

Google Summer of Code 2015

End User Flash Tool

Building

1. Install required software:

- edid-decode package
- packages required by flashrom:
http://flashrom.org/Downloads#Installation_from_source
- packages required by coreboot:
http://www.coreboot.org/Build_HOWTO#Requirements
- Qt >= 4.8.6: <https://download.qt.io/archive/qt/4.8/4.8.6/>
- doxygen package (optional)

2. Download source files of application and external tools:

- git clone https://github.com/lukaszdmittrowski/coreboot_flashtool.git
- git clone <https://github.com/lukaszdmittrowski/flashrom.git>
- git clone <https://github.com/lukaszdmittrowski/libcbfstool.git>
- git clone <https://github.com/lukaszdmittrowski/libbiosext.git>
- git clone <http://review.coreboot.org/p/coreboot>

Note: All directories should be on the same level, except coreboot directory which can be either moved to coreboot_flashtool directory (app searches there for coreboot by default) or its path can be selected in application preferences

3. Go to coreboot folder:

- git submodule update -init -checkout
- make crossgcc

4. Go to flashrom folder:

- `git checkout libflashrom`
- `make libflashrom.a`
- `cp libflashrom.a libflashrom.h ../coreboot_flashtool/libflashrom`

5. Go to libcbfstool folder:

- `./build_libcbfstool.sh`

6. Go to libbiosextract folder:

- `./build bios_extract.sh`

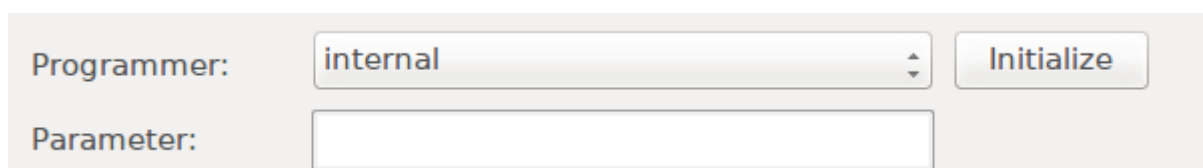
7. Go to coreboot_flashtool folder:

- `qmake`
- `make`
- `doxygen Doxyfile` (optional)

Using application

Please run application as root - most functionalities will not work without it

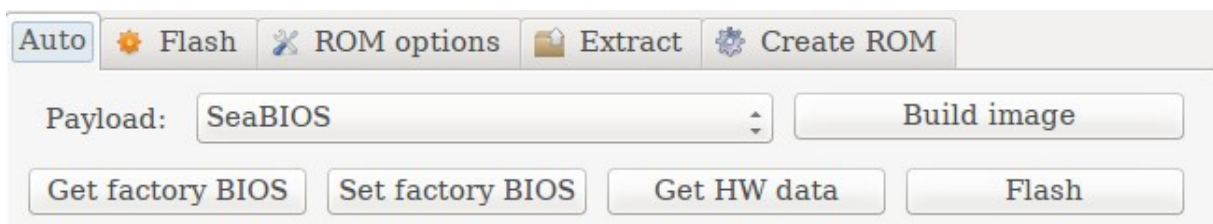
In top area of main window we can select programmer from combo box and provide its parameters:



The screenshot shows a graphical user interface with a light gray background. On the left, the label "Programmer:" is followed by a dropdown menu containing the word "internal". To the right of the dropdown is a small up/down arrow icon. Below the dropdown is a text input field labeled "Parameter:". To the right of the input field is a button labeled "Initialize".

Auto tab:

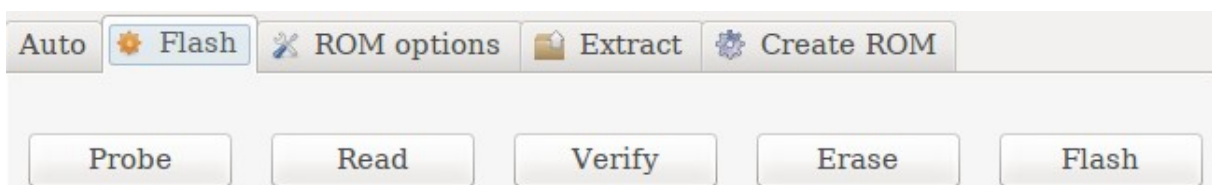
- **Get factory BIOS:** dumps factory BIOS
- **Set factory BIOS:** used to select factory BIOS file
- **Get HW data:** - saves output of `lspci -nn, /sys/class/drm/card0-LVDS-1/edid | edid-decode, dmidecode -t 2` and dumps VGABIOS from memory, also creates `hardware_data.tar` file with all listed files
- **Payload combobox:** used to select a payload added to coreboot image
- **Build image:** after selecting file with gathered hardware data it checks if working configuration for this particular hardware set is known, if yes, it checks if building an image needs additional option roms, extracts it from factory BIOS and checks if these option roms are correct (by comparing hashes), then it builds image with use of coreboot
- **Flash:** flashes previously built image to chip



There are also tabs for more experienced users. These tabs are actually just GUI for tools: `flashrom`, `cbfs_tool` and `bios_extract`.

Flash tab (flashrom):

- **Probe:** probes for all known chips, if multiple chips are found, outputs dialog popup with probed chips
- **Read:** saves chip content to specified directory
- **Verify:** checks if chip contents is the same as specified ROM file
- **Erase:** erases chip
- **Flash:** flashes chip with specified ROM file



Rom options tab (cbfs_tool):

- **Select:** loads specified ROM file with CBFS file system and show its content
- **+:** adds particular component to ROM file with name and type specified
- **-:** removes particular component from ROM file

The screenshot shows the 'ROM options' tab of a BIOS flashing utility. At the top, there are five tabs: 'Auto', 'Flash', 'ROM options' (which is active), 'Extract', and 'Create ROM'. Below the tabs, there is a 'ROM:' label followed by a 'Select' button and two buttons with '+' and '-' symbols. A table with five columns (Index, Name, Offset, Type, Size) lists several components. The first four rows are visible: 1. cmos.default (Offset 0x0, Type cmos_default, Size 256), 2. cmos_layout.bin (Offset 0x140, Type cmos_layout, Size 1824), 3. pci8086,27a2.rom (Offset 0x8c0, Type optionrom, Size 32768), and 4. fallback/dsdt.aml (Offset 0x8900, Type raw, Size 12037). A fifth row is partially visible. Below the table, a text box displays summary information: 'coreboot_secure_noint.rom: 2048 kB, bootblocksize 984, romsize 2097152, offset 0x0 alignment: 64 bytes, architecture: x86'.

	Name	Offset	Type	Size
1	cmos.default	0x0	cmos_default	256
2	cmos_layout.bin	0x140	cmos_layout	1824
3	pci8086,27a2.rom	0x8c0	optionrom	32768
4	fallback/dsdt.aml	0x8900	raw	12037
5	etc/acpi/keyboard	0xb840	raw	0

coreboot_secure_noint.rom: 2048 kB, bootblocksize 984, romsize 2097152, offset 0x0 alignment: 64 bytes, architecture: x86

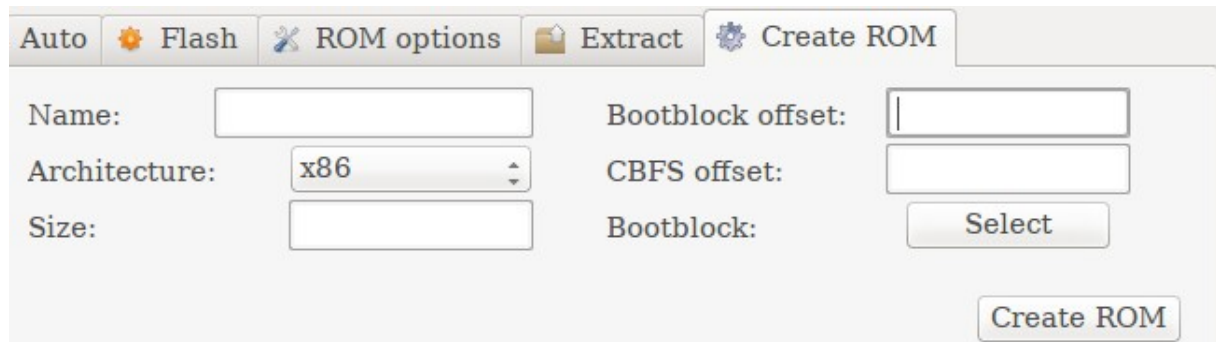
Extract tab (bios_extract):

- **Select ROM:** used to specify ROM file to be extracted
- **Select output dir:** used to specify directory for extracted files
- **Extract:** extracts ROM contents

The screenshot shows the 'Extract' tab of the same BIOS flashing utility. The top tabs are 'Auto', 'Flash', 'ROM options', 'Extract' (which is active), and 'Create ROM'. Below the tabs, there are two labels: 'ROM:' and 'Output dir:', each followed by a 'Select' button. On the right side of the tab, there is an 'Extract' button.

Create ROM (cbfs_tool):

- **Create ROM:** creates ROM according to specified options



The screenshot shows a software interface with five tabs: 'Auto', 'Flash', 'ROM options', 'Extract', and 'Create ROM'. The 'Create ROM' tab is selected. Below the tabs, there are several input fields and buttons. On the left, there are three rows: 'Name:' with a text input field, 'Architecture:' with a dropdown menu showing 'x86', and 'Size:' with a text input field. On the right, there are two rows: 'Bootblock offset:' with a text input field, and 'CBFS offset:' with a text input field. Below these, there is a 'Bootblock:' label followed by a 'Select' button. At the bottom right, there is a 'Create ROM' button.

Adding working configurations

To add more working configurations:

- hardware_data.tar file created with 'Get HW data' button
- short description of how you normally proceed with building image for your system (important or unusual steps)
- are additional option roms or other kinds of action needed? if it needs factory VGABIOS, which one is working - extracted from memory or extracted from factory BIOS? please also attach needed files
- .config file from coreboot for your configuration

If you would like to help with adding more working configurations please send this data to lukasz.dmitrowski@gmail.com. Then I will be able to add such configuration. Thanks in advance!