お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

H8/300L SLP シリーズ

方向検出 (DirFind)

要旨

本アプリケーションノートでは，H8/38024 SLP MCU をアナログホール効果センサにインタフェースする方法を説明します。このセンサは正弦波・余弦波の 1 対の波形を出力します。出力した正弦波・余弦波の波形は，MCU で方向情報に変換されて，英数字のドットマトリクスディスプレイに表示されます。この簡易システムは, コンパスや方位検出器として使用できます。H8/38024 SLP MCU を使用する利点は, この MCU が, A/D 変換器，シリアルコミュニケーションインタフェース (SCI)，タイマ，などの豊富な周辺機能を内蔵していることです。

動作確認デバイス

H8/38024 SLP

目次

1. システムの概要

図 1 に，本アプリケーションノートで説明するコンパスのブロック図を示します。このシステムは以下の部品を含みます。

H8/38024 SLP MCU
Dinsmore 社製のアナログホール効果センサ (1525)
英数字のドットマトリクスディスプレイ
RS-232C インタフェース

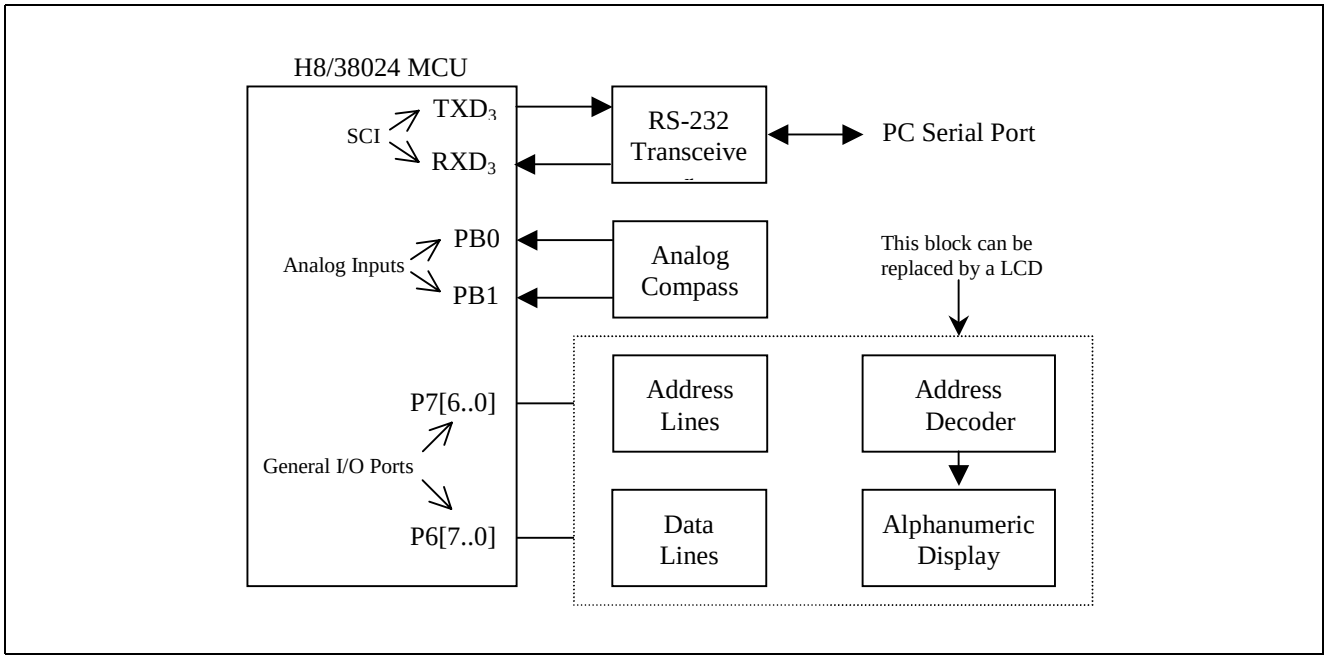


図 1 システムブロック図

本アプリケーションノートでは，MCU の以下の機能を使用します。

アナログセンサからの出力を同価なデジタル値に変換する。10 ビットの分解能を持つ 8 チャネルのアナログ入力チャネルのうち 2 チャネルをアナログセンサとのインタフェースに使用する。
角度を決定する。
方向情報を表示する。
方向情報を RS-232C シリアルポートに送信する (デバッグする)。

2. ハードウェアの適用

使用する周辺機能を以下に示します。

表 1 使用する周辺機能

周辺機能	機能
ポート 6[7..0]	**データバス**
ポート 7[6..0]	**アドレスバス**
P77	WRITE/READ_N
SCI (TXD_{32}，RXD_{32})	**ホスト** PC **と通信する**
P43	**コンパス用の測定ピン，測定が完了すると** Low **になる**
PB0，PB1	**アナログセンサの出力** 1 **と** 2

2.1 電源

本アプリケーションの凡例には，以下の 3 つの独立した電源が必要です。

6V の入力電圧
74HCT138，レベルシフタ，英数字の表示用 5V 電源
MCU，RS-232C トランシーバ，レベルシフタ用 3.3V 電源

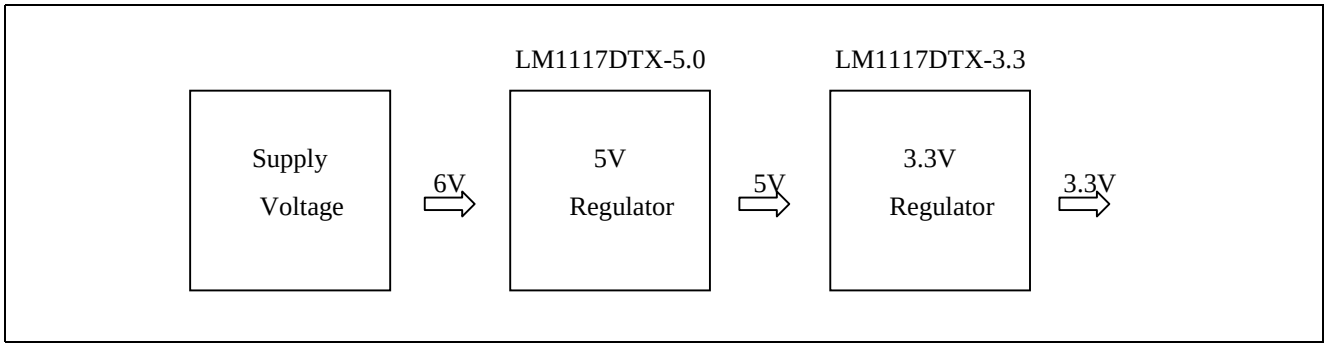


図 2 電源

2.2 アドレスバスとデータバス

図 3 で示すように，H8/38024 SLP MCU をメモリマップされた外付けデバイス，メモリ，周辺機器にアクセスするには，汎用 I/O ポートを使用して，制御信号の WRITE/READ_N と，アドレスバスおよびデータバスを，それぞれ別に作成する必要があります。MCU は 3.3V の電源で動作しますが，他のいくつかのデバイスは 5V で動作します。レベルシフタで，MCU を 5V のデバイスにインタフェースします。

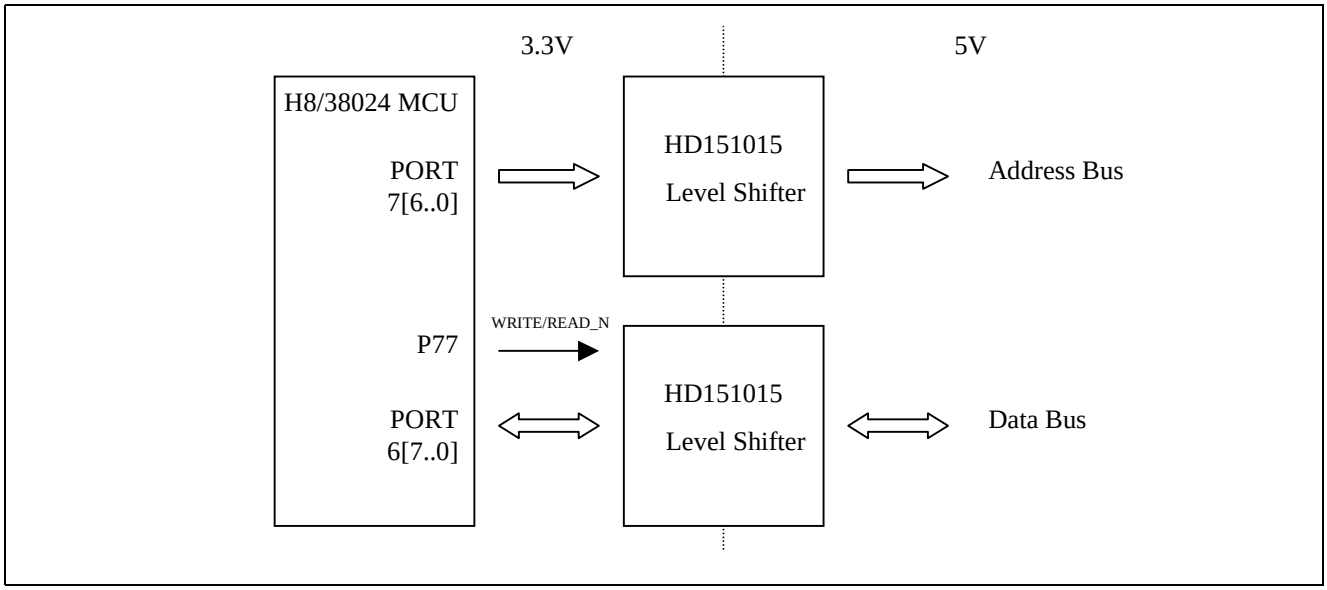


図 3　アドレスバスとデータバス

2.3　アドレスデコーダ

3-8 ラインデコーダ 74HCT138 を使用し，メモリマッピングされた英数字のディスプレイ (DLR1414) を選択します。表 2 にアドレスマップ一覧を示します。

表 2　アドレスマップ

アドレス (16 進数)	デバイス
C0	ディスプレイ (最初の桁)
C1	ディスプレイ (2 番目の桁)
C2	ディスプレイ (3 番目の桁)
C3	ディスプレイ (4 番目の桁)

2.4　RS-232C トランシーバ

シリアルコミュニケーションインタフェース (SCI) の TXD_{32} ピンと RXD_{32} ピンを Sipex 社製の RS-232C トランシーバ SP3232 に接続します。これにより MCU はホスト PC と通信できます。

2.5　アナログホール効果センサ

Dinsmore 社製のアナログホール効果センサ 1525 は 5.00V に厳密に安定化された DC 入力を必要とし，レシオメトリックな DC 出力に対応します。消費電流は，およそ 18~19mA です。出力は，図 4 に示すように，正弦・余弦の曲線が非常に類似しています。中心値が 2.4V で 2.5V～2.9V に上昇し，およそ 2.1V で最小になります。MCU は AVCC = 3.3V で動作するので，これらの出力は AD 変換器の入力に直接接続することができます。

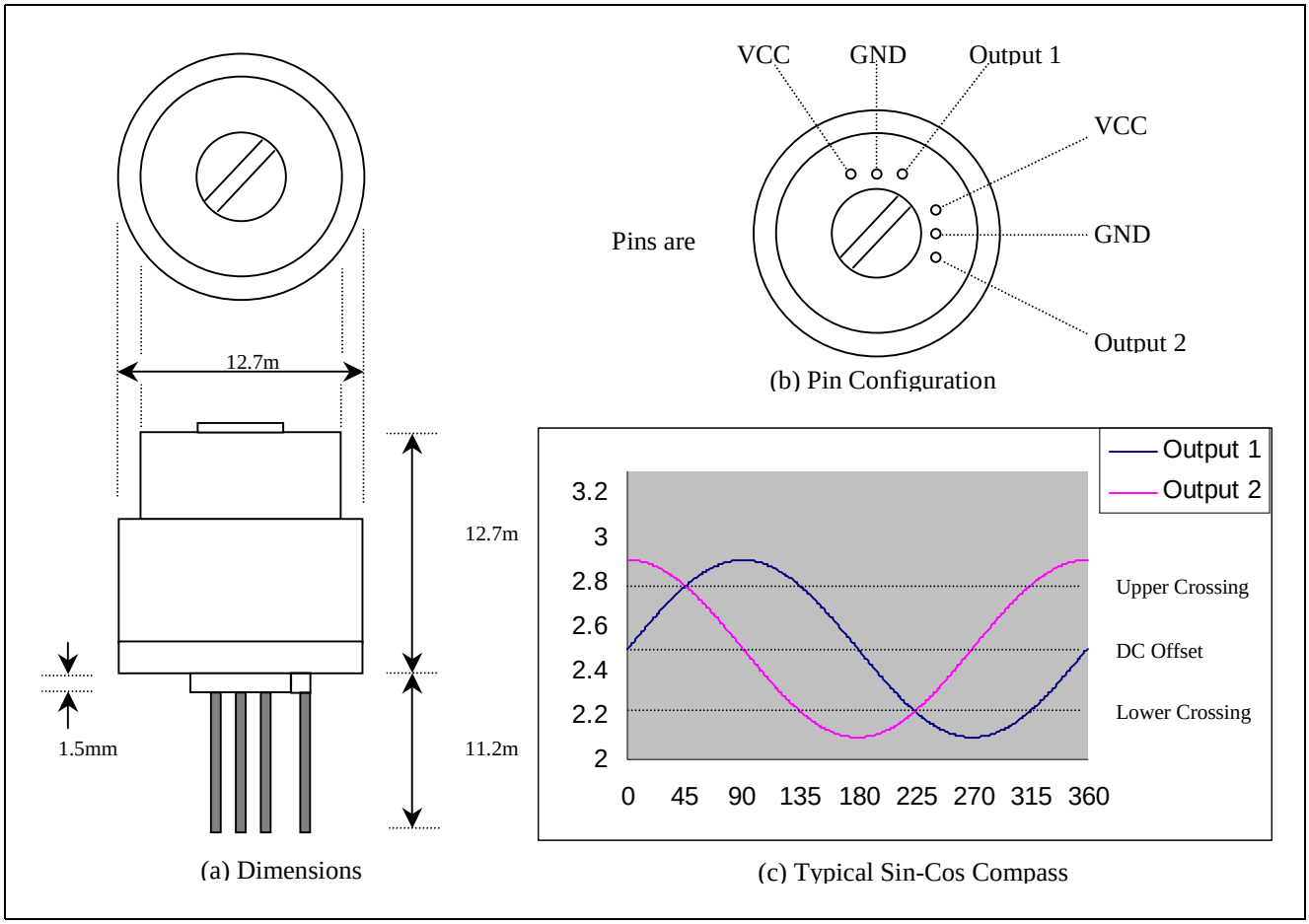


図 4　Dinsmore 社製のアナログホール効果センサ (1525)

出力 1 = 振幅·sin (ωt) + DC オフセット
出力 2 = 振幅·cos (ωt) + DC オフセット

正弦波と余弦波が同じ数値になる交差ポイントの上下の曲線を比較してください。それぞれ正弦波と余弦波の曲線は，上下の交差ポイント間で比較的直線の急勾配です。たとえば，正弦波の直線部分では，0°から 45°, 135°から 225°，315°から 360°の間で直線です。余弦波では，45°から 135°，225°から 315°の間で直線です。

範囲		比較的直線である曲線部分
始め	終わり	
0°	45°	**正弦波**
45°	135°	**余弦波**
135°	225°	**正弦波**
225°	315°	**余弦波**
315°	360°	**正弦波**

特定の値のアークサインとコサインの曲線は，2 つ以上の角度に対応するので，センサからの 2 番目の出力は，角度を識別することに使用します。アークサイン関数を使用する前に，正弦波と余弦波の曲線を正規化しなければいけません。つまり，DC オフセットを曲線から除かなければいけません。

$$\text{DC Offset} = \left(\frac{\text{Maximum} + \text{Minimum}}{2}\right)$$

$$\text{Amplitude} = \left(\frac{\text{Maximum} - \text{Minimum}}{2}\right)$$

$$\text{Angle}(\text{radians}) = \sin^{-1}\left(\frac{\text{Output 1} - \text{DC Offset}}{\text{Amplitude}}\right) \text{ or } \cos^{-1}\left(\frac{\text{Output 2} - \text{DC Offset}}{\text{Amplitude}}\right)$$

$$\text{Angle}(\text{degrees}) = \left[\text{Angle}(\text{radians}) \times \frac{180}{\pi}\right]$$

2.6 A/D 変換器とのインタフェースの方法

“Application Note on Detailed Usage of ADC” で推奨している A/D 変換器とのインタフェースの手引きの一部を本アプリケーションノートで採用しています。

ノイズを抑える簡単な方法は，図 5 に示すように，低速で低電流のアナログ機能と，高速の中電流のデジタル機能用に，それぞれ別の電源を用意することです。バイパスコンデンサとフェライトビーズを併用すると，ローパスフィルタのネットワークを形成でき，高周波ノイズを減少することができます。また，変化する電流を抑え，両方の GND が低 AC インピーダンスになります。

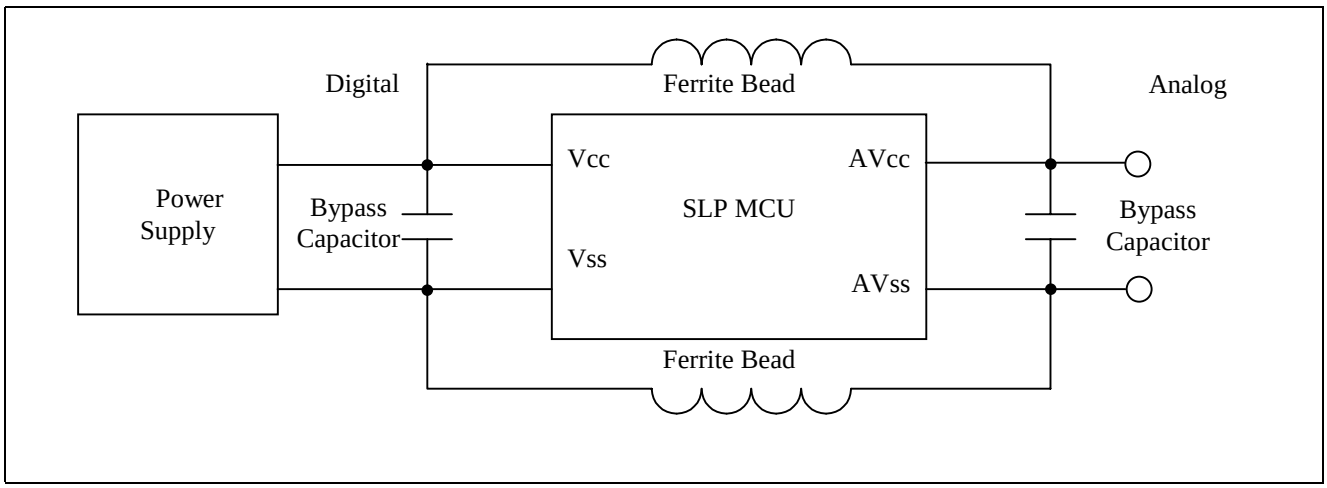


図 5 分割電源

図 6 に示すように，高速のアナログ信号を変換するときは，低インピーダンスバッファを挿入してください。このバッファにより，センサに対して高インピーダンスになり，A/D 変換器に対しては低インピーダンスになります。

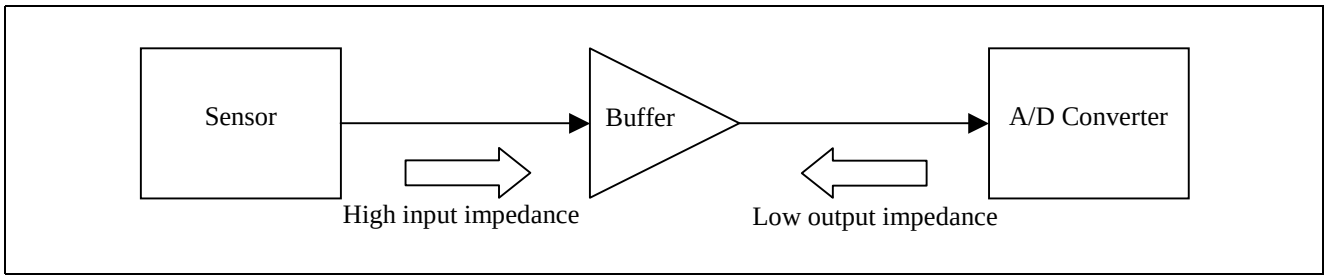


図 6　信号の隔離

すべての未使用アナログ入力ピンは，AVCC にプルアップされます。

3.　ソフトウェアの適用

プログラムコードは，簡単に適用できるように C 言語で書かれています。また，HEW のバージョン 2.2 (リリース 15) 用の無償の H8 Tiny/SLP のツールチェーン (バージョン 5.0.0) を使用し，コンパイルされます。

ソースプログラムコードは以下の機能を持ちます。

I/O ポート，A/D 変換器，シリアルポートの初期化

キャリブレーション

— DC オフセットの平均値と出力波形の振幅を “offset_value”と “amplitude_value”のプログラムにそれぞれ格納します。

— キャリブレーションが必要な場合は，J3 のジャンパ接続ピンである 2 ピンと 3 ピンを取り除きます。新たな最大と最小の出力値を得るために，アナログセンサを最低 1 回転してください。キャリブレーションが終了したら，2 ピンと 3 ピンにまたがるピン J3 ジャンパを取り付けます。これらの値から振幅を計算します。

アナログ入力を角度表示へ変換します。角度表示は英数字でディスプレイに表示され，SCI を介して PC に送信されます。PC のシリアルポートを 38400bps，8 データビット，1 ストップビット，パリティディセーブルを設定してください。ハイパターミナルまたは Tera Term Pro のようなターミナルをエミュレーションするソフトゥエアを使用してください。

main() 関数のフローチャートを図 7 に示します。

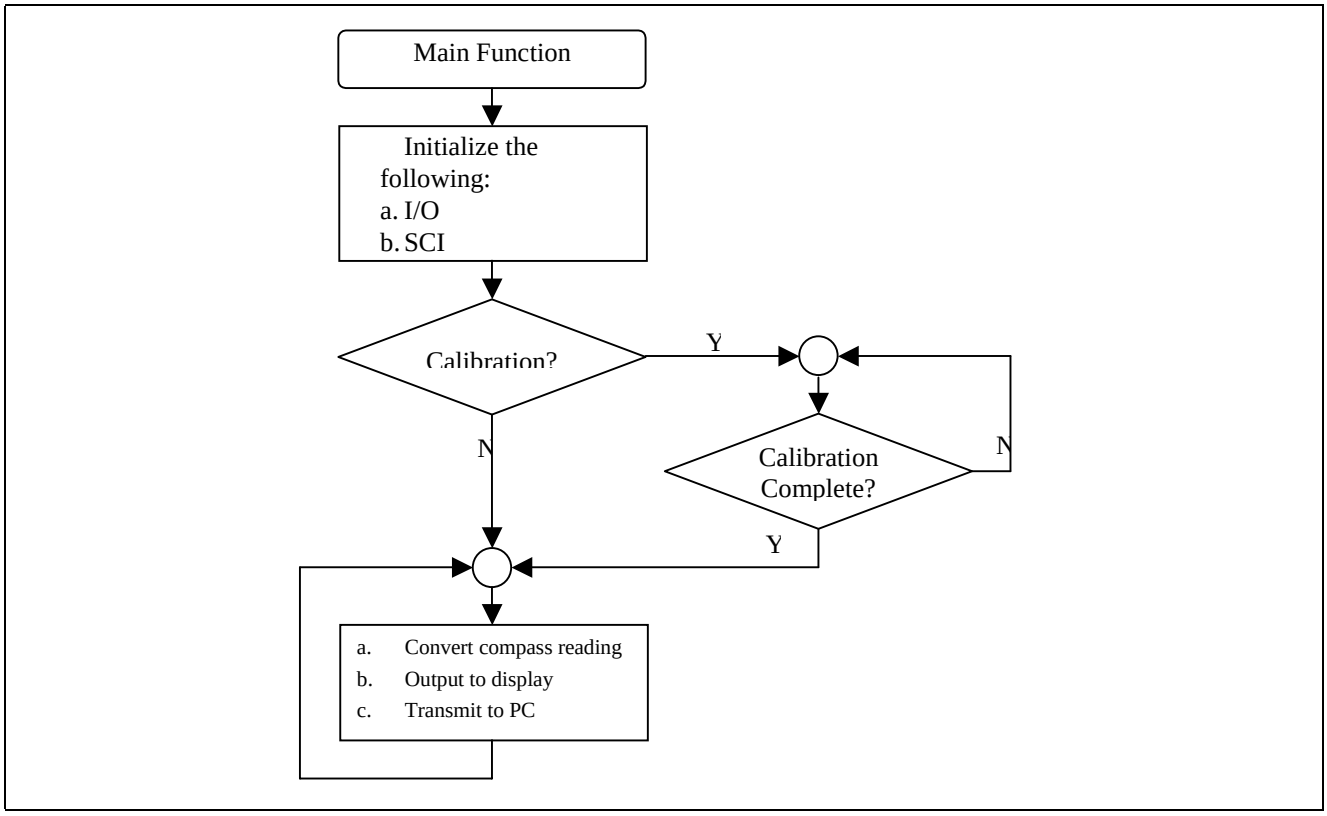


図 7　フローチャート

図 8 に示すように，プロトタイプボードは低価格エミュレータ ALE300L-H8/3800 と接続できます。

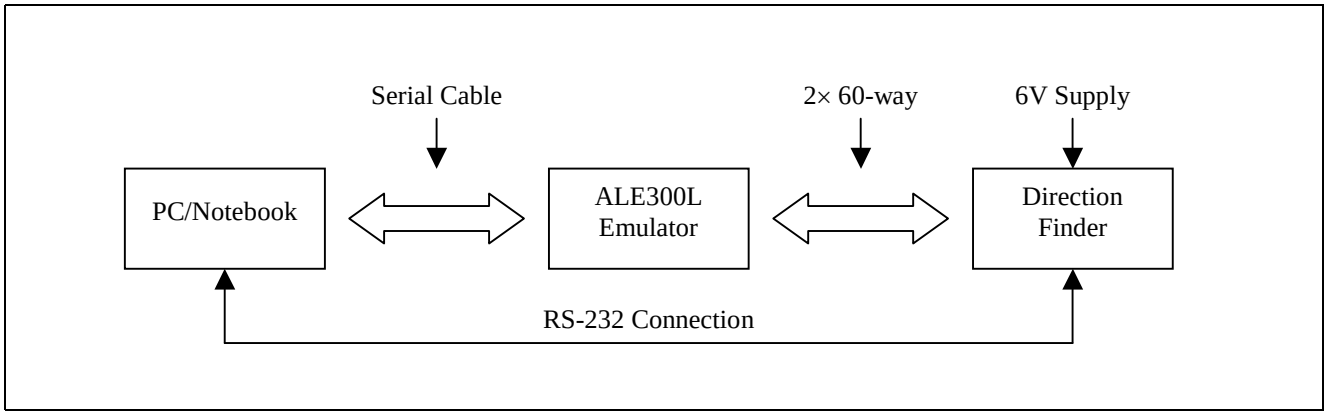


図 8　システムのセットアップ

3.1 ソースプログラムコード

ソースプログラムコードはおもに DirFind.c.にあり，以下が詳細です。

main 関数を含みます。
I/O ポート，シリアルコミュニケーションインタフェース (SCI)，A/D 変換器を初期化します。
コンパスのキャリブレーションを行います。
コンパス読み取り結果を変換します。

```
/*********************************************************************/
/*                                                                   */
/*  FILE       :DirFind.c                                            */
/*  DATE       :Mon, Feb 09, 2004                                     */
/*  DESCRIPTION :Main Program                                         */
/*  CPU TYPE   :H8/3802                                               */
/*                                                                   */
/*  This file is generated by Renesas Project Generator (Ver.2.1).    */
/*                                                                   */
/*********************************************************************/

#ifdef __cplusplus
extern "C" {
#endif
void abort(void);
#ifdef __cplusplus
}
#endif

#include "iodefine.h"
#include <machine.h>
#include <math.h>

//-----------------------------------------------------------------------
//Constant Declarations
//-----------------------------------------------------------------------

//Constants for Address Decoder
#define first_digit           0xC0
#define second_digit          0xC1
#define third_digit           0xC2
#define fourth_digit          0xC3

#define   de_select     0x3F

//ASCII Constants
#define   blank               0x20

//Others
#define   address_bus         P_IO.PDR7.BYTE          //Address Bus
#define   data_bus            P_IO.PDR6.BYTE          //Data Bus
#define   calibration_input   P_IO.PDR4.BIT.P43       //Calibration Input

#define pi                    3.141592

//Average sensor values computed
//If calibration is not performed, then these values will be used
#define   offset_value  786.25
```

```
#define   amplitude_value      132.8



//---------------------------------------------------------------------
//Function Prototypes
//---------------------------------------------------------------------
void init_sci(void);
void char_put(char);
void PutStr(char *);

void init_port(void);

void display_char(unsigned char, unsigned char);

void display_word(unsigned int);

void serial_transmit(unsigned int);

void init_adc(unsigned char);
void start_adc(unsigned char);

unsigned int ADC_value (void);
void find_value(void);

void delay(void);

//---------------------------------------------------------------------
//global variables
//---------------------------------------------------------------------

unsigned int max, min;

//---------------------------------------------------------------------

void main(void)
{
   //-------------------------------------------------------

   unsigned int f1, f2, f3, direction, directionS, directionC;
   unsigned int sin_value, cos_value;

   float           output1, output2;
   float           degreeS, degreeC;
   float        dc_offset, amplitude;

   //-------------------------------------------------------

   init_port();

   init_sci();

   //-------------------------------------------------------

   dc_offset = offset_value;
   amplitude = amplitude_value;
```

```
display_char(fourth_digit, blank);            //Blank display
display_char(third_digit, blank);       //Blank display
display_char(second_digit, blank);            //Blank display
display_char(first_digit, blank);       //Blank display

//-----------------------------------------------------
//This loop is required for calibration.
//Need to rotate the compass first for at least 1 full revolution
//Calibration is considered "done" when the jumper is shorted

if (calibration_input)
{
   display_char(fourth_digit, 0x43);   //Display 'C'

   min = 1024;
   max = 0;

   do
   {
      find_value();
   } while (calibration_input == 1);

   dc_offset = (max + min)/2;
   amplitude = (max - dc_offset);

   delay();
   display_char(fourth_digit, blank);  //End of calibration : blank
}

//-----------------------------------------------------

while(1)
{
   init_adc(0);                  //Initialize ADC

   start_adc(1);
   while (P_AD.ADSR.BYTE & 0x80);      //If ADSR = 1, A/D conversion in progress
   sin_value = ADC_value();

   init_adc(1);                        //Initialize ADC

   start_adc(1);

   while (P_AD.ADSR.BYTE & 0x80);//If ADSR = 1, A/D conv in progress
   cos_value = ADC_value();

   //Calculation of Degree
   //range for asin() is -90 to 90
   //range for acos() is 0 to 180
   output1 = (sin_value - dc_offset) / amplitude;
   degreeS = asin(output1) * 180.0 / pi;

   output2 = (cos_value - dc_offset) / amplitude;
   degreeC = acos(output2) * 180.0 / pi;

   //-----------------------------------------------------
```

```
if (cos_value >= dc_offset)         //1st or 4th quadrant
{
   if (degreeS > 0)                 //1st quadrant : 0 to 90
   {
      directionS = degreeS;
   }
   else
   {
      directionS = 360 + degreeS;   //4th quadrant : 270 to 360
   }
}
else
{
   directionS = 180 - degreeS;      //2nd & 3rd quadrants : 90 to 270
}

//-----------------------------------------------------------

if (sin_value >= dc_offset)         //1st & 2nd quadrants : 0 to 180
{
   directionC = degreeC;
}
else
{
   if (degreeC < 0)
   {
      directionC = 180 - degreeC;   //3rd quadrant : 180 to 270
   }
   else
   {
      directionC = 360 - degreeC;   //4th quadrant : 270 to 360
   }
}

//-----------------------------------------------------------

if (((directionC > 45) && (directionC < 135)) ||
   ((directionC > 225) && (directionC < 315)))
   direction = directionC;
else if (((directionS >= 135) && (directionS <= 225)) ||
         ((directionS >= 315) || (directionS <= 45)))
   direction = directionS;

//-----------------------------------------------------------

f1 = floor(direction / 100);
f2 = floor((direction - (100 * f1)) / 10);
f3 = floor(direction - (f1 * 100) - (f2 * 10));
display_char(third_digit, f1 + 0x30);
char_put(f1 + 0x30);
display_char(second_digit, f2 + 0x30);
char_put(f2 + 0x30);
display_char(first_digit, f3 + 0x30);
char_put(f3 + 0x30);
PutStr("¥r¥n");
delay();
```

```
    }
}

//---------------------------------------------------------------------

/*
    init_port() : Set up the I/O ports

    a. Port 6[7..0] -> Data[7..0]
    b. Port 7[7..0] -> Address[7..0]
       Note that Port 7_7 also functions as the WRITE/READ_N signal
*/

void init_port(void)
{
    P_LCD.LPCR.BYTE = 0x00;                 //SEG[32..1] as I/O Port

    P_IO.PCR6.BYTE  = 0xFF;                 //Set Port 6 as all output

    P_IO.PCR7.BYTE  = 0xFF;                 //Set Port 7 as all output

    data_bus     = 0xFF;              //Set Data Bus to all '1'

    address_bus     = 0xFF;                 //Set Address Bus to all '1'
}

//---------------------------------------------------------------------

/*
    display_char()

    a. Port 6[7..0] -> Data[7..0]
    b. Port 7[7..0] -> Address[7..0]
       Note that Port 7_7 also functions as the WRITE/READ_N signal
*/

void display_char(unsigned char digit_position, unsigned char digit_info)
{
    P_IO.PCR6.BYTE  = 0xFF;                 //Set Port 6[7..0] as output

    address_bus &= de_select;

    data_bus = digit_info;                  //Data

    address_bus = digit_position;           //Address
    address_bus &= de_select;
}

//---------------------------------------------------------------------

/*
    display_word()
*/

void display_word(unsigned int display_data)
{
    unsigned char   position, digit_info, digit_position;
```

```
    P_IO.PCR6.BYTE  = 0xFF;                 //Set Port 6[7..0] as output

    for (position = 4 ; position != 0 ; position--)
    {
       switch (position)
       {
          case 1:
             digit_position = first_digit;
             digit_info = (unsigned char)(display_data & 0x000F);
             break;

          case 2:
             digit_position = second_digit;
             digit_info = (unsigned char)((display_data & 0x00F0) >> 4);
             break;

          case 3:
             digit_position = third_digit;
             digit_info = (unsigned char)((display_data & 0x0F00) >> 8);
             break;

          default:
             digit_position = fourth_digit;
             digit_info = (unsigned char)((display_data & 0xF000) >> 12);
             break;
       }

       if ((digit_info >= 0) && (digit_info <= 9))
          digit_info += 0x30;
       else
       {
          if ((digit_info >= 0xA) && (digit_info <= 0xF))
          {
             digit_info -= 0xA;
             digit_info += 0x41;
          }
       }

       address_bus &= de_select;

       data_bus = digit_info;               //Data

       address_bus = digit_position; //Address

       address_bus &= de_select;
    }
}

//---------------------------------------------------------------------

/*
   serial_transmit()
*/

void serial_transmit(unsigned int display_data)
{
```

```
    unsigned char   position, digit_info;

    for (position = 4 ; position != 0 ; position--)
    {
       switch (position)
       {
          case 1:
             digit_info = (unsigned char)(display_data & 0x000F);
             break;

          case 2:
             digit_info = (unsigned char)((display_data & 0x00F0) >> 4);
             break;

          case 3:
             digit_info = (unsigned char)((display_data & 0x0F00) >> 8);
             break;

          default:
             digit_info = (unsigned char)((display_data & 0xF000) >> 12);
             break;
       }

       if ((digit_info >= 0) && (digit_info <= 9))
          digit_info += 0x30;
       else
       {
          if ((digit_info >= 0xA) && (digit_info <= 0xF))
          {
             digit_info -= 0xA;
             digit_info += 0x41;
          }
       }

       char_put(digit_info);
    }
}
//---------------------------------------------------------------------

/*
    init_sci() : Sets up the Serial Communication Interface for debugging
                 purposes.
*/

void init_sci(void)
{
    //SCR3 : |TIE|RIE|TE|RE|MPIE|TEIE|CKE1|CKE0|
    //TIE : Transmit interrupt enable
    //RIE : Receive interrupt enable
    //TE  : Transmit enable
    //RE  : Receive enable
    //MPIE : Multiprocessor interrupt enable
    //TEIE : Transmit end interrupt enable
    //CKE1 : Clock enable 1
    //CKE0 : Clock enable 0

    //CKE1 = CKE0 = 0
```

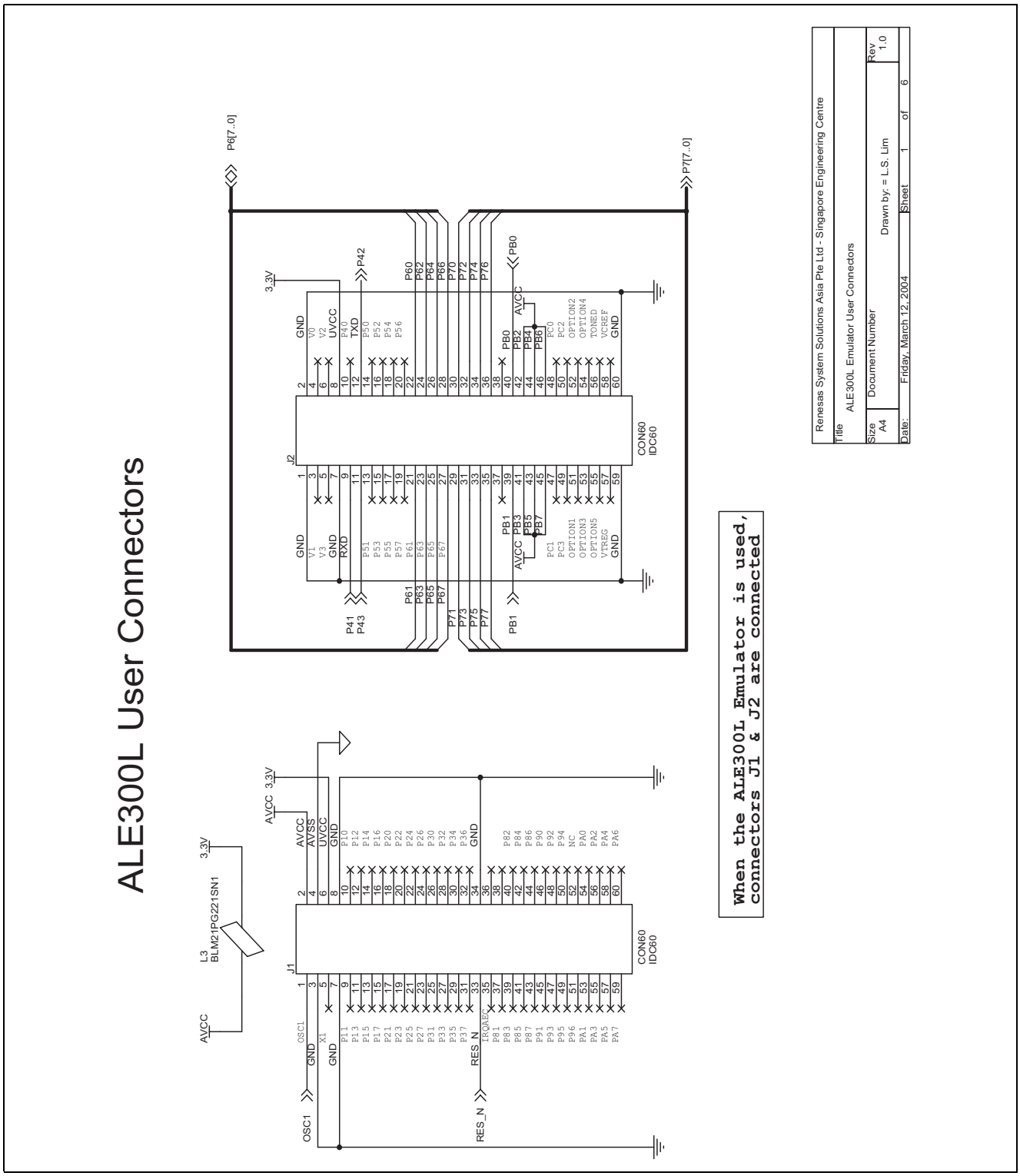
```
    //asynchronous mode, internal clock source, SCK32 functions as I/O port
    P_SCI3.SCR3.BYTE &= 0x00;  //clear TE & RE

    //SMR : |COM|CHR|PE|PM|STOP|MP|CKS1|CKS0| : |0|0|0|0|0|0|0|0|
    //COM : Communication Mode  : 0 : asynchronous mode
    //CHR : Character Length    : 0 : character length = 8 bits
    //PE  : Parity Enable       : 0 : parity bit addition and checking disabled
    //PM  : Parity Mode         : 0 : even parity (no effect since parity is already disabled)
    //STOP: Stop Bit Length     : 0 : 1 stop bit
    //MP  : Multiprocessor Mode : 0 : multiprocessor communication function disabled
    //|CKS1|CKS0| : Clock Select: |0|0| : clock source for baud rate generator = clk
    P_SCI3.SMR.BYTE = 0x00;

    //For clk = 10MHz, bit rate = 2400 bps, n = 0, N = 64
    //P_SCI3.BRR = 64;

    //For clk = 10MHz, bit rate = 38400 bps, n = 0, N = 3
    P_SCI3.BRR = 3;

    //minimum of 1-bit delay = 417ns
    nop();
    nop();
    nop();

    //SPCR : |---|---|SPC32|---|SCINV3|SCINV2|---|---| : |1|1|1|0|0|0|0|0|
    //SPC32 = 1 : P42 functions as TXD32 output pin
    //need to set TE bit in SCR3 after setting this bit to 1
    //SCINV3 = 0 : TXD32 output data is not inverted
    //SCINV2= 0 : RXD32 input data is not inverted
    //Bits 7 and 6 are reserved and always read as 1
    //Bits 4, 1 and 0 are reserved and only 0 can be written to these bits
    P_SCI3.SPCR.BYTE = 0xE0;

    P_SCI3.SCR3.BYTE |= 0x30;  //Set TE & RE
}

//---------------------------------------------------------------------

/*
    char_put() : Transmits a character to the PC for debugging purposes.
*/

void char_put(char OutputChar)           //Serial Port
{
    //SSR  : |TDRE|RDRF|OER|FER|PER|TEND|MPBR|MPBT|
    //TDRE : transmit data register empty
    //RDRF : receive data register full
    //OER  : overrun error
    //FER  : framing error
    //PER  : parity error
    //TEND : transmit end
    //MPBR : Multiprocessor bit receive
    //MPBT : Multiprocessor bit transfer

    while ((P_SCI3.SSR.BIT.TDRE) == 0);          //Wait for TDRE = 1

    P_SCI3.TDR = OutputChar;
```

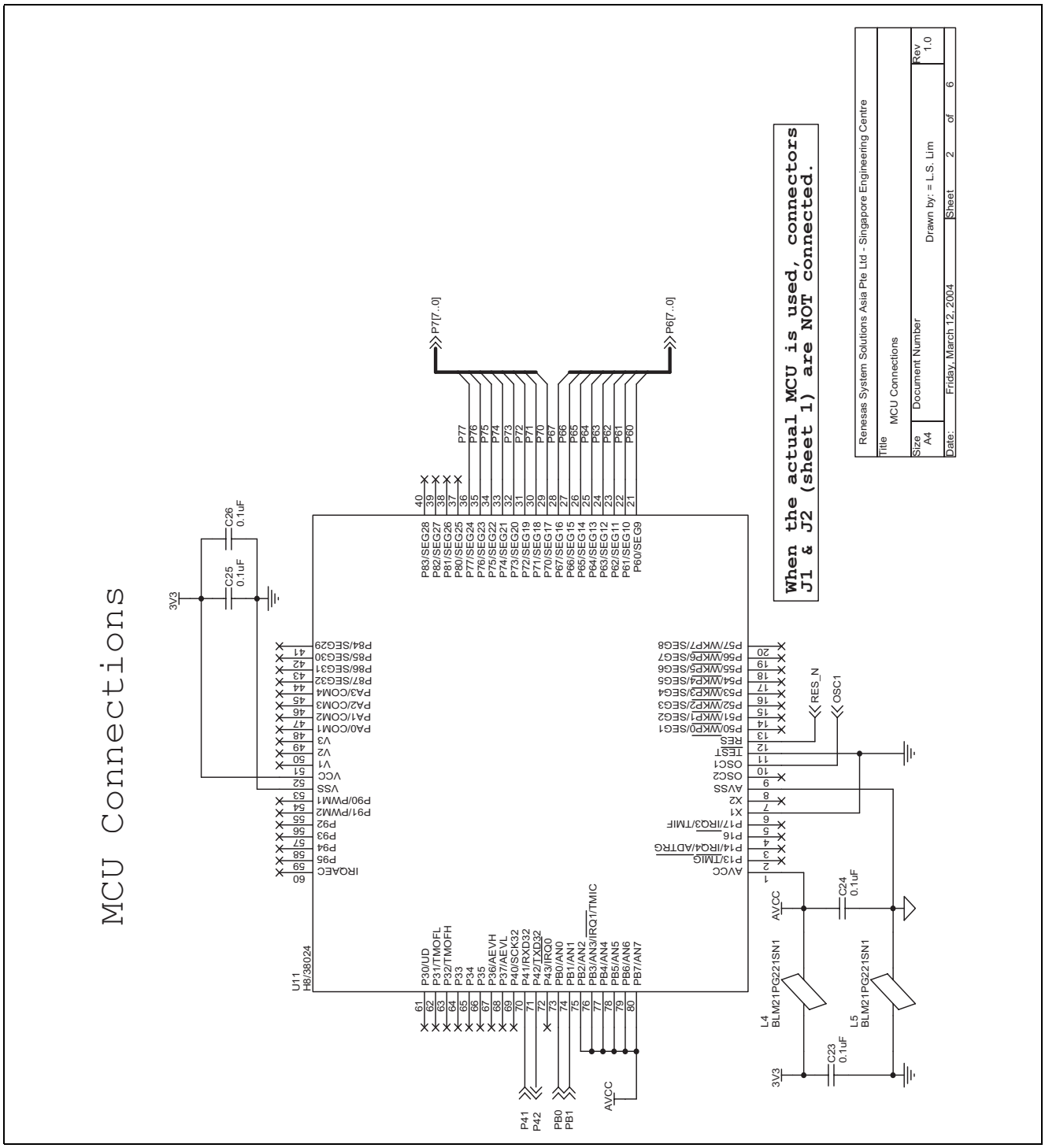
```
    while ((P_SCI3.SSR.BIT.TEND) == 0);          //Wait for TEND = 1

    P_SCI3.SSR.BIT.TEND = 0;
}

//---------------------------------------------------------------------

/*
    PutStr() : Transmits a string of characters to the PC for debugging purposes.
*/

void PutStr(char *str)
{
    while (*str != 0)
    {
        char_put(*str++);
    }
}

//---------------------------------------------------------------------

/*
    init_adc()
*/
void init_adc(unsigned char Input_CH)
{
    //CKS = 0 -> A/D Conversion period = 62/phi
    P_AD.AMR.BIT.CKS= 0;

    //TRGE = 0 -> Disable start of A/D conversion by external trigger
    P_AD.AMR.BIT.TRGE = 0;

    //Input_CH = 0-7 => Select ADC input channel
    P_AD.AMR.BIT.CH = (Input_CH + 4);

    //ADC Module standby mode is cleared
    P_SYSCR.CKSTPR1.BIT.ADCKSTP = 1;
}

//---------------------------------------------------------------------

/*
    start_adc(start)

*/

void start_adc(unsigned char start)
{
    if (start == 1)
        P_AD.ADSR.BYTE |= 0x80;       //Set ADSF : start A/D conversion
    else
        P_AD.ADSR.BYTE &= 0x7F;       //Set ADSF : stop A/D conversion
}

//---------------------------------------------------------------------
```

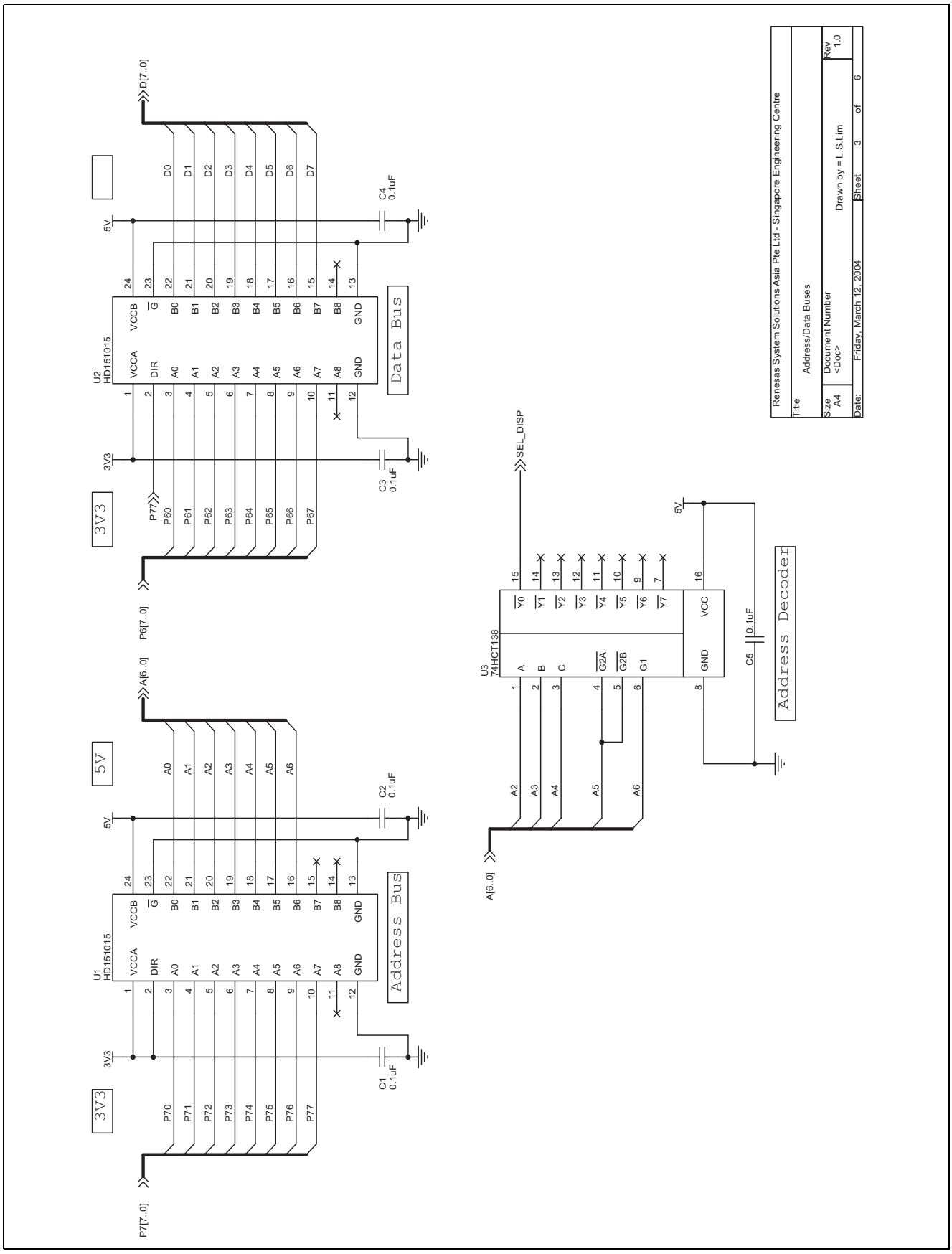
```
/*
   ADC_value()
*/

unsigned int ADC_value(void)
{
   unsigned int adrrL, adrrH;
   unsigned int valueL,valueH;
   unsigned int D_value;

   adrrH = P_AD.ADRR >> 8;                //Capture the ADC value from AN#
   adrrL = P_AD.ADRR << 8;
   valueH = adrrH << 2;
   valueL = adrrL >> 14;
   D_value = valueL | valueH;
   return (D_value);
}

//---------------------------------------------------------------------

/*
   find_value() - find the maximum and minimum values of output1
*/

void find_value(void)
{
   unsigned int temp_output1;

   init_adc(0);               //Initialize ADC channel 0

   start_adc(1);                    //Start ADC

   while (P_AD.ADSR.BYTE & 0x80);   //If ADSR = 1, A/D conversion in progress

   temp_output1 = ADC_value();

   if (temp_output1 > max)
   {
      max = temp_output1;
   }

   if (ADC_value() < min)
   {
      min = temp_output1;
   }
}

//---------------------------------------------------------------------

/*
   delay() - software delay routine
*/

void delay(void)
{
   unsigned int delay_loop;
```

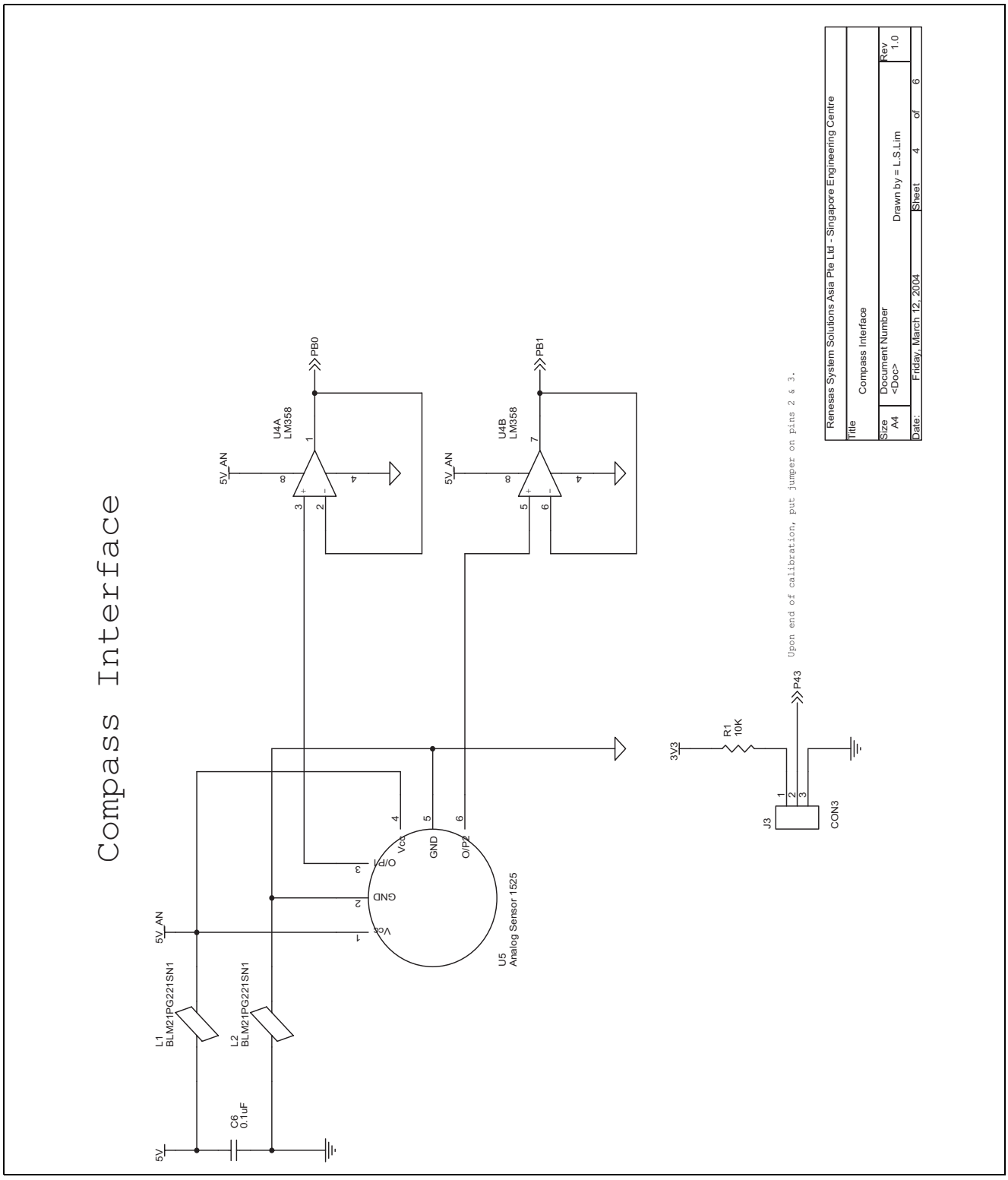
```
    for (delay_loop = 0 ; delay_loop < 30000 ; delay_loop++);
}

//-----------------------------------------------------------------------

void abort(void)
{

}
//-----------------------------------------------------------------------
```

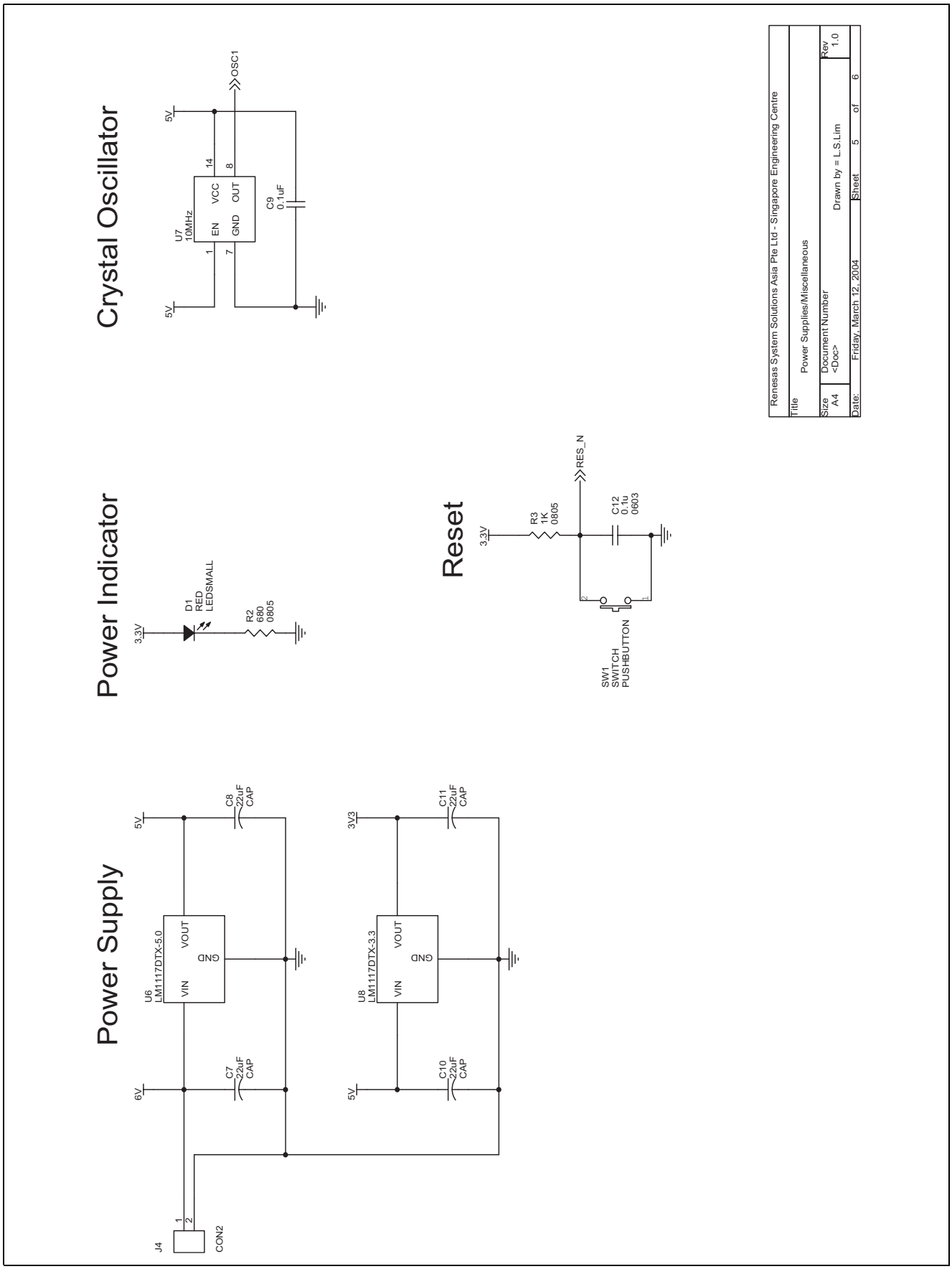
4. ハードウェアの概略図

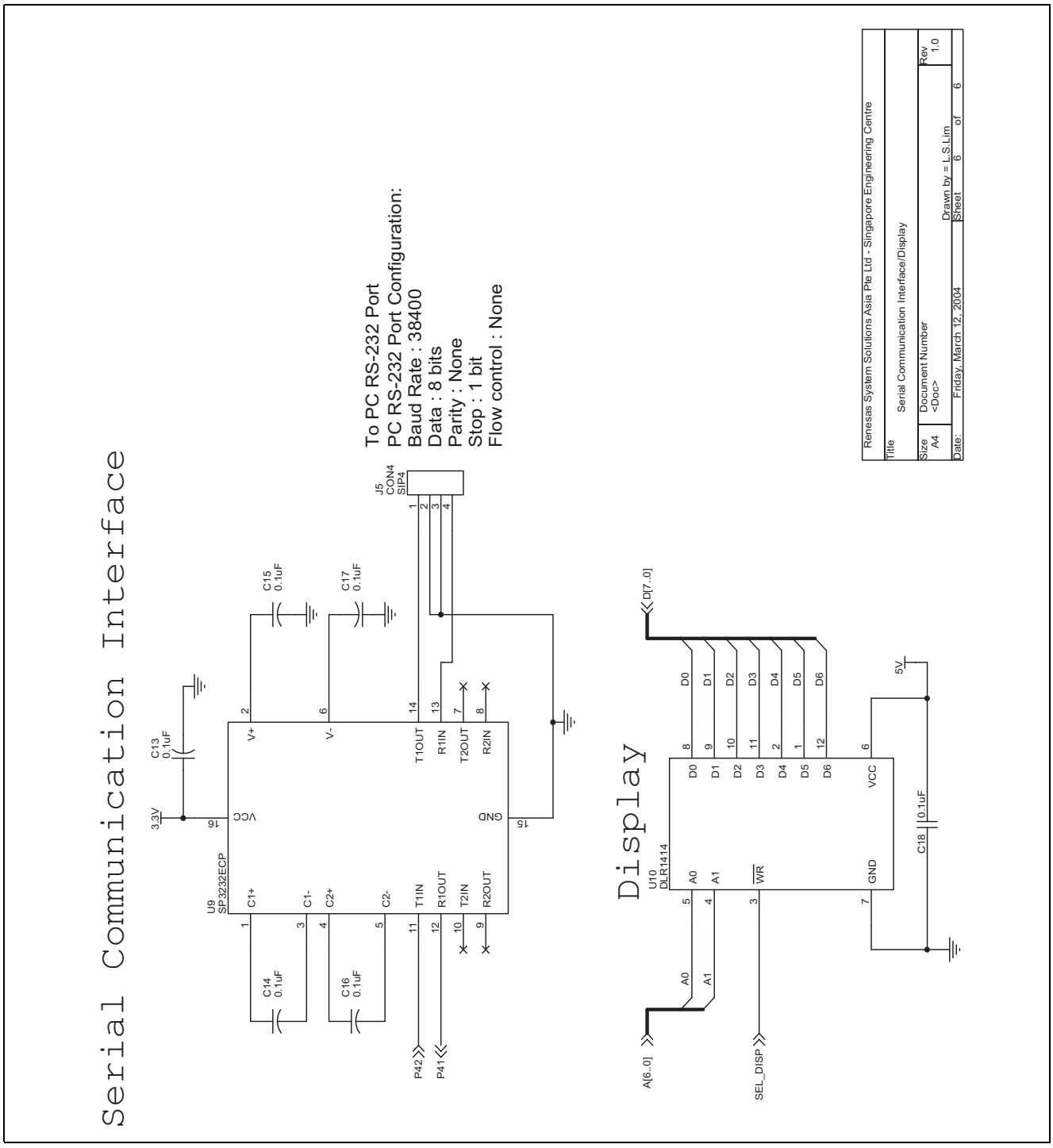



MCU Connections
When the actual MCU is used, connectors J1 & J2 (sheet 1) are NOT connected.
Renesas System Solutions Asia Pte Ltd - Singapore Engineering Centre
Title MCU Connections
Size A4
Document Number
Rev 1.0
Drawn by: = L.S. Lim
Date: Friday, March 12, 2004
Sheet 2 of 6
U11 H8/38024
P7[7..0]
P6[7..0]
C26 0.1uF
C25 0.1uF
C24 0.1uF
C23 0.1uF
L4 BLM21PG221SN1
L5 BLM21PG221SN1
3V3
AVCC
RES_N
OSC1


Renesas System Solutions Asia Pte Ltd - Singapore Engineering Centre
Title
Address/Data Buses
Size A4
Document Number <Doc>
Rev 1.0
Drawn by = L.S.Lim
Date: Friday, March 12, 2004
Sheet 3 of 6
U1 HD151015
U2 HD151015
U3 74HCT138
3V3
5V
Address Bus
Data Bus
Address Decoder
SEL_DISP
A[6..0]
P7[7..0]
P6[7..0]
D[7..0]
C1 0.1uF
C2 0.1uF
C3 0.1uF
C4 0.1uF
C5 0.1uF


Compass Interface
Renesas System Solutions Asia Pte Ltd - Singapore Engineering Centre
Title
Compass Interface
Size
A4
Document Number
<Doc>
Rev
1.0
Drawn by = L.S.Lim
Date:
Friday, March 12, 2004
Sheet
4
of
6
Upon end of calibration, put jumper on pins 2 & 3.
PB0
PB1
P43
U4A
LM358
U4B
LM358
5V_AN
3V3
R1
10K
J3
CON3
U5
Analog Sensor 1525
O/P1
GND
O/P2
Vcc
L1
BLM21PG221SN1
L2
BLM21PG221SN1
C6
0.1uF
5V


Crystal Oscillator
OSC1
5V
U7
10MHz
EN
GND
VCC
OUT
14
8
1
7
C9
0.1uF
Power Indicator
3.3V
D1
RED
LEDSMALL
R2
680
0805
Reset
RES_N
3.3V
R3
1K
0805
C12
0.1u
0603
SW1
SWITCH
PUSHBUTTON
Power Supply
5V
6V
3V3
U6
LM1117DTX-5.0
U8
LM1117DTX-3.3
VIN
VOUT
GND
C8
22uF
CAP
C7
22uF
CAP
C11
22uF
CAP
C10
22uF
CAP
J4
CON2
Renesas System Solutions Asia Pte Ltd - Singapore Engineering Centre
Title
Power Supplies/Miscellaneous
Size
A4
Document Number
<Doc>
Drawn by = L.S.Lim
Rev
1.0
Date:
Friday, March 12, 2004
Sheet
5
of
6


Serial Communication Interface
To PC RS-232 Port
PC RS-232 Port Configuration:
Baud Rate : 38400
Data : 8 bits
Parity : None
Stop : 1 bit
Flow control : None
Display
Renesas System Solutions Asia Pte Ltd - Singapore Engineering Centre
Serial Communication Interface/Display
Document Number
<Doc>
Rev 1.0
Drawn by = L.S.Lim
Friday, March 12, 2004
Sheet 6 of 6
U9 SP3232ECP
U10 DLR1414
J5 CON4 SIP4
C13 0.1uF
C14 0.1uF
C15 0.1uF
C16 0.1uF
C17 0.1uF
C18 0.1uF
3.3V
5V
P42
P41
A[6..0]
D[7..0]
SEL_DISP

5. 参考文献

1. Dinsmore Sensing Systems General Information, Dinsmore Sensors (http://www.dinsmoresensors.com/).
2. LM1117/LM1117I 800mA Low-Dropout Linear Regulator, 2002, National Semiconductor Corporation.
3. LM158/LM258/LM358/LM2904 Low Power Dual Operational Amplifiers, 2002, National Semiconductor Corporation.
4. HD151015 9-bit Level Shifter/Transceiver with 3 State Outputs, 3rd Edition, June 1993, Renesas Technology Corp. (ref. no. REJ03D0300-0400, http://renesas.com)
5. Detailed Usage of ADC, 2003, Renesas Technology Corp. (Application Note ref. no: RJS06B0026-0100 http://renesas.com,)
6. DLR1414 4-character 5 × 7 Dot Matrix Alphanumeric Intelligent Display with Memory/Decode/Driver, Infineon Technologies.

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2004.08.06	—	初版発行

安全設計に関するお願い

1. 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

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ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ルネサス テクノロジ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。
2. 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、ルネサス テクノロジは責任を負いません。
3. 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、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 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、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 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ルネサス テクノロジ、ルネサス販売または特約店までご照会ください。