
Trust&Go Step by Step Guide - Loading Manifest & Connect to AWS-IoT

Table of Contents

1	<i>Introduction</i>	3
1.1	<i>Getting started with Jupyter Notebook Tutorials</i>	3
1.1.1	Starting Jupyter Notebook.....	3
1.2	<i>Jupyter Notebook Basics</i>	3
1.2.1	The Notebook dashboard	3
1.3	<i>Introduction to Jupyter Notebook GUI</i>	4
2	<i>Jupyter Notebook Tutorials</i>	6
3	<i>Manifest Generation Notebook</i>	7
4	<i>Use Case Prototyping</i>	11
4.1	<i>Running AWS IoT example on Jupyter Notebook</i>	11
4.2	<i>Running AWS IoT example on Embedded platform</i>	18
4.2.1	Atmel Studio:	18
4.2.2	MPLAB:	22
4.3	<i>CryptoAuth TrustPlatform Factory reset</i>	26

1 Introduction

This document explains step by step process involved in uploading a manifest file to AWS cloud. If you are already familiar with Jupyter Notebook you can skip this section and move to Section 2.

1.1 Getting started with Jupyter Notebook Tutorials

Jupyter Notebook is open source web application which allows you to create documents that contain code that you can execute in place as well as narrative text. It provides GUI elements, ability to execute code in place, ability to add images and gives it the look and feel that normal code files lack.

Jupyter notebooks are mainly used to explain/evaluate code in an interactive way.

1.1.1 Starting Jupyter Notebook

Jupyter notebook can be launched from the Anaconda Navigator main window.



1.2 Jupyter Notebook Basics

It is recommended to become familiar with Jupyter basic concepts with the online documentation, <https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html>

Some of the content is duplicated here for convenience. The online documentation should always be used as a reference.

1.2.1 The Notebook dashboard

When you first start the notebook server, your browser will open Notebook dashboard. The dashboard serves as a home page for the notebook. Its main purpose is to display the Notebooks and files in the current directory.

For example, here is a screenshot of the Jupyter dashboard. The top of the notebook list displays clickable breadcrumbs of the current directory. By clicking on these breadcrumbs or on sub-directories in the notebook list, you can navigate your file system.



1.3 Introduction to Jupyter Notebook GUI.

Jupyter Notebooks contain cells where you can either write code or markdown text. Notebooks contain multiple cells, some set as code and others markdown. Code cells contain code that can be executed live, and markdown contains text and images to explain the code.

Below image shows some options in a typical Jupyter Notebook. Individual cells can be executed by pressing on the RUN button as shown in the below image.

All cells in the Notebook can be executed in order by **Kernel->Restart & Run All**.

The screenshot shows a Jupyter Notebook window with the following elements and annotations:

- Toolbar:** A red box highlights the **+** icon with the label "Create new cell". Another red box highlights the **Run** button (a play icon) with the label "Run selected cell". A third red box highlights the **Markdown** dropdown menu with the label "Says if a particular cell is code or markdown".
- Cell Selection:** A blue vertical band on the left side of the notebook is highlighted with a red box and labeled "Blue band represents currently selected cell".
- 2. Tutorial:** The main content area shows a tutorial with the following sections:
 - 2.1. Establish the communication between the host (PC) and the ECC608A:** A red box highlights this section, labeled "MARKDOWN CELL". It contains two steps:
 1. Initialize the library which also loads it
 2. Check if the correct device is connected to the PC. If the wrong device is connected, a ValueError exception will be raised.
 - 2.1.1. Select the CryptoAuthLib Hardware Abstraction layer to KIT USB HID:** A red box highlights this section, labeled "CODE CELL". It contains a code cell with the following Python code:

```
In [ ]: ATCA_KIT_I2C_IFACE = 1
cfg = cfg_ateccx08a_kithid_default()
cfg.devtype = get_device_type_id('ATECC608A')
cfg.cfg.atcahid.dev_interface = ATCA_KIT_I2C_IFACE
cfg.cfg.atcahid.dev_identity = 0x6C #TFLXTLS device I2C address
ATCA_SUCCESS = 0
print("Set devtype as ECC608A - Successfull")
```
 - 2.1.2. Establish the connection with the ECC608A-TLXTLS device on the USB dongle and check its type:** A red box highlights this section. It contains a code cell with the following Python code:

```
In [ ]: # Initialize interface
assert atcab_init(cfg) == ATCA_SUCCESS, "Can't connect to the USB dongle"

# Get connected device type
info = bytearray(4)
assert atcab_info(info) == ATCA_SUCCESS, "Can't read the ECC608A device information"
dev_type = get_device_type_id(get_device_name(info))

# Checking if the connected device matches with selected device
if dev_type != cfg.devtype:
    assert atcab_release() == ATCA_SUCCESS
    raise ValueError('Device not supported')

print("Device initialization - Successfull")
```

To run all cells in sequence.

The screenshot shows the **Kernel** menu in the Jupyter Notebook interface. The menu is open, and the **Restart & Run All** option is highlighted with a red box. A tooltip for this option reads: "Restart the Kernel and re-run the notebook". The menu also includes other options: **Interrupt**, **Restart**, **Restart & Clear Output**, **Reconnect**, **Shutdown**, and **Change kernel**.

2 Jupyter Notebook Tutorials

The TrustPlatform Design Suite comes with a Notebook Tutorials to easily prototype popular use cases for Trust&GO devices. Here are the available Jupyter Notebook Tutorials.

Jupyter Notebook Tutorials	Relative Path	Applicable Devices
Manifest Generation	TrustnGO\00_resource_generation\TNGTLS_manifest_file_generation.ipynb	Trust&GO
AWS Connect	TrustnGO\01_aws_connect\notebook\ TNGTLS_aws_connect.ipynb	Trust&GO
GCP Connect	TrustnGO\02_gcp_connect\notebook\TNGTLS_GCP_connect.ipynb	Trust&GO

3 Manifest Generation Notebook

Trust&GO device is one of the three devices available in the Trust Platform USB Dongle Board.

Trust&GO devices come with pre-programmed certificates in slots 10, 11 and 12, also slots 0-4 have pre-generated private keys, other than the previously mentioned slots all the other slots are locked.

The secure element manifest format is designed to convey the unique information about a device including its unique ID (e.g. serial number), public keys, and certificates. The manifest file generated can be used to register the device to cloud providers.

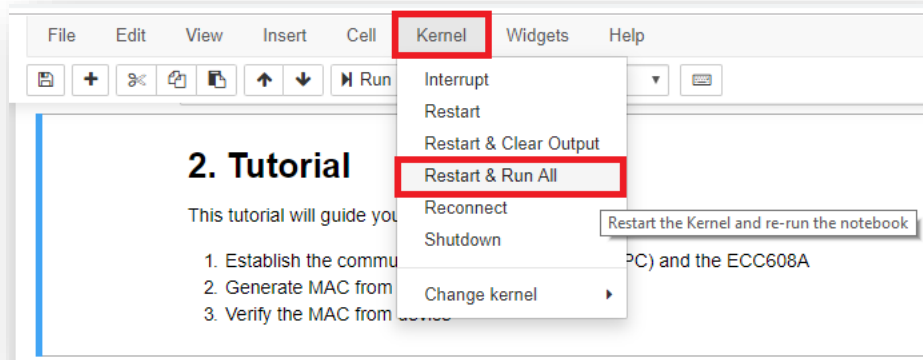
By default, Jupyter starts in Users directory (\$HOME for MacOS or Linux systems). For the remainder of this document, it will be assumed that the trust_platform folder is contained in Users directory. If this is not the case, please move trust_platform folder to your Users directory

Within the Jupyter Dashboard, navigate **trust_platform\DesignTools\TrustnGO\00_resource_generation** folder to open **TNGTLS_manifest_file_generation.ipynb**



Run all cells of the TNGTLS_manifest_file_generation Notebook: Kernel->Restart & Run All

Note: Before executing the cells on Crypto Trust Platform, its required to have factory default program running on SAMD21 of Trust Platform. Refer to [4.3 CryptoAuth TrustPlatform Factory reset](#) section for reloading default program.



If all the steps ran without errors, you will see result as shown below.

Loading logger key

Generating self-signed logging certificate

Saving to log_signer.crt

TNG Root Certificate:

Crypto Authentication Root CA 002

-----BEGIN CERTIFICATE-----

```
MIIB8TCCAzegAwIBAgIQd9NtlW7IrmIF5Y46y5hagTAKBggqhkJOPQQDAjBPMSEw
HwYDVQQKDBhNaWNyb2NoaXAgVG9jaG5vbG9neSBjb2N0aW50aW50aW50aW50aW50
byBBdXR0ZW50aW50aW50aW50aW50aW50aW50aW50aW50aW50aW50aW50aW50aW50
MDU4MTEwODE5MTIxOVowTzEhMB8GA1UECgwYTWljcm9jaGlvIFRlY2hub2xvZ3kg
SW5jMSowKAYDVQQDDCFDcnlwdG8gQXV0aGVudG9jaG5vbG9neSBjb2N0aW50aW50
WTATBgqhkJOPQIBBgqhkJOPQMBBwNCAAS9VOZt44dUhABrU64VgNUKoGnnit9V
eNhc4tVN1bgwKWv/3W5vclb72Z7xoRaxHTOtSRA6oYWHOdz65DfhnWNOo1MwUTAd
BgNVHQ4EFgQUeu19bca3eJ2yOAGl6EqMsKQOKowwHwYDVR0jBBgwFoAUeu19bca3
eJ2yOAGl6EqMsKQOKowwDwYDVR0TAQH/BAUwAwEB/zAKBggqhkJOPQQDAgNIADBF
AiEAodxjRZDsgZ7h3luBEmVRrdTCxPj1lSgu4EvnaOx8AnMCID5rp06eTArWjCSw
+y7nk9LmvpRlyhXQ6lvIf1V5mVyt
```

-----END CERTIFICATE-----

TNG Root Public Key:

-----BEGIN PUBLIC KEY-----

```
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEvVTmbeOHVlQAa1OuFYDVCqBp54rf
VXjYXOLVTDW4MClr/91ub3JW+9me8aEWsR0zrUkQOqGFhznc+uQ34Z1jTg==
```

-----END PUBLIC KEY-----

Validate Root Certificate:

OK

TNG Signer Certificate:

Generated the manifest file 012342bb675dc30501_manifest.json

The Notebook will be used to generate a manifest file which can be uploaded into the public cloud provider of your choice (Google GCP, AWS IoT and soon to be supported Microsoft Azure). TNGTLS Manifest Generation notebook needs to be run for all Trust&Go example Notebooks that require a Manifest file.

4 Use Case Prototyping

This hands-on lab is intended to demonstrate the usage of Trust&GO to secure an AWS IoT connection based on a custom PKI.

Here are the steps that will be required to complete this Tutorial:

- Configure AWS CLI
- Upload Manifest File
- Build the AWS IoT device source code and flash it to the USB Dongle Board

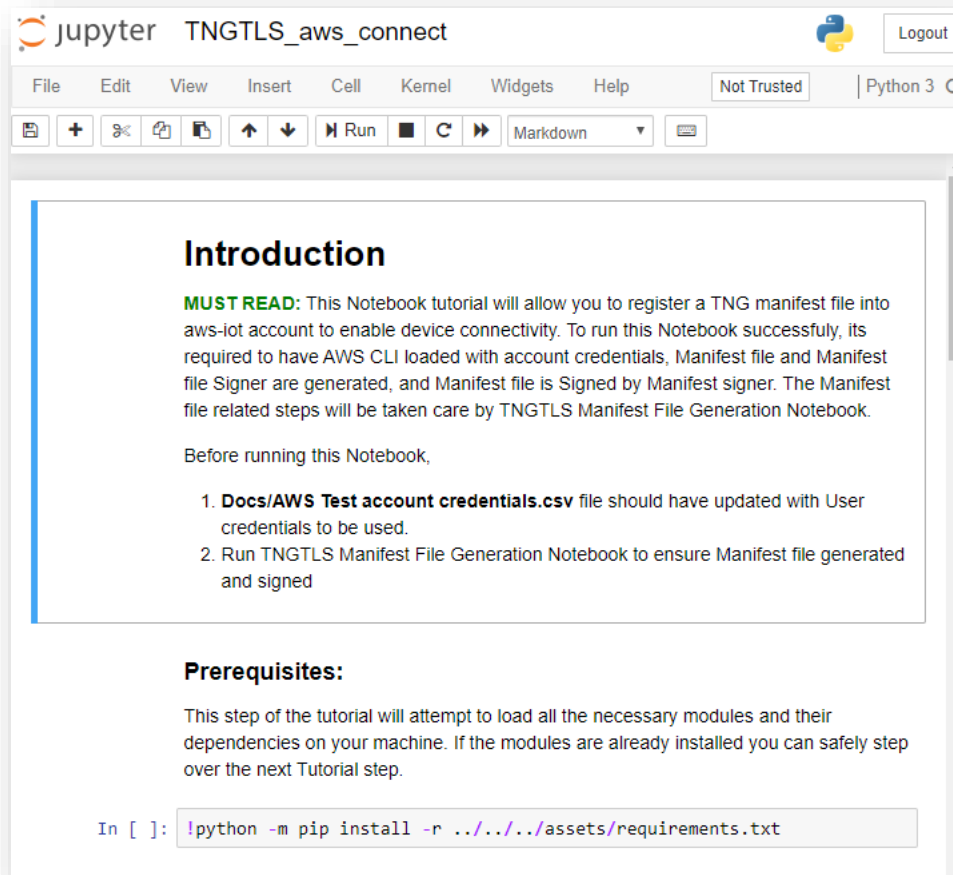
4.1 Running AWS IoT example on Jupyter Notebook

By running following step, we can configure AWS CLI and able to upload manifest file. To upload manifest file, we would be using the manifest file and logger file generated in the previous section 3. TNGTLS Manifest File generation notebook. The Manifest file contains information about the device including serial number, public keys and certificates.

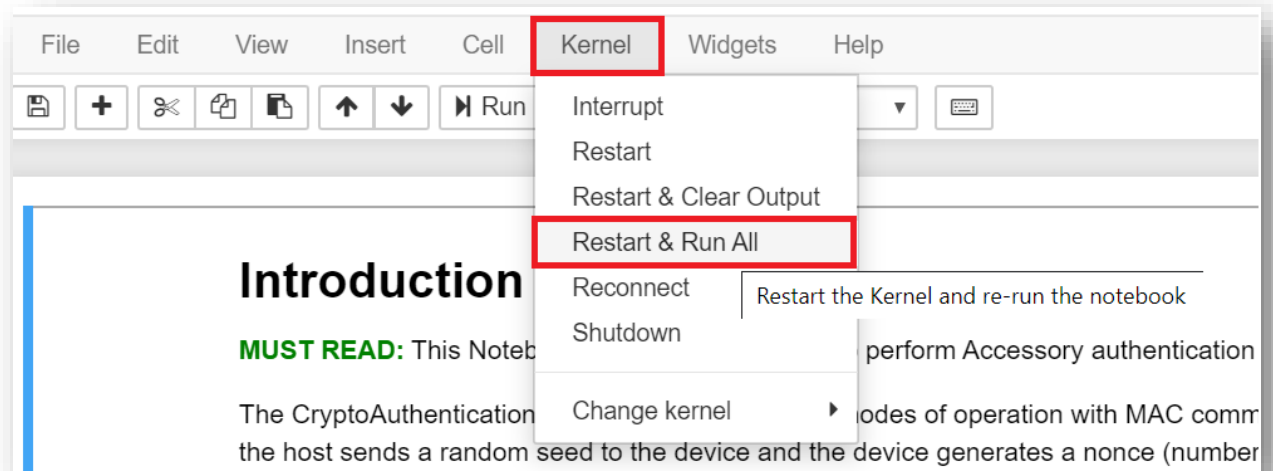
1. From the Jupyter Home page, navigate to **TrustnGO\01_aws_connect\notebook\TNGTLS_aws_connect.ipynb** notebook file and open it.



Opening the notebook from Jupyter home page should load the following on the browser.



2. Run All cells by using Kernel -> Restart & Run all



3. After running all cells in notebook, it will create four buttons 1. "Config AWS-CLI", 2. "Load Manifest JSON File", 3. "Load Validation CERT File" and 4. "AWS GUI".
4. Press "**Config AWS-CLI**" button to configure AWS command line interface with AWS account credentials.

Once you press the button, it will ask AWS User Region: where region name has to be mentioned as shown in below image.

Name	Value	Type	Location
profile	<not set>	None	None
access_key	*****5SSB	shared-credentials-file	
secret_key	*****pWV0	shared-credentials-file	
region	us-east-2	config-file	~/.aws/config

Step1. Config AWS-CLI

Step2 a. Load Manifest JSON File (0)

Step2 b. Load Validation CERT File (0)

Step2 c. Upload Manifest File

Before clicking AWS GUI its required to have Manifest file uploaded and C and wifi credentials. Click below AWS GUI button ONLY after establishing

Step3. AWS GUI

AWS User Region:

After enter your region press 'enter'. On successful AWS CLI configuration, you will see the result as shown in below image

Step1. Config AWS-CLI

Step2 a. Load Manifest JSON File (0)

Step2 b. Load Validation CERT File (0)

Step2 c. Upload Manifest File

Before clicking AWS GUI its required to have Manifest file uploaded and C and wifi credentials. Click below AWS GUI button ONLY after establishing

Step3. AWS GUI

AWS User Region:

Setting aws access key...
 Setting aws secret access key...
 Setting aws region...

Name	Value	Type	Location
profile	<not set>	None	None
access_key	*****5S5B	shared-credentials-file	
secret_key	*****pwV0	shared-credentials-file	
region	us-east-2	config-file	~/.aws/config

- Press **"Load Manifest JSON File"** button, it will open file explorer window, there you need to navigate **TrustnGO\00_resource_generation** and choose the manifest file generated using TNG Manifest Generation Notebook.

Name	Value	Type	Location
profile	<not set>	None	None
access_key	*****5S5B	shared-credentials-file	
secret_key	*****pwV0	shared-credentials-file	
region	us-east-2	config-file	~/.aws/config

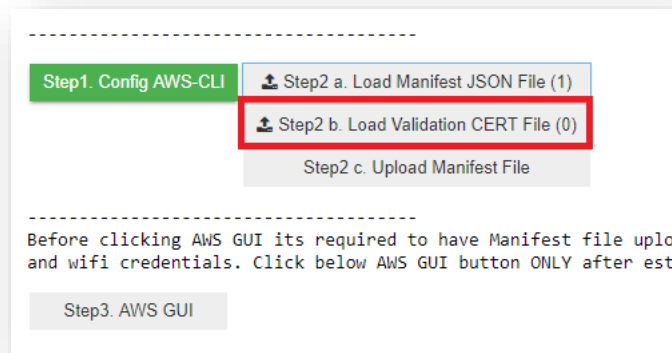
Step1. Config AWS-CLI

Step2 a. Load Manifest JSON File (0)

Step2 b. Load Validation CERT File (0)

Step2 c. Upload Manifest File

- Press **"Load Validation CERT File"** button, it will open file explorer window, there you need to navigate **TrustnGO\00_resource_generation** and choose the validation certificate file generated using TNG Manifest Generation Notebook.



7. Press “**Upload Manifest File**” button to upload manifest file to aws cloud.



Once it is successfully uploaded, the result will be as shown below.

```
number of certificates: 1

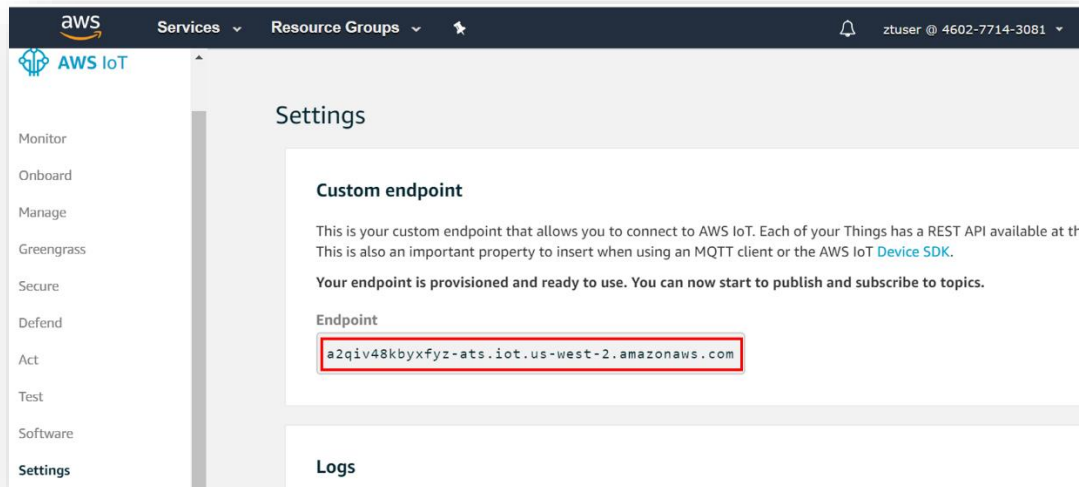
Loading the manifest item
uniqueId: 01236c7dc62e6dc001
About to try certificate import
Response: {'ResponseMetadata': {'RequestId': '8a4596b5-b71c-49ad-a453-bedd44e281c8', 'HTTPStatusCode': 200, 'HTTPHeaders': {'date': 'Sat, 09 Nov 2019 13:29:31 GMT', 'content-type': 'application/json', 'content-length': '209', 'connection': 'keep-alive', 'x-amzn-requestid': '8a4596b5-b71c-49ad-a453-bedd44e281c8', 'access-control-allow-origin': '*', 'x-amz-apigw-id': 'C5LRQEEEXPHcF djw=', 'x-amzn-trace-id': 'Root=1-5dc6bf3b-a36debb6c3255f8972bddcbb'}, 'RetryAttempts': 0}, 'certificateArn': 'arn:aws:iot:us-west-2:460277143081:cert/461095d8771f95db78d442d2015415804e3093bcae3fd56d78a486aa5b7b59de', 'certificateId': '461095d8771f95db78d442d2015415804e3093bcae3fd56d78a486aa5b7b59de'}
Certificate import complete - returning
MANIFEST_IMPORT SUCCESS arn:aws:iot:us-west-2:460277143081:cert/461095d8771f95db78d442d2015415804e3093bcae3fd56d78a486aa5b7b59de
arn:aws:iot:us-west-2:460277143081:thing/01236c7dc62e6dc001
number of thingIds to check: 1

Checking the manifest_item
uniqueId: 01236c7dc62e6dc001
Manifest was loaded successfully
```

Note down your unique id from the manifest upload log which is the **Thing name** of your device and is needed for AWS GUI step.

Once this step is completed, manifest file is successfully uploaded to AWS IoT. Continue to next steps when connecting to AWS using TNGTLS device.

8. Login to the AWS account, From Services select IoT core. In the IoT core, Click setting and note down your endpoint as highlighted below. This endpoint is needed for the embedded c project.



NOTE: Make sure that you executed C project successfully before executing the next step in the Jupyter notebook. To execute C project, refer "Running AWS IoT example on Embedded platform" next section.

AWS GUI:

Code block of this step generates "**AWS GUI**" button.

```
-----  
Before clicking AWS GUI its required to have Manifest file up  
and wifi credentials. Click below AWS GUI button ONLY after e
```

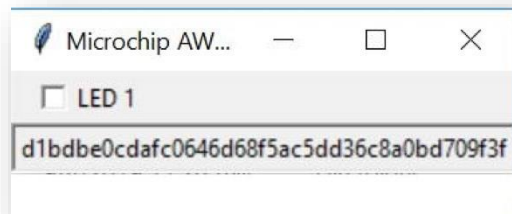
Step3. AWS GUI

Clicking the button asks for the Thing name. Enter the number which we noted from the previous step and press enter.

Enter Thing Name:

it will create a very basic graphical interface that will display the device ID and will allow to switch the board LED status.

Below screenshot display the graphical interface



Using this interface, Custom PKI CryptoAuth Trust Platform can able to communicate with AWS IoT. Upon successful communication, you have now a device connected to AWS IoT through a secure TLS session with a custom PKI using a Crypto Trust Platform.

4.2 Running AWS IoT example on Embedded platform

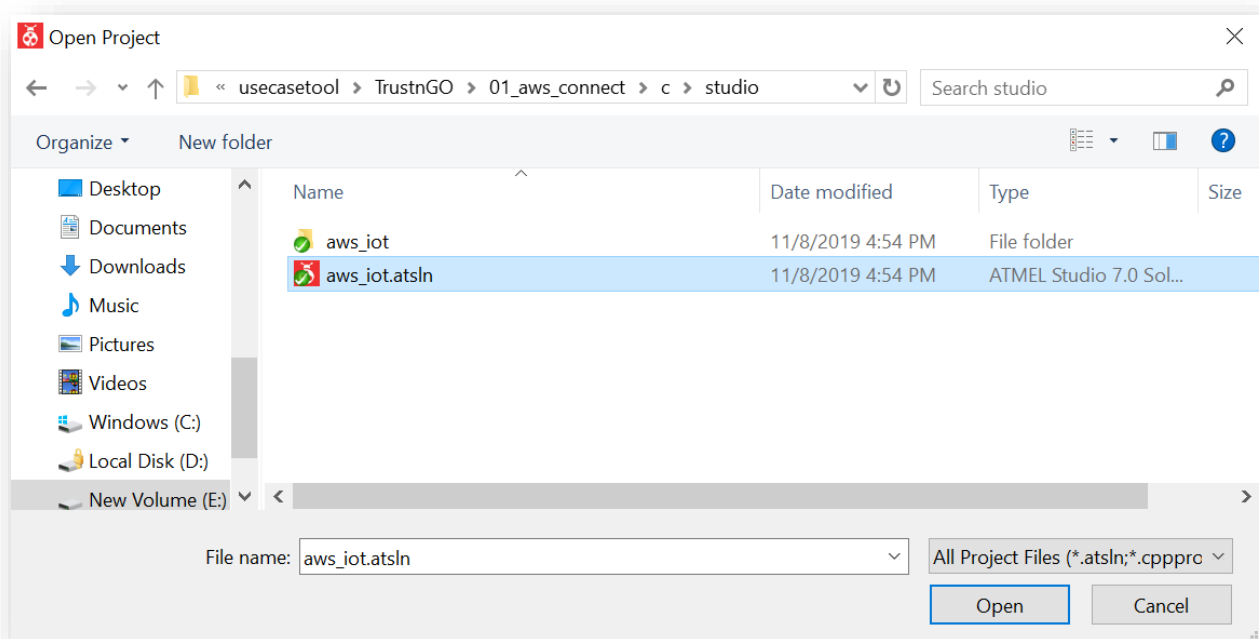
Once the resources are generated, both Atmel Studio and MPLAB projects provided can be used to run the use case on CryptoAuth Trust Platform.

This project can configure the Wi-Fi credentials, establish a TLS connection, subscribe to MQTT topic and establish communication but not upload manifest file to AWS IoT. It is required to use the AWS IoT Jupyter notebook to upload manifest file.

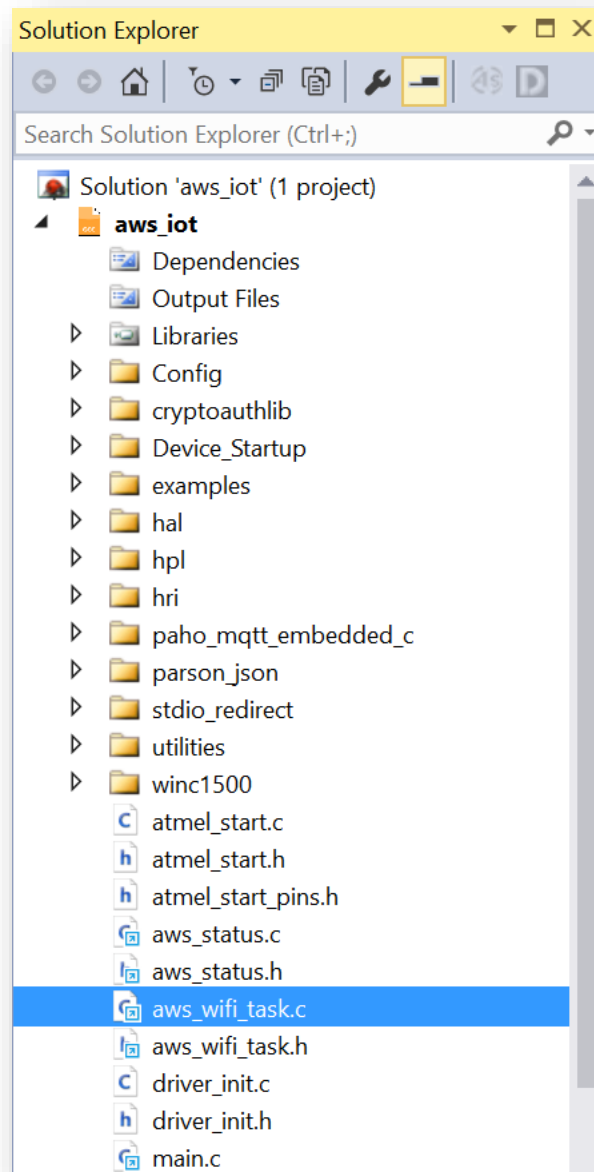
Once the manifest file uploaded to AWS IoT then these embedded projects can be executed.

4.2.1 Atmel Studio:

1. Open **aws_iot.atsln** project by navigating
TrustnGO\01_aws_connect\c\studio\ aws_iot.atsln



2. In the project navigate to **aws_iot -> aws_wifi_task.c** file



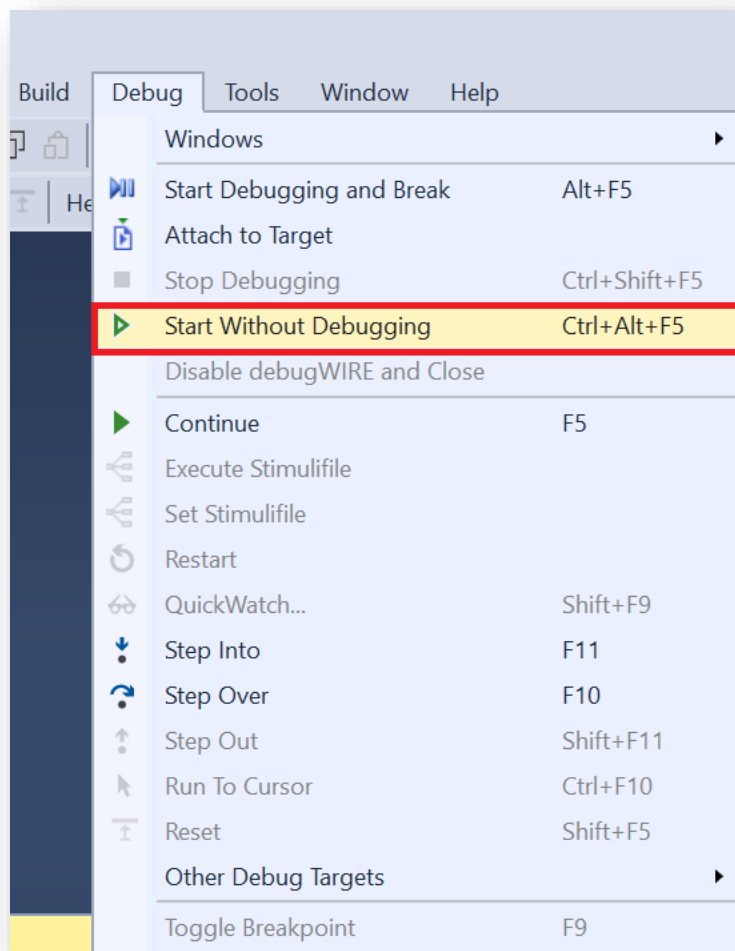
update the following constants before building the project:

- MAIN_WLAN_SSID
- MAIN_WLAN_PSK
- AWS_HOST_ENDPOINT

```
#define MAIN_WLAN_SSID      "xxxxxx"
#define MAIN_WLAN_AUTH     M2M_WIFI_SEC_WPA_PSK
#define MAIN_WLAN_PSK      "xxxxxxxxxx"

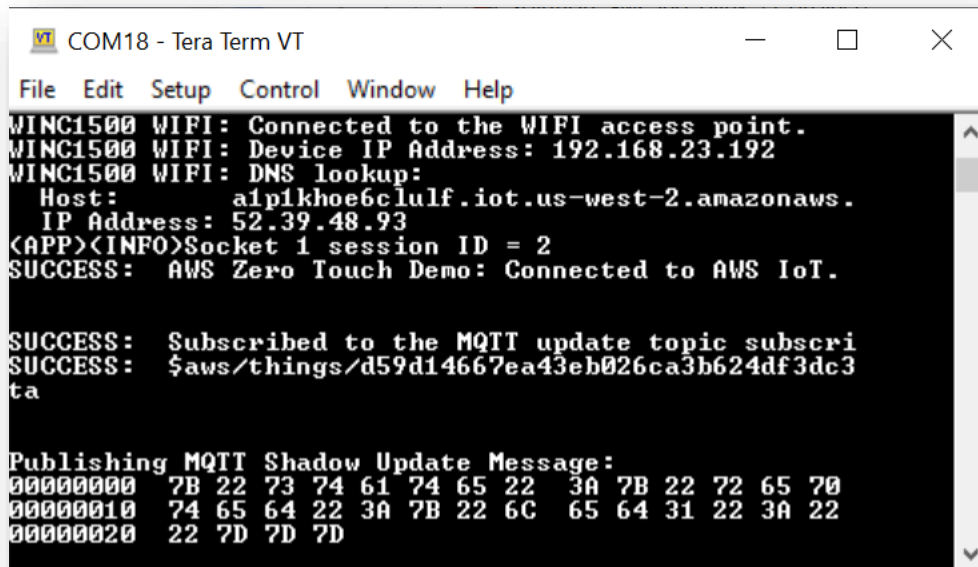
#define AWS_HOST_ENDPOINT  "xxxxxxx.iot.us-west-2.amazonaws.com"
```

3. Program the CryptoAuth Trust Platform by navigating to **Debug -> Start Without Debugging**



This step may take some time, wait for Atmel Studio to compile and program the device.

Once the programming is done, reset the hardware (press the reset button) and view the Console messages by using applications like 'Tera Term'. Open the application with the COM related to CryptoAuth Trust Platform with 115200-8-N-1 settings.



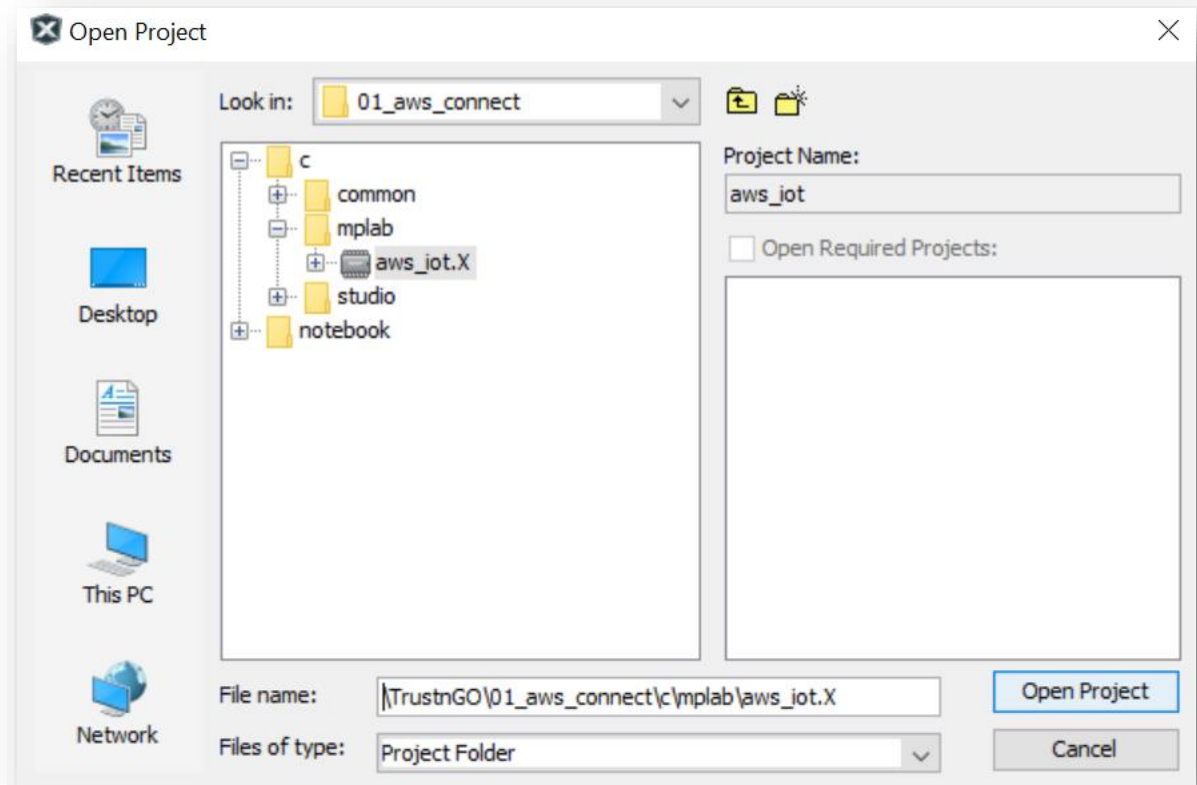
The screenshot shows a Tera Term VT window titled 'COM18 - Tera Term VT'. The window has a menu bar with 'File', 'Edit', 'Setup', 'Control', 'Window', and 'Help'. The main area displays the following text:

```
WINC1500 WIFI: Connected to the WIFI access point.  
WINC1500 WIFI: Device IP Address: 192.168.23.192  
WINC1500 WIFI: DNS lookup:  
  Host:      alpikhoe6clulf.iot.us-west-2.amazonaws.  
  IP Address: 52.39.48.93  
<APP><INFO>Socket 1 session ID = 2  
SUCCESS:  AWS Zero Touch Demo: Connected to AWS IoT.  
  
SUCCESS:  Subscribed to the MQTT update topic subscri  
SUCCESS:  $aws/things/d59d14667ea43eb026ca3b624df3dc3  
ta  
  
Publishing MQTT Shadow Update Message:  
00000000  7B 22 73 74 61 74 65 22  3A 7B 22 72 65 70  
00000010  74 65 64 22 3A 7B 22 6C  65 64 31 22 3A 22  
00000020  22 7D 7D 7D
```

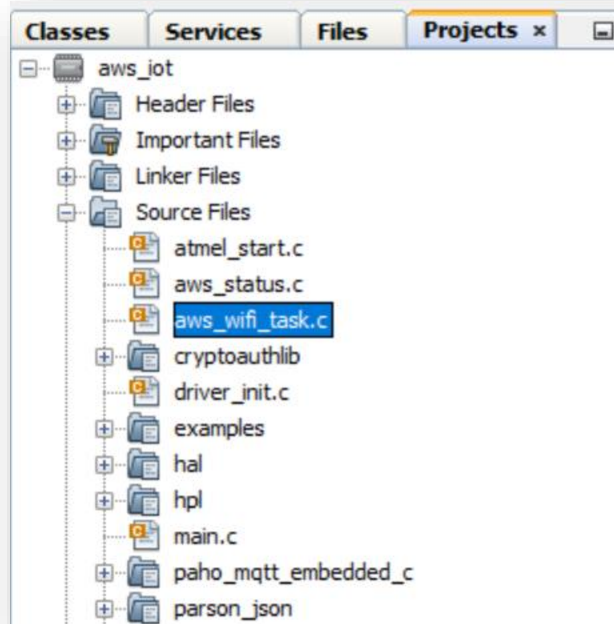
Once successfully programmed the CryptoAuth Trust Platform, now we can run the last step in the Jupyter Notebook. Just navigate to previous section 4.1 to run the last step (AWS GUI) in the Jupyter Notebook.

4.2.2 MPLAB:

1. Open **aws_iot.X** project by navigating to MPLAB -> File -> Open Project -> **TrustnGO\01_aws_connect\c\mplab \aws_iot.X**



2. Open **aws_wifi_task.c** file by navigating to **aws_iot -> Source Files -> aws_wifi_task.c**

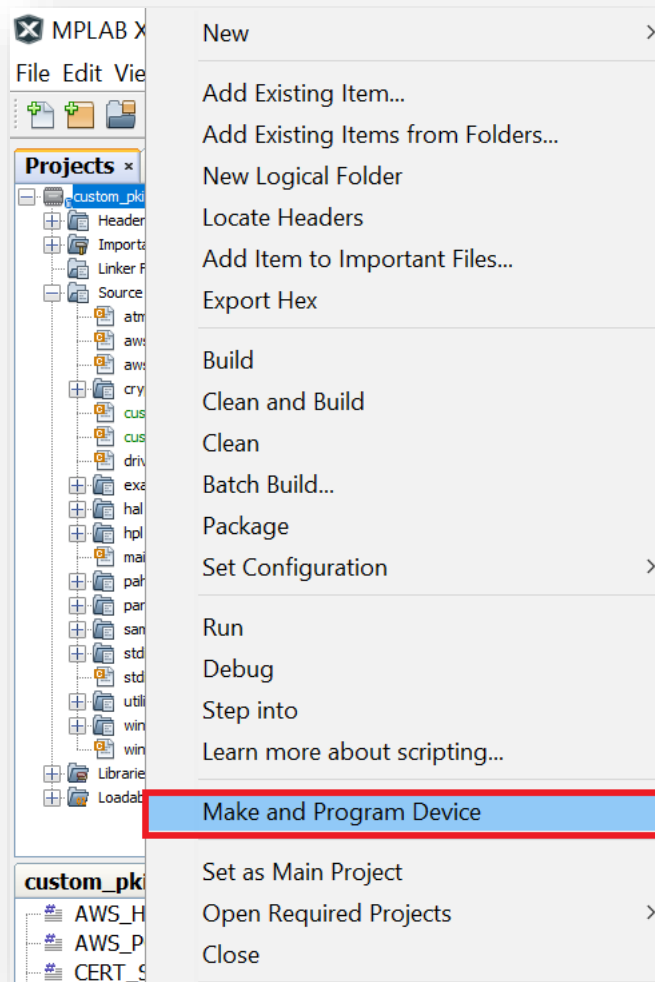


update the following constants before building the project:

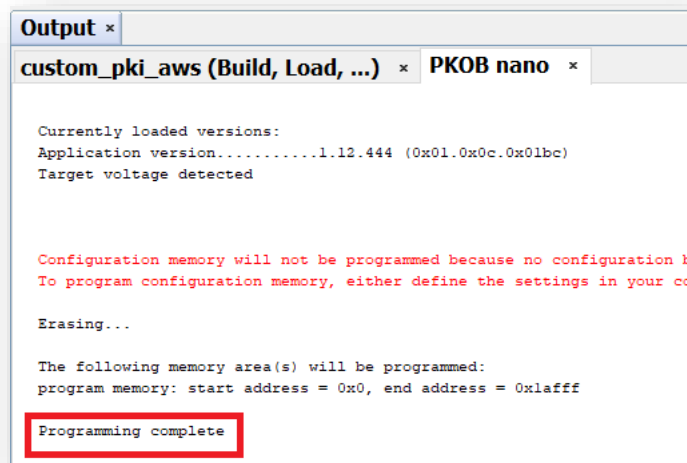
- MAIN_WLAN_SSID
- MAIN_WLAN_PSK
- AWS_HOST_ENDPOINT

```
49
50 #define MAIN_WLAN_SSID      "XXXXXXX"
51 #define MAIN_WLAN_AUTH     M2M_WIFI_SEC_WPA_PSK
52 #define MAIN_WLAN_PSK      "XXXXXXX"
53
54 #define AWS_HOST_ENDPOINT   "XXXXXXXXX.iot.us-west-2.amazonaws.com"
55
```

3. Program the CryptoAuth Trust platform by navigating to **aws_iot -> Make and Program Device**



This step may take some time, wait for MPLAB to program the device. Once it is done programming you will see "**Programming complete**" message in Output Window.



```
Output x
custom_pki_aws (Build, Load, ...) x PKOB nano x

Currently loaded versions:
Application version.....1.12.444 (0x01.0x0c.0x01bc)
Target voltage detected

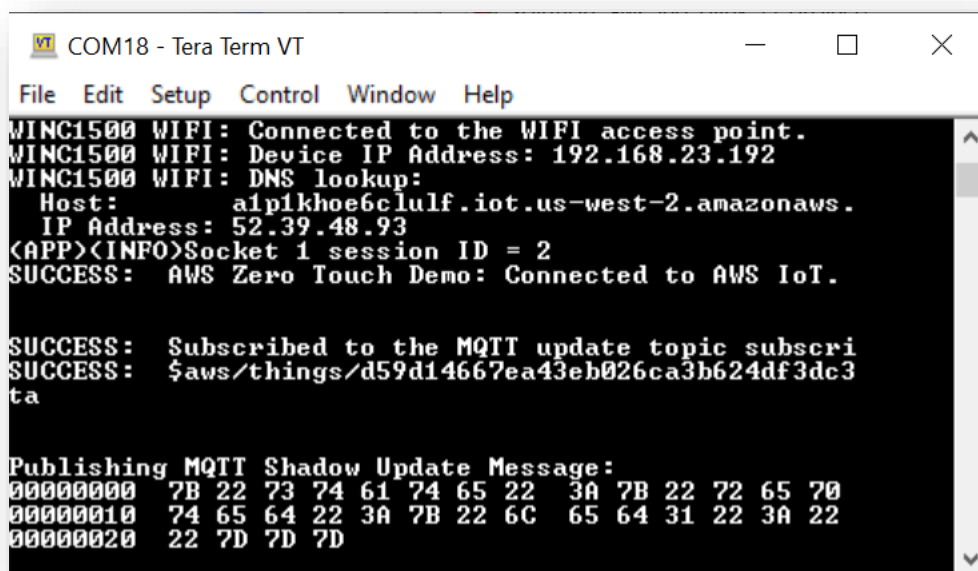
Configuration memory will not be programmed because no configuration b
To program configuration memory, either define the settings in your co

Erasing...

The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0xffff

Programming complete
```

Once the programming is done, reset the hardware (press the reset button) and view the Console messages by using applications like 'Tera Term'. Open the application with the COM related to CryptoAuth Trust Platform with 115200-8-N-1 settings.



```
COM18 - Tera Term VT
File Edit Setup Control Window Help
WINC1500 WIFI: Connected to the WIFI access point.
WINC1500 WIFI: Device IP Address: 192.168.23.192
WINC1500 WIFI: DNS lookup:
Host: aipikho6clulf.iot.us-west-2.amazonaws.
IP Address: 52.39.48.93
(APP)<INFO>Socket 1 session ID = 2
SUCCESS: AWS Zero Touch Demo: Connected to AWS IoT.

SUCCESS: Subscribed to the MQTT update topic subscri
SUCCESS: $aws/things/d59d14667ea43eb026ca3b624df3dc3
ta

Publishing MQTT Shadow Update Message:
00000000 7B 22 73 74 61 74 65 22 3A 7B 22 72 65 70
00000010 74 65 64 22 3A 7B 22 6C 65 64 31 22 3A 22
00000020 22 7D 7D 7D
```

Once successfully programmed the CryptoAuth Trust Platform, now we can run the last step in the Jupyter Notebook. Just navigate to previous section 4.1 to run the last step (AWS GUI) in the Jupyter Notebook

4.3 CryptoAuth TrustPlatform Factory reset

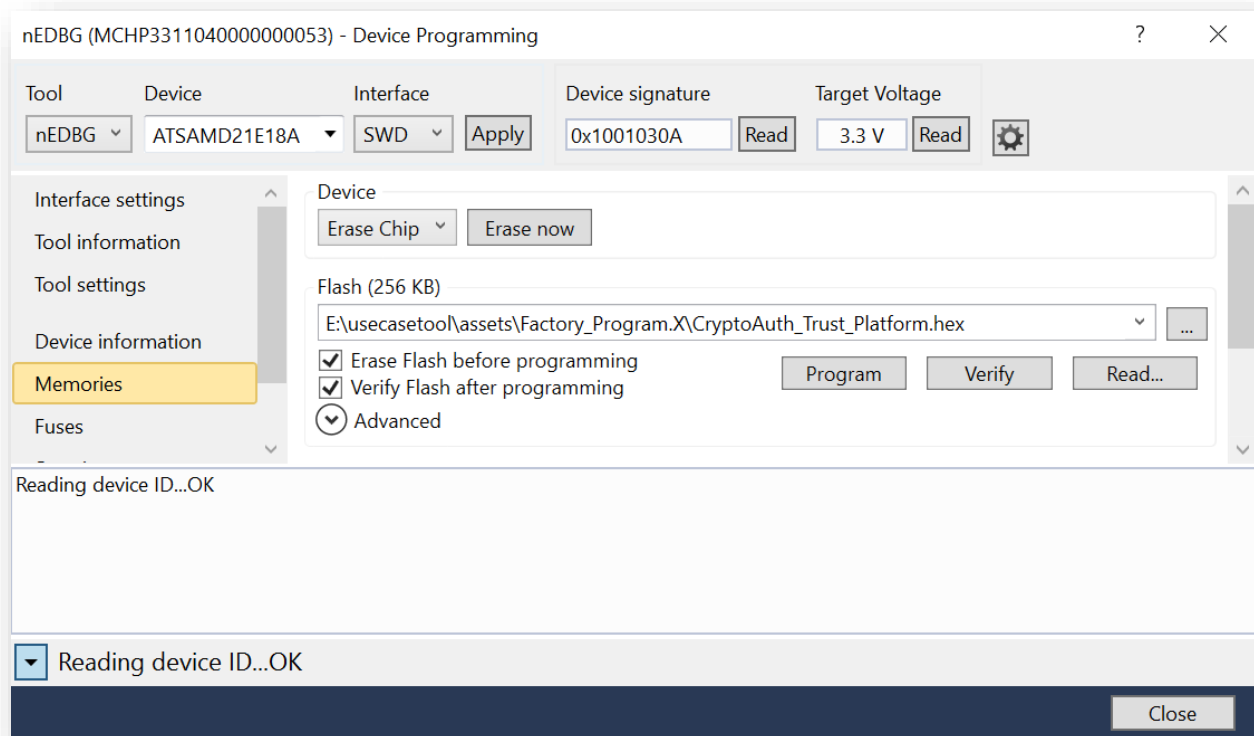
Once any of the embedded project is loaded to CryptoAuth TrustPlatform, the default program that enables interaction with TrustPlatform tools will be erased.

Before using the Platform with any other notebook or tools on PC, its required to reprogram the default .hex file. Default hex file is available at

assets\Factory_Program.X\CryptoAuth_Trust_Platform.hex

To reprogram using Atmel Studio:

1. Navigate to AtmelStudio -> Tools -> Device Programming
2. Select Tool as nEDBG and Apply
3. Go to Memories and navigate to above path under Flash dropdown
4. Check both Erase Flash and Verify Flash
5. Click on Program



To reprogram using MPLAB:

1. Open **assets\Factory_Program.X** project in MPLAB IDE
2. Program the Crypto Trust platform by navigating to
CryptoAuth_Trust_Platform_Factory_Program -> Make and Program Device

Now, Crypto Trust Platform contains factory programmed application that enables interactions with Notebooks and/or PC tools.

The Microchip Web Site

Microchip provides online support via our web site at <http://www.microchip.com/>. This web site is used as

a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at <http://www.microchip.com/>. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support.

Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: <http://www.microchip.com/support>

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY

OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE.

Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KeeLoq,

Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB,

OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, SAM-BA, SpyNIC, SST,

SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight

Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming,

ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient

Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE,

Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2018, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN:

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California

and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.



Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com Atlanta Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 Austin, TX Tel: 512-257-3370 Boston Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 Chicago Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 Dallas Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 Detroit Novi, MI Tel: 248-848-4000 Houston, TX Tel: 281-894-5983 Indianapolis Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 Los Angeles Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 Raleigh, NC Tel: 919-844-7510 New York, NY Tel: 631-435-6000 San Jose, CA Tel: 408-735-9110 Tel: 408-436-4270 Canada - Toronto Tel: 905-695-1980 Fax: 905-695-2078	Australia - Sydney Tel: 61-2-9868-6733 China - Beijing Tel: 86-10-8569-7000 China - Chengdu Tel: 86-28-8665-5511 China - Chongqing Tel: 86-23-8980-9588 China - Dongguan Tel: 86-769-8702-9880 China - Guangzhou Tel: 86-20-8755-8029 China - Hangzhou Tel: 86-571-8792-8115 China - Hong Kong SAR Tel: 852-2943-5100 China - Nanjing Tel: 86-25-8473-2460 China - Qingdao Tel: 86-532-8502-7355 China - Shanghai Tel: 86-21-3326-8000 China - Shenyang Tel: 86-24-2334-2829 China - Shenzhen Tel: 86-755-8864-2200 China - Suzhou Tel: 86-186-6233-1526 China - Wuhan Tel: 86-27-5980-5300 China - Xian Tel: 86-29-8833-7252 China - Xiamen Tel: 86-592-2388138 China - Zhuhai Tel: 86-756-3210040	India - Bangalore Tel: 91-80-3090-4444 India - New Delhi Tel: 91-11-4160-8631 India - Pune Tel: 91-20-4121-0141 Japan - Osaka Tel: 81-6-6152-7160 Japan - Tokyo Tel: 81-3-6880-3770 Korea - Daegu Tel: 82-53-744-4301 Korea - Seoul Tel: 82-2-554-7200 Malaysia - Kuala Lumpur Tel: 60-3-7651-7906 Malaysia - Penang Tel: 60-4-227-8870 Philippines - Manila Tel: 63-2-634-9065 Singapore Tel: 65-6334-8870 Taiwan - Hsin Chu Tel: 886-3-577-8366 Taiwan - Kaohsiung Tel: 886-7-213-7830 Taiwan - Taipei Tel: 886-2-2508-8600 Thailand - Bangkok Tel: 66-2-694-1351 Vietnam - Ho Chi Minh Tel: 84-28-5448-2100	Austria - Wels Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 Denmark - Copenhagen Tel: 45-4450-2828 Fax: 45-4485-2829 Finland - Espoo Tel: 358-9-4520-820 France - Paris Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 France - Saint Cloud Tel: 33-1-30-60-70-00 Germany - Garching Tel: 49-8931-9700 Germany - Haan Tel: 49-2129-3766400 Germany - Heilbronn Tel: 49-7131-67-3636 Germany - Karlsruhe Tel: 49-721-625370 Germany - Munich Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 Germany - Rosenheim Tel: 49-8031-354-560 Israel - Ra'anana Tel: 972-9-744-7705 Italy - Milan Tel: 39-0331-742611 Fax: 39-0331-466781 Italy - Padova Tel: 39-049-7625286 Netherlands - Drunen Tel: 31-416-690399 Fax: 31-416-690340 Norway - Trondheim Tel: 47-7289-7561 Poland - Warsaw Tel: 48-22-3325737 Romania - Bucharest Tel: 40-21-407-87-50 Spain - Madrid Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 Sweden - Gothenberg Tel: 46-31-704-60-40 Sweden - Stockholm Tel: 46-8-5090-4654 UK - Wokingham Tel: 44-118-921-5800 Fax: 44-118-921-5820