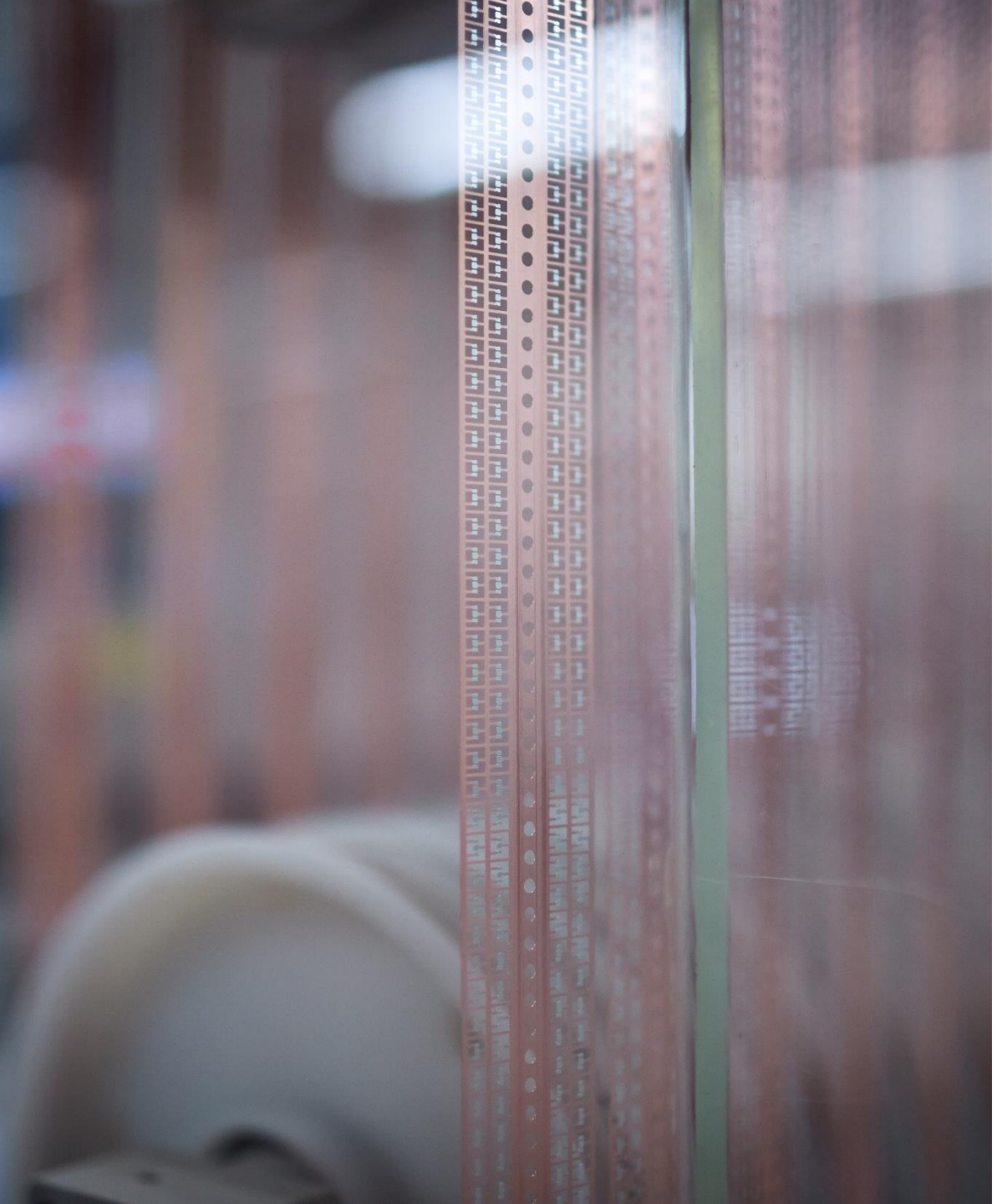
A71CH – Plug & trust for IoT

Session 2: Getting started with A71CH product support package and i.MX6UltraLite examples

JORDI JOFRE
26/04/2018

PUBLIC

SECURE CONNECTIONS FOR A SMARTER WORLD



A71CH – Plug & trust for IoT

Session 1: A71CH product introduction

Get familiar with A71CH key security features, key benefits, use cases and product support package.

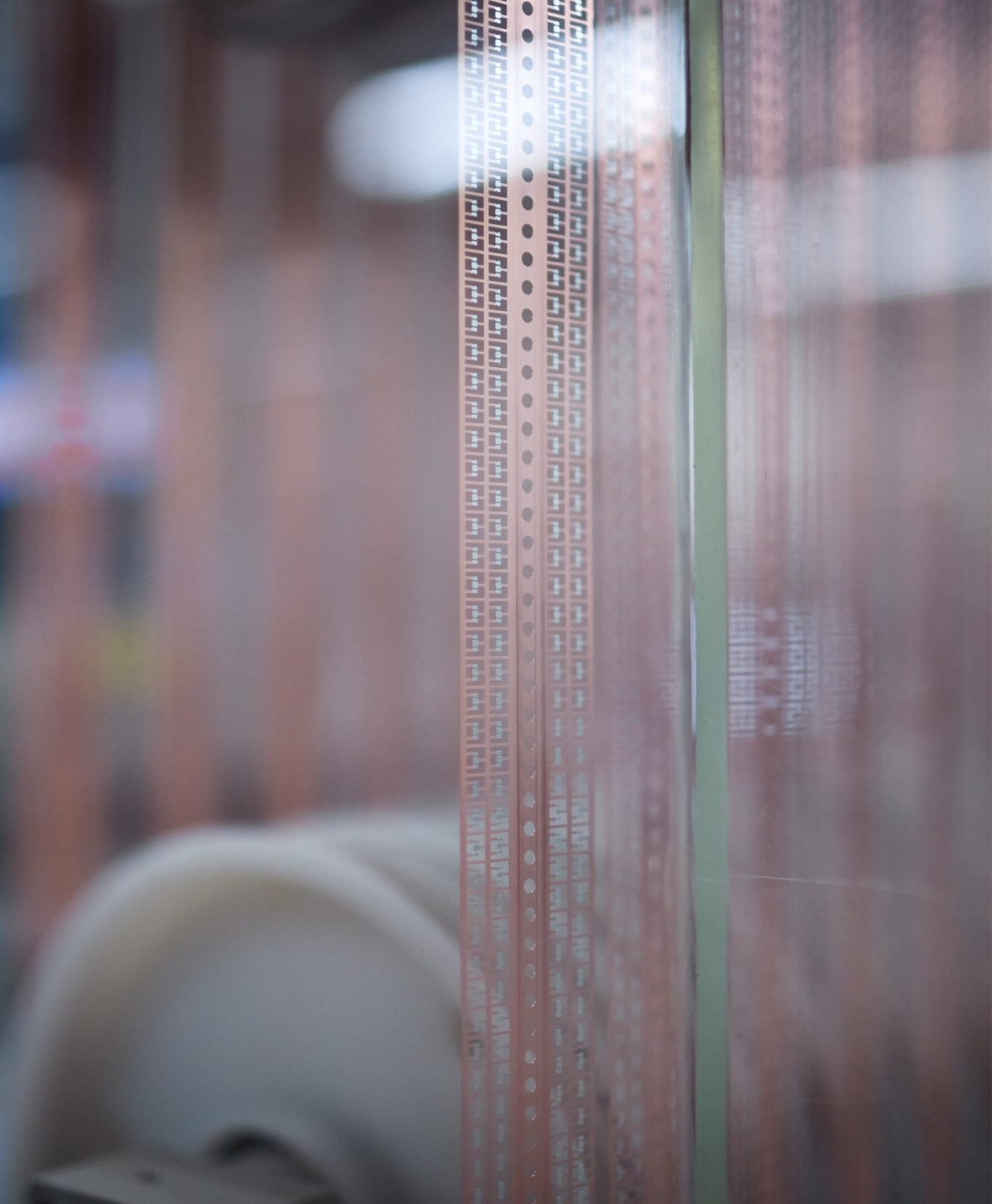
April 24th, 2018 - 10 AM CEST and 08 AM PDT

Recording:
https://register.gotowebinar.com/recording/5952422091538558979

Session 2: Getting started with A71CH product support package and i.MX6UltraLite examples

Learn how to get started with A71CH and its support package, including an example with i.MX6UltraLite.

April 26th, 2018 - 10 AM CEST and 08 AM PDT

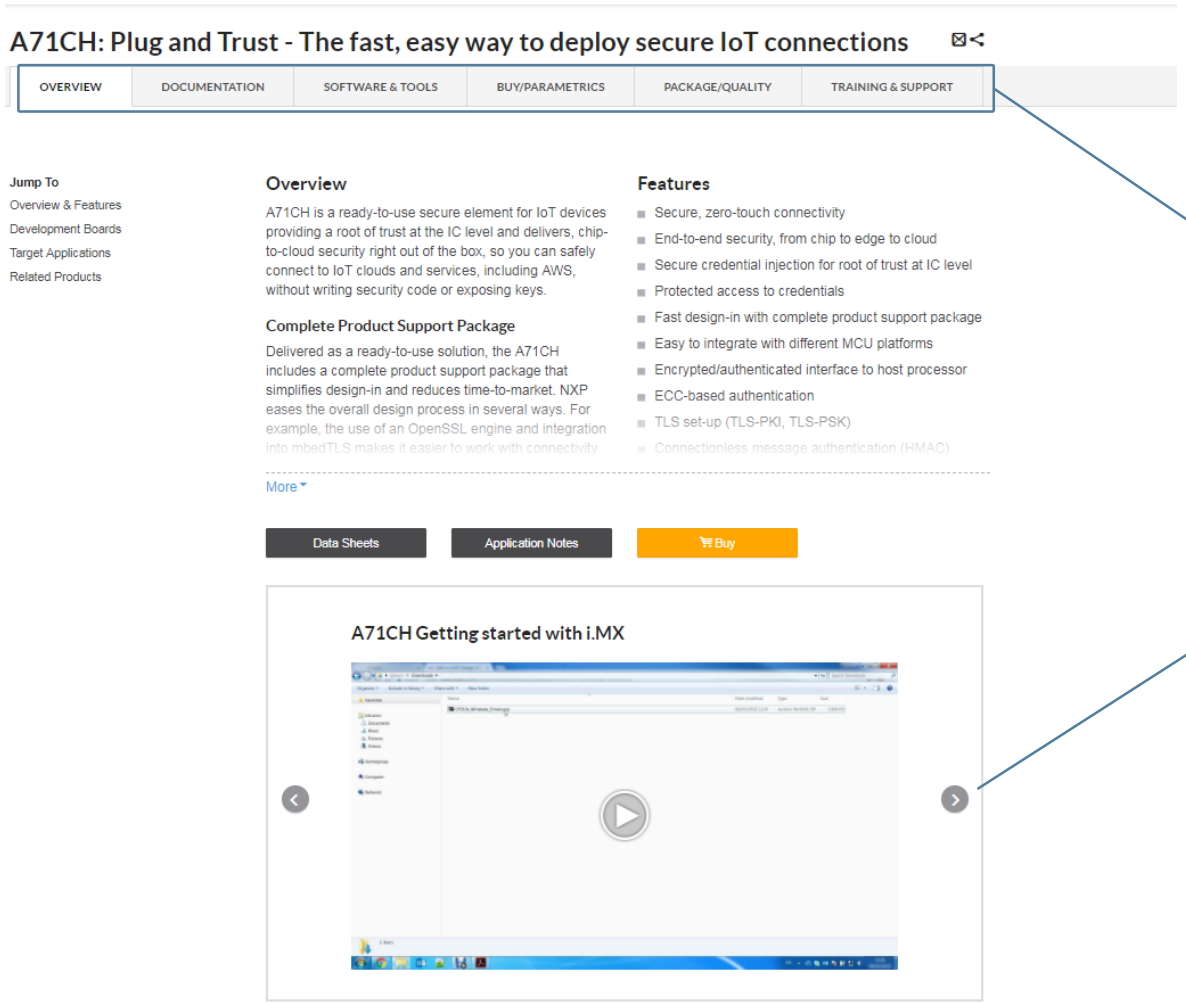


Agenda

- Finding A71CH product support package.
- Getting started with i.MX6UltraLite.
- Using A71CH Configure tool.
- Using A71CH OpenSSL Engine examples.
- Using A71CH Host API examples.

Finding A71CH product support package

A71CH product website



How to get there: www.nxp.com/A71CH

Navigate through the tabs:

- Overview
- Documentation
- Software and tools
- Buy / parametrics
- Package / quality
- Training and support

Highlighted material:

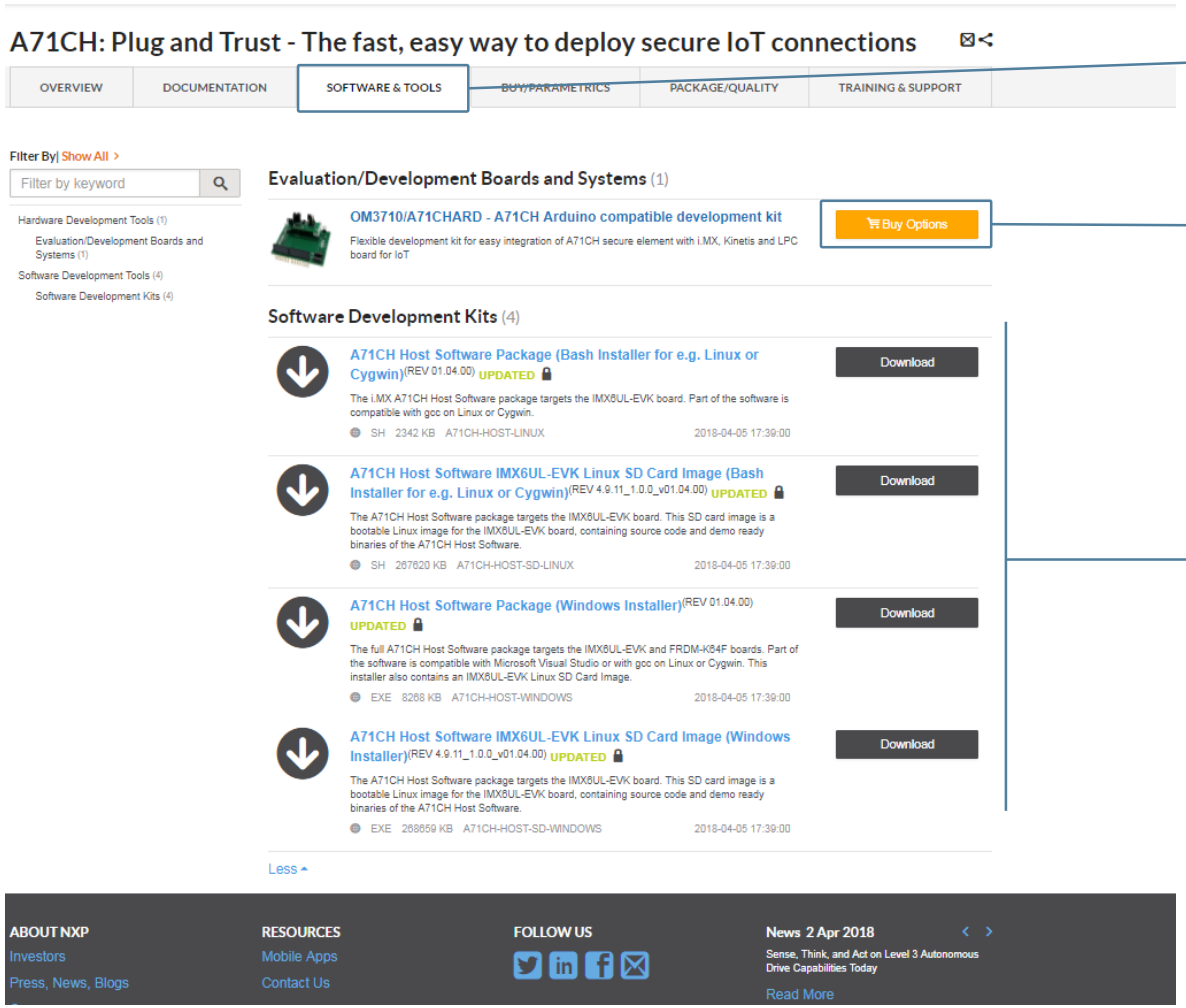
- Videos
- Architecture diagrams
- Interviews
- Etc.

Scroll down for more:

- Target applications
- Related products
- Etc.

Training Mobile Knowledge

A71CH product website - design tools



How to get there:

www.nxp.com/products/:A71CH?tab=Design_Tools_Tab

Order your A71CH Arduino compatible development kit

Download software and software images:

- A71CH Host software package installer for Windows
- A71CH Host software package installer for Linux
- A71CH Host software package and i.MX6UL SW image installer for Windows
- A71CH Host software i.MX6UL SW image installer for Linux

Training Mobile Knowledge

A71CH getting started with i.MX6UltraLite

A71CH getting started with i.MX6UltraLite

What do you need?

A71CH Arduino compatible development kit

Contents
A71CH mini PCB board
Arduino interface header board

Part number: OM3710/A71CHARD
12NC: 935368997598
URL: www.nxp.com/OM3710

i.MX6UltraLite evaluation Kit

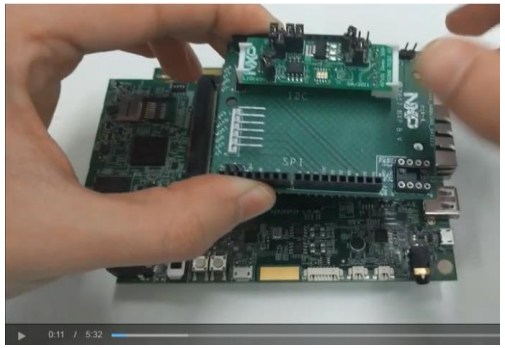
Contents
i.MX6UltraLite CPU board
Base board
Power supply
USB Cable
Micro-SD card

Part number: MCIMX6UL-EVKB
12NC: 935328353598
ww.nxp.com/products/i.mx6ultralite-evaluation-kit:MCIMX6UL-EVK

Development PC

Laptop
Standard laptop running Linux or Windows environment

Video tutorial and / or AN12129



URL video tutorial:
www.nxp.com/video/:A71CH-STARTED-IMX

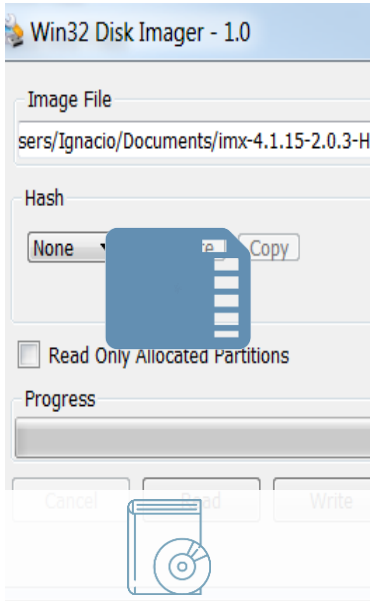
URL AN12129:
www.nxp.com/docs/en/application-note/AN12119.pdf

Training Mobile Knowledge

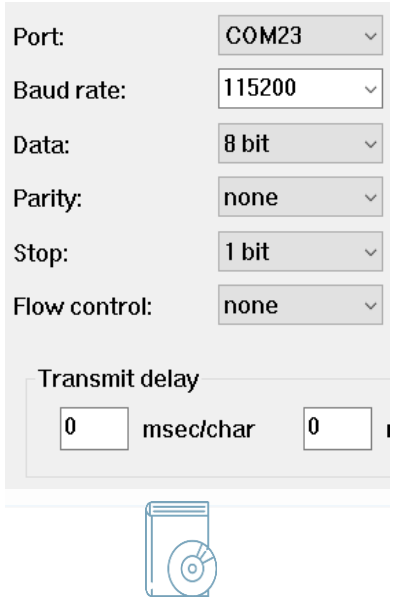
A71CH getting started with i.MX6UltraLite

Steps

Prepare the hardware

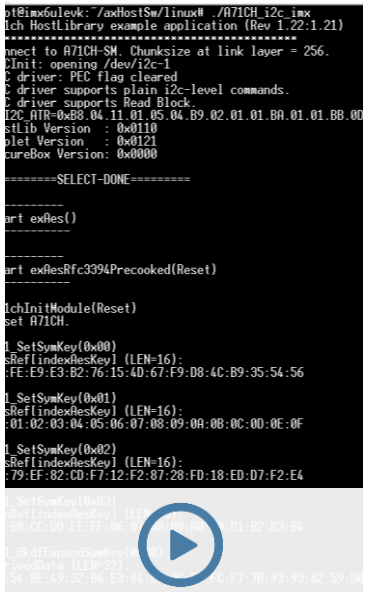


Flash the microSD card image



Install a terminal emulator

Boot the system



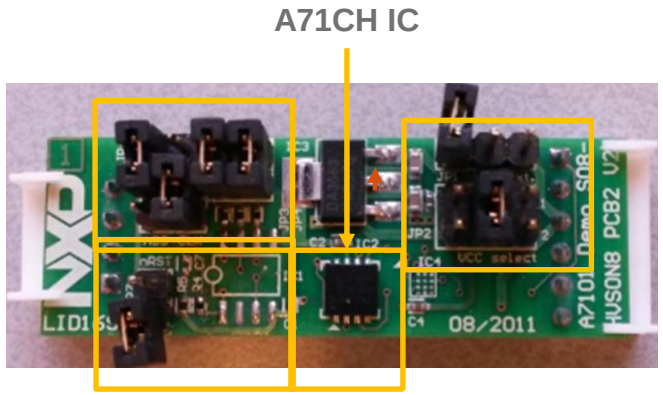
Run the sample applications

Training Mobile Knowledge

A71CH getting started with i.MX6UltraLite

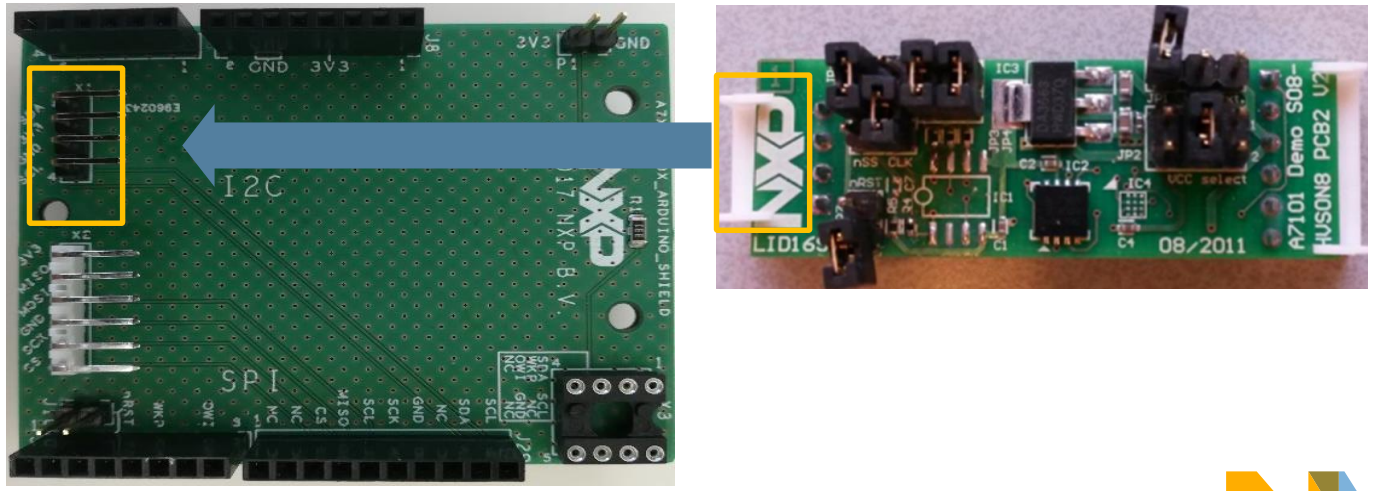
Hardware preparation

1 Make sure A71CH Mini PCB jumpers are configured for I²C interface



Jumper	Setting	Use
JP1	Not set	External VCC connection
JP2	3-4	Connect A71CH to 3.3V regulator on miniPCB
JP3	Set	Connect I2C SDA pull-up resistor
JP4	Set	Connect I2C SCL pull-up resistor
JP5	1-2	Use I2C address 0x92/0x93
	2-3	(Default) Use I2C address 0x90/0x91
JP6	1-2	Active I2C interface
JP7	Not set	(Default) A71CH operates
	Set	A71CH reset

2 Plug the A71CH Mini PCB board to the Arduino header adaptor

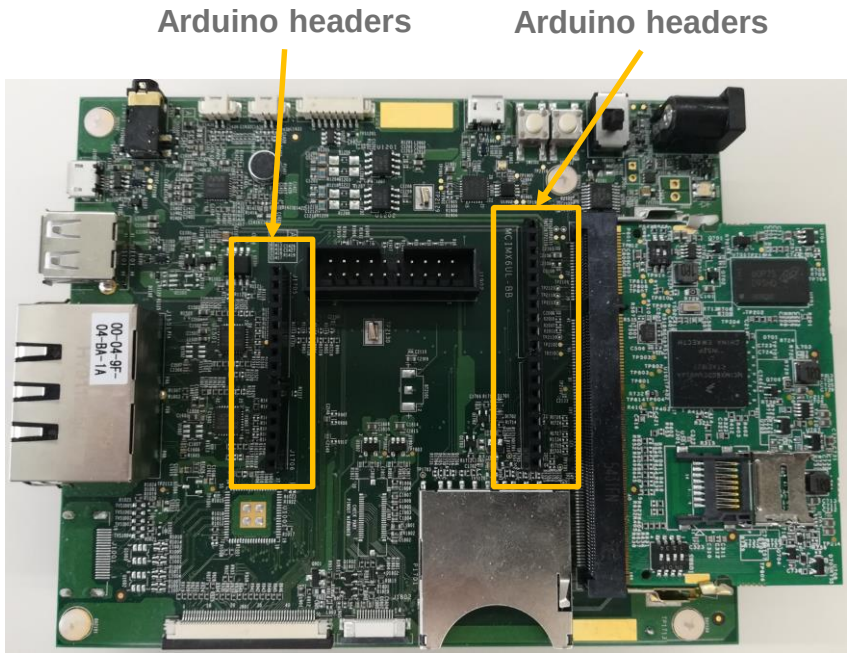


Training Mobile Knowledge

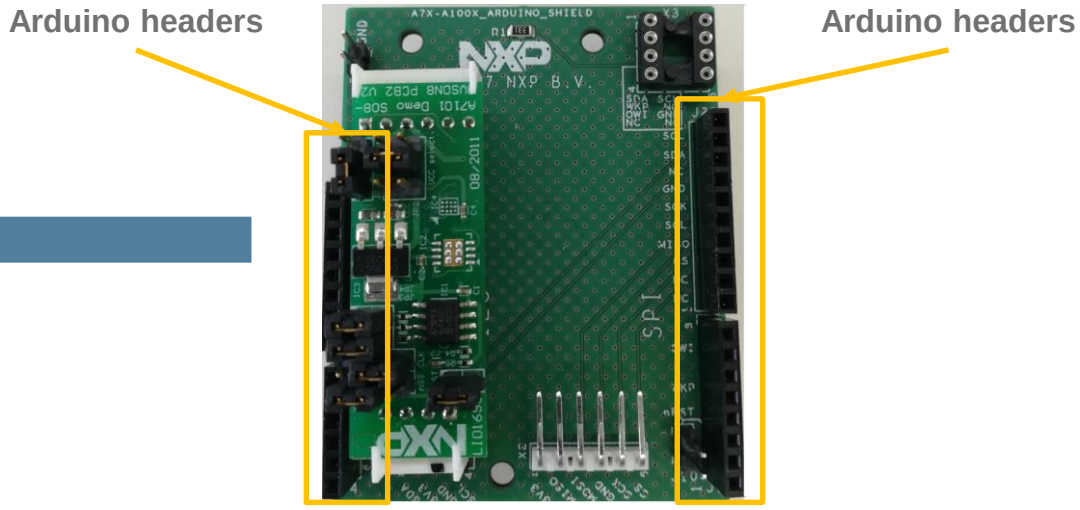
A71CH getting started with i.MX6UltraLite

Hardware preparation (II)

3 Plug the A71CH into the i.MXUltraLite board using the Arduino adaptors



***Note:** Might require soldering of Arduino headers



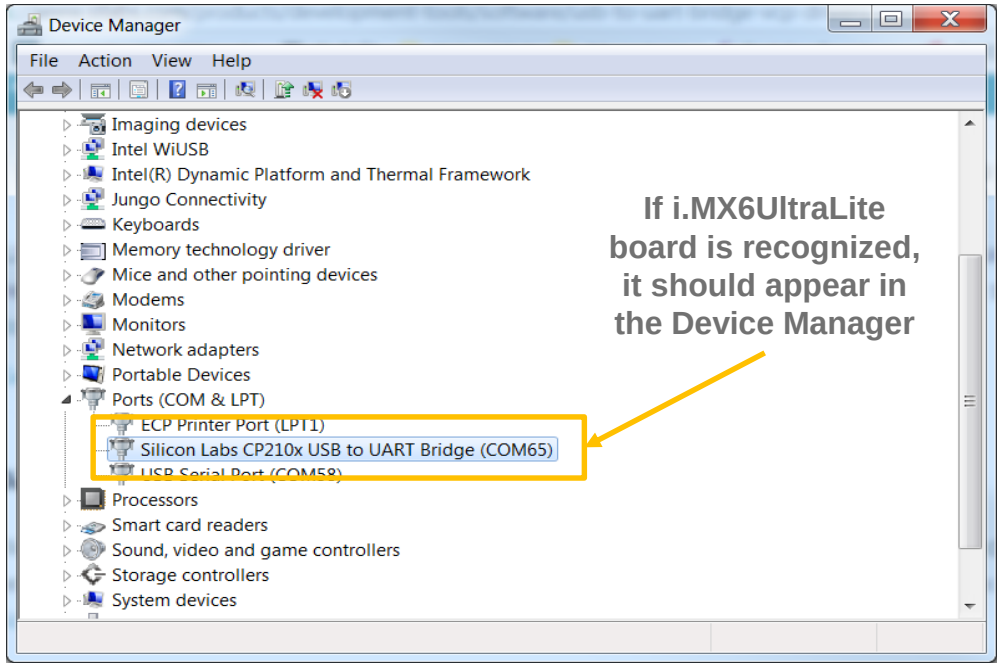
***Note:** The Arduino shield board comes with male connectors below

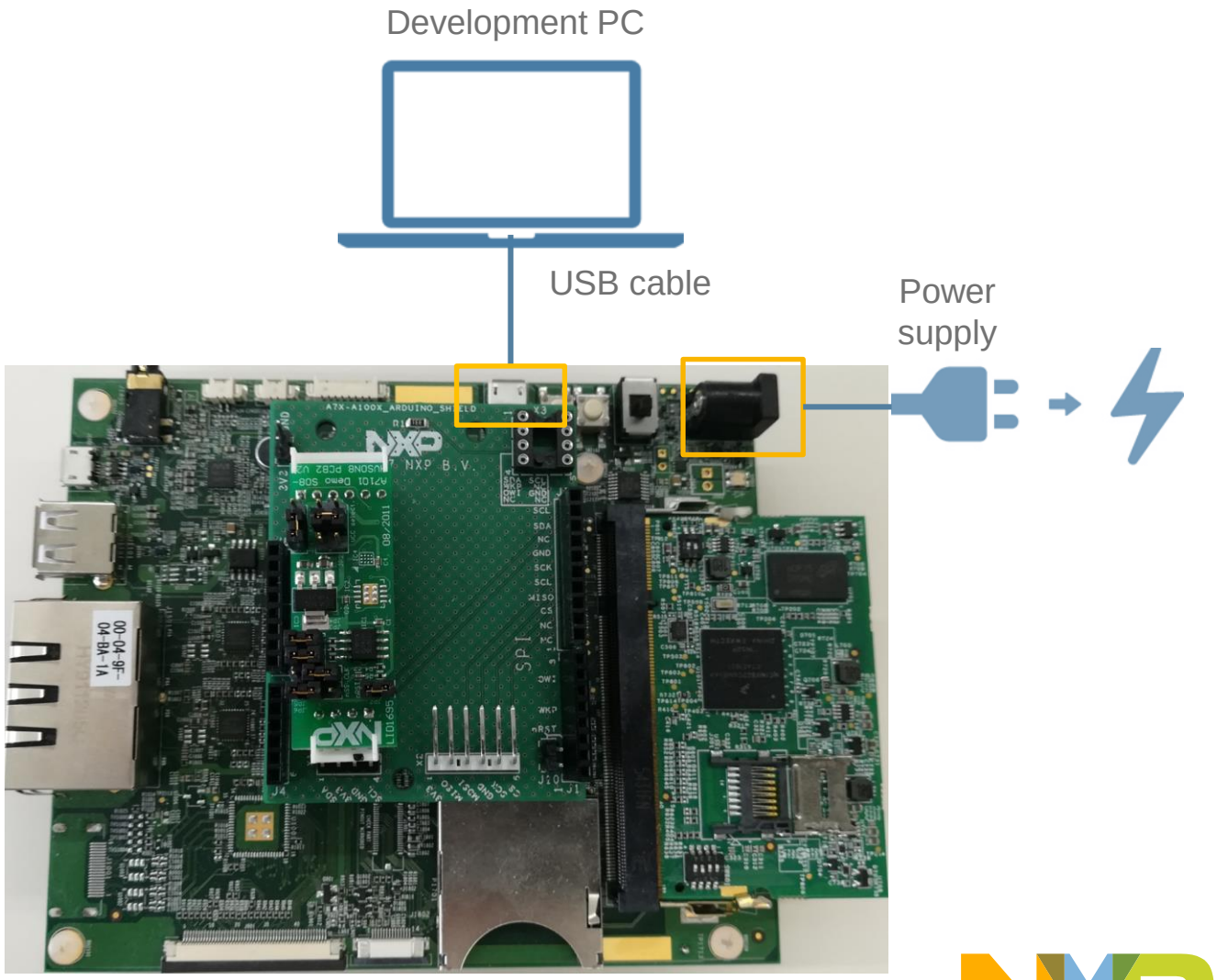
Training Mobile Knowledge

A71CH getting started with i.MX6UltraLite

Hardware preparation (III)

4 Install ***USB to UART Bridge Virtual COM Port*** drivers (if needed)





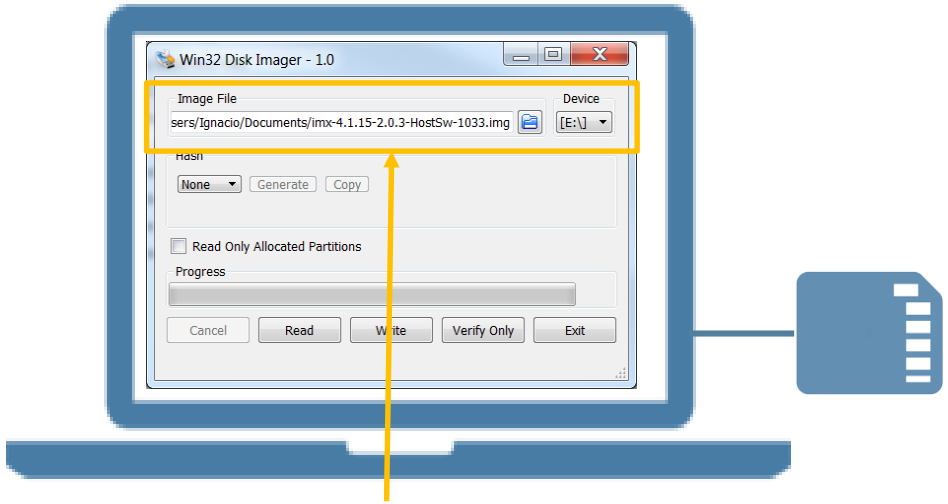
USB to UART Bridge Virtual COM port driver:
https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

A71CH getting started with i.MX6UltraLite

Flash the SD card image

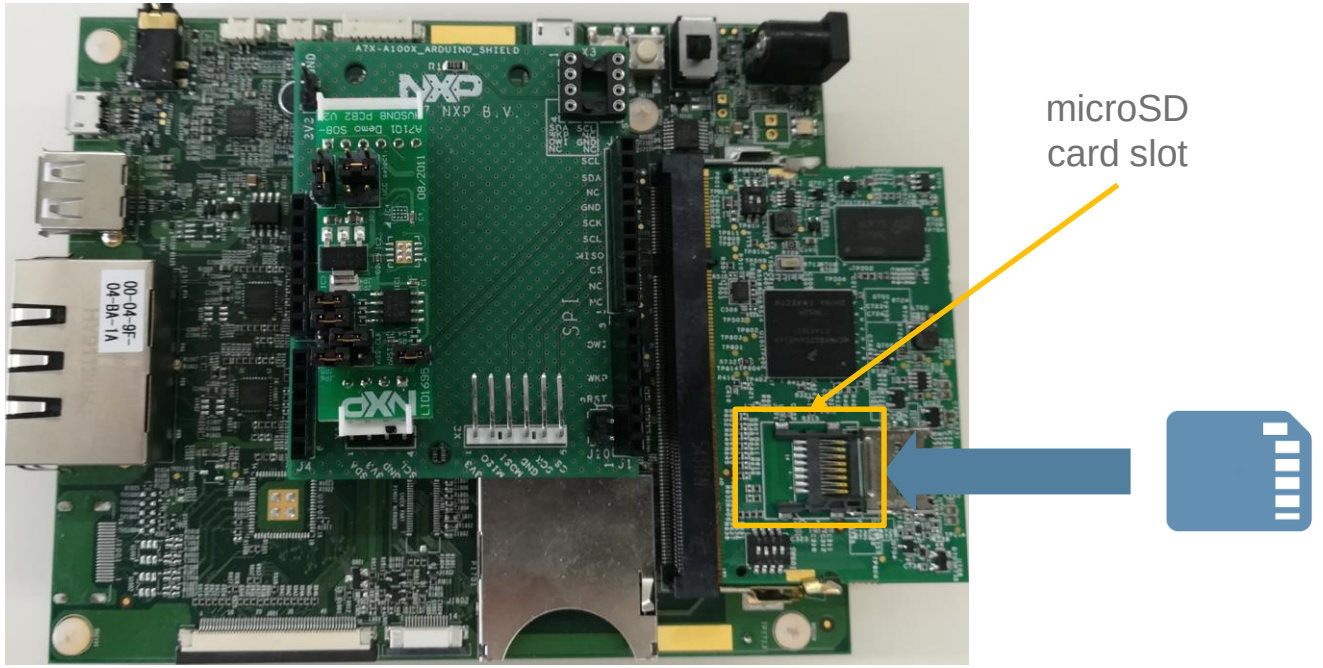
5 Flash the microSD card with the NXP-prepared Linux image

Development PC



Use Win32 Disk Imager, or any other software to flash the Linux image into the microSD card

The NXP-prepared Linux image is ready to run in i.MX6UltraLite board and includes A71CH Host Library and software examples integrated.



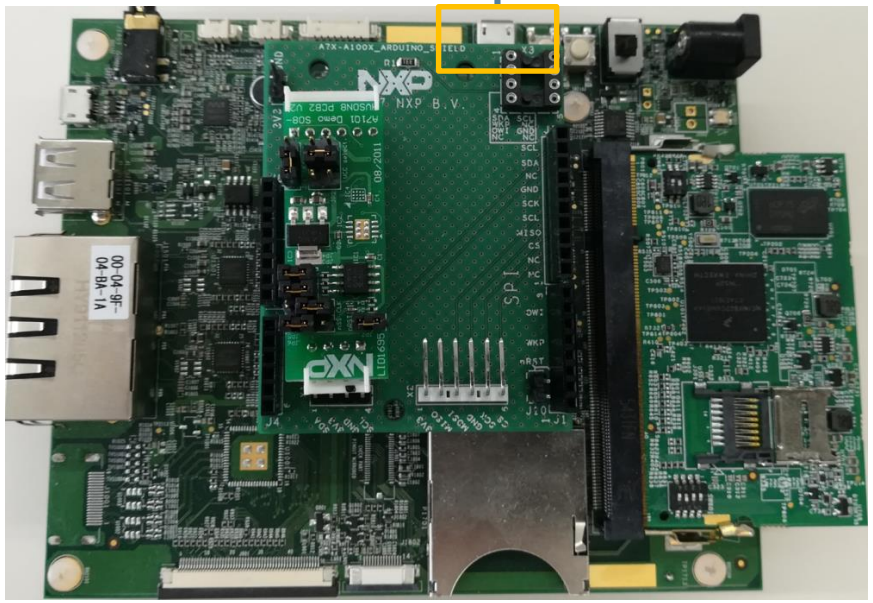
Linux SD card image for i.MX6UltraLite can be downloaded from:
www.nxp.com/products/:A71CH?tab=Design_Tools_Tab

A71CH getting started with i.MX6UltraLite

TeraTerm terminal application install & configuration

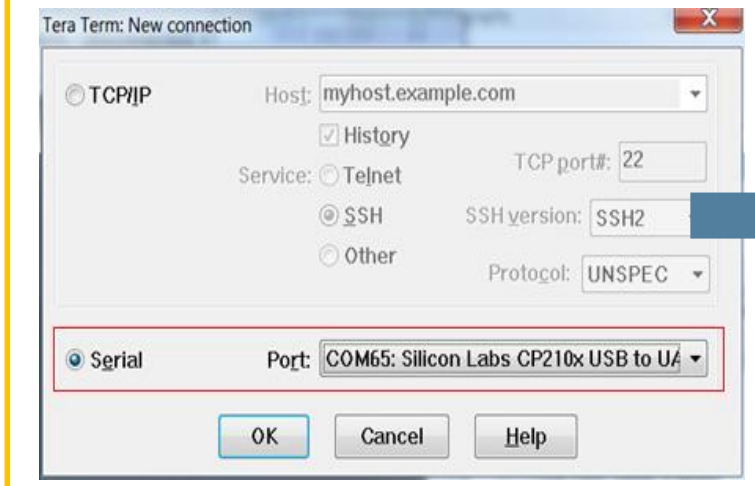
Development PC

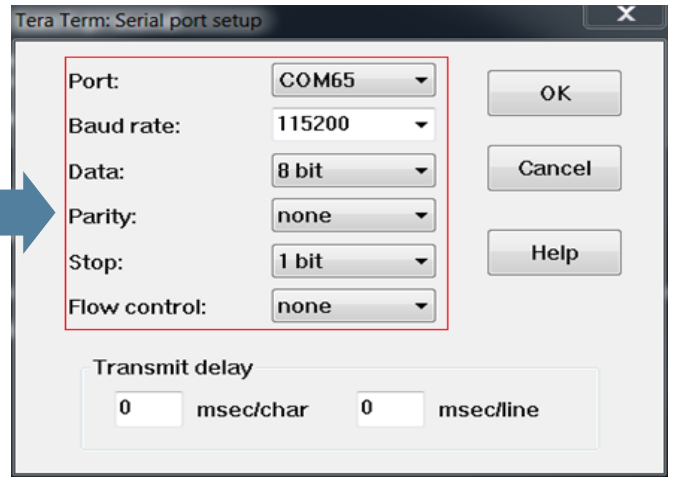
USB cable



6 Install and configure TeraTerm terminal application

TeraTerm terminal configuration





TeraTerm terminal application: <link>

A71CH getting started with i.MX6UltraLite

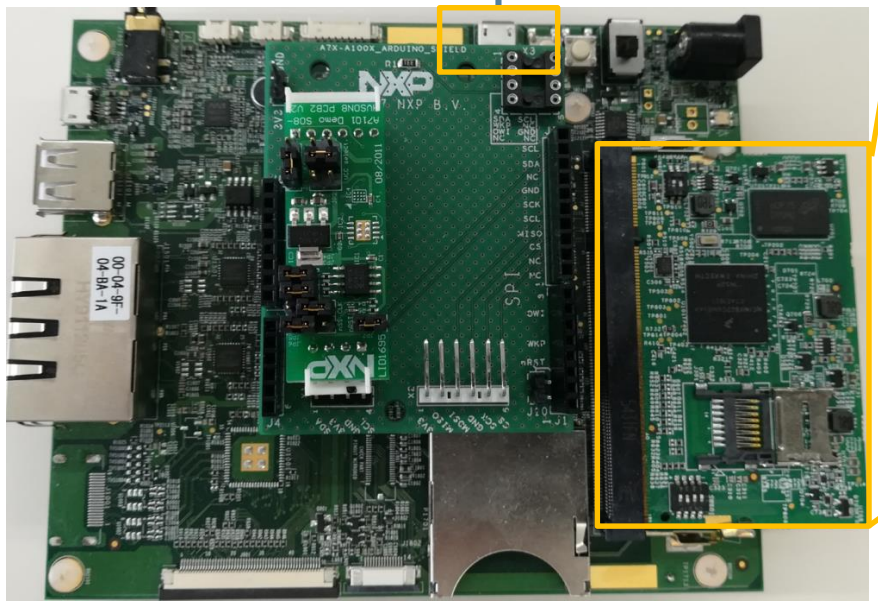
Booting the system

7 Set-up the i.MXUltraLite daughterboard switches and boot up the system

Development PC

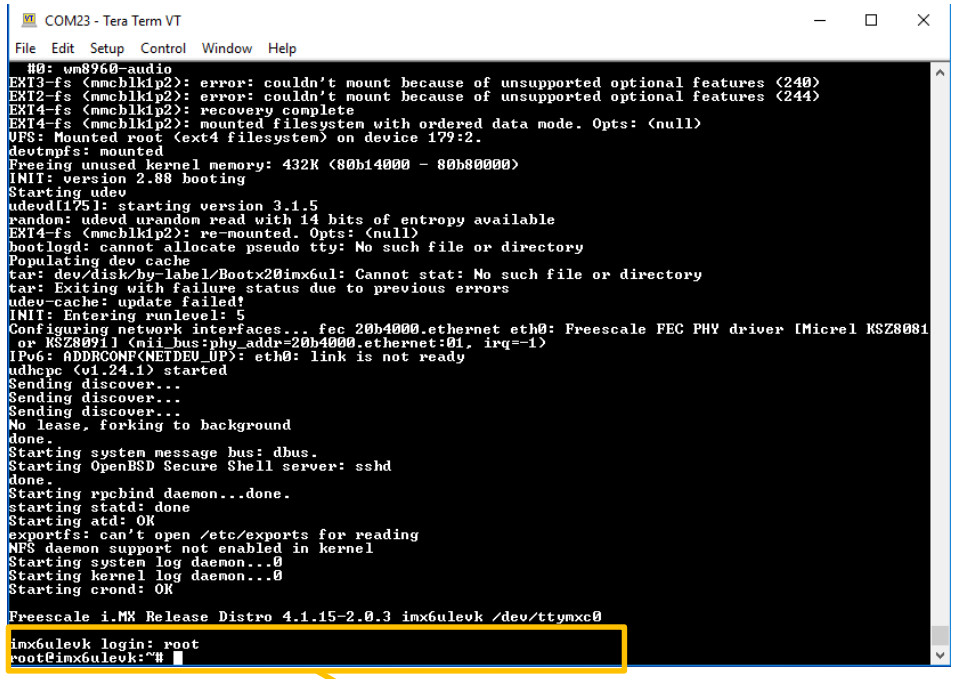
TeraTerm

USB cable



Boot Mode Select Switch SW602: ON, OFF (from 1-2 switch)

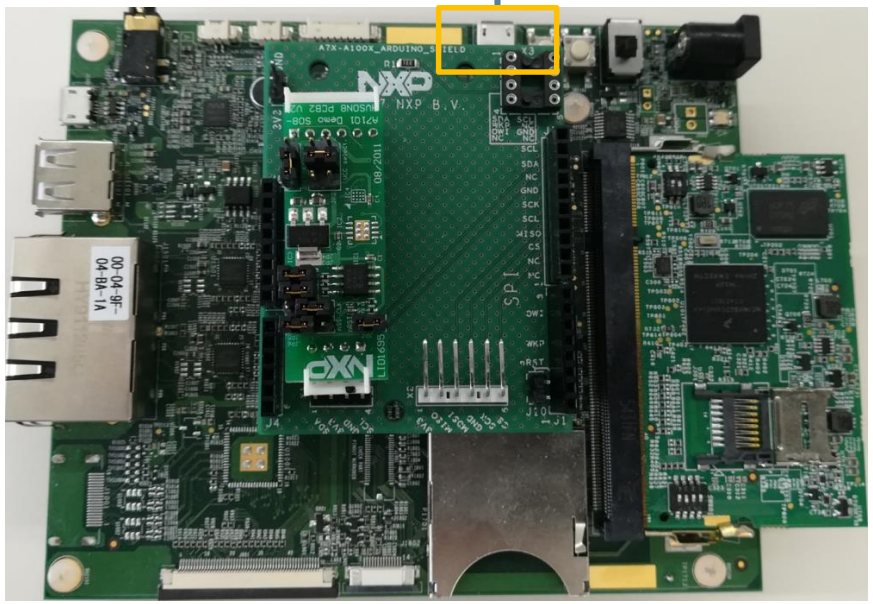
Boot Device Select Switch SW601: OFF, OFF, ON, OFF (from 1-4 switch)



- **Account name**: root
- **Password:** <set your own password>

A71CH getting started with i.MX6UltraLite

Running the applications



8

Running the A71CH Configuration tool:

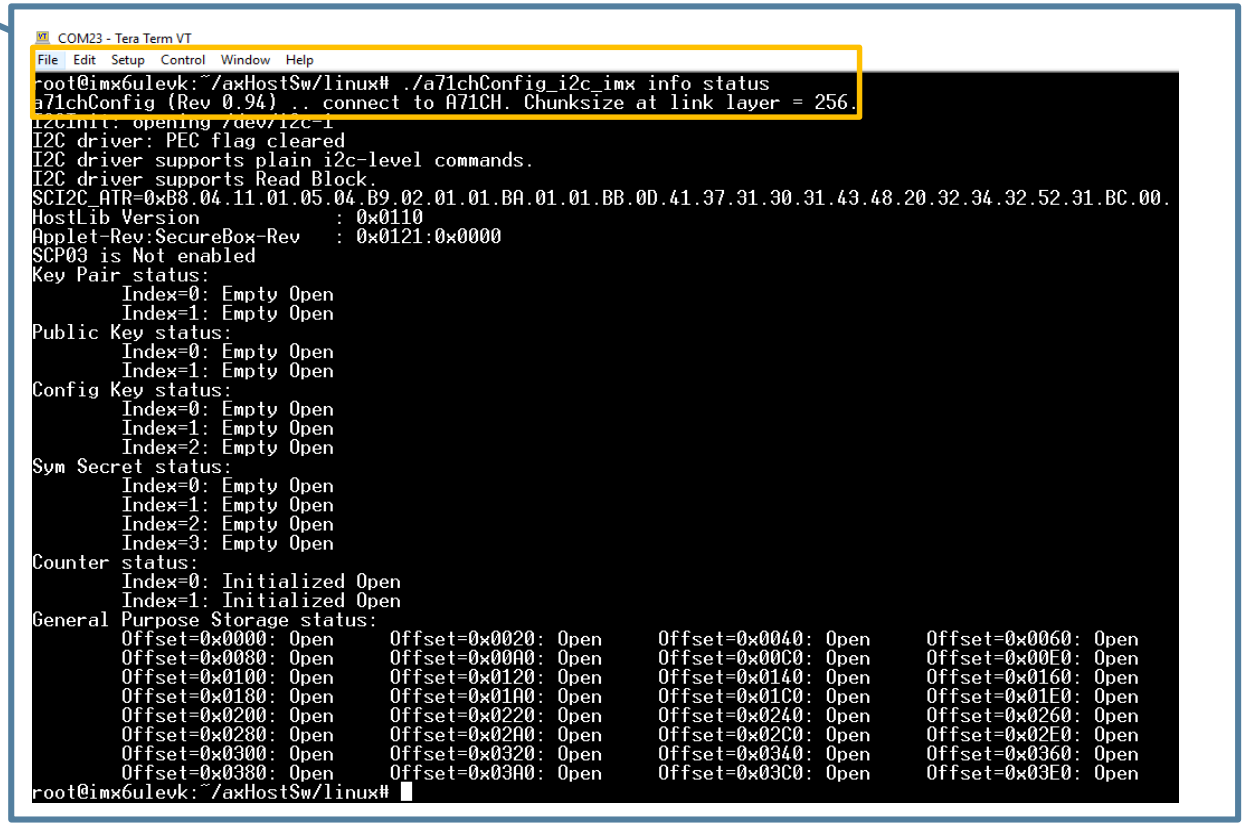
```
root@imx6ulevk:~/axHostSw/linux#
./a71chConfig_i2c_imx info status
```

Running the Host API usage examples:

```
root@imx6ulevk:~/axHostSw/linux#
./A71CH_i2c_imx
```

Running the A71CH OpenSSL Engine examples:

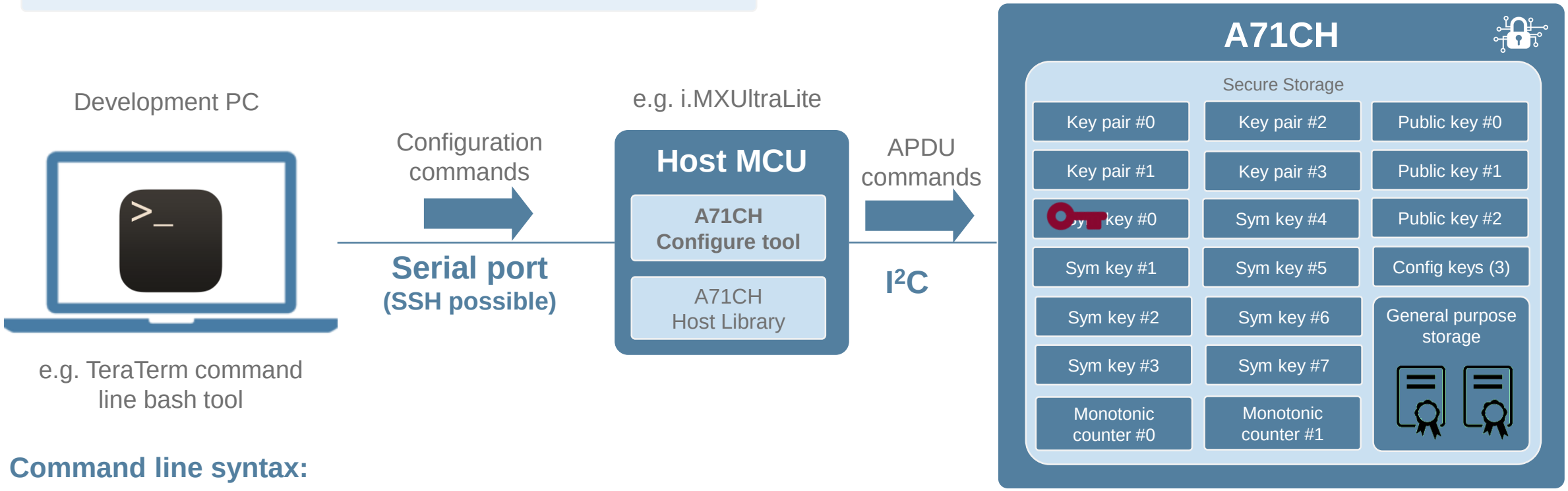
```
root@imx6ulevk:~/axHostSw/hostLib/e
mbSeEngine/a71chDemo/scripts#
./a71chPrepareEcc.sh
```



Using the A71CH Configure tool

A71CH Configure tool

The A71CH Configure tool is a command line tool that supports the insertion of credentials into the A71CH.



e.g. A71CH mini PCB

Command line syntax:

The command line syntax uses: a mandatory command name <cmd-n>, followed by an optional command qualifier <cmd-q>, followed by '0 to n' (option, value) pairs.

```
> <cmd-n> [<cmd-q>] [-option <option-value>]*
```


Training Mobile Knowledge

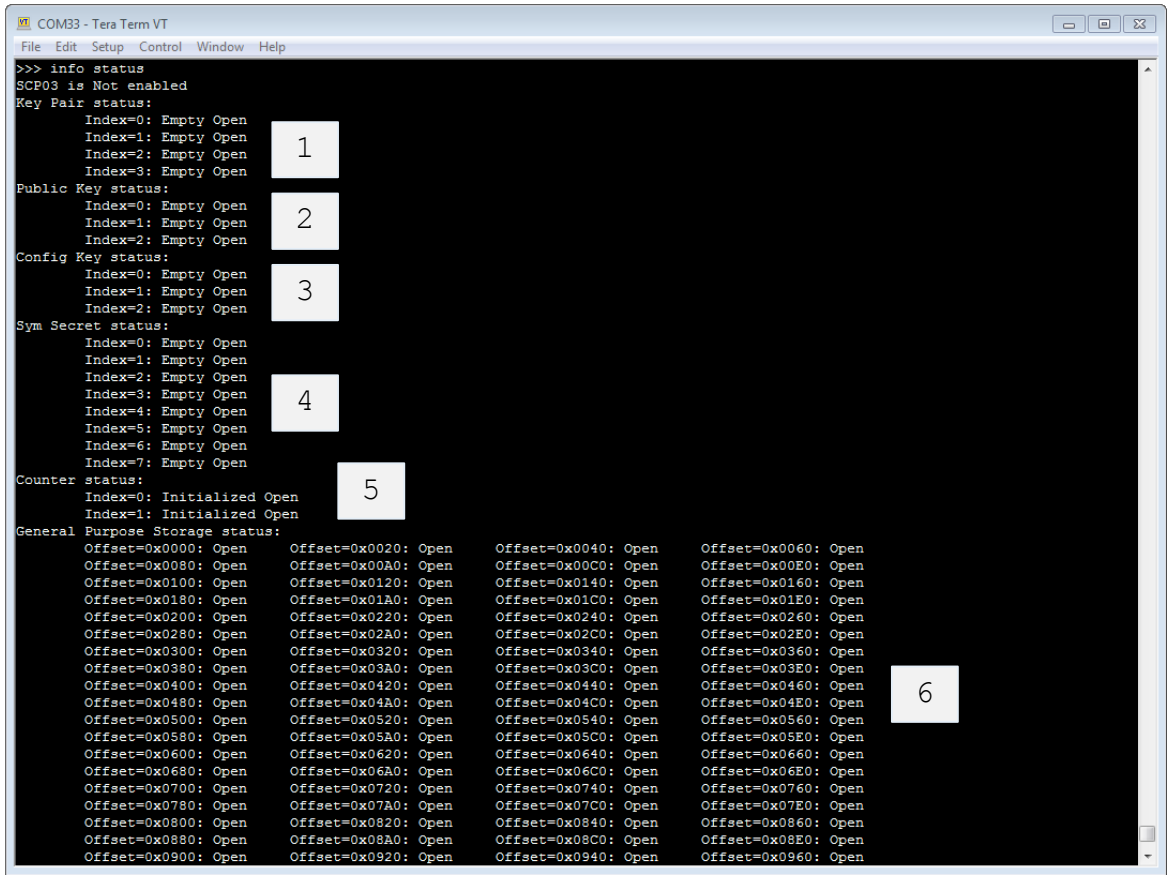
A71CH Configure tool: Info command

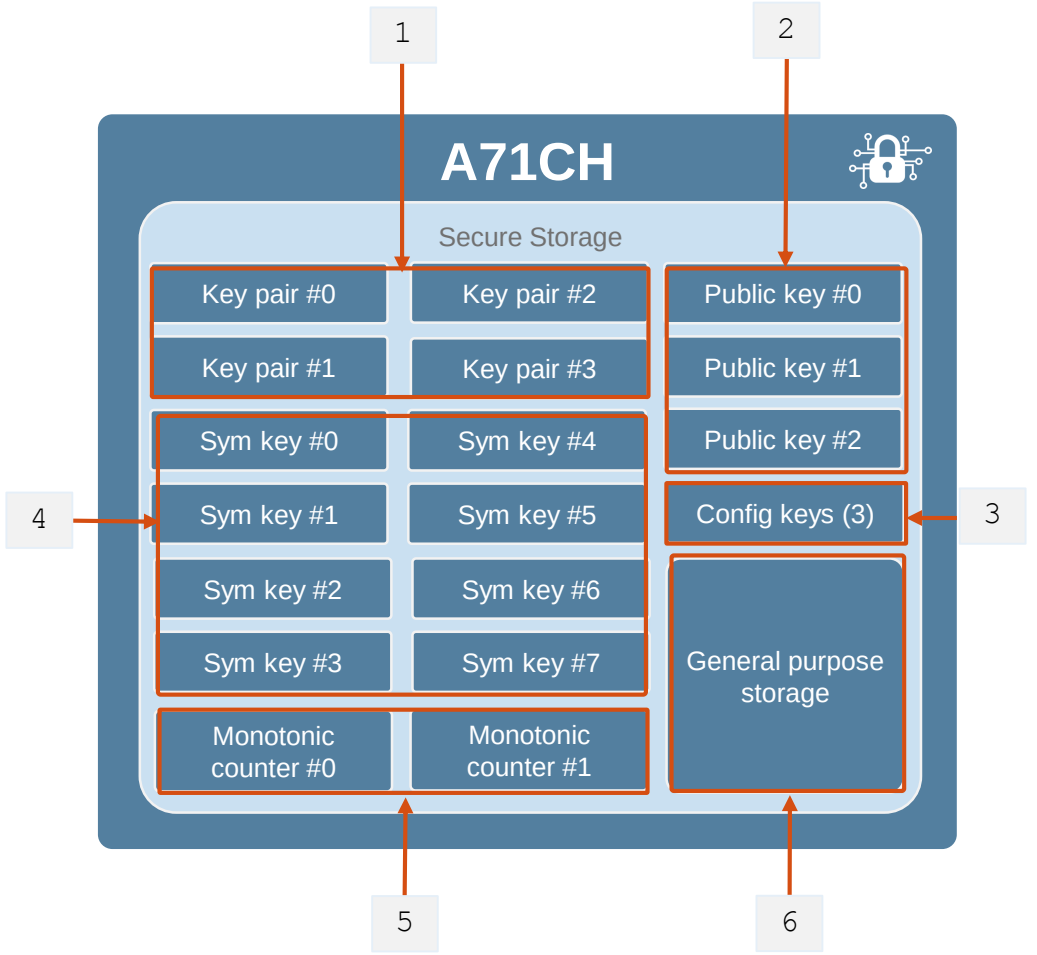
Info command

```
> info [all|device|cnt|pair|pub|sym|status]
```

Echo the status of the A71CH or its stored credentials to the console

```
> info status
```





A71CH Configure tool: Generate ECC key pair

Generate ECC key pair command:

```
> gen pair -x <int>
```

Generate ECC key pair in index #0:

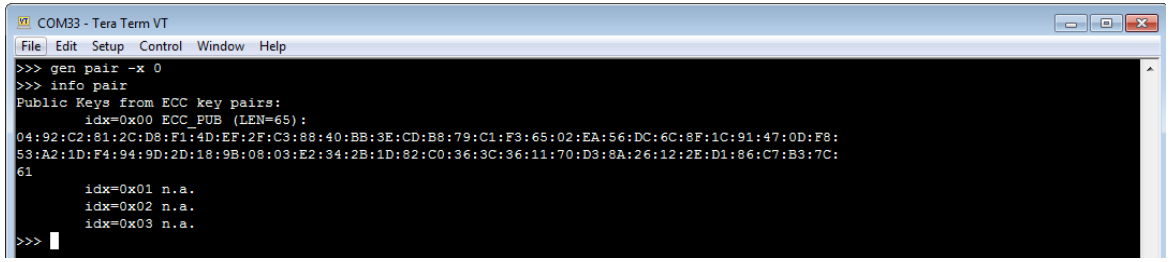
```
> gen pair -x 0
```

1

Show public key pair generated in index #0:

```
> info pair
```

2

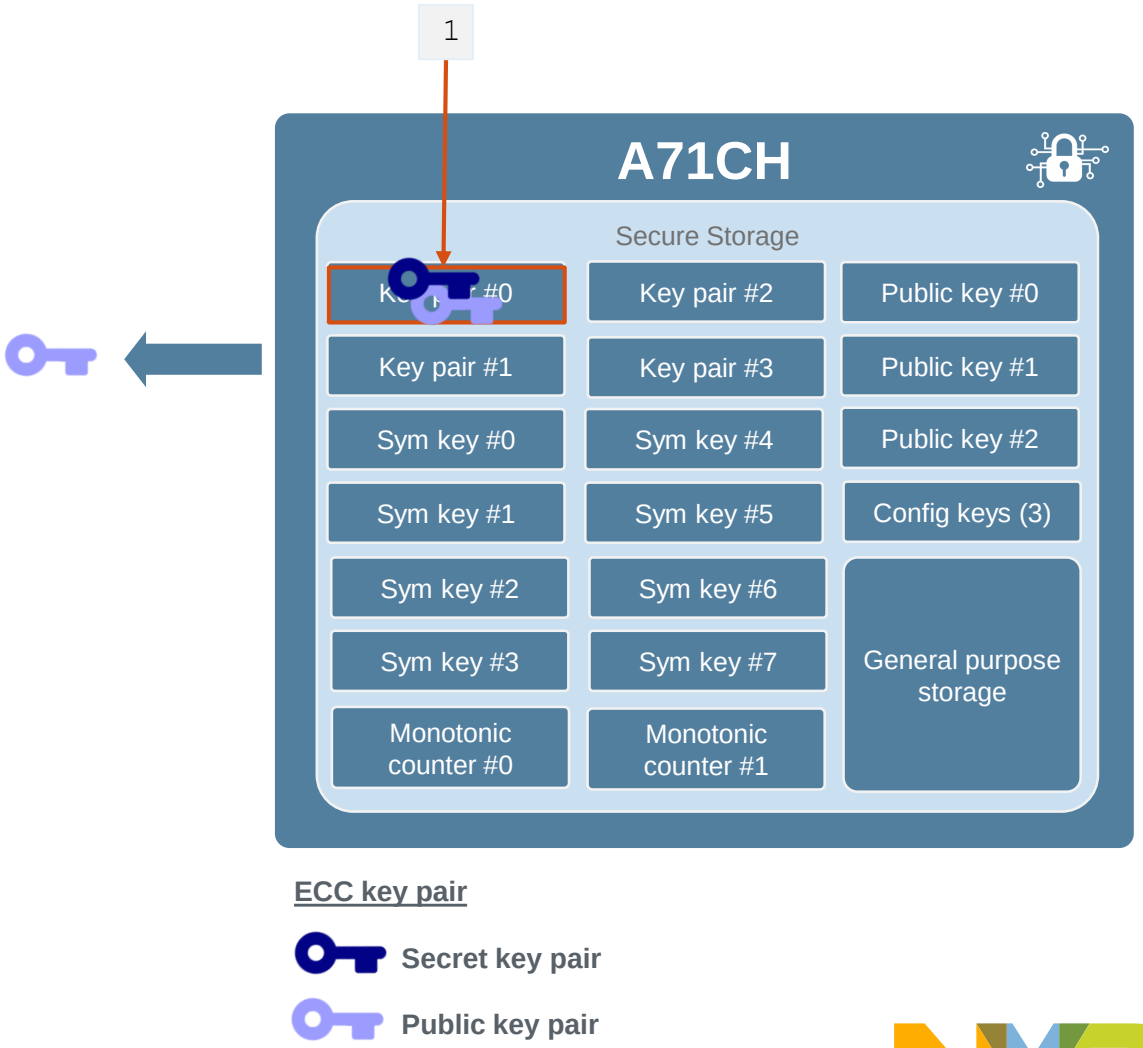


Retrieve public key command:

```
> get pub -c <hex_value> -x <int> -k <keyfile.pem>
```

Retrieve public key pair from index #0 and store it in myPublicKey.pem file

```
> get pub -c 10 -x 0 -k myPublicKey.pem
```



Training Mobile Knowledge

A71CH Configure tool: Inject an ECC key pair

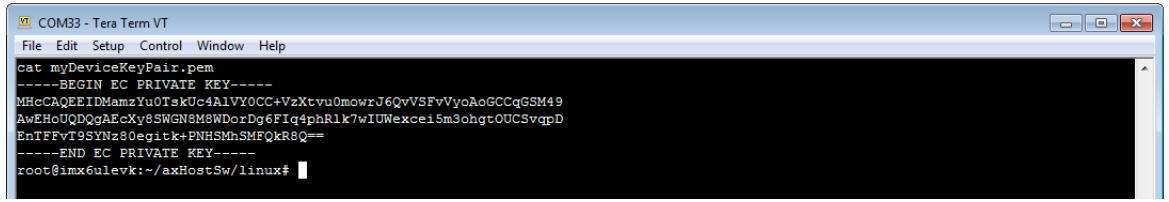
Generate a device key pair (e.g., using OpenSSL):

Generate ECC parameters:

```
> openssl ecparam -name prime256v1 -out eccparams
```

Generate myDeviceKeyPair keys:

```
> openssl ecparam -in eccparams -genkey -noout -out
myDeviceKeyPair.pem
```



1

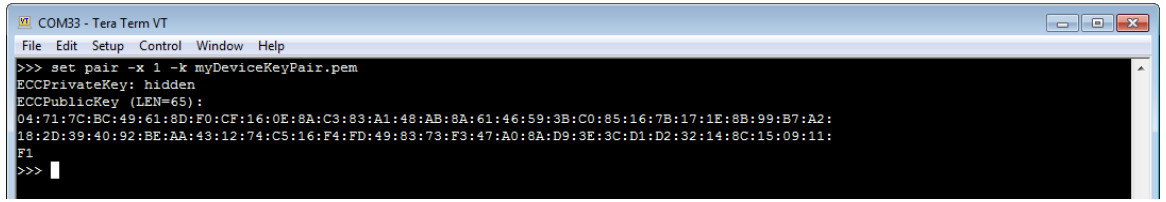
Inject key pair command:

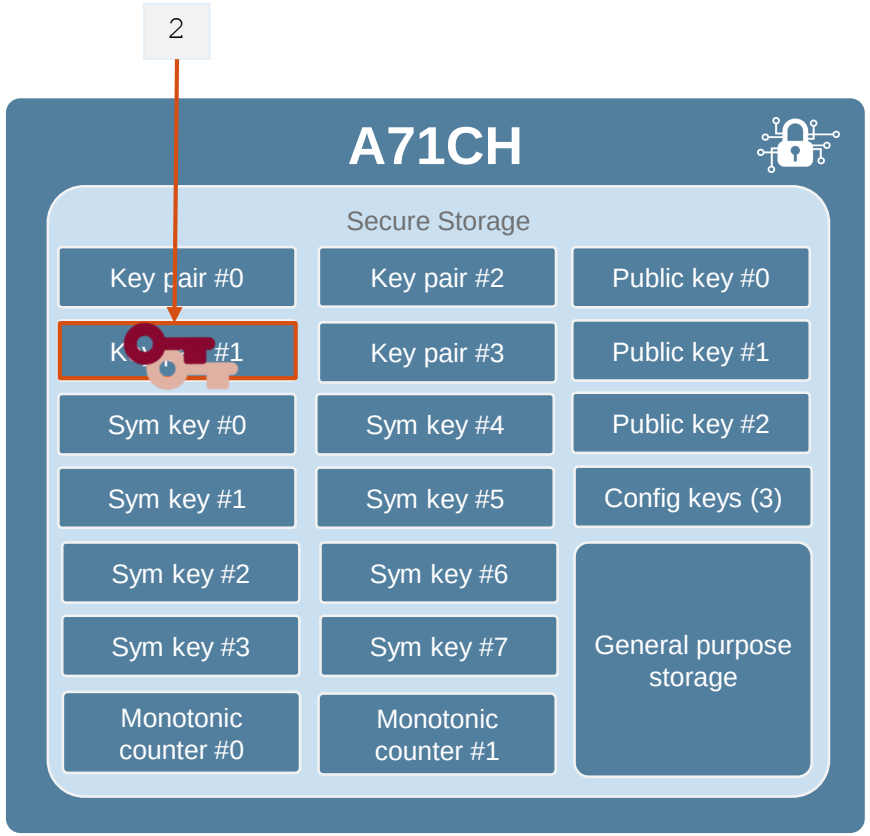
```
> set pair -x <int> [-k <keyfile.pem> | -h
<hexvalue_pub> -h <hexvalue_priv>]
```

Inject myDeviceKeyPair in slot **#1**:

```
> set pair -x 1 -k myDeviceKeyPair.pem
```

2





ECC key pair

Secret key pair

Public key pair

A71CH Configure tool: Generate a certificate with OpenSSL

Generate a device certificate pair (e.g., using OpenSSL):

Generate self-signed Root CA cert

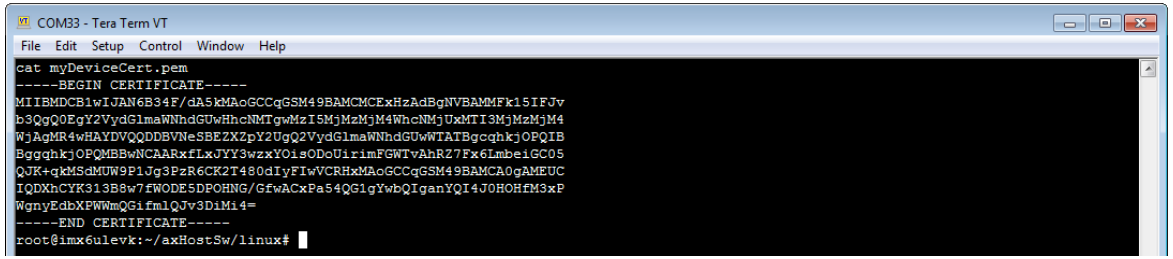
```
> openssl req -x509 -new -nodes -key
CArootecckeys.pem -subj "/CN=My Root CA certificate"
-days 2800 -out CArootCert.cer
```

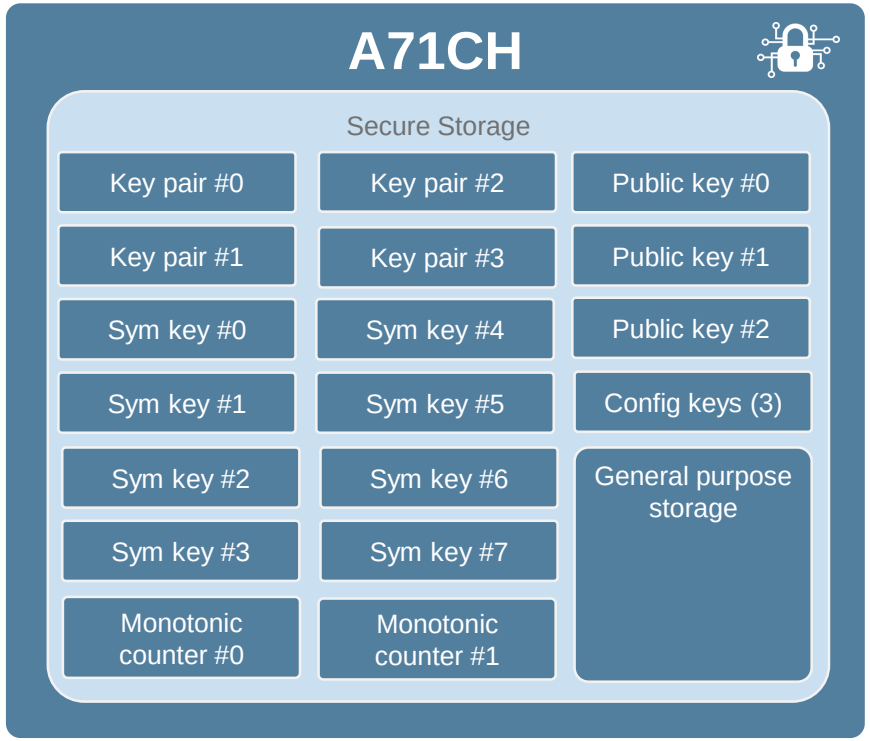
Generate MyDeviceCSR.csr certificate signing request:

```
> openssl req -new -key myDeviceKeyPair.pem -subj
"/CN=My Device Certificate" -out MyDeviceCSR.csr
```

Generate myDeviceCert signed device certificate

```
> openssl x509 -req -sha256 -days 2800 -in
MyDeviceCSR.csr -CAcreateserial -CA CArootCert.cer -
CAkey CArootecckeys.pem -out myDeviceCert.pem
```





Training Mobile Knowledge

A71CH Configure tool: Inject a certificate in A71CH GP area

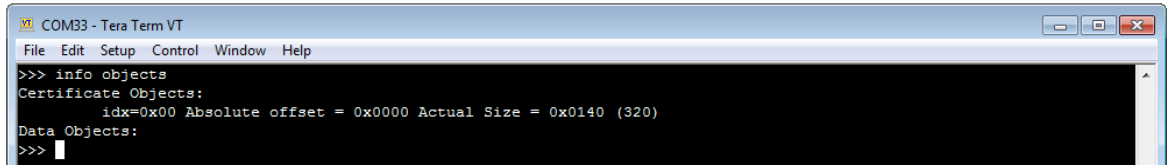
Inject certificate into GP storage area:

Write myDeviceCert certificate in GP storage starting in index #0:

```
> wcrt -x 0 -p myDeviceCert.pem
```

Retrieve objects stored in the GP area (including certificates and unstructured data)

```
> info objects
```

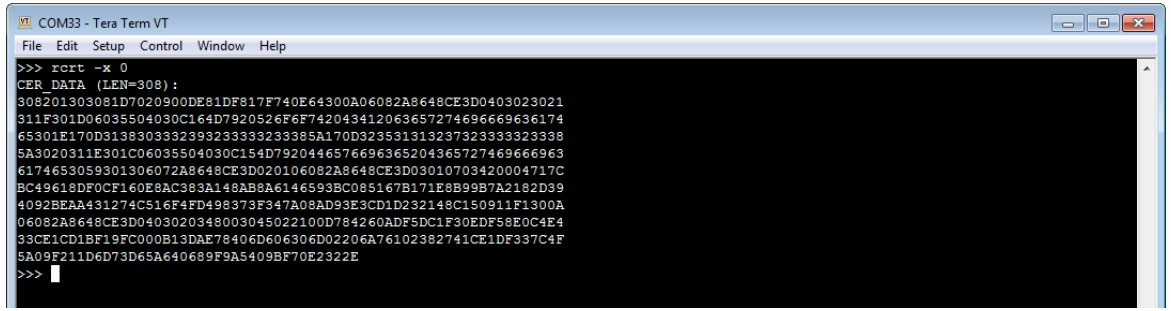


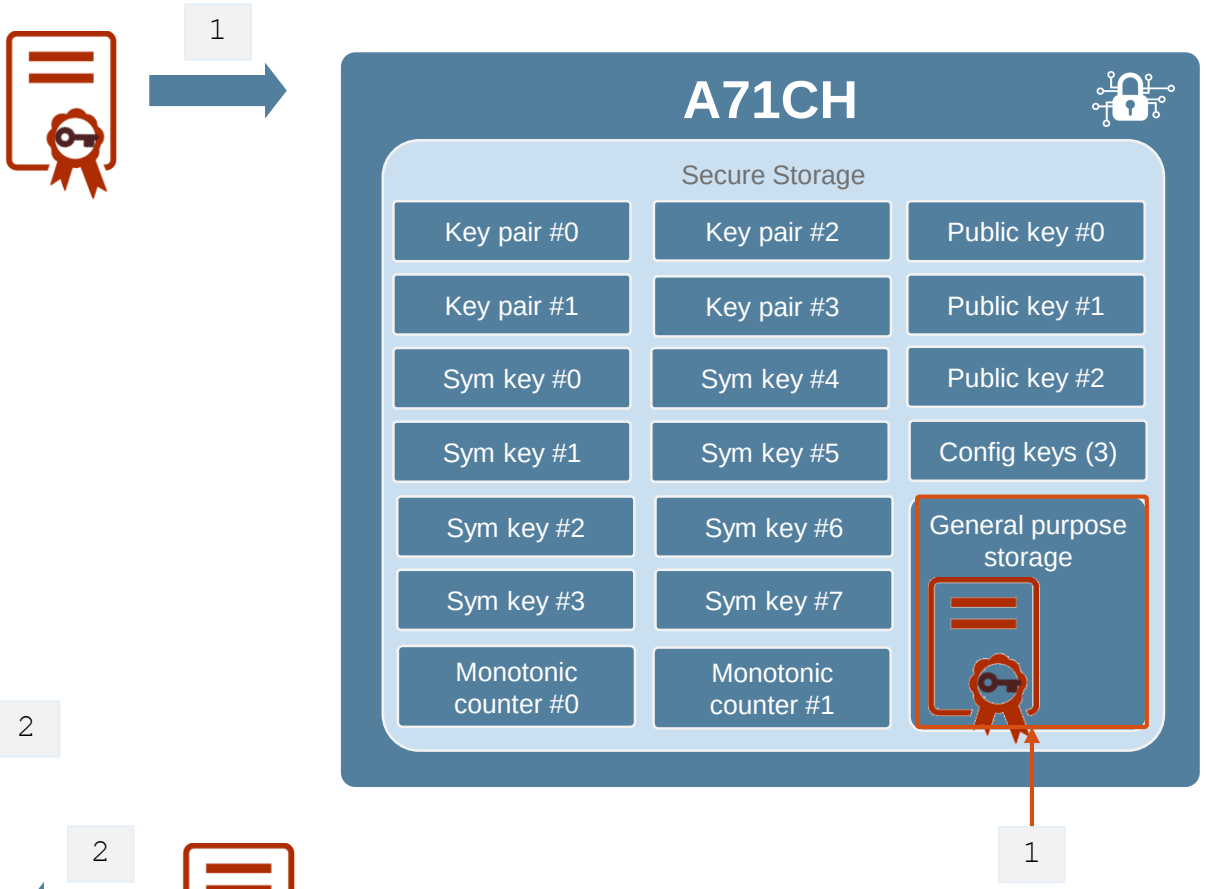
Read certificate from GP storage area:

Read certificate from GP storage area in slot #0

```
> rcrt -x 0
```

2





A71CH Configure tool: Inject symmetric keys

Inject symmetric key command:

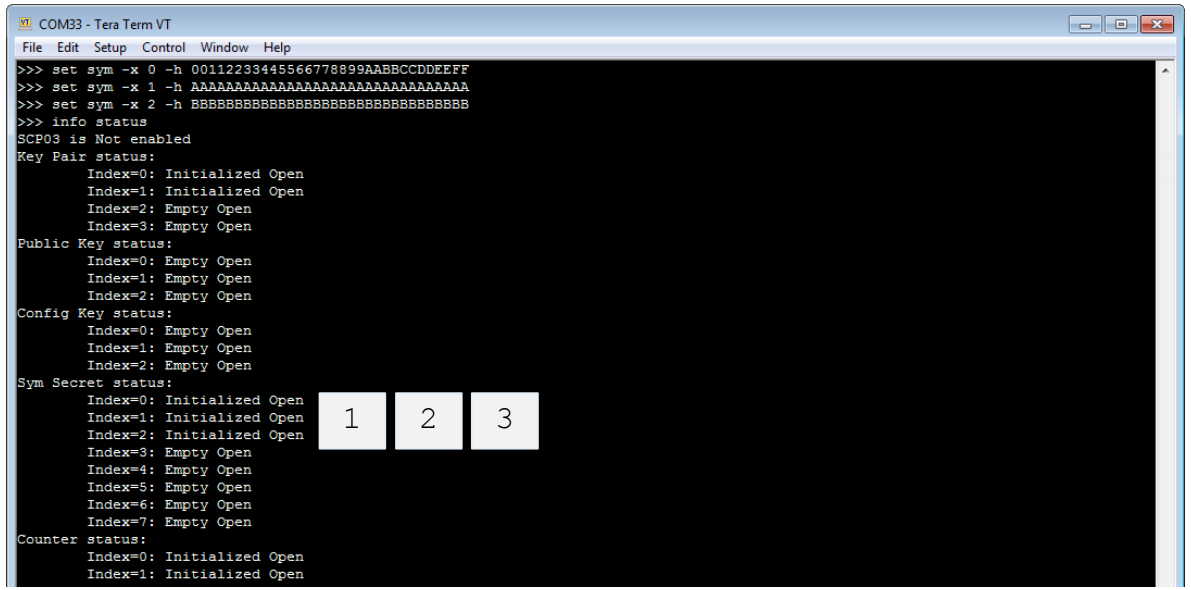
```
> set [cfg|cnt|sym]  -x <int> -h <hexvalue>
```

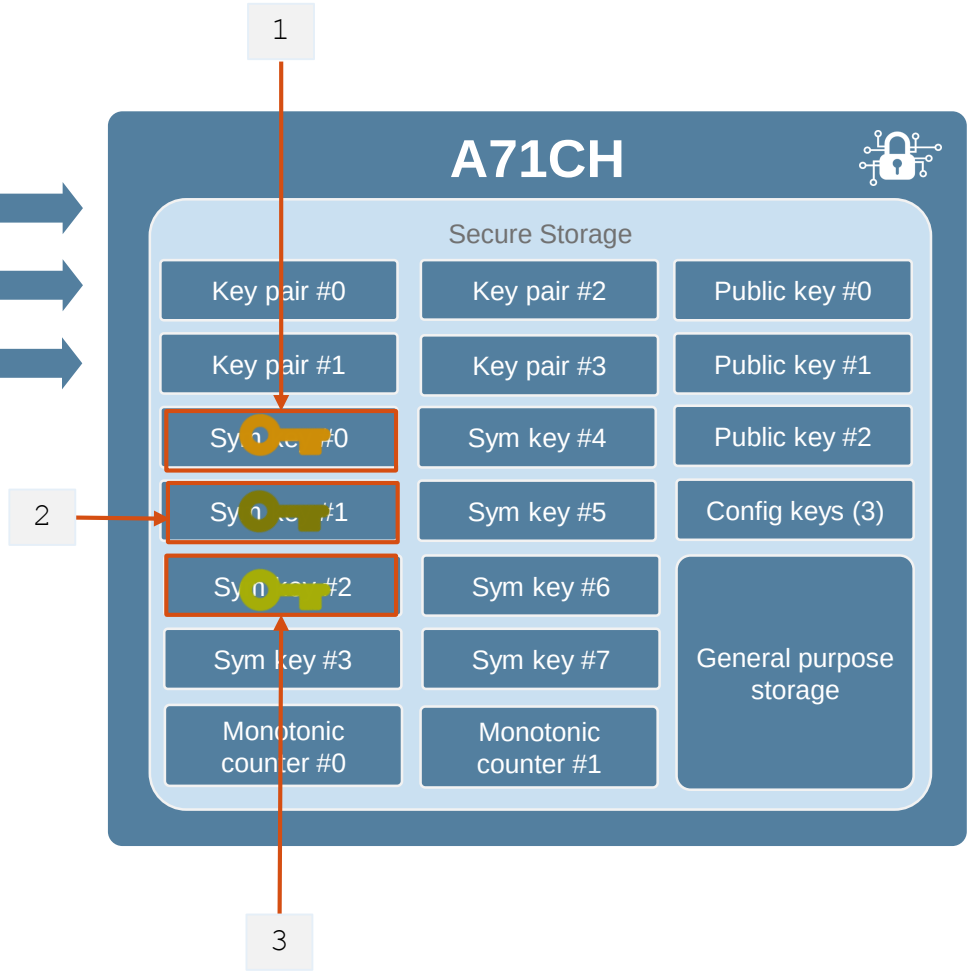
Inject three symmetric keys in index #0, #1 and #2:

```
> set sym -x 0 -h 00112233445566778899AABBCCDDEEFF
```
1

```
> set sym -x 1 -h AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
```
2

```
> set sym -x 2 -h BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
```
3





Training Mobile Knowledge

A71CH Configure tool: Increase a monotonic counter

Set or increase a monotonic counter value command:

```
> set [cfg|cnt|sym]  -x <int> -h <hexvalue>
```

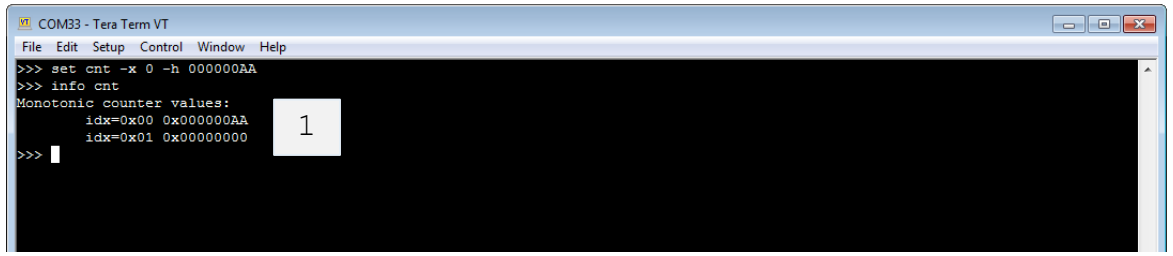
Increases a monotonic counter #0 in AAh units.

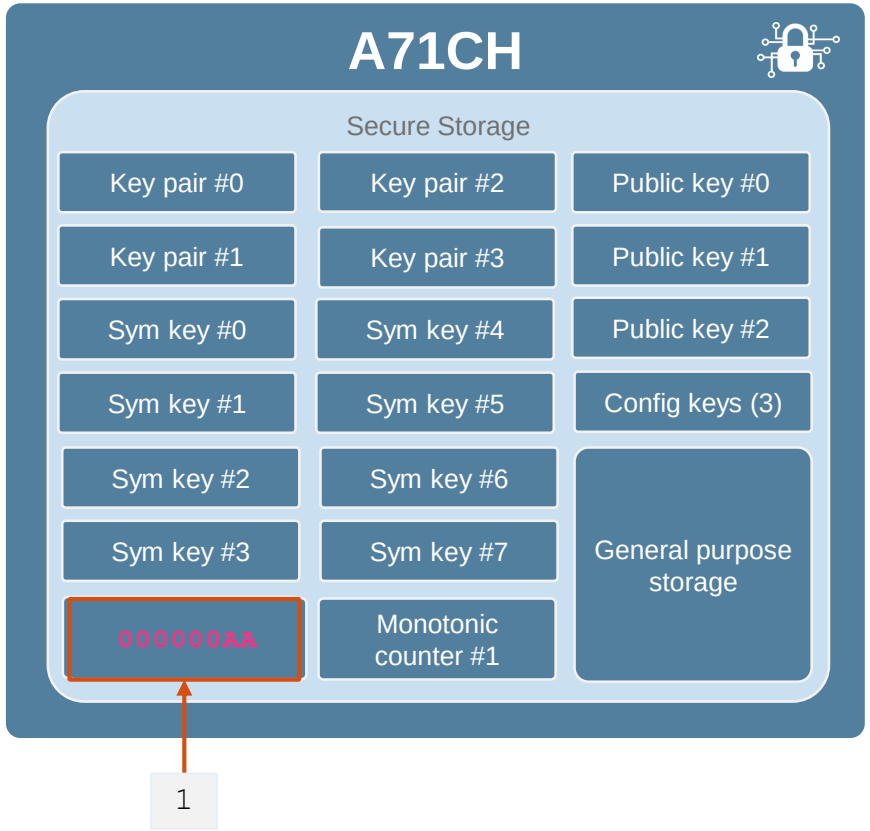
```
> set cnt -x 0 -h 000000AA
```

1

Read the monotonic counter values command:

```
info cnt
```



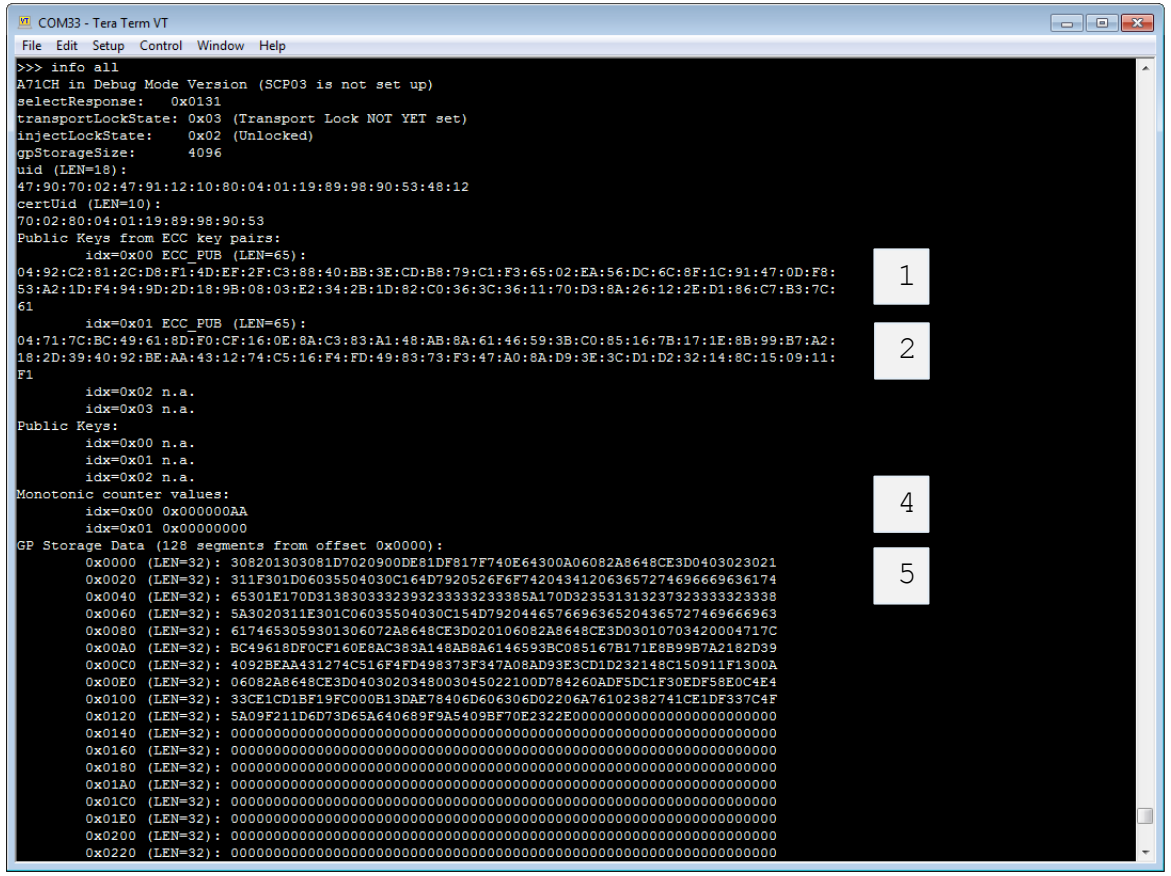


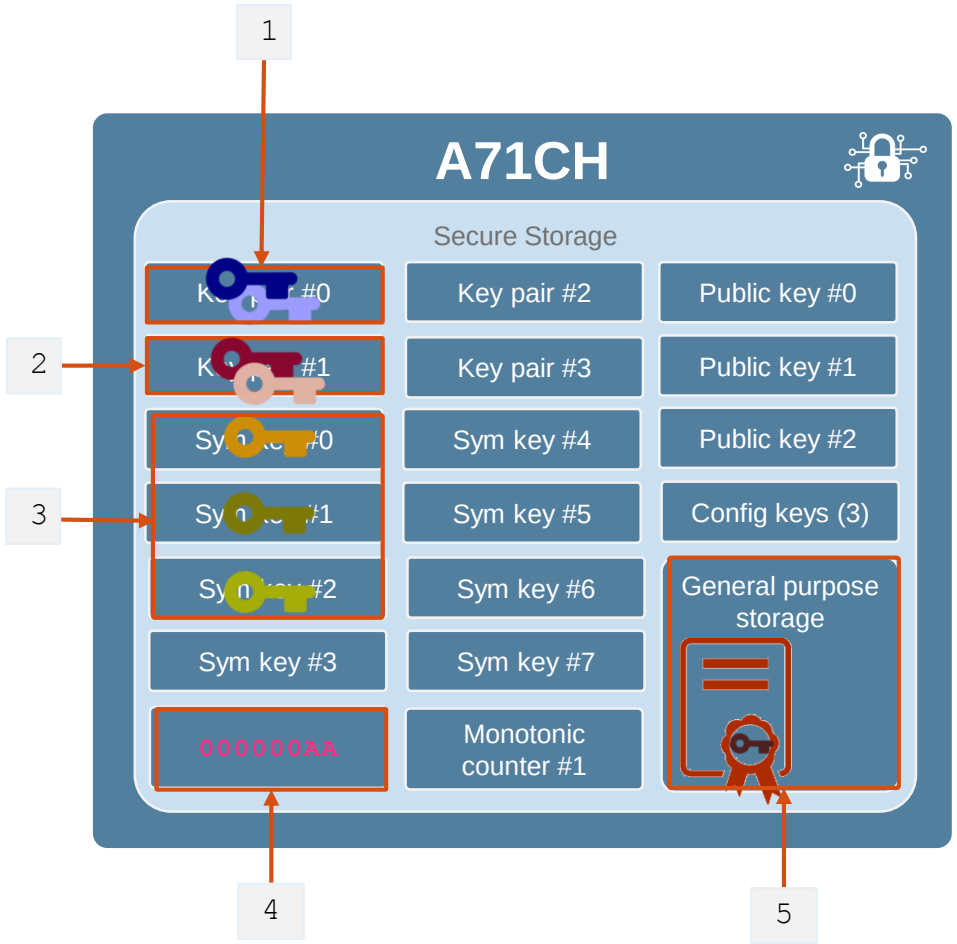
Training Mobile Knowledge

A71CH Configure tool: A71CH provisioning status

Retrieve A71CH actual provisioning values:

```
> info all
```





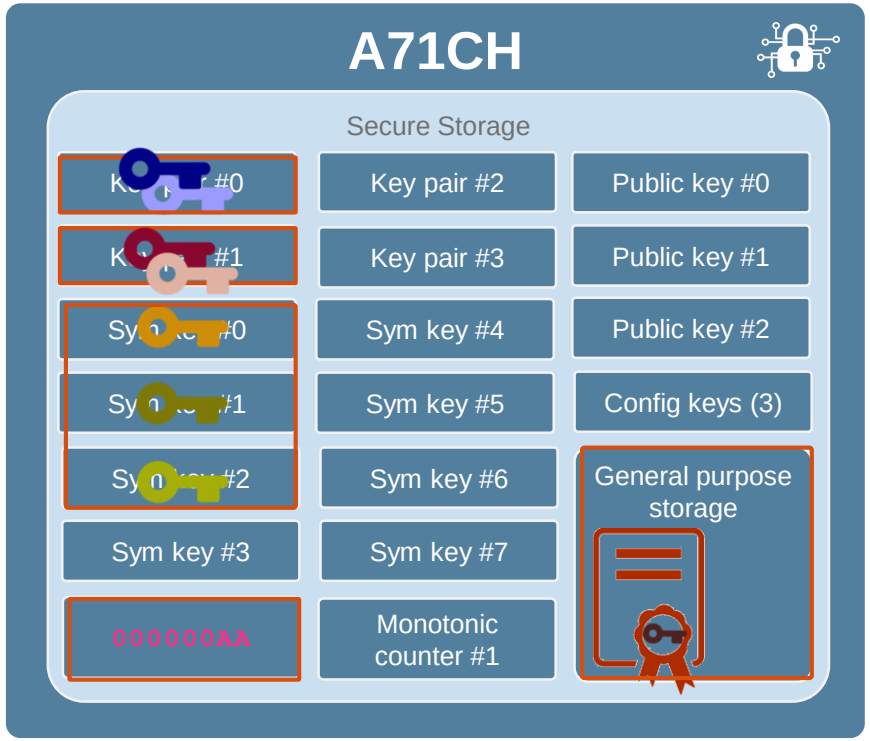
A71CH Configure tool: Other commands

List of supported commands:

COM33 - Tera Term VT

File Edit Setup Control Window Help

```
>>>
[apdu|debug|erase|gen|info|lock|scp|set|help|quit] <OptArg>
    apdu -cmd <hexvalue> -sw <hexvalue>
    connect [close|open]
    debug [permanently_disable_debug|reset]
    ecrt -x <int>
    erase [cnt|pair|pub|sym] -x <int>
    gen pair -x <int>
    get pub -c <hex_value> -x <int> -k <keyfile.pem>
    info [all|cnt|device|objects|pair|pub|status]
    info gp -h <hexvalue_offset> -n <segments>
    lock [pair|pub|sym] -x <int>
    lock gp -h <hexvalue_offset> -n <segments>
    lock inject_plain
    obj erase -x <int>
    obj get -x <int> [-h <hexvalue_offset>] [-s <hexvalue_size>]  [-f <data.txt> -t [hex_16|hex_32]]
    obj update -x <int> -h <hexvalue_offset> [-f <data.txt> -t [hex_16|hex_32] | -h <hexvalue_data>]
    obj write -x <int> [-f <data.txt> -t [hex_16|hex_32] | -h <hexvalue_data> | -n <segments>]
    rcrt -x <int> [-c <certfile.crt>]
    refpem -c <hex_value> -x <int> [-k <keyfile.pem>] -r <ref_keyfile.pem>
    script -f <script.txt>
    scp [put|auth] -h <hexvalue_keyversion> -k <keyfile>
    scp clear_host
    set gp -h <hexvalue_offset> -h <hexvalue_data>
    set gp -h <hexvalue_offset> -c <certfile.pem>
    set pair -x <int> [-k <keyfile.pem> | -h <hexvalue_pub> -h <hexvalue_priv>] [-w <hexvalue_wrap_key>]
    set pub  -x <int> [-k <keyfile.pem> | -h <hexvalue>] [-w <hexvalue_wrap_key>]
    set [cfg|cnt|sym]  -x <int> -h <hexvalue> [-w <hexvalue_wrap_key>]
    transport [lock|unlock -h <hexvalue_tpkey>]
    ucrt -x <int> [-c <certfile.crt> | -h <hexvalue_data> | -p <certfile.pem>]
    wcrt -x <int> [-c <certfile.crt> | -h <hexvalue_data> | -p <certfile.pem>] [-n <padding-segments>]
>>>
```



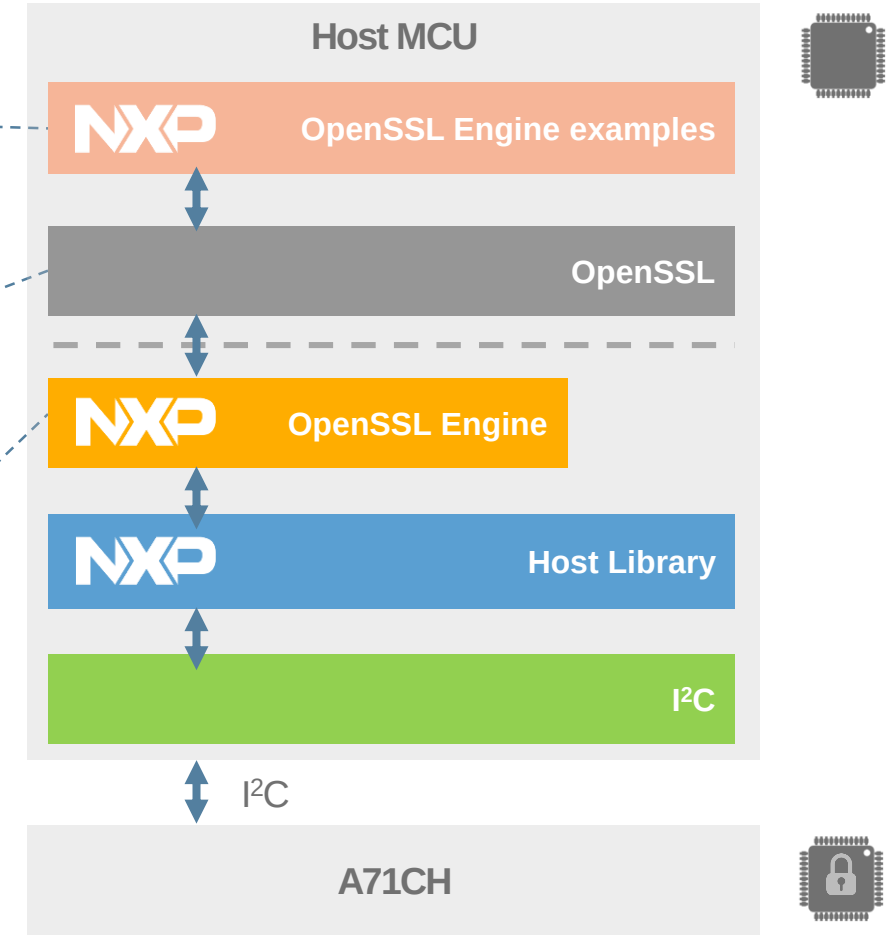
Using A71CH
OpenSSL Engine examples

Software architecture for the OpenSSL Engine examples

A set of examples that illustrate how to use the standard OpenSSL tools in combination with the OpenSSL Engine for A71CH.

OpenSSL is a software library in C language that contains an open-source implementation of the SSL and TLS protocols and also implements basic cryptographic functions and utility functions.

OpenSSL exposes an Engine API, which allows us to plug in alternative implementations of some or all of the cryptographic operations implemented by OpenSSL. **NXP OpenSSL Engines provides a hardware implementation of specific cryptographic operations through A1CH security IC.**



Note: A71CH Host software package **v1.4.0** release includes OpenSSL and mbedTLS support

Software architecture for the OpenSSL Engine examples

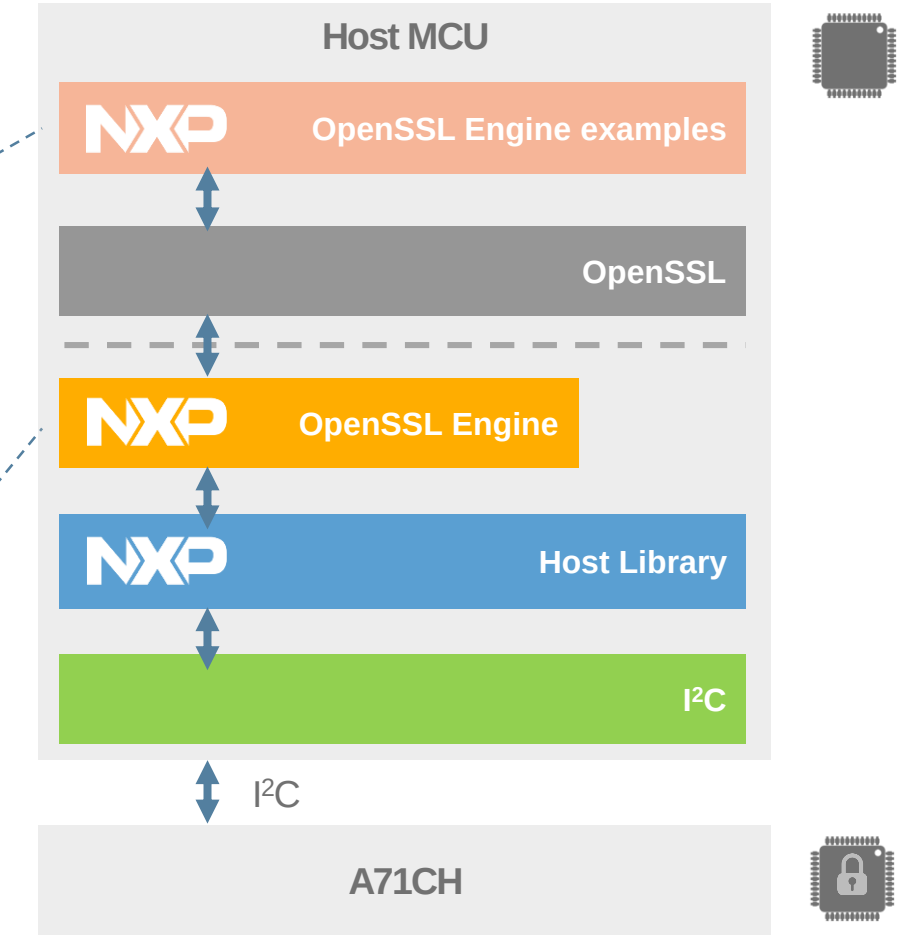
List of OpenSSL Engine examples:

```
./a71chPrepareEcc.sh
./a71chRandDemo.sh
./a71chEccCsrDemo.sh
./a71chEcDhKa.sh
./a71chEccSignDemo.sh
./tlsCreateCredentialsRunOnClientOnce.sh
./tlsPrepareClient.sh
./tlsServer.sh
./tlsSeClient.sh
./a71chTlsClient.sh
```

List of OpenSSL Engine supported functions:

The NXP OpenSSL Engine provides a hardware implementation of specific cryptographic operations through A1CH security IC.:

- Random number generation
- ECC sign
- ECC verify
- ECDH compute_key



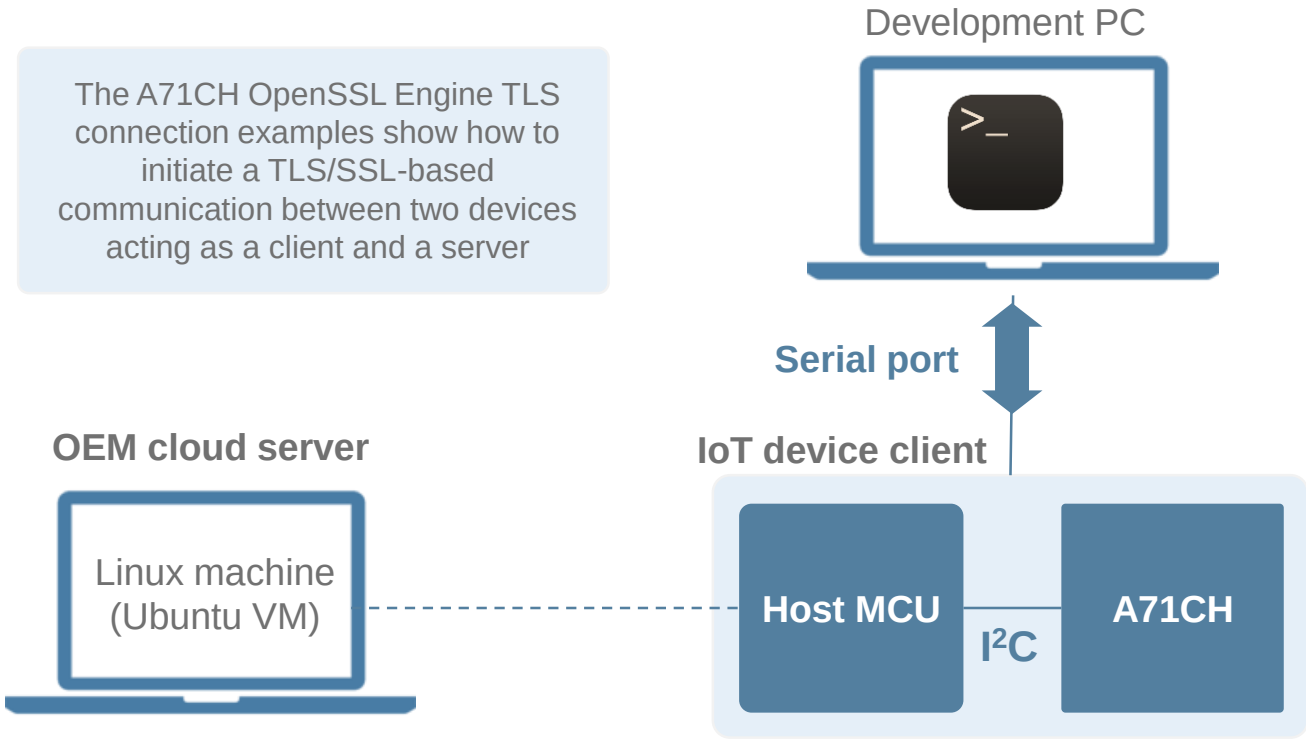
Training Mobile Knowledge

A71CH OpenSSL Engine TLS communication examples

OpenSSL Engine TLS scripts order

- `tlsCreateCredentialsRunOnClientOnce.sh`
- `tlsPrepareClient.sh`
- `tlsServer.sh`
- `tlsSeClient.sh`

The A71CH OpenSSL Engine TLS connection examples show how to initiate a TLS/SSL-based communication between two devices acting as a client and a server



Note: The A71CH OpenSSL Engine example scripts are meant purely to demonstration purposes and needs to be adjusted and adapted for commercial deployment.

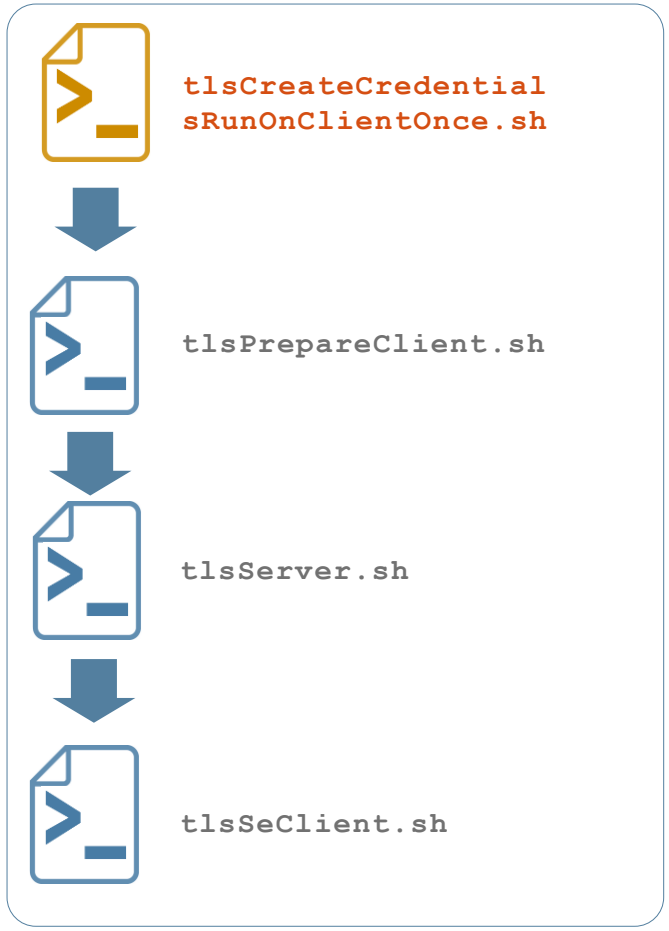
NEW

A71CH Host software package **v1.4.0** release **also** includes a TLS client in source code (***a71chTlsClient.c***).

Training Mobile Knowledge

A71CH OpenSSL Engine TLS communication examples

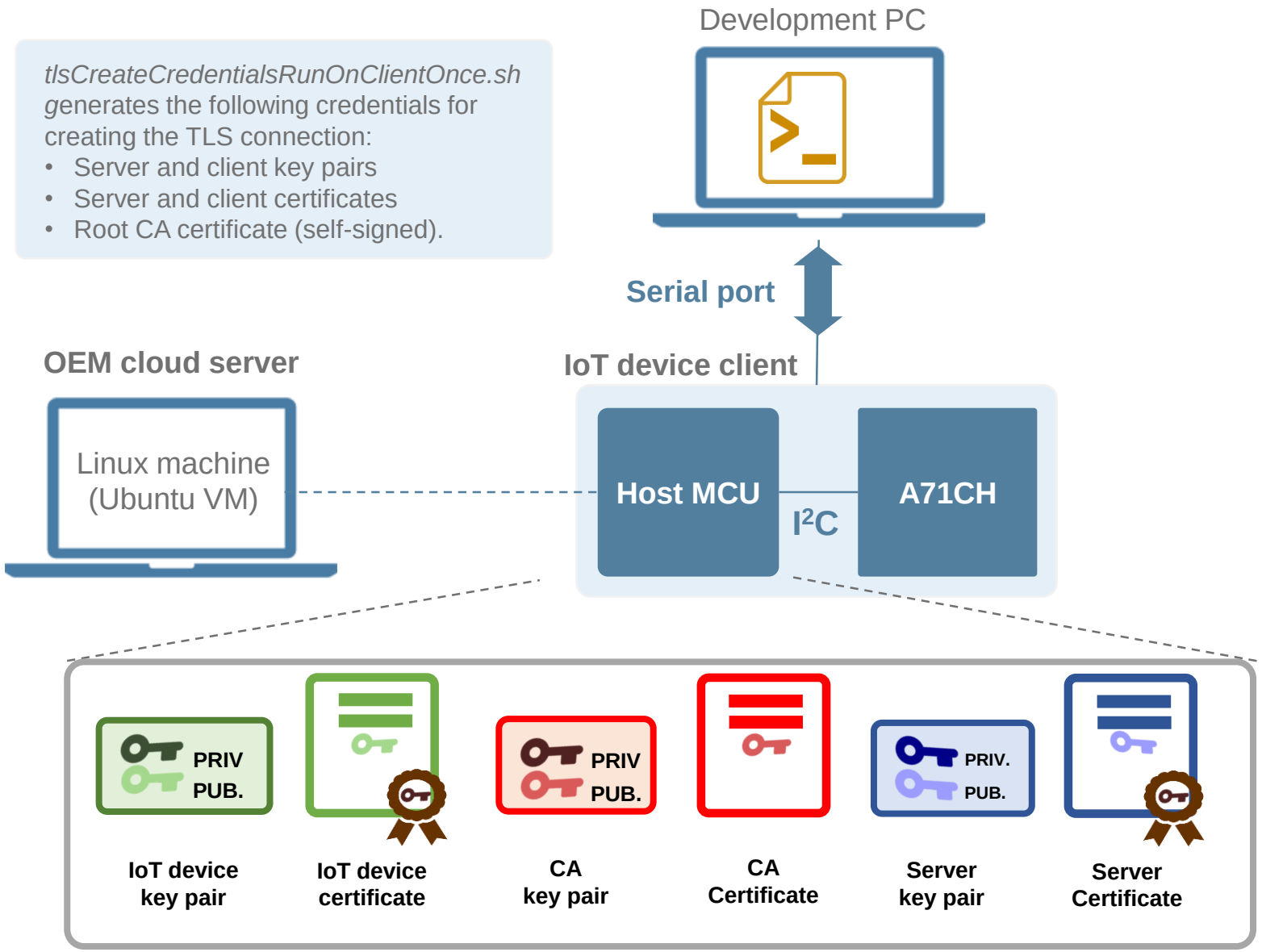
OpenSSL Engine TLS scripts order



Signature generated by the CA

tlsCreateCredentialsRunOnClientOnce.sh generates the following credentials for creating the TLS connection:

- Server and client key pairs
- Server and client certificates
- Root CA certificate (self-signed).



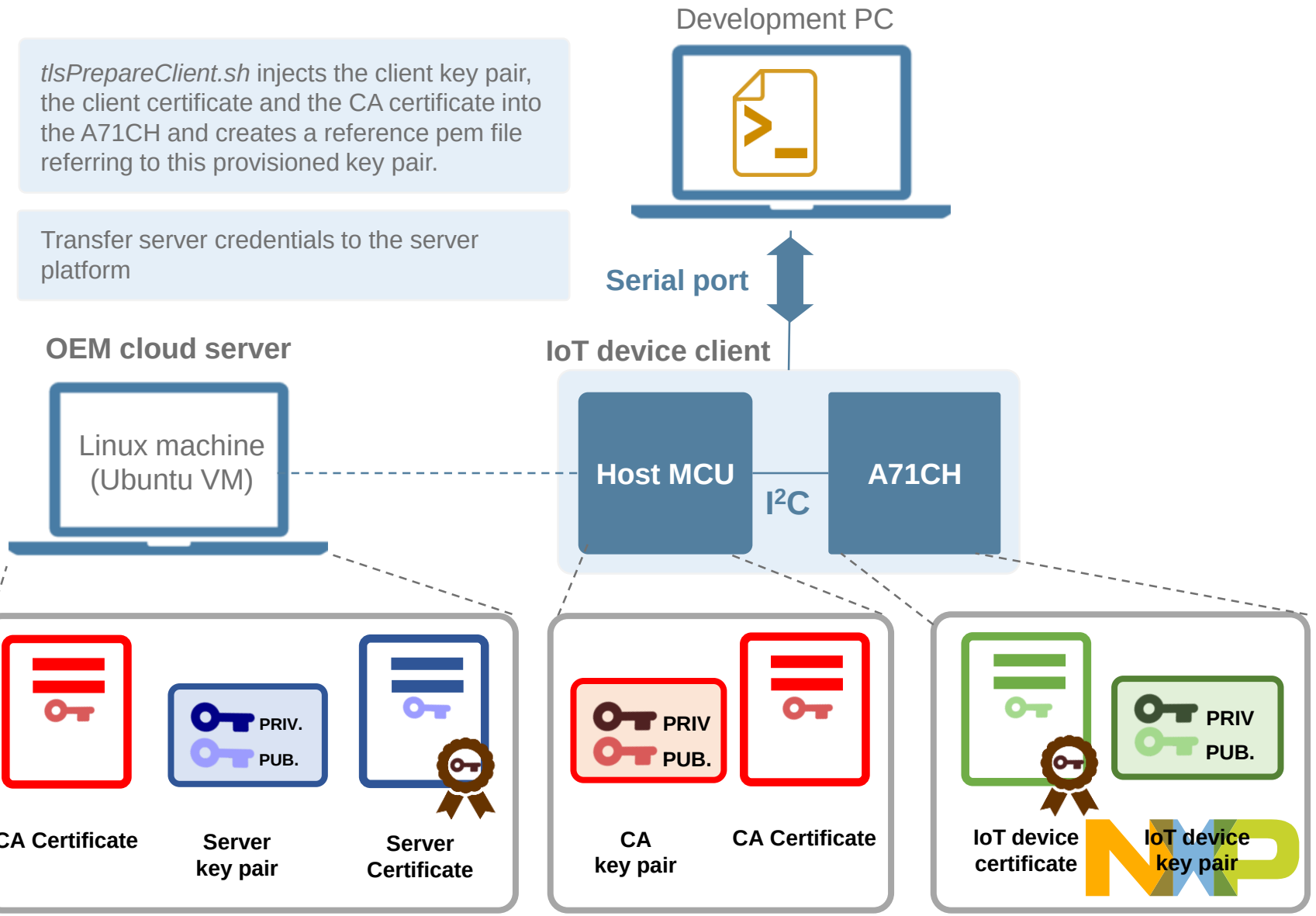
A71CH OpenSSL Engine TLS communication examples

OpenSSL Engine TLS scripts order

- `tlsCreateCredentialsRunOnClientOnce.sh`
- `tlsPrepareClient.sh`
- `tlsServer.sh`
- `tlsSeClient.sh`

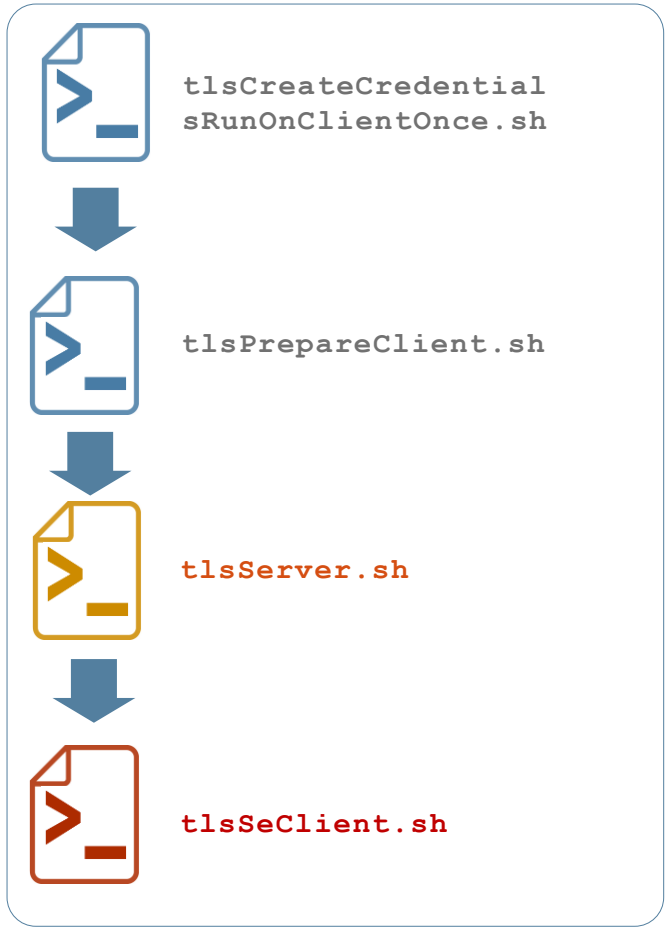
tlsPrepareClient.sh injects the client key pair, the client certificate and the CA certificate into the A71CH and creates a reference pem file referring to this provisioned key pair.

Transfer server credentials to the server platform



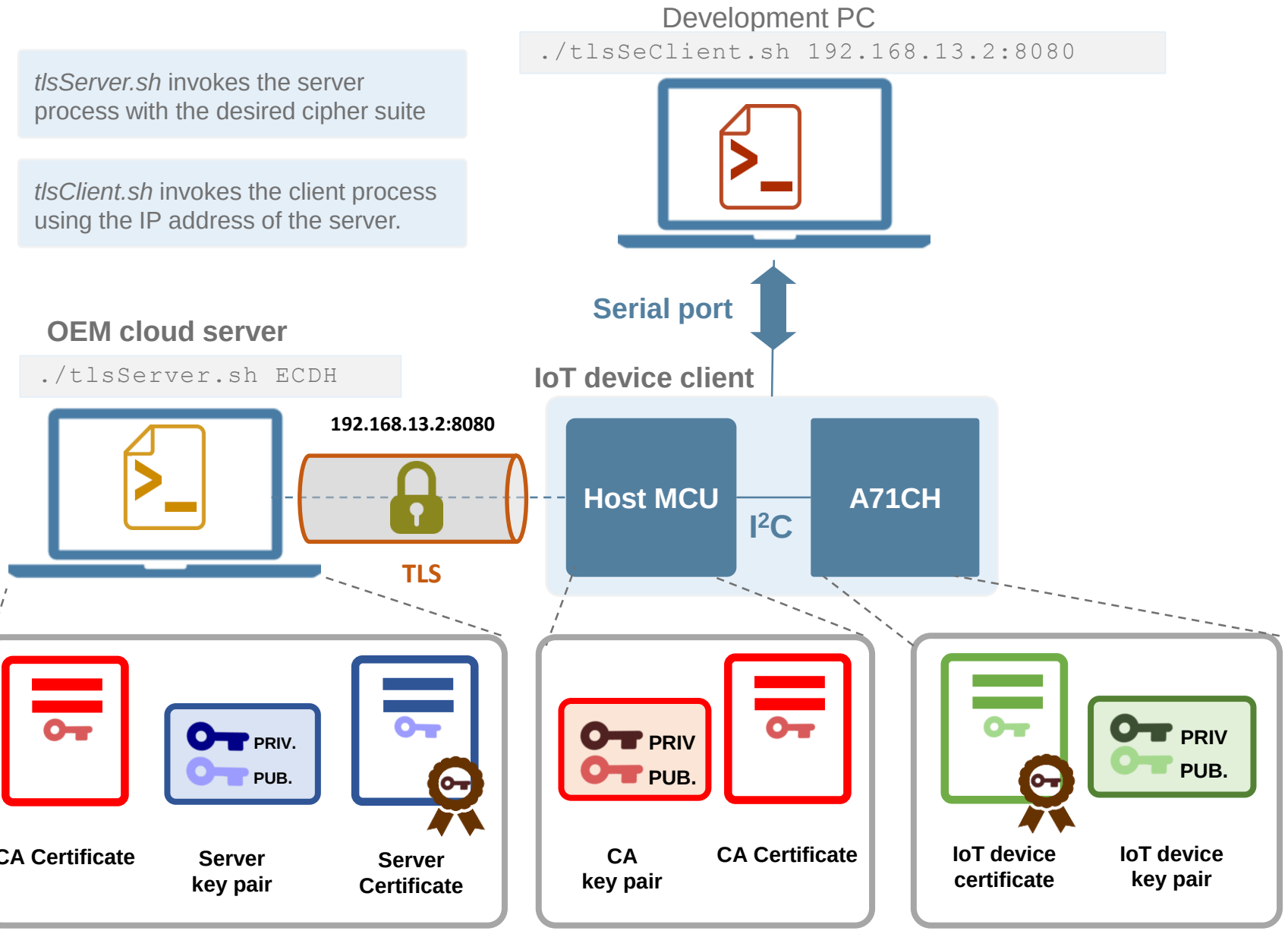
A71CH OpenSSL Engine TLS communication examples

OpenSSL Engine TLS scripts order



tlsServer.sh invokes the server process with the desired cipher suite

tlsClient.sh invokes the client process using the IP address of the server.

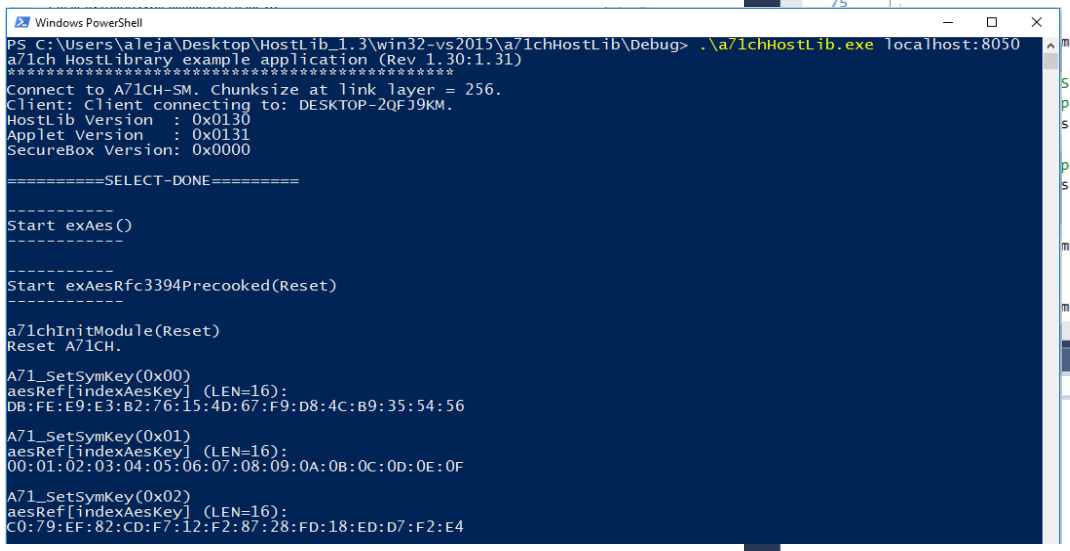


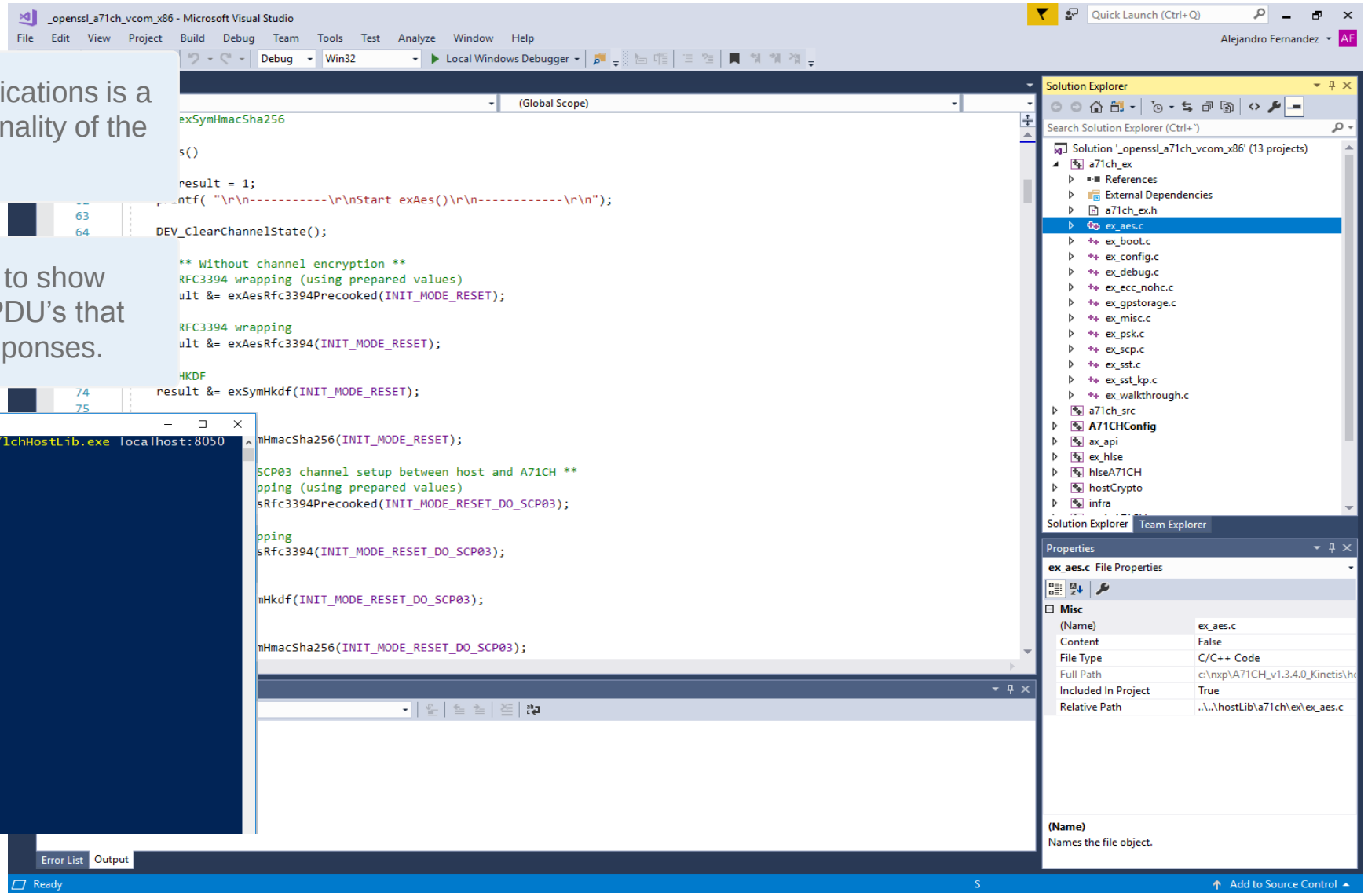
Using A71CH
Host API examples

A71CH Host API examples build environments - Windows

mainA71CH Host API usage example applications is a sample project oriented to show the functionality of the A71CH Host library using A71CH Host API

A set of use cases is sequentially executed to show the user each Host API function call, the APDU's that are sent to the A71CH and the received responses.



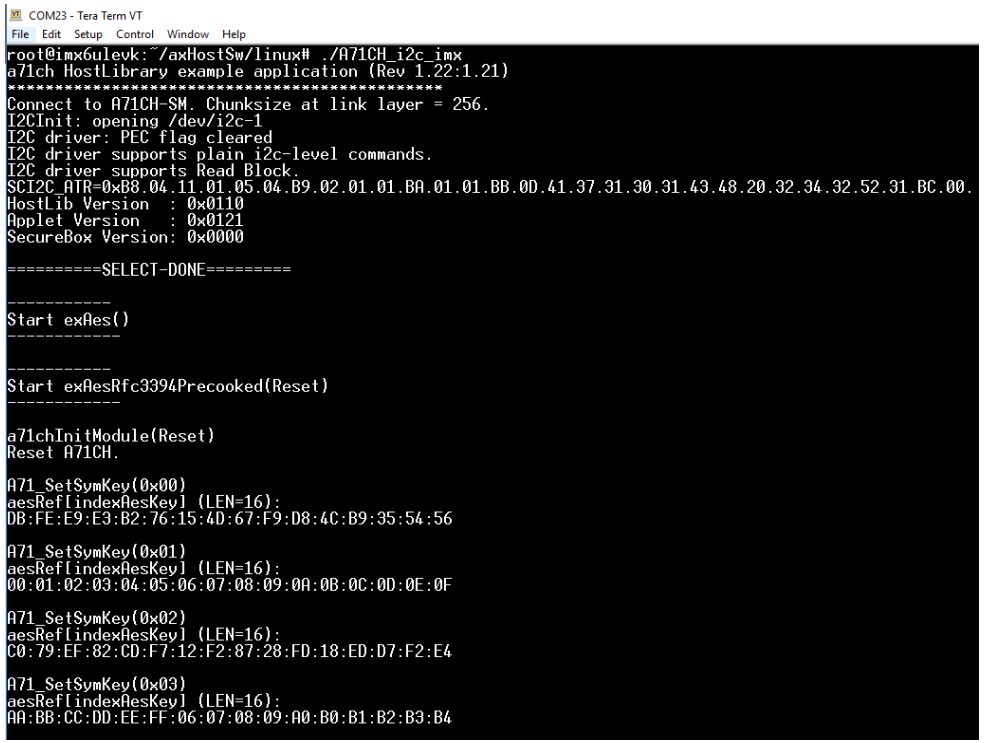


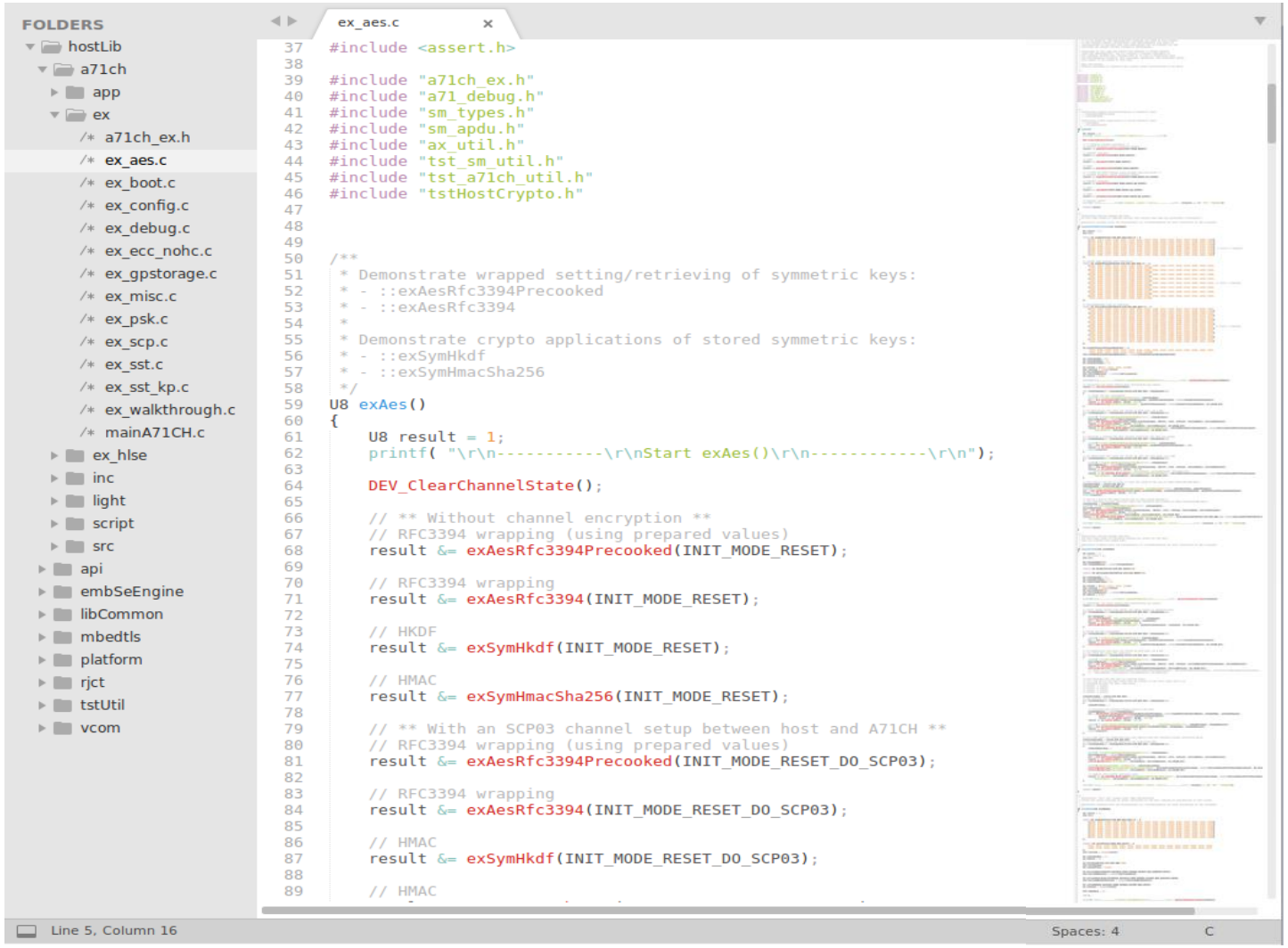
A71CH Host API examples build environments - Linux

The Host SW package comes bundled with a makefile in the directory ***~/…/axHostSw/linux: Makefile_A71CH***.

Requirements:

- A gcc compiler suite installed.
- OpenSSL 1.0.x library (also the development files) is available on the platform.





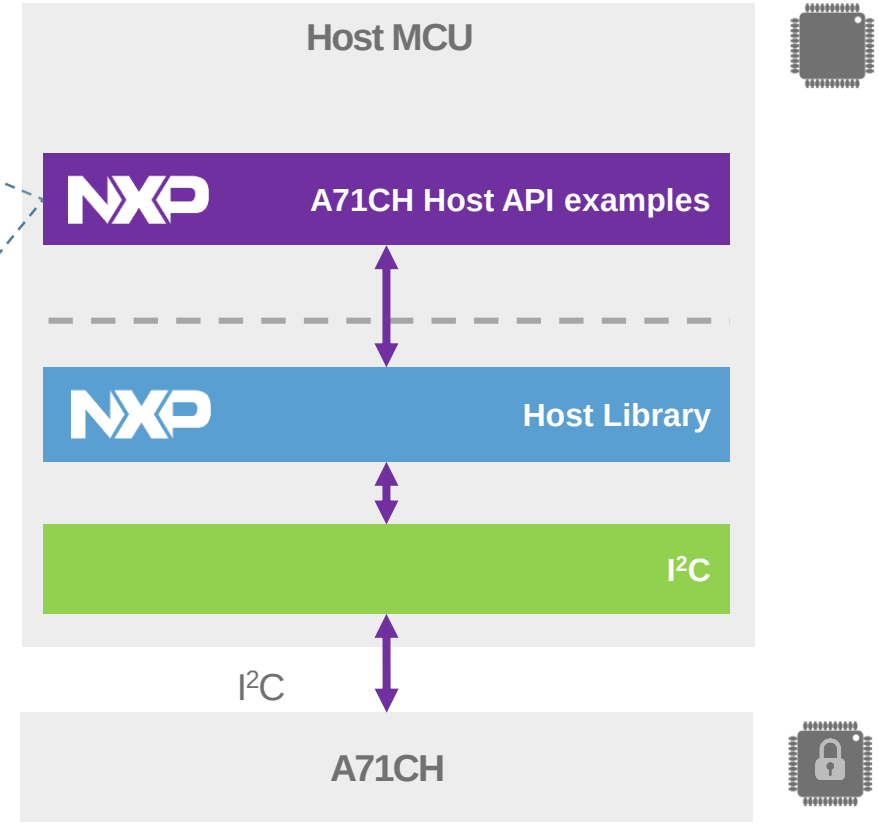
Linux Cross Compile for i.MX6UltraLite: [i.MX_Yocto_Project_User's_Guide]

A71CH Host API examples

The A71CH Host API examples are source code examples demonstrating the use of the A71CH Host API.

List of A71CH Host API examples:

- **ex_aes.c:** Example invocation of symmetric key related crypto functionality of the A71CH.
- **ex_config.c:** Example of storage and usage of configuration keys.
- **ex_gpstorage.c:** Example of the use of the general purpose storage and monotonic counter functionality..
- **ex_misc.c:** Example of other features (fetch a random number, calculate SHA256, module info, etc).
- **ex_psk.c:** Example of plain or ECDH enhanced pre-shared, key-based master key creation.
- **ex_scp.c:** Example of how to set up an SCP03 channel between the Host and the A71CH.
- **ex_sst.c:** Example of storage of symmetric and public keys..
- **ex_boot.c:** Example of the handover of SCP03 session keys from bootloader to OS.
- **ex_walkthrough.c:** Example of the usage of the A71CH from a system integrator perspective.
- ...and more



Using A71CH
TLS client example

A71CH TLS client example

A71CH TLS client example:

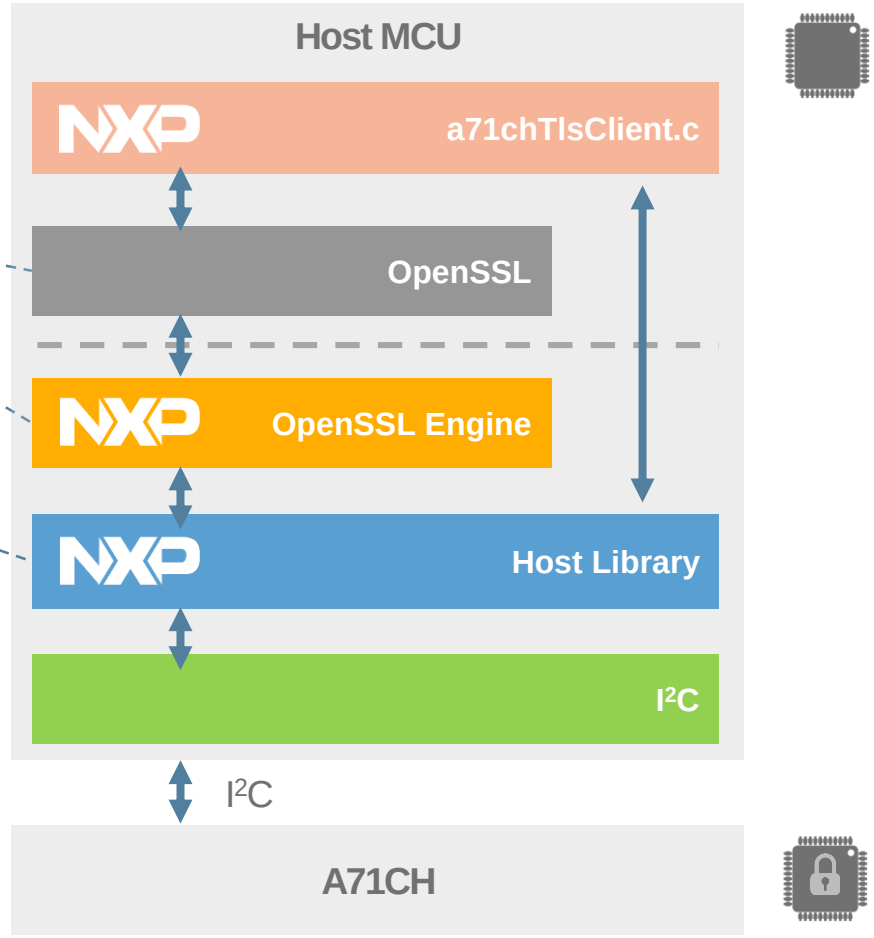
It uses the A71CH OpenSSL Engine and the OpenSSL API to establish the TLS link

It uses the A71CH Host Library to access the device certificate and it has an example that reads out data from a data object in GP storage.

`a71chTlsClient.c` example provided is a lot easier to digest/understand than the `openssl s_client` that comes with an OpenSSL installation

`./a71chTlsClient.sh:`

This bash script wraps the execution of a71chTlsClient.c binary



Training Mobile Knowledge

Closure

A71CH Plug & trust for IoT: Summary

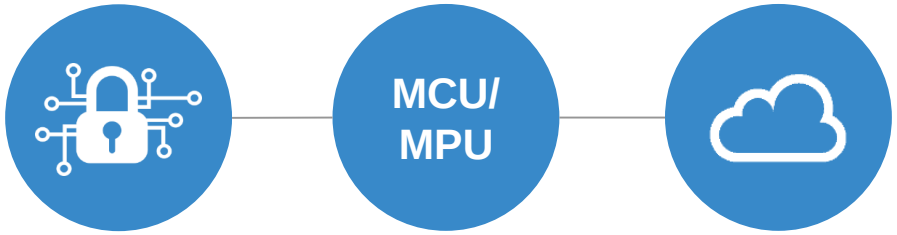
A71CH key benefits

- Root of trust for IoT applications
- End-to-end security, from chip to edge to cloud
- Plug & Trust: Ready to use solution for easy system integration

Support for:

- MPU with i.MX 6 available now
- MCU with Kinetis K64F, KW41Z, K82 and more in April

A71CH as an easy add-on to MPU & MCU for Secure Cloud Connection & Mutual Authentication



Product website: www.nxp.com/A71CH

Development kit: www.nxp.com/OM3710

Order info:

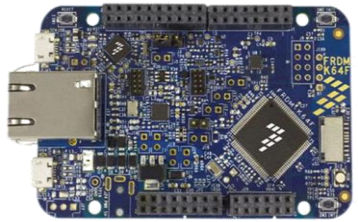
Item	Description	Package	12NC
A7101CHTK2	Security IC with standard temp range (-25 to +85 °C)	HVSON8, Reel, MoQ = 6k	9353 680 97118
A7102CHTK2	Security IC with extended temp range (-40 to +90 °C)	HVSON8, Reel, MoQ = 6k	9353 635 15118
OM3710/A71CHARD	OM3710/A71CHARD A71CH Arduino-compatible development kit		9353 689 97598

Coming soon: A71CH Kinetis support!

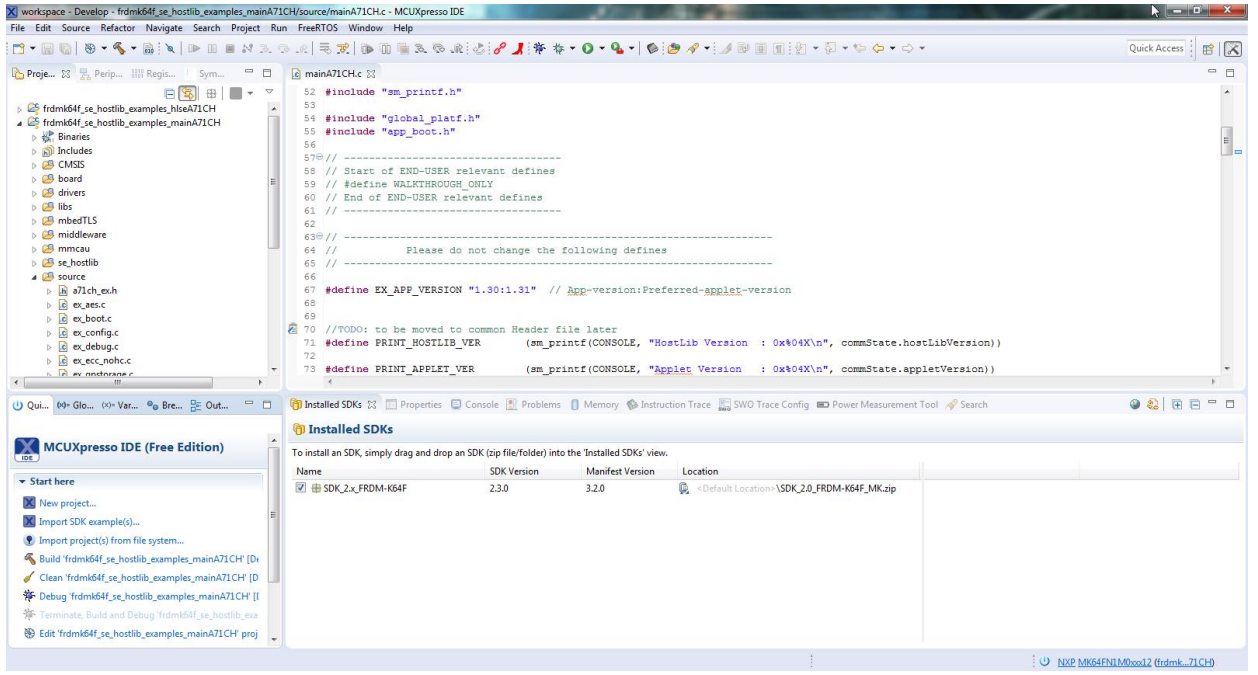
A71CH Arduino compatible development kit

FRDM-KW41Z

FRDM-K64F



FRDM-K82F



Stay tuned!

Training Mobile Knowledge

A71CH – Plug & trust for IoT

Jordi Jofre (Speaker)
Angela Gemio (Host)

Thank you for your kind attention!

Please remember to fill out our **evaluation survey** (pop-up)

Check your email for **material download** and on-demand **video** addresses

Please check NXP and MobileKnowledge websites for **upcoming webinars** and **training sessions**

http://www.nxp.com/support/classroom-training-events:CLASSROOM-TRAINING-EVENTS

www.themobileknowledge.com/content/knowledge-catalog-0

MobileKnowledge

MobileKnowledge is a team of HW, SW and system engineers, experts in **smart, connected and secure** technologies for the IoT world. We are your ideal **engineering consultant** for any specific support in connection with your **IoT** and **NFC** developments. We design and develop secure HW systems, embedded FW, mobile phone and secure cloud applications.

Our services include:

- **Secure hardware design**
- **Embedded software development**
- **NFC antenna design** and **evaluation**
- **NFC Wearable**
- **EMV L1 pre-certification support**
- **Mobile** and **cloud application development**
- **Secure e2e system design**

www.themobileknowledge.com
mk@themobileknowledge.com

We help companies leverage
the **secure IoT revolution**