Appendix 1

メモリ・サイズの小さいTCP/IPプロトコル・スタック

uIPの概要と移植

関連データ

横田 敬久

TOPPERS/JSPは，第4章で解説したTINETを標準のTCP/IPプロトコル・スタックとしています．しかしTINET未チューニング状態で，付属ColdFire基板マイコンのメモリ環境で動作させるのは少々つらいものがあります．

TINET以外の組み込み向けで，少ないメモリでも動作するTCP/IPプロトコル・スタックにuIPがあります．TOPPERS向けに作られたものではありません．こちらも移植してみましょう．

uIPはSwedish Institute of Computer ScienceのAdam Dunkels氏によって開発された8ビット/16ビット・マイコン向け組み込み用TCP/IPプロトコル・スタックです[注1]．TINETと同じようにIPv4，IPv6に対応しながらも非常に小さいサイズです．uIP自体はマイコン向けに作られているわけですが，標準添付のデバイス・ドライバとしてUNIX系OSやLinuxで動作するTAPデバイスを扱ったものもあり，移植性に優れているのが特徴です．また，前章で紹介したTINETのような，μITRON専用のTCP/API仕様ではなく，UNIXのソケット・インターフェースに近いAPIを持っています．

● TOPPERS対応とuIP対応のデバイス・ドライバ

uIP対応のデバイス・ドライバは，以下のものを移植することで実現可能となります．

- ネットワーク・デバイス初期化関数 fecdev_init()
- ネットワーク・フレーム受信関数 fecdev_read()
- ネットワーク・フレーム送信関数 fecdev_send()
- クロック・時間関係の関数 clock_time()

uIPは割り込みによるイベントでの送受信ができないような環境でも動作可能なように作られています．パケットの送受信もそれぞれが異なったタスクで動作しているわけではなく，メイン・ループでフレームの送受信関数の処理をしてパケットの処理を実現しています．

基本的なEthernetコントローラ(FEC)の初期化ルーチンはTINETのドライバとほぼ同じです．大きな違いはfecdev_send，fecdev_read共に割り込みを用いずに実装できることです．MACアドレスとエンディアンの設定はuip_conf.hで行います(**リスト1**)．

● uIPのアプリケーションの基本構造

uIPはマルチタスク(マルチスレッド)のシステムとシングル・タスク(シングル・スレッド)のシステムの両方で動作可能です．両方に対応するため，標準状態ではmain関数によるメイン・ループ内でパケットのやり取りを行います．

それでは，uIPをTOPPERSで実装した例を見てみましょう．**リスト2**のソースの例では，タイマ関係やネットワーク・デバイスの初期化，uIPの初期化とアドレスのセットを行った後，while(1)でフレームの送受信を行います．

● uIPのディレクトリの構造

今回移植したuIPは，TOPPERS/JSP上では**図1**のようなディレクトリ構成になっています．uIPのアプリケーションはuip/appディレクトリ以下に配置されます．サンプルのアプリケーションとしてSMTPサーバやWebサーバ，telnetサーバなどが用意されている点もTINETとほぼ同じです．

● uIPのアプリケーションの仕組みとプロト・スレッド

uIPのアプリケーションはメイン・ループ内部で動作します．そのため，uIPは「プロト・スレッド」というスレッド・ライブラリ風の仕組みを持っています．プロト・スレッドのアプリケーションでは，プロト・ソケット・ライブラリを使用します．プロト・ソケット・ライブラリはプロト・スレッドで実装されています．プロト・ソケットをスレッド内で使用している間はプロト・スレッドの関数はPT_THREAD以外は特に意識する必要はないでしょう．

uIPのAPI一覧を**表1**に，プロト・ソケットのAPI一覧を**表2**

リスト1　MACアドレスの設定部分

```
/*!
 *  uIPで使用するMACアドレス
 *
 */
#if defined(SILENT) || defined (SILENT_STUB)
#define UIP_ETHADDR0 ((UB)sil_reb_mem((VP)0x000000F8))
#define UIP_ETHADDR1 ((UB)sil_reb_mem((VP)0x000000F9))
#define UIP_ETHADDR2 ((UB)sil_reb_mem((VP)0x000000FA))
#define UIP_ETHADDR3 ((UB)sil_reb_mem((VP)0x000000FB))
#define UIP_ETHADDR4 ((UB)sil_reb_mem((VP)0x000000FC))
#define UIP_ETHADDR5 ((UB)sil_reb_mem((VP)0x000000FD))
#else
#define UIP_ETHADDR0 0x00
#define UIP_ETHADDR1 0x1F
#define UIP_ETHADDR2 0xE8
#define UIP_ETHADDR3 0x49
#define UIP_ETHADDR4 0xFF
#define UIP_ETHADDR5 0xFF
#endif /**/
```

注1：http://www.sics.se/~adam/uip/index.php/Main_Page

```
sample_uip/
uip/app
uip/uip
uip/lib
uip/toppers
uip/unix
uip/README
```

図1
uIPのディレクトリ構成

KEYWORD ——uIP，TCP/IP，TOPPERS，プロト・スレッド

リスト2　uIPをTOPPERSで実装した例

```
void
main_task(VP_INT exinf)
{
  int i;
  uip_ipaddr_t ipaddr;
  struct timer periodic_timer, arp_timer;
  vmsk_log(LOG_UPTO(LOG_INFO), LOG_UPTO(LOG_EMERG));
  syslog(LOG_NOTICE, "Sample program starts (exinf = %d).",
exinf);

  syscall(serial_ctl_por(TASK_PORTID,
          (IOCTL_CRLF | IOCTL_FCSND | IOCTL_FCRCV)));

  timer_set(&periodic_timer, CLOCK_SECOND / 2);
  timer_set(&arp_timer, CLOCK_SECOND * 10);
  fecdev_init();
  uip_init();

  uip_ipaddr(ipaddr, 192,168,1,12);
  uip_sethostaddr(ipaddr);
  uip_ipaddr(ipaddr, 192,168,1,11);
  uip_setdraddr(ipaddr);
  uip_ipaddr(ipaddr, 255,255,255,0);
  uip_setnetmask(ipaddr);

  httpd_init();

～省略～

  smtp_configure("localhost", ipaddr);
  SMTP_SEND("adam@sics.se", NULL, "uip-testing@example.com",
        "Testing SMTP from uIP",
        "Test message sent by uIP\r\n");*/

  /*
    webclient_init();
    resolv_init();
    uip_ipaddr(ipaddr, 195,54,122,204);
    resolv_conf(ipaddr);
    resolv_query("www.sics.se");*/

  while(1) {
    uip_len = fecdev_read();
    if(uip_len > 0) {
      if(BUF->type == htons(UIP_ETHTYPE_IP)) {
    uip_arp_ipin();
    uip_input();
    /* If the above function invocation resulted in data that
       should be sent out on the network, the global variable
       uip_len is set to a value > 0. */
    if(uip_len > 0) {
      uip_arp_out();
      fecdev_send();
    }
      } else if(BUF->type == htons(UIP_ETHTYPE_ARP)) {
    uip_arp_arpin();
    /* If the above function invocation resulted in data that
       should be sent out on the network, the global variable
       uip_len is set to a value > 0. */
    if(uip_len > 0) {
      fecdev_send();
    }
      }
    } else if(timer_expired(&periodic_timer)) {
      timer_reset(&periodic_timer);
      for(i = 0; i < UIP_CONNS; i++) {
    uip_periodic(i);
    /* If the above function invocation resulted in data that
       should be sent out on the network, the global variable
       uip_len is set to a value > 0. */
    if(uip_len > 0) {
      uip_arp_out();
      fecdev_send();
    }
      }

#if UIP_UDP
      for(i = 0; i < UIP_UDP_CONNS; i++) {
    uip_udp_periodic(i);
    /* If the above function invocation resulted in data that
       should be sent out on the network, the global variable
       uip_len is set to a value > 0. */
    if(uip_len > 0) {
      uip_arp_out();
      fecdev_send();
    }
      }
#endif /* UIP_UDP */

      /* Call the ARP timer function every 10 seconds. */
      if(timer_expired(&arp_timer)) {
    timer_reset(&arp_timer);
    uip_arp_timer();
      }
    }
  }
  return;
}
```


表4　uIPのマクロ

関　数	説　明
uip_send	TCPデータを送信する
uip_abort()	現在の接続の中断
uip_acked()	前回の送信データに対してACKが返されたかどうか
uip_aborted()	現在のコネクションが送信先から中断されたか
uip_stop()	送信中のデータの送信停
uip_stopped()	現在の接続が前回uip_stopによって送信中止されたかどうか
uip_restart()	現在の接続で再接続を行う
uip_rexmit()	データの再送が必要かどうか
uip_udp_send()	UDPデータの送信を開始する
uip_polled()	現在のコネクションがuIPによってポーリングされたかどうか
uip_udp_bind()	udpの接続をローカル・ポートにバインドする
uip_udp_connection()	現在の接続がUDPの接続かどうかの判定
uip_udp_remove()	UDPの接続を削除する
uip_timedout()	接続がタイムアウトしたかどうか
uip_close()	現在の接続を閉じる
uip_closed()	現在の接続が閉じているかどうかの確認(0以外なら閉じている)
uip_connected()	現在の接続が接続しているかどうかの確認(0以外なら接続している)
uip_datalen()	現在利用可能なuip_appdataに入っている受信データの長さを返す
uip_newdata()	新しいデータが受信されているかどうか
uip_initalmss()	現在の接続の最大セグメント・サイズを初期化する
uip_mss()	最大セグメント・サイズ(maxium segment size)を得る

に示します．また，uIPのライブラリ・ルーチンを**表3**に，uIPで使用するマクロを**表4**に示します．

● アプリケーションをコンパイルする

今回作成したTOPPERS用のアプリケーションはsample_uip/以下にあります．uIPのアプリケーションの設定はuip/toppers/Makefile.uipにあります．Makefile.uip中のAPPS定義を元にuip/app/以下のアプリケーションがコンパイルされ，組み込まれるようになっています．もし，自前でuIPのアプリケーションを作成する際には，この部分から始めるとよいでしょう．Webサーバの場合，