

DANANG INTERNATIONAL INSTITUTE OF TECHNOLOGY









Smart Campus of Danang LoRa Technology



Outline

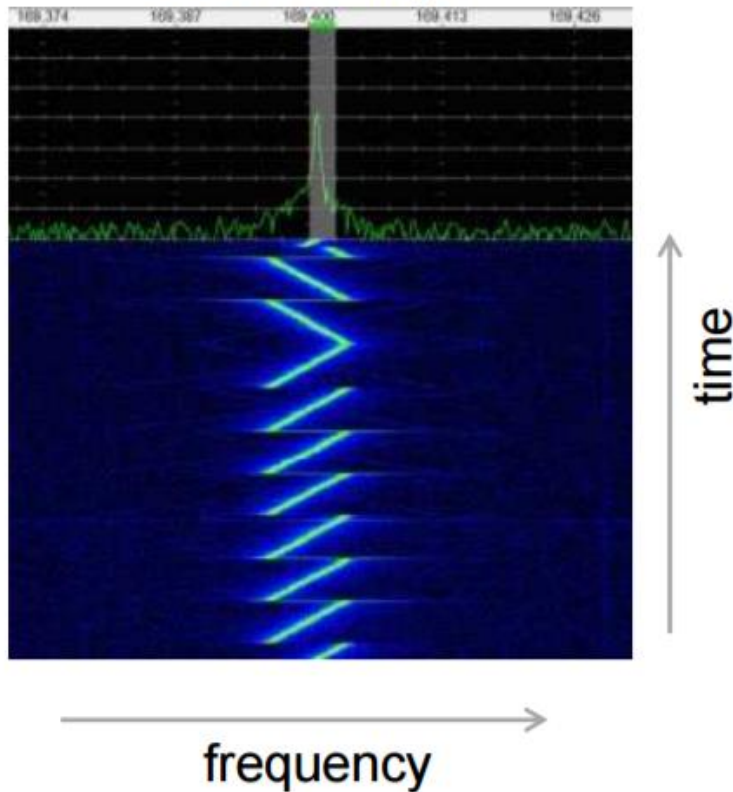
- LoRa technology
- Difference between LoRa and LoRaWAN
- **Node and Gateway** : LoRa board with Arduino
- **Gateway** : LoRa gateway with Raspberry Pi
- **DEMO**

LP-WAN connectivity overview

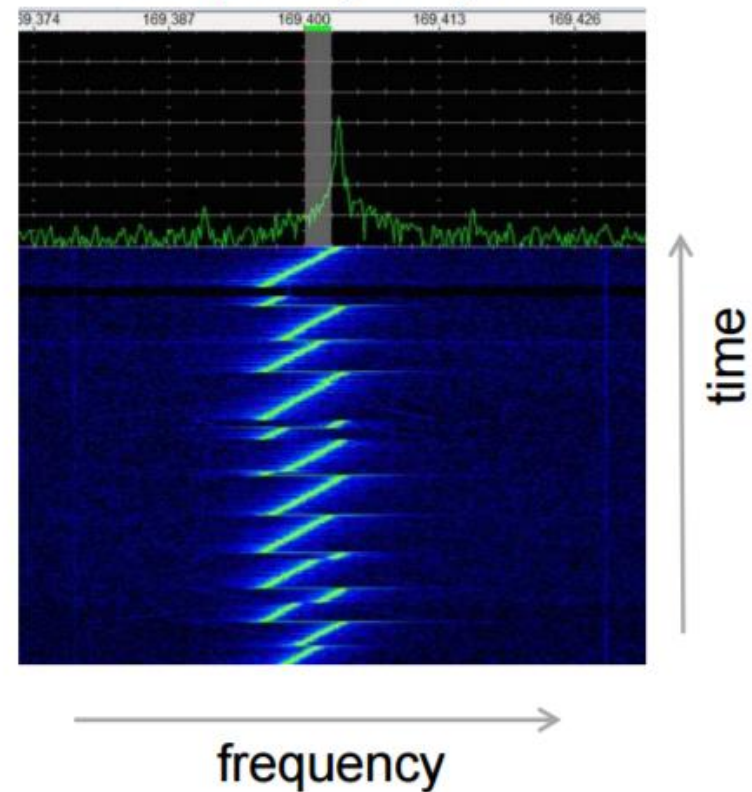
	SIGFOX 	LoRa 	clean slate cloT	NB LTE-M Rel. 13 	LTE-M Rel. 12/13 	EC-GSM Rel. 13 	5G (targets) 
Range (outdoor) MCL	<13km 160 dB	<11km 157 dB	<15km 164 dB	<15km 164 dB	<11km 156 dB	<15km 164 dB	<15km 164 dB
Spectrum Bandwidth	Unlicensed 900MHz 100Hz	Unlicensed 900MHz <500kHz	Licensed 7-900MHz 200kHz or dedicated	Licensed 7-900MHz 200kHz or shared	Licensed 7-900MHz 1.4 MHz or shared	Licensed 8-900MHz 2.4 MHz or shared	Licensed 7-900MHz shared
Data rate	<100bps	<10 kbps	<50kbps	<150kbps	<1 Mbps	10kbps	<1 Mbps
Battery life	>10 years	>10 years	>10 years	>10 years	>10 years	>10 years	>10 years
Availability	Today	Today	2016	2016	2016	2016	beyond 2020

LoRa Chirp Spread Spectrum (CSS)

LoRa pre-amble signal:
8 symbols or “chirps”,
1 reverse “chirp”.



LoRa data signal:
A symbol is a “chirp” with
a frequency “hop”.



Chirp Spread Spectrum (CSS)

- LoRa Spreading factor

$$R_b = SF * \frac{1}{\left[\frac{2^{SF}}{BW}\right]} \text{ bits/sec} \quad \text{Where:}$$

SF = spreading factor (7..12)

BW = modulation bandwidth (Hz)

Mode	Equivalent bit rate (kb/s)	Sensitivity (dBm)	Δ (dB)
FSK	1.2	-122	-
LoRa SF = 12	0.293 kb/s	-137	+15
LoRa SF = 11	0.537 kb/s	-134.5	+12.5
LoRa SF = 10	0.976 kb/s	-132	+10
LoRa SF = 9	1757 b/s	-129	+7
LoRa SF = 8	3125 b/s	-126	+4
LoRa SF = 7	5468 b/s	-123	+1
LoRa SF = 6	9375 b/s	-118	-3

Table 1: Link Budget Comparison for Narrowband FSK

LoRa modulation schemes

LoRa Technology Modulation

- **Spreading Factor (SF)**

The higher the SF the more information is transmitted per bit; therefore higher processing gain = increase receive sensitivity

Programmable SF : 7, 8, 9, 10, 11, 12

- **Bandwidth (BW)**

For a given SF, a narrower BW = increase received sensitivity; however increase time on air

Programmable signal BW settings : 125 kHz, 250 kHz, 500 kHz

- **Forward Error Correction Code Rate (CR):**

Additional coding rate provides more redundancy to detect errors and correct them

Class A: Average Current for Pout = +14 dBm

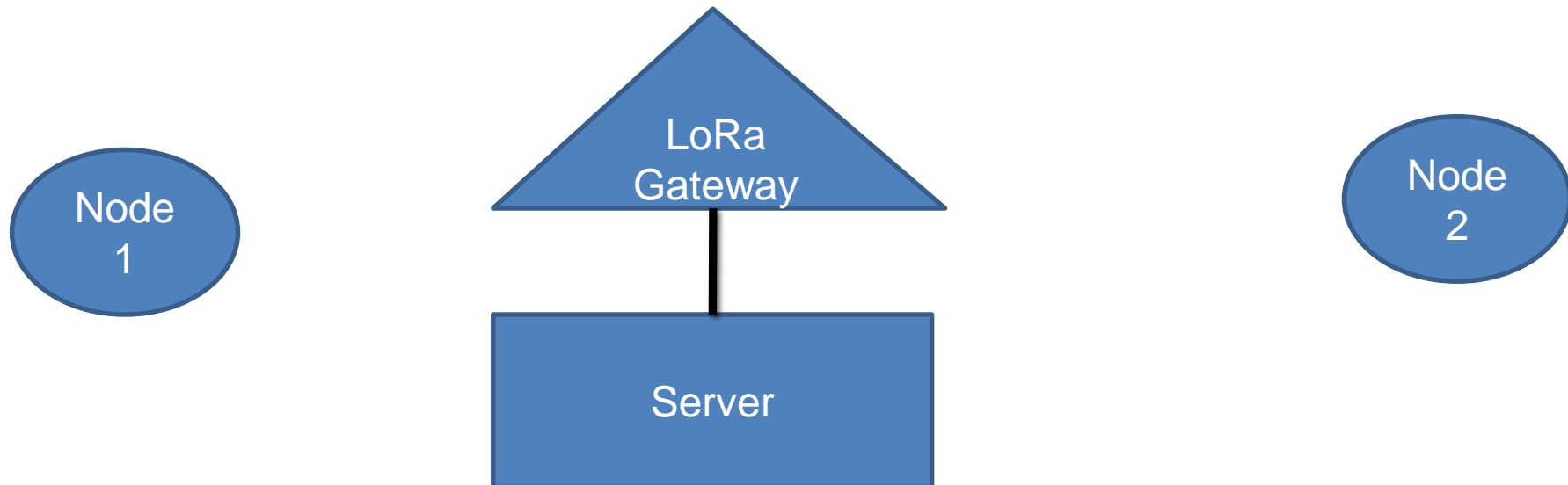
Assumptions:

- 10 packets / day
- Sleep current ~1 uA (includes the MCU)
- MCU is mostly Off during Tx
- The energy usage of the 2 unused Rx windows is low (<10%)
- Pout = +14 dBm, IDDTX = 32 mA

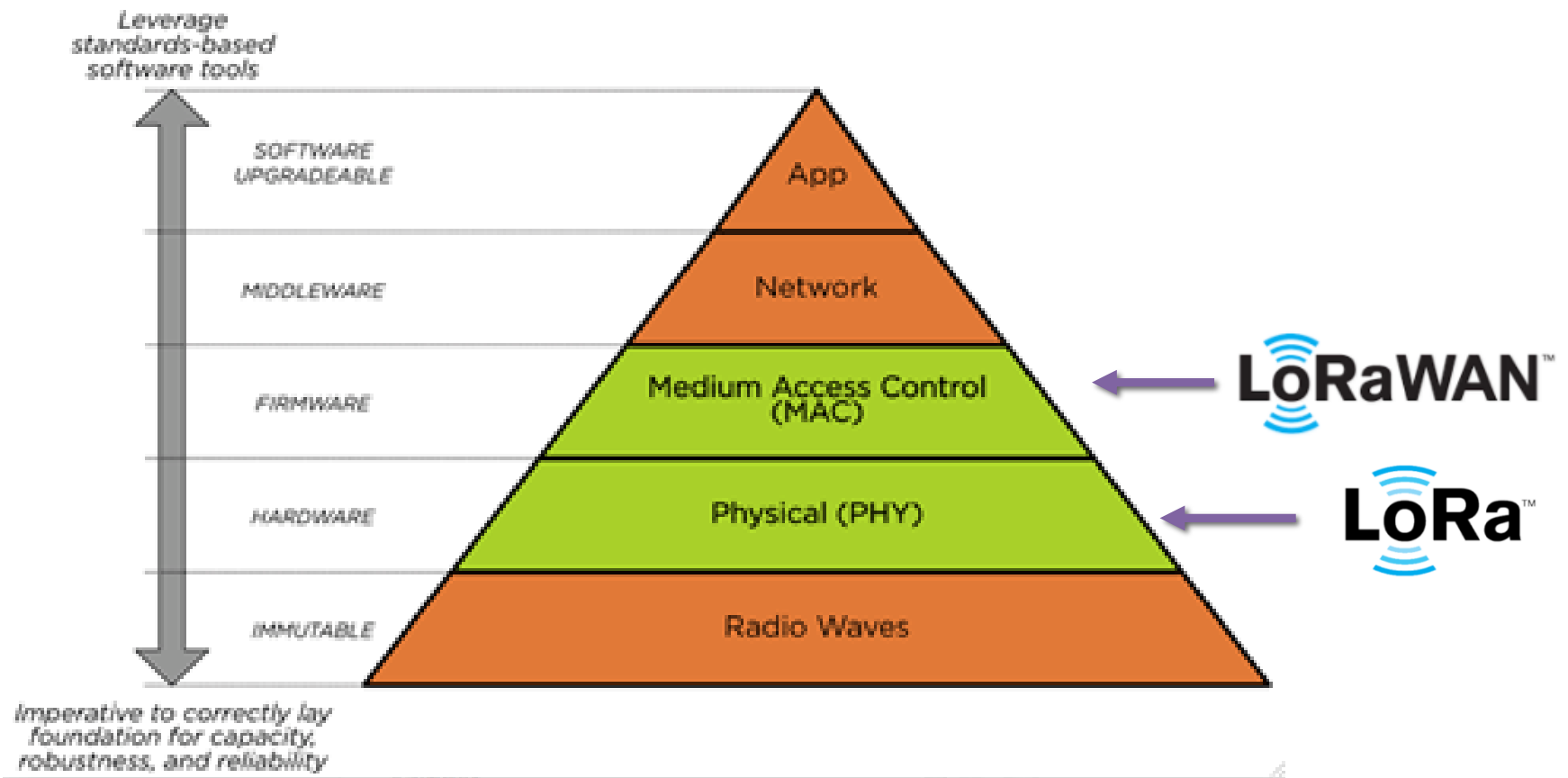
FRMPayload size (Bytes)	240 bps SF12/125k	1 kbps SF10/125k	5.5 kbps SF7/125k
4	~5 uA	~2.2 uA	~1.2 uA
16	~7 uA	~2.5 uA	~1.3 uA
30	~9 uA	~3 uA	~1.4 uA

LoRa vs LoRaWan

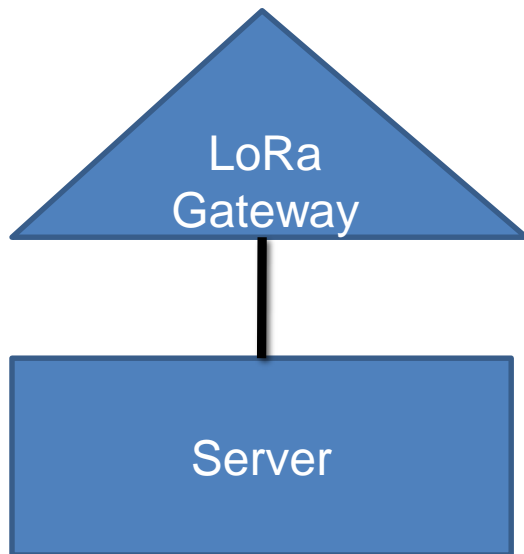
- LoRa is the modulation, any type of protocols can be used
- LoRaWAN is the protocol developped by IBM using LoRa modulation, it is standardized and adopted by many telecom operators
- A gateway can be LoRaWAN or just be LoRa



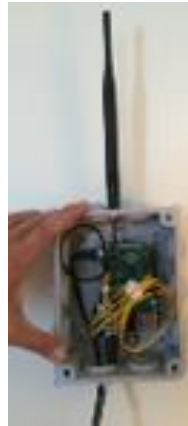
LoRa vs LoRaWan



LoRa vs LoRaWan : Difference on the gateway



Low cost Gateway



Based on SX1272 chip

Receive 1 freq band

Receive only one SF

About 10 nodes

Cost 10\$

LoRaWAN gateway



Based on SX1301 chip

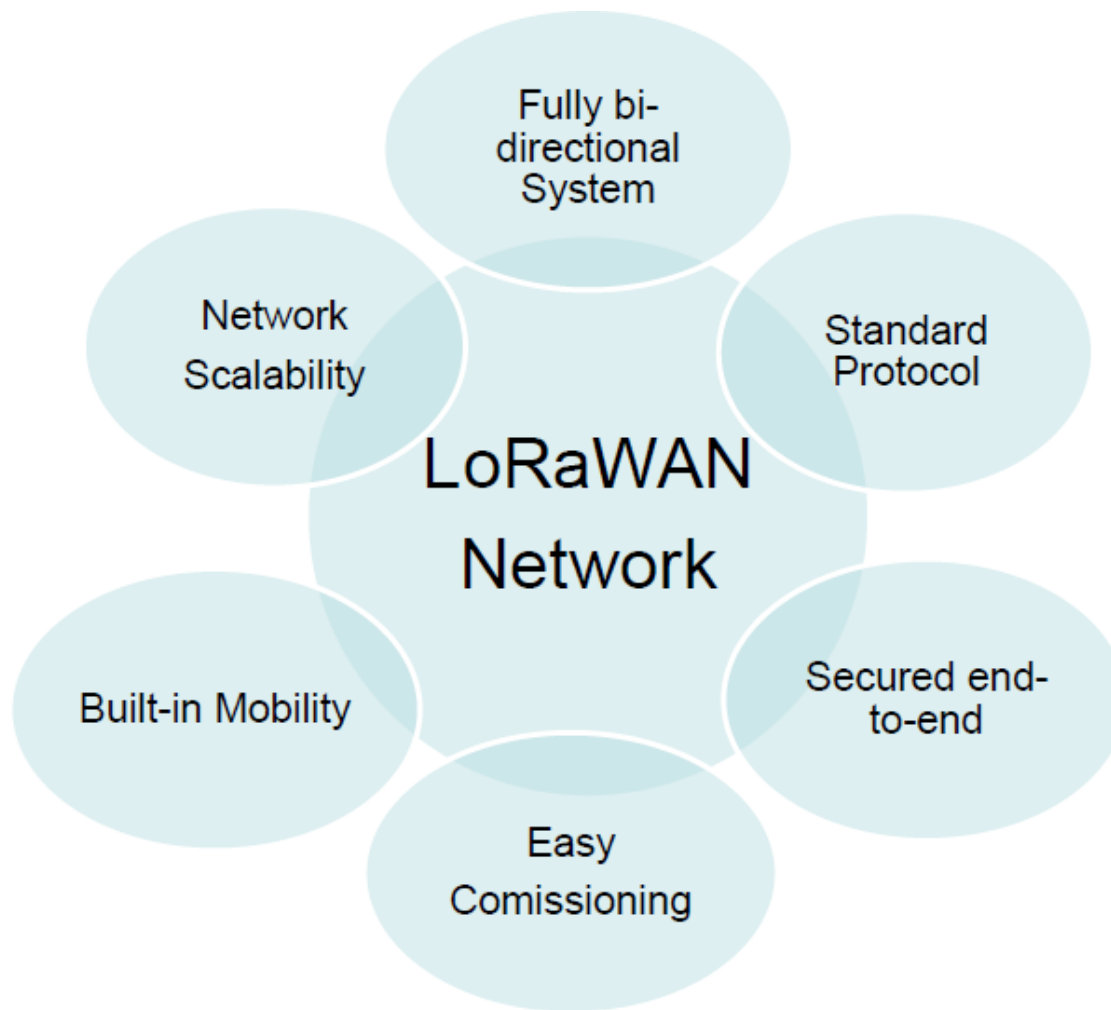
Receive 8 freq band

Receive all the SF
(7,8,9,10,11,12)

10000 nodes

Cost 300\$

LoRa vs LoRaWan



Class A

Report status a few times per day
No planned actuation required
Extremely low energy



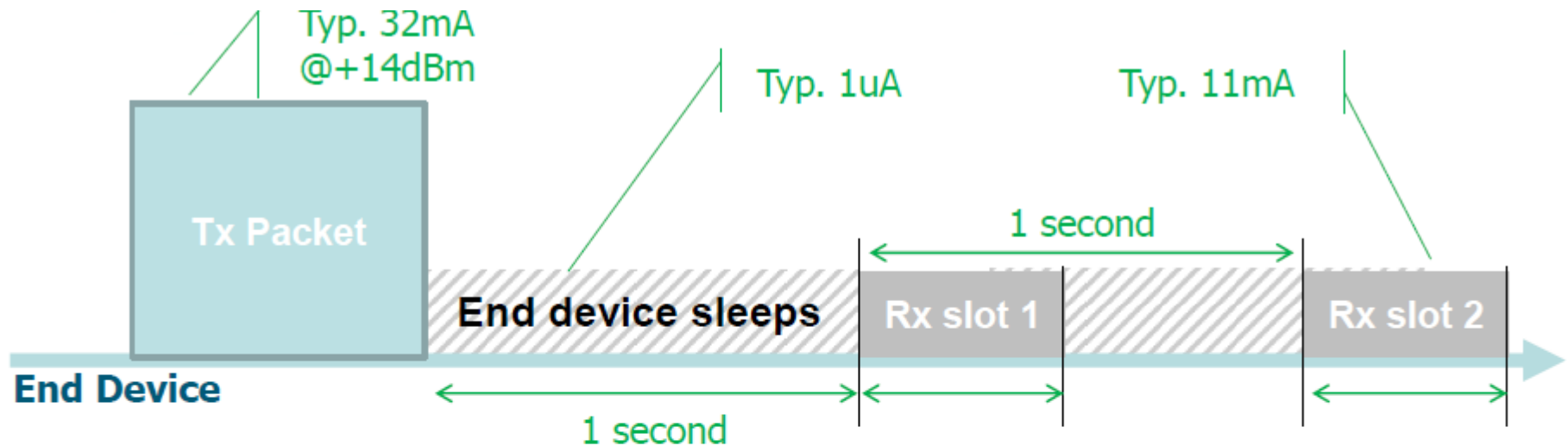
Class B

Report moisture, t° a few times per day
Turn valves on or off with a few minutes latency
Very low-energy, which depends on latency

Class C

Maintenance and index info a few times / day
Constantly listens for network «ping»
For low-latency actuation

LoRa vs LoRaWan : Class A exemple



Minimum Rx wake-up time = 5 Symbols :

5.1 ms @ SF7

10.2 ms @ SF8

...

164 ms @ SF12

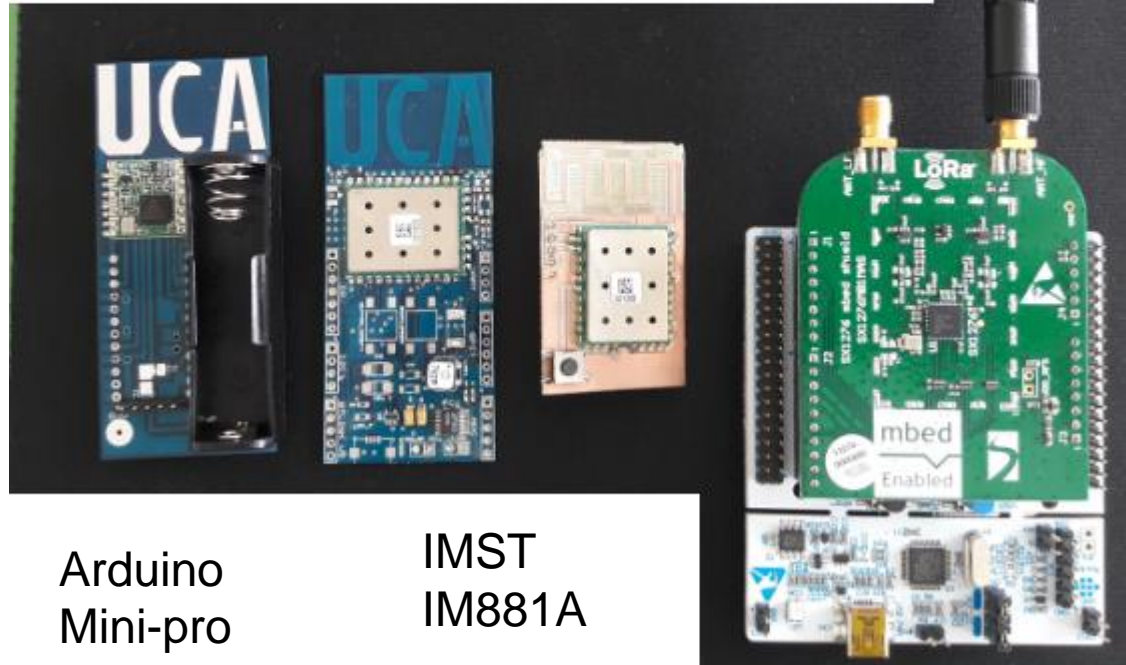
- ❑ No (wake-up time only) Rx Slot OR Rx Slot 1 OR Rx Slot 2 is used
- ❑ The energy drain when no downlink is \ll the Tx energy

LoRa Node

- Several platform available
- All with advantage and disadvantage

Arduino is the most simple one

Semtech

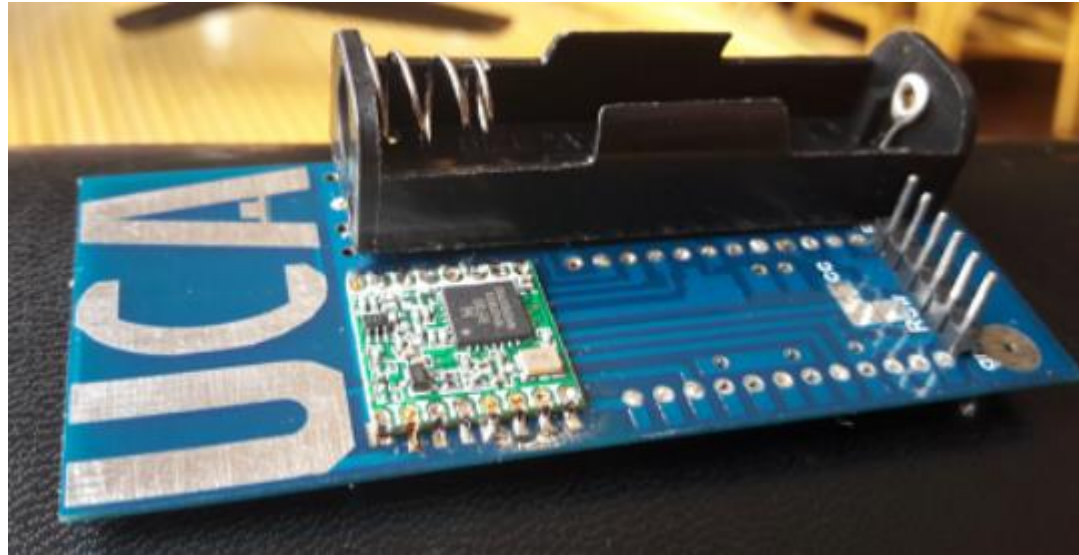


Arduino
Mini-pro

IMST
IM881A

LoRa with Arduino

Integrated antenna



Low cost platform :

- PCB < 1\$
- Arduino mini pro = 75000d (<http://hshop.vn/products/arduino-pro-mini-3-3v-8mhz>)
- LoRa RFM92W = 186000d (ebay)

Need a USB-UART programmer =55000d : <http://hshop.vn/products/mach-chuyen-usb-uart-cp2102>

Compatible with LoRa and LoRaWan

LoRa with Arduino



Prof. Cong Duc Pham from Pau University

Available code, tutorial, exemple of projects !

For Gateway and end-node

<http://cpham.perso.univ-pau.fr/>

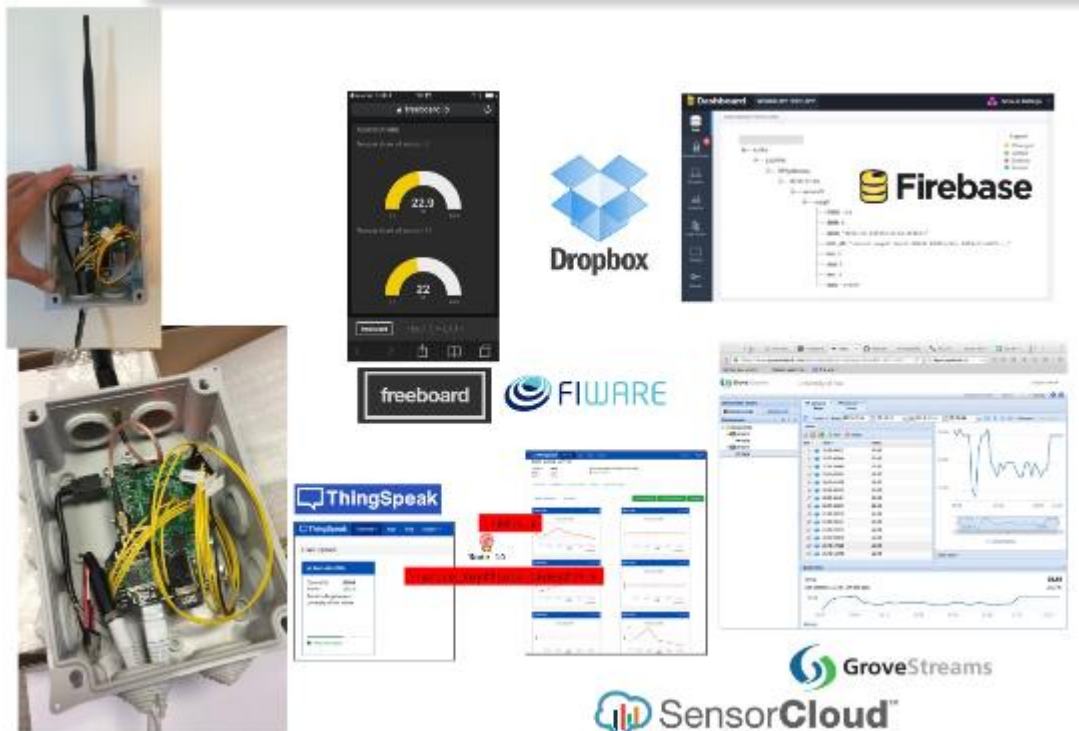
Codes available in : <https://github.com/CongducPham/LowCostLoRaGw>

Vetnam bands are 920-923MHz, it is explain how to change it in the code:

<https://github.com/CongducPham/LowCostLoRaGw/tree/master/Arduino#adapting-frequency-for-your-country>

LoRa with Raspberry Pi

Also tutorial to connect the gateway to the cloud



<https://github.com/CongducPham/LowCostLoRaGw>

Conclusion

- The first projects should be done on LoRa “only” systems
- As soon as we receive the LoRaWan gateway, the project can be migrated to LoRaWan (if needed)
- For future, boards with STM32 chip could be used (lower power consumption) but work on LoRaWan

TIME FOR THE DEMO

References

- “LoRa modulation basics”
<http://www.semtech.com/images/datasheet/an1200.22.pdf>
- CongDuc Pham webpage :
<http://cpham.perso.univ-pau.fr/>

Demo Steps by Steps

- How to connect a LoRa end-node with a LoRa Gateway
- Go to website
<https://github.com/CongducPham/LowCostLoRaGw>
- Click « Clone or download »
- Unzip the downloaded file
- Install Arduino IDE :
<https://www.arduino.cc/en/Main/Software>
- Copy all library in « \LowCostLoRaGw-master\Arduino\libraries » to your Arduino library core directory
- Open Arduino and Arduino_LoRa_Gateway_1_4.ino project

Demo Steps by Steps

- In Arduino_LoRa_Gateway_1_4.ino code, **uncomment the #define PABOOST line**

```
// IMPORTANT when using an Arduino only. For a Raspberry-based gateway the distribution uses a radio.makefile file
//
// please uncomment only 1 choice
//
// uncomment if your radio is an HopeRF RFM92W, HopeRF RFM95W, Modtronix inAir9B, NiceRF1276
// or you known from the circuit diagram that output use the PABOOST line instead of the RFO line
#define PABOOST
//
```

- Several parameters can be changed, but the LoRA modes, SF and frequency band must be the same between the GW and the node
- It is time to connect your board

Demo Steps by Steps

- Connection between the USB-UART programmer to the Arduino mini pro
 - Vcc – 3.3V
 - Rx -> Tx
 - Tx -> Rx
 - Gnd -> Gnd
 - GRN-> DTR-RST
- Select the right port in Arduino IDE
- Compile and upload the code on your board (the UART-USB should blink during the operation)
- Open your serial monitor and use 38400 baud

Demo Steps by Steps

- You should have this screen :

```
401 bytes of free memory.  
SX1272 detected, starting  
...  
^$*****Power ON: state 0  
^$Default sync word: 0x12  
^$LoRa mode 1  
^$Setting mode: state 0  
^$Channel CH_10_868: state 0  
^$Set LoRa power dBm to 14  
^$Power: state 0  
^$Get Preamble Length: state 0  
^$Preamble Length: 8  
^$LoRa addr 1: state 0  
^$SX1272/76 configured as LR-BS. Waiting RF input for transparent RF-serial bridge
```

- Now the gateway is ready to receive data

Demo Steps by Steps

- Now open a second Arduino IDE program (not a new window, really a second application)
- Open the Arduino_LoRa_Ping_Pong.ino sketch
- In the code, **uncomment the #define PABOOST line**

```
// IMPORTANT when using an Arduino only. For a Raspberry-based gateway the distribution uses a radio.makefile file
//
// please uncomment only 1 choice
//
// uncomment if your radio is an HopeRF RFM92W, HopeRF RFM95W, Modtronix inAir9B, NiceRF1276
// or you known from the circuit diagram that output use the PABOOST line instead of the RFO line
#define PABOOST
//
```

- Check that the gateway and node configuration are the same (frequency band, mode, etc ...)
- Upload your code on the board

Demo Steps by Steps

End-node side

```
Simple LoRa ping-pong with the gateway
Arduino Pro Mini detected
ATmega328P detected
SX1276 detected, starting
SX1276 LF/HF calibration
...
Setting Mode: state 0
Setting Channel: state 0
Setting Power: state 0
Setting node addr: state 0
SX1272 successfully configured
--> CAD duration 547
OK1
--> waiting for 1 CAD = 62
--> CAD duration 549
OK2
--> RSSI -124
Sending Ping
wait for ACK

Starting 'getACK'
## ACK received:
Destination: 8
Source: 1
ACK number: 0
ACK length: 2
ACK payload: 0
ACK SNR of rcv pkt at gw: -3
##

Packet sent, state 0
Pong received from gateway!
```

Gateway side side

```
401 bytes of free memory.
SX1272 detected, starting
...
^$*****Power ON: state 0
^$Default sync word: 0x12
^$LoRa mode 1
^$Setting mode: state 0
^$Channel CH_10_868: state 0
^$Set LoRa power dBm to 14
^$Power: state 0
^$Get Preamble Length: state 0
^$Preamble Length: 8
^$LoRa addr 1: state 0
^$SX1272/76 configured as LR-BB. Waiting RF input for transparent RF-serial bridge
## ACK set and written in FIFO ##
## ACK to send:
Destination: 8
Source: 1
ACK number: 7
ACK length: 2
ACK payload: 0
ACK SNR last rcv pkt: 6
##

^$ACK requested by 8
--- rxlor. dst=1 type=0x18 src=8 seq=7 len=4 SNR=6 RSSIpkt=-42 BW=125 CR=4/5 SF=12
^p1,24,8,7,4,6,-42
^r125,5,12
ybpPing
```

Demo Steps by Steps

- With the board very close, your RSSI should be form -20 to -45
- If not, try to upload the code again
- Add a battery to the node
- And run !
- Then you can add sensor, actuator ...
- Let's start your project