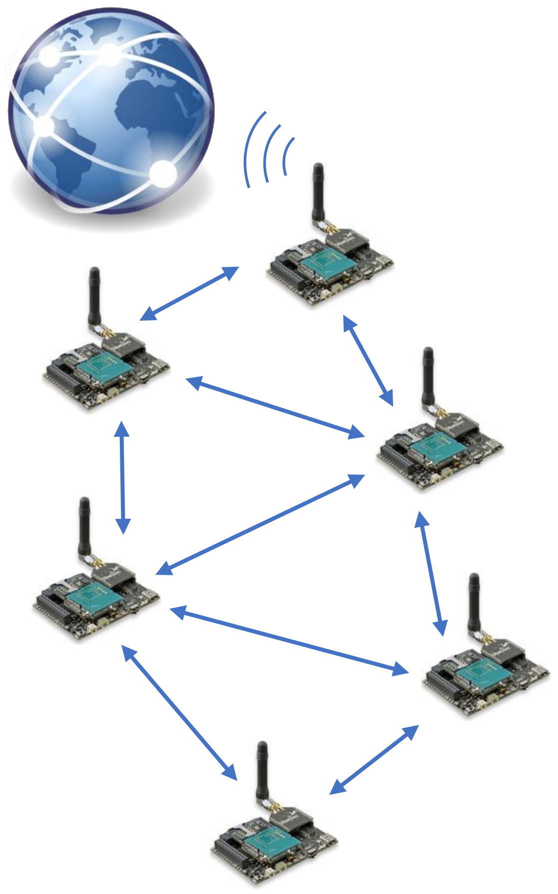
Low-power Wireless Networking for the Internet of Things

Lab9: Ultra-wideband ranging

Matteo Trobinger (matteo.trobinger@unitn.it)

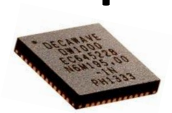
Credits for some slides to:

Enrico Soprana, Davide Vecchia

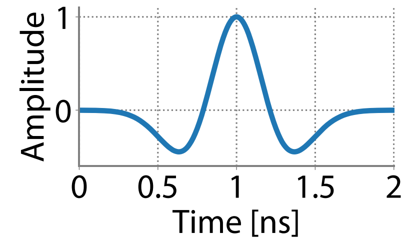


Ultra-wideband (UWB) in a Nutshell

Short pulses ⟶
- Large bandwidth (≥ 500MHz)
- Low power spectral density



Small, cheap, low-power chips

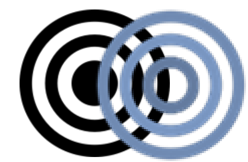


Advantages for Ranging-Localization & Communication

Outstanding time resolution (and ranging performance)

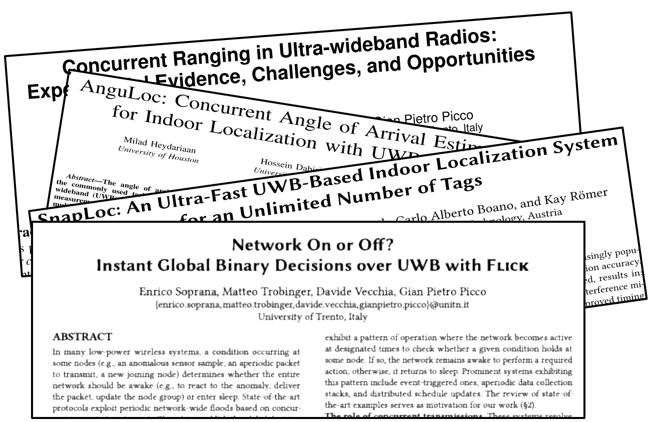
High data throughput

Multipath and noise resilience



Limited co-existence issues

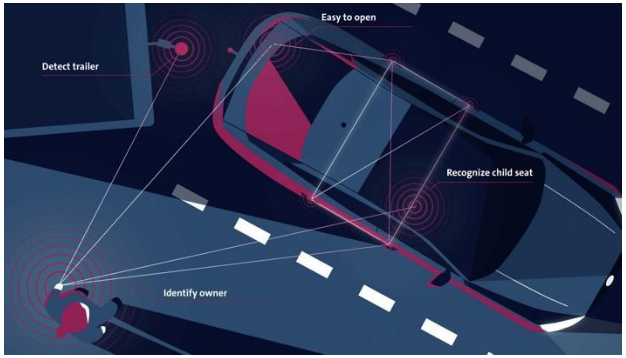
Rising interest



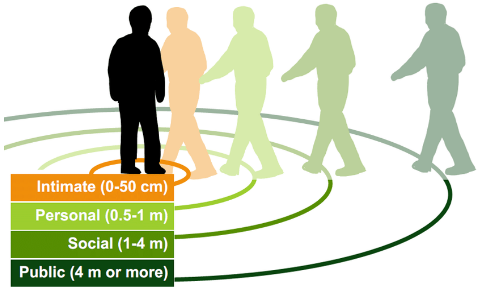
A few UWB Applications

Smart homes

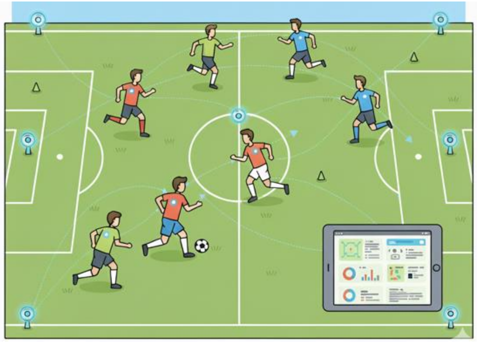
Retail



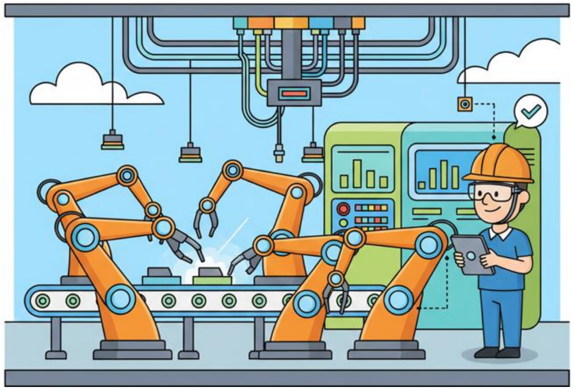
Secure keys and driving assistance



Proxemics and social studies

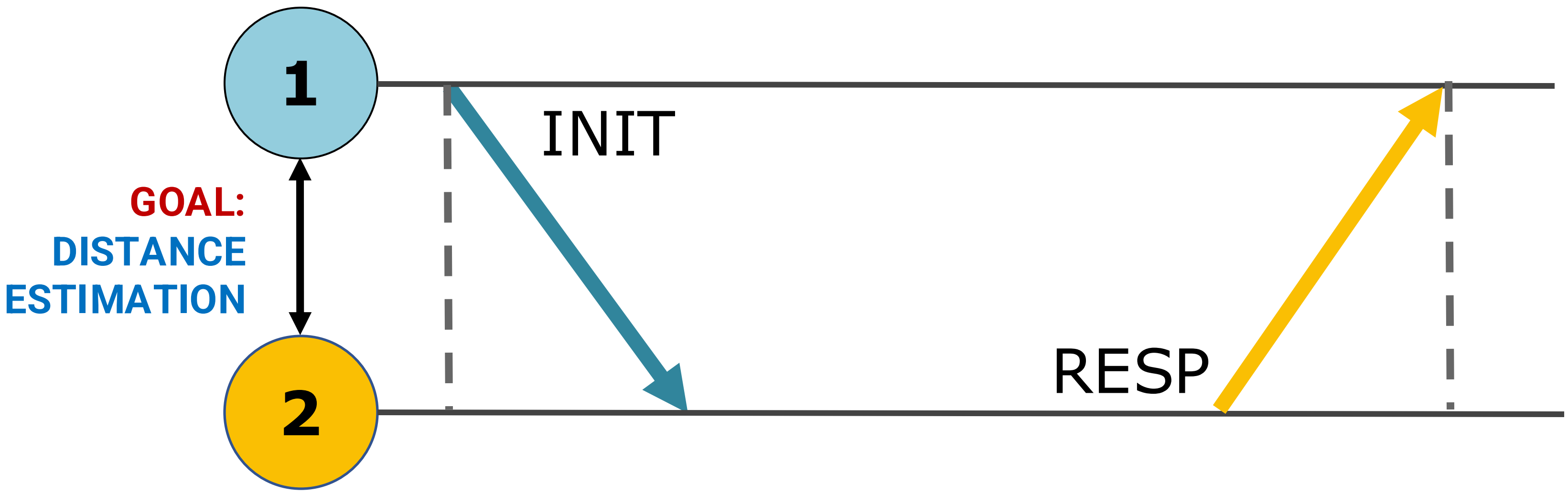


Sport analytics



Industrial automation & asset tracking

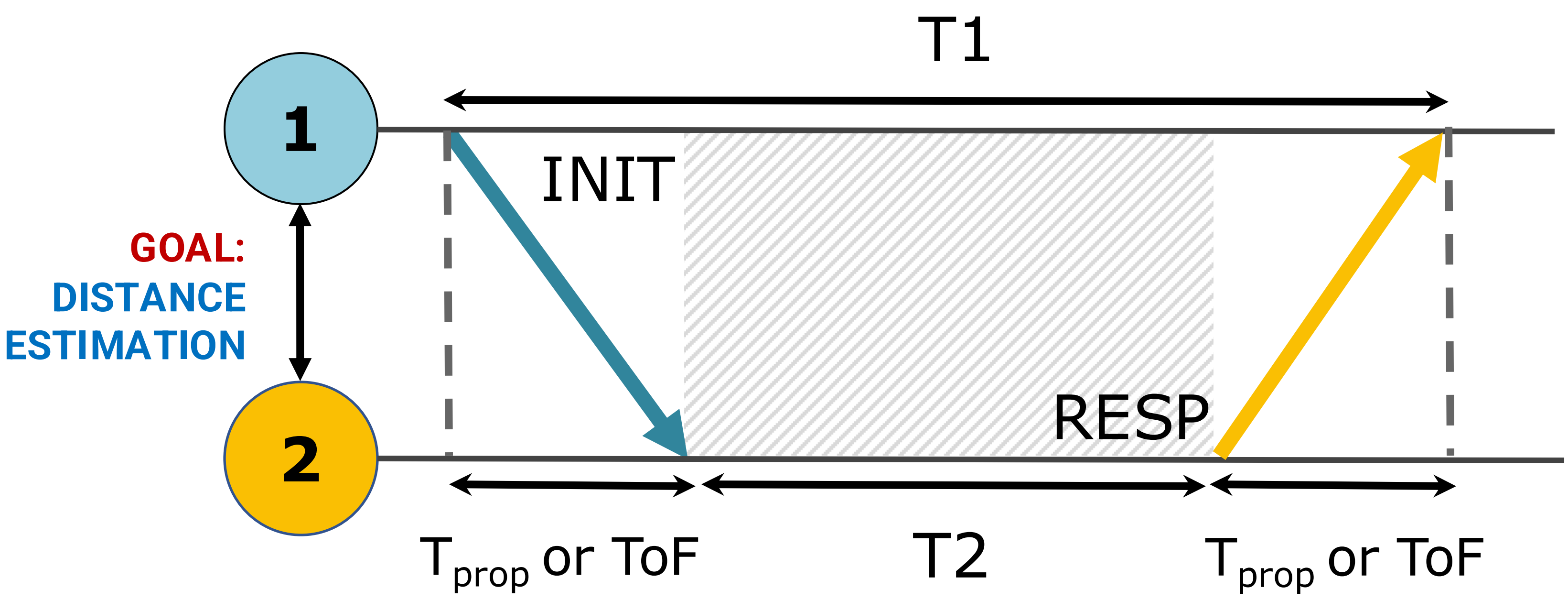
Single-sided Two-way Ranging



The procedure entails the exchange of only 2 messages

- `INIT`, sent by the device that needs to estimate the distance
- `RESP`, the reply from the recipient of the `INIT` message

Single-sided Two-way Ranging

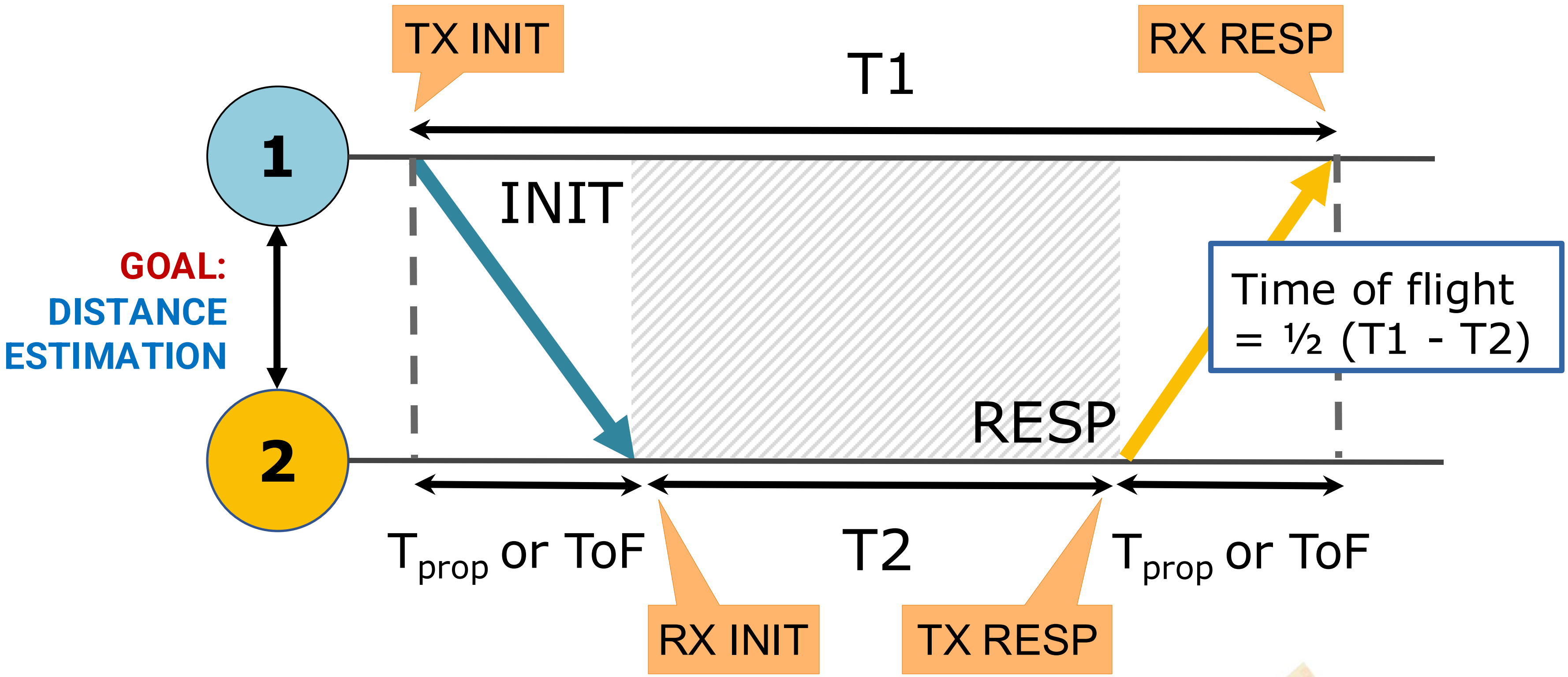


Time of flight (ToF) $= \frac{1}{2}\,(T1 - T2)$

Distance = ToF × Speed of light

Challenge: How to compute T1 and T2?

Single-sided Two-way Ranging



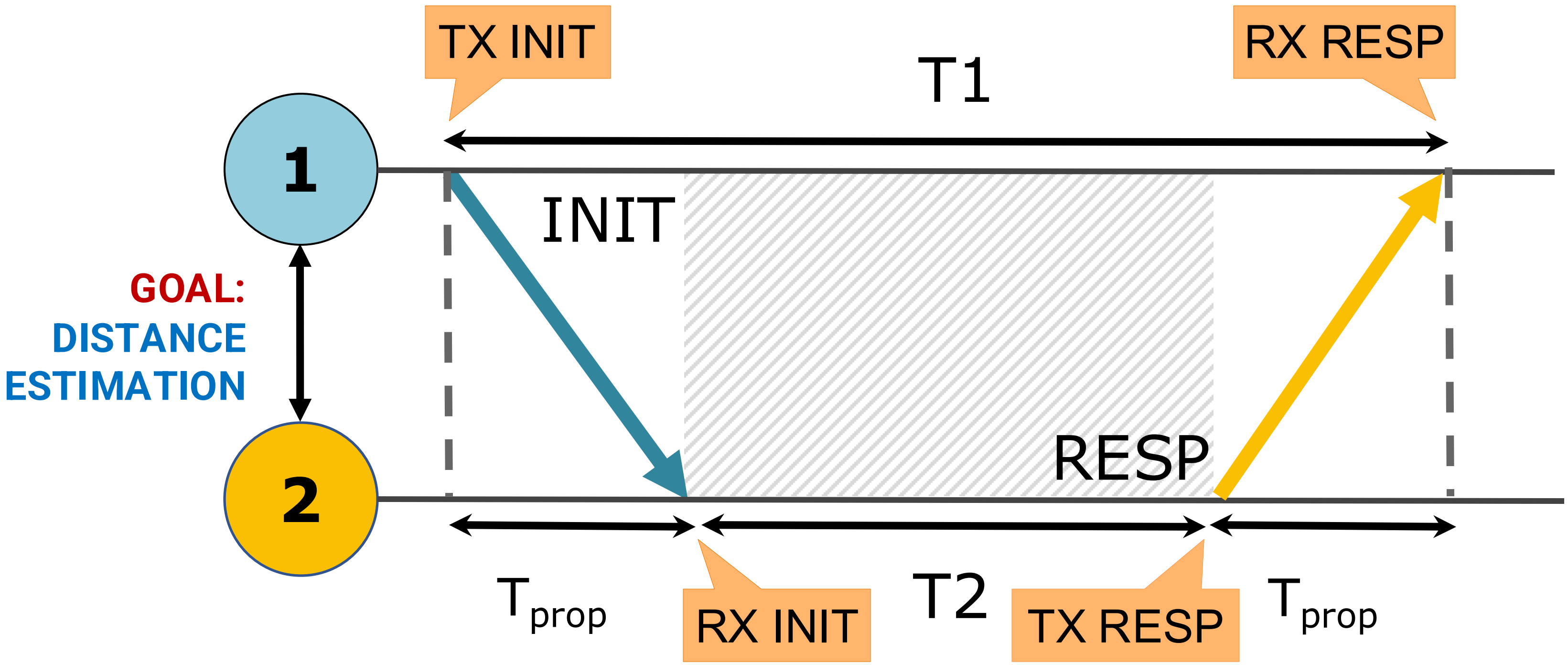
T1 = RX RESP – TX INIT

T2 = TX RESP – RX INIT

EVB1000

Based on accurate timestamps enabled by UWB physical layer properties

Two-way ranging with DW1000

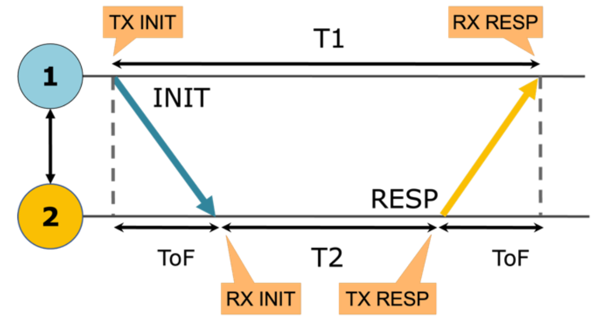


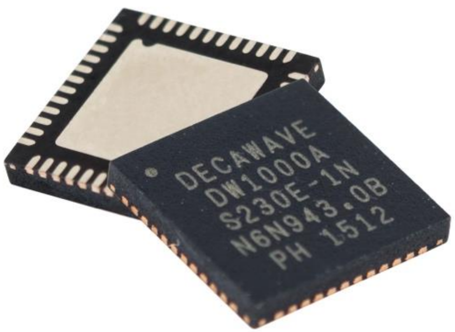
To let node 1 precisely estimate T2, node 2 must embed in the `RESP` message `RX INIT` and `TX RESP`

The `RESP` <u>transmission timestamp</u> must be embedded in the packet… **how can the sender know the TX timestamp in advance?**

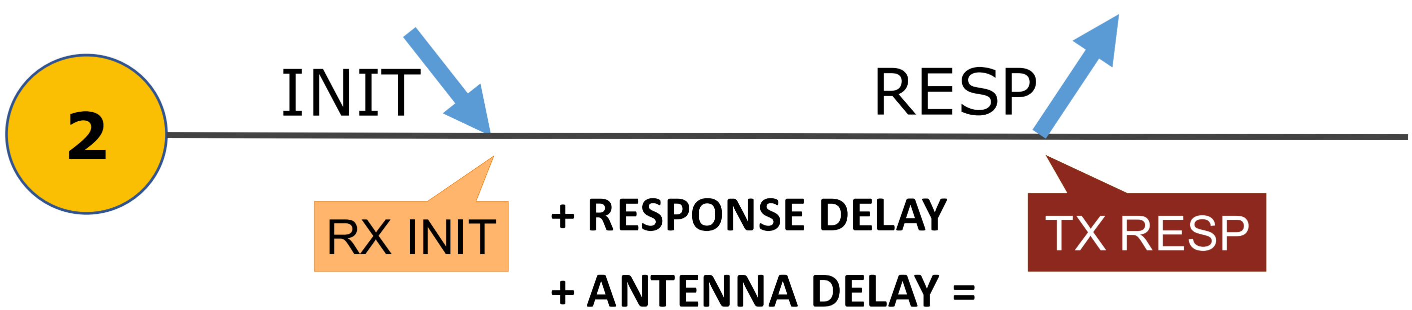
Two-way ranging with DW1000

The `RESP` transmission timestamp must be embedded in the packet… **how can the sender know the TX timestamp in advance?**

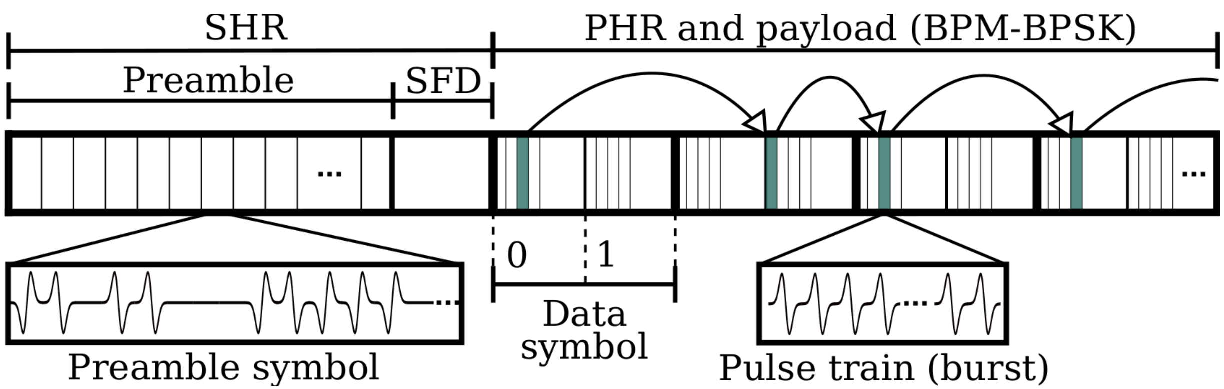




The DW1000 can schedule transmissions to start at a specified moment in the future <u>with a resolution of 8 ns</u>



How does ranging work?
A step back: UWB physical encoding

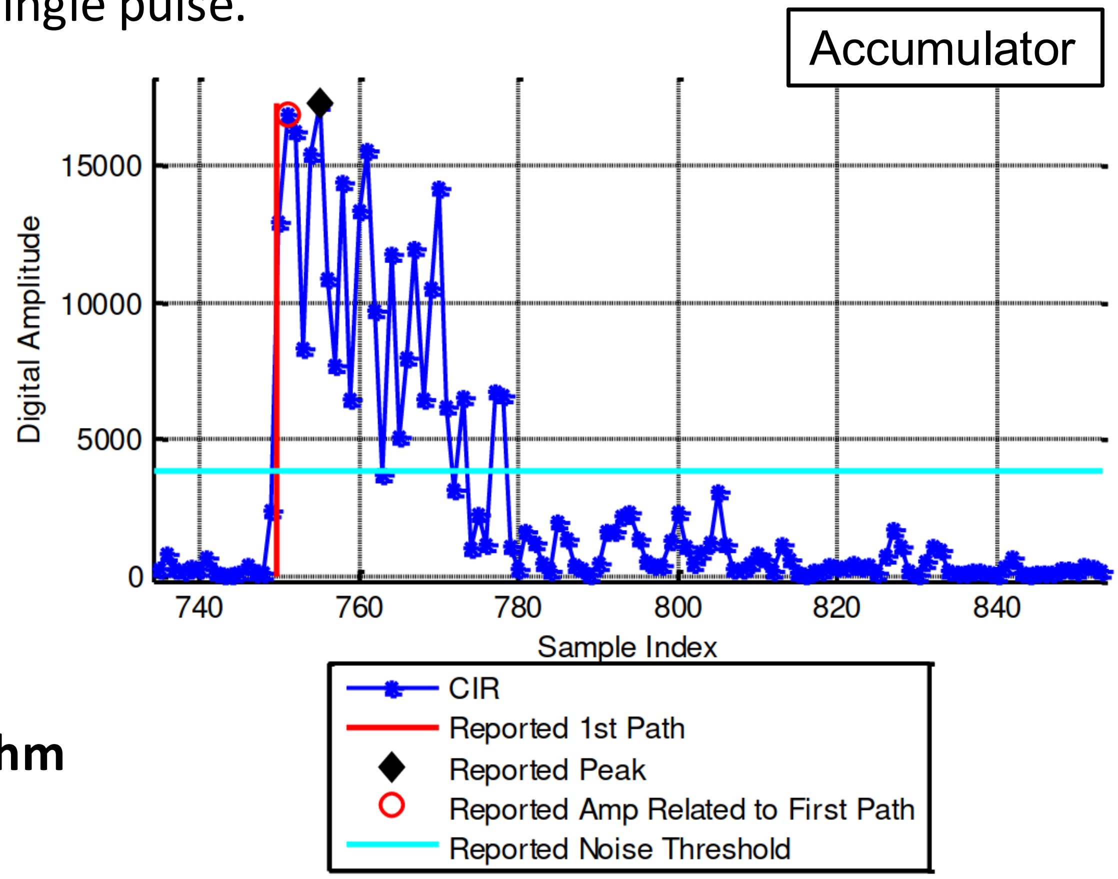


- **Preamble**: PLEN repetitions of the preamble symbol. Meant for signal detection and synchronization.
- **Start of Frame Delimiter (SFD)**: special pulse sequence that ends the preamble.
- **RMARKER**: the first pulse after the SFD, at the beginning of the physical header (PHR). It is <u>the pulse that is timestamped</u>.
- **Payload**: modulation based on pulse positions and phase (BPM-BPSK). Includes Forward Error Correction (FEC) mechanisms.

First path and timestamping

The receiver cross-correlates the incoming signal with the known reference preamble signal and stores the results in the dedicated accumulator. Since preamble symbols have perfect periodic auto-correlation, the result in the accumulator should be a single pulse.

In reality, **reflections** are also detected and contribute to the amplitudes in the accumulator.



The DW1000 searches for **the first peak above a threshold (first path)** and then **refines the estimate with a proprietary algorithm** (likely based on peak amplitude and slope).

Non-Line-of-Sight

https://www.decawave.com/wp-content/uploads/2018/10/APS006_Part-3-DW1000-Diagnostics-for-NLOS-Channels_v1.1.pdf

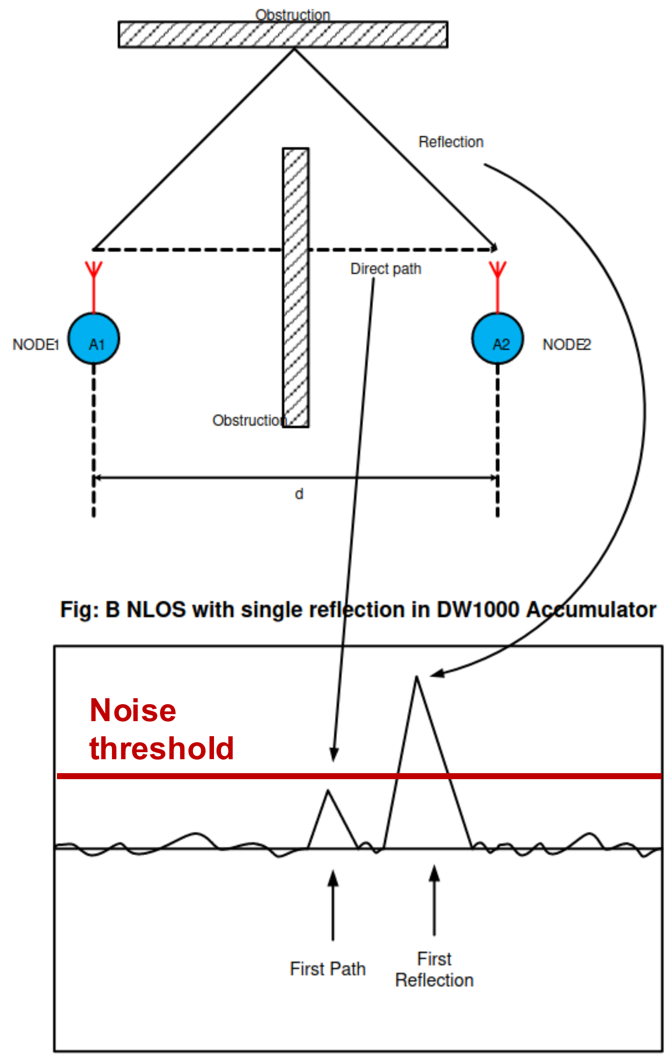
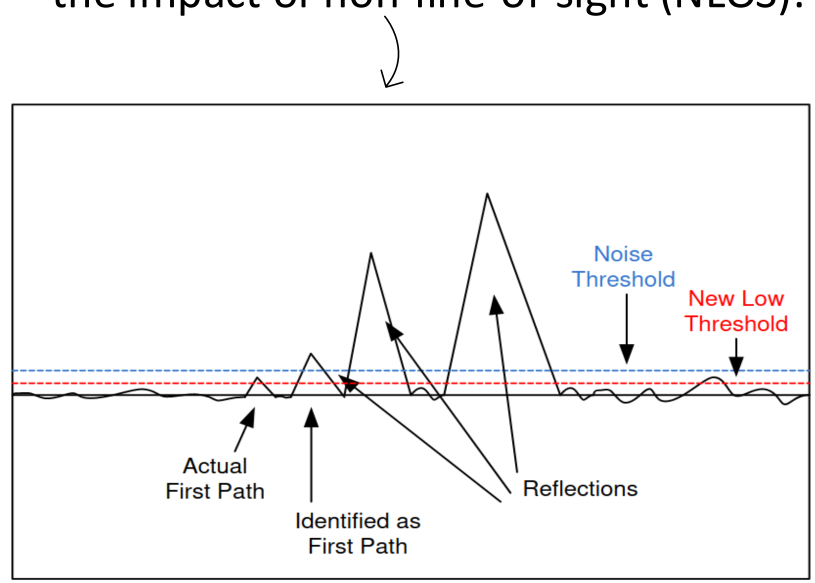


Fig: B NLOS with single reflection in DW1000 Accumulator

Non-Line-of-Sight (NLOS) can cause the radio to detect a reflection instead of the first path leading to **large (meters level) errors**

Possible solution:
Use radio configurations to reduce the impact of non-line-of-sight (NLOS).



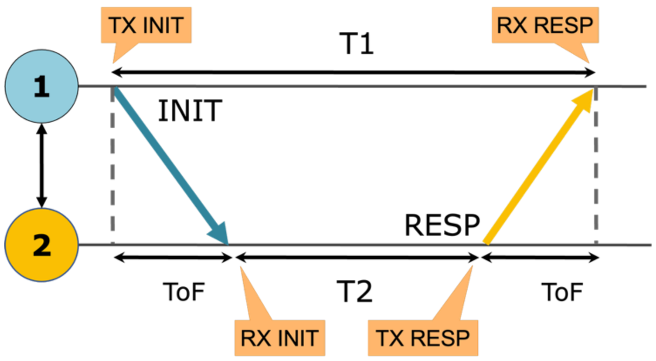
Ranging accuracy

Several factors affect ranging accuracy:

1. Obstacles causing *reduced signal strength in line-of-sight* and/or *powerful reflections*.
2. Clock drift between initiator and responder.
   A long delay before the reply affects timestamps and distance estimation in turn!
3. Interference on the channel.
4. Different antenna delays among nodes.

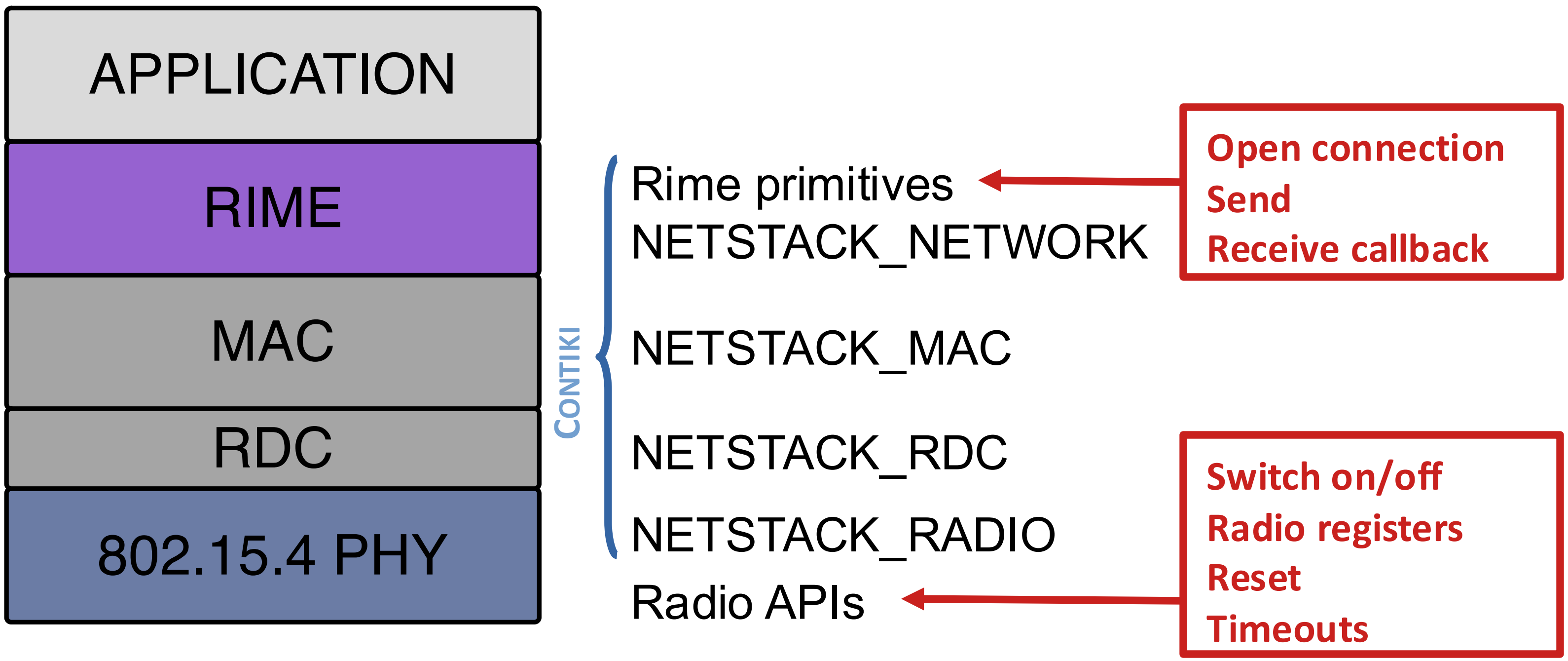
Some solutions:

1. Use radio configurations to reduce the impact of non-line-of-sight (NLOS).
2. Estimate the drift between nodes, and use it to correct ranging results.
3. Ranging filtering, channel hopping, STS.
4. Perform per-node delay calibration.



Exercise: Estimating distances with TWR

In the previous labs, we used Rime primitives to send and receive data



In this lab, we will implement ranging **using the DW1000 radio APIs**

Code template

Download and unzip the provided code

```
$ unzip Lab9-exercise.zip
```

Go to the code directory

```
$ cd Lab9-exercise/uwb-rng-radio-exercise
```

- DEPT_evb1000_map.csv
- experiment.json
- Makefile
- project-conf.h
- rng_eval.py

- rng-init.c
- rng-resp.c

- rng-support.c
- rng-support.h

TO-DOs: implement the initiator and responder(s) firmware

TO CHECK: Understand how to exploit the abstractions we provide

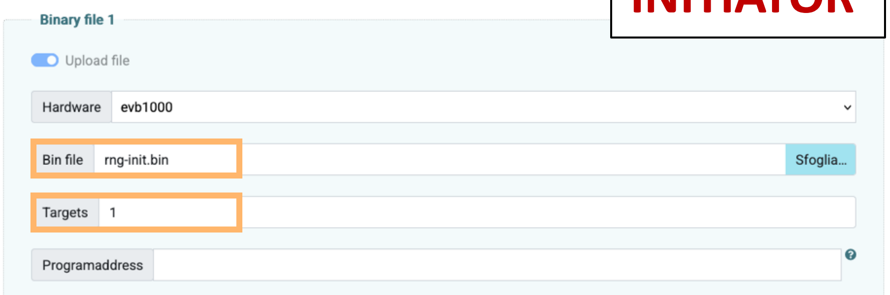
To Compile

```
$ make TARGET=evb1000
```

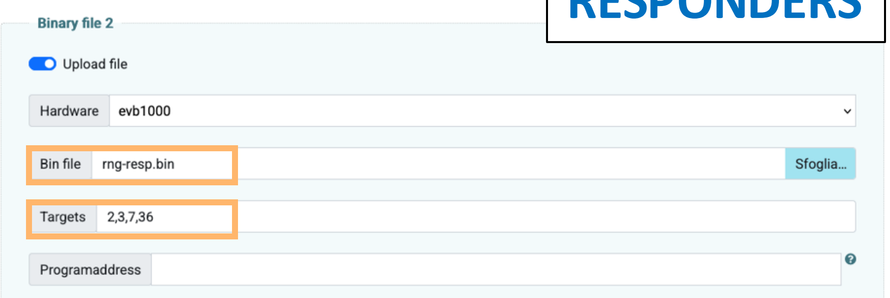
Once it compiles without errors, test it in the testbed!

Choosing correct targets

INITIATOR



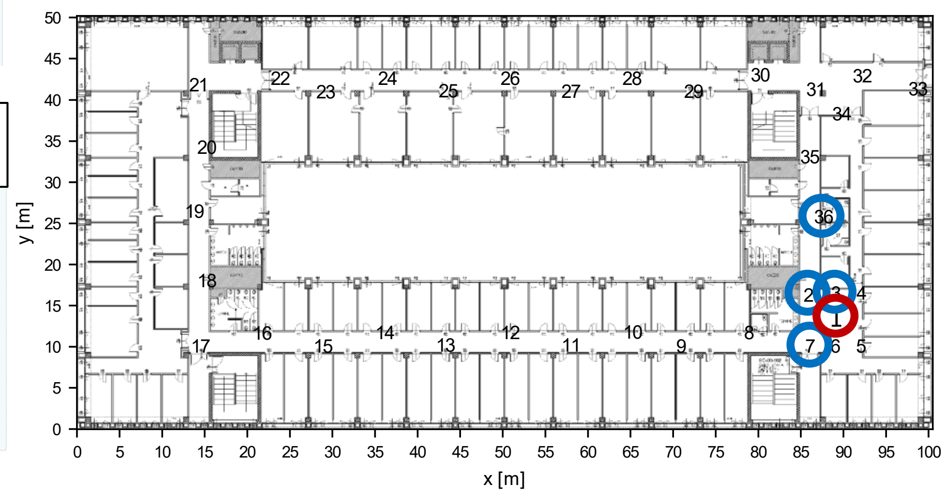
RESPONDERS



Today you will develop **2 firmwares:**

- Initiator
- Responders

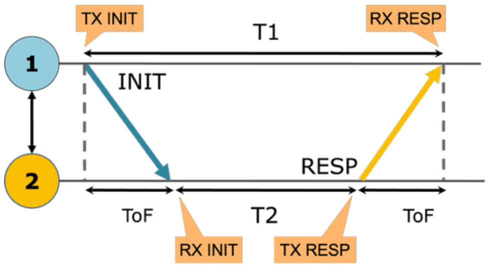
Remember to upload them correctly!
(select the correct binary and targets)!



If you use the testbed client, you can directly exploit the `experiment.json` file in the code template!

NB: To debug your code, please run **<u>short experiments</u>** (e.g., 2 minutes long)

Exercise TO-DOs



The initiator code (`rng-init.c`) is almost complete

1. Get `RX INIT` and `TX RESP` from the `RESP` message (check `rng-support` files)
2. Compute the ToF based on the acquired timestamps
3. Find the distance in meters and millimeters, and print it

The responder code (`rng-resp.c`) is missing the transmission of the `RESP` message

Use `rng-support` functions to

- Acquire `RX INIT` and determine `TX RESP` timestamps
- Embed them in the packet (`resp_msg`)
- Transmit the packet with `start_tx()`

Exercise – Analyze Ranging Data

Run experiments in CLOVES, wait for the logs and:

- Analyze the obtained ranging measurements with the provided `rng_eval.py` script

  `$ python3 rng_eval.py job_ID/job.log`

  Pandas module required… **Suggestion:** run it on your laptop!

- Example output:

  `Ranging error [1->7] -197 (dist 4547 true dist 4744)`

  Error in **millimetres.** Computed based on the known positions (and distances) of nodes in CLOVES

Check the results and try to answer these questions:

1. Does the ranging performance change if the response delay (`RESP_DELAY`) is modified?
2. Do obstacles have an impact on the ranging performance?
3. Does the ranging performance change depending on the position of the nodes? Try other setups, e.g., Init = Node 6; Resp = [2, 7, 8, 36]