

Fabric Configuration 用户手册

(Version: V1.0)

深圳市紫光同创电子有限公司

版权所有 侵权必究

文档版本修订记录

版本号	发布日期	修订记录
V1.0	2022.07.12	初始版本

目录

1 软件简介	9
2 下载电缆介绍	10
2.1 USB 下载电缆	10
2.2 并口下载电缆	10
2.3 下载电缆的指示灯状态	11
3 软件使用入门	12
3.1 连接 JTAG SERVER	12
3.2 设置 CABLE 参数	13
3.3 多 CABLE 同时使用	14
3.4 扫描 JTAG 链	14
3.5 配置逻辑位流文件	15
3.6 编程逻辑位流文件	16
3.7 校验逻辑位流文件	17
4 FABRIC CONFIGURATION 软件说明	19
4.1 用户界面介绍	19
4.1.1 用户主界面	19
4.2 菜单栏与工具栏	20
4.2.1 File 菜单	20
4.2.2 Edit 菜单	20
4.2.3 View 菜单	20
4.2.4 Operations 菜单	21
4.2.5 Debug 菜单	21
4.2.6 Help 菜单	21
4.2.7 工具栏基本操作	22
4.2.8 Device Properties 窗口	22
4.3 功能说明	23
4.3.1 Jtag 边界扫描添加器件	23
4.3.2 解析目标位流添加器件	26
4.3.3 配置新的位流文件	27
4.3.4 Jtag 编程 FPGA 器件	29
4.3.5 Jtag 校验 FPGA 器件	29
4.3.6 Jtag 回读 FPGA 器件	30
4.3.7 Jtag 操作 Compact 系列的特征控制位	31
4.3.8 Jtag 操作 Compact 系列的嵌入式 Flash	32
4.3.9 Jtag 编程 eFuse 寄存器	34
4.3.10 Jtag 软复位 FPGA	38
4.3.11 Jtag 扫描 Flash 器件	39
4.3.12 SPI 扫描 Flash 器件	40
4.3.13 SPI 擦除 Flash 器件	41
4.3.14 SPI 编程 Flash 器件	42
4.3.15 SPI 校验 Flash 器件	42
4.3.16 SPI 回读 Flash 器件	43

4.3.17 SPI 读写 Flash 寄存器	44
4.3.18 设置操作器件属性.....	44
4.3.19 FPGA 位流文件转换多种格式数据流.....	47
4.3.20 Jtag 通用配置文件(SVF & PEF)	62
4.3.21 Tcl 脚本的生成与执行	68
4.3.22 设置用户自定义信息.....	71
4.3.23 链完整性测试.....	72
4.3.24 设置用户自定义 Flash	72
4.3.25 添加用户自定义器件.....	74
4.3.26 Jtag 间接烧写外部 flash.....	75
4.4 TCL 命令使用入门.....	76
4.4.1 Tcl 命令简介	76
4.4.2 查看支持的所有 Tcl 命令	76
4.4.3 查询具体 Tcl 命令的参数	76
4.4.4 获取界面操作的 Tcl 命令	77
4.4.5 执行 Tcl 命令	78
4.4.6 Tcl 的使用示例	80
4.5 TCL 命令具体参数查询.....	80
4.5.1 通用命令的说明和用法	80
4.5.2 针对 FPGA 器件的命令	82
4.5.3 针对 Flash 器件命令	87
4.5.4 针对 Jtag Flash 器件命令	89
4.5.5 针对板上自动化.....	90
4.5.6 转换文件格式.....	92
4.6 终端模式介绍.....	96
5 附录.....	98
5.1 指令寄存器各位表示的功能	98
5.1.1 Titan 系列指令寄存器说明.....	98
5.1.2 Logos 系列指令寄存器说明	98
5.1.3 Logos2 系列指令寄存器说明	99
5.1.4 Compact 系列指令寄存器说明	99
5.1.5 Titan2 系列指令寄存器说明.....	99
5.2 状态寄存器各位表示的功能	100
5.2.1 Titan 系列状态寄存器说明.....	100
5.2.2 Logos 系列状态寄存器说明	100
5.2.3 Logos2 系列状态寄存器说明	101
5.2.4 Compact 系列状态寄存器说明	102
5.2.5 Titan2 系列状态寄存器说明.....	103
5.3 COMPACT 系列的特征控制位说明	104
5.3.1 PGC 1K、2K 系列支持的特征控制位	104
5.3.2 PGC 4K、7K、10K 系列支持的特征控制位	105
5.4 目前支持的 FLASH 器件及其匹配的读写模式	106
6 免责声明.....	109

表目录

表 1 Titan2 系列的编程属性.....	45
表 2 Logos 系列的编程属性	45
表 3 Compact 系列的编程属性.....	46
表 4 器件系列与文件类型对应表	52
表 5 CRAM SVF 参数	64
表 6 嵌入式 Flash SVF 配置参数含义	66
表 7 特殊指令含义	66
表 8 用户操作与 Tcl 命令的对应操作	69
表 9 Titan 系列指令寄存器说明	98
表 10 Logos 系列指令寄存器说明	98
表 11 Logos2 系列指令寄存器说明	99
表 12 Compact 系列指令寄存器说明.....	99
表 13 Titan2 系列指令寄存器说明	99
表 14 Titan 系列状态寄存器说明	100
表 15 Logos 系列状态寄存器说明	101
表 16 Logos2 系列状态寄存器说明	102
表 17 Compact 系列状态寄存器说明.....	103
表 18 Titan2 系列状态寄存器说明	103
表 19 PGC 1K、2K 系列支持的特征控制位	105
表 20 PGC 4K、7K、10K 系列支持的特征控制位	106
表 21 目前支持的 Flash 器件及其匹配的读写模式.....	108

图目录

图 2-1 USB Cable I 下载电缆连接示意图	10
图 2-2 USB Cable II 下载电缆连接示意图	10
图 2-3 并口下载电缆连接示意图	11
图 3-1 连接 Jtag Server.....	12
图 3-2 选择 Cable 参数	13
图 3-3 设置 Cable 参数	13
图 3-4 工作区鼠标右键单击菜单.....	14
图 3-5 初始化链成功.....	15
图 3-6 为 JTAG 链上的器件添加配置文件	15
图 3-7 在 FPGA 上单击右键的菜单.....	16
图 3-8 下载配置文件成功.....	17
图 3-9 校验下载配置的文件.....	18
图 4-1 Fabric Configuration 主界面	19
图 4-2 File 菜单	20
图 4-3 Edit 菜单	20
图 4-4 View 菜单	21
图 4-5 Operations 菜单	21
图 4-6 Debug 菜单	21
图 4-7 Help 菜单	22
图 4-8 工具栏.....	22
图 4-9 Device Properties 窗口	23
图 4-10 工作区鼠标右键单击菜单	23
图 4-11 初始化链并添加.sbit 文件	24
图 4-12 配置正确的位流文件	25
图 4-13 配置错误的位流文件	25
图 4-14 初始化链失败	26
图 4-15 扫到未知器件设置指令长度	26
图 4-16 添加不支持器件的位流文件	27
图 4-17 在器件上单击右键的菜单	28
图 4-18 添加了不匹配的配置文件	28
图 4-19 下载配置文件成功	29
图 4-20 下载失败	29
图 4-21 FPGA 校验失败	30
图 4-22 FPGA 校验成功	30
图 4-23 FPGA 回读	31
图 4-24 操作嵌入式 Flash	32
图 4-25 选择操作 eFuse 类型	35
图 4-26 配置秘钥文件	36
图 4-27 配置秘钥信息	36
图 4-28 配置永久秘钥时的警告	37
图 4-29 选择秘钥文件写入 BBRAM	37
图 4-30 32bits User eFuse 配置界面	38
图 4-31 PGL50H 操作菜单	39
图 4-32 JTAG 操作 Flash 器件	40

图 4-33 添加.sfc 文件	41
图 4-34 扫描 SPI Device 成功	41
图 4-35 SPI Flash 支持的操作	43
图 4-36 回读 SPI Flash 数据存储文件	43
图 4-37 SPI Flash 状态寄存器	44
图 4-38 Convert File Dialog 对话框	48
图 4-39 Generate Flash Programming File	49
图 4-40 Generate Daisy Chain File	50
图 4-41 Generate Multi Revision Programming File	51
图 4-42 地址输入为空不能生成	52
图 4-43 地址输入不规范不能生成	53
图 4-44 Logos、Logos2 和 Titan2 系列组合数据流的位流格式 1	54
图 4-45 Logos、Logos2 和 Titan2 系列组合数据流的位流格式 2	55
图 4-46 Compact 系列组合数据流的位流格式	56
图 4-47 PGL 系列方案级位流产生窗口	57
图 4-48 双启动位流设置界面	58
图 4-49 配置文件导入界面	58
图 4-50 配置文件生成界面	59
图 4-51 双启动位流其余设置	59
图 4-52 tcl 提示界面	60
图 4-53 多启动位流输入设置界面	61
图 4-54 多启动位流输出设置界面	61
图 4-55 Svf 文件合并功能	62
图 4-56 需要生成 pef	63
图 4-57 生成 CRAM 类型 svf	64
图 4-58 选择生成的 SVF 文件路径与参数	65
图 4-59 带特殊指令的 SVF	67
图 4-60 执行 SVF 文件	68
图 4-61 Operations 菜单	69
图 4-62 执行 tcl 脚本窗口	70
图 4-63 Preference 对话框	71
图 4-64 标题栏中用户信息	72
图 4-65 执行 Tcl 后的日志文件	72
图 4-66 测试链完整性成功	72
图 4-67 测试链完整性失败	72
图 4-68 Flash Operator 主界面	73
图 4-69 新增 Flash 厂商系列选择界面	73
图 4-70 器件信息设置界面	74
图 4-71 添加用户自定义器件	74
图 4-72 Jtag 间接烧写外部 flash 选项位置	75
图 4-73 Jtag 间接烧写外部 flash 界面	75
图 4-72 cfg_help 显示所有支持的 Tcl 命令	76
图 4-73 控制台查询 cfg_program_user_efuse 的具体用法	77
图 4-74 烧写临时秘钥文件的操作界面	78
图 4-75 tcl 文件拖入控制台窗口执行	79
图 4-76 Windows 下用 cmd 调用 cdt_cfg.exe	79

图 4-77 cfg_shel 显示 97

1 软件简介

Fabric Configuration 软件主要功能是用于 FPGA 芯片的配置，负责把用户逻辑设计生成的位流文件下载到 FPGA 芯片中。

支持以下 SPI 串行 Flash 芯片类型

Micron 系列: MT25Q512, M25P128, M25P64, M25P32, M25P16, N25Q32, N25Q64, N25Q128, N25Q256, N25Q512

Giga 系列: GD25Q80C, GD25Q32C, GD25Q64C, GD25Q128C, GD25Q256D, GD25Q512C, MD25Q80C

WinBond 系列: W25Q40CL, W25Q80, W25Q16, W25Q32, W25Q64Q, W25Q128, W25Q256, W25Q512, W25M512

XMC 系列: XM25QH16B, XM25QH32B, XM25QH64A, XM25QH128A, XM25QH256B

SSMEC 系列: SM25QH256M

CYPRESS 系列: S25FL64, S25FL128, S25FL256L, S25FL256S, S25FL512

ZBIT 系列: ZB25VQ16, ZB25VQ32

MXIC 系列: MX25L16, MX25L32, MX25L64, MX25L128, MX25L256, MX25L512

ISSI 系列: IS25LP16D, IS25LP32D, IS25LP64D, IS25LP128D, IS25LP256D, IS25LP512D, IS25WP16D, IS25WP32D, IS25WP64D, IS25WP128D, IS25WP256D, IS25WP512D

ESMT 系列: EN25Q80C

PUYA 系列: P25Q80H

ADESTO 系列: AT45DB081E

FENTECH 系列: FH25VQ80D

Spansion 系列: S25FL064L, S25FL128L, S25FL256S, S25FL512S

- a. 支持 Pango USB Cable I, Pango USB Cable II, Pango Parallel Cable I 下载电缆
- b. 支持 FPGA 的 ID、user code 及状态寄存器读取功能，逻辑位流下载、回读及校验
- c. 支持 SPI Flash 的下载、删除、回读及校验功能

2 下载电缆介绍

2.1 USB 下载电缆

使用 USB 下载电缆需要安装对应的驱动，安装步骤参见文档 Pang Design Suite Windows Install Guide 中软件安装部分的介绍。驱动安装成功后插入 USB 下载电缆，如果在设备管理器中多了新的设备“Programming cables”，则说明下载电缆可以正常使用了。

目前支持 Pango USB Cable I 和 Pango USB Cable II，USB 下载电缆负责把 PC 机的 USB 信号转换为器件所需要的 JTAG 信号或 SPI 信号。下载电缆线的物理连接以下载线包装盒内提供的说明文档为准，其连接方式如图 2-2 所示：

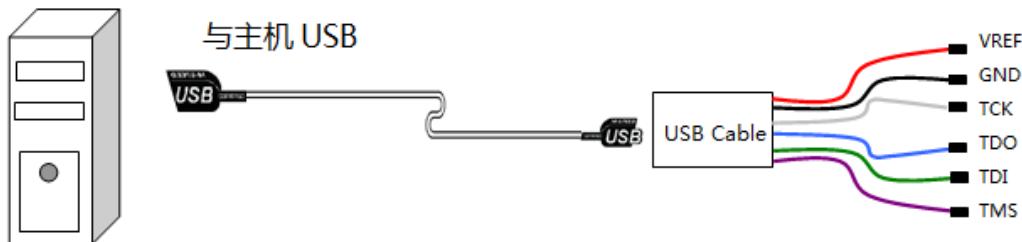


图 2-1 USB Cable I 下载电缆连接示意图

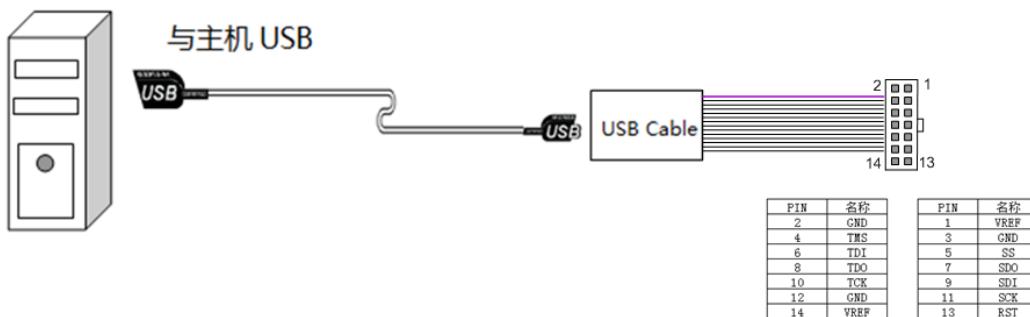


图 2-2 USB Cable II 下载电缆连接示意图

注：图中 14PIN 的连接头中，PIN1 的 VREF 给 JTAG 接口供电，PIN14 的 VREF 给 SPI 接口供电，两者可不同。

2.2 并口下载电缆

使用并口下载电缆，首先要确定 PC 机的主板上有集成的并口卡，并且在 BIOS 中设置启用，如果要使用 ECP 模式，还需要在 BIOS 中设置启用并口 ECP 模式。其次要安装并口驱动，安装步骤参见文档 Pang Design Suite Windows Install Guide 中软件安装部分的介绍。

目前支持 Pango Parallel Cable I， 并口下载电缆负责把 PC 机的并行信号转换为器件所

需要的 JTAG 信号或 SPI 信号。下载线的物理连接以下载线包装盒内提供的说明文档为准，其连接方式如下图所示：

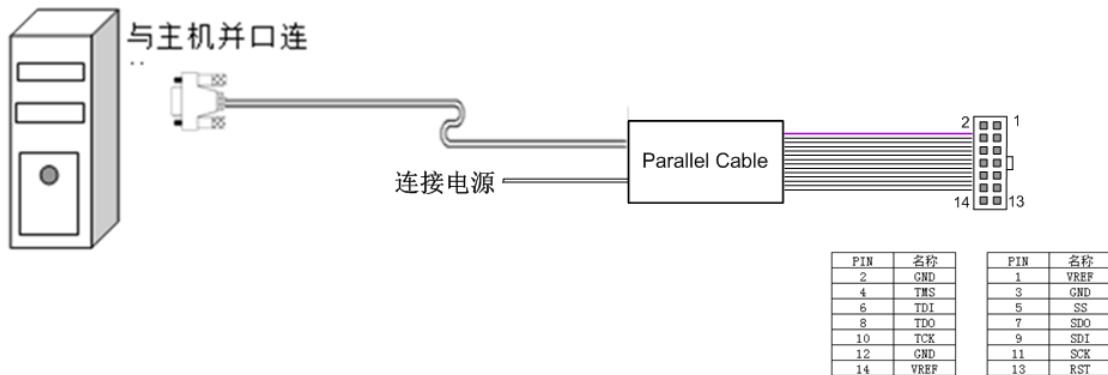


图 2-3 并口下载电缆连接示意图

注：图中 14PIN 的连接头中，PIN1 的 VREF 给 JTAG 接口供电，PIN14 的 VREF 给 SPI 接口供电，两者可不同。

2.3 下载电缆的指示灯状态

Pango USB Cable II 和 Pango Parallel Cable I 都有一个指示灯可以显示红色和绿色，Pango USB Cable I 无指示灯功能，定义每次对 FPGA 和 Flash 的读写都是一次操作，指示灯的显示规律如下：

- 初始状态红灯常亮，绿灯灭
- 操作进行中绿灯闪，红灯灭
- 操作结果出错红灯常亮，绿灯灭
- 操作结果正常绿灯常亮，红灯灭

3 软件使用入门

首先，通过一个简单的例子来演示 Fabric Configuration 软件如何下载逻辑位流。本例子所采用 FPGA 芯片为 Pango Logos 系列的 PGL50H，所要配置的逻辑位流文件名为 pgl50h.sbit。

3.1 连接 Jtag Server

Fabric Configuration 软件是与相应的 Jtag Server 软件配对使用的，且两者之间的版本需一致，否则将不能正常进行工作。在启动 Fabric Configuration 软件后，在工具栏点击按钮 ，弹出连接到 Jtag Server 端 Cable 的向导，按照向导的指示便可连接到服务端。本软件采用 TCP/IP 协议与服务端进行通信，所以需要指定 IP 地址和端口，IP 地址既可以指定为本机 IP 地址也可以指定为远端的 IP 地址。

当指定为本地 IP 地址时，由于默认的地址为本机地址，因此直接点击连接即可。如果指定为远程 IP 地址时，需要先在远程启动 Jtag Server，在安装目录的 bin 目录下找到名为 cdt_js 的软件双击启动即可；用户也可以使用命令行方式启动，具体启动方法为：切换到可执行程序所在的目录，然后启动操作系统的命令行终端，输入命令 cdt_js -port 65420 即可，更多使用参数可以输入命令 cdt_js -help 查看。

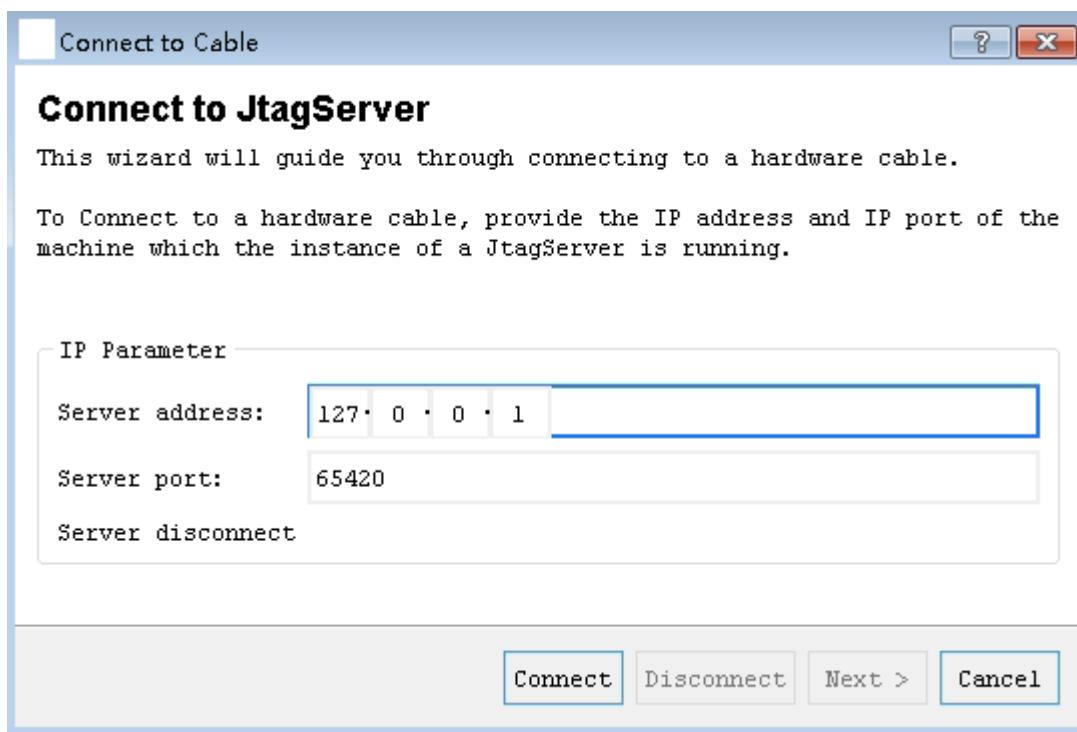


图 3-1 连接 Jtag Server

3.2 设置 Cable 参数

在连接 Jtag server 成功后，连接界面会列出当前所有可用的 Cable，可以通过单击选中一个 Cable 和设置 TCK 频率，默认选择第一个 Cable。

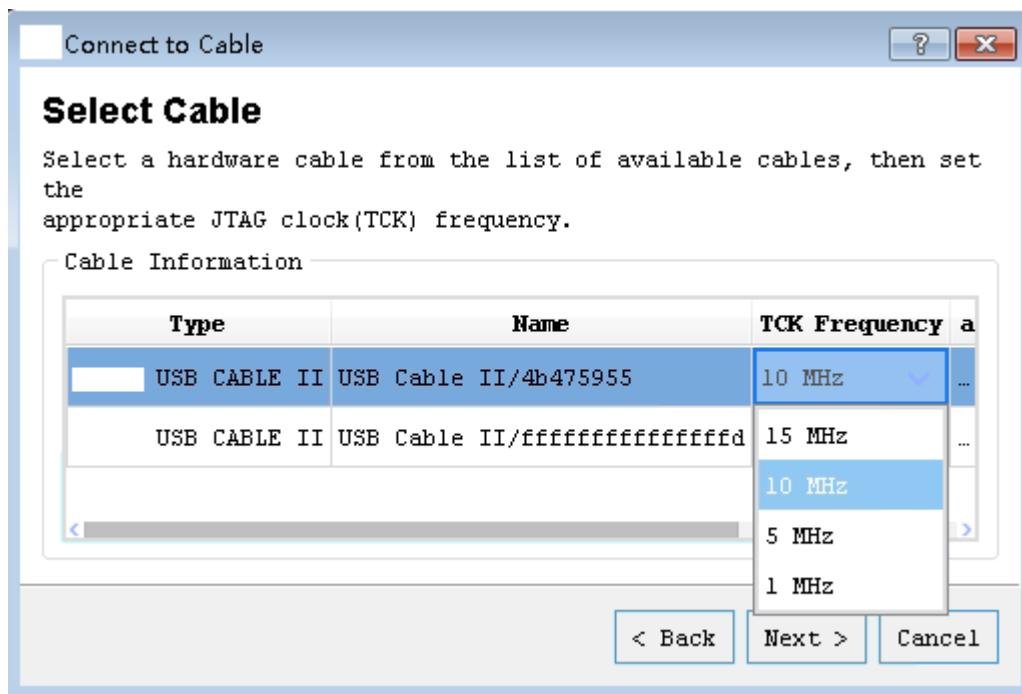


图 3-2 选择 Cable 参数

在选择所需要的 Cable 后，点击 Next 按钮跳到总结页面，该页面会列出服务端连接信息和 Cable 参数，此时 cable 参数设置还未生效，点击 Back 可以回到上一步重新选择，点击 Cancel 可以取消本次设置，点击 Next 会将配置信息发送到 Jtag Server 端并进行设置。

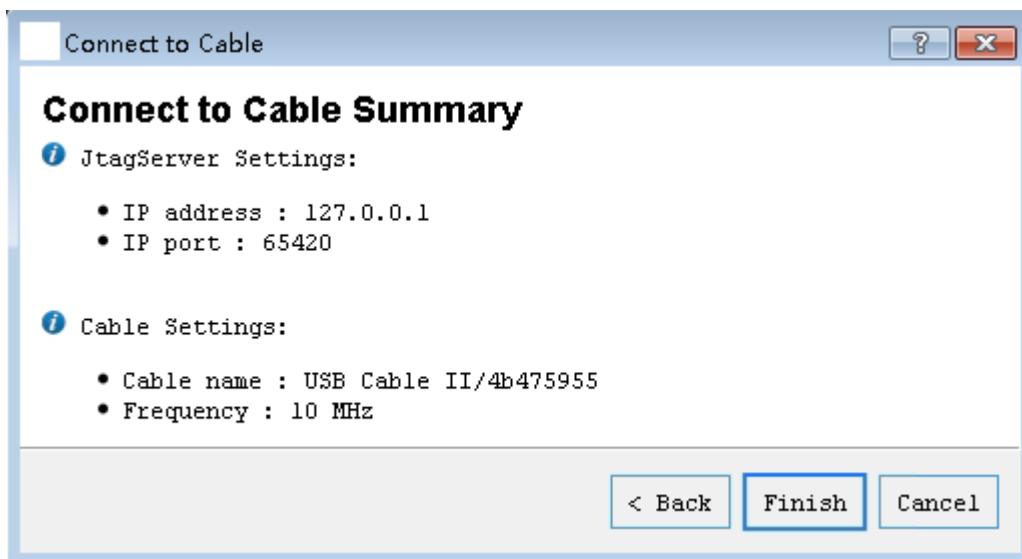


图 3-3 设置 Cable 参数

3.3 多 Cable 同时使用

如果需要在一台机器上同时使用多个 Cable，可以通过以下步骤来实现（这里以同时使用两个 Cable 为例）：

- 将两个 Cable 连接到 USB 端口，并连接好对应的电路板，打开电源
- 切换到 PDS 软件安装目录的 bin 文件夹（后续操作均在此目录下进行）
- 启动命令行终端，输入命令 cdt_js –port 65420
- 启动命令行终端，输入命令 cdt_js –port 65421
- 启动 Fabric Configuration 软件，点击按钮 ，在连接到 Jtag Server 的向导中 Server Port 设置为 65420，选择第一个 Cable，其余参数默认，完成第一个 Cable 的设置
- 启动 Fabric Configuration 软件，点击按钮 ，在连接到 Jtag Server 的向导中 Server Port 设置为 65421，选择第二个 Cable，其余参数默认，完成第二个 Cable 的设置
- 分别测试两个 Cable 的扫链，配置功能即可

3.4 扫描 JTAG 链

在主界面的左侧 Configuration Mode 窗口，点击“Boundary Scan”，然后在工作区进行鼠标右键单击，将会弹出如图 3-4 所示菜单，左键单击“Scan Device”可以执行初始化链的操作。

注：打开 Fabric Configuration 软件后可以直接进行扫描 Jtag 链操作，软件本身会自动触发连接 Jtag Server 的操作，Jtag Server 默认的 IP 地址是 127.0.0.1，端口号为 65420，连接成功后会进行版本的检查，版本检查成功后将进行扫描 Jtag 链操作。



图 3-4 工作区鼠标右键单击菜单

如果初始化链成功，会将链上扫描到的所有器件显示于工作区内，并在器件属性窗口显示当前器件的器件信息，并弹出对话框使用户能够为器件添加配置文件，如图 3-5 所示：

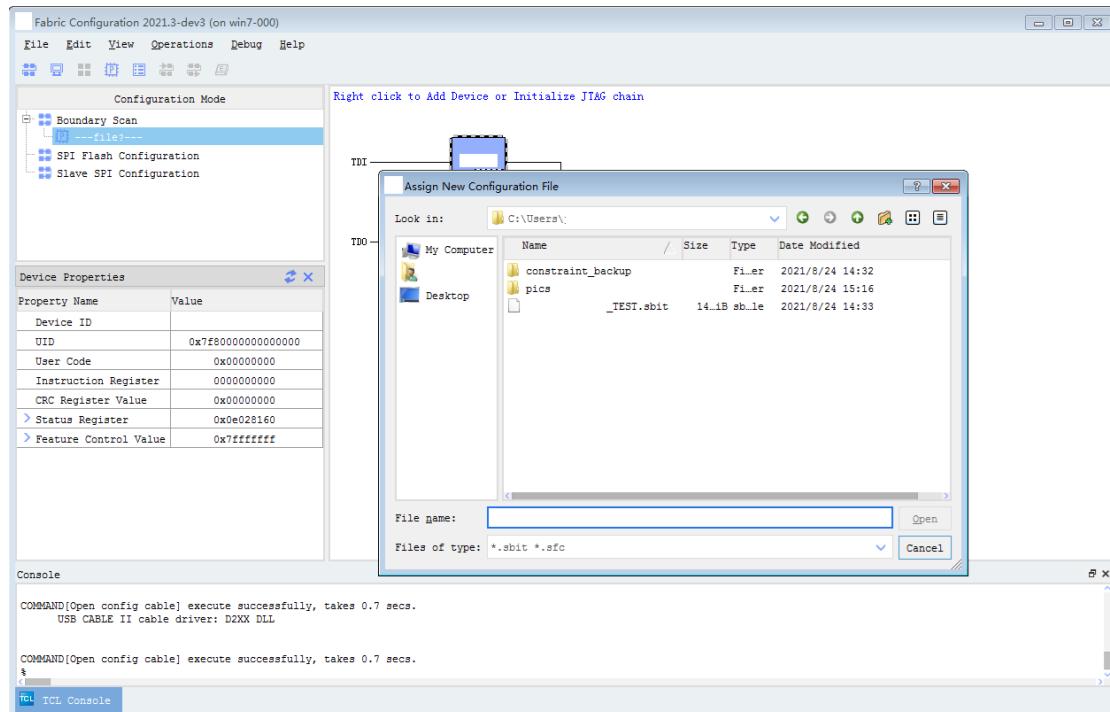


图 3-5 初始化链成功

3.5 配置逻辑位流文件

在文件配置对话框选择目标位流文件，添加该配置文件，提示所载入文件的绝对路径并在信息栏中显示，如图 3-6 所示：

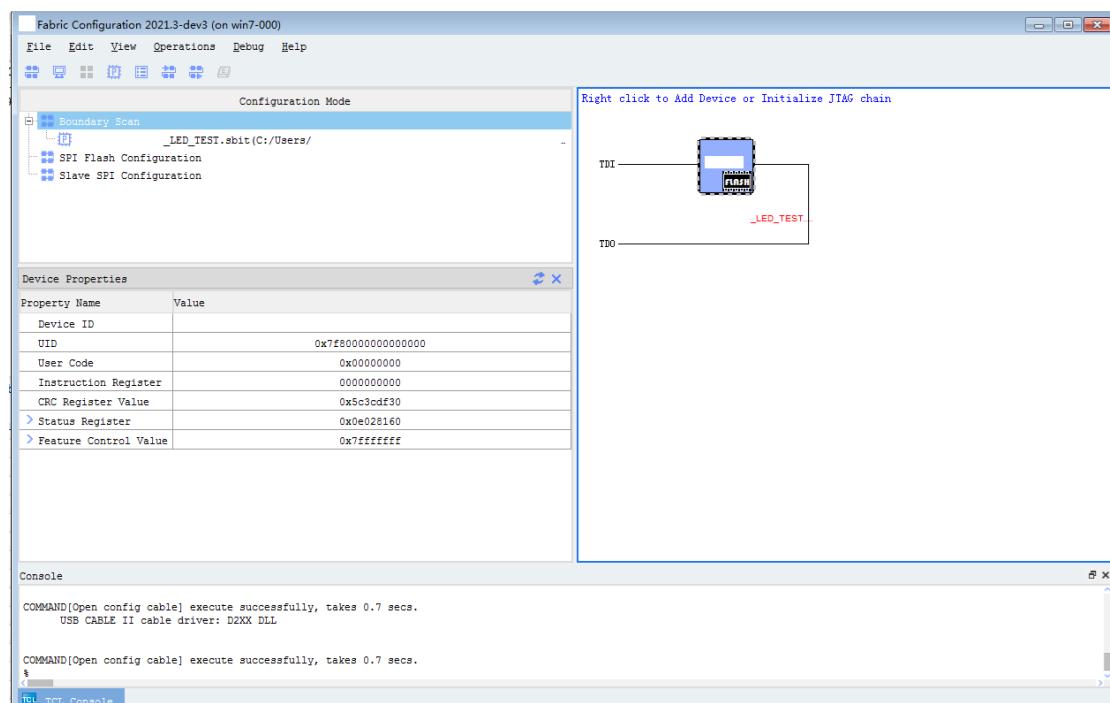


图 3-6 为 JTAG 链上的器件添加配置文件

3.6 编程逻辑位流文件

在配置逻辑位流文件后，便可将该文件下载到器件中。在执行下载操作之前要确保已经连上 Jtag Server 和已经配置了位流文件。在满足下载的条件后，首先选中初始化链检测到的 FPGA 器件，然后在该 FPGA 上单击鼠标右键，将会弹出如图 3-7 所示的菜单，通过鼠标左键单击“Program”操作，软件开始将所配置的位流文件 pgl50h.sbit 载入到 PGL50H 器件中。

下载的进度条显示完成后可观察板上的 DONE 灯是否亮着，如果亮着表示位流已经成功下载并运行；或者通过 Device Properties 窗口的 Status Register 栏查看状态寄存器值中的 done 位，如果 done 位置 1 说明位流已成功下载并运行。下载成功后，器件属性窗口的信息被刷新，如图 3-8 所示。

注：状态寄存器的详细说明见第五章附录->第二节状态寄存器各位表示的功能

注：对于某些器件的 done 管脚有可能被复用成其他功能，只能通过状态寄存器来判断下载是否成功。

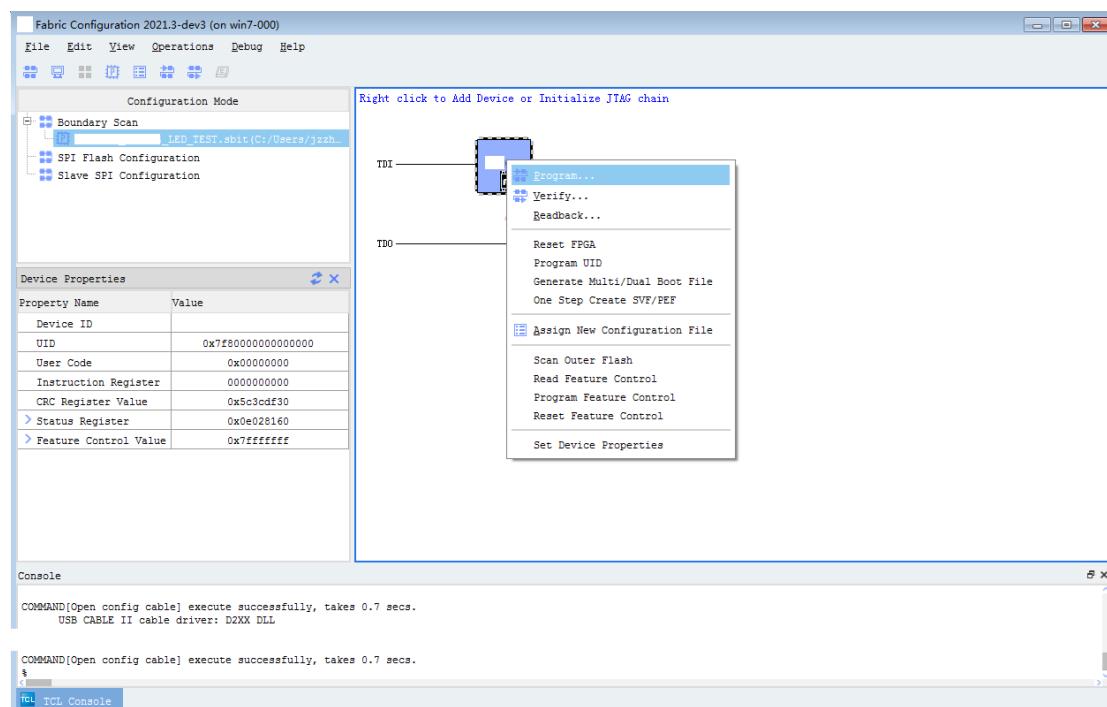


图 3-7 在 FPGA 上单击右键的菜单

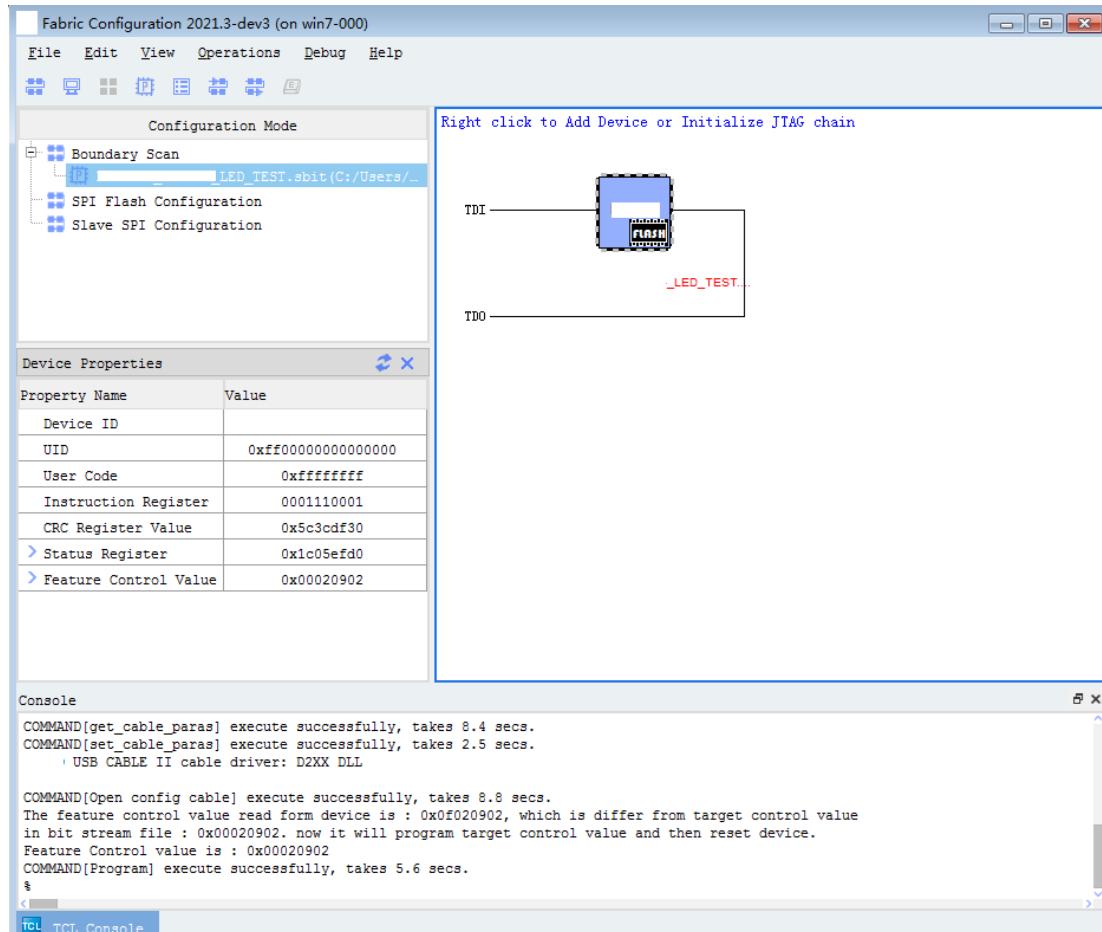


图 3-8 下载配置文件成功

3.7 校验逻辑位流文件

在下载位流文件操作完成后，为了确保所下载的位流文件在下载过程中没有出现差错，可通过校验操作来验证。该操作主要将所配置的位流文件中的信息和下载到 FPGA 器件中的逻辑信息相比较，如果两者一致，则说明校验成功，否则校验失败。首先选中需要进行校验的 FPGA 器件，然后在该 FPGA 上单击鼠标右键，会弹出如图 3-7 所示菜单，然后通过鼠标左键单击“Verify”，软件会开始进行校验，校验成功后如图 3-9 所示：

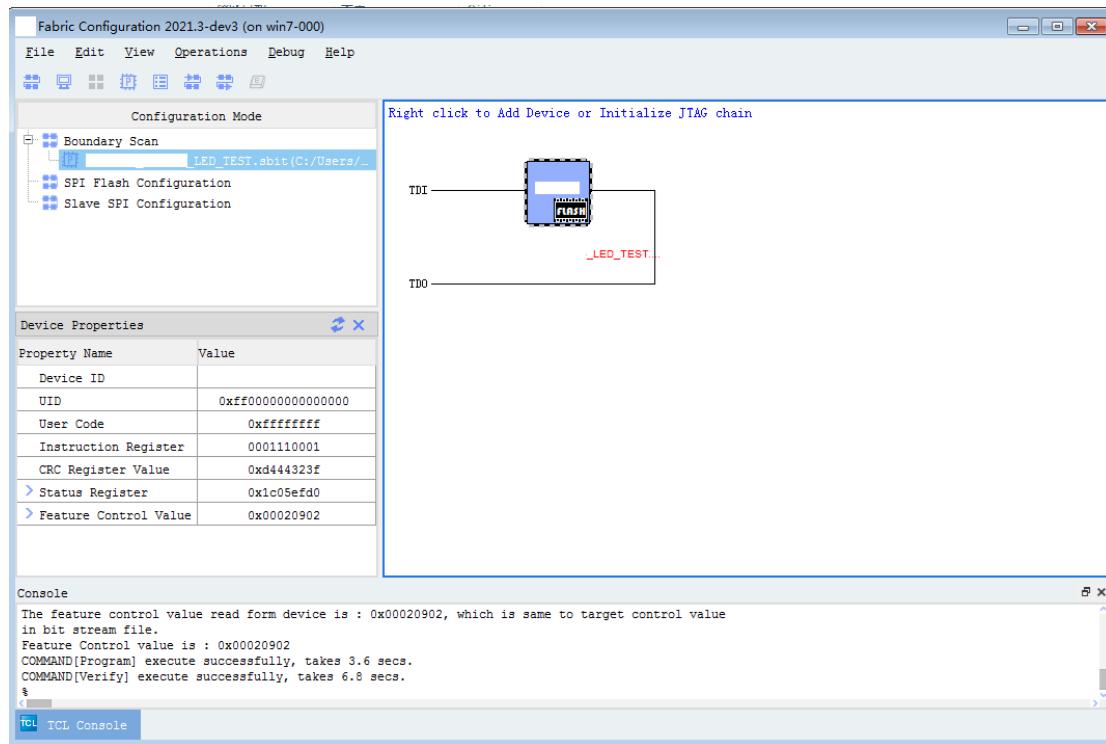


图 3-9 校验下载配置的文件

4 Fabric Configuration 软件说明

4.1 用户界面介绍

4.1.1 用户主界面

在 PDS 界面的工具栏点击  可启动 Fabric Configuration 软件，或者在 PDS 软件的安装目录下的 bin 目录双击“cdt_cfg.exe”启动该软件，用户也可以在操作系统的 shell 终端启动，切换到可执行程序所在的目录，输入命令 cdt_cfg.exe 即可，更多使用参数可以输入命令 cdt_cfg.exe –help 查看。软件启动后如图 4-1 所示：

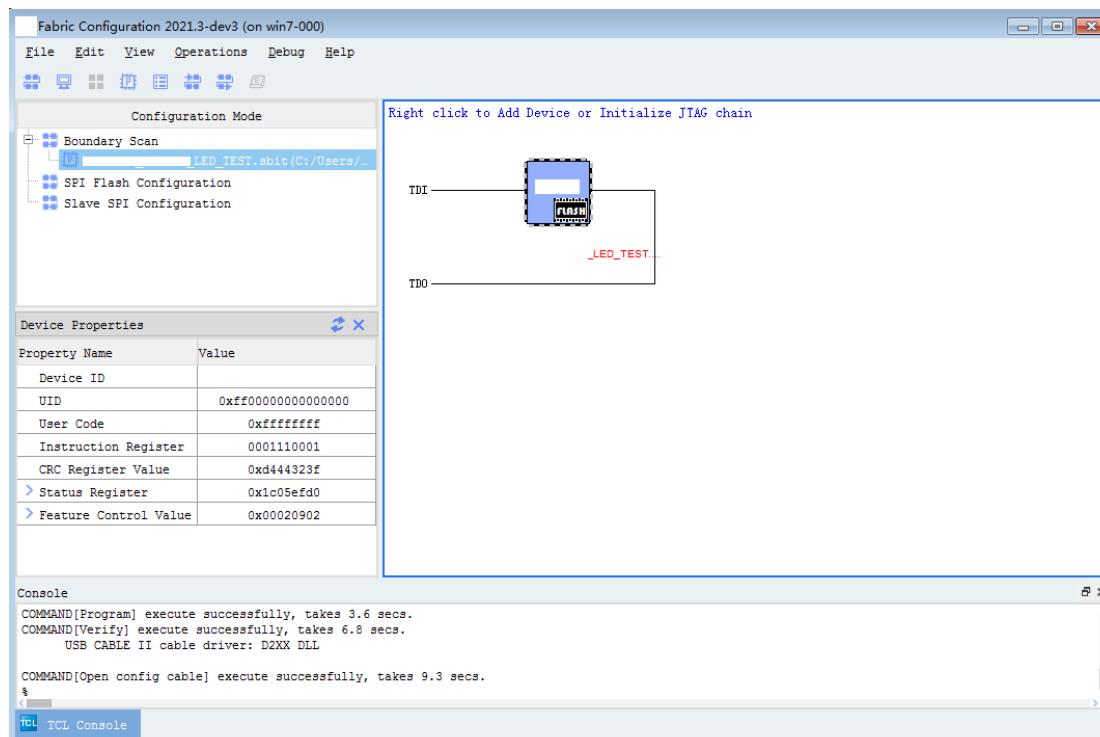


图 4-1 Fabric Configuration 主界面

- ◆ 菜单栏：主要包括文件（File）、编辑（Edit）、视图（View）、操作（Operations）、调试（Debug）和帮助（Help）6个下拉菜单。其使用方法和常用的 Windows 软件类似。
- ◆ 工具栏：主要包含了常用功能的快捷按钮，方便用户的操作。
- ◆ 工作区：显示 JTAG 链上扫描到的器件，或者用户手动添加的器件。
- ◆ 信息显示区：显示 Fabric Configuration 中的处理信息，如操作步骤信息、警告信息和错误信息等。
- ◆ 器件属性窗口：显示器件的基本属性信息，如 ID、指令寄存器值、状态寄存器值等。

4.2 菜单栏与工具栏

4.2.1 File 菜单

如图 4-2 所示，该菜单主要涉及与 Jtag Server 的通信及扫描器件。

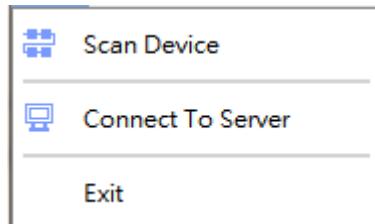


图 4-2 File 菜单

【Scan Device】: 用于检测并初始化器件, 如果在 Configuration Mode 窗口选择 Boundary Scan 模式, 则扫描 JTAG 链上所有的如果器件的配置电路 JTAG 检测正确, 会将链上扫描到的所有器件列于工作区内; 如果在 Configuration Mode 窗口选择 SPI Flash Configuration 模式, 则扫描所连接的 SPI 器件, 并将该器件列于工作区内。

【Connect To Server】 : 进行 JTAG Server 连接操作。

【Exit】 : 用于退出 Fabric Configuration 软件。

4.2.2 Edit 菜单

如图 4-3 所示:

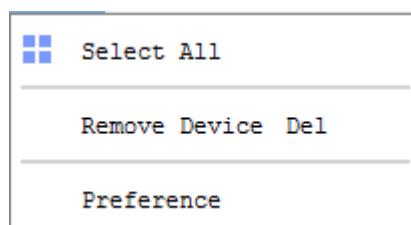


图 4-3 Edit 菜单

【Select All】 : 选中工作区内的所有器件。

【Remove Device】 : 用于将选中的器件从界面上显示的 JTAG 链中删除, 快捷键为“Delete”键。

【Preference】 : 设置用户自定义信息, 并在标题栏进行显示, 用以标识当前软件的特性; 控制是否在器件属性窗口显示 CRC 值、Efuse 值、是否默认修改 idcode (pgc) 和是否默认生成 pef 文件。

4.2.3 View 菜单

如图 4-4 所示:

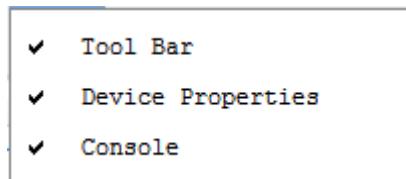


图 4-4 View 菜单

【Tool Bar】：用户设置工具栏的显示与隐藏。

【Device Properties】：用户设置器件属性窗口的显示与隐藏。

【Console】：用户设置 Console 窗口的显示与隐藏。

4.2.4 Operations 菜单

如图 4-5 所示：

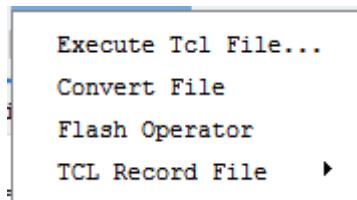


图 4-5 Operations 菜单

【Execute Tcl File】：用于执行 Tcl 文件。

【Convert File】：用于文件转换，主要有：①将 FPGA 位流文件转换成 Flash 下载文件 sfc 格式；②用于合成功能级联数据流；③用于合成功能远程升级数据流，多启动数据流，多功能组合数据流，主自加载双启动数据流；④用于生成功能级联 svf/pef 文件。

【Flash Operator】：用于新增用户 FLASH。

【TCL Record File】：用于记录 Tcl 命令并生成用户指定的 Tcl 脚本（包含创建和附加 TCL 文件两种方式）。

4.2.5 Debug 菜单

如图 4-6 所示

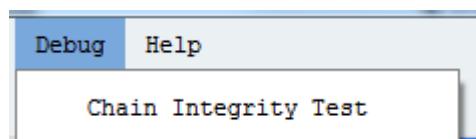


图 4-6 Debug 菜单

【Chain Integrity Test】：JTAG 链完整性测试，测试当前 JTAG 链是否为通路。

4.2.6 Help 菜单

如图 4-7 所示

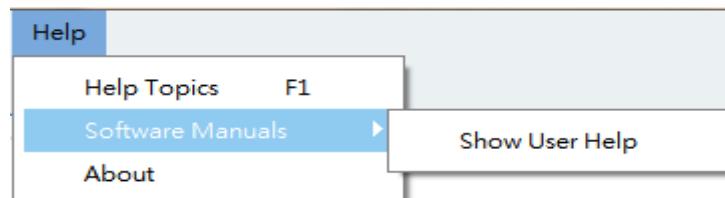


图 4-7 Help 菜单

【About】：用于显示 Fabric Configuration 版本信息。

【Show User Help】：用于打开 Configuration 用户帮助文档。

【Help Topics】：用于打开 Pango Assistance 的帮助信息。

4.2.7 工具栏基本操作

如图 4-8 所示



图 4-8 工具栏

【Scan Device 】：对应菜单栏 File-> Scan Device。

【Connect To Server 】：对应菜单栏 File->Connect To Server。

【Select All 】：对应菜单栏 Edit->Select All。

【Assign New Configuration File 】：对 FPGA 器件或者 SPI 器件进行逻辑位流文件配置。

【Add Pango Device 】：通过选择逻辑位流文件来添加 FPGA 器件。

【Program 】：对所选中的 FPGA 器件或者 SPI Flash 器件进行编程下载。

【Verify 】：对所选中的 FPGA 器件或者 SPI Flash 器件进行校验操作。

【Erase 】：对所选中的 Flash 器件进行擦除操作。

4.2.8 Device Properties 窗口

如图 4-9 所示，器件属性窗口用于显示器件的基本属性信息，如 ID、状态寄存器值等。用户在软件进行每一个操作后都会实时去更新这些属性信息。

Device Properties	
Property Name	Value
Device ID	
UID	0x00000000000000000000
User Code	0x00000000
Instruction Register	0000000000
CRC Register Value	0x5c3cdf30
➤ Status Register	0x00000000
➤ Feature Control Value	0x00000000

图 4-9 Device Properties 窗口

【Device ID】：显示 FPGA 器件的唯一识别码。

【UID】：显示 FPGA 器件序列码。

【User Code】：显示用户当前位流的逻辑版本或功能信息。

【Instruction Register】：显示 FPGA 当前状态的指令寄存器值。

【CRC Register Value】：显示当前 FPGA 所配位流的 CRC 值。

【Status Register】：显示 FPGA 当前状态寄存器的值。

注：指令寄存器的详细说明见第五章附录->第一节指令寄存器各位表示的功能

注：状态寄存器的详细说明见第五章附录->第二节状态寄存器各位表示的功能

4.3 功能说明

4.3.1 Jtag 边界扫描添加器件

在主界面的左侧 Configuration Mode 窗口，选择 Boundary Scan 模式，然后在工作区单击鼠标右键，会弹出如图 4-10 所示菜单，左键单击 Scan Device，或者在 File 菜单或工具栏中左键单击 Scan Device，都将会执行初始化链的操作。



图 4-10 工作区鼠标右键单击菜单

如果初始化链成功，会将链上扫描到的所有器件列于工作区内，在工作区中器件下显示相应的器件型号，并弹出对话框使用户能够为器件添加配置文件，如图 4-11 所示：

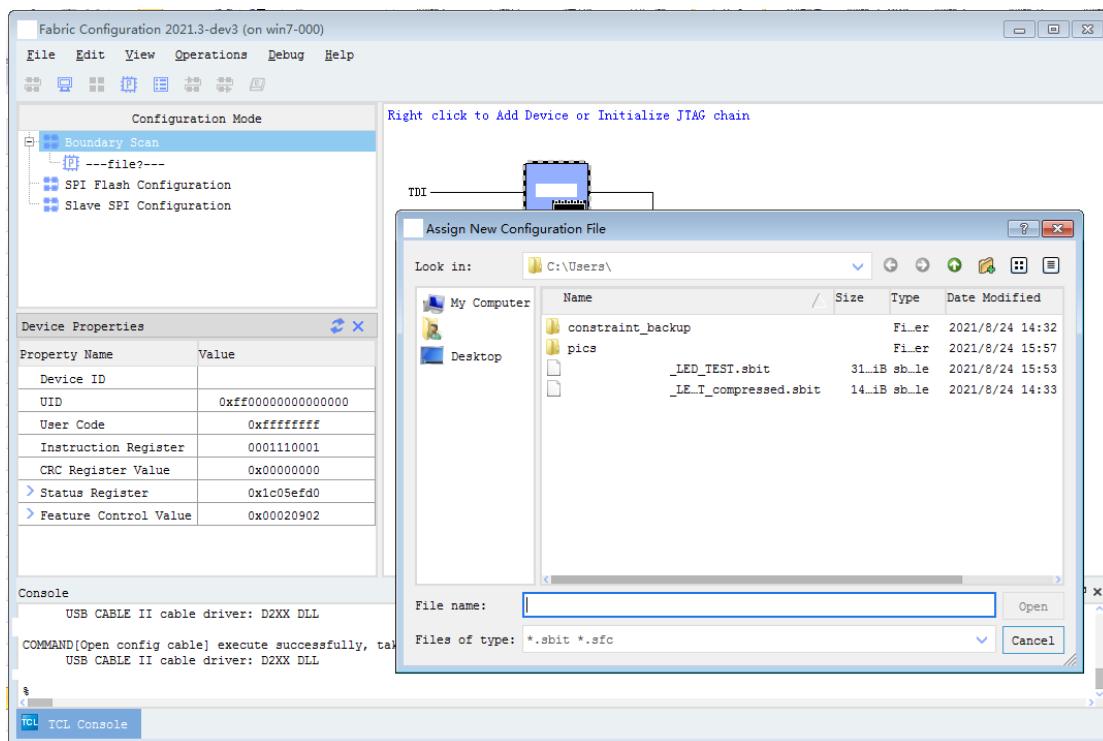


图 4-11 初始化链并添加.sbit 文件

若所选配置文件与初始化链检测到的器件匹配，则添加该配置文件，并在信息栏中提示所载入文件的绝对路径，如图 4-12 所示（PGC 器件可以通过扫链添加 sfc 文件，若配置的是 sfc 文件则只能对嵌入式 Flash 进行操作，无法对 cram 进行操作）。若所选配置文件与初始化链检测到的器件不匹配（sbit 文件与实际器件不符），Fabric Configuration 会阻止该文件的添加，并弹出警告提示用户，如图 4-13 所示。如果初始化链不成功，其弹出的对话框如图 4-14 所示。导致初始化链失败的原因有多种，解决方法一般为：首先，检查 USB 下载电缆是否与主机正确连接；其次，检查下载电缆的 USB 驱动是否正确安装；再次，检查 FPGA 芯片所在单板是否正常工作，并检查配置电路的 JTAG 链是否完整；最后，尝试重启 Fabric JtagServer 软件。

若扫描链扫到了未知器件，即链上存在未知器件时，将会要求用户必须设置未知器件的指令长度，默认值为 10，如图 4-15。用户必须保证其设置的未知器件的指令长度正确性，否则链上的一系列操作将可能发生未知的错误。用户可根据需要个性化设置未知器件的器件名称。用户还可以通过添加与链上器件完全匹配的*.jci 文件来设置链上未知器件的指令长度和器件名称，添加文件后的链信息会在界面实时更新，检查无误后可以点击 OK 来确认设置。除此之外，如果用户没有该链的*.jci 文件，用户可以在界面手动设置完链信息后，勾选保存成*.jci 文件，以用于配置或传输。

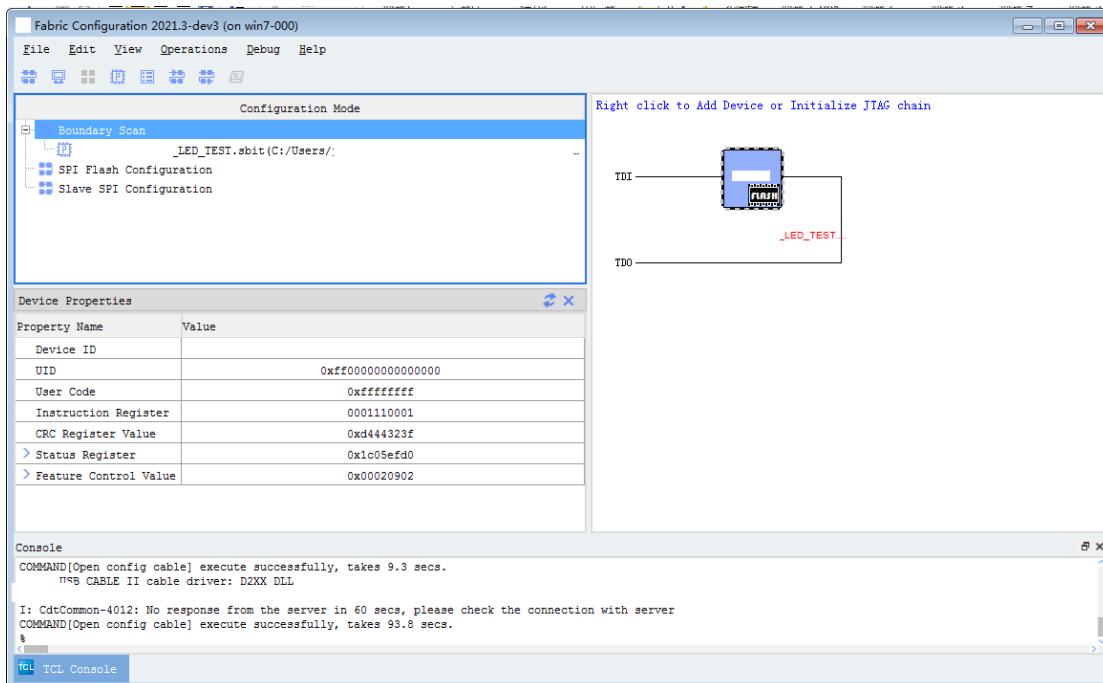


图 4-12 配置正确的位流文件

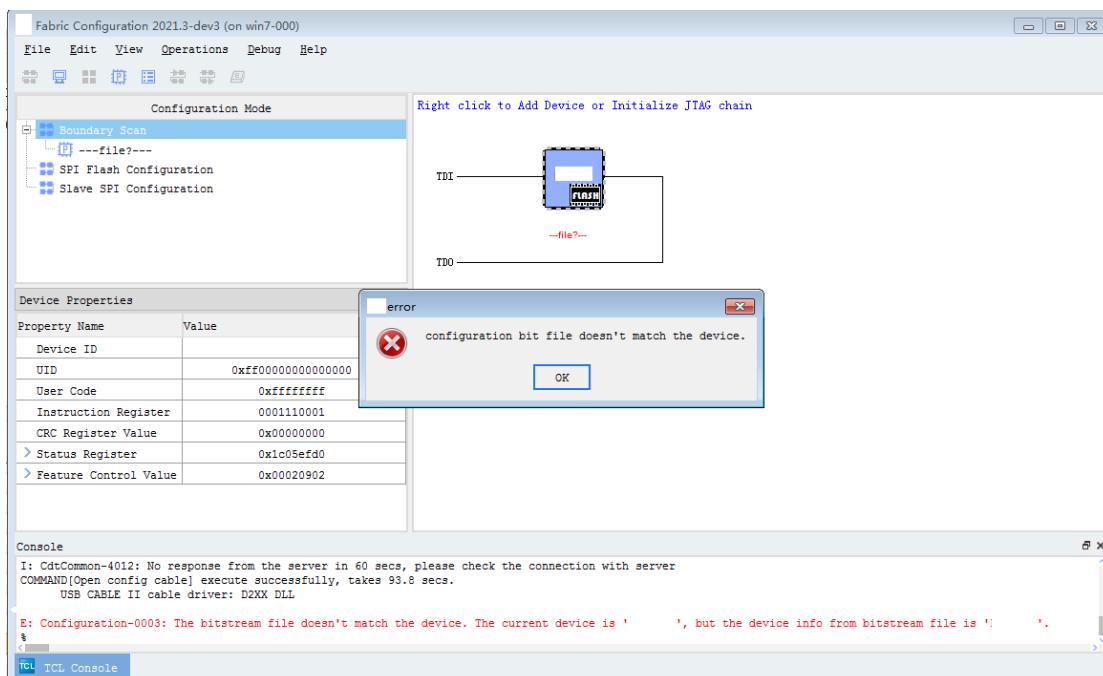


图 4-13 配置错误的位流文件

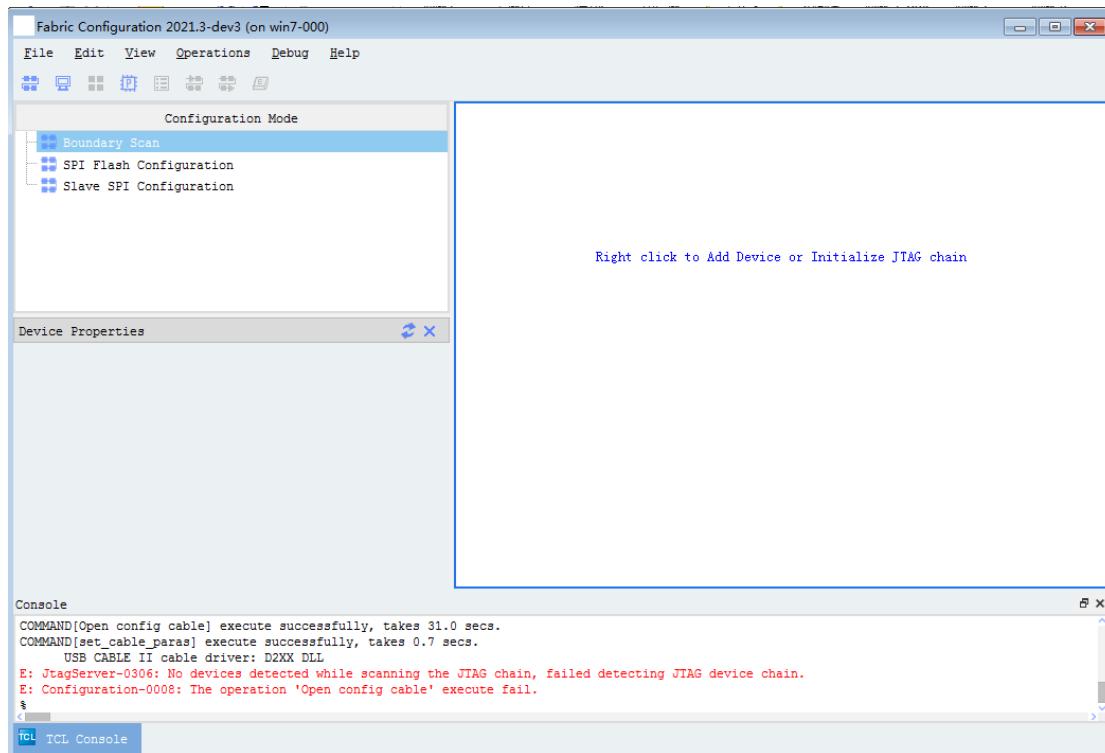


图 4-14 初始化链失败

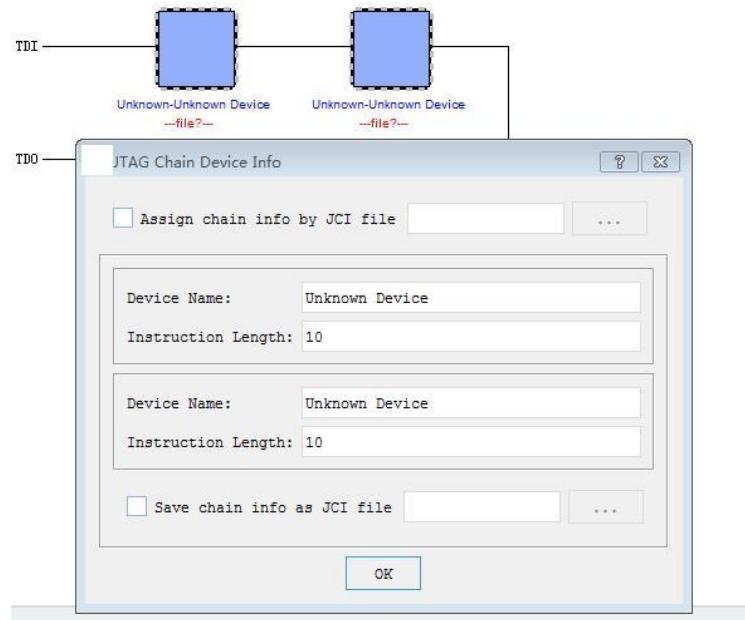


图 4-15 扫描未知器件设置指令长度

4.3.2 解析目标位流添加器件

除了通过扫描 Jtag 链的方式外，用户还可以通过选择逻辑位流文件的方式来手动添加 FPGA 器件。具体操作为：在工作区鼠标右键单击，会弹出如图 4-10 所示菜单，左键单击“Add Pango Device”，或者在工具栏中左键单击“Add Pango Device”，执行添加器件的操作。如果

链上已有器件存在，再添加时需要在要添加器件的地方单击左键，然后右键选择添加器件，这样会将器件添加到 JTAG 指定的位置。

如果添加的位流文件所对应的器件为 Fabric Configuration 所支持的器件，则将其加入工作区，与初始化边界扫描链成功的图 4-12 相同；若该配置文件不被识别，则提示如图 4-16 所示：

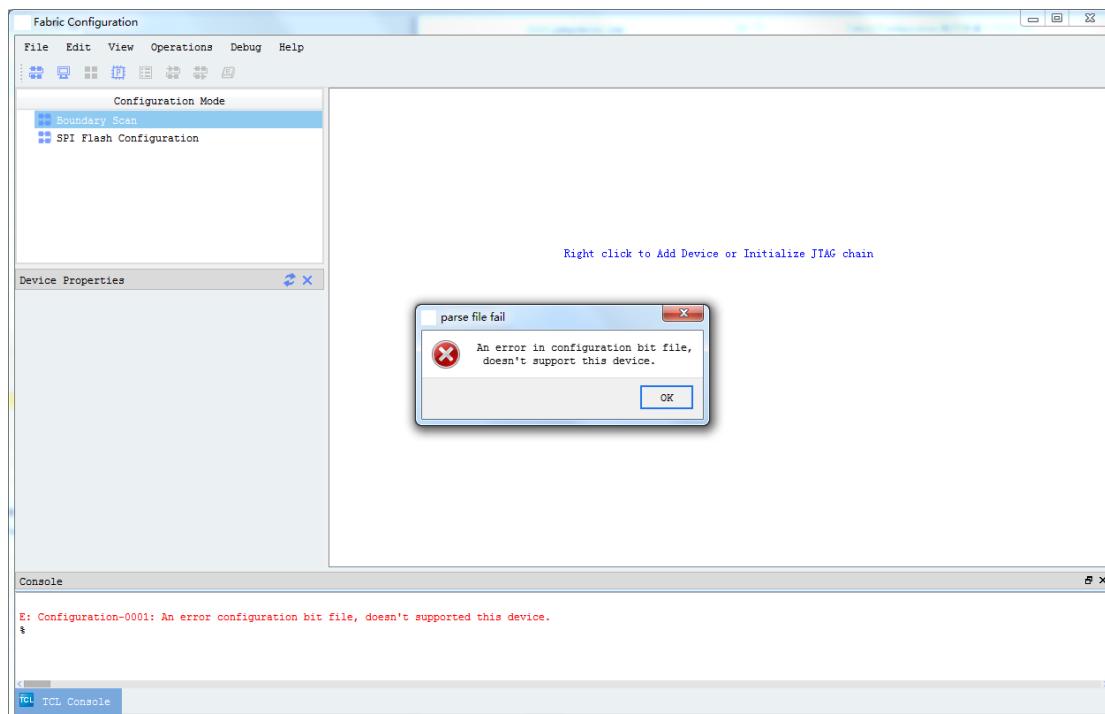


图 4-16 添加不支持器件的位流文件

4.3.3 配置新的位流文件

如果用户想为 JTAG 链上的器件重新添加一个配置文件，可以使用该功能。用户首先选中需要重新添加配置文件的器件，在该器件上右键单击，会弹出如图 4-17 所示菜单，然后左键单击“Assign New Configuration File”，或者在工具栏中左键点击“Assign New Configuration File”都可以执行重新添加配置文件的操作。

如果成功执行，则原有的配置文件将被替换；如果失败，则说明该配置文件与目前侦测到 JTAG 链上的器件不匹配，会弹出如图 4-18 所示的对话框。

注：Configuration 软件中对所有文件的名称和文件路径有如下限制：

- 1、文件夹名：只允许字母数字 下划线（_）杠（-）点（.）@ 和空格（ ），但空格不能出现在路径名首尾。
- 2、文件名：只允许字母数字 下划线（_）杠（-）点（.）。

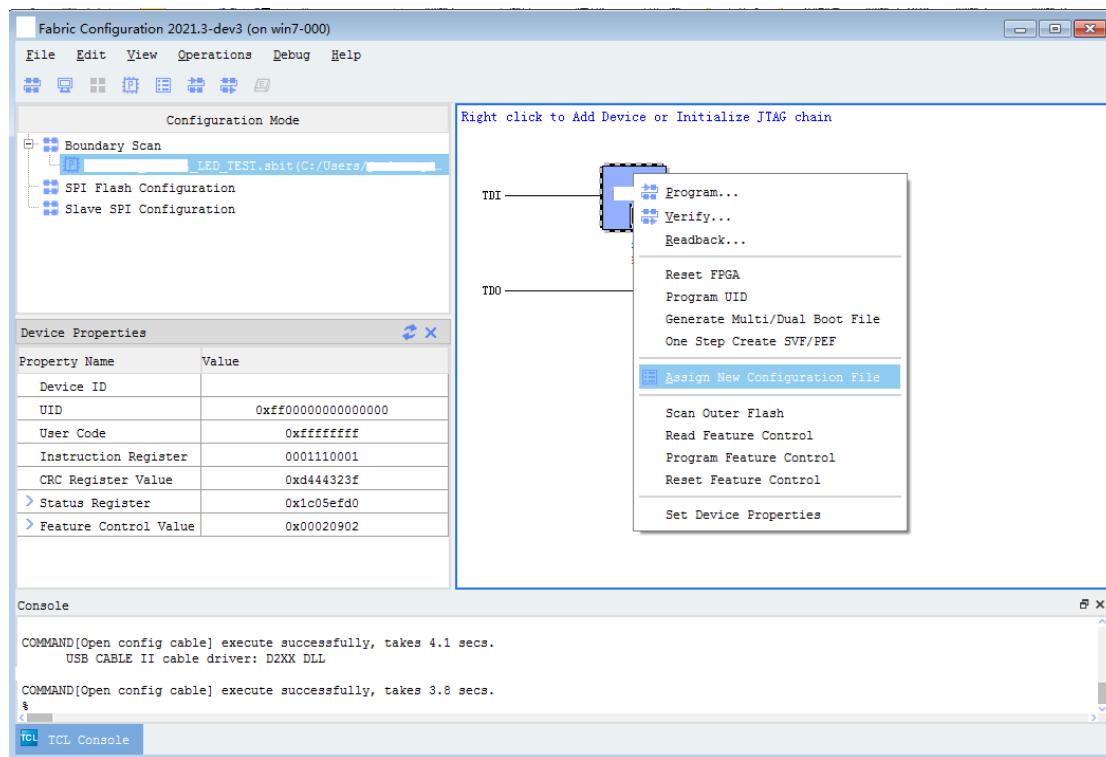


图 4-17 在器件上单击右键的菜单

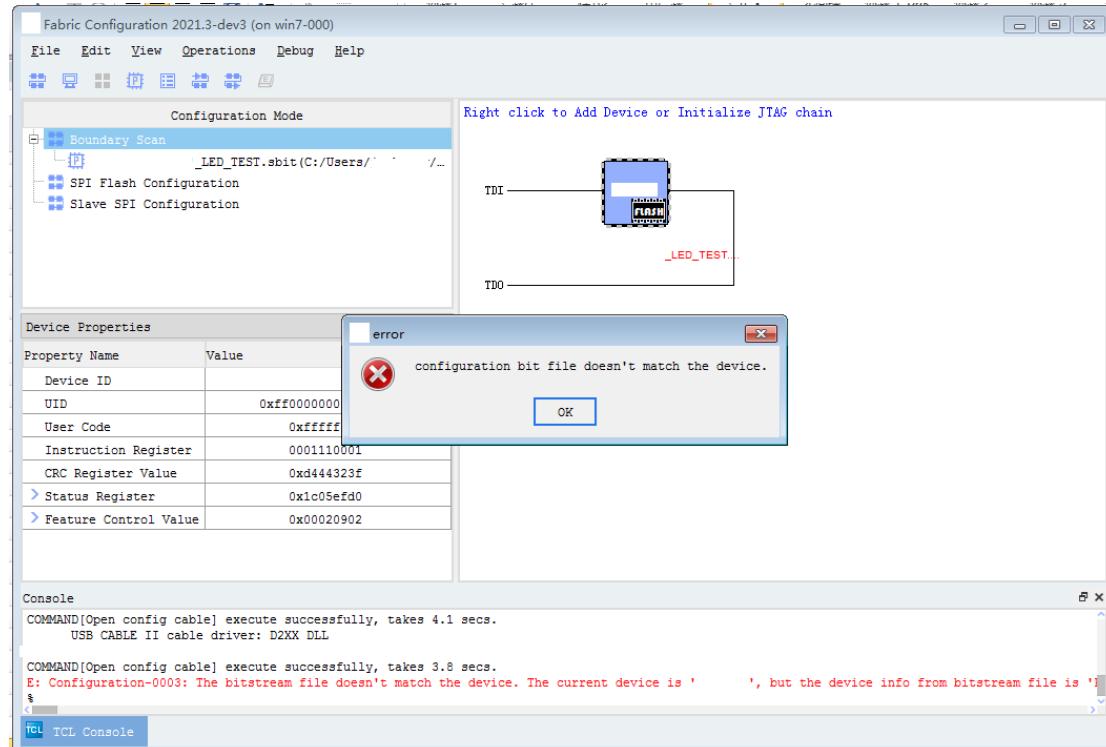


图 4-18 添加了不匹配的配置文件

4.3.4 Jtag 编程 FPGA 器件

在配置逻辑位流文件后，便可将该文件下载到器件中。在执行下载操作之前要确保已经连上 Jtag Server 和已经配置了位流文件。在满足下载的条件后，首先选中 Jtag 链上的 FPGA 器件，然后在该 FPGA 上单击鼠标右键，会弹出如图 4-17 所示菜单，左键单击“Program”，或者在工具栏中左键点击“Program”，Fabric Configuration 会按照用户之前的准备(JTAG 链上器件、配置文件)，向所选中的器件下载配置文件。

若下载成功，如图 4-19 所示；若不成功，有很多种可能，可能是硬件或连接问题或者配置文件与器件不匹配，如图 4-20 所示。

注：编程秘钥文件(nky 文件)操作见本章第 10 小节 Jtag 编程 eFuse 寄存器。

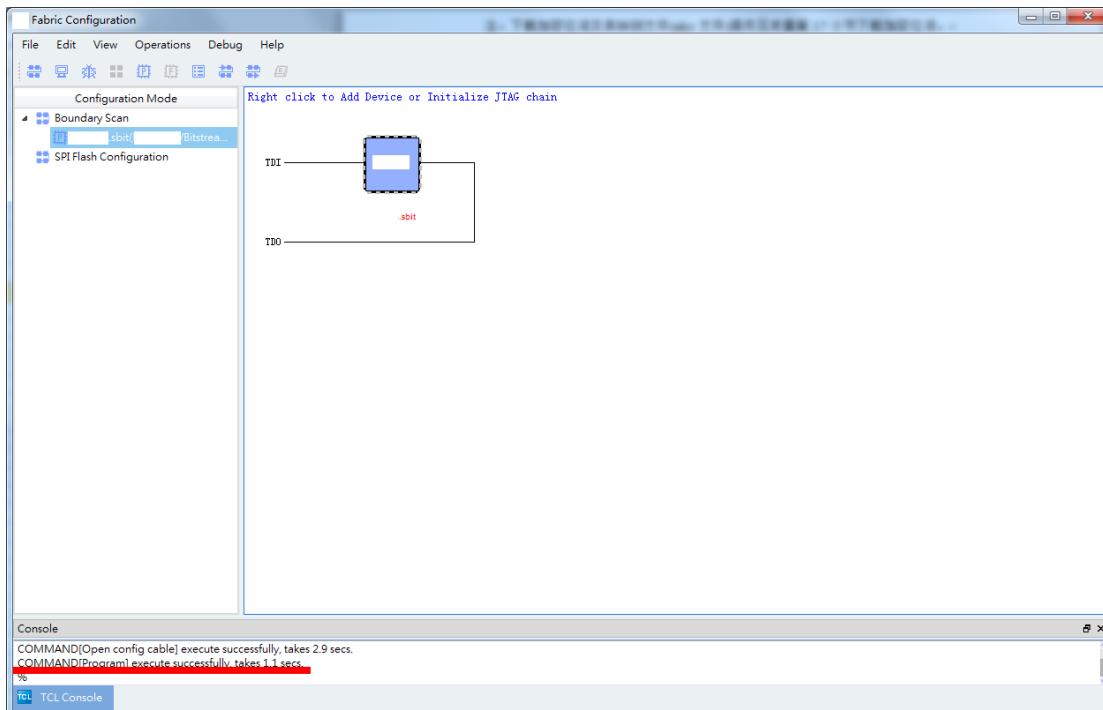


图 4-19 下载配置文件成功

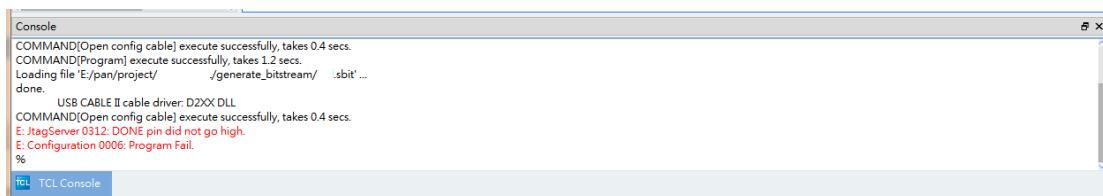


图 4-20 下载失败

4.3.5 Jtag 校验 FPGA 器件

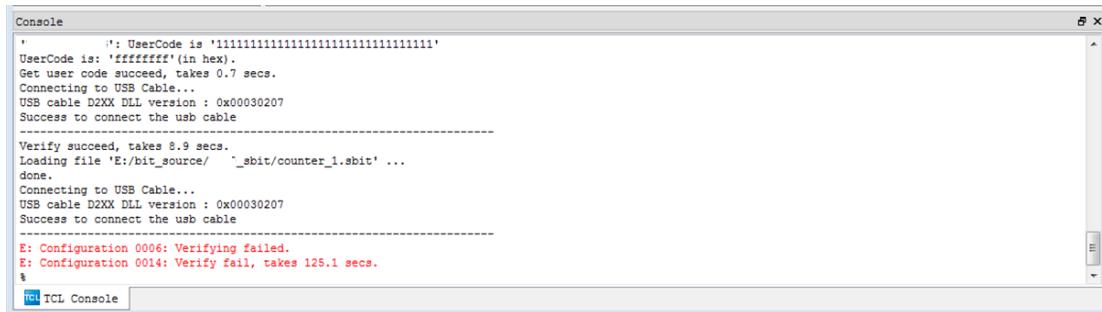
在编程位流文件操作完成后，为了确保所下载的位流文件在下载过程中没有出现差错，可通过校验操作来验证。校验的过程首先是回读下载入器件的数据，将读出的数据与载入

的数据相比较，用来验证载入数据的正确性。

FPGA 的校验必须提供校验所需的 ssmk 文件，在 verify 操作的执行过程中用户可中止该操作，但界面会提示校验失败的信息。对于 FPGA，若校验失败，则会在信息栏中显示校验出错信息并将校验的结果保存在 cfg_verify_result.sbit 文件中(应用程序 cdt_cfg.exe 所在的目录的 log 文件夹中或者工程目录所在的目录的 log 文件夹中)，同时会将回读文件已回读时间命名保存在 rd_年_月_日_时_分_秒文件中（和 cfg_verify_result.sbit 在同层目录）。如图 4-21 所示；若校验成功，如图 4-22 所示。

注：verify 操作只比对非 DRM 数据。

注：verify 操作只使用于所配置的位流为非压缩位流和非加密位流。



```

Console
'          ':UserCode is '11111111111111111111111111111111'
UserCode is: 'ffffffff'(in hex).
Get user code succeed, takes 0.7 secs.
Connecting to USB Cable...
USB cable D2XX DLL version : 0x00030207
Success to connect the usb cable
-----
Verify succeed, takes 8.9 secs.
Loading file 'E:/bit_source/_sbit/counter_1.sbit' ...
done.
Connecting to USB Cable...
USB cable D2XX DLL version : 0x00030207
Success to connect the usb cable
-----
E: Configuration 0006: Verifying failed.
E: Configuration 0014: Verify fail, takes 125.1 secs.
'
TCL Console

```

图 4-21 FPGA 校验失败



```

Console
Bit 0:      0 id_err
Status Register: '0009fbf0'(in hex).
Read status succeed, takes 0.7 secs.
Connecting to USB Cable...
USB cable D2XX DLL version : 0x00030207
Success to connect the usb cable
-----
'          ':UserCode is '11111111111111111111111111111111'
UserCode is: 'ffffffff'(in hex).
Get user code succeed, takes 0.7 secs.
Connecting to USB Cable...
USB cable D2XX DLL version : 0x00030207
Success to connect the usb cable
-----
Verify succeed, takes 8.9 secs.
'
TCL Console

```

图 4-22 FPGA 校验成功

4.3.6 Jtag 回读 FPGA 器件

当用户所下载的位流文件为非压缩位流和非加密位流时，用户可将已经下载到 FPGA 芯片的位流数据回读回来并将其存储于用户所指定的文件中。

首先选中Jtag链上的FPGA器件，然后在该FPGA上单击鼠标右键，会弹出如图4-16所示菜

单, 左键单击“Read Back”, Fabric Configuration会弹出对话框让用户选择或者新建一个文件, 将回读出的数据存储入所选文件, 如图 4-23所示。

注: Read Back 操作只使用于所配置的位流为非压缩位流和非加密位流。

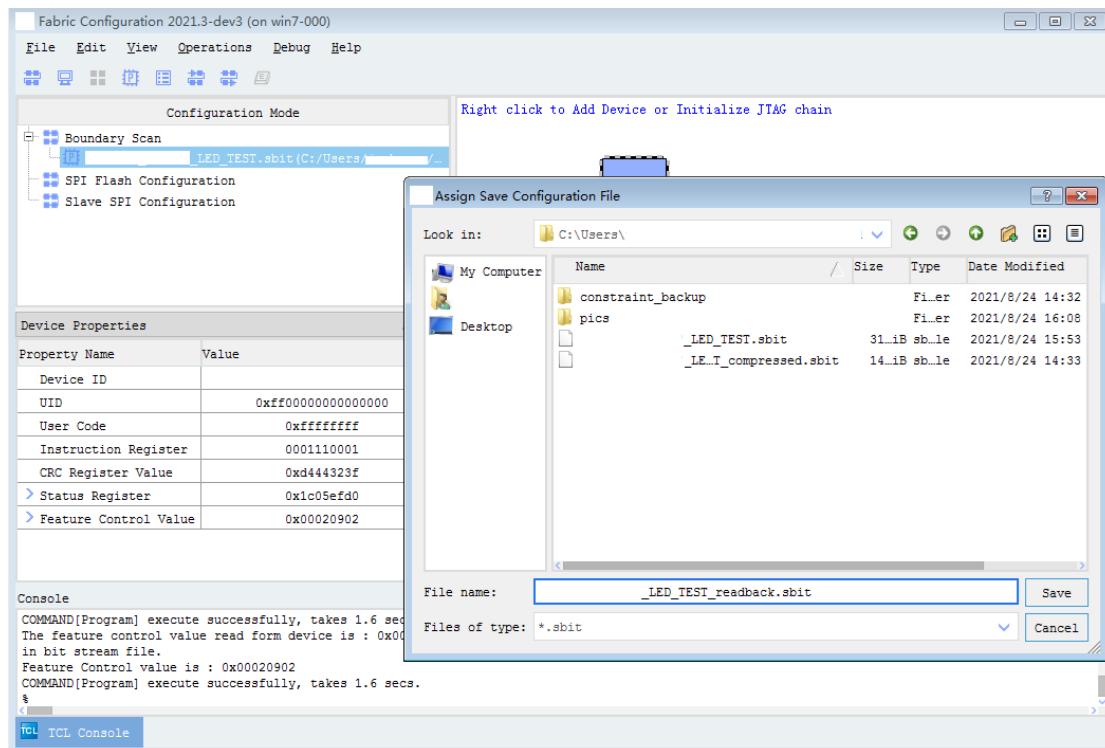


图 4-23 FPGA 回读

4.3.7 Jtag 操作 Compact 系列的特征控制位

Compact 系列可以通过配置器件的特征控制位来使用或实现一些特殊的功能或者接口。用户可以设置特征控制位并进行编程特征控制位, 复位特征控制位, 读取特征控制位等操作。用户可通过右击菜单, 点击相应的菜单进行操作当前器件的特征控制位。

注: PGC 的编程配置选项, 包含特征控制位, 特征控制位是与嵌入式 flash、外部 flash 相互配合使用, 才能完成 PGC 的配置。

注: 其他嵌入 Flash 的操作不需要复位, 一般来说, 特征控制位写一次即可。

注: 如果用户对特征控制位不是特别了解, 可直接勾选 default config 选项, 便能使芯片正常进行工作。

以主自加载模式加载 FPGA 的过程为例:

1、在编程配置选项中, 勾选 Enable Master Auto Mode In Configuration 选项, 并在界面

上点击 Program Feature Control，进行使能自加载模式控制。

2、操作嵌入式 Flash，通过配置位流文件（默认是当前器件的 sbit 文件），右击嵌入式 Flash 的图标点击 Program 进行数据的烧录。

3、复位 FPGA，通过电路板上的复位键进行复位，此时，FPGA 通过主自加载模式从嵌入式 flash 中进行加载数据流。

4.3.8 Jtag 操作 Compact 系列的嵌入式 Flash

Compact 系列器件内部带有 Flash 模块，支持如图 4-24 所示操作。对 Embed Flash 进行操作时，用户可通过右键菜单->Set Device Properties->编程选项中的 Embed Flash Properties 项进行嵌入式 Flash 的下载属性配置。具体设置信息请参考本章第 19 小节设置操作器件属性->设置 Compact 系列的操作器件属性。

注：当对嵌入式 Flash 进行下载操作时，位流文件与主器件相同，若用户想给嵌入式 Flash 配置不同于主器件的位流，可扫描 JTAG 链时添加指定的 sbit 文件或 SFC 文件即可。

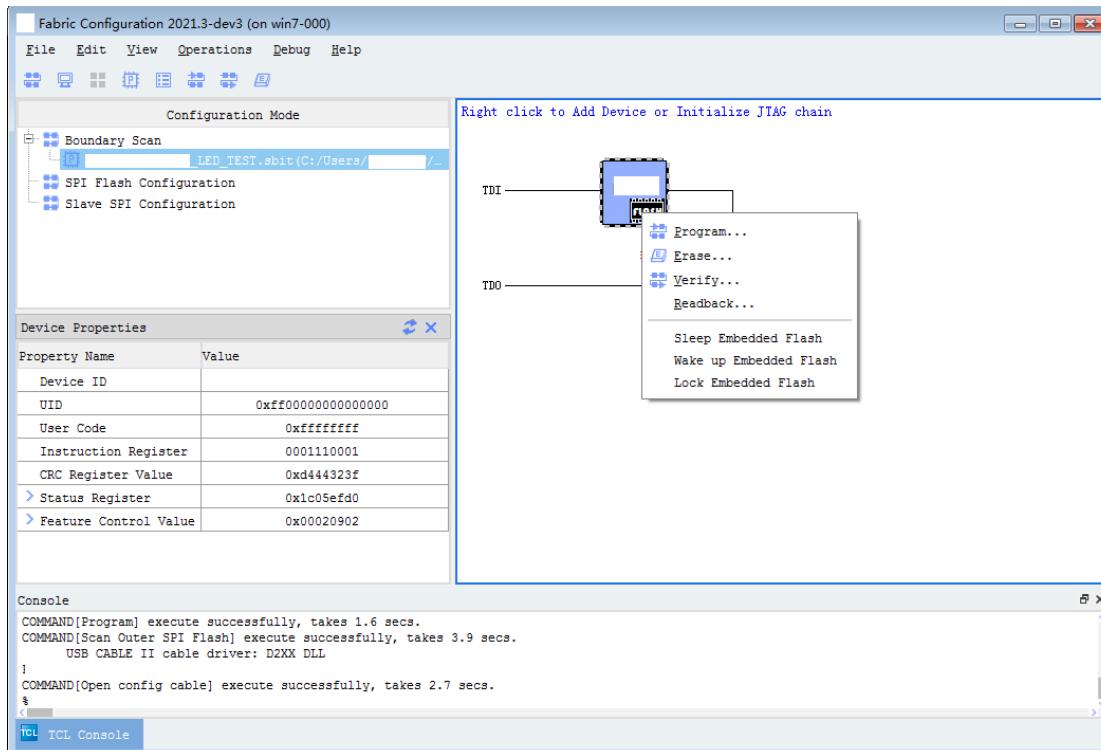


图 4-24 操作嵌入式 Flash

1、Program Embedded Flash

将位流文件下载进 flash 芯片，下载的位流默认和 PGC 芯片配置的位流一致；如果想下载指定的位流可通过扫描 JTAG 链时添加指定的 sbit 文件或 SFC 文件即可。

在进行编程下载时， 默认在下载后 Verify， 如果不想进行这项操作， 可通过右键菜单->Set Device Properties->Embed Flash Properties 去掉相应的勾选项来实现。

如果需要从某个指定页开始下载位流， 可通过设置 Embed Flash Start Page Index 来实现。假设设置值为 20，则从 Embed Flash 的第 20 页开始下载位流。

2、 Erase Embedded Flash

擦除 flash 的数据，有页擦除和全片擦除两种模式；默认情况下为整片擦除，可通过右键菜单->Set Device Properties->Embed Flash Properties ->Erase Entire Flash 来进行模式的选择。

如果需要从某个指定页开始擦除，可通过设置 Embed Flash Start Page Index 来实现。假设设置值为 20，则从 Embed Flash 的第 20 页开始擦除 Embed Flash，如果选择擦除的方式为整片擦除则该设置参数无效。

注：Compact 系列的嵌入式 Flash 擦除后所有存储单元全被置 0，SPI Flash 及外部 Flash 在擦除后所有存储单元均被置 1。

3、 Verify Embedded Flash

验证下载的位流是否与配置的位流相一致，首先根据所给的文件大小从 Flash 器件中读取出相应数据，然后将所读出的数据与载入的数据相比较，从而验证载入数据的正确性。若比较失败会生成 FlashReadBack_年_月_日_时_分_秒.sfc 文件中（文件会存储于 cdt_cfg.exe 软件同层目录）。

4、 Read Back Embedded Flash

该功能用于读取 Flash 中的数据并将所读取的数据存储于用户所指定的文件。如果下载的位流是从某个指定页开始下载位流的，那么在进行回读操作时，也需要设置从某个指定页开始回读位流。可通过设置 Embed Flash Start Page Index 来实现。

该功能还支持指定回读内容的大小，可通过右键菜单->Set Device Properties->Embed Flash Properties ->Embed Flash Read Back Size 来进行设置。假使 Embed Flash Start Page Index 的设置值为 20， Embed Flash Read Back Size 的设置值为 1000，则从 Embed Flash 的第 20 页开始回读，并且回读的内容大小为 1000 个字节。

5、 Sleep Embedded Flash

睡眠操作是让嵌入式 Flash 处于睡眠状态，不处理关于 flash 的任何操作。

6、 Wake up Embedded Flash

唤醒操作是让嵌入式 Flash 处于工作状态，进行处理相应的操作。

7、 Lock Embedded Flash

锁定操作是根据编程配置的起始地址和 flash 文件大小进行相关区域的锁定，不能进行读写。解除锁定状态需要擦除整片 flash。默认锁定范围为整片锁定。如果只需要从开始到某个指定页，可通过设置 Embed Flash Tail Page Index 来实现。假设设置值为 20，则锁定 Embed Flash 的第一页到第二十页。

4.3.9 Jtag 编程 eFuse 寄存器

1、编程秘钥文件

nky文件是对位流文件进行加解密的配置文件，nky文件中含有对位流文件进行解密的密钥。密钥的配置有两种方式：

①、inner key，可以多次向器件写入密钥配置信息，但是在器件断电的情况下，配置的密钥信息会消失，需要重新进行写入。

②、efuse key，只能够向器件写入一次密钥配置信息，之后信息不能更改并且信息会永久保存在器件中，在器件断电的情况下，配置的密钥信息不会消失。当inner key的nky文件下载到器件中后，其后的位流文件必须是通过nky文件中的密钥进行加密的位流文件，这样才会在器件中下载成功；否则，未加密的位流文件不会下载成功。efuse key的nky没有这个限制。

具体操作如下：

1、在右键点击弹出的菜单中选择“Program eFuse Registers”，将会弹出如下图 4-25，选择Program Key File操作类型，并点击Next，进入选择秘钥文件窗口。

2、在Configure Key File窗口中，用户在文件输入框选择将要编程的nky文件，软件会解析该秘钥文件是否与当前所选中的器件是否匹配，如果匹配将会把秘钥信息打印出来，如图 4-26所示。选择好秘钥文件后，点击Next。

3、在选择好秘钥文件后，在该summary页将会把所选择的操作类型及秘钥文件信息打印出来，供用户进行确认，如图 4-27所示。如果想修改秘钥文件则只需退回选择新文件即可。确认信息无误后，点击Finish即可进行编程。如果编程的秘钥为永久秘钥，因永久秘钥只能编程一次，因此需用户再次进行确认，如下图 4-28所示。

注意：永久秘钥进行编程后将不能再修改秘钥文件，请谨慎操作！

4、下载完秘钥文件 nky 后就可以进行相对应加密位流的配置了。选择目标加密位流，进行下载即可。若选择的加密位流是所下载的秘钥加密所得位流，则可以下载成功，反之则下载失败。

注：加密位流不可进行回读和校验操作。

注：配置秘钥文件后只能配置相对应的加密位流，但也可以配置其他非加密位流。

此外，对于Logos2和Titan2系列的器件还有编程BBRAM（电源供电密钥寄存器）的功能，用户可以将秘钥文件 (*.nky) 烧写到电源供电RAM中。只需要在Set Key Location页面选择BBRAM即可，如图 4-29。

注：选择烧写秘钥文件的eFUSE和BBRAM这两种寄存器，只能选择使用其中的一种寄存器，不能同时使用两者，否则容易产生错误。

注：永久秘钥进行编程后将不能再修改秘钥文件，请谨慎操作！

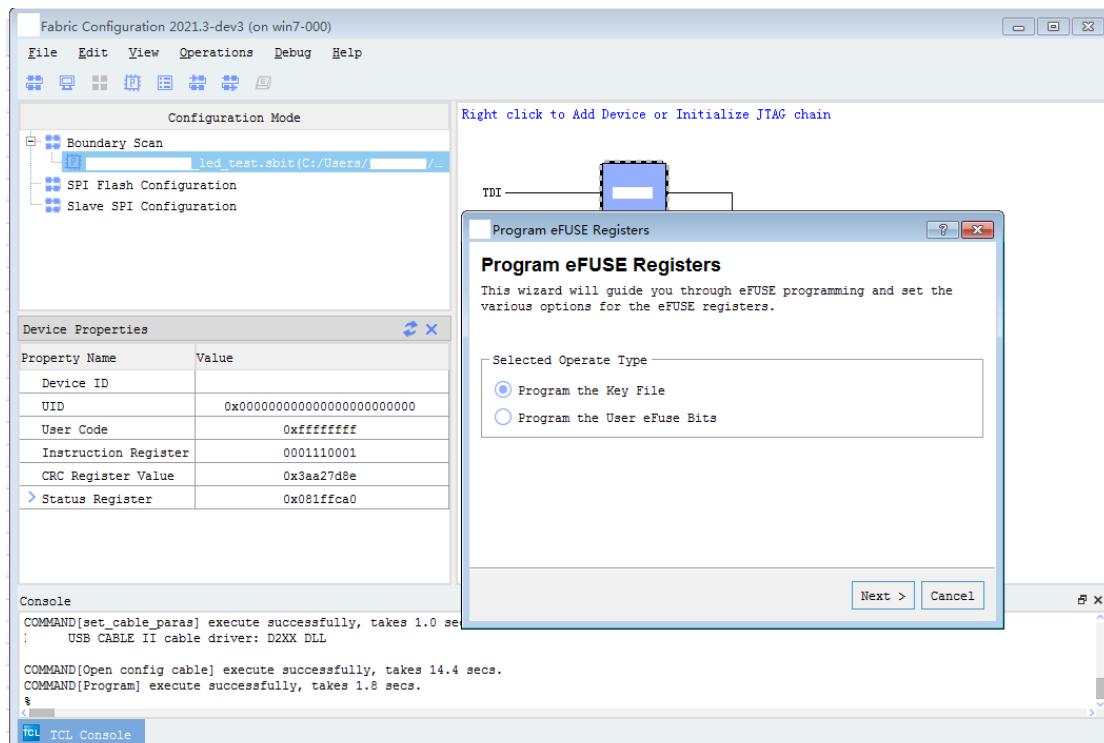


图 4-25 选择操作 eFuse 类型

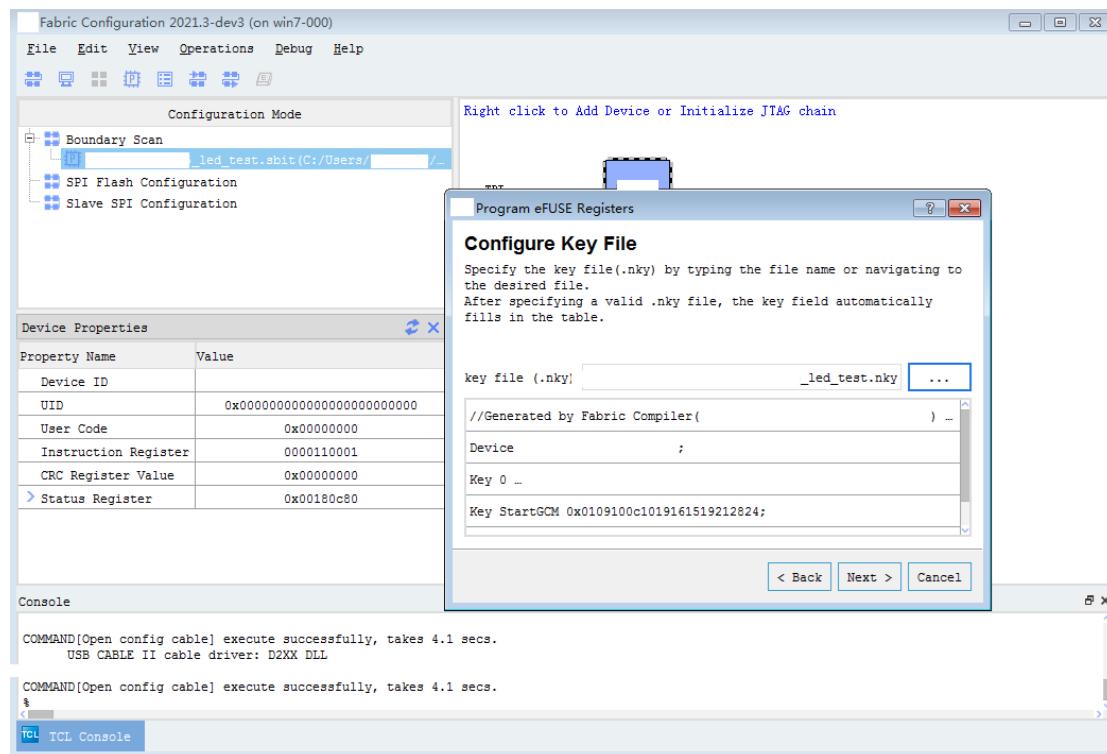


图 4-26 配置秘钥文件

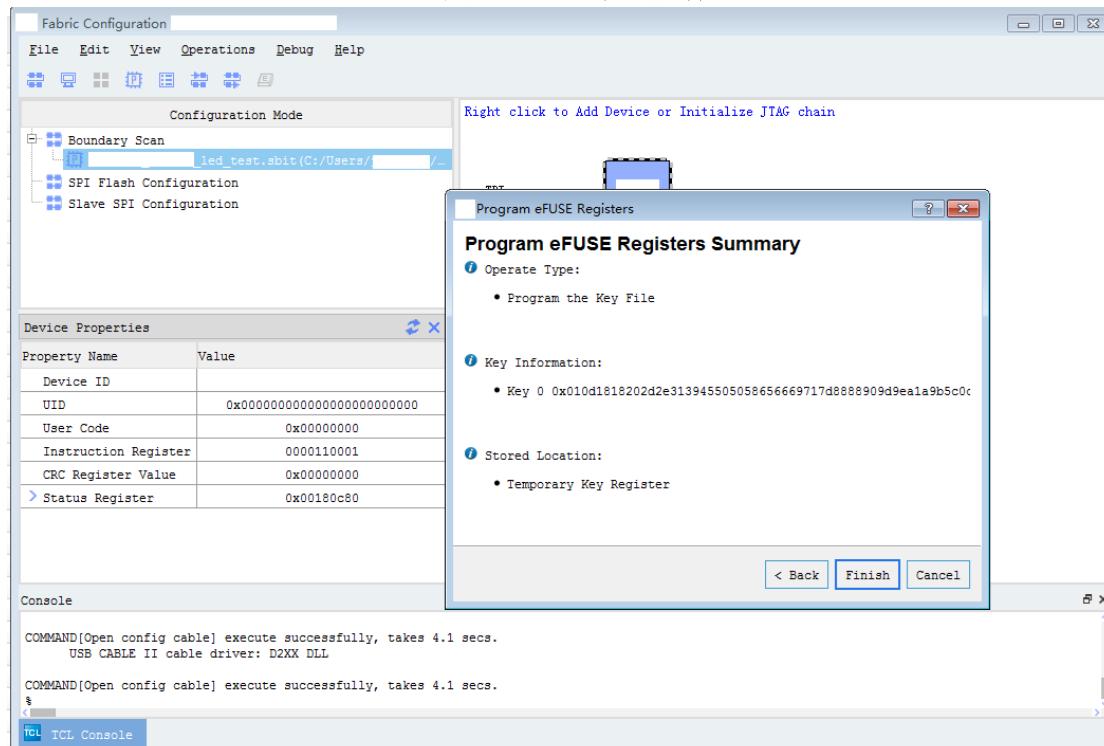


图 4-27 配置秘钥信息

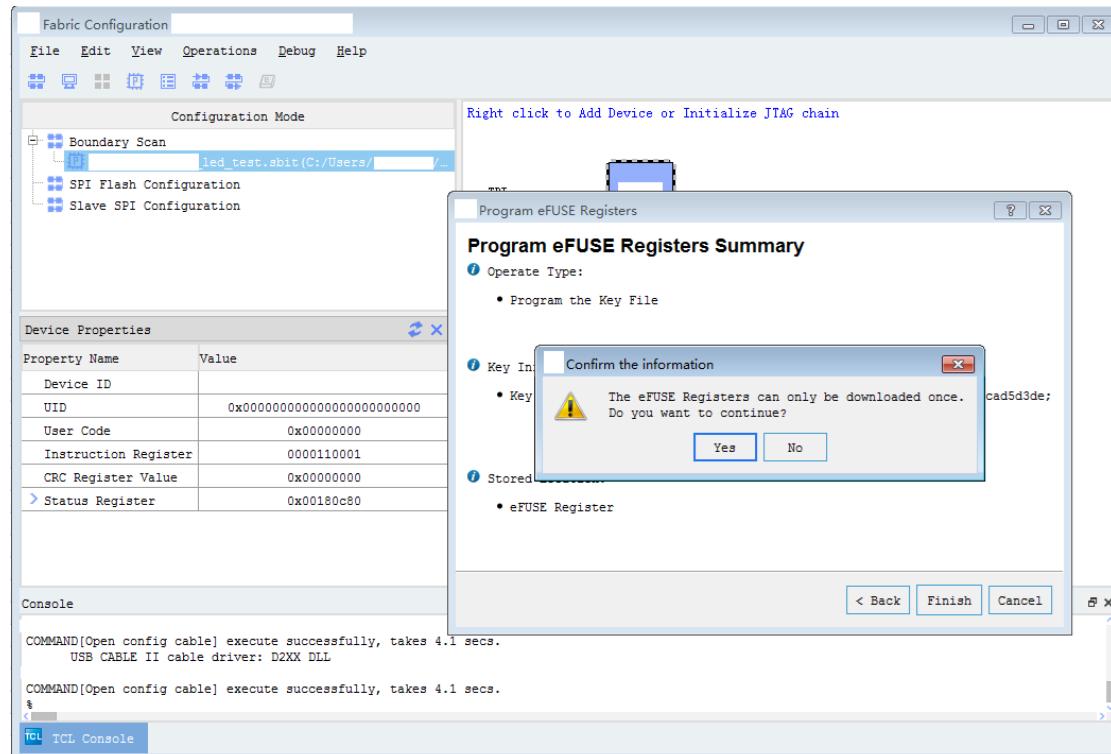


图 4-28 配置永久秘钥时的警告

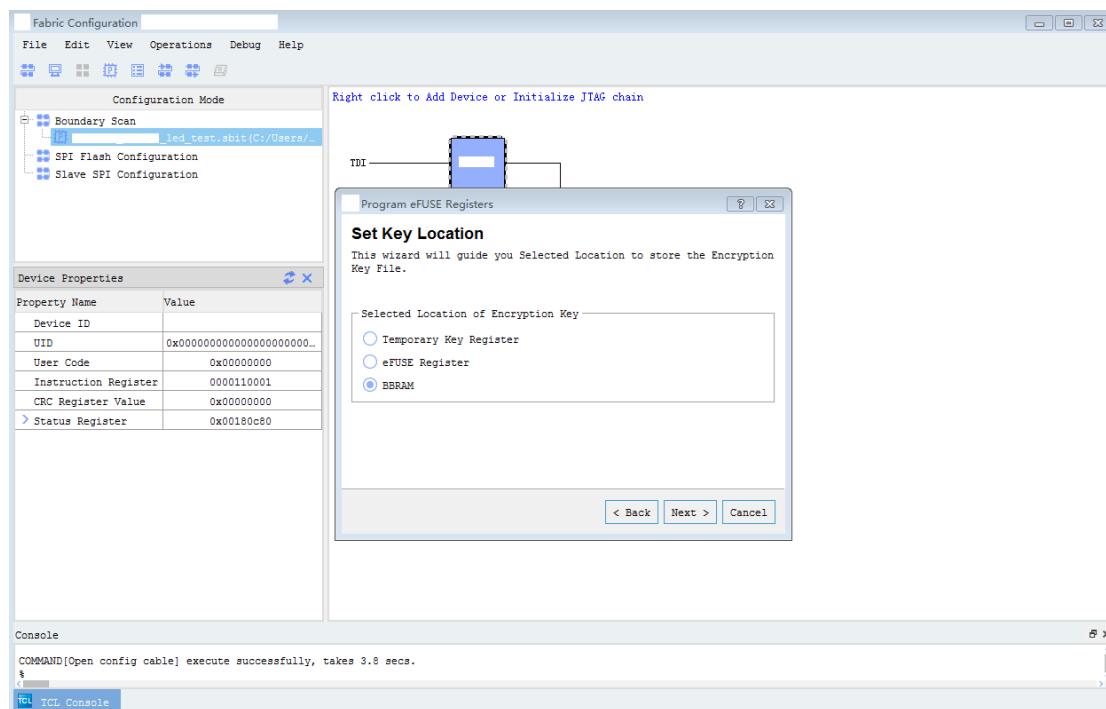


图 4-29 选择秘钥文件写入 BBRAM

2、编程 32 位 User eFuse

Efuse是一次性可编程非易失存储器, 32位Efuse (用于存储密钥key) 未编程的fuse值为0, 编程后的fuse值为1。Config Efuse Value 操作主要针对Titan系列和Logos系列, Config Efuse Value只能向目标器件进行一次性写操作, 应谨慎操作。

当器件准备好时, 就可以对器件进行配置Efuse存储器, 具体的操作如下: 在图 4-25中选择“Program User eFuse Bits”操作类型, 点击Next, 将会弹出如图 4-30的输入框, 用户可输入32位efuse二进制信息。输入完成后, 点击Next到summary页点击Finish进行编程。

由于向Efuse存储器件中写入32位efuse二进制信息是一次性操作, 写入之后, Efuse存储器中的信息不可更改, 需要谨慎操作。在点击Finish后将会弹出确认的弹框, 再次确认后方可向eFuse编程。

若写入Efuse配置成功, 在信息栏中显示相应的写入成功信息, 并在Device Properties窗口的eFuse->User Bits查看32bits efuse值。若不成功, 有很多种可能, 可能是硬件或连接问题或者配置文件与器件不匹配。

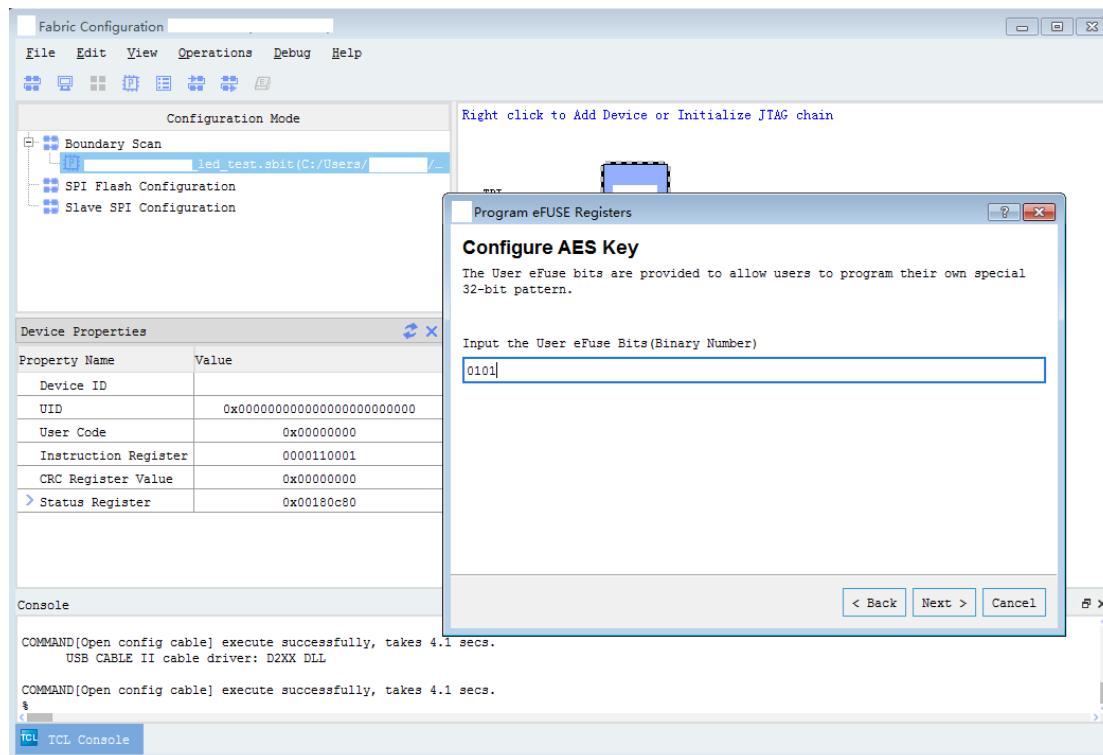


图 4-30 32bits User eFuse 配置界面

4.3.10 Jtag 软复位 FPGA

Fabric Configuration支持通过软件发送命令来实现复位操作。操作如下: 在所选中器件上

右键单击，会弹出如图 4-17 所示菜单，左键单击“Reset FPGA”，Fabric Configuration 会发送复位指令使其进行复位。

对于 Compact 系列的器件来说，某些操作在复位后才能生效，如修改特征控制位。如果不希望通过掉电进行复位，那么就可以使用该软件复位操作来实现。

4.3.11 Jtag 扫描 Flash 器件

Logos 和 Compact 系列支持直接编程外部 Flash，可通过右击弹出菜单，点击 Scan Outer Flash 可扫描外部 Flash 器件，以 PGL50H 为例，具体如下图 4-31 所示。

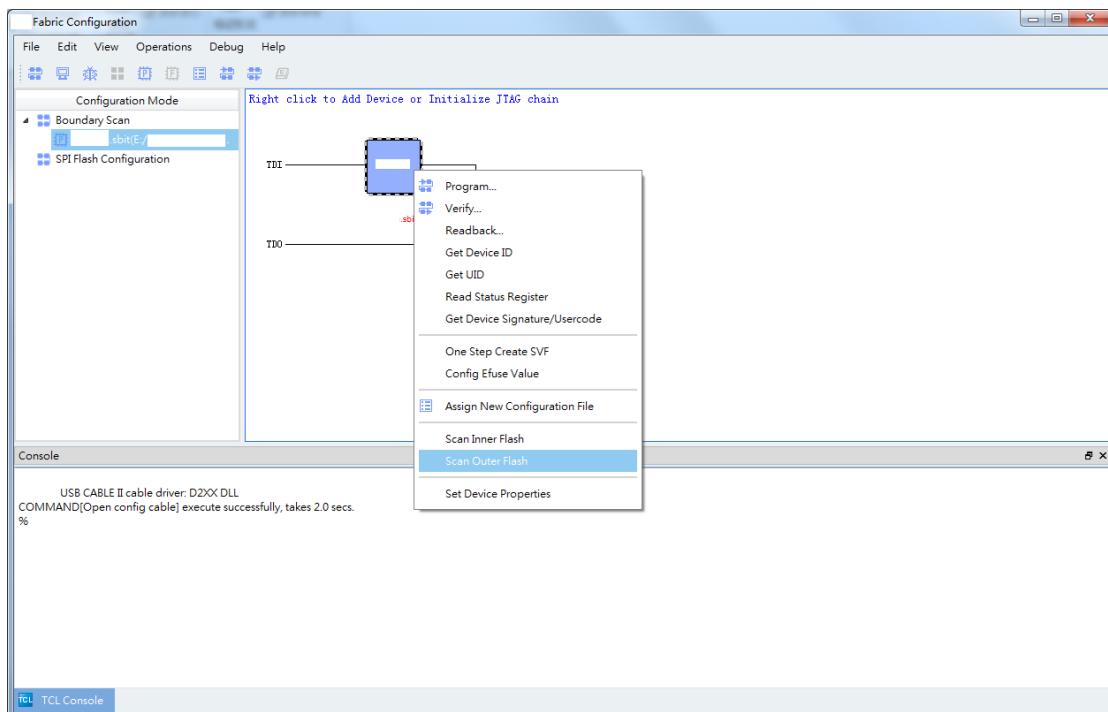


图 4-31 PGL50H 操作菜单

扫描外部 Flash 成功后可使用 JTAG 接口对 Flash 器件进行 Erase、Program、Verify 操作，具体如下图 4-32 所示。Flash 的下载属性配置同 SPI Flash。

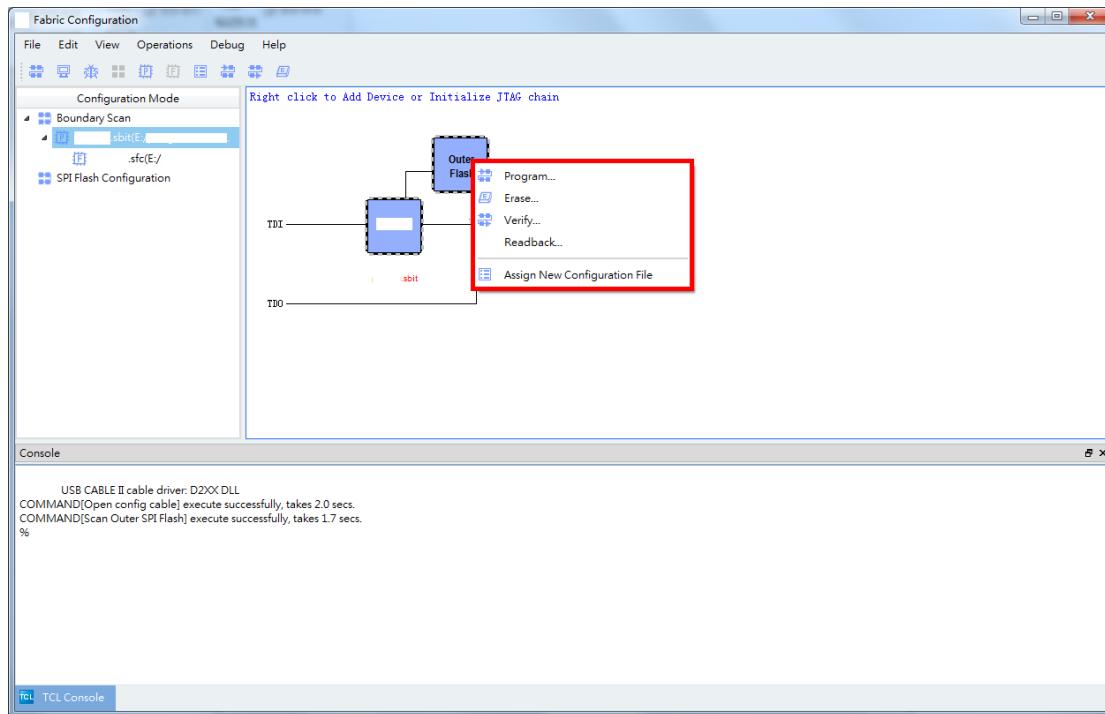


图 4-32 JTAG 操作 Flash 器件

4.3.12 SPI 扫描 Flash 器件

在主界面的左侧 Configuration Mode 窗口，选择 SPI Flash Configuration 模式，然后在工作区单击鼠标右键，左键单击 Scan Device，或者在 File 菜单或工具栏中左键单击 Scan Device，都将会执行扫描 SPI Device 的操作。

如果扫描 SPI Device 成功，会将扫描到的所有器件列于工作区内，在信息栏里显示器件型号，并弹出对话框使用户能够为器件添加配置文件如图 4-33 所示，扫描 SPI Device 成功示意图如图 4-34 所示。

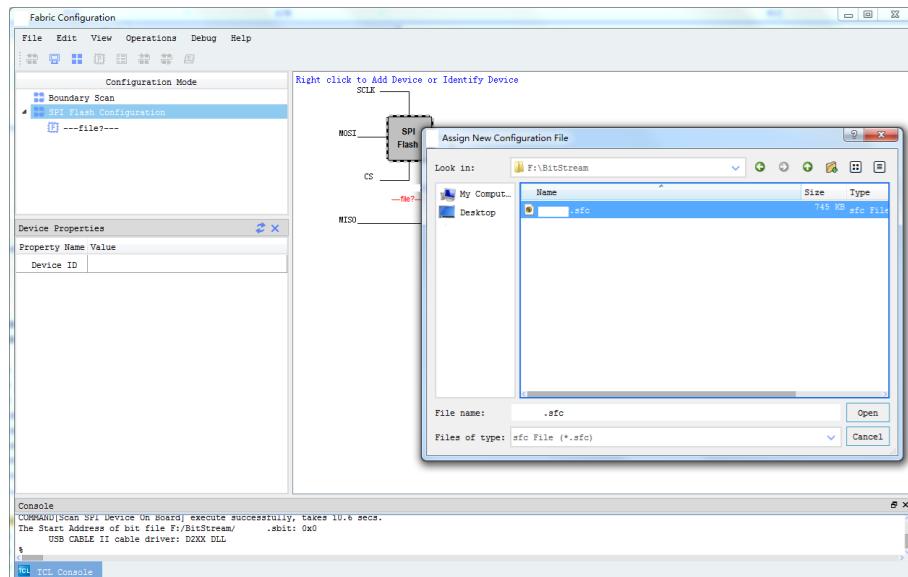


图 4-33 添加.sfc 文件

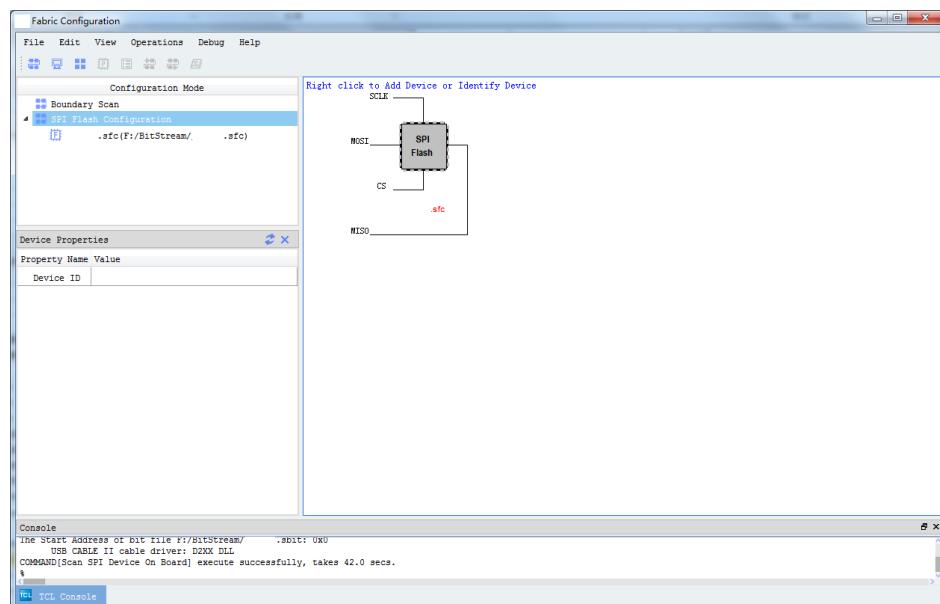


图 4-34 扫描 SPI Device 成功

4.3.13 SPI 擦除 Flash 器件

如图 4-35所示，Erase功能可以擦除SPI器件的内容。具体操作如下：右键单击，选择“Erase”，或者在工具栏中左键点击“Erase”，Fabric Configuration 会对SPI Flash器件执行擦除操作。用户可以通过右键菜单->Set Device Properties来选择擦除方式和擦除的起始块。目前Configuration支持块擦除、整片擦除、扇擦除、页擦除四种方式，Auto方式为根据所选位流的

大小及SPI Flash的容量大小自动选择擦除方式。若用户选择了整片擦除并且还设置了擦除的起始块，那么这个设置的起始块地址将被忽视掉。

Flash的擦除属性具体描述请查看19小节设置器件属性SPI Flash的擦除属性部分。

注：SPI Flash及外部Flash在擦除后的存储单元均被置1，Compact系列的嵌入式Flash擦除后的存储单元全被置0。

4.3.14 SPI 编程 Flash 器件

如图 4-35所示，Program功能将位流配置信息加载到SPI器件中。具体操作如下：右键单击，选择“Program”，或者在工具栏中左键点击“Program”，Fabric Configuration 会对SPI Flash 器件执行下载操作。用户可以通过右键菜单->Set Device Properties来选择下载方式和下载的起始块。目前Configuration依据每下载完一页Flash所等待的时间来分成Typic Program(等待时间居中)、Fast Program(等待时间最短)和Slow Program(等待时间最长)三种模式。Slow Program一定能保证Flash下载成功，其他在Cable设置频率不是很高的情况下，也能成功。

总的来说，如果SPI Flash编程失败，则应该考虑降低Cable的频率和选择等待时间较长的下载模式。

4.3.15 SPI 校验 Flash 器件

如图 4-35所示，SPI校验的过程首先是回读下载入SPI Flash器件中的数据，将读出的数据与载入的数据相比较，用来验证载入数据的正确性。

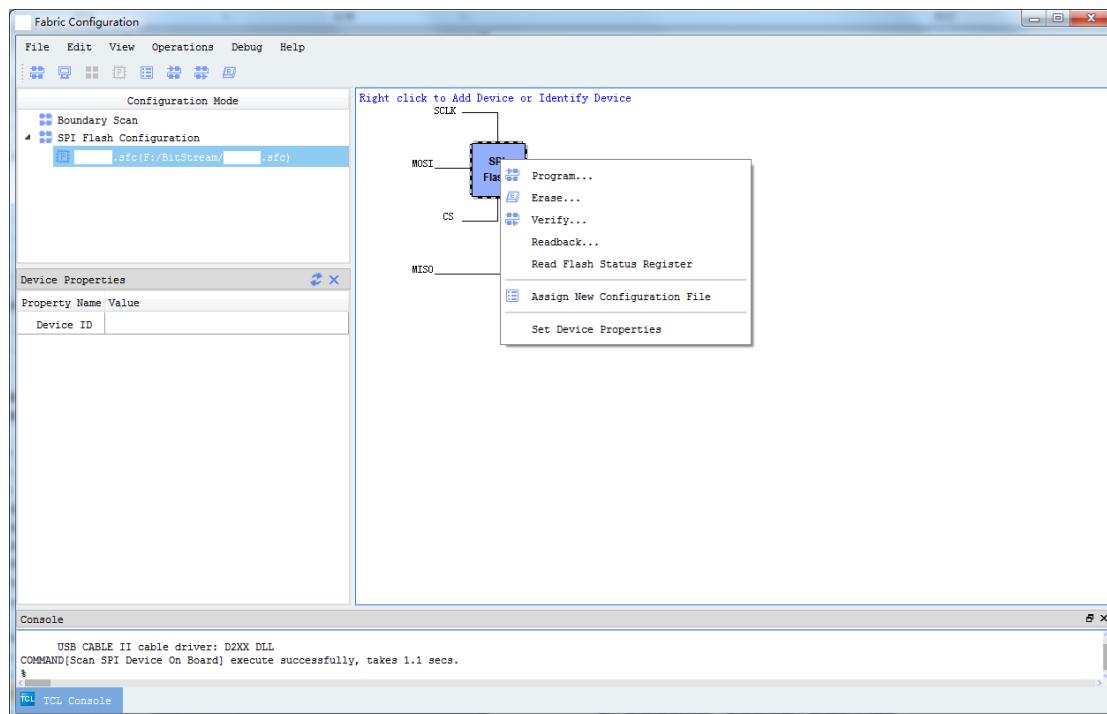


图 4-35 SPI Flash 支持的操作

4.3.16 SPI 回读 Flash 器件

如图 4-36所示，软件支持回读SPI Flash中的数据并将其存储入用户指定的文件。在SPI Flash器件上右键单击，然后左键单击“Read Back”，Fabric Configuration会弹出对话框让用户选择或新建回读数据存储的文件。

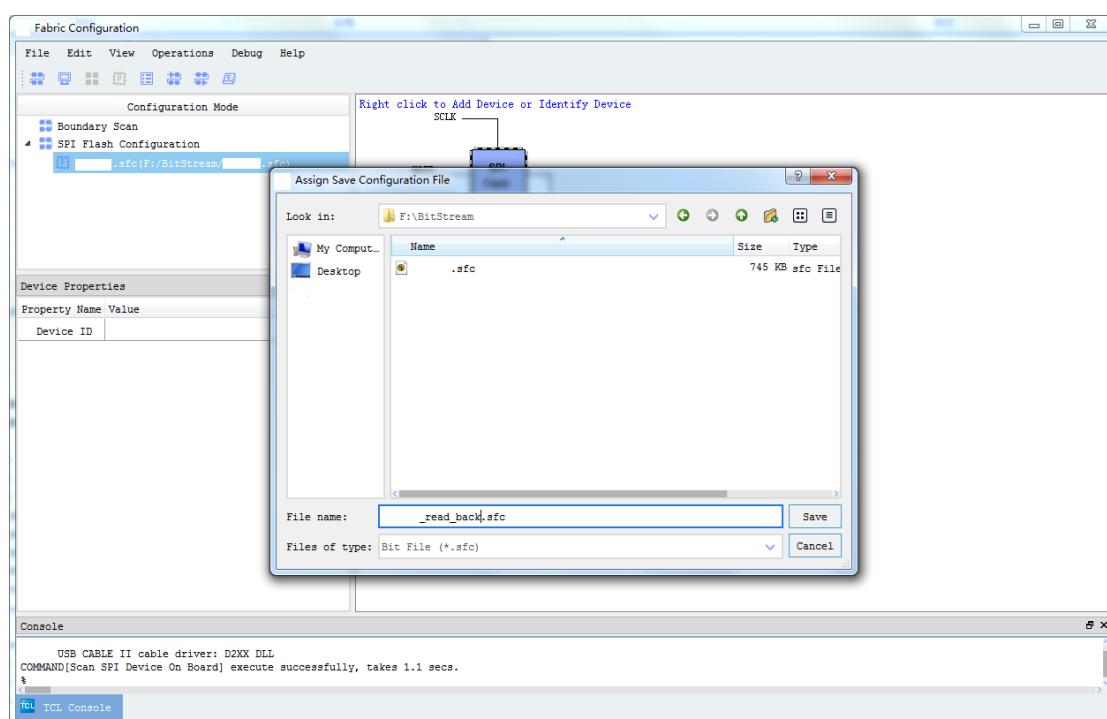


图 4-36 回读 SPI Flash 数据存储文件

4.3.17 SPI 读写 Flash 寄存器

如图 4-35 所示，用户可通过读 SPI Flash 寄存器来了解当前 Flash 处于哪种模式下，由于每个厂商的寄存器表示值不一致，因此需要参考相应的手册来了解寄存器每一位所表达的含义。

具体操作如下：右键单击，选择“Read Flash Status Register”，或者在工具栏中左键点击“Read Flash Status Register”，Fabric Configuration 会对 SPI Flash 器件执行读寄存器操作。若读取成功，则将寄存器值打印于信息显示栏，如图 4-37 所示。

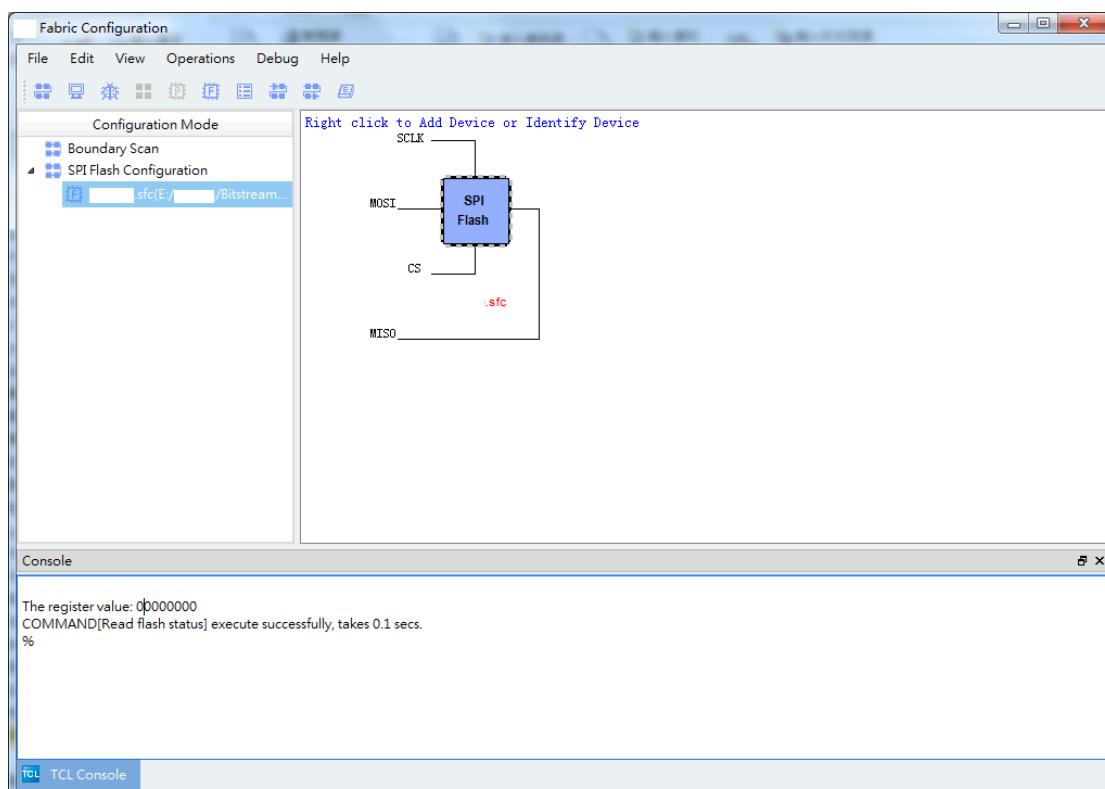


图 4-37 SPI Flash 状态寄存器

写寄存器值只能通过 Tcl 命令 `cfg_flash_write_register` 来实现，具体请参考第四节 Tcl 命令介绍章节 SPI Flash 部分。

4.3.18 设置操作器件属性

点击右键菜单下的 Set Device Properties 选项，可以设置 FPGA 和 Flash 相关器件的 Program 属性、Read Back 属性及 Flash 特有的 Erase 属性，FPGA 器件的器件属性设置支持 Titan2 系列、Logos/Logos2 系列和 Compact 系列，Flash 的仅支持外部 Flash(Jtag Flash)和 SPI Flash。

1、设置 Titan2 系列的操作器件属性

如表 1，Titan2 系列所具体支持编程选项如下所示：

属性	含义
Enable WakeDownReconfigure	表示是否允许关断重配。
Enable ISC Configure	表示是否使用ISC进行编程。
Enable Wake Down	表示是否允许关断回读。
Enable ISC Read Back	表示是否使用ISC进行回读。

表 1 Titan2 系列的编程属性

2、设置 Logos/Logos2 系列的操作器件属性

Logos/Logos2 系列支持的配置选项如表 2 所示：

属性	含义
Enable WakeDownReconfigure	表示是否允许关断重配。
Enable ISC Configure	表示是否使用ISC进行编程。
Enable Wake Down	表示是否允许关断回读。
Enable ISC Read Back	表示是否使用ISC进行回读。

表 2 Logos 系列的编程属性

3、设置 Compact 系列的操作器件属性

Compact 系列支持的配置选项如表 3 所示：

属性	含义
Enable WakeDownReconfigure	表示是否允许关断重配。
Enable ISC Configure	表示是否使用ISC进行编程。
Enable HighZDis Configure	表示是否使用离线编程
Enable Wake Down Read Back	表示是否允许关断回读。
Enable ISC Read Back	表示是否使用ISC进行回读。
Enable HighZDis Read Back	表示是否进行离线回读
Enable Erase before Program EFlash	表示在下载位流之前是否使能擦除EFlash
Enable Verify after Program EFlash	表示在下载位流之后是否使能验证EFlash
Enable Erase Entire Embed Flash	表示擦除Eflash时，是否整片擦除
Enable Reset FPGA after Erase Embed EFlash	表示擦除EFlash之后是否使能复位
Enable Reset FPGA after Program Embed EFlash	表示编程EFlash之后是否使能复位
Enable Reset FPGA after Program Feature Control	表示编程特征控制位之后是否使能复位
Enable Program Feature Control after Program EFlash	表示在下载位流之后是否使能编程特征控制位
Enable Program Feature Control before Program CRAM	表示在编程CRAM之前是否使能编程特征控制位

Embed Flash Start Page Index	操作Embed Flash 的起始地址页，用于设置编程、擦除、回读的起始地址页。
Embed Flash Tail Page Index	操作Embed Flash 的尾地址页，用于锁定Embed flash 从开始到该指定页的内容。
Embed Flash Read Back Size	指定回读从操作Embed Flash 的起始地址页开始的内容大小，单位为Byte。

表 3Compact 系列的编程属性

其中，Enable ISC Configure 和 Enable HighZDis Configure 互斥，Enable ISC Read Back 和 Enable HighZDis Read Back 互斥，即不能同时选中它们。

注：针对 Compact 系列特征控制位的操作说明见本章第 8 小节操作特征控制位

注：针对 Compact 系列嵌入式 Flash 的编程操作见本章第 9 小节操作嵌入式 Flash。

4、设置 Flash 的操作器件属性

用户在操作外部 Flash(Jtag Flash)和 SPI Flash 时可在该页面进行相应属性的配置，具体信息如下所示。

1、Flash Device Properties

Flash Start Address：显示 Flash 编程下载或者擦除或者回读的起始地址。

Flash Start Block Index(Block Size: 64KB)：Flash 编程下载或者擦除或者回读的起始块索引，设置的单位为 64KB。（例：如果 Flash Start Block Index 设置的值为 1，那么表示 Flash 编程下载的起始地址为 0x10000）。

Flash Start Sector Index(Block Size: 4KB)：Flash 扇区擦除时所设的起始扇区索引，设置的单位为 4KB。（例：如果 Flash Start Block Index 设置的值为 1，Flash Start Sector Index 设置的值为 2，那么表示 Flash 擦除的起始地址为 0x12000）。

Flash Read Back Size(Unit: Byte)：设置回读 Flash 的大小。

2、Flash Program Properties

Flash Program Mode：SPI Flash 编程下载时下载速率的级别选择，分别为 Typic、Fast、Slow 三种模式，速率依次降低，如果用户在下载 Flash 时不稳定，可降低下载速率。

Flash Erase before Program 选项表示，在编程之前会自动进行擦除 flash 中的内容，

如擦除成功后，再进行编程下载。

Flash Verify after Program 选项表示，在编程成功后会自动进行验证在 flash 中的内容和 sfc 文件内容的一致性。

3、Flash Erase Properties

Flash Erase Mode 包含五种模式：Sector Erase、Block Erase、Chip Erase、Page Erase 、Auto。

Sector Erase：表示按扇区进行擦除。

Block Erase：表示按块进行擦除。

Chip Erase：表示对 Flash 器件进行全部擦除。

Page Erase：表示对 Flash 器件进行页擦除。

Auto：表示根据要擦除的空间，程序自动选择合适的 Block 或 Chip 进行擦除，保证擦除时间达到最少。

4.3.19 FPGA 位流文件转换多种格式数据流

在Operations菜单中点击“Convert File”选项后弹出对话框如图 4-38所示，其主要功能是将FPGA位流文件转换成各种格式的文件，如用于Flash配置、菊花链器件以及一些FPGA器件功能的实现。具体有①Generate Flash Programming File；②Generate Daisy Chain File；③Generate Multi Revision File；④Generate Chain Svf File四种形式的文件转换。

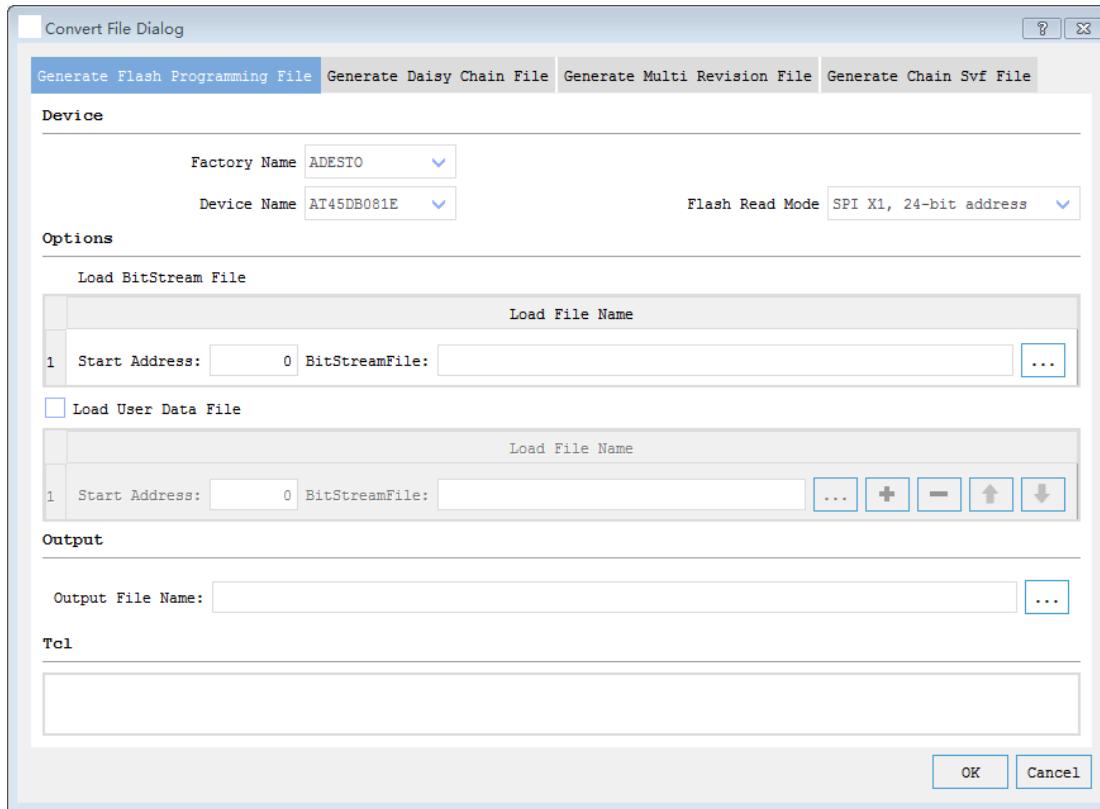


图 4-38 Convert File Dialog 对话框

1、Generate Flash Programming File

该功能主要用于转换生成可配置Flash的SFC文件。由图 4-39所示，在Generate Flash Programming File窗口，首先选择对应的Flash器件及其读写模式。然后选择要转换的位流文件，如果转换的位流文件为Logos系列，则还可选择数据采样的边沿和是否生成SPI X8模式，如图 4-38所示。

如果用户想配置位流在Flash的起始位置，可通过在Start Address输入地址来修改位流的起始地址，如图 4-39 Generate Flash Programming File所示，用户输入的地址为0x24， 则软件在转换时将会在位流开始处添加0x24个FF，换句话说，位流的起始地址相对于Flash芯片的开始处偏移了0x24个字节。如果用户不需要添加用户数据，此时可直接点击OK进行转换。

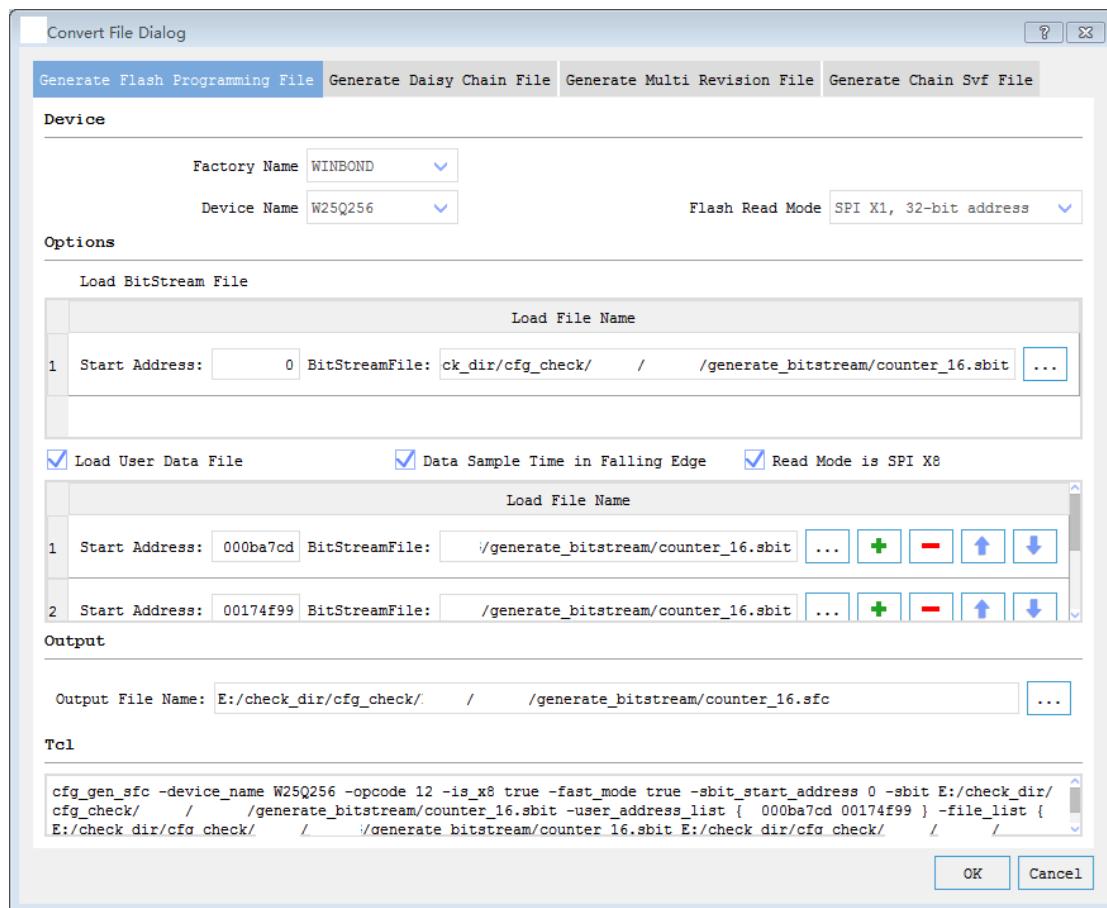


图 4-39 Generate Flash Programming File

如果用户需要在位流后面添加自己数据，此时可勾选Load User Data，然后在Load File Name表格中添加自己的用户数据，并可根据提供的按钮对用户数据进行位置的调整和删除，如图 4-39所示。一般来说，用户数据的起始地址默认跟随在位流后面，软件会自动计算出起始地址。如果用户想修改用户数据的起始地址，一定要注意起始地址值一定要大于位流文件的大小和位流文件起始地址所偏移的大小之和。如果输入的地址不正确，软件会自动更正该地址值为默认值。

注：添加用户数据条件为：必须要有位流数据。

位流文件以及调整好用户数据文件的位置后，点击OK按钮，便可生成SFC文件。如果添加的用户数据文件过多，可能会导致生成的时间较长，需要等待一段时间。

目前支持的flash器件及其所匹配的读写模式如附录四所示。

2、Generate Daisy Chain File

该功能主要是生成级联数据流下载文件。由图 4-40所示，在Generate Daisy Chain File窗口，添加产生级联数据流的sbit文件，文件前面的序号为实际FPGA链的位置。用户可以通过按钮来调整位置，一定要注意的是，生成级联数据流各个数据流所对应的器件一定要和下载

到实际FPGA链中的器件位置相对应。

用户可选择勾选生成bin文件，这样将同时生成sbit文件和bin文件。还可勾选两种翻转功能，仅会对生成的bin文件翻转，不会翻转生成的sbit文件。

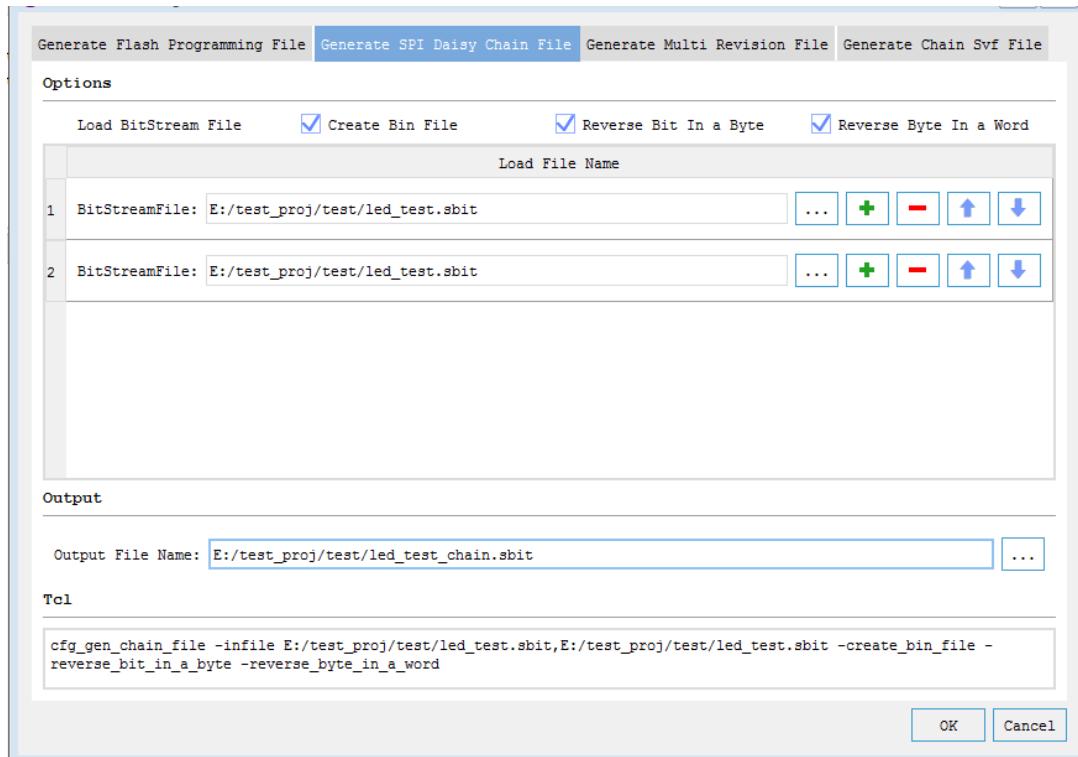


图 4-40 Generate Daisy Chain File

选择好位流文件以及调整好文件的位置后，点击OK按钮，便可生成级联数据流。如果添加的位流文件过多，生成的时间较长，需要等待一段时间。

3、Generate Multi Revision File

如图 4-41 所示，其主要功能是生成黄金数据流及组合数据流的 Flash 下载文件。用户可在 Device Family Type 选择相应的器件并在 Data Stream Type 选择框选择所要生成的位流格式，如表 4 所示。目前所支持的格式有：

- ①、SPI Upgrade Data Stream: SPI (Serial Peripheral Interface) 远程升级数据流，至少有两个应用数据流时才能正确生成。
- ②、BPI Upgrade Data Stream: BPI (Byte-wide Peripheral Interface) 数据流，至少有两个应用数据流时才能正确生成。
- ③、Multi Boot Data Stream: 多启动数据流，仅 Logos、Logos2、Titan2 系列支持。至少有两个应用数据流时才能正确生成。

④、Multi Function Data Stream: 多功能数据流，仅 Logos 系列支持。至少存在两个应用数据流时才能正确生成。

⑤、Master Auto And Dual Boot Data Stream。主自加载双启动数据流，仅 Compact 系列支持。当且仅当有两个应用数据流时，才能进行生成。

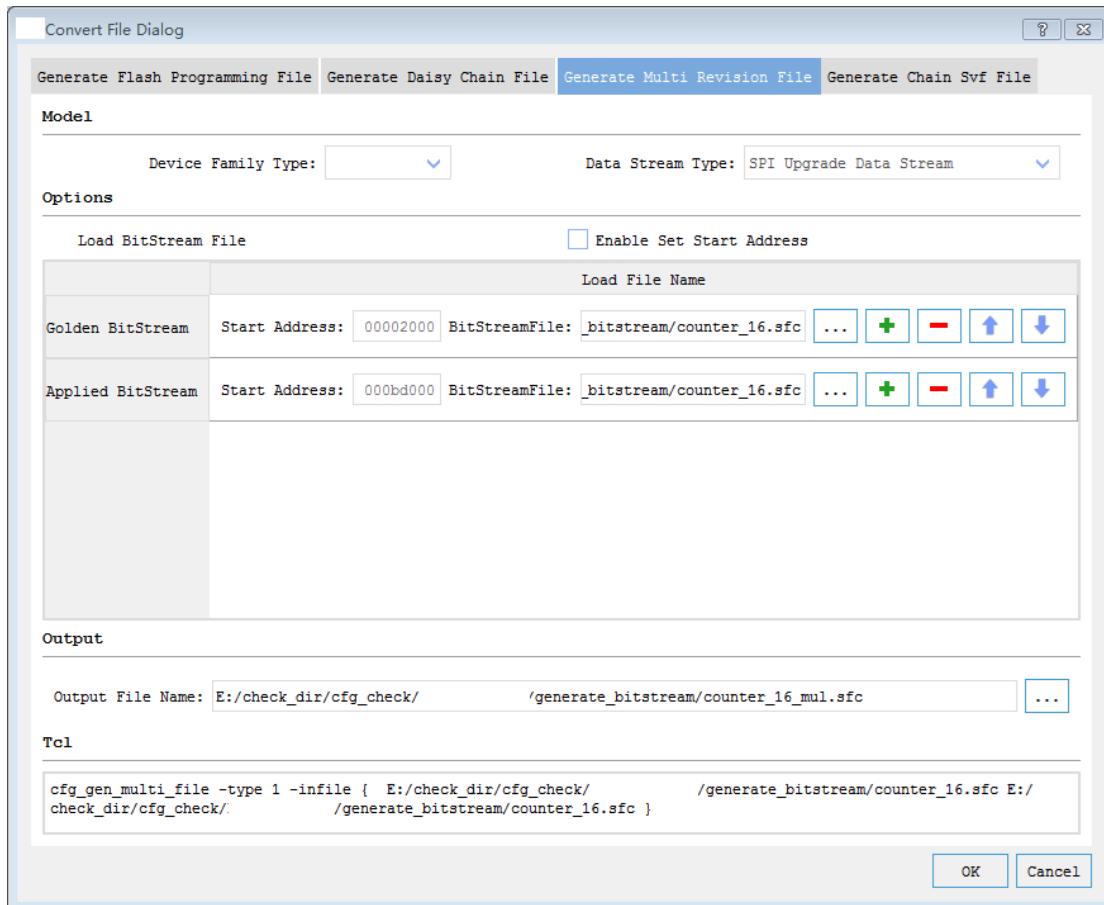


图 4-41 Generate Multi Revision Programming File

Device Family Type	Data Stream Type
Compact	Remote Upgrade Data Stream Master Auto and Dual Boot Data Stream
Logos	SPI Upgrade Data Stream BPI Upgrade Data Stream Multi Boot Data Stream Multi Function Data Stream
Logos2	SPI Upgrade Data Stream BPI Upgrade Data Stream Multi Boot Data Stream
Titan2	SPI Upgrade Data Stream BPI Upgrade Data Stream

	Multi Boot Data Stream
--	------------------------

表 4 器件系列与文件类型对应表

3.1、生成操作流程

图 4-41，通过点击按钮添加产生黄金组合数据流的SFC文件，文件前面的序号为实际 revision 的位置，通过移动按钮调整下载到Flash相应revision位置。Output File Name 为产生的数据流的名称，默认为最后一个添加位流文件名后加_mul.sfc，也可以使用选择文件按钮指定文件名。

选择好位流文件以及调整好文件的位置后，如果用户不需要对位流的地址进行设置，那么就点击OK按钮，生成Multi Revision数据流。

3.2、Logos、Logos2 和 Titan2 系列配置数据流的起始地址

在对 Logos、Logos2 和 Titan2 系列的器件生成数据流时，软件支持对各个添加数据流的起始位置进行修改。如果用户需要修改数据流的起始地址，那么就应该计算填上所有数据流的起始地址，不能仅仅修改某一个数据流，否则不能生成数据流，如图 4-42。

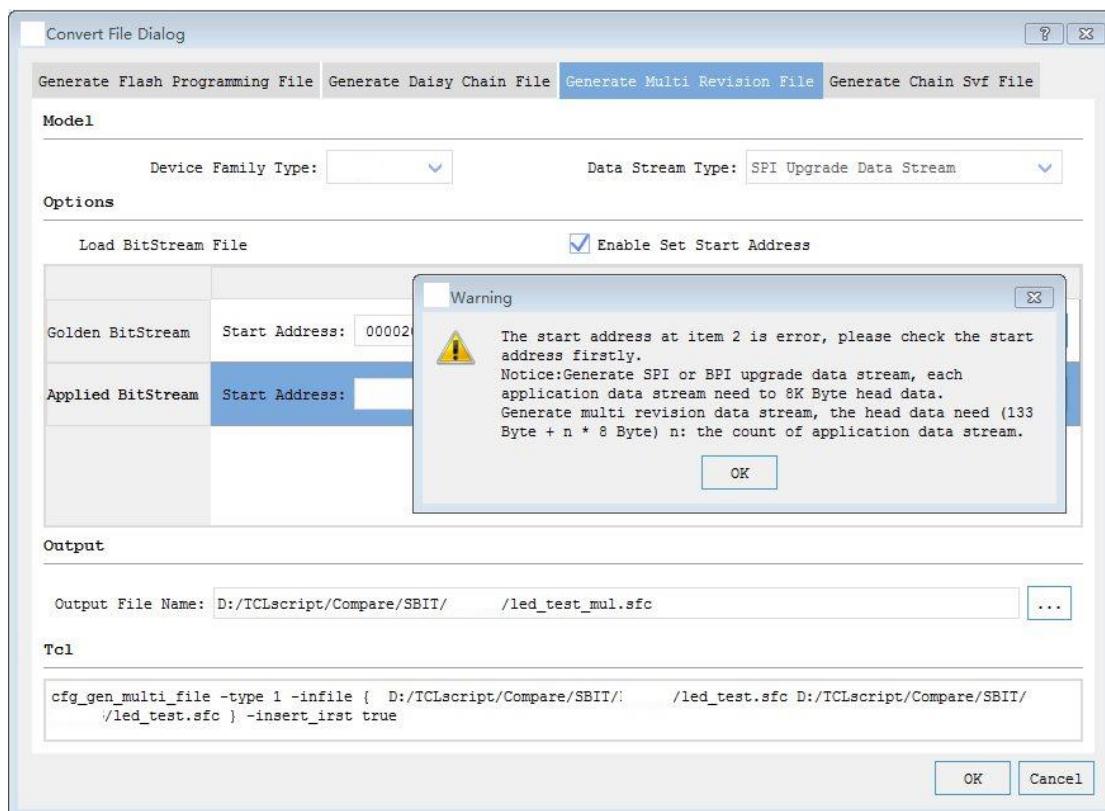


图 4-42 地址输入为空不能生成

在勾选了 Enable Set Start Address，便可在 Start Address 输入框输入数据流的起始地址值。由于一个位流必须从扇区的开始处开始存放，因此输入的起始地址值必须是4K的整数倍(扇

区的大小为4K Byte)。如果输入的值不是4K的整数倍，在输入完成后，软件会自动提示输入的值不符合规范，是否需要进行自动更正，如果点击否，则该输入值标红，在修改成符合规范的值后方可进行生成数据流，如图 4-43所示。

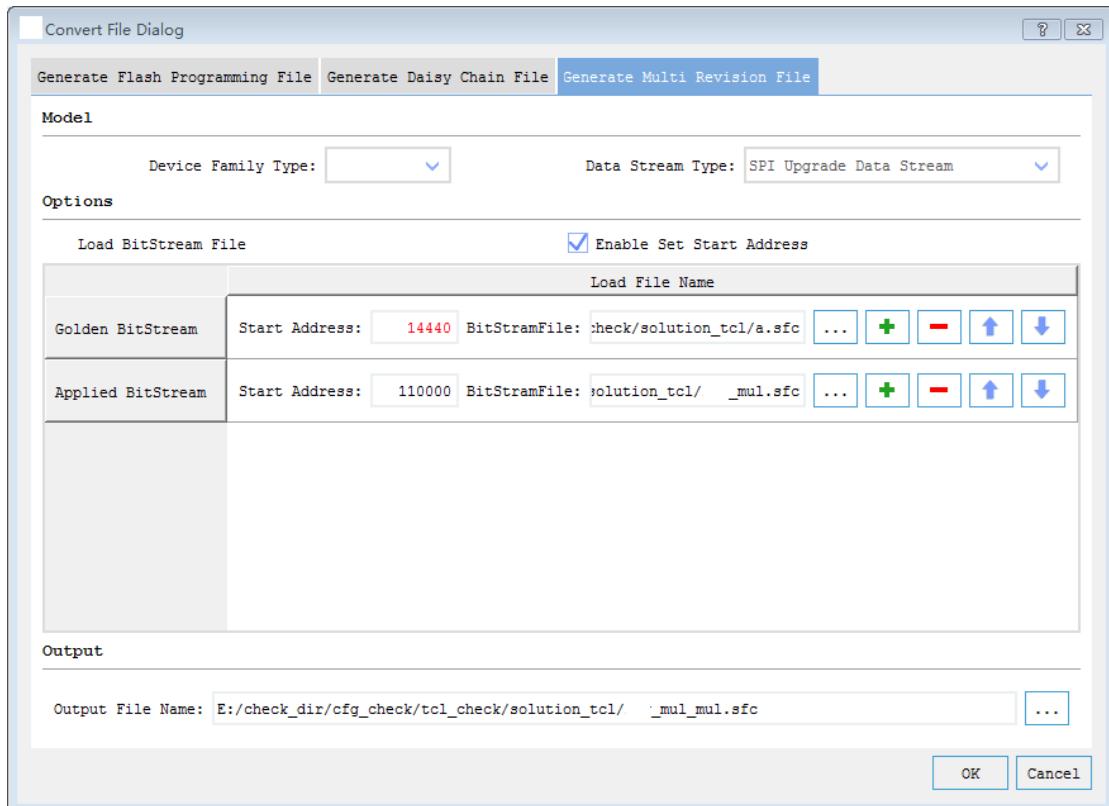


图 4-43 地址输入不规范不能生成

在计算数据流地址时，应当考虑生成数据流所应添加的头部数据。如图 4-44 所示，每一类组合数据流都是由头部数据和用户的位流数据组成的，在生成时，软件会自动在每个位流数据的添加 **108 个字节** 的同步数据组成新的位流数据，即每个位流都比原生位流多 108 个字节，在计算起始地址时不要忽略掉这部分地址。

注：这里所指的头部数据是指整个生成数据流的头部数据，应当与每个数据流的位流头部区分开来。

在生成SPI Update Data Stream数据流时和BPI Update Data Stream数据流时，每一个应用数据流就需要额外添加8KB的头部数据。比如说：现在有一个**黄金数据流大小为800KB**，两个**应用数据流大小都是800KB**。那么头部数据所占的大小为16KB，所以黄金数据流的起始地址至少为0x4000(16KB=0x4000)。如果黄金数据流的起始地址为0x4000，那么第一个应用数据流的起始地址至少为

$0x4000 + 0xC8000 + 0x6C = 0xCC06C$ (800K=0xC8000, 108 = 0x6C)。因一个数据流必须从扇区的开始处开始存放，所以应向上取整为0xCD000,即第一个应用数据流的起始地址

至少为0xCD000。

同理，第一个应用数据流的起始地址为0xCD000，那么第二个应用数据流的起始地址至少为

$0xCD000 + 0xC8000 + 0x6C = 0x19506C$ ($800K=0xC8000$, $108 = 0x6C$)。因一个数据流必须从扇区的开始处开始存放，所以应向上取整为0x196000,即第二个应用数据流的起始地址至少为0x196000。

同理，在生成Multi Function Data Stream数据流时，头部数据需要额外添加133Byte + $n \times 8\text{Byte}$ ， n 为添加数据流的数目，其他起始地址计算过程同SPI数据流起始地址计算。

3.3、Logos、Logos2 和 Titan2 系列数据流格式

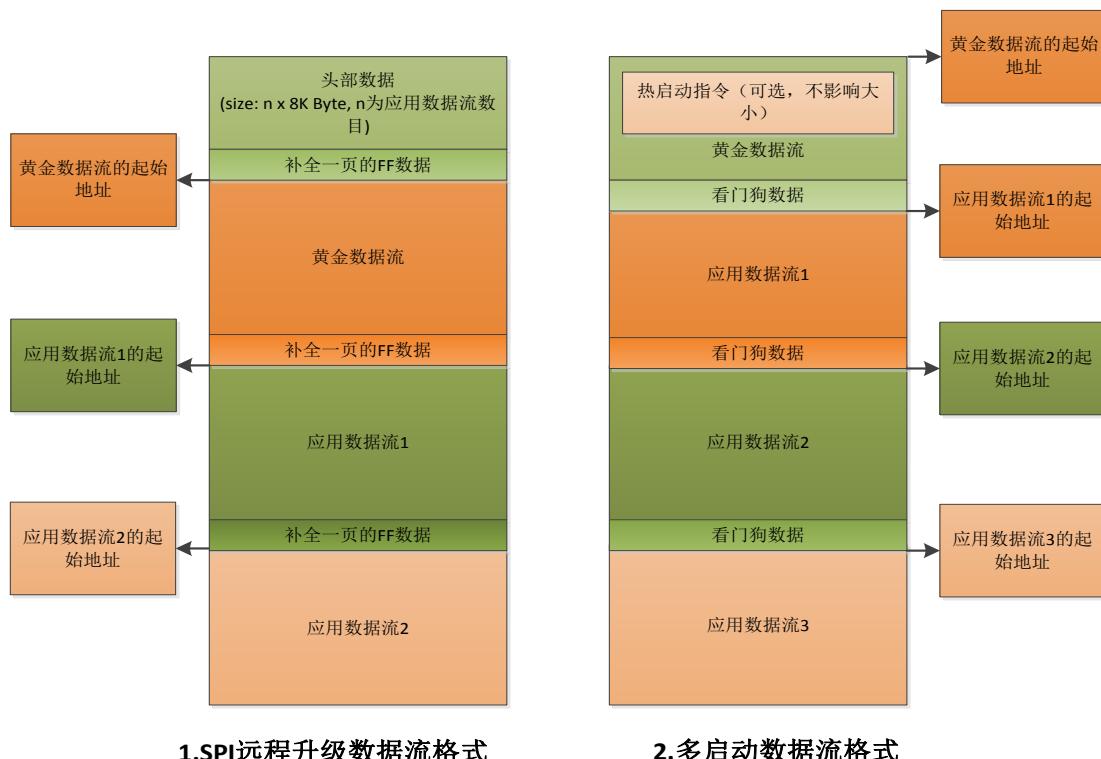


图 4-44 Logos、Logos2 和 Titan2 系列组合数据流的位流格式 1

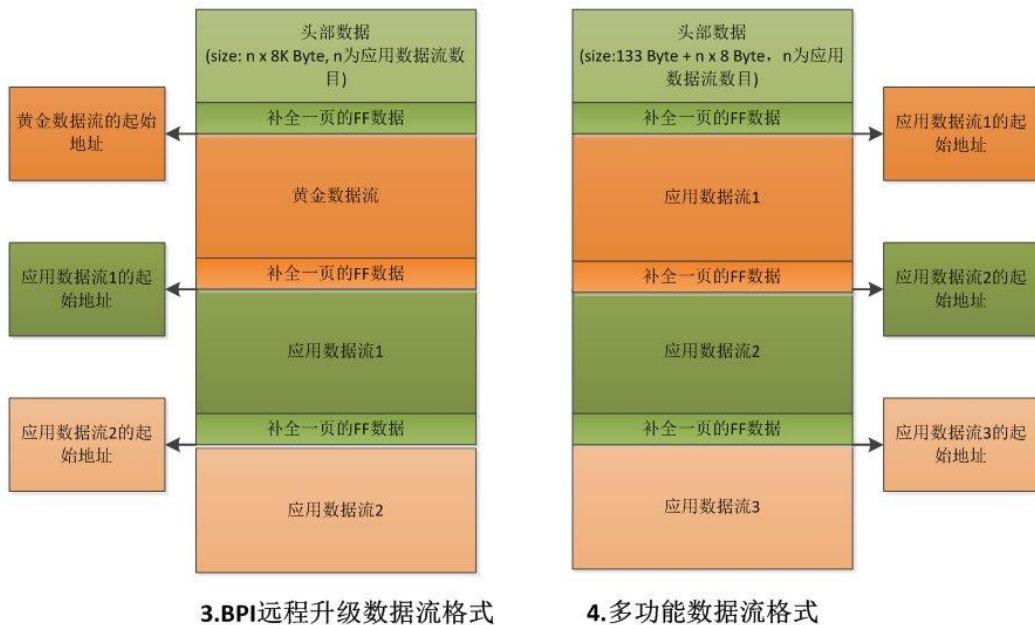


图 4-45 Logos、Logos2 和 Titan2 系列组合数据流的位流格式 2

3.4、Compact 系列配置数据流的起始地址

Compact 系列的器件在生成 Remote Upgrade Data Stream、Dual Boot Data Stream 数据流时，软件支持对各个数据流的起始位置进行修改，其修改规则和约束与 Logos 系列的器件大体相同。但存在以下差异：①、Compact 系列的 Remote Upgrade Data Stream 只需添加 512Byte 的头部数据，且每个数据流之前不需要去同步。②、Compact 系列的黄金数据流是在普通位流的基础上插入 64Byte 的数据形成的。③、Compact 系列的 Dual Boot Data Stream 没有头部数据。其数据流格式如下

图 4-46 所示

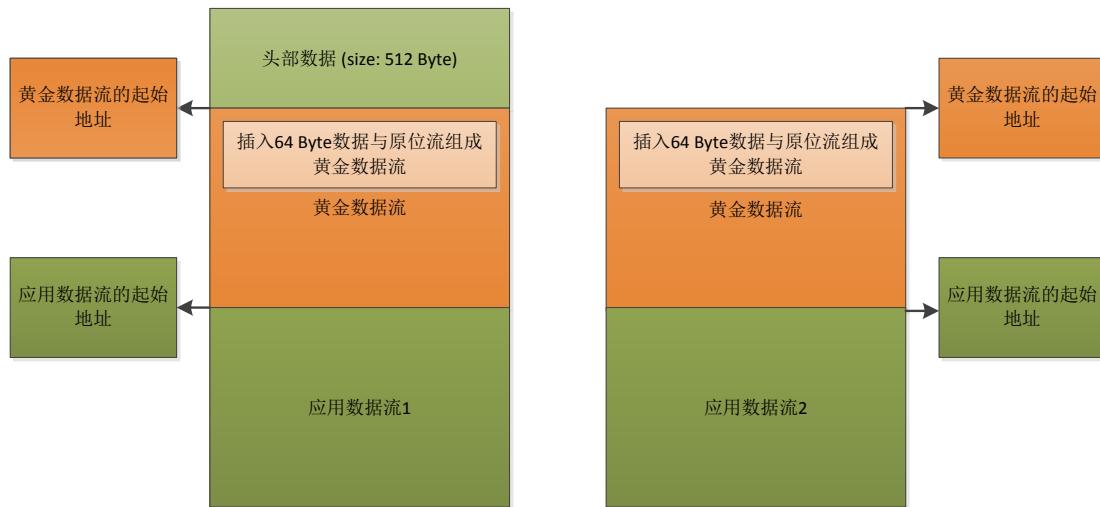


图 4-46 Compact 系列组合数据流的位流格式

4、Generate Multi/Dual Boot File (向导方式)

该功能为向导方式生成方案级位流。添加器件并连接 server 后右击器件，在弹出菜单中，选择 Generate Multi/Dual Boot File 选项。会弹出如图 4-47 所示窗口（该窗口显示内容因器件而异）。

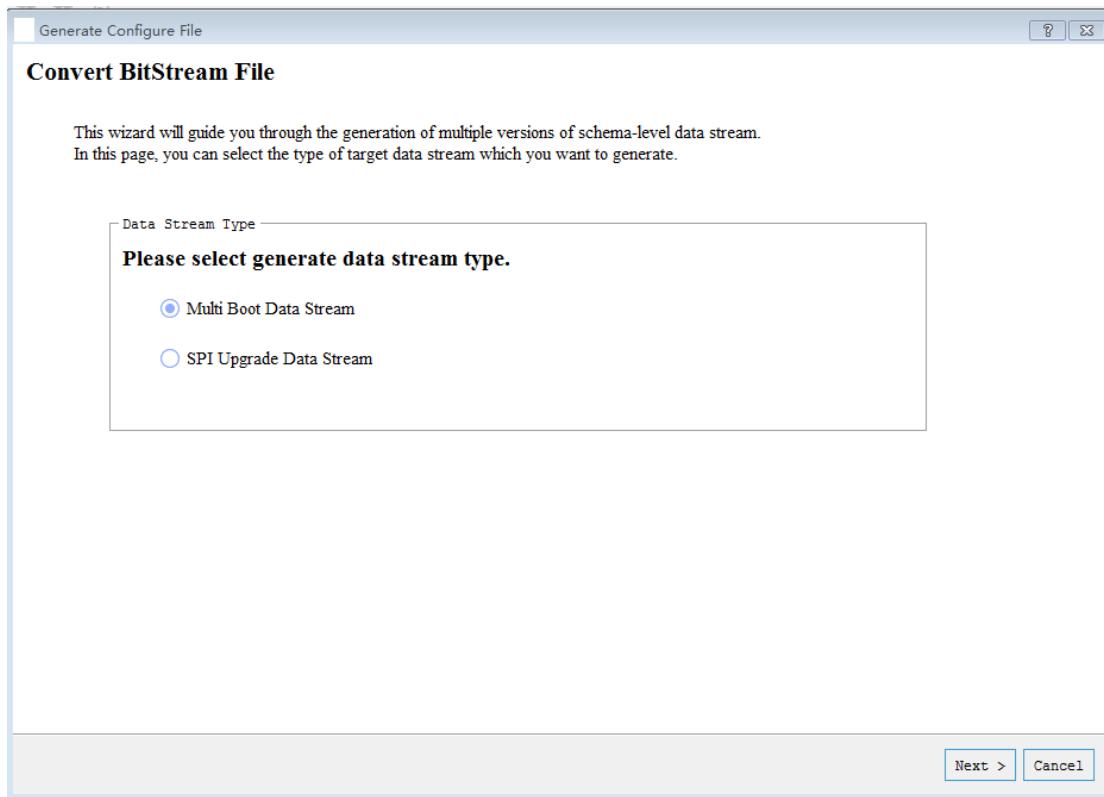


图 4-47 PGL 系列方案级位流产生窗口

4.1 Master Auto and Dual Boot Data Stream

如图 4-47 所示，该选项为生成双启动位流选项。

单击 Next 进入双启动位流设置界面图 4-48 在 Golden BitStream 中设置黄金数据流。在 Applied BitStream 中设置应用数据流。设置好后可在如图 4-51 处勾选需要设置的部分，包含设置特征控制位，生成 SVF 文件（相关参数请参考对应章节）和分离生成方案级位流的功能选择。

分离生成主自加载双启动位流功能会将主自加载双启动位流拆分成_golden.sfc 和_app.sfc 两个内容，若有勾选生成 svf，则会同时生成_golden.svf 和_app.svf（带地址设置）两个文件。

注：若结合使用分离生成，生成 svf，以及设置特征控制位功能，在分离的两个 svf 中都会进行特征控制位的配置，且要求特征控制位的值保持一致。若黄金数据位流和应用数据位流中两者的特征控制位值不一致，在生成双启动位流时将会产生告警。

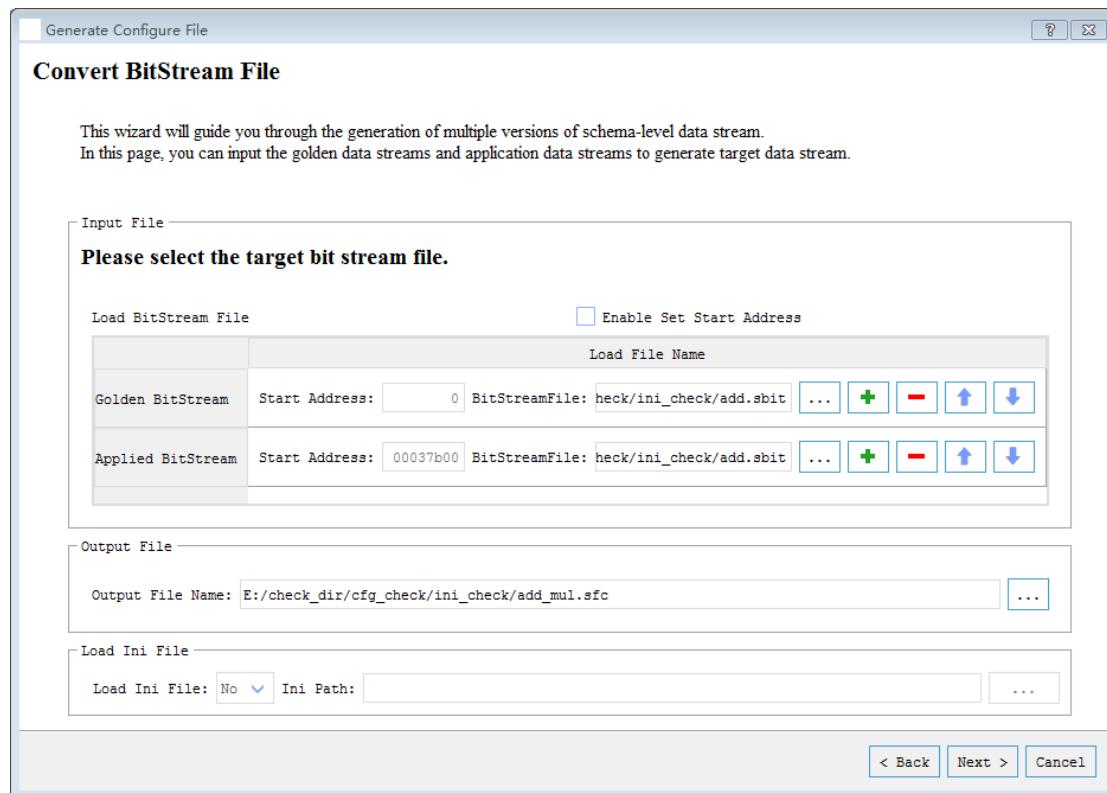


图 4-48 双启动位流设置界面

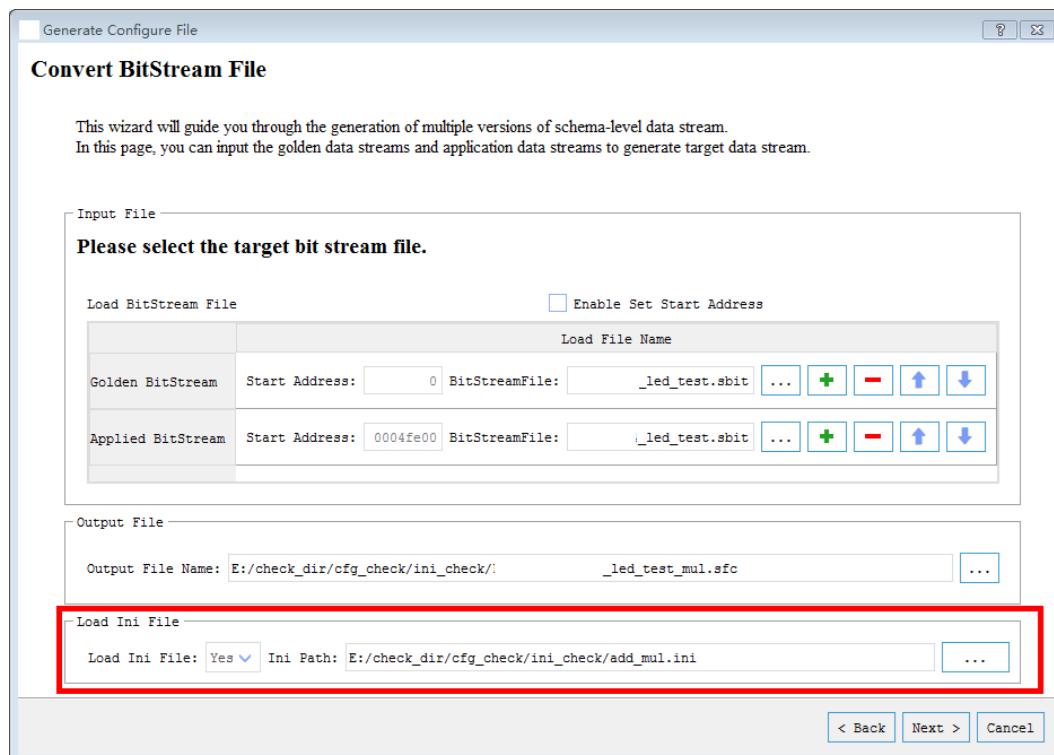


图 4-49 配置文件导入界面

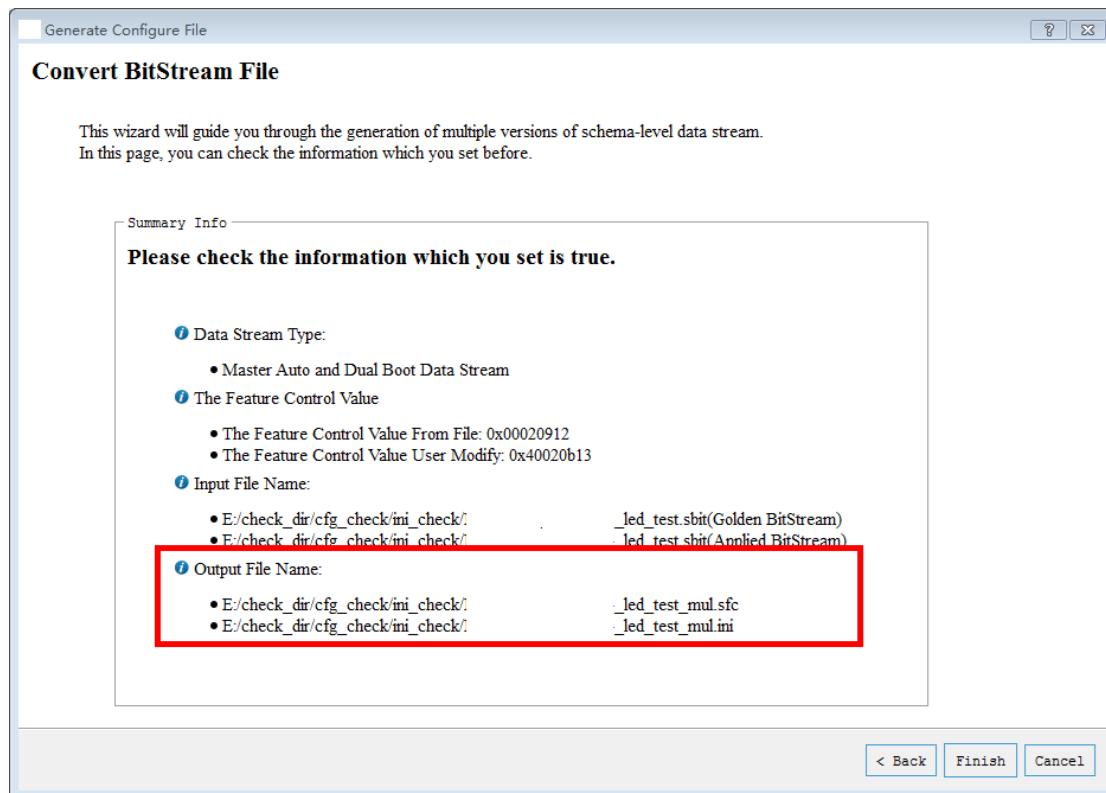


图 4-50 配置文件生成界面

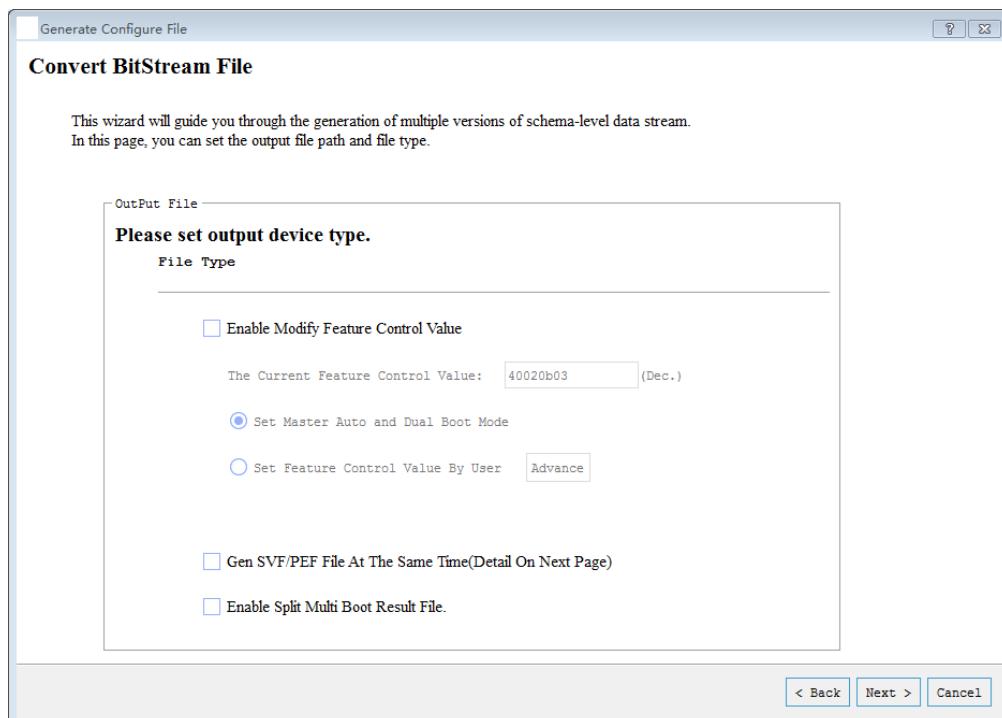


图 4-51 双启动位流其余设置

方案级位流支持使用配置文件 (*.ini) 加载设置，用户可通过导入配置文件以进行后续参数设置，如图 4-49 所示。若导入配置文件 (*.ini) 则无需再设置其余参数，向导会直接跳到 summary 页面。

若用户没有配置文件(*.ini) ,则在选择不导入文件(*.ini)的情况下软件会自动生成配置文件(*.ini) ,界面设置如图 4-48 所示。自动生成的文件 (*.ini) 路径会在 summary 页面的 Output File 栏显示 ,一般会和生成的 Output File 文件 (*.sfc) 在同个目录下 , 如图 4-50 所示。

summary 页面点击 next 后会跳转到最后一页 , 也就是 tcl 提示界面 , 如图图 4-52 所示 , 该提示内容可复制到 tcl 输入框内进行指令生成方案级位流。

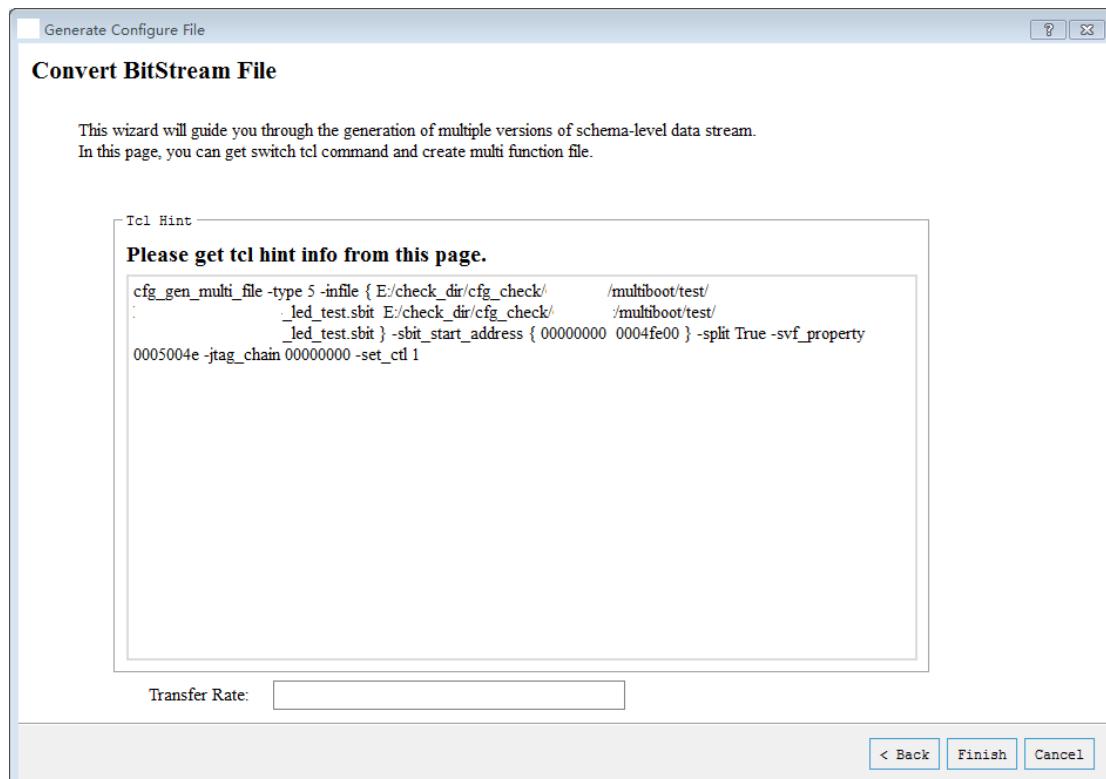


图 4-52 tcl 提示界面

4.2 Multi Boot Data Stream

多启动位流适用于 PGL 、 PG2L 系列器件。和双启动类似 , 也是先设置黄金数据量和应用位流 , 同时可以插入 IRST 指令到数据流中 (如图 4-53) 。然后设置输出文件路径和输出 SPI 模式即可 (如图 4-54) 。

支持用户手动设置应用数据流前的看门狗数据频率 , 勾选 Use user setting watch dog frequency 即可。

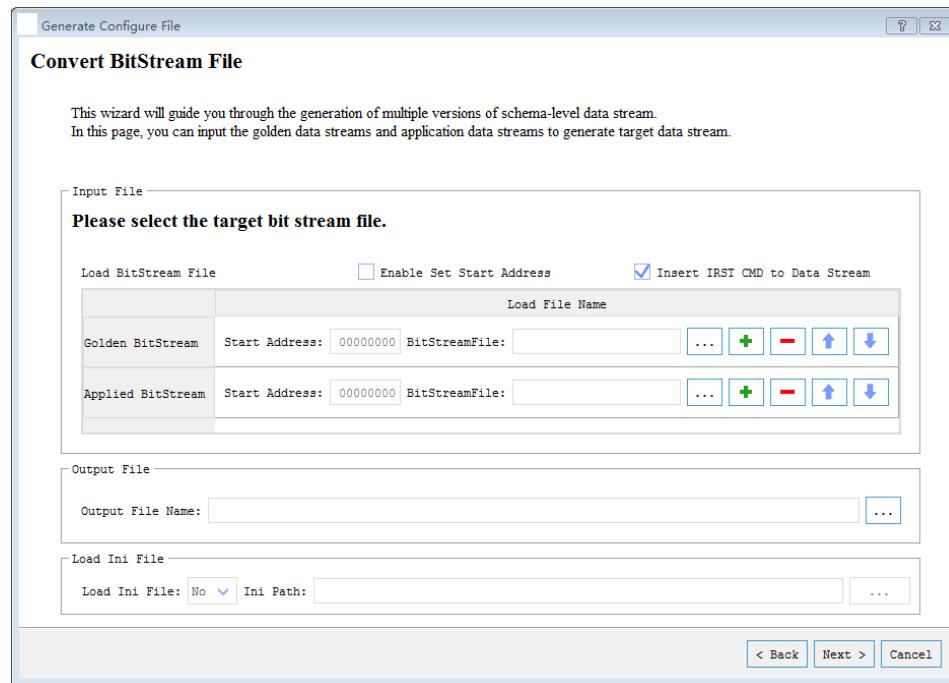


图 4-53 多启动位流输入设置界面

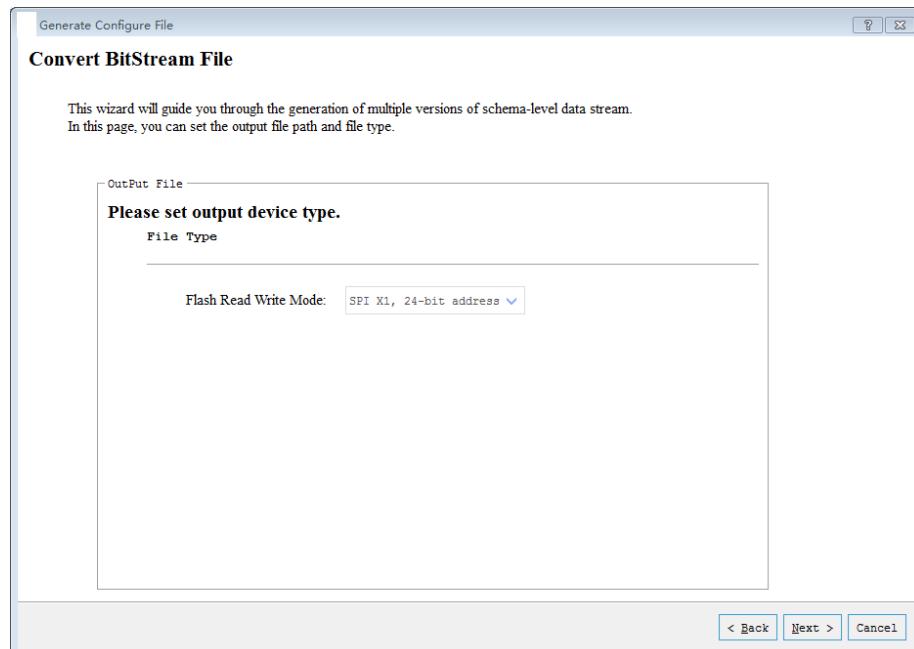


图 4-54 多启动位流输出设置界面

4.3 SPI Upgrade Data Stream

SPI 升级数据流和多启动位流设置界面一致，同样为设置输入的黄金数据量和应用位流后设置输出文件路径和输出 SPI 模式即可。

5 Svf 文件合并

该功能主要是级联svf/pef文件生成。如图 4-55所示，在Generate Chain Svf File窗口，添加产生级联数据流的svf文件，文件前面的序号为实际FPGA链的位置。用户可以通过按钮来调整位置，一定要注意的是，生成级联数据流各个数据流所对应的器件一定要和下载到实际FPGA链中的器件位置相对应。

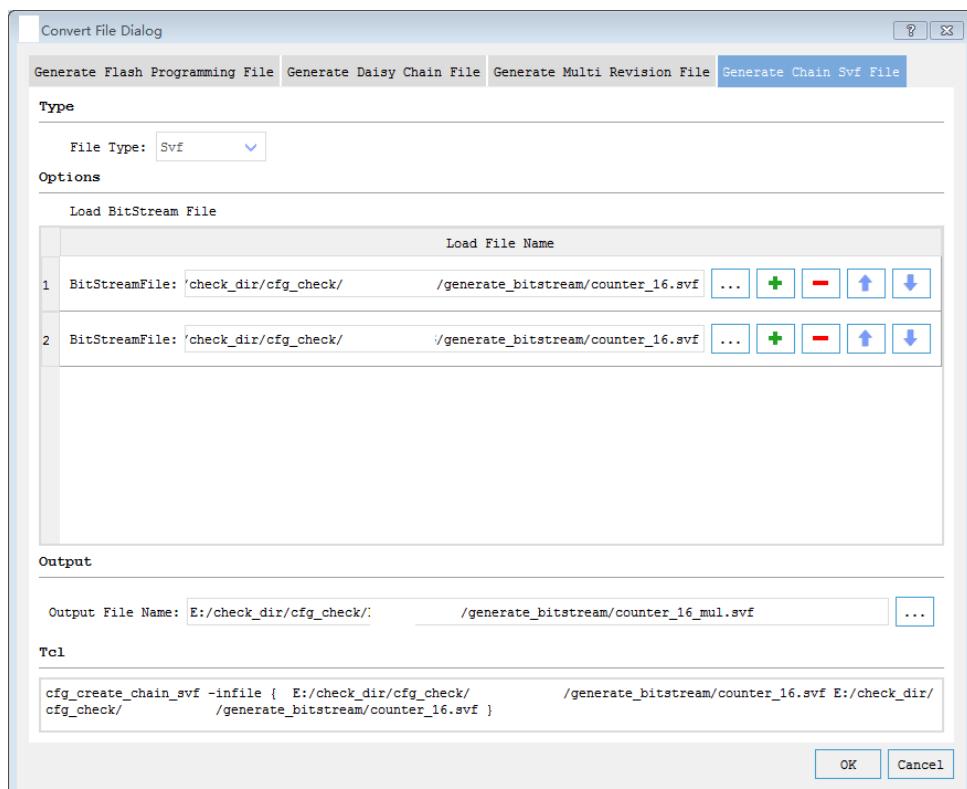


图 4-55 Svf 文件合并功能

选择好svf文件以及调整好文件的位置后，点击OK按钮，便可生成级联svf。如果添加的svf文件过多，生成的时间较长，需要等待一段时间。

4.3.20 Jtag 通用配置文件(SVF & PEF)

SVF 的全称是 Serial Vector Format，是 IEEE1149.1 的一部分，SVF 已经被芯片厂商作为 JTAG 数据交换标准。SVF 文件内容为 ASCII 文本格式，包含操作命令和数据，通过控制 JTAG 收发数据来实现对可编程逻辑器件的配置。适用于所有 Pango 器件，可用于对嵌入式 Flash 进行编程、擦除、校验或对 CRAM 进行配置。

PEF（Pango Embedded File）文件是 SVF 文件的压缩版本，是紫光同创自定义的二进制格式文件，需要使用紫光同创提供的驱动才能使用。适用于所有 Pango 器件，可用于对嵌入式 Flash 进行编程、擦除、校验或对 CRAM 进行配置。

1、生成 SVF 文件

Fabric Configuration 支持用户进行一键生成可执行 SVF 文件的操作，在该选择的器件上右键单击，在弹出的菜单中选择 One Step Create SVF 选项，点击执行后会出现如图 4-58 所示设置 SVF 文件参数与要生成 SVF 文件的路径，点击 OK 即可生成 SVF，默认会同步生成 PEF 文件。若不需要生成 PEF 文件则需在菜单栏 Edit->perference 中勾选 Enable Default Not Create Pef，如图 4-56 所示。

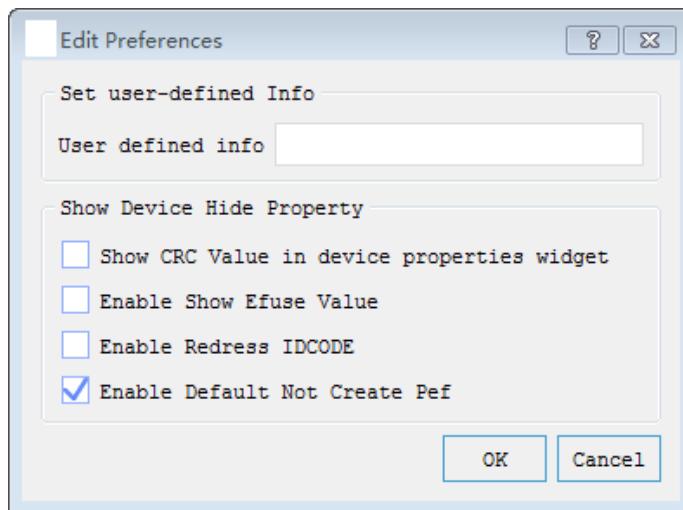


图 4-56 需要生成 pef

1.1、生成 Cram 类型的 SVF 文件

生成用于 cram 的 svf 文件，如图 4-57 所示需要设置相关参数与 SVF 输出文件路径，点击 ok 即可。用户初次使用后可拷贝 tcl 提示框内指令，以便后续快捷生成 svf。相关参数介绍如表 5 所示。

Enable Create Flatten SVF	是否产生 SDR/SIR 填充链信息的 SVF（注：使用此功能将不能使用 pef）
Enable Write Header and Comments	是否启用文件头和注释，勾选表示启用，默认勾选
Disable Stopover Test Logic Reset	位流配置流程禁止跳转到 TLR
Enable SEC option of RUNTEST Command	RUNTEST 指令用于指定 JTAG 状态机跳转到特定状态下时需要保持的等待时间（具体的等待时长和操作选项相关），等待时间有两种实现方式，TCK 时钟打拍和系统延时函数：该选项勾选时，通过系统延时函数实现等待时间；该选项不勾选时，通过 TCK 时钟打拍实现等待时间，默认勾选

Set SVF Config Frequency	表示设置 SVF 的配置频率， 默认为 1MHz。 当勾选“Enable SEC option of RUNTEST Command”时，RUNTEST 指令的等待时间与频率无关；当不勾选“Enable SEC option of RUNTEST Command”时，RUNTEST 指令对应的 TCK 打拍时钟数与频率成正比，此场景下，用户板上的 TCK 频率需要和软件设置的频率一致，或者低于设置频率；如果板上 TCK 频率大于软件设置频率，可能导致配置失败。
--------------------------	---

表 5 CRAM SVF 参数

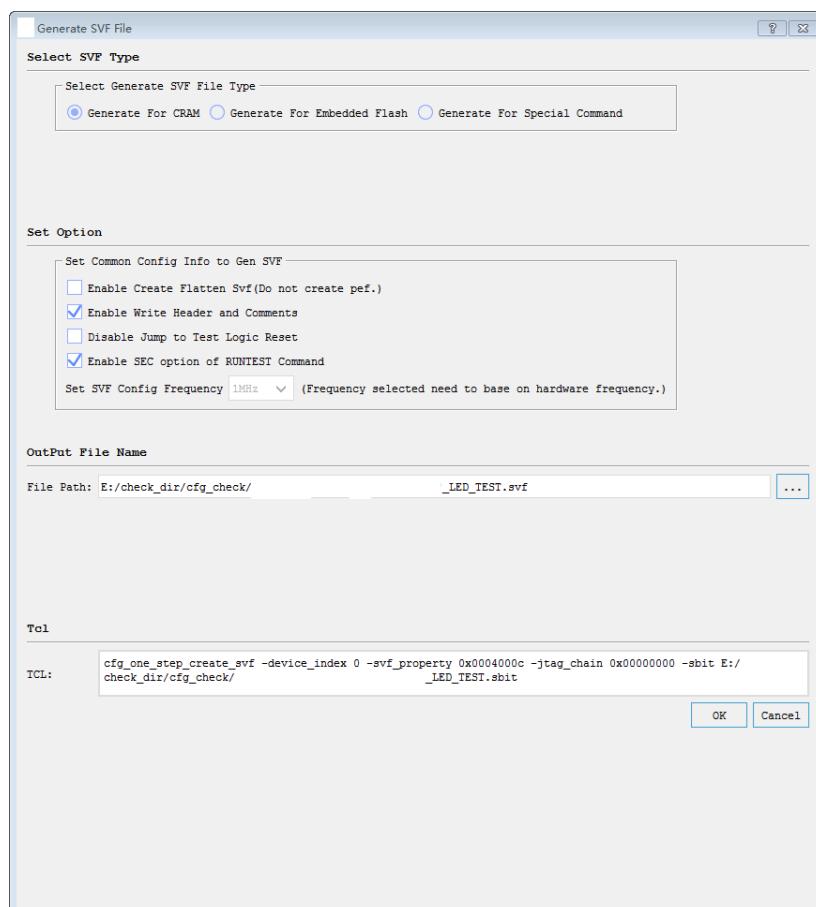


图 4-57 生成 CRAM 类型 svf

1.2、生成下载到嵌入式 Flash 的 SVF

对于 Compact 系列，用户既可生成配置 FPGA 的 SVF 文件，也可生成配置嵌入式 Flash 的 SVF 文件，用户可在设置器件属性中 Embed Flash Properties 进行设置。如图 4-58 所示，Set Common Config Info to Gen SVF 组框中包含为普通 SVF 属性设置，Set Eflash Config Info to Gen SVF 中为嵌入式 Flash 相关参数设置。具体参数含义请见表 6。用户初次使用后可拷

见 tcl 提示框内指令，以便后续快捷生成 svf。

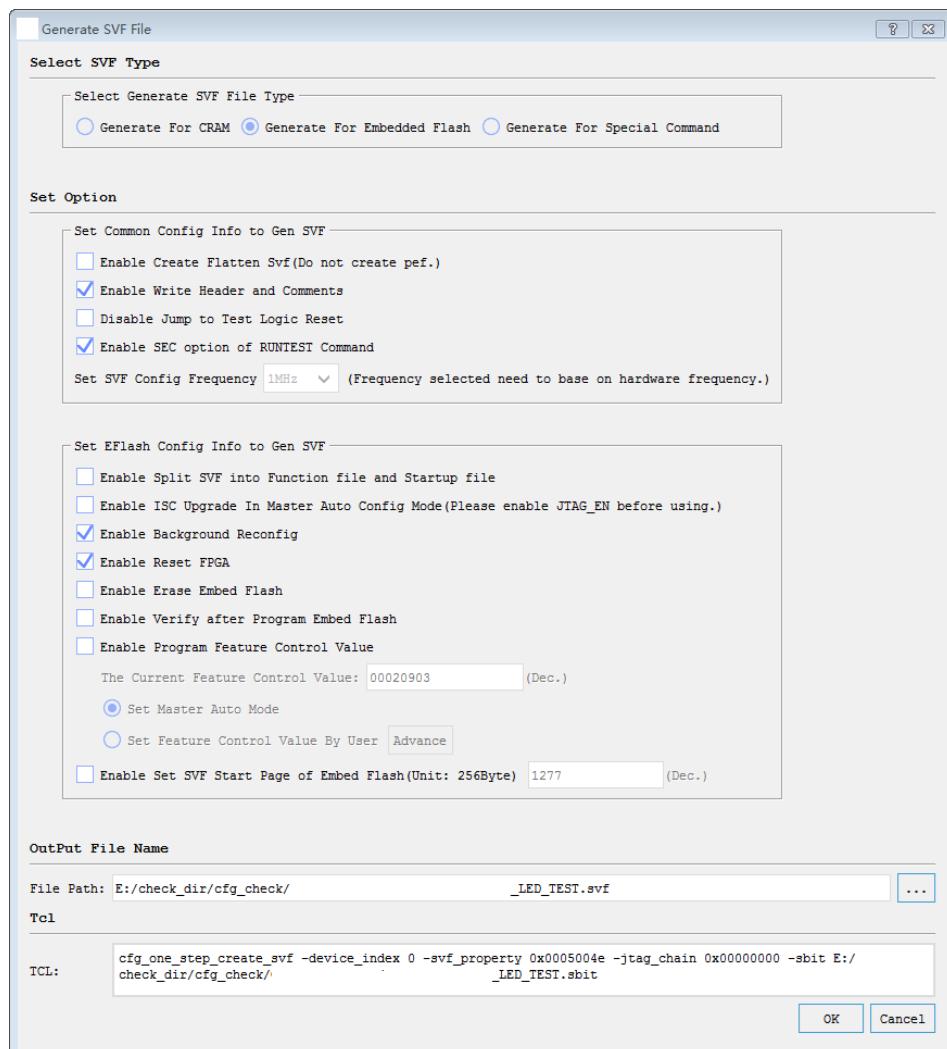


图 4-58 选择生成的 SVF 文件路径与参数

Enable Create Flatten SVF	是否产生SDR/SIR填充链信息的SVF(注：使用此功能将不能使用pef)
Enable Write Header and Comments	是否启用文件头和注释，勾选表示启用，默认勾选
Disable Jump to Test Logic Reset	位流配置流程禁止跳转到TLR
Enable SEC option of RUNTEST Command	RUNTEST指令用于指定JTAG状态机跳转到特定状态下时需要保持的等待时间（具体的等待时长和操作选项相关），等待时间有两种实现方式，TCK时钟打拍和系统延时函数：该选项勾选时，通过系统延时函数实现等待时间；该选项不勾选时，通过TCK时钟打拍实现等待时间，默认勾选
Set SVF Config Frequency	表示设置SVF的配置频率，默认为1MHz。 当勾选“Enable SEC option of RUNTEST Command”时，RUNTEST指令的等待时间与频率无关；当不勾选“Enable SEC option of RUNTEST Command”时，RUNTEST 指令对应的TCK打拍时钟数与频率成正比，此场景下，用户板上的TCK频率需要和软件设置的频率一致，或者低于设置频率；

	如果板上TCK频率大于软件设置频率，可能导致配置失败。
Enable Background Reconfig	是否使能背景加载
Enable Split SVF into Function file and Startup file	把SVF分割成功能文件和启动文件，需先配置功能文件，再配置启动文件
Enable ISC Upgrade In Master Auto Config Mode	主自加载模式下使能在系统配置（In-System Configuration）升级，在芯片远程升级的整个过程中，IO端口状态将保持不变。使用此功能需保证芯片引出的JTAG_EN管脚在芯片远程升级过程中始终被外部拉高
Enable Reset FPGA	使能复位FPGA，默认使能
Enable Set SVF Start Page of Embed Flash	设置SVF文件的起始地址，默认值为合成位流设置下，应用位流的起始页地址
Enable Erase Embed Flash	在编程嵌入式Flash前使能页擦除操作，默认禁用；该操作不是必须的，编程嵌入式Flash的流程中自带页擦除操作，使能该操作会影响SVF配置时间
Enable Verify After Program Embed Flash	在编程嵌入式Flash后使能Verify
Enable Program Feature Control Value	使能编程特征控制位
Set Master Auto Mode	仅设置主自加载模式
Set Feature Control Value by User	设置特征控制位，可设置每一个bit

表 6 嵌入式 Flash SVF 配置参数含义

1.3、生成带特殊指令的 SVF

如图 4-59 所示若要生成带特殊指令的 SVF 则需在 Select Generate SVF File Type 中选取 Generate For Special Command 选项。只有 Compact 系列支持生成带特殊指令的 SVF。用户初次使用后可拷贝 tcl 提示框内指令，以便后续快捷生成 svf。

Select Command to Generate 中包含的特殊指令内容如表 7 所示。

Reset FPGA Command	复位FPGA
Check Done Bit of Status Register Command	检查Done状态是否拉高
Lock Embed Flash Command	锁定嵌入式Flash
Entire Embed Flash Command	擦除整片嵌入式Flash
Enable Verify after Erase Embed Flash	使能擦除完embed flash后verify
Enable Verify after Program Embed Flash	使能编程完embed flash后verify

表 7 特殊指令含义

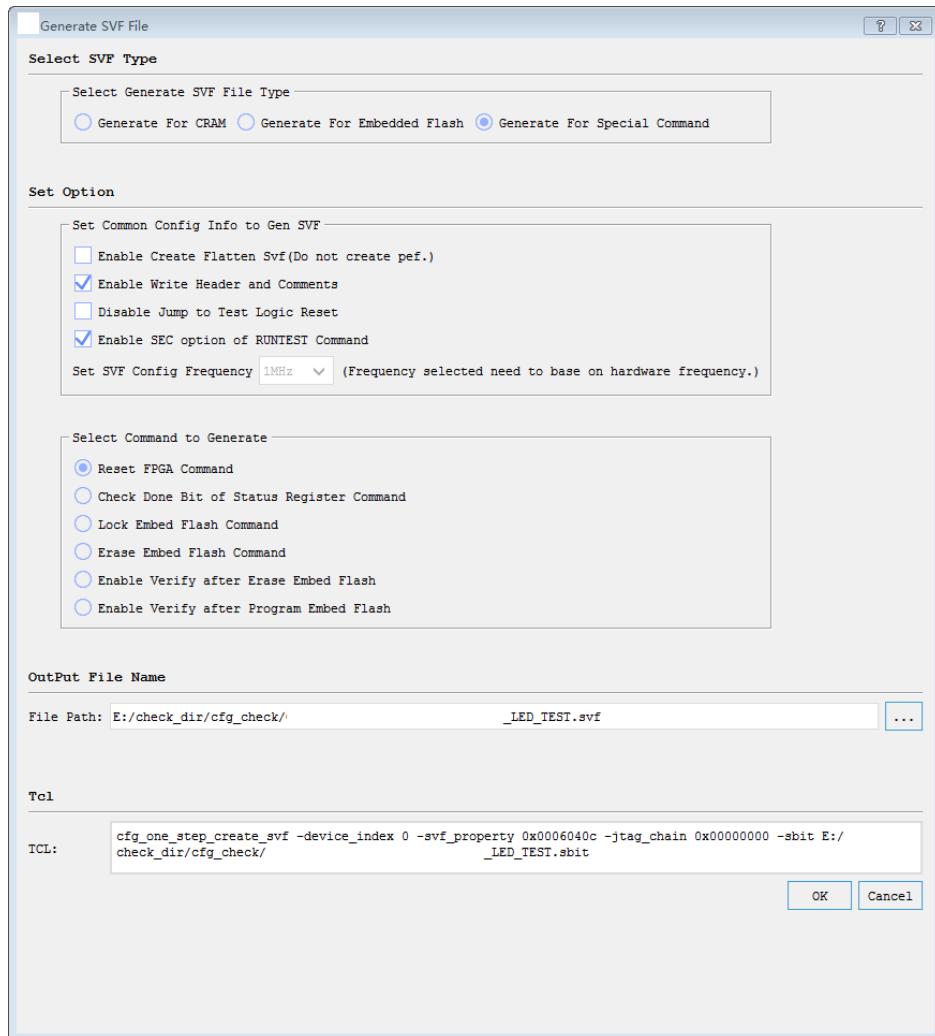


图 4-59 带特殊指令的 SVF

2、生成 PEF 文件

PEF 是对 SVF 文件以一定格式压缩后的二进制文件，在选择一键产生 SVF 时会同步产生 PEF 文件，或者使用 Tcl 命令来产生，命令为 `cfg_svf_to_pef`，具体细节参考 Tcl 命令章节。

3、执行 SVF/PEF 文件

首先要 connect to server（注：扫链操作无法添加），然后右键点击显示器件的空白处，选中下拉菜单（如图 4-10 所示）中的“Add Pango Device”。在弹出的选择框中选择刚才生成的 SVF 文件或者 PEF 文件，添加成功后如图 4-60 所示，此时，用户便可以对 SVF 进行执行操作。在器件处右击弹出菜单，点击 Execute SVF/PEF File 选项，软件将会执行该 SVF 文件或者是 PEF 文件。

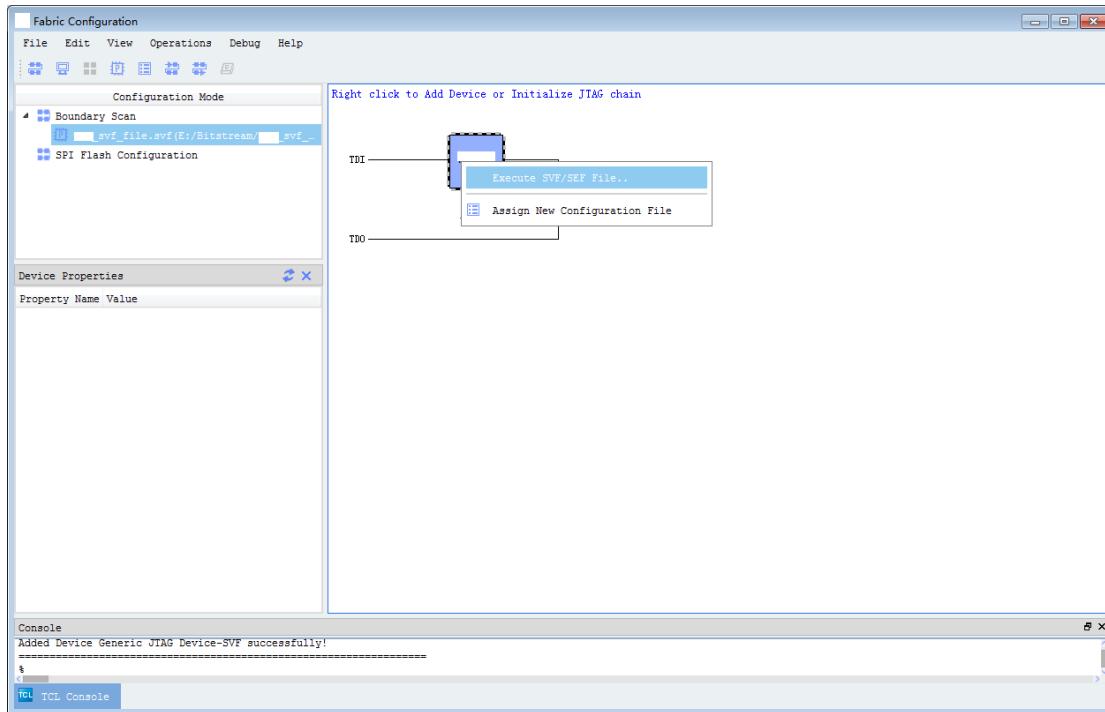


图 4-60 执行 SVF 文件

4.3.21 Tcl 脚本的生成与执行

1、记录用户操作功能生成 Tcl 脚本

在 Operations 菜单如下图 4-61 所示, Tcl Record File 以 tcl 命令形式记录用户操作功能, 点击 Create Tcl Record File 菜单选项可以创建新的 tcl 文件进行记录, 也可通过点击 Append Tcl Record File 菜单选项附加在已存在的 tcl 文件中, 然后可使用 operation 菜单中的 Execute tcl file 功能执行 tcl 文件, 记录时, tcl 文件中记录的是相对路径, 并且在 tcl 文件的当前路径创建 tcl_bit_file 文件夹保存用户的操作的位流文件, 将 tcl 文件和 tcl_bit_file 文件夹同时移到其他路径, 也可执行 tcl 文件。

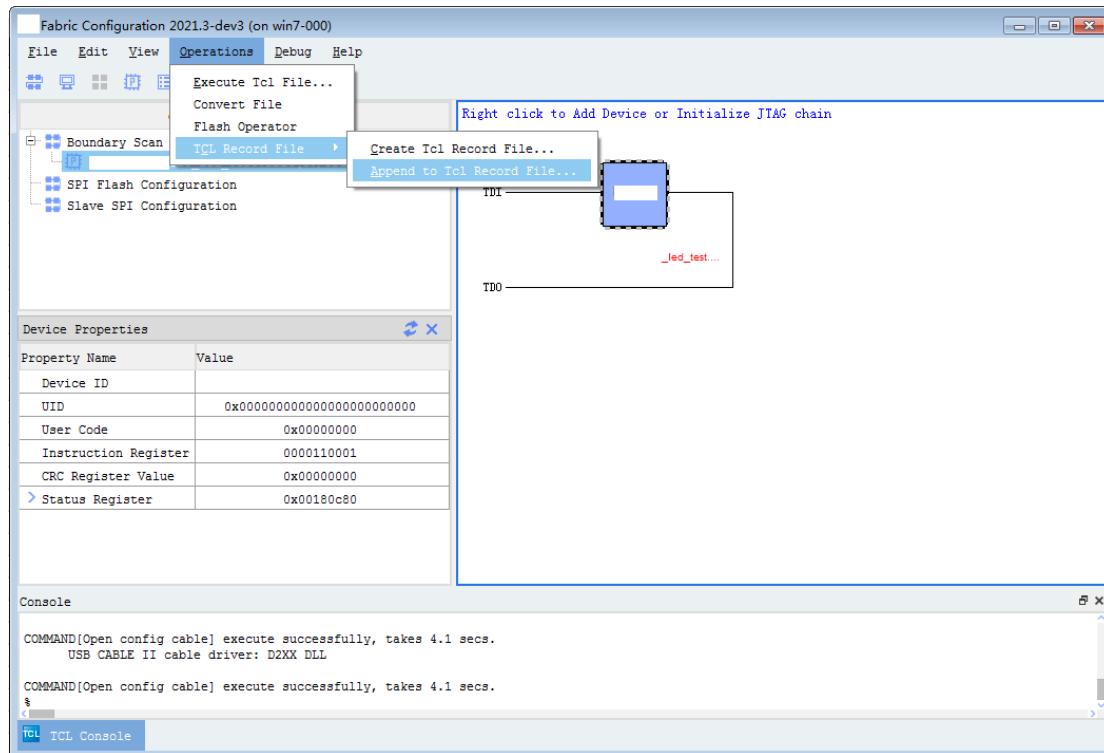


图 4-61 Operations 菜单

其中部分用户操作与 Tcl 命令的对应操作如下所示

Tcl命令	用户操作
cfg_connect	连接Jtag server
cfg_disconnect	断开Jtag server
cfg_add_device	添加FPGA器件到边界扫描链上
cfg_assign_file	对FPGA器件配置位流文件
cfg_program	Jtag编程FPGA器件
cfg_readback	Jtag回读FPGA器件
cfg_verify	Jtag校验FPGA器件
cfg_read_device_property -mode 0	Jtag读取FPGA器件的ID
cfg_read_device_property -mode 1	Jtag读取FPGA器件的UID
cfg_read_device_property -mode 2	Jtag读取FPGA器件的user code
cfg_read_device_property -mode 3	Jtag读取FPGA器件的状态寄存器
cfg_remove_devices	删除边界扫描链上的FPGA器件
cfg_flash_scan_device	SPI扫描Flash器件
cfg_flash_assign_file	对SPI Flash器件配置文件
cfg_flash_program	SPI编程Flash器件
cfg_flash_erase	SPI擦除Flash器件
cfg_flash_readback	SPI回读Flash器件
cfg_flash_verify	SPI校验Flash器件
cfg_flash_remove_device	删除SPI Flash器件操作

表 8 用户操作与 Tcl 命令的对应操作

此项操作需注意以下几点：

1> 记录用户操作作为 tcl 文件时，当用户操作位流文件如对器件配置文件操作、对 SPI Flash 器件配置文件操作等，会将用户操作的文件复制一份到 tcl 文件所在路径的 tcl_bit_file 文件夹下，所记录的路径也是相对 tcl 文件的路径。

2> tcl 记录的是相对路径，因此可以将 tcl 文件以及 tcl_bit_file 文件夹移动到其他路径或其他电脑上进行执行。

2、执行 Tcl 脚本

Fabric Configuration 支持用户进行批量化执行操作(Tcl 命令)，点击菜单栏 Operations 菜单下的 Execute Tcl File 选项，出现如图 4-62 所示选择 tcl 脚本的窗口，选择相应的 Tcl 脚本便可执行。

Fabric Configuration 还支持在 shell 端启动执行脚本，在 shell 端启动时，不会显示 Fabric Configuration 界面。具体操作的方法为：切换到可执行程序所在的目录，然后启动操作系统的 shell 终端，输入命令“cdt_cfg.exe -i p 目标主机 IP -port 目标主机端口 -file Tcl 脚本绝对路径” 例如： cdt_cfg.exe -ip 127.0.0.1 -port 65420 -file E:\pan\project\sbit\PGL50H _tcl_script.tcl 即可。

注：Tcl 命令使用教程见本章第四大节 Tcl 命令介绍。

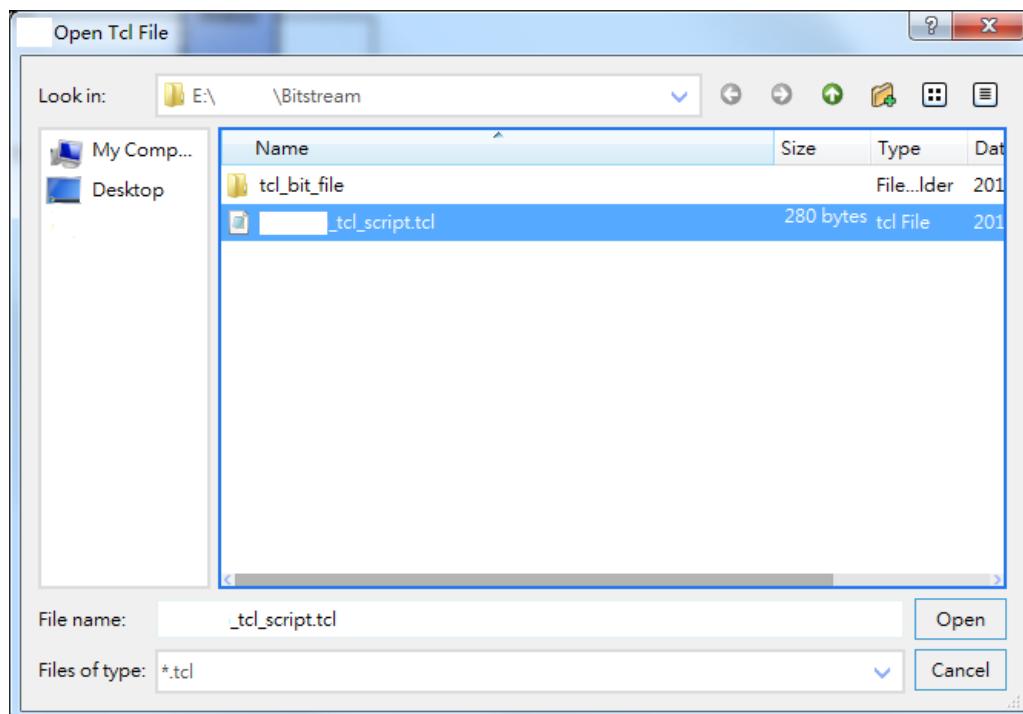


图 4-62 执行 tcl 脚本窗口

4.3.22 设置用户自定义信息

在 Edit 菜单下点击 Preference，弹出的对话框中进行用户自定义信息的设置，如图 4-63 所示。用户控制是否在器件属性窗口中显示位流的 CRC 值和设置用户自定义软件标题。

当打开多个 configuration 软件时，用户可以设置标识区分信息来区别多个 configuration 软件的不同用处。

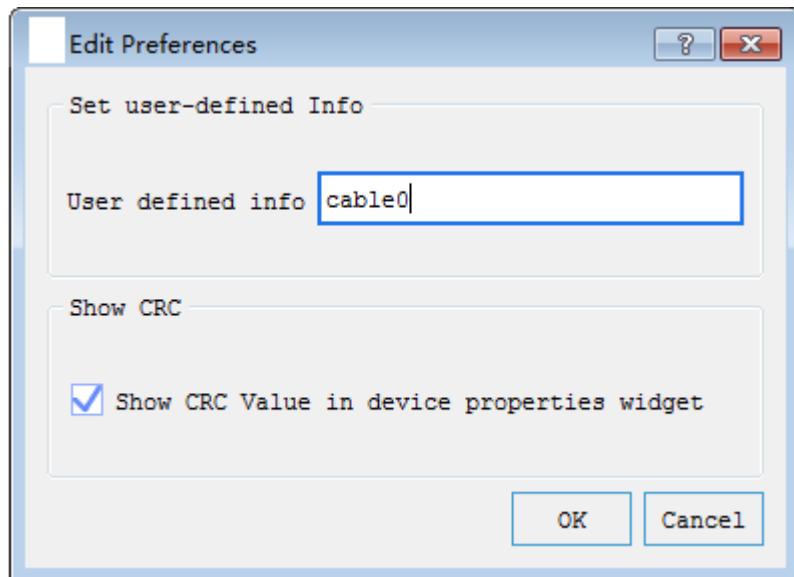


图 4-63 Preference 对话框

设置用户自定义信息成功后，用户的自定义信息会显示在标题栏中，如图 4-64 所示。用户在执行相应的 tcl 脚本时，会生成带用户自定义信息为文件名的 tcl 脚本运行日志，如下图 4-65 所示。

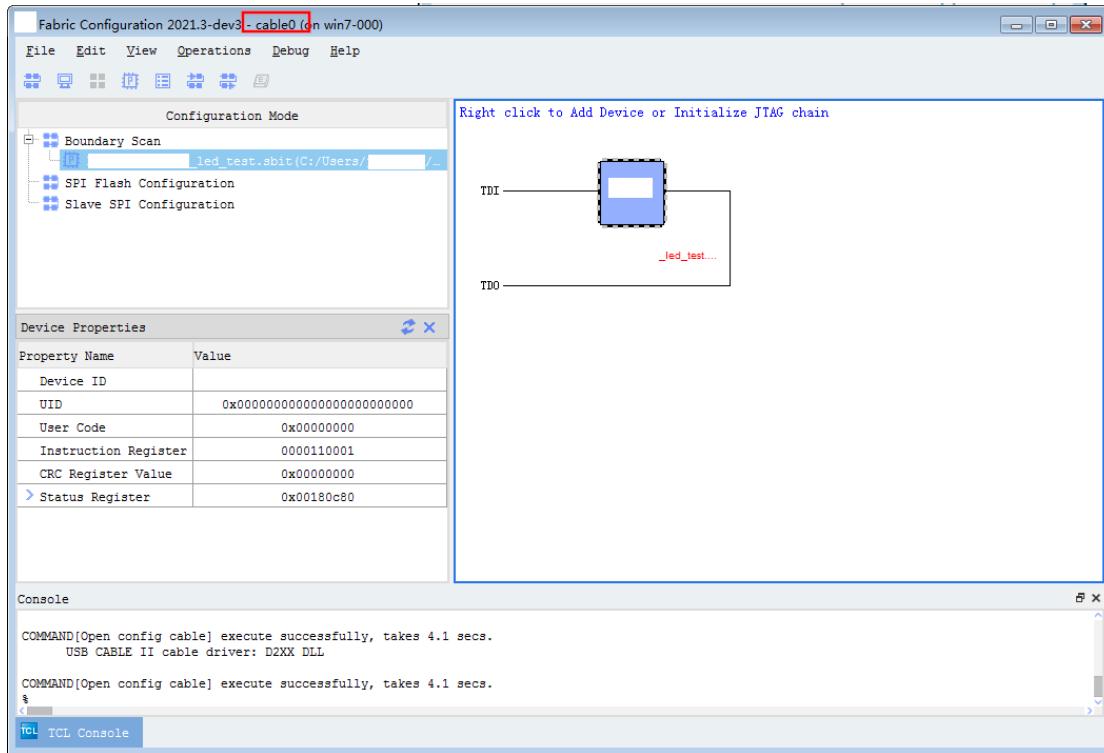


图 4-64 标题栏中用户信息

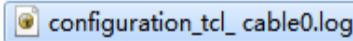


图 4-65 执行 Tcl 后的日志文件

4.3.23 链完整性测试

在 Debug 菜单通过点击 Chain Integrity Test 选项，测试当前所连接的 Jtag 链是否处于通路的状态。如果当前所选择的 Jtag 链处于通路的状态，则显示当前链上器件的数量，如图 4-66 所示。

```
begin to test chain integrity..
The total device number in the chain is : 1
Jtag chain integrity test succeed, takes 0.5 secs.
```

图 4-66 测试链完整性成功

如果测试链连通性失败，则会显示如图 4-67 所示。

```
begin to test chain integrity..
E: Configuration 0006: Fail to detect jtag chain device
E: Configuration 0014: Jtag chain integrity test fail, takes 0.0 secs.
```

图 4-67 测试链完整性失败

4.3.24 设置用户自定义 Flash

在 Operations 菜单通过点击 Flash Operator 选项，弹出的对话框中进行用户自定义 FLASH 的设置，如图 4-68 所示。用户通过选择 Flash Type 来显示现在以支持的全部器件或者用户自定义添加的全部器件（只有用户自定义添加的器件才可以删除）。

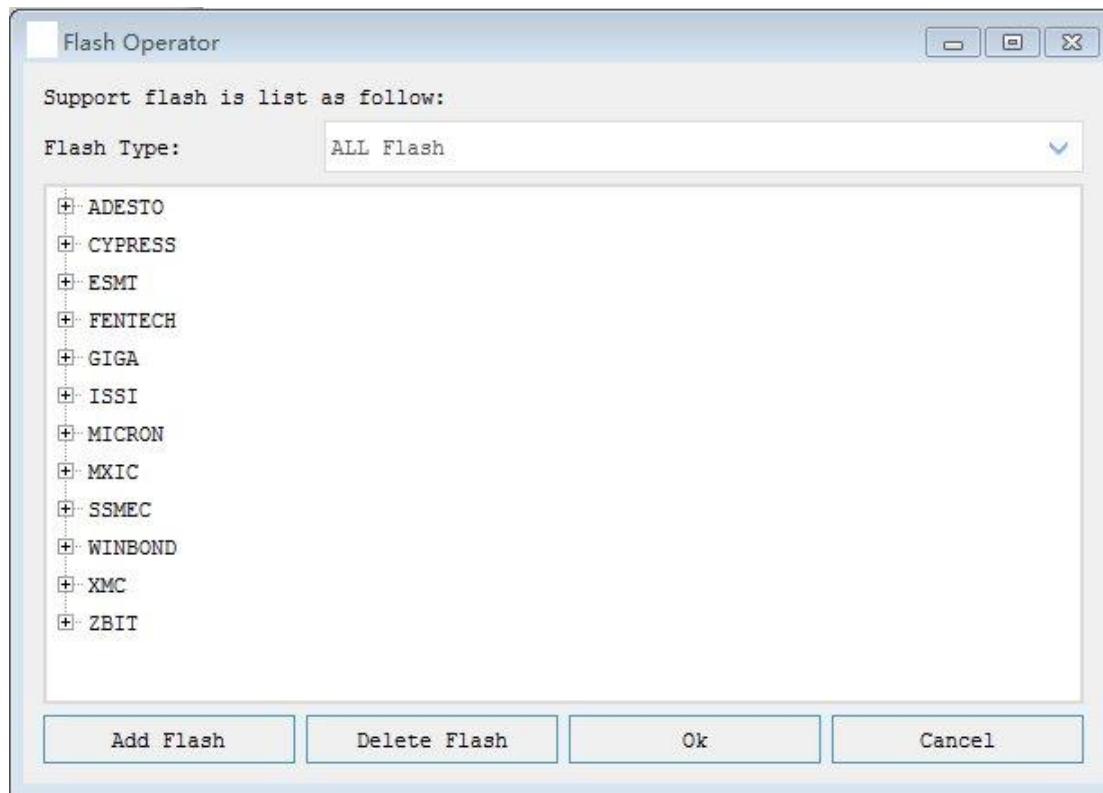


图 4-68 Flash Operator 主界面

用户通过点击 Add Flash 按钮来添加新器件。添加界面如图 4-69 所示，首先选择需要添加的厂商和系列，点击 next 后进入器件具体信息设置界面，包括器件名和器件 ID (如图 4-70)。最后进入总结界面点击 Finish 完成添加。

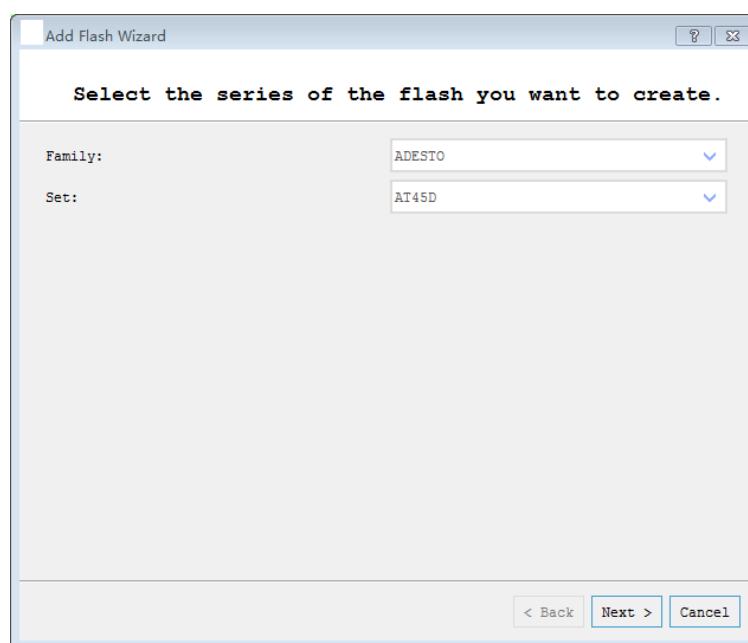


图 4-69 新增 Flash 厂商系列选择界面

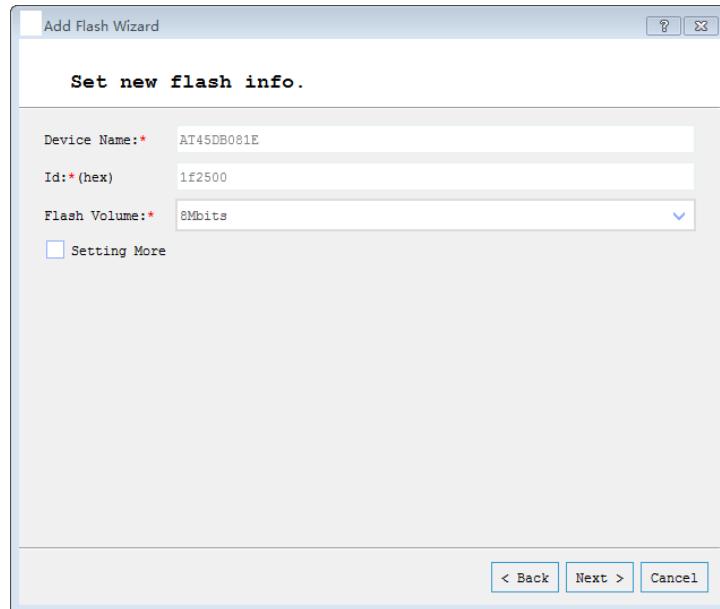


图 4-70 器件信息设置界面

添加 Flash 完成后要点击 Ok 按钮才能完成真正的添加，点击 Cancel 或者关闭窗口将会回退相关添加。

4.3.25 添加用户自定义器件

在工作区鼠标右键单击，会弹出如图 4-10 所示菜单，左键单击“Add Non-Pango Device”，界面如图 4-71 所示，执行添加用户自定义器件的操作。用户可以通过手动输入器件名称和器件指令长度来添加自定义的器件。扫链扫到 Non-Pango Device 时也需要在弹出的窗口输入器件信息，否则无法进行其他操作。

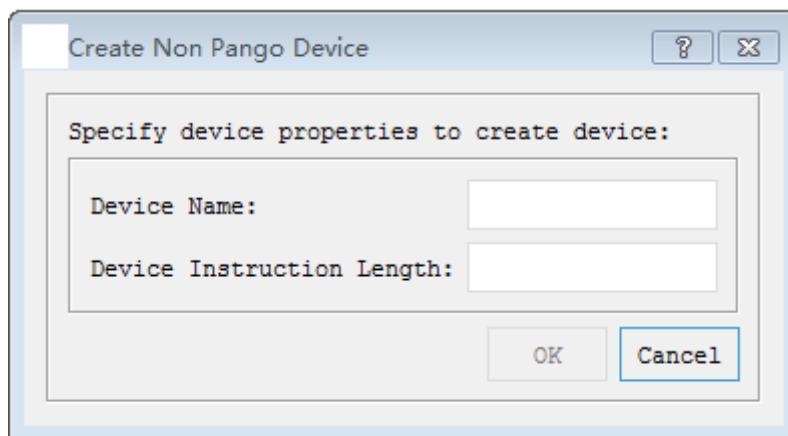


图 4-71 添加用户自定义器件

4.3.26 Jtag 间接烧写外部 flash

Logos、Logos2、Titan2 部分器件支持间接烧写外部 flash，以 Titan2 系列器件为例，在所选中器件上右键单击，会弹出如图 4-73 所示选项栏，点击“operate outer flash through fpga”，弹出如图 4-74 所示界面。

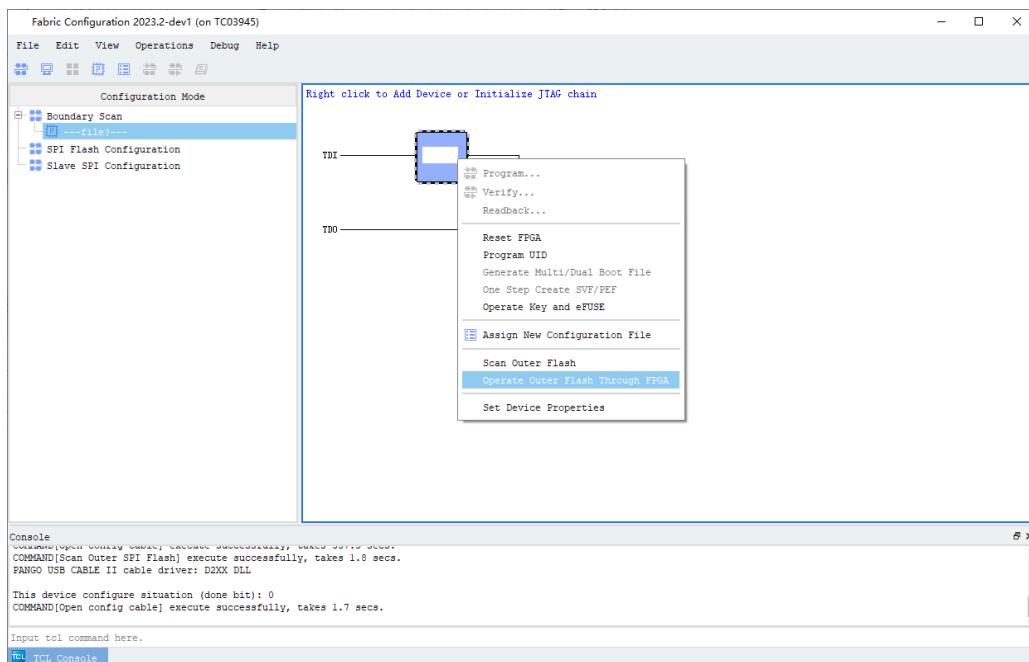


图 4-72 Jtag 间接烧写外部 flash 选项位置

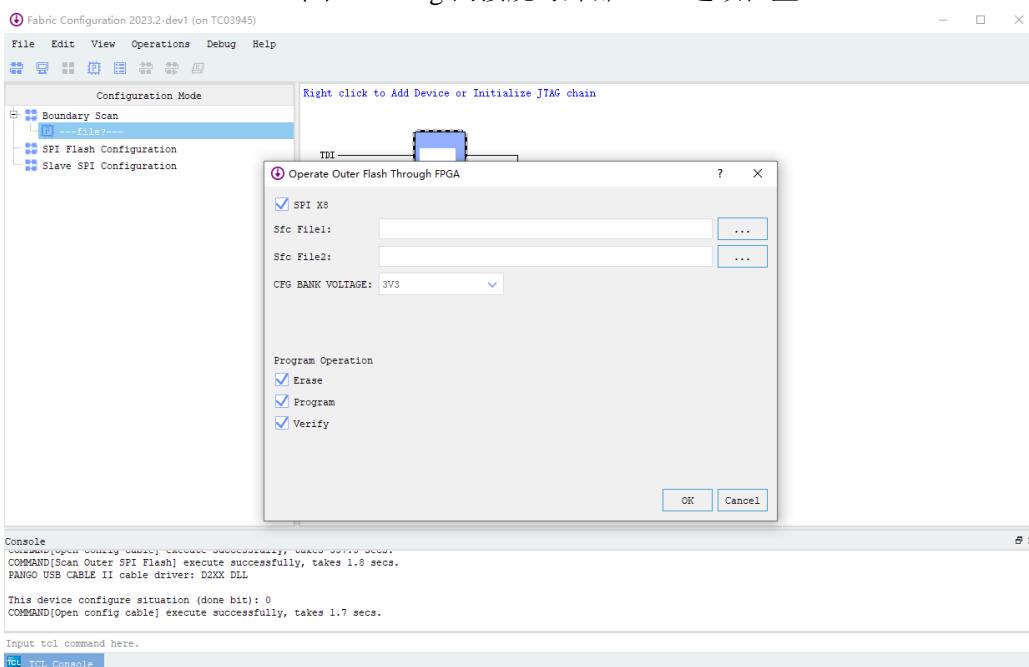


图 4-73 Jtag 间接烧写外部 flash 界面

如果 sfc 文件是 SPI X8 格式（Logos 不支持 X8），需要勾选“SPI X8”，输入两个 sfc 文件，勾选“SPI X8”后才能选择“Sfc2 file”文件地址，“CFG BANK VOLTAGE”需根据实际应用选择对应的电压。通过勾选“Erase”、“Program”、“Verify”选择要做的操作，只有勾选了“Program”，才能勾选“Verify”。

4.4 Tcl 命令使用入门

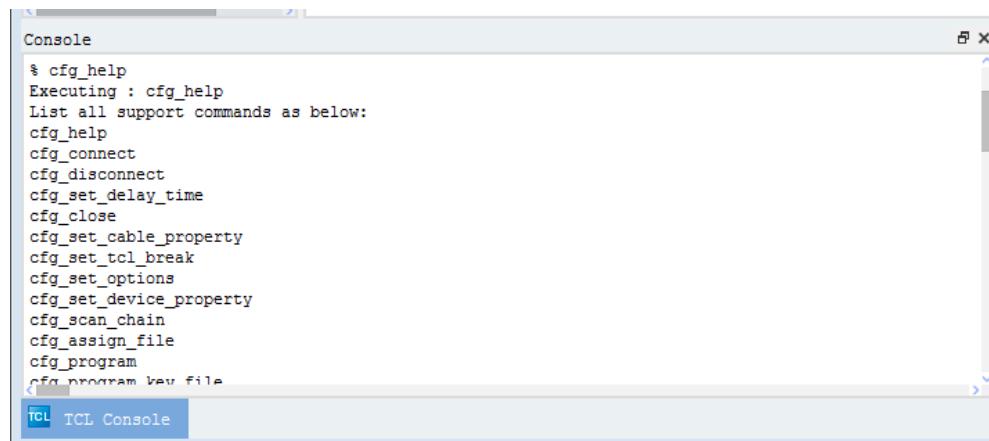
4.4.1 Tcl 命令简介

Fabric Configuration 工具支持部分原生的 Tcl 语言命令如 cd, source 等，并且支持一些我司自定义的 Tcl 命令如 cfg_program 等。

对于一系列重复性的界面操作，用户可以通过在 Tcl 控制台一次性输入多条 Tcl 指令或执行 (*.tcl) 文件，让软件自动执行完这些操作，以提高用户工作效率。

4.4.2 查看支持的所有 Tcl 命令

在 Fabric Configuration 工具的控制台执行“cfg_help”这条指令，可查看当前版本支持的所有 Tcl 指令，如图 4-74 所示。



```
Console
% cfg_help
Executing : cfg_help
List all support commands as below:
cfg_help
cfg_connect
cfg_disconnect
cfg_set_delay_time
cfg_close
cfg_set_cable_property
cfg_set_tcl_break
cfg_set_options
cfg_set_device_property
cfg_scan_chain
cfg_assign_file
cfg_program
cfg_program_hex_file
```

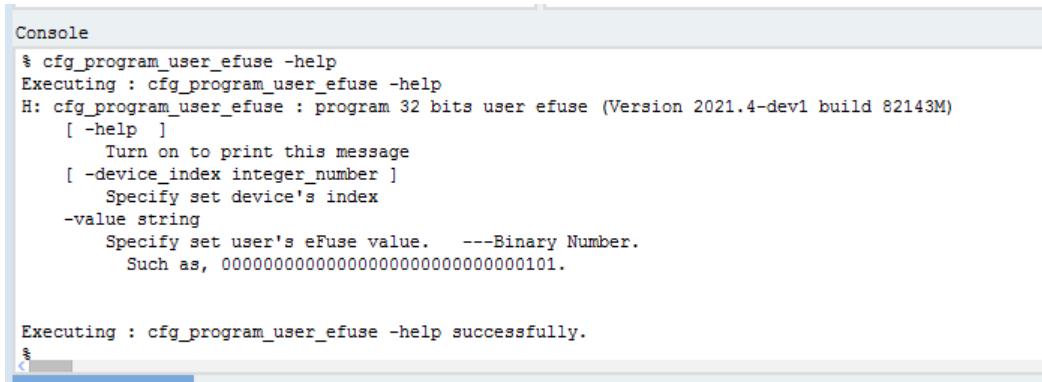
图 4-74 cfg_help 显示所有支持的 Tcl 命令

4.4.3 查询具体 Tcl 命令的参数

1、控制台查询

用户想要查询具体某个指令的用法，可通过在控制台执行 “cfg_xxx -help” 来打印出来该指令的具体使用方法。

如下图图 4-75 在控制台输入 “cfg_program_user_efuse -help”，可以查询到该指令有三个参数选项，其中带中括号[]为可填选项，不带中括号[]为必填选项。每条 Tcl 指令均有 [-help] 选项方便用户查询此条 Tcl 命令的具体用法。



```
Console
$ cfg_program_user_efuse -help
Executing : cfg_program_user_efuse -help
H: cfg_program_user_efuse : program 32 bits user efuse (Version 2021.4-dev1 build 82143M)
[ -help ]
    Turn on to print this message
[ -device_index integer_number ]
    Specify set device's index
-value string
    Specify set user's eFuse value. ---Binary Number.
    Such as, 00000000000000000000000000000101.

Executing : cfg_program_user_efuse -help successfully.
$
```

图 4-75 控制台查询 cfg_program_user_efuse 的具体用法

2、查询用户手册

用户还可以通过查看 Fabric_Configuration_User_Guide (Configuration 用户使用手册)

第 4 章 Fabric Configuration 软件说明的(五) Tcl 命令具体参数查询来查询某个具体指令的用法。

注：若提示命令格式非法，则是用户所输入的参数格式不符合该参数选项的格式要求，用户可以输入“cfg_xxx -help”来查看该命令的输入参数的格式要求。

用户在输入 Tcl 命令时应注意命令，选项，参数之间都要用空格隔开；

参数选项前的“-”为英文输入法下的“-”，若用户输入中文的“-”将会提示所输入命令为无效命令；

如图 4-75 所示的 Tcl 命令，用户在实际使用时只需在控制台输入

cfg_program_user_efuse -value 1010

4.4.4 获取界面操作的 Tcl 命令

对于一系列重复性的界面操作，用户希望能够通过输入多条 Tcl 命令或执行 (*.tcl) 文件，让软件自动执行完这些操作，以提高用户工作效率，但是用户一开始可能并不知道哪些操作对应哪些 Tcl 命令，以及 Tcl 命令的正确执行顺序（注：执行多条 Tcl 命令时，错误的执行顺序会使 Tcl 命令执行失败）。

因此 Fabric Configuration 工具提供了两种方式能够方便用户快速获取自己执行的一系列界面操作所对应的一系列 Tcl 命令。这两种获取方式在用户使用 Tcl 命令的过程中非常实用。

1、从操作界面下方的 Tcl 框里获取

在一些特定功能的操作界面下方（如 svf/pef 文件生成，文件转换功能，编程 efuse 等），

会根据用户在界面设置的参数，在 Tcl 框内实时生成对应的一系列 Tcl 命令，如图 4-76 所示。

用户既可以在界面通过点击相关按钮直接实现该功能，也可以通过复制框内的 Tcl 命令去控制台粘贴来实现该功能。对于重复性高的操作，用户可以保存生成的 Tcl 命令，方便下次直接使用 Tcl 命令替代一系列界面操作。

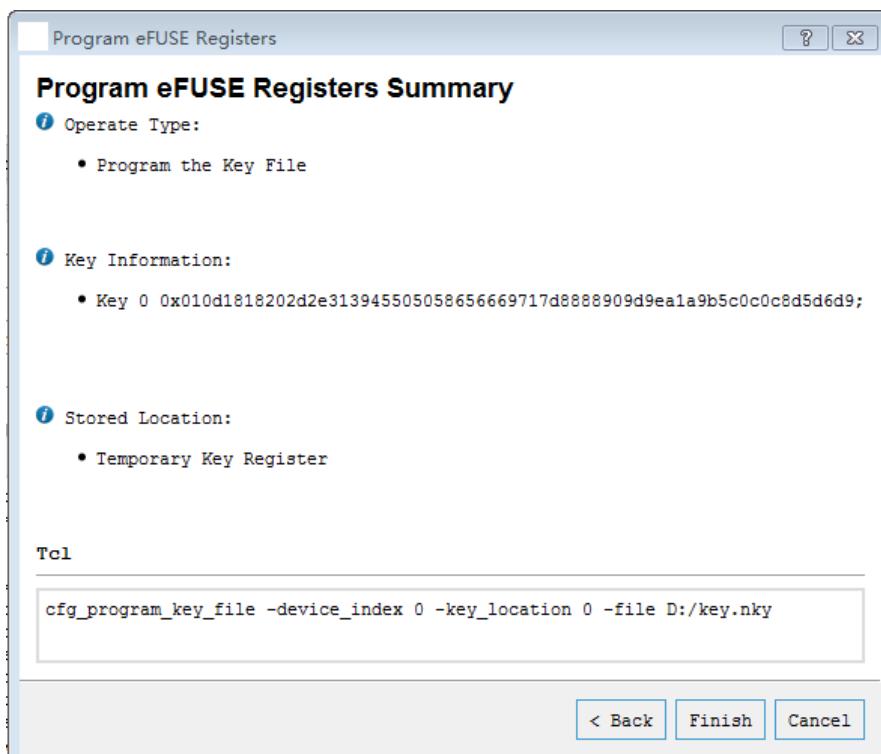


图 4-76 烧写临时秘钥文件的操作界面

2、从 Tcl 记录功能生成的 tcl 文件里获取

用户还可以通过点击菜单栏 Operations - TCL Record File - Create Tcl Record File，创建 (*.tcl) 文件，用来记录用户接下来的所有界面操作所对应的 Tcl 命令。 (*.tcl) 文件里记录的 Tcl 命令可供用户参考使用。Tcl 记录功能使用说明详见 **4.3.21 Tcl 脚本的生成与执行**。

4.4.5 执行 Tcl 命令

用户获取 Tcl 命令后，有很多执行 Tcl 命令的方法，用户可根据实际使用情况挑选合适的执行方式。

1、在 Configuration 软件的控制台下执行（包括 Windows 版和 Linux 版）

在 Configuration 软件的控制台有四种方式可以执行 Tcl 命令：

方式一：在控制台输入或粘贴 Tcl 指令，如 `cfg_program_user_efuse -value 1010`。

方式二：在控制台输入 `source` 指令和 tcl 文件的地址，如 `source D:/test.tcl`。

方式三：点击菜单栏 Operations - Execute Tcl File，选择目标 tcl 文件。

方式四：直接把 tcl 文件拖入控制台窗口，以执行 tcl 文件里的指令，如图 4-77。

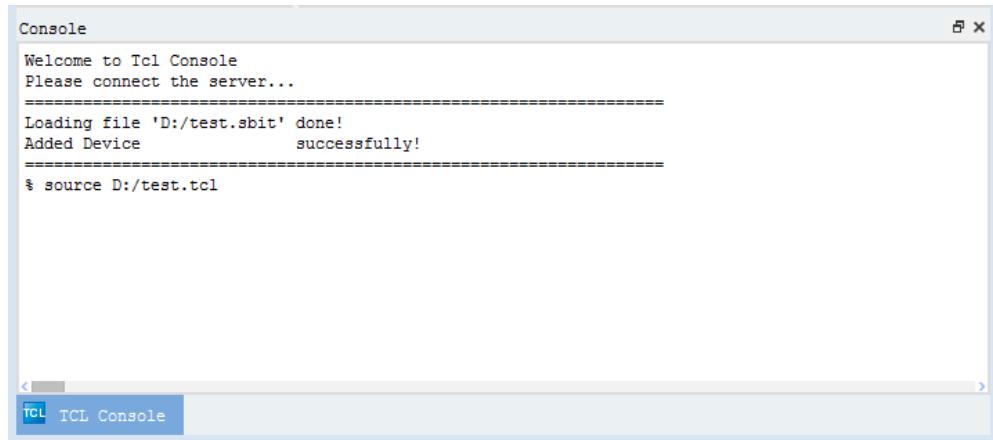


图 4-77 tcl 文件拖入控制台窗口执行

2、在 Windows cmd 命令行下执行

在 Windows 环境下还可以使用 cmd 调用 cdt_cfg.exe 执行 tcl 文件，如图 4-78 所示。支持以下参数设定：

[-file] 要解析的目录*.tcl

[-args] 用于顺序替换 tcl 文件里 %s 占位符的数据，不同数据用空格分隔，用花括号包围数据。

用法：在 cmd 下调用 cdt_cfg.exe 时，支持用户在 (*.tcl) 文件中使用 %s 作为占位符，来替代 tcl 文件中的某些参数或者指令。然后用户在 cmd 输入 -file *.tcl -args { data1 data2 ... datan } 来执行该 (*.tcl) 文件。这样花括号里的 data 将会按顺序依次替换 tcl 文件中的 %s 占位符，花括号里的 data 数据需用空格分隔开。

使用示例：

假设 test.tcl 文件中有命令语句 cfg_add_device -file %s

在 cmd 下调用 cdt_cfg.exe，并输入 -file test.tcl -args { D:/a.sbit }

即可将 test.tcl 文件里的 %s 占位符替换成 D:/a.sbit。

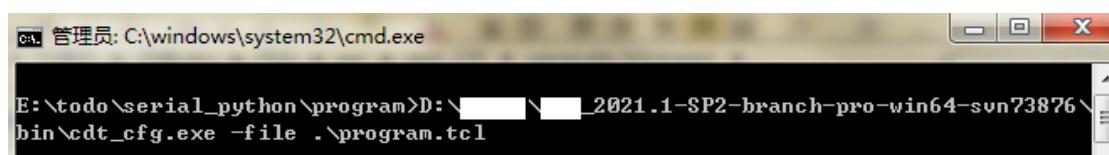


图 4-78Windows 下用 cmd 调用 cdt_cfg.exe

3、在 Configuration 的终端模式下执行

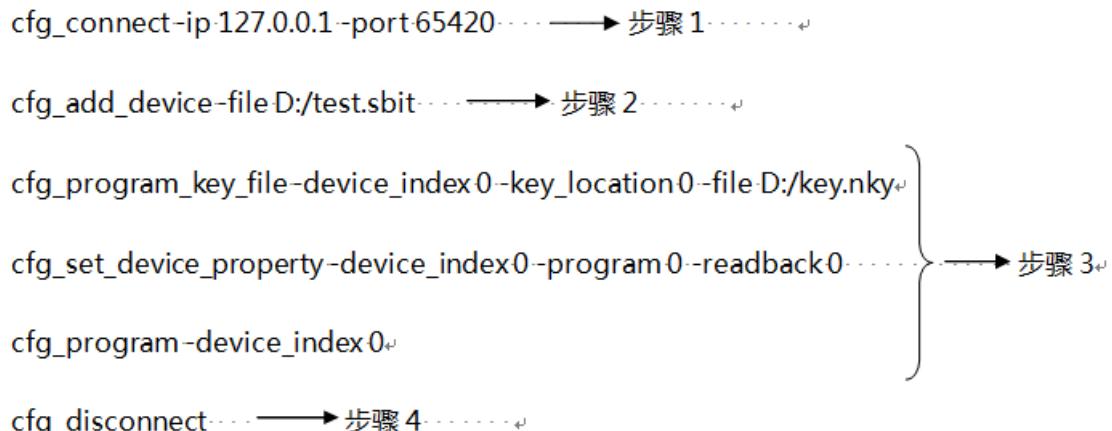
启动“cdt_cfg_shell.exe”，在 Configuration 的终端模式下执行 Tcl 命令或者 tcl 文件，

具体操作详见章节 4.6。

4.4.6 Tcl 的使用示例

下图为 tcl 文件里 Tcl 命令的使用示例，并对各部分 Tcl 命令进行了解释。

```
cfg_connect -ip 127.0.0.1 -port 65420 ..... → 步骤 1 .....  
cfg_add_device -file D:/test.sbit ..... → 步骤 2 .....  
cfg_program_key_file -device_index 0 -key_location 0 -file D:/key.nky .....  
cfg_set_device_property -device_index 0 -program 0 -readback 0 .....  
cfg_program -device_index 0 .....  
cfg_disconnect ..... → 步骤 4 .....
```



步骤 1：连接 JTAG server，这是进行 JTAG server 操作前的必要步骤。其中-ip、-port 默认为连接本地电脑，若连接本地电脑的 JTAG server，保持上述默认值即可。

步骤 2：添加目标器件并指定目标文件，目标文件需替换为客户实际文件（若添加多个器件，则最后添加的器件对应 device_index 为 0，并且靠近 TDI）。

步骤 3：可选执行操作，根据用户具体要执行的操作决定。上述示例为在芯片中烧写临时秘钥文件 key.nky 并烧写用此秘钥加密过的位流文件 test.sbit 的操作。秘钥文件需替换为用户实际秘钥文件名。

步骤 4：断开 JTAG server 连接，结束操作。

注：步骤 1,2,4 为必要的常规步骤，步骤 3 为用户每次具体需要进行的操作。

4.5 Tcl 命令具体参数查询

目前 Configuration 工具支持如下我司自定义的 Tcl 命令。

4.5.1 通用命令的说明和用法

1. source

source 命令是 Tcl 的内置命令，用来执行 Tcl 脚本。

用法： source C:/Users/Test/Desktop/test.tcl

2. **cfg_help**

cfg_help, 列出当前所有支持的自定义 Tcl 命令。 无参数

用法： cfg_help

3. **cfg_connect**

cfg_connect, 用于连接 JtagServer 服务器的命令，包含 2 个参数 -ip, -port

用法： cfg_connect -ip 192.29.103.191 -port 65420

4. **cfg_disconnect**

cfg_disconnect, 用于断开与 JtagServer 服务器连接的命令，无参数

用法： cfg_disconnect

5. **cfg_set_delay_time**

cfg_set_delay_time, Fabric Configuration 延时命令，以秒(seconds)为单位，该命令会使 CFG 延时等待所设置的时间。包含 1 个参数-time。

用法： cfg_set_delay_time -time 3

6. **cfg_close**

cfg_close, 关闭 Fabric Configuration 软件，无参数

用法： cfg_close

7. **cfg_set_cable_property**

cfg_set_cable_property, 设置当前 Jtag Server 上所连接的 cable 参数，包含 2 个参数。

-index : 选择将要进行操作的 cable 序号

-freq: 设置 cable 的频率值，当前支持的频率值有：{15 MHz,10 MHz,7.5 MHz, 6 MHz ,5 MHz ,2 MHz, 1 MHz ,500 KHz ,300 KHz, 100 KHz}。

用法： cfg_set_cable_property -index 0 -freq 10Mhz:表示将序号为 0 的 cable 频率设置为 10MHz。

8. **cfg_set_tcl_break**

cfg_set_tcl_break, 设置 Tcl 遇到 error 是否往下执行(执行 Tcl 文件)

用法: `cfg_set_tcl_break -flag true` 表示 Tcl 遇到 error 中断不继续往下执行。

9. **cfg_set_options**

cfg_set_options, 设置 Jtag Server 的运行属性, 一个参数

- report_detail 设置板上自动化是否写详细报告, 参数类型 Boolean_value
(TRUE/FALSE)

[-server_delay_time] 设置 server 的延时时间, 即延时多长后, server 开始读取数据。参数类型为大于 0 的整数

用法: `cfg_set_options -rpt_detail true -server_delay_time integer-value`

4.5.2 针对 FPGA 器件的命令

包含对 FPGA 的 ID、user code 及状态寄存器读取功能, 逻辑位流下载、回读及校验等 Tcl 命令。

1. **cfg_scan_chain**

cfg_scan_chain, 扫描 Jtag 链命令 (初始化链操作) 无参数

用法: `cfg_scan_chain`

2. **cfg_assign_file**

cfg_assign_file, 配置位流文件命令, 包含 2 个参数

-file 配置位流文件的路径

[-device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: `cfg_assign_file -file D:/bs/counter.sbit -device_index 0`

3. **cfg_set_device_property**

cfg_set_device_property, 设置器件编程、回读、嵌入式 Flash 属性, 包含 4 个参数

[-device_index] 指定 device 索引, 可选, 默认缺省为 0

[-program] 指定编程属性, 为 16 进制数

设置举例: 若 Bit0 和 Bit1 同时置 1, 则 program 值为 3

注：对于 Compact 系列，Bit1 和 Bit2 不能同时置 1

Bit0：关断配置（Compact\Logos\Logos2\Titan2）

Bit1：ISC 配置（Compact\Logos\Logos2\Titan2）

Bit2：离线配置（Compact）

[-readback] 指定回读属性，为 16 进制数

设置举例：若 Bit0 和 Bit1 同时置 1，则 readback 值为 3

注：对于 Compact 系列，Bit1 和 Bit2 不能同时置 1

Bit0：关断回读（Compact\Logos\Logos2\Titan2）

Bit1：ISC 回读（Compact\Logos\Logos2\Titan2）

Bit2：离线回读（Compact）

[-eflash] 指定嵌入式 Flash 属性，为 16 进制数

设置举例：若只将 Bit0 和 Bit6 同时置 1，则 eflash 值为 41

设置情况：

① 仅设置 Bit0-Bit7 属性，则软件自动补全 Bit8-Bit51 属性值

② 同时设置 Bit0-Bit51 属性，有两种方式可设置。第一种，可通过界面设置指定的 eflash 属性，点击 OK 后 Tcl 控制台会打印出该属性值，具体属性设置操作详见 4.3.19；第二种，使用记录用户操作功能，首先设置器件属性，编程器件，即可生成 tcl 脚本，打开脚本文件即可获取 tcl 命令设置属性，具体操作详见 4.3.22

注：eflash 默认值为 4fd004fc000e6，仅 Compact 系列支持该属性设置

Bit0：表示在下载位流之前是否使能擦除 EFlash

Bit1：表示在下载位流之后是否使能验证 EFlash

Bit2：表示擦除 Eflash 时，是否整片擦除

Bit3：表示擦除 EFlash 之后是否使能复位

Bit4：表示编程 EFlash 之后是否使能复位

Bit5：表示编程特征控制位之后是否使能复位

Bit6：表示在下载位流之后是否使能编程特征控制位

Bit7：表示在编程 CRAM 之前是否使能编程特征控制位

Bit19-Bit8：操作 Embed Flash 的起始地址页，用于设置编程、擦除、回读的起始地址页

Bit31-Bit20：操作 Embed Flash 的尾地址页，用于锁定 Embed flash 从开始到该指定

页的内容

Bit51-Bit32: 指定回读从操作 Embed Flash 的起始地址页开始的内容大小，单位为 Byte

用法: cfg_set_device_property -device_index *integer-value* -program *hex-value* –readback *hex-value* -eflash *hex-value*

4. **cfg_program**

cfg_program, 编程位流命令, 包含 2 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_program -device_index 0

5. **cfg_program_key_file**

cfg_program_key_file, 编程位流加密密钥命令, 包含 3 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

[–key_location] 指定密钥存储位置, 默认为临时密钥寄存器;

0: Temporary Key Register; 1: eFuse Register; 2: Battery Backed RAM

–file: 指定密钥文件的路径信息

用法: cfg_program_key_file -device_index 0 -key_location 0 -file D:/bs/ led.nky

6. **cfg_readback**

cfg_readback, 回读位流文件命令, 包含 2 个参数

–file: 指定回读文件的路径信息

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_readback -file D:/bs/test.sbit -device_index 0

7. **cfg_verify**

cfg_verify, 验证位流文件命令, 包含 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_verify -device_index 0

8. **cfg_read_device_property**

cfg_read_device_property, 获得器件属性信息, 包含 2 个参数

-mode 获取器件属性的模式即获取哪一个属性

- 0: 获取器件 ID
- 1: 获取器件 UID
- 2: 获取器件 UserCode
- 3: 获取器件状态寄存器
- 4: 读取器件 EFuse 的值

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法:

读取 ID: cfg_read_device_property –mode 0 -device_index 0

读取 UID: cfg_read_device_property –mode 1 -device_index 0

读取 UserCode: cfg_read_device_property –mode 2 -device_index 0 读取状态寄存器: cfg_read_device_property –mode 3 -device_index 0

9. cfg_reset_fpga

cfg_reset_fpga, 对 FPGA 器件进行软复位, 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_reset_fpga –device_index *integer-value*

10. cfg_operate_ctl

cfg_operate_ctl, 操作 Compact 系列器件的特征控制位, 3 个参数;

-mode 操作特征控制位的模式(0:编程, 1:回读, 2:擦除)。

[-value] 当操作模式为编程时, 为必选参数, 输入的十六进制值为所下载的特征值。输入值得范围为 0x00000000 – 0xFFFFFFFF。

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_operate_ctl –mode *integer-value* –value *hex-value*

–device_index *integer-value*

11. cfg_operate_eflash

cfg_operate_eflash, 操作 Compact 系列的嵌入式 Flash, 3 个参数;

-mode 对嵌入式 Flash 的操作模式(0:休眠, 1:唤醒, 2:锁定, 3:编程, 4:回读, 5:擦除, 6:verify)。

[-file] 操作嵌入式 Flash 所需 sbit 文件, 下载位流文件或者回读文件

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_operate_eflash –mode *integer-value* –device_index *integer-value* –file
E:/pan/eflash.sbit

12. cfg_read_register

cfg_read_register, 读取指定地址寄存器的内容。包含 2 个参数,
-address (指定寄存器的地址)
[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_read_register –address 123 –device_index 0

13. cfg_write_register

cfg_write_register, 写指定地址寄存器的内容。包含 3 个参数,
-address (指定寄存器的地址)
-value (给定写入指定寄存器的值)
[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_write_register –address 123 –value 123 -device_index 0

14. cfg_add_device

cfg_add_device, 添加器件命令。包含 1 个参数,
-file 配置的文件路径
用法: cfg_add_device -file D:/bs/counter.sbit

15. cfg_remove_device

cfg_remove_device, 删除指定器件命令。包含 1 个参数,
[–device_index] 指定 device 索引, 可选, 默认缺省为 0
用法: cfg_remove_device -device_index 0

16. cfg_remove_devices

cfg_remove_devices, 删除多个指定的器件的命令。包含 1 个参数, -device_index (要
删除的多个器件索引集合)
用法: cfg_remove_devices -device_index {0 1}

17. cfg_remove_all_devices

cfg_remove_all_devices, 删 除所有器件命令，无参数。

用法: cfg_remove_all_devices

18. cfg_program_user_efuse

cfg_program_user_efuse, 编程 32 位用户 eFuse 寄存器命令。包含 2 个参数,

-value (给定写入 eFuse 寄存器的值, 32 位二进制数)

[-device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_program_user_efuse -value 000111 -device_index 0

19. cfg_debug_operate

cfg_debug_operate, 一些调试功能命令, 包含一些功能指令必要或可选的参数。

-mode 选择具体某个调试功能的模式

0: 读正反检测位

1: 写正反检测位

2: 读 RDDEBUGR 寄存器

3: 读指令寄存器

4: 读校准状态寄存器

5: 读边界扫描链管脚输出值

6: 回读 PGC7KD 的嵌入式 flash 的 SM 页数据到指定文件里

200: 禁止器件 JTAG 接口

201: 禁止扫描链模式

[-value string] 写入的数值 (Hex) ; 指定回读的 txt 文件

用法: cfg_debug_operate -mode 1 -value 0x1

cfg_debug_operate -mode 6 -value *.txt

4.5.3 针对 Flash 器件命令

该集合下的 Tcl 命令主要通过 SPI 接口对板上的 Flash 器件进行下载、删除、回读及校验等操作。

1. **cfg_flash_scan_device**

cfg_flash_scan_device, 扫描 SPI flash 器件命令。

用法: cfg_flash_scan_device

2. **cfg_flash_assign_file**

cfg_flash_assign_file, 配置 flash 位流文件命令, 包含 1 个参数,

-file (要配置的文件)

用法: cfg_flash_assign_file -file D:/source/bit/counter_16.sfc

3. **cfg_flash_program**

cfg_flash_program, 下载 flash 位流文件命令, 无参数。

用法: cfg_flash_program

4. **cfg_flash_erase**

cfg_flash_erase, 擦除 flash 位流文件命令, 1 个参数。

[-mode integer_number] 设置 Flash 的擦除模式

0: auto selected Mode by file size 1: Block Erase 2: Chip Erase

3: Sector Erase 4: Page Erase

用法: cfg_flash_erase -mode integer_number

5. **cfg_flash_readback**

cfg_flash_readback, 回读 flash 位流文件命令, 包含 1 个参数,

-file (要回读位流保存的文件)

用法: cfg_flash_readback -file D:/source/bit/testLUT5_readback.sfc

6. **cfg_flash_verify**

cfg_flash_verify, 验证 flash 位流文件, 无参数。

用法: cfg_flash_verify

7. **cfg_flash_remove_device**

cfg_flash_remove_device, 删 除 flash 器件命令, 无参数。

用法: cfg_flash_remove_device

8. cfg_flash_read_register

cfg_flash_read_register, 读 Flash 寄存器值, 无参数。

用法: cfg_flash_read_register

9. cfg_flash_write_register

cfg_flash_write_register, 写 Flash 寄存器值, 1 个参数
-value string 给定写入指定寄存器的值。

用法: cfg_flash_write_register -value 0x1c

4.5.4 针对 Jtag Flash 器件命令

该集合下的 Tcl 命令主要通过 Jtag 接口对板上的 Flash 器件进行下载、删除、回读及校验等操作。由于通过 Jtag 接口操作 Flash 依赖于 FPGA 的端口, 因此执行这些命令时, 需确保 CFG 已经获取到了板上 FPGA 信息, 即进行了扫描 FPGA 链操作。

1. cfg_jtag_flash_scan_device

cfg_jtag_flash_scan_device, 扫描外部 Flash 器件, 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_jtag_flash_scan_device –device_index 0

2. cfg_jtag_flash_assign_file

cfg_jtag_flash_assign_file, 给外部 Flash 配置位流文件 sfc, 2 个参数

–file 配置位流文件的路径

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_jtag_flash_assign_file -file D:/bs/counter.sfc -device_index 0

3. cfg_jtag_flash_erase

cfg_jtag_flash_erase, 擦除外部 Flash 器件, 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_jtag_flash_erase –device_index 0

4. cfg_jtag_flash_program

cfg_jtag_flash_program, 编程外部 Flash 器件, 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_jtag_flash_program –device_index 0

5. cfg_jtag_flash_verify

cfg_jtag_flash_verify, 校验外部 Flash 器件所配位流, 1 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

用法: cfg_jtag_flash_verify –device_index 0

6. cfg_jtag_flash_readback

cfg_jtag_flash_readback, 回读外部 Flash 器件所配位流, 2 个参数

[–device_index] 指定 device 索引, 可选, 默认缺省为 0

–file 回读位流文件的路径

用法: cfg_jtag_flash_readback –device_index 0

4.5.5 针对板上自动化

板上自动化测试中的一个测试用例为一个文件夹, 所包含的文件格式必须符合以下三个格式集的任意一个: {*.sbit, *.lcf, *.vcd}、{*.sbit, *.atp}、{*.sbit, *.avc}。

具体命令如下:

1. cfg_load_dir

cfg_load_dir, 解析测试用例目录生成 Tcl 脚本, 包含 5 个参数

–file (要解析的目录)

[-clock_mode] 设置板上自动化的时钟, 默认为 JTAG 时钟。

[-time] 设置在 program 与 load excite 之间延时时间, 参数类型为大于 0 的整数

[-pkg] 指定封装信息, 默认是基于位文件

[-scale_time] 为生成*.vcd 文件指定时间范围

[-count] 设置测试用例循环配置芯片次数, 参数范围 $0 < \text{count} < 1000$, 参数大于 1000 则自动设置为 1000

[-break] 设置 Tcl 遇到 error 是否往下执行, 参数类型 Boolean_value (TRUE/FALSE), true 表示 Tcl 遇到 error 不继续往下执行

用法: cfg_load_dir -file D:/auto_test –time *integer-value* 表示将对 D:/auto_test 文件夹及其子目录进行解析, 并且 program 与 load excite 之间延时 *integer-value*, 在每个测试用例的文件夹下都会生成相应的 TclScript.tcl 脚本。

2. **cfg_write_all_tcl_file**

cfg_write_all_tcl_file, 收集指定目录和其子目录下的 TclScript.tcl 文件形成一个总的 AllTclScript.tcl 包含一个参数 -file(要解析的目录)

用法: cfg_write_all_tcl_file -file D:/auto_test

3. **cfg_load_excite**

cfg_load_excite, 加载激励 包含 5 个参数,

-device_index(器件的索引 默认为 0)

-efile (vcd 文件)

-lfile (lcf 文件, 或 fdc 文件。 默认为与 vcd 同名文件)

-rfile (report 文件名 默认为 report.txt)

-pfile(新建一个文件夹用来存储)

-clock_mode (时钟模式, 默认为 JTAG 时钟)

[-append] 设置是否保存多次循环配置生成的报告, 参数类型 Boolean_value (TRUE/FALSE), true 表示保存

用法: cfg_load_excite -efile D:/auto_test/test.vcd

4. **cfg_write_all_report_file**

cfg_write_all_report_file, 收集指定目录下的所有 report 文件, 形成一个总的概略性的名为 AllReport.xlsx 的 excel 文件 包含一个参数 -file(要解析的目录)

用法: cfg_write_all_report_file -file D:/auto_test

5. **cfg_write_auto_test**

cfg_write_auto_test, 对指定目录生成一个自动化测试脚本, 包含 5 个参数 -file(要解析的目录)

[-report_detail] 设置板上自动化是否写详细报告, 参数类型 Boolean_value (TRUE/FALSE)

[-clock_mode] 设置板上自动化的时钟, 默认为 JTAG 时钟。

[-time] 设置在 program 与 load excite 之间延时时间, 参数类型为大于 0 的整数

[-server_delay_time] 设置 server 的延时时间, 即延时多长后, server 开始读取数据。参数类型为大于 0 的整数

[**-count**] 设置测试用例循环配置芯片次数，参数范围 $0 < \text{count} < 1000$ ，参数大于 1000 则自动设置为 1000

[**-break**] 设置 Tcl 遇到 error 是否往下执行，参数类型 Boolean_value (TRUE/FALSE)，true 表示 Tcl 遇到 error 不继续往下执行

用法：cfg_write_auto_test –file D:/auto_test –report_detail true –clock_mode 1
–server_delay_time *integer-value* –time *integer-value*

4.5.6 转换文件格式

1. **cfg_gen_sfc**

cfg_gen_sfc，产生关于 flash 的 SFC 文件，5 个参数；

-sbit 指定要转换的 sbit 文件

[**-sfc**] 指定生成的 sfc 文件名，默认值为 sbit 文件同路径下的同名的 sfc 文件

[**-opcode**] 指定操作码，即 Flash 的读写模式。

11(0x0B): "Fast Read, 24-bit address"

59(0x3B): "Dual Output Fast Read, 24-bit address"

107(0x6B): "Quad Output Fast Read, 24-bit address"

12(0x0C): "Fast Read, 32-bit address"

60(0x3C): "Dual Output Fast Read, 32-bit address"

108(0x6C): "Quad Output Fast Read, 32-bit address"

默认的缺省值为 11(0x0B)。

[**-device_name**] 指定兼容的 flash 器件，缺省默认 N25Q256

[**-is_x8**] 转换成 X8 数据流，默认为 false。

[**-sbit_start_address**] 设置位流文件起始位置

[**-fast_mode**] 设置芯片为快速模式还是非快速模式 (pgc 系列没有)。

[**-user_address_list**] 设置用户的所有数据起始地址列表

[**-file_list**] 输入文件列表

用法：cfg_gen_sfc –sbit sbit 文件

2. cfg_gen_chain_file

cfg_gen_chain_file, 生成级联数据流下载文件。包含两个参数,

-infile (指定选择的 sbit 文件, 文件与文件之间用“,”隔开)

[-outfile] (指定输出级联数据流的文件, 可选, 默认缺省为第一个位流文件名后加 _chain.sbit)

[-create_bin_file] 选择输出 sbit 文件的同时输出 bin 文件, 可选

[-reverse_bit_in_a_byte] 对输出的 bin 文件进行每一字节里的 8 位进行翻转, 可选, 仅支持对 bin 文件翻转, 输出的 sbit 的文件不翻转

[-reverse_byte_in_a_word] 对输出的 bin 文件进行每一个字里的 4 字节进行字节层面的翻转, 可选, 仅仅支持对 bin 文件翻转, 输出的 sbit 的文件不翻转

用法: cfg_gen_chain_file -infile E:/pan/counter.sbit, E:/pan/led.sbit -outfile E:/pan/leSS.sbit -create_bin_file -reverse_bit_in_a_byte -reverse_byte_in_a_word

3. cfg_gen_multi_file

cfg_gen_multi_file, 生成多种数据流格式文件, 具体参数的含义请参考第四章 Fabric Configuration 软件说明->第 20 小节器件 FPGA 位流文件转换多种格式数据流。

-infile 指定输入的文件列表, 即要进行转换的文件

-type 指定生成数据流的格式

0:多启动数据流

1:SPI 远程升级数据流

2:BPI 远程升级数据流

3:多功能数据流

4:远程升级数据流

5:主自加载双启动数据流

[-outfile] 指定输出文件的名称, 默认和输入的第一个文件的名称一致

[-sbit_start_address] 设置位流的起始地址

[-insert_irst] 生成多启动数据流时, 是否插入热启动命令

[-steup_index] 指定启动应用位流

[-svf_property] 设置 svf 属性 (16 进制/仅用于主自加载双启动数据流), 16 进制具体位对应含义如下所示。

1 : Enable Background ReConfig. default:Enable

2 : Enable Write Header and Comments default:Enable

3 : Enable SEC option of RUNTEST Command default:Enable

4 : Enable Split SVF into Function and Startup default:Disable

5 : Enable ISC Upgrade In Master Auto Config Mode default:Disable

6 : Enable Reset FPGA default:Enable

7: Enable Program Feature Control Value default:Disable

8: Enable Page Erase Embed Flash default:Disable
9: Enable Verify after Program Embed Flash
10 : Reset FPGA Command default:Enable
11 : Check Done of Status Register Command default:Disable
13 : Entire Erase Embed Flash Command default:Disable
14 : Enable Verify after Erase Embed Flash default:Disable
15 : Enable Verify after Program Embed Flash default:Disable
17 - 16 : Gen SVF Mode. (0:CRAM 1 : Embed Flash 2 : Command) default:1 : Embed Flash
19 - 18 : SVF Frequency. (100KHz(0), 1MHz(1), 5MHz(2), 10MHz(3)) default:1MHz
31 - 20 : SVF Start Page Address(Unit : 256Byte)

[**-jtag_chain**] 手动设置 jtag 链 (16 进制/仅用于主自加载双启动数据流)
) , 具体含义如下。bit 0:7 previous device number
bit 8:15 in the later device number
bit 16:23 previous device command len
bit 24:31 in the later device command len

[**-set_ctl**] 是否设置特征控制位 (仅用于主自加载双启动数据流)
0: 不做设置
1: 使用主自加载模式的特征控制位
2: 设置特征控制位具体值

[**-split**] 是否分离生成主自加载数据流 (仅用于主自加载双启动数据流)
[**-spi_mode**] 设置 spi 模式

[**-flatten**]是否生成 SDR/SIR 填充链信息的 svf.

[**-use_user_freq**]是否使用用户频率

4. **cfg_svf_to_pef**

cfg_svf_to_pef, 把 SVF 按照一定的格式进行压缩生成 PEF 文件。

-infile 指定输入的 SVF 文件名称, 即要进行转换的文件。建议输入绝对路径。
[-outfile] 指定输出文件的名称, 默认和输入的文件的名称一致
[-compress] 指定是否压缩, 默认压缩
[-frequency] 指定重写 SVF 文件的频率, 如果没有提供参数默认与原 SVF 文件一致或者为 1 MHz
[-vendor] 指定 SVF 文件厂商, 默认为 JTAG 标准

5. **cfg_one_step_create_svf**

cfg_one_step_create_svf, 生成可配置 FPGA 和嵌入式 Flash 的 svf 文件, 具体参数的含义请参考第四章 Fabric Configuration 软件说明->第 19 小节器件设置操作器件属性的 SVF 属性。

[**-device_index**] 指定操作器件的索引, 可选, 默认第一个器件

[-svf_file_name] 指定生成 SVF 的文件名称，可选，默认与位流文件名称一致
[-svf_property] 设置 svf 属性（16 进制），16 进制具体位对应含义如下所示。

1 : Enable Background ReConfig. default:Enable
2 : Enable Write Header and Comments default:Enable
3 : Enable SEC option of RUNTEST Command default:Enable
4 : Enable Split SVF into Function and Startup default:Disable
5 : Enable ISC Upgrade In Master Auto Config Mode default:Disable
6 : Enable Reset FPGA default:Enable
7: Enable Program Feature Control Value default:Disable
8: Enable Page Erase Embed Flash default:Disable
9: Enable Verify after Program Embed Flash
10 : Reset FPGA Command default:Enable
11 : Check Done of Status Register Command default:Disable
13 : Entire Erase Embed Flash Command default:Disable
14 : Enable Verify after Erase Embed Flash default:Disable
15 : Enable Verify after Program Embed Flash default:Disable
17 – 16 : Gen SVF Mode. (0:CRAM 1 : Embed Flash 2 : Command) default:1 : Embed Flash
19 – 18 : SVF Frequency. (100KHz(0), 1MHz(1), 5MHz(2), 10MHz(3)) default:1MHz
31 – 20 : SVF Start Page Address(Unit : 256Byte)

[-jtag_chain] 手动设置 jtag 链（16 进制），具体含义如下。

bit 0:7 previous device number
bit 8:15 in the later device number
bit 16:23 previous device command len
bit 24:31 in the later device command len

[-ctl_value] 此参数用于设置特征控制位。注：若要设置特征控制位的值则要先把 svf_property 的第 8 位设置为 1.

[-not_create_pef] 设置是否生成 pef 文件，默认为 false

[-sbit] 手动选择输入位流文件

[-flatten] 是否生成 SDR/SIR 填充链信息的 svf.

6. cfg_execute_svf

cfg_execute_svf, 执行 SVF 文件或者 PEF 文件

-svf_file_name 指定输入的 SVF 或者 PEF 文件名称。

[-device_index] 设备位置

7. cfg_create_chain_svf

cfg_create_chain_svf 合并 svf 文件

-infile 输入 svf 文件路径，以空格区分，至少输入两个

[-outfile string] 输出文件路径

8. cfg_pef_to_svf

cfg_pef_to_svf 将 pef 文件转换为 svf 文件

-infile 输入 pef 文件路径

[-outfile] 输出文件路径

9. cfg_switch_flatten_svf

实现普通 svf 和链式 svf 相互转换

-file 输入 svf 文件路径

[-outfile] 输出文件路径

[-mode] 转换模式，0：普通 svf 转换成链式 svf

1：链式 svf 转换成普通 svf

10. cfg_gen_remote_backup_sfc

实现 remote 备份位流转换。(仅支持 pgc7kd)

-file 输入 sbit 文件路径

4.6 终端模式介绍

若要在非图形界面系统下使用 Fabric Configuration 功能，可以使用 Fabric Configuration 的终端版本。通过在 PDS 软件的安装目录下的 bin 目录双击“cdt_cfg_shell.exe”启动该软件，

启动后界面如图 4-79 所示。

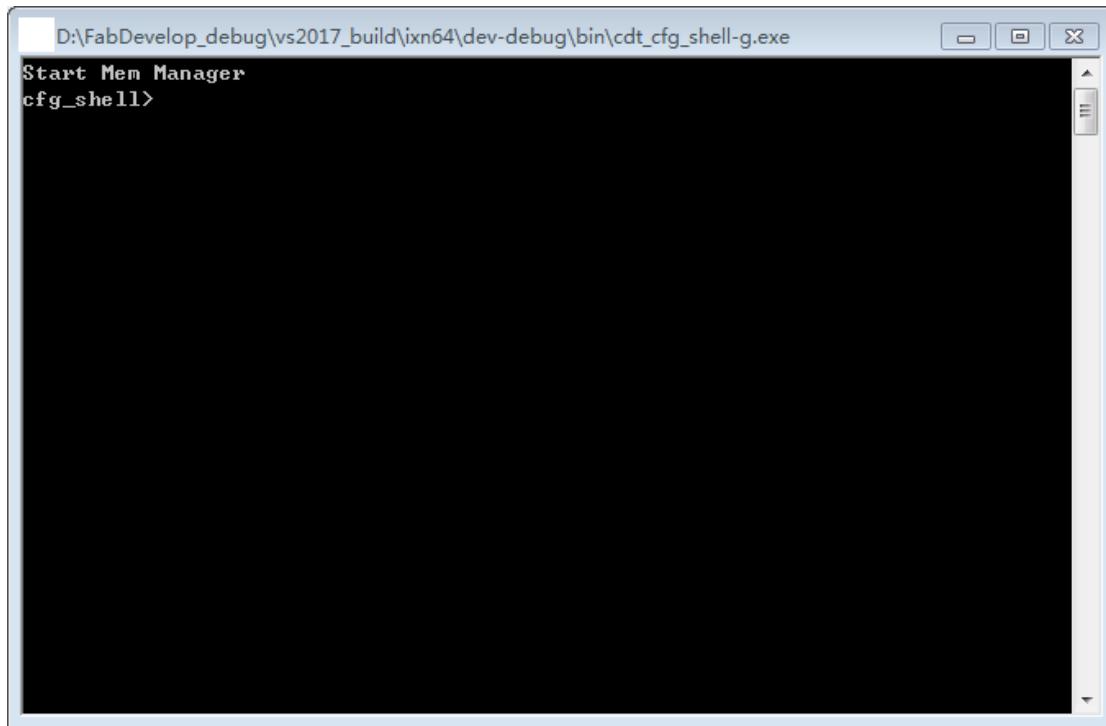


图 4-79 cfg_shel 显示

终端模式是通过直接输入 Tcl 命令或 source tcl 文件来实现相关功能的, 具体 Tcl 命令的功能可见章节 4.5 描述。退出软件时可输入 exit 和 cfg_close 指令。

5 附录

5.1 指令寄存器各位表示的功能

5.1.1 Titan 系列指令寄存器说明

位	名称	描述
[9]	保留	值为0
[8]	init_b	
[7]	efuse_prog_over	指示eFuse编程结束
[6]	wakedown_over	指示唤醒关断完成
[5]	wakeup_over	指示唤醒完成
[4]	init	
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]		为固定值01，实现对扫描链路完整性的检测和故障定位。

表 9 Titan 系列指令寄存器说明

5.1.2 Logos 系列指令寄存器说明

位	名称	描述
[9]	保留	值为0
[8]	efuse_prog_over	指示eFuse编程结束
[7]	wakedown_over	指示唤醒关断完成
[6]	wakeup_over	指示唤醒完成
[5]	init_b	
[4]	init	
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]		为固定值01，实现对扫描链路完整性的检测和故障定位。

表 10 Logos 系列指令寄存器说明

5.1.3 Logos2 系列指令寄存器说明

位	名称	描述
[9]	保留	值为0
[8]	efuse_prog_over	指示eFuse编程结束
[7]	wakedown_over	指示唤醒关断完成
[6]	wakeup_over	指示唤醒完成
[5]	init_n	
[4]	init	init_complete
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]		为固定值01，实现对扫描链路完整性的检测和故障定位。

表 11 Logos2 系列指令寄存器说明

5.1.4 Compact 系列指令寄存器说明

位	名称	描述
[9]	保留	值为0
[8]	busy	指示嵌入式FLASH处于忙碌状态
[7]	wakedown_over	指示唤醒关断完成
[6]	wakeup_over	指示唤醒完成
[5]	init_n	
[4]	init	
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]		为固定值01，实现对扫描链路完整性的检测和故障定位。

表 12 Compact 系列指令寄存器说明

5.1.5 Titan2 系列指令寄存器说明

位	名称	描述
[9]	保留	值为0
[8]	efuse_prog_over	指示eFuse编程结束
[7]	wakedown_over	指示唤醒关断完成
[6]	wakeup_over	指示唤醒完成
[5]	init_n	
[4]	init	init_complete
[3]	isc_enabled	
[2]	isc_done	
[1]		
[0]		为固定值01，实现对扫描链路完整性的检测和故障定位。

表 13 Titan2 系列指令寄存器说明

5.2 状态寄存器各位表示的功能

5.2.1 Titan 系列状态寄存器说明

位	名称	描述
[31:22]		保留
[21]	flg_x8_pal	并行x8模式
[20]	flg_x16_pal	并行x16模式
[19]	flg_x32_pal	并行x32模式
[18:16]	m[2:0]	模式选择
[15]	key_lock	KEY读写禁止
[14]	init_b	init_b
[13]	init_complete	配置完成和错误指示
[12]	done_i	外部done
[11]	wakeup_over	唤醒结束
[10]	wakedown_over	关断唤醒结束
[9]	glogen	glogen
[8]	done	done
[7]	gwen	gwen
[6]	grsn	grsn
[5]	gouten	gouten
[4]	overflow	配置溢出指示
[3]	flg_abort_ipal	内部并行abort指示
[2]	flg_abort_pal	并行abort指示
[1]	crc_err	CRC检测结果 0: CRC正确 1: CRC错误
[0]	id_err	ID检测结果 0: 正确 1: 错误

表 14 Titan 系列状态寄存器说明

5.2.2 Logos 系列状态寄存器说明

位	名称	描述
[31:30]		保留
[29]	over_temp	过温度标志
[28]	flg_x32	从并行模式32位数据位宽指示
[27]	flg_x16	从并行模式16位数据位宽指示
[26]	flg_x8	从并行模式8位数据位宽指示
[25:24]	ipal_m[1:0]	内部并行接口数据位宽选择
[23]	key_lock	密钥锁定标志
[22]	fallback	回退指示标志
[21]	dci_match	DCI匹配标志
[20]	pll_lock	PLL锁定标志
[19]	gwen	全局写使能
[18]	grsn	全局寄存器置位复位
[17]	gouten	全局IO输出使能
[16]	vddt_n	LDO开关使能
[15]	glogen_fb	全局逻辑使能反馈
[14]	glogen	全局逻辑使能

[13]	done_i	CFG_DONE管脚输入
[12]	done	器件唤醒成功标志
[11]	init_b	init_flag_n管脚输入
[10]	init_complete	初始化完成和配置错误指示
[9:7]	m[2:0]	模式选择
[6]	wakedown_over	唤醒关断结束
[5]	wakeup_over	唤醒结束
[4]	overflow	FIFO溢出
[3]	timeout	看门狗超时
[2]	rbcrc_err	回读CRC检测结果 0: CRC正确 1: CRC错误
[1]	crc_err	CRC检测结果 0: CRC正确 1: CRC错误
[0]	id_err	ID检测结果 0: 正确 1: 错误

表 15 Logos 系列状态寄存器说明

5.2.3 Logos2 系列状态寄存器说明

位	名称	描述
[31:30]		保留
[29]	prcfg_over	局部重配完成标志
[28]	prcfg_err	局部重配错误标志
[27]	over_temp	过温度标志
[26]	f1g_x32	从并行模式32位数据位宽指示
[25]	f1g_x16	从并行模式16位数据位宽指示
[24]	f1g_x8	从并行模式8位数据位宽指示
[23:22]	ipal_m[1:0]	内部并行接口数据位宽选择
[21]	fallback	回退指示标志
[20]	dci_match	DCI匹配标志
[19]	pll_lock	PLL锁定标志
[18]	gwen	全局写使能
[17]	grsn	全局寄存器置位复位
[16]	gouten	全局IO输出使能
[15]	glogen_fb	全局逻辑使能反馈
[14]	glogen	全局逻辑使能
[13]	done_i	CFG_DONE管脚输入
[12]	done	器件唤醒成功标志
[11]	init_n	INIT_N管脚输入
[10]	init_complete	初始化完成和配置错误指示
[9:7]	m[2:0]	模式选择
[6]	wakedown_over	唤醒关断结束
[5]	wakeup_over	唤醒结束
[4]	timeout	看门狗超时
[3]	rbcrc_err	回读CRC检测结果 0: CRC正确 1: CRC错误

[2]	aut_err	认证结果 0: 认证通过 1: 认证失败
[1]	crc_err	CRC检测结果 0: CRC正确 1: CRC错误
[0]	id_err	ID检测结果 0: 正确 1: 错误

表 16 Logos2 系列状态寄存器说明

5.2.4 Compact 系列状态寄存器说明

位	名称	描述
[31:30]	保留	
[29]	persist_mspi	用户模式下主SPI接口使能
[28]	persist_si2c	用户模式下从I ₂ C接口使能
[27]	persist_sspi	用户模式下从SPI接口使能
[26]	persist_jtag	用户模式下JTAG接口使能
[25]	persist_done	用户模式下DONE管脚使能
[24]	persist_init	用户模式下INIT_N管脚使能
[23]	persist_rstn	用户模式下RST_N管脚使能
[22]	pass_cal	嵌入式FLASH校准成功标志 4K/7K器件此位保留
[21]	busy	嵌入式FLASH忙碌标志
[20]	lock	嵌入式FLASH锁定标志 嵌入式FLASH锁定后，不能编程和读主自加载数据流和用户FLASH，只能擦除主自加载数据流和用户FLASH。
[19]	wake	嵌入式FLASH唤醒标志
[18]	sleep	嵌入式FLASH休眠标志
[17]	fallback	回退指示标志
[16]	pll_lock	PLL锁定标志
[15]	gwen	全局写使能
[14]	grsn	全局寄存器置位复位
[13]	gouten	全局IO输出使能
[12]	保留	
[11]	glogen_fb	全局逻辑使能反馈
[10]	glogen	全局逻辑使能
[9]	done_i	DONE管脚输入
[8]	done	器件唤醒成功标志
[7]	init_n	INIT_N管脚输入
[6]	init_complete	初始化完成和配置错误指示
[5]	wakedown_over	唤醒关断结束
[4]	wakeup_over	唤醒结束
[3]	timeout	看门狗超时
[2]	rbcrc_err	回读CRC检测结果

		0: CRC正确 1: CRC错误
[1]	crc_err	CRC检测结果 0: CRC正确 1: CRC错误
[0]	id_err	ID检测结果 0: 正确 1: 错误

表 17 Compact 系列状态寄存器说明

5.2.5 Titan2 系列状态寄存器说明

位	名称	描述
[31:30]		保留
[29]	prcfg_over	局部重配完成标志
[28]	prcfg_err	局部重配错误标志
[27]	over_temp	过温度标志
[26]	f1g_x32	从并行模式32位数据位宽指示
[25]	f1g_x16	从并行模式16位数据位宽指示
[24]	f1g_x8	从并行模式8位数据位宽指示
[23:22]	ipal_m[1:0]	内部并行接口数据位宽选择
[21]	fallback	回退指示标志
[20]	dci_match	DCI匹配标志
[19]	pll_lock	PLL锁定标志
[18]	gwen	全局写使能
[17]	grsn	全局寄存器置位复位
[16]	gouten	全局IO输出使能
[15]	glogen_fb	全局逻辑使能反馈
[14]	glogen	全局逻辑使能
[13]	done_i	CFG_DONE管脚输入
[12]	done	器件唤醒成功标志
[11]	init_n	INIT_N管脚输入
[10]	init_complete	初始化完成和配置错误指示
[9:7]	m[2:0]	模式选择
[6]	wakedown_over	唤醒关断结束
[5]	wakeup_over	唤醒结束
[4]	timeout	看门狗超时
[3]	rbcrc_err	回读CRC检测结果 0: CRC正确 1: CRC错误
[2]	aut_err	认证结果 0: 认证通过 1: 认证失败
[1]	crc_err	CRC检测结果 0: CRC正确 1: CRC错误
[0]	id_err	ID检测结果 0: 正确 1: 错误

表 18 Titan2 系列状态寄存器说明

5.3 Compact 系列的特征控制位说明

5.3.1 PGC 1K、2K 系列支持的特征控制位

位	名称	默认值	描述
[0]	Enable Master Auto Mode In Configuration	0:默认不使能 0:不使能 1:使能	是否使能主自加载模式 0:不使能 1:使能
[1]	Disable RST_N Pin In Configuration Mode	1:默认禁用 0:不禁用 1:禁用	是否禁用配置模式下 RST_N 管脚 0:不禁用 1:禁用
[2]	Enable INIT Pin In Configuration Mode	0:默认不使能 0:不使能 1:使能	是否使能配置模式下 INIT 管脚 0:不使能 1:使能
[3]	Enable Done Pin In Configuration Mode	0:默认不使能 0:不使能 1:使能	是否使能配置模式下 DONE 管脚 0:不使能 1:使能
[4]	Enable Master SPI Mode In Configuration	0:默认不使能 0:不使能 1:使能	是否使能主 SPI 模式 0:不使能 1:使能
[5]	Disable JTAG Mode In Configuration Mode	0:默认不禁用 0:不禁用 1:禁用	是否禁用配置模式下从 Jtag 接口 0:不禁用 1:禁用
[6]	Disable Slave SPI Mode In Configuration	0:默认不禁用 0:不禁用 1:禁用	是否禁用配置模式下从 SPI 接口 0:不禁用 1:禁用
[7]	Disable Slave I2C Mode In Configuration	0:默认不禁用 0:不禁用 1:禁用	是否禁用配置模式下从 I2C 接口 0:不禁用 1:禁用
[8]	Disable Slave Parallel Mode In Configuration	1:默认禁用 0:不禁用 1:禁用	是否禁用配置模式下从并接口 0:不禁用 1:禁用
[9]	Enable Double Start Up	0:默认不使能 0:不使能 1:使能	是否使能双启动模式 0:不使能 1:使能
[10]	Enable Fast Master SPI Mode	0:默认不使能 0:不使能 1:使能	是否使能主 SPI 模式 0:不使能 1:使能
[11]	Disable Persist RST_N Pin In User Mode	1:默认不禁用 0:不禁用 1:禁用	是否禁用用户模式下 RST_N 管脚 0:不禁用 1:禁用
[12]	Enable Persist INIT Pin In User Mode	0:默认不使能 0:不使能 1:使能	是否使能用户模式下 INIT 管脚 0:不使能 1:使能
[13]	Enable Persist Done Pin In User Mode	0:默认不使能 0:不使能 1:使能	是否使能用户模式下 DONE 管脚 0:不使能 1:使能
[14]	Disable Persist JTAG Mode In User Mode	0:默认不禁用 0:不禁用 1:禁用	是否禁用用户模式下 JTAG 接口 0:不禁用 1:禁用
[15]	Disable Persist Slave SPI Mode In User Mode	0:默认不禁用 0:不禁用 1:禁用	是否禁用用户模式下从 SPI 接口 0:不禁用 1:禁用
[16]	Disable Persist Slave I2C Mode In User Mode	0:默认不禁用 0:不禁用 1:禁用	是否禁用用户模式下从 I2C 接口 0:不禁用 1:禁用
[17]	Disable Persist Slave Parallel Mode In User Mode	1:默认禁用 0:不禁用 1:禁用	是否禁用用户模式下从并接口 0:不禁用 1:禁用
[27:18]	I2C Address	默认为 2 'b0000000000	I2C 设备地址
[28]	Enable Persist Master SPI Mode In User Mode	0:默认不使能 0:不使能 1:使能	是否使能用户模式下主 SPI 接口 0:不使能 1:使能

[29]	Enable Slowly Read Mode	0:默认不使能	慢读使能 0: 主自加载, 读嵌入式 FLASH 指令, 读特征控制位指令, 读 trim 指令时, 嵌入式 FLASH 接口采用正常读模式 1: 所有嵌入式 FLASH 读操作均采用慢读模式
[31:30]	保留		
	注: Compact 1K、2K 系列特征控制位的默认值为: 0x00020902		

表 19 PGC 1K、2K 系列支持的特征控制位

5.3.2 PGC 4K、7K、10K 系列支持的特征控制位

位	名称	默认值	描述
[0]	Enable Master Auto Mode In Configuration	0:默认不使能	是否使能主自加载模式 0:不使能 1:使能
[1]	Disable RST_N Pin In Configuration Mode	1:默认禁用	是否禁用配置模式下 RST_N 管脚 0:不禁用 1:禁用
[2]	Enable INIT Pin In Configuration Mode	0:默认不使能	是否使能配置模式下 INIT 管脚 0:不使能 1:使能
[3]	Enable Done Pin In Configuration Mode	0:默认不使能	是否使能配置模式下 DONE 管脚 0:不使能 1:使能
[4]	Enable Master SPI Mode In Configuration	0:默认不使能	是否使能主 SPI 模式 0:不使能 1:使能
[5]	Disable JTAG Mode In Configuration Mode	0:默认不禁用	是否禁用配置模式下从 Jtag 接口 0:不禁用 1:禁用
[6]	Disable Slave SPI Mode In Configuration	0:默认不禁用	是否禁用配置模式下从 SPI 接口 0:不禁用 1:禁用
[7]	Disable Slave I2C Mode In Configuration	0:默认不禁用	是否禁用配置模式下从 I2C 接口 0:不禁用 1:禁用
[8]	Disable Slave Parallel Mode In Configuration	1:默认禁用	是否禁用配置模式下从并接口 0:不禁用 1:禁用
[9]	Enable Double Start Up	0:默认不使能	是否使能双启动模式 0:不使能 1:使能
[10]	Enable Fast Master SPI Mode	0:默认不使能	是否使能主 SPI 模式 0:不使能 1:使能
[11]	Disable Persist RST_N Pin In User Mode	1:默认不禁用	是否禁用用户模式下 RST_N 管脚 0:禁用 1:不禁用
[12]	Enable Persist INIT Pin In User Mode	0:默认不使能	是否使能用户模式下 INIT 管脚 0:不使能 1:使能
[13]	Enable Persist Done Pin In User Mode	0:默认不使能	是否使能用户模式下 DONE 管脚 0:不使能 1:使能
[14]	Disable Persist JTAG Mode In User Mode	0:默认不禁用	是否禁用用户模式下 JTAG 接口 0:不禁用 1:禁用

[15]	Disable Persist Slave SPI Mode In User Mode	0:默认不禁用 0:不禁用 1:禁用	是否禁用用户模式下从 SPI 接口 0:不禁用 1:禁用
[16]	Disable Persist Slave I2C Mode In User Mode	0:默认不禁用	是否禁用用户模式下从 I2C 接口 0:不禁用 1:禁用
[17]	Disable Persist Slave Parallel Mode In User Mode	1:默认禁用	是否禁用用户模式下从并接口 0:不禁用 1:禁用
[27:18]	I2C Address	默认为 2 ‘b0000000000	I2C 设备地址
[28]	Enable Persist Master SPI Mode In User Mode	0:默认不使能	是否使能用户模式下主 SPI 接口 0:不使能 1:使能
[29]	Enable Slowly Read Mode	0:默认不使能	慢读使能 0: 主自加载, 读嵌入式 FLASH 指令, 读特征控制位指令, 读 trim 指令时, 嵌入式 FLASH 接口采用正常读模式 1: 所有嵌入式 FLASH 读操作均采用慢读模式
[30]	msdboot_en	0:默认不使能	主自加载双启动使能
[31]	保留		
	注: Compact 4K、7K、10K 系列特征控制位的默认值为: 0x00020902		

表 20 PGC 4K、7K、10K 系列支持的特征控制位

5.4 目前支持的 Flash 器件及其匹配的读写模式

Flash 器件名	支持的读写模式
M25P16	SPI X1, 24-bit address
M25P32	SPI X1, 24-bit address
M25P64	SPI X1, 24-bit address
M25P128	SPI X1, 24-bit address
MT25Q512	SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X4, 32-bit address
N25Q32	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q64	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q128	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address
N25Q256	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
N25Q512	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address SPI X1, 32-bit address; SPI X1, 32-bit address; SPI X1, 32-bit address
GD25Q80C	SPI X1, 24-bit address, SPI X2, 24-bit address, SPI X4, 24-bit address
GD25Q32C	SPI X1, 24-bit address; SPI X1, 24-bit address; SPI X1, 24-bit address

GD25Q64C	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
GD25Q128C	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
GD25Q256D	SPI X1, 24-bit address; SPI X1, 32-bit address;	SPI X1, 24-bit address; SPI X1, 32-bit address;	SPI X1, 24-bit address; SPI X1, 32-bit address
GD25Q512D	SPI X1, 32-bit address; SPI X1, 24-bit address;	SPI X2, 32-bit address; SPI X2, 24-bit address;	SPI X4, 32-bit address SPI X4, 24-bit address
GD25B512ME	SPI X1, 32-bit address; SPI X1, 24-bit address;	SPI X4, 32-bit address; SPI X4, 24-bit address	
MD25Q80C	SPI X1, 24-bit address,	SPI X2, 24-bit address,	SPI X4, 24-bit address
W25Q40CL	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q80	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q16	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q32	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q64	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q128	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
W25Q256	SPI X1, 24-bit address; SPI X1, 32-bit address;	SPI X1, 24-bit address; SPI X1, 32-bit address;	SPI X1, 24-bit address; SPI X1, 32-bit address
W25Q512	SPI X1, 32-bit address, SPI X1, 24-bit address,	SPI X2, 32-bit address, SPI X2, 24-bit address,	SPI X4, 32-bit address SPI X4, 24-bit address
W25M512	SPI X1, 32-bit address, SPI X1, 24-bit address,	SPI X2, 32-bit address, SPI X2, 24-bit address,	SPI X4, 32-bit address SPI X4, 24-bit address
XM25QH16B	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
XM25QH32B	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
XM25QH64A	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
XM25QH128A	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address
XM25QH256B	SPI X1, 32-bit address, SPI X2, 24-bit address	SPI X2, 32-bit address;	SPI X1, 24-bit address
SM25QH256M	SPI X1, 32-bit address; SPI X2, 24-bit address	SPI X2, 32-bit address;	SPI X1, 24-bit address
S25FL64	SPI X1, 24-bit address;	SPI X2, 24-bit address;	SPI X4, 24-bit address
S25FL128	SPI X1, 24-bit address;	SPI X2, 24-bit address;	SPI X4, 24-bit address
S25FL256L	SPI X1, 32-bit address; SPI X1, 24-bit address;	SPI X2, 32-bit address; SPI X2, 24-bit address;	SPI X4, 32-bit address SPI X4, 24-bit address
S25FL256S	SPI X1, 32-bit address; SPI X1, 24-bit address;	SPI X2, 32-bit address; SPI X2, 24-bit address;	SPI X4, 32-bit address SPI X4, 24-bit address
S25FL512	SPI X1, 32-bit address; SPI X1, 24-bit address;	SPI X2, 32-bit address; SPI X2, 24-bit address;	SPI X4, 32-bit address SPI X4, 24-bit address
ZB25VQ16	SPI X1, 24-bit address;	SPI X1, 24-bit address;	SPI X1, 24-bit address

ZB25VQ32	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L16	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L32	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L64	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L128	SPI X1, 24-bit address; SPI X2, 24-bit address
MX25L256	SPI X1, 32-bit address; SPI X2, 32-bit address
MX25L512	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25LP16D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP32D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP64D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25LP128D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25LP256D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25LP512D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP16D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP32D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP64D	SPI X1, 24-bit address; SPI X2, 24-bit address
IS25WP128D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP256D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
IS25WP512D	SPI X1, 32-bit address; SPI X2, 32-bit address; SPI X1, 24-bit address SPI X2, 24-bit address
EN25Q80C	SPI X1, 24-bit address; SPI X2, 24-bit address
P25Q80H	SPI X1, 24-bit address; SPI X2, 24-bit address; SPI X4, 24-bit address
AT45DB081E	SPI X1, 24-bit address; SPI X2, 24-bit address
FH25VQ80D	SPI X1, 24-bit address; SPI X2, 24-bit address

表 21 目前支持的 Flash 器件及其匹配的读写模式

6 免责声明

版权声明

本文档版权归公司所有，并保留一切权利。未经书面许可，任何公司和个人不得将此文档中的任何部分公开、转载或以其他方式披露、散发给第三方。否则，公司必将追究其法律责任。

免责声明

1、本文档仅提供阶段性信息，所含内容可根据产品的实际情况随时更新，恕不另行通知。如因本文档使用不当造成的直接或间接损失，本公司不承担 任何法律责任。

2、本文档按现状提供，不负任何担保责任，包括对适销性、适用于特定用途或非侵权性的任何担保，和任何提案、规格或样品在他处提到的任何担保。本文档在此未以禁止反言或其他方式授予任何知识产权使用许可，不管是明示许可还是暗示许可。

3、公司保留任何时候在不事先声明的情况下对公司系列产品相关文档的修改权利。