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

Table of Contents

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.

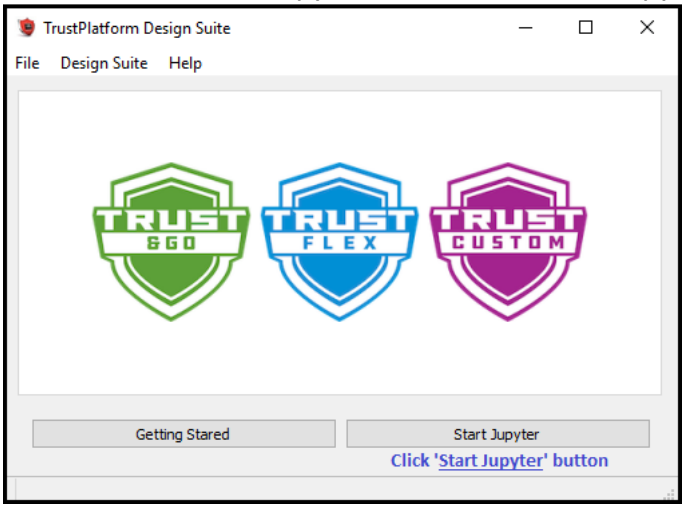
1.1 Getting started with Jupyter Notebook Tutorials

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.

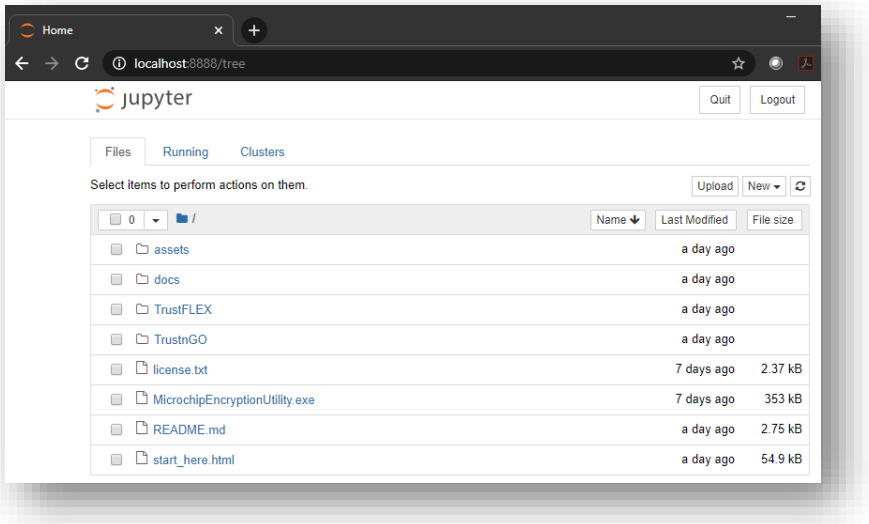
Jupyter notebooks are mainly used to explain/evaluate code in an interactive way.

1.1.1 Starting Jupyter Notebook

Jupyter notebook can be launched from Trust Platform GUI Main window. Run START -> Trust Platform x.x.x icon. Click on 'Start Jupyter' button to launch Jupyter local server.



Clicking on Start Jupyter should be web browser tab like below,



1.2 Jupyter Notebook Basics

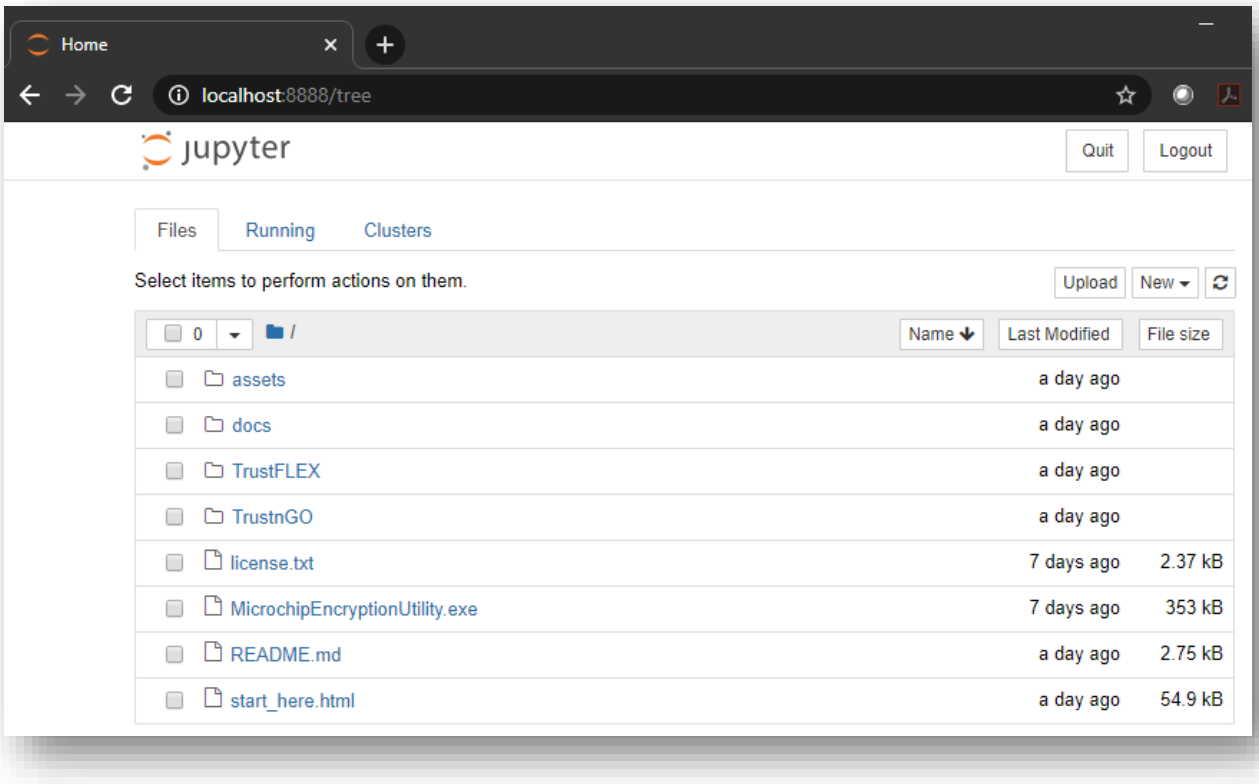
It is recommended to become familiar with Jupyter basic concepts with the online documentation, https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html

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.

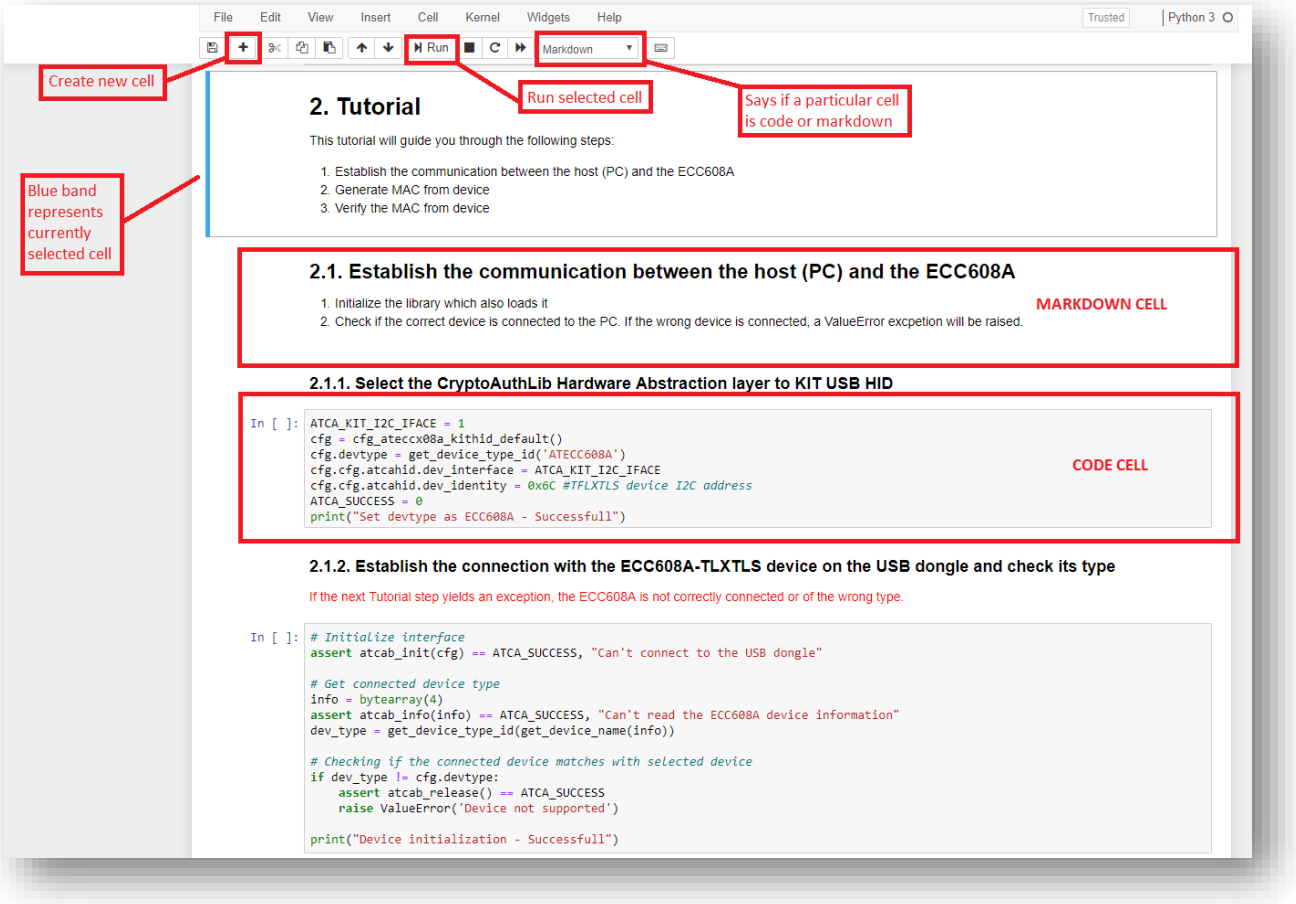


1.3 Introduction to Jupyter Notebook GUI.

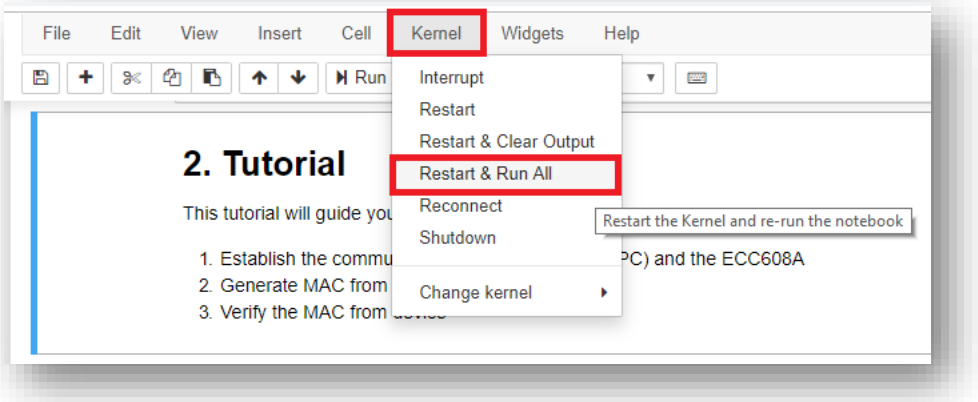
Jupyter Notebooks contain cells where you can either write code or markdown text. Notebooks contain multiple cells, some set as code and others markdown. Code cells contain code that can be executed live, and markdown contains text and images to explain the code.

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**.



To run all cells in sequence.



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.

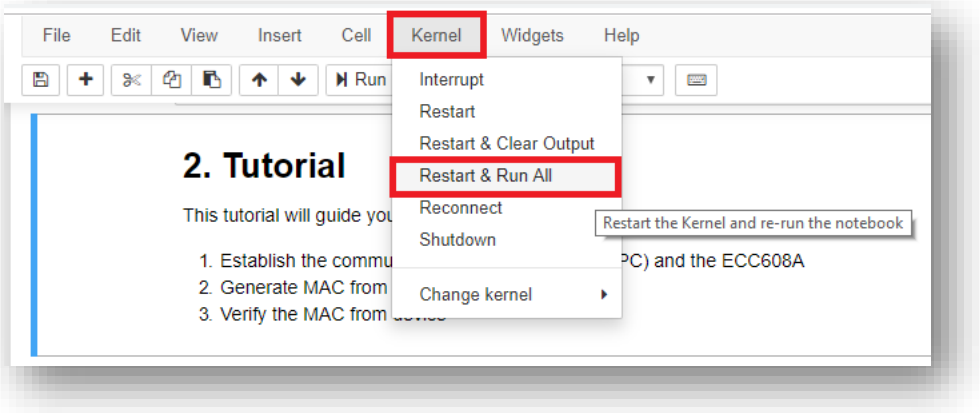
Trust&GO devices come with pre-programmed certificates in slots 10, 11 and 12, also slots 0-4 have pre-generated private keys, other than the previously mentioned slots all the other slots are locked.

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.

Within the Jupyter Dashboard, navigate **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:
Certificate:
   Data:
      Version: 3 (0x2)
      Serial Number:
         77:d3:6d:95:6e:c8:ae:62:05:e5:8e:3a:cb:98:5a:81
      Signature Algorithm: ecdsa-with-SHA256
      Issuer: O = Microchip Technology Inc, CN = Crypto Authentication Root CA 002
      Validity
         Not Before: Nov  8 19:12:19 2018 GMT
         Not After : Nov  8 19:12:19 2058 GMT
      Subject: O = Microchip Technology Inc, CN = Crypto Authentication Root CA 002
      Subject Public Key Info:
         Public Key Algorithm: id-ecPublicKey
            Public-Key: (256 bit)
            pub:
               04:bd:54:e6:6d:e3:87:54:84:00:6b:53:ae:15:80:
               d5:0a:a0:69:e7:8a:df:55:78:d8:5c:e2:d5:4d:d5:
               b8:30:29:6b:ff:dd:6e:6f:72:56:fb:d9:9e:f1:a1:
               16:b1:1d:33:ad:49:10:3a:a1:85:87:39:dc:fa:e4:
               37:e1:9d:63:4e
            ASN1 OID: prime256v1
            NIST CURVE: P-256
      X509v3 extensions:
         X509v3 Subject Key Identifier:
            7A:ED:7D:6D:C6:B7:78:9D:B2:38:01:A5:E8:4A:8C:B0:A4:0E:2A:8C
```

```
            X509v3 Authority Key Identifier:
               keyid:7A:ED:7D:6D:C6:B7:78:9D:B2:38:01:A5:E8:4A:8C:B0:A4:0E:2A:8C

            X509v3 Basic Constraints: critical
               CA:TRUE
      Signature Algorithm: ecdsa-with-SHA256
           30:45:02:21:00:a1:dc:63:45:90:ec:81:9e:e1:de:5b:81:12:
           65:51:ad:d4:c2:c4:f8:e5:95:28:2e:e0:4b:e7:68:ec:7c:02:
           73:02:20:3e:6b:a7:4e:9e:4c:0a:d6:8c:24:b0:fb:2e:e7:93:
           d2:e6:be:94:65:ca:15:d0:ea:5b:c8:7f:55:79:99:5c:ad
-----BEGIN CERTIFICATE-----
MIIB8TCCAZegAwIBAgIQd9NtlW7IrmIF5Y46y5hagTAKBggqhkjOPQQDAjBPMSEw
HwYDVQQKDBhNaWNyb2NoaXAgVGVjaG5vbG9neSBJbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXRoZW50aWNhdGlvbiBSb290IENBIDAwMjAgFw0xODExMDgxOTEyMTlaGA8y
MDU4MTEwODE5MTIxOVowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSowKAYDVQQDDCFDcnlwdG8gQXV0aGVudGljYXRpb24gUm9vdCBDQSAwMDIw
WTATBgcqhkjOPQIBBggqhkjOPQMBBwNCAAS9VOZt44dUhABrU64VgNUKoGnnit9V
eNhc4tVN1bgwKWv/3W5vclb72Z7xoRaxHTOtSRA6oYWHOdz65DfhnWNOo1MwUTAd
BgNVHQ4EFgQUeu19bca3eJ2yOAGl6EqMsKQOKowwHwYDVR0jBBgwFoAUeu19bca3
eJ2yOAGl6EqMsKQOKowwDwYDVR0TAQH/BAUwAwEB/zAKBggqhkjOPQQDAgNIADBF
AiEAodxjRZDsgZ7h3luBEmVRrdTCxPjllSgu4EvnaOx8AnMCID5rp06eTArWjCSw
+y7nk9LmvpRlyhXQ6lvIf1V5mVyt
-----END CERTIFICATE-----

Validate Root Certificate:
OK

TNG Signer Certificate:
Certificate:
   Data:
      Version: 3 (0x2)
      Serial Number:
         7c:31:16:e0:34:06:5b:25:e0:53:bf:b0:9e:26:1a:66
      Signature Algorithm: ecdsa-with-SHA256
      Issuer: O = Microchip Technology Inc, CN = Crypto Authentication Root CA 002
      Validity
         Not Before: Dec 14 19:00:00 2018 GMT
         Not After : Dec 14 19:00:00 2049 GMT
      Subject: O = Microchip Technology Inc, CN = Crypto Authentication Signer F640
      Subject Public Key Info:
         Public Key Algorithm: id-ecPublicKey
            Public-Key: (256 bit)
            pub:
               04:0e:7f:32:95:f2:eb:46:40:f4:aa:39:4c:e5:e5:
               20:d7:d9:b5:0b:b9:54:f5:c3:f8:bb:6f:3f:2c:18:
               5a:4d:b9:90:91:bc:af:49:ea:06:fc:4b:cc:5c:2a:
               27:c8:d0:02:2b:1a:9b:f0:1b:f4:46:dd:9a:c5:62:
               30:4c:7a:38:76
            ASN1 OID: prime256v1
            NIST CURVE: P-256
      X509v3 extensions:
         X509v3 Key Usage: critical
            Digital Signature, Certificate Sign, CRL Sign
```

```
            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
            X509v3 Subject Key Identifier:
                EB:62:03:2B:8C:81:58:16:42:9A:1C:AB:F1:BE:8C:22:1E:B7:A6:C9
            X509v3 Authority Key Identifier:
                keyid:7A:ED:7D:6D:C6:B7:78:9D:B2:38:01:A5:E8:4A:8C:B0:A4:0E:2A:8C

    Signature Algorithm: ecdsa-with-SHA256
         30:46:02:21:00:e0:78:ed:06:27:15:f0:31:32:f0:ef:31:97:
         22:a5:49:e2:e1:86:28:73:b5:f2:b7:5f:28:3e:e3:7d:13:6e:
         bc:02:21:00:df:68:76:be:05:22:ae:7f:e9:15:84:82:fa:08:
         45:8d:da:9c:f3:08:17:2f:d6:60:74:fc:11:90:7a:30:47:7b
-----BEGIN CERTIFICATE-----
MIICBTCCAaqgAwIBAgIQfDEW4DQGWyXgU7+wniYaZjAKBggqhkjOPQQDAjBPMSEw
HwYDVQQKDBhNaWNyb2NoaXAgVGVjaG5vbG9neSBJbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXRoZW50aWNhdGlvbiBSb290IENBIDAwMjAgFw0xODEyMTQxOTAwMDBaGA8y
MDQ5MTIxNDE5MDAwMFowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSowKAYDVQQDDCFDcnlwdG8gQXV0aGVudGljYXRpb24gU2lnbmVyIEY2NDAw
WTATBgcqhkjOPQIBBggqhkjOPQMBBwNCAAQOfzKV8utGQPSqOUzl5SDX2bULuVT1
w/i7bz8sGFpNuZCRvK9J6gb8S8xcKifI0AIrGpvwG/RG3ZrFYjBMejh2o2YwZDAO
BgNVHQ8BAf8EBAMCAYYwEgYDVR0TAQH/BAgwBgEB/wIBADAdBgNVHQ4EFgQU62ID
K4yBWBZCmhyr8b6MIh63pskwHwYDVR0jBBgwFoAUeu19bca3eJ2yOAGl6EqMsKQO
KowwCgYIKoZIzj0EAwIDSQAwRgIhAOB47QYnFfAxMvDvMZcipUni4YYoc7Xyt18o
PuN9E268AiEA32h2vgUirn/pFYSC+ghFjdqc8wgXL9ZgdPwRkHowR3s=
-----END CERTIFICATE-----

Validate Signer Certificate:
OK

TNG Device Certificate:
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            71:4f:f4:a3:24:1b:e8:18:dc:6d:8b:b0:c5:53:aa:6f
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: O = Microchip Technology Inc, CN = Crypto Authentication Signer F640
        Validity
            Not Before: Aug  1 00:00:00 2019 GMT
            Not After : Aug  1 00:00:00 2047 GMT
        Subject: O = Microchip Technology Inc, CN = 0123CEDC3CD337AA01 ATECC
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    04:97:24:09:48:8c:20:89:19:6f:24:47:b0:8a:4d:
                    90:cc:74:d6:cc:14:37:60:b0:aa:b5:76:4d:cd:bf:
                    03:f6:06:e9:46:80:18:e5:2c:9f:75:4a:3b:c8:3f:
                    17:2d:b5:24:d7:5b:c4:ba:f7:34:92:f3:a3:45:58:
                    c2:eb:ae:69:34
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
```

```
            X509v3 Basic Constraints: critical
                CA:FALSE
            X509v3 Key Usage: critical
                Digital Signature, Key Agreement
            X509v3 Subject Key Identifier:
                09:61:2F:3F:E6:E6:AE:20:35:71:7D:CB:C7:A7:49:AC:63:E5:FC:66
            X509v3 Authority Key Identifier:
                keyid:EB:62:03:2B:8C:81:58:16:42:9A:1C:AB:F1:BE:8C:22:1E:B7:A6:C9

    Signature Algorithm: ecdsa-with-SHA256
        30:46:02:21:00:a1:52:e2:bb:be:01:1d:91:8e:12:0c:9b:a6:
        67:67:6a:55:78:e9:66:6a:72:84:ba:17:2f:ce:9b:72:f7:6c:
        3b:02:21:00:c7:29:b1:2c:5d:43:cb:7c:f2:67:7d:12:c8:59:
        82:eb:4f:78:0e:49:8a:16:11:62:55:2a:15:45:f8:2a:f4:a5
-----BEGIN CERTIFICATE-----
MIIB9jCCAZugAwIBAgIQcU/0oyQb6BjcbYuwxVOqbzAKBggqhkjOPQQDAjBPMSEw
HwYDVQQKDBhNaWNyb2NoaXAgVGVjaG5vbG9neSBJbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXRoZW50aWNhdGlvbiBTaWduZXIgRjY0MDAgFw0xOTA4MDEwMDAwMDBaGA8y
MDQ3MDgwMTAwMDAwMFowRjEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSEwHwYDVQQDDBgwMTIzQ0VEQzNDRDMzN0FBMDEgQVRFQ0MwWTATBgcqhkjO
PQIBBggqhkjOPQMBBwNCAASXJAlIjCCJGW8kR7CKTZDMdNbMFDdgsKq1dk3NvwP2
BulGgBjlLJ91SjvIPxcttSTXW8S69zSS86NFWMLrrmk0o2AwXjAMBgNVHRMBAf8E
AjAAMA4GA1UdDwEB/wQEAwIDiDAdBgNVHQ4EFgQUCWEvP+bmriA1cX3Lx6dJrGPl
/GYwHwYDVR0jBBgwFoAU62IDK4yBWBZCmhyr8b6MIh63pskwCgYIKoZIzj0EAwID
SQAwRgIhAKFS4ru+AR2RjhIMm6ZnZ2pVeOlmanKEuhcvzpty92w7AiEAxymxLF1D
y3zyZ30SyFmC6094DkmKFhFiVSoVRfgq9KU=
-----END CERTIFICATE-----

Validate Device Certificate:
OK

------------------------------------------------------
Generated the manifest file 0123cedc3cd337aa01_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.

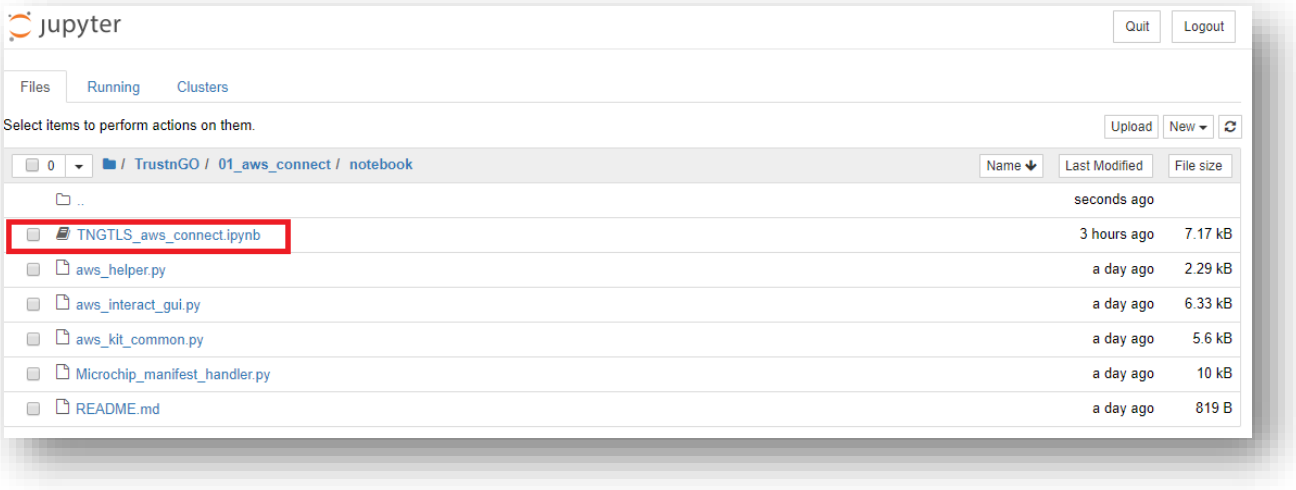
Here are the steps that will be required to complete this Tutorial:

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

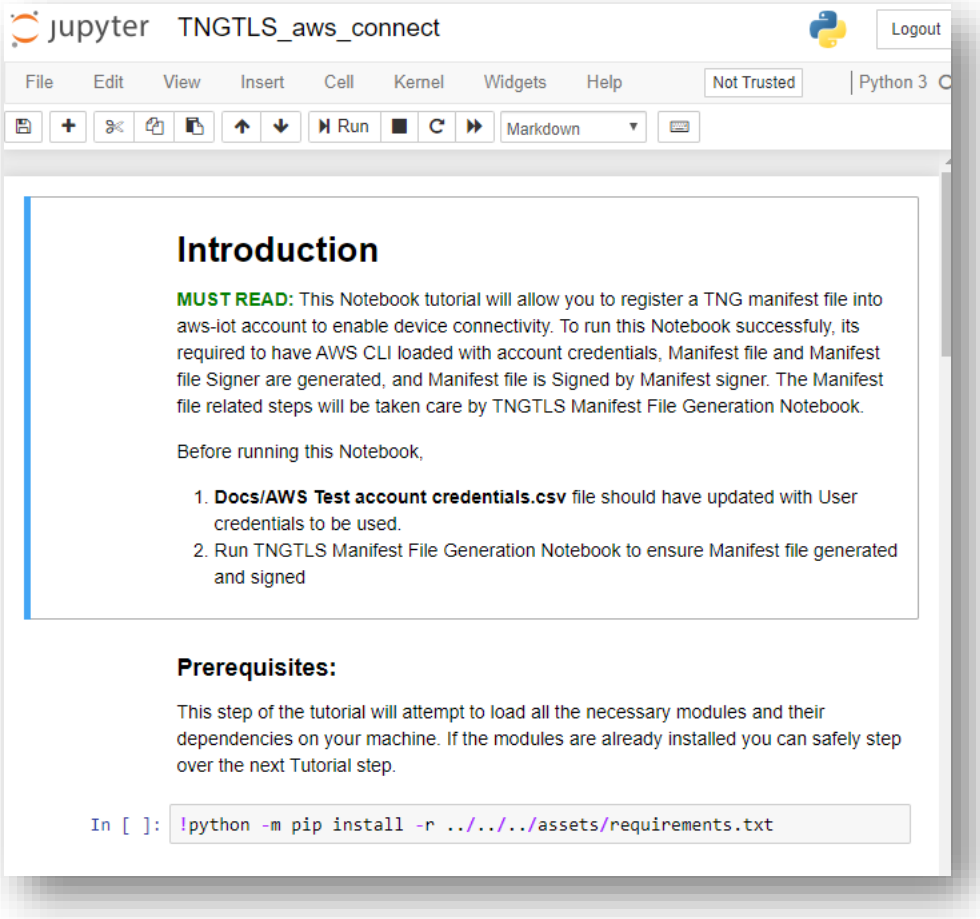
4.1 Running AWS IoT example on Jupyter Notebook

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.

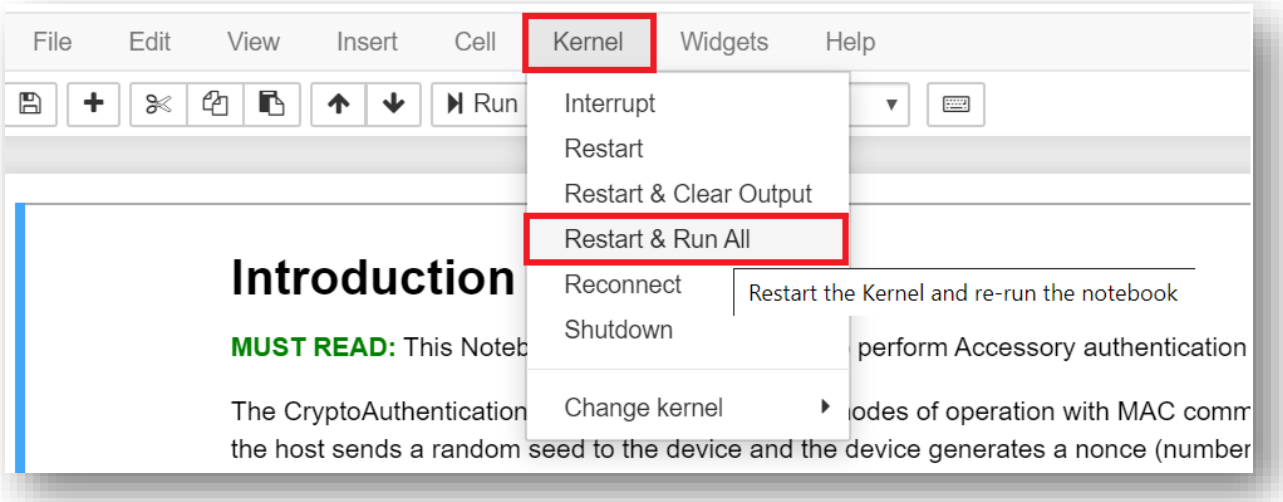
1. From the Jupyter Home page, navigate to **TrustnGO\01_aws_connect\notebook\TNGTLS_aws_connect.ipynb** notebook file and open it.



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

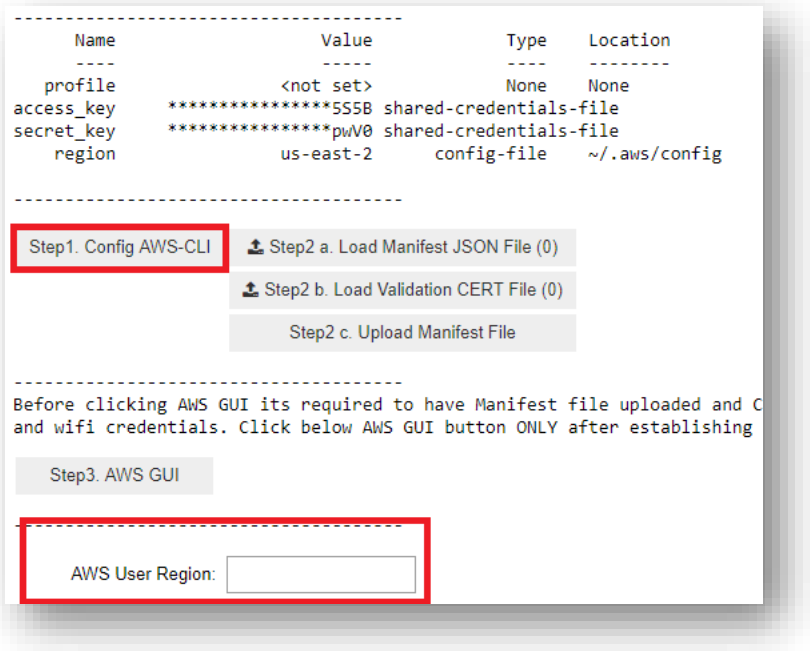


2. Run All cells by using Kernel -> Restart & Run all

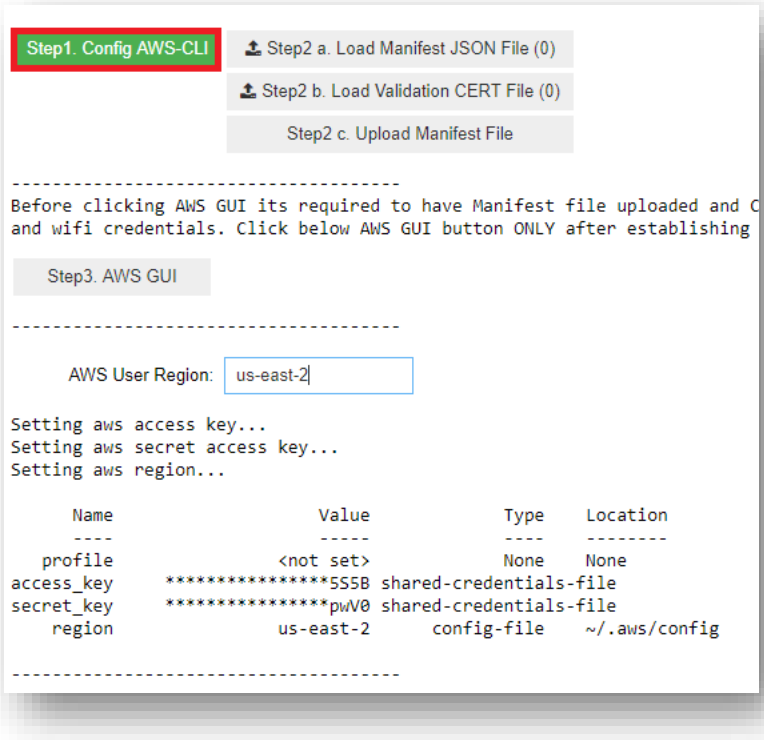


3. After running all cells in notebook, it will create four button 1. “Config AWS-CLI”, 2. “Load Manifest JSON File”, 3. “Load Validation CERT File” and 4. “AWS GUI”.
4. Press “**Config AWS-CLI**” button to configure AWS command line interface with AWS account credentials.

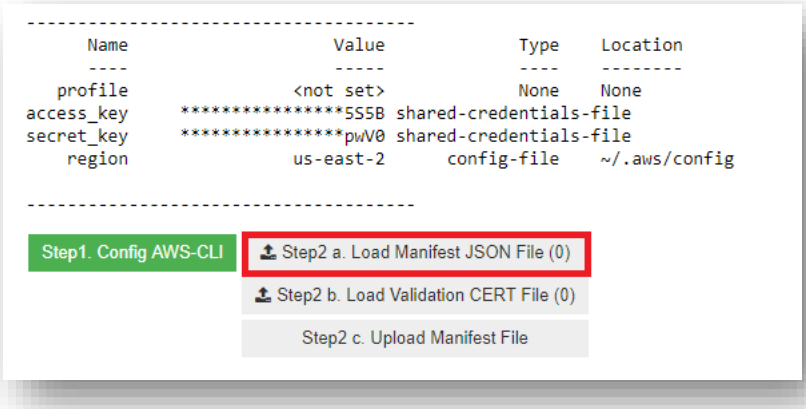
   Once you press the button, it will ask AWS User Region: where region name has to be mentioned as shown in below image.



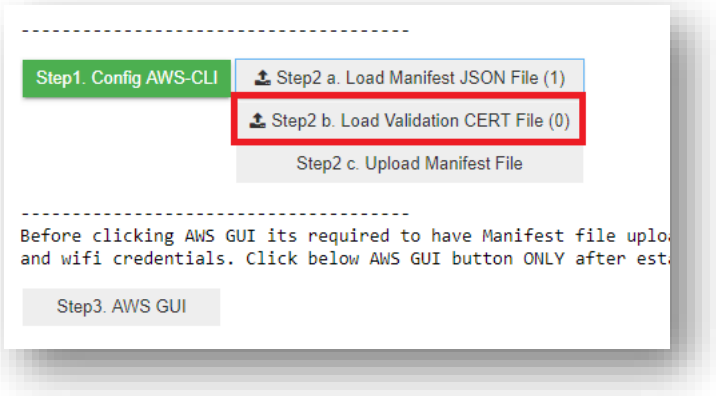
After enter your region press ‘enter’. On successful AWS CLI configuration, you will see the result as shown in below image



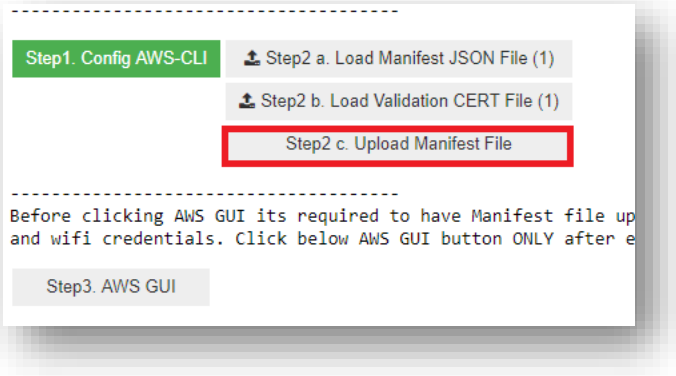
5. Press "**Load Manifest JSON File**" button, it will open file explorer window, there you need to navigate **TrustnGO\00_resource_generation** and choose the manifest file generated using TNG Manifest Generation Notebook.



6. 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.

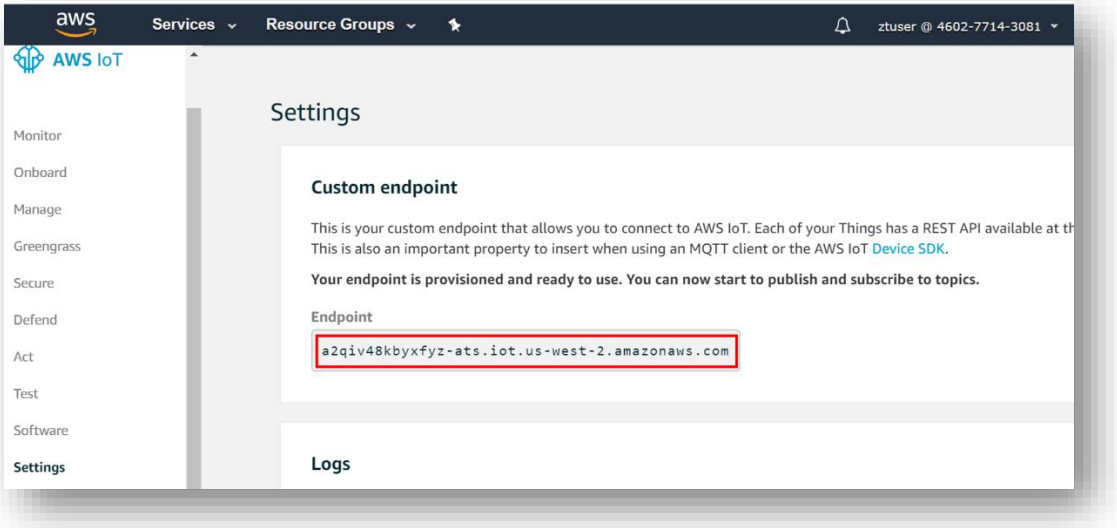
```
number of certificates: 1

Loading the manifest item
uniqueId: 01236c7dc62e6dc001
About to try certificate import
Response: {'ResponseMetadata': {'RequestId': '8a4596b5-b71c-49ad-a453-bedd44e281c8', 'HTTPStatusCode': 200, 'HTTPHeaders': {'da
te': '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': 'C5LRQEEXPHcF
djw=', 'x-amzn-trace-id': 'Root=1-5dc6bf3b-a36debb6c3255f8972bddcbb'}, 'RetryAttempts': 0}, 'certificateArn': 'arn:aws:iot:us-w
est-2:460277143081:cert/461095d8771f95db78d442d2015415804e3093bcae3fd56d78a486aa5b7b59de', 'certificateId': '461095d8771f95db78
d442d2015415804e3093bcae3fd56d78a486aa5b7b59de'}
Certificate import complete - returning
MANIFEST_IMPORT SUCCESS arn:aws:iot:us-west-2:460277143081:cert/461095d8771f95db78d442d2015415804e3093bcae3fd56d78a486aa5b7b59d
e       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.

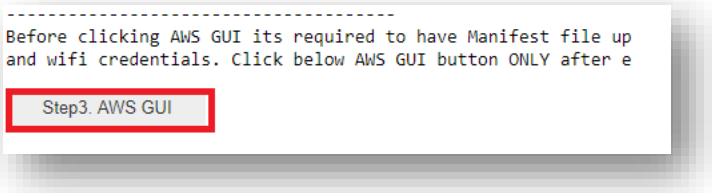
Continue to next steps when connecting to AWS using TNGTLS device.

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.

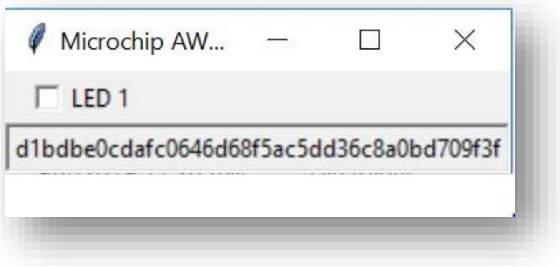
AWS GUI:
Code block of this step generates "**AWS GUI**" button.



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

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

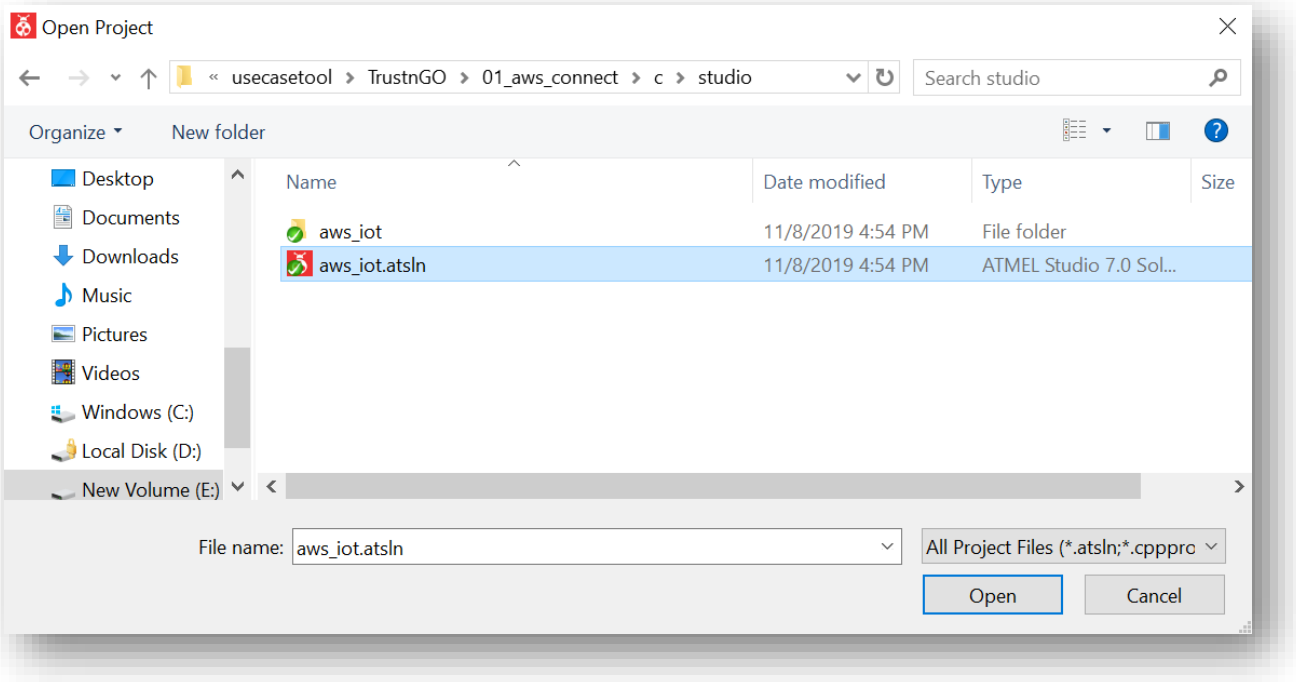
Once the resources are generated, both Atmel Studio and MPLAB projects provided can be used to run the use case on CryptoAuth Trust Platform.

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.

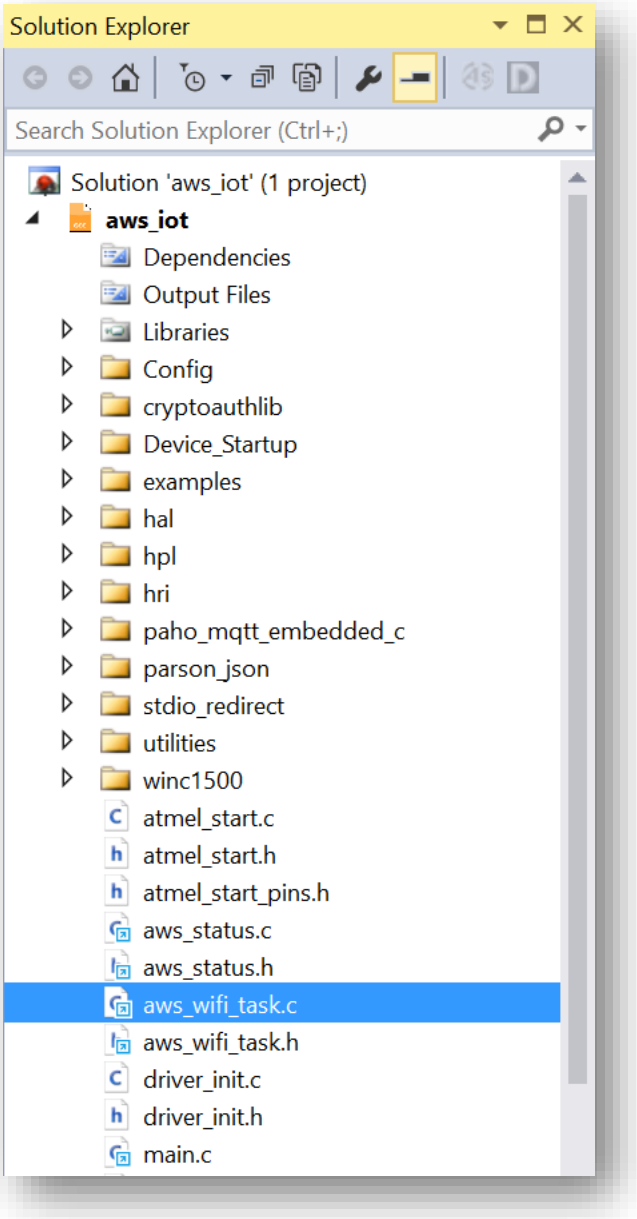
Once the manifest file uploaded to AWS IoT then these embedded projects can be executed.

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

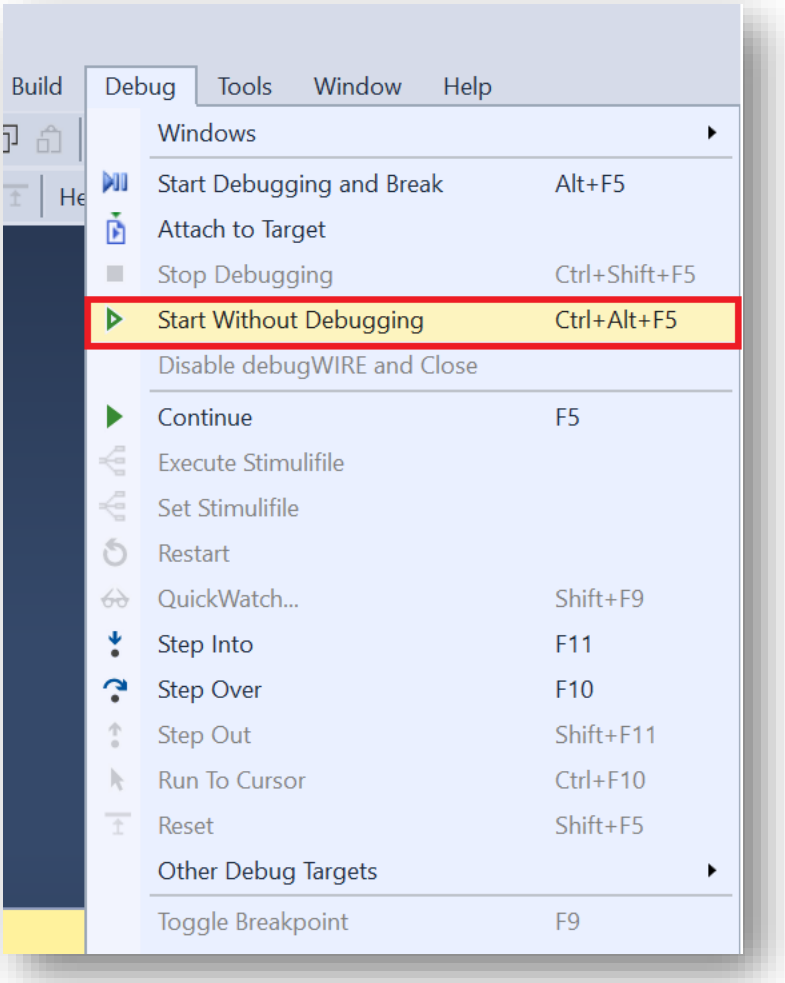


update the following constants before building the project:

- MAIN_WLAN_SSID
- MAIN_WLAN_PSK
- AWS_HOST_ENDPOINT

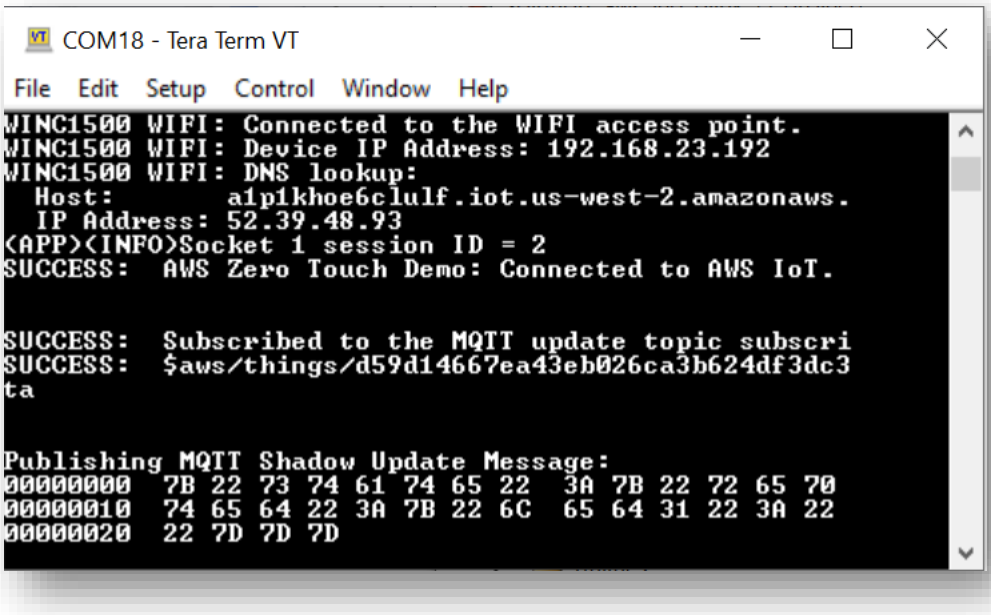
```
#define MAIN_WLAN_SSID      "xxxxxx"
#define MAIN_WLAN_AUTH      M2M_WIFI_SEC_WPA_PSK
#define MAIN_WLAN_PSK       "xxxxxxxxxx"

#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.

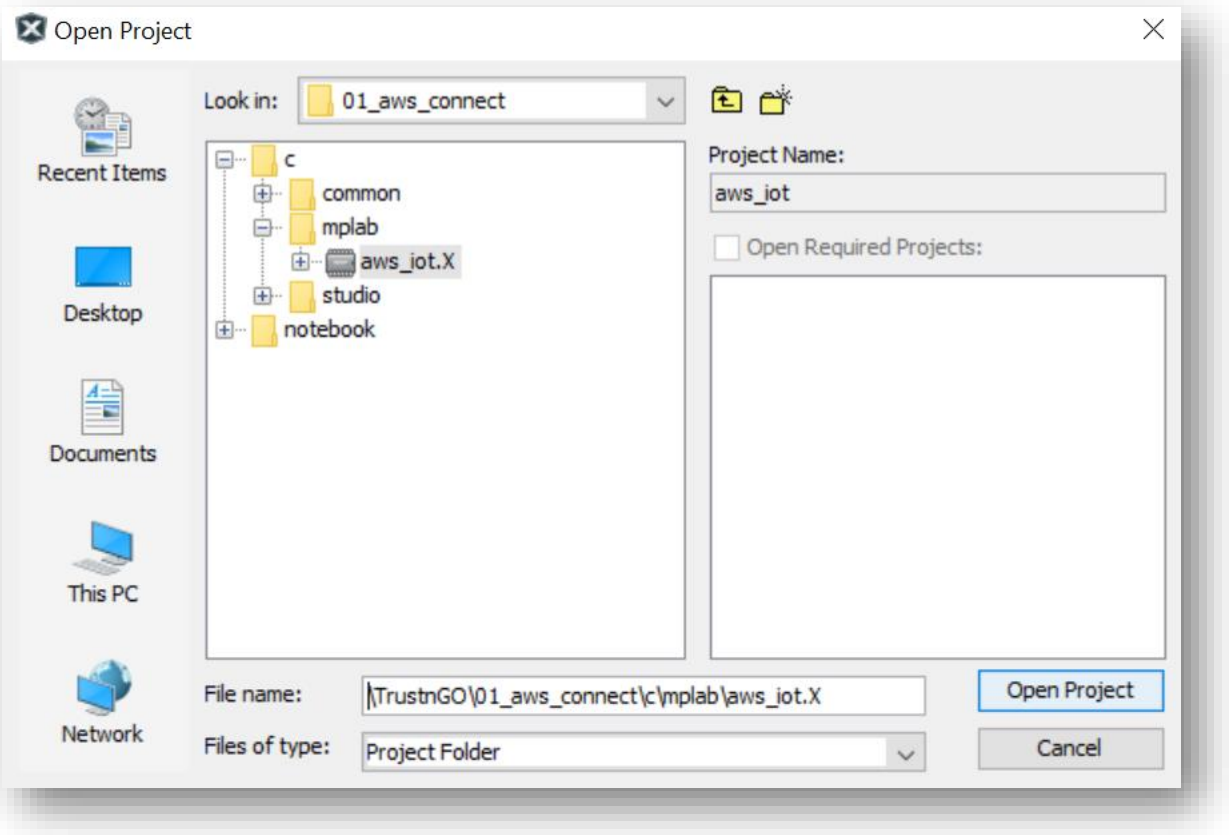
Once the programming is done, reset the hardware (press the reset button) and view the Console messages by using applications like 'Tera Term'. Open the application with the COM related to CryptoAuth Trust Platform with 115200-8-N-1 settings.



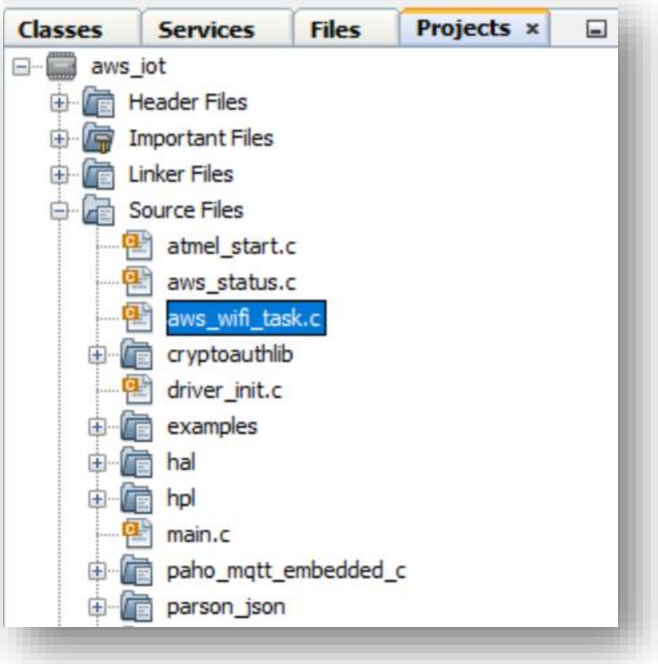
Once successfully programmed the CryptoAuth Trust Platform, now we can run the last step in the Jupyter Notebook. Just navigate to previous section 4.1 to run the last step (AWS GUI) in the Jupyter Notebook.

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**

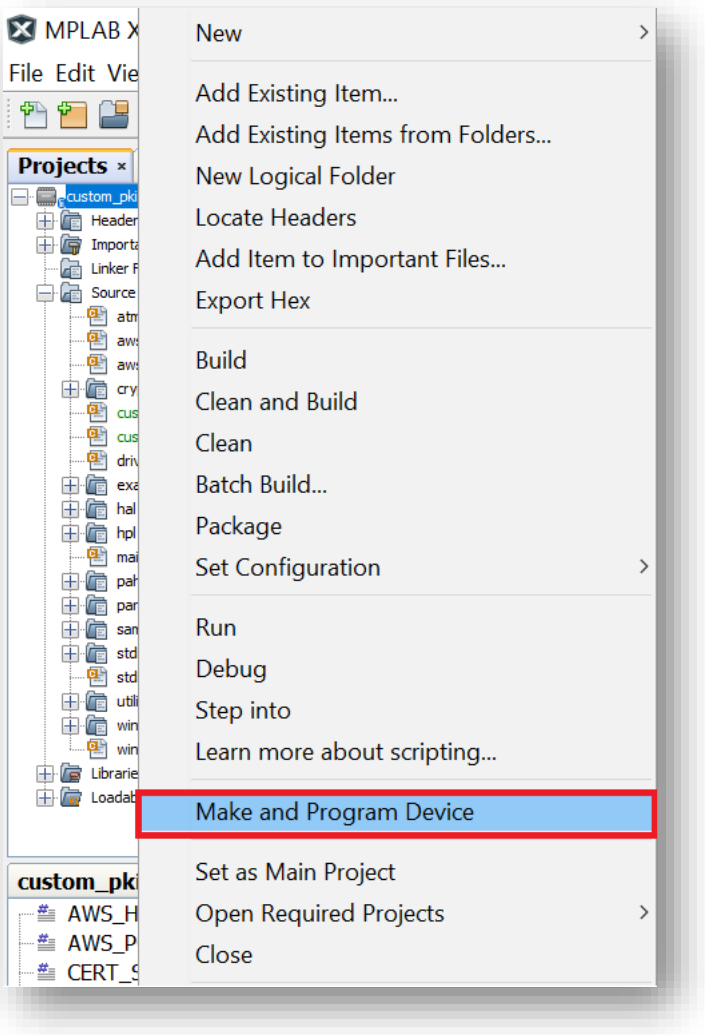


update the following constants before building the project:

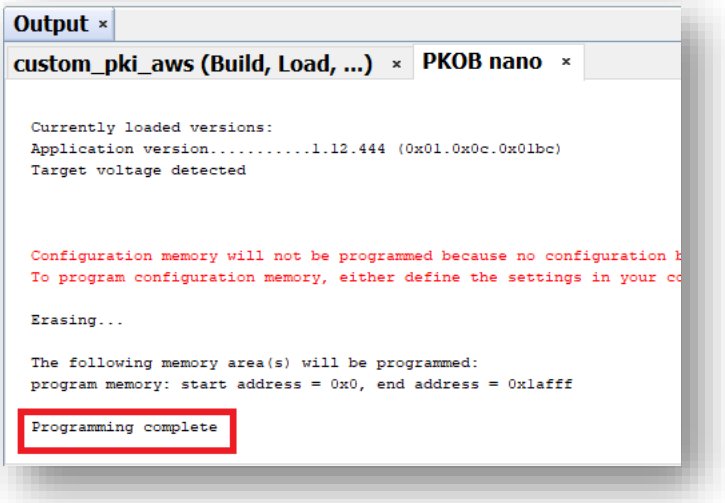
- MAIN_WLAN_SSID
- MAIN_WLAN_PSK
- AWS_HOST_ENDPOINT

```
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
```

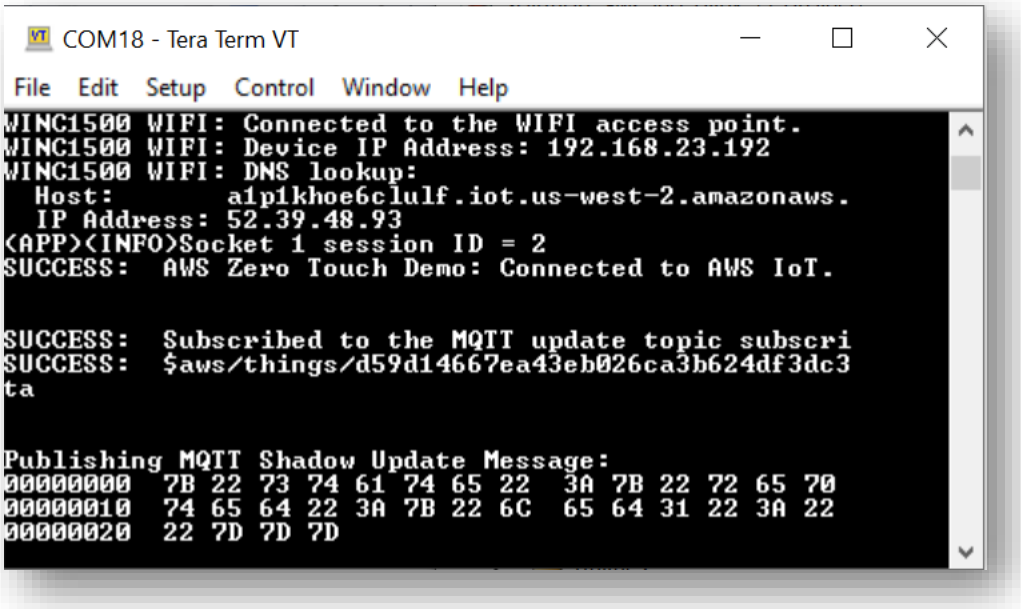
3. Program the CryptoAuth Trust platform by navigating to **aws_iot -> Make and Program Device**



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.



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.



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

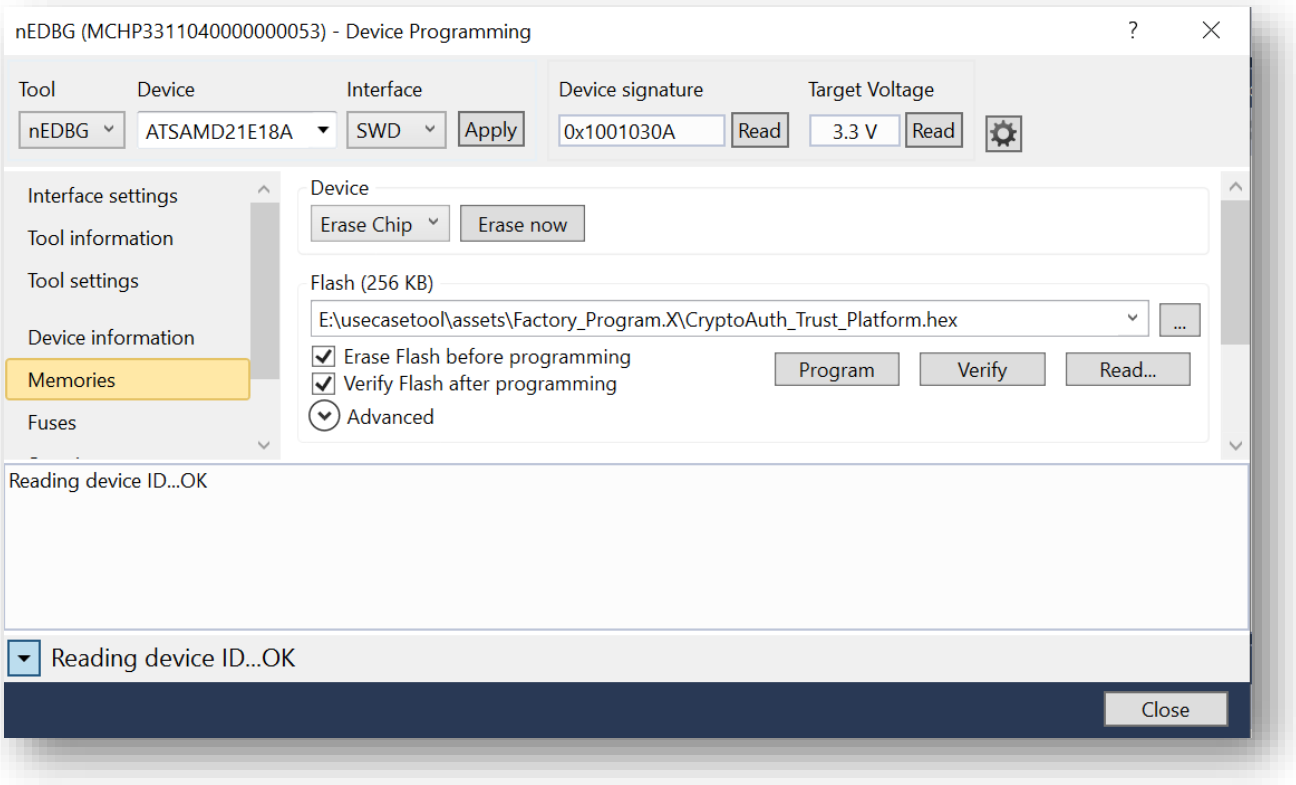
4.3 CryptoAuth TrustPlatform Factory reset

Once any of the embedded project is loaded to CrytoAuth 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

Trademarks

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

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

ASIA/PACIFIC

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

ASIA/PACIFIC

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

EUROPE

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