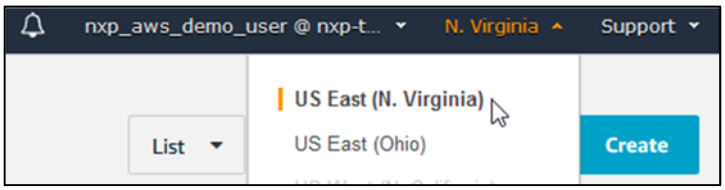
Lab 1 – Create a Thing on AWS

Registering a Thing on AWS IoT

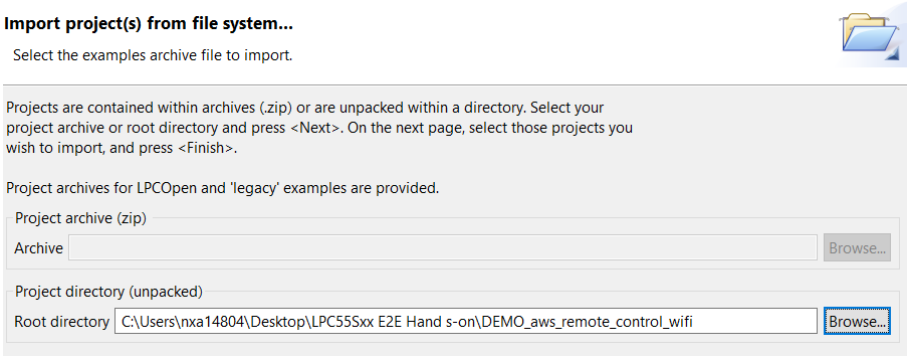
1. Log into AWS – Device Console using a **temporary** IAM account
   1.1. Goto: https://aws.amazon.com/
   1.2. Select "**My Account**" drop-down, and select "**AWS Management Console**"
   1.3. Enter the following account alias: **nxp-techday**
   1.4. Log in with the following credentials:
      - IAM user name: **nxp_aws_demo_user**
      - Password: **nxp_aws_pw**
2. Access the **IoT Device Management Console**
   2.1. In the AWS services search, begin typing in "**iot device management**", once the desire results appear in the result window, click the result to access the Management Console
   2.2. Verify that the Region in the upper-right corner is set to "US East"



   2.3. Ensure you are on the **Manage -> Things** page

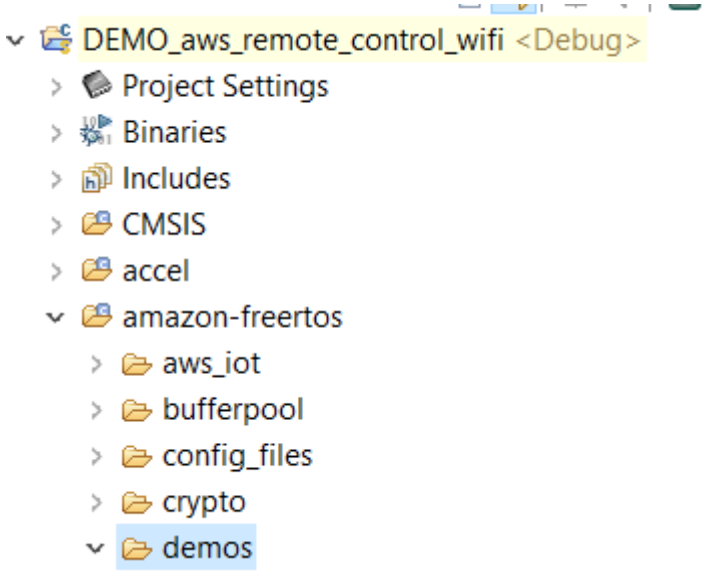
3. Create a new Thing
   3.1. Click the "**Create**" button to start process (Top Right)
   3.2. Click the "**Create a single thing**" option
   3.3. Provide a unique name for your thing. You will use this name throughout the class.
   3.4. Scroll-down and select **Next**
4. Generate security certificates for your Thing
   4.1. Use the "**One-click certificate creation**" option to by clicking the "**Create certificate**" button.
   4.2. **Important #1**: Download the certificate (*.pem) and private key (*.private.key)
   4.3. **Important #2**: Click the **Activate** button
   4.4. Next proceed to the "Attach a policy" step, selecting the "TechDay-GG-Group_Core-policy"
5. Finish the process by selecting "**Register Thing**"

Updating the Client Credentials (DEMO)

6. Import the **DEMO_aws_remote_control_wifi** SDK examples into the MCUXpresso IDE
   6.1. Open MCUXPresso IDE v11 (icon on desktop)
   6.2. Create a new Workspace LPCLAB1 for example.
   6.3. NOTE: The **SDK_2.x_LPCXpresso55S69** is pre-installed and is unzipped in the working folder
   6.4. Use the Quickstart Panel to **Import projects from file system…**
      6.4.1. Navigate to \Desktop\LPC55Sxx E2E Hand s-on \
      6.4.2. 

      6.4.3. Select **Next** then Click **Finish** to import the SDK example project
7. Explore the structure of the project and view the **readme.txt** file in the doc subfolder
   7.1. Middleware subfolders: amazon-freertos, mbedTLS, usb, wifi_qca
8. Generate the client credential keys source file
   8.1. From Chrome, see bookmarked Certificate Configurator (\LPC55Sxx E2E Hand s-on\SDK_2.6.2_LPCXpresso55S69\rtos\amazon-freertos\tools\certificate_configuration)
   8.2. Generate the **aws_clientcredential_keys.h** file
      8.2.1. Browse to and select the **Certificate PEM file (*-certificate.pem)** that you downloaded in Lab 1
      8.2.2. Browse to and select the **Private Key PEM file (*-private.pem.key)** that you downloaded in Lab 1
      8.2.3. Click the "**Generate and save**" button, which is also be save to the "Downloads" library
   8.3. Open a file explorer window and locate the **aws_clientcredential_keys.h** file in the **Downloads** library
   8.4. Use **CTRL-C** to copy the file **aws_clientcredential_keys.h** *(the actual file, not the contents)*
   8.5. Return to the MCUXpresso IDE and select the **amazon-freertos -> demos** subfolder. With the **include** folder selected press **CTRL-V** to paste the file into directory. You should get an Overwrite confirmation message, select **Overwrite**

8.6.



8.7. Double-click the **aws_clientcredential_keys.h** in the IDE to confirm that is contains a large block of random characters in the certificate sections

```
#define keyCLIENT_PRIVATE_KEY_PEM \
"-----BEGIN RSA PRIVATE KEY-----\n"\
"MIIEowIBAAKCAQEAsysLPe2CIzZrjFpduXaXp1sjQV5uua6gg2tU0/HCKsOx3nGo\n"\
"8rnkVXqltJW8yG2fa54MCVwTeJYXOI62WxXQV9KOHtvScfjkbpNyYmOHS21yf+xZ\n"\
"DIJiz/Y2hjbJbv9asateHyxVk/A7BfxIu/MzPooMOP9pSjmBbpm/uzzLfF0/RYyk\n"\
"IRjN88eohoXkFd+dnpz2B2cmq441M7Ya45uzznNEH9WkSmZR/LAFfCKfKNcdBKoJ\n"\
"lR5FyWN2y4ktFvz6WKNhJty9H7vV800TfigNrB3CaJU8pCZr0w9/fBGQ9MEccP8i\n"\
"R4SnOmr/cttmSDKRI+9ev+nWswjlRexrpJUPMQIDAQABAoIBAG0tzavvD15lyyO7\n"\
"7WEW41MYNmOfJ9n+j9GFliuRJsDof3sNx+YhYQK1ukRQ+rkoE/2AUdViRns0+vF8\n"\
"BaN7p3nc1L35NkQ9ZvX647kQ7WEJULyjSaBV0+DmKFOhDx6UfK/HvZNBcOS1f0Im\n"\
```

9. Obtain the AWS endpoint address **(IF NEEDED)**

   9.1. Return to the AWS IoT tab of your browser

   9.2. Click on the name of the Thing you just created
   Note: every class participants Thing will appear on this page, ensure you are only selecting your's throughout this lab.

   9.3. Select the **Interact** section and copy the **REST API Endpoint** from the HTTPS section:
   a3********.iot.us-east-1.amazonaws.com**

   9.4. Highlight and copy this string using CTRL-C

10. Configure additional Client Credentials

   10.1. Return to the MCUXpresso IDE application

   10.2. Double-click the **aws_clientcredential.h** file in the IDE Project Explorer (**amazon-freertos/demos** subfolder) to open it in the IDE editor *(note this is a similar name, but not the file we used in the previous step)*

   10.3. Paste the REST API Endpoint (also referred to as the **MQTT_Broker_Endpoint**) into **line 38**, keeping the quotation marks

   10.4. Update the **IOT_THING_NAME** on Line 43 to match the name of your Thing exactly, keeping the quotation marks

   10.5. Update WiFi SSID and Password **(IF NEEDED)**

      10.5.1. Line 58: replace "Paste WiFi SSID here." with **"NXPTRAINER"**

      10.5.2. Line 63: replace "Paste WiFi password here." with "**Use4Internet**"

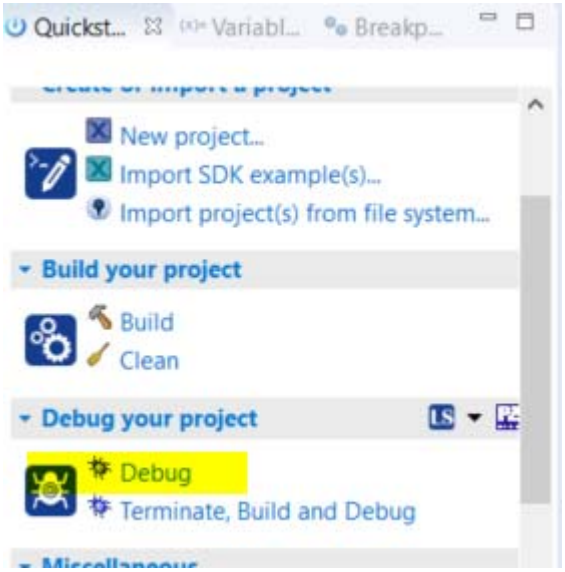
   10.6. Save changes to the **aws_clientcredential.h** file by selecting save button from top left.

11. From the **Quickstart Panel** select **Build** and confirm there are no errors once the build completes.

```
Building target: lpc54018iotmodule_aws_led_wifi_qspi_xip.axf
Invoking: MCU Linker
arm-none-eabi-gcc -nostdlib -L"C:\Users\nxa08675\Documents\MCUXpressoI
Memory region         Used Size  Region Size  %age Used
     BOARD_FLASH:      361548 B        16 MB      2.15%
           SRAMX:       57332 B       192 KB     29.16%
    SRAM_0_1_2_3:       96352 B       160 KB     58.81%
        USB_RAM:        5584 B         8 KB     68.16%
Finished building target: lpc54018iotmodule_aws_led_wifi_qspi_xip.axf

make --no-print-directory post-build
Performing post-build steps
arm-none-eabi-size "lpc54018iotmodule_aws_led_wifi_qspi_xip.axf"; # ar
   text    data     bss     dec     hex filename
 361548       0  102960  464508   7167c lpc54018iotmodule_aws_led_wifi


23:06:52 Build Finished (took 2m:5s.345ms)
```

General Debugging Procedures: Important

1. Power and connect the board using the P6 USB micro connector (closest to the 10pin JTAG header)
2. Use the **Quickstart Panel** to **Build** the application, ensure there are no compile errors
3. Use the **Quickstart Panel** to **Debug** the application



3.1.

4. Use the **Resume** (play) button to start the application
5. For first debug session: open Tera Term from windows tool bar

5.1. Select Serial

5.2. Select Setup from the Menu bar and then select Restore Setup

5.3. Navigate to the working folder and select the TERATERM.ini file (115200 baud, local echo is enabled)

6. The IDE “Restart” button that be used to restart the application without reflashing.

6.1. Press the “Restart” icon 

6.2. Jump to Step 4 above

Wi-Fi Connection

1. Run the Application

    1.1. Use the **Quickstart Panel** to **Build** and **Debug** the application

    1.2. Start the application using the **Resume** (play) button. .

2. Review the Console output to ensure you have connected to the WiFi Access point and have acquired an IP address 192.168.12.xxx

Shadow Client Connection to AWS

3. Open Shadow Page

    3.1. Click on your "**Thing**" on the IoT Device Management page

    3.2. Click on "**Shadow**" to view the cloud copy of the shadow

4. Response to Delta Messages

    4.1. Manually edit the Shadow Document

        4.1.1. Select the **Edit** action on the page

        4.1.2. Carefully update the desired LEDstate to 1

        4.1.3. Press **Save**

5. **You are now ready to proceed to LAB2**

Lab 2 – Create PUF data for Private PEM

Import the PUF AN Project

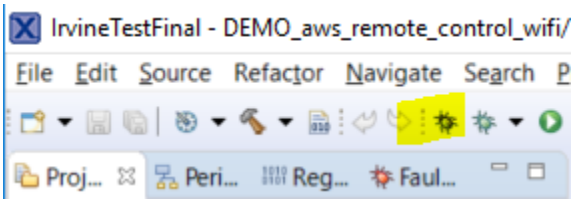
1.1. Use the Quickstart Panel to **Import projects from file system…**

1.1.1. Navigate to \Desktop\LPC55Sxx E2E Hand s-on \PART2\ lpcxpresso55s69_puf_aes_an12324sw

1.1.2. Select **Next**, then **Finish** to import the SDK example project

Download and debug

With the board connected press the download and debug button (Blue Bug)



Enroll PUF

From the Terminal Type 1 and press Enter

Store Activation Code in Flash

Type 2 and press enter

```
1136: fe 12 55 4a  2 36 95 ba ea 25 a2 c0 6d 42 d9 62
1152:  2 69 52 6a 6a 39 f8 54 61 da 20 cd a7 b3 a3 49
1168:  9 e6 5a 77 4e b0 53 fc 65 bc 77 50 53 b8 bc 8d
1184: 7b dd f4 d4 4a f5 1c bd
Store AC to
1. RAM activationCode
2. FLASH activationkeycode
2
```

START PUF

Type 2 and press enter

Choose Flash by **Typing 2** an pressing enter again

```
****************** PUF state ***********************
Allowed operations: Enroll  Start   SetKey  GetKey
                      no      no      yes     no

PUF Status:          Busy   Success  Error
                       no      yes     no
****************************************************

1. Enroll PUF
2. Start and load AC to PUF
3. Misc. PUF commands
4. Generate Key Code
5. Get Key from Key Code
6. Encrypt / Decrypt AES block
7. Back
2
Choose AC source
1. RAM
2. Flash
3. CMPA key store
2
Activation Code:
   0: 3f 41 c8 b2 16 17 d8 90 27 fa 3f 7f 19 2b 8c e2
  16:  b 83 f5 a9 52  d 6f 84 c1 98 7c 8f 75 f7 76 a6
```

GENERAT PUF KEY CODES

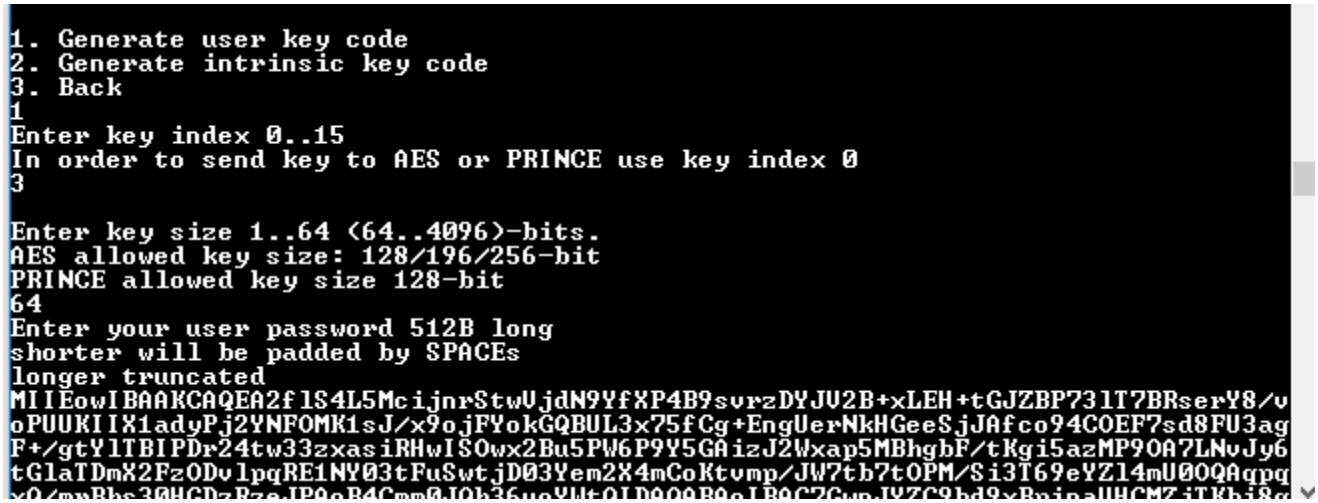
Type 4 and enter to Generate Key Codes

```
1. Enroll PUF
2. Start and load AC to PUF
3. Misc. PUF commands
4. Generate Key Code
5. Get Key from Key Code
6. Encrypt / Decrypt AES block
7. Back
4

****************** PUF state ***********************
```

Type 1 for user Generate User key code

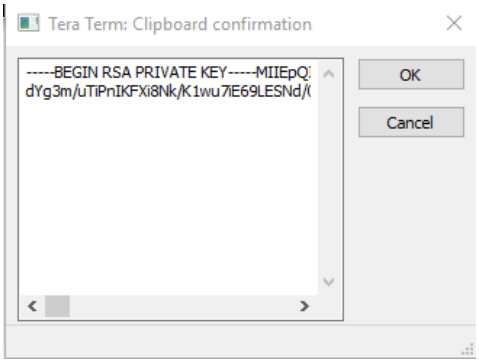
Type 3 for Index 3

Type 64 for maximum size (512 bytes)



Open Private PEM file in Notepad, choose edit, select all and copy

Return to the terminal window and right mouse click and press OK



Press Enter, then store the Key Code information to flash. The size of the PEM file requires 4 key codes. These must be entered in order

3 – Enter

4 – Enter

5 – Enter

6 - Enter

When done you will see the below Menu Screen

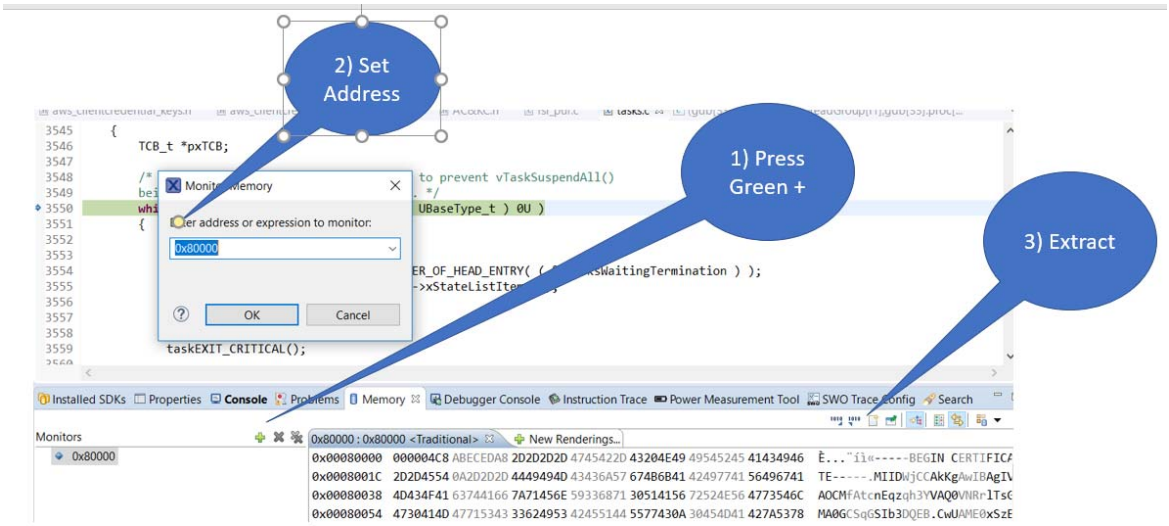
```
****************** PUF state **********************
Allowed operations: Enroll  Start   SetKey  GetKey
                      no      no      yes     yes

PUF Status:          Busy   Success  Error
                       no      yes     no
***************************************************

1. Generate user key code
2. Generate intrinsic key code
3. Back
```

HALT DEBUGGER and extract PUF Data

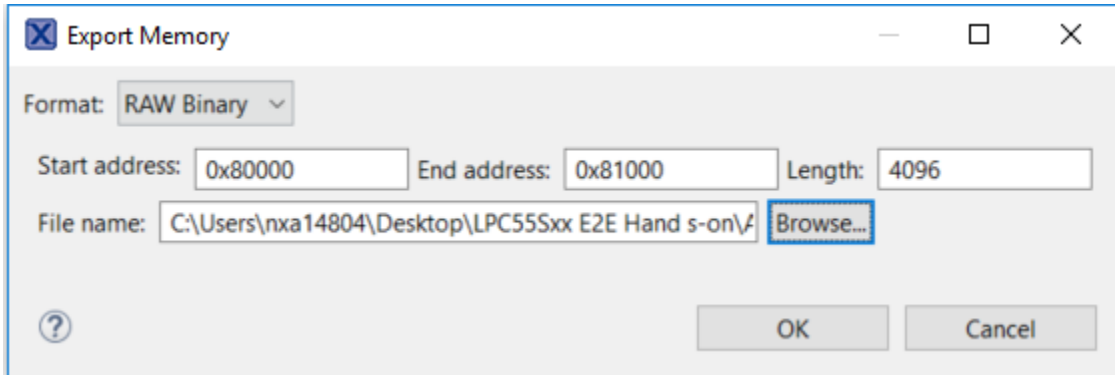
Press Pause button to Halt debugger

Extract data using Memory window – At the bottom of the debug window – First select the memory tab (Step 0)

Then follow the below guide. Set Address to 0x80000.



Set the format to **RAW Binary** (set by pull down), set size and file name

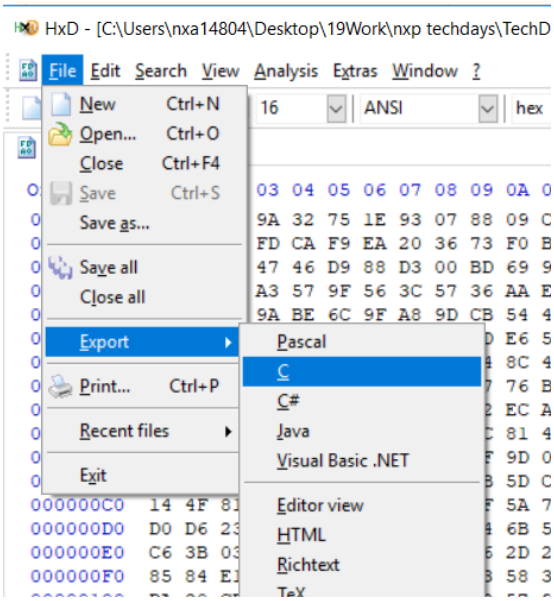


OPEN PUF Data in Hex Editor and Create C code

Open Hex Editor (see windows tool bar)

Drag the raw binary file into the Hex Editor

Select File to export C formatted data



Name the file and save it to your working folder

Press Stop to end the debug session.

Now You are ready for LAB 3

Lab 3 – Connect with PUF protected PEM

Import the WiFi with PUF project

1.1. Open a new Instance of MCUXpresso and Make a new Workspace “LPCLab3”

1.2. Use the Quickstart Panel to **Import projects from file system…**

1.2.1. Navigate to \Desktop\LPC55Sxx E2E Hand s-on \PART3\ PUF_DEMO_aws_remote_control_wifi

1.2.2. Select Next then **Finish** to import the SDK example project

Update public credentials

Three (3) items must be updated.

Thing Name, REST API endpoint (Broker Endpoint) and The Device Certificate.

Thing Name and REST API endpoint are in aws_client_credential.h.

Set these to match your Thing Name and Endpoint

```
/*
 * MQTT Broker endpoint.
 */
static const char clientcredentialMQTT_BROKER_ENDPOINT[] = "a3w3unlx9x1150-ats.iot.us-east-1.amazonaws.com";


/* Use of a "define" and not a "static const" here to be able to
* use pre-compile concatenation on the string. */
#define clientcredentialIOT_THING_NAME "Donnie"
```

In aws_client_credential_keys.h, update the Client Certificate PEM File information.

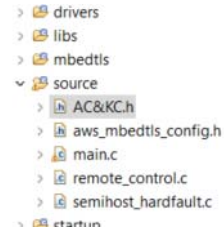
Copy the info from the previous AWS project that we used in LAB 1.

```
static const char clientcredentialCLIENT_CERTIFICATE_PEM[] =
"-----BEGIN CERTIFICATE-----\n"
"MIIDWjCCAkKgAwIBAgIVAOCMfAtcnEqzqh3YVAQ0VNRrlTsGMA0GCSqGSIb3DQEB\n"
"CwUAME0xSzBJBgNVBAsMQkFtYXpvbiBXZWIgU2VydmljZXMgTz1BbWF6b24uY29t\n"
"IEluYy4gTD1TZWF0dGxlIFNUPVdhc2hpbmd0b24gQz1VUzAeFw0xOTA1MjgyMDIz\n"
"MzFaFw00OTEyMzEyMzU5NTlaMB4xHDAaBgNVBAMME0FXUyBJb1QgQ2VydGlmaWNh\n"
"dGUwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDZ+VLgvkxyKOetK3BW\n"
"N031h9c/gH2y+vMNglXYH7EsQf60YlkE/veVPsFGx6tjz++g9RQoghfVp3I+PZg0\n"
"U4wrWwn/H2iMViiQZAFQvfHvl8KD4SeBR6s2QcZ55KMkB9yj3gI4QXux3wVTdqAX\n"
"7+C1iVMEg8Ovbi3DffPFqyJEfAhI7DHYG7k9bo/1jkYCLMnZbFqnkwGGBsX+0qCL\n"
"lrMw/04Dss28nLq0aVpMOZfYXM4O+WmpETU1jTe0W5LC2MPTdh6bZfiYKgq2+an8\n"
"lbu1vu048z9KLdPr15hmXiZTQ5ACqmrFD+asGGzfQcYPNHN4k8CgHgKabQlCHfq+\n"
"hha1AgMBAAGjYDBeMB8GA1UdIwQYMBaAFO7iOlmHvHACKrUJi6wzeqQlLunOMB0G\n"
"A1UdDgQWBBT7PR/zFHMuOH9DashNypffeEUwQDAMBgNVHRMBAf8EAjAAMA4GA1Ud\n"
"DwEB/wQEAwIHgDANBgkqhkiG9w0BAQsFAAOCAQEAtJWsMa8v0hHkQhPb19GvEaOl\n"
"zWpJsVk9Q1mKGj3yFaWsgseS02pVcK1ZZv+u0E22/g95dnjHjh2cr5WhjnEBYw0d\n"
"ChMBg1F07k1yIbxlUWU5m+z9+ApB8QAymiOZ0LYi3APPRlgufHsjN5i7JLelda+x\n"
"qrOHaVn57gOl0kmB692i/8azy3e0Ut4KXxRG2AqSPCQunVEmeeRpK/ogWHwcP4WV\n"
"tsnqVnGkNIwnVO8pyNcz/MafRhjbrOhJ8S4o9uIrfJkzJi8yBa0B/TWiM/iq2mMC\n"
"6x76XUXhn6yUt13W/GTsYFqu4TWdm/cXGsz29VF/2a8Xqh8J2WoLI4VYGYdUiA==\n"
"-----END CERTIFICATE-----";

/*
 * PEM-encoded client private key.
 *
 * Must include the PEM header and footer:
 * "-----BEGIN RSA PRIVATE KEY-----"
 * "...base64 data..."
 * "-----END RSA PRIVATE KEY-----";
 */
static const char clientcredentialCLIENT_PRIVATE_KEY_PEM[] =
"-";

/*
```

Update the PUF Data

Open the C file data created by Hex Editor (you can open from MCUXpresso, File-> Open File

Copy Array data and paste inside AC&KC.h file that is in the source folder




```
__RODATA(Flash2) static const uint8_t AC_DATA[4096] = {

    0x3F, 0x41, 0xC8, 0xB2, 0x16, 0x17, 0xD8, 0x90, 0x27, 0xFA, 0x3F, 0x7F,
    0x19, 0x2B, 0x8C, 0xE2, 0x0B, 0x83, 0xF5, 0xA9, 0x52, 0x0D, 0x6F, 0x84,
    0xC1, 0x98, 0x7C, 0x8F, 0x75, 0xF7, 0x76, 0xA6, 0x9C, 0x45, 0x0A, 0x80,
    0xFB, 0xCD, 0xD7, 0x5D, 0x2F, 0xB5, 0xC2, 0x02, 0x4F, 0x97, 0x2D, 0xBF,
    0x2F, 0x40, 0x5B, 0x1B, 0x84, 0x06, 0xD1, 0x18, 0xA8, 0x87, 0x45, 0x4B,
    0x79, 0x27, 0x6E, 0x63, 0x51, 0x06, 0xDD, 0xBC, 0x95, 0xBE, 0xFB, 0x96,
    0x5D, 0x91, 0x6F, 0x46, 0xF7, 0x81, 0xA4, 0x17, 0xA0, 0x57, 0x9C, 0xEB,
    0x89, 0x9A, 0x7B, 0xFC, 0xED, 0xDB, 0xC6, 0xA8, 0x26, 0xB1, 0xCE, 0x43,
    0xB0, 0xE7, 0x2A, 0x06, 0x19, 0x16, 0xA7, 0xDB, 0x58, 0xBF, 0x09, 0x79,
    0xDB, 0xF2, 0x27, 0xFF, 0x2E, 0x06, 0x35, 0x33, 0x62, 0x79, 0x5C, 0x86,
    0xF7, 0x50, 0x9B, 0x9E, 0xDE, 0x33, 0xD5, 0xF2, 0xE2, 0x10, 0xC5, 0x4C,
    0xE9, 0x99, 0x6E, 0xFC, 0xD9, 0x69, 0x2D, 0xF1, 0x78, 0xA3, 0x92, 0x3D,
    0x8B, 0xA3, 0x06, 0x37, 0x82, 0x1F, 0xDA, 0x0C, 0x9C, 0xFF, 0xDA, 0x0B,
    0xA1, 0xB7, 0x10, 0xD3, 0x9C, 0x96, 0xF7, 0x40, 0xC3, 0xD9, 0xF2, 0x6B,
    0x39, 0x67, 0x0A, 0x71, 0x3E, 0x16, 0x1B, 0x8A, 0x96, 0xAE, 0x1D, 0x80,
    0xA6, 0xE0, 0x7B, 0xE0, 0x3D, 0x71, 0x28, 0x58, 0x0C, 0xA3, 0x76, 0xD3,
```

Download and Debug with PUF info only

With the project highlighted - Press the Blue Bug and run the project. 

```
Key:

   0: 6d 74 78 2f 49 6d 38 4e 2b 55 43 76 55 64 6d 42
  16: 51 41 66 67 6d 46 33 6b 4a 51 61 33 70 63 4c 4a
  32: 41 5a 71 37 32 63 58 52 75 42 74 33 4b 38 48 4d
  48: 6c 41 5a 4a  0 20 20 20 0 0 [Tmr Svc] Starting key provisioning...
1 0 [Tmr Svc] Write root certificate...
2 10 [Tmr Svc] Write device private key...
3 424 [Tmr Svc] Write device certificate...
4 435 [Tmr Svc] Key provisioning done...
5 436 [Tmr Svc] Starting WiFi...
6 1705 [Tmr Svc] WiFi module initialized.
7 6734 [Tmr Svc] WiFi connected to AP nxp.
8 6735 [Tmr Svc] IP Address acquired 192.168.43.120
9 6744 [AWS-RemoteCtrl] [Shadow 0] MQTT: Creation of dedicated MQTT client succe
eded.
10 7133 [MQTT] Looked up a3w3unlx9x1150-ats.iot.us-east-1.amazonaws.com as 54.24
2.222.183
11 23959 [AWS-RemoteCtrl] [Shadow 0] MQTT: Connect succeeded.
12 24169 [AWS-RemoteCtrl] [Shadow 0] MQTT: Subscribe to accepted topic succeeded
.
13 24381 [AWS-RemoteCtrl] [Shadow 0] MQTT: Subscribe to rejected topic succeeded
.
14 24390 [AWS-RemoteCtrl] [Shadow 0] MQTT: Publish to operation topic succeeded.
```