Standalone Solution

# Prerequisites

## Software

Following are the software prerequisites to get started with the standalone solution:

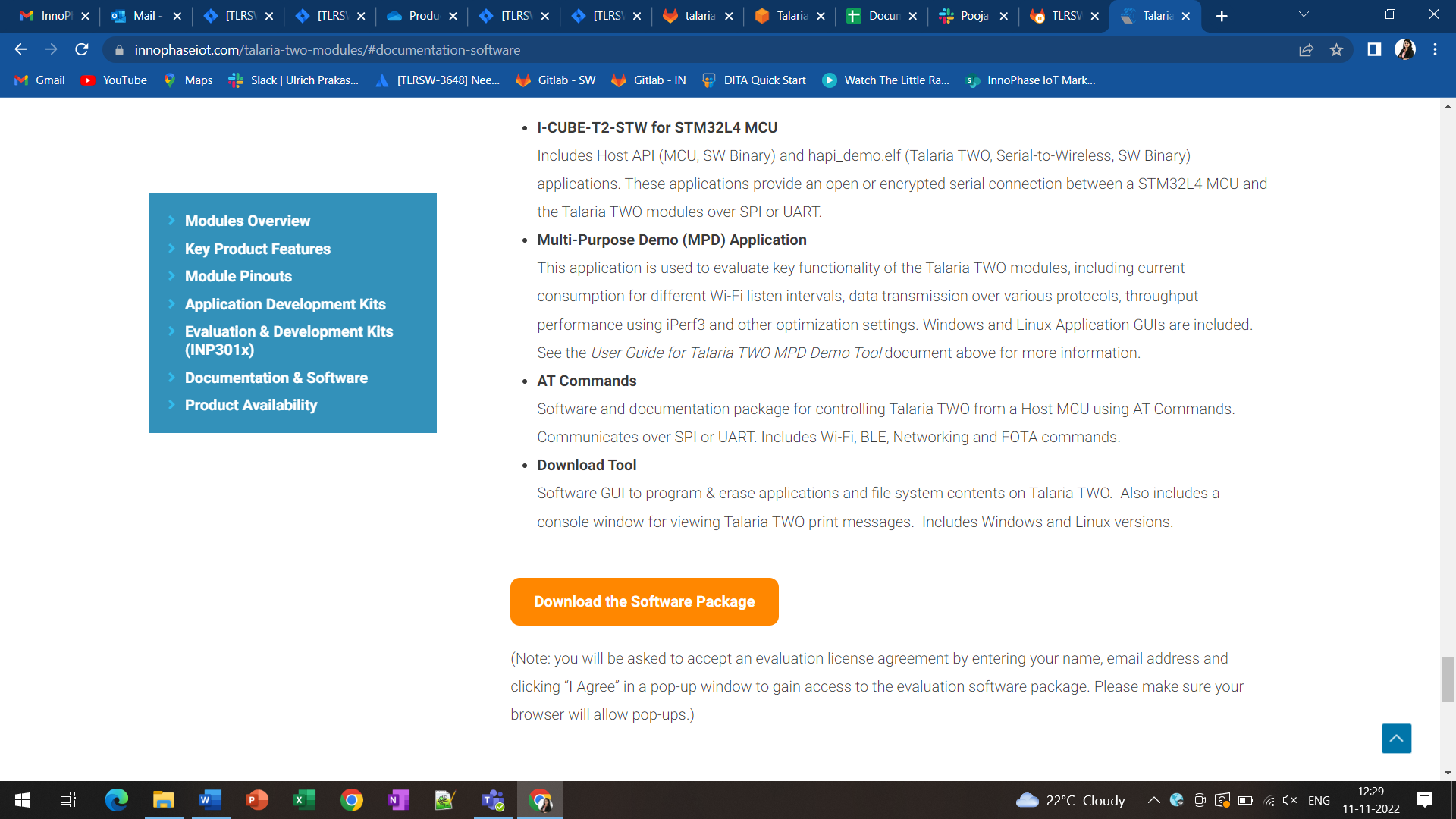
1. Windows/Linux PC
2. Wi-Fi Access Point
3. Minicom/TeraTerm VT Utility installed on PC (add a download link)
4. EVK package
5. SDK package

### Evaluation Kit (EVK)

#### Download the Evaluation Software Package (EVK)

The EVK package (Open\_Web\_SDK\_x.y) includes various evaluation binaries and the associated programming tools for the INP301x EVB-A boards to enable quick evaluation of the INP101x modules.

Download the software package from the InnoPhase website: <https://innophaseiot.com/talaria-two-modules/#product-availability>.



#### EVK Package Contents Walkthrough

The EVK package contains the following folders:

1. alexa\_ready: Firmware images for Alexa\_Ready application which can be used to understand the protocols and functionality supported on Talaria TWO.
2. at: Firmware images for AT Commands application.
3. Download\_Tool: Talaria TWO programming tool (available for both Windows & Linux platforms)
4. I-CUBE-T2-STW-lib: STM software package for STM’s NUCLEO-STM32L4A6ZG or NUCLEO-STM32L433RC-P with ready-to-run firmware examples to support quick evaluation and development of MQTT/HTTP/AWS/AZURE and HTTPS IoT Cloud applications on STM32L4 Nucleo boards with InnoPhase Talaria TWO Wi-Fi add-on boards.

Folder details are as follows:

* 1. Documentation: This folder includes all applications related to Talaria TWO and STMCubeL4.
  2. Drivers: This folder includes BSP files for STM32L4 Nucleo boards. Refer to release notes for further details
  3. Middleware: This folder includes Middleware FreeRTOS and third-Party Middleware InnoPhase-HAPI to act as a host interface for Talaria TWO Modules (INP101X) Wi-Fi devices
  4. Projects: This folder includes the applications to demonstrate the INP101X features such as Wi-Fi, BLE, cloud apps on modules 32L4A6ZG-NUCLEO and 32L433RC-P-NUCLEO. Refer to release notes for further details.
  5. Utilities: This folder includes Download tool GUI and the firmware binary for Talaria TWO EVB (INP301x)

1. MPD: Multi-purpose Demo GUI for evaluating example applications and network protocols

### Software Development Kit

#### Download the Software Development Package (SDK)

To download the software development package, execute the following steps:

1. Go to the InnoPhase website (<https://innophaseiot.com/>) and click on Register.

A screenshot of a computer

Description automatically generated

1. Provide the appropriate details to register onto the InnoPhase Customer Portal.

**Note**: InnoPhase requires a signed MNDA and Development Tool License Agreement (DTLA) to be signed prior to granting access to the Customer Portal.

1. On successfully registering to the Customer Portal, the following screen will appear:

A screenshot of a computer

Description automatically generated

1. Navigate to the Software Tab and download the appropriate software package(s):

A screenshot of a computer

Description automatically generated

#### SDK Package Contents Walkthrough

The SDK software package (sdk\_x.y.zip) provides software APIs with ready-to-run firmware examples to support quick evaluation and development of MQTT/HTTP/HTTPS/AWS and AZURE IoT Cloud applications with Talaria TWO.

Folder details are as follows:

1. **pc\_tools**: Programming (Download), multi-purpose demo GUI and T2\_Flasher tools. These tools are available for both Windows and Linux platforms.
2. **binaries**: Firmware images which can be used to understand the protocols and functionality supported on Talaria TWO. The eval folder consists of firmware image for MPD and Alexa\_Ready application, while the product folder consists of the firmware image for AT commands.
3. **apps**: production ready example applications that the user can run on Talaria TWO. Following applications are available in the release package:

|  |  |
| --- | --- |
| **Protocol** | **Description** |
| alarm | Describes alarm functionality in Talaria TWO. |
| filesystem | Contains APIs to check and mount Talaria TWO's file system |
| fota | Demonstrates the FOTA (Firmware Over The Air) process |
| gordon | Flashing utility using UART interface |
| gordon-jtag | Flashing utility using JTAG / SWD interface |
| helloworld | Basic helloworld application |
| iot\_aws | Demonstrates using Talaria TWO board and the SDK with Amazon Web Services (AWS) IoT |
| iot\_azure | Readme file containing the link to IoT\_Azure application note present in Github |
| iperf3 | iperf3 server application on Talaria TWO |
| ssbl | Demonstrates running and switching between multiple applications on Talaria TWO using SSBL |
| stw | Serial to Wi-Fi application over HIO |
| stw\_multi\_proto | Serial to Wi-Fi application used in hosted mode of operation |

1. **examples**: Demo/sample applications that the user can run on Talaria TWO firmware. Following examples are available:

|  |  |
| --- | --- |
| **Protocol** | **Description** |
| adc | Example for using the Analog to Digital Convertor (ADC) peripheral of Talaria TWO modules |
| at\_custom\_cmd | Demonstrates custom AT commands which the user can use apart from the standard commands |
| ble\_beacons | Example codes describing the basic concept of Eddystone Beacon   * Eddystone UID * Eddystone URL * Eddystone TLM |
| ble\_wifi\_bridge | Example code for receiving a text message from a connected BLE client and publishing it to a CloudMQTT broker |
| chip\_monitor | Describes the application for fetching the changes in the values of device core temperature, Voltage of VBAT, external ADC and estimated current consumption of Talaria TWO device |
| crash\_handling | Example code for using the crash handler API to handle and debug error cases |
| gpio | Example codes for using the interface for GPIO |
| http\_client | Example codes for using HTTP client APIs to connect to HTTP servers in secured (HTTPS) and non-secured way |
| i2c | Demonstrates usage of I2C on Talaria TWO |
| i2s\_audio | Example codes for using I2S peripheral of INP1010/INP1011/INP1012/INP1013 Talaria TWO modules for playing audio |
| ifttt | Example application for using Talaria TWO with IFTTT |
| innoos\_memory\_mgmt | Example code for basic memory management and error handling |
| innoos\_msg\_q | Example codes describing the use of message queue APIs |
| innoos\_threads\_semaphores | Example codes describing the use of threads and semaphore APIs |
| innoos\_timers\_callouts | Example codes describing the use of Timers and Callout APIs |
| innoos\_work\_q | Example codes describing the use of Work Queue APIs |
| lp\_scan | Demonstrates the basics of the Low Power Wi-Fi scan feature in InnoOS™ |
| lp\_uart | Example codes describing the use of UART APIs |
| mdns | Demonstrates using the mDNS APIs provided by the mDNS module |
| mqtt | Example codes for using the publish/subscribe operation of MQTT in both secured and non-secured modes |
| prov | A demo Provisioning application using BLE for provisioning AP credentials at Talaria TWO from a mobile application |
| pwm | Demonstrates usage of Pulse Width Modulation peripheral of INP1010/INP1011/INP1012/INP1013 Talaria TWO modules |
| radio\_module\_params | Describes using the Radio and Module parameters on the Talaria TWO module |
| sleep\_enable\_disable | Demonstrates the basics of sleep management in InnoOS™ (os\_suspend\_enable() and os\_suspend\_disable() functions) |
| socket\_wakeup | Demonstrates the basics of sleep management in InnoOS™ (Talaria TWO wake-up from sleep mode) |
| spi | Demonstrates the usage of the Software SPI Master (SSM) on Talaria TWO |
| spi\_flash | Example codes describing using of SPI Flash APIs for Talaria TWO EVK |
| un-assoc | Example codes describing Wi-Fi un-associated mode transmission APIs available in the SDK, call-back events, notifications and associated data structures |
| using\_ble | Introduction to BLE APIs through code samples consisting of a server and client application |
| using\_filesystem | Demonstrates using the filesystem APIs to show case the filesystem functionalities on the Talaria TWO EVK |
| using\_hio | Demonstrates the fundamentals of developing HIO based application on both host and the Talaria TWO EVK |
| using\_sntp | Demonstrates fetching time from NTP server using SNTP |
| using\_wifi | Example codes describing the Wi-Fi connection manager APIs |
| watchdog\_timer | Demonstrates managing Talaria TWO watchdog timer using the functions provided by the watchdog driver |
| wcm\_multi\_ap | Demonstrates the application example of Wi-Fi Connection with Multi-Access Point (WCM\_MULTI\_AP) available in the SDK |
| wcm\_pm | Demonstrates the Wi-Fi Connection Manager power management APIs |
| websocket | Demonstrates using the WebSocket client APIs provided by the WebSocket module |

1. **conf**: Make and debugger configuration files. These files include Id and linker scripts and make file rules used for firmware configuration.
2. **doc**:

The doc folder contains the following sub-folders:

* 1. reference\_guides
     1. api\_reference\_guide: API reference guide describes the programmers APIs.
     2. bootargs\_reference\_guide: Bootargs reference guide describes the use of Talaria TWO boot arguments (bootargs) in different scenarios using different feature and protocols.
     3. cli\_reference\_guide: CLI reference guide serves as a reference guide for Talaria TWO CLI commands, its usage, use case or examples in different scenarios supported by Talaria TWO modules.
  2. user guides.
     1. ug\_evb\_a: Talaria TWO Evaluation Board an overview of the evaluation board explaining its key features and functions
     2. ug\_eclipse\_setup\_windows: Eclipse setup in Windows describes developing an application using Eclipse and Talaria TWO SDK
     3. ug\_eclipse\_setup\_linux: Eclipse setup in Linux describes the procedure to build and debug an application on Eclipse IDE using Talaria TWO SDK
     4. ug\_env\_setup\_linux: Environment set-up with Talaria TWO for Linux describes setting up the development environment for Talaria TWOTM SDK on an Ubuntu VirtualBox with a Windows 10 host
     5. ug\_wsl: Windows Subsystem for Linux describes developing an application using Windows Subsystem for Linux (WSL) and Talaria TWO SDK
     6. ug\_coredump\_generation\_and\_anaysis: Coredump generation and analysis describes generating a coredump file used for analyzing the cause of the crash
     7. ug\_debugging\_using\_gdb: Debugging using GDB describes the procedure for debugging the applications using GDB to work with OpenOCD
     8. porting\_guide\_freertos\_to\_innoos: Porting guide describes the procedure to port FreeRTOS code to InnoOS
     9. porting\_guide\_zephyros\_to\_innoos: Porting guide describes porting of the ZephyrOS code to InnoOS
     10. ug\_firmware\_sdk: Firmware SDK user guide describes developing applications for the Talaria TWO device
     11. ug\_programming\_using\_black\_magic\_board: Black magic board user guide describes the Black Magic Programmer Board which provides a programming and debugging interface for Talaria TWO modules
     12. ug\_programming\_using\_INP3000: The INP3000 programming user guide describes the INP3000 programmer board which provides a programming interface for Talaria TWO modules
     13. ug\_visual\_studio\_setup\_windows: Visual Studio Setup in Windows user guide describes the procedure to build and debug an application on Visual Studio Code Editor in Windows using Talaria TWO SDK

1. **components**: Files which serve as common components which all applications can make use of. It acts as a library, where an application like HTTP, SNTP, etc., can link to this library and directly use the function. The following protocols are available in the components folder serving as a library for applications:
   1. alarm
   2. checksum
   3. fota
   4. http
   5. json
   6. mdns
   7. mqtt
   8. out
   9. prov
   10. sntp
   11. ssl\_wrap
   12. utils
   13. websocket
2. **include**: SDK include files (.h files) for all applications/examples in the release package.
3. **lib**: SDK library files which can be used by all applications/examples.
4. **script**: Helper scripts/utilities used to achieve multiple functionalities like program, read, write and so on.
5. **tools**: Certain tools, supporting files and scripts used for development on Talaria TWO. This directory further contains the following three directories:
   1. fletcher32 - tool used to create checksum files for checking the integrity of the configuration files. Currently part.json, boot.json and fota\_config.json files’ integrity is checked using the checksum.
   2. mklittlefs - tool used to create the file system image.
   3. partition\_files - flash partition files used with SSBL setup (ssbl\_part\_table.json) and without SSBL setup (standard\_part\_table.json).
6. **root\_fs**: Basic files that need to be present in root\_fs image as required for SSBL. It also contains the script to build the root\_fs.
7. **solutions**: Contains pre-built binaries, source code for custom applications, library, json file and root files for Dual-Stack solution.