HOME CHALLENGE #1: TINY OS REPORT

The goal of this activity is to Create a Cooja simulation with three TinyOS (sky) motes that communicate over the radio.

In order to successfully complete the exercise we took in consideration the “RadioCountLedsApp” code and we used a GitHub repository to simplify the team work.

* The **first request** was to create a message structure composed by a counter and a sender id. In order to do so, we modified the parameter nx\_struct radio\_count\_msg*.* The message structure required one more field: an unsigned Int of 16 bits for defining the variable senderId.

typedef nx\_struct radio\_count\_msg {  
 nx\_uint16\_t counter;  
 nx\_uint16\_t senderId;  
} radio\_count\_msg\_t;

* The **second request** was to define the frequency at which every mote sends messages. We have solved this problem by introducing the macro TOS\_NODE\_ID that defines the three motes ids from 1 to 3. Based on the id we set the sending period at 1000ms, 333ms and 200ms. The sender id was needed to be included into the message sent, so in the message we recall the value of the TOS\_NODE\_ID variable.

uint16\_t periods[] = {1000, 333, 200};  
*// ...*  
period = periods[TOS\_NODE\_ID - 1];  
call MilliTimer.startPeriodic(period);

* The **third request** was to control each led and make it turn on every time the corresponding mote sends a message (the first led associated to the first mote, and so on). To do so the program checks the senderId values to decide when to turn a led on using the ﻿ledXToggle() function.
* The **fourth request** was to turn off all the leds when the variable counter (that has been increasing of a unit for every received message) reaches the value of 10. The program checks if the rest of the division by 10 of the counter variable is 0 or not.

if (rcm->counter % 10 == 0) {  
 printf("Received message with counter %u, LEDs off.\n", rcm->counter);  
 call Leds.led0Off();  
 call Leds.led1Off();  
 call Leds.led2Off();  
 } else if (rcm->senderId == 1) {  
 call Leds.led0Toggle();  
 printf("Toggling LED %u.\n", rcm->senderId);  
 } else if (rcm->senderId == 2) {  
 call Leds.led1Toggle();  
 printf("Toggling LED %u.\n", rcm->senderId);  
 } else if (rcm->senderId == 3) {  
 call Leds.led2Toggle();  
 printf("Toggling LED %u.\n", rcm->senderId);  
 }

It’s worth noting that, since the debug messages weren’t showing for us in the Cooja simulation, we found a working solution by importing the printf library from TinyOS. This requires adding the SerialPrintfC component and adding the library path in the CFLAGS environmental variable inside the Makefile.

Repository link and contacts

The project repository can be found at the following link: <https://github.com/NonSvizzero/IoT2020>.

Team members:

* Giuseppe Maria Fiorentino (10590418, giuseppemaria.fiorentino@mail.polimi.it)
* Riccardo Novic (10496965, riccardo.novic@mail.polimi.it).
* Raffaele Zenga (10611699, raffaele.zenga@mail.polimi.it)