AN80002

应用笔记

Rev 1　2022/05/23

通过 pyOCD 操作 MM32-LINK Series 编程

简介

本应用笔记旨在说明使用 pyOCD 操作 MM32-LINK Series 对目标芯片编程的方法。

表 1-1 适用系列型号

系列	型号
MM32-LINK	MM32-LINK Series

1 关于 pyOCD

1.1 pyOCD 简介

pyOCD 是基于 Python 的工具和 API，用于调试，编程和探索 ARM Cortex 微控制器。

其包含六大特性：

- Gdbserver：通过命令行或 IDE 调试 GDB。由 VSCODE Cortex-Debug 插件和 Eclipse 嵌入 CDT 支持。
- Python API：通过直接的 API 完全控制目标设备，非常适合 CI，定制测试和调试工具，安全研究等。
- CMSIS-Packs：访问由 CMSIS 设备家族包支持的全套 ARM Cortex-M 设备。包可以通过 pyOCD 管理并按零件号安装，或者自己下载包装。
- Flash 编程：将固件和数据镜像编程为内部或外部闪存。当然，镜像也可以加载到 RAM 中。
- 半主机与 SWV：使用控制台和文件 I/O 的完整半主机实现。SWV 基本 printf 输出，通过 TCP/IP 流式传输 RAW SWO 数据，在 Python 中构建 SWO 事件处理数据流程图。
- 内置设备：内置了 70 多个常用 ARM Cortex-M 设备的支持。

2 使用 pyOCD

2.1 安装 pyOCD

2.1.1 环境需求及配置

PyOCD 需要 Python 3.6 或更高版本，以及最新版本的 libusb。它可以在 macOS、Linux、FreeBSD 和 Windows 平台上运行。

所以要运行 pyOCD 请准备以下内容：

- 操作系统：Windows（macOS/Linux/FreeBSD/Windows，以 Windows 为例）
- Python 3.6 及以上：https://www.python.org/
- libusb：https://libusb.info/

根据上述环境下载 Python 与 libusb 的安装包或压缩包，并安装 Python，安装完毕后，将 libusb 解压并将 libusb.dll 的路径添加到环境变量或将 libusb.dll 复制到 Python 根目录。

2.1.2 安装 pyOCD

可以通过 pip 安装或升级 pyOCD 的最新稳定版本， 如下所示：

```
pip install -U pyocd
```

如果安装过程缓慢请尝试通过国内镜像源安装：

```
pip install -U pyocd -i https://pypi.tuna.tsinghua.edu.cn/simple
```

安装成功后在命令行中输入 pyOCD 应得到以下结果：

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

PyOCD debug tools for Arm Cortex devices

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

```
subcommands:

    commander (cmd)
                    Interactive command console.
    erase           Erase entire device flash or specified sectors.
    load (flash)    Load one or more images into target device memory.
    gdbserver (gdb)
                    Run the gdb remote server(s).
    json            Output information as JSON.
    list            List information about probes, targets, or boards.
    pack            Manage CMSIS-Packs for target support.
    reset           Reset a target device.
    server          Run debug probe server.
      rtt           SEGGER RTT Viewer.
```

2.1.3 安装目标芯片支持

通过内置支持和 CMSIS-Packs，pyOCD 支持市场上几乎所有可用 Cortex-M 的 MCU。

通过下面的命令可以查看当前系统支持的目标类型表，包括目标名称、供应商和部件号。此外，它会打印每个目标是内置的还是来自 CMSIS-Pack。

```
pyocd list --targets
```

PyOCD 提供了一个名为“cortex_m”的通用目标类型。这种目标类型将能够连接和调试几乎所有正确实现 CoreSight 架构的 Cortex-M 设备。然而，闪存不能被编程，并且没有提供存储器映射。此外，它可能不适用于某些需要特殊处理操作（如复位和暂停）的设备。

由于通用“cortex_m”目标的限制，如果目标类型为“cortex_m”，将记录如下所示的警告。

```
0000183:WARNING:board:Generic cortex_m target type is selected; is this
intentional? You will be able to debug but not program flash. To set the
target type use the '--target' argument or 'target_override' option. Use
'pyocd list --targets' to see available targets.
```

如果您的目标不在设备列表中，可以通过 CMSIS-Packs 方式安装支持。

您可直接通过下面的命令从网络直接获取支持（以 MM32F3277 为例）：

```
pyocd pack install mm32f3277
```

如果您无法直接访问网络可以通过提前下载对应的芯片 pack 文件，

- 官方 CMSIS-Pack 列表：http://www.keil.com/dd2/pack/
- 灵动官网：https://www.mindmotion.com.cn/support/software/keil_pack/

有两种方法可以使用手动下载的包。

最简单的选项是将 `--pack` 选项传递给 pyocd 工具，指定 .pack 文件的路径。PyOCD 不会缓存有关以这种方式使用的包的任何信息，因此除了其他参数之外，每次调用都必须传递此参数。例如，要运行 GDB 服务器，您可以执行 `pyocd gdbserver --pack=MindMotion.MM32F3270_DFP.1.0.5.pack`。请注意，您可以将多个 `--pack` 参数传递给 pyOCD，这在 pyOCD 的脚本执行中可能很有用。

要获得更持久的解决方案，请使用 pyocd.yaml 配置文件。在配置文件中，pack 将会话选项设置为单个 .pack 文件路径或路径列表。现在，当您运行该 pyocd 工具时，它会自动选择要使用的包文件。

```
pack:
  - /Users/admin/CMSIS-Packs/MindMotion.MM32F3270_DFP.1.0.5.pack
  - /Users/admin/CMSIS-Packs/MindMotion.MM32F0270_DFP.0.1.1.pack
```

有关 pyocd.yaml 介绍详细请见 https://pyocd.io/docs/configuration.html

安装 DFP 后，该 `pyocd list –targets` 命令将在其输出中显示新目标，您可以立即开始将目标支持与其他 pyocd 子命令一起使用。要获取所有已安装包的列表，请使用 `pack show` 子命令。

2.2 使用 pyOCD 烧写程序

2.2.1 使用 pyOCD 擦除目标芯片

pyOCD 的擦除命令为 `pyocd erase` 。 通过 `pyocd erase --help` 命令查看擦除帮助。

以下命令演示了如何擦除基于 MM32F3277 为目标的全部闪存：

```
pyocd erase -t MM32F3277G7P --chip
```

输出结果：

```
$ pyocd erase -t MM32F3277G7P --chip
0001535 W Invalid coresight component, cidr=0x0 [rom_table]
0001559 I Erasing chip... [eraser]
0001801 I Done [eraser]
$
```

2.2.2 使用 pyOCD 烧写目标芯片

pyOCD 的编程命令为 `pyocd flash` 。 通过 `pyocd flash --help` 命令查看编程帮助。

以下命令演示了如何为基于 MM32F3277 的目标编程：

```
pyocd flash -t MM32F3277G7P .\mm32f3270_bl_crc.bin
```

输出结果

```
$ pyocd flash -t MM32F3277G7P .\mm32f3270_bl_crc.bin
0001588 W Invalid coresight component, cidr=0x0 [rom_table]
0001611 I Loading C:\Users\liuyu\mm32f3270_bl_crc.bin [load_cmd]
[==============================================] 100%
0004237 I Erased 30720 bytes (30 sectors), programmed 30720 bytes (30 pages), skipped 17408 bytes (17 pages) at
17.90 kB/s [loader]
$
```

以下命令演示了如何为基于 MM32F3277 的目标在特定地址编程：

```
pyocd flash -t MM32F3277G7P --base-address 0x08001000 Sample.bin
```

3 小结

通过上面的介绍，简要描述了如何通过 pyOCD 操作 MM32-LINK Series 对 MM32F3277G7P 的编程操作（擦除、烧录）。有关 pyOCD 及 MM32-LINK Series 的更详细的使用说明请参阅 https://pyocd.io/ 和 MM32-LINK Series 相关使用文档。

4 修改记录

表 4-1 修改记录

日期	版本	内容
2022/05/23	1	初始版本发布