	V1.0
	202012

APT32F102

CSI

IP

CDKV2.6

Revision History

V1.0	2020-12		

1.

APT32F102x CSI Chip Standard Interface

2.

CSI csp csp CSI sys
CSI sys API

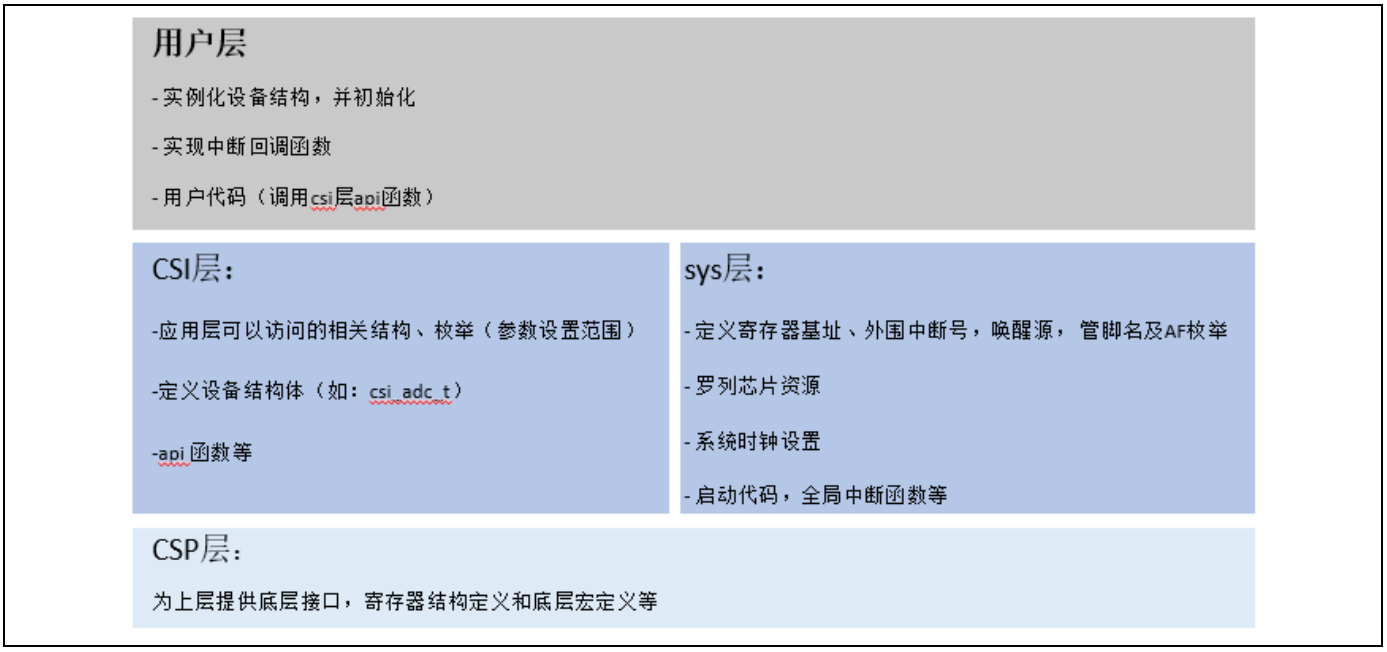


Figure 1

CDK

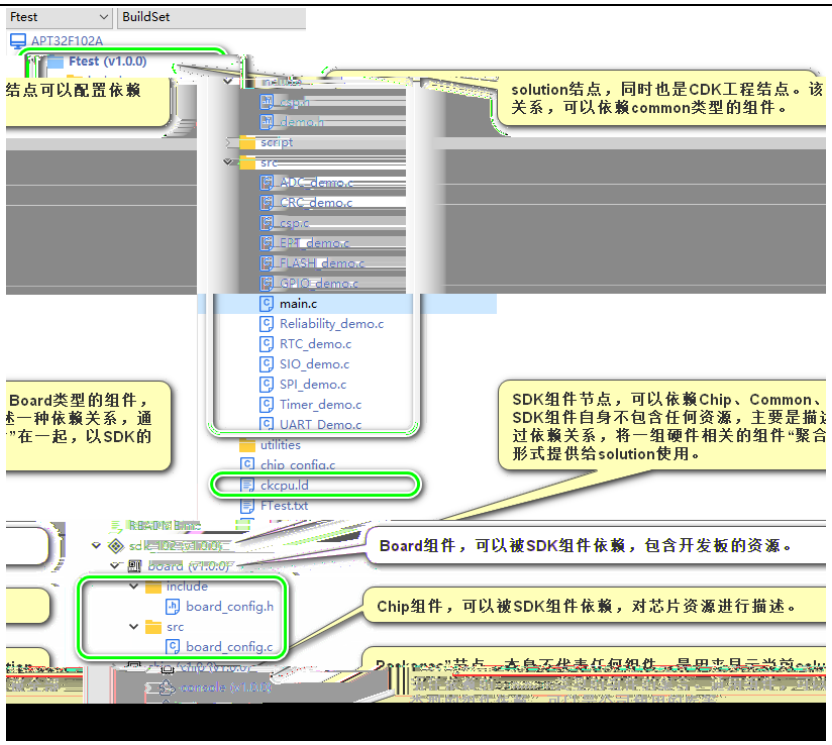


Figure 2

1

2 CDK2.6 ***CDK V2.6 .pdf***

3.

IP

sdk_102 -> board -> include -> board_config.h

\board\include

```
/* example pin manager */

#define CONSOLE_IDX                     0
#define CONSOLE_TXD                     PA00
#define CONSOLE_RXD                     PA01
#define CONSOLE_TXD_FUNC                PA01_UART0_TX
#define CONSOLE_RXD_FUNC                PA00_UART0_RX

#define EXI_PIN                         PA09
#define EXI_PIN_FUNC                    PIN_FUNC_INPUT

#define SPI_IDX                         0
#define SPI_MOSI_PIN                    PA014
#define SPI_MISO_PIN                    PA015
#define SPI_NSS_PIN                     PB05
#define SPI_SCK_PIN                     PB04
#define SPI_MOSI_PIN_FUNC               PA014_SPI_MOSI
#define SPI_MISO_PIN_FUNC               PA015_SPI_MISO
#define SPI_NSS_PIN_FUNC                PB05_SPI_NSS
#define SPI_SCK_PIN_FUNC                PB04_SPI_SCK
```

Figure 3

xxx_IDX IP 0 0 UART0 g_soc_info[ ]
sdk_102 -> chip -> sys > devices.c

```
const csi_perip_info_t g_soc_info[] = {

    {CK801_ADDR_BASE,           CORET_IRQn,             0,    DEV_CORET_TAG},
    {APB_SYS_BASE,              SYSCON_IRQn,            0,    DEV_SYSCON_TAG},
    {APB_IFC_BASE,              IFC_IRQn,               0,    DEV_IFC_TAG},
    {APB_ADC0_BASE,             ADC_IRQn,               0,    DEV_ADC_TAG},
    {APB_EPT0_BASE,             EPT0_IRQn,              0,    DEV_EPT_TAG},
```

Figure 4

4.

sdk_102 -> board -> src -> board_config.c

\board\src

```
/// system clock configuration parameters to define source, source freq(if selectable), sdiv and pdiv
const system_clk_config_t g_tSystemClkConfig[1] = {

    {SRC_HFOSC, HFOSC_48M_VALUE, SCLK_DIV2, PCLK_DIV1}
    //{SRC_EMOSC, 20000000, SCLK_DIV1, PCLK_DIV2}
    //{SRC_IMOSC, IMOSC_5M_VALUE, SCLK_DIV1, PCLK_DIV1}
    //{SRC_HFOSC, HFOSC_48M_VALUE, SCLK_DIV2, PCLK_DIV1}
    //{SRC_IMOSC, IMOSC_5M_VALUE, SCLK_DIV1, PCLK_DIV1}
};
```

Figure 5

g_tSystemClkConfig 4 SCLK PCLK sys_clk.h
sdk_102 -> chip -> sys

5.

5.1

__main() SRAM main

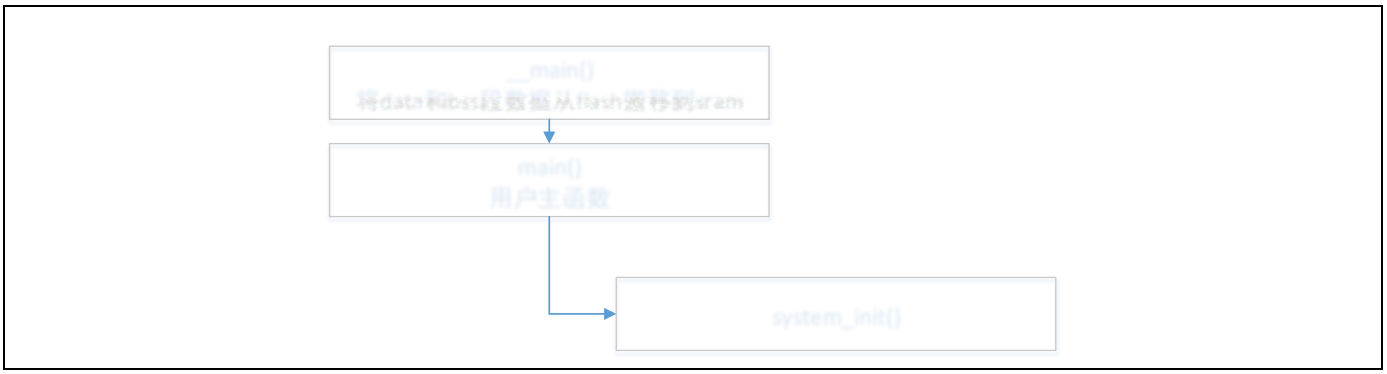


Figure 6

main system_init()

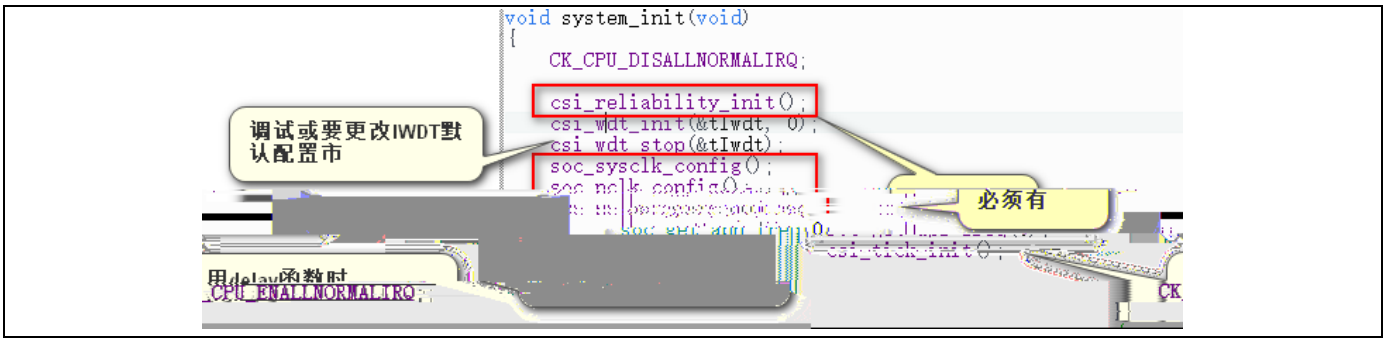


Figure 7 system_init()

5.2

board_config.h

```
csi_pin_set_mux(ADC_PIN, ADC_PIN_FUNC);
```

Figure 8

5.3

5.3.1

CSI

ADC

●

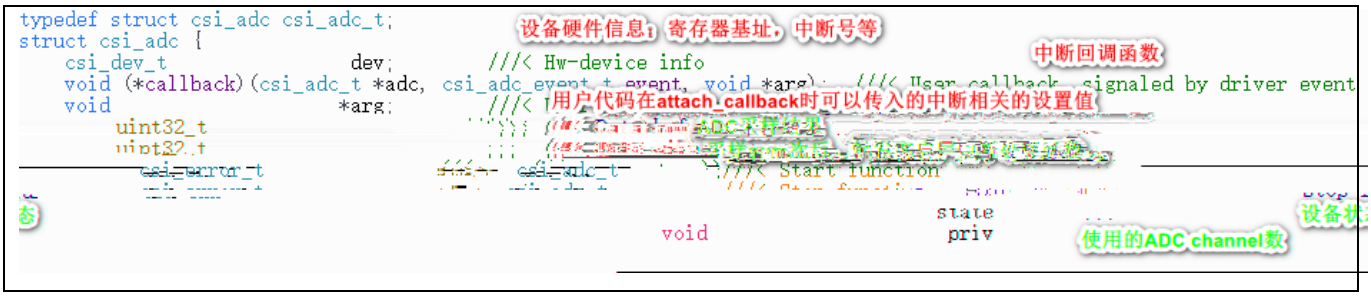


Figure 9 ADC

- CSI_xxx_init

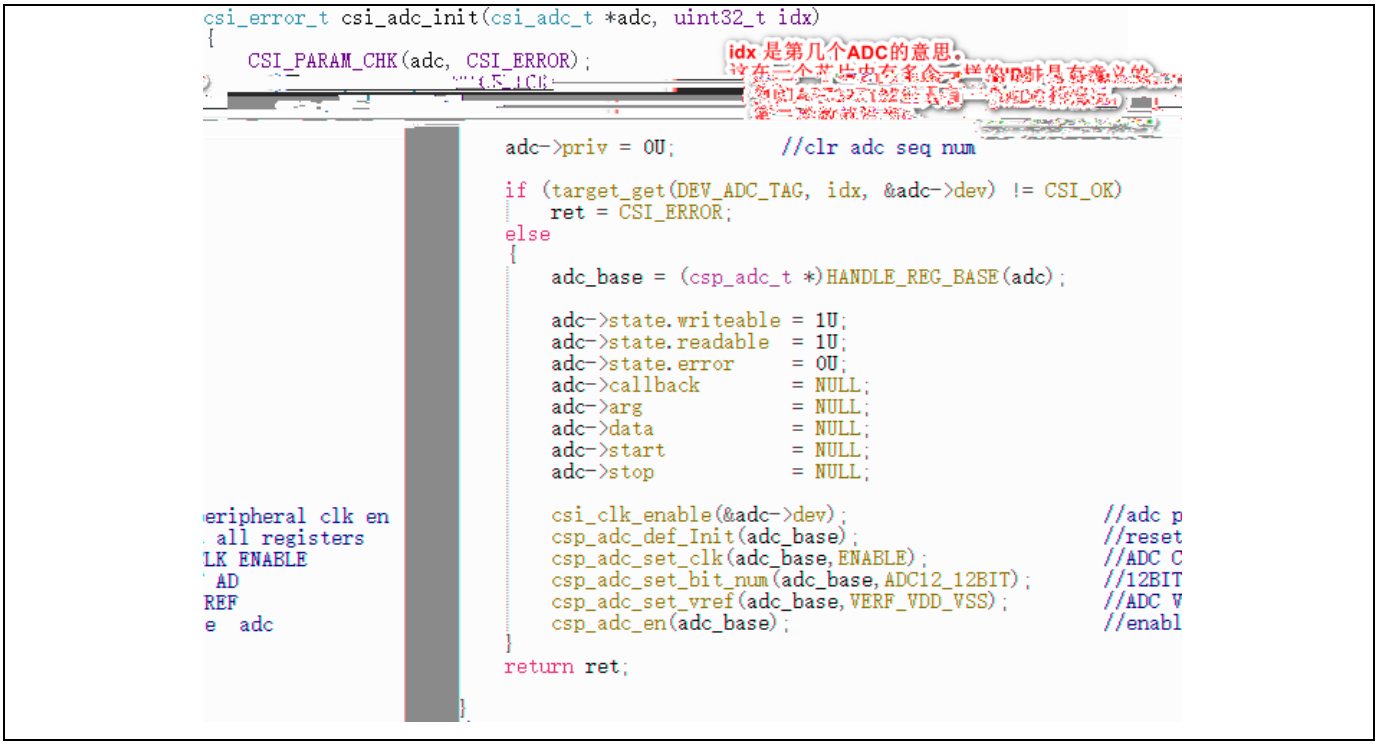


Figure 10 ADC

5.3.2

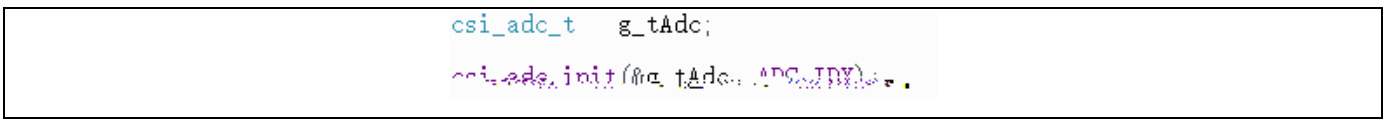


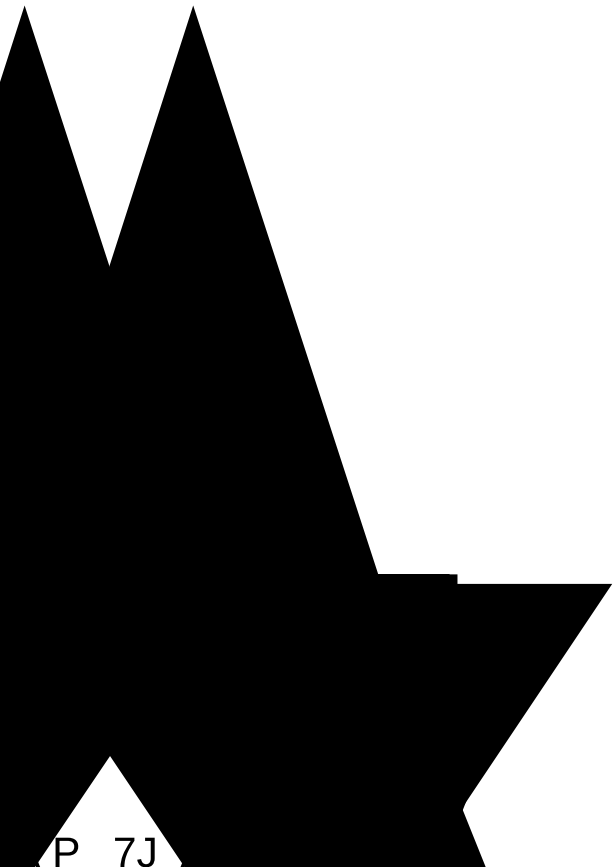
Figure 11 ADC

ADC IP

APT32F102 ADC 0 g_soc_info[ ]

sdk_102 -> chip -> sys > devices.c

6.



```
static void apt_adc_irqhandler(void *args)
{
    csi_adc_t *adc      = (csi_adc_t *)args;
    csp_adc_t *adc_base = (csp_adc_t *)HANDLE_REG_BASE(adc);

    uint8_t i;
    uint32_t wChnlNUm = (uint32_t)adc->priv;

    if(adc->data != NULL)
    {
        if(adc->num > 0)
        {
            for(i = 0; i < wChnlNUm; i++)
            {
                if(csp_adc_get_status(adc_base,ADC12_SEQ(i)))
                {
                    *(adc->data + i*s_byBufLen + adc->num- 1) = csp_adc_get_data(adc_base, i);
                    csp_adc_clr_status(adc_base,ADC12_SEQ(i));
                }
            }
            adc->num -- ;
        }

        if(adc->num == 0)
        {
            if (adc->callback)
            {
                adc->callback(adc, ADC_EVENT_CONVERT_COMPLETE, adc->arg);
                adc->state.readable = 1U;
            }
        }
    }
}
```

Figure 13 adc_irqhandler

6.1.2

1. attach_callback

```
CSI_adc_attach_callback(&g_tAdc, user_adc_event, (void *)arg);
```

2.

```
void user_adc_event(CSI_adc_t *adc, CSI_adc_event_t event, void *arg)
```

arg .c

7. Q & A

7.1 Q1 如果CSI的代码架构不能满足系统对实时性的要求怎么办

A1

1) ETCB

ETCB

2)

do_irq Figure 11
csp_ifc_irq_handler

sdk_102 -> chip -> sys -> interrupt.c
interrupt.c csp.h GPIOA0->CONLR = 0x22222222

3) csp.h csp

7.2 Q2 如果CSI现有代码 API 不能满足特定的应用场合 怎么办

A2 CSI API API
csp.h csp

7.3 Q3：CSI代码可以在老版本CDK上运行吗？

A3 CSI CDK CDK2.6
CSI CDK CDK

8. 1 CDKV2.6

CDK ***CDK V2.6*** ***.pdf***

8.1

CDK2.6 virtual folder

8.2

CDK2.6 flash

Figure 14