MDSmartBracelets

Paolo Maralit - 10631828
Niccoló Didoni - 10623061

June 2022

1 Operating system

As operating system, we decided to use TinyOS since the application is mainly based on timers, which are easy to handle in TinyOS. Moreover, we decided to build a single application in common for the child and the parent to simulate the interaction both with Cooja and Tossim. We couldn't find a way to use Tossim with different applications, yet we preferred to use it since it's more dynamic and allows to define more test cases.

2 Communication

2.1 Pairing

The pairing protocol allows to pair two devices A and B. The protocol works as follows:

1. Mote A broadcasts a `PAIR` message with its 20-bit key.
2. Mote B receives a `PAIR` message from A.
3. Mote B checks if the 20-bit key in the received message is the same as the one stored in memory.
4. Mote B replies with a `PAIREND` message.
5. Mote A, upon receiving the `PAIREND` message from B, stops broadcasting `PAIR` messages.
6. Mote A acknowledges B's `PAIREND` message.

The same procedure happens with inverted role and the whole procedure ends when both devices have received a `PAIR` message and the acknowledgement for the `PAIREND` response.

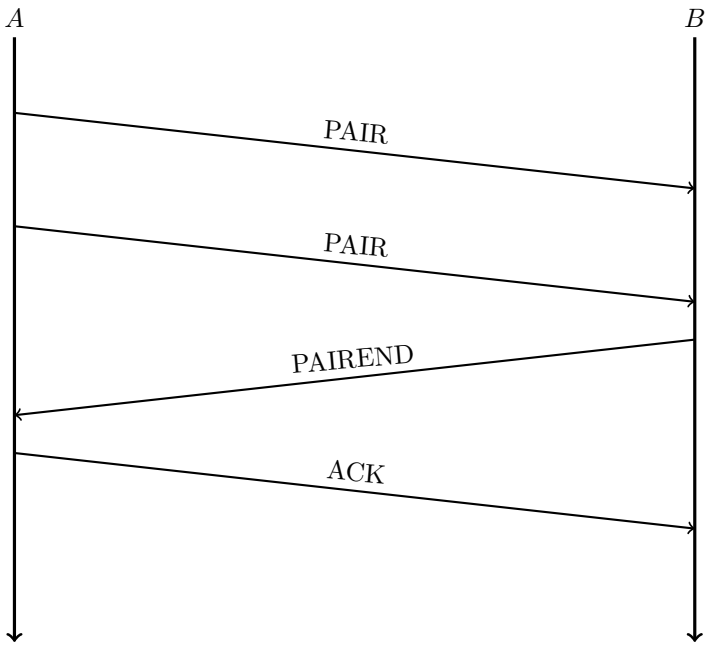

Figure 1: Pairing protocol from A's perpective.

2.2 State machine

A mote can logically be in one of three states:

- `BROADCASTING`: the mote broadcasts `PAIR` messages.
- `PAIRING`: the mote is pairing with another mote. No `PAIR` message is sent.
- `OPERATION`: the mote is exchanging information messages with the paired mote.

Initially, a mote is in the `BROADCASTING` state. Whenever a mote receives a message from a node with the same key as its, the mote goes in the `PAIRING` state. A mote in the `PAIRING` state goes in the `OPERATION` state as soon as

- The mote receives an acknowledgement for a `PAIREND` message.
- The mote receives a `PAIREND` message.

2.3 Missing alerts

This section describes how missing alerts are handled. A parent can receive a missing alert when:

- The child is too far.

- The child's bracelet turns off.

To handle both cases we decided to reset the parent device, whenever it receives a missing alert. This solution works well if the child's bracelet turns off (both bracelets are in the `BROADCASTING` state), however it requires more attention for the cases in which the bracelet is out of reach. In particular, the child's bracelet keeps sending `INFO` messages, however the parent's send `PAIRING` messages. To solve this problem, we decided to implement a keep-alive mechanism in which the parent's bracelet periodically sends a `KEEP_ALIVE` message so that, when the child's bracelet doesn't receive a `KEEP_ALIVE`, it realises that it went out of reach and it resets to the initial state (i.e., both devices are in the `BROADCAST` state). We know that, this solution is not power efficient, however, it allows to handle both cases in which a missing alert may rise. We also realised that, the same mechanism could be implemented with acknowledgements (i.e., the parent acknowledges `INFO` messages from the child), however our solution is more flexible since it allows to add some information in the `KEEP_ALIVE` message if required and it allows us to decide the rate at which messages are exchanged.