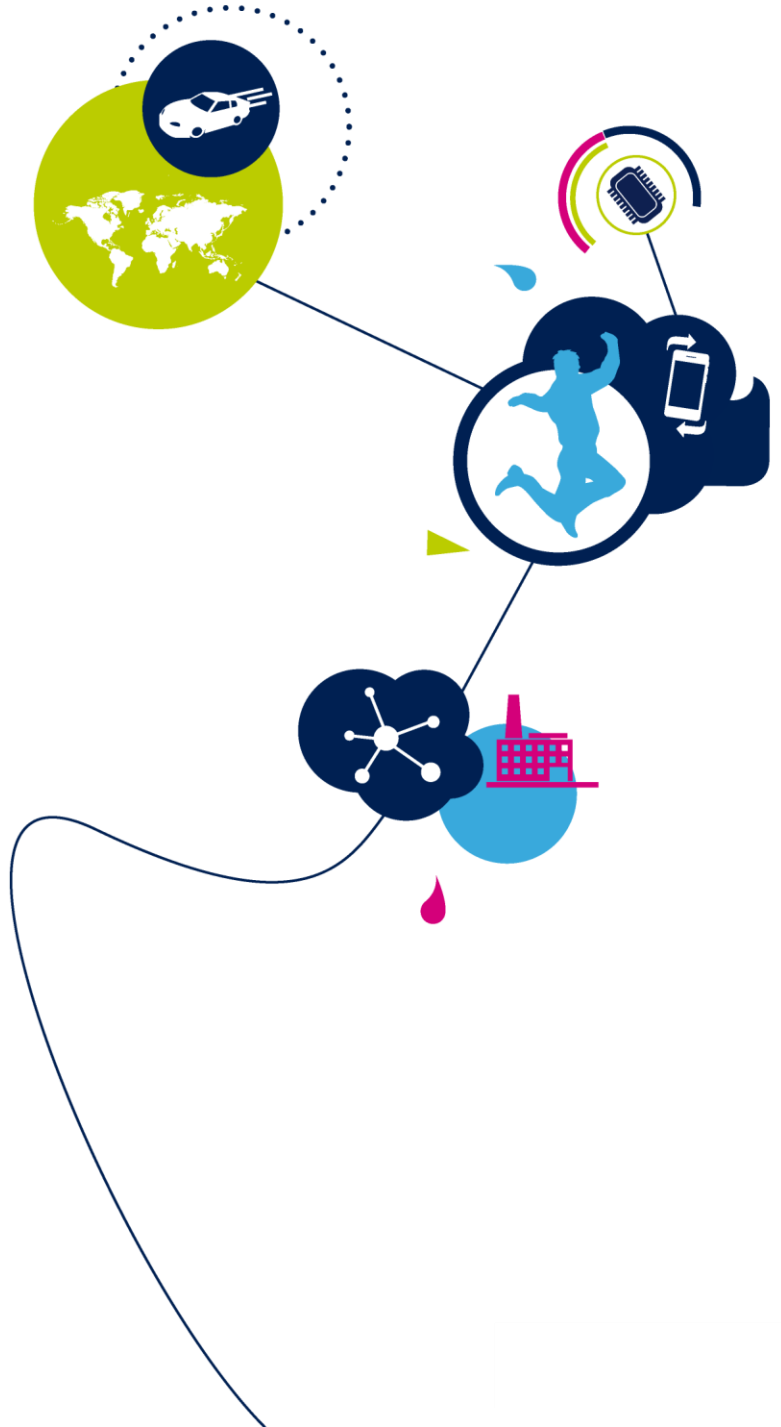
SensorTile.box 第三部分: 编程模式 (Pro mode)介绍



life.augmented

编程模式

SensorTile.box 具备三种工作模式

初级入门模式

应用预定义应用探索传感器功能

专家模式

配置和定义新的应用

编程模式

应用开发环境开发特定功能

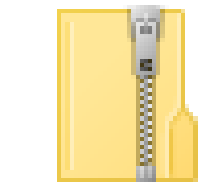
使用STM32 ODE进行编程

life.augmented

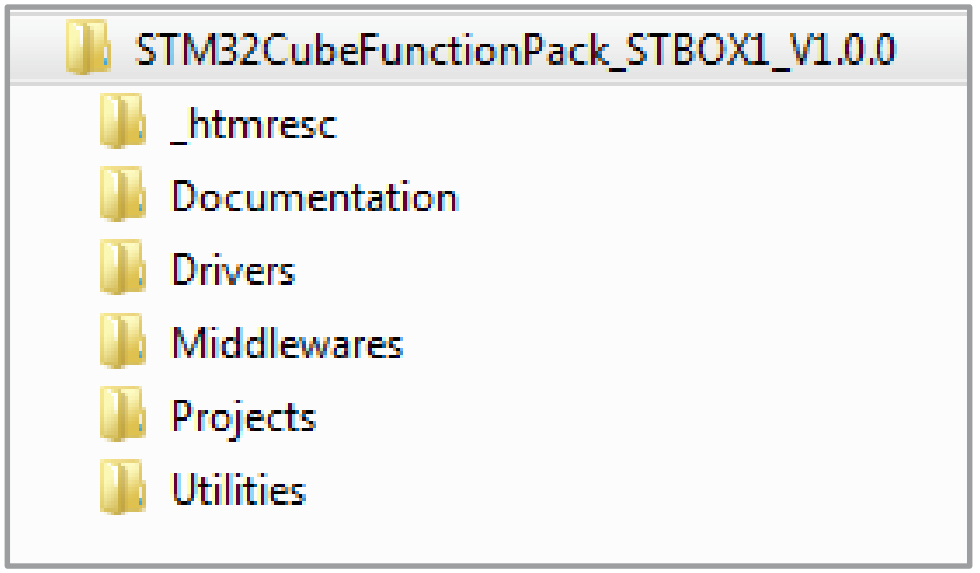
安装FP-SNS-STBOX1

STM32Cube功能包

- 几个应用程序示例的Zip文件
- 目录结构嵌套很深，文件夹的名称很长
- 建议在C:\TEMP或类似的短路径中解压缩它，以避免超过最大路径长度。

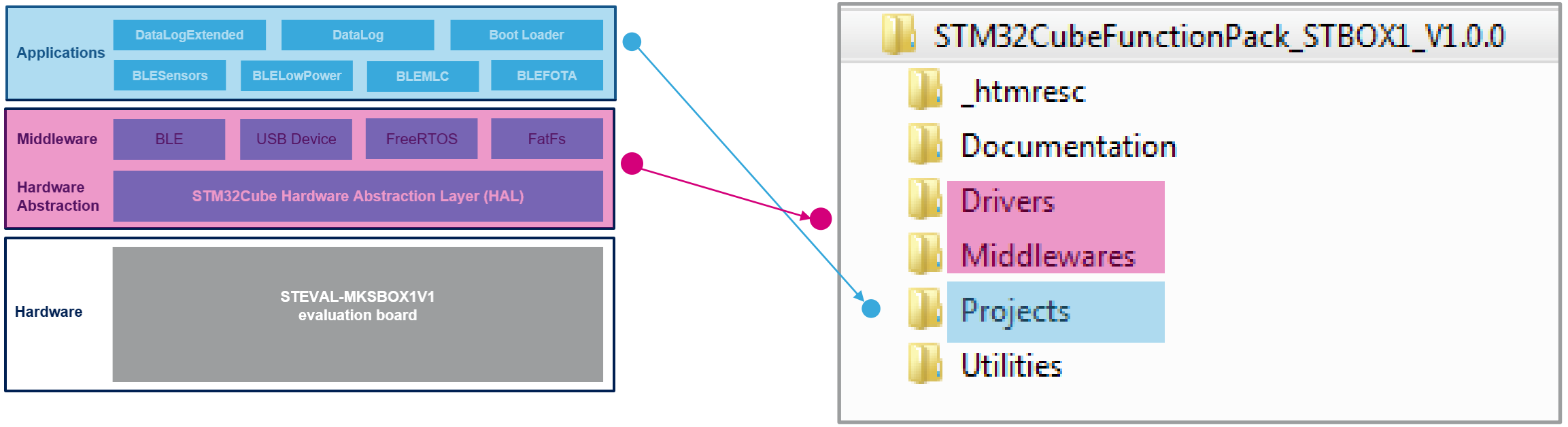


FP-SNS-ST
BOX1_V1.0.
0..zip



FP-SNS-STBOX1

STM32Cube功能包



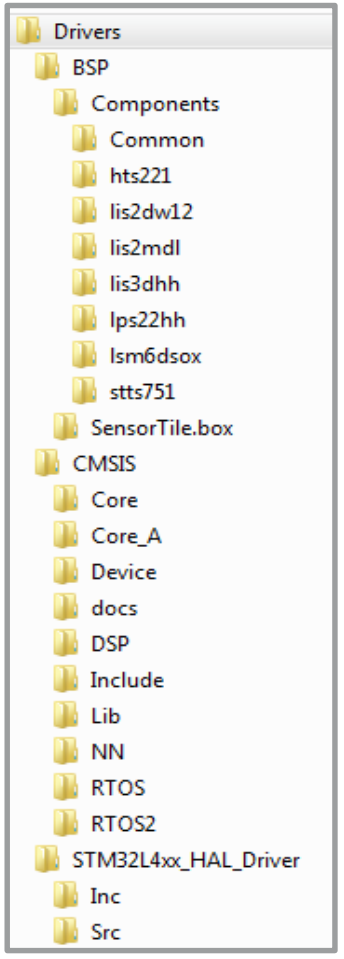
www.st.com上提供的最新信息

FP-SNS-STBOX1

STM32Cube功能包

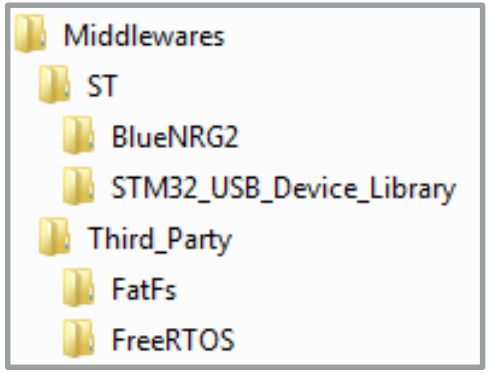
驱动

- BSP 板级支持包
  - 器件 (MEMS)
  - 板子 (SensorTile.Box)
- CMSIS(Cortex MCU 软件接口Std)
  - DSP Digital Sig Proc.
  - NN Neural Net
  - RTOS
- HAL 硬件抽象层



中间件

- BLE 蓝牙低功耗协议栈
- USBD 设备库
- FatFS 文件系统
- FreeRTOS 实时操作系统



life.augmented

FP-SNS-STBOX1

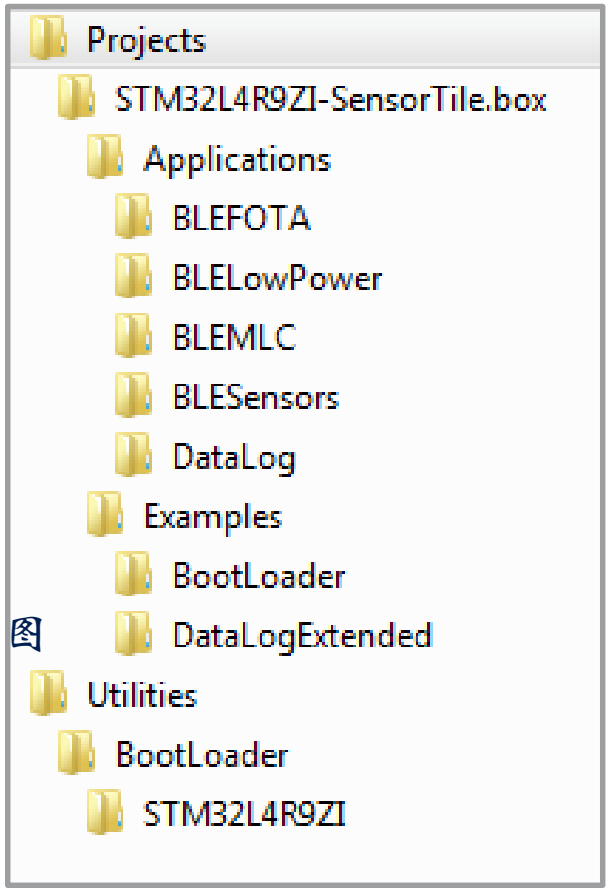
应用

- **DataLog** 以最大的速度保存到SD卡 w/RTOS
- **BLESensors** 抓取数据到ST BLE Sensor应用
- **BLELowPower**抓取数据到ST BLE Sensor应用 w/RTOS
- **BLEMLC** 机器学习核心演示(DecTree by **Unico GUI**)
- **BLEFOTA** 固件空中升级

示例

- **Bootloader** 引导加载持续以实现 BLEFOTA 功能
- **DataLogExtended**使用USB虚拟串口将传感器数据传送到PC上的Unicleo图形用户界面

- 程序
  - **Bootloader** 预编译二进制文件



实时操作系统(RTOS)

BLELowPower和**DataLog**: RTOS

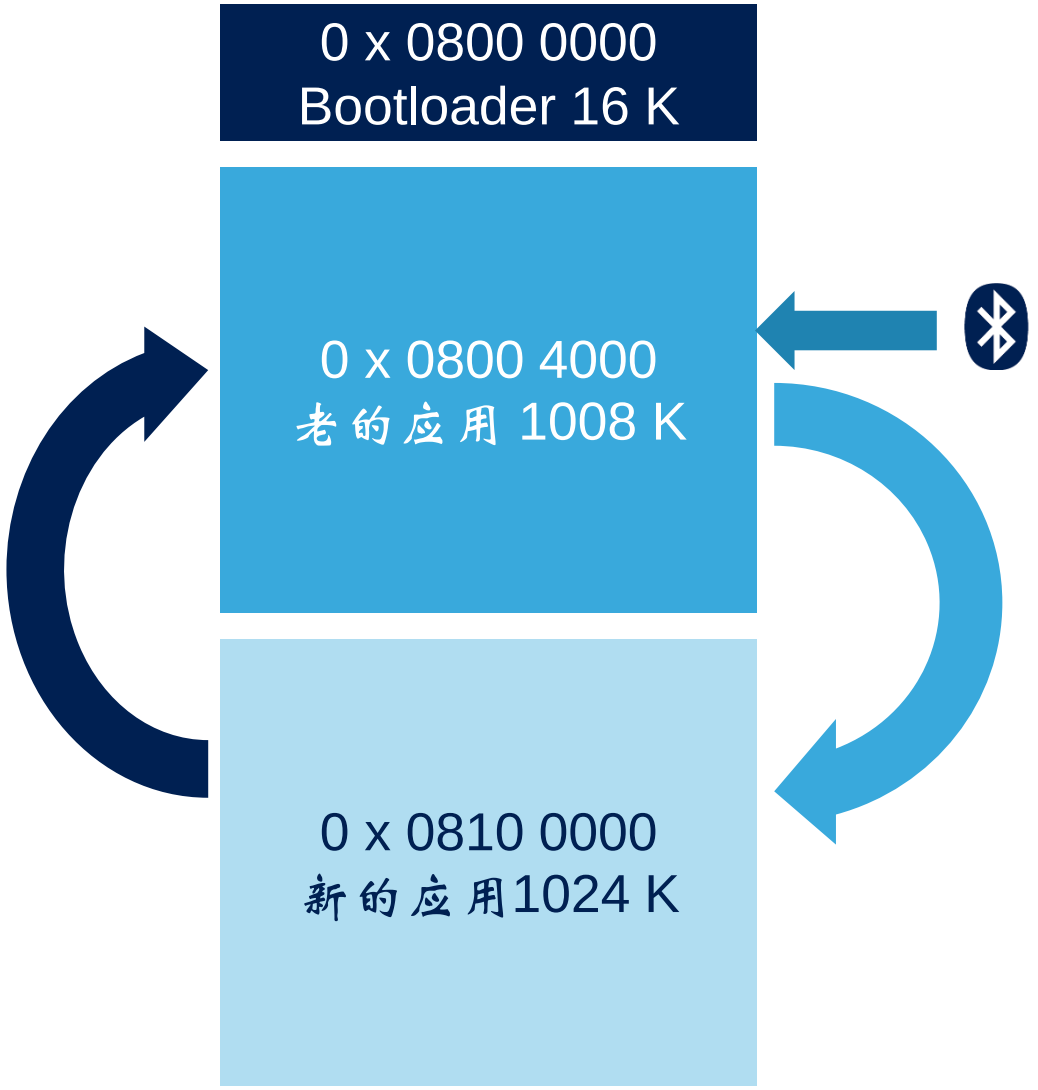
- 使能低功耗操作: 微控制器可以设置为睡眠时，没有任务调度执行。
- 使能**max speed**: 微控制器可以缓冲从传感器读取的数据，而SD卡上的操作写入仍在进行中

固件空中升级(FOTA)

BLEFOTA 应用和启动引导程序

- 0 x 0800 0000 启动引导程序
  - 如果新的应用是好的，用它覆盖现在的应用
  - 在0 x 0800 0000跑目前的应用
- 0 x 0800 4000 目前的应用(老的)
  - 通过BLE接受FOTA应用，然后写到

  0 x 0810 0000
- 0 x 0810 0000 FOTA 应用(新的)



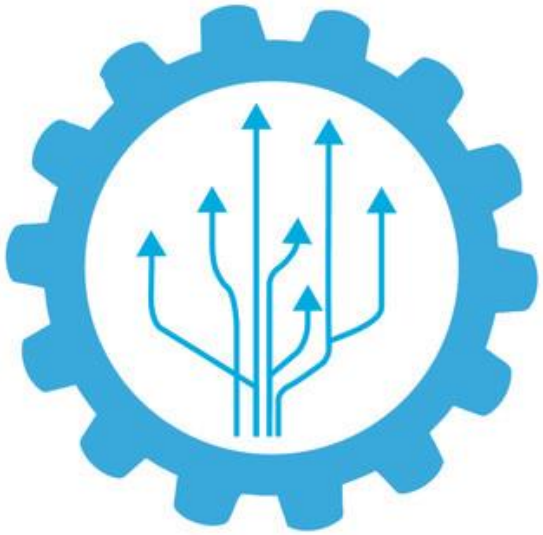
示例

人类活动识别算法

life.augmented

测试BLEMLC

LSM6DSOX惯性测量单元（IMU）中的机器学习核心（MLC）配置了用于人类活动识别的决策树



准备BLEMLC演示

二进制**SensorTileBox-BLEMLC.bin** 必须使用DFU(Direct Firmware Upgrade over USB)进行加载

逐步程序:

1. **运行ST BLE Sensor应用进入DFU模式**
   - 将应用程序连接到设备，打开调试控制台并发出“DFU”命令

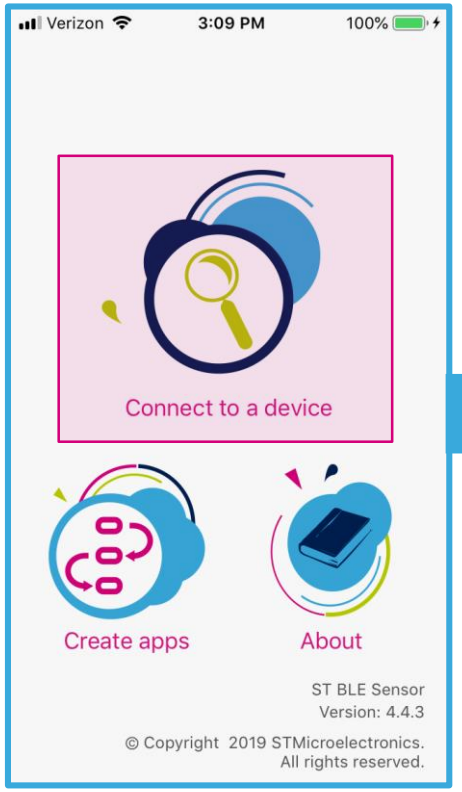
2. **运行STM32CubeProg去给器件进行编程**
   - 选择USB和“Connect”, “Mass Erase”，然后编写新二进制文件

3. **运行ST BLE Sensor应用去测试新的固件**
   - 将应用程序连接到您的设备，测试BLEMLC固件应用程序

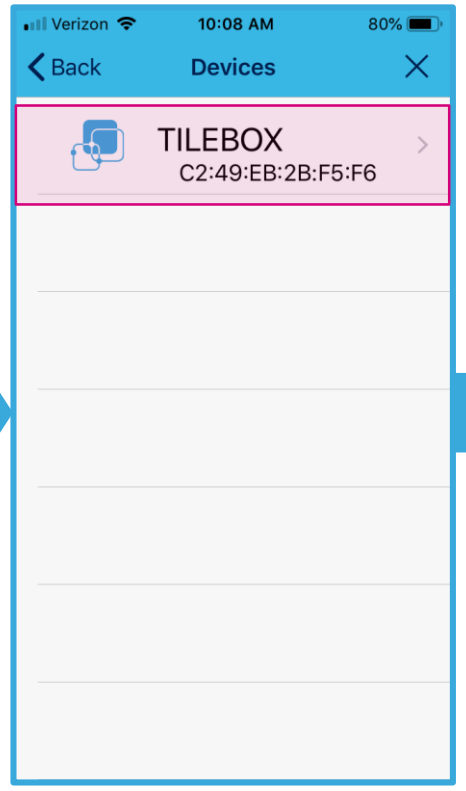
进入DFU模式

运行ST BLE Sensor应用

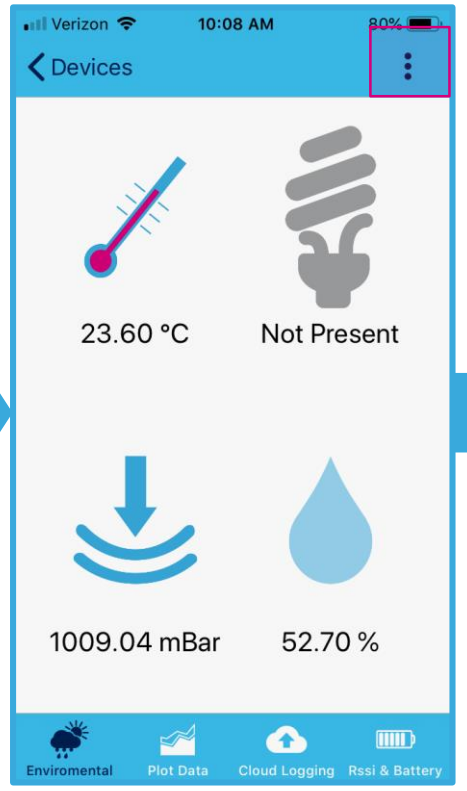
1. 点击 **Connect to a device**



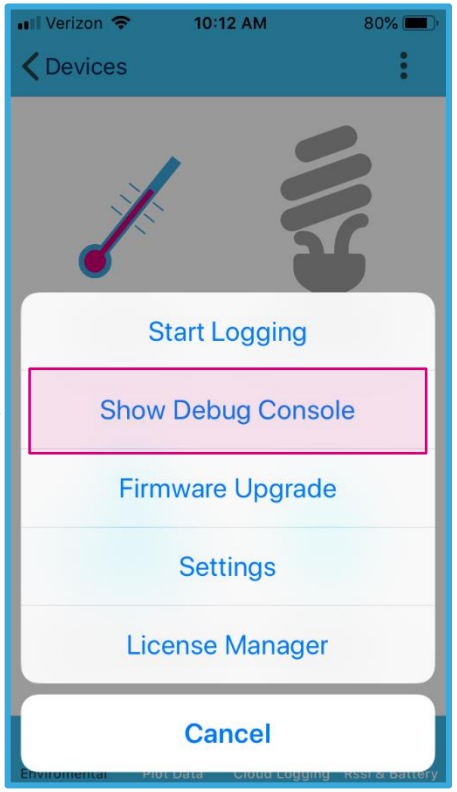
2. 选择你的 **SensorTile.box**



3. 在右上角点击按钮



4. 选择**Show Debug Console**

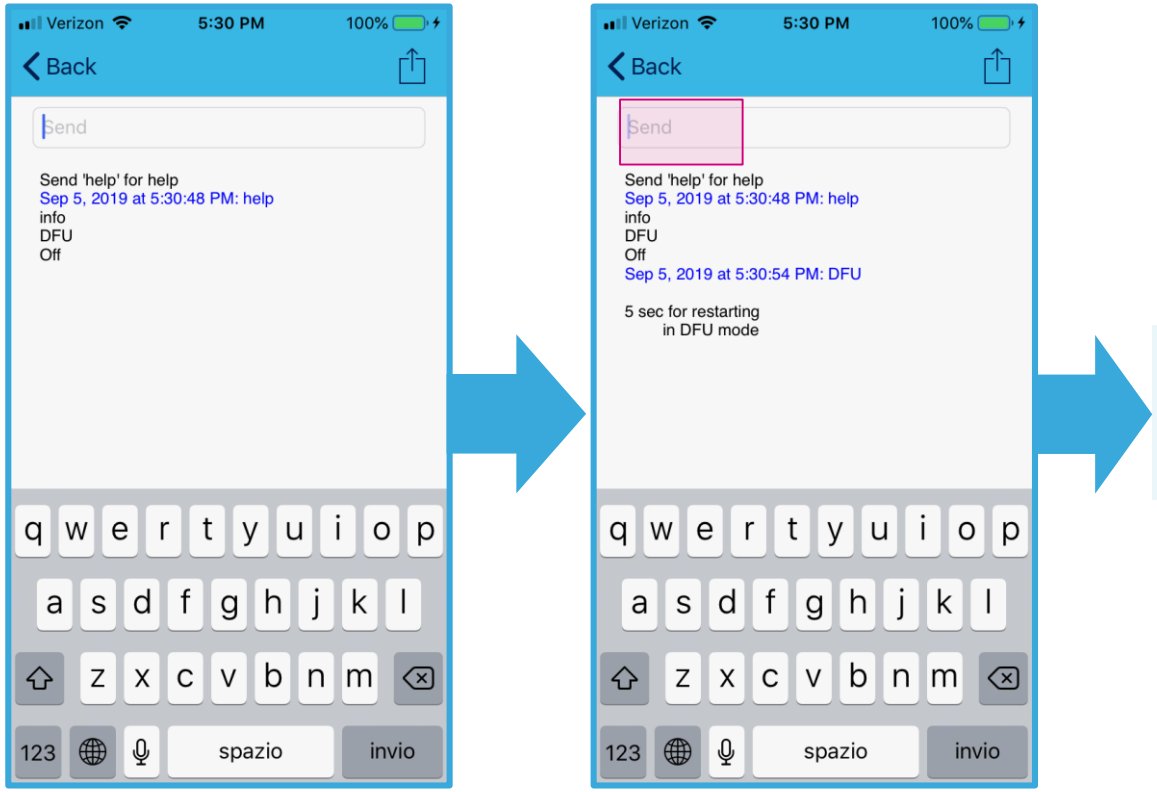


life.augmented

进入DFU直接固件升级模式

1. 输入“help”可以看命令的列表

2. 输入“DFU”(大写)

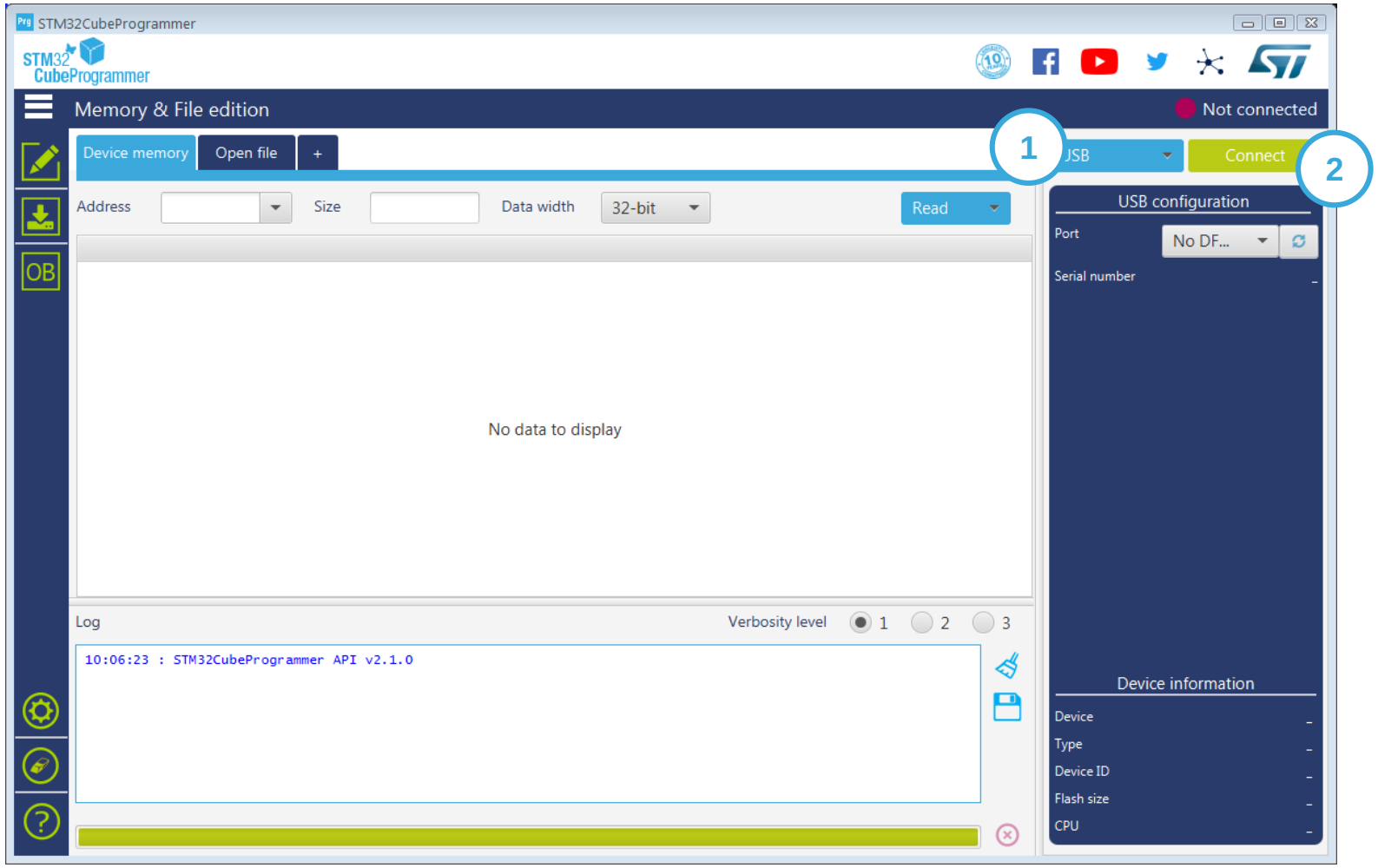


你的设备已经工作在DFU模式

烧写BLEMLC

运行STM32CubeProg

1. 选择 USB
2. 点击Connect

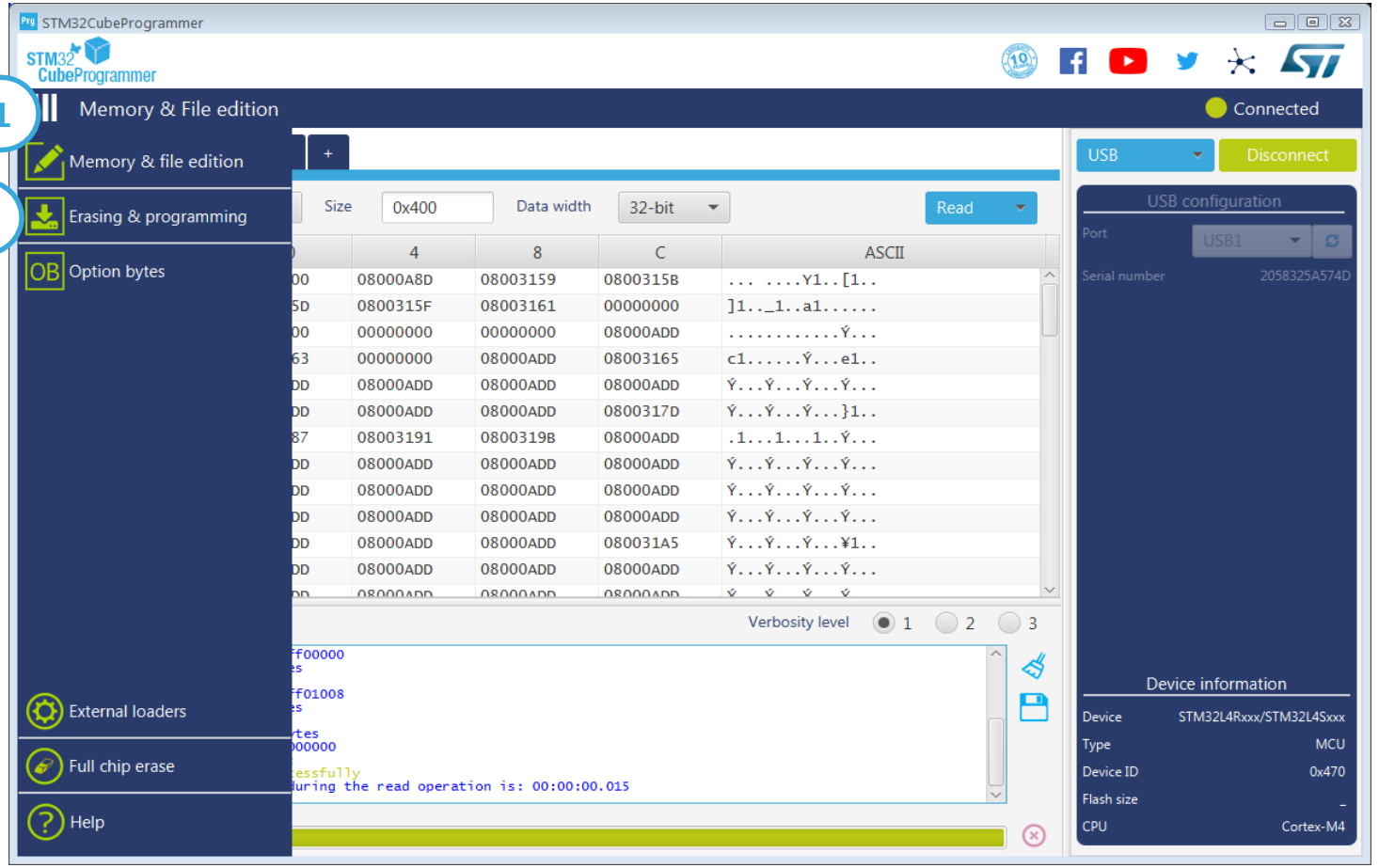


烧写BLEMLC

读取到目标微控制器的闪存

1. 点击Menu图标
   可以看到所有选项

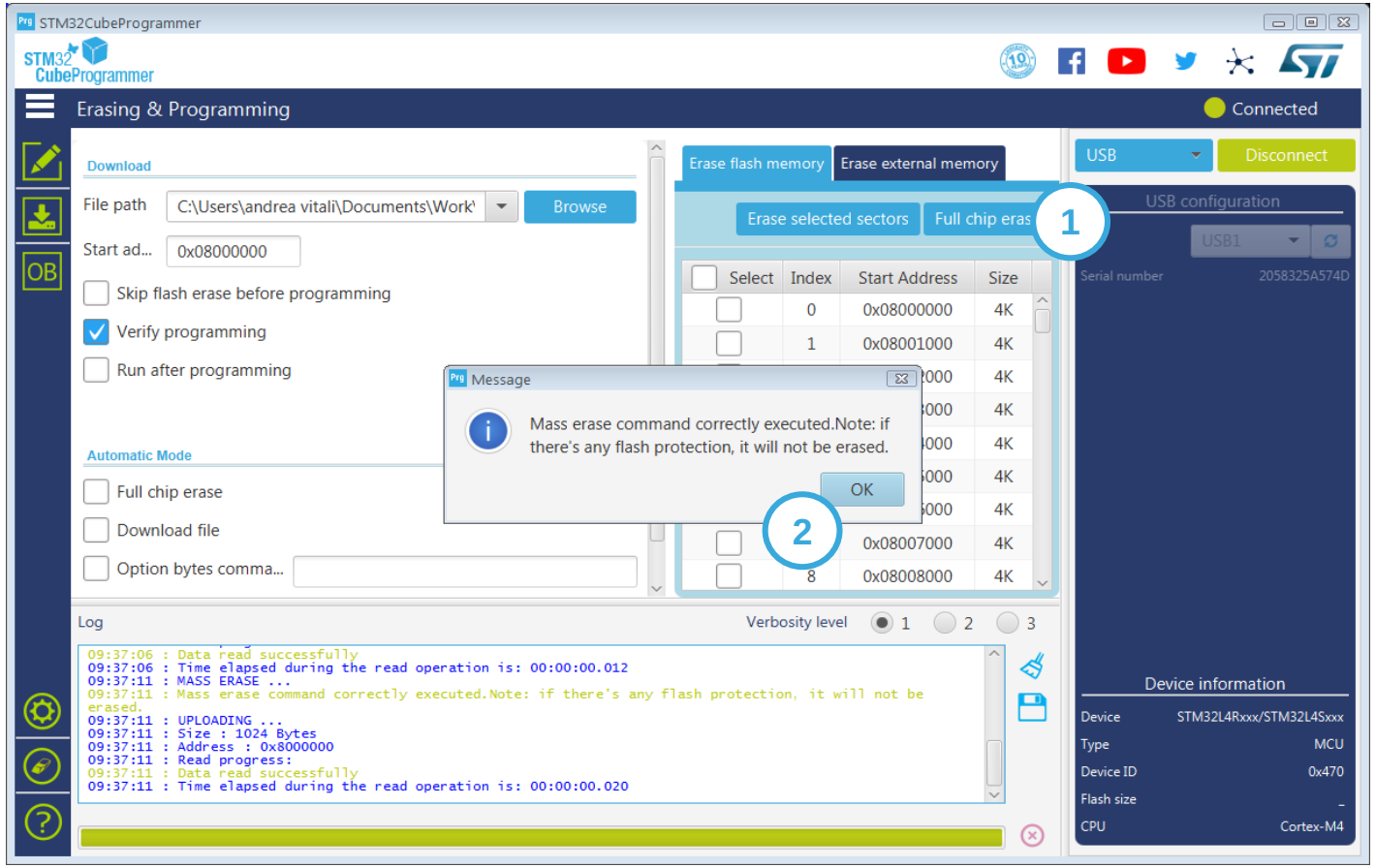
2. 选择**Erase & programming**



烧写BLEMLC

通过执行全芯片擦除进行清理

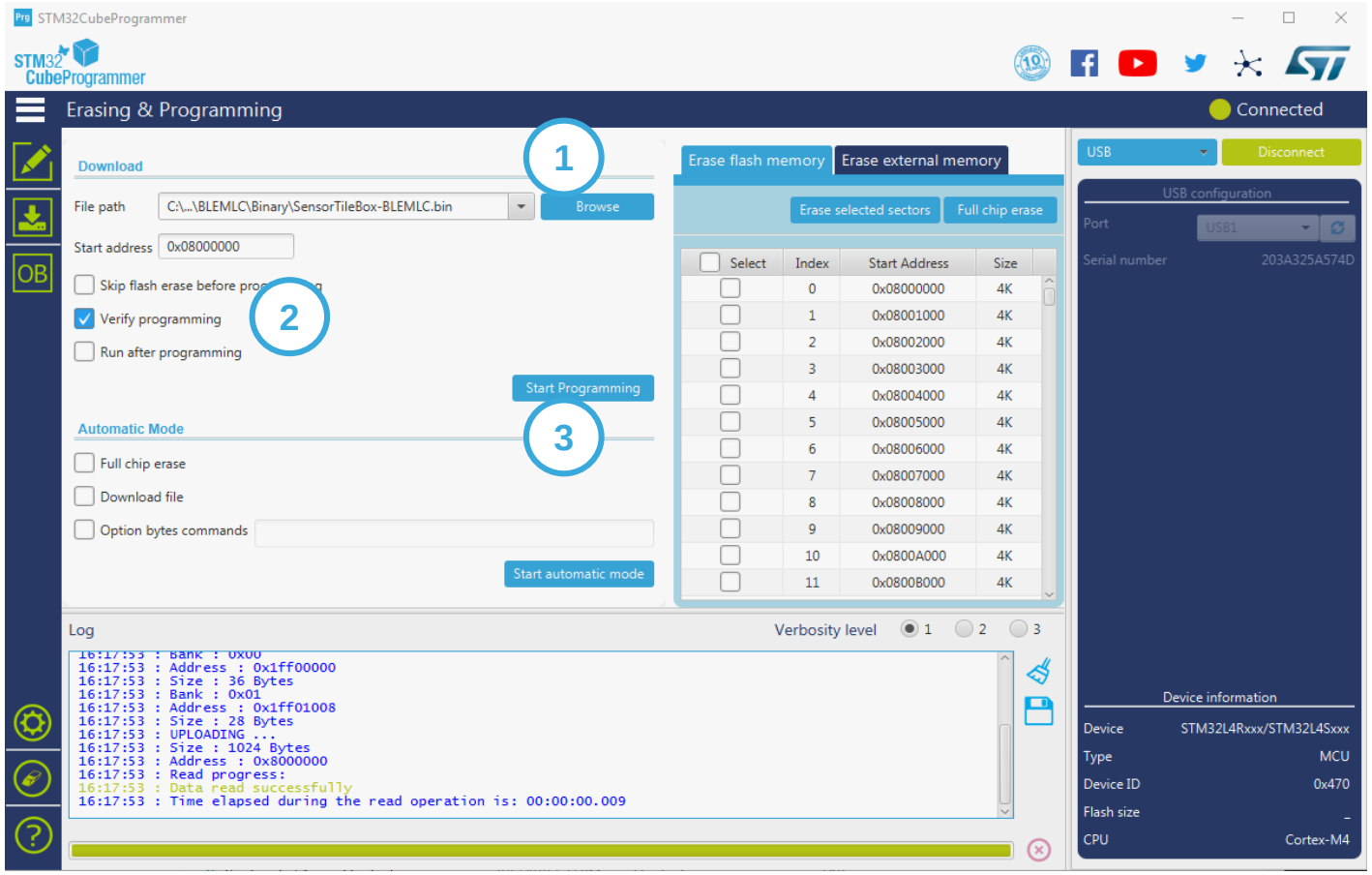
1. 选择**Full Chip Erase**
2. 确认 **OK**



烧写BLEMLC

烧写固件Projects/ STM32L4…/ Applications/ BLEMLC/ Binary/ *BLEMLC.bin

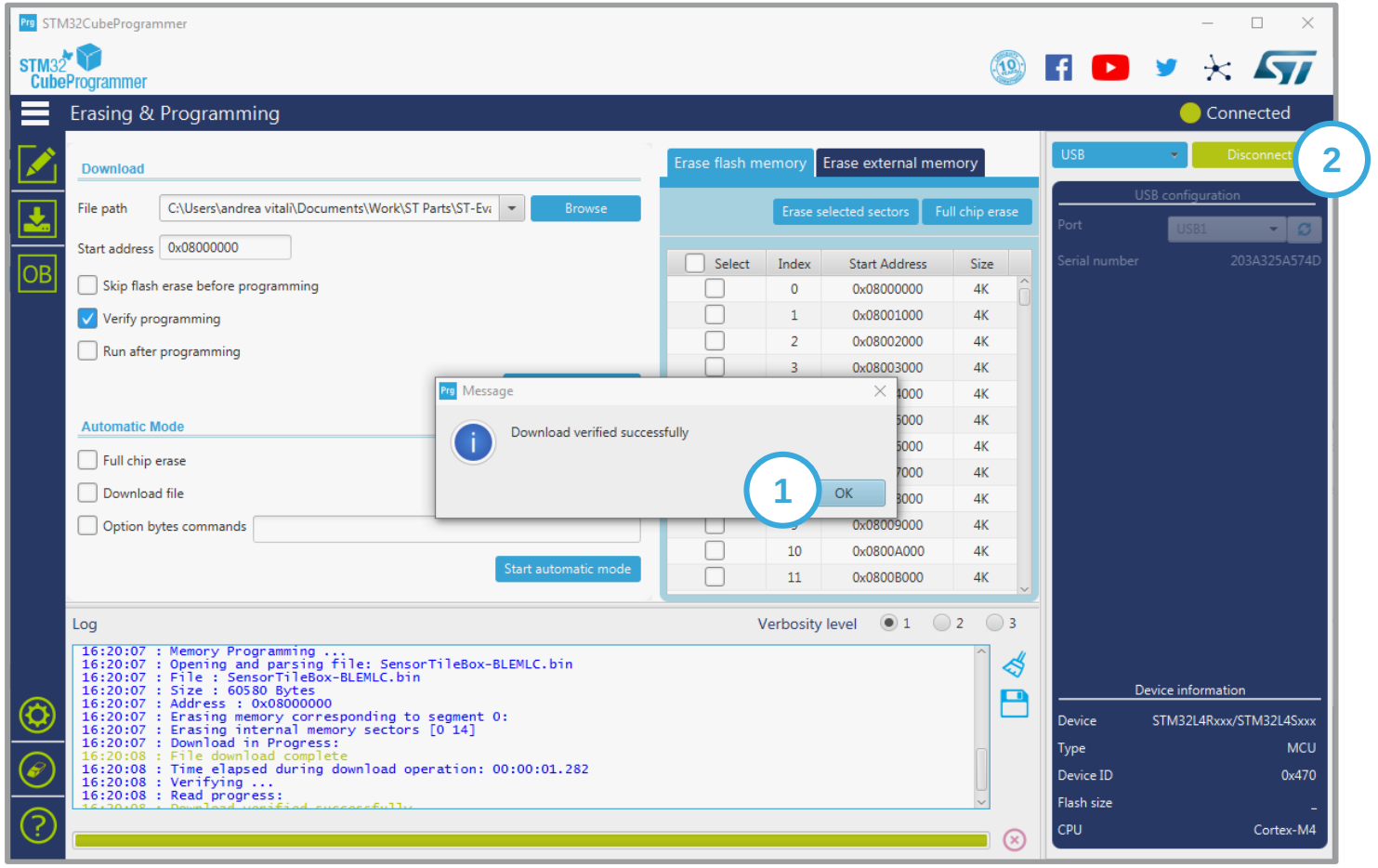
1. 点击Browse, 找到和选择 *BLEMLC.bin
2. 检查验证编程 (可选)
3. 点击Start Program



烧写BLEMLC

等待二进制文件加载并验证

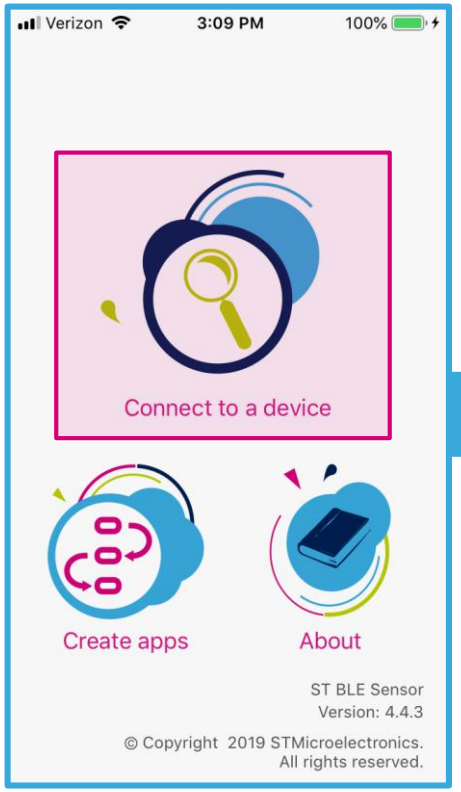
1. 点击 **OK**
2. 点击 **Disconnect**



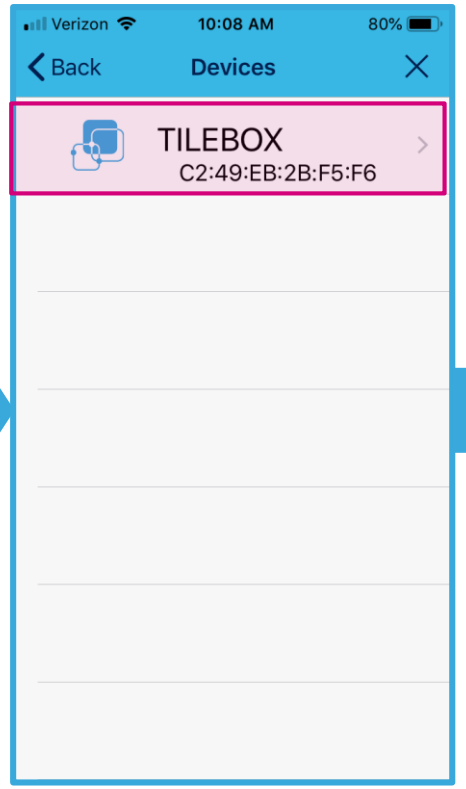
测试BLEMLC

运行ST BLE Sensor应用

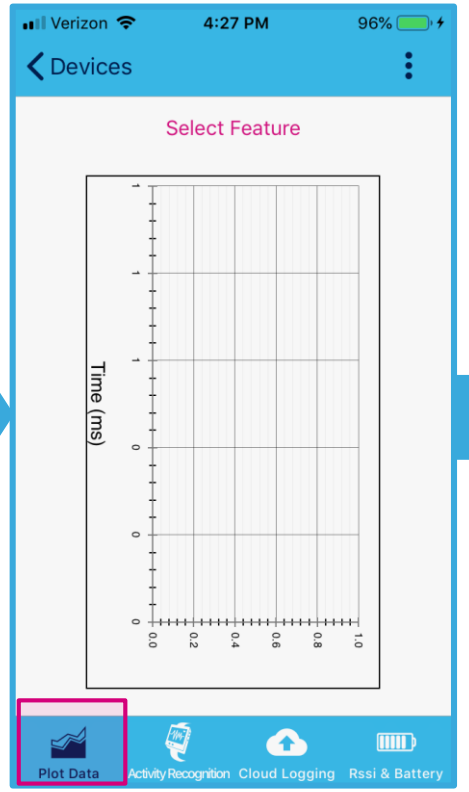
1. 点击按钮 **Connect to a device**



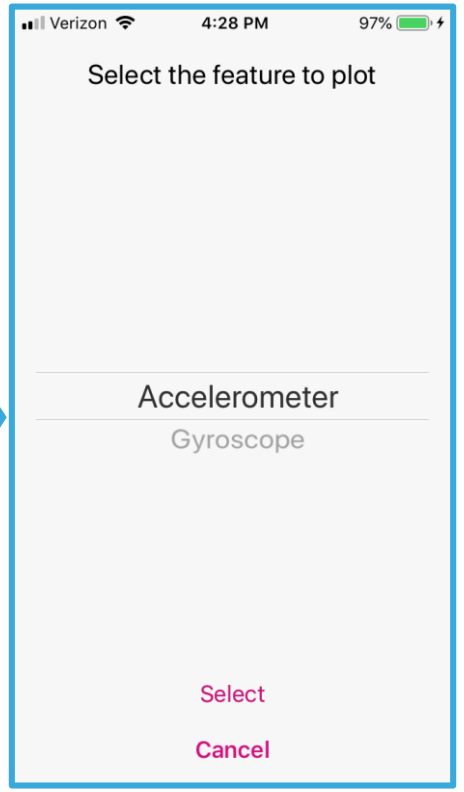
2. 选择你的 **SensorTile.Box**



3. 实时显示数据



4. 实时绘制数据

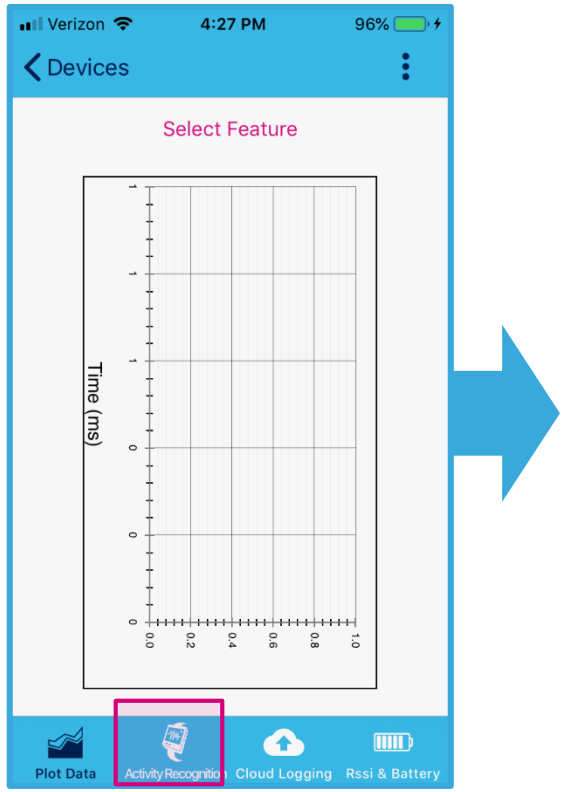


life.augmented

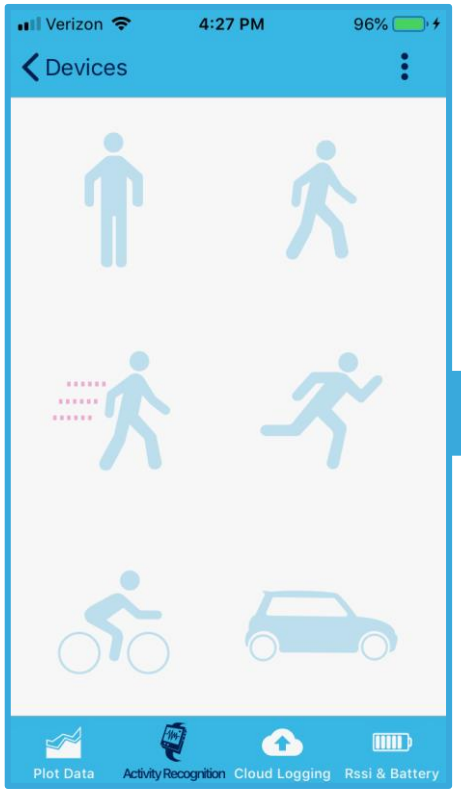
测试BLEMLC

运行ST BLE Sensor应用

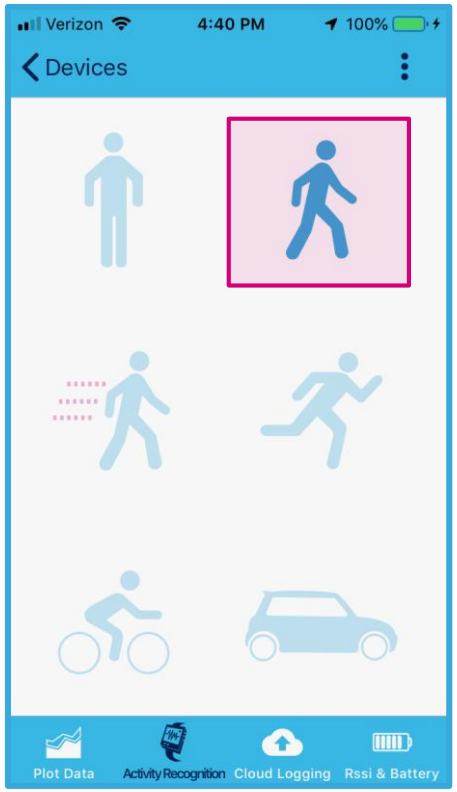
1. 选择
Activity Recognition



2. 以正常的节奏摇晃手机



3. 实时显示活动



life.augmented

BLEMLC 细节

life.augmented

BLE: 我们正在通过蓝牙传输到智能手机上运行的**ST BLE Sensor**应用程序

MLC: 我们使用的嵌入在LSM6DSOX IMU 中的**Machine Learning Core**
机器学习核心包括:

- 多个可编程二阶IIR滤波器
- **编程时间窗中的特征提取**
  (mean, variance, sum of squares, zero crossings, peak counters, min, max, peak to peak)
- 多个并发决策树。它们的输出可以发送到多个有限状态机，这些状态机可以由可配置的元分类器进一步处理

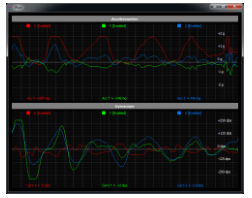
Unico GUI: 收集数据，配置和测试**Machine Learning Core**

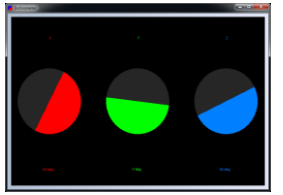
第三方工具用于设计**decision trees:** WEKA, RapidMiner, Python Scikit-Learn,
Matlab Statistics 和 Machine Learning Toolbox

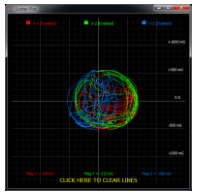
BLEMLC和Unico图形用户界面

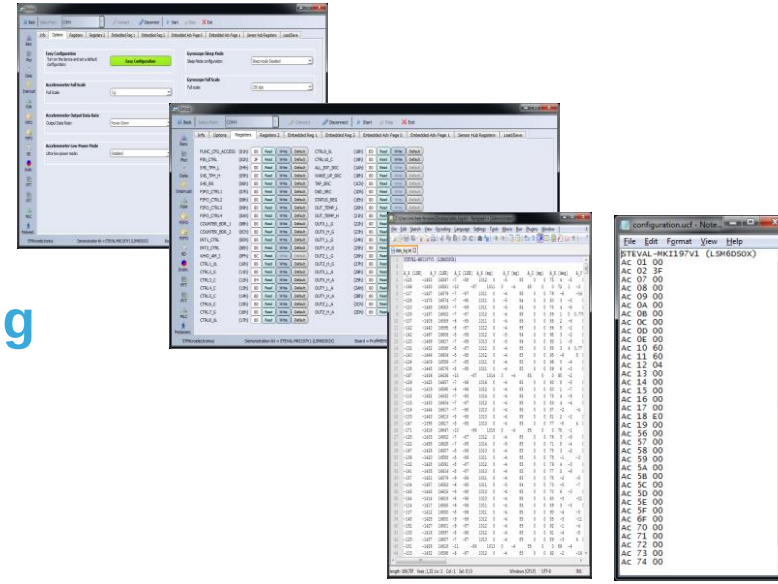
Unico GUI (图形用户界面)

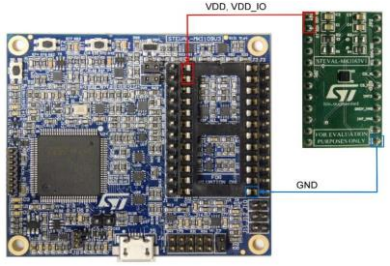
- 显示连接传感器的数据
  (time plot, scatter plot, 3D plot, FFT)
- 保存数据, 保存/加载 器件配置 (.ucf)
- 简单/默认配置, 基本配置(full scale,
  ODR, power mode) or reads/writes registers
- 高级配置**FSM (Finite State Machine)**, **MLC (Machine Learning Core)**, Pedometer
- 产生C 代码 (.h), 即使在传感器不在线的情况下
- 设置供电电压, I2C/SPI, 测量实时功耗
- 支持以下MEMS评估板
  - STEVAL-MKI109V2 “eMotion”
  - STEVAL-MKI109V3 “profiMEMS”
  - And DIL24 adapter of target MEMS

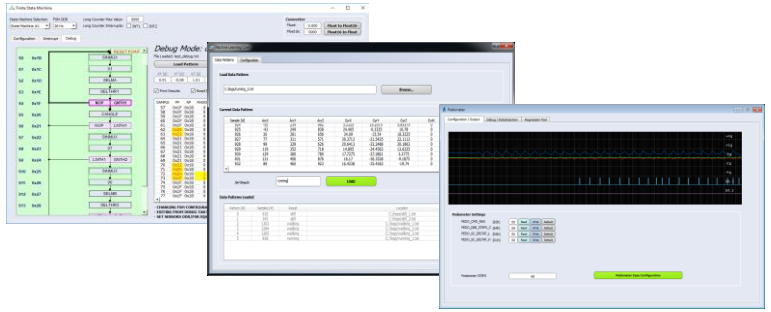






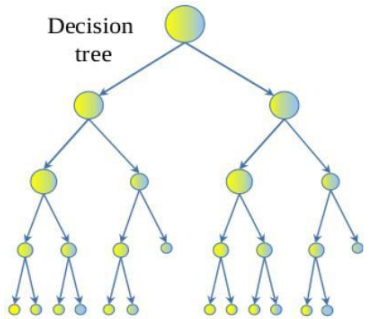


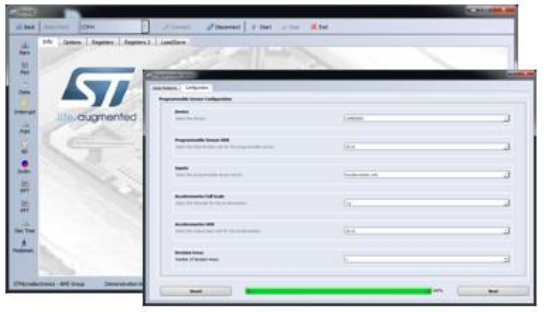




BLEMLC和Unico图形用户界面

1. 定义需要识别的类型
   - (e.g. 活动识别: 走, 跑, 开车)



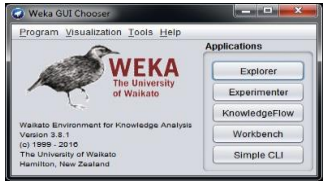


Unico

2. 给每一个类别采集多组数据
   - (e.g. 不同的人做同样的活动)

3. 离线数据分析:
   - 定义更好地描述所定义类的特征（一些统计参数，如：均值、方差、能量、峰-峰）
   - 使用WEKA等机器学习工具实现决策树（阈值、节点数、输出等）

4. 配置LSM6DSRX / LSM6DSOX IMUs
   - 决策树运行在器件中，减少MCU的功耗



Unico

通过机器学习工具创建决策树

决策树生成的外部工具

决策树可以由一个专用的机器学习工具生成，比如 WEKA

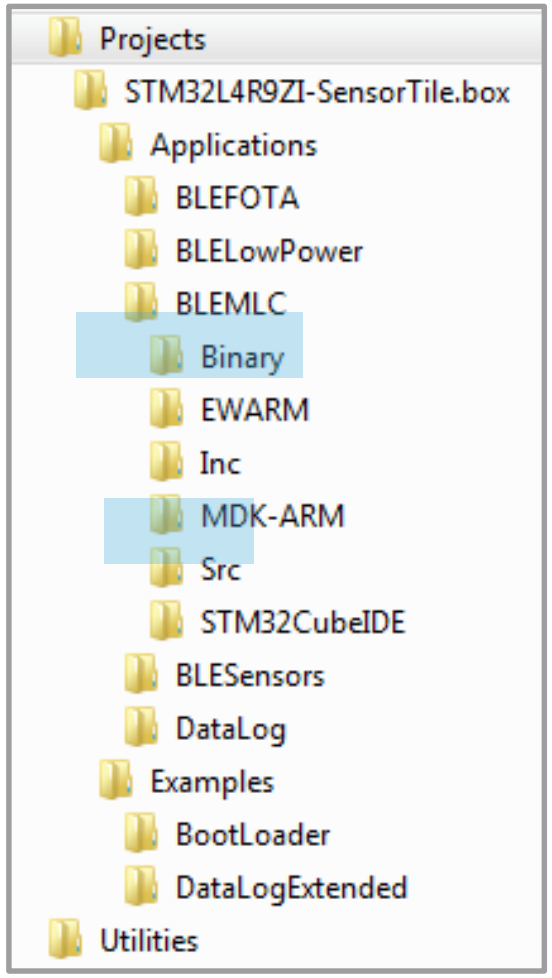
- 属性选择(从 .ARFF 文件)
- 数据过滤
- 决策树生成
- 决策树性能评估

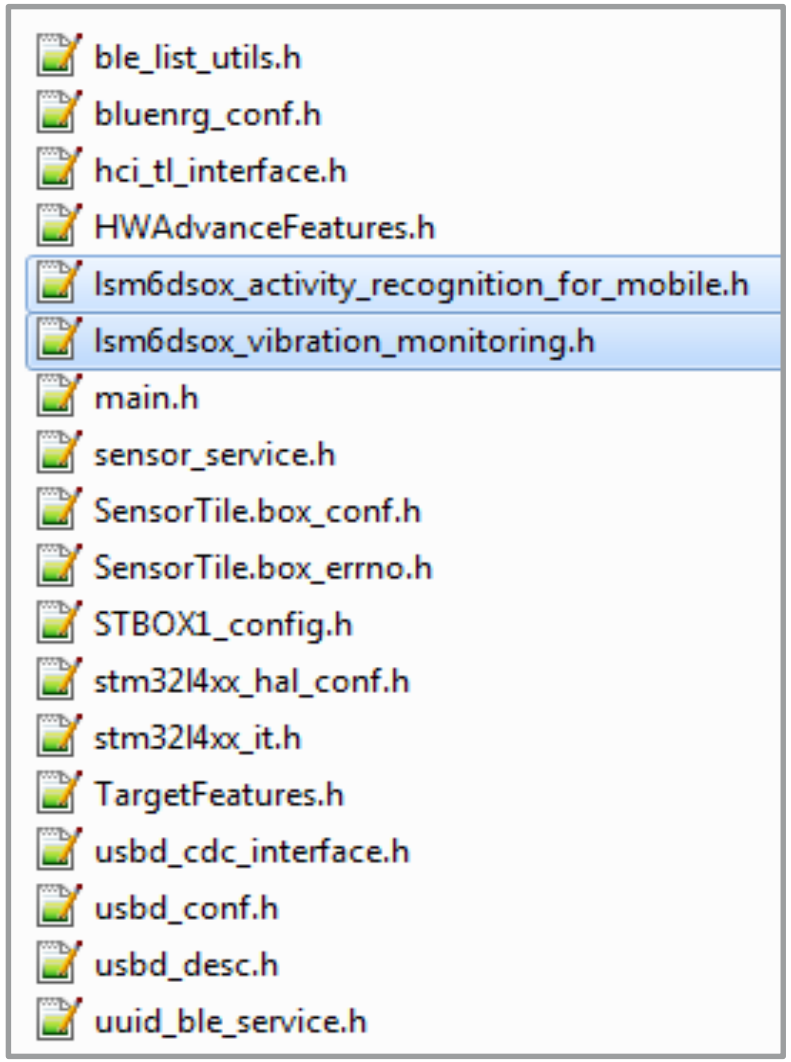
(e.g. 节点数量，精度，混合矩阵，etc…)

或者另外可替换的工具:

- RapidMiner → 在AN5259 rev2中提供了相关示例
- Matlab → 在Github上提供了相关脚本
  https://github.com/STMicroelectronics/STMems_Machine_Learning_Core/tree/master/tools/matlab
- Python →在Github上提供了相关脚本
  https://github.com/STMicroelectronics/STMems_Machine_Learning_Core/tree/master/tools/python

BLEMLC和Unico图形用户界面



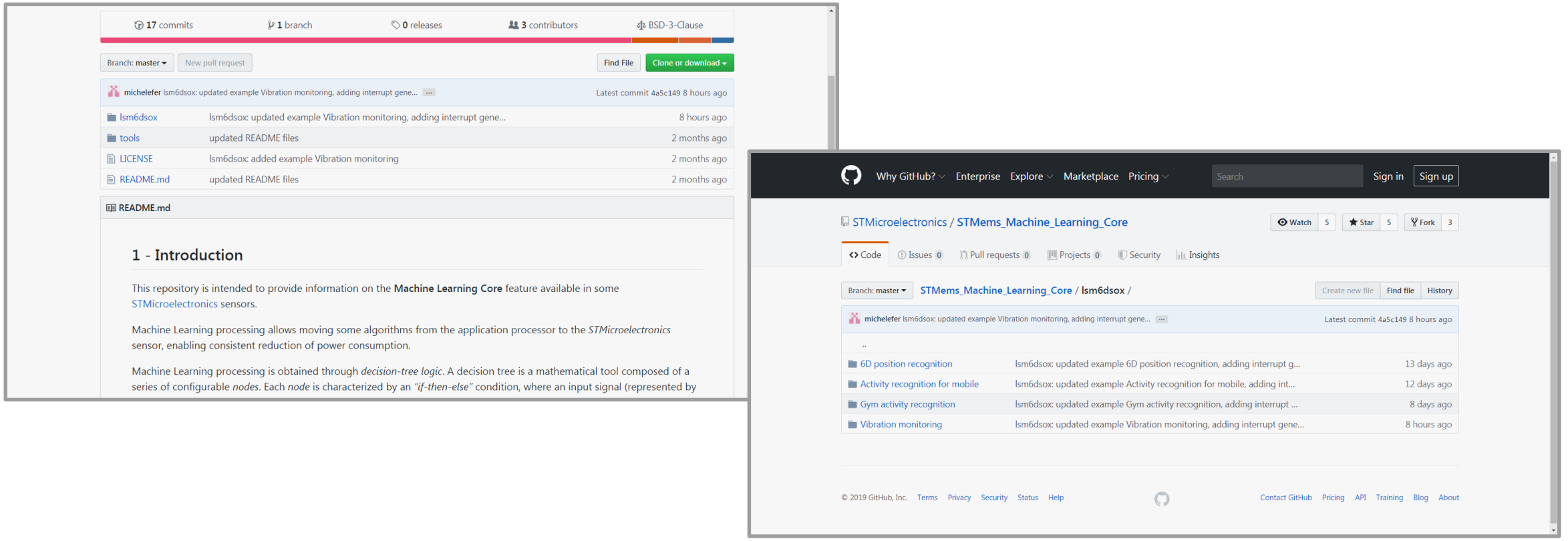



```
/** Common data block definition **/
typedef struct {
  uint8_t address;
  uint8_t data;
} ucf_line_t;

#endif /* MEMS_UCF_SHARED_TYPES */

/** Configuration array generated from Unico Tool **/
const ucf_line_t lsm6dsox_activity_recognition_for_mobile[] = {
  {.address = 0x10, .data = 0x00,},
  {.address = 0x11, .data = 0x00,},
  {.address = 0x01, .data = 0x80,},
  {.address = 0x05, .data = 0x00,},
  {.address = 0x17, .data = 0x40,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xEA,},
  {.address = 0x09, .data = 0x8C,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xEB,},
  {.address = 0x09, .data = 0x03,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xEC,},
  {.address = 0x09, .data = 0x9A,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xED,},
  {.address = 0x09, .data = 0x03,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xEE,},
  {.address = 0x09, .data = 0x00,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xEF,},
  {.address = 0x09, .data = 0x00,},
  {.address = 0x02, .data = 0x11,},
  {.address = 0x08, .data = 0xF0,},
```

BLEMLC和Unico图形用户界面



https://github.com/STMicroelectronics/STMems_Machine_Learning_Core

life.augmented

Thank you!

life.augmented