EFM8 Bootloader

Release 0.0.2

Nov 05, 2021

Contents:

Talks to the factory Bootloader on EFM8 to flash firmware. The protocol is documented in AN945.

CHAPTER 1

Installation

Communication is over USB-HID. This is implemented via the hidapi pthon wrapper for the hidapi native library.

On linux you can use udev to grant access:

```
echo 'SUBSYSTEM=="usb", ATTRS{idVendor}=="10c4", MODE="0666"' | sudo tee /etc/udev/
↪rules.d/70-silabs.rules
udevadm trigger
```

Then install some native prerequisites:

```
sudo apt install libusb-1.0-0-dev libudev-dev python-dev
```

Then *pip* install:

```
pip install efm8
```

CHAPTER 2

efm8

Flash via AN945: EFM8 Factory Bootloader HID.

```
usage: efm8 [-h] [-p PRODUCT] [-s SERIAL] firmware
```

firmware
Intel Hex format file to flash

-h, --help
show this help message and exit

-p <product>, **--product** <product>
USB Product ID of device to program

-s <serial>, **--serial** <serial>
Serial number of device to program

Flash via AN945: EFM8 Factory Bootloader HID.

exception efm8.**BadChecksum**
Checksum mismatch.

exception efm8.**BadResponse**
Command not confirmed.

exception efm8.**Unsupported**
Input file not understood.

efm8.**crc**(*data*)
CITT-16, XModem.

efm8.**create_frame**(*cmd*, *data*)
Bootloader frames start with '$', 1 byte length, 1 byte command, x bytes data.

efm8.**flash**(*manufacturer*, *product*, *serial*, *frames*)
Send bootloader frames over HID, and check confirmations.

efm8.**read_flash**(*manufacturer*, *product*, *serial*, *length*)
Exploit CRC to read back firmware.

efm8.**read_intel_hex**(*filename*)
Read simple Intel format Hex files into byte array.

efm8.**to_frames**(*data*, *checksum=True*, *run=True*)
Convert firmware byte array into sequence of bootloader frames.

efm8.**toaddr**(*addr*)
Split a 16bit address into two bytes (dosn't check it is a 16bit address ;-).

efm8.**twos_complement**(*input_value*, *num_bits=8*)
Calculate unsigned int which binary matches the two's complement of the input.

efm8.**write_hex**(*buf*, *filename*)
Write an Intel Format Hex file.

CHAPTER 3

efm8_read

Flash via AN945: EFM8 Factory Bootloader HID.

```
usage: efm8_read [-h] [-p PRODUCT] [-s SERIAL] [-l LENGTH] firmware
```

firmware
Intel Hex format file to flash

-h, --help
show this help message and exit

-p <product>, **--product** <product>
USB Product ID of device to program

-s <serial>, **--serial** <serial>
Serial number of device to program

-l <length>, **--length** <length>
Length to read

CHAPTER 4

u2fzero

Extra utils for U2F-Zero devices.

```
usage: u2fzero [-h] [-p PRODUCT] [-s SERIAL] firmware
```

firmware
Intel Hex format file to flash

-h, --help
show this help message and exit

-p <product>, **--product** <product>
USB Product ID of device to program

-s <serial>, **--serial** <serial>
Serial number of device to program

Extra utils for U2F-Zero devices.

efm8.u2fzero.**main**()
Command line.

efm8.u2fzero.**reset**(*manufacturer*, *product*, *serial*)
Send zeroU2F jump to bootloader, trigger the host to see the device change.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

e

Index

Symbols

B

C

E

F

M

R

T

U

W