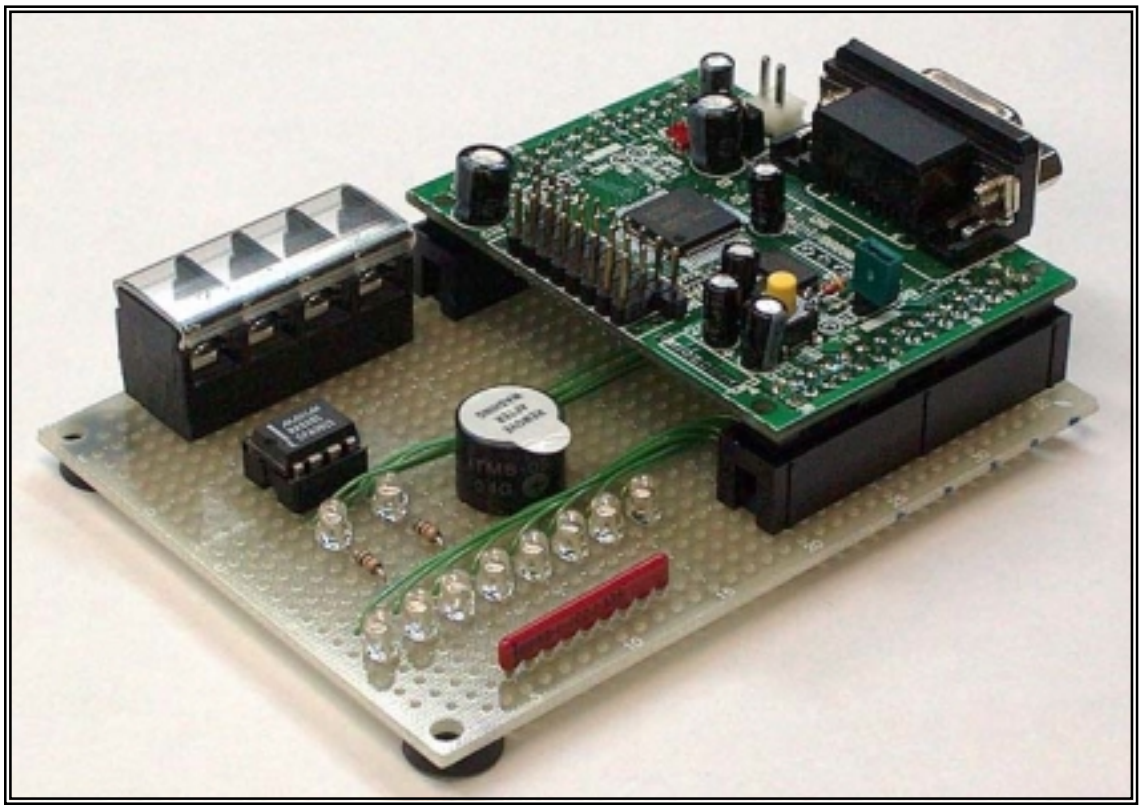
TK-3687miniオプションキット

RS-485実習キット

Version 1.01



はじめに

回路図と部品表

組み立て

動作確認

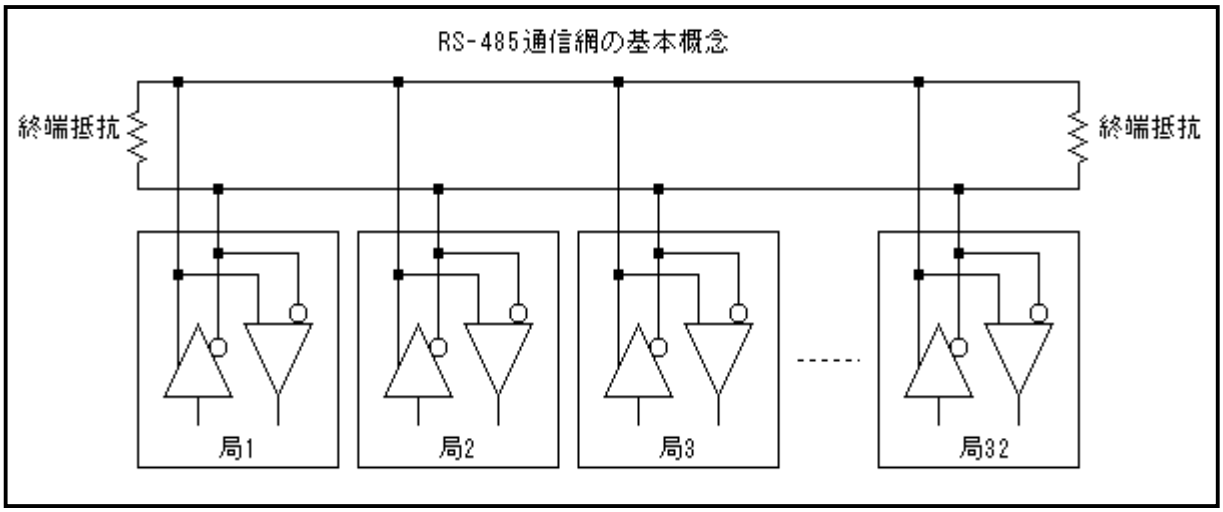
サンプルプログラム

(株)東洋リンクス

はじめに

少し前はどのパソコンにもシリアルポートが実装されていました。パソコンのシリアルポートは RS-232C という規格です。1 対 1 で接続するためによく使われました。しかし，伝送速度が遅く，不平衡伝送のためノイズに弱いという欠点があり，最近の高速・長距離伝送に対応できなくなりました。

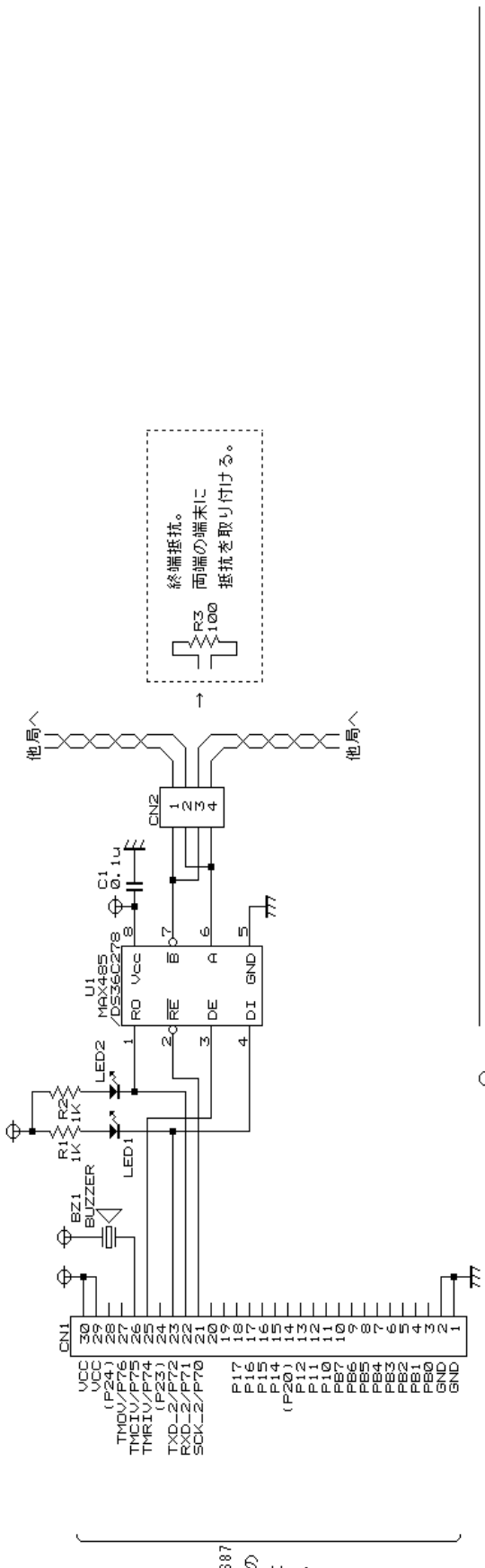
このキットで学習する RS-485 はこれらの欠点を改善した規格です。規格上は 100K ビット/s で 1.2km まで（距離を短くすればもっと早くできる）接続できます。また，平衡伝送を採用しているためノイズにも強くなりました。さらに，バス方式に対応し 1 本のライン上に複数の端末を接続できるようになりました。



このキットは RS-485 の基本的な動作を学習するための教材です。サンプルプログラムとして，送信データを自局で受信するセルフチェックプログラムと，H8/3687 のマルチプロセッサ通信機能を利用したセルフチェックプログラムを載せてあります。セルフチェックプログラムなので 1 台だけで動作します。（MAX485 のドライバとレシーバを両方イネーブルにすることで,自分で送信したデータを自分で受信します。）

シリアル通信の詳細については「H8/3687 シリーズ　ハードウェアマニュアル」の「16. シリアルコミュニケーションインタフェース（SCI3）」をご覧下さい。

回路図と部品表



部品番号	型番	メーカ	個数	備考
U1	MAX485 or DS36C278	MAXIM or NS	1	相当品可。RS-485トランシーバ。
R1	1KΩ		1	
R2	1KΩ		1	
R3	100Ω		1	梱外参照(*1)。
C1	0.1μF(セラミック)		1	パスコン。
RA1	M9-1-102J	BIテクノロジー	1	相当品可。1KΩ,8素子入り。
LED1			1	送信インジケータ,高輝度タイプ。
LED2			1	受信インジケータ,高輝度タイプ。
LED3-10			8	データ表示,高輝度タイプ。
CN1,3	HIF3FC-30PA-2.54DSA	HRS	2	相当品可。TK-3687miniとスタッキング。
CN2	端子台(4端子)		1	梱外参照(*3)。
BZ1	CPM121A0A or TMB05	メガセラ or スター精機	1	相当品可。
基板			1	ユニバーサル基板。
	ゴム足		4	
	ラッピングケーブル		1m	配線用。
	メッキ線			Vcc,GND等ハンダ面結線用,梱外参照(*2)。

*1:R3は通信線の両端になるキットの端子台に取り付けます。このマニュアルのサンプルプログラムは未実装でも動作します。
*2:ラッピングケーブルの被服をはがし2本をよじって使用します。また,抵抗やコンデンサの足も流用できます。
*3:ユニバーサル基板のスルーを1.2φのドリルで広げて実装します。

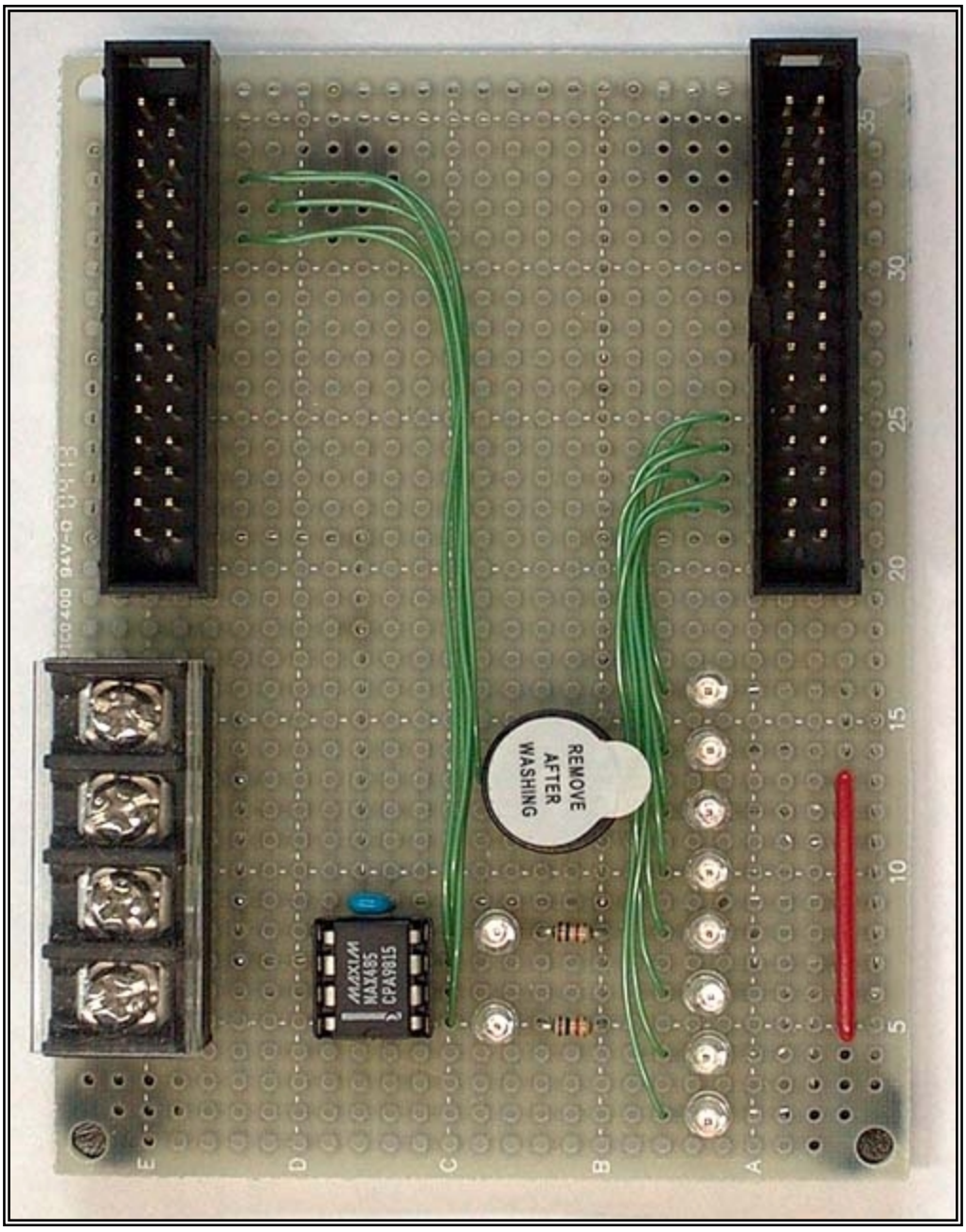
組み立て

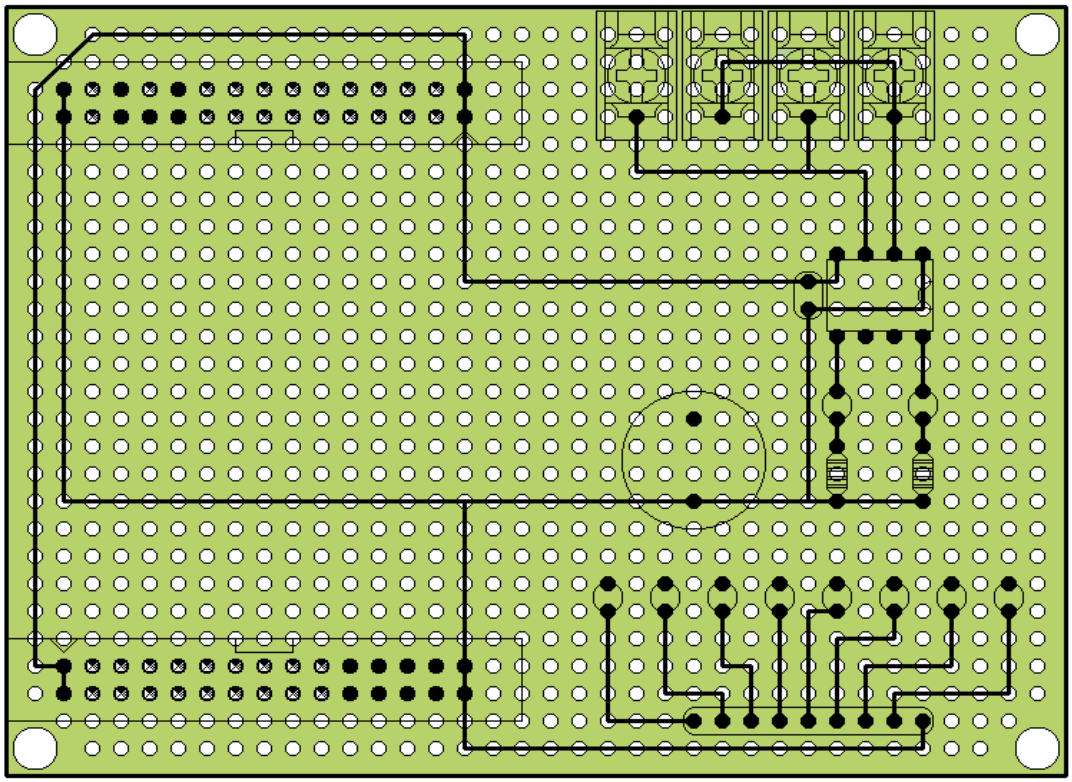
ユニバーサル基板にハードを組み上げます。プリント基板と異なり，ユニバーサル基板は全ての配線を自分で結線しなければなりません。回路図を見ながら部品をハンダ付けしていくことによってハードの構成を理解していきましょう。

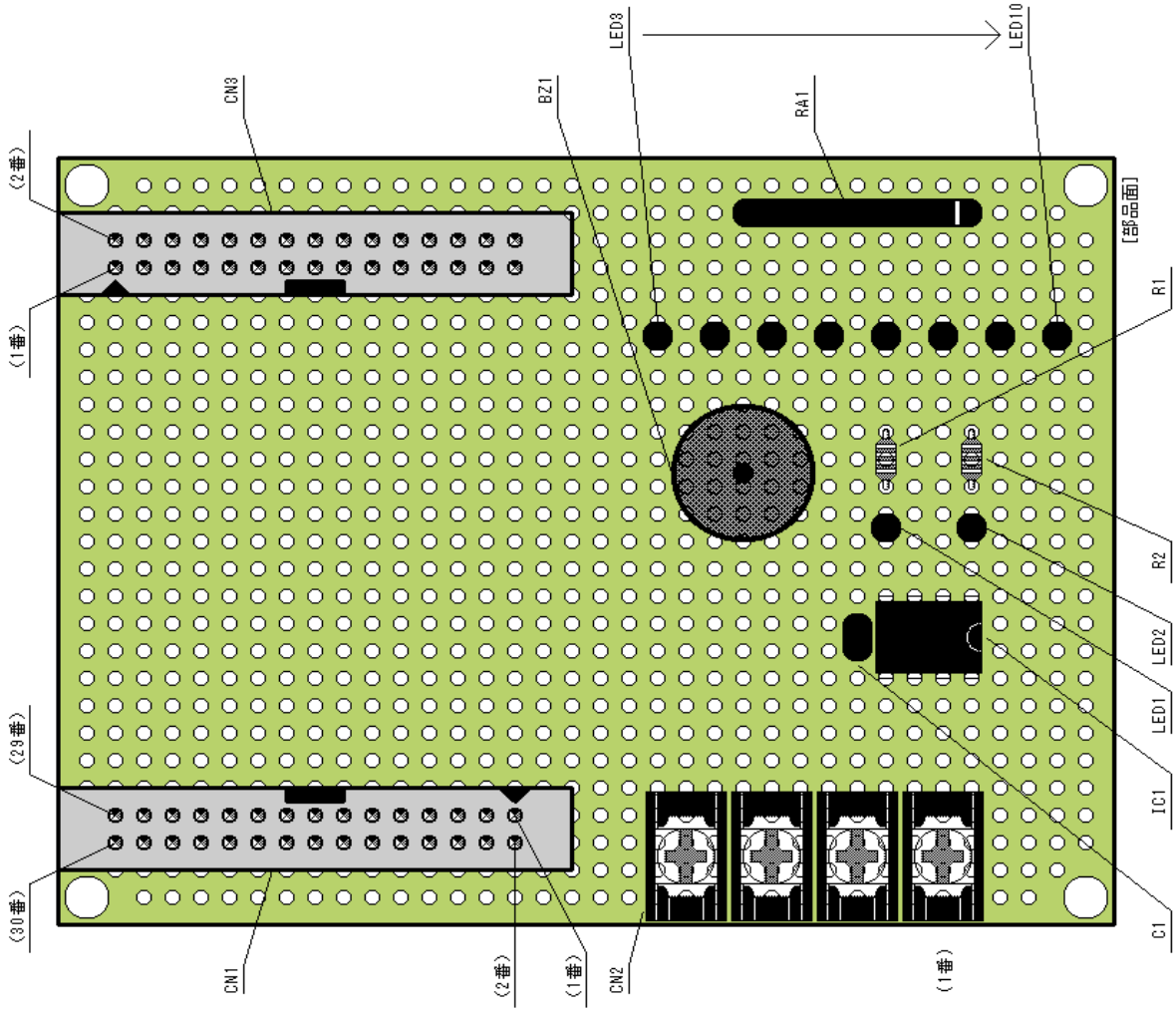
まず工具と部品の確認を行ないます。回路図中の部品表と照らし合わせて全ての部品がそろっているか確認して下さい。用意する工具は下記の通りです。

ハンダゴテ，ハンダ，ニッパ，ワイヤストリッパ，ピンセット，テスタ

部品の確認が済みましたら，いよいよ組み立てです。回路図を見ながらハンダ付けを行なってください。電源や GND，交差しない信号線などはハンダ面でメッキ線や部品の余ったリード線を使って接続していきます。ハンダ面だけでは配線できない信号線はラッピングケーブルで配線していきます。下の写真と次ページの実装図を参考にして下さい。なお，端子台はユニバーサル基板の穴径では細すぎるので，あらかじめドリルで穴を広げています（1.2φ）。

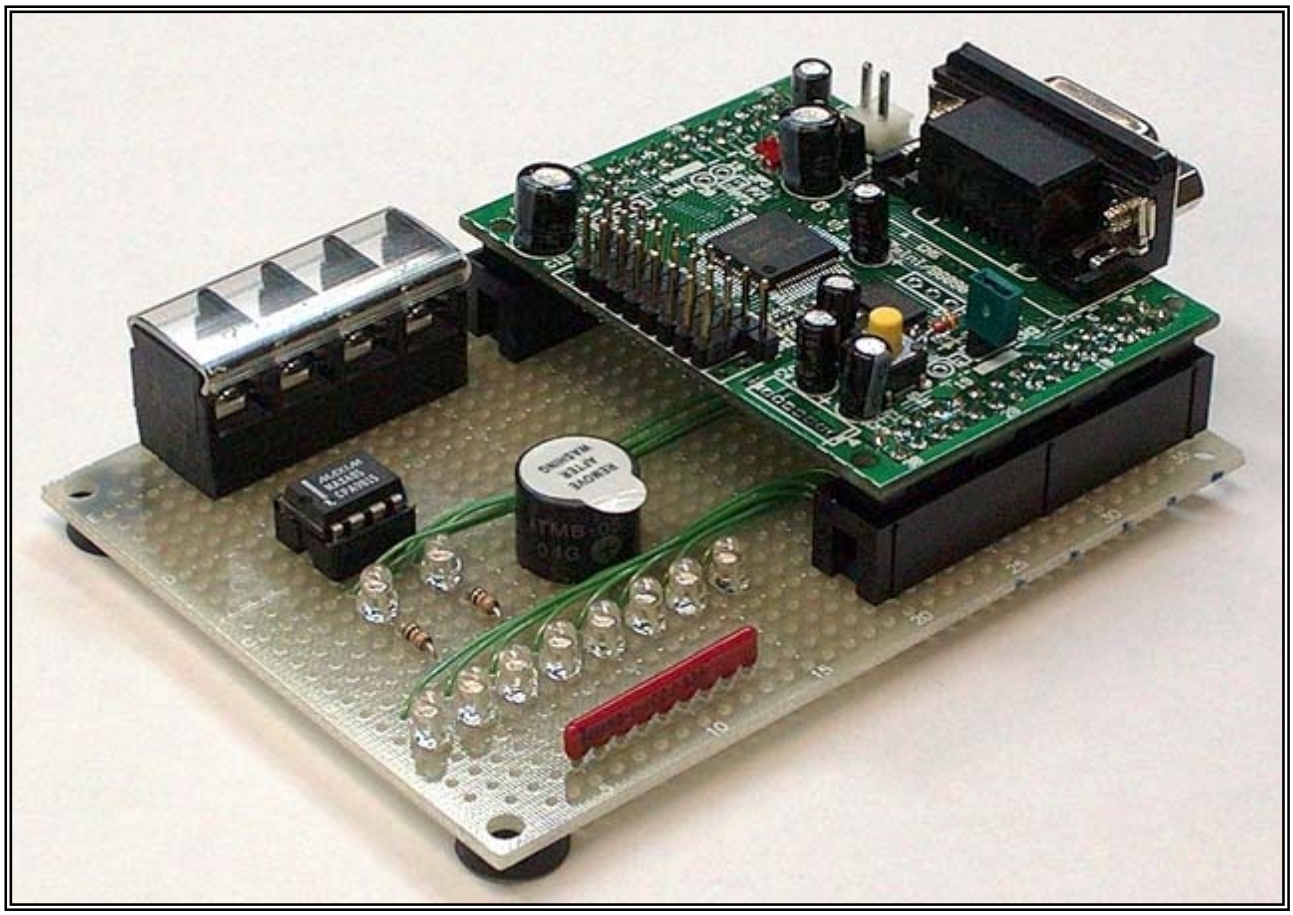



[半田面]


LED3
LED10
CN3
BZ1
RA1
(2番)
(1番)
[部品面]
R1
R2
LED2
LED1
IC1
C1
(29番)
(30番)
CN1
(2番)
(1番)
CN2
(1番)

■ **TK-3687mini との接続**

全てのハンダ付け作業が終了したら，TK-3687mini と接続します。下の写真を見て下さい。CN1 と CN3 を介して，TK-3687mini を 2 段重ねにします。取り付け方向に注意して下さい。



動作確認

配線が正しいか回路図と比較しながらもう一度確認して下さい。テスタの導通チェッカー機能を利用します。

間違いがないようなら動作確認を行ないます。パソコンと TK-3687mini を接続して，あらかじめ書き込まれているモニタプログラム，ハイパーH8 を起動します。モニタでファイルのロードと実行のコマンド“LG”を入力し，CD-ROM に収録されているプログラム（rs485_01.mot）をロード・実行します。ファイルの場所は“D:¥TK-3687mini¥ｵﾌﾟｼｮﾝ¥RS485¥ﾌﾟﾛｸﾞﾗﾑ”です（CD-ROM が D ドライブの場合）。LED に 2 進数で 00h～FFh まで順に表示され，LED が全点灯したときにブザーが「ピッ」となれば OK です。動作しないときはもう一度配線とハンダ付けを確認して下さい。なお，このプログラムの内容は次ページのサンプルプログラム-1 です。

サンプルプログラム-1

MAX485 のドライバとレシーバを同時にイネーブルにし，送信データを自分で受信します。受信したデータは LED に表示します。なお，送信データは送信するたびにインクリメントし，0 になったときにブザーを鳴らします。ファイル名は“rs485_01.mot”，ファイルの場所は“D:¥TK-3687mini¥ｵﾌﾟｼｮﾝ¥RS485¥ﾌﾟﾛｸﾞﾗﾑ”です（CD-ROM が D ドライブの場合）。

```
/************************************************************************/
/*                                                                      */
/*  TK-3687 & RS-485 テストプログラム                                   */
/*  送信データをそのまま受信し，LEDに受信データを表示する              */
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
/*  FILE        :rs485_01.c                                             */
/*  DATE        :Mon, Aug 30, 2004                                      */
/*  DESCRIPTION :Main Program                                           */
/*  CPU TYPE    :H8/3687                                                */
/*  BOARD       :TK-3687mini(B6090) and TK-3687(B6081)                  */
/*                                                                      */
/*  This file is programed by TOYO-LINX Co.,Ltd. / yKikuchi             */
/*                                                                      */
/************************************************************************/

/************************************************************************
    履歴
************************************************************************/
/*
2004-08-30 : 作成開始
2005-04-08 : TK-3687miniに対応
*/

/************************************************************************
    インクルードファイル
************************************************************************/
#include  <machine.h>
#include  "iodefine.h"

/************************************************************************
    関数の定義
************************************************************************/
void      ini_port(void);
void      ini_sci3_2(void);
void      main(void);

/************************************************************************
    メインプログラム
************************************************************************/
void main(void)
{
     unsigned char  TxData = 0x00;
     unsigned long  i;

//----- イニシャライズ ----------------------------------------------
```

```
    ini_port();
    ini_sci3_2();

//----- メインループ ----------------------------------------------------
    while(1){
        // 1データ送信
        while(P_SCI3_2.SSR2.BIT.TDRE==0){}
        P_SCI3_2.TDR2 = TxData;

        // 1データ受信,ポート5とポート6に表示する
        while(P_SCI3_2.SSR2.BIT.RDRF==0){}
        P_PORT.PDR5.BYTE = P_SCI3_2.RDR2;
        P_PORT.PDR6.BYTE = ~P_PORT.PDR5.BYTE;

        // 次の送信データをセット
        TxData++;

        // TxData=0x00のときブザーを鳴らす
        if (TxData==0x00){
            P_PORT.PDR7.BIT.P75 = 0;     //ブザーオン
            for (i=0; i<500000; i++){}
            P_PORT.PDR7.BIT.P75 = 1;     //ブザーオフ
        }

        // ウェイト
        for (i=0; i<50000; i++){}
    }
}

/************************************************************************
    I/O PORTの初期化
************************************************************************/
void ini_port(void)
{
    P_PORT.PMR5.BYTE   =   0x00;     //P50-57 使用
    P_PORT.PCR5.BYTE   =   0xff;     //P50-57 Out
    P_PORT.PDR5.BYTE   =   0x00;     //P50-57=Low

    P_PORT.PCR6.BYTE   =   0xff;     //P60-67 Out
    P_PORT.PDR6.BYTE   =   0xff;     //P60-67=High

    P_PORT.PCR7.BYTE   =   0x31;     //P70,74,75 Out
    P_PORT.PDR7.BYTE   =   0x30;     //P70(-RE)=Low,P74(DE)=High
                                     // ドライバ&レシーバ イネーブル
}

/************************************************************************
    SCI3の初期化
************************************************************************/
void ini_sci3_2(void)
{
    #define       MHz       20             // Clock=20MHz
    #define       BAUD      625000         // baudrate
    #define       BITR      (MHz*1000000)/(BAUD*32)-1
    #define       WAIT_1B   (MHz*1000000)/6/BAUD
```

```
    unsigned  long i;

    P_PORT.PMR1.BIT.TXD2    =    1;            // TXD_2端子イネーブル
    P_SCI3_2.SCR3_2.BYTE    =    0x00;         // 動作停止
    P_SCI3_2.SMR2.BYTE      =    0x00;         // 非同期･8bit･P-non･CLK/1
    P_SCI3_2.BRR2           =    BITR;         // ビットレート
    for  (i=0;i<WAIT_1B;i++){};                // 1bit期間 wait
    P_SCI3_2.SCR3_2.BYTE    =    0x30;         // 送受信イネーブル
}

/************************************************************************
    End of Program.
************************************************************************/
```

サンプルプログラム-2

H8/3687 のマルチプロセッサ通信機能を使用したセルフチェックプログラムです。マルチプロセッサ通信機能を使用すると，マルチプロセッサビットを付加した調歩同期式シリアル通信により複数のプロセッサ間で通信回線を共有してデータの送受信を行うことができ，RS-485 のバス方式の通信網に応用することができます。

マルチプロセッサ通信では受信局に各々固有の ID コードを割付けます。そして，送信局は受信局の ID コードにマルチプロセッサビット‘1’を付加したデータを送信し，続いてその受信局に送りたいデータにマルチプロセッサビット‘0’を付加して送信します。受信局はマルチプロセッサビットが‘1’のデータを受信すると自局の ID と比較し，一致した場合は続いて送信されるデータを受信します。一致しなかった場合は再びマルチプロセッサビットが‘1’のデータを受信するまでデータを読み飛ばします。H8/3687 の SCI3 にはマルチプロセッサビットが‘1’のデータを受信するまで受信データを読み飛ばす機能と，マルチプロセッサビットを付加する機能が内蔵されています。自局の ID コードの管理，及び受信した ID コードとの比較，ID コードが一致したときに続いて送信されるデータの受信はソフトウェアで行ないます。

このサンプルプログラムでは，MAX485 のドライバとレシーバを同時にイネーブルにし，送信データを自分で受信します。送信データの ID コードを 00h から FFh まで順番に変化させながら送信し，自局の ID コード（55h）を受信したときのデータを LED に表示しブザーを鳴らします。なお，送信データも順にインクリメントします。ファイル名は“rs485_02.mot”，ファイルの場所は“D:¥TK-3687mini¥ｵﾌﾟｼｮﾝ¥RS485¥ﾌﾟﾛｸﾞﾗﾑ”です（CD-ROM が D ドライブの場合）。

マルチプロセッサ通信機能の詳細については「H8/3687 シリーズ ハードウェアマニュアル」の「16. シリアルコミュニケーションインタフェース（SCI3）」，「16. 6 マルチプロセッサ通信機能」をご覧下さい。

```
/*****************************************************************************/
/*                                                                           */
/*  TK-3687 & RS-485 テストプログラム                                        */
/*  マルチプロセッサ通信機能のセルフチェック                                 */
/* ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ */
/*  FILE        :rs485_02.c                                                  */
/*  DATE        :Wed, Sep 01, 2004                                           */
/*  DESCRIPTION :Main Program                                                */
/*  CPU TYPE    :H8/3687                                                     */
/*  BOARD       :TK-3687mini(B6090) and TK-3687(B6081)                       */
/*                                                                           */
/*  This file is programed by TOYO-LINX Co.,Ltd. / yKikuchi                  */
/*                                                                           */
/*****************************************************************************/

/*****************************************************************************
    履歴
*****************************************************************************/
/*
2004-09-01 : 作成開始
2005-04-08 : TK-3687miniに対応
*/

/*****************************************************************************
    インクルードファイル
*****************************************************************************/
#include  <machine.h>
```

```
#include "iodefine.h"

/************************************************************************
    変数の定義
************************************************************************/
unsigned char TxData = 0x00;     //送信データ
unsigned char RxData;            //受信データ
unsigned char TxId = 0x00;       //送信IDコード
unsigned char MyId = 0x55;       //自局IDコード
unsigned char RxId;              //受信IDコード
unsigned char TxStatus = 0;      //送信ステータス
                                 //  0:IDコード送信
                                 //  1:データ送信
                                 //  3-FF:次の送信待ち
unsigned char RxStatus = 0;      //受信ステータス
                                 //  0:IDコード受信待ち
                                 //  1:自局データ受信待ち

/************************************************************************
    関数の定義
************************************************************************/
void     ini_port(void);
void     ini_sci3_2(void);
void     main(void);

/************************************************************************
    メインプログラム
************************************************************************/
void main(void)
{
    unsigned long i;

//----- イニシャライズ -------------------------------------------------
    ini_port();
    ini_sci3_2();

//----- メインループ ---------------------------------------------------
    while(1){
        switch (TxStatus){
            case 0:       //IDコード送信
                if (P_SCI3_2.SSR2.BIT.TDRE==1){
                    P_SCI3_2.SSR2.BIT.MPBT = 1;
                    P_SCI3_2.TDR2 = TxId;
                    TxId++;
                    TxStatus = 1;
                }
                break;
            case 1:       //データ送信
                if (P_SCI3_2.SSR2.BIT.TDRE==1){
                    P_SCI3_2.SSR2.BIT.MPBT = 0;
                    P_SCI3_2.TDR2 = TxData;
                    TxStatus = 2;
                }
                break;
            default: //次のデータ送信待ち
                TxStatus++;
```

```
                    if ((TxId==0x00)&&(TxStatus==0)) {TxData++;}        //送信データの更新
          }

          switch (RxStatus){
               case 0:       //IDコード受信
                    if (P_SCI3_2.SSR2.BIT.RDRF==1){
                         RxId = P_SCI3_2.RDR2;
                         if ((P_SCI3_2.SSR2.BIT.MPBR==1)&&(RxId==MyId)){ //自局IDコード受信
                              P_SCI3_2.SCR3_2.BIT.MPIE = 0;     //ﾏﾙﾁﾌﾟﾛｾｯｻｲﾝﾀﾗﾌﾟﾄﾃﾞｨｾｰﾌﾞﾙ
                              RxStatus = 1;
                         }
                         else{     //他局IDコード受信
                              P_SCI3_2.SCR3_2.BIT.MPIE = 1;     //ﾏﾙﾁﾌﾟﾛｾｯｻｲﾝﾀﾗﾌﾟﾄｲﾈｰﾌﾞﾙ
                         }
                    }
                    break;
               case 1:       //自局データ受信
                    if (P_SCI3_2.SSR2.BIT.RDRF==1){
                         RxData = P_SCI3_2.RDR2;
                         P_PORT.PDR5.BYTE = RxData;        //受信データをP50-57に表示
                         P_PORT.PDR6.BYTE = ~RxData;       //受信データをP60-67に表示
                         P_SCI3_2.SCR3_2.BIT.MPIE = 1;     // ﾏﾙﾁﾌﾟﾛｾｯｻｲﾝﾀﾗﾌﾟﾄｲﾈｰﾌﾞﾙ
                         RxStatus = 0;
                         P_PORT.PDR7.BIT.P75 = 0;     //ブザーオン
                         for (i=0; i<250000; i++){}
                         P_PORT.PDR7.BIT.P75 = 1;     //ブザーオフ
                    }
                    break;
          }
     }
}

/************************************************************************
     I/O PORTの初期化
************************************************************************/
void ini_port(void)
{
     P_PORT.PMR5.BYTE   =    0x00;     //P50-57 使用
     P_PORT.PCR5.BYTE   =    0xff;     //P50-57 Out
     P_PORT.PDR5.BYTE   =    0x00;     //P50-57=Low

     P_PORT.PCR6.BYTE   =    0xff;     //P60-67 Out
     P_PORT.PDR6.BYTE   =    0xff;     //P60-67=Low

     P_PORT.PCR7.BYTE   =    0x31;     //P70,74,75 Out
     P_PORT.PDR7.BYTE   =    0x30;     //P70(-RE)=Low,P74(DE)=High
                                       // ドライバ&レシーバ イネーブル
}

/************************************************************************
     SCI3の初期化
************************************************************************/
void ini_sci3_2(void)
{
     #define       MHz       20             // Clock=20MHz
     #define       BAUD      625000         // baudrate
```

```
    #define       BITR     (MHz*1000000)/(BAUD*32)-1
    #define       WAIT_1B  (MHz*1000000)/6/BAUD

    unsigned  long i;

    P_PORT.PMR1.BIT.TXD2    =    1;           // TXD_2端子イネーブル
    P_SCI3_2.SCR3_2.BYTE    =    0x00;        // 動作停止
    P_SCI3_2.SMR2.BYTE      =    0x04;        // 非同期･8bit･P-non･CLK/1･ﾏﾙﾁﾌﾟﾛｾｯｻﾓｰﾄﾞ
    P_SCI3_2.BRR2           =    BITR;        // ビットレート
    for  (i=0;i<WAIT_1B;i++){};               // 1bit期間 wait
    P_SCI3_2.SCR3_2.BYTE    =    0x38;        // 送受信イネーブル,ﾏﾙﾁﾌﾟﾛｾｯｻｲﾝﾀﾗﾌﾟﾄｲﾈｰﾌﾞﾙ
}

/************************************************************************
    End of Program.
************************************************************************/
```

株式会社東洋リンクス

※ご質問はメール，または FAX で…
ユーザーサポート係（月～金 10:00～17:00，土日祝は除く）
〒102-0093　東京都千代田区平河町 1-2-2　朝日ビル
TEL:03-3234-0559
FAX:03-3234-0549
E-mail:toyolinx@va.u-netsurf.jp
URL:http://www2.u-netsurf.ne.jp/~toyolinx

20050411