

マップメータ(電子物差し)

概要

地図を見るときに目的地までの距離を知りたくなるときがあります。今回は、地図の上をなぞることによって、その距離を測定し、結果を表示させてみました。この値を尺度換算すると実際の距離がわかります。

動作原理

地図の上をなぞる方法は、ロータリエンコードを使うことにしました。古いマウスが手元にありましたので、その内部で使用されているロータリエンコーダが使えないかと考えました。

取り外して、調べてみると、次のような仕様なので何とか使えそうです。

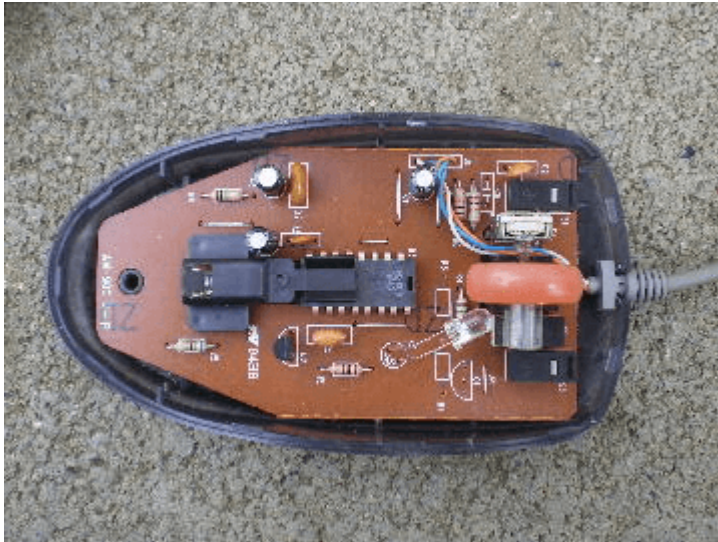
- 1回転で48の変化(A相、B相の信号) 回路図上の図Bを参照してください。
- ローラの直径が22mm(1回転で約69mm)

この仕様より、距離を求める式は、次のようになります。

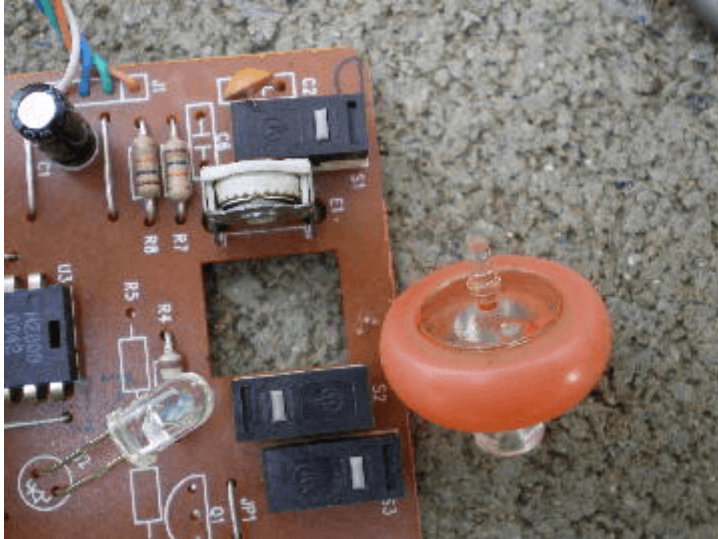
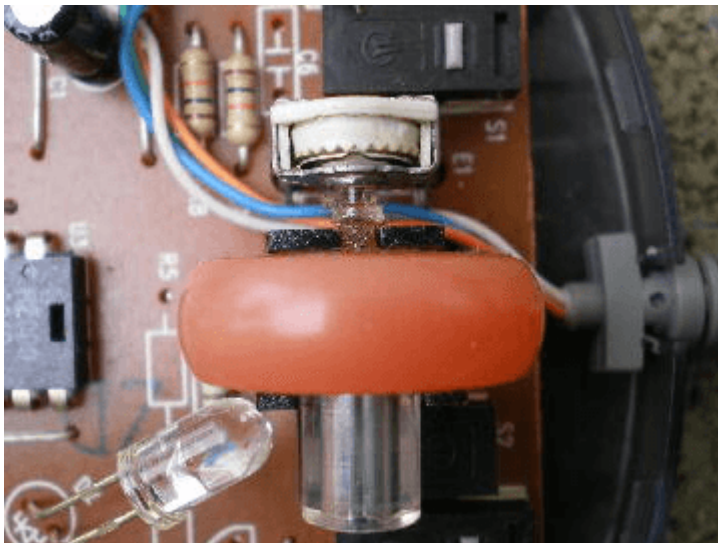
- $\text{距離} = (22\text{mm} \div 48) \times \text{ロータリエンコーダの変化数} \times 3.14159$



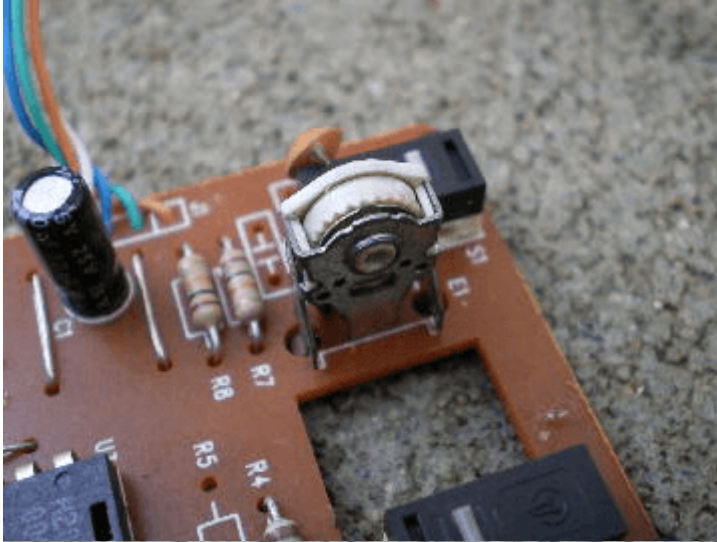
マウスのふたを外してみました。



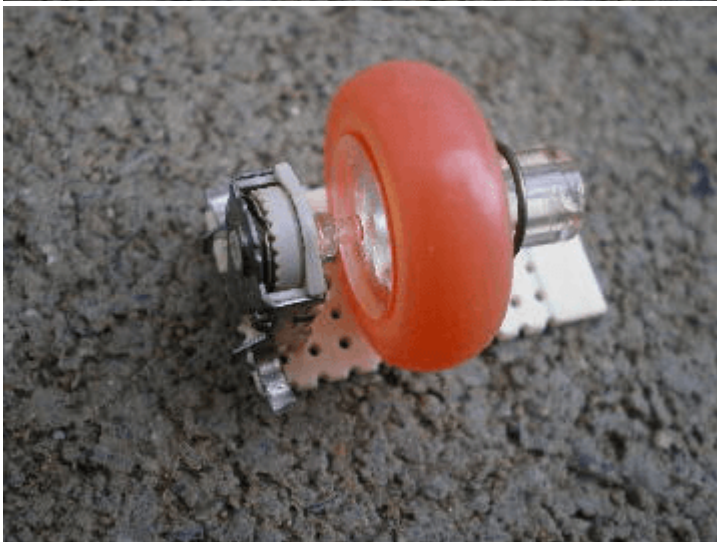
小さなロータリエンコーダが使われていました。

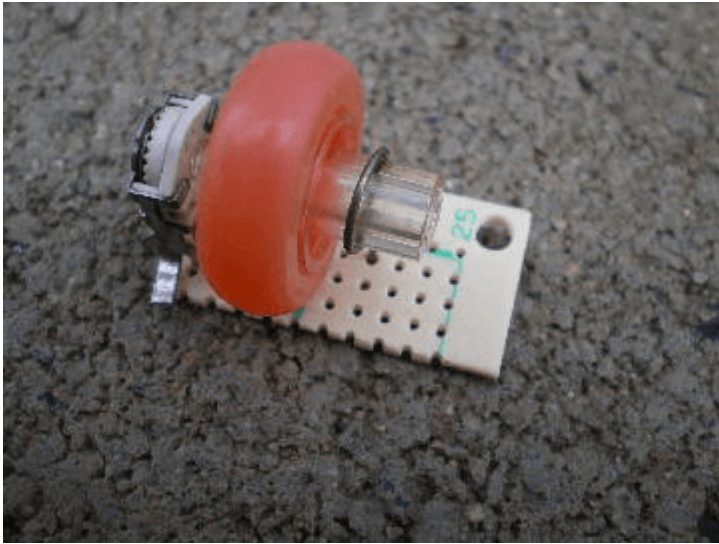


基板から外しました。

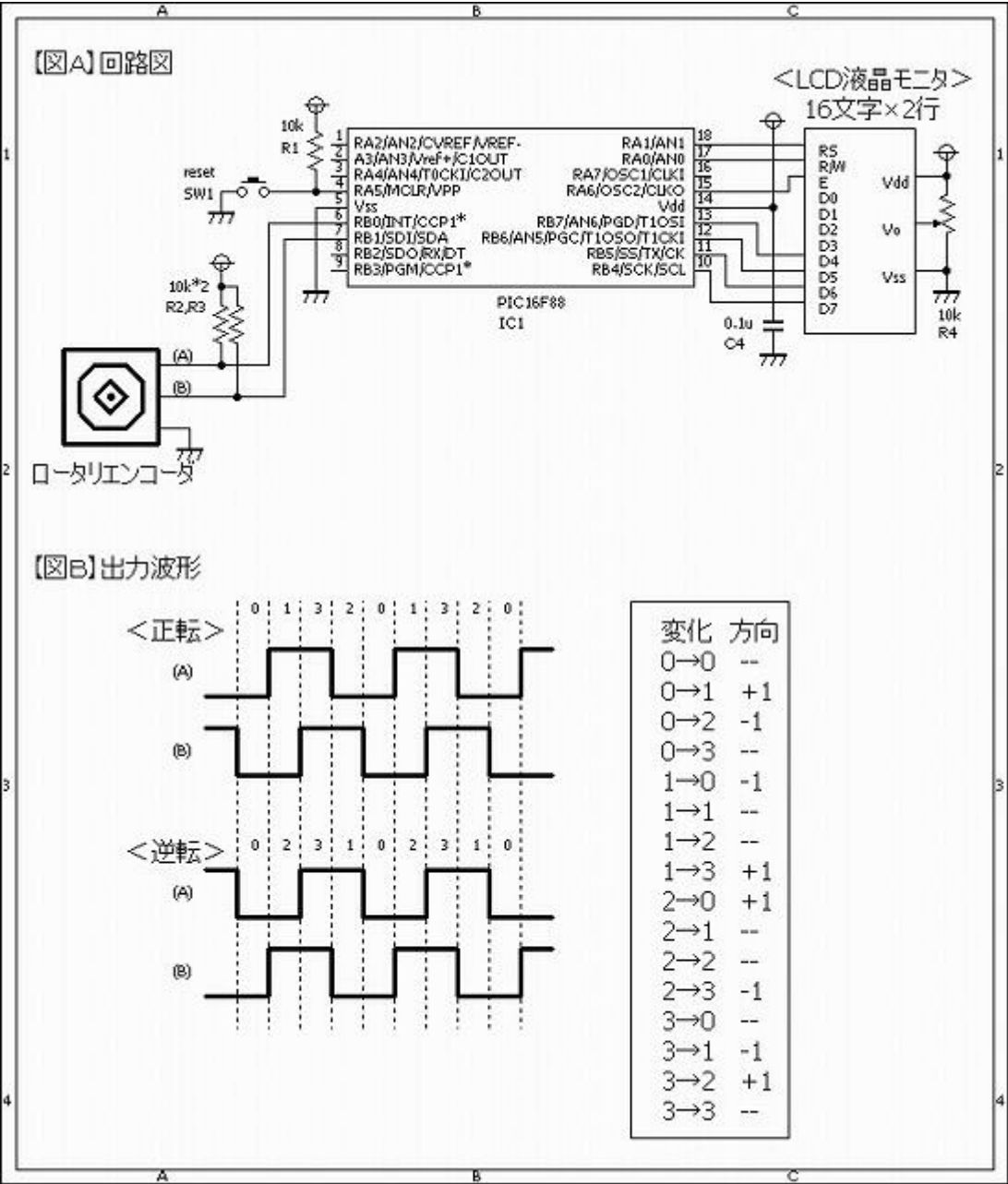


小さな基板に組み直しました。





回路図



ソースコード

ロータリエンコーダの回転の有無および正転逆転の向きを知る方法 1msec毎の割り込み処理の中で、図Bの表で示す変化を検出し判断します。

[MapMeter.c](#)

```
//*****  
*  
/*  
  <マップメータ（電子物差し）>  
  ■ 機能概要  
    ・ロータリエンコーダ  
      分解能（12ステップ/1回転）  
      直径□□□□  
  ■ 表示内容
```

```
距離□□□単位)
□■□□□□□の設定
  □LVP_OFF
□□□MCLR_OFF
□□□WDT_OFF
□□□INTRC_IO
*/
//*****
*

#define      SW0      PORTA.F5
#define      SW1      PORTB.F0
#define      SW2      PORTB.F1

#define      LED1      PORTB.F2
#define      LED2      PORTB.F3

//*****
*

static unsigned   char    RE_dat, RE_tmp;
static int        RE_val;

void  interrupt()
{
    //□□□□□□の割り込み処理
    if (PIR1.CCP1IF == 1) {
        PIR1.CCP1IF = 0;
        //
        RE_tmp = PORTB & 0x03;
        //前回の値との変化でロータリエンコンダの回転の有無を判断
        switch (RE_dat) {
            case 0:
                switch (RE_tmp) {
                    case 0:
                        break;
                    case 1:
                        RE_val++;
                        break;
                    case 2:
                        RE_val--;
                        break;
                    case 3:
                        break;
                }
                break;
            case 1:
                switch (RE_tmp) {
                    case 0:
                        RE_val--;

```

```
        break;
    case 1:
        break;
    case 2:
        break;
    case 3:
        RE_val++;
        break;
    }
    break;
case 2:
    switch (RE_tmp) {
    case 0:
        RE_val++;
        break;
    case 1:
        break;
    case 2:
        break;
    case 3:
        RE_val--;
        break;
    }
    break;
case 3:
    switch (RE_tmp) {
    case 0:
        break;
    case 1:
        RE_val--;
        break;
    case 2:
        RE_val++;
        break;
    case 3:
        break;
    }
    break;
}
RE_dat = RE_tmp;
}
}

//*****
*

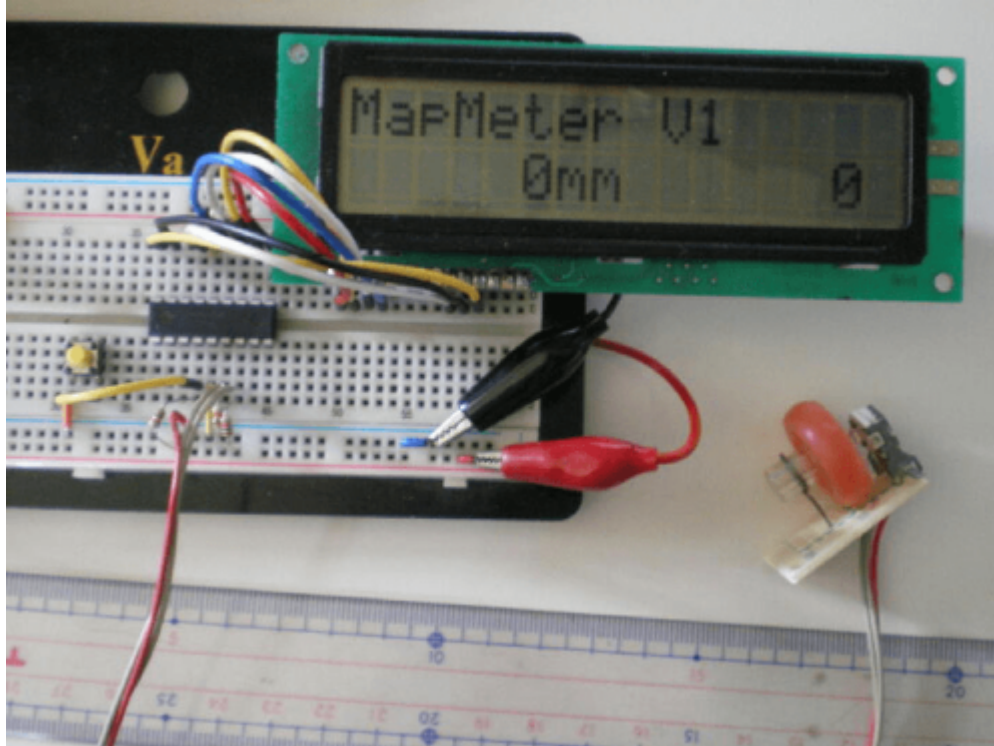
void main()
{
    static unsigned char buf[20];
    static int length;
    //
```

```
OSCCON = 0b01110000;    // クロックは8Mhz
CMCON = 0b00000111;     // コンパレータは使用しない。
// アナログの設定
ANSEL = 0b00000000;     // □□□変換は使用しない。
// ポートの設定
TRISA = 0b10111100;
TRISB = 0b00000011;
OPTION_REG.F7 = 0;       // PORTBをプルアップする。
// CCPの設定
PIE1.CCP1IE = 1;
PIR1.CCP1IF = 0;
CCP1CON = 0b00001011;
CCPR1L = 0xD0;           // 0.001sec...(1÷8000000)*4*2000
CCPR1H = 0x07;
// TIMER1の設定
PIE1.TMR1IE = 0;
PIR1.TMR1IF = 0;
TMR1L = 0;
TMR1H = 0;
T1CON.T1CKPS0 = 0;
T1CON.T1CKPS1 = 0;
T1CON.TMR1ON = 1;
// 変数の初期化
RE_dat = 0;
RE_val = 0;
// □□□□液晶モニタ)の初期化
Lcd_Custom_Config(&PORTB,4,5,6,7,&PORTA,1,0,6);
Lcd_Custom_Cmd(LCD_CURSOR_OFF);
Lcd_Custom_Out(1, 1, "MapMeter V1");
// 割り込み(全体)の設定
INTCON.PEIE = 1;
INTCON.GIE = 1;
//
while(1) {
    // 長さの換算
    length = (22.0 / 48.0) * (float)RE_val *
3.1415926535897932384626433832795;
    // 長さの表示
    IntToStr(length, buf);
    Lcd_Custom_Out(2, 1, buf);
    Lcd_Custom_Out(2, 7, "mm");
    // カウント値の表示
    IntToStr(RE_val, buf);
    Lcd_Custom_Out(2, 10, buf);
    // スリープ□□□□□□□□
    Delay_ms(100);
    // ゼロクリア
    if (SW0 == 0) {
        RE_val = 0;
        Lcd_Custom_Out(2, 1, " ");
    }
}
```

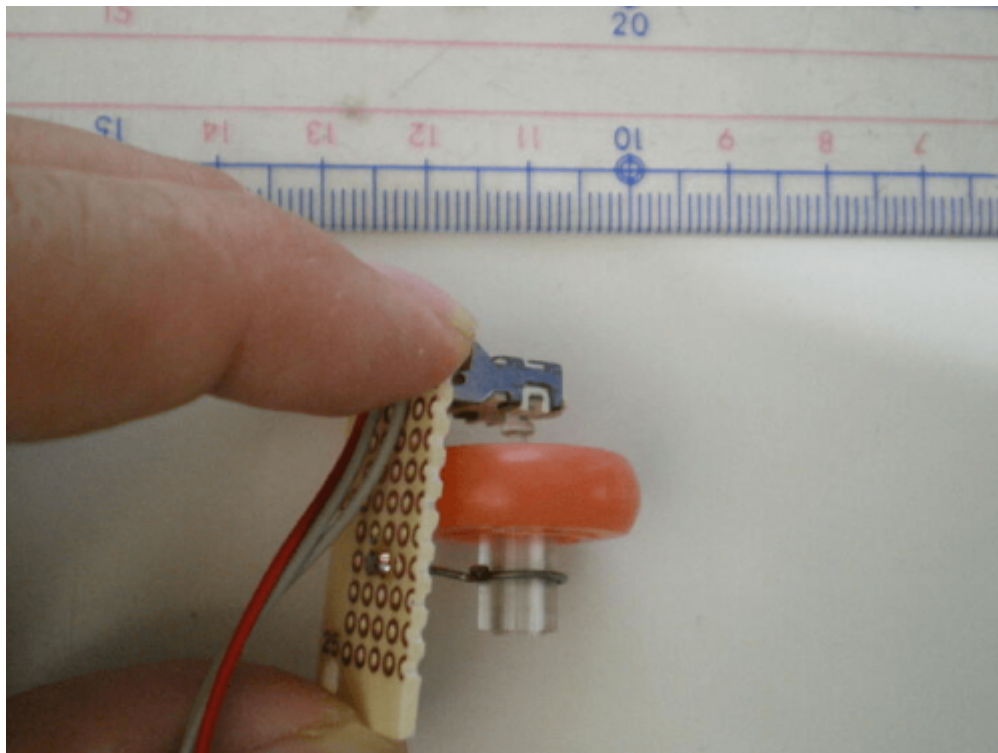
```
}  
}  
}  
  
// ****  
*
```

動作確認

いつものブレッドボードで確認してみました。左側の黄色のスイッチは、ゼロクリア用です。



ロータリエンコーダを取り付けた基板を手で持って転がします。ここはもう少し工夫したいところですね。



30cmの物差しを測ってみ

ました。まあまあの精度です。



如何ですか? ロータリエン

コードも工夫次第でいろいろな使い方が出来ますね! 😊 }

From:
<http://www.deepsky.jp/wiki/> - うごくといいな

Permanent link:
<http://www.deepsky.jp/wiki/doku.php?id=elechobby:picdic:pic16f88:84&rev=1588161500>

Last update: 2025/10/17 14:28

