

## AT32F407 IAP using ETH

## 前言

本应用笔记专为使用AT32F407微控制器的开发人员编写。它提供了如何使用AT32F407以太网通信接口实现在应用中编程(IAP)的解决方案。

有两种基于LwIP TCP/IP协议栈的解决方案：

- 使用TFTP(简单文件传输协议)的IAP
- 使用HTTP(超文本传输协议)的IAP

支持型号列表：

支持型号	AT32F407xx
------	------------

## 目录

<b>1</b>	<b>IAP 概述.....</b>	<b>5</b>
1.1	工作原理.....	5
1.2	使用 MCU 以太网接口实现 IAP.....	5
1.3	通过以太网在 AT32F407 上实现 IAP .....	5
1.3.1	使用 TFTP 实现 IAP 方法 .....	6
1.3.2	使用 HTTP 实现 IAP 方法.....	6
<b>2</b>	<b>使用 TFTP 实现 IAP.....</b>	<b>7</b>
2.1	TFTP 概述 .....	7
2.2	使用 TFTP 为 AT32F407 实现 IAP .....	8
2.3	使用软件 .....	9
<b>3</b>	<b>使用 HTTP 实现 IAP .....</b>	<b>10</b>
3.1	HTTP 文件上传概述.....	10
3.2	使用 HTTP 在 AT32F407 上实现 IAP .....	11
3.3	使用软件 .....	13
3.4	已知限制 .....	13
3.4.1	二进制文件中添加的额外字节 .....	13
<b>4</b>	<b>环境.....</b>	<b>14</b>
4.1	MAC 地址和 IP 地址设置 .....	14
4.2	软件文件组成.....	14
4.3	构建 IAP 映像 .....	14
<b>5</b>	<b>版本历史 .....</b>	<b>15</b>

## 表目录

表 1	TFTP 操作码 .....	7
表 2	介绍了项目源文件 .....	14
表 3	文档版本历史 .....	15

## 图目录

图 1	IAP 操作流程.....	5
图 2	TFTP 数据包 .....	7
图 3	使用 TFTP 实现 IAP 流程图 .....	8
图 4	TFTP64 对话框 .....	9
图 5	文件上传 HTML 表单的浏览器画面.....	10
图 6	IE11 HTTP 头文件格式 .....	10
图 7	Mozilla Firefox HTTP 头文件格式 .....	10
图 8	登录 web 页面 .....	11
图 9	文件上传完成 web 页面.....	12
图 10	使用 HTTP 实现 IAP 的流程图.....	12

# 1 IAP 概述

## 1.1 工作原理

在应用中编程(IAP)是一种在现场通过 MCU 通信接口（例如 USART、USB、CAN 和以太网）进行固件升级的方式。

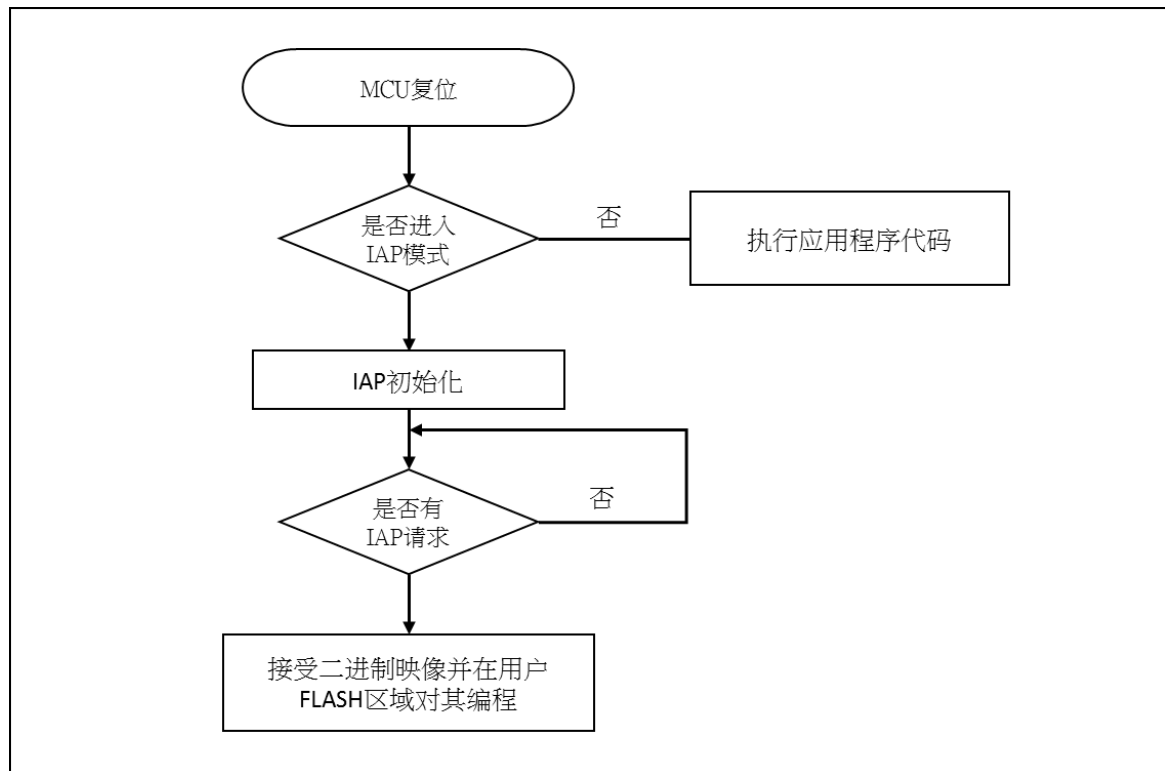
启动微控制器时，可以选择让其在以下任一模式运行：

- IAP 模式，用于执行 IAP 代码
- 正常模式，用于执行应用程序代码

无论是 IAP 代码还是应用程序代码都位于微控制器的内置 FLASH 中，IAP 代码通常存储在 MCU FLASH 的第一页，而用户应用程序代码则占据剩余的 FLASH 区域。

图 1 介绍了 IAP 操作流程

图 1 IAP 操作流程



## 1.2 使用 MCU 以太网接口实现 IAP

如果有以太网可用，则它通常是嵌入式系统中实现 IAP 功能的首选接口，其优势包含：

- 高速通信接口(10/100 Mbps)
- 通过网络(LAN 或 WAN)进行远程编程
- 可以使用 FTP、TFTP、HTTP 等基于 TCP/IP 栈的标准应用协议实现 IAP

## 1.3 通过以太网在 AT32F407 上实现 IAP

本应用笔记将介绍两种使用以太网通信外设在 AT32F407 上实现 IAP 的解决方案：

- 使用 TFTP（简单文件传输协议）的 IAP
- 使用 HTTP（超文本传输协议）的 IAP

这两种解决方案均基于 LwIP 栈(2.1.2)，它是轻量级的 TCP/IP 协议栈

### 1.3.1 使用 TFTP 实现 IAP 方法

使用 TFTP 实现 IAP 的方法广泛应用于需要具有固件升级功能的嵌入式系统应用中（例如，嵌入式 Linux bootloader 中）。

TFTP 是一种在 UDP 传输层上执行的简单文件传输协议。此协议非常适合在局域网环境中使用。它基于客户端/服务器架构，在这种架构中，客户端会向服务器发出文件传输请求（读取或写入操作）。为实现 IAP，需要在 LwIP 协议栈上实现一个简单的 TFTP 服务器，服务器只须处理来自 PC 的 TFTP 客户端的写入请求即可。

### 1.3.2 使用 HTTP 实现 IAP 方法

使用 HTTP 协议进行固件升级没有使用 TFTP 常见，但是在需要通过 Internet 进行远程编程时，这种解决方案就显得极为有用。这时，需要使用 TCP 传输协议来实现 HTTP 服务。

HTTP 基于 TCP 协议运行，它提供了一种以 HTML 表单形式从 Web 客户端(Mozilla Firefox 或 Microsoft Internet Explorer)发送一个二进制文件的方式。这称为 HTTP 文件上传(RFC1867)。

本文档中的后续章节将详细介绍这两种 IAP 方法的实现，并会对如何使用软件进行说明。

## 2 使用 TFTP 实现 IAP

### 2.1 TFTP 概述

TFTP是一种基于UDP的简单文件传输协议。文件传输由TFTP客户端发起，会向TFTP服务器发送读取或写入请求。服务器确认请求后，即开始进行文件数据传输。数据将以固定大小的块尽情发送（例如每块含512个字节）。

必须在每个发出的数据块都得到接收方确认后，才可以发送下一个数据块。这种确认机制通过随各个数据块一同发送的编块号来实现。数据块小于固定块大小表示文件传输的结束。

图2妙数了各种TFTP数据包的格式：

图 2 TFTP 数据包

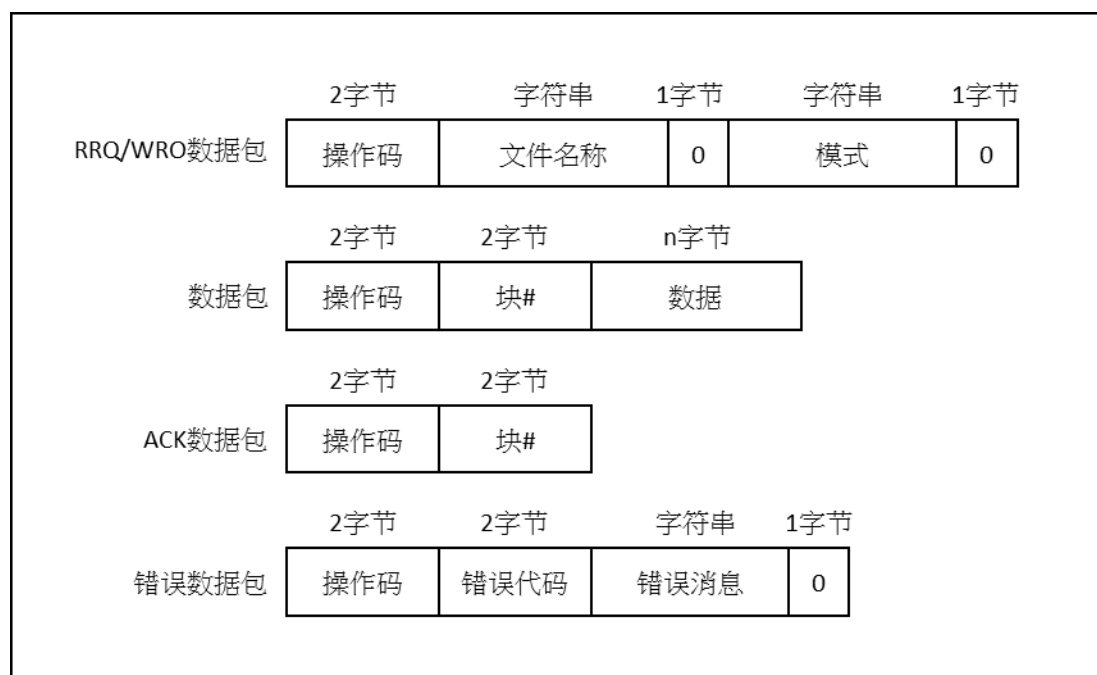


表1列出了TFTP操作码。

表 1 TFTP 操作码

操作码	操作
0x1	读请求(RRQ)
0x2	写请求(WRQ)
0x3	数据
0x4	确认(ACK)
0x5	错误

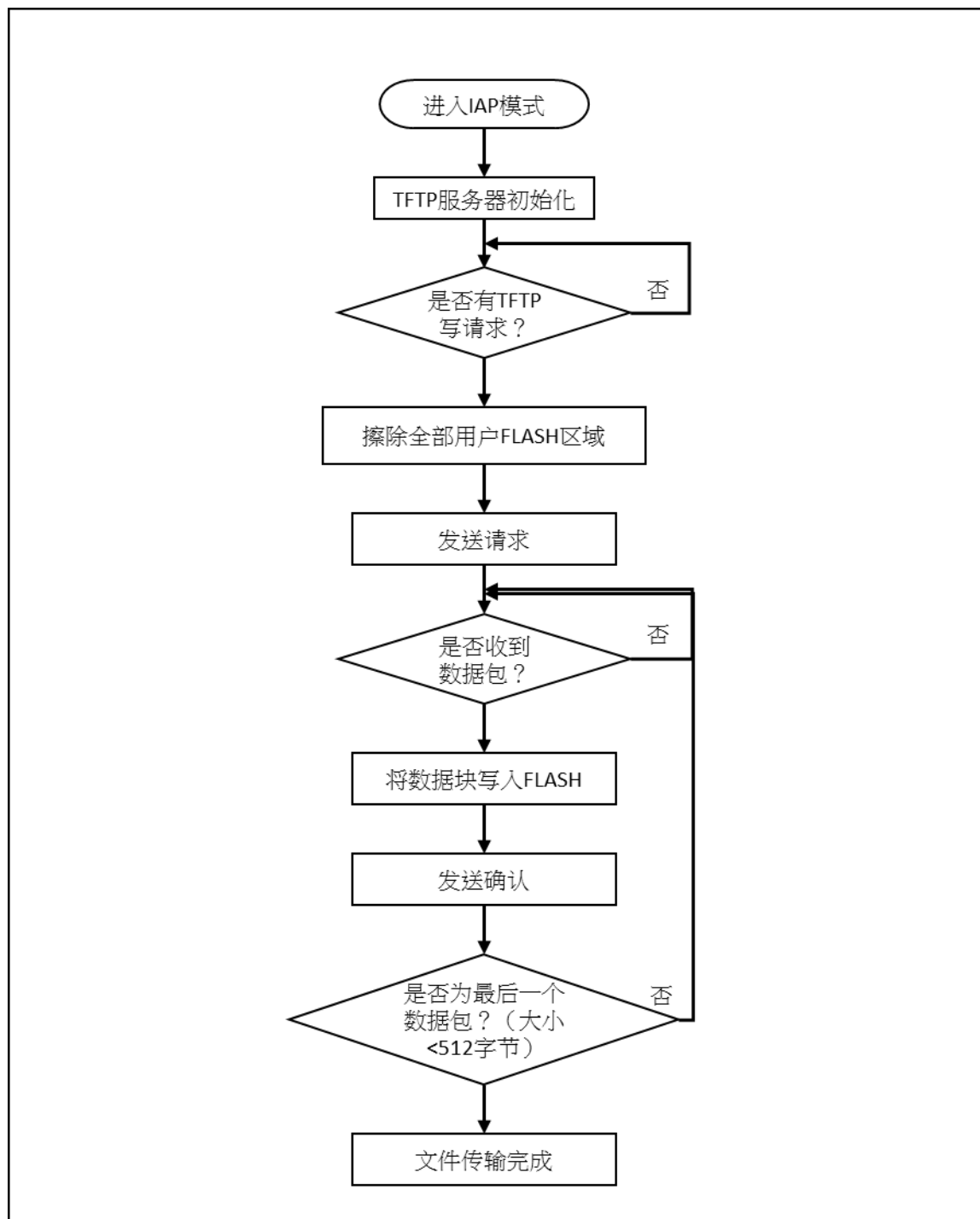
## 2.2 使用 TFTP 为 AT32F407 实现 IAP

此 IAP 实现由基于 LwIP TCP/IP 栈的 TFTP 服务器组成。  
此服务器会对远程 TFTP 客户端(PC)发来的写请求做出响应。  
TFTP 读请求会被忽略。

TFTP 通常会将接收到的文件写入到文件系统，但是该服务器却并非如此，它会将接受到的数据块写入到 MCU FLASH（用户 FLASH 区域中）。

注：在这个实现过程中，数据块大小固定为 512 个字节。  
图 3 概述了使用 TFTP 实现 IAP 操作的过程。

图 3 使用 TFTP 实现 IAP 流程图



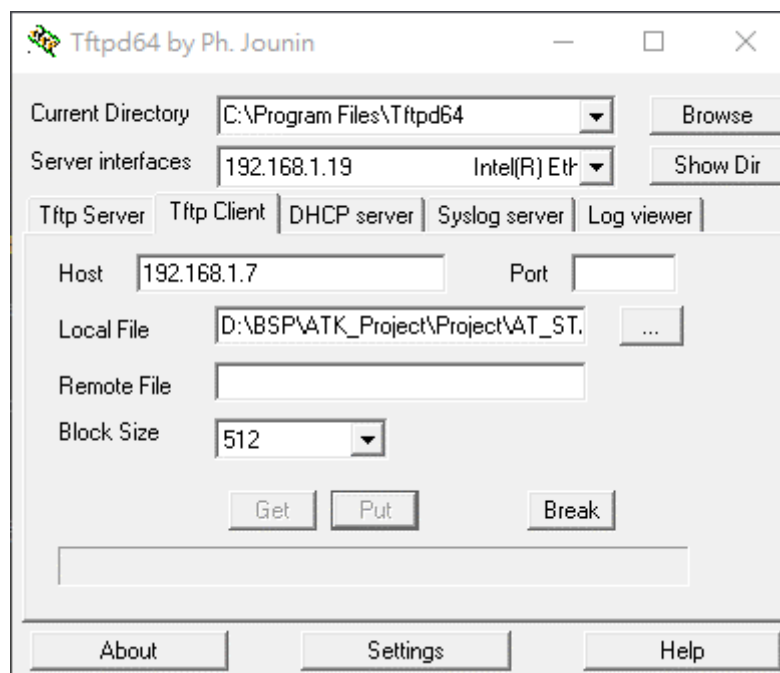


## 2.3 使用软件

要通过 TFTP 对 IAP 进行测试，需执行以下步骤：

1. 在 main.h 文件中，取消 USE\_IAP\_TFTP 选项的注释，另外，还可以根据需要取消注释/注释其他选项，例如 USE\_DHCP。
2. 重新编译软件。使用生成的映射文件，确保 IAP 代码区域之间没有重叠（从地址 0x0 开始），而且用户 FLASH 区域从以下地址开始：USER\_FLASH\_FIRST\_PAGE\_ADDRESS（在 flash.h 中定义）。
3. 在 AT32 FLASH 中编写并运行软件程序。
4. 要进入 IAP 模式，需要在按住按键的同时按下再释放复位按键。
5. 分配完 IP 后（可以是静态或动态地址），用户即可启动 IAP 流程。
6. 在 PC 侧，打开 TFTP 客户端（例如 Tftpd64），然后配置 TFTP 服务器地址（Tftpd64 中的主机地址）
7. 单击 Tftpd64 实用程序中的 Put（写入）按钮，启动文件写请求
8. 在 IAP 操作结束时，可以复位开发板并在 AT32 FLASH 中运行刚刚编写的应用程序

图 4 TFTP64 对话框



## 3 使用 HTTP 实现 IAP

### 3.1 HTTP 文件上传概述

RFC1867 中定义了使用 HTTP 进行文件上传。此文件上传方法是基于 HTTP 表单。发送原始二进制数据时，要使用 HTTP POST 方法而不是 GET 方法。

以下是一个 HTML 代码示例，用于实现基于表单的文件上传：

图 5 文件上传 HTML 表单的浏览器画面

Please specify a binary file to upload into AT32F407 flash:

D:\BSP\ATK\_Project\Project\AT\_START\_F407\ 浏览...

Upload

注：在发送文件数据前，Web 客户端会首先发送 HTTP 头文件数据，其中包含诸如文件名称和内容长度等信息，Web 服务器必须对其中的一些信息进行解析。

Web 客户端使用的 HTTP 头文件格式并不总是相同。图 x 显示的是 Internet Explorer 在 POST 请求中的 HTTP 头文件格式。图 x 显示的是 Mozilla Firefox 的 HTTP 头文件格式。

HTTP Web 服务器必须能够处理这些不同的格式。

图 6 IE11 HTTP 头文件格式

```
▼ Hypertext Transfer Protocol
  > POST /upload.cgi HTTP/1.1\r\n
    Accept: text/html, application/xhtml+xml, image/jxr, */*\r\n
    Referer: http://192.168.1.7/checklogin.cgi\r\n
    Accept-Language: zh-Hant-TW,zh-Hant;q=0.7,ja;q=0.3\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko\r\n
    Content-Type: multipart/form-data; boundary=-----7e42053521056\r\n
    Accept-Encoding: gzip, deflate\r\n
    Host: 192.168.1.7\r\n
    > Content-Length: 2255\r\n
    Connection: Keep-Alive\r\n
    Cache-Control: no-cache\r\n
    \r\n
    [Full request URI: http://192.168.1.7/upload.cgi]
    [HTTP request 1/1]
    File Data: 2255 bytes
  ▼ MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "-----7e42053521056"
    [Type: multipart/form-data]
    First boundary: -----7e42053521056\r\n
    > Encapsulated multipart part: (application/octet-stream)
    Last boundary: \r\n-----7e42053521056--\r\n
```

图 7 Mozilla Firefox HTTP 头文件格式

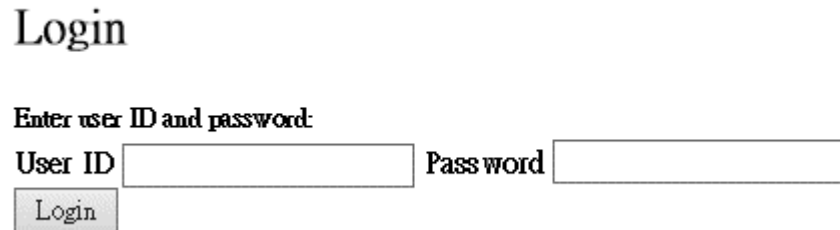
```
▼ Hypertext Transfer Protocol
  > POST /upload.cgi HTTP/1.1\r\n
    Host: 192.168.1.7\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:80.0) Gecko/20100101 Firefox/80.0\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8\r\n
    Accept-Language: zh-TW,zh;q=0.8,en-US;q=0.5,en;q=0.3\r\n
    Accept-Encoding: gzip, deflate\r\n
    Content-Type: multipart/form-data; boundary=-----220078908128723137894205532457\r\n
    > Content-Length: 2289\r\n
    Origin: http://192.168.1.7\r\n
    Connection: keep-alive\r\n
    Referer: http://192.168.1.7/checklogin.cgi\r\n
    Upgrade-Insecure-Requests: 1\r\n
    \r\n
    [Full request URI: http://192.168.1.7/upload.cgi]
    [HTTP request 1/1]
    File Data: 2289 bytes
  ▼ MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "-----220078908128723137894205532457"
    [Type: multipart/form-data]
    First boundary: -----220078908128723137894205532457\r\n
    > Encapsulated multipart part: (application/octet-stream)
    Last boundary: \r\n-----220078908128723137894205532457--\r\n
```

## 3.2 使用 HTTP 在 AT32F407 上实现 IAP

此 IAP 实现由基于 LwIP 栈的 HTTP Web 服务器组成。

在浏览器中输入的 AT32 的 IP 地址后，将显示登录 Web 页面（图 x）。此登录 Web 页面只有已获授权的用户才能使用 IAP 文件上传功能。

图 8 登录 web 页面

The image shows a web browser window displaying a login page. At the top, the word "Login" is centered in a large, serif font. Below it, the text "Enter user ID and password:" is displayed. Underneath this text are two input fields: "User ID" and "Password". The "User ID" field contains the text "user" and the "Password" field contains the text "at32". Below the input fields is a button labeled "Login".

- 注：
1. 默认的用户 ID（用户 ID）为 user, Password（密码）为 at32
  2. 如果 User ID（用户 ID）或 Password（密码）不正确，登录 Web 页面会重新加载。

登录成功后，浏览并选择要上传到 AT32 FLASH 的二进制文件

- 注：
- 确保二进制文件大小不超过 AT32 用户 FLASH 区域的总容量。

单击 Upload（上传）按钮后（参见图 x），将向服务器发出 POST 请求。这时，服务器开始擦除用户 FLASH 区域的全部内容，等待接受二进制文件原始数据。然后将接收到的数据写入用户 FLASH 区域。

注意，要接收的数据总长度信息将从传输开始时发出的 HTTP 头文件数据中提取。

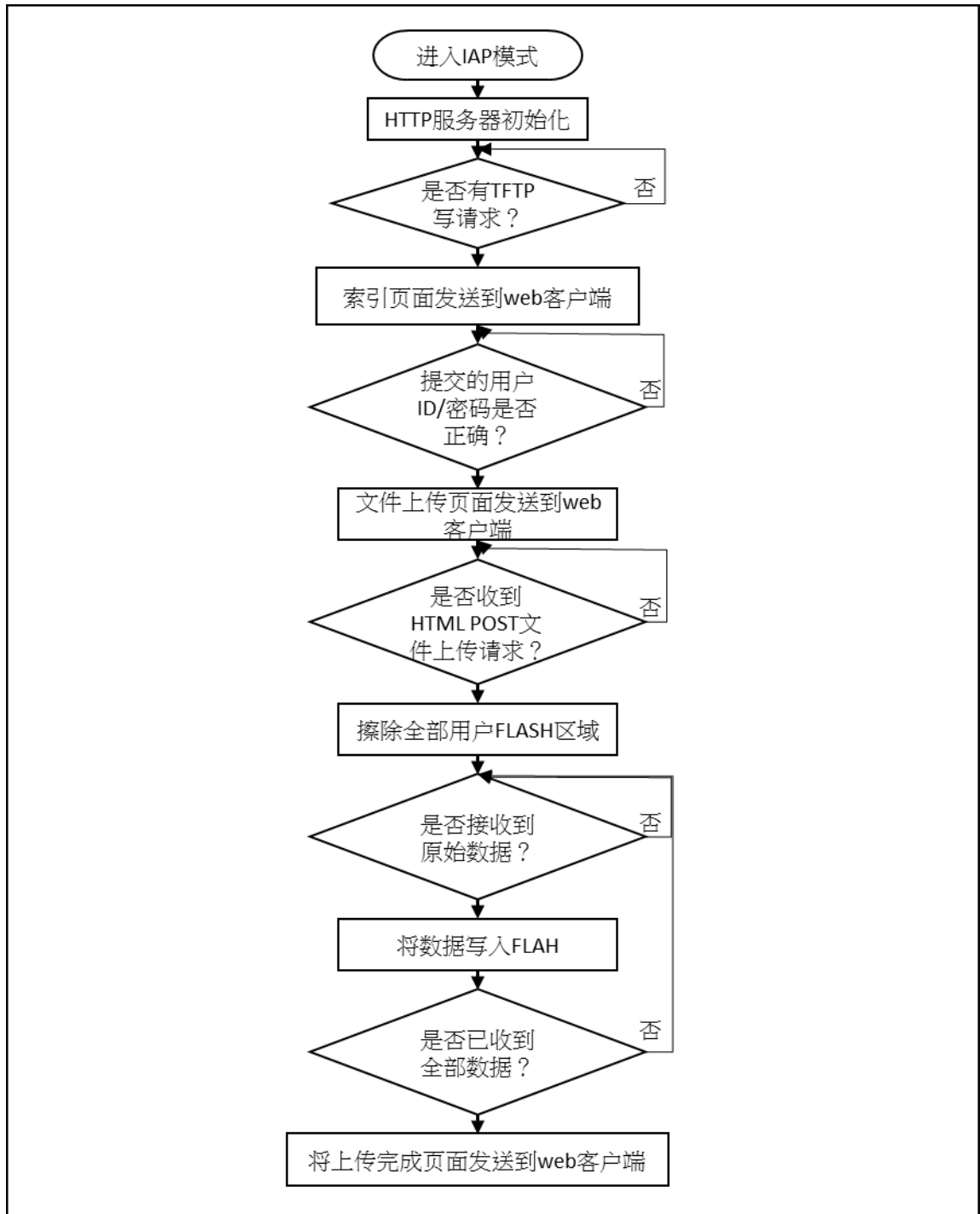
在 IAP 操作结束后，Web 页面将只是 IAP 操作成功，同时显示一个可用于复位 MCU 的按钮。

图 9 文件上传完成 web 页面



图 10 对使用 HTTP 实现 IAP 方法进行了总结

图 10 使用 HTTP 实现 IAP 的流程图



## 3.3 使用软体

要使用 HTTP 对 IAP 进行测试，需执行以下步骤：

1. 在 main.h 文件中，取消选项 USE\_IAP\_HTTP 的注释，另外还可以根据需要取消注释/注释其他选项，例如 USE\_DHCP。
2. 重新编译软件，使用生成的映射文件，确保 IAP 区域代码之间没有重叠（从地址 0x0 开始），而且用户 FLASH 区域从以下地址开始：USER\_FLASH\_FIRST\_PAGE\_ADDRESS（在 flash.h 中定义）。
3. 在 AT32 FLASH 中编写并运行软件程序。
4. 要进入 IAP 模式，需要再按住按键按钮的同时按下再释放复位按钮。
5. 分配完 IP 地址后（可以是静态或动态地址），用户即可启动 IAP 流程
6. 打开 Web 客户端（Mozilla Firefox 或 Internet Explorer），输入 AT32 IP 地址
7. 会显示登录 Web 页面。在 User ID（用户 ID）字段中输入“user”，在 Password（密码）字段中输入“at32”，然后按下 Login（登录）按钮。
8. IP 操作结束后，将加载新的 Web 页面，只是文件上传操作已经成功完成。
9. 可以按下 Reset MCU（复位 MCU）按钮复位 MCU，然后在 AT32 FLASH 中运行刚刚编写的应用程序。

注：使用以下 Web 客户端对软件进行测试：Microsoft Internet Explorer 11 和 Mozilla Firefox 80.0

## 3.4 已知限制

### 3.4.1 二进制文件中添加的额外字节

Internet 浏览器（Microsoft Internet Explorer 或 Mozilla Firefox）会在上传的二进制文件的末尾添加一个随机边界标记（根据 RFC1521 规定，此标记不得超过 72 个字节）。在最新的 IAP 软件版本中，并没有删除此边界标记，而是在空间足够的情况下将其存储在 FLASH 中。如果没有足够空间，则不会在 FLASH 中写入额外字节，也不会返回错误。

## 4 环境

### 4.1 MAC 地址和 IP 地址设置

在 netconf.h 文件中对 MAC 地址进行了定义。

默认的 MAC 地址固定为：00:00:44:45:56:01。

在 main.h 文件中对 IP 地址进行了定义。

IP 地址可以设置为静态地址，也可以设置为由 DHCP 服务器分配的动态地址。默认的静态地址为：192.168.1.7。

可以通过在 lwipopts.h 文件中使能 LWIP\_DHCP 来选择 DHCP 模式。

### 4.2 软件文件组成

表 2 介绍了项目源文件

文件名称	说明
main.c	主应用程序文件
main.h	主配置文件
httpserver.c/.h	HTTP 服务器实现
tftpserver.c/.h	TFTP 服务器实现
flash.c/.h	高级别 FLASH 访问函数
netconf.c/.h	高级别以太网接口函数
at32f4xx_eth.c/.h	AT32F4xx 以太网硬件配置
at32f4xx_it.c/.h	中断处理程序
fsdata.c	作为 ROM 文件系统的 HTML 文件
lwipopts.h	LwIP 配置选项

注：表格中没有列出标准固件库和 LwIP 栈中所使用的文件。

### 4.3 构建 IAP 映像

为了构建 IAP 映像（将会使用 IAP 软件加载），应确保以下几点：

1. 编译/链接的软件必须从用户 FLASH 区域的起始地址开始运行（此地址应与 flash.h 的 USER\_FLASH\_FIRST\_PAGE\_ADDRESS 中所定义地址相同）。
2. 将向量表的起始地址配置为用户 FLASH 区域的起始地址：
  - A. 在应用程序代码中，使用 misc.h/.c 驱动程序的 NVIC\_SetVectorTable 函数来重新定位应用程序加载地址的向量表。  
例如，将向量表基本位置设置为 0x08010000：  
NVIC\_SetVectorTable(NVIC\_VectTab\_FLASH, 0x10000);
  - B. 通过修改 system\_at32f4xx.c 文件中定义的 VECT\_TAB\_OFFSET 常量的值。  
例如，将向量表基本位置设置为 0x08010000：  
#define VECT\_TAB\_OFFSET 0x10000
3. 编译后的软件大小不超过用户 FLASH 区域的总容量。

## 5 版本历史

表 3 文档版本历史

日期	版本	变更
2020.09.01	1.0.0	最初版本

**重要通知 - 请仔细阅读**

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和/或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途(及其依据任何司法管辖区的法律的对应情况)，或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：(A) 对安全性有特别要求的应用，如：生命支持、主动植入设备或对产品功能安全有要求的系统；(B) 航空应用；(C) 汽车应用或汽车环境；(D) 航天应用或航天环境，且/或(E) 武器。因雅特力产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险由购买者单独承担，并且独力负责在此类相关使用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和/或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。

© 2020 雅特力科技 (重庆) 有限公司 保留所有权利