



SBT80

Multi-Modality Sensor Board for TelosB wireless motes

Plug-in multi-sensor board for IEEE 802.15.4 compliant TelosB wireless motes

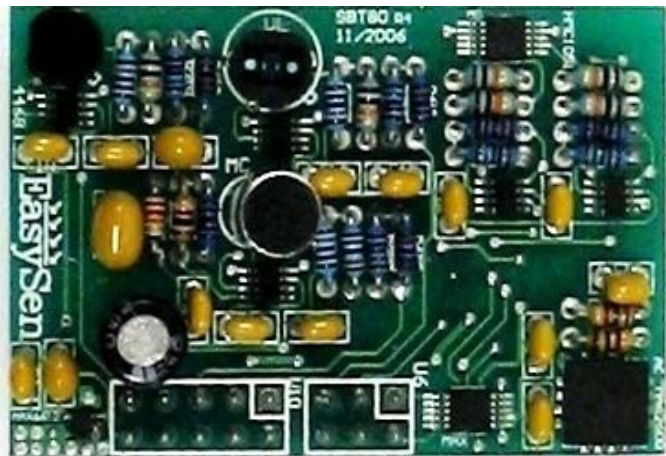
Eight Sensor channels including Visual Light, Infrared, Acoustic, Temperature, Magnetometer (X and Y axis), Accelerometer (X and Y axis)

Product Description

The SBT80 is a low-power, multi-channel sensor board designed for use with IEEE 802.15.4 compliant and TelosB^[1] wireless sensor network platforms. The SBT80 features visual light, infrared, acoustic, temperature, dual-axis magnetometer and dual-axis accelerometer sensors. It can be directly connected to the expansion connectors of the TelosB platform using an IDC header.

The SBT80 is accompanied by tested TinyOS and Java code that can be used to gather data over 8 different sensor channels, display sensor readings on a computer, and to switch the SBT80 into a power saving "sleep" mode.

The SBT80 has a smaller form-factor than TelosB platforms. By leveraging industry standard wireless sensor platforms, the SBT80 enables a wide variety of multi-modal sensor applications for security, surveillance using wireless sensor networks.



Key Features

- Can be directly plugged in to the external connector of IEEE 802.15.4 compliant TelosB motes
- Eight Sensor channels with high sensitivity sensors
- Smaller form factor than TelosB motes
- TinyOS and Java code support available for sampling sensor channels, display readings on PC and switching the sensor board to a power saving mode
- Can be used for multi-modal surveillance and monitoring applications using sensor networks

[1] CrossBow Inc, TelosB (TPR2400CA) Product Datasheet, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf



SBT80

Multi-Modality Sensor Board for
TelosB wireless nodes

Table of Contents

1. Board Description.....	3
2. Sensor Description.....	4
2.1. Visual Light Sensor.....	4
2.2. Infrared Sensor.....	4
2.3. Acoustic Sensor.....	4
2.4. Temperature Sensor.....	5
2.5. Dual axis Magnetometer	5
2.6. Dual axis Accelerometer	5
3. Connection of SBT80 to TelosB.....	5
3.1. Instruction for Soldering Connector to TelosB:	6
4. TinyOS and Java code for reading sensors.....	6
5. Power	11
5.1. Typical Operating Conditions	11
6. Mechanical Characteristics	12
7. General Information	12
7.1. Document History	12
7.2. Disclaimer	12
7.3. Contact Information	12
7.4. Headquarters.....	12



SBT80

Multi-Modality Sensor Board for
TelosB wireless motes

1. Board Description

The SBT80 is a low power, flexible sensor board with multiple sensor modalities (visual light, infrared, acoustic, temperature, dual-axis magnetometer and dual-axis accelerometer) that is specifically designed to be plugged into TelosB wireless motes for use in sensor networks, data fusion, rapid application prototyping and monitoring applications. A front layout of SBT80 is shown in Figure 1.

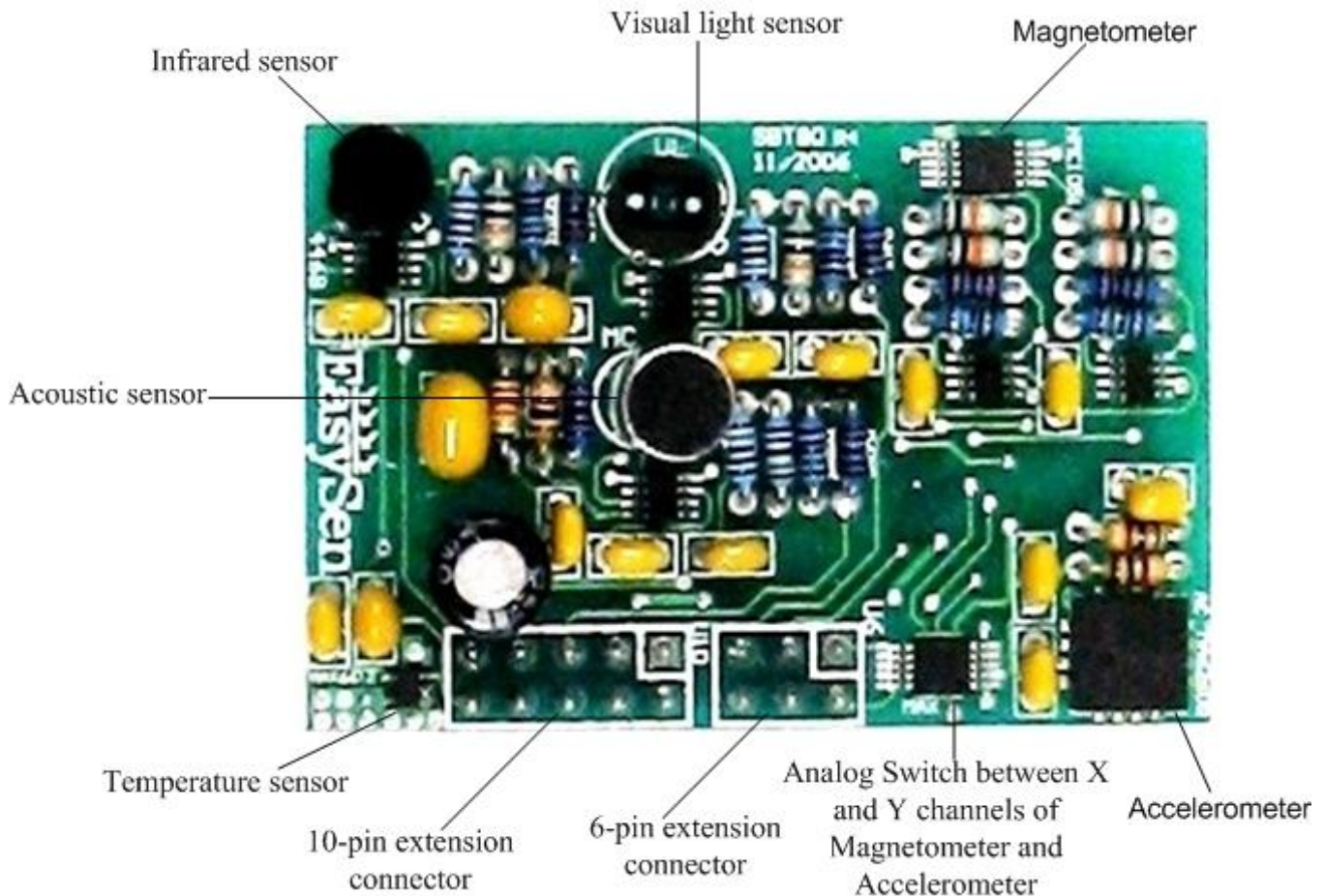
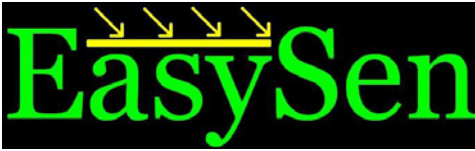


Figure 1. Top view of SBT80



SBT80

Multi-Modality Sensor Board for
TelosB wireless nodes

2. Sensor Description

2.1. Visual Light Sensor

The visual light sensor (VTB9412B, <http://optoelectronics.perkinelmer.com/>) is a small area planar silicon photodiode in a recessed ceramic package. The package incorporates an infrared rejection filter. The sensor has very high shunt resistance and good blue response. It has a wide viewing angle, high sensitivity, and an excellent linearity. The spectral response of the visual light sensor is depicted in Figure 2.

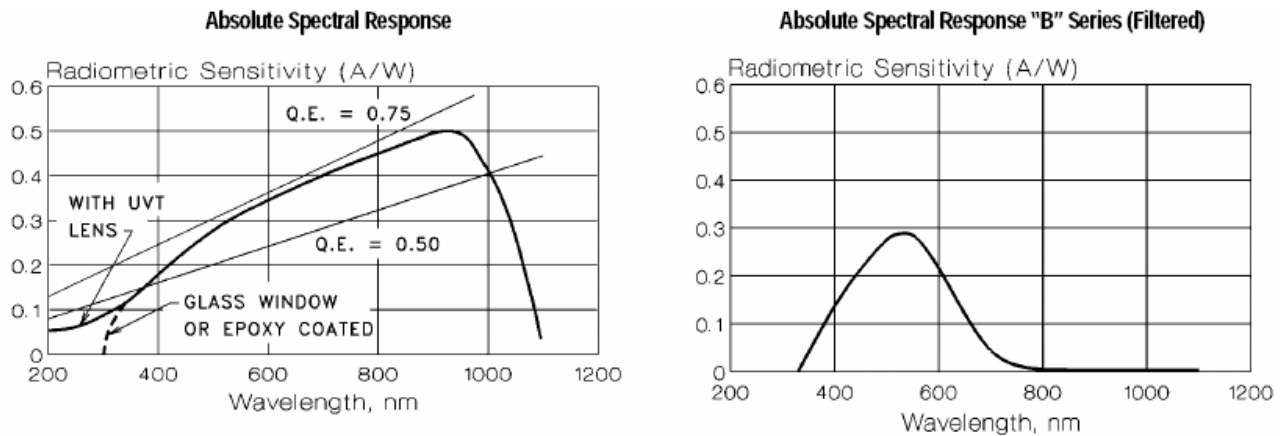


Figure 2. Spectral Response of the Visual Light Sensor

2.2. Infrared Sensor

The infrared sensor (PDB-C139F-ND) is a blue enhanced PIN silicon photodiode in a photoconductive mode with a daylight filter, packaged in a T1 3/4 plastic package. It has a large active area and high responsive speed. The spectral response of the infrared sensor is given in Figure 3.

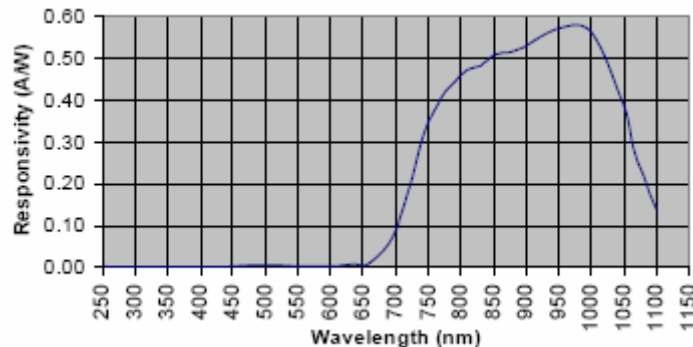


Figure 3. Spectral Response of the Infrared Sensor

2.3. Acoustic Sensor

The acoustic sensor (EM6050P-423, <http://www.digikey.com/>) is a high sensitivity omni directional condenser microphone. It consists of a high voltage internal electret membrane, a metal electrode and a field effect transistor (FET). Its adoption of a high quality FET ensures its reliability and durability.



SBT80

Multi-Modality Sensor Board for TelosB wireless nodes

2.4. Temperature Sensor

The temperature sensor (MAX6612MXK, <http://www.maxim-ic.com/>) is a low-power precision analog output temperature sensor in a tiny 5-pin SC70 package. The sensitivity of the output voltage to temperature is a high 19.53 mV/°C. This sensitivity provides superior noise rejection. The temperature sensor provides an analog voltage output proportional to temperature. Accuracy is $\pm 1.2^\circ\text{C}$ (max) at $+25^\circ\text{C}$, $\pm 3.0^\circ\text{C}$ (max) from $T_A=0^\circ\text{C}$ to $T_A=70^\circ\text{C}$ and $\pm 5.5^\circ\text{C}$ (max) from $T_A=-10^\circ\text{C}$ to 125°C . Useful measurements can be obtained at temperatures as high as -150°C . Self-heating effects are negligible due to its low current consumption ($35\mu\text{A}$ max).

2.5. Dual axis Magnetometer

The magnetometer (HMC1052, <http://www.ssec.honeywell.com/>) is a high performance magnetoresistive sensor with perfectly orthogonal two-axis sensing and low power capability in miniature surface mount packages. It is configured as a 4-element wheatstone bridge to convert magnetic fields to differential output voltages. Capable of sensing fields below 0.1 milligauss, the magnetometer offers a compact, high sensitivity and highly reliable solution for low field magnetic sensing.

2.6. Dual axis Accelerometer

The accelerometer (MMA6260Q, <http://www.freescale.com/>) is a capacitive micromachined accelerometer featuring signal conditioning, a 1-pole low pass filter and temperature compensation. It has a linear output with low noise, low power consumption and high sensitivity. It has a typical sensitivity of 800 mV/g with a typical acceleration range of $\pm 1.5g$.

3. Connection of SBT80 to TelosB

SBT80 provides two interfacing male connectors to Telos nodes, a 6-pin IDC header at position U6 and a 10-pin IDC header at U10. After soldering the female parts on expansion connectors of the Telos nodes, SBT80 can be directly plugged into these nodes (please see below for soldering instructions). The functionality layouts of these two IDC connectors are shown in Figure 4 and 5.

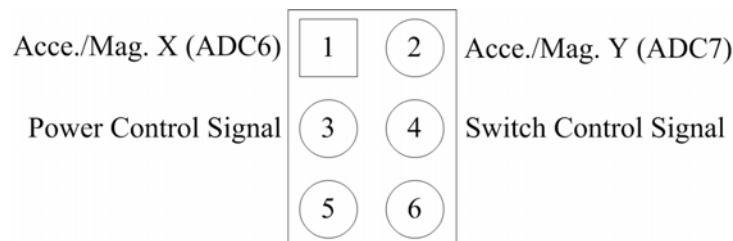
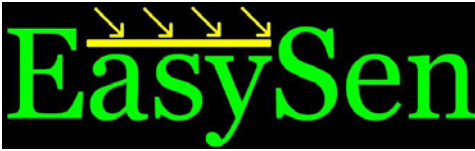


Figure 4. Functionality of the 6-pin IDC connector (U6) (top view)



SBT80

Multi-Modality Sensor Board for
TelosB wireless nodes

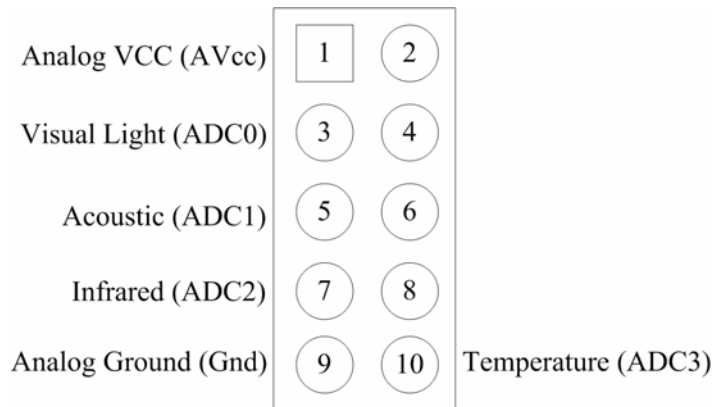


Figure 5. Functionality of the 10-pin IDC connector (U10) (top view)

Note: The ADC channel labels correspond to the expansion connector layout of the TelosB mote. The power of SBT80 is supplied by the connected radio mote (through pin 1 and 9 of U10). The power control signal is used to switch SBT80 between the “work” mode and the power saving “sleep” mode. The switch control signal is used to switch the source of the digital outputs at ADC6 and ADC7 (pin 1 and 2 of U6) between the accelerometer and magnetometer.

3.1. Instruction for Soldering Connector to TelosB

In order to connect a SBT80 to a TelosB mote, a 10-pin female connector (S4305-ND, www.digikey.com) and a 6-pin female connector (S4303-ND, www.digikey.com) need to be soldered to the 10-pin (U2) and 6-pin (U28) expansion connectors on the TelosB mote. After soldering these two connectors, the SBT80 can be directly plugged into these connectors and interfaced with the TelosB mote, as shown in Figure 6.

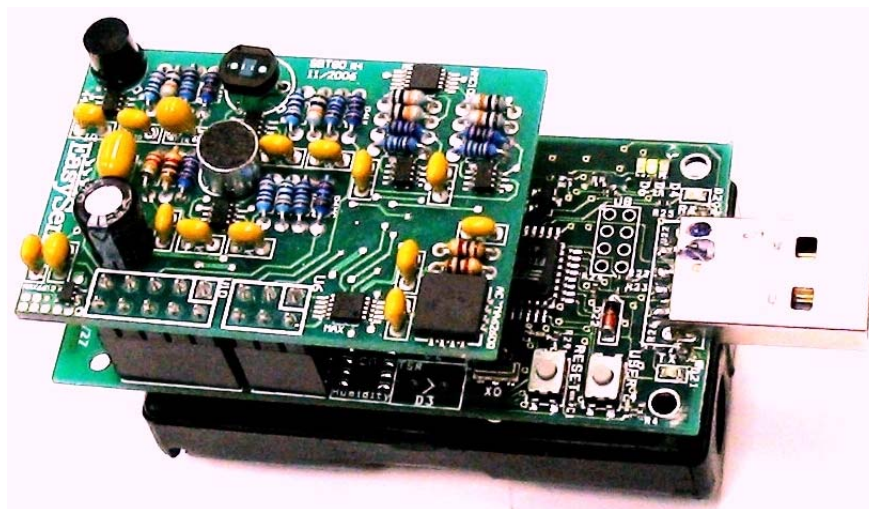


Figure 6. SBT80 plugged into a TelosB mote (top view)



SBT80

Multi-Modality Sensor Board for
TelosB wireless nodes

4. TinyOS and Java code for reading sensors

The TinyOS and Java code for reading 8 sensor channels of SBT80 over the TelosB nodes are available at <http://www.easysen.com/support/SBT80>

The sample application folder **SBT80app** consists of 2 header files (SBT80ADCmap.h and SBT80Msg.h), a configuration file (MobileNode.nc) and a module file (MobileNodeM.nc). It samples all 8 sensor channels in a sequence and transmit data to a PC.

The file ListenSBT80.java reads and displays the packets transmitted using the MobileNode application above.

Programming sequence:

1. Program one TelosB node with the **MobileNode** application given here.
 - (a) Copy and unzip the folder **SBT80app** into the /opt/tinyos1.x/apps folder in your TinyOS installation
 - (b) Connect TelosB node to the PC USB connector and execute the command **make telosB install** inside the folder SBT80app to install the application
 - (c) Connect the sensor board SBT80 to the expansion connector of the TelosB node
2. Program a base station with the generic **TOSBase** application
 - (a) Connect another TelosB node to the PC USB connector and execute the command **make telosB install** inside the folder /opt/tinyos1.x/apps/TOSBase
3. Display sensor readings on PC
 - (a) Copy the java program ListenSBT80.java to the folder /opt/tinyos1.x/tools/java/net/tinyos/tools in your TinyOS installation
 - (b) compile by executing the command **javac ListenSBT80.java** inside the same folder
 - (c) set the MOTECOMM variable to read the USB port properly according to the TinyOS tutorial guide
 - (d) run the java application by executing the command **java net/tinyos.tools.ListenSBT80**

This should produce a display of sensor readings from all 8 channels on the cygwin window.

Configuration file (MobileNode.nc) Explained

```
includes SBT80Msg; // include header files for SBT80
includes SBT80ADCmap;

configuration MobileNode { }
implementation
{
  components Main, MobileNodeM
    , TimerC
    , LedsC
    , GenericComm as Comm
    , ADCC
    , MSP430GeneralIOCC ;
```



SBT80

Multi-Modality Sensor Board for TelosB wireless nodes

```
Main.StdControl -> MobileNodeM;
Main.StdControl -> TimerC;
Main.StdControl -> ADCC;

MobileNodeM.SampleTimer -> TimerC.Timer[unique("Timer")];
MobileNodeM.SwitchTimer -> TimerC.Timer[unique("Timer")];
MobileNodeM.Leds -> LedsC;

MobileNodeM.RadioControl -> Comm;
MobileNodeM.RadioSend -> Comm.SendMessage[AM_SBT80MSG];

MobileNodeM.ADCCControl -> ADCC.ADCCControl;
MobileNodeM.VL -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC0_PORT]; // Map TelosB ADC ports
MobileNodeM.MIC -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC1_PORT];
MobileNodeM.IR -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC2_PORT];
MobileNodeM.TEMP -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC3_PORT];
MobileNodeM.ACMGchX -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC6_PORT];
MobileNodeM.ACMGchY -> ADCC.ADC[TOS_ADC_EXTERNAL_ADC7_PORT];

MobileNodeM.SBcontrol -> MSP430GeneralIO.Port23; // Map TelosB external output ports
MobileNodeM.SBswitch -> MSP430GeneralIO.Port26;
}
```

Module file (MobileNodeM.nc) Explained

```
includes SBT80Msg;
includes SBT80ADCmap;

module MobileNodeM
{
  provides {
    interface StdControl;
  }
  uses {
    interface Timer as SampleTimer;
    interface Timer as SwitchTimer;
    interface Leds;

    interface ADC as VL; // Map TelosB ADC ports to sensors
    interface ADC as MIC;
    interface ADC as IR;
    interface ADC as TEMP;
    interface ADC as ACMGchX;
    interface ADC as ACMGchY;

    interface MSP430GeneralIO as SBcontrol; // Map TelosB output port to SBT80 control pin
    interface MSP430GeneralIO as SBswitch; // Map TelosB output port to SBT80 switch pin

    interface ADCCControl;
    interface StdControl as RadioControl;
    interface SendMessage as RadioSend;
  }
}

implementation
{
  uint8_t currentMsg;
  uint16_t sampleTime;
  uint16_t sampledChannelNo;
}
```



SBT80

Multi-Modality Sensor Board for TelosB wireless motes

```
TOS_Msg  msg[2];
char  ACMGswitch;
enum  {AC,MG};

command result_t StdControl.init() {
    call Leds.init();
    call Leds.yellowOff(); call Leds.redOff(); call Leds.greenOff();
    call RadioControl.init();

    // Initialize TelosB ADC ports
    call ADCControl.init();
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC0_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC0_PORT);
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC1_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC1_PORT);
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC2_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC2_PORT);
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC3_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC3_PORT);
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC6_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC6_PORT);
    call ADCControl.bindPort(TOS_ADC_EXTERNAL_ADC7_PORT, TOSH_ACTUAL_ADC_EXTERNAL_ADC7_PORT);

    atomic {
        currentMsg = 0;
        sampledChannelNo = 0;
        sampleTime = 0;
    }
    return SUCCESS;
}

command result_t StdControl.start() {
    call SampleTimer.start(TIMER_REPEAT,1000);
    call RadioControl.start();

    call SBcontrol.setLow(); // Wake up the sensor board
    call SBcontrol.makeOutput();
    call SBcontrol.selectIOFunc();
    return SUCCESS;
}

command result_t StdControl.stop() {
    call SampleTimer.stop();
    call RadioControl.stop();
    return SUCCESS;
}

// ----- TASK : Send packet after all sensor data is gathered -----
task void dataTask() {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        sampledChannelNo = 0;
        pack->timeStamp[5] = 0x00;
    }
    pack->sourceMoteID = TOS_LOCAL_ADDRESS;

    // Send packet
    if (call RadioSend.send(TOS_BCAST_ADDR, sizeof(struct SBT80Msg), &msg[currentMsg]))
    {
        atomic {currentMsg ^= 0x1;}
        call Leds.greenToggle();
    }
}

// ----- TASK : Set the Switch for reading ACcel or MaGnetic sensors -----
task void switchTask() {
    if (ACMGswitch == MG) {call SBswitch.setHigh();} // High = Magnetic
    else {call SBswitch.setLow();} // Low = Accel
}
```



SBT80

Multi-Modality Sensor Board for TelosB wireless nodes

```
    call SBswitch.makeOutput();
    call SBswitch.selectIOFunc();
}

// ----- EVENT: SampleTimer fired -----
event result_t SampleTimer.fired() {
    result_t ok1;

    ACMGswitch = AC;
    post switchTask();

    call Leds.redOn();
    ok1 = call VL.getData(); // Read Visual Light Sensor VL
    return ok1;
}

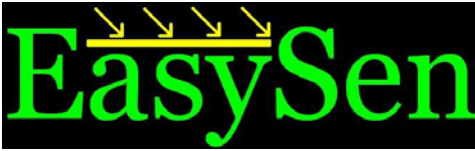
// ----- EVENT: VL data ready - Read MIC data -----
async event result_t VL.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        pack->data[sampledChannelNo++] = data;
    }
    call MIC.getData(); // Read Microphone MIC
    return SUCCESS;
}

// ----- EVENT: MIC data ready - Read IR data -----
async event result_t MIC.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        pack->data[sampledChannelNo++] = data;
    }
    call IR.getData(); // Read Infrared Sensor IR
    return SUCCESS;
}

// ----- EVENT: IR data ready - Read TEMP data -----
async event result_t IR.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        pack->data[sampledChannelNo++] = data;
    }
    call TEMP.getData(); // Read Temperature Sensor TMP
    return SUCCESS;
}

// ----- EVENT: TEMP data ready - Read ACcel channel X data over ADC6-----
async event result_t TEMP.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        pack->data[sampledChannelNo++] = data;
    }
    call ACMGchX.getData(); // Read Acceleration or Magnetic Sensor AC/ MG chX
    return SUCCESS;
}

// ----- EVENT: ACMGchX data ready -----
async event result_t ACMGchX.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
```



SBT80

Multi-Modality Sensor Board for
TelosB wireless motes

```
    pack = (struct SBT80Msg *)msg[currentMsg].data;
    pack->data[sampledChannelNo++] = data;
}
call ACMGchY.getData(); // Read Acceleration or Magnetic Sensor AC/MG chY
return SUCCESS;
}

// ----- EVENT: ACMGchY data ready -----
async event result_t ACMGchY.dataReady(uint16_t data) {
    struct SBT80Msg *pack;
    atomic {
        pack = (struct SBT80Msg *)msg[currentMsg].data;
        pack->data[sampledChannelNo++] = data;
    }
    if (ACMGswitch == AC){ // Switch between Acceleration and Magnetic Sensors
        ACMGswitch = MG;
        post switchTask();
        call SwitchTimer.start(TIMER_ONE_SHOT,20);
        // short delay to stabilize after switching between AC and MG
    }
    else{
        call Leds.redOff();
        post dataTask();
    }
    return SUCCESS;
}

// ----- EVENT: SwitchTimer fired -----
event result_t SwitchTimer.fired() {
    call ACMGchX.getData(); // Read Acceleration or Magnetic Sensor AC/ MG chX
    return SUCCESS;
}

// ----- EVENT: Packet Tx Done -----
event result_t RadioSend.sendDone(TOS_MsgPtr sent, result_t success) {
    return SUCCESS;
}
}
```

5. Power

SBT80 is powered by the attached TelosB mote through the 10-pin interfacing connector. The mote itself is powered by two AA batteries or a host computer if it is attached to its USB. The operating voltage for SBT80 falls between 2.4V and 3.6V. At no point should the input voltage exceed 3.6V—doing so may damage certain components. A power control pin is provided on the 6-pin connector (see Figure 4), which switches SBT80 to the normal “work” mode if input is low, and to the “sleep” mode if input is high. In the “sleep” mode, all the sensor channels are turned off and SBT80 consumes a minimal current of less than 1.5mA, while in the normal “work” mode, it consumes a maximal current of 8.0mA or less.

5.1. Typical Operating Conditions

	MIN	MAX	UNIT
Supply voltage	2.4	3.6	V
current Consumption: “Sleep” mode		1.5	mA
current Consumption: “Awake” mode		8.0	mA



SBT80

Multi-Modality Sensor Board for
TelosB wireless nodes

6. Mechanical Characteristics

Parameter	Value	UNIT
width	1.26	in
length	1.89	in
height (PCB)	0.06	in
height (with connector and components)	0.86	in

7. General Information

7.1. Document History

Revision	Date	Notes
1.0	07/18/2006	Initial Release
1.0.1	07/22/2006	Minor revision on Accelerometer Sensor
1.0.2	10/19/2006	Minor revision on programming guide
1.0.3	02/22/2007	Revision on Visual Light Sensor

7.2. Disclaimer

Easysen LLC trusts that the information contained in this document is correct and accurate at the time of this printing. EasySen does however reserve all rights to make changes to this product without prior notice. Easysen does not take on any liability for the use of the described product; neither does it pass on any license under its patent rights, or the rights of others. This product is not designed for use in critical or life support systems where failure of the product to perform affects safety or effectiveness or results in any personal injury to the user. EasySen does not take any responsibility for the misuse or resale of our product by our customers. Customers using or selling our product do so at their own risk. They further agree to fully assure EasySen not to seek any compensation for damage or injury claims resulting from inappropriate use or sale of our product. To our best ability, major changes of product specifications and functionality, will be published at the EasySen website. The latest updates are available from the EasySen website at www.easysen.com or by contacting EasySen directly.

7.3. Contact Information

Web site: <http://www.easysen.com>
E-mail: info@easysen.com
Technical Support E-mail: support@easysen.com
Sales Contact E-mail: sales@easysen.com
Phone Number: +1.574.607.5047

7.4. Headquarters

EasySen, LLC.
401 North Coquillard Dr.
South Bend, IN 46617

© 2005-2007 EasySen LLC.