# lwIP官网介绍

## 前言

本文对lwIP官网做简要的介绍

## 主页简介

lwIP的代码已经交由Savannah托管，项目主页即lwIP官网

地址：

http://savannah.nongnu.org/projects/lwip/

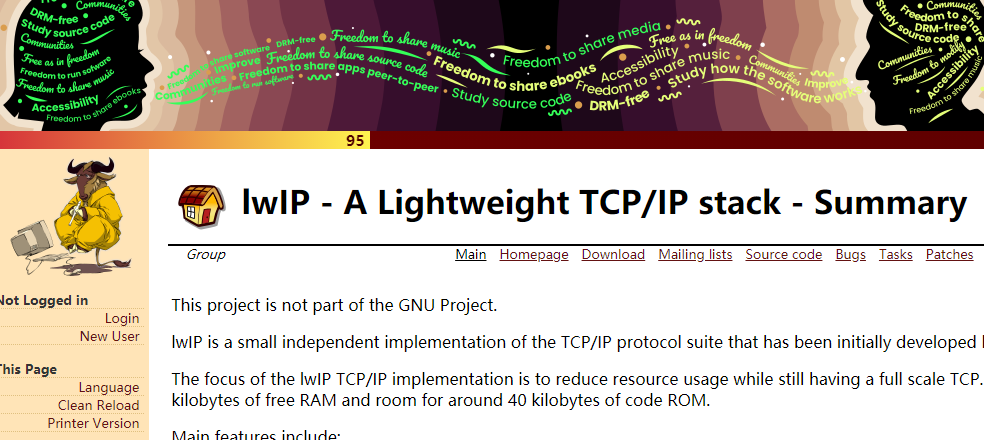


图2-1 LwIP项目主页

## 子网页Project Homepage

单击Project Homepage，会打开一个网页，如图3-1所示。这个网页可以看作lwIP的官方说明文档。我们可以通过这个网页获得关于lwIP的很多信息，包括使用lwIP的注意事项、数据的复制、系统初始化流程、多线程中要注意的问题、优化方法、内核模块的分类介绍、内核数据结构、内核重要全局变量、内核源码文件等。这些内容的专业性比较强，不建议初学时在这方面花费精力，并且里面的很多内容在本书的后续章节中会有所讲解，目前只需要了解即可。

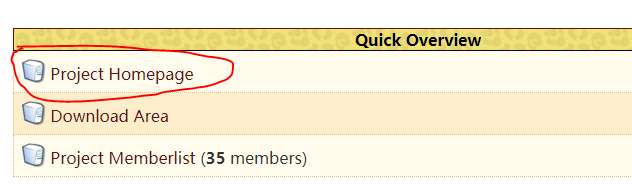


图3-1 Project Homepage链接

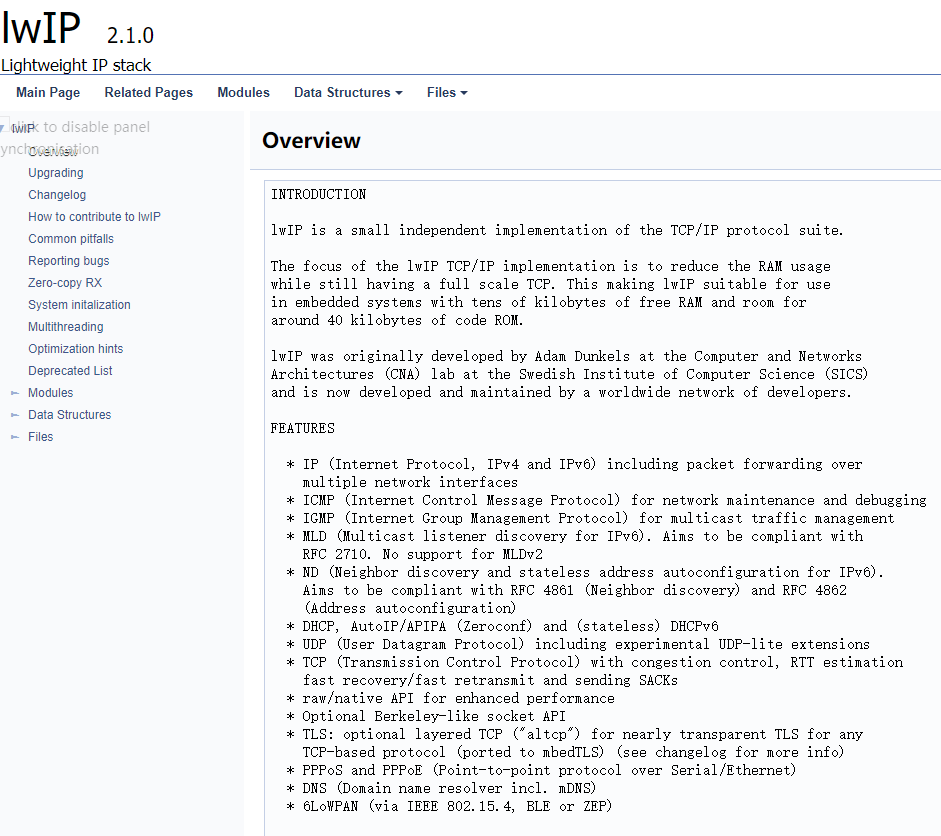


图3-2 lwIP官方说明文档

## 子网页Download Area

### 网页说明

单击Download Area打开的网页如图4-1所示。通过这个网页，我们可以下载lwIP所有版本的源代码包和contrib包。每单击一个红色字体的资源链接，浏览器就会打开一个ftp连接，帮助你下载想要的文件。但是这个页面提供的下载链接在国内一般无法打开。这个网页最下方的黑字内容推荐我们使用另外一个下载页面：http://download-mirror.savannah.gnu.org/releases/。在这个页面下，用户可以下载到所有在Savannah托管的开源软件，但我们只关心LwIP。利用浏览器的搜索功能，按Ctrl+F快捷键可以快速找到lwip目录。在这里为了便于读者下载，我们直接给出最终的下载链接http://download-mirror.savannah.gnu.org/releases/lwip/。

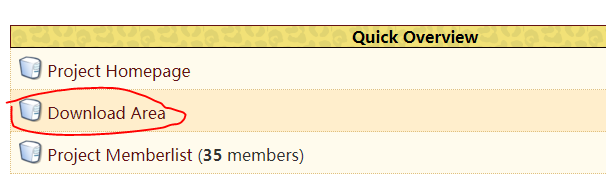


图4-1 Download Area链接

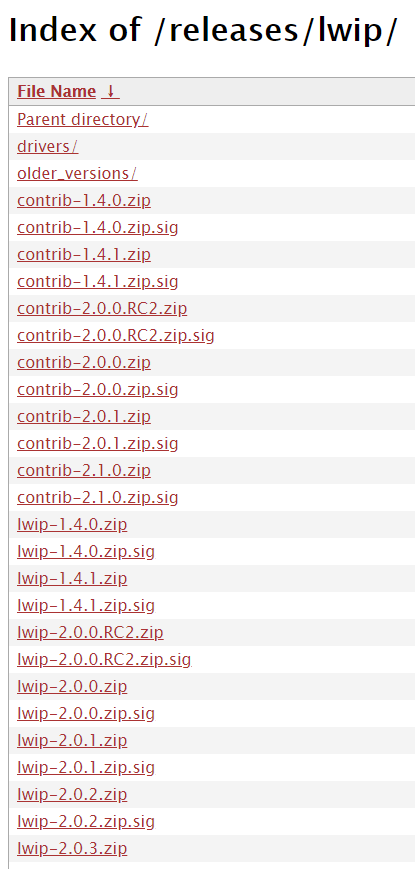


图4-2 LwIP的资源下载

### lwip-x.x.x.zip

源代码包中装的主要是LwIP内核的源码文件

### contrib包

contrib包中装的是移植和应用LwIP的一些demo，即应用示例。contrib包不属于LwIP内核的一部分，里面的很多内容来自开源社区，因此对contrib包的版本管理不像内核源码那样严格和规范，但contrib包也是很有参考价值的。LwIP源码面世越久，开源社区对它的贡献就越大，所以越高版本的contrib包，提供的应用示例就越丰富，越有参考价值。在大版本区别不大的情况下，建议下载最新的contrib包。后文中我们会对contrib包中提供的应用示例进行讲解。

### sig文件

些后缀为．sig的文件，这些文件是数字签名，忽略即可。

## lwIP源码

我们下载两个包：lwip-2.1.3.zip（源码包）和contrib-2.1.0.zip（contrib包）。解压以后会得到两个文件夹，如图5-1所示。



图5-1 下载解压后得到的源码包和contrib包

### lwip-2.1.3目录

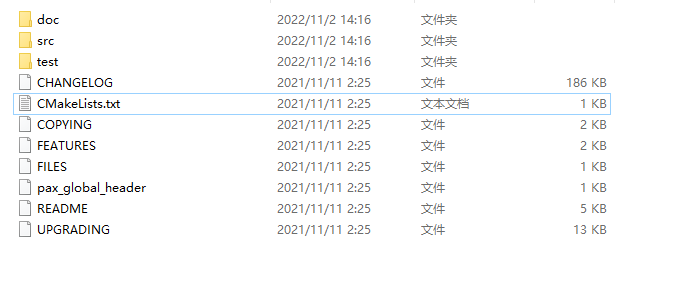


图5-2 源码包的目录

#### 文件

（1）CHANGELOG 文件记录了 LwIP 在版本升级过程中源代码发生的变化。

（2）CMakeLists.txt编译规则文件

（3）COPYING 文件记录了 LwIP 这个开源软件的 license。一个软件开源，不代表你能无限制地使用它，你需要在使用它的过程中遵守一定的规则，这些规则就是 license。大家可以用记事本打开这个 COPYING 文件看看它的内容。开源软件的 license 有很多种， LwIP 的属于 BSD License。LwIP 的开源程度是很高的，你几乎可以无限制地使用它。

（4）FEATURES特性说明

（5）FILES 文件用于介绍当前目录下的目录信息。

（6）pax\_global\_header包含git commit ID 的 extended header ，如果你的tar版本较旧，就会出出现一个警告信息，进而产生一个pax\_global\_header的文件

（7）README 文件对 LwIP 进行了一个简单的介绍。

（8）UPGRADING 文件记录了 LwIP 每个大版本的更新，会对用户使用和移植 LwIP 造成的影响。所谓大版本更新指的是：1.3.x - 1.4.x –2.0.x –2.1.x。小版本更新，比如2.0.1 –2.0.2 –2.0.3，这个过程只是一些bug的修复和性能的改善，不会对用户的使用造成影响。用户只要将原有工程的目录中与 LwIP 相关的旧版本文件替换成新版本的文件，重新编译，就能直接使用。

#### 文件夹

（1）doc文件夹里面是关于LwIP的一些文档，可以看成是应用和移植 LwIP 的指南。但是这些文档比较零散，不成体系，而且纯文本阅读起来很费劲，阅读意义不是很大。

（2）test文件夹里面是测试LwIP内核性能的源码，将它们和LwIP源码加入到工程中一起编译，调用它们提供的函数，可以获得许多与LwIP内核性能有关的指标。这种内核性能测试功能，只有非常专业的人士才用的到。

（3）src文件夹里面就是我们最关心的LwIP源码文件。

### src目录

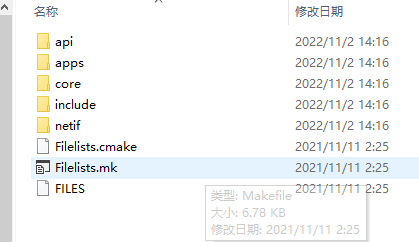


图5-3 src目录（LwIP源码文件所在的目录）

### core目录

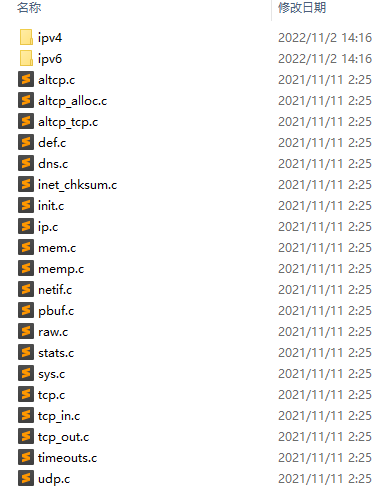


图5-4 core目录

ipv4 文件夹里面是与 IPv4 模块相关的源文件，它们实现了 IPv4 协议规定的对数据包的各种操作。ipv4 文件夹中还包括一些并非属于 IP 协议，但会受 IP 协议影响的协议源文件，包括 DHCP、ARP、 ICMP、 IGMP。

ipv6 文件夹里面是与 IPv6 模块相关的源文件，它们实现了 IPv6 协议规定的对数据包的各种操作。ipv6 文件夹中还包括一些并非属于 IP 协议，但会受 IP 协议影响的协议源文件，包括 DHCP、ARP、 ICMP、 IGMP。

altcp.c、 altcp\_alloc.c、 altcp\_tcp.c 等文件是应用程序分层 TCP 连接 API，从 TCPIP 线程使用，是一个抽象层，可以模拟应用程序的 tcp 回调 API，同时防止直接链接，这样，应用程序可以使用其他应用程序层协议在 TCP 之上而不知道细节（例如 TLS，代理连接），此类接口我们并没有怎么使用，或者如果选择使用安全的加密传输的话，可以配合 mbed TLS 使用。

def.c 文件定义了一些基础类函数，比如主机序和网络序的转换、字符串的查找和比较、整数转换成字符串等，这些函数会被 LwIP 内核的很多模块所调用。在 include 目录里面的 def.h 文件对外声明了 def.c 所实现的函数，同时定义了许多宏，能实现一些基础操作，比如取最大值、取最小值、计算数组长度等，这些宏同样也被内核的许多模块所调用。我们经常可以看到某个内核的源文件在开始的地方 #include “def.h”。

dns.c 文件实现了域名解析的功能，有了它，用户就可以在知道服务器域名的情况下，获得该服务器的 IP 地址。很多时候我们只记得服务器域名而不记得服务器IP地址，例如“www.baidu.com”就是一个域名，通过 dns 功能，我们就可以得到与服务器域名对应的 IP 地址，这给用户使用带来很大的方便。

inet\_chksum.c 文件提供了 LwIP 所需的校验和功能，在 IP、 UDP、 TCP 协议的实现中，需要计算校验和。

init.c 文件对 LwIP 的用户宏配置进行了检查，会将配置错误和不合理的地方，通过编译器的 #error和 #warning 功能表示出来。另外， init.c 定义了 lwip\_init 初始化函数，这个函数会依次对 LwIP 的各个模块进行初始化。

ip.c 文件实现了 IP 协议相关的函数，但只是封装了 ipv4 和 ipv6 文件夹中的函数。

mem.c 文件实现了动态内存池管理机制，使得 LwIP 内核的各个模块可以灵活地申请和释放内存。

memp.c 文件实现了静态内存堆管理机制，使得 LwIP 内核的各个模块可以快速地申请和释放内存。

netif.c 文件实现了网卡的操作，比如注册/删除网卡、使能/禁能网卡、设置网卡 IP 地址等等。 netif.c与 include 目录中的 netif.h 文件共同构成了 LwIP 的 netif 模块，它对网卡进行了抽象，使得 LwIP内核可以方便地管理多个特性各异的物理网卡。

pbuf.c 文件实现了 LwIP 对网络数据包的各种操作。网络数据包在 LwIP 内核中以 pbuf 结构体的形式存在，这提高了 LwIP 内核对数据包处理效率，以及提高了数据包在各层之间递交的效率。pbuf 结构体也是我们使用 RAW/Callback API 进行网络应用程序开发的关键。

raw.c 文件实现了一个传输层协议的框架，我们可以在它的基础上修改和添加代码，实现自定义的传输层协议，与 UDP/TCP 一样，它可以与 IP 层直接进行交互。这类似 RAW Socket。在实际的应用中，我们常用 UDP 和 TCP 作为传输层协议。但有时，底层网络开发人员会嫌 UDP的可靠性太差，或者 TCP 虽然可靠性强，但是很耗费时间和内存，他们需要根据实际需求，平衡利弊，定义自己的传输层协议。 LwIP 的 raw 模块可以满足他们的需求。

stat.c 文件实现了 LwIP 内核的统计功能，使用户可以实时地查看 LwIP 内核对网络数据包的处理情况。

sys.c 文件构成了 LwIP 的 sys 模块，它提供了与临界区相关的操作。

tcp.c、 tcp\_in.c 和 tcp\_out.c 文件实现了 TCP 协议，包括对 TCP 连接的操作、对 TCP 数据包的输入输出操作和 TCP 定时器，它们和 include 目录中名称带 tcp 的头文件共同构成了 LwIP 的 TCP 模块。 TCP 模块的实现是 LwIP 的最大特点，它以很小的资源开销几乎实现了 TCP 协议中规定的全部内容。 TCP 协议是非常复杂的协议，这几个与 TCP 模块相关的文件占据了 LwIP 内核的绝大部分。

timeouts.c文件 定义了 LwIP 内核的超时处理机制。 LwIP 内核中多个模块的实现需要借助超时处理机制，包括 ARP 表项的时间统计、 IP 分片报文的重装、 TCP 的各种定时器、实现各种应用层协议需要的超时处理。

udp.c 文件实现了 UDP 协议，包括对 UDP 连接的操作和 UDP 数据包的操作。

### contrib文件夹

打开之前下载好的contrib-2.1.0文件夹，如图5-5所示（后面LwIP的基础例程主要直接使用或参考源码里的示例即可）。

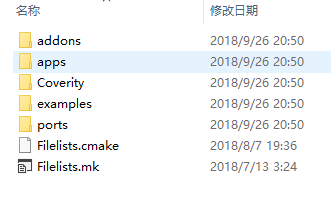


图5-5 contrib包中的文件和文件夹

（1） addons 目录。 LwIP 中很多模块的实现，都是可以由用户干预的，比如校验和、 TCP 初始序列号。 LwIP 的内核代码，通过宏编译选项的设置，可以将内核中某些模块的实现方法配置成 LwIP默认的方法，或者用户自定义的方法。用户自定义的方法通常需要用户在钩子函数中实现。在实际应用中，我们采用内核默认的方法就足够了，只有在非常特定的场合下，为了性能、资源开销等因素的考虑，我们可能会需要自己实现相关的模块，或者说编写相应的钩子函数。

（2） apps 目录里实现了很多应用层协议。 LwIP 源码包中也有 apps 目录，但源码包中 apps 目录下的应用程序全部用 RAW/Callback API 实现，属于内核代码的一部分。而此 apps 目录里的应用程序可以是由三种 API 中的任何一种实现的。读者可以把它看成是内核源码所提供的应用程序的一个补充。

（3） examples 目录里是一些 LwIP 的应用示例。在使用 LwIP 开发应用程序时会出现的典型问题，比如如何移植网卡、如何使用 LwIP 的 API、如何使用源码中提供的应用程序，对于这些问题，这个目录为我们提供了参考。我们在后续的章节中，会使用这个目录中的例子来讲解 LwIP 的应用程序。

（4） ports 目录里是一些移植文件，它可以帮助我们将 LwIP 移植到某个具体的操作系统中。目前这个目录所提供的移植文件，支持 RTOS、 UNIX、 Win32。

## 总结

## 参考资料

版权声明：本文参考了其他CSDN博主的文章，遵循CC 4.0 BY-SA版权协议，现附上原文出处链接及本声明。

1. https://blog.csdn.net/as480133937/article/details/123740365
2. https://blog.csdn.net/weibo1230123/article/details/80210097
3. LwIP应用开发实战指南：基于STM32