TCPIP

编写：张前福

该文档适用于基于DJYOS的TCPIP做二次开发人员，也适用于继续开发DJYOS的TCPIP的人员。

CPU：cortex-m7

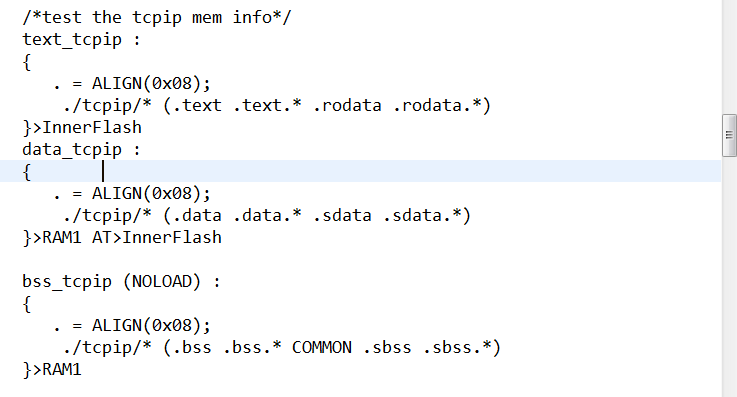
编译器：yagarto4.9(gcc)

优化：O0级，即不优化。

# TCPIP资源消耗情况

## 静态资源消耗

关于静态资源，在编译链接阶段已经确立的ROM RAM等，即我们常说的text段 data段bss段。由于DJYOS采用使用就编译的手法，因此只要使用DJYOS的TCPIP，即相当于将整个协议栈编译链接进来（开发人员可以手动修改tcpipmain.c文件以达到进一步裁剪的目的），因此我们测试的是整个协议栈的资源使用情况。我们修改lds如下：



编译完后查看map文件的这三个存储段的大小，即可得到我们的TCPIP静态资源的占用情况：







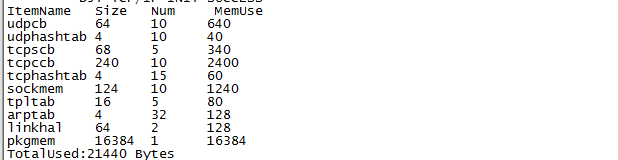
即：整个协议栈（包括FTP TFTP SNTP TELNET等所有网络服务以及各种用于调试的shell）占用的静态资源如下：

|  |  |
| --- | --- |
| 项目 | 占用资源(字节) |
| 代码段（包含程序段和CONST数据） | 0x18610 |
| data段（初始化的数据） | 0x30a |
| Bss段（未初始化的数据） | 0xed0 |

## 动态资源占用情况

在TCPIP协议初始化过程中，会根据用户的配置从系统申请内存用作协议栈的内部运行：

申请的内存主要有以下几项：ARP表 ,TCP控制块，UDP控制块，TCPhash查找表，UDPhash查找表，套接字数据结构，当系统运行起来后，可以使用tcpipmem查看。如下：



待优化的地方就是我们的信号量和互斥体，这两个数据结构有点大（44Byte）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 配置项目 | 数量 | 内存占用bytes | 配置常量 | 简述 |
| Udpcb | 10 | 640 | gUdpSockNum | 可以同时存在的UDP套接字数目 |
| Udphashtab | 10 | 40 | gUdpHashLen | Udphash查找表长度 |
| TCPSCB | 5 | 340 | gTcpServerNum | 可以同时存在的tcp server套接字个数 |
| TCPCCB | 10 | 2400 | gTcpClientNum | 可以同时存在的tcp client套接字个数 |
| SOCKMEM | 10 | 1240 | gSockNum | 可以同时存在的套接字个数 |
| Arptab | 32 | 80 | gArpItemHashLen | Arp hash查找表长度 |
| PKGMEM | 1 | 16384 | gNetPkgMemSize | 收发缓冲区大小，可减小 |
| 以下不可配置 |  |  |  |  |
| Tcphashtab | 15 | 60 |  |  |
| Tpltab | 5 | 128 |  |  |
| Linkhal | 64 | 128 |  |  |
| 总计 |  | 21440 |  |  |

在做基于TCPIP的二次开发的时候，基本上使用的都是套接字，比方SNTP TFTP FTP TELNET等等应用，那么对于一个套接字究竟占用多少资源呢？

以TCP（客户端）为例：

Tcpclientsocketfd = socket空间（124）+ tcp\_ccb(240) = 364Byte

以Udp为例：

Udpsocketfd = socket空间（124）+ udp\_cb(64) = 188Byte

Socket套接字空间稍大，这个是因为后续即将支持IPV6，一个IPV6四元组空间占用就是（32+2）\*2=64Bytes

TCP的CCB占用也稍大，这个是因为我们并发的支持套接字的同时读写，那么用到的同步量就是三个（44\*3=132Bytes），同时窗口缩放、定时重传、保活机制、流量监控等等机制也占用部分空间，才让我们的CCB稍大。

话说明码标价，用户可以根据自己的需要私人订制，额外不再收取费用（资源）！

另外：TCPIP中目前使用了三个任务：tcp的TICK任务，用于处理TCP的各种定时，ipprocess，用于处理接收的IP报文，ARPprocess,用于处理接收的ARP报文。该三个任务的栈空间以及优先级都是可以配置的（详见tcpipconfig.c），之所以可配置，根绝实际情况调整。