
TrustFLEX Step by Step Guide

Accessory Asymmetric Authentication

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>	4
1.2.1	The Notebook dashboard	4
1.3	<i>Introduction to Jupyter Notebook GUI</i>	4
2	<i>Jupyter Notebook Tutorials</i>	6
3	<i>Resource Generation Notebook</i>	7
4	<i>Use Case Prototyping</i>	13
4.1	<i>Running Accessory Asymmetric Authentication example on Jupyter Notebook:</i>	13
4.2	<i>Running Accessory Asymmetric Authentication example on Embedded platform</i>	17
4.2.1	Atmel Studio	17
4.2.2	MPLAB:	20
4.3	<i>CryptoAuth TrustPlatform Factory reset</i>	22
5	<i>FAQ</i>	23

1 Introduction

This document gives a detailed walk through of the Accessory Authentication use case implementation. If familiar with Jupyter Notebook, can skip this section and move to Section 2.

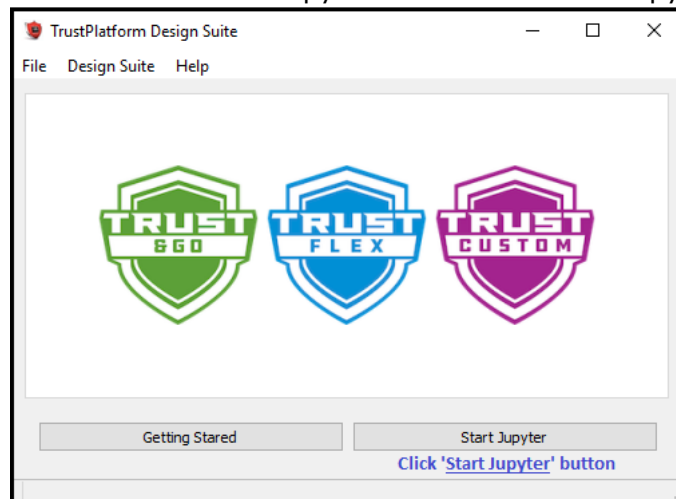
1.1 Getting started with Jupyter Notebook Tutorials

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

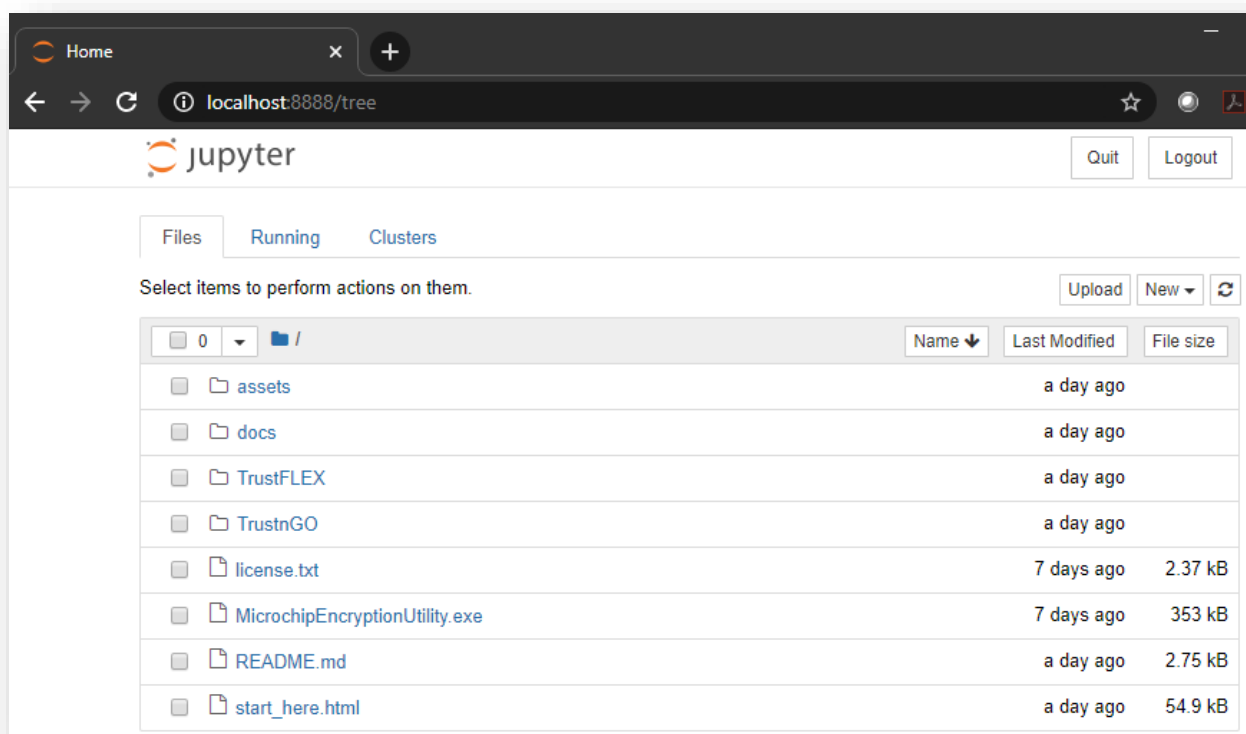
It is recommended to become familiar with Jupyter basic concepts with the online documentation, <https://jupyter-notebook.readthedocs.io/en/stable/examples/Notebook/Notebook%20Basics.html>

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 to the 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.

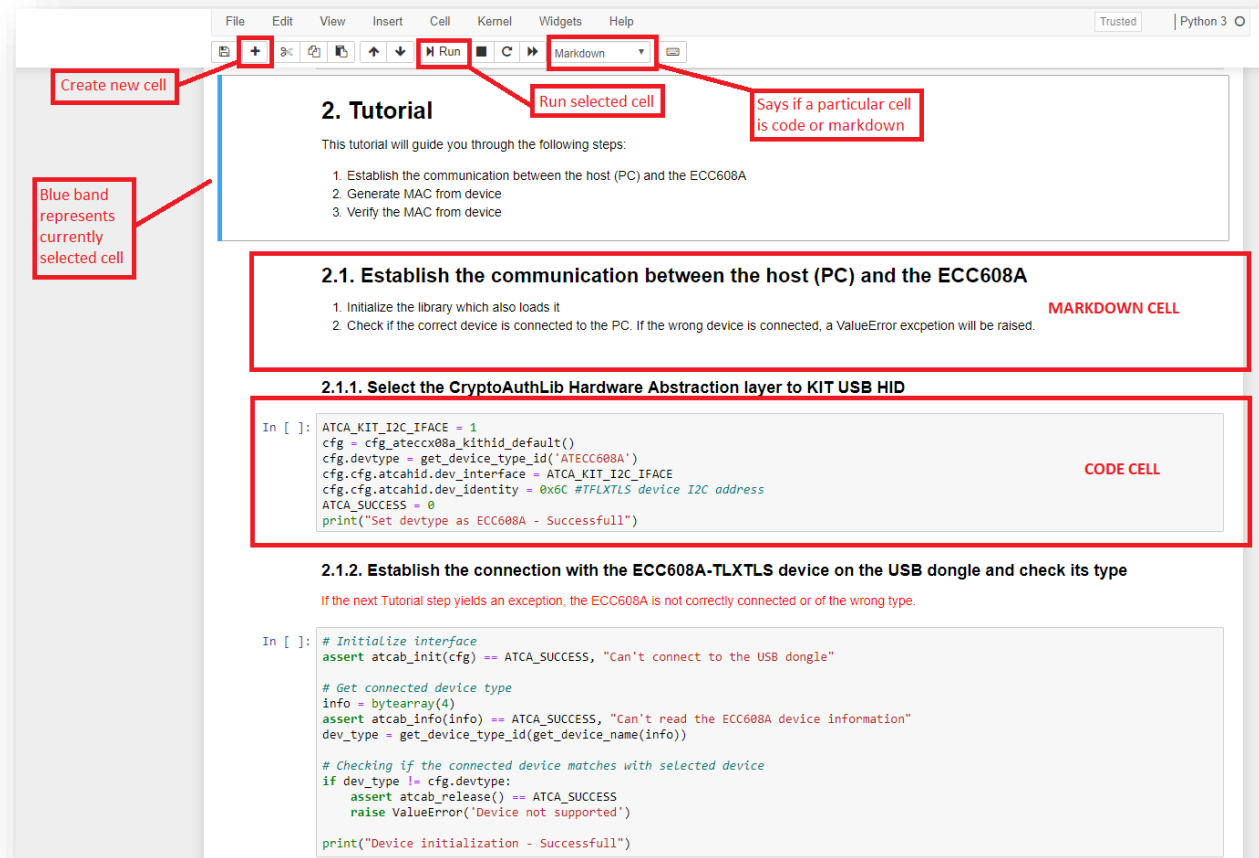


1.3 Introduction to Jupyter Notebook GUI.

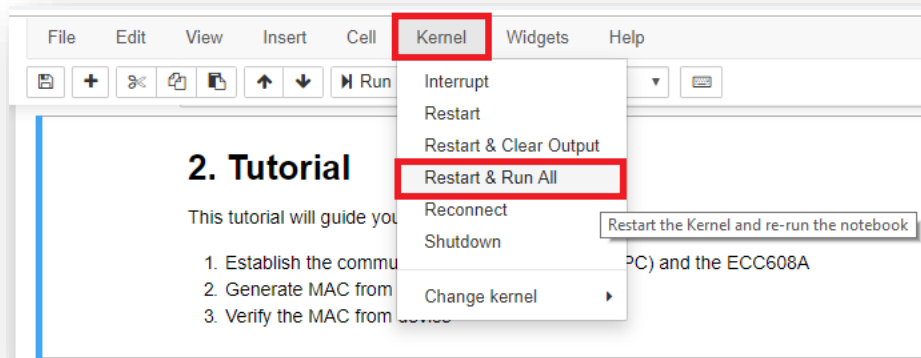
Jupyter Notebooks contain cells where you can either write code or markdown text. Notebooks contain multiple cells, some set as code and others markdown. Code cells contain code that can be executed live, and markdown contains text and images to explain the code.

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 Trust Platform Design Suite comes with Notebook Tutorials to easily prototype popular use cases for TrustFLEX. Here is the list of Jupyter Notebook Tutorials.

Jupyter Notebook Tutorials	Relative Path	Applicable Devices
Resource Generation	TrustFLEX\00_resource_generation\TFLXTLS_resource_generator.ipynb	TrustFLEX
Accessory Authentication	TrustFLEX\01_accessory_authentication\notebook\TFLXTLS_accessory_authentication.ipynb	
Firmware Validation	TrustFLEX\02_firmware_validation\notebook\TFLXTLS_firmware_validation.ipynb	
GCP Connect	TrustFLEX\03_gcp_connect\notebook\TFLXTLS_GCP_connect.ipynb	
IP Protection	TrustFLEX\04_ip_protection\notebook\ TFLXTLS_IP_protection.ipynb	
Secure Public Key Rotation	TrustFLEX\05_public_key_rotation\notebook\TFLXTLS_public_key_rotation.ipynb	
AWS Custom PKI	TrustFLEX\06_custom_pki_aws\notebook\ TFLXTLS_aws_connect.ipynb	
Azure Connect	TrustFLEX\07_custom_pki_azure\notebook\ TLFXTLS_azure_connect.ipynb	
Accessory Asymmetric Authentication	TrustFLEX\08_asymmetric_authentication\notebook\TFLXTLS_asymmetric_authentication.ipynb	

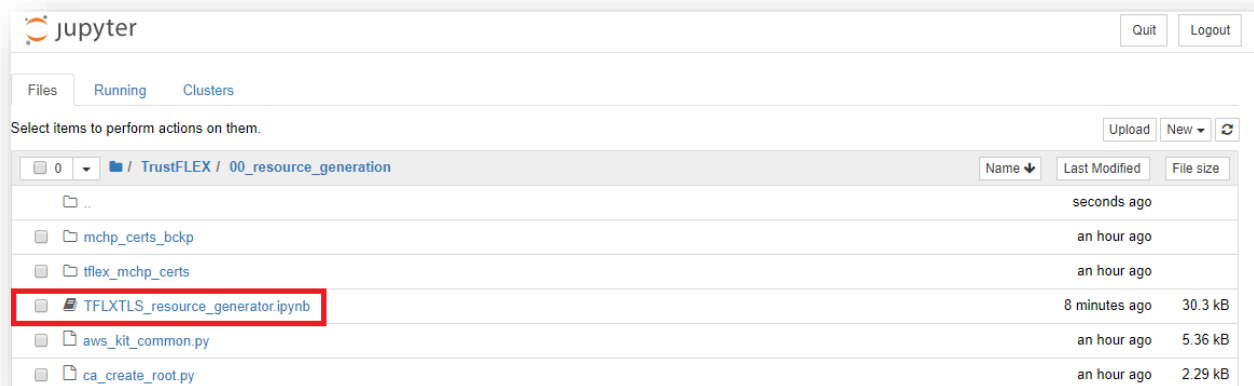
3 Resource Generation Notebook

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

TrustFLEX devices come with pre-programmed certificates in slots 10, 11 and 12, also slots 0-4 have pre-generated private keys, other than the mentioned slots all the other slots have no data in them.

The Resource Generator Notebook will create development keys and certificates for all slots that can be further customized. Keys and Certificate chains are stored in the PC filesystem. These keys should never be used for production purposes as their generation is not handled in a secure environment. These development keys will be later used by the other notebooks to implement the various pre-defined use cases.

Within the Jupyter Dashboard, navigate **TrustFLEX\00_resource_generation** folder to open **TFLXTLS_resource_generator.ipynb** notebook



Run all cells of the Crypto Resource Generator 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.



Crypto Resource Generator notebook is common for all the use case which comes with option to load the signer certificate and device certificate.

So, it will execute and prompt you to choose between MCHP certificate and a custom certificate chain, press "MCHP Cert" option for this use case.

The Notebook will generate several keys and certificates. Make sure you have an error free output before continuing to the next steps of the training.

```
create_manifest_log_signer()
print('\n\nSelect the Certificate Type to prototype')
display(widgets.HBox((mchp_cert_button, cust_cert_button)))

----- Creating Manifest Log Signer -----
Loading Manifest logger key

Generating self-signed logging certificate
Saving to log_signer.crt
-----

Select the Certificate Type to prototype
MCHP Cert Custom Cert
```



```
MCHP Certs processing...
MCHP certificates found in the device
```

Backing up certificates from device - Success

MIIB8TCCAzegAwIBAgIQd9NtIW7IrmIF5Y46y5hagTAKBggqhkJOPQQDAjBPMSEw
HwYDVQQKBhNaWNYb2NoaXAgVGvjaG5vbG9neSBjbmMxKjAoBgNVBAMMIUNyeXB0
byBBdXRoZW50aWNhdGlvbISBs290IENBIDAuMjAgFw0xODExMDgxOTEyMTIaGA8y
MDU4MTEwODE5MTIxOVowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRIY2hub2xvZ3kg
SW5jMSowKAYDVOQDDCFDcnIwdG8qOXV0aGVudGljYXRpb24qUm9vdCBDQSAwMDIw

WTATBgcqhkJOPQIBBggqhkjOPQMBBwNCAAS9VOZt44dUhABrU64VgNUKoGnnit9V
eNhc4tVN1bgwKWv/3W5vclb72Z7xoRaxHTOtSRA6oYWHOdZ65DfhnWNOo1MwUTAd
BgNVHQ4EFgQUeu19bca3eJ2yOAGI6EqMsKQOKowwHwYDVR0jBBgwFoAUeu19bca3
eJ2yOAGI6EqMsKQOKowwDwYDVR0TAQH/BAUwAwEB/zAKBggqhkjOPQDDAgNIADBF
AiEAodxjRZDsgZ7h3luBEmVRrdTCxPjllSgu4EvnaOx8AnMCID5rp06eTArWjCSw
+y7nk9LmvpRlyhXQ6lvIf1V5mVyt
-----END CERTIFICATE-----

Validate Root Certificate:
OK

Signer Certificate:

Certificate:

Data:

Version: 3 (0x2)

Serial Number:

79:0a:a7:d5:7d:73:dc:e9:6d:65:db:66:8b:76:b2:5e

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 F600

Subject Public Key Info:

Public Key Algorithm: id-ecPublicKey

Public-Key: (256 bit)

pub:

04:76:47:41:70:b2:63:e7:99:54:bc:85:bb:12:e9:

fe:70:0c:5b:8d:d4:d6:93:45:98:c2:29:a7:68:02:

0e:4e:0b:6d:48:75:d0:ed:a1:ee:f6:5f:91:5f:c6:

b1:16:46:c5:a1:ca:63:1f:62:55:68:74:47:69:c5:

de:83:b5:89:6a

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:

FB:DC:AA:12:8A:FA:C1:B5:92:8F:CD:AB:11:DB:09:3E:CF:4D:BE:F6

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:c6:30:31:e9:a9:8b:30:4e:68:7e:06:c5:39:

79:2a:c5:7a:5c:01:4d:30:17:de:dc:d2:7d:d5:1d:cd:86:37:

ff:02:21:00:c6:a2:2c:6e:b1:ae:5f:85:91:49:cb:5d:e7:77:

8b:a3:f3:0b:e9:3d:9b:80:6f:94:bf:3d:90:a5:84:78:61:dc

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

MIICBTCCAaqgAwIBAgIQeQqn1X1z30ltZdtmi3ayXjAKBggqhkjOPQDDAgjBPMSEw
HwYDVQQKBHNaWNyY2NoaXAgVGJjaG5vbG9neSBjb2N0bWxKjAoBgNVBAMMIUNyeXB0
byBBdXRoZW50aWNhdGlviBSb290IENBIDAuMjAgFw0xODEyMTQxOTAwMDBaGA8y

MDQ5MTIxNDE5MDAwMFowTzEhMB8GA1UECgwYTWljcm9jaGlwIFRIY2hub2xvZ3kg
SW5jMSowKAYDVQQDDCFDcnlwdG8gQXV0aGVudGljYXRpb24gU2lnbmVvIEY2MDAw
WTATBgcqhkjOPQIBBggqhkjOPQMBBwNCAAR2R0FwsmPnmVS8hbsS6f5wDFuN1NaT
RZjCKadoAg5OC21IddDtoe72X5FfxrEWRsWhymMfYIVodEdpxd6DtYlqo2YwZDAO
BgNVHQ8BAf8EBAMCAYYwEgYDVR0TAQH/BAgwBgEB/wIBADAdBgNVHQ4EFgQU+9yq
Eor6wbWSj82rEdsJP9NvvYwHwYDVR0jBBgwFoAUeu19bca3eJ2yOAGl6EqMsKQO
KowwCgYIKoZIzj0EAwIDSQAwwRgIhAMyWMempizBOaH4GxTI5KsV6XAFNMBfe3NJ9
1R3Nhjf/AiEAXqIsbrGuX4WRsctd53eLo/ML6T2bgG+Uvz2QpYR4Ydw=
-----END CERTIFICATE-----

Validate Signer Certificate:
OK

Device Certificate:
Certificate:

Data:

Version: 3 (0x2)

Serial Number:

5a:cb:a3:f7:cf:bf:c5:28:92:cd:e1:9f:a3:ac:9d:17

Signature Algorithm: ecdsa-with-SHA256

Issuer: O = Microchip Technology Inc, CN = Crypto Authentication Signer F600

Validity

Not Before: Aug 21 22:00:00 2019 GMT

Not After : Aug 21 22:00:00 2047 GMT

Subject: O = Microchip Technology Inc, CN = 0123867D566FFB7701 ATECC

Subject Public Key Info:

Public Key Algorithm: id-ecPublicKey

Public-Key: (256 bit)

pub:

04:fc:57:67:b6:fb:ae:50:60:ca:96:5a:ef:41:b1:
c5:d6:a1:60:61:87:8e:a4:78:f4:4d:18:d0:76:9d:
ad:62:24:b3:68:c2:1a:62:cb:0a:fd:ef:f5:b4:0c:
e3:55:ec:f0:40:bb:41:83:61:02:ef:20:3c:63:93:
32:d4:90:41:ab

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:

43:9E:4F:45:79:35:CE:DC:D4:35:B9:4F:4A:23:69:E1:2D:89:33:04

X509v3 Authority Key Identifier:

keyid:FB:DC:AA:12:8A:FA:C1:B5:92:8F:CD:AB:11:DB:09:3E:CF:4D:BE:F6

Signature Algorithm: ecdsa-with-SHA256

30:45:02:21:00:83:32:78:25:9c:5a:07:7c:4a:04:f8:b5:c4:
57:d6:08:70:ee:c3:d4:79:9c:b6:14:8e:5e:86:54:38:50:cf:
ec:02:20:58:e1:cf:e1:f6:e2:17:08:c3:5a:fc:86:91:31:ef:
65:09:e0:e4:ba:7e:02:8e:4c:49:d1:4b:e3:ac:35:33:f7

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

MIIB9TCCAZugAwIBAgIQWsu98+/xSiSzeGfo6ydFzAKBggqhkhjOPQQDAjBPMSEw

HwYDVQQKDBhNaWNyY2NoaXAgaGVjaG5vbG9neSBJamMxKjAoBgNVBAMMIUNyeXB0
byBBdXR0ZW50aWNhdGlviBTaWduZXIgaG9yYwMDAgFw0xOTA4MjEyMjAwMDBaGA8y
MDQ3MDgyMTIyMDAwMFowRjEhMB8GA1UECgwYTWljcm9jaGlwIFRIY2hub2xvZ3kg
SW5jMSEwHwYDVQQDDDBgwMTIzODY3RDU2NkZGQjc3MDEgQVRFRQ0MwWTATBgcqhkiO
PQIBBgqhkiOPQMBBwNCAAT8V2e2+65QYMqWWu9BscXWoWBhh46kePRNGNB2na1i
JLNowhpiywr97/W0DONV7PBAu0GDYQLvIDxjkzLUkEGro2AwXjAMBgNVHRMBAf8E
AjAAMA4GA1UdDwEB/wQEAwIDiDAdBgNVHQ4EFgQUUQ55PRXk1ztzUNblPSiNp4S2J
MwQwHwYDVR0jBBgwFoAU+9yqEor6wbWSj82rEdsJP9NvvYwCgYIKoZIzj0EAwID
SAAwRQIhAIMyeCWcWgd8SgT4tcRX1ghw7sPUeZy2FI5ehIQ4UM/sAiBY4c/h9uIX
CMNa/IaRMe9lCeDkun4CjKxJ0UvjrDUz9w==
-----END CERTIFICATE-----

Validate Device Certificate:
OK

Generated the manifest file 0123867d566ffb7701_manifest.json
MCHP Certificate processing completed successfully

The Notebook will also generate a manifest file to be uploaded into the public cloud of your choice (Google GCP, AWS IoT and Microsoft Azure).

After running this Notebook, it generates the required resources and program data zone with generated secrets, keys and certificates.

For this use case, Signer and Device certificates are required along with Root Public Key. When the MCHP certificates are available on device, MCHP cert definition files to be used. When the Cust certificates are generated and loaded to device, generated definition files to be used.

4 Use Case Prototyping

This hands-on lab is intended to demonstrate the usage of host to authenticate Accessory device. It uses asymmetric authentication where host reads certificates from accessory to validate chain of trust, followed by device private key.

This process uses a challenge-response model. In this model, host authenticates the accessory device based on response. Response (Signature) is calculated on the accessory device to prove that it holds the private key associated to its certificate shared to the host. Then the response will be verified by the host using Public key in Device certificate to authenticate the accessory.

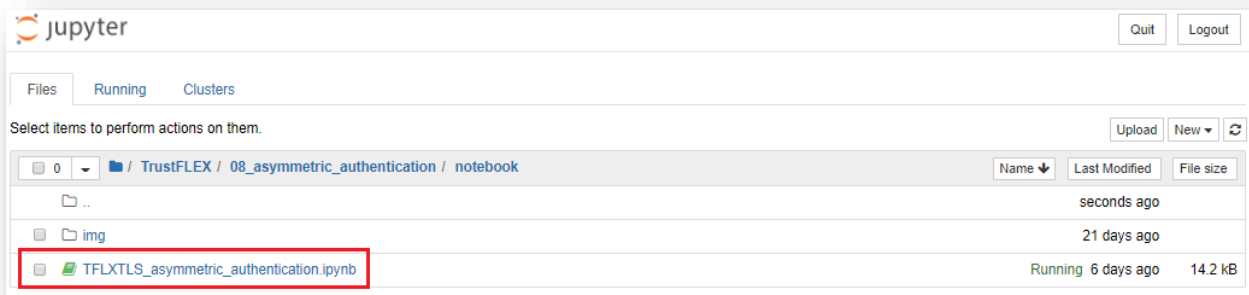
This lab is developed by simulating TrustFLEX device as Accessory and host to authenticate the accessory. Since Trust&GO use case demonstrate using MCHP certificates, here we would be Custom certificates to verify the authentication.

The resource generation for TrustFLEX device will load prototyping certificates to device along with custom certificates definition files.


Following sections provides detail steps to execute the usecase both on Jupyter Notebook and on Embedded project

4.1 Running Accessory Asymmetric Authentication example on Jupyter Notebook:


1. From the Jupyter Home page, navigate to **TrustFLEX\08_asymmetric_authentication\notebook\TFLXTLS_asymmetric_authentication.ipynb** notebook file and open it.



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


TFLXTLS_asymmetric_authentication
Last Checkpoint: Last Tuesday at 10:03 AM (autosaved)
Logout

File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3



Introduction

MUST READ: This Notebook tutorial will allow you to perform Asymmetric Authentication aka Node Authentication with TFLXTLS device.

In this use case example, we will authenticate an Object/Node. It can be an accessory, peripheral, battery, or cartridge. Generally, an object that is removable and replaceable by the consumer. The purpose of authenticating an object is to ensure that it is genuine and it is authorized to connect to a product. Another purpose is to prevent cloning and counterfeiting. Asymmetric authentication uses asymmetric key algorithms (also known as public key cryptography) where each entity has a public and private key.


Node authentication occurs between two devices in a host-client configuration when the client's identity must be verified before its connection to the host can be established.

Prerequisites:

This step of the tutorial will attempt to load all the necessary modules and their dependencies on your machine. If the modules are already installed you can safely step over the next Tutorial step.

Note

1. Installation time for prerequisites depends upon system and network speed.
2. Installing prerequisites for the first time takes more time and watch the kernel status. Following image helps to locate the Kernel status,



Watch status circle next to Python3. Dark indicates Kernel is running and Blank indicates Kernel is idle
3. Installing prerequisites gives the following error and it can be safely ignored. Functionality remains unaffected.
 - azure-cli 2.0.76 has requirement colorama~0.4.1, but you'll have colorama 0.3.9 which is incompatible.
 - azure-cli 2.0.76 has requirement pytz==2019.1, but you'll have pytz 2019.3 which is incompatible.

```

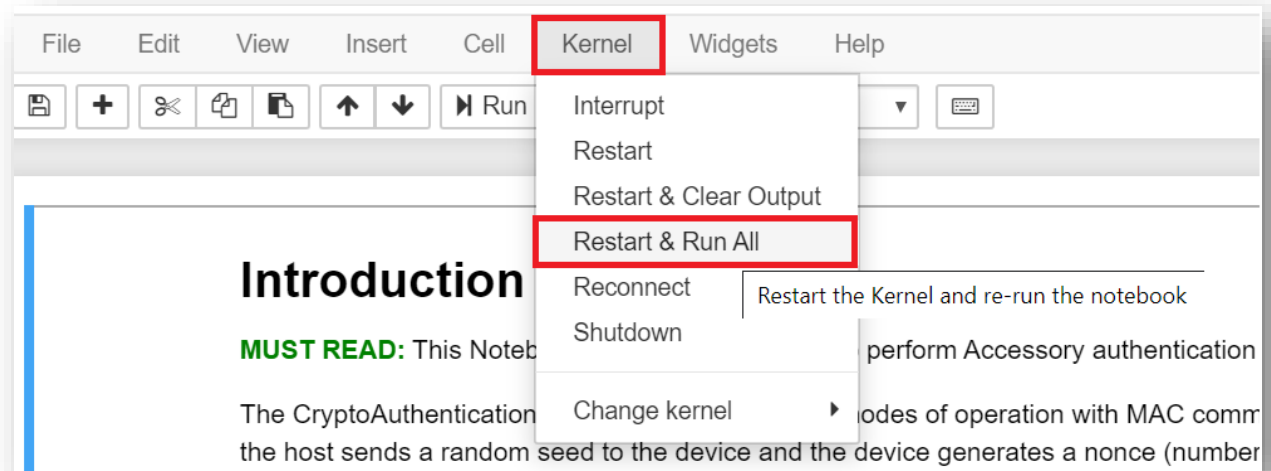
In [ ]: import sys, os

home_path = os.path.dirname(os.path.dirname(os.path.dirname(os.path.realpath(os.getcwd()))))
module_path = os.path.join(home_path, 'assets', 'python')
if not module_path in sys.path:
    sys.path.append(module_path)

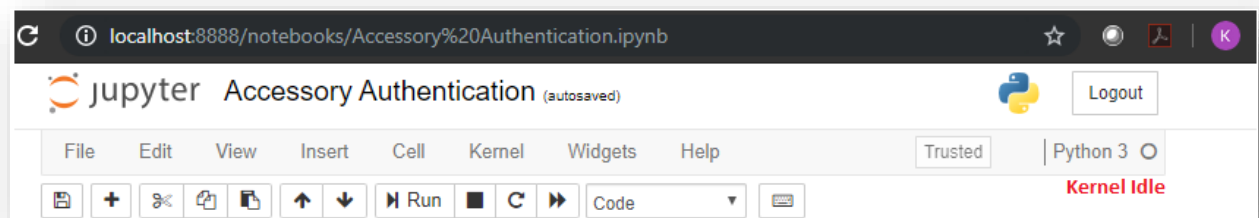
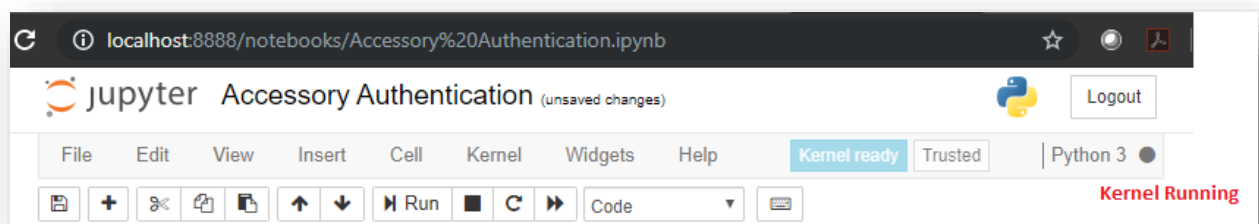
from requirements_helper import requirements_installer
obj = requirements_installer(os.path.join(home_path, 'assets', 'requirements.txt'))

```

2. Run All Cells by using Kernel -> Restart & Run All



3. It may take a while to complete, wait for the kernel to complete all processing i.e. from Kernel Running to Kernel Idle state (Check circle above **RED** text)



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

5. There are 2 major steps in this lab

Verify Certificate Chain

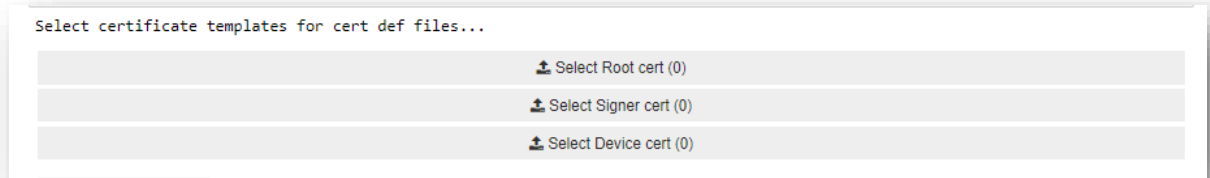
The certificate chain verification process includes reading the certificates from device, validating its signatures using CA's Public Key.

The host MCU contains the Root's Public Key stored on it. Certificate chain verification process starts by reading both Signer and Device certificates. Since, these certificates are stored as compressed certificates on the device, its required to rebuild them using certificate definition information.

These definition files can be regenerated based on the certificate templates.

Select certificate templates

On executing all the cells, the following buttons will be available to choose the certificate templates.



Select certificate templates for cert def files...

Select Root cert (0)

Select Signer cert (0)

Select Device cert (0)

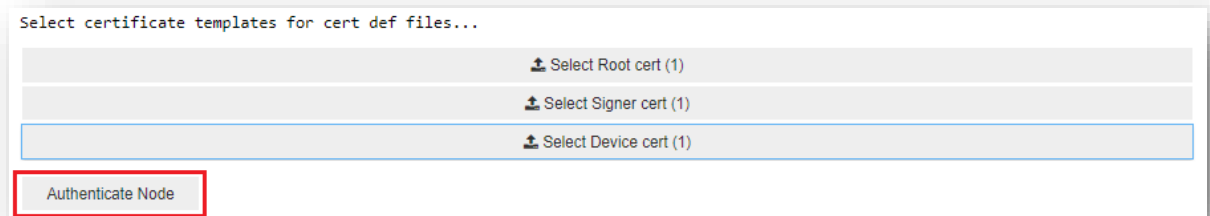
Since the root certificate or its details are not stored on Secure element, its required to provide the actual root that should be used to validate Signer certificate.

For Signer and Device certificates select the template certificates i.e. one generated in resource generation.

Authenticate Accessory/Node

In the above step, we have provided required certificates information to fetch and validate Signer and Device certificates from Secure element.

On clicking 'Authenticate Node' button, the process of rebuilding certificates, validation and Accessory authentication gets triggered.



Select certificate templates for cert def files...

Select Root cert (1)

Select Signer cert (1)

Select Device cert (1)

Authenticate Node

Client Rebuild Certificates:

In this step Signer and Device certificate definitions will be regenerated based on root certificate and template certificates provided.

Once the definitions are generated, host triggers set of commands to know the max size of the certificate and read the actual certificates from secure element. One can see root, signer and device certificates in the log.

Host to verify certificate chain:

On reading the certificates, host starts certificate chain verification. This step is currently limited to signature verification only.

In the resource generation root certificate is created as self-signed certificate. Hence, root certificate is validated using its own public key. Once its validated, Signer certificate is validated using root public key and device certificate is validated using Signer public key provided in the signer certificate.

Challenge-Response-Verify:

Once the certificates are validated, its important to check the accessory holds the original private key used to generate device certificate signing request. This will be done through a challenge response method.

Host generates as random challenge and sends to accessory for signing. Accessory would sign the challenge using the private key used for Device CSR. Once, host receives the signature from accessory, it verifies the same using public key provided in the Device certificate.

The result will be success only if the private and public key corresponds to each other, this indicates the connected accessory is authentic.

In case if private key is not associated to public key in the device certificate, this verification step would fail, this indicates the connected accessory is not authentic.

Pressing the button, turns it Green or Red. Green indicates that the device is authenticated by host and Red indicates the authentication is failed.

4.2 Running Accessory Asymmetric Authentication example on Embedded platform

This usecase can also be executed on Embedded platform. Once the resources are generated, both Atmel Studio and MPLAB projects provided can be used to run the application on CryptoAuth Trust Platform.

Note: This usecase requires resource generation notebook executed prior to using embedded projects.

4.2.1 Atmel Studio

1. Open **asymmetric_auth.atsln** project by navigating to Atmel Studio -> File -> open -> **TrustFLEX\08_asymmetric_authentication\c\studio\asymmetric_auth.atsln**

2. The application source code `asymmetric_auth.c` is available at **TrustFLEX\08_asymmetric_authentication\c\asymmetric_auth.c**. Other supporting files can be found under **assets\dependencies**
3. Program the Crypto 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, the firmware will do Accessory Asymmetric Authentication operation. Depending on the result, the CryptoAuth Trust Platform board's status LED will blink at different rates.

If **succeeds**, LED blinks once every second.

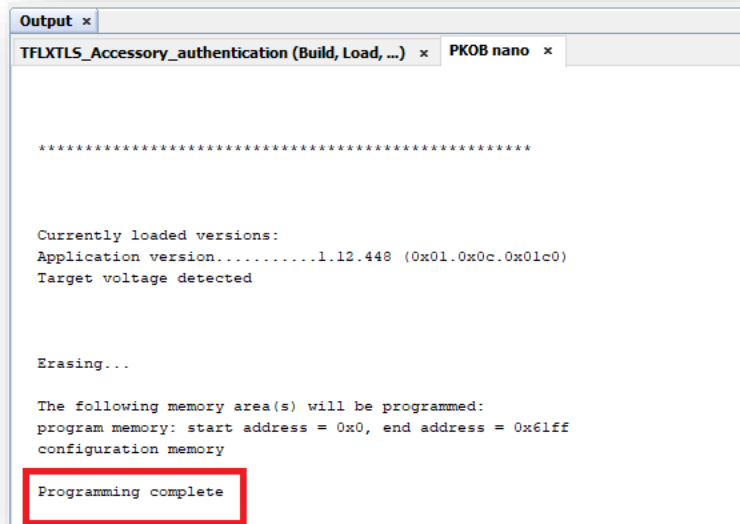
If **fails**, LED blinks five times every second.

It is also possible to view the Console messages by using applications like TeraTerm. Open the application with the COM related to CryptoAuth TrustPlatform with 115200-8-N-1 settings

4.2.2 MPLAB:

1. Open **asymmetric_auth.X** project by navigating to MPLAB -> File -> Open Project -> **TrustFLEX\08_asymmetric_authentication\c\mplab\asymmetric_auth.X**
2. The application source code `asymmetric_auth.c` is available at **TrustFLEX\08_asymmetric_authentication\c\asymmetric_auth.c**. Other supporting files can be found under **assets\dependencies**
3. Program the Crypto Trust platform by navigating to **asymmetric_auth -> 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.



```
*****

Currently loaded versions:
Application version.....1.12.448 (0x01.0x0c.0x01c0)
Target voltage detected

Erasing...

The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0x61ff
configuration memory

Programming complete
```

Once the programming is done, the firmware will do Accessory Asymmetric Authentication operation. Depending on the output, the Cryptoauth Trust Platform board's status LED will blink at different rates.

If **succeeds**, LED blinks once every second.

If **fails**, LED blinks five times every second.

It is also possible to view the Console messages by using applications like TeraTerm. Open the application with the COM related to CryptoAuth TrustPlatform with 115200-8-N-1 settings

File Edit Setup Control Window Help

```
Device serial number: 01 23 29 79 CB 29 3B A5 01
```

-----BEGIN CERTIFICATE

CLIENT: Rebuilt Device Certificate:

-----BEGIN CERTIFICATE

```
HOST: Signer certificate verified against root public key!
HOST: Device certificate verified against signer public key!
```

HOST: Generated challenge:

06	E6	31	DB	55	4B	B3	62	3A	37	71	5F	21	19	2B	F7
B5	B2	92	E3	2C	95	FB	BB	77	FB	63	29	70	5C	2A	5F

CLIENT: Calculated response to host challenge:

4F	DB	B1	06	07	FC	6B	2B	7B	37	4E	00	99	03	19	B1
83	9E	41	D3	AE	F1	86	14	E7	1A	F6	6C	48	0E	8A	20
C2	CF	48	44	53	1E	35	90	0A	9C	58	93	55	D1	08	B1
95	93	4A	12	69	EC	04	63	CE	61	8A	15	58	13	01	D4

HOST: Device public key from certificate:

99	D8	EA	9D	D1	1E	80	10	3B	8E	6F	0C	5D	14	BC	09
8D	3D	7A	CA	AE	C7	1A	A4	E1	89	A2	E9	90	DD	0A	78
6E	6A	64	E2	65	D0	70	8F	74	1A	16	A6	51	2C	0D	CC
36	EA	AD	17	EA	E2	0C	BA	AC	AB	45	32	FC	F3	74	5E

```
HOST: Device response to challenge verified!
```

Accessory device authenticated successfully

Execution completed with status 00

4.3 CryptoAuth TrustPlatform Factory reset

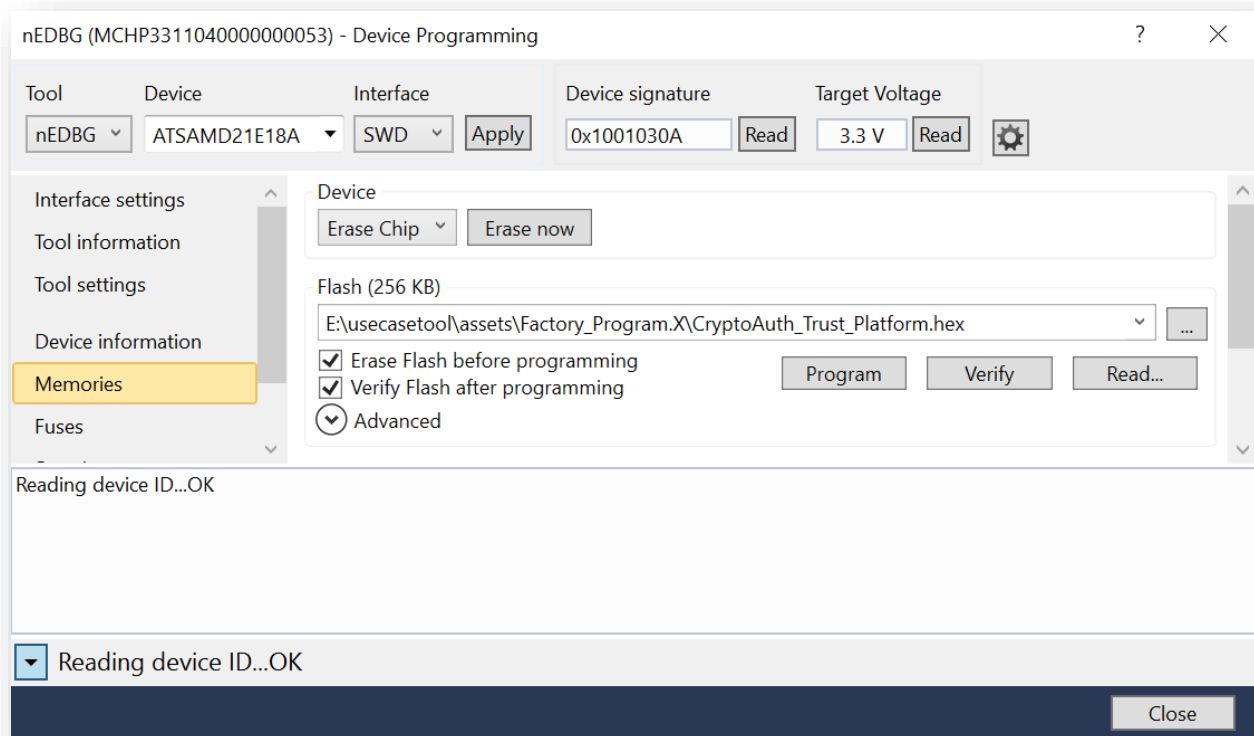
Once any of the embedded project is loaded to CryptoAuth TrustPlatform, the default program that enables interaction with TrustPlatform tools will be erased.

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

5 FAQ

1. What are the reasons for “**AssertionError: Can't connect to the USB dongle**” error?

There are many possibilities like,

1. Crypto Trust Platform is having different application than factory reset firmware. Refer to “CryptoAuth TrustPlatform Factory reset” section any usecase TrustFLEX Guide for reloading it
2. Check the switch positions on Crypto Trust Platform and/or ATECC608A Trust board
 - a. Correct Trust device should be connected and only one device of that type is allowed on the I2C bus. Multiple devices with same address results in error
3. Check USB connections to Crypto Trust Platform

2. How to reload factory default application to Crypto Trust Platform?

Refer to “CryptoAuth TrustPlatform Factory reset” section any usecase TrustFLEX Guide for reloading it.

3. Why does my C projects generates No such file or directory with ../../../../TFLXTLS_resource_generation/?

C project generates this error when the resources are not generated prior to using embedded projects. Running the resource generation notebook ensures these files and secrets are generated.

4. Before running any use case notebook and/or C project, why is it mandate to execute resource generation?

When resource generation notebook is executed, it generates and programs the required resources like secrets, keys and certificates. These are only prototyping keys and cannot be used for production. These keys will be used part of Usecase notebooks and C projects

5. How to know the resources being used in a use case?

Refer to individual Usecase description html for details on transaction diagrams, resources being used and other details. The resources required for given use case is mentioned in INFER CRYPTOGRAPHIC ASSETS section.

6. When should I select Custom certificates while doing resource generation?

Custom certificates are required when user wants to have their own root, signer instead of MCHP provided. The difference would be organization name, common name and validity are configurable

7. How to know whether C project is executing on Trust Platform or not after programming?

Once the programming is done, the firmware will do use case operation. Depending on the use case operation's output, the Crypto Trust Platform board's status LED will blink at different rates.

If use case operation succeeds, LED blinks once every second. If it fails, LED blinks five times every second.

It is also possible to view the Console messages by using applications like TeraTerm. Open the application with the COM related to Crypto Trust Platform with 115200-8-N-1 settings

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: 678-975-9614 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