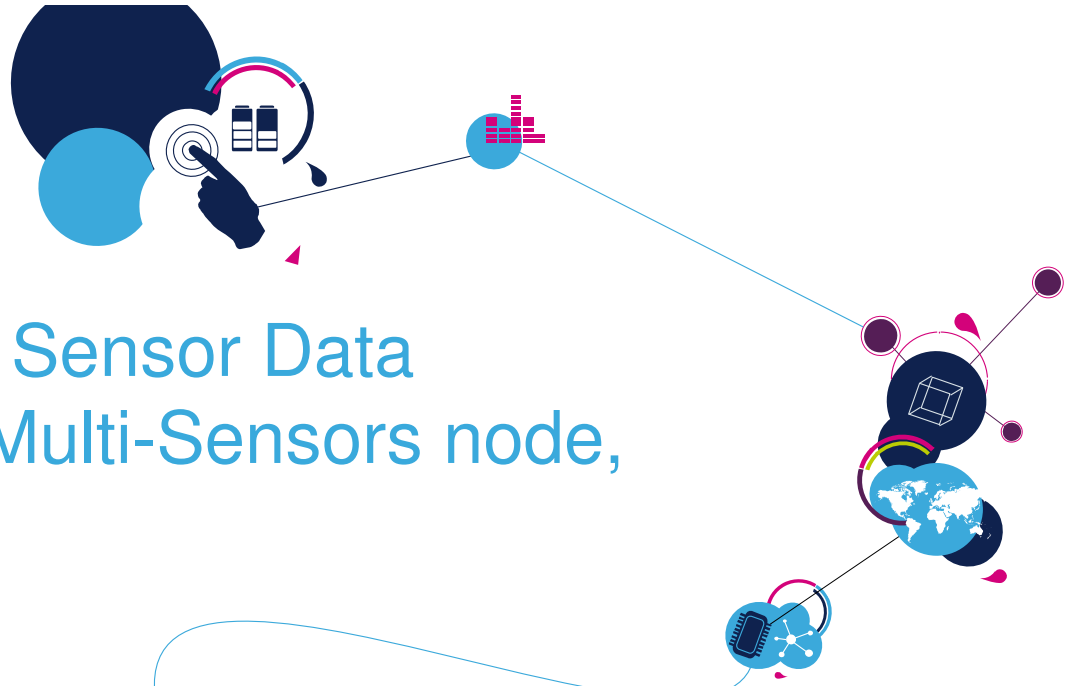


Simplifying Integration of Sensor Data Using the NFC Enabled Multi-Sensors node, STEVAL-SMARTAG1

Hands-on Workshop

John Tran
STMicroelectronics

life.augmented

Technology Tour 2019
Minneapolis, MN | October 24

What’s NFC?

life.augmented

Radio Frequency IDentification

RFID is a short range contactless communication technology

Employs an active reader/writer and a passive tag/transponder

The reader powers the tag and initiates the communication

Frequency Bands

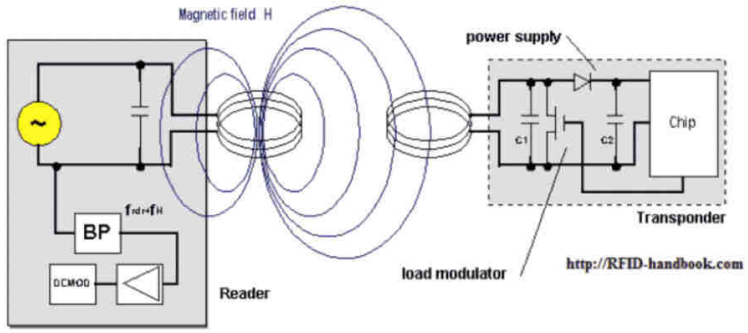
- LF (120-150 KHz)
- HF (13.56 MHz)
- UHF (433 to 960 MHz)

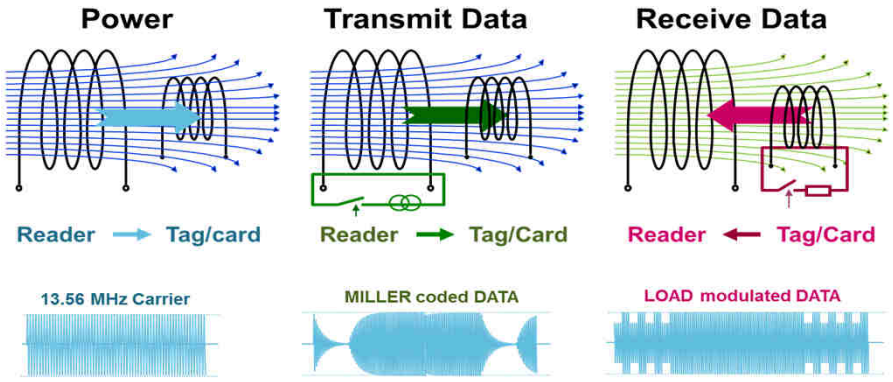
Operating ranges

- Proximity (few cm)
- Vicinity (up to 1m)
- Long Range (up to 10m)

Applications

- Transit, payment, inventory tracking, building and car access, etc.





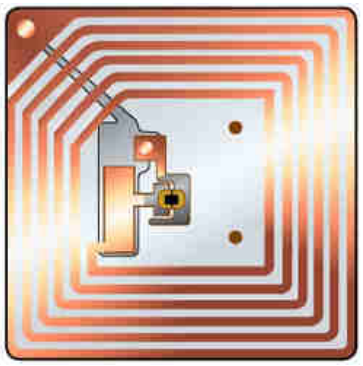
ISO14443 signaling

RFID Technologies at a Glance

RFID	LF	HF	UHF
Coupling mode	Inductive	Inductive	Electro-magnetic backscatter
Operating frequency	125kHz – 134kHz	13.56MHz	860MHz – 960MHz
Antenna	Coil	Coil	Dipole
Max operating distance	up to 1m	Vicinity: up to 1m Proximity: up to 10cm	~10m
Regulation	Worldwide harmonized	Worldwide harmonized	Different regulations per country
Standards	ISO14223 ISO18000-2	ISO14443 A/B ISO15693 ISO18092 ISO18000-3 **NFC Forum**	ISO18000-6 B/C EPC Class 1 Gen 2 **RAIN RFID**
Environmental influences	Small influence on operating distance Works in metal and industrial environment	Small influence on operating distance Works in metal and industrial environment	Influence on operating distance by reflection and absorption (metal and liquids)
Applications	Animal tagging	Product identification Public transport / Libraries Access control / Payment	Pallets and container ID Retail / Logistics Authentication
ST solutions		X	X

Comparison of RFID vs. Barcode

- Works in harsh and contaminated environments
- Options to implement data security / encryption
- Protocol supports “anti-collision” which allows reading/writing of an individual tag when multiple tags are in the reader field
- Unlike barcodes, line of sight is not necessary
- Offers data storage with options to interface to local processing (i.e. MCUs, FPGAs, etc.)



life.augmented

NFC Technology at a Glance

An interactive technology enabling engagement with IoT devices

- Near Field Communication, a short range wireless technology
  - Operating at 13.56MHz
  - Based on the RFID HF standard (ISO14443 & ISO15693)
- Interactive and zero power, enabling convenient connection to the Internet of Things

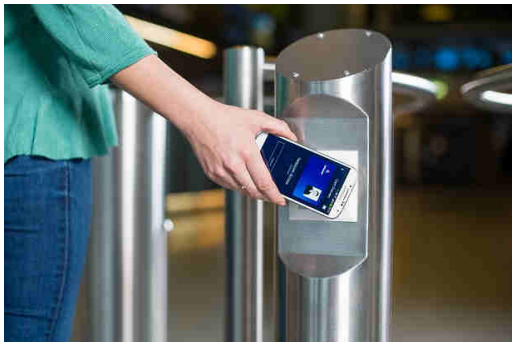
➔ NFC-enabled mobile phone can engage with items by a simple tap

- NFC is maintained by the NFC Forum
  - Ensures Interoperability between devices
  - Standardized use cases (web link, Bluetooth handover,…)
- Fast growing deployment in Mobile phone
  - Since 2010 virtually all Android phones support NFC
  - Apple has use NFC ApplePay since 2014, and in 2017 Apple added support of NFC reader mode in iOS11 onward

life.augmented

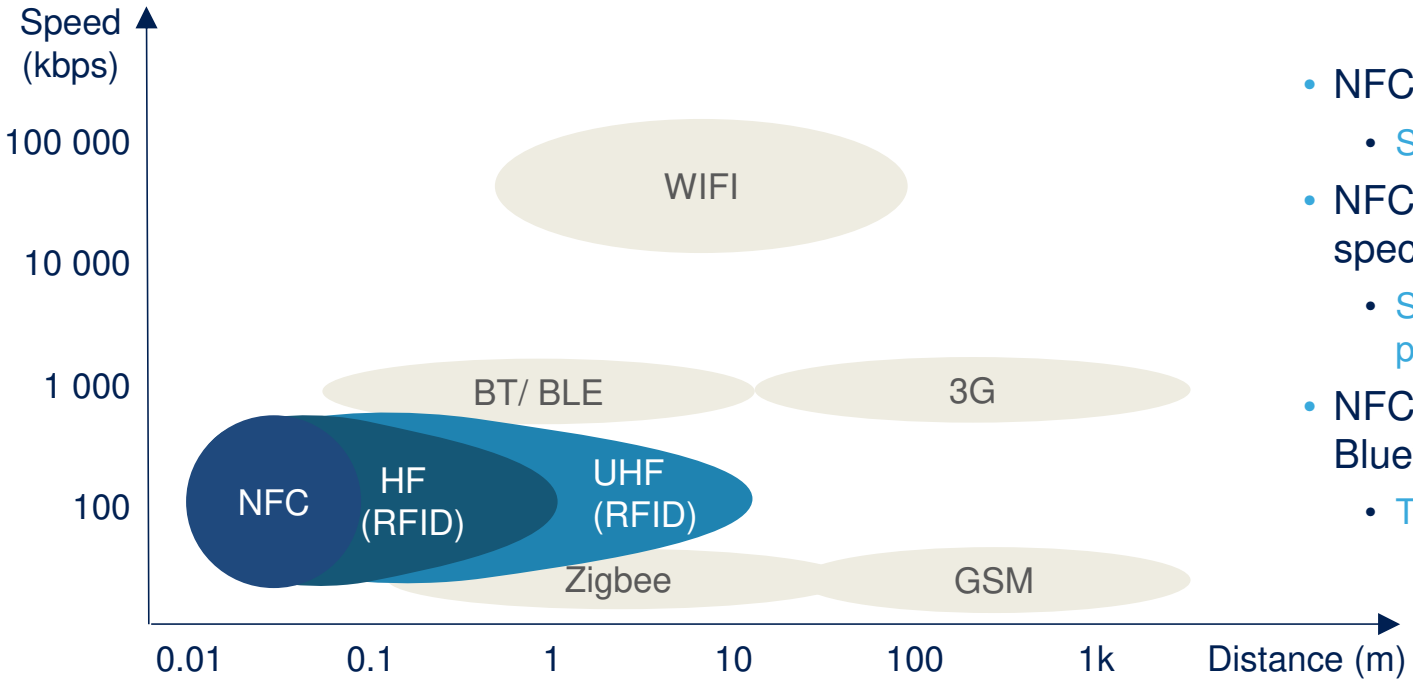
NFC in Depth

- Requires an action such as bringing your card/phone near the reader in order to use
- Maximum data transfer rate is 424 kbps
  - Proprietary modes can go up to 6.8Mbps
- NFC operating modes
  - Read/Write (reader-to-passive tag/card)
  - **Card Emulation (e.g. Apple Pay, Android Pay)**
  - **Peer-to-Peer (e.g. reader-to-reader)**
- Applications include access control, payments, electronic passports, transportation ticketing, device pairing, data file exchange (e.g. Android Beam)
- Various combinations of memory and security
- Standards and specifications
  - ISO14443A, ISO14443B, Sony FeliCa, ISO15693



life.augmented

NFC in the Wireless Spectrum



- NFC is a flavor of RFID
  - Subset of RFID HF standard: 13.56MHz
- NFC is complementary in the wireless spectrum
  - Short distance, Low data-rate & Zero power for applications
- NFC is complementary to Wi-Fi and Bluetooth technologies
  - Tap & Pair: convenient pairing use case

life.augmented

History of NFC

- 1983 RFID Patented
- 2003 NFC approved as an ISO/IEC standard
- 2004: Philips, Sony and Nokia create the NFC Forum
- 2006: First specifications for NFC tags are released
- 2006 Nokia releases the Nokia 6131 NFC compatible phone
- 2007: Commercial roll-outs in US, Europe, and China
- 2008: NFC Forum membership increases to +150 members
- 2010: Samsung releases the first Android-based NFC phone, the Nexus S
- 2011: Google Wallet introduced
- 2012: NFC in Windows 8
- 2013: Apple Pay introduced. Rolled out to mass market in 2014.
- 2017: Apple Core NFC introduced in iOS11

life.augmented

NFC Forum Tag Types

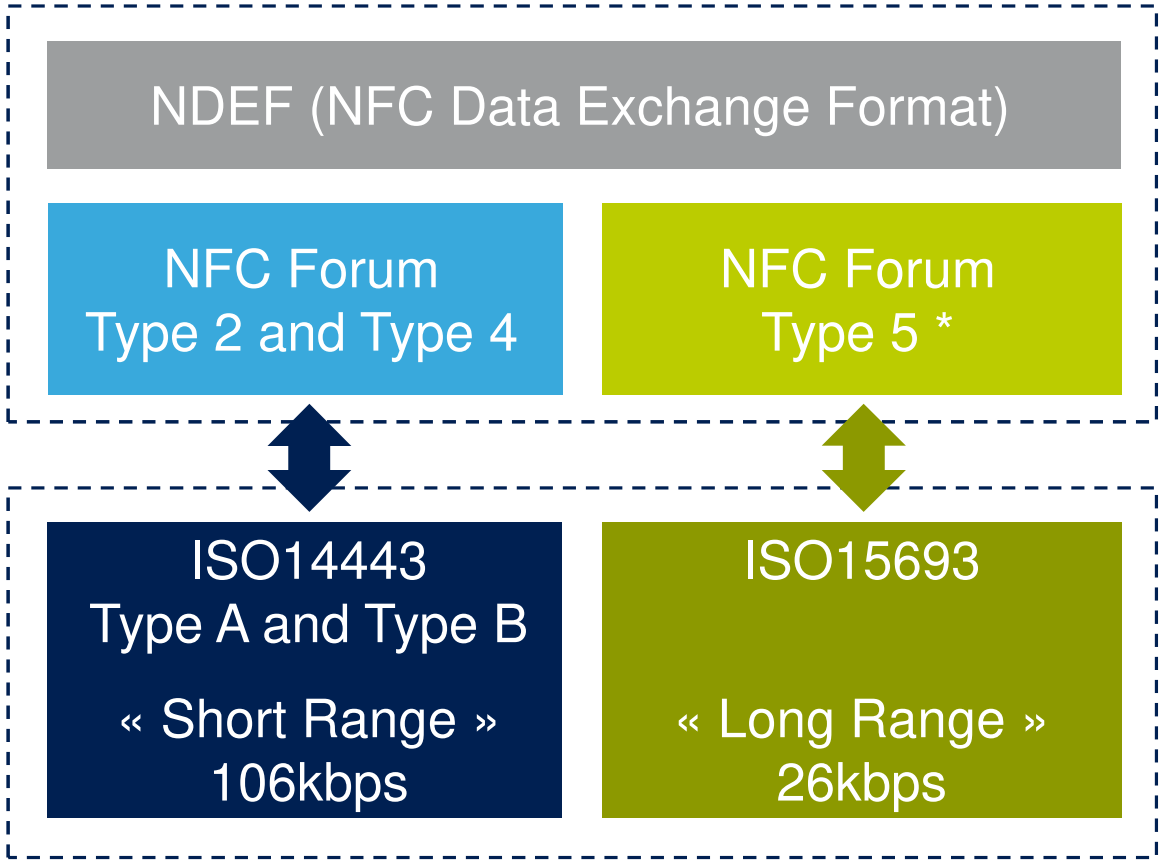
	Type 1	Type 2	Type 3	Type 4	Type 5
Products	BROADCOM "Topaz"	NXP NTAG MIFARE	SONY "FELICA"	ST25T(A&B) DESFire	ST25TV iCode
Specification	ISO 14443-A	ISO 14443-A	JIS X 6319-4	ISO 14443-A/B	ISO15693
Data Rate	106 kbit/s	106 kbit/s	212/424 kbit/s	106-424 kbit/s	26kbit/s
Protocol	Specific command Set	Specific command Set	FeliCa protocol	ISO 14443-4 ISO 7816-4	ISO/IEC 21481
Cost	Low	Low	Moderate	Moderate	Low
Use cases	Tags with small and fixed memory for single applications		Flexible tags with larger memory offering multi-application capabilities.		Long range tags with multiple applications
Memory type	Memory cards		CPU cards/Memory Cards0		Memory Cards

life.augmented

NFC Standards

NFC specification
→ **Upper layer SW**

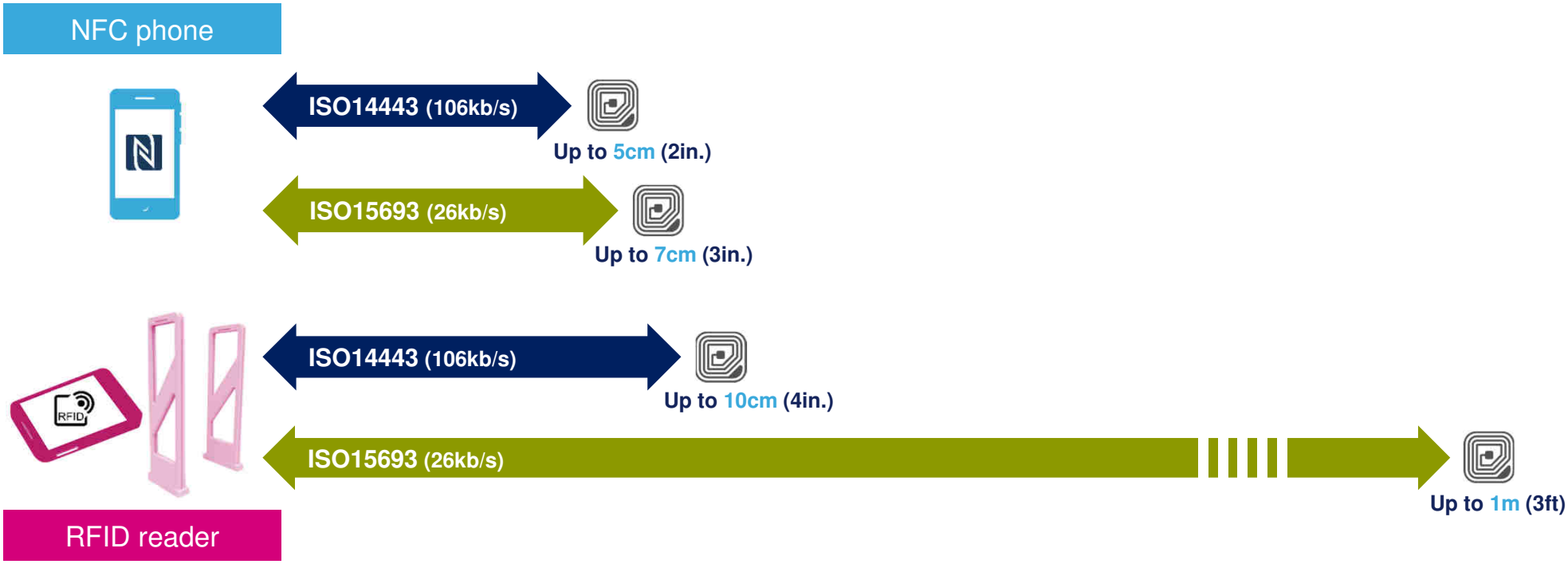
RFID HF ISO standards
→ **HW / SW protocol**



(*) ISO15693 integrated in NFC Forum specifications in October 2015 as NFC Forum type 5 (aka type V)

life.augmented

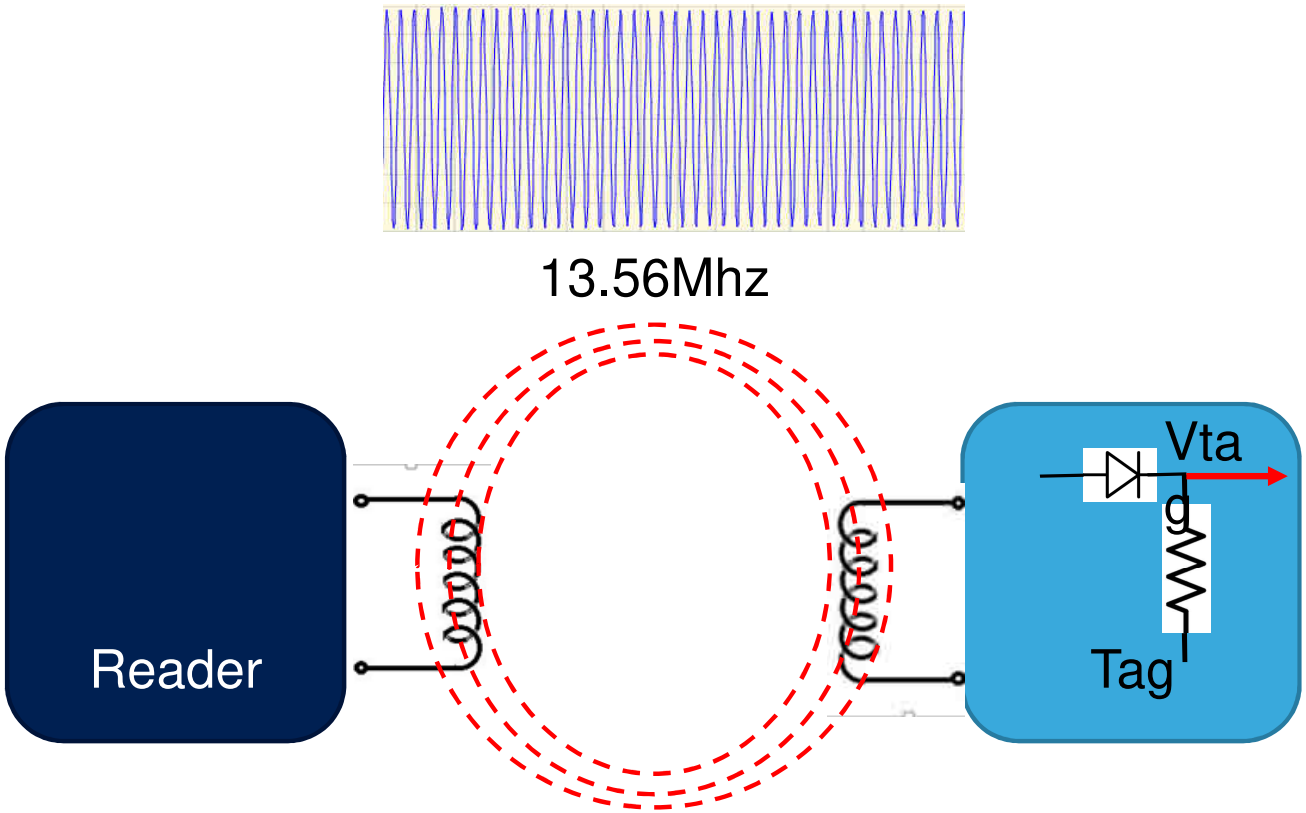
Typical NFC / RFID range



- ISO14443 is called « **short range** » standard while with higher RF speed
- ISO15693 is called « **long range** » standard

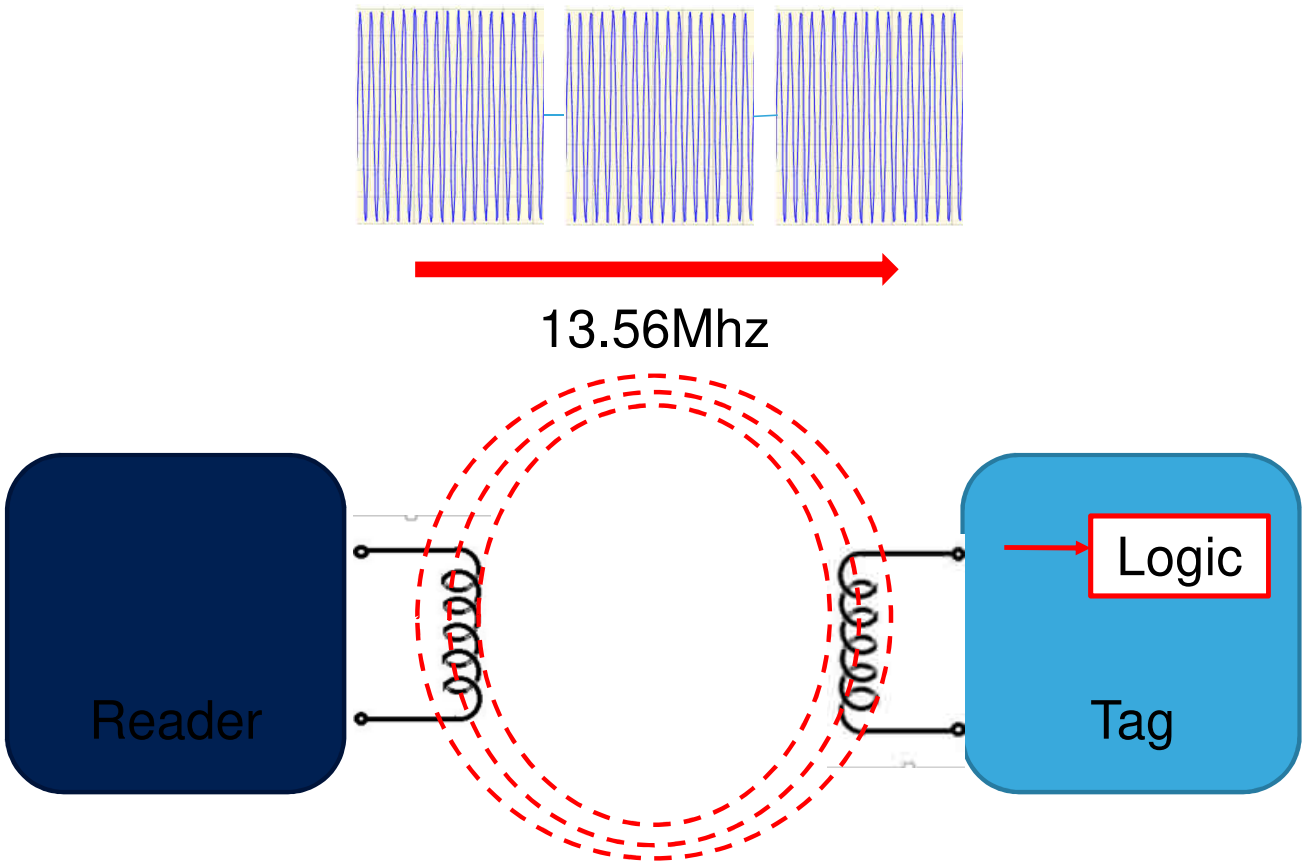
life.augmented

HF Communication – The Basis



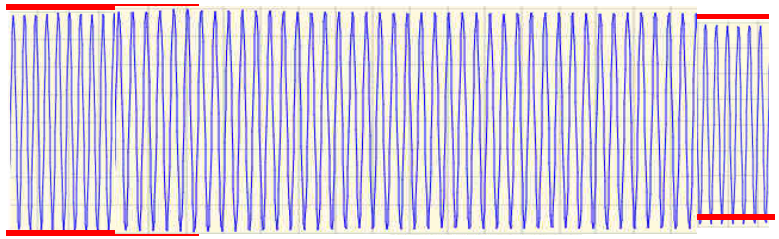
HF Communication

Reader to Tag



HF Communication

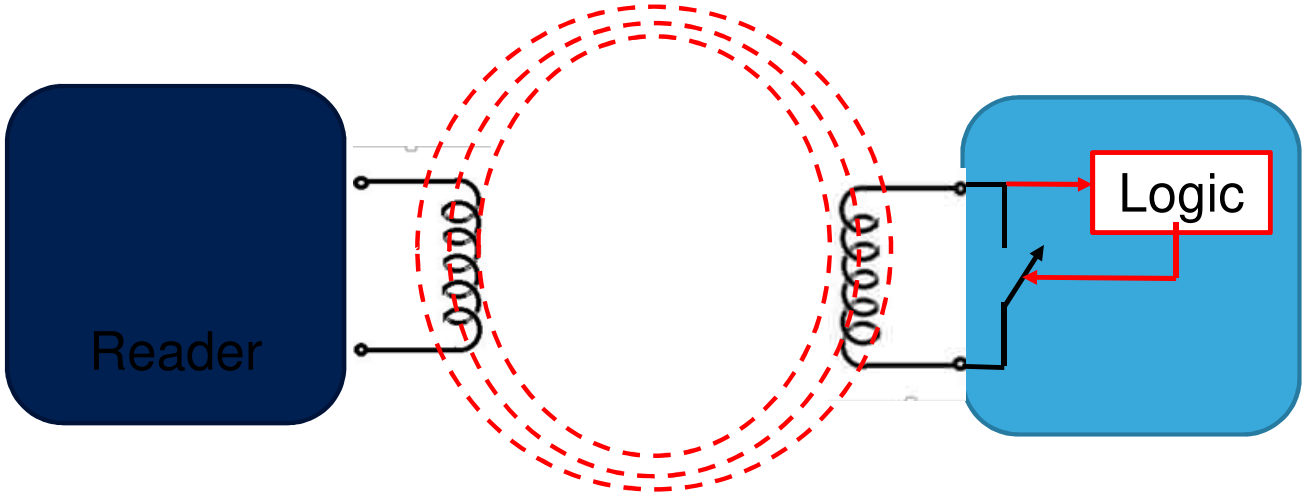
Tag to reader



Data rate based on standard

Tag

13.56Mhz



HF Communication

Communication	ISO14443A	ISO14443B	ISO15693	Felica
Reader to Tag	100% ASK Miller modified coding	10% ASK NRZ Coding	10% or 100% ASK Manchester Coding	8 – 30% ASK Manchester Coding
Tag to Reader	Subcarrier fc/16 OOK Manchester	Subcarrier fc/16 BPSK NRZ-L	Single or Double Subcarrier fc/32 or fc/28 Manchester	>12% ASK load modulation Manchester Coding

life.augmented

ST NFC Sensor TAG

life.augmented

ST **NFC Sensor TAG**

NFCSensorTAG is **a NFC enabled sensor node** that can sense temperature, humidity, pressure, vibration, motion and **transmit the data when triggered by an NFC reader**. It is a **reference platform** that can be scaled down/up based on requirement of final applications and use cases.

An **alternative way of connectivity** for applications that:

- Are extremely **low POWER** (also full passive) and **low COST**;
- Require small real estate (**reduced BOM**) and **fast implementation**;
- Do **NOT** require **Real-Time Remote** monitoring (Near Field Communication)

life.augmented

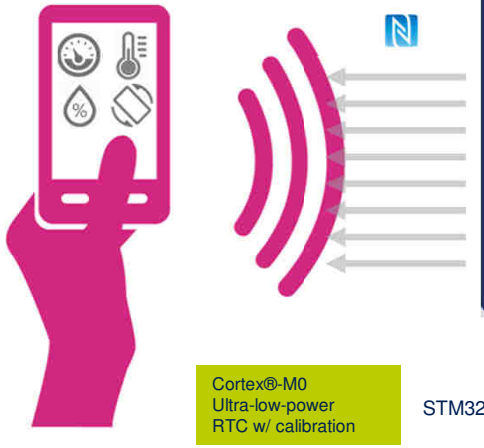
Leveraging ST Technology

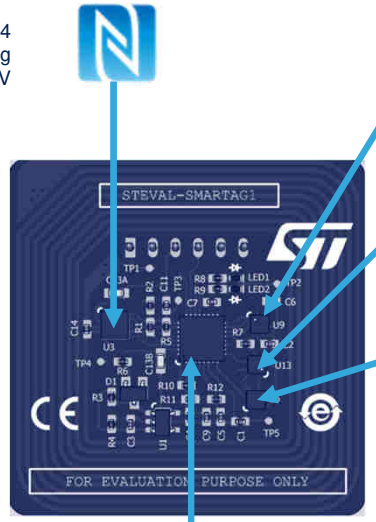
life.augmented

What's on the Board

Energy Harvesting
Low power I2C bus
Up to 64Kb NVM

ST25DV64
NFC Dynamic Tag
ISO15693 TypeV





LIS2DW12
Accelerometer

Ultra-low-power 3-axis "accelerometer
50 nA in power-down mode
±2g/±4g/±8g/±16g selectable full scales

LPS22HB
Barometer

Pressure 260-1260 hPa
water res/ dust immune. Accuracy 6cm

HTS221
Humidity+Temp.

Rel. Humidity +Temperature
Hum accuracy ± 3.5% rH
Temp accuracy ± 0.5 °C

Cortex®-M0
Ultra-low-power
RTC w/ calibration

STM32L0x1 MCU

Opt 1: No Battery (Full Passive)
Opt 2: Battery assisted (not Rechargeable)
Opt 3: Battery Assisted (Rechargeable)

Back view

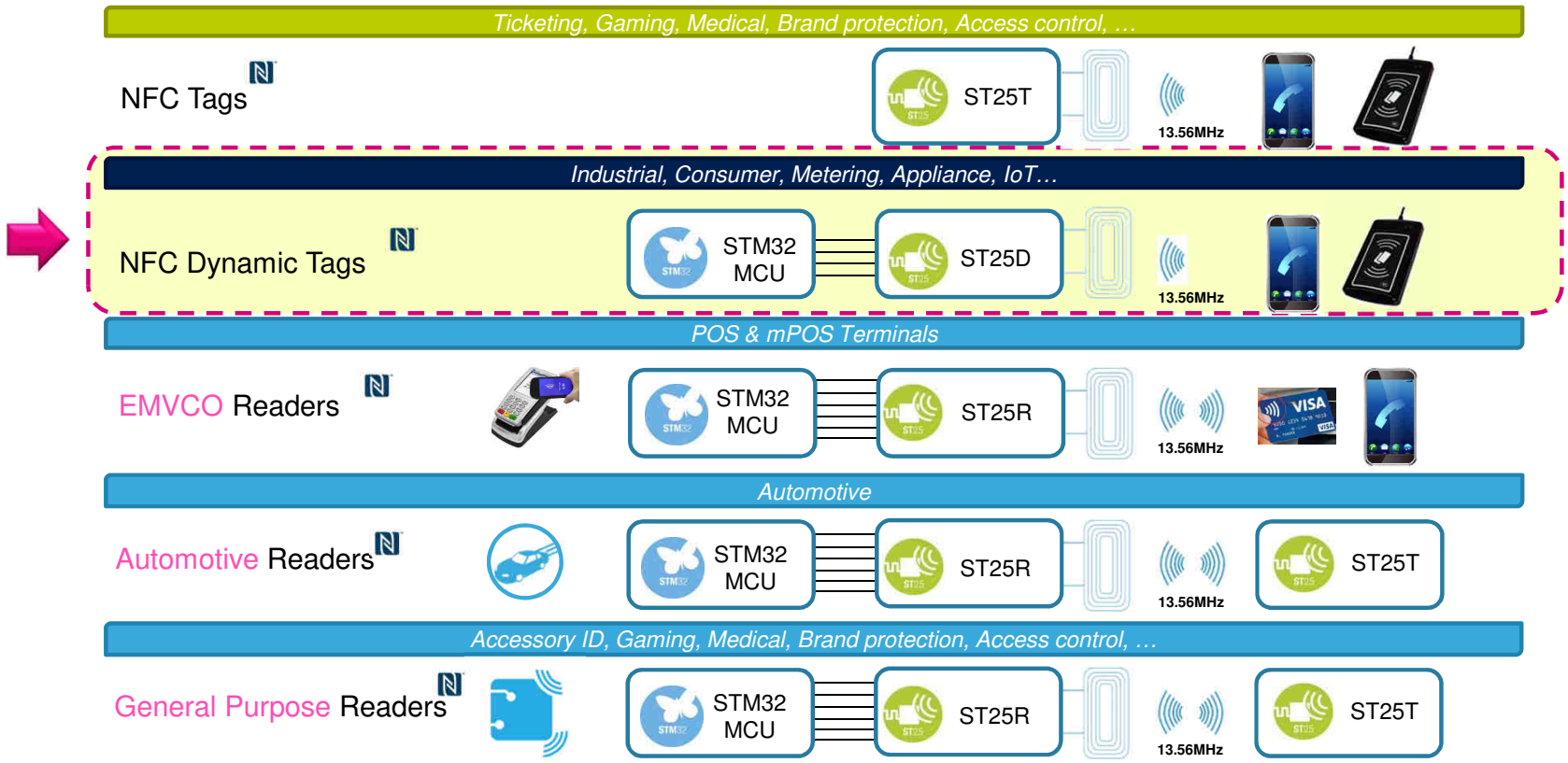
life.augmented

ST25 NFC Value Proposition

NFC / RFID Tags and Readers
Simply More Connected
Covering all NFC application needs and leveraging a rich ecosystem

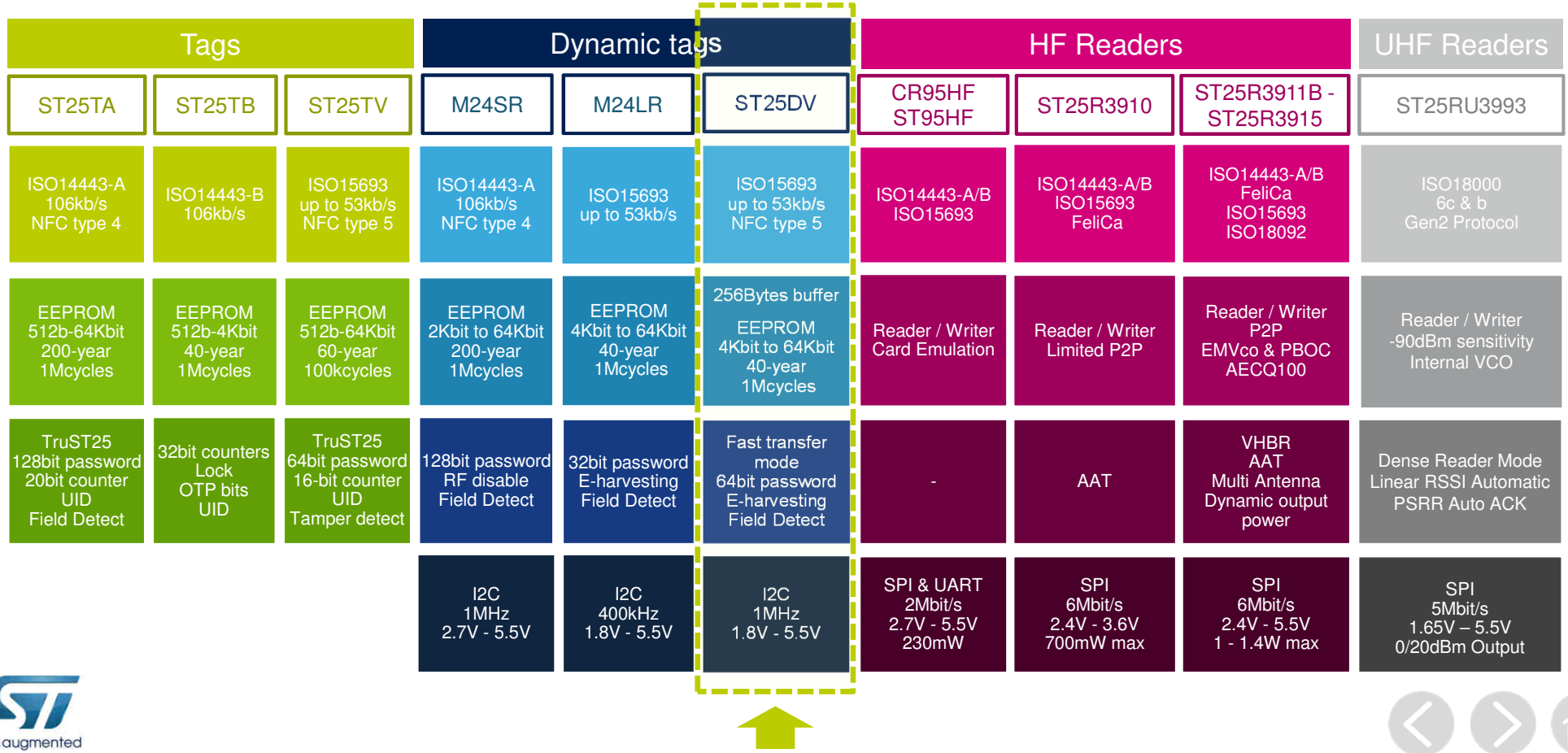
life.augmented

ST25 Product Lines



life.augmented

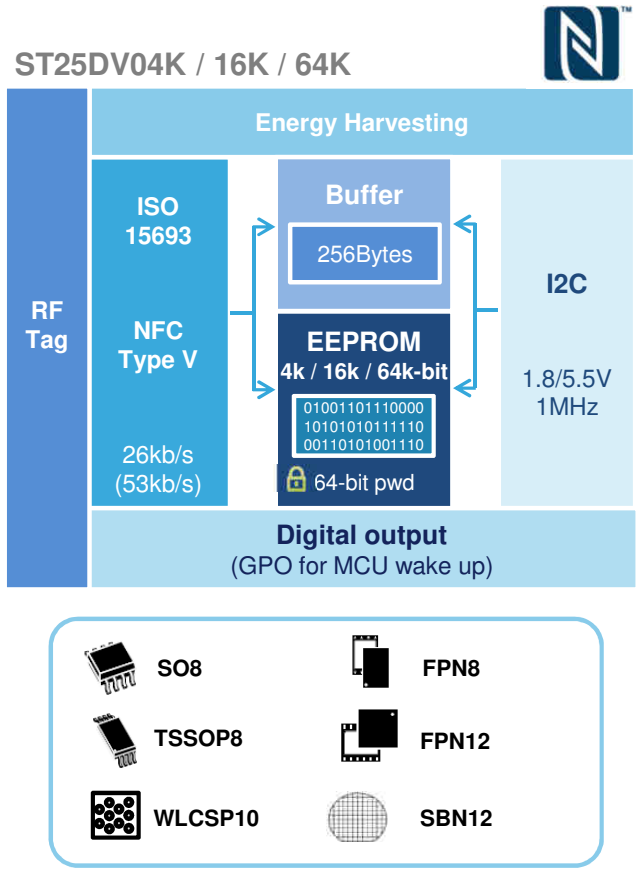
ST25 Product Portfolio



life.augmented

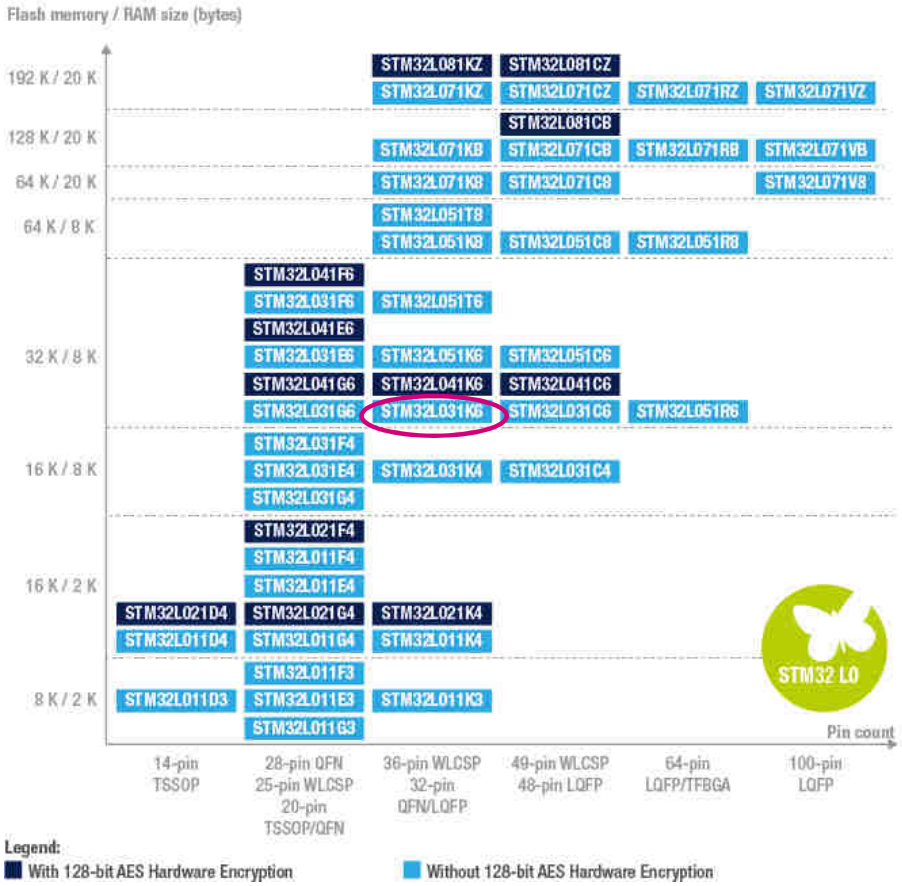
Focus on ST25DV Type5 Dynamic Tag

	ST25DV series
Contactless Interface	ISO15693 / NFC Forum Type 5
RF speed	up to 53kbps (26kbps std)
Single supply voltage	1.8V to 5.5V
Serial Interface	I2C @1MHz
Extra Features	GPO: 7 interrupts modes (OD or CMOS) Energy Harvesting Low Power Mode (<1uA stby)
Memory format & size	EEPROM data - 4 / 16 / 64-kbit
Data retention	40-year at +55°C
Fast Transfer Mode	256 Bytes memory buffer
Erase/Write cycles	1M cycles
Data protection	Password 64-bit
Temperature range	-40°C to +85°C
Package	SO8/TSSOP8/FPN8-12/WLCSP/SBN12



life.augmented

STM32L0 ULP MCU Series



life.augmented

STM32L031 ULP MCU

All the ingredients for your application

32bits CPU
8KB SRAM
1KB EEPROM

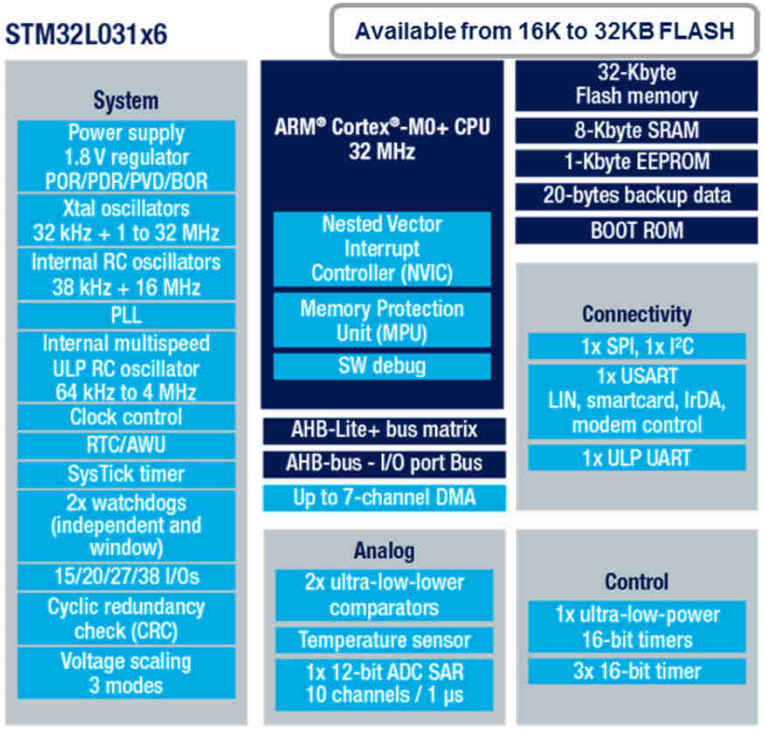
Control
Connectivity
Analog

Battery friendly

250nA standby

Goes everywhere

Down to 2 x 2,5 mm package



life.augmented

Smart Things

ST Leading Sensors

Ultra Low Power **Accelerometer & Gyroscope**	Accelerometer & Magnetometer **Compass**	High Accuracy **Pressure Sensor**	Combo **Humidity & Temperature**	Analog & Digital **Microphones**
LIS2DW12 / LSM6DSL	**LIS2MDL / LSM303AH**	**LPS22HB / LPS33HW**	**HTS221**	**MP23AB01DH / MP34DT05-A**
Low power & noise for UI, IoT, wearable http://www.st.com/content/st_com/en/products/mems-and-sensors/accelerometers/lis2dw12.html  http://www.st.com/en/mems-and-sensors/lsm6dsl.html	Accuracy, with pedometer (LSM303AH) http://www.st.com/content/st_com/en/products/mems-and-sensors/e-compasses/lis2mdl.html  http://www.st.com/content/st_com/en/products/mems-and-sensors/e-compasses/lsm303ah.html	Compact, low power, water resistant http://www.st.com/content/st_com/en/products/mems-and-sensors/pressure-sensors/lps22hb.html  http://www.st.com/content/st_com/en/products/mems-and-sensors/pressure-sensors/lps33hw.html	High accuracy Humidity and Temp  http://www.st.com/en/mems-and-sensors/hts221.html	Better sound quality http://www.st.com/en/audio-ics/mp23ab01dh.html  http://www.st.com/en/audio-ics/mp34dt05-a.html
ST Advantage: • Flexibility Power Consumption vs. Noise • Ecosystem (SW, libraries, ref design, Nucleo boards …)	ST Advantage: • Power Consumption • Thermal Stability • Precision	ST Advantage: • LPS22HB: Stability over time • LPS33HW: resistant to harsh environment (automotive gel, metal lid, ceramic substrate)	ST Advantage: • Low power consumption 2uA @ 1Hz • -40C to +120C operation • SPI/I2C host interface	ST Advantage: • High performance 135dB AOP 65dB SNR

life.augmented

ST Motion Sensors

LIS2DW12

3-axis Accelerometer

Package

2x2x0.7 mm

Applications examples

- Wearables
- Gesture/Position detect
- IoT
- Vibration/Motion monitor
- Anti-theft

Accelerometer

- ±2/±4/±8/±16 g selectable Full Scales
- Noise/accuracy level flex

12 to 14bit res

- Low current consumption

380nA LP Mode and 40nA Standby

Advantages

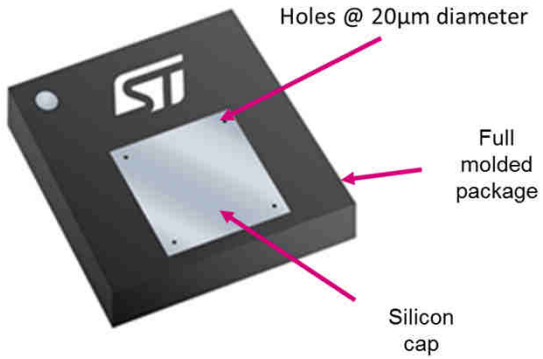
- High Accuracy for precise motion tracking
- Low Power Consumption for long battery life
- Embedding Digital Features, including Freefall, Wakeup, Orientation, Tap
- Easy integration, Software & Tools available

life.augmented

Pressure/Altimeter Sensor

LPS22HB

Pressure Barometer/Altimeter Sensor



Applications examples

- Weather Stations
- Indoor and Outdoor Navigation

Pressure

- Range 260-1260 hPa
- Relative accuracy of pressure measurement:
  - < 10 µbar
  - 6cm resolution

Advantages

- World's Smallest pressure sensor in production
- Full-mold Package with silicon cap and four micro holes guaranteeing sensor moisture and dust resistance
- Very low Power consumption 3µA
- Embedded Temperature compensation

life.augmented

Humidity & Temperature Sensor

HTS221

Relative Humidity and Temperature combo

HLGA-6L
(2.0 x 2.0 x 0.9mm)

Applications examples

- Environmental stations/monitoring
- HVA Conditioning
- Medical equipment
- Home appliances
- White goods

Humidity

- Range: 0%RH : 100%RH
- Accuracy: ±3.5%RH

Temperature

- Range: -40°C : +125°C
- Accuracy: ± 0.5°C, from 15°C to +40°C

Advantages

- High Accuracy
- Low power consumption 2 µA
- Extended operative supply voltage

life.augmented

Benefits

life.augmented

ST NFC Sensor TAG Benefits

- **Low Cost** compared to other Wireless Technologies. Reduced BOM (doesn't require RF matching circuitry), Lower cost RF IC, simplified Layout (low cost PCB), low cost certification.
- **Low Power**: Ultra Low Power Connectivity, Computing and Sensing Technologies allow long battery life and battery less operations.
- **Easy to implement**: easy certifications (Passive RF)
- **Flexible**: scalable to the right configuration for many applications
- Enhanced **added values features** can be implemented though NFC TAG (ID, Error Logs, Security, FW upgrade…)

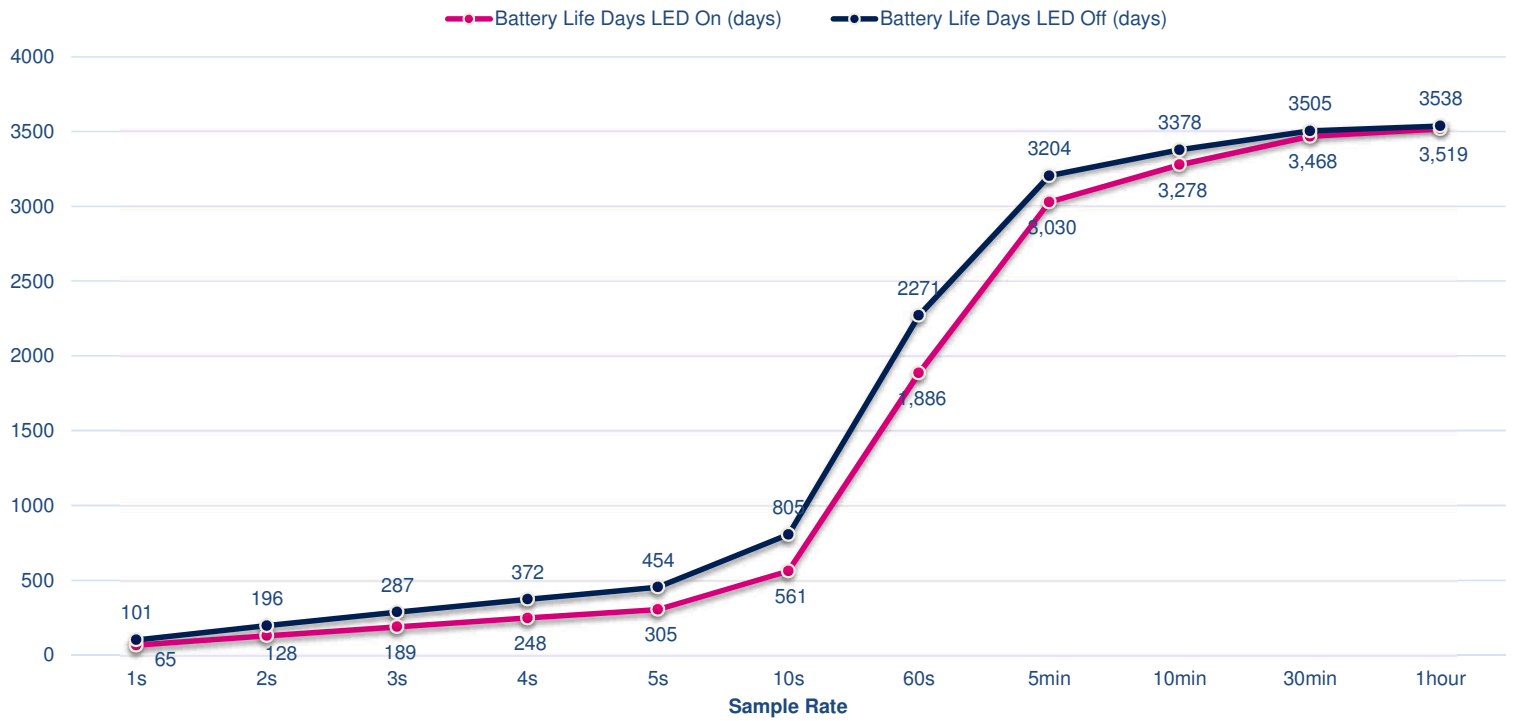
life.augmented

Battery Life

life.augmented

Battery Life Profile / Average

(Ref. CR2032 240mA/h batt.)



Measuring Condition:

- All sensors are selected (active)
- All sensors values are stored written) to the NFC DTag each time (at selected sample rate)

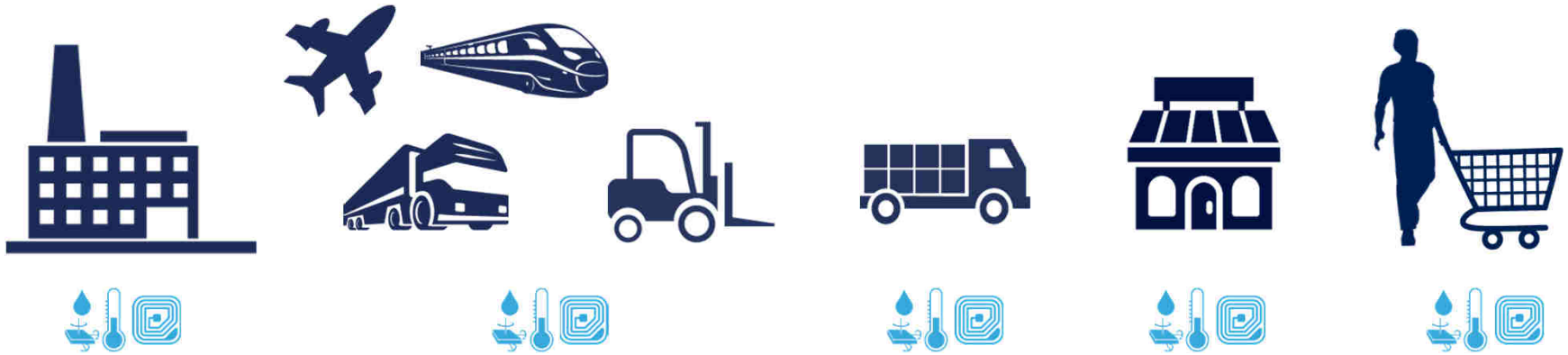
life.augmented

Addressing multiple Needs & Markets

life.augmented

Asset Tracking

Monitoring goods from Manufacturing to the end User

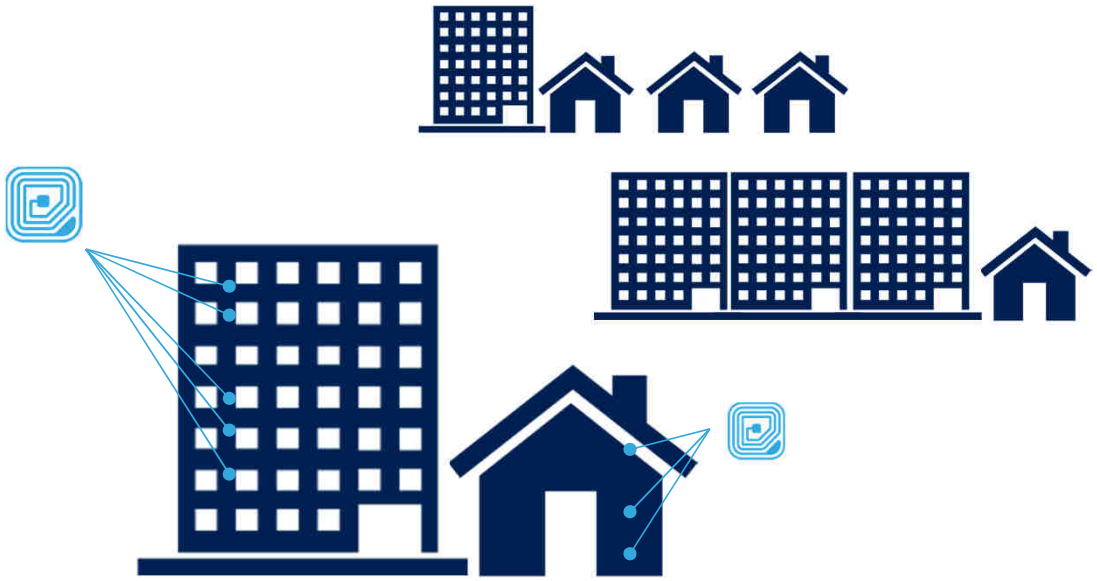


NFC → Secure (Near Field). Interoperability with Smartphone and/or dedicated Reader → Cloud
Temperature, Humidity & Pressure → Monitoring for Goods sensitive to environmental changes
Motion Sensors to detect **Vibrations**, Shocks, Freefall etc…which can damage/alter the goods
Low Power → Passive RF Tag (also battery less operation) / Long lasting battery life
Low Cost → Reduced BOM / Low cost connectivity.
Scalable → easy to add or remove sensors based on use case.

life.augmented

Smart Buildings/Cities

- Environmental and Motion Data Monitoring (Smart Badge / Weather Stations…)
- Building/Constructions Structure Monitoring (Vibration alert, Pipe Leaks…)



life.augmented

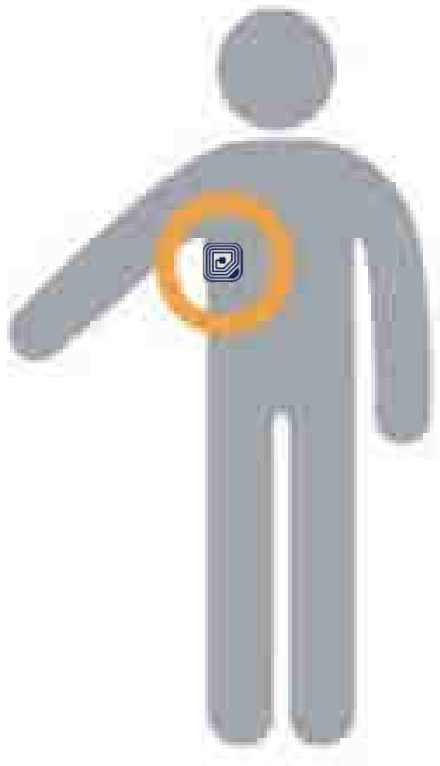
Smart Packaging

- Connect the “Unconnected”
- Temperature, Humidity, Vibration
- Level of Content and Use information
- Good ID / info / Commerce / Experience

life.augmented

Healthcare

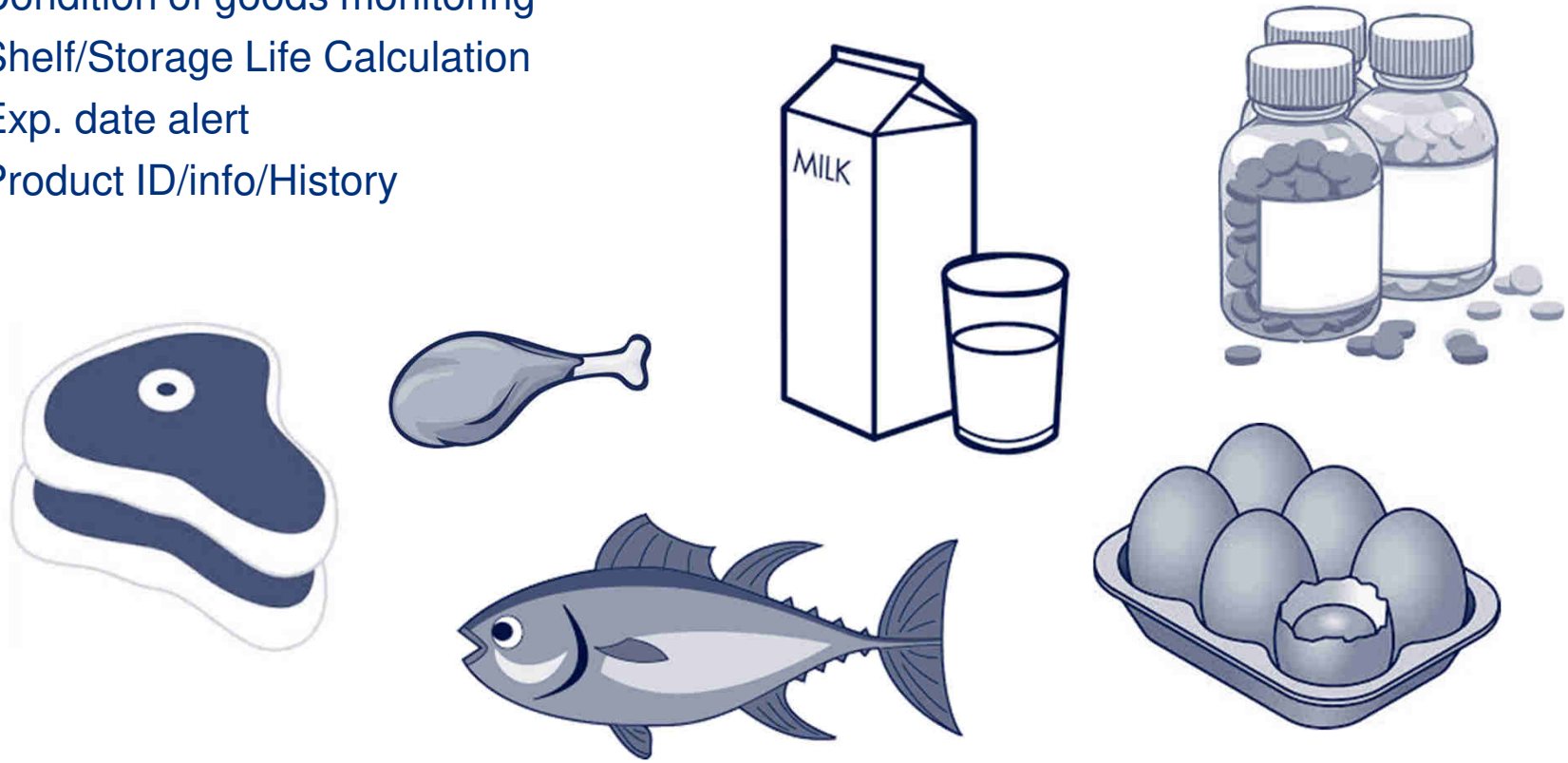
- Temperature Patches (also disposable)
- Patient activity tracker (also disposable)



life.augmented

Perishable Goods

- Condition of goods monitoring
- Shelf/Storage Life Calculation
- Exp. date alert
- Product ID/info/History



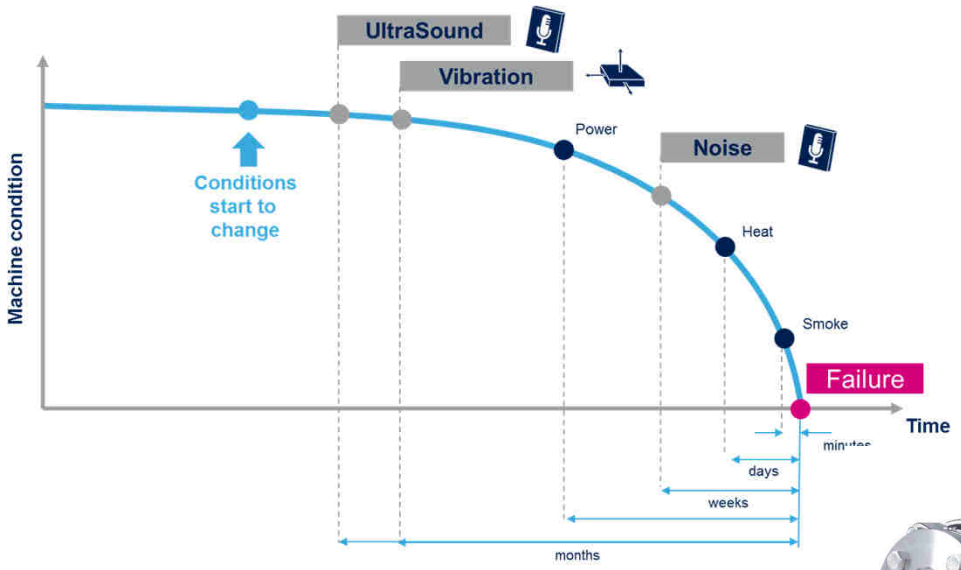
life.augmented

Smart Baggage

- Baggage Check-in/out
- Baggage ID and Geo-Location
- In-flight condition monitoring

life.augmented

Predictive Maintenance



Audio

Microphones to capture noise and for ultrasound emissions

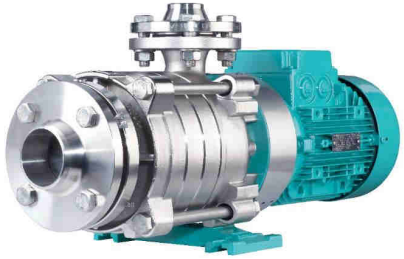
Motion
MEMS Sensors

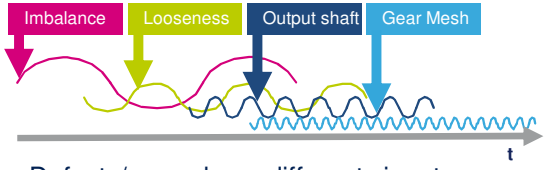
Environmental
Humidity and Pressure

Temp Sensors

Microphone
For Audio and Ultrasound

Vibration Measurements





Defects/wears have different signature

Condition monitoring enables to implement a predictive maintenance strategy by taking regular **vibration measurements** from sensors and comparing them to a baseline to detect health degradation

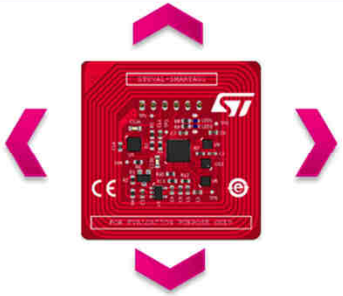
Use Case Example - Summary

Supply Chain & QC
- from manufacturing to end user.

Smart building/home/city:
- Environmental and Motion Data
- Building/Constructions Structure

Agriculture & Planting
Humidity, Temperature, light…

Retail/ Fitness
Smart Clothing / Patches: Activity, Temperature, Sweat, mktg…



Healthcare:
Medical patches & loggers: Patient temperature & activity monitoring

Animal tracking
Activity, Temperature, big data monitoring production, health…

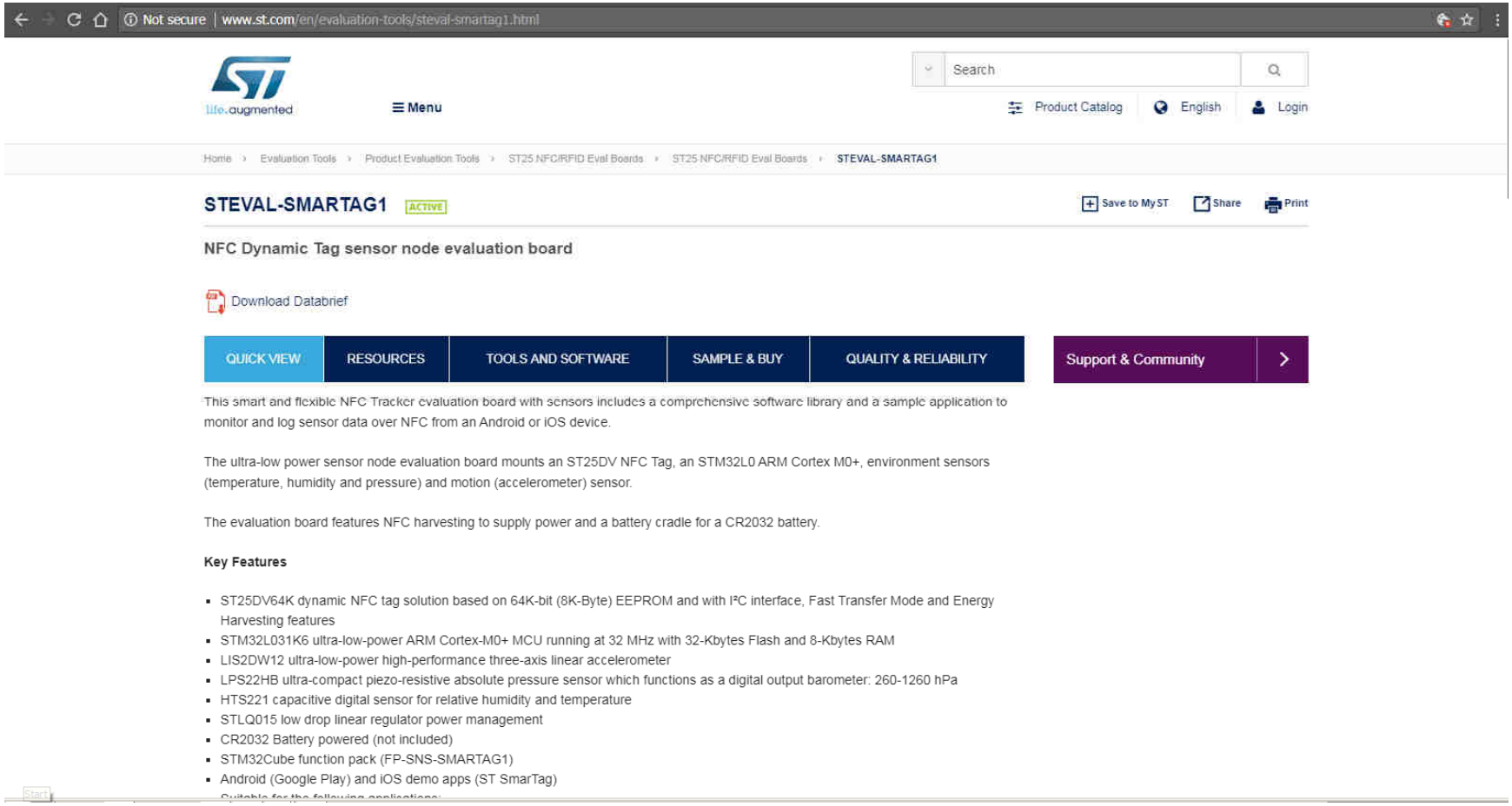
Perishable Goods:
Exp. date alert, Shelf/Storage Life Calculation/Condition of goods

Smart Packaging:
Temperature, Vibration Use, ID, . .

And much more…

life.augmented

www.st.com/nfcsensortag



Technical Documentation

Databrief

Technical Documentation

Product Specifications			
	Description	Version	Size
	DB3533: NFC Dynamic Tag sensor node evaluation board	2.0	626 KB

www.st.com/nfcsensortag

life.augmented

Hardware Resources

Gerbers

BOM

SCHEMATIC

Hardware Resources

Board Manufacturing Specifications			
	Description	Version	Size
	STEVAL-SMARTAG1 gerber files	1.0	70 KB

Bill of Materials			
	Description	Version	Size
	STEVAL-SMARTAG1 BOM	1.0	66 KB

Schematic Pack			
	Description	Version	Size
	STEVAL-SMARTAG1 schematic	1.0	90 KB

www.st.com/nfcsensortag

life.augmented

Miscellaneous

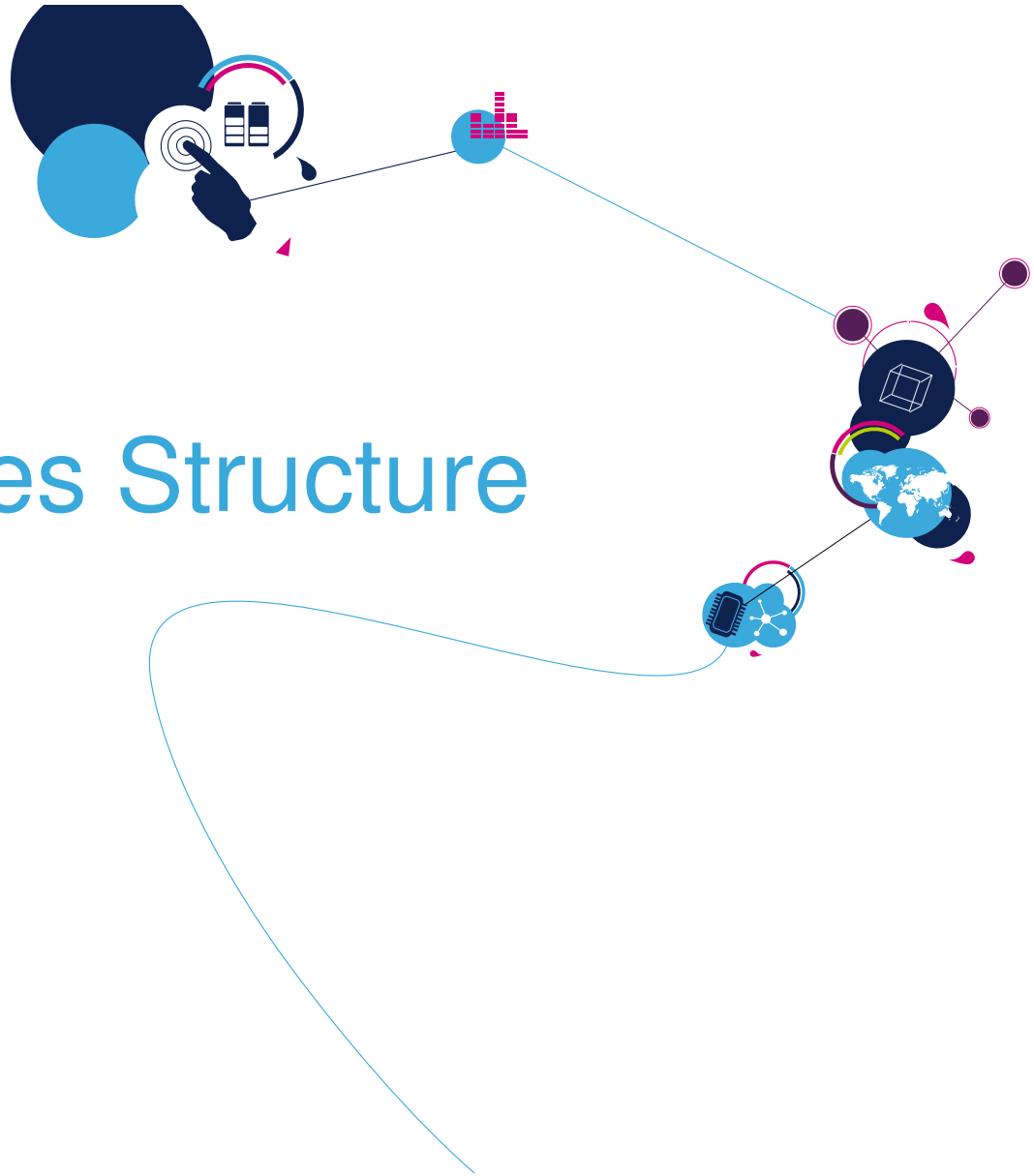
License agreements and certifications

Legal

License Agreement			
	Description	Version	Size
	Evaluation products license agreement	1.4	126 KB
	Open Reference Material License Agreement v5		42 KB

www.st.com/nfcsensortag

life.augmented



Firmware Packages Structure

life.augmented

ST Ecosystem: STM32 ODE

HARDWARE

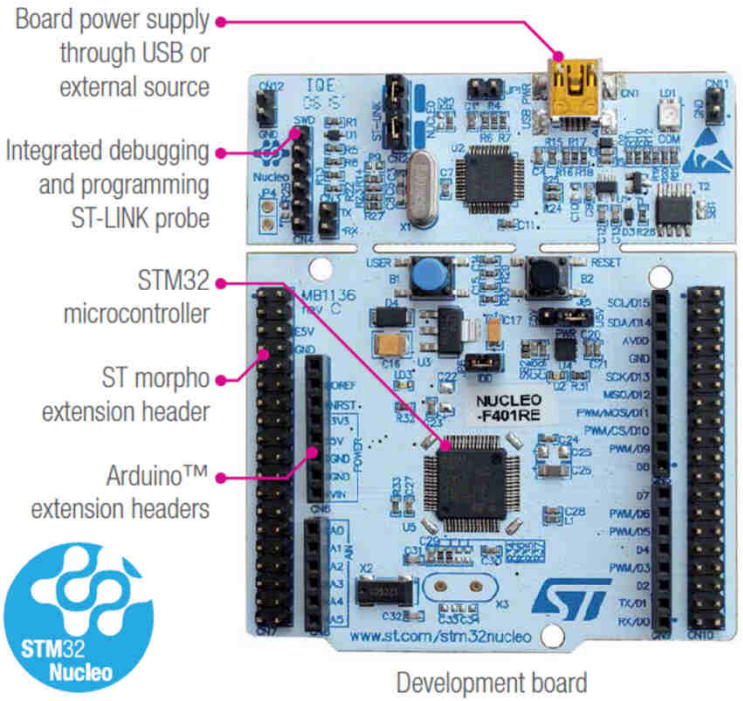
SOFTWARE

16
NUCLEO
L0 to L4
F0 to F7

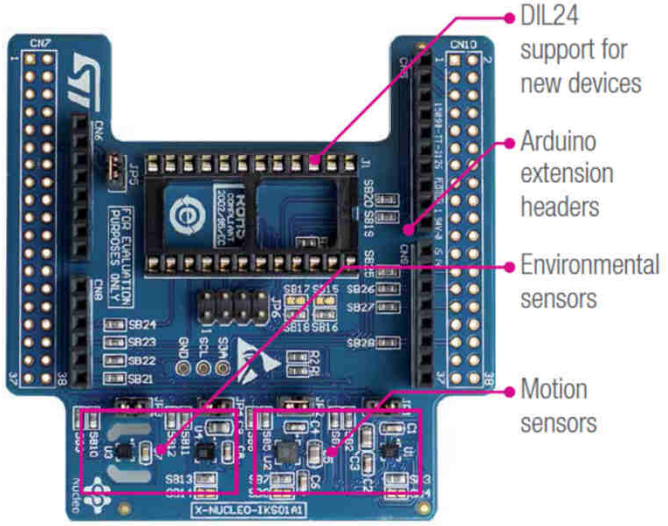
33
X-NUCLEO

life.augmented

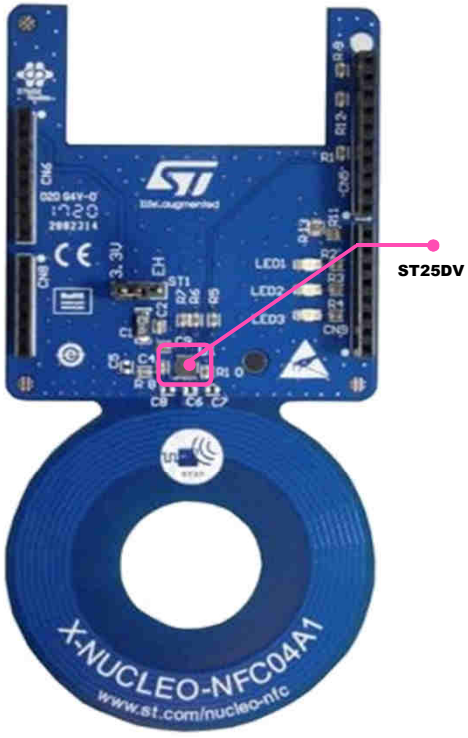
Nucleo / X-Nucleo



Development board



Expansion board (X-NUCLEO)



life.augmented

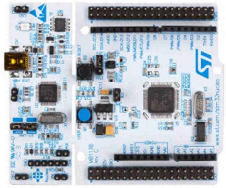
Nucleo / X-Nucleo & NFCSensorTAG

- Modular development system
- Rich set of **firmware packages**

- Form-factor development system
- Same set of firmware packages & **more**

X-NUCLEO-NFC04A1

X-NUCLEO-IKS01A2

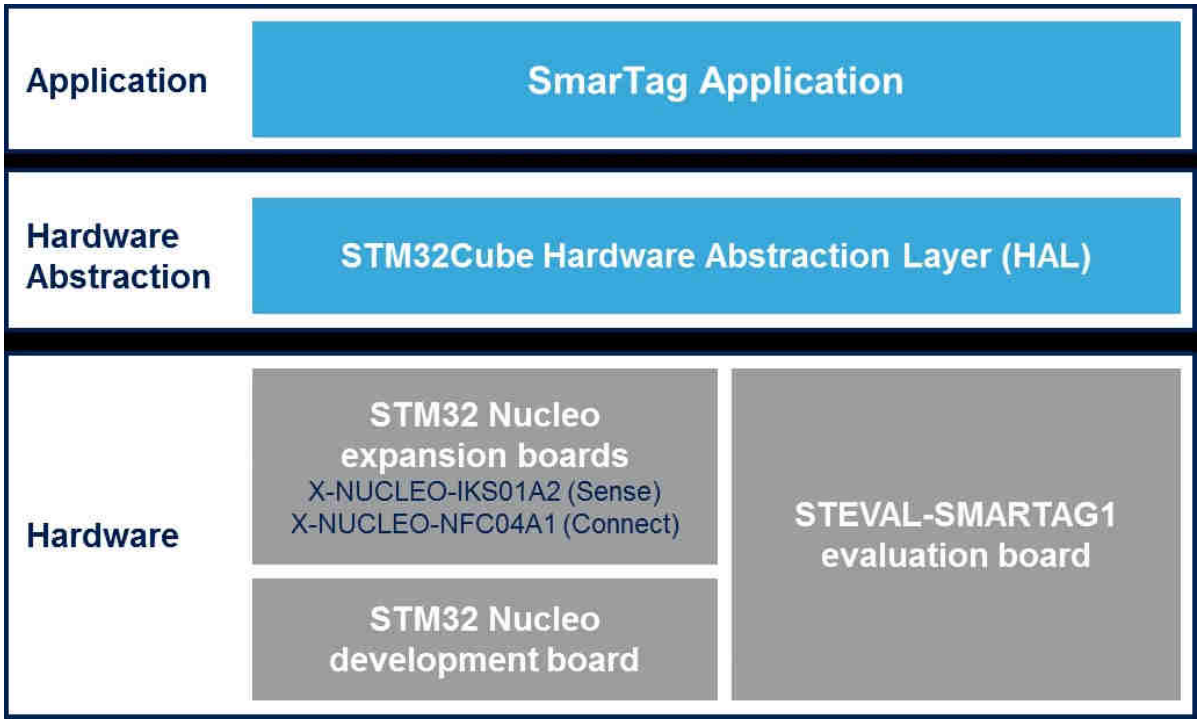


NUCLEO-L053RE

FP-SNS-SMARTAG1

life.augmented

Hardware / Software Block Diagram



Application
(e.g. ST25PC)

HAL and BSP

Boards

life.augmented

Function Packages

STM32ODE software package

Open Development Environment – src code

- **FP-SNS-SMARTAG1** Version 1.1.1

Technical Documentation

Product Specifications			
	Description	Version	Size
	DB3553: STM32Cube function pack for IoT node with Dynamic NFC Tag, environmental and motion sensors	1.0	293 KB

life.augmented

Function Pack User Manual

User Manuals			
	Description	Version	Size
	UM2389: Getting started with the STM32Cube function pack for IoT node with Dynamic NFC Tag, environmental and motion sensors	1.0	2 MB

Function Pack Presentations

Presentations & Training Material

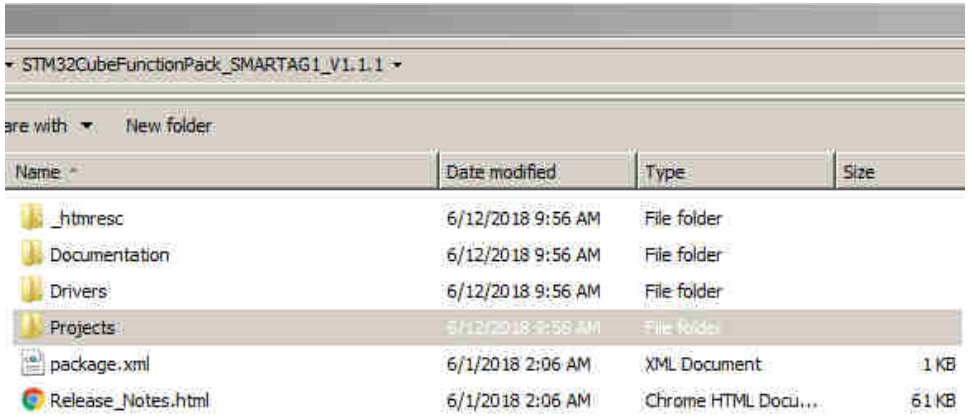
Presentations			
	Description	Version	Size
	FP-SNS-SMARTAG1 Quick Start Guide	1.1	2 MB
	STM32 and STM8 embedded software solutions	5.0	3 MB

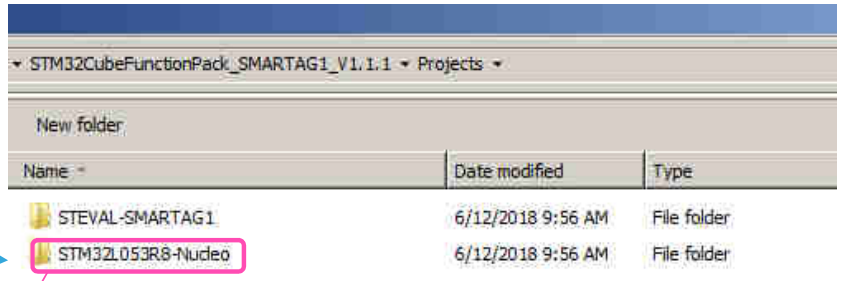
Miscellaneous

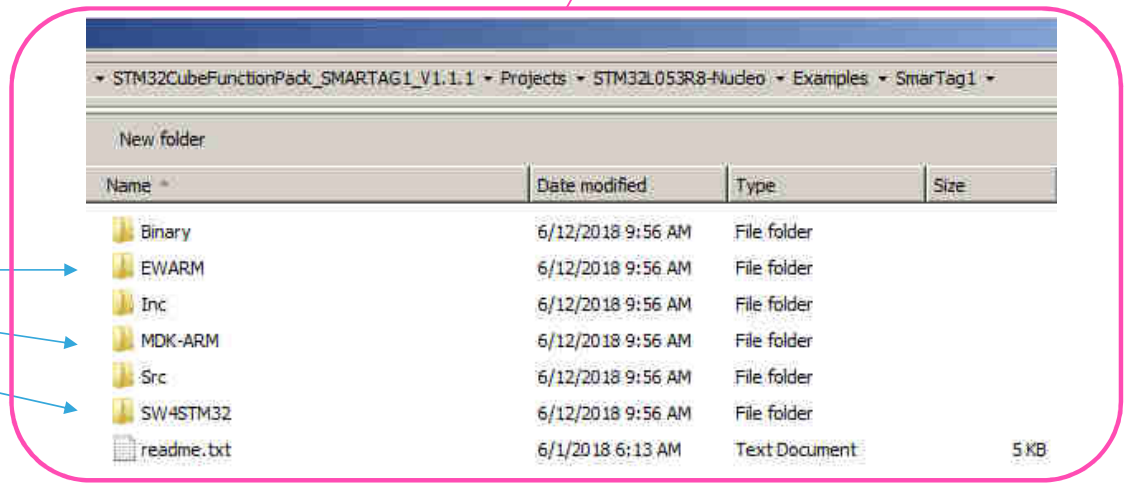
Legal

License Agreement			
	Description	Version	Size
	SLA0055: SOFTWARE LICENSE AGREEMENT ("Agreement")	4.13	122 KB

Function Pack and Nucleo Boards Stack







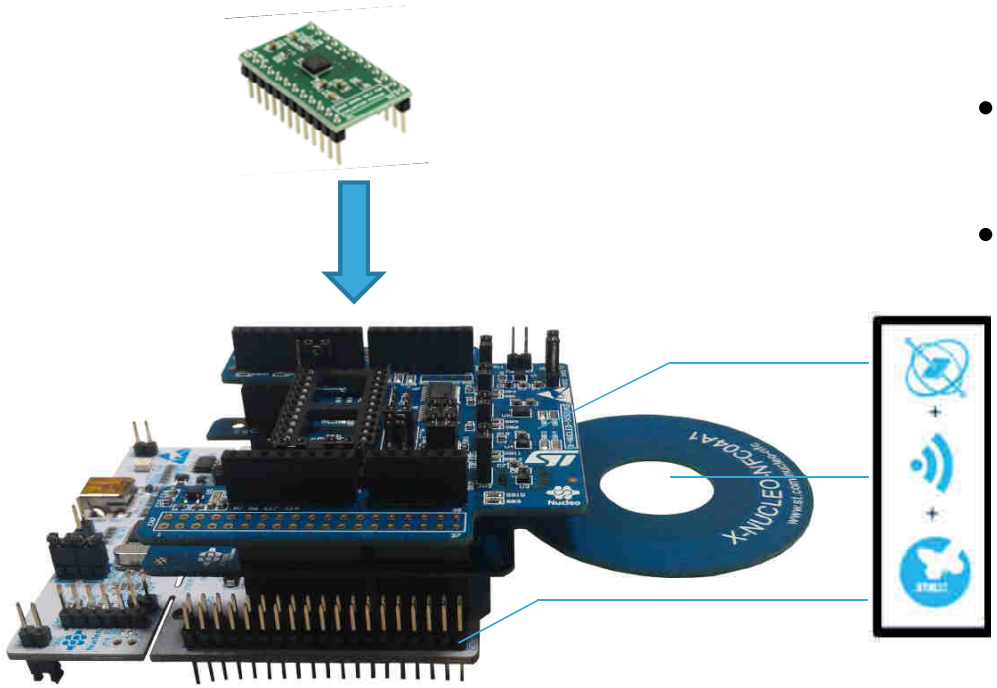
IAR project files

Keil project files

SystemWorkbench

life.augmented

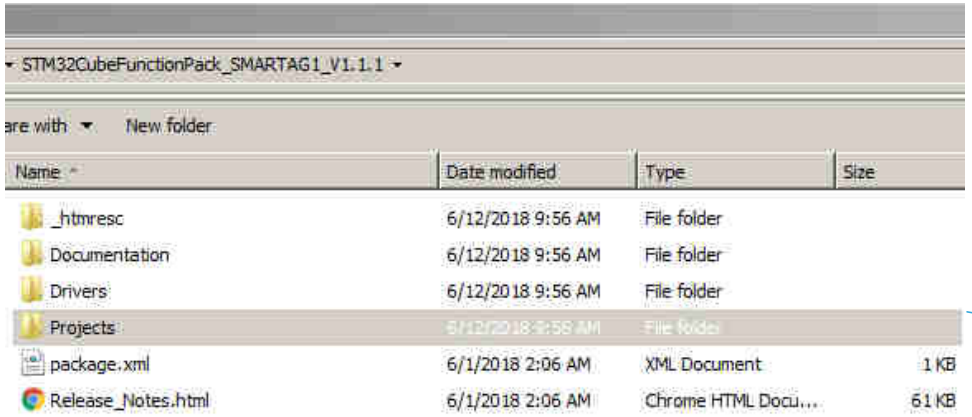
Benefits of using Nucleo Stack

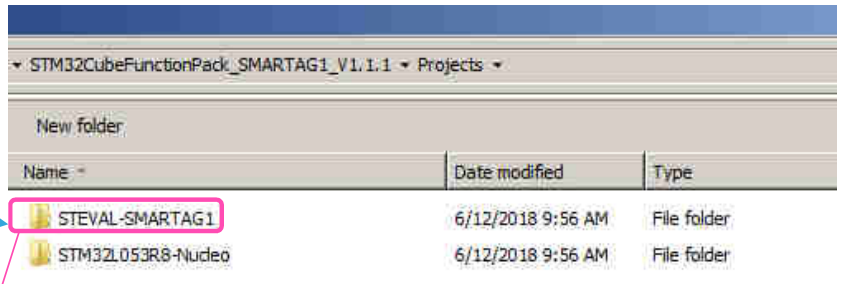


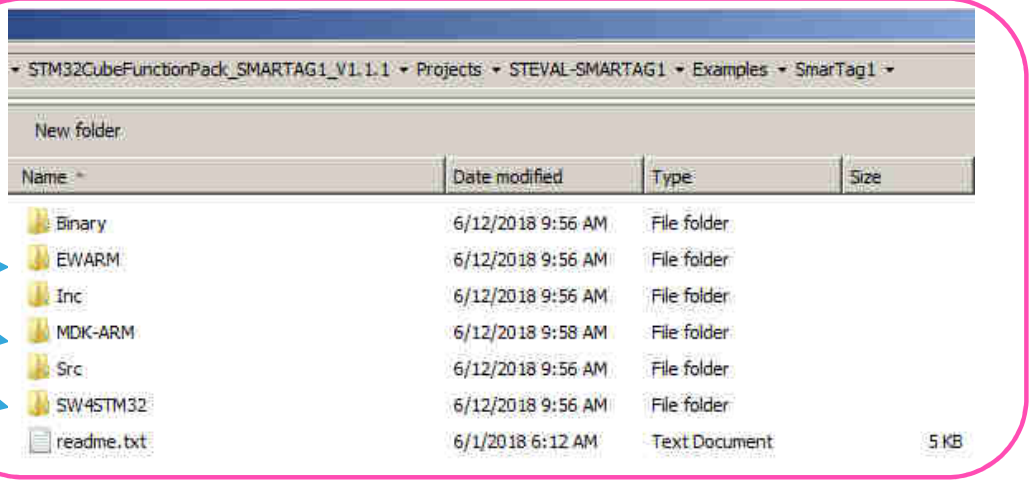
- Flexibility and Scalability with STM32 Variants (L0,L1,L4,F4,F3 and more)
- Note: Functional pack example only include L053
- DIL24 socket on IKS01A2 for additional MEMs adapter and other sensors

life.augmented

Function Pack and STEVAL-SMARTAG1







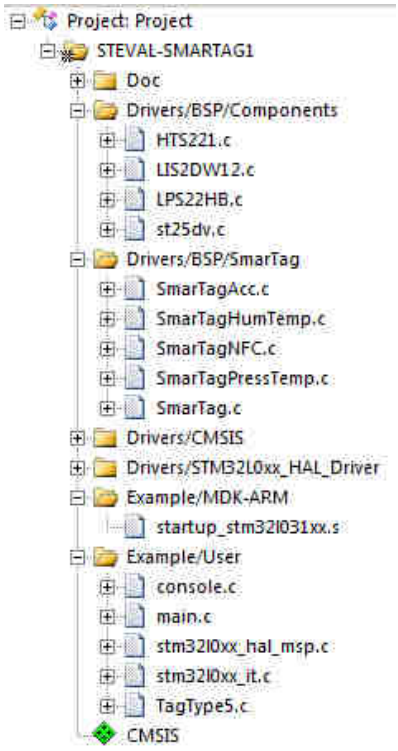
IAR project files

Keil project files

SystemWorkbench

life.augmented

Folder Structure (KEIL)



BSP = Board Support Package

- Components (typ. MEMS sensors)
- Boards (SensorTile, Nucleo, Nucleo-expansion)

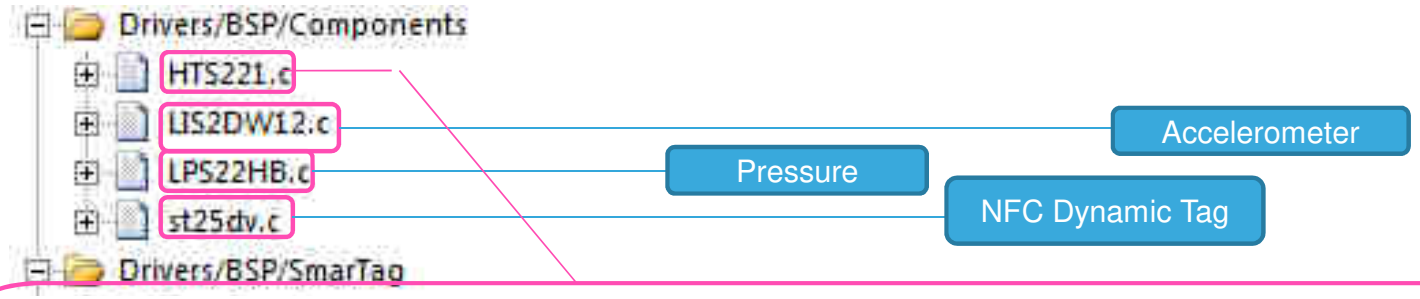
CMSIS = Cortex Microcontroller Software Interface Standard

- DSP library collection (fixed / float)

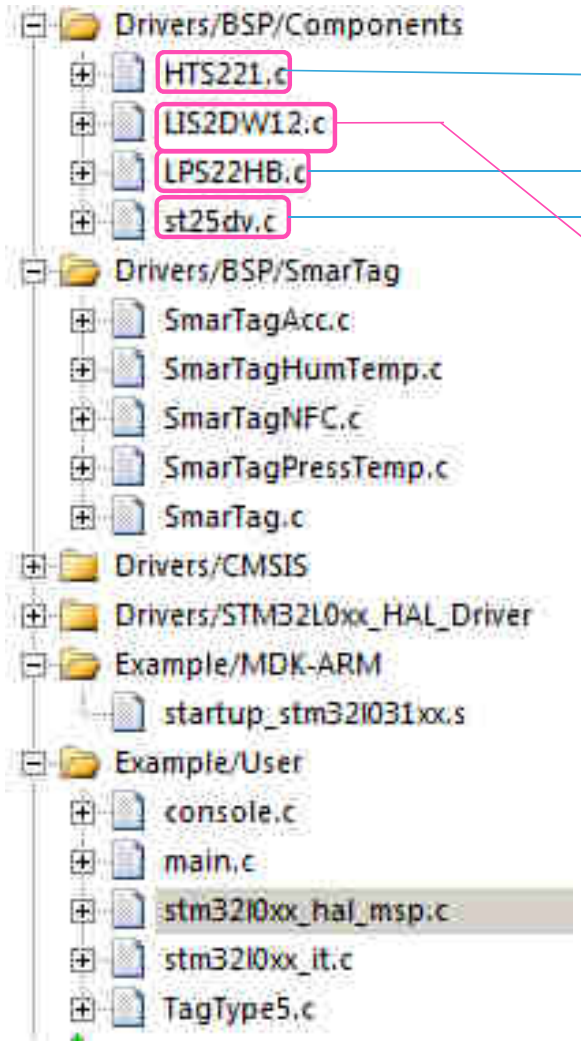
HAL = Hardware Abstraction Layer

- STM32 specific hardware drivers

Main.c is in Example\...\Src\




```
HTS221_Error_et HTS221_ReadReg( void *handle, uint8_t RegAddr, uint16_t NumByteToRead, uint8_t *Data );
HTS221_Error_et HTS221_WriteReg( void *handle, uint8_t RegAddr, uint16_t NumByteToWrite, uint8_t *Data );

HTS221_Error_et HTS221_Get_DeviceID(void *handle, uint8_t* deviceid);

HTS221_Error_et HTS221_Set_InitConfig(void *handle, HTS221_Init_st* pxInit);
HTS221_Error_et HTS221_Get_InitConfig(void *handle, HTS221_Init_st* pxInit);
HTS221_Error_et HTS221_DeInit(void *handle);
HTS221_Error_et HTS221_IsMeasurementCompleted(void *handle, HTS221_BitStatus_et* Is_Measurement_Completed);

HTS221_Error_et HTS221_Get_Measurement(void *handle, uint16_t* humidity, int16_t* temperature);
HTS221_Error_et HTS221_Get_RawMeasurement(void *handle, int16_t* humidity, int16_t* temperature);
HTS221_Error_et HTS221_Get_Humidity(void *handle, uint16_t* value);
HTS221_Error_et HTS221_Get_HumidityRaw(void *handle, int16_t* value);
HTS221_Error_et HTS221_Get_TemperatureRaw(void *handle, int16_t* value);
HTS221_Error_et HTS221_Get_Temperature(void *handle, int16_t* value);
HTS221_Error_et HTS221_Get_DataStatus(void *handle, HTS221_BitStatus_et* humidity, HTS221_BitStatus_et* temperature);
HTS221_Error_et HTS221_Activate(void *handle);
HTS221_Error_et HTS221_DeActivate(void *handle);

HTS221_Error_et HTS221_Set_AvgHT(void *handle, HTS221_Avgh_et avgh, HTS221_Avgt_et avgt);
HTS221_Error_et HTS221_Set_AvgH(void *handle, HTS221_Avgh_et avgh);
HTS221_Error_et HTS221_Set_AvgT(void *handle, HTS221_Avgt_et avgt);
HTS221_Error_et HTS221_Get_AvgHT(void *handle, HTS221_Avgh_et* avgh, HTS221_Avgt_et* avgt);
HTS221_Error_et HTS221_Set_BduMode(void *handle, HTS221_State_et status);
HTS221_Error_et HTS221_Get_BduMode(void *handle, HTS221_State_et* status);
HTS221_Error_et HTS221_Set_PowerDownMode(void *handle, HTS221_BitStatus_et status);
HTS221_Error_et HTS221_Get_PowerDownMode(void *handle, HTS221_BitStatus_et* status);
HTS221_Error_et HTS221_Set_Odr(void *handle, HTS221_Odr_et odr);
HTS221_Error_et HTS221_Get_Odr(void *handle, HTS221_Odr_et* odr);
HTS221_Error_et HTS221_MemoryBoot(void *handle);
HTS221_Error_et HTS221_Set_HeaterState(void *handle, HTS221_State_et status);
HTS221_Error_et HTS221_Get_HeaterState(void *handle, HTS221_State_et* status);
HTS221_Error_et HTS221_StartOneShotMeasurement(void *handle);
HTS221_Error_et HTS221_Set_IrqActiveLevel(void *handle, HTS221_DrdyLevel_et status);
HTS221_Error_et HTS221_Get_IrqActiveLevel(void *handle, HTS221_DrdyLevel_et* status);
HTS221_Error_et HTS221_Set_IrqOutputType(void *handle, HTS221_OutputType_et value);
HTS221_Error_et HTS221_Get_IrqOutputType(void *handle, HTS221_OutputType_et* value);
HTS221_Error_et HTS221_Set_IrqEnable(void *handle, HTS221_State_et status);
HTS221_Error_et HTS221_Get_IrqEnable(void *handle, HTS221_State_et* status);
```

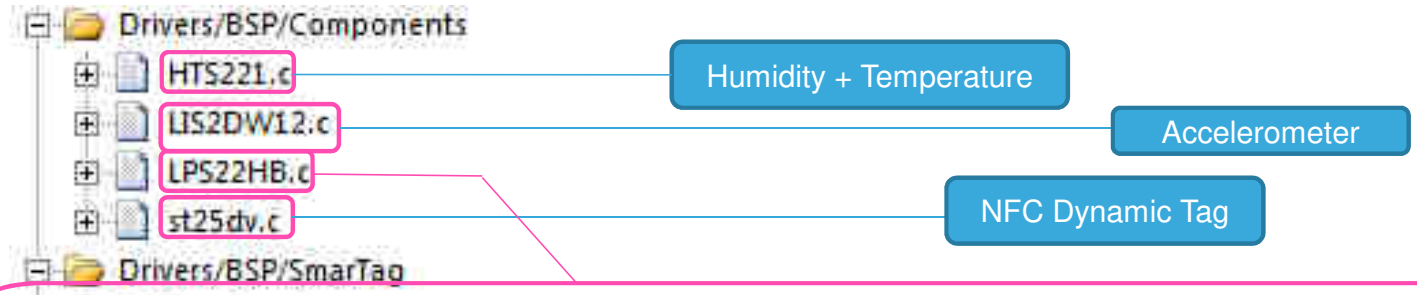


Humidity + Temperature

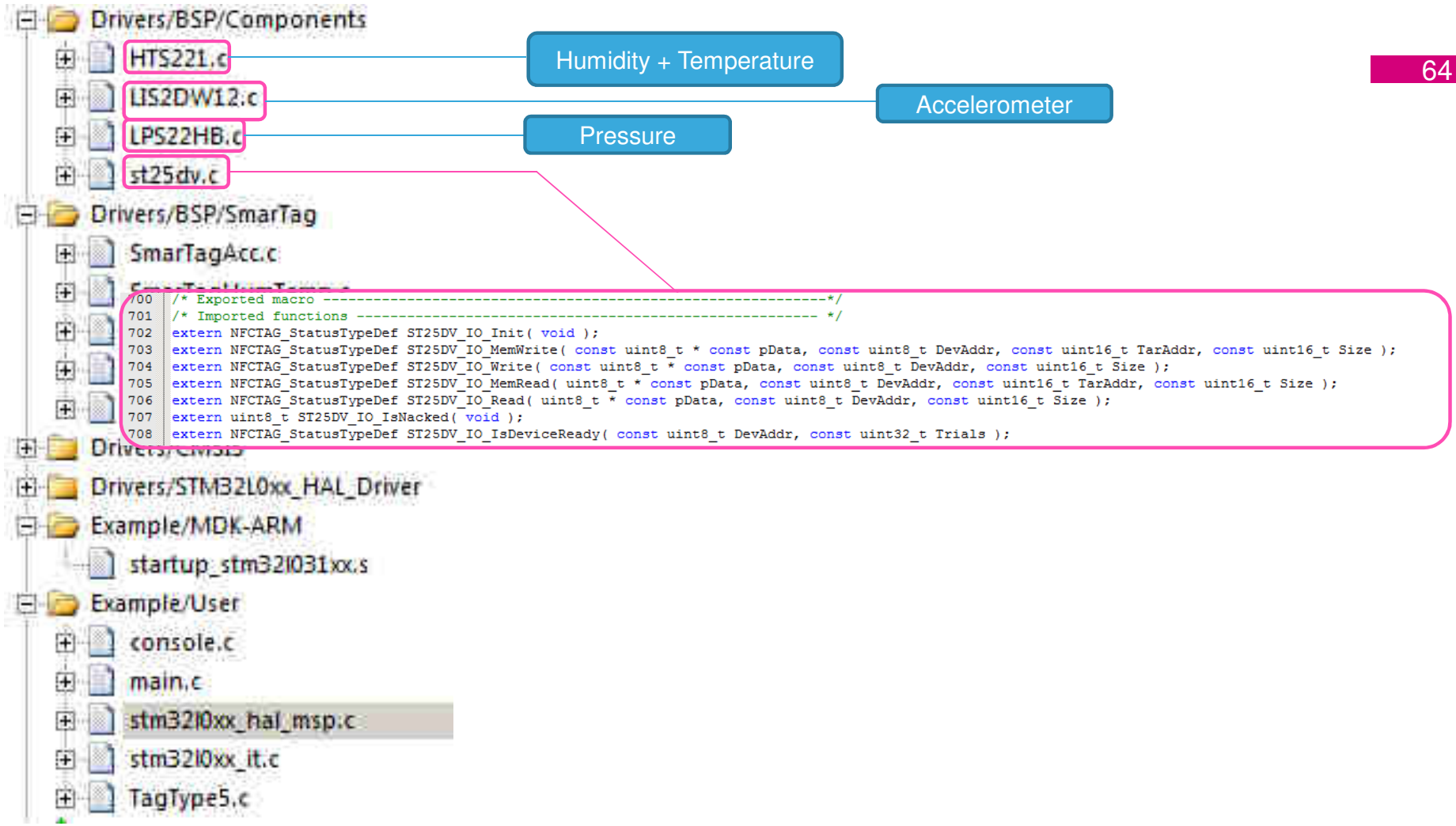
Pressure

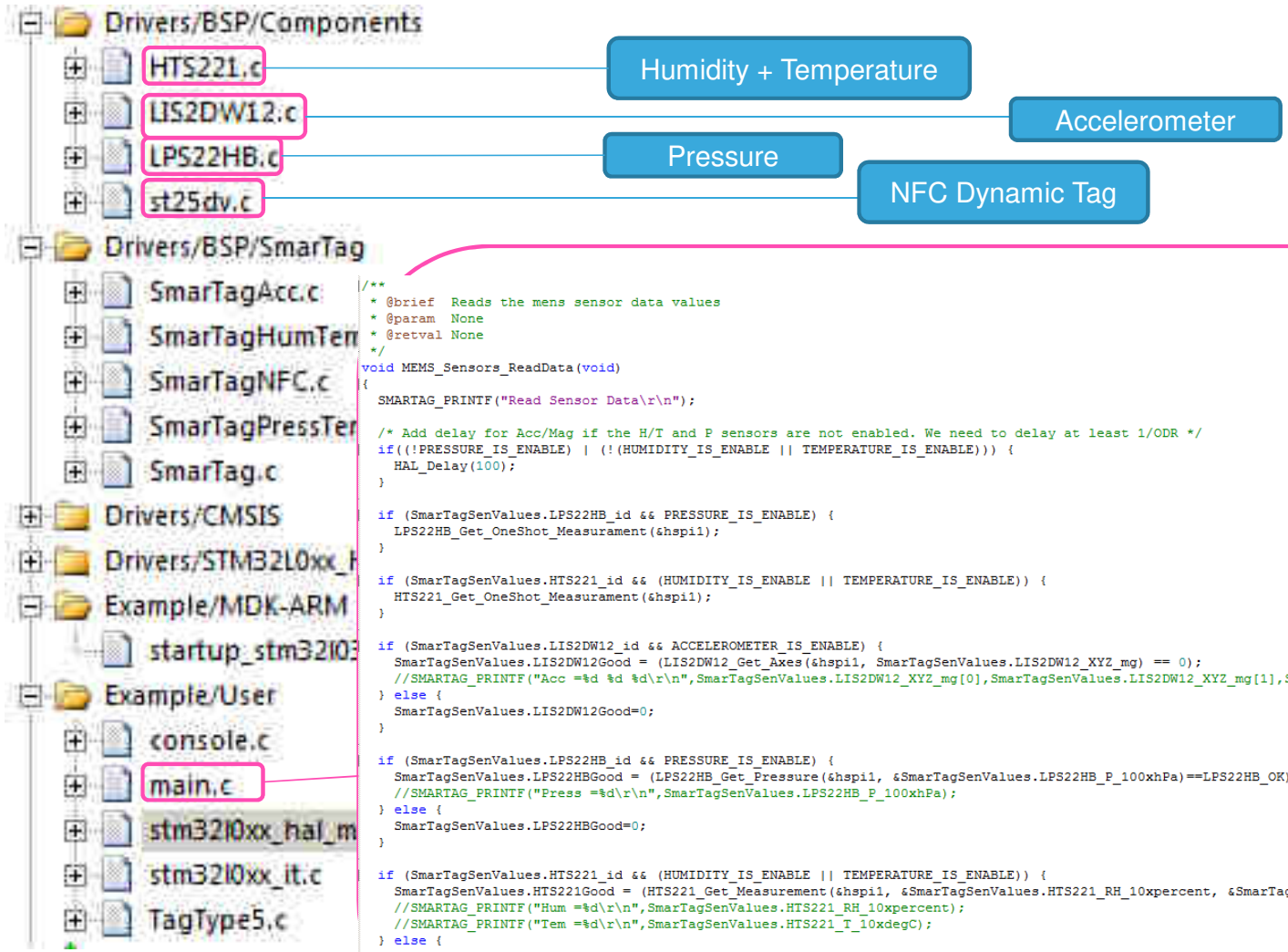
NFC Dynamic Tag

```
/*******************************************************************************
  * Function Name  : status_t LIS2DW12_ACC_Get_Acceleration(u8_t *buff)
  * Description    : Read Acceleration output register
  * Input          : pointer to [u8_t]
  * Output         : Acceleration buffer u8_t
  * Return         : Status [MEMS_ERROR, MEMS_SUCCESS]
*******************************************************************************/
status_t LIS2DW12_ACC_Get_Acceleration(void *handle, u8_t *buff)
{
  u8_t i, j, k;
  u8_t numberOfByteForDimension;

  numberOfByteForDimension=6/3;

  k=0;
  for (i=0; i<3;i++ )
  {
    for (j=0; j<numberOfByteForDimension;j++ )
    {
      if( !LIS2DW12_ACC_ReadReg(handle, LIS2DW12_ACC_OUT_X_L+k, &buff[k], 1))
        return MEMS_ERROR;
      k++;
    }
  }

  return MEMS_SUCCESS;
}
```




```
/**
* @brief  Get the LPS22HB raw presure value
* @param  The buffer to empty with the pressure raw value
* @retval Error Code [LPS22HB_ERROR, LPS22HB_OK]
*/
LPS22HB_Error_et LPS22HB_Get_RawPressure(void *handle, int32_t *raw_press);

/**
* @brief  Get the LPS22HB Pressure value in hPA.
* @param  The buffer to empty with the pressure value that must be divided by 100 to get the value in hPA
* @retval Error Code [LPS22HB_ERROR, LPS22HB_OK]
*/
LPS22HB_Error_et LPS22HB_Get_Pressure(void *handle, int32_t* Pout);

/**
* @brief  Read LPS22HB output register, and calculate the raw temperature.
* @param  The buffer to empty with the temperature raw value
* @retval Error Code [LPS22HB_ERROR, LPS22HB_OK]
*/
LPS22HB_Error_et LPS22HB_Get_RawTemperature(void *handle, int16_t *raw_data);

/**
* @brief  Read the Temperature value in °C.
* @param  The buffer to empty with the temperature value that must be divided by 10 to get the value in ['C]
* @retval Error Code [LPS22HB_ERROR, LPS22HB_OK]
*/
LPS22HB_Error_et LPS22HB_Get_Temperature(void *handle, int16_t* Tout);

/**
* @brief  Set One Shot bit to start a new conversion (ODR mode has to be 000)
* @param  void
* @retval Error Code [LPS22HB_ERROR, LPS22HB_OK]
*/
LPS22HB_Error_et LPS22HB_StartOneShotMeasurement(void *handle);
```


Drivers/BSP/Components
HTS221.c
LIS2DW12.c
LPS22HB.c
st25dv.c
Humidity + Temperature
Accelerometer
Pressure
Drivers/BSP/SmarTag
SmarTagAcc.c
Drivers/STM32L0xx_HAL_Driver
Example/MDK-ARM
startup_stm32l031xx.s
Example/User
console.c
main.c
stm32l0xx_hal_msp.c
stm32l0xx_it.c
TagType5.c
700 /* Exported macro --*/
701 /* Imported functions --- */
702 extern NFCTAG_StatusTypeDef ST25DV_IO_Init(void);
703 extern NFCTAG_StatusTypeDef ST25DV_IO_MemWrite(const uint8_t * const pData, const uint8_t DevAddr, const uint16_t TarAddr, const uint16_t Size);
704 extern NFCTAG_StatusTypeDef ST25DV_IO_Write(const uint8_t * const pData, const uint8_t DevAddr, const uint16_t Size);
705 extern NFCTAG_StatusTypeDef ST25DV_IO_MemRead(uint8_t * const pData, const uint8_t DevAddr, const uint16_t TarAddr, const uint16_t Size);
706 extern NFCTAG_StatusTypeDef ST25DV_IO_Read(uint8_t * const pData, const uint8_t DevAddr, const uint16_t Size);
707 extern uint8_t ST25DV_IO_IsNacked(void);
708 extern NFCTAG_StatusTypeDef ST25DV_IO_IsDeviceReady(const uint8_t DevAddr, const uint32_t Trials);




```
/**
 * @brief  Reads the mens sensor data values
 * @param  None
 * @retval None
 */
void MEMS_Sensors_ReadData(void)
{
  SMARTAG_PRINTF("Read Sensor Data\r\n");

  /* Add delay for Acc/Mag if the H/T and P sensors are not enabled. We need to delay at least 1/ODR */
  if((!PRESSURE_IS_ENABLE) | (!(HUMIDITY_IS_ENABLE || TEMPERATURE_IS_ENABLE))) {
    HAL_Delay(100);
  }

  if (SmarTagSenValues.LPS22HB_id && PRESSURE_IS_ENABLE) {
    LPS22HB_Get_OneShot_Measurament(&hspi1);
  }

  if (SmarTagSenValues.HTS221_id && (HUMIDITY_IS_ENABLE || TEMPERATURE_IS_ENABLE)) {
    HTS221_Get_OneShot_Measurament(&hspi1);
  }

  if (SmarTagSenValues.LIS2DW12_id && ACCELEROMETER_IS_ENABLE) {
    SmarTagSenValues.LIS2DW12Good = (LIS2DW12_Get_Axes(&hspi1, SmarTagSenValues.LIS2DW12_XYZ_mg) == 0);
    //SMARTAG_PRINTF("Acc =%d %d %d\r\n",SmarTagSenValues.LIS2DW12_XYZ_mg[0],SmarTagSenValues.LIS2DW12_XYZ_mg[1],SmarTagSenValues.LIS2DW12_XYZ_mg[2]);
  } else {
    SmarTagSenValues.LIS2DW12Good=0;
  }

  if (SmarTagSenValues.LPS22HB_id && PRESSURE_IS_ENABLE) {
    SmarTagSenValues.LPS22HBGood = (LPS22HB_Get_Pressure(&hspi1, &SmarTagSenValues.LPS22HB_P_100xhPa)==LPS22HB_OK);
    //SMARTAG_PRINTF("Press =%d\r\n",SmarTagSenValues.LPS22HB_P_100xhPa);
  } else {
    SmarTagSenValues.LPS22HBGood=0;
  }

  if (SmarTagSenValues.HTS221_id && (HUMIDITY_IS_ENABLE || TEMPERATURE_IS_ENABLE)) {
    SmarTagSenValues.HTS221Good = (HTS221_Get_Measurement(&hspi1, &SmarTagSenValues.HTS221_RH_10xpercent, &SmarTagSenValues.HTS221_T_10xdegC)==HTS221_OK);
    //SMARTAG_PRINTF("Hum =%d\r\n",SmarTagSenValues.HTS221_RH_10xpercent);
    //SMARTAG_PRINTF("Tem =%d\r\n",SmarTagSenValues.HTS221_T_10xdegC);
  } else {
    SmarTagSenValues.HTS221Good=0;
  }
}
```

life.augmented

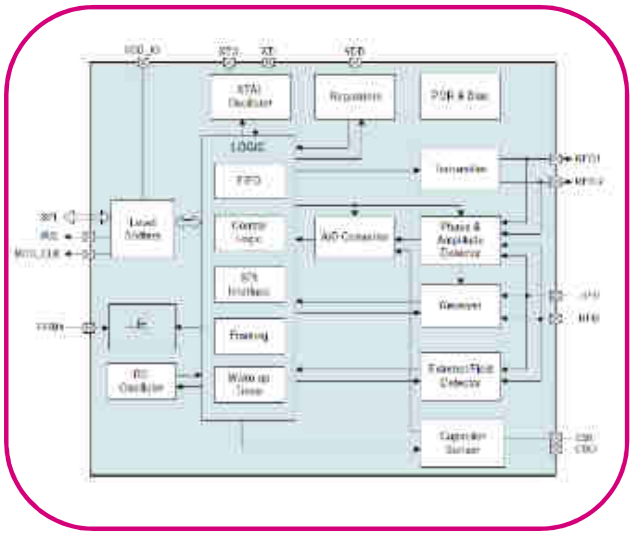
NFC Reader Discovery Board

life.augmented

ST25R3911B NFC / RFID Reader

1.4W High Power reader solution

ST25R3911B



QFN32

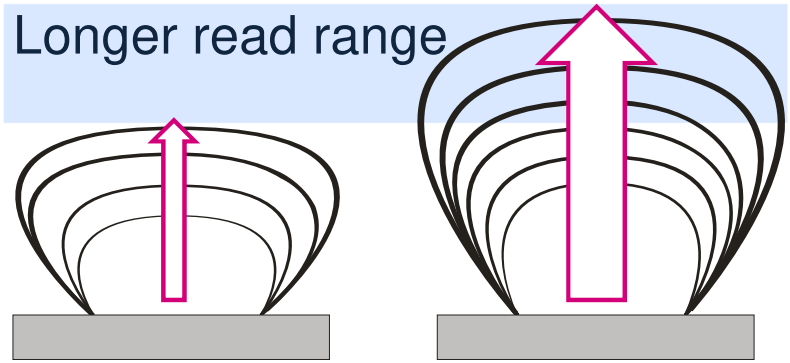
Wafer

- Use cases
  - Ideal for Payment Applications
  - Access Control, Gaming, eGovernment
- Key features
  - NFC forum compatible (no passive target)
  - **1.4W** output power at 5V
  - Passes **EMVco** & **PBOC** certification without external power amplifier
  - Automatic Antenna Tuning
  - **VHBR** support up to 6.8Mb/s
  - -40°C to **125°C** temperature range
- Key benefits
  - Low power operation & standby
  - Works in challenging environment
  - Enhanced fast transfer rate for Passport
  - Easy-to-use evaluation / development kits
  - Reference designs, application notes

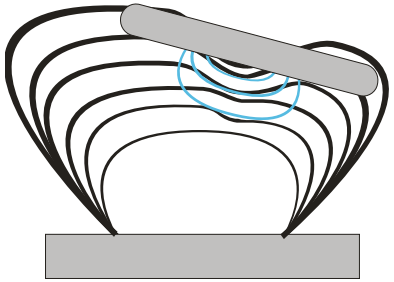
life.augmented

Output power & sensitivity

- Higher output power
  - The ST25R3911B includes low impedance drivers capable of generating >1 W of output power
  - EMVco certification easily possible without external boosters
  - “Slave” devices like interface tags are able to harvest far more energy for batteryless devices
  - The ST25R3911B is able to operate in metal encapsulation like doorlocks
- Higher sensitivity
  - The ST25R3911B has a 10x higher sensitivity than any competitor.



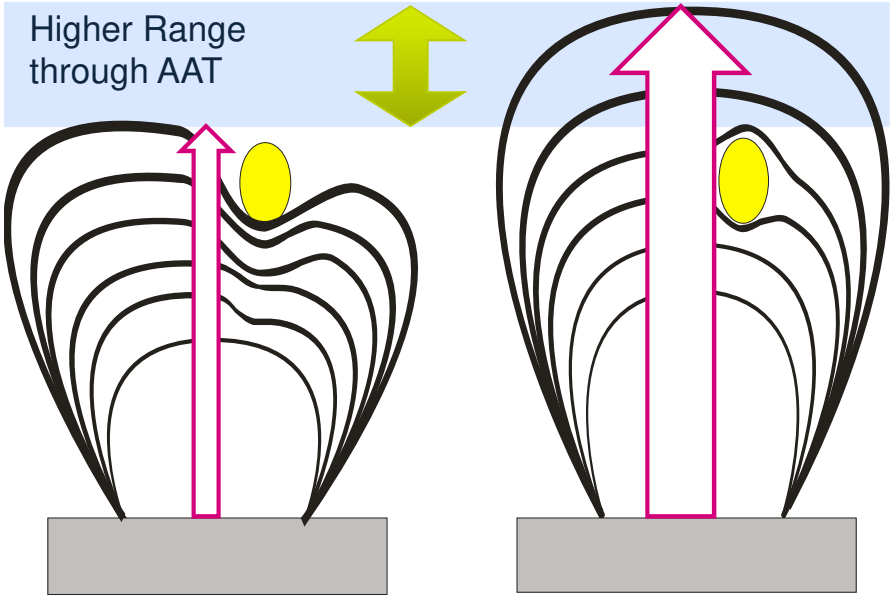
10x more sensitivity



life.augmented

Automatic Antenna Tuning

- Range & Field strength
  - AAT increases the range of an HF reader in bad environmental conditions and sustains maximum output power to the field with best efficiency
- Compensates for environment
  - Automatic antenna tuning analyses the phase shift of the antenna and retunes automatically
- Reduces production cost
  - The antenna can be tuned with an automatic procedure during production to fine adjust the design to different housings.



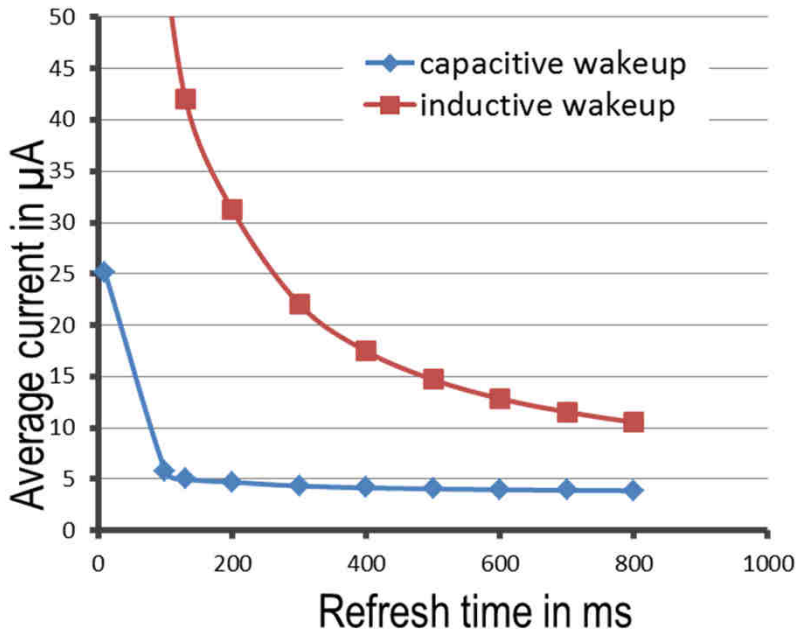
Multiple Tag placement

Multiple tags in the field can be compensated to transfer a maximum of power for each.

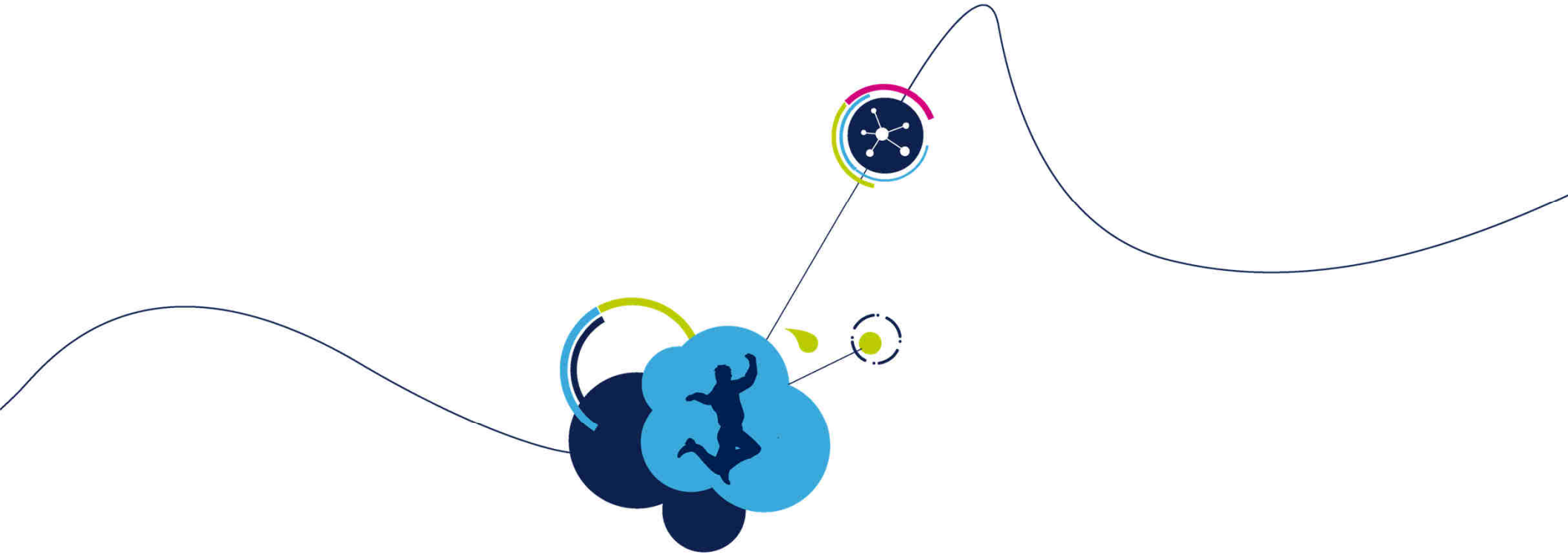
life.augmented

Low Power Wake-up

- Internal wakeup circuitry
  - The ST25R3911B includes a fully programmable wake-up scheme. All relevant parameters like cycle time & sensitivity can be programmed.
  - No MCU required to run the wake-up
  - Capacitive & Inductive wakeup can be combined for sophisticated wake-up scripts
- Capactive wake-up
  - ST25R3911B can detect capacitive changes. E.g. the approach of a hand
- Inductive wake-up
  - The inductive wakeup is dedicated to detect approaching cards only



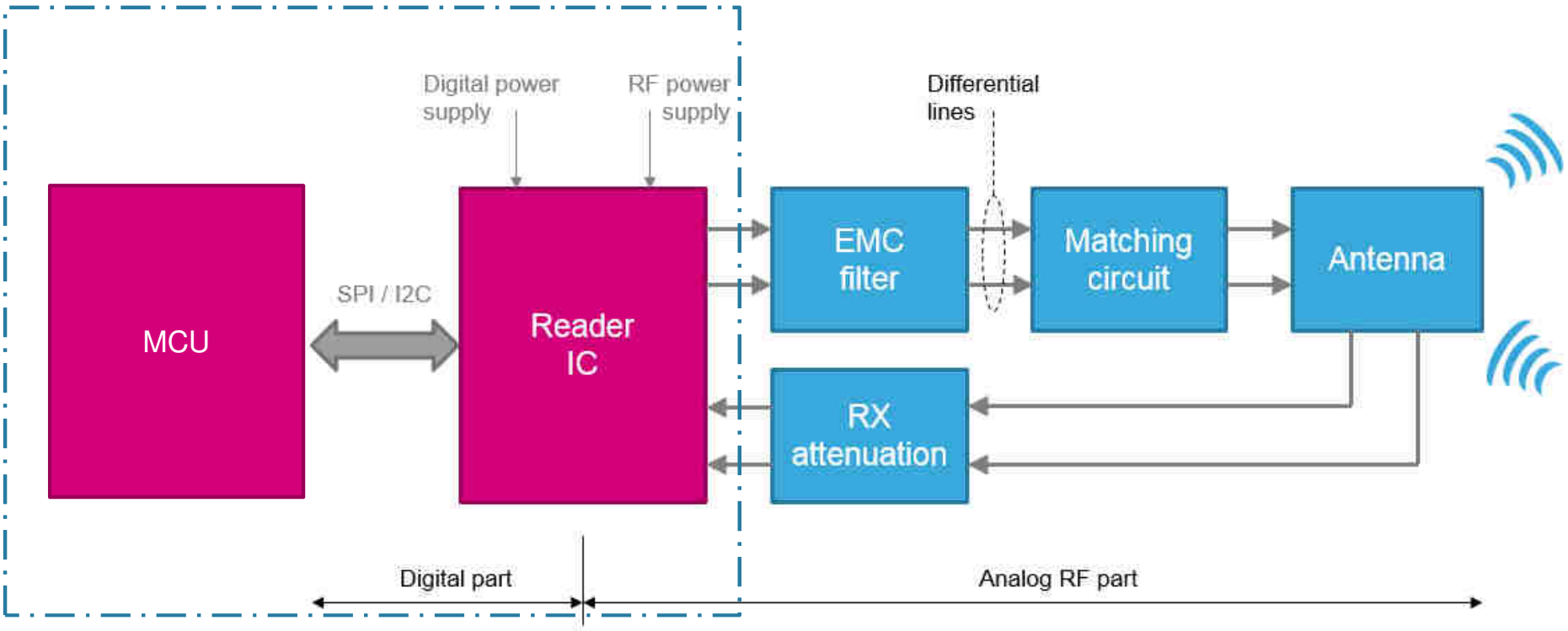
life.augmented



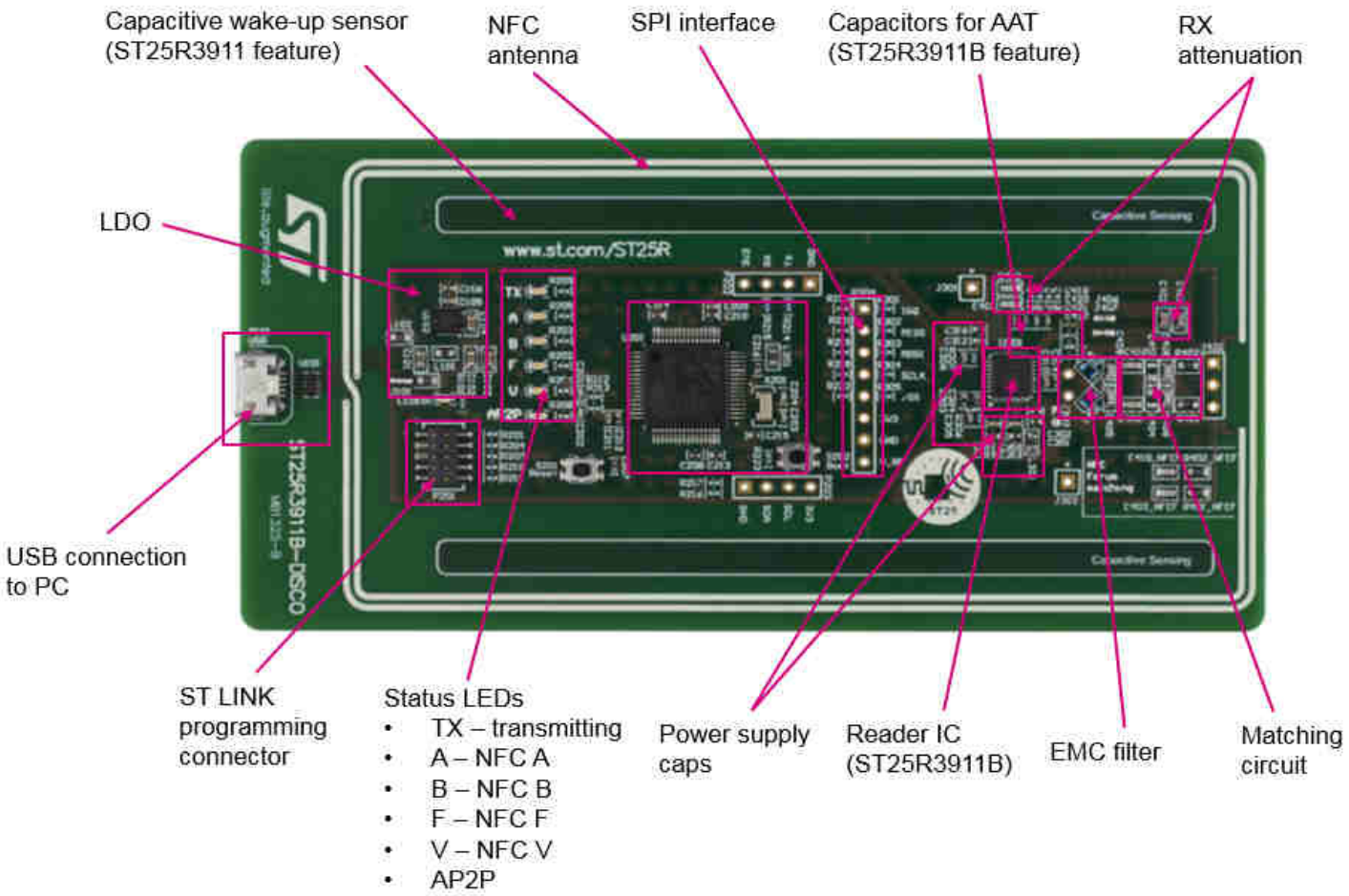
Discovery System Block Diagram

life.augmented

Discovery System Block Diagram



Discovery Board



life.augmented

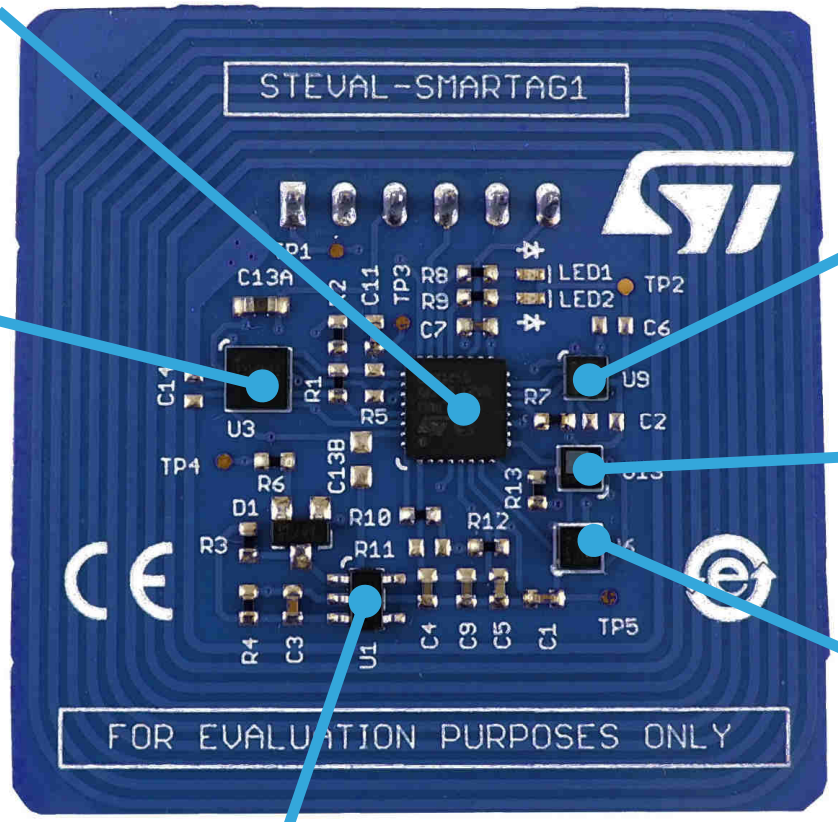
Sensor Tag Hardware Features

life.augmented

Take a good look at the sensor tag

STM32L031K6U6
ARM Cortex-M0+ 32bit Microcontroller

ST25DV64K-JFR6D3
64Kbit dynamic NFC/RFID tag

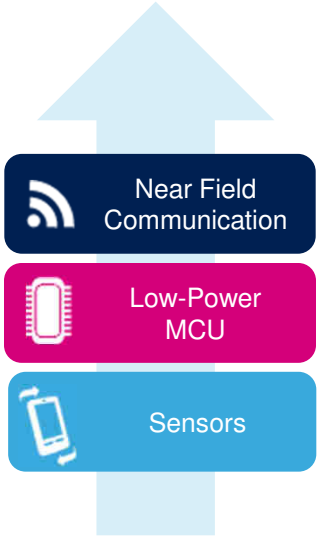


LIS2DW12
Ultra-low power 3-axis digital Accelerometer

LPS22HB
Low-power digital sensor for ambient Pressure

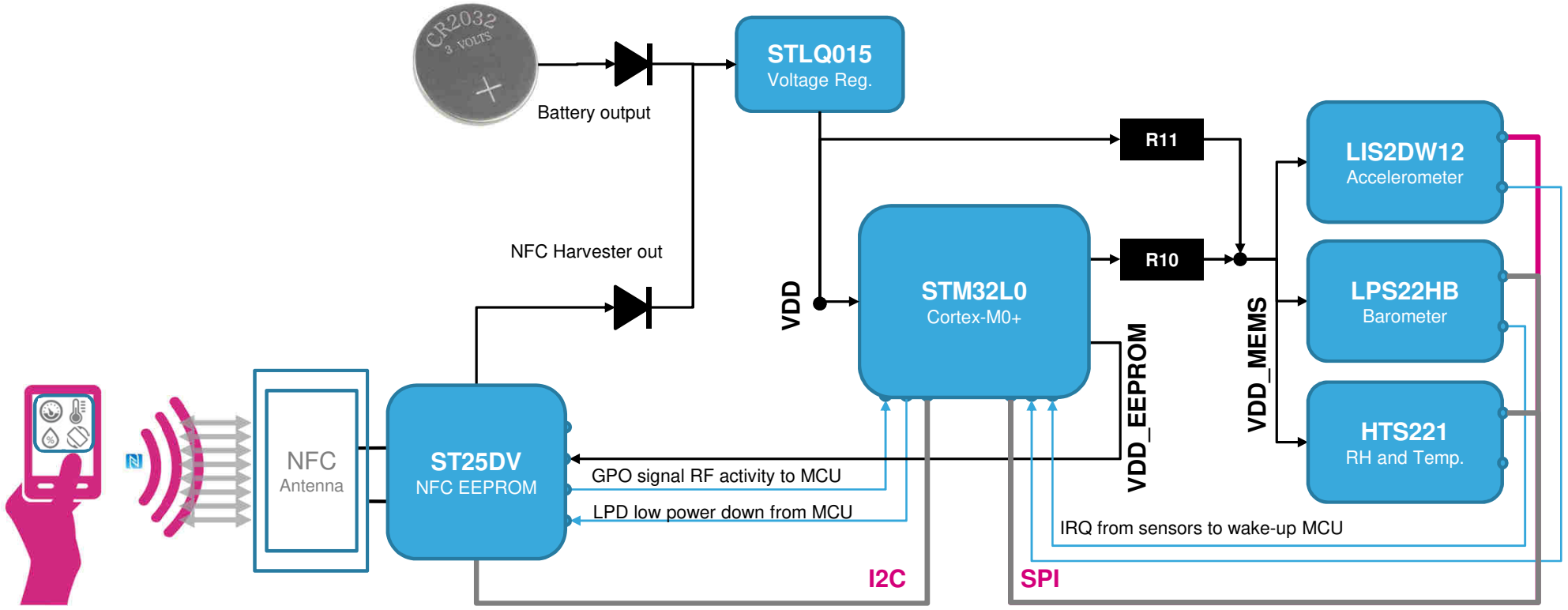
HTS221
Capacitive digital sensor for Relative Humidity and Temperature

STLQ015M18R
Low dropout linear regulator



life.augmented

SmartTag Block Diagram



life.augmented

Power configuration

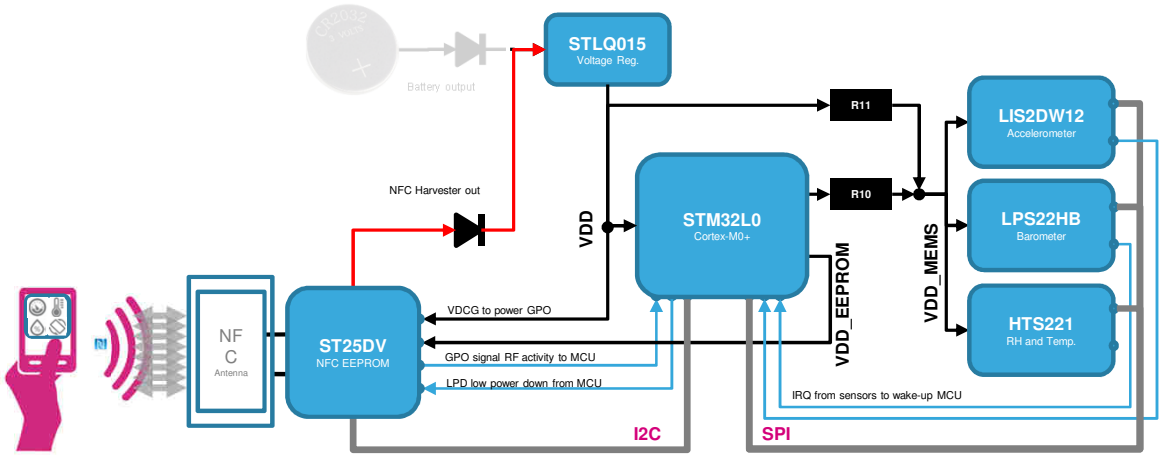
R10 (MEMS power gating) and R11 (always-on MEMS) are mutually exclusive.

Table 1. Solder bridge details for power Path configuration

Solder bridge	Power source	Power sink
R11 (enables always-on MEMS)	VDD (OUT of STLQ015)	VDD_MEMS
R10 (enables MEMS power gating)	VDD_SENS (PB8 of STM32L0)	VDD_MEMS

Battery-less Mode

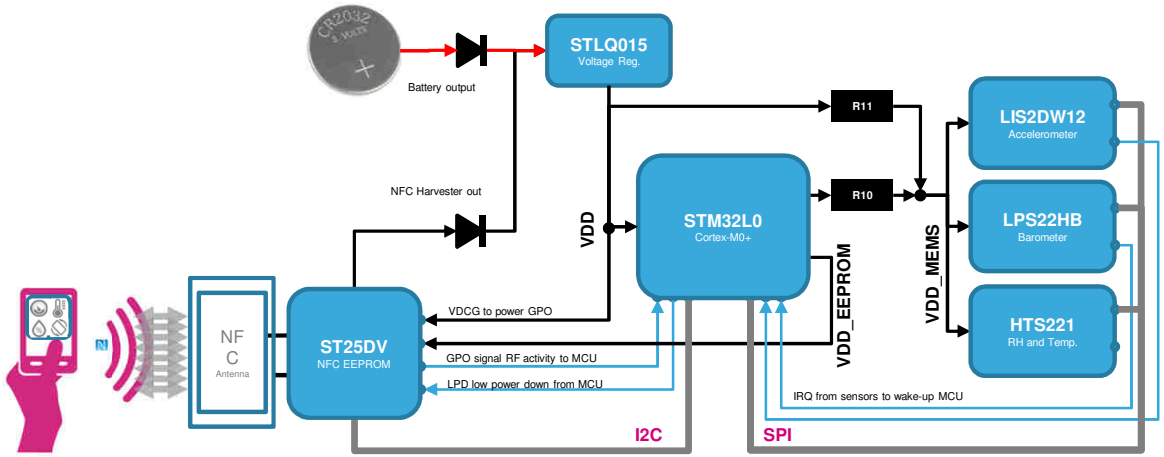
- No battery needed.
- The entire sensor tag is powered by NFC Reader RF field.
- One can perform ONE-SHOT mode



life.augmented

Battery Mode

- Synchronous sampling and logging at interval that is controlled by the MCU Real Time Clock
- Asynchronous sampling and logging when an event detection login embedded in MEMs sensors
- Only accelerometer and ambient pressure sensor can wake-up the microcontroller.

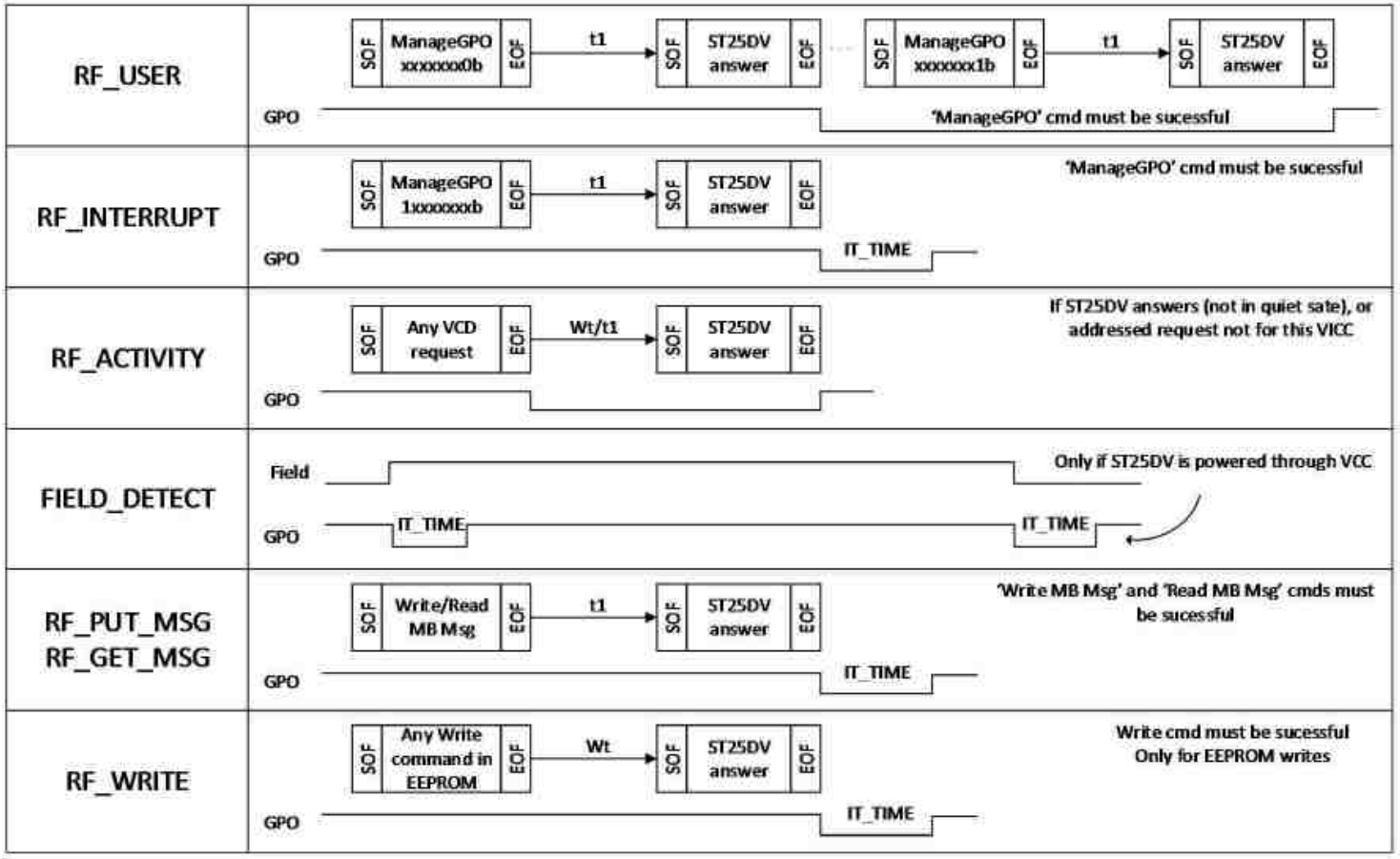


life.augmented

Arbitration and Timing(I2C and RF)

- MCU communicates with ST25DV via I2C
- NFC Reader communicates with ST25DV via RF
- Communication is First Come First Serve
- When RF transaction is in progress, I2C commands are Nack (no acknowledgement)
- When I2C communication is in progress, any RF request receives no-response code of 0xFh

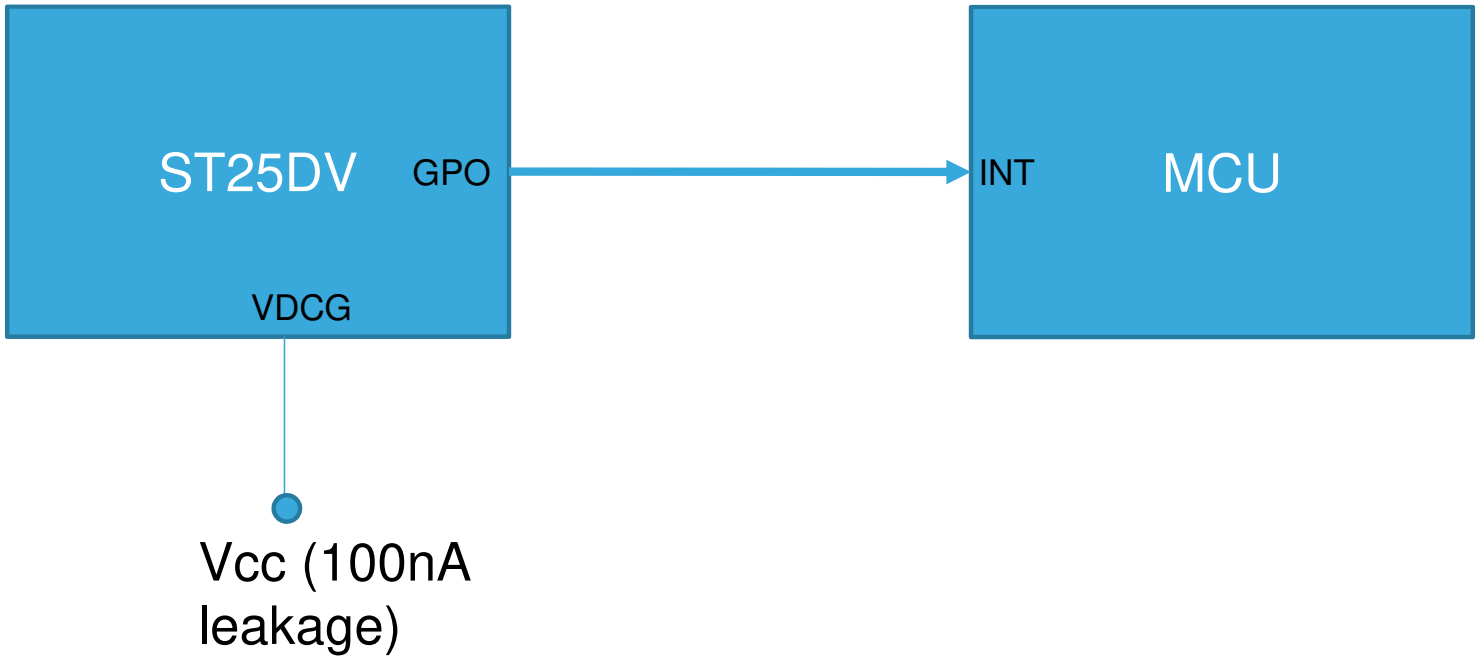
Using ST25DV GPO for arbitration



life.augmented

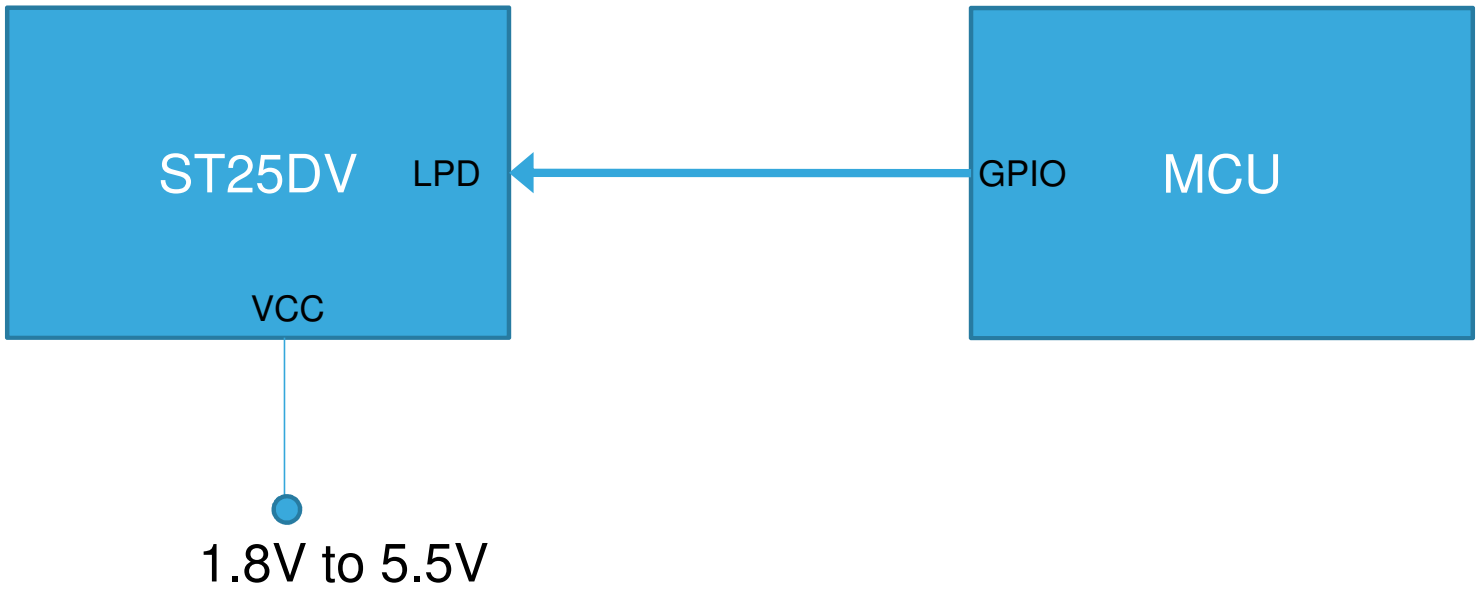
ST25DV GPO Power Block

In this configuration, a RF field detection will wakeup the MCU.

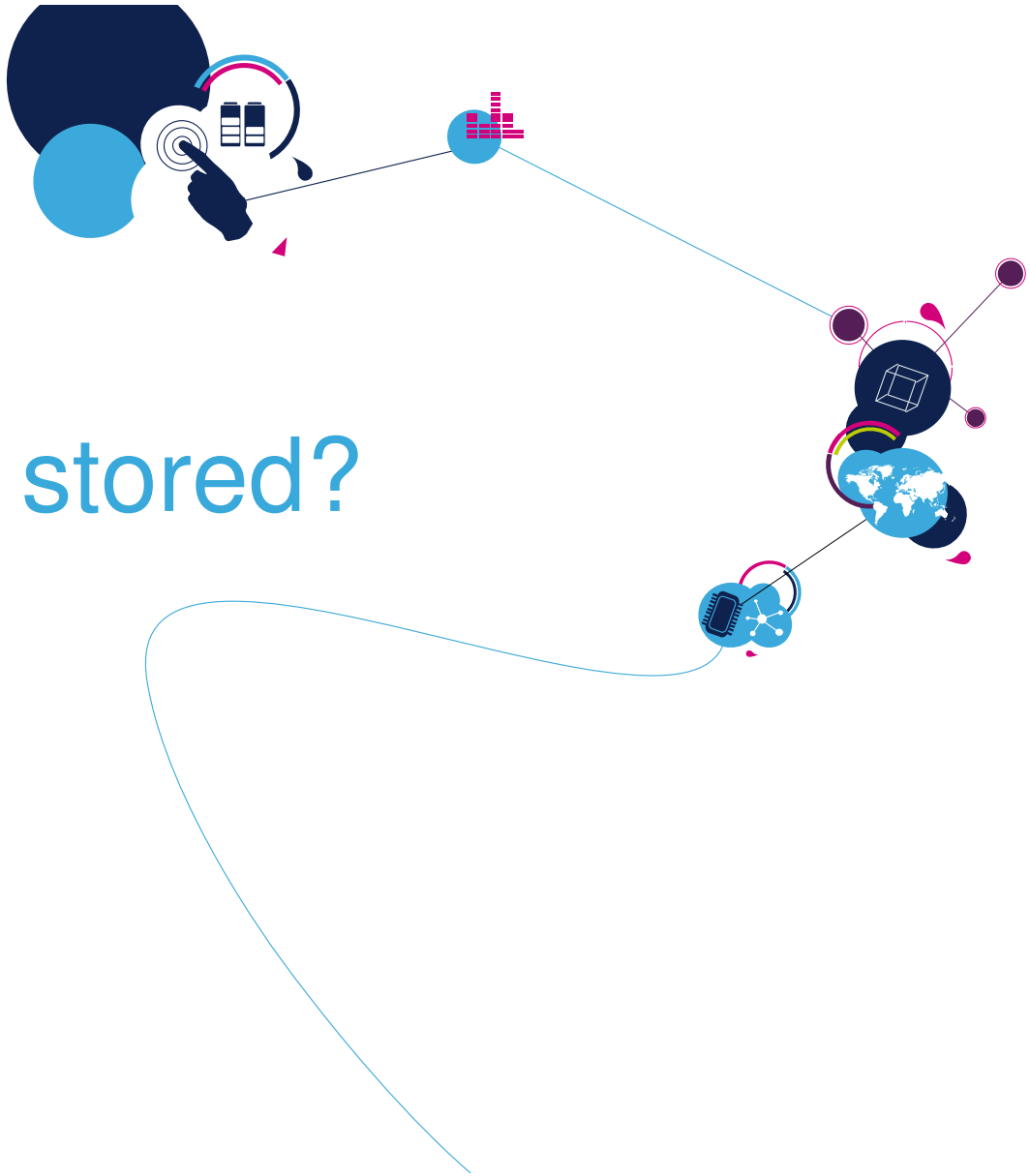


ST25DV LPD

In this configuration, MCU puts the ST25DV in low-power mode consuming less than 1uA by driving the LPD pin HIGH.



life.augmented



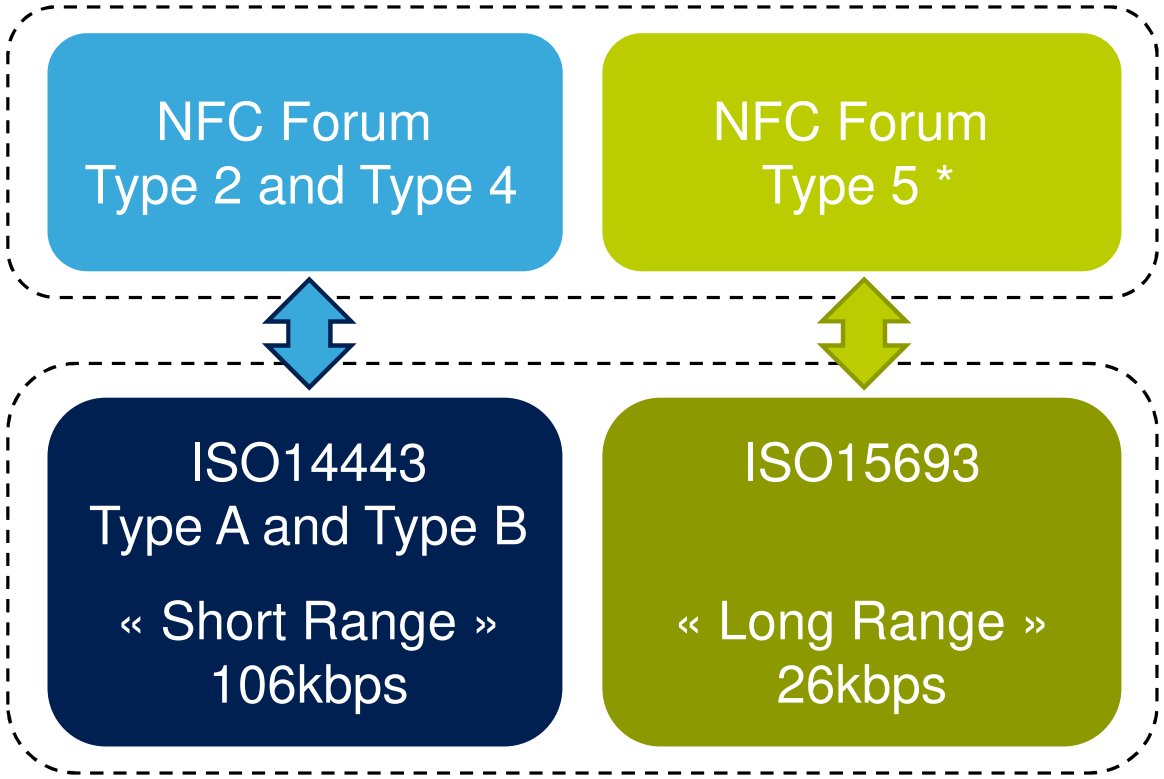
How is data being stored?

life.augmented

NFC and RFID

NFC specification
→ Upper layer SW

ISO standards
→HW/SW protocol
→13.56 MHz



(*) ISO15693 integrated in NFC Forum specifications in October 2015 as NFC Forum type 5 (aka type V)

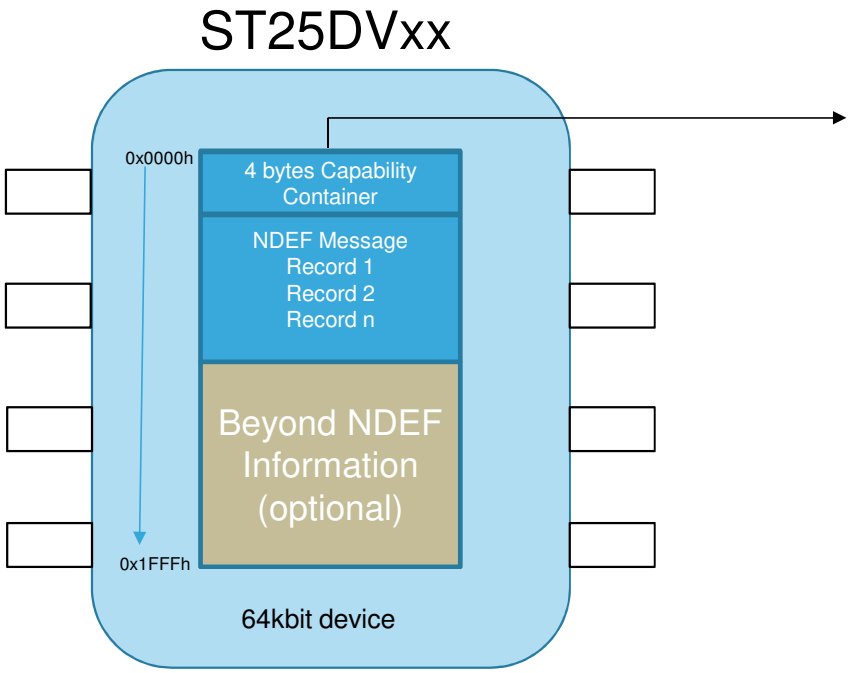
What is a NDEF record?

- Contain the User data.
- The NFC forum defines the format of the NDEF record.
- Different kinds of NDEF records :
  - URL
  - Text
  - SMS
  - Wi-Fi pairing
  - Bluetooth pairing
  - Smart poster….

→ for detailed Info for NDEF record structure, please refer to the Technical Specifications of the NFC Forum!

life.augmented

Sensor data stored as NDEF message



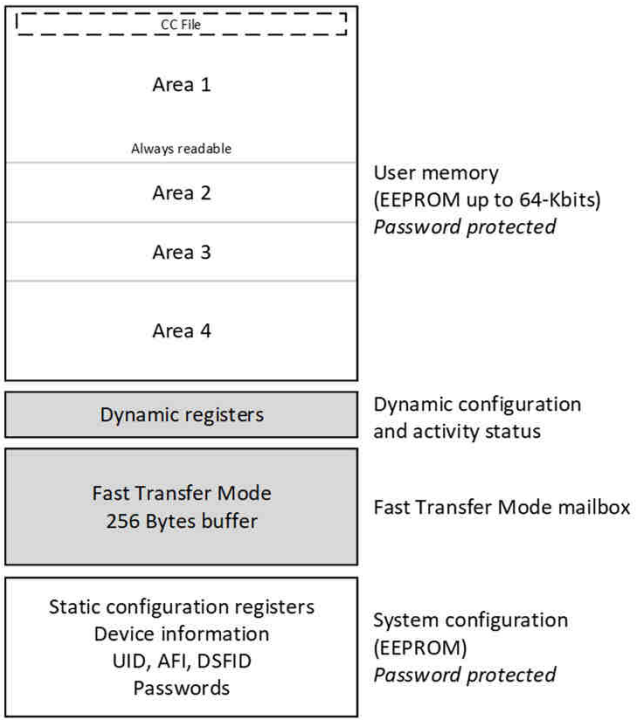
Area	Block	Data	ASCII
01	00	E2 40 00 01	â @ . .
01	01	00 00 03 FF	. . . ÿ
01	02	03 FF 00 FF	. ÿ . ÿ
01	03	44 0E 00 00	D . . .
01	04	00 EB 73 74	. ë s t
01	05	2E 63 6F 6D	. c o m
01	06	3A 73 6D 61	: s m a
01	07	72 74 61 67	r t a g
01	08	01 01 00 02	. . . .
01	09	05 00 01 3F	. . . ?
01	0A	DC A7 CC 00	Ü § Ì .
01	0B	41 40 32 28	A @ 2 (
01	0C	54 4B 38 04	T K 8 .
01	0D	00 01 00 00	. . . .
01	0E	05 00 22 00	. . " .
01	0F	77 B9 CC 00	w ¹ Ì .
01	10	05 00 22 00	. . " .
01	11	0A 00 22 00	. . " .
01	12	40 40 3B 3A	@ @ ; :
01	13	05 00 22 00	. . " .
01	14	12 A9 CC 00	. © Ì .
01	15	F0 A8 CC 00	ð ¨ Ì .
01	16	E9 86 6E 04	é † n .
01	17	05 00 88 00	. . ˆ .
01	18	05 00 22 00	. . " .
01	19	C4 0E 98 6E	Ä . ˜ n
01	1A	0A 00 22 00	. . " .
01	1B	84 0E 98 6E	„ . ˜ n
01	1C	0F 00 22 00	. . " .

- NDEF NFC Data Exchange Format

ST25DV Memory

- Memory organization
  - User memory
    - 4Kbits/16Kbits/64Kbits EEPROM
    - 4 configurable areas
    - Areas protectable by passwords
  - System configuration
    - EEPROM
    - Static registers, define behavior at boot
    - Device identifiers and passwords
    - Protected by password
  - Dynamic configuration
    - Dynamic registers
    - Status and temporary configuration
  - Fast transfer mode buffer
    - Message exchange between I2C and RF
    - 256 Bytes mailbox buffer

ST25DV memory organization



life.augmented

Sensor data structure

- Each event requires 8 bytes
  - 4 bytes are time stamp
  - 4 bytes are sensor data
- When orientation is also logged, 12 bytes are required vs. 8 bytes
- When memory is full, last sample pointer will be at the beginning of data

Tag configuration data and addresses

Value	Byte Addr	Add	Add I2C	Add RF	Note					
SmarTag SW Versions	RecordVersion	0	0	0						
	Major FW Version	1	1	0						
	Minor	2	2	0						
	Patch	3	3	0						
SmarTag Conf	Sample rate (Low)	4	4	1						
	Sample rate (High)	5	5	1						
	Log Mode	6	6	1	0x0 Inactive	0x1 Sampling Log	0x3 Sampling with Ths	0x4 Save next Sample		
	Sensor Enable Flags	7	7	1	0x01 Temp	0x02 Hum	0x04 Press	0x08 Acc	0x10 6DOrientation	
TimeStamp		8	8	2	Year = (((TimeStamp >> 26)) & 0x1F);	Minutes = (((TimeStamp >> 6)) & 0x3F);				
		9	9	2	Date = (((TimeStamp >> 21)) & 0x1F);	Seconds = (((TimeStamp)) & 0x3F);				
		10	A	2	Month = (((TimeStamp >> 17)) & 0x0F);					
		11	B	2	Hours = (((TimeStamp >> 12)) & 0x1F);					
Thresholds	T Max	12	C	3						
Used for Log Mode==0x3	T Min	13	D	3						
	H Max	14	E	3						
	H Min	15	F	3						
	Pmax(12) Pmin(12)	16	10	4						
		17	11	4						
		18	12	4						
	AccMax(8bit)	19	13	4						

life.augmented

Tag configuration data and addresses

Value	Byte Addr	Add	Add I2C	Add RF		
ADDR COMMAND REPLY	Read New Conf	20	14	5	Read New Configuration and new RTC value setted by App	
	Single Shot Ready	21	15	5	Written Sigle shot result by SmarTag	
	RFU	22	16	5		
	RFU	23	17	5		
MAX_T_32BITDATATIME_ADDR		24	18	6	TimeStamp = 0 <<31;	TimeStamp |= (((uint32_t) (Hours)) << 12);
		25	19	6	TimeStamp |= (((uint32_t) (Year &0x1F))	TimeStamp |= (((uint32_t) (Minutes)) << 6);
		26	1A	6	TimeStamp |= (((uint32_t) (Date)) <<	TimeStamp |= (((uint32_t) (Seconds)));
		27	1B	6	TimeStamp |= (((uint32_t) (Month)) << 17);	

life.augmented

Tag configuration data and addresses

Value	Byte Addr	Add	Add I2C	Add RF
MIN_T_32BITDATATIME_ADDR		28	1C	7
		29	1D	7
		30	1E	7
		31	1F	7
MAX_H_32BITDATATIME_ADDR		32	20	8
		33	21	8
		34	22	8
		35	23	8
MIN_H_32BITDATATIME_ADDR		36	24	9
		37	25	9
		38	26	9
		39	27	9
Max&Min for T and H	Max T	40	28	A
	MinT	41	29	A
	MaxH	42	2A	A
	MinH	43	2B	A
MAX_P_32BITDATATIME_ADDR		44	2C	B
		45	2D	B
		46	2E	B
		47	2F	B
MIN_P_32BITDATATIME_ADDR		48	30	C
		49	31	C
		50	32	C
		51	33	C

MIN_P_32BITDATATIME_ADDR		48	30	C
		49	31	C
		50	32	C
		51	33	C
MAX_Acc_32BITDATATIME_ADDR		52	34	D
		53	35	D
		54	36	D
		55	37	D
Max&Min for P and Max Acc	Pmax(12) Pmin(12)	56	38	E
		57	39	E
		58	3A	E
	AccMax(8bit)	59	3B	E

life.augmented

Tag configuration data and addresses

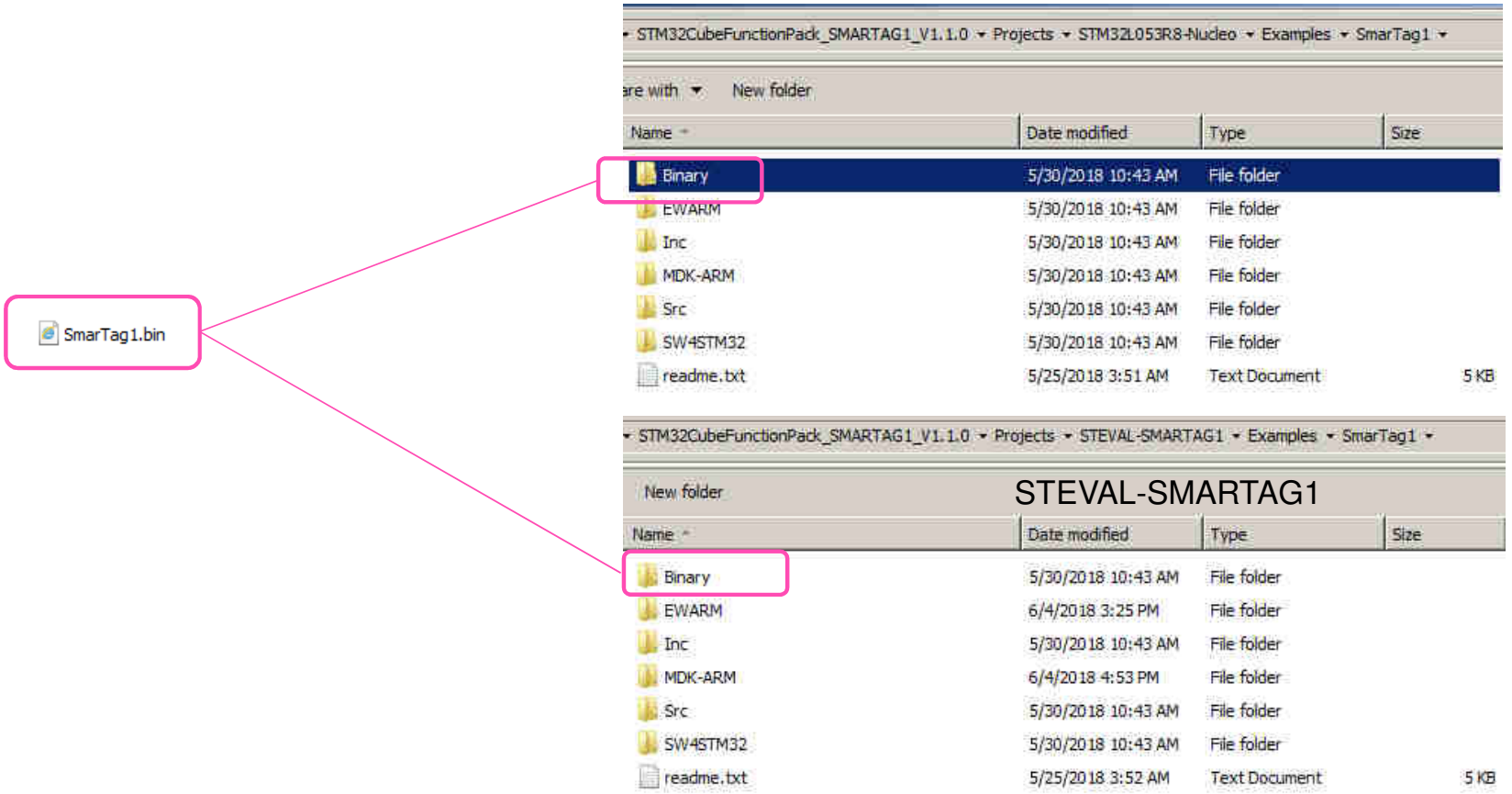
Value	Byte Addr	Add	Add I2C	Add RF			
DataLog (TimesTamp 4 bytes + Values 4 bytes)	TimeStamp1	64	40	10	TimeStamp = Sync/Async <<31;	TimeStamp |= (((uint32_t) (Hours)) << 12);	
(1015 Possible values without Header NDEF)		65	41	10	TimeStamp |= (((uint32_t) (Year &0x1F))	TimeStamp |= (((uint32_t) (Minutes)) << 6);	
		66	42	10	TimeStamp |= (((uint32_t) (Date)) <<	TimeStamp |= (((uint32_t) (Seconds)));	
		67	43	10	TimeStamp |= (((uint32_t) (Month)) << 17);		
	Values1	68	44	11	For Sync Event:		
		69	45	11	For Values from MSB to LSB		
		70	46	11	Press 12 bits, Temp 7Bits, Hum 7Bits, Acc 6Bits		
		71	47	11			
	TimeStamp2	72	48	12			
		73	49	12			
		74	4A	12			
		75	4B	12			
	Values2	76	4C	13	For Async Event:		
					First 3 bits: Orientation type:		Next 6 bits: Acc max for WakeUp event
					#define ORIENTATION_UNDEF 0x00	Next 6 bits: Type of event:	
					#define ORIENTATION_RIGHT 0x01	#define ACC_WAKEUP 0x01	
					#define ORIENTATION_TOP 0x02	#define ACC_6D_ORIENTATION 0x02	
					#define ORIENTATION_LEFT 0x03	#define ACC_SINGLE_TAP 0x04	
					#define ORIENTATION_BOTTOM 0x04	#define ACC_DOUBLE_TAP 0x08	
					#define ORIENTATION_UP 0x05	#define ACC_FREE_FALL 0x10	
		77	4D	13	#define ORIENTATION_DOWN 0x06	#define ACC_TILT 0x20	

life.augmented

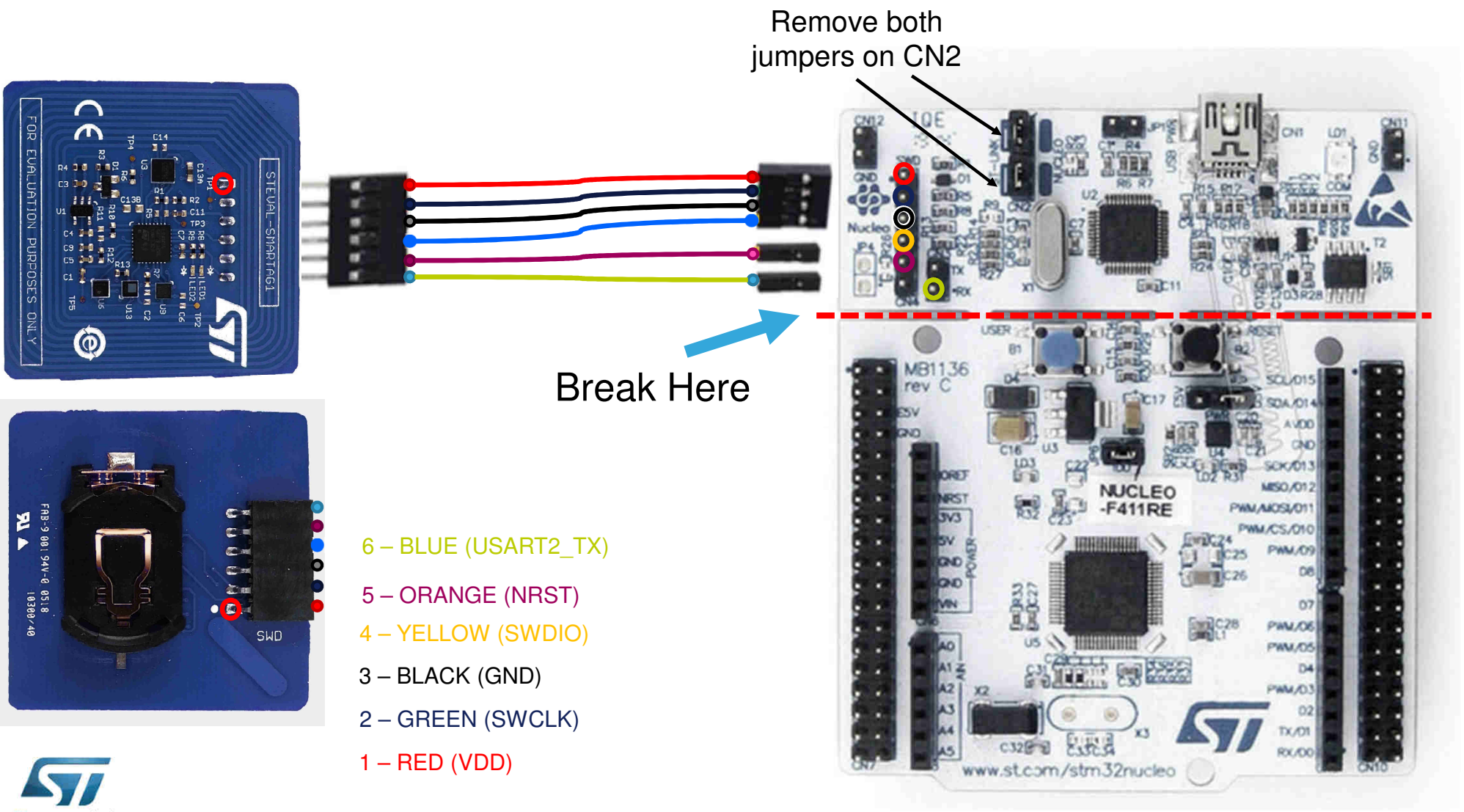
Downloading Firmware

life.augmented

Downloading Binary



life.augmented



6 – BLUE (USART2_TX)

5 – ORANGE (NRST)

4 – YELLOW (SWDIO)

3 – BLACK (GND)

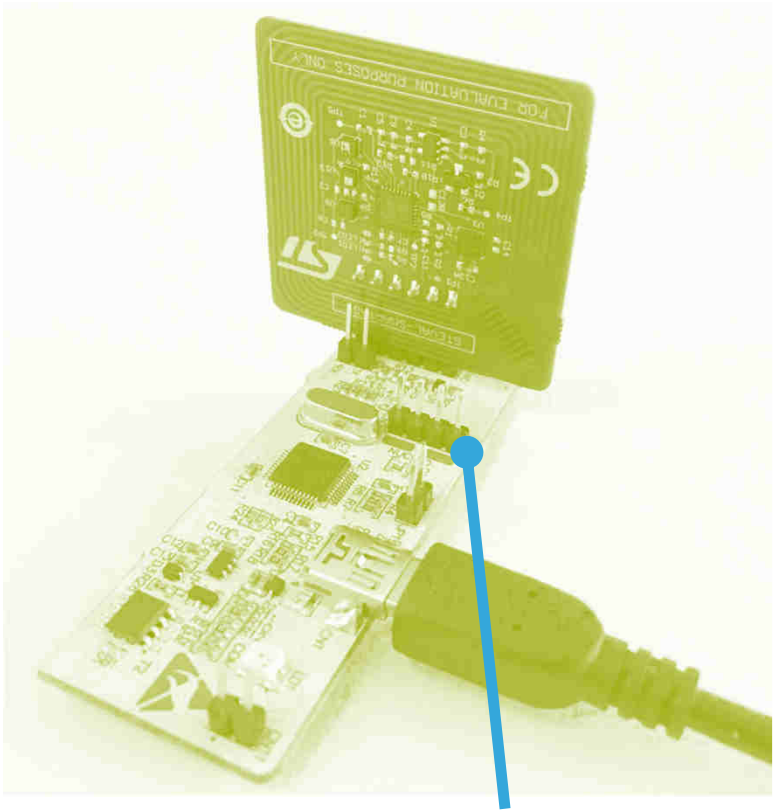
2 – GREEN (SWCLK)

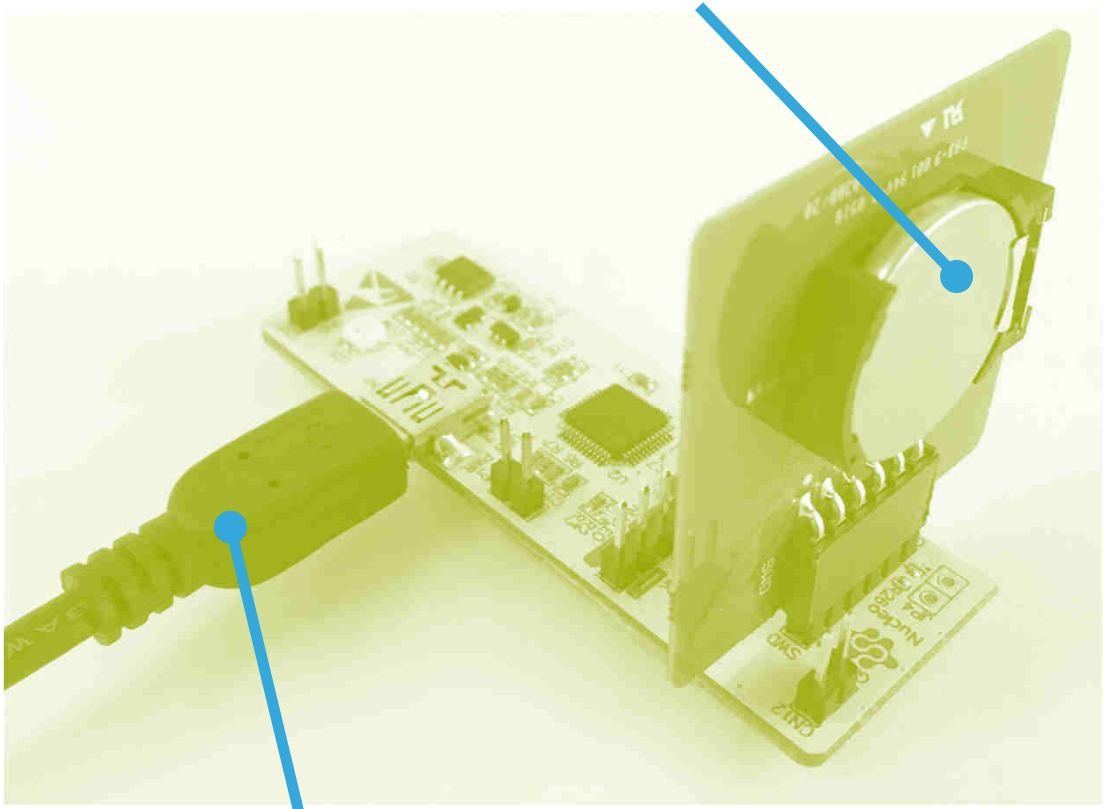
1 – RED (VDD)

life.augmented

CR2032 battery

NFCSensorTAG must be powered
to be able to flash and debug





CN2

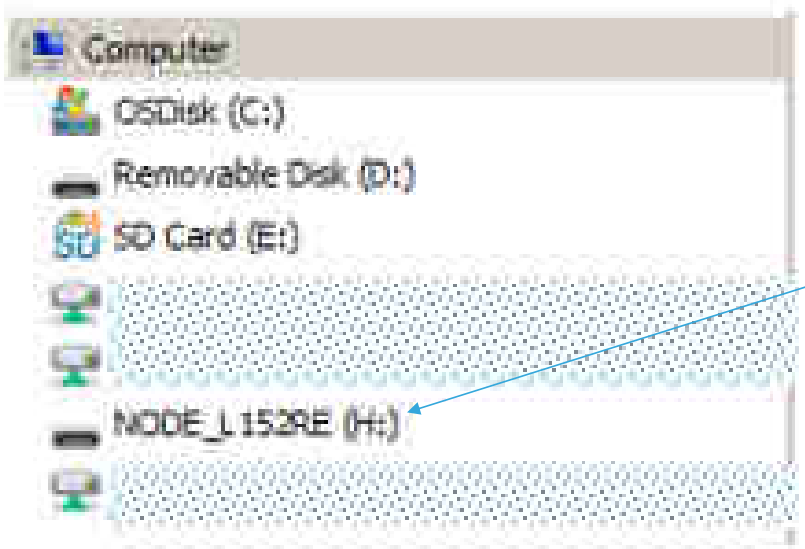
Remove jumpers to be able
to flash and debug

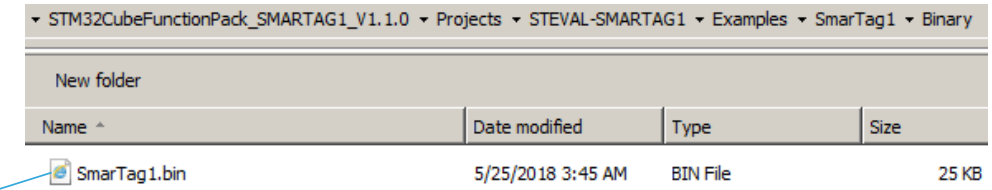
USB

Connect to PC to be able to
flash and debug

life.augmented

Binary Drag and Drop

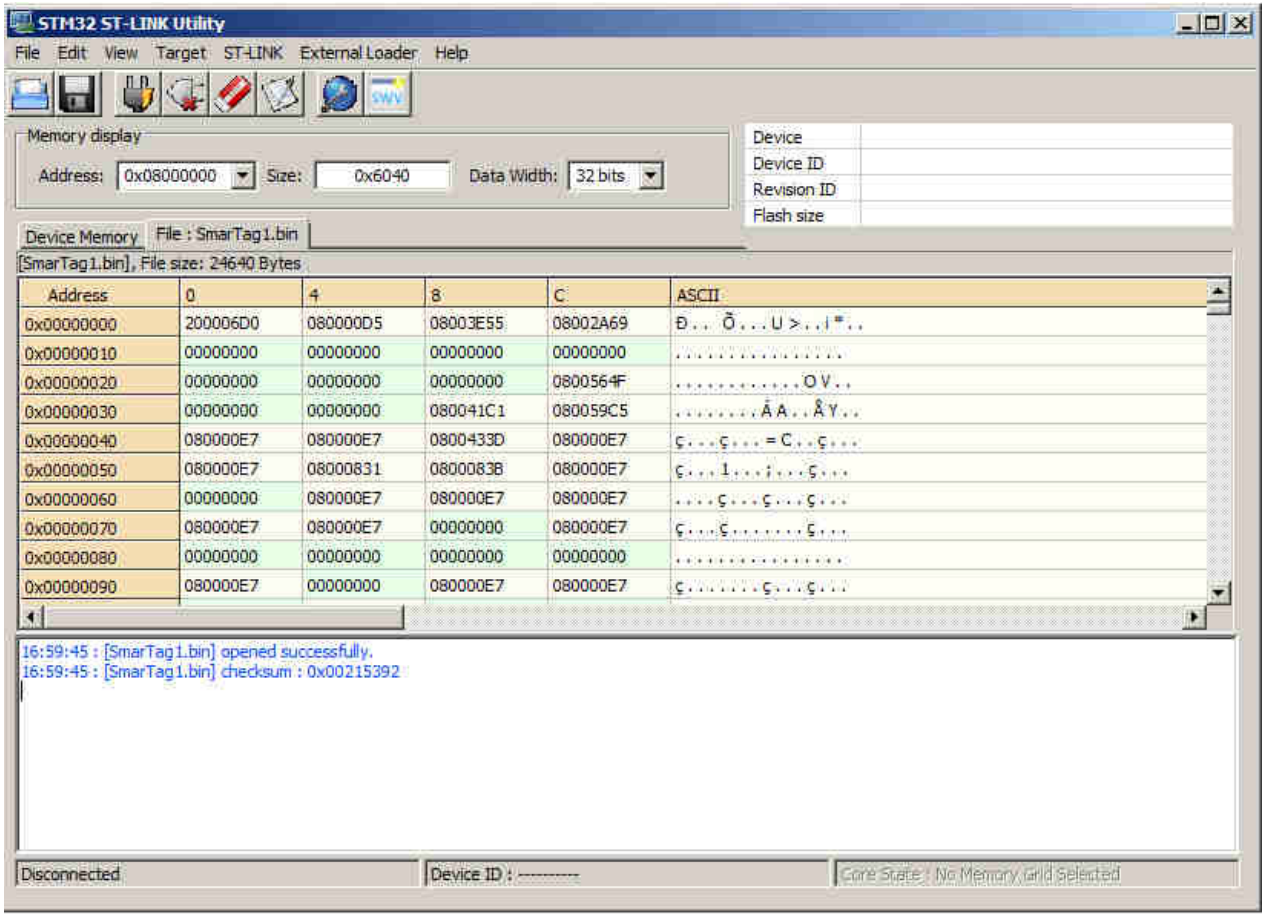




Note: The name of the drive node does not matter. You can use the STLINK part from any STM32 Nucleo board

life.augmented

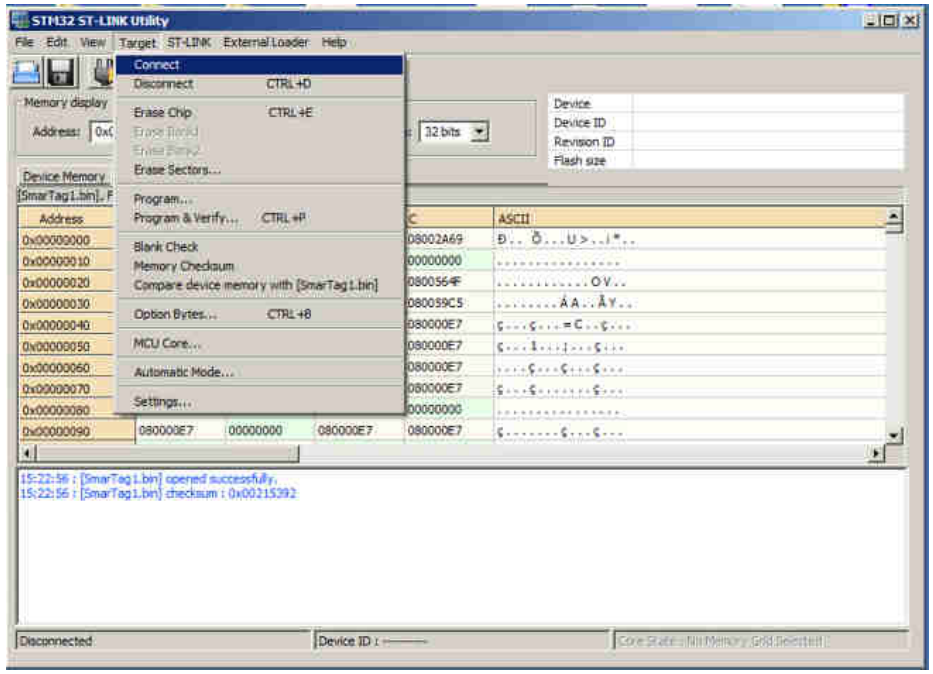
Using STM32 ST-LINK Utility

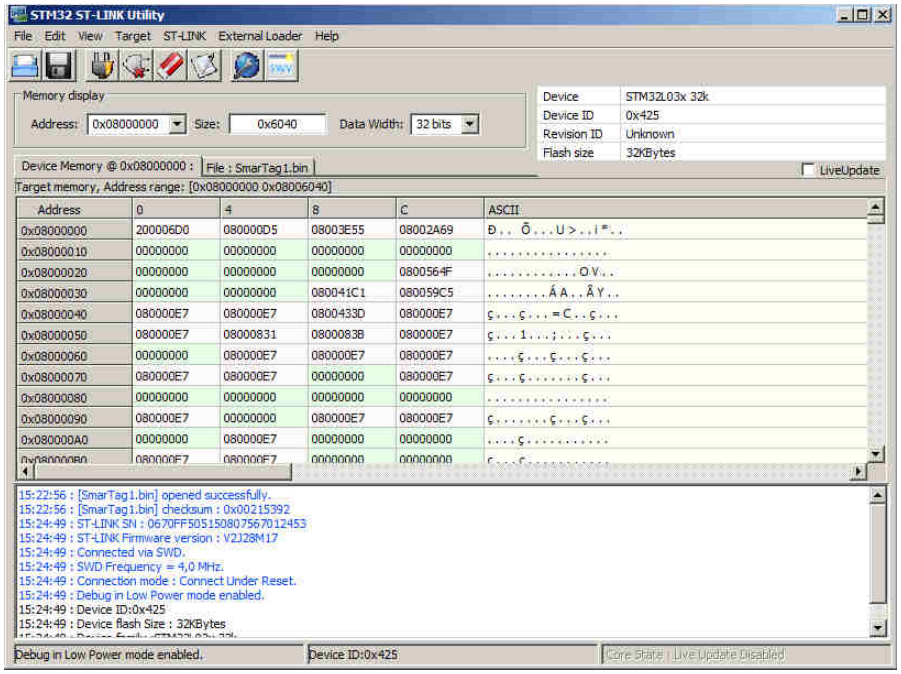


life.augmented

https://www.st.com/en/development-tools/stsw-link004.html

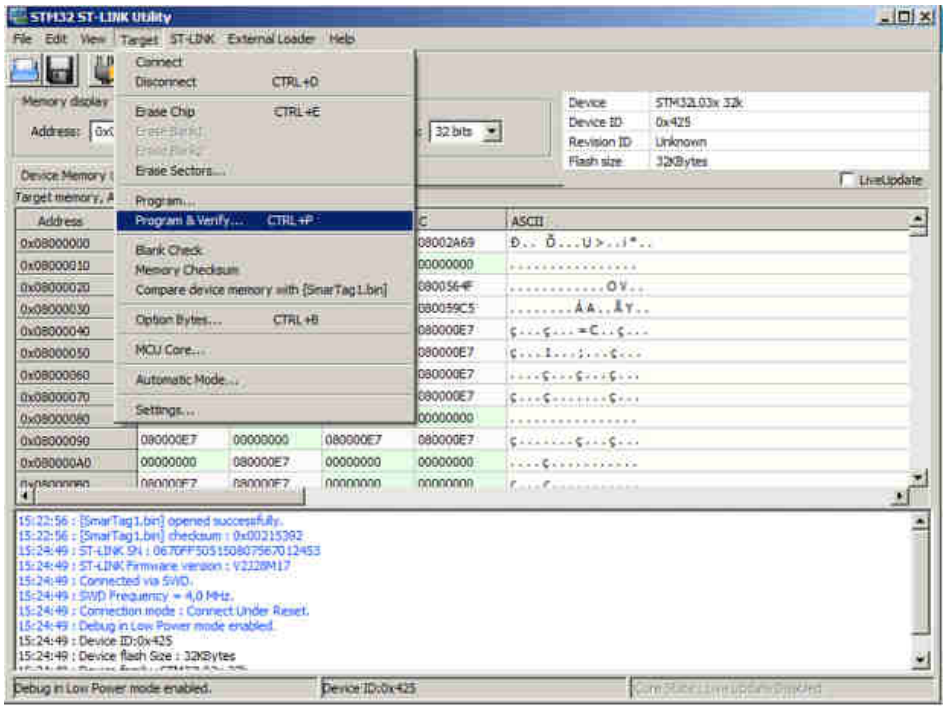
Connecting to ST-LINK

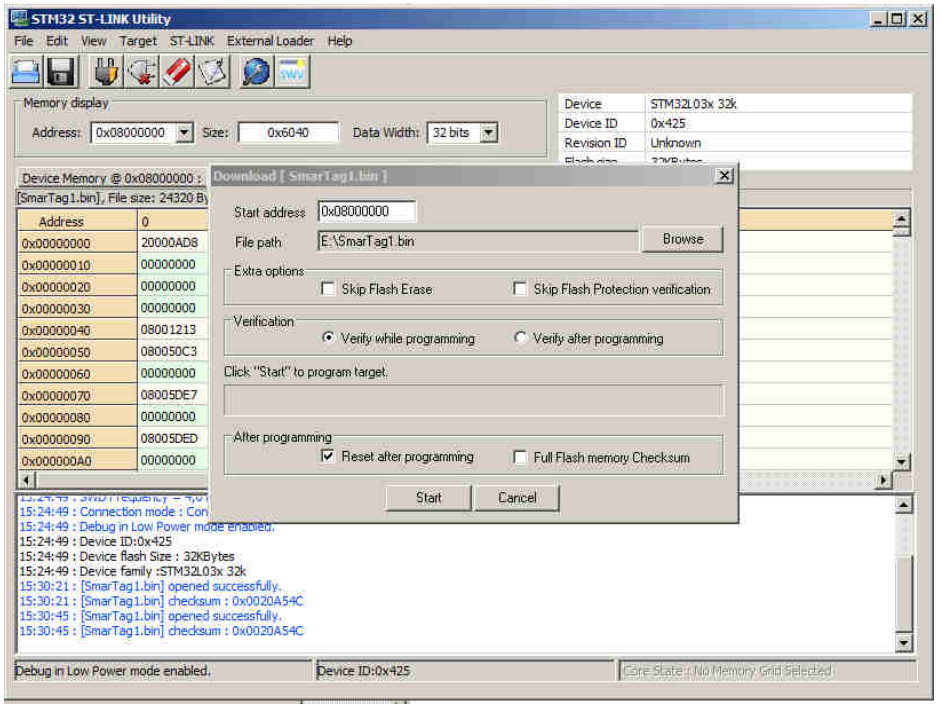




life.augmented

Start Programming





life.augmented

Let’s get to the Hands-On Section

life.augmented

LAB Preparation

- For the workshop ST will provide

ST USB Key
with presentation of the workshop

Micro USB Cable

ST25R3911B Discovery Kit

CR2032 Battery

NFC Sensor Tag Evaluation Board

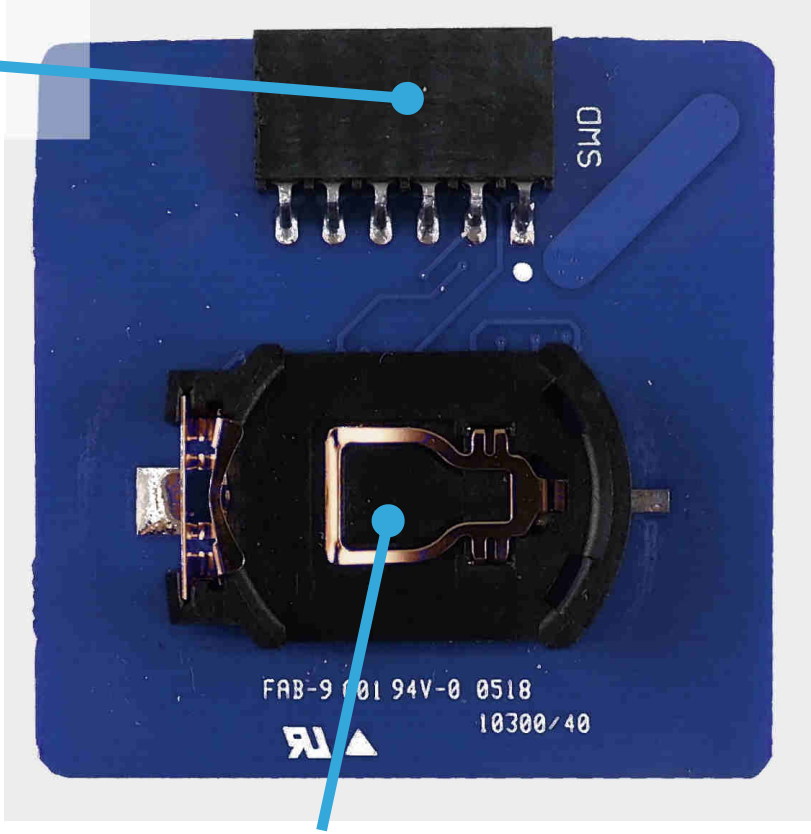
life.augmented

Installation of ST25PC Software

- Locate the executable en.ST25PC-NFC.exe in the flash drive.
- Click and install it. In some computers, the installation might need internet access to download certain Microsoft Visual C++ Redistributable (x86)
- Say “yes” to driver install.

Install the battery

SWD connector
To ST-Link/V2



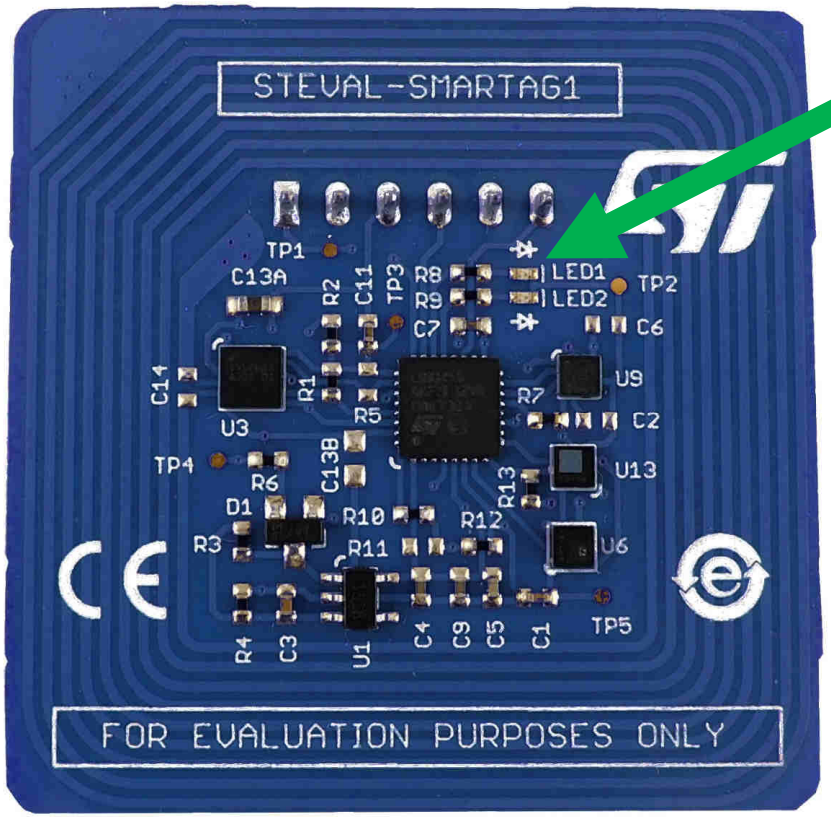
Battery holder
CR2032

Important Note:
Load your coin cell battery with + facing out.

life.augmented

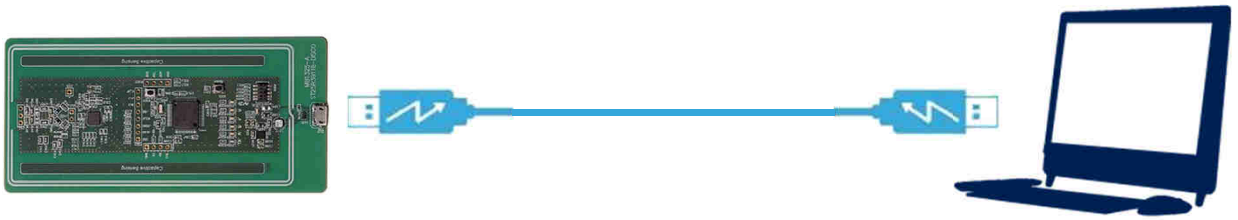
Check if battery and tag are good!

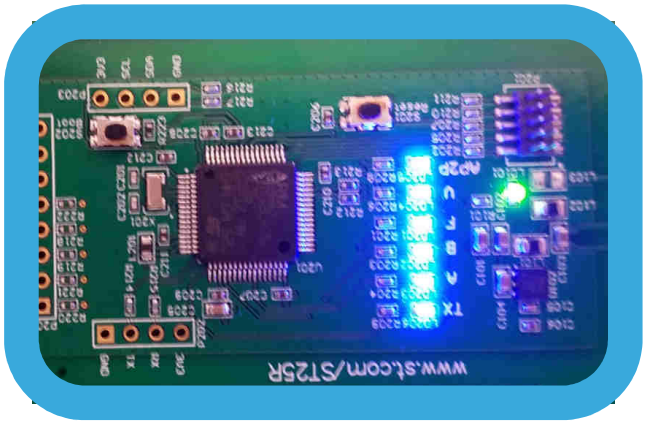
After 8 seconds, LED1 should blink every 5 seconds



life.augmented

Connect your reader

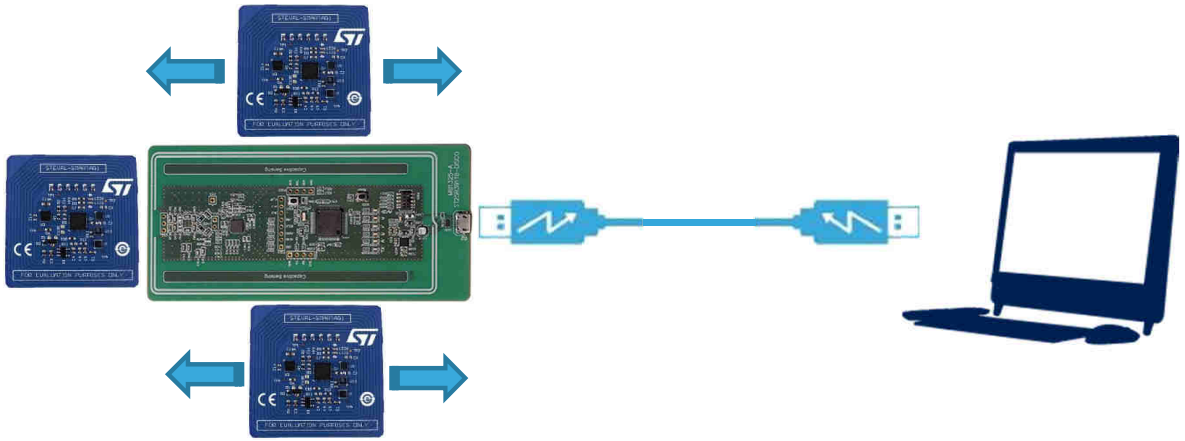




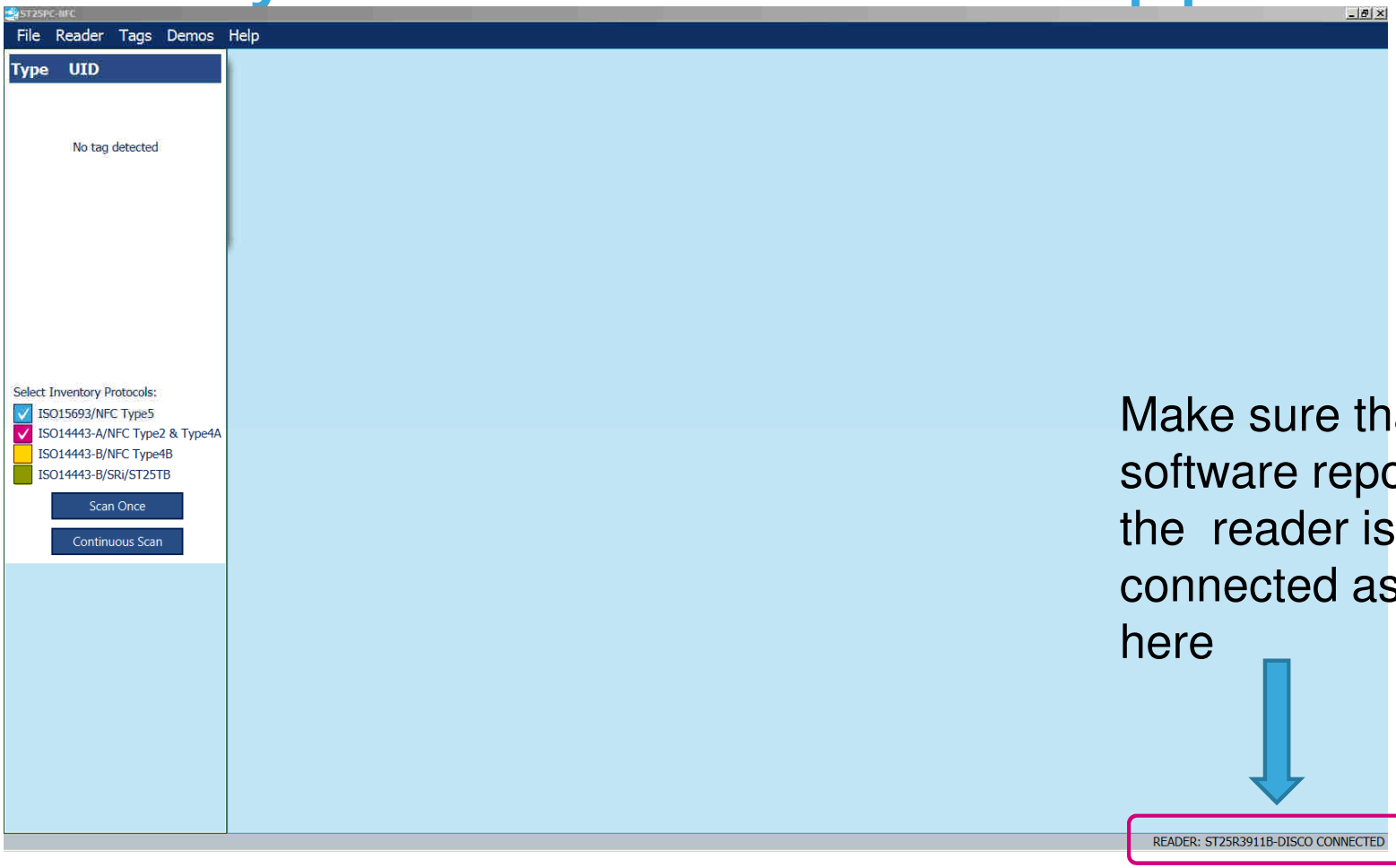
Tag Placement while reading

Important Note:
You can place the tag on top or under the ST25R3911B ONLY if there is insulation between them.

Or you will risk shorting out the components!

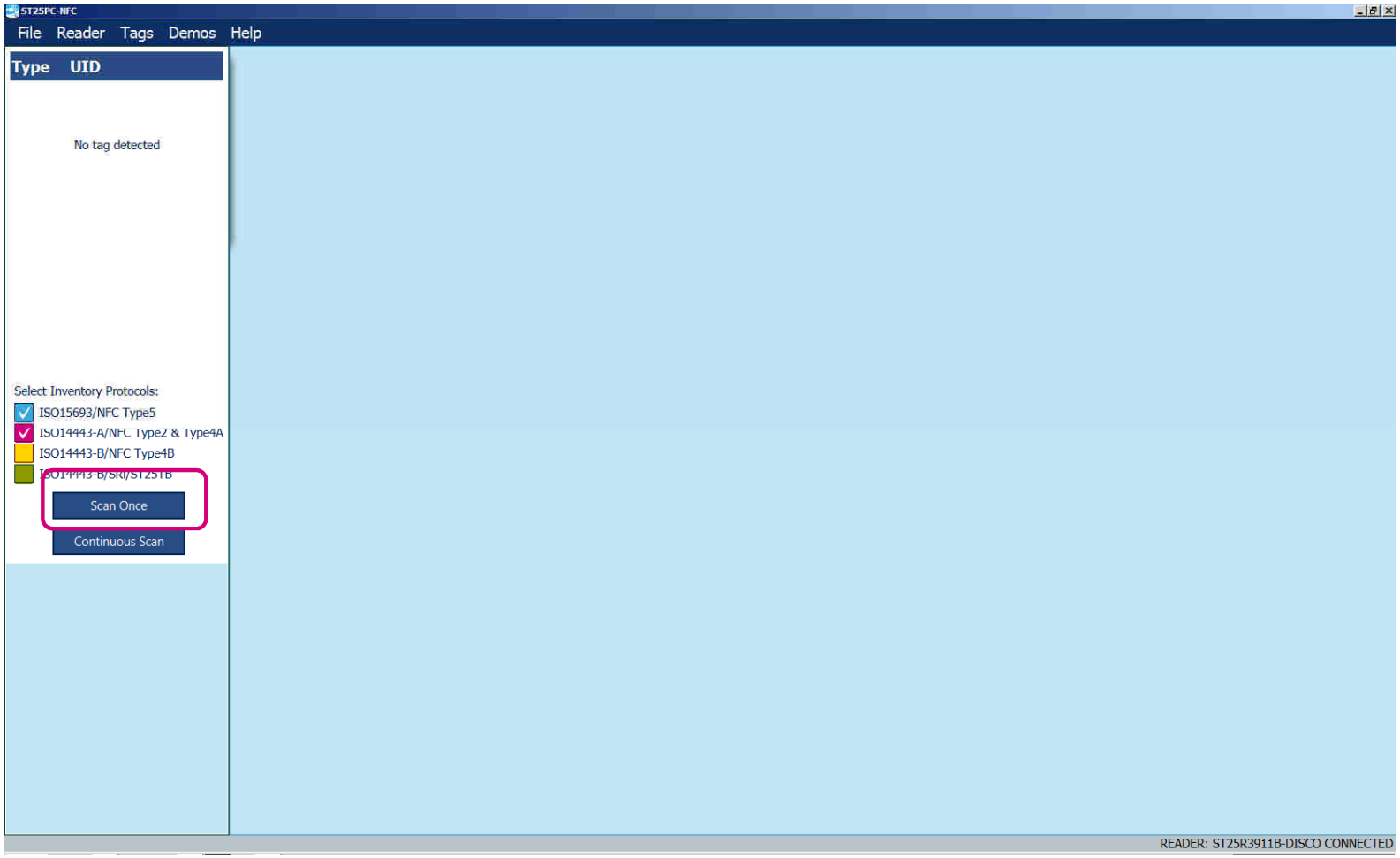


Now run your ST25PC-NFC Application

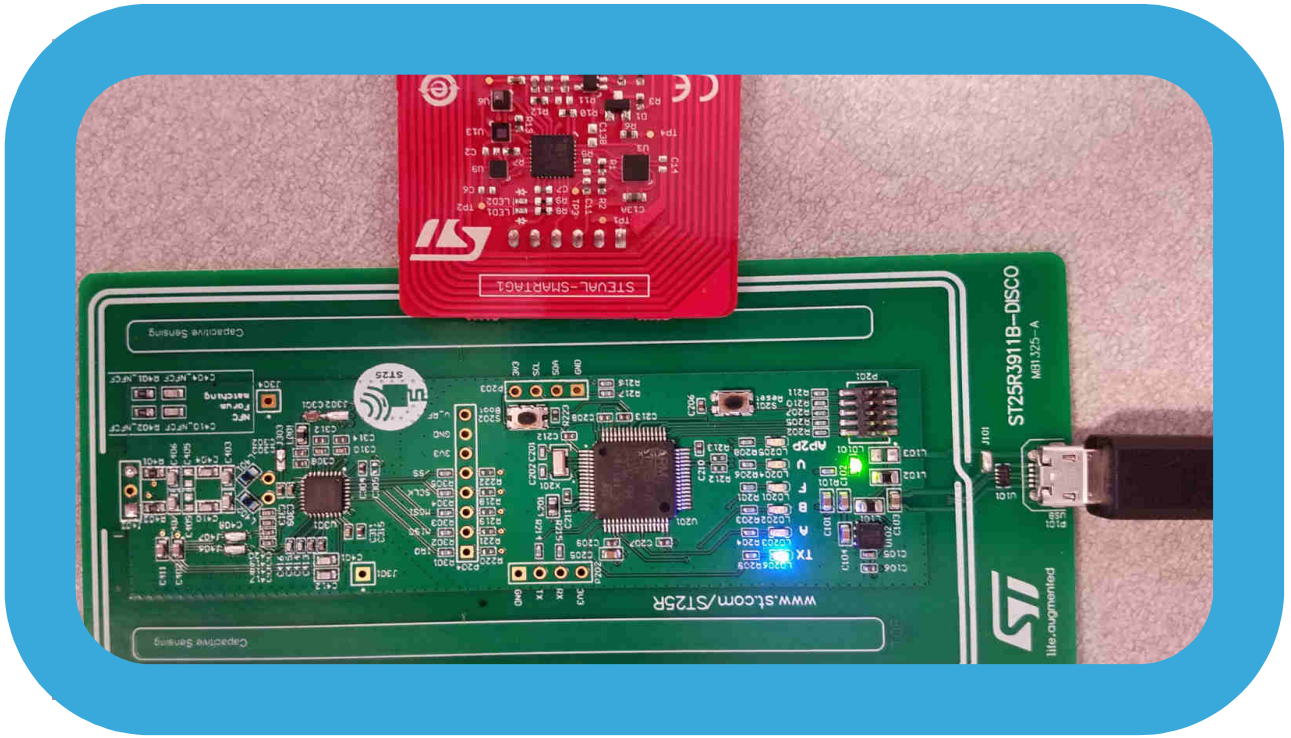


Make sure that the software reports that the reader is connected as seen here

First, scan for the tag

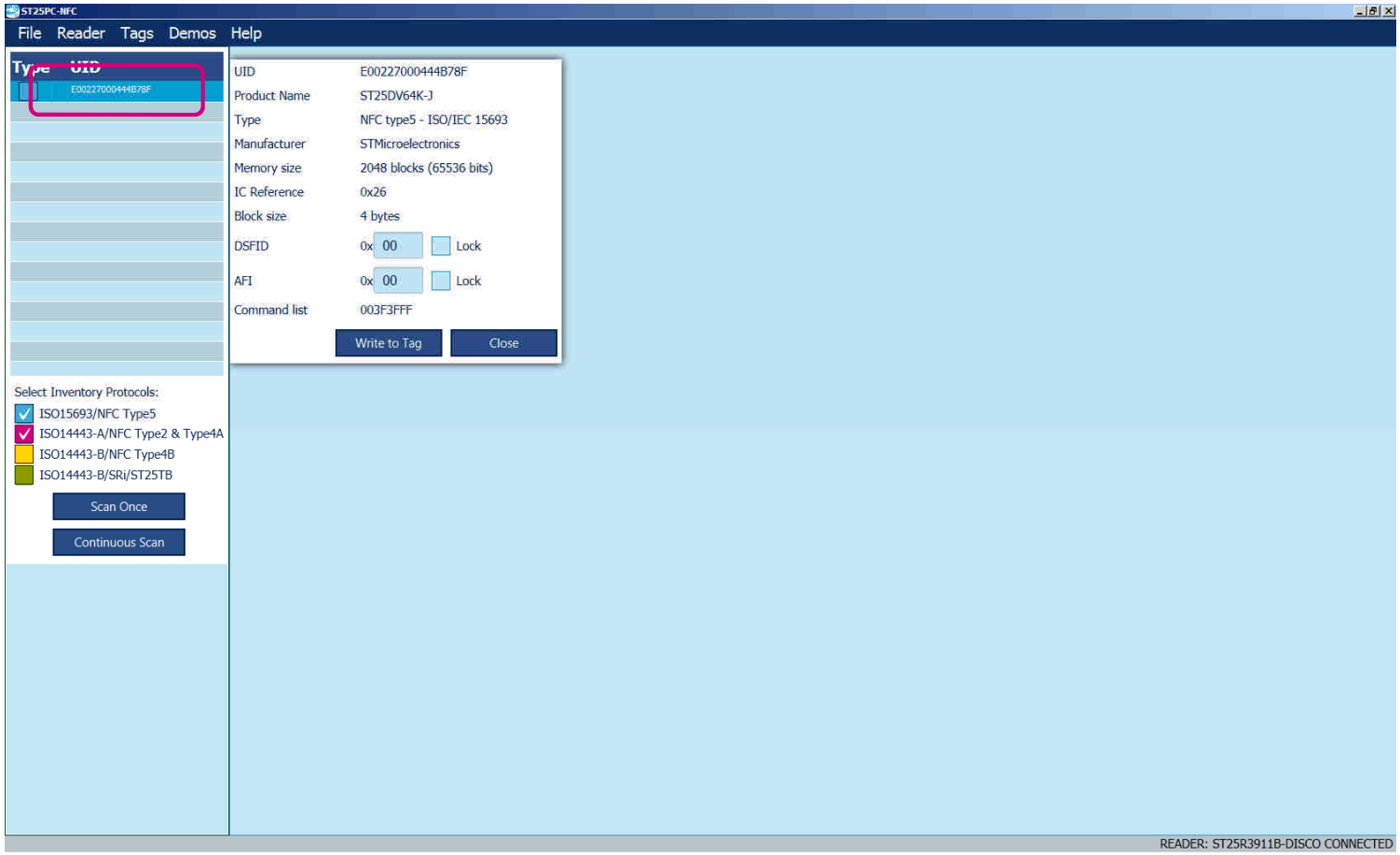


Correct LED patterns on the ST25R3911B



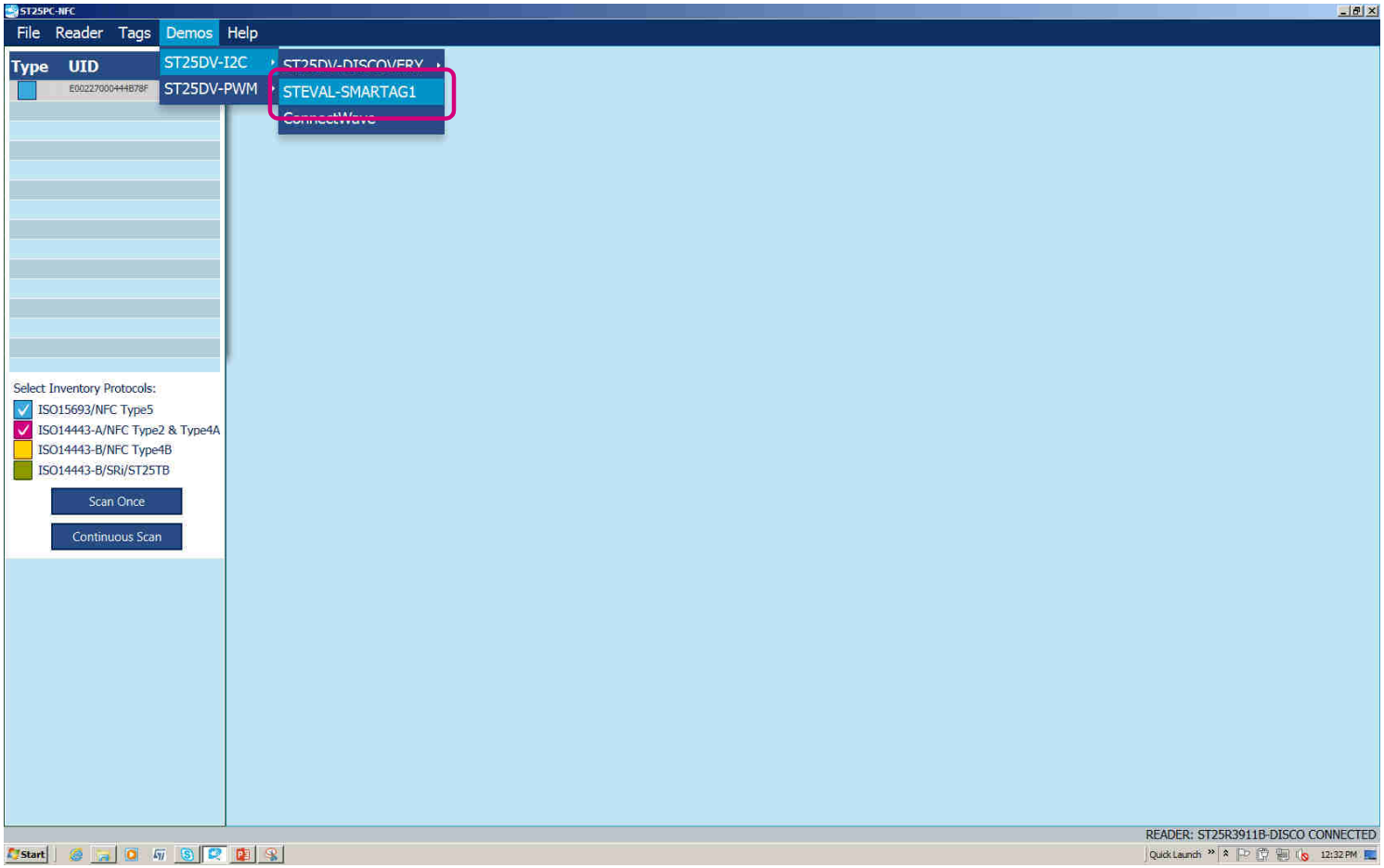
life.augmented

View tag UID, memory size and more



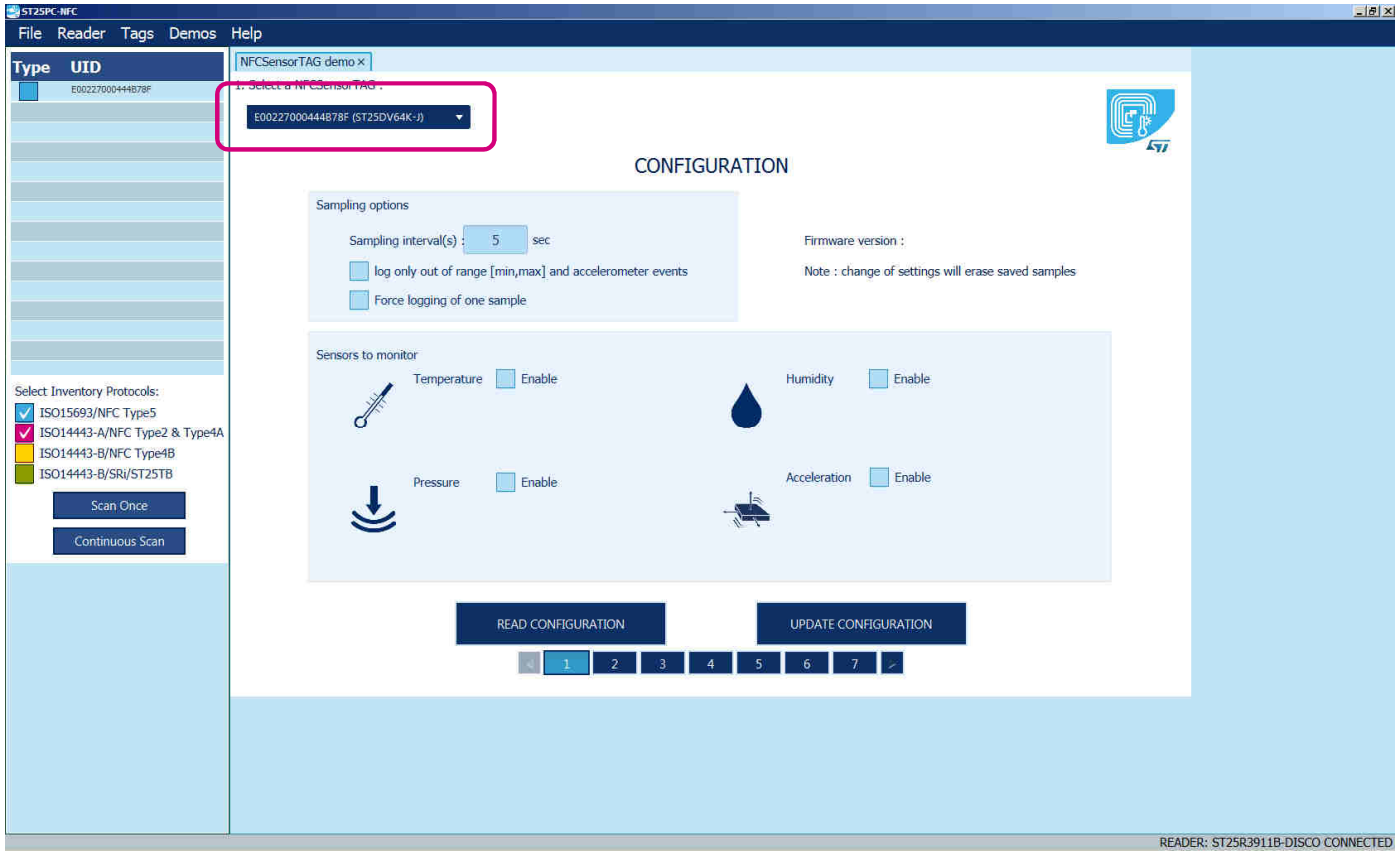
life.augmented

Running the sensor tag demo



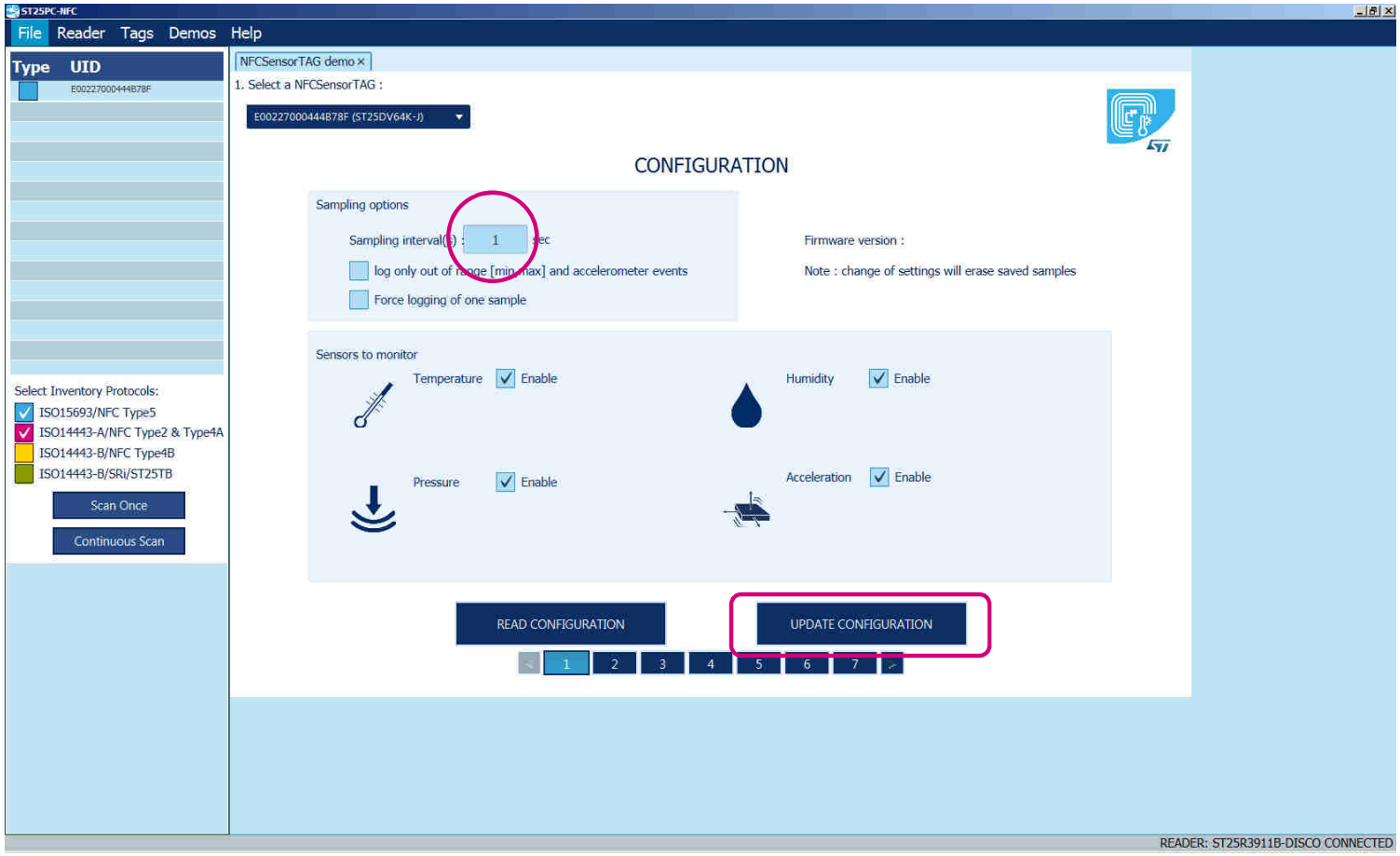
life.augmented

Sensor default configuration



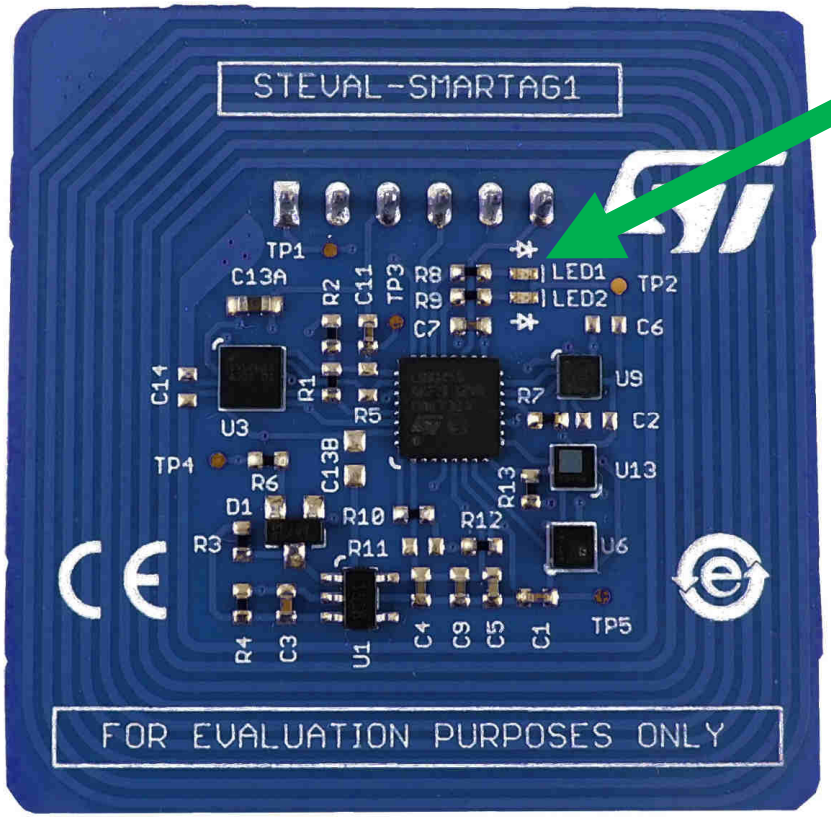
life.augmented

Setting new configuration



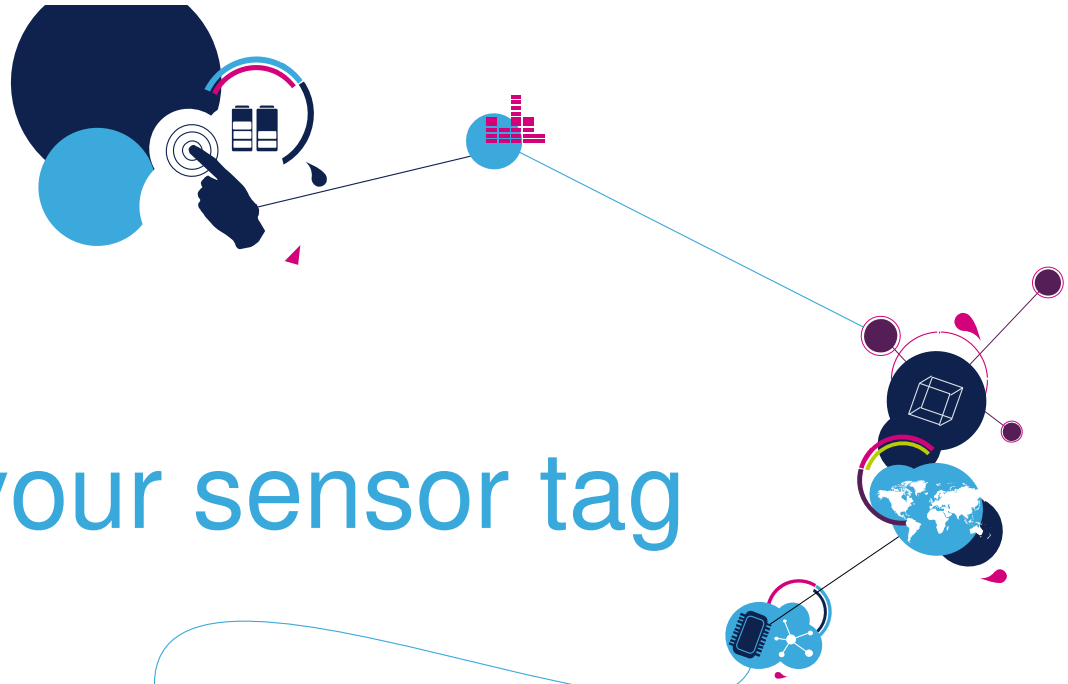
life.augmented

Check if your successful!



- Remove the tag from the reader RF field
- LED1 now should blink every 1 seconds

life.augmented

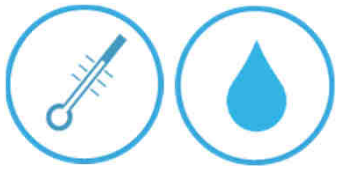


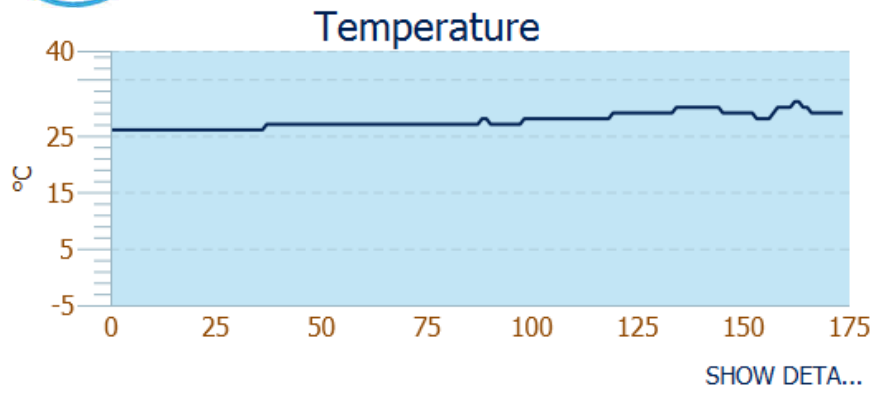
Time to exercise your sensor tag

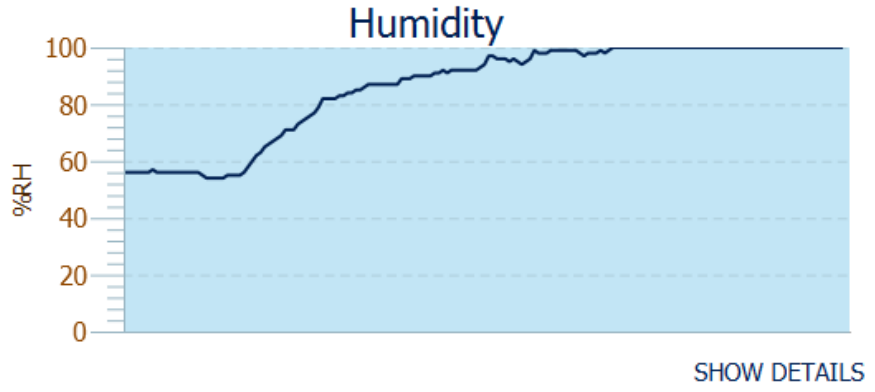
life.augmented

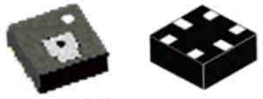
HTS221

Relative Humidity and Temperature combo









HLGA-6L
(2.0 x 2.0 x 0.9mm)

Humidity

- Range: 0%RH : 100%RH
- Accuracy: ±3.5%RH

Temperature

- Range: -40°C : +125°C
- Accuracy: ± 0.5°C, from 15°C to +40°C

Advantages

- High Accuracy
- Low power consumption 2 µA
- Extended operative supply voltage

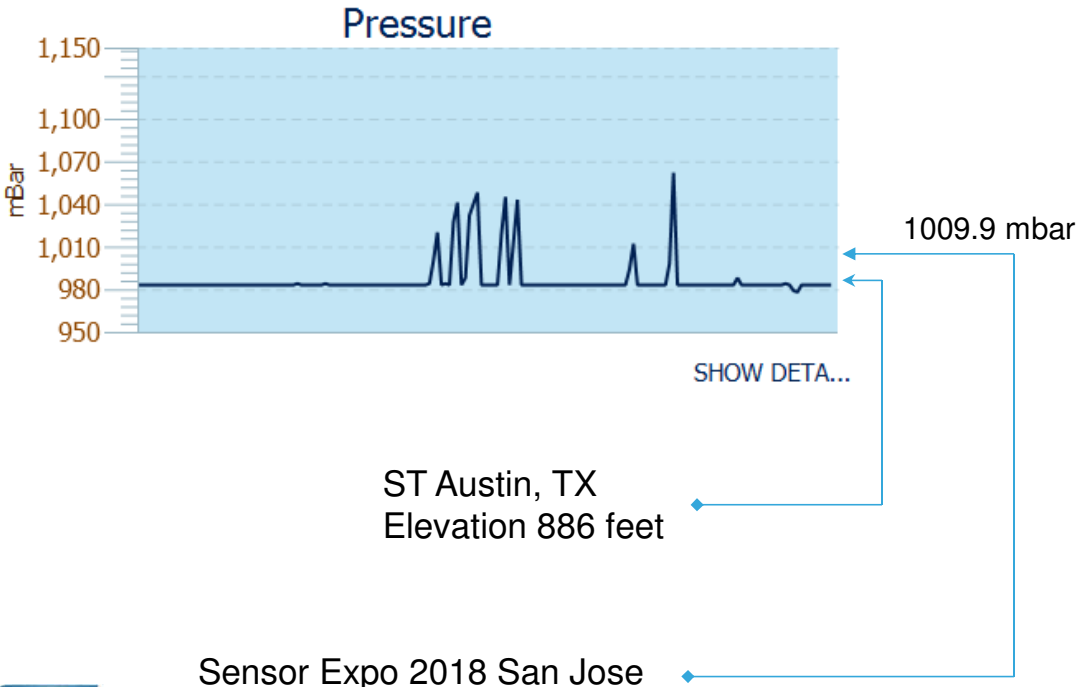
life.augmented

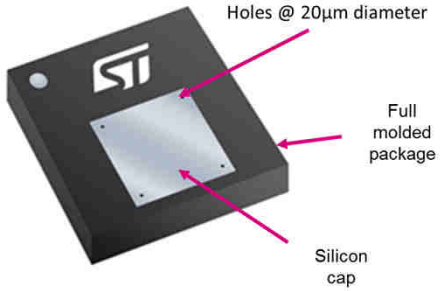
Changing the humidity and temperature

- Blowing on the sensor will increase the humidity at point of measurement.
- Put the sensor tag into the bag and inflating with your breath will put it in a high humidity environment.
- Please don’t put water on the tag because it will short out the circuit.
- Warm your hand up against a cup of coffee and keep the tag in your hand to log higher temperature

LPS22HB

Pressure Barometer/Altimeter Sensor





Pressure

- Range 260-1260 hPa
- Relative accuracy of pressure measurement:
  - < 10 µbar
  - 6cm resolution

life.augmented

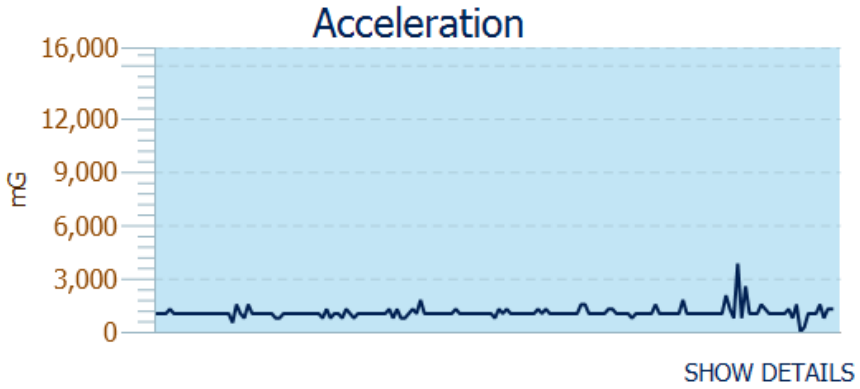
Changing the pressure

- Change pressure by inflating the Ziploc bag

- Close the Ziploc bag and lightly press on the “air bag”. Try not to rupture the bag. Each time you do this you will increase the pressure.

LIS2DW12

3-axis Accelerometer



Package

2x2x0.7 mm

Accelerometer

- ±2/±4/±8/±16 g selectable Full Scales
- Noise/accuracy level from 12 to 14bit resolution
- Low current consumption

380nA LP Mode and 40nA Standby

life.augmented

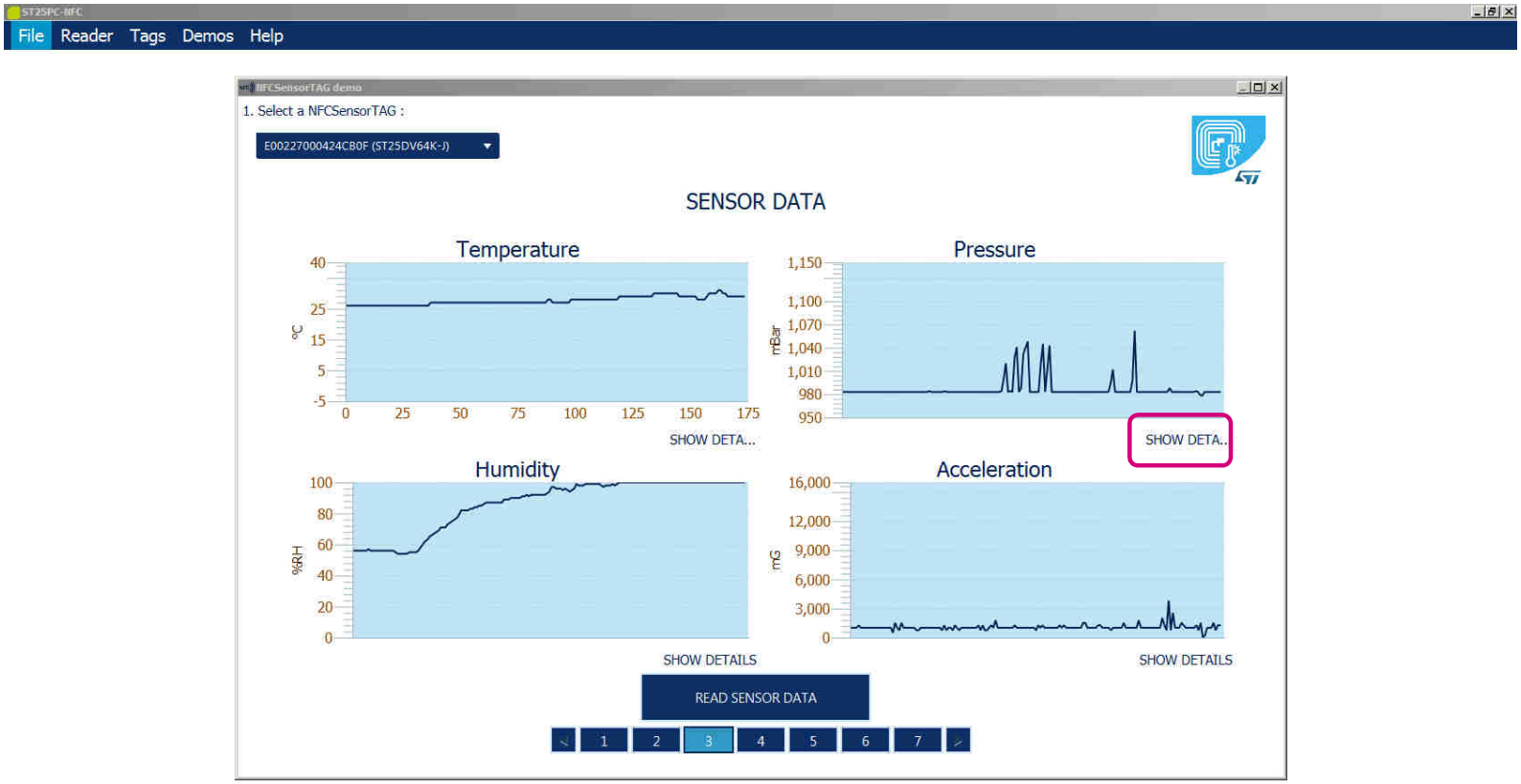
Shock and Impact

- Register the following action and make note of it.
- Drop onto a carpeted floor
- A tap of your finger against the body of the tag
- Please don’t throw the sensor tag against a hard surface as components might come loose.

Let’s view your data!

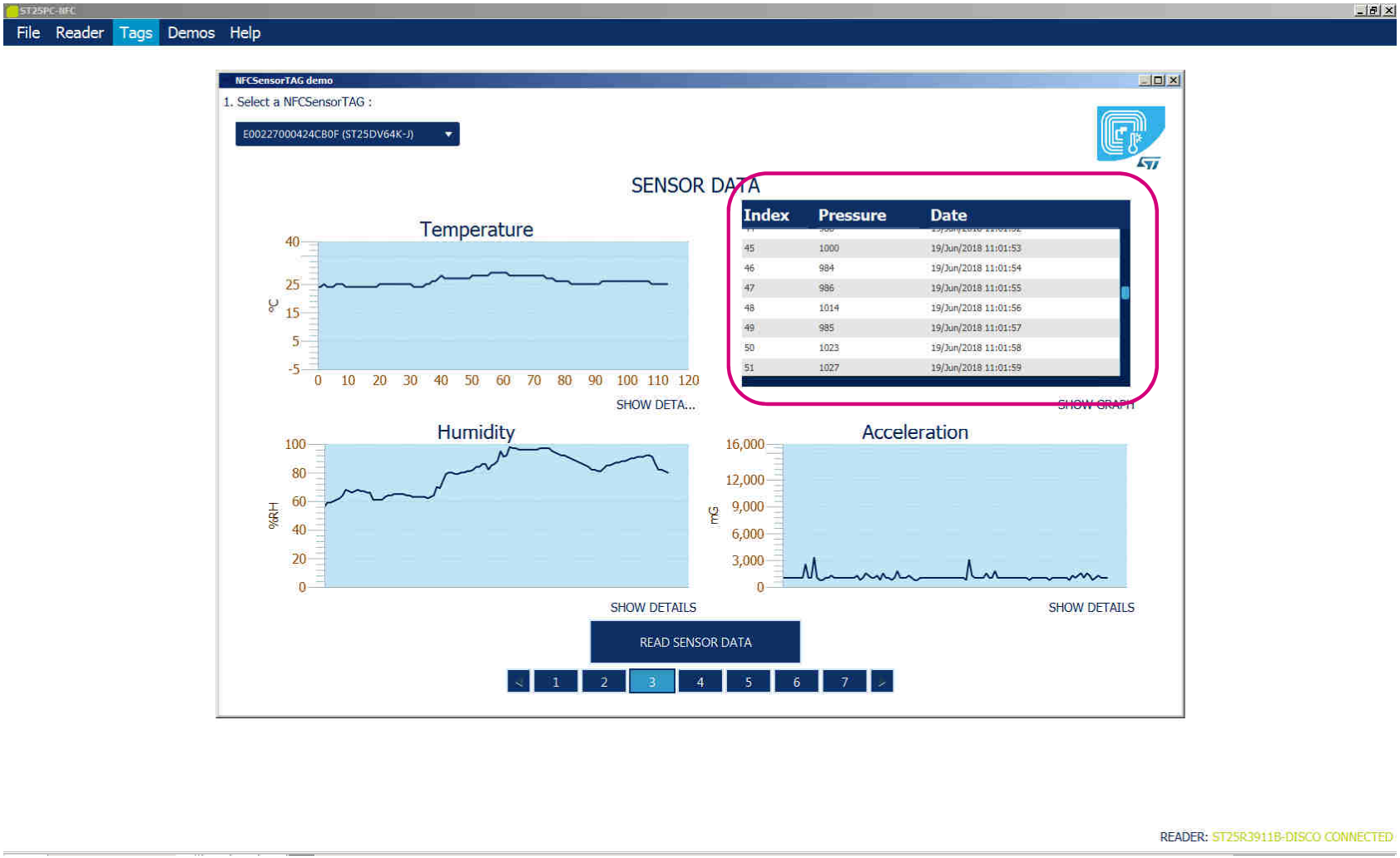
life.augmented

Show details

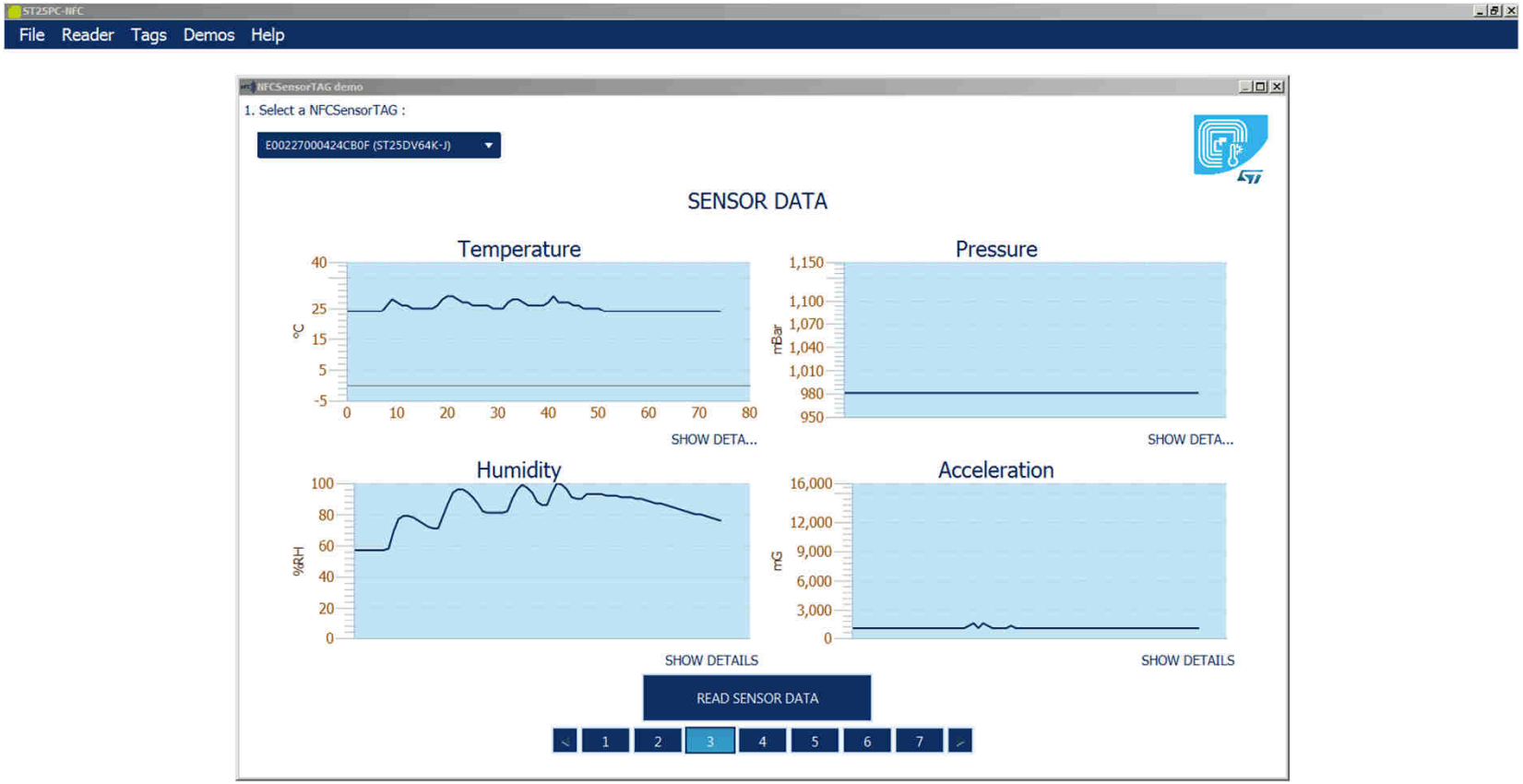


life.augmented

Show Details Panel



My sample data



Process of data logging on the tag in Battery Assisted Mode


```
/* Receive one interrupt from Timer */
if((RFActivityStatus==FIELD_FALLING) | (ForceStart==1)) {
  if(ForceStart) {
    RFActivityStatus=FIELD_FALLING;
    ForceStart=0;
  }

  if( (ReadSensorAndLog & SYNC_EVENT ) ||
      (ReadSensorAndLog & ASYNC_EVENT) ) {
    if(NFCStatus == NFC_STATUS_OFF) {
      PowerOnNFC();
      /* rise time required by VDD_EEPROM for NFC */
      HAL_Delay(200);
      NFCStatus = NFC_STATUS_ON;
    }

    SmarTag_LED_GREEN_On();

    if (NFC_EEPROM_Data.LogMode == SMARTAG_LOGMODE_INACTIVE) {
      /* Do Nothing */
      goto SMARTAG_SLEEP;
    }

    if (!NFC_EEPROM_Data.EnableFlags) {
      /* Do Nothing */
      goto SMARTAG_SLEEP;
    }

    /* beginning of Active log */
    if(ReadSensorAndLog & SYNC_EVENT) {
      /* Init SmarTag sensor */
      if(NFC_EEPROM_Data.LogMode != SMARTAG_LOGMODE_ACTIVE_THS) {
        InitSmarTagSensor();
```

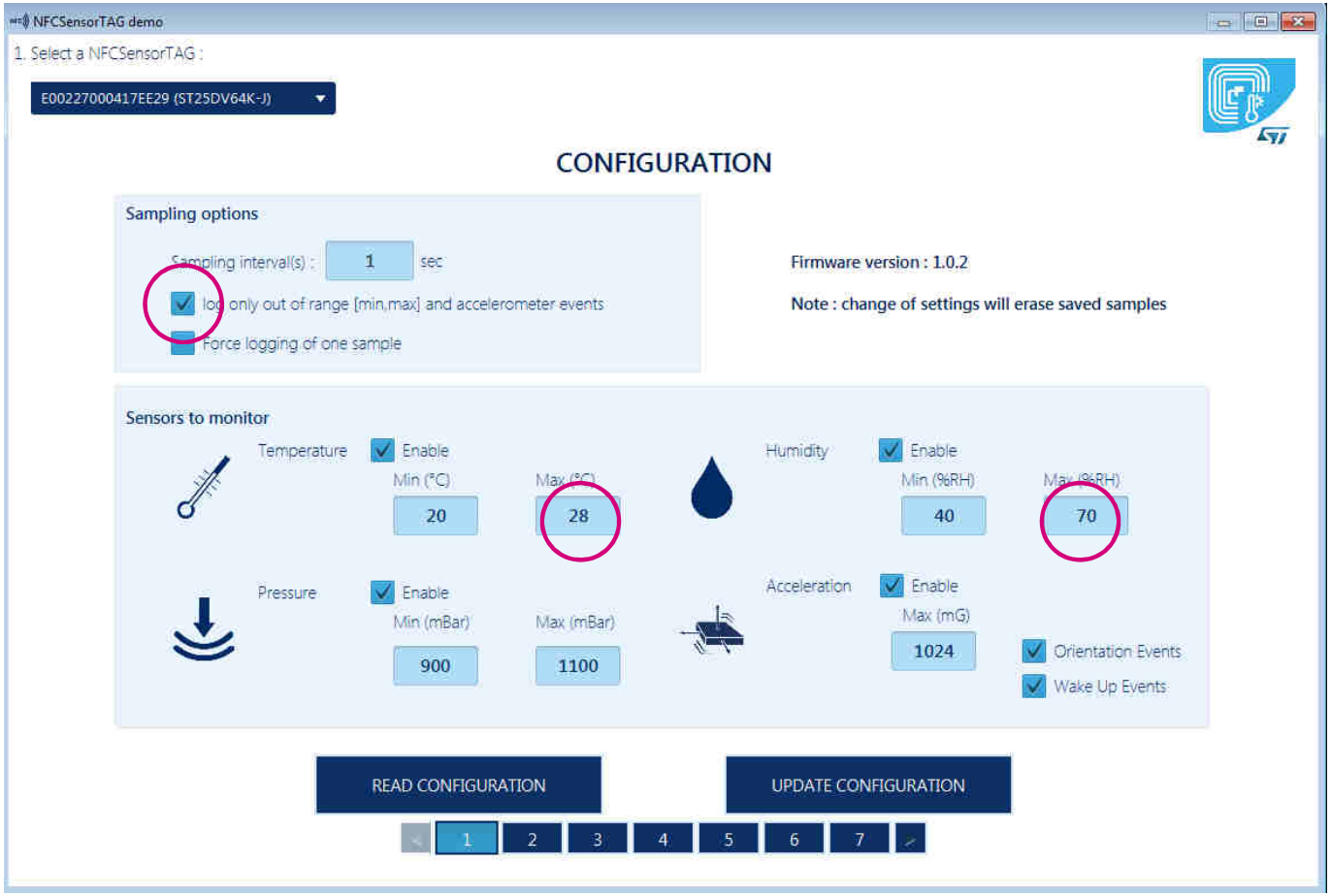
If RF field is detected, don't data log. Read in Progress

Turn on LED1

Begin the logging process and subsequently writing to the memory

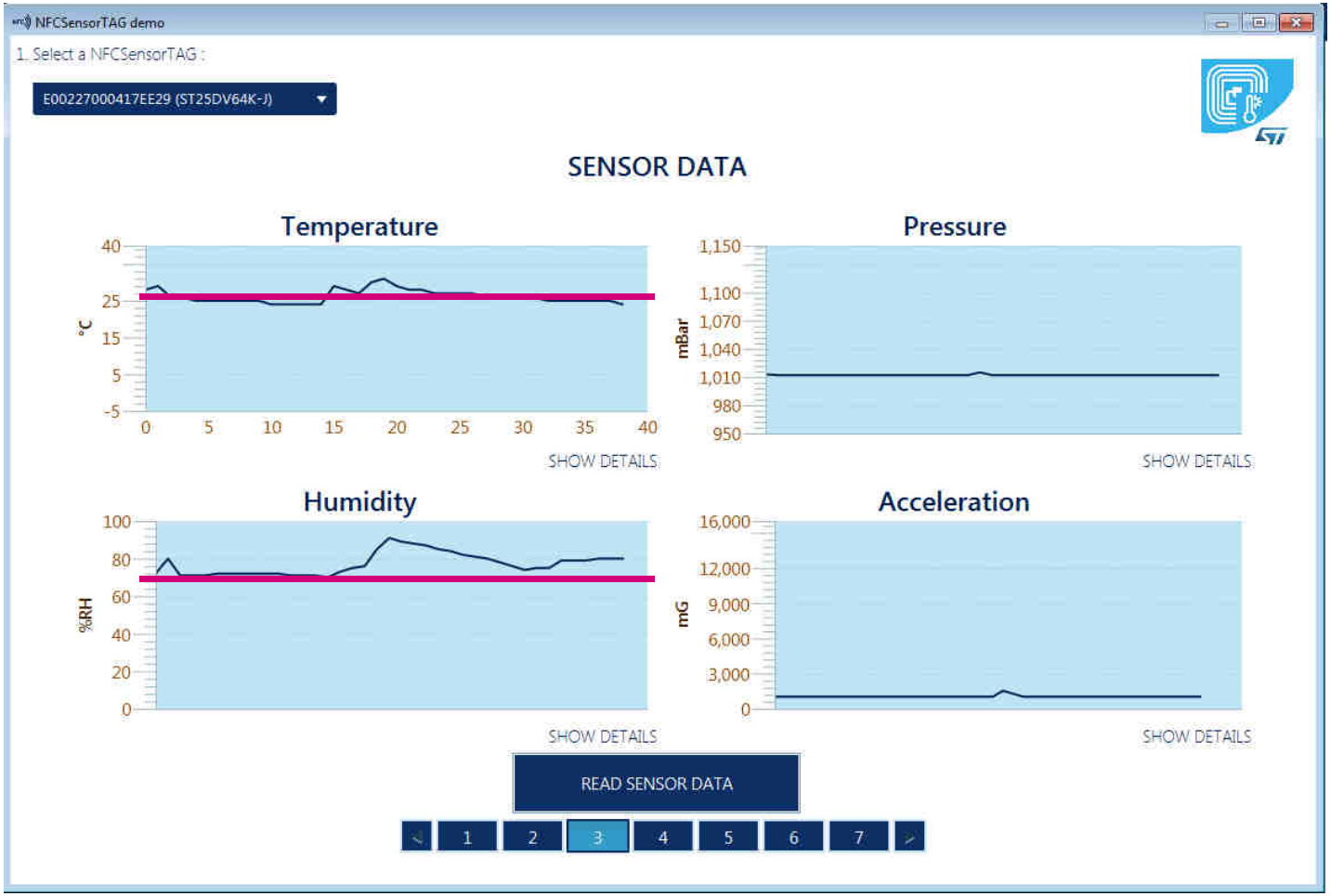
life.augmented

Setting min-max in Temp and Humidity



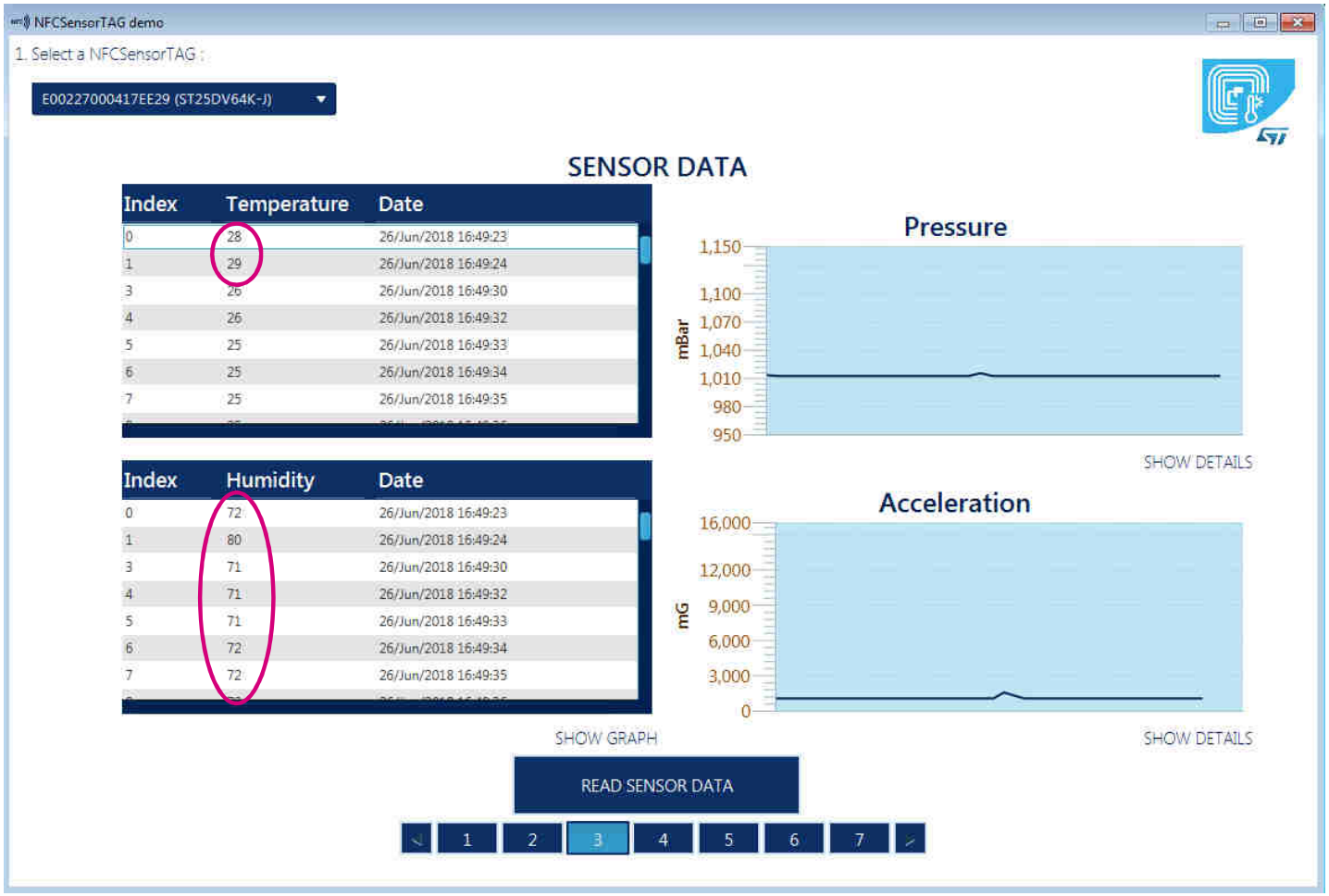
life.augmented

Data Log only when min-max values



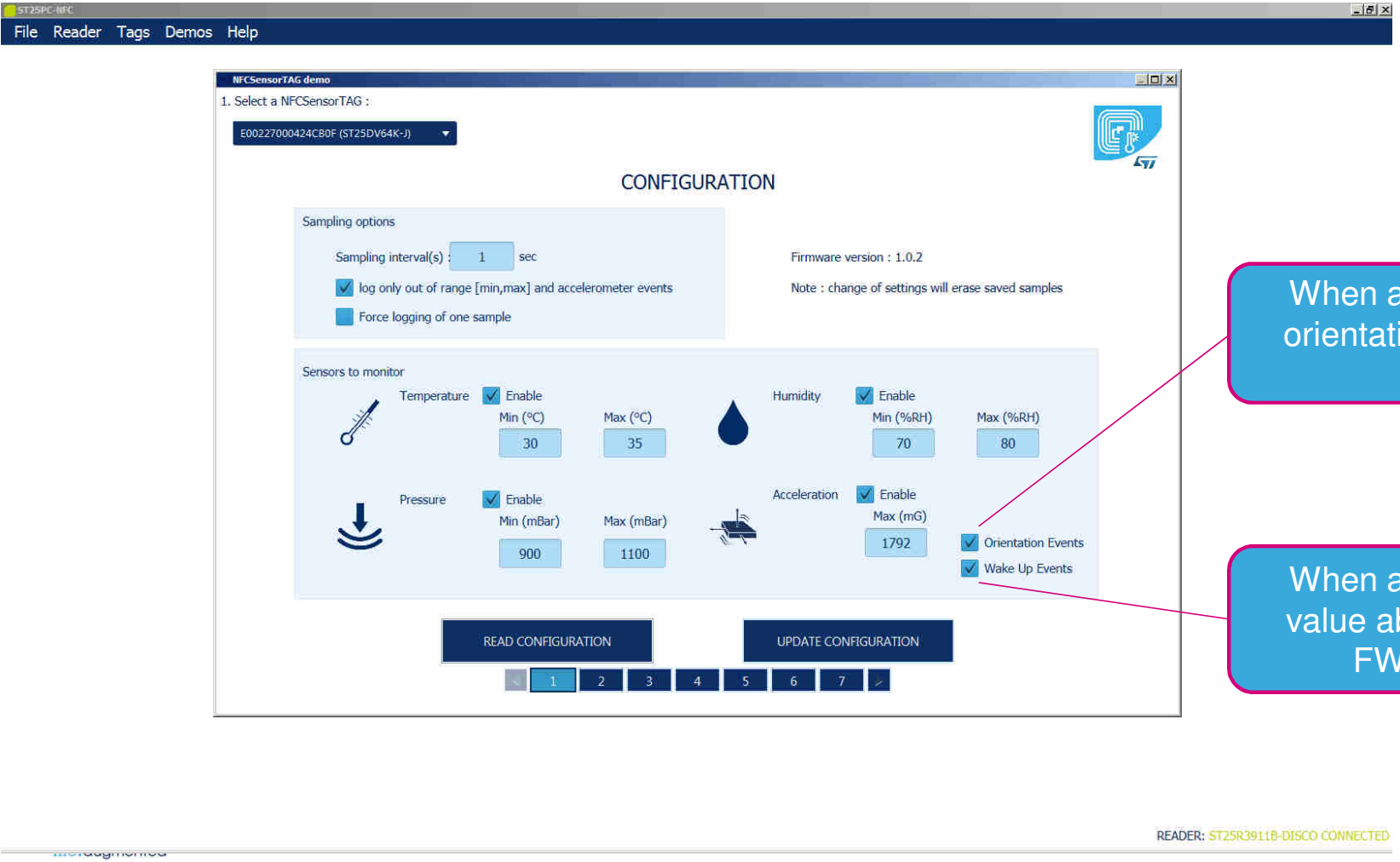
life.augmented

Data Log only when min-max values



life.augmented

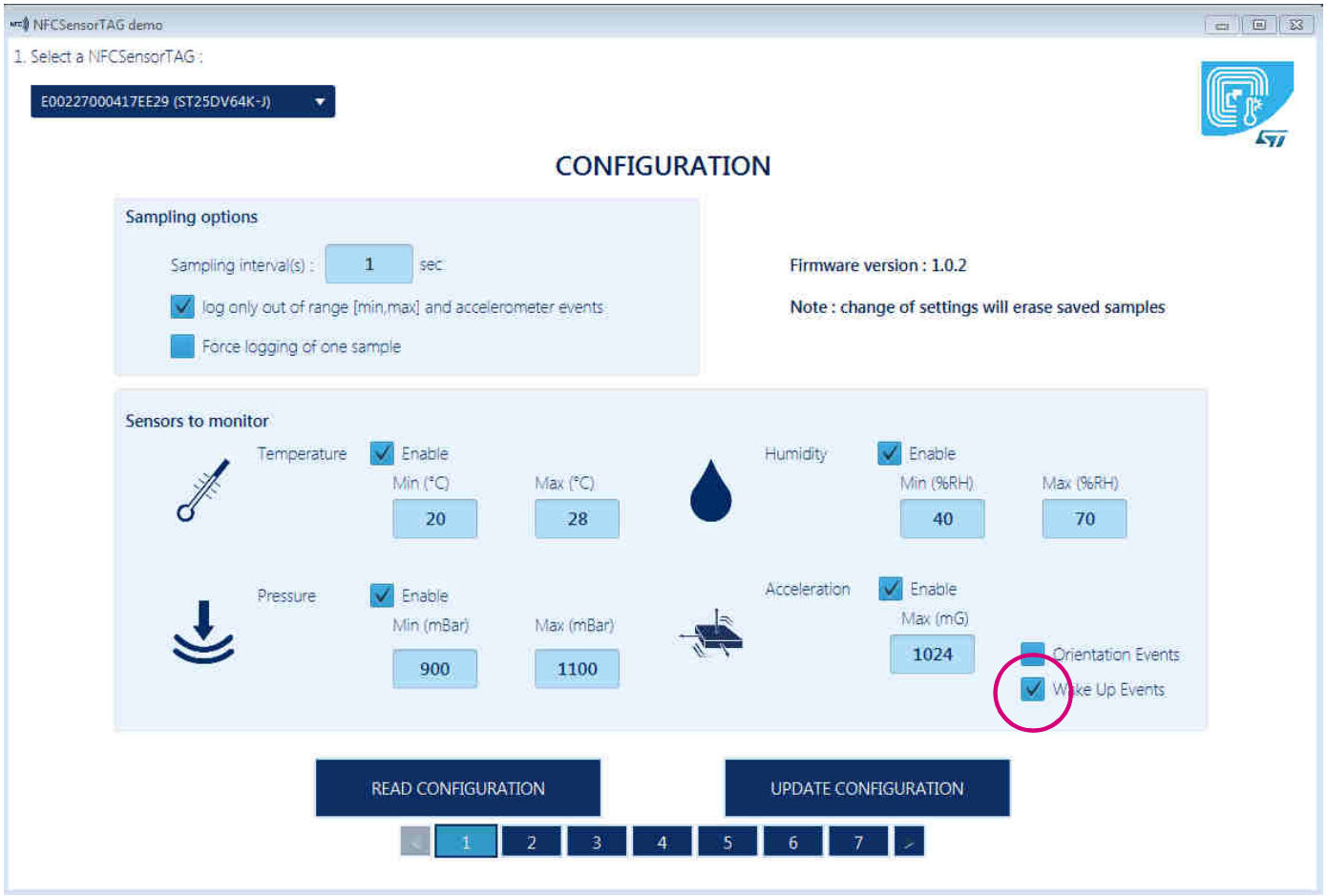
Event Driven Logging



When accelerometer detects orientation change, it logs the data.

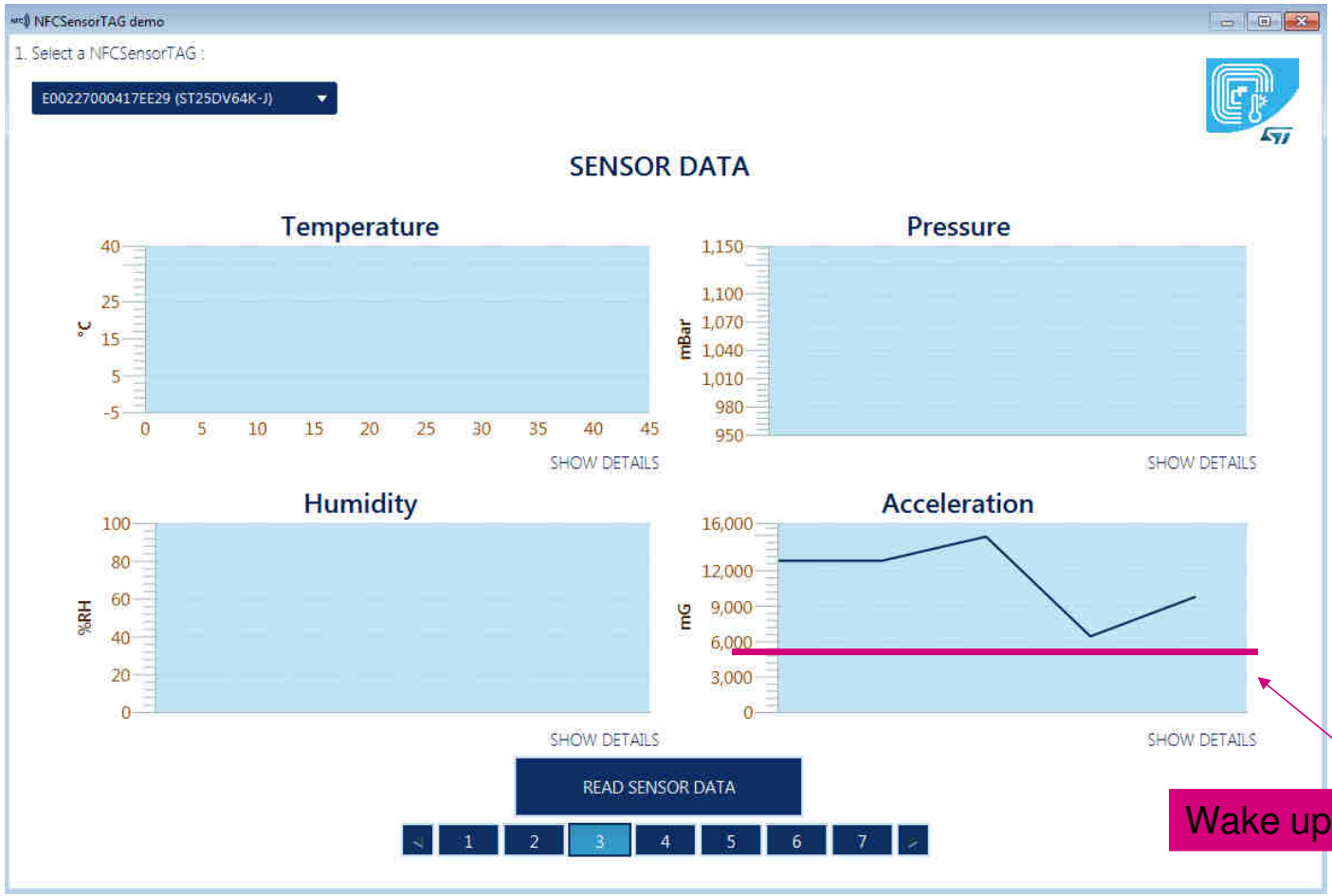
When accelerometer detects value above 4G (value set by FW), it logs the data

Setting threshold event for single sensor



life.augmented

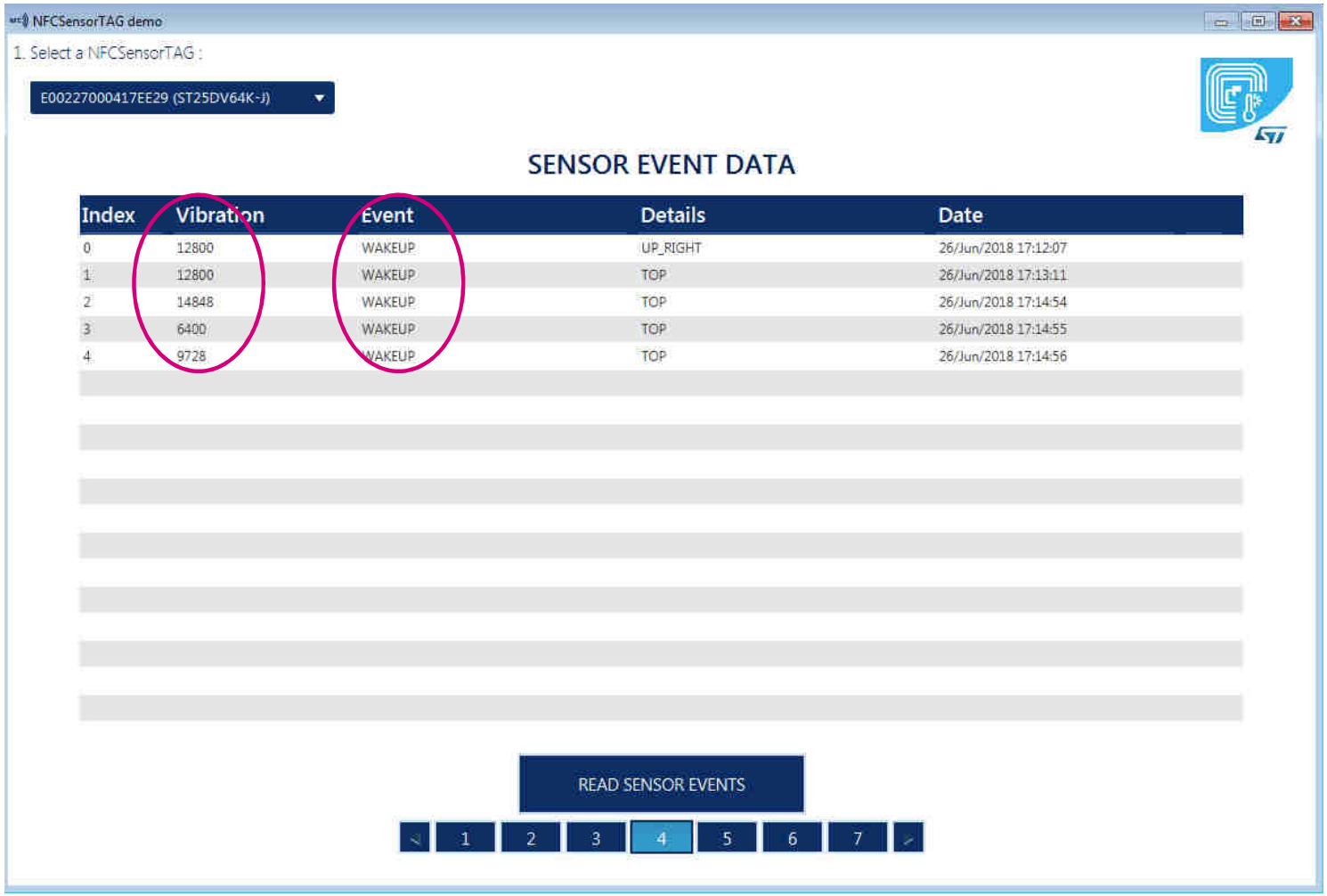
Setting threshold event for single sensor



Wake up detection (>4G)

life.augmented

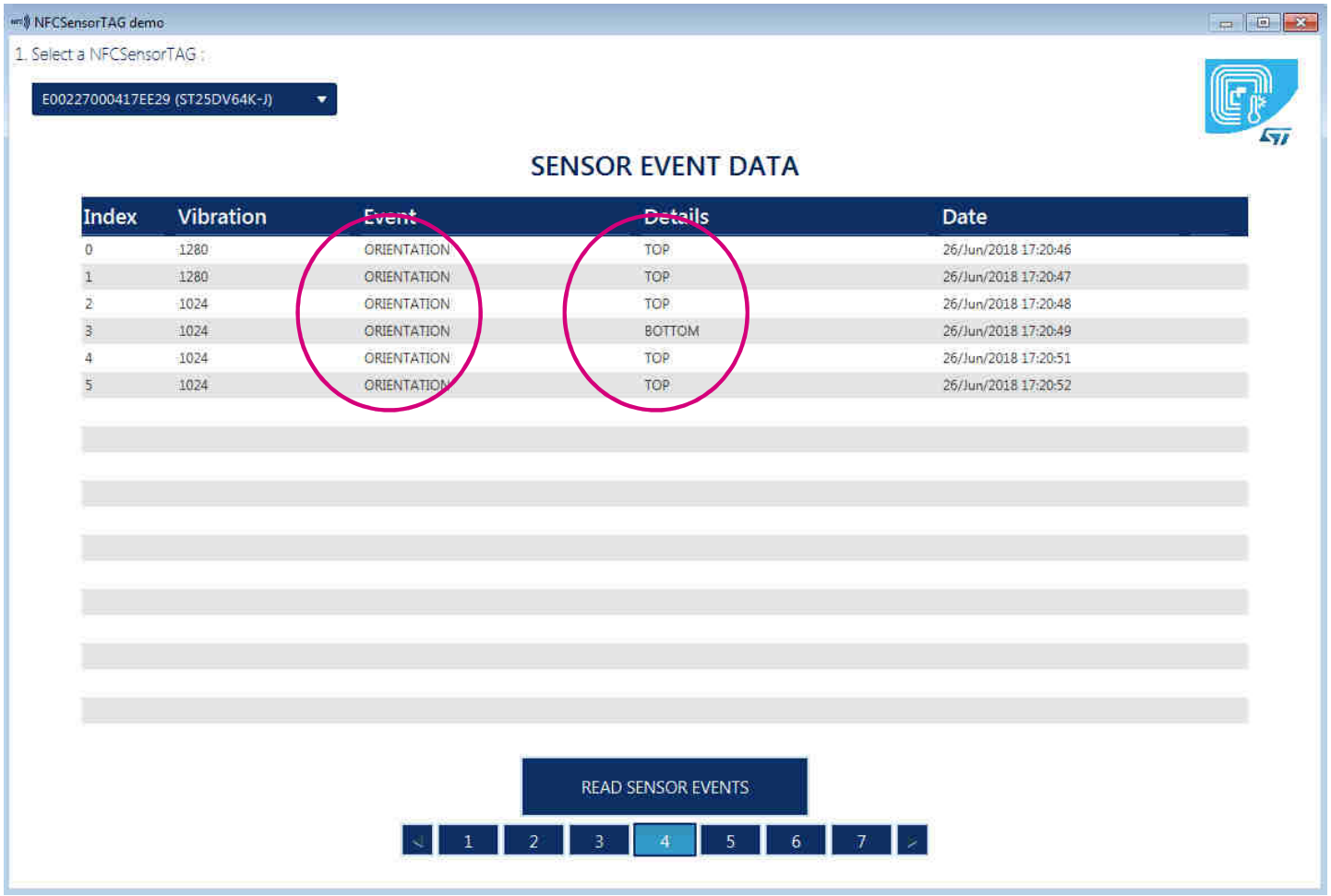
Setting threshold event for single sensor



Setting threshold event for single sensor

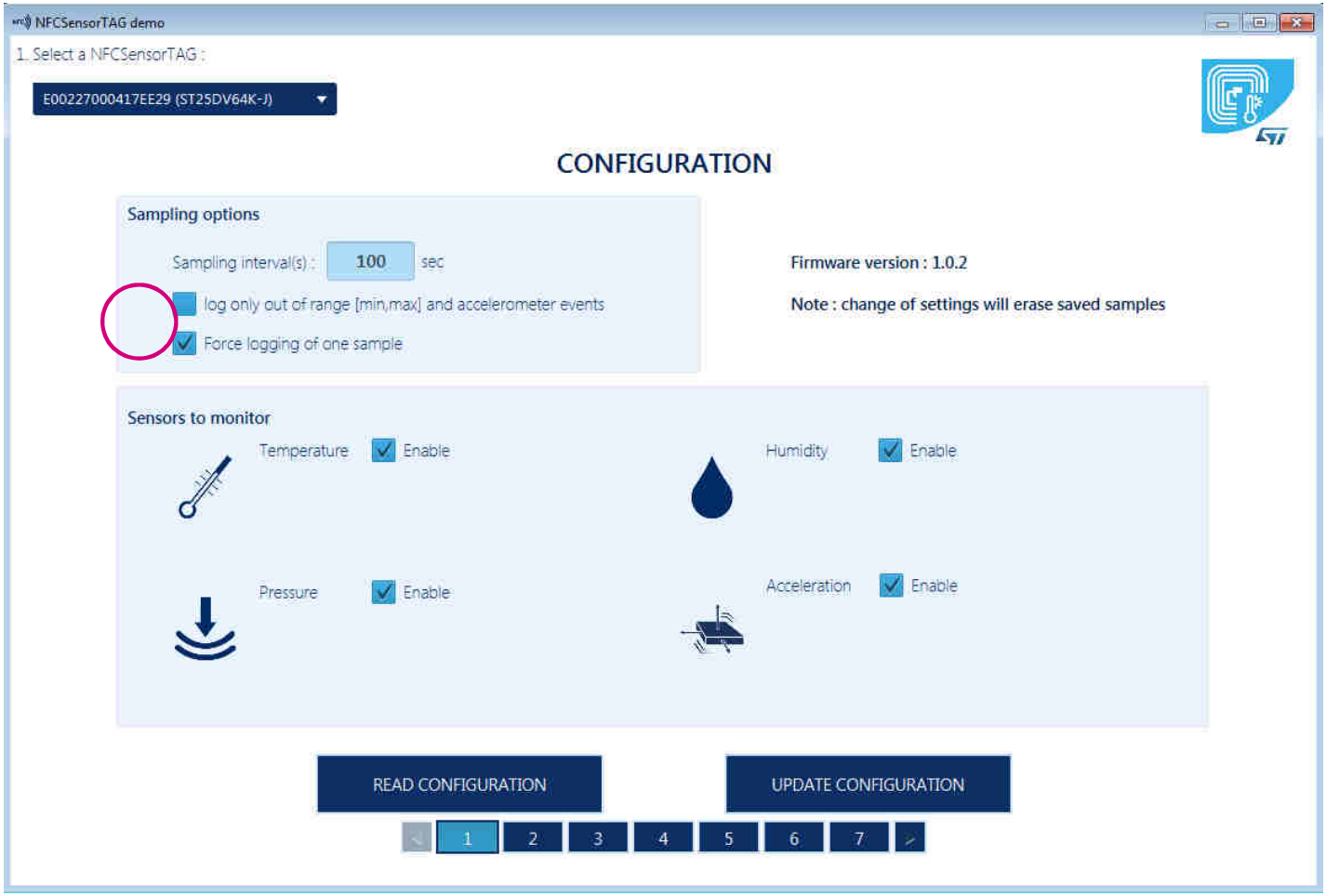
life.augmented

Setting threshold event for single sensor



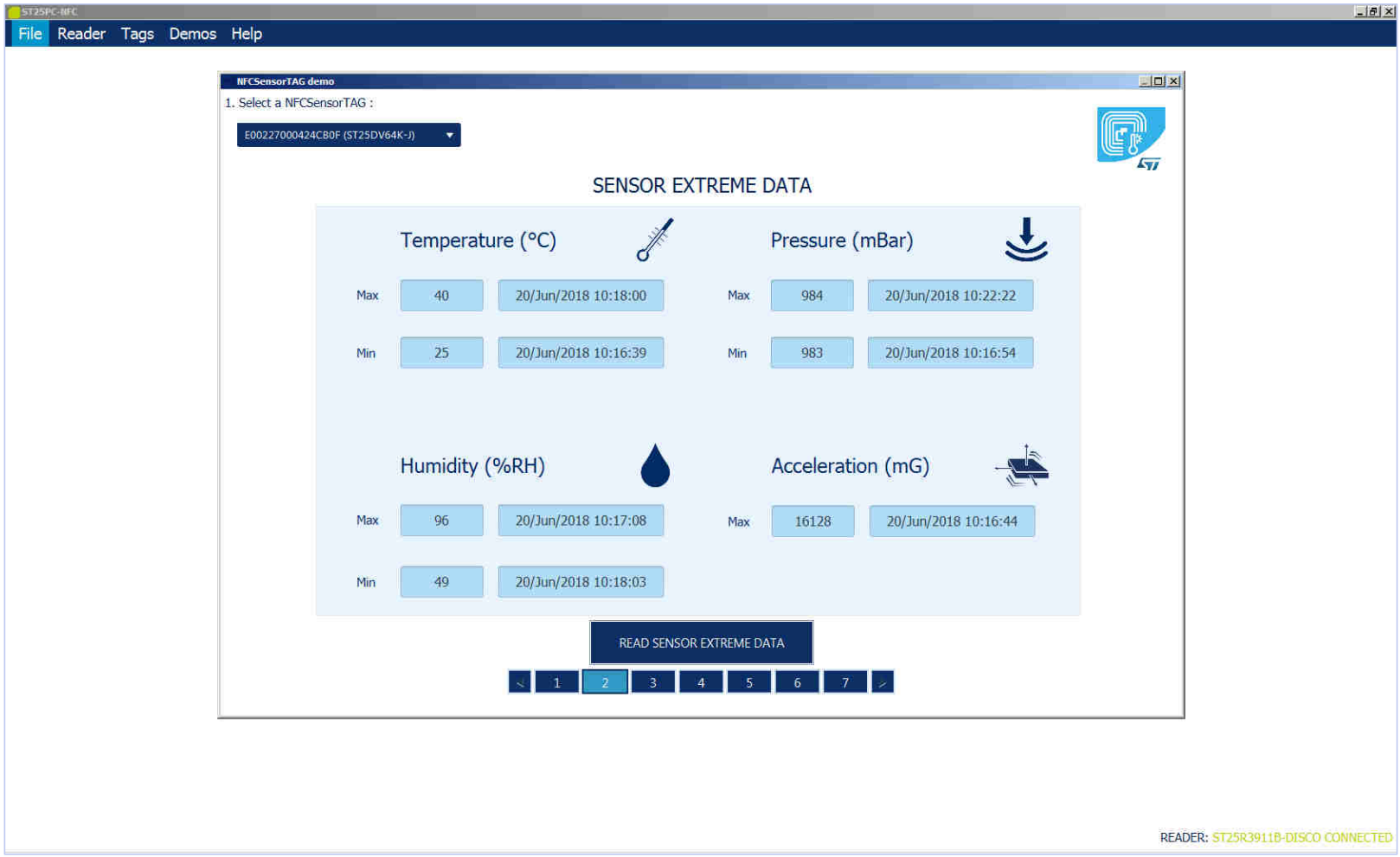
life.augmented

Setting threshold event for single sensor



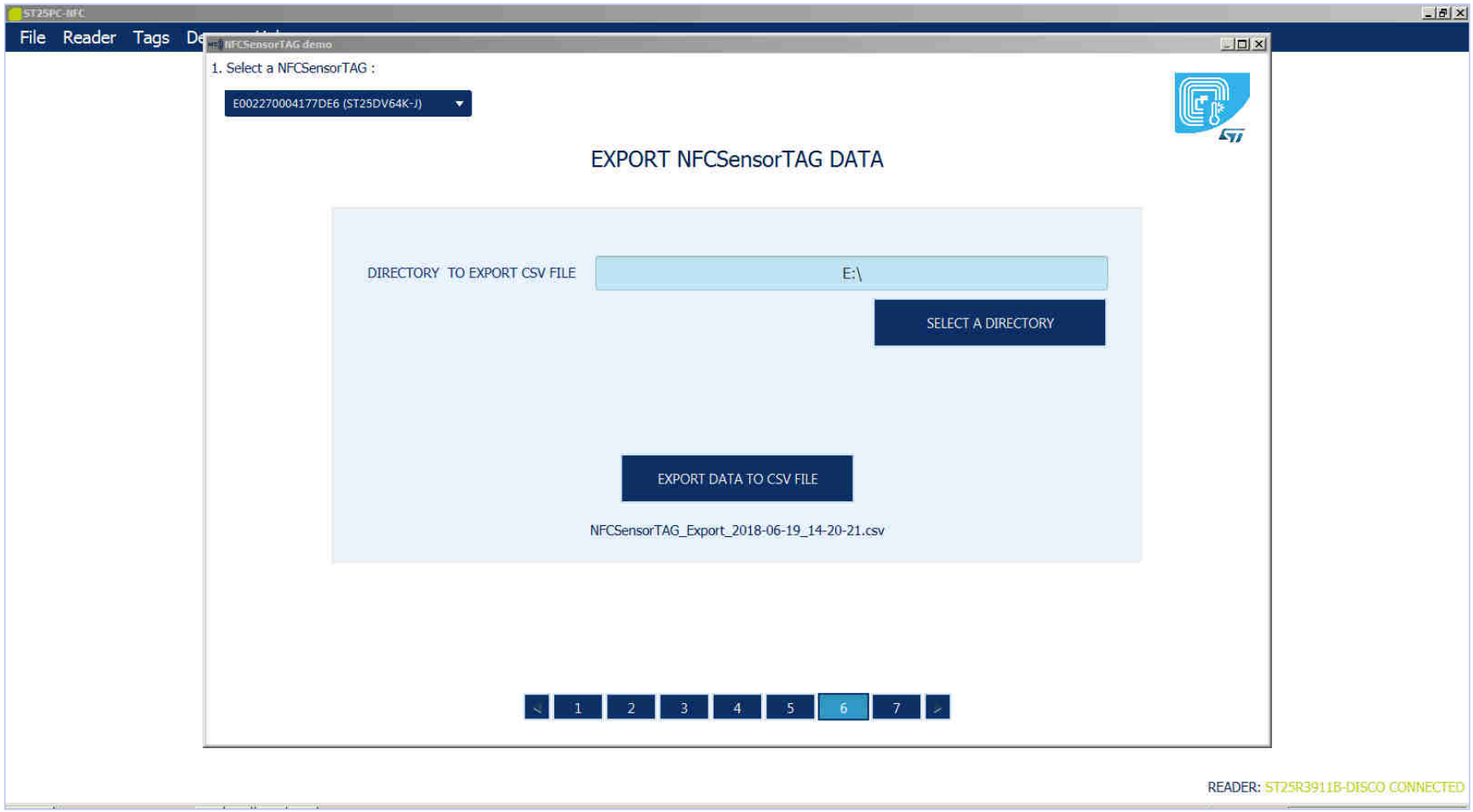
life.augmented

Viewing sensor extreme data



life.augmented

Exporting Sensor Data

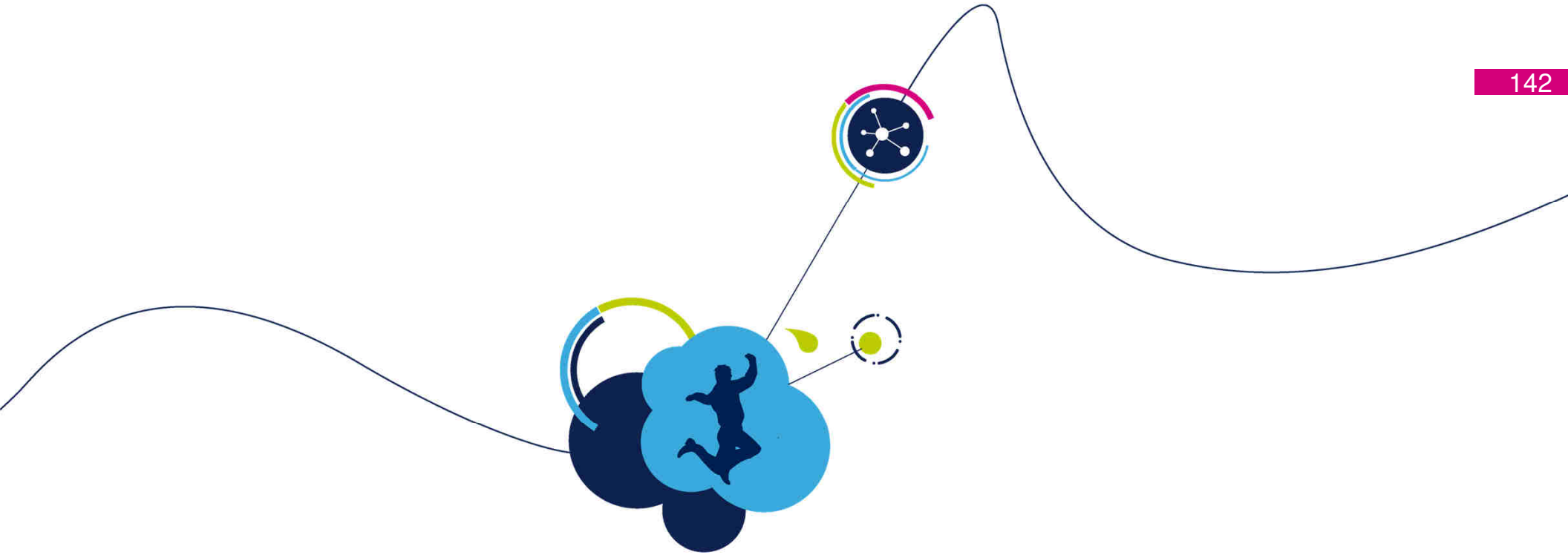


life.augmented

Using exported data

	A	B
1	Sampling Interval;1;seconds	
2	Temperature Enabled;Yes	
3	Humidity Enabled;Yes	
4	Pressure Enabled;Yes	
5	Acceleration Enabled;Yes	
6		
7		
8	Threshold;Min;Max	
9	Temperature;24;25	
10	Humidity;40;50	
11	Pressure;900;1100	
12	Acceleration;NaN;1024	
13		
14		
15	Data Log	
16	Date;Temperature (°C);Humidity (%RH);Pressure (mBar);Acceleration (mG)	
17	19/Jun/2018 14:17:01;30;52;974;768	
18	19/Jun/2018 14:17:02;30;53;974;1024	
19	19/Jun/2018 14:17:03;30;53;974;1024	
20	19/Jun/2018 14:17:04;30;53;974;1024	
21	19/Jun/2018 14:17:05;30;53;974;1024	
22	19/Jun/2018 14:17:06;30;53;974;1024	
23	19/Jun/2018 14:17:07;30;53;974;1024	

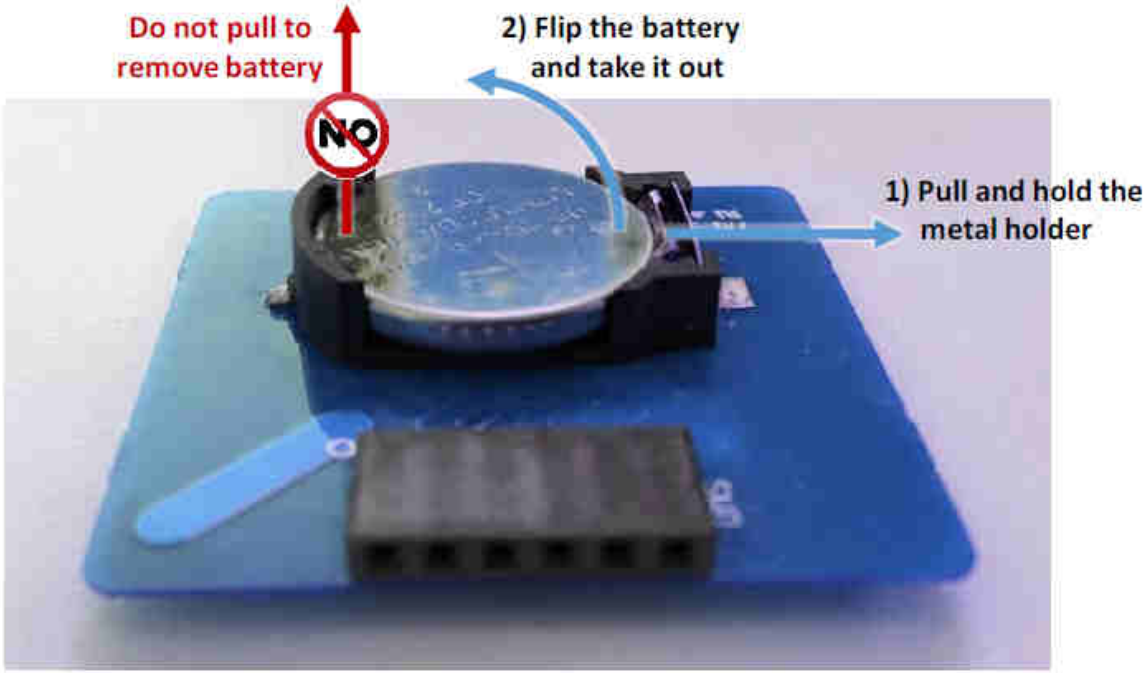
NFCSensorTAG_Export_2018-06-19_



Battery-less operation

life.augmented

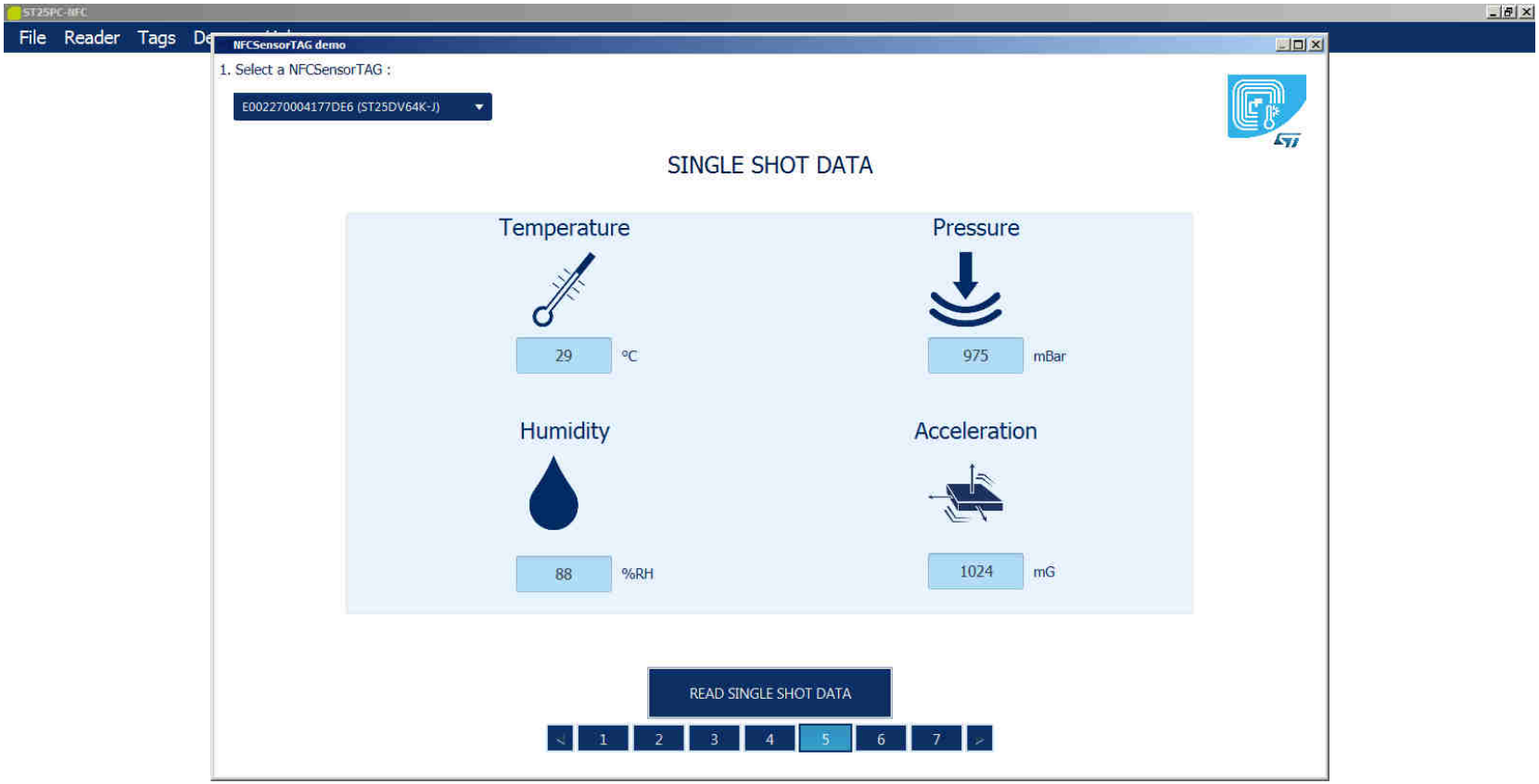
Correct way to remove coin cell battery



Exercise

- Remove the battery from your sensor tag
- Set for single shot mode from the PC Application Tab #5
- Put the tag in the plastic bag and put it right on top of the reader
- If the placement is good, you will be able to read sensor data.

Single Shot Reading



Storing 1-shot data on the tag


```
/* Normal Start */
NfcType5_NDEFInitHeader();

/* Init Environment Variables */
InitEnvVariables();

/*********************************************/
/* Like default we will make the single shot */
/*********************************************/

/* Init SmarTag sensor */
InitSmarTagSensor();

MEMS_Sensors_ReadData();

/* De-Init SmarTag sensor */
DeInitSmarTagSensor();

SensorDataToCompactData();

OneShotWrite();
```

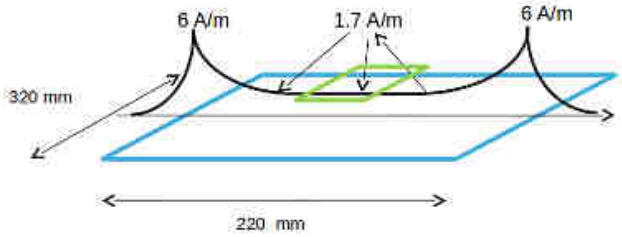
We initialized the ST25DV with an NDEF message header

We compact the sensor data

We store the data in the tag

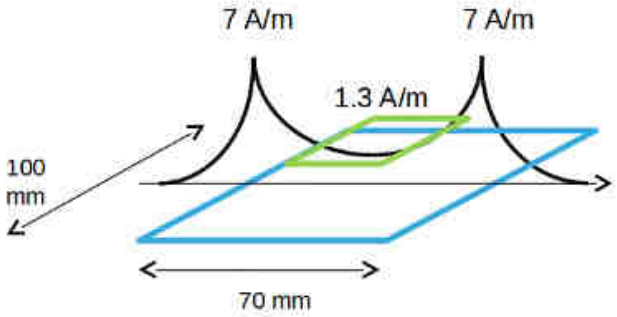
life.augmented

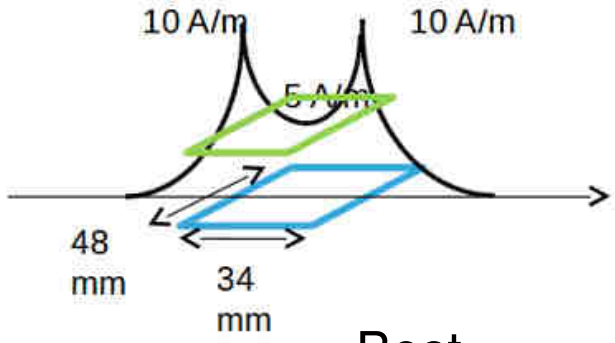
Tag Placement for Optimal EH



Good

Better





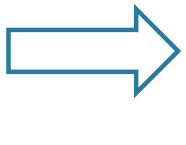
Best

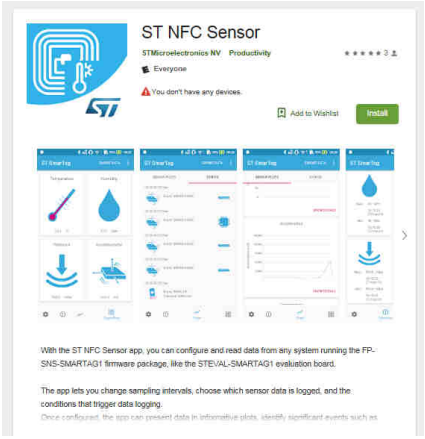
life.augmented

Optimized Architecture for Battery-less Application

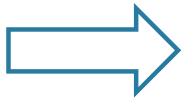
- Don’t need large EEPROM memory
- Use ST25DV 256bytes buffer (mail box)
- No issue with EEPROM endurance

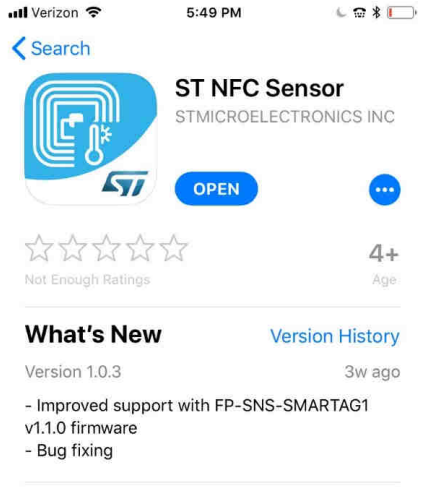
Smartphone APK





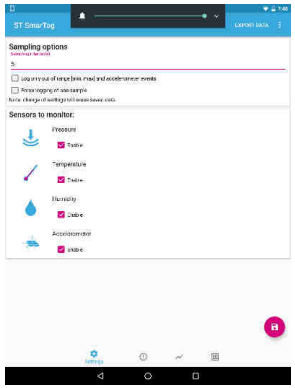
iOS





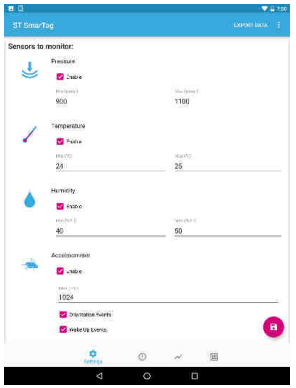
life.augmented

Android App

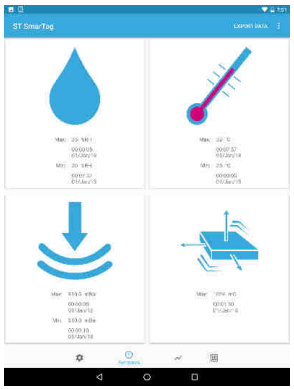


Continuous Log

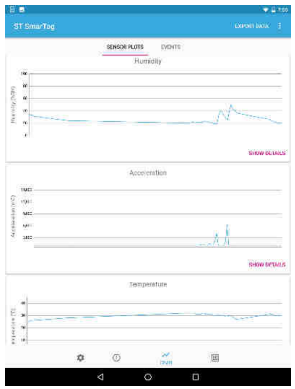
Out of Thresholds



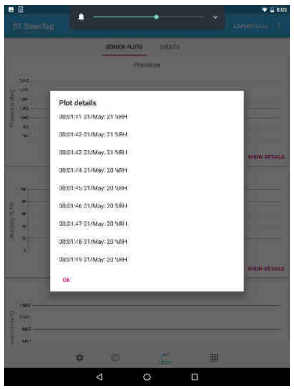
Min Max Rec



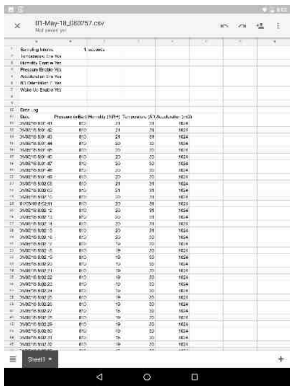
Data Plot



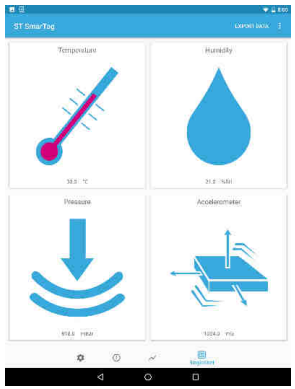
Plot Details



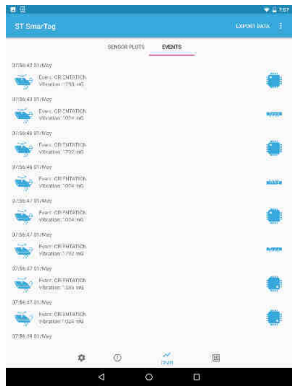
Data Export



One Shot EH



Event Logging



life.augmented

iOS support of NFC tag reader mode

- A new core NFC function of Apple iOS11 adds support for NFC tag reading to iPhone7 and iPhone7 Plus as well as the new iPhone8 and 8 Plus and iPhoneX

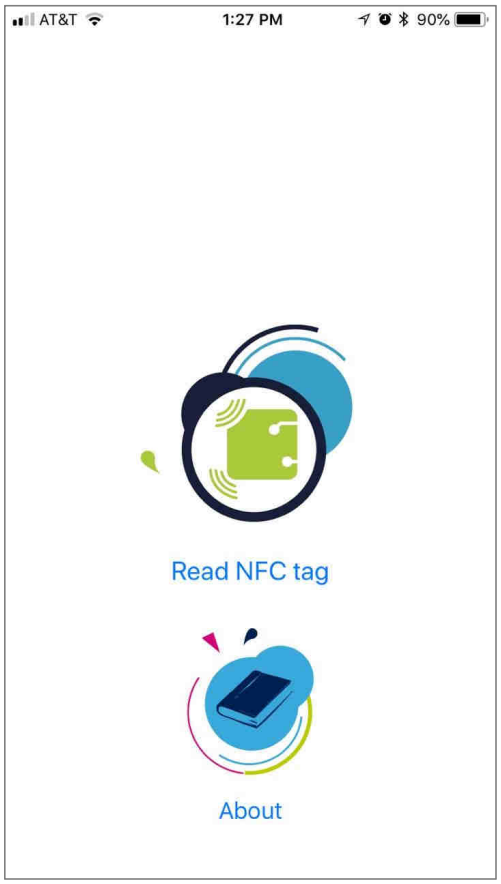
- iOS11 use cases
  - Read tags of **types 1 through 5** with NDEF (*)
  - Need iOS application (not «native» as Android)

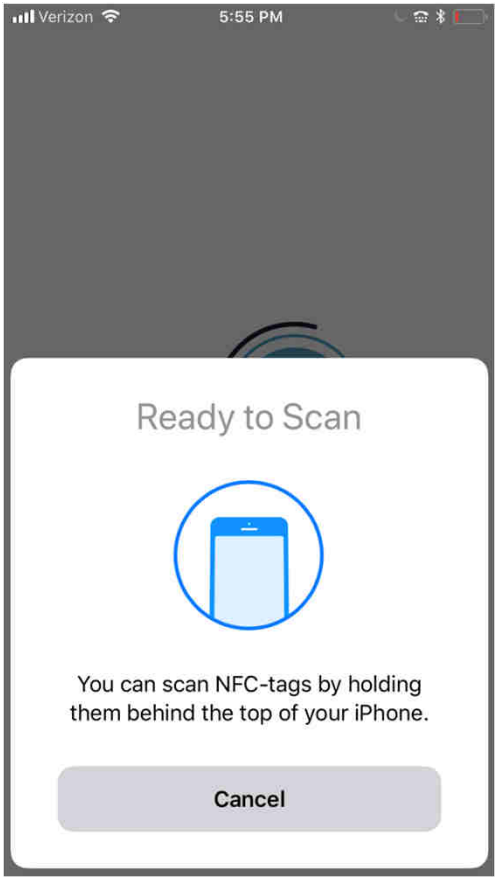
- Download the NFC Sensor Tag App on iTunes

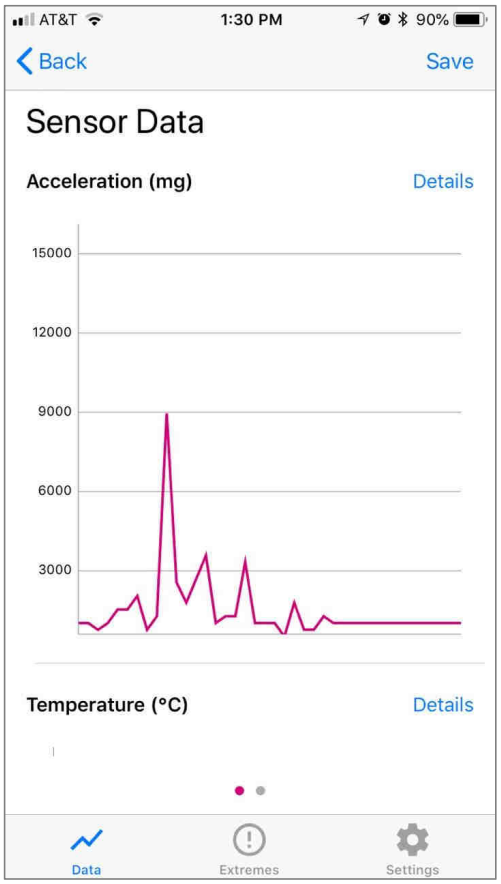
life.augmented

(*) NDEF: NFC Data Exchange Format

iPhone App





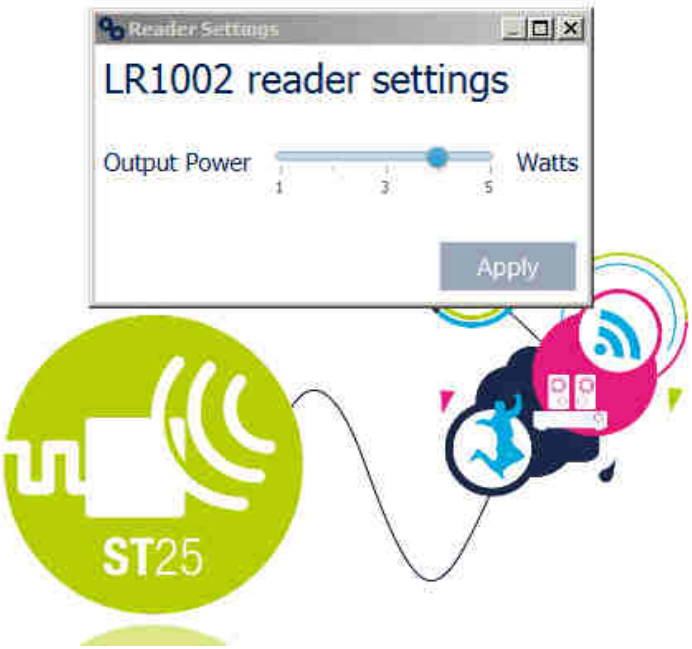


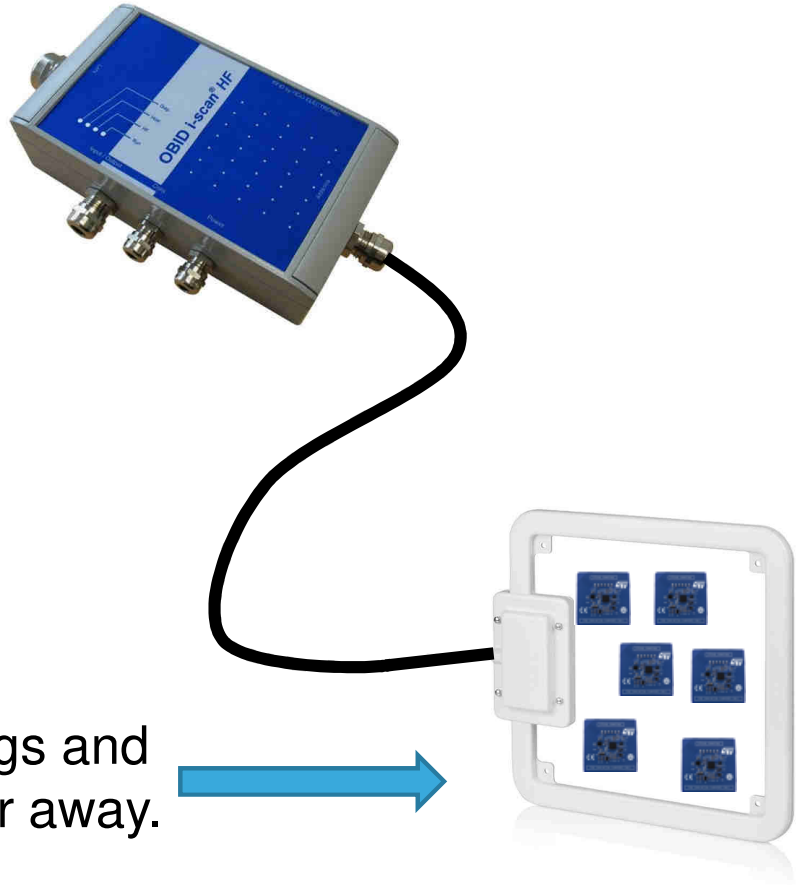
life.augmented

Iphone Capability

- iOS 11 is currently does not allow writing NDEF message. Only reading
- Cannot change default configuration of the sensor tag (e.g. logging time interval and thresholds)

Using more powerful NFC readers



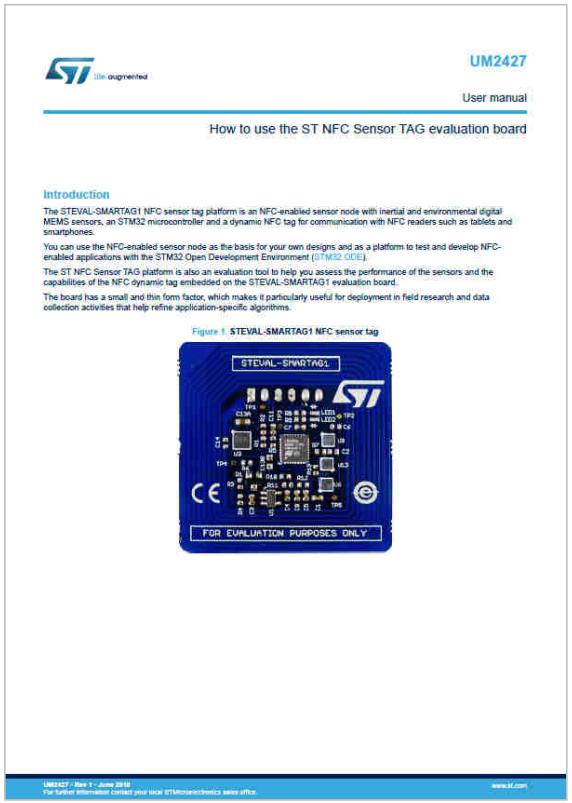


Read multiple tags and
read them further away.

ST NFC Sensor to Cloud

life.augmented

What do you need


UM2427

User manual

How to use the ST NFC Sensor TAG evaluation board

Introduction

The STEVAL-SMARTAG1 NFC sensor tag platform is an NFC-enabled sensor node with inertial and environmental digital MEMS sensors, an STM32 microcontroller and a dynamic NFC tag for communication with NFC readers such as tablets and smartphones.

You can use the NFC-enabled sensor node as the basis for your own designs and as a platform to test and develop NFC-enabled applications with the STM32 Open Development Environment (STM32 ODE).

The ST NFC Sensor TAG platform is also an evaluation tool to help you assess the performance of the sensors and the capabilities of the NFC dynamic tag embedded on the STEVAL-SMARTAG1 evaluation board.

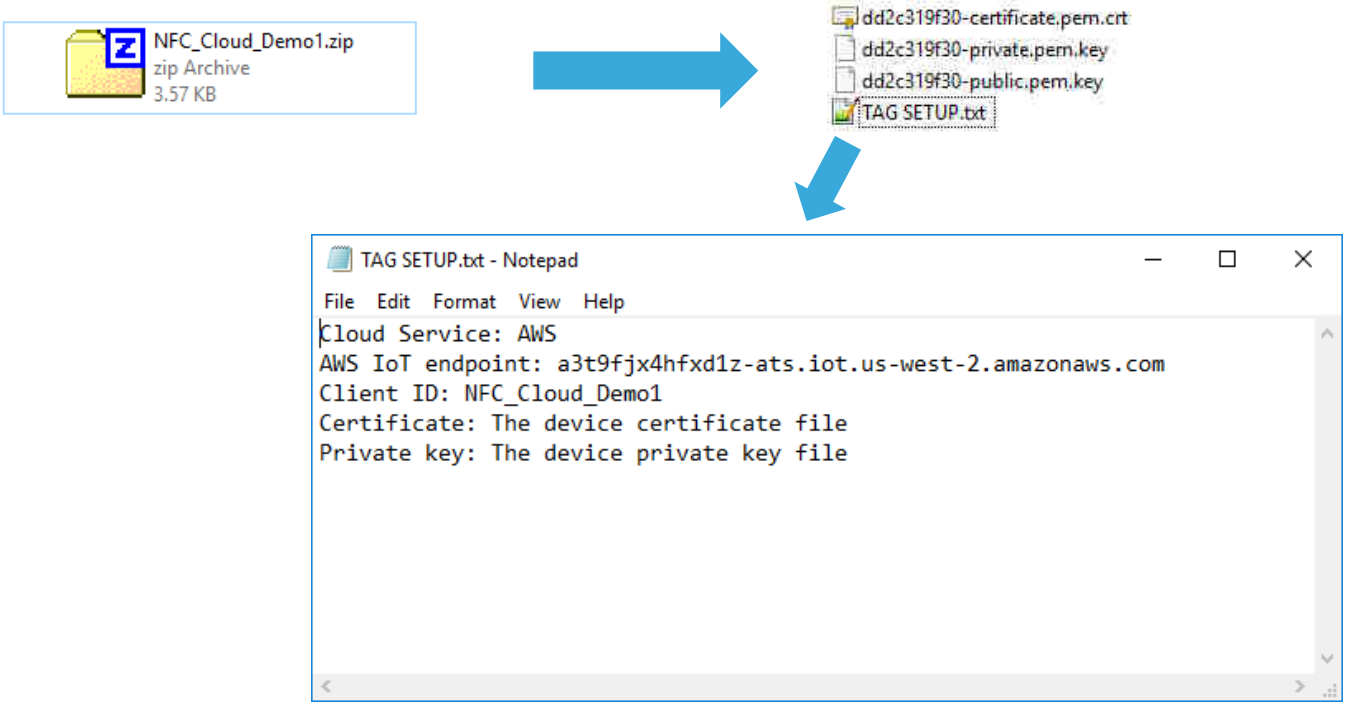
The board has a small and thin form factor, which makes it particularly useful for deployment in field research and data collection activities that help refine application-specific algorithms.

Figure 1. STEVAL-SMARTAG1 NFC sensor tag

- Android 6.0.1 or greater Tablet/Phone
- ST NFC Sensor V1.1.0
- **STEVAL-SMARTAG1**
- CR2032 Battery
- **UM2427**: How to use the ST NFC Sensor TAG evaluation board
- NFC_Cloud_Demo1.zip

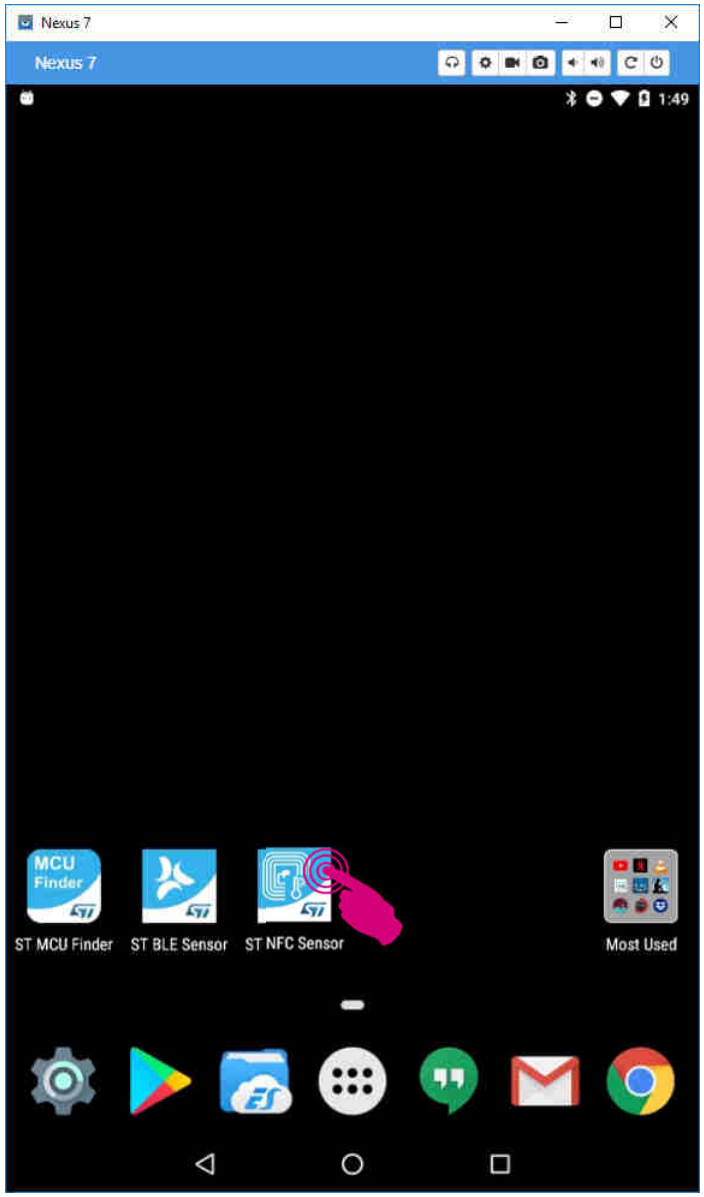
life.augmented

NFC_Cloud_Demo1.zip



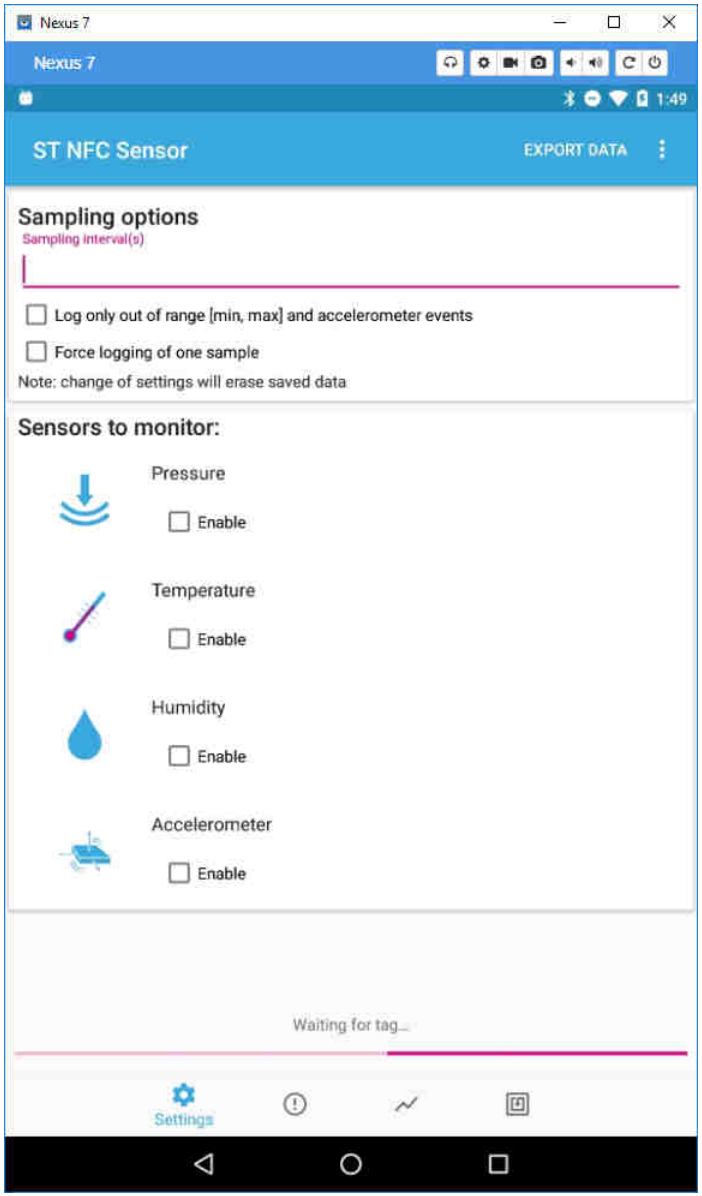
Copy the extracted folder in a known location on the Android Tablet (you'll need this later)

life.augmented



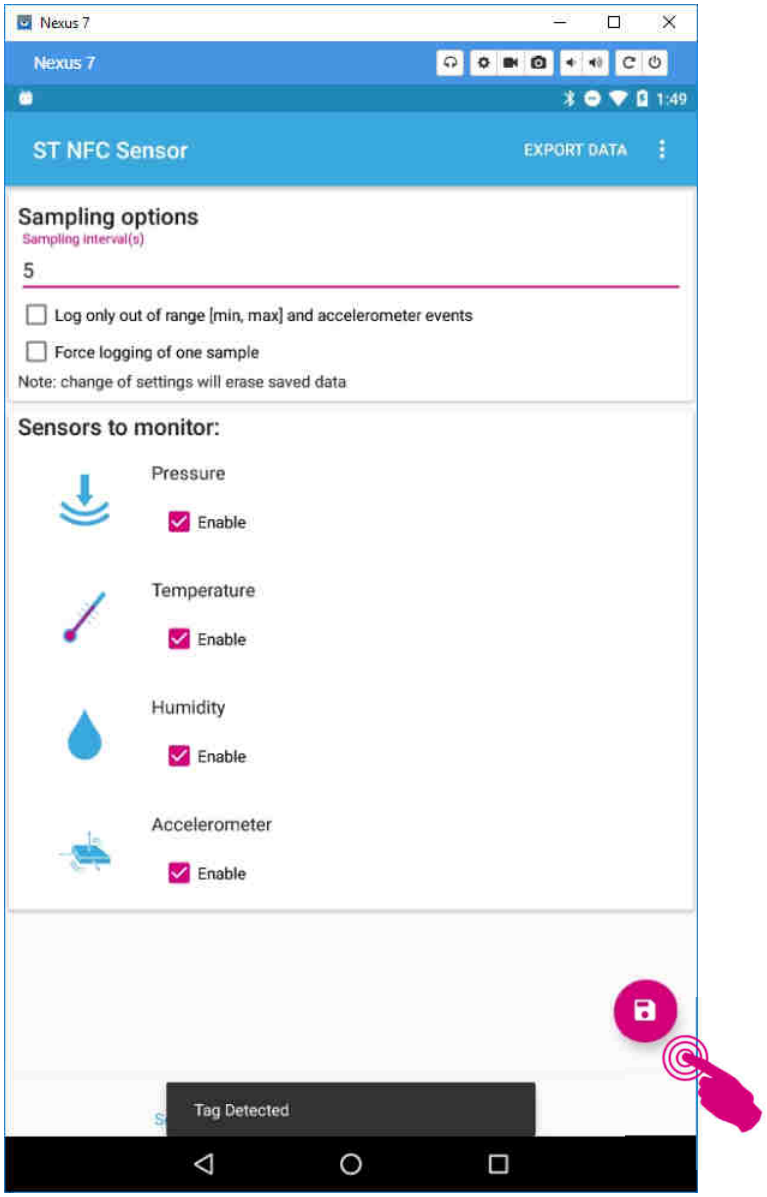
- Open ST NFC Sensor

life.augmented



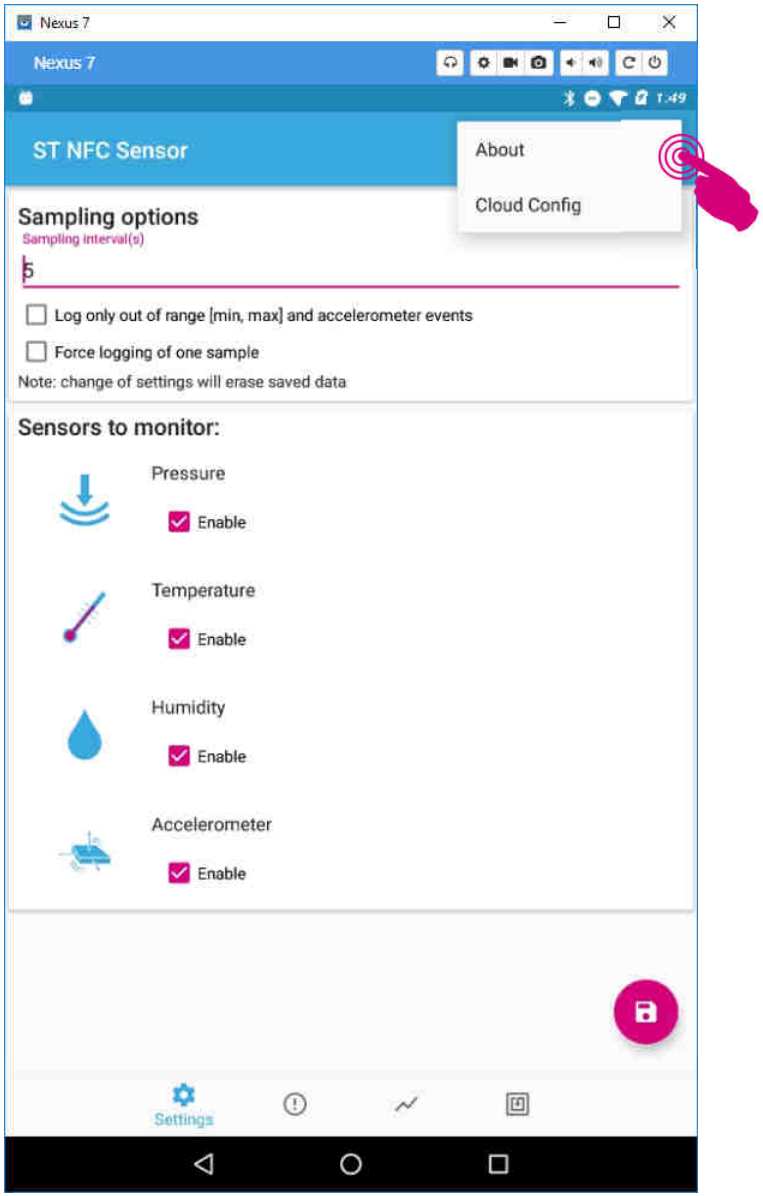
- While on Settings TAB, tap the ST NFC Sensor

life.augmented



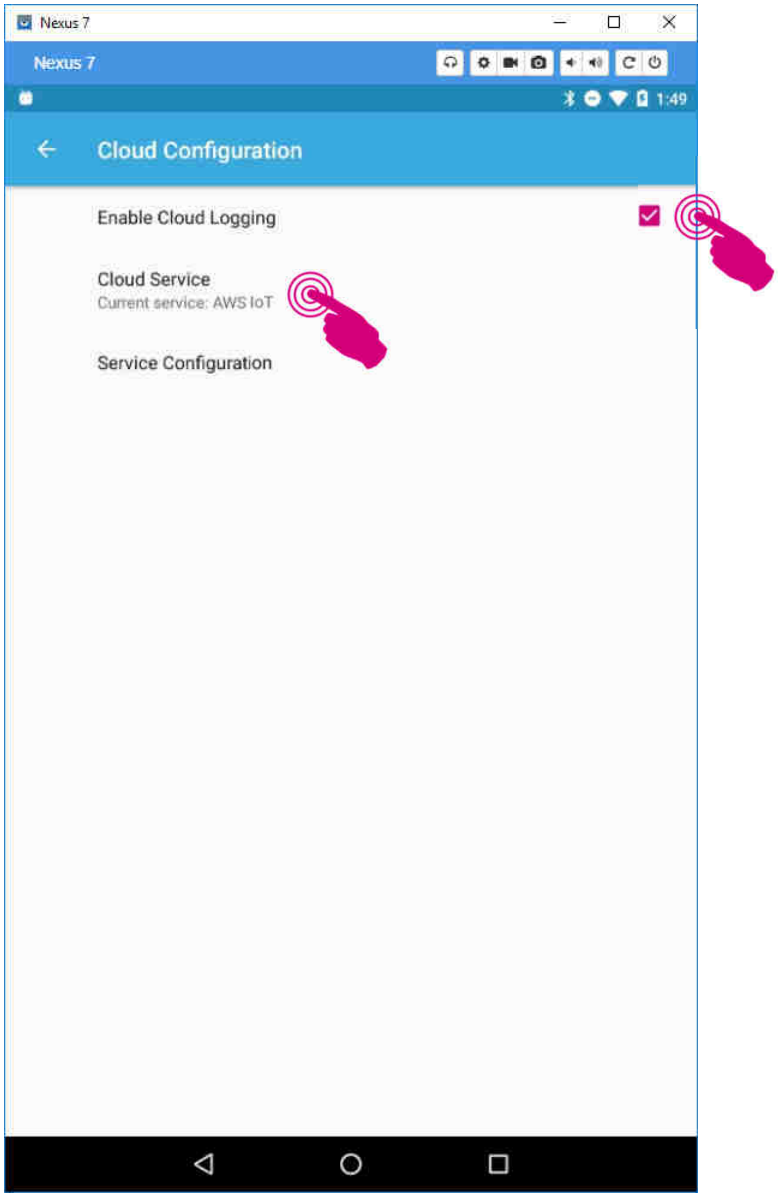
- Previous configuration will be loaded
- You can change to:
  - Sampling option 5s
  - All sensor enabled
- Write tapping on Disc pink icon with ST NFC Sensor in range

life.augmented



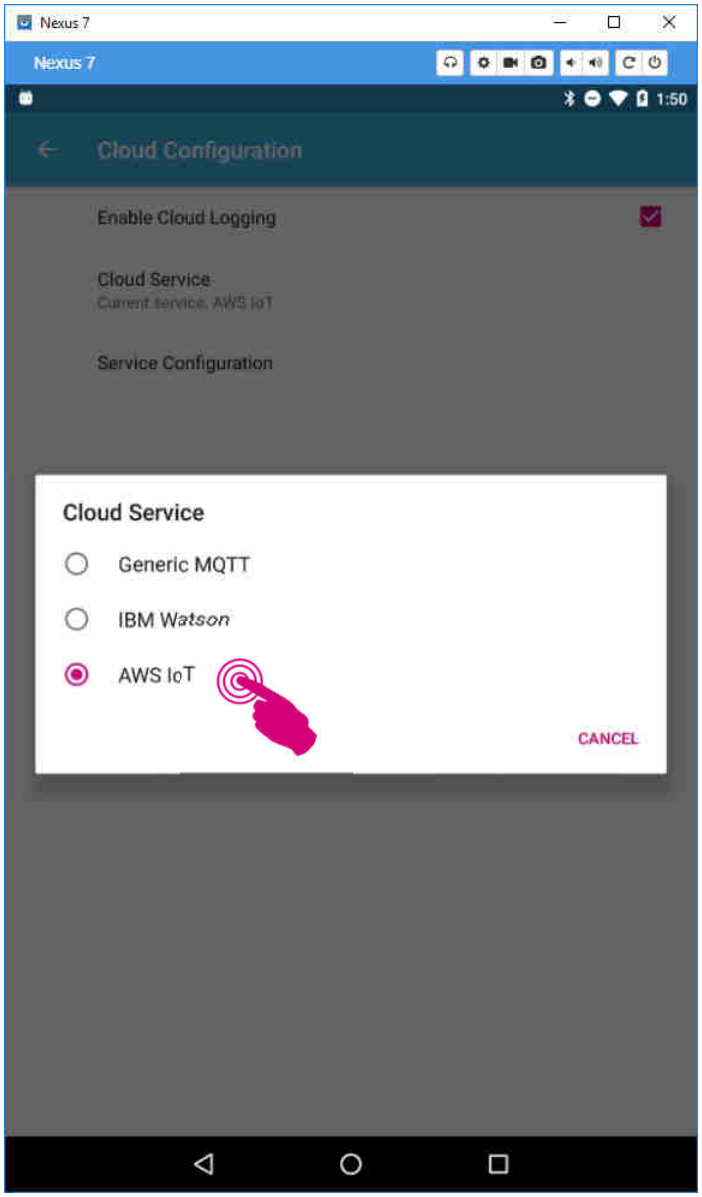
- Tap on the 3 dots on top right corner
- Tap on **Cloud Config**

life.augmented



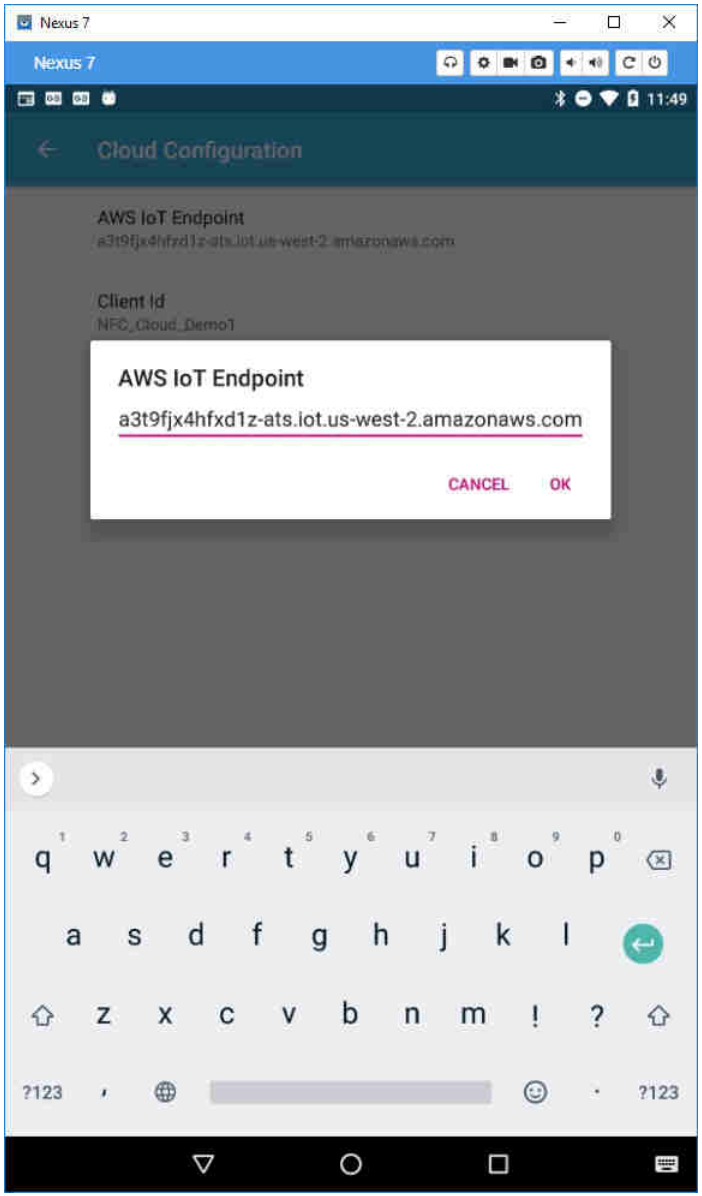
- Tap on **Enable Cloud Logging**
- Tap on **Cloud Service**

life.augmented



- Tap on **AWS IOT**

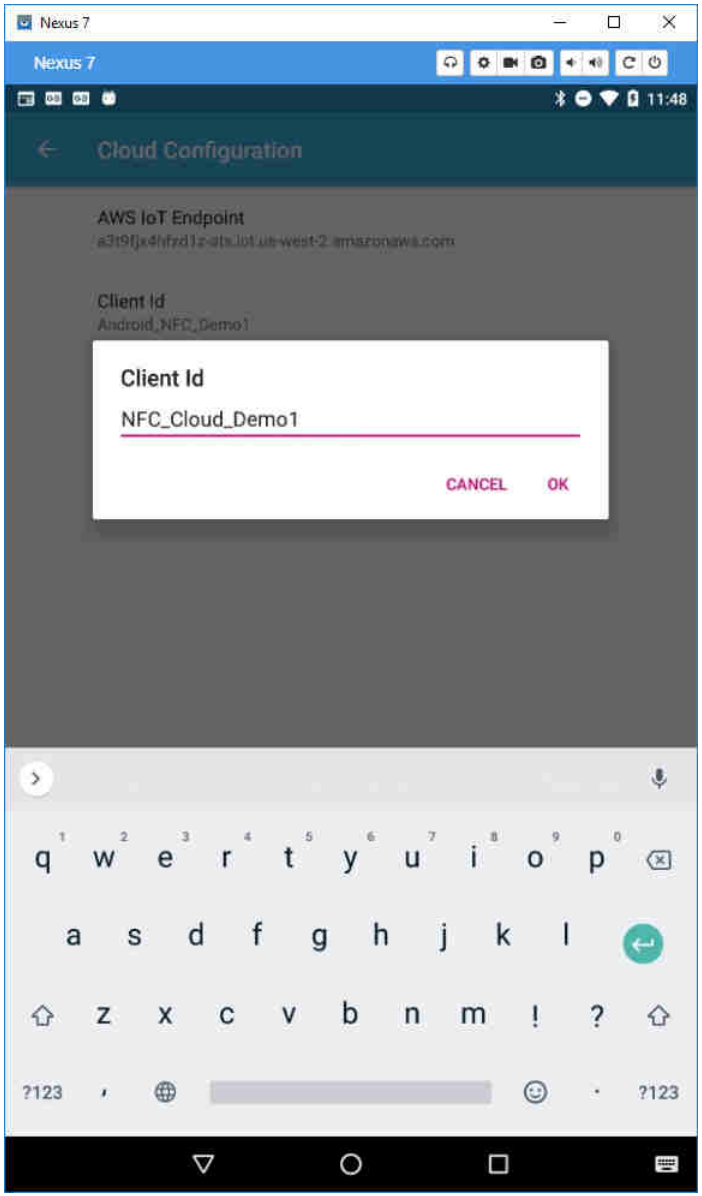
- On the Cloud configuration tool we'll now need to insert
  - **AWS IoT Endpoint**
  - **Cliend ID**
  - **Certificate**
  - **Private Key**
- Recommendation is to have a txt file with AWS IoT Endpoint and client ID to paste in the field (TAG SETUP.txt)
- Certificate and Private Key in separate files (downloaded from AWS).



- Cut and paste AWS IoT Endpoint

```
a3t9fjx4hfxd1z-ats.iot.us-
west-2.amazonaws.com
```

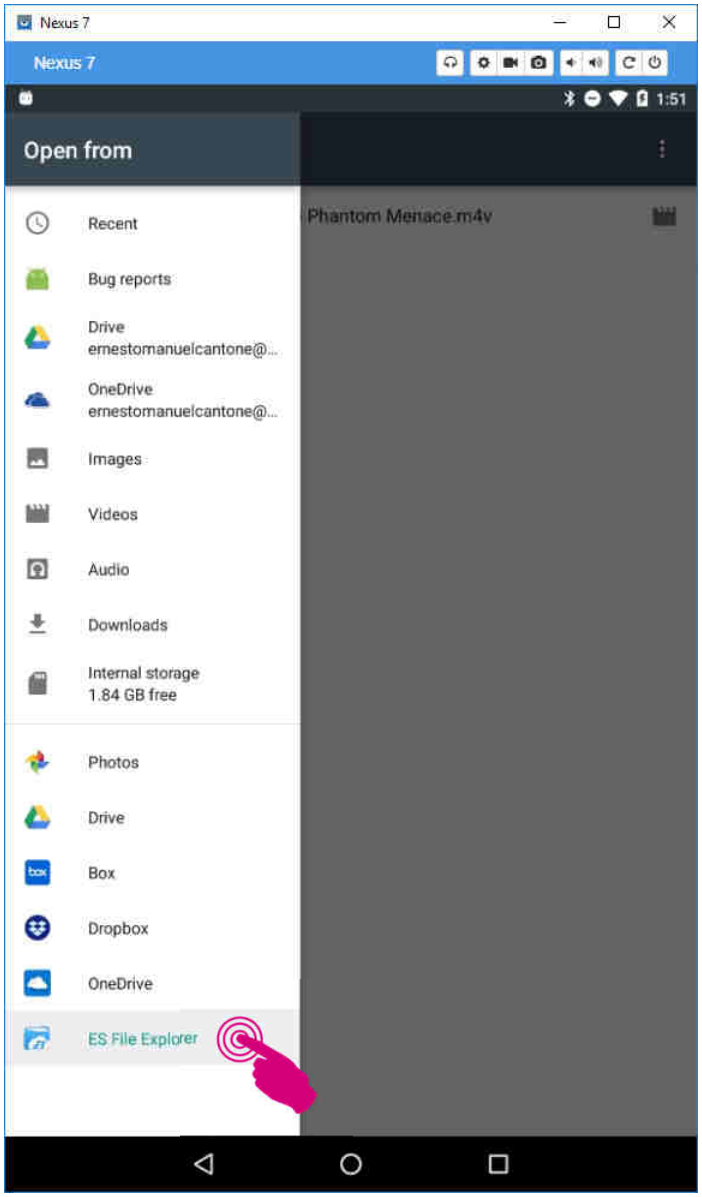
life.augmented



- Cut and paste Client Id

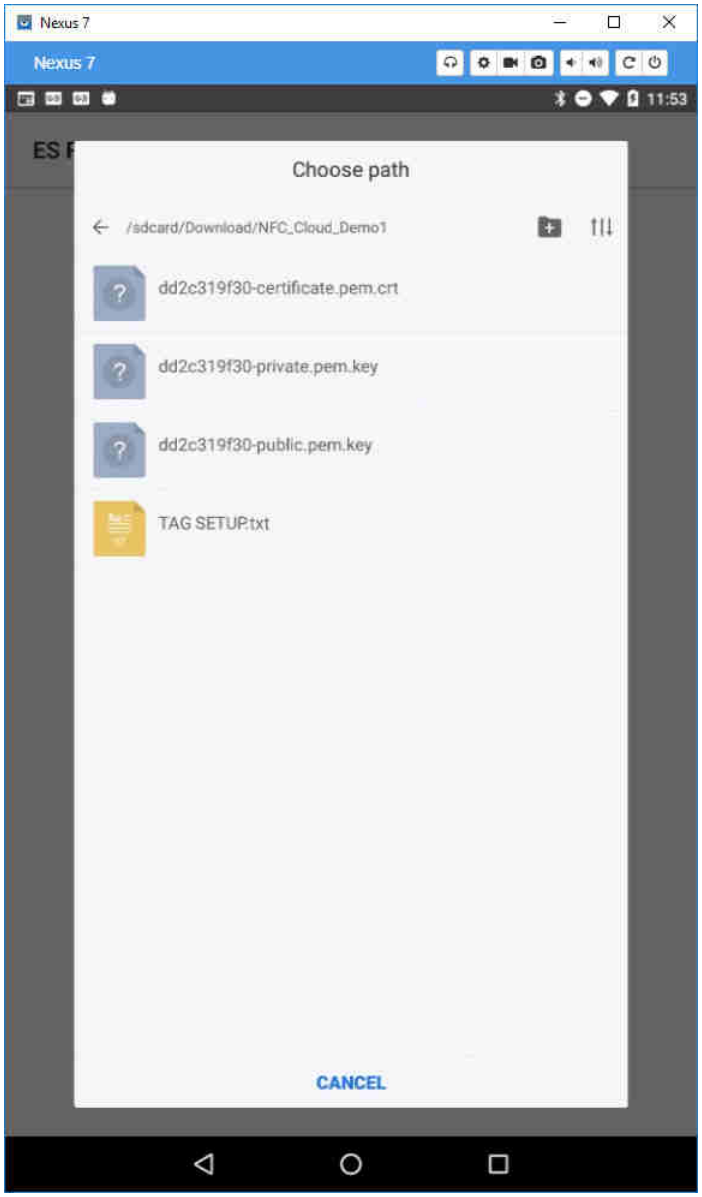
`Android_NFC_Demo1`

life.augmented



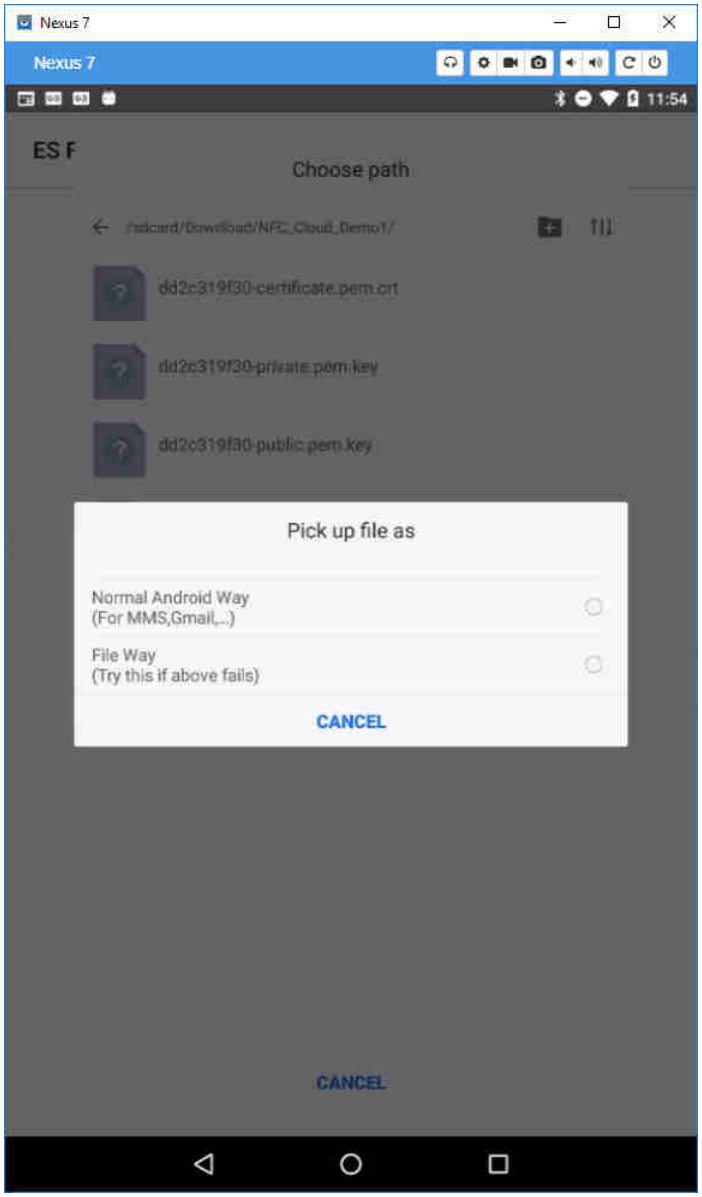
- Browse your tablet with a File Explorer app (ES file Explorer in this example) to locate the correct certificate/key
  - NOTE: a cloud location can also be used (Box, Dropbox, Google Drive, OneDrive, …)

life.augmented



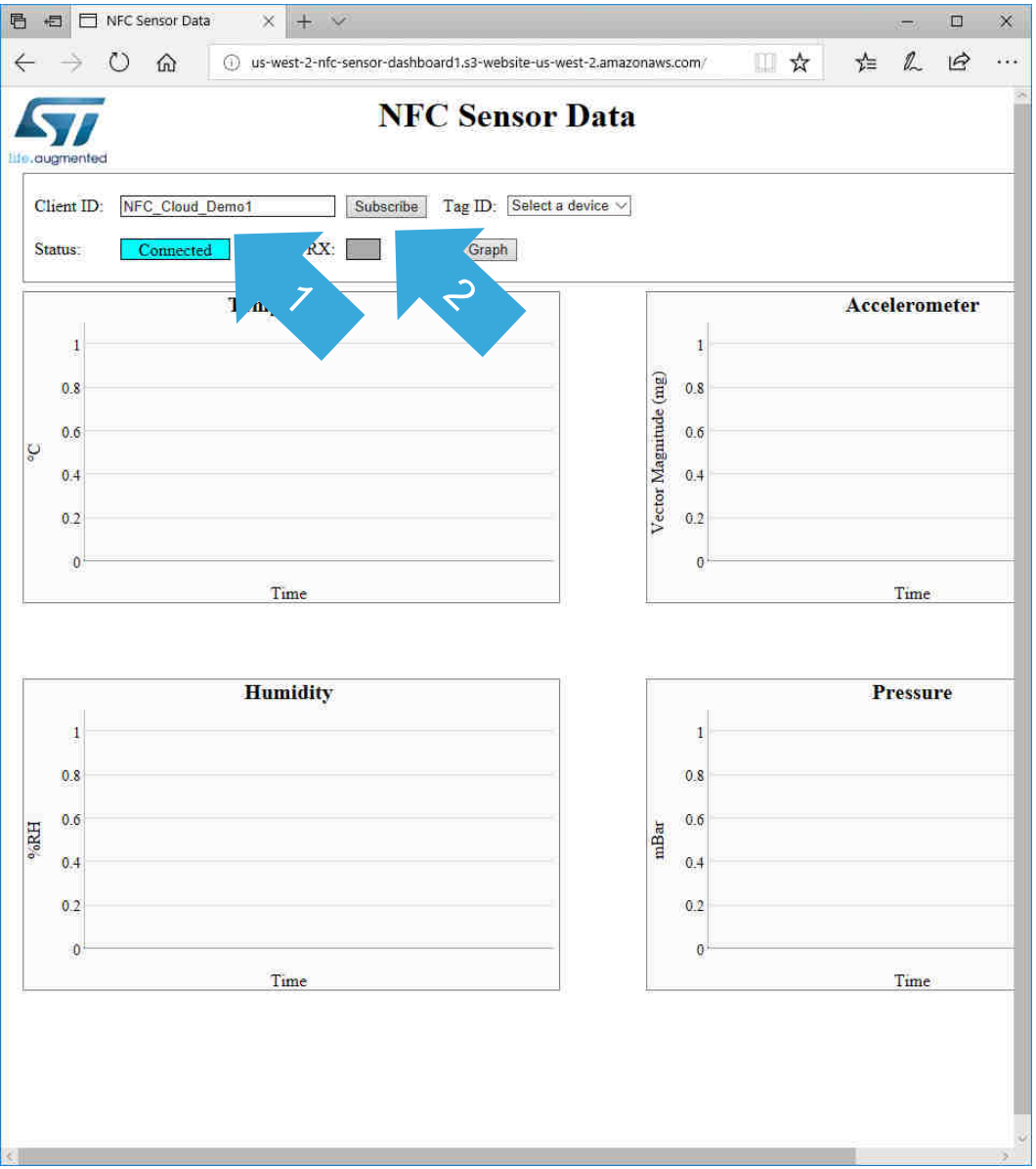
- In this example we stored the files in the folder `..\Download/NFC_Cloud_Demo1`
- In this location we have the following files:
  - `dd2c319f30-certificate.pem.crt`
  - `dd2c319f30-private.pem.key`
  - `dd2c319f30-public.pem.key`

life.augmented



- Link the Certificate and Private Key with the provided methods
  - In the example “File Way” was used

- Once every field is filled you can go back and they will be stored in the app for next access



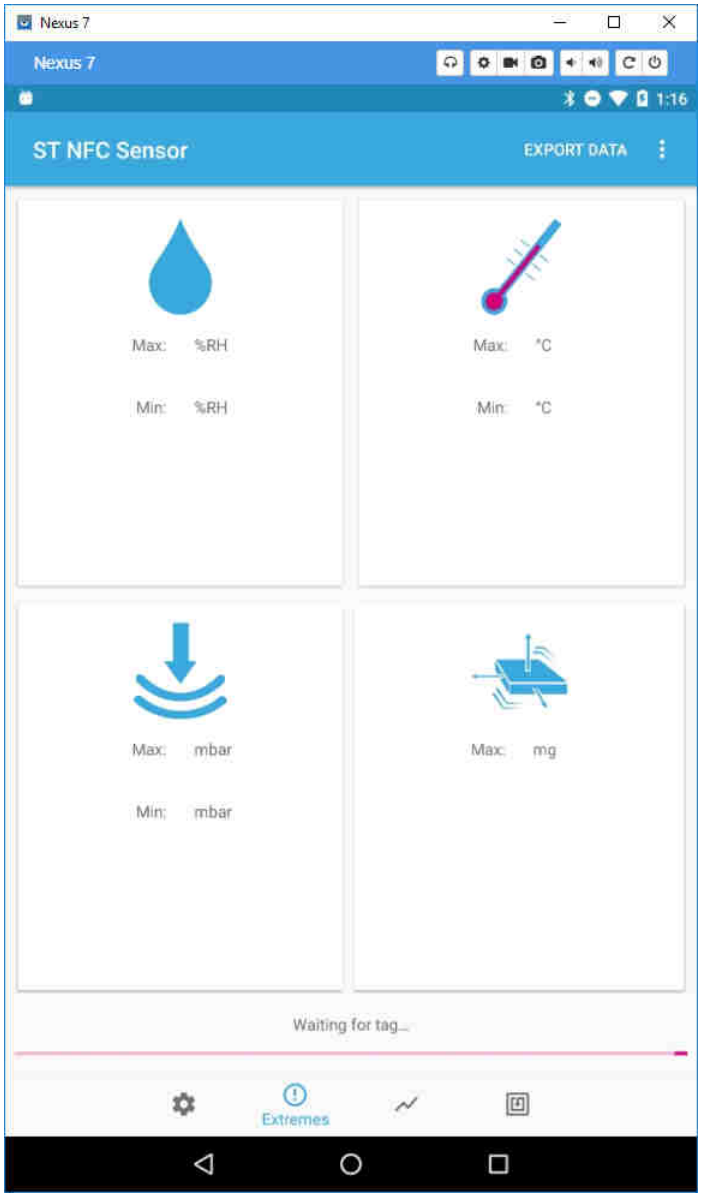
- On a browser of your choice (Chrome is visualized) go to

http://us-west-2-nfc-sensor-dashboard1.s3-website-us-west-2.amazonaws.com/

- On the Client ID field insert

NFC_Cloud_Demo1

- Click Subscribe

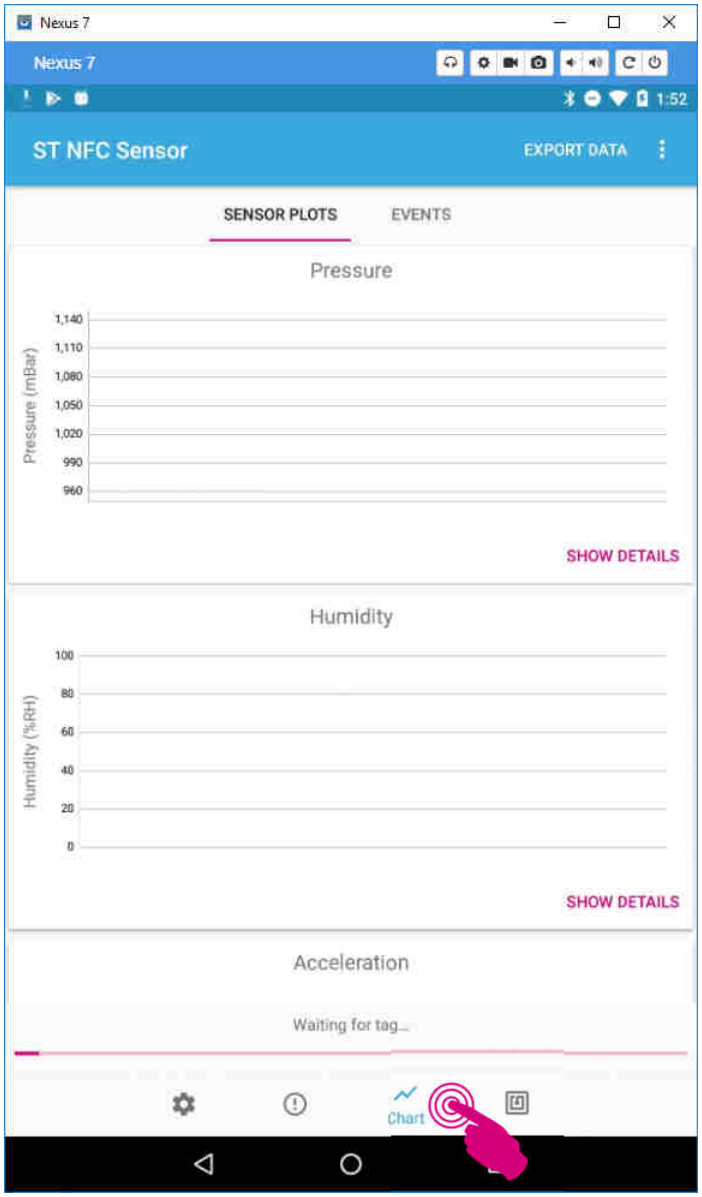


- Select **Extremes** Tab

life.augmented

- Tap Tag to set Extremes

life.augmented

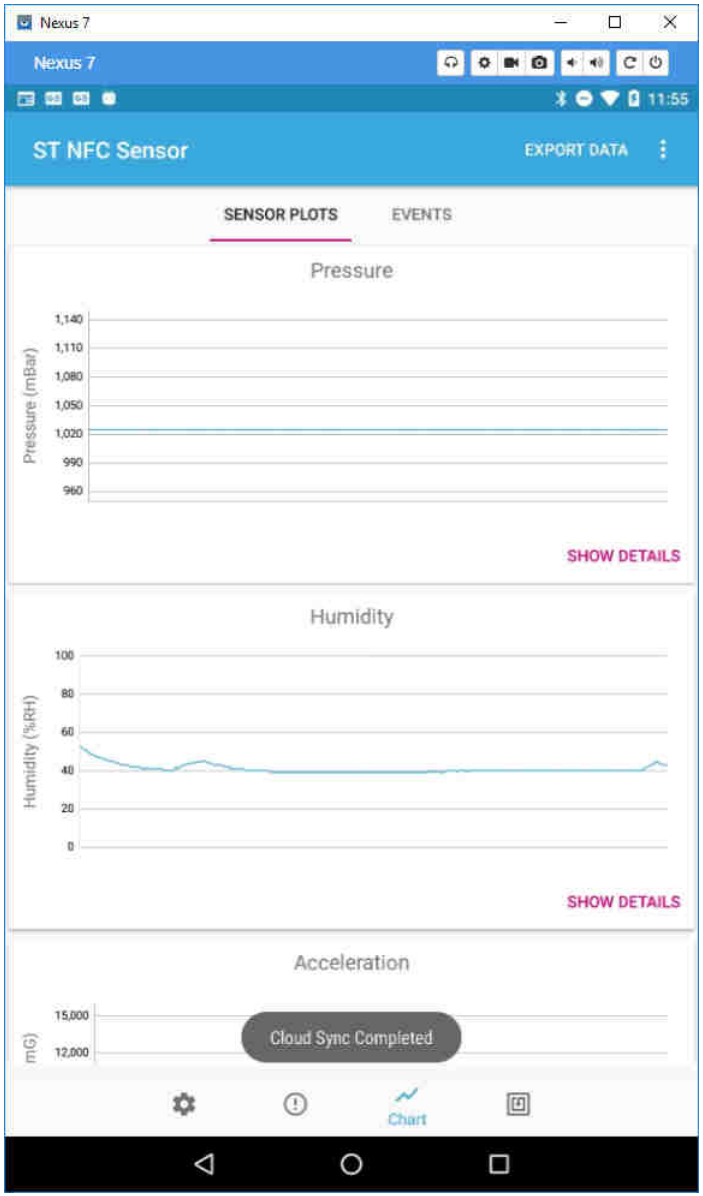


- Select **Chart** Tab

life.augmented

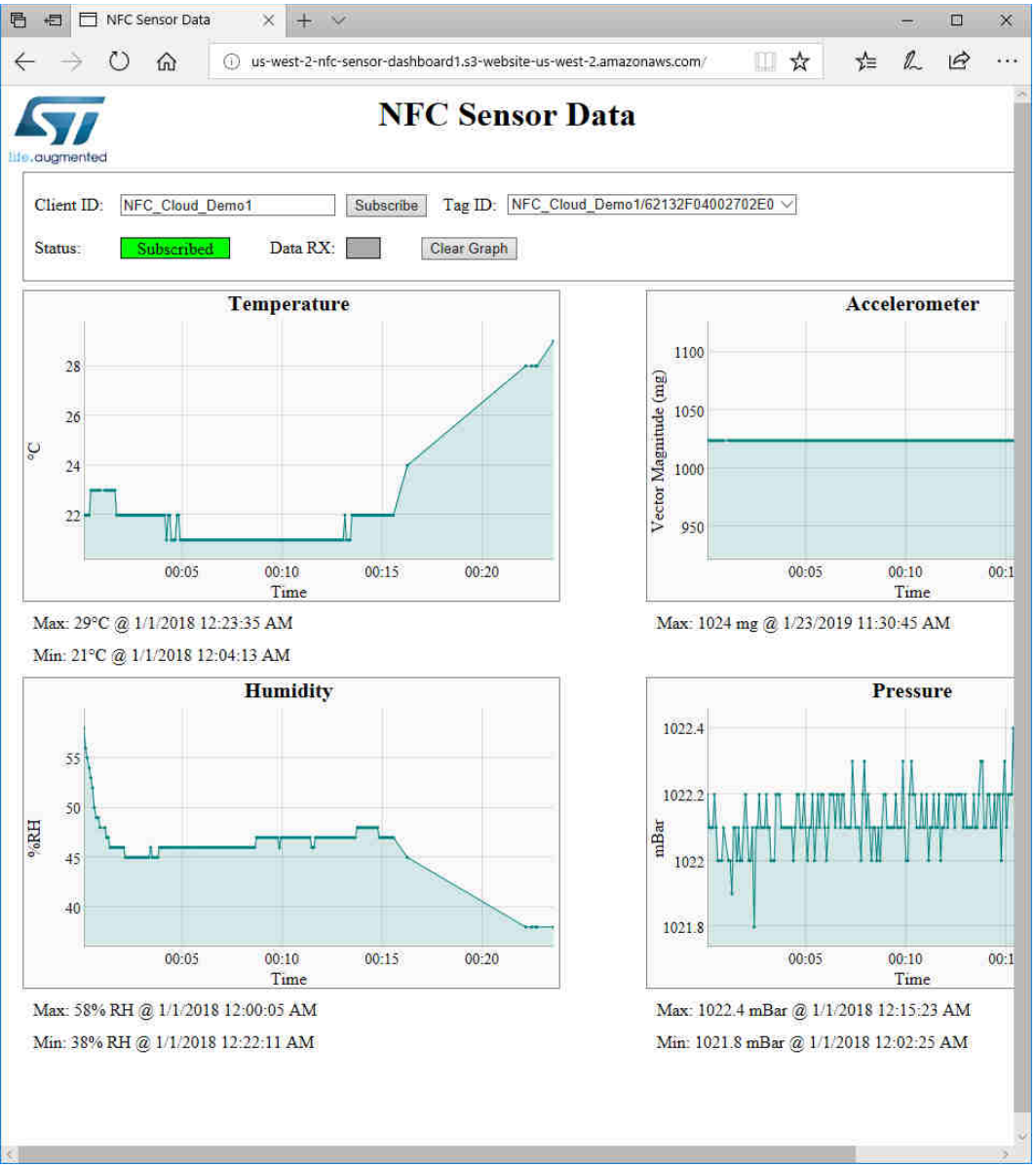
- Tap the NFC Sensor

life.augmented

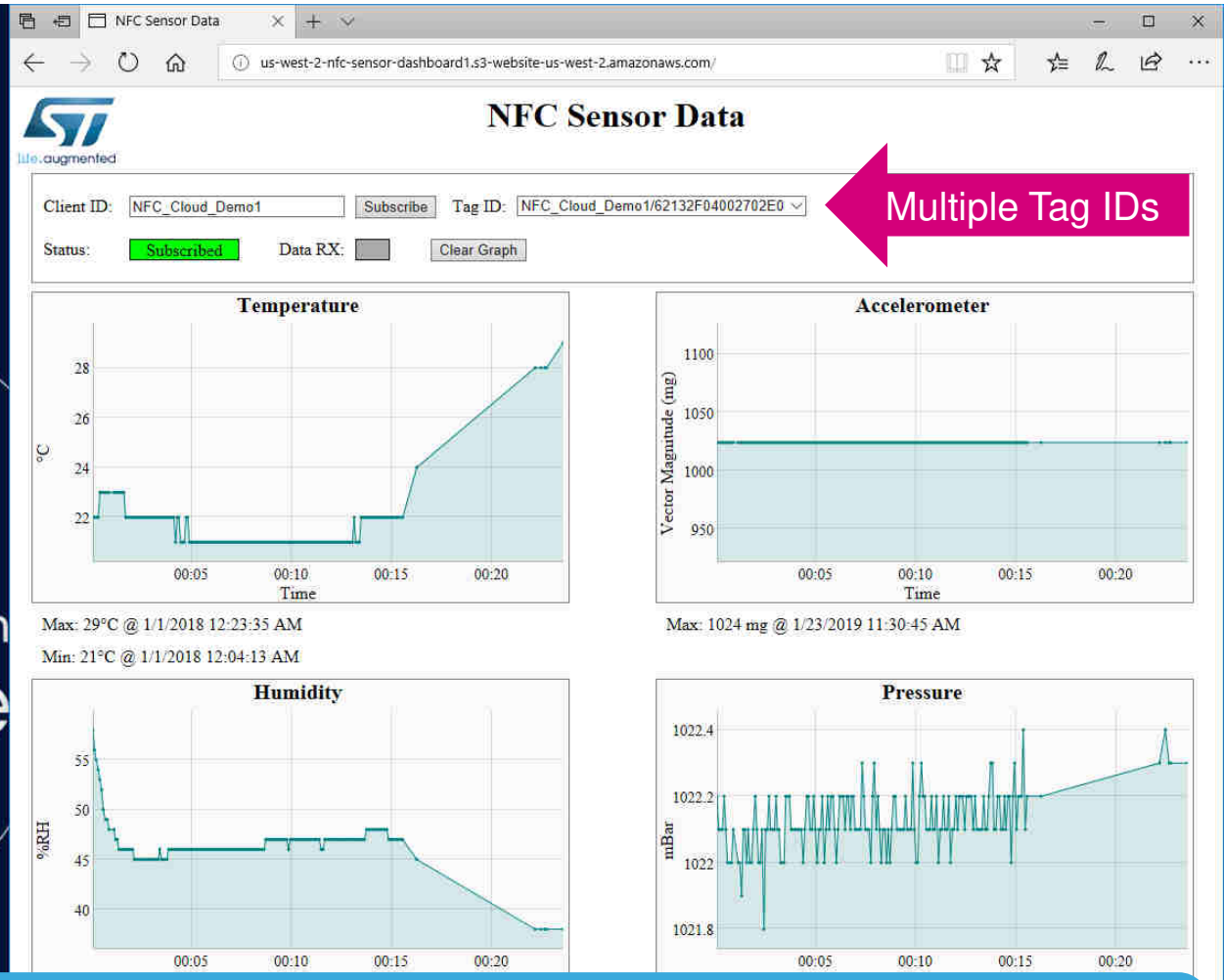


- Data will be automatically sync'd with the Cloud

life.augmented



- When a NFC Sensor is now read from a tablet data will be displayed



At every tap from the NFC Sensor data will now be visualized both on the tablet and on the cloud dashboard.
EXAMPLE: Multiple users in different moments and with different handheld devices can now tap the same NFC Sensor and post on the same Cloud dashboard to rebuild a shipment history

What’s happening in the background?

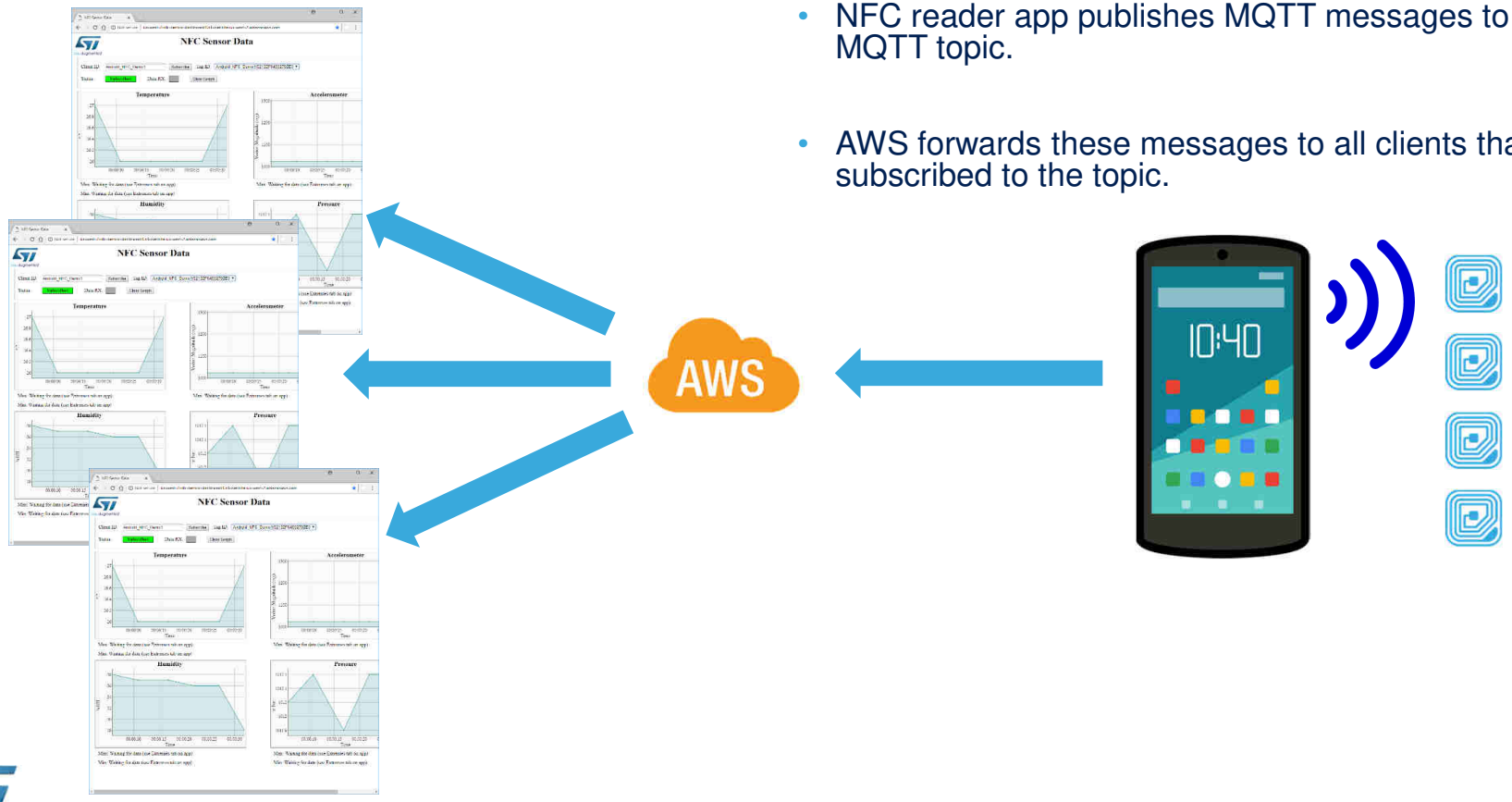
Web Dashboard

- Web dashboard is a browser-hosted JavaScript application based around the open-source Paho MQTT client.
  - MQTT is a simple one-to-many messaging protocol
  - Clients publish and subscribe to one or many MQTT “topics”
- Web dashboard HTML is hosted in an AWS S3 bucket
  - Provides data storage, public web hosting, and DNS
- NFC reader app connects to the AWS cloud and publishes sensor data to an MQTT topic of the form:

  <Client ID>/<Tag Serial Number>/<Data Type>

  Where:
  - Client ID is the AWS Thing name used by the NFC reader app
  - Tag Serial Number is the S/N exposed by the NFC tag itself
  - Data Type is “extremes” (for min/max data) or “sensorData” (for charts)
- Web dashboard subscribes to MQTT messages sent from <Client ID> and parses messages based on Tag S/N and data type.

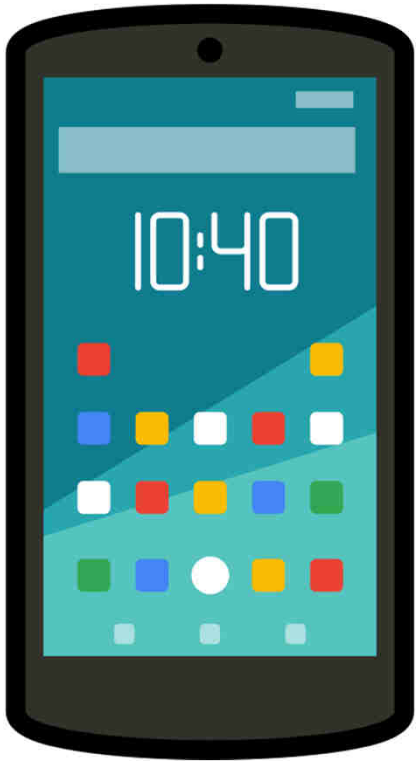
MQTT Message Publication



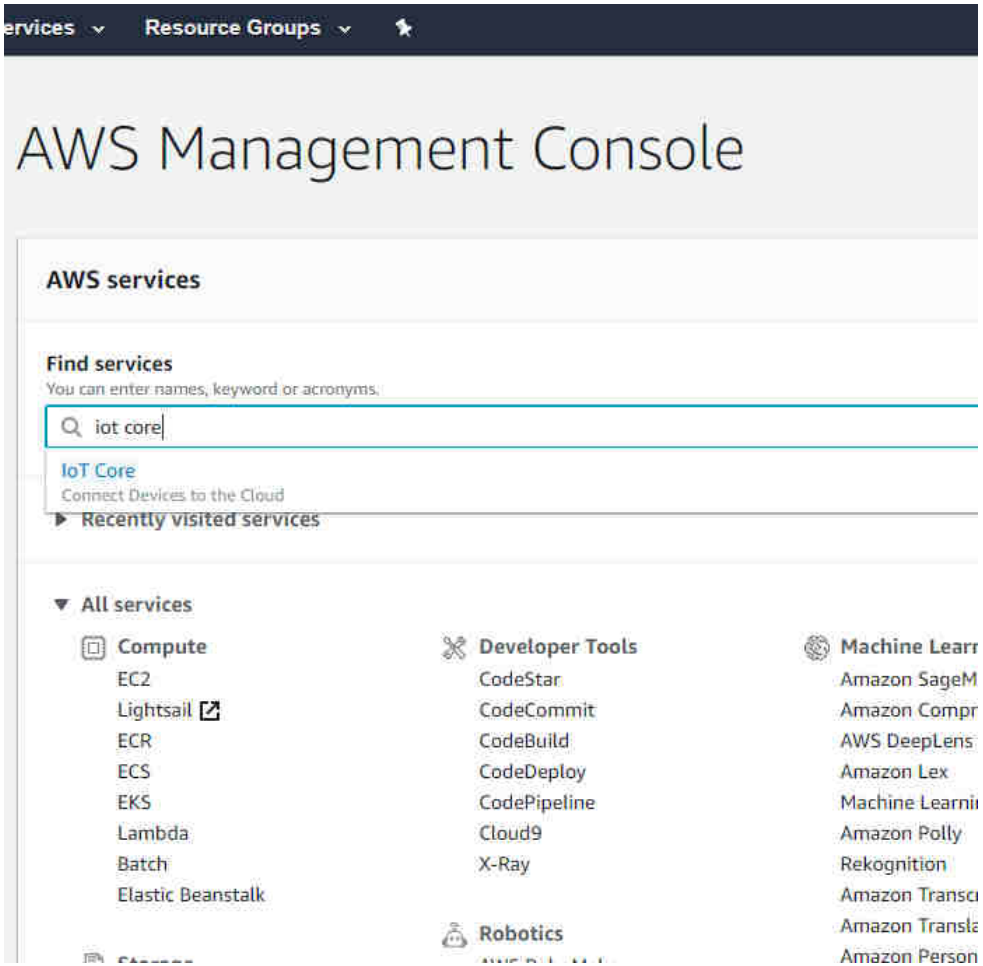
- NFC reader app publishes MQTT messages to a given MQTT topic.
- AWS forwards these messages to all clients that are subscribed to the topic.

life.augmented

AWS Thing Creation



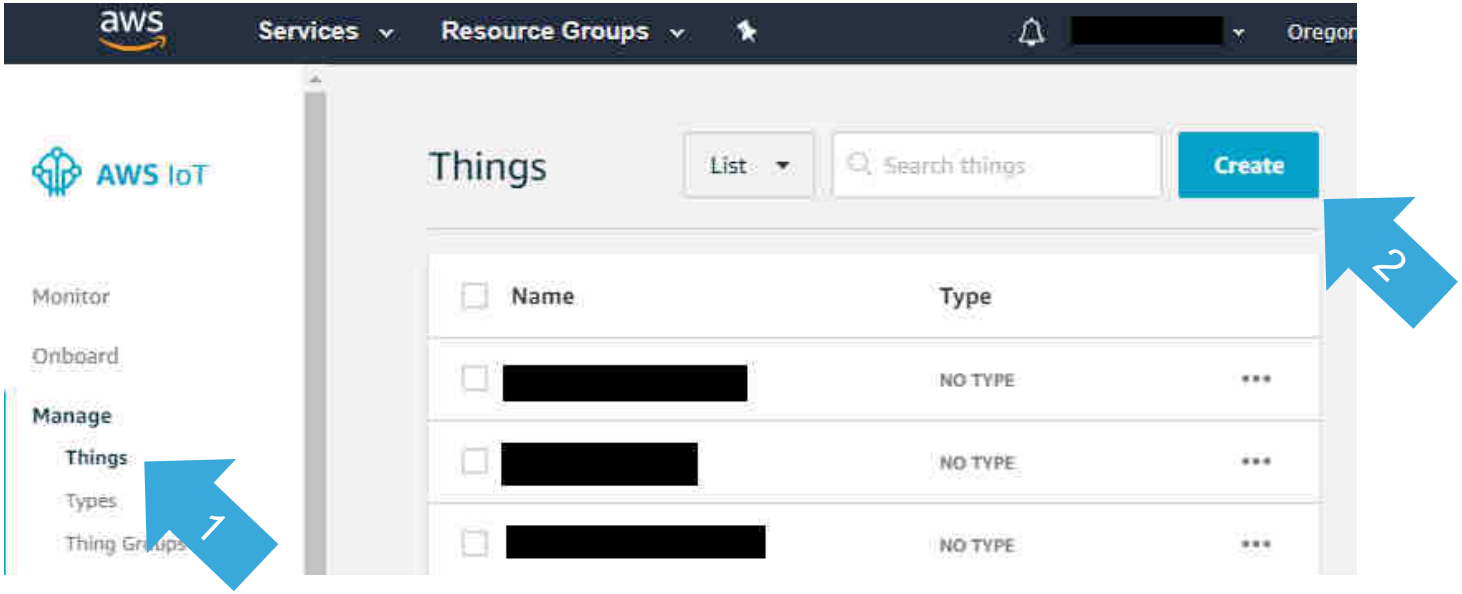
- NFC reader app is an AWS IoT “Thing”
- Thing must be created and registered with AWS prior to use.



AWS Thing Creation

- Login to AWS console
- Go to the AWS IoT Core control panel

AWS Thing Creation



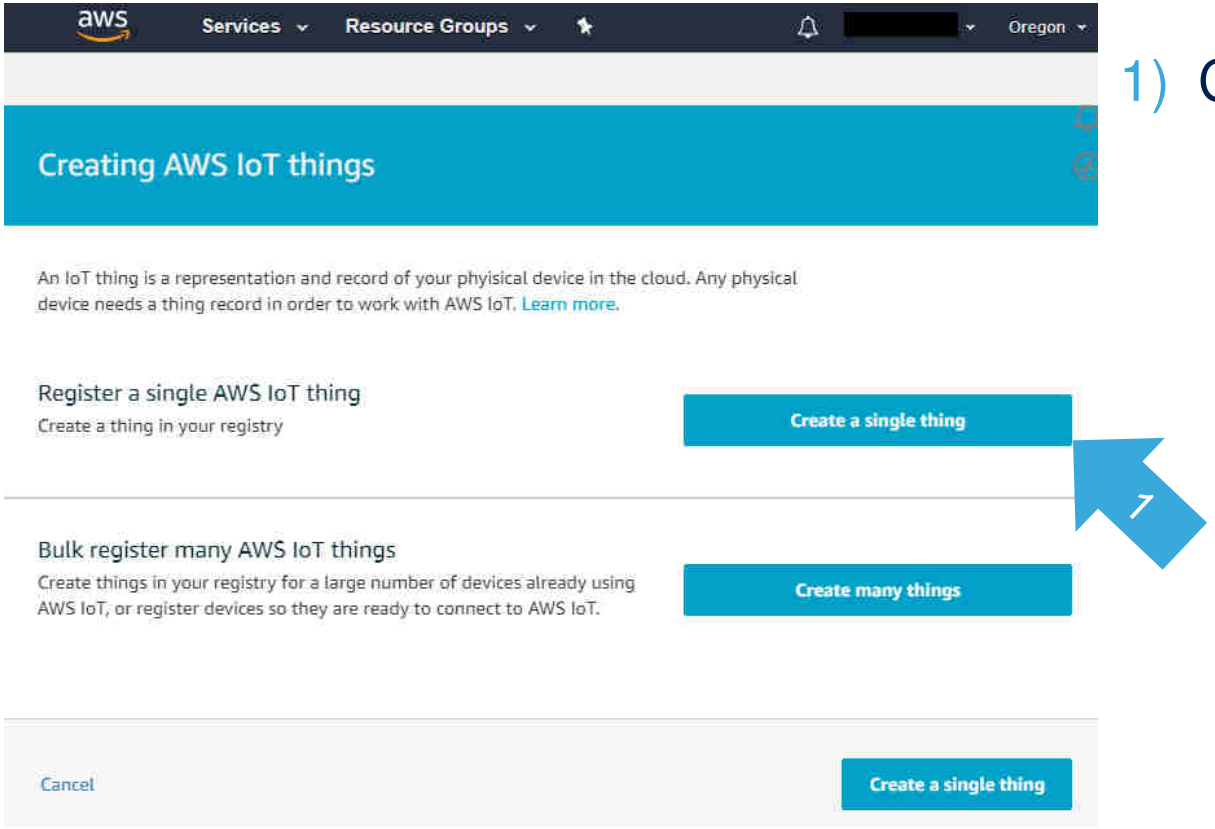
1) Go to Manage -> Things

2) Go to the IoT Core control panel

life.augmented

AWS Thing Creation

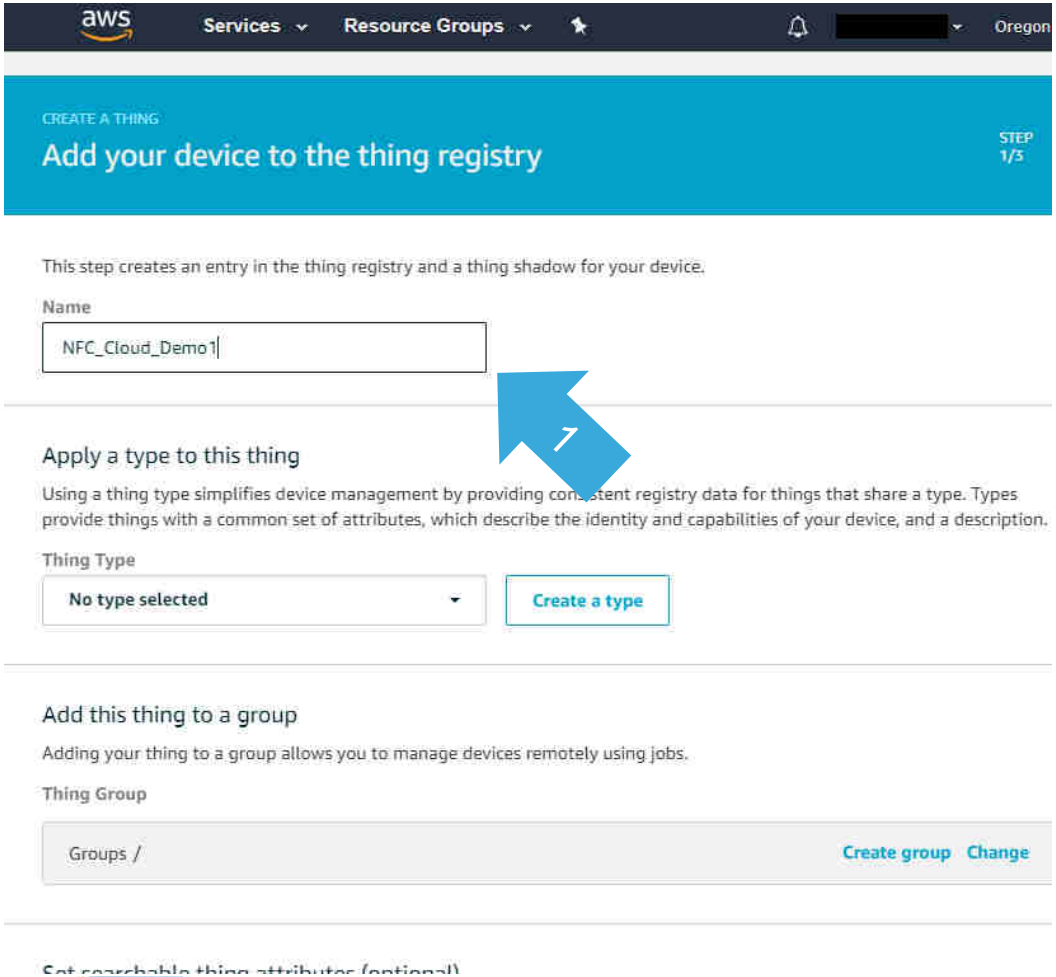
1) Click “Create a single thing”



life.augmented

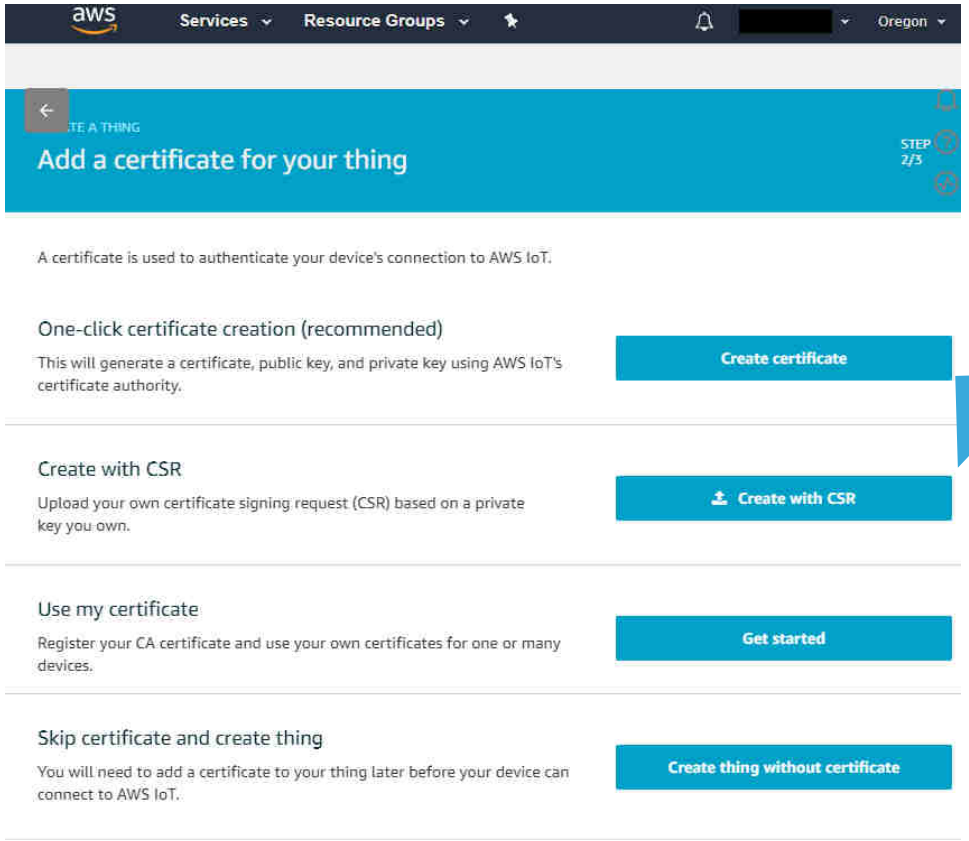
AWS Thing Creation

1) Give the new thing a unique name



life.augmented

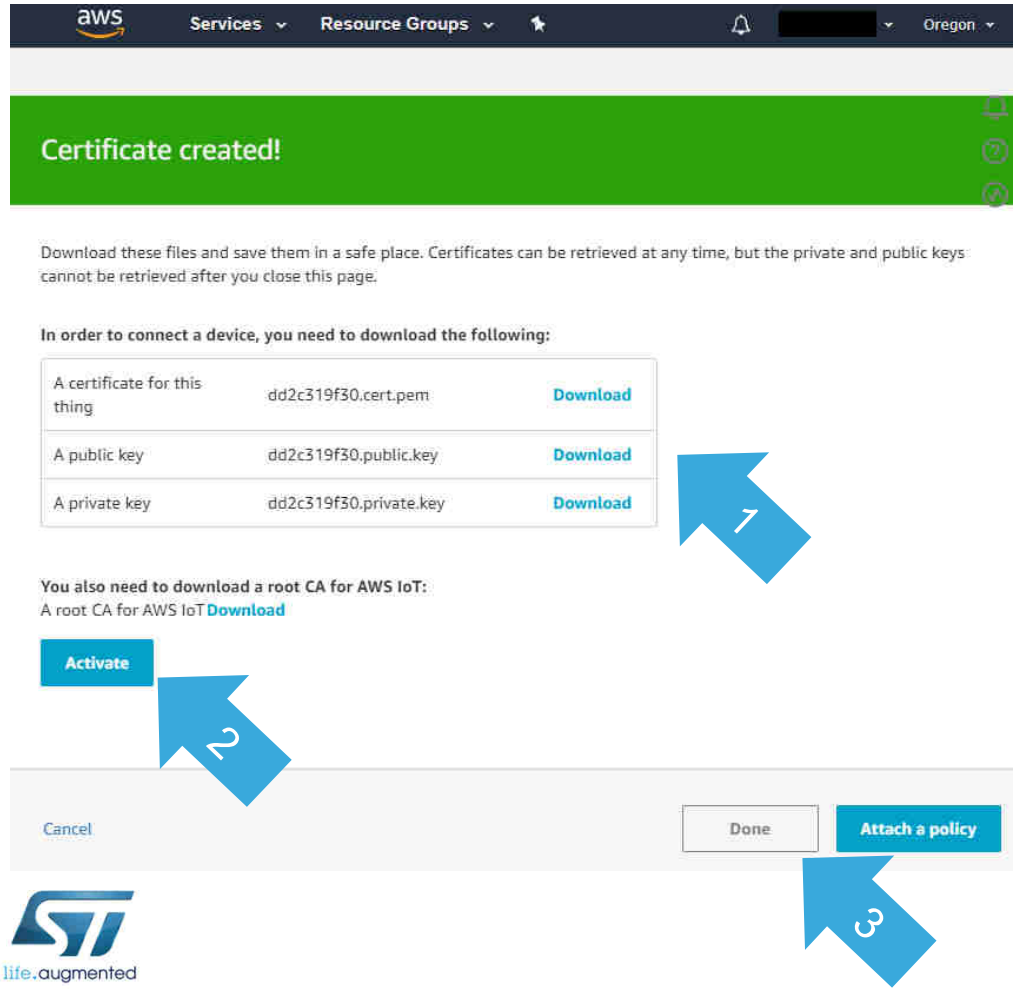
AWS Thing Creation



1) Create a device certificate using the One-click certificate creation method.

life.augmented

AWS Thing Creation



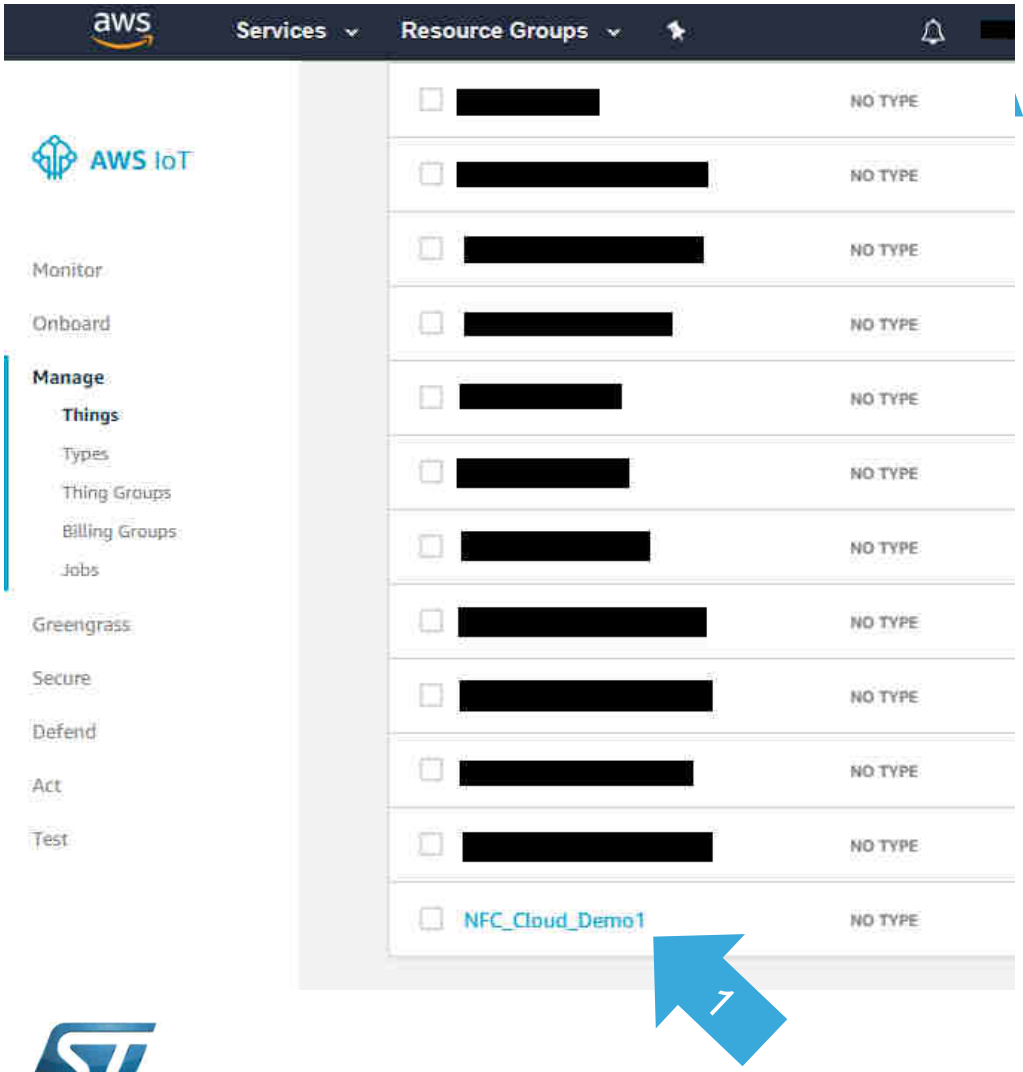
1) The device certificate and key pair have been created. Download these now and store them in a safe location. **This is your only chance to download the private key.**

2) Click Activate

3) Click Done (we'll attach a policy later)

life.augmented

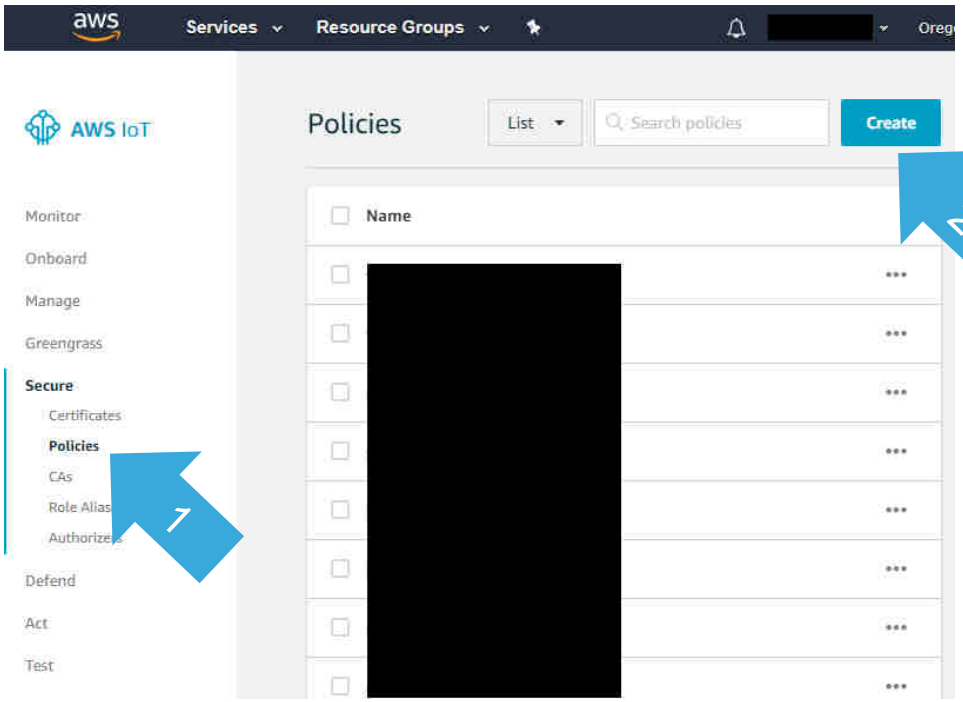
AWS Thing Creation



1) The new AWS Thing should now appear in the list of registered Things.

life.augmented

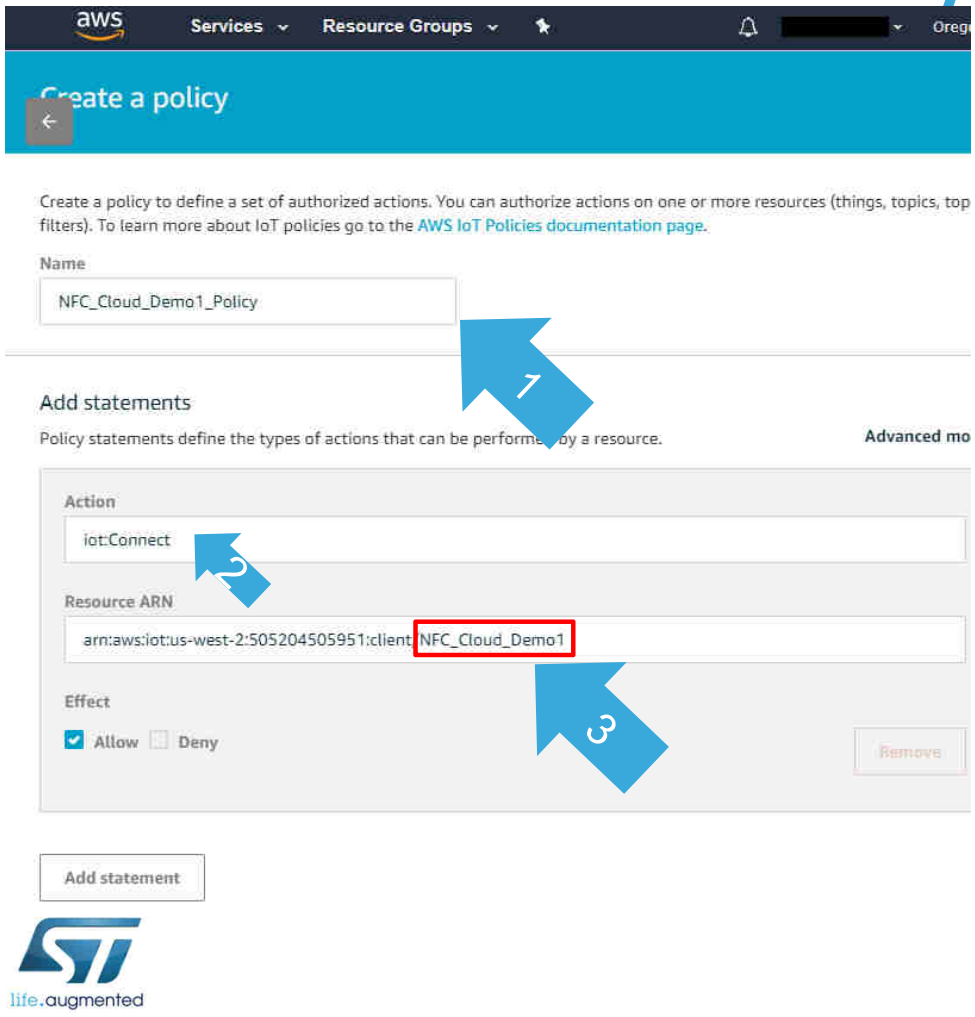
AWS Thing Creation



1) Go to Secure → Policies

2) Click on Create

AWS Thing Creation



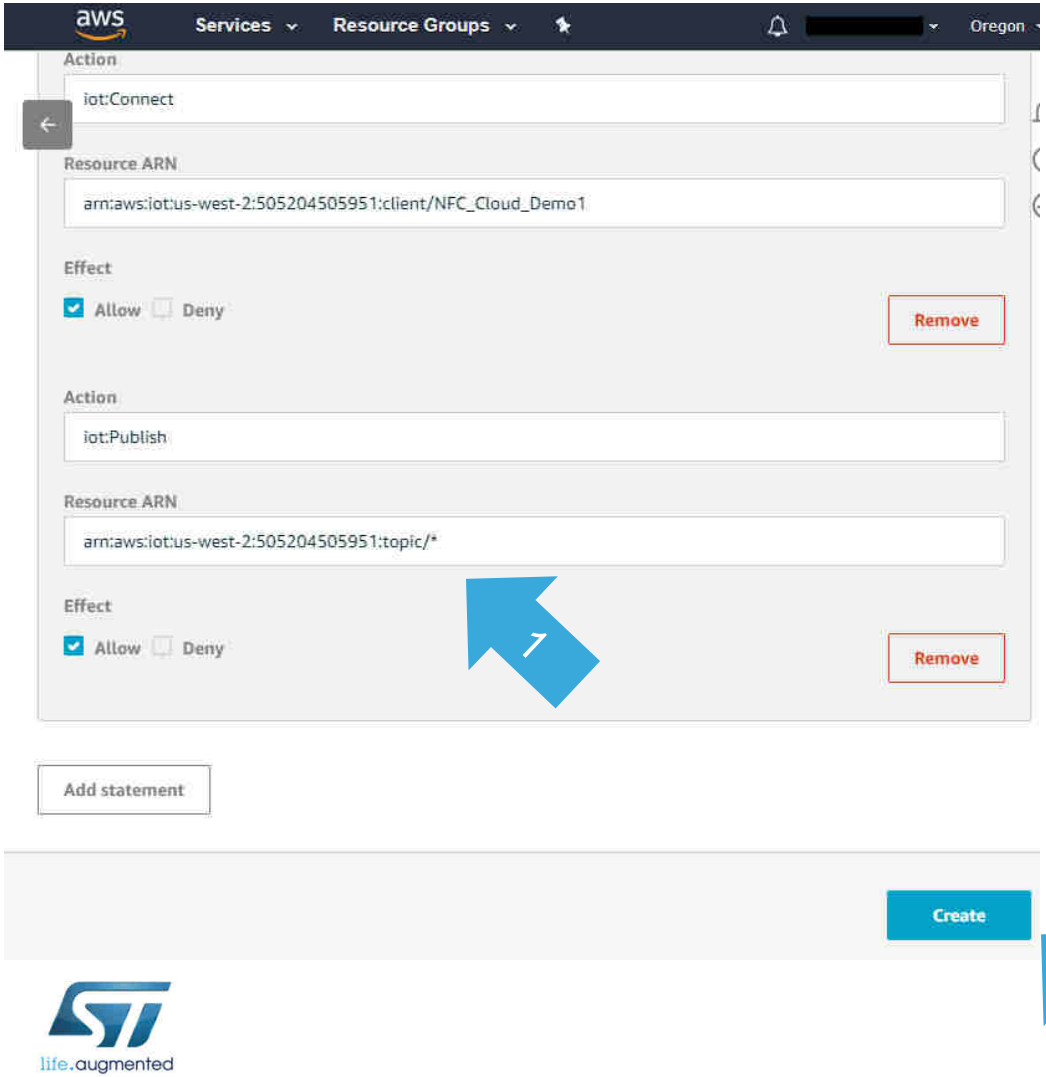
1) Give the new policy a unique name

2) Add a new statement as follows:
   a) Action: “iot:Connect”
   b) Resource ARN: Change client name at end of string to match the AWS thing name you’ve created in the previous steps
   c) Effect: Allow

3) Click Add statement

life.augmented

AWS Thing Creation



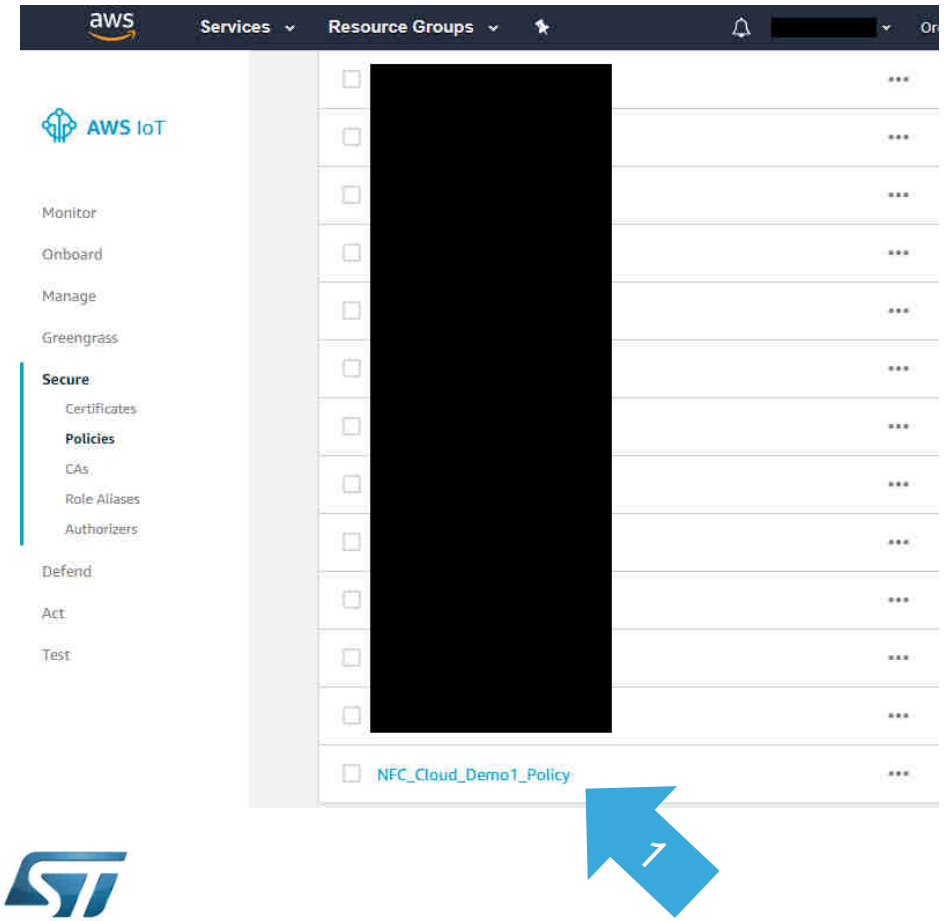
1) Add a new statement as follows:
   a) Action: “iot:Publish”
   b) Resource ARN: Change topic name at end of string to “*” to represent all topics
   c) Effect: Allow

These two statements combined allow the NFC_Cloud_Demo1 thing to connect to AWS IoT Core services and publish MQTT messages to any topic.

2) Click "Create“

life.augmented

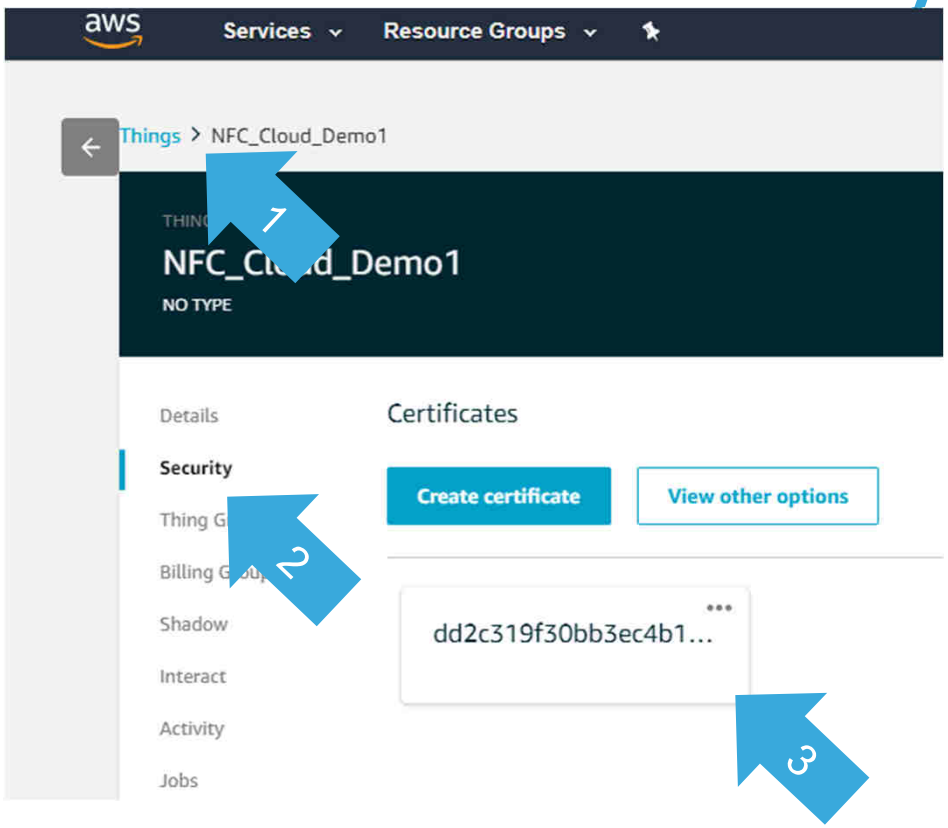
AWS Thing Creation



1) The policy list should now include the newly-created policy.

life.augmented

AWS Thing Creation



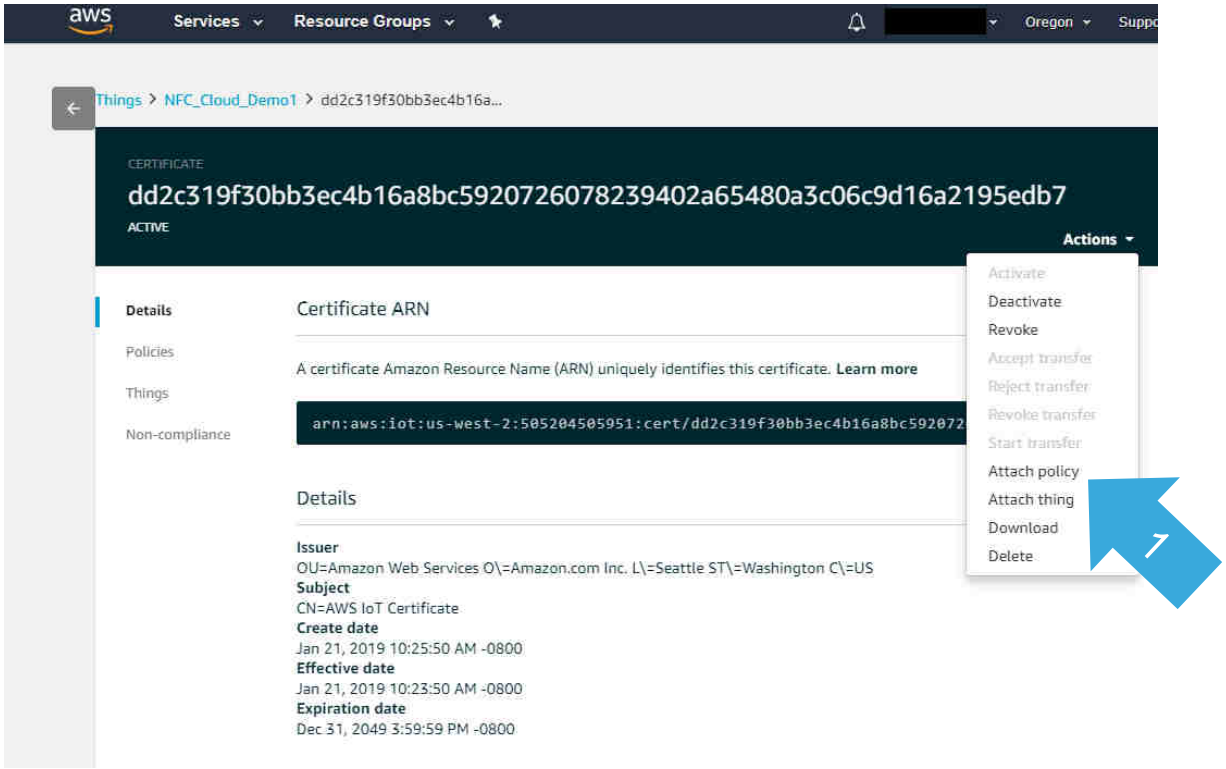
1) Go back to Manage → Things and click on your newly-created thing.

2) Go to the Security tab

3) Click on the certificate

life.augmented

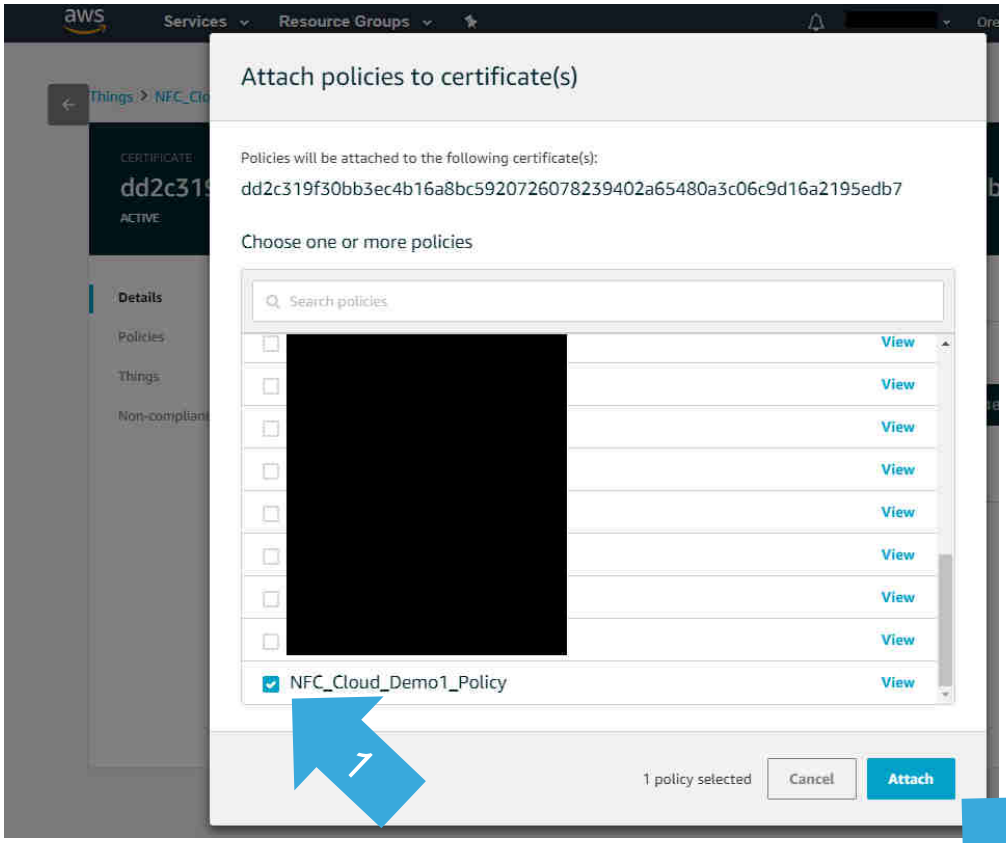
AWS Thing Creation



1) Click on Actions → Attach policy

life.augmented

AWS Thing Creation

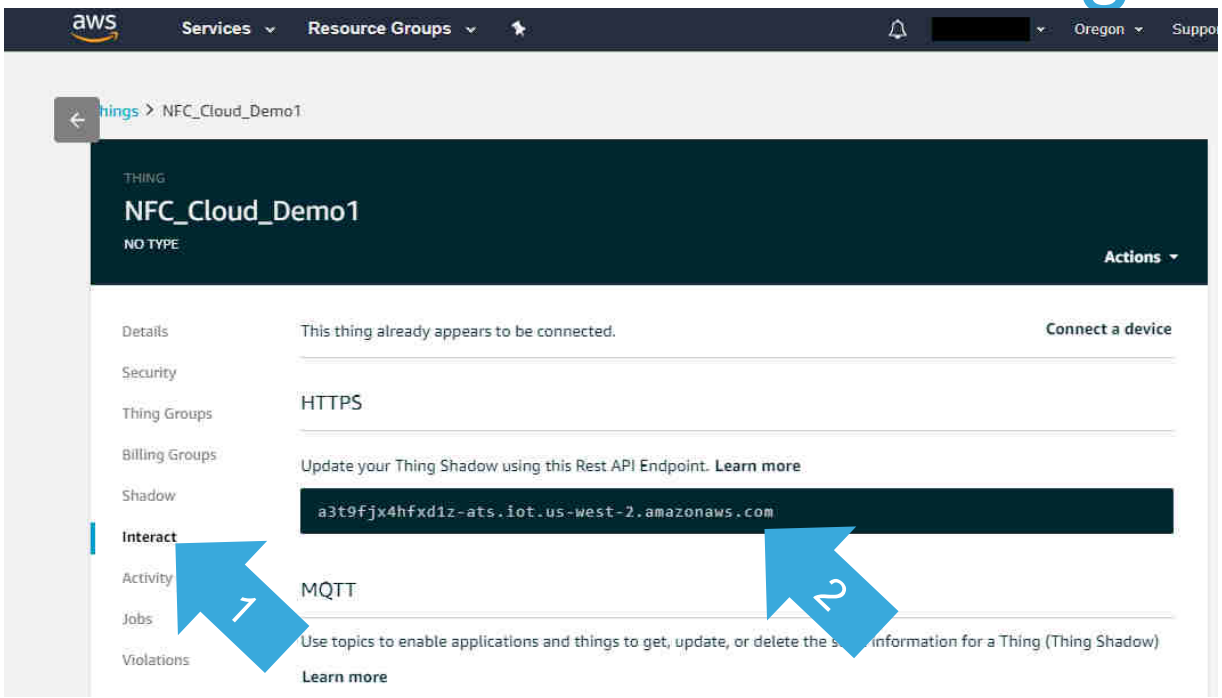


1) Select the newly-created policy

2) Click Attach

life.augmented

AWS Thing Creation

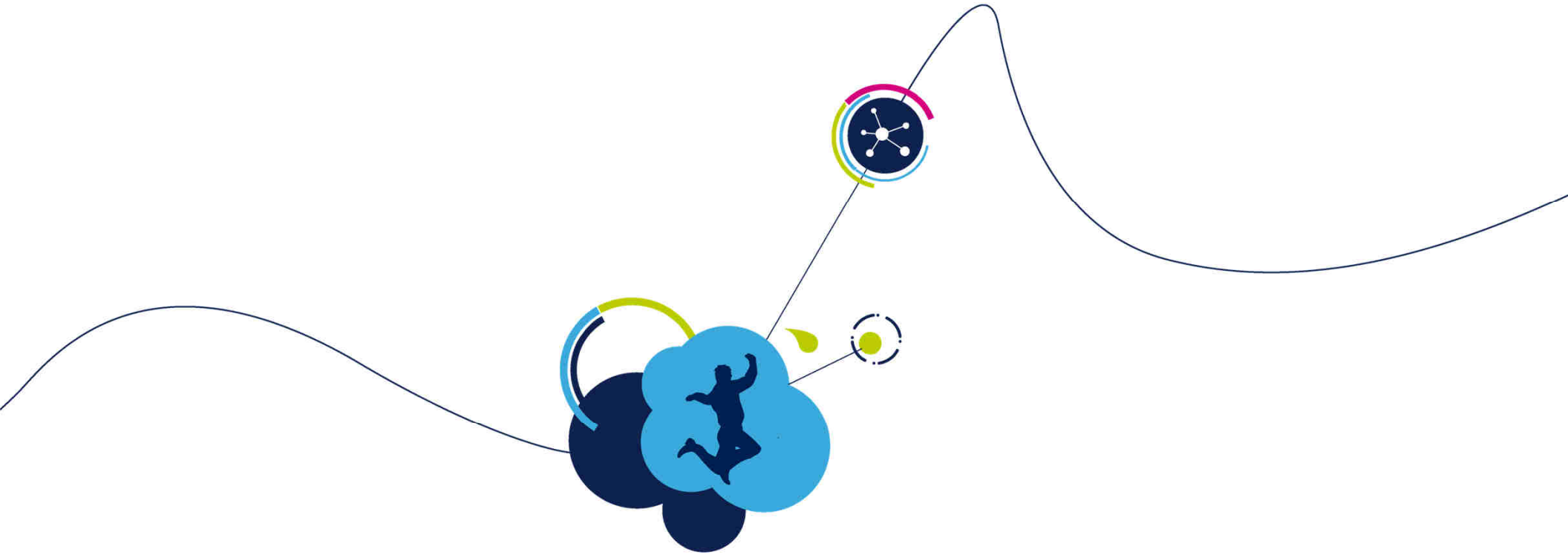


1) Go to the Interact tab

2) Note the endpoint URL

3) Use the endpoint URL along with the device certificate and private key to configure NFC application.

life.augmented



Thank You!

life.augmented