Swissbit UBoot SDK User Manual

Version: 1.4

Create date: 21.06.2019

Author: Swissbit Germany AG

Table of Content

1 Document Information

1.1 About this Document

This document describes how to safeguard the integrity of the boot media of a Raspberry Pi board.

A Swissbit Data Protection (DP) card is a microSD/SD card that allows encryption and access protection of data stored on the card. The DP card safeguards a data policy that is enforced with minimum interaction of the host system.

Swissbit provides a U-Boot Software Development Kit (SDK) to integrate a Swissbit Data Protection (DP) card into a U-Boot boot environment.

This document describes how to install and use various components of the Swissbit U-Boot SDK together with a Swissbit Data Protection (DP) card.

Chapter 2 outlines two use-cases: Protecting the integrity of a Raspberry Pi boot media.

Chapter 3 gives an overview of the contents of the SDK, its various files and its purposes. Chapter 4 describes the configuration of the Swissbit DP card and the installation process of the Swissbit U-Boot SDK binary files. Chapter 5 gives an overview of the provided sample source code and how to build the binary files. Chapter 6 gives links to further information and to software sources mentioned in this document.

1.2 Document History

Version	Updated on	Updated by	Short description
1.3	October 19th, 2018	Swissbit AG	Released version of the document
1.4	June 19th, 2019	Swissbit AG	Document update after Review

1.3 Glossary

Abbreviation	Description
API	Application Programming Interface
DP	Data Protection
SDK	Software Development Kit
GUI	Graphical User Interface
CLI	Command Line Interface
SO	Security Officer
SHA	Secure Hash Algorithm
PIN	Personal Identification Number Note: In this document PIN is synonym for password as any binary value can be defined. In practice the password will most probably be a ASCII PIN
NVRAM	Non-Volatile Random Access Memory

2 Overview of Use-Cases

This chapter outlines two exemplary use cases how to protect data and system integrity on Raspberry PI boards.

Depending on the policy you will need the following hardware for the use-cases described below:

- A Raspberry PI 1, 2 or 3 and its peripherals
- A Windows machine for configuring the Swissbit DP card(s).
- A Swissbit DP card.
- In case of choosing USB Policy: an additional Swissbit DP card
- In case of choosing NET Policy: an additional Raspberry Pi board.
- In case of building U-Boot binaries from sources: a machine capable of running x86-64 Ubuntu 18.04.

2.1 Protecting the Integrity of a Raspberry PI Boot Media

A Raspberry Pi board boots from a SD (RPI 1) or micro SD (RPI 2 and 3) card inserted into the board. A default Raspbian installation installs the kernel on the boot partition and the root files system on a separate second partition. If standard storage cards are used typically all data and files in both partitions can be read, modified and deleted by anybody.

Swissbit DP cards allow to restrict access to data on the card by various configurable policies. The boot image can be set read-only to prevent from unauthorized modification. In the Swissbit customized pre-boot phase authorization is performed to unlock access for a user or further boot.

Following authentication methods are available:

- PIN input by the user
- an presentation of an authorization dongle (requiring a separate Swissbit DP device) or
- authorization through a network server.

Please refer to the chapter about the various policies for details.

In the herein described setup all files and data in the boot partition are read only and cannot be modified. The root file system of the OS can only be read after authentication. Thus an authentication failure during boot will prevent the kernel from reading the OS root file system resulting in a boot failure.

3 Contents of the SDK

The Swissbit U-Boot SDK comes with U-Boot binaries and configuration files for Raspberry 1, 2 and 3 boards, sample U-Boot source code and managing applications to configure a Swissbit DP card. Prebuild U-Boot binaries are available for Raspberry Pi 1, 2 and 3 boards, managing applications for MS Windows (Windows 7 and higher). This chapter describes where to find the particular components.

The Swissbit U-Boot SDK is packed in the file SwissbitUBootSDK.zip. After unpacking in a directory, the SDK has this directory structure:

```
.
├─── Raspberry
│   └─── RPI1              | U-Boot binaries and configuration files for RPI 1
│   └─── RPI2              | U-Boot binaries for RPI 2
│   └─── RPI3              | U-Boot binaries for RPI 3
│       └─── RPI3B         | U-Boot configuration script for RPI 3 B
│       └─── RPI3BPlus     | U-Boot configuration script for RPI 3 B Plus
├─── Apps                  | Managing application for Swissbit DP card
│   └─── Windows
│       └─── bin           | Binary files for Windows
│           └─── platforms | Windows specific QT binary
├─── doc                   | Location of this document
├─── src                   | U-Boot source code (not part of this distribution)
│   └─── patchces          | Patches for U-Boot sources (not part of this distribution)
│           └─── 0718      | Patches for Raspberry PI builds (not part of this distribution)
```

3.1 U-Boot Binary Files

The U-Boot Binary can be found in the respective folders for the Raspberry Pi. Please note that the bootcode.bin should not be older as 30.07.2018

3.1.1 Raspberry

Raspberry U-Boot binary files are specific for the Raspberry models:

3.1.1.1 RPI 1

U-Boot binary: <sdkroot>\Raspberry\RPI1\u-bootRPI1.bin

Binary U-Boot boot script: <sdkroot>\Raspberry\ RPI1\boot.scr.uimg

U-Boot script <sdkroot>\Raspberry\ RPI1\boot.scr

3.1.1.2 RPI 2

U-Boot binary: <sdkroot>\Raspberry\ RPI2\u-bootRPI2.bin

Binary U-Boot boot script: <sdkroot>\Raspberry\ RPI2\boot.scr.uimg

U-Boot script <sdkroot>\Raspberry\ RPI2\boot.scr

3.1.1.3 RPI 3

U-Boot binary: <sdkroot>\Raspberry\ RPI3\u-bootRPI3.bin

The configuration scripts for the Raspberry Pi 3 depend on the version of the Rapberry Pi

3.1.1.3.1 RPI 3 B

Binary U-Boot boot script: <sdkroot>\Raspberry\ RPI3\B\boot.scr.uimg

U-Boot script <sdkroot>\Raspberry\ RPI3\B\boot.scr

3.1.1.3.2 RPI 3 B Plus

Binary U-Boot boot script: <sdkroot>\Raspberry\ RPI3\BPlus\boot.scr.uimg

U-Boot script <sdkroot>\Raspberry\ RPI3\BPlus\boot.scr

3.2 Application for Managing DP-Cards

Before using the Swissbit DP card it has to be configured under Windows with the Swissbit Device Manager.

Device Manager excecutable: <sdkroot>\Apps\Windows\bin\cardManager.exe

Device Manager CLI executable: <sdkroot>\Apps\Windows\bin\cardManagerCLI.exe

Device Manger install script : <sdkroot>\Apps\Windows\install.bat

3.3 Source Code

Swissbit provides one additional file to be added to the U-Boot sources. It implements the DP U-Boot command. Refer to chapter 5 for details of the build process.

Implementation of the dp command: <sdkroot>\src\dp.c

4 DP Card Configuration and U-Boot Installation

Before a Swissbit DP protection is active the following steps have to be executed in this order:

- Set Security Flags
- Set a Protection Policy
- Configure USB Authentication Dongle (only if USB policy is chosen)
- Set up network authentication server (only if NET policy is chosen)
- Install OS on Swissbit DP card (only for Raspberry Pi)
- Set Protection Profile
- Install U-Boot binary
- Activate Protection

4.1 DP Card Configuration

The DP card has to be configured before using the Swissbit Device Manger under Windows. The Device Manager can be found at <sdkroot>\Apps\Windows\bin\cardManager.exe. It can be started from that location or be installed permanently using the install script at at <sdkroot>\Apps\Windows\install.bat.

4.1.1 Preparing a DP Card

Configuring the DP card requires these steps: Setting Security Flags, Setting a Policy and Setting a Protection Profile. You have to perform the first two steps before you install the Operating System (in case of the Raspberry use-case)

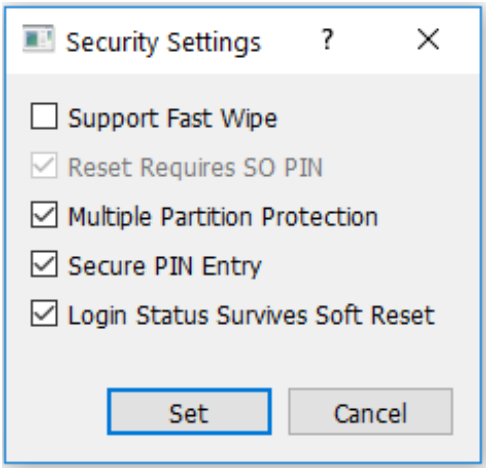
You must set a Protection Profile only after you have installed the OS because the Protection Profile depends on the partitioning scheme.

4.1.1.1 Setting Security Flags

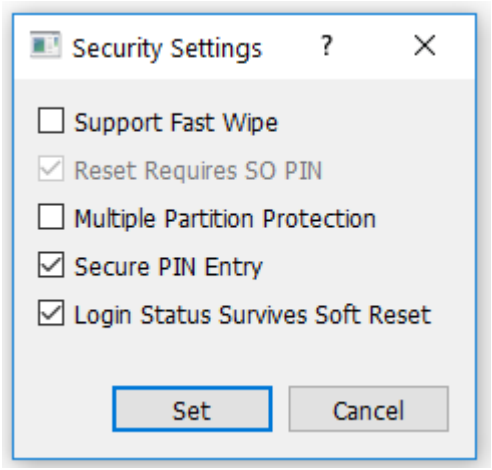
Security flags are set in Device Manager. Use the menu entry Manage->Security Settings. The Security Settings dialog opens. You must choose these settings:

- Support Fast Wipe: not checked
- Reset Requires SO PIN: checked
- Multiple Partition Protection: checked
- Secure PIN Entry: checked
- Login Status Survives Soft Reset: checked

Multiple Partition Protection has to checked for the OS integrity (Raspberry) use case.



Security Settings for the OS Integrity Protection use case (Raspberry PI with multiple partitions).

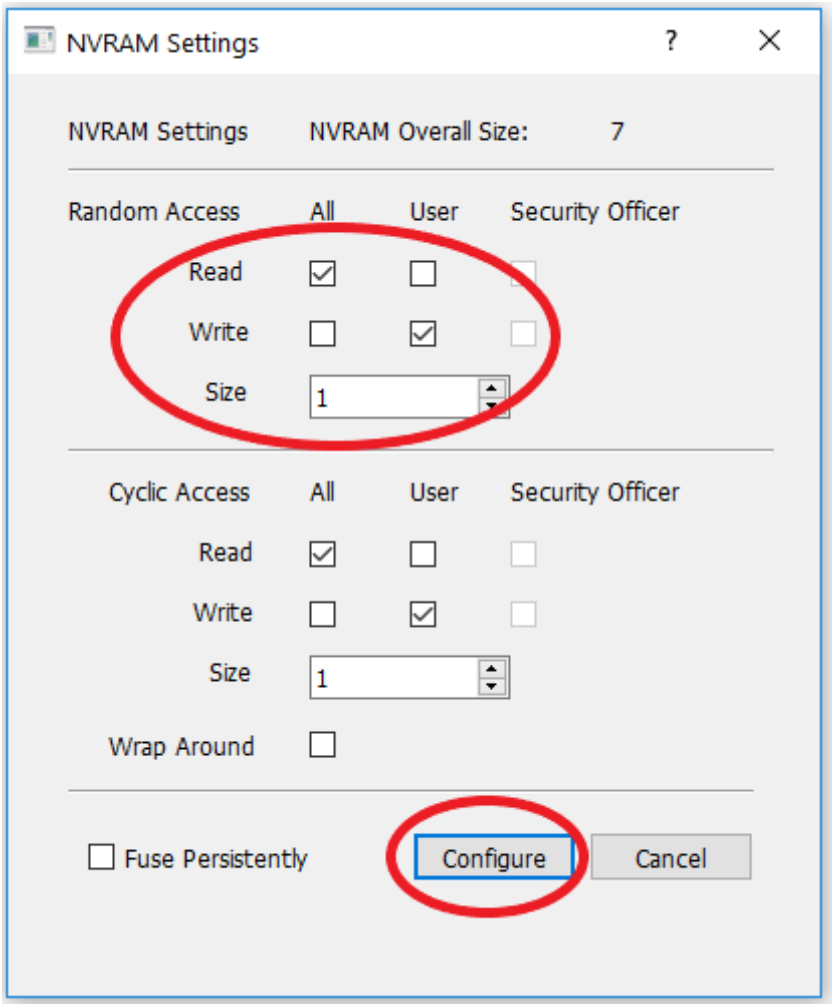


Security Settings for the Data Protection use case.

Click Set to confirm your choices. Afterwards you will have to close the Device Manager. Unplug the DP card and plug it in again.

4.1.1.2 Setting a Protection Policy

Swissbit U-Boot requires setting a protection policy used by U-Boot. There are three policies PIN, USB and NET. Policies are written to the first block of the random access NVRAM. This can be done with the Device Manager. Since the policy is written to the random access NVRAM, it must contain at least one block and have correct access rights. Configuring NVRAM is done in Device Manager menu entry NVRAM->Configure. In the NVRAM dialog select for both Size fields the value 1 and check the column for Read and Write access rights as shown below. Now you can set the policy itself. For the various policies refer to the subsections of this chapter.



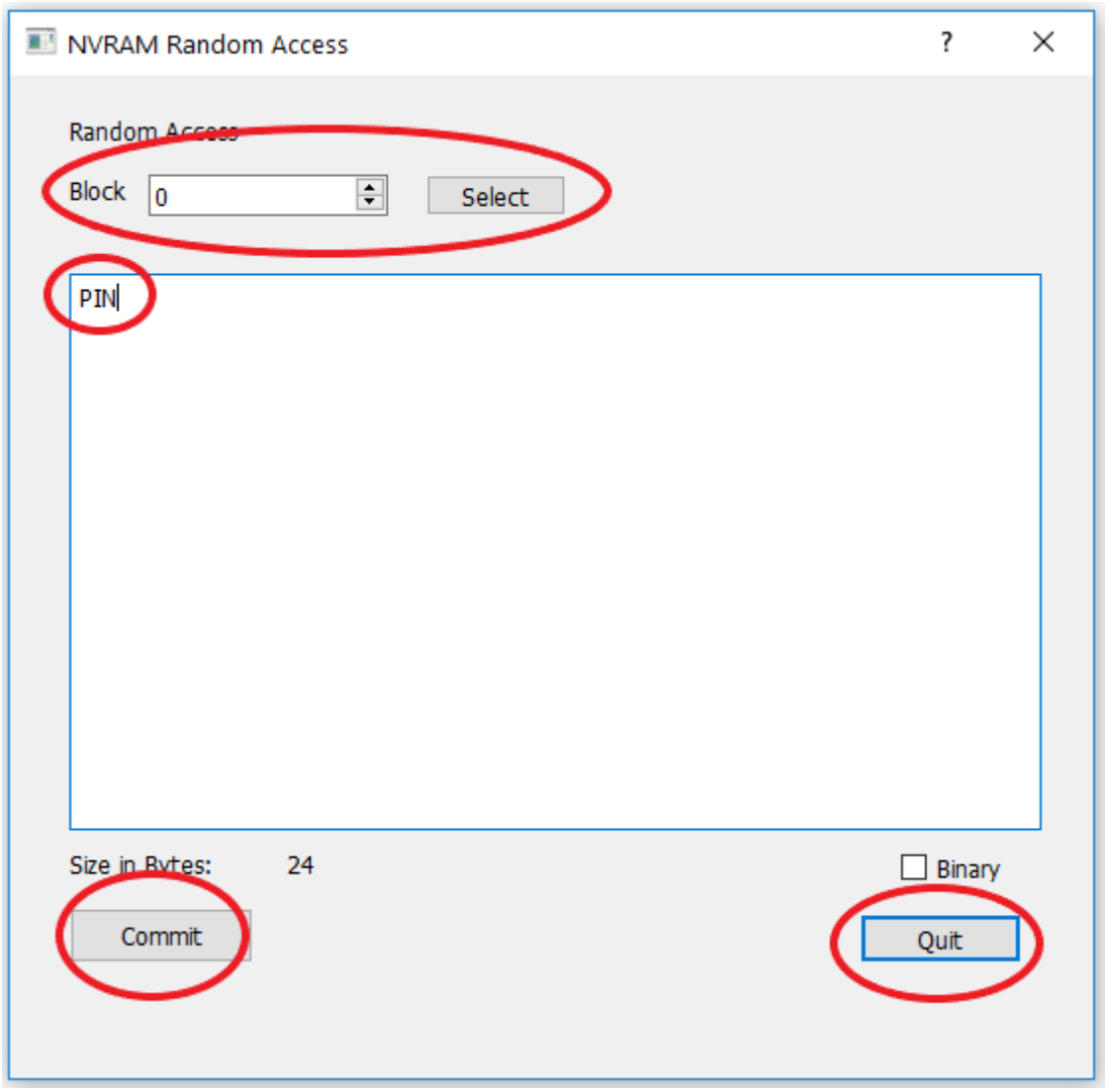
Configuring the NVRAM.

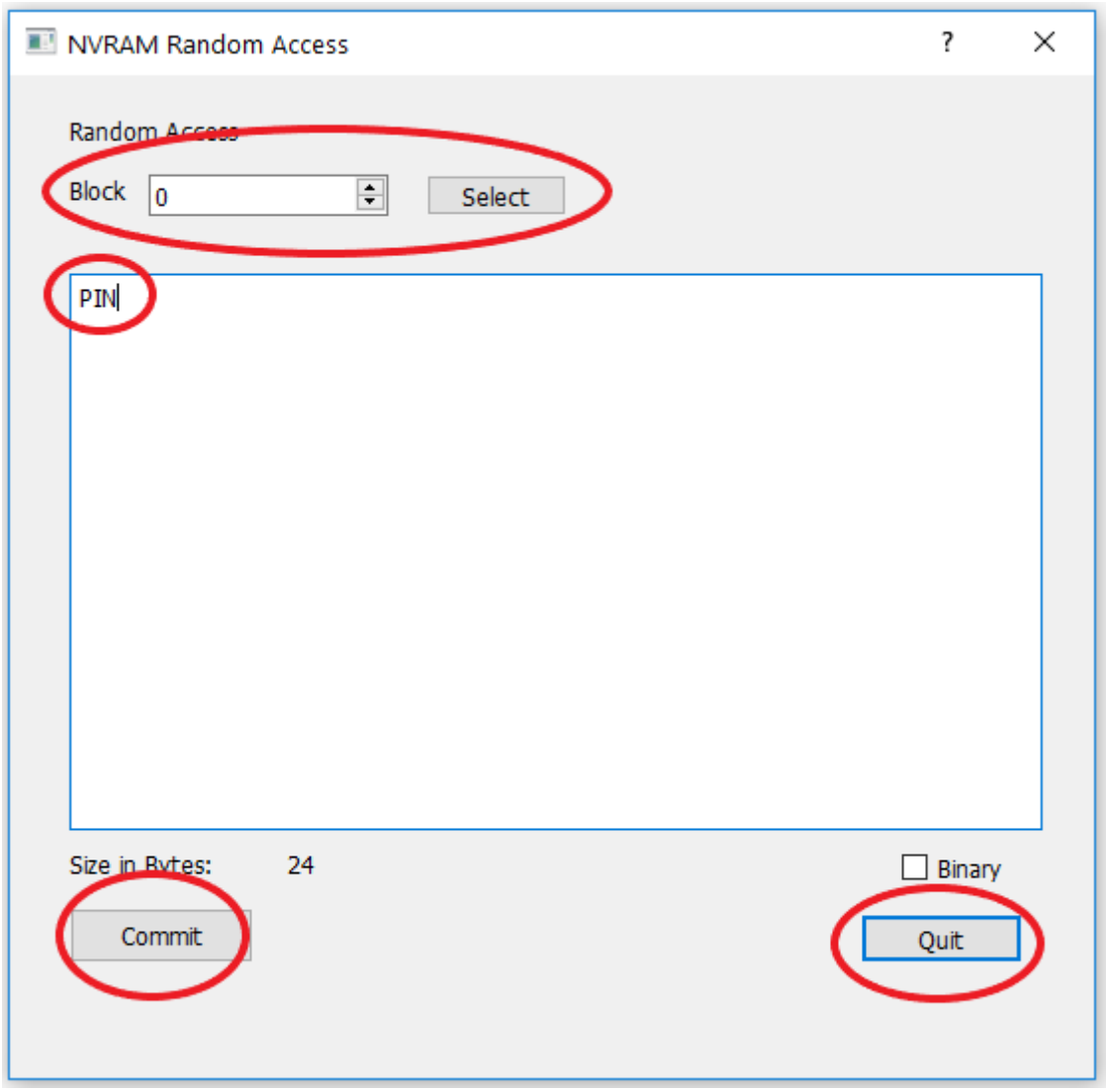
4.1.1.3 Setting U-Boot PIN-Policy

PIN policy means the user has to enter a PIN during the boot process to unlock the card.

PIN policy is set by writing the string PIN into the NVRAM. Use Device Manager menu entry NVRAM-> Read/Write Random Access Memory. In the NVRAM Random Access dialog enter 0 as the value for Block, Click Select and write PIN into the edit field. Write with Commit the policy to NVRAM. Leave the dialog with Quit.

Note: PIN policy should not be used on a BBB, because getting access to U-Boot output and providing an interface to enter a PIN on a BBB requires access to the BBB serial output and transferring it to a terminal on another machine.


NVRAM Random Access
Random Access
Block
0
Select
PIN
Size in Bytes:
24
Binary
Commit
Quit



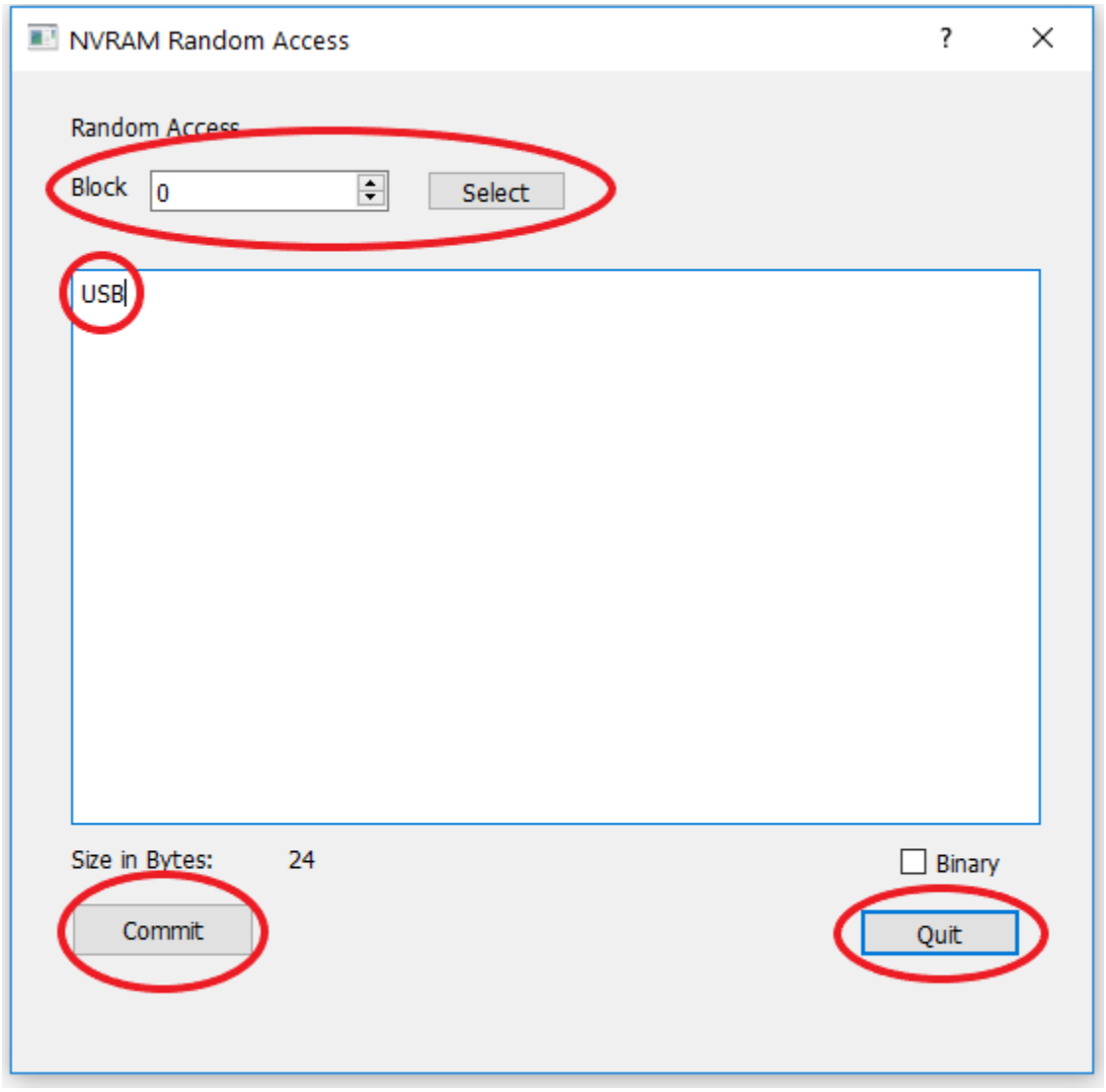
Setting a PIN policy

4.1.1.4 Setting USB-Policy

USB policy means there is an additional Swissbit DP device with CCID capabilities present in an USB slot at the board that is booted. This DP CCID device holds the unlock PIN in encrypted form and sends it at boot time to the U-Boot authentication function.

USB policy is set by writing the string USB into the NVRAM. Use Device Manager menu entry NVRAM->Read/Write Random Access Memory. In the NVRAM Random Access dialog enter 0 as the value for Block, Click Select and write USB into the edit field. Write with Commit the Policy to NVRAM.

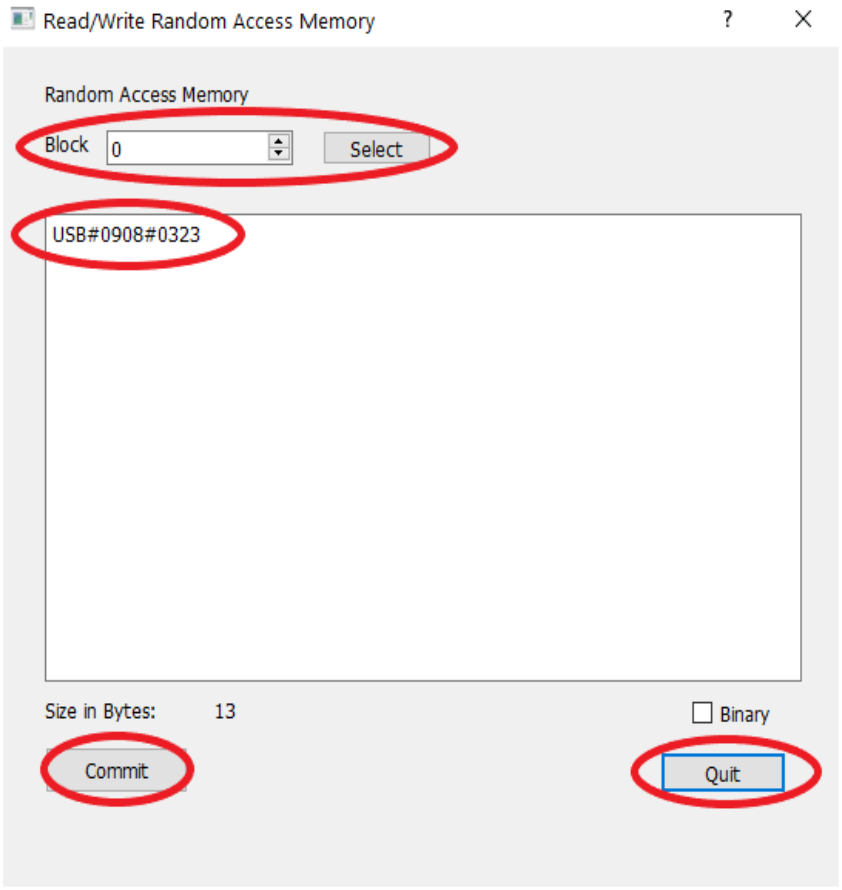
The additional DP CCID device has to be configured in Device Manager as described in the subsection of this paragraph.



Setting a USB policy.

4.1.1.4.1 Restricting the USB Authentication Dongle to Defined Devices

The USB Authentication Dongle can be restricted to match a defined USB vendor and product id. In this case vendor and product id have to be appended. The policy string in the NVRAM will take this format: USB#<vendorId>#<productId> . <vendorId> is a four-digit hexadecimal value designating the vendor id to match and <productId> is a four digit hexadecimal value designating the product id to match. Thus a sample policy might be USB#0908#0323 .



Setting a USB policy and restricting the USB Authentication Dongle to USB vendor id 0x0908 and product id 0x0323.

Restricting the USB Authentication Dongle to a match a USB vendor and product id is an optional feature.

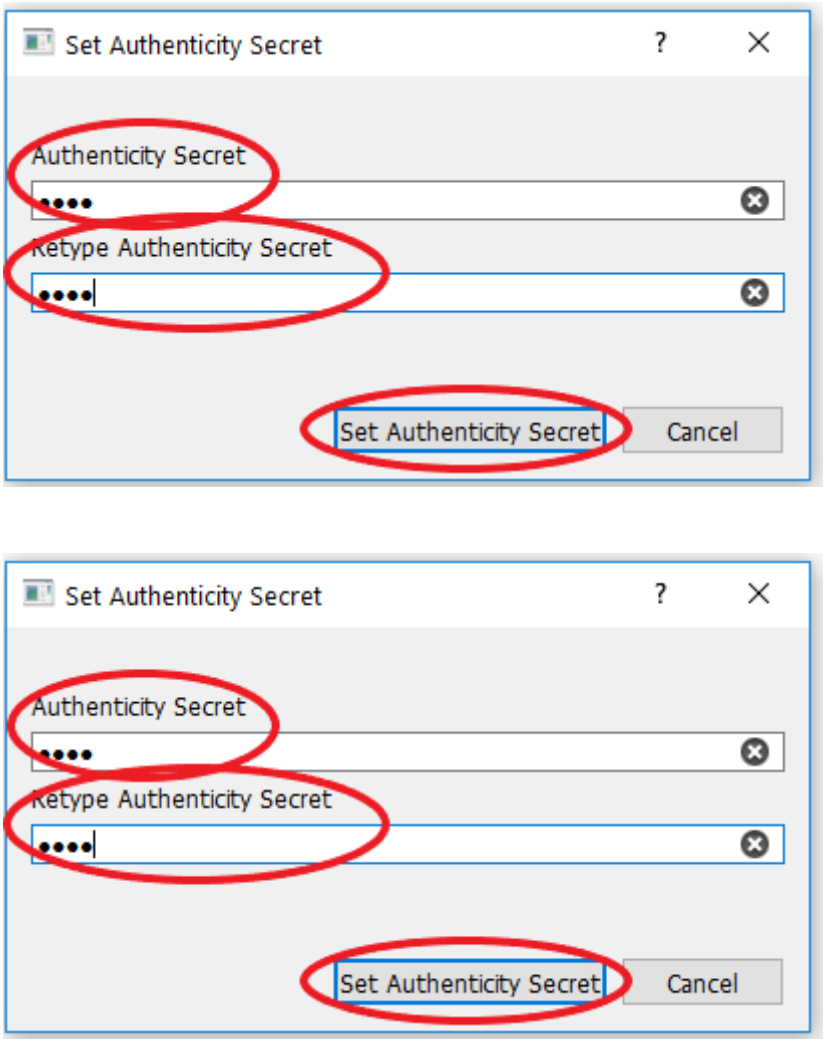
A list of USB vendor and product ids can be found here . The Linux command *lsusb* lists all USB devices in a system with their product and vendor ids.

4.1.1.4.2 Configuring the USB Authentication Dongle

Unplug the DP card you used to set the policy and plug in the additional DP device.

Use Device Manger menu entry Manage->Set Authenticity Secret. Enter a PIN as an Authenticity Secret and store it with set.

Note: Please remember the PIN as you need to set the same value as the Password in the Activation Dialog later on for the DP card.



Configuring the additional DP device as an authentication dongle when using a USB policy

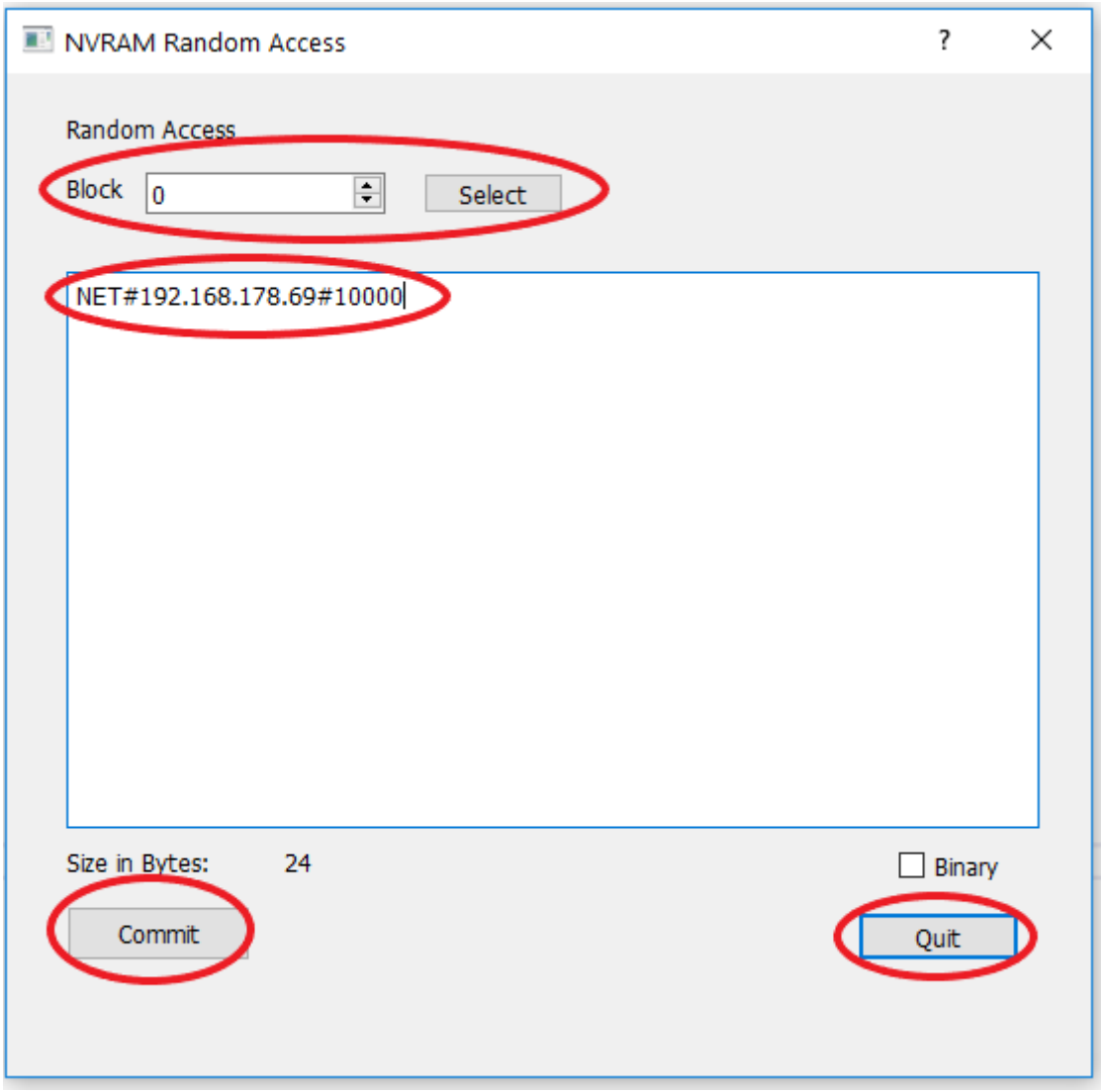
4.1.1.5 NET-Policy

NET policy means during boot U-Boot retrieves authentication information from an authentication server in the network. How to set up an authentication server is covered in the document Swissbit Authentication Server User Manual.

The NET policy has this format NET#<ipaddr>#<port>. <ipaddr> designates the IPv4 address or the name of the authentication server. <port> means the port the UDP port the server script is listening to. It should be set to 12375. Thus a properly formatted NET policy string would look like this:

NET#192.168.178.75#12375 indicating an authenticity server with the IP address 192.168.178.75 listening on port 12375.

NET policy is set by writing the policy string into the NVRAM. Use Device Manager menu entry NVRAM-> Read/Write Random Access Memory. In the Read/Write Random Access Memory dialog enter 0 as the value for Block, Click Select and write the NET policy string into the edit field. Write with Commit the Policy to NVRAM.



Setting a NET policy. Please use 12375 as the port address and above 10000 is just an example.

4.1.2 Installing the OS

This section applies to the Rapberry Pi only because the BBB boots from the internal eMMC. Please download the latest Raspbian image from:

https://www.raspberrypi.org/downloads/raspbian/

and follow the installation guide at

https://www.raspberrypi.org/documentation/installation/installing-images/windows.md

using Etcher.

After you installed the OS verify you can boot your Raspberry PI from this card and apply all updates.

4.1.3 Setting the Protection Profile

A Protection Profile has to be set only in case Multiple Partition Protection has been selected when choosing.

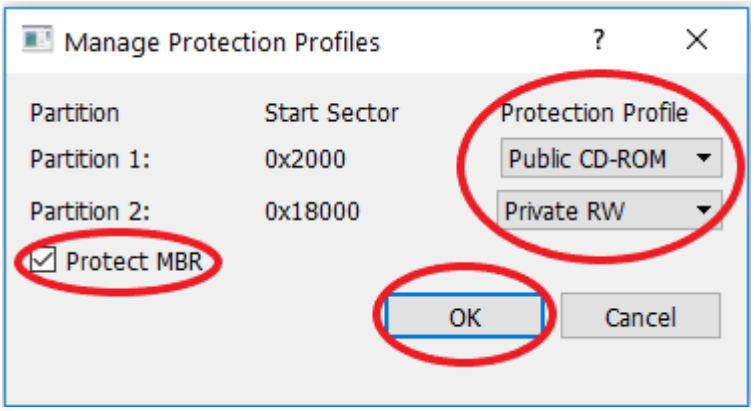
Note: If Multiple Partition Support has not been activated (see section Setting Security Flags) this step cannot be applied since the protection profile is applied implicitly.

The Protection Profile determines which kind of protection is in force after security has been activated on the card. Protection profiles are aligned to partition boundaries. Each partition can have exactly one profile type assigned.

It is strongly recommended to check "Protect MBR". With this setting the card's MBR can be read but not be modified. Even in unlocked state the MBR is immutable and the card cannot be repartitioned.

Note: Repartitioning of the MBR is possible by the Admin and requires deactivation of the card's security first. Refer to the section Deactivating Protection on a DP Card in the Appendix.

The OS integrity use case (e.g. for the Raspberry Pi) assumes two partitions. A boot partition that must be readable at any time and a root file system partition that must be accessible only after authentication. This means setting the first partition to the Protection Profile Type *Public CD-ROM* and the second partition to the type *Private RW*. Setting a protection profile is done in Device Manager menu entry Manage->Manage Protection Profiles. In the Manage Protection Profiles dialog, please select for the first partition *Public CD-ROM* and for the second *Private RW*. Please check also *Protect MBR*.



Setting Protection Profiles for a Raspberry Pi installation.

4.2 U-Boot Installation

Format and destination location of the U-Boot binaries depends on the board U-Boot is intended to run. On the Raspberry Pi U-Boot binary is installed on the Swissbit DP card, on the BBB the U-Boot binaries are installed in the internal eMMC memory. After you have installed U-Boot verify you can boot your board with U-Boot, but with security still not activated.

4.2.1 Raspberry

For Raspberry the U-Boot files required for the Swissbit U-Boot implementation consist of a U-Boot binary and a binary U-Boot configuration script.

4.2.1.1 Raspberry PI 1

Copy the file <sdkroot>\Raspberry\RPI1\u-bootRPI1.bin to \boot\ u-bootRPI1.bin on the first partition.
Copy the file <sdkroot>\Raspberry\ RPI1\boot.scr.uimg to \boot\boot.scr.uimg on the first partition

Add the line

kernel=u-bootRP1.bin

to \boot\config.txt on the first partition

4.2.1.2 Raspberry PI 2

Copy the file <sdkroot>\Raspberry\RPI2\u-bootRPI2.bin to \boot\ u-bootRPI2.bin on the first partition.
Copy the file <sdkroot>\Raspberry\ RPI2\boot.scr.uimg to \boot\boot.scr.uimg on the first partition

Add the line

kernel=u-bootRP2.bin

to \boot\config.txt on the first partition

4.2.1.3 Raspberry PI 3

Copy the file <sdkroot>\Raspberry\RPI3\u-bootRPI3.bin to \boot\ u-bootRPI3.bin on the first partition.

Add the line

kernel=u-bootRPI3.bin

to \boot\config.txt on the first partition

The U-Boot script file depends on the hardware version of the Raspberry PI 3. Refer to the subsections for your particular version

4.2.1.3.1 Raspberry PI 3 B

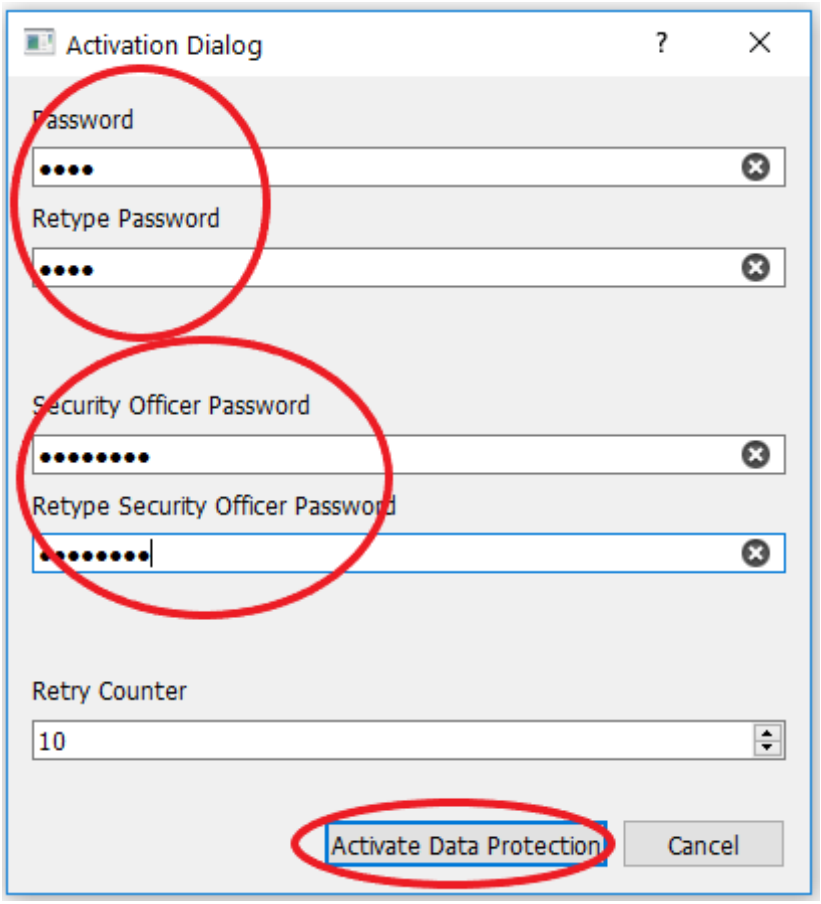
Copy the file <sdkroot>\Raspberry\RPI3\B\boot.scr.uimg to the boot partition. (first partition)

4.2.1.3.2 Raspberry PI 3 B Plus

Copy the file <sdkroot>\Raspberry\RPI1\BPlus\boot.scr.uimg to the boot partition. (first partition)

4.3 Activating DP Card Data Protection

When you have verified you can boot with U-Boot installed and the authentication server is running, you can activate the protection of the DP card. This is done in the Device Manager menu entry Manage->Activate Data Protection. In the Activation Dialog set a password (ie. a user PIN) and a Security Officer Password. Please make sure Password is at least four characters long and SO Password at least eight characters. If you have chosen USB policy, the password must match the authenticity secret of the authentication dongle. Click Activate Data Protection to activate protection.


Activation Dialog
Password
Retype Password
Security Officer Password
Retype Security Officer Password
Retry Counter
10
Activate Data Protection
Cancel

5 Building U-Boot Binaries from Source Code

Swissbit provides an extension to the U-Boot command line adding the dp command. It is implemented in a single source code file and can be simply added to the mainline U-Boot sources. This chapter describes how to build the U-Boot binaries from the sources.

This documentation assumes a cross-compile build on a x86-64 Ubuntu 18.04 machine.

5.1 U-Boot Source Code

Building U-Boot binaries from the sources requires these steps in the following order:

- Checking preconditions on the build machine
- Retrieving U-Boot source code and patches
- Adding Swissbit modifications to the U-Boot sources
- Building from the source code

5.1.1 Preconditions for Building U-Boot from the Sources

These tools have to be installed on the build machine:

- git
- gcc
- Cross-compile tool chain
- U-Boot tools
- bison
- flex

5.1.1.1 git

Install:

sudo apt install git

Check installation:

git --version

5.1.1.2 gcc

Install:

sudo apt install gcc

Check installation:

gcc --version

5.1.1.3 Cross-Compile Tool Chain

Install:

*sudo apt-get install gcc-arm**

Check installation:

arm-linux-gnueabi-gcc --version

5.1.1.4 U-Boot Tools

Install:

sudo apt install u-boot-tools

Check installation:

mkimage –V

5.1.1.5 bison

Install:

sudo apt install bison

Check installation:

bison --version

5.1.1.6 flex

Install:

sudo apt install flex

Check installation:

flex --version

5.1.2 Retrieving Mainline U-Boot Source Code from git Repository

U-Boot source code is retrieved from the main U-Boot repository. Depending whether you build for Raspberry Pi you will have to checkout different release versions. Please make sure it is no later than 30.07.2018

5.1.2.1 Retrieving Sources for Raspberry Pi

git clone git://git.denx.de/u-boot.git

cd u-boot

git checkout v2018.07 -b v2018.07

5.1.3 Adding Patches

5.1.3.1 Adding Raspberry PI Patches

Swissbit patches for the Raspberry PI are located in the directory <sdkroot>/src/patches/0718. Copy all files from that directory to the root of the U-Boot build directory and apply the patches in this order:

patch -p1 < swissbit2.patch

patch -p1 < swissbit3.patch

patch -p1 < swissbit4.patch

5.1.4 Applying Swissbit Modifications

Copy <sdkroot>/src/dp.c to u-boot/cmd/dp.c

5.2 Building from the sources

Regardless of which board will be the build target, these environment variables have to be set:

export ARCH=arm

export CROSS_COMPILE=arm-linux-gnueabi-

5.2.1 Raspberry Pi

The Raspberry PI U-Boot binary will be u-boot.bin. For convenience the binary file should be renamed to know to which Raspberry Pi model it applies.

When prompted for input during the build for input, enter 0.

Additionally U-Boot for the Raspberry PI needs a compiled configuration script boot.scr.uimg. It is built from boot.scr in the respective subfolders.

5.2.1.1 RPI 1

5.2.1.1.1 U-Boot Binary

make distclean

make rpi_defconfig

make V=1

mv u-boot.bin u-bootRPI1.bin

5.2.1.1.2 Configuration Script

Use boot.scr from <sdkroot>\Raspberry\ RPI1\boot.scr

Build compiled configuration script:

mkimage -A arm -O linux -T script -C none -n boot.scr -d boot.scr boot.scr.uimg

5.2.1.2 RPI 2

5.2.1.2.1 U-Boot Binary

make distclean

make rpi_2_defconfig

make V=1

mv u-boot.bin u-bootRPI1.bin

5.2.1.2.2 Configuration Script

Use boot.scr from <sdkroot>\Raspberry\ RPI2\boot.scr

Build compiled configuration script:

mkimage -A arm -O linux -T script -C none -n boot.scr -d boot.scr boot.scr.uimg

5.2.1.3 RPI 3

5.2.1.3.1 U-Boot Binary

make distclean

make rpi_3_32b_defconfig

make V=1

mv u-boot.bin u-bootRPI3.bin

5.2.1.3.2 Configuration Script

There are various Raspberry PI 3 versions requiring different configuration scripts: RPI 3 B and RPI 3 B Plus:

5.2.1.3.2.1 Configuration Script for RPI 3 B

Use boot.scr from <sdkroot>\Raspberry\ RPI3\B\boot.scr

Build compiled configuration script:

mkimage -A arm -O linux -T script -C none -n boot.scr -d boot.scr boot.scr.uimg

5.2.1.3.2.2 Configuration Script for RPI 3 B Plus

Use boot.scr from <sdkroot>\Raspberry\ RPI3\BPlus\boot.scr

Build compiled configuration script:

mkimage -A arm -O linux -T script -C none -n boot.scr -d boot.scr boot.scr.uimg

6 Reference Material

6.1 Swissbit

Swissbit Net Policy Console User Manual

6.2 U-Boot

https://www.denx.de/wiki/view/DULG/UBoot

http://www.denx.de/wiki/DULG/Faq

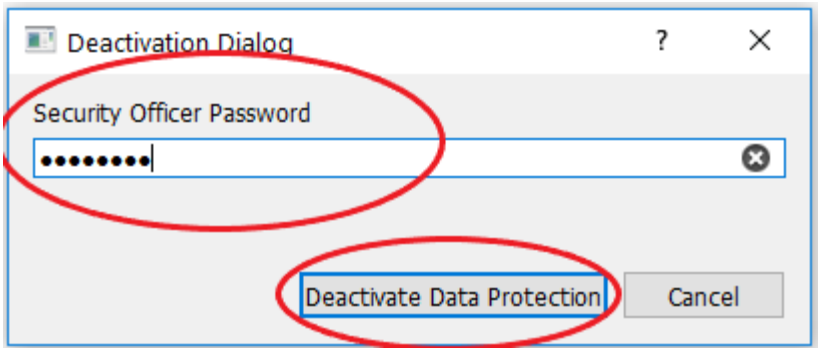
6.3 Raspberry Pi

https://elinux.org/RPi_U-Boot

7 Apendix

7.1 Deactivating Protection on a DP Card

If you want to make changes to the boot partition of the Swissbit DP card, you can do this only when the card is in transparent mode. If protection has been activated you will need to deactivate using Device Manger menu entry Manage->Deactivate Data Protection. In the Deactivation Dialog enter the Security Officer Password (ie. SO PIN) and click Deactivate Data Protection.



Deactivating Data Protection