

### 前言

本应用笔记说明了 PY32 微控制器自举程序中使用的 USB DFU 协议。它详细说明了每个支持的指令。若需器件自举程序 USB 硬件资源和要求的更多信息, 请参考“PY32\_Bootloader\_UserManual”。

## 目录

|     |                       |   |
|-----|-----------------------|---|
| 1   | 自举程序代码序列 .....        | 3 |
| 2   | 自举程序命令集.....          | 4 |
| 2.1 | 器件相关的自举程序参数 .....     | 5 |
| 2.2 | Get 命令 .....          | 5 |
| 2.3 | Get ID 命令 .....       | 5 |
| 2.4 | Read Memory 命令 .....  | 5 |
| 2.5 | Go 命令 .....           | 6 |
| 2.6 | Write Memory 命令 ..... | 6 |
| 2.7 | Erase Memory 命令 ..... | 7 |
| 3   | 版本历史.....             | 9 |



## 2 自举程序命令集

下表中列出了支持的命令。本部分将详细说明其中的每一个命令。

表 3-1. USB DFU 自举程序命令

| 命令           | 命令代码 | 命令说明                                     |
|--------------|------|--|
| Get          | 0x00 | 获取当前自举程序版本及允许使用的命令                       |
| Get ID       | 0x02 | 获取芯片 ID                                  |
| Read Memory  | 0x11 | 从应用程序指定的地址开始读取最多 256 个字节的存储器空间           |
| Go           | 0x21 | 跳转到内部 Flash 或 SRAM 内的应用程序代码              |
| Write Memory | 0x31 | 从应用程序指定的地址开始将最多 256 个字节的数据写入 RAM 或 Flash |
| Erase Memory | 0x44 | 使用双字节寻址模式擦除一个到全部 Flash 页面                |

1. 如果接收到拒绝命令或在执行命令期间出现错误，自举程序则会发送 **NACK** 字节并返回检查命令状态。

### 通信安全

编程工具(PC)到器件的所有通信均通过如下方式验证：

1. 校验和：接收到的数据字节块进行异或运算。每个通信结尾增加一个字节（校验和字节），包含前面所有字节异或运算的结果。异或运算所有接收到的字节，即数据包加上校验和字节，结果必须为 **0x00**
2. 针对每条命令，主机都会发送一个字节及其补码（异或结果 = **0x00**）
3. UART：激活奇偶校验（偶校验）

每个数据包或者被接受（**ACK** 应答）或者被丢弃（**NACK** 应答）：

- **ACK = 0x79**
- **NACK = 0x1F**

### 2.1 器件相关的自举程序参数

虽然所有 PY32 器件的 USB DFU 自举程序协议命令集和序列均相同，但某些参数与具体器件相关。对一些命令，某些参数值可能取决于所使用的器件。相关参数如下：

- PID (产品 ID)，该参数因器件而异
- 执行 Read Memory、Go 和 Write Memory 命令时，自举程序允许的有效存储器地址（RAM、Flash、系统存储器、选项字节区域）。
- 执行 Write Protect 命令时使用的 Flash 扇区的大小。

要了解所使用器件中这些参数值的详细信息，请参见芯片参考手册。

### 2.2 Get 命令

用户通过 Get 命令可获取自举程序的版本及支持的命令。自举程序接收到 Get 命令后，会将自举程序版本和支持命令的代码发送给主机。

表 3.2-1. Get 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号 | 数据   | 描述                                     |
|-------|------|------|--|
| 主机端   | 1    | 0x00 | Get 命令                                 |
|       | 2    | 0xFF | 命令补码                                   |
| 器件端   | 1    | 0x79 | ACK                                    |
|       | 2    | 0x06 | N = 字节数 - 1，除当前字节和 ACK 之外              |
|       | 3    | 0x10 | 自举程序版本 (0 < 版本 < 255)，示例：0x10 = 版本 1.0 |
|       | 4    | 0x00 | Get 命令                                 |
|       | 5    | 0x02 | Get ID 命令                              |
|       | 6    | 0x11 | Read Memory 命令                         |
|       | 7    | 0x21 | Go 命令                                  |
|       | 8    | 0x31 | Write Memory 命令                        |
|       | 9    | 0x44 | Extended Erase 命令                      |
|       | 10   | 0x79 | ACK                                    |

### 2.3 Get ID 命令

Get ID 命令用于获取芯片 ID (标识) 的版本。自举程序接收到此命令后，会将产品 ID 发送给主机。

表 3.3-1. Get ID 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号 | 数据   | 描述                        |
|-------|------|------|---------------------------|
| 主机端   | 1    | 0x02 | Get ID 命令                 |
|       | 2    | 0xFD | 命令补码                      |
| 器件端   | 1    | 0x79 | ACK                       |
|       | 2    | 0x01 | N = 字节数 - 1，除当前字节和 ACK 之外 |
|       | 3    | 0x00 | PID MSB                   |
|       | 4    | 0x64 | PID LSB                   |
|       | 5    | 0x79 | ACK                       |

### 2.4 Read Memory 命令

Read Memory 命令用于从 RAM、Flash 和信息块（系统存储器或选项字节区域）中的任何有效存储器地址（参见注释）读取数据。

自举程序接收到 Read Memory 命令后，会将 ACK 字节发送到应用程序。发送 ACK 字节后，自举程序会等待一个地址（4 个字节，字节 1 表示地址 MSB，字节 4 表示 LSB）和一个校验和字节，

然后检查接收到的地址。如果接收到的地址有效且校验和正确，则自举程序将发送一个 ACK 字节，否则将发送一个 NACK 字节并中止此命令。

如果接收到的地址有效且校验和正确，则自举程序将等待要发送的字节数 - 1 ( N 字节) 及其补充字节 (校验和)。如果校验和正确，则自举程序将从接收的地址开始向应用程序发送所需数据 ( (N + 1) 字节)。如果校验和错误，则会在中止命令之前发送 NACK。

表 3.4-1. Read Memory 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号    | 数据   | 描述                            |
|-------|---------|------|-------------------------------|
| 主机端   | 1       | 0x11 | Read Memory 命令                |
|       | 2       | 0xEE | 命令补码                          |
| 器件端   | 1       | 0x79 | ACK                           |
| 主机端   | 1~4     | -    | 起始地址 (字节 1: MSB, 字节 4: LSB)   |
|       | 5       | -    | 校验和: 异或 (字节 1、字节 2、字节 3、字节 4) |
| 器件端   | 1       | 0x79 | ACK                           |
| 主机端   | 1       | -    | 待读取的字节数 - 1 (0 < N ≤ 255)     |
|       | 2       | -    | 校验和: 异或字节 8 (字节 8 的补码)        |
| 器件端   | 1       | 0x79 | ACK                           |
|       | 2~(N+2) | -    | 待读取的数据 (N+1 字节)               |

2.5 Go 命令

Go 命令用于从应用程序指定的地址开始执行已下载的代码或其它任何代码。自举程序接收到 Go 命令后，会将 ACK 字节发送到应用程序。发送 ACK 字节后，自举程序会等待一个地址 ( 4 个字节，字节 1 表示地址 MSB，字节 4 表示 LSB) 和一个校验和字节，然后检查接收到的地址。如果接收到的地址有效且校验和正确，则自举程序将发送一个 ACK 字节，否则将发送一个 NACK 字节并中止此命令。

若接收的地址有效且校验和正确，则自举程序固件将执行以下操作：

- 初始化自举程序使用的外设寄存器，将其设置为默认复位值
- 初始化用户应用程序的主栈指针
- 跳转到接收的“地址 + 4” (与应用程序中复位处理程序的地址相对应) 中指定的存储器位置。

例如，如果接收的地址为 0x0800 0000，则自举程序将跳转到地址为 0x0800 0004 的存储器位置。一般来说，主机应发送基准地址，在该地址指定应用程序的跳转目标位置。

表 3.5-1. Go 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号 | 数据   | 描述                            |
|-------|------|------|-------------------------------|
| 主机端   | 1    | 0x21 | Go 命令                         |
|       | 2    | 0xDE | 命令补码                          |
| 器件端   | 1    | 0x79 | ACK                           |
| 主机端   | 1~4  | -    | 跳转地址 (字节 1: MSB, 字节 4: LSB)   |
|       | 5    | -    | 校验和: 异或 (字节 1、字节 2、字节 3、字节 4) |
| 器件端   | 1    | 0x79 | ACK                           |

2.6 Write Memory 命令

Write Memory 命令用于将数据写入 RAM、Flash 或选项字节区域的任意有效存储器地址 (参见下文注释)。

自举程序接收到 Write Memory 命令后，会将 ACK 字节发送到应用程序。发送 ACK 字节后，自举

程序会等待一个地址（4 个字节，字节 1 表示地址 MSB，字节 4 表示 LSB）和一个校验和字节，然后检查接收到的地址。对于选项字节区域，起始地址必须为选项字节区域（参见注释）的基准地址，以避免在此区域中不当写入数据。

注：

- 1. 对 Flash/SRAM 的写入操作必须按字（32 位）对齐且写入数据应为 4 字节的位数。如果写入数据少于分配的存储空间，则应以 0xFF 填充剩余字节。
- 2. 有关所用器件有效存储器地址的详细信息，请参见芯片参考手册。

如果接收到的地址有效且校验和正确，则自举程序将发送一个 ACK 字节，否则将发送一个 NACK 字节并中止此命令。若接收的地址有效且校验和正确，则自举程序将执行以下操作：

- 获取字节 N，其中包含特接收的数据字节数
- 接收用户数据（(N + 1) 字节）以及校验和（N 以及所有数据字节的异或运算结果）
- 从接收的地址开始针对存储器进行用户数据编程
- 在命令结束时，如果写操作成功完成，则自举程序将发送 ACK 字节；否则将 NACK 字节发送给应用程序并中止命令

对 PY32 执行写操作时，允许写入的最大数据块长度为 256 个字节。

如果将 Write Memory 命令发送到选项字节区域，则在写入新值前会清除所有的选项，并由自举程序在命令结束时启动系统复位来实施新的选项字节配置。

注：

- 1. 写入 RAM 时，应注意避免与自举程序固件使用的第一个 RAM 存储器发生重叠。
- 2. 在具有写保护的扇区中执行写操作时不会返回错误信息。
- 3. 起始地址无效时也不会返回错误信息。

表 3.6-1. Write Memory 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号  | 数据   | 描述                          |
|-------|-------|------|-----------------------------|
| 主机端   | 1     | 0x31 | Write Memory 命令             |
|       | 2     | 0xCE | 命令补码                        |
| 器件端   | 1     | 0x79 | ACK                         |
| 主机端   | 1~4   | -    | 起始地址（字节 1：MSB，字节 4：LSB）     |
|       | 5     | -    | 校验和：异或（字节 1、字节 2、字节 3、字节 4） |
| 器件端   | 1     | 0x79 | ACK                         |
| 主机端   | 1     | -    | 待写入的字节数 - 1（0 < N ≤ 255）    |
|       | 2~N+2 |      | N + 1 数据字节（最多 256 字节）       |
|       | N+3   | -    | 校验和：异或运算结果（N、N+1 数据字节）      |
| 器件端   | 1     | 0x79 | ACK                         |

2.7 Erase Memory 命令

主机可通过 Erase Memory 命令使用双字节寻址模式擦除 Flash 页面。自举程序接收到 Erase Memory 命令后，会将 ACK 字节发送到主机。发送 ACK 字节之后，自举程序会接收两个字节（待擦除的页数）、Flash 页面代码（采用双字节编码，MSB 在前）和一个校验和字节（已发送字节的异或运算结果）；若校验和正确，自举程序将擦除存储器并向主机发送一个 ACK 字节。否则将发送一个 NACK 字节到主机并中止命令。

Erase Memory 命令规范：

- 1. 自举程序接收一个包含 N（要擦除的页数 - 1）的半字（两个字节）：
  - a)  $N = 0xFFFY$ （Y 范围为 0 到 F）时，执行特定擦除操作：
    - $0xFFFF$ ，执行全局批量擦除
  - b)  $N_{(MSB)} = 0x10$  时执行页擦除操作，针对  $0 \leq N_{(LSB)} < \text{最大页数之间的其它值}$ ：擦除  $N_{(LSB)} + 1$  页。
  - c)  $N_{(MSB)} = 0x20$  时执行扇区擦除操作，针对  $0 \leq N_{(LSB)} < \text{最大扇区数之间的其它值}$ ：擦除  $N_{(LSB)} + 1$  扇区。
- 2. 自举程序：
  - a) 执行特定擦除时，接收一个字节：之前字节的校验和：
    - $0xFFFF$  对应  $0x00$
  - b) 执行  $N_{(LSB)} + 1$  页擦除时，自举程序接收  $(2 \times (N_{(LSB)} + 1))$  字节，每个半字中包含一个页数（采用双字节编码，MSB 在前）。之后将接收所有之前字节的校验和（一个字节）。
  - c) 执行  $N_{(LSB)} + 1$  扇区擦除时，自举程序接收  $(2 \times (N_{(LSB)} + 1))$  字节，每个半字中包含一个扇区数（采用双字节编码，MSB 在前）。之后将接收所有之前字节的校验和（一个字节）。

表 3.3-1. Erase Memory 命令：主机端(PC)、器件端(PY32)

| 主机/器件 | 字节编号      | 数据   | 描述                               |
|-------|-----------|------|----------------------------------|
| 主机端   | 1         | 0x44 | Erase Memory 命令                  |
|       | 2         | 0xBB | 命令补码                             |
| 器件端   | 1         | 0x79 | ACK                              |
| 主机端   | 1         | 0x20 | 扇区擦除识别码（页擦除这里为 0x10）             |
|       | 2         | -    | 待擦除的扇区数 - 1 ( $0 < N \leq 255$ ) |
|       | 3~2*(N+2) |      | 扇区（采用双字节编码）MSB 在前                |
|       | 2*(N+2)+1 |      | 校验和：上述所有已发送字节的校验和                |
| 器件端   | 1         | 0x79 | ACK                              |



3 版本历史

| 版本   | 日期         | 更新记录 |
|------|------------|------|
| V1.0 | 2023-07-10 | 初版   |
|      |            |      |
|      |            |      |
|      |            |      |
|      |            |      |
|      |            |      |
|      |            |      |



Puya Semiconductor Co., Ltd.

**IMPORTANT NOTICE**

Puya Semiconductor reserves the right to make changes without further notice to any products or specifications herein. Puya Semiconductor does not assume any responsibility for use of any its products for any particular purpose, nor does Puya Semiconductor assume any liability arising out of the application or use of any its products or circuits. Puya Semiconductor does not convey any license under its patent rights or other rights nor the rights of others.