文档版本	V1.0.0
发布日期	20221026

APT32F110x 基于 CSI 库 IFC 应用指南

目录

1 概述

本文介绍了在APT32F110x中使用IFC的应用范例。

2. 适用的硬件

该例程使用于 APT32F110x 系列学习板

3. 应用方案代码说明

基于 APT32F110x 完整的库文件系统，可以对 IFC 进行配置。

3.1 IFC 模块介绍

APT32F110x 系列片上带有 64K/32K 字节的闪存(PROM)，支持通过 ISP 来更新闪存内容。芯片上电后，CPU 从 PROM 取指令并且执行。APT32F110X 系列还支持额外的数据闪存(DROM)存储空间，让用户在掉电之前存储一些应用程序需要的数据。

- **主要特性：**

  1) 程序闪存(PROM)大小：64K/32K Bytes

  2) 数据闪存(DROM)大小：2K Bytes

  3) 编程支持 ISP 模式和专用的工具模式

  4) 页大小：256 Bytes（PROM），64 Bytes（DROM）

  5) 可擦除单元：页

  6) 可靠性：PROM 和 DROM 都为 100,000 次

  7) 可自定义的选项（称为 User Option）支持 iWDT 使能和禁止，配置复位管脚

  8) 支持各种保护：调试接口保护，硬件保护和读保护

- **硬件配置：**

APT32F110x 系列闪存由程序存储单元(PROM)，数据存储单元(DROM)，用户配置单元(User

Option)，保护选项和客户信息区域构成。PROM 有 256/128 个页空间，每页有 256 字节。最小的擦除和烧写单元为页空间，用户只可以对整个页空间进行擦除或者烧写，不能擦除或者烧写某个指定的字节(Word)。

- **注意事项：**

1. 写数据时，起始地址必须是 4 的倍数

2. 在同一页中写数据时，若起始地址并非该页的开始地址，则该地址前面的数据会被擦除掉

3. 写闪存操作不要使用并行模式

4. 闪存控制器支持最大 16MHz 系统频率下的 0-wait 读取。当频率超过 16MHz 时，CPU 读取闪存时需要增加额外的等待周期。不同的 CPU 频率下，WAIT 和 SPEED 的值参考如下。

	WAIT	SPEED
24MHz < SYSCLK ≤ 48MHz	2	1
16MHz < SYSCLK ≤ 24MHz	1	1
SYSCLK ≤ 16MHz	0	0

图 3.1.1 等待周期

3.2 IFC 写数据

选择内部主频 48MHz 作为系统时钟，可在 user_demo.c 文件中 ifc_program ()函数进行配置。从 0xfe78 地址(PROM)开始，写入 3 个 word 数据；从 0x10000078 地址(DROM)开始，写入 5 个 word 数据。

```
void ifc_program(void)
{   csi_error_t tRet;



    tRet = csi_ifc_program(IFC, 0xfef8, wWriteData, 3);
    if (tRet == CSI_ERROR)
            my_printf("program fail!\n");
    else
            my_printf("program pass!\n");

    tRet = csi_ifc_program(IFC, 0x10000078, wWriteData, 5);
    if (tRet == CSI_ERROR)
```

```
            my_printf("program fail!\n");
    else
            my_printf("program pass!\n");
}
```

- **代码说明：**

  csi_ifc_program(): ---- 往 Flash 区域写入内容，带校验功能。支持 PFLASH 和 DFLASH。

- **函数参数说明：**

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

  ptIfcBase：指向IFC控制寄存器结构体的指针，用于进行寄存器操作

  wAddr：操作Flash的目标首址

  pwData:指向需要写入数据的首地址

  wDataNum:写入数据的长度，以 word 为单位。

- **数据验证：**

0x10000058	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x10000078	0xcdcdcdcd	0x23232323	0x45454545	0x67676767	0x89898989
0x10000098	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

图 3.2.1 DROM 写入数据

0x0000fee8	0x00000000	0x00000000	0x00000000	0x00000000
0x0000fef8	0xcdcdcdcd	0x23232323	0x45454545	0x00000000
0x0000ff08	0x00000000	0x00000000	0x00000000	0x00000000
0x0000ff18	0x00000000	0x00000000	0x00000000	0x00000000

图 3.2.2 PROM 写入数据

3.3 IFC 读数据

选择内部主频 48MHz 作为系统时钟，可在 user_demo.c 文件中 ifc_read()函数进行配置。从 0x10000078 地址(DROM)开始，读出 5 个 word 数据。

```
void ifc_read(void)
{
    csi_ifc_read(IFC,0x10000078, wReadBuf, 5);
    my_printf("read flash data: 0x%x, 0x%x \n", wReadBuf[0], wReadBuf[1]);
```

```
}
```

- **代码说明：**

  ifc_read(): ------ 读取 Flash 区域的内容，支持 DROM 和 PROM。

- **函数参数说明：**

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

  ptIfcBase: 指向IFC控制寄存器结构体的指针，用于进行寄存器操作。

  wAddr:读取 Flash 数据的首地址。

  wData:指向目标数据首地址指针。

  wDataNum:读取数据长度。

- **数据验证：**

Expression	Value	Type
⊟ wReadBuf	[10]	uint32_t [10]
0	0xcdcdcdcd	uint32_t
1	0x23232323	uint32_t
2	0x45454545	uint32_t
3	0x67676767	uint32_t
4	0x89898989	uint32_t
5	0x00000000	uint32_t
6	0x00000000	uint32_t
7	0x00000000	uint32_t
8	0x00000000	uint32_t
9	0x00000000	uint32_t

图 3.3.1 IFC 读取数据

4. 程序下载和运行

1. 将目标板与仿真器连接，分别为 VDD、SCLK、SWIO、GND。
2. 程序编译后仿真运行。
3. 可在 CDK 的 Debug 模式下查看数据是否正确，例如图 3.2.1、图 3.2.2、图 3.3.1 所示。。