



专注于高端实时控制芯片设计

GS32-DSP 系列 GS32Prog 用户手册

2025 年 08 月 22 日

格见构知（上海）半导体有限公司 | 深圳格见构知半导体有限公司（简称“格见半导体”）是一家专注于高端实时控制芯片设计公司，在芯片产品定义、设计研发、量产导入、销售运营等领域都具备丰富经验。

GS32-DSP 是格见半导体自主研发的实时控制微处理器系列产品。公司致力于为数字能源、数字电源、工业自动化、智能汽车、机器人、高端家电等领域提供芯片解决方案。

更多产品资料：sales@gejian-semi.com; support@gejian-semi.com

更多详情访问：www.gejian-semi.com



更多官方资讯

目录

1 GS32Prog 简介	2
1.1 说明	2
1.2 目录说明	2
2 安装与启动	3
2.1 安装与启动	3
2.2 界面	3
3 Openocd 页签	4
3.1 工程选择和工程保存	4
3.2 连接选项	4
3.3 连接测试	4
3.4 烧录	4
3.5 读取烧录程序	5
3.6 读取芯片 hex	6
3.7 擦除指定地址段	7
3.8 全片擦除	7
3.9 连接以及烧录记录	7
4 BOOTROM 页签	8
4.1 说明	8
5 jlink 页签	9
5.1 说明	9
6 Hex Modification 页签	10
6.1 说明	10
6.2 插入 CRC 值	10
6.3 To 32byte	11
6.4 Re-write	11
6.5 Hex2Bin	12
7 Bootloader 页签	13
7.1 说明	13
7.2 镜像头部的添加	13
7.3 镜像文件的烧录	14
8 OptionBytes 页签	15
8.1 说明	15
8.2 RDP	15
9 Image Tools	16
9.1 Merge Hex	16
9.2 Format Conversion	17
9.3 Show Hex	17
10 附录	18
10.1 命令方式执行烧录、OptionBytes、CRC 验证	18
10.2 命令方式执行 hex 合并	19
10.3 命令方式获取 FTDI 连接项	19
11 修订历史	20

1 GS32Prog 简介

1.1 说明

GS32Prog 为用户提供离线烧录、映射文件修改处理等相关功能；

1.2 目录说明

文件	说明
openocd_bin	gs-openocd 存放位置
scripts	烧录算法存放位置
tools	USB 工具存放位置
GS32Prog.exe	启动程序

2 安装与启动

2.1 安装与启动

GS32Prog 为绿色软件，只需要解压之后，运行根目录下的 GS32Prog.exe 即可启动。



图 2.1-1 GS32Prog 安装目录的内容

2.2 界面

GS32Prog 分为 6 个页签：Openocd、BOOTROM、Jlink、Hex Modification、Bootloader、OptionBytes



图 2.2-1 GS32Prog 界面

3 Openocd 页签

3.1 工程选择和工程保存

工程按钮点击后可以进行工程的保存和选择，便于下次的烧录等快速进行

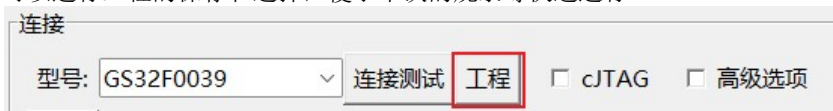


图 3.1-1 工程按钮

弹出工程管理界面后，可以对新建工程、保存工程、加载工程、加载外部工程

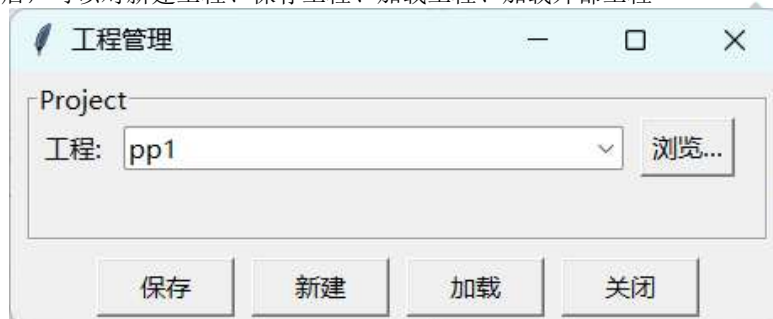


图 3.1-2 工程管理界面

3.2 连接选项

选项包括 cJTAG、JTAG 频率(KHZ)、芯片主频(HZ)、需要连接的端口，其中 JTAG 频率(KHZ)和芯片主频(HZ)如果不选择，则使用 cfg 中的默认设置

端口按钮点击后如果显示相关的选项，则说明满足端口选择的条件，可以指定需要连接到端口

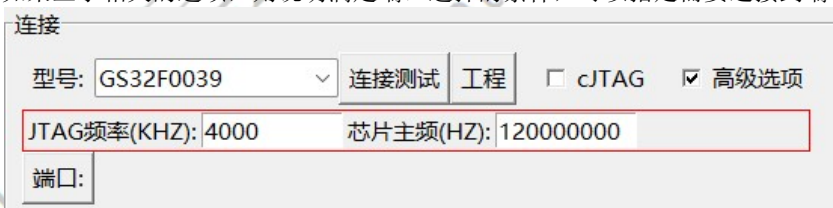


图 3.2-1 连接选项

3.3 连接测试

连接测试用户测试当前的调试器与硬件连接是否正常，需要按以下步骤进行选择。

预备阶段先选择适合的连接选项

- (1) 选择适合的型号
- (2) 点击连接测试

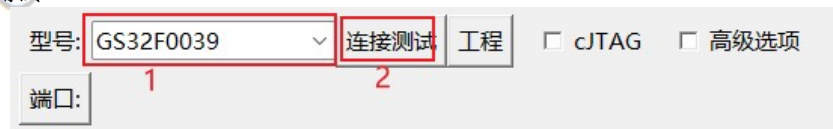


图 3.3-1 连接选择

3.4 烧录

可以选择 bin、hex、elf 进行烧录，多 image 可以同时烧录。

步骤如下：



图 3.4-1 烧录步骤

注意：如果选择 bin 进行烧录，弹出起始地址输入界面，输入需要的起始地址后点击确定即可

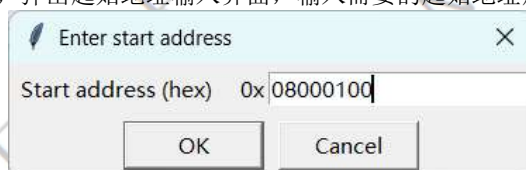


图 3.4-2 bin 起始地址选择输入

3.5 读取烧录程序

单击读取文件内容，会执行烧录程序文件的读取，然后展示 hex 信息。



图 3.5-1 单击读取文件内容

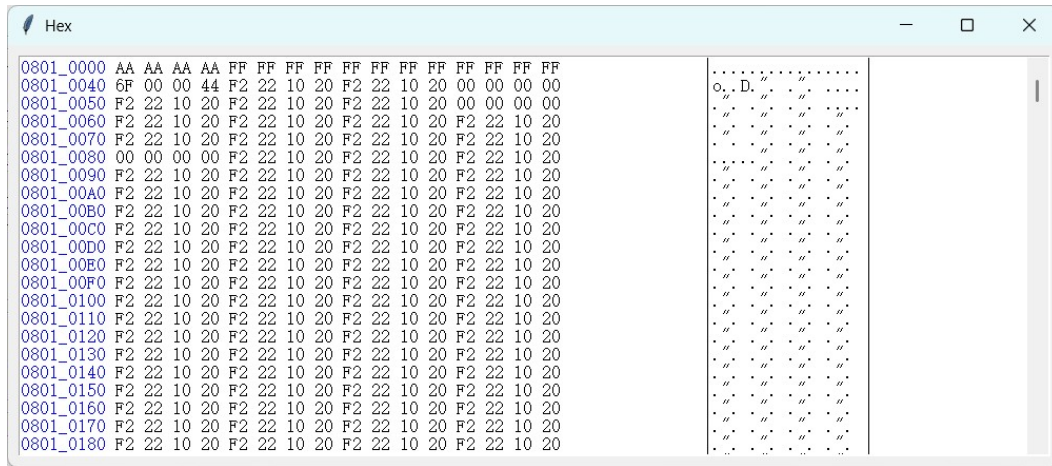


图 3.5-2 烧录文件 hex 显示

3.6 读取芯片 hex

设置的起始位置和需要读取的大小，执行读取任务。



图 3.5-1 设置和执行读取芯片 hex

显示 hex 信息，以及确定是否需要保存

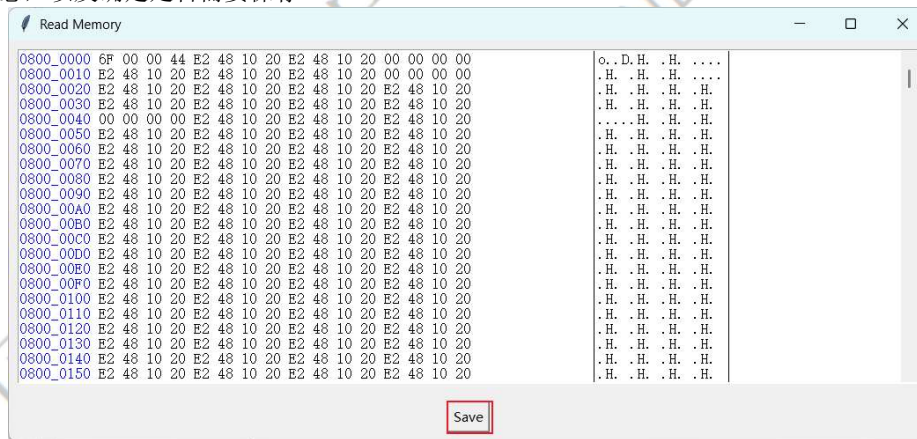


图 3.6-2 显示 hex

保存时可以选择格式：hex、bin

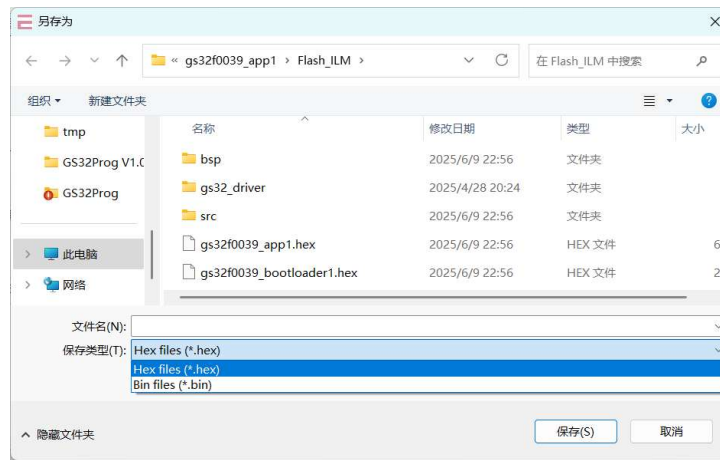


图 3.6-3 保存界面

3.7 擦除指定地址段

设置的起始位置和需要擦除的范围，擦除芯片指定地址段范围的内容。

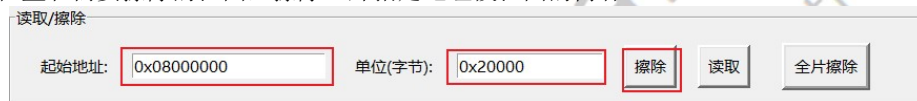


图 3.7-1 擦除设置和执行

3.8 全片擦除

执行全片擦除任务。

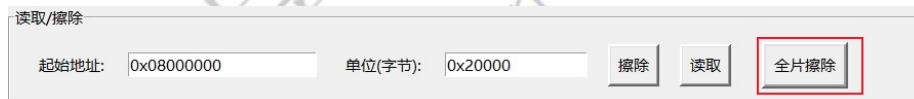


图 3.8-1 执行全片擦除

3.9 连接以及烧录记录

记录使用过程中的成功与失败，且有相关的 log 日志进行分析，日志位置当前程序的 Logs 文件夹中

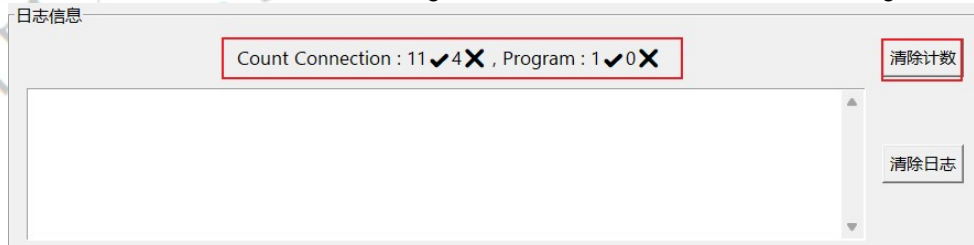


图 3.9-1 执行全片擦除

4 BOOTROM 页签

4.1 说明

此功能要求下位机带有 bootrom，控制 APP 图像的烧写，该功能通过 UART 串口实现，速率为 21k/s。用户可以烧录多个 APP 图像在互不冲突的起始地址。点击“Add Row”，增加行，若确定此次烧录，勾选左边的小框，若需要删除行，点击每行对应的“Delete”。

该功能支持 Hex 文件和 bin 文件，加载若为 Hex 文件会自动转换成 bin 文件，保存在自动生成的“output”文件夹中。

加载好对应的起始地址和图像路径后选择想要的串口，点击“Write”开始烧录。（在此之前，确认负责接收图像的 bootrom 已开始运行）。每一行最后的进度条显示该图像烧录的进度。

可以选择合适的频率以及端口进行烧录。

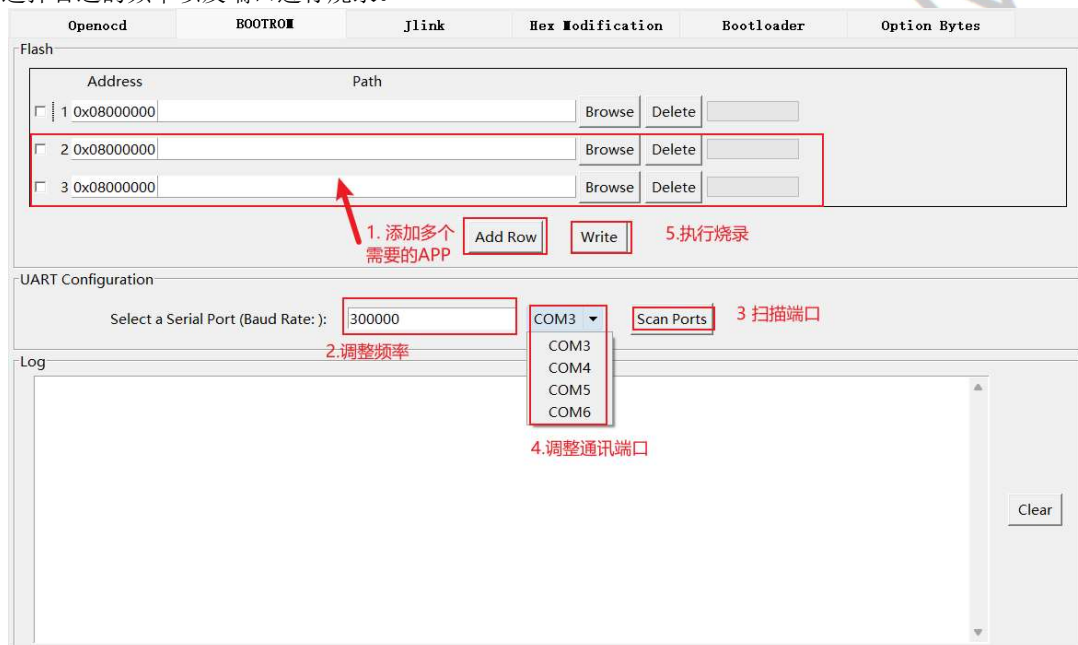


图 4.1-1BOOTROOM 界面

5 jlink 页签

5.1 说明

或者改为该功能需要提前安装好 Jlink 在 “C:\Program Files (x86)\SEGGER\JLink\jlink.exe”。如图 10，确定好 MCU 类型、接口、速度，以及命令。命令有烧录、擦除、读取。

加载好对应的图像路径、地址及尺寸后，点击 “Start”。在 Log 框会返回 Jlink 的输出信息。若命令为 “read”，读取出的内容会保存在自动生成的 “output” 文件夹中，为 txt 格式。

该功能目前不完善：开启 Jlink 后只能实现单个命令，只有关闭 Jlink 后才能读取 Jlink 的输出。导致执行一个命令后要关闭 Jlink，以观察返回结果成功与否

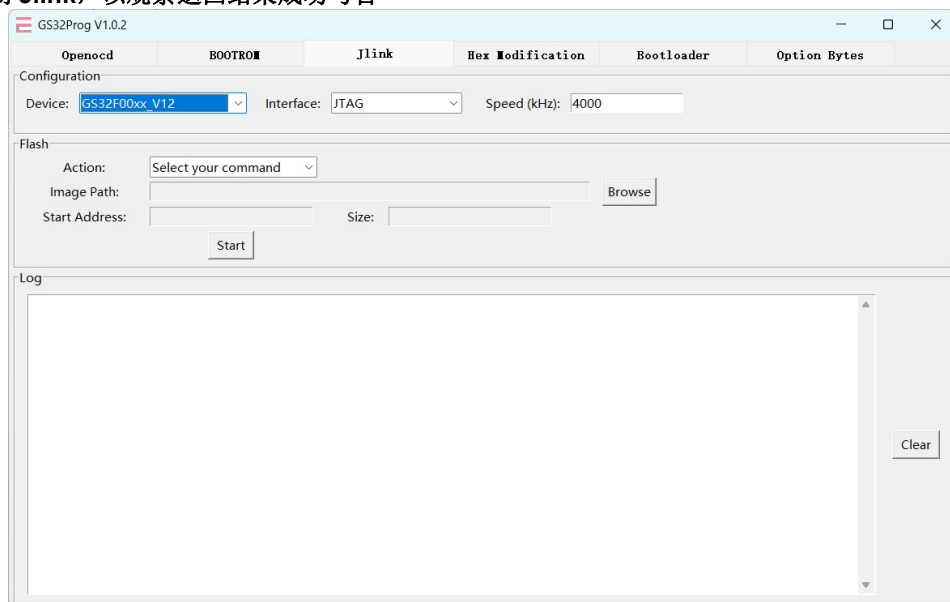


图 5.1-1 jlink 界面

6 Hex Modification 页签

6.1 说明

该界面提供更改 16byt 数据为一行的 Hex 文件格式。如增加 CRC 值，16byte 数据格式转 32byte 数据格式，数据编辑，hex 格式转 bin 格式。

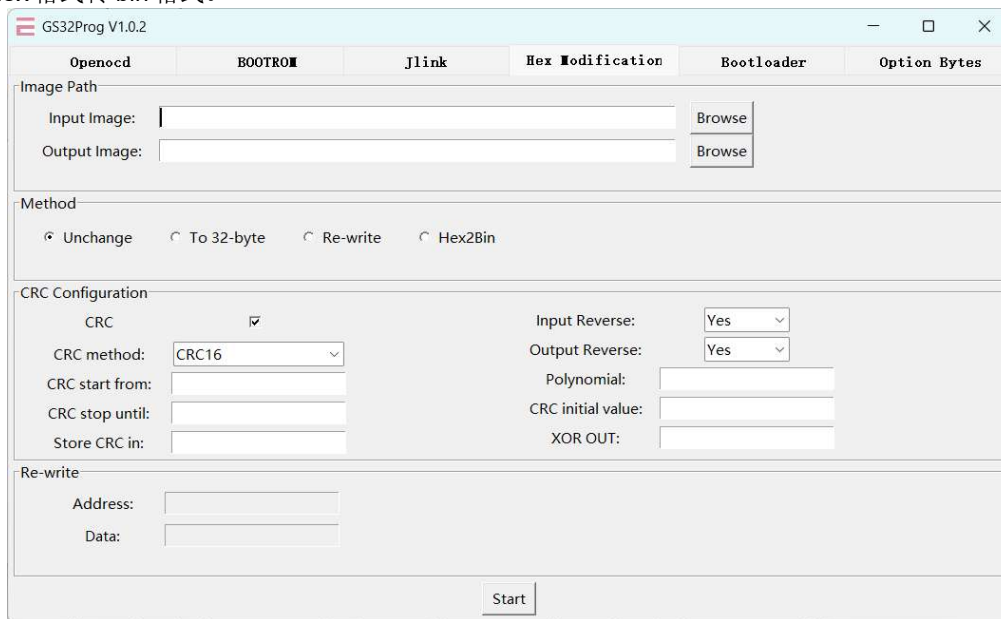


图 6.1-1 Hex Modification 界面

6.2 插入 CRC 值

若待转换 Hex 文件对 CRC 校准有要求，勾选 CRC，如图 13。（注意，若需要插入一个 CRC 校验值，则该图像必须先前定义了一个地址供 CRC 值存储。）该 CRC 校验值可供下位机比较接收图像的一致性。

依次填入需要计算 CRC 校验值的图像起始地址和结束地址、定义好的 CRC 值存储地址，CRC 校验方法、多项式值、CRC 初始值、XOR OUT、输入反向、输出反向。（注意，输入的多项式值、CRC 初始值、XOR OUT 的长度需要与 CRC 校验方法对应。并且填入的数据都以 16 进制为准，如 0x08000000）点击“Start”开始转换。

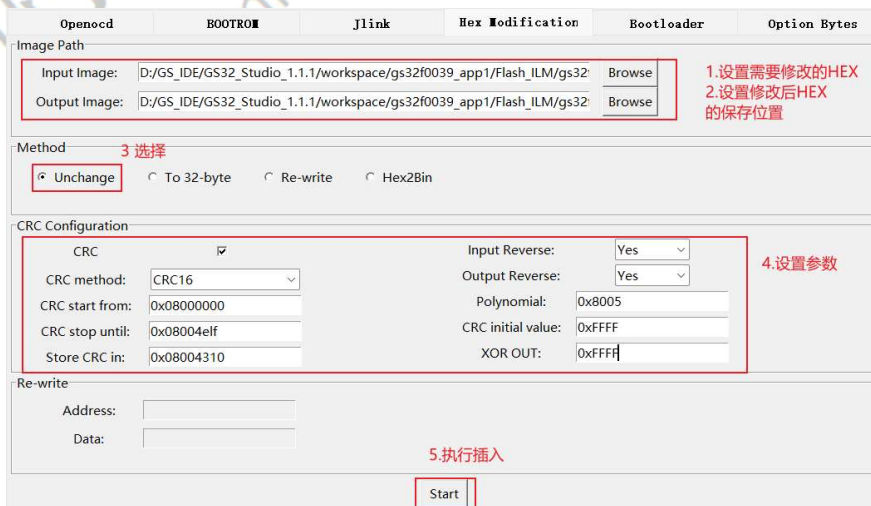


图 6.2-1 插入 CRC 步骤

6.3 To 32byte

将 16byte 数据为一行的 hex 格式转化为 32byte 数据为一行。可选择是否加入 CRC 值，如取消勾选，则不可编辑。

Openocd	BOOTROM	Jlink	Hex Modification	Bootloader	Option Bytes
Image Path					
Input Image:		D:/GS_IDE/GS32_Studio_1.1.1/workspace/gs32f0039_app1/Flash_ILM/gs32		Browse	
Output Image:		D:/GS_IDE/GS32_Studio_1.1.1/workspace/gs32f0039_app1/Flash_ILM/gs32		Browse	
Method					
<input type="radio"/> Unchange <input checked="" type="radio"/> To 32-byte <input type="radio"/> Re-write <input type="radio"/> Hex2Bin					
CRC Configuration					
CRC		<input type="checkbox"/>		Input Reverse: Yes	
CRC method:		CRC16		Output Reverse: Yes	
CRC start from:		0x08000000		Polynomial: 0x8005	
CRC stop until:		0x08004elf		CRC initial value: 0xFFFF	
Store CRC in:				XOR OUT: 0xFFFF	
Re-write					
Address:					
Data:					
Start					

图 6.3-1 To 32byte 界面

6.4 Re-write

重新编辑原 hex 文件内的数据，补充想要编辑的起始地址和数据即可。注意，预编辑数据长度只能限制在同一行。

Openocd	BOOTROM	Jlink	Hex Modification	Bootloader	Option Bytes
Image Path					
Input Image:		D:/GS_IDE/GS32_Studio_1.1.1/workspace/gs32f0039_app1/Flash_ILM/gs32		Browse	
Output Image:		D:/GS_IDE/GS32_Studio_1.1.1/workspace/gs32f0039_app1/Flash_ILM/gs32		Browse	
Method					
<input type="radio"/> Unchange <input type="radio"/> To 32-byte <input checked="" type="radio"/> Re-write <input type="radio"/> Hex2Bin					
CRC Configuration					
CRC		<input type="checkbox"/>		Input Reverse: Yes	
CRC method:		CRC16		Output Reverse: Yes	
CRC start from:		0x08000000		Polynomial: 0x8005	
CRC stop until:		0x08004elf		CRC initial value: 0xFFFF	
Store CRC in:				XOR OUT: 0xFFFF	
Re-write					
Address:		0x08001000			
Data:		12334567			
Start					

图 6.4-1 Re-write 界面

6.5 Hex2Bin

只需加载输入输出文件路径，注意输出路径需为 bin 格式。

Openocd	BOOTROM	Jlink	Hex Modification	Bootloader	Option Bytes
Image Path					
Input Image: D:/gs32f0039_app1.hex			Browse		
Output Image: D:/gs32f0039_app111.bin			Browse		
Method					
<input type="radio"/> Unchange <input type="radio"/> To 32-byte <input type="radio"/> Re-write <input checked="" type="radio"/> Hex2Bin					
CRC Configuration					
<input type="checkbox"/> CRC			Input Reverse: Yes <input type="button" value="v"/>		
CRC method: CRC16 <input type="button" value="v"/>			Output Reverse: Yes <input type="button" value="v"/>		
CRC start from: 0x08000000			Polynomial: 0x8005		
CRC stop until: 0x08004elf			CRC initial value: 0xFFFF		
Store CRC in:			XOR OUT: 0xFFFF		
Re-write					
Address: 0x08001000					
Data: 12334567					
<input type="button" value="Start"/>					

图 6.5-1 Hex2Bin 界面

7 Bootloader 页签

7.1 说明

此页面主要提供 APP 镜像头部的添加和镜像烧写功能。烧写功能要求下位机带有 bootloader

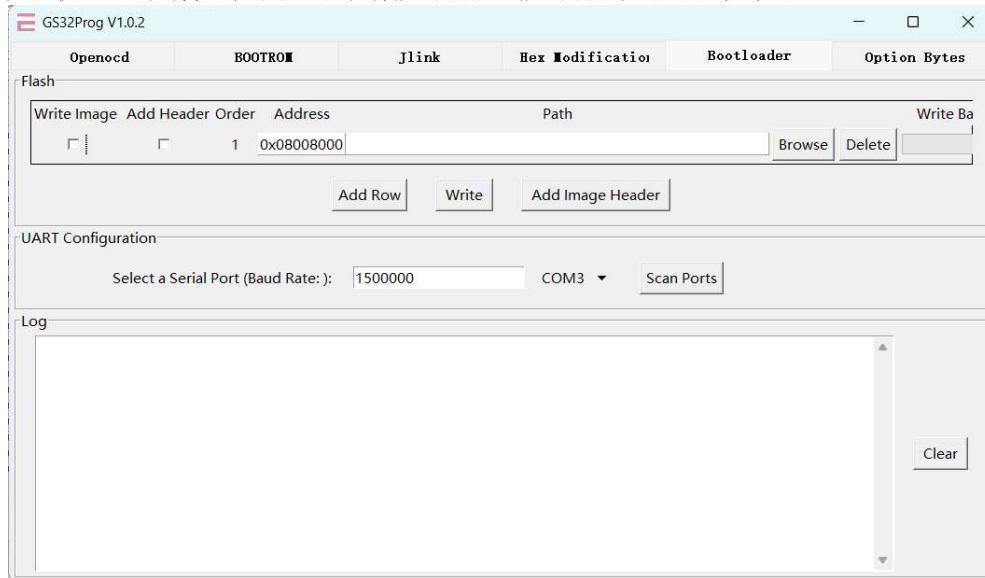


图 7.1-1 Bootloader 界面

7.2 镜像头部的添加

需要点击 **Browse** 按钮添加镜像文件（bin 文件）同时勾选对应行左侧的 **Add Header** 勾选框。点击下方 **ADD Image Header** 按钮，上位机开始为所有勾选过 **Add Header** 的行添加镜像头部。同时日志输出至下方 **log** 框中。

添加头部完成后，对应行左右侧会出现 ‘H’ 标记，标记此镜像已添加头部选项，生成的已添加头部文件位于 **output** 目录下。

文件名：原始文件名.bin -> 原始文件名_with_header.bin

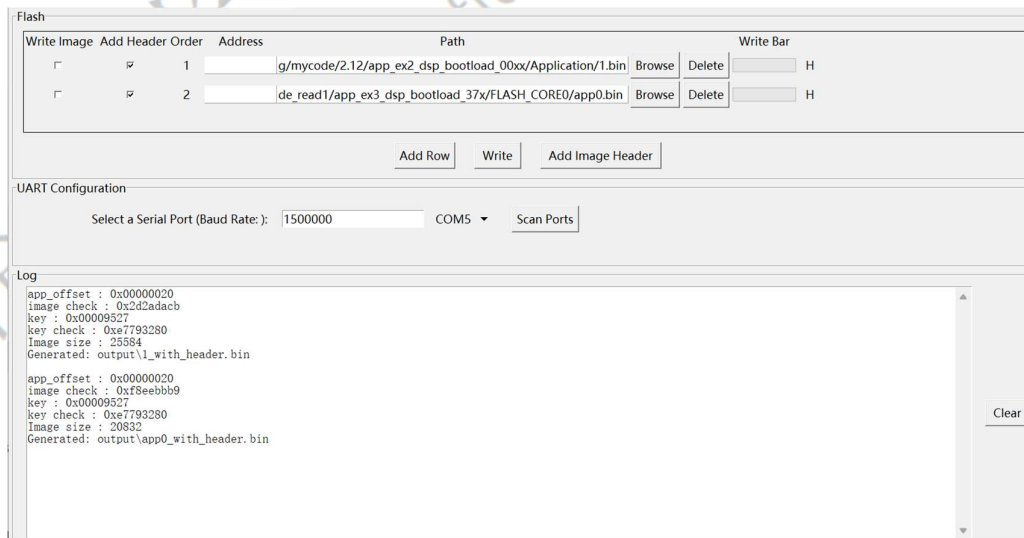


图 7.1-1 头部添加

7.3 镜像文件的烧录

烧录功能通过 UART 串口实现，要求下位机带有 bootloader。如图 19，用户可以烧录多个 APP 镜像。点击“Add Row”，增加行，若确定此次烧录，勾选左边的 Write Image 勾选框，若需要删除行，点击每行对应的“Delete”。

输入合法 Address 后下方点击 Write 按钮烧写。上位机开始烧写每行勾选过 Write Image 选项的对应镜像。烧写镜像时若未添加头部（右侧无‘H’标记），则会烧写 Browse 按钮所选文件。若已添加头部，则实际烧写 output 文件夹下对应生成的已添加头部的镜像文件。

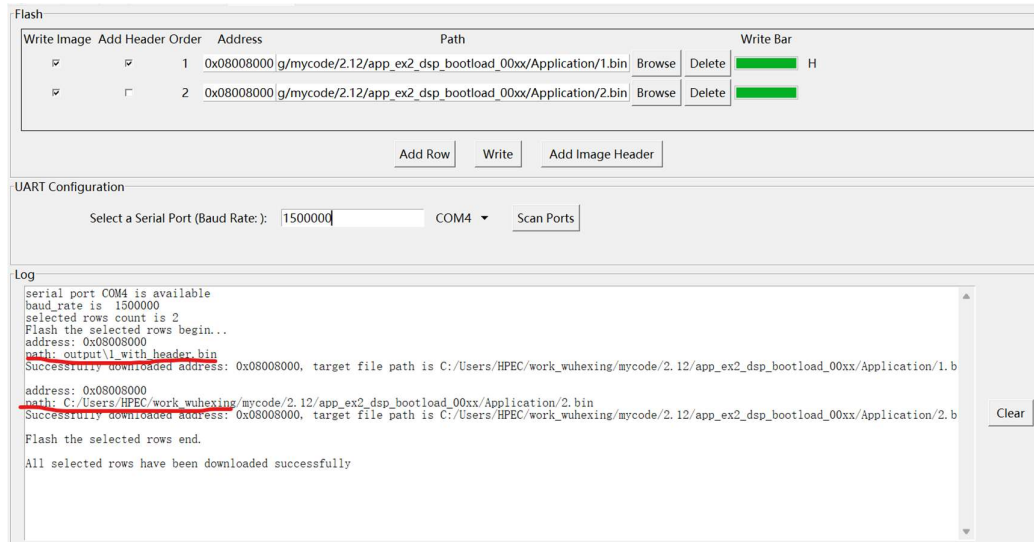


图 7.1-1 烧录

8 OptionBytes 页签

8.1 说明

此页面主要提供 option bytes 的 rdp 升降级操作，后续会增加其他的相关选项。

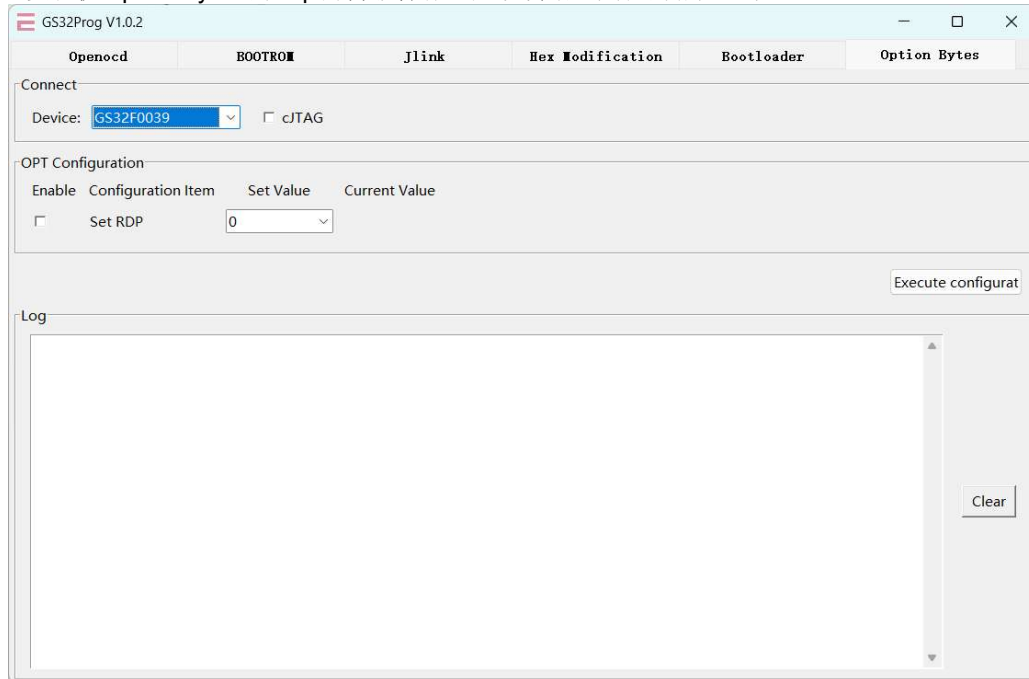


图 8.1-1 OptionBytes 界面

8.2 RDP

设置 RDP 等级。注意最高等级 2 设置后芯片将无法降级。等级 1 将为 0 会进行擦除动作，防止内部程序被解

密

等级 0: 无保护

等级 1: 无法读取和调试相关程序

等级 2: 永久锁定芯片



图 8.2-1 RDP 设置

9 Image Tools

9.1 Merge Hex

合并多个 hex 文件

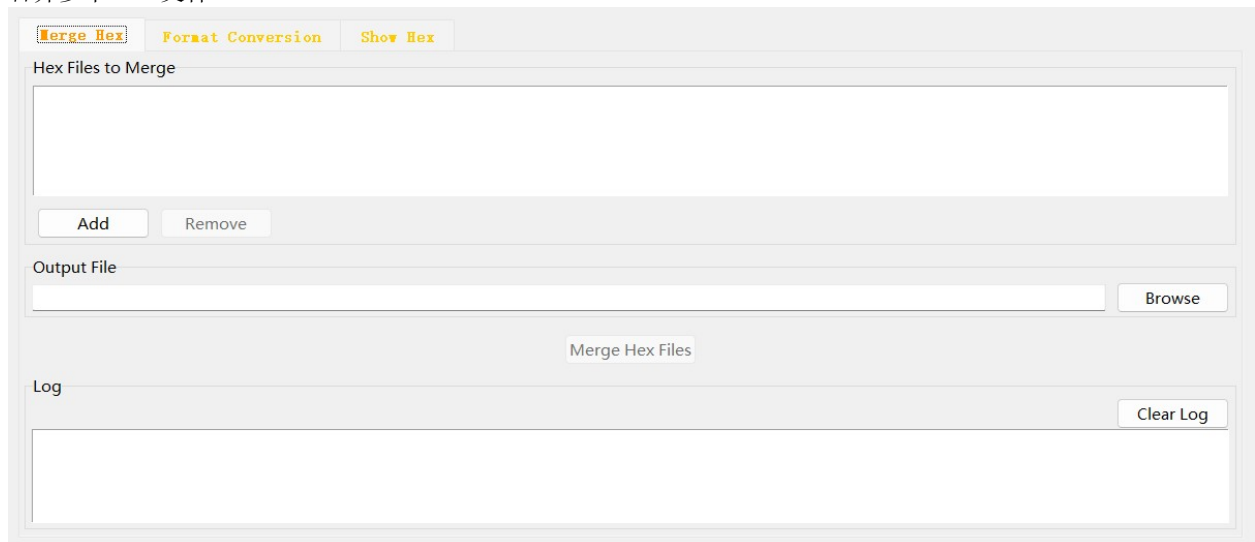


图 8.2-1 合并 hex 界面

9.2 Format Conversion

批量转换格式，支持格式 bin、elf、hex，支持转换格式 bin、elf、hex

图 8.2-1 格式转换界面

9.3 Show Hex

显示 bin、elf、hex 的 hex 内容

图 9.3-1 显示 hex 界面

10 附录

10.1 命令方式执行烧录、OptionBytes、CRC 验证

使用 GS32Prog 目录下的 GS32Prog.exe -h 获取以下帮助信息，此命令可以进行命令行方式的批量烧录、OptionBytes、CRC 校验

```
usage: GS32Prog.py [-h] [-nogui] [-v] [-dev modelname] [-usb_num USB_NUM] [-adapter_speed ADAPTER_SPEED] [-chip_freq
CHIP_FREQ]
                        [-cfg FILE] [-image FILE[@ADDR] [FILE[@ADDR] ...]] [-cjttag] [-erase_chip]
                        [-opt opt_name=value [opt_name=value ...]] [-optfile OPT_FILE] [-crc CRC8/CRC16/CRC32] [-
crc_start_addr address]
                        [-crc_stop_addr address] [-crc_save_addr address] [-poly value] [-init_val value] [-xor_out value]
                        [-reflected_in on/off] [-reflected_out on/off] [-out FILE]

GS32Prog.exe -nogui -dev GS32F0039 .\gs32f0039_dsp.elf
GS32Prog.exe -nogui -dev GS32F0039 -image .\gs32f0039_dsp.bin@0x08000
GS32Prog.exe -nogui -dev GS32F0039 -image .\gs32f0039_dsp1.hex .\gs32f0039_dsp2.hex
GS32Prog.exe -nogui -cfg .\scripts\gs32f0039_ftdi_dsp_flash.cfg -image .\gs32f0039_dsp.elf
GS32Prog.exe -nogui -image .\test\gs32f0039_dsp.hex -crc CRC32 -crc_start_addr 0x08000410 -crc_stop_addr 0x08000500 -
crc_save_addr 0x08004F20 -poly 0x8408 -init_val 0xFFFF -xor_out 0 -reflected_in on -reflected_out on -out ./xxx.hex

optional arguments:
  -h, --help            show this help message and exit
  -nogui                Use command mode/Use headless mode
  -v, --version          Version Information
  -dev modelname         Model information:
                        ['GS32F0039P', 'GS32F0039', 'GS32F0039H', 'GS32F0037', 'GS32F0037H', 'GS32F0033',
                        'GS32FMT5000', 'GS32F0049', 'GS32F0049H', 'GS32F0045', 'GS32F0041', 'GS32FMT5000A', 'GS32F0025A', 'GS32F00157A',
                        'GS32F00137A', 'GS32F379D', 'GS32F377D', 'GS32F374D', 'GS32F379DH', 'GS32F377DH', 'GS32F374DH', 'GS32F379S',
                        'GS32F377S', 'GS32F374S', 'GS32F379SH', 'GS32F377SH', 'GS32F374SH', 'GS32F388DA', 'GS32F075', 'GS32F075D',
                        'GS32FP65DK', 'GS32FP65DJ', 'GS32FP65DKH', 'GS32FP65DJH', 'GS32FP65SK', 'GS32FP65SJ', 'GS32FP65SKH', 'GS32FP65SJH',
                        'GS32FMT5000', 'GS32FMT4000', 'GS32F0025', 'GS32F0025H', 'GS32F0023', 'GS32F0039A', 'GS32F00157', 'GS32F00157H',
                        'GS32F00156', 'GS32F00155', 'GS32F00154', 'GS32F00153', 'GS32F00152', 'GS32F035', 'GS32F035H', 'GS32F035P',
                        'GS32F0049A', 'GS32F00137', 'GS32F00137H', 'GS32F00135', 'GS32F00132']
  -usb_num USB_NUM      1/2/3...
                        Please first use 'GS32ftdi.exe -info' to obtain the number corresponding to the USB serial.
  -adapter_speed ADAPTER_SPEED
                        Adjust JTAG frequency, unit: KHZ
  -chip_freq CHIP_FREQ  Adjust system main frequency during burning, unit: HZ
  -cfg FILE             openocd configuration file
  -image FILE[@ADDR] [FILE[@ADDR] ...]
                        File programming
                        Supports multiple file programming
                        Can specify start address with @ for programming
                        If you use CRC verification, please input the address of a single hex file
  -cjttag               Connect with cJTAG
  -erase_chip           Erase all/erase all before burning
  -opt opt_name=value [opt_name=value ...]
                        Option Bytes command: rdp=0/1/2
  -optfile OPT_FILE     File type .optfile
  -crc CRC8/CRC16/CRC32
                        CRC8,CRC16,CRC32
  -crc_start_addr address
                        Example:08000000.
                        crc verification start address.
  -crc_stop_addr address
                        Example:08000000.
                        crc verification end address, verification does not include this address.
  -crc_save_addr address
                        Example:08000000
                        crc verification result save address.
  -poly value           Used generator polynomial value.
                        Example:
                        1 0001 0000 0010 0001
                        0001 0000 0010 0001 = 0x1021 - Normal Representation
                        0001 0000 0010 0001 -> reversed : 1000 0100 0000 1000 = 0x8408 - Reversed representation
                        1 0001 0000 0010 000 -> rearranged: 1000 1000 0001 0000 = 0x8810 - Koopman representation
  -init_val value       The value used to initialize the CRC value / register.
                        In the examples above, always 0 is used, but it could be any value.
  -xor_out value        The Final XOR value is xored to the final CRC value before being returned.
                        This is done after the 'Result reflected' step. Obviously a Final XOR value of 0 has no
                        impact.
  -reflected_in on/off If this value is on, each input byte is reflected before being used in the calculation.
                        Reflected means that the bits of the input byte are used in reverse order.
                        So this also means that bit 0 is treated as the most significant bit and bit 7 as least
                        significant.
  -reflected_out on/off
                        If this value is on, the final CRC value is reflected before being returned.
                        The reflection is done over the whole CRC value,
                        so e.g. a CRC-32 value is reflected over all 32 bits.
  -out FILE             hex file after crc checksum
```

图 10.1-1 OptionByets 命令行帮助

10.2 命令方式执行 hex 合并

使用 GS32Prog 目录下的 GS32MergeHex.exe 进行 hex 合并

```
usage: GS32MergeHex.py [-h] [-image HEXFILE [HEXFILE ...]] [-out new_hex_file]

GS32MergeHex.exe -images .\xx1.hex .\xx2.hex -out .\xxall.hex

optional arguments:
  -h, --help            show this help message and exit
  -image HEXFILE [HEXFILE ...]
                        Input multiple hex files
  -out new_hex_file     If there is an existing hex file, it will be replaced directly.
```

图 10.2-1 GS32MergeHex 命令行帮助

10.3 命令方式获取 FTDI 连接项

使用 GS32Prog 目录下的 GS32ftdi.exe 获取连接信息

```
usage: GS32ftdi.py [-h] [-info]

GS32ftdi.exe -info

optional arguments:
  -h, --help            show this help message and exit
  -info                 Print FTDI Serials.
```

图 10.3-1 GS32ftdi 命令行帮助

11 修订历史

版本	时间	内容描述	审核人	批准人
v0.1	2024.7.6	初始版本，依据公司规定格式重新编写	Jason	
v0.2	2024.7.10	增加 OptionBytes 说明	Jason	
V0.3	2024.8.2	增加 Image Tools、增加命令行	Jason	
V0.4	2024.8.22	修改烧录信息、增加命令行	Jason	

格见半导体 Draft Only
Gejian Semi. confidential - NDA Restrictions