Raffaele Cicellini 10626081

Francesca Forbicini 10628756

**Project Report – Smart Bracelets – Internet of Things**

To implement this project, we decided to use TinyOS with TOSSIM as a simulation tool.

We divided the source code into 4 different files, like in Challenge4:

1. ProjectC.nc contains the main logic of the program
2. ProjectAppC.nc contains the “wiring” of the needed components to implement the mote
3. FakeSensorC.nc is used to implement the reading of the status of the child (position and kinetic status)
4. FakeSensorP.nc contains the “wiring” for FakeSensorC.nc.

We followed the order of the operations performed by the mote to implement them:

* Once the mote boots, we start a periodic timer used for the pairing phase. Every time this timer is fired the mote sends in broadcast a pairing message (msg type=0) containing its pairing key (this message does not require an ack). This timer stops only when the paired bracelet replies with a confirmation. The keys were generated by a simple python script and then we preloaded them saving them in an array in Project.h file.
* Once a mote receives a pairing message, it checks if the key contained in the payload is equal to its one: if this is the case, it saves the source address (it is the address of the paired bracelet) and calls function send\_pairing() to reply with a confirmation. The function simply sends a message of type 1 (confirmation) containing the key and it is sent in unicast to the paired bracelet (requiring an ack). Once the ack is received the mote can enter the operational mode. WARNING: only the mote which receives the broadcast message (the one of type 0) enters operational mode by sending the confirmation, meanwhile it must also send a broadcast message so that also the paired device can complete it (saving the paired device address, etc) and enters the operational mode. Only when both the motes complete the pairing phase, they can exchange operational messages.
* We implemented the operational mode for the two different motes (child and parent bracelet) using the same main file (ProjectC.nc) but using an if statement after the pairing up phase is completed to choose between the operational mode of the child bracelet of the parent bracelet. For the child node, the operational mode consists of a timer that is periodically fired every 10 seconds: every time this happens the node reads from a fake sensor the values of the kinetic status and the position of the child. These values are generated randomly by FakeSensorP.nc according to what is written in the specification of the project. If the read is not successful, it is repeated until it’s not failed; while if it is successful, once the values are read, the child node sends an info message containing them to the paired parent device using the function send\_INFO\_msg(), requiring also an ack. For the parent node, instead, the operational mode consists of a one-shot timer of 60 seconds after which it sends a missing alarm printing the last known status of the child. Moreover, every time the node receives a message from the paired child device it reads the status of the child (printing it in debug), saves the current status and position, and if the kinetic status is equal to “FALLING” it produces a falling alarm indicating the current position of the child. Finally, it stops the 60 seconds timer (since it received a message) and it starts it again.
* For what concerns the TOSSIM simulation script, we created different debug channels to print everything we need for the log file:
  + The application is booted
  + The device is ready
  + A timer starts or is fired or is stopped
  + The “Pairing Phase” or “Operational Mode Phase” starts, ends, or failed (i.e. if the node didn’t receive an ACK for an info message)
  + The sensor sends a pairing packet and sends or receives an ACK
  + The sensor sends an INFO message to a node and receives an ACK or not from it
  + When a node receives an INFO message, prints the position and status and if the status is a ‘FALLING’ status, an ALERT message is printed
  + If a node doesn’t correctly receive the INFO message, pairing phase message, or operational mode phase message, it is printed and it starts again the procedure

Then we created 4 nodes starting at the same time. We also created the noise and the topology file by using the previous example of Challenge 4 and adapting it to this case. Finally, we simulated execution by running the simulation for 200 seconds and simulating the missing event by turning off a node, as suggested in the project specification.