FOSDEM

SAMSUNG
Open Source Group

TIZEN: RT

A lightweight RTOS platform for low-end IoT devices

#FOSDEM, IoT Track
Brussels, Belgium <2018-02-04>

Philippe Coval
Samsung Open Source Group / SRUK
philippe.coval@osg.samsung.com

Who is Philippe Coval?

SAMSUNG
Open Source Group

- Software engineer for Samsung Research
  - Open Source Group, EU team (@UK + DE + *FR* + CZ...)
- Interest: IoT, demos, usages, OS/hardware support, community
  - Projects: IoTivity, Tizen, Yocto, Automotive, etc
- Ask me online for help:
  - https://wiki.tizen.org/wiki/User:Pcoval

Agenda

SAMSUNG
Open Source Group

- Technology overview:
  - **Tizen**, the OS of everything ?
  - Software platforms
  - for **Low end** devices
    - NuttX, Tinyara, TizenRT
    - & relationships
  - Features & differentiation
- Crash course:
  - build sources,
  - log in, shell & apps
  - develop apps:
    - Native or Javascript?
- Demo & tips
- Resources and QA

What do you know of TIZEN ?

SAMSUNG
Open Source Group

- Tizen is an **Operating System** based on FLOSS
  - Powered by Linux Kernel
  - Open to platform and application developers:
    - Native (EFL) , Web (HTML5), .Net...
- Shipped into **consumer electronics** products
  - TVs, Wearables, Home kitchen appliances...
  - With connectivity features (OCF, SmartThings. S-Connect)
- Supports: Architecture: ARM/Intel 32/64 bits (+ SBC)
  - 2013: Tizen-2.0: RD210: 1.2 GHz
  - 2014: Tizen-2.2 Gear2: 1GHz, 512MB, +4GB
  - 2016: Tizen-3 Wearable GearS3: 1GHz*2, 768MB





SAMSUNG
Open Source Group

"I'm not crazy. My **reality**
is just different from yours."
~ Lewis Carroll

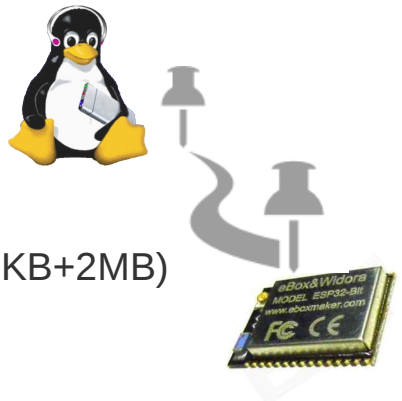
Tizen the OS of Everything?

SAMSUNG
Open Source Group

- What about **low-end** and low-cost IoT devices?
  - Without display or rich UI/UX (just small LCD or buttons)
  - Battery powered, low consumption matters!
  - running ubiquitous **micro controllers** (MCU)
    - Cheap: Cost (<10$) and usage (Low Consumption <mW)
    - But can be very constrained in RAM+ROM
      - RFC7228: Class 0 from <10KiB+100KiB
- TizenRT is targeting middle configurations (RAM+Flash):
  - Cortex M3 (30KB+512KB), M4 (256KB+16MB), R4 (2MB+16MB)

Different software stack for low end devices

SAMSUNG
Open Source Group



- For developers, there is gap between:
  - **Linux** kernel is flexible to some extend,
    - Typically: RAM=8MB + ROM=2MB (down for XIP: RAM=1MB ROM=4MB)
    - uClinux is lowering requirements (No MMU, Reported, STM32F4's M4: 256KB+2MB)
  - Baremetal: Optimal but not flexible: costly, slow
- For dedicated Oses: Genericity vs Speciality (~ trade of).
- Consider features, requirements, **learning curve** (tools), **licensing**
  - BSD: **NuttX**, TinyOS, MIT: FreeRTOS* Contiki,
  - Apache: Mbed, Zephyr, GPL: RIOT, ChibiOS, Inferno..
  - ~~Unfree/Closed source: NucleusOS, eCosPro, ThreadX, VxWorks, QNX, uC/OS, RTX...~~

TizenRT's origin

SAMSUNG
Open Source Group

- **TinyAra** 2015 project to collect, store, and deliver IoT sensor data using
  - IoTivity: IoT framework for seamless connectivity (+LWIP port for IPv6, +LWM2M)
  - AraStorage: Data management (SQL, b+ tree index)
  - sources released to public on tizen.org (SOSCON2016)
- Based on **NuttX RealTime** kernel (deterministic and priorities)
  - Initial release in 2007, community led by main author Gregory Nutt
  - Stable & Mature (2.5M LoC)
  - BSD Licensed:
    - used in many other OS projects or products, Industry adoption
      - PX4/PixHawk (Drones), Thingsee (IoT box)
- TizenRT is the whole stack: TinyAra kernel + middlewares (TDC2017)

NuttX is easy for Linux devs

SAMSUNG
Open Source Group

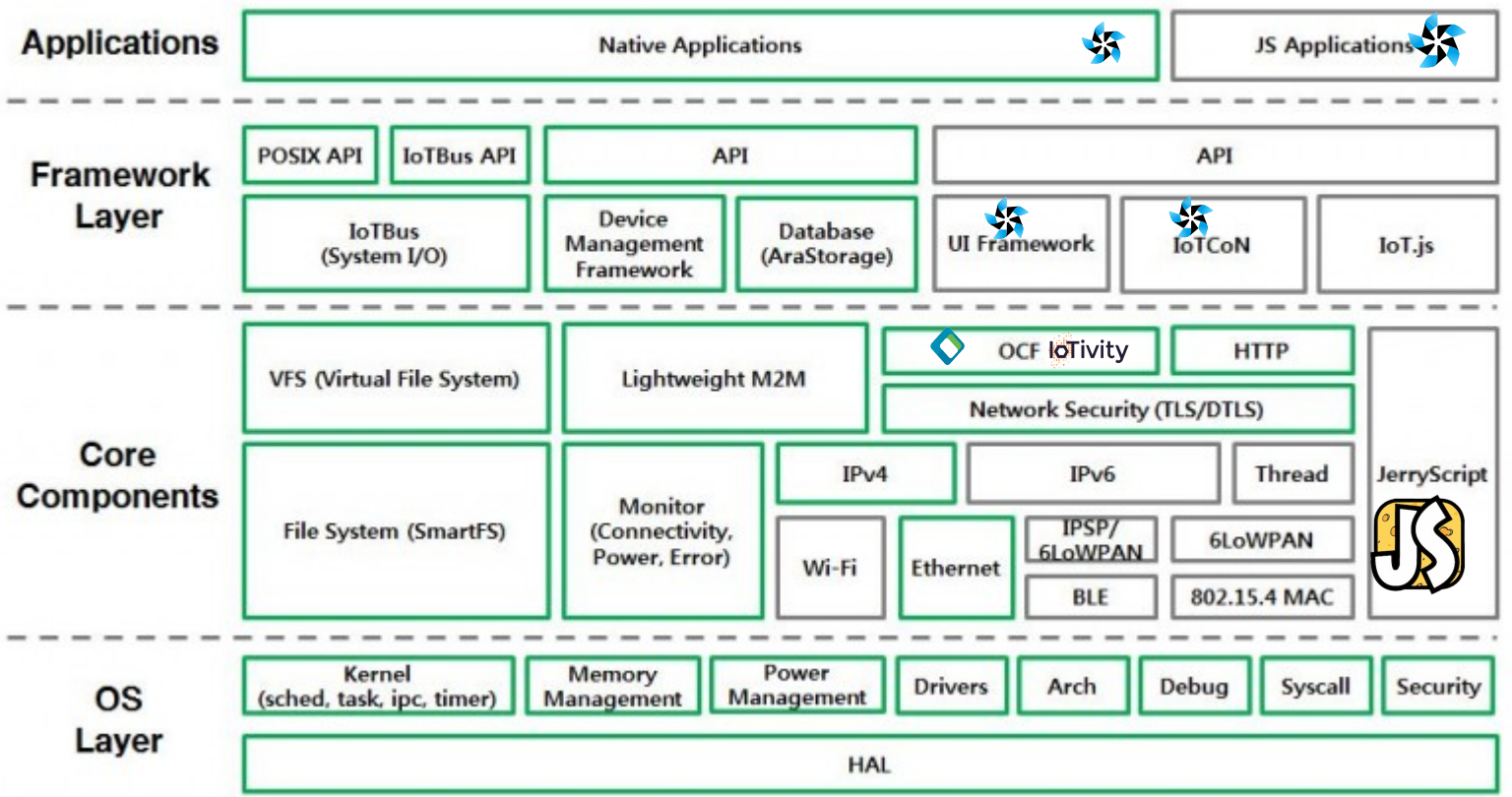
- **UNIX/Linux** inspired OS (+ shell & apps)
- Comply with **standards** (POSIX/ANSI):
  - uses custom C library & C++ (uClibc++ or LLVM)
  - uses GNU tools, GCC, gmake, Kconfig, GDB, openocd...
- Filesystem (RO,RW), VFS (/dev/, /proc), MTD
  - handled by drivers (read, write, iotctl opts)
- Network: BSD sockets (uIP: TCP, UDP, IPv6, 6lowpan) NTP, FTP, HTTP etc
- Concurrency: Multi tasks & pthread support
  - +mutexes, message queues, signals, TLS, SMP, IPC, FIFO shed, preempt...
- **Modular** and configurable and scalable (Kconfig),
  - Low requirements: footprint is <16KB. Supports Many BSP (8 to 32 bits, 27 arch, 200 boards)

Tizen:RT Architecture and progress overview

SAMSUNG

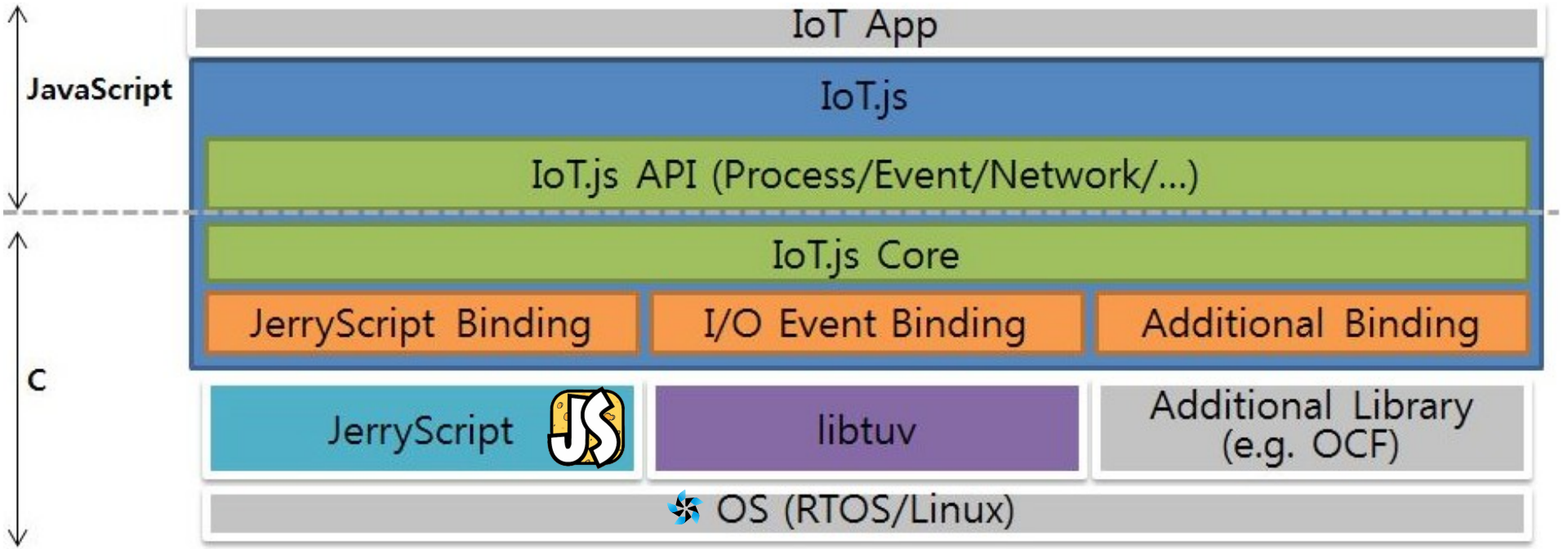
2016 2017

Current status: V1.1 + ~650 patches



JavaScript runtime:
“IoT.js is to NodeJs what JerryScript is to v8”

SAMSUNG
Open Source Group

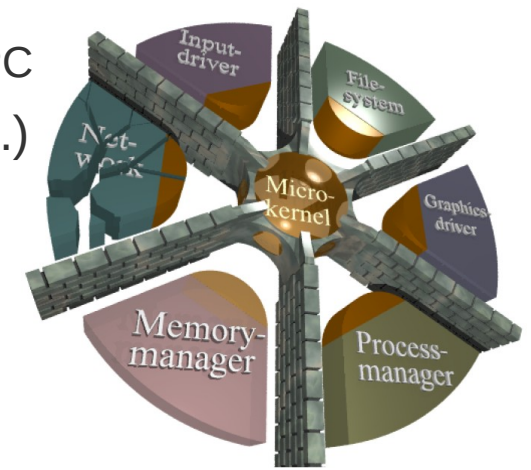


(For more, Welcome to FOSDEM's IoT room at 15h05 for Ziran's Sun presentation)

Tizen:RT plan for reliability

SAMSUNG
Open Source Group

- **Memory protection unit** (MPU > MMU)
  - User/Kernel separation
    - Per thread mem protection
- **Micro kernel** architecture
  - Only for scheduling tasks, memory, IPC
- + user space services (Net, drivers...)
- Fault Tolerance
  - Self healing
    - restart services (and dependencies)
  - Fallback option: Live update (DM)



Tizen:RT for IoT: Connectivity & Security

SAMSUNG
Open Source Group

- **Connectivity**:
  - Standard protocols: LWIP (IPv6), mDSN, DHCP, BSD/Web/Sockets, MQTT (Eclipse's mosquitto)
  - OCF's IoTivity: CoAP Discovery, Messaging (CA, RA, Cloud), Security (PM, DTLS, SRM)
  - WiFi, WPASupplicant, APIs for onboarding (ARTIK app using QR codes)...
  - Cloud: ARTIK=>SmartThings cloud, AWS... + S.Connect App
- Device Management Framework: Monitor connectivity & power, report
  - OMA-based Lightweight M2M
    - Over the air (FOTA...)
    - Eclipse Wakaama (formerly liblwm2m)
- More **security** features: Crypto (AES 128/256, RSA, ECC....)
  - Secure: Boot, Flash Storage, Channel (DTLS using mbedtls), Certs

IoTivity

Works with SmartThings™

Samsung ARTIK™ Cloud

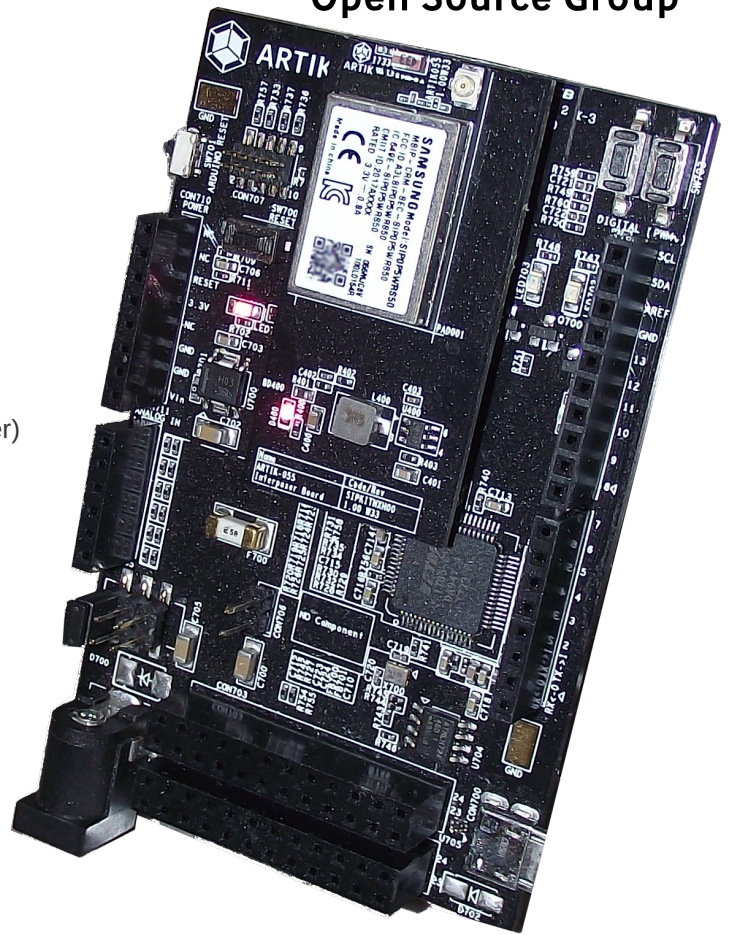
Samsung ARTIK™

Hardware support

SAMSUNG
Open Source Group

Samsung ARTIK™: 05x (053, 053s, 055, 055S)

- Ie: SoM ARTIK 055**S** + Interposer board:
  - 32-bit ARM® Cortex® **R**4 @ 320MHz
    - R for **Real Time** and safety critical
    - S for **Secure**: SE+TEE, Secure Sub System (sssfw.bin)
      - Crypto (Certs, Key, ECDSA, RNG, PUF),
      - Boot: ROM>bl1.bin> U-Boot (bl2.bin)> TizenRT OS (signed with artik05x_codesigner)
  - 1280KB RAM, 8MB Flash, WiFi (firmware wlanfw.bin)
  - IO: GPIO*29 (3.3VDC), ADC*4 (1.8V 12bits@6Mhz), PWM*5
    - UART*4 (2-pin), SPI*2, I2C*2, I2S, RTC, JTAG (lock)
- Others? : QEmu, SIDK_S5JT200
  - docs/HowToAddnewBoard.md
  - Or port NuttX BSPs (STM32, ESP32? ...)



SAMSUNG
Open Source Group

"**Simplicity**
is the ultimate sophistication."
~Leonardo da Vinci

First boot of “TizenRT ARTIK SDK” on 055s

SAMSUNG
Open Source Group

```
sudo screen /dev/ttyUSB1 115200
```

```
U-Boot 2017.01-g3129855-dirt

CPU:   Exynos200 @ 320 MHz
Model: ARTIK-053 based on Ex
DRAM:  722 KiB
WARNING: Caches not enabled
BL1 released at 2017-3-13 1
SSS released at 2016-12-30
WLAN released at ????-??-??
Flash: 8 MiB
 **** Warning - bad CRC, us

In:    serial@80180000
Out:   serial@80180000
Err:   serial@80180000
Hit any key to stop autobo
gpio: pin gpg16 (gpio 46)
## Starting application a
s5j_sflash_init: FLASH Qu
uart_register: Registerin
uart_register: Registerir
(…)
System Information:
        Version: 1.0
        Commit Hash: 137
(...)
```

```
TASH>>Hello, World!!

TASH>> help
        TASH command list
        -----------------
cat         cd          date        df
dhcpd       exit        free        getenv
heapinfo    hello       help        ifconfig
ifdown      ifup        iperf       kill
killall     logm        ls          mkdir
mkrd        mksmartfs   mount       ntpclient
onboard     ping        ps          pwd
reboot      rm          rmdir       security_api
setenv      sh          sleep       stkmon
tls_client  tls_server  umount      unsetenv
uptime      wifi
```

Explore system using TASH Shell and apps

SAMSUNG
Open Source Group

```
TASH>>mount
 /mnt type smartfs
 /proc type procfs
 /sss type smartfs
```

```
TASH>>cat /proc/version
     Version: 1.0
     Commit Hash: 13711f7a
     Build User: root@gate
     Build Time: 2017-08-1
```

```
TASH>>df
  Block   Number
   Size   Blocks   Used Available Mounted on
    512     2800     15      2785 /mnt
      0        0      0         0 /proc
    512     1024     14      1010 /sss
```

```
TASH>>cat /proc/partitions
 Name            Start     Size
 bl1                 0
 sssro
 bl2
 sssfw
 wlanfw
 os
 factory
 ota
 user
 nvram
 sssrw
```

```
TASH>>free
              total    used     free  largest
Data:        779296  153632   625664   610288
```

```
TASH>>http
TASH>>gpio
TASH>>cloud
TASH>>sdk modules
TASH>>sensorbd
```

```
TASH>>stkmon

TASH>>=================================
PID    STATUS      SIZE PEAK_STACK PEAK_HEAP    TIME THREAD
---------------------------------------------------------------
---------------------------------------------------------------
    0 ACTIVE      1024       1024      81840   30755 Idle Task
    (…)
    6 ACTIVE      4076        844       3584   30755 tash
    (…)
    3 ACTIVE      2028        300      10256   30755 logm

TASH>>killall logm
```

```
TASH>>wifi
TASH>>wifi startsta
TASH>>scan
TASH>>wifi join $SSID  $PASS
TASH>>ifconfig wl1 dhcp
```

```
TASH>>websocket connect wss://echo.websocket.org/
TASH>>mqtt_sub  -d -h 198.41.30.241 -t $SYS/
```

SAMSUNG
Open Source Group

"Talk is cheap.
Show me **the code**."
~ *Linus Torvalds*

Build from source

SAMSUNG
Open Source Group

- **git** clone \
  https://github.com/samsung/tizenrt
- cd tizenrt && TIZENRT_BASEDIR="$PWD"
- Setup toolchain
  - tar xvjf gcc-arm-none-eabi-4_9-2015q3-20150921-linux.tar.bz2 ; export PATH=${PWD}/...:$PATH

Configure:

- cd ${TIZENRT_BASEDIR}/os/tools
- ./configure.sh <board>/<configuration_set>
- make -C ${TIZENRT_BASEDIR}/os **menuconfig**

- Build
  - make -C ${TIZENRT_BASEDIR}/os
- Eventually sign image (artik05x_codesigner -h)
- Flash using **openocd**
  - make -C ${TIZENRT_BASEDIR}/os download

Add sample native app

SAMSUNG
Open Source Group

- Inspire from sample app: TizenRT/apps/examples/hello
  - hello_main.c:
  - Makefile
    - to build & register as a **TASH** command
  - Kconfig_ENTRY & Kconfig
    - to be enabled on build configuration
      - make menuconfig
  - Make.def
    - Build app in FW image (no DL, static libs)

```
#include <tinyara/config.h>
#include <stdio.h>
#ifdef CONFIG_BUILD_KERNEL
int main(int argc, FAR char *argv[])
#else
int hello_main(int argc, char *argv[])
#endif
{ printf("Hello, World!!\n"); }
```

- C API are mostly there
  - functions could missing: enable | use alternative | reimplement
- Exercise: create netcat app using <sys/socket.h> and <arpa/inet.h>

App sample JS app

SAMSUNG
Open Source Group

- Enable **IoT.js**
  - in menuconfig (runtime):
- To check: edit? simple javascript line:
  - `console.log(JSON.stringify(process));`
  - `TASH>> iotjs /mnt/index.js`

```
{ "env":{ (...) },
  "platform":"tizenrt", (...) ,
  "iotjs":{ (…) }
```

- Exercise:
  - Upload js from host to target
    - using netcat, websocket, http etc

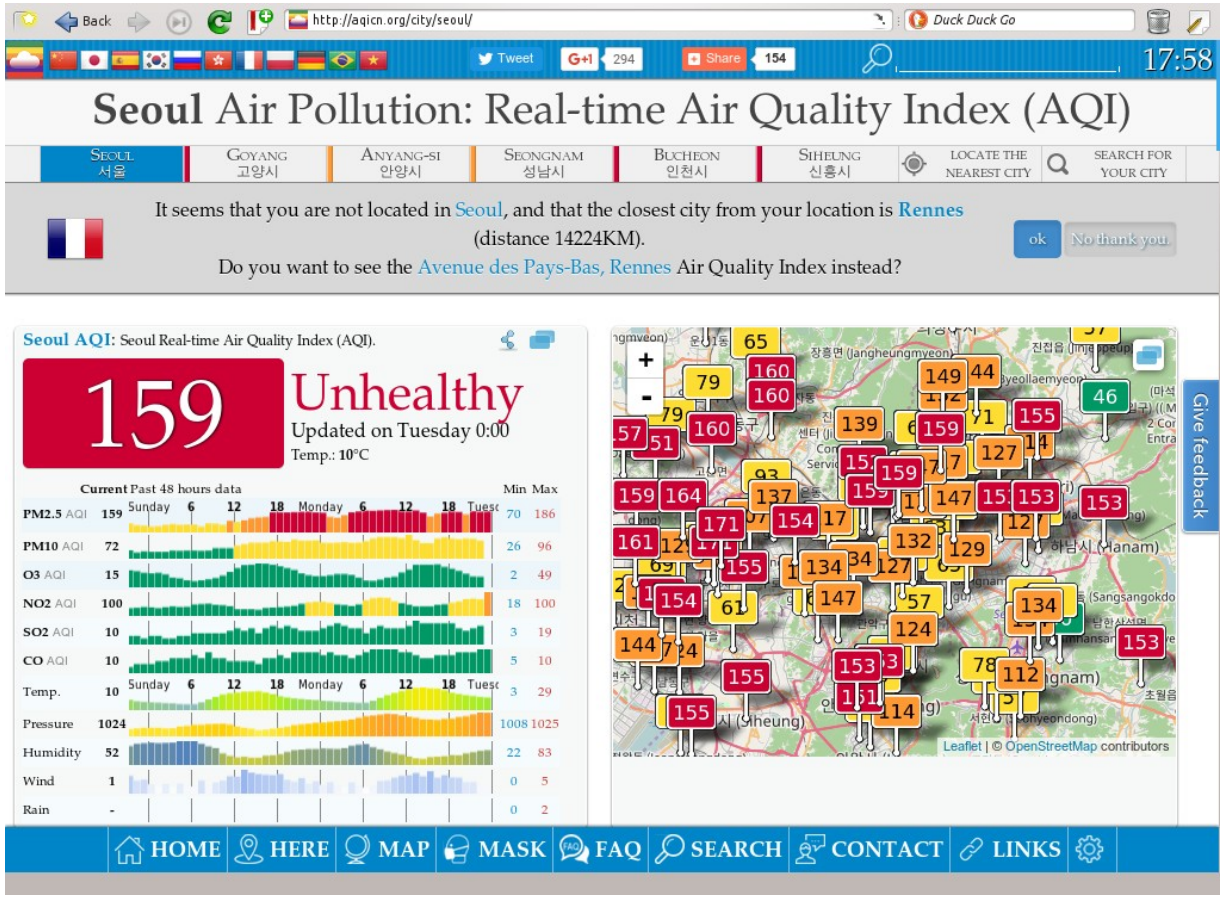
- Then load .js from **ROM partition**
  - Enable ROMFS with menuconfig:
    - CONFIG_FS_ROMFS=y
    - *_FLASH_PART_* (400, romfs, rom)
    - *_AUTOMOUNT_ROMFS_*: /dev/mtdblock11 on /rom
  - Install javascript to FS dir and rebuild FW:
    - ./TizenRT/tools/fs/contents/*.js
  - `TASH>> iotjs /rom/index.js`
- Exercise:
  - Make a luncher app to run iotjs at boot

SAMSUNG
Open Source Group

"The secret to getting ahead
is getting **started**."
~Mark Twain

First world problems: Air Quality

SAMSUNG
Open Source Group



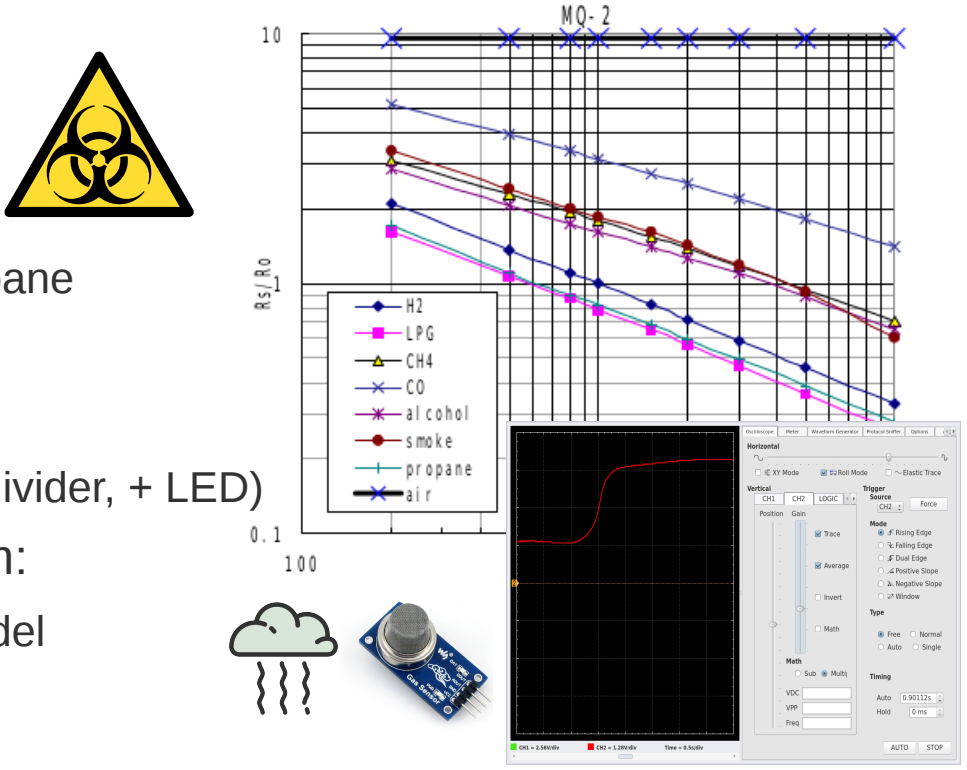
- World Health Org. :
  - 92% of population
  - 11% cause of death

- Survival plan:
  - Collect and present **data**
    - To **systems** or **users**
  - **Change** behaviors

Monitoring air quality? let's prototype:

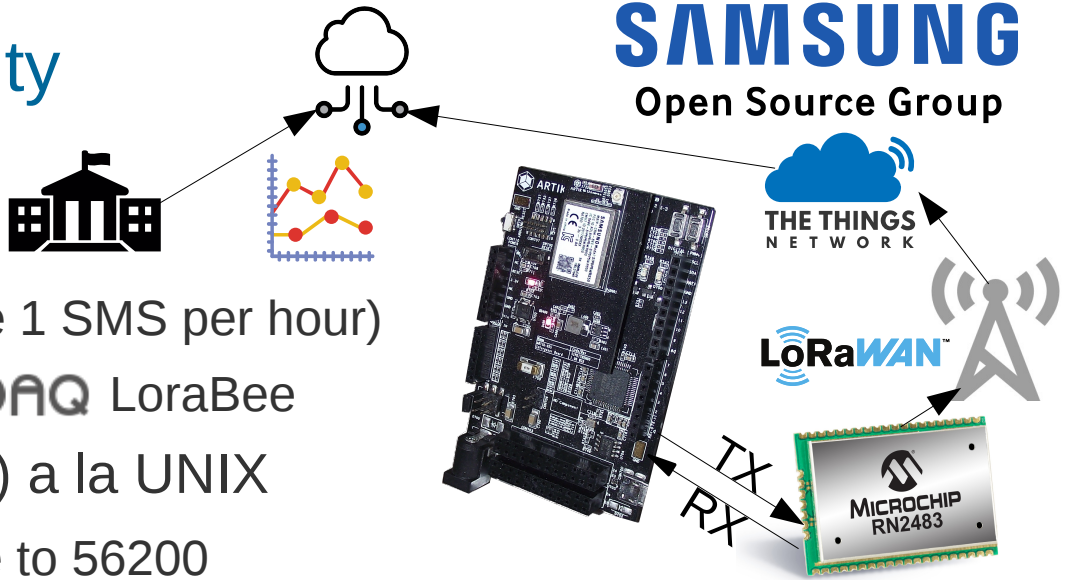
SAMSUNG
Open Source Group

- Input Sensor(s):
  - Particle Pollution (PM), Dust, Gas
    - AQ-*, MQ-*, MG-811, DSM501A
  - MQ-2 Analog **5V** sensor for CO, smoke, propane
- TizenRT supports **analog** inputs
  - + IoT.js module to handle ADC (/dev/adc0)
  - For ARTIK S55 **1.8V** in pin A0 (Add Voltage divider, + LED)
- Let's create a javascript class inspired from:
  - W3C® **generic sensor** and OCF airquality model
  - Configure, loop on read, emit ondata event



Emit radio alerts to Smart City

SAMSUNG
Open Source Group



- Using LpWan (LoRa, SigFox...)
  - SubGHz low bandwidth radio (like 1 SMS per hour)
  - Try LoRa® RN2483 modem: SODAQ LoraBee
- TizenRT supports UART (RX/TX) a la UNIX
  - KConfigure **/dev/ttyS1**'s baudrate to 56200
  - Use IoT.js UART module to handle port
    - (First I prototyped on node.js then ported serialport.js)
- Register device on favorite LoRaWAN network
  - BTW: Greetings to Rennes.fr's IoT communities.

```
> mac set devaddr BADC0DE1
< ok
> mac set nwkskey B4DC0DE2...
< ok
> mac set appskey BADC0D33...
< ok
> mac join abp
< ok
< accepted
> mac tx uncnf 1 142
< ok
< mac_tx_ok
```

SAMSUNG
Open Source Group

"Any sufficiently advanced technology is indistinguishable from **magic**."

~ *Arthur C. Clarke*

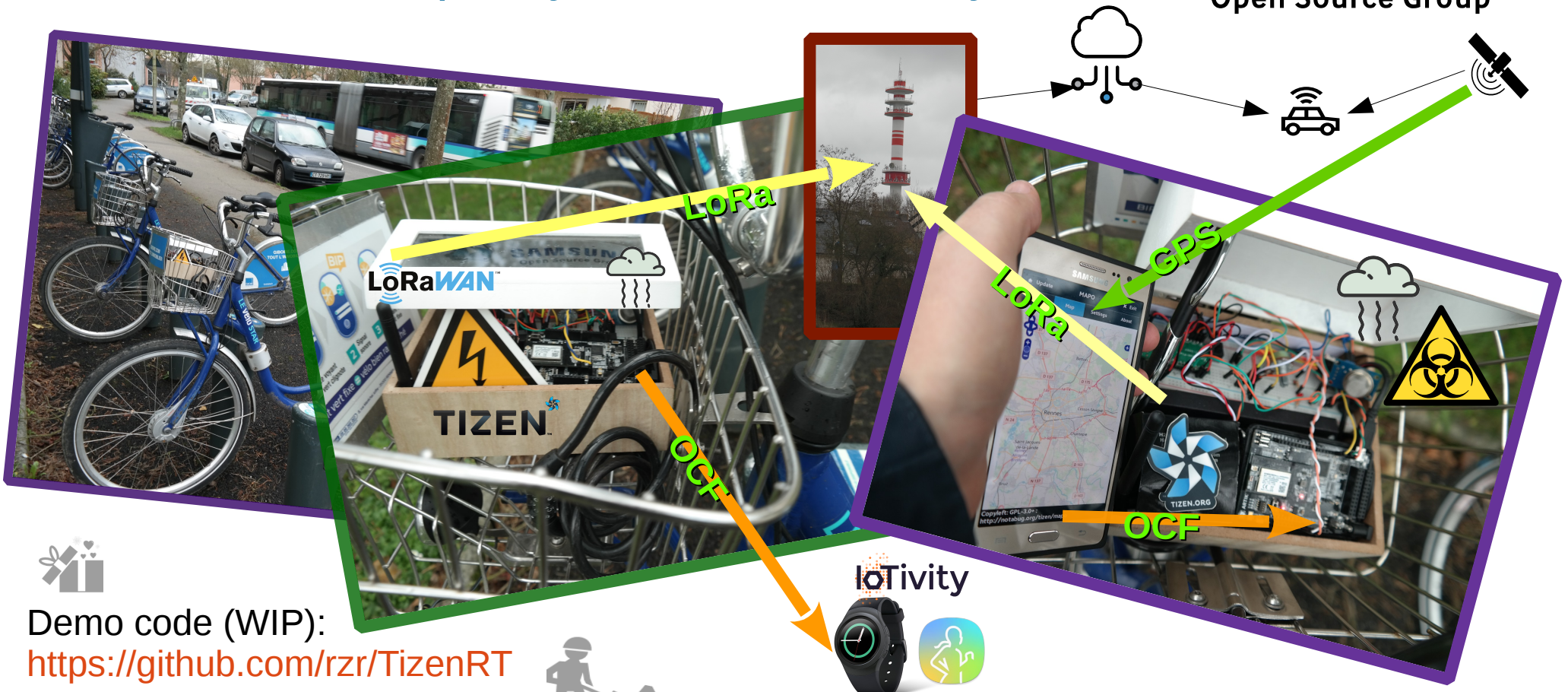
Live demo !

https://youtu.be/S7zpBpnpfIU#tizen-rt-lpwan-20180204rzr

SAMSUNG Open Source Group

DIY: mobile air quality monitor, and beyond?

SAMSUNG
Open Source Group



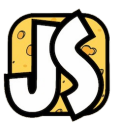
Demo code (WIP):
https://github.com/rzr/TizenRT

Summary

SAMSUNG
Open Source Group

- Tizen:1,2,3,4... is based on Linux kernel for high end devices
- Tizen ecosystem is now addressing more **constrained** IoT devices
  - Tizen:RT was originally based on NuttX kernel,
    - then evolved into micro kernel architecture

- Tizen:RT is open to application developers by providing
  - **Native**, C, C++ POSIX standard libraries
  - Middleware: Security, Connectivity, Device management, Database, IoTbus, Audio...
  - **Javascript** runtime, using IoT.js + JerryScript interpreter



- **Open Community** (github.com/Samsung) & Tools: SDK (CLI or IDE)

References

SAMSUNG
Open Source Group

- Entry points:
  - https://fosdem.org/2018/schedule/event/tizen_rt/
  - https://wiki.tizen.org/Category:RT
  - https://github.com/Samsung/TizenRT/
- Keep in touch online:
  - https://blogs.s-osg.org/author/pcoval/
  - https://wiki.tizen.org/wiki/Meeting
  - https://wiki.tizen.org/wiki/Events
  - https://wiki.tizen.org/wiki/FOSDEM

Bonus tip:
for demo convenience
I am drafting helper recipes (WIP)
to rebuild all from scratch using docker:

git clone https://github.com/rzr/TizenRT
./run.sh
make help menuconfig deploy run console

More:

- https://source.tizen.org/documentation/tizen-rt
- https://www.artik.io/modules/artik-05x/
- https://developer.artik.io/documentation/artik-05x/getting-started/
- http://www.nuttx.org/
- http://youtube.com/channel/UC0QciIlcUnjJkL5yJJBmluw
- https://www.slideshare.net/MyungJooHam/tdc-mjham-armkernelintizen3noani
- http://opensource.samsung.com/

FOSDEM

SAMSUNG
Open Source Group

Thank you

Merci, Danke Schoen, Gracias,
맙습니다 , ありがとう , 谢谢 ,شكرا, Trugarez !

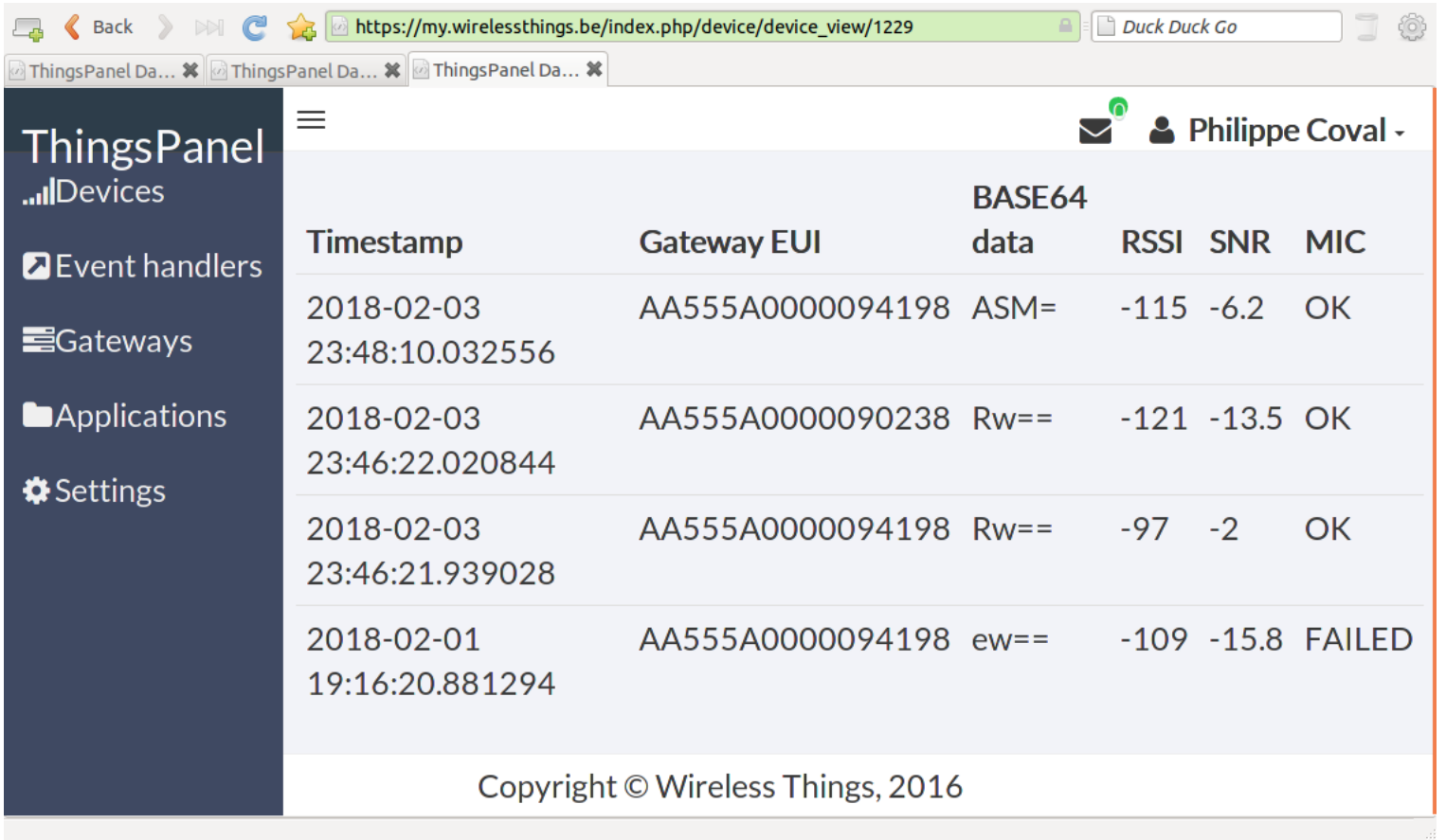
https://wiki.tizen.org/wiki/User:Pcoval

BTW greetings to :

Gulliver LAB FAB ambassad'air THE THINGS NETWORK acklio Institut Mines-Télécom Wireless THINGS Samsung ARTIK™

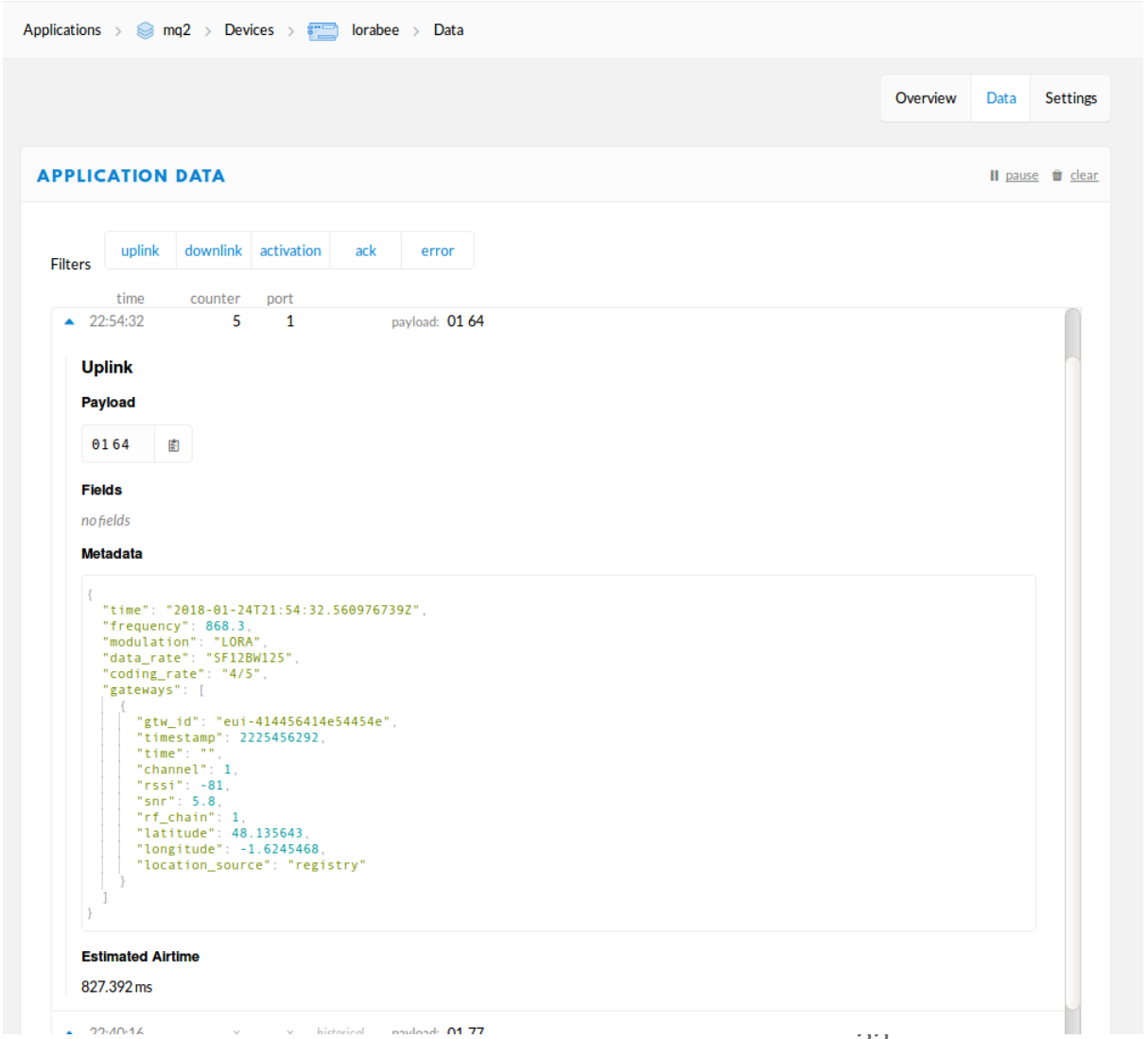
Resources: flaticons CC

SAMSUNG
Open Source Group



Wireless THINGS

FOSDEM



SAMSUNG
Open Source Group

THE THINGS
NETWORK