Getting started with the IOTA Distributed Ledger Technology software expansion for STM32Cube

Introduction

The X-CUBE-IOTA1 expansion software package for STM32Cube runs on the STM32 and includes middleware to enable the IOTA Distributed Ledger Technology (DLT) functions.

The IOTA DLT is a transaction settlement and data transfer layer for the Internet of Things (IoT). IOTA allows people and machines to transfer money and/or data without any transaction fees in a trustless, permissionless and decentralized environment. This technology even makes micro-payments possible without the need of a trusted intermediary of any kind.

The expansion is built on STM32Cube software technology to ease portability across different STM32microcontrollers.

The current version of the software runs on the B-L4S5I-IOT01A discovery kit for IoT node and connects to the Internet through the attached Wi-Fi interface.

RELATED LINKS

Visit the STM32Cube ecosystem web page on www.st.com for further information

https://www.iota.org/get-started/what-is-iota

https://docs.iota.org/docs/getting-started/1.1/introduction/overview

https://iota-beginners-guide.com

https://chrysalis.docs.iota.org

https://iota-beginners-guide.com/future-of-iota/iota-1-5-chrysalis

https://www.boazbarak.org/cs127/Projects/iota.pdf

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
DLT	Distributed ledger technology
IDE	Integrated development environment
IoT	Internet of things
PoW	Proof-of-Work

2 X-CUBE-IOTA1 software expansion for STM32Cube

2.1 Overview

The X-CUBE-IOTA1 software package expands STM32Cube functionality with the following key features:

- Complete firmware to build IOTA DLT applications for STM32-based boards
- Middleware libraries featuring:
  - STSAFE secure element for a secure hardware root of trust (randomness, signatures, storage)
  - Wi-Fi management
  - encryption, hashing, message authentication, and digital signing (Cryptolib)
  - transport-level security (MbedTLS)
  - IOTA Client API to interact with the Tangle
- Complete driver to build applications accessing motion and environmental sensors
- Examples to help understand how to develop an IOTA DLT Client application
- Example to help understand how to build and send to the Tangle an encrypted authenticated stream based on L2Sec, a Layer 2 lightweight security protocol designed to meet the constrained requirements of embedded IoT devices
- Easy portability across different MCU families, thanks to STM32Cube
- Free, user-friendly license terms

The software expansion provides the middleware to enable the IOTA DLT on an STM32 microcontroller.

The IOTA DLT is a transaction settlement and data transfer layer for the Internet of Things (IoT). IOTA allows people and machines to transfer money and/or data without any transaction fees in a trustless, permissionless and decentralized environment. This technology even makes micro-payments possible without the need of a trusted intermediary of any kind.

2.1.1 IOTA 1.0

Distributed Ledger Technologies (DLTs) are built on a node network which maintains a distributed ledger, which is a cryptographically secured, distributed database to record transactions. Nodes issue transactions through a consensus protocol.

IOTA is a distributed ledger technology specifically designed for IoT.

The IOTA distributed ledger is called the Tangle and is created by the transactions issued by the nodes in the IOTA network.

To publish a transaction in the Tangle, a node has to:

1. validate two unapproved transactions called tips
2. create and sign the new transaction
3. perform sufficient Proof-of-Work
4. broadcast the new transaction to the IOTA network

The transaction is attached to the Tangle together with two references pointing to the validated transactions.

This structure can be modeled as a directed acyclic graph, where the vertices represent single transactions and the edges represent references among pairs of transactions.

A genesis transaction is at the Tangle root and includes all the available IOTA tokens (MIOTA).

IOTA 1.0 uses a rather unconventional implementation approach based on trinary representation: every element in IOTA is described using trits = -1, 0, 1 instead of bits, and trytes of 3 trits instead of bytes. A tryte is represented as an integer from -13 to 13, encoded using letters (A-Z) and number 9.

IOTA 1.5 (Chrysalis) replaces the trinary transaction layout with a binary structure.

The IOTA network includes nodes and clients. A node is connected to peers in the network and stores a copy of the Tangle. A client is a device with a seed to be used to create addresses and signatures.

The client creates and signs transactions and sends them to the node so that the network can validate and store them. Withdrawing transactions must contain a valid signature. When a transaction is considered valid, the node adds it to its ledger, updates the balances of the affected addresses and broadcasts the transaction to its neighbors.

2.1.2 IOTA 1.5 - Chrysalis

The objective of the IOTA Foundation is to optimize the IOTA main net before Coordicide and to offer an enterprise-ready solution for the IOTA ecosystem. This is achieved by an intermediate update called Chrysalis.

The main upgrades introduced by Chrysalis are:

- Reusable addresses: the adoption of the Ed25519 signature scheme, replacing the Winternitz one time signature scheme (W-OTS), allows the users to safely send tokens from the same address several times;
- No more bundles: IOTA 1.0 uses the concept of bundles to create transfers. Bundles are a set of transactions linked together by their root reference (trunk). With the IOTA 1.5 update, the old bundle construct is removed and replaced by the simpler Atomic transactions. The Tangle vertex is represented by the Message which is a sort of container that can have arbitrary payloads (i.e., Token payload or Indexation payload);
- UTXO model: originally, IOTA 1.0 used an account-based model for tracking individual IOTA tokens: each IOTA address held a number of tokens and the aggregated number of tokens from all IOTA addresses was equal to the total supply. Instead, IOTA 1.5 uses the unspent transaction output model, or UTXO, based on the idea of tracking unspent amounts of tokens via a data structure called output;
- Up to 8 Parents: with IOTA 1.0, you always had to reference 2 parent transactions. With Chrysalis, a greater number of referenced parent nodes (up to 8) is introduced. To obtain the best results, at least 2 unique parents at a time are recommended.

RELATED LINKS

For more information about Chrysalis, please refer to this documentation page

2.1.3 Proof-of-Work

The IOTA protocol uses Proof-of-Work as a means to rate-limit the network.

IOTA 1.0 used the Curl-P-81 trinary hash function and required a hash with the matching number of trailing zero trits to issue a transaction to the Tangle.

With Chrysalis, it is possible to issue binary messages of arbitrary size. This RFC describes how to adapt the existing PoW mechanism to the new requirements. It aims at being as less disruptive as possible to the current PoW mechanism.

2.1.4 L2Sec protocol for IOTA

The L2Sec is a Layer 2 lightweight security protocol for IOTA, which leverages the services exposed by the STSAFE-A110 secure element that is embedded in the B-L4S5I-IOT01A discovery board.

L2Sec is an alternative approach to IOTA Streams that is a framework used to build cryptographic messaging protocols.

IOTA Streams includes a built-in protocol called Channels to send authenticated messages among two or more parties on the Tangle. Streams is not suitable for the STM32 ecosystem as it is developed in Rust.

Instead, L2Sec is a lightweight protocol native for IOTA Crysalis and written in C. It thus meets the requirements for constrained IoT devices like the B-L4S5I-IOT01A discovery board.

The key features of the L2Sec protocol are:

- usage of the Chrysalis indexation message (data, index) as a transport
- chaining of the channel messages via index, next_index mechanism
- usage of sodium crypto primitives
- digital signature for message integrity and channel ownership proof
- message encryption for data privacy
- digital signature for (origin) authentication
- use of the STSAFE-A110 for a secure hardware root of trust (randomness, authentication signature, storage)

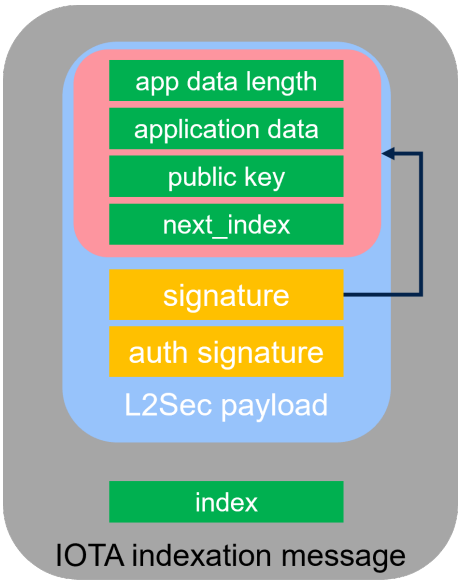
The STSAFE-A110 secure element provides support for:

- true random byte array generation (seed, shared key for encryption, nonce for encryption)
- storage of seed and shared key for encryption
- authentication signature to ensure that a message has been generated by a specific instance of an STM32 device

The following figure shows the fields that compose the IOTA L2Sec payload.

Note: The digest for the signature is calculated over the first four fields, while the digest for the authentication signature is calculated over all the five previous fields.

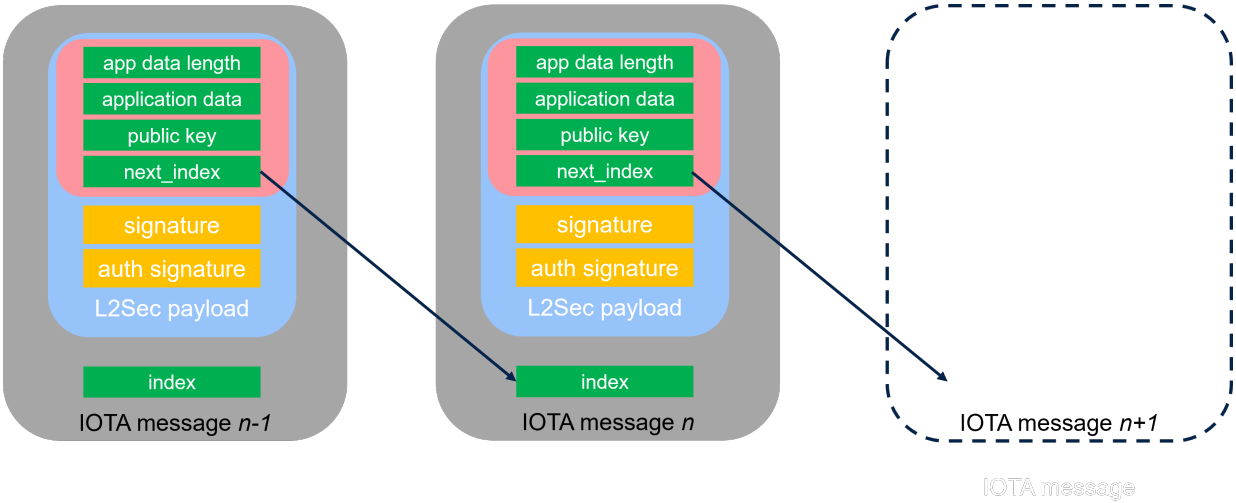
Figure 1. L2Sec payload



The picture below shows how the index, next_index mechanism is used to build an L2Sec message chain.

To generate the index and the next_index, the current implementation uses two random byte arrays: the seed and the next_seed. When moving to the next message generation, the next_index of the current message becomes the index of the next message and a fresh next_index is randomly generated.

Figure 2. L2Sec message chain



On the receiver side, the signature and the public key prevent other recipients to use the discovered next_index to append their chain of messages as they do not know the keypair used to derive the next_index.

RELATED LINKS

IOTA Streams

2.2 Architecture

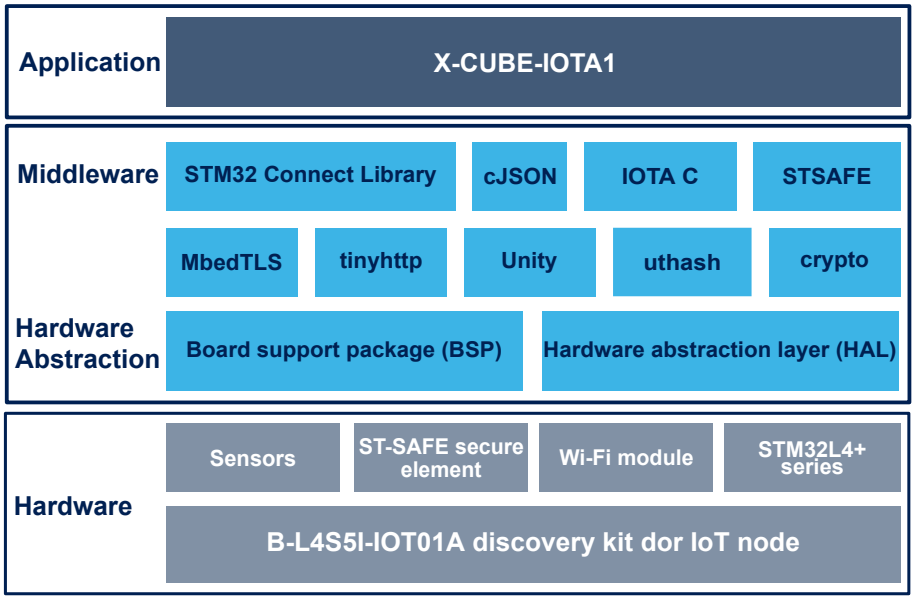
This STM32Cube expansion enables development of applications accessing and using the IOTA DLT middleware.

It is based on the STM32CubeHAL hardware abstraction layer for the STM32 microcontroller and extends STM32Cube with a specific board support package (BSP) for the microphone expansion board and middleware components for audio processing and USB communication with a PC.

The software layers used by the application software to access and use the microphone expansion board are:

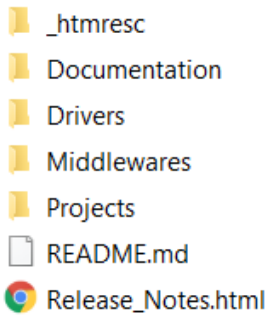
- STM32Cube HAL layer: provides a generic, multi-instance set of APIs to interact with the upper layers (the application, libraries and stacks). It consists of generic and extension APIs based on a common architecture which allows other layers like the middleware layer to function without specific Microcontroller Unit (MCU) hardware configurations. This structure improves library code reusability and guarantees easy device portability.
- Board Support Package (BSP) layer: is a set of APIs which provides a programming interface for certain board specific peripherals (LED, user button etc.). This interface also helps in identifying the specific board version and provides support for initializing required MCU peripherals and reading data.

Figure 3. X-CUBE-IOTA1 software architecture



2.3 Folder structure

Figure 4. X-CUBE-IOTA1 folder structure



The following folders are included in the software package:

- **Documentation**: contains a compiled HTML file generated from the source code and detailed documentation of the software components and APIs

- **Drivers**: contains the HAL drivers and the board-specific drivers for supported board and hardware platforms, including those for the on-board components and the CMSIS vendor-independent hardware abstraction layer for the Arm® Cortex®-M processor series
- **Middlewares**: contains libraries featuring STSAFE secure element services;; Wi-Fi management; encryption, hashing, message authentication, and digital signing (Cryptolib); transport-level security (MbedTLS); IOTA Client API to interact with the Tangle
- **Projects**: contains examples to help you develop an IOTA DLT Client application for the supported STM32-based platform (B-L4S5I-IOT01A), with three development environments, IAR Embedded Workbench for ARM (EWARM), RealView Microcontroller Development Kit (MDK-ARM) and STM32CubeIDE

2.4 API

Detailed technical information with full user API function and parameter description are in a compiled HTML file in the “Documentation” folder.

2.5 IOTA-Client application description

The project files for the IOTA-Client application can be found in: $BASE_DIR\Projects\B-L4S5I-IOT01A\Applications\IOTA-Client.

Ready-to-build projects are available for multiple IDEs.

The user interface is provided via serial port and must be configured with the following settings:

Figure 5. Tera Term - terminal setup

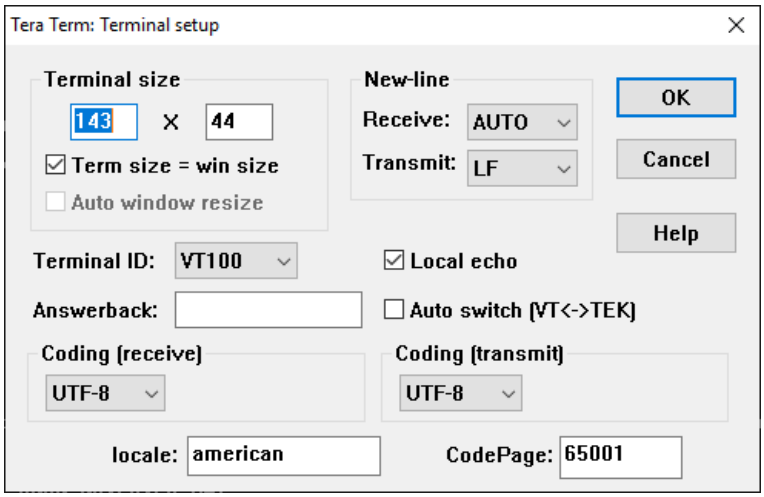
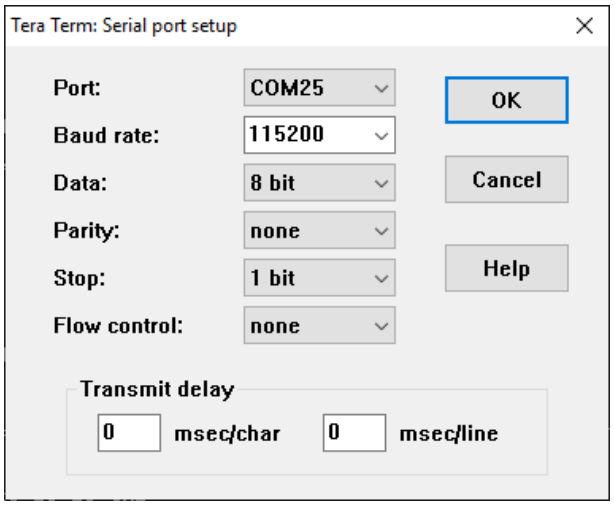


Figure 6. Tera Term - serial port setup



To run the application, follow the procedure below.

Step 1. Open a serial terminal to visualize the log of messages.

Step 2. Enter your Wi-Fi network configuration (SSID, security mode, and password).

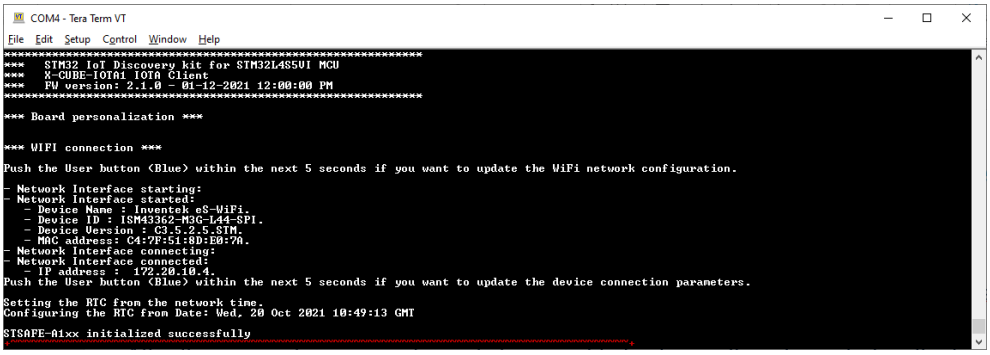
Step 3. Set the TLS root CA certificates.

Step 4. Copy and paste the contents of Projects\B-L4S5I-IOT01A\Applications\IOTA-Client\usertrust_thetangle.pem.

The device uses them to authenticate the remote hosts through TLS.

Note: After configuring the parameters, you can change them by restarting the board and pushing the user button (blue button) within 5 seconds. This data is saved in the Flash memory.

Figure 7. Wi-Fi parameter settings



Step 5. Wait for the message “Press any key to continue” to appear.
The screen is then refreshed with the list of the main functions:

- Get node info
- Get a message payload
- Send a generic indexation message
- Send an indexation message including sensor data
- Send an encrypted indexation message
- Enter the L2Sec example submenu
- Enter the STASFE example submenu
- Other test functions

Figure 8. Main menu

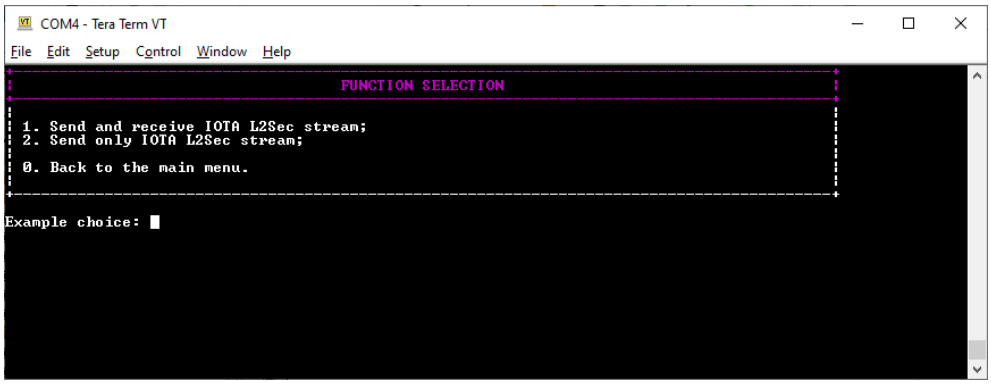
Step 6. Choose option 6 to test the L2Sec message stream:

- Send only
- Send and receive

Note: Enter the number of messages to be sent, otherwise a default number (defined by the constant `N_MESSAGES` in the I2sec_example.c source file) is used.

Each message contains the following application data: <DeviceName>,<Timestamp>,<Temperature>, and <Humidity>.

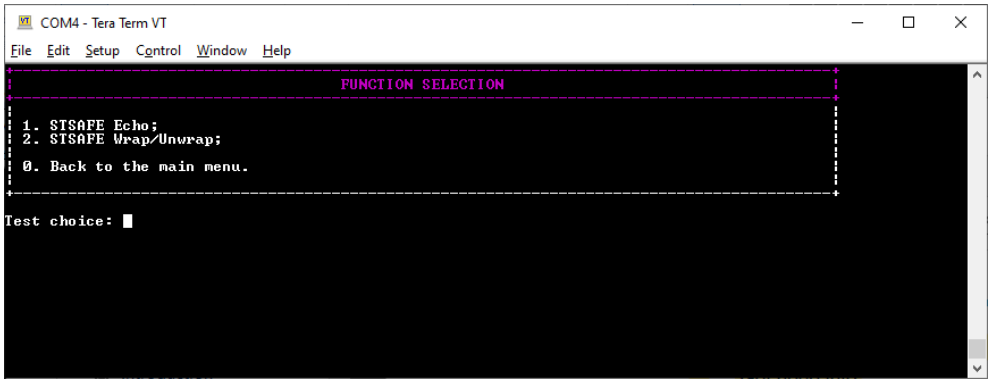
Figure 9. L2Sec submenu



Step 7. Choose option 7 to test some basic STSAFE demonstrations:

- Echo
- Wrap/Unwrap local envelope

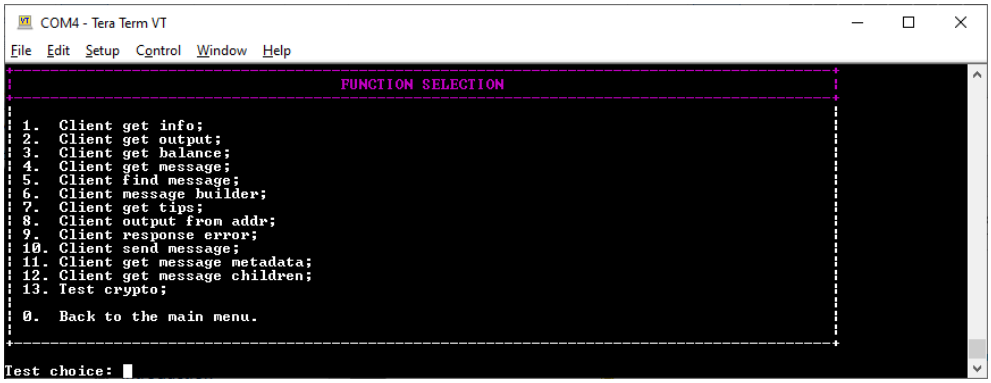
Figure 10. STSAFE submenu



Step 8. Choose option 8 to test one of the following IOTA basic functions:

- Get node info
- Get output
- Get balance
- Get message
- Find message
- Message builder
- Get tips
- Get output from address
- Response error
- Send message
- Get message metadata
- Get message children
- Test crypto

Figure 11. Test functions submenu



RELATED LINKS

For further details about IOTA 1.5 functions, refer to the IOTA C Client documentation

3 System setup guide

3.1 Hardware description

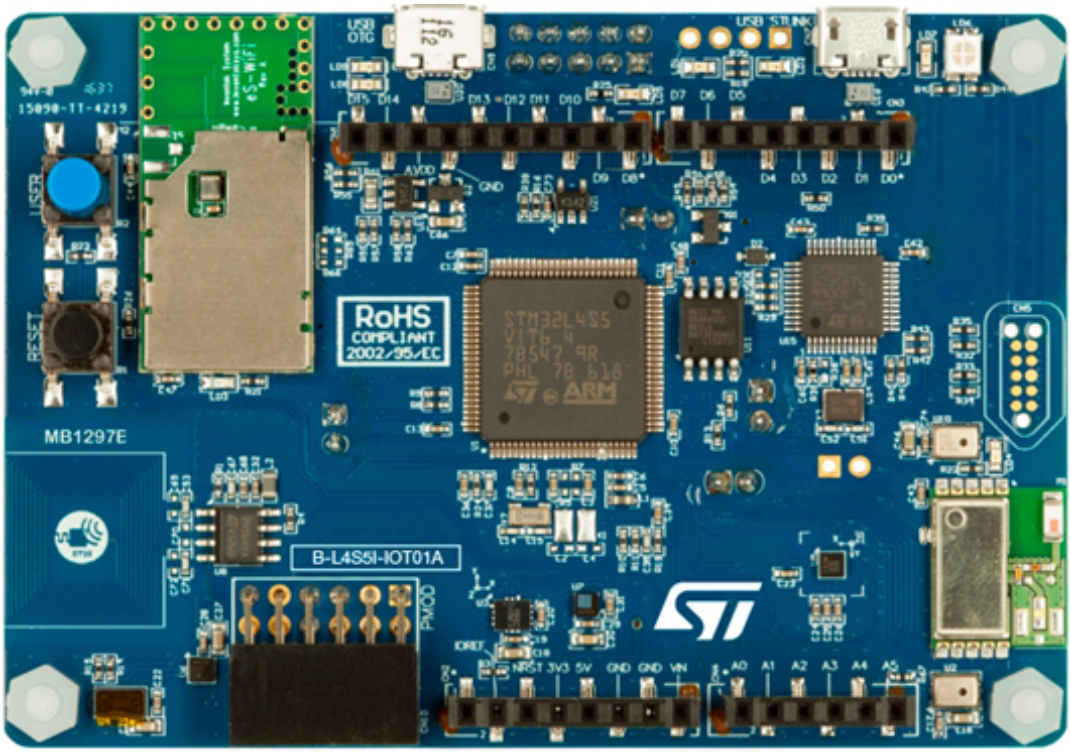
3.1.1 STM32L4+ discovery kit for IoT node

The B-L4S5I-IOT01A discovery kit for IoT node allows you to develop applications to directly connect to cloud servers.

The Discovery kit enables a wide variety of applications by exploiting low-power communication, multi-way sensing and Arm® Cortex® -M4+ core-based STM32L4+ series features.

It supports Arduino Uno R3 and PMOD connectivity providing unlimited expansion capabilities with a large choice of dedicated add-on boards.

Figure 12. B-L4S5I-IOT01A discovery kit



3.2 Hardware setup

The following hardware components are needed:

1. one STM32L4+ Discovery kit for IoT node equipped with Wi-Fi interface (order code: B-L4S5I-IOT01A)
2. a USB type A to Mini-B USB Type B cable to connect the STM32 discovery board to the PC

3.3 Software setup

The following software components are needed to set up the development environment for creating IOTA DLT applications for the B-L4S5I-IOT01A:

- X-CUBE-IOTA1: firmware and related documentation is available on st.com

- development tool-chain and compiler: the STM32Cube expansion software supports the following environments:
  - IAR Embedded Workbench for ARM® (EWARM) toolchain + ST-LINK/V2
  - RealView Microcontroller Development Kit (MDK-ARM) toolchain + ST-LINK/V2
  - STM32CubeIDE + ST-LINK/V2

3.4 System setup

The B-L4S5I-IOT01A Discovery board allows the exploitation of the IOTA DLT features.

The board integrates the ST-LINK/V2-1 debugger/programmer. You can download the relevant version of the ST-LINK/V2-1 USB driver at STSW- LINK009.

Revision history

Table 2. **Document revision history**

Date	Revision	Changes
13-Jun-2019	1	Initial release
18-Jun-2019	2	Updated Section 3.4.8.1 TX_IN and TX_OUT, Section 3.4.8.3 Sending data through zero-value transactions and Section 3.4.8.4 Sending funds through transfer transactions.
06-May-2021	3	Updated Introduction, Section 1 Acronyms and abbreviations, Section 2.1 Overview, Section 2.1.1 IOTA 1.0, Section 2.1.3 Proof-of-Work, Section 2.2 Architecture, Section 2.3 Folder structure, Section 3.2 Hardware setup, Section 3.3 Software setup and Section 3.4 System setup. Removed Section 2 and replaced by a link in the Introduction. Removed Section 3.1.2 Transactions and bundles, Section 3.1.3 Account and signatures, Section 3.1.5 Hashing. Section 3.4 How to write applications and related sub-sections, Section 3.5 IOTALightNode application description and related subsections, and Section 4.1.1 STM32 Nucleo platform. Added Section 2.1.2 IOTA 1.5 - Chrysalis, Section 2.5 IOTA-Client application description, Section 2.4 API and Section 3.1.1 STM32L4+ Discovery kit IoT node.
15-Nov-2021	4	Updated Section 2.1 Overview, Section 2.1.1 IOTA 1.0, Section 2.2 Architecture, Section 2.3 Folder structure and Section 2.5 IOTA-Client application description. Added Section 2.1.4 L2Sec protocol for IOTA.

Contents

List of figures

List of tables