



GPIO 模拟 JTAG 工程 移植说明

版本号：V1.0

日期：2023.01.16

声明

商标声明：



作为北京知存科技有限公司的商标, 本文件中提到的所有其他

商标和商号均为其持有人的财产。

版权声明：

Copyright © 2023 北京知存科技有限公司. All rights reserved.

内容声明：

本文件中的信息如有更改, 恕不另行通知。为了确保内容的准确性, 文章会做出相关的确认, 但本文件中的所有声明、信息和建议不构成任何明示或暗示的保证。

北京知存科技有限公司

地址: 北京市海淀区北四环西路 56 号辉煌时代大厦西座 1502

网址: <http://www.witmem.com>

目 录

| | | |
|---|--|----|
| 1 | 文档用途 | 4 |
| 2 | 模拟 JTAG 介绍 | 4 |
| | 2.1 JTAG (四线) 与 cJtag (两线) 对比 | 4 |
| | 2.2 JTAG (四线方式) 与 STM32F429 管脚连接 | 4 |
| | 2.3 cJTAG (两线方式)与 STM32F429 管脚连接 | 5 |
| 3 | 工程移植说明 | 5 |
| | 3.1 适配与移植 | 6 |
| | 3.2 JTAG 与 cJTAG 切换方法 | 8 |
| | 3.3 使用宏跳过底层 IO 映射 | 9 |
| | 3.4 工程演示说明与现象 | 9 |
| 4 | STM32F429 模拟 JTAG 的性能 | 10 |
| | 测试数据 | 11 |
| 5 | 版本修订记录 | 12 |

1 文档用途

此文档对使用 GPIO 方式模拟 JTAG 工程的移植进行说明。参考工程为 STM32F429 上使用 GPIO 方式模拟 JTAG 对 WTM2101 进行控制或内存读写。用户可将工程代码移植到其他平台从而对 WTM2101 进行控制与内存读写。对应工程为“STM32F429_jtag”

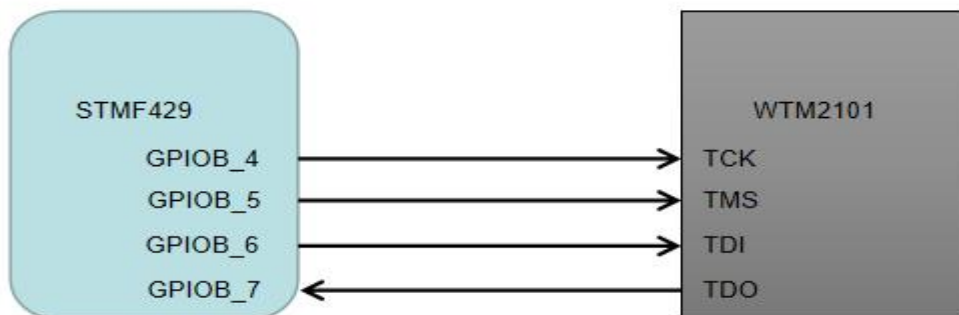
2 模拟 JTAG 介绍

WTM2101 JTAG 支持两种模式：JTAG（四线）与 cJTAG（两线）。

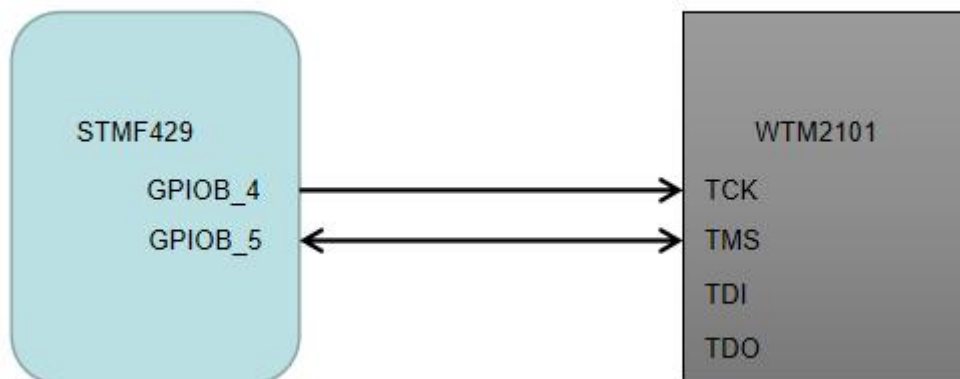
2.1 JTAG（四线）与 cJtag（两线）对比

| JTAG四线方式 | cJTAG两线方式 |
|---|--|
| <ul style="list-style-type: none"> 使用GPIO模拟JTAG需要四根I/O，分别为TCK、TMS、TDI、TDO。WTM2101稳定工作下TCK最大速度约为30M。TCK:时钟输出，TCK频率越高，JTAG数据传输率越高。TMS:输出信号，控制TAP状态。TDI: 输出信号，WTM2101端为输入。TDO:输入信号，WTM2101端为输出。 | <ul style="list-style-type: none"> 使用GPIO模拟cJTAG需要两根I/O，分别为TCK、TMS。其中TMS需要在输入输出之间进行切换。WTM2101稳定工作下TCK最大速度约为20M。TCK: 时钟输出，TCK频率越高cJTAG数据传输率越高。TMS:在cJTAG模式下TMS为双向信号。同TCK频率下，cJTAG数据传输率大约为JTAG四分之一。 |

2.2 JTAG（四线方式）与 STM32F429 管脚连接



2.3 cJTAG (两线方式)与 STM32F429 管脚连接



3 工程移植说明

工程名：“STM32F429_jtag”

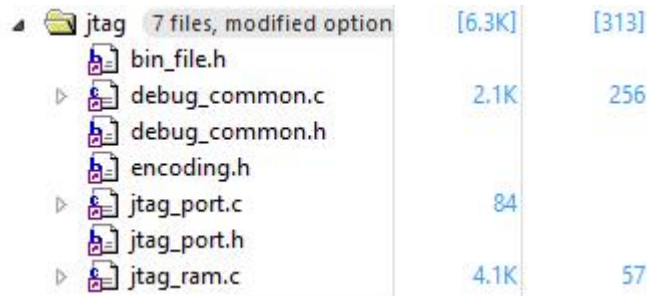
开发环境：SEGGER Embedded Studio 5.60 for ARM

工程目录如图 3-1 所示：

| 名称 | 修改日期 | 类型 |
|-----------|-----------------|-----|
| boot | 2023/1/11 14:24 | 文件夹 |
| cmsis | 2023/1/11 14:24 | 文件夹 |
| jtag | 2023/1/11 14:24 | 文件夹 |
| lib | 2023/1/11 14:24 | 文件夹 |
| Libraries | 2023/1/11 14:25 | 文件夹 |
| proj | 2023/1/16 9:26 | 文件夹 |
| rtt | 2023/1/11 14:25 | 文件夹 |
| spt | 2023/1/11 14:25 | 文件夹 |
| src | 2023/1/11 14:25 | 文件夹 |
| XML | 2023/1/11 14:25 | 文件夹 |

图 3-1 工程目录

图 3-2 所示为 jtag 目录下为需要移植的模拟 JTAG 相关的.c 与.h 文件。



| File Name | Size | Line Count |
|----------------|------|------------|
| bin_file.h | | |
| debug_common.c | 2.1K | 256 |
| debug_common.h | | |
| encoding.h | | |
| jtag_port.c | 84 | |
| jtag_port.h | | |
| jtag_ram.c | 4.1K | 57 |

图 3-2 工程文件

上述文件中 bin_file.h 中为演示所使用的 bin 文件无需改动。debug_common.c 与 debug_common.h 与 jtag_ram.c 与 encoding.h 为 JTAG 软件模拟的逻辑层无需用户改动。

3.1 适配与移植

用户需要改动的地方为 jtag_port.h 与 jtag_port.c 这两个文件与底层硬件（GPIO）相关。在 jtag_port.c 中需要用户完成模拟所使用 GPIO 的初始化工作 JTAG 为四个 IO；cJTAG 为两个 IO。

```
void Jtag_Io_Config()
{
    #if USE_HARDWARE
        GPIO_InitTypeDef GPIO_InitStructure;
        RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA, ENABLE);
        RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
        RCC_APB2PeriphClockCmd(RCC_APB2Periph_SYSCFG, ENABLE);
    #ifdef CJTAG_MODE
        GPIO_InitStructure.GPIO_Pin = TCK | TMS;
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
        GPIO_Init(GPIOB, &GPIO_InitStructure);
        GPIO_ResetBits(GPIOB, TCK);
        GPIO_ResetBits(GPIOB, TMS);
    #else
        GPIO_InitStructure.GPIO_Pin = TCK | TMS | TDI;
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
        GPIO_Init(GPIOB, &GPIO_InitStructure);

        GPIO_InitStructure.GPIO_Pin = TDO;
        GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
        GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
        GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
        GPIO_Init(GPIOB, &GPIO_InitStructure);
        GPIO_ResetBits(GPIOB, TCK);
        GPIO_ResetBits(GPIOB, TMS);
        GPIO_ResetBits(GPIOB, TDI);
    #endif
    #endif
}
```

在文件 jtag_port.h 中定义了所使用的具体 GPIO 的 PIN 脚与 GPIO 的翻转逻辑。用户修改此处指定相应 GPIO 的 PIN 定义。

```
#define TCK          GPIO_Pin_4 // JTAG Test Clock
#define TMS          GPIO_Pin_5 // JTAG Mode Select
#define TDI          GPIO_Pin_6 // JTAG Data Input
#define TDO          GPIO_Pin_7 // JTAG Data Output
```

以下函数含义如下（需用户根据自己平台进行修改）：

```
#define DEBUG_PRINT printf
#define DELAY() {__NOP();__NOP();__NOP();}
#define Delay_ms() \
{ \
    for(int i = 0; i < 180000; i++) \
        __NOP(); \
}

#define JTAG_IO_SET_TMS_HIGH() {GPIOB->BSRR = TMS;}
#define JTAG_IO_SET_TMS_LOW() {GPIOB->BSRRH = TMS;}
#define JTAG_IO_SET_TMS(VALUE) {if(VALUE) {GPIOB->BSRR = TMS;} else {GPIOB->BSRRH = TMS;}}
#define JTAG_IO_SET_TCK_HIGH() {GPIOB->BSRR = TCK;}
#define JTAG_IO_SET_TCK_LOW() {GPIOB->BSRRH = TCK;}
#define JTAG_STROBETCK() {GPIOB->BSRR = TCK;DELAY();GPIOB->BSRRH = TCK;DELAY();}
#define CJTAG_STROBETCK() {GPIOB->BSRRH = TCK;DELAY();GPIOB->BSRR = TCK;DELAY();}
#define JTAG_SET_TMS_INPUT() {GPIOB->MODER = 0x1180;}
#define JTAG_SET_TMS_OUTPUT() {GPIOB->MODER = 0x1580;}
#define JTAG_IO_SET_TCK(VALUE) {if(VALUE) {GPIOB->BSRR = TCK;} else {GPIOB->BSRRH = TCK;}}
#define JTAG_IO_SET_TDI(VALUE) {if(VALUE) {GPIOB->BSRR = TDI;} else {GPIOB->BSRRH = TDI;}}
#define JTAG_IO_SET_PIN(VALUE) {if(VALUE) {GPIOB->BSRR = PIN;} else {GPIOB->BSRRH = PIN;}}
#define JTAG_READ_TDO() (GPIOB->IDR & TDO)
#define JTAG_READ_TMS() (GPIOB->IDR & TMS)
```

定义打印，如果没有打印可写为空
延时三个时钟周期

延时1MS
拉高TMS管脚
拉低TMS管脚
拉高TMS管脚根据输入变量拉低或者拉高TMS管脚
拉高TCK管脚
拉低TCK管脚
四线模式翻转TCK管脚，TCK先拉高后拉低
两线模式翻转TCK管脚，TCK先拉低后拉高
将TMS配置为输入
将TMS管脚配置为输出
根据输入变量拉低或者拉高TCK管脚
根据输入变量拉低或者拉高TDI管脚
根据输入变量拉低或者拉高对应管脚
读取输入管脚TDO输入电压
读取管脚TMS为输入状态时的输入电压

注意：关于 DELAY() 的使用需要用户根据自己平台的 IO 翻转速度自行调整。四线模式要求 TCK 翻转速度不超过 30M，两线模式要求 TCK 翻转速度不超过 15M。

3.2 JTAG 与 cJTAG 切换方法

工程中使用预定义宏进行切换的示例如图 3-3 所示。

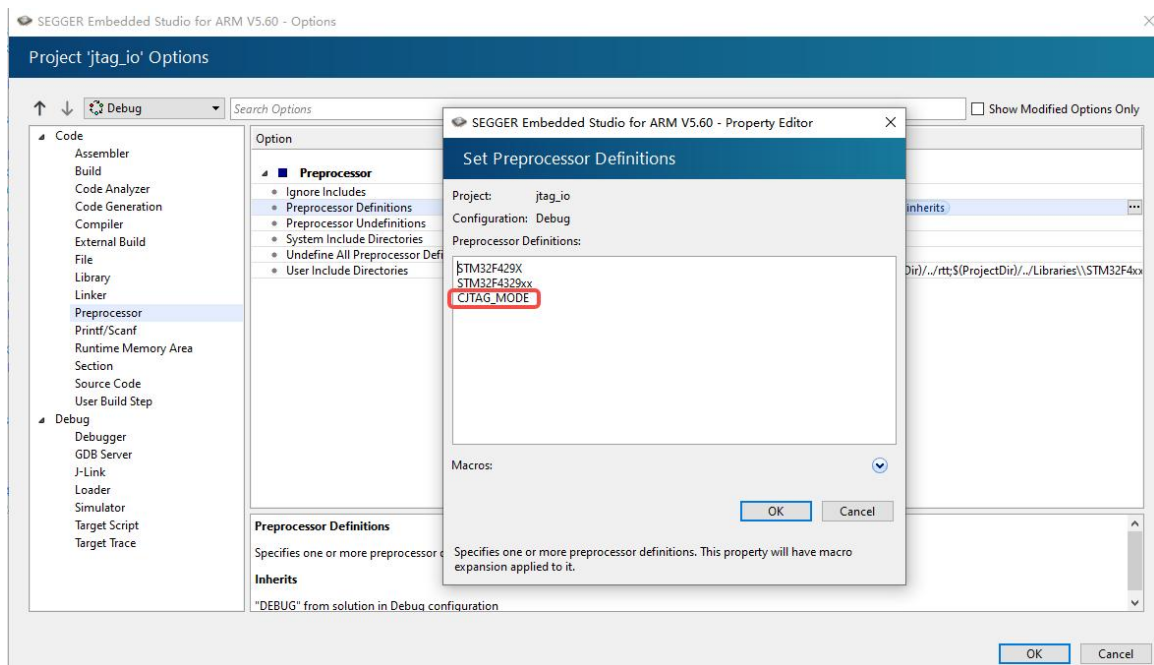


图 3-3 JTAG 与 cJTAG 切换预定义宏

如不添加宏定义 CJTAG_MODE，则工程使用 JTAG 方式；添加宏定义 CJTAG_MODE，则工程使用两线方式。

3.3 使用宏跳过底层 IO 映射

工程中使用预定义宏的方式控制是否编译底层 IO 映射代码的实例如图 3-4 所示。

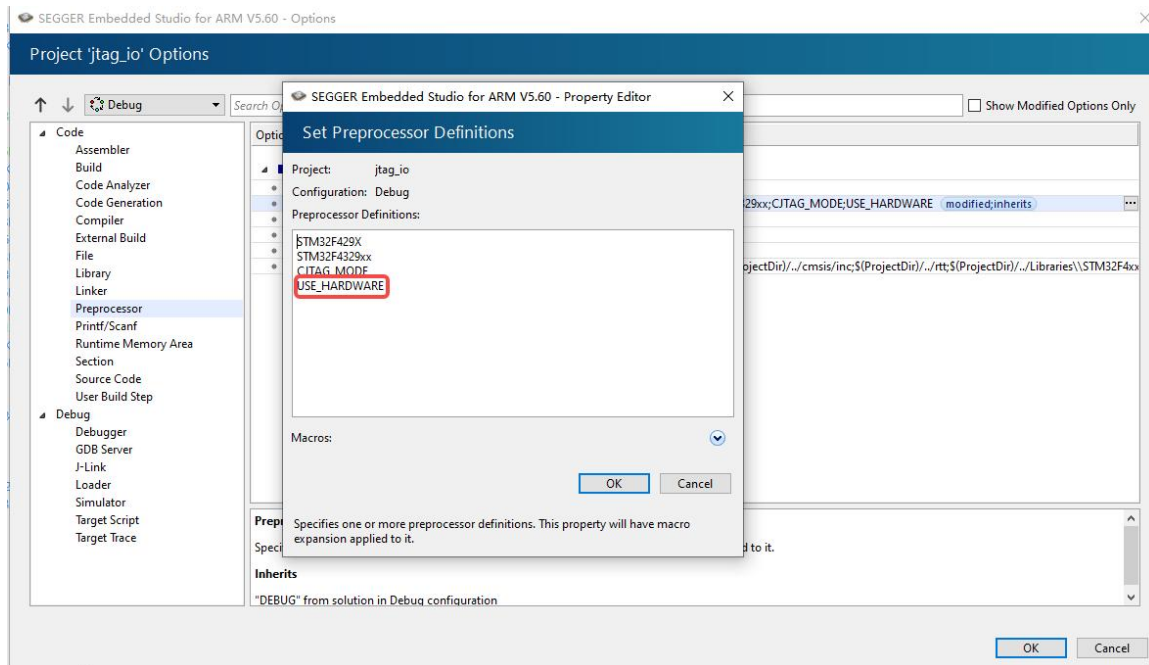


图 3-4 是否编译底层 IO 代码预定义宏

如不添加宏定义 `USE_HARDWARE`，则工程忽略底层 IO 映射部分、只编译软件逻辑代码；添加宏定义 `USE_HARDWARE`，则工程编译底层 IO 映射部分代码。

3.4 工程演示说明与现象

在 `jtag_ram.c` 中函数 `void Jtag_Download_Bin` 的作用是将 `bin_file.h` 中的文件数据下载到 WTM2101 的 `0x00000000` 地址并运行。如运行结果正确可以看到终端窗口输出如图 3-5 所示内容。

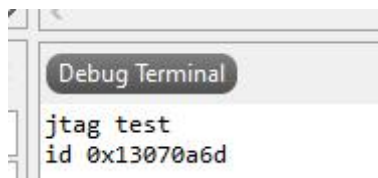


图 3-5 终端输出结果

将 WTM2101 GPIO0 与 LED0 使用跳冒连接（图 3-6 中①所示），可以观察到 LED0 闪烁（图 3-6 中②所示）。

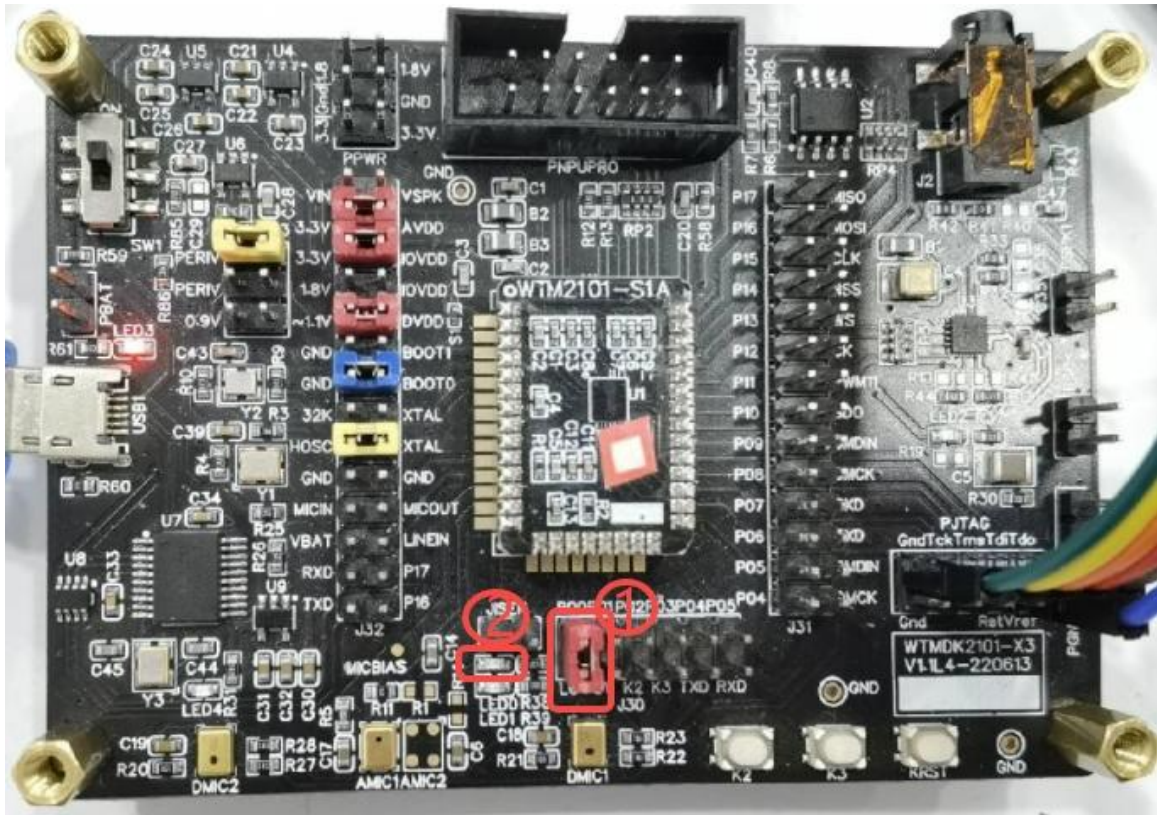


图 3-6 WTM2101-X3 开发板 LED 连接图

4 STM32F429 模拟 JTAG 的性能

STM32F429 平台为软件 JTAG 下载器，以 WTMDK2101-X3 开发板为目标设备，分别对两线/四线模式下的 JTAG 固件进行性能测试。STM32F429 平台开启 FLASH Prefetch、DCache 及 ICache 功能，IO 电压 3.3V（WTM2101 支持 3.3V 或 1.8V IO），测试所用待传输二进制文件大小为 9.5KB。测试结果如表 1-1。

测试数据

表 1-1 STM32F429

| 测试项 | JTAG（四线） | cJTAG（两线） |
|-----------|-----------|-----------|
| TCK 最高速率 | 90MHz | 33MHz |
| 实际下载速率 | 38.03KB/s | 9.4KB/s |
| JTAG 固件大小 | 约 3KB | 约 4.2KB |
| 编译器优化等级 | O3 | O0 |

注：1.TCK 最高速率项为当前环境监测最高速率；
2.平台最高系统时钟为 180MHz；
3.编译器版本为 GCC (SEGGER Embedded Studio 5.60)；
4.使用电平转换芯片可能导致性能损失，使 TCK 无法达到最高速率，实际使用时需参考电平转换芯片手册。

5 版本修订记录

表 2-1 修订记录

| 版本 | 日期 | 说明 |
|-----|------------------|------|
| 1.0 | 2023 年 01 月 16 日 | 初始版本 |