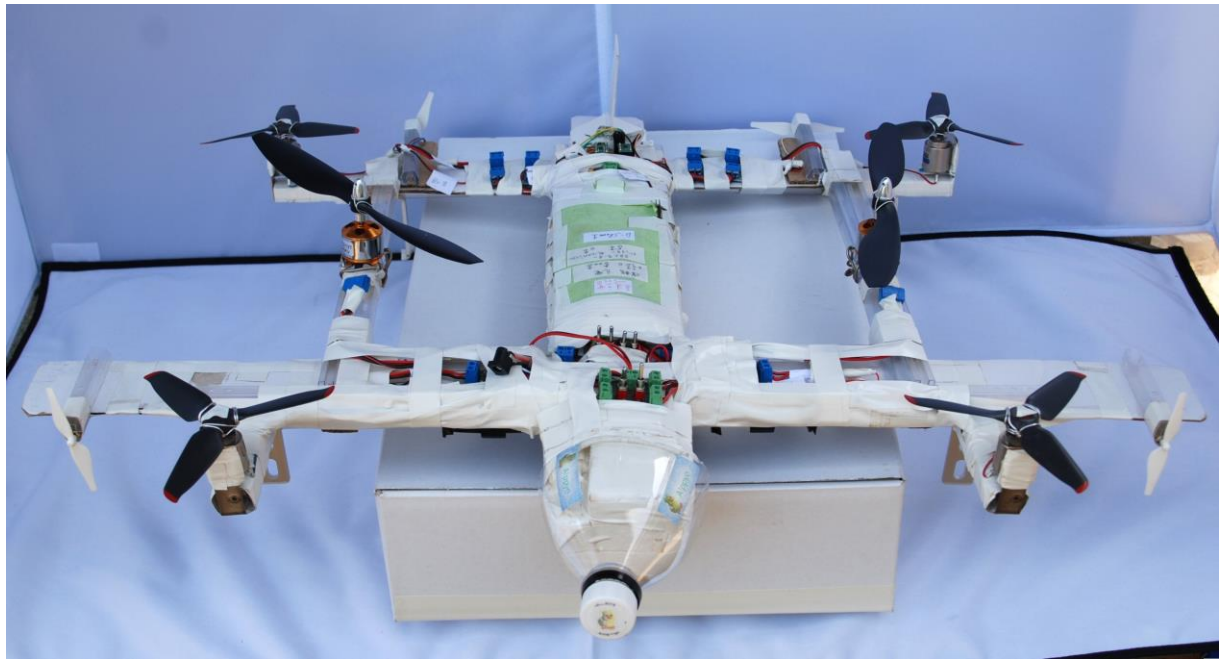
Drone1 (D-slave1)



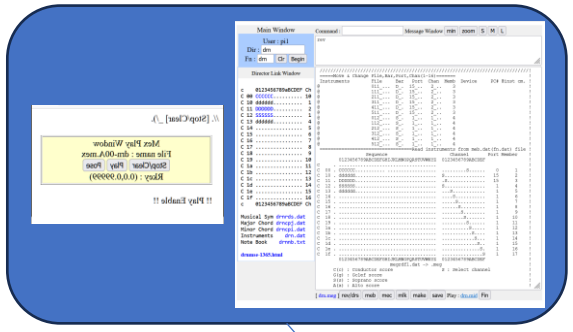
Music Drone Project

製作開始 2022/09/07

アッピョミュージックスタジオ事務所

2024/07/12 h.

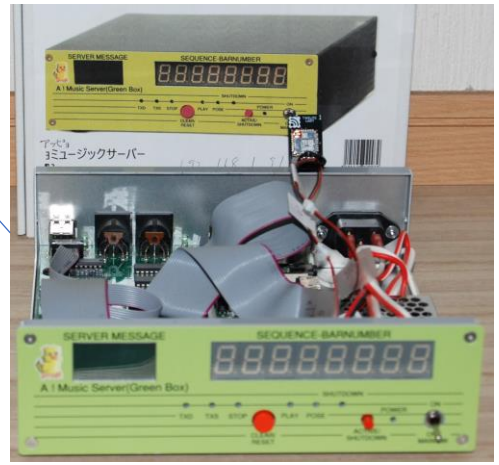
全体構成



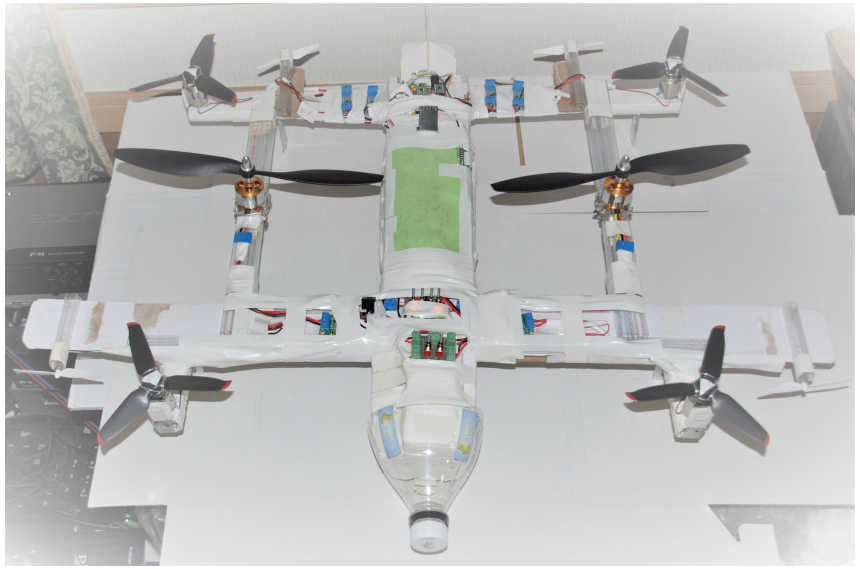
パソコン
Webブラウザ

LAN接続

無線



A! MusicServer



Drone1

部品、材料

プロペラ　大　x２　　モーター　A2212/13T 1000KV
プロペラ　中　x４　　モーター　RE-140RA Mabuchi motor
プロペラ　小　x４　　モーター　Crozepony

LPC基盤
Arduino Uno
AE-Motor8830 x 8
BNO055　９軸センサーモジュール
ICM-20948　９軸センサーモジュール
TWELITE
電池ケース、 Ni-MH Battery
L型金具、アルミ線

本体
発砲スチロール、段ボール、布テープ、ビニールテープ、他

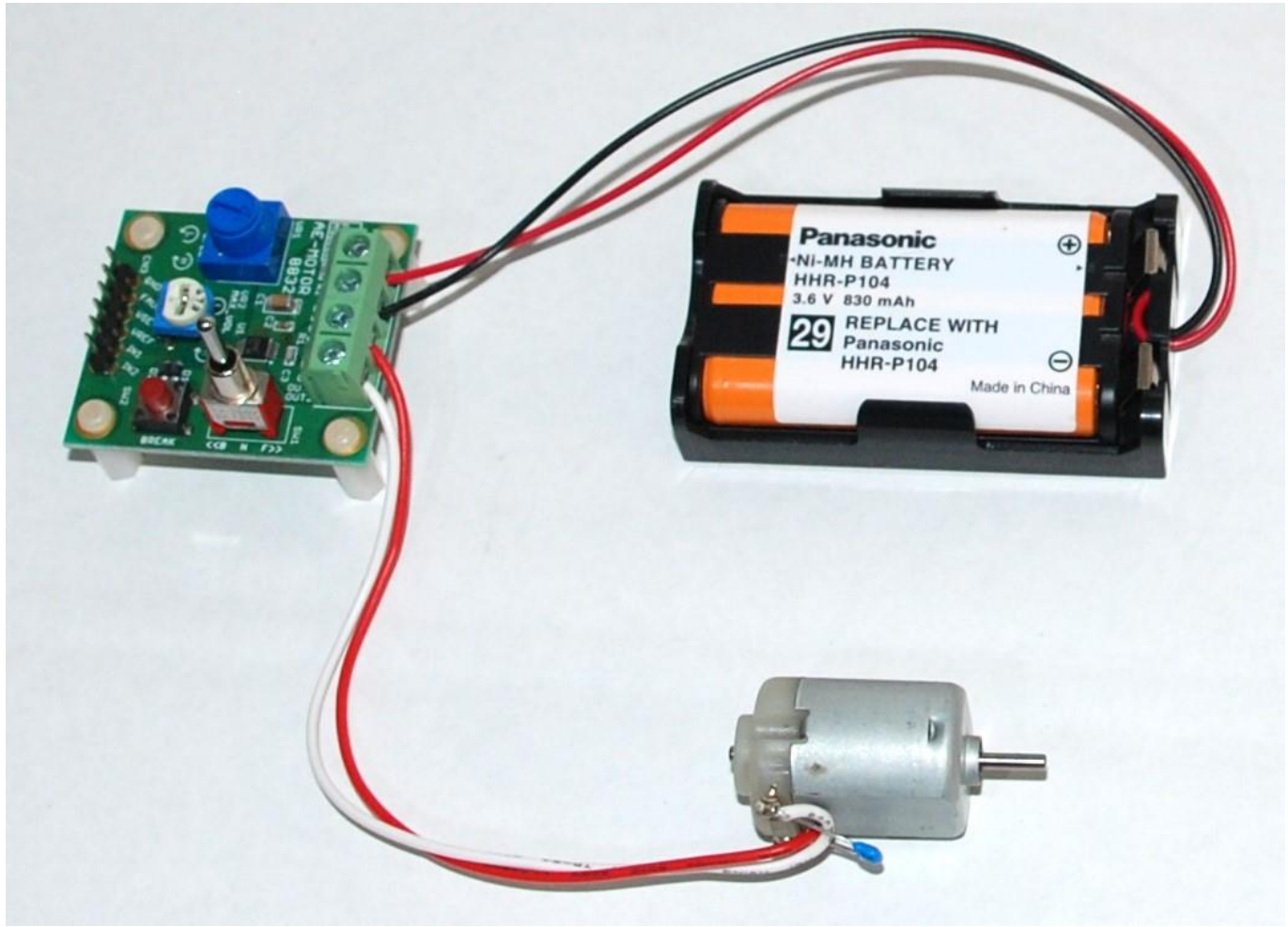
LPC基板 NXP LPC1788 ,ARM ,Cortex-M3
開発環境 IAR Embedded Workbanch for Arm
基板製造 Appyo Music Studio
利用 Interface
　I2c0　DRV8830 x 8 ,　BNO055
　UART0　Arduino Uno との通信
　UART1　TWELITE uart（Music Server との通信）

Arduino Uno
開発環境 Arduino IDE
利用 Interface
　PWM(3,5)　ブラシレスモーター駆動
　　Unoプログラム(sketch_brc_test-2-19)
　UART LPC基盤 との通信
　　Unoプログラム (sketch_dec22a)
　I2C ICM-20948との通信

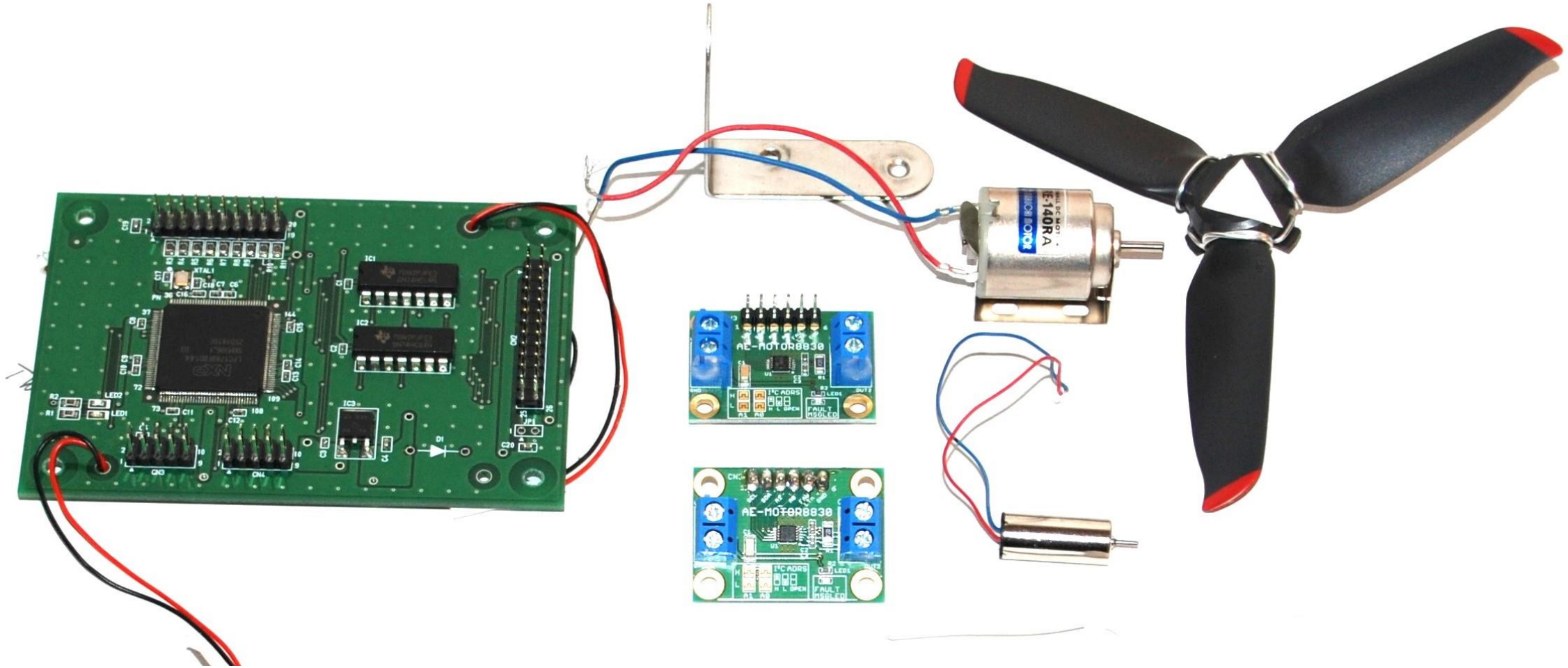
DCモータテスト

DRV8832 FA-130RA

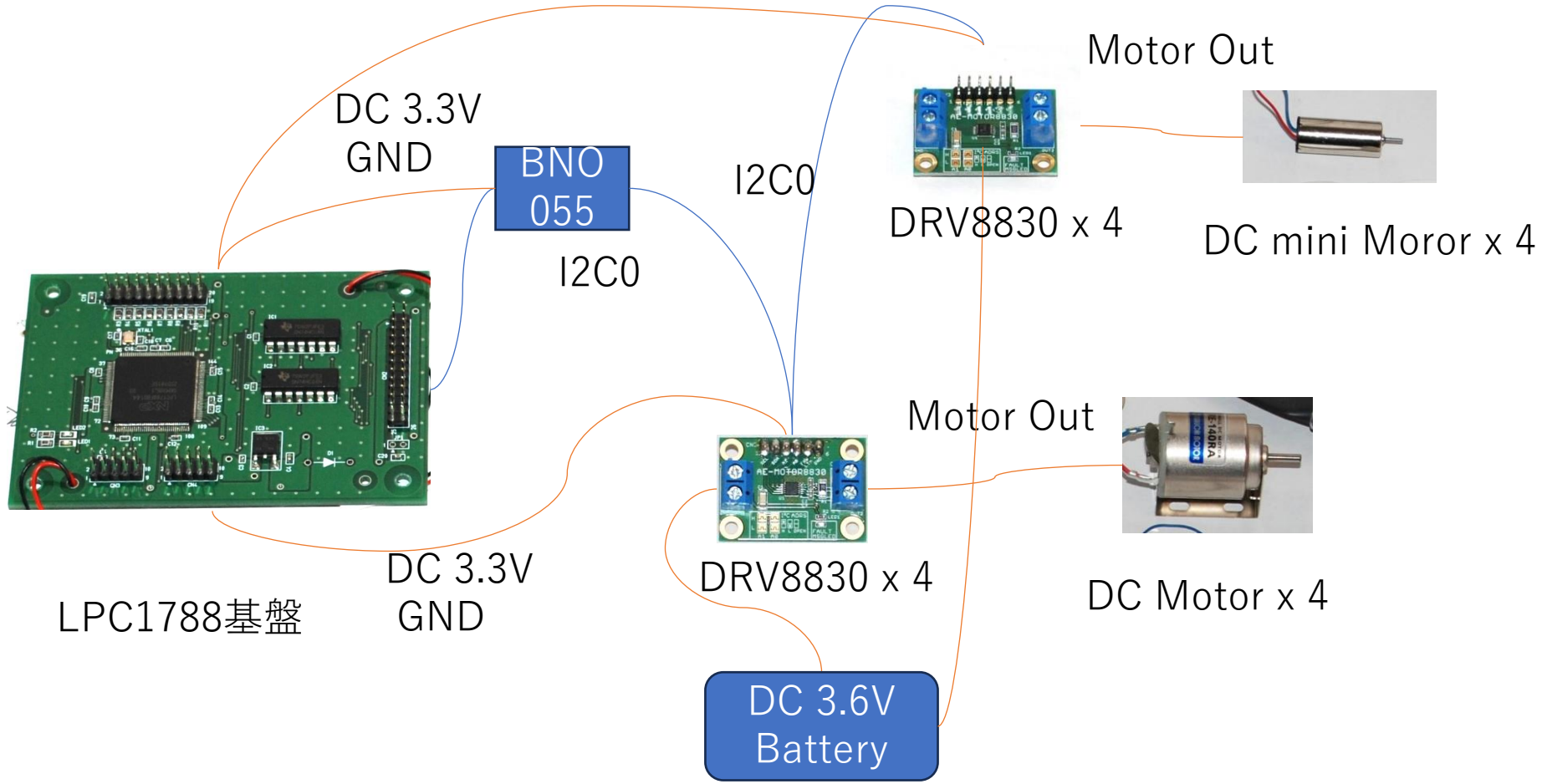
DC 3.6V 正、逆回転 OK



DC motor control , Texas DRV8830 , I2c



DC motor control , Texas DRV8830 , I2c



```
/*****************************************************************************
 **
 ** I2C0
 **
 *****************************************************************************/
/*
int I2C_InitMaster (unsigned long BusSpeed);
int I2C_MasterWrite (unsigned char addr, unsigned char *pMsg , unsigned long numMsg);
int I2C_MasterRead (unsigned char addr, unsigned char *pMsg , unsigned long numMsg);

int I2C_Transfer (unsigned char addr, unsigned char *pMsg , unsigned long numMsg,
        LPC_I2C_TransMode_t transMode, unsigned long numWrite);

void I2C_HandleMasterState(void);
*/

#define IOCON_P5_2      ((volatile unsigned int *)(0x4002C288))
#define IOCON_P5_3      ((volatile unsigned int *)(0x4002C28C))
```

```
// Main Loop
while(1) {
  *IOCON_P1_18 = 0x00;          // GPIO 000:P1[18] 001:USB_UP_LED1
  *IOCON_P5_2 = 0x05;  // pull-up 10K resister need
  *IOCON_P5_3 = 0x05;  // pull-up resister 10K need
  *FGPIO_FIO1DIR =0x00040000;   // Set P1[18] to OutPut
  LED_OFF(); // test
  // LED_ON();

  I2C_HandleMasterState();
  //I2C_MasterWrite (0xc8, pMsg , 2);
  i2c0_init(); // at Drv8830.c  // ICM Solo OK
  //DrvBrA_stp();  // DRV8830 all stop
```

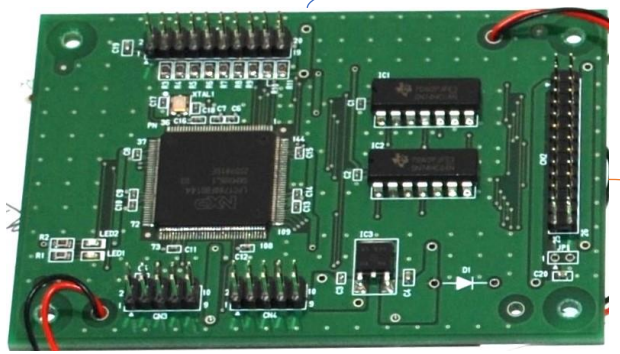
```
/*****************************************************************************
 * Function Name: I2C_InitMaster
 * Parameters: unsigned long BusSpeed
 *
 * Return: int
 *               0: success
 *       non-zero: error number
 * Description: Initialize the current device as I2C bus master.
 *
 *****************************************************************************/
int I2C_InitMaster (unsigned long BusSpeed)
{

  PCONP_bit.PCI2C0 = 1;       // enable I2C0 clock

  if (BusSpeed > I2C_MAXSPEED)
    return 1;
  I2C_DisableI2C();

  Int32U clk = CLK_GetClock(CLK_PERIPH);

  // value of I2SCLH and I2SCLL must be different
  I2C0SCLH = ((clk / BusSpeed) / 2) + 1;
  I2C0SCLL =  (clk / BusSpeed) / 2;

  if (I2C0SCLH < 4 || I2C0SCLL < 4)
    return 1;

  I2CState = I2C_IDLE;

  //
  //IOCON_P2_14 = 0x32;  // pull up resister enabled
  //IOCON_P2_15 = 0x32;  // pill-up resister enabled

  IOCON_P5_02 = 0x05;  // pull-up 10K resister need
  IOCON_P5_03 = 0x05;  // pull-up resister 10K need

        // Enable I2C
  I2C_EnableI2C();
  __I2C_ClearFlag(I2CON_STAC | I2CON_SIC | I2CON_AAC);

  return 0;
}
```

```
//void i2c0_init(addr)
//unsigned char addr;
void i2c0_init()   // 2022.10.3 h.
{
  /*
22.10.1 Initialization routine
Example to initialize I2C Interface as a Slave and/or Master.
1. Load the I2ADR registers and I2MASK registers with values to configure the own
Slave Address, enable General Call recognition if needed.
2. Enable I2C interrupt.
3. Write 0x44 to I2CONSET to set the I2EN and AA bits, enabling Slave functions. F
Master only functions, write 0x40 to I2CONSET
  */

    PCONP_bit.PCI2C0 = 1;   // enable I2C0 clock

    // i2c0 disable
    I2C0CONCLR = (1 << 6);  // 0x40

    I2C0SCLH = 240; // i2c0 OK
    I2C0SCLL = 240; // i2c0 OK

    //I2C0SCLH = 300; // i2c0. OK (ICM Solo)
    //I2C0SCLL = 302; // i2c0 OK (ICM Solo)

    //IOCON_P5_02 = 0x05;  // pn 81 i2c0 SDA
    //IOCON_P5_03 = 0x05;  // pn 98 i2c0 SCL

    *IOCON_P5_2 = 0x05;  // pn 81 i2c0 SDA
    *IOCON_P5_3 = 0x05;  // pn 98 i2c0 SCL

    //I2C0ADR0 = addr;

    // i2c0 enable
    //I2C0CONSET |= (1 << 6);  // 0x40 OK
    I2C0CONSET |= 0x40;  // I2EN

    I2C0CONCLR = (I2CON_STAC | I2CON_SIC | I2CON_AAC); // i2c clear flag

}
```

```
/*
// DRV8830 i2c 2022.9.24 h.  ---
*/
#include "includes.h"
#include "playlpc.h"
#include "drone.h"

extern void UPutch0();
extern void UPutch1();
extern void UPutch2();
extern void UPutch3();
extern void UPutch4();

int drval[] = {0x06,0x09,0x0d,0x12,0x15,0x19,0x1c,0x2d,0x35,0xc3d};

void Drone_Drv8830()
{
    int  i;
    //unsigned char dAddr;
    //unsigned char bAddr;
    //unsigned char pMsg[4];
    //unsigned char bMsg[32];
    char msg[80];

    //dAddr = 0xc8 >> 1;  // DRV8830 c8 br1
    //dAddr = 0xc6 >> 1;  // DRV8830 c6 br2
    //dAddr = 0xce >> 1;  // DRV8830 ce br3
    //dAddr = 0xc0 >> 1;  // DRV8830 c0 br4
    //dAddr = 0xc2 >> 1;  // DRV8830 c2 br5
    //dAddr = 0xca >> 1;  // DRV8830 ca br6
    //dAddr = 0xcc >> 1;  // DRV8830 cc br7
    //dAddr = 0xd0 >> 1;  // DRV8830 d0 br8

    // int I2C_MasterWrite (unsigned char addr, unsigned char *pMsg , unsigned lor
    //pMsg[0] = 0x00;  // Sub Address
    //pMsg[1] = 0x49;  // 1.45 V Forward  0x48:Standby 0x49:Forward 0x4a:Reverse
    //pMsg[1] = 0x31;  // 0.96 V forward
    //pMsg[1] = 0x1e;  // 0.56 V Reverse
    //pMsg[1] = 0x1d;  // 0.56 V Forward
    //pMsg[1] = 0x19;  // 0.48 V Forward
    //pMsg[1] = 0x21;  // 0.64 V Forward
    //pMsg[1] = 0x99;  // 3.05 V Forward
    //pMsg[1] = 0x9a;  // 3.05 V Forward
    //pMsg[1] = 0xc9;  // 4.02 V Forward
    //pMsg[1] = 0xf4;  // 4.90 V Forward

    strcpy(msg,"///Drone_Drv8830()//¥r¥n");
    //UPuts1(msg);
    //for(i=0;i<30;i++) DrvBr1_tst();
    DrvBr1_tst();  // front L
    DrvBr2_tst();  // front R
    DrvBr3_tst();  // rear L
    DrvBr4_tst();  // rear R
    DrvBr5_tst();  // front L run
    DrvBr6_tst();  // front R run
    DrvBr7_tst();  // rear L run
    DrvBr8_tst();  // rear R run
    //DrvBrA_tst();  // all test
    //DrvBrA_stp();  // all stop
    DrvBrA_act();  // all active
    for( i=0; i < 20000000; i++); //delay
    for(i=0;i<2;i++) DrvBrA_stp();  // all stop

    //DrvBr3_tst_i2c1();  // i2c1 rear L test NG ?
}

void DrvBr1_tst()
{
    int  i;

    unsigned char dAddr;
    unsigned char pMsg[4];

    dAddr = 0xc8 >> 1;  // DRV8830 c8 br1

    I2C_MasterWrite (dAddr, pMsg, 0);
    i2c0_forward_1(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_standby(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_forward_4(dAddr);
    for( i=0; i < 40000000; i++); //delay
    i2c0_standby(dAddr);
}

void DrvBr2_tst()
{
    int  i;

    unsigned char dAddr;
    unsigned char pMsg[4];

    dAddr = 0xc6 >> 1;  // DRV8830 c6 br2

    I2C_MasterWrite (dAddr, pMsg, 0);
    i2c0_forward_1(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_standby(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_forward_4(dAddr);
    for( i=0; i < 40000000; i++); //delay
    i2c0_standby(dAddr);
}

void DrvBr3_tst()
{
    int  i;

    unsigned char dAddr;
    unsigned char pMsg[4];

    dAddr = 0xce >> 1;  // DRV8830 ce br3

    I2C_MasterWrite (dAddr, pMsg, 0);
    i2c0_forward(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_standby(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_forward_4(dAddr);
    for( i=0; i < 40000000; i++); //delay
    i2c0_standby(dAddr);
}


void DrvBr4_tst()
{
    int  i;

    unsigned char dAddr;
    unsigned char pMsg[4];

    dAddr = 0xc0 >> 1;  // DRV8830 c8 br4

    I2C_MasterWrite (dAddr, pMsg, 0);
    i2c0_forward(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_standby(dAddr);
    for( i=0; i < 10000000; i++); //delay
    i2c0_forward_4(dAddr);
    for( i=0; i < 40000000; i++); //delay
    i2c0_standby(dAddr);
}
```

LPC1788 I2c BNO055
（９軸センサーモジュール）

LPC1788基板



I2C0 SCL/SDA

DC 3.3V
/GND

BNO055 ９軸センサーモジュール

次ページ、LPC1788 BNOソースコード

```c
/*
// Fusion BNO055 i2c 2022.11.1 h.  ---
*/
#include "includes.h"
#include "playlpc.h"
#include "drone.h"

extern void UPutch0();
extern void UPutch1();
extern void UPutch2();
extern void UPutch3();
extern void UPutch4();

extern int twflg;

extern int  btf;    // Bno temperature ON/OFF flag
extern int  bcf;    // Bno Calib_Stat ON/OFF flag
extern int  bgf;    // Bno Gyroscope ON/OFF flag
extern int  baf;    // Bno Acceleration ON/OFF flag
extern int  bmf;    // Bno Magnetmerer ON/OFF flag
extern int  bhf;    // Bno Heading Roll Pitch ON/OFF flag
extern int  bqf;    // Bno Quaternion ON/OFF flag
extern int  blf;    // Bno Linear Acceleration ON/OFF flag
extern int  bvf;    // Bno graVity Vector ON/OFF flag

int Drone_BNO055()
{
    int  i,j;
    //unsigned char dAddr;
    unsigned char bAddr;
    unsigned char pMsg[4];

    //unsigned char bMsg[32];
    //bAddr = 0x50 >> 1;  // BNO055 test
    bAddr = 0x28;  // BNO055 0x28
    I2C_MasterWrite (bAddr, pMsg, 0);
    //dAddr = 0xc8 >> 1;  // DRV8830 c8

    // BNO055 ////
    // OPR_MODE Reg. 0x3d Def 0x10 Config Mode
    // Fusion Mode -> IMU , COMPASS , M4G , NDOF_FMC_OFF , NDOF
    // Test Use -> NDOF Absolute orientation

    for( i=0; i < 300000; i++); //delay
    pMsg[0] = 0x3d;      // Opreating Mode reg.
    //pMsg[1] = 0x08;    // Set IMU Mode
    //pMsg[1] = 0x09;    // Set Compass Mode
    //pMsg[1] = 0x0a;    // Set M4G Mode
    //pMsg[1] = 0x0b;    // Set NDOF_FMC_OFF mode
    pMsg[1] = 0x0c;    // Set NDOF Mode

    I2C_MasterWrite (bAddr, pMsg, 2);
    for( i=0; i < 300000; i++); //wait setting

    // ***********************************
    //  Read Fusion data
    for( j=0; j < 1; j++) {

        if(btf) Read_Temperature(bAddr);

        if(bcf) Read_Calib_Stat(bAddr);

        //  Gyroscope
        if(bgf) Read_Gyroscope_data(bAddr);

        //  Acceleration
        if(baf) Read_Acceleration_data(bAddr);

        //  Magetometer
        if(bmf) Read_Magnetometer_data(bAddr);

        //  EUL Heading Roll Pitch
        if(bhf) Read_Heading_data(bAddr);

        //  Quaternion
        if(bqf) Read_Quaternion_data(bAddr);

         //  Linear Acceleration
        if(blf) Read_Linear_Acceleration(bAddr);

         //  Gravity vector
        if(bvf) Read_Gravity_vector_data(bAddr);
    }
    // ***********************************

    // UPutch1(ESC); // ESC moniter fin.

    return(0);
}
```

```c
void Read_Temperature(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x34; //  Acceleration
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 1);
        if(twflg) {
            sprintf(msg,"Bno-Temperature: x%02x¥r¥n",bMsg[0]);
            UPuts1(msg);
        }
        else {
            UPutch1('T');
            UPutch1(':');
            UPutch1(bMsg[0]);
        }
        //UPuts2(msg);  // test
    }
}

void Read_Calib_Stat(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x35; //  Acceleration
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 1);
        sprintf(msg,"Bno-Calib_Stat: x%02x¥r¥n",bMsg[0]);
        UPuts1(msg);
    }
}

void Read_Acceleration_data(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x08; //  Acceleration
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 6);
        sprintf(msg,"Bno-Acceleration: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5]);
        UPuts1(msg);
    }
}

void Read_Magnetometer_data(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x0e; //  Magnetometer
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 6);
        sprintf(msg,"Bno-Magnetometer: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5]);
        UPuts1(msg);
    }
}
```

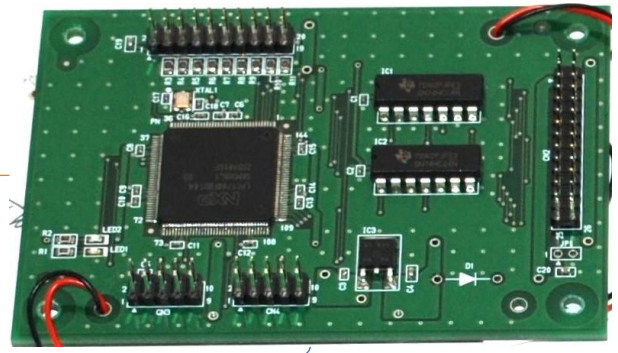
```c
void Read_Gyroscope_data(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x14; //  Gyroscope
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 6);
        sprintf(msg,"Bno-Gyroscope: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5]);
        UPuts1(msg);
    }
}


void Read_Heading_data(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x1a; //  Heading Roll Pich
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 6);
        sprintf(msg,"Bno-Heading: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5]);
        UPuts1(msg);
    }
}


void Read_Quaternion_data(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x20; //  Quaternion
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 8);
        sprintf(msg,
         "Bno-Quaternion: x%02x x%02x x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5],bMsg[6],bMsg[7]);
        UPuts1(msg);
    }
}

void Read_Linear_Acceleration(bAddr)
unsigned char bAddr;
{
    int  i,j;
    unsigned char pMsg[4];
    unsigned char bMsg[8];
    char msg[80];

    pMsg[0] = 0x28; // Linear Acceleration
    I2C_MasterWrite (bAddr, pMsg, 1);
    for( j=0; j < 1; j++) {
        for( i=0; i < 300000; i++); //delay
        I2C_MasterRead (bAddr, bMsg, 6);
        sprintf(msg,"Bno-Linear_Acceleration: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        bMsg[0],bMsg[1],bMsg[2],bMsg[3],bMsg[4],bMsg[5]);
        UPuts1(msg);
    }
}
```

LPC1788 UART TWELITE 通信

TWLITE UART
(Blue)

DC 3.3V
/GND

LPC1788基板



サーバーとの接続無線

UART1

次ページ、TWELITE通信ソースコード

```
/*****************************************************************************
 * Function Name: UART1_IRQHandler
 * Parameters: none
 *
 * Return: none
 *
 * Description: UART 1 interrupt routine
 *
 *****************************************************************************/
void UART1_IRQHandler(void)
{
    //LED_ON();// int check
    recv1_data= *COM_RBR1;      // Recive

    //recv1_flg = 1;
    pcomrv_drn();   // U1 recive
    NVIC_ClrPend(NVIC_UART1);  // Clear interrupt in NVIC.
}
```

```
// CPU Clock 96 MHz , PCLK 48MHz , Baud Rate 19200 bps
void U1_48M_19200_int(void)
{
    int  Div;
    int  AddDiv;
    int  Mul;
    //int  Tmp;

    //*POW_PCONP = *POW_PCONP & (1 << 4);  // bit4 -> 1 UART1 ON
    PCONP_bit.PCUART1 = 1;  // Power ON

    Div = 104;
    AddDiv = 1;
    Mul = 2;           // MuVal

    *COM_LCR1 = 0x80;      // Enable access to devider Latches
    *COM_DLL1 = Div & 0xFF;
    *COM_DLM1 = (Div >> 8) & 0xFF;
    *COM_FDR1 = AddDiv + (Mul << 4);
    U1LCR_bit.DLAB = 0;    // Disable access to devider Latches
    *COM_LCR1 = 0x03;      // No Party , 1 Stop bit / 8 bits
    *COM_IER1 = 1;         // int use

    *IOCON_P0_15 = 0x31;     // TXD Type_D func:001_TXD2 Mode:Pull-UP HYS:1
    *IOCON_P0_16 = 0x31;     // RXD Type_D func:001_TXD2 Mode:Pull-UP HYS:1

    // *ISER0 = *ISER0 | 0x40;      // UART1 interrupt enable
    // Transmit enable
    // U1TER_bit.TXEN = 1;
    // Tmp = U1IER;  // Clear pending interrupts
    // U1IER = 0x01;  // RBR interrupt Enable
    // Enable NVIC UART1 Interrupt
    NVIC_IntEnable(NVIC_UART1);

    recv1_flg = 0;              // Uart1 Recive flag Clear
}
```

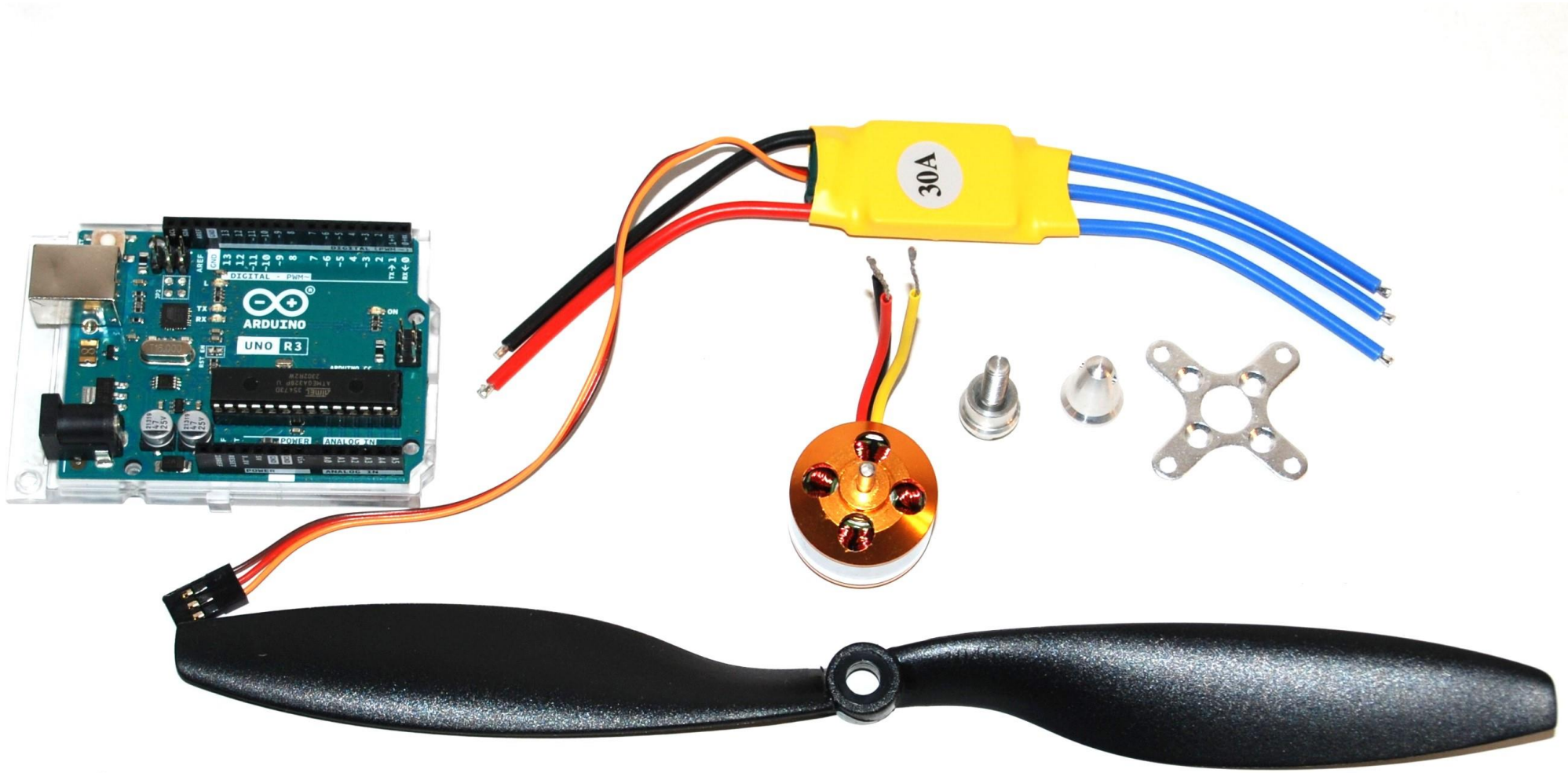
```
/* UART1 recive routine, this is called by int routine
*/
void pcomrv_drn()
{
    if(!full) {
        //playsf = 1;
        size = getsize();
        if(size > OFFLIM1) {
            switch (dcflg) {
                case   0: // UPutch1(XOFF);
                          //dcflg++;
                          break;
                case   1: if(size > OFFLIM2) {
                              // if(size == OFFLIM2) {
                              //UPutch1(XOFF);
                              //UPutcs1_tm(XOFF);
                              //dcflg++;
                          }
                          break;
                default : //if(size == OFFLIM3) UPutcs1_tm(XOFF);
                          break;
            }
        }
        *tail = recv1_data;
        tail++;
        dcflg++;
        if(tail == (buffer + BUFSIZE)) tail = buffer;
        if(head == tail) full = TRUE;
    }
}
```

```c
/* read recive data from buffer
*/
unsigned char qretrive_drn()
{
    unsigned char c,g;
    int  i,j;
    int  twf;
    char rmsg[180];
    int  k,kc;

    if(head == tail) {
        if(full) {
            full = FALSE;
        }
        else  {
            full = TRUE;
        }
    }
    i = 0;
    k = 0;
    twf = 0;
    while(1) {
        if(dcflg > 24) {
            g = *head;
            head++;
            dcflg--;
            rmsg[i] = g;
            if(g == 'U') {
                 i = 0;
                 rmsg[i] = g;
                 twf = 1;
            }
            i++;
            if(i > 180) i = 0;
            if( twf && g == ';') twf++;  // U; ; ; ; ; ; ;data;¥r¥n
            if(twf == 9) {
                 twf = 0;
                 j = 0;
                 kc = 0;
                 while( rmsg[j] != '¥0') {
                     if(rmsg[j] == ';') kc++;
                     //if(rmsg[j] == ',') kc++;
                     j++;
                     if(j > 180) break;
                 }
                 if(kc == 8) {
                     if(k == 0) c = TWEdecode_drn(rmsg); // Decode TWELITE data  mpiw.c
                     for(j=0; j<180 ;j++) rmsg[j] = 0x00;
                     return(c);
                 }
            }
            //head++;
            if(head+1 == tail) while(dcflg <= 0);
            if(head == (buffer + BUFSIZE)) head = buffer;
            //if(dcflg && (getsize() < ONLIMIT)) {
                //UPutcs1_tm(XON);
                //dcflg = 0;
            //}
        }
    }
    //return(ESC);
}
```

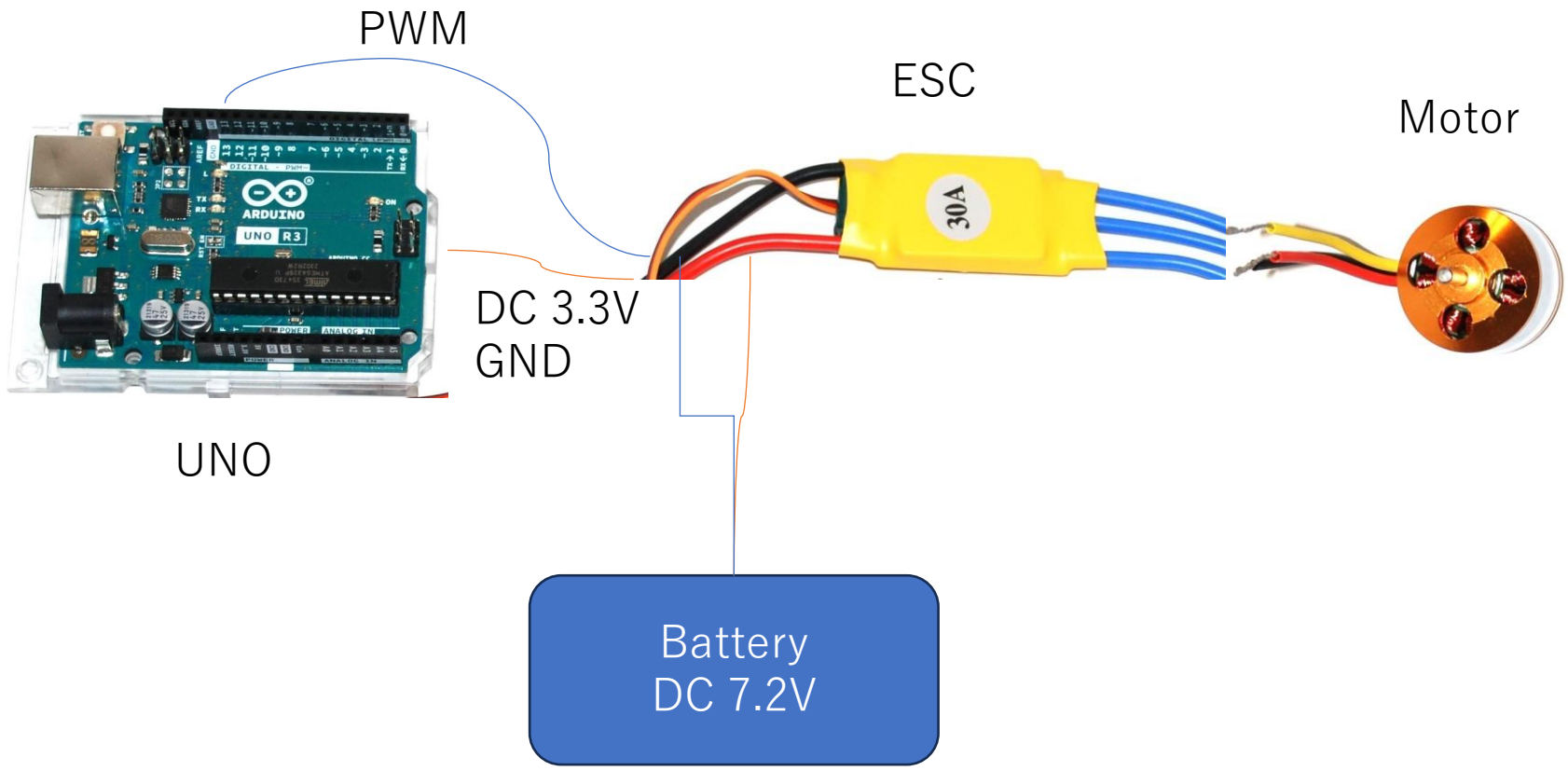
```c
// Decore TWELITE data
unsigned char TWEdecode_drn(rmsg)
char rmsg[180];
{
    unsigned char c;
    int  i,j,k,m;
    //char wk[180];
    char rd[80];
    int  slen;
    char *hex = "0123456789abcdef";
    int ic,ic1,ic2;

    if(rmsg[0] == 'U' && rmsg[1] == ';') {
       slen = strlen(rmsg);
       k = 0;
       m = 0;
       for( i = slen ; i > 0 ; i--) {
            if(rmsg[i] == ';') k++;
            if(rmsg[i] == ',') k++;
            //if(rmsg[i] == ',') m++;
            if( k == 3) break;
       }
       j = 0;
       i++;
       while(1) {
            if(rmsg[i] == ';') break;
            rd[j] = rmsg[i];
            i++;
            j++;
            if(j > 80) break;

       }
       rd[j] = '¥0';
       if(strlen(rd) == 2) {
            for(m = 0; m < 16 ; m++) {
                 if(rd[0] == hex[m]) {
                    ic1 = m;
                    ic1 = ic1 << 4;
                 }
            }
            for(m = 0; m < 16 ; m++) {
               if(rd[1] == hex[m]) ic2 = m;
            }
            ic = ic1 | ic2;
            c = (char) ic;
            //sprintf(wk,"c -> %02x ic1=%02x ic2=%02x¥n¥n",c,ic1,ic2);
            //UPuts1(wk);
            //UPutcs1_tm(c);  // test
       }
    }
    //else return(ESC);
    return(c);
}
```

```
int midiplay_drn()
{
    unsigned char c;
    //char wk[80];

    if(dcflg) c = qretrive_drn();
    if(c == 0x00) return(0);
    //sprintf(wk,"midyplay() -> %02x¥n¥n",c);
    //UPuts1(wk);

    midiBrchk_drn(c);

    //midiP15c1_drn(c);
    //if( c == 0x90 ) midiP15c1_drn(c);
    //if( c == 0x91 ) midiP15c2_drn();
    return(0);
}

int midiBrchk_drn(c)
unsigned char c;
{
    int  vd;
    int  vi;
    char wk[80];

    vi = c & 0xf0;
    vi = vi >> 4;
    vd = c & 0x0f;
    switch(vi) {
        case  1 : // Br1
                  if(vd == 0) sprintf(wk,"T Stop Br1 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br1 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b1_drn(vd);
                  break;
        case  2 : // Br2
                  if(vd == 0) sprintf(wk,"T Stop Br2 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br2 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b2_drn(vd);
                  break;
        case  3 : // Br3
                  if(vd == 0) sprintf(wk,"T Stop Br3 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br3 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b3_drn(vd);
                  break;
        case  4 : // Br4
                  if(vd == 0) sprintf(wk,"T Stop Br4 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br4 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b4_drn(vd);
                  break;
        case  5 : // Br5
                  if(vd == 0) sprintf(wk,"T Stop Br5 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br5 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b5_drn(vd);
                  break;
        case  6 : // Br6
                  if(vd == 0) sprintf(wk,"T Stop Br6 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br6 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b6_drn(vd);
                  break;
        case  7 : // Br7
                  if(vd == 0) sprintf(wk,"T Stop Br7 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br7 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b7_drn(vd);
                  break;
        case  8 : // Br8
                  if(vd == 0) sprintf(wk,"T Stop Br8 > %02x¥n¥n",c);
                  else sprintf(wk,"T Run Br8 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b8_drn(vd);
                  break;
        case  9 : // Brc1
                  sprintf(wk,"T Run Brc1 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15b9_drn(vd);
                  break;
        case 10 : // Brc2
                  sprintf(wk,"T Run Brc2 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15ba_drn(vd);
                  break;
        case 11 : // Brc1,2
                  sprintf(wk,"T Run Brc1 Brc2 > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15bb_drn(vd);
                  break;
        case 12 : //
                  midiP15bc_drn(vd);
                  break;
        case 13 : // set Bno & Icm-UNO flags ON
                  if(vd == 1) sprintf(wk,"T Bno Temperature on > %02x¥n¥n",c);
                  if(vd == 2) sprintf(wk,"T Bno Calib_Stat on > %02x¥n¥n",c);
                  if(vd == 3) sprintf(wk,"T Bno Gyro on > %02x¥n¥n",c);
                  if(vd == 4) sprintf(wk,"T Bno Acceleration on > %02x¥n¥n",c);
                  if(vd == 5) sprintf(wk,"T Bno Mag. on > %02x¥n¥n",c);
                  if(vd == 6) sprintf(wk,"T Bno Heading on > %02x¥n¥n",c);
                  if(vd == 7) sprintf(wk,"T Bno Quatererion on > %02x¥n¥n",c);
                  if(vd == 8) sprintf(wk,"T Bno Linear Accel. on > %02x¥n¥n",c);
                  if(vd == 9) sprintf(wk,"T Bno Gravity on > %02x¥n¥n",c);
                  if(vd == 10) sprintf(wk,"T ICM Gravity on > %02x¥n¥n",c);
                  if(vd == 11) sprintf(wk,"T ICM Acceleration on > %02x¥n¥n",c);
                  if(vd == 12) sprintf(wk,"T ICM Mag on > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15bd_drn(vd);
                  break;
        case 14 : // set Bno & Icm-UNO flags OFF
                  //sprintf(wk,"T Bno & ICM off > %02x¥n¥n",c);
                  if(vd == 1) sprintf(wk,"T Bno Temperature off > %02x¥n¥n",c);
                  if(vd == 2) sprintf(wk,"T Bno Calib_Stat off > %02x¥n¥n",c);
                  if(vd == 3) sprintf(wk,"T Bno Gyro off > %02x¥n¥n",c);
                  if(vd == 4) sprintf(wk,"T Bno Acceleration off > %02x¥n¥n",c);
                  if(vd == 5) sprintf(wk,"T Bno Mag. off > %02x¥n¥n",c);
                  if(vd == 6) sprintf(wk,"T Bno Heading off > %02x¥n¥n",c);
                  if(vd == 7) sprintf(wk,"T Bno Quatererion off > %02x¥n¥n",c);
                  if(vd == 8) sprintf(wk,"T Bno Linear Accel. off > %02x¥n¥n",c)
                  if(vd == 9) sprintf(wk,"T Bno Gravity off > %02x¥n¥n",c);
                  if(vd == 10) sprintf(wk,"T ICM Gravity off > %02x¥n¥n",c);
                  if(vd == 11) sprintf(wk,"T ICM Acceleration off > %02x¥n¥n",c)
                  if(vd == 12) sprintf(wk,"T ICM Mag off > %02x¥n¥n",c);
                  UPuts1(wk);
                  midiP15be_drn(vd);
                  break;
        case 15 : // All Stop
          if(c == 0xf0) {
                  sprintf(wk,"T Run All stop > %02x¥n¥n",c);
                  UPuts1(wk);
                      DrvBrA_stp();  // Br All stop
                      MsgAll_stp();  // Bno & Icm Msg stop
                  }
                  break;
        default  : break;
    }
    return(0);
```

DCブラシレスモーター, UNO, I2c



DCブラシレスモーター配線

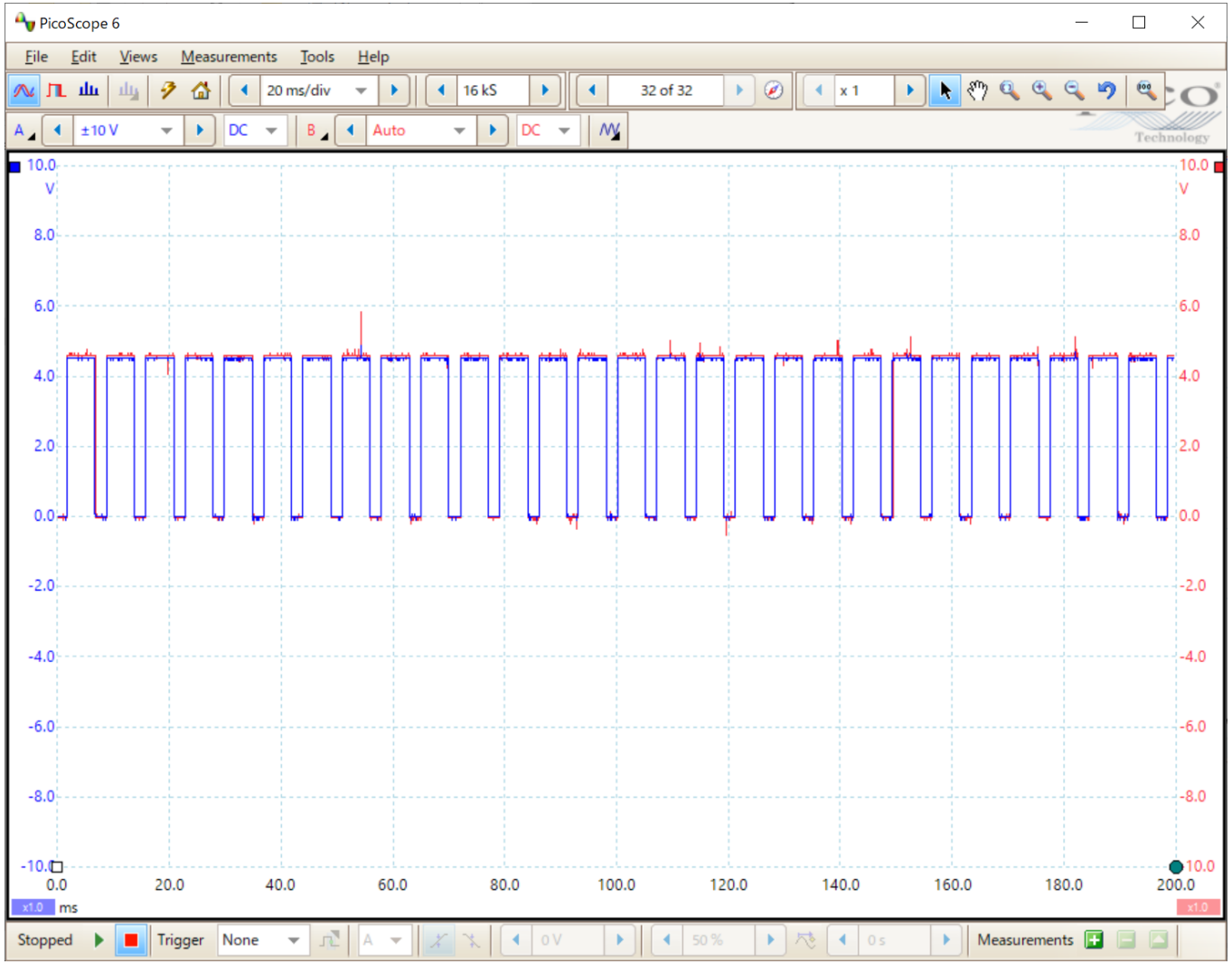


次ページ UNO ソースコード

```
#include <Wire.h>

int i;
int rvc;
int data = 0;
int data1;
int nonf = 0;
int addr = 0x68;  // ICM-20948
int addr2 = 0x0c; // AK09916 Mag address
int d1,d2,d3,d4,d5,d6;
char msg[30];
unsigned char md1,md2,md3;
int ESCLoopTimer;
int ESC1_timer;
int ESC2_timer;
int ESC3_timer;
int ESC4_timer;

void setup() {
  // put your setup code here, to run once:
  //Serial.begin(19200);
  //Wire.begin();
  pinMode(3,OUTPUT);
  pinMode(5,OUTPUT);
  //pinMode(6,OUTPUT);
  //pinMode(9,OUTPUT);
  //Serial.write("////// Arduino setup() ///////¥r¥n");
}

void loop() {
  // put your main code here, to run repeatedly:
  // UART check
  actESC();
  pwmESC();
}

void actESC()
{
  for(i=0; i <1000;i++) {
    digitalWrite(3,LOW);
    digitalWrite(5,LOW);
    delay(2);  // ms
    //for(i=0; i < 5000; i++);
    digitalWrite(3,HIGH);
    digitalWrite(5,HIGH);
     //for(i=0; i < 5000; i++);
    delay(5);
  }
}

void pwmESC()
{
  while(1) {
    // PWM 3,5,6,9 pins -> high
    digitalWrite(3,HIGH);
    digitalWrite(5,HIGH);
    //digitalWrite(6,HIGH);
    //digitalWrite(9,HIGH);

    ESCLoopTimer = micros();
    //ESCLoopTimer = 300;
    //ESC1_timer = 9900 + ESCLoopTimer; // Run OK
    ESC1_timer = 100 + ESCLoopTimer; // RUN OK ? temp
    ESC2_timer = 125 + ESCLoopTimer;
    //ESC3_timer = 1500 + ESCLoopTimer;
    //ESC4_timer = 2000 + ESCLoopTimer;

    //while(digitalRead(3) + digitalRead(5) + digitalRead(6) + digitalRead(9) > 0) {
    //while(digitalRead(3) > 0) {
    while(digitalRead(3) + digitalRead(5) > 0) {
      ESCLoopTimer = micros();
      //ESCLoopTimer += 20;
      if(ESC1_timer <= ESCLoopTimer && digitalRead(3) > 0) {
        digitalWrite(3,LOW);
      }

      if(ESC2_timer <= ESCLoopTimer && digitalRead(5) > 0) {
        delay(2);
        digitalWrite(5,LOW);
      }
      //if(ESC3_timer <= ESCLoopTimer) digitalWrite(6,LOW);
      //if(ESC4_timer <= ESCLoopTimer) digitalWrite(9,LOW);
    }

    //while(digitalRead(3) + digitalRead(5) + digitalRead(6) + digitalRead(9) <= 0) {
    //while(digitalRead(3) <= 0) {
    while(digitalRead(3) + digitalRead(5) <= 0) {
      ESCLoopTimer = micros();
      //ESCLoopTimer += 300;
      //delay(1);
      if(ESC1_timer > ESCLoopTimer && (digitalRead(3) <= 0)) {
        digitalWrite(3,HIGH);
        delay(2);
        break;
      }
      //delay(1);
      //ESCLoopTimer += 250;
      if(ESC2_timer > ESCLoopTimer && (digitalRead(5) <= 0)) {
        digitalWrite(5,HIGH);
        break;
      }
      //if(ESC3_timer > ESCLoopTimer) digitalWrite(6,HIGH);
      //if(ESC4_timer > ESCLoopTimer) digitalWrite(9,HIGH);
    }
  }
}
```

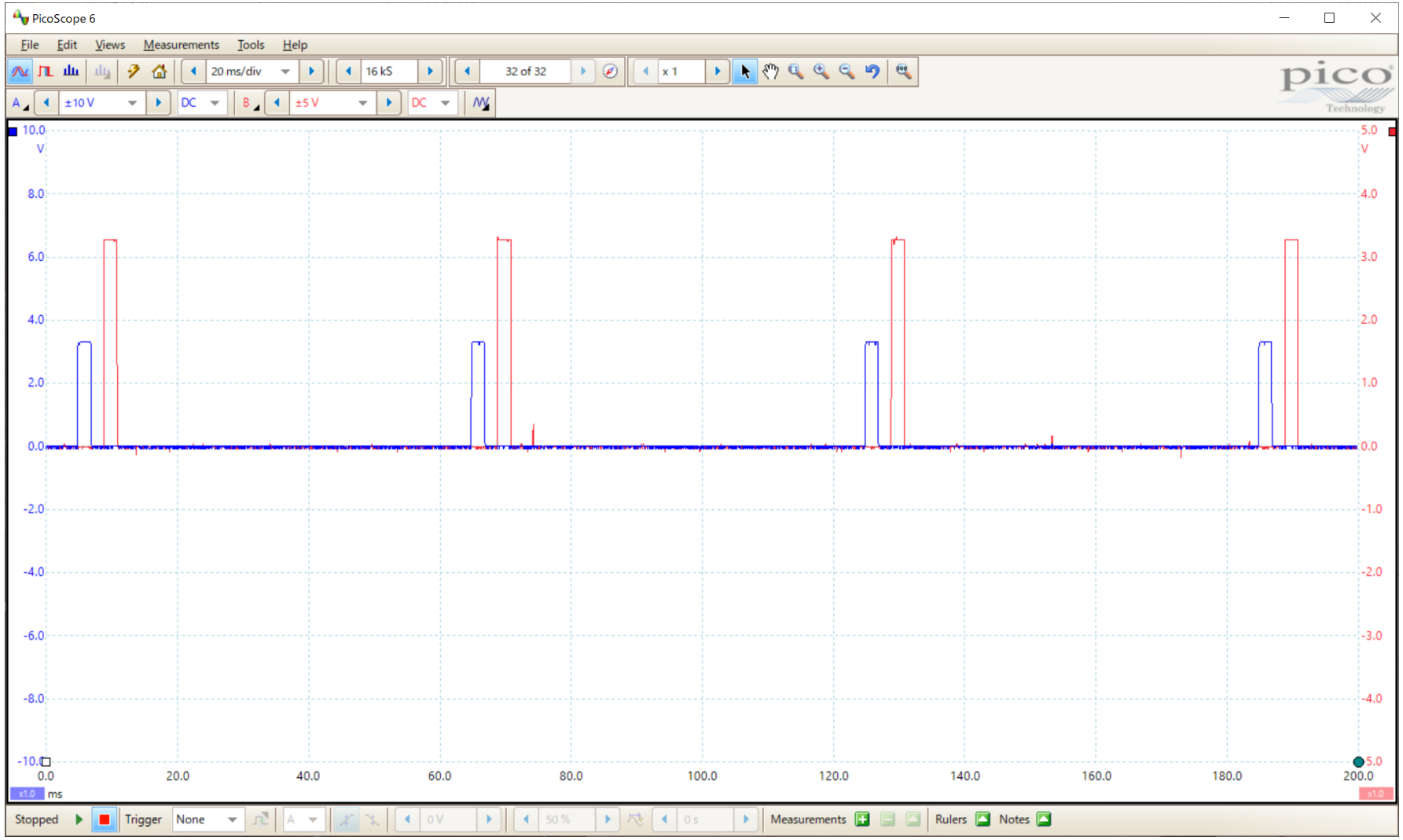
DCブラシレスモーター駆動

PWM キャリブレーション

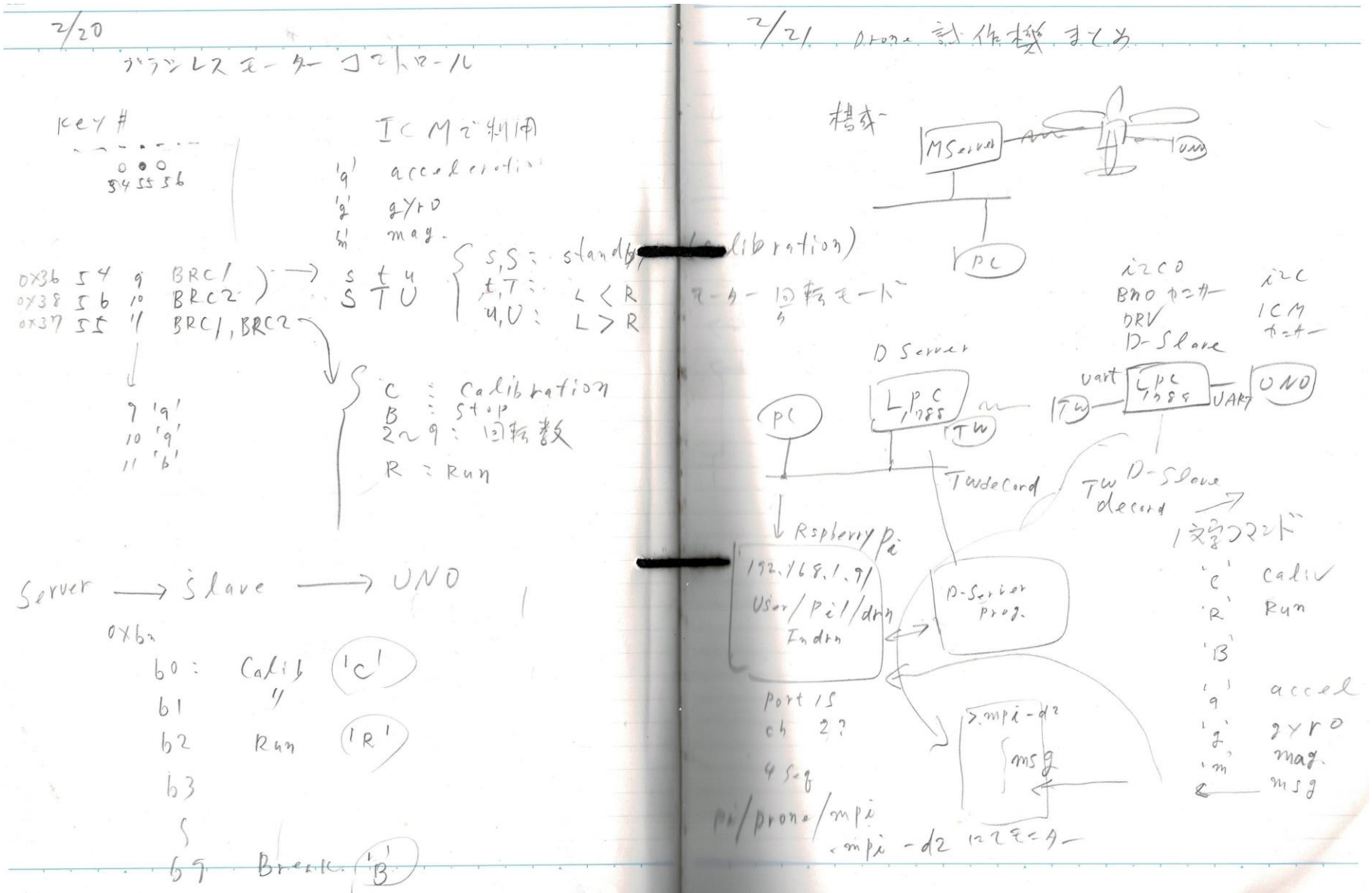


DCブラシレスモーター駆動

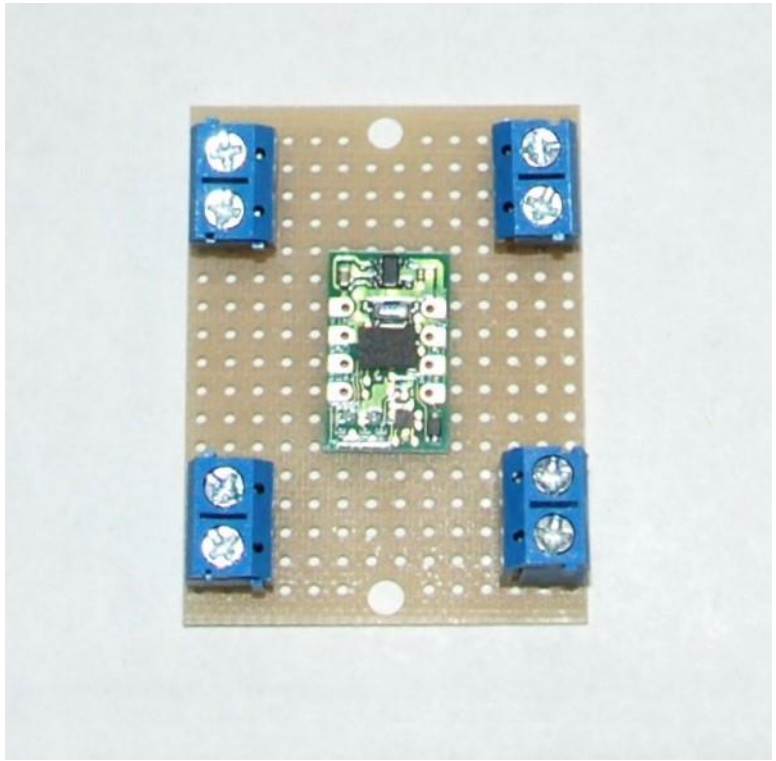
PWM Run

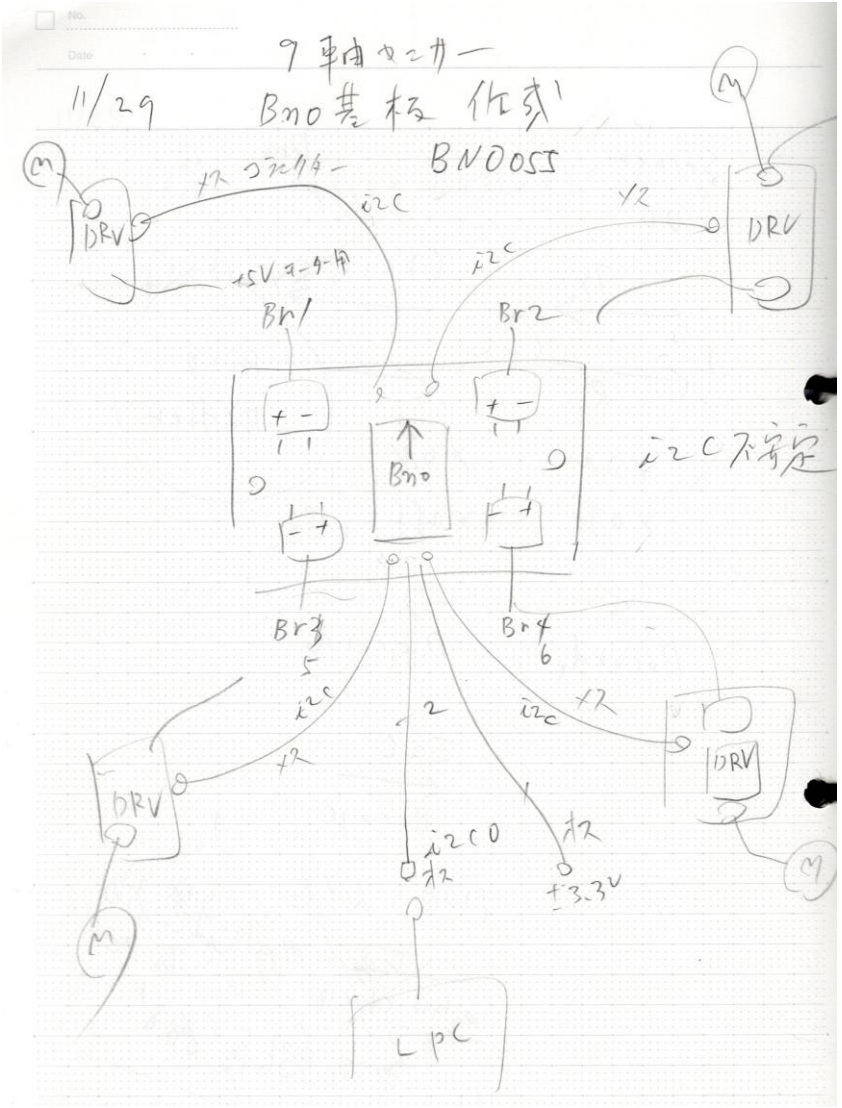


DC motor control メモ



9軸センサー BNO055基板作成





```
int  btf;    // Bno temperature ON/OFF flag
int  bcf;    // Bno Calib_Stat ON/OFF flag
int  bgf;    // Bno Gyroscope ON/OFF flag
int  baf;    // Bno Acceleration ON/OFF flag
int  bmf;    // Bno Magnetmerer ON/OFF flag
int  bhf;    // Bno Heading Roll Pitch ON/OFF flag
int  bqf;    // Bno Quaternion ON/OFF flag
int  blf;    // Bno Linear Acceleration ON/OFF flag
int  bvf;    // Bno graVity Vector ON/OFF flag
```

```
/***********************************************************************
 * Function Name: TMR0_IRQHandler
 * Parameters: none
 *
 * Return: none
 *
 * Description: Timer 0 interrupt handler
 *
 ***********************************************************************/
void TMR0_IRQHandler (void)
{
    //LED_ON();  // int check
    if(!(t_cnt % 2000)) Drone_BNO055();

    if(!(t_cnt % 2400)) {
        if(iaf) UPutch0('a');  // ICM-20948 acceleration UNO
        if(igf) UPutch0('g');  // ICM-20948 gyro UNO
        if(imf) UPutch0('m');  // ICM-20948 magnetometer UNO
    }

    t_cnt++;
    /*clear interrupt*/
    TOIR_bit.MR0INT = 1;
    //LED_OFF();
    /*Dummy read*/
    TOIR;
}
```

```
case  13 : // set Bno & Icm-UNO flags ON
          if(vd == 1) sprintf(wk,"T Bno Temperature on > %02x¥n¥n",c);
          if(vd == 2) sprintf(wk,"T Bno Calib_Stat on > %02x¥n¥n",c);
          if(vd == 3) sprintf(wk,"T Bno Gyro on > %02x¥n¥n",c);
          if(vd == 4) sprintf(wk,"T Bno Acceleration on > %02x¥n¥n",c);
          if(vd == 5) sprintf(wk,"T Bno Mag. on > %02x¥n¥n",c);
          if(vd == 6) sprintf(wk,"T Bno Heading on > %02x¥n¥n",c);
          if(vd == 7) sprintf(wk,"T Bno Quatererion on > %02x¥n¥n",c);
          if(vd == 8) sprintf(wk,"T Bno Linear Accel. on > %02x¥n¥n",c);
          if(vd == 9) sprintf(wk,"T Bno Gravity on > %02x¥n¥n",c);
          if(vd == 10) sprintf(wk,"T ICM Gravity on > %02x¥n¥n",c);
          if(vd == 11) sprintf(wk,"T ICM Acceleration on > %02x¥n¥n",c);
          if(vd == 12) sprintf(wk,"T ICM Mag on > %02x¥n¥n",c);
          UPuts1(wk);
          midiP15bd_drn(vd);
          break;
case  14 : // set Bno & Icm-UNO flags OFF
          //sprintf(wk,"T Bno & ICM off > %02x¥n¥n",c);
          if(vd == 1) sprintf(wk,"T Bno Temperature off > %02x¥n¥n",c);
          if(vd == 2) sprintf(wk,"T Bno Calib_Stat off > %02x¥n¥n",c);
          if(vd == 3) sprintf(wk,"T Bno Gyro off > %02x¥n¥n",c);
          if(vd == 4) sprintf(wk,"T Bno Acceleration off > %02x¥n¥n",c);
          if(vd == 5) sprintf(wk,"T Bno Mag. off > %02x¥n¥n",c);
          if(vd == 6) sprintf(wk,"T Bno Heading off > %02x¥n¥n",c);
          if(vd == 7) sprintf(wk,"T Bno Quatererion off > %02x¥n¥n",c);
          if(vd == 8) sprintf(wk,"T Bno Linear Accel. off > %02x¥n¥n",c)
          if(vd == 9) sprintf(wk,"T Bno Gravity off > %02x¥n¥n",c);
          if(vd == 10) sprintf(wk,"T ICM Gravity off > %02x¥n¥n",c);
          if(vd == 11) sprintf(wk,"T ICM Acceleration off > %02x¥n¥n",c)
          if(vd == 12) sprintf(wk,"T ICM Mag off > %02x¥n¥n",c);
          UPuts1(wk);
          midiP15be_drn(vd);
          break;
case  15 : // All Stop
  if(c == 0xf0) {
          sprintf(wk,"T Run All stop > %02x¥n¥n",c);
          UPuts1(wk);
              DrvBrA_stp();  // Br All stop
              MsgAll_stp();  // Bno & Icm Msg stop
          }
          break;
default  : break;
}
```

```
// set Bno & Icm-UNO flags O
int midiP15bd_drn(vd)
int  vd;
{
     if(vd == 1) btf = 1;
     if(vd == 2) bcf = 1;
     if(vd == 3) bgf = 1;
     if(vd == 4) baf = 1;
     if(vd == 5) bmf = 1;
     if(vd == 6) bhf = 1;
     if(vd == 7) bqf = 1;
     if(vd == 8) blf = 1;
     if(vd == 9) bvf = 1;

     if(vd == 10) igf = 1;
     if(vd == 11) iaf = 1;
     if(vd == 12) imf = 1;

     return(0);
}


// set Bno & Icm-UNO flags O
int midiP15be_drn(vd)
int  vd;
{
     if(vd == 1) btf = 0;
     if(vd == 2) bcf = 0;
     if(vd == 3) bgf = 0;
     if(vd == 4) baf = 0;
     if(vd == 5) bmf = 0;
     if(vd == 6) bhf = 0;
     if(vd == 7) bqf = 0;
     if(vd == 8) blf = 0;
     if(vd == 9) bvf = 0;

     if(vd == 10) igf = 0;
     if(vd == 11) iaf = 0;
     if(vd == 12) imf = 0;

     return(0);
}
```

```
extern int  baf;    // Bno Acceleration ON/OFF flag
extern int  bmf;    // Bno Magnetmerer ON/OFF flag
extern int  bhf;    // Bno Heading Roll Pitch ON/OFF flag
extern int  bqf;    // Bno Quaternion ON/OFF flag
extern int  blf;    // Bno Linear Acceleration ON/OFF flag
extern int  bvf;    // Bno graVity Vector ON/OFF flag

int Drone_BNO055()
{
    int  i,j;
    //unsigned char dAddr;
    unsigned char bAddr;
    unsigned char pMsg[4];

    //unsigned char bMsg[32];
    //bAddr = 0x50 >> 1;  // BNO055 test
    bAddr = 0x28;  // BNO055 0x28
    I2C_MasterWrite (bAddr, pMsg, 0);
    //dAddr = 0xc8 >> 1;  // DRV8830 c8

    // BNO055 ////
    // OPR_MODE Reg. 0x3d Def 0x10 Config Mode
    // Fusion Mode -> IMU , COMPASS , M4G , NDOF_FMC_OFF , NDOF
    // Test Use -> NDOF Absolute orientation

    for( i=0; i < 300000; i++); //delay
    pMsg[0] = 0x3d;       // Opreating Mode reg.
    //pMsg[1] = 0x08;     // Set IMU Mode
    //pMsg[1] = 0x09;     // Set Compass Mode
    //pMsg[1] = 0x0a;     // Set M4G Mode
    //pMsg[1] = 0x0b;     // Set NDOF_FMC_OFF mode
    pMsg[1] = 0x0c;     // Set NDOF Mode

    I2C_MasterWrite (bAddr, pMsg, 2);
    for( i=0; i < 300000; i++); //wait setting

    // ***********************************
    //  Read Fusion data
    for( j=0; j < 1; j++) {

        if(btf) Read_Temperature(bAddr);

        if(bcf) Read_Calib_Stat(bAddr);

        //  Gyroscope
        if(bgf) Read_Gyroscope_data(bAddr);

        //  Acceleration
        if(baf) Read_Acceleration_data(bAddr);

        //  Magetometer
        if(bmf) Read_Magnetometer_data(bAddr);

        //  EUL Heading Roll Pitch
        if(bhf) Read_Heading_data(bAddr);

        //  Quaternion
        if(bqf) Read_Quaternion_data(bAddr);

         //  Linear Acceleration
        if(blf) Read_Linear_Acceleration(bAddr);

         //  Gravity vector
        if(bvf) Read_Gravity_vector_data(bAddr);
    }
    // ***********************************

    // UPutch1(ESC); // ESC moniter fin.

    return(0);
}
```

ICM-20948 TCA9406基板

LPC1788にてテスト メモ

12/5

ICM 9軸カラー

TCA i2c電圧変換モジュール

3.3v ① 3.3v

②

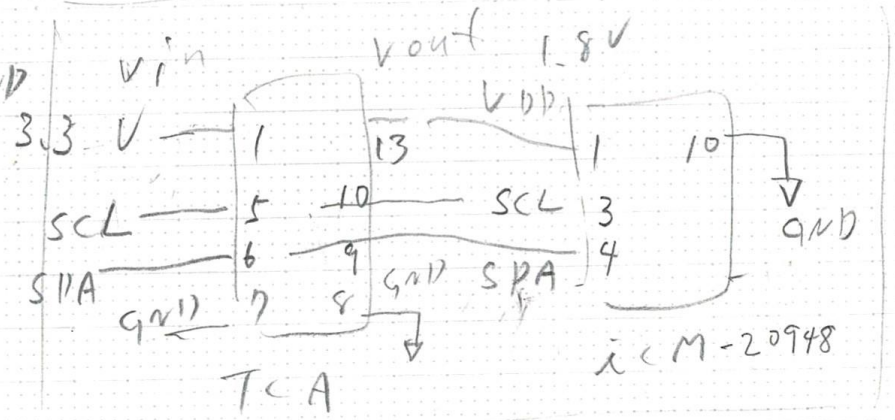
③ i2c0 OK Solo

④ i2c1 NG → UNOへ

SCL ⑤

SDA ⑥

GND ⑦ GND



LPC1788 ICM-20948 I2C test

```
//void i2c1_init(addr)
//unsigned char addr;
void i2c1_init()    // 2022.12.11 h.
{
    PCONP_bit.PCI2C1 = 1;    // enable I2C1 clock

    I2C1CONCLR = (1 << 6);  // 0x40

    //I2C0SCLH = 280; // i2c1 OK ? (ICM NG)
    //I2C0SCLL = 280; // i2c1 OK ? (ICM NG)

    //I2C1SCLH = 310; // i2c1   OK? (ICM NG)
    //I2C1SCLL = 310; // i2c1   OK? (ICM NG)

    I2C1SCLH = 310; // i2c1   OK? (ICM NG)
    I2C1SCLL = 311; // i2c1   OK? (ICM NG)

    *IOCON_P0_19 = 0x03;  // pn 85 i2c1 SDA
    *IOCON_P0_20 = 0x03;  // pn 83 i2c1 SCL

    // i2c1 enable
    //I2C0CONSET |= (1 << 6);  // 0x40 OK
    I2C1CONSET |= 0x40;  // I2EN

    I2C1CONCLR = (I2CON_STAC | I2CON_SIC | I2CON_AAC); // i2c clear flag

}
```

I2C0不安定なためICM-20948はUNOのI2C利用に変更

```
int Drone_ICM_i2c1()
{
    int  i,j;
    unsigned char iAddr;

    unsigned char pMsg[4];
    unsigned char iMsg[32];
    char msg[80];

    iAddr = 0x68 ;        // ICM20948 0b1101000
    pMsg[0] = 0x00;        // WHO_AM_I

    for(j=0;j<1;j++) {
    I2C1_MasterWrite (iAddr, pMsg, 0);
    for(i=0;i<3;i++) {
        for( i=0; i < 3000; i++); //delay
        I2C1_MasterRead (iAddr, iMsg, 1);
        sprintf(msg,"ICM-Who_AM_I read: x%02x¥r¥n",iMsg[0]);
        UPuts1(msg);
    }
    }

    //for( i=0; i < 3000; i++); //delay
    pMsg[0] = 0x06;        // PWR_MGMT_1
    pMsg[1] = 0x00;
    I2C1_MasterWrite (iAddr, pMsg, 2);
    sprintf(msg,"ICM-PWR_MGMT_1 write: 0x00¥r¥n");
    UPuts1(msg);

    //for( i=0; i < 3000; i++); //delay
    pMsg[0] = 0x0f;        // INT_PIN_CFG
    pMsg[1] = 0x02;
    I2C1_MasterWrite (iAddr, pMsg, 2);
    sprintf(msg,"ICM-INT_PIN_CFG (BYPASS_EN) write: 0x02¥r¥n");
    UPuts1(msg);

    // Mag
    iAddr = 0x0c ;        // ICM20948 0b0001100
    pMsg[0] = 0x00;        // read Co.ID

    I2C1_MasterWrite (iAddr, pMsg, 0);
    for(i=0;i<3;i++) {
        I2C1_MasterRead (iAddr, iMsg, 1);
        sprintf(msg,"ICM-j Co.ID read: x%02x¥r¥n",iMsg[0]);
        UPuts1(msg);
    }

    //for( i=0; i < 300000; i++); //delay
    for(j=0; j<3;j++) {
        iAddr = 0x68 ;        // ICM20948 0b1101000
        //for( i=0; i < 10000000; i++); //delay

        pMsg[0] = 0x2d;  // ACCEL data
        I2C1_MasterWrite (iAddr, pMsg, 1);

        for( i=0; i < 6; i++) iMsg[j] = 0x00; // clear iMsg
        I2C1_MasterRead (iAddr, iMsg, 6);
        sprintf(msg,"ICM-acceleration read: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
        iMsg[0],iMsg[1],iMsg[2],iMsg[3],iMsg[4],iMsg[5]);
        UPuts1(msg);

    /*
        for( i=0; i < 100; i++); //delay
        pMsg[0] = 0x33;  // GYRO data
        I2C1_MasterWrite (iAddr, pMsg, 1);

        for( i=0; i < 6; i++) iMsg[j] = 0x00; // clear iMsg
        I2C1_MasterRead (iAddr, iMsg, 6);
        sprintf(msg,"ICM-gyro read: x%02x x%02x x%02x x%02x x%02x x%02x¥r¥n",
            iMsg[0],iMsg[1],iMsg[2],iMsg[3],iMsg[4],iMsg[5]);
        UPuts1(msg);
    */
    }

    return(0);
}
```

UNO I2C ICM-20948 接続
（付属TCA9406 1.8V/3.3Vレギュレーター）

I2C SCL/SDA

I2Cレベル変換モジュール
TCA9406

9軸センサーモジュール
ICM-20948

DC 3.3V / GND

UART接続、LPC1788 UNO

UART0 – TX,RX

LPC1788

UNO

次ページLPC1788(UART0)　UNOソースコード

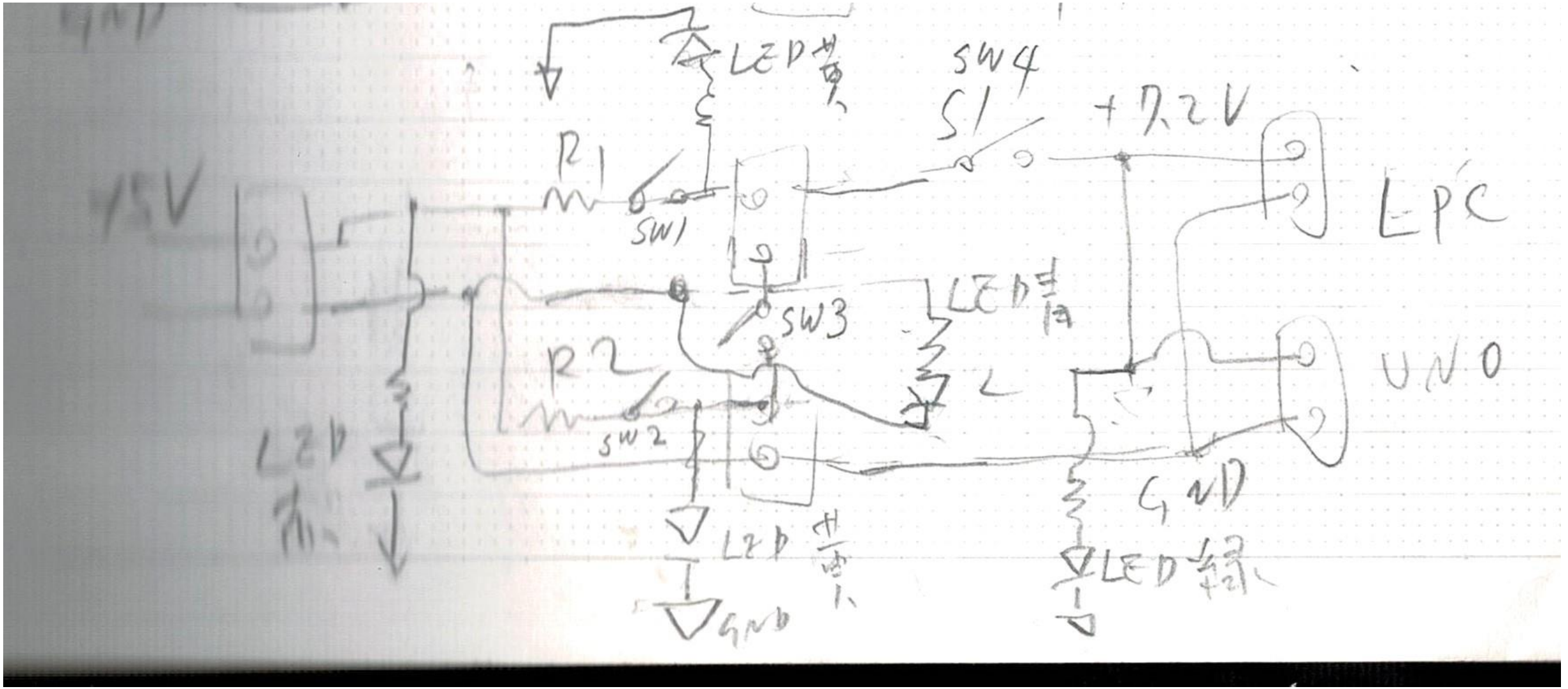
LPC1788 UART0 – UNO プログラム

```
/*****************************************************************************
 * Function Name: TMR0_IRQHandler
 * Parameters: none
 *
 * Return: none
 *
 * Description: Timer 0 interrupt handler
 *
 *****************************************************************************/
void TMR0_IRQHandler (void)
{
    //LED_ON();  // int check
    if(!(t_cnt % 2000)) Drone_BNO055();

    if(!(t_cnt % 2400)) {
        if(iaf) UPutch0('a');  // ICM-20948 acceleration UNO
        if(igf) UPutch0('g');  // ICM-20948 gyro UNO
        if(imf) UPutch0('m');  // ICM-20948 magnetometer UNO
    }

    t_cnt++;
    /*clear interrupt*/
    T0IR_bit.MR0INT = 1;
    //LED_OFF();
    /*Dummy read*/
    T0IR;
}
```

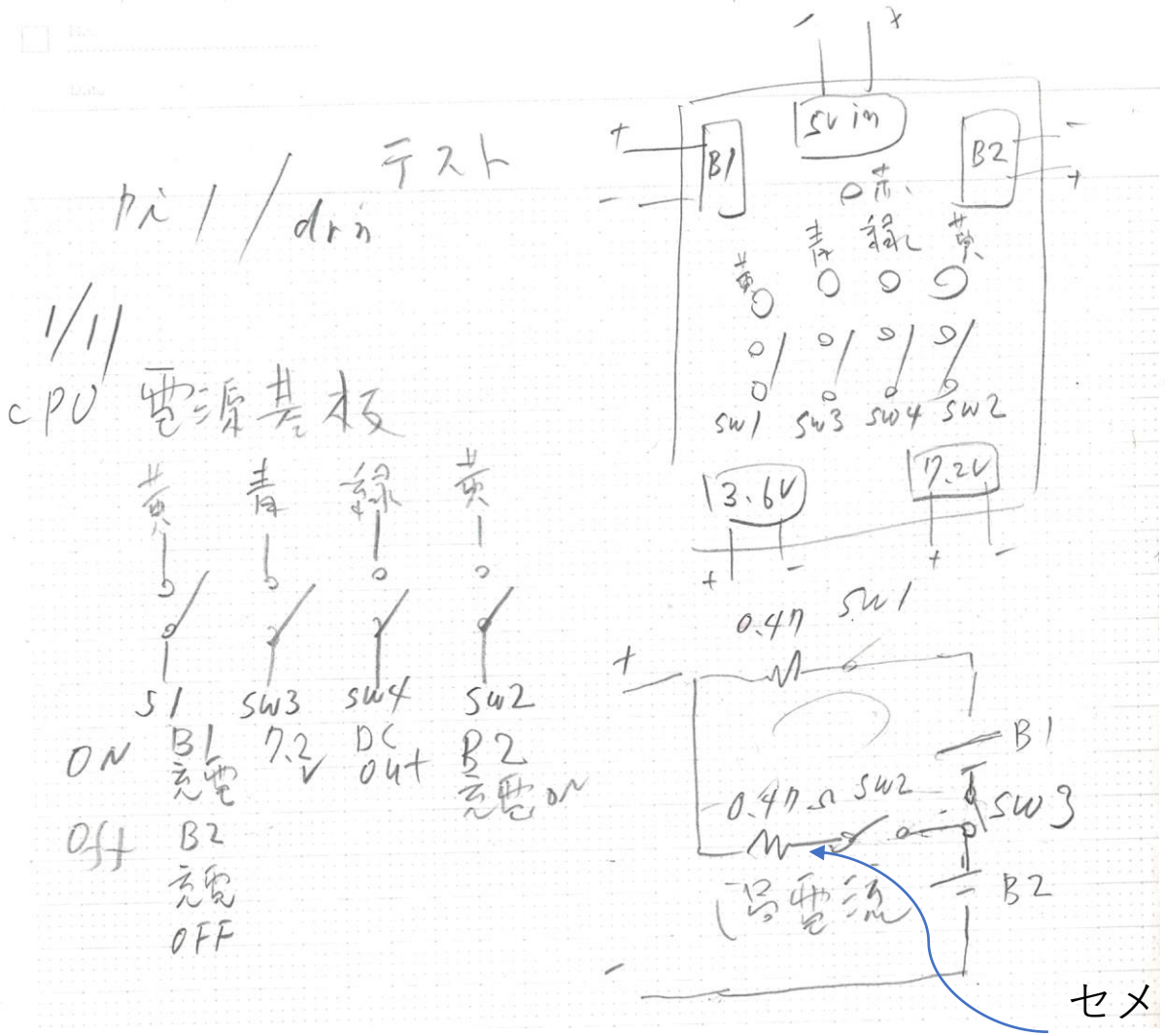
```
// BRC1
int midiP15b9_drn(vd)
int  vd;
{
    int  v,val;

    if(vd > 0 && vd < 10) {
        val = drval[vd];
        v = val << 2;
        v = v | 0x01;
        // DrvBr8_val(v);
    }

    switch(vd) {
        case  0 : // none
                  break;
        case  1 : UPutch0('s');   // BRC1 standby(Calibration)
                  break;
        default : UPutch0('t');   // Power BRC1 > BRC2 Run
                  break;
    }
    return(0);
}

// BRC2
int midiP15ba_drn(vd)
int  vd;
{
    int  v,val;

    if(vd > 0 && vd < 10) {
        val = drval[vd];
        v = val << 2;
        v = v | 0x01;
        // DrvBr8_val(v);
    }

    switch(vd) {
        case  0 : // none
                  break;
        case  1 : UPutch0('S');   // BRC2 Standby(Calibration)
                  break;
        default : UPutch0('T');   // Power BRC2 > BRC1 Run
                  break;
    }
    return(0);


}

// BRC1,BRC2
int midiP15bb_drn(vd)
int  vd;
{
    int  v,val;

    //vd = vd/12;  // vd/12.8
    if(vd > 0 && vd < 10) {
        val = drval[vd];
        v = val << 2;
        v = v | 0x01;
        // DrvBr8_val(v);
    }

    switch(vd) {
        case  0 : UPutch0('C');   // BRC1,2 Calibration
                  break;
        case  1 : UPutch0('C');   // BRC1,2 Calibration
                  break;
        case  2 : UPutch0('R');   // BRC1,2 val.2
                  break;
        case  3 : UPutch0('R');   // BRC1,2 val.3.
                  break;
        case  4 : UPutch0('R');   // BRC1,2 val.4
                  break;
        case  5 : UPutch0('R');   // BRC1,2 val.5
                  break;
        case  6 : UPutch0('R');   // BRC1,2 val.6
                  break;
        case  7 : UPutch0('R');   // BRC1,2 val.7
                  break;
        case  8 : UPutch0('R');   // BRC1,2 val.8
                  break;
        case  9 : UPutch0('B');   // BRC1,2 val.9 break;
                  break;
        default : // Power BRC1 = BRC2 Run
                  break;
    }
    return(0);
}
```

```
#include <Wire.h>

//int i;
int rvc;
int data = 0;
int data1;
int nonf = 0;
int addr = 0x68;  // ICM-20948
int addr2 = 0x0c; // AK09916 Mag address
int d1,d2,d3,d4,d5,d6;
char msg[30];
unsigned char md1,md2,md3;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(19200);
  Wire.begin();
  //Serial.write("////// Arduino setup() ///////¥r¥n");
}

void loop() {
  // put your main code here, to run repeatedly:
  // UART check
  rvc = 0;
  if(rvc=Serial.available() > 0) {
    data1 = Serial.read();
    //Serial.write("UART rvc=");
    //Serial.print(rvc,DEC);
    //Serial.write(" x");
    //Serial.print(data1,HEX);
    //Serial.write(" b");
    //Serial.print( data1,BIN);
    //Serial.write("¥r¥n");
    Serial.flush();
    //rvc = 0;
  }

  // i2c ICM-20948
  //addr = 0x68;
  //Serial.write("icm-20948 i2c Slave Address 0x68 ¥r¥n");
  Wire.beginTransmission(addr);
  Wire.write(0x00);
  Wire.endTransmission();

  Wire.requestFrom(addr, 1);
  if (Wire.available() >= 1) {
      data = Wire.read();
      //Serial.write("icm-Who_AM_I -> 0x");
      //Serial.print(data,HEX);
      //Serial.write(" ¥r¥n");
  }

  //for(i=0;i<30000;i++); //wait
  Wire.beginTransmission(addr);
  Wire.write(0x06);
  Wire.write(0x00);
  Wire.endTransmission();

  Wire.beginTransmission(addr);
  Wire.write(0x0f);
  Wire.write(0x02);
  Wire.endTransmission();

  //for(i=0;i<30000;i++); //wait

  if(data1 == 'a') {
    data1 = ' ';
    //addr = 0x68;
    Serial.write("info--> Acceleration¥r¥n");  // need i char
    Wire.beginTransmission(addr);
    Wire.write(0x2d);  // Accel data ,,
    Wire.endTransmission();
    //for(i=0;i<30000;i++); //wait
    Wire.requestFrom(addr, 6);
    if (Wire.available() >= 1) {
      d1 = Wire.read();
      d2 = Wire.read();
      d3 = Wire.read();
      d4 = Wire.read();
      d5 = Wire.read();
      d6 = Wire.read();
      Serial.write("Ar_ICM-acceleration read: x");
      Serial.print(d1,HEX);
      Serial.write(" x");
      Serial.print(d2,HEX);
      Serial.write(" x");
      Serial.print(d3,HEX);
      Serial.write(" x");
      Serial.print(d4,HEX);
      Serial.write(" x");
      Serial.print(d5,HEX);
      Serial.write(" x");
      Serial.print(d6,HEX);
      Serial.write(" ¥r¥n");
    }
  }

  if(data1 == 'g') {  // 'g'
    data1 = ' ';
    //addr = 0x68;
    Serial.write("info--> Gyro¥r¥n");
    Wire.beginTransmission(addr);
    Wire.write(0x33);  // Gyro data ,,
    Wire.endTransmission();
    //for(i=0;i<30000;i++); //wait
    Wire.requestFrom(addr, 6);
    if (Wire.available() >= 1) {
      d1 = Wire.read();
      d2 = Wire.read();
      d3 = Wire.read();
      d4 = Wire.read();
      d5 = Wire.read();
      d6 = Wire.read();
      Serial.write("Ar_ICM-gyro read: x");
      Serial.print(d1,HEX);
      Serial.write(" x");
      Serial.print(d2,HEX);
      Serial.write(" x");
      Serial.print(d3,HEX);
      Serial.write(" x");
      Serial.print(d4,HEX);
      Serial.write(" x");
      Serial.print(d5,HEX);
      Serial.write(" x");
      Serial.print(d6,HEX);
      Serial.write(" ¥r¥n");
    }
  }

  if(data1 == 'm') {
    data1 = ' ';
    // AK09916 Magnetic data read
    //for(i=0;i<30000;i++); //wait
    //addr2 = 0x0c;
    Serial.write("info--> Mag¥r¥n");
    Wire.beginTransmission(addr2);
    Wire.write(0x00);
    Wire.endTransmission();

    Wire.requestFrom(addr2, 1);
    if (Wire.available() >= 1) {
      data = Wire.read();
      //Serial.write("icm-Mag Co.ID -> 0x");
      //Serial.print(data,HEX);
      //Serial.write(" ¥r¥n");
    }

    Wire.beginTransmission(addr2);
    Wire.write(0x31);  // Control 2
    Wire.write(0x02);  // Ad start
    Wire.endTransmission();
    Wire.beginTransmission(addr2);
    Wire.write(0x11);
    Wire.endTransmission();

    Wire.requestFrom(addr2, 8);
    if (Wire.available() >= 1) {
      d1 = Wire.read();
      d2 = Wire.read();
      d3 = Wire.read();
      d4 = Wire.read();
      d5 = Wire.read();
      d6 = Wire.read();
      data = Wire.read();  // Dummy
      data = Wire.read();  // Status2
      Serial.write("Ar_ICM_magnetic read: x");
      Serial.print(d1,HEX);
      Serial.write(" x");
      Serial.print(d2,HEX);
      Serial.write(" x");
      Serial.print(d3,HEX);
      Serial.write(" x");
      Serial.print(d4,HEX);
      Serial.write(" x");
      Serial.print(d5,HEX);
      Serial.write(" x");
      Serial.print(d6,HEX);
      Serial.write(" ST2 x");
      Serial.print(data,HEX);
      Serial.write(" ¥r¥n");
      //Serial.write(" ¥r");
    }
  }
}
```

電源回路


LED赤
SW4
S1
+7.2V
R1
SW1
LPC
SW3
LED青
R2
SW2
UNO
LED
GND
LED緑
LED
GND

電源回路



※重要

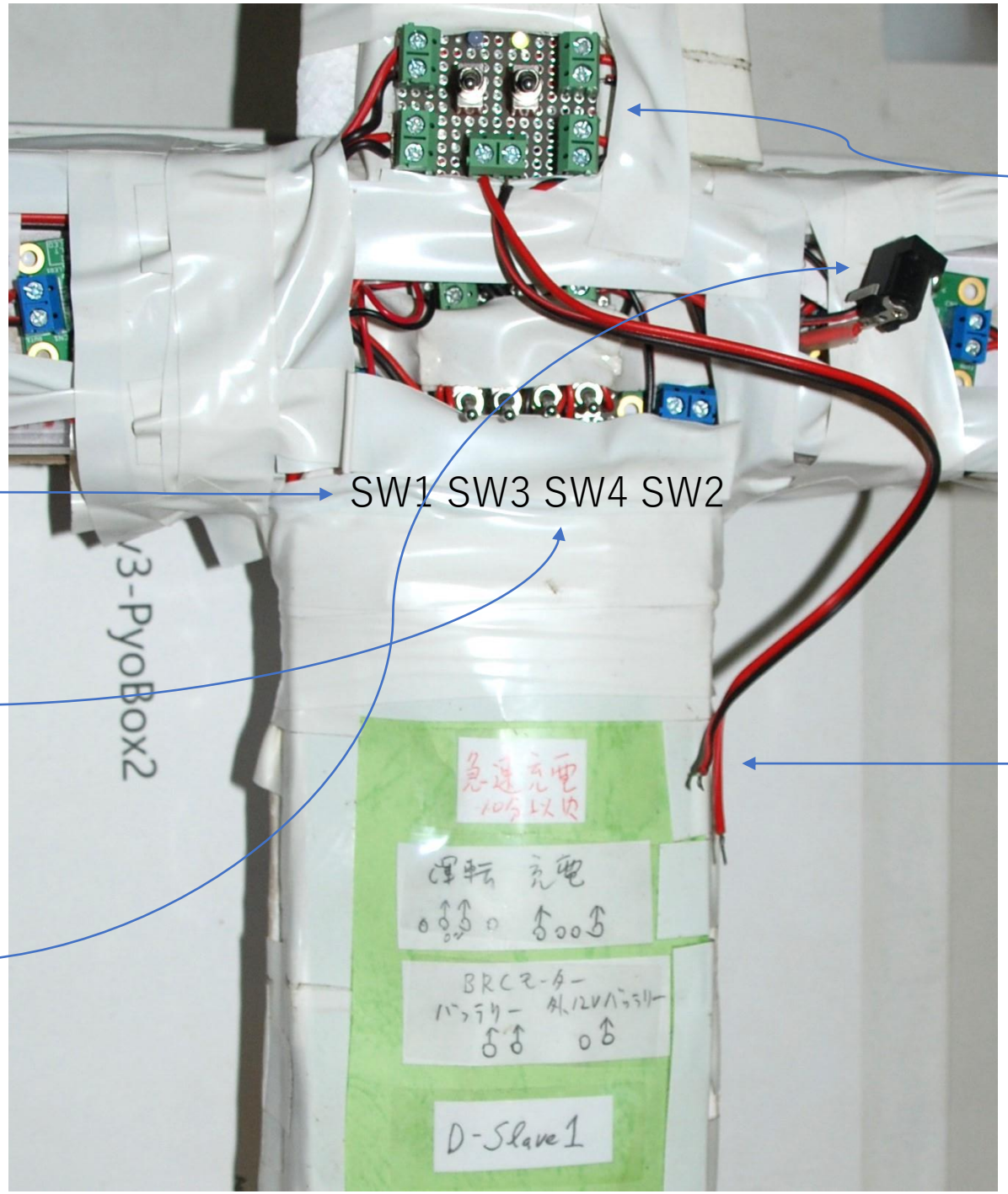
SW1	SW3	SW4	SW2		
ON	×	×	ON	OK	充電 7.2V out
×	ON	ON	×	OK	3.6V out
×	×	×	×	OK	
ON	ON	ON	ON	NG	過電流
ON	ON	×	×	NG	〃
ON	ON	ON	×	NG	〃
○×	ON	ON	ON	?	—

セメント抵抗
5W 1Ωが良い

運転
SW1 SW3 SW4 SW2
OFF ON ON OFF

充電
SW1 SW3 SW4 SW2
ON OFF OFF ON

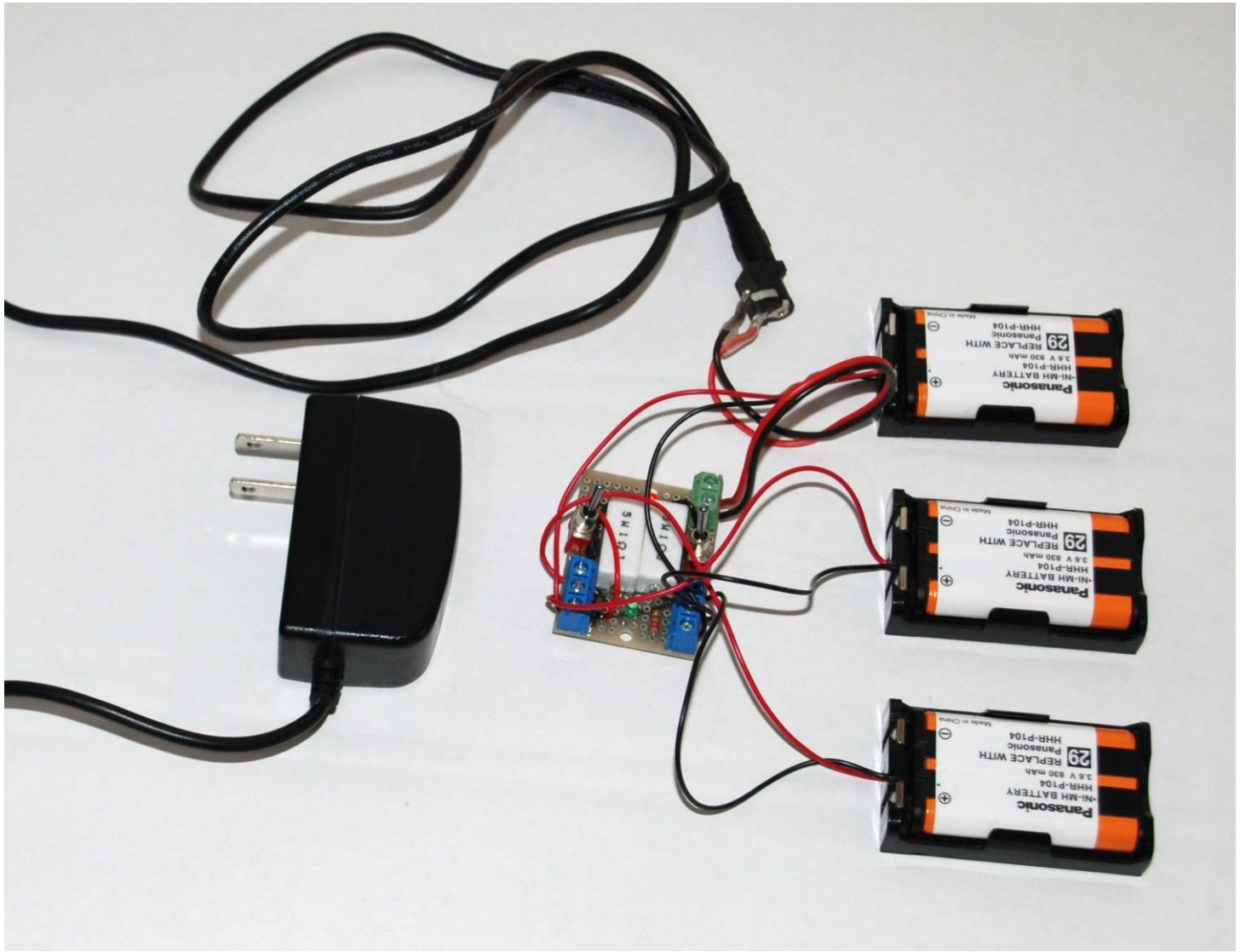
DC 5V 外部電源
（急速充電用）
重要
電池の発熱あり短時間のみ可(5分以内)



ブラシレスモーター
用スイッチ L,R

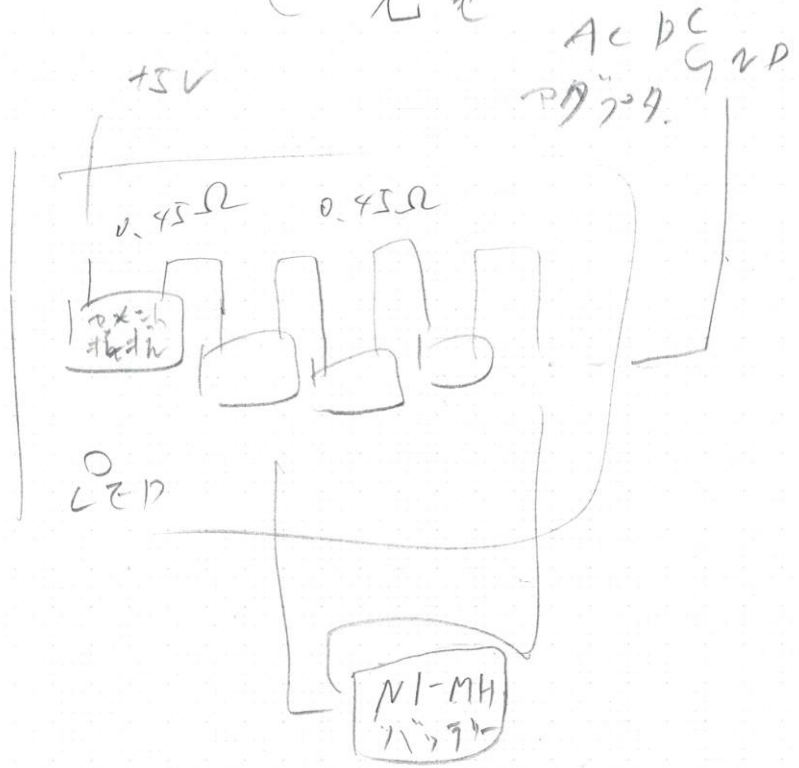
ブラシレスモーター
外部電源接続
DC 7.2V ~ 12V

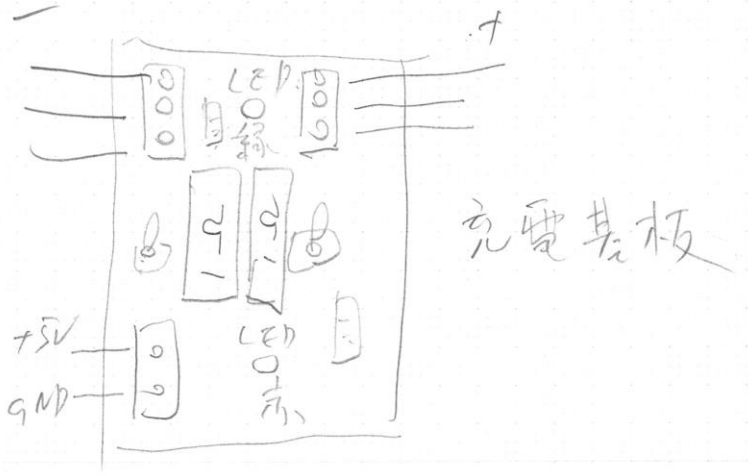
バッテリー充電


Panasonic
HHR-P104
REPLACE WITH
Panasonic
3.6 V 830 mAh
Made in China

ﾆｯｹﾙ水素

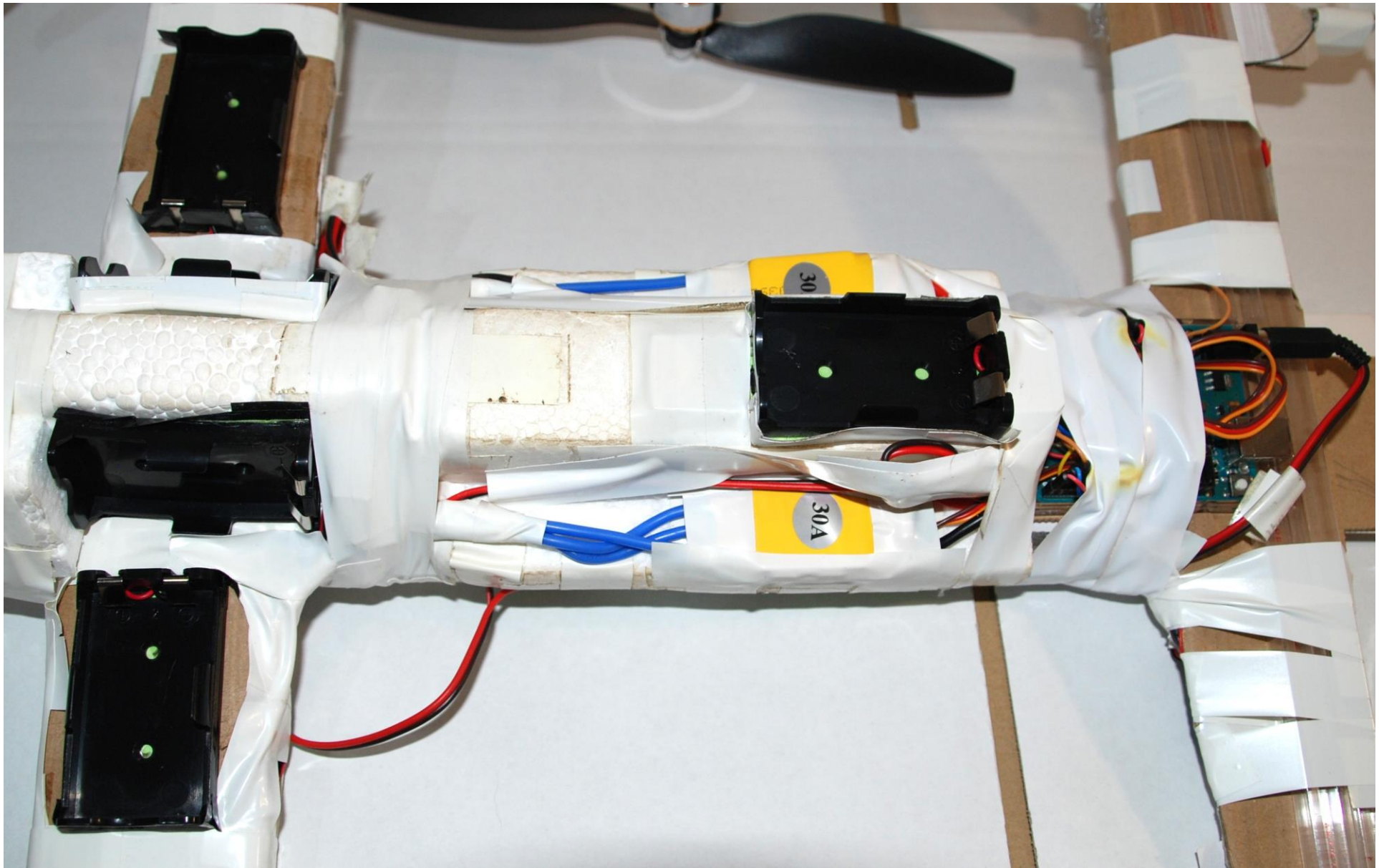
電池 充電 Test



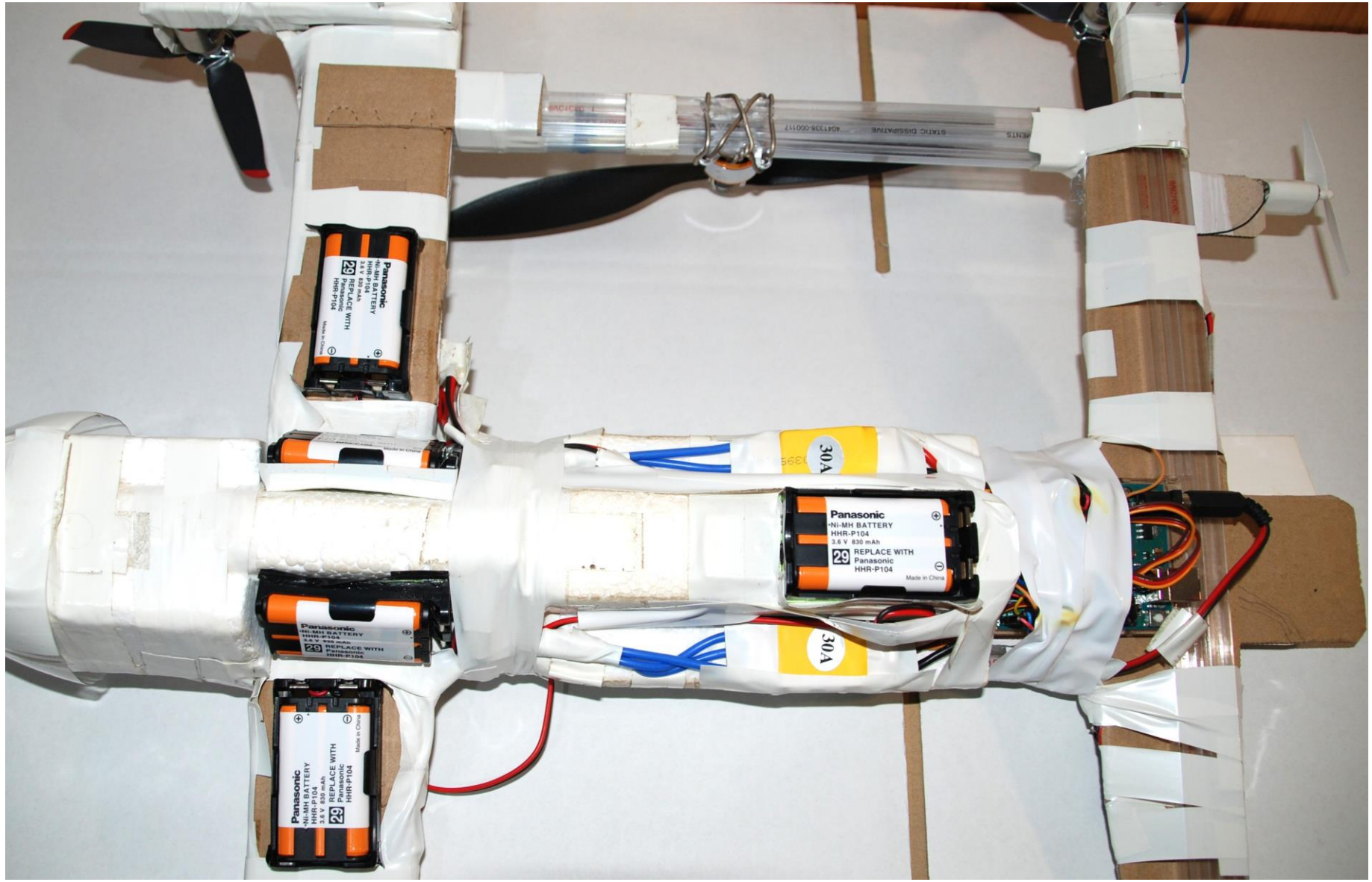


充電基板

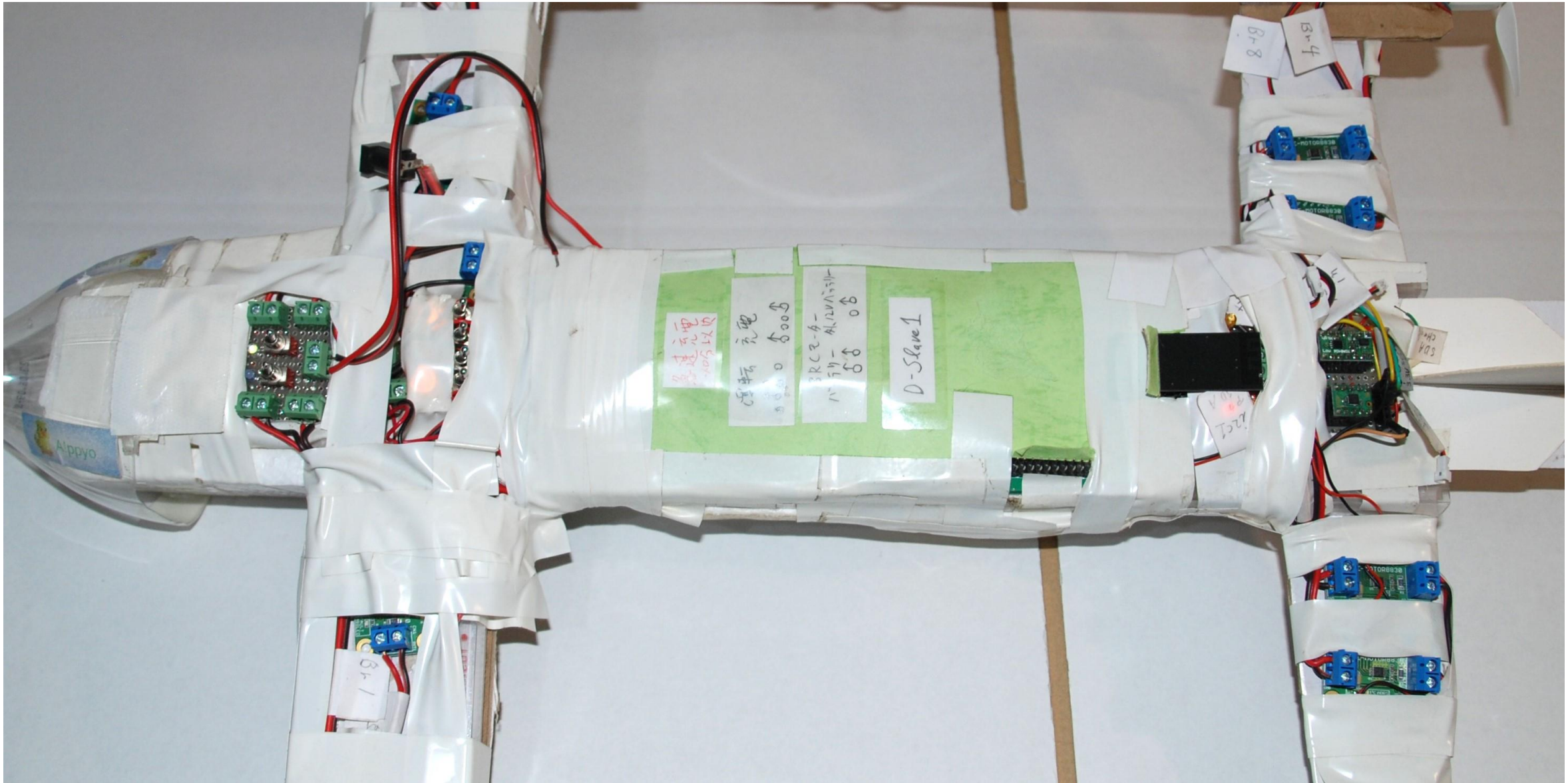
運転(バッテリー接続)



運転(バッテリー接続)


Panasonic
Ni-MH BATTERY
HHR-P104
3.6 V 830 mAh
REPLACE WITH
Panasonic
HHR-P104
Made in China
30A

動作手順



動作

運転
SW1 SW3 SW4 SW2
OFF ON ON OFF

ブラシレスモーター用スイッチ L,R ON

モーター回転チェック
Br1 Br2 Br3 Br4 Br5 Br6 Br7 Br8 順番に

ブラシレスモーターはキャリブレーション状態で待機

サーバーからのコントールデータ受信を待つ

Dr-Server 転送データ

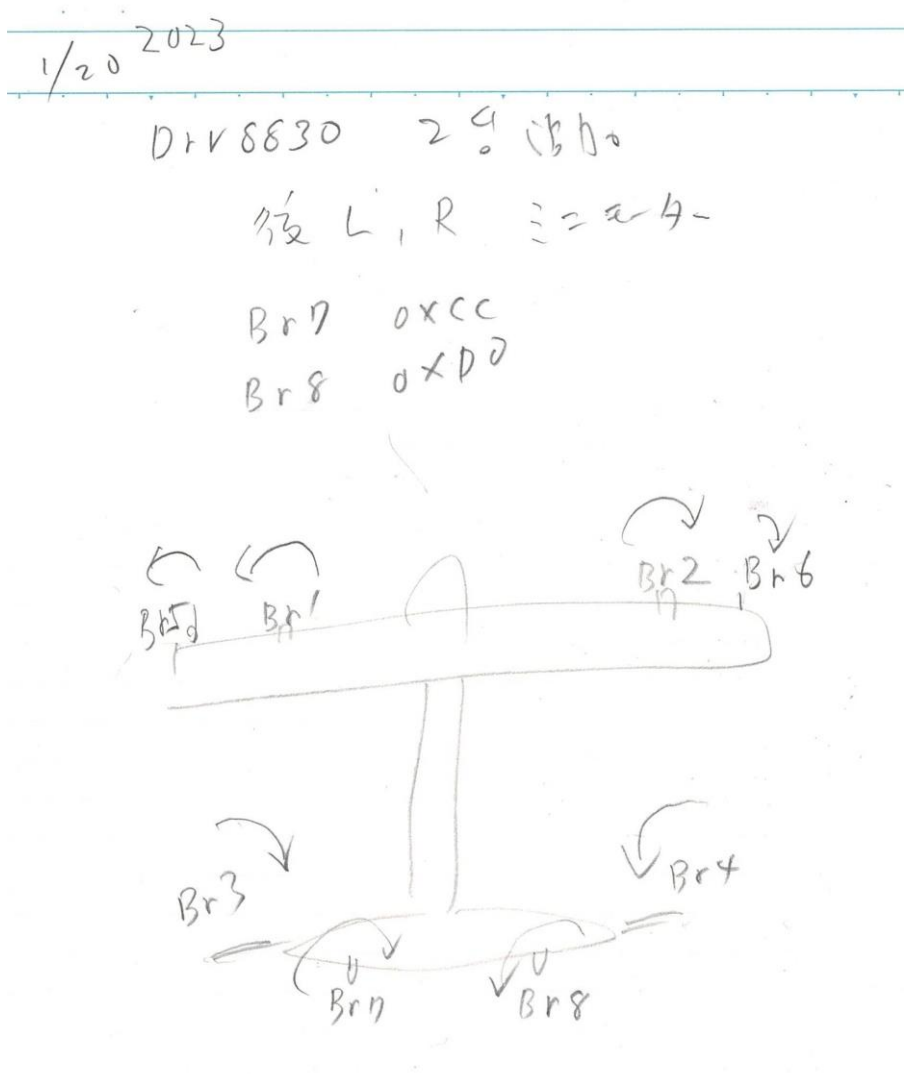
2 Byte数字データ作成

Drone1

Port15 MIDI Ch1 Note On -> 2Byteデータを送信

例

```
Note ON   0x90 0x31 0x2b
  //c1 = 0x90 c2 = 0x31 c3 = 0x2b;
  // brn : motor No.
  if(c2 == 0x31) brn = 1;
  if(c2 == 0x33) brn = 2;
  if(c2 == 0x3b) brn = 3;
  if(c2 == 0x3d) brn = 4;
  if(c2 == 0x2c) brn = 5;
  if(c2 == 0x2e) brn = 6;
  if(c2 == 0x40) brn = 7;
  if(c2 == 0x42) brn = 8;

  brv = 0; // motor voltage
  brv = (int)c3/12;
  bv = brn <<4;
  bv = bv | brv;
 sprintf(msg,"%02x",bv);
  UPutch0(msg[0]);  // Uart0 -> TWELITE
  UPutch0(msg[1]);
  UPutch0('¥r');
  UPutch0('¥n');
```



まとめ

プロペラ　大　x２　　モーター　A2212/13T 1000KV
プロペラ　中　x４　　モーター　RE-140RA Mabuchi motor
プロペラ　小　x４　　モーター　Crozepony

モーターRE-140RA プロペラ中、回転数が上がらない。
（揚力がたりない）
モーターCrozepony フロペラ小、回転数が上がらない。
（揚力がたりない）
モーターA2212/13T プロペラ大
（揚力はOK、バッテリーの選択が重要）

I2Cの動作が不安定

機体の重量を減らす必要がある。

距離センサーが必要

スペック

サイズ
　機首-後ろ590mm　　主翼720mm　後翼520mm
　高さ 70mm

重量
　バッテリーなし
　　1000g

　バッテリーあり
　　1200g

今後の計画

モーターの回転を速くすることで揚力を上げる。
モーターの数を増やして揚力を上げる。
衝突防止に距離センサーを付ける。
機体の重量を減らす。