



life.augmented

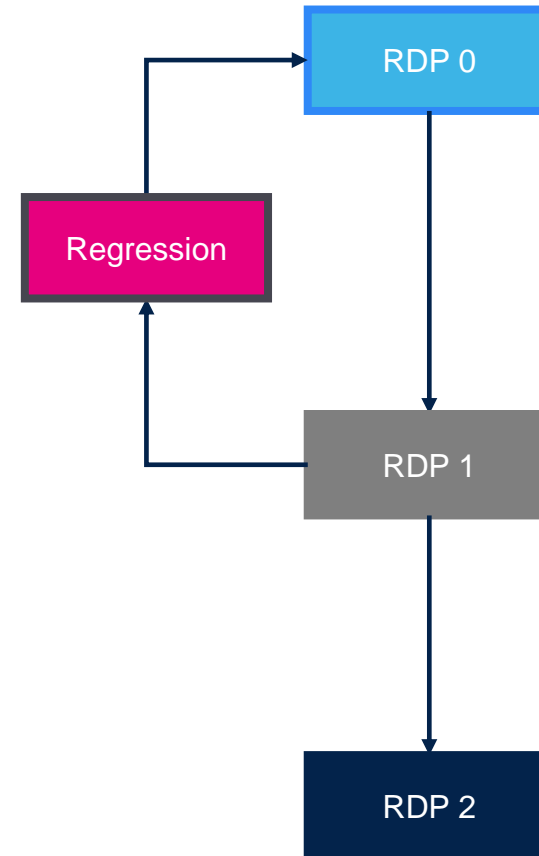
# STM32H5 Device Lifecycle Management

# Introduction

- STM32H5 implements a new mechanism for the protection of the code stored in internal flash
- Let's see the impact of this change

# Reminder of RDP protection on legacy STM32

- RDP 0 : Open state dedicated to development
- RDP 1
  - Firmware in flash is protected from readout
  - Debugger can attach and read ram content
  - Possible regression to RDP0 with automatic flash erase
  - State mostly used because of this regression capability
- RDP2
  - No debugger access, no possible regression

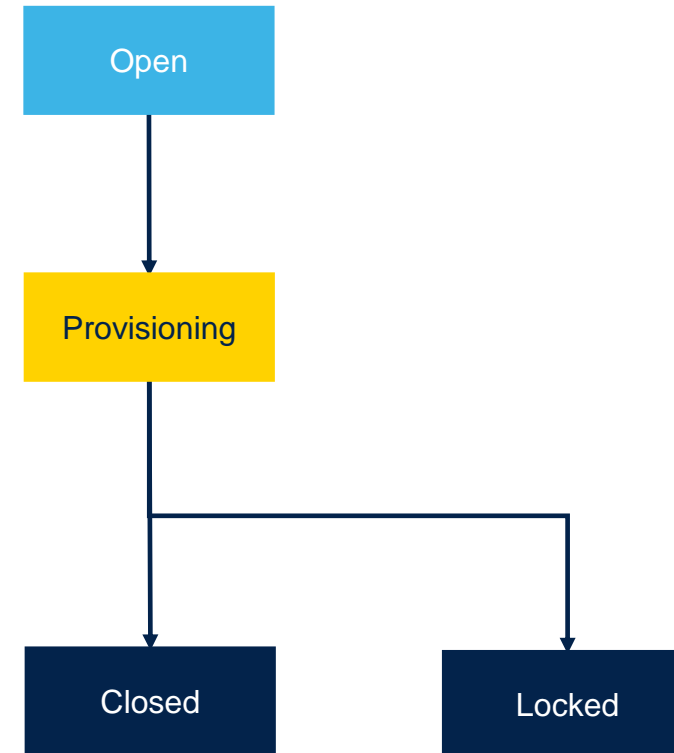


# STM32H5 evolution

- RDP levels replaced by PRODUCT\_STATE
  - Option byte in both cases
  - RDP values fixed for RDP 0 (0xAA) and RDP 2 (0xCC). All other values mean RDP 1
  - PRODUCT\_STATE have fixed value for each state. No default
- Regression control mechanism : Debug Authentication
  - JTAG dedicated access point
  - ADAC protocol defined by ARM
  - Password used for regression
  - Certificate used when enabling TrustZone of Cortex M33 (Case not detailed here)

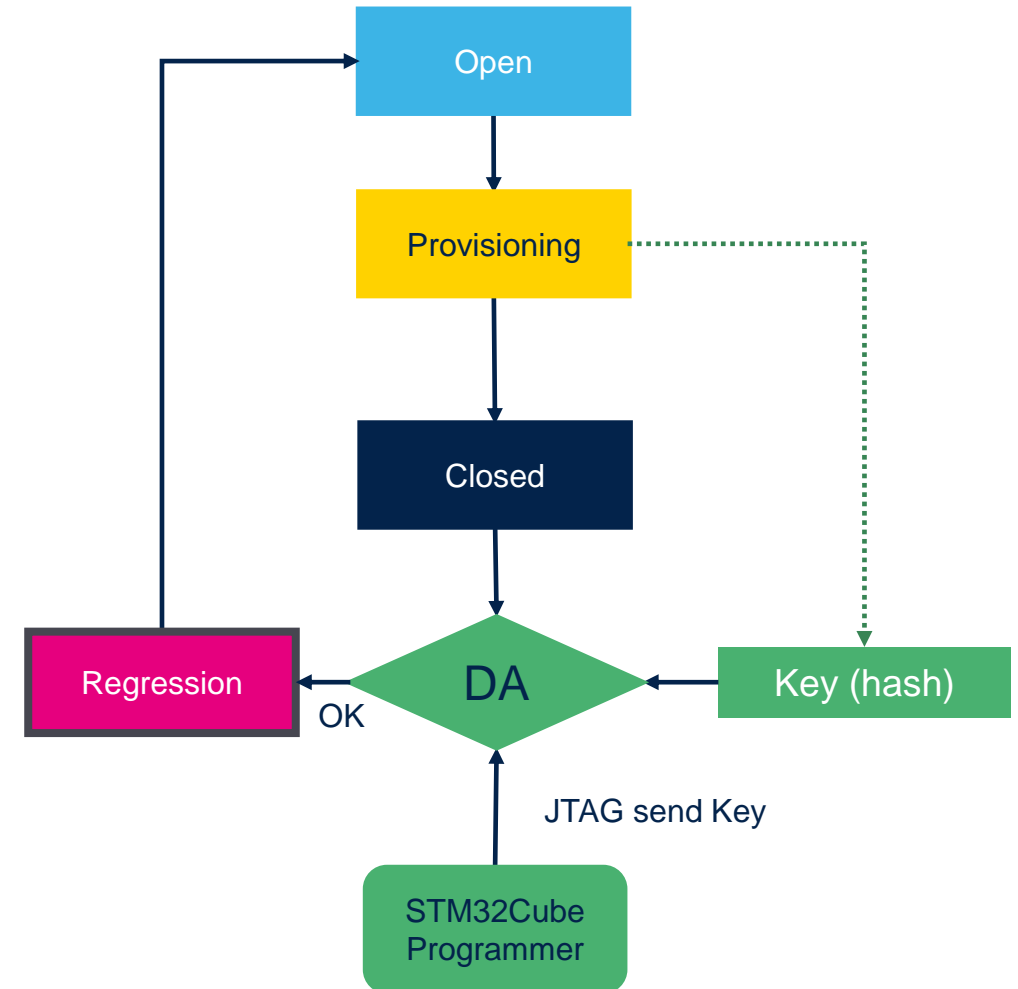
# Product State

- Open state dedicated to development
- Provisioning allows transmitting specific file containing keys and data to be provisioned
- Closed and Locked are used in the field to protect device.
  - Closed state allows possible regression
  - Locked state is definitive



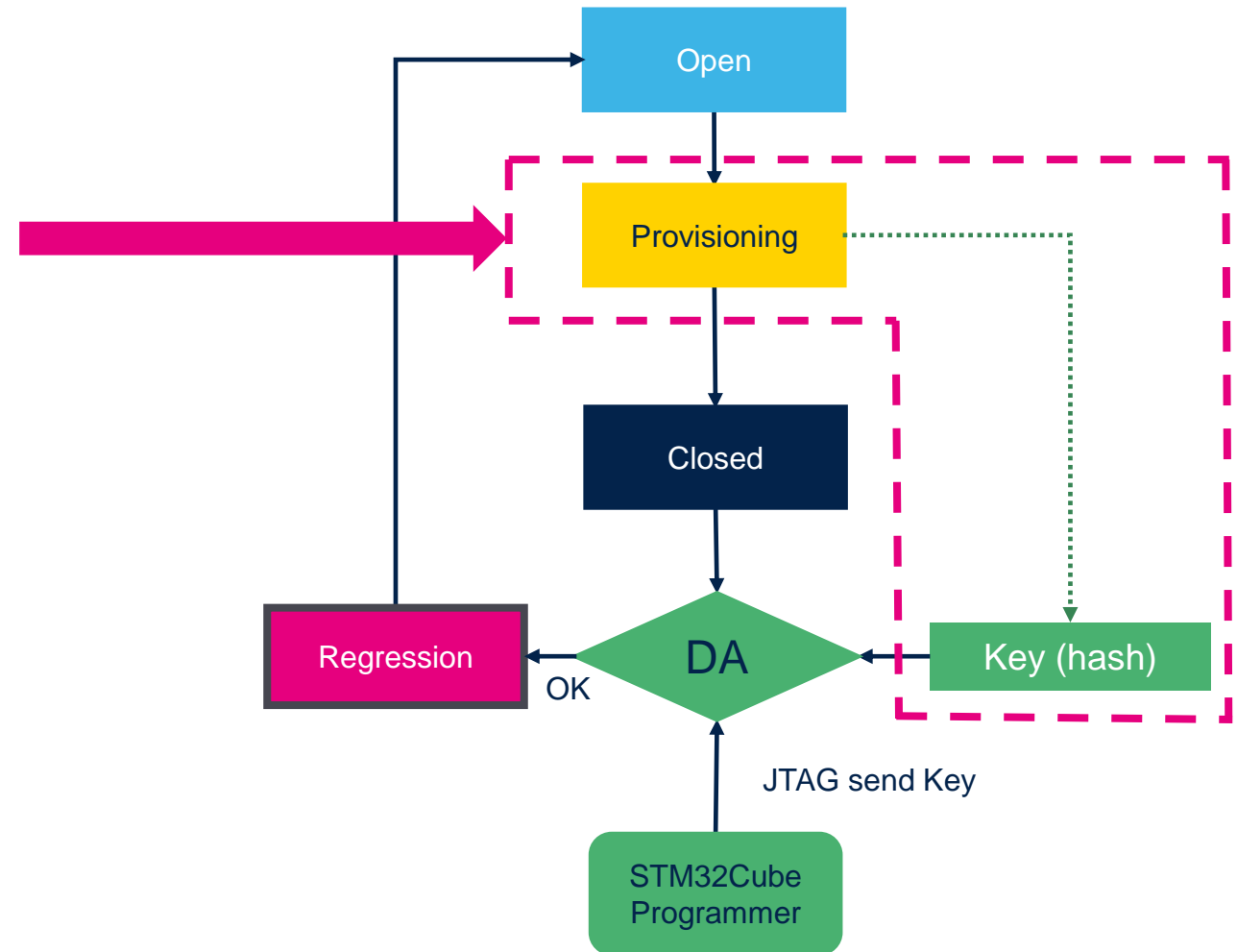
# Regression controlled by Debug Authentication

- Firmware can be flashed in open state
- Provisioning is used in production to transmit auth key to the secure storage
- Close device : no more debug access
- Field return : use key to open the device securely through JTAG/SWD interface using dedicated access point. Flash content will be erased.



# Impact on production process

New step required to have the capability to perform regression



# Let's see how it works

- Next part will show you how to ensure the regression capability when flash code protection is activated on STM32H5
- We will use the tools provided in STM32 ecosystem

- STM32TrustedPackageCreator



- STM32CubeProgrammer





# Hands-on purpose

- Application firmware with TZ disabled
- Use case could be compared to RDP usage in STM32F4
- Simple LED blink application
- When TZ is disabled we use a password for debug authentication
- This demo shows the steps to provision this password in order to enable the regression

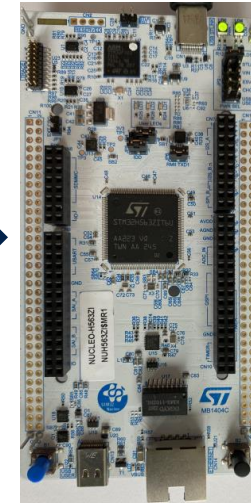
# Debug Authentication Hands-on Steps

- TZEN disabled

- Step 1 : Create provisioning file (obk file)



- Step 2 : Flash blinking LED binary
- Step 3 : Move to **PROVISIONING** state
- Step 4 : Provision the obk file to the board
- Step 5 : Move to **CLOSED** state
- Step 6 : Back to original state: regression



# Demo sum-up

In this short demo we have seen how to provision a password on STM32H5 to enable secure regression capability

# Take away

- STM32H5 replaces RDP by PRODUCT\_STATE
- STM32H5 introduces Debug Authentication feature to control regression with a password
- The provisioning of this password adds a step in production
- By disabling debug interface, this new mechanism increases the robustness of the flash protection
- The Debug Authentication offers even more capabilities when TrustZone is enabled. This will be presented in STM32H5 security workshop



life.augmented

# **Debug Authentication hands-on TZ disabled**



# Hands-on purpose

- Application firmware with TZ disabled
- Use case could be compared to RDP usage in STM32F4
- Simple LED blink application
- When TZ is disabled we use a password for debug authentication
- This demo shows the steps to provision this password in order to enable the regression

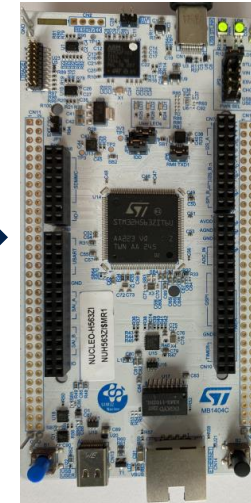
# Debug Authentication Hands-on Steps

- TZEN disabled

- Step 1 : Create provisioning file (obk file)



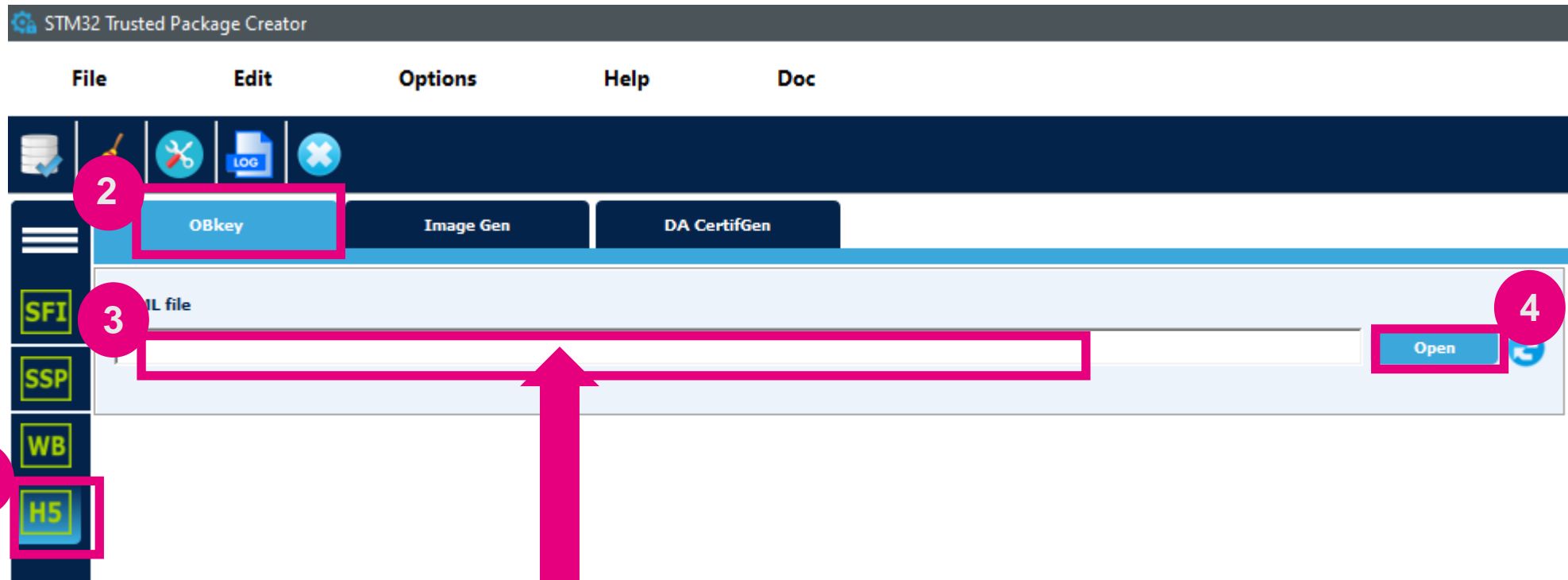
- Step 2 : Flash blinking LED binary
  - Step 3 : Move to **PROVISIONING** state
  - Step 4 : Provision the obk file to the board
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression





# Step 1 : Create provisioning file (obk file)

- Launch STM32Trusted Package creator  
Select DebugAuthentication\_NoTZ\Config\DA\_ConfigWithPassword\_H563



...\\DebugAuthentication\_NoTZ\\Config\\DA\_ConfigWithPassword\_H563.xml



# Step 1 : Create provisioning file (obk file)

The screenshot displays the STM32 Trusted Package Creator application window. The 'OBkey' tab is selected in the top navigation bar. On the left sidebar, the 'H5' option is highlighted. The main workspace is divided into two sections. The left section contains an 'XML file' field with the path 'pp/STM32H5\_Generic/DebugAuthentication\_NoTZ/Config/DA\_ConfigWithPassword\_H563.xml' and an 'Open' button. Below it is a 'Password' field containing the value '0123456789012345', which is highlighted with a pink box and labeled 'Password value'. The right section, titled 'DA Config', contains an 'Information' table with the following data:

Information	Value
OBkey Destination Address	0x0ffd0100
Encryption Option	disable
Size	

Annotations with arrows point from the table to labels: 'Storage address' points to '0x0ffd0100', 'No encryption on H563' points to 'disable', and 'Provisioning data size, computed at generation' points to the empty 'Size' field. Below the table is an 'Output File' field with the path '..\..\Binary\DA\_ConfigWithPassword\_H563.obk' and a 'Select Path' button. A 'Tool Description' section provides information about the tool's purpose and lists possible XML configurations. At the bottom right, a pink box highlights the 'Generate OBkey' button. A warning message 'Encryption option is disabled!' is displayed at the bottom center.

STM32 Trusted Package Creator

File Edit Options Help Doc

life.augmented

OBkey Image Gen DA CertifGen

SFI SSP WB H5

XML file

pp/STM32H5\_Generic/DebugAuthentication\_NoTZ/Config/DA\_ConfigWithPassword\_H563.xml Open

Password

0123456789012345

Password value

DA Config

Information

Information	Value
OBkey Destination Address	0x0ffd0100
Encryption Option	disable
Size	

Storage address

No encryption on H563

Provisioning data size, computed at generation

Output File

..\..\Binary\DA\_ConfigWithPassword\_H563.obk Select Path

Tool Description:

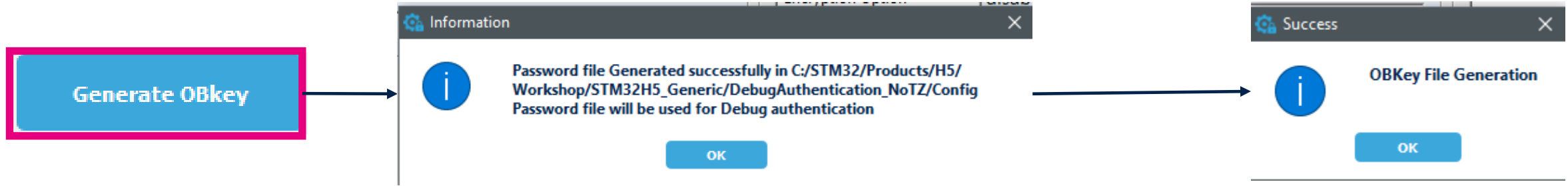
The purpose of this tool is to generate provisioning data also called obk files.  
The file can be generated only with XML input.  
Possible XML configuration for OBKey are:  
1.DA (Debug Authentication) configuration XML  
2.STiROT configuration XML  
3.OEMiROT configuration XML  
4.OEMuROT configuration XML  
-Examples of XML can be found in CubeFW package in projects\<board>\ROT\_Provisioning

Encryption option is disabled!

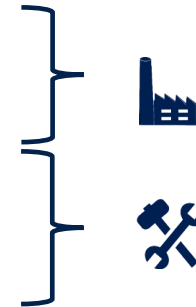
Generate OBkey



# Step 1 : Create provisioning file (obk file)



- Output files
  - OBK provisioning file: **Binary**\DA\_ConfigWithPassword\_H563.obk
  - AssociatedFormatted password file : **Config**\password.bin



# Debug Authentication Hands-on Steps

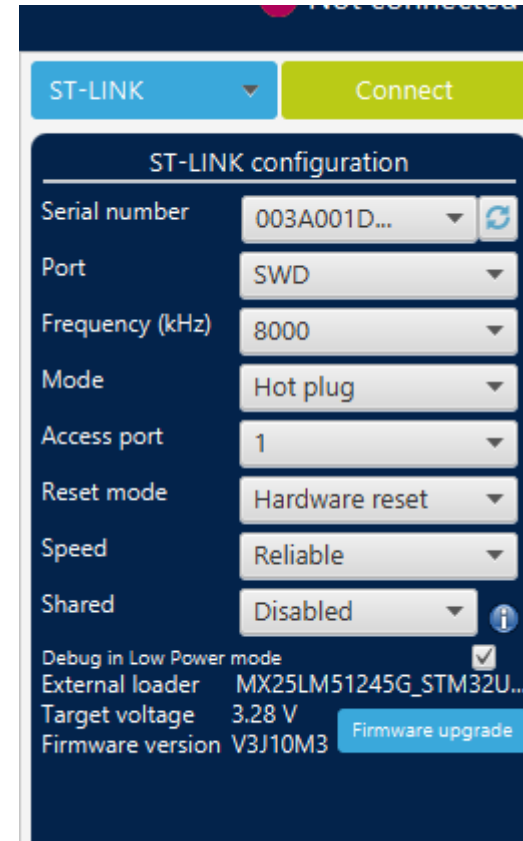
- TZEN disabled
  - ~~Step 1 : Create provisioning file (obk file)~~
  - Step 2 : Flash blinking LED binary
  - Step 3 : Move to **PROVISIONING** state
  - Step 4 : Provision the obk file to the board
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression



Prg

## Step 2 : Flash blinking LED binary

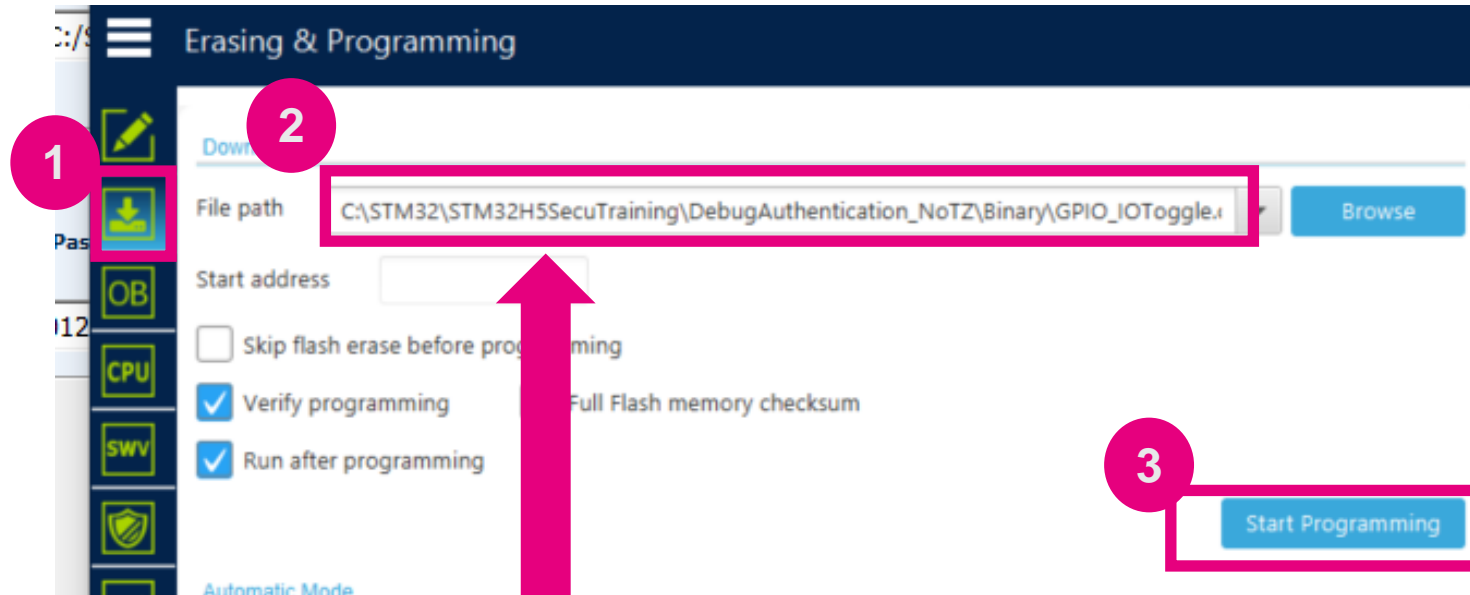
- Open STM32CubeProgrammer
- Connect to target in hotplug mode



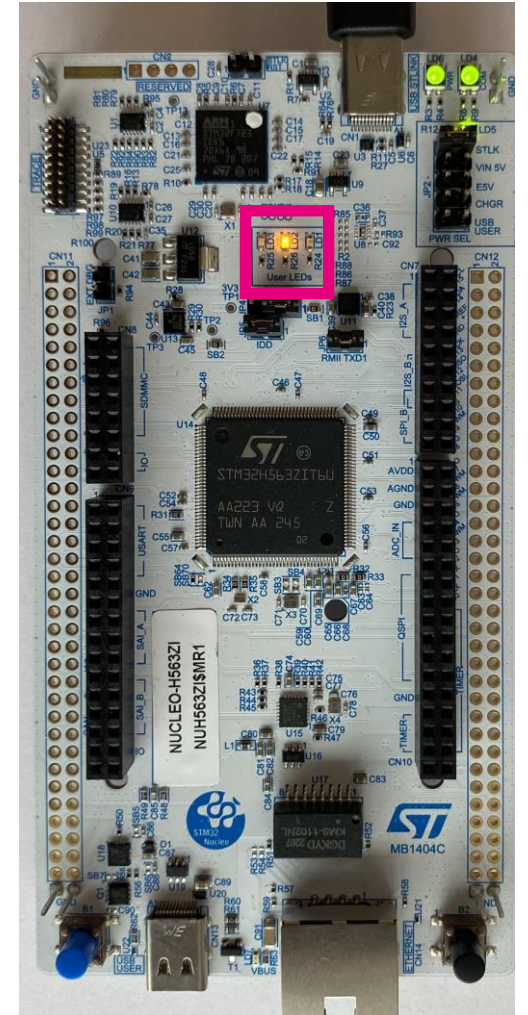


Prg

## Step 2 : Flash blinking LED binary



...\DebugAuthentication\_NoTZ\Binary\GPIO\_IOToggle.out



# Debug Authentication Hands-on Steps


- TZEN disabled
  - ~~Step 1 : Create provisioning file (obk file)~~
  - ~~Step 2 : Flash blinking LED binary~~
  - Step 3 : Move to **PROVISIONING** state
  - Step 4 : Provision the obk file to the board
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression



Prg

# Step 3 : Move to **PROVISIONING** state

Option bytes

1 

2


Name	Value
PRODUCT_STATE	ED
BOR Level	17
User Configuration	2E
User Configuration 2	C6
Boot Configuration	72
Bank1 - Flash watermark area definition	5C
Write sector group protection 1	9A
	A3

Life state code.  
ED : Open  
17 : Provisioning, Debug partially opened (only non-secure)  
2E : iRoT-provisioned, Debug partially opened (only non-secure)  
C6 : TZ-Closed, Debug partially opened (only non-secure)  
72 : Closed, Debug disabled, regression is possible

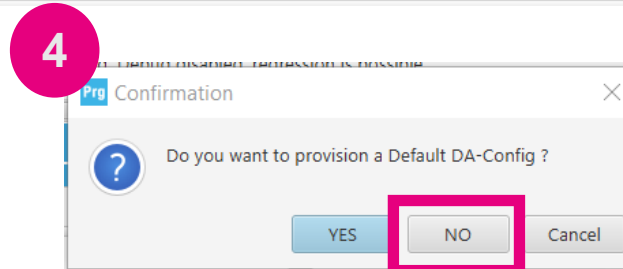
Product state

Name	Value
PRODUCT_STATE	17

Life state code.  
ED : Open  
17 : Provisioning, Debug partially opened (only non-secure)  
2E : iRoT-provisioned, Debug partially opened (only non-secure)  
C6 : TZ-Closed, Debug partially opened (only non-secure)  
72 : Closed, Debug disabled, regression is possible

3 

Some of the option bytes might be hidden or clipped, Use the mouse wheel or the touch pad to scroll down



# Debug Authentication Hands-on Steps

- TZEN disabled
  - ~~Step 1 : Create provisioning file (obk file)~~
  - ~~Step 2 : Flash blinking LED binary~~
  - ~~Step 3 : Move to **PROVISIONING** state~~
  - Step 4 : Provision the obk file to the board
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression





Prg

## Step 4 : Provision the obk file to the board

1

2

3

RDP regression with password SFI/ SFlx **OBKey Provisioning** Debug Authentication SSP

OBKey file path

Select File

OBKey Provisioning Description

- The OB key file is used to program OBKey content.
- OBKey files are generated by STM32 Trusted Package Creator based on XML template.
- OBK file info:
  - 1) Destination address: OBKey physical address.
  - 2) DoEncrypt: whether it's needed to request OBKey encryption or not.



Prg

## Step 5 : DA provisioning

Secure programming

RDP regression with password | SFI/ SFlx | **OBKey Provisioning** | Debug Authentication | SSP

1

Select File: C:\STM32\Products\H5\Workshop\STM32H5\_Generic\DebugAuthentication\_NoTZ\Binary\DA\_C

Browse

OBKey Provisioning Description

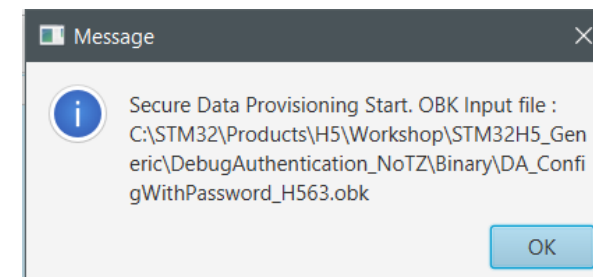
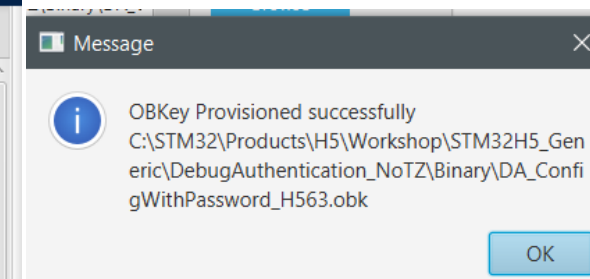
- The OB key file is used to program OBKey con
- OBKey files are generated by STM32 Trusted P ge Creator based on XML template.
- OBK file info:
  - 1) Destination address: OBKey physical ad
  - 2) DoEncrypt: whether it's needed to requ OBKey encryption or not.

Note for DA:  
DA provisioning with certificate is valid only when  
DA provisioning with password is valid only when

name	value
Destination	0x0ffd0100
Size	0x00000060
Do Encrypt	00000000

2

Start Provisioning



...\DebugAuthentication\_NoTZ\Binary\DA\_ConfigWithPassword\_H563.obk

# Debug Authentication Hands-on Steps

- TZEN disabled
  - ~~Step 1 : Create provisioning file (obk file)~~
  - ~~Step 2 : Flash blinking LED binary~~
  - ~~Step 3 : Move to **PROVISIONING** state~~
  - ~~Step 4 : Provision the obk file to the board~~
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression



Prg

## Step 5 : Move to **CLOSED** state

1

2

3

Name	Value
PRODUCT_STATE	72

Life state code.  
ED : Open  
17 : Provisioning, Debug partially opened (only non-secure)  
2E : iRoT-provisioned, Debug partially opened (only non-secure)  
C6 : TZ-Closed, Debug partially opened (only non-secure)  
72 : Closed. Debug disabled. regression is possible

BOR Level  
User Configuration  
User Configuration 2

Some of the option bytes might be hidden or clipped, Use the mouse wheel or the touch pad to scroll down

Apply

Warning: Connection to device 0x484 is lost

Error: Reloading Option Bytes Data failed

Error: Uploading Option Bytes bank: 0 failed

Error: failed to reconnect after reset !

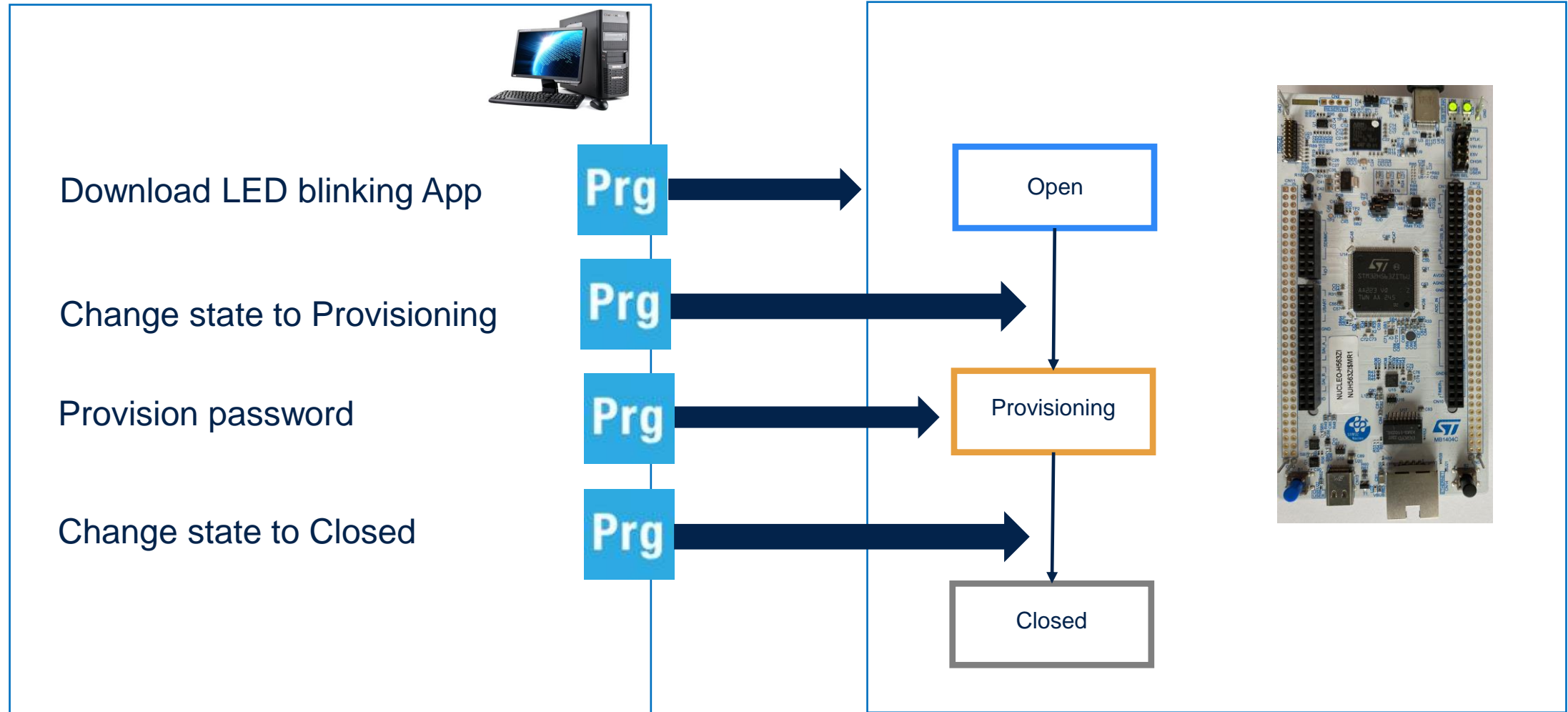
## Move to **CLOSED** state

- Device is now closed.
- No JTAG DEBUG connection is possible
- Firmware runs normally
- **The device is ready to go to the field**



TZEN=0

# STM32H5 Security Debug Authentication Provisioning



# Debug Authentication Hands-on Steps

- TZEN disabled
  - ~~Step 1 : Create provisioning file (obk file)~~
  - ~~Step 2 : Flash blinking LED binary~~
  - ~~Step 3 : Move to **PROVISIONING** state~~
  - ~~Step 4 : Provision the obk file to the board~~
  - Step 5 : Move to **CLOSED** state
  - Step 6 : Back to original state: regression



# Step 6 : Back to original state: regression

First disconnect the programmer !

Secure programming

2

RDP regression with password SFI/ SFIx OBKey Provisioning **Debug Authentication** SSP

3

Discover

1

Password File Path

Select File  Browse Full Regression

name	value
Locking Mechanism	Password
Soc ID	0x00000000 0x00000000 0x00000000 0x00000000
Life Cycle	<b>ST_LIFECYCLE_CLOSED</b>
Device ID	0x484

Step 1: Path selection.

Step 2: Execution.





Prg

## Step 6 : Back to original state: regression

Secure programming

RDP regression with password | SFI/ SFlx | OBKey Provisioning | Debug Authentication | SSP

Pass File Path

1

Select File

C:\STM32\Products\H5\Workshop\STM32H5\_Generic\DebugAuthentication\_NoTZ\Config\pass

Browse

2

Full Regression

Discover

name	value
Locking Mechanism	Password
Soc ID	0x00000000 0x00000000 0x00000000 0x00000000
Life Cycle	ST_LIFECYCLE_CLOSED
Device ID	0x484

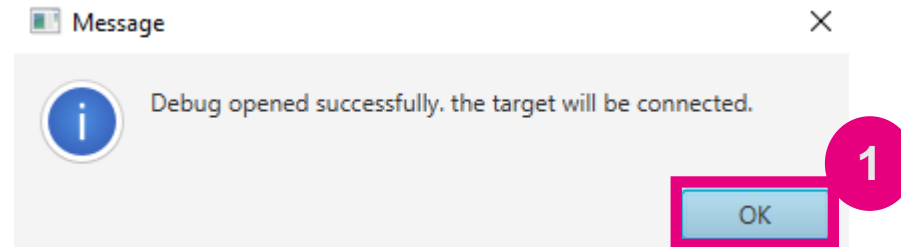
Step 1: Path selection.

Step 2: Execution.

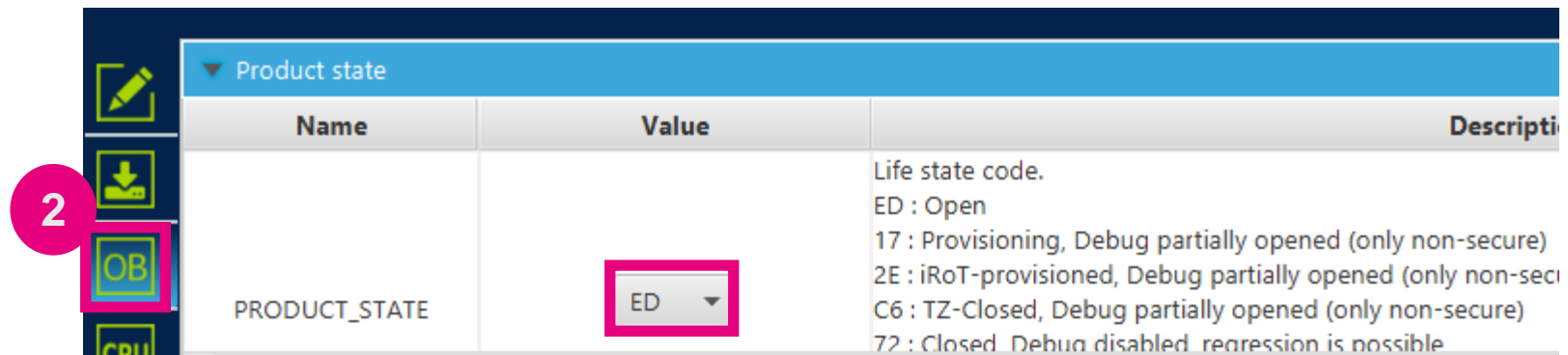
...\DebugAuthentication\_NoTZ\Config\password.bin



## Step 6 : Back to original state: regression



**Check product state after regression : It is open and LED is not blinking as flash is erased during the regression**

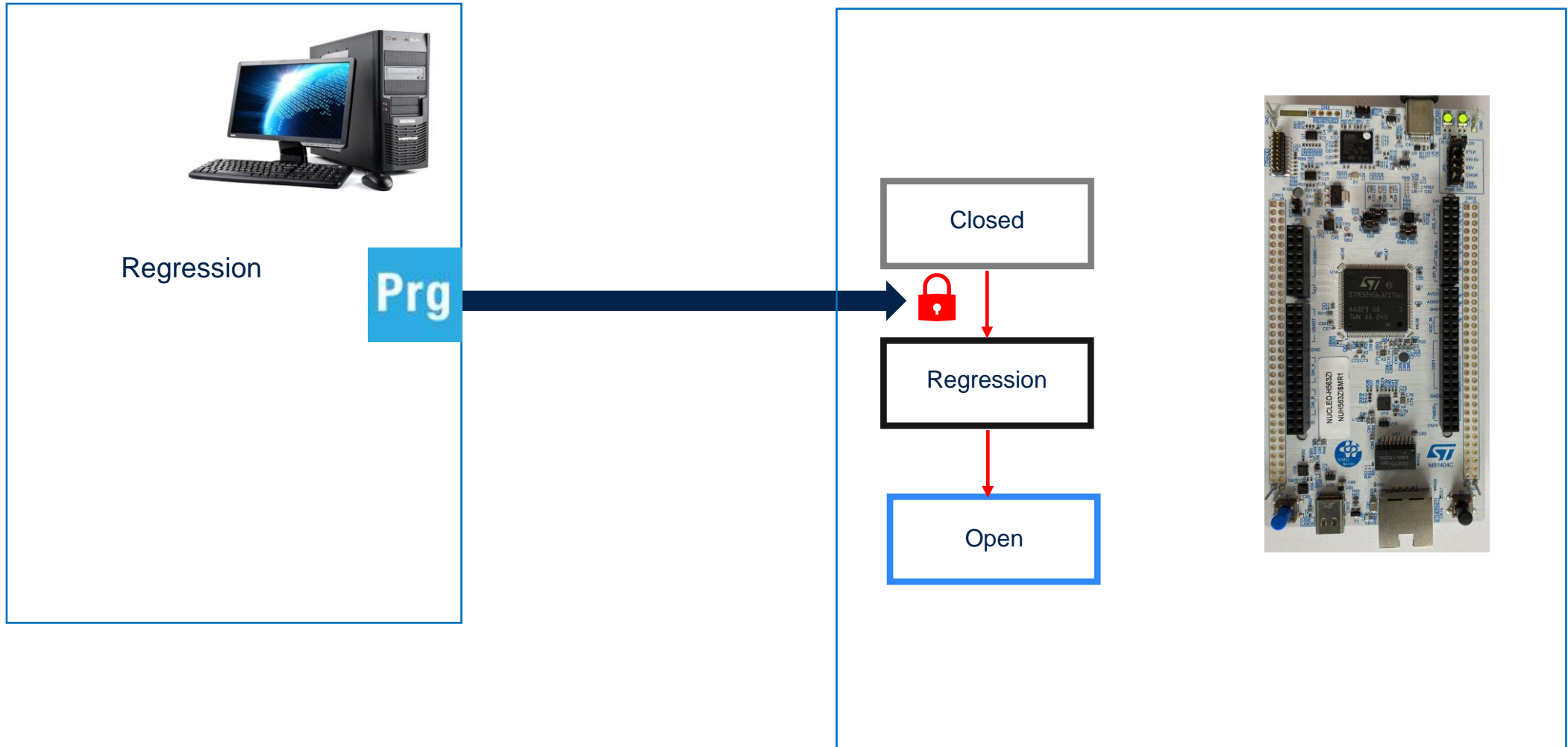




TZEN=0

# STM32H5 Security

## Debug Authentication for regression



# DA Hands-on no TZ take away

- We learned
  - How to create a password provisioning file (obk) with STM32TrustedPackageCreator
  - How to provision this obk file to the STM32H5 with STM32CubeProgrammer
  - How this provisioning is done in the production flow
  - How to use debug authentication interface to do a regression of the device