

# Mioty, telegram splitting for high density



**Paul Pinault**

Blog/contact : [www.disk91.com](http://www.disk91.com)

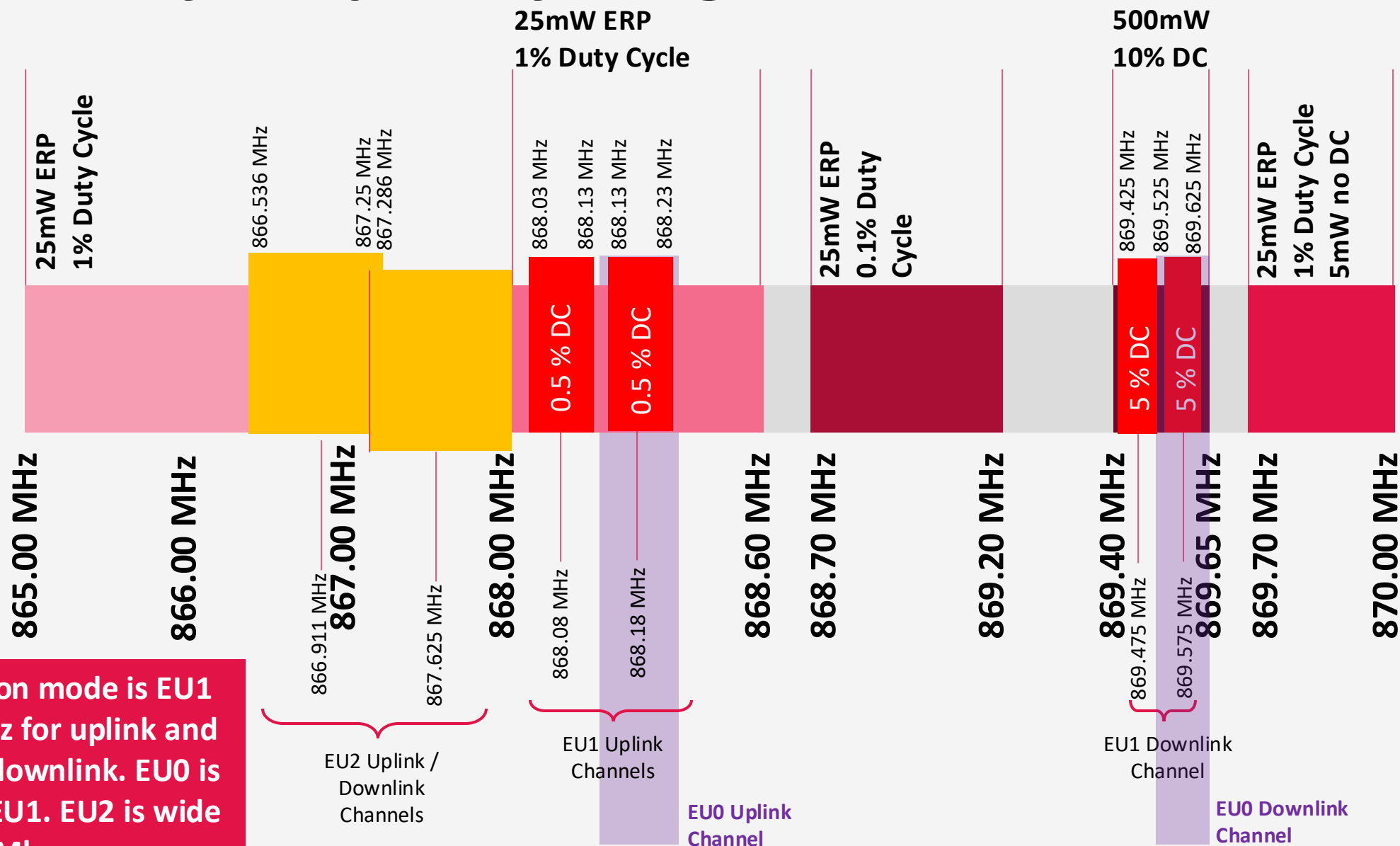
Twitter : @disk\_91

YouTube: <https://www.youtube.com/c/PaulPinault>



Mioty defines different mode of use (Narrow, Standard and Wide)

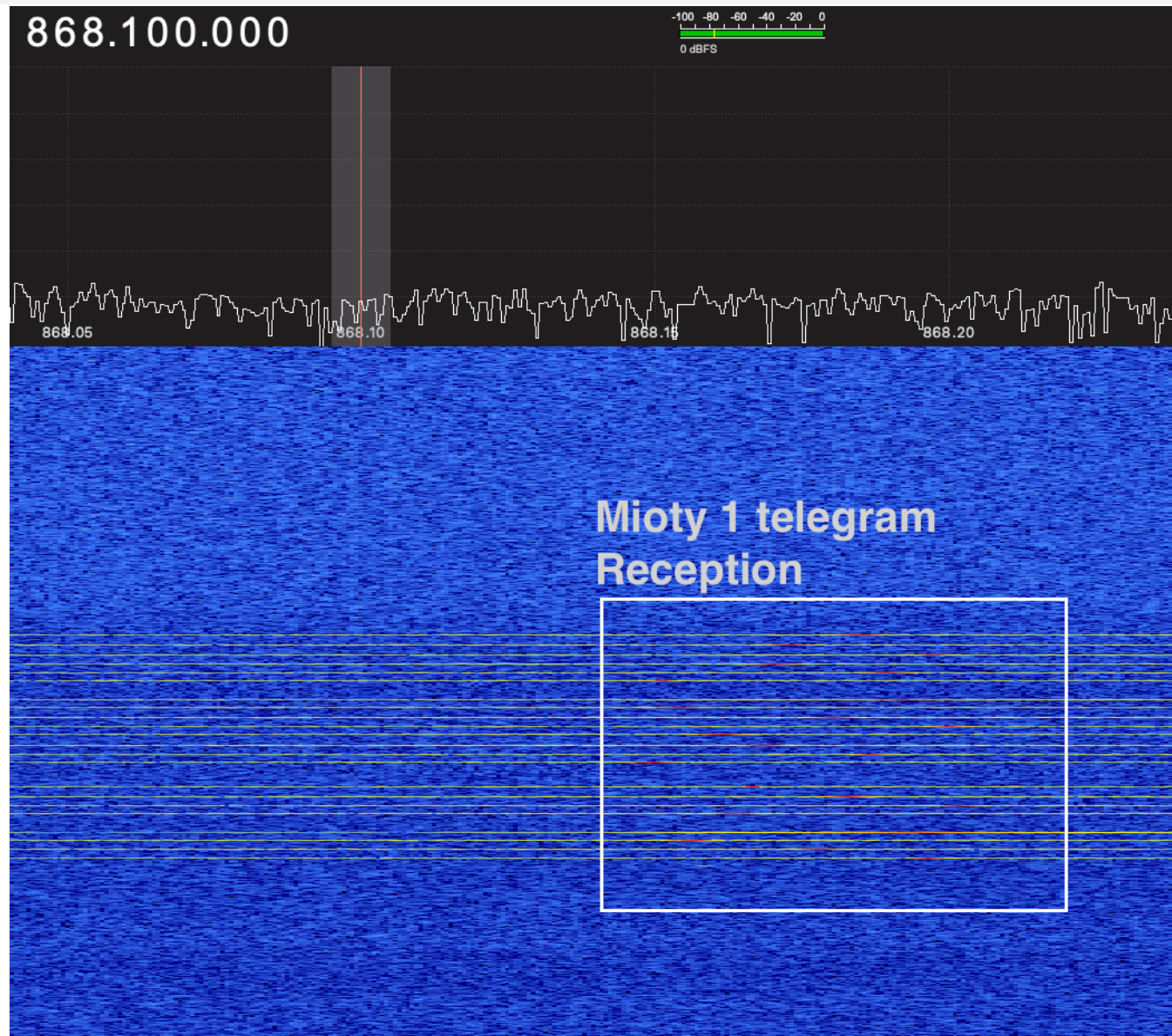
# Mioty frequency usage



Most common mode is EU1 using 200KHz for uplink and 200KHz for downlink. EU0 is a subset of EU1. EU2 is wide band with 2Mhz occupancy



- > Packet (telegram) split into a minimum of 24 burst (core frame)
- > Each burst is sent on a different frequency (up to 34)
- > Each burst is separated from the others by a certain time
- > Each burst includes 1/3 of sync data (but no identification)
- > Information is triplicated to support collision
- > Creates time and frequency diversity with redundancy



Each of the burst is transmitted at 2.38KHz. There is also a long-range mode at 400Hz (less common)

Between each of the burst a variable inter-burst silent is part of the pattern (between 360ms and 600ms)

The minimum transmission time about 8s total with 363ms over-the-air radio emission.

Using the low latency pattern will reduce the inter-burst time and accelerate the transmission to 800ms.

# Telegram Splitting

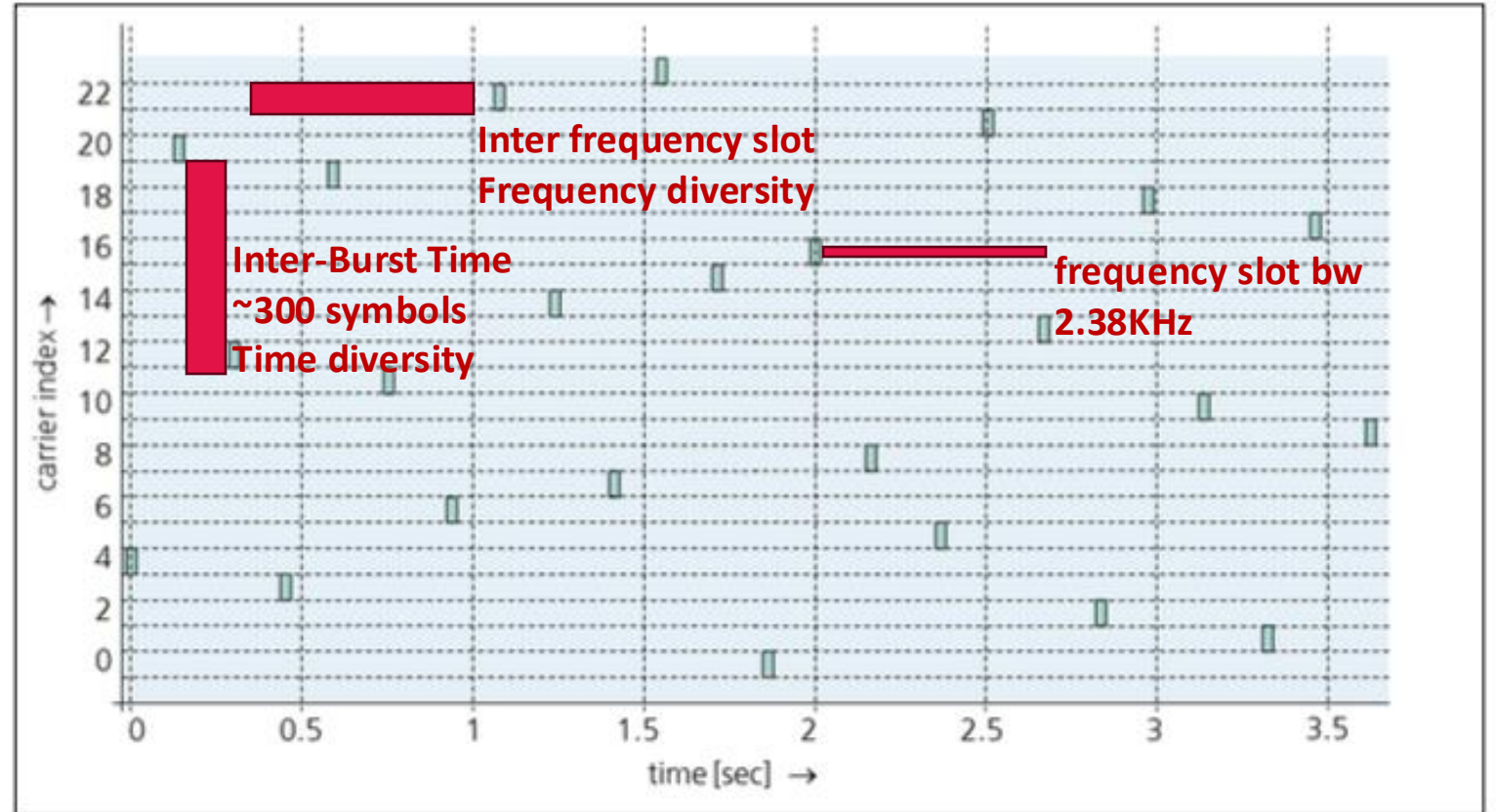
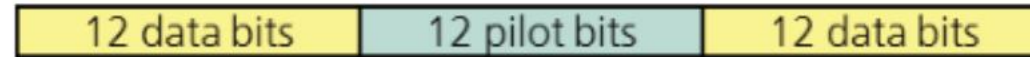


Figure 7: Example of a time-frequency pattern for the TS-UNB uplink core frame



Pilot sequences:

core frame: 0111 0100 0010

extension frame: 0100 1111 1010

Every burst is a 36 bits sequence

2x12 bits of data

12 bits of synchronization (50% overhead)

# How to receive frames ?



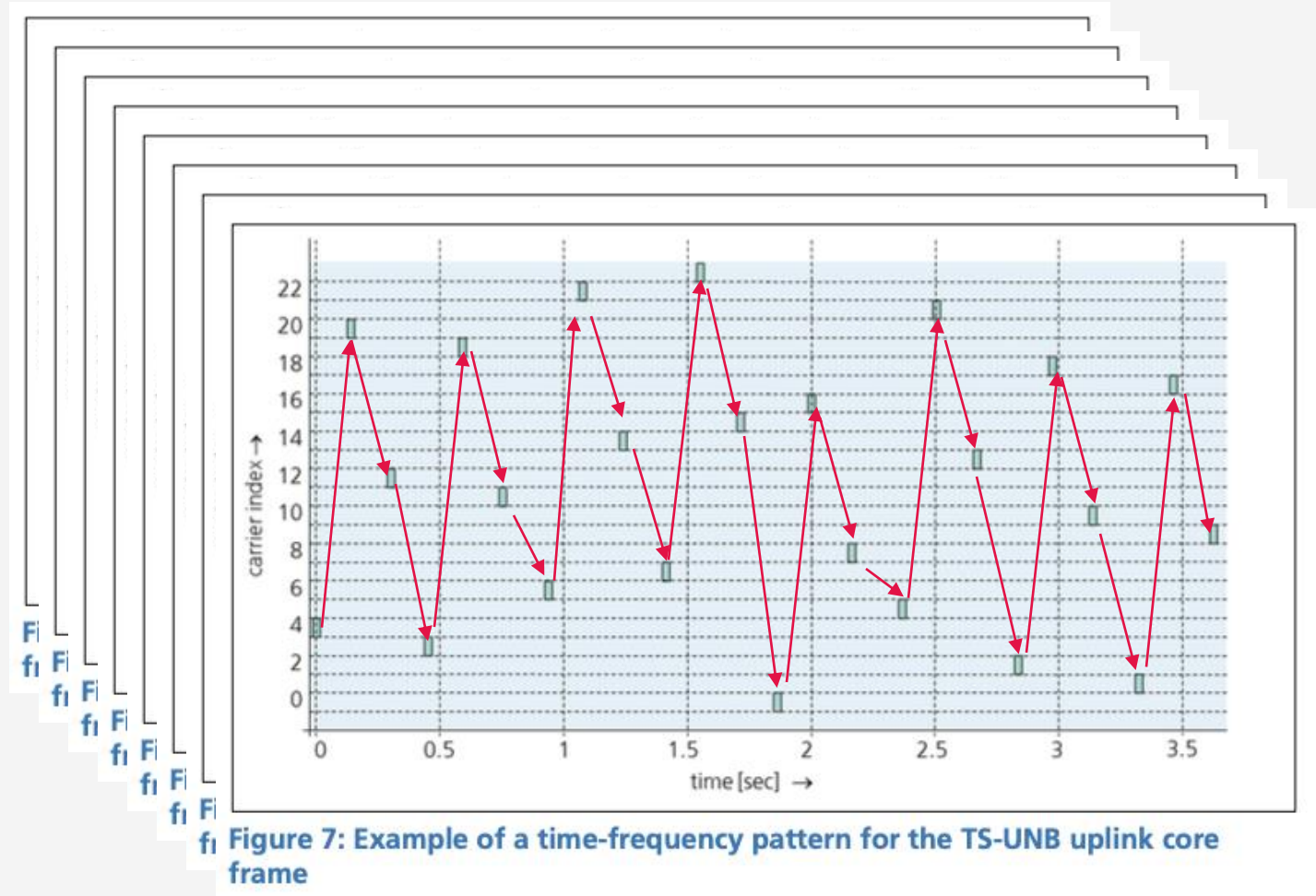
# Freq hopping patterns

8 different patterns are used for a telegram transfer (based on CRC)

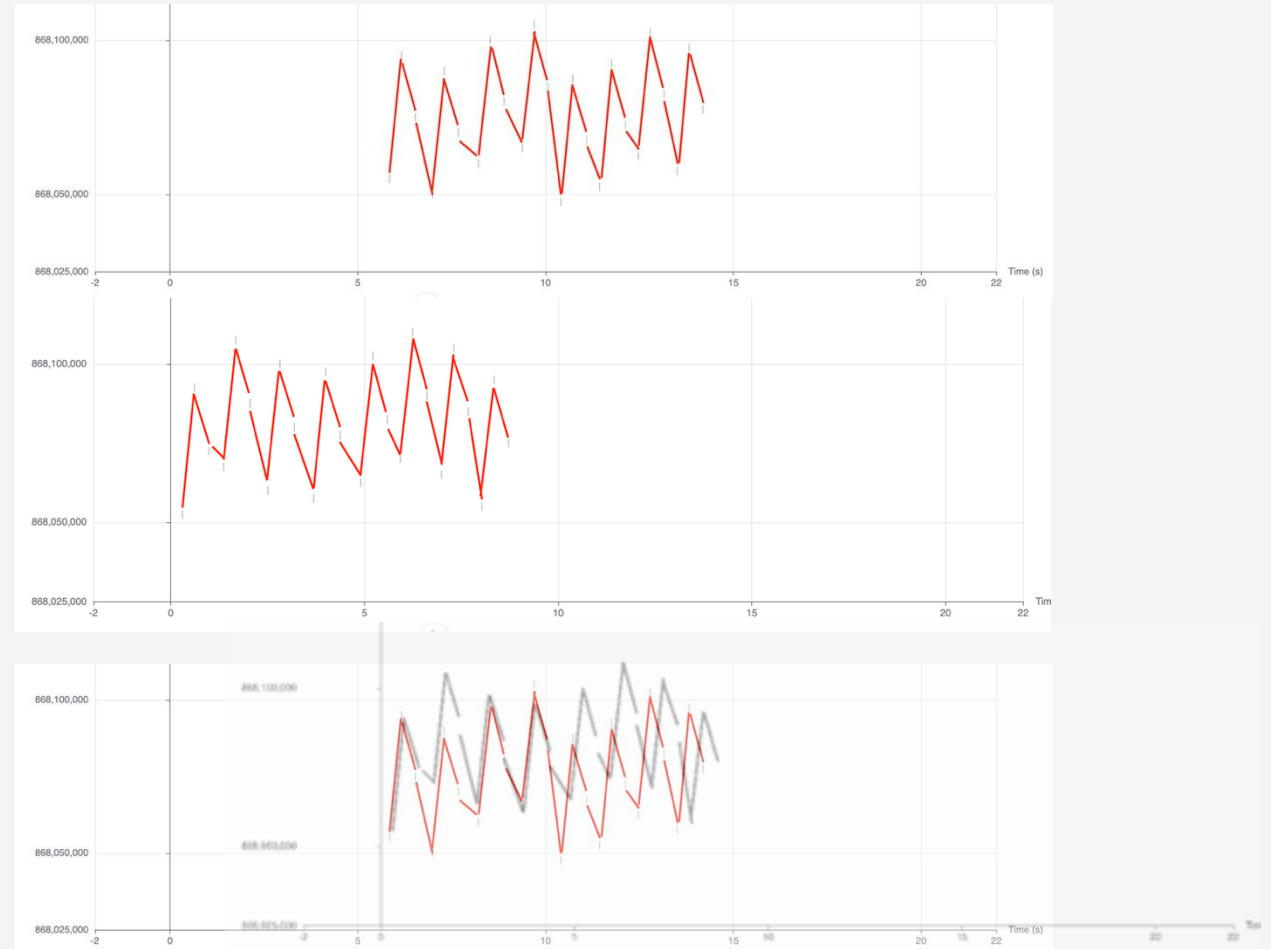
+1 extra pattern to reduced inter-burst time allows faster delivery (low latency mode)

Frame with repeat flag set (primary & repeats) will use a different patterns.

In dual bank mode (like EU1) a device is switching from a 100KHz channel to the other based on CRC bit 0



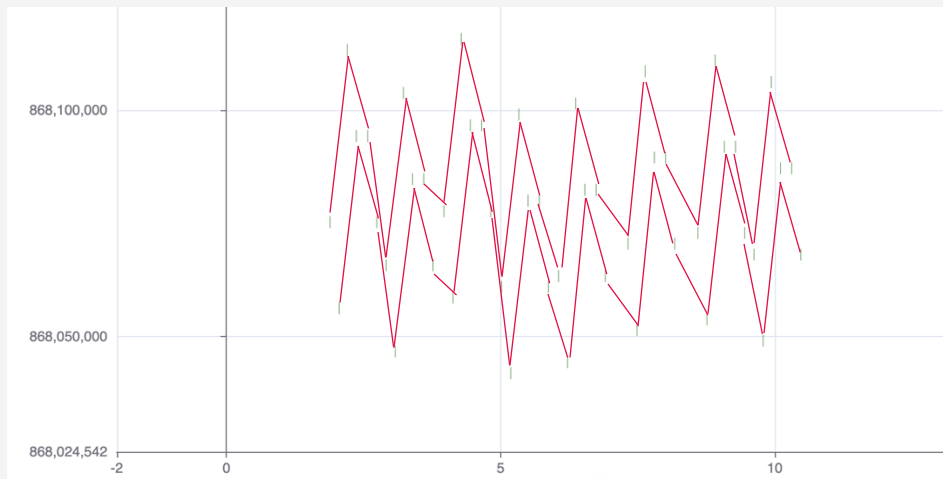
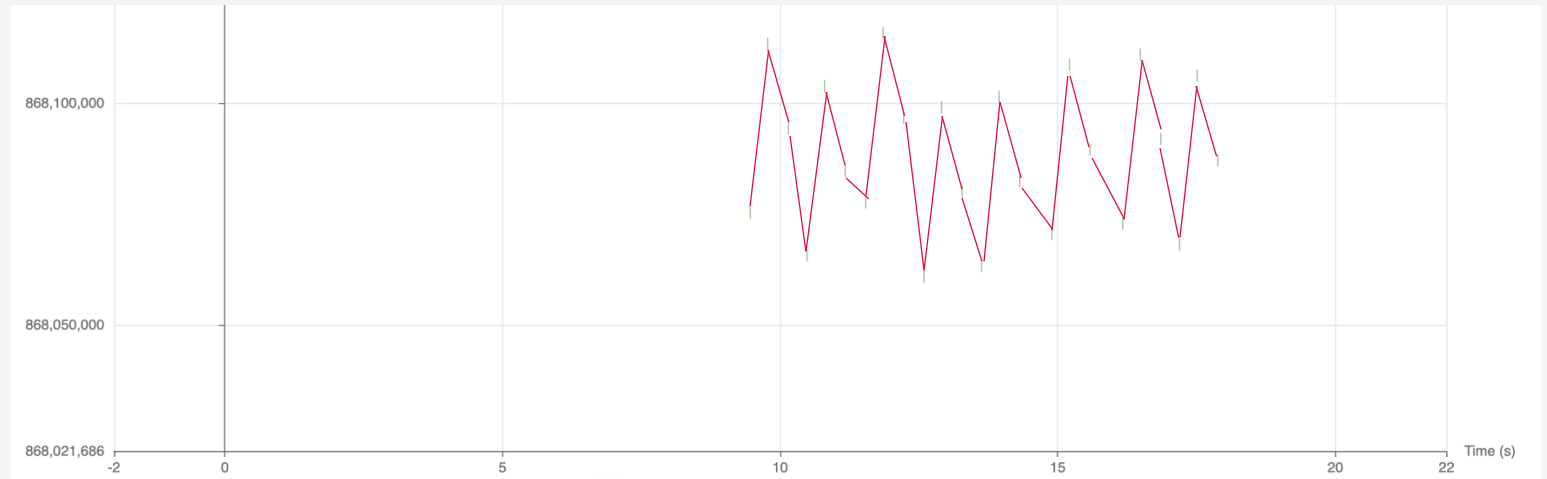
- 2 different patterns will have different frequencies and inter-burst sleep timetable.
- The **core frame** (first 24 mandatory burst) never reuse a frequency in the pattern. Longer frame will reuse the frequency slots.
- For all patterns, there are some common inter-burst durations:
  - 330ms for first %3 burst
  - 387ms for second %3 burst
- And a pattern specific durations from 360 to 620ms for the 3<sup>rd</sup> split of the group of 3.





- 1 single pattern can be used with different offset frequencies; device randomly select this offset.
- The Offset is from  $\pm 1$  or  $\pm 5$  channels based on CRC value.
- Choice between  $\pm 1$  or  $\pm 5$  is based on device crystal precision (  $< 10\text{ppm}$  )
- From 24 channel reference, the total channels is 34 with frequency offset.

# Find the pattern



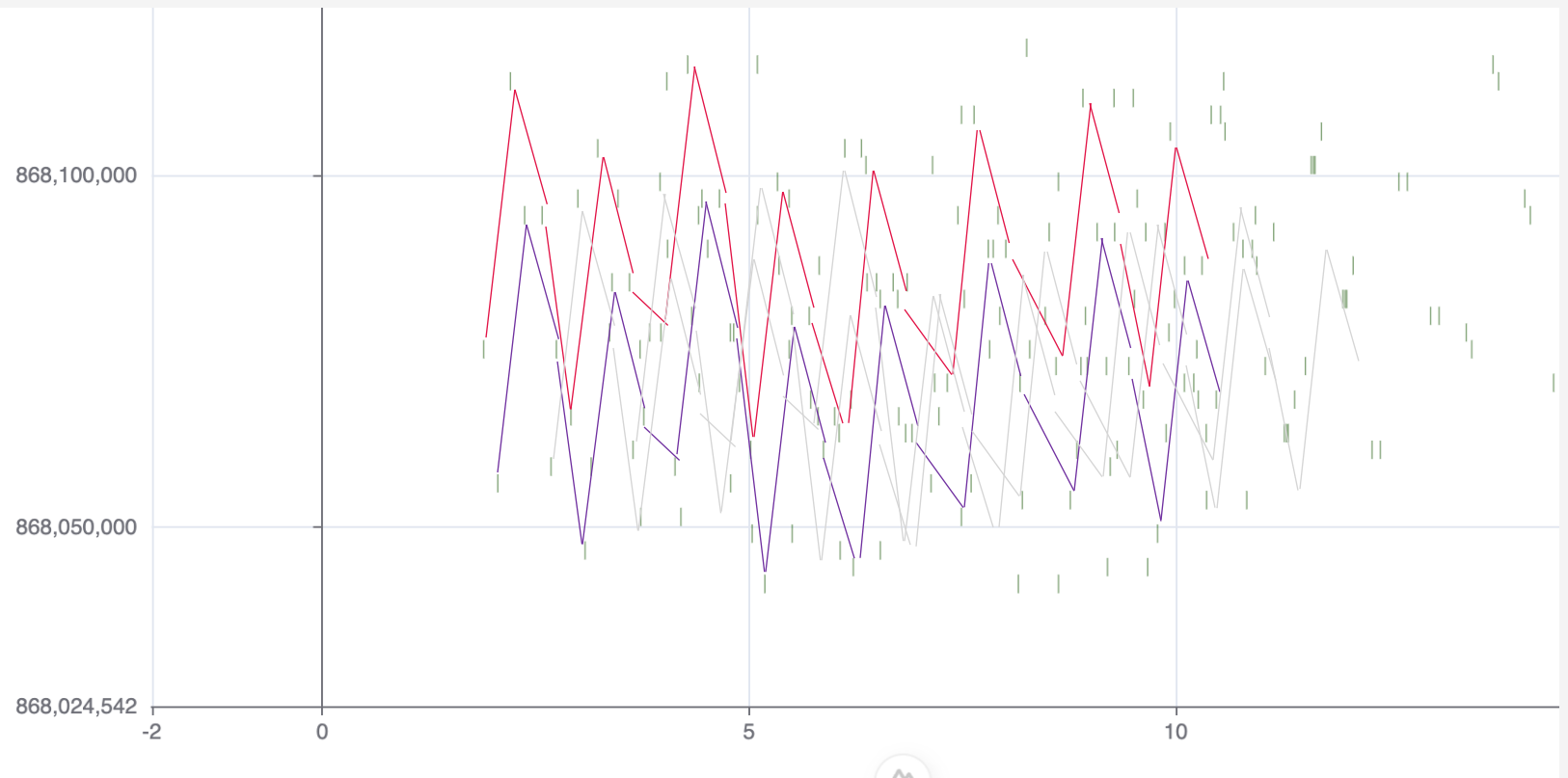
Select the core frame with the burst signalization.

Match the patterns on the unaffected burst to search what match with the best probability.

This requires analysis on a large window frame (>8s) and processing is growing with the number of frame / s

Validation made thanks CRC (coherence : pattern, offset, bank)

# Receive frames

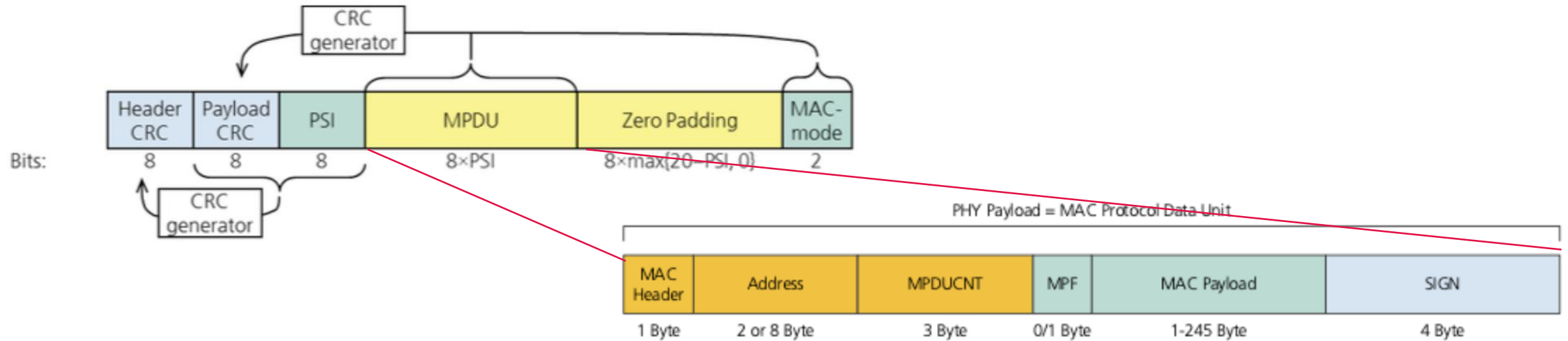


# Downlink

**Downlink use the same principle with:**

- **The pattern in use is indicated by some bits of the header CRC**
- **The frequency derivates from the uplink frequency (channel)**

**With these elements, it's possible for an end device to receives downlink without the necessity to listen the whole channel but just the right frequencies at the right time.**



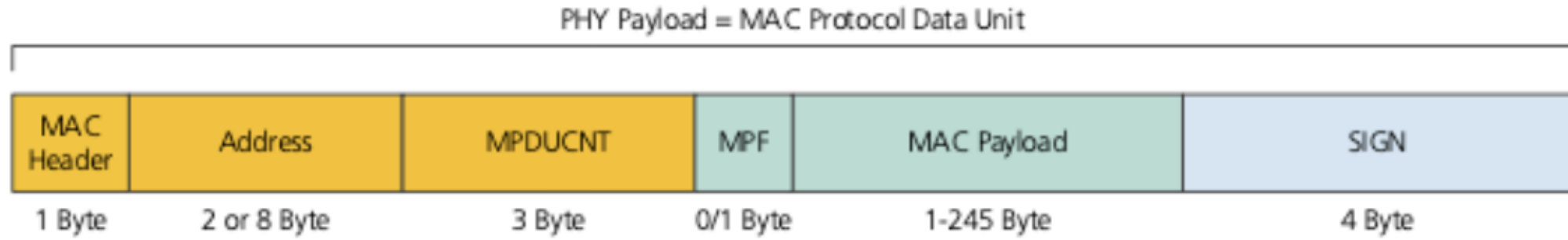
Physical frame contains header, size information and CRC. CRC will be used to determine the downlink pattern to be used.

The minimal size is 186 bits with a 160 bits payload. When smaller the rest is Zero Padded.

The MAC payload contains the Headers, Device Address, Frame counter (MPDUCNT) used for encryption and Signature

Security is based on a Pre-shared key used for AES-128-CTR encryption and AES-128-CBC like signature.

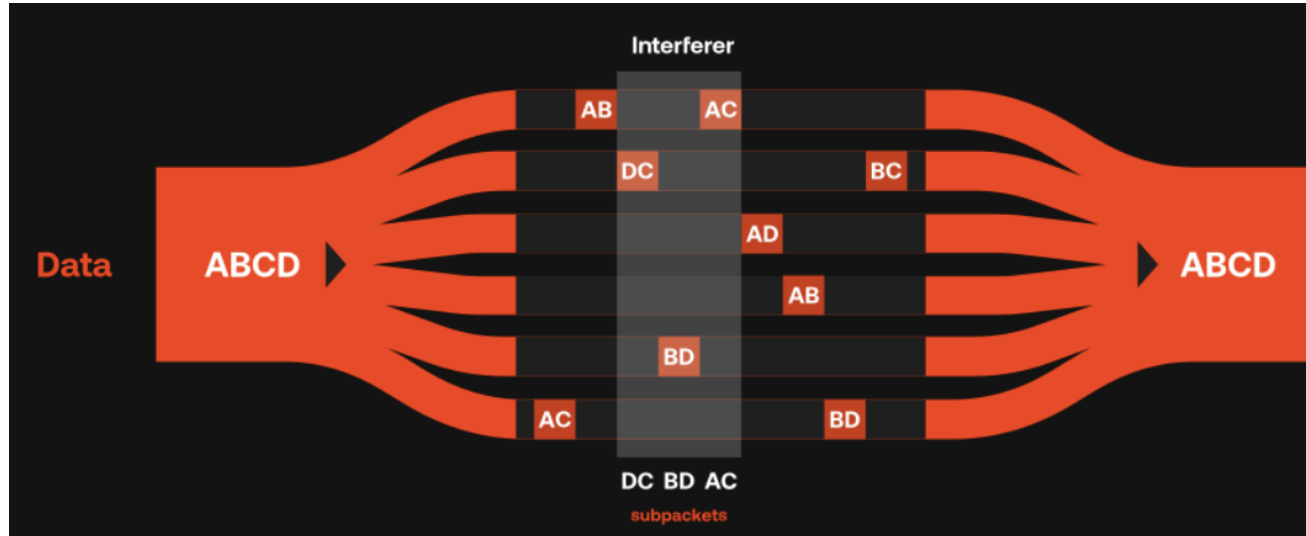
Core frame transports 3 bytes to 10 bytes for functional data



Address can be a 64bits IEEE but it can be optimized with a 16bits short address attribution from the network server.

MPDUCNT is a 24 bits counter, but signature and encryption are using a 32 bits counter internally.

Short addresses are reusable as the Signature is the real authentication method for the device.



Before transmission, the frame construction, triple the bit stream and transform it to improve the time & spectral distribution of each of the bit to be transmitted.

Consequently, if some of the splits are not received, the message reconstruction will be possible, most of the time, from the received one.

The consequence is a transfer efficiency divider by 3.

In regard of this and the pilot bits, every bits of payload requires a transmission of 4.5 bits over the air.

With the headers and minimal core frame size, transmission of 8 bits of data requires 837 bits. For 100 Bytes of data the transmitted total will be about 510 Bytes.



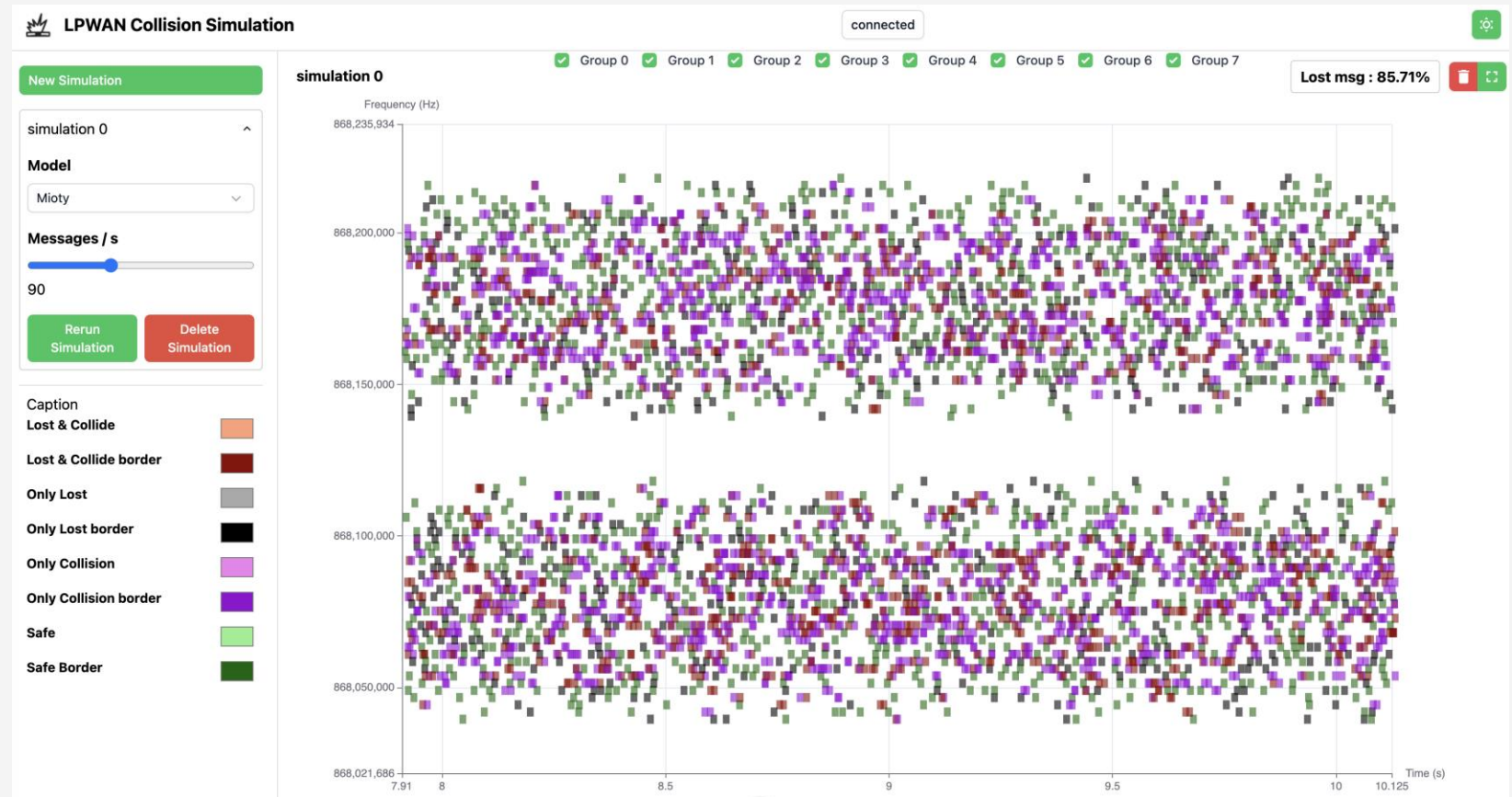
Simulator gives >80% loss for 90+ frame per seconds (core frame) – eq > 80.000 devices

We can see a lot of unused frequency space

This is the inflexion point with Sigfox & LoRaWan

Rq : simulator needs to be challenged

# Scalability



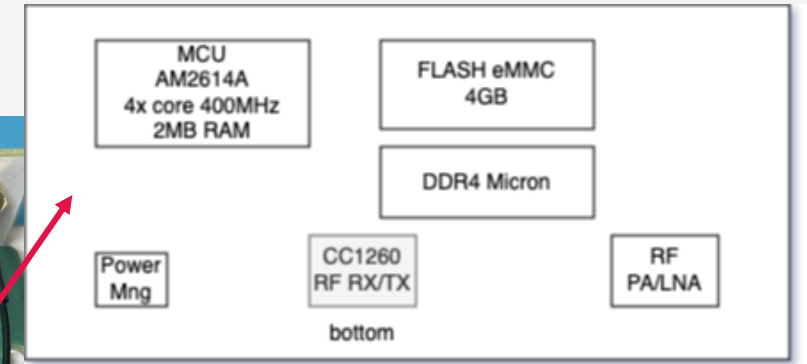
The Mioty concentrator is a system running an embedded Linux system processing the signal with SDR

The concentrator directly connects to the Mioty Network Server.

The Main board is just a network passthrough system

It's possible to access the Mioty concentrator dashboard redirecting some ports with ssh

# Hardware / concentrator

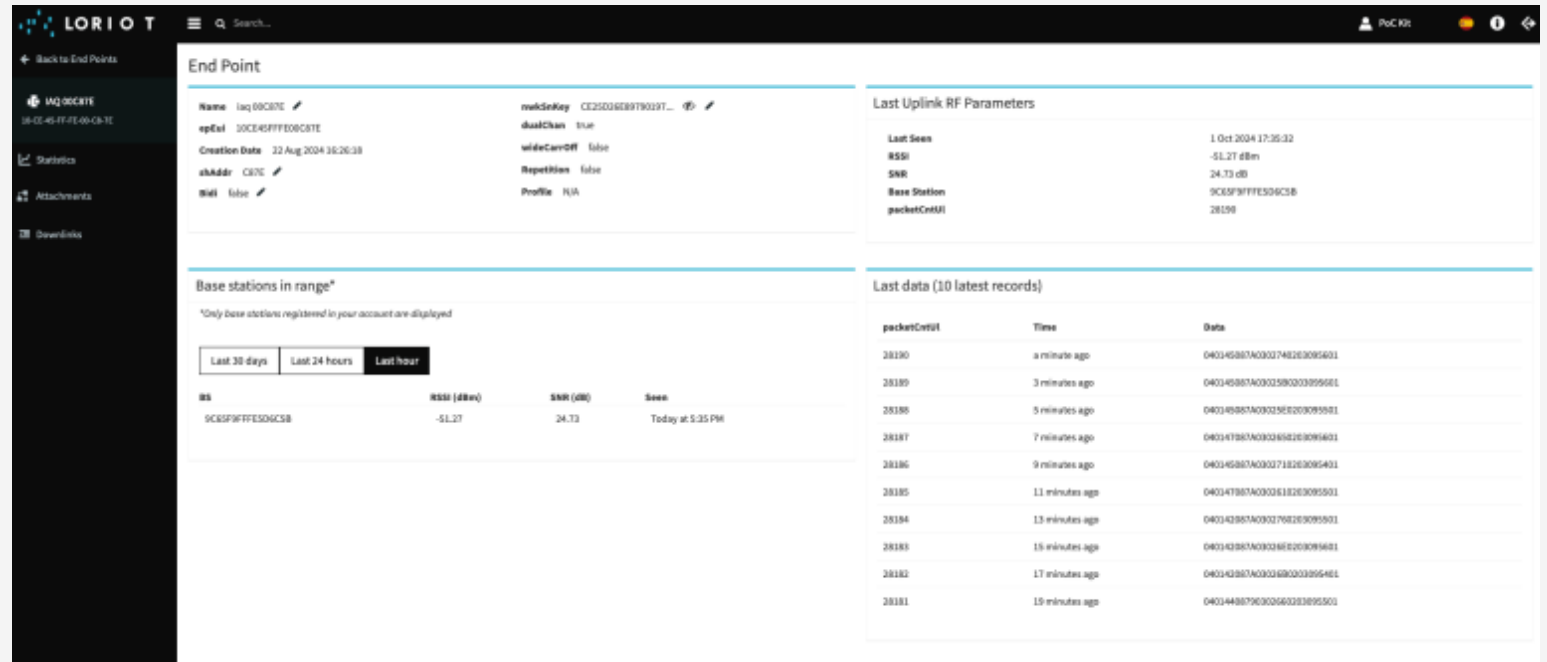


The Mioty network server works as a LoRaWAN Network Server

Not sure about an open-source Mioty Network Server

Network Server is setup inside the concentrator

# Mioty Network Server



The screenshot displays the LORIO T web interface, which is a LoRaWAN Network Server. The interface is divided into several sections:

- End Point:** This section shows details for a specific end point. The details include:
  - Name: laq00C7C
  - epEui: 00CE45FFFE08C7C
  - Creation Date: 22 Aug 2024 16:26:38
  - chAddr: C7C
  - isM: false
  - roleKey: CE25D36E89790297...
  - isM: true
  - validCarryOff: false
  - Repetition: false
  - Profile: N/A
- Last Uplink RF Parameters:** This section shows the last uplink RF parameters for the end point. The parameters include:
  - Last Seen: 1 Oct 2024 17:35:32
  - RSSI: -51.27 dBm
  - SNR: 24.73 dB
  - Base Station: 9CE5F8FFFE5D6C5B
  - packetCtrlId: 28198
- Base stations in range\*:** This section shows a list of base stations in range. The table has columns for BS, RSSI (dBm), SNR (dB), and Seen. The data shows one base station in range:
 

BS	RSSI (dBm)	SNR (dB)	Seen
9CE5F8FFFE5D6C5B	-51.27	24.73	Today at 5:35 PM
- Last data (10 latest records):** This section shows a list of the last 10 data records. The table has columns for packetCtrlId, Time, and Data. The data shows 10 records, each with a packetCtrlId, a time relative to the current time (e.g., 1 minute ago), and a data value (e.g., 040345887A03027482E3095601).

Mioty requires License to be used  
(even with the radiocraft modem)

# Mioty Devices

Solution	Rx / Tx	Development	Comment
Radiocraft modem	RX & TX	AT commands	Based on CC1310
STM32WL	TX only	Licensed SDK	
RFM69W	TX only	Open-source (but not license free)	
Weptech modules	RX & TX		Based on CC and STL32+S2LP
STM32WL3	RX & TX		Not evaluated
EFR32F62S	RX & TX		Not evaluated

Product category	Standard Rate	Compliant Rate
<b>Unidirectional End Point</b>	EUR 0.40	EUR 0.32
<b>Bidirectional End Point</b>	EUR 1.00	EUR 0.80
<b>Gateway</b>	EUR 10.00	EUR 8.00

## COLLISION SIMULATOR Initiated by Paul Pinault

### API & FRONT Integration by

- Lucas Blames
- Remi Boudrie

(ISIMA 2A)

### → Rules here

- 3x Redundancy for all
- 10B payload for all
- No downlink / ack considerations
- No noise / rssi / snr considerations
- 200KHz ( Sigfox & Mioty )
- 1MHz (LoRaWan)

Rq : simulator needs to be challenged

# Demo Time

<https://github.com/disk91/lpwan-collision-simulation-front>

<https://github.com/disk91/lpwan-collision-simulation-backend>

