3.39.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eTrgIn： 选择用于事件的同步源

byTrgChx： 选择通道号

2. 返回值

无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eTrgIn	typedef enum{
	EPT_TRGIN_SYNCEN0 = 0, //start up or reset count
	EPT_TRGIN_SYNCEN1, //reg updata
	EPT_TRGIN_SYNCEN2, //capture
	EPT_TRGIN_SYNCEN3, //count inc or dec
	EPT_TRGIN_SYNCEN4, //change output status(pwm)
	EPT_TRGIN_SYNCEN5 //change output status(pwm)
	}csi_ept_trgin_e;
		在ept.h中定义

byTrgChx	0/1	

19.3.40 csi_ept_set_sync_filter

*csi_error_t csi_ept_set_sync_filter(csp_ept_t *ptEptBase, csi_ept_filter_config_t *ptFilter)*

19.3.40.1 功能描述

同步事件滤波器控制

19.3.40.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

ptFilter： 定义详见 csi_ept_filter_config_t

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

ptFilter	typedef struct {
	uint8_t byFiltSrc; //filter input signal source
	uint8_t byWinInv; //window inversion
	uint8_t byWinAlign; //window alignment
	uint8_t byWinCross; //window cross
	uint16_t byWinOffset; //window offset
	uint16_t byWinWidth; //window width
	} csi_ept_filter_config_t;
		在ept.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.41 csi_ept_rearm_sync

*void csi_ept_rearm_sync(csp_ept_t *ptEptBase,csi_ept_trgin_e eTrgin)*

19.3.41.1 功能描述

在单次模式下同步事件重新使能

19.3.41.2 参数/返回值说明

1. 参数

   ptEptBase： 定义详见 csp_ept_t

   eTrgin： 定义详见 csi_ept_trgin_e

2. 返回值

   无返回值

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eTrgin	typedef enum{
	EPT_TRGIN_SYNCEN0 = 0, //start up or reset count
	EPT_TRGIN_SYNCEN1, //reg updata
	EPT_TRGIN_SYNCEN2, //capture
	EPT_TRGIN_SYNCEN3, //count inc or dec
	EPT_TRGIN_SYNCEN4, //change output status(pwm)
	EPT_TRGIN_SYNCEN5 //change output status(pwm)
	}csi_ept_trgin_e;
		在ept.h中定义

19.3.42 csi_ept_set_evtrg

*csi_error_t csi_ept_set_evtrg(csp_ept_t *ptEptBase, csi_ept_trgout_e eTrgOut, csi_ept_trgsrc_e eTrgSrc)*

19.3.42.1 功能描述

事件触发控制设置

19.3.42.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

eTrgOut： 定义详见 csi_ept_trgout_e

eTrgSrc 定义详见 csi_ept_trgsrc_e

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义

eTrgOut	typedef enum{
	EPT_TRGOUT0 = 0, //trigger out0
	EPT_TRGOUT1, //trigger out1
	EPT_TRGOUT2, //trigger out2
	EPT_TRGOUT3 //trigger out3
	}csi_ept_trgout_e;	

eTrgSrc	typedef enum { EPT_TRGSRC_DIS = 0, EPT_TRGSRC_ZRO, EPT_TRGSRC_PRD, EPT_TRGSRC_ZRO_PRD, EPT_TRGSRC_CAU, EPT_TRGSRC_CAD, EPT_TRGSRC_CBU, EPT_TRGSRC_CBD, EPT_TRGSRC_CCU, EPT_TRGSRC_CCD, EPT_TRGSRC_CDU, EPT_TRGSRC_CDD, EPT_TRGSRC_EX, EPT_TRGSRC_PE0, EPT_TRGSRC_PE1, EPT_TRGSRC_PE2, EPT_TRGSRC_PEND }csi_ept_trgsrc_e;	在ept.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.43 csi_ept_set_evcntinit

*csi_error_t csi_ept_set_evcntinit(csp_ept_t *ptEptBase, uint8_t byCntChx, uint8_t byCntVal, uint8_t byCntInitVal)*

19.3.43.1 功能描述

事件输出计数功能配置

19.3.43.2 参数/返回值说明

1. 参数

ptEptBase： 定义详见 csp_ept_t

byCntChx： 事件通道选择0~3

byCntVal 事件周期值

byCntInitVal 事件周期初始

2. 返回值

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其变量位置
ptEptBase	csp_ept_t中说明	在csp_ept.h中定义
byCntChx	EPT_TRG_OUT0~ EPT_TRG_OUT3	
byCntVal	事件周期值	
byCntInitVal	事件周期初始	
csi_error_t	csi_error_t 中定义值	在common.h中定义

19.3.44 csi_ept_reglk_config

*void csi_ept_reglk_config(csp_ept_t *pteptBase,csi_ept_feglk_config_t *Global)*

19.3.44.1 功能描述

链接寄存功能设置

19.3.44.2 参数/返回值说明

1. 参数

pteptBase： 定义详见 csp_gptb_t

Global： 定义详见 csi_gptb_feglk_config_t

2. 返回值：无

3. 参数/返回值说明表

参数	说明	概述及其变量位置
pteptBase	csp_gptb_t中说明	在csp_ gptb.h中定义
Global	typedef struct { uint8_t byPrdr; uint8_t byCmpa; uint8_t byCmpb; uint8_t byGld2; uint8_t byRssr; uint8_t byEmslclr; uint8_t byEmhlclr; uint8_t byEmicr; uint8_t byEmfrcr; uint8_t byAqosf; uint8_t byAqcsf; } csi_gptb_feglk_config_t;	在gptb.h中定义

19.3.45 csi_ept_burst_enable

*csi_error_t csi_ept_burst_enable(csp_ept_t *ptEptBase,uint8_t byCgsrc,uint8_t byCgflt, bool bEnable)*

19.3.45.1 功能描述

群脉冲模式配置

19.3.45.2 参数/返回值说明

1. 参数

pteptBase： 定义详见 csp_gptb_t

byCgsrc：群脉冲门控输入源选择（0：GPTA_CHA 1:GPTA_CHB）

byCgflt：门控输入数字滤波控制(0~7)

bEnable：ENABLE/DISABLE

2. 返回值：

CSI_OK： 设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数	说明	概述及其变量位置
pteptBase	csp_ept_t中说明	在csp_ept.h中定义
byCgsrc	8位参数，0：GPTA_CHA 1:GPTA_CHB	
byCgflt	8位参数，范围为0~7，详解如下： typedef enum { GPTA_CGFLT_BP = 0, GPTA_CGFLT_2, GPTA_CGFLT_3, GPTA_CGFLT_4, GPTA_CGFLT_6, GPTA_CGFLT_8, GPTA_CGFLT_16, GPTA_CGFLT_32 }csp_gpta_cnflt_e;	在csp_ept.h中定义
bEnable	ENABLE/DISABLE	在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

20 LPT

20.1 概述

LPT是一个低功耗定时器，可以选择PCLK/4，ISCLK，IMCLK，EMCLK或者外部管脚输入信号作为时钟源。LPT支持在低功耗模式下将系统唤醒。

ATP CSI接口LPT的设计中，提供丰富的配置及其操作。配置方面包括基本的定时功能配置、PWM输出模式配置，ETCB触发模式配置等。

20.2 API列表

Table 20-1 LPT CSI接口函数

API	说明	函数位置
csi_lpt_int_enable	LPT中断使能功能	lpt.c
csi_lpt_timer_init	初始化LPT定时器功能	
csi_lpt_uninit	软件复位控制	
csi_lpt_stop	计数器停止控制	
csi_lpt_count_mode	设置计数器工作模式	
csi_lpt_get_remaining_value	获取计数器剩余值	
csi_lpt_get_load_value	获取LPT加载值	
csi_lpt_is_running	检测LPT运行状态	
csi_lpt_pwm_para_updata	更新LPT PWM输出参数	
csi_lpt_rearm_sync	软件重置同步功能	
csi_lpt_set_evtrg	同步触发输出端口控制	
csi_lpt_set_fre	LPT频率设置	
csi_lpt_pwm_init	PWM输出初始化	
csi_lpt_pwm_start_sync	外部触发同步启动PWM功能	
csi_lpt_change_duty	改变LPT占空比	
csi_lpt_start	启动LPT功能	
csi_lpt_start_sync	启动LPT同步触发功能	
csi_lpt_set_sync	设置LPT同步触发功能	

csi_lpt_swsync_enable	LPT软件同步使能控制	

20.3 API详细说明

20.3.1 csi_lpt_int_enable

*void csi_lpt_int_enable(csp_lpt_t *ptLptBase, lpt_int_e eLptInt,bool bEnable)*

20.3.1.1 功能描述

LPT中断使能功能。

20.3.1.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eLptInt：中断模式，枚举定义详见csi_lpt_intsrc_e。

bEnable：启用中断或禁止中断。

2. 返回值：无返回值。

3. 参数/返回值说明

参数	说明	概述及其枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	LPT寄存器结构体指针类型 csp_lpt_t在csp_lpt.h中定义
eLptInt	typedef enum { LPT_INTSRC_NONE = (0x00ul << 0), LPT_INTSRC_TRGEV0 = (0x01ul << 0), LPT_INTSRC_MATCH = (0x01ul << 1), LPT_INTSRC_PEND = (0x01ul << 2), }csi_lpt_intsrc_e;	LPT的中断类型 lpt_int_e在csp_lpt.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

20.3.2 csi_lpt_timer_init

*csi_error_t csi_lpt_timer_init(csp_lpt_t *ptLptBase,csi_lpt_clksrc_e eClk, uint32_t wTimeOut)*

20.3.2.1 功能描述

初始化LPT定时器功能。

20.3.2.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eClk：时钟选择，枚举定义详见csi_lpt_clksrc_e。

wTimeOut：定时时间。

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
eClk	typedef enum { LPT_CLK_PCLK_DIV4 = (0x00ul), LPT_CLK_ISCLK = (0x01ul), LPT_CLK_IMCLK_DIV4 = (0x02ul), LPT_CLK_EMCLK = (0x03ul), LPT_CLK_IN_RISE = (0x04ul), LPT_CLK_IN_FALL = (0x05ul), }csi_lpt_clksrc_e;	在lpt.h中定义
wTimeOut	定时时间， uint32类型，单位ms	
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.3 csi_lpt_uninit

*void csi_lpt_uninit(csp_lpt_t *ptLptBase)*

20.3.3.1 功能描述

LPT软件复位。

20.3.3.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

2. 返回值：无返回值。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作	在csp_lpt.h中定义

20.3.4 csi_lpt_stop

*void csi_lpt_stop(csp_lpt_t *ptLptBase)*

20.3.4.1 功能描述

LPT停止控制。

20.3.4.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t。

2. 返回值

   无返回值。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义

20.3.5 csi_lpt_count_mode

*void csi_lpt_count_mode(csp_lpt_t *ptLptBase, csi_lpt_cntmode_e eCntMode)*

20.3.5.1 功能描述

设置LPT计数器工作模式。

20.3.5.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t。

eCntMode：计数工作模式，枚举定义详见csi_lpt_cntmode_e。

2. 返回值

无返回值。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
eCntMode	typedef enum { LPT_CNT_CONTINU = (0x00ul), LPT_CNT_ONCE = (0x01ul) }csi_lpt_cntmode_e;	在lpt.h中定义

20.3.6 csi_lpt_get_remaining_value

*uint32_t csi_lpt_get_remaining_value(csp_lpt_t *ptLptBase)*

20.3.6.1 功能描述

获取计数器剩余值。

20.3.6.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t。

2. 返回值

计数器剩余值，uint32_t类型。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
remaining value	返回计数器剩余值，uint32_t类型	

20.3.7 csi_lpt_get_load_value

*uint32_t csi_lpt_get_load_value(csp_lpt_t *ptLptBase)*

20.3.7.1 功能描述

获取计数器加载值。

20.3.7.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t。

2. 返回值

   计数器加载值，uint32_t类型。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
load value	返回计数器加载值，uint32_t类型	

20.3.8 csi_lpt_is_running

*bool csi_lpt_is_running(csp_lpt_t *ptLptBase)*

20.3.8.1 功能描述

查询LPT运行状态。

20.3.8.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t。

2. 返回值

   LPT运行状态，bool类型，true为正在运行，false为停止运行。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
running status	返回lpt运行状态，true 或 false	在mm.h中定义 #define true 1 #define false 0

20.3.9 csi_lpt_pwm_para_updata

*void csi_lpt_pwm_para_updata(csp_lpt_t *ptLptBase, uint16_t hwCmp, uint16_t hwPrdr, csi_lpt_updata_e eModeUpdata)*

20.3.9.1 功能描述

更新LPT PWM输出参数。

20.3.9.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

hwCmp：设定比较值。

hwPrdr：设定输出周期值。

eModeUpdata:模式设置，枚举定义详见csi_lpt_updata_e

2. 返回值

无返回值。

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
hwCmp	设定比较值,数据类型uint16_t	
hwPrdr	设定输出周期值,数据类型uint16_t	
eModeUpdata	typedef enum { UPDATA_IM = 0, UPDATA_PEND }csi_lpt_updata_e;	在lpt.h中定义

20.3.10 csi_lpt_rearm_sync

*csi_error_t csi_lpt_rearm_sync(csp_lpt_t *ptLptBase, uint8_t bySync)*

20.3.10.1 功能描述

软件重置同步功能。

20.3.10.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

bySync： 同步通道，配置0有效，当大于等于1的时候，返回CSI_ERROR

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
bySync	数据类型 uint8_t，同步通道，0有效	
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.11 csi_lpt_set_evtrg

*csi_error_t csi_lpt_set_evtrg(csp_lpt_t *ptLptBase, csi_lpt_trgout_e eTrgOut, csi_lpt_trgsrc_e eTrgsrc, uint8_t byTrgprd)*

20.3.11.1 功能描述

同步触发输出端口控制。

20.3.11.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eTrgOut： 事件触发通道，LPT_TRGOUT0有效，其余返回CSI_ERROR

eTrgsrc：TRGEV0事件的触发源选择，枚举定义详见csi_lpt_trgsrc_e

byTrgprd：TRGEV0事件计数的周期设置

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义

eTrgOut	事件触发通道， LPT_TRGOUT0有效，其余返回CSI_ERROR
	`typedef enum`
	`{`
	`LPT_TRGOUT0 = 0,`
	`}csi_lpt_trgout_e;`
		在lpt.h中定义

eTrgsrc	typedef enum { LPT_TRGSRC_DIS = (0x00ul), LPT_TRGSRC_ZRO = (0x01ul), LPT_TRGSRC_PRD = (0x02ul), LPT_TRGSRC_ZRO_PRD = (0x03ul), LPT_TRGSRC_CMP = (0x04ul) }csi_lpt_trgsrc_e;	在lpt.h中定义
byTrgprd	TRGEV0事件计数的周期设置,参数范围1-16	
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.12 csi_lpt_set_fre

*csi_error_t csi_lpt_set_fre(csp_lpt_t *ptLptBase, csi_lpt_clksrc_e eClk, uint16_t hwHz)*

20.3.12.1 功能描述

LPT频率设置。

20.3.12.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eClk：时钟选择，枚举定义详见csi_lpt_clksrc_e。

hwHz：频率设置

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明

参数/返回值	说明	枚举定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
eClk	csi_lpt_clksrc_e中定义，请参阅13.3.2.2中参数说明	在lpt.h中定义
hwHz	频率设置，uint16_t类型	
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.13 csi_lpt_pwm_init

*csi_error_t csi_lpt_pwm_init(csp_lpt_t *ptLptBase, csi_lpt_pwm_config_t *ptLptPara)*

20.3.13.1 功能描述

PWM输出初始化

20.3.13.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

ptLptPara：LPT PWM输出参数设置结构体指针，具体定义详见csi_lpt_pwm_config_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
ptLptPara	typedef struct { uint8_t byStartpol; uint8_t byIdlepol; uint8_t byClksrc; uint8_t byCycle; uint8_t byInt; uint8_t byRev1; uint8_t byRev2; uint8_t byRev3; uint32_t wFreq; }csi_lpt_pwm_config_t;	在lpt.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.14 csi_lpt_pwm_start_sync

*csi_error_t csi_lpt_pwm_start_sync(csp_lpt_t *ptLptBase, csi_lpt_pwm_config_t *ptLptPara)*

20.3.14.1 功能描述

外部触发同步启动PWM功能。

20.3.14.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

ptLptPara： LPT PWM输出参数设置结构体指针，具体定义详见csi_lpt_pwm_config_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
ptLptPara	csi_lpt_pwm_config_t中定义，请参阅13.3.12.2中参数说明	在lpt.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.15 csi_lpt_change_duty

*csi_error_t csi_lpt_change_duty(csp_lpt_t *ptLptBase, uint32_t wDutyCycle)*

20.3.15.1 功能描述

更改LPT PWM输出占空比

20.3.15.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

wDutyCycle： 占空比设置0-100

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
wDutyCycle	占空比设置， 0%<占空比<=100	
csi_error_t	csi_error_t中定义值。	在common.h中定义

20.3.16 csi_lpt_start

*csi_error_t csi_lpt_start(csp_lpt_t *ptLptBase)*

20.3.16.1 功能描述

启动LPT功能。

20.3.16.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.17 csi_lpt_start_sync

*csi_error_t csi_lpt_start_sync(csp_lpt_t *ptLptBase, csi_lpt_clksrc_e eClk, uint32_t wTimeMs)*

20.3.17.1 功能描述

启动LPT同步触发功能。

20.3.17.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eClk：时钟选择，枚举定义详见csi_lpt_clksrc_e。

wTimeMs： 定时时间

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
eClk	csi_lpt_clksrc_e中定义，请参阅13.3.2.2中参数说明	在lpt.h中定义
wTimeMs	定时时间设置，单位ms	
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.18 csi_lpt_set_sync

*csi_error_t csi_lpt_set_sync(csp_lpt_t *ptLptBase, csi_lpt_trgin_e eTrgin, csi_lpt_trgmode_e eSyncMode, bool bARearmEnable)*

20.3.18.1 功能描述

设置LPT同步触发功能

20.3.18.2 参数/返回值说明

1. 参数

ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

eTrgin：同步触发输入，LPT_TRG_SYNCIN0有效，具体定义详见csi_lpt_trgin_e。

eSyncMode：模式选择，连续模式和一次模式，具体定义详见csi_lpt_syncmode_e

bARearmEnable： 0：禁止硬件自动REARM，1：周期结束时，自动REARM

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
bySync	typedef enum { LPT_TRG_SYNCIN0 = 0, }csi_lpt_trgin_e;	在lpt.h中定义

eSyncMode	typedef enum { LPT_SYNC_CONT = (0x00ul), LPT_SYNC_ONCE = (0x01ul) }csi_lpt_syncmode_e;	在lpt.h中定义
bARearmEnable	bool类型数值，ENBALE/DISABLE	ENBALE：周期结束时，自动REARM DISABLE：禁止硬件自动REARM 在common.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

20.3.19 csi_lpt_swsync_enable

*void csi_lpt_swsync_enable(csp_lpt_t *ptLptBase, bool bEnable)*

20.3.19.1 功能描述

LPT软件同步使能控制

20.3.19.2 参数/返回值说明

1. 参数

   ptLptBase：LPT寄存器结构体指针，具体定义详见csp_lpt_t

   bEnable：使能软件同步或禁止软件同步

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptLptBase	csp_lpt_t 类型指针，指向LPT的基地址，用于进行寄存器操作。	在csp_lpt.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止软件同步 在common.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

21 RTC

21.1 概述

RTC 是一个实时日历模块，可以用来实现时钟、闹表、定时功能。时钟源可以选择 IMOSC/4，ISOSC，ESOSC，EMOSC/4。选择的时候，请考虑振荡器的容差会对时间的准确程度产生直接的影响。

RTC 的很多寄存器不会受到复位的影响。

21.2 API列表

Table20-1RTC CSI接口函数

API	说明	函数位置
csi_rtc_init	初始化RTC，主要是时钟源和RTC计时方式（24H/12H）。	rtc.c
csi_rtc_rb_enable	使能/禁止回读功能。	
csi_rtc_change_fmt	改变RTC的计时方式（24H/12H）。	
csi_rtc_set_time	设置时间。	
csi_rtc_get_time	读取当前时间。	
csi_rtc_set_alarm	设置闹表时间。	
csi_rtc_get_alarm_remaining_time	获得距离闹表的倒计时时间。单位:s	
csi_rtc_cancel_alarm	取消闹表。	
csi_rtc_start_as_timer	将RTC当做一个定时器使用。	
csi_rtc_int_enable	使能RTC的某个中断。	
csi_rtc_start	开始运行。	
csi_rtc_stop	停止运行。	
csi_rtc_set_evtrg	设置RTC向外输出的触发事件。	

21.3 API详细描述

21.3.1 csi_rtc_init

*void csi_rtc_init(csp_rtc_t *ptRtc, csi_rtc_config_t *tConfig)*

21.3.1.1 功能描述

用于对 RTC 进行初始化，包括时钟源、计时模式（12 小时制/24 小时制）。

21.3.1.2 参数说明

参数	说明	位置
ptRtc	指向 RTC 控制寄存器结构体的指针，用于进行寄存器操作。	csp_rtc.h

tConfig	结构体变量。
	typedef struct {
	uint8_t byClkSrc; /// RTC_ISOSC/RTC_EMOSC/RTC_IMOSC_DIV4/RTC_EMOSC_DIV4
	uint8_t byFmt; // RTC_24FMT/RTC_12FMT
	} csi_rtc_config_t;
		rtc.h

21.3.2 csi_rtc_rb_enable

*void csi_rtc_rb_enable(csp_rtc_t *ptRtc, bool bEnable)*

21.3.2.1 功能描述

RTC 的回读功能是可以开关的。打开后，可以通过 RTC_TIMR 和 RTC_DATR 读取当前的实时时间。关闭就不支持回读，对功耗敏感的应用或时间段可以尝试关闭回读功能。

21.3.2.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   bEnable：bool型参数，可选值ENABLE/DISABLE，或1/0。

2. 返回值：无

21.3.3 csi_rtc_change_fmt

*void csi_rtc_change_fmt(csp_rtc_t *ptRtc, rtc_fmt_e eFmt)*

21.3.3.1 功能描述

改变 RTC 时间模式（12H <-> 24H）。

21.3.3.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   eFmt：枚举型参数，可选值RTC_24FMT / RTC_12FMT。

2. 返回值：无

21.3.4 csi_rtc_set_time

*csi_error_t csi_rtc_set_time(csp_rtc_t *ptRtc, csi_rtc_time_t *rtctime)*

21.3.4.1 功能描述

设置改变 RTC 的时间。

21.3.4.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   csi_rtc_time_t：时间结构体参数。

2. 返回值：csi_error_t型数据。当写入的时间超出允许的范围时，会返回CSI_ERROR。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptRtc	RTC寄存器结构体指针，用于进行寄存器操作。	csp_rtc.h

rtctime	时间结构体参数。
	typedef struct {
	int iSec; ///< Second. [0-59]
	int iMin; ///< Minute. [0-59]
	int iHour; ///< Hour. [0-23]
	int iMday; ///< Day. [1-31]
	int iMon; ///< Month. [1-12]	

	int iYear; ///< Year-1900. [70-] !NOTE:100=2000
	int iWday; /// weekday [1-7]
	int iYday; ///< Days in year.[0-365] !NOTE:January 1st = 0
	int iIsdst; /// Non-0 if daylight savings time is in effect
	int iPm; ///< PM. [0/1]
	} csi_rtc_time_t;	

21.3.5 csi_rtc_get_time

*void csi_rtc_get_time(csp_rtc_t *ptRtc, csi_rtc_time_t *rtctime)*

21.3.5.1 功能描述

读取 RTC 当前的时间。如果回读功能没有开启，则一直读回设置值。

21.3.5.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   csi_rtc_time_t：时间结构体参数。同参数说明。

2. 返回值：无。

21.3.6 csi_rtc_set_alarm

*csi_error_t csi_rtc_set_alarm(csp_rtc_t *ptRtc, csi_rtc_alarm_e eAlm, csi_rtc_time_t *ptAlmTime)*

21.3.6.1 功能描述

用于设置闹表的时间和闹表模式。

21.3.6.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   eAlm：APT32F110x可以使用闹表AlarmA、AlarmB，所以可以传入RTC_ALMA、RTC_ALMB。

ptAlmTime：时间结构体参数。同参数说明。

2. 返回值：csi_error_t型枚举变量。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptRtc	RTC寄存器结构体指针，用于进行寄存器操作。	csp_rtc.h
eAlm	csi_rtc_alarm_e枚举类型变量，APT32F110x可以使用闹表AlarmA、AlarmB，所以可以传入RTC_ALMA、RTC_ALMB。	rtc.h
ptAlmTime	csi_rtc_time_t结构变量。 如果设置结构体成员，如tm_sec= 0xff，表示屏蔽该位的比较。	rtc.h

21.3.7 csi_rtc_get_alarm_remaining_time

*uint32_t csi_rtc_get_alarm_remaining_time(csp_rtc_t *ptRtc, csi_rtc_alarm_e eAlm)*

21.3.7.1 功能描述

获得距离闹表的倒计时时间。单位:s。

21.3.7.2 参数说明

1. 参数

ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

eAlm：APT32F110x可以使用闹表AlarmA、AlarmB，所以可以传入RTC_ALMA、RTC_ALMB。

2. 返回值：uint32_t型变量。如果闹表时间距离当前时间超过一年，当前驱动会返回CSI_UNSUPPORTED。

21.3.8 csi_rtc_cancel_alarm

*void csi_rtc_cancel_alarm(csp_rtc_t *ptRtc, csi_rtc_alarm_e eAlm)*

21.3.8.1 功能描述

关闭闹表，关闭闹表中断。

21.3.8.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   eAlm：APT32F110x可以使用闹表AlarmA、AlarmB，所以可以传入RTC_ALMA、RTC_ALMB。

2. 返回值：无。

21.3.9 csi_rtc_start_as_timer

*void csi_rtc_start_as_timer(csp_rtc_t *ptRtc, csi_rtc_timer_e ePrd)*

21.3.9.1 功能描述

将 RTC 当做定时器来使用，但是定时间隔只能在几个选项中选择。

21.3.9.2 参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   ePrd：csi_rtc_timer_e枚举变量，定时间隔。

2. 返回值：无。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptRtc	RTC寄存器结构体指针，用于进行寄存器操作。	csp_rtc.h
ePrd	csi_rtc_timer_e枚举变量。 typedef enum { RTC_TIMER_DIS = 0, RTC_TIMER_0_5S, RTC_TIMER_1S, RTC_TIMER_1MIN, RTC_TIMER_1H, RTC_TIMER_1DAY, RTC_TIMER_1MON }csi_rtc_timer_e;	rtc.h

21.3.10 csi_rtc_int_enable

*void csi_rtc_int_enable(csp_rtc_t *ptRtc, csi_rtc_intsrc_e eIntSrc, bool bEnable)*

21.3.10.1功能描述

中断使能。

21.3.10.2参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   eIntSrc：csi_rtc_intsrc_e枚举变量。

   bEnable：bool型变量

2. 返回值：无。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptRtc	RTC寄存器结构体指针，用于进行寄存器操作。	rtc.h

eIntSrc	rtc_int_e枚举变量。
	typedef enum{
	RTC_INTSRC_NONE = 0x00, //无中断
	RTC_INTSRC_ALMA = 0x01, //AlarmA 中断
	RTC_INTSRC_ALMB = 0x02, //AlarmB 中断
	RTC_INTSRC_CPRD = 0x04, //CPRD 定时中断
	RTC_INTSRC_TRGEV0 = 0x08, //输出触发事件 1 中断
	RTC_INTSRC_TRGEV1= 0x10 //输出触发事件 2 中断
	}csi_rtc_intsrc_e;	

bEnable	bool型变量，ENABLE/DISABLE	common.h

21.3.11 csi_rtc_start

*void csi_rtc_start(csp_rtc_t *ptRtc)*

21.3.11.1功能描述

RTC 开始工作。

21.3.11.2参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

2. 返回值：无。

21.3.12 csi_rtc_stop

*void csi_rtc_stop(csp_rtc_t *ptRtc)*

21.3.12.1功能描述

RTC 停止工作。

21.3.12.2参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

2. 返回值：无。

21.3.13 csi_rtc_set_evtrg

*csi_error_t csi_rtc_set_evtrg(csp_rtc_t *ptRtc, csi_rtc_trgsel_e eEvtrg, csi_rtc_trgsrc_e eTrgSrc, uint8_t byTrgPrd)*

21.3.13.1功能描述

设置 RTC 输出触发事件，包括事件通道、事件源、几次事件后生成对外触发事件。

21.3.13.2参数说明

1. 参数

   ptRtc：RTC寄存器结构体指针，用于进行寄存器操作。

   eEvtrg：指定事件通道号，RTC_TRGSEL0或RTC_TRGSEL1。

eTrgSrc：指定事件源。

byTrgPrd：指定几次事件后生成对外触发事件，

2. 返回值：

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptRtc	RTC寄存器结构体指针，用于进行寄存器操作。	csp_rtc.h
eEvtrg	csi_rtc_trgsel_e枚举型变量。 typedef enum{ RTC_TRGSEL0 = 0, RTC_TRGSEL1 }csi_rtc_trgsel_e;	rtc.h

eTrgSrc	csi_rtc_trgsrc_e枚举型变量。
	typedef enum{
	RTC_TRGOUT_DIS = 0, //禁止触发输出
	RTC_TRGOUT_ALRMA, //AlarmA 时间到触发
	RTC_TRGOUT_ALRMB, //AlarmB 时间到触发
	RTC_TRGOUT_ALRMA_ALRMB, //AlarmA 或 B 时间到触发
	RTC_TRGOUT_CPRD //CPRD 定时事件到触发
	}csi_rtc_trgsrc_e;
		rtc.h

22 DMA

22.1 概述

DMA（直接存储器访问）在源和目标间直接传输数据，源或目标可以是 SRAM，U(S)ART, SPI, CMP, TKEY or IIC, Flash 存储器和 ADC 等，其中 Flash 存储器和 ADC 只能为源。

DMA 有四个通道 DMA0~DMA3。

22.2 API列表

Table 20-1 RTC CSI接口函数

API	说明	函数位置
csi_dma_ch_init	初始化DMA	dma.c
csi_dma_ch_start	开启DMA通道转换。	
csi_dma_int_enable	使能/禁止DMA中断。	
csi_dma_ch_stop	停止DMA转换。	
csi_dma_soft_rst	软件复位DMA模块。	
csi_dma_get_msg	获取DMA中断信息并清除/保留状态	

22.3 API详细描述

22.3.1 csi_dma_ch_init

*csi_error_t csi_dma_ch_init(csp_dma_t *ptDmaBase, csi_dma_ch_e eDmaCh, csi_dma_ch_config_t *ptChCfg)*

22.3.1.1 功能描述

对 DMA 进行初始化，包括通道选择和模式设置等。

22.3.1.2 参数说明

1. 参数

ptDmaBase：DMA寄存器结构体指针，指向DMA基地址。

eDmaCh：DMA通道，详见枚举定义csi_dma_ch_e。

ptChCfg：DMA配置结构体指针，结构体定义详见csi_dma_ch_config_t。

2. 返回值：

csi_error_t：CSI_OK/CSI_ERROR。

3. 参数/返回值说明表

参数	说明	位置
ptDmaBase	指向 DMA 控制寄存器结构体的指针，指向 DMA 基地址。	在csp_dma.h中定义
eDmaCh	typedef enum { DMA_CH0 = 0, DMA_CH1, DMA_CH2, DMA_CH3, } csi_dma_ch_e;	DMA有四个通道：DMA0~3。 在dma.h中定义

ptChCfg	typedef struct
	{
	uint8_t bySrcLinc; //low transfer count src addr inc control
	uint8_t bySrcHinc; //high transfer count src addr inc control
	uint8_t byDetLinc; //lowtransfer count det addr inc control
	uint8_t byDetHinc; //high transfer count det addr inc control
	uint8_t byDataWidth; //transfer data size width
	uint8_t byReload; //auto reload
	uint8_t byTransMode; //dma serve(transfer) mode
	uint8_t byTsizeMode; //Tsize transfer mode
	uint8_t byReqMode; //request mode
	uint32_t wInt; //interrupt
	} csi_dma_ch_config_t;
		bySrcLinc：低位源地址递增控制
		bySrcHinc：高位源地址递增控制
		byDetLinc：低位目的地址递增控制
		byDetHinc：高位目的地址递增控制
		byDataWidth：传输数据宽度
		byReload：自动重载
		byTransMode：DMA传输模式
		byTsizeMode：传输数据大小
		byReqMode：DMA请求模式
		wInter：中断配置
		在dma.h中定义

22.3.2 csi_dma_ch_start

*ccsi_error_t csi_dma_ch_start(csp_dma_t *ptDmaBase, csi_dma_ch_e eDmaCh, void *pSrcAddr, void *pDstAddr, uint16_t hwHTranNum, uint16_t hwLTranNum)*

22.3.2.1 功能描述

开启 DMA 传输。

22.3.2.2 参数说明

1. 参数

ptDmaBase：DMA寄存器结构体指针，指向DMA基地址。

eDmaCh：DMA通道，详见枚举定义csi_dma_ch_e。

pSrcAddr：指向源数据首地址。

pDstAddr：指向目的首地址。

hwHTranNum：高传输数据长度，总的传输长度 = 高传输长度 * 低传输长度。

hwLTranNum：低传输数据长度，总的传输长度 = 高传输长度 * 低传输长度。

2. 返回值：

csi_error_t：CSI_OK/CSI_ERROR

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptDmaBase	csp_usart_t类型指针，请参阅22.3.1.2参数说明	在csp_dma.h中定义
eDmaCh	枚举变量，选择DMA通道，请参阅22.3.1.2参数说明	在dma.h中定义
pSrcAddr	void 类型指针，指向源数据缓存首地址	调用时需强制转换下(void *)
pDstAddr	void 类型指针，指向目的缓存首地址	调用时需强制转换下(void *)
hwHTranNum	uint16_t 类型数据，高传输的数据长度	
hwLTranNum	uint16_t 类型数据，高传输的数据长度	
return value	CSI_OK/CSI_ERROR	

22.3.3 csi_dma_int_enable

*void csi_dma_int_enable(csp_dma_t *ptDmaBase, csi_dma_ch_e eDmaCh, csi_dma_intsrc_e eIntSrc, bool bEnable)*

22.3.3.1 功能描述

使能/禁止 DMA 中断。

22.3.3.2 参数说明

1. 参数

ptDmaBase：DMA寄存器结构体指针，指向DMA基地址。

eDmaCh：DMA通道，详见枚举定义csi_dma_ch_e。

eIntSrc：枚举变量，用于按选择DMA中断，详见枚举定义csi_dma_intsrc_e。

bEnable：bool类型，使能/禁止中断。

2. 返回值：无

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptDmaBase	csp_usart_t类型指针，请参阅22.3.1.2参数说明	在csp_dma.h中定义
eDmaCh	枚举变量，选择DMA通道，请参阅22.3.1.2参数说明	在dma.h中定义

eIntSrc	typedef enum
	{
	DMA_INTSRC_NONE = (0x00ul << 0), //NONE interrupt
	DMA_INTSRC_LTCIT = (0x01ul << 29), //LTICT interrupt
	DMA_INTSRC_TCIT = (0x01ul << 30) //LTICT interrupt
	}csi_dma_intsrc_e;
		DMA_INTSRC_NONE：无中断
		DMA_INTSRC_LTCIT：低传输计数
		DMA_INTSRC_TCIT：传输计数
		在dma.h中定义

bEnable	bool类型，使能/禁止中断。	

22.3.4 csi_dma_ch_stop

*void csi_dma_ch_stop(csp_dma_t *ptDmaBase, csi_dma_ch_e eDmaCh)*

22.3.4.1 功能描述

停止DMA传输。如果当前有原子传输，则DMA在当前原子传输完成后立即停止。若当前没有原子传输，则DMA立即停止。

22.3.4.2 参数说明

1. 参数

ptDmaBase：DMA寄存器结构体指针，指向DMA基地址。

eDmaCh：DMA通道，详见枚举定义csi_dma_ch_e。

2. 返回值：无。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptDmaBase	csp_usart_t类型指针，请参阅22.3.1.2参数说明	在csp_dma.h中定义
eDmaCh	枚举变量，选择DMA通道，请参阅22.3.1.2参数说明	在dma.h中定义

22.3.5 csi_dma_soft_rst

*void csi_dma_soft_rst(csp_dma_t *ptDmaBase)*

22.3.5.1 功能描述

软件复位 DMA 模块。

22.3.5.2 参数说明

1. 参数

ptDmaBase：DMA寄存器结构体指针，指向DMA基地址。

2. 返回值：无。

3. 参数/返回值说明表

参数/返回值	说明	位置
ptDmaBase	csp_usart_t类型指针，请参阅22.3.1.2参数说明	在csp_dma.h中定义

22.3.6 csi_dma_get_msg

bool csi_dma_get_msg(csi_dma_ch_e eDmaCh, bool bClrEn)

22.3.6.1 功能描述

获取 DMA 中断状态，并清除/保留该状态

22.3.6.2 参数说明

1. 参数

eDmaCh：DMA通道，详见枚举定义csi_dma_ch_e。

bClrEn：bool类型，清除/保留当前中断状态

2. 返回值：bool类型，true/false。

3. 参数/返回值说明表

参数/返回值	说明	位置
eDmaCh	枚举变量，选择DMA通道，请参阅22.3.1.2参数说明	在dma.h中定义
bClrEn	bool类型，清除/保留当前中断状态	ENABLE:清除 DISABLE：保留
Return value	bool类型，true表示有中断，false表示没有中断	

23 UART

23.1 概述

UART是一个简单通用的异步串行接收和发送接口，支持8位的数据通信，支持校验位，每次发送都以一个停止位结束。

APT CSI接口提供了UART包括发送、接收等相关配置和操作。

23.2 API列表

Table 21-1 UART CSI接口函数

API	说明	函数位置
csi_uart_init	初始化UART	uart.c
csi_uart_start	开启UART收发功能	
csi_uart_stop	关闭UART收发功能	
csi_uart_set_buffer	配置串口接收数据缓存(buffer)	
csi_uart_putc	发送一个字符	
csi_uart_getc	接收一个字符	
csi_uart_send	发送数据	
csi_uart_receive	接收数据	
csi_uart_dma_rx_init	初始化UART DMA接收模式	
csi_uart_dma_tx_init	初始化UART DMA发送模式	
csi_uart_recv_dma	UART DMA 接收数据	
csi_uart_send_dma	UART DMA 发送数据	
csi_uart_get_msg	获取UART接收/发送是否完成消息	
csi_uart_clr_msg	清除UART接收/发送消息(设置为空闲)	

23.3 API详细说明

23.3.1 csi_uart_init

*csi_error_t csi_uart_init(csp_uart_t *ptUartBase, csi_uart_config_t *ptUartCfg)*

23.3.1.1 功能描述

初始化UART

23.3.1.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

ptUartCfg：UART配置结构体指针，结构体定义详见csi_uart_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，UART0/UART1/UART2，指向对应UART的基地址	系统有三个UART，即UART(0/1/2)，定义了对应的结构体指UART0、UART1、UART2，指向对应UART的基地址。 在devices.c中定义

ptUartCfg	typedef struct {
	uint32_t wBaudRate; //baud rate
	uint32_t wInt; //interrupt
	uint8_t byParity; //parity type
	uint8_t byTxMode; //send mode:
	uint8_t byRxMode; //recv mode:
	} csi_uart_config_t;
		初始化配置参数：
		wBaudRate：波特率
		wInter：中断源选择
		byParity：校验
		byTxMode：发送模式
		byRxMode：接收模式
		在uart.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.2 csi_uart_start

*csi_error_t csi_uart_start(csp_uart_t *ptUartBase, csi_uart_func_e eFunc)*

23.3.2.1 功能描述

开启(使能)UART收发功能

23.3.2.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eFunc：UART的RX/TX使能，可以全部使能，也可以单独对RX/TX中某一个使能，枚举定义详见csi_uart_func_e。

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eFunc	typedef enum{
	UART_FUNC_RX = 0, //uart only support rx
	UART_FUNC_TX, //uart only support tx
	UART_FUNC_RX_TX //uart support rx and tx
	}csi_uart_func_e;
		UART_FUNC_RX：使能RX
		UART_FUNC_TX：使能TX
		UART_FUNC_RX_TX：使能RX和TX
		在uart.h钟定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.3 csi_uart_stop

*csi_error_t csi_uart_stop(csp_uart_t *ptUartBase, csi_uart_func_e eFunc)*

23.3.3.1 功能描述

关闭(禁止)UART收发功能

23.3.3.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eFunc：UART的RX/TX禁止，可以全部禁止，也可以单独对RX/TX中某一个禁止，枚举定义详见csi_uart_func_e。

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数/返回值说明

参数参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eFunc	typedef enum {
	UART_FUNC_RX = 0, //uart only support rx
	UART_FUNC_TX, //uart only support tx
	UART_FUNC_RX_TX //uart support rx and tx
	}csi_uart_func_e;
		UART_FUNC_RX：禁止RX
		UART_FUNC_TX：禁止TX
		UART_FUNC_RX_TX：禁止RX和TX
		在uart.h钟定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.4 csi_uart_set_buffer

*void csi_uart_set_buffer(csp_uart_t *ptUartBase, ringbuffer_t *ptRingbuf, uint8_t *pbyRdBuf, uint16_t hwLen)*

23.3.4.1 功能描述

配置串口接收数据缓存(buffer)，中断接收模式时调用

23.3.4.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

ptRingbuf：循环buf(ringbuf)结构体指针，结构体定义详见ringbuffer_t。

pbyRdBuf：串口接收数据缓存(buffer)数组指针，指向buffer首地址。

hwLen：接收缓存大小(接收数据数组长度)，用户定义。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义
ptRingbuf	typedef struct ringbuffer { uint8_t *pbyBuf; uint16_t hwSize; uint16_t hwWrite; uint16_t hwRead; uint16_t hwDataLen; } ringbuffer_t;	参数说明： pbyBuf：buf指针，指向缓存 hwSize：循环buf大小 hwWrite：写入数据长度 hwRead：读取数据长度 hwDataLen：数据长度 在ringbuf.h中定义
pbyRdBuf	uint8_t 类型指针，指向接收数据缓存区首地址	指向接收数据缓存(接收数据数组)，赋值给循环buf的pbyBuf
hwLen	uint16_t 类型数据，缓存大小，即接收数组长度	赋值给循环buf的hwSize

23.3.5 csi_uart_putc

*void csi_uart_putc(csp_uart_t *ptUartBase, uint8_t byData)*

23.3.5.1 功能描述

UART发送一个字符。

23.3.5.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

byData：要发送的字符

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义
byData	uint8_t 类型数据，要发送的字符	阻塞方式发送一个字符

23.3.6 csi_uart_getc

*uint8_t csi_uart_getc(csp_uart_t *ptUartBase)*

23.3.6.1 功能描述

UART接收一个字符

23.3.6.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

2. 返回值

接收到的字符。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义
return value	uint8_t 类型数据，接收到的字符	阻塞方式接收一个字符

23.3.7 csi_uart_send

*int32_t csi_uart_send(csp_uart_t *ptUartBase, const void *pData, uint16_t hwSize)*

23.3.7.1 功能描述

发送数据，UART发送有两种模式(轮训/中断)，初始化时用户可配置。

23.3.7.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

pData：指向要发送数据buffer首地址。

hwSize：要发送数据长度。

2. 返回值

已发送数据长度/ CSI_OK/ CSI_ERROR。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义
pData	void 类型指针，指向发送数据缓存首地址	发送数据缓存类型为uint8_t，调用时需强制转换下(void *)
hwSize	uint16_t 类型数据，要发送数据的长度	
return value	轮训模式：发送完成的数据长度 中断模式：CSI_OK/CSI_ERROR	UART发送数据两种模式返回数据不一样，具体如下： 轮训模式，返回发送数据长度 中断模式：返回发送成功/失败

23.3.8 csi_uart_receive

*uint16_t csi_uart_receive(csp_uart_t *ptUartBase, void *pData, uint16_t hwSize, uint32_t wTimeOut)*

23.3.8.1 功能描述

获取UART接收到的数据，UART接收有三种模式(轮训/中断1/中断2)，中断模式1为接收用户指定长度数据；中断模式2为动态接收一串字符，字符长度为动态(不固定)；接收模式初始化时用户可配置。

23.3.8.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

pData：指向用户获取数据buffer首地址，即把接收到的数据读取到用户定义的buffer中。

hwSize：要获取的数据长度，轮训模式和中断模式1时参数有意义；中断模式2忽略此参数。

wTimeOut：获取UART串口数据超时处理，轮训模式时参数有意义；另外两种模式忽略此参数。

2. 返回值

获取到的数据长度。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义
pData	void 类型指针，指向发送数据缓存首地址	获取数据缓存类型为uint8_t，调用时需强制转换下(void *)
hwSize	uint16_t 类型数据，要获取数据的长度	用户指定的数据长度
return value	uint16_t 类型数值，获取到的数据长度	实际返回的数据长度

23.3.9 csi_uart_dma_rx_init

*csi_error_t csi_uart_dma_rx_init(csp_uart_t *ptUartBase, csi_dma_ch_e eDmaCh, csi_etb_ch_e eEtbCh)*

23.3.9.1 功能描述

UART DMA接收模式初始化

23.3.9.2 参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eDmaCh：DMA的通道选择，枚举定义详见csi_dma_ch_e。

eEtbCh：ETB触发通道，枚举定义详见csi_etb_ch_e。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eDmaCh	typedef enum {
	DMA_CH0 = 0, //dma channel 0
	DMA_CH1, //dma channel 1
	DMA_CH2, //dma channel 2
	DMA_CH3, //dma channel 3
	} csi_dma_ch_e;
		DMA有4个通道可选，
		DMA_CH0~3
		在dma.h中定义

eEtbCh	typedef enum {
	ETB_CH0 = 0, //etb channel 0
	ETB_CH1, //etb channel 1
	ETB_CH2, //etb channel 2
	ETB_CH3, //etb channel 3
	ETB_CH4,
	ETB_CH5,
	ETB_CH6,
	ETB_CH7,
	ETB_CH8, //DMA channel
	ETB_CH9, //DMA channel
	ETB_CH10, //DMA channel
	ETB_CH11 //DMA channel
	} csi_etb_ch_e;
		ETB有12个通道，可用作DMA触发的通道有4个，ETB_CH8~11
		在etb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.10 csi_uart_dma_tx_init

*csi_error_t csi_uart_dma_tx_init(csp_uart_t *ptUartBase, csi_dma_ch_e eDmaCh, csi_etb_ch_e eEtbCh)*

23.3.10.1功能描述

UART DMA发送模式初始化

23.3.10.2参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eDmaCh：DMA的通道选择，枚举定义详见csi_dma_ch_e。

eEtbCh：ETB触发通道，枚举定义详见csi_etb_ch_e。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置

ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eDmaCh	typedef enum {
	DMA_CH0 = 0, //dma channel 0
	DMA_CH1, //dma channel 1
	DMA_CH2, //dma channel 2
	DMA_CH3, //dma channel 3
	} csi_dma_ch_e;
		DMA有4个通道可选，
		DMA_CH0~3
		在dma.h中定义

eEtbCh	typedef enum {
	ETB_CH0 = 0, //etb channel 0
	ETB_CH1, //etb channel 1
	ETB_CH2, //etb channel 2
	ETB_CH3, //etb channel 3
	ETB_CH4,
	ETB_CH5,
	ETB_CH6,
	ETB_CH7,
	ETB_CH8, //DMA channel
	ETB_CH9, //DMA channel
	ETB_CH10, //DMA channel
	ETB_CH11 //DMA channel
	} csi_etb_ch_e;
		ETB有12个通道，可用作DMA触发的通道有4个，ETB_CH8~11
		在etb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.11 csi_uart_recv_dma

*csi_error_t csi_uart_recv_dma(csp_uart_t *ptUartBase, csi_dma_ch_e eDmaCh, void *pData, uint16_t hwSize)*

23.3.11.1功能描述

UART通过DMA接收数据

23.3.11.2参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eDmaCh：DMA的通道选择，枚举定义详见csi_dma_ch_e。

pData：指向接收数据buffer的指针。

hwSize：接收数据的长度

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eDmaCh	typedef enum {
	DMA_CH0 = 0, //dma channel 0
	DMA_CH1, //dma channel 1
	DMA_CH2, //dma channel 2
	DMA_CH3, //dma channel 3
	} csi_dma_ch_e;
		DMA有4个通道可选，
		DMA_CH0~3
		在dma.h中定义

pData	Void 类型指针	指向接收数据buffer指针
hwSize	uint16_t 类型数据	要接收数据的长度
csi_error_t	csi_error_t 中定义值	在common.h中定义

23.3.12 csi_uart_send_dma

*csi_error_t csi_uart_send_dma(csp_uart_t *ptUartBase, csi_dma_ch_e eDmaCh, const void *pData, uint16_t hwSize)*

23.3.12.1功能描述

UART通过DMA发送数据

23.3.12.2参数/返回值说明

1. 参数

ptUartBase：UART寄存器结构体指针，指向UART基地址，结构体定义详见csp_uart_t。

eDmaCh：DMA的通道选择，枚举定义详见csi_dma_ch_e。

pData：指向发送数据buffer的指针。

hwSize：接收数据的长度

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数说明

参数/返回值	说明	概述及其结构体定义位置

ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eDmaCh	typedef enum {
	DMA_CH0 = 0, //dma channel 0
	DMA_CH1, //dma channel 1
	DMA_CH2, //dma channel 2
	DMA_CH3, //dma channel 3
	} csi_dma_ch_e;
		DMA有4个通道可选，
		DMA_CH0~3
		在dma.h中定义

pData	Void 类型指针	指向发送数据buffer指针
hwSize	uint16_t 类型数据	要发送数据的长度
csi_error_t	csi_error_t 中定义值	在common.h中定义

23.1.1 csi_uart_get_msg

*bool csi_uart_get_msg(csp_uart_t *ptUartBase, csi_uart_wkmode_e eWkMode, bool bClrEn)*

23.1.1.1 功能描述

获取 UART 接收/发送数据是否完毕消息

23.1.1.2 参数/返回值说明

1. 参数

   ptUartBase：UART 寄存器结构体指针，指向 UART 基地址，结构体定义详见 csp_uart_t。

   eWkMode：UART 工作状态，接收/发送，枚举定义详见 csi_uart_wkmode_e。

   bClrEn：ENABLE/DISABLE,，获取到消息时，是否清除接收/发送状态（设置为空闲），使用时一般使能此选项。

2. 返回值

   布尔类型变量，true/ false(1/0)，true 表示接收/发送完毕，反之亦然。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eWkMode	typedef enum{
	UART_SEND = 0, //uart send
	UART_RECV = 1, //uart receive
	}csi_uart_wkmode_e;
		发送/接收两种模式
		在uart.h中定义

bClrEn	布尔类型，ENABLE/DISABLE	是否清除UART接收/发送状态（设置为空闲）

返回值	布尔类型，true/false(真/假)	true：1 false：0

23.1.2 csi_uart_clr_msg

*void csi_uart_clr_msg(csp_uart_t *ptUartBase, csi_uart_wkmode_e eWkMode)*

23.1.2.1 功能描述

清除 UART 接收/发送消息，即设置为空闲。

23.1.2.2 参数/返回值说明

1. 参数

   ptUartBase：UART 寄存器结构体指针，指向 UART 基地址，结构体定义详见 csp_uart_t。

   eWkMode：UART 工作状态，接收/发送，枚举定义详见 csi_uart_wkmode_e。

2. 返回值：无。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUartBase	csp_uart_t 类型指针，请参阅23.3.1.2参数说明	在devices.c中定义

eWkMode	`typedef enum{`
	`  UART_SEND = 0, //uart send`
	`  UART_RECV = 1, //uart receive`
	`}csi_uart_wkmode_e;`
		发送/接收两种模式
		在uart.h中定义

24 USART

24.1 概述

USART是一个简单通用的同步异步串行接收和发送接口，支持5 到 9 位的数据通信，支持校验位，每次发送都以1/1.5/2个停止位结束。

APT CSI接口提供了USART包括发送、接收等相关配置和操作。

24.2 API列表

Table 21-1 USART CSI接口函数

API	说明	函数位置
csi_usart_init	初始化USART	usart.c
csi_usart_int_enable	使能/禁止USART中断	
csi_usart_start	开启USART收发功能	
csi_usart_stop	关闭USART收发功能	
csi_usart_set_buffer	配置串口接收数据缓存(buffer)	
csi_usart_putc	发送一个字符	
csi_usart_getc	接收一个字符	
csi_usart_send	发送数据	
csi_usart_receive	接收数据	
csi_usart_dma_rx_init	USART的DMA接收模式初始化	
csi_usart_dma_tx_init	USART的DMA发送模式初始化	
csi_usart_send_dma	USART通过DMA方式发送	
csi_usart_recv_dma	USART通过DMA方式接收	
csi_usart_get_msg	获取USART接收/发送完成信息，并清除/保留状态	
csi_usart_clr_msg	清除USART接收/发送完成状态信息	

24.3 API详细说明

24.3.1 csi_usart_init

*csi_error_t csi_usart_init(csp_usart_t *ptUsartBase, csi_usart_config_t *ptUsartCfg)*

24.3.1.1 功能描述

初始化USART

24.3.1.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

ptUsartCfg：USART配置结构体指针，结构体定义详见csi_usart_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，USART0，指向对应USART的基地址	系统有一个USART，即USART0，定义了结构体指针USART0，指向对应USART的基地址。 在csp_usart.h中定义

ptUsartCfg	typedef struct {
	uint32_t wBaudRate; //baud rate
	uint32_t wInt; //interrupt
	uint8_t byParity; //parity type
	uint8_t byDatabit; //data bits
	uint8_t byStopbit; //stop bits
	uint8_t byClkSrc; //clk source
	uint8_t byMode; //usart mode, sync/async
	uint8_t byTxMode; //send mode: polling/interrupt
	uint8_t byRxMode; //recv mode: polling/interrupt0/interrupt1
	bool bClkOutEn; //enable usartclk out
	} csi_usart_config_t;
		初始化配置参数：
		wBaudRate：波特率
		wInter：中断源选择
		byParity：校验
		byDatabit：数据位数
		byStopbit：停止位数
		byClkSrc：时钟源选择
		byTxMode：发送模式
		byRxMode：接收模式

		bClkOutEn：USART CLK输出使能 在usart.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.2 csi_usart_int_enable

*void csi_usart_int_enable(csp_usart_t *ptUsartBase, csi_usart_intsrc_e eIntSrc, bool bEnable)*

24.3.2.1 功能描述

使能/禁止USART中断

24.3.2.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

eIntSrc：USART中断源结构体，结构体定义详见csi_usart_intsrc_e。

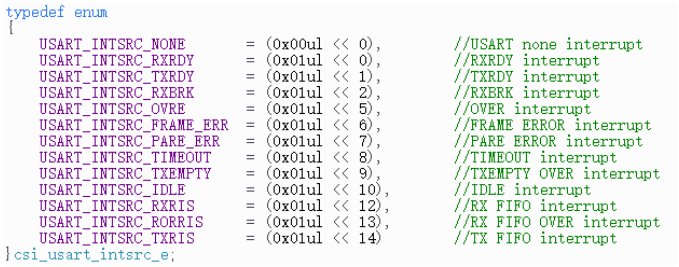
bEnable：使能/禁止中断。

2. 返回值

无返回值

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，USART0，指向对应USART的基地址	系统有一个USART，即USART0，定义了结构体指USART0，指向对应USART的基地址。 在csp_usart.h中定义

eIntSrc		初始化配置参数： USART_INTSRC_NONE：无中断 USART_INTSRC_RXRDY：接收端待机 USART_INTSRC_TXRDY：发送端待机 USART_INTSRC_RXBRK：接收端Break USART_INTSRC_OVRE：溢出错误 USART_INTSRC_FRAME_ERR：帧错误 USART_INTSRC_PARE_ERR：校验错 USART_INTSRC_TIMEOUT：超时 USART_INTSRC_TXEMPTY： 发送缓冲空闲 USART_INTSRC_IDLE：空闲 USART_INTSRC_RXRIS：接收FIFO USART_INTSRC_RORRIS：接收FIFO溢出 USART_INTSRC_TXRIS：发送FIFO 在usart.h中定义
bEnable	bool类型，使能/禁止中断	

24.3.3 csi_usart_start

*csi_error_t csi_usart_start(csp_usart_t *ptUsartBase, csi_usart_func_e eFunc)*

24.3.3.1 功能描述

开启(使能)USART收发功能

24.3.3.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

eFunc：UART的RX/TX使能，可以全部使能，也可以单独对RX/TX中某一个使能，枚举详见csi_usart_func_e。

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

eFunc	typedef enum{
	USART_FUNC_RX = 0, //uart only support rx
	USART_FUNC_TX, //uart only support tx
	USART_FUNC_RX_TX //uart support rx and tx
	}csi_usart_func_e;
		USART_FUNC_RX：使能RX
		USART_FUNC_TX：使能TX
		USART_FUNC_RX_TX：使能RX和TX
		在usart.h钟定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.4 csi_usart_stop

*csi_error_t csi_usart_stop(csp_usart_t *ptUsartBase, csi_usart_func_e eFunc)*

24.3.4.1 功能描述

关闭(禁止)USART收发功能

24.3.4.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

eFunc：UART的RX/TX禁止，可以全部禁止，也可以单独对RX/TX中某一个禁止，枚举详见csi_usart_func_e。

2. 返回值

CSI_OK：成功。

CSI_ERROR：失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

eFunc	typedef enum{
	USART_FUNC_RX = 0, //uart only support rx
	USART_FUNC_TX, //uart only support tx
	USART_FUNC_RX_TX //uart support rx and tx
	}csi_usart_func_e;
		USART_FUNC_RX：禁止RX
		USART_FUNC_TX：禁止TX
		USART_FUNC_RX_TX：禁止RX和TX
		在usart.h钟定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.5 csi_usart_set_buffer

*void csi_usart_set_buffer(csp_usart_t *ptUsartBase, ringbuffer_t *ptRingbuf, uint8_t *pbyRdBuf, uint16_t hwLen)*

24.3.5.1 功能描述

配置串口接收数据缓存(buffer)，中断接收模式时调用

24.3.5.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

ptRingbuf：循环buf(ringbuf)结构体指针，结构体定义详见ringbuffer_t。

pbyRdBuf：串口接收数据缓存(buffer)数组指针，指向buffer首地址。

hwLen：接收缓存大小(接收数据数组长度)，用户定义。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

ptRingbuf	typedef struct ringbuffer { uint8_t *pbyBuf; uint16_t hwSize; uint16_t hwWrite; uint16_t hwRead; uint16_t hwDataLen; } ringbuffer_t;	参数说明： pbyBuf：buf指针，指向缓存 hwSize：循环buf大小 hwWrite：写入数据长度 hwRead：读取数据长度 hwDataLen：数据长度 在ringbuf.h中定义
pbyRdBuf	uint8_t 类型指针，指向接收数据缓存区首地址	指向接收数据缓存(接收数据数组)，赋值给循环buf的pbyBuf
hwLen	uint16_t 类型数据，缓存大小，即接收数组长度	赋值给循环buf的hwSize

24.3.6 csi_usart_putc

*void csi_usart_putc(csp_usart_t *ptUsartBase, uint8_t byData)*

24.3.6.1 功能描述

USART发送一个字符。

24.3.6.2 参数/返回值说明

1. 参数

   ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

   byData：要发送的字符

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
byData	uint8_t 类型数据，要发送的字符	阻塞方式发送一个字符

24.3.7 csi_usart_getc

*uint8_t csi_usart_getc(csp_usart_t *ptUsartBase)*

24.3.7.1 功能描述

USART接收一个字符

24.3.7.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

2. 返回值

接收到的字符。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
return value	uint8_t 类型数据，接收到的字符	阻塞方式接收一个字符

24.3.8 csi_usart_send

*int16_t csi_usart_send(csp_usart_t *ptUsartBase, const void *pData, uint16_t hwSize)*

24.3.8.1 功能描述

发送数据，USART发送有两种模式(轮训/中断)，初始化时用户可配置。

24.3.8.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

pData：指向要发送数据buffer首地址。

hwSize：要发送数据长度。

2. 返回值

已发送数据长度/ CSI_OK/ CSI_ERROR。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
pData	void 类型指针，指向发送数据缓存首地址	发送数据缓存类型为uint8_t，调用时需强制转换下(void *)
hwSize	uint16_t 类型数据，要发送数据的长度	
return value	轮训模式：发送完成的数据长度 中断模式：CSI_OK/CSI_ERROR	USART发送数据两种模式返回数据不一样，具体如下： 轮训模式，返回发送数据长度 中断模式：返回发送成功/失败

24.3.9 csi_usart_receive

*uint16_t csi_usart_receive(csp_usart_t *ptUsartBase, void *pData, uint16_t hwSize, uint32_t wTimeOut)*

24.3.9.1 功能描述

获取USART接收到的数据，USART接收有三种模式(轮训/中断1/中断2)，中断模式1为接收用户指定长度数据；中断模式2为动态接收一串字符，字符长度为动态(不固定)；接收模式初始化时用户可配置。

24.3.9.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

pData：指向用户获取数据buffer首地址，即把接收到的数据读取到用户定义的buffer中。

hwSize：要获取的数据长度，轮训模式和中断模式1时参数有意义；中断模式2忽略此参数。

wTimeOut：获取USART串口数据超时处理，轮训模式时参数有意义；另外两种模式忽略此参数。

2. 返回值

获取到的数据长度。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
pData	void 类型指针，指向发送数据缓存首地址	获取数据缓存类型为uint8_t， 调用时需强制转换下(void *)
hwSize	uint16_t 类型数据，要获取数据的长度	用户指定的数据长度
return value	uint16_t 类型数值，获取到的数据长度	实际返回的数据长度

24.3.10 csi_usart_dma_rx_init

*csi_error_t csi_usart_dma_rx_init(csp_usart_t *ptUsartBase, csi_dma_ch_e eDmaCh, csi_etb_ch_e eEtbCh)*

24.3.10.1 功能描述

USART通过DMA方式收数据初始化配置。

24.3.10.2 参数/返回值说明

1. 参数

   ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

   eDmaCh：选择DMA通道，枚举定义详见csi_dma_ch_e。

   eEtbCh：选择ETCB通道，注意只有通道8-11支持DMA触发，枚举定义详见csi_etb_ch_e。

2. 返回值

   CSI_OK/CSI_ERROR。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
eDmaCh	typedef enum { DMA_CH0 = 0, DMA_CH1, DMA_CH2, DMA_CH3, } csi_dma_ch_e;	DMA_CH0：DMA0通道 DMA_CH1：DMA1通道 DMA_CH2：DMA2通道 DMA_CH3：DMA3通道

		在dma.h中定义

eEtbCh	typedef enum {
	ETB_CH0 = 0, //etb channel 0 id number
	ETB_CH1, //etb channel 1 id number
	ETB_CH2, //etb channel 2 id number
	ETB_CH3, //etb channel 3 id number
	ETB_CH4,
	ETB_CH5,
	ETB_CH6,
	ETB_CH7,
	ETB_CH8, //DMA channel
	ETB_CH9, //DMA channel
	ETB_CH10, //DMA channel
	ETB_CH11 //DMA channel
	} csi_etb_ch_e;
		ETB_CH0~ETB_CH7：ETCB通道0~7
		ETB_CH8~ETB_CH11：ETCB通道8~11，支持DMA触发通道
		在etb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.11 csi_usart_dma_tx_init

*csi_error_t csi_usart_dma_tx_init(csp_usart_t *ptUsartBase, csi_dma_ch_e eDmaCh, csi_etb_ch_e eEtbCh)*

24.3.11.1 功能描述

USART通过DMA方式发送数据初始化配置。

24.3.11.2 参数/返回值说明

1. 参数

   ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

   eDmaCh：选择DMA通道，枚举定义详见csi_dma_ch_e。

   eEtbCh：选择ETCB通道，注意只有通道8-11支持DMA触发，枚举定义详见csi_etb_ch_e。

2. 返回值

   CSI_OK/CSI_ERROR。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

eDmaCh	typedef enum { DMA_CH0 = 0, DMA_CH1, DMA_CH2, DMA_CH3, } csi_dma_ch_e;	DMA_CH0：DMA0通道 DMA_CH1：DMA1通道 DMA_CH2：DMA2通道 DMA_CH3：DMA3通道 在dma.h中定义

eEtbCh	typedef enum {
	ETB_CH0 = 0, //etb channel 0 id number
	ETB_CH1, //etb channel 1 id number
	ETB_CH2, //etb channel 2 id number
	ETB_CH3, //etb channel 3 id number
	ETB_CH4,
	ETB_CH5,
	ETB_CH6,
	ETB_CH7,
	ETB_CH8, //DMA channel
	ETB_CH9, //DMA channel
	ETB_CH10, //DMA channel
	ETB_CH11 //DMA channel
	} csi_etb_ch_e;
		ETB_CH0~ETB_CH7：ETCB通道0~7
		ETB_CH8~ETB_CH11：ETCB通道8~11，支持DMA触发通道
		在etb.h中定义

csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.12 csi_usart_send_dma

*csi_error_t csi_usart_send_dma(csp_usart_t *ptUsartBase, const void *pData, uint8_t byDmaCh, uint16_t hwSize)*

24.3.12.1 功能描述

USART通过DMA方式发送数据，在发送FIFO状态满足设置的条件，将发出DMA请求信号。该信号经过ETCB（事件触发控制器）模块选择后发送到DMA对应的通道x（0=0~3）。

24.3.12.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

pData：指向要发送数据buffer首地址。

byDmaCh：DMA通道，可配参数详见csi_dma_ch_e。

hwSize：要发送数据长度。

2. 返回值

CSI_OK/CSI_ERROR。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义
pData	void 类型指针，指向发送数据缓存首地址	发送数据缓存类型为uint8_t，调用时需强制转换下(void *)
byDmaCh	选择USART发送数据的DMA通道,可选通道DMA_CH0/DMA_CH1/DMA_CH2/DMA_CH3。	
hwSize	uint16_t 类型数据，要发送数据的长度	
csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.13 csi_usart_recv_dma

*csi_error_t csi_usart_recv_dma(csp_usart_t *ptUsartBase, void *pData, uint8_t byDmaCh, uint16_t hwSize)*

24.3.13.1 功能描述

USART通过DMA方式接收数据。在接收FIFO状态满足USART_DMACR寄存器的RXMODE位设置的条件，将发出DMA请求信号。该信号经过ETCB（事件触发控制器）模块选择后发送到DMA对应的通道x（0=0~3）。

24.3.13.2 参数/返回值说明

4. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

pData：指向要存放接收数据buffer首地址。

byDmaCh：选择DMA通道，可配参数详见csi_dma_ch_e。

hwSize：要接收数据长度。

5. 返回值

CSI_OK/CSI_ERROR。

6. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

pData	void 类型指针，指向存放接收数据缓存首地址	接收数据缓存类型为uint8_t，调用时需强制转换下(void *)
byDmaCh	选择USART接收数据的DMA通道,可选通道DMA_CH0/ DMA_CH1/DMA_CH2/DMA_CH3。	
hwSize	uint16_t 类型数据，接收数据的长度	
csi_error_t	csi_error_t 中定义值	在common.h中定义

24.3.14 csi_usart_get_msg

*bool csi_usart_get_msg(csp_usart_t *ptUsartBase, csi_usart_wkmode_e eWkMode, bool bClrEn)*

24.3.14.1 功能描述

获取USART接收和发送完成状态，并且清除/保留该状态。

24.3.14.2 参数/返回值说明

1. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

eWkMode：USART工作模式，即USART工作于接收还是发送模式，枚举定义详见csi_usart_wkmode_e。

bClrEn：清除/保留发送/接收完成状态。

2. 返回值

bool类型，true/false

3. 参数说明

参数	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

eWkMode	`typedef enum{`
	`  USART_SEND = 0, //usart send`
	`  USART_RECV = 1, //usart receive`
	`}csi_usart_wkmode_e;`
		USART工作模式：
		USART_SEND：发送
		USART_RECV：接收
		在usart.h中定义

bClrEn	清除/保留USART接收/发送完成状态。	ENABLE：清除状态

		DISABLE：保留状态
Return value	接收/发送完成状态返回true，反之，false	

24.3.15 csi_usart_clr_msg

*void csi_usart_clr_msg(csp_usart_t *ptUsartBase, csi_usart_wkmode_e eMode)*

24.3.15.1 功能描述

清除USART接收/发送完成状态，即接收/发送状态设置为空闲。

24.3.15.2 参数/返回值说明

4. 参数

ptUsartBase：USART寄存器结构体指针，指向USART基地址，结构体定义详见csp_usart_t。

eWkMode：USART工作模式，即USART工作于接收还是发送模式，枚举定义详见csi_usart_wkmode_e。

5. 返回值

无返回值。

6. 参数说明

参数	说明	概述及其结构体定义位置
ptUsartBase	csp_usart_t 类型指针，请参阅24.3.1.2参数说明	在csp_usart.h中定义

eWkMode	`typedef enum{`
	`  USART_SEND = 0, //usart send`
	`  USART_RECV = 1, //usart receive`
	`}csi_usart_wkmode_e;`
		USART工作模式：
		USART_SEND：发送
		USART_RECV：接收
		在usart.h中定义

25 SPI

25.1 概述

APT CSI接口SPI的设计中，提供SPI丰富的配置及其操作。

25.2 API列表

Table 25-1 SPI外设CSI接口函数

API	说明	函数位置
csi_spi_init	spi模块初始化	spi.c
csi_spi_uninit	spi模块去初始化	
csi_spi_mode	spi主从机模式设置	
csi_spi_cp_format	spi时钟极性与相位的设置	
csi_spi_baud	spi通讯速率的设置	
csi_spi_frame_len	spi数据帧长度的设置	
csi_spi_get_state	spi读写状态的获取	
csi_spi_send	Spi发送接口函数	
csi_spi_receive	Spi接收接口函数	
csi_spi_send_receive	Spi收发接口函数	
csi_spi_clr_rxfifo	Spi接收fifo的清除	
spi_irqhandler	Spi中断参考函数	
csi_spi_send_receive_1byte	Spi同步收发一个字节	
csi_spi_buff_send	Spi发送一个buff接口函数	
csi_spi_send_receive_x8	Spi发送小于或者等于八个数据接口函数	
csi_spi_send_receive_d8	Spi发送大于或者等于八个数据接口函数	
csi_spi_send_dma	Spi使用DMA发送数据	
csi_spi_recv_dma	Spi使用DMA接收数据	

25.3 API详细说明

25.3.1 csi_spi_init

*csi_error_t csi_spi_init(csp_spi_t *ptSpiBase , csi_spi_config_t *ptSpiCfg)*

25.3.1.1 功能描述

Spi的初始化，主要配置主从机，通讯速率，时钟极性与相位，数据帧长，中断源等。

25.3.1.2 参数/返回值说明

1. 参数

ptSpiBase：spi外设寄存器的地址指针。

ptSpiCfg：spi初始化结构体指针，结构体定义详见csi_spi_config_t。

2. 返回值

CSI_OK：设置成功； CSI_ERROR：设置失败。

3. 参数说明表

参数	说明	概述及其枚举/结构体定义位置

ptSpiBase	参数：SPI0
	csp_spi_t *SPI0 = (csp_spi_t *)(APB_SPI0_BASE);
	typedef struct
	{
	__IOM uint32_t CR0; //Control 0
	__IOM uint32_t CR1; //Control 1
	__IOM uint32_t DR; //Rx/Tx data
	__IM uint32_t SR; //Status
	__IOM uint32_t CPSR; //Clock prescale
	__IOM uint32_t IMSCR; //Interrupt enable
	__IM uint32_t RISR; //Raw interrupt
	__IM uint32_t MISR; //Mask interrupt
	__OM uint32_t ICR; //Interrupt clear
	__IOM uint32_t DMACR; //Dma Control Register
	__OM uint32_t SRR; //Software Reset Register
	} csp_spi_t;
		该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在devices.c,指针类型定义csp_spi.h

ptSpiCfg	typedef struct { uint32_t wSpiBaud; uint8_t bySpiMode; uint8_t bySpiPolarityPhase; uint8_t bySpiFrameLen; uint8_t byInt; uint8_t byTxMode; uint8_t byRxMode; uint8_t byTxRxMode; }csi_spi_config_t;	该参数是一个指针，指向spi配置参数结构体。定义在spi.h。主要包含了，主从机，时钟极性和相位，帧长度，spi波特率，内部中断源,工作模式等。
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.2 csi_spi_uninit

*csi_error_t csi_spi_uninit(csp_spi_t *ptSpiBase)*

25.3.2.1 功能描述

对于已经配置的spi,假如需要进行去初始化，可以调用该函数，该函数包括时钟的关闭，中断的关闭，spi寄存器的重置等

25.3.2.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针。

2. 返回值

   CSI_OK：设置成功； CSI_ERROR：设置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.3 csi_spi_mode

*csi_error_t csi_spi_mode(csp_spi_t *ptSpiBase , csi_spi_mode_e eMode)*

25.3.3.1 功能描述

设置spi工作在主机还是从机模式

25.3.3.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针。

   eMode：主机还是从机模式，枚举定义详见csi_spi_mode_e。

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
eMode	typedef enum { SPI_MASTER, SPI_SLAVE }csi_spi_mode_e;	两种模式：主机，从机。 在spi.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.4 csi_spi_cp_format

*csi_error_t csi_spi_cp_format(csp_spi_t *ptSpiBase , csi_spi_cp_format_e eFormat)*

25.3.4.1 功能描述

设置spi的时钟极性和相位

25.3.4.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针。

   eFormat：时钟极性和相位配置参数，枚举定义详见csi_spi_cp_format_e。

2. 返回值

   CSI_OK：设置成功

   CSI_ERROR：设置失败

3. 参数/返回值说明表

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
eFormat	typedef enum { SPI_FORMAT_CPOL0_CPHA0 = 0, SPI_FORMAT_CPOL0_CPHA1, SPI_FORMAT_CPOL1_CPHA0, SPI_FORMAT_CPOL1_CPHA1, }csi_spi_cp_format_e;	四种模式： 1：时钟空闲为0，相位为0 2：时钟空闲为0，相位为1 3：时钟空闲为1，相位为0 4：时钟空闲为1，相位为1 在spi.h中定义
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.5 csi_spi_baud

*uint32_t csi_spi_baud(csp_spi_t *ptSpiBase , uint32_t wBaud)*

25.3.5.1 功能描述

设置spi的通讯速率

25.3.5.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针。

   wBaud：期望的通信速率。

2. 返回值

   uint32_t类型的值，返回spi的波特率

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
wBaud	Spi工作的波特率	假如期望通讯速率是2兆，这个参数就设置为2000000；如果期望的速率是8兆，则设置为8000000；主从机的极限频率请参阅使用手册
uint32_t类型的值	实际spi的波特率	该类型为uint32_t

25.3.6 csi_spi_frame_len

*csi_error_t csi_spi_frame_len(csp_spi_t *ptSpiBase , csi_spi_frame_len_e eLength)*

25.3.6.1 功能描述

设置一帧数据的长度（4-16bit）

25.3.6.2 参数/返回值说明

1. 参数

ptSpiBase：spi外设寄存器的地址指针。

eLength：格式帧长度，枚举类型参考csi_spi_frame_len_e。

2. 返回值

CSI_OK：设置成功 CSI_ERROR：设置失败

3. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
eLength	typedef enum { SPI_FRAME_LEN_4 = 4, SPI_FRAME_LEN_5, SPI_FRAME_LEN_6, SPI_FRAME_LEN_7, SPI_FRAME_LEN_8, SPI_FRAME_LEN_9, SPI_FRAME_LEN_10, SPI_FRAME_LEN_11, SPI_FRAME_LEN_12, SPI_FRAME_LEN_13, SPI_FRAME_LEN_14, SPI_FRAME_LEN_15, SPI_FRAME_LEN_16 } csi_spi_frame_len_e;	一帧数据长度为：4-16bit 枚举类型在spi.h中定义。
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.7 csi_spi_get_state

int8_t csi_spi_get_state(csi_spi_work_mode_e eWorkMode)

25.3.7.1 功能描述

获取当前spi是否为可读可写状态

25.3.7.2 参数/返回值说明

1. 参数

eWorkMode：工作模式，读或者写，详见枚举定义csi_spi_work_mode_e。

2. 返回值

spi读写模式下的状态或者是返回错误

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
csi_spi_work_mode	typedef enum{ SPI_SEND = 0, SPI_RECV }csi_spi_work_mode_e;	SPI_SEND：发送模式 SPI_RECV：接收模式 在spi.h中定义。
Int8_t	typedef enum { SPI_STATE_IDLE = 0, SPI_STATE_BUSY }csi_spi_state_e;	csi_spi_state_e在spi.h中定义

25.3.8 csi_spi_send

*int32_t csi_spi_send(csp_spi_t *ptSpiBase , void *pData , uint32_t wSize)*

25.3.8.1 功能描述

Spi 发送 buff 数据,这个函数只发不收，使用的时候请确认你是否不用关心接收。如果这时候你要接收，那在中断里接收会比较合适

25.3.8.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针

   pData：该指针指向待发送数据buff的地址

   wSize：待发送数据的个数

2. 返回值

   返回值为发送数据的个数或者返回错误。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pData	该空指针指向待发送数据buff的地址	
wSize	待发送数据的个数	该类型为uint32_t
int32_t类型的值	发送了多少个数据	该类型为int32_t

25.3.9 csi_spi_receive

*int32_t csi_spi_receive(csp_spi_t *ptSpiBase, void *pData, uint32_t wSize)*

25.3.9.1 功能描述

读取接收fifo里面的数据到指定buff

25.3.9.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针

   pData：该指针指向待接收数据buff的地址

   wSize：待接收数据的个数。

2. 返回值

   返回值为读取数据的个数或者返回错误

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pData	该空指针指向待接收数据buff的地址	
wSize	待读取数据的个数	该类型为uint32_t
int32_t类型的值	读取了多少个数据	该类型为int32_t

25.3.10 csi_spi_send_receive

*int32_t csi_spi_send_receive(csp_spi_t *ptSpiBase, void *pDataout, void *pDatain, uint32_t wSize)*

25.3.10.1 功能描述

发送指定发送buff的数据，同时接收数据到指定接收buff。使用这个函数基本可以满足所有从机的指令需求。

25.3.10.2 参数/返回值说明

1. 参数

ptSpiBase：spi外设寄存器的地址指针。

pDataout：该指针指向待发送数据buff的地址

pDatain：该指针指向待接收数据buff的地址

wSize：收发数据的长度

2. 返回值

实际收发数据的长度或者错误

3. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pDataout	该空指针指向待发送数据buff的地址	
pDatain	该空指针指向待接收数据buff的地址	
int32_t类型的值	收发了多少个数据	该类型为int32_t

25.3.11 csi_spi_clr_rxfifo

*void csi_spi_clr_rxfifo(csp_spi_t *ptSpiBase)*

25.3.11.1 功能描述

清除spi接收fifo的数据，当你准备读取fifo数据，但是又不确定接收fifo中是否还留有上一次的未读走数据。这时你就可以使用这个函数。

25.3.11.2 参数/返回值说明

1. 参数

   ptSpiBase：spi外设寄存器的地址指针。

2. 返回值

   无返回值

3. 参数/返回值说明

参数	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h

25.3.12 spi_irqhandler

*__attribute__((weak)) void spi_irqhandler(csp_spi_t *ptSpiBase)*

25.3.12.1 功能描述

这个是spi的参考中断函数，用户可以直接参考里面的代码进行中断处理，或者自己重新编写。SPI_TXIM_INT中断和SPI_RXIM_INT中断发生时，直接在中断函数里面处理即可，无需手动清除这两个标志。其它的中断标志则需要手动软件清除，比如SPI_RORIM_INT(接收溢出)，SPI_RTIM_INT（接收超时）。

25.3.12.2 参数/功能描述

1. 参数

ptSpiBase：spi外设寄存器的地址指针

2. 返回值

无返回值

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h

25.3.13 csi_spi_send_receive_1byte

*uint8_t csi_spi_send_receive_1byte(csp_spi_t *ptSpiBase,uint8_t byData)*

25.3.13.1 功能描述

同步发送接收一个字节，这个函数是spi控制从机的最小单元，结合各种控制spi命令时序，可以衍生出各种命令接口函数，用户可以自由发挥

25.3.13.2 参数/功能描述

1. 参数

ptSpiBase：spi外设寄存器的地址指针

byData：待发送的数据

2. 返回值

返回接收到的一个数据

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
byData	待发送的一个数据，默认为一个字节	
Uint8_t类型的值	读取的一个字节数据	该类型为uint8_t

25.3.14 csi_spi_buff_send

*csi_error_t csi_spi_buff_send(csp_spi_t *ptSpiBase,void *pDataOut,uint8_t bySize)*

25.3.14.1 功能描述

Spi同步发送buff数据,这个函数的功能和csi_spi_send类似，只是速度快更快，快大约一倍

25.3.14.2 参数/功能描述

1. 参数

ptSpiBase：spi外设寄存器的地址指针

pDataOut：待发送的数据buff地址

bySize：待发送数据的个数

2. 返回值

CSI_OK：成功

CSI_ERROR：失败

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pDataOut	该指针指向待发送数据buff的地址	
bySize	待发送数据的个数	该类型为uint8_t
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.15 csi_spi_send_receive_x8

*csi_error_t csi_spi_send_receive_x8(csp_spi_t *ptSpiBase, void *pDataOut,void *pDataIn,uint32_t wSize)*

25.3.15.1 功能描述

Spi发送小于或者等于八个数据接口函数，当数据个数小于等于8的时候用这个函数发送，速度会比较快

25.3.15.2 参数/功能描述

1. 参数

ptSpiBase：spi外设寄存器的地址指针

pDataOut：待发送的数据buff地址

pDataIn：该指针指向待接收数据buff的地址

wSize：待发送数据的个数

2. 返回值

CSI_OK：成功

CSI_ERROR：失败

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pDataOut	该指针指向待发送数据buff的地址	
pDataIn	该指针指向待接收数据buff的地址	
wSize	待发送数据的个数	该类型为uint32_t
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.16 csi_spi_send_receive_d8

*csi_error_t csi_spi_send_receive_d8(csp_spi_t *ptSpiBase, uint8_t *pDataOut,uint8_t *pDataIn, uint32_t wSize)*

25.3.16.1 功能描述

Spi发送大于或者等于八个数据接口函数，当数据个数大于等于8的时候用这个函数发送，速度会比较快

25.3.16.2 参数/功能描述

1. 参数

ptSpiBase：spi外设寄存器的地址指针

pDataOut：待发送的数据buff地址

pDataIn：该指针指向待接收数据buff的地址

wSize：待发送数据的个数

2. 返回值

CSI_OK：成功

CSI_ERROR：失败

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pDataOut	该指针指向待发送数据buff的地址	
pDataIn	该指针指向待接收数据buff的地址	
wSize	待发送数据的个数	该类型为uint32_t
csi_error_t	csi_error_t中定义值	在common.h中定义

25.3.17 csi_spi_send_dma

*csi_error_t csi_spi_send_dma (csp_spi_t *ptSpiBase, const void *pData, uint16_t hwSize, csp_dma_t *ptDmaBase,uint8_t byDmaCh)*

25.3.17.1 功能描述

当使用DMA发送Spi数据时，前面我们需要初始化相关etcb,dma,spi,这三个初始化好之后，再调用该接口，便可使能dma通道，同时等待对应dma请求并触发spi发送。详细参看spi dma发送示例。

25.3.17.2 参数/功能描述

1. 参数

ptSpiBase：指向spi外设寄存器的指针

pData：待发送的数据buff地址

hwSize：待发送数据的个数

byDmaCh：dma的通道数

2. 返回值

CSI_OK/CSI_ERROR。

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pData	该指针指向待发送数据buff的地址	
hwSize	待发送数据的个数	
byDmaCh	Dma通道数，110x系列芯片支持0~3	在common.h中定义

25.3.18 csi_spi_recv_dma

*csi_error_t csi_spi_recv_dma (csp_spi_t *ptSpiBase, void *pbyRecv, uint16_t hwSize, csp_dma_t *ptDmaBase,uint8_t byDmaCh)*

25.3.18.1 功能描述

当使用DMA接收Spi数据时，前面我们需要初始化相关etcb,dma,spi,这三个初始化好之后，再调用该接口，便可使能dma通道，同时等待对应dma请求并触发spi接收。详细参看spi dma接收示例。

25.3.18.2 参数/功能描述

1. 参数

ptSpiBase：指向spi外设寄存器的指针

pbyRecv：待接收数据的buff地址

hwSize：待接收数据的个数

byDmaCh：dma的通道数

2. 返回值

CSI_OK/CSI_ERROR。

3. 参数返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSpiBase	参数：SPI0 指向csp_spi_t结构体的指针，请参阅25.3.1.2中参数说明	该参数是一个外设寄存器结构体指针，固定为SPI0。SPI0的指针定义在decices.c,指针类型定义csp_spi.h
pbyRecv	该指针指向待接收数据的buff地址	
hwSize	待接收数据的个数	
byDmaCh	Dma通道数，110x系列芯片支持0~3	在common.h中定义

26 IIC

26.1 概述

为了方便客户使用，加快客户上手速度，ATP CSI接口IIC的设计中，提供简单方便的配置及其操作。配置方面包括IIC的主机模式配置，从机模式配置等一系列的配置接口。操作方面从机的收发函数，及主机的读函数和写函数等。

26.2 API列表

Table 23-1 IIC CSI 接口函数

API	说明	函数位置
csi_iic_enable	使能IIC	iic.c
csi_iic_disable	关闭IIC	
csi_iic_slave_init	IIC从机初始化	
csi_iic_master_init	IIC主机初始化	
csi_iic_write_byte	IIC主机写1byte	
csi_iic_write_nbyte	IIC主机写Nbyte	
csi_iic_read_byte	IIC主机读1byte	
csi_iic_read_nbyte	IIC主机读Nbyte	
csi_iic_slave_receive_send	IIC从机数据收发	
csi_iic_set_slave_buffer	设置从机收发缓冲区的地址和长度	
void i2c_irqhandler	IIC中断调用函数	

26.3 API详细说明

26.3.1 csi_iic_enable

*void csi_iic_enable(csp_i2c_t *ptIicBase)*

26.3.1.1 功能描述

使能IIC。

26.3.1.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

2. 返回值

无返回值。

3. 结构体/枚举说明表

结构体	结构体/枚举说明	定义位置

csp_i2c_t	typedef struct
	{
	__IOM uint32_t CR; //控制寄存器，偏移地址 0x0000
	__IOM uint32_t TADDR; //目标地址寄存器，偏移地址 0x0004
	__IOM uint32_t SADDR; //从机地址寄存器，偏移地址 0x0008
	__IOM uint32_t RSVD1; //保留，0x000C
	__IOM uint32_t DATA_CMD; //数据和命令寄存器，偏移地址 0x0010
	__IOM uint32_t SS_SCLH; //标准模式 SCL 高电平长度寄存器，偏移地址 0x0014
	__IOM uint32_t SS_SCLL; //标准模式 SCL 低电平长度寄存器，偏移地址 0x0018
	__IOM uint32_t FS_SCLH; //高速模式 SCL 高电平长度寄存器，偏移地址 0x001C
	__IOM uint32_t FS_SCLL; //高速模式 SCL 低电平长度寄存器，偏移地址 0x0020
	__IOM uint32_t RSVD2; //保留，0x0024
	__IOM uint32_t RSVD3; //保留，0x0028
	__IOM uint32_t RX_FLSEL; //接收 FIFO 阈值寄存器，偏移地址 0x002C
	__IOM uint32_t TX_FLSEL; //发送 FIFO 阈值寄存器，偏移地址 0x0030
	__IOM uint32_t RX_FL; //接收 FIFO 状态寄存器，偏移地址 0x0034
		csp_i2c.h

```
    __IOM  uint32_t  TX_FL;       //发送 FIFO 状态寄存器，偏移地址 0x0038
    __IOM  uint32_t  I2CENABLE; //使能寄存器，偏移地址 0x003C
    __IM   uint32_t  STATUS;      //状态寄存器，偏移地址 0x0040
    __IOM  uint32_t  RSVD4;       //保留，0x0044
    __IOM  uint32_t  SDA_TSETUP;//SDA SETUP 时间寄存器，偏移地址 0x0048
    __IOM  uint32_t  SDA_THOLD; //SDA HOLD 时间寄存器，偏移地址 0x004C
    __IOM  uint32_t  SPKLEN;      //毛刺干扰滤波控制寄存器，偏移地址 0x0050
    __IOM  uint32_t  RSVD5;       //保留，0x0054
    __IM   uint32_t  MISR;        //中断状态寄存器，偏移地址 0x0058
    __IOM  uint32_t  IMCR;        //中断使能寄存器，偏移地址 0x005C
    __IM   uint32_t  RISR;        //原始中断状态寄存器，偏移地址 0x0060
    __OM   uint32_t  ICR;         //中断清除寄存器，偏移地址 0x0064
    __IOM  uint32_t  RSVD6;       //保留，0x0068
    __IOM  uint32_t  SCL_TOUT;  //SCL 锁死超时控制寄存器，偏移地址 0x006C
    __IOM  uint32_t  SDA_TOUT;  // SDA 锁死超时控制寄存器，偏移地址 0x0070
    __IM   uint32_t  TX_ABRT;     //发送终止状态寄存器，偏移地址 0x0074
    __IOM  uint32_t  GCALL;       //General call 控制寄存器，偏移地址 0x0078
    __IOM  uint32_t  NACK;        //从机 NACK 控制寄存器，偏移地址 0x007C
} csp_i2c_t;
```

26.3.2 csi_iic_disable

*void csi_iic_disable(csp_i2c_t *ptIicBase)*

26.3.2.1 功能描述

关闭IIC。

26.3.2.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

2. 返回值

无返回值。

3. 结构体/枚举说明表

参数/返回值	说明	定义位置
ptIicBase	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.3 csi_iic_slave_init

*csi_error_t csi_iic_slave_init(csp_i2c_t *ptIicBase, csi_iic_slave_config_t *ptIicSlaveCfg)*

26.3.3.1 功能描述

IIC从机初始化配置函数

26.3.3.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

*ptIicSlaveCfg：IIC从机配置结构体指针。

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

csi_iic_slave_config_t	typedef struct csi_iic_slave_config {
	uint16_t hwSlaveAddr; //从机地址设置
	uint8_t bySpeedMode; //速度设置：标准、高速和超高速
	uint8_t byAddrMode; //地址模式 7/10 bit
	uint16_t hwInt; //中断标志位使能设置
	uint32_t wSdaTimeout; //SDA 超时时间设置
	uint32_t wSclTimeout; //SCL 超时时间设置
	}csi_iic_slave_config_t;
		iic.h

csi_error_t	csi_error_t中定义值	common.h

26.3.4 csi_iic_master_init

*csi_error_t csi_iic_master_init(csp_i2c_t *ptIicBase, csi_iic_master_config_t *ptIicMasterCfg)*

26.3.4.1 功能描述

主机模式初始化

26.3.4.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

*ptIicSlaveCfg：IIC主机配置结构体指针。

2. 返回值

CSI_OK：设置成功

CSI_ERROR：设置失败

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

csi_iic_master_config_t	typedef struct csi_iic_master_config {
	uint8_t bySpeedMode; //速度设置：标准、高速和超高速模式
	uint8_t byAddrMode; //地址模式 7/10 bit
	uint8_t byReStart; //重复起始位使能/失能
	uint16_t hwInt; //中断标志位使能设置
	uint32_t wSdaTimeout; //SDA 超时时间设置
	uint32_t wSclTimeout; //SCL 超时时间设置
	}csi_iic_master_config_t
		iic.h

csi_error_t	csi_error_t中定义值	common.h

26.3.5 csi_iic_write_byte

*void csi_iic_write_byte(csp_i2c_t *ptIicBase,uint32_t wdevaddr, uint32_t wWriteAdds, uint8_t byWriteAddrNumByte, uint8_t byData)*

26.3.5.1 功能描述

主机写1byte

26.3.5.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

wdevaddr：写的目标设备地址。

wWriteAdds：写入数据的起始地址。

byWriteAddrNumByte：写的起始地址长度，单位byte。

byData：需要写入的数据

2. 返回值

无返回值

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.6 csi_iic_write_nbyte

*void csi_iic_write_nbyte(csp_i2c_t *ptIicBase,uint32_t wDevAddr, uint32_t wWriteAdds, uint8_t byWriteAddrNumByte,volatile uint8_t *pbyIicData,uint32_t wNumByteToWrite)*

26.3.6.1 功能描述

主机写N byte 数据

26.3.6.2 参数/返回值说明

1. 参数

   *ptIicBase：IIC 寄存器结构体地址。

   wDevAddr：目标设备地址。

   wWriteAdds：写入数据的起始地址。

   byWriteAddrNumByte：写的起始地址长度，单位byte。

   *pbyIicData：主机准备写入的数据缓冲区地址

   wNumByteToWrite：需要写入的数据长度，单位byte

2. 返回值

   无返回值

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.7 csi_iic_read_byte

*uint8_t csi_iic_read_byte(csp_i2c_t *ptIicBase,uint32_t wDevAddr, uint32_t wReadAdds, uint8_t wReadAddrNumByte)*

26.3.7.1 功能描述

IIC主机读取1byte数据

26.3.7.2 参数/返回值说明

1. 参数

   *ptIicBase：IIC 寄存器结构体地址。

   wDevAddr：目标设备地址。

   wWriteAdds：读取数据的起始地址。

   byWriteAddrNumByte：读取的起始地址长度，单位byte。

2. 返回值

读取到的数据

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.8 csi_iic_read_nbyte

*void csi_iic_read_nbyte(csp_i2c_t *ptIicBase,uint32_t wdevaddr, uint32_t wReadAdds, uint8_t wReadAddrNumByte,volatile uint8_t *pbyIicData,uint32_t wNumByteRead)*

26.3.8.1 功能描述

通过pin name获取pin number

26.3.8.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

wdevaddr：目标设备地址。

wReadAdds：读取数据的起始地址。

wReadAddrNumByte：读取数据的起始地址长度，单位byte。

*pbyIicData：主机读取的数据存放缓冲区地址

wNumByteRead：读取的数据长度，单位byte。

2. 返回值

无返回值

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.9 csi_iic_slave_receive_send

*void csi_iic_slave_receive_send(csp_i2c_t *ptIicBase)*

26.3.9.1 功能描述

从机数据收发

26.3.9.2 参数/返回值说明

1. 参数

*ptIicBase：IIC 寄存器结构体地址。

2. 返回值

无返回值。

3. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

26.3.10 csi_iic_set_slave_buffer

*void csi_iic_set_slave_buffer(volatile uint8_t *pbyIicRxBuf,uint16_t hwIicRxSize,volatile uint8_t *pbyIicTxBuf,uint16_t hwIicTxSize)*

26.3.10.1 功能描述

设置从机收发缓冲区地址及缓冲区大小。

26.3.10.2 参数/返回值说明

1. 参数

*pbyIicRxBuf：接收数据的存放地址。

hwIicRxSize：接收缓冲区大小。

*pbyIicTxBuf：发送数据的存放地址。

hwIicTxSize：接收缓冲区大小。

2. 返回值

   无返回值

3. 结构体/枚举说明表

   无

26.3.11 i2c_irqhandler

*__attribute__((weak)) void i2c_irqhandler(csp_i2c_t *ptIicBase)*

26.3.11.1 功能描述

IIC驱动中断处理函数。

26.3.11.2 参数/返回值说明

4. 参数

5. *ptIicBase：IIC 寄存器结构体地址。

6. 返回值

   无返回值

7. 结构体/枚举说明表

结构体/枚举	说明	定义位置
csp_i2c_t	请参阅26.3.1.2中参数说明	csp_i2c.h

27 SIO

27.1 概述

单线串行输入输出(SIO)可模拟多种串行通讯协议，支持发送/接收双向数据传输，支持多种通讯速率。CSI接口提供了SIO发送、接收等相关配置和操作。

27.2 API列表

Table 27-1 SIO CSI接口函数

API	说明	函数位置
csi_sio_tx_init	初始化SIO发送配置	sio.c
csi_sio_rx_init	初始化SIO接收配置	
csi_sio_break_rst	配置接收BREAK复位	
csi_sio_timeout_rst	配置接收采样超时复位	
csi_sio_set_mode	配置SIO工作模式(接收/发送)	
csi_sio_int_enable	使能SIO中断	
csi_sio_set_buffer	配置SIO接收数据缓存(buffer)	
csi_sio_send	发送数据	
csi_sio_receive	接收数据	
csi_sio_get_recv_status	获取SIO接收状态	
csi_sio_clr_recv_status	清除SIO接收状态(设置为空闲)	

27.3 API详细说明

27.3.1 csi_sio_tx_init

*csi_error_t csi_sio_tx_init(csp_sio_t *ptSioBase, csi_sio_tx_config_t *ptTxCfg)*

27.3.1.1 功能描述

初始化SIO发送配置

27.3.1.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

ptTxCfg：SIO配置结构体指针，结构体定义详见csi_sio_tx_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置

ptSioBase	参数：SIO0
	csp_sio_t *SIOI0 = (csp_sio_t *)(APB_SIO0_BASE);
	typedef struct
	{
	__IOM uint32_t CR;
	__IOM uint32_t TXCR0;
	__IOM uint32_t TXCR1;
	__OM uint32_t TXBUF;
	__IOM uint32_t RXCR0;
	__IOM uint32_t RXCR1;
	__IOM uint32_t RXCR2;
	__IM uint32_t RXBUF;
	__IM uint32_t RISR;
	__IM uint32_t MISR;
	__IOM uint32_t IMCR;
		该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h

	__OM uint32_t ICR; __OM uint32_t SRR; } csp_sio_t;	
ptTxCfg	typedef struct { uint8_t byD0Len; uint8_t byD1Len; uint8_t byDLLen; uint8_t byDHLen; uint8_t byDLLsq; uint8_t byDHHsq; uint8_t byTxDir; uint8_t byIdleLev; uint8_t byTxCnt; uint8_t byTxBufLen; uint8_t byInt; uint32_t wTxFreq; } csi_sio_tx_config_t;	初始化配置参数： byD0Len：D0长度，输出0 byD1Len：D1长度，输出1 byDLLen：DL长度，输出byDLLsq值 byDHLen：DH长度，输出byDHLsq值 byDLLsq：DL序列数值，用户定义 byDHHsq：DH序列数值，用户定义 byTxDir：TX数据输出方向，LSB/MSB byIdleLev：SIO引脚空闲时输出电平 byTxCnt：总的发送BIT数(0~256) byTxBufLen：TXBUF BIT数(0~16) byInter：中断，sio发送模式不支持 wTxFreq：发送每个BIT的频率 在sio.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

27.3.2 csi_sio_rx_init

*csi_error_t csi_sio_rx_init(csp_sio_t *ptSioBase, csi_sio_rx_config_t *ptRxCfg)*

27.3.2.1 功能描述

初始化SIO接收配置

27.3.2.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

ptRxCfg：SIO配置结构体指针，结构体定义详见csi_sio_rx_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
ptRxCfg	typedef struct { uint8_t byDebPerLen; uint8_t byDebClkDiv; uint8_t byTrgEdge; uint8_t byTrgMode; uint8_t bySpMode; uint8_t bySpBitLen; uint8_t bySpExtra; uint8_t byHithr; uint8_t byRxDir; uint8_t byRxCnt; uint8_t byRxBufLen; uint8_t byInt;	初始化配置参数： byDebPerLen：RX去抖周期 byDebClkDiv：去抖时钟分频 byTrgEdge：RX采样触发边沿 byTrgMode：RX采样触发模式 bySpMode：RX采样模式 bySpBitLen：每个BIT采样次数 bySpExtra：BIT提取策略 byHithr：BIT提取判定值(BIT=1/0) byRxDir：RX数据移入RXBUF方向 byRxCnt：总的接收BIT数(0~256) byRxBufLen：RXBUF BIT个数

	uint32_t wRxFreq; } csi_sio_rx_config_t;	(0~32) byInter：中断源 wRxFreq：接收每个BIT的频率 在sio.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

27.3.3 csi_sio_break_rst

*csi_error_t csi_sio_break_rst(csp_sio_t *ptSioBase, csi_sio_bklev_e eBkLev, uint8_t byBkCnt, bool bEnable)*

27.3.3.1 功能描述

配置接收BREAK复位

27.3.3.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

eBkLev：BREAK复位检测电平，高/低电平，枚举详见csi_sio_bklev_e。

byBkCnt：复位检测周期，即byBkCnt 个BIT采样长度内接收到的电平为eBkLev的配置。

bEnable：使能/禁止BREAK功能，ENABLE/DISABLE。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
eBkLev	typedef enum{ SIO_BKLEV_LOW = 0, SIO_BKLEV_HIGH }csi_sio_bklev_e;	接收BREAK检测电平： LOW：低电平 HIGH：高电平 在sio.h中定义
byBkCnt	uint8_t 类型数据，范围：1~0x1f； 复位检测周期 = byBkCnt *（单个BIT采样时间）	复位检测周期，即在这个周期内检测到的电平为eBkLev定义电平(高/低电平)。则复位接收模块

bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

27.3.4 csi_sio_timeout_rst

*csi_error_t csi_sio_timeout_rst(csp_sio_t *ptSioBase, uint8_t byToCnt ,bool bEnable)*

27.3.4.1 功能描述

配置接收采样超时复位

27.3.4.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

byToCnt：采样超时长度，即采样触发后，byToCnt个BIT采样长度内未检测到采样触发边沿。

bEnable：使能/禁止采样超时功能，ENABLE/DISABLE。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
byToCnt	uint8_t 类型数据，范围：1~0xff 采样超时时间 = byToCnt *（单个BIT采样时间）	采样超时，即采样触发后，在这个时间范围内未检测到后续的采样触发边沿，则采样超时，复位接收模块
bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

27.3.5 csi_sio_set_mode

*void csi_sio_set_mode(csp_sio_t *ptSioBase, csi_sio_wkmode_e eWorkMd)*

27.3.5.1 功能描述

配置SIO工作模式

27.3.5.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

eWorkMd：SIO工作模式，即接收/发送模式，枚举详见csi_sio_wkmode_e。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h

eWorkMd	typedef enum{
	SIO_SEND = 0, //sio (tx)send mode
	SIO_RECV = 1, //sio (rx)receive mode
	}csi_sio_wkmode_e;
		两种工作(传输)模式：
		SEND_MODE：发送模式
		RECV_MODE：接收模式
		在common.h中定义

27.3.6 csi_sio_int_enable

*void csi_sio_int_enable(csp_sio_t *ptSioBase, csi_sio_intsrc_e eIntSrc, bool bEnable)*

27.3.6.1 功能描述

使能SIO中断

27.3.6.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

eIntSrc：SIO中断源，枚举详见csi_sio_intsrc_e。

bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
eIntSrc	typedef enum { SIO_INTSRC_NONE = (0x00ul << 0), SIO_INTSRC_TXDNE = (0x01ul << 0), SIO_INTSRC_RXDNE = (0x01ul << 1), SIO_INTSRC_TXBUFEMPT= (0x01ul << 2), SIO_INTSRC_RXBUFFULL= (0x01ul << 3), SIO_INTSRC_BREAK = (0x01ul << 4), SIO_INTSRC_TIMEOUT = (0x01ul << 5) }csi_sio_intsrc_e;	SIO共6个中断源供用户选择，SIO_INTSRC_NONE表示不使用中断。

bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

27.3.7 csi_sio_set_buffer

*csi_error_t csi_sio_set_buffer(uint32_t *pwData, uint16_t hwLen)*

27.3.7.1 功能描述

配置SIO接收数据缓存(buffer)

27.3.7.2 参数/返回值说明

1. 参数

pwData：接收数据缓存指针，指向接收数据buffer首地址，SIO接收数据存放在该指针指向的buffer。

hwLen：接收数据缓存大小，即接收buffer长度，用户定义。

2. 返回值

无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
pwData	uint32_t 类型指针，指向接收数据buffer	指针，指向接收数据buffer首地址，buffer长度由用户根据实际需求来定义
hwLen	uint16_t 类型数据，接收数据buffer长度	用户定义

27.3.8 csi_sio_send

*int32_t csi_sio_send(csp_sio_t *ptSioBase, const uint32_t *pwData, uint16_t hwSize)*

27.3.8.1 功能描述

发送数据，支持轮询模式和中断模式。

27.3.8.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

pwData：发送数据缓存指针，指向发送缓存(buffer)首地址。

hwSize：要发送数据的长度。

2. 返回值

已发送数据长度/ CSI_UNSUPPORTED。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
pwData	uint32_t 类型指针，指向发送数据buffer	指针，指向发送数据buffer首地址，buffer长度由用户根据实际需求来定义
hwLen	uint16_t 类型数据，发送数据长度	长度用户定义
return value	int32_t 类型数据，具体返回值如下 轮训模式：已发送数据长度 中断模式：CSI_UNSUPPORTED，既不支持	发送数据仅支持轮训模式， 中断模式暂不支持

27.3.9 csi_sio_receive

*int32_t csi_sio_receive(csp_sio_t *ptSioBase, uint32_t *pwRecv, uint16_t hwLen)*

27.3.9.1 功能描述

获取SIO接收到的数据，只支持中断模式。

27.3.9.2 参数/返回值说明

1. 参数

ptSioBase：SIO寄存器结构体指针，指向SIO基地址，结构体定义详见csp_sio_t。

pwRecv：指向用户获取数据buffer首地址，即把接收到的数据读取到用户定义的buffer中。

hwSize：要获取的数据长度。

2. 返回值

获取到数据长度/ CSI_UNSUPPORTED。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptSioBase	csp_sio_t 类型指针，SIO0；请参阅27.3.1.2参数说明	该参数是一个外设寄存器结构体指针，固定为SIO0。SIO0的指针定义在devices.c,指针类型定义csp_sio.h
pwRecv	uint32_t 类型指针，指向用户获取数据buffer	指针，指向用户获取数据buffer首地址
hwLen	uint16_t 类型数据，要获取的数据长度	长度用户定义
return value	int32_t 类型数据，具体返回值如下 中断模式：获取的数据长度 轮训模式：CSI_UNSUPPORTED，即不支持	接收数据仅支持中断模式，轮训模式暂不支持

27.3.10 csi_sio_get_recv_status

csi_sio_state_e csi_sio_get_recv_status(void)

27.3.10.1 功能描述

获取SIO接收状态

27.3.10.2 参数/返回值说明

1. 参数

   无参数。

2. 返回值

   SIO接收状态，枚举定义详见csi_sio_state_e。

3. 返回值说明

返回值	说明	概述及其枚举定义位置

return value	typedef enum {
	SIO_STATE_IDLE = 0, //sio idle(rx/tx)
	SIO_STATE_RECV, //sio receiving
	SIO_STATE_SEND, //sio sending
	SIO_STATE_FULL, //sio receive complete(full)
	SIO_STATE_DONE, //sio send complete
	SIO_STATE_ERROR, //sio recv/send error
	SIO_STATE_TIMEOUT //sio receive timeout
	} csi_sio_state_e;
		UART工作状态枚举定义，接收用到状态说明：
		SIO_STATE_IDLE：空闲
		SIO_STATE_RECV：接收中
		SIO_STATE_FULL：接收完成
		SIO_STATE_ERROR：接收错误
		SIO_STATE_TIMEOUT：接收超时
		在sio.h中定义

27.3.11 csi_sio_clr_recv_status

void csi_sio_clr_recv_status(void)

27.3.11.1 功能描述

清除SIO接收状态，即设置接收状态为空闲状态。

27.3.11.2 参数/返回值说明

1. 参数

   无参数。

2. 返回值

   无返回值。

3. 参数/返回值说明

28 LCD

28.1 概述

MCU内嵌了一个LCD显示驱动模块，该LCD模块有2-COM x 32-SEG、 4-COM x 30-SEG 或8-COM x 26-SEG驱动能力，即最大能驱动64、120或208 LCD段（面板），每个段都可由LCD内部显示数据寄存器对应的位控制。CSI接口提供了LCD初始化、写数据、闪烁等相关配置和操作。

28.2 API列表

Table 24-1 SIO CSI接口函数

API	说明	函数位置
csi_lcd_init	初始化LCD显示配置	lcd.c
csi_lcd_start	开启（使能）LCD	
csi_lcd_int_enable	LCD中断配置	
csi_lcd_set_blink	LCD闪烁配置	
csi_lcd_write_data	写LCD显示数据（更新LCD显示数据）	
csi_lcd_gpio_init	LCD的GPIO管脚配置	

28.3 API详细说明

28.3.1 csi_lcd_init

*csi_error_t csi_lcd_init(csp_lcd_t *ptLcdBase, csi_lcd_config_t *ptLcdCFg)*

28.3.1.1 功能描述

初始化LCD

28.3.1.2 参数/返回值说明

1. 参数

ptLcdBase：LCD寄存器结构体指针，指向LCD基地址，结构体定义详见csp_lcd_t。

ptLcdCFg：LCD配置结构体指针，结构体定义详见csi_lcd_config_t。

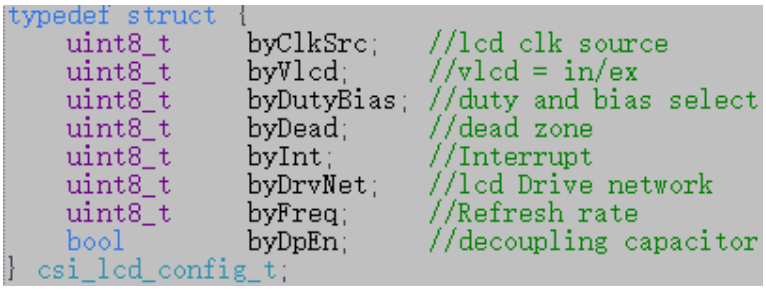
2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_lcd_t 类型指针，LCD，指向LCD的基地址	系统有一个LCD，即LCD，定义了对应的结构体指针LCD，指向LCD基地址。 在devices.c中定义

ptTxCfg	
	typedef struct {
	uint8_t byClkSrc; //lcd clk source
	uint8_t byVlcd; //vlcd = in/ex
	uint8_t byDutyBias; //duty and bias select
	uint8_t byDead; //dead zone
	uint8_t byInt; //Interrupt
	uint8_t byDrvNet; //lcd Drive network
	uint8_t byFreq; //Refresh rate
	bool byDpEn; //decoupling capacitor
	} csi_lcd_config_t;
		初始化配置参数：
		byClkSrc：时钟源选择
		byVlcd：VLCD选择，内部/外部
		byDutyBias：占空比、偏置
		byDead：死区
		byInter：中断
		byDrvNet：驱动电阻网络
		byFreq：帧刷新频率
		byDpEn：外部去耦电容使能

csi_error_t	csi_error_t 中定义值	在common.h中定义

28.3.2 csi_lcd_start

*void csi_lcd_start(csp_lcd_t *ptLcdBase)*

28.3.2.1 功能描述

开启（使能）LCD。

28.3.2.2 参数/返回值说明

1. 参数

   ptLcdBase：LCD寄存器结构体指针，指向LCD基地址，结构体定义详见csp_lcd_t。

2. 返回值：无

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_lcd_t 类型指针，LCD；请参阅28.3.1.2参数说明	在devices.c中定义

28.3.3 csi_lcd_int_enable

*void csi_lcd_int_enable(csp_lcd_t *ptLcdBase, csi_lcd_intsrc_e eIntSrc, bool bEnable)*

28.3.3.1 功能描述

配置LCD中断

28.3.3.2 参数/返回值说明

1. 参数

   ptLcdBase：LCD寄存器结构体指针，指向LCD基地址，结构体定义详见csp_lcd_t。

   eIntSrc：LCD中断源，高/低电平，枚举定义详见csi_lcd_intsrc_e。

   bEnable：使能/禁止中断，ENABLE/DISABLE。

2. 返回值：无。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptLcdBase	csp_lcd_t 类型指针，LCD；请参阅28.3.1.2参数说明	在devices.c中定义
eIntSrc	typedef enum { LCD_INTSRC_NONE = (0x00ul << 0), LCD_INTSRC_SOF = (0x01ul << 1), LCD_INTSRC_UDD = (0x01ul << 3), }csi_lcd_intsrc_e;	SOF: 帧开始 UDD: 显示更新完成 在lcd.h中定义
bEnable	bool 类型数值，ENBALE/DISABLE	ENBALE：使能 DISABLE：禁止 在common.h中定义

28.3.4 csi_lcd_set_blink

*csi_error_t csi_lcd_set_blink(csp_lcd_t *ptLcdBase, csi_lcd_blink_md_e eBkMode, csi_lcd_blink_fre_e eBkFre, uint16_t hwF2Fre)*

28.3.4.1 功能描述

LCD闪烁配置

28.3.4.2 参数/返回值说明

1. 参数

ptLcdBase：LCD寄存器结构体指针，指向LCD基地址，结构体定义详见csp_lcd_t。

eBkMode：闪烁模式，枚举定义详见csi_lcd_blink_md_e。

eBkFre：闪烁频率，枚举定义详见csi_lcd_blink_fre_e。

hwF2Fre：闪烁频率，eBkFre = LCD_BLINK_FRE_F2,时的闪烁频率，其余模式无效。

2. 返回值

CSI_OK：配置成功。

CSI_ERROR：配置失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptLcdBase	csp_lcd_t 类型指针，LCD：请参阅28.3.1.2参数说明	在devices.c中定义

eBkMode	typedef enum
	{
	LCD_BLINK_DIS = 0x00ul, //LCD none interrupt
	LCD_BLINK_SEG8_COM0 = 0x01uL, //SEG8 of COM0
	LCD_BLINK_SEG8 = 0x02ul, //SEG8 of all COM
	LCD_BLINK_ALL = 0x03ul //all SEG and COM
	}csi_lcd_blink_md_e;
		在lcd.h中定义

eBkFre	typedef enum { LCD_BLINK_FRE_DIV8 = 0, LCD_BLINK_FRE_DIV16, LCD_BLINK_FRE_DIV32, LCD_BLINK_FRE_DIV64, LCD_BLINK_FRE_DIV128, LCD_BLINK_FRE_DIV256, LCD_BLINK_FRE_DIV512, LCD_BLINK_FRE_F2, }csi_lcd_blink_fre_e;	在lcd.h中定义
hwF2Fre	uint16_t 类型数据，闪烁频率	eBkFre =LCD_BLINK_FRE_F2时闪烁频率值为此参数，其余模式无效。
csi_error_t	csi_error_t 中定义值	在common.h中定义

28.3.5 csi_lcd_write_data

*void csi_lcd_write_data(csp_lcd_t *ptLcdBase, uint8_t *pbyData, uint8_t byStaPos, uint8_t bySegNum)*

28.3.5.1 功能描述

写（更新）LCD显示数据。

28.3.5.2 参数/返回值说明

1. 参数

   ptLcdBase：LCD寄存器结构体指针，指向LCD基地址，结构体定义详见csp_lcd_t。

   pbyData：指向写入LCD数据buffer指。

   byStaPos：LCD的SEGx开始位置，共32个SEG0~SEG31，范围0~31。

   bySegNum：LCD的SEG数量，范围1~32.

2. 返回值：无。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptLcdBase	csp_lcd_t 类型指针，LCD；请参阅28.3.1.2参数说明	在devices.c中定义
pbyData	uint8_t 类型指针，指向写入(更新)LCD数据buffer	
byStaPos	uint8_t 类型数据，LCD的SEGx开始位置	起始位置开始，连续写入
bySegNum	uint8_t 类型数据，LCD的SEG数量	连续写入的个数

28.3.6 csi_lcd_gpio_init

void csi_lcd_gpio_init(uint32_t wSegMask, uint8_t byComMask)

28.3.6.1 功能描述

LCD的GPIO配置，包括COM和SEG，支持2COMx32SEG/4COMx30SEG/8COMx26SEG

28.3.6.2 参数/返回值说明

1. 参数

wSegMask：LCD的SEG MASK值，范围0x01~0-xFFFFFFFF；比如选择16个SEG，wSegMask = 0xFFFF；24个，wSegMask = 0xffffff。

byComMask：LCD的COM MASK值，围0x01~0-xFF；比如选择2COM，byComMask = 0x03；4COM，byComMask = 0x0F。

2. 返回值：无

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
wSegMask	uint32_t 类型数据，所选SEG的MASK值	SEG的范围：SEG0~31
byComMask	uint8_t 类型数据，所选COM的MASK值	COM的范围：COM0~7

29 CMP

29.1 概述

模拟比较器通过比较两个模拟输入电压量，输出一个数字标志用以表示两个输入量的幅值大小，是模拟电路和数字电路的的一种转换接口。模拟比较器在数模混合电路中作为非常重要的一种构建单元，可以提供独立于程序运行的模拟功能。

ATP CSI 接口 CMP 的设计中，提供丰富的配置及其操作。包含基本配置功能、数字滤波配置功能及窗口滤波控制功能等。

29.2 API列表

Table 29-1 CMP CSI接口函数

API	说明	函数位置
csi_cmp_int_enable	使能CMP中断	sio.c
csi_cmp_init	初始化CMP基本功能	
csi_cmp_start	开始CMP功能	
csi_cmp_stop	停止CMP功能	
csi_cmp_dflt_config	配置CMP数字滤波功能	
csi_cmp_wfcr_config	配置CMP窗口滤波功能	
csi_cmp_set_evtrg	同步触发输出端口控制	
csi_cmp_get_out	获取输出状态	
csi_cmp_int_clear	清除中断状态	
csi_cmp_get_misr	获取中断状态	
csi_cmp_lpwken_enable	使能CMP deepsleep唤醒模式	

29.3 API详细说明

29.3.1 csi_cmp_int_enable

*void csi_cmp_int_enable(csp_cmp_t *ptCmpBase, csi_cmp_intsrc_e eIntSrc,bool bEnable)*

29.3.1.1 功能描述

使能CMP中断

29.3.1.2 参数/返回值说明

1. 参数

   ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

   eIntSrc：中断模式，枚举定义详见csi_cmp_intsrc_e。

   bEnable：启用中断或禁止中断。

2. 返回值：无返回值

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
eIntSrc	typedef enum { CMP_INTSRC_NULL = (0x00ul << 0), CMP_INTSRC_EDGEDET = (0x01ul << 0), CMP_INSRCT_RAWDET = (0x01ul << 16), }csi_cmp_intsrc_e;	CMP的中断类型 csi_cmp_intsrc_e在cmp.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能中断 DISABLE：禁止中断 在common.h中定义

29.3.2 csi_cmp_init

*csi_error_t csi_cmp_init(csp_cmp_t *ptCmpBase,csi_cmp_config_t *ptCmpCfg)*

29.3.2.1 功能描述

初始化CMP基本功能

29.3.2.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

ptCmpCfg：CMP基本配置结构体指针，结构体定义详见csi_cmp_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
ptCmpCfg	typedef struct { uint8_t byNsel; uint8_t byPsel; uint8_t byPhystpol; uint8_t byPhystsel; uint8_t byPolarity; uint8_t byCpoSel; uint32_t wInt; }csi_cmp_config_t;	在cmp.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

29.3.3 csi_cmp_start

*void csi_cmp_start(csp_cmp_t *ptCmpBase)*

29.3.3.1 功能描述

开始CMP功能

29.3.3.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

2. 返回值：无

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义

29.3.4 csi_cmp_stop

*void csi_cmp_stop(csp_cmp_t *ptCmpBase)*

29.3.4.1 功能描述

停止CMP功能

29.3.4.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

2. 返回值：无

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义

29.3.5 csi_cmp_dflt_config

*csi_error_t csi_cmp_dflt_config(csp_cmp_t *ptCmpBase,csi_cmp_dflt_config_t *ptCmpDfltCfg)*

29.3.5.1 功能描述

配置CMP数字滤波功能

29.3.5.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

ptCmpDfltCfg：CMP数字滤波配置结构体指针，结构体定义详见csi_cmp_dflt_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptSioBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
ptCmpDfltCfg	typedef struct { uint8_t byDepth1; uint8_t byDivn1; uint8_t byDivm1; uint8_t byDepth2; uint8_t byDivn2; uint8_t byDivm2; }csi_cmp_dflt_config_t;	在cmp.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

29.3.6 csi_cmp_wfcr_config

*csi_error_t csi_cmp_wfcr_config(csp_cmp_t *ptCmpBase,csi_cmp_wfcr_config_t *ptCmpWfcrCfg)*

29.3.6.1 功能描述

配置CMP窗口滤波功能

29.3.6.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

ptCmpWfcrCfg：CMP窗口滤波配置结构体指针，结构体定义详见csi_cmp_wfcr_config_t。

2. 返回值

CSI_OK：初始化成功。

CSI_ERROR：初始化失败。

3. 参数说明

参数	说明	概述及其枚举/结构体定义位置
ptSioBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
ptCmpWfcrCfg	typedef struct { uint8_t byWfalign; uint8_t byWoset; uint8_t byClkDiv; uint8_t byDcnt; uint16_t hwWcnt; }csi_cmp_wfcr_config_t;	在cmp.h中定义
csi_error_t	csi_error_t 中定义值	在common.h中定义

29.3.7 csi_cmp_set_evtrg

*void csi_cmp_set_evtrg(csp_cmp_t *ptCmpBase, bool bEnable,csi_eve_sel_e eEveSel)*

29.3.7.1 功能描述

同步触发输出端口控制

29.3.7.2 参数/返回值说明

1. 参数

   ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

   bEnable：启用中断或禁止触发同步功能。

   eEveSel：触发模式，枚举定义详见csi_eve_sel_e。

2. 返回值

   无返回值。

3. 参数说明

参数	说明	概述及其结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
eEveSel	typedef enum { EVE_DOWN = 0x00, EVE_UP, EVE_DOWN_UP, EVE_UP1 }csi_eve_sel_e;	CMP的触发类型 csi_eve_sel_e在cmp.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能同步触发 DISABLE：禁止同步触发 在common.h中定义

29.3.8 csi_cmp_get_out

*uint8_t csi_cmp_get_out(csp_cmp_t *ptCmpBase,uint8_t byOutCh)*

29.3.8.1 功能描述

获取输出状态

29.3.8.2 参数/返回值说明

1. 参数

   ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

byOutCh：输出通道选择。

2. 返回值

输出状态,0：低电平；1：高电平。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
byOutCh	uint8_t 类型数据，通道源选择0：CMP0;1:CMP1	输出通道源选择
return value	uint8_t 类型数据，具体返回值如下 返回值,0：低电平；1：高电平。	输出状态

29.3.9 csi_cmp_int_clear

*void csi_cmp_int_clear(csp_cmp_t *ptCmpBase,csi_cmp_intsrc_e eIntMode)*

29.3.9.1 功能描述

清除中断状态。

29.3.9.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

eIntMode：中断模式，枚举定义详见csi_cmp_intsrc_e。

2. 返回值

无返回值。

3. 参数/返回值说明

参数/返回值	说明	概述及其结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义

eIntMode	typedef enum { CMP_INTSRC_NULL = (0x00ul << 0), CMP_INTSRC_EDGEDET = (0x01ul << 0), CMP_INSRCT_RAWDET = (0x01ul << 16), }csi_cmp_intsrc_e;	CMP的中断类型 csi_cmp_intsrc_e在cmp.h中定义

29.3.10 csi_cmp_get_misr

*uint32_t csi_cmp_get_misr(csp_cmp_t *ptCmpBase)*

29.3.10.1 功能描述

获取中断状态

29.3.10.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

2. 返回值

uint32_t类型的返回数据，返回的为中断状态。

3. 返回值说明

返回值	说明	概述及其枚举定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
Return value	uint32_t类型的数据，中断状态包含以下四种： #define CMP0_EDGEDET0_INT (1<<0) #define CMP1_EDGEDET1_INT (1<<1) #define CMP0_RAWDET0_INT (1<<16) #define CMP1_RAWDET1_INT (1<<17)	返回中断状态

29.3.11 csi_cmp_lpwken_enable

*void csi_cmp_lpwken_enable(csp_cmp_t *ptCmpBase, bool bEnable)*

29.3.11.1 功能描述

使能CMP deepsleep唤醒模式

29.3.11.2 参数/返回值说明

1. 参数

ptCmpBase：CMP寄存器结构体指针，指向CMP基地址，结构体定义详见csp_cmp_t。

bEnable：启用或禁止deepsleep唤醒。

2. 返回值

无返回值。

3. 参数/返回值说明

参数/返回值	说明	概述及其枚举/结构体定义位置
ptCmpBase	csp_cmp_t 类型指针，指向CMP的基地址	CMP寄存器结构体指针类型 csp_cmp_t在csp_cmp.h中定义
bEnable	bool类型数值，ENBALE/DISABLE	ENBALE：使能deepsleep唤醒 DISABLE：禁止deepsleep唤醒 在common.h中定义

30 LED

APT32F110x 集成了 LED 驱动模块，最大可以驱动 8（SEG）x10（COM）的 8 段数码管。通过不同时间槽分割来点亮 10 个 8 段数码管。在每个扫描时间槽（单独一个 COM）内，SEGMENT 输出口会将相应的预设值输出到 IO 口上，COM 扫描的通道可以自由定义。该 LED 控制器仅用于控制共阴 LED 数码管。

30.1 API列表

Table 26-1 LED CSI接口函数

API	说明	函数位置
csi_led_init	LED初始化函数。	led.c
csi_led_lighton	LED输出驱动信号。	
csi_led_lightoff	LED停止输出驱动信号。	
csi_led_write_data	往不同的数码管写驱动内容。	
csi_led_set_blink_pattern	设置闪烁方式。	
csi_led_int_enable	中断使能/关闭控制。	

30.2 API详细说明

30.2.1 csi_led_init

*csi_error_t csi_led_init(csp_led_t *ptLedBase, csi_led_config_t *tLedCfg)*

30.2.1.1 功能描述

LED 的初始化包括使用哪些 COM 口、一个 COM 显示周期的长度、每个 COM 有效输出时长、相邻 COM 输出间隔、是否使能中断等。

如果一个 COM 显示周期的长度或相邻 COM 输出间隔的设置太大，超过 spec 的范围，将会返回 CSI_ERROR。

30.2.1.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h

tLedCfg	指向LED设置结构体的指针。
	`typedef struct csi_led_config {`
	`  uint8_t byClk; //clk configure`
	`  uint8_t byBrt; //brightness configure`
	`  uint16_t hwComMask; //COM enable`
	`  uint8_t byOnTime; //scanning timing: COM on cycles`
	`  uint8_t byBreakTime; //scanning timing: cycles between COMs`
	`  uint8_t byInt; //Interrupt Source`
	`}csi_led_config_t;`
		led.h

30.2.1.3 返回值说明

返回值类型	说明	位置
csi_error_t	如果一个 COM 显示周期的长度或相邻 COM 输出间隔的设置太大，超过 spec 的范围，将会返回 CSI_ERROR。否则返回 CSI_OK。	common.h

30.2.2 csi_led_lighton

*void csi_led_lighton(csp_led_t *ptLedBase)*

30.2.2.1 功能描述

LED 启动自动扫描。

30.2.2.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h

30.2.2.3 返回值说明（无）

30.2.3 csi_led_lightoff

*void csi_led_lighton(csp_led_t *ptLedBase)*

30.2.3.1 功能描述

LED 停止自动扫描。

30.2.3.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h

30.2.3.3 返回值说明（无）

30.2.4 csi_led_write_data

*void csi_led_write_data(csp_led_t *ptLedBase, uint8_t byCom, uint8_t byData)*

30.2.4.1 功能描述

往不同的数码管写驱动内容。

30.2.4.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h
byCom	选择对应的8单元数码管	

byData	写入的内容。
	`uint8_t g_byLedData[11] = {0x3f, 0x6, 0x5b, 0x4f, 0x66, 0x6d, 0x7d, 0x07,0x7f, 0x6f, 0x80} ; //没有小数点时需要显示0~9时的写入值。`
	`uint8_t g_byLedData8[10] = {0x7f, 0x7f, 0x7f, 0x7f, 0x7f, 0x7f, 0x7f, 0x7f,0x7f, 0x7f} ; //有小数点时需要限制0~9时的写入值。`	

30.2.4.3 返回值说明（无）

30.2.5 csi_led_set_blink_pattern

*void csi_led_set_blink_pattern(csp_led_t *ptLedBase, uint16_t hwOnMsk)*

30.2.5.1 功能描述

设置闪烁的方式，即使能哪几个数码管（COM）的扫描。

30.2.5.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h
hwOnMsk	开启扫描的COM口，对应的位置1。如果，hwOnMsk = 0x3，即只有COM0和COM1开启扫描，其余都关闭扫描（禁止COM在扫描周期内输出）。	

30.2.5.3 返回值说明（无）

30.2.6 csi_led_int_enable

*void csi_led_int_enable(csp_led_t *ptLedBase, csi_led_intsrc_e eIntSrc, bool bEnable)*

30.2.6.1 功能描述

使能/禁止 LED 中断。如果是“使能”操作，会判断当前是否已经有中断使能，如果没有，则额外打开 NVIC 端的中断使能，否则仅仅打开 IP 端的中断使能。如果是“禁止”操作，则会判断当前中断是否为 MISR 仅存的中断能源，如果是则关闭 NVIC 端的中断使能，如果不是则仅仅关闭 IP 端的中断使能。

30.2.6.2 参数说明

参数	说明	位置
ptLedBase	指向LED控制寄存器结构体的指针，用于进行寄存器操作。	csp_led.h

csi_led_intsrc_e	LED的中断枚举。
	`typedef enum {`
	`  LED_INTSRC_NONE = 0,`
	`  LED_INTSRC_ICEND = 0x1<<0,`
	`  LED_INTSRC_IPEND = 0x1<<1,`
	`}csi_led_intsrc_e;`
		led.h

bool	ENABLE/DISABLE	

30.2.6.3 返回值说明（无）