

```

#include <TinyWireM.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/sleep.h>
#define ADDRESS 0x76 //BME280アドレスGND接続
#define ST7032i_ADDRESS 0b0111110 //ST7032iアドレス
#define Command 0b00000000 //Co:0/Rs:0
#define Address 0b01000000 //Co:0/Rs:1

//BME280データ格納用
unsigned long int hum_raw, temp_raw, pres_raw;
signed long int t_fine;
uint16_t temp, pres, hum;
//補償係数
uint16_t dig_T1, dig_P1;
int16_t dig_T2, dig_T3, dig_P2, dig_P3, dig_P4, dig_P5, dig_P6,
        dig_P7, dig_P8, dig_P9, dig_H2, dig_H4, dig_H5;
int8_t dig_H1, dig_H3, dig_H6;
//画面切り替えスイッチ用
bool x=0; //押下フラグ
uint8_t y,z; //チャタリング防止用
//タイマ割り込みフラグ
volatile bool k=0;
//タイマ割り込み用カウンタ
volatile uint8_t tcnt;
//スリープまでの待機カウント(ボタンを押されることでリセット)
volatile uint8_t scnt;
//BME280モード
uint8_t mode00 =0b00; //00:スリープ
uint8_t mode01 =0b01; //01:強制測定
uint8_t mode11 =0b11; //11:通常測定

//タイマ割り込み関数フラグ切り替えのみ
ISR(TIM0_COMPA_vect){
    if(tcnt>=20){ //約50ms × tcnt
        k = !k;
        tcnt=0;
        scnt++;
    }
    tcnt++;
    PORTB &= ~0b00010000;
}

void readCompensation_factor()
{
    uint8_t data[32], i = 0;
    TinyWireM.beginTransmission(ADDRESS);
    TinyWireM.send(0x88);
    TinyWireM.endTransmission();
    TinyWireM.requestFrom(ADDRESS, 25);
    while (TinyWireM.available()) {
        data[i] = TinyWireM.read();
        i++;
    }

    TinyWireM.beginTransmission(ADDRESS);
    TinyWireM.send(0xE1);
    TinyWireM.endTransmission();
    TinyWireM.requestFrom(ADDRESS, 7);
    while (TinyWireM.available()) {
        data[i] = TinyWireM.read();
        i++;
    }
    dig_T1 = (data[1] << 8) | data[0];
    dig_T2 = (data[3] << 8) | data[2];
    dig_T3 = (data[5] << 8) | data[4];
    dig_P1 = (data[7] << 8) | data[6];
    dig_P2 = (data[9] << 8) | data[8];
    dig_P3 = (data[11] << 8) | data[10];
    dig_P4 = (data[13] << 8) | data[12];
    dig_P5 = (data[15] << 8) | data[14];
    dig_P6 = (data[17] << 8) | data[16];
    dig_P7 = (data[19] << 8) | data[18];
    dig_P8 = (data[21] << 8) | data[20];
    dig_P9 = (data[23] << 8) | data[22];
    dig_H1 = data[24];
    dig_H2 = (data[26] << 8) | data[25];
    dig_H3 = data[27];
    dig_H4 = (data[28] << 4) | ((0x0F & data[29]));
    dig_H5 = (data[30] << 4) | ((data[29] >> 4) & 0x0F);
    dig_H6 = data[31];
}

```

```

void sendBME280(uint8_t reg_address, uint8_t data)
{
    TinyWireM.beginTransmission(ADDRESS);
    TinyWireM.send(reg_address);
    TinyWireM.send(data);
    TinyWireM.endTransmission();
}

void readBME280()
{
    int i = 0;
    uint32_t data[8];
    TinyWireM.beginTransmission(ADDRESS);
    TinyWireM.send(0xF7);
    TinyWireM.endTransmission();
    TinyWireM.requestFrom(ADDRESS, 8);
    while (TinyWireM.available()) {
        data[i] = TinyWireM.read();
        i++;
    }
    pres_raw = (data[0] << 12) | (data[1] << 4) | (data[2] >> 4);
    temp_raw = (data[3] << 12) | (data[4] << 4) | (data[5] >> 4);
    hum_raw = (data[6] << 8) | data[7];
}

//分解能:0.01°C. 返り値は小数点2桁の整数で返されるので100で割っている。
signed long int BME280_compensate_T(signed long int adc_T)
{
    signed long int var1, var2, T;
    double temp;
    var1 = (((adc_T >> 3) - ((signed long int)dig_T1 << 1))) * ((signed long int)dig_T2) >> 11;
    var2 = (((((adc_T >> 4) - ((signed long int)dig_T1)) * ((adc_T >> 4) - ((signed long int)dig_T1))) >> 12) * ((signed long int)dig_T3)) >
    t_fine = var1 + var2;
    T = ((t_fine * 5 + 128) >> 8);
    return T;
}

//出力値96386.2Pa = 963.862hPaに相当します。。
unsigned long int BME280_compensate_P(signed long int adc_P)
{
    signed long int var1, var2;
    unsigned long int P;
    var1 = (((signed long int)t_fine >> 1) - (signed long int)64000;
    var2 = (((var1 >> 2) * (var1 >> 2)) >> 11) * ((signed long int)dig_P6);
    var2 = var2 + ((var1 * ((signed long int)dig_P5)) << 1);
    var2 = (var2 >> 2) + (((signed long int)dig_P4) << 16);
    var1 = (((dig_P3 * ((var1 >> 2) * (var1 >> 2)) >> 13)) >> 3) + (((signed long int)dig_P2) * var1) >> 1)) >> 18;
    var1 = (((32768 + var1)) * ((signed long int)dig_P1)) >> 15);
    if (var1 == 0)
    {
        return 0;
    }
    P = (((unsigned long int)(((signed long int)1048576) - adc_P) - (var2 >> 12))) * 3125;
    if (P < 0x80000000)
    {
        P = (P << 1) / ((unsigned long int) var1);
    }
    else
    {
        P = (P / (unsigned long int)var1) * 2;
    }
    var1 = (((signed long int)dig_P9) * ((signed long int)(((P >> 3) * (P >> 3)) >> 13))) >> 12;
    var2 = (((signed long int)(P >> 2)) * ((signed long int)dig_P8)) >> 13;
    P = (unsigned long int)((signed long int)P + ((var1 + var2 + dig_P7) >> 4));
    return P;
}

//出力値4744=47.44%
unsigned long int BME280_compensate_H(signed long int adc_H)
{
    signed long int v_x1;

    v_x1 = (t_fine - ((signed long int)76800));
    v_x1 = ((((((adc_H << 14) - (((signed long int)dig_H4) << 20) - (((signed long int)dig_H5) * v_x1)) +
        (((signed long int)16384)) >> 15) * ((((((v_x1 * ((signed long int)dig_H6)) >> 10) *
        ((v_x1 * ((signed long int)dig_H3)) >> 11) + ((signed long int) 32768))) >> 10) + (( signed long int)2097152)) *
        ((signed long int) dig_H2) + 8192) >> 14));
    v_x1 = (v_x1 - (((((v_x1 >> 15) * (v_x1 >> 15)) >> 7) * ((signed long int)dig_H1)) >> 4));
    v_x1 = (v_x1 < 0 ? 0 : v_x1);
    v_x1 = (v_x1 > 419430400 ? 419430400 : v_x1);
    return (unsigned long int)(v_x1 >> 12);
}

```

```

void initBME280(uint8_t mode){
    // BME280 initial
    uint8_t osrs_t = 0b001;    //温度オーバーサンプリング x 1
    uint8_t osrs_p = 0b001;    //気圧オーバーサンプリング x 1
    uint8_t osrs_h = 0b001;    //湿度オーバーサンプリング x 1
    uint8_t t_sb = 0b000;    //待機時間 0.5ms
    uint8_t filter = 0b000;    //フィルタオフ
    uint8_t spi3w_en = 1;    //4線SPI

    uint8_t ctrl_meas_reg = (osrs_t << 5) | (osrs_p << 2) | mode;
    uint8_t ctrl_meas_reg_s = (osrs_t << 5) | (osrs_p << 2) | mode;
    uint8_t config_reg = (t_sb << 5) | (filter << 2) | spi3w_en;
    uint8_t ctrl_hum_reg = osrs_h;

    sendBME280(0xF2, ctrl_hum_reg);
    sendBME280(0xF4, ctrl_meas_reg);
    sendBME280(0xF5, config_reg);
}

void commWrite(byte cData){
    TinyWireM.beginTransmission(ST7032i_ADDRESS);
    TinyWireM.send(Command);
    TinyWireM.send(cData);
    TinyWireM.endTransmission();
}

void dataWrite(byte dData){
    TinyWireM.beginTransmission(ST7032i_ADDRESS);
    TinyWireM.send(Address);
    TinyWireM.send(dData);
    TinyWireM.endTransmission();
}

void dispInit(){
    commWrite(0x38); //Function set:8bit-bus,2line-display
    commWrite(0x39); //external table use
    commWrite(0x14); //VDD3V:183Hz/VDD5V:192Hz
    commWrite(0x73); //コントラストC3,C2,C1,C0
    commWrite(0x52); //ICON-off/内部昇圧回路-off/コントラストC5,C4
    commWrite(0x6C); //内部フォロワ回路off,V0の増幅比
    commWrite(0x38); //Function set:8bit-bus,2line-display
    commWrite(0x0C); //display-on/cursor-off/画面反転-off
    commWrite(0x01);
}

void lcdTemp(uint16_t ltemp){

    commWrite(0b10000000);
    dataWrite(0b10110000|0b0101); // オ
    dataWrite(0b11010000|0b1101); // ン
    dataWrite(0b11000000|0b0100); // ト
    dataWrite(0b11010000|0b1110); // °

    commWrite(0b11000000);
    dataWrite(0b00110000|(ltemp/1000));
    dataWrite(0b00110000|(ltemp%1000)/100);

    commWrite(0b11000010);
    dataWrite(0b00100000|0b1110); // 点

    commWrite(0b11000011);
    dataWrite(0b00110000|(ltemp%1000)%100/10);
    dataWrite(0b00110000|(ltemp%1000)%100%10/1);

    commWrite(0b11000101);
    dataWrite(0b11010000|0b1111); // °
    dataWrite(0b01000000|0b0011); // C
    dataWrite(0b10100000|0b0000); // ""
}

```

```

void lcdHumi(uint16_t lhum){

commWrite(0b10000000);
dataWrite(0b10110000|0b1100);// シ
dataWrite(0b11000000|0b0010);// ツ
dataWrite(0b11000000|0b0100);// ト
dataWrite(0b11010000|0b1110);// *

commWrite(0b11000000);
dataWrite(0b00110000|hum/10000);
dataWrite(0b00110000|hum%10000/1000);

commWrite(0b11000010);
dataWrite(0b00100000|0b1110);// 点

commWrite(0b11000011);
dataWrite(0b00110000|(hum%10000%1000)/100);
dataWrite(0b00110000|(hum%10000%1000)%100/10);

commWrite(0b11000101);
dataWrite(0b00100000|0b0101);// %
dataWrite(0b10100000|0b0000);// ""
dataWrite(0b10100000|0b0000);// ""
}

void lcdPres(uint16_t lpres){

commWrite(0b10000000);
dataWrite(0b10110000|0b0111);// キ
dataWrite(0b10110000|0b0001);// ア
dataWrite(0b11000000|0b0010);// ツ
dataWrite(0b10100000|0b0000);// ""

commWrite(0b11000000);
dataWrite(0b00110000|(lpres/1000));
dataWrite(0b00110000|(lpres%1000)/100);
dataWrite(0b00110000|(lpres%1000)%100/10);
dataWrite(0b00110000|(lpres%1000)%100%10/1);

commWrite(0b11000100);
dataWrite(0b01100000|0b1000);// h
dataWrite(0b011010000|0b0000);// P
dataWrite(0b01100000|0b0001);// a
}

void setup() {
TinyWireM.begin();//マスター設定
displInit();
readCompensation_factor();

DDRB &= ~0b00001000;//PB3インプット
DDRB |= 0b00010000;//PB4アウトプット
PORTB |= 0b00001000;//PCINT3ブルアップ

//Timer0設定@内部クロック4MHz
TCCR0A = 0b00000010;//CTC.比較出力切断
TCCR0B = 0b00000101;//1024分周
OCR0A = 195;//49.92ms
TIMSK |= 0b00010000;//比較A割り込み
}

void loop() {

//タイマ割り込みにより強制測定
while(k){
initBME280(mode01);//強制測定モード
readBME280();
temp = BME280_compensate_T(temp_raw);
hum = BME280_compensate_H(hum_raw);
pres = BME280_compensate_P(pres_raw)/100;
k=!k;
PORTB |= 0b00010000;
}

x=~(PINB>>3)&1;//PB3状態抽出
while(x){
static uint32_t cnt;//チャタリング防止カウンタ
cnt++;
if(cnt>=100000){
y=1;
x=0;
cnt=0;
scent=0;
}
}
}

```

```

//モーメンタリスイッチ押下検知
if(y){
    z++;
    y=0;
}
//押下回数による機能切り替え
switch(z){
    case 0:
        lcdTemp(temp);
        break;
    case 1:
        lcdHumi(hum);
        break;
    case 2:
        lcdPres(pres);
        break;
    case 3:
        z=0;
        break;
}

//スリープモード条件
uint8_t r=0;
while(scnt>=30){
    switch (r){
        case 0:
            commWrite(0x01);
            r=1;
            break;
        case 1:
            r=2;
            commWrite(0b110000111);
            dataWrite(0b01110000|0b0011);//s
            dataWrite(0b01100000|0b1100);//l
            dataWrite(0b01100000|0b0101);//e
            dataWrite(0b01100000|0b0101);//e
            dataWrite(0b01110000|0b0000);//p
            dataWrite(0b00100000|0b1110);//.
            dataWrite(0b00100000|0b1110);//.
            dataWrite(0b00100000|0b1110);//.
            break;
        case 2:
            break;
    }

    if(scnt>=32){
        commWrite(0x08);//display-off/cursor-off/画面反転-off
        commWrite(0x01);
        PORTB = 0;
        set_sleep_mode(SLEEP_MODE_PWR_DOWN);
        sleep_mode();//スリープ実行(復帰はリセット)
    }
}
}
}

```