Olsonet Communications

Pawel Gburzynski

BMP085 pressure/temp

driver

March 5, 2014

Introduction

BMP085 is an air pressure/temperature combination sensor manufactured by Bosch Sensortec. Documentation available from the manufacturer (data sheet + specification) covers all the technical intricacies. That documentation is not required to understand how to use the driver and what you can expect from the sensor.

The sensor replaces SCP1000 in the “white” version of CHRONOS.

Summary of the sensor's functionality

The sensor is quite simple in terms of its perceived functionality. It generates no events providing only on-demand readouts of air pressure and temperature. The temperature is required to correctly transform the pressure reading into a proper (meaningful) pressure. The transformation formula is rather complicated and involves (in addition to the temperature reading) eleven 16-bit values stored in the sensor's EPROM. The transformation is not automatic, i.e., it has to be carried out in software, either by the driver or by the user.

The only parameter of the internal sample collection process is the so-called *oversampling* which can be used to improve the accuracy of pressure measurement while increasing the amount of time needed to collect a sample (and thus increasing the amount of power needed for the reading). This parameter can take values from 0 to 3. The improved accuracy results from averaging multiple “takes”. The number of takes corresponding to the four oversampling values is 1, 2, 4, 8, respectively, and the corresponding amount of time needed to complete the pressure reading is 4.5, 7.5, 13.5, and 25.5 milliseconds. The operation *read_sensor* sleeps the thread until the values are available, so the state argument to the operation is meaningful.

The driver can be compiled into two versions, depending on the whether the constant BMP085_AUTO_CALIBRATE is defined or not (it is defined by default for CHRONOS_WHITE). If the constant is not defined, the temperature and pressure readings returned by the sensor are *raw* (i.e., uncalibrated) and the user has to do the calibration externally. Note that by “calibration” we mean in fact “transformation”, i.e., the “uncalibrated” values do not even approach anything resembling temperature or pressure readings. To be able to carry out the transformation, the user needs the 11 coefficients which can be obtained by reading an additional dummy sensor made available by the driver. Thus, in this mode, the sensor is visible as two logical sensors: one returning the raw temperature/pressure readings (which come together as a pair of 16-bit values), the other returning the values of calibration coefficients (as an array of 16-bit words). Those coefficients are constants (for a given specimen of the sensor) and there is no need to read them more than once.

When BMP085_AUTO_CALIBRATE is defined, the driver performs the requisite transformation internally before presenting the values via *read_sensor*. The primary reason behind the two modes is that the transformation appeared to me rather costly, especially in terms of code (a rather lengthy expression involving multiplication and division of long numbers)[1] with the flavor of an action that would be best performed by the OSS. However, as being able to read true pressure (and maybe even altitude) directly on CHRONOS is rather important, the default for the CHRONOS_WHITE board is now the auto-calibrated variant.

The interface

[1] The calibration contributes over 1KB of code.

For the (default) auto-calibrated variant, the value returned by the sensor (the last argument of *read_sensor*) is described by this structure:

```
typedef struct {
    lword press;
    sint temp;
} bmp085_data_t;
```

defined in *bmp085.h*. The *press* attribute returns the pressure in Pascals while temp returns the temperature in degrees centigrade times 10, i.e., in tenths of a degree.

In the raw variant, the corresponding structure looks like this:

```
typedef struct {
    word press;
    sint temp;
} bmp085_data_t;
```

and the two attributes are raw values that have to be further processed. The requisite coefficients can be read from a second sensor (via the same *read_sensor* operation) into this structure:

```
typedef struct {
    sint ac1;
    sint ac2;
    sint ac3;
    word ac4;
    word ac5;
    word ac6;
    sint b1;
    sint b2;
    sint mb;
    sint mc;
    sint md;
} bmp_085_calibration_data_t;
```

The transformation procedure (derived from the datasheet) is as follows (*t* and *p* are the raw temperature and pressure readings, *cd* is the table of coefficients, *temp* and *press* are the target transformed values):

```
lint A, B, C, D;
…
A = (((lint)t - (lint)cd.ac6) * (lint)cd.ac5) / 32768;
A += ((lint)cd.mc * 2048) / (cd_tmp + cd.md);
temp = ((A + 8) / 16);
A -= 4000;
B = (A * A) >> 12;
B = (cd.b2 * B) / 2048;
B += (cd.ac2 * A) / 2048;
C = (((((lint)cd.ac1) * 4 + B)) + 2) / 4;
B = (cd.ac3 * A) / 8192;
B += (cd.b1 * ((A * A) >> 12)) / 65536;
B = (B + 2) / 4;
D = (cd.ac4 * (lword) (B + 32768)) / 32768;
A = ((lword)(p - C) * 50000);
if ((A & 0x80000000) == 0)
```

```
        A = ((lword)A * 2) / D;
else
        A = ((lword)A / D) * 2;

B = (lword)A / 256;
B *= B;
B = (B * 3038) / 65536;
press = (lword)A + (B + (((lword)A * (-7357)) /
        65536) + 3791) / 16;
```

Note that the above procedure yields the temperature in tenths of degrees centigrade and the pressure in Pascals. The temperature is signed and can be negative, while the pressure is nonnegative.

On (white) CHRONOS, the main sensor has the number 1 (available as constant SENSOR_PRESSTEMP). In raw mode, the second sensor (used to access the 11 calibration coefficients) has the number –3 (constant SENSOR_PRESSTEMP_CALIB).

This function, available in both modes,

```
void bmp085_oversample (byte ov);
```

sets the oversampling parameter (0 – 3). Its default value is zero.