

GDLink-OB 调试器用户手册

1 功能概述

GDLink 是 GigaDevice（兆易创新）针对旗下 GD32 微控制器推出的一款调试器，其功能类似 STM32 系列微控制器的 STLink 调试器。

GDLink 适用于 GD32 系列的 Cortex M3、M4、M23 内核产品，尤其是对于 ARMv8-M 架构的 Cortex M23 内核如 GD32E23x 系列支持较好。

GDLink 功能强大但成本较高，GDLink-OB 对 GDLink 不常用的功能进行了删减，只保留常用的 SWD 调试功能，降低了成本，使用上兼容 GD32 的 START 系列开发板板载调试器。

GDLink-OB 使用 ARM 标准的 CMSIS-DAP 协议，理论上可用于所有 ARM Cortex M 系列 MCU，同时 USB 接口使用 HID 协议，免去安装驱动的麻烦。

目前 GDLink-OB 已经升级为 GDLink-S 和 GDLink-Lite，支持官方版本除离线下载之外的完整功能。

1.1 主要特性

- ◆ 小巧便携，仅优盘大小，重量不足 8 克。
- ◆ 使用标准 CMSIS-DAP 协议，支持所有 ARM Cortex M 系列 MCU 的调试。
- ◆ USB 使用 HID 协议，Windows 下免驱动。
- ◆ 支持 MDK、pyOCD、GD-Link Programmer 等软件。
- ◆ 内置 500mA 自恢复保险丝。
- ◆ 支持固件升级。

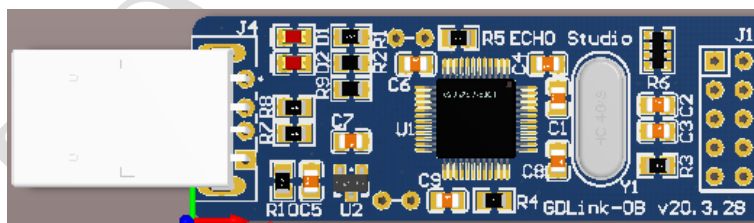


图 1 GDLink-OB 正面外观外观

1.2 接口定义

GDLink-OB 输出使用 2x5 PIN 2.54 排针，排针定义见背面丝印图 2。



图 2 2x5PIN 接口定义

SWD 调试通常只需要连接 5.0V、GND、CLK、DIO 四个管脚，**注意 CLK 和 DIO 在靠外的排针上，方形焊盘为 TDI，不是 CLK**；如目标板独立供电、可不连接 5.0V 电源，3.3V 电源对外输出能力有限，不建议使用

3.3V 电压对目标板供电。

GD32E23x 系列为 Cortex M23（ARMv8-M）内核，实测需要连接 TRESET 才能正常连接。

GDLink-OB 目前暂不支持标准 JTAG，因此 TCK、TMS、TDI、TDO 四个信号仅供参考。

短接 TDO/SWO 到 GND 后连接电脑，GDLink-OB 进入 IAP 模式，可升级固件，详见 3。

1.3 版本描述

GDLink-OB: GDLink Programmer 限制了部分功能，已停产。

GDLink-S: 热缩管塑封，USB-A 接口。

GDLink-Lite: 带外壳，MiniUSB 接口。

S 版本和 Lite 版本功能完全相同，只有外观的和 USB 接口的差别。

2 使用方法

2.1 Keil MDK

测试 MDK 版本为 5.29，在 MDK 中正确设置调试参数后，可正常下载、调试代码。

Cortex M4 内核的 GD32F330 调试设置见图 3，调试器选择“CMSIS-DAP Debugger”。

其它 Cortex M3/M0 内核的 MCU 设置相同，不再赘述。

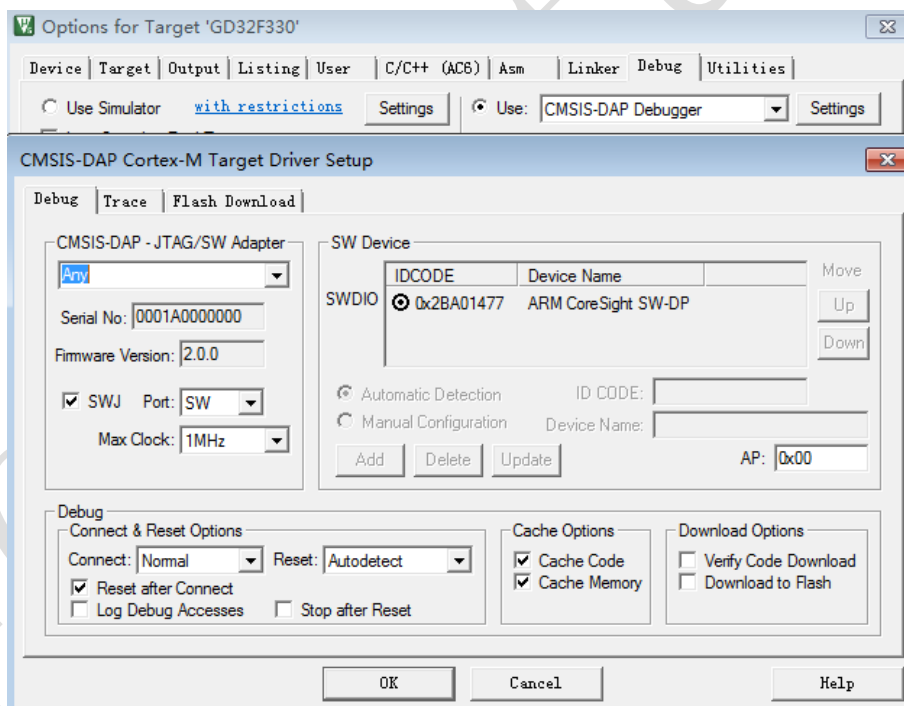


图 3 Cortex M4 内核的 GD32F330 调试设置

Cortex M23 内核的 GD32E230 调试设置见图 4，调试器选择“CMSIS-DAP ARMv8-M Debugger”。

注意此时调试器的 TRESET 管脚要连接目标芯片的 NRST 管脚，M0/M3/M4 内核不需要。

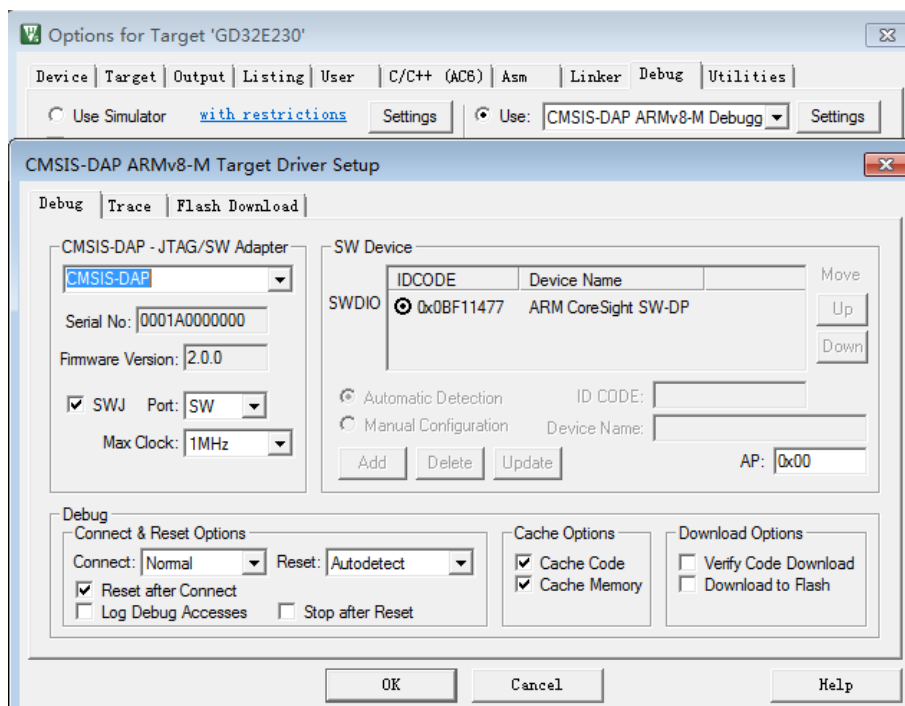


图 4 Cortex M23 内核的 GD32E230 调试设置

2.2 GD-Link Programmer

目前 GDLink-S 支持的 GD-Link Programmer 版本为 3.0.0.5950。

GD-Link Programmer 最新版本为 4.3.7.9954，实测该版本配合官方 GDLink 无法连上一些目标芯片，原因不明，因此 GDLink-S 暂未适配 GD-Link Programmer 4.3.7.9954。

2.2.1 GDLink-S/Lite

GDLink-S 为 GDLink-OB 的升级版本，两者主要差别在使用 GD-Link Programmer 时，GDLink-S 不会弹出图 6 的提示，并且支持 Target 菜单下的全部闪存操作，包括：

- 设置、解除读保护
- 整片擦除、页擦除闪存
- 闪存查空与对比
- 闪存编程
- 闪存整片读取与部分读取
- 运行程序

GDLink-Lite 为 GDLink-S 的带外壳版本，两者功能完全相同。

使用 GDLink-S 读取整片闪存见图 5。

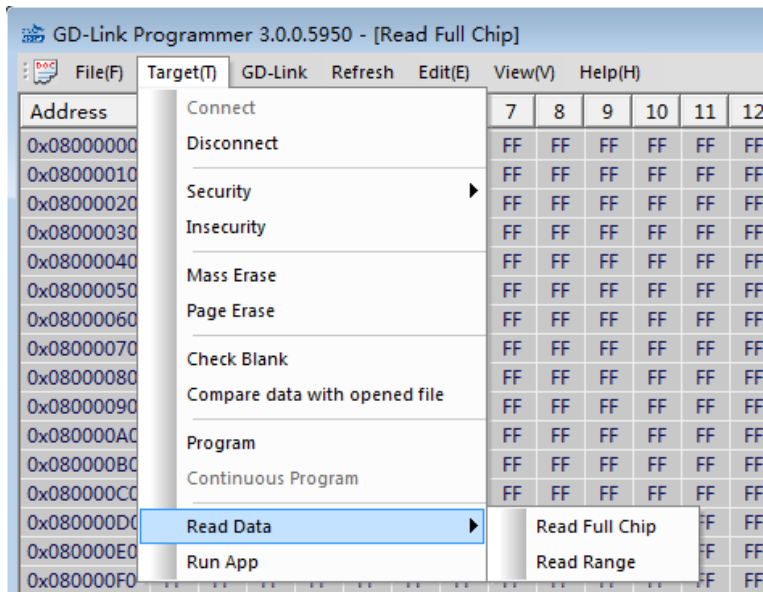


图 5 使用 GDLink-S 读取整片闪存

2.2.2 GDLink-OB

使用 GDLink-OB 连接好目标板，点击菜单“Target->Connect”连接目标板弹出提示见图 6，提示 GD-Link 为板载版本，功能受限。

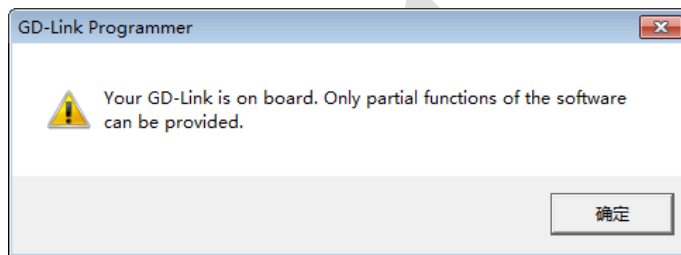


图 6 GD-Link Programmer 运行提示

点击“确定”后可查看 GD-Link 和目标芯片信息见图 7

option bytes 信息见图 8，option bytes 开始两个字节“A5 5A”表示没有开启读出保护。

on board 版本的 GD-Link 在 GD-Link Programmer 里面仅支持使能、关闭读保护。

目前 GDLink-OB 已经停产，请使用功能更加强大的 GDLink-S 和 GDLink-Lite。

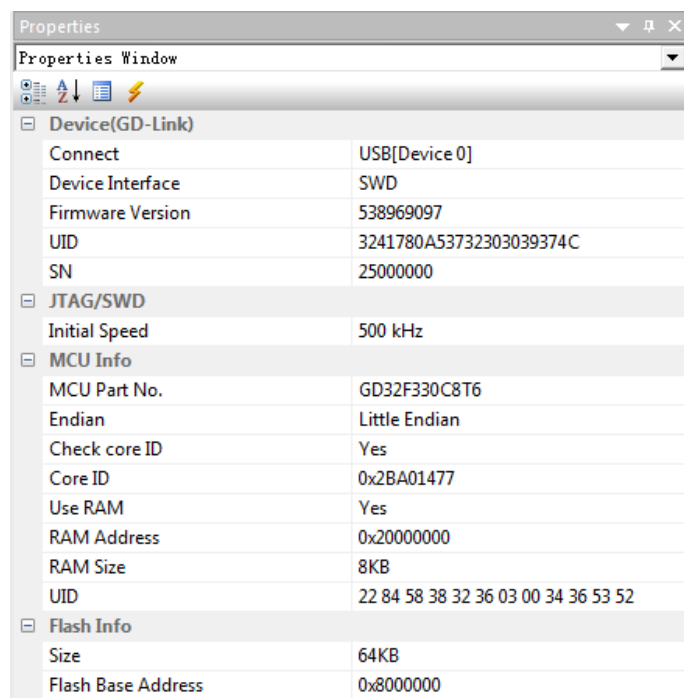


图 7 Properties 页面

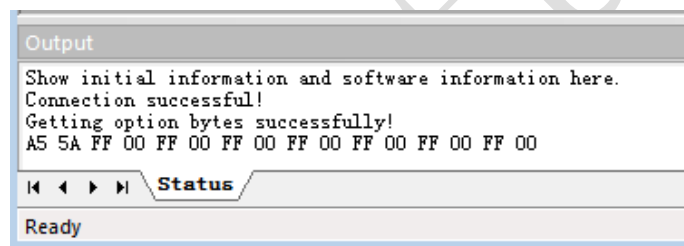


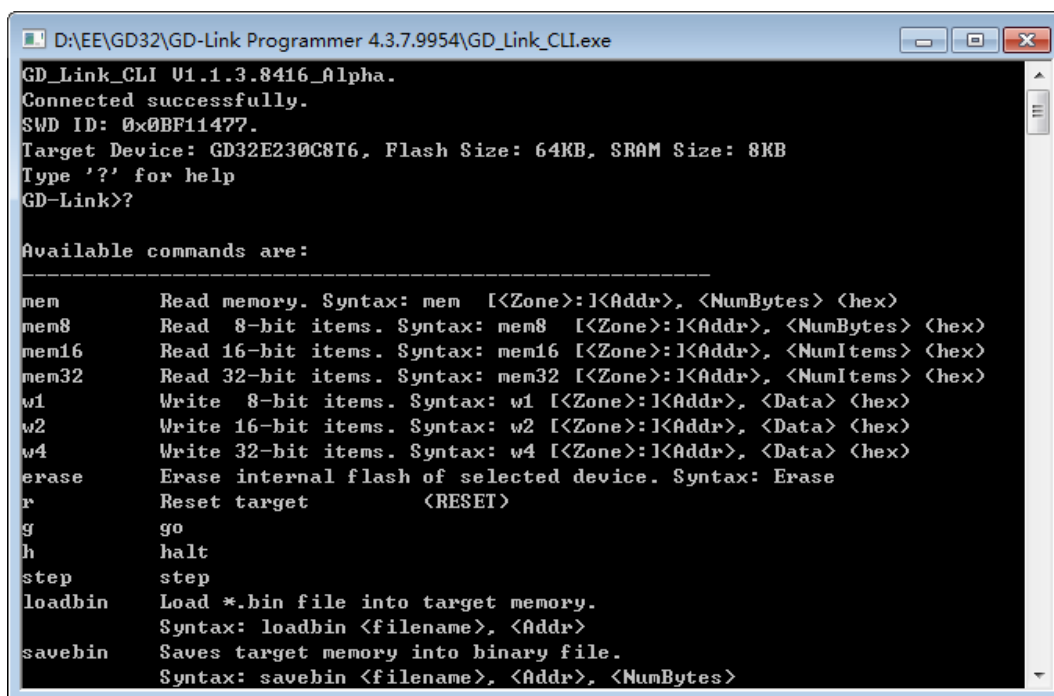
图 8 Status 页面

2.3 GD_Link_CLI

该工具来自 GDLink Programmer 4.3.7.9954 的安装包。直接点击 GD_Link_CLI.exe 即可运行，如图 9。输入 '?' 可以查看在线帮助。命令简介如下：

- mem/mem8/mem16/mem32 命令查看内存。
- w1/w2/w4 命令写内存。
- erase 命令擦除闪存。
- r/g/h/step 命令控制程序运行。
- laodbin/savebin 命令加载保存 bin 文件。
- SetPC 命令设置程序计数器。
- SetRDP 命令设置读保护。
- ReadAP/WriteAP/ReadDP/WriteDP 命令读写 CoreSight 寄存器。
- si 命令切换调试接口（GDLink-S 只支持 SWD，该命令无效）
- q 命令退出。

操作演示视频: <https://www.bilibili.com/video/bv18Z4y1w7Sy>



```
GD_Link_CLI V1.1.3.8416_Alpha.
Connected successfully.
SWD ID: 0x0BF11477.
Target Device: GD32E230C8T6, Flash Size: 64KB, SRAM Size: 8KB
Type '?' for help
GD-Link>?

Available commands are:
-----
mem      Read memory. Syntax: mem [<Zone>:]<Addr>, <NumBytes> <hex>
mem8     Read 8-bit items. Syntax: mem8 [<Zone>:]<Addr>, <NumBytes> <hex>
mem16    Read 16-bit items. Syntax: mem16 [<Zone>:]<Addr>, <NumItems> <hex>
mem32    Read 32-bit items. Syntax: mem32 [<Zone>:]<Addr>, <NumItems> <hex>
w1       Write 8-bit items. Syntax: w1 [<Zone>:]<Addr>, <Data> <hex>
w2       Write 16-bit items. Syntax: w2 [<Zone>:]<Addr>, <Data> <hex>
w4       Write 32-bit items. Syntax: w4 [<Zone>:]<Addr>, <Data> <hex>
erase    Erase internal flash of selected device. Syntax: Erase
r        Reset target          (RESET)
g        go
h        halt
step     step
loadbin  Load *.bin file into target memory.
          Syntax: loadbin <filename>, <Addr>
savebin  Saves target memory into binary file.
          Syntax: savebin <filename>, <Addr>, <NumBytes>
```

图 9 GD_Link_CLI 运行界面

2.4 pyOCD

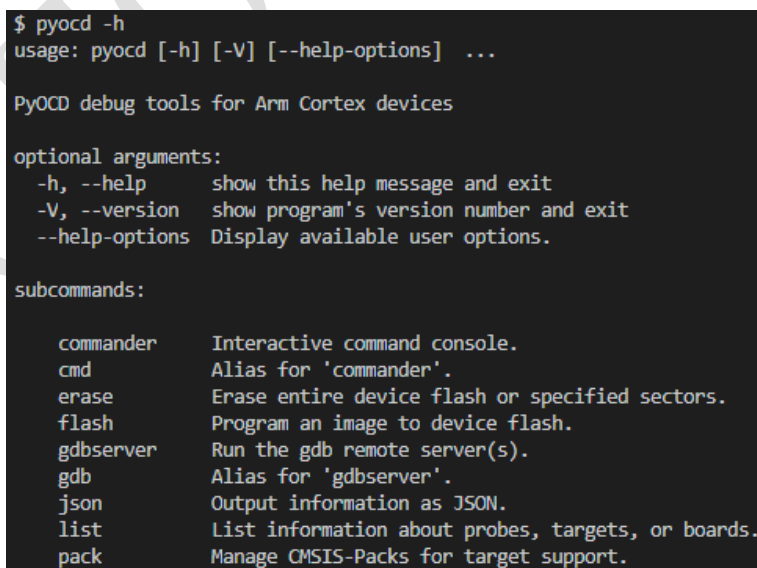
pyOCD 是一款使用 CMSIS-DAP 协议对 ARM Cortex-M 系列 MCU 进行编程、调试的开源 Python 库，功能十分强大。

项目主页: <https://github.com/mbedmicro/pyOCD>

GDLink-OB 完全支持使用 pyOCD 对目标芯片进行擦除、编程、调试等操作。

可使用"pip install pyocd"命令安装 pyocd，建议使用 Python3.6 来安装 pyocd。

安装后，帮助信息输出如下：



```
$ pyocd -h
usage: pyocd [-h] [-V] [--help-options] ...

PyOCD debug tools for Arm Cortex devices

optional arguments:
  -h, --help            show this help message and exit
  -V, --version          show program's version number and exit
  --help-options        Display available user options.

subcommands:
  commander            Interactive command console.
  cmd                  Alias for 'commander'.
  erase                Erase entire device flash or specified sectors.
  flash                Program an image to device flash.
  gdbserver            Run the gdb remote server(s).
  gdb                  Alias for 'gdbserver'.
  json                 Output information as JSON.
  list                 List information about probes, targets, or boards.
  pack                 Manage CMSIS-Packs for target support.
```

图 10 pyocd -h 命令输出

使用"pyocd list"命令列出当前的调试器信息：

```
$ pyocd list
#   Probe                Unique ID
-----
0   GD32 ARM CMSIS-DAP    0001A0000000
```

图 11 pyocd list 查看当前调试器信息

pyocd 需要使用 PACK 包来支持对应的目标器件，PACK 包是一个.pack 后缀的压缩文件，内部存储了器件的支持信息。在 MDK 中，通过菜单：Project->Manage->Pack Installer 打开 PACK 包管理器。

为了输入命令方便，建议设置一个全局环境变量\$PACKS 来记录 PACK 包的绝对路径。通常这个目录在 MDK 的安装目录下，相对路径为"Keil_v5\ARM\PACK\Download"。

使用"pyocd erase"命令擦除一颗 GD32E230C8 芯片的闪存内容，见图 12。

```
$ pyocd erase --chip --target gd32e230c8 --pack=$PACKS/GigaDevice.GD32E230_DFP.1.0.0.pack
0000895:INFO:eraser:Erasing chip...
0000977:INFO:eraser:Done
```

图 12 pyocd erase 命令擦除器件闪存

使用"pyocd flash"命令烧录一个 APP.hex 固件，见图 13。

```
$ pyocd flash APP.hex --target gd32e230c8 --pack=$PACKS/GigaDevice.GD32E230_DFP.1.0.0.pack
[=====] 100%
0004706:INFO:loader:Erased 57344 bytes (56 sectors), programmed 57344 bytes (56 pages), skip
```

图 13 pyocd erase 命令写入.hex 固件

使用"pyocd cmd"命令进入交互式命令行，可以执行更多的操作，比如查看寄存器、查看内存、外设，写入内存外设等操作。见图 14。

```
$ pyocd cmd --target gd32e230c8 --pack=$PACKS/GigaDevice.GD32E230_DFP.1.0.0.pack
Connected to GD32E230C8 [Lockup]: 0001A0000000
>>> r 0x08004000 0x80
08004000: 66 69 72 73 74 2e 0a 00 f8 b5 b9 4c 38 21 20 46 first.....L8! F
08004010: 00 f0 2e f9 30 b1 b6 48 39 21 00 f0 29 f9 08 b1 ....0..H9!..)...
08004020: 00 20 f8 bd 60 20 21 5a 48 48 c1 80 62 21 61 5a . ..`!ZH..b!aZ
08004030: 01 81 66 21 61 5a 41 81 21 88 62 88 12 04 55 18 ..f!aZA!.b...U.
08004040: 43 4b 1d 60 68 21 61 5a 81 81 42 49 42 4a 95 42 CK.`h!aZ..BIBJ.B
08004050: 03 d9 21 80 19 60 0b 0c 63 80 23 8a 65 8a 2d 04 ..!..`.c.#.e.-.
08004060: ed 18 3e 4b 1d 60 95 42 03 d9 21 82 19 60 0b 0c ..>K.`.B..!...`..
08004070: 63 82 23 8c 65 8c 2d 04 eb 18 39 4e 33 60 39 4d c.#.e.-...9N3`9M
>>> show cores
Cores: 1
Core 0 type: Cortex-M23
```

图 14 pyocd cmd 命令进入交互命令行

对于常用的擦除、编程操作，可以将操作写成脚本。图 15 是一个对 GD32E230 芯片进行擦除、编程的脚本。保存为 gd32e23x.sh 放入命令搜索路径。

使用"gd32e23x.sh erase"命令擦除器件。

使用"gd32e23x.sh flash APP.hex"命令写入固件。

使用"gd32e23x.sh info"命令查看器件信息。

更多脚本见：<https://github.com/xjtuecho/CMSIS-DAP/tree/master/GDLink-OB/Scripts>


```

1  #!/bin/sh
2
3  TARGET="gd32e230c8"
4  PACK_NAME="$PACKS/GigaDevice.GD32E230_DFP.1.0.0.pack"
5
6  Usage(){
7      echo "Usage: $0 [erase | flash | info]."
8  }
9
10 if [ $# -eq 0 ]; then
11     Usage
12 else
13     case $1 in
14     erase)
15         echo "Erase Device..."
16         pyocd erase --chip --target $TARGET --pack=$PACK_NAME
17         ;;
18     flash)
19         if [ $# -eq 2 ]; then
20             echo "Flash Device..."
21             pyocd flash $2 --target $TARGET --pack=$PACK_NAME
22         else
23             Usage
24             echo "Usage: $0 erase [APP.hex]"
25         fi
26         ;;
27     info)
28         echo "Try unlock Device..."
29         pyocd-flashtool --unlock --target $TARGET --pack=$PACK_NAME
30     esac
31 fi

```

图 15 一个擦除、编程 GD32E230 的脚本

3 常见问题

3.1 为什么连不上目标芯片？

根据实际客户反馈，连不上目标芯片大部分原因是杜邦线接线问题。包括但不限于以下情况：

- 连接 JTAG 接口，GDLink-S 只支持 SWD，不支持 JTAG。
- CLK 和 DIO 管脚接错，从背面看排针丝印 CLK 和 DIO 在靠外的那排排针上。图 2 中的方形焊盘不是 CLK，而是 TDI。
- CLK 和 DIO 管脚交叉，SWD 调试的 CLK 和 DIO 是直连，不是交叉。
- 杜邦线不通。可以用万用表通断档排除该问题。

另外一个常见连接不稳定原因是使用 GDLink-S 板载的 3.3V 给目标板供电，板载的 3.3V 电源输出能力有限，不建议对外进行供电。目标板请独立供电，或者使用 GDLink-S 的 5V 给目标板供电。

此外，如果目标板应用程序使用了 PA14(SWCLK)、PA15(SWDIO)两个管脚，也容易导致连接不上或者连接不稳定，请连接 TRESET 到目标板 NRST，连接上以后先擦除目标板 MCU 中的固件。

3.2 GDLink Programmer 提示软件过时？

目前 GDLink-S 暂时只支持 GDLink Programmer 3.0.0.5950 版本。

使用 GDLink Programmer 4.3.7.9954 会弹出图 16 提示，无法使用。

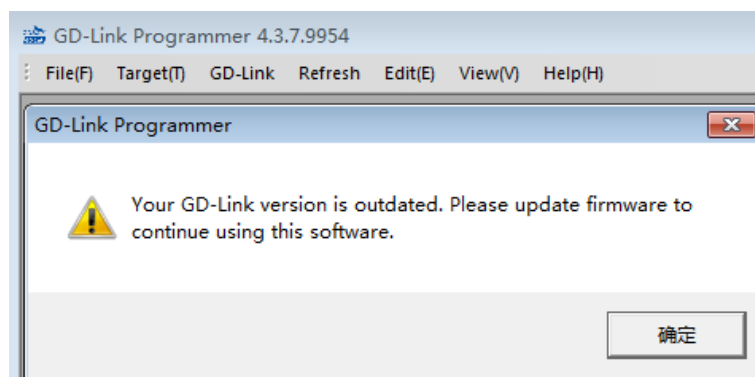


图 16 4.3.7.9954 版本提示

由于实测官方 GDLink 使用 4.3.7.9954 版本无法连上 GD32F150/GD32E230，原因未知，GDLink-S 暂未适配 4.3.7.9954 版本，请使用 3.0.0.5950 版本。

4.3.7.9954 版本自带了一个命令行模式程序：GD_Link_CLI.exe，与 GDLink-S 配合工作良好，详情见 2.3。

4 固件更新

使用短路冒短接 GDLink-OB 输出端子上的 TDO/SWO 和 GND，然后插入电脑，安装驱动以后，设备管理器中会出现一个虚拟串口，使用“超级终端”连接该串口更新固件。可参考 UIMeter 使用 XBOOT 更新固件的视频：

<https://www.bilibili.com/video/av83660645>

注：执行 ymodem 命令会自动擦除固件，如果没有写入新固件导致设备无响应，重新执行 ymodem 命令写入固件即可。

如果您的设备使用正常，不建议进行固件升级操作。

5 更新记录

更新日期	更新类型	更新人	更新内容
2020/4/10	A	Echo	新建文档
2020/7/30	A	Echo	增加 XBOOT 中使用超级终端更新固件说明
2020/12/3	A	Echo	增加 GDLink-S 产品描述
2021/3/22	A	Echo	增加 GDLink-Lite 产品描述，完善文档

注：

M-->修改

A -->添加

ECHO Studio 保留本文档最终解释权。

请使用 PDF 书签阅读本文档，快速定位所需内容！

项目主页：<https://github.com/xjtuecho/CMSIS-DAP>

国内镜像：<https://gitee.com/xjtuecho/CMSIS-DAP>