通用 FSMC 控制器

2024.08.29

一 修订

版本	日期	编辑人	内容
1.00	2024.08.29	陈家耀	创建了第一个正式版本
1.10	2024.08.30	陈家耀	修复了 AXI-Lite 从接口中的 BUG

二 简介和特性

通用 FSMC 控制器带有 AHB-Lite 或 AXI-Lite 从接口，用于实现系统总线与外部存储映射的连接，可挂载外部存储器（如 PSRAM、NOR/NAND FLASH、SRAM 等）和 TFT LCD 显示屏。通用 FSMC 控制器具有以下特性：

- **带有 AHB-Lite 或 AXI-Lite 从接口**
- **运行时可配置的 FSMC 时序参数（地址建立、数据建立、数据保持周期数）**
- **16 位 FSMC 接口**

通用 FSMC 控制器结构简单、资源消耗少，其组成如图 2-1 所示。**AHB 或 AXI 从接口**分为两个域，一个是用于配置运行时参数的寄存器片，另外一个是 FSMC 存储映射区。位于 FSMC 存储映射区的 AHB 或 AXI 传输，其传输参数将被锁存，交由 **FSMC 控制器**来驱动外部的存储映射设备。

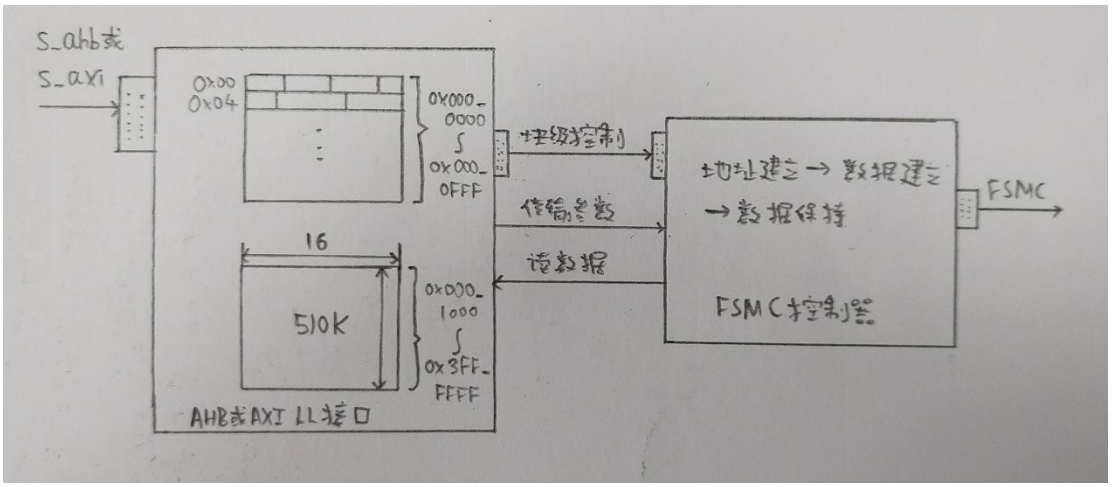

图 2-1 通用 FSMC 控制器组成框图

三 IP 功能

通用 FSMC 控制器带有 **AHB-Lite** 或 **AXI-Lite** 接口，内部划分为两个域：

（1）配置寄存器片（0x000_0000~0x000_FFFF）

偏移地址 0x00：7~0：地址建立周期数 - 1

15~8：数据建立周期数 - 1

偏移地址 0x02：7~0：数据保持周期数 - 1

（2）FSMC 存储映射区（0x000_1000~0x3FF_FFFF）

通用 FSMC 控制器将 AHB 或 AXI 传输转换为 FSMC 传输，写传输波形图如图 3-1 所示，读传输波形图如图 3-2 所示。FSMC 传输分为**地址建立**、**数据建立**、**数据保持**三个阶段，这三个阶段持续的周期数均可通过配置寄存器片进行设置，其含义在波形图中已标出。

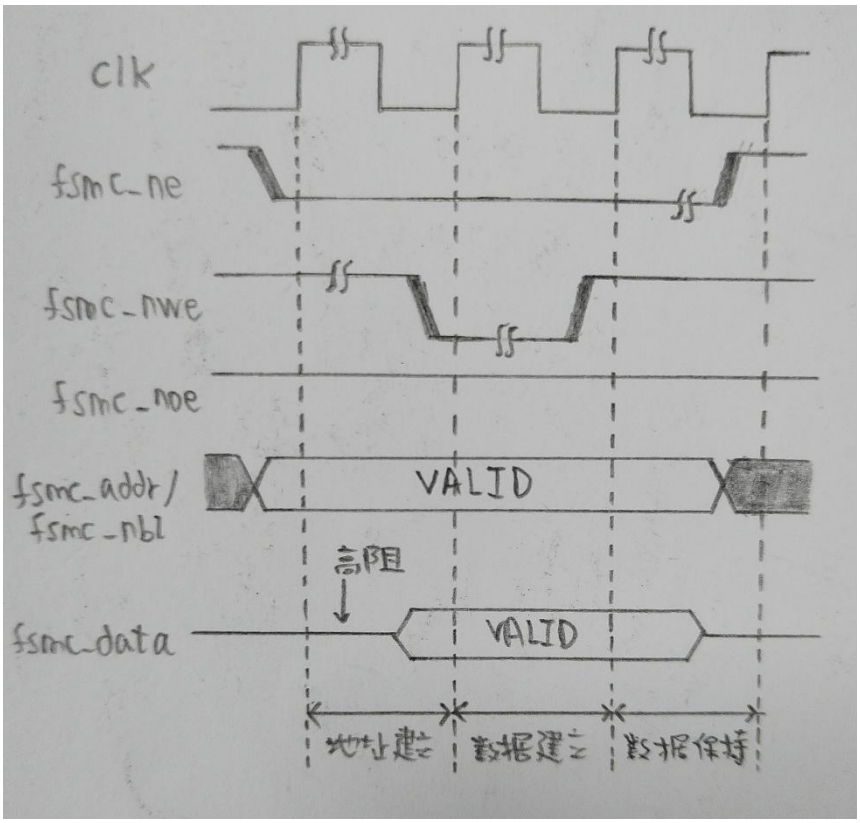

图 3-1 FSMC 写传输波形图

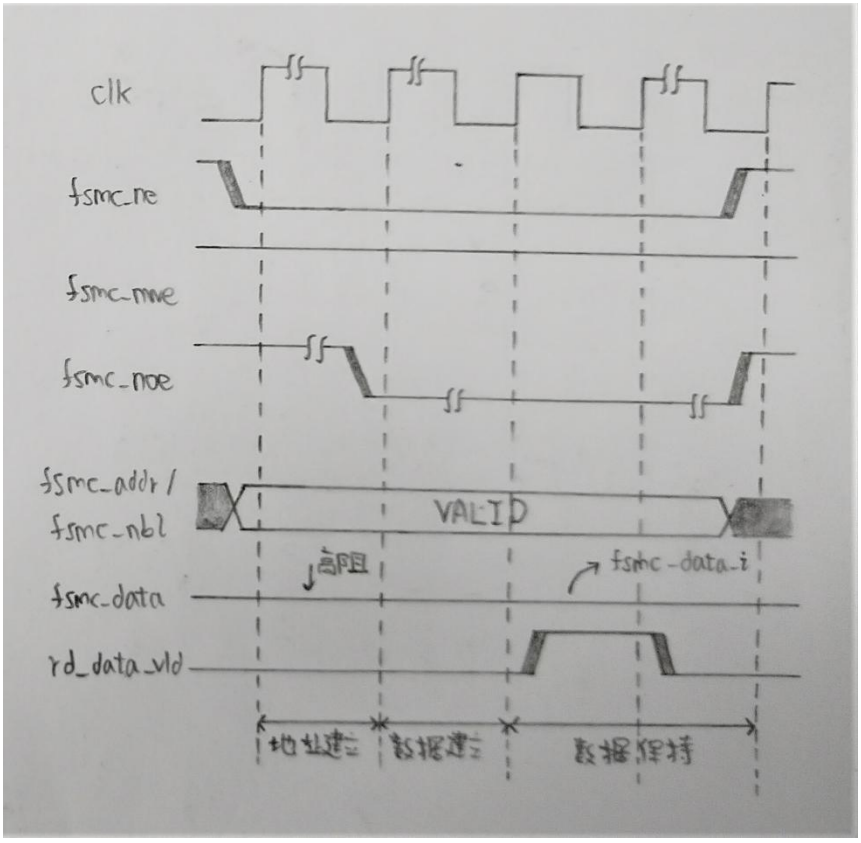

图 3-2 FSMC 读传输波形图

四 IO 描述

表 4-1 通用 FSMC 控制器 IO 表

端口名	方向	位宽	含义
时钟和复位			
clk	input	1	时钟
rst_n	input	1	复位，低有效
AHB-Lite 从接口（**AHB-Lite 或 AXI-Lite 从接口二选一**）			
… …			
AXI-Lite 从接口（**AHB-Lite 或 AXI-Lite 从接口二选一**）			
… …			
FSMC 主接口			
fsmc_nbl	output	2	FSMC 数据掩码
fsmc_addr	output	26	FSMC 行地址线
fsmc_nwe	output	1	FSMC 写使能
fsmc_noe	output	1	FSMC 输出使能（读使能）
fsmc_ne	output	1	FSMC 片选
fsmc_data_i	input	16	FSMC 数据三态输入
fsmc_data_o	output	16	FSMC 数据三态输出
fsmc_data_t	output	16	FSMC 数据三态方向，1 为输入，0 为输出

五 可配置参数描述

表 5-1 APB-I2C 可配置参数表

配置参数名	含义	可取值
simulation_delay	仿真延时，可用于仿真时模拟 D 到 Q 延迟	0.1f~100.0f

六 应用指南

6.1 RTL 设计指南

通用 FSMC 控制器可直接挂载到系统总线上，请根据需要选择使用 **AHB-Lite** 或 **AXI-Lite** 从接口。如果需要 **AHB-Lite** 从接口，请使用 **ahb_fsmc.v** 文件；如果需要 **AXI-Lite** 从接口，请使用 **axi_fsmc.v** 文件。

需要注意的是，虽然通用 FSMC 控制器带有 32 位的 **AHB-Lite** 或 **AXI-Lite** 从接口，但由于 FSMC 接口是 16 位的，因此 **AHB-Lite** 或 **AXI-Lite** 从接口的突发大小固定为 16 位，且地址对齐到半字。

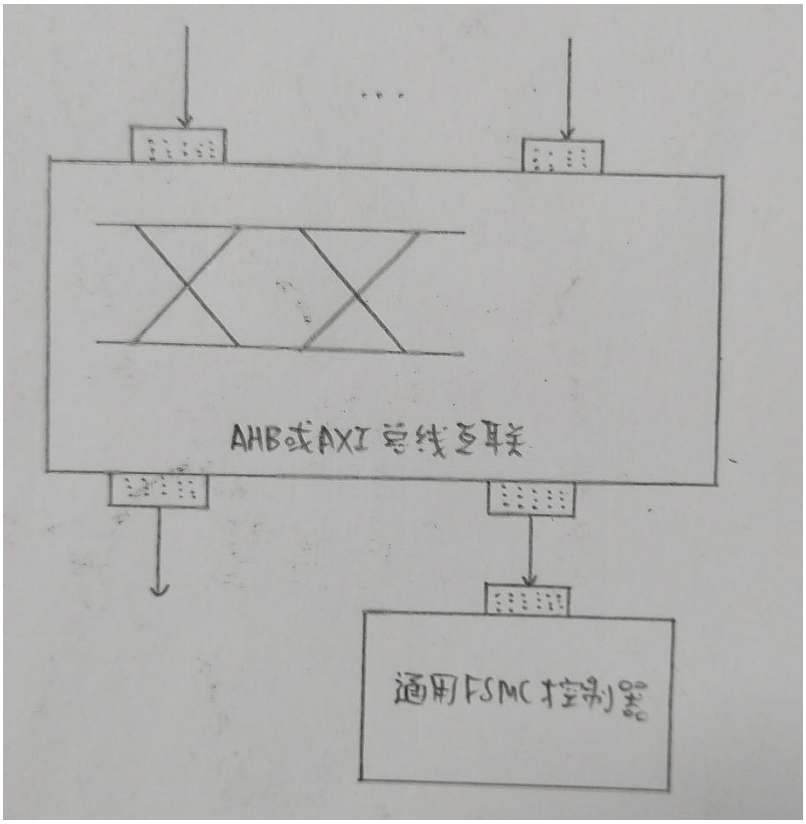

图 6-1-1 通用 FSMC 控制器应用图

通用 FSMC 控制器的 FSMC 接口是 16 位的，如果需要使用 **8 位 FSMC 接口**，那么可以使用地址线的高 25 位（fsmc_addr[25:1]）和数据线的低 8 位（fsmc_data_i[7:0]、fsmc_data_o[7:0]、fsmc_data_t[7:0]）。通用 FSMC 控制器的 FSMC 接口兼容 LCD 显示屏的 **8080 接口**，可使用地址线的第 4 位（fsmc_addr[3]）作为数据/命令指示信号。

6.2 软件编程指南

6.2.1 软件驱动 API

1、类型定义

• Fsmc 结构体（FSMC 外设结构体）
uint32_t base_addr_regs：配置寄存器片基地址
uint32_t base_addr_mem：FSMC 基地址

2、函数
• void init_fsmc(uint32_t base_addr, Fsmc* fsmc);
简介：初始化 FSMC 控制器
参数：**base_addr** FSMC 外设基地址
fsmc FSMC 外设（结构体指针）
返回值：无

• void fsmc_cfg_rt_pars(Fsmc* fsmc,
uint8_t addr_set, uint8_t data_set, uint8_t data_hold);
简介：配置 FSMC 控制器时序参数
参数：**fsmc** FSMC 外设（结构体指针）
addr_set 地址建立周期数 - 1
data_set 数据建立周期数 - 1
data_hold 数据保持周期数 - 1
返回值：无

• void fsmc_write_16(Fsmc* fsmc, uint32_t ofs, uint16_t data);
简介：FSMC 控制器以 16 位总线位宽写数据
参数：**fsmc** FSMC 外设（结构体指针）
ofs FSMC 存储偏移地址
data 待写的半字数据
返回值：无

• void fsmc_write_8(Fsmc* fsmc, uint32_t ofs, uint8_t data);
简介：FSMC 控制器以 8 位总线位宽写数据
参数：**fsmc** FSMC 外设（结构体指针）
ofs FSMC 存储偏移地址
data 待写的字节数据
返回值：无

• uint16_t fsmc_read_16(Fsmc* fsmc, uint32_t ofs);
简介：FSMC 控制器以 16 位总线位宽读数据
参数：**fsmc** FSMC 外设（结构体指针）
ofs FSMC 存储偏移地址
返回值：读取到的半字数据

• uint8_t fsmc_read_8(Fsmc* fsmc, uint32_t ofs);
简介：FSMC 控制器以 8 位总线位宽读数据
参数：**fsmc** FSMC 外设（结构体指针）
ofs FSMC 存储偏移地址

返回值：读取到的字节数据

6.2.2 软件编程示例

本示例基于通用 FSMC 控制器实现了对外部 SRAM 的读写，外部 SRAM 芯片的型号为 IS62WV51216。

```
/*********************************************************************************************************
FSMC 控制器示例代码
@brief  基于 FSMC 控制器实现外部 SRAM 读写
@date   2024/08/29
@author 陈家耀
@eidt   2024/08/29 1.00 创建了第一个正式版本
*********************************************************************************************************/

#include "../generic_fsmc_ctrler.h"

////////////////////////////////////////////////////////////////////////////////////////////////////////

#define BASEADDR_FSMC 0x60000000 // FSMC 外设基地址

////////////////////////////////////////////////////////////////////////////////////////////////////////

static Fsmc fsmc; // FSMC 控制器外设结构体

////////////////////////////////////////////////////////////////////////////////////////////////////////

void fsmc_example(void){
    init_fsmc(BASEADDR_FSMC, &fsmc); // 初始化 FSMC 外设

    /*
    配置 FSMC 控制器时序参数

    地址建立周期数 = 10
    数据建立周期数 = 10
    数据保持周期数 = 10
    */
    fsmc_cfg_rt_pars(&fsmc, 9, 9, 9);

    /** 写外部 SRAM **/
    for(uint32_t i = 0;i < 50;i++){
        fsmc_write_16(&fsmc, i << 1, i + 1);
```

```
36.         }
37.
38.         /** 读外部 SRAM **/
39.         uint8_t verify_ok = 1;
40.
41.         for(uint32_t i = 0;i < 50;i++){
42.             uint16_t d = fsmc_read_16(&fsmc, i << 1);
43.
44.             if(d != (i + 1)){
45.                 verify_ok = 0;
46.
47.                 break;
48.             }
49.         }
50.
51.         /** 验证读写数据一致性 **/
52.         if(verify_ok){
53.             // 读写数据一致
54.             // ...
55.         }else{
56.             // 读写数据不一致
57.             // ...
58.         }
59.
60.         while(1);
61.     }
```