# Lab Exercise 1 – Contiki Introduction

## Objectives:

- Introduction to Contiki, an event-based operating system.
- Become familiar with the Contiki concepts, syntax and source file structure.
- Learn how to use the SensorTag wireless embedded platform.

**Note:** Please be gentle with the hardware. The SensorTags and programming boards are rather delicate. Please handle carefully.

## Give:

You must submit your code via *give* by the assessment date (which is the day of your lab in the following week) or you will receive a mark of 0. You may submit as many times as you wish. Your latest submission will override previous files.

## Introduction:

This exercise introduces you to Contiki, an operating system designed to work on resource constrained wireless embedded sensing platforms such as SensorTags used in our labs. The goal is to understand the basics of Contiki, gain familiarity with the programing environment and the SensorTags. You will be asked to create a blink application and modify it to communicate with the PC over a serial connection.

### 1. Contiki Hello World Example

Follow Tutorial 1, to setup your programming environment. Your first application will be the LED example. You must attempt to compile and download it to the SensorTags. Use kermusb to view the serial output of the SensorTags.

### 2. Serial Input, Processes and Timers

Follow Tutorials 2, 3 and 4 to learn how to use serial input, create processes and how to control timing events.

**Please finish the provided tutorials before attempting to solve this exercise. Following a systematic approach will save you a lot of time and heartache.**

### Marking criteria

**Demonstration (5 marks) - Due next week**

Modify the Hello-World example to perform a number of tasks (detailed below) based on PC serial input. Create a folder called *prac1* in your repo folder and set *prac1* as your contiki project name (Makefile). See the main lab page on the website for submission instructions.

The user will enter simple commands (see Serial-Interface example) using a serial input terminal (kermusb). Each key press should be seen in the serial terminal. **Note**: each command must be followed by holding down the “ctrl key” and then pressing “enter”. The application will respond by toggling LEDs, or by printing output to the PC. You must use a thread to receive serial input and control the LEDs flashing period and status.

**Task 1 (2 marks):**
Use ‘r’, ’g’, ’a’ key strokes to start/stop toggling LED Red/Green/all on the sensor tag with a frequency of 1Hz. (hint: you have to use timer to toggle the LED)

**Task 2 (2 marks):**
Use the ‘b’ key stroke to turn on/off the buzzer (**1 mark**). Use ‘i’, and ‘d’ key strokes to increase or decrease the sound frequency by 50Hz for 5 seconds. (**1 mark**) (hint: use global variables *buz_state* and *buz_freq*)

**Task 3 (1 mark):**
Use the ‘n’ key stroke to print the sensor tag’s IEEE Address in hexadecimal. (hint: see tutorial 2)