
Trust&GO Step by Step Guide - Azure Cloud Platform Connect

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
2	<i>Jupyter Notebook Basics</i>	4
2.1.1	The Notebook dashboard	4
2.2	<i>Introduction to Jupyter Notebook GUI</i>	4
3	<i>Jupyter Notebook Tutorials</i>	6
4	<i>Manifest Generation Notebook</i>	7
5	<i>Use Case Prototyping</i>	12
5.1	<i>Running Azure connect example on Jupyter Notebook</i>	12
5.2	<i>Running Azure connect example on Embedded platform</i>	18
5.2.1	MPLAB:	18
5.3	<i>Crypto Auth Trust Platform Factory reset</i>	22

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.

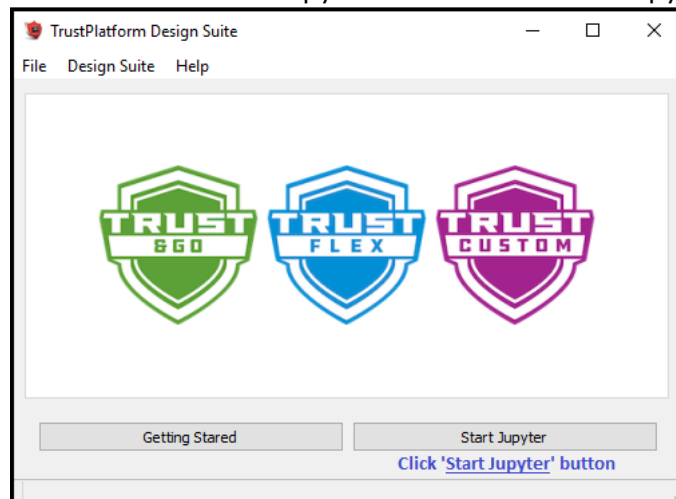
1.1 Getting started with Jupyter Notebook Tutorials

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,



2 Jupyter Notebook Basics

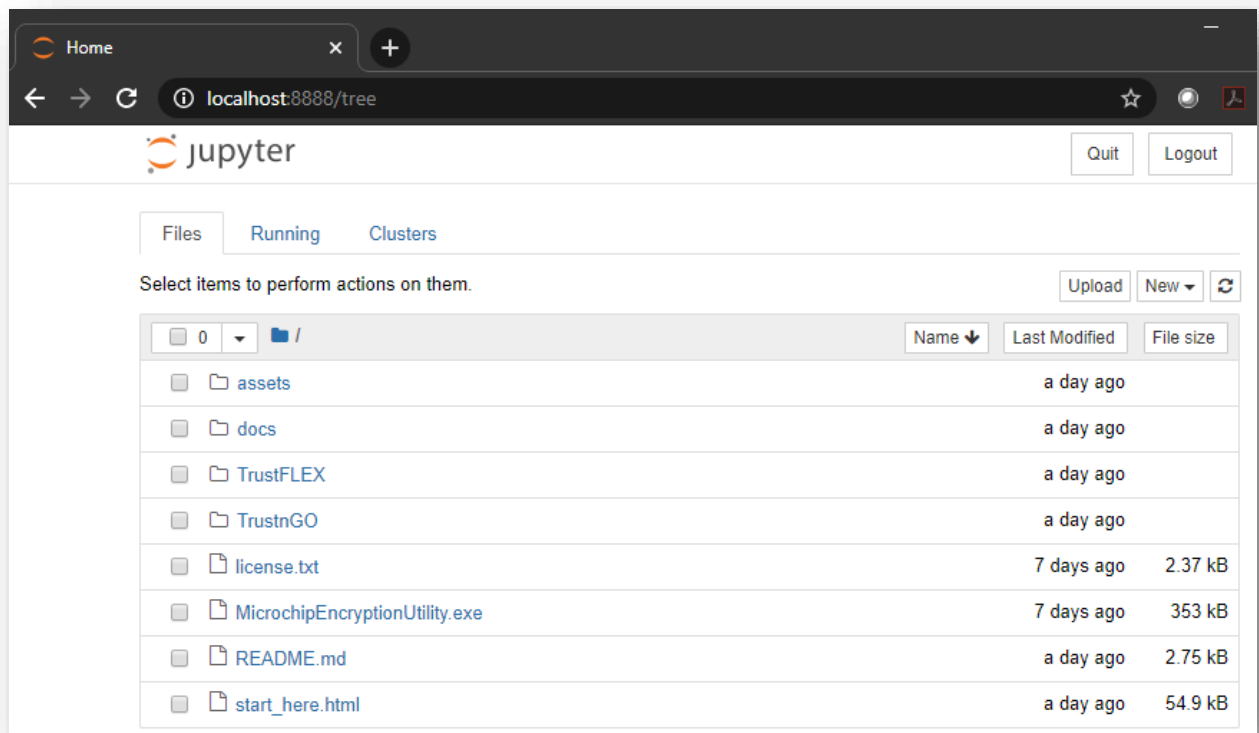
It is recommended to become familiar with Jupyter basic concepts with the online documentation, <https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html>

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

2.1.1 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.



2.2 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 **Run** button (a play icon).
- Cell Type Selector:** A red box highlights the **Markdown** dropdown menu.
- Cell Creation:** A red box highlights the **+** button to create a new cell.
- Selected Cell:** 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".
- Cell Content:**
 - 2. Tutorial:** A markdown cell containing a list of steps: 1. Establish the communication between the host (PC) and the ECC608A, 2. Generate MAC from device, 3. Verify the MAC from device.
 - 2.1. Establish the communication between the host (PC) and the ECC608A:** A markdown cell containing 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. This cell is labeled "MARKDOWN CELL".
 - 2.1.1. Select the CryptoAuthLib Hardware Abstraction layer to KIT USB HID:** A code cell containing Python code for initializing the device. This cell is labeled "CODE CELL".
 - 2.1.2. Establish the connection with the ECC608A-TLXTLS device on the USB dongle and check its type:** A code cell containing Python code for connecting to the device and checking its type.

To run all cells in sequence.

The screenshot shows the **Kernel** menu in the Jupyter Notebook interface. The menu options are:

- Interrupt
- Restart
- Restart & Clear Output
- Restart & Run All** (highlighted with a red box)
- Reconnect
- Shutdown
- Change kernel

A tooltip for the **Restart & Run All** option reads: "Restart the Kernel and re-run the notebook".

3 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
GCP Connect	TrustnGO\05_cloud_connect\notebook\gcp\TNGTLS_GCP_connect.ipynb	Trust&GO
AWS Connect	TrustnGO\05_cloud_connect\notebook\aws\TNGTLS_aws_connect.ipynb	Trust&GO
Azure Connect	TrustnGO\05_cloud_connect\notebook\azure\TNGTLS_azure_connect.ipynb	Trust&GO
Resource Generation	TrustFLEX\00_resource_generation\TFLXTLS_resource_generator.ipynb	TrustFLEX
Accessory Authentication	TrustFLEX\01_accessory_authentication\notebook\TFLXTLS_accessory_authentication.ipynb	TrustFLEX
Firmware Validation	TrustFLEX\02_firmware_validation\notebook\TFLXTLS_firmware_validation.ipynb	TrustFLEX
IP Protection	TrustFLEX\04_ip_protection\notebook\ TFLXTLS_IP_protection.ipynb	TrustFLEX
Secure Public Key Rotation	TrustFLEX\05_public_key_rotation\notebook\TFLXTLS_public_key_rotation.ipynb	TrustFLEX
Asymmetric authentication	08_asymmetric_authentication\notebook\TFLXTLS_asymmetric_authentication.ipynb	TrustFLEX
GCP Connect	TrustFLEX\10_cloud_connect\notebook\gcp\TFLXTLS_GCP_connect.ipynb	TrustFLEX
AWS Custom PKI	TrustFLEX\10_cloud_connect\notebook\aws\ TFLXTLS_aws_connect.ipynb	TrustFLEX
Azure Connect	TrustFLEX\10_cloud_connect\notebook\azure\TLFXTLS_azure_connect.ipynb	TrustFLEX

4 Manifest Generation Notebook

Trust&GO device is one of the three devices available in the Crypto Auth Trust Platform 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.

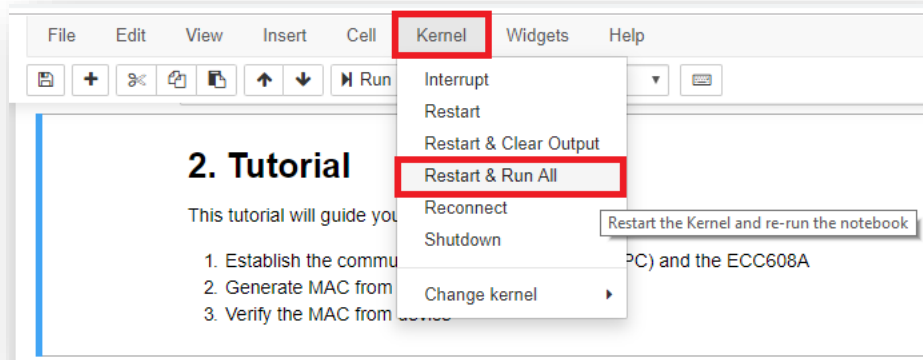
The secure element manifest format is designed to convey the unique information about a device including its unique ID (e.g. serial number), public keys, and certificates. The manifest file generated can be used to register the device to cloud providers.

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 [Crypto Auth Trust Platform Factory reset](#) section for reloading default program.



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

```
Root Certificate loading from Device...OK
```

```
-----BEGIN CERTIFICATE-----
```

```
MIIB8TCCAzegAwIBAgIQd9Nt1W7IrmIF5Y46y5hagTAKBggqhkJOPQQDAjBPMSEw
HwYDVQQKBhNaWNyb2NoaXAgaGVhZG9neSBjbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXR0ZW50aWNhdGlvbiBSb290IENBIDAuMjAgFw0xODExMDgxOTEyMTlaGA8y
MDU4MTEwODE5MTIxOVowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSowKAYDVQQDDCFCdnlwdG8gQXV0aGVudG1jYXRpb24gUm9vdCBDQSAwMDIw
WTATBgqhkJOPQIBBggqhkJOPQMBBwNCAAS9VOZt44dUhABrU64VgNUKoGnnit9V
eNhc4tVN1bgwKWv/3W5vclb72Z7xoRaxHTotSRA6oYWHOdZ65DfhnWNOo1MwUTAd
BgNVHQ4EFgQUeu19bca3eJ2yOAGl6EqMsKQOKowwHwYDVR0jBBgwFoAUeu19bca3
eJ2yOAGl6EqMsKQOKowwDwYDVR0TAQH/BAUwAwEB/zAKBggqhkJOPQQDAgNIADBF
AiEAodxjRZDsgZ7h3luBEmVRrdTCxPj1lSgu4EvnaOx8AnMCID5rp06eTArWjCSw
+y7nk9LmvpRlyhXQ6lvIf1V5mVyt
-----END 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
```

Validate Root Certificate...OK

Signer Certificate loading from Device...OK

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

```
MIICBTCCAaqqAwIBAgIQfDEW4DQGWyXgU7+wniYaZjAKBggqhkJOPQQDAjBPMSEw
HwYDVQQKBHNaWNYb2NoaXAgaGVGVjaG5vbG9neSBjbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXRozW50aWNhdGlvbiBSb290IENBIDAwMjAgFw0xODEyMTQxOTAwMDBaGA8y
MDQ5MTIxNDE5MDAwMFowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSowKAYDVQDDCFDcnldG8gQXV0aGVudGljYXRpb24gU2lnbmVyeIEY2NDaw
WTATBgqhkJOPQIBBggqhkJOPQMBBwNCAAOQfzKV8utGQPSqOUz15SDX2bULuVT1
w/i7bz8sGFpNuZCRvK9J6gb8S8xcKifI0AIrGpvwG/RG3ZrFYjBMejh2o2YwZDAO
BgNVHQ8BAf8EBAMCAYYwEgYDVR0TAAQH/BAgwBgEB/wIBADAdBgNVHQ4EFgQU62ID
K4yBWBZCmhyr8b6MIh63pskwHwYDVR0jBBgwFoAUeu19bca3eJ2yOAGl6EqMsKQO
KowwCgYIKoZIzj0EAwIDSQAARgIhAOB47QYnFfAxMvDvMZcipUni4YYoc7Xyt18o
PuN9E268AiEA32h2vgUirn/pFYSC+ghFjdqc8wgXL9ZgdPwRkHowR3s=
```

-----END 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

Validate Signer Certificate...OK

Device Certificate loading from Device...OK

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

MIIB9TCCAzugAwIBAgIQc0PaLGk8Q6DyF0sMb9xx7TAKBggqhkJOPQQDAjBPMSEw
HwYDVQQKBHNaWNYb2NoaXAgaGVGVjaG5vbG9neSBjb2NBMmIUNyeXB0
byBBdXRoZW50aWNhdGlvbiBTaWduZXIgaG50MDAgFw0xOTA3MzEyMzAwMDBaGA8y
MDQ3MDczMTIzMDAwMFowRjEhMB8GA1UECgwYTWljcm9jaGlwIFRlY2hub2xvZ3kg
SW5jMSEwHwYDVQQDDDBGwMTIzOUE2REYyRUNFQ0RDMDEgQVRFRQ0MwWTATBgqhkJOP
QIBBggqhkJOPQMBBwNCAAQYjZv6hNvOGfiXtqRPqKJr7hnh0Hf6AI68KjrRy8/
93zhXWizlG2VexKLeER97Y6wU2fysMJ4rWQjUgQ54iX5o2AwXjAMBGNVHRMBAf8E
AjaAMA4GA1UdDwEB/wQEAwIDiDAdBgNVHQ4EFgQUUnbEcKNb3ZxBz/s1zs0GfTC95
UfEwHwYDVR0jBBgwFoAU62IDK4yBWBZCmhyr8b6MIh63pskCgYIKoZIZj0EAwID
SAAwRQIhAMG4O+JnJdJ+4qwg6HEyZu/sHkqSUqnbnW5jfSCsSQjSAiB3rimVHLb9
bIheMqsIbK2tXTjtLhCs5s15WvpNvKevlQ==

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

Certificate:

Data:

Version: 3 (0x2)
Serial Number:
73:43:da:2c:69:3c:43:a0:f2:17:4b:0c:6f:dc:71:ed
Signature Algorithm: ecdsa-with-SHA256
Issuer: O=Microchip Technology Inc, CN=Crypto Authentication Signer F640
Validity
Not Before: Jul 31 23:00:00 2019 GMT
Not After : Jul 31 23:00:00 2047 GMT
Subject: O=Microchip Technology Inc, CN=01239A6DF2ECECDC01 ATECC
Subject Public Key Info:
Public Key Algorithm: id-ecPublicKey
Public-Key: (256 bit)
pub:
04:18:8e:66:6f:ea:13:6f:38:67:e2:5e:da:91:3e:
a2:89:af:b8:67:87:41:df:e8:02:3a:f0:a8:eb:47:
2f:3f:f7:7c:e1:5d:62:33:94:6d:95:7b:12:8b:78:
44:7d:ed:8e:b0:53:67:f2:b0:c2:78:ad:64:23:52:
04:39:e2:25:f9
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:
9D:B1:1C:28:D6:F7:67:10:73:FE:CD:73:B3:41:9F:4C:2F:79:51:F1
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
30:45:02:21:00:c1:b8:3b:e2:67:25:d2:7e:e2:ac:20:e8:71:
32:66:ef:ec:1e:4a:92:52:a9:db:99:6e:63:7d:20:ac:49:08:
d2:02:20:77:ae:29:95:1c:b6:fd:6c:88:5e:32:ab:08:6c:ad:

ad:5d:38:ed:2e:10:ac:e6:cd:79:5a:fa:4d:bc:a7:af:d5

Validate Device Certificate...OK

Generating manifest data...OK (saved to TNGTLS_devices_manifest.json)

By default, TNGTLS_devices_manifest.json, manifest_ca.key and manifest_ca.crt files will be created. manifest_ca.crt to be used as cert to verify the content while providing manifest file.

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 Microsoft Azure). TNGTLS Manifest Generation notebook needs to be run for all Trust&Go example Notebooks that require a Manifest file.

5 Use Case Prototyping

This hands-on lab is intended to demonstrate the usage of TrustnGO to secure an Azure IoT connection based on TNGTLS device.

The Azure IoT device reference implementation is provided as an MPLAB X project and the manifest generation is achieved through the execution of Jupyter Notebook Tutorials.

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

- Configure Azure CLI
- Register device
- Build the Azure IoT device source code and flash it to Crypto Auth Trust Platform Board

Note: It is required to have an Azure IoT test account setup. Instruction to setup the account quickly is provided in

docs\TrustFLEX_guide_Azure_demo_account_setup.pdf. Once the account is setup, IoT hub HostName should be updated in **docs\Azure_iot_hub_details.csv**

5.1 Running Azure connect example on Jupyter Notebook

By running this following step, we can configure the Azure command line with Azure credentials, register the device certificate to Azure IoT.

1. From the Jupyter Home page, navigate to **TrustnGO\05_cloud_connect\notebook\azure\TNGTLS_azure_connect.ipynb** notebook file and open it.



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

jupyter TNGTLS_azure_connect (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Introduction

MUST READ: This Notebook tutorial will allow you to setup with Azure account for Azure connection using TNGTLS. By using this Notebook, its possible to update account credentials to Azure cli, Register the Signer and device.

Before running this Notebook,

1. docs/Azure_iot_hub_details.csv file should have updated with azure iot hub name.
2. Make sure CryptoAuth Trust Platform is having factory reset program

1.1.Tutorial Prerequisites:

The following code is runs on python 3.x environment. This step of the tutorial will attempt to load all the necessary modules and their dependencies on your machine. If the above modules are already installed you can safely step over the next Tutorial step.

```
pip install -U module_name
```

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

File Edit View Insert Cell Kernel Widgets Help

Interrupt
Restart
Restart & Clear Output
Restart & Run All
Reconnect
Shutdown
Change kernel

Restart the Kernel and re-run the notebook

perform Accessory authentication

codes of operation with MAC comm

The CryptoAuthentication the host sends a random seed to the device and the device generates a nonce (number

Introduction

MUST READ: This Noteb

3. Navigate through different cells output for the description of the step and result from the execution.

4. There are 2 major steps:

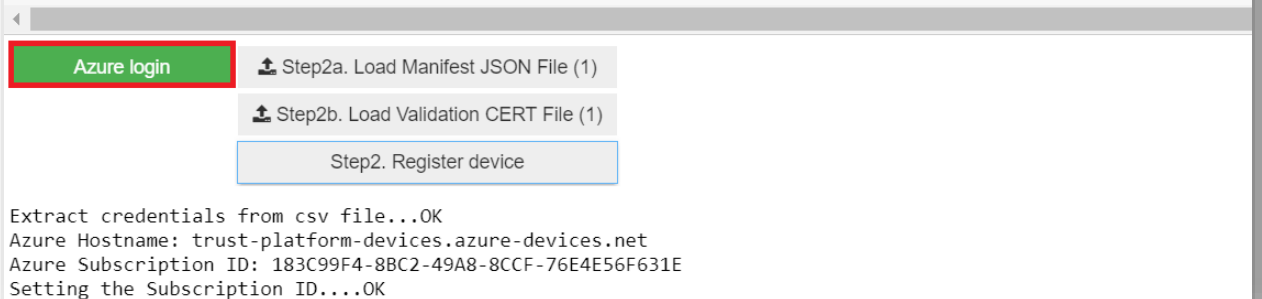
Login to Azure account:

Before we can interact with Azure, we need to login with the appropriate Azure credentials. These credentials are composed of the **azure account login**. Clicking the button will open azure login portal in new tab in browser. Enter the credentials of your account.

Upon successful login, the log should appear like below screenshot.

```
validation_cert = FileUpload(description='Step2b. Load Validation CERT File', accept='.cert', layout=widgets.L
reg_device = widgets.Button(description = "Step2. Register device", layout=widgets.Layout(width='auto'))
reg_device.on_click(execute_device_register)

display(widgets.HBox((Azure_login, widgets.VBox((manifest_file, validation_cert, reg_device)))))
```



Extract credentials from csv file...OK
Azure Hostname: trust-platform-devices.azure-devices.net
Azure Subscription ID: 183C99F4-8BC2-49A8-8CCF-76E4E56F631E
Setting the Subscription ID....OK

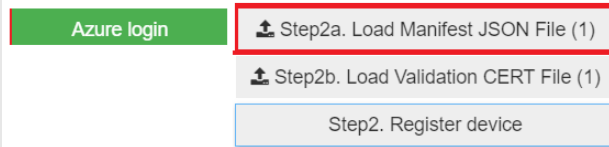
Register Device:

To register a device to Azure cloud, its required to have its manifest file and validation certificate. Both these will be generated as part of Trust&GO resource/manifest generation.

1. 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.

```
validation_cert = FileUpload(description='Step2b. Load Validation CERT File', accept='.crt', layout=widgets.L
reg_device = widgets.Button(description = "Step2. Register device", layout=widgets.Layout(width='auto'))
reg_device.on_click(execute_device_register)

display(widgets.HBox((Azure_login, widgets.VBox((manifest_file, validation_cert, reg_device)))))
```

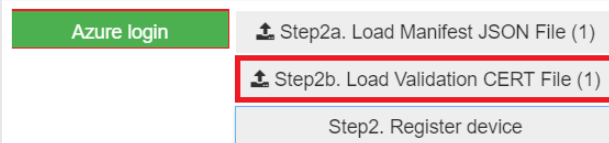


```
Extract credentials from csv file...OK
Azure Hostname: trust-platform-devices.azure-devices.net
Azure Subscription ID: 183C99F4-8BC2-49A8-8CCF-76E4E56F631E
Setting the Subscription ID....OK
```

2. 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.

```
validation_cert = FileUpload(description='Step2b. Load Validation CERT File', accept='.crt', layout=widgets.L
reg_device = widgets.Button(description = "Step2. Register device", layout=widgets.Layout(width='auto'))
reg_device.on_click(execute_device_register)

display(widgets.HBox((Azure_login, widgets.VBox((manifest_file, validation_cert, reg_device)))))
```

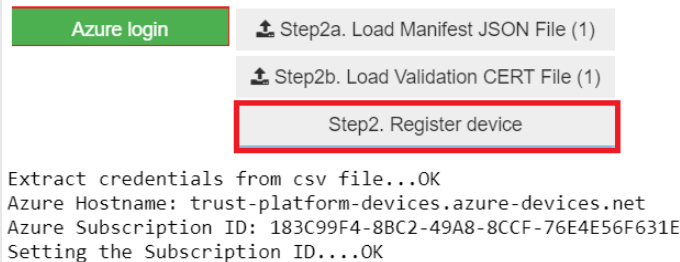


```
Extract credentials from csv file...OK
Azure Hostname: trust-platform-devices.azure-devices.net
Azure Subscription ID: 183C99F4-8BC2-49A8-8CCF-76E4E56F631E
Setting the Subscription ID....OK
```

3. Press **"Register device"** button to register device using the information in Manifest file.

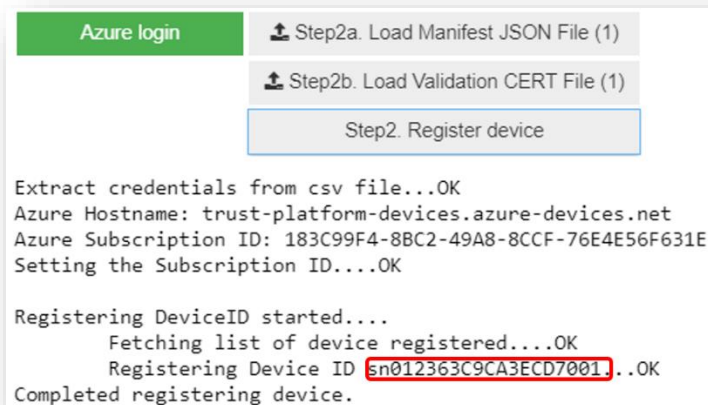
```
validation_cert = FileUpload(description='Step2b. Load Validation CERT File', accept='.crt', layout=widgets.L
reg_device = widgets.Button(description = "Step2. Register device", layout=widgets.Layout(width='auto'))
reg_device.on_click(execute_device_register)

display(widgets.HBox((Azure_login, widgets.VBox((manifest_file, validation_cert, reg_device)))))
```



Extract credentials from csv file...OK
 Azure Hostname: trust-platform-devices.azure-devices.net
 Azure Subscription ID: 183C99F4-8BC2-49A8-8CCF-76E4E56F631E
 Setting the Subscription ID...OK

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



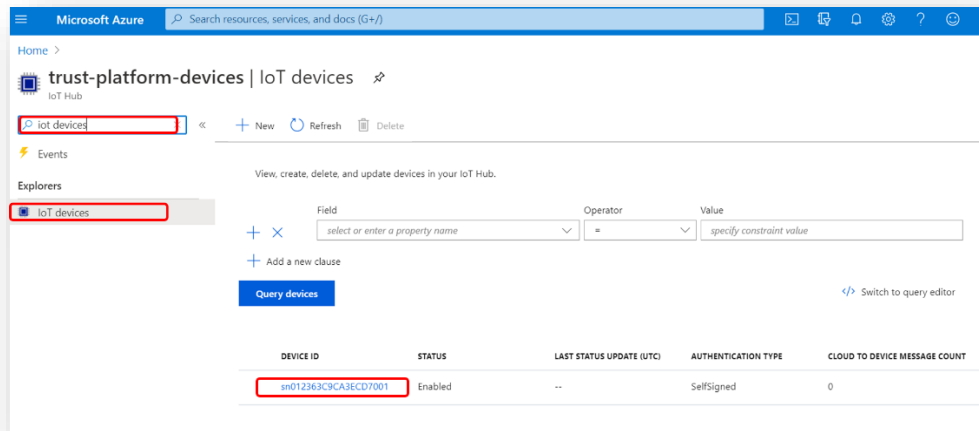
Extract credentials from csv file...OK
 Azure Hostname: trust-platform-devices.azure-devices.net
 Azure Subscription ID: 183C99F4-8BC2-49A8-8CCF-76E4E56F631E
 Setting the Subscription ID...OK
 Registering DeviceID started....
 Fetching list of device registered...OK
 Registering Device ID sn012363C9CA3ECD7001...OK
 Completed registering device.

NOTE: Make sure that you executed C project successfully before executing the next step .To execute C project, refer "[Running Azure connect example on Embedded platform](#)" section.

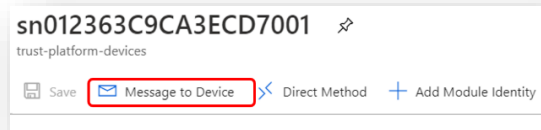
Control Crypto Auth Trust Platform board LED from Azure Cloud

In this section we are going to control the status led on the Trust platform device from the Azure cloud.

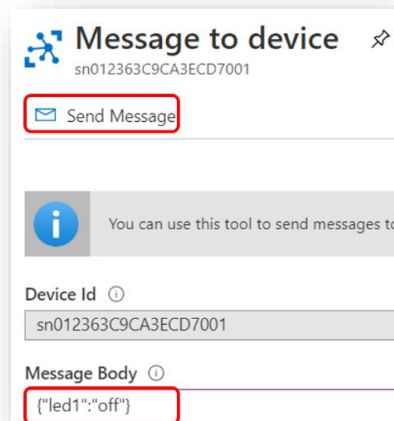
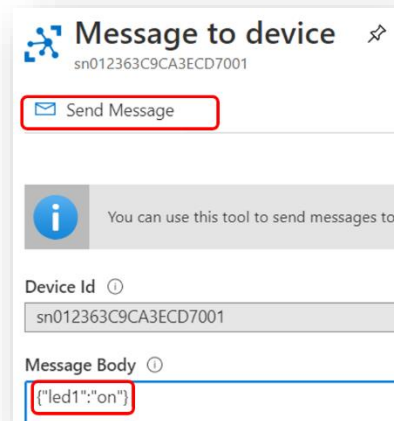
1. In the <https://portal.azure.com/> select your IoT hub created, search for "iot device" and select **IoT devices** from Explorers.
2. From the Device ID select the device id that want to be controlled. The Device ID Should be same as the highlighted one previous step **Register Device**



3. In the Device ID window, select the Message to Device



4. Now to Turn the led on or off the Trust platform board, paste the message **`{"led1": "on"}`** or **`{"led1": "off"}`** to the message body and click Send Message.



5.2 Running Azure connect example on Embedded platform

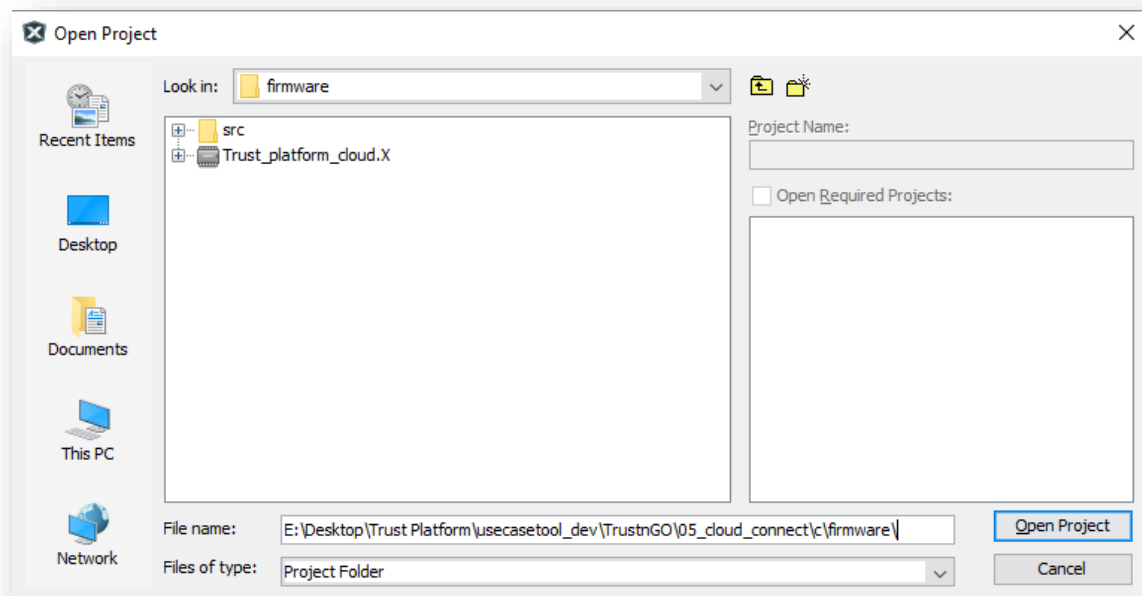
Once the device is registered MPLAB projects provided can be used to run the use case on Crypto Auth Trust Platform.

This project can configure the Wi-Fi credentials, establish a TLS connection, subscribe to MQTT. It is required to use the Azure IoT Jupyter notebook to register the signer module.

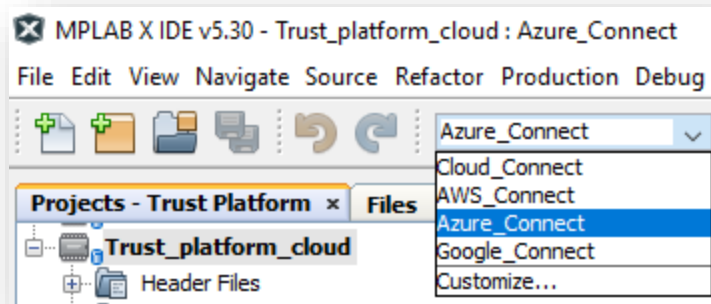
Prerequisite: It is required that WINC firmware is updated to latest version / version that is available in this package. Update the WINC firmware using package available in cloned repository at **assets\winc_firmware_upgrade**

5.2.1 MPLAB:

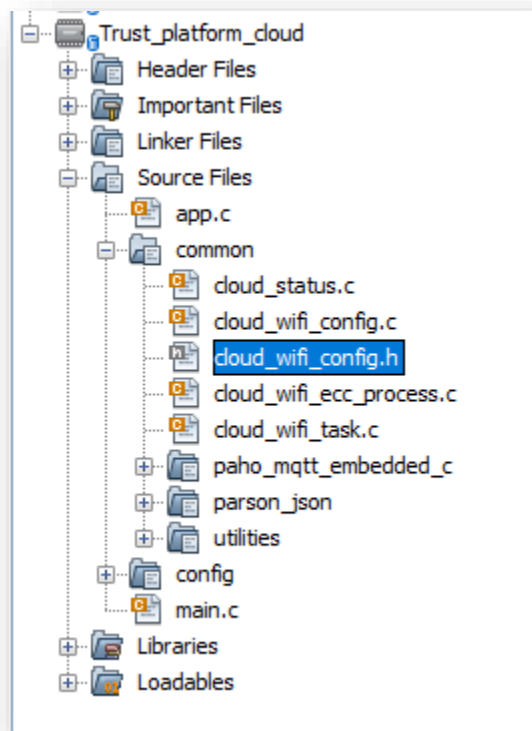
1. Open **Trust_platform_cloud.X** project by navigating to MPLAB -> File -> Open Project -> **TrustnGO\05_cloud_connect\firmware**



2. Select the Build configuration as Azure_Connect



3. Open **cloud_wifi_config.h** file by navigating to **Trust_platform_cloud-> Source Files ->common**

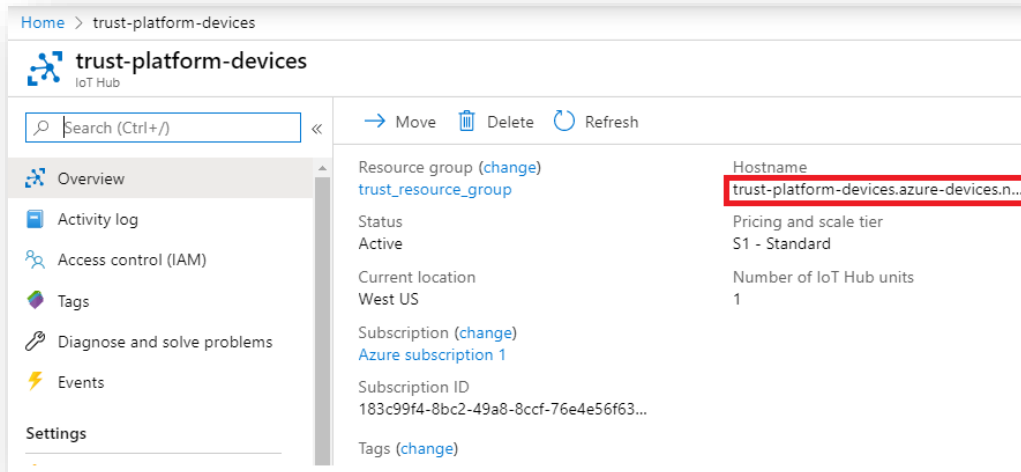


Update the following constants before building the project:

- MAIN_WLAN_SSID
- MAIN_WLAN_PSK
- CLOUD_ENDPOINT for CLOUD_CONFIG_AZURE

```
//#define WLAN_AUTH_OPEN
#define WLAN_AUTH_WPA_PSK
#define WLAN_SSID "xxxxxxxxxxxxxxxx"
#define WLAN_PSK "xxxxxxxxxxxxxxxx"
```

CLOUD_ENDPOINT string should be same as your account Hostname. Refer to following image for reference of this data.

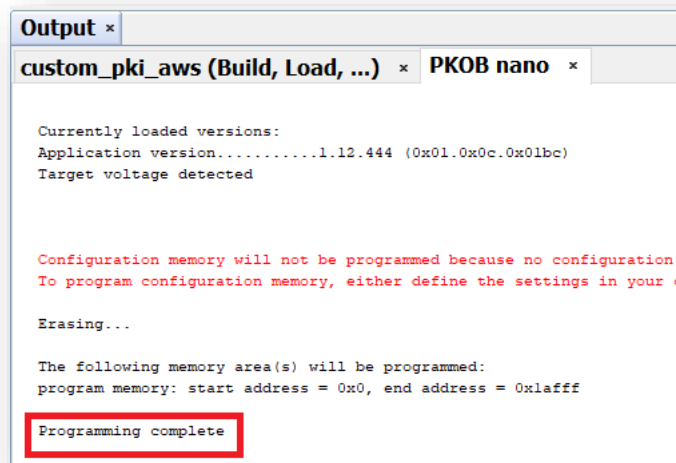


Update CLOUD_ENDPOINT with Hostname from above screen.

```
#elif defined(CLOUD_CONFIG_AZURE)
#define SSL_CIPHER_SUITE_SELECTION SSL_ECC_ALL_CIPHERS
#define CLOUD_ENDPOINT "xxxxxxxxxxxxxxxx.azure-devices.net"
```

4. Program the CryptoAuth Trust platform by navigating to **Trust_platform_cloud -> 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 = 0x1afff

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.



```
Attempting to connect to Azure IoT ...
SSID: karthi
Password: karthikeyan
WINC1500 WIFI: Connected to the WIFI access point.
WINC1500 WIFI: Device IP Address: 192.168.119.192
WINC1500 WIFI: DNS lookup:
Host: trust-platform.azure-devices.net
IP Address: 40.83.177.42
<APP><INFO>Socket 0 session ID = 1
SUCCESS: Azure Demo: Connected to Azure IoT.

SUCCESS: Subscribed to the MQTT update topic subscription:
SUCCESS: devices/0123A2682F50872801_ATECC/messages/devicebound/#

Publishing MQTT Shadow Update Message:
00000000 7B 22 6C 65 64 31 22 3A 22 6F 66 66 22 7D <"led1":"off">
```

Once the device is connected go to Azure portal to control the LED status. Refer to [DeviceControl](#) for details.

5.3 Crypto Auth Trust Platform Factory reset

Once any of the embedded project is loaded to Crypto Auth Trust Platform, the default program that enables interaction with Trust Platform 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 in cloned repository at

assets\Factory_Program.X\CryptoAuth_Trust_Platform.hex

If Trust Platform GUI is provided with MPLAB X IDE installation location, notebooks can program the Factory reset hex file if its not available by default.

This can also be done manually by MPLAB

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 Auth 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, Helder, 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