Using FIT/CorteXlab for the Evaluation of Access Policies in Dense IoT Networks through S3CAP, a New Modular Framework

GRCon 2022

Amaury Paris, Leonardo S. Cardoso, Jean-Marie Gorce

INSA de Lyon, INRIA, CITI Lab

September 29, 2022

Multiple access in IoT

- IoT projections: the number of devices will more than double in the next years.
- Big problem to solve: huge number of devices sending very small packets in the uplink
- ALOHA-style medium access policies will become less viable for crowded IoT networks
- Collisions and re-transmissions: increased latency, higher energy consumption and shorter device battery life
- Going beyond simulations: inaccurate capture of interference patterns - real transmissions and repeatability

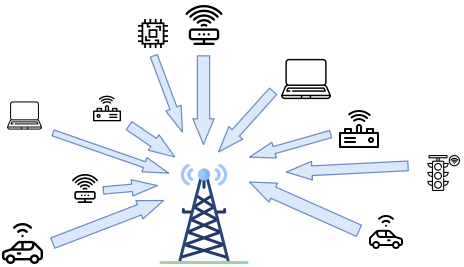
Multiple access in IoT

- IoT projections: the number of devices will more than double in the next years.
- Big problem to solve: huge number of devices sending very small packets in the uplink
- ALOHA-style medium access policies will become less viable for crowded IoT networks
- Collisions and re-transmissions: increased latency, higher energy consumption and shorter device battery life
- Going beyond simulations: inaccurate capture of interference patterns - real transmissions and repeatability
- Our response to this need: Slotted-Synchronized multi-Source framework for experimentation on Channel Access Policies

S3-CAP framework

S3-CAP framework

- **Channel access policies**
  Development and evaluation of new channel access policies
- **Robust coding at the transmitter**
  Study of interference, non-orthogonal communication, error-correcting systems
- **Improved multi-level receiver**
  Successive interference cancellation, joint decoding



Presentation Outline

1. FIT/CorteXlab testbed
2. Framework implementation
3. Usage

FIT/CorteXlab testbed

FIT/CorteXlab

- Future Internet of Things - COgnitive Radio Testbed and EXperimentation LAB
- 180m^2 shielded and semi-anechoized experimentation room: at least 60dB isolation
- 40 software defined radios among USRPs and PicoSDRs, all GNU Radio compatible
- Reproducible experiments: stable propagation environment exempt from outside interference
- Dockerized operation: any version of GNU Radio can be used (as well as any other UHD based code)
- Octoclocks: full synchronization of USRPs is possible
- **Free to use** and remotely accessible over the internet (https://wiki.cortexlab.fr)

Figure: USRP NI-2932, USRP N2944R, PicoSDR, Octoclock


PSDR 2x2 41
PSDR 2x2 42
PSDR 4x4 22
USRP N2932 24
PSDR 2x2 26
USRP N2932 28
USRP N2944 30
USRP N2932 32
USRP N2932 34
PSDR 2x2 36
USRP N2932 38
PSDR 4x4 40
PSDR 2x2 21
USRP N2932 23
USRP N2932 25
USRP N2932 27
PSDR 2x2 29
USRP N2944 31
USRP N2932 33
USRP N2932 35
USRP N2932 37
PSDR 2x2 39
PSDR 2x2 2
USRP N2932 4
USRP N2932 6
USRP N2932 8
USRP N2944 10
PSDR 2x2 12
USRP N2932 14
USRP N2932 16
USRP N2932 18
PSDR 2x2 20
PSDR 4x4 1
USRP N2932 3
PSDR 2x2 5
USRP N2932 7
USRP N2932 9
USRP N2944 11
USRP N2932 13
PSDR 2x2 15
USRP N2932 17
PSDR 4x4 19

Opportunities with FIT/CorteXlab

- Research! Test your multi-radio techniques and architectures in FIT/CorteXlab
- Teaching: Radio, propagation, etc.. lab sessions (during confinements)
- Learn GNU Radio and USRP operation autonomously: GNU Radio's or CorteXlab's tutorials
- Automated test for validation of UHD and GNU Radio code on actual hardware
- Any other usage you may think of...

S3_CAP framework

Framework Layout

Uplink channel access

- Access policies;
- Robust coding.

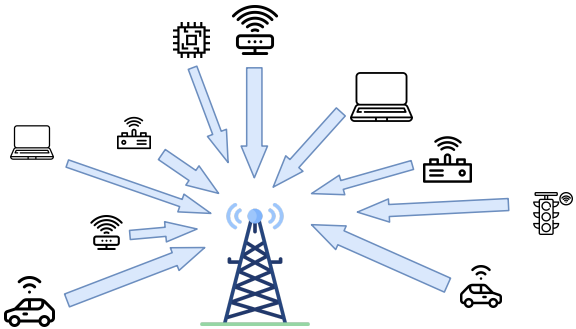


Figure: A star system

Framework Layout

Uplink channel access

- Access policies;
- Robust coding.

RF uplink channel

- Time framed;
- Slotted.

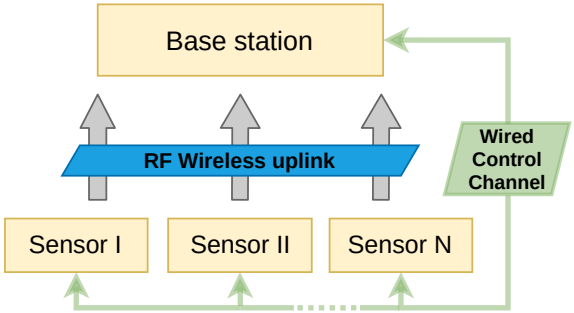


Figure: A star system

Framework Layout

Uplink channel access

- Access policies;
- Robust coding.

RF uplink channel

- Time framed;
- Slotted.

Wired control channel

- Uplink;
- Downlink;
- Error-free.

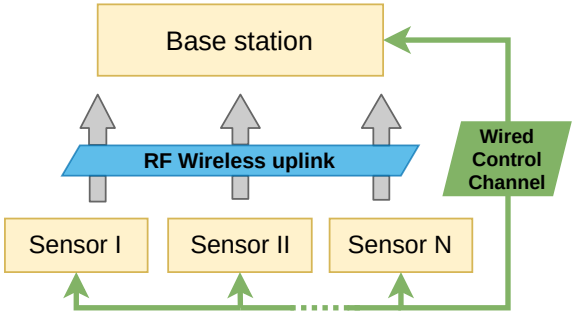


Figure: A star system

Framework Layout

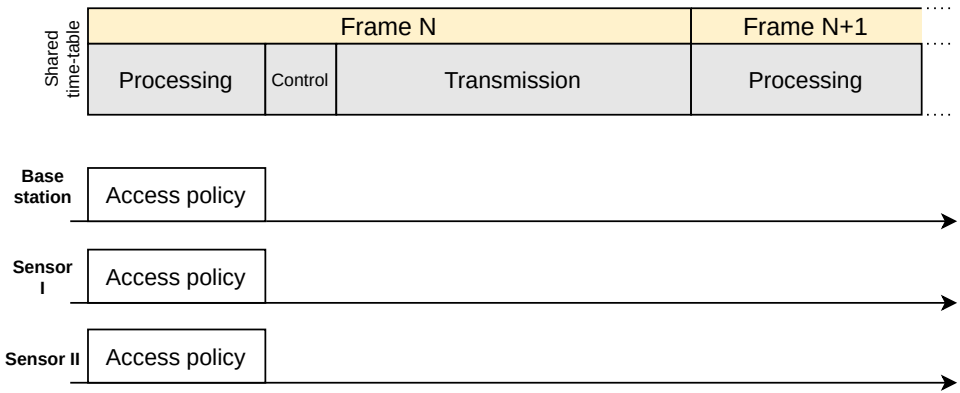


Figure: Shared time table

Framework Layout

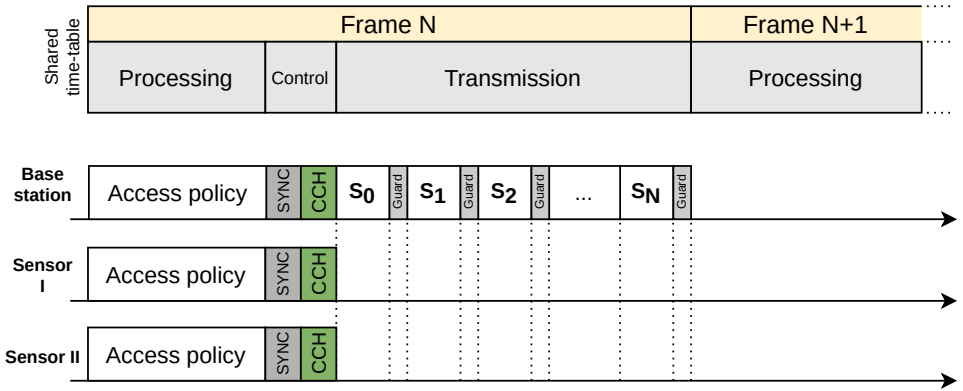


Figure: Shared time table

Framework Layout

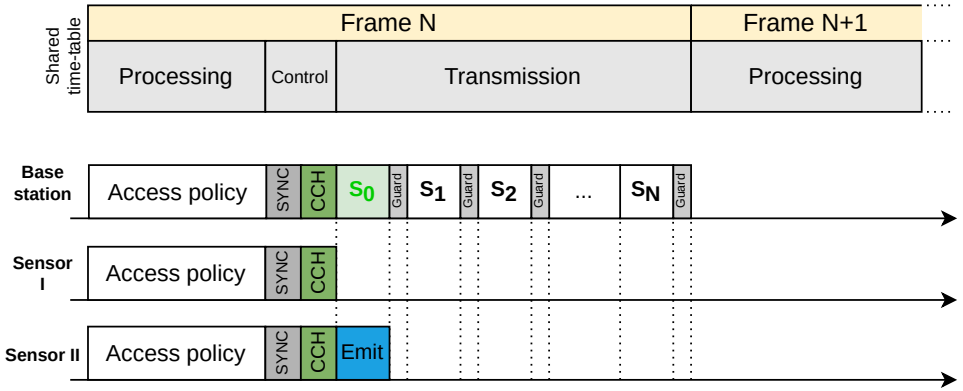


Figure: Shared time table

Framework Layout

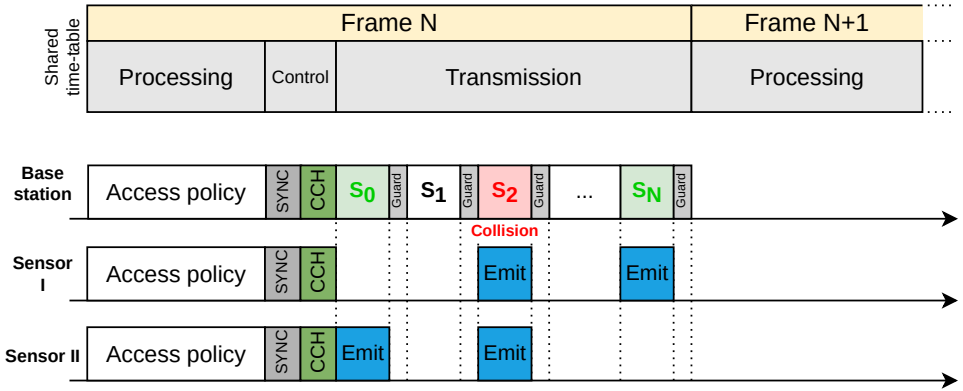


Figure: Shared time table

Framework Layout

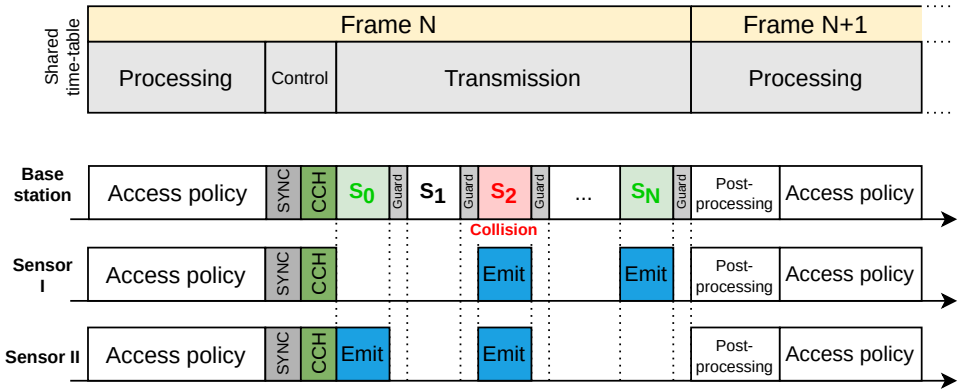


Figure: Shared time table

Adaptive implementation

Implement MAC layer in GNU Radio

- Time consuming;
- Unnecessarily difficulty.

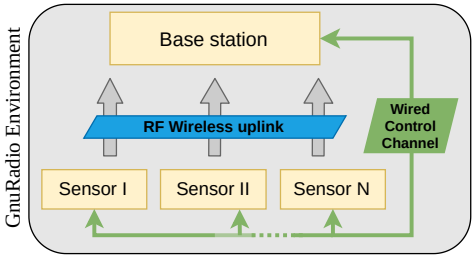


Figure: Access policy implementation

Adaptive implementation

Implement MAC layer in GNU Radio

- Time consuming;
- Unnecessarily difficulty.

External implementation

- Centralized;

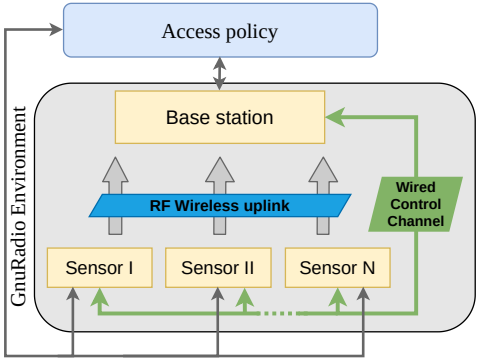


Figure: Access policy implementation

Adaptive implementation

Implement MAC layer in GNU Radio

- Time consuming;
- Unnecessarily difficulty.

External implementation

- Centralized;
- Decentralized.

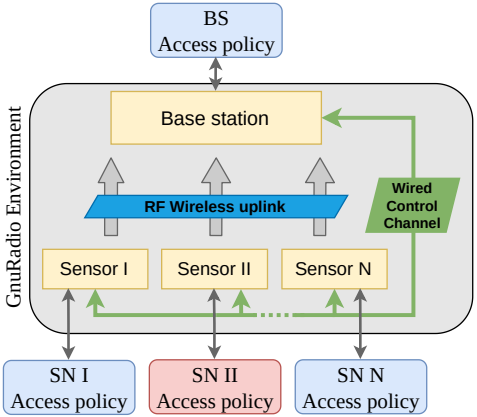


Figure: Access policy implementation

Adaptive implementation

Implement MAC layer in GNU Radio

- Time consuming;
- Unnecessarily difficulty.

External implementation

- Centralized;
- Decentralized.

Distributed implementation

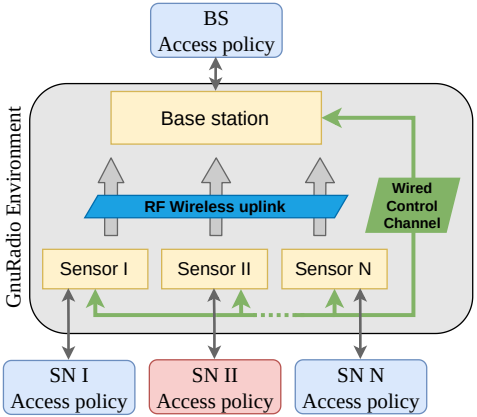


Figure: Access policy implementation

Adaptive implementation

Implement MAC layer in GNU Radio

- Time consuming;
- Unnecessarily difficulty.

External implementation

- Centralized;
- Decentralized.

Distributed implementation

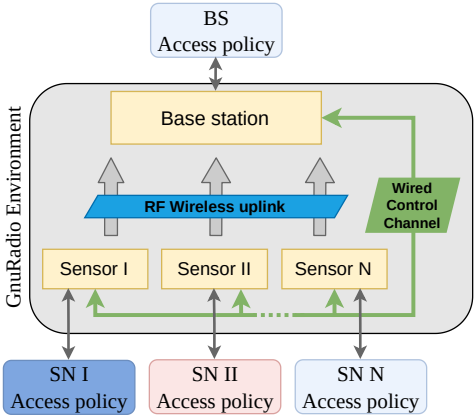


Figure: Access policy implementation

Dummy access policy

```
up = upper()

while True:
    pp = up.extract(up.recv())
    print("SN : %s, Frame : %s" % (pp["NODE"], pp["FRAME"]))

    if pp["DLCCH"] == "GR_Con_22":
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    True, [[0, "X"], [1, False]],
                                    ulcch = "Hello there")
    else:
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    False, [], ulcch = "Hello there")

    up.send(inst)
```

Dummy access policy

```
up = upper()

while True:
    pp = up.extract(up.recv())
    print("SN : %s, Frame : %s" % (pp["NODE"], pp["FRAME"]))

    if pp["DLCCH"] == "GR_Con_22":
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    True, [[0, "X"], [1, False]],
                                    ulcch = "Hello there")
    else:
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    False, [], ulcch = "Hello there")

    up.send(inst)
```

ZMQ message passing

Dummy access policy

```
up = upper()

while True:
    pp = up.extract(up.recv())
    print("SN : %s, Frame : %s" % (pp["NODE"], pp["FRAME"]))

    if pp["DLCCH"] == "GR_Con_22":
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    True, [[0, "X"], [1, False]],
                                    ulcch = "Hello there")
    else:
        inst = up.create_SN_inst(   pp["NODE"], pp["FRAME"] +1,
                                    False, [], ulcch = "Hello there")

    up.send(inst)
```

ZMQ message passing

PMT dict

Modular physical layer

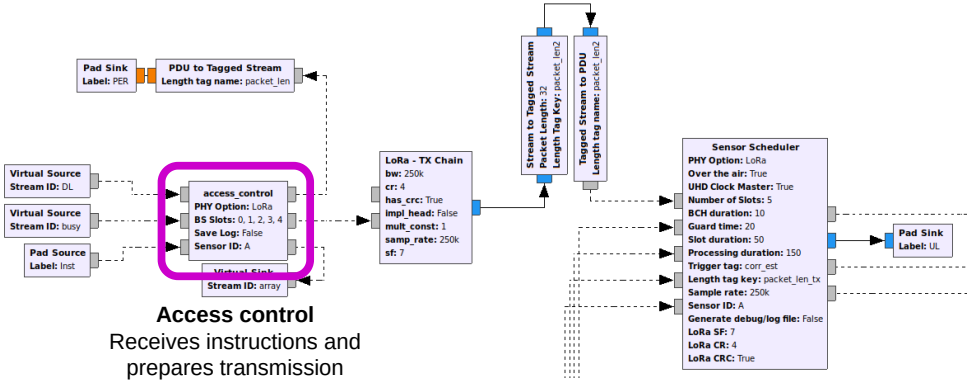


Figure: Sensor architecture

Modular physical layer

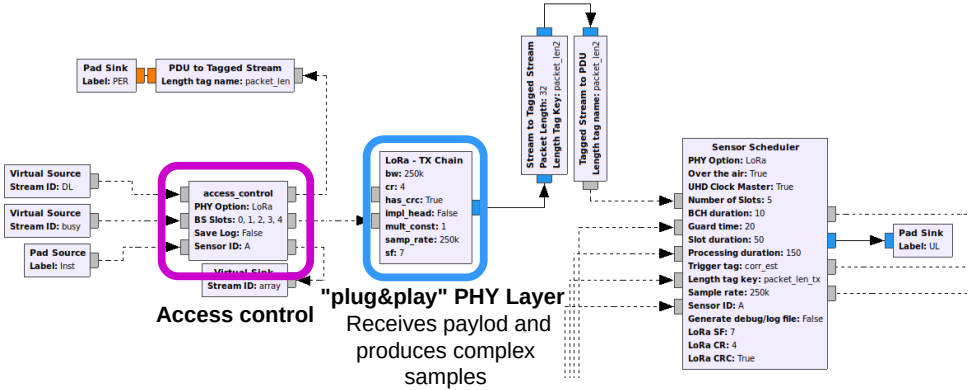


Figure: Sensor architecture

Modular physical layer

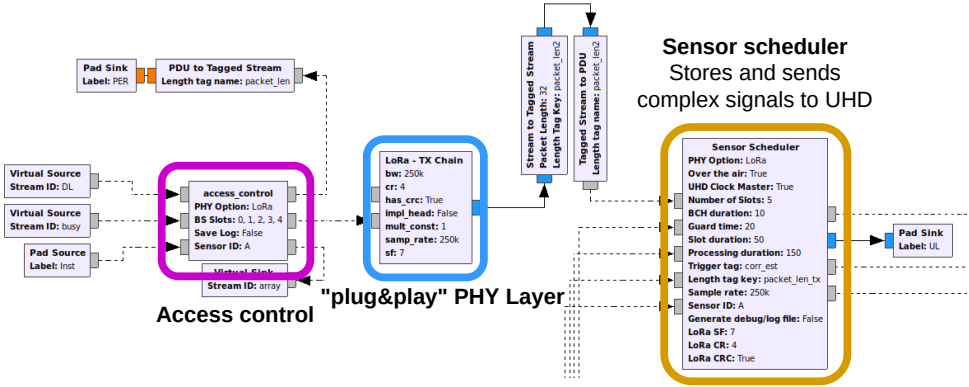
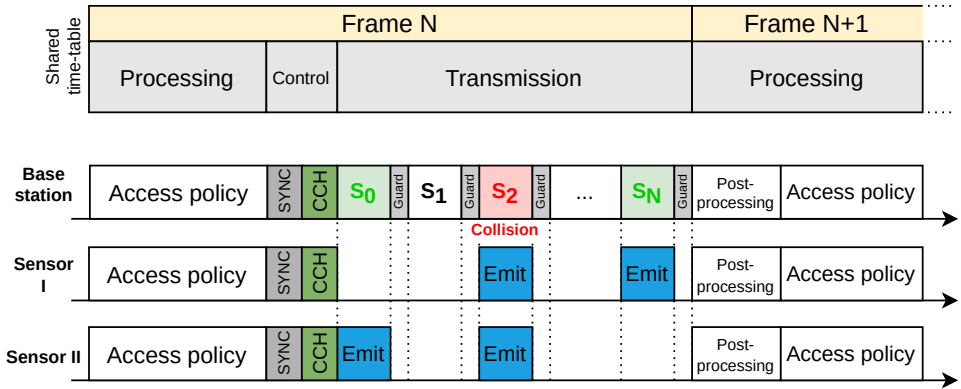
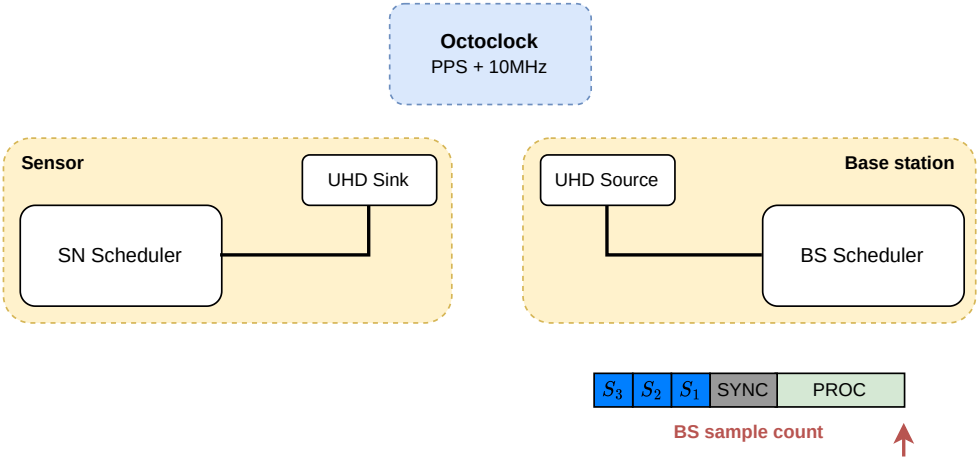


Figure: Sensor architecture

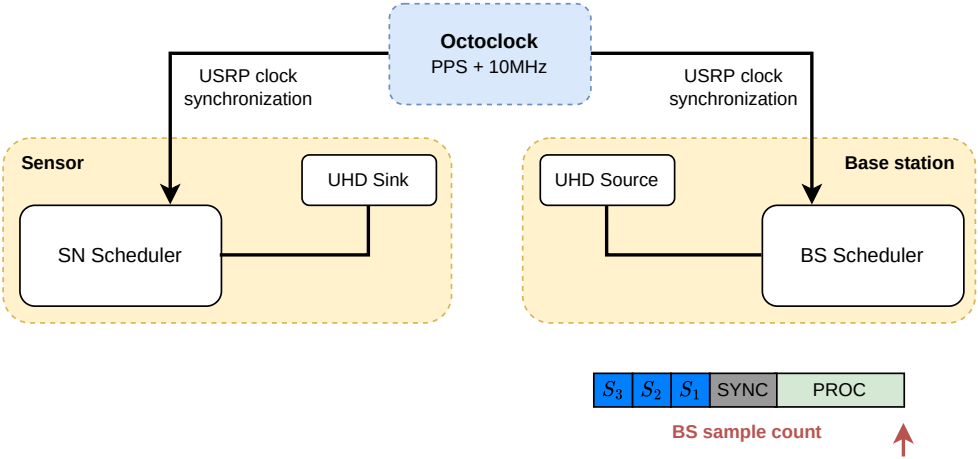
Synchronization



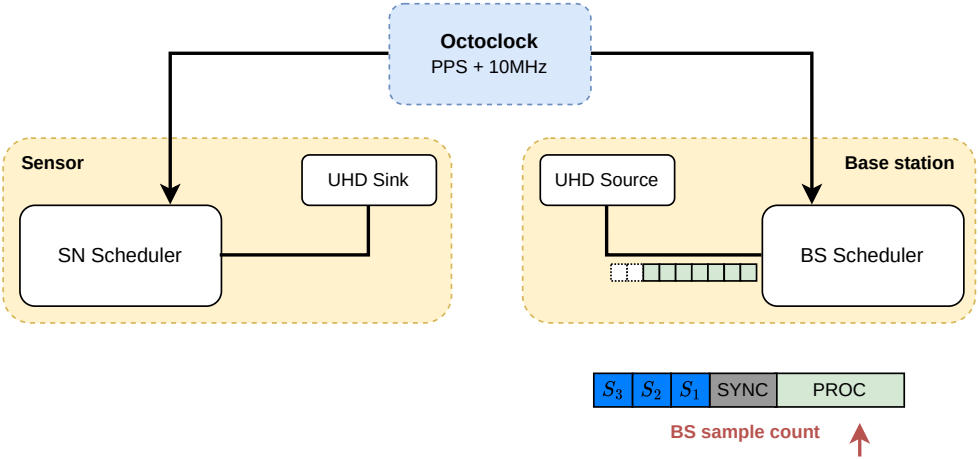
Synchronization



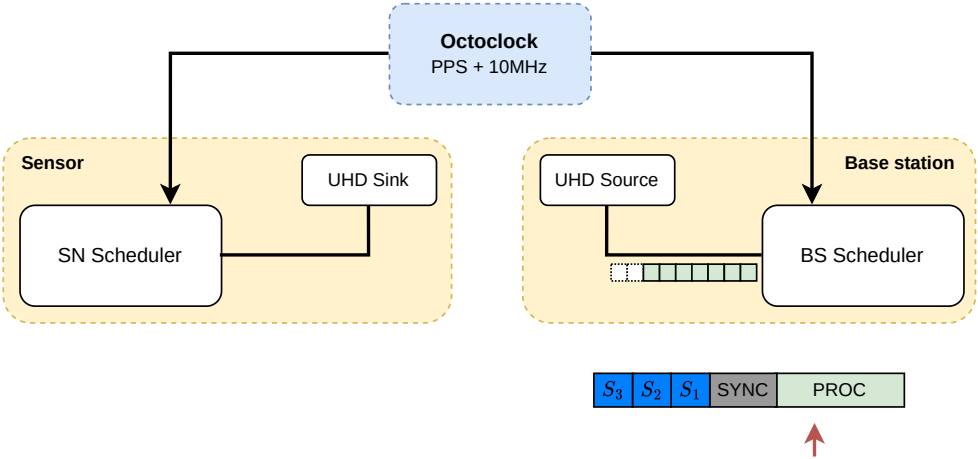
Synchronization



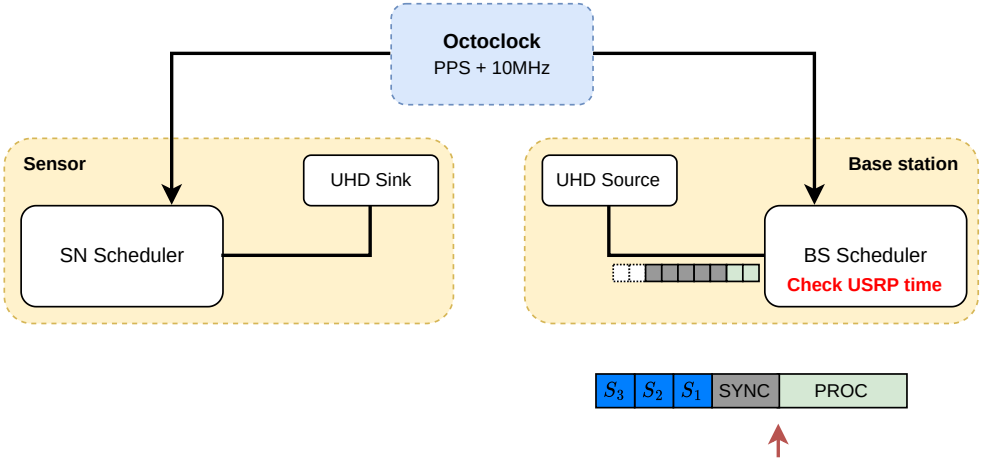
Synchronization



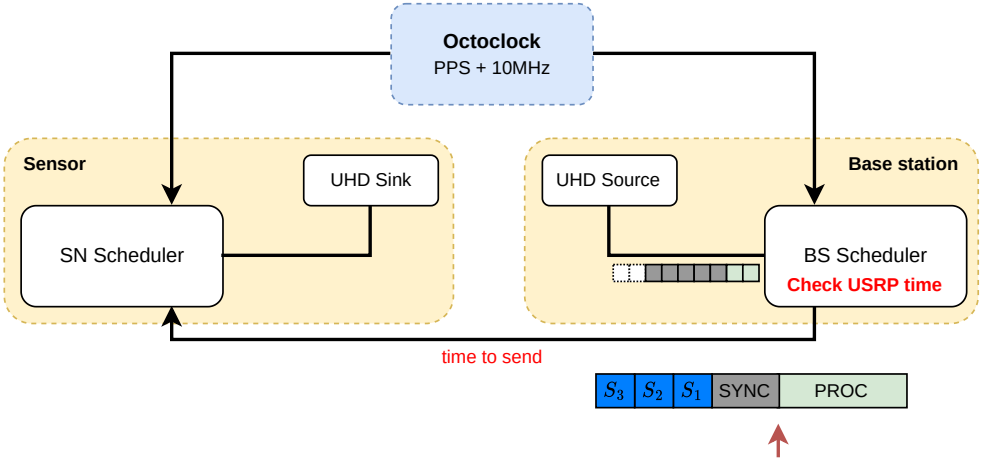
Synchronization



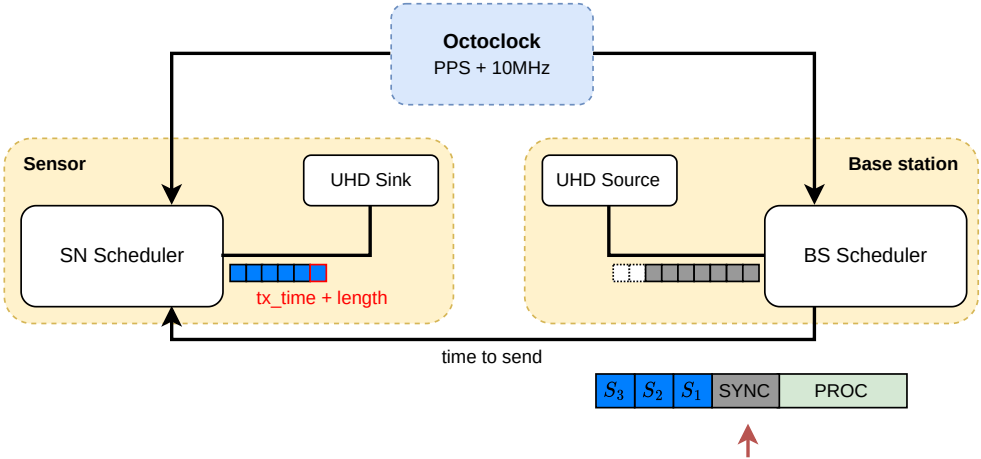
Synchronization



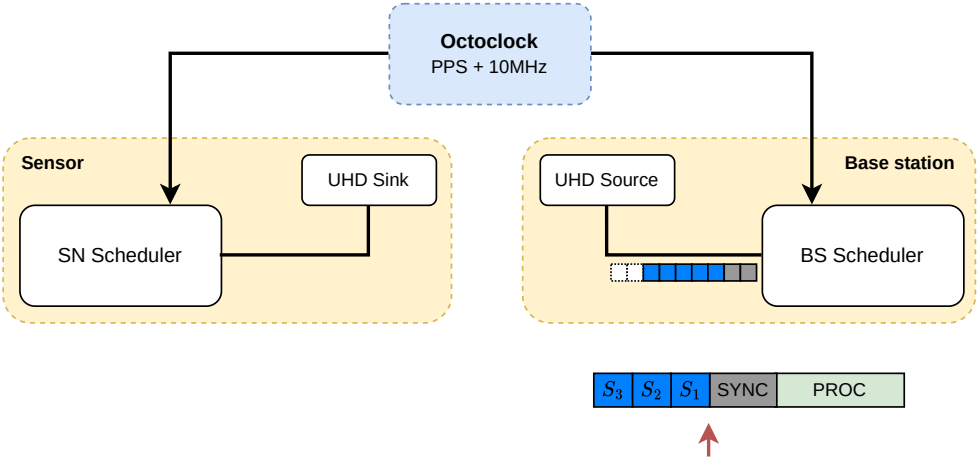
Synchronization



Synchronization



Synchronization



Synchronization

TX synchronization

- $\mathbf{0 \pm 150ns}$ delay for two nodes transmitting at the same time[1]

[1] Measurement made @10MHz samp_rate within FIT/CorteXlab by Cyrille Morin, Centrale Supelec, France

Synchronization

TX synchronization

- **0 ± 150ns** delay for two nodes transmitting at the same time[1]

RX synchronization

- **avg 0.7ms** delay between one node transmission and RX window

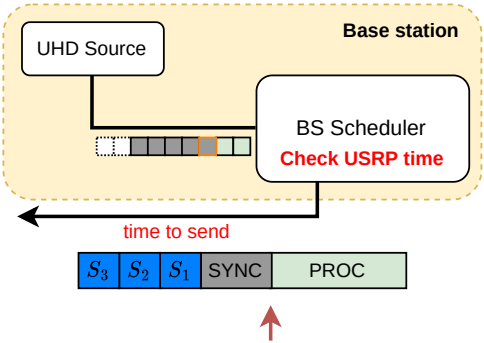


Figure: RX Synchronization

[1] Measurement made @10MHz samp_rate within FIT/CorteXlab by Cyrille Morin, Centrale Supelec, France

Synchronization

TX synchronization

- **0 ± 150ns** delay for two nodes transmitting at the same time[1]

RX synchronization

- **avg 0.7ms** delay between one node transmission and RX window

Corrected RX synchronization

- **Expecting 1.1μs** delay[1]

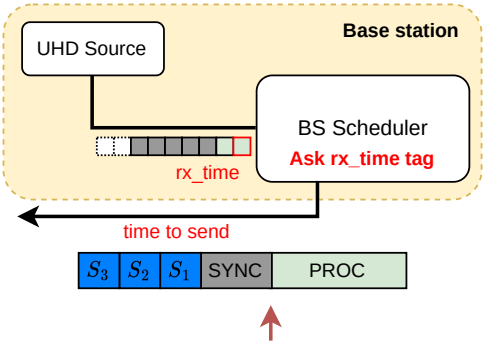


Figure: RX Synchronization

[1] Measurement made @10MHz samp_rate within FIT/CorteXlab by Cyrille Morin, Centrale Supelec, France

Synchronization

TX synchronization

- **0 ± 150ns** delay for two nodes transmitting at the same time[1]

RX synchronization

- **avg 0.7ms** delay between one node transmission and RX window

Corrected RX synchronization

- **Expecting 1.1μs** delay[1]

Generation of dataset

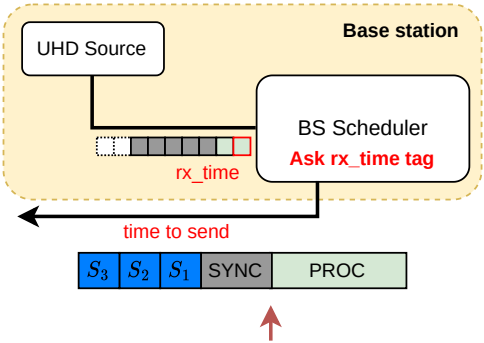


Figure: RX Synchronization

[1] Measurement made @10MHz samp_rate within FIT/CorteXlab by Cyrille Morin, Centrale Supelec, France

Usage

Running the framework on your own local computer :

- Development phase;
- Theoretical results.

[2] *"An Open-Source LoRa Physical Layer Prototype on GNU Radio"*, J. Tapparel et al., EPFL

Usage

Running the framework on your own local computer :

- Development phase;
- Theoretical results.

Available PHY layer

- LoRa [2]
- Nb IoT, under development
- Yours ...

[2] *"An Open-Source LoRa Physical Layer Prototype on GNU Radio"*, J. Tapparel et al., EPFL

Usage

Running the framework on your own local computer :

- Development phase;
- Theoretical results.

Available PHY layer

- LoRa [2]
- Nb IoT, under development
- Yours ...

Possibility to emulate nodes

[2] *"An Open-Source LoRa Physical Layer Prototype on GNU Radio"*, J. Tapparel et al., EPFL

Tutorial

Available on the FIT/CorteXlab wiki [under development]:

- Two nodes
- Random access policies

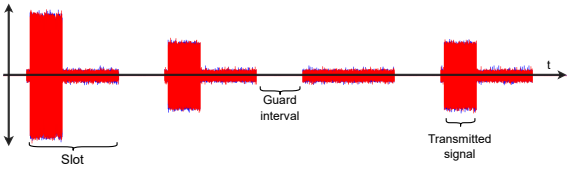


Figure: Received signal at the BS

S3-CAP tutorial : `https://wiki.cortexlab.fr/doku.php?id=ephyl_framework_v2`

S3_CAP Framework

Current and future work :

- **Nb IoT** : Finalisation of the Nb Iot PHY layer
- **RL access policies** : Apply a RL access policies method designed for LoRa system
- **Interference statistics** : Study the distribution of baseband IQ samples during collision

Thanks !

If you have any additional questions or requests, please contact us at :

amaury.paris@insa-lyon.fr

leonardo.cardoso@insa-lyon.fr

- **FIT/CortexXlab** : `http://www.cortexlab.fr/`
- **Wiki FIT/CorteXlab** : `https://wiki.cortexlab.fr/doku.php`
- **S3-CAP tutorial** : `https://wiki.cortexlab.fr/doku.php?id=ephyl_framework_v2`