

Sensor Programming Tutorial

Sensors in Tmote Sky

- Most IoT devices include sensors, and the data streams from these sensors are an important component of the IoT concept
- The Tmote Sky mote has integrated sensors for temperature, humidity and light intensity
- Sensors are managed in Contiki OS as follows
 - The library “dev/sht11/sht11-sensor.h” is used to manage the temperature and humidity sensors
 - The library “dev/light-sensor.h” is used to manage the light intensity sensor

Sensor Simulation Example

- The simplest way to open the simulation is to select “Sensor Simulation” in the IoTrain-Sim interface
- Alternatively, you can open it manually as follows
 - Open Cooja
 - Click File > Open simulation > Browse...
 - Go to the folder “iotrain-sim/database/fundamental_training/single_node/sensing/simulation/”
 - Select the file “sensor.csc”
 - Click Open
- Once the simulation control window appears, click the “Start” button to begin the simulation
 - The mote will print data from all sensors as described next
 - The logical running time for this simulation is set to 10 minutes

Source Code Commentary

- Print data from all sensors every two seconds
 - Source code: iotrain-sim/database/fundamental_training/single_node/sensing/simulation/sensor.c

```
#include "contiki.h"
```

```
#include "dev/light-sensor.h"
```

```
#include "dev/sht11/sht11-sensor.h"
```

} Include sensor libraries

```
#include <stdio.h>
```

```
#include <math.h>
```

```
/*-----*/
```

```
PROCESS(sensor_acq_process,"Sensor Acquisition");
```

```
AUTOSTART_PROCESSES(&sensor_acq_process);
```

Source Code Commentary (cont.)

```
PROCESS_THREAD(sensor_acq_process, ev, data)
{
    static struct etimer et;
    static int val;
    static float s = 0;
    static int dec;
    static float frac;
    PROCESS_BEGIN();
    printf("Starting Sensor Example.\n");
    while(1)
    {
        etimer_set(&et, CLOCK_SECOND * 2);    // Set timer to repeat the iterations every 2 seconds

        SENSORS_ACTIVATE(light_sensor);
        SENSORS_ACTIVATE(sht11_sensor);    }    Activate light_sensor to measure the light intensity and
                                           sht11_sensor to measure temperature and humidity

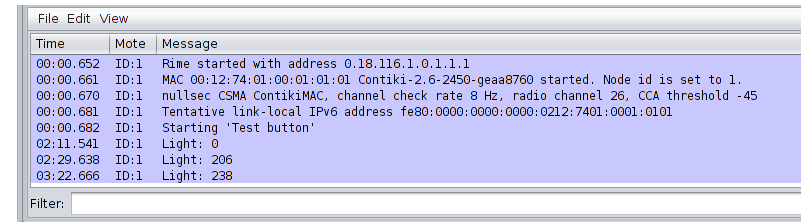
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        val = sht11_sensor.value(SHT11_SENSOR_TEMP);    // Get the temperature sensor value
        if(val != -1)
        {
            s = ((0.01*val) - 39.60);    // Calibrate the sensor value by doing some calculation
            dec = s;
            frac = s - dec;
            printf("\nTemperature=%d.%02u C (%d)\n", dec, (unsigned int)(frac * 100), val);
        }
    }
}
```

Source Code Commentary (cont.)

```
    val=sht11_sensor.value(SHT11_SENSOR_HUMIDITY); // Get the humidity
    if(val != -1)
    {
        s= (((0.0405*val) - 4) + ((-2.8 * 0.000001)*(pow(val,2))));
        dec = s;
        frac = s - dec;
        printf("Humidity=%d.%02u %% (%d)\n", dec, (unsigned int)(frac * 100),val);
    }
    val = light_sensor.value(LIGHT_SENSOR_TOTAL_SOLAR); // Get the light
    if(val != -1)
    {
        s = (float)(val * 0.4071);
        dec = s;
        frac = s - dec;
        printf("Light=%d.%02u lux (%d)\n", dec, (unsigned int)(frac * 100),val);
    }
    etimer_reset(&et);
    SENSORS_DEACTIVATE(light_sensor);
    SENSORS_DEACTIVATE(sht11_sensor); } Deactivate all the sensors
} //end of while
PROCESS_END();
}
```

Exercise

- Create an application with a button that, when pressed, displays the value of the light sensor as in the figure
- Verify the program by running it in Cooja and checking the console output when the button is pressed
- Hints
 - Remember to modify the Makefile by adding the new file name to “CONTIKI_PROJECT”
 - You can check the following file for a possible solution: “iotrain-sim/database/fundamental_training/single_node/sensing/simulation/button-light-sensor.c”



The screenshot shows a Cooja console window with a menu bar (File, Edit, View) and a table of simulation logs. The table has three columns: Time, Mote, and Message. The logs show the initialization of a Rime node, MAC address assignment, CSMA channel check, and the starting of a 'Test button'. The light sensor value is shown as 0 at 02:11.541 and 206 at 02:29.638.

Time	Mote	Message
00:00.652	ID:1	Rime started with address 0.18.116.1.0.1.1.1
00:00.661	ID:1	MAC 00:12:74:01:00:01:01:01 Contiki-2.6-2450-geaa8760 started. Node id is set to 1.
00:00.670	ID:1	nullsec CSMA ContikiMAC, channel check rate 8 Hz, radio channel 26, CCA threshold -45
00:00.681	ID:1	Tentative link-local IPv6 address fe80:0000:0000:0000:0212:7401:0001:0101
00:00.682	ID:1	Starting 'Test button'
02:11.541	ID:1	Light: 0
02:29.638	ID:1	Light: 206
03:22.666	ID:1	Light: 238

Filter: