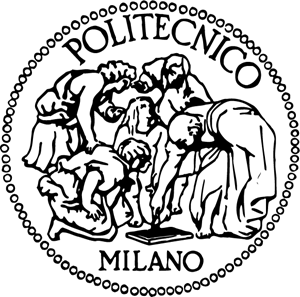
HOME CHALLENGE #2: TOSSIM (SendACK) 

The goal of this activity is to Create a Tossim simulation with two motes talking to each other.

In order to successfully complete the exercise we took in consideration the previous challenge’s code and the draft of the SendACK code provided to us.

In this report we highlight the differences and the additions with respect to the last challenge.

The message structure is now composed by a counter, a sender id and a value from the fake sensor. In order to do so, we modified the payload of the message nx\_struct radio\_count\_msg*.* adding the value field. In order to distinguish the two different messages without using typing checks, we’ve also added a field to check at runtime the type of message received by the mote.

typedef nx\_struct my\_msg {

nx\_uint8\_t msg\_type;

nx\_uint16\_t counter;

nx\_uint16\_t value;

} my\_msg\_t;

Worth noting that now the motes do not communicate in broadcast anymore, but instead talk on a point-to-point channel. The connection graph is described in the topology.txt file.

As described in the RunSimulationScript.py file, the mote #1 boots at t=0s and sends requests messages (REQ) every 1000ms with an incremental counter to the mote #2, which wakes up at t=5s. After 5 initial messages without an ACK received, the second mote is finally able to reply with a RESP message containing the fake sensor value. This behavior can be seen by running the python script:

DEBUG (1): [t=0:0:0.000000000] The Application is now booted.

DEBUG (1): [t=0:0:0.976562510] Sent request to node 2. (counter=0)

ACK not received.

DEBUG (1): [t=0:0:1.953125010] Sent request to node 2. (counter=1)

ACK not received.

DEBUG (1): [t=0:0:2.929687510] Sent request to node 2. (counter=2)

ACK not received.

DEBUG (1): [t=0:0:3.906250010] Sent request to node 2. (counter=3)

ACK not received.

DEBUG (1): [t=0:0:4.882812510] Sent request to node 2. (counter=4)

ACK not received.

DEBUG (2): [t=0:0:5.000000000] The Application is now booted.

DEBUG (1): [t=0:0:5.859375010] Sent request to node 2. (counter=5)

DEBUG (2): [t=0:0:5.867111170] Message received with counter=5.

ACK received.

DEBUG (2): [t=0:0:5.875976572] Sent response to node 1.

DEBUG (2): Value read from sensor: 50421.

DEBUG (1): [t=0:0:5.884429892] Message received with counter=5.

DEBUG (1): Value received from node 2: 50421

ACK received.

To handle the ACK messages, the module PacketAcknowledgements has been used. Since the first mote is the only one using a timer, we can safely stop the timer after an ACK has been received without checking the TOS\_NODE\_ID variable.

if (call PacketAcknowledgements.wasAcked(buf)) {

dbg\_clear("radio\_ack","ACK received.\n");

call MilliTimer.stop();

} else {

dbg\_clear("radio\_ack","ACK not received.\n");

}

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)