Dual-bank and single-bank updates

To safely perform a Device Firmware Update, the new firmware image is not copied to the final location in memory until it has been validated. This ensures that only complete and valid images are activated. If an error occurs during the transfer, the firmware is not updated and the old firmware is still available.

This process of storing the received firmware in free memory and then copying it to the intended memory location during activation is called a *dual-bank update*. Dual-bank updates are the preferred method of updating firmware, because the current application is retained until the new firmware is verified and activated.

A dual-bank update is only possible if there is enough free space between the end of the current application and the beginning of the application data to store the new firmware image. If the new firmware image is bigger than the available space, it must be transferred in a *single-bank update*. In this process, the firmware image overwrites the existing application. If an error occurs during a single-bank update, no valid application will be left on the device. In this case, the device stays in DFU mode in the bootloader, and the firmware update process can be retried.

If NRF_DFU_SINGLE_BANK_APP_UPDATES is set, then single-bank update is preferred. Otherwise, all firmware updates are performed as dual-bank updates. However, if the new image (SoftDevice, SoftDevice and bootloader, or application) is larger than the available space, the application is deleted to make room for the firmware image, and a single-bank update is performed.

For safety reasons, you can disable single-bank updates in the DFU configuration (see NRF_DFU_FORCE_DUAL_BANK_APP_UPDATES). However, disabling single-bank updates will limit the size of what can be updated to the free pages between the application and the reserved application data. This free space might not be sufficient for a combined update of SoftDevice and bootloader.

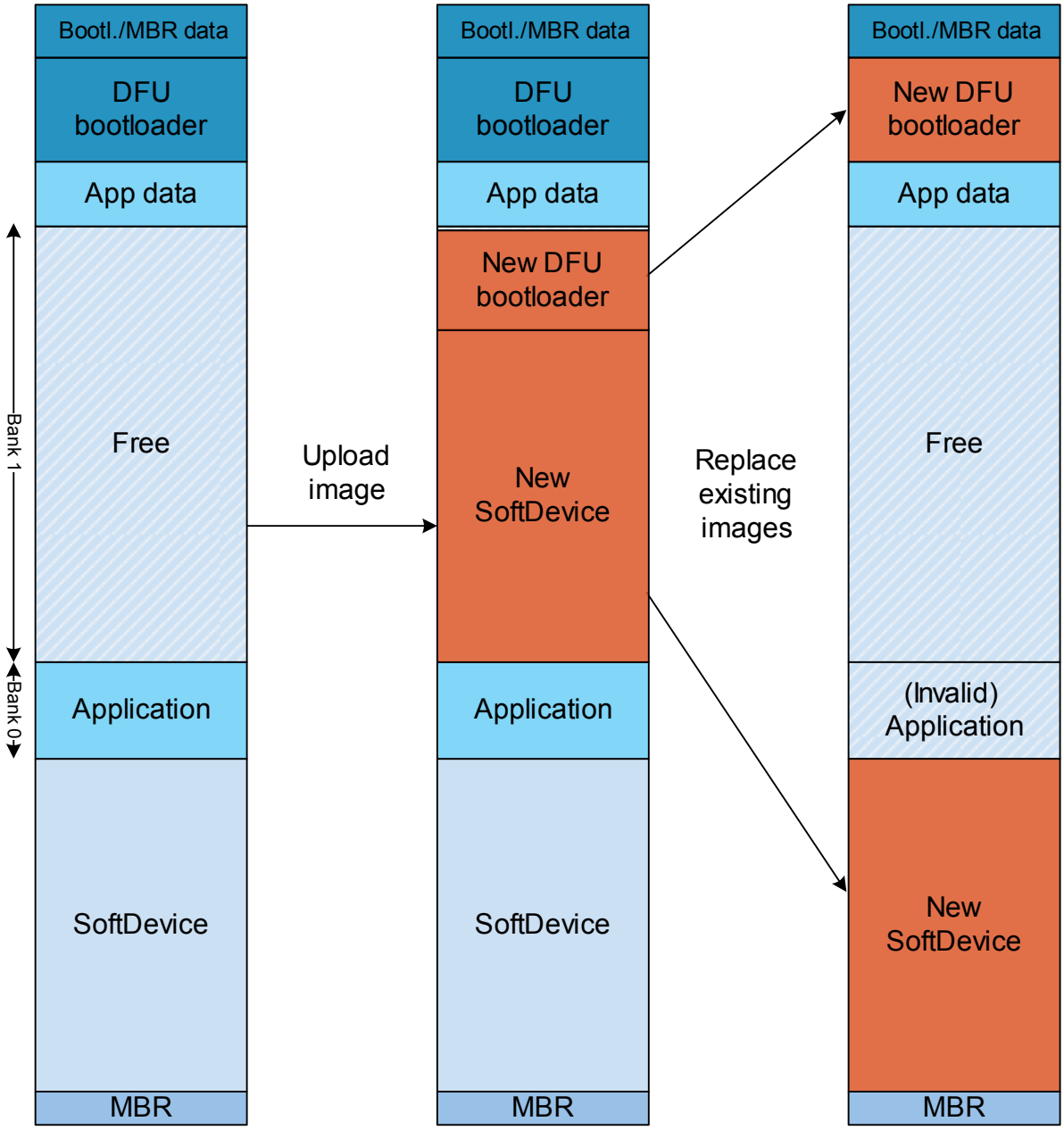
Dual-bank updates

During a dual-bank update, the existing application is preserved until the new firmware image is activated. If the firmware update process fails, you can still reboot the device to start the existing application.

The memory area between the end of the SoftDevice and the beginning of the application data is divided into two banks. Bank 0 holds the existing application, and bank 1 is used to store the received image.

SoftDevice and bootloader

The following figure shows the DFU process for a combined image of bootloader and SoftDevice. If the transferred image contains only a SoftDevice or only a bootloader, the general process is the same, but only the SoftDevice or the bootloader is replaced.

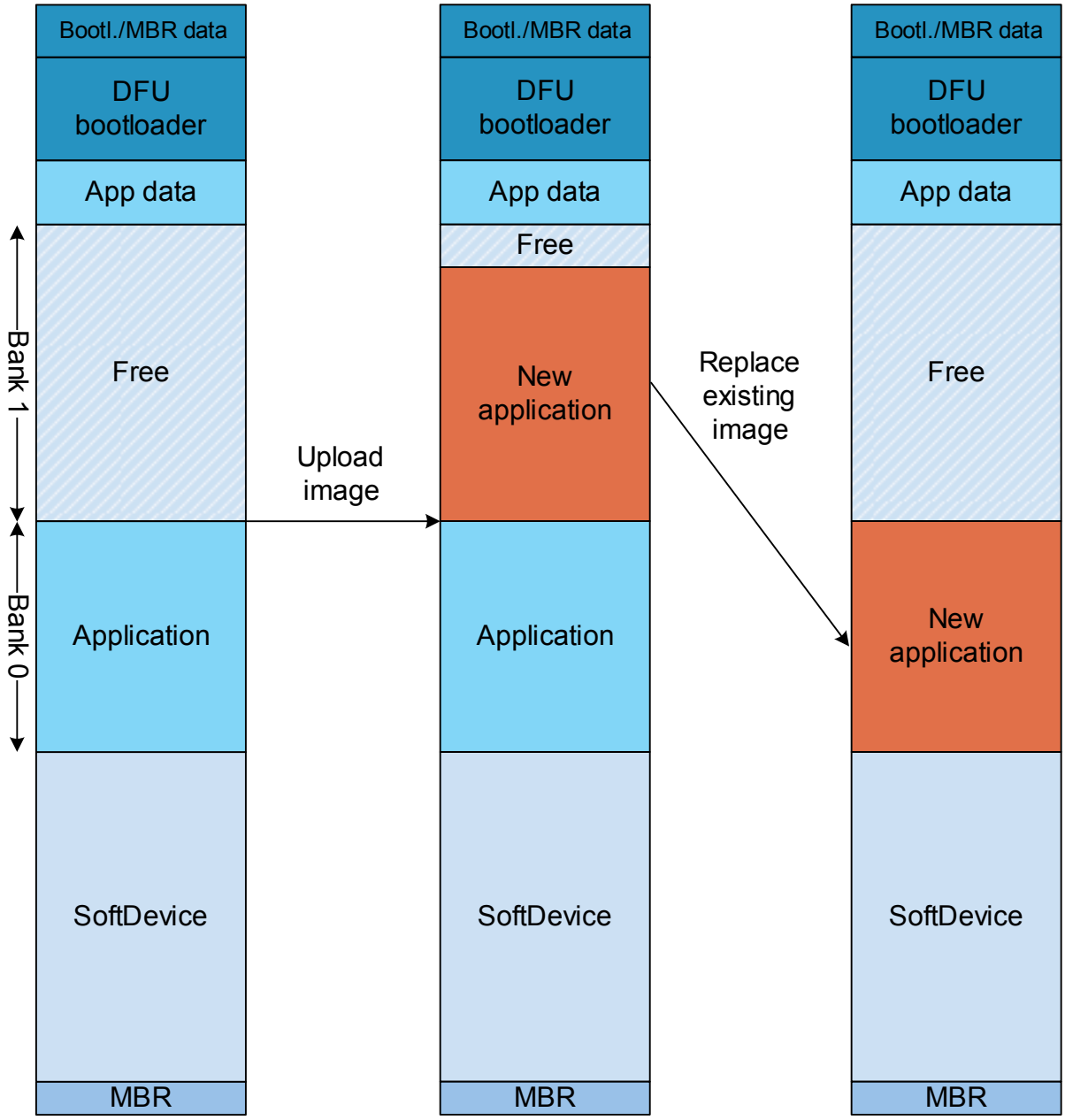


DFU flash operations for a dual-bank SoftDevice and/or bootloader update

The transferred image is stored in the free memory area. Existing application data can be retained; see Preserving application data for more information. After validation, the new SoftDevice and the new bootloader are copied to replace the existing firmware. The application is retained during this process, but it might be invalid because of API changes in the SoftDevice, or because the new SoftDevice has a different size than the existing one.

Application

When updating the application in a dual-bank update, the existing application is retained.



DFU flash operations for a dual-bank application update

The original application is located in memory bank 0. The transferred image is stored in bank 1. After the new application image is received, both the old and the new application are present. This ensures that fallback to the old application is possible if the new application cannot be activated. If the new application can be activated, it is copied from bank 1 to bank 0. Existing application data can be retained; see Preserving application data for more information.

Single-bank updates

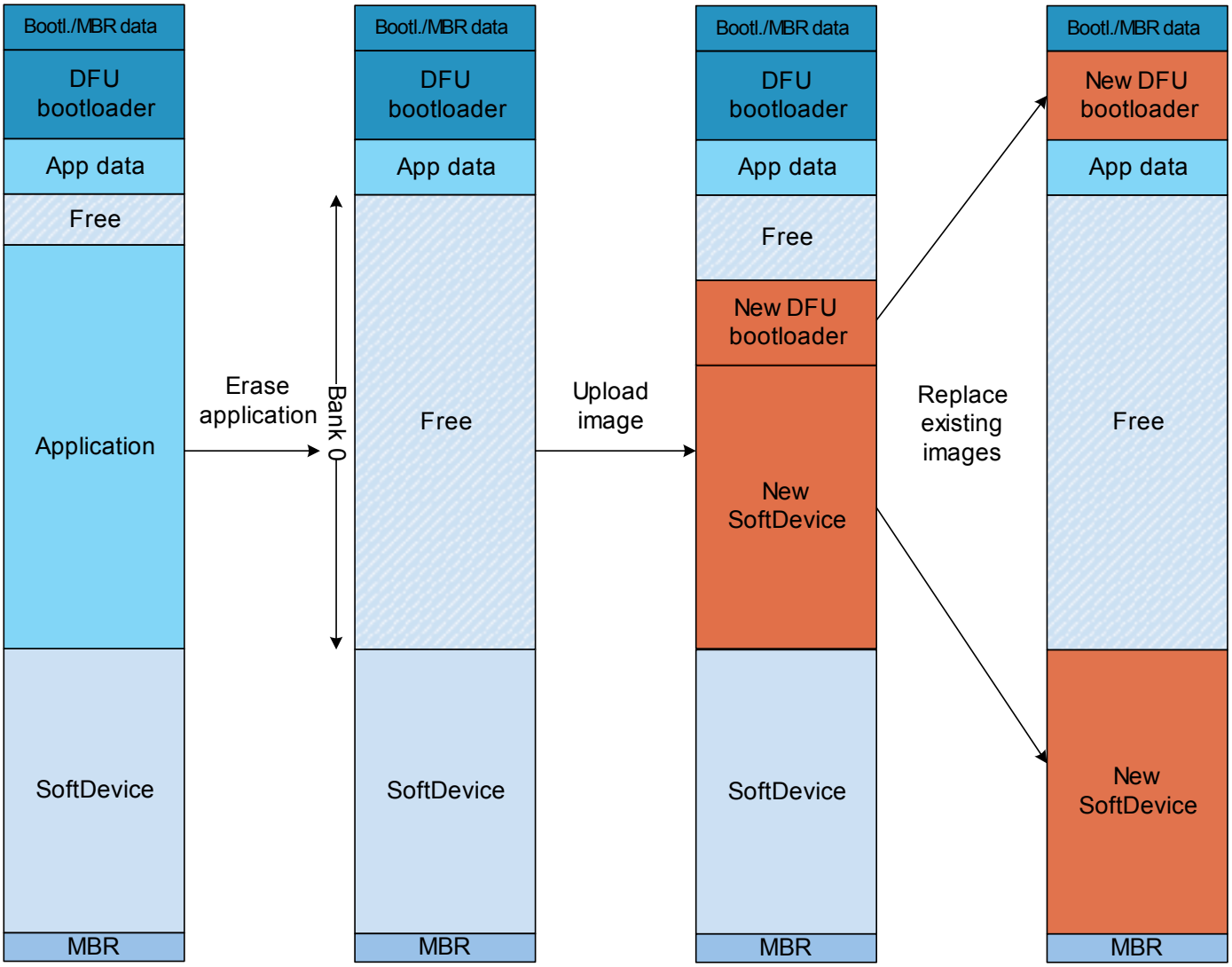
In a single-bank update, the existing application is replaced with the new firmware image. If an error occurs and the device is left without a valid application, the system reverts to DFU mode and you can restart the update process.

The DFU bootloader checks if a dual-bank update is possible. Only if it is not (because the free memory is not sufficient), it will revert to a single-bank update.

Existing application data can be retained; see Preserving application data for more information.

SoftDevice and bootloader

The following figure shows the DFU process for an application in single-bank mode.

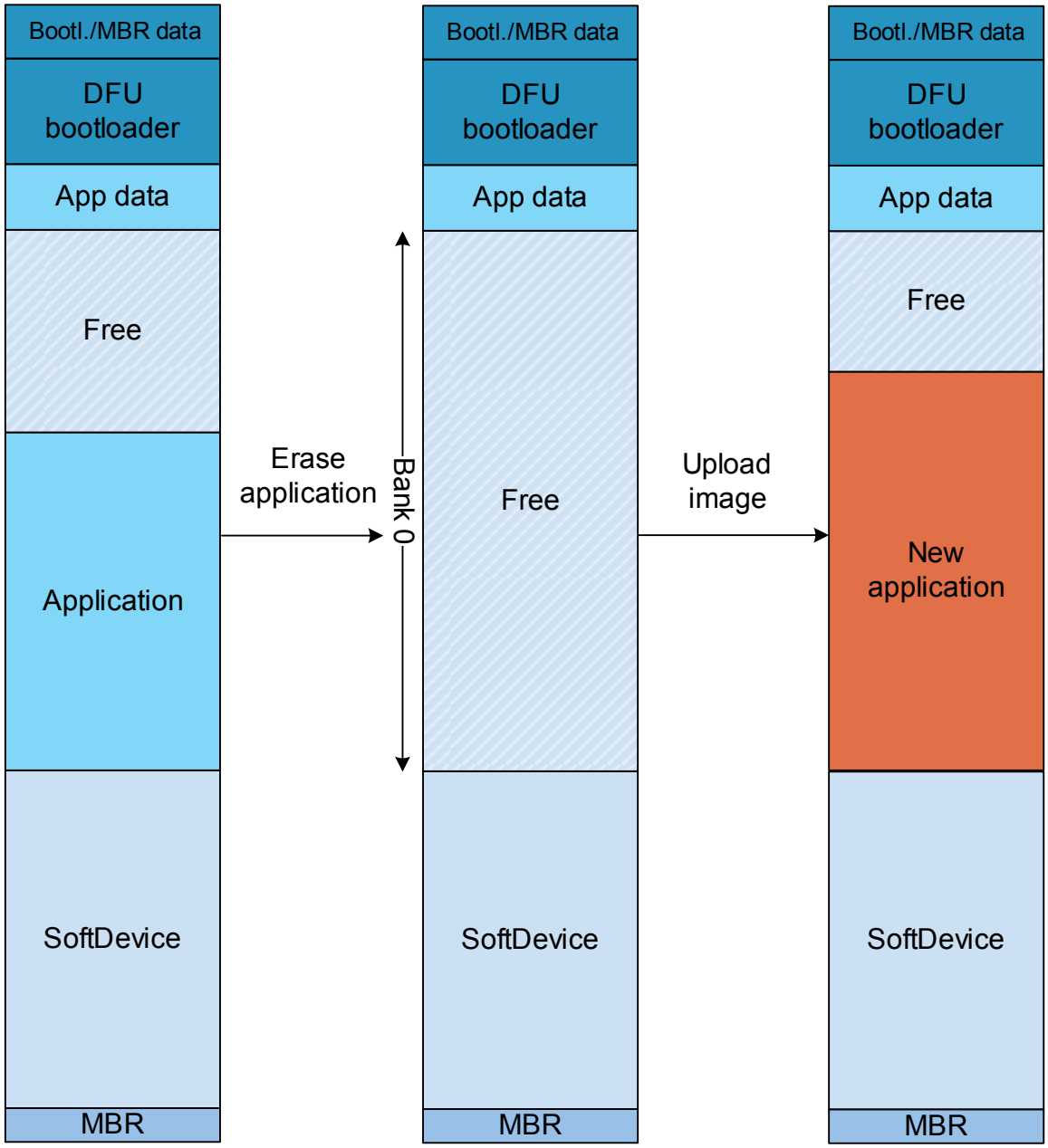


DFU flash operations for a single-bank SoftDevice and/or bootloader update

The existing application is erased to make room for the new firmware image. After validation, the new SoftDevice and the new bootloader are copied to replace the existing firmware. Finally, the device restarts and enters DFU mode (because there is no valid application on the device).

Application

The following figure shows the DFU process for an application in single-bank mode.



DFU flash operations for a single-bank application update

The existing application is erased to make room for the new application image. When the transfer is completed, the bootloader validates the new application. If it is valid, the bootloader activates it. If it is not valid, the bootloader resets, starts in DFU mode, and waits for a new image to be uploaded.

Preserving application data

Application data (like bonding information, system attributes, or data that the application wants to preserve between resets) that should be retained during a Device Firmware Update must be stored in a specific memory area between the application and the bootloader, right before the beginning of the bootloader. This is the default location where the examples provided in this SDK save Peer Manager data.

To preserve application data during a DFU, configure a value for DFU_APP_DATA_RESERVED in `nrf_dfu_types.h`. The default value is 3:

```
#define DFU_APP_DATA_RESERVED           CODE_PAGE_SIZE * 3
```

The value must be a multiple of the page size, so for example 0x1000, 0x2000, 0x3000, and so on for a page size of 0x1000.