Lab 2

Rafik Zitouni

ECE Paris

rafik.zitouni@ece.fr

Timer and Sensors

Timers of Contiki

Trigger events at a given time as **blinking a LED every 5 seconds**

- **Simple Timer:** The application can check manually if the timer has expired. You can see for more details: **core/sys/timer.h**
- **Simple Timer:** When a timer expires it can callback a given function **core/sys/ctimer.h**
- **Event timer:** Application trigger an event after timer expires**. core/sys/etimer.h**
- **Real Time Timer:** It handles the scheduling and execution of real time tasks **core/sys/rtimer.h**

Timer and Sensors

Example 1 : Timers

When timer expires, an application changes its behavior → **Event Timer**

CLOCK_SECOND is a value related to the number of the microcontroller's ticks per second. **CLOCK_SECOND** also differs.

PROCESS_WAIT_EVENT() waits for *any* event to happen.

```
#include "contiki.h"
#include "dev/leds.h"
#include <stdio.h>
/*---------------------------------------------------*/
#define  SECONDS 2
/*---------------------------------------------------*/
PROCESS(test_timer_process, "Test timer");
AUTOSTART_PROCESSES(&test_timer_process);
/*---------------------------------------------------*/
PROCESS_THREAD(test_timer_process, ev, data)
{
  PROCESS_BEGIN();
  static struct etimer et;
  while(1) {
    etimer_set(&et, CLOCK_SECOND*SECONDS);
    PROCESS_WAIT_EVENT();
   if(etimer_expired(&et)) {
     printf("Hello world!\n");
     etimer_reset(&et);
   }
 }
 PROCESS_END();
```

Timer and Sensors

Exercise 1:

1) Print the value of CLOCK_SECOND to count how many ticks you have in one second?

2) Blink a LED for a certain number of seconds.

3) Add an application related to your timer and button. The timer starts only when the button is pressed and stops when the button is pressed again.

Timer and Sensors

Sensors

Electrical and optical signals are the output of a sensed variable.

Contiki implements sensors as follow:

SENSORS_SENSOR (sensor, SENSOR_NAME, value, configure, status);

The method **configure** the **sensor**, poll the sensor **status** and request a **value**. The sensor structure contains pointers to these functions. The arguments for each function are shown below.

```
struct sensors_sensor {
   char *       type;
   int          (* value)     (int type);
   int          (* configure) (int type, int value);
   int          (* status)    (int type);
};
```

For analog sensors: **platform/zoul/dev/adc-sensors.c**

Timer and Sensors

Phidget Sensors

%Enable or configure the sensor
SENSORS_ACTIVATE(sensor)

%Desable the sensor
SENSORS_DEACTIVATE(sensor)

%Request a type of value to the sensor (tempertature
sensor.value(type);

RE-Mote allows up to **6 analog sensors** to be connected, but only **two *phidget* connectors** can be soldered at the same time.

It is possible to connect **3,3V** and **5V** analog sensor with ADC1 and ADC 3, respectively.

Timer and Sensors

Parameter of a Phidget light sensor

Parameter	Value
Sensor type	Light
Light level min	1 lux
Supply Voltage Min	2.4 V
Supply Voltage Max	5.5 V
Max current consumption	5mA
Light level max (3.3 V)	660 lux
Light level max (5 V)	**1000 lux**

One lux is equal to one lumen per square meter **1 lx = 1 lm/m^2**

Timer and Sensors

To read data from an attached sensor (as we did in the previous example)

ANALOG_GROVE_LIGHT
In the case of light sensor

ADC driver header

Choose which ADC channel to enable (here all channels)

```
#include "dev/zoul-sensors.h"
adc_sensors.configure(SENSORS_HW_INIT, REMOTE_SENSORS_ADC_ALL);
printf("Phidget ADC2 = %d raw\n", adc_sensors.value(REMOTE_SENSORS_ADC2));
printf("Phidget ADC3 = %d raw\n", adc_sensors.value(REMOTE_SENSORS_ADC3));
```

Request a sensed value

example/zolertia/zoul/zoul-demo.c gives more details.

See also the example : ***example/zolertia/zoul/*****test-grove-light-sensor.c**

Timer and Sensors

Exercise 2:

1) Make the sensor take readings as fast as possible.

2) Print to the screen the ADC raw values and the millivolts (as this sensor is linear, the voltage corresponds to the luxes). What are the max and min values you can get?

3) What is the average light value of the room?

4) Create an application that turns the red LED on when it is dark and the blue LED on when the room is bright. In between, all the LEDs should be off. Add a timer and measure the light every 10 seconds.

5) Send the reading of the sensor to a mote connected to the PC (Sink). You should choose
   1) Unicast application
   2) Broadcast application