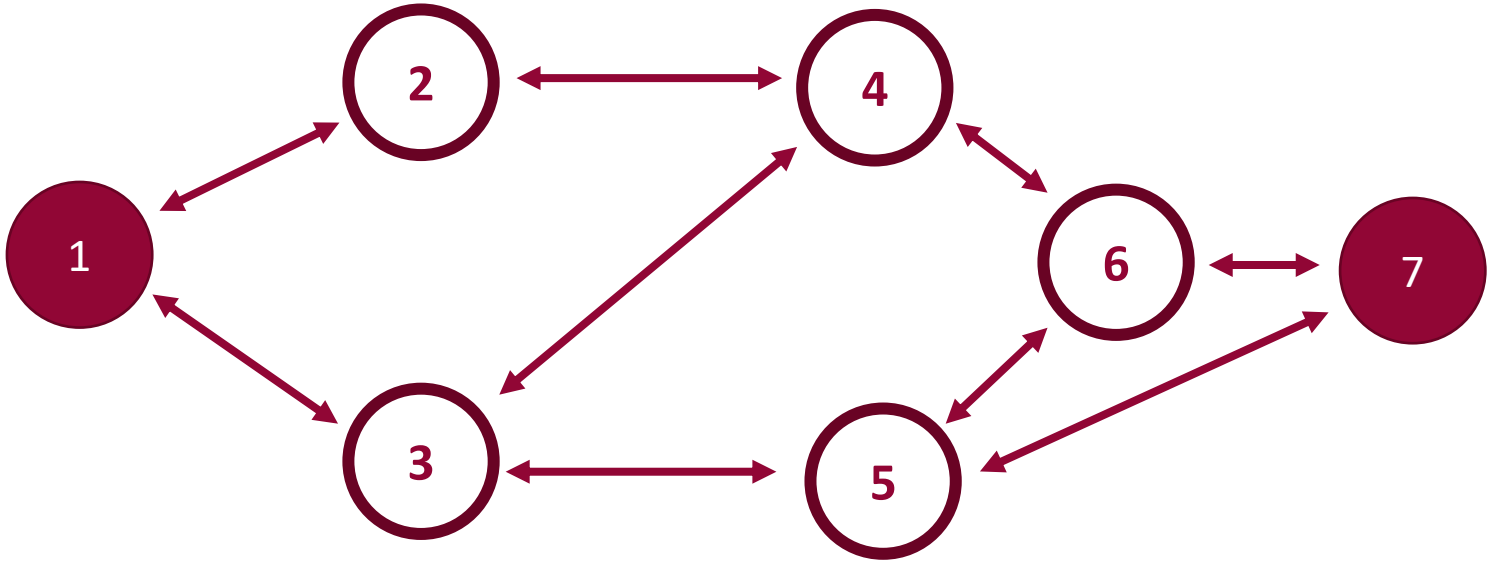
POLITECNICO
MILANO 1863

IoT Challenge #3

TinyOS and TOSSIM

What to do?

- Develop a TinyOS application to be simulated with **TOSSIM**
- Implementation of a simple routing protocol based on broadcasting **STARTING FROM OUR SKETCH (find the RadioRoute.zip in WeBeep)**
- Simulation of the network with the following topology (7 nodes)



ANTLAB

Routing protocol specifications (1)

- Before transmitting a message, each node checks its **routing table** to see if a route is present for the selected destination:
  - If the destination is present, the message is forwarded to the **next hop** indicated in the routing table.
  - Otherwise a ROUTE_REQ message is sent in **broadcast**, containing the selected destination in the message.

Routing table example (Node 1)

Destination	Next Hop	Cost
4	2	2
5	3	2

ANTLAB

Routing protocol specifications (2)

If I receive a ROUTE_REQ I should:

- **Broadcast** it if the ROUTE_REQ is a new one (i.e. requesting for a node not in my routing table and not me)
- If I am the node requested, I reply in **broadcast** with a ROUTE_REPLY, setting the ROUTE_REPLY cost to 1
- If the node requested is in my routing table, I reply in **broadcast** with a ROUTE_REPLY, setting the ROUT_REPLY cost to the cost in my routing table + 1

If I receive a ROUTE_REPLY I should:

- If I am the requested node in the reply:
  - Do nothing
- If my table does not have entry or if the new cost is lower than my current cost:
  - I update my routing table
  - I forward the ROUTE_REPLY in **broadcast** by **incrementing** its cost by 1
- Otherwise: Do nothing

Message formats

Data messages:

Type = 0
Sender - integer
Destination - integer
Value – integer

Route Request messages:

Type = 1
Node Requested – integer

Route Reply messages:

Type = 2
Sender - integer
Node Requested – integer
Cost – integer

USE A SINGLE MESSAGE STRUCT in the header file (RadioRoute.h), **NOT 3 DIFFERENT** !

ANTLAB

What to do (2)

Every time a node receives a message (of any type), it updates the status of the LEDs as it follows:

Take the leader person code: i.e. **10692911**

Starting from the first digit of your person code, in a round robin cycle, **toggle** the LED with index led_index = **digit modulo 3**

At each message, the digit is changed, (for the first request take the first, then the second and so on) going back to the first digit after it reaches the end

Example:

First msg: digit = 1 1 modulo 3 = 1-> I update the LED with index 1 (**toggle** led1)

Second msg : digit = 0 0 modulo 3 = 0 -> I update the LED with index 0 (**toggle** led0)

Third msg : digit = 6 6 modulo 3 = 0 -> I update the LED with index 0 (**toggle** led0)

…. At the 9th message you go back to the first digit

Simulation settings in TOSSIM

- Set the topology of the simulation (topology.txt) as in the **slide 2**
  Note that links are all bi-directional (**use -60.0 dBm** as gain for all entries)

- **All the routing tables are empty at the beginning of the simulation,** they should be filled with proper ROUTE_REQ and ROUTE_REPLY when needed

- Set all the nodes to boot at time t=0.

- **5 seconds after** its Radio is ON, **Node 1** wants to transmit a data message with **value 5** with destination **Node 7**. Since its routing table is empty, it will issue a ROUTE_REQ.
  At the reception of the ROUTE_REPLY in node 1, the actual DATA message should be sent (hop-by-hop based on routing table, **NO BROADCAST!!!**
  If more ROUTE_REPLY for Node 7 are received in node 1, **only send DATA the first time**
  (only 1 DATA is sent out from node 1)
- Use the same **meyer-heavy.txt** noise file as the RadioToss project

ANTLAB

Important Note (to do 3)!

- To simplify the outcome and reduce the number of messages, **limit the number of ROUTE_REQ and ROUTE_REPLY that a node sends to 1** **(1 ROUTE_REQ and 1 ROUTE_REPLY)**

- If a node has already sent one ROUTE_REQ, it **should not** send other ROUTE_REQ msgs

- If a node has already sent one ROUTE_REPLY, it **should not** send other ROUTE_REPLY msgs

- Enable debugs for all the important events you think are useful to report
- IMPORTANT: Print debug for leds at every message reception

Challenge deliverables

What to deliver:

- A **PDF** report containing the explanation of the code logic.
  Organize a **clean** report! Very bad reports will be penalized
- **TinyOS Project** folder containing **all required files**
  COMMENT AND INDENT YOUR CODE (not clear codes are penalized)
- **TOSSIM LOG (export in a txt file)**
- LED status history for node 6 **as explained in the following slide**
- YOUR NAMEs and **PERSON CODEs**

The files should be included in a ZIP which should be named as follows:

2-teams: <personcode1>_<personcode2>.zip
Single: <person_code>.zip
E.g. 10692911.zip or 10692911_10692912.zip

ANTLAB

Challenge delivery: HOW?

How to deliver?

- Upload the files in a zip archive as .zip file on the **folder #3** on WeBeep "Assignments" folder

- Fill this **form** with the LED status history of Node 6
  The format of the LED status is the following, with one entry every time one of the LED of Node 6 changes status (separated by a comma):

  000,111,010,100,101,111

  Where each of the three digits indicates the status of the LED

  So 010 means LED0 OFF, LED1 ON, LED2 OFF

PLESE FOLLOW THE FORMAT CORRECTLY!

ANTLAB

For two-people teams:

- Choose your team leader and name the file as:
  <leader_personcode>_<other_personcode>.zip

- **Only the teamleader** should upload the challenge in WeBeep
  Do note upload the same challenge twice

- *Can I take the challenges with the other class students (Prof. Redondi)*?
  YES, **but** only the team leader should upload the challenge in WeBeep

Delivery Deadline

- **STRICT** Deadline:

  June 1st, 2023 h 23.59 (FIRM)

- Max 2 people
- Up to **1.5** points

Good Luck!

ANTLAB