VocalFusion® XVF3000 / XVF3100 DESIGN ADVISORY

XM-014244-DA : 90nm to 65nm Flash Memory Change

Important Notice

This document provides essential information and actions to be taken following a product update to a range of XMOS devices which have integrated flash memory. Failure to implement this advisory could cause incorrect factory programming and the inability to execute in-service upgrade processes.

This Design Advisory should be read in conjunction with Product Change Notice **PCN# 014242**.

VocalFusion part numbers covered by this document are:

XVF3000-TQ128-C

XVF3100-TQ128-C

XVF3101-TQ128-C

Bringing technology to life

Contents

1. OVERVIEW

1.1. DESCRIPTION OF CHANGE

Flash memory supplier ISSI has updated the process they use to produce the flash memory device integrated into a range of XMOS xCORE-200 products. The ISSI flash memory devices based on a 90nm process (part number IS25LQ016B) are now at End-of-Life (EOL) and have been replaced by a form, fit and function direct replacement (IS25LP016D) which is manufactured on a 65nm process.

These flash devices are integrated in a range of XMOS devices and from 4Q2020 these devices will be discontinued and replaced with new, compatible parts with the new 65nm version of the flash memory integrated.

Existing part number structures are retained, but the new parts are differentiated with the addition of an ‘A’ suffix on the replacement part order codes.

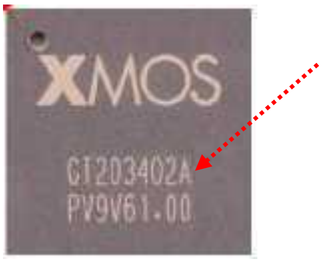
Within the scope of this Design Advisory the following devices are affected.

XVF3000-TQ128-C is replaced by XVF3000-TQ128-CA

XVF3100-TQ128-C is replaced by XVF3100-TQ128-CA

XVF3101-TQ128-C is replaced by XVF3101-TQ128-CA

The new parts can be identified from the date code line on the top marking of new 65nm integrated flash devices. The printed date code line will have an ‘A’ suffix as shown in the figure below:



Identification of package with new 65nm embedded flash device

This change has been notified in XMOS Product Change Notification (PCN) document XM-014242-PN and this Design Advisory has been produced to detail the changes required to the device firmware as a result of this product change.

1.2. IMPACT OF CHANGE

The performance and functionality of the replacement parts are unchanged, and they can continue to be used in existing and new designs.

However, the new IS25LP016D flash devices have a different JEDEC Standard Identification Code (JEDEC ID), which is an embedded hardware identifier for flash devices. The JEDEC ID of the new embedded flash device has changed from 0x9D4015 to 0x9D6015.

This JEDEC ID is read by the XMOS software used to programme the flash image. This ID is then cross-checked with a list of supported devices to verify that a valid flash device is being programmed. The ID reported by the new devices is not contained in this list, so it is not recognised as valid by the programming tools which causes the programming process to abort. This is a deliberate action to prevent misconfiguration of the flash which could be irreversible.

The same issue affects in service Device Firmware Upgrade (DFU) as the same verification process takes place during the DFU operation. A change needs to be made in the production firmware to prevent DFU operations aborting when an in-service firmware update is attempted.

The update to the internal flash therefore impacts two areas:

Area	Products Impacted	Impact of new device	Action Required
Device Firmware Upgrade (DFU)	XVF3000, XVF3100, XVF3101	Firmware change will abort. Upgrade and factory revert no longer possible	Recompile firmware image with new flash spec and install on all product built with the new ‘A’ parts before running field upgrades
Factory Programming (xflash tools)	xTIMEcomposer 14.3 and earlier	Unable to programme new ‘A’ devices with default settings	Use updated .spispec file for device programming

The following sections describe the specific modifications to the firmware required to support the new device. The changes to the source code are very contained and do not impact normal in-service operation of the device. Once implemented will allow the same firmware to run correctly on both new and old parts and allow the DFU function to operate.

2. DEVICE FIRMWARE UPGRADE (DFU)

2.1. DFU PROCESS AND CUSTOM APPLICATION FLASH ACCESS

The DFU process comprises the host (eg. a PC) establishing a USB connection with the XMOS device and sending a request to USB Endpoint 0 that triggers the DFU process. The DFU process writes an upgrade image, transferred via the USB, to the flash memory. The existing firmware on the device executing a DFU process or custom application flash access on the xCORE is responsible for managing the co-packaged flash chip.

To support a range of different flash devices, the generic *quadflash* library used by DFU includes a mandatory cross-check of the JEDEC ID, read from the flash chip, against several flash specification files provided with the *quadflash* library.

However, the built-in flash specifications in the *quadflash* library of current firmware builds do NOT INCLUDE the specification for the new flash device used in the XMOS devices covered by this design advisory. When the existing DFU firmware executes on one of these replacement devices it detects an invalid flash device and aborts the update operation.

If the new ‘A’ parts are programmed with the existing firmware images, the consequence of this is that it will no longer be possible to modify the version of firmware stored in these devices in the field via the DFU mechanism.

2.2. ACTION REQUIRED

To ensure that DFU is supported for new parts users will need to update the XMOS device software with a new firmware image as follows.

2.2.1. PREREQUISITES

Please ensure that the correct version of xTIMEcomposer is installed; release 14.3.2, or later is required for this product release. This tools version can be downloaded from the XMOS website:

https://www.xmos.ai/software/tools/archive

Before compiling the new firmware, ensure that the correct version of the tools is installed and in the command path by executing the following command in a terminal window.

```
xcc –version
```

If the version is something other than 14.3.2 or the command fails, follow the documentation included in the tools release.

2.2.2. STANDARD PRODUCT

For devices that use the standard XMOS reference design the updated source code and precompiled images can be downloaded from the XMOS web site:

https://www.xmos.ai/software/vocalfusion

XVF3x00 firmware release 1.1.3 or higher supports the new XVF3x00-TQ128-CA parts.

These firmware images should be programmed onto the device using `xflash` as described in section 3.2 below.

2.2.3. SOURCE CODE MODIFICATIONS

For devices that use custom firmware that is based on the VocalFusion reference design source code with modifications, this source code must be updated. The a new binary image should then be compiled and used for new 'A' devices.

The reference version of the vocal fusion source code can be downloaded as shown in 2.2.2 above.

MANDATORY CHANGE

VocalFusion release version v1.1.3 is identical to v1.1.2 apart from two source files which have been changed to support the new devices. These two files are:

```
sw_vocalfusion/app_vf_spk_base/src/core/customdefines.h
sc_usb_audio/module_usb_audio/flashlib_user.c
```

The specific changes in these files are shown in Appendix A of this document.

There are two options to implement the required change:

A) If these files have not been modified in a product implementation, simply copy these source files from the v1.1.3 release source tree into the equivalent directories in the local VocalFusion Speaker source tree, replacing the original versions.

B) If these files have been customised, the necessary changes are shown in the difference reports in Appendix A. These changes can be applied manually by copying the appropriate sections from the supplied V1.1.3 reference source code.

UPDATE VERSION NUMBER

Update the product version from v1.1.2 to v1.1.3 by modifying:

```
sw_vocalfusion/module_vocalfusion/include/shareddefines.h
```

and change

```
#define BCD_DEVICE_N 2
```

to

```
#define BCD_DEVICE_N 3
```

and update the CHANGELOG with the new version number.

Note: If a custom, implementation specific version number has been used in a design, the appropriate update should be done instead.

COMPILATION

This source code can then be recompiled to generate a new version of the firmware.

Navigate to the sw_vocalfusion/app_vf_spk_base directory and run

```
xmake CONFIG=<desired config>.
```

For example, to build the **1i1o2_cir43_asr** config, run

```
xmake CONFIG=1i1o2_cir43_asr
```

The resulting .xe file in

```
bin/1i1o2_cir43_asr
```

now contains the necessary compatibility changes for the new flash part. This can be loaded onto the XVF3000 device via the xflash tool as described in Section 3.2.2 below

2.3. FREQUENTLY ASKED QUESTIONS ON DFU

2.3.1. CAN THE NEW FIRMWARE BE USED ON BOTH ORIGINAL AND NEW XMOS PARTS?

YES. The modification documented in section 2.2.3 above includes the flash specifications for both the original and new XMOS parts. This ensures that the modified firmware (i.e. recompiled following the modification above) can be successfully deployed onto either original or new XMOS parts. In both cases these parts will support DFU and custom application flash access.

2.3.2. CAN EXISTING FIRMWARE BE RETAINED?

NOT RECCOMENDED. If DFU or custom flash access is NOT required in existing applications, then it is possible to use the new parts as a direct replacement for the original parts and continue to deploy that same firmware image to Flash.

However, by deploying a firmware image based on releases earlier than v1.1.2 onto new parts, it will **never be possible** to make in-the-field changes (i.e. you CANNOT Upgrade, Downgrade or Revert to Factory) to these parts. In order to avoid this XMOS recommends that all existing firmware is updated to support the new devices before they are introduced.

2.3.3. WHAT HAPPENS IF DFU IS ATTEMPTED ON OLD FIRMWARE?

If a DFU is attempted on a new part loaded with the original firmware, it will result in a runtime exception (in the code running on the XMOS part). The device will recover to its factory image following a power-cycle, the flash contents will be un-changed. This is true for download, revert and upload operations, so it will never be possible to change the version of firmware on the device.

To avoid this scenario, XMOS strongly recommends that all production firmware images are updated to the new version before introducing the new parts into a production flow. This will reduce the risk of the older firmware version being loaded as a factory image on these new ‘A” parts.

In certain scenarios the host software may not display an error, but in most cases there will be a visible error reported. XMOS recommends checking the behaviour of products in the expected use cases following the DFU operation.

2.3.4. WHAT HAPPENS IF A USER UPDATES TO AN OLDER VERSION OF THE FIRMWARE?

If a user attempts to load an older version of the firmware onto a board with new “A” devices the DFU process will install this older version and boot the device from it. This image will **no longer be able to change the flash device** and it is no longer possible to modify the firmware image via DFU. Any further changes require physical access to the device via the JTAG/ interface and the use of the `xflash` tool.

It is critical that system designers implement version controls at the product level to prevent users loading incorrect versions of software onto this device.

2.4. SUMMARY OF PROGRAMMING AND DFU SCENARIOS

Key:

Original firmware image	A firmware image created for the current XMOS parts, prior to the modification.
Future unmodified firmware image	An updated version of the firmware but EXCLUDING the modifications to the firmware (as described in 2.2.3 above)
New firmware image	A firmware image created for the new XMOS parts with the new Flash integrated and including the modifications to the firmware (as described in 2.2.3 above)
Future new firmware image	A subsequent firmware image created for the new XMOS parts (with the new Flash integrated) and including the modifications to the firmware (as described in 2.2.3 above)

ID	PART	FACTORY PROGRAMMED	ATTEMPT UPGRADE TO	OUTCOME
1	Original part	Original firmware image	New firmware image Or Future unmodified firmware image	From the factory, device operates correctly and is upgradeable. Attempts to DFU a "Future firmware image" will succeed.
2	Original part	New firmware image	Future firmware image	From the factory, device operates correctly and is upgradeable. Attempts to DFU a "Future firmware image" will succeed.
3	Original part	New firmware image	Original firmware image	This DFU will succeed and the part remains upgradeable as per (1) or (2) above.
4	New part	Original firmware image	New firmware image	From the factory, device operates correctly but is not upgradeable. ⚠ Attempts to DFU the "New firmware image" will fail. See 2.3.3 above Device will revert to operation according to its factory image on a power-cycle.
5	New part	New firmware image	Future new firmware image	From the factory, device operates correctly and is upgradeable. Attempts to DFU a "Future firmware image" will succeed.
6	New part	New firmware image	Original firmware image	The DFU will succeed but will leave the device in the state of (4) above and so ⚠ All subsequent attempts to DFU (including revert and down-grade) will fail. See 2.3.4 above After this event the device will work correctly as per the original firmware image.

3. XFLASH

3.1. IMPACT ON XFLASH

The `xflash` tool used for programming the firmware image into the device. When using the replacement XMOS parts, certain combinations of `xflash` version and firmware configuration will cause `xflash` to print an error message during factory programming and stop instead of completing successfully.

To check the version of xflash installed enter the command line:

```
xflash --version
```

3.1.1. XFLASH VERSION 14.4.1

xTIMEcomposer V14.4.1 already incorporates support of the IS25LP016D flash device so, provided the .xe file has been complied with the modifications described above included, programming of the embedded device will occur correctly.

3.1.2. XFLASH 14.3.3 AND EARLIER

[Note: For XVF3xxx-TQ128-CA products, xTIMEcomposer version 14.3.2 is the baseline tool set]

With earlier versions of xTIMEcomposer, an error message is displayed by `xflash` when it is run:

```
Error on tile[0]: failed to connect to flash device. Please verify that SQI type is supported and
that the correct SQI ports are defined within your xn file.
```

If the `--no-inq` command option is used, a different error message is printed:

```
Error: F03013 Failed to run : 0x7ffee7480b50
```

Note: The memory address in the latter error message may differ in specific design configurations.

3.2. CORRECTING XFLASH ERRORS

The errors in section 3.1 are generated because the `xflash` tool versions earlier than 14.4.1 do not automatically recognise the flash device. The error messages shown above are removed by creating a new specification file (.spispec file) for the embedded flash device and adding this as a command line argument to `xflash`.

3.2.1. .SPISPEC FILE

The `IS25LP016D.spispec` required for the new ‘A’ parts must have the following contents:

```
21,                         /* Enum value to identify the flashspec in a list */
256,                        /* page size */
8192,                       /* num pages */
3,                          /* address size */
4,                          /* log2 clock divider */
0x9F,                       /* QSPI_RDID */
0,                          /* id dummy bytes */
3,                          /* id size in bytes */
0x9D6015,                   /* device id */
0x20,                       /* QSPI_SE */
4096,                       /* Sector erase is always 4KB */
0x06,                       /* QSPI_WREN */
0x04,                       /* QSPI_WRDI */
PROT_TYPE_NONE,             /* no protection */
{{0,0},{0x00,0x00}},        /* QSPI_SP, QSPI_SU */
0x02,                       /* QSPI_PP */
0xEB,                       /* QSPI_READ_FAST */
1,                          /* 1 read dummy byte */
SECTOR_LAYOUT_REGULAR,      /* mad sectors */
{4096,{0,{0}}},             /* regular sector sizes */
0x05,                       /* QSPI_RDSR */
0x01,                       /* QSPI_WRSR */
0x01,                       /* QSPI_WIP_BIT_MASK */
```

This .spispec file can be stored anywhere on the computer used for programming, but it is advised to copy it into the current working directory where the `xflash` command is run from.

3.2.2. XFLASH COMMAND LINE CHANGE

A typical factory programming procedure could include the following steps, or similar.

```
xflash --noinq --factory app_vf_spk_base_xyz.xe -o flash.bin
xflash --write-all flash.bin
```

When used with new “A” parts the second command will yield an error.

This is corrected by adding `--spi-spec IS25LP016D.spispec` to the second command, which then becomes:

```
xflash --spi-spec IS25LP016D.spispec --write-all flash.bin
```

This changed command must be made only applied for boards built with the new “A” parts. For the older parts the `xflash` command should be run without the `--spi-spec` option.

Note filename of the .xe file containing the firmware will vary depending on the configuration used. For example, the binaries included in the release of VF 3000/3100 on xmos.com are:

- app_vf_spk_base_1i1o2_cir43_asr.xe
- app_vf_spk_base_1i1o2_lin33_asr.xe
- app_vf_spk_base_1i2o2_cir43_48khz.xe
- app_vf_spk_base_1i2o2_lin33_48khz.xe
- app_vf_spk_base_1i6o2_cir43_keyword.xe
- app_vf_spk_base_1i6o2_lin33_keyword.xe

4. TESTING

XMOS strongly recommends that any modifications made based on this Design Advisory are to be fully and robustly tested. Customers are responsible for how they manage their firmware updates. Modified software, utilities or tools may impact the advice provided above, and changes must be tested in the final application environment before release.

4.1. VERIFICATION OF MODIFICATION

The following notes provide guidance to implementers on verification that the modifications described in this design advisory have been implemented correctly.

4.1.1. CHECK THE BCD DEVICE VERSION NUMBER

STEP 1 - CHECK CONNECTED USB DEVICES FOR THE UPDATED VERSION NUMBER:

- xmosdfu (Linux & macOS)
  Linux or macOS users can check the BCD device version number with the **xmosdfu** utility by running

```
xmosdfu --list-devices.
```

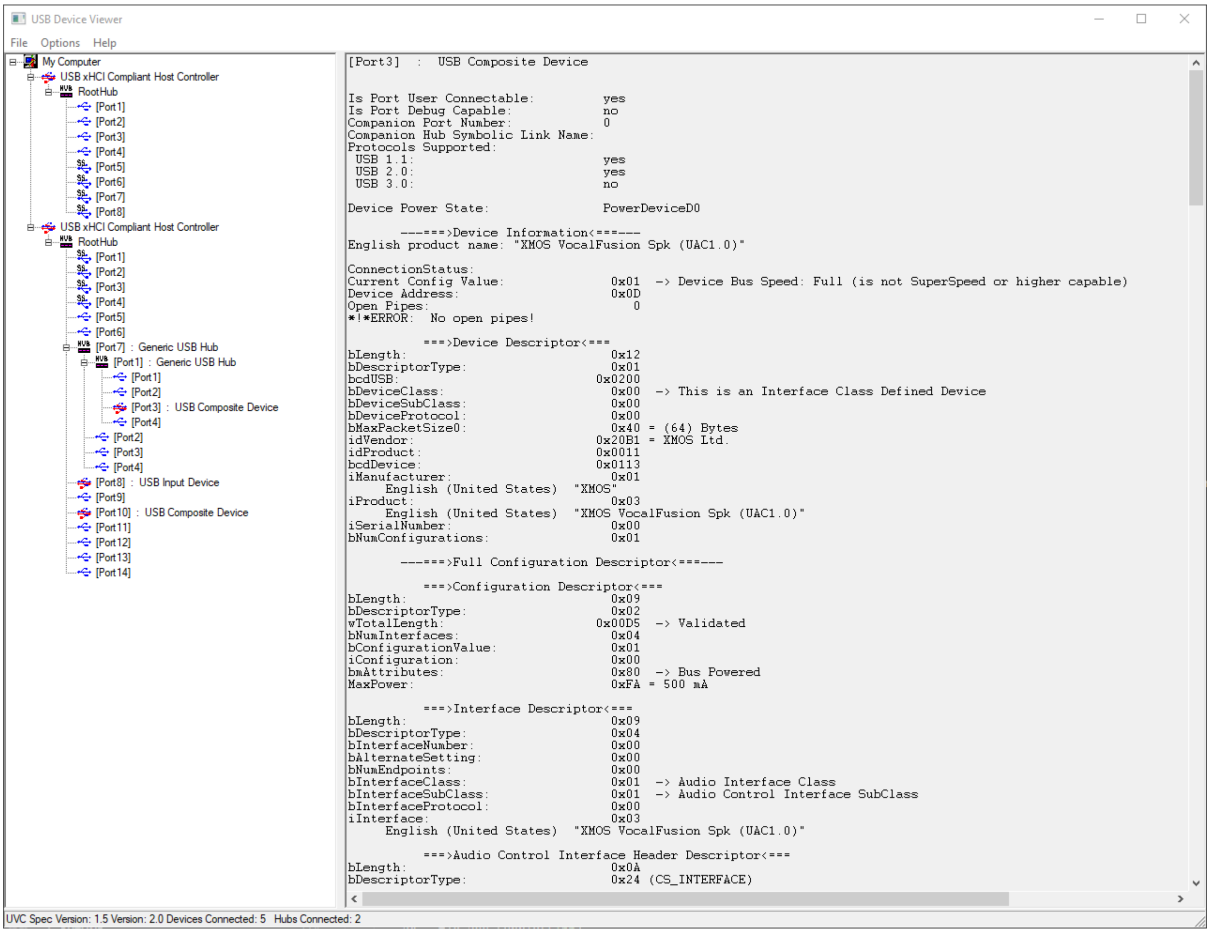
  - Look for the line in the output where the VID and PID match the following:

```
VID = 0x20b1, PID = 0x11, BCDDevice: 0x113
```

  - Note the BCDDevice number

Go to step 2 below

- USBView (Windows)
  - Download USBView Following the instructions on this page: https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/usbview
  - Open USBView in File Explorer: Open the install directory (C:\Program Files (x86)\Windows Kits\10\Debuggers\x64) and select USBView.exe
  - Find the VocalFusion Spk device by selecting devices in the left pane and checking the iProduct value under Device Descriptor in the right pane.


USB Device Viewer
File Options Help
My Computer
USB xHCI Compliant Host Controller
RootHub
[Port1]
[Port2]
[Port3]
[Port4]
[Port5]
[Port6]
[Port7]
[Port8]
USB xHCI Compliant Host Controller
RootHub
[Port1]
[Port2]
[Port3]
[Port4]
[Port5]
[Port6]
[Port7] : Generic USB Hub
[Port1] : Generic USB Hub
[Port1]
[Port2]
[Port3] : USB Composite Device
[Port4]
[Port2]
[Port3]
[Port4]
[Port8] : USB Input Device
[Port9]
[Port10] : USB Composite Device
[Port11]
[Port12]
[Port13]
[Port14]
UVC Spec Version: 1.5 Version: 2.0 Devices Connected: 5 Hubs Connected: 2

```
[Port3]  :  USB Composite Device


Is Port User Connectable:         yes
Is Port Debug Capable:            no
Companion Port Number:            0
Companion Hub Symbolic Link Name:
Protocols Supported:
 USB 1.1:                         yes
 USB 2.0:                         yes
 USB 3.0:                         no

Device Power State:               PowerDeviceD0

       ---===>Device Information<===---
English product name: "XMOS VocalFusion Spk (UAC1.0)"

ConnectionStatus:
Current Config Value:              0x01  -> Device Bus Speed: Full (is not SuperSpeed or higher capable)
Device Address:                    0x0D
Open Pipes:                           0
*!*ERROR:  No open pipes!

          ===>Device Descriptor<===
bLength:                           0x12
bDescriptorType:                   0x01
bcdUSB:                          0x0200
bDeviceClass:                      0x00  -> This is an Interface Class Defined Device
bDeviceSubClass:                   0x00
bDeviceProtocol:                   0x00
bMaxPacketSize0:                   0x40 = (64) Bytes
idVendor:                        0x20B1 = XMOS Ltd.
idProduct:                       0x0011
bcdDevice:                       0x0113
iManufacturer:                     0x01
     English (United States)  "XMOS"
iProduct:                          0x03
     English (United States)  "XMOS VocalFusion Spk (UAC1.0)"
iSerialNumber:                     0x00
bNumConfigurations:                0x01

       ---===>Full Configuration Descriptor<===---

          ===>Configuration Descriptor<===
bLength:                           0x09
bDescriptorType:                   0x02
wTotalLength:                    0x00D5  -> Validated
bNumInterfaces:                    0x04
bConfigurationValue:               0x01
iConfiguration:                    0x00
bmAttributes:                      0x80  -> Bus Powered
MaxPower:                          0xFA = 500 mA

          ===>Interface Descriptor<===
bLength:                           0x09
bDescriptorType:                   0x04
bInterfaceNumber:                  0x00
bAlternateSetting:                 0x00
bNumEndpoints:                     0x00
bInterfaceClass:                   0x01  -> Audio Interface Class
bInterfaceSubClass:                0x01  -> Audio Control Interface SubClass
bInterfaceProtocol:                0x00
iInterface:                        0x03
     English (United States)  "XMOS VocalFusion Spk (UAC1.0)"

          ===>Audio Control Interface Header Descriptor<===
bLength:                           0x0A
bDescriptorType:                   0x24 (CS_INTERFACE)
```

  - Look in the right pane for bcdDevice.

STEP 2 CHECK THE BCDDEVICE VALUE

- If BCDDevice is 0x113 - Firmware is successfully updated.
- If BCDDevice is 0x112 – Firmware is the old , unmodified version of the firmware.
- If BCDDevice is 0x9901 – Device is running a DFU test build of the firmware.

4.1.2. VERIFICATION OF EVALUATION BINARY IMAGES

Use the latest v1.1.3 release of the required configuration and flash it onto the device following the instructions provided in the documentation.

Follow instructions in section 4.1.1 above to check the BCD Device Version Number to verify that the device is running the new firmware.

4.1.3. VERIFICATION OF CUSTOM BINARY IMAGES

To build VocalFusion firmware from source, ensure that the source code has been updated with the required changes described in Section 2.2.3 above and detailed in the Appendix of this document. This source code should then be recompiled and flashed onto the device. If the BCD Device version number has been changed to a custom value, this can be verified using the instructions in section 4.1.1 above.

Re-build the firmware, but this time add TEST_DFU_1=1 to the `xmake` command.

For example, if the original `xmake` command was:

```
xmake CONFIG=1i1o2_cir43_asr
```

then run the command:

```
xmake CONFIG=1i1o2_cir43_asr TEST_DFU_1=1
```

Rename the resulting .xe file to “dfu.xe”.

Created a DFU binary by running the following `xflash` command:

```
xflash --factory-version 14.3 --upgrade 1 dfu.xe -o dfu.bin --no-compression
```

With the device connected, download the DFU image to the device by running the following `xmosdfu` command:

```
sudo ./xmosdfu XMOS_VF_SPK_BASE_AUDIO1_PID --download dfu.bin
```

Check the BCD Device Version number by following instructions in Section 4.1.1 to verify that the device is running the DFU test version of the firmware.

5. FURTHER INFORMATION

DOCUMENT TITLE	DOWNLOAD
xTIMEcomposer Tools (including xflash)	https://www.xmos.ai/software-tools/
xTIMEcomposer Tools User Guide	https://www.xmos.ai/file/tools-user-guide/
VocalFusion Firmware	https://www.xmos.ai/software/vocalfusion

Questions regarding this Design Advisory can be addressed by raising a support ticket via https://www.xmos.ai/support/

6. REVISION HISTORY

DOCUMENT VERSION	RELEASE DATE	CHANGE DESCRIPTION
XM-014244-DA-4	4 September 2020	Initial Release

APPENDIX A: DETAILS OF SOURCE CODE CHANGES REQUIRED

The updated source code files can be found in the VocalFusion software V1.1.3 which can be downloaded from the XMOS web site:

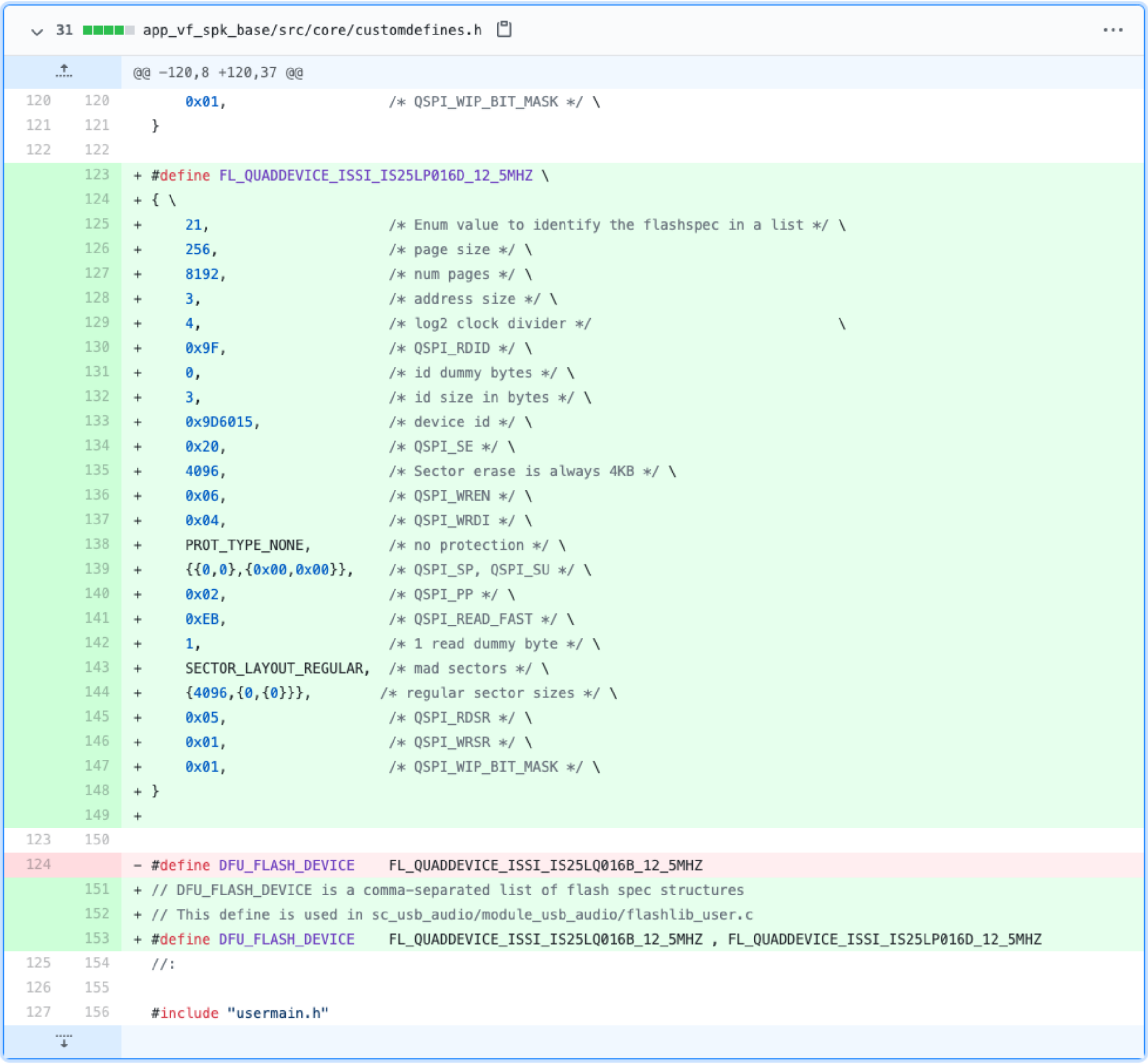
https://www.xmos.ai/software/vocalfusion

There are two changes that must be made exactly as specified in all implementations, and a third which can be adapted for specific implementations if required.

Source lines below highlighted in green indicate code added; in red code that should be removed.

File #1: sw_vocalfusion/app_vf_spk_base/src/core/customdefines.h

This change adds the new flash spec to the DFU_FLASH_DEVICE pre-processor variable.

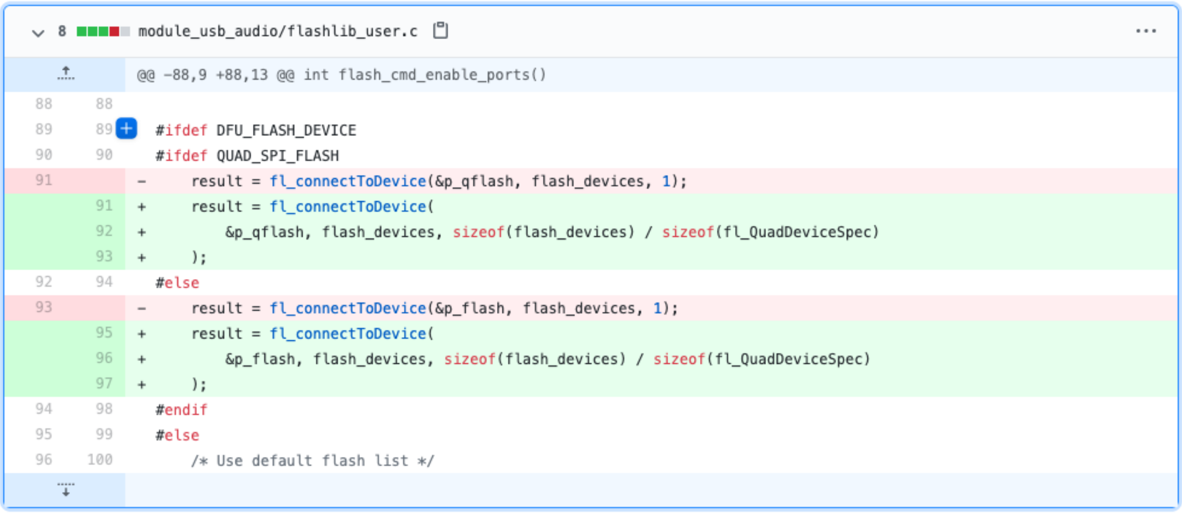


31 app_vf_spk_base/src/core/customdefines.h

```
@@ -120,8 +120,37 @@
      0x01,                   /* QSPI_WIP_BIT_MASK */ \
  }

+ #define FL_QUADDEVICE_ISSI_IS25LP016D_12_5MHZ \
+ { \
+     21,                     /* Enum value to identify the flashspec in a list */ \
+     256,                    /* page size */ \
+     8192,                   /* num pages */ \
+     3,                      /* address size */ \
+     4,                      /* log2 clock divider */                             \
+     0x9F,                   /* QSPI_RDID */ \
+     0,                      /* id dummy bytes */ \
+     3,                      /* id size in bytes */ \
+     0x9D6015,               /* device id */ \
+     0x20,                   /* QSPI_SE */ \
+     4096,                   /* Sector erase is always 4KB */ \
+     0x06,                   /* QSPI_WREN */ \
+     0x04,                   /* QSPI_WRDI */ \
+     PROT_TYPE_NONE,         /* no protection */ \
+     {{0,0},{0x00,0x00}},    /* QSPI_SP, QSPI_SU */ \
+     0x02,                   /* QSPI_PP */ \
+     0xEB,                   /* QSPI_READ_FAST */ \
+     1,                      /* 1 read dummy byte */ \
+     SECTOR_LAYOUT_REGULAR,  /* mad sectors */ \
+     {4096,{0,{0}}},        /* regular sector sizes */ \
+     0x05,                   /* QSPI_RDSR */ \
+     0x01,                   /* QSPI_WRSR */ \
+     0x01,                   /* QSPI_WIP_BIT_MASK */ \
+ }
+

- #define DFU_FLASH_DEVICE    FL_QUADDEVICE_ISSI_IS25LQ016B_12_5MHZ
+ // DFU_FLASH_DEVICE is a comma-separated list of flash spec structures
+ // This define is used in sc_usb_audio/module_usb_audio/flashlib_user.c
+ #define DFU_FLASH_DEVICE    FL_QUADDEVICE_ISSI_IS25LQ016B_12_5MHZ , FL_QUADDEVICE_ISSI_IS25LP016D_12_5MHZ
  //:

  #include "usermain.h"
```

File #2: sc_usb_audio/module_usb_audio/flashlib_user.c

This change enables support for multiple flash specs defined in DFU_FLASH_DEVICE

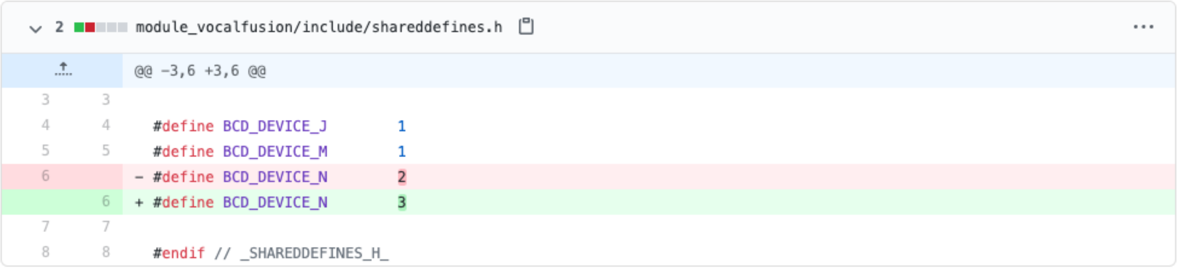


```
@@ -88,9 +88,13 @@ int flash_cmd_enable_ports()

  #ifdef DFU_FLASH_DEVICE
  #ifdef QUAD_SPI_FLASH
-     result = fl_connectToDevice(&p_qflash, flash_devices, 1);
+     result = fl_connectToDevice(
+         &p_qflash, flash_devices, sizeof(flash_devices) / sizeof(fl_QuadDeviceSpec)
+     );
  #else
-     result = fl_connectToDevice(&p_flash, flash_devices, 1);
+     result = fl_connectToDevice(
+         &p_flash, flash_devices, sizeof(flash_devices) / sizeof(fl_QuadDeviceSpec)
+     );
  #endif
  #else
      /* Use default flash list */
```

File #3: sw_vocalfusion/module_vocalfusion/include/shareddefines.h

This change increments the product version number from v1.1.2 to v1.1.3

Note – If a custom release numbering system has been implemented, then the version number defined here should be adapted as needed.



```
@@ -3,6 +3,6 @@

  #define BCD_DEVICE_J        1
  #define BCD_DEVICE_M        1
- #define BCD_DEVICE_N        2
+ #define BCD_DEVICE_N        3

  #endif // _SHAREDDEFINES_H_
```