OAKS16/16-62P，OAKS32 対応

OAKS16-LAN Board マニュアル

Ver. 2.4

小豆のように小さな通信プロトコルスタック AzkiTCP/IP API 解説書

[適応 CPU]

・M16C/62 ， M32C/80 シリーズ

[適応ボード]

・OAKS16 シリーズ CPU ボード　　OAKS16-M30620FCAFP

・OAKS16-62P シリーズ CPU ボード　OAKS16-M30626FHPFP

・OAKS32 シリーズ CPU ボード　　OAKS32-M30833FJFP（100PIN）

[ご注意]

この説明書では、OAKS16 シリーズを例に解説しています。特に断りの無い限り、他シリーズをお使いの場合は製品名称等を読み替えてください。

安全設計に関するお願い

● 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

● 本資料は、お客様が用途に応じた適切な製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてオークス電子および情報を提供いただいた各社が所有する知的財産権その他の権利の実施、使用を許諾するものではありません。

● 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、オークス電子は責任を負いません。

● 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、オークス電子は特性改良などにより予告なしに変更することがあります。

● 本資料に記載の図、表に示す技術的な内容、及びプログラム、アルゴリズムを流用する場合、お客様の責任において実施してください。また、組み込んだプログラム、アルゴリズム単体で評価するだけでなく、システム全体で充分に評価してください。オークス電子は、一切責任を負いません。

● 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、オークス電子へご照会ください。

● 本資料の転載、複製については、文書によるオークス電子の事前の承諾が必要です。

● 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらオークス電子までご照会ください。

はじめに

このたびは、OAKS16-LANBoard をお買い上げいただきまして誠にありがとうございます。このマニュアルは、OAKS16-LANBoard に含まれるハードウェアおよびソフトウェアのセットアップ方法、通信プロトコルスタックの解説、使用上の注意点について述べたものです。

この OAKS16-LANBoard は、手軽にイーサネットの LAN やインターネットに接続するためのハードウエア基板、ソフトウエア通信プロトコルスタックがすべて用意されたキットです。

LAN コントローラは REALTEK 社 RTL8019AS を搭載し、10Base-T のモジュラジャックを装備しています。ネットワークの構成によっては OAKS16-LANBoard にグローバル IP を付与することもできるようユニークな MAC アドレスを持っています。グローバル IP アドレスを付与すれば、インターネットに接続されたすべてのコンピュータや、携帯電話の i-mode などからの制御も可能です。

この LANBoard は OAKS16 シリーズの M16C/62 マイコンボード（OAKS16-M30620FCAFP）と接続することでご使用になれます。もしこの OAKS16-LANBoard のみを単体でご購入された場合は、別途、OAKS16-FullKIT をご用意下さい。

また､これらキットに含まれるソフトウエアの使用権は OAKS16 シリーズでの開発限られています。他のターゲットシステムやハードウエアなどでの開発には別途ライセンスをご購入していただく必要があります。ご注意ください。

【ヒント】この OAKS16-LANBoard の通信プロトコルスタック（AzkiTCP/IP）はアンカーシステムズ株式会社提供のリアルタイム OS（AzkiRTOS）の上で動作しています。AzkiRTOS のタスクモニタ、関数・変数クロスリファレンスなどは、AnchorPlace Lite（別売¥9,800） が必要になります。AnchorPlaceLite を使うと、編集からコンパイル、リンク、デバッグまでをシームレスに行うことができます。

【ヒント】サンプルには C コンパイラ付属統合化開発環境 TM を使ったプロジェクトと、アンカーシステムズ製統合開発環境 AnchorPlace Lite（KIT 種類により別売）を使った開発プロジェクトが入っています。

【ヒント】OAKS16-FullKIT や OAKS16-BoardKIT の使い方はそれぞれのマニュアルをご覧ください。

目次

１．製品パッケージの内容

OAKS16-LANBoard 製品パッケージの包装内容を示します。開封時に包装内容をご確認下さい。

１．１．包装製品一覧表

本製品の包装内容を、表 1-1 に示します。

表 1-1 OAKS16-LANBoard 包装内容一覧表

製品名	数量	備考
OAKS16-LANBoard	1 枚	OAKS16/32 マイコンボード用 10Base-T LAN ボード
CD-R	1 枚	マニュアル、サンプル等

１．２．CD-R

CD-R にはプログラム開発に必要なソフトウェアプログラム、PDF マニュアルなどが含まれています。以下に CDROM の構成を示します。

【注意】サンプルやソースファイルは CPU ボードごとにフォルダが分かれています。それぞれのフォルダにあるファイルをご使用ください。

表 1-2 CD-R 構成一覧表

フォルダ		内容
doc		本マニュアル、回路図
OAKS16TCP	AzNC30	リアルタイム OS、AzkiRTOS、ソースコード、解説
	AzTCPIP	通信プロトコルスタック、ソースコード
	Src	サンプルプログラム
OAKS16-62PTCP	AzNC30	リアルタイム OS、AzkiRTOS、ソースコード、解説
	AzTCPIP	通信プロトコルスタック、ソースコード
	Src	サンプルプログラム
OAKS32TCP	AzNC308	リアルタイム OS、AzkiRTOS、ソースコード、解説
	AzTCPIP	通信プロトコルスタック、ソースコード
	Src	サンプルプログラム
udp_cli_nonblk		UDP/IP クライアントサンプル（Visual C++ 6.0 以上）

１．３．ソフトウエア製品

C コンパイラ、リンカ、Flash ライタなど言語ツールは、本 CD には入っていません。OAKS16-BoardKIT や OAKS16-FullKIT に含まれています。使用条件等はそちらをご覧ください。

表 1-3 ソフトウエアライセンスポリシー一覧表

製品	ライセンスポリシー
AzkiRTOS	無償で制限無くご使用いただけます。著作権はアンカーシステムズ株式会社にあります。
AzkiTCP/IP	OAKS16 シリーズでの開発ができます。OAKS16 シリーズ以外のターゲットでのご使用には別途開発ライセンスが必要です。AzkiTCP/IP の著作権はアンカーシステムズ株式会社にあります。

１．４．ハードウエア構成部品

表 1-4 OAKS16-LANBoard 構成部品一覧

記号	型番	メーカ	数量	備考
PCB		アンカーシステムズ	1	OAKS16 LAN BOARD
IC1	RTL8019AS	REALTEK	1	
IC2	AT93C46-10PI-2.7	ATMEL	1	Serial EEPROM
X1	HC-49/U-S 20MHz	KSS	1	20.0MHz
L1	20F001N	YCL	1	PulseTrans
CN6	TM5RJ3-88	HIROSE	1	RJ45
D1,2,3,4,5			5	LED 赤*1,緑*1,黄*3
R1,2,3,5,6			5	1KΩ
R4			1	200Ω
C1,2			2	0.1uF
C3,4			2	22pF
C5,6			2	0.01uF
C7,8			2	0.01uF-0.001uF/1KV

【ヒント】製品には代替品が使われることがあります。 基板に Q1、D6、R7、IC3 の印刷がある場合がありますが、部品はありません。

２．保証ならびにサポートについて

２．１．保証

本キットは評価用という位置付けですので、欠品、破損、初期不良時の無償交換のみのサポートとさせていただきます。それ以外の保証については行いません。

２．２．サポート

本キットに関する電話によるサポートはお受けできません。OAKS16 シリーズのサポートならびに最新情報ついてはオークス電子ホームページを照会してください。

http://www.oaks-ele.com/

AzkiRTOS、AzkiTCP/IP、統合開発環境 AnchorPlaceLite については、アンカーシステムズのホームページをご覧ください。サポート会議室も開いています。

http://www.anchorsystems.co.jp/

３．システム構成

３．１．ハードウエア構成

OAKS16-LANBoard を動作させるためには、M16C/62 マイコンボード OAKS16-M30620FCAFP が必要です。このマイコンボードは、OAKS16-BoardKIT または OAKS16-FullKIT に含まれています。

３．２．開発システム構成

OAKS16-LANBoard と組み合わせた M16C/62 マイコンボード OAKS16-M30620FCAFP のプログラム開発を行うには、OAKS16-FullKIT または、OAKS16-BoardKIT＋OAKS16-LCDBoard が必要です。

ホストコンピュータとの接続は OAKS16-BoardKIT、OAKS16-FullKIT の該当項をご覧ください。

４．ハードウエアセットアップ

４．１．LANBoard 組み立て

OAKS16-LanBoard は、単体では動作しません。M16C/62 マイコンボード（OAKS16-M30620FCAFP）と接続する必要があります。

（１）OAKS16-LANBoard の上に M16C/62 マイコンボードを重ねてコネクタ穴位置を合わせて挿します。（下図参照）

図 4-1 外観図

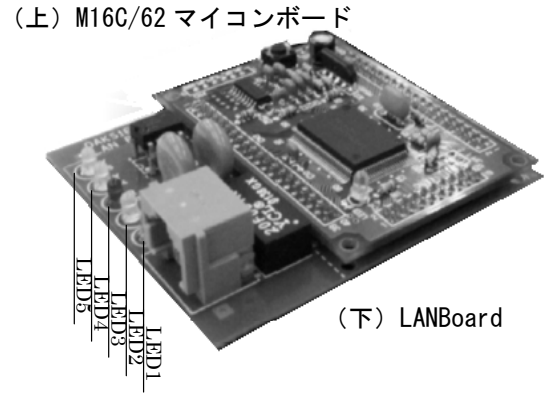
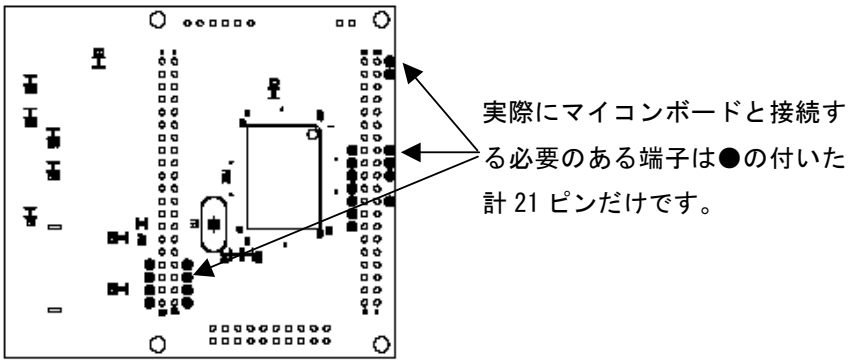


表 4-1 LED 動作

LED 番号	動作
LED1（緑）	リンク時点灯
LED2（黄）	データ受信時点灯
LED3（赤）	データ送信時点灯
LED4（黄）	予備
LED5（黄）	予備（サンプルで使用）

図 4-2 裏面図



（２）図 4-2 裏面図にある●印のついた PIN を半田づけします。

【ヒント】半田づけはできるだけ PIN の根元だけにしてください。PIN の先まで半田が付着してしまうと PIN が太ってしまうため、拡張ボードなどのヘッダソケットに挿入しにくくなってしまいます。

【ヒント】LANBoard のランド穴の中はスルーホールメッキされていませんので、後で半田を吸取線などで除去すれば簡単にマイコンボードから外せます。

（３）拡張ボードがある場合はそのソケットに作成したボードを挿します。

５．サンプルプログラム

ここでは、組み立てた LANBoard をサンプルプログラムで動作させます。

使用する環境により、■コンパイラ付属統合化開発環境 TM と■アンカーシステムズ製統合開発環境 AnchorPlaceLite（KIT 種類により別売）の２種類のプロジェクトファイルが入っています。また、それぞれのツールの使い方は各ツールの説明書をご覧ください。以下の説明では、これらツールがインストールされ基本機能は使える状態であることを想定しています。必要なサンプルをフォルダごとローカルハードディスクにコピーしてください。

【ヒント】TM の.tmk プロジェクトファイルは<work>フォルダ内に、AnchorPlaceLite の.app プロジェクトファイルはソースと同じ階層にあります。

５．１．ネットワーク環境の準備

OAKS16-LANBoard と LAN をネットワークケーブルで接続します。また OAKS16-LANBoard を接続するネットワーク環境について、サンプルプログラムでは以下の情報が必要です。

（１）OAKS16-LANBoard に設定する IP アドレス

（２）所属するネットワークのネットマスク

（３）※デフォルトゲートウエイ IP アドレス

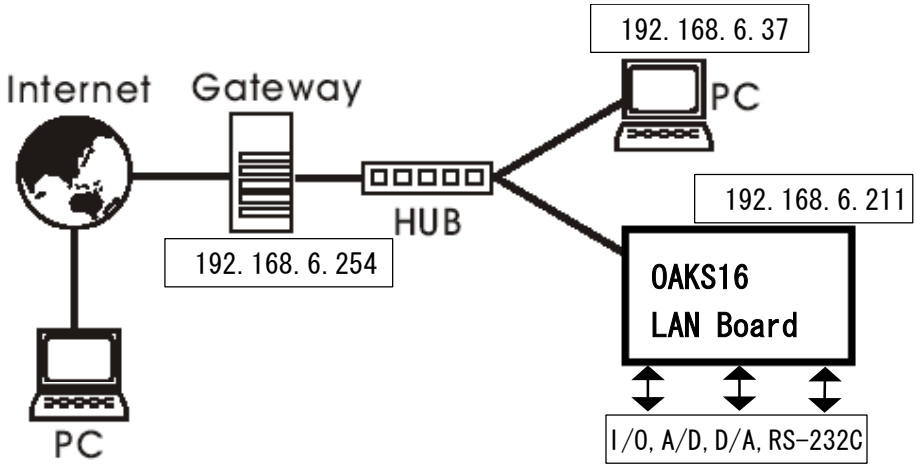
（４）※DHCP サーバが存在し、DHCP クライアントとなるかどうか

（５）※SMTP、POP3 サーバの IP アドレス、ポート番号

【ヒント】※印の項目は無くてもかまいません。デフォルトゲートウエイが無い場合は同一ネットワークアドレス内での通信しかできません。また DHCP クライアントとして動作させる場合には、同一ネットワーク内に DHCP サーバが立ち上がっている必要があります。

【ヒント】ネットワークの管理上独自の IP アドレスを OAKS16-LANBoard 用に使えない場合は、PC と１対１で接続することで動作可能です。この時、HUB を介さない場合は LAN ケーブルはクロス接続用のものを使います。

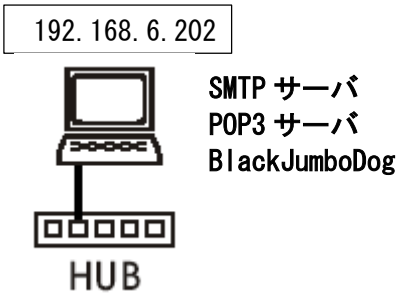
図 5-1a ネットワーク概略図



上図はネットワークアドレス 192.168.6.0 、ネットマスク 255.255.255.0 のネットワークの概念図です。IP アドレスはあなたのネットワーク IP に読み替えてください。後でサンプルプログラムのヘッダを編集します。以下の解説では、各 IP アドレスが上図に示す IP アドレスが割り振られているものとします。

加えて、E メール送受信サンプルを動作させるにはメールサーバが必要です。本説明書では、192.168.6.202 に SMTP サーバと POP3 サーバを起動させています。

図 5-1b SMTP サーバ



【ヒント】図 5-1A 例の 192.168.6.37 の PC でメールサーバを起動することもできます。

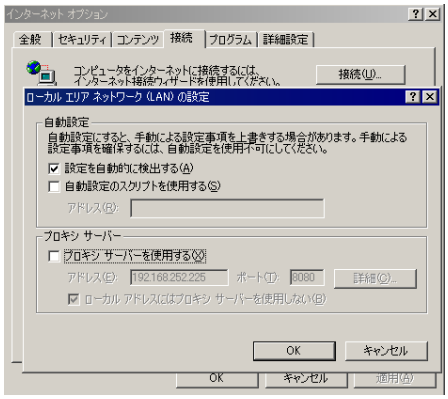
【ヒント】BlackJumboDog は、SAPPORO WORKS さんが提供しているフリーソフトです。

５．２．ブラウザの準備

TCP/IP を使った HTTP サーバがサンプルプログラムにあります。これを操作するにはネットワークに接続された PC にブラウザが必要です。ここでは代表的な Windows に付属する IE (InternetExplorer) を例にします。

プロキシサーバには対応していませんので、メニューから[ツール]-[インターネットオプション]-[接続]で、[プロキシサーバを使用する]の項目を OFF にしてください。

図 5-2 ブラウザ設定



【ヒント】通常プロキシを使っている環境において、ブラウザの詳細設定で、OAKS16-LANBoard が属しているネットワークのアドレスに対してプロキシサーバを外すだけでかまいません。

５．３．サンプルプログラムの編集－デバッガ起動

図 5-3　ethernet.h 編集項目

```
//////////////////////////////////////////////////////////////
// Azki TCPIP definition
//////////////////////////////////////////////////////////////
#define DO_DHCP          FALSE              //「TRUE」の場合DHCPクライアントになる。

//DO_DHCP   FALSE の場合
#define MYIPADDRESS      {192,168,6,211}    //CPUボードのIPアドレス
#define MASKADDRESS      {255,255,255,0}    //ネットマスク
#define GWIPADDRESS      {192,168,6,254}    //デフォルトGatewayのIPアドレス

#define CEP_MAX          6                  //EndPoint数（TCP/UDP ソケット数）
#define ARPTBL_MAX       CEP_MAX*2          //IP-MAC テーブル数

#define TCP_RX_WINDOW_SIZE       1024-54            //TCP受信可能サイズ
#define BUF_SIZE_RX      TCP_RX_WINDOW_SIZE + 54    //Etherパケット受信バッファサイズ

// HTTP layer
#define      HTTP_STR_MAX        1460    //HTTP（TCP）で一回で送信できる制限バイト数。
                                         //受信ではHTTPヘッダなどを含んだクエリ受信枠。
```

（１）　サンプルにあるファイル ethernet.h　の 120 行目付近の上図にある記述で、MYIPADDRESS、MASKADDRESS、GWIPADDRESS をあなたのネットワークのものに書き換えて保存してください。

DO_DHCP	DHCP サーバに IP アドレスを割り当ててもらう場合は TRUE に変更します。その場合 MYIPADDRESS 以下の定義は不要です。
MYIPADDRESS	OAKS16-LANBoard に設定する IP アドレスです。
MASKADDRESS	所属するネットワークのネットマスクです。
GWIPADDRESS	所属するネットワークのデフォルトゲートウエイの IP アドレスです。

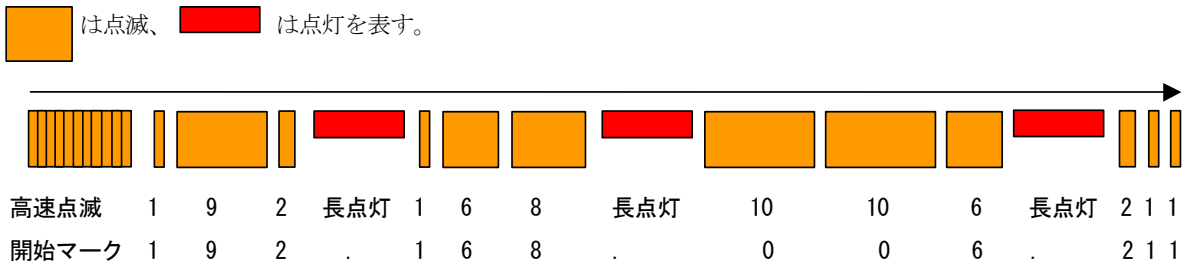
（２）　コンパイル、リンクします。

（３）　ターゲットボードの電源を ON にしてデバッガを起動し、RUN させてください。マイコンボード上の LED が点滅しているはずです。※「IP アドレス表示」参照

IP アドレス表示

このマイコンボード上の LED 点滅は、設定された自局 IP アドレスを表示しています。従って DHCP クライアントとして動作したときに、何番に割り当てられたかをデバッガを使わなくても知ることができます。

LED 点滅の点灯回数が 10 進数の数字一桁を表します。例えば、3 回点滅したときは数字の 3 を表します。IP アドレスは、3 桁ごとに 4 セット分点滅表示します。セット間には、数秒の点灯パターンを挿みます。また、10 回点滅は 0 を表します。数秒間の高速点滅は、表示開始を知らせるパターンです。



つまりこの点滅で IP アドレス、 **192.168.6.211** であることを表示している。

【ヒント】DHCP サーバからうまく割り当て IP アドレスを受け取れなかったときには、数秒間の高速点滅だけを数 10 秒ごとに繰り返します。

【ヒント】ここで LED(LANBoard でなくマイコンボード上の)が点滅していないときは LANBoard ではなく、マイコンボード側の問題があります。LANBoard のサンプル動作を中止し、マイコンボードに付属している何かのサンプルを動作させてみてください。

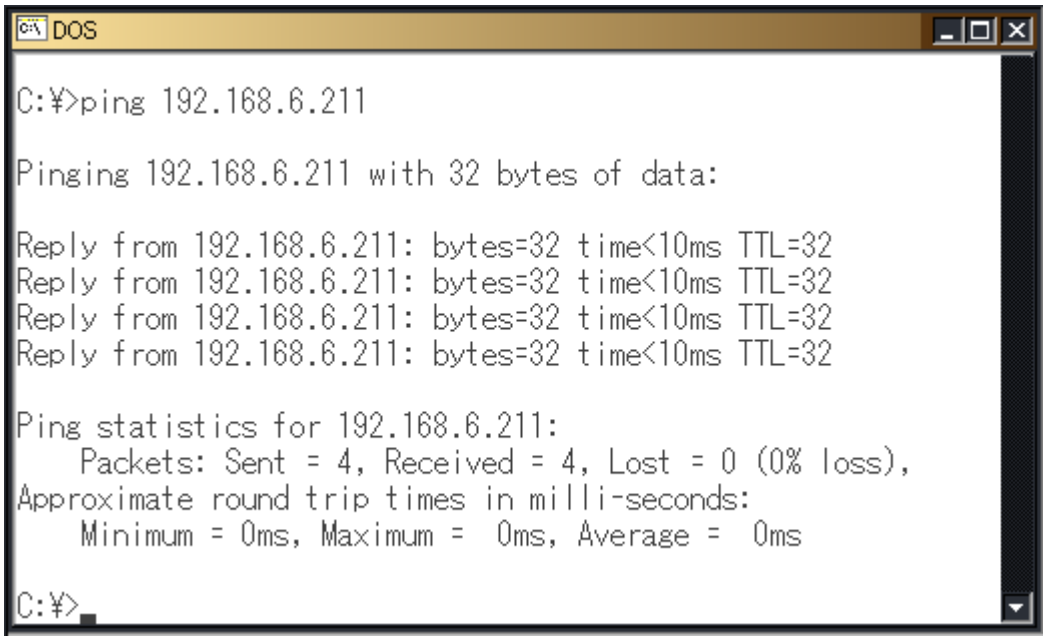
【ヒント】サンプルソースコードの AzkiRTOS は ASSERT マクロが有効な DEBUG モードになっています。無効にするには、azrtos(308).h の _AZ_EMU_DEBUG をコメントアウトするか、ASSERT マクロを書き換えてください。

５．４．ICMP－ECHO

まずは接続しているかどうかを確かめます。

DOS 窓を開いて、 『 ping 192.168.6.211 』と入力してください。

図 5-4 ping 応答の様子



上図のように OAKS16-LANBoard から応答があるはずです。

５．５．UDP/IP サーバ

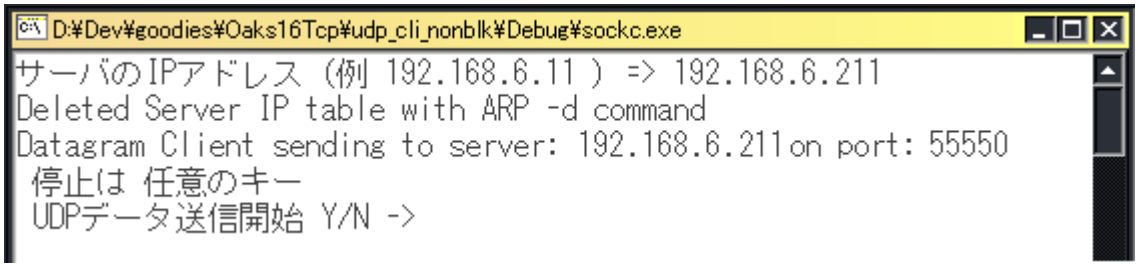
この UDP/IP は IP 通信にポートを追加してアプリケーションプログラムからデータ通信を行えるようにしたものです。

サンプル CD には、クライアントプログラムサンプルが付いています。WinSock を使ったクライアント例です。

このサンプルでは、クライアントの PC から文字列を送信し、OAKS16LANBoard が受信し、受信文字列を加工して送り返す単純なものです。これをもとに仕様を拡張することができます。

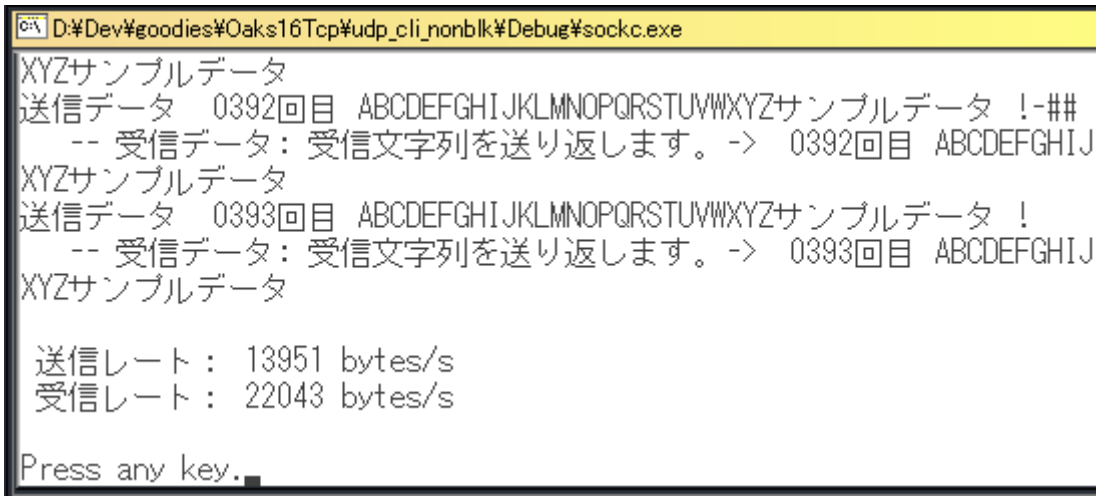
サンプル CD の<udp_cli_nonblk>の中の<debug>フォルダにある sockc.exe を起動してください。

図 5-5a UDP クライアントサンプルプログラム起動



1 行目は、OAKS16-LANBoard の IP アドレスを入力しリターンします。Y を入力してリターンすれば送受信を開始します。このときサンプルでは、ポート番号 55550 を使っています。任意のキーを押すと送信を中止します。

図 5-5b UDP クライアントサンプルプログラム終了



５．６．TCP/IP

AzkiTCP/IP は制御目的に使う最低限の通信プロトコルをサポートしています。TCP においては、シーケンス番号確認による再送処理、ウインドウコントロールによるオーバーフロー防止を行っていません。またデータはストリームではなくパケット単位（最大 1460 バイト）で送受信されています。従ってアプリケーション側で異常時の対応、データストリームの再生をする必要があります。TCP/IP だけのサンプルは特に付けておりません。興味のある方は WinSock の参考書などをご覧ください。付属サンプルでは、次の HTTP サーバ、E メール送受信に TCP を使っています。

【注意】TCP/IP プロトコルは必要最低限しか実装していません。※通信 API 参照

５．７．HTTP サーバ

HTTP は、ブラウザで使われているプロトコルです。OAKS16-LANBoard は、外部メモリを一切使わず HTTP サーバを実現しています。

インターネットブラウザを起動し、アドレス入力欄に、『 http://192.168.6.211/ 』と入力します。下図のような画面がブラウザ上に表示されます。OAKS16-LANBoard の HTTP サーバサンプルプログラムが、ブラウザの要求に従って、HTML データを送信しています。

■ブラウザから受け取ったクエリ文字列

ここには、ブラウザで操作された内容が OAKS16-LANBoard に送信され、それを再度ブラウザに返信して表示されています。

■I/O 制御サンプル

画面にある ON、OFF ボタンで、OAKS16-LANBoard にある LED5（黄）の ON,OFF 制御ができます。

■入力データ送信サンプル

入力したデータを OAKS16-LANBoard に送信します。サンプルでは何もしていません。

■E メール送信サンプル

入力したデータに文字列を追加して指定宛先に E メールを送信します。

■E メール受信サンプル

指定されたアカウントの最新メールを受信し、件名を表示します。

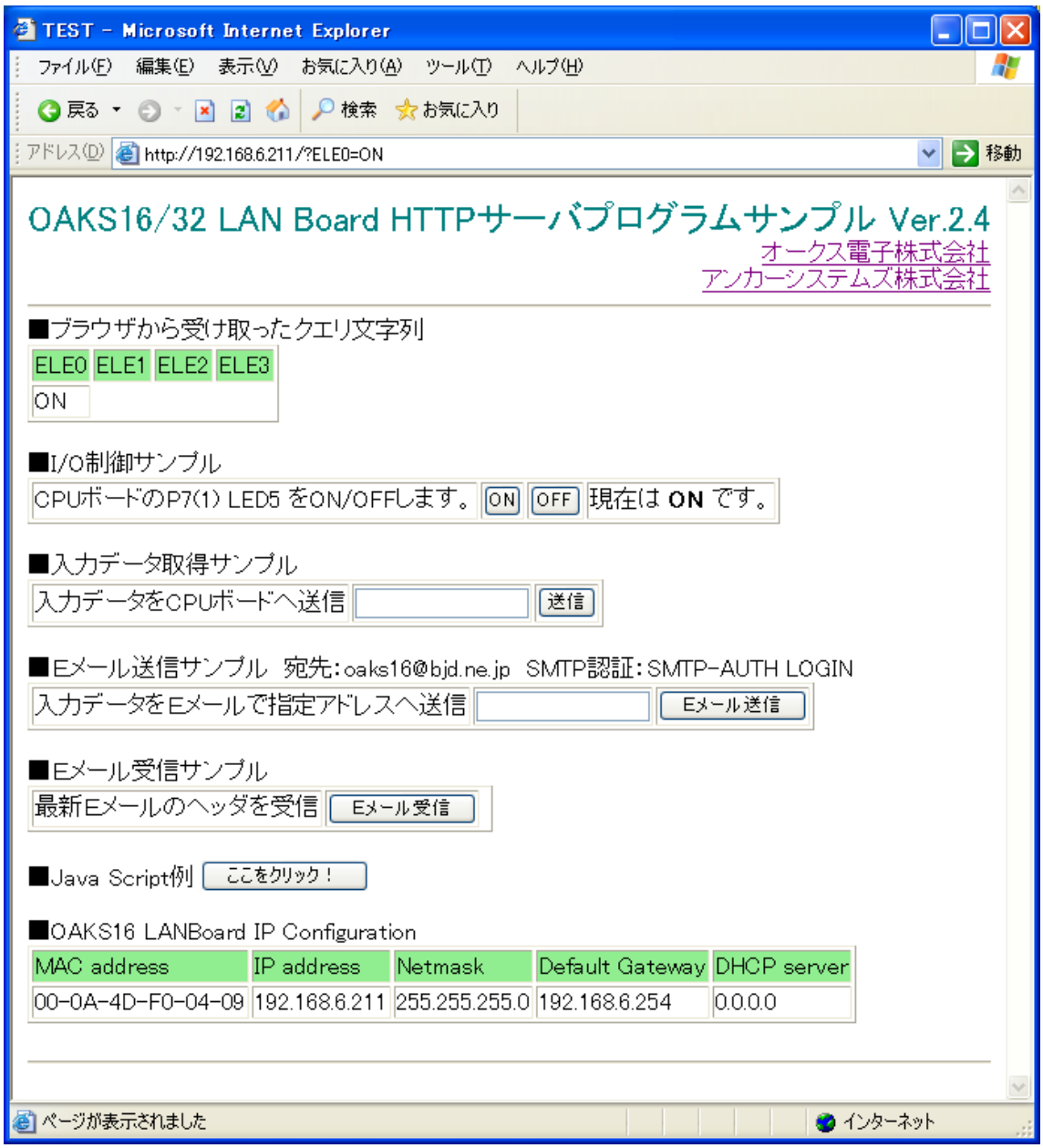
■E メールで I/O 制御サンプル

携帯電話や PC からの E メールを受信し、LED5 を ON,OFF します。

■OAKS16 LANBoard IP Configuration

OAKS16-LANBoard のネットワーク関連設定値を表示しています。

図 5-7 HTTP サーバサンプル画面



この HTTP サーバサンプルを拡張させていけばいろんな I/O を自由にブラウザから制御できるようになります。

【ヒント】HTTP サーバを構成するには、組み込み C 言語の知識以外に HTML（HyperText Markup Language）言語の知識と CGI（CommonGatewayInterface）の知識が必要になります。サンプルには最低限の CGI 要素が書かれていますが、より高度なアプリケーションを作成するにはそれぞれの参考書をご覧ください。十分実用的なアプリケーションが作成できるはずです。

５．８．E メール送受信 SMTP・POP3

図 5-7 HTTP サーバサンプル画面の中ほどに■E メール送信サンプルがあります。これは、右のテキスト入力欄に記入した文字列を宛先アドレスへ E メールとして SMTP プロトコルで送信するサンプルです。

図 5-8 SMTP、POP3、E メール送受信設定　（mailer.h）

```
//メーラ基本設定
//【注意】以下LOCAL環境での架空アドレスです。
// 実験のため、送信者と受信者を同じアカウントにしています。
// 外部ネットワークへ接続する場合は必ず正規のアカウントで送信してください。

#define SMTPAUTH_LOGIN      FALSE       //SMTP-AUTH LOGIN 認証を使うとき TRUE （Portは通常587）

#define SENDERADDRESS       "oaks16@bjd.ne.jp"      //送信者Eメールアドレス
#define RECVERADDRESS       "oaks16@bjd.ne.jp"      //宛先Eメールアドレス

#define SMTPSERVER      {192,168,6,202}     //SMTPサーバのIPアドレス
#define POP3SERVER      {192,168,6,202}     //POPサーバのIPアドレス

#define SMTPPORT        25              //SMTPサーバのPORT番号　SMTP-AUTH認証有りのときは通常587
#define POP3PORT        110             //POPサーバのPORT番号

#define USERNAME        "oaks16"        //POPサーバLOGIN用 User ID
#define PASSWORD        "oaks16"        //POPサーバLOGIN用 パスワード
#define B64LENGTHMAX    50              //UserID、パスワードをMIME処理するバッファサイズ

#define EMAIL_COMMANDER     FALSE       //EメールでI/O制御 のサンプルを起動する

#endif  // __MAILER_H__
```

図 5-8 は E メール送受信で必要な設定項目です。ファイル＜mailer.h＞の該当項目をご自身の環境に合わせて書き換えてください。

【注意】OAKS16-LANBoard には DNS のレゾルバはありませんので、SMTP や POP3 サーバは、名前ではなく IP アドレスを直接記述してください。

サンプルでは、E メール送信を HTTP サーバのブラウザ画面のボタンクリックで行っていますが、プログラム内のタイマやイベントをトリガに送信するのが普通の使い方です。OAKS16 を組み込んだ機器から定期的に内部の状態を E メールで指定アドレスへレポートを送るというような使い方ができます。

E メール受信 POP3 もサンプルとして掲載していますが、OAKS16-LANBoard で E メールを受信して本文を読み込んでも応用が困難なため、件名だけを受信します。E メール受信ボタンをクリックすることで E メール受信します。

また POP before SMTP のサーバに対して、ID、Password 認証を行えるだけの実装も可能です。サブ

ミッションポート 587 を用いた SMTP 認証は、LOGIN に対応しています。プロバイダによってはポート 25 が使えませんので、その場合、SMTPAUTH_LOGIN を TRUE に設定してください。

【ヒント】別途メールサーバが必要です。契約しているプロバイダのメールサーバを使うか、SAPPORO WORKS さんが公開されている「Black Jumbo Dog」のようなローカル環境でのメールサーバを実験的に使うことができます。「Black Jumbo Dog」はフリーウエアで商用利用も可能です。実験している同じ PC に難解な設定なしにメールサーバを起動することができます。

【注意】基本的に本実験はローカル環境で行うことをお勧めします。またインターネット上へ E メールを送信するときは、必ず正規のアカウントを設定してください。またすべてのプロバイダのメールサーバで E メール送受信が行えることを保証するものではありません。あくまでサンプルとしてご利用ください。

【注意】送信できる文字列は ASCII 英数字のみです。それ以外の文字はエンコードする必要があります。

５．９．E メールで I/O 制御

図 5-8 メーラ設定の下部に #define MAIL_COMMANDER の定義があります。これを TRUE にすると server.c 内のメールコマンダ部実装が起動し、設定したメールアカウントに届いたメールで LED5 を ON,OFF できます。サンプル実装では以下のコマンドが使えます。

外部メーラ（Outlook、携帯 i-mode など）から指定アカウントに E メールを送信

※件名に命令として　　LED5=ON　または、　LED5=OFF

の文字列を記述します。文字は ASCII 大文字とします。メール本文は空にしておきます。

サンプル実装では 60 秒に一度、POP3 サーバを読みに行き、最新メールの件名（命令）を解析して、コマンドを見つけたらその動作（LED5 の点灯消灯）を行うようになっています。

５．１０．DHCP クライアント

図 5-3 ethernet.h 編集項目の DO_DHCP 定義を TRUE にすることで、DHCP クライアントの動作をします。

DHCP クライアントとは、ネットワークに接続して電源を ON するだけで自身の IP アドレス、ネットマスク、ゲートウエイ IP アドレスを自動的に DHCP サーバから取得して設定してくれるものです。このサンプルでも OAKS16-LANBoard が起動すれば、同じネットワーク内にある DHCP サーバを探し、新しい IP アドレスを割り振ってくれるようネゴシエーションを行います。

この付属サンプルでは、DHCP で取得できた IP アドレスを知りたいときは、一度デバッガをブレークさせてグローバル変数 MyIP を確認する、もしくは RAM モニタを ON しておき変数 MyIP を確認することになります。CPU ボード上の LED 点滅パターンにて IP アドレスを知る方法もあります。

【ヒント】同一ネットワーク外の DHCP リレーエージェントには対応していません。

６．サンプルネットワークプロトコルスタック概説

６．１．プロトコルスタック概要

OAKS16-LANBoard に付属している通信プロトコルスタックは、アンカーシステムズ株式会社提供の AzkiRTOS 上で動作する AzkiTCP/IP です。この AzkiTCP/IP は、制御目的に使うワンチップマイコンの内臓メモリのみで最低限のイーサネット通信を行うために開発されたものです。

必要メモリは、●RAM 2K バイト ●ROM 10K バイト という超コンパクトな設計です。

搭載してあるプロトコルは、 ■ARP ■ICMP-ECHO ■IP ■UDP ■TCP ■HTTP サーバ ■SMTP ■POP3 ■DHCP クライアント です。

これら通信プロトコルはすべてソースコードで提供されていますので、自由に拡張、変更して使うことができます。

組み込み機器などで独自の通信を行う場合でも必要なのは通常 UDP、または TCP です。それ以外のプロトコルは自動的に動作していますのでサンプルプログラムの server.c を拡張させていくことで独自の処理プログラムを開発できます。

【ヒント】イーサネット通信の詳細を知りたい場合はサンプルすべてのソースコードを読んでください。ネットワークを搭載したアプリケーションを作成したい場合は、通信 API の解説と server.c だけを参考にすれば手軽に開発できます。

【注意】これらプロトコルがすべての環境で動作することを保証するものではありません。あくまでサンプルプログラムとして動作、参照してください。

【注意】この AzkiTCP/IP は、OAKS16 シリーズのマイコンボードを開発するためのものです。それとは違うターゲットに搭載する場合、別途アンカーシステムズ株式会社より開発ライセンスをご購入ください。

６．２．レイヤー

AzkiTCP/IP 通信プロトコルスタックのレイヤ構成です。

図 6-2 レイヤ

<table>
<tr><td>アプリケーション層</td><td>HTTP サーバ</td><td>SMTP</td><td>POP3</td><td>DHCP</td><td>UDP サーバ</td></tr>
<tr><td>トランスポート層</td><td colspan="3">TCP</td><td colspan="2">UDP</td></tr>
<tr><td rowspan="2">ネットワーク層</td><td colspan="5">IP</td></tr>
<tr><td colspan="2">ICMP</td><td colspan="2">ARP</td></tr>
<tr><td>データリンク層</td><td colspan="5">RTL8019 デバイスドライバ</td></tr>
</table>

６．３．ファイル構成

表 6-3 構成ファイル一覧

ファイル名	内容
funcreg.h	M16C/62 マイコンボードの機能レジスタアドレスアサイン
ethernet.h	ethernet.c に関するヘッダファイル。LANBoard の IP アドレスなどコンフィギュレーションを含む。
ethernet.c	RTL8019AS デバイスドライバ、データリンク層、ネットワーク層を担当する TskNetworkLayer タスクと、パケット受信による割り込みハンドラ。ICMP,ARP、IP 処理、TCP 状態遷移、デフォルト Gateway に対する定期 ARP タスク。
ethertcp.c	TCP、UDP 通信端点（EndPoint）での API を提供。
server.h	TCP、UDP サーバ記述部ヘッダ
server.c	TCP,UDP の API 使用例として、TCP を使った HTTP サーバサンプル、UDP サーバサンプル。E メールで制御コマンド実行サンプル
mailer.h	E メールクライアント記述部ヘッダ
mailer.c	SMTP、POP3 クライアントとして E メールを送受信するサンプル。
dhcp.h	dhcp クライアントプロトコルヘッダ
dhcp.c	dhcp クライアントプロトコル。

7.通信 API

7．1．通信 API 特徴

● 通信 API はできるだけμITRON TCP/IP 設計仕様に合わせた名前にしています。

● AzkiRTOS 上に構成するため、すべて静的な API になっています。従って静的 API が大文字で記述されているわけではありません。

● 通信 API は、AzkiRTOS のタスクとして登録されたタスクの中からのみ呼び出し可能です。

● 階層間コピーはゼロです。

● 送受信 IP バッファは 1 つしか用意していません。

● TCP･UDP を使うアプリケーション層のタスク優先度は IP 層のタスクより高くしなければなりません。

● IP パケットの最大は 1460 バイトです。

● 1 つの通信端点 CEP に対して 1 つのタスクを対応させる必要があります。同じ CEP を複数のタスクで共有することはできません。

● エラーは E_TMOUT のみ返します。

● 通信端点（CEP:Communication End Point）は、プロトコルや動作機能別に持つ必要があります。これはソケットのようなものですが、AzkiTCP/IP では、CEP というグローバル変数の構造体として実装されています。

● ブロッキングされる API にはタイムアウト引数があります。

● ノンブロッキングコールによるコールバック方式の実装はされていません。

● タイムアウト値は、AzkiRTOS 側のシステム時間設定に依存しています。（デフォルトで 1 単位 10mSec です。）

● TCP での TCP 受付口（REP:Reception Point）の概念はありません。CEP を TCP と UDP で共通に使っています。

● CEP の ID 自動割当はできません。プログラムで指定します。

● CEP は必要個数、コンフィギュレーションで静的に確保しておく必要があります。

● TCP では、データをストリームとして扱っていません。シーケンス番号による再送要求やウィンドウ制御を行いません。あくまでもパケットデータとしての扱いです。

7．2．コンフィギュレーション

AzkiTCP/IP のコンフィギュレーション情報は、ethernet.h にあります。

図 7-2　コンフィギュレーション項目

```
//////////////////////////////////////////////////////////
// Azki TCPIP definition
//////////////////////////////////////////////////////////
#define DO_DHCP         FALSE               //「TRUE」の場合DHCPクライアントになる。

//DO_DHCP   FALSE の場合
#define MYIPADDRESS     {192,168,6,211}     //CPUボードのIPアドレス
#define MASKADDRESS     {255,255,255,0}     //ネットマスク
#define GWIPADDRESS     {192,168,6,254}     //デフォルトGatewayのIPアドレス

#define CEP_MAX         6                   //EndPoint数（TCP/UDP ソケット数）
#define ARPTBL_MAX      CEP_MAX*2           //IP-MAC テーブル数

#define TCP_RX_WINDOW_SIZE      1024-54             //TCP受信可能サイズ
#define BUF_SIZE_RX     TCP_RX_WINDOW_SIZE + 54     //Etherパケット受信バッファサイズ

// HTTP layer
#define     HTTP_STR_MAX        1460    //HTTP（TCP）で一回で送信できる制限バイト数。
                                        //受信ではHTTPヘッダなどを含んだクエリ受信枠。
```

表 7-2 コンフィギュレーション項目

DO_DHCP	DHCP クライアントとして動作するかどうか。DHCP クライアントになる場合は以下 3 項目は無効です。
MYIPADDRESS	OAKS16-LANBoard に設定する IP アドレスです。
MASKADDRESS	所属するネットワークのネットマスクです。
GWIPADDRESS	所属するネットワークのデフォルトゲートウエイの IP アドレスです。
CEP_MAX	通信端点 CEP の最大数です。通信種類ごとに 1 つ必要です。
ARPTBL_MAX	IP アドレスと MAC アドレスを一時記憶するテーブルの最大数です。
TCP_RX_WINDOW_SIZE	TCP ハンドシェークの送受信サイズです。
BUF_SIZE_RX	IP パケットの受信バッファサイズです。
HTTP_STR_MAX	HTTP で使う送受信文字列サイズです。

７．３．通信 API で使う構造体

相手側 IP アドレス/PORT 番号構造体

```
typedef struct
{
        UW      ipaddr;                 //IP Address
        UH      portno;                 //Port number
} T_IPV4EP;
```

通信端点 CEP(Communication End Point)

```
typedef struct
{
        //cre_cep で設定
        ID      Id;             //自局プログラムの TaskID
        UH      SrcPort;        //自局プログラムの Port 番号
        UB      Type;           //プロトコルタイプ TCP/UDP
        //自局サーバ時：IP 受信で設定、自局クライアント時：自分で設定
        UW      DestIP;         //相手の IP
```

```
UH      DestPort;       //相手の PORT 番号
UB*     pDATA;          //UDP/TCP データ（IPP）へのポインタ
INT     DataLen;        //UDP/TCP データ長さ
//for TCP
UB      State;          //State   TCP 状態遷移用
UH      SegSize;        //送信セグメントサイズ（WINDOW_SIZE で初期化）
UW      SrcSeqNo;       //シーケンス番号（乱数で初期化）
UW      AckSeqNo;       //相手への ACK 番号
```

７．４．エラーコード

AzkiTCP/IP で使っているエラーコードは以下だけです。（すべて azrtos.h 内での定義）

```
#define  E_OK           0               /* 正常終了 */
#define  E_SYS          -5              /* システムエラー */
#define  E_TMOUT        -85             /* タイムアウト */
```

７．５．API 詳細

一般

vini_tcpip

【形式】

```
ER vini_tcpip();
```

【引数】

【戻値】

【解説】

イーサネット関連のハードウエア、ソフトウエアの初期化を行います。

GetMAC

【形式】

```
INT         GetMAC( UB* pMAC , const UB* pIP)
```

【引数】

UB* pMAC	取得した MAC アドレスを格納するバッファへのポインタ。6 バイト必要です。
UB* pIP	調べる IP アドレスの入った配列へのポインタ。

【戻値】

INT	IP アドレスに対応する MAC アドレスがあれば 1。無ければ 0

【解説】

指定した IP アドレスに対応する MAC アドレスを ARP テーブルから検索して指定バッファに格納します。ARP テーブルにないときは戻値に 0 を返します。

RequestARP

【形式】

```
void        RequestARP(const UB*       pIP)
```

【引数】

UB* pIP　　ARP 要求する IP アドレスへのポインタ。

【戻値】

【解説】

指定した IP アドレスの ARP REQUEST を送信します。対応した REPLY が受信できたかどうかは検出しません。結果が受信できていれば ARP テーブルに格納されています。

---

htonl　htons　ntohl　ntohs

【形式】

```
UW          htonl(const UW hl);

UH          htons(const UH hs);

UW          ntohl(const UW nl);

UH          ntohs(const UH ns);
```

【引数】

UW (long) または UH (short) の長さの変換前の値

【戻値】

UW (long) または UH (short) の長さの変換後の値

【解説】

ホストバイトオーダーとネットワークバイトオーダーの変換をします。

htonl ホスト→ネットワークバイトオーダー変換 long

htons ホスト→ネットワークバイトオーダー変換 short

ntohl ネットワーク→ホストバイトオーダー変換 long

ntohs ネットワーク→ホストバイトオーダー変換 short

UDP 通信 API

udp_cre_cep

【形式】

ER udp_cre_cep(T_CEP* pCep, const UH port , const ID task)

【引数】

T_CEP* pCep	割り当てる通信端点 CEP へのポインタ
UH port	サーバ側待ち受けポート番号
ID task	タスク ID

【戻値】

ER	エラーコード

【解説】

UDP 通信で使う通信端点 CEP を生成します。生成といえども、静的に CEP は用意されていますので、どの CEP を使うかという指定と、UDP 通信で使うポート番号、UDP 通信を使う AzkiRTOS のタスク ID を設定します。以後、受信 API のブロック状態で、UDP 通信の待ち受けポート番号のデータが来たときに、受信されたデータがブロックされていた API に送られ、タスクが起床されます。

ここで登録するタスクはタスクの優先度をネットワーク層受信タスク TskNetworkLayer より高くしておく必要があります。

udp_rcv_dat

【形式】

ER udp_rcv_dat(T_CEP* pCep, T_IPV4EP* p_sipv4,

VP data, const INT len ,const TMO tmout)

【引数】

T_CEP* pCep	割り当てられた通信端点 CEP へのポインタ
T_IPV4EP* p_sipv4	相手側 IP アドレス/PORT 番号構造体へのポインタ
VP data	受信データを格納するためのバッファへのポインタ
INT len	最大受信バイト数
TMO tmout	タイムアウト指定

【戻値】

ER	受信データバイト数。負の値はエラーコード

【解説】

UDP 通信で指定ポート番号のデータを受信します。

最大受信バイト数で指定したバイト以下のデータがバッファに格納されます。

タイムアウト指定は、正の数字は、1 単位 10mSec（AzkiRTOS システム単位時間）、TMO_POL：ポーリング 、TMO_FEVR：タイムアウトなし、のいずれかを指定します。TMO_NBLK：ノンブロッキング、はサポートしていません。

相手側 IP アドレス/PORT 番号構造体に受信したデータを送信したクライアントの IP アドレスとポート番号が入っていますから、その情報を使って次のデータ送信を行います。

この API コールはキューイングできません。つまり同じ通信端点 CEP に対し同じ API を呼び出すことはできません。

---

udp_snd_dat

【形式】

```
ER udp_snd_dat(T_CEP* pCep, const T_IPV4EP* psiv4,
                                const VP data, const INT len ,const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てられた通信端点 CEP へのポインタ
T_IPV4EP* psipv4	相手側 IP アドレス/PORT 番号構造体へのポインタ
VP data	送信データバッファへのポインタ
INT len	送信バイト数
TMO tmout	タイムアウト指定（無効）

【戻値】

ER	送信データバイト数。負の値はエラーコード

【解説】

UDP 通信でデータを指定ポート番号で送信します。

相手側 IP アドレス/PORT 番号構造体に送信したいクライアントの IP アドレスとポート番号を入れてデータ送信を行います。

この API コールはキューイングできません。つまり同じ通信端点 CEP に対し同じ API を呼び出すことはできません。

TCP 通信 API

tcp_acp_cep

【形式】

```
ER tcp_acp_cep(T_CEP* pCep, T_IPV4EP* p_sipv4,
                    const UH svrport , const ID task ,const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てる通信端点 CEP へのポインタ
T_IPV4EP* p_sipv4	相手側 IP アドレス/PORT 番号構造体へのポインタ
UH svrport	サーバ側待ち受けポート番号
ID task	タスク ID
TMO tmout	タイムアウト指定

【戻値】

ER	エラーコード

【解説】

TCP 通信で使う通信端点 CEP を生成し、接続要求を待ちます。生成といえども、静的に CEP は用意されていますので、どの CEP を使うかという指定と、TCP 通信で使うポート番号、TCP 通信を使う AzkiRTOS のタスク ID を設定します。以後、受信 API のブロック状態で、TCP 通信の待ち受けポート番号のデータが来たときに、受信されたデータが、ブロックされていた API に送られ、タスクが起床されます。AzkiTCP/IP では、TCP 受付口の REP を特に持っていません。これを省略し、TCP のデータを単にパケット単位で扱い、静的な CEP を通信端点として使います。CEP は TCP の状態遷移を含み TCP 受信 API と連動します。

ここで登録する API を使う AzkiRTOS のタスクはタスクの優先度をネットワーク層受信タスク TskNetworkLayer より高くしておく必要があります。

tcp_rcv_dat

【形式】

```
ER tcp_rcv_dat(T_CEP* pCep, VP data, const INT len ,const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てられた通信端点 CEP へのポインタ
VP data	受信データを格納するためのバッファへのポインタ
INT len	最大受信バイト数
TMO tmout	タイムアウト指定

【戻値】

ER	受信データバイト数。負の値はエラーコード

【解説】

TCP 通信でデータを受信します。

最大受信バイト数で指定したバイト以下のデータがバッファに格納されます。AzkiTCP/IP では TCP をストリームとして扱っていません。あくまでもパケット単位のデータです。

タイムアウト指定は、正の数字は、1 単位 10mSec（AzkiRTOS システム単位時間）、TMO_POL：ポーリング 、TMO_FEVR：タイムアウトなし、のいずれかを指定します。TMO_NBLK：ノンブロッキング、はサポートしていません。

この API コールはキューイングできません。つまり同じ通信端点 CEP に対し同じ API を呼び出すことはできません。

---

tcp_snd_dat

【形式】

```
ER tcp_snd_dat(T_CEP* pCep, const VP data, const INT len ,const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てられた通信端点 CEP へのポインタ
VP data	送信データバッファへのポインタ
INT len	送信バイト数
TMO tmout	タイムアウト指定（無効）

【戻値】

ER	送信データバイト数。負の値はエラーコード

【解説】

TCP 通信でデータを送信します。

この API コールはキューイングできません。つまり同じ通信端点 CEP に対し同じ API を呼び出すことはできません。

---

tcp_con_cep

【形式】

```
ER      tcp_con_cep(T_CEP* pCep,  const T_IPV4EP* p_srcip ,
                    const T_IPV4EP* p_destip ,  const ID task , const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てる通信端点 CEP へのポインタ
T_IPV4EP* p_srcip	source 側 IP アドレス/PORT 番号構造体へのポインタ
T_IPV4EP* p_destip	destination 側 IP アドレス/PORT 番号構造体へのポインタ
UH svrport	サーバ側待ち受けポート番号
ID task	タスク ID
TMO tmout	タイムアウト指定（無効）

【戻値】

ER	エラーコード

【解説】

TCP クライアントとして通信で使う通信端点 CEP を生成し、サーバに接続を試みます。静的に CEP は用意されていますので、どの CEP を使うかという指定と、TCP 通信で使うポート番号、サーバ側の IP、待ち受けポート番号、TCP 通信を使う AzkiRTOS のタスク ID を設定します。コネクション成立後は、tcp_rcv_dat()、tcp_snd_dat() でデータを送受信できます。データはストリームになっていません。すべてパケット単位です。クライアント側からの通信切断は tcp_cls_cli() を使います。

ここで登録する API を使う AzkiRTOS のタスクはタスクの優先度をネットワーク層受信タスク TskNetworkLayer より高くしておく必要があります。

---

tcp_cls_cep

【形式】

```
ER      tcp_cls_cep(T_CEP* pCep ,const TMO tmout)
```

【引数】

T_CEP* pCep	割り当てられた通信端点 CEP へのポインタ
TMO tmout	タイムアウト指定（無効）

【戻値】

ER	エラーコード

【解説】

接続していた TCP 通信のコネクションを切断します。

タイムアウト指定は、正の数字は、1 単位 10mSec（AzkiRTOS システム単位時間）、TMO_POL：ポーリング 、TMO_FEVR：タイムアウトなし、のいずれかを指定します。TMO_NBLK：ノンブロッキング、はサポートしていません。

tcp_rst_cep

【形式】

ER　　tcp_rst_cep (T_CEP* pCep)

【引数】

T_CEP* pCep　　割り当てられた通信端点 CEP へのポインタ

【戻値】

ER　　エラーコード

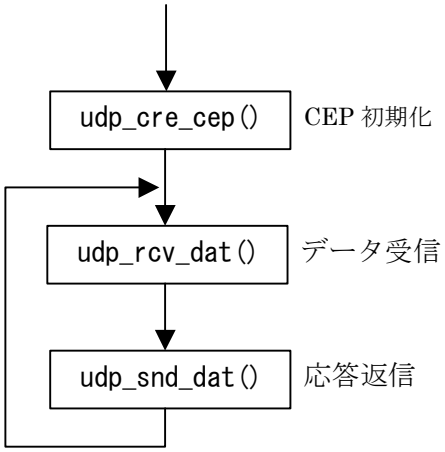
【解説】

TCP 通信相手に RST を送信して強制的にコネクション切断を試みます。

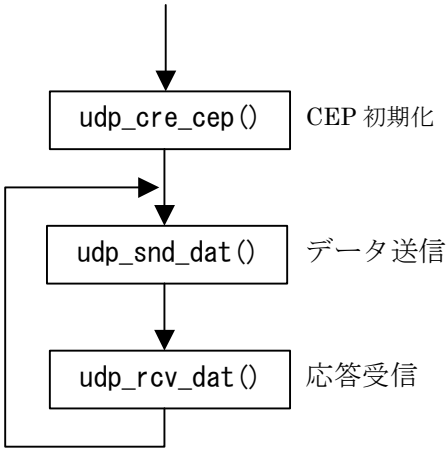
OAKS16 では同じポート番号を使い続けるので、サーバが TIME_WAIT 状態になっているときに再接続できなくなります。通信を close した後に RST を送っておくことで TIME_WAIT を解除する目的で使えます。

図 7-5 API 呼び出し例

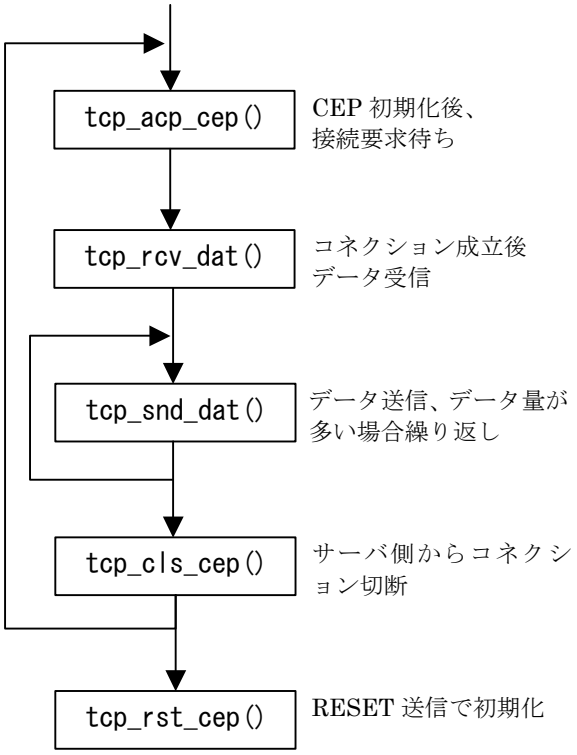
UDP サーバ実装



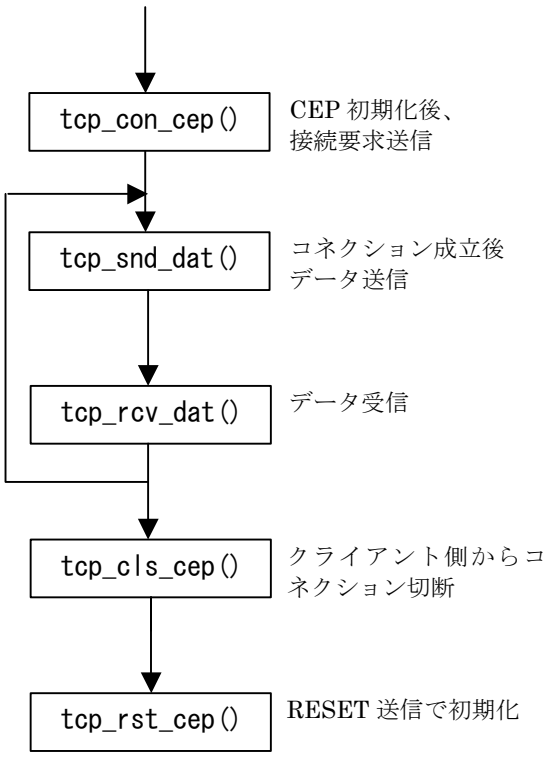
UDP クライアント実装



TCP サーバ実装



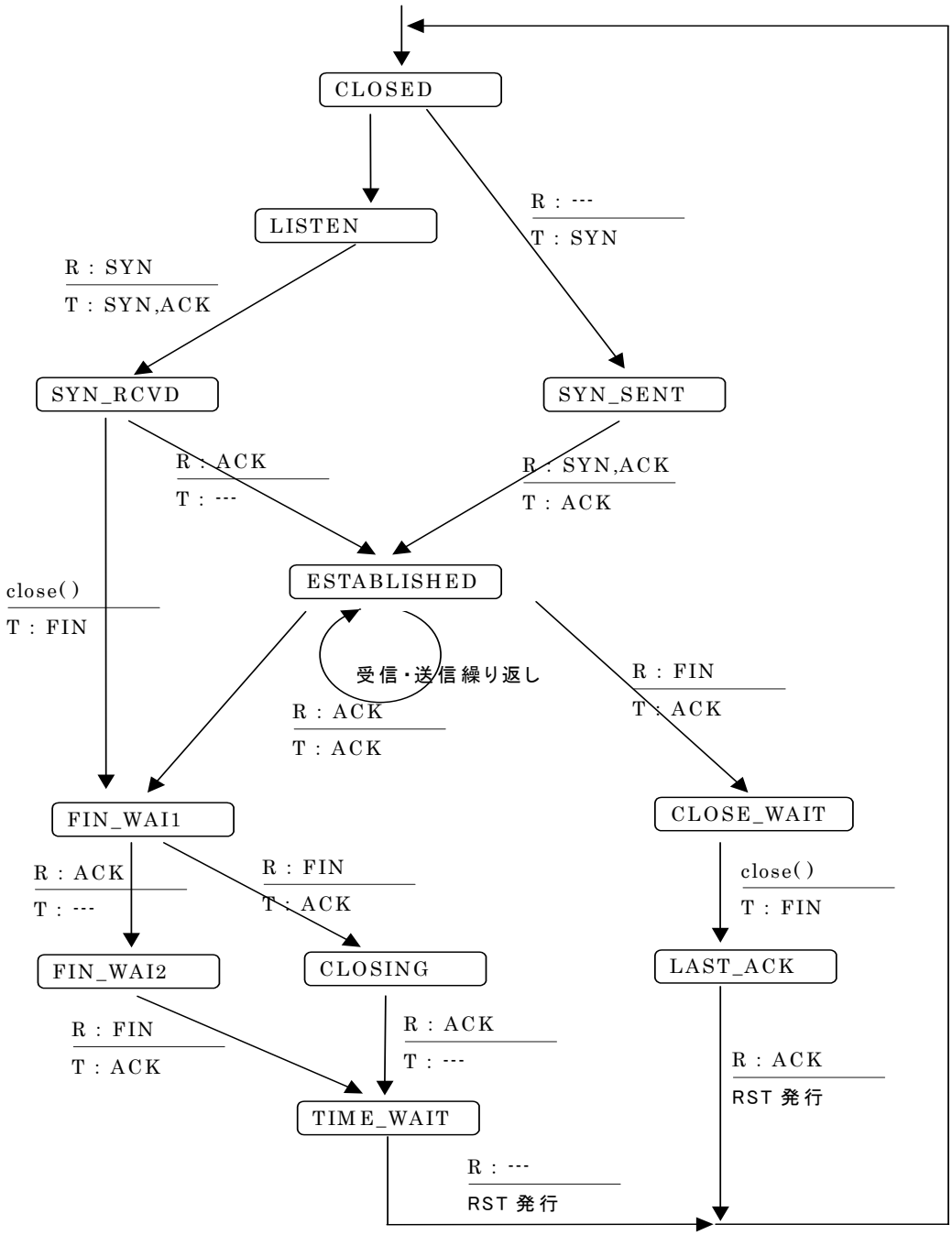
TCP クライアント実装



７．６．AzkiTCP 状態遷移

【ヒント】AzkiTCP/IP の TCP 状態遷移は、必要最小限の動作のみ実装されています。コネクション成立から受信送信をパケット単位で行い最後にクローズです。パケットデータはストリームではなくあくまでパケット単位での処理となります。またシーケンス番号、ウインドウ処理は行っていません。

図 7-6　AzTCP 状態遷移図



７．７．イーサネットパケットフレーム図

【ヒント】IP バッファ内の生データを解析したいときに便利な表です。フレームデータのオフセット値はイーサネットフレームの先頭からの値です。AzkiTCP の各関数に渡されているポインタのオフセットは関数の先頭に記述があります。

【ヒント】データ部： IP データグラム：46～1500 オクテット、ARP：28 オクテット

図 7-5 OAKS16-LAN Board　Ethernet フレーム Dump

EtherFrame － ARP

	+0	+2	+4	+6	+8	+A	+C	+E
00	宛先 MAC アドレス			発信元 MAC アドレス			0806:ARP	0001
10	0800	0604	0001:要求 0002:応答	発信元 MAC アドレス			発信元 IP アドレス	
20	宛先 MAC アドレス			宛先 IP アドレス				

EtherFrame － IP － ICMP (Echo)

	+0	+2	+4	+6		+8	+A	+C	+E
00	宛先 MAC アドレス			発信元 MAC アドレス				0800:IP	4500
10	全データ長	識別子 ++	フラグメント	TTL	1:ICMP	チェックサム	発信元 IP アドレス		宛先 IP⇔
20	⇔アドレス	0800:要求 0000:応答	チェック サム	識別子		シーケンス 番号	Ping データ		

EtherFrame － IP － UDP

	+0	+2	+4	+6		+8	+A	+C	+E
00	宛先 MAC アドレス			発信元 MAC アドレス				0800:IP	4500
10	全データ長	識別子 1づつ増加	フラグメント	TTL	17:UDP	チェックサム	発信元 IP アドレス		宛先 IP⇔
20	⇔アドレス	発信元 ポート番号	宛先 ポート番号	UDP データ長		チェックサム	以下データ		

EtherFrame － IP － TCP

	+0	+2	+4	+6		+8	+A	+C	+E
00	宛先 MAC アドレス			発信元 MAC アドレス				0800:IP	4500
10	全データ長	識別子 1づつ増加	フラグメント	TTL	6:TCP	チェックサム	発信元 IP アドレス		宛先 IP⇔
20	⇔アドレス	発信元 ポート番号	宛先 ポート番号	シーケンス番号			確認応答番号		ヘッダ長 フラグ
30	ウインドウ サイズ	チェックサム	緊急 ポインタ	オプション(可変長)　データ					

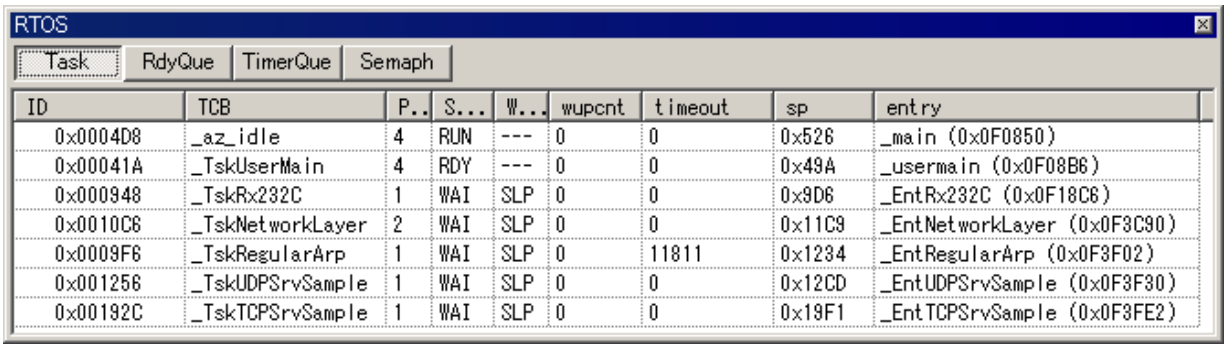
Appendix

A. 統合開発環境 AnchorPlace Lite について。

この LANBoard は AzkiRTOS の環境下で動作しています。従って通信アプリケーションは AzkiRTOS のタスクとして動作する必要があります。

AnchorPlace Lite は、AzkiRTOS のタスクの状態をモニタできるデバッガを搭載しています。現在どのタスクが RUN 中か、レディキューにどのタスクが入っているか、などタスクの状態がモニタできます。

図 A-1 RTOS ウインドウの例



ID	TCB	P..	S...	W...	wupcnt	timeout	sp	entry
0x0004D8	_az_idle	4	RUN	---	0	0	0x526	_main (0x0F0850)
0x00041A	_TskUserMain	4	RDY	---	0	0	0x49A	_usermain (0x0F08B6)
0x000948	_TskRx232C	1	WAI	SLP	0	0	0x9D6	_EntRx232C (0x0F18C6)
0x0010C6	_TskNetworkLayer	2	WAI	SLP	0	0	0x11C9	_EntNetworkLayer (0x0F3C90)
0x0009F6	_TskRegularArp	1	WAI	SLP	0	11811	0x1234	_EntRegularArp (0x0F3F02)
0x001256	_TskUDPSrvSample	1	WAI	SLP	0	0	0x12CD	_EntUDPSrvSample (0x0F3F30)
0x00192C	_TskTCPSrvSample	1	WAI	SLP	0	0	0x19F1	_EntTCPSrvSample (0x0F3FE2)

また、統合環境ですから、エディット、ビルド、デバッグをシームレスに行うことができるようになります。RTOS を使ったプログラム開発や、複雑で大きなプログラム開発には必須のバージョン管理、インスペクタ、関数・変数定義参照クロスリファレンスなどが使える環境です。

【ヒント】AnchorPlace Lite は、AnchorPlace の動作を OAKS16 シリーズに限定したもので、それ以外の制限はありません。 （価格 ¥9,800） 詳しい情報は、アンカーシステムズ株式会社のホームページ http://www.anchorsystems.co.jp/ **をご覧ください。**

ご購入窓口

■オークス電子株式会社 mailto:moritake@oaks-ele.com へご連絡

■アンカーシステムズ株式会社 http://www.anchorsystems.co.jp/ からオンライン購入

■お急ぎの場合は、ライセンスキーの即時発行可能なダウンロード販売 ベクター株式会社

http://sw.vector.co.jp/swreg/detail.info?srno=SR029467 へアクセスしてください。

Ｂ．LANBoard 改造

OAKS16-LANBoard はワンチップマイコン M30620 のポート P1,P7,P8 を使っています。ユーザアプリケーションを設計する上でこれらポートから移動させたいときは、LANBoard のパターンをカットしてジャンパ線等で移動先へ配線してください。尚、LANBoard 上必要な配線上にはティアドロップ型のランドを付けています。

マイコンボードとの接続ランドはすべてスルーホールメッキになっていませんので、マイコンボードと LANBoard を分離する場合は半田吸い取り線などで丁寧に半田を吸い取ってください。簡単にはずすことができます。

【ヒント】本 LANBoard を OAKS16-LCDBoard と接続すると、LCD ボードの LED で使っているポートと競合します。無意味に LED が点滅してもかまわない場合はそのまま使えますが、LCD ボードの LED を有効に使いたい場合は、競合するポートを移動することで使えます。

Ｃ．MAC アドレス

本 OAKS16-LANBoard には、正規の MAC アドレスが EEPROM に書き込まれています。従って IP アドレスとしてグローバルアドレスを付与することも可能です。また誤って EEPROM を消去してしまった場合 MAC アドレスだけの再発行はできません。

OAKS16-LANBoard ユーザーズマニュアル Ver. 2.4

2007 年 3 月 15 日 発行

編集　　アンカーシステムズ株式会社

発行　　オークス電子株式会社