実践報告

組込み機器向け，小規模TCP/IPプロトコルスタックの製作

東北ポリテクカレッジ　生産情報システム技術科　**谷本　富男**
（東北職業能力開発大学校）

1．はじめに

最近「ユビキタス（ubiquitous）」という言葉をよく耳にするようになった。語源はラテン語で，いたるところに存在する（遍在）という意味である。ユビキタス・ネットワーク社会とは，有線や無線によってネットワークに接続しているさまざまな種類のコンピュータがそれぞれ単独で存在するのではなく，通信しながら協調動作する社会のことを意味している。

一方，マイコンは応用分野としてさまざまな機器に組み込まれ，インテリジェント化に貢献してきた。ユビキタス社会の到来には，さまざまな機器間の有機的な接続が必要となり，組込みマイコンの役割にも，機器の制御に加えてネットワーク機能の追加が期待されている。

今回，小規模な組込みマイコンをターゲットにしたTCP/IPプロトコルスタックを製作したので紹介をする。

2．組込み機器事情

最近の組込み用途向けマイコンボードには，従来のシリアルインターフェース以外にもイーサネットのインターフェースを装備したものが多く見られるようになってきた。パソコンで使用されるような同程度のスペックを持ったマイコンボードはもちろん，Z80で代表される8ビット系マイコンボードにもイーサネットのインターフェースを装備しているマイコンボードも販売されている。さらにイーサネットのインターフェース以外に，パソコン市場ではすでに一般的なUSBやIrDA（赤外線通信）といった通信インターフェースがマイコンボードにも搭載されるようになってきている。

TCP/IP通信機能の実現には，開発の容易さや開発期間，費用（予算），性能，機能（サービス）等の要件により，①ネットワークOS（LinuxやWindows等）を組み込み機器用OSとして使用する方法，②他社からプロトコルスタックの提供を受ける方法，③ターゲットに実装するプロトコルスタックを絞って自作する方法等が考えられる。

一般的に小規模な組込み機器には，限られた記憶容量，処理能力ながらリアルタイム性が要求される場合が多く，通信機能の実現にパソコンのように必要十分な記憶容量，処理能力を割り当てることがコスト的にできない場合が多い。また，事務処理で使用される情報に比べ計測・制御分野での情報は，小容量ながら頻繁に送受信される特徴があり，パソコン環境をそのまま制御分野のセルレベル，フィールドレベルに当てはめることが必ずしも効果的と言えない。ここでは，それらの問題点を考慮し，オープンネットワークとして広く知られるようになったTCP/IP通信を題材に，限られた通信サービスであるが軽敏な通信プロトコルスタックの製作を目標にしてみた。開発言語はコストや速度，プログラム容量の点でアセンブリ言語が有利といえるが，可読性，移植性，拡張性等の点で有利なC言語を使用した。

3．ターゲットのハードウェア構成

上記で記述したように，ここでは小規模な組込み機器を意識したターゲットを用意した。すなわち限られた記憶容量，処理能力で，どの程度の性能が出せるかを測定してみた。図1に今回使用したターゲットのブロック構成図を示す。

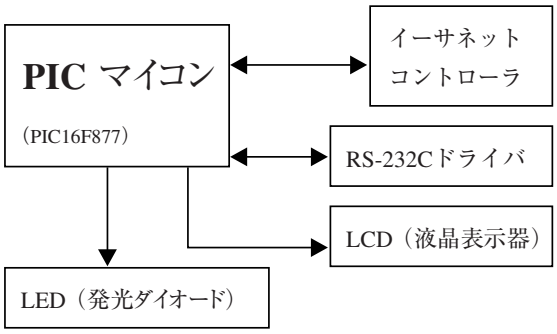

図1　ターゲットのブロック構成図

4．開発環境

開発方法としては，ホストPCでターゲット側プログラムを作成し，デバッグを行った後PICライターで書き込み，PICをターゲットボードに差し替え動作確認をするという手順を繰り返しながら作成した。差し替えが頻繁となるので，PICライターやターゲットのICソケットにゼロプレッシャソケットを用いている。

図2に開発環境の構成図を示す。

ホストPCには，開発統合環境（Microchip社 MPLAB IDE）をインストールし，C言語コンパイラ（HI-TECH社 PICC）で開発を行った。パケットの解析にはフリーソフトのetherealを使用した。

5．プロトコルスタック

今回製作したプロトコルスタックを図3に，仕様を以下に示す。ただし今回はTCP/IP通信用API（ソケットインターフェース）の実装を目標としていない。

主な仕様

・MACアドレスはあらかじめシリアルROMに記録

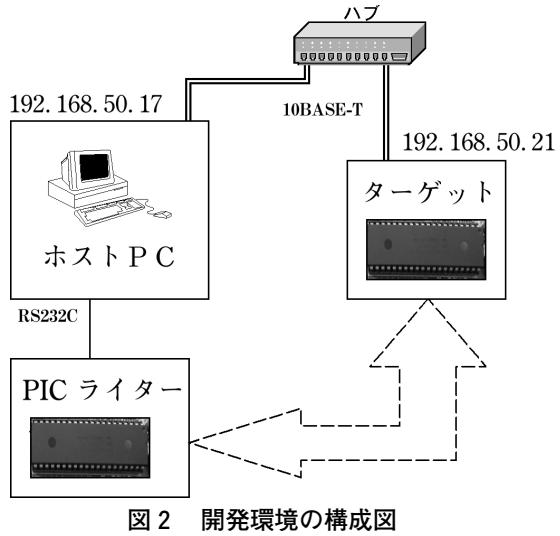

図2　開発環境の構成図

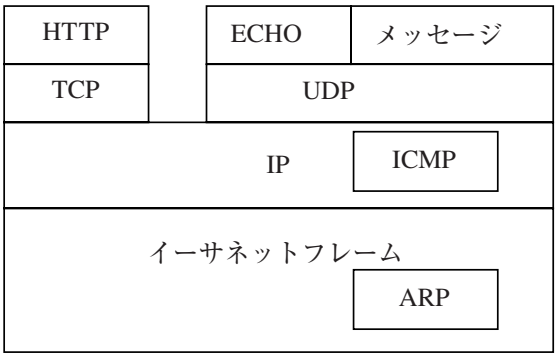

図3　プロトコルスタック

されているものを使用する。

・固定IPアドレスだが，実行中に他のPCよりICMPで変更できる。

・イーサネットフレームは原則250バイト以下を扱う（HTTP以外）。

・ICMPはエコー要求に対する応答を実装する。

・TCPは同時に1対1の接続のみが行える（同時複数の接続を扱わない）。

・メッセージはUDP 5000番で受け取ったデータをLCDに表示する。

6．開発手順と動作確認方法

プロトコルスタックの開発手順を以下に示す。

① LCDの初期化と表示

② RTL8019ASの初期化

③ MACアドレスの読み出しと設定

④　ARPプロトコルの製作
⑤　IP，ICMPプロトコルの製作
⑥　UDP，ECHOプロトコルの製作
⑦　TCP，HTTPプロトコルの製作
⑧　チューニングと最適化

マイコン系のシステム開発では，ICE（インサーキット・エミュレータ）を使用するのが一般的だが，今回は使用していない。そのかわりのデバッグの手段としてLCDに表示させた。①のLCDの初期化と表示は，完成時の表示のみならず，開発時のデバッグ作業に有効な手段となった。

上記①～③までで，MACアドレスをRTL8019ASに設定し，初期化が終了したことを写真1のようにLCDに表示させた。

写真1　初期化後

上記④のARPプロトコルの動作確認には，写真2のようにLCDに表示させた。また，etherealでパケット解析によっても確認できる。

写真2　ARPリクエストの受信

上記⑤のIP，ICMPプロトコル動作確認には，ホストPCよりPINGコマンドによって確認できる。そのときの動作を写真3のようにLCDに表示させた。

写真3　ICMPエコー要求と応答

上記⑥のUDP，ECHOプロトコルの動作確認には，ホストPCよりUDPパケットを送信するプログラム（写真4）を作成して動作確認を行った。

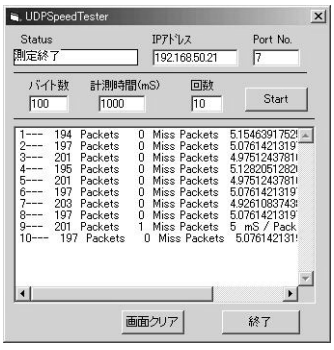

写真4　UDPSpeedTester

UDPSpeedTester（写真4）は，指定時間内に指定バイトのUDPパケットをホストPCとターゲット間で，何回送受信できるかを測定するために作成した自作ツールである。その時のLCDの表示を写真5に示す。このツールは後の性能測定のデータを得るためにも使用した。

写真5　UDP ECHOプロトコル

上記⑦のTCP，HTTPプロトコルでは，Webブラウザを使用して写真6のように表示させてみた。またetherealでのパケット解析の状況を写真7に示す。

ホストPCのIPアドレスが192.168.50.17で，ターゲットが192.168.50.21である。

写真6　TCP，HTTPプロトコル

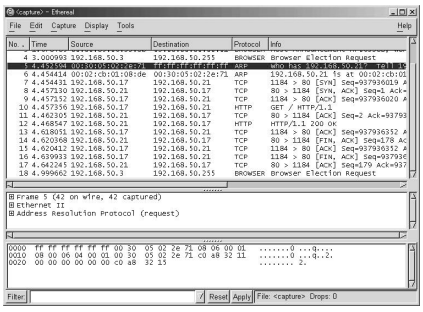

写真７　TCP，HTTPプロトコルの解析結果

⑧では，PICマイコンからRTL8019ASの制御のタイミングを安定限界まで絞り込む作業となる。また，プログラムの無駄な部分を排除するため，改めてプログラムを見直し，コンパイラに最適化やデバッグ情報の削除を指定し，プログラム容量のスリム化を図った。

７．性能測定

最初に通信速度では，ホストPC－UNIXワークステーション間と，ホストPC－ターゲット間とのPingコマンド（ICMP）で測定してみた。その場合，ターゲットのLCDに文字を表示しながらの測定は圧倒的に不利となることから，LCD表示部分をはずしたプログラムに変更している。

その結果，ホストPC－UNIXワークステーション間が約0.4mS，ホストPC－ターゲット間が約3.4mSとなった。ホストPC－UNIXワークステーション間とは９倍近くの応答速度差が生じていることがわかる。

次に，上記のUDPSpeedTester（UDP）で測定してみた。その時の結果を写真８から写真13に示す。

写真８　ホストPC－UNIXワークステーション間（10Bytes）

写真９　ホストPC－UNIXワークステーション間（100Bytes）

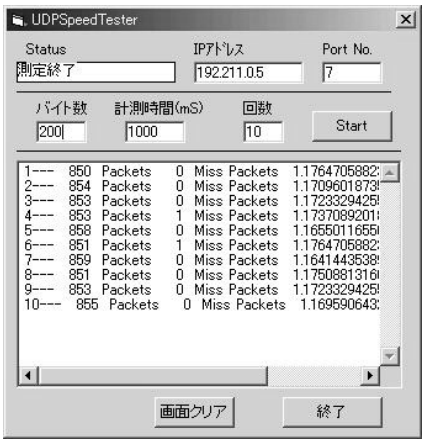

写真10　ホストPC－UNIXワークステーション間（200Bytes）

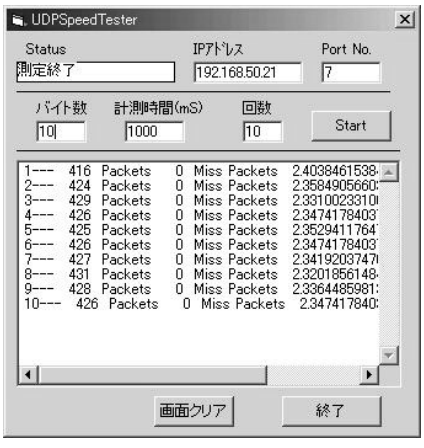

写真11　ホストPC－ターゲット間（10Bytes）

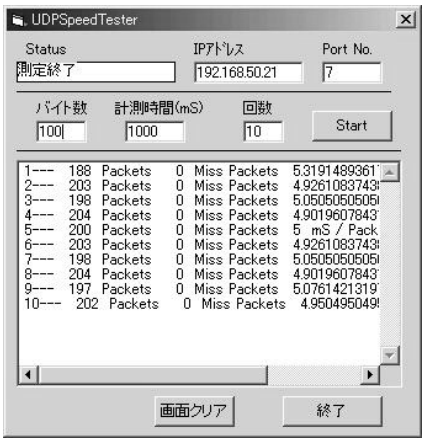

写真12　ホストPC－ターゲット間（100Bytes）

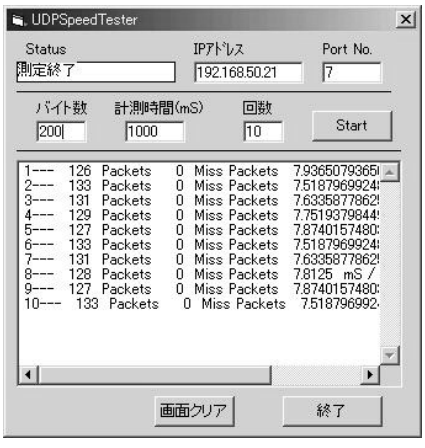

写真13　ホストPC－ターゲット間（200Bytes）

UDPSpeedTesterの見方は写真８で，UDPデータを10バイト，測定時間を１秒間，それを10回測定している。１回目は2247回の送受信が完了し，その時のミスパケットは検出されず，平均して0.44503……mSだったことを示す。

それぞれの平均的な値で見ると，UNIXワークステーション間とPICマイコン間では６～７倍の差が生じている。またプログラム容量の点では，TCP，HTTPとLCD表示を除けば，2kステップ（１ステップ14ビット），含めると5kステップ，使用RAMが約360バイト（内270バイトはRTL8019ASとのバッファとして使用）で製作できた。

８．まとめ

最近までの計測・制御分野では，それぞれ独自の通信プロトコルが使用され，FAとかCIMが構築されてきた経過がある。しかし，近年のインターネットで代表されるオープンネットワーク（TCP/IP通信）の爆発的ともいえる普及により，計測・制御分野のフィールドレベル（ハードウェアに近い領域の通信）領域を除いてはTCP/IP通信が可能な（イーサネットのインターフェースを持っている）機器が見られるようになった。顧客ニーズの多様化により，もはやメーカサイドも１社でFAシステム全体をサポートできない時代となり，顧客サイドやメーカサイドの双方がオープンネットワーク化によるオープンシステム構築が無視できない時代になってきている。そのことはオープンネットワークに接続された機器間が有機的に協調動作ができる環境が整いつつあるともいえる。

しかし，パソコン等で使用される通信プロトコルをそのまますべて実装することができない分野も存在する。そういった分野の機器には限られた条件範囲にあわせる必要が生じる。例えば小規模な機器（例えば５万円以下）で，TCP/IP通信をさせるためにパソコン性能相当のマイコンボードを組み込むと，製品価格的にも容積的にも大きく影響する。今回はそのような場面を想定し，小規模な組込みマイコン用プロトコルスタックを製作してみた。価格的には，ターゲットがLCDを入れなければ3,000～5,000円程度で製作可能と考える。使用したC言語はHI-TECH社のPICCで120,000円程度となる。PIC用のＣコンパイラがフリーソフトもしくは低価格のものが使用できれば，その分費用が抑えられる[注１)]。

また，通信速度の改善策では，

① アセンブリ言語を使用する。
② CPUクロックを上げる。
③ 高性能CPUに替える。
④ CPUのローカルバスにNIC（RTL8019AS）を直接接続する。
⑤ NIC（RTL8019AS）とは16ビットアクセスを行う。

などで対処することが考えられる。

筆者自身，TCP/IP通信の経験はソケットプログラミング程度であったが，製作するにつれて，プロトコルスタックの細かな部分や，パケットフィルタリング等のセキュリティ関係の理解も深めることができた。

筆者は決して通信分野の専門ではないが，以前より通信分野の知識の必要性を感じていた。その点では絶好の教材となった。

最後に，同僚の大内敬仁氏からの多くのアドバイスに感謝したい。

〈参考文献〉

1）『PICNICソースコード（アセンブリ言語）』，TriState社.
2）「電子工作の実験室」 http://www.picfun.com/
3）『PIC16F87X Data sheet』，Microchip社.
4）「RTL8019ASの概要と使い方」，『トランジスタ技術』，2001年１月号.
5）『RTL8019AS Realtec Full-Duplex Ethernet Controller With Plug and Play Function』.
6）『Writing Driver for the DP8369 NIC Family of Ethernet Controllers』.
7）「チェックサムの計算方法」 http://www.sowa.is.utec.ac.jp/~ueno/material/tcp.ip/etc/checksum.html
8）『TCP/IPプロトコル徹底解析』，日経BP社.
9）その他 TCP/IP関連の専門書およびHP

注１）2002年11月より，PIC16F877用Ｃコンパイラは，PICC Lite として HI-TECH社のホームページから無償でダウンロードできるようになった。