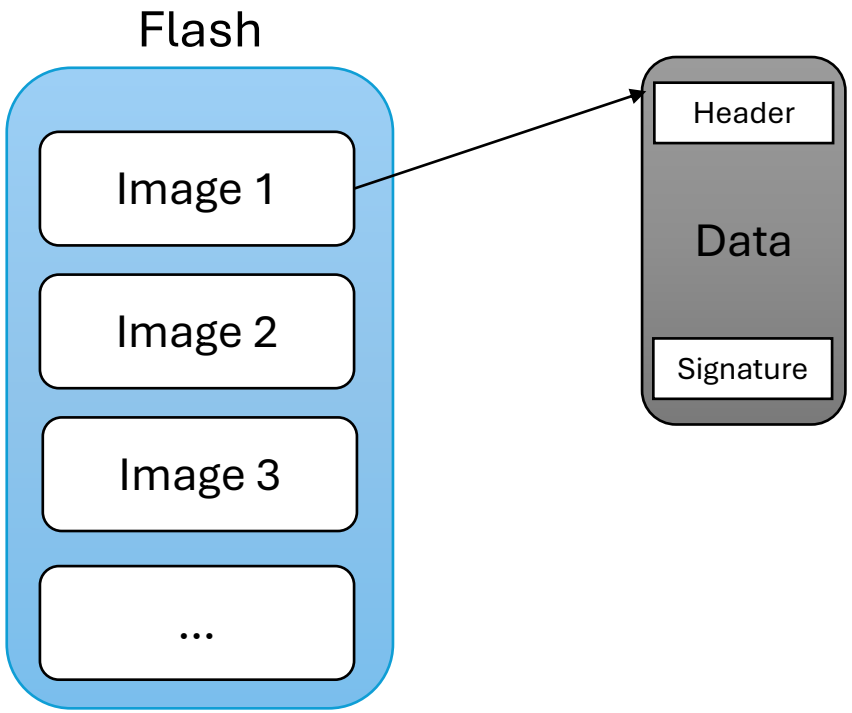
Single Signature Check for Multi Image

Sadik Ozer

MCUboot Multi Image Feature

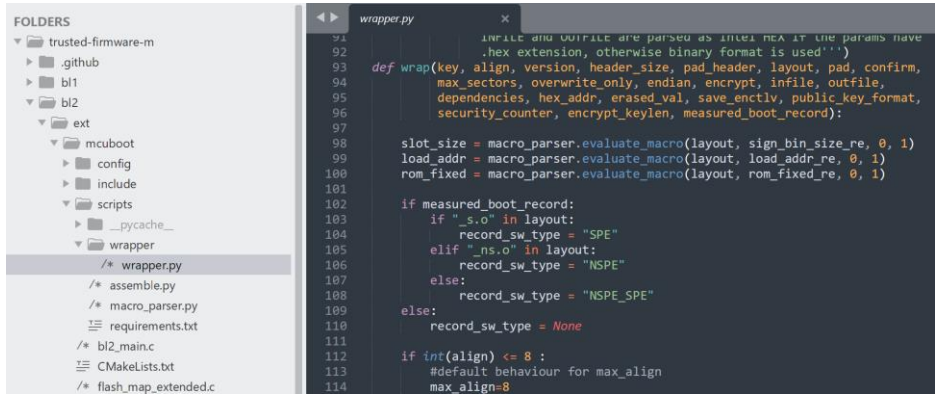
- MCUboot designed to support multiple images
  - Each image process independently and individually.
  - Each image has it is header & footer sections and able to be updated independently.
  - **MCUBOOT_IMAGE_NUMBER** macro is used to set number of image.

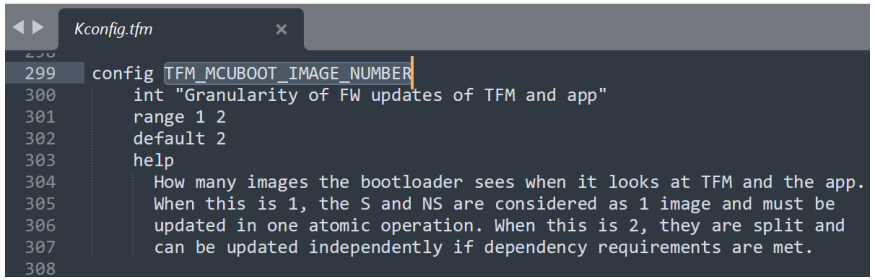



```
243  /** Private state maintained during boot. */
244  struct boot_loader_state {
245      struct {
246          struct image_header hdr;
247          const struct flash_area *area;
248          boot_sector_t *sectors;
249          uint32_t num_sectors;
250      } imgs[BOOT_IMAGE_NUMBER][BOOT_NUM_SLOTS];
251
```

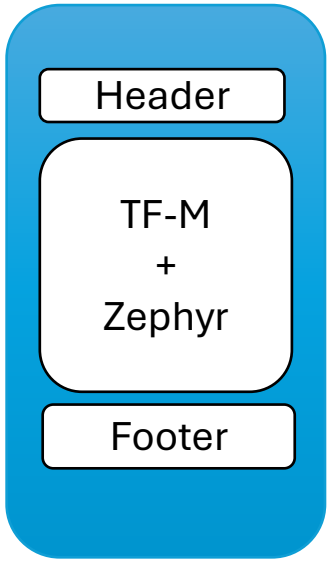
Multi Image with TFM

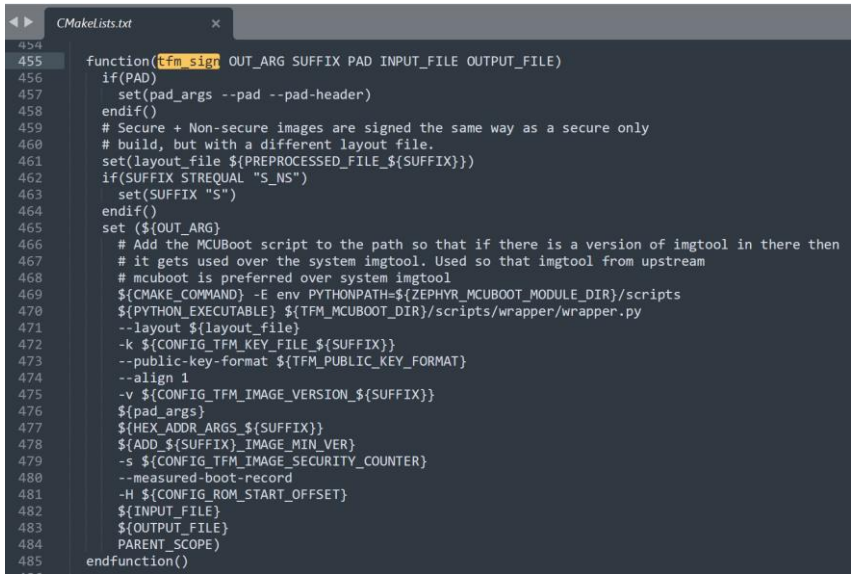
- TFM allow concatenate zephyr and tfm images and process them as a single image.
- Requires to set **CONFIG_TFM_MCUBOOT_IMAGE_NUMBER**=1
- Single signature check.
- There is wrap.py script that generate signature for single image





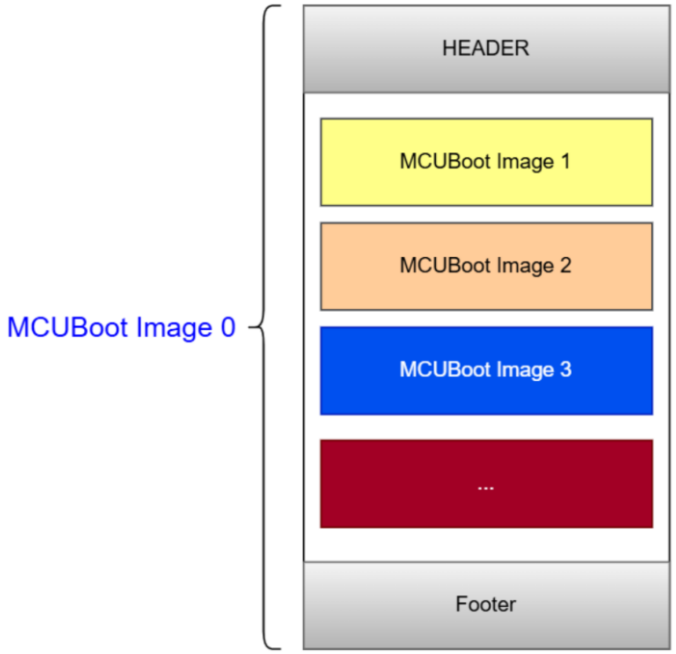
Single Image

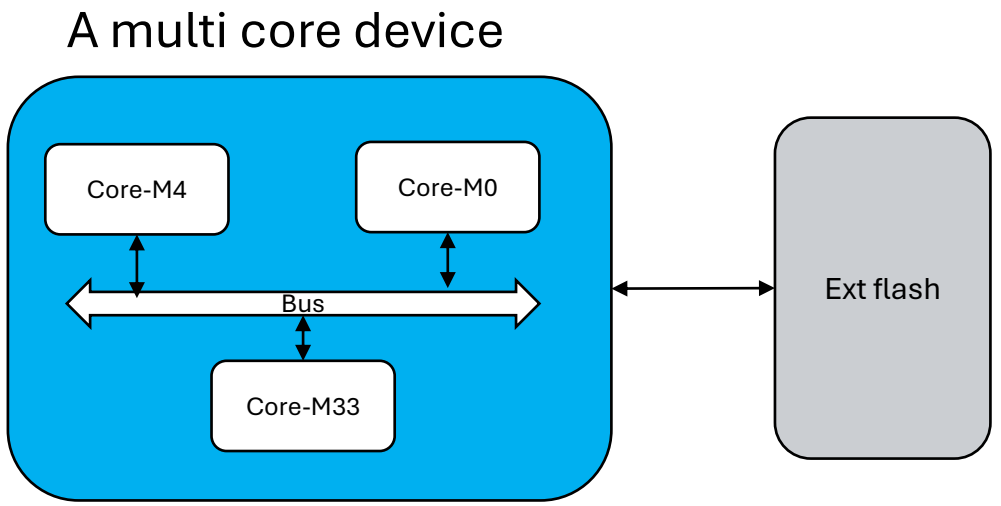




Proposal

- Reason: Multi core system may not need multiple signature check due to some boot time restrictions.
- Add a python script to generate image individually. Similar to TF-M solution.
- Then concatenate them and generate main image.
- MCUboot will validate main image and copy sub images to the target address.
- https://github.com/mcu-tools/mcuboot/pull/2465





Script to combine images

- As TF-M solution (wrapper.py) ***combine_images.py*** script added to combines images.
- Added ***combine_images.yaml*** file which defines each images input parameter
- https://github.com/mcu-tools/mcuboot/pull/2465
- *python scripts/combine_images.py --config <USER_YAML_FILE> --imgtool imgtool –output <OUT_FOLDER>*

53 scripts/combine_images.yaml

```
+
+ combined_app_pack:
+     outputfile: combined
+
+     header:
+         private_signing_key: ../root-rsa-2048.pem
+         header-size: 0x400
+         align: 4
+         load-addr: 0x20080000
+         pad-header: yes
+         version: 1.0.0
+         slot-size: 0x40000
+
+     image1_pack:
+         outputfile: image1
+         header:
+             private_signing_key: ../root-rsa-2048.pem
+             header-size: 0x400
+             align: 4
+             load-addr: 0x20010000
+             pad-header: yes
+             version: 1.0.0
+             slot-size: 0x10000
+         image: image1.bin
+
+     image2_pack:
+         outputfile: image2
+         header:
+             private_signing_key: ../root-rsa-2048.pem
+             header-size: 0x400
+             align: 4
+             load-addr: 0x20020000
+             pad-header: yes
+             version: 1.0.0
+             slot-size: 0x10000
+         image: image2.bin
```

178 scripts/combine_images.py

```
+ def main():
+     global img_tool
+     global output_dir
+     global config_path
+
+     parser = argparse.ArgumentParser(description="Create application package", allow_abbrev=False)
+     parser.add_argument('--config', help="The path to config yaml file", required=True)
+     parser.add_argument('--imgtool', help="The path to ImgTool", required=True)
+     parser.add_argument('--output', help="Output directory", required=True)
+
+     args = parser.parse_args()
```

Thank You!

Analog Devices Non-Confidential Information. ©2021 Analog Devices, Inc. All rights reserved.