Trust X BootCamp Quick Reference

Important:

The Bootcamp source code is forked from official Infineon GitHub. It is frozen and modified for training purposes. Hence, it is neither updated regularly nor qualify for production. Please refer to official Infineon GitHub https://github.com/Infineon/arduino-optiga-trust-x for the updated and latest driver release.

Trust X BootCamp Github: http://bit.ly/trustxbootcamp

Full URL: https://github.com/gilatoes/arduino-optiga-trust-x

BootCamp Task Description: http://bit.ly/bootcamptasks

Full URL: https://github.com/gilatoes/arduino-optiga-trust-x/blob/master/missions/tasks.md

Additional Boards Manager URL:

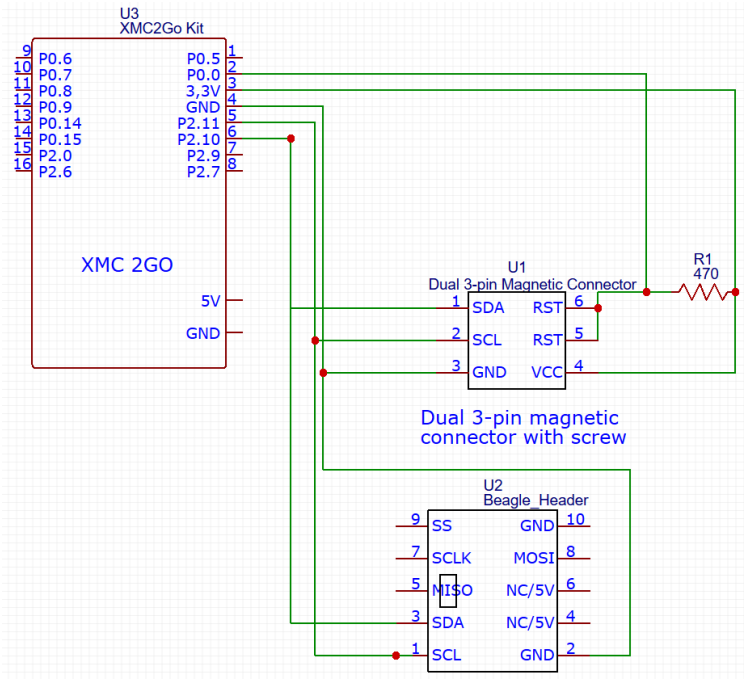
https://github.com/Infineon/Assets/releases/download/current/package_infineon_index.json

Arduino Trust X Library Path:

C:\Users\<*username*>\Documents\Arduino\libraries\arduino-optiga-trust-x\

Trust X Quiz: https://kahoot.it/

Exerciser Schematic:



Task 0 Checklist:

Check list below will help to determine if you have completed Task 0 correctly.

- [ ] Take note of the UART communication port.

Windows Key► Device Manager► Port (COM & LPT) ► JLink CDC UART Port

- [ ] Start E00_Minimal example and ensure that it can be compiled without any error.
- [ ] Check if OpenSSL installation.
  1. Windows Key► cmd ►path ►C:\OpenSSL-Win64\bin
  2. Start Windows Powershell. Windows Key►Windows Powershell
  3. “where.exe openssl” - check the correct path of the Openssl.
  4. "openssl version" - check if the openssl is installed correctly.
  5. “openssl speed” - check that it is operational.
- [ ] Access the OpenSSL sandbox folder from default path:

C:\Users\<username>\Documents\Arduino\libraries\arduino-optiga-trust-x\OpenSSL_sandbox

Try out some of the following commonly used OpenSSL commands for this Bootcamp:

Check OpenSSL version

```
openssl version
```

Generate ECC Private key

```
openssl ecparam -name prime256v1 -genkey -noout -out <private.key.pem>
openssl ecparam -name secp384r1 -genkey -noout -out <private.key.pem>
```

Generate ECC Public Key

```
openssl ec -in <private.key.pem> -outform PEM -pubout -out <public.key.pem>
```

Display certificate

```
openssl x509 –noout –text –in <cert.pem>
```

Verify certificate chain

```
openssl verify -CAfile <RootCA_cert> -untrusted <IntermediateCA_cert>
<EndDevice_cert>
```

Hash a file message

```
openssl dgst -sha256 <message_file>
```

Create a new self-signed CSR

```
openssl req –new –key <privatekey.key> -out request.csr –sha256
```

Verify CSR

```
openssl req -noout -text -in <cert_request.csr>
```

Convert DER to PEM format

```
openssl x509 –inform der –in <cert.der> –out <cert.pem>
```

Convert PEM to DER format

```
openssl x509 –outform der –in <cert.pem> –out <cert.der>
```

Note: For this BootCamp, focus only on ECC and SHA256 algorithm. Ignore RSA.

TASKS AND MISSION

Task 1: HelloBootCamp

- [ ] What is the current XMC and Trust X library version?

- [ ] Find out which file stores the Trust X library version number?

Task 2: Understanding I2C and GPIO

- [ ] Download the pre-captured Saleae Logic file. What is the I2C bus speed used in the Logic Analyzer trace?

- [ ] The default host I2C driver is 100KHz. Modify to increase your host I2C driver I2C bus frequency.

- [ ] Get a round mold Trust X and replace it in your Exerciser. Determine its address. Modify the I2C host library to communicate with this Trust X and read out its UID.

- [ ] Restore the host library Trust X address to the original value.

- [ ] Make sure that your host library is using the default I2C value.

Implements the Trust X reset using GPIO:

1. Implement the reset control using GPIO. Reset control improves the system stability.
2. Execute the H02_ChangeI2CAddress example which will temporarily change the I2C address.
3. If the reset is successfully implemented, the I2C address will be restored to the default value during initialization. You can use other examples such as E03_GetUniqueID to check. In the event of no proper reset, I2C address will not be found.
4. Manually, power down and up the device will also restore the default Trust X address.

Task 3: Trust X Object IDs

- [ ] Decode the Trust X UID to determine the Trust X identity.

- [ ] Read Meta data of the Device Public Key certificate issued by Infineon data object.

- [ ] Read and decode Trust X factory default certificate. Determine the signature algorithm, certificate serial number, issue date, valid period, public key and signature value issued by Infineon CA.

Task 4: Concepts of Digital Signature using ECC Asymmetric keys

- [ ] If we purely use the public key and secret key to perform verification of data message what is the potential problem?

- [ ] What is the potential weakness using such approach?

- [ ] After verifying the message, can Bob really trust its contents?

Task 5: Replay attack

1. What is the significance of Trust X generating the signature?
2. Why the verification process passes even on different signature?
3. What is the ECC signature? What are the components of ECC signature?
4. Is it possible to prevent such replay attack?

Task 6: Simplified Server Authentication

Hands on task

Task 7: Unique authentication and optimization

Compare hardware and software verification approach.

- [ ] Compete with your camp mates for the least Exerciser memory foot print. Check the compiled memory footprint, least memory usage wins!
- [ ] Compete the speed of the 1-way Authentication. Least time wins!

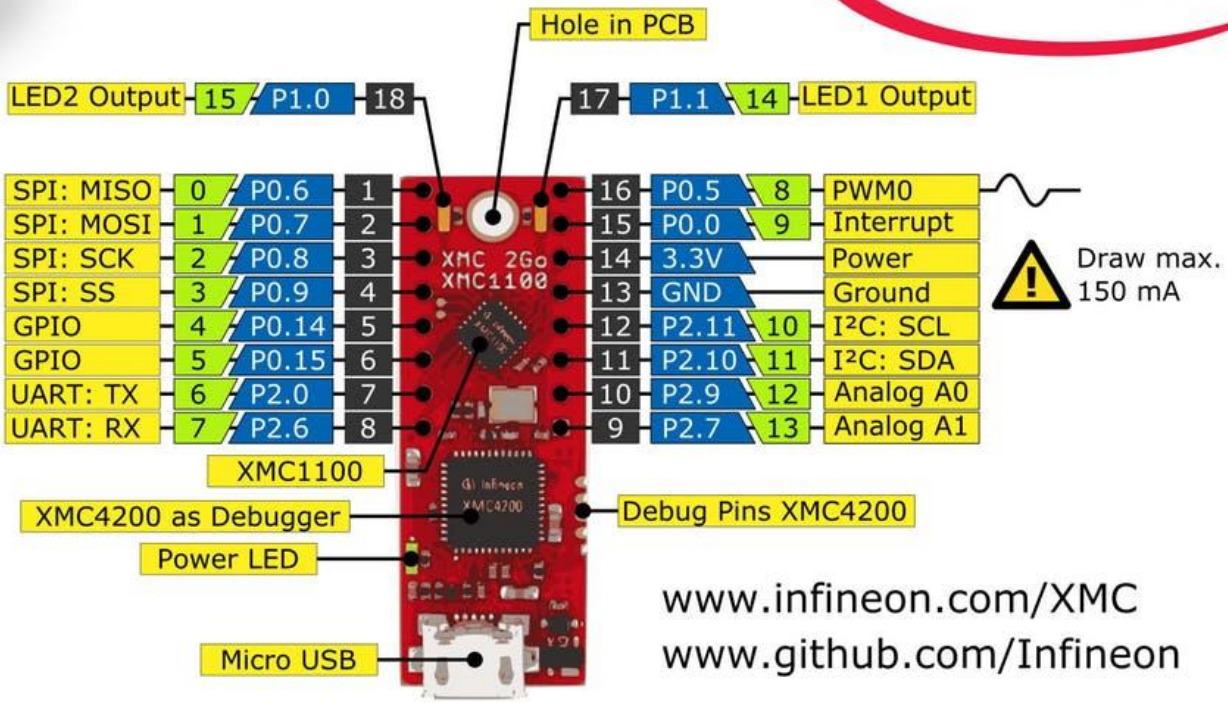
	Result
Memory footprint	
Processing Speed	

Mission: Simplified Firmware Update

	Public Key (68 bytes)
My Generated key (Provision Process) Important note: Disable provision me macro after generating the key	
Exchanged Public Key	
Calculated Shared Secret	
Derived Secret	

Hint : useful websites such as https://crytii.com can be used to convert data format.

The XMC1100 XMC2Go Pin Out for Arduino



Legend

- Information
- Labelling of Pins in Datasheet
- Pin Number in Arduino IDE
- Physical Pin Number
- Warning
- Additional Information

www.infineon.com/XMC
www.github.com/Infineon

- The LED1 and LED2 are exclusively connected to the respective pins
- If board is powered through 3.3V pin, it is not recommended to power through USB and vice versa
- Only two communication protocolls can be used at the same time, e.g. SPI and UART, but than not I²C
- On-board UART TX 6 / RX 7 share not the same channel as the debug UART via the USB connection

V1.0.2