
Keysight Software and Hardware Crypto Training Target

DS1030A/31A Software and Hardware Crypto Training Target
v3.2 (with PQC)

Notices

© Keysight Technologies, Inc. 2024

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies, Inc. as governed by United States and international copyright laws.

Trademark Acknowledgments

Manual Part Number

DS1030-90003

Edition

Edition 1, October 2024

Published by:
Keysight Technologies
1400 Fountain Grove Parkway
Santa Rosa, CA 95403

Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED "AS IS," AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND

THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is "commercial computer software," as defined by Federal Acquisition Regulation

("FAR") 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement ("DFARS") 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at

https://www.keysight.com/find/sw_eula The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software

documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFARS 227.7103-5 (c), as applicable in any technical data.

Safety Notices

CAUTION

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a CAUTION notice until the indicated conditions are fully understood and met.

WARNING

A **WARNING** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

Where to Find the Latest Information

Documentation is updated periodically. For the latest information about these products, including instrument software upgrades, application information, and product information, browse to one of the following URLs, according to the name of your product:

<https://www.keysight.com/us/en/product/DS1030A/software-crypto-training-target.html>

<https://www.keysight.com/us/en/product/DS1031A/hardware-crypto-training-target.html>

To receive the latest updates by email, subscribe to Keysight Email Updates at the following URL:

<https://support.keysight.com>

Information on preventing instrument damage can be found at:

<https://www.keysight.com/find/PreventingInstrumentDamage>

Is your product software up-to-date?

Periodically, Keysight releases software updates to fix known defects and incorporate product enhancements. To search for software updates for your product, go to the Keysight Technical Support website at:

<https://www.keysight.com/find/techsupport>

Product and Solution Cybersecurity

Keysight complies with multinational regulations for the cybersecurity of its own products and is committed to providing information to assist you in protecting your products and solutions from external cyber threats. For more information, see:

<https://www.keysight.com/us/en/about/quality-and-security/security/product-and-solution-cyber-security.html>

Keysight also recommends that you secure your IT environments using appropriate third-party tools. For instruments that run the Microsoft Windows operating system, Keysight concurs with Microsoft's recommendations for ensuring that the instrument is protected:

- Get the latest critical Windows updates
- For network-connected instruments, use an Internet firewall (in Keysight instruments, Windows Firewalls enabled by default)
- For network-connected instruments, use up-to-date antivirus and anti-spyware software

Responsible Disclosure Program

Keysight recommends that security researchers share the details of any suspected vulnerabilities across any asset owned, controlled, or operated by Keysight (or that would reasonably impact the security of Keysight and our users) using this form:

<https://www.keysight.com/us/en/contact/responsible-disclosure-program.html>

Report a Product Cybersecurity Issue

If you discover a cybersecurity issue that you suspect may involve Keysight's proprietary software, or third-party software supplied by Keysight as part of a product, or that may affect the operation of Keysight products, we encourage you to report it to us using this form:

<https://www.keysight.com/us/en/about/quality-and-security/security/product-and-solution-cyber-security.html>

Contents

Document Revisions	7
What's in the Box?	8
Hardware Description of the Hardware Crypto Training Target	9
Software Description of the Software Crypto Training Target and Default Program Flow.....	13
Default Settings for Crypto Training Target	15
Basic Setup for Crypto Training Target – UART Communication	17
Please follow these steps:	17
Verifying proper configuration	19
Basic Debug Setup for Crypto Training Target – Serial Over USB Communication	24
Initial steps:	24
Verifying proper configuration	26
Crypto Training Target Commands	30
Default command syntax and trigger behavior.....	30
Crypto Training Target common command list	31
Crypto Training Target command description	34
How to Develop Code for the Crypto Training Target.....	73
Before starting ColDE (do this only once in new computers)	73
How to modify the Crypto Training Target code with ColDE.....	74
How to Generate a DFU File for Updating Crypto Training Target Firmware Over USB	80
Requirements	80
Steps	80
How to Flash the Crypto Training Target via USB with a DFU Binary File.....	83
Requirements	83
Steps	83
Troubleshooting and Frequently Asked Questions.....	87
The board is not responding to commands	87
The board is not powering up (both LEDs are off)	87
My power traces look very different from the manual. Why?	87
My board works but the oscilloscope does not trigger with the signal from the PC2 pin	88
How can I find out if my Crypto Training Target has a hardware crypto engine?	88
My Crypto Training Target replies “BadCmd” to some commands. Why?	89




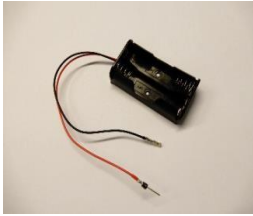

My question is not answered in the manual	89
---	----

Document Revisions

Version	Date	Notes
2.3.2	2020-10-20	Added Box content checklist
2.3.1	2018-11-09	Fixed extra 'F' typo in SPI payload description of command 0xCE
2.3	2018-05-09	Added ECC25519 scalar multiplication command
2.2	2017-12-21	Added two SM4 cipher implementations & misaligned DES command
2.1	2017-11-23	Added more FAQs & updated some pictures
2.0	2017-10-30	Manual update corresponding to Piñata board code v2.1
1.0	2015-02-27	Initial revision of the document
1.0a	2015-03-02	Minor revision of the text
1.0b	2015-03-06	Improved images

What's in the Box?

The box contains the Crypto Training Target and all accessories to connect it to a computer.

Quantity ¹	Description	Photo	Identifier ²
1	Crypto Training Target		
1	Communication cable: USB-A - mini-USB		USB
1	Communication cable: USB-A to 8-pin header, with FTDI converter chip to RS232 signals		FTDI
-	Battery holder: — For 2x AA battery's — Male - Female header pin connections		
1	Header cable: 10x Female - Female		
-	This "DS1030A-31A Software and Hardware Crypto Training Target User Manual v2.3"		

1. The number of items registered on shipment list.

2. Identifier used in this document to refer to the item.

Hardware Description of the Hardware Crypto Training Target

The Hardware Crypto Training Target is a development board based on an ARM Cortex-M4F core working at a 168MHz clock speed. The board has been physically modified and programmed to be a training target for passive side channel analysis (SCA) attacks such as differential power analysis (DPA) or active fault injection attacks such as differential fault analysis (DFA). Additionally, the source code and IDE is provided to allow users to extend the functionality of the board and use it as a development/prototyping board.

The main features of the STM32F407IG/STM32F417IG CPU in the board are the following:

- 1Mbyte internal FLASH memory, 196 Kbytes internal SRAM. Firmware updateable.
- Floating point unit for single-precision arithmetic.
- I/O interfaces: 6x UART, 3x SPI, 3x I2C, 2x CAN, serial over USB, Ethernet, 140x GPIO pins.
- External memory controller for NAND/NOR FLASH, SRAM, PSRAM, CF.
- True Random Number Generator (TRNG).
- Hardware crypto engine for DES, TDES, AES, hashing (MD5, SHA1), and HMAC. This option is only available in the hardware-crypto enabled model of the board.

The development board has two rows of pin headers that expose all the 176 pins of the IC, plus some additional pins for power input. In the following image you can find the most important sections.

Figure 1 **Hardware Crypto Training Target**



3.3V pin

You can use this pin for inputting 3.3V directly to the chip (bypassing the onboard voltage regulator). You need to use this pin for performing VCC Fault Injection or SCA on hardware crypto implementations.

CAUTION

Do not connect a USB cable or 5V power supply if you use this pin to power the board.

5V pin

You can use this pin for inputting 5V to the board and use the onboard voltage regulator instead of the MiniUSB connector.

CAUTION

Do not connect a USB cable or 3.3V power supply if you use this pin to power the board.

Trigger pin PC2

You can use this pin as a trigger input for your analysis (e.g. start of the cryptographic operation for DPA).

IO selection jumper

Place a jumper between PA9 and VBUS for enabling the Serial over USB I/O interface (also known as Virtual COM port or VCP). If the jumper is placed, the UART serial port is disabled. Remove the jumper between PA9 and VBUS for enabling the UART serial port (available via the JTAG Port connector pins) and disable the Serial over USB I/O interface. Note: it is mandatory to use the UART communication interface for Fault Injection attacks, as well as DPA on hardware crypto implementations.

MiniUSB connector

You can use this pin for providing power to the board, as well for communication (if the board is set to communicate via USB instead of the UART interface).

CAUTION

Do not power the board with a 3.3V or 5V power supply if you use this connector to power the board.

Power input selector

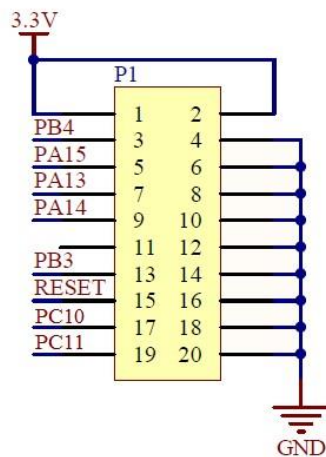
Set this switch to USB if you intend to power the board with a USB cable. Set this switch to 5Vin if you intend to power the board via the 5V pin (5V VCC, uses onboard voltage regulator) or the 3.3V pin (3.3V VCC, bypasses onboard voltage regulator).

Boot config selector

Set this pin to FLASH for booting the normal code. Set this pin to SYSTEM only if you intend to reprogram the board.

JTAG port

This port exposes the JTAG/SWD interface and a serial interface (UART). Pin 1 of the port is the upper-left pin, as shown in the board image. The description of the physical pinout can be found in the following diagram:



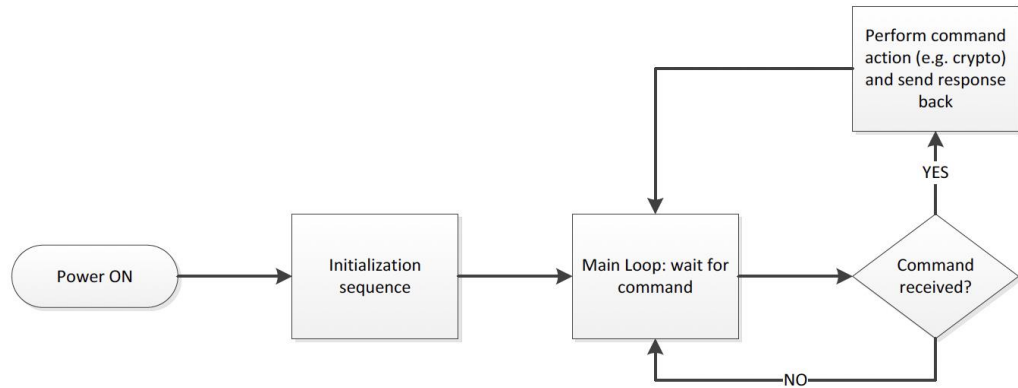
In the default board configuration, PC10 is the TX pin of the serial interface and PC11 is the RX pin of the serial interface (highlighted in a blue circle in Figure 1). You can use any of the GND pins for the serial interface connectors. There is also one jumper between pin 1 and pin 2 by default. This jumper prevents an accidental short-circuit between the 3.3V line and the adjacent GND pins.

Reset Button

This button reboots the board once pressed. This means that the board repeats the initial configuration sequence and waits for a command. It is mandatory to press the Reset button if you change the boot mode selector between the normal board program mode (FLASH) and the board programming mode (SYSTEM). It is also mandatory to press the RESET button if you want to switch between I/O interfaces after placing/removing the IO configuration jumper.

Software Description of the Software Crypto Training Target and Default Program Flow

The following state diagram describes the different states in the Software Crypto Training Target default main program.



A detailed description of each state follows:

Power ON

The board enters this state from power off when it has power in the 5V inputs (MiniUSB or 5V pin) or in the 3.3V input pin. This will cause the PWR led is ON, and depending on the power input selector the VBUS LED may be also ON.

Initialization sequence

1. The board will set up all the internal components in the following order: system clock configuration (board switches from 16MHz to 168MHz clock speed), I/O configuration (UART/USB depending on jumper between PA9 and VBUS), and peripherals (e.g. OLED screen).
2. After this configuration, the board loads the default keys in the different cipher implementations.
3. After this key loading, there is a loop with an incremental counter that can be used as a target for boot glitching. The loop length is defined by the variable `bootLoopCount`, and you can use the pin PC1 as a trigger pin (there is a 3.3V pulse as long as the loop duration). If the loop instruction count is incorrect, the board sets a global variable `glitchedBoot` to 1.
4. If the OLED screen is attached to the board (optional peripheral), it displays a message "Boot check ok". If the variable `glitchedBoot` is 1, then the OLED screen displays the message "Boot check glitched!!"

- 5. After all these steps, the board finishes init and enters the main loop waiting for commands.**

Main loop

The board is waiting for a command from the I/O interface. The full command description can be found in the section “Piñata board command list” (now Crypto Training Target). If the command sent to the board is incorrect, the board will send as a response the string “BadCmd\n\00” (8 bytes). Note that in every iteration of the main loop the I/O buffers are zeroed before waiting for commands.

Perform command action

The board will perform the requested action (e.g. perform DES encryption). If the operation is a cryptographic operation, the board will set to 3.3V the PC2 trigger pin before the operation starts and to 0V once the operation ends. After the action finishes, the board will send the results (if any) via the I/O interface (UART or USB) and return to the main loop.

Default Settings for Crypto Training Target

Keysight provides each Crypto Training Target with the following default configuration:

- Internal FLASH memory is pre-programmed with the default board code (provided with the board). This code contains the following commands (not exhaustive list):
 - Boot-time loop counter.
 - Software DES, AES, and RSA implementations (with and without countermeasures).
 - Loop counter with up-counter and down-counter.
 - Password check commands for FI purposes.
 - Software DES and AES with double checks for advanced FI purposes.
 - Administration commands e.g. for modifying the clockspeed or querying the sw version.

Additionally, in the hardware-crypto enabled models, these commands are available:

- Hardware DES implementation (32-bit implementation)
- Hardware TDES implementation (32-bit implementation)
- Hardware AES implementation (32-bit implementation)
- Hardware SHA1 and HMAC-SHA1 implementation
- Communication interface is set to the Serial over USB interface (there is a jumper between PA9 and VBUS) by default. Note that it is needed to install the Virtual COM port drivers from ST for communication via this interface. This I/O interface is only recommended for EM analysis.
- UART pins have been rewired to the JTAG port connector. This is the recommended I/O interface for Side Channel Analysis attacks. Additionally, this is the mandatory I/O interface for Fault Injection attacks.
- Decoupling capacitors have been removed for enabling Side Channel Analysis and Fault Injection attacks on the board.

- There is a jumper between pin 1 and pin 2 of the JTAG port to avoid accidental short-circuits between 3.3V and GND in the JTAG port connector.
- The Power input selector is set to USB and the Boot config selector is set to FLASH.

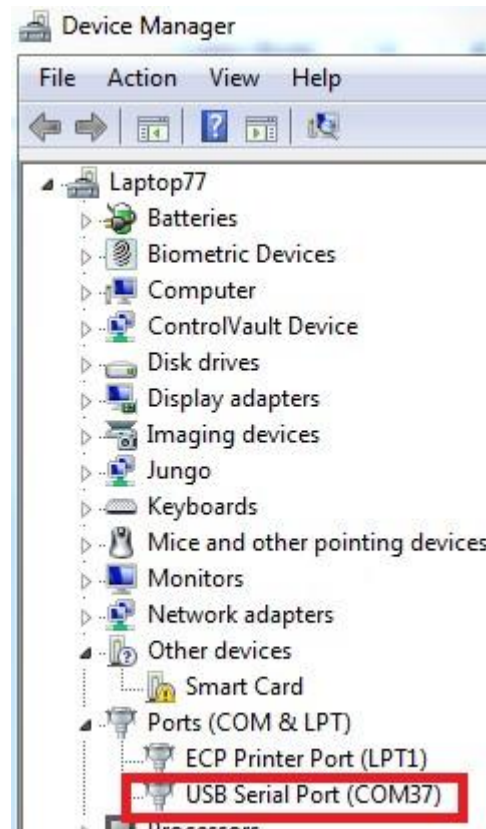
Basic Setup for Crypto Training Target – UART Communication

WARNING

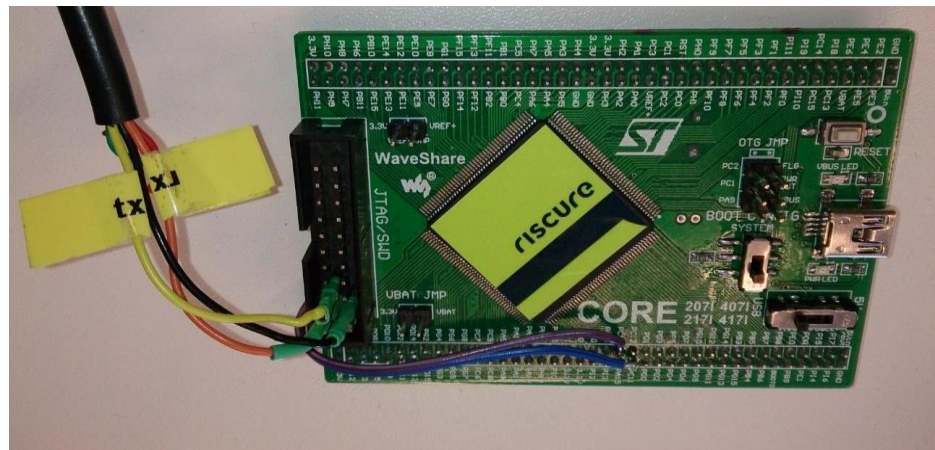
Make sure there is no jumper between PA9 and VBUS and switch Boot Config is set to FLASH!

Please follow these steps:

1. Make sure that the board is not connected to any cable. Then, verify that the board is configured as follows (default configuration):
 - No jumper between PA9 and VBUS (IO set to UART).
 - Boot config selector switch set to FLASH.
 - Power input selector switch set to USB.
2. If you have not already installed the FTDI cable drivers from the provided software package, please install them. To do so:
 - Locate the driver installer found in the software package, folder “FTDI cable Driver”:
 - Windows 32bit & 64bit systems: CDM v2.12.00 WHQL Certified.exe
 - Run the installer with Administrator rights and proceed with the driver installation until the installation is finished successfully.
3. Connect the provided FTDI cable to the PC. Windows will find a new device and will install the drivers for the cable. Reboot the computer if prompted by the drivers. The FTDI cable should be listed in the Ports (COM & LPT) section of the device manager. Remember the COM port number, as you will need it in a later step.



4. Connect the FTDI cable to the Software Crypto Training Target. The required connections are the following:
- FTDI Yellow cable > Pin PC10 in JTAG Port connector (TX)
 - FTDI Orange cable > Pin PC11 in JTAG Port connector (RX)
 - FTDI Black cable > Any ground pin in JTAG Port connector (GND)
 - The following image shows the cables connected to the board:



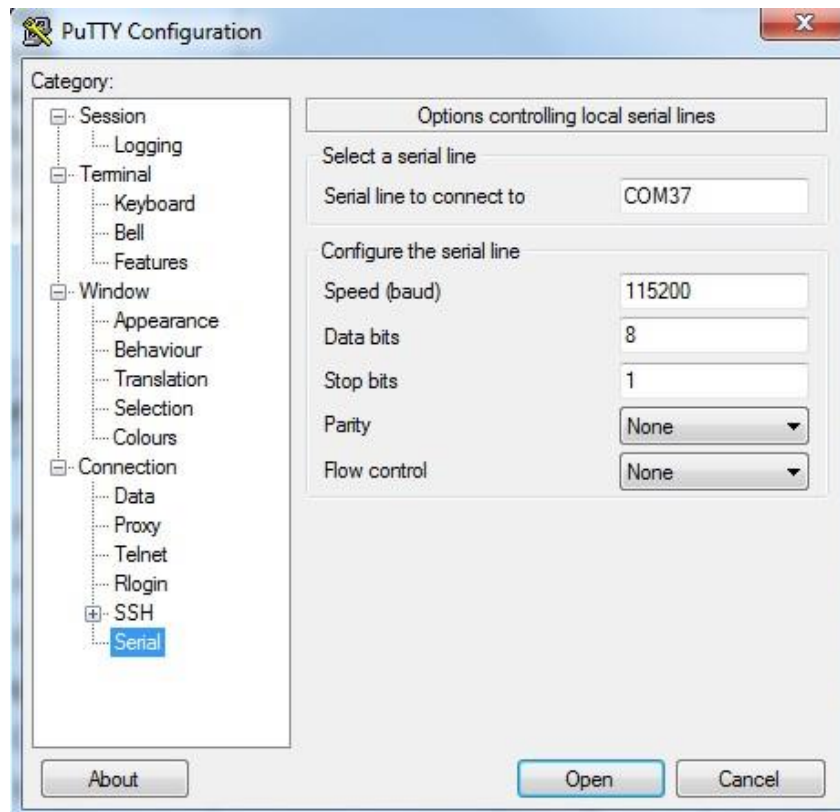
5. Connect the MiniUSB cable. Verify that both LEDs are ON. Check in the Windows Device Manager in the *Ports (COM & LPT)* section that there is no STMicroelectronics Virtual Com Port device. If this device is listed, remove the jumper in the Board between PA9 and VBUS (if any) and press the RESET button.

Verifying proper configuration

For verifying the proper configuration of your computer and the board, you have two options:


Without Inspector SCA/FI software:

1. Open a serial connection program (e.g. PuTTY) with the following settings:
 - COM port: the one found in the device manager (e.g. COM37).
 - Baud rate: 115200 bps.
 - Data bits: 8.
 - Stop bits: 1.
 - Parity: None, Flow control: None.



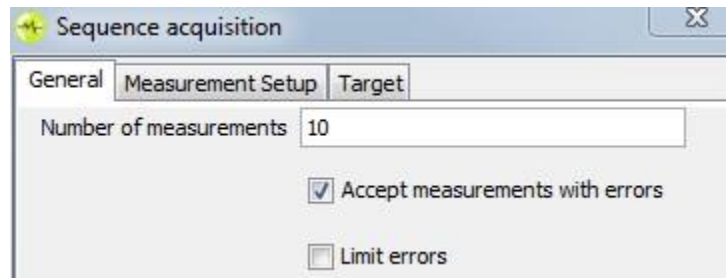
2. Press the spacebar for sending an ASCII character 0x20. The Crypto Training Target should reply the string “BadCmd” followed by a new line character (0x 0A) and a null character (0x 00). In total the board sends 8 bytes.

With Inspector SCA/FI software (version 4.8.2 or higher required):

3. Copy the provided Piñata board .java modules from the “Inspector Sequences” folder in the provided software package into your Inspector user module folder (e.g. c:\users\myUser\Inspector\modules). These files contain Inspector Sequences for communication with the board.
4. Open the user module PinataDESEncrypt.java by clicking in the menu File >> Open user module...
5. Compile and load the module in Inspector by pressing F9 or by clicking in the “Compile and load module”  button in the tool bar. After compilation, the message “Compilation succeeded” should appear in the Log.
6. We will perform an acquisition with Inspector by opening the module Acquisition >> Scope – Sequence. Configure the module as follows:

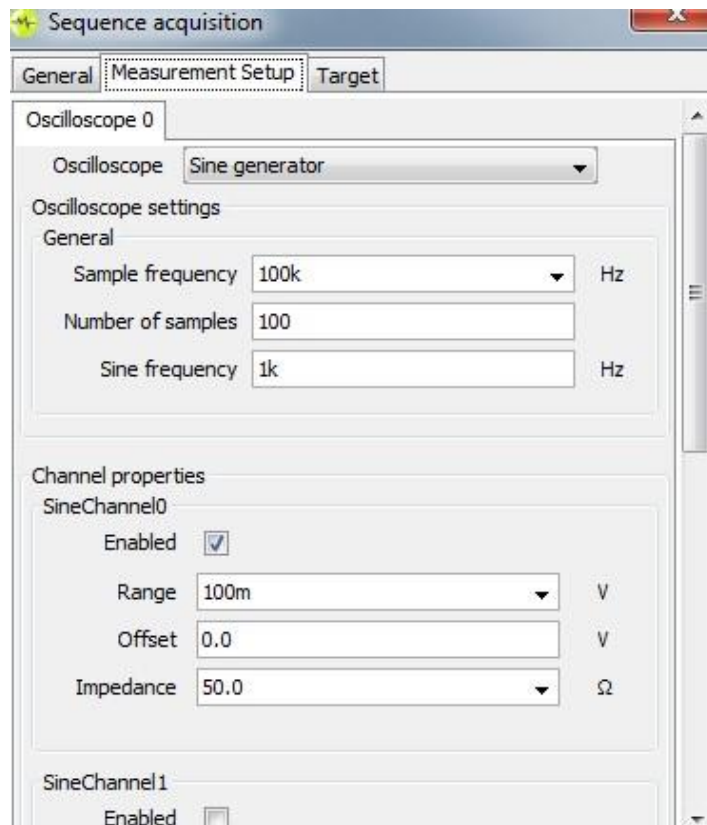
General tab

- Number of measurements: 10.
- Accept measurements with errors: enabled.
- Limit errors: disabled.



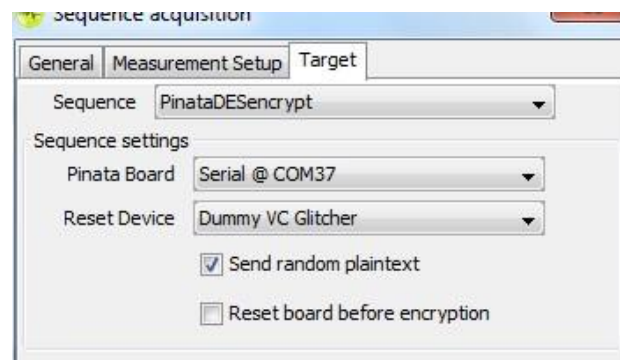
Measurement setup tab

- Oscilloscope: select the Sine generator.
- Enable only one SineChannel.
- Rest of settings left unchanged.

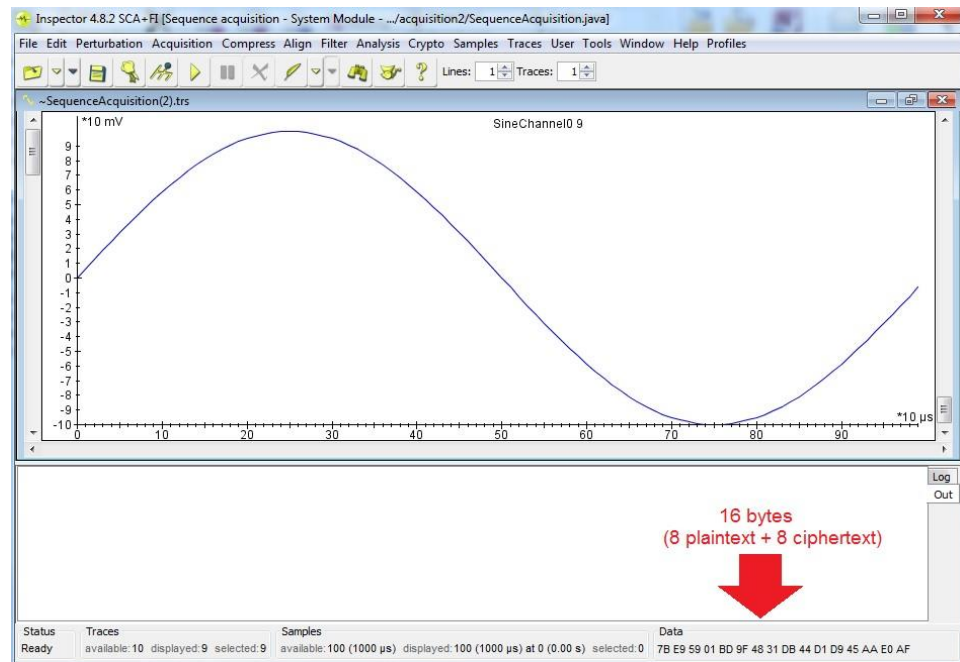


Target tab

- Sequence: select “PinataDESEncrypt”.
- Piñata Board: select the same COM port as shown in the Windows Device Manager (e.g. COM37).
- Reset Line Device (if displayed): select Dummy VC glitcher.
- Send random plaintext (if displayed): enabled.
- Reset board before encryption (if displayed): disabled.



7. Start the acquisition module. In the Data section of the status bar, each acquired trace should display 16 bytes of data. These bytes correspond to 8 bytes input data (plaintext), and the associated 8 bytes output data (ciphertext) encrypted with the default DES key.



If you have successfully completed one of the two steps in this “Verifying a proper configuration” section, your Crypto Training Target is properly configured with your system and working properly. If you experience any problem during this initial configuration, please go to the Troubleshooting section of the manual.

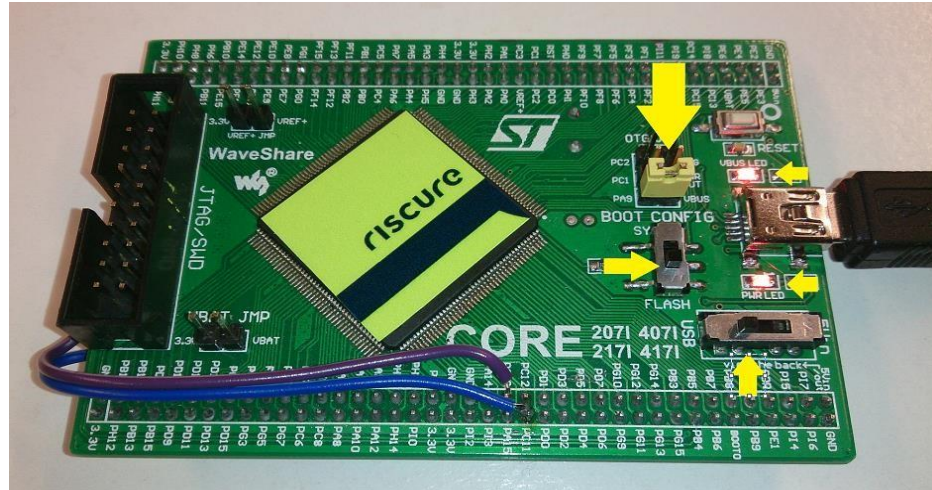
Basic Debug Setup for Crypto Training Target – Serial Over USB Communication

CAUTION

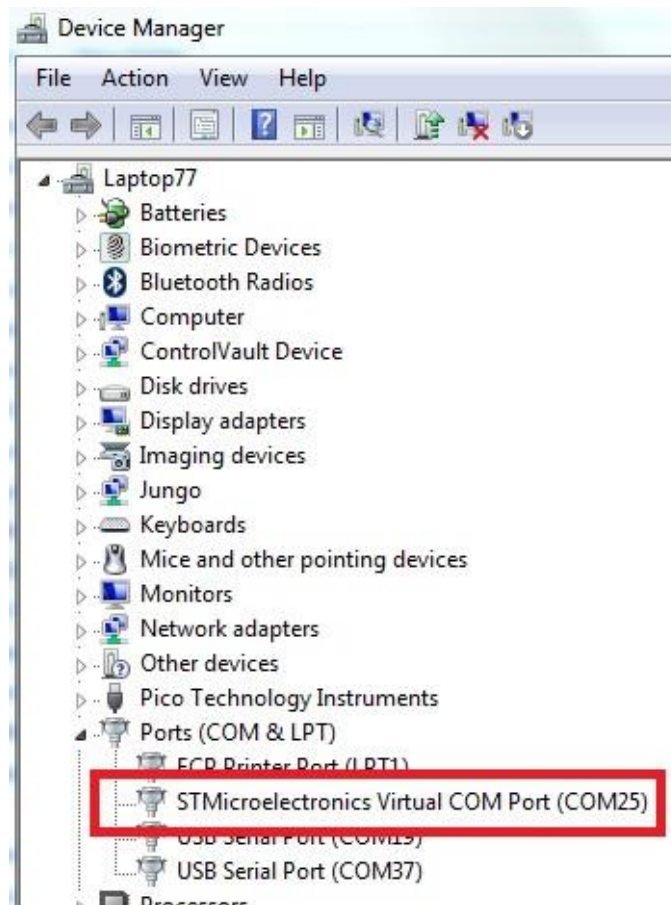
Important: this I/O interface is not recommended for performing SCA attacks. Additionally, this I/O interface should NOT be used for performing FI attacks. It is recommended only for testing/debugging.

Initial steps:

1. Make sure that the board is not connected to any cable. Then, verify that the board is configured as follows (default configuration):
 - Jumper placed between PA9 and VBUS (IO set to Serial over USB).
 - Boot config selector switch set to FLASH.
 - Power input selector switch set to USB.
2. If you have not already installed the Virtual COM port drivers from the provided software package, please install them. To do so:
 - Locate the driver installer found in the software package, folder USB Virtual Com Port Driver:
 - Windows 32bit systems: dpinst_x86.exe.
 - Windows 64bit systems: dpinst_amd64.exe.
 - Run the installer with Administrator rights and proceed with the driver installation until the installation is finished successfully.
3. Connect the provided MiniUSB cable to the PC and the Crypto Training Target. Windows will find the Crypto Training Target and will start installing the new device drivers. After connecting the USB cable, the board LEDs and switches should look similar to the picture below (make sure that the components indicated by the arrow are in the same configuration).



4. Once the device has been configured successfully in Windows, open the Device Manager with Administrator rights. If the board is properly configured, it should be found in the Ports (COM & LPT) category as shown in the following picture (highlighted in red). Remember the COM port number, as you will need it later.



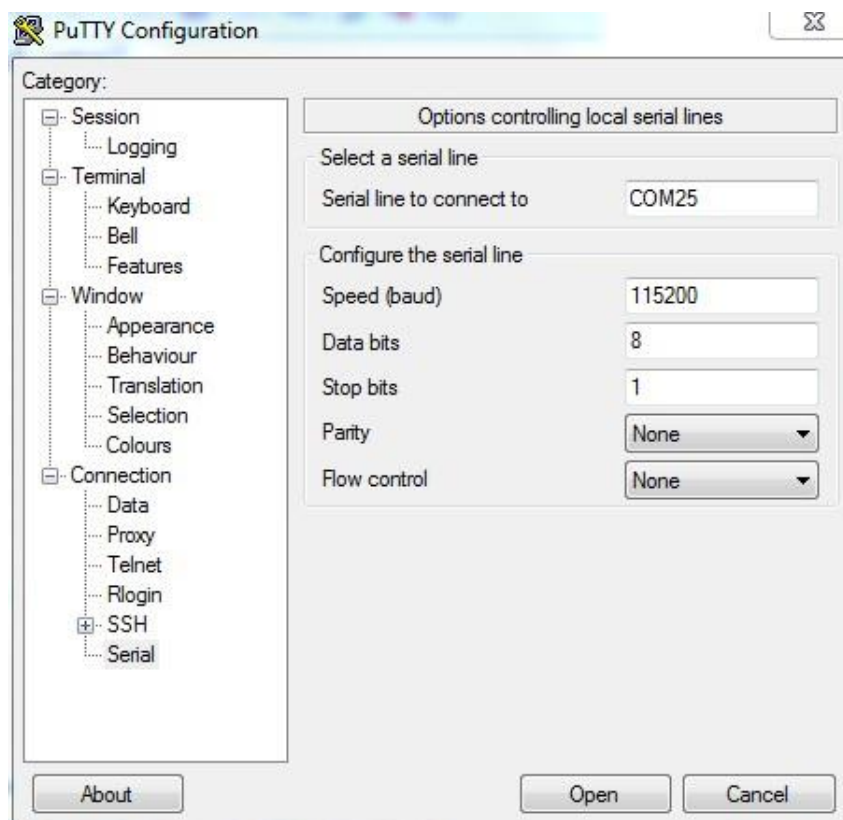
Verifying proper configuration

For verifying the proper configuration of your computer and the board, you have two options:

Without Inspector SCA/FI software:


1. Open a serial connection program (e.g. PuTTY) with the following settings:

- COM port: the one found in the device manager (e.g. COM25).
- Baud rate: 115200 bps.
- Data bits: 8.
- Stop bits: 1.
- Parity: None, Flow control: None.



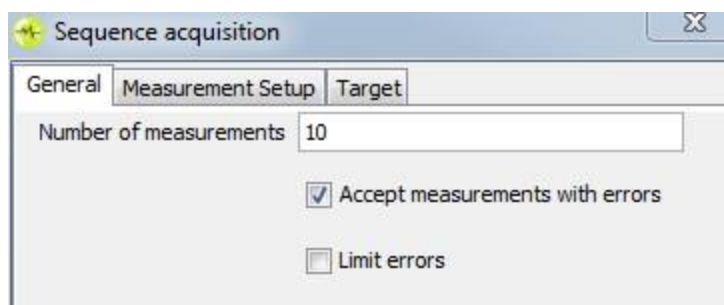
- 2. Press the spacebar for sending an ASCII character 0x20. The Piñata board should reply the string “BadCmd” followed by a new line character (0x 0A) and a null character (0x 00). In total the board sends 8 bytes.**

With Inspector SCA/FI software (version 4.8.2 or higher required):

1. Copy the provided Piñata board .java modules from the “Inspector Sequences” folder in the provided software package into your Inspector user module folder (e.g. c:\users\myUser\Inspector\modules). These files contain Inspector *Sequences* for communication with the board.
2. Open the user module PinataDESEncrypt.java by clicking in the menu *File* > > *Open user module...*
3. Compile and load the module in Inspector by pressing F9 or by clicking in the “Compile and load module”  button in the tool bar. After compilation, the message “Compilation succeeded” should appear in the Log.
4. We will perform an acquisition with Inspector by opening the module *Acquisition* > > *Scope – Sequence*. Configure the module as follows:

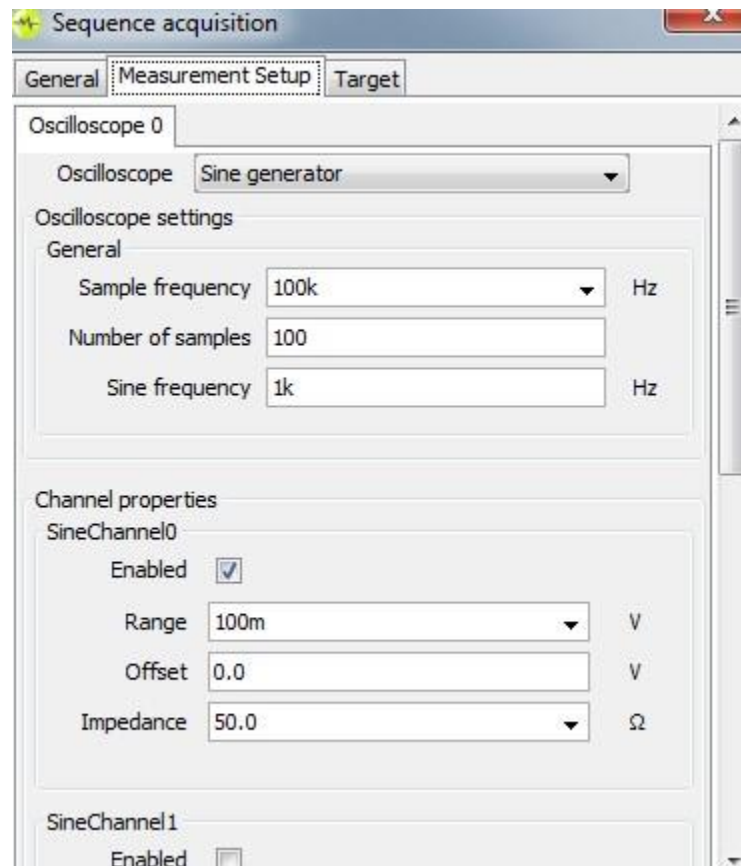
General tab

- Number of measurements: 10.
- Accept measurements with errors: enabled.
- Limit errors: disabled.



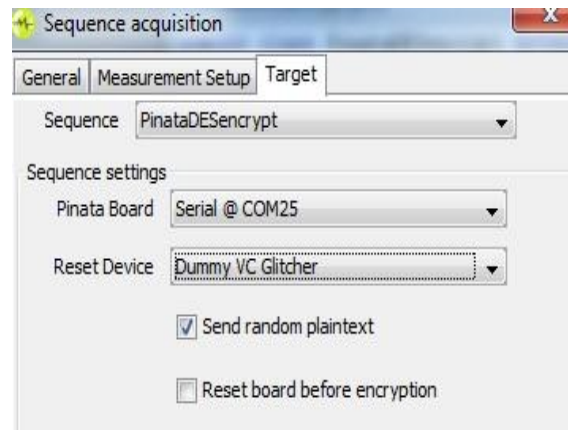
Measurement setup tab

- Oscilloscope: select the Sine generator.
- Enable only one SineChannel.
- Rest of settings left unchanged.

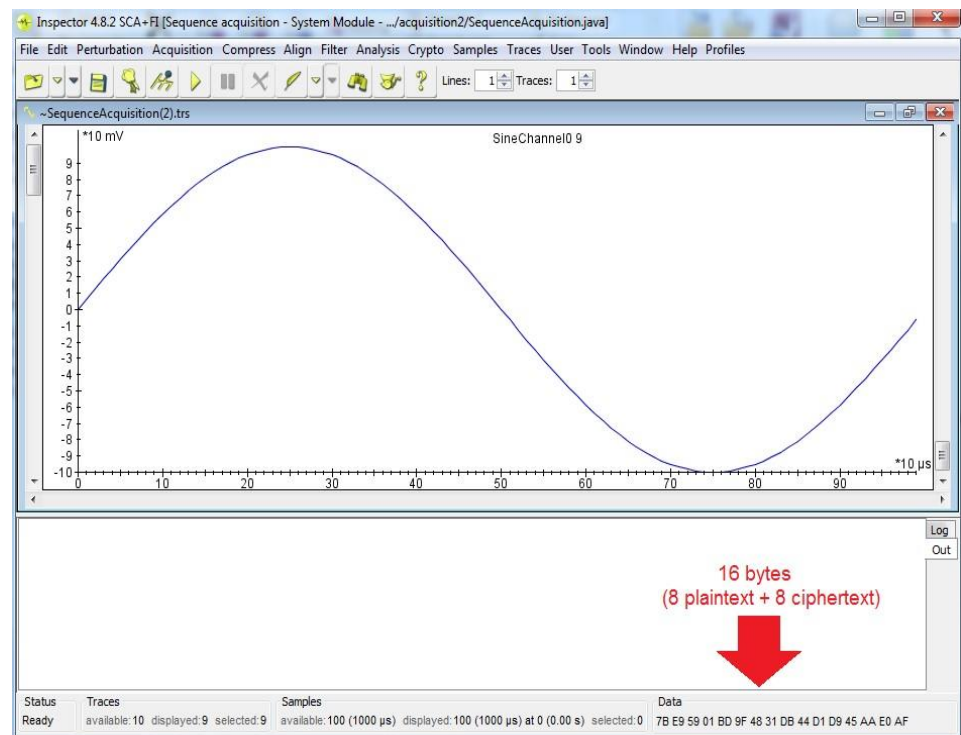


Target tab

- Sequence: select “PinataDESencrypt”.
- Piñata Board: select the same COM port as shown in the Windows Device Manager (e.g. COM25).
- Reset Line Device: select Dummy VC glitcher
- Send random plaintext: enabled.
- Reset board before encryption: disabled.



5. Start the acquisition module. In the Data section of the status bar, each acquired trace should display 16 bytes of data. These bytes correspond to 8 bytes input data (plaintext), and the associated 8 bytes output data (ciphertext) encrypted with the default DES key.



If you have successfully completed one of the two steps in this “Verifying a proper configuration” section, your Crypto Training Target is properly configured with your system and working properly. If you experience any problem during this initial configuration, please go to the Troubleshooting section of the manual

Crypto Training Target Commands

Default command syntax and trigger behavior

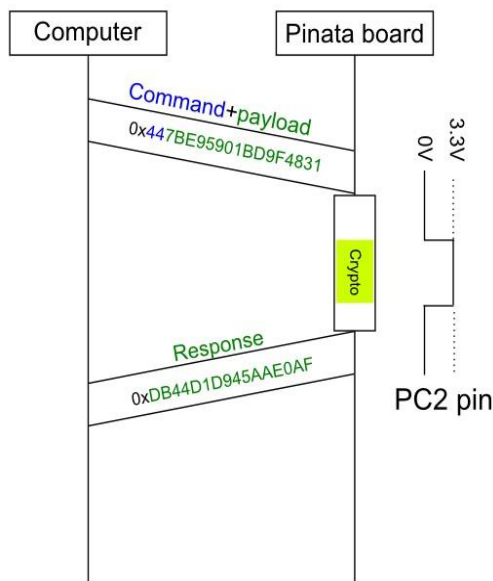
In Crypto Training Target all operations are started by sending one Command byte followed by an optional payload via the active I/O interface (Serial over USB or UART). If the requested command generates some output, it will be sent back via the same I/O interface upon command completion. The following diagram illustrates the command syntax in the case of a software DES encryption.

Pinata command syntax

Command byte	Payload (0...n bytes)
4 4	7B E9 59 01 BD 9F 48 31

A trigger signal is embedded during the command execution. It is a 3.3V TTL positive pulse that starts just immediately before the crypto operation and ends just after the end of the crypto operation. The trigger signal is available by default in pin PC2. Note that some commands have other pins or no trigger.

The following time sequence diagram describes the execution of a DES encryption of the plaintext 0x7BE95901BD9F4831 with the key 0xCAFEBADEADBEEF (default DES key preloaded in the Crypto Training Target). The board sends back the response (ciphertext) upon command completion and waits for the next command.



Crypto Training Target common command list

The following table summarizes the most frequently used commands in the Crypto Training Target code (version 3.1). Note that if the command byte does not match any of the available commands, the board will send the string BadCmd\n\00 through the I/O interface (8 bytes). For a detailed description of a command, as well as a complete list of commands, please check the section Crypto Training Target command description.

Command byte (Hex)	Command name	Command payload length in bytes	Command response length in bytes
0x 44	Software DES – encrypt	8	8
0x 45	Software DES – decrypt	8	8
0x 46	Software TDES – encrypt	8	8
0x 47	Software TDES – decrypt	8	8
0x AE	Software AES128 – encrypt	16	16
0x EA	Software AES128 – decrypt	16	16
0x CE	Sw AES128 – encrypt without trigger	16	16
0x 60	Software AES256 – encrypt	16	16
0x 61	Software AES256 – decrypt	16	16
0x 41	Software AES T-tables – encrypt	16	16
0x 50	Software AES T-tables – decrypt	16	16
0x AA	Software RSA-CRT 1024 – decrypt	2+128	2+128
0x DF	Software RSA-512 SFM – decrypt	2+64 (typically)	2+64 (typically)

Command byte (Hex)	Command name	Command payload length in bytes	Command response length in bytes
0x 90	Software Dilithium3 NIST Round 3 get mode	0	1
0x 91	Software Dilithium3 NIST Round 3 set public private key pair	1952 + 4016	0
0x 92	Software Dilithium3 NIST Round 3 verify	3293 + 16	1
0x 93	Software Dilithium3 NIST Round 3 sign	16	1 or 1 + 3293
0x 94	Software Dilithium3 NIST Round 3 get key sizes	0	2 + 2
0x 4A	Sw DES – encrypt with random delays	8	8
0x 4B	Sw DES – encrypt with random sbox	8	8
0x 73	Software masked AES128 – encrypt	16	16
0x 83	Software masked AES128 – decrypt	16	16
0x 75	Sw AES128 – encrypt with rnd delays	16	16
0x 85	Sw AES128 – encrypt with rnd sbox	16	16
0x 4C	Hardware HMAC-SHA1	4+20	20
0x 27	Hardware SHA1	4+16	20
0x BE	Hardware DES – encrypt	8	8

Command byte (Hex)	Command name	Command payload length in bytes	Command response length in bytes
0x EF	Hardware DES – decrypt	8	8
0x CA	Hardware AES – encrypt	16	16
0X FE	Hardware AES – decrypt	16	16
0X C0	Hardware TDES – encrypt	8	8
0X 01	Hardware TDES – decrypt	8	8
0X D7	Change DES key	8	8
0X E7	Change AES-128 key	16	16
0X C7	Change TDES key	8	8
0X DD	Loop counter	2	6
0X A2	Password check – single check	4	4
0X A7	Password check – double check	4	4
0X 29	Sw DES – encrypt with FI check	8	8
0X 88	Sw AES128 – encrypt with FI check	16	16
0X 11	Get random number from TRNG	0	4
0x 99	Infinite loop for FI	0	8 (if glitched)
0x 38	SRAM-SRAM key copy	16	16

Crypto Training Target command description

Software DES – Encrypt (0x44)

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. DES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8).

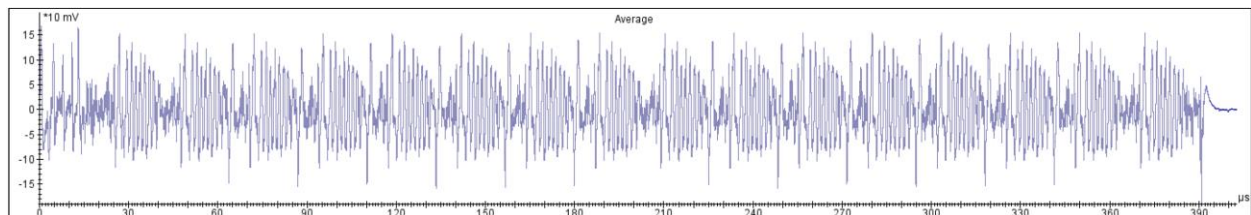
Default DES key: 0xCAFEBADEADBEEF.

Command byte: 0x44.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for Side Channel Analysis (SCA) and Fault Injection (FI) attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st and 2nd round > contained in the range [20, 66] us after rising edge in PC2 trigger signal.

DFA: 14th and 15th round > [308, 360] us after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values.

Software DES – Decrypt (0x45)

Command description: the board will perform a DES decryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. DES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8).

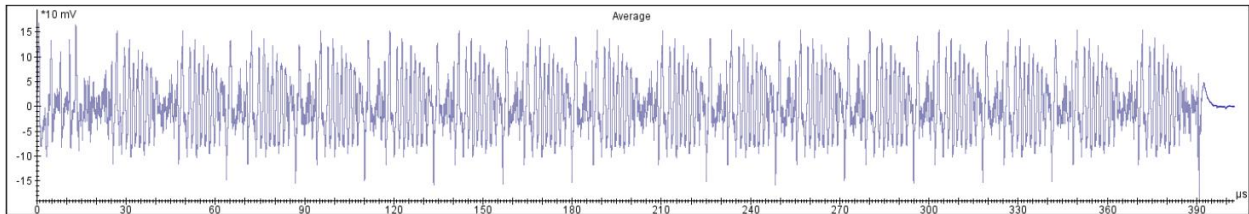
Default DES key: 0xCAFEBABEDEADBEEF.

Command byte: 0x45.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for SCA and FI attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st and 2nd round > contained in the range [20, 66] us after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values.

Software TDES – Encrypt (0x46)

Command description: the board will perform a Triple DES (TDES) encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. TDES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8). Note that default TDES key is a TDES2 key mode (key is $k_1k_2k_1$), and the DES operations are performed as follows: $\text{DESencrypt}(k_1)$ - $\text{DESdecrypt}(k_2)$ - $\text{DESencrypt}(k_1)$.

Default DES key:

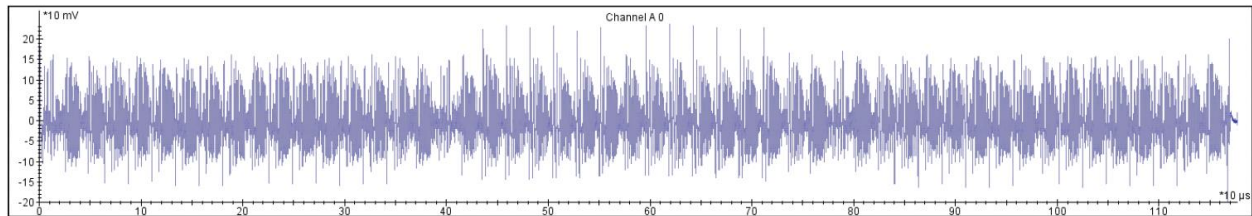
0xCAFEBABEDEADBEEF0001020304050607CAFEBABEDEADBEEF.

Command byte: 0x46.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~1.18ms.

Software TDES – Decrypt (0x47)

Command description: the board will perform a Triple DES (TDES) decryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. TDES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8). Note that default TDES key is a TDES2 key mode (key is k1k2k1), and the DES operations are performed as follows: DESdecrypt(k1)-DESencrypt(k2)-DESdecrypt(k1).

Default DES key:

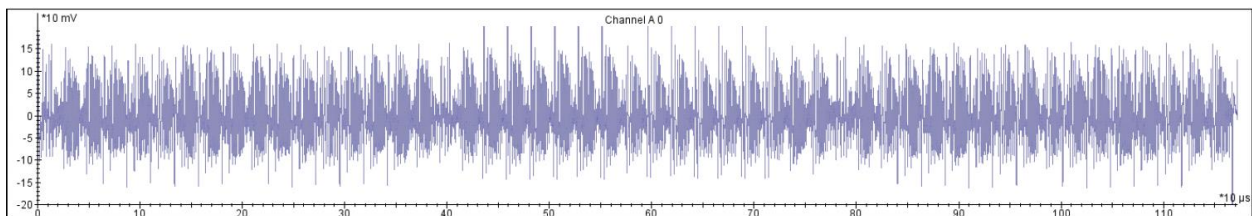
0xCAFEBADEADBEF0001020304050607CAFEBADEADBEF.

Command byte: 0x46.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~1.18ms.

Software DES – Encrypt with random delays (0x4A)

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. DES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8). As a countermeasure, a random delay is inserted before each S-box operation.

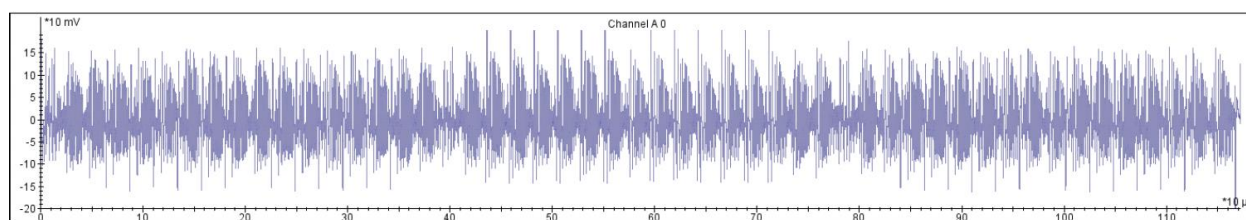
Default DES key: 0xCAFE BABE DEAD BEEF.

Command byte: 0x4A.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software DES – Encrypt with random order of S-boxes (0x4B)

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. DES implementation is performed with byte operations. As a countermeasure, the order of the S-box operations in each DES round is randomly permuted.

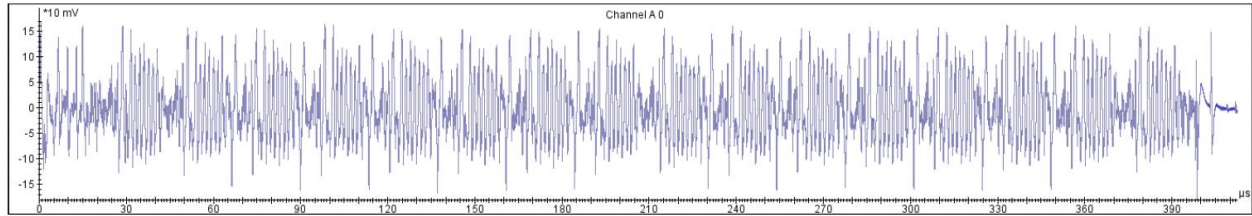
Default DES key: 0xCAFE BABE DEAD BEEF.

Command byte: 0x4B.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software DES – Encrypt with integrity check (0x29)

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes), with an integrity check to avoid Fault Injection attacks. The cipher is executed in Electronic Code Book (ECB) mode. DES implementation is performed with byte operations. DES S-boxes are implemented in order (S-box 1 to S-box 8).

As a Fault Injection countermeasure, a second execution of the DES encryption is performed, and the command will check that the output of both executions matches. If both outputs match, the output of the first execution is sent; otherwise, the board will not send any reply.

The trigger pin PC2 is active during the two consecutive DES operations; the check happens immediately after the trigger pin goes low.

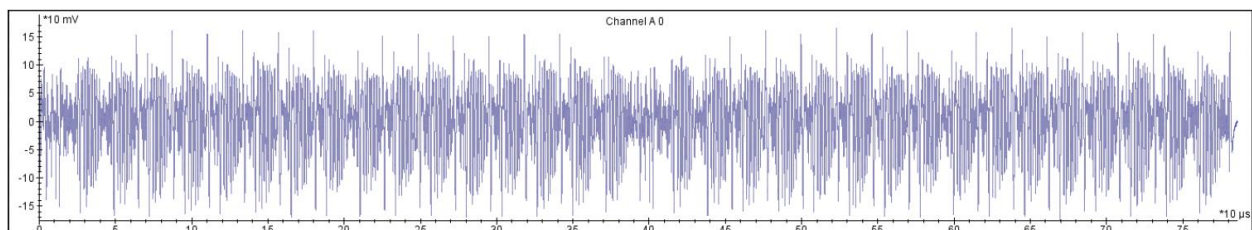
Default DES key: 0xCAFEBADEADBEEF.

Command byte: 0x29.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~785us.

Software DES with misalignment– Encrypt (0x14)

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes). The DES implementation is the same as the one used for the command byte 0x44, but has a random delay after the trigger is enabled in PC2 (before the DES operation starts). The purpose of this random delay is to force the attacker to use a Static Alignment step for performing a DPA/DEMA attack on the DES encryption. The random delay (will have variable length) is highlighted in the power trace shown below.

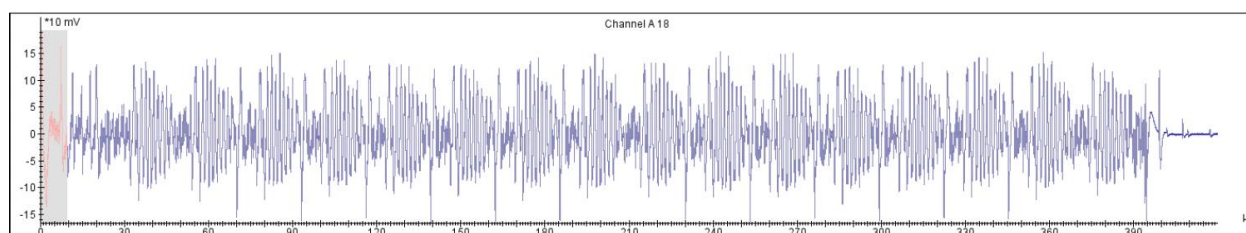
Default DES key: 0xCAFEBADEADBEEF.

Command byte: 0x14.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Software AES-128 – Encrypt (0xAE)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. Key schedule is computed before the trigger pulse in pin PC2.

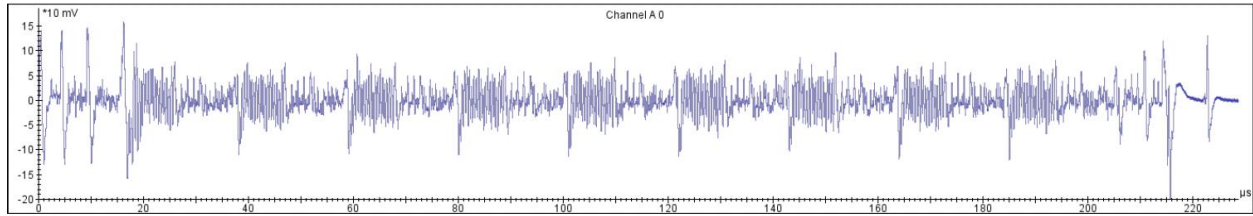
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0xAE.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for SCA and FI attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st round > contained in the range $[0, 40]$ us after rising edge in PC2 trigger signal.

DFA: 9th round > $[172, 194]$ us after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values.

Software AES-128 – Decrypt (0xEA)

Command description: the board will perform an AES128 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. Key schedule is computed before the trigger pulse in pin PC2.

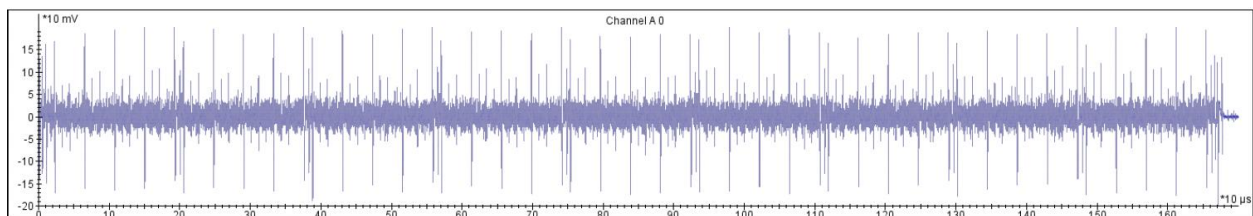
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0xEA.

Payload length: 16 bytes (ciphertext).

Response length: 16 bytes (plaintext).

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for SCA and FI attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st round > contained in the range [0, 210] us after rising edge in PC2 trigger signal.

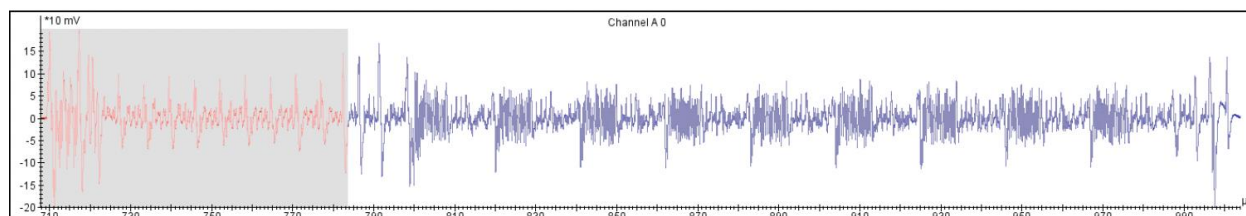
Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values Crypto execution time @168MHz: ~1.68ms.

Software AES-128– Encrypt without trigger (0xCE)

Command description: this operation is the same as the Software AES-128 Encrypt command, but without any pulse in pin PC2. Instead, the command will send the sequence of 4 bytes 0xDECAFFED via a SPI interface prior to the AES-128 encryption. You can use a SPI sniffing tool (such as Riscure Spider) to generate a trigger signal based on such communication.

The pins for the SPI interface in the Piñata board are: SCK> PB13, SS>PF4, MOSI>PB15 Note that before the actual AES computation, the key schedule of AES is computed. This is highlighted in red (with gray background) in the power trace shown below.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software AES-128 T-tables – Encrypt (0x41)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The AES implementation is optimized for 32-bit architectures, identical to the implementation present in the public crypto library OpenSSL. The implementation uses T-tables for faster execution of the cipher. Key schedule is computed before the trigger pulse in pin PC2.

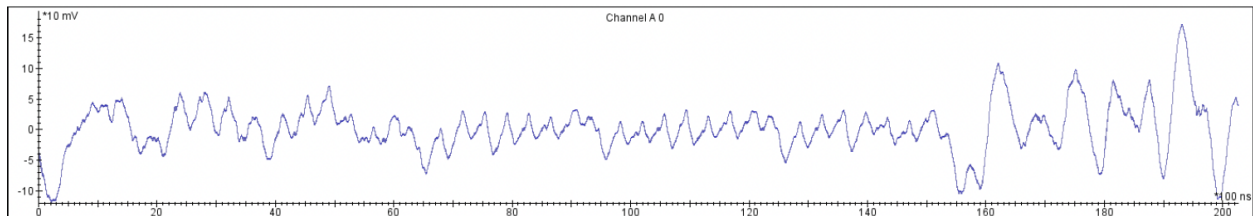
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x41.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for SCA and FI attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st round contained in the range $[0, 5]$ us after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming weight of intermediate values in a chosen-plaintext attack scenario (all input bytes constant except one) Crypto execution time @168MHz: ~20us.

Software AES-128 T-tables – Decrypt (0x50)

Command description: the board will perform an AES128 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The AES implementation is optimized for 32-bit architectures, identical to the implementation present in the public crypto library OpenSSL. The implementation uses T-tables for faster execution of the cipher. Key schedule is computed before the trigger pulse in pin PC2.

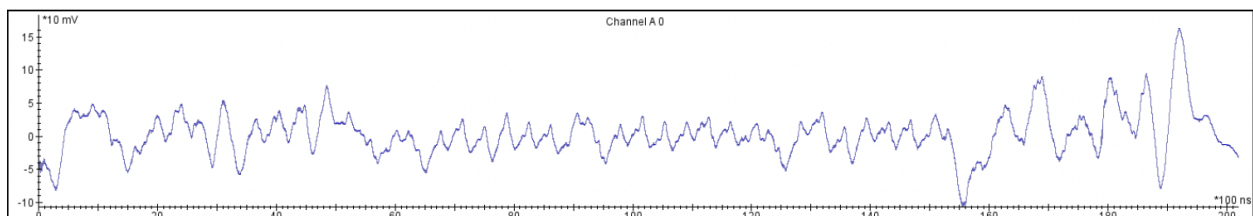
Default AES128 key: 0xCAFE BABE DEAD BEEF 0001 0203 0405 0607.

Command byte: 0x50.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Interesting time ranges for SCA and FI attacks:

($f=168\text{MHz}$, trigger signal PC2 1V threshold level, typical values).

DPA: 1st round contained in the range [0, 5] μs after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming weight of intermediate values in a chosen-plaintext attack scenario (all input bytes constant except one).

Software AES-256 – Encrypt (0x60)

Command description: the board will perform an AES256 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The implementation is ported from code from Ilya O. Levin (www.literatecode.com). It has been included due to its curious power trace. Key schedule is computed during the bootstrap of the Crypto Training Target.

Trigger pin for this function is pin PC2.

Default AES256 key:

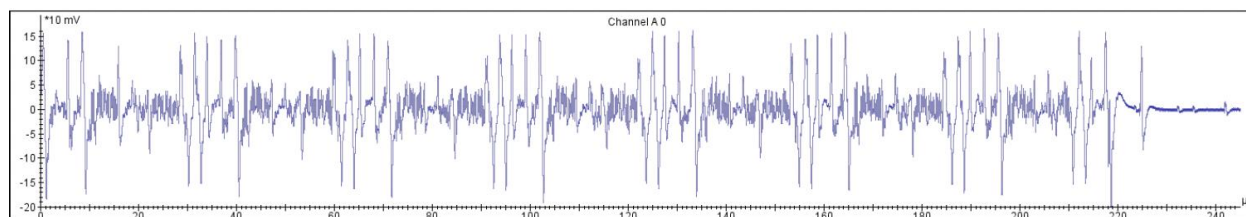
0xcafebabedeadbeef0001020304050607dabadabad00000c00001c0ffee55dead

Command byte: 0x60.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~230 μs .

Software AES-256 – Decrypt (0x61)

Command description: the board will perform an AES256 decryption of the message contained in the command payload (16 bytes). The cipher is executed

in Electronic Code Book (ECB) mode. The implementation is ported from code from Ilya O. Levin (www.literatecode.com). It has been included due to its curious power trace. Key schedule is computed during the bootstrap of the Crypto Training Target.

Trigger pin for this function is pin PC2.

Default AES256 key:

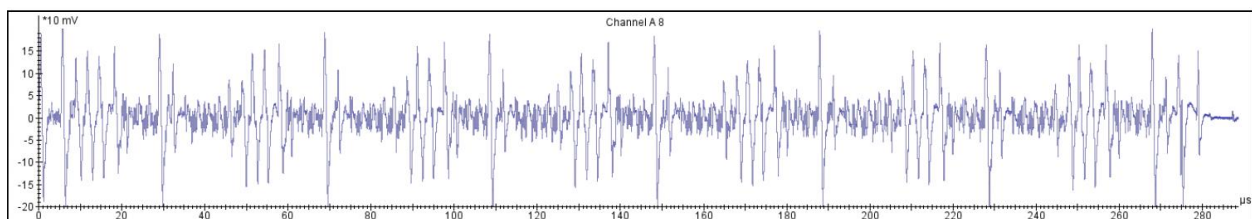
0xcafebabedeadbeef0001020304050607dabadabad00000c00001c0ffee55dead

Command byte: 0x61.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~280us.

Software AES-128 with masking – Encrypt (0x73)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. As a countermeasure, the S-boxes have been masked in a “textbook” style, such as described in the book “Power Analysis Attacks: Revealing the Secrets of Smart Cards” (ISBN 978-0-387-30857-9). The trigger generated in the PC2 pin is high during the masking process as well as the AES rounds.

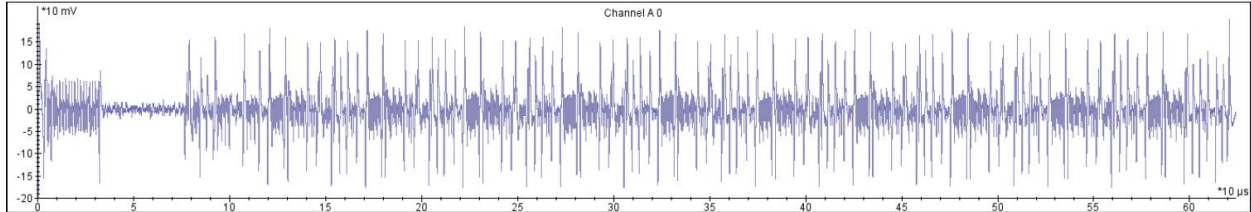
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x73.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software AES-128 with masking – Decrypt (0x83)

Command description: the board will perform an AES128 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. As a countermeasure, the S-boxes have been masked in a “textbook” style, such as described in the book “Power Analysis Attacks: Revealing the Secrets of Smart Cards” (ISBN 978-0-387-30857-9). The trigger generated in the PC2 pin is high during the masking process as well as the AES rounds.

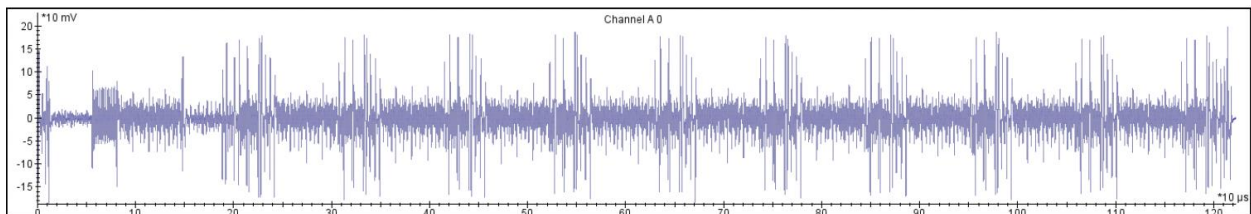
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x83.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Crypto execution time @168MHz: ~1.22ms.

Software AES-128 – Encrypt with random delays (0x75)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. As a countermeasure, a random delay is inserted before each S-box operation during the SubBytes phase in each AES round.

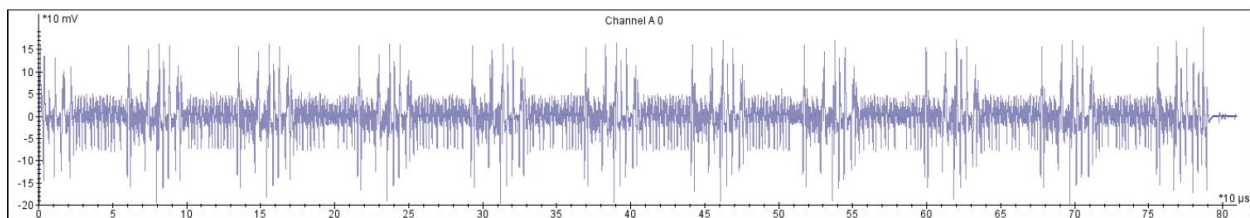
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x75.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software AES-128 – Encrypt with random ordering of S-box (0x85)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. As a countermeasure, the order of the S-box operations during the SubBytes phase in each AES round is randomly permuted.

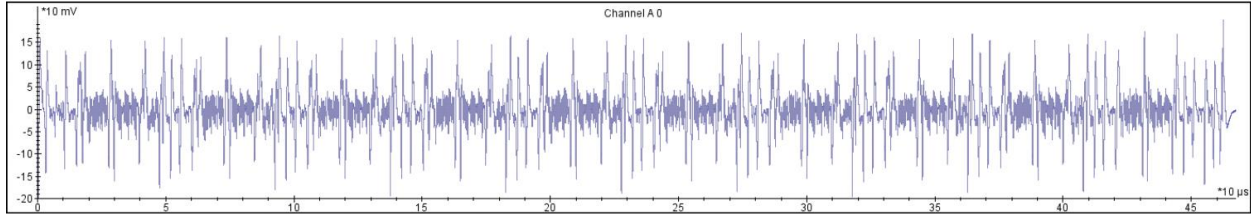
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x85.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software AES-128 with integrity check (0x88)

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes) and an integrity check to avoid Fault Injection attacks. The cipher is executed in Electronic Code Book (ECB) mode. AES implementation is performed with byte operations. As a countermeasure, the produced ciphertext is decrypted with an AES T-tables implementation. The decrypted text is compared to the input text: if the two match, the ciphertext is sent; otherwise, the board does not send any message. Note that the T-tables implementation is much faster (roughly 10x faster) than the “textbook” AES encryption.

The trigger generated in the PC2 pin is high during the first AES computation; the AES T-tables and integrity check are performed just after the trigger signal goes low.

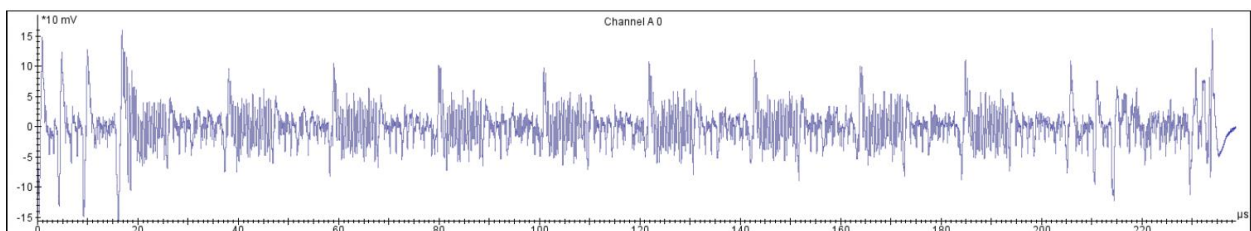
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0x88.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Software SM4 textbook implementation – Encrypt (0x54)

Command description: the board will perform a SM4 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. SM4 implementation is performed as described in the SM4 cipher standard. Key schedule is computed before the trigger pulse in pin PC2.

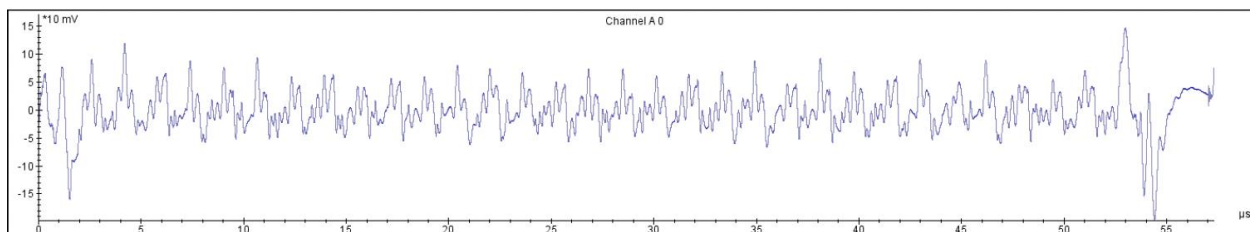
Default SM4 key: 0x526973637572654368696E6132303137.

Command byte: 0x54.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Keysight Current Probe):



Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values.

Software SM4 textbook implementation – Decrypt (0x55)

Command description: the board will perform a SM4 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. SM4 implementation is performed as described in the SM4 cipher standard. Key schedule is computed before the trigger pulse in pin PC2.

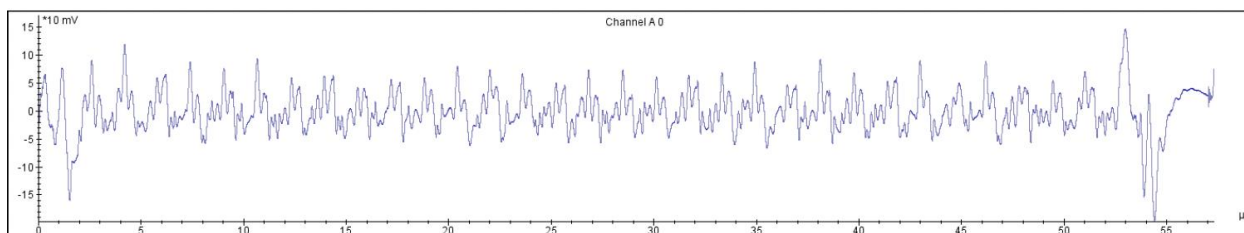
Default SM4 key: 0x526973637572654368696E6132303137.

Command byte: 0x55.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Suitable model for SCA: this implementation leaks the hamming weight of the S-Box output values.

Software SM4 OpenSSL implementation – Encrypt (0x64)

Command description: the board will perform a SM4 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. SM4 implementation is the same as the OpenSSL software SM4 implementation. Key schedule is computed before the trigger pulse in pin PC2.

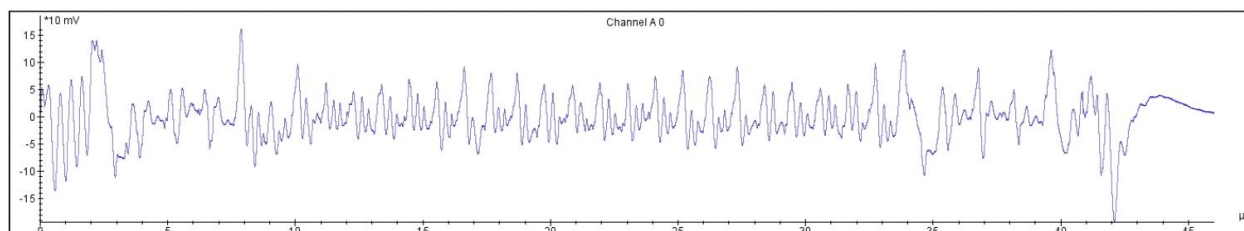
Default SM4 key: 0x526973637572654368696E6132303137.

Command byte: 0x64.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Software SM4 OpenSSL implementation – Decrypt (0x65)

Command description: the board will perform an SM4 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. SM4 implementation is the same as the OpenSSL software SM4 implementation. Key schedule is computed before the trigger pulse in pin PC2.

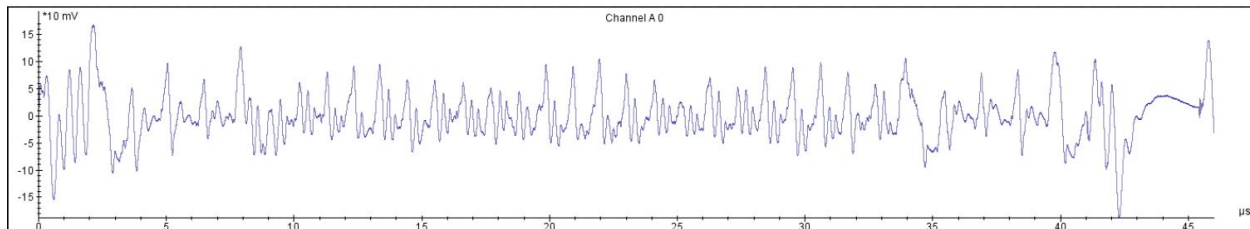
Default SM4 key: 0x526973637572654368696E6132303137.

Command byte: 0x65.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Software RSA-1024bit, CRT implementation – Decrypt (0xAA)

Command description: the board will perform an RSA decryption of the message contained in the command payload using a Chinese Remainder Theorem (CRT) implementation. The RSA-CRT implementation performs the decryption using two exponentiations ($c^{dp} \bmod p$, $c^{dq} \bmod q$). The length of the message is defined by the first two bytes of the payload (most significant byte first, 128 bytes = 0x0080). In this implementation, message length should be 128 bytes. Modular exponentiations are computed in a binary left-to-right implementation (using square or square & multiply operations). Trigger pulse in pin PC2 only bounds the modular exponentiation operations.

In parallel to the trigger signal in pin PC2, there is a trigger pulse in pin PH2 for the exponentiation mod p and a trigger pulse in pin PH3 for the exponentiation mod q.

Public key, RSA modulus n: 0x BB F8 2F 09 06 82 CE 9C 23 38 AC 2B 9D A8 71 F7 36 8D 07 EE D4 10

43 A4 40 D6 B6 F0 74 54 F5 1F B8 DF BA AF 03 5C 02 AB 61 EA 48 CE EB 6F CD 48 76 ED 52 0D 60 E1 EC 46 19 71 9D 8A 5B 8B 80 7F AF B8 E0 A3 DF C7 37 72 3E E6 B4 B7 D9 3A 25 84 EE 6A 64 9D 06 09

53 74 88 34 B2 45 45 98 39 4E E0 AA B1 2D 7B 61 A5 1F 52 7A 9A 41 F6 C1 68 7F E2 53 72 98 CA 2A

8F 59 46 F8 E5 FD 09 1D BD CB

Public key, RSA exponent e: 0x 11

Private key, RSA-CRT exponent dp: 0x 54 49 4C A6 3E BA 03 37 E4 E2 40 23 FC D6 9A 5A EB 07 DD

DC 01 83 A4 D0 AC 9B 54 B0 51 F2 B1 3E D9 49 09 75 EA B7 74 14 FF 59 C1
F7 69 2E 9A 2E 20 2B 38

FC 91 0A 47 41 74 AD C9 3C 1F 67 C9 81

Private key, RSA-CRT exponent dq: 0x 47 1E 02 90 FF 0A F0 75 03 51 B7 F8 78
86 4C A9 61 AD BD 3A

8A 7E 99 1C 5C 05 56 A9 4C 31 46 A7 F9 80 3F 8F 6F 8A E3 42 E9 31 FD 8A E4
7A 22 0D 1B 99 A4 95

84 98 07 FE 39 F9 24 5A 98 36 DA 3D

Private key, RSA-CRT coefficient qInv: 0x B0 6C 4F DA BB 63 01 19 8D 26 5B
DB AE 94 23 B3 80 F2 71

F7 34 53 88 50 93 07 7F CD 39 E2 11 9F C9 86 32 15 4F 58 83 B1 67 A9 67 BF
40 2B 4E 9E 2E 0F 96 56

E6 98 EA 36 66 ED FB 25 79 80 39 F7

Private key, RSA Prime p: 0x EE CF AE 81 B1 B9 B3 C9 08 81 0B 10 A1 B5 60
01 99 EB 9F 44 AE F4 FD

A4 93 B8 1A 9E 3D 84 F6 32 12 4E F0 23 6E 5D 1E 3B 7E 28 FA E7 AA 04 0A
2D 5B 25 21 76 45 9D 1F

39 75 41 BA 2A 58 FB 65 99

Private key, RSA Prime q: 0x C9 7F B1 F0 27 F4 53 F6 34 12 33 EA AA D1 D9
35 3F 6C 42 D0 88 66 B1

D0 5A 0F 20 35 02 8B 9D 86 98 40 B4 16 66 B4 2E 92 EA 0D A3 B4 32 04 B5
CF CE 33 52 52 4D 04 16

A5 A4 41 E7 00 AF 46 15 03

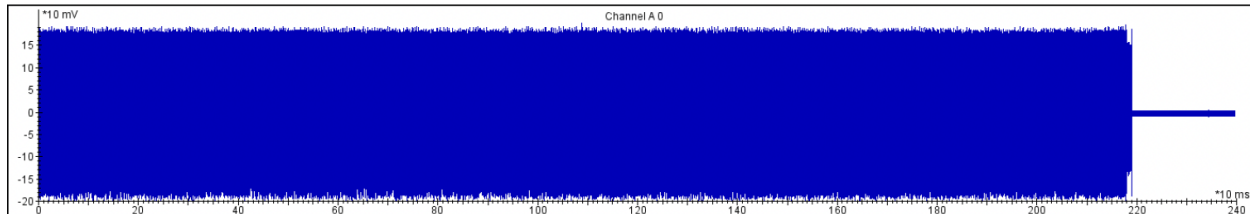
Command byte: 0xAA.

Payload length: 130 bytes (2 bytes message length with value 0x0080, 128
bytes message).

Response length: 130 bytes (2 bytes message length with value 0x0080, 128
bytes message).

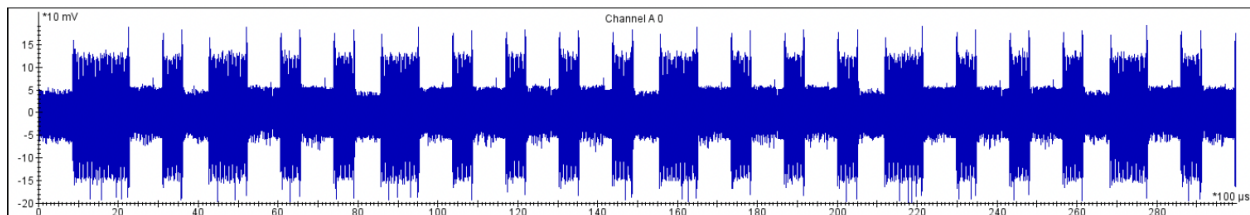
Computation time: typically, 2.18 seconds (f=168MHz).

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Power trace capture (only first 30 ms shown):



Trigger signal PC2 1V threshold level.

Suitable model for SCA: this implementation is vulnerable to Simple Power Analysis (SPA), square and multiply patterns are visible in the power measurement (square operations are shorter than multiply operations).

Software RSA-512bit SFM (straightforward method) – Decrypt (0xDF)

Command description: the board will perform an RSA decryption of the message contained in the command payload using a straightforward implementation (RSA-SFM). As a countermeasure, this implementation performs the exponentiation using a square and multiply always in the Montgomery domain. Trigger pulse in pin PC2 only bounds the decryption exponentiation operation. Note that the SFM private key (decryption exponent, d) can be changed with other command, as well as the implementation of the square and multiply operation.

Default public RSA512 modulus n:

0xEDDE30A352E66256E593EFBCA54E20853459534B71AFAB0F058A2CA582C
E15617D9592C87DF72

0C674416652C1CF90F5D898DD383968CBD33DE9F11C71F5D471.

Default public key, RSA512 encryption exponent e: 0x010001 (65537 in decimal) Default private key, RSA512 decryption exponent d:

0xA99329F775451ABC3A44A1B749DD8D9F88B759834253EACDF5B410AA19
 BB534F6240ECA14FDD61276BF97A8A229D411BB48AA43C4D0C71CF04BCB
 DBD97D7F141.

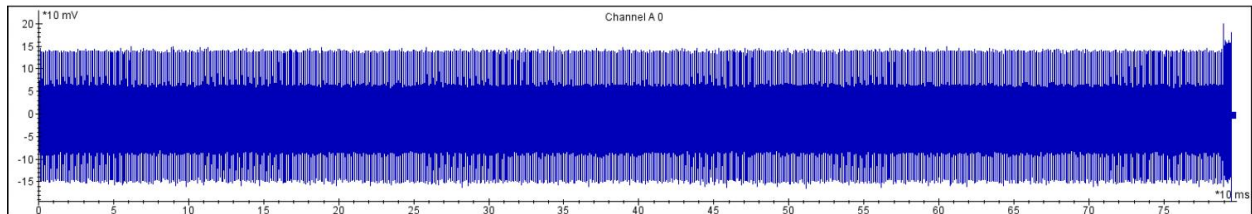
Command byte: 0xDF

Payload length: 66 bytes (2 bytes message length with value 0x0040, 64 bytes message).

Response length: 66 bytes (2 bytes message length with value 0x0040, 64 bytes message).

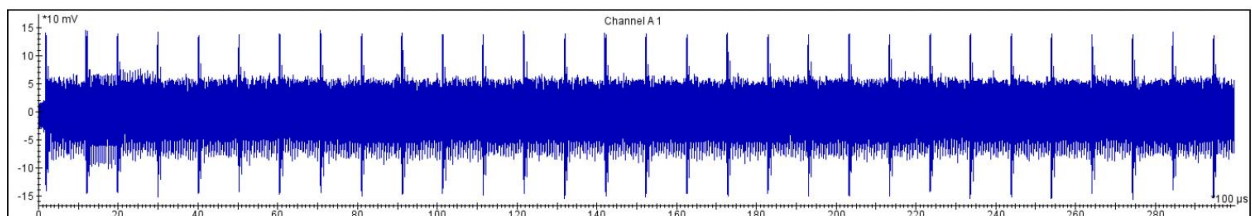
Crypto execution time @168MHz: typically 790 ms (795ms including I/O).

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Power trace capture (only first 30 ms shown):



Trigger signal PC2 1V threshold level.

Get current key from software RSA-SFM (0xD8)

Command description: the board will send back 2 bytes containing the length of the decryption exponent (d) from RSA-SFM, followed by the exponent (d).

Note that the SFM private key (decryption exponent, d) can be changed with other command, as well as the implementation of the square and multiply operation.

Command byte: 0xD8.

Response length: 2+n bytes (2 bytes defining length of exponent d, e.g. 0x0040 + n bytes message).

Default response after reboot (with default key d stored in firmware): 0x 00 40 A9 93 29 F7 75 45

1A BC 3A 44 A1 B7 49 DD 8D 9F 88 B7 59 83 42 53 EA CD F5 B4 10 AA 19 BB 53 4F 62 40 EC A1 4F DD

61 27 6B F9 7A 8A 22 9D 41 1B B4 8A A4 3C 4D 0C 71 CF 04 BC BD BD 97 D7 F1 41

Set RSA512-SFM exponentiation implementation (0xD9)

Command description: this command will change which implementation is used in the multiply operation in the RSA-SFM command. The board has two implementations: one using conditional branching (i.e. an if() statement), and other implementation using an XOR instruction for avoiding branching.

Command byte: 0xD9.

Payload length: 1 byte (0x00 for using conditional branching, 0x01 for using XOR instruction) Response length: 1 byte (should match the payload of the command if the implementation is selected correctly).

Set private key d for software RSA512-SFM (0xDB)

Command description: the command will change the private key exponent (d) used in the RSA512SFM implementation.

The payload of this command is composed by 2 bytes defining the length of the private key (typically 64 bytes, 0x0040) followed by the private key (exponent d).

Command byte: 0xDB.

Payload length: 2 bytes (typically 0x0040) + n bytes (typically 64 bytes for a 512bit exponent d) Response length: 1 byte (0xDB).

Set key generation method for RSA512-SFM (0xDC)

Command description: the command will change the source of the private key exponent (d) used in the RSA512-SFM implementation. The private key exponent d can be set to the default key stored in the firmware or to a user-provided private key (sent with another command byte).

Command byte: 0xDC.

Payload length: 1 byte (0x00 for a user-provided private key, 0x01 for internally stored default key) Response length: 1 byte (same as payload if key has been selected correctly).

Software Dilithium3 NIST Round 3 get mode (0x90)

Command description: get the mode used for this Dilithium3 implementation. The mode can be one of 2, 3, or 5. Currently, this method always returns 3.

Command byte: 0x90.

Payload length: 0 bytes.

Response length: 1 byte which contains the mode as an unsigned integer.

Software Dilithium3 NIST Round 3 set public private key pair (0x91)

Command description: set the public and private key to be used for subsequent Dilithium3 NIST round 3 sign/verify operations.

Command byte: 0x91.

Payload: Public key of size 1952 bytes, followed by private key of size 4016 bytes.

Response: 1 byte that is always zero.

Software Dilithium3 NIST Round 3 verify (0x92)

Command description: Verify a signed message.

Command byte: 0x92

Payload: Signature of length 3293 bytes, followed by message of length 16 bytes. In other words, a “signed message”.

Response: 1 byte; the byte is zero if the signature of the message is valid, non-zero otherwise.

Software Dilithium3 NIST Round 3 sign (0x93)

Command description: Sign a message.

Command byte: 0x93.

Payload: Message of length 16 bytes.

Response: If the signing failed, then the response is 1 byte. This byte is non-zero. If the signing succeeds, then the response is a zero byte, followed by the signature of length 3293 bytes.

Software Dilithium3 NIST Round 3 get key sizes (0x94)

Command description: Get the sizes of the public and private keys used in the Dilithium3 algorithm. This is a debugging command to make sure your key sizes are all the same. Command byte: 0x94 Payload: Nothing.

Response: 2-byte unsigned integer in little-endian order that contains the public key size, followed by a 2-byte unsigned integer in little-endian order that contains the private key size.

Software ECC25519 scalar multiplication (0xEC)

Command description: the board will perform a scalar multiplication on the Montgomery form of curve 25519. The implementation is based on the one from <http://munacl.cryptojedi.org/curve25519-cortexm0.shtml>.

As an additional countermeasure, this implementation performs coordinate re-randomization. For the coordinate re-randomization, a pseudo random number generator (PRNG) is used that is based on the AES algorithm in CTR mode. The seed for the PRNG (consisting of the key and the initialization vectors used for the AES algorithm) is passed as a parameter to the command. Trigger pulse in pin PC2 only bounds the scalar multiplication operation.

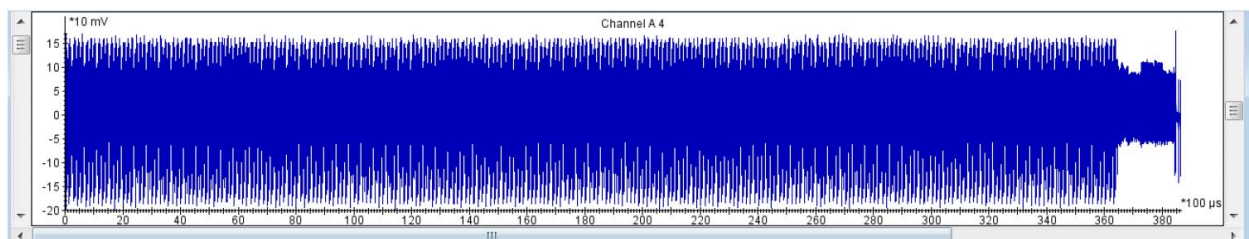
Command byte: 0xEC Payload length: 96 bytes (16 bytes key used for AES algorithm for PRNG, 16 bytes initialization vectors used for AES algorithm for PRNG, 32 bytes scalar k, 32 bytes input point P).

Response length: 32 bytes (output point R).

Crypto execution time @168MHz: ~41 ms.

Power trace capture: signal-to-noise ratio for captured power / EM traces depends on acquisition details: used oscilloscope, used current / EM probe.

Power trace capture (3.3V input, only first ~38ms shown):



Trigger signal PC2 1V threshold level.

Get random number from TRNG (0x11)

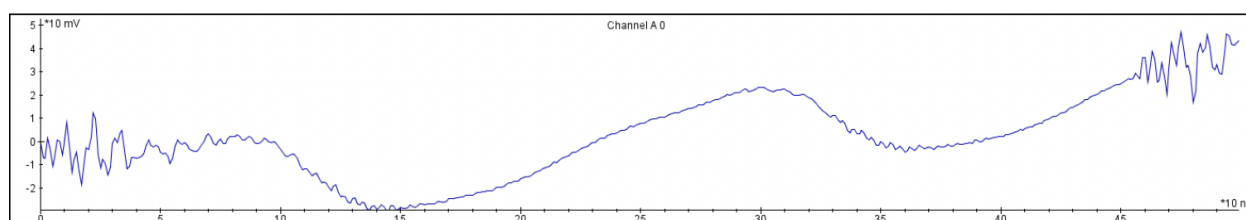
Command description: the command will enable the True Random Number Generator on the chip, generate a 32-bit random number and send it back through the serial interface. After the command completes the TRNG is disabled to avoid noise in SCA measurements of subsequent commands. The trigger pin PC2 is high while the TRNG is active.

Command byte: 0x11.

Response length: 4 bytes (generated random number, MSB sent first).

Generation time @168MHz: typically less than 460 ns.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level. Note that the ringing at the beginning and end of the power trace is due to the trigger signal; input range used for this capture was $\pm 50\text{mV}$.

Hardware DES – Encrypt (0xBE)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will perform a DES encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware DES implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

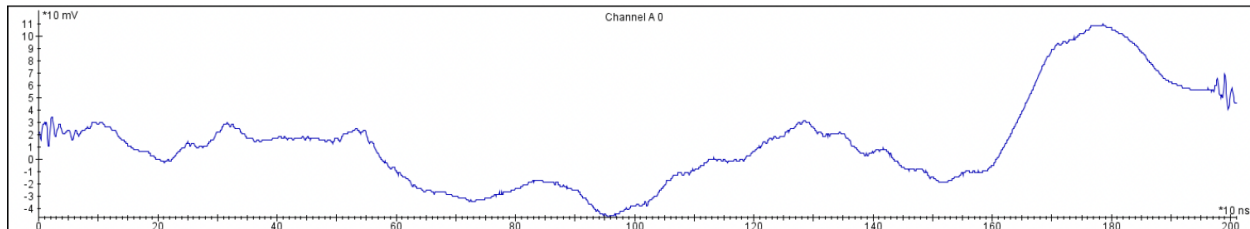
Default DES key: 0xCAFE BABE DEAD BEEF.

Command byte: 0xBE.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 2000ns, actual DES operation contained in the range [700, 1600] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming distance of input between DES rounds.

Hardware DES – Decrypt (0xEF)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will perform a DES decryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware DES implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

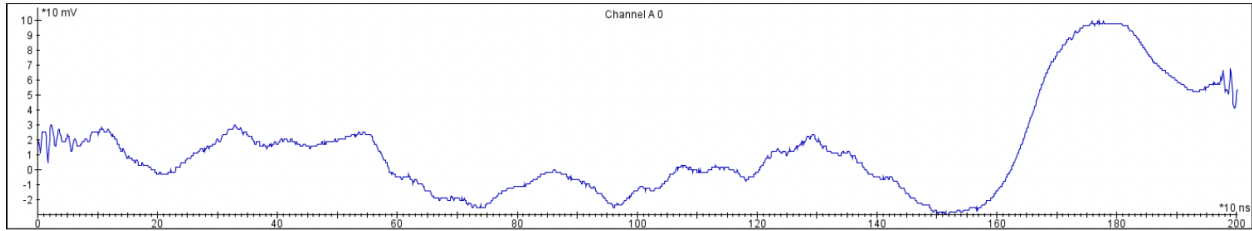
Default DES key: 0xCAFEBADEADBEEF.

Command byte: 0xEF.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 2000ns, actual DES operation contained in the range [700, 1600] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming distance of input between DES rounds.

Hardware AES128 – Encrypt (0xCA)

Note: this command is only available in the H version of the Crypto Training Target; the S version will return zeroes to this command.

Command description: the board will perform an AES128 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware AES128 implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This command is only available for the Crypto Training Target model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

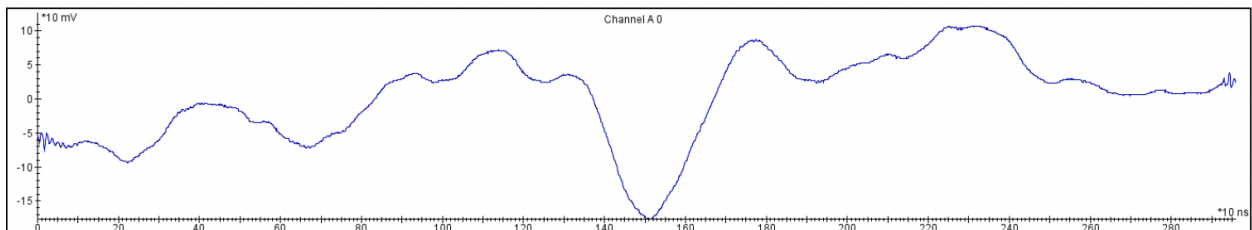
Default AES128 key: 0xCAFE BABE DEAD BEEF 0001 0203 0405 0607.

Command byte: 0xCA.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 3000ns, actual AES operation contained in the range [1100, 2200] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: a chosen-plaintext attack is recommended for attacking this implementation.

Hardware AES128 – Decrypt (0xFE)

Note: this command is only available in the H version of the Crypto Training Target; the S version will return zeroes to this command.

Command description: the board will perform an AES128 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware AES128 implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This command is only available for the Crypto Training Target model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

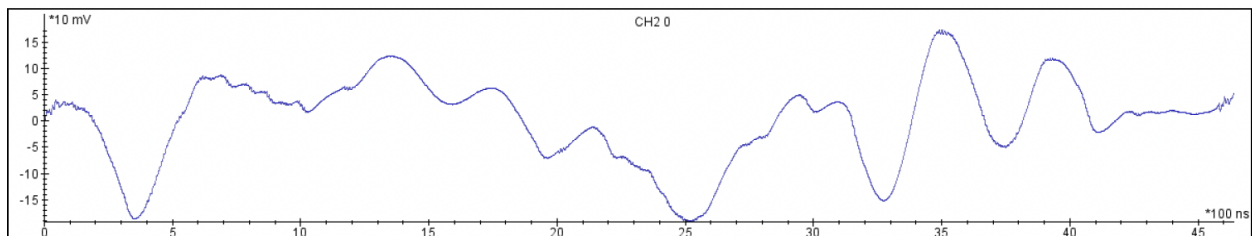
Default AES128 key: 0xCAFEBADEADBEEF0001020304050607.

Command byte: 0xFE.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 4600ns, actual AES operation contained in the range [3000, 4000] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: a chosen-plaintext attack is recommended for attacking this implementation.

Hardware AES256 – Encrypt (0x7A)

Note: this command is only available in the H version of the Crypto Training Target; the S version will return zeroes to this command.

Command description: the board will perform an AES256 encryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware AES256 implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

Default AES256 key:

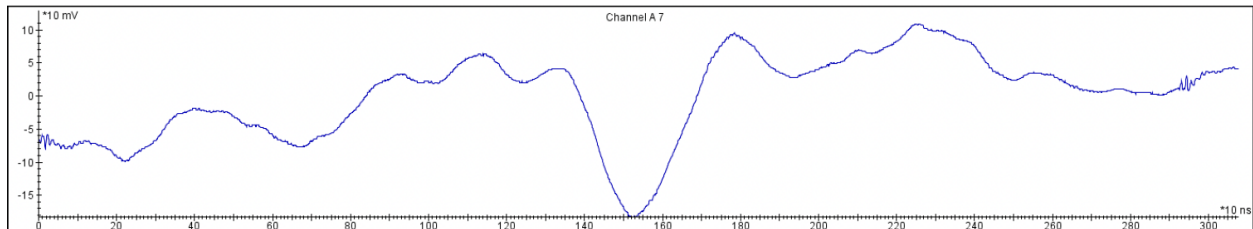
0xCAFEBABE DEAD BEEF 0001 0203 0405 0607 DABADABAD 000000 C00001 C0FFEE 55DEAD.

Command byte: 0x7A.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware AES256 – Decrypt (0x7E)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will perform an AES256 decryption of the message contained in the command payload (16 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware AES256 implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. This

command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

Default AES256 key:

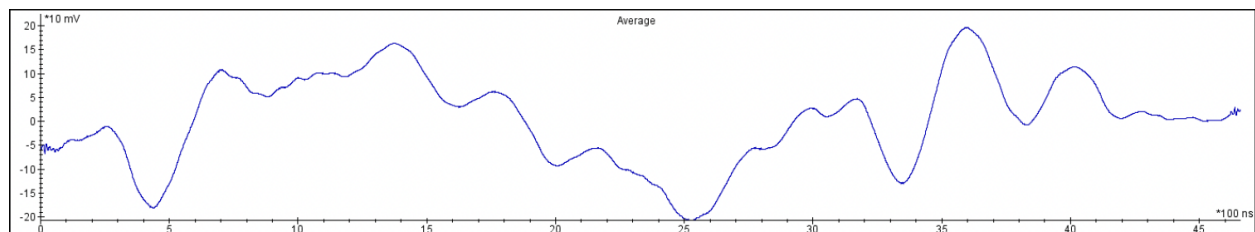
0xCAFEBABEDEADBEEF0001020304050607DABADABAD00000C00001C0FFEE55DEAD.

Command byte: 0x7E.

Payload length: 16 bytes.

Response length: 16 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware TDES – Encrypt (0xC0)

Note: this command is only available in the H version of the Crypto Training Target; the S version will return zeroes to this command.

Command description: the board will perform a Triple DES encryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware TDES implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. The default key for TDES contains the same key value for the first and third DES operation (TDES in 2-key mode). This command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

Default TDES key:

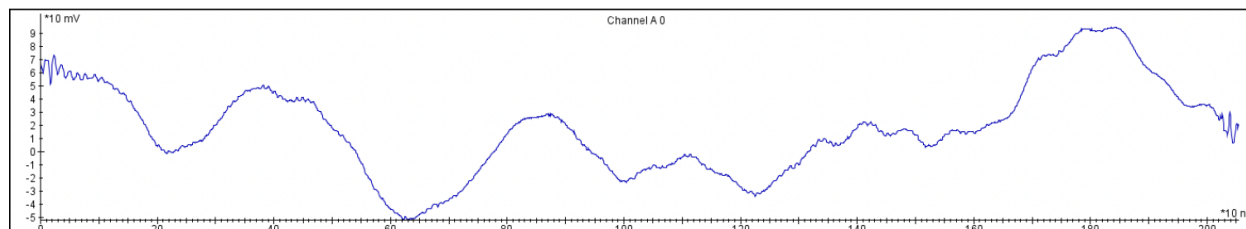
0xCAFEBABEDEADBEEF0001020304050607CAFEBABEDEADBEEF.

Command byte: 0xC0.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 2000ns, actual TDES operation contained in the range [700, 1600] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming distance of input between DES rounds.

Hardware TDES – Decrypt (0x01)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will perform a Triple DES decryption of the message contained in the command payload (8 bytes). The cipher is executed in Electronic Code Book (ECB) mode. The hardware TDES implementation performs operations using a word size of 32 bits. Key transfer to the cryptographic core is performed before rising edge of trigger pin PC2. The default key for TDES contains the same key value for the first and third DES operation (TDES in 2-key mode). This command is only available for the Piñata board model with hardware crypto support. For more implementation information of the hardware implementation, please refer to the Reference manual of the STM32F417IG chip from the manufacturer ST.

Default TDES key:

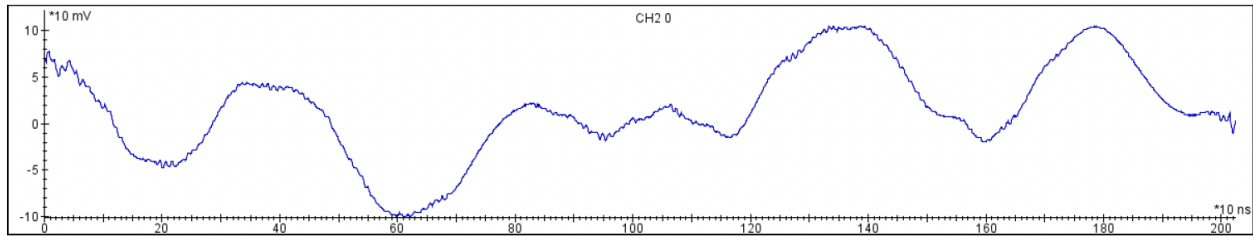
0xCAFEBABE DEADBEEF0001020304050607CAFEBABE DEADBEEF.

Command byte: 0x01.

Payload length: 8 bytes.

Response length: 8 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware cryptographic accelerator is active for 2000ns, actual TDES operation contained in the range [700, 1600] ns after rising edge in PC2 trigger signal.

Suitable model for SCA: this implementation leaks the hamming distance of input between DES rounds.

Hardware SHA1 hash (0x27)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will compute the SHA1 digest of a 16-byte given message. The output hash can be fed as an input to another SHA1 iteration in a loop, by providing the number of iterations as a parameter. As an example, if the number of iterations is 2, the output of this command would be $\text{output} = \text{SHA1}(\text{SHA1}(\text{input}))$.

The trigger pin in PC2 is high during each execution of the SHA1 algorithm.

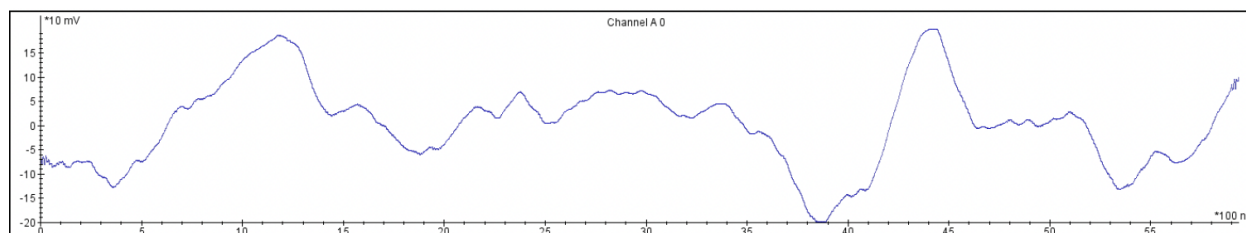
If the parameter with the number of iterations is set to zero (0x00000000), a single run of SHA1 will be computed.

Command byte: 0x27.

Payload length: 4 bytes (number of iterations, uint32 MSB first) + 16 bytes (input message).

Response length: 20 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Hardware HMAC-SHA1 hash (0x4C)

Note: this command is only available in the H version of the Piñata board; the S version will return zeroes to this command.

Command description: the board will compute the keyed-hash message authentication code of a 20-byte given input using the SHA1 hash function (HMAC-SHA1). The command accepts as a parameter the number of iterations to repeat the HMAC-SHA1 operation on the SHA1 of the previous input text as described in the following pseudocode:

for $i < \text{iterations}$:

trigger enabled in PC2

output = HMACSHA1(input)

trigger disabled in PC2 input = SHA1(input)

$i++$

write(output)

If the parameter with the number of iterations is set to zero (0x00000000), a single run of HMACSHA1 will be computed.

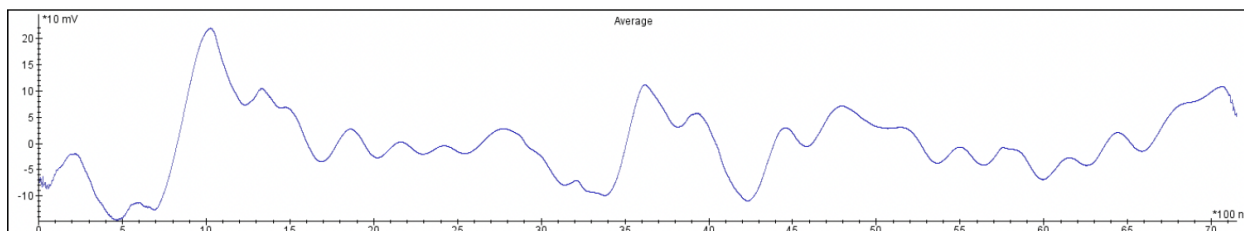
The trigger pin in PC2 is high during each execution of the HMAC-SHA1 algorithm.

Command byte: 0x4C.

Payload length: 4 bytes (number of iterations, uint32 MSB first) + 20 bytes (input message).

Response length: 20 bytes.

Power trace capture for reference (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Change DES key (0xD7)

Command description: the board will replace the default DES key by the supplied DES key in the command payload (8 bytes). After successfully updating the DES key, the board will send the new DES key as a response. The supplied key will be used for subsequent DES operations (software and hardware implementations) until the board is reset. Note that a board reset loads the default DES key from non-volatile memory during the board initialization sequence.

Command byte: 0xD7

Payload length: 8 bytes

Response length: 8 bytes

Change AES-128 key (0xE7)

Command description: the board will replace the default AES-128 key by the supplied AES-128 key in the command payload (16 bytes). After successfully updating the AES key, the board will send the new AES key as a response. The supplied key will be used for subsequent AES-128 operations (software and hardware implementations) until the board is reset. Note that a board reset loads the default AES-128 key from non-volatile memory during the board initialization sequence.

Command byte: 0xE7.

Payload length: 16 bytes.

Response length: 16 bytes.

Change TDES key (0xC7)

Command description: the board will replace the default TDES key by the supplied TDES key in the command payload (8 bytes). After successfully updating the TDES key, the board will send the new TDES key as a response. The supplied key will be used for subsequent TDES operations (software and

hardware implementations) until the board is reset. Also, the HMAC-SHA1 command uses this key. Note that a board reset loads the default TDES key from non-volatile memory during the board initialization sequence.

Command byte: 0xC7.

Payload length: 24 bytes.

Response length: 24 bytes.

Change AES-256 key (0xF7)

Command description: the board will replace the default AES-256 key by the supplied AES-256 key in the command payload (32 bytes). After successfully updating the AES key and reinitializing the AES-256 key schedule for the software implementation, the board will send the new AES key as a response. The supplied key will be used for subsequent software/hardware AES-256 operations until the board is reset. Note that a board reset loads the default AES-256 key from non-volatile memory during the board initialization sequence.

Command byte: 0xF7 Payload length: 32 bytes.

Response length: 32 bytes.

Change password (0xA5)

Command description: the board will replace the default password (used by e.g. the password check commands) by the supplied 4 byte password in the command payload (4 bytes). After successfully updating the password, the board will set the “authenticated” flag to 0 (not authenticated) and send the new password as a response. The supplied password will be used for subsequent operations that use this password until the board is reset. Note that a board reset loads the default password from non-volatile memory during the board initialization sequence.

Command byte: 0xA5.

Payload length: 4 bytes (new password).

Response length: 4 bytes (new password).

Loop counter (0xDD)

Command description: this command is used for verifying the behavior of the target in a Fault Injection scenario. The command payload is a 16 bit unsigned integer value, most significant byte first. The board initializes in volatile memory

(RAM) two counters: an up-counter, which counts from zero to the value defined in the payload, and a down-counter, which counts from the value defined in the payload to zero. The board will then execute a loop as many times as indicated in the value defined in the payload. Each loop iteration the up-counter is incremented by 1, and the downcounter is decreased by 1. After the loop finishes, the board sends as a response a byte array composed by the fixed value 0xA5, the value of the down-counter (most significant byte first), the value of the up-counter (most significant byte first) and a fixed value 0xA5.

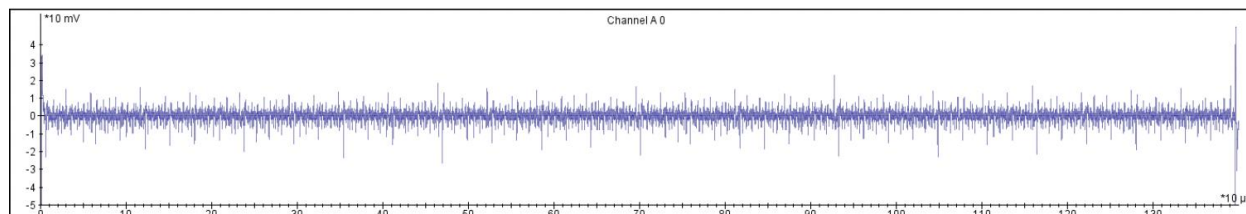
Command byte: 0xDD.

Payload length: 2 bytes (16 bit unsigned integer, most significant byte first).

Response length: 6 bytes (1 byte 0xA5, 2 bytes down-counter, 2 bytes up-counter, 1 byte 0xA5).

Example expected behavior: Command 0x DD 3030. Response 0x A5 0000 3030 A5.

Power trace corresponding to command payload 0x3030 (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Command execution for a loop count of 0x3030 (12336 iterations) is 1.40ms long (excluding I/O).

OLED screen test (0x30)

Command description: the Crypto Training Target can drive a 128x32 pixel OLED screen based on a SSD1306 controller (SPI interface). This command will display the string Hi ! in the OLED screen.

Command byte: 0x30.

Payload length: 0 bytes.

Response length: 0 bytes.

Get code revision (0xF1)

Command description: the Crypto Training Target will send 8 bytes containing an ASCII string with the code version loaded on the board. This command returns “BadCmd\n\00” in code versions <2.0.

Command byte: 0xF1.

Payload length: 0 bytes.

Response length: 8 bytes (ASCII string).

Get chip UID (0x1D)

Command description: the Crypto Training Target will send the 96bit UID of the onboard STM32F4 chip. The returned value will be different for each chip.

Command byte: 0x1D Payload length: 0 bytes.

Response length: 12 bytes (MSB first).

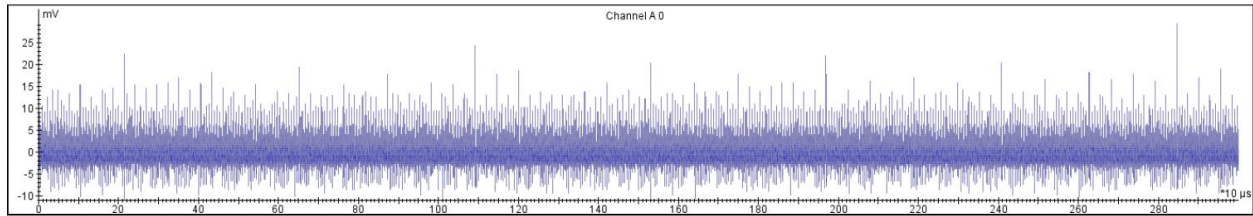
Infinite Loop for FI (0x99)

Command description: this command is used for practicing Fault Injection attacks. Once the command is sent, the board enters an infinite loop in which it is performing an increasing count; the counter value is reset each half second (approximately), and a SPI transmission is performed each time that the counter is reset. Note that the board will not respond anymore to commands unless the board is reset or successfully glitched!. The pin PC2 is set high while the board is in the infinite loop.

If a FI attack manages to exit the infinite loop, immediately after the code block of the infinite loop there is a NOP-sled (series of NOP operations) and the board will send the ASCII string “Glitched!” via the UART serial interface. The UART over USB interface should not be used for this command.

Command byte: 0x99.

Response length: 8 bytes only if infinite loop is exited and code below is executed Power trace from first 3 ms (3.3V input, measured with Keysight Passive Current Probe):



Trigger signal PC2 1V threshold level.

SRAM-SRAM key copy (0x38)

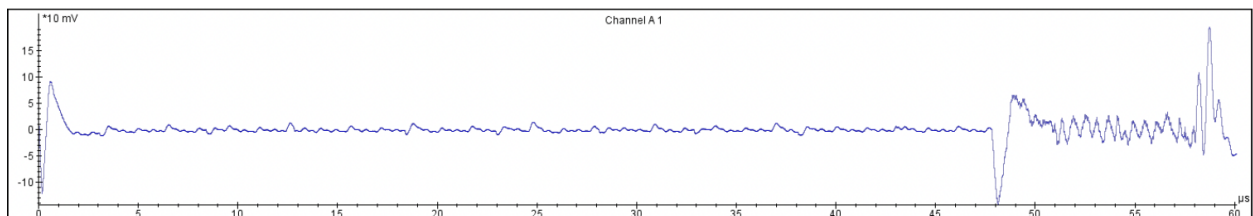
Command description: the board will perform a byte-wise copy of 16 bytes sent to the board. The purpose of this command is to simulate SRAM-SRAM copying of key material e.g. for Template Analysis attacks. The key copy has a long fixed delay (~47us) before starting the copy (implemented with a busy wait) as well as a small delay between each byte copy, in order to make more visible each key byte copy. The trigger pin PC2 is set high while the copy is performed.

Command byte: 0x38.

Payload length: 16 bytes.

Response length: 16 bytes (should match the payload).

Power trace (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level

Password check command: single check (0xA2)

Command description: this command performs a password check of a 4 byte password. There is a small fixed delay before the check to stabilize the power consumption to make the check more evident in SCA traces.

If the password sent as payload matches the current password in the board, the command will reply with 0x9000. Otherwise, the command will reply with 0x6986. Note that a successful verification of the password also sets the global “authenticated” flag to 1 (meaning “true”).

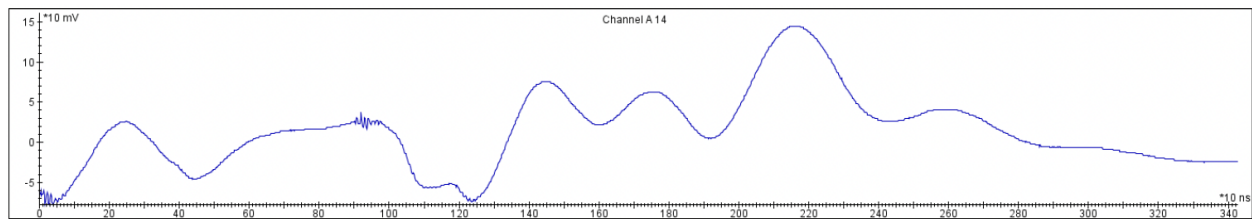
The default password is 0x02060208.

Note that the trigger pin PC2 is high while the password is compared and goes low just before the actual check (is digits ok==4?) is performed. The check happens around 1 us (see the trigger signal).

Command byte: 0xA2.

Payload length: 4 bytes (password).

Response length: 2 bytes (status code) Power trace (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Password check command: double check (0xA7)

Command description: this command performs a double password check of a 4 byte password. There is a small, fixed delay before the check to stabilize the power consumption to make the first check more evident in SCA traces, and a small delay between the first and second check.

If the password sent as payload matches the current password in the board, the command will reply with 0x9000. Otherwise, the command will reply with 0x6900 or 0x6986 if you skip the first check with FI. Note that a successful verification of the password also sets the global “authenticated” flag to 1 (meaning “true”).

The default password is 0x02060208.

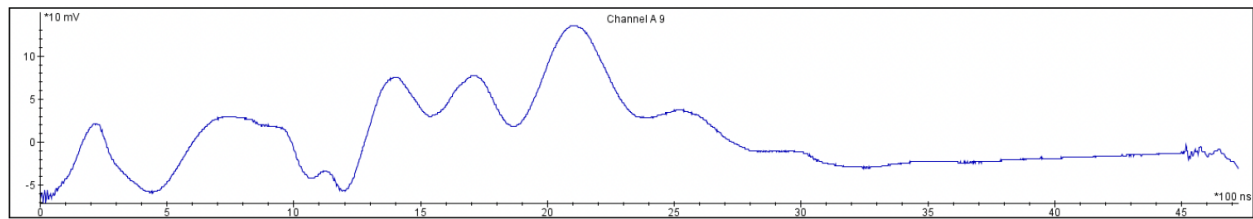
Note that the trigger pin PC2 is high while the password is compared and the checks are performed.

Command byte: 0xA7.

Payload length: 4 bytes (password).

Response length: 2 bytes (status code).

Power trace (3.3V input, measured with Passive Current Probe):



Trigger signal PC2 1V threshold level.

Set CPU speed (0xF2)

Command description: the Crypto Training Target attempt to change the clockspeed of the STM32F4 chip. Currently supported speeds are 30MHz, 84MHz and 168MHz (default). If the clockspeed is set correctly, the board will reply a byte with the current speed (unsigned byte).

If the parameter is not one of the supported clockspeeds, the board will attempt to set the clockspeed to the default value of 168MHz.

Note that lower clockspeeds will result in lower power consumption. This is normal, as power consumption is proportional to the clockspeed².

Command byte: 0xF2.

Payload length: 1 byte (0x1E for 30MHz, 0x54 for 84MHz, 0xA8 for 168MHz)

Response length: 1 byte (current clockspeed).

Set clock source (0xF3)

Command description: this command will set the clock source to the internal PLL (default) or to an external clock source. Usage of this command is discouraged, as it will typically stall the board until the next reset if the external clock source is not set up correctly. For more information about how to supply an external clock supply to the chip, please refer to the Reference manual of the STM32F407IG or STM32F417IG chip from the manufacturer ST.

Command byte: 0xF3.

Payload length: 1 byte (0x00 for external clock, different value for internal PLL).

Response length: 1 byte (0x00 for external clock, different value for internal PLL).

How to Develop Code for the Crypto Training Target

Before starting ColDE (do this only once in new computers)

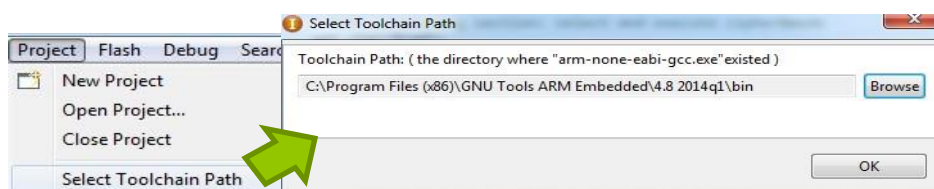
The following programs need to be installed in the following order from the Crypto Training Target software package:

1. ST-Link Drivers\st-link_v2_usbdriver.exe.
2. ST-Link Drivers\STM32 ST-LINK Utility_v3.3.0.exe (or newer version).
3. gcc-arm-none-eabi-4_8-2014q1-20140314-win32.exe (or newer version).
4. ColDE-1.7.1 or newer version (should be 1.x, not ColDE 2).

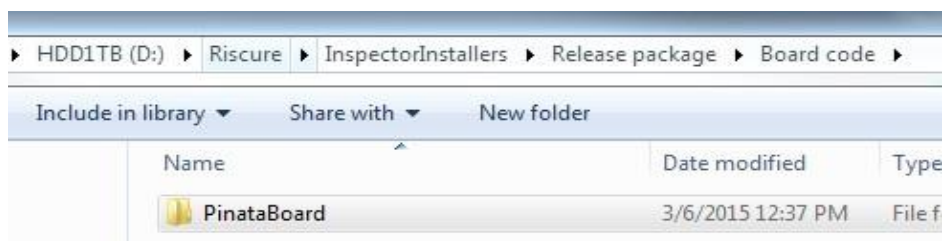
Please install all programs with default options. If you do not have the Crypto Training Target software package, please contact Keysight Support for it (<https://support.keysight.com>).

How to modify the Crypto Training Target code with CoIDE

1. Open the CoIDE program. The first time that you start CoIDE you need to configure where the GCC ARM toolchain is installed. Please provide the proper path where it was installed in the first step of the document (section Before starting CoIDE). By default this path is C:\Program Files (x86)\GNU Tools ARM Embedded\4.8 2014q1\bin. You can supply this path if prompted by CoIDE, or by clicking on Project >> Select Toolchain Path. Click on OK once you selected the correct path.

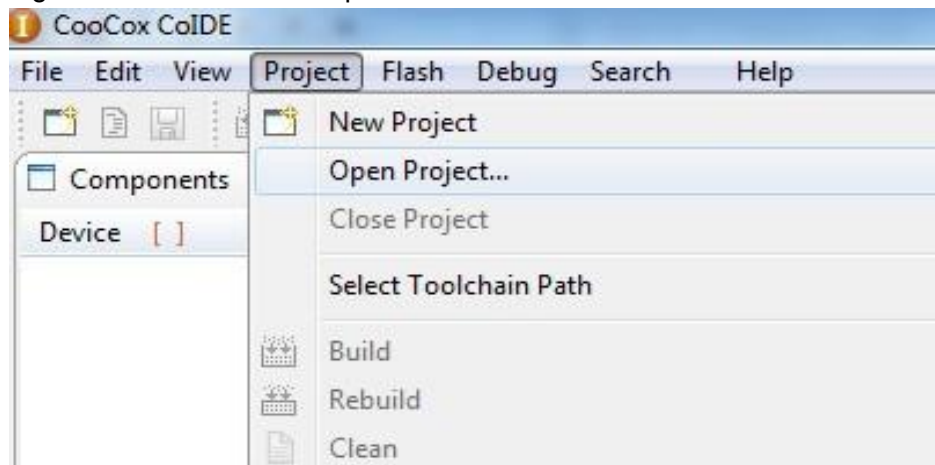


Additionally, open a Windows explorer window, and copy the complete “PinataBoard” folder inside the “Board code” folder of the Piñata software package into your hard disk drive (e.g. C:\development\pinataBoard).

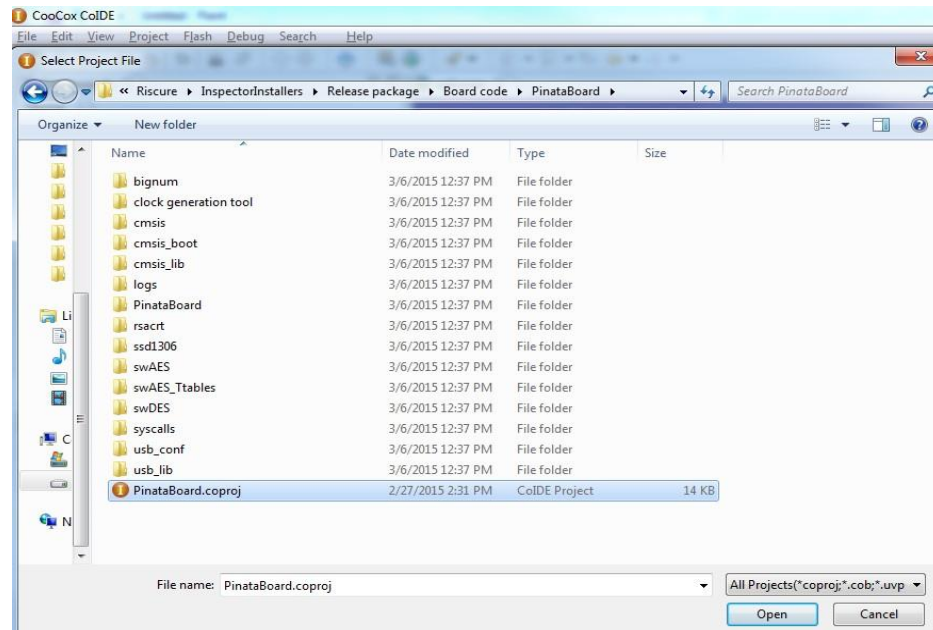


2. CoIDE is an Eclipse-based IDE for developing code for the Piñata board (based on Eclipse). To open the Piñata board code, click on *Project >> Open project...*

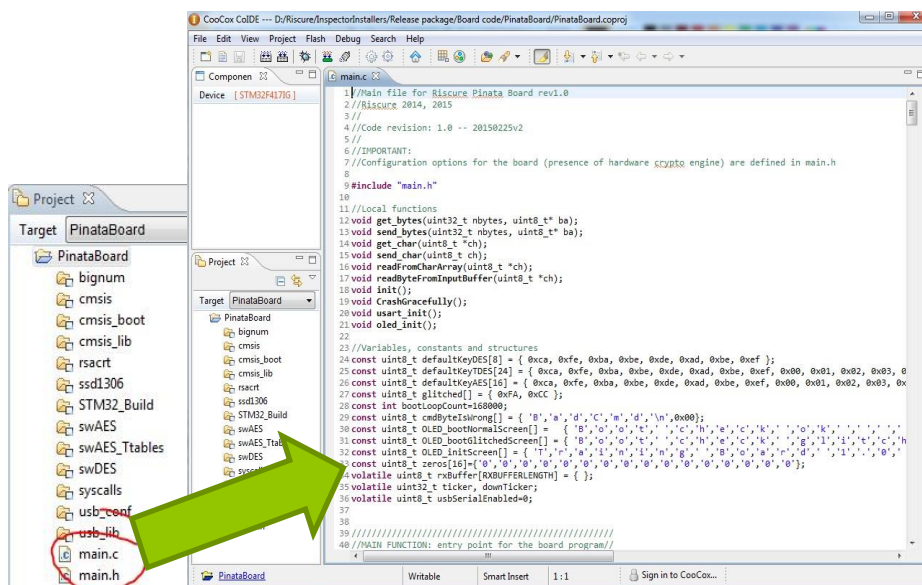
Figure 67 Power trace capture



3. In the Open Project dialog, browse to the folder where you copied the “PinataBoard” folder, and select the PinataBoard.coproj file. Click on the *Open* button.



4. The Pinata board project will be loaded in the “Project” window. In the project tree, double click on the “main.c” file and it will be displayed in the main code window.
5. Now you can modify the board source code. Typically, all your changes should be done in the “main.c” file, inside the `int main(void){...}` function, just before the “MAIN FUNCTION LOOP” if your function should be performed at boot time, or inside the “MAIN FUNCTION LOOP”.



Example places where you can insert code in main.c:

```
.....
if(glitchedBoot){
    oled_sendchars(21,OLED_bootGlitchedScreen);
}
else{
    oled_sendchars(21,OLED_bootNormalScreen);
}
oled_sendchars(36,OLED_initScreen);
```

=====INSERT CODE HERE IF YOUR CODE SHOULD BE EXECUTED AT BOOT TIME=====

```
////////////////////
//MAIN FUNCTION LOOP//
////////////////////

//Note: comment all hardware cryptographic commands if using
STM32F407IGT6 instead of using STM32F417IGT6 while (1) {
    //Main loop variable (re)initialization
    for (i = 0; i < RXBUFFERLENGTH; i++) rxBuffer[i] = 0; //Zero
the rxBuffer
    for (i = 0; i < MAXAESROUNDS; i++) keyScheduleAES[i] = 0;
//Zero the AES key schedule (T-tables AES implementation)
    charIdx = 0; //RSA: Global variable with offset for reading the
ciphertext, init to 0
    payload_len = 0x0000; //RSA: Length of the ciphertext; init to
zero, expected value for 1024bit RSA=128byte
    cmd=0;
```

```

//Main processing section: select and execute cipher&mode
get_char(&cmd);          switch (cmd) {
=====INSERT CODE HERE AS A CASE(cmdId) BLOCK TO EXECUTE YOUR
COMMAND AFTER SENDING THE cmdId BYTE TO THE BOARD=====
        ===== YOU CAN MODIFY AN EXISTING COMMAND TO
REPLACE/EXTEND THE CURRENT FUNCTIONALITY =====
        //////////Software crypto commands//////////
        //Software DES - encrypt          case (0x44):
        get_bytes(8, rxBuffer); // Receive DES plaintext
        GPIOC->BSRRL = GPIO_Pin_2; //Trigger on
        des(keyDES, rxBuffer, ENCRYPT); // Perform software DES
encryption          GPIOC->BSRRH = GPIO_Pin_2; //Trigger off
        send_bytes(8, rxBuffer); // Transmit back ciphertext via
UART
        break;
.....

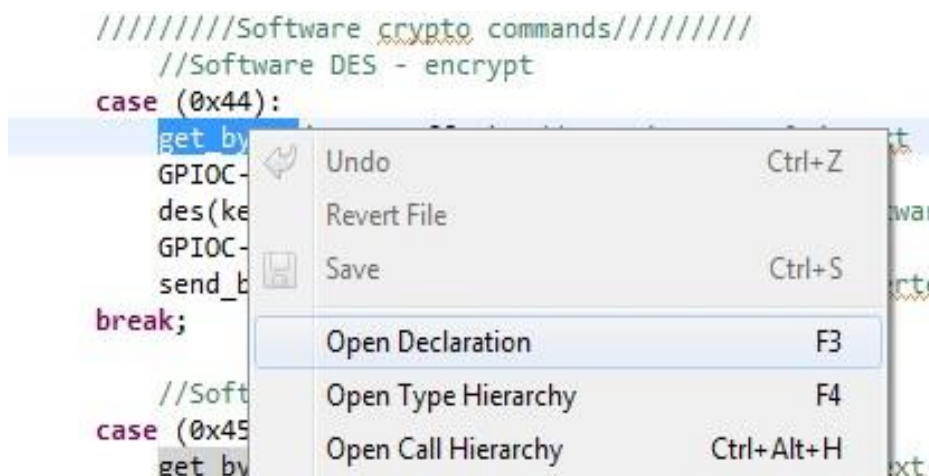
```

6. Some important remarks for developing code with the Crypto Training Target and CoIDE:

- In the file “main.h” there is a global definition that specifies which Piñata Board model you are using. Please comment or uncomment this line according to your board model. By default, it is uncommented (i.e. configured for the Piñata Board with hardware cryptographic engine STM32F417IGT6, tagged with an H letter).



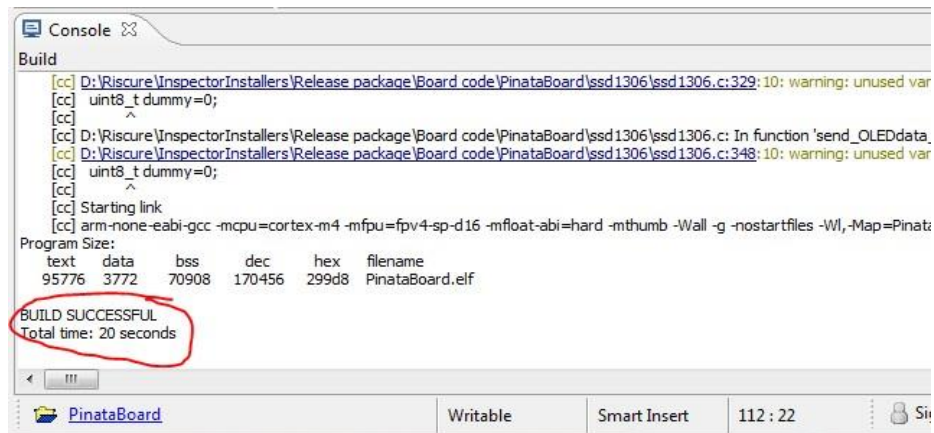
- Code is refactored in different files for a cleaner structure. To find where the code is for some function, right click on it and select Open Declaration to open the file where the implementation code is found (or highlight the function and press F3).



- The Crypto Training Target (ARM Cortex M4F @168MHz) has 196kB of RAM. Please avoid using deep recursion in functions to avoid exhausting the stack size. Default stack size is 16Kbyte; this value can be changed in the `STACK_SIZE` definition inside the file found in `cmsis_boot/startup/startup_stm32f4xx.c`
 - For message input/output, use exclusively the helper functions in the Piñata Board source code:
 - `get_bytes`, `send_bytes`, `get_char`, `send_char`. Note that `printf()` and `scanf()` are not implemented.
 - See the source code inline comments for the function descriptions.
 - Maximum program size is 1MByte. Check program size after compilation.
- 7. Once you have finished the code modification, please rebuild the project.** To do so, click on the “Rebuild” button or press Ctrl+R. Alternatively, you can click on *Project >> Rebuild*. Note that this action will save all your code modifications in the source code files.



8. If the code is valid, the compilation will succeed. CoIDE will display messages and will generate a “PinataBoard.hex” file under the “PinataBoard\Debug\bin”. Note that the provided code generates warning messages; you can safely ignore them.



If there are errors in the code, the build will not be successful; please scroll through the console message, and you can click on the error line to check what the problem was.

NOTE

A successful compilation does not imply that your program will be executing as expected (there may be other errors not detected at compilation time, e.g. illegal memory accesses or infinite loops).

9. Once the code is successfully compiled, for loading the code on the board please follow the steps of the section. Programming Piñata Board over USB.
10. If you want to debug code on the board, you need to buy separately a ST-LINKv2 debug tool (<http://www.st.com/web/en/catalog/tools/PF251168>). This tool can be attached to the JTAG port of the Crypto Training Target, and then you can click on *Debug* > > *Debug* for on-board code depuration. If you do not have this tool, please contact Keysight for references on how to acquire one.

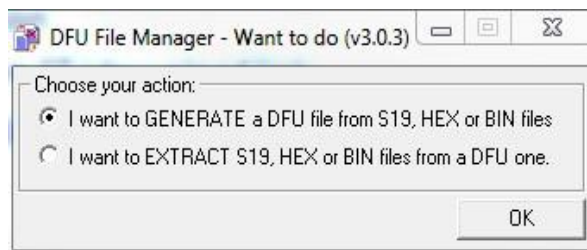
How to Generate a DFU File for Updating Crypto Training Target Firmware Over USB

Requirements

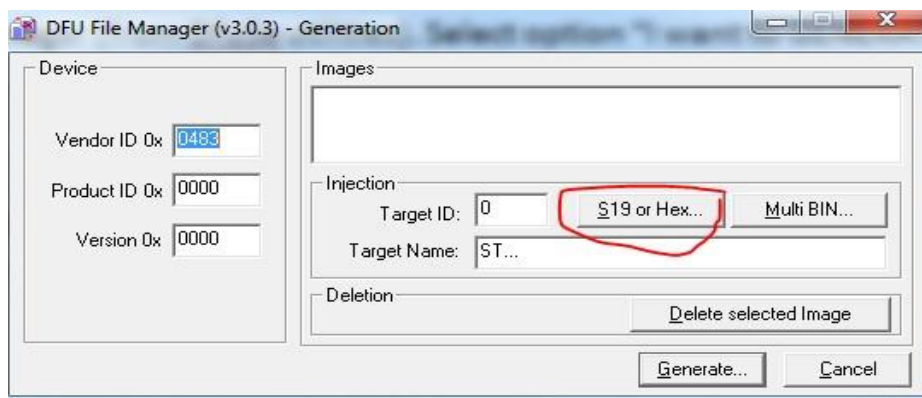
- Working ColDE installation.
- Install DfuSe utilities from ST: (download link)
<http://www.st.com/web/en/catalog/tools/FM147/CL1794/SC961/SS1533/PF257916>

Steps

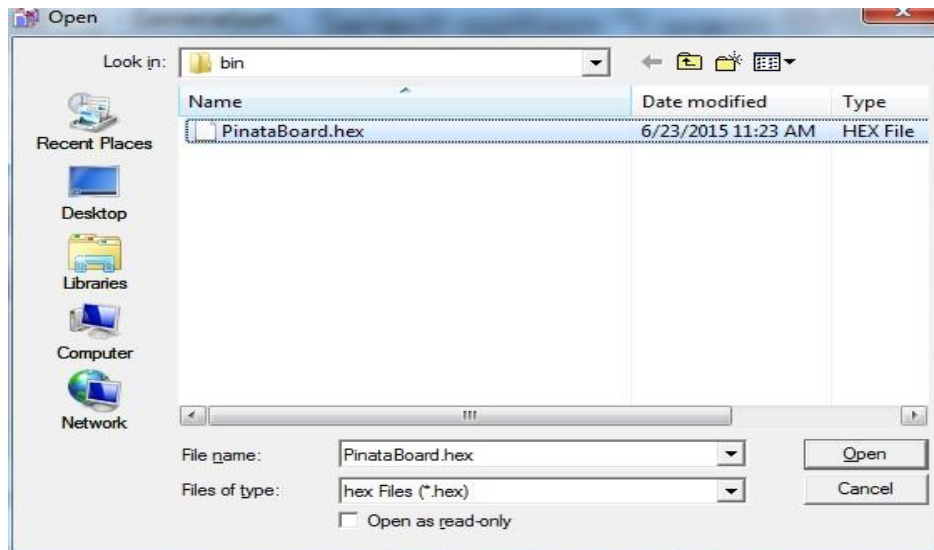
1. Compile the ColDE project successfully. It will generate a PinataBoard.hex file in the PinataBoard/debug/bin folder inside your project folder.
2. Open DFU file manager (from DfuSe utilities). Select option “I want to GENERATE...”, and click on OK.



3. On the Generation dialog that shows up, click on “S19 or Hex..” button.



4. Select the “hex Files (*.hex)” in the Files of type field, click on the PinataBoard.hex file and then click on Open.



5. Configure the device settings as follows:

Device panel:

Vendor ID: 0483

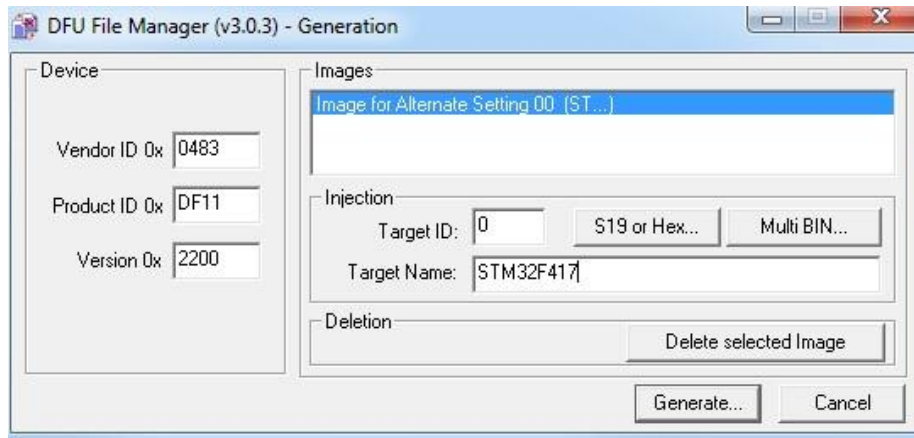
Product ID: DF11

Version: 2200

Injection panel:

- Target ID: 0
- Target name:
 - For Piñata S boards: STM32F407
 - For Piñata H boards: STM32F417

Example configuration for HARDWARE board:



6. Click on Generate... and save the .dfu file. There should be a popup indicating the Success of the operation.

The generated .dfu file is required for flashing the Crypto Training Target over USB with a DFU flashing tool such as DfuSe.

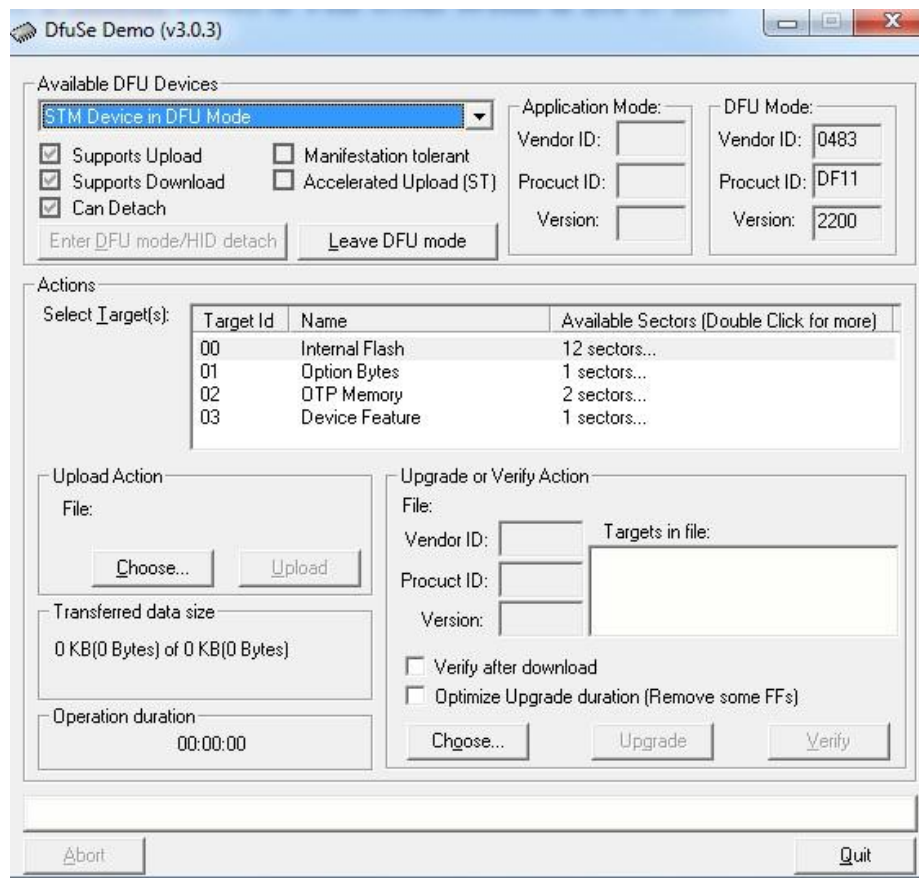
How to Flash the Crypto Training Target via USB with a DFU Binary File

Requirements

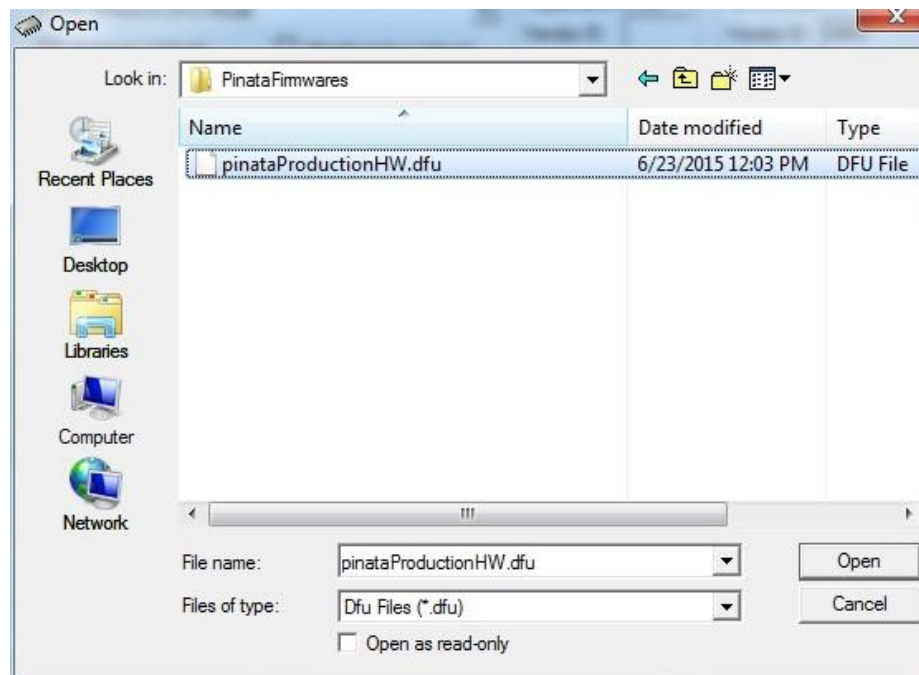
- Install DfuSe utilities from ST: (download link)
<http://www.st.com/web/en/catalog/tools/FM147/CL1794/SC961/SS1533/PF257916>

Steps

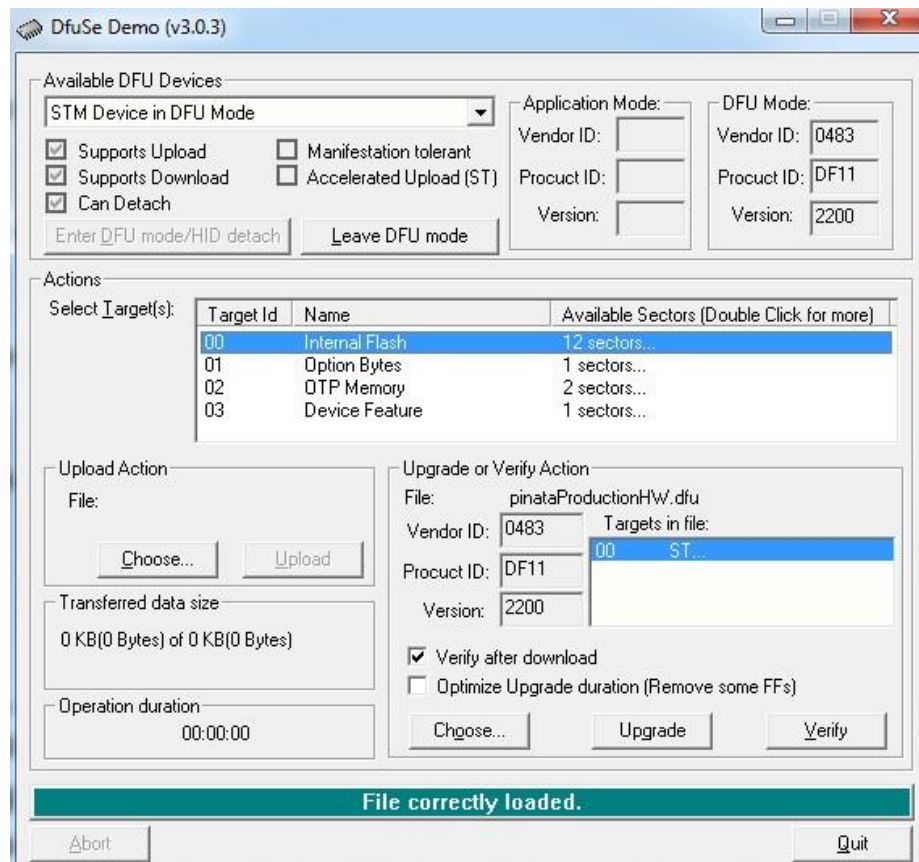
- 1.** Generate a DFU file from the source code if a DFU file was not provided. See the section How to generate a DFU file for updating Piñata firmware over USB for performing this.
- 2.** Open the DfuSe Demonstration tool in windows
- 3.** Set the “Boot config” switch in the Crypto Training Target to the SYSTEM position, “Power input selector” switch to USB and connect the Piñata board to the PC with the MiniUSB cable only. Wait for driver installation to be successful if Windows starts installing DFU drivers (this happens the first time that the board is connected in programming mode; drivers are found in the DfuSe tool installation folder).
- 4.** Press the RESET button on the Piñata board for one second. After releasing the reset button, the DfuSe Demo tool will automatically detect the Piñata board in programming mode:



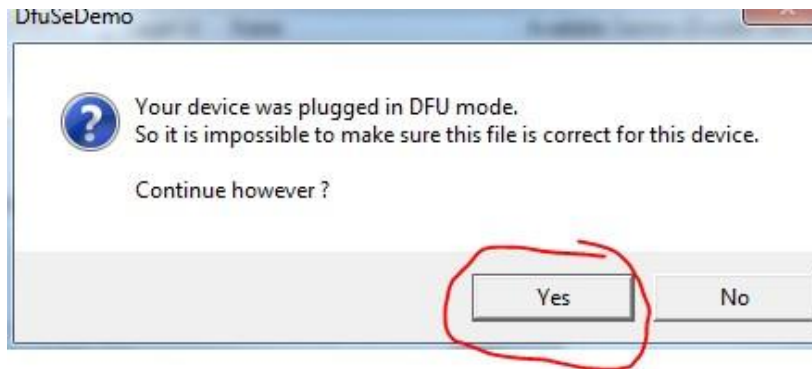
5. In the Upgrade or Verify Action, click on Choose... and open your DFU file.



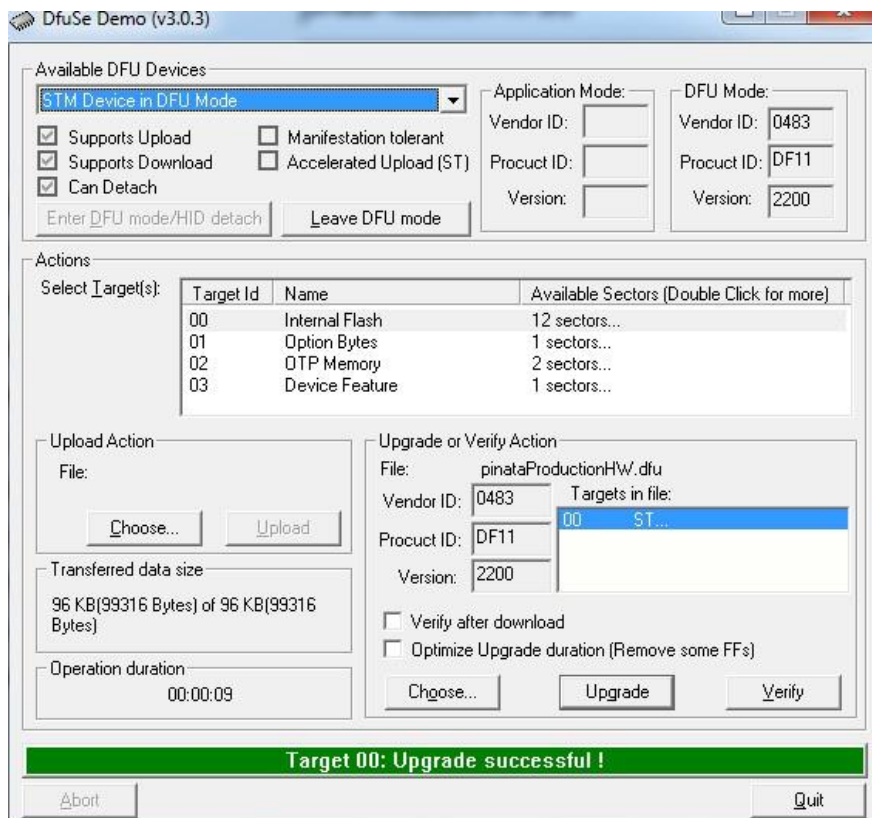
6. The DfuSe demo program will display a green message “File correctly loaded”.
7. For flashing the board, perform the following sequence of operations:
 - Disable the “Verify after download” option in the Upgrade or Verify Action panel (if not already disabled). This option may crash DfuSe if enabled.
 - Disable the “Optimize Upgrade duration” option in the Upgrade or Verify Action panel (if not already disabled).
 - Click on the “00 ST..” row in the Targets in file: table from the Upgrade or Verify Action panel. It should be highlighted.
 - In the Actions panel, click on the “Internal Flash” row from the Select Target(s) table. It should be highlighted. The DfuSe tool should look similar to the following screenshot:



- Click on the “Upgrade” button. When the program displays the following warning message, click on “Yes”.



- The tool will start programming the board and displays the progress. Once programming is finished, the tool will display a message “Upgrade successful!”.



8. Once programming is finished, set the switches of the board to the normal (default) mode:
 - Boot config switch: FLASH.
 - Power input switch: USB.
9. Press the RESET button to restart the board and execute the new program. Your board should be ready to use!

Troubleshooting and Frequently Asked Questions

The board is not responding to commands

- Verify that the boot selector switch is set to FLASH.
- Press the RESET button for one second and retry communication.
- If you are using the FTDI cable for UART communication: verify that the cables are properly connected, and the I/O selection jumper is removed from the board. Verify that the board is properly powered (PWR led is ON) and press the RESET button.
- If you are using the Serial over USB communication: verify that the cables are properly connected, and the I/O selection jumper is placed in the board between PA9 and VBUS. Verify that the board is properly powered (both LEDs are ON) and press the RESET button.
- Verify that the RESET pin in the JTAG Port connector is NOT connected to GND (0V).

The board is not powering up (both LEDs are off)

- Verify that you are using only one power source at a time: 5V pin, 3.3V pin, or MiniUSB cable.
- Verify the polarity of the power supply cables. Inverting the polarity of the power supply cables may permanently damage the board!
- Verify that you are using the appropriate voltage level for the different power inputs. Do not feed 5V into the 3.3V pin as this may permanently damage the board!
- Verify that the Power input selector is at one of the extremes of the switch; putting the switch in the central position may result in unexpected behavior or failure to power the board.
- Verify that the boot selector switch is set to FLASH.

My power traces look very different from the manual. Why?

- Verify that you are using an appropriate sampling speed (minimum 500MSa/s, recommended 1GSa/s or higher), I/O is as expected (e.g. not zeroes) and triggering is set up correctly.

- Verify that your measurement is a power trace; EM measurements look very different.
- Verify that you are measuring in the 3.3V input. Measuring in the 5V input or between the 3.3V and VBAT pins produces different measurements.
- If you are not using Keysight hardware equipment, your power measurements may vary.

My board works but the oscilloscope does not trigger with the signal from the PC2 pin

- Verify that the board is properly working and the I/O communication channel works as described in the sections “Basic setup for Crypto Training Target – UART communication” or “Basic setup for Crypto Training Target – Serial over USB”. After this verification, check that it still works with your original measurement setup.
- Verify that the trigger signal is properly connected to your oscilloscope. If you use an oscilloscope probe, verify that it is set to x1 attenuation and the ground connector is properly connected to any of the GND pins.
- Verify that the trigger port from the oscilloscope is properly configured in the trigger configuration of your acquisition setup.
- Verify that the timeout setting for the trigger signal is long enough in your configuration.
- The trigger signal may be missed by the oscilloscope if the oscilloscope does not arm fast enough in repeated acquisitions. Introduce / increase a post-arming delay time in the communication sequence for allowing the oscilloscope to detect the trigger signal.

How can I find out if my Crypto Training Target has a hardware crypto engine?

- The Crypto Training Target has two models: with a hardware cryptographic accelerator, and without it. To distinguish between the two models, look for the letter in the Crypto Training Target on top of the CPU. The model with the hardware crypto engine has an H in the label. The model without the crypto engine has an S in the label. Please see the picture below as a reference:



- Please note that the default firmware for the Crypto Training Target without the crypto engine (Software Crypto Training Target) will reply ASCII zeroes (0x30) to the hardware accelerated commands.

My Crypto Training Target replies “BadCmd” to some commands. Why?

- Some commands from the Crypto Training Target Firmware version v2.1 are not included in previous versions. If you want to know which firmware version is programmed in your board, please see the description of the command “Get code revision (0xF1)”.
- If you send a longer/shorter payload for certain commands, the board may interpret some command payload bytes as new commands. Please reset the board to remove this behavior, and check that the length of the payload matches the expected length from the command.

My question is not answered in the manual

Please contact Keysight via our support portal at <https://support.keysight.com>.

