ステッピングモータ制御ユニット(I2C対応)

概要

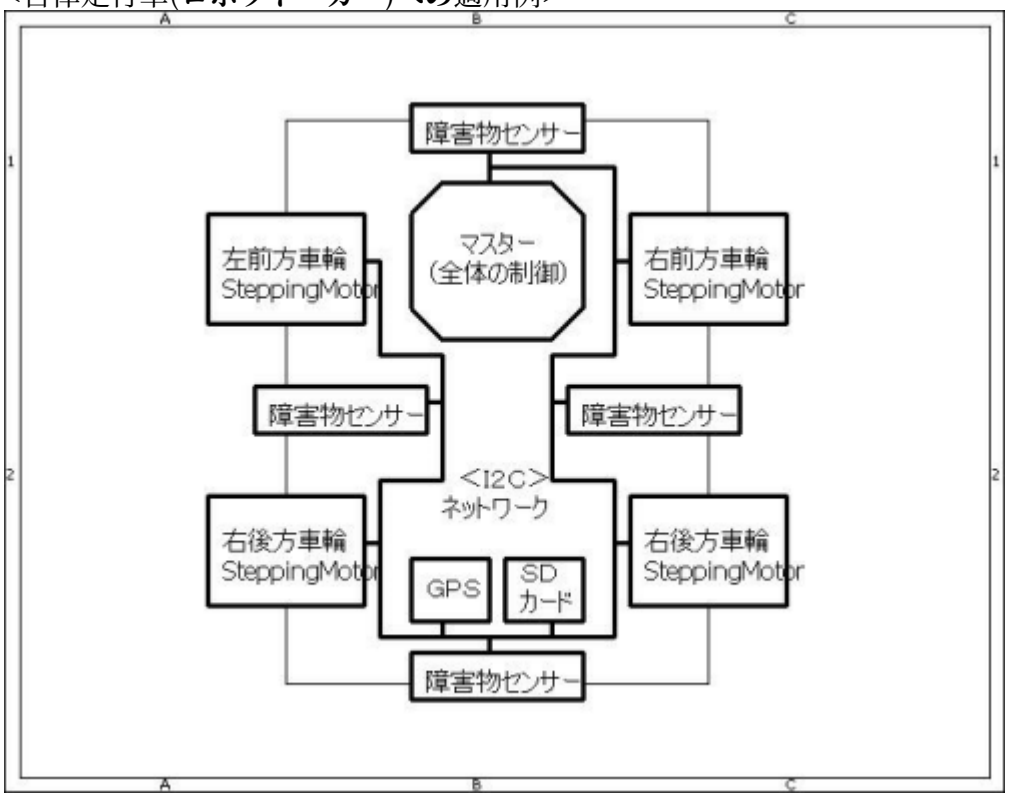
ロボットやCNC(Computerized Numerical Control)の製作にも興味があります。 その時には、複数のステッピングモータを制御することが求められます。 そこで、最大8台まで接続可能な、I2C対応のステッピングモータ制御ユニットを製作しました。

今回使用した、ステッピングモータは、以前に秋月電子通商で購入した、多摩川精機製の「TS3103N124」を使用しましたが、相当のものであれば特に問題はありません。

<TS3103N124の規格>

- 2相ユニポーラ型
- ステップ数200(1.8度)
- 駆動電圧(12~24V)
- コイル電流140mA
- コイル抵抗86Ω/相

<自律走行車(ロボット・カー)への適用例>



動作原理

I2Cのスレーブ機能としての、基本的な構造は、前回製作した「LCDモニタ」(I2C対応)と同じです。 ステッピングモータの制御については、以前に製作した、ステッピングモータ制御を参照して下さい。

<メモリ構造> ステッピングモータの制御(方向、回転数、速度等)をするために、5バイトのメモリを使用します。 「0x00」MODE(0:停止、1:運転) 「0x01」DIR(0:正転、1:逆転) 「0x02」STEP(0~65535回転数の上位バイト) 「0x03」STEP(0~65535回転数の下位バイト) 「0x04」SPEED(0~255」単位は、msec)

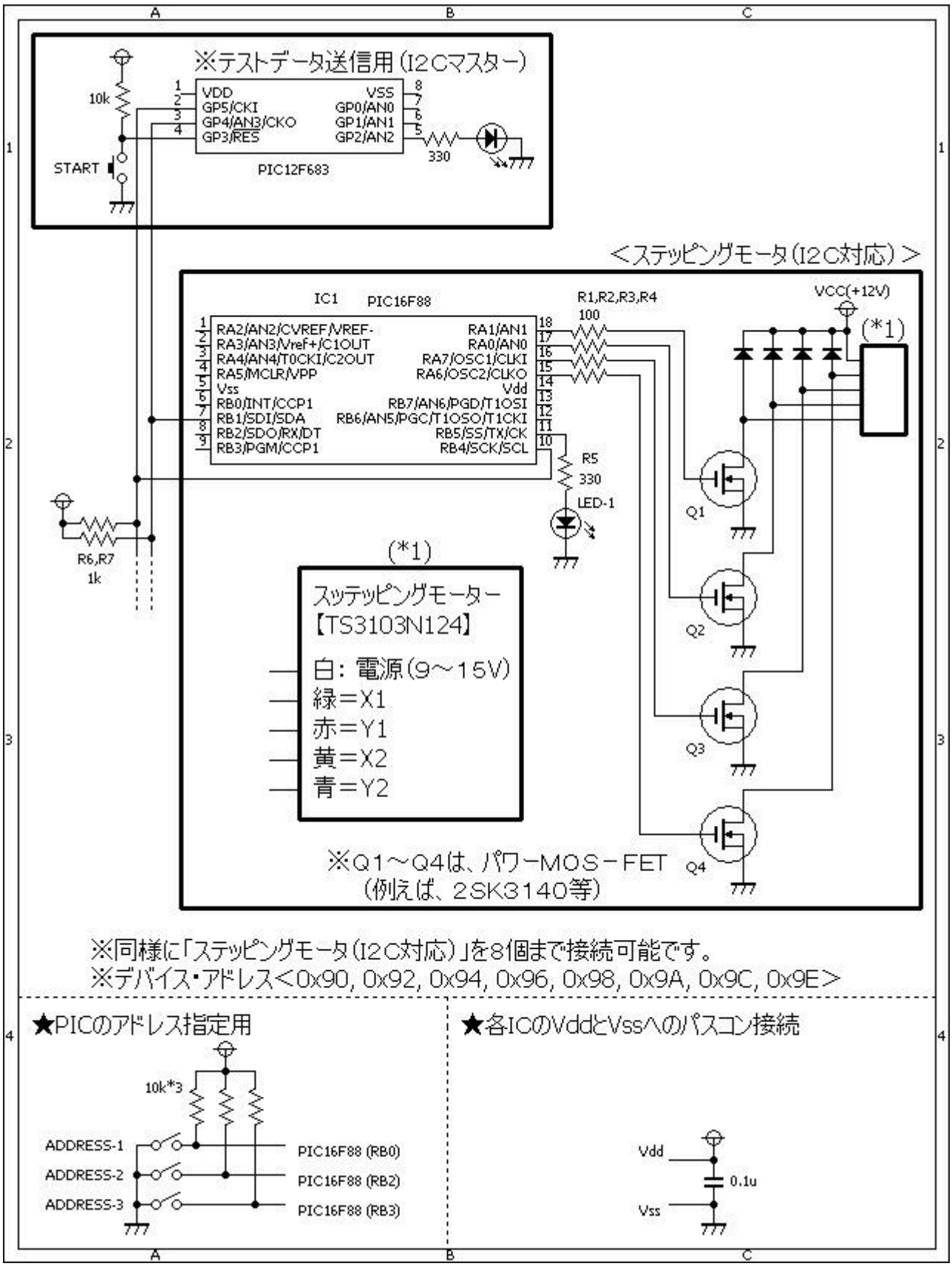
<処理の流れ>

1. I2Cのストップ処理が完了するまで待機する。(マスターからの指示を待つ)
2. MODEが運転であれば、DIR、STEP、SPEEDを読み込みセットする。
3. ステッピングモータを指示通りに動作させる。
4. 動作完了後に、MODEを停止にする。
5. 1.へ戻る。

<マスター側からの制御方法>

- ステッピングモータを動作させる。(運転)
    1. Soft_I2C_Start();
    2. Soft_I2C_Write(0x9E);//アドレスセット
    3. Soft_I2C_Write(0x00);//メモリアドレスセット
    4. Soft_I2C_Write(1);//運転
    5. Soft_I2C_Write(0);//正転
    6. Soft_I2C_Write(0);
    7. Soft_I2C_Write(200);//200ステップ
    8. Soft_I2C_Write(100);//速度100msec/step
    9. Soft_I2C_Stop();
- ステッピングモータの停止を確認する。
    1. Soft_I2C_Start();
    2. Soft_I2C_Write(0x9E);
    3. Soft_I2C_Write(0x00);
    4. Soft_I2C_Start();
    5. Soft_I2C_Write(0x9F);
    6. dat = Soft_I2C_Read(NO_ACK);
    7. Soft_I2C_Stop();
    8. datの値が、0x00であれば停止している。それ以外なら動作中なので1.へ戻る。

回路図



ソースコード

SteppingMotor_i2c.c

```
//*********************************************************************
*
/*
  ＜ステッピングモータ□□□□対応）＞

    ■TS3103N124
     白：電源□□□□□V□
     緑□ X1□PORTA.F6  0 1 1 0
     赤□ Y1□PORTA.F7  0 0 1 1
     黄□ X2□PORTA.F0  1 0 0 1
     青□ Y2□PORTA.F1  1 1 0 0
*/
//*********************************************************************
*

#define       ADDR_BASE    0x90

#define       SW_ADDR1    PORTB.F0
#define       SW_ADDR2    PORTB.F2
#define       SW_ADDR3    PORTB.F3

#define       ON            1
#define       OFF            0

#define       MEMORY_SIZE    5

#define       LED            PORTB.F5

#define       SM_X1        PORTA.F6
#define       SM_Y1        PORTA.F7
#define       SM_X2        PORTA.F0
#define       SM_Y2        PORTA.F1

//*********************************************************************
*

void   i2c_Write(unsigned short dat)
{
    while (SSPSTAT.BF == 1)
        ;
    while (1) {
        SSPCON.WCOL = 0;
        SSPBUF = dat;
        if (SSPCON.WCOL == 1)
            continue;
        //
        SSPCON.CKP = 1;
        return;
    }
}
```

```
//*****************************************************************************
*

static unsigned    short    i2c_memory[MEMORY_SIZE], i2c_pnt, i2c_flg,
i2c_tmp, i2c_start_flg, i2c_stop_flg;
/*
i2c_memory[0]    MODE(0:停止、1:運転)
i2c_memory[1]    DIR(0:右回転、1:左回転)
i2c_memory[2,3]  STEP(0～65535回転数)
i2c_memory[4]    SPEED(0～255msec)
*/
//*****************************************************************************
*

void   i2c_Handler()
{
    if (SSPSTAT.S == 1)         //スタートビットを検出。
        i2c_start_flg = 1;
    if (SSPSTAT.P == 1)         //ストップビットを検出。
        i2c_stop_flg = 1;
    //
    i2c_tmp = SSPSTAT & 0b00101101;
    //
    if (i2c_tmp == 0b00001001) {    //書き込みモード、デバイスアドレス
        i2c_tmp = SSPBUF;
        i2c_flg = 0;
        i2c_pnt = 0;
        return;
    }
    if (i2c_tmp == 0b00101001) {    //書き込みモード、データ
        if (i2c_flg == 0) {
            i2c_pnt = SSPBUF;
            i2c_flg = 1;
            return;
        }
        if (i2c_flg == 1) {
            if (i2c_pnt < MEMORY_SIZE) {
                i2c_memory[i2c_pnt] = SSPBUF;
                i2c_pnt++;
            }
            return;
        }
    }
    if (i2c_tmp == 0b00001100) {    //読み込みモード、デバイスアドレス
        i2c_Write(i2c_memory[i2c_pnt]);
        i2c_pnt++;
        return;
    }
    if (i2c_tmp == 0b00101100) {    //読み込みモード、データ□□□□□
        i2c_Write(i2c_memory[i2c_pnt]);
        i2c_pnt++;
```

```
        return;
    }
    if (i2c_tmp == 0b00101000) {    //読み込みモード、データ□□□□□□□□
        i2c_tmp = SSPBUF;
        SSPCON = 0b00111110;
        return;
    }
}

//*****************************************************************************
*

void   interrupt()
{
    if (PIR1.SSPIF == 1) {
        PIR1.SSPIF = 0;
        //
        LED = ON;
        i2c_Handler();
        LED = OFF;
    }
}

//*****************************************************************************
*

short  pnt;

void   SteppingMotorTurn(short dir, unsigned int step, short speed)
{
    unsigned    int        cnt;
    unsigned    short    cnt2;
    //
    for (cnt = 0; cnt < step; cnt++) {
        if (dir == 0) {        //正転
            if (pnt < 3)
                pnt++;
            else
                pnt = 0;
        } else {            //逆転
            if (pnt > 0)
                pnt--;
            else
                pnt = 3;
        }
        switch (pnt) {
        case 0:
            SM_X2 = ON;
            SM_Y2 = ON;
            SM_X1 = OFF;
            SM_Y1 = OFF;
```

```
            break;
        case 1:
            SM_X2 = OFF;
            SM_Y2 = ON;
            SM_X1 = ON;
            SM_Y1 = OFF;
            break;
        case 2:
            SM_X2 = OFF;
            SM_Y2 = OFF;
            SM_X1 = ON;
            SM_Y1 = ON;
            break;
        case 3:
            SM_X2 = ON;
            SM_Y2 = OFF;
            SM_X1 = OFF;
            SM_Y1 = ON;
            break;
        }
        for (cnt2 = 0; cnt2 < speed; cnt2++) {
            Delay_ms(1);
        }
    }
}

//*********************************************************************
*

void   SteppingMotorCntl()
{
    unsigned    short    dir, speed, mode;
    unsigned    int        step;
    //
    pnt = 0;
    SM_X2 = ON;
    SM_Y2 = ON;
    SM_X1 = OFF;
    SM_Y1 = OFF;
//    i2c_memory[0] = 1;
//    i2c_memory[1] = 0;
//    i2c_memory[2] = 0;
//    i2c_memory[3] = 200;
//    i2c_memory[4] = 100;
    //
    while (1) {
        if (i2c_stop_flg == 1) {
            i2c_stop_flg = 0;
            //
            mode = i2c_memory[0];
            dir = i2c_memory[1];
```

```
            step = i2c_memory[2];
            step = (step << 8) + i2c_memory[3];
            speed = i2c_memory[4];
            //
            if (mode == 1) {
                SteppingMotorTurn(dir, step, speed);
                i2c_memory[0] = 0;
            }
        }
    }
}

//*****************************************************************************
*

void   main()
{
    unsigned    short    cnt;
    //
    CMCON  = 0b00000111;        //コンパレータは使用しない。
    ANSEL  = 0b00000000;        //A/Dコンバータは使用しない。
    OSCCON = 0b01110000;        //クロックは内臓8MHzを使用する。
    TRISA  = 0b00111100;        //PORTAを設定する。
    TRISB  = 0b00011111;        //PORTBを設定する。
    OPTION_REG.NOT_RBPU = 0;    //PORTBをプルアップする。
    //□□□を設定する。
    SSPSTAT.SMP = 1;
    SSPSTAT.CKE = 1;
    SSPCON.WCOL = 0;
    SSPCON.SSPOV = 0;
    SSPCON.SSPEN = 1;
    SSPCON.CKP = 1;
    SSPCON.SSPM0 = 0;
    SSPCON.SSPM1 = 1;
    SSPCON.SSPM2 = 1;
    SSPCON.SSPM3 = 1;
    SSPADD = ADDR_BASE + ((SW_ADDR3 == 1) ? 8 : 0) + ((SW_ADDR2 == 1) ?
4 : 0) + ((SW_ADDR1 == 1) ? 2 : 0);
    PIE1.SSPIE = 1;
    PIR1.SSPIF = 0;
    //
    LED = OFF;
    i2c_pnt = 0;
    i2c_flg = 0;
    i2c_start_flg = 0;
    i2c_stop_flg = 0;
    for (cnt = 0; cnt < MEMORY_SIZE; cnt++) {
        i2c_memory[cnt] = 0x00;
    }
    //
    for (cnt = 0; cnt < 10; cnt++) {
```

```
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
    //割り込み(全体)の設定
    INTCON.PEIE = 1;
    INTCON.GIE = 1;
    //
    while (1) {
        SteppingMotorCntl();
    }
}

//*********************************************************************
*
```

<参考> ※テストデータ送信用(I2Cマスタ□/PIC12F683)のプログラムです。

SteppingMotor_i2c_master.c

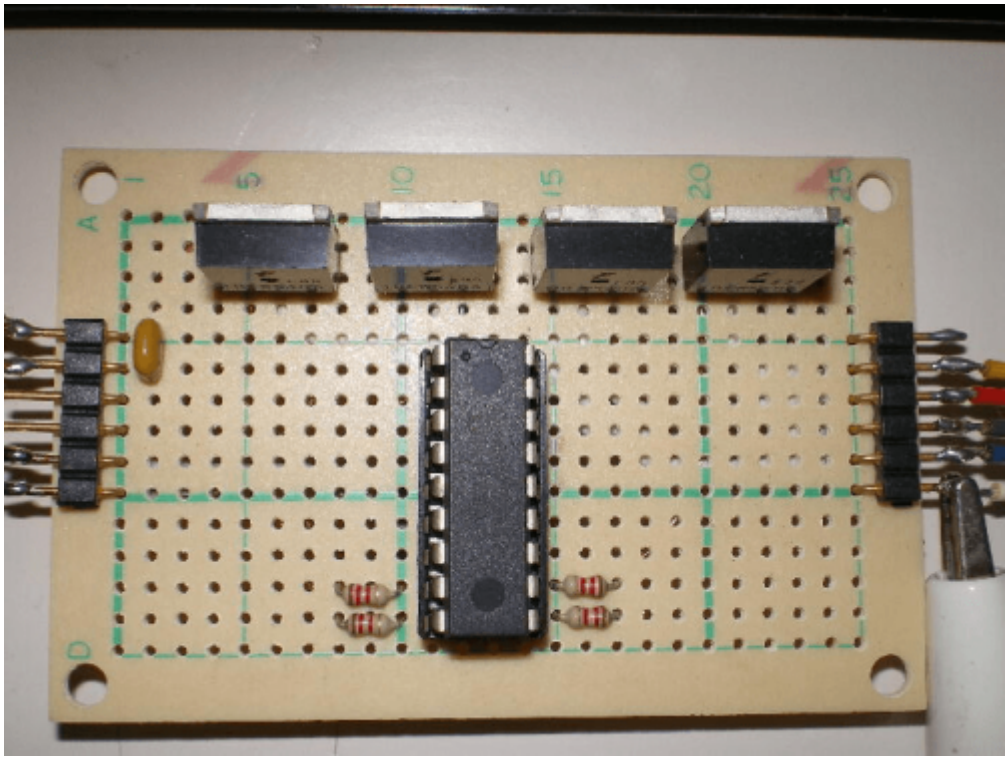
```
//*********************************************************************
*
/*
  『ステッピングモータのテストデータ送信用（マスター）』
*/
//*********************************************************************
*

#define       SW                GPIO.F3
#define       LED                GPIO.F2

#define       ON                1
#define       OFF                0

#define       ACK                1
#define       NO_ACK            0

//*********************************************************************
*

void   SwitchONcheck()
{
    while (Button(&GPIO, 3, 1, 0) == 0)
        ;
    while (Button(&GPIO, 3, 1, 1) == 0)
        ;
}

//*********************************************************************
```

```
*

void main()
{
    unsigned    short    cnt, dat;
    //
    CMCON0 = 0b00000111;
    ANSEL.ANS0 = 0;
    ANSEL.ANS1 = 0;
    ANSEL.ANS2 = 0;
    ANSEL.ANS3 = 0;
    ADCON0.VCFG = 0;
    TRISIO = 0b00001011;
    OSCCON = 0b01110000;
    //
    for (cnt = 0; cnt < 10; cnt++) {
        LED = ON;
        Delay_ms(50);
        LED = OFF;
        Delay_ms(50);
    }
    //
    Soft_I2C_Config(&GPIO, 4, 5);    // SDA, SCL
    //
    while (1) {
        SwitchONcheck();
        //
        Soft_I2C_Start();
        Soft_I2C_Write(0x9E);
        Soft_I2C_Write(0x00);
        Soft_I2C_Write(1);
        Soft_I2C_Write(0);
        Soft_I2C_Write(0);
        Soft_I2C_Write(200);
        Soft_I2C_Write(100);
        Soft_I2C_Stop();
        //
        SwitchONcheck();
        //
        while (1) {
            Soft_I2C_Start();
            Soft_I2C_Write(0x9E);
            Soft_I2C_Write(0x00);
            Soft_I2C_Start();
            Soft_I2C_Write(0x9F);
            dat = Soft_I2C_Read(NO_ACK);
            Soft_I2C_Stop();
            //
            if (dat == 0)
                break;
        }
```

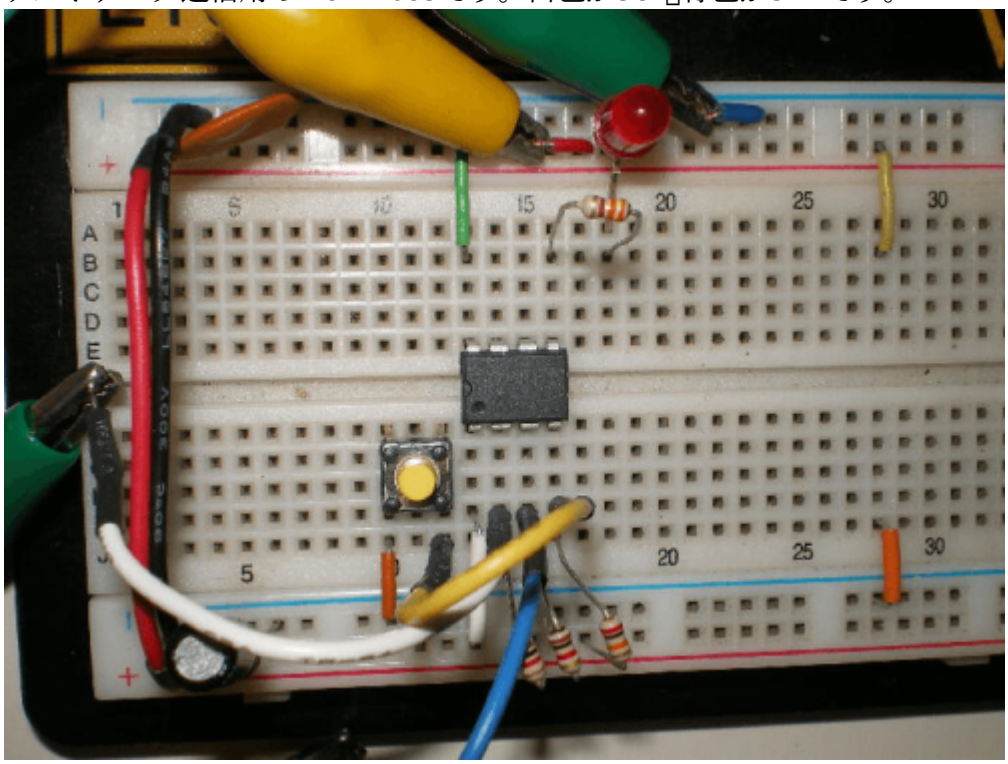
```
        //
        SwitchONcheck();
        //
        Soft_I2C_Start();
        Soft_I2C_Write(0x9E);
        Soft_I2C_Write(0x00);
        Soft_I2C_Write(1);
        Soft_I2C_Write(1);
        Soft_I2C_Write(0);
        Soft_I2C_Write(200);
        Soft_I2C_Write(100);
        Soft_I2C_Stop();
        //
        SwitchONcheck();
        //
        while (1) {
            Soft_I2C_Start();
            Soft_I2C_Write(0x9E);
            Soft_I2C_Write(0x00);
            Soft_I2C_Start();
            Soft_I2C_Write(0x9F);
            dat = Soft_I2C_Read(NO_ACK);
            Soft_I2C_Stop();
            //
            if (dat == 0)
                break;
        }
        //
        for (cnt = 0; cnt < 10; cnt++) {
            LED = ON;
            Delay_ms(50);
            LED = OFF;
            Delay_ms(50);
        }
    }
}

//*****************************************************************************
*
```

動作確認

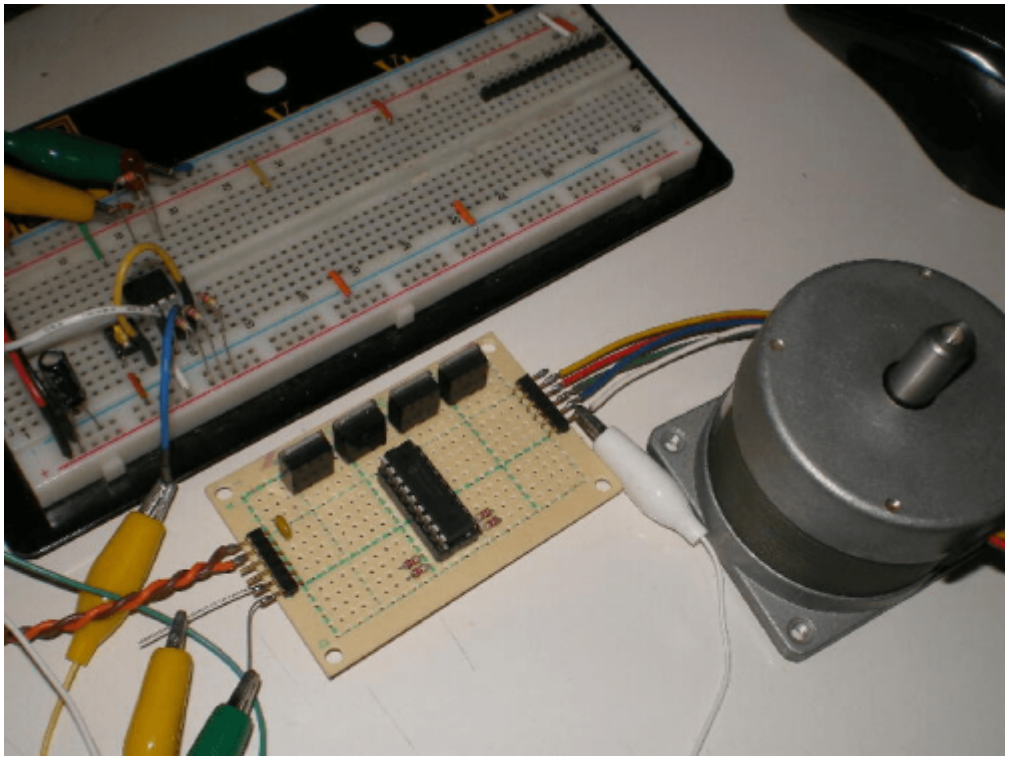
以前に製作した、ステッピングモータ制御の基板を少し改良しました。



テストデータ送信用のPIC12F683です。白色がSCL□青色がSDAです。



マスター側のスイッチを押す毎に、正転(200ステップ/360度)、逆転(200ステップ/360度)を繰り返しま



す。