UNIVERSITY OF CALIFORNIA

Los Angeles

Internet-of-Things

Instructional Platform

for Electrical and Computer Engineering

A thesis submitted in partial satisfaction

of the requirements for the degree

Master of Science in Electrical and Computer Engineering

by

Xu Zhang

2019

ProQuest Number: 22624375

ProQuest 22624375

www.manaraa.com

© Copyright by

Xu Zhang

2019

ABSTRACT OF THE THESIS

Internet-of-Things

Instructional Platform

for Electrical and Computer Engineering

by

Xu Zhang

Master of Science in Electrical and Computer Engineering

University of California, Los Angeles, 2019

Professor William J. Kaiser, Chair

The motivation of this thesis is to design an interactive and scalable educational platform that delivers latest industrial technology and related electrical and computer engineering fundamentals to engineering students with inspiring and intuitive project examples. Current platform mainly demonstrates merged topics including Internet-of-Things (IoT) network, embedded system, data processing and computing, and machine learning.

The platform is composed of hardware system, software system, and curriculum system. The hardware system is consisted of STMicroelectronics SensorTile kits, STMicroelectronics Nucleo board, and Beaglebone, which provide different levels of IoT hardware components with reasonable costs. The software system is constituted with open-source integrated development

environment (IDE), System WorkBench, which is a professional tool to support bare-metal microcontroller programming. The curriculum system has series of tutorials and reference designs, which provide various intuitive projects with detailed instructions and programming guidance.

There are nine tutorials covering fundamental topics including firmware level programming, sensor system signal acquisition, motion sensing, audio sampling and signal processing, Bluetooth low energy (BLE), and inertial sensing. Reference design provides complete, end-to-end experience in development of a system. This experience prepares developers for innovation and implementation of new systems.

This platform is initiated through UCLA Engineering 96C course, which becomes one of the engineering course requirement, and UCLA ECE 180D course, which is the senior capstone design course. UCLA students have collaborated together and developed novel systems for motion classification with SensorTile data sources and machine learning methods developed. Those extraordinary projects are tidily compiled as student project reference designs to inspire other future developments. In addition, the platform is widely adopted by various universities including Columbia University, California State University – North Bridge, Georgia Tech, University of Maine, and so on.

The thesis of Xu Zhang is approved.

Chih-Kong Ken Yang

Gregory J. Pottie

William J. Kaiser, Committee Chair

University of California, Los Angeles

2019

iv

For my parents and my beloved Annie,

who always support me, encourage me and love me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ADC	Analog to Digital Converter
ATT	Attribute Protocol
BLE	Bluetooth Low Energy
CLI	Command Line Interface
DAC	Digital to Analog Converter
DC	Direct Current
DoF	Degree of Freedom
DSP	Digital Signal Processing
FS	Full Scale
FSM	Finite State Machine
GATT	Generic Attributes
GCC	GNU C Compiler
GPIO	General Purpose Input Output
GUI	Graphical User Interface
HAL	Hardware Abstract Layer
I2C	Inter-integrated Circuit
IC	Integrated Circuit
IDE	Integrated Development Environment
IIR	Infinite Impulse Response
IoT	Internet-of-Things
ML	Machine Learning
MEMS	Microelectromechanical Systems
ODR	Output Data Rate
PCM	Pulse Coded Modulation
PDM	Pulse Density Modulation
SPI	Serial Peripheral Interface
SWD	Serial Wire Debug
USB	Universal Serial Bus
UUID	Universally Unique Identifier

ACKNOWLEDGEMENTS

Chapter 1

Introduction

IoT, embedded computation, and machine learning (ML) are becoming the most important and critical state-of-art technologies that enable enormous data acquisition and data mining capabilities in variety of fields to deliver end-to-end system solutions. The rapid processor development provides significantly improved computing power that makes it possible to merge data acquisition, processing and computation on embedded devices. However, we have found that the most significant challenge while learning embedded computing system is a roadblock which hinders progress due to a lack of guidance resources. This leads to frustration and a loss of motivation. Therefore, there is an urgent and strong demand on introducing an educational platform to help new engineers and students, who encounter this technology for the first time, make rapid progress, truly learn, and benefit from new inspiration and understanding of embedded computing and IoT at early stage.

In order to introduce those "advanced" topics with a straightforward method, we have determined to arrange the curriculum with sets of effortless tutorials and reference designs with SensorTile. SensorTile in this curriculum plays a critical role that exposes a super compact IoT hardware platform with an industrial standard. Meanwhile, the widely supported STM32 platform and open source professional IDE introduced in this curriculum, System WorkBench, provides the opportunities to truly engage in bare-metal programming and in-depth embedded development instead of high-level abstract "toy" project. To maximize the learning experience regardless of students' background, each tutorial is compiled with very specific programming guidance, hardware assembling instruction, and graphical illustrations to present an interesting and practical project that has comprehensive outcomes. In this way, students can establish

1

confidence through the learning progress because the course is not only intuitive but fun as it encourages experimentation. The reference design on the other hand provides complete, end-to-end experience in development of a system. By using reference design as a template, students are able to innovate and implement their own new systems rapidly.

This educational platform is not only designed for electrical and computer engineering students but also for all engineering students who are willing to engage in the field. Now, it is crucial to have interdisciplinary knowledge to innovate an engineering solution. Therefore, the topics introduced in the tutorials primarily include motion sensing, audio sampling, signal processing, and Bluetooth Low Energy communication, which are frequently used fundamentals across multiple engineering disciplines like mechanical engineering, aerospace engineering, and biomedical engineering. Through tutorial and reference design, this proposed educational platform helps students in all engineering fields focus on rapid start in computing and it also creates excitement and enthusiasm for engineering required to sustain student through their early stage learning.

The last motivation of this educational platform is to maintain course technology platform advances every year that adhere and advance best principles. This requires an academic community that can continuously create expanding shared educational resources in IoT. The community will benefit from SensorTile IoT vision and its compelling nature and accessibility to students. Through dedicated support of STMicroelectronics, the educational community is well established with both educator community sharing approaches and instructional resources and student community including student forums and student project reference designs. In the end of this thesis, community achievements including student project reference design and curriculum adoption will be discussed.

Chapter 2

System Overview

This chapter provides a system overview of the educational IoT platform which is composed of hardware system, software system, and IoT curriculum. It first introduces the hardware system, STMicroelectronic SensorTile development kit, which is the main IoT module used in the platform. It then introduces the software system, System WorkBench, which is an open source professional integrated development environment. Eventually, this chapter briefly illustrates the IoT curriculum including tutorials and reference designs.

2.1 Hardware System

This section introduces the full hardware system including on-board sensors of SensorTile, SensorTile cradle boards, and debugging interface Nucleo board.

2.1.1 SensorTile

SensorTile (STEVAL-STLCS01V1) is a new IoT system provided by STMicroelectronics integrating state-of-the-art processor, wireless interfaces, and sensor systems. It is a tiny, square-shaped IoT module that packs powerful processing capabilities leveraging an 80 MHz STM32L476JGY microcontroller and Bluetooth low energy connectivity based on BlueNRG-MS network processor as well as a wide spectrum of motion and environmental microelectromechanical systems (MEMS) sensors, including a digital microphone [1]. It can form the foundation for wearable consumer devices, wearable medical devices, residential IoT systems and vehicle IoT systems. The SensorTile system provides an exceptionally powerful and well-supported platform for introduction to IoT technology.

3

The SensorTile is remarkably compact as shown in Figure 2.1, which occupies a very small 13.5x13.5 mm square outline, with all the electronic components on the top side and small connector on the bottom side to plug it onto the cradle expansion board.
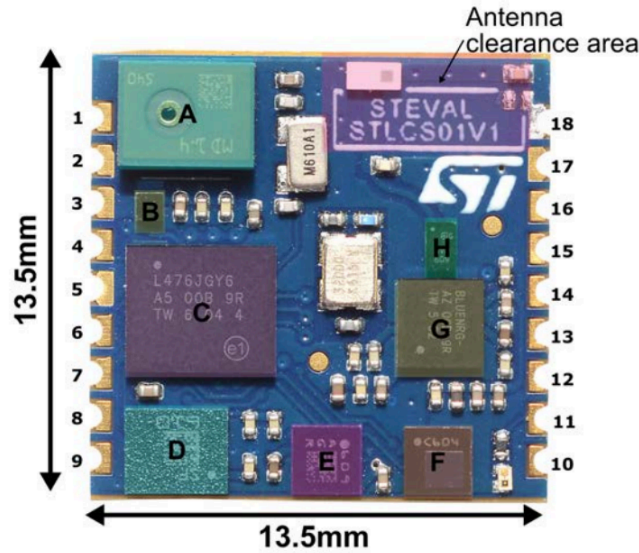
Figure 2.1: SensorTile main components [2]

Reference	Device	Description
A	MP34DT05-A	MEMS audio sensor digital microphone
B	LD39115J18R	150 mA low quiescent current low noise LDO 1.8 V
C	STM32L476 MCU	ARM Cortex-M4 32-bit microcontroller
D	LSM6DSM	iNEMO inertial module: low-power 3D accelerometer and 3D gyroscope
E	LSM303AGR	Ultra-compact high-performance eCompass module: ultra-low power 3D accelerometer and 3D magnetometer
F	LPS22HB	MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer
G	BlueNRG-MS	Bluetooth low energy network processor
H	BALF-NRG-02D3	50 Ω balun with integrated harmonic filter

Table 2.1: SensorTile main components descriptions [2]

Table 2.1 provides the main board components description corresponding to the reference area shown in Figure 2.1. The SensorTile's main microcontroller communicates with motion

4

sensors, barometer, Bluetooth wireless processor through serial peripheral interface (SPI) and microphone through pulse density modulation (PDM). SensorTile functional block diagram is presented in Figure 2.2.
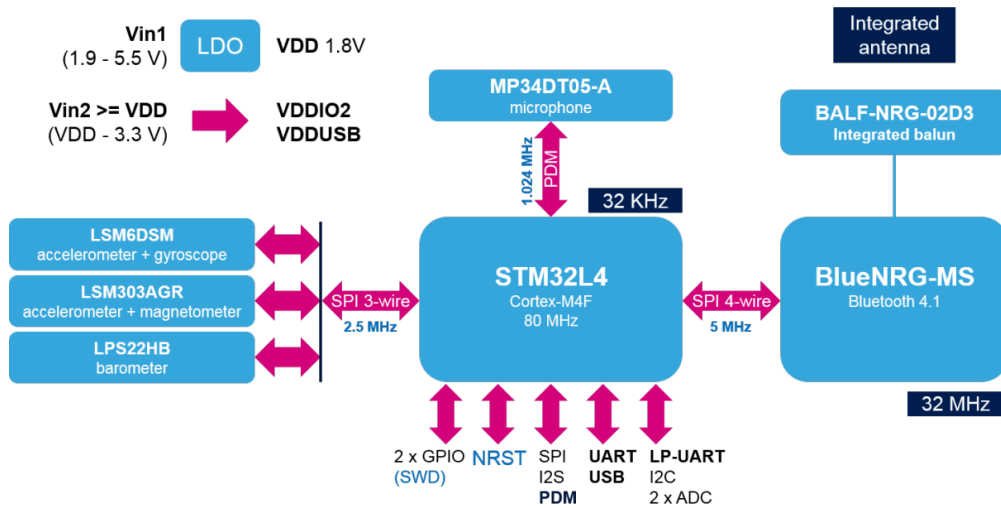
Figure 2.2: SensorTile function block diagram [2]

In addition to SensorTile, the development kit also includes two cradle boards including the expansion cradle board (STLCX01V1) and the cradle board (STLCR01V1). The expansion cradle board with Sensortile plug connector supports convenient mounting, 16-bit stereo audio DAC, 3.5 mm stereo audio jack and USB power supply as shown in Figure 2.3. The main components of expansion cradle board are presented in Table 2.2.

5
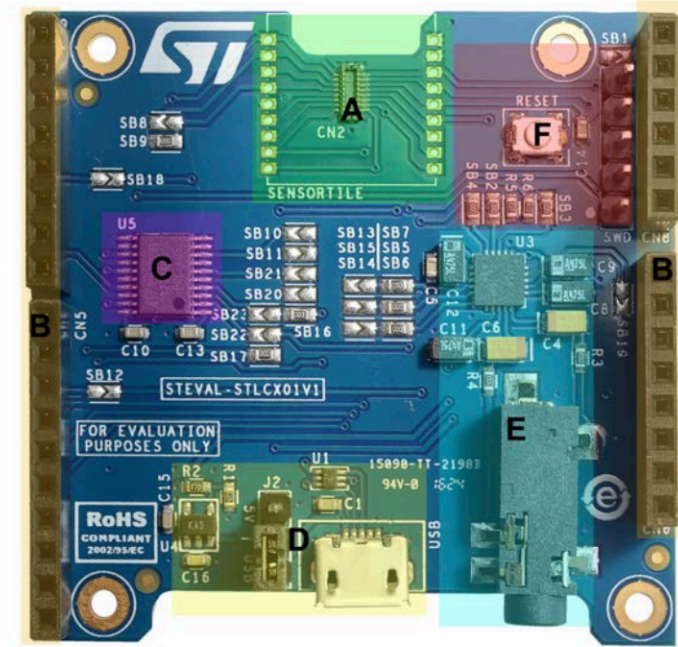
Figure 2.3: SensorTile expansion cardle board (STLCX01V1) [2]

Reference	Device	Description
A	SensorTile connector and footprint	To plug or solder the SensorTile board
B	Arduino UNO R3 UNO R3 connector	For STM32 Nucleo board compatibility
C	ST2378ETTR	8-bit dual supply 1.71 V to 5.5 V level translator
D	micro-USB connector, USBLC6-2P6 (U1), LDK120M-R (U4)	micro USB power supply /communication port and 3.3 V voltage regulation
E	Audio DAC, phono jack	16-Bit, low-power stereo audio DAC and 3.5 mm stereo phono jack
F	SWD connector, Reset button	5-pin SWD connector for programming debugging and board reset button

Table 2.2: SensorTile expansion cardle board (STLCX01V1) main components description [2]

The cradle board, shown in Figure 2.4, is a companion board with battery, humidity and temperature sensor, and micro-SD card socket that supports rapid prototypes and wearable devices development. Main components of the cradle board in Figure 2.4 are described in Table 2.3. The cradle board with soldered SensorTile can be assembled in the dedicated factor case as

shown in Figure 2.5. The plastic case provides reliable protection and convenient mounting capabilities.
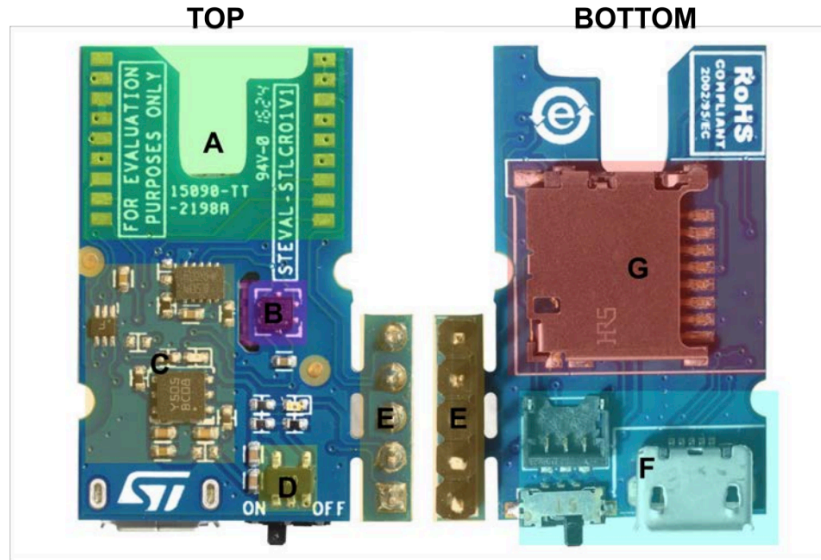
Figure 2.4: STLCR01V1 cradle board [2]

Reference	Device	Description
A	SensorTile footprint	To solder the SensorTile board
B	HTS221	Capacitive digital sensor for relative humidity and temperature
C	STBC08PMR, STC3115, LDK120M-R, USBLC6-2P6	800 mA standalone linear Li-Ion battery charger with thermal regulation, Gas gauge IC, 200 mA low quiescent current very low noise LDO, very low capacitance ESD protection
D	Power on/off switch	
E	SWD connector	5-pin SWD connector for programming and debugging
F	Micro USB connector, 3-pin battery connector	Micro USB battery charging supply /communication port and connector for Li-Ion battery power supply
G	Micro-SD card socket	

Table 2.3: STLCR01V1 cradle board main components description

7
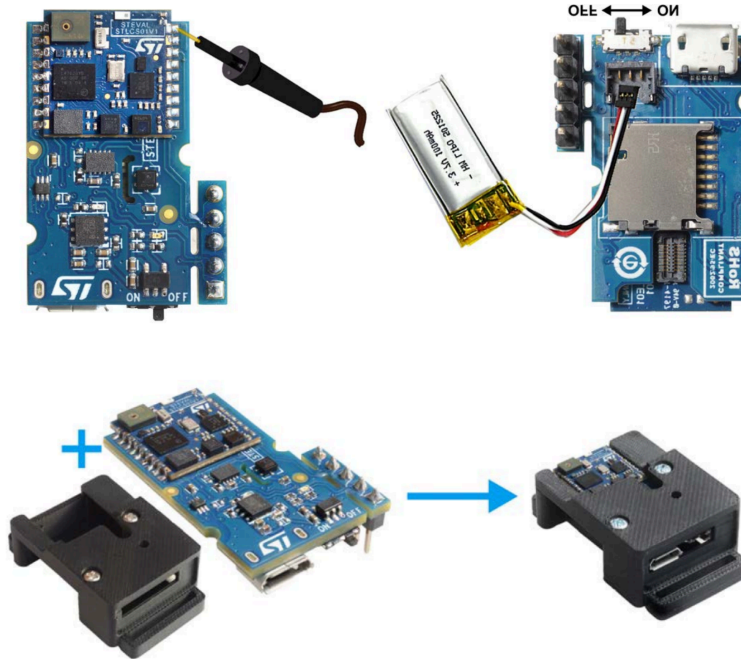
Figure 2.5: Assembled SensorTile with cradle board in plastic protection case [2]

2.1.2 SensorTile Programming Interface

To effectively program the SensorTile, we need to connect an extra STM32 Nucleo board, which

embeds ST-LINK debugger and programmer, with the serial wire debug (SWD) connector on the

cradle board. The complete duo board configuration is shown in Figure 2.6. The ST-LINK

debugger on Nucleo board is only working with specific jumper removed as shown in Figure 2.7.

8
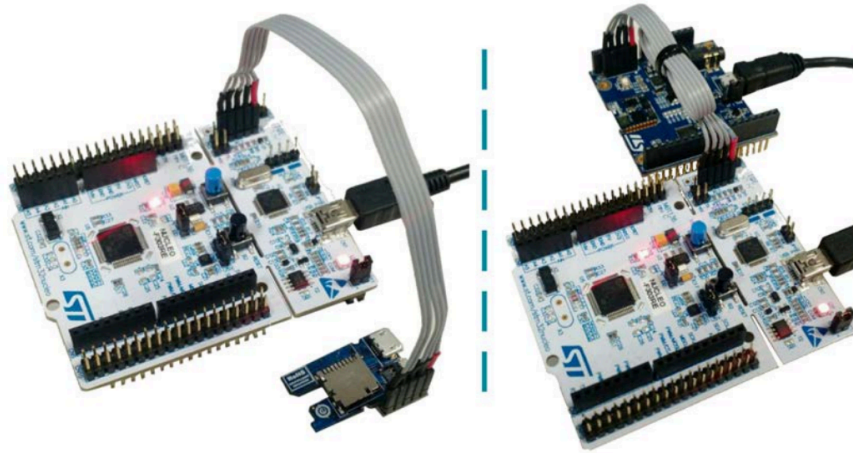
Figure 2.6: SWD connections between Nucleo board and cradle board [2]
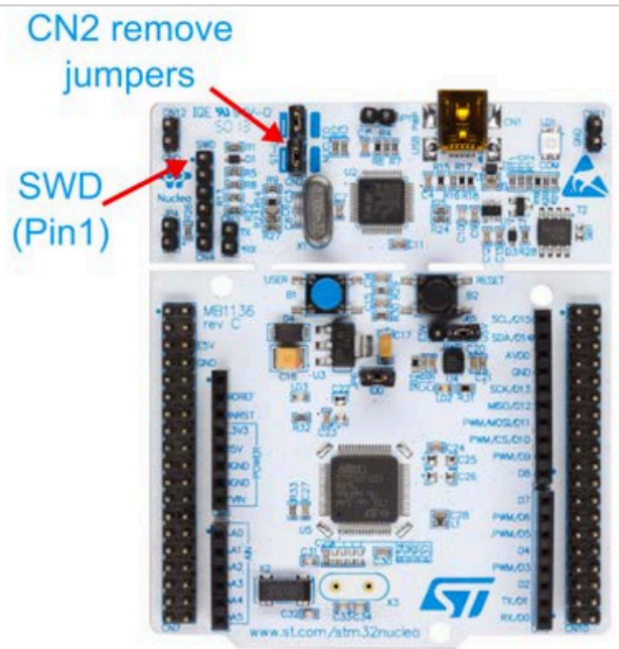
Figure 2.7: Jumper removal for ST-LINK debugging interface on Nucleo board [2]

9

2.1.3 Sensors and Bluetooth Low Energy Model

In this section, major sensors and communication module of SensorTile will be introduced. This thesis mainly uses 1) motion sensors including LSM6DSM (3D digital accelerometer and 3D digital gyroscope) and LSM303AGR (3D accelerometer and 3D magnetic sensor; 2) digital MEMS microphone (MP34DT05-A); and 3) Bluetooth low energy processor (BLUENRG-MS). The LSM6DSM is a system-in-package featuring a 3D digital accelerometer and a 3D digital gyroscope that supports high-performance mode at 0.65 mA and always-on low-power mode. The LSM6DSM as MEMS sensor module leverages the robust and mature manufacturing processes and matches the characteristics of the sensing element. The LSM6DSM has a full-scale acceleration range of ±2/±4/±8/±16 g and an angular rate range of ±125/±245/±500/±1000/±2000 dps. The LSM6DSM fully supports EIS and OIS applications as the module includes a dedicated configurable signal processing path for OIS and auxiliary SPI configurable for both gyroscope and accelerometer.

LSM303AGR is an ultra-low-power high-performance system-in-package featuring a 3D digital linear acceleration sensor and a 3D digital magnetic sensor. The LSM303AGR has linear acceleration full scales of ±2g/±4g/±8g/±16g and a magnetic field dynamic range of ±50 gauss. It includes an I_2C serial bus interface that supports standard, fast mode, fast mode plus, and high-speed (100 kHz, 400 kHz, 1 MHz, and 3.4 MHz) and an SPI serial standard interface. The MP34DT05-A is an ultra-compact, low power, omnidirectional, digital MEMS microphone that can detect acoustic waves with a 64 dB signal-to-noise ratio and –26 dBFS ±3 dB sensitivity.

The BlueNRG-MS is a very low power Bluetooth low energy single-mode network processor, compliant with Bluetooth specification v4.1 supporting multiple roles simultaneously

and can act at the same time as Bluetooth smart sensor and hub device. In addition to BLE stack running on the embedded ARM Cortex-M0 core with on-chip non-volatile Flash memory, the BlueNRG-MS offers options of interfacing with external microcontrollers via SPI transport layer, which is utilized by SensorTile's main microprocessor for BLE data transmission.

2.2 Software Platform

In this section, the integrated development environment and embedded software samples will be introduced.

2.2.1 Integrated Development Environment – System WorkBench

The System WorkBench toolchain [3], called SW4STM32, is an open-source integrated development environment based on Eclipse, which supports the full range of STM32 microcontrollers and boards. The System WorkBench toolchain is able to support multiple operating systems including Windows, MAC OS, and Linux, which is crucial for educational distribution. The System WorkBench provides comprehensive support for STM32 microcontrollers and STM32 firmware with GCC C/C++ compiler, GDB-based debugger, ST-LINK and unlimited code size.

2.2.2 Embedded Software for SensorTile

STMicroelectronics provides embedded software examples for SensorTile including motion sensor data streaming via USB and BLE, data logging on SD card, and audio acquisition. This thesis modified the original embedded software to further introduce inertial sensing, digital signal processing, finite system machine, gesture recognition, and embedded machine learning.

11

2.2.2.1 STSW-STLKT01 [4] – Beginner Application Software Samples

STSW-STLKT01 is the embedded software for SensorTile, which provides simple examples of data logging of multiple sensors, audio acquisition, and BLE communications. The three sample projects, DataLog, AudioLoop, and BLE_sample, are updated to introduce motion sensing, digital signal processing (DSP), and Bluetooth communication. The DataLog project is updated to achieve displacement estimation using inertial sensors and motion sensing based gesture recognition. The AudioLoop project is modified with embedded digital signal processing for the audio signal acquired by the microphone. The BLE_sample project provides a demonstration of motion data streaming via BLE.

2.2.2.2 FP-SNS-ALLMEMS1 [5] – Advanced Application Software Samples

FP-SNS-ALLMEMS1 is the STM32 ODE function pack for IoT node with BLE connectivity, digital microphone, environmental and motion sensors. This thesis modifies it and utilizes this firmware to support high sampling rate motion data acquisition via BLE communication.

2.3 IoT Curriculum

This IoT curriculum has been designed to support large student course enrollment by developing a set of rapid learning Tutorials and system development Reference Designs.

2.3.1 Tutorials

Tutorials are designed for the purpose of rapid learning with practical SensorTile projects. The set of tutorials is written with detailed instructions and plenty of figures to give comprehensive

12

and complete guidance. Tutorials intend to provide experience in novel application development and help students effortlessly accomplish the project and assignment.   There are total nine tutorials introducing the hardware setup, usage of development environment, motion data acquisition, event detection, gesture recognition, audio sampling and signal processing, integrated firmware programming, inertial sensing, and BLE communications. Tutorial details will be further discussed in chapter 3.

2.3.2 Reference Design

The SensorTile Reference Designs provide complete, end-to-end experience in development of a system. This experience prepares students for innovation, implementation of new systems, and independent project missions. This thesis presents a reference design including a SensorTile system that integrates development experience from the previous tutorials. It introduces a motion controlled audio signal processing system, which acquires SensorTile motion sensor data source, detects orientation, and selects one of the two signal processing system according to orientation. The reference design will be further discussed in chapter 4.

Chapter 3

Tutorials

This chapter introduces the motivation and objective of each tutorial. All tutorials together provide a thorough entry-level introduction of electrical and computer engineering fundamentals, state-of-art internet-of-things technology, computing systems, and innovative system design using practical projects as examples.

3.1 Tutorial 1: Introduction to STMicroelectronics Development Environment

This Tutorial provides an introduction to the SensorTile hardware platform and its development tools. There are two different versions of this tutorial, where one is designed for Apple Mac OS platform and the other one is designed for Microsoft Windows platform. This provides the multi-platform support of the curriculum and manages the scalability of the curriculum.

Development environments are essential to development of software for IoT systems and other products. These provide support to developers for both creation of systems, testing, debugging, and installation of software systems on platforms. This development environment is referred as an Integrated Development Environment. This includes all software tools required to create a software distribution for the SensorTile, compile this software system into the processor instruction set using a Build capability, execute this system using a Debug capability, and create a binary "image" file that can be installed in the SensorTile non-volatile storage.

The first section of Tutorial 1 introduces the installation of the Integrated Development Environment, System WorkBench, on a personal computer. The IDE installer is introduced to manage the installation procedures and environment dependencies. For Windows users, STM32 ST-LINK USB driver, Vitural COM Port driver, graphical user interface (GUI) STM32 ST-

14

LINK Utilities debugging tool [6], and PuTTy [7] are introduced as additional development environment resources. For Mac users, Mac command line interface (CLI) Terminal [8] and CLI STM32 ST-LINK Utilities debugging tool [9] are included in the development environment.

The second section of Tutorial 1 presents the usage of the IDE and the acquisition of embedded software example project, STSW-STLKT01 v1.2.0 [2]. This will specifically include the procedures of Import, Build, Run, Debug and Flash the SensorTile board to run the example Data Logging project, a system that demonstrates sensor data streaming via USB and logging on SD card. Import, Build, Run, Debug and Flash procedures are demonstrated through series of screenshot figures and explanations. For example, Figure 3.1 presents the Import procedure and Figure 3.2 shows the Build interface.
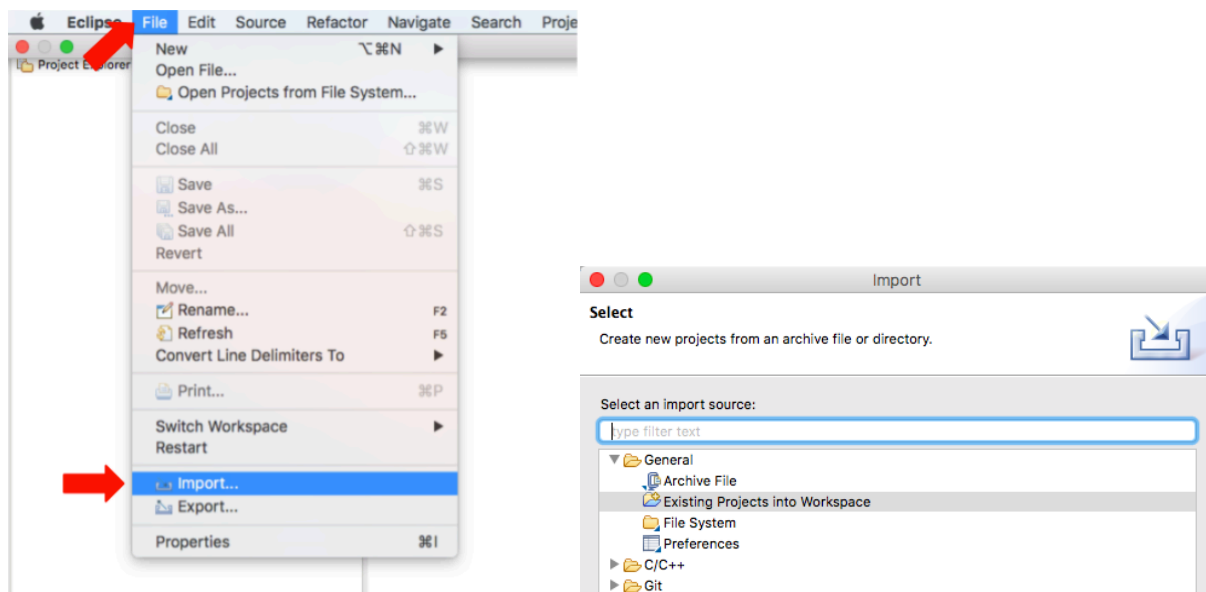
Figure 3.1: Importing an existing project into the workspace
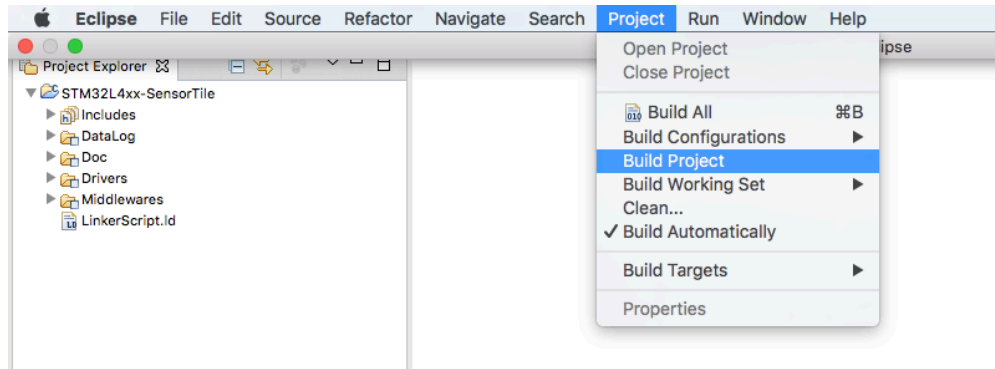
Figure 3.2: Compiling the source code for the example project

In addition to software IDE, Tutorial 1 also demonstrates the SensorTile hardware setup including removal of Nucleo board jumpers in Figure 3.3, mounting of SensorTile on extended cradle board in Figure 3.4, SWD cable connection of Nucleo and SensorTile in Figure 3.5, and boards connection with laptop in Figure 3.6.
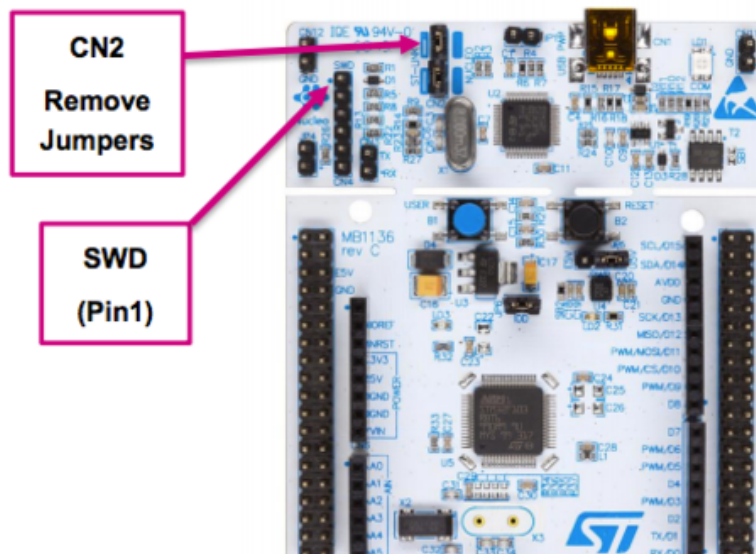
Figure 3.3: Removing the CN2 Jumpers from the Nucleo-L47RG board

16

Figure 3.4: Correct mounting of SensorTile on extended crable board
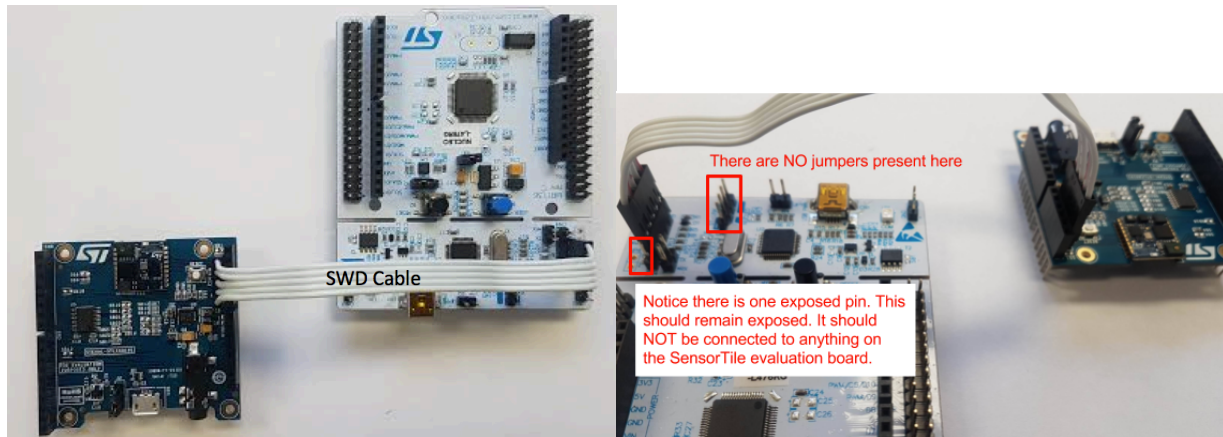
Figure 3.5: SWD connection between Nucleo board and SernsorTile
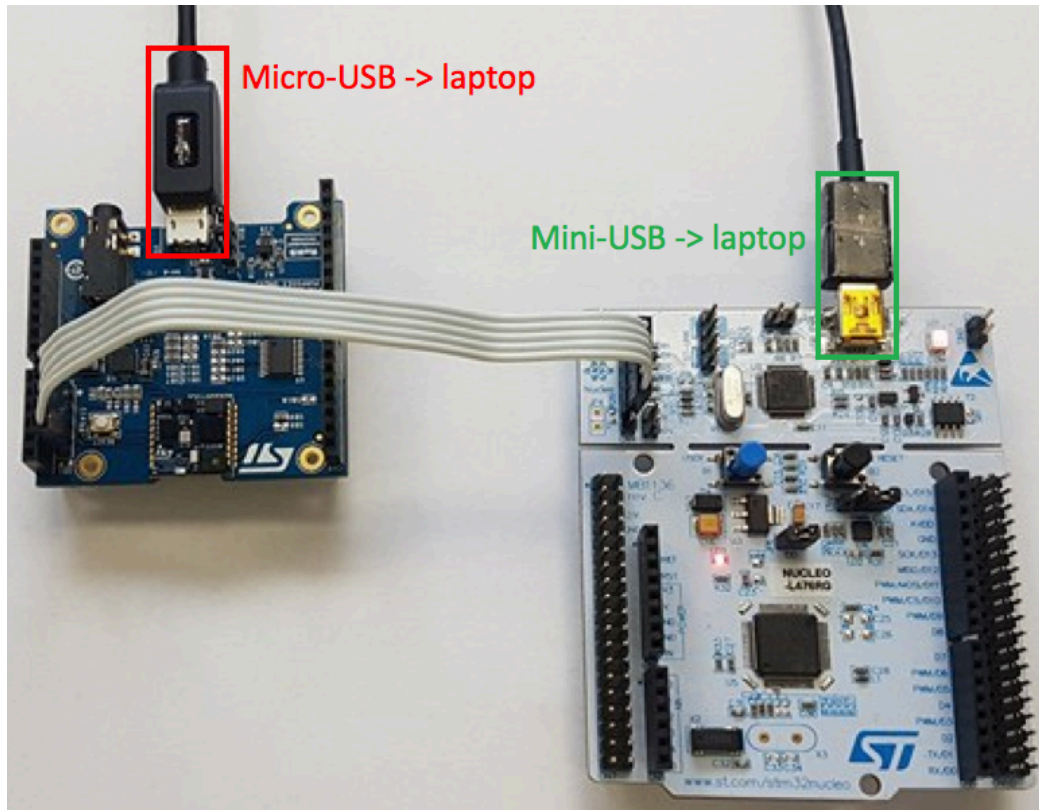
Figure 3.6: Establish a USB-wireline connection between each board and your PC

Through the hardware configurations, practical fundamentals such as USB mini port, USB micro port, and on-board pin orientations are briefly introduced.

The last section of Tutorial 1 introduces the visualization of real-time motion data streaming via serial connection through Terminal Screen for Mac platform and PuTTy for Windows platform as shown in Figure 3.7. This provides an introduction of a complete end-to-end IoT system development covering hardware, software, professional IDE, and final application as motion data acquisition.

Figure 3.7: SensorTile data streaming captured by Terminal

3.2 Tutorial 2: Sensor System Signal Acquisition, Event Detection and Configuration

Tutorial 2 provides experience in the development of applications that acquire IoT system sensor signals, detect events in the sensor system data stream, and also configure sensor systems.

Tutorial 2 first provides an introduction to the control of sensor signal acquisition and sensor system configuration, which are fundamental to IoT system development. The DataLog system will first initialize STM32L4x HAL library to configure the Flash prefetch, instruction, and data caches. It additionally configures the Systick to generate an interrupt each 1 millisecond. Then, the GPIO pins are registered with SPI interfaces to enable 9 Degree-of-Freedom (DoF) motion sensors used in DataLog system. After the registration, each sensor is

19

enabled with default sensor sensitivity, output data rate (ODR) and full scale (FS). Eventually, the DataLog system utilized the Systick as a timer to control the motion signal sampling, data acquisition, and USB data streaming through sensor handler.

Secondly, Tutorial 2 introduces sensor signal detection and notifications with the sensor handler function. Using accelerometer sensor handler as an example, the handler will first verify whether the accelerometer is initialized and then acquire the acceleration data of three axes in the unit of milli-g. The handler function will further convert integer accelerometer data into string format as output. Lastly, the handler will fill the USB TX buffer with the converted string output to achieve serial motion data streaming. The users can receive the streaming data from USB RX using Terminal Screen or PuTTy.

In addition to default DataLog system, Tutorial 2 provides an experience in software system development for the SensorTile IoT system demonstrating important capabilities of the System WorkBench Integrated Development Environment in accelerating system development. Those capabilities include IDE features such as "Open function declaration" and "Open function/variable call hierarchy" to help students rapidly understand and manipulate large-scale software/firmware system and framework.

The last section of Tutorial 2 introduces acceleration magnitude, event detection, and sensor full scale by modifying the default DataLog system. Acceleration magnitude, a_{mag}, can be calculated by formula $a_{mag} = \sqrt{a_x{}^2 + a_y{}^2 + a_z{}^2}$. Tutorial 2 event detection assignment requires the students to send a warning massage through USB once the acceleration magnitude exceeds a threshold value, which indicates a moving instance. Tutorial 2 guides students to calculate the acceleration magnitude and send USB RX with notification message. Tutorial 2 full scale assignment will introduce the method for adjusting sensor measurement range. More

20

specifically, large acceleration magnitude will not be detected if the sensor full scale range is configured small. Least but not last, gyroscope and angular velocity are briefly introduced in the end of Tutorial 2.

3.3 Tutorial 3: Accelerometer Sensor Systems and Orientation and Event Detection with Finite State Machine

Tutorial 3 provides additional experience in sensor system data processing including recognition of gesture motion. Experience from this Tutorial provides guidance for the development of capable IoT systems that recognize and may even guide specific motion.

Tutorial 3 provides first step as an introduction to accelerometer sensor systems and methods for detection of sensor orientation by exploiting gravitational acceleration signals. It also introduces the Cartesian coordinate to Polar coordinate conversion in 3D motion sensing and measurement of orientation by the polar coordinate system.

The second section of Tutorial 3 mainly provides an introduction to gesture recognition through the use of sensor information and state machine systems for characterizing specific behavior. In the tutorial, the specific gesture is a "flip" defined as a) keep the SensorTile "upside down" for at least τ seconds; and b) keep the SensorTile "right-side up" for at least τ seconds. The z-axis acceleration will be compared with a threshold value around 1g to evaluate the sensor orientation and the Systick will be used to trace the time. If the gesture is successfully detected, a notification message will be sent over USB. We will model this gesture recognition system as a simple finite state machine, which is shown in Figure 3.8.
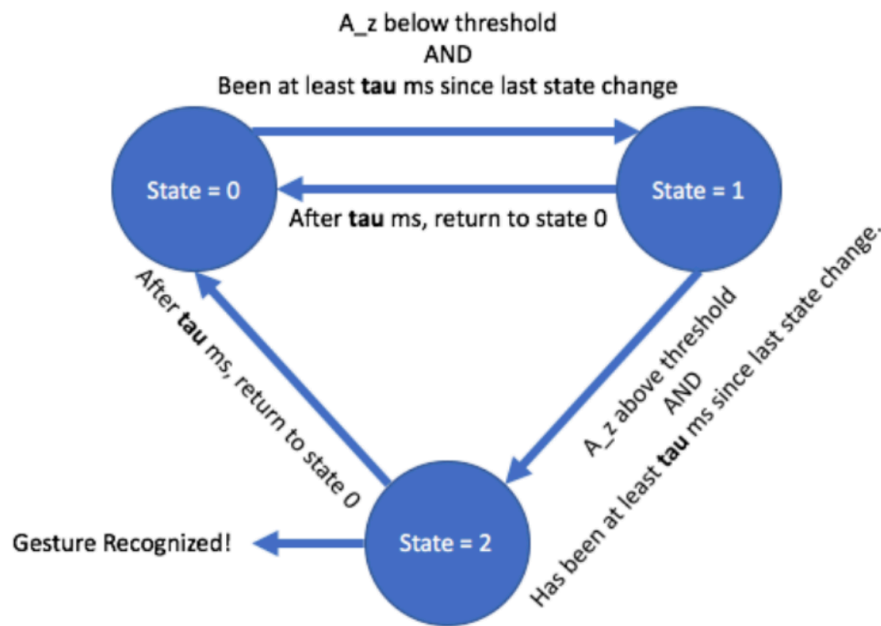
Figure 3.8: State machine for gesture recognition

For this gesture recognition task, Tutorial 3 introduces fundamentals of embedded C programming including header file, pointer, parameter passing by reference, and using conditions to implement finite state machine.

3.4 Tutorial 4: Introduction to Audio Sampling and Signal Processing

This Tutorial provides experience in sensing and generation of audio signals using the state-of-the-art microphone on the SensorTile system. Tutorial 4 also introduces Digital Signal Processing with methods to both hear and visualize the capabilities of DSP systems.

The SensorTile audio interface can be visualized in Figure 3.9.
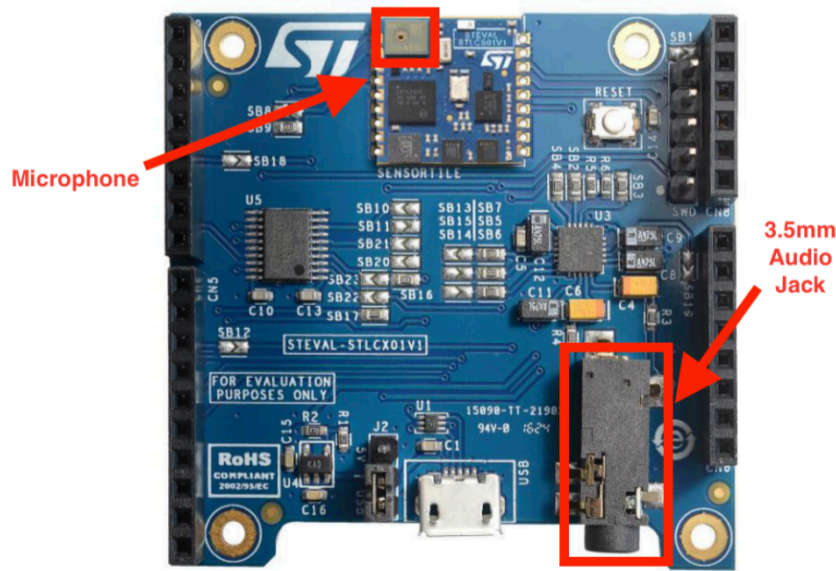
Figure 3.9: Audio interface of SensorTile

The first section of Tutorial 4 introduces audio sensor signal acquisition systems, AudioLoop. AudioLoop is an application included in STSW-STLKT01V1 software package, which sends audio signals acquired by the microphone to an on-board Digital to Analog Converter (DAC) via an I_2C interface. A digital MEMS microphone samples the analog sound signal and generates a Pulse-Density Modulation stream, which is converted into a Pulse-Coded Modulation (PCM) output stream by the hardware. The output stream is then passed to the on-board DAC, allowing the students to play these sounds on speakers or headphones, or record them on a host PC.

In AudioLoop, the acquired sound signals are sampled from the MEMS microphone input sensor and converted to a digital data stream by the Analog to Digital Converter (ADC). This data is then applied to the output DAC to provide analog output signals that will be applied to the audio output. After the data is sampled and prior to its being applied to the output DAC, we can introduce signal processing algorithms that operate on this data. This provides an

23

outstanding introduction to signal processing and associated IoT system development. The signal processing capability is demonstrated by adding discrete-time Infinite-Impulse-Response (IIR) filters to the original AudioLoop application in this section.

The second section of Tutorial 4 will guide students to implement two types of digital filters: low pass filters and high pass filters. Low pass filters attenuate signals with frequencies higher than a defined corner frequency, while high pass filters attenuate signals with frequencies lower than a defined corner frequency. Here, we will modify the AudioLoop application to add first-order low and high pass filters and examine the effects of the filters by hearing and visualizing the output signals.

In tutorial 4, we will first set cut-off frequency as 1.5 KHz and then generate the sound of consonant | s | as "ss" in "miss", which corresponds to a waveform with frequency above 3KHz [6] and the sound of vowel | u | as "oo" in "boot", which corresponds to a waveform with frequency below 1KHz [10]. Then, we apply different filters when we generate those sound. By listening to the different combination of sound and filter applied, student will intuitively understand the usage of different filters where the low pass filter will attenuate the high frequency content dominating | s | sound and the high pass filter will attenuate the low frequency content dominating | u | sound.

In addition to capability of hearing consequence of applying the filters, Tutorial 4 also supports visual examination of signal waveforms and signal spectrograms through open source tool, Audacity [11] shown in Figure 3.10.

24
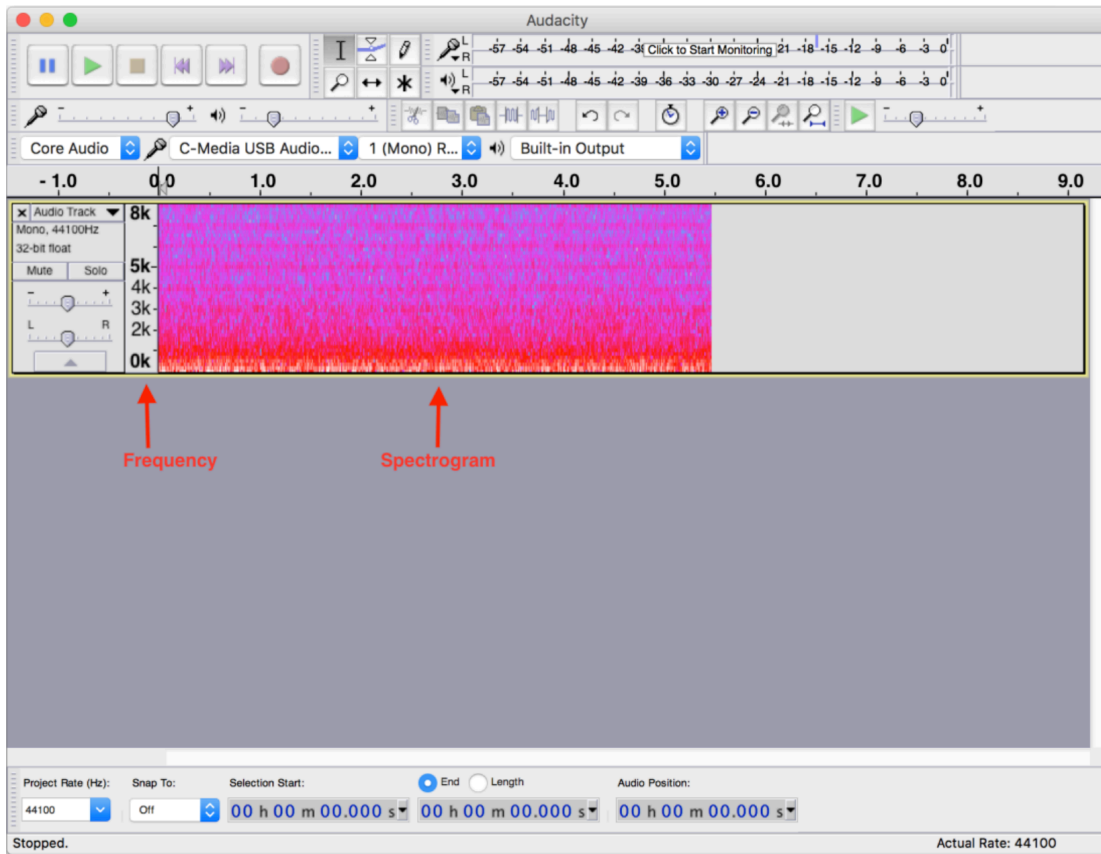
Figure 3.10: Audacity showing a spectrogram

The installation and usage of Audacity will be first introduced. Then, we will set filter cut-off frequency to 3 KHz and play 4 KHz and 1 KHz sine wave test tone accordingly using the cellphone. Eventually, we will generate the spectrogram of each waveform with no filter, low-pass filter, and high-pass filter in Audacity and visually examine the effect of the filters, where 4 KHz band in spectrogram will become narrower when low-pass filter is applied.

Tutorial 4 provides an Audacity spectrogram example of three sound of "sss" followed by three sounds of "ooo" with low-pass filter and high-pass filter applied accordingly. Figure 3.11 provides a spectrogram with low-pass filter applied. Note that the "sss" sound is attenuated in volume while the "ooo" sounds of predominantly low frequency components are preserved.

25
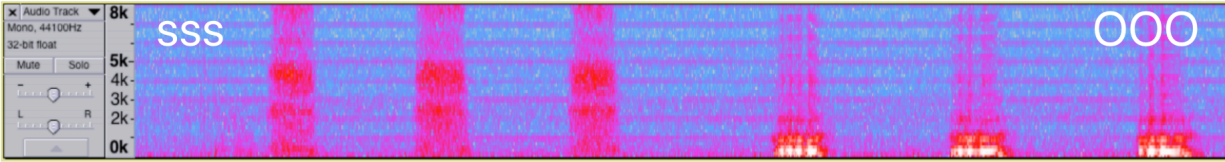
Figure 3.11: Spectrogram of low-pass filtered signal

Figure 3.12 provides a spectrogram with high-pass filter applied. Note that the "sss" sound is increased in relative volume while the "ooo" sounds of predominantly low frequency components are now distorted.
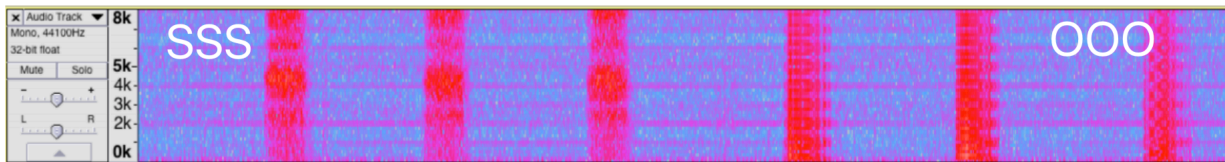
Figure 3.12: Spectrogram of high-pass filtered signal

3.5 Tutorial 5: SensorTile Firmware Programming

This Tutorial provides valuable experience in the creation of firmware applications using the state-of-the-art SensorTile system and tools. The Tutorial steps provide an introduction to integration of source code and header files into new SensorTile projects; experience in configuring the firmware build system; and complete development experience valuable for next Tutorials that integrate multiple system resources and associated capabilities.

Tutorial 5 serves as an advanced guidance of integrating and configuring the project in IDE System WorkBench and provides a template for complicated SensorTile system applications in System WorkBench.

26

3.6 Tutorial 6: Introduction to Bluetooth Low Energy Wireless Interfaces

This Tutorial provides an introduction to the important principles of Bluetooth Wireless Interfaces. This includes also valuable experience in data transport between the SensorTile system and an embedded Linux platform - the BeagleBone. Tutorial 6 will prepare students for many next steps in wireless IoT system development that will be appearing in the following Tutorials and Reference Designs. This will also include development of new firmware providing access to multiple sensors on the SensorTile platform.

SensorTile has a Bluetooth module, BLUENRG-MS, to support the BLE communication. In this tutorial, we will introduce the BLE_SampleApp project in the starter firmware package, STSW-STLKT01V1, to explore the BLE communication on SensorTile.

The first section of Tutorial 6 briefly introduces installation of Bluetooth Low Energy starter firmware. A minor firmware code change is required to enable the BLE debug interface via USB streaming on SensorTile. The BLE debug interface will provide necessary information of the SensorTile hardware such as MAC address.

The second section of Tutorial 6 provides an introduction to BLE connection to SensorTile via BlueZ [13] on Linux system. SensorTile is able to communicate with mobile devices like smartphones and embedded Linux devices such as BeagleBone and Raspberry Pi through Bluetooth. BlueZ is the tool we will use on embedded Linux to handle the BLE communication. In this tutorial, we will use BeagleBone Wireless Green as gateway to introduce BlueZ.

BlueZ is the official Bluetooth stack for Linux kernel-based family of operating systems. Its goal is to program an implementation of the Bluetooth wireless standards specifications for

27

Linux. BlueZ also provides support for the core Bluetooth layers and protocols. It is flexible, efficient and uses a modular implementation.

Tutorial 6 introduces installation of BlueZ, bluetoothctl utility in BlueZ, and CLI operations using bluetoothctl. Bluetoothctl has variety of commands to list device Bluetooth controller MAC address; configure default Bluetooth controller; control power of Bluetooth controller; configure Bluetooth agent; scan and list available Bluetooth devices; pair, connect, and disconnect with specific Bluetooth device; and acquire more information of the broadcasting Bluetooth device.

Following Tutorial 6, users should be able to establish Bluetooth connection between the BeagleBone and the SensorTile. LED on SensorTile and USB debug interface can be utilized to check Bluetooth connection status of SensorTile.

3.7 Tutorial 7: Introduction to Bluetooth Low Energy Communication and the GATT [14] Profile

This Tutorial introduced Bluetooth Wireless Communication principles and methods. This includes hands-on experience with communication of SensorTile sensor data to the BeagleBone platform. Tutorial 7 provides experience in the communication protocol central to global IoT applications: The most important Generic Attributes (GATT) profile and associated communication protocols. The BlueZ Bluetooth system interface, hosted on Linux platforms, is also included. It also includes a demonstration of bi-directional communication with control of events on the SensorTile by an application on the BeagleBone.

More specifically, Tutorial 7 will demonstrate how to use BlueZ gatttool utility to read environmental data from SensorTile Bluetooth Generic Attribute Profile and how to control the SensorTile LED through GATT.

The Generic Attributes profiles define a hierarchical data structure that is exposed to connected Bluetooth Low Energy devices. GATT is built on top of the Attribute Protocol (ATT), which uses GATT data to define the way that two Bluetooth Low Energy devices send and receive standard messages. In this tutorial, we will use SensorTile as the GATT server to provide services and use BeagleBone as the GATT client to send requests to the GATT server to get environmental data collected on SensorTile.

Tutorial 7 will first introduce the gatttool, which is a BlueZ utility. Similar to bluetootctl utility introduced in Tutorial 6, Tutorial 7 explains variety of critical gatttool commands including connection to Bluetooth devices; discovery primary services of GATT server; discovery all defined characteristics and characteristic descriptors (UUID with handles) of GATT server; read characteristic value using handles; and write/modify characteristic value using handles. Through the gatttool, the user can review and take advantage of service handles to request data and even control the SensorTile via Bluetooth. For example, command "char-read-hnd" can be used to request SensorTile environmental data; command "char-write-req" can be used to control SensorTile environmental data streaming; and command "char-write-cmd" can be used to control SensorTile LED. The GATT profile, service uuid, handles, and configuration commands are defined in the SensorTile BLE firmware. The gatttool plays a role in exploring services and communicating with the GATT server, SensorTile. The details of examples can be accessed in Tutorial 7 in Appendix.

29

The remaining of Tutorial 7 introduces data parsing, data requesting and data recording using non-interactive gatttool mode. This provides the capability to store requested Bluetooth data for future development and processing.

3.8 Tutorial 8: Introduction to Motion Data Acquisition via Bluetooth Low Energy Communication

This Tutorial introduces new capabilities that are provided by new firmware enabling direct access to multi-axis motion sensor data. This includes the installation of the FP-SNS-ALLMEMS1 [15] firmware application provided by STMicroelectronics. Through use of gatttool, Tutorial 8 provides the BeagleBone system with a high performance, wireless SensorTile motion sensor. This addresses an important need for compact and wearable motion sensing systems for IoT systems.

Tutorial 8 firstly provides an introduction to download, modify, and flash advanced firmware, FP-SNS-ALLMEMS1. The complete FP-SNS-ALLMEMS1 is an STM32 ODE function pack which lets you connect your IoT node to a smartphone via BLE and use a suitable Android™ or iOS™ application, like the BlueMS app, to view real-time environmental sensor data, motion sensor data, digital microphone levels and battery level [15]. In this tutorial, we will concentrate on motion sensor data acquisition via BLE communication. The modification of the firmware mainly enables the USB debug interface and provides a powerful tool to understand the complicated firmware.

Secondly, Tutorial 8 guides users to further explore the GATT profiles of the FP-SNS-ALLMEMS1 firmware using bluetoothctl utility and gatttool utility. Tutorial 8 introduces commands and handles that can communicate with SensorTile to request motion data. By cross-

30

comparing motion data presented in USB debug interface and hexadecimal data received in gatttool, motion data including acceleration data, gyroscope data, and magnetometer data parsing is thoroughly explained with manually calculated examples. The unit of each motion sensing data is also addressed in the Tutorial.

Eventually, Tutorial 8 introduces the command to save requested motion data to text file for further processing and other applications.

In summary, this tutorial provides a complete end-to-end motion sensing IoT system framework composed of a BeagleBone gateway and Bluetooth Wireless motion sensors, that can be rapidly adopted for embedded computing projects and machine learning based motion classification projects. Chapter 4 will provide more examples of student's reference design sand customized projects using this IoT system framework.

3.9 Tutorial 9: Introduction to Inertial Sensing

This Tutorial provides an introduction to Inertial Sensing through a demonstration of displacement sensing. This includes a) an extension of the Datalog application with increased sampling rate and addition of anti-aliasing and drift-reduction filters; and b) the Python SensorTile Animation System that permits visualization of SensorTile motion sensor data in real time. As a result, Tutorial 9 provides users with the background required to develop new motion classification systems based on displacement of the SensorTile.

The first section of Tutorial 9 provides an introduction of installation of Displacement Estimation project and Python animation system. Displacement Estimation project enables all the acceleration sensing, data processing, and data streaming via serial communication on

31

SensorTile. The Python SensorTile Animation System mainly serves as a graphical interface to read the data from serial port and display the data streaming. Users need to download Displacement Estimation project, import the project into System WorkBench, and flash the SensorTile to load the default Displacement Estimation firmware. The Python animation system requires Python installation and Python package installation. The installation procedures are different for Windows users and Mac users. Therefore, two versions of Python animation system are included in Tutorial 9. After successful installation of both Displacement Estimation project and Python animation system, users will have the capability to visualize the x-axis displacement and acceleration of SensorTile in real-time shown in Figure 3.13.
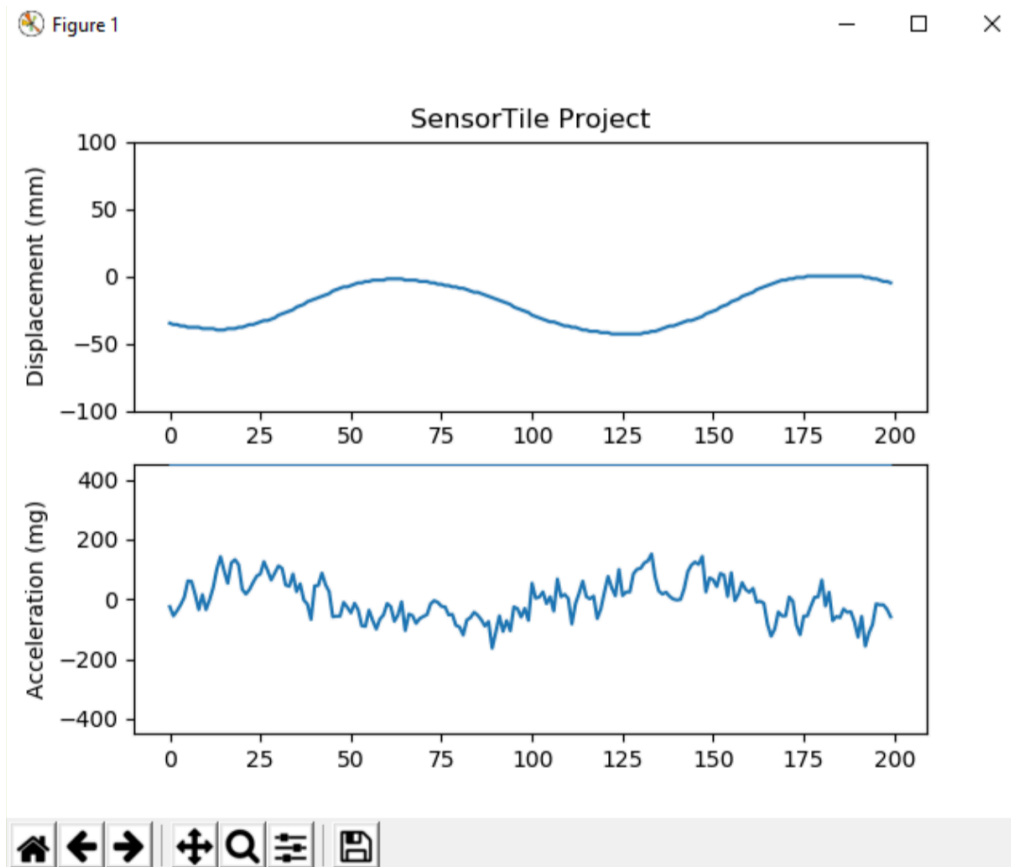
Figure 3.13: Python Animation system with Displacement Sensing

32

The second section of Tutorial 9 interprets the displacement estimation system in detail. It firstly introduces displacement estimated by discrete-time Trapezoidal integral of acceleration. Fundamental physics describes the familiar relationship between acceleration, velocity, and displacement by continuous integral below.

$$v(t) = \int a(t)\,dt + v_0$$

$$d(t) = \int v(t)\,dt + d_0$$

However, the SensorTile acceleration data source is a discrete time, sampled signal, where acceleration is sampled at 100 Hz, a discrete trapezoidal integration is computed on velocity and displacement using the relationships below. Δt in this case equals the sampling period of $1/(100\text{ Hz}) = 10$milli-seconds.

$$v(1) = v(0) + \left[a(0) + \frac{a(1) - a(0)}{2}\right] * \Delta t$$

$$d(1) = d(0) + \left[v(0) + \frac{v(1) - v(0)}{2}\right] * \Delta t$$

where a(1) is the current acceleration sample, a(0) is the previous acceleration sample, v(1) is current velocity computation, v(0) is the previous velocity computation, d(1) is the current displacement computation, d(0) is the previous displacement computation, and Δt is 0.01s. In addition, a(x) is the acceleration signal based on acceleration sampled by the SensorTile system.

33

The second Section of Tutorial 9 then introduces acceleration sensor signal processing and computed displacement data processing while addressing common problems including aliasing, acceleration offset and estimated displacement drift.

According to Nyquist sampling theorem, a band-limited continuous-time signal can be sampled and perfectly reconstructed from its samples only if the sampling rate is over twice its highest frequency components. In other words, accelerometer with sample rate at 100 Hz can only capture up to 50 Hz acceleration activities. Any frequency components above 50 Hz in our case become indistinguishable from a low-frequency component, called aliasing. Therefore, the raw acceleration data cannot be directly used for integral until we remove the aliasing effect. In Tutorial 9, an anti-aliasing filter, IIR low-pass filter with 30 Hz cut-off frequency, is applied to the raw acceleration data as preprocessing.

The acceleration offset is mainly produced by imperfect mounting of the sensor. All accelerations are characterized by gravity force (mg). Therefore, when the sensor's x-y plane is not perfectly parallel to the horizontal the surface due the tilted sensor or non-flat surface, an acceleration offset will be introduced into the system. If we do not remove the offset from the raw acceleration, we will get a non-zero acceleration when the SensorTile is static. Eventually, this non-zero acceleration will contribute into large drift in velocity and displacement integral. Another drift contributor is the static acceleration noise where there is a really low frequency, less than 0.1 Hz, noise from the sensor.

In order to completely remove the drift effect, an adaptive filter or zero-velocity [12] state based calibration is usually preferred. However, this will introduce complexities for students to understand the system. Therefore, this Tutorial proposes a simplified signal

34

processing solution that we will first apply anti-aliasing filter to the raw acceleration data and then apply a high-pass filter with super low cut-off frequency at 0.1 Hz to the pre-processed acceleration data, velocity after integral, and displacement after integral. The high-pass filter with low cut-off frequency in this case will remove the DC offset, which is the acceleration bias, low frequency noise from sensor, and drift, which is low frequency component. However, the high-pass filter introduces another issue that only the displacement extreme result, which is the peak or valley displacement, or displacement in movement can describe the displacement accurately because the high-pass filter will attenuate the displacement signal when the SensorTile is static. This eventually forces displacement to return to zero after specific time period. This is a trade-off solution for the drift issue.

In summary, the Displacement Estimation system workflow is shown in Figure 3.14.
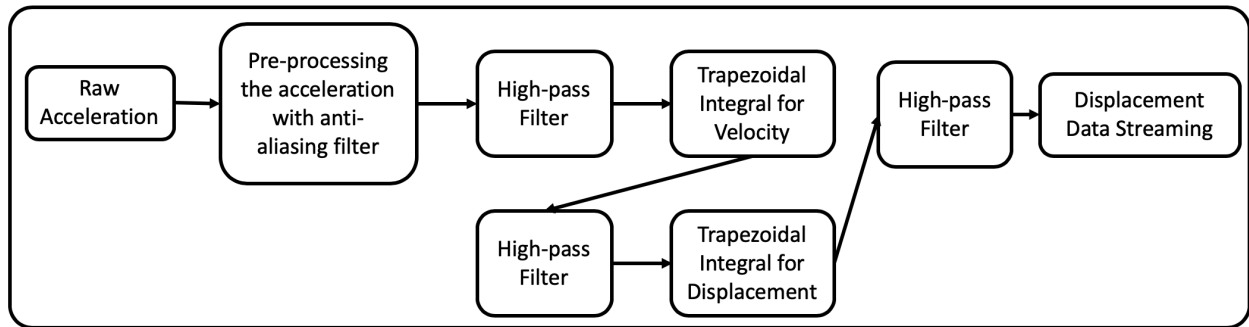
Figure 3.14: Displacement Estimation system workflow

Chapter 4

Reference Design

The SensorTile Reference Design provides complete, end-to-end experiences in development of a system. This experience prepares developers for innovation and implementation of new systems. In this chapter, three different types of reference design will be introduced. Those include SensorTile reference design, IoT machine learning reference design with SensorTile and BeagleBone, and student project reference designs.

4.1 STMicroelectronics SensorTile Reference Design: Motion-Controlled Audio Signal Processing System

This Reference Design includes a SensorTile system that integrates development experience from the previous Tutorials. These Tutorials should all be completed prior to starting on this Reference Design. This system combines capability for motion detection with audio signal processing. It produces a Sensor Tile system that detects the users motion and orientation of the SensorTile to control audio signal processing.

The first section of this reference design provides guidance of properly importing motion sensors driver to the AudioLoop firmware introduced before and implementing low-pass filter and high-pass filter.

The second section of this reference design introduces orientation detection with airplane principle axes including roll, pitch, and yaw. In this system, we can use the roll angle as the representation of the current orientation of a SensorTile board, which can be calculated using the following equation,

$$\text{Roll} = \text{atan}\left(\frac{aY}{\sqrt{aX^2 + aZ^2}}\right) * \frac{180}{\pi}$$

where aX, aY, and aZ are the acceleration in x axis, y axis, and z axis, respectively.

Eventually, this reference design merges the orientation sensing and digital signal processing to build an audio filtering system whose filter type can be controlled by orientation.

4.2 IoT Machine Learning Reference Design with STMicroelectronics SensorTile and BeagleBone

This reference design introduces a new IoT Machine Learning system integrating the SensorTile wireless sensor system with the BeagleBone platform and the Fast Artificial Neural Network [16] (FANN) library, which is a free open source neural network library that implements multilayer artificial neural networks in C with support for both fully connected and sparsely connected networks. We will build a system to implement and train a neural network to determine the orientation of SensorTile. This will provide the experience needed to launch a broad range of IoT Machine Learning systems.

In this reference design, we will first introduce the hardware and software configuration including the installation of a very powerful machine learning neural network open source c library, FANN. Then, we will build a neural network system including 1) data collection: collect acceleration data via BLE communication from SensorTile and parse the data; 2) train the neural net to determine the orientation; and 3) test the neural network performance.

37

For data collection, users are expected to collect SensorTile motion data with three different orientations and each orientation for 5 seconds via BLE communication. Then, the stored data will be interpreted, labelled, and compiled into FANN training file format. The labels in this case are one hot encoded for the purpose of classification. In this case, the raw acceleration data provide significant features to differentiate orientations. However, further signal processing and feature extraction will be necessary to support complicated classification.

In summary, this reference design presents a template for general IoT motion classification system that BeagleBone will serve as an embedded Linux platform gateway to receive and process data, train and test the neural network and SensorTile will serve as a compact and high-performance Wireless BLE motion sensor.

4.3 Student Project Reference Design

Many UCLA students have collaborated in the development of IoT SensorTile systems in course projects. The senior capstone design course has included teams who have developed novel systems for motion classification with SensorTile data sources and machine learning methods. This has included systems with single and dual SensorTile devices along with signal processing, signal feature extraction, neural network design, neural network training, and finally in-field performance analysis. Reference Designs have been developed by these student teams that include design and development documentation as well as source code for your use. These are available for evaluation and guidance with the objective that these may inspire other future development.

38

Currently, there are six student-designed machine learning based motion classification systems including: 1) basketball free-throw classification system that detects characteristics of the basketball free-throw motion; 2) basketball hook-shot classification system that detects the characteristic motion of arm and hand associated with optimal and suboptimal Basketball Hook-shot motion; 3) tennis motion classification system that detects tennis motion swing types and swing quality; 4) resistance training motion classification system that detects proper and improper resistance training motions; 5) climbing motion classification system that classifies the complex motions occurring in climbing; and 6) shoulder motion classification system that classifies proper and improper motion required in shoulder rehabilitation. Majority of those student classification systems apply dual SensorTiles on different parts of body to acquire motion data. Each system described above used the templates provided in the IoT machine learning reference design to implement the end-to-end IoT classification system that relies on complete set of signal acquisition from SensorTile, signal processing, feature extraction and machine learning.

Student project reference design is an important part of the educational community that the curriculum tries to create. It will be further discussed in Chapter 5.

Chapter 5

Educational Community and Student Achievement

This platform is initiated through UCLA Engineering 96C course, a freshman level course, which has become computer science course requirement in UCLA, and UCLA ECE 180D course, which is the senior level capstone design course. Moreover, this educational platform is dedicated to support all students over the world and establish an open and active community that can ultimately share the educational resources and maintain the instructional contents updated.

The educational community has been well established through STMicroelectronics by creating an instructor community and a student community. As a major contributor and founder of SensorTile curriculum, we have supported multiple universities including Columbia University, California State University – North Bridge and Georgia Tech to adopt our IoT and embedded computing curriculum. Now, there are many extraordinary curriculums created by faculties in other universities. For example, Prof. Zhu from the University of Maine contributed a curriculum to introduce embedded system with drone. The original IoT network and embedded computation curriculum does not only serve as a single course but also truly create an active and creative educational platform that can continuously benefit the students and education mission.

From another perspective, I want to feature the student's experience and achievement through this unique educational platform. Multiple student project reference designs have already been introduced in Chapter 4. There are many other student project examples from Engineering 96C. One of the most impressive student projects in Engineering 96C is the number recognition system that SensorTile will automatically distinguish the digital number while it moves as digital number shown in Figure 5.1. More specifically, a digital one movement reflecting on SensorTile will be moving SensorTile downward twice. The team composed two freshman students with

limited backgrounds in electrical and computer engineering designed a finite state machine

system that detects four-direction xy-plane motions and classifies all the digital movement. An

additional reset state is included as start/end indication movement of z-axis motion. The digital

motion detection system worked robustly when the student team demonstrated their final system

in the final presentation. To further address their incredible progress of IoT development, this

system is accomplished within three weeks by only using their one-hour Friday recitation and

their volunteered extra hours on weekend.
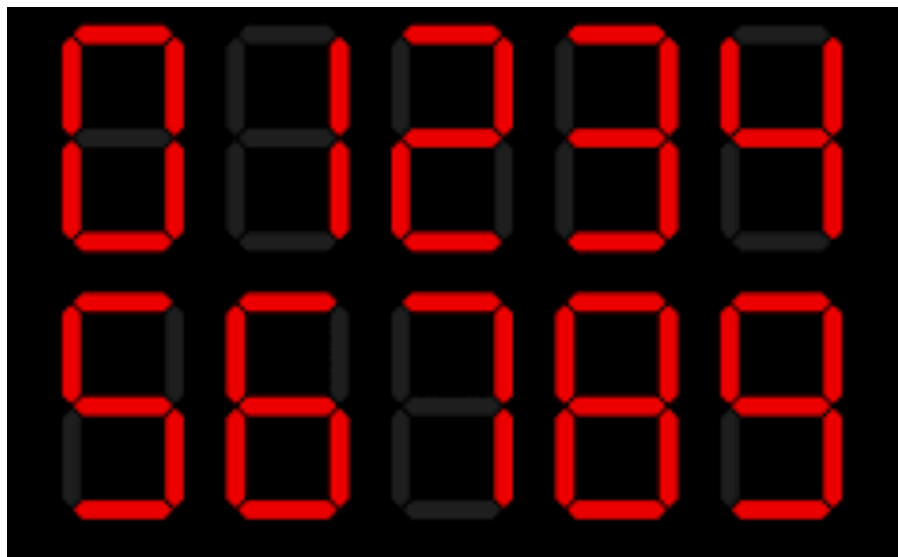
Figure 5.1: Digital number display

Another extraordinary outcome example is Charles Zaloom, who is a graduate of

Engineering 96C course. He continued his interests in embedded machine learning and

eventually developed a complete Embedded Machine Learning system on SensorTile. This

machine learning system is composed of a complete set of C libraries including automated

topography generation, back propagation, neural network training, and activation functions. The

machine learning library is well optimized that it can support a three-layer fully connected neural network training on the SensorTile's 80MHz STM32L476JGY microcontroller. His impressive work will be updated into the current curriculum to introduce embedded machine learning on SensorTile in the future.

Chapter 6

Conclusion and Future Work

This thesis presented an educational platform for IoT network and embedded computation that enables rapid end-to-end system development capabilities through effortless tutorials and reference designs, and provides valuable guidance for diverse student background. In addition, this thesis incorporates variety of topics including data acquisition, motion sensing, data sampling, signal processing, BLE communication, machine learning, and embedded computation to motivate students and encourage innovations. Last but not least, this thesis advocates an open and active educational community that can continuously create expanding shared educational resources in the field IoT and embedded learning.

The platform has room for improvement from several perspectives. The first component I want to improve is the BLE communication library. The current version of BLE communication relies on BlueZ gatttool utility on Ubuntu. However, the gatttool utility introduces high abstract of the low-level BLE protocols and only provides command line interface accessibilities. This eventually introduces complexity in automatically parsing the data in BLE communication and system delay. Therefore, I propose to utilize the standard BlueZ C library to create my own BLE communication embedded C library to expose the details of BLE protocols and manage the BLE communication through GATT profiles in the future.

The current SensorTile curriculum can be further expanded to include embedded Machine Learning and Neural Network contents developed by Charles Zaloom. In this way, the SensorTile curriculum will provide a complete introduction to the operation and training of neural network. In addition, it enables students to learn the most fundamental principles at the very foundation of Neural Network systems. The proposed embedded ML material will

43

introduce Neural Network reference system, forward propagation, backward propagation, error computation, weights, chain rules and gradient descent. This will fully enable a complete learning experience of Neural Network and embedded Machine Learning.

Lastly, the SensorTile curriculum can further introduce energy-aware IoT development by using the Gas gauge Integrated Circuit (IC) on SensorTile cradle board. One of the critical concern of IoT developments is the power usage, which determines the activities and capabilities of different IoT applications. All sensors embedded on SensorTile have a low-power mode that automatically manages the power consumption. However, the system level power management solution to efficiently manage the power consumption through microcontroller is not available on SensorTile. Therefore, it will be valuable to design power management capabilities through Gas gauge IC and algorithm optimization.

APPENDIX A

Tutorial 1

Introduction to STMicroelectronics Development Environment

Mac OS Version

STMicroelectronics SensorTile Tutorial:
Introduction to STMicroelectronics Development Environment and DataLog Project Example
for
Apple Mac Platforms

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 1 of 32

Table of Contents

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 2 of 32

1. Introduction

The SensorTile is a new Internet of Things (IoT) system provided by STMicroelectronics integrating state-of-the-art processor, wireless interfaces, and sensor systems. The SensorTile can form the foundation for wearable consumer devices, wearable medical devices, residential IoT systems and vehicle IoT systems.

The SensorTile system provides an exceptionally powerful and well-supported platform for introduction to IoT technology. The SensorTile is remarkably compact as shown in Figure 1.

Figure 1. The STMicroelectonics SensorTile Platform with a pencil reference indicating its compact geometry.

The SensorTile includes these components:

1) The SensorTile Processor System is an STM32L4 microprocessor based on the ARM Cortex M4 system. This provides introduction to the ARM processor architecture that is deployed on nearly every smartphone on earth.
2) The SensorTile Sensors includes:
   a) The LSM6DSM combining microaccelerometer and microgyroscope.
   b) The LSM303AGR combining microaccelerometer and magnetometer for compass heading
   c) The LPS22HB barometric pressure sensor for determination of altitude and atmospheric pressure.
   d) The MP34DT04 microphone
3) The SensorTile also includes a Bluetooth Low Energy (Bluetooth Smart) wireless interface the BlueNRG-MS system.
4) The SensorTile also includes non-volatile flash storage that stores the executable code that enables IoT system operation.
5) The SensorTile also includes a cradle accessory with additional features including:
   a) SD Card Flash Storage System
   b) STC3115 Battery Monitor providing detailed energy monitoring for the SensorTile
   c) HTS221 Humidity and Temperature environmental sensors

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 3 of 32

48

www.manaraa.com

2. Introduction to This Tutorial

This Tutorial introduces the development environment for the SensorTile system.

Development environments are essential to development of software for IoT systems and other products. These provide support to developers for both creation of systems, testing, debugging, and installation of software systems on platforms.

This development environment is referred to as an Integrated Development Environment (IDE). This includes all of the software tools required to create a software distribution for the SensorTile, compile this software system into the processor instruction set using a Build capability, execute this system using a Debug capability, and also create an "image" file that can be installed in the SensorTile non-volatile storage.

This tutorial is intended for users that have Apple Mac platforms, and will guide users through the tasks listed below. *Please note that there is a companion Tutorial for users that have personal computers with the Windows operating system.*

The Tutorial steps include:

1. Installing an Integrated Development Environment (IDE) on Mac.
2. Obtaining reference design example project software. This will specifically include a sensor Data Logging system.
3. Usage of the IDE to Import, Build, Run, Debug and Flash the SensorTile board to run the example Data Logging project.

For more information regarding the SensorTile board, please open the following link on a web-browser on your Mac.

www.st.com/sensortile

2.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Mac with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 4 of 32

49

3. Integrated Development Environment Installation

This portion of the document will guide users through the System WorkBench Integrated Development Environment (IDE) installation process.

3.1.Prerequisite Steps

1. Open the following link to register an account with the OpenSTM32 community.

   http://www.openstm32.org/tiki-register.php

2. Update your Mac's operating system (OS) to OS X 10.10 Yosemite or a newer version.

3. Download and Install the latest version of Xcode from the App store. Open the following link on a web-browser on your Mac for more details.

   https://developer.apple.com/xcode/

4. Accept the license agreement by opening the terminal utility and issuing the following command.

   ```
   $ sudo xcodebuild –license
   ```

   When prompted to enter your password, enter the password used to log into your user account on your Mac.

   Press the **[Space]** key to scroll through the license.

   Once you reach the end of the license, type *agree* followed by the **[Enter]** key.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 5 of 32

3. Integrated Development Environment Installation

This portion of the document will guide users through the System WorkBench Integrated Development Environment (IDE) installation process.

3.1. Prerequisite Steps

1. Open the following link to register an account with the OpenSTM32 community.

   http://www.openstm32.org/tiki-register.php

2. Update your Mac's operating system (OS) to OS X 10.10 Yosemite or a newer version.

3. Download and Install the latest version of Xcode from the App store. Open the following link on a web-browser on your Mac for more details.

   https://developer.apple.com/xcode/

4. Accept the license agreement by opening the terminal utility and issuing the following command.

   ```
   $ sudo xcodebuild –license
   ```

   When prompted to enter your password, enter the password used to log into your user account on your Mac.

   Press the **[Space]** key to scroll through the license.

   Once you reach the end of the license, type *agree* followed by the **[Enter]** key.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

3.2. Install System WorkBench (IDE)

1. Open the following link on a web-browser on your Mac to download the System Workbench Installer.

   https://drive.google.com/open?id=1Tgtktoqpkq_5b71bX4CAHy1_hwkIde7T

2. Move the installer (*install_sw4stm32_macos_64bits-v2.2.run*) to Desktop.

3. Open your **Terminal** on Mac.

4. Navigate into Desktop directory.

   $ cd Desktop

5. Change the permission of the installer file.

   $ chmod 755 install_sw4stm32_macos_64bits-v2.2.run

6. Disable the gate keeper restriction on your Mac.

   $ sudo spctl --master-disable

7. Install System WorkBench.

   $ ./install_sw4stm32_macos_64bits-v2.2.run

   Then, you should see the installation information in terminal like Figure 2.

   ```
   |mbp-194:Desktop cliccuser$ ./install_sw4stm32_macos_64bits-v2.2.run
   Checking integrity... (could take a while, -m to bypass)
   Extracting JRE... done
   Logging initialized at level 'INFO'
   Commandline arguments:
   Detected platform: mac_osx,version=10.12.6,arch=x86,symbolicName=null,javaVersio
   n=1.8.0_131
   ```
   Figure 2: System WorkBench Installation information in terminal.

8. Wait for a few seconds, you will see a GUI installation interface popup as Figure 3.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 6 of 32

52

www.manaraa.com
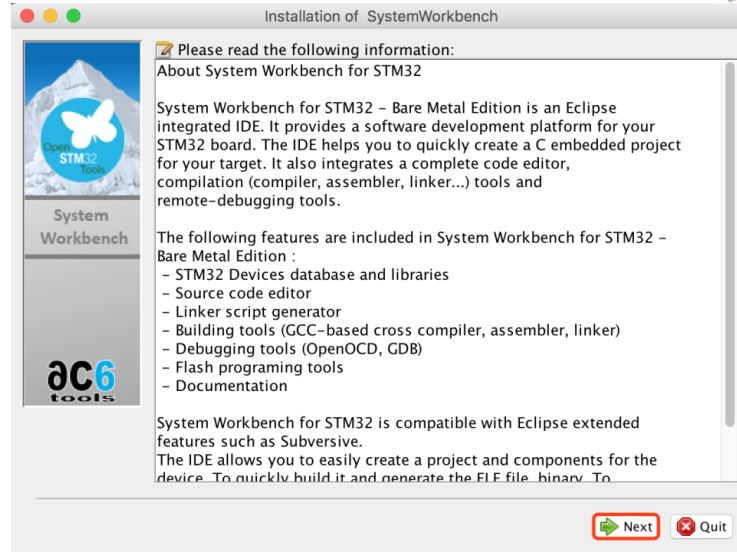
Figure 3: GUI installation interface of System WorkBench

9. Click on Next to continue. Accept all the license agreement and click next to continue as Figure 4.
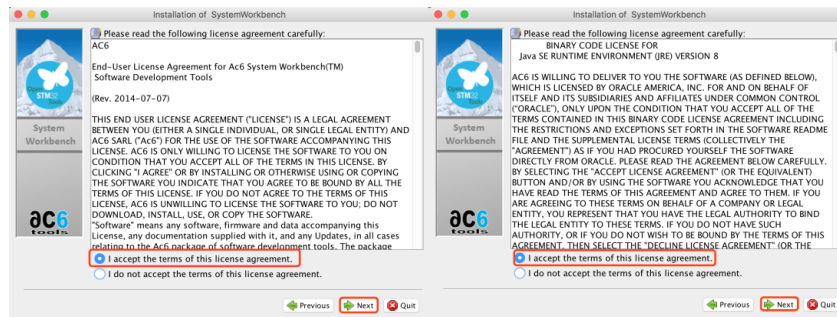
Figure 4: Agree the license agreement for System WorkBench.

10. Use its **default** installation directory and click next to continue. See Figure 5.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms
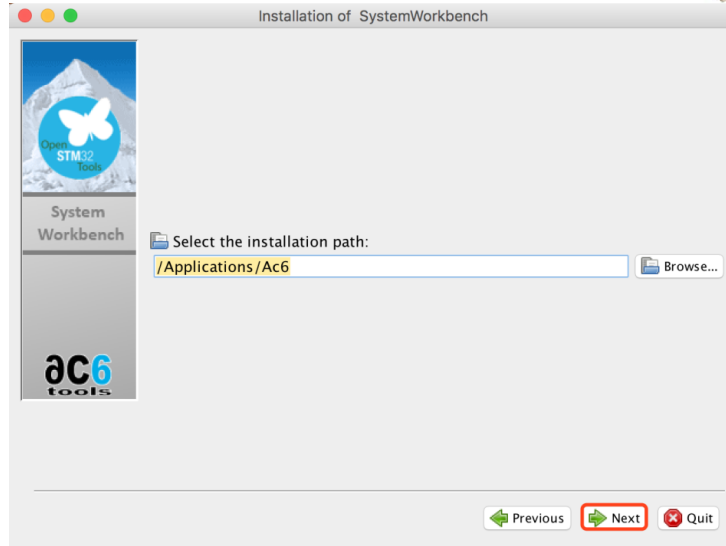
Figure 5: Use default installation directory

11. Click OK in the pop-up window. See Figure 6.

Figure 6: Create the installation directory

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms
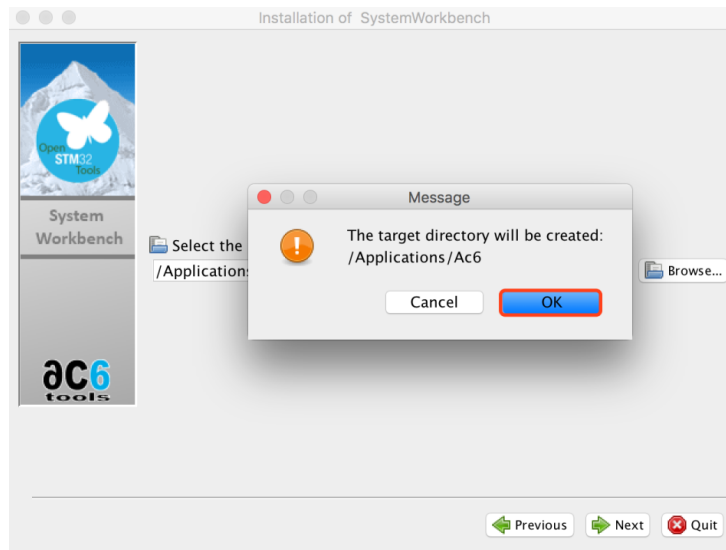
12. Make sure that all installation options have been checked and click next to continue. See Figure 7.
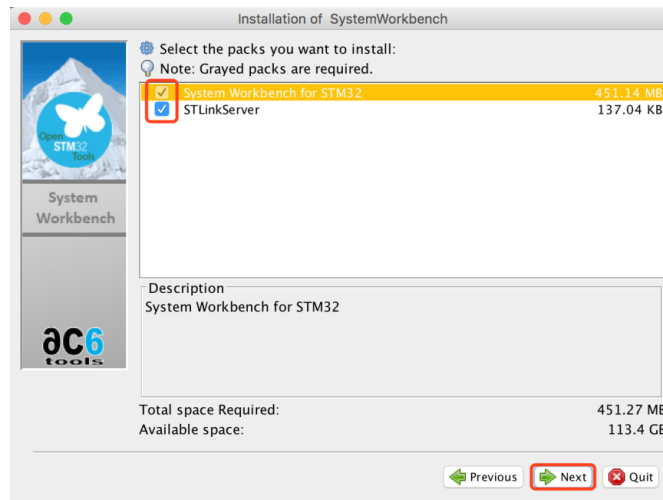
Figure 7: Install everything in the package

13. Wait until the installation has been finished. See Figure 8. **_IF THE INSTALLATION IS STUCK at 2/2, CHECK YOUR TERMINAL AND ENTER YOUR MAC PASSWORD IF IT REQUIRES._**
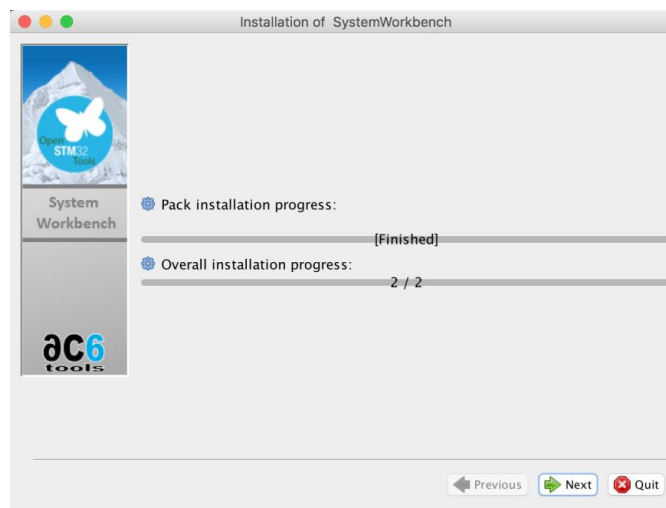
Figure 8: Installation has been finished

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

14. Open your *Finder* and find the *System WorkBench* according to Figure 9 (*Applications-> Ac6 -> SystemWorkBench*).
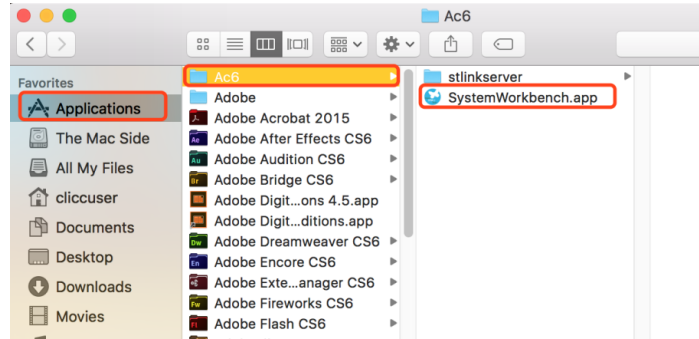
Figure 9: System WorkBench application location

15. Make sure your computer is connected to internet and double-click SystemWorkbench.app. Use the default workspace directory and wait for the final step of installation as Figure 10.
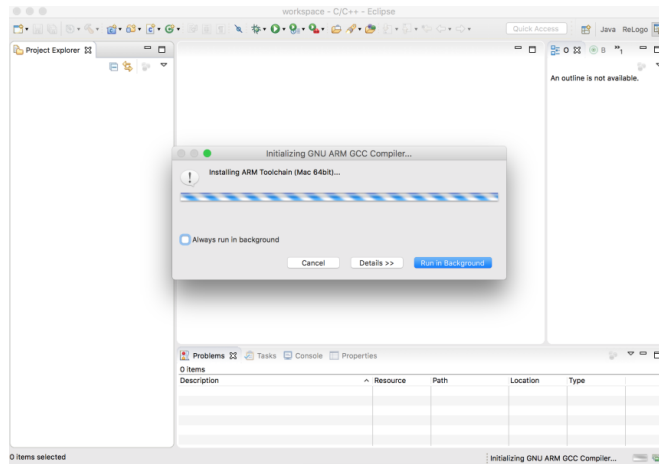
Figure 10: Last step of System Workbench (IDE) installation

16. After step 15, you System Workbench (IDE) is ready to go. System Workbench is an Eclipse with proper configuration and compiler tools.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

4. ST-Link Utilities for STM32

This section will guide users through the process of downloading and installing a flash tool.

1. Open a terminal on your Mac. To do this, open spotlight search by clicking the magnifying glass in the top right-hand corner of your Mac toolbar (see Figure), and search for "Terminal". Double click the result highlighted in the red box in Figure .
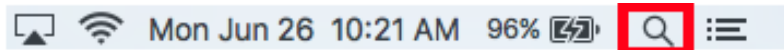
Figure 11: Opening Spotlight search on Mac.
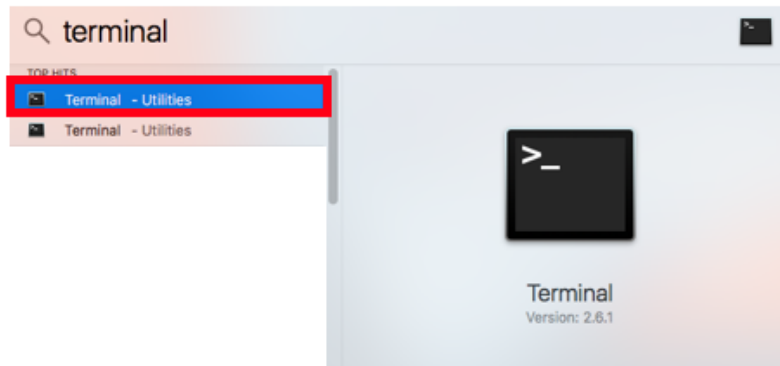
Figure 12: How to open the "Terminal" application on a Mac.

2. Follow the instructions online at https://brew.sh/ to install Homebrew (a package manager). Alternatively, issue the command shown below in the terminal session from step 1. Follow the onscreen prompts.

```
$ /usr/bin/ruby -e "$(curl –fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Note that the above command appears as one line.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Figure 13: Installing Homebrew.

3. Issue the following command in the terminal session from step 1.

```
$ brew install stlink
```

4. To check for successful installation, type **st** into the terminal session from step 1 **without pressing [Enter]**.

5. While the prompt displays **st**, Press **[Tab]** twice and the terminal session will list some programs through the auto-complete feature. If the terminal session displays the following items (highlighted in the red box in Figure), the installation completed successfully.

Figure 14: Three st tools should appear when using the autocomplete feature.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 12 of 32

58

www.manaraa.com

5. Example Data Logging Project

5.1. Download STSW-STLKT01 v1.2.0

Follow the instructions below to download an existing data-logging project for the SensorTile.

First, please note that this Tutorial supported Version 1.2.0 of the SensorTile Firmware, STSW-STLKT01

Instructions to download firmware from STMicroelectronics website.

1. Open the following link on a web-browser on your Mac.

   http://www.st.com/content/st_com/en/premium-content/sensortile-curriculum-stsw-stlkt01_zip.html

2. Scroll to the bottom of the page, and click "Get Software" for the entry **STSW-STLKT01**.

GET SOFTWARE

Part Number	Software Version	Marketing Status	Supplier	Order from ST
STSW-STLKT01	1.2.0	Active	ST	Get Software

Figure 15: This figure depicts what to click in order to acquire the example project files.

3. When the license agreement appears (Figure), read through it.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 13 of 32

59

After reading the agreement, click "Accept" at the top of the page.

Figure 16: License agreement that appears once you click "Get Software".

4. Either register an account, or enter your name and email address when prompted by the popup depicted in Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 14 of 32

60

www.manaraa.com

Then, click "Download" at the bottom of the popup screen.

Get Software

If you have an account on my.st.com, login and download the software without any further validation steps.

Login/Register

If you don't want to login now, you can download the software by simply providing your name and e-mail address in the form below and validating it.

This allows us to stay in contact and inform you about updates of this software.

For subsequent downloads this step will not be required for most of our software.

First Name:

Last Name:

E-mail address:

Please enter a valid e-mail address.

☐ I would like to stay up to date with ST's latest products and subscribe to the ST newsletters.

Download

Figure 17: Dialog box that collects developer information prior to enabling acquisition of example project files.

5. The popup box should resemble the Figure below if all steps have proceeded successfully.

Your registration has been successfully submitted!

To validate your e-mail and start the download, please click on the link inside the e-mail that has been sent to you. This link will be valid for 24 hours. Please check your spam filters in case you did not receive the e-mail.

Figure 18: This is a successful registration notification box.

6. Check your email **inbox** and **spam** folders for the email. Follow the link that you were sent to download the example code. The email should resemble Figure .

Note: this link will not function if your web-browser is in "Incognito" or "Private" mode.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 15 of 32
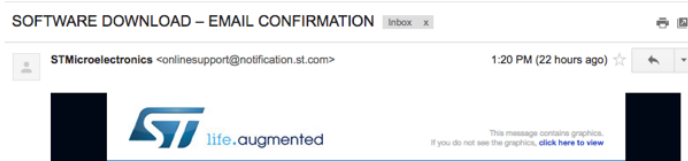
Figure 19: Email from STMicroelectronics containing weblink to enable access to download example project.

7. Extract the contents of the archive. The default output folder name should be **v1.2.0**. For consistency with this tutorial, please extract the archive into the Downloads directory.

The full path of the directory should resemble "/Users/<username>/Downloads/v1.2.0".

Figure 20: Contents of folder "v1.2.0".

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 16 of 32

62

www.manaraa.com

5.2. Import

Follow the instructions below in order to import the example project into the System WorkBench IDE. Note: while we call this System WorkBench, it will be listed as "Eclipse" in your applications.

1. Open the System WorkBench application.

2. Use the default workspace directory, and click "Ok". See Figure 21.
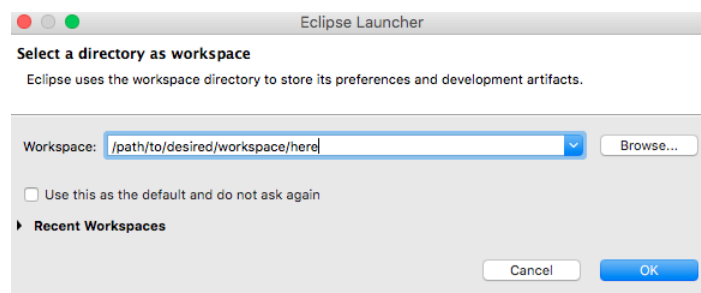
Figure 21: Selecting your workspace directory. It is recommended that you use the default location. It will resemble the form "/Users/<username_here>/Documents/workspace".

3. If you see a welcome screen, close it by clicking the "x" in the top left corner. See Figure .

Figure 22: Closing the welcome splash page.

4. Once you are on the main screen of System WorkBench, click "File > Import …" to begin importing the project. See Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 17 of 32

63

www.manaraa.com
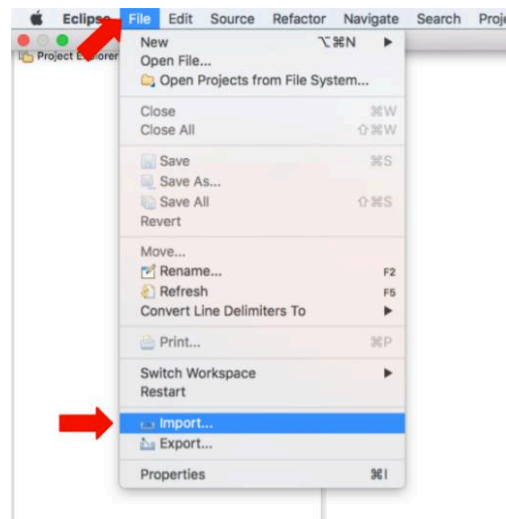
Figure 23: Importing an existing project into the workspace.

5. When the new window popup appears, click double click the entry "General". See Figure .

6. Double click the "Existing Projects into Workspace" option that appears under "General". See Figure .
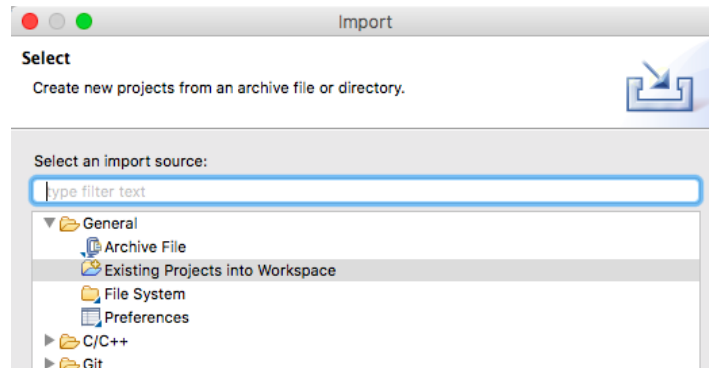
Figure 24: Instructions on the "Select" dialog box.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

7. **DO NOT CLICK FINISH UNTIL YOU COMPLETELY READ THROUGH STEP 8**.

   There are three projects in the v1.2.0 folder. If you import more than one project at a time, compilation will fail. This is because the three projects are dependent on other files in the folder.

   On the new popup window, enter the path to the folder **v1.2.0** in the text field "Select root directory". **DO NOT CLICK FINISH**.
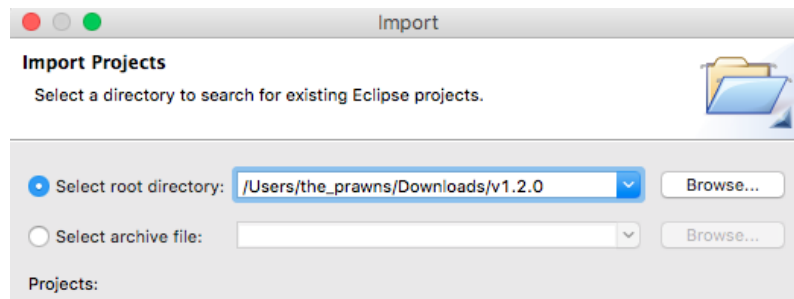
Figure 25: Selecting the folder where the existing project was extracted.

8. The projects box should be populated with 3 projects.
   - 8.1. AudioLoop (DE-SELECT THIS).
   - 8.2. BLE_SampleApp (DE-SELECT THIS).
   - 8.3. DataLog (Keep this selected).

   Uncheck the boxes next to AudioLoop and BLE_SampleApp.

Figure 26: Only selecting one project from the existing files to import into the workspace.

If you have selected more than one project and clicked "Finish", you will need to open your workspace directory on the Finder application on your Mac and manually delete all of the files.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

```
'SensorTile/Applications/AudioLoop/SW4STM32/STM32L4xx-SensorTile)
'SensorTile/Applications/BLE_SampleApp/SW4STM32/STM32L4xx-SensorTile)
'SensorTile/Applications/DataLog/SW4STM32/STM32L4xx-SensorTile)
```

Figure 27: Close-up image of text from Figure.

Click "Finish" with **only one project** selected.

9. Your screen should resemble the screenshot depicted below.
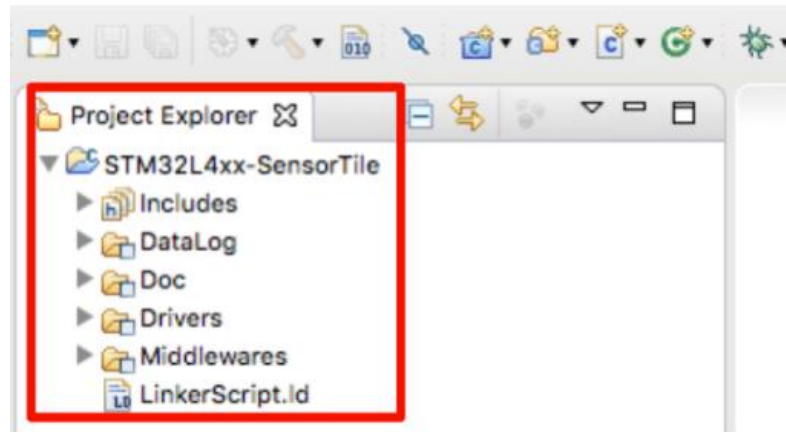
Figure 28: Contents of "Project Explorer" after successfully importing the example project files.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 20 of 32

66

5.3. Build

This portion of the document will guide users through the process of compiling the C-source code into a binary file that can be loaded onto the SensorTile board.

1. Select STM32L4xx-SensorTile -> DataLog in Project Explorer. Then, click Project -> Build Project (see the Figure below).
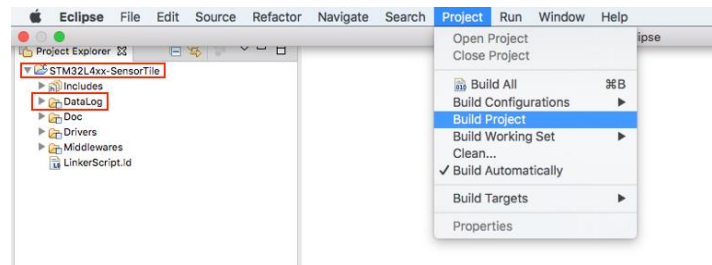
Figure 29: Compiling the source code for the example project.

2. If step 1 completed successfully, a folder named "Release" should appear and contain a **.bin** file. See the Figure below.
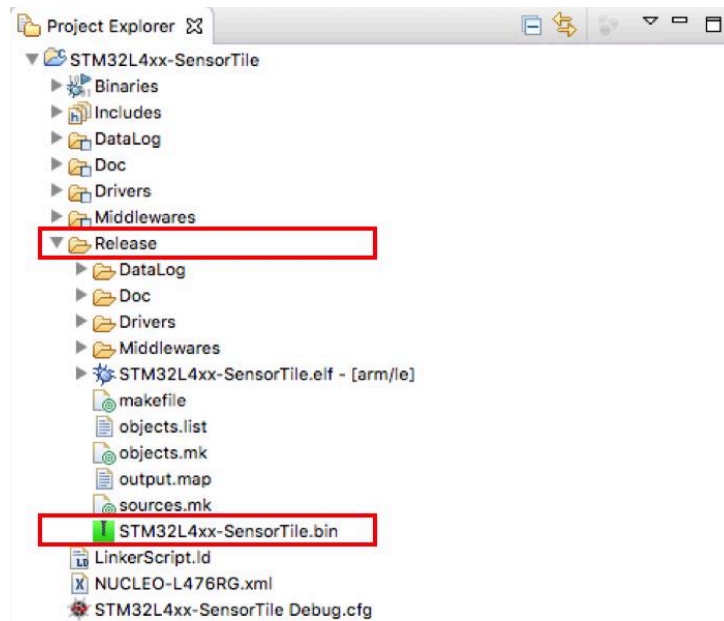
Figure 30: Project directory contents after successful compilation of project source code.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

5.4. SensorTile Hardware Platform

This section describes how to configure the hardware. Be very careful in this section, if the wire connections are not configured correctly, the boards could be permanently damaged. Do not proceed with the tutorial until an instructor has verified that your board is correctly configured.

1. Remove the Nucleo - L476RG board from its packaging.

2. Remove the CN2 Jumpers. (Remove both of them). See Figure .
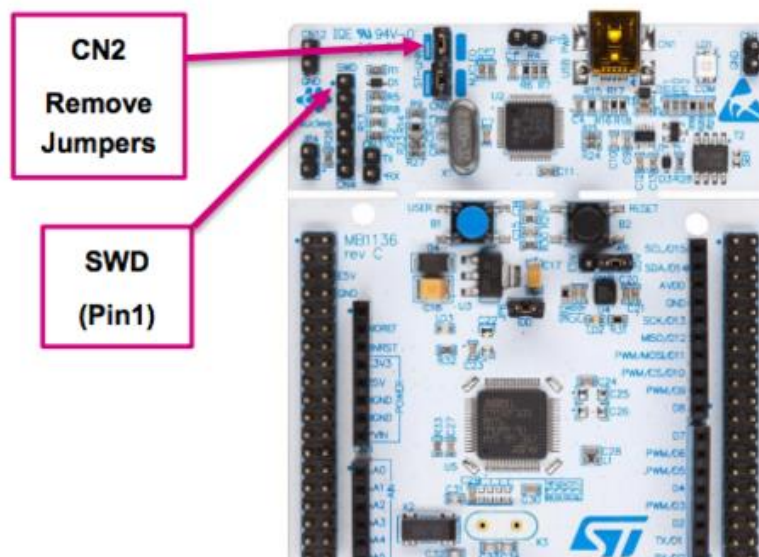
Figure 31: Removing the CN2 Jumpers from the Nucleo-L47RG board.

3. Place the SensorTile on the larger evaluation board. Ensure the orientation of the SensorTile matches Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Figure 32: Ensure the orientation of the SensorTile on the larger evaluation board matches this figure. There should be a green protrusion with a little metalic hole right next to the ST logo.

4. Connect the boards by attaching an SWD connector from the Nucleo board to the SensorTile board. Be **very** careful in this step. Please examine the figures in the next page. Do not proceed with this tutorial until you have verified with an instructor that the hardware is correctly configured.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms
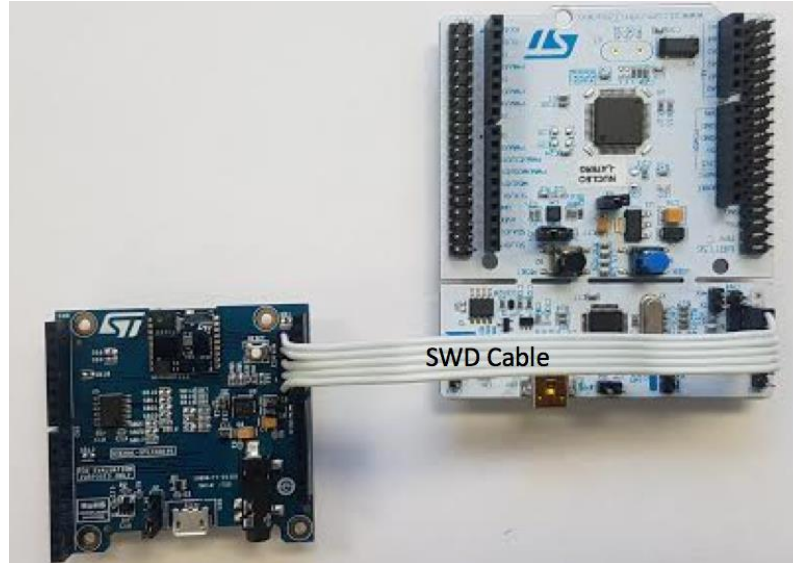
Figure 33: First figure depicting correct hardware configuration. Ensure that the SWD cable is oriented such that the pins marked "SWD" are connected via the same wire.
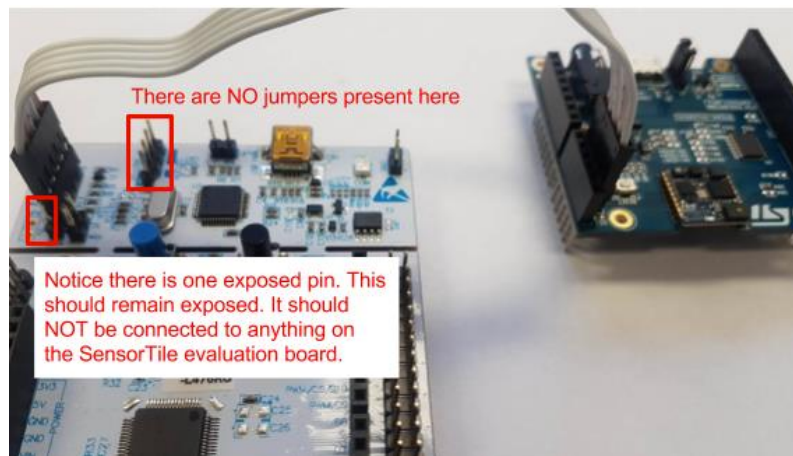
Figure 34: Second figure depicting correct hardware configuration.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 24 of 32

70

www.manaraa.com

5.5.Debug

This section will guide users through the process of running the **DataLog** application in debug mode on the SensorTile board.

1. Attach a mini-USB cable from the **Nucleo** to your Mac. Attach a micro-USB cable from the **SensorTile** to your Mac. See Figure .
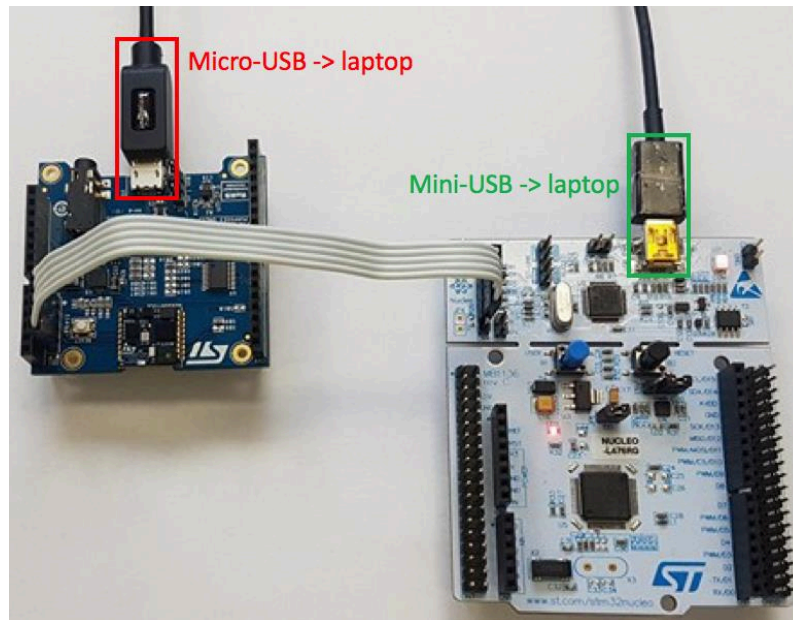
Figure 35: Establish a USB-wireline connection between each board and your Mac.

2. Select the project folder, "STM32L4xx-SensorTile", in Project Explorer. Then, click "Run -> Debug As -> AC6 STM32 C/C++ Application". See Figure in the next page.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms
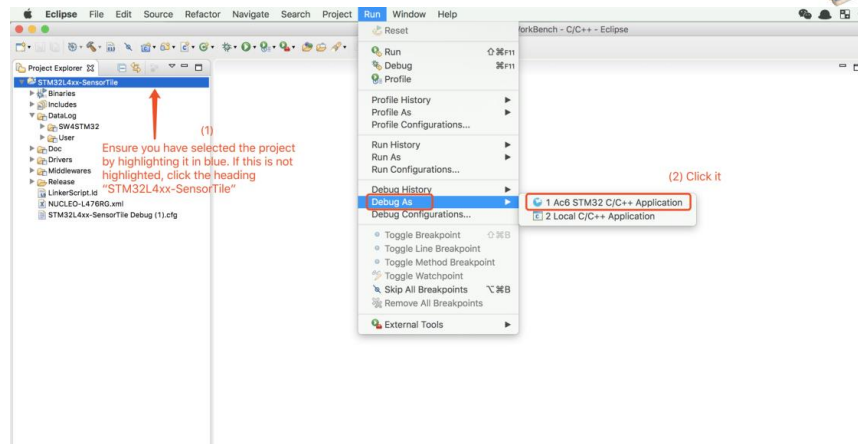
Figure 36: Launching the DataLog program in debug mode.

3. A dialog box may appear. Click "Yes" on this dialog box.
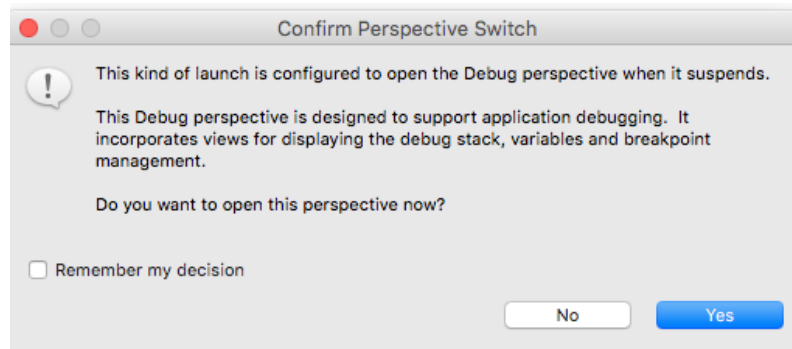
Figure 5: Click "Yes" on this dialog box.

4. A debugging interface should appear.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 26 of 32

72

www.manaraa.com
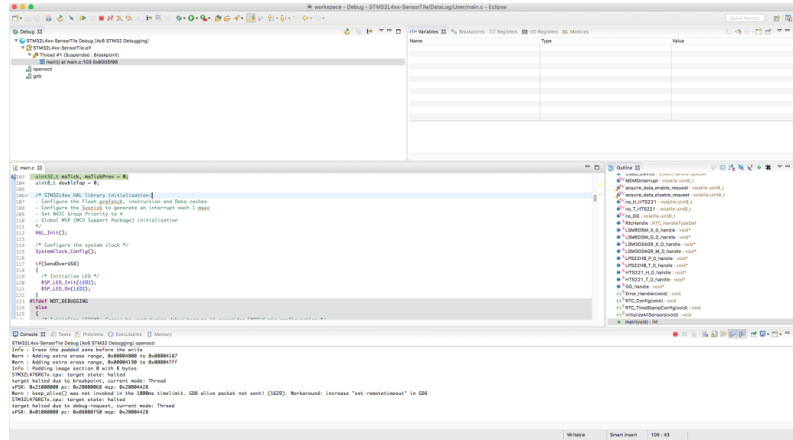
Figure 38: Debugging interface.

If you wish to switch back to the C/C++ view, click the top right corner button labeled "C/C++". See the Figure below.
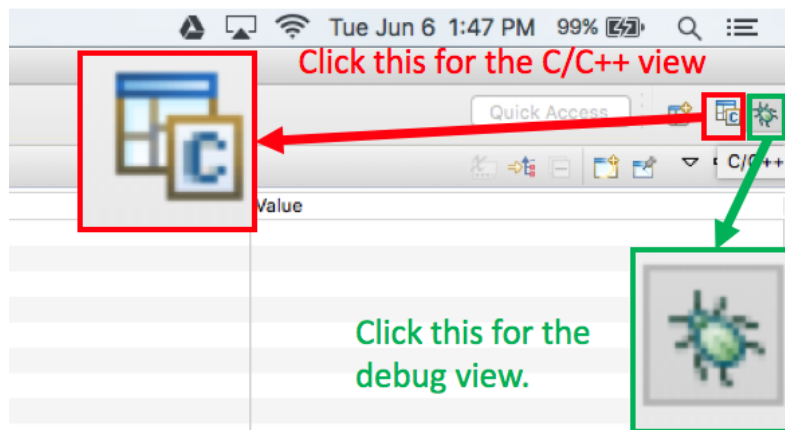
Figure 39: To enter the "C/C++" view, click the button highlighted in the red box. To enter the "debug mode" view, click the button highlighted in the green box.

5. Examine your SensorTile device. Notice how none of the LED's are activated.

6. Press the "green arrow with the yellow bar on its left" button that indicates "Resume" when you mouse over it. See Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms
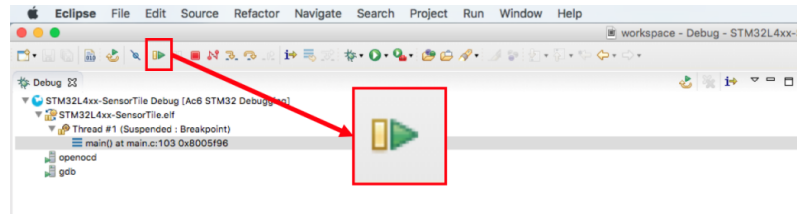
Figure 40: Starting the application in debug mode.

7. Examine your SensorTile device. Notice how the orange LED is rapidly blinking.

8. Open a terminal and issue the command shown in Figure .

```
$ ls /dev/cu.usb*
```
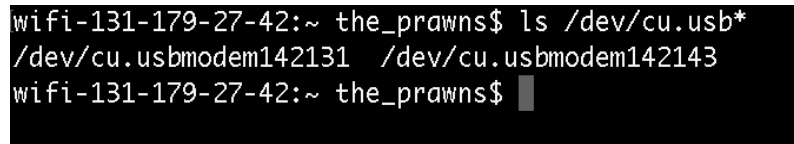

Figure 41: Inspecting the devices connected via USB.

Notice how there are two devices listed.

The device ending in "3" is the Nucleo board.

The device ending in "1" is the SensorTile.

9. **If you receive an error in this step, please proceed to step 10.**

Inspect the data the SensorTile is sending over serial wireline USB connection by issuing the command shown in Figure . Make sure this command is edited to suit the name of the device as it appears in your terminal session. The output should resemble Figure .

```
$ screen /dev/cu.usbmodem<SENSORTILE_ID> 9600
```
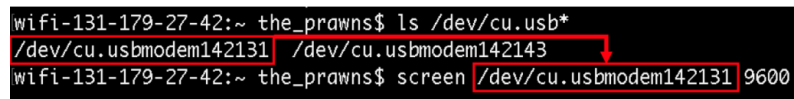

Figure 42: Examining the output from the SensorTile.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 28 of 32

Figure 43: Data being captured by the SensorTile.

10. If you receive the error shown in Figure , pause the program by clicking the two parallel yellow bars next to the run button (see Figure), unplug the SensorTile board from the Mac, wait 10 seconds, and reconnect it. Once the cable is reconnected, repeat steps 6-9.

`Cannot open line '/dev/cu.usbmodem142131' for R/W: Resource busy`

Figure 44: Common error message when examining data from the SensorTile using "screen".

Figure 45: Pause button to temporarily halt execution of code on the SensorTile board.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 29 of 32

5.6. Flash

This section will guide users through the process of uploading a compiled binary file onto the SensorTile for execution. Once the program is uploaded to the board, it will run every time the SensorTile is supplied with power. The SensorTile will no longer need to be connected to the Nucleo board, nor will users have to interface with System WorkBench (the IDE).

1. Terminate and remove all existing applications on the SensorTile board as shown in Figure .

   Note: Ensure to remove ALL existing applications. Figure only contains one application. If there are more, delete all of them.
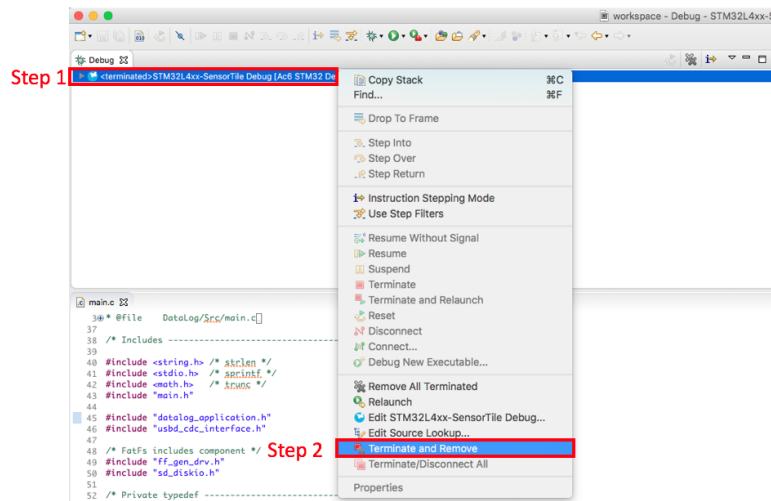
Figure 46: Terminating and removing existing applications on the SensorTile board.

2. Open a terminal session, and navigate to the **v1.2.0** directory.

```
wifi-131-179-27-42:~ the_prawns$ cd /Users/the_prawns/Downloads/v1.2.0/
wifi-131-179-27-42:v1.2.0 the_prawns$
```

Figure 47: Navigating to the v1.2.0 directory.

3. Issue the command shown below and depicted in Figure .

```
$ cd Projects/SensorTile/Applications/DataLog/SW4STM32/STM32L4xx-SensorTile/Release
```

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Figure 48: Navigating to the directory containing the binary file compiled earlier in this tutorial.

4. Ensure that the directory contains the same files as seen in Figure .

5. Issue the command shown below and depicted in Figure .

```
$ st-flash write STM32L4xx-SensorTile.bin 0x8004000
```


Figure 49: 'st-flash' command and resulting output to terminal.

6. Navigate back to the **v1.2.0** directory as seen in step 2.

7. Issue the command below and depicted in Figure . Ensure the contents match Figure .

```
$ cd Utilities/BootLoader/STM32L476RG
```


Figure 50: Navigating to ".../v1.2.0/Utilities/BootLoader/STM32L476RG". Contents of said directory.

8. Issue the command below and depicted in Figure .

```
$ st-flash write BootLoaderL4.bin 0x8000000
```

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development
Environment and DataLog Project Example for Apple Mac Platforms

Figure 51: Flashing the bootloader.

9. Disconnect the USB cables connecting the SensorTile and the Nucleo board's to the Mac. This will power down the devices, enabling us to safely disconnect the SWD connection between the SensorTile board and the Nucleo board.

10. Disconnect the SWD cable connecting the SensorTile board to the Nucleo board.

11. Reconnect SensorTile board to your Mac. Do not reconnect the Nucleo board to your Mac.

12. Notice that the SensorTile device immediately starts streaming data over serial USB connection to your Mac by examining the blinking LED.

13. Examine the data transmitted from the SensorTile by repeating step 9 from 5.5 Debug.

14. You will now observe complete system operation with a stand-alone, automatically operating SensorTile IoT system.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Apple Mac Platforms

Page 32 of 32

78

APPENDIX B

Tutorial 1

Introduction to STMicroelectronics Development Environment

Windows Version

STMicroelectronics SensorTile Tutorial:
Introduction to STMicroelectronics Development Environment and DataLog Project Example
for
Windows Platforms

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 1 of 39

Table of Contents

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
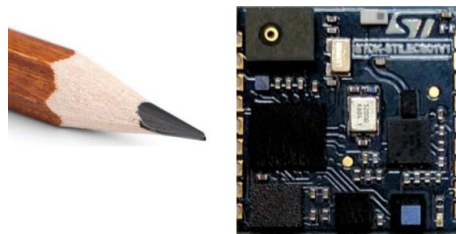
1. Introduction

The SensorTile is a new Internet of Things (IoT) system provided by STMicroelectronics integrating state-of-the-art processor, wireless interfaces, and sensor systems. The SensorTile can form the foundation for wearable consumer devices, wearable medical devices, residential IoT systems and vehicle IoT systems.

The SensorTile system provides an exceptionally powerful and well-supported platform for introduction to IoT technology. The SensorTile is remarkably compact as shown in Figure 1.

Figure 1. The STMicroelectonics SensorTile Platform with a pencil reference indicating its compact geometry.

The SensorTile includes these components:

1) The SensorTile Processor System is an STM32L4 microprocessor based on the ARM Cortex M4 system. This provides introduction to the ARM processor architecture that is deployed on nearly every smartphone on earth.
2) The SensorTile Sensors includes:
    a) The LSM6DSM combining microaccelerometer and microgyroscope.
    b) The LSM303AGR combining microaccelerometer and magnetometer for compass heading
    c) The LPS22HB barometric pressure sensor for determination of altitude and atmospheric pressure.
    d) The MP34DT04 microphone
3) The SensorTile also includes a Bluetooth Low Energy (Bluetooth Smart) wireless interface the BlueNRG-MS system.
4) The SensorTile also includes non-volatile flash storage that stores the executable code that enables IoT system operation.
5) The SensorTile also includes a cradle accessory with additional features including:
    a) SD Card Flash Storage System
    b) STC3115 Battery Monitor providing detailed energy monitoring for the SensorTile
    c) HTS221 Humidity and Temperature environmental sensors

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

2. Introduction to This Tutorial

This Tutorial introduces the development environment for the SensorTile system.

Development environments are essential to development of software for IoT systems and other products. These provide support to developers for both creation of systems, testing, debugging, and installation of software systems on platforms.

This development environment is referred to as an Integrated Development Environment (IDE). This includes all software tools required to create a software distribution for the SensorTile, compile this software system into the processor instruction set using a Build capability, execute this system using a Debug capability, and create an "image" file that can be installed in the SensorTile non-volatile storage.

This tutorial is intended for users that have Windows platforms, and will guide users through the tasks listed below. *Please note that there is a companion Tutorial for users that have Apple Mac Platforms.*

The Tutorial steps include:

1. Installing an Integrated Development Environment (IDE) on Window.
2. Obtaining reference design example project software. This will specifically include a sensor Data Logging system.
3. Usage of the IDE to Import, Build, Run, Debug and Flash the SensorTile board to run the example Data Logging project.

For more information regarding the SensorTile board, please open the following link on a web-browser on your PC.

www.st.com/sensortile

2.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Windows with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 4 of 39

3. Integrated Development Environment Installation

This portion of the document will guide users through the System WorkBench Integrated Development Environment (IDE) installation process.

3.1. Prerequisite Steps

1. Open the following link to register an account with the OpenSTM32 community.

   http://www.openstm32.org/tiki-register.php

2. Update your Windows operating system (OS) to the latest version (Windows 10 or newer). Open the following link on a web-browser on your Windows for more details.

   https://support.microsoft.com/en-us/help/12373/windows-update-faq

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 5 of 39

3.2. Install System WorkBench (IDE)

1. Open the following link on a web-browser on your PC to download the System Workbench Installer.

   https://drive.google.com/open?id=1SXOPiF9RF3OSRB8lELfcFkH3tLJbgo5E

2. Move the installer (*install_sw4stm32_win_64bits-v2.2*) to Desktop.

3. Install System WorkBench by double-clicking the installer. Then, you should see the installation information like Figure 2.
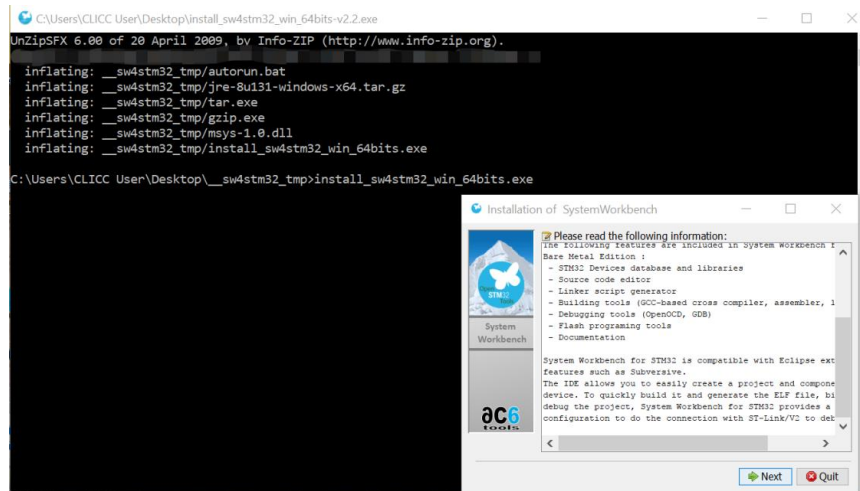

Figure 2: System WorkBench Installation information

4. Wait for a few seconds, you will see a GUI installation interface popup as Figure 3.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 6 of 39

85
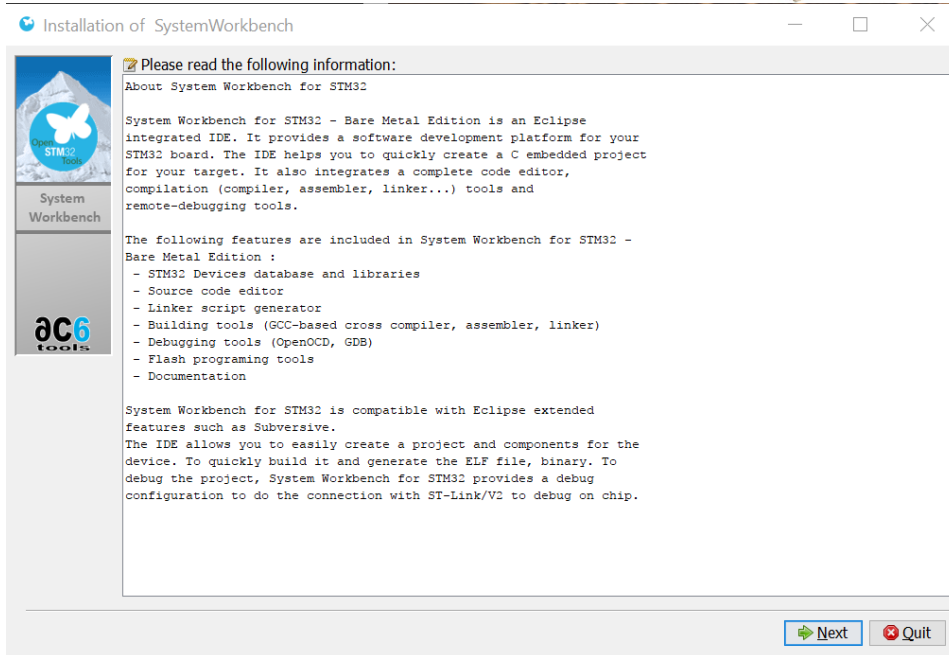
Figure 3: GUI installation interface of System WorkBench

5. Click on Next to continue. Accept all the license agreement and click next to continue as Figure 4.
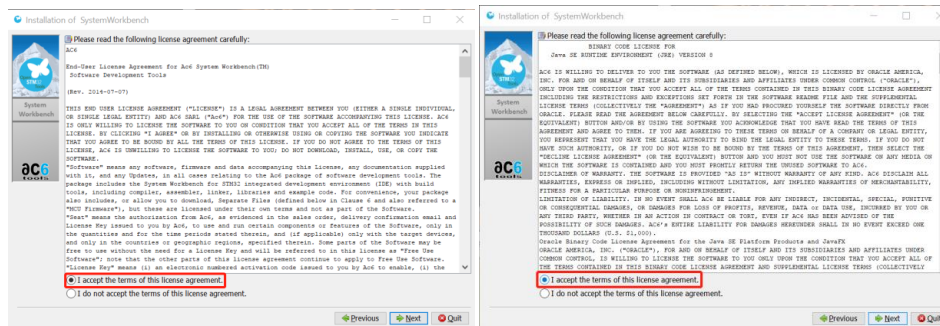
Figure 4: Agree the license agreement for System WorkBench.

6. Use its **default** installation directory and click next to continue. See Figure 5.
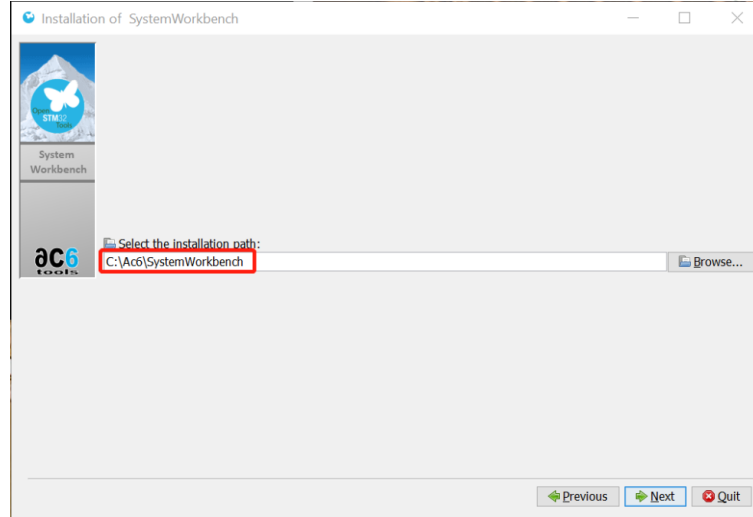
STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Figure 5: Use default installation directory

7. Click OK in the pop-up window. See Figure 6.
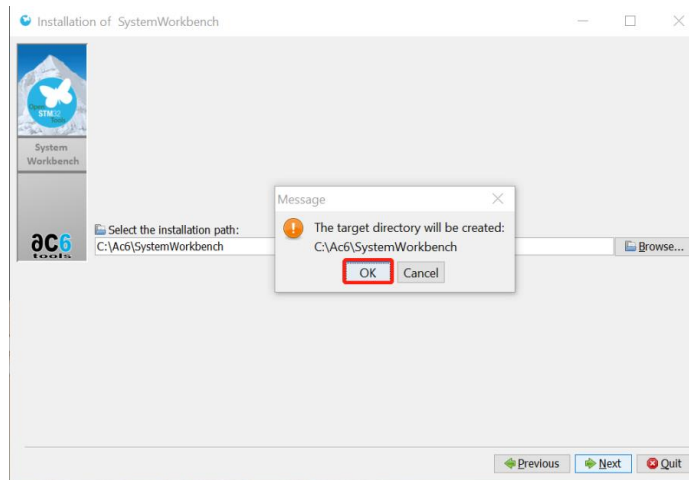
Figure 6: Create the installation directory

8. Make sure that all installation options have been checked and click next to continue. See Figure 7.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

87

www.manaraa.com
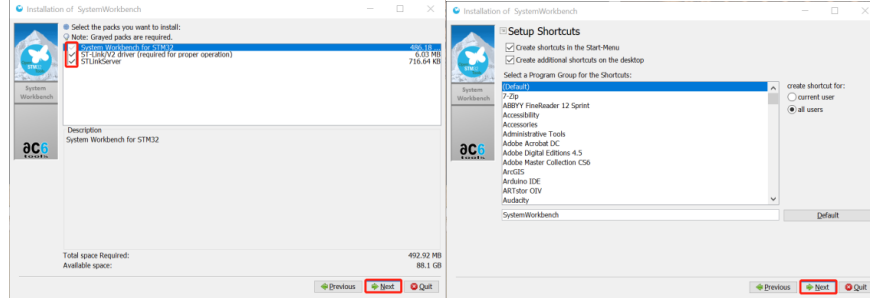
Figure 7: Install everything in the package

9. Wait until the installation has been finished. See Figure 8.
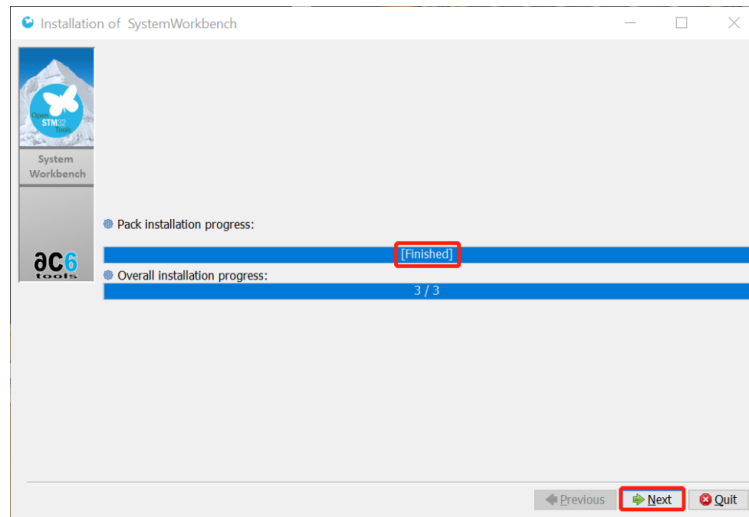
Figure 8: Installation has been finished

10. Install the driver according to the installer interface.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
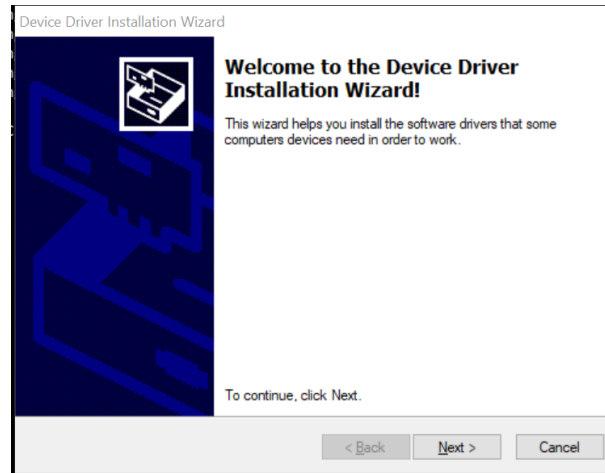
Figure 9: Driver installation

11. Make sure your computer is connected to the Internet and double-click SystemWorkbench shortcut on your desktop. Use the default workspace directory and wait for the final step of installation as Figure 10.
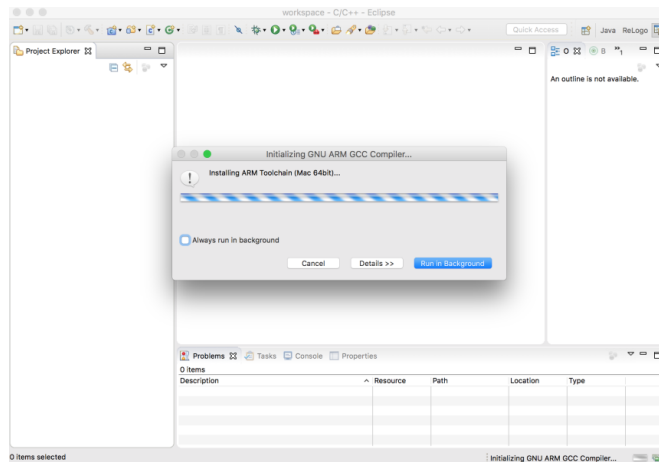
Figure 10: Last step of System Workbench (IDE) installation

12. After step 11, you System Workbench (IDE) is ready to go. System Workbench is an Eclipse with proper configuration and compiler tools.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

89

3.3. ST-Link Utilities for STM32

This section will guide users through the process of downloading and installing a flash tool.

1. Restart your PC.

2. Ensure that you have administrative privileges on your PC before proceeding.

3. Download the STM32 ST-LINK utilities from the following weblink. Follow a similar process to section "5.1 Download", but look for the file labelled **STSW-LINK004** instead of **STSW-STLKT01**.

   http://www.st.com/en/embedded-software/stsw-link004.html

4. Extract the contents of the archive labelled "en.stsw-link004.zip".

5. Double click file that was extracted to begin installation.

Name	Date modified	Type	Size
STM32 ST-LINK Utility v4.0.0 setup.exe	6/15/2017 2:15 PM	Application	24,103 KB

his PC > Downloads > en.stsw-link004
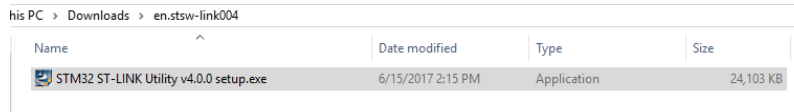
Figure 12: The file used to start the install process for the ST-LINK utilities.

6. Download the STM ST-LINK USB Driver from the following weblink. Follow a similar process to section "5.1 Download", but look for the file labelled **STSW-LINK009** instead of **STSW-STLKT01**.

   Copy and paste this link into your browser if it does not function as intended.

   http://www.st.com/content/st_com/en/products/embedded-software/development-tool-software/stsw-link009.html

7. Extract the contents of the archive labelled "en.stsw-link009.zip".

8. If you are using a 64-bit operating system, double click the file labelled "dpinst_amd64.exe" to begin installation. Otherwise, double click the file labelled "dpinst_x86.exe". See Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Figure 13: The file highlighted in blue is the installer used to install ST-LINK USB drivers.

9. Download the STM32 Virtual COM Port Driver from the following weblink. Follow a similar process to section "5.1 Download", but look for the file labelled **STSW-STM32102** instead of **STSW-STLKT01**.

    http://www.st.com/en/development-tools/stsw-stm32102.html

    From the extracted files, select the one whose filename contains "W8" if your system is Windows 8 or newer. Otherwise, select the one whose filename contains "W7".

    If you are using a 64-bit operating system, select the file ended with "64bits". Otherwise, select the one ended with "32bits".

    Launch the installer by double clicking the selected file.

    Use the default install location for consistency with the rest of this tutorial.

Figure 14: The ST32 Virtual COM Port Driver installers.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

4. PuTTy

PuTTY is an SSH and telnet client, developed originally by Simon Tatham for the Windows platform. PuTTY is open source software that is available with source code and is developed and supported by a group of volunteers.

PuTTy is the program that will enable us to see the data transmitted by the SensorTile board over serial USB connection to your personal computer.

1. Open the following web-link on your personal computer.

   https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

2. Download the correct executable binary file for your operating system as highlighted in Figure .

Figure 15: Downloading the correct PuTTy binary file.

3. Double click the file to launch it. Note: This file is an **EXECUTABLE BINARY FILE**, this file is NOT an installer. As such, make sure you do not delete this file after you execute it once. You may also want to create a shortcut to this application on your desktop for ease of access.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 13 of 39

5. Example Data Logging Project

5.1. Download

Follow the instructions below to download an existing data-logging project for the SensorTile.

First, please note that this Tutorial supported Version 1.2.0 of the SensorTile Firmware, STSW-STLKT01

Instructions to download firmware from STMicroelectronics websites.

1. Open the following link on a web-browser on your PC.

   http://www.st.com/content/st_com/en/premium-content/sensortile-curriculum-stsw-stlkt01_zip.html

2. Scroll to the bottom of the page, and click "Get Software" for the entry **STSW-STLKT01**.

GET SOFTWARE

Part Number	Software Version	Marketing Status	Supplier	Order from ST
STSW-STLKT01	1.2.0	Active	ST	Get Software

Figure 16: This figure depicts what to click to acquire the example project files.

3. When the license agreement appears (Figure), read through it.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 14 of 39

93

www.manaraa.com

After reading the agreement, click "Accept" at the top of the page.

License Agreement

ACCEPT

IMPORTANT-READ CAREFULLY: This Production Limited License Agreement ("PLLA") for ST materials is made between you on behalf of yourself or on behalf of any entity by which you are employed or engaged (collectively referred to in this PLLA as "You" or "Licensee") and STMicroelectronics International NV, a company incorporated under the laws of the Netherlands acting for the purpose of this PLLA through its Swiss branch located at 39, Chemin du Champ des Filles, 1228 Plan-les-Ouates, Geneva, Switzerland (hereinafter "ST"). Affiliates shall mean any corporation, partnership, or other entity that, directly or indirectly, owns, is owned by, or is under common ownership with ST, for so long as such ownership exists. For the purposes of the foregoing, "own", "owned," or "ownership" shall mean ownership of more than fifty percent (50%) of the stock or other equity interests entitled to vote for the election of directors or an equivalent governing body.
The ST materials licensed under this PLLA shall mean the software made available by ST and/or its Affiliates upon agreeing to this PLLA, including any associated Documentation (collectively the "Licensed Materials"). Documentation shall mean and include any comments, annotations, instructions, manuals, and other materials, whether in printed or electronic form, including without limitation installation manuals, user's guides, and programmer guides, related to any software made available under this PLLA. The Licensed Materials include any software updates, and supplements that ST and/or its Affiliates may provide You or make available to You after the date You obtain the

Figure 17: License agreement that appears once you click "Get Software".

4. Either register an account, or enter your name and email address when prompted by the popup depicted in Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

94

Then, click "Download" at the bottom of the popup screen.

Get Software

If you have an account on my.st.com, login and download the software without any further validation steps.

Login/Register

If you don't want to login now, you can download the software by simply providing your name and e-mail address in the form below and validating it.

This allows us to stay in contact and inform you about updates of this software.

For subsequent downloads this step will not be required for most of our software.

First Name:

Last Name:

E-mail address:

Please enter a valid e-mail address.

☐ I would like to stay up to date with ST's latest products and subscribe to the ST newsletters.

Download

Figure 18: Dialog box that collects developer information prior to enabling acquisition of example project files.

5. The popup box should resemble Figure if all steps have proceeded successfully.

Your registration has been successfully submitted!

To validate your e-mail and start the download, please click on the link inside the e-mail that has been sent to you. This link will be valid for 24 hours. Please check your spam filters in case you did not receive the e-mail.

Figure 19: This is a successful registration notification box.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

95

www.manaraa.com

6. Check your email **inbox** and **spam** folders for the email. Follow the link that you were sent to download the example code. The email should resemble Figure .

Note: this link will not function if your web-browser is in "Incognito" or "Private" mode.
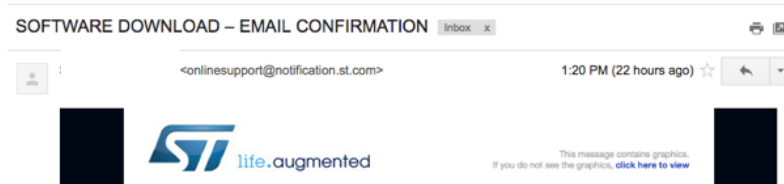
Figure 20: Email from STMicroelectronics containing weblink to enable access to download example project.

7. Extract the contents of the archive. The default output folder name should be **v1.2.0**. For consistency with this tutorial, please extract the archive into the Downloads directory.

The full path of the directory should resemble the following.

"C:\Users\<USERNAME>\Downloads\en.stsw-stlkt01\v1.2.0".

Note: if the path to the folder labelled **v1.2.0** has a space in it, move it to a location where the path does not contain any spaces.

The IDE will not be able to compile projects that have spaces in their paths.
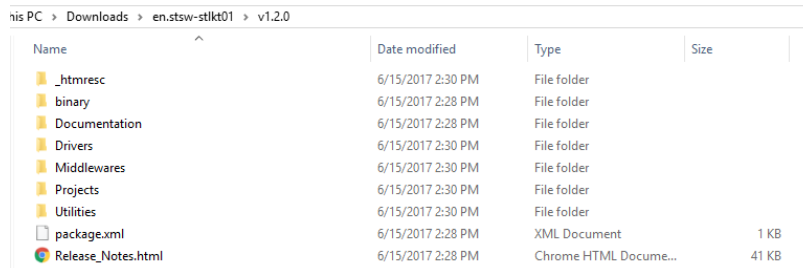
Figure 21: Contents of folder "v1.2.0".

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 17 of 39

96

5.2. Import

Follow the instructions below to import the example project into the System WorkBench IDE. Note: while we call this System WorkBench, it will be listed as "Eclipse" on your desktop.

1. Open the System WorkBench application.

2. Use the default workspace directory, and click "Ok". See Figure .

   Note: Ensure that the workspace you select does not contain any spaces. If the path to the workspace contains spaces, the IDE will not be able to compile the projects.
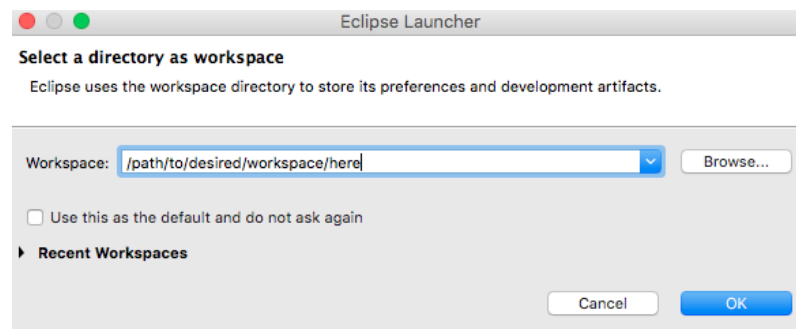
Figure 22: Selecting your workspace directory. It is recommended that you use the default location. It will resemble the form "C:\Users\<USERNAME>\workspace".

3. If you see a welcome screen, close it by clicking the "x" in the top left corner. See Figure .

Figure 23: Closing the welcome splash page.

4. Once you are on the main screen of System WorkBench, click "File > Import …" to begin importing the project. See Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
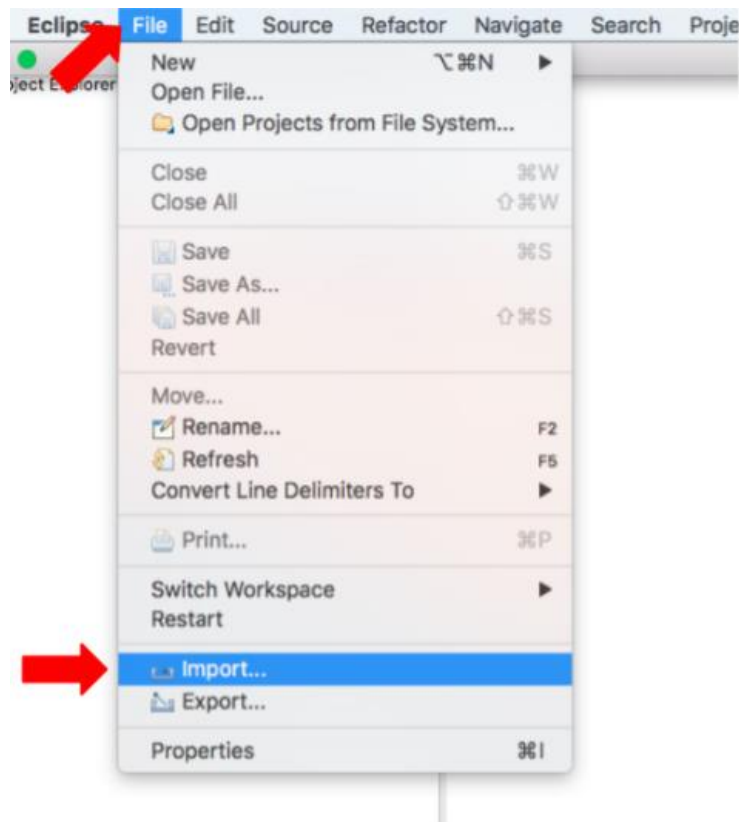
Figure 24: Importing an existing project into the workspace.

5. When the new window popup appears, click double click the entry "General". See Figure .

6. Double click the "Existing Projects into Workspace" option that appears under "General". See Figure .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 19 of 39

98

www.manaraa.com
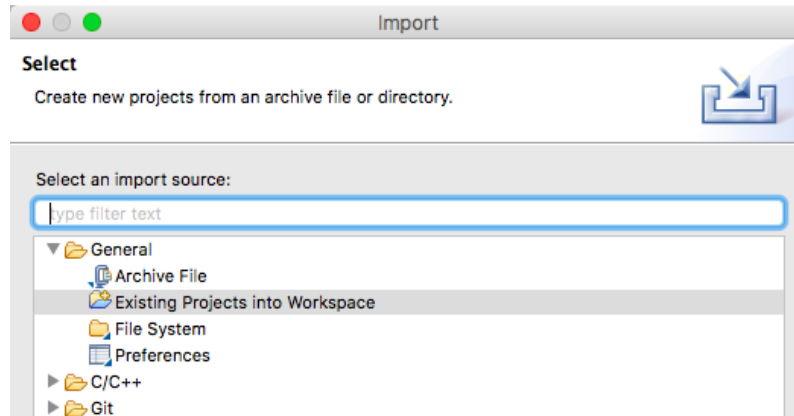
Figure 25: Instructions on the "Select" dialog box.

7. **DO NOT CLICK FINISH UNTIL YOU COMPLETELY READ THROUGH STEP 8**.

There are three projects in the v1.2.0 folder. If you import more than one project at a time, compilation will fail. This is because the three projects are dependent on other files in the folder.

On the new popup window, enter the path to the folder **v1.2.0** in the text field "Select root directory". **DO NOT CLICK FINISH**.
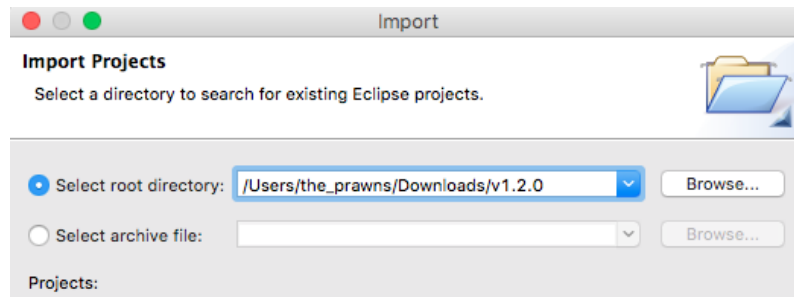
Figure 26: Selecting the folder where the existing project was extracted.

8. The projects box should be populated with 3 projects.
   8.1. AudioLoop (DE-SELECT THIS).
   8.2. BLE_SampleApp (DE-SELECT THIS).
   8.3. DataLog (Keep this selected).

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Uncheck the boxes next to AudioLoop and BLE_SampleApp.

Figure 27: Only selecting one project from the existing files to import into the workspace.

If you have selected more than one project and clicked "Finish", you will need to open your workspace directory on the Finder application on your Windows and manually delete all of the files.

```
'SensorTile/Applications/AudioLoop/SW4STM32/STM32L4xx-SensorTile)
'SensorTile/Applications/BLE_SampleApp/SW4STM32/STM32L4xx-SensorTile)
'SensorTile/Applications/DataLog/SW4STM32/STM32L4xx-SensorTile)
```

Figure 28: Close-up image of text from Figure .

Click "Finish" with **only one project** selected.

9. Your screen should resemble the screenshot depicted below.
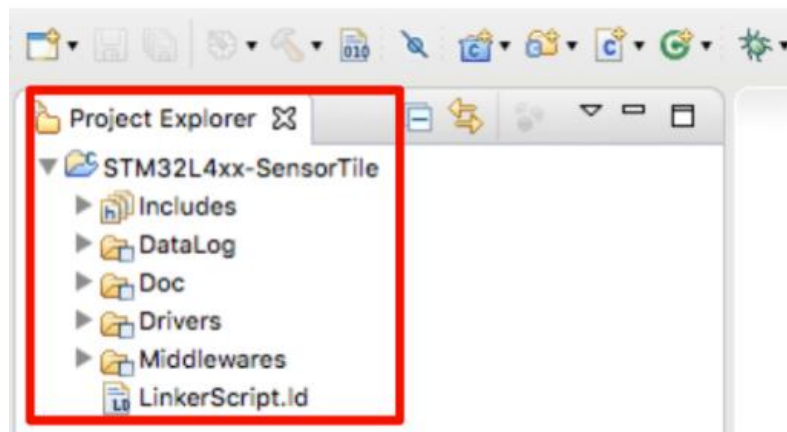
Figure 29: Contents of "Project Explorer" after successfully importing the example project files.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

5.3. Build

This portion of the document will guide users through the process of compiling the C-source code into a binary file that can be loaded onto the SensorTile board.

1. Select STM32L4xx-SensorTile -> DataLog in Project Explorer. Then, click Project -> Build Project (see Figure).
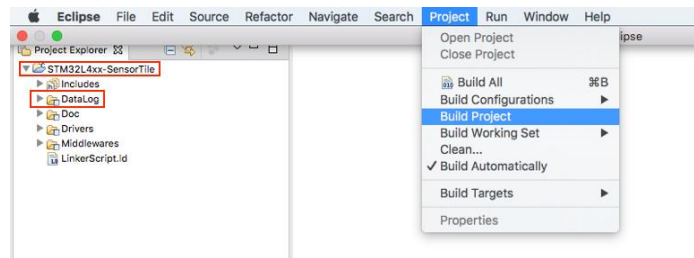
Figure 30: Compiling the source code for the example project.

2. If step 1 completed successfully, a folder named "Release" should appear and contain a **.bin** file. See Figure . Skip to section 5.4 SensorTile Hardware Platform if this file appears.
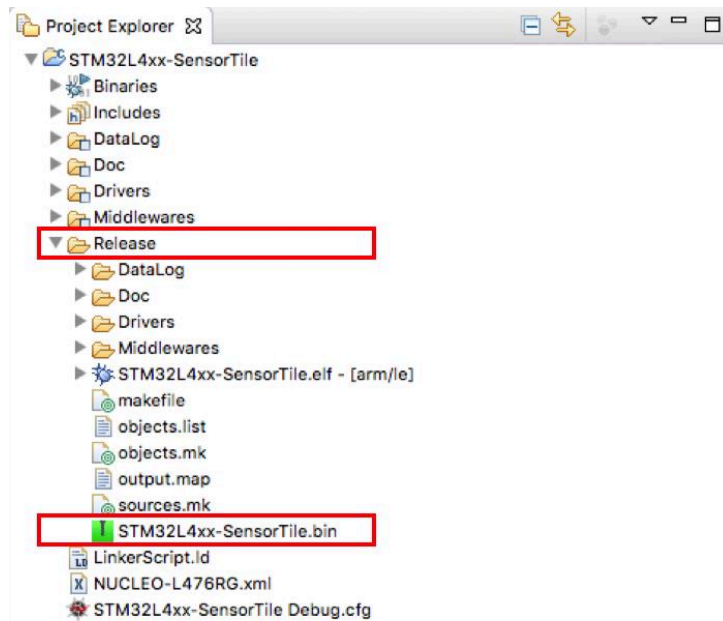
Figure 31: Project directory contents after successful compilation of project source code.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

3. If this file did not appear, ensure that none of the paths have a space in their names. If any path has a space, it will cause the make command to fail.

If any path has a space in it, repeat all installation steps ensuring that the path no longer has a space in it. Then, repeat step 1. If the binary file appears, skip to section 5.4 SensorTile Hardware Platform.

4. Attempt to build the project using Eclipse's internal builder. First, right-click the project, then click "Properties". Navigate to the subheading "C/C++ Build", and change the value of the field "Builder type" to be "Internal builder". See Figure for more details.
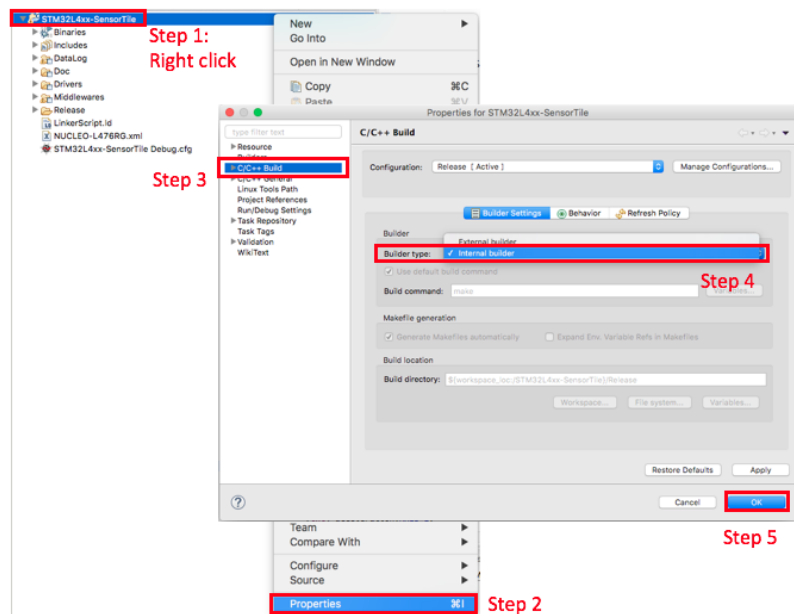
Figure 32: Using the Internal Builder to compile the project.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 23 of 39

102

5.4. SensorTile Hardware Platform

This section describes how to configure the hardware. Be very careful in this section, if the wire connections are not configured correctly, the boards could be permanently damaged. Do not proceed with the tutorial until an instructor has verified that your board is correctly configured.

1. Remove the Nucleo - L476RG board from its packaging.

2. Remove the CN2 Jumpers. (Remove both of them). See Figure 5.
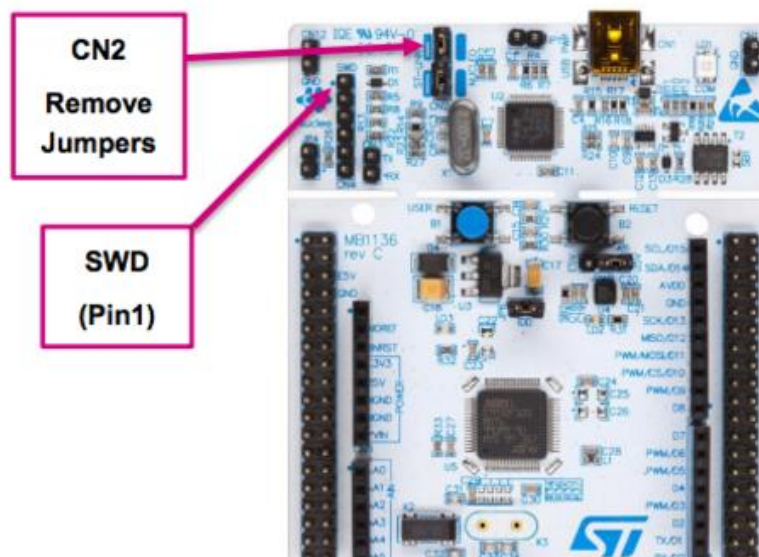
Figure 5: Removing the CN2 Jumpers from the Nucleo-L47RG board.

3. Place the SensorTile on the larger evaluation board. Ensure the orientation of the SensorTile matches Figure 6.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 24 of 39

Figure 6: Ensure the orientation of the SensorTile on the larger evaluation board matches this figure. There should be a green protrusion with a little metalic hole right next to the ST logo.

4. Connect the boards by attaching an SWD connector from the Nucleo board to the SensorTile board. Be **very** careful in this step. Please examine the figures Figure 7 - Figure 8. Do not proceed with this tutorial until you have verified with an instructor that the hardware is correctly configured.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 25 of 39

104
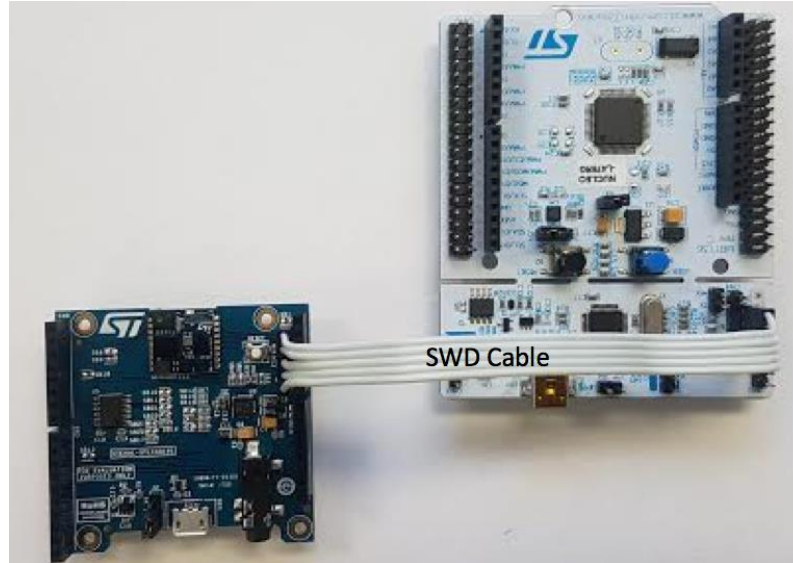
Figure 7: First figure depicting correct hardware configuration. Ensure that the SWD cable is oriented such that the pins marked "SWD" are connected via the same wire.
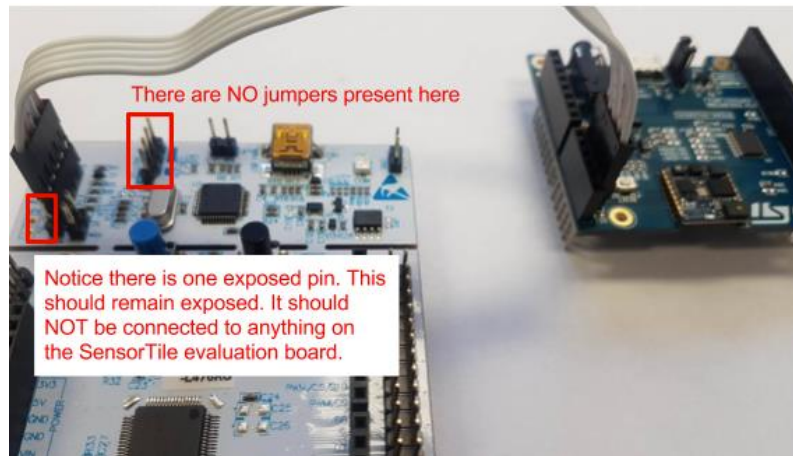
Figure 8: Second figure depicting correct hardware configuration.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 26 of 39

105

www.manaraa.com

5.5. Debug

This section will guide users through the process of running the **DataLog** application in debug mode on the SensorTile board.

1. Attach a mini-USB cable from the **Nucleo** to your PC. Attach a micro-USB cable from the **SensorTile** to your PC. See *Figure* .
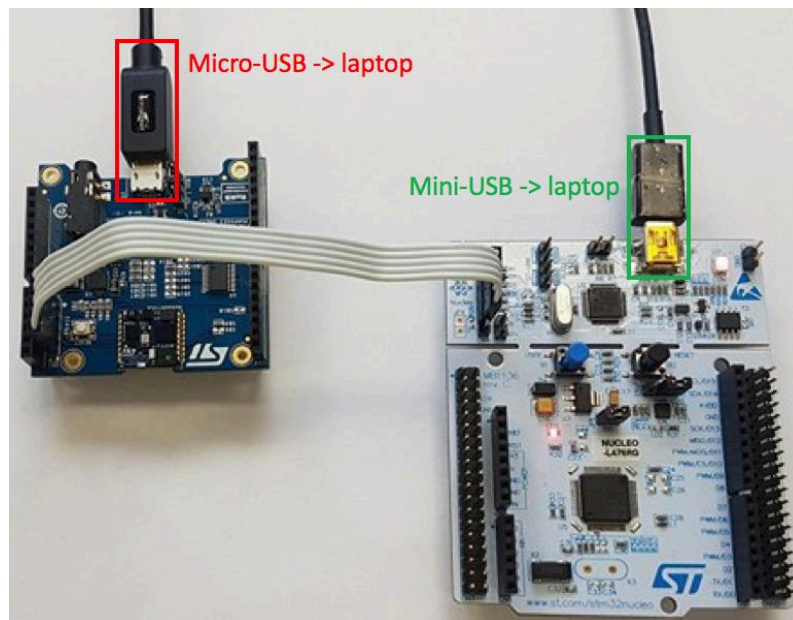
Figure 37: Establish a USB-wireline connection between each board and your PC.

2. Select the project folder, "STM32L4xx-SensorTile", in Project Explorer. Then, click "Run -> Debug As -> AC6 STM32 C/C++ Application". See *Figure* in the next page.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
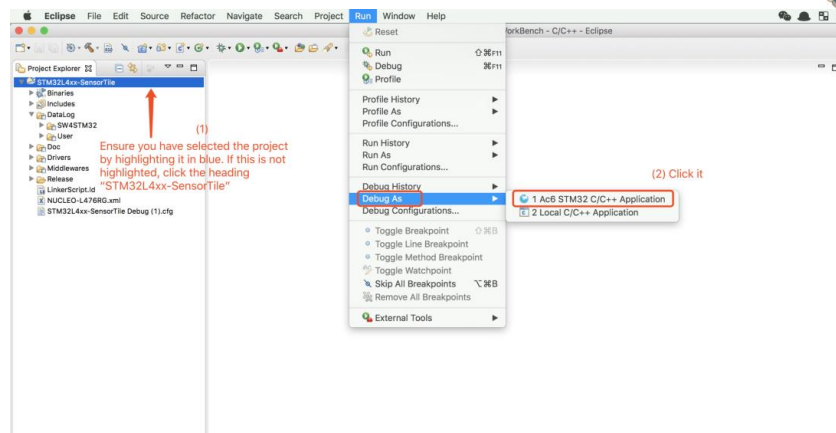
Figure 38: Launching the DataLog program in debug mode.

3. A dialog box may appear. Click "Yes" on this dialog box.
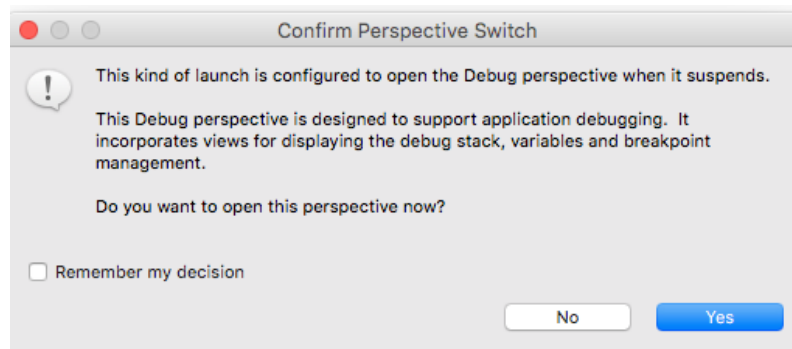
Figure 39: Click "Yes" on this dialog box.

4. A debugging interface should appear *Figure*.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
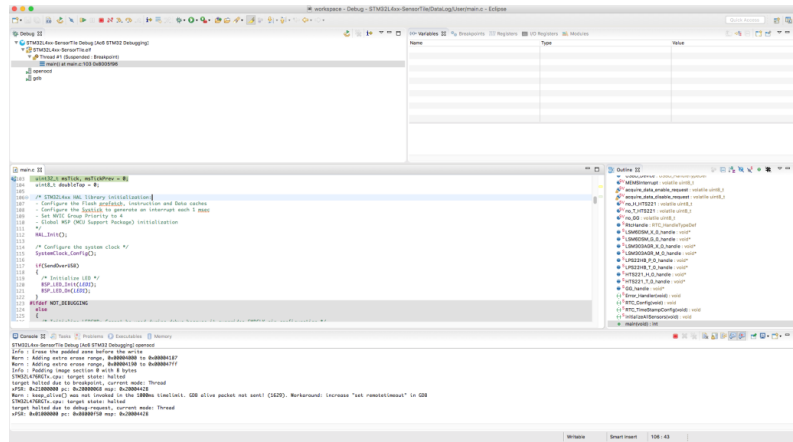
Figure 40: Debugging interface.

If you wish to switch back to the C/C++ view, click the top right corner button labeled "C/C++". See *Figure* .
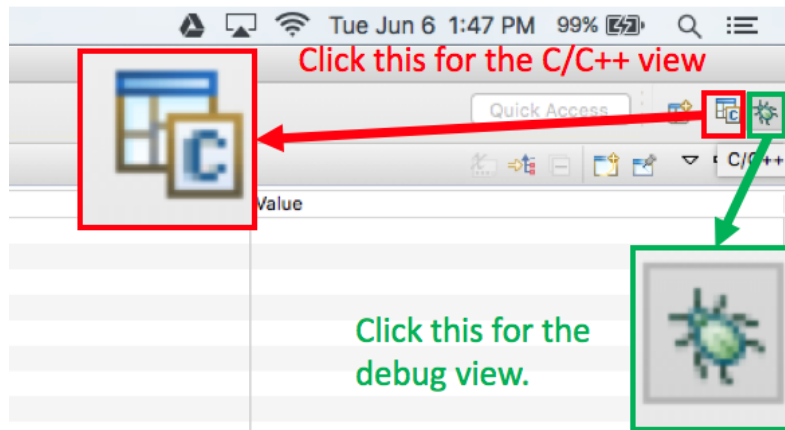
Figure 41: To enter the "C/C++" view, click the button highlighted in the red box. To enter the "debug mode" view, click the button highlighted in the green box.

5. Examine your SensorTile device. Notice how none of the LED's are activated.

6. Press the "green arrow with the yellow bar on its left" button that indicates "Resume" when you mouse over it. See *Figure* .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

108
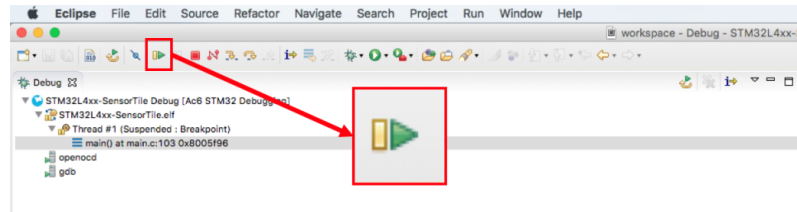
Figure 42: Starting the application in debug mode.

7. Examine your SensorTile device. Notice how the orange LED is rapidly blinking.

8. Find the COM port assigned to the SensorTile board by opening the device manager. For more information on how to open the device manager, please follow the weblink below.

   https://support.microsoft.com/en-us/instantanswers/005a1acb-776e-4320-b9f2-3a2302a320da/open-device-manager
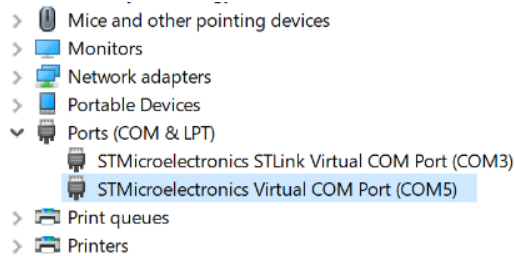
Figure 43: Inspecting the devices connected via USB.

Notice how there are two devices listed.

The device listed as "STLink Virtual COM port" is the Nucleo board.

The device highlighted in blue and listed as "STMicroelectronics Virtual COM Port" is the SensorTile.

Note: The device will not be detected by the Device Manager until the program is running. So ensure that you have performed step 6 if you do not see your device.

9. Inspect the data the SensorTile is sending over serial wireline USB connection opening PuTTy and entering the information as shown in *Figure* . Make sure this command is edited to suit the COM port of the device as it appears in your device manager. The output should resemble *Figure* .

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms
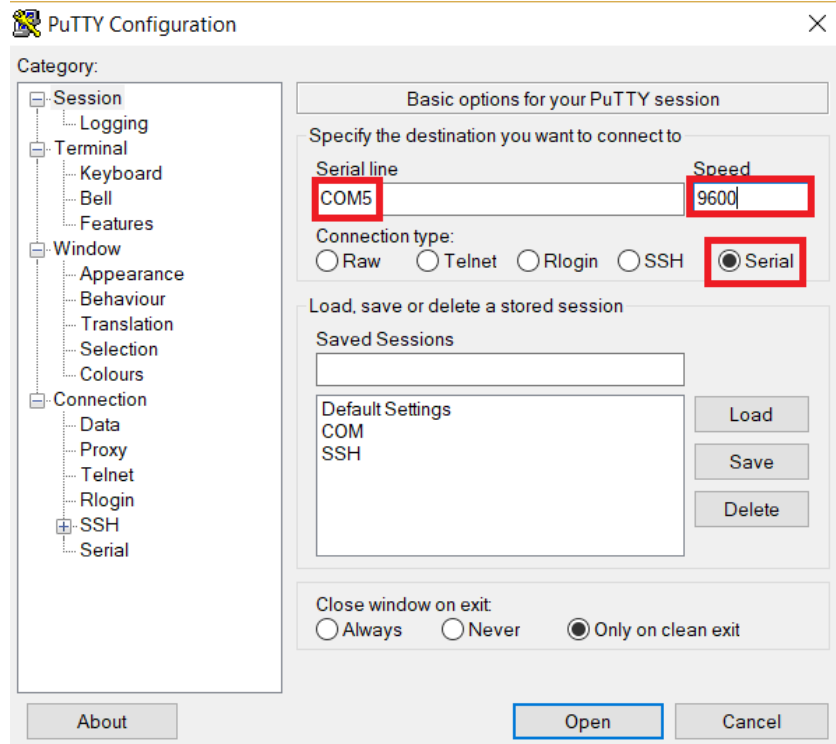
Figure 44: Examining the output from the SensorTile. Ensure that COM5 is replaced with the relevant COM port found from Figure . Ensure that the Speed is set to 9600. If you do not see these two fields in PuTTy, ensure that the Connection Type has been set to Serial.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 31 of 39

110

www.manaraa.com

Figure 45: Data being captured by the SensorTile.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 32 of 39

111

5.6. Flash

This section will guide users through the process of uploading a compiled binary file onto the SensorTile for execution. Once the program is uploaded to the board, it will run every time the SensorTile is supplied with power. The SensorTile will no longer need to be connected to the Nucleo board, nor will users have to interface with System WorkBench (the IDE).

1. Terminate and remove all existing applications on the SensorTile board as shown in *Figure* .

   Note: Ensure to remove ALL existing applications. Figure only contains one application. If there are more, delete all of them.
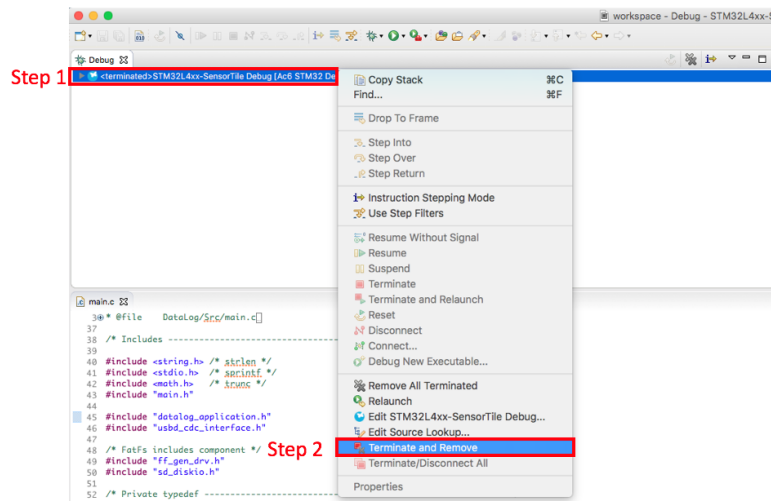
Figure 46: Terminating and removing existing applications on the SensorTile board.

2. Open the ST-LINK utility you downloaded at the start of this tutorial. There should be an icon on your desktop for this.

Figure 47: ST-LINK Utility Desktop Shortcut Icon.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

3. A new window should appear.
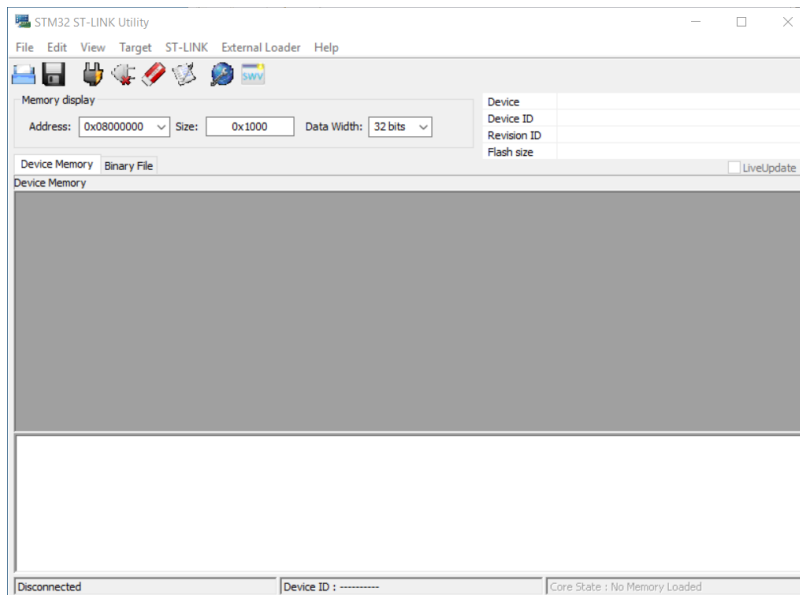
Figure 48: The ST-LINK Utility Window.

4. Click File > Open Files. A file browsing window should appear.

5. Navigate to the folder where you extracted the Example project source code.

6. Navigate to the following directory.

   ...\v1.2.0\Projects\SensorTile\Applications\DataLog\SW4STM32\STM32L4xx-SensorTile\Release

7. Click the file labelled **STM32l4xx-SensorTile.bin**. Click the "open" button.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 34 of 39

113

www.manaraa.com
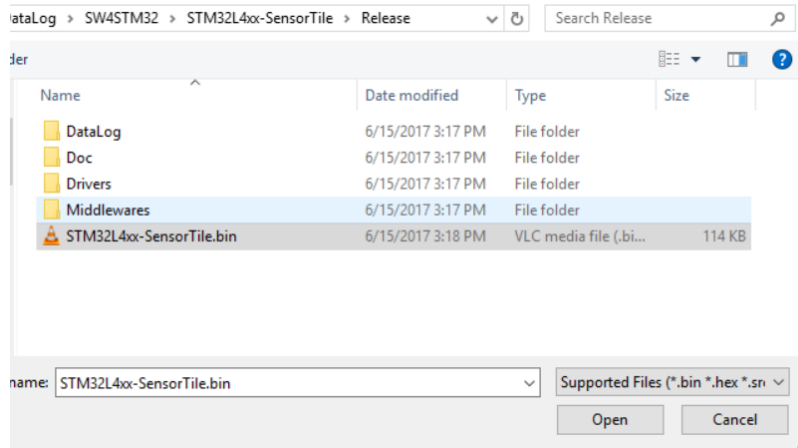
Figure 49: Opening the compiled binary file.
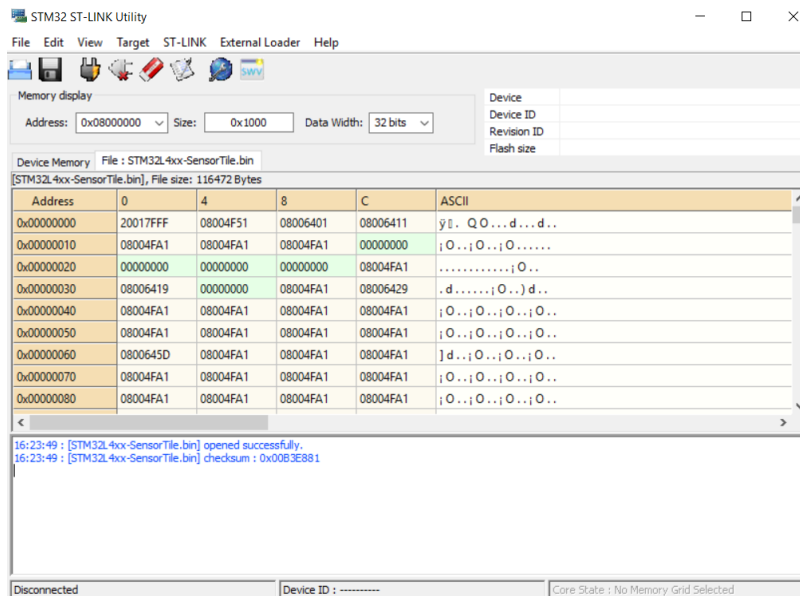
Figure 50: How the ST-LINK window should appear after clicking open.

8. Change the value of the "Address" field to be 0x08004000.

9. Change the value of the "Size" field to be 0x1000.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

10. Your window should match
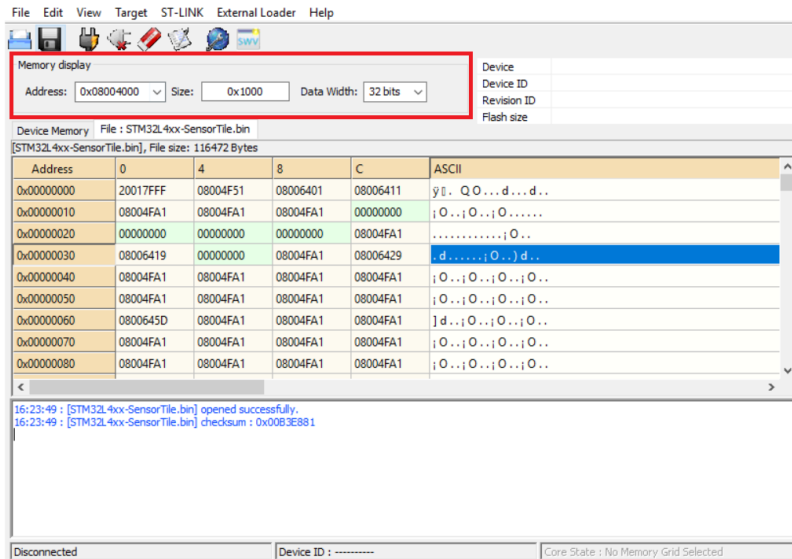
Figure 51: Updated value of Address and Size.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

Page 36 of 39

115

11. Click Target > Connect.
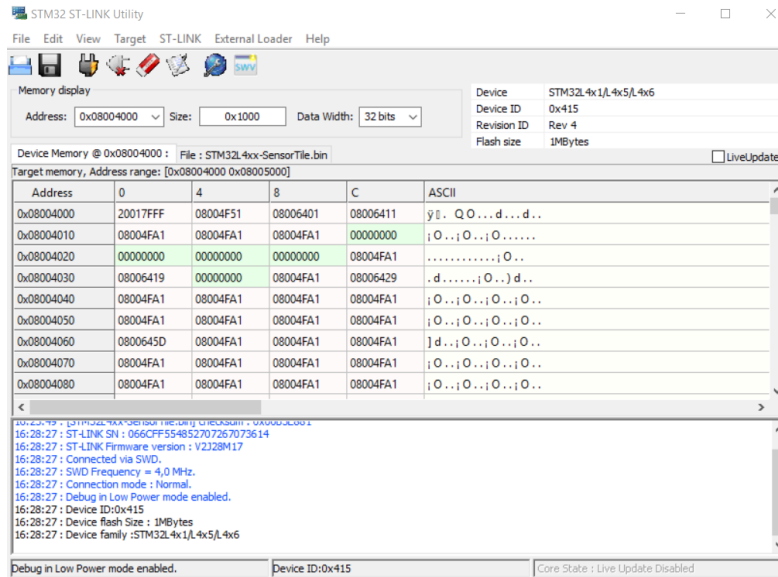
Figure 52: Window after clicking "Target > Connect".

12. Click Target > Program. Modify "Start Address" to be 0x08004000.
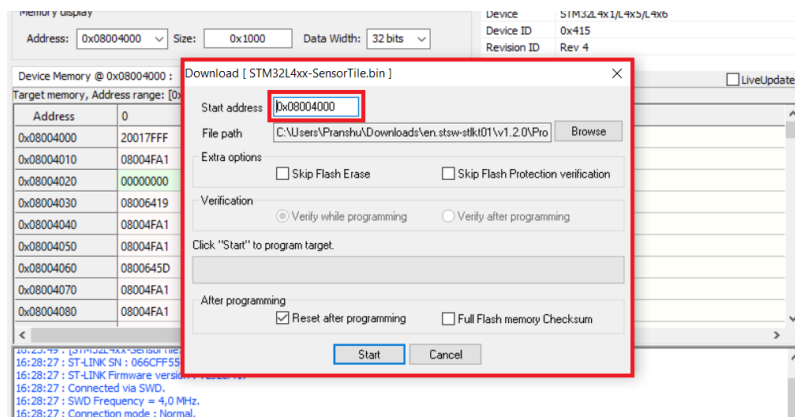
Figure 53: Target > Program window after modifying "Start Address".

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

116

www.manaraa.com

13. Click Start.

14. Click File > Close File.

15. Click File > Open File.

16. Navigate to the folder where you extracted the Example project source code.

17. Navigate to the following directory.

...\v1.2.0\Utilities\BootLoader\STM32L476RG
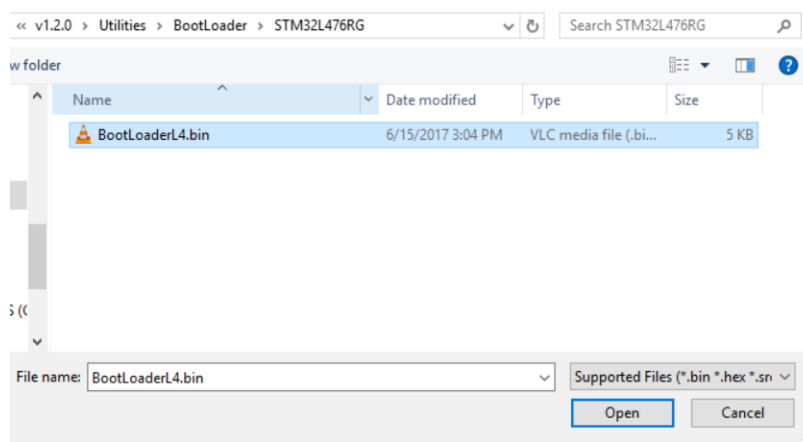
Figure 54: Opening the BootLoader file.

18. Click the file labelled "BootLoaderL4.bin" (also highlighted in *Figure*) and click Open.

19. Change the Address field to be 0x08000000.

20. Click Target > Program.

21. Change the start address to be 0x08000000.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

117
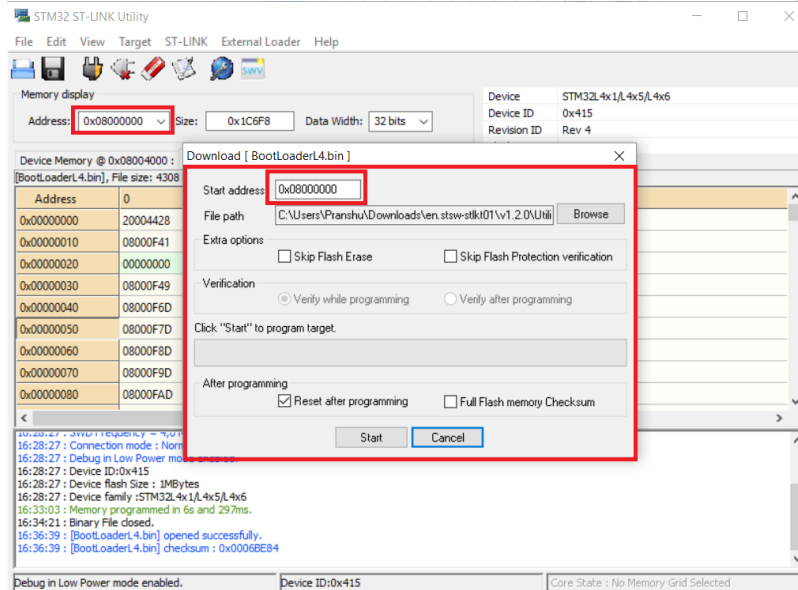
Figure 55: Updating the BootLoader.

22. Click start.

23. Disconnect the USB cables connecting the SensorTile and the Nucleo boards to your PC. This will power down the devices, enabling us to safely disconnect the SWD connection between the SensorTile board and the Nucleo board.

24. Disconnect the SWD cable connecting the SensorTile board to the Nucleo board.

25. Reconnect SensorTile board to your Windows. Do not reconnect the Nucleo board to your PC.

26. Notice that the SensorTile device immediately starts streaming data over serial USB connection to your PC by examining the blinking LED.

27. Examine the data transmitted from the SensorTile by Repeat step 9 from 5.5 Debug.

28. You will now observe complete system operation with a stand-alone, automatically operating SensorTile IoT system.

STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example for Windows Platforms

APPENDIX C

Tutorial 2

Sensor System Signal Acquisition, Event Detection

and Configuration

STMicroelectronics SensorTile Tutorial:
Sensor System Signal Acquisition, Event Detection and Configuration

Table of Contents

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

121

1. Introduction to This Tutorial

The Tutorial steps provide:

1. An introduction to control of sensor signal acquisition and sensor system configuration. These topics are fundamental to IoT system development.
2. An introduction to sensor signal detection and notifications.
3. Experience in software system development for the SensorTile IoT system demonstrating important capabilities of the System WorkBench Integrated Development Environment in accelerating system development.

For more information regarding the SensorTile board, please open the following link. www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. Introduction to STMicroelectronics Development Environment and DataLog Project Example.

Your instructor will also ensure that you have received an introduction to C code programming methods.

2. Getting Started

In the last tutorial document, we explored how to install and use the IDE. We will now examine more features such as "Open Declaration" to find where functions or variables are defined and how they are implemented. Users will first be guided through the process of opening the C-code source file labelled "main.c" in order to become more familiar with these features.

1. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.**

   Select the same workspace as in Tutorial 1.

2. Once the IDE is open, open **main.c**. If **main.c** is not the first file the IDE automatically opens, double click the file labelled "main.c" found under the following directory. Examine Figure 1 for more details.

   "**STM32L4xx-SensorTile > DataLog > User**"
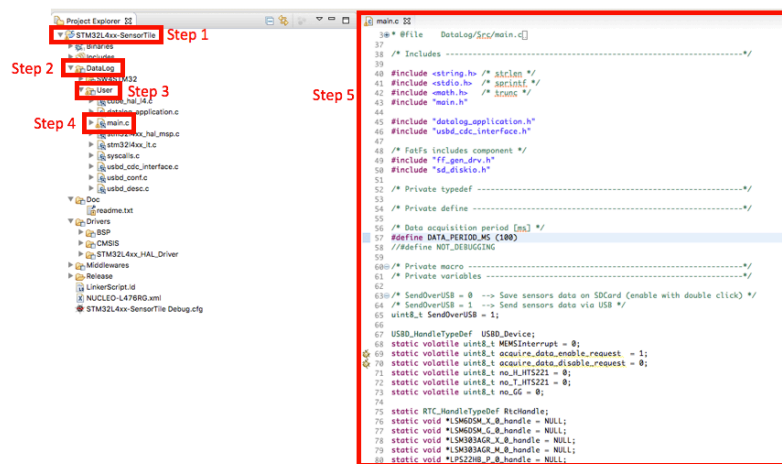
Figure 1: Opening 'main.c'. Make sure that the file looks similar to the file labelled "Step 5", and that the title is 'main.c' as shown in this figure.

3. Find the line in the file that says "**int main(void)**".

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

Page 4 of 21

123

3. Modifying Sampling Rate: Output Data Rate

3.1. Examining the Sensor Sampling Rate and Output Data Rate

1. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**. Disconnect and reconnect the SensorTile from your personal computer.

2. Build the project without making any modifications. Run this on the SensorTile in debug mode as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**.

3. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**.

4. Examine the messages regarding "TimeStamp". These are highlighted in Figure 2.

Figure 2: Examining differences in TimeStamp.

5. Calculate the difference between 5 successive timestamps (i.e. t1 = T2-T1, t2 = T3-T2, ..., t5 = T5-T4). Average these differences (t1 + t2 + t3 + t4 + t5)/5. What is this average difference? See Figure 2.

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

We shall refer to this average difference in timestamps as the "USB data period".

This is not the rate at which data is recorded by the sensor. This is the rate at which data is transmitted to the host computer over the serial USB data transport. The sensor sampling rate will be adjusted in future tutorials.

6. First, we must find the location where the data rate is being referenced. Once we find this location, we can inspect where the variable to set the output data rate is being defined.

   Once we have found where the variable is defined, we can edit the value.

   Once the value is edited, the code can be recompiled and uploaded to the board to affect system behavior.

7. Find the line where the output data rate is being used in **main.c**. The line is highlighted in Figure 3.


```
160    /* Initialize and Enable the available sensors */
161    initializeAllSensors();
162    enableAllSensors();
163
164    while (1)
165    {
166        /* Get sysTick value and check if it's time to execute the task */
167        msTick = HAL_GetTick();
168        if(msTick % DATA_PERIOD_MS == 0 && msTickPrev != msTick)
169        {
170            msTickPrev = msTick;
171            if(SendOverUSB)
172            {
173                BSP_LED_On(LED1);
174            }
175 #ifdef NOT_DEBUGGING
176            else if (SD_Log_Enabled)
177            {
178                BSP_LED_On(LEDSWD);
179            }
180 #endif
181            RTC_Handler( &RtcHandle );
```

These lines will help you find the line of interest to us

Look for this line

Figure 3: Finding where output data rate is being used.

Explanation:

`msTick = HAL_GetTick(); // reads the time counter on the SensorTile board.`

HAL_GetTick() is useful because it returns how many milliseconds have elapsed since the SensorTile started recording data.

```
msTick % DATA_PERIOD_MS
```

This C code mathematical operation is the modulo divide. Here, **msTick** is increasing at each millisecond. When **msTick** is evaluated in this line of execution and when **msTick** acquires a value that is an exact multiple of **DATA_PERIOD_MS**, the modulo operation will return 0. For more information regarding modulo arithmetic, please open the following weblink.

https://en.wikipedia.org/wiki/Modulo_operation

```
msTickPrev != msTick
```

We only want the SensorTile to perform calculations once every time the **DATA_PERIOD_MS** has been reached. However, the processor may reach the **HAL_GetTick()** function several times before 1ms has elapsed. As such, we must track the previous values of **msTick**. If the value has not changed, no action should be taken.

8. Let's examine the time period over which the SensorTile should be outputting data.

   First, click the variable **DATA_PERIOD_MS**.

   Once the variable is highlighted, right click it.

   Once the drop-down menu appears, click "Open Declaration" as shown in Figure 4.

Figure 4: Opening a variable declaration.

Note: this can be done with *any* variable or *any* function. This is a crucial feature of the System WorkBench IDE, as it enables developers to quickly find and inspect functions and variables. These functions and variables could be otherwise impossible to find as large projects contain multiple files and folders where the functions and variables are

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

being implemented or initialized.

9. Your screen should now advance to another line of code in the same file shown in Figure 5.


```
56  /* Data acquisition per'
57  #define DATA_PERIOD_MS
58  //#define NOT_DEBUGGING
59
60⊖ /* Private macro ------
61   /* Private variables --
```

Figure 5: Jumping to the relevant line in the main.c file.

10. Read the definition of DATA_PERIOD_MS. Does it match the value calculated in step 4 of section 3.1 Examining the Sensor Sampling Rate and Output Data Rate? If there is a difference, can you explain the difference?

3.2. Modifying the USB Data Output Rate

1. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**.
   Disconnect and reconnect the SensorTile from your personal computer.

2. Change the variable of **DATA_PERIOD_MS** to be 1000. See Figure 6.

```
55
56  /* Data acquisition period [ms] */
57  #define DATA_PERIOD_MS (1000)
58  //#define NOT_DEBUGGING
59
```

Figure 6: Updating the DATA_PERIOD_MS variable.

3. Save the changes you made to **main.c** and build the project.

4. Run the project on the SensorTile in debug mode.

5. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**.

6. Examine the messages regarding "TimeStamp". These are highlighted in Figure 2.

127

www.manaraa.com

7. Calculate the difference between 5 successive timestamps (i.e. t1 = T2-T1, t2 = T3-T2, …, t5 = T5-T4). Average these differences (t1 + t2 + t3 + t4 + t5)/5. What is this average difference? See Figure 2. Does this agree with your expectations?

4. Creating New Data, New Messages, and Event Detection

4.1. Computing Vector Magnitude Acceleration

This section will guide users through the process of adding information regarding computing vector magnitude acceleration from signals obtained from the accelerometer.

The SensorTile accelerometer measures acceleration on each of three orthogonal axes, X, Y, and Z with acceleration values of a_x, a_y, and a_z, respectively.

Vector magnitude acceleration, a_{mag}, is defined as,

$$a_{mag} = \sqrt{a_x{}^2 + a_y{}^2 + a_z{}^2}$$

In this section, the steps required to compute and the transmit vector magnitude acceleration are demonstrated.

1. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project*. Disconnect and reconnect the SensorTile from your personal computer.

2. Open main.c, change the variable of **DATA_PERIOD_MS** to be 100. See Figure 6. (Note: change the value to 100 instead of 1000).

3. Open the declaration for the function **Accelero_Sensor_Handler** the same way you found the declaration for **DATA_PERIOD_MS** in step 8 from section 3.1 Examining the Sensor Sampling Rate and Output Data Rate.

    The function is called in main.c. It can be found in the **while (1)** loop we were inspecting earlier. See Figure 7 for more details.
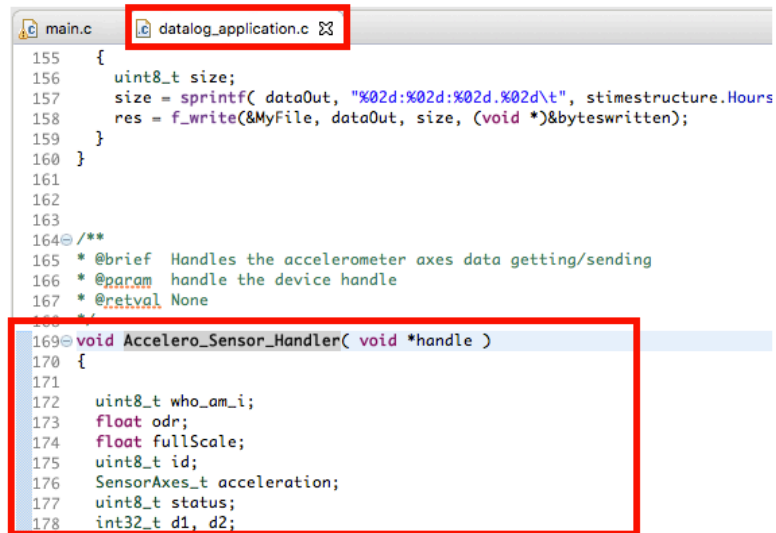
129

```
while (1)
{
    /* Get sysTick value and check if it's time to execute the task */
    msTick = HAL_GetTick();
    if(msTick % DATA_PERIOD_MS == 0 && msTickPrev != msTick)
    {
        msTickPrev = msTick;
        if(SendOverUSB)
        {
            BSP_LED_On(LED1);
        }
#ifdef NOT_DEBUGGING
        else if (SD_Log_Enabled)
        {
            BSP_LED_On(LEDSWD);
        }
#endif
        RTC_Handler( &RtcHandle );

        Accelero_Sensor_Handler( LSM6DSM_X_0_handle );
```

Figure 7: Inspecting the Accelero_Sensor_Handle() function.

4. A new file should appear on your tab, and should be open to the function declaration as seen in Figure 8.


```
    main.c        datalog_application.c
155     {
156         uint8_t size;
157         size = sprintf( dataOut, "%02d:%02d:%02d.%02d\t", stimestructure.Hours
158         res = f_write(&MyFile, dataOut, size, (void *)&byteswritten);
159     }
160  }
161
162
163
164  /**
165   * @brief  Handles the accelerometer axes data getting/sending
166   * @param  handle the device handle
167   * @retval None
        */
169  void Accelero_Sensor_Handler( void *handle )
170  {
171
172      uint8_t who_am_i;
173      float odr;
174      float fullScale;
175      uint8_t id;
176      SensorAxes_t acceleration;
177      uint8_t status;
178      int32_t d1, d2;
```

Figure 8: Definition of the Accelero_Sensor_Handle() function.

5. Scroll down until you find the **sprintf** function as highlighted in Figure 9.

```
BSP_ACCELERO_Get_Instance( handle, &id );

BSP_ACCELERO_IsInitialized( handle, &status );

old_verbose = verbose;
if ( status == 1 )
{
  if ( BSP_ACCELERO_Get_Axes( handle, &acceleration ) == COMPONENT_ERROR )
  {
    acceleration.AXIS_X = 0;
    acceleration.AXIS_Y = 0;
    acceleration.AXIS_Z = 0;
  }

  if(SendOverUSB) /* Write data on the USB */
  {
    sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d", (int)acceleration.AXIS_X, (int)acceleration.AXIS_Y, (int)acceleration.
    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
```

Figure 9: Calling sprintf() and CDC_Fill_Buffer() to send data over USB serial.

For more information regarding **sprintf** please open the following weblink on a browser on your personal computer.

https://linux.die.net/man/3/sprintf

6. We are going to add some information to the end of the Accelerometer output to inform the user of the magnitude of the acceleration vector. To do this we must modify the code such that it matches the screenshot in Figure 10.

```
if ( status == 1 )
{
  if ( BSP_ACCELERO_Get_Axes( handle, &acceleration ) == COMPONENT_ERROR )
  {
    acceleration.AXIS_X = 0;
    acceleration.AXIS_Y = 0;
    acceleration.AXIS_Z = 0;
  }

  if(SendOverUSB) /* Write data on the USB */
  {
    uint32_t abs_acc;
    abs_acc = ((int)acceleration.AXIS_X * (int)acceleration.AXIS_X);
    abs_acc += ((int)acceleration.AXIS_Y * (int)acceleration.AXIS_Y);
    abs_acc += ((int)acceleration.AXIS_Z * (int)acceleration.AXIS_Z);
    abs_acc = sqrt((float) abs_acc);

    sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d, |ACC|: %d",
            (int)acceleration.AXIS_X,
            (int)acceleration.AXIS_Y,
            (int)acceleration.AXIS_Z,
            (int) abs_acc
            );
    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
```

Figure 10: Computing and displaying the magnitude of the acceleration vector.

7. Save the changes you made and build the project.

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

5. Scroll down until you find the **sprintf** function as highlighted in Figure 9.

Figure 9: Calling sprintf() and CDC_Fill_Buffer() to send data over USB serial.

For more information regarding **sprintf** please open the following weblink on a browser on your personal computer.

https://linux.die.net/man/3/sprintf

6. We are going to add some information to the end of the Accelerometer output to inform the user of the magnitude of the acceleration vector. To do this we must modify the code such that it matches the screenshot in Figure 10.

Figure 10: Computing and displaying the magnitude of the acceleration vector.

7. Save the changes you made and build the project.

8. Run the project on the SensorTile in debug mode.

9. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*.

10. Take a screenshot of the messages you receive over the USB Serial port.

133

4.2. Detecting Events: Acceleration Vector Magnitude Threshold Crossing

This section will demonstrate how sensor signals may be monitored and events may be detected. In addition, this introduces a method for adding a message to indicate that the acceleration vector magnitude has exceeded a certain threshold value.

1. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*. Disconnect and reconnect the SensorTile from your personal computer.

2. Open the declaration for the function **Accelero_Sensor_Handler** the same way you found the declaration for **DATA_PERIOD_MS** in step 3 from section 4.1 Computing Vector Magnitude Acceleration.

3. Modify the function such that it matches Figure 11.

```
if(SendOverUSB) /* Write data on the USB */
{
    uint32_t abs_acc;
    abs_acc = ((int)acceleration.AXIS_X * (int)acceleration.AXIS_X);
    abs_acc += ((int)acceleration.AXIS_Y * (int)acceleration.AXIS_Y);
    abs_acc += ((int)acceleration.AXIS_Z * (int)acceleration.AXIS_Z);
    abs_acc = sqrt((float) abs_acc);

    sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d, |ACC|: %d",
            (int)acceleration.AXIS_X,
            (int)acceleration.AXIS_Y,
            (int)acceleration.AXIS_Z,
            (int) abs_acc
            );
    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));

    if (abs_acc > 1550)
    {
        sprintf( dataOut, "\n\r\t\t\tAcceleration Vector Magnitude > 1.5g!");
        CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
    }
}
```

Figure 11: Adding acceleration threshold crossing detection.

4. Save the changes you made and build the project.

5. Run the project on the SensorTile in debug mode.

6. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*.

134

7. Shake the sensor gently enough to make sure you don't disconnect any of the cables, but strongly enough to ensure the new message appears.

   Take a screenshot this message appearing over the USB Serial port.

5. Modifying Data Acquisition Parameters

This section demonstrates methods for modification of sensor data acquisition parameters. In particular, this focuses on methods for adjusting sensor measurement range. The SensorTile microaccelerometer Full-Scale measurement range (FS) is adjusted.

5.1. Adjust Scaling Factor (Full-Scale FS)

1. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*. Disconnect and reconnect the SensorTile from your personal computer.

2. Open the declaration for the function **Accelero_Sensor_Handler** the same way you found the declaration for **DATA_PERIOD_MS** in step 3 from section 4.1 Computing Vector Magnitude Acceleration.

3. Modify the function such that it matches Figure 12.

```
if(SendOverUSB) /* Write data on the USB */
{
    uint32_t abs_acc;
    abs_acc = ((int)acceleration.AXIS_X * (int)acceleration.AXIS_X);
    abs_acc += ((int)acceleration.AXIS_Y * (int)acceleration.AXIS_Y);
    abs_acc += ((int)acceleration.AXIS_Z * (int)acceleration.AXIS_Z);
    abs_acc = sqrt((float) abs_acc);

    sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d, |ACC|: %d",
            (int)acceleration.AXIS_X,
            (int)acceleration.AXIS_Y,
            (int)acceleration.AXIS_Z,
            (int) abs_acc
            );
    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));

    if (abs_acc > 3550)    Old value: 1550. New value: 3550.
    {
        sprintf( dataOut, "\n\r\t\t\tAcceleration Vector Magnitude > 3.5g!");
        CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
    }
```

Figure 12: Modifying acceleration threshold crossing detection.

4. Save the changes you made and build the project.

5. Run the project on the SensorTile in debug mode.

6. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project*

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

136

www.manaraa.com

Example.

7. To demonstrate operation, shake the sensor (taking care not to disconnect cables).

   Notice how the message no longer appears as a result of this change in Full-Scale range.

   This is to be expected. Notice that the absolute value of the acceleration for each of the three axes (ACC_X, ACC_Y, ACC_Z) never exceed 2000.

   If we calculate the maximum vector magnitude, we should expect the following.

   $$|A_{max}| = \left\| \begin{pmatrix} 2000 \\ 2000 \\ 2000 \end{pmatrix} \right\| \cong 3464$$

   As such, if we wish to see the magnitude exceed a value of 3.5g, we would need to adjust the parameters of the sensor such that it can record values greater than 2000 for each axis.

8. Terminate and remove all previous applications from the SensorTile as instructed in the document labelled ***STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project***. Disconnect and reconnect the SensorTile from your personal computer.

9. Open **main.c** and find the **while (1)** loop. Modify the file directly before the **while (1)** loop to match Figure 13. We make the modification **outside** of the while loop because we do not want to be constantly adjusting the Full-Scale range of the values the sensor returns. We only want to change it once.

```
/* Initialize and Enable the available sensors */
initializeAllSensors();
enableAllSensors();

BSP_ACCELERO_Set_FS_Value( LSM6DSM_X_0_handle , 4.0f);

while (1)
{
    /* Get sysTick value and check if it's time to execute the task */
    msTick = HAL_GetTick();
    if(msTick % DATA_PERIOD_MS == 0 && msTickPrev != msTick)
    {
        msTickPrev = msTick;
        if(SendOverUSB)
        {
            BSP_LED_On(LED1);
        }
#ifdef NOT_DEBUGGING
        else if (SD_Log_Enabled)
        {
            BSP_LED_On(LEDSWD);
        }
#endif
        RTC_Handler( &RtcHandle );

        Accelero_Sensor_Handler( LSM6DSM_X_0_handle );
```

Figure 13: Adjusting the full scale (FS) range of each axis of data read from the accelerometer.

STMicroelectronics SensorTile Tutorial: Sensor System Control and Configuration

10. Save the changes you made and build the project.

11. Run the project on the SensorTile in debug mode.

12. Examine the messages received by your personal computer over the serial connection as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*.

13. Shake the sensor gently enough to make sure you don't disconnect any of the cables, but strongly enough to ensure the new message appears.

    Take a screenshot this message appearing over the USB Serial port.

138

5.2. Making Similar Modifications for Other Parameters

The tables below summarize some of the important functions available to adjust the data acquisition parameters for the Accelerometer and Gyroscope sensors.

Accelerometer (acceleration):

Example Function Call	Function Use	List of Valid Second Arguments	Unit
`BSP_ACCELERO_Set_FS_Value( LSM6DSM_X_0_handle, 4.0f)`	Set the range of the axes	2.0f, 4.0f, 8.0f, 16.0f	g (x 9.81 ms^{-2})
`BSP_ACCELERO_Set_ODR_Value( LSM6DSM_X_0_handle, 13.0f)`	Sets output data rate of sensor. This is different from the rate at which data is reported over serial USB.	13.0f, 26.0f, 52.0f, 104.0f, 208.0f, 416.0f, 833.0f, 1660.0f, 3330.0f 6660.0f	Hz

Gyroscope (angular velocity):

Example Function Call	Function Use	List of Valid Second Arguments	Unit
`BSP_GYRO_Set_FS_Value( LSM6DSM_G_0_handle, 245.0f)`	Set the range of the axes	245.0f, 500.0f, 1000.0f 2000.0f	Degrees per second
`BSP_GYRO_Set_ODR_Value( LSM6DSM_G_0_handle, 4.0f)`	Sets output data rate of sensor. This is different from the rate at which data is reported over serial USB.	13.0f, 26.0f, 52.0f, 104.0f, 208.0f, 416.0f, 833.0f, 1660.0f, 3330.0f 6660.0f	Hz

139

6. Introduction to the Gyroscope

Complete the following tasks in order to develop a deeper understanding of the Gyroscope systems available on the SensorTile board.

1. Open the declaration for the function **Gyro_Sensor_Handler**. The function is called in main.c. It can be found in the **while (1)** loop we were inspecting earlier. It can be found directly after the call to **Accelero_Sensor_Handler** as seen in Figure 7.

2. Modify this function such that it now computes the vector magnitude computation and transmits this computed value over the serial USB connection. This should be similar to the procedure followed to generate Figure 10. The key data structure we need to inspect here is **angular_velocity** instead of **acceleration**.

   Terminate and remove all previous programs from the SensorTile board.

   Save the changes you made and build the project.

   Run the project on the SensorTile in debug mode.

   Examine the messages received by your personal computer over the serial connection.

   Take a screenshot these messages appearing over the USB Serial port.

   Take a screenshot of the largest gyroscope angular velocity on a single axis (i.e. not the vector magnitude) when shaking the sensor.

3. Modify the system to send a message over the serial USB connection when the vector magnitude of the angular velocity exceeds a threshold value of **40,000** (this corresponds to an angular velocity of 40 degrees per second). This should be similar to the procedure followed to generate Figure 11.

   Terminate and remove all previous programs from the SensorTile board.

   Save the changes you made and build the project.

   Run the project on the SensorTile in debug mode.

   Examine the messages received by your personal computer over the serial connection.

   Shake the sensor gently enough to make sure you don't disconnect any of the cables, but strongly enough to ensure the new message appears.

Take a screenshot this message appearing over the USB Serial port.

4. Modify the system so that the Full-Scale range of the Gyroscope sensor is 245.0 dps. This should be similar to the procedure followed to generate Figure 13. Refer to section 5.2 Making Similar Modifications for Other Parameters for more information on what function to use to make this modification.

    Terminate and remove all previous programs from the SensorTile board.

    Save the changes you made and build the project.

    Run the project on the SensorTile in debug mode.

    Take a screenshot of the largest gyroscope angular velocity on a single axis (i.e. not the vector magnitude) when shaking the sensor.

141

APPENDIX D

Tutorial 3

Accelerometer Sensor Systems

and Orientation and Event Detection

with Finite State Machine

STMicroelectronics SensorTile Tutorial: Accelerometer Sensor Systems and Orientation and Event Detection

Table of Contents

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

144

1. Introduction to This Tutorial

The Tutorial steps provide:

1. An introduction to accelerometer sensor systems and methods for detection of sensor orientation by exploiting gravitational acceleration signals.
2. Measurement of orientation by the polar coordinate system.
3. An introduction to gesture recognition through the use of sensor information and state machine systems for characterizing specific behavior.

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. Introduction to SensorTile and the System WorkBench Integrated Development Environment (IDE)
2. Sensor System Signal Acquisition, Event Detection and Configuration.

Your instructor will ensure you have the required background regarding the Cartesian and Polar coordinate systems.

2. Cartesian to Polar Coordinate Conversion in Multiaxis Sensing

In the last tutorial document, we explored how to make various modifications to the DataLog example program. We will now examine how to make further modifications, and understand why these modifications are useful building blocks for systems that aim to perform gesture recognition.

1. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.**

   Select the same workspace as in Tutorial 1.

2. Once the IDE is open, open the declaration for the function labelled **Accelero_Sensor_Handler** as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Sensor System Signal Acquisition, Event Detection and Configuration**.

3. Modify the function such that it no longer contains the modifications we made in Tutorial 2. See Figure 1.

```
if(SendOverUSB) /* Write data on the USB */
{
    uint32_t abs_acc;
    abs_acc = ((int)acceleration.AXIS_X * (int)acceleration.AXIS_X);
    abs_acc += ((int)acceleration.AXIS_Y * (int)acceleration.AXIS_Y);
    abs_acc += ((int)acceleration.AXIS_Z * (int)acceleration.AXIS_Z);
    abs_acc = sqrt((float) abs_acc);

    sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d, |ACC|: %d",
            (int)acceleration.AXIS_X,
            (int)acceleration.AXIS_Y,
            (int)acceleration.AXIS_Z,
            (int) abs_acc
    );
    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));

    if (abs_acc > 3550)
    {
        sprintf( dataOut, "\n\r\t\t\tAcceleration Vector Magnitude > 3.5g!");
        CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
    }
```

Figure 1: Delete the lines found enclosed in the red box.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

146

4. Modify the variable declaration section of the **Accelero_Sensor_Handler** function such that it matches Figure 2.

```
void Accelero_Sensor_Handler( void *handle )
{

    uint8_t who_am_i;
    float odr;
    float fullScale;
    float r, theta, phi;
    float x, y, z;
    uint8_t id;
    SensorAxes_t acceleration;
    uint8_t status;
    int32_t d1, d2, d3, d4, d5, d6;
    const float RADIAN = 57.2957795;
```
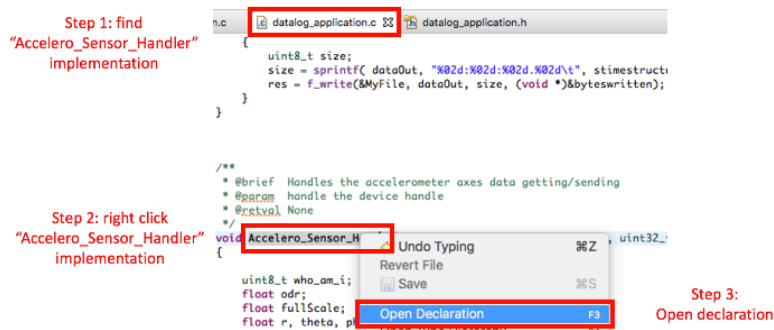
Figure 2: Ensuring the correct variables are declared.

5. Note that in the computation of angles using trigonometric function values the multiplying factor of RADIAN = 180 degree/π radians defined above is required.
6. Add the lines of code found in Figure 3 to the **Accelero_Sensor_Handler** function.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

```
if(SendOverUSB) /* Write data on the USB */
{
  // convert the in32_t data type into float data type to enable floating point computation
  // Store float variable in respective variables
  x = (float) acceleration.AXIS_X;
  y = (float) acceleration.AXIS_Y;
  z = (float) acceleration.AXIS_Z;

  // Convert Cartesian representation of acceleration components to polar coordinate components
  r = sqrt(x*x + y*y + z*z);
  theta = acos(z/r)*RADIAN;
  phi = atan2(y, x)*RADIAN;

  // Convert floating point values to integer for compatibility
  // with USB serial data transport
  floatToInt(r, &d1, &d2, 3);
  floatToInt(theta, &d3, &d4, 3);
  floatToInt(phi, &d5, &d6, 3);
  sprintf( dataOut,"\n\rr: %d.%03d, theta: %d.%03d, phi: %d.%03d",
          (int)d1, (int)d2,
          (int)d3, (int)d4,
          (int)d5, (int)d6
  );
  CDC_Fill_Buffer((uint8_t *)dataOut, strlen(dataOut));
  sprintf( dataOut, "\n\rA_x: %d, A_y: %d, A_z: %d, |A|: %d.%03d",
          acceleration.AXIS_X,
          acceleration.AXIS_Y,
          acceleration.AXIS_Z,
          (int)d1, (int)d2
  );
  CDC_Fill_Buffer((uint8_t *)dataOut, strlen(dataOut));
```

Figure 3: Adding polar coordinate conversion to SensorTile system.

7. Terminate and remove all previous applications from the SensorTile board.

8. Compile and run the DataLog application on the SensorTile board in debug mode.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

9. Examine the data transmitted over the Serial USB connection to your personal computer. Take a screenshot.

10. Orient the SensorTile, such that the value for A_z is approximately +1000, and the two other axes (A_x, A_y) are as close to 0 as possible. Record the values of r, phi and theta.

11. Orient the SensorTile upside down such that the value for A_z is approximately -1000, and the two other axes (A_x, A_y) are as close to 0 as possible. Record the values of r, phi and theta.

12. Repeat steps 9 and 10 until you can fill in the following table.

	R (magnitude)	Phi (azimuth)	Theta (inclination)
$A_z \sim +1000, A_x{\sim}0, A_y{\sim}0$			
$A_z \sim -1000, A_x{\sim}0, A_y{\sim}0$			
$A_Y \sim +1000, A_x{\sim}0, A_Z{\sim}0$			
$A_Y \sim -1000, A_x{\sim}0, A_Z{\sim}0$			
$A_X \sim +1000, A_Z{\sim}0, A_y{\sim}0$			
$A_X \sim -1000, A_Z{\sim}0, A_y{\sim}0$			

149

3. Demonstration of Gesture Recognition Systems

3.1. Introduction

This section will guide users through the process of modifying the **Accelero_Sensor_Handler** function such that it will be able to recognize a simple gesture. The simple gesture is defined as follows.

1. Place the sensor such that

   The z-axis acceleration, **A_z < -z_thresh** (specifically, the z-component of the acceleration vector is below a certain threshold value) for at least a time, **tau** milliseconds.

   In terms of physical orientation, this corresponds to the sensor being "upside down" for at least "tau" milliseconds.

2. Within the next time, **tau** milliseconds, the sensor must be reoriented such that

   The z-axis acceleration, **A_z > +z_thresh** (Specifically, the z-component of the acceleration vector is above a certain threshold value) for at least **tau** milliseconds.

   In terms of physical orientation, this corresponds to the sensor being "right side up" for at least "tau" milliseconds.

We will model this as a simple state machine, which can be seen in Figure 4 below.
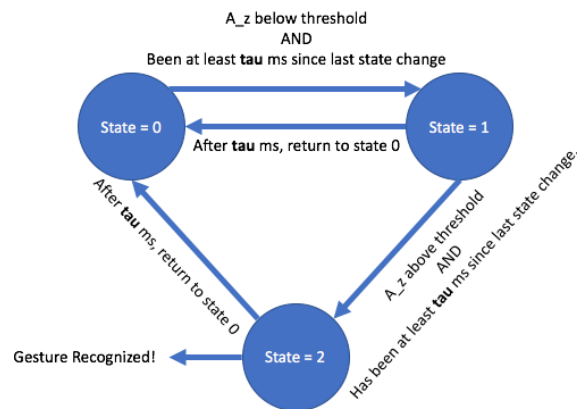
Figure 4: State machine for gesture recognition.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

3.2. Modifications to main.c

Make the following modifications to main.c. We are making these modifications so that the **Accelero_Sensor_Handler** function can have access to variables that will help track the previous state and timing information.

1. Declare and initialize the variable **msTickStateChange** as a **uint32_t** to have a value of **0**.

   This variable will be used to track the time at which the state was changed.

   See Figure 5.

2. Declare and initialize the variable **state** as a **uint8_t** to have a value of **0**.

   This variable will be used to track the current state of the system.

   See Figure 5.

```
int main( void )
{

    uint32_t msTick, msTickPrev = 0, msTickStateChange = 0;
    uint8_t doubleTap = 0, state = 0;
```

Figure 5: Adding the variables to track the time at which the system last observed a change of state and the state variable itself.

3. Pass these variables to the function **Accelero_Sensor_Handler** as shown in Figure 6.

   Notice: there is an ampersand ('&') in front of the variables **msTickStateChange** and **state**. This is because we want the function **Accelero_Sensor_Handler** to modify the values of these variables. In order to modify them, the functions must accept these parameters **by reference**.

   Your instructor will provide further clarification on passing variables by reference, and passing pointers to functions.

```
Accelero_Sensor_Handler( LSM6DSM_X_0_handle, msTick, &msTickStateChange, &state);
```
Figure 6: Modification to calling the function Accelero_Sensor_Handler.

4. Modify the declaration of **Accelero_Sensor_Handler** so that the project can compile successfully. See Figure 7 for more details.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

5. Add the variables **tau** and **z_thresh** as seen in Figure 7.

Figure 7: Modifying Accelero_Sensor_Handler implementation to enable access to state selection variables. Notice how we use the '*' character here instead of '&'. Your instructor will provide additional guidance on what these symbols mean.

6. This is the **implementation** of the function **Accelero_Sensor_Handler**. The function is currently defined in another file labelled **"datalog_application.h"**. We need to modify this declaration to match the implementation of the function such that the project can compile.

   To find this file, first right-click the function name **"Accelero_Sensor_Handler"** as seen in Figure 8. Then click "Open Declaration". This should open the file labelled **"datalog_application.h"**. Modify the relevant line to match Figure 9.

Figure 8: Open the declaration (not implementation) of Accelero_Sensor_Handler.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

Figure 9: Modify the function declaration to match our implementation.

7. Switch back to the tab labelled **"datalog_application.c"** and add the lines highlighted in the red box in Figure 10 to the file.

Figure 10: Adding code to perform simple gesture detection.

8. Terminate and remove all previous applications from the SensorTile board.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

9. Build the project. Solve any errors that you may see. Be very careful in matching the code exactly according to the figures above.

10. Run the application in debug mode on the SensorTile device. Inspect the data transmitted by the SensorTile device to your personal computer.

11. Perform the steps required from the state machine in Figure 4 to make the "Flipping Gesture Detected!" appear on your personal computer. Take a screenshot of this message appearing.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

APPENDIX E

Tutorial 4

Introduction to Audio Sampling

and Signal Processing

STMicroelectronics SensorTile Tutorial: Audio Sampling and Signal Processing

156

Table of Contents

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

1. Introduction to This Tutorial

The Tutorial steps provide:

1. An introduction to audio sampling from the SensorTile MEMS Microphone and the audio Analog to Digital Converter (ADC).
2. An introduction to audio output via the SensorTile audio signal Digital to Analog Converter (DAC).
3. Experience in digital signal processing. This will be provided by your modifying the AudioLoop project. Here, you will add brief C code segments that include Infinite Input Response (IIR) discrete time filters to the acquired sound signals.
4. Guidance to sample and actually hear the effects of digital signal processing.
5. Guidance to examine waveforms and spectrograms using an open source tool, Audacity.

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) 1x Headphone or Loudspeaker
7) 1x 3.5mm Audio Cable
8) 1x USB Sound Adaptor (*for Mac Users Only*)
9) Network access to the Internet.

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. Introduction to SensorTile and the System WorkBench Integrated Development Environment (IDE)
2. Sensor System Signal Acquisition, Event Detection and Configuration.

Your instructor will ensure you have the required background regarding digital signal processing.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

2. Audio Sampling by AudioLoop

AudioLoop is an application included in STSW-STLKT01V1 software package, which sends audio signals acquired by the microphone to an on-board DAC via an I^{2}C interface. A digital MEMS microphone samples the analog sound signal and generates a Pulse-Density Modulation (PDM) stream, which is converted into a Pulse-Coded Modulation (PCM) output stream by the hardware. The output stream is then passed to the on-board DAC, allowing the user to play these sounds on speakers or headphones, or record them on a host PC. Please follow the steps below to build and launch AudioLoop on your SensorTile:

1. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the document labelled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.* Select the same workspace as in Tutorial 1.

2. Close any project in the workspace by right-clicking "STM32L4xx-SensorTile" and selecting "delete". Unselect the "Delete project contents on disk" box and press OK.
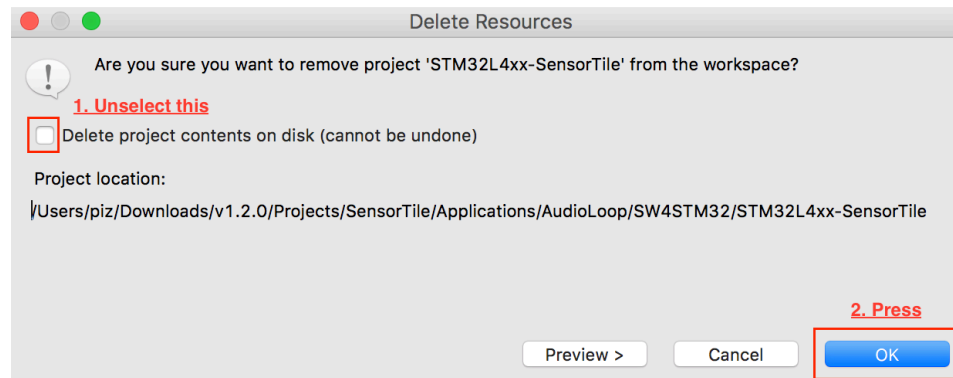
Figure 1: Deleting the existing project in your workspace.

3. Follow Section 7.2 of the Windows tutorial or Section 5.2 of the Mac tutorial *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.* In step 8, uncheck the boxes next to DataLog and BLE_SampleApp to select AudioLoop.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

4. Open the file "**STM32L4XX-SensorTile > Drivers > BSP > SensorTile > SensorTile_audio_out.c**" and modify the "Includes" section by including an additional header file, as shown in Figure 2.
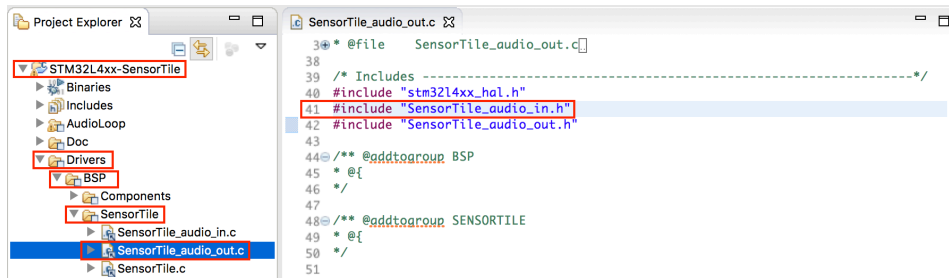
Figure 2: Including an additional header file in "SensorTile_audio_out.c".

5. Build and run AudioLoop in debug mode on the SensorTile board as instructed in **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.**

6. Plug a headphone or loudspeaker into the 3.5mm audio jack. If AudioLoop is running properly on your SensorTile, speak to the microphone and you will be able to hear the sound in your headphone or loudspeaker.
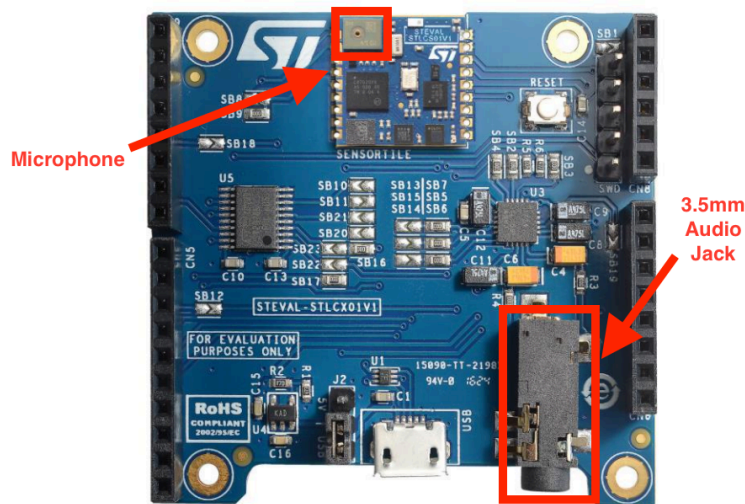
Figure 3: The locations of microphone and audio jack on SensorTile

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

Page 5 of 14

160

www.manaraa.com

7. Play a 4kHz test tone using your PC or Mac notebook. You can find a 4kHz test tone at the following link:

https://www.youtube.com/watch?v=D7M7qWYFpvs

8. Play a 1kHz test tone to the microphone. You can find a 1kHz test tone at the following link:

https://www.youtube.com/watch?v=3FBijeNg_Gs&t=27s

9. We can also generate, via speech, signals that are characteristic of human speech.

10. For most individuals, the generation of the consonant "s" letter produces a sound creating a waveform composed of signals with frequencies above 3000 Hz. To generate this sound, consider the word "hiss" with a lengthy extension of the "ss" segment.

11. Now, certain words or utterances in human speech produce low frequency sound. In particular, vowels produce low frequency sound. The vowel "o" is an ideal example. The sound generated by uttering the "boot" but not pronouncing the "b" or "t" characters and with a lengthy extension of the "oo" segment. This produces a waveform with frequencies predominantly below 1000 Hz.

12. Listen to these sounds now. We note that no filtering of the signals is applied. The sound will change in character once we apply digital signal processing filters to the audio output in the next section.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

3. Digital Signal Processing – Discrete-Time IIR Filtering

3.1. Introduction

In AudioLoop, the acquired sound signals are sampled from the MEMS microphone input sensor and converted to a digital data stream by the ADC. This data is then applied to the output DAC to provide analog output signals that will be applied to the audio output. This audio output signal is then available to drive a pair of ear phones, headphones, or a speaker system.

After the data is sampled and prior to its being applied to the output DAC, we may introduce signal processing algorithms that operate on this data. This provides an outstanding introduction to signal processing and associated IoT system development.

The signal processing capability is demonstrated by adding discrete-time Infinite-Impulse-Response (IIR) filters to the original AudioLoop application in this section.

We will implement two types of digital filters: low pass filters and high pass filters. Low pass filters attenuate signals with frequencies higher than a defined corner frequency, while high pass filters attenuate signals with frequencies lower than a defined corner frequency.. Here, we will modify the AudioLoop application to add first-order low and high pass filters and examine the effects of the filters by hearing and visualizing the output signals.

3.2. Modifications in main.c

1. Open the IDE and open AudioLoop as instructed in **Section 2: Audio Sampling by AudioLoop** in this tutorial. Make sure that you have included the proper header file as indicated in step 4.

2. Once the IDE is opened, navigate to AudioLoop/User/main.c. Double click on main.c to open **main.c**. Modify the "Private define" declaration section as in Figure 4.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

```
38  /* Includes ------------------------------------------------------------*/
39
40  #include <string.h> /* strlen */
41  #include <stdio.h>  /* sprintf */
42  #include <math.h>   /* trunc */
43  #include "main.h"
44
45  /* Private typedef -----------------------------------------------------*/
46  /* Private define ------------------------------------------------------*/
47  #define AUDIO_CHANNELS                    1
48  #define AUDIO_SAMPLING_FREQUENCY             48000
49  #define AUDIO_IN_BUF_LEN   (AUDIO_CHANNELS*AUDIO_SAMPLING_FREQUENCY/1000)
50  #define AUDIO_OUT_BUF_LEN  (AUDIO_IN_BUF_LEN*8)
51
52  /* Define for DSP -- IIR Filter start -------------*/
53  #define F0 3000
54  #define HIGH_PASS_FILTER 0
55  #define LOW_PASS_FILTER 1
56  #define NO_FILTER 2
57  #define GAIN 1024
58  /* Define for DSP -- IIR Filter end --------------*/
```

Figure 4: Declaring DSP IIR filter parameters and variables.

3. Add the lines of code in the red box into **AudioProcess** function as shown in Figure 5.

4. Note that these define a corner frequency (3000 Hz).

5. These also define values or the conditions of Low Pass, High Pass, and No Filter operation.

6. Since we wish to perform all arithmetic with methods that avoid the requirements for floating point computation we will declare all critical variables as integer data types.

7. This leads to a limitation since some values of our signal are small integers. This leads to inevitable rounding errors in computation.

8. These rounding errors can be avoided, however, by first multiplying these variables by a gain factor, defined here as GAIN.

9. After computation, the output signal will be divided by GAIN to restore the proper amplitude.

```
123    static uint32_t IndexOut = 0;
124    static uint32_t AudioOutActive = 0;
125    uint32_t indexIn;
126
127    /* IIR filter setup */
128    int16_t filter_type = NO_FILTER;
129    int16_t IWon;
130    int16_t c[3];
131    int16_t prev_in, cur_in;
132    int32_t new_out;
133
134    IWon = (int) AUDIO_SAMPLING_FREQUENCY / (3.14159 * F0) ;
135    if (filter_type == LOW_PASS_FILTER) {
136        c[0] = GAIN / (1.0f + IWon);
137        c[1] = c[0];
138        c[2] = c[0] * (1.0f - IWon);
139    }
140    else if (filter_type == HIGH_PASS_FILTER) {
141        c[0] = GAIN - (GAIN / (1.0f + IWon));
142        c[1] = -c[0];
143        c[2] = (1.0f / (1.0f + IWon)) * (GAIN - IWon * GAIN);
144    } else if (filter_type == NO_FILTER) {
145        c[0] = GAIN;
146        c[1] = 0;
147        c[2] = 0;
148    }
149    /* IIR filter setup end */
150
151    for(indexIn=0;indexIn<AUDIO_IN_BUF_LEN;indexIn++)
```

Figure 5: Adding IIR filter setup section to the AudioProcess function.

10. Apply filters and assign filtered audio data to audio_out_buffer as shown in Figure 6.

```
150    /* IIR filter setup end */
151
152    for(indexIn=0;indexIn<AUDIO_IN_BUF_LEN;indexIn++)
153    {
154        if(indexIn == 0) {
155            audio_out_buffer[IndexOut++] = PCM_Buffer[indexIn];
156            audio_out_buffer[IndexOut++] = PCM_Buffer[indexIn];
157        } else {
158            cur_in = PCM_Buffer[indexIn];
159            new_out = cur_in * c[0] + prev_in * c[1] - audio_out_buffer[IndexOut - 2] * c[2];
160            audio_out_buffer[IndexOut++] = new_out / GAIN;
161            audio_out_buffer[IndexOut++] = new_out / GAIN;
162            prev_in = cur_in;
163        }
164
165        /*  audio_out_buffer[IndexOut++] = PCM_Buffer[indexIn];
166            audio_out_buffer[IndexOut++] = PCM_Buffer[indexIn];
167        */
168    }
169
170    if(!AudioOutActive && IndexOut==AUDIO_OUT_BUF_LEN/2)
```

Figure 6: Applying filter to audio output data.

11. Terminate and remove all previous applications from the SensorTile board.

12. Compile and run the AudioLoop application on the SensorTile board in debug mode.

13. Speak the "ss" and "oo" sounds and play the 4kHz and 1kHz test tones again. Listen to the unfiltered sounds in your headphone or speaker.

14. Change the value of the variable "*filter_type*" shown in Figure 5 from "*NO_FILTER*" to "*LOW_PASS_FILTER*" and to "*HIGH_PASS_FILTER*" and repeat step 11 to step 13.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

4. Examining waveforms and spectrograms in Audacity

4.1. Introduction

This section will guide the user to examine the waveforms and spectrograms of the audio output from SensorTile using Audacity, an audio software for recording and editing. Please open the following link on a web-browser on your PC and follow the directions to download the software:

http://www.audacityteam.org/

You can examine the effect of the filters by analyzing the output waveforms and spectrograms. For more information about spectrograms, please click on the following link from Wikipedia on a web-browser on your PC:

https://en.wikipedia.org/wiki/Spectrogram

4.2. Recording and Playing a Sound in Audacity

1. Compile and run AudioLoop with filters as instructed in Section 3 of this tutorial

2. Instead of plugging in a headphone/loudspeaker, plug in one end of a 3.5mm audio cable into the audio jack of your SensorTile, and plug another end of the cable into the audio jack of your PC. Note that Mac users should have a USB sound adaptor available and connect the audio cable to the Mac through the adaptor.

3. Open Audacity on your PC, you should see a window as shown in Figure 7.
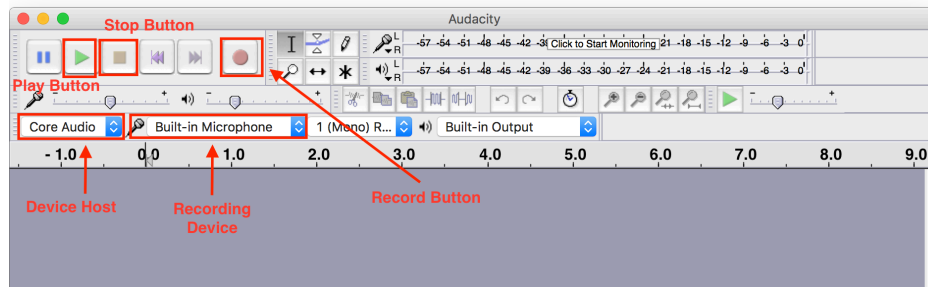
Figure 7: Audacity GUI.

For more information about the Audacity GUI, please go to **Audacity > Help > Manual.**

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

4. Select the corresponding recording device to your SensorTile. In a Windows machine, the name can be "Microsoft Sound Mapper". In a Mac machine with a USB sound adaptor connector connected, the name can be "C-Media USB Audio Device".

5. If you cannot find the corresponding device, please restart Audacity after plugging your audio cable or your USB sound adaptor in your PC. If this solution cannot work, please consult your instructor.

6. Press the Record Button as shown in Figure 7 to start recording.

7. Press the Stop Button to stop recording. Figure 8 shows a waveform of silence (pure noise). You can close it by clicking the Close Button.

Figure 8: Audacity showing an audio waveform.

8. Press the Play Button to play the recorded sound. If there are multiple tracks available at the same time, you can click the Mute Button to mute a track, or select the Solo Button to play just that track.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering

4.3. Analyzing Spectrograms in Audacity

1. Click the Audio Track Drop Down Menu shown in Figure 8, select "Spectrogram", and you should see a spectrogram as in Figure 9. Note that the blue color is the least energy and the red and white are the most.
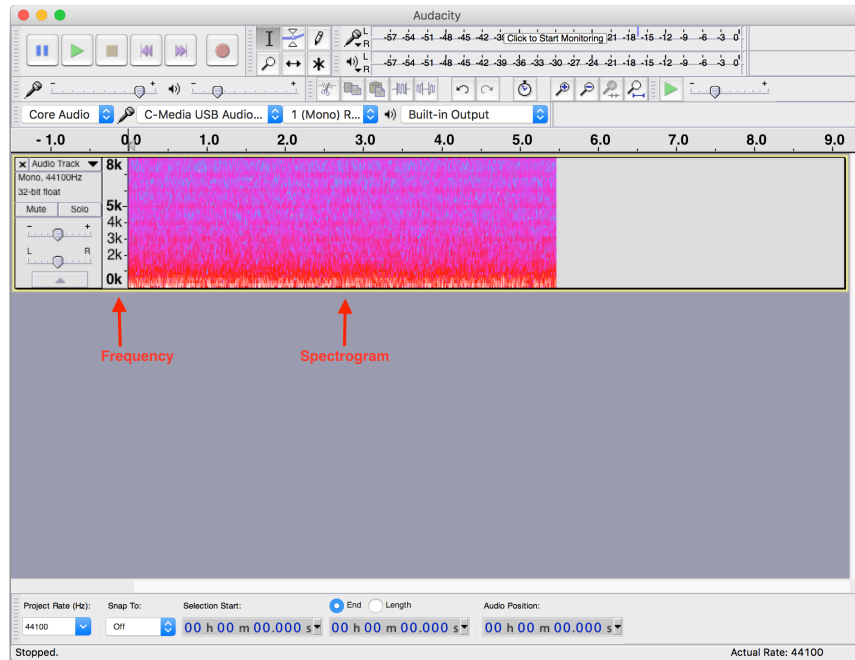
Figure 9: Audacity showing a spectrogram.

2. Examine the waveform and spectrogram of the "ss" sound after applying each type of filter (Low Pass Filter and High Pass Filter). Can you see any difference? In which frequency region does the most of energy reside for the output after each type of filter?

3. Repeat step 2 for the "oo" sound.

4. Repeat step 2 for the 4kHz test tone.

5. Repeat step 2 for the 1 kHz test tone.

6. Please see the figures in the next page for example waveforms and spectrograms from step 2 and step 3.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering
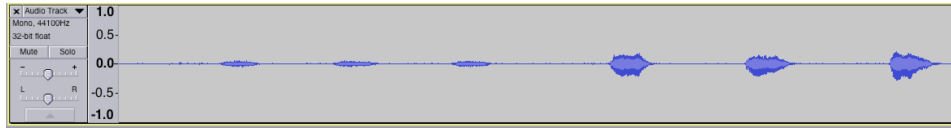
Waveform display with Low Pass Filter applied with three sounds of "sss" and "ooo". Note that the "sss" sound is attenuated in volume while the "ooo" sounds of predominantly low frequency components are preserved.
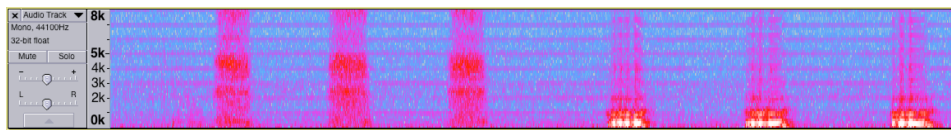
Spectrogram with Low Pass Filter applied with three sounds of "sss" and "ooo". Note that the "sss" sound is attenuated in volume while the "ooo" sounds of predominantly low frequency components are preserved.

High Pass Filter applied with three sounds of "sss" and "ooo". Note that the "sss" sound is increased in relative volume while the "ooo" sounds of predominantly low frequency components are now distorted.

STMicroelectronics SensorTile Tutorial: Audio Sampling and Filtering
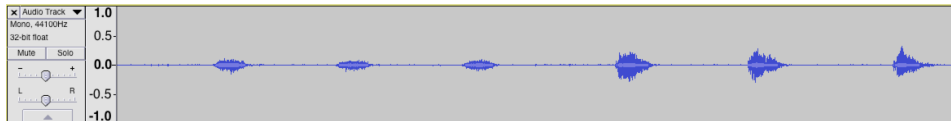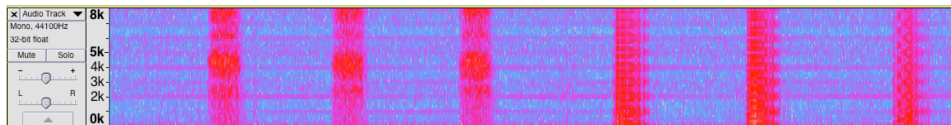
APPENDIX F

Tutorial 5

SensorTile Firmware Programming

STMicroelectronics SensorTile Tutorial: Firmware Programming

Table of Contents

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

www.manaraa.com

1. Introduction to This Tutorial

The Tutorial steps provide:

1. An introduction to customize the starter firmware, STSW-STLKT01 package
2. Guidance to add new source files, create new linking paths, and properly configure a project.
3. Demonstration of the customized DataLog application, which will call a function from a newly added source to calculate the acceleration vector amplitude.

For more information regarding the SensorTile board, please open the following link. www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. Introduction to SensorTile and the System WorkBench Integrated Development Environment (IDE)
2. Sensor System Signal Acquisition, Event Detection and Configuration.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

2. Adding New Source Files, Paths, and Project Configuration

It is not uncommon to split a complete project into multiple source and header files for elegance and clearness. You may have noticed that the STSW-STLKT01 package manages the projects exactly in this way. We will now go through the process of adding your own source file, linking the corresponding header file, and configuring your project appropriately. This enables you to customize your own project.

1. Navigate to the local folder containing the STSW-STLK01 package *"../v1.2.0/"* and create a subdirectory "**Tutorial5**", as shown in Figure 1.
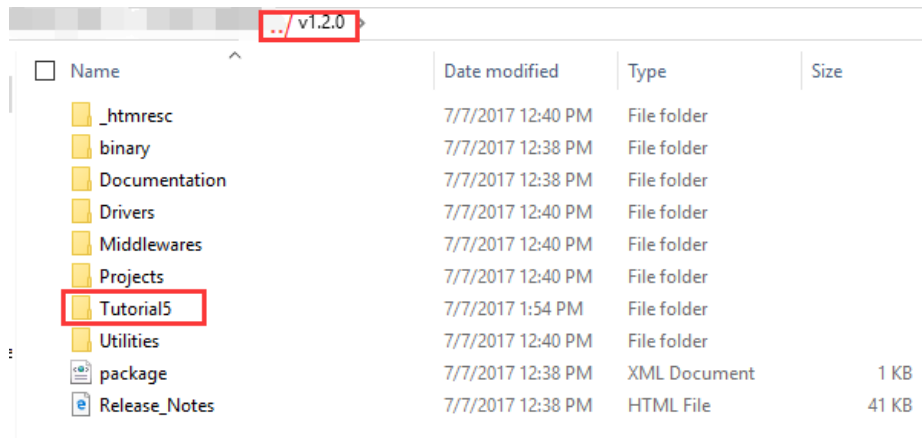
Figure 1: Add new directory into current project source file directory

2. Navigate to "**Tutorial5**" and download **SensorTile_abs_acc.c** and **SensorTile_abs_acc.h** into the folder. See Figure 2.

   You can download these files from the link:

   http://www.st.com/content/st_com/en/premium-content/sensortile-curriculum-SensorTIle_abs_acc_zip.html

   Fill out the form and a download link will be sent to your email.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

Name	Date modified	Type	Size
SensorTile_abs_acc.c	7/7/2017 12:56 PM	C Source	1 KB
SensorTile_abs_acc.h	7/7/2017 12:55 PM	C/C++ Header	1 KB

Figure 2: Download required files to the new directory.

3. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the document labelled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example.**

   Select the same workspace as in Tutorial 1.

4. ***Save and remove*** all other active projects in System WorkBench. Import the ***DataLog*** project as instructed in the previous tutorials.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

Page 5 of 15

5. **Once you successfully import the DataLog project again**, select **STM32L4xx-SensorTile -> DataLog -> User** in the Project Explorer, right-click on User, and select import. Make sure that the User folder is highlighted in blue. See Figure 3.
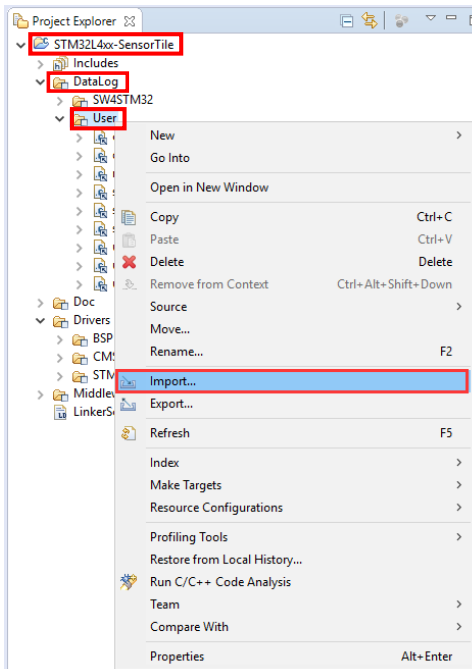
Figure 3: Importing a new source file

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

Page 6 of 15

176

6. In the pop-up window, select **General -> File System** and then Press Next. See Figure 4.
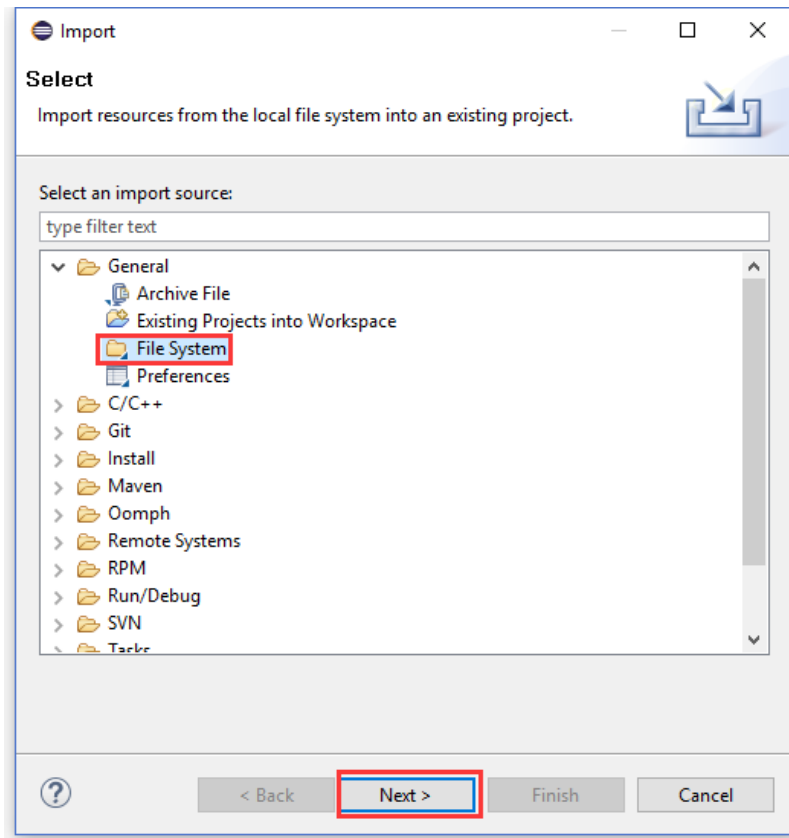
Figure 4: Importing a source file from File System.

7. In the next pop-up window, click on the top "Browse" button to select the "Tutorial5" folde, as shown in Figure 5. Make sure that the "Into folder" is correct, as shown in step 3 of Figure 5.
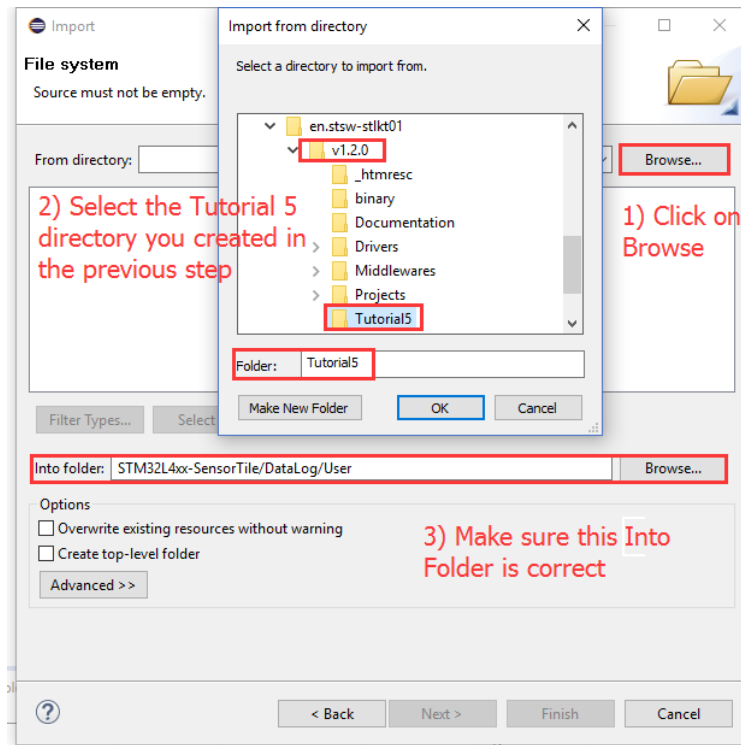
Figure 5: Selecting the directory containing the source code.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

8. Check the file to be imported, click on **Advanced** and check **Create links in workspace**, and click **Finish**. See Figure 6.
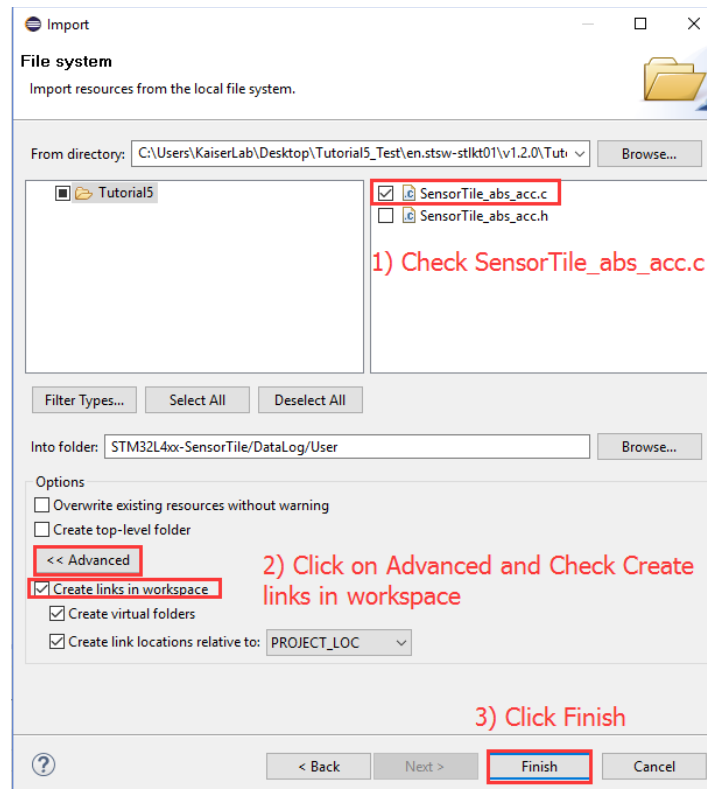
Figure 6: Checking import files and creating links in workspace in Advanced settings.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

9. Make sure the file is imported successfully into the correct folder, as shown in Figure 7.
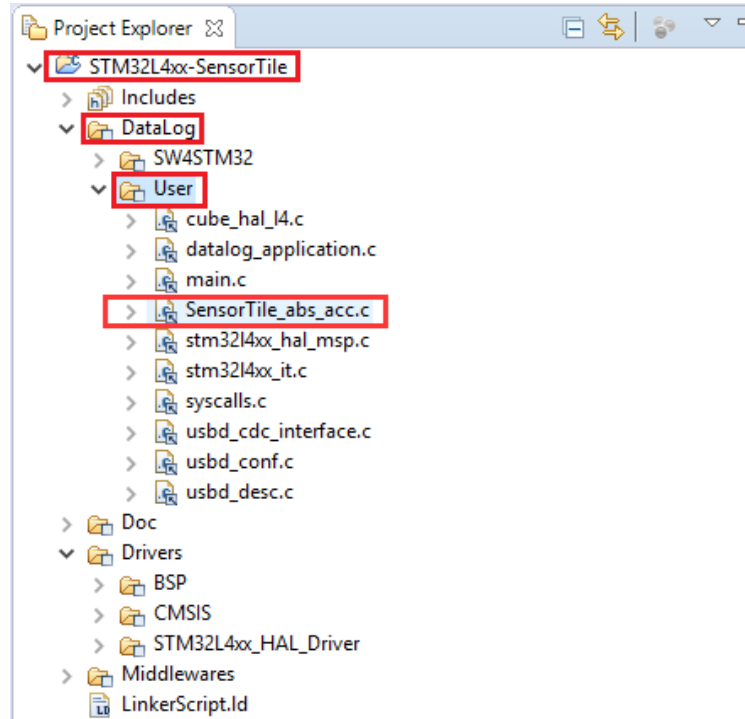
Figure 7: Imported SensorTile_abs_acc.c.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

10. Right-click on **STM32L4xx-SensorTile** and select **Properties**. See Figure 8.
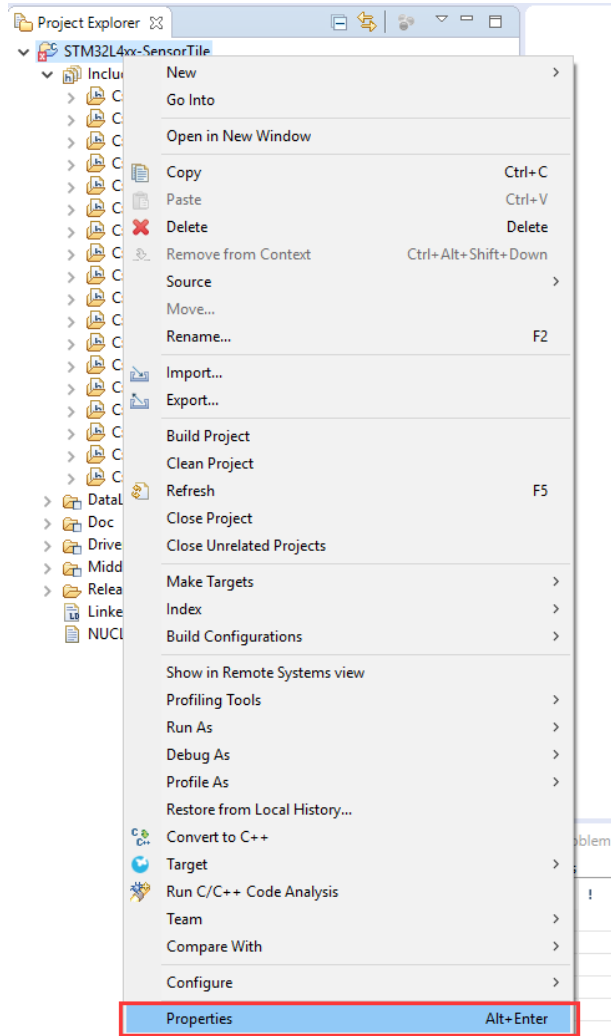
Figure 8: Open project properties.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

11. Add the header to the includes paths as instructed in Figure 9. This enables System WorkBench to find the dependent header files for the related sources files.
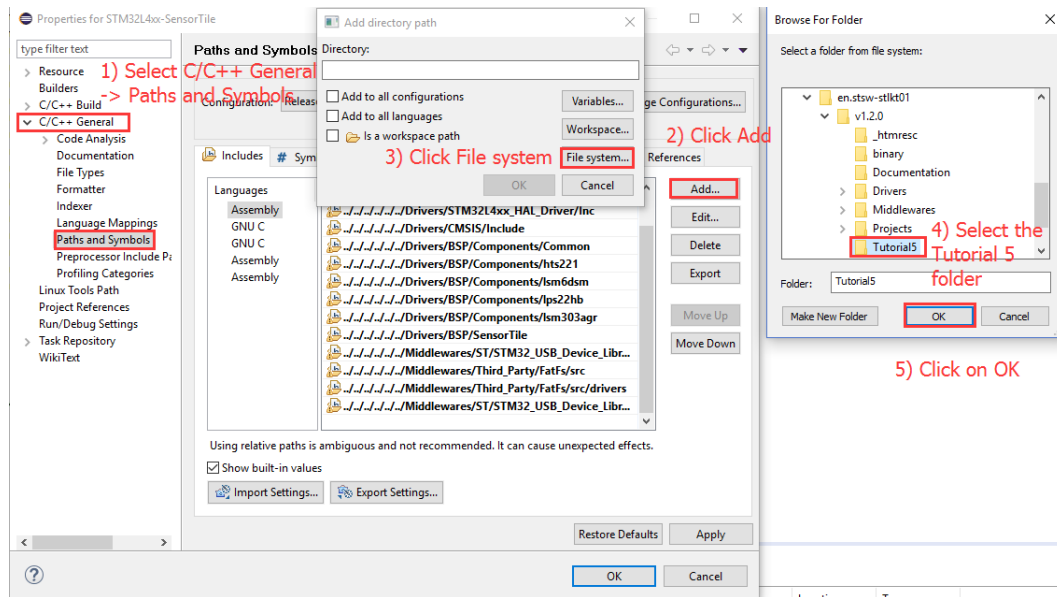
Figure 9: Adding a new path.

12. Check **Add to all configuration** and **Add to all languages**. Press OK. See Figure 10.
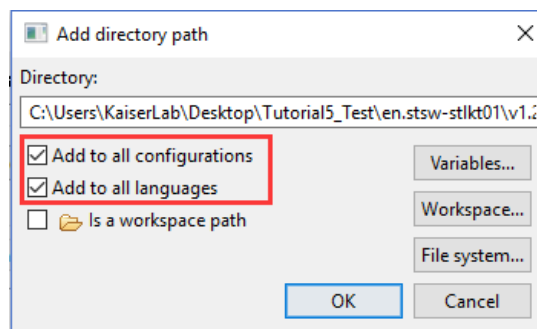
Figure 10: Adding directory path to all configurations and all languages.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

13. You can see that Tutorial 5 directory is successfully added into the project path. See Figure 11.
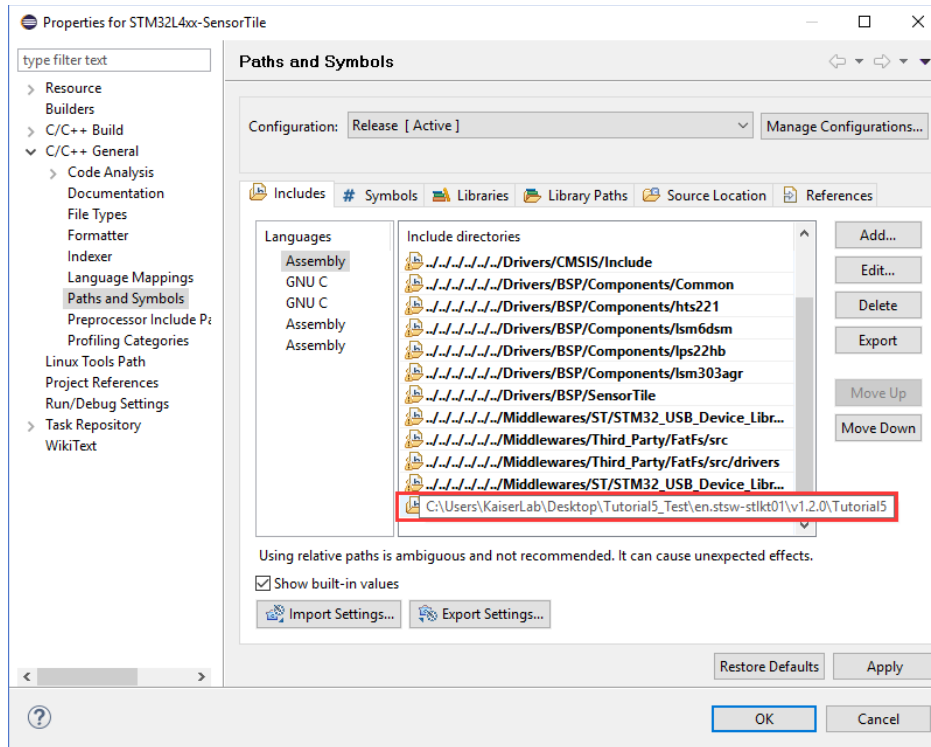
Figure 11: Newly added directory path.

14. Now we need to navigate to datalog_application.c and include the header file. See Figure 12.

```
.c datalog_application.c ⊠   .c SensorTile_abs_acc.c

  3⊕   * @file     DataLog/Src/datalog_application.c
 28
 29   /* Includes --------------------------------
 30   #include "datalog_application.h"
 31   #include "main.h"
 32   #include "string.h"
 33   #include "SensorTile.h"
 34   #include <math.h>
 35   #include "SensorTile_abs_acc.h"
 36
 37   /* FatFs includes component */
 38   #include "ff_gen_drv.h"
 39   #include "sd_diskio.h"
```

Figure 12: Add header files.

15. Open the declaration for the function labelled **Accelero_Sensor_Handler** as instructed in the document labelled ***STMicroelectronics SensorTile Tutorial: Sensor System Signal Acquisition, Event Detection and Configuration***.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

16. Modify the code as shown in Figure 13.

```
if(SendOverUSB) /* Write data on the USB */
{
  sprintf( dataOut, "\n\rACC_X: %d, ACC_Y: %d, ACC_Z: %d, |ACC|: %d",
           (int)acceleration.AXIS_X, (int)acceleration.AXIS_Y, (int)acceleration.AXIS_Z,
           get_abs_acc(acceleration.AXIS_X, acceleration.AXIS_Y, acceleration.AXIS_Z));
  CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));

  if ( verbose == 1 )
  {
    if ( BSP_ACCELERO_Get_WhoAmI( handle, &who_am_i ) == COMPONENT_ERROR )
    {
      sprintf( dataOut, "WHO AM I address[%d]: ERROR\n", id );
    }
    else
    {
      sprintf( dataOut, "WHO AM I address[%d]: 0x%02X\n", id, who_am_i );
    }

    CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));

    if ( BSP_ACCELERO_Get_ODR( handle, &odr ) == COMPONENT_ERROR )
    {
      sprintf( dataOut, "ODR[%d]: ERROR\n", id );
    }
    else
    {
      floatToInt( odr, &d1, &d2, 3 );
      sprintf( dataOut, "ODR[%d]: %d.%03d Hz\n", (int)id, (int)d1, (int)d2 );
    }
```

Figure 13: Calling the get_abs_acc() function to calculate acceleration vector magnitude.

17. Terminate and remove all previous applications from the SensorTile board.

18. Compile and run the DataLog application on the SensorTile board in debug mode.

19. Examine the data transmitted over the Serial USB connection to your personal computer. Take a screenshot.

STMicroelectronics SensorTile Tutorial: Sensor Orientation Detection

APPENDIX G

Tutorial 6

Introduction to Bluetooth Low Energy Wireless Interfaces

STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces

Table of Contents

188

1. Introduction to This Tutorial

SensorTile has a Bluetooth module (BLUENRG-MS) to support the BLE communication. In the tutorial, we will introduce the BLE_SampleApp project in the starter firmware package to explore the BLE communication on SensorTile.

The Tutorial steps provide:

1. An introduction to installation of Bluetooth Low Energy (BLE) starter firmware.
2. An introduction to BLE connection to SensorTile via BlueZ on Linux system.

For more information regarding the SensorTile board, please open the following link. www.st.com/sensortile

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.
7) A Linux machine (Beaglebone, Intel Edison, Rpi, and etc)

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. All previous tutorials.

Your instructor will ensure you have the required background regarding BLE fundamentals including BLE architecture and BLE protocol.

STMicroelectronics SensorTile Tutorial: BLE Introduction

2. BLE Firmware Installation

In the previous documents, we explored the DataLog application and AudioLoop application in the SensorTile starter firmware package. We will now dig into BLE firmware and examine how to communicate with SensorTile using BLE via Bluez (Bluetooth communication utility for Linux platform) on Linux platform.

1. Open the IDE (Eclipse or System WorkBench) on your personal computer as instructed in the Tutorial 1*.* Select the same workspace as in Tutorial 1.

2. Once the IDE is open, first remove your current project in your IDE and import the BLE_SampleApp project from corresponding directories as instructed in the document labeled *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example*. In order to properly import the BLE project, you need to make sure you *uncheck* DataLog and AudioLoop. See Figure 1.
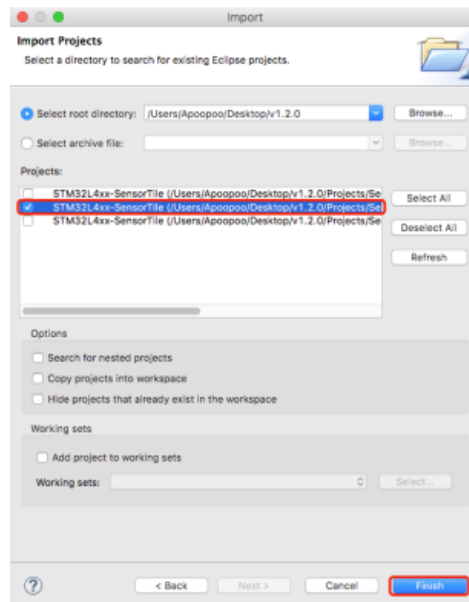
Figure 1: Import BLE_SampleApp project from starter firmware package.

STMicroelectronics SensorTile Tutorial: BLE Introduction

3. Once you successfully import the BLE project. Open main.c and main.h. See Figure 2.
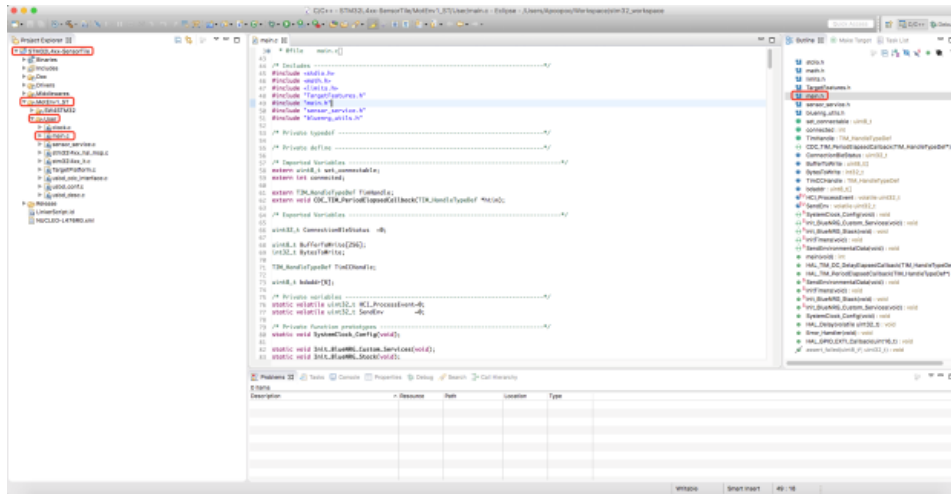
Figure 2: Open main.c and main.h.

4. Modify the define parameter at line 61 in main.h such that it matches Figure 3. This will enable the USB debug interface for the BLE firmware.
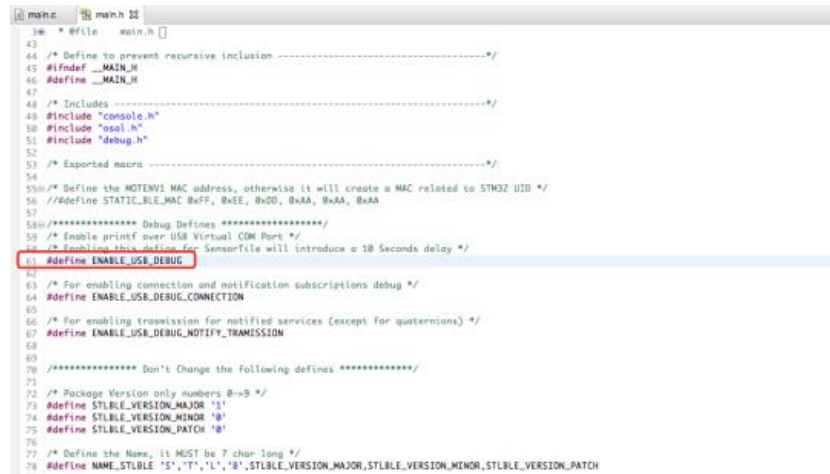
Figure 3: Modify the define parameter and enable USB debug interface for BLE.

STMicroelectronics SensorTile Tutorial: BLE Introduction

5. Save the modification.

6. Terminate and remove all previous applications from the SensorTile board.

7. Compile and run the BLE_SampleApp application on the SensorTile board in debug mode.

8. Examine the LED on SensorTile. Your SensorTile should start to blink once every second, which indicates that SensorTile is running BLE firmware properly.

9. Now you can check the BLE debug interface through USB streaming. You can use terminal screen command (Mac user) or Putty serial connection (Windows user) to check the BLE operations on SensorTile as you check the DataLog data streaming, which is illustrated in *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example* -> *Example Data Logging Project -> Debug*.
The BLE debug interface is indicated as Figure 4. Make sure you do not close the debug interface because you will use it in the following session.

Figure 4: BLE debug interface via USB streaming.

192

www.manaraa.com

3. BLE Communication via BlueZ

SensorTile is able to communicate with mobile devices like smartphones and embedded Linux devices such as BeagleBone and Rpi through Bluetooth. BlueZ is the tool we will use on embedded Linux to deal with the BLE communication. In this session, we will use BeagleBone Wireless Green to introduce BlueZ.

3.1. BlueZ Introduction

BlueZ is the official Bluetooth stack for Linux kernel-based family of operating systems. Its goal is to program an implementation of the Bluetooth wireless standards specifications for Linux. BlueZ also provides support for the core Bluetooth layers and protocols. It is flexible, efficient and uses a modular implementation. For more information, you can refer to BlueZ website http://www.bluez.org/.

3.2. BlueZ Installation and Update

1. Log into your BeagleBone and make sure your BeagleNone is successfully connected with WiFi.

2. Update your apt-get by using command **$ sudo apt-get update** in BeagleBone.

3. Type the command **$ sudo apt-get install bluez** to install BlueZ or check your current BlueZ version if it has already been installed. See Figure 5.

```
root@beaglebone:~# sudo apt-get install bluez
Reading package lists... Done
Building dependency tree
Reading state information... Done
bluez is already the newest version.
The following packages were automatically installed and are no longer required:
  libaudio2 libmng1 libqt4-network libqt4-xml libqtcore4 libqtdbus4 libqtgui4
  qtcore4-l10n
Use 'apt-get autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 127 not upgraded.
root@beaglebone:~#
```

Figure 5: Use apt-get to check installation of BlueZ.

STMicroelectronics SensorTile Tutorial: BLE Introduction

3.3. Bluetoothctl Utilities in BlueZ

1. Once you have confirmed that you have the BlueZ installed on your BeagleBone, you can use the command **$ bluetoothctl** to use the bluetoothctl utilities interactively in BeagleBone. See Figure 6.

Figure 6: Bluetoothctl utilities in BlueZ.

2. Type **list** in bluetoothctl and this will show your device Bluetooth controller MAC address. See Figure 7.

Figure 7: Bluetoothctl Bluetooth controller MAC address.

3. Type **select MAC_ADDRESS** as your default Bluetooth device controller as Figure 8.

Figure 8: Select the default controller.

4. Type **power on** to turn on the power as Figure 9.

Figure 9: Turn on controller power.

STMicroelectronics SensorTile Tutorial: BLE Introduction

5. Type *agent on* to turn on agent. See Figure 10.

Figure 2: Turn on agent.

6. Type *scan on* to scan the Bluetooth devices and the device with name STLB100 is your SensorTile device. See Figure 11.

Figure 11: Scan Bluetooth devices.

7. Type *pair MAC_ADDRESS_SensorTile* to make your BeagleBone pair with SensorTile. See Figure 12. The Beaglebone automatically connects with the SensorTile after pairing.

Figure 12: Pair devices.

8. Now, you can switch back to the BLE debug interface, you can find that you have already connected with your Beaglebone. See Figure 13. You can also check the LED on SensorTile, which *stops* blinking.

STMicroelectronics SensorTile Tutorial: BLE Introduction

Figure 13: BLE debug interface return message.

9. Switch back to BeagleBone and type *info MAC_ADDRESS_SensorTile* in bluetoothctl utilities. You can see more information related to your SensorTile BLE information. See Figure 14.

Figure 14: Information of SensorTile BLE.

10. You can disconnect the BLE connection with the SensorTile by typing *disconnect MAC_ADDRESS_SensorTile* in bluetoothctl utilities. See Figure 15.

STMicroelectronics SensorTile Tutorial: BLE Introduction

Page 10 of 11

196

www.manaraa.com

Figure 15: BLE disconnection with SensorTile.

11. You can switch back to the BLE debug interface to check the feedback information. See Figure 16. You can also check the LED on SensorTile. It starts to blink again because SensorTile is advertising itself when there is no BLE connection with it.

Figure 16: SensorTile debug interface.

12. Now you can switch back to BeagleBone and type **connect MAC_ADDRESS_SensorTile** in bluetoothctl utilities to re-connect with SensorTile. You can also type **help** to explore all the utilities in bluetoothclt. When there is an interaction with SensorTile, you can always check the SensorTile BLE debug interface for more information.

13. Type **exit** to quit bluetoothclt utility.

STMicroelectronics SensorTile Tutorial: BLE Introduction

APPENDIX H

Tutorial 7

Introduction to Bluetooth Low Energy Communication

and the GATT Profile

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile

199

Table of Contents

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

200

1. Introduction to This Tutorial

In *STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces*, we introduced how to use BlueZ to discover, pair, and connect with SensorTile on Linux platform. In this tutorial, we will use BlueZ gatttool utility to read environmental data from SensorTile Bluetooth Generic Attribute Profile.

The Generic Attributes (GATT) define a hierarchical data structure that is exposed to connected Bluetooth Low Energy devices. GATT is built on top of the Attribute Protocol (ATT), which uses GATT data to define the way that two Bluetooth Low Energy devices send and receive standard messages. In this tutorial, we will use SensorTile as the GATT server to provide services and use BeagleBone as the GATT client to send requests to the GATT server to get environmental data collected on SensorTile.

The Tutorial steps provide:

1. An introduction to *gatttool*
2. An introduction to request environmental data using gatttool
3. An introduction to control the LED on SensorTile using gatttool

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

For more information regarding Bluetooth, please refer to Bluetooth official website.

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.
7) A Linux machine (BeagleBone, Intel Edison, Rpi, and etc)

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

1. STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces

Your instructor will ensure you have the required background regarding Bluetooth.

2. Introduction to Gatttool

Gatttool is the BlueZ utility that handles BLE communication with GATT. We will use it to discover all the services and characteristics provided by SensorTile, which is the GATT server. Before you start this part, make sure you have successfully installed BlueZ on your Linux machine and flashed the BLE_SampleApp firmware to SensorTile as instructed in *STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces.*

1. Connect your SensorTile with your computer and use terminal screen command (MAC user) or Putty serial connection (Windows user) to access the USB debug interface of BLE_SampleApp firmware as instructed in *STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example -> Example Data Logging Project -> Debug*. See Figure 1.

Figure 1: BLE debug interface via USB streaming.

2. Connect your BeagleBone with your computer and log into BeagleBone.

3. In BeagleBone, type command *$ bluetoothctl* to use BlueZ in interactive mode. See Figure 2.

Figure 2: Use bluetootctl in BeagleBone

4. As instructed in **STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces,** discover, pair, and **disconnect** your SensorTile device in bluetoothctl. Use command *info* in bluetoothctl to check that your SensorTile is paired but **disconnected**. See Figure 3.

Figure 3: Use bluetoothctl to check if the SensorTile is paired and disconnected

5. Copy the MAC address of SensorTile and exit bluetoothctl. See Figure 4.

Figure 4: Exit bluetoothctl

6. In BeagleBone, use command **$ gatttool –b <MAC_ADDRESS_SENSORTILE> –t random –I** to run gatttool in interactive mode. **–b** is the option to indicate the mac address. **–t** is the option to declare the Bluetooth MAC address type. In BLE_SampleApp firmware, this is set to type random. **–I** is the option to use gatttool in interactive mode. See Figure 5.

Figure 5: Use gatttool with proper options in interactive mode

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

7. In gatttool, use command ***connect*** to connect BeagleBone with SensorTile. If the
    connection is successful, the MAC address should turn into color of blue. See Figure 6.

```
[C0:6E:2C:31:25:48][LE]> connect
Attempting to connect to C0:6E:2C:31:25:48
Connection successful
[C0:6E:2C:31:25:48][LE]>
```

Figure 6: Connect with SensorTile in gatttool interactive mode

8. In USB debug interface, you can also see the successful connection information. See
    Figure 7.

```
STMicroelectronics STLBLE1:
        Version 1.0.0
        SensorTile
OK Temperature Sensor1
OK Pressure Sensor
Enabled Temperature Sensor1
Enabled Pressure Sensor
        (HAL 1.5.1_0)
        Compiled Aug 14 2017 14:24:28 (openstm32)
        Send Every  500mS Temperature/Humidity/Pressure
Debug Connection          Enabled
Debug Notify Trasmission Enabled
SERVER: BLE Stack Initialized
                Board type=SensorTile HWver=49, FWver=7.2.c
                BoardName= STLB100
                BoardMAC = c0:6e:2c:31:25:48

HW      Service W2ST added successfully
Config  Service W2ST added successfully
>>>>>>CONNECTED 5c:f8:21:d6:9d:2d
```

Figure 7: BLE_SampleApp USB debug interface for successful connection

9. In gatttool, use command ***primary*** to discover all the primary services. See Figure 8.

```
[C0:6E:2C:31:25:48][LE]> primary
attr handle: 0x0001, end grp handle: 0x0004 uuid: 00001801-0000-1000-8000-00805f
9b34fb
attr handle: 0x0005, end grp handle: 0x000b uuid: 00001800-0000-1000-8000-00805f
9b34fb
attr handle: 0x000c, end grp handle: 0x0012 uuid: 00000000-0001-11e1-9ab4-0002a5
d5c51b
attr handle: 0x0019, end grp handle: 0x001c uuid: 00000000-000f-11e1-9ab4-0002a5
d5c51b
```

Figure 8: GATT primary services discovery in gatttool

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

10. In gatttool, use command **characteristics** to discover all the characteristics. See Figure 9.


```
[C0:6E:2C:31:25:48][LE]> characteristics
handle: 0x0002, char properties: 0x20, char value handle: 0x0003, uuid: 00002a05
-0000-1000-8000-00805f9b34fb
handle: 0x0006, char properties: 0x4e, char value handle: 0x0007, uuid: 00002a00
-0000-1000-8000-00805f9b34fb
handle: 0x0008, char properties: 0x4e, char value handle: 0x0009, uuid: 00002a01
-0000-1000-8000-00805f9b34fb
handle: 0x000a, char properties: 0x02, char value handle: 0x000b, uuid: 00002a04
-0000-1000-8000-00805f9b34fb
handle: 0x000d, char properties: 0x12, char value handle: 0x000e, uuid: 00140000
-0001-11e1-ac36-0002a5d5c51b
handle: 0x0010, char properties: 0x12, char value handle: 0x0011, uuid: 20000000
-0001-11e1-ac36-0002a5d5c51b
handle: 0x001a, char properties: 0x14, char value handle: 0x001b, uuid: 00000002
-000f-11e1-ac36-0002a5d5c51b
```

Figure 9: GATT characteristics discovery in gatttool

11. In gatttool, use command **char-desc** to discover all the characteristic descriptors. See Figure 10.


```
[C0:6E:2C:31:25:48][LE]> char-desc
handle: 0x0001, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0002, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0003, uuid: 00002a05-0000-1000-8000-00805f9b34fb
handle: 0x0004, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x0005, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x0006, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0007, uuid: 00002a00-0000-1000-8000-00805f9b34fb
handle: 0x0008, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0009, uuid: 00002a01-0000-1000-8000-00805f9b34fb
handle: 0x000a, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x000b, uuid: 00002a04-0000-1000-8000-00805f9b34fb
handle: 0x000c, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x000d, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x000e, uuid: 00140000-0001-11e1-ac36-0002a5d5c51b
handle: 0x000f, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x0010, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x0011, uuid: 20000000-0001-11e1-ac36-0002a5d5c51b
handle: 0x0012, uuid: 00002902-0000-1000-8000-00805f9b34fb
handle: 0x0019, uuid: 00002800-0000-1000-8000-00805f9b34fb
handle: 0x001a, uuid: 00002803-0000-1000-8000-00805f9b34fb
handle: 0x001b, uuid: 00000002-000f-11e1-ac36-0002a5d5c51b
handle: 0x001c, uuid: 00002902-0000-1000-8000-00805f9b34fb
```

Figure 10: GATT characteristic descriptors discovery in gatttool

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

3. Request Environmental Data Using Gatttool Interactively

Refer to Figure 10, there are different handles corresponding to different uuids. You can use handles to communicate with SensorTile for BLE services and characteristics.

Handles in the red rectangular are very important:
> 0x000e is characteristic value handle of environmental data.
> 0x000f is client characteristic configuration handle for environmental data transmission.
> 0x001b is SensorTile LED configuration handle.
You will use these three handles for the following sessions.

In this session, you will use gatttool to request environmental data.

1. Make sure that the SensorTile is connected with BeagleBone in gatttool.

2. In gatttool, use command **char-read-hnd 000e** to read characteristic value (environmental data) with handle 0x000e. See Figure 11. The data will be explained the step 4.

Figure 11: Read characteristic value (environmental data) by handle 0x000e

3. In gatttool, use command **char-write-req 000f 0100** to modify the characteristic value of handle 0x000f and therefore enable the notification of environmental data (handle 0x000e) on SensorTile. After you enter the command, SensorTile will start to send environmental data continuously and BeagleBone will listen to the data transmission in gatttool. See Figure 12.

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

Figure 12: Characteristic write to enable environmental data notification in gatttool

4. Switch to the USB debug interface. The debug information will be similar to figure 13.

Figure 13: BLE_SampleApp USB debug interface for enabling environmental data notification

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

Page 10 of 15

208

5. In gatttool, use command **char-write-req 000f 0000** to disable the notification of environmental data. See Figure 14.

Figure 14: Characteristic write to disable environmental data notification in gatttool

6. Switch to USB debug interface. Debug information shows that the environmental data notification is off (disabled). See Figure15.


```
--->Env=ON
Sending: Press=100011 Temp1=271
Sending: Press=100011 Temp1=271
Sending: Press=100010 Temp1=271
Sending: Press=100010 Temp1=271
Sending: Press=100011 Temp1=271
Sending: Press=100010 Temp1=270
Sending: Press=100013 Temp1=270
Sending: Press=100011 Temp1=270
Sending: Press=100010 Temp1=270
Sending: Press=100010 Temp1=270
Sending: Press=100010 Temp1=270
Sending: Press=100010 Temp1=270
Sending: Press=100011 Temp1=270
Sending: Press=100011 Temp1=271
Sending: Press=100013 Temp1=271
Sending: Press=100013 Temp1=270
--->Env=OFF
```

Figure 15: BLE_SampleApp USB debug interface for disabling environmental data notification

7. By comparing Figure 12 and Figure 13, we can interpret the environmental data sent from SensorTile.

In Figure 12, all the data are sent in the format of hexadecimal. The first 4 bits in the yellow rectangle (c2 7a) are timestamps. The next 8 bits in the red rectangle (ab 86 01 00) are air pressure data. The last 4 bits in the blue rectangle (0f 01) are temperature data from temperature sensor 1.

In Figure 13, pressure data are 100011, which are actually 1000.11 mbar. Temperature data are 271, which are actually 27.1 °C.

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

The environmental data sent by notification should be read from right to left (little endian rule), which means that the pressure data are 0x000186ab and the temperature data are 0x010f. By transferring hexadecimal to decimal, the pressure data is 11*1 + 10*16 + 6*256 + 8*4096 + 1*65536 = 100011 and the temperature data is 256*1 + 15 = 271, which is the same as the pressure value and the temperature value in USB debug interface (Figure 13).

You can compare other lines of data in gatttool and USB debug interface to verify the data interpretation above.

210

4. Control SensorTile LED Using Gatttool Interactively

In this session, you will use gatttool to control SensorTile LED.

1. Make sure that that the SensorTile is connected with BeagleBone in gatttool.

2. In gatttool, use command **char-write-cmd 001b 2000000001** to write configuration command (20000000**01**) to handle 0x001b to turn **on** the SensorTile LED. You can check if the LED on your SensorTile is turned on. See Figure 16.

```
[C0:6E:2C:31:25:48][LE]> char-write-cmd 001b 2000000001
```
Figure 16: Characteristic write to handle 0x001b with command 2000000001

3. In USB debug interface, you can see the debug information that the configuration command to turn on the SensorTile LED is composed of feature mask (F), which is 20000000, and command (C), which is 01 (1). See Figure 17.

```
Conf Sig F=20000000 C= 1
```
Figure 17: BLE_SampleApp USB debug interface for turn on LED

4. In gatttool, use command **char-write-cmd 001b 2000000000** to write configuration command (20000000**00**) to handle 0x001b to turn off the SensorTile LED. You can check if the LED on your SensorTile is turned off. See Figure 18.

```
[C0:6E:2C:31:25:48][LE]> char-write-cmd 001b 2000000000
```
Figure 18: Characteristic write to handle 0x001b with command 2000000000

5. In USB debug interface, you can see the debug information that the configuration command to turn off the SensorTile LED is composed of feature mask (F), which is 20000000, and command (C), which is 00 (0). See Figure 19.

```
Conf Sig F=20000000 C= 0
```
Figure 19: BLE_SampleApp USB debug interface for turn off LED

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

5. Save Requested Environmental Data to Text File

In this session, you will use gatttool to request environmental data and then save the data to text file for future development.

1. In gatttool, use command *disconnect* and *exit* to disconnect BeagleBone and SensorTile and exit gatttool as Figure 20.

Figure 20: Disconnect SensorTile and exit gatttool

2. Make sure that the SensorTile is successfully paired but disconnected. In BeagleBone, use command *$ gatttool -b <MAC_ADDRESS_SENSORTILE> -t random --char-write-req --handle=0x000f --value=0100 --listen* to enable the notification of environmental data on SensorTile and listen to the data transmission. See Figure 21. Use **Ctrl + C** to **terminate** the data transmission.

Figure 21: Use gatttool to request environmental data

3. In BeagleBone, use command *$ gatttool -b <MAC_ADDRESS_SENSORTILE> -t random --char-write-req --handle=0x000f --value=0100 --listen > test.txt*, wait 5 seconds, use **Ctrl + C** to terminate the gatttool, and use command *$ ls* to check if test.txt is saved. Now, you have saved the 5 seconds environmental data in test.txt file for future use. See Figure 22.

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

Figure 22: Request environmental data and save data in test.txt

4. You can use command **$ cat test.txt** to quickly check the data in test.txt. See Figure 23.

```
root@beaglebone:~# cat test.txt
Characteristic value was written successfully
Notification handle = 0x000e value: ce 89 20 87 01 00 09 01
Notification handle = 0x000e value: 0c 8a 20 87 01 00 09 01
Notification handle = 0x000e value: 4b 8a 20 87 01 00 07 01
Notification handle = 0x000e value: 89 8a 20 87 01 00 07 01
Notification handle = 0x000e value: c8 8a 1f 87 01 00 07 01
Notification handle = 0x000e value: 06 8b 20 87 01 00 07 01
Notification handle = 0x000e value: 45 8b 20 87 01 00 07 01
Notification handle = 0x000e value: 83 8b 20 87 01 00 07 01
Notification handle = 0x000e value: c2 8b 1f 87 01 00 07 01
Notification handle = 0x000e value: 00 8c 1f 87 01 00 07 01
Notification handle = 0x000e value: 3f 8c 1f 87 01 00 07 01
Notification handle = 0x000e value: 7d 8c 20 87 01 00 07 01
Notification handle = 0x000e value: bc 8c 1f 87 01 00 07 01
Notification handle = 0x000e value: fa 8c 1f 87 01 00 07 01
Notification handle = 0x000e value: 39 8d 1f 87 01 00 07 01
Notification handle = 0x000e value: 77 8d 20 87 01 00 06 01
```

Figure 23: Verify saved environmental data in test.txt

STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ Gatttool

APPENDIX I

Tutorial 8

Introduction to Motion Data Acquisition

via Bluetooth Low Energy Communication

STMicroelectronics SensorTile Tutorial: Introduction to Motion Data Acquisition via BLE Communication

Table of Contents

216

1. Introduction to This Tutorial

In *STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile*, we introduced how to use BlueZ gatttool utility to read environmental data from SensorTile Bluetooth Generic Attribute Profile.

In this tutorial, we will first introduce a new firmware, FP-SNS-ALLMEMS1, to support motion data acquisition via BLE communication.

The Tutorial steps provide:

1. An introduction to download, modify and flash advanced firmware, FP-SNS-ALLMEMS1.
2. An introduction to request motion data using gatttool
3. An introduction to save requested motion data to text file

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

For more information regarding Bluetooth, please refer to Bluetooth official website.

1.1. List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.
7) A Linux machine (BeagleBone, Intel Edison, Rpi, and etc)

1.2. Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1. STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile

2. Introduction to Firmware FP-SNS-ALLMEMS1

FP-SNS-ALLMEMS1 firmware package enables real-time motion data streaming via BLE communication. To download FP-SNS-ALLMEMS1 firmware, please refer to the following link: www.st.com/content/st_com/en/premium-content/sensortile-curriculum-fp-sns-allmems1_firmware_zip.html

Fill out the form and a download link will be sent to your email.

1. Download FP-SNS-ALLMEMS1 firmware v3.1 from the google drive link above.

Figure 1: Download FP-SNS-ALLMEMS1 Firmware

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

2. Extract *en.fp-sns-allmems1_firmware.zip* to proper directory and you should get a folder named *STM32CubeFunctionPack_ALLMEMS1_ V3.1.0*, which contains the FP-SNS-ALLMEMS1 firmware. See Figure 2.
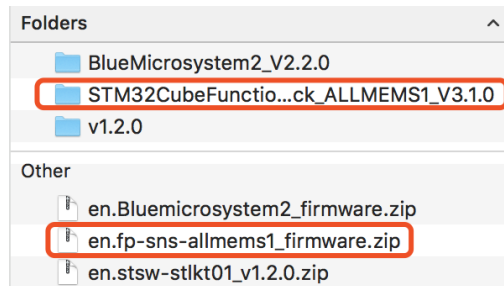
Figure 2: Extract FP-SNS-ALLMEMS1 firmware

3. Open the IDE (Eclipse or System WorkBench) on your personal computer. Once the IDE is open, *remove* your current active project in your IDE and then *import* the FP-SNS-ALLMEMS1 project by selecting **STM32CubeFunctionPack_ALLMEMS1_V3.1.0** as root directory and **unchecking** the first four projects, so only the project with 'SensorTile' in the name will be imported. See Figure 3.
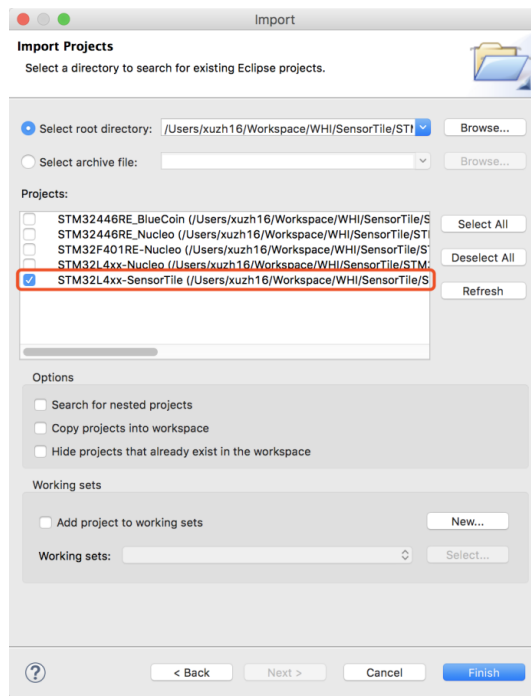
Figure 3: Import FP-SNS-ALLMEMS1 project from folder
STM32CubeFunctionPack_ALLMEMS1_V3.1.0

4. Once you successfully import the FP-SNS-ALLMEMS1 project. Open *main.c* and *main.h* according to Figure 4. Ignore the Error message in the Problems window.

220
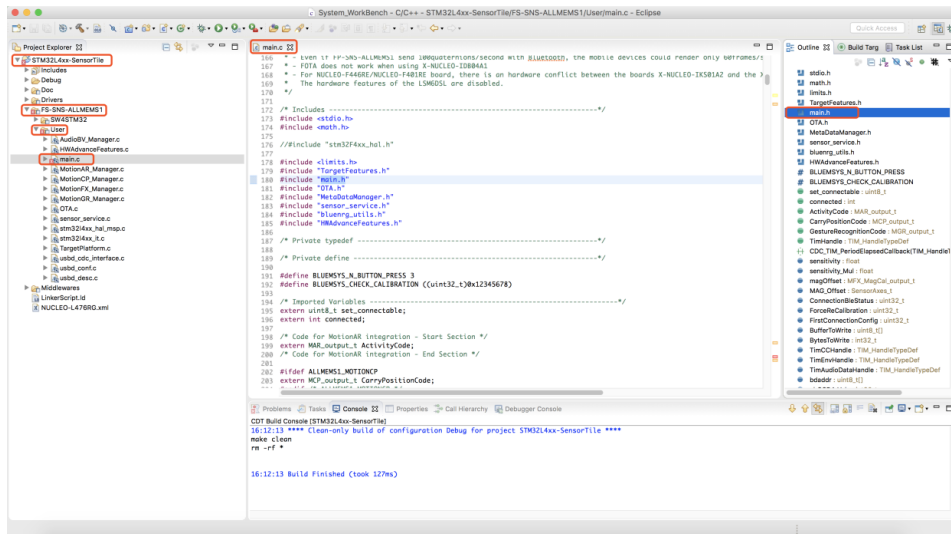
Figure 4: Open main.c and main.h in FP-SNS-ALLMEMS1 project

5. Navigate to function *SendMotionData()* at line 940 in *main.c*. Add code as highlighted in Figure 5. New added code will support motion data in USB debug interface. **Save** the mortification.

221
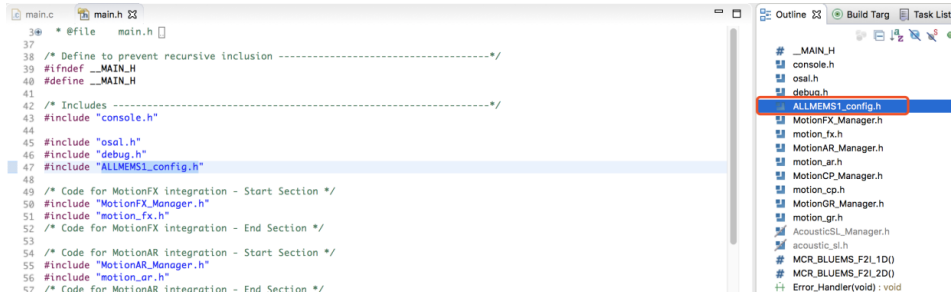
```
   main.c 🔀    main.h     ALLMEMS1_config.h

933        BSP_ACCELERO_Get_Axes(TargetBoardFeatures.HandleAccSensor,&ACC_Value);
934
935        /* Read the Magneto values */
936        BSP_MAGNETO_Get_Axes(TargetBoardFeatures.HandleMagSensor,&MAG_Value);
937
938        /* Read the Gyro values */
939        BSP_GYRO_Get_Axes(TargetBoardFeatures.HandleGyroSensor,&GYR_Value);
940
941  #ifdef ALLMEMS1_DEBUG_NOTIFY_TRAMISSION
942        int16_t acc_x_to_send, acc_y_to_send, acc_z_to_send;
943        int16_t gyr_x_to_send = 0, gyr_y_to_send = 0, gyr_z_to_send = 0;
944        int16_t mag_x_to_send, mag_y_to_send, mag_z_to_send;
945
946        acc_x_to_send = ACC_Value.AXIS_X;
947        acc_y_to_send = ACC_Value.AXIS_Y;
948        acc_z_to_send = ACC_Value.AXIS_Z;
949        ALLMEMS1_PRINTF("ACC_X=%d ",acc_x_to_send);
950        ALLMEMS1_PRINTF("ACC_Y=%d ",acc_y_to_send);
951        ALLMEMS1_PRINTF("ACC_Z=%d ",acc_z_to_send);
952
953        gyr_x_to_send = GYR_Value.AXIS_X / 100;
954        gyr_y_to_send = GYR_Value.AXIS_Y / 100;
955        gyr_z_to_send = GYR_Value.AXIS_Z / 100;
956        ALLMEMS1_PRINTF("GYR_X=%d ",gyr_x_to_send);
957        ALLMEMS1_PRINTF("GYR_Y=%d ",gyr_y_to_send);
958        ALLMEMS1_PRINTF("GYR_Z=%d ",gyr_z_to_send);
959
960        mag_x_to_send = MAG_Value.AXIS_X;
961        mag_y_to_send = MAG_Value.AXIS_Y;
962        mag_z_to_send = MAG_Value.AXIS_Z;
963        ALLMEMS1_PRINTF("MAG_X=%d ",mag_x_to_send);
964        ALLMEMS1_PRINTF("MAG_Y=%d ",mag_y_to_send);
965        ALLMEMS1_PRINTF("MAG_Z=%d ",mag_z_to_send);
966
967        ALLMEMS1_PRINTF("\r\n");
968  #endif /* ALLMEMS1_DEBUG_NOTIFY_TRAMISSION */
969
970        AccGyroMag_Update(&ACC_Value,&GYR_Value,&MAG_Value);
```

Figure 5: Modify SendMotionData() in main.c

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

6. Open *ALLMEMS1_config.h* from *main.h* according to Figure 6.
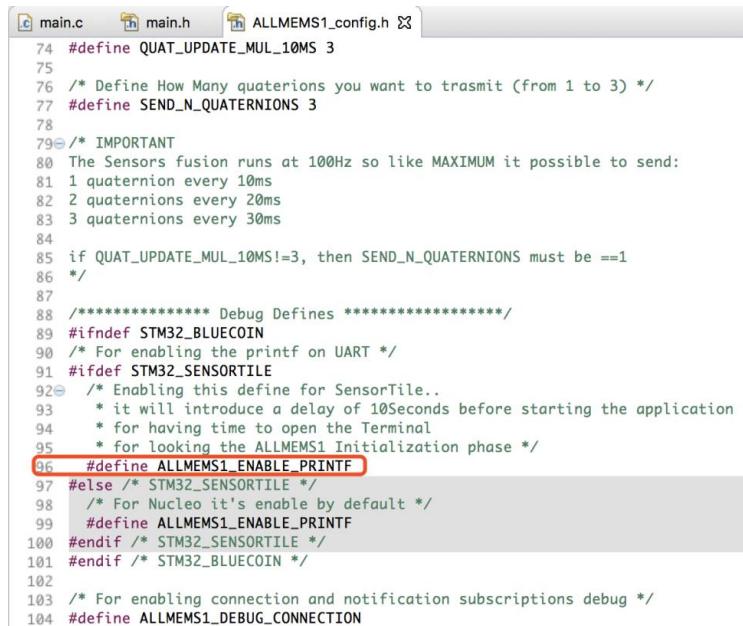
Figure 6: Open ALLMEMS1_config.h in FP-SNS-ALLMEMS1 project

7. Modify **line 96** in *ALLMEMS1_config.h* such that it matches Figure 7. This will enable the USB debug interface for the FP-SNS-ALLMEMS1 firmware.

Figure 7: Modify ALLMEMS1_config.h to enable USB debug interface for FP-SNS-ALLMEMS1.

8. **Save** the modification. **Terminate** and **remove** all previous applications from the SensorTile board. Compile and run the FP-SNS-ALLMEMS1 application on the SensorTile board in debug mode.

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

9. Examine the LED on SensorTile. Your SensorTile should start to blink once every second, which indicates that SensorTile is running FP-SNS-ALLMEMS1 firmware properly.

10. Connect your SensorTile with your computer and use terminal screen command (MAC user) or Putty serial connection (Windows user) to access the USB debug interface of FP-SNS-ALLMEMS1 firmware as we accessed the USB debug interface of BLE_SampleAPP firmware in *STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile*. The USB debug interface is indicated as Figure 8. Make sure you do not close the debug interface because you will use it in the following session.

224

Figure 8: FP-SNS-ALLMEMS1 USB debug interface

11. Connect your BeagleBone with your computer and log into BeagleBone.

12. In BeagleBone, type command *$ bluetoothctl* to use BlueZ in interactive mode.

13. As instructed in *STMicroelectronics SensorTile Tutorial: Introduction to Bluetooth Low Energy (BLE) Interfaces,* discover, pair, and **disconnect** your SensorTile device in bluetoothctl. See Figure 9. The device MAC address with name **AM1V310** is your SensorTile MAC address. You should always use this MAC address for BLE communication with SensorTile.

Figure 9: Use bluetoothctl to discover, pair, and disconnect SensorTile (AM1V310)

www.manaraa.com

14. Use command *info* in bluetoothctl to check that your SensorTile is paired but **disconnected**. See Figure 10.

Figure 10: Use bluetoothctl to check if the SensorTile is paired and disconnected

15. Copy the MAC address of SensorTile and **exit** bluetoothctl. Meanwhile, you can switch to USB debug interface to see debug information.

16. In BeagleBone, use command *$ gatttool –b <MAC_ADDRESS_SENSORTILE> –t random – I* to run gatttool in interactive mode as instructed in *STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile*. See Figure 11.

Figure 11: Use gatttool with proper options in interactive mode

17. In gatttool, use command *connect* to connect BeagleBone with SensorTile. If the connection is successful, the MAC address should turn into color of blue. See Figure 12. You can also see the successful connection information in USB debug interface.

Figure 12: Connect with SensorTile in gatttool interactive mode

18. In gatttool, 1) use command *primary* to discover all the primary services; 2) use command *characteristics* to discover all the characteristics; 3) use command *char-desc* to discover all the characteristic descriptors as instructed in *STMicroelectronics SensorTile Tutorial: BLE Communication via BlueZ and the GATT Profile*.

3. Request Motion Data Using Gatttool

In this session, you will use gatttool to request motion data.

In the previous session, you have discovered all the characteristic descriptors. Handle **0x0012** is client characteristic configuration handle to request motion data.

1. Make sure that the SensorTile is connected with BeagleBone in gatttool.

2. In gatttool, use command *char-write-req 0012 0100* to modify the characteristic value of handle 0x0012 and therefore enable the notification of motion data (handle 0x0011) on SensorTile. After you enter the command, SensorTile will start to send motion data every 50ms as indicated in the USB debug interface and BeagleBone will listen to the data transmission in gatttool. See Figure 13.

Figure 13: char-write-req to enable motion data notification in gatttool

3. In gatttool, use command *char-write-req 0012 0000* to disable the notification of motion data. See Figure 14.

Figure 14: char-write-req to disable motion data notification in gatttool

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

Page 14 of 18

228

www.manaraa.com

4. Switch to the USB debug interface. You will see similar debug information that as Figure 15. You may only see the numbers and "--->Acc/Gyro/Mag = OFF" because SensorTile sends motion data every 50 ms.


```
--->Acc/Gyro/Mag= ON
ACC_X=-27 ACC_Y=-93 ACC_Z=1018 GYR_X=0 GYR_Y=-14 GYR_Z=-4 MAG_X=-246 MAG_Y=4 MAG
_Z=-325
ACC_X=-26 ACC_Y=-95 ACC_Z=1012 GYR_X=-1 GYR_Y=-14 GYR_Z=-4 MAG_X=-247 MAG_Y=-4 M
AG_Z=-327
ACC_X=-27 ACC_Y=-96 ACC_Z=1010 GYR_X=-2 GYR_Y=-14 GYR_Z=-4 MAG_X=-244 MAG_Y=-6 M
AG_Z=-328
ACC_X=-26 ACC_Y=-97 ACC_Z=1015 GYR_X=-2 GYR_Y=-14 GYR_Z=-4 MAG_X=-241 MAG_Y=7 MA
G_Z=-328
ACC_X=-27 ACC_Y=-96 ACC_Z=1016 GYR_X=-1 GYR_Y=-14 GYR_Z=-4 MAG_X=-247 MAG_Y=3 MA
G_Z=-325
ACC_X=-27 ACC_Y=-96 ACC_Z=1014 GYR_X=-1 GYR_Y=-14 GYR_Z=-4 MAG_X=-246 MAG_Y=10 M
AG_Z=-318
ACC_X=-27 ACC_Y=-95 ACC_Z=1012 GYR_X=-1 GYR_Y=-15 GYR_Z=-4 MAG_X=-238 MAG_Y=1 MA
G_Z=-322
ACC_X=-27 ACC_Y=-96 ACC_Z=1013 GYR_X=-1 GYR_Y=-15 GYR_Z=-4 MAG_X=-247 MAG_Y=6 MA
G_Z=-322
ACC_X=-26 ACC_Y=-96 ACC_Z=1013 GYR_X=-2 GYR_Y=-15 GYR_Z=-5 MAG_X=-244 MAG_Y=0 MA
G_Z=-327
ACC_X=-28 ACC_Y=-94 ACC_Z=1014 GYR_X=0 GYR_Y=-14 GYR_Z=-4 MAG_X=-253 MAG_Y=-3 MA
G_Z=-328
ACC_X=-29 ACC_Y=-93 ACC_Z=1014 GYR_X=0 GYR_Y=-15 GYR_Z=-4 MAG_X=-246 MAG_Y=0 MAG
_Z=-319
ACC_X=-28 ACC_Y=-96 ACC_Z=1014 GYR_X=-1 GYR_Y=-14 GYR_Z=-4 MAG_X=-246 MAG_Y=7 MA
G_Z=-327
ACC_X=-27 ACC_Y=-97 ACC_Z=1015 GYR_X=-1 GYR_Y=-15 GYR_Z=-4 MAG_X=-238 MAG_Y=1 MA
G_Z=-325
ACC_X=-27 ACC_Y=-96 ACC_Z=1014 GYR_X=-1 GYR_Y=-14 GYR_Z=-4 MAG_X=-243 MAG_Y=4 MA
G_Z=-327
ACC_X=-27 ACC_Y=-99 ACC_Z=1016 GYR_X=-3 GYR_Y=-14 GYR_Z=-5 MAG_X=-243 MAG_Y=6 MA
G_Z=-324
ACC_X=-23 ACC_Y=-96 ACC_Z=1009 GYR_X=0 GYR_Y=-15 GYR_Z=-4 MAG_X=-249 MAG_Y=6 MAG
_Z=-325
--->Acc/Gyro/Mag= OFF
```

Figure 15: USB debug interface for enable/disable motion data notification

5. The motion data sent from SensorTile are parsed in similar method as environmental data in the BLE_SampleApp firmware. We can still compare Figure 13 (motion data in gatttool) and Figure 15 (motion data in USB debug interface).

In Figure 13, all the data are sent in the format of hexadecimal byte quantities. The first 2 bytes indicated in the red rectangle (e1 95) are timestamp values, where e1 is the first byte and 95 is the second byte. The next 6 bytes in the blue rectangle (e5 ff a3 ff fa 03) are the 3-axis accelerometer data. The second to last 12 bytes in the yellow rectangle (00 00 f2 ff fc ff) are 3-axis gyroscope data. The last 6 bytes in the green rectangle (0a ff 04 00 bb fe) are 3-axis magnetometer data. In each case, the X-axis sensor data value appears in the first two

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

bytes (corresponding to a 16b encoding of the signal) while the Y and Z axis values appear in the following two byte pairs, respectively.

Accelerometer data, gyroscope data and magnetometer data have similar data formats composed of the 3-axis data values for each sensor. Therefore, we will use accelerometer data as an example to interpret the motion data. For example, 3-axis accelerometer data, the first 2 bytes (e5 ff) are x-axis data. The second 2 bytes (a3 ff) are y-axis data. The last 2 bytes (fa 03) are z-axis data.

Similar to BLE_SampleAPP firmware environmental data, motion data sent by notification should be interpreted for purposes of determining signal value with the rule that the second byte is the high order byte and the first byte of the pair is the low order byte. This means that the x-axis acceleration value is computed from this data as 0xffe5, the y-axis acceleration value is 0xffa3, and the z-axis acceleration value is 0x03fa.

Environmental data are **unsigned** because all data are positive. However, motion data is encoded to represent both positive and negative values. Also, for the acceleration gain configuration here, the magnitude of acceleration is represented in milli-g. Specifically, the most positive acceleration value indicated by the accelerometer will be decimal 32767 (corresponding to the hexadecimal value of 0x7FFF and represented in two's complement binary encoding as 0111 1111 1111 1111.) The most negative acceleration value indicated by the accelerometer will be decimal -32768 (corresponding to the hexadecimal value of 0x8000 and represented in as 1000 0000 0000 0000.)

Therefore, in the example above, the x-axis acceleration decimal value is -27 (0xffe5); the y-axis acceleration decimal value is -93 (0xffa3); the z-axis acceleration decimal value is 1018 (0x03fa), which is exactly in agreement with the motion data observed from the USB debug interface value above.

The x-axis gyroscope data are 0 (0x0000); the y-axis gyroscope data are -14 (0xfff2); the z-axis gyroscope data are -4 (0xfffc).

The x-axis magnetometer data are -246 (0xff0a); the y-axis magnetometer data are 4 (0x0004); the z-axis magnetometer data are -325 (0xfebb).

The acceleration data are provided in units of **milli-g**. The gyroscope data are provided in units of **degree per second** (dps) and the magnetometer data are provided in units of **milli-Gauss**.

The accelerometer and magnetometer physical sensor signal quantities are directly indicated by their respective integer data values and physical units as above. However, the

gyroscope data signal quanties are encoded such these are equal to the corresponding data value integer multiplied by a gain factor of 10. This provides additional resolution for indicating the physical signal with the corresponding integer data value. For example, if the x-axis gyroscope data value is 155, then this represents a physical signal of 15.5 degree per second angular velocity of of the x axis.

Thus, the motion data may be computed as below:

   x-axis acceleration data: -27 milli-g
   y-axis acceleration data: -93 milli-g
   z-axis acceleration data: 1018 milli-g

   x-axis gyroscope data: 0.0 dps
   y-axis gyroscope data: -1.4 dps
   z-axis gyroscope data: -0.4 dps

   x-axis magnetometer data: -246 milli-Gauss
   y-axis magnetometer data: 4 milli-Gauss
   z-axis magnetometer data: -325 milli-Gauss

Conversion between the hexadecimal value and decimal signal for accelerometer, gyroscope, and magnetometer are accomplished in this way. First, the properly ordered hexadecimal two byte sensor values may be converted to decimal. For decimal values that are greater than 32767, correction of the offset due to two's complement encoding is performed by subtracting 65536. For decimal values less than or equal to 32767 no offset correction is applied.

231

4. Save Requested Motion Data to Text File

In this session, you will use gatttool to request motion data and then save the data to text file for future development.

1. In gatttool, use command ***disconnect*** and ***exit*** to disconnect BeagleBone and SensorTile and exit gatttool as Figure 16.

```
[C0:83:2C:31:25:48][LE]> disconnect
[C0:83:2C:31:25:48][LE]> exit
root@beaglebone:~#
```

Figure 16: Disconnect SensorTile and exit gatttool

2. Make sure that the SensorTile is successfully paired but disconnected. In BeagleBone, use command ***$ gatttool -b <MAC_ADDRESS_SENSORTILE> -t random --char-write-req -- handle=0x0012 --value=0100 --listen > motion_data.txt***, wait 5 seconds, use **Ctrl + C** to terminate the gatttool. See Figure 17.

```
root@beaglebone:~# gatttool -b C0:83:2C:31:25:48 -t random --char-write-req --ha
ndle=0x0012 --value=0100 --listen > motion_data.txt
^C
root@beaglebone:~#
```

Figure 17: Request motion data and save data in motion_data.txt

3. Use command ***$ ls*** to check if *motion_data.txt* is saved. Now, you have saved the 5 seconds motion data in motion_data.txt file for future use.

4. You can use command ***$ cat motion_data.txt*** to quickly check the data in *motion_data*.txt.

STMicroelectronics SensorTile Tutorial: Motion Data Acquisition via BLE Communication

APPENDIX J

Tutorial 9

Introduction to Inertial Sensing

STMicroelectronics SensorTile Tutorial: Introduction to Inertial Sensing

Table of Contents

1. Introduction to This Tutorial

We have learned inertial motion sensing, finite state machine based posture recognition, and digital signal processing in the previous tutorials. In this tutorial, we will further update DataLog application and estimate the SensorTile x-axis displacement using SensorTile inertial accelerometer. In addition, we will have a Python animation system to visualize the acceleration and displacement in real time.

The Tutorial steps provide:

1. An introduction to installation of Displacement Estimation project and Python animation system.
   o Displacement estimation project installation
   o Python animation system installation
2. An introduction to Displacement Estimation by Inertial Sensor system.
   o Acceleration processing
   o Displacement processing

For more information regarding the SensorTile board, please open the following link.
www.st.com/sensortile

List of Required Equipment and Materials

1) 1x STMicroelectronics SensorTile kit.
2) 1x STMicroelectronics Nucleo Board.
3) 1x Personal Computer with two USB type-A inputs OR you must have a powered USB hub.
4) 1x USB 2.0 A-Male to Micro-B Cable (micro USB cable).
5) 1x USB 2.0 A-Male to Mini-B Cable (mini USB cable).
6) Network access to the Internet.
7) 1x Small piece of foam pad

Prerequisite Tutorials

It is recommended that users have completed and are familiar with the contents of the following tutorials before proceeding.

1) Tutorials up to tutorial 4.

Your instructor will ensure you have the required background regarding motion sensing, digital signal processing, and basic physics of acceleration, velocity, and displacement.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Page 3 of 26

236

2. Displacement Estimation System Setup

In the previous tutorials, we explored the DataLog application and AudioLoop application in the SensorTile starter firmware package. In this project, we further update DagtaLop application to estimate displacement using acceleration in DataLog application. You can visualize the x-axis acceleration and displacement by Python animation system through USB data streaming.

2.1 SensorTile Displacement Estimation System Installation

1. **Download the project from google drive** (https://drive.google.com/open?id=1msaomIBQ2QJ7QXi3mmlH6nnCPaHCGwKs) and decompress it to your working directory.
   Note: Please download this project above. Please do not use the Datalog project you had used in Tutorials 1 – 3. This project has been modified as will be described to meet the requirements for Inertial Sensing.

2. Open the IDE, System WorkBench, on your personal computer as instructed in the Tutorial 1**.** Select the same workspace as in Tutorial 1.

3. Once System WorkBench is open, first remove your current project in System WorkBench and import the DataLog project from corresponding directories as instructed in the document labeled **STMicroelectronics SensorTile Tutorial: Introduction to STMicroelectronics Development Environment and DataLog Project Example**. Uncheck AudioLoop and BLE projects (first two projects) to properly import the DataLog project. See Figure 1.
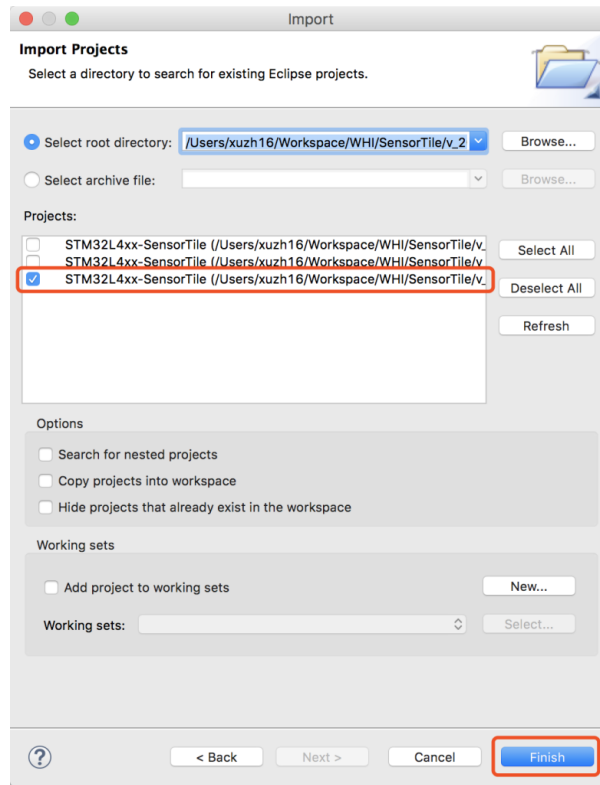
Figure 1: Import DataLog project from modified start firmware package.

4. Once you successfully import the DataLog project, you can build the project. Connect the SensorTile and Neuclo Board with your computer. You can then either run the project in **Debug** mode or utilize *st-flash tool* on Mac or *ST-LINK utilities tool* on Windows introduced in Tutorial 1 to flash the latest firmware to SensorTile.

2.2 Python Animation System Installation

Python is a programming language that lets you work quickly and integrate systems more effectively. Python is also the most popular programming platform for machine learning. For more information, please refer to Python official website (https://www.python.org/).

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

In this section, we will introduce Python installation, Python package installation, and Python animation system setup for both Mac OS and Windows.

Download the Python animation system from google drive

https://drive.google.com/open?id=1Y6ttQigFdCwF0mP2VoTRmEfCVXveUR6c

and unzip it to your working directory. You should find that the animation system is composed of **SensorTile_Animation_args.py** and **SensorTile_Serial.py.**

You can follow the instructions for Mac OS or Windows according to your computer operating system. If you have any concerns about the installation, please ask your course mentors or your teaching assistant, or instructor for assistance.

Mac OS Installation and Operation

2.2.1 Python Installation

You should have Python installed by default in your Mac. Issue command "$ *python*" in a Terminal session to check if Python is installed. This SensorTile Animation system is compatible with both Python 2 and Python 3. You only need one version of Python to properly use the system.

If you have not installed Python in your Mac, use the link (https://www.python.org/downloads/) to download Python 3.7.2 and install accordingly.

2.2.2 Run Python

Open terminal and issue command "$ **python**". If you have multiple Python version installed, make sure you run the correct command to use the corresponding Python. For example, you could issue "$ **python**" to start Python 2 and issue "$ **python3**" to start Python 3. See Figure 2.

```
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ python
Python 2.7.14 |Anaconda custom (64-bit)| (default, Oct  5 2017, 02:28:52)
[GCC 4.2.1 Compatible Clang 4.0.1 (tags/RELEASE_401/final)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ python3
Python 3.7.2 (default, Jan 13 2019, 12:50:15)
[Clang 10.0.0 (clang-1000.11.45.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 2: Python 2 and Python 3. Either version is fine.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

You can use command "*exit()*" to exit Python.

2.2.3 Install Required Python Packages

You can find and install software developed and shared by the Python community through PyPl (pip), where PyPl is embedded with Python. You need to install Python packages to operate the SensorTile Python Animation system.

Upgrade pip

In terminal, first issue command "$ *python -m pip install --upgrade pip*". See Figure 3.

If an error message appears of "No module pip" then issue the command
"$ **sudo easy_install pip**". Then, proceed with "$ *python -m pip install --upgrade pip*" as in Figure 3.

```
[Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ python -m pip insta]
ll --upgrade pip
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Ple
ase upgrade your Python as Python 2.7 won't be maintained after that date. A fut
ure version of pip will drop support for Python 2.7.
Requirement already up-to-date: pip in /anaconda2/lib/python2.7/site-packages (1
9.0.3)
```

Figure 3: Upgrade pip

Install PySerial

In terminal, issue command "$ **sudo** *pip install pyserial==3.4*" to install PySerial. See Figure 4.

```
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ pip install pyseria
l==3.4
```

Figure 4: Install PySerial

Install matplotlib

In terminal, issue command "$ **sudo** *pip install matplotlib==3.0.2*" to install matplotlib. See Figure 5.

```
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ pip install matplot
lib==3.0.2
```

Figure 5: Install matplotlib

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Verify packages are installed

In Python, issue command according to Figure 6.

```
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ python3
Python 3.7.2 (default, Jan 13 2019, 12:50:15)
[Clang 10.0.0 (clang-1000.11.45.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
[>>> import serial
[>>> serial.__version__
'3.4'
[>>> import matplotlib
[>>> matplotlib.__version__
'3.0.2'
>>> 
```

Figure 6: Verify packages installed

2.2.4 Execute the SensorTile Python Animation System

Find the serial port address of the SensorTile

Referring to Tutorial 1, you can now find the serial port address of SensorTile that you have used with Screen to visualize the USB data streaming.

Operate the SensorTile Python Animation System

Navigate into the Python animation system directory and issue the command
$ python SensorTile_Animation_args.py {SerialAddress}

to start the system. SerialAddress should be serial address found in previous step. You shoud notice that the device ending in "1" is the SensorTile and device ending in "2" is the Nucleo board while checking the serial address.

On a typical Mac, the serial port device file address is "/dev/cu.usbmodem1461421". Therefore, an example of command to initialize the Python animation system on this machine is

$ python SensorTile_Animation_args.py /dev/cu.usbmodem1461421

as shown in Figure 7.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

```
Xus-MacBook-Pro:SensorTile_Inertial_Sensing_Tutorial xuzh16$ python SensorTile_A
nimation_args.py /dev/cu.usbmodem1461421
```

Figure 7: Start Python animation system

2.2.5 Python Animation System

The Python Animation System is designed to plot x-axis displacement and acceleration data in real time in Python.

The animation system window will will pop up after issuing the command. The animation system is shown in Figure 8 below. There will be an expected delay of data plotting during about the first 5 – 10 seconds of the animation caused by buffering of data at the serial port of the Mac. Later on, the plotting should perform well in real time. However, a constant delay may be observed for some machines. However, this does not limit operation or usability of the SensorTile Animation System.
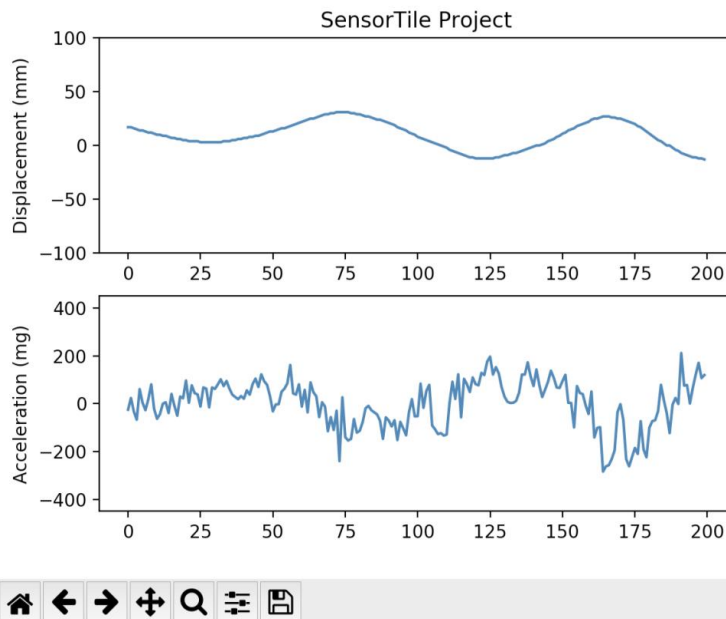
Figure 8: Python Animation System of x-axis displacement and acceleration

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Page 9 of 26

242

You may move SensorTile along its x-axis to visualize the acceleration and displacement.

2.2.6 Shutdown the Python Animation System

It is very important to close the animation system properly, otherwise the system will fail to close the serial connection, which will eventually lead to serial buffer overflow and system failure.

In order to properly shutdown the animation system, you need to click the red "x" in the left-top corner shown in Figure 9.
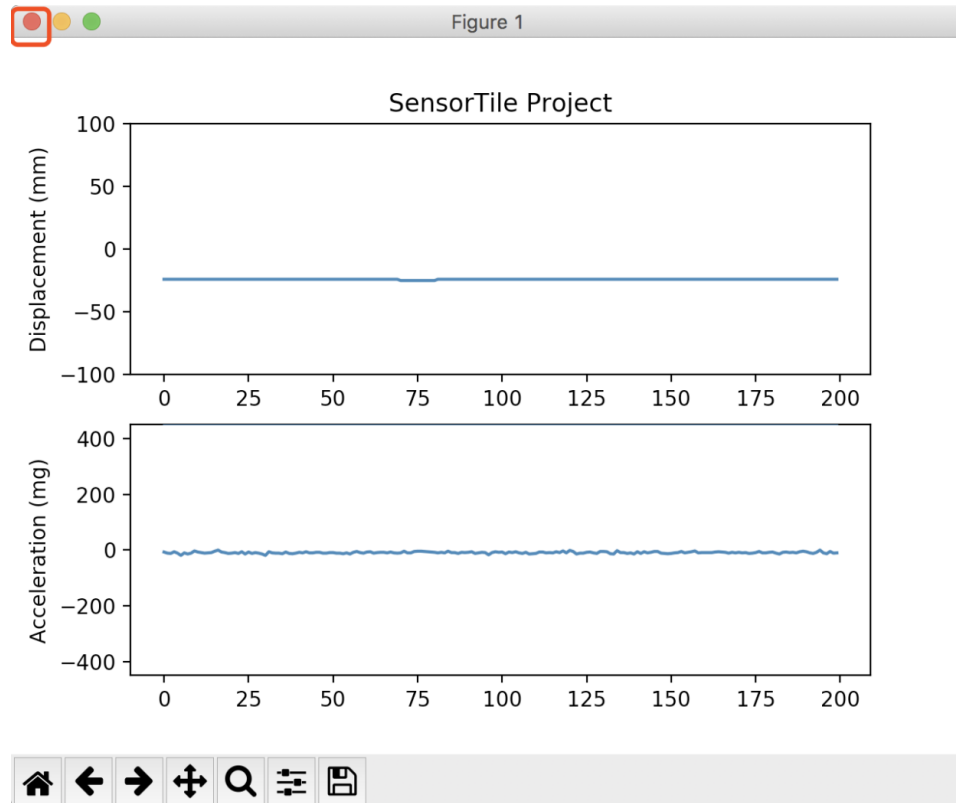
Figure 9: Shutdown the Python Animation System

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

If the system is closed properly, you are able to see "Close serial connection" message in terminal referring to Figure 10.

Figure 10: Properly shutdown the Python animation system

Windows System Installation and Operation

2.2.1 Python Installation

This step is required for students who have not installed Python before. If you have any versions of Python (the default Python distribution from python.org, Anaconda Python, PyCharm IDE, or others) installed in your system, make sure that you install all the packages required in Python Package Installation section. The system is compatible for both Python 2 and Python 3. You only need one version of Python to properly use the system.

If you have not installed Python before in Windows, please refer to the link (https://www.python.org/downloads/) to download latest Python 3.7.2. See Figure 11.
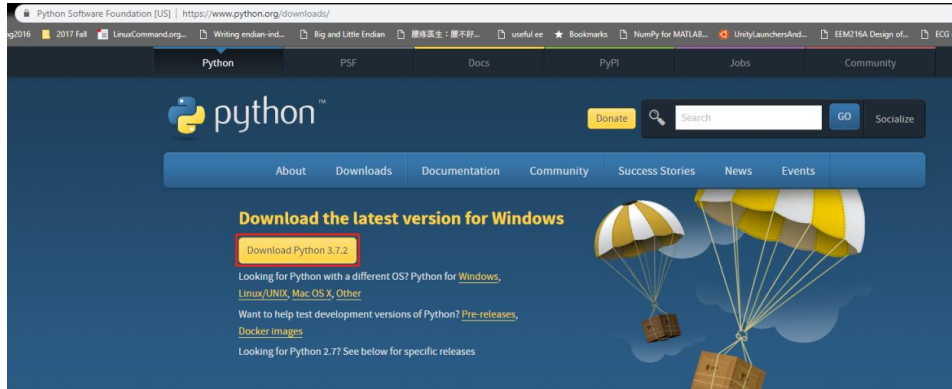
Figure 11: Download Python 3.7.2 for Windows.

Find the downloaded "**python-3.7.2.exe**" link for your machine and double click it to start the installation. Check "**Add Python 3.7 to PATH**". **This step is very important to ensure that users may directly run Python at the Windows command prompt.** See Figure 12 to install Python accordingly.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing
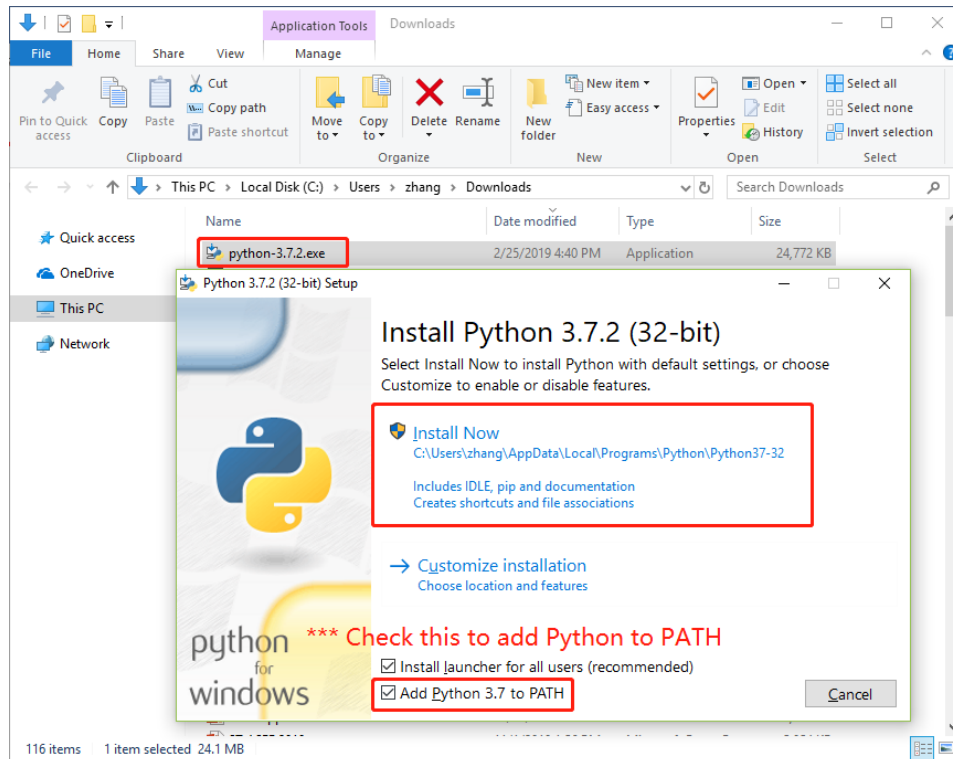
Figure 12: Python Installation

2.2.2 Issue Command Prompt to Check Python Installation

Enter "command" in Windows searching bar and open **Command Prompt** as shown in Figure 13.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

245
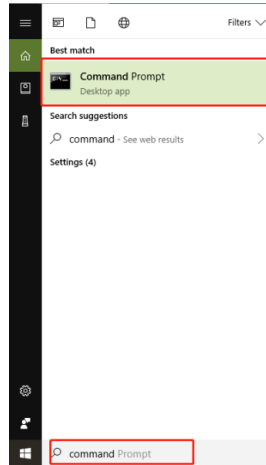
Figure 13: Open Command Prompt in Windows.

Issue "*python*" in the Command Prompt and check the Python version. See Figure 14.
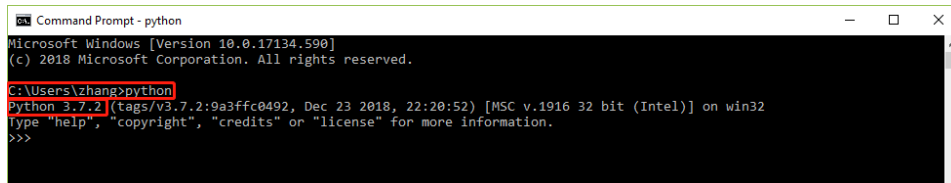
Figure 14: Python in Command Prompt.

Now, you have successfully installed Python 3.7.2. You can use command "*exit()*" to exit Python.

2.2.3 Install Required Python Packages

You can find and install software developed and shared by the Python community through PyPl (pip), where PyPl is embedded with Python. You need to install Python packages to operate the Python animation system.

Upgrade pip

Issue command "*python -m pip install --upgrade pip*" in Windows Command Prompt. See Figure 15.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Figure 15: Upgrade pip.

Install PySerial

Issue command "*pip install pyserial==3.4*" in Windows Command Prompt to install pyserial package. See Figure 16.

Figure 16: Install PySerial

Install matplotlib

Issue command "*pip install matplotlib==3.0.2*" in Windows Command Prompt to install matplotlib package. See Figure 17.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Figure 17: Install matplotlib

Verify package installed

Follow instruction as shown in Figure 18 to verify if Python packages are successfully installed.

Figure 18: Verify packages installed

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

2.2.4 Run the SensorTile Python Animation System

Find the directory of Python animation system

Right click on the Python file and check the properties. Find the Location of the Python animation system as shown in Figure 19.
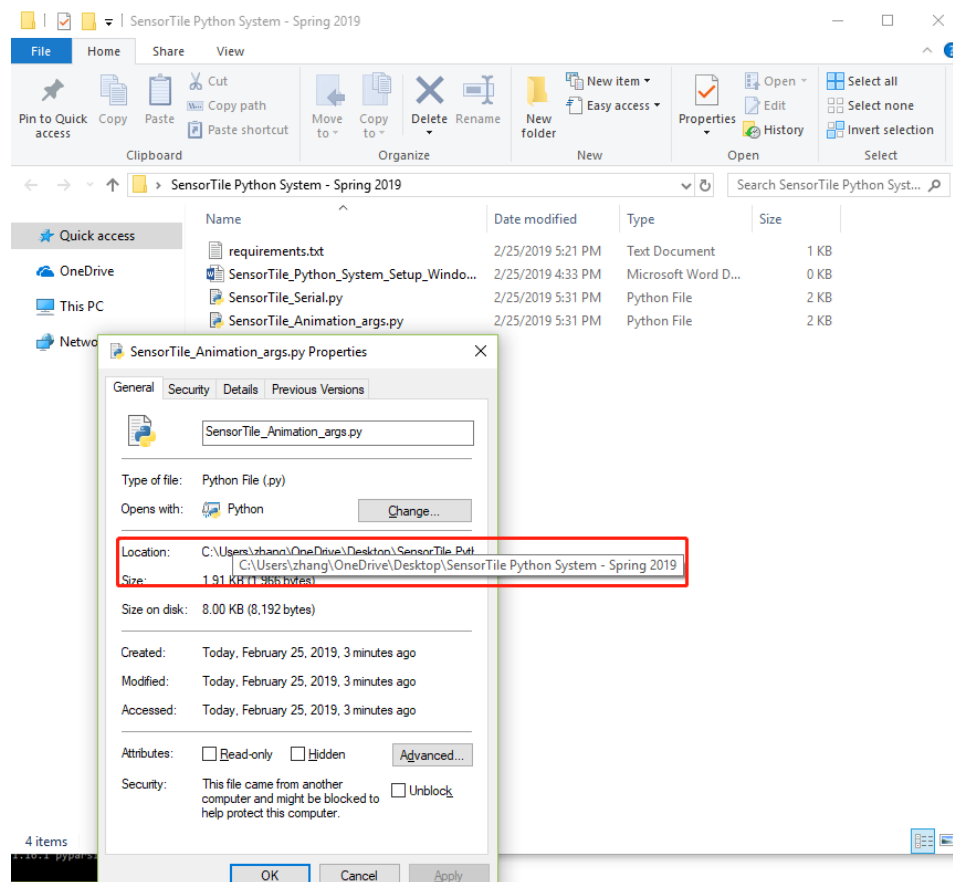

Figure 19: Directory Location of Python System

Navigate into the system directory and operate the system

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Navigate into the animation system directory in Windows Command Prompts as shown in Figure 20.


```
C:\Users\zhang>cd OneDrive

C:\Users\zhang\OneDrive>cd Desktop

C:\Users\zhang\OneDrive\Desktop>cd "SensorTile Python System - Spring 2019"

C:\Users\zhang\OneDrive\Desktop\SensorTile Python System - Spring 2019>dir
 Volume in drive C has no label.
 Volume Serial Number is 2EDC-3075                  Equivalent to ls

 Directory of C:\Users\zhang\OneDrive\Desktop\SensorTile Python System - Spring 2019

02/25/2019  05:31 PM    <DIR>          .
02/25/2019  05:31 PM    <DIR>          ..
02/25/2019  05:21 PM                34 requirements.txt
02/25/2019  05:31 PM             1,966 SensorTile Animation args.py
02/25/2019  04:33 PM                 0 SensorTile_Python_System_Setup_Windows.docx
02/25/2019  05:31 PM             1,564 SensorTile_Serial.py
               4 File(s)          3,564 bytes
               2 Dir(s)  60,351,696,896 bytes free

C:\Users\zhang\OneDrive\Desktop\SensorTile Python System - Spring 2019>
```

Figure 20: Navigate into the animation system directory in Windows Command Prompts

Find the serial port address of the SensorTile

You need to find the serial port in Device Manager in Windows system according to Tutorial 1. For example, on a typical machine, the SensorTile serial port is "COM3" as shown in Figure 21.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing
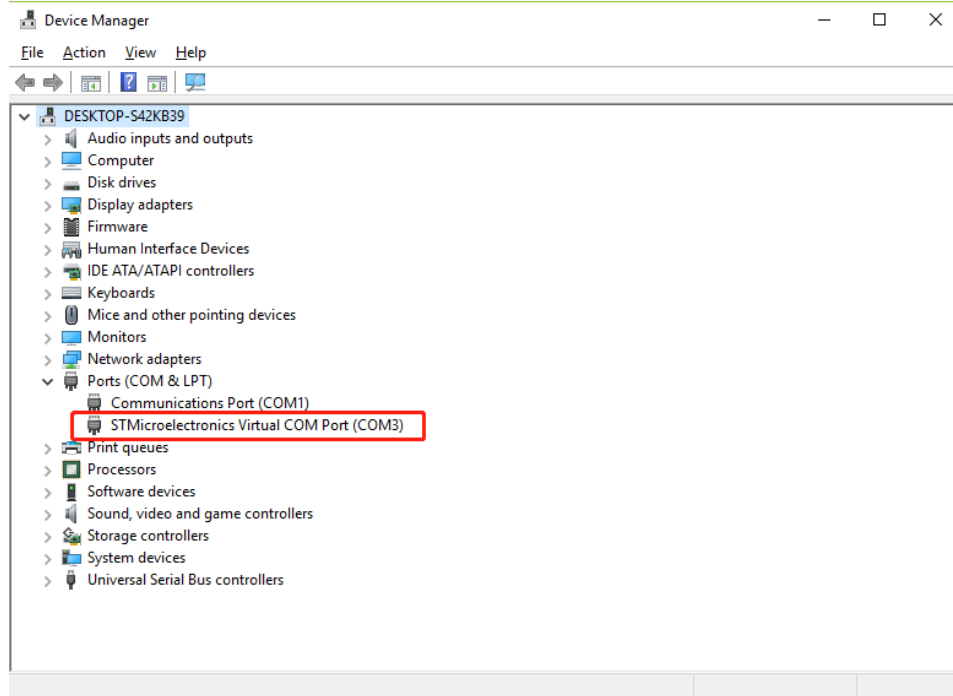
Figure 21: COM port of SensorTile in Device Manager

Operate the Python Animation System

Make sure you have navigated into the Python animation system directory. Then you may issue the command "**python SensorTile_Animation_args.py serialAddress**". For example, the serial port is "COM3" and therefore I should issue "**python SensorTile_Animation_args.py COM3**" to start the Python animation system shown in Figure 22.

```
C:\Users\zhang\OneDrive\Desktop\SensorTile Python System - Spring 2019>python SensorTile_Animation_args.py COM3
```

Figure 22: Command to start Python animation system

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing
Page 18 of 26

2.2.5 Python Animation System

The Python Animation System is designed to plot x-axis displacement and acceleration data in real time in Python.

The animation system will pop up after issuing the command and you should observe message "Start Serial Connection" in the command prompt. The animation system is shown in Figure 23 below. There will be an expected delay of data plotting during about the first 5 – 10 seconds of the animation caused by serial buffer issues. Later on, the plotting will perform in near real time. However, a constant delay is also normal for some machines. However, this does not limit the usability of the SensorTile Animation system.
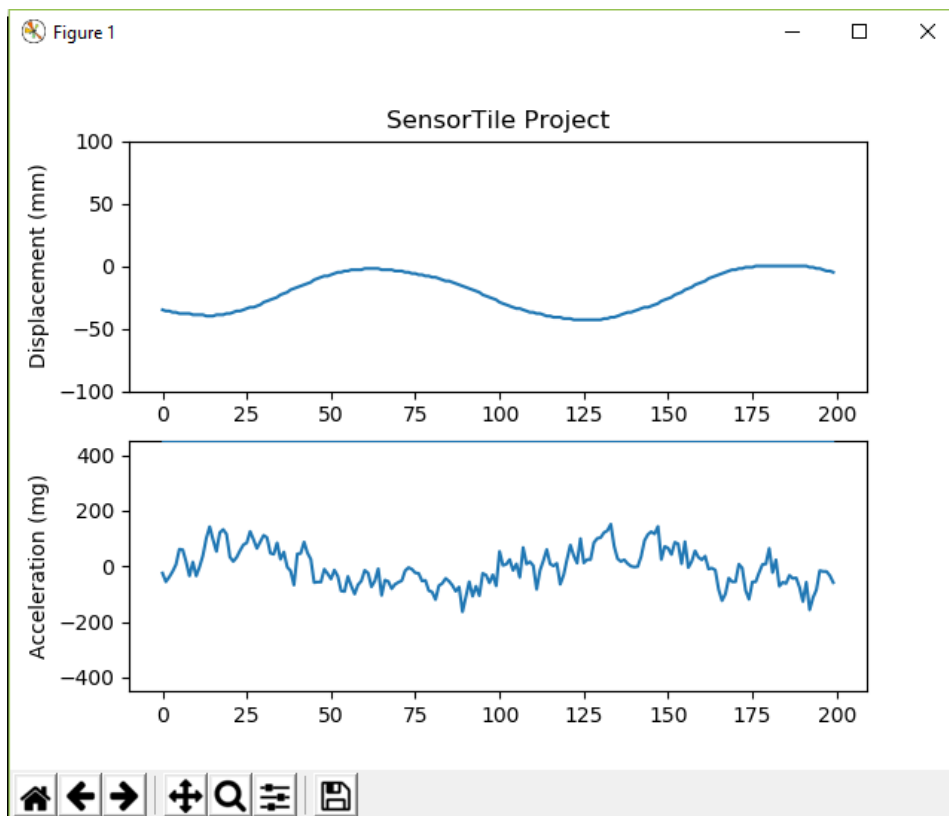
Figure 23: Python animation system of x-axis displacement and acceleration

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

You may move SensorTile along x-axis to visualize the acceleration and displacement.

2.2.6 Shutdown the Python Animation System

It is very important to close the animation system properly, otherwise the system will fail to close the serial connection, which will eventually lead to serial buffer overflow and system failure.

In order to properly shutdown the animation system, you need to click the red "x" in the right-top corner shown in Figure 24.
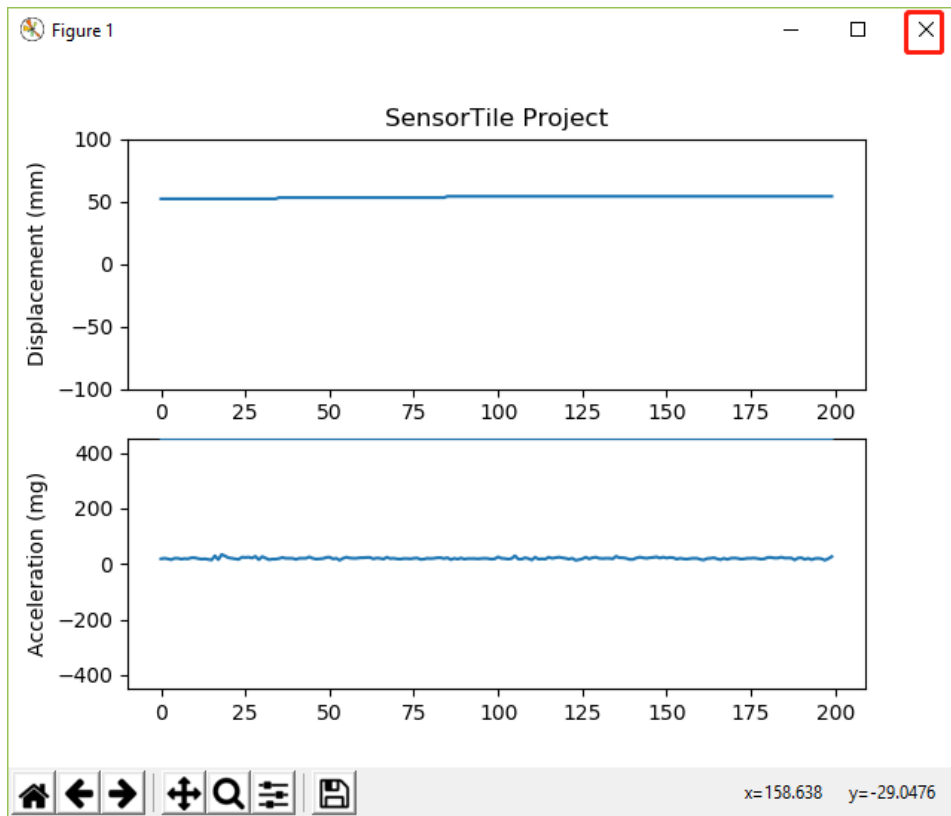

Figure 24: Shutdown the Python Animation System

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

If the system is closed properly, you are able to see "Close serial connection" message in command prompt referring to Figure 25.


```
['54', '22']
['54', '19']
['54', '18']
['54', '22']
Close Serial Connection

C:\Users\zhang\OneDrive\Desktop\SensorTile Python System - Spring 2019>_
```

Figure 25: Properly shutdown the Python animation system

3. The Displacement Estimation System

3.1 Displacement Estimation: Trapezoidal integral

Fundamental physics describes the familiar relationship between acceleration, velocity, and displacement. Velocity is determined by performing an integral over time of acceleration. Displacement is determined by performing an integral over time of velocity.

$$v(t) = \int_0^t a(\tau)\, d\tau + v(t = 0)$$
$$d(t) = \int_0^t v(\tau)\, d\tau + d(t = 0)$$

Since the SensorTile acceleration data source is a discrete time, sampled signal, where acceleration is sampled at 100 Hz, a discrete trapezoidal integration is computed on velocity and displacement using the relationships below. Δt in this case equals the sampling period of 1/(100 Hz) = 10milli-seconds.

$$v(1) = v(0) + \left[a(0) + \frac{a(1) - a(0)}{2}\right] * \Delta t = v(0) + \frac{a(1) + a(0)}{2} * \Delta t$$
$$d(1) = d(0) + \left[v(0) + \frac{v(1) - v(0)}{2}\right] * \Delta t = d(0) + \frac{v(1) + v(0)}{2} * \Delta t$$

where a(1) is the current acceleration sample, a(0) is the previous acceleration sample, v(1) is current velocity computation, v(0) is the previous velocity computation, d(1) is the current displacement computation, d(0) is the previous displacement computation, and Δt is 0.01s. In addition, a(x) is the acceleration signal based on acceleration sampled by the SensorTIle system.

In the section, we will introduce 1) Sensor sampling rate, 2) Acceleration sensor signal processing, and 3) Computation and signal processing of displacement data.

3.2 System Sampling Frequency and Sensor Handler Setup

The Datalog application operates with a sensor signal sampling period of 100 milli-seconds, or 10 Hz. In order to provide adequate resolution for displacement computation, In the displacement estimation system, the data acquisition period is reduced to 10 ms in **main.c** as shown in Figure 26. Therefore, the system operates with sampling frequency 100 Hz.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

```
  main.c  X
  52  /* Private typedef --------------------------------------------------*/
  53
  54  /* Private define ----------------------------------------------------*/
  55
  56  /* Data acquisition period [ms] */        Sampling period: 10ms
  57  #define DATA_PERIOD_MS (10)
```

Figure 26: Displacement estimation system data acquisition period

In addition, this application enables only the **Accelero_Sensor_Handler()** in **main.c** as shown in Figure 27. You will be able to enable rate gyroscope signals next.

```
  main.c  X
  168    while (1)
  169    {
  170      /* Get sysTick value and check if it's time to execute the task */
  171      msTick = HAL_GetTick();
  172      if(msTick % DATA_PERIOD_MS == 0 && msTickPrev != msTick)
  173      {
  174        msTickPrev = msTick;
  175        if(SendOverUSB)
  176        {
  177          BSP_LED_On(LED1);
  178        }
  179 #ifdef NOT_DEBUGGING
  180        else if (SD_Log_Enabled)
  181        {
  182          BSP_LED_On(LEDSWD);
  183        }
  184 #endif
  185        RTC_Handler( &RtcHandle );
  186
  187        Accelero_Sensor_Handler( LSM6DSM_X_0_handle );     Disable other sensor
  188        /*                                                  handlers that are not
  189        Gyro_Sensor_Handler( LSM6DSM_G_0_handle );         needed
  190
  191        Magneto_Sensor_Handler( LSM303AGR_M_0_handle );
  192
  193        Pressure_Sensor_Handler( LPS22HB_P_0_handle );
  194
  195        if(!no_T_HTS221)
  196        {
  197          Temperature_Sensor_Handler( HTS221_T_0_handle );
  198        }
  199        if(!no_H_HTS221)
  200        {
  201          Humidity_Sensor_Handler( HTS221_H_0_handle );
  202        }
```

Figure 27: Displacement estimation system sensor handler management

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

Page 23 of 26

256

www.manaraa.com

3.3 Digital Signal Processing Discrete Time Filters

Two different filters in the system: 1) An anti-aliasing filter to avoid aliasing during signal acquisition, and 2) High pass filtering to eliminate drift in displacement sensing. The filter coefficients are calculated as shown in Figure 28.

```c
232
233    /*
234     * Low Pass Filter coefficients for Anti-Aliasing filter
235     */
236
237    float fo_AA = 5;
238    float Wo_AA, IWon_AA, iir_0_AA, iir_1_AA, iir_2_AA;
239
240    Wo_AA = 2 * 3.141592654 * fo_AA;
241    IWon_AA = 2 / (Wo_AA * Tsample);
242    iir_0_AA = 1 / (1 + IWon_AA);
243    iir_1_AA = iir_0_AA;
244    iir_2_AA = iir_0_AA * (1 - IWon_AA);
245
246    /*
247     * High Pass Filter coefficients for filter operating on velocity and displacement
248     */
249
250    float fo_h = 0.3;
251    float Wo_h, IWon_h, iirh_0, iirh_1, iirh_2;
252
253    Wo_h = 2 * 3.141592654 * fo_h;
254    IWon_h = 2 / (Wo_h * Tsample);
255    iirh_0 = 1 - 1/(1 + IWon_h);
256    iirh_1 = -iirh_0;
257    iirh_2 = (1/(1+IWon_h))*(1-IWon_h);
258
```

Figure 28: Filter coefficients calculation

Anti-aliasing filter is applied to the sampled acceleration (red box) and the high pass filter is applied to calculated velocity and displacement (green box) as shown in Figure 29.

257

www.manaraa.com

Figure 29: Filtering and displacement estimation by integration.

3.4 Velocity and displacement by Integration

According to the trapezoidal integral formula introduced above, we can estimate the velocity and displacement (blue box) accordingly as shown in Figure 29 above.

3.5 Data Streaming

Then, the integer part of the x-axis displacement and acceleration is transmitted through the USB serial port as shown in Figure 30. It is important to note that the 9600 Baud rate serial transmission limits the data payload that may be transmitted during each 10 milli-second period. Thus, only the integer parts of the signals are transmitted to reduce data rate below the maximum available.

STMicroelectronics SensorTile Tutorial: Displacement Estimation by Inertial Sensing

```
  c  datalog_application.c  ⊠
313
314⊝              /*
315               * Assignment of integer values for output (only integer part of value is supplied due to
316               * communication rate limits)
317               *
318               */
319
320              floatToInt((accel_x_direct), &d1_ax, &d2_ax, 4);
321              floatToInt(accel_x_direct_filter, &d1_x, &d2_x, 4);
322              floatToInt(velocity_filter, &d1_v, &d2_v, 4);
323              floatToInt(displacement_filter, &d1_df, &d2_df, 4);
324
325⊝              /*
326               * Data transmission
327               * Note that during each 10ms period, less than 10 characters may be transmitted at the rate of 9600
328               * baud and at 10 bits per character.  Thus, communication payload must be reduced to at most
329               * two integers.
330               *
331               */
332
333              sprintf( dataOut, "%i\t%i\r\n",
334                      (int)d1_df, (int)d1_ax);
335              CDC_Fill_Buffer(( uint8_t * )dataOut, strlen( dataOut ));
336
```

Figure 30: Displacement and acceleration data streaming.

259

www.manaraa.com

APPENDIX K

Reference Designs

SensorTile Reference Design

https://drive.google.com/open?id=1pIEeOH5m76unlHzc4IcOvx2Zm1pCkhAy

IoT Machine Learning Reference Design

https://drive.google.com/open?id=1_x-St2bkU8pEozk_O36HhV0iDnYEBZW0

Student Reference Design (1)

https://drive.google.com/open?id=1YtGW5OVgw_TQLhRWc0EZrpQroNsPONAb

Student Reference Design (2)

https://drive.google.com/open?id=1s1stCDo8bU3-r7bt5RyhklK3iKlhsTOi

Student Reference Design (3)

https://drive.google.com/open?id=1DzFee-GwmhqSnetLpt1uVbp-UaNlVKEn

Student Reference Design (4)

https://drive.google.com/open?id=1BOEk4zwXd3S4bDC9WduEcDqfpA90bJ3I

Student Reference Design (5)

https://drive.google.com/open?id=1k5p7aWy24BSce-xHWiFUoTQx_XaF8AIL

Student Reference Design (6)

https://drive.google.com/open?id=1FZMlvfvjH45BuM2gm4ONi3vC6d5zfDVS

APPENDIX L

IoT Curriculum Website

STMicroelectronics SensorTile Internet of Things Curriculum

https://sites.google.com/view/ucla-stmicroelectronics-iot/home

STMicroelectronics SensorTile and BeagleBone Internet of Things and Machine Learning

https://sites.google.com/view/ucla-stmicroelectronics-iot-ml/home

261

REFERENCES

[1]   STMicroelectronics. SensorTile development kit. Available at
        https://www.st.com/en/evaluation-tools/steval-stlkt01v1.html

[2]   STMicroelectronics. User manual of SensorTile development kit. Available at
        https://www.st.com/content/ccc/resource/technical/document/user_manual/group0/bc/b1/
        ad/c8/36/de/40/92/DM00320099/files/DM00320099.pdf/jcr:content/translations/en.DM0
        0320099.pdf

[3]   Open STM32 Community. The STM32 system resources. Available at
        http://www.openstm32.org/HomePage

[4]   STMicroelectronics. STSW-STLKT01 embedded software samples for SensorTile.
        Available at https://www.st.com/content/st_com/en/products/embedded-
        software/evaluation-tool-software/stsw-stlkt01.html

[5]   STMicroelectronics. FP-SNS-ALLMEMS1 STM32 ODE function pack for IoT node.
        Available at https://www.st.com/content/st_com/en/products/embedded-software/mcu-
        mpu-embedded-software/stm32-embedded-software/stm32-ode-function-pack-sw/fp-sns-
        allmems1.html

[6]   STMicroelectronics. STM32 ST-LINK utility. Available at
        https://www.st.com/en/development-tools/stsw-link004.html

[7]   PuTTY: a free SSH and Telnet client. Available at
        https://www.chiark.greenend.org.uk/~sgtatham/putty/

[8]   Github user texane. Open source version of the STMicroelectronics Stlink Tools.
        Available at https://github.com/texane/stlink

[9]   Apple. Terminal User Guidance. Available at
        https://support.apple.com/guide/terminal/welcome/mac

[10]   Rabiner, Lawrence R and Schafer, Ronald W., 1938- *Theory and applications of digital
        speech processing* (1st ed). Pearson, Upper Saddle River, 2011.

[11]   Audacity. Available at https://www.audacityteam.org/

[12]   Park, Sang Kyeong, and Young Soo Suh. "A zero velocity detection algorithm using
        inertial sensors for pedestrian navigation systems." *Sensors (Basel, Switzerland)* vol.
        10,10 (2010): 9163-78. doi:10.3390/s101009163

[13]   BlueZ. Official Linux Bluetooth protocol stack. Available at bluez.org

262

[14] Bluetooth. Bluetooth Generic Attribute (GATT). Available at
      https://www.bluetooth.com/specifications/gatt/

[15] STMicroelectronics. FP-SNS-ALLMEMS1, STM32 ODE Function Pack SW. Available
      at https://www.st.com/en/embedded-software/fp-sns-allmems1.html

[16] Nissen, S. (2003). Implementation of a Fast Artificial Neural Network Library
      (fann). Report, Department of Computer Science University of Copenhagen (DIKU), 31.

263