Im Using STM32F767ZI Nucleo Board which has 2Mbytes of flash for code storage. It also allows for the flash to be evenly divided between two banks. This gives bank 1 one Mbyte of data, and bank 2 one Mbyte of data.

Bank 1 Address offset in Flash is 0x08000000

Bank 2 Address offset in Flash is 0x08100000

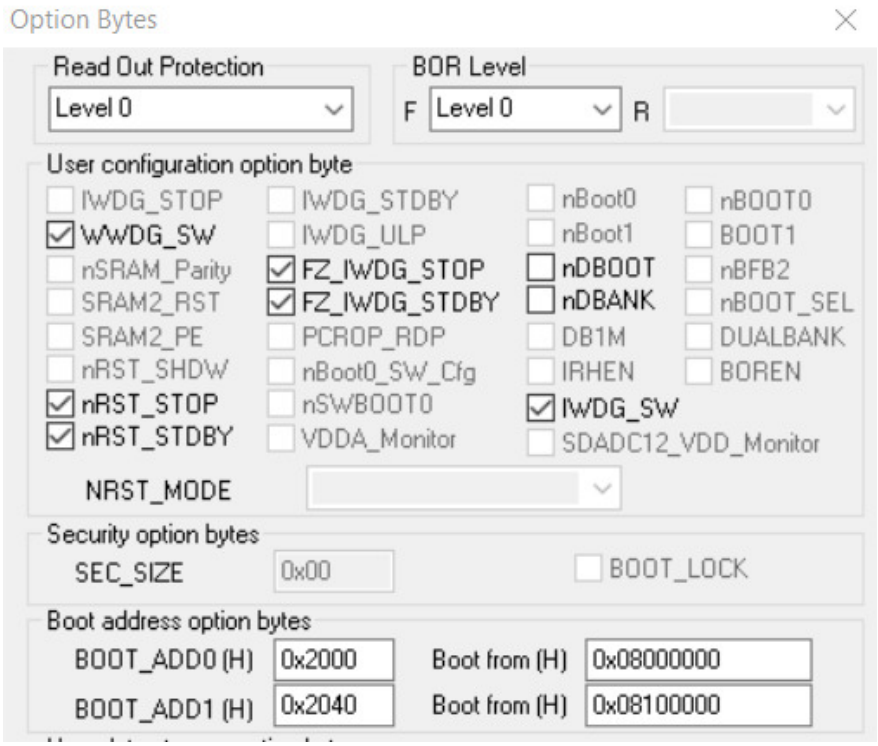
With the use of nDBOOT = nDBANK = 0 option bytes the memory is configured in the dual bank, dual boot mode.

My Current Testing Procedure:

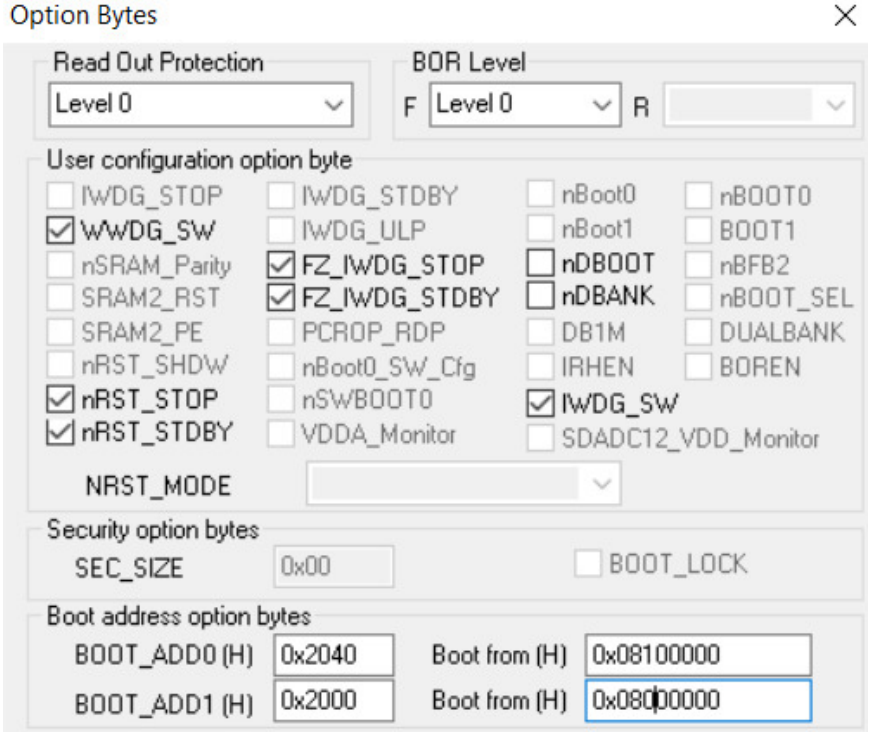
Step 1) I compile the firmware which has a single byte indicating the version of the firmware. I compile it twice once with version byte as 1, and once with version byte as 2. Otherwise its identical code.

Step 2) I use STM32 ST-Link Utility to manually program each version into each bank and run it. I do this for both version with both banks, so 4 times.

I use the following option bytes to run from bank 1 while programming the .bin at offset for Bank 1:



And i use the following option bytes to run from bank 2 while programming the .bin at offset for Bank 2:



All of this up to this point works as expected and im able to read out the different versions from each of the banks.

Step 3) I load version 1 into bank 1 using method above and i use the following code to programmatically erase/program bank 2 with Version 2 firmware.

```
HAL_FLASH_Unlock(); /* Unlock the Flash to enable the flash control register access *************/
HAL_FLASH_OB_Unlock(); /* Allow Access to option bytes sector */
FLASH_EraseInitTypeDef pEraseInit;
if ((SYSCFG->MEMRMP & SYSCFG_MEMRMP_SWP_FB) == RESET) //Test if Bank == 1
{
     pEraseInit.Banks = FLASH_BANK_2;
     pEraseInit.TypeErase = FLASH_TYPEERASE_MASSERASE;
     pEraseInit.VoltageRange  = VOLTAGE_RANGE_3;
     uint32_t SectorError = 0;
     halstatus = HAL_FLASHEx_Erase(&pEraseInit, &SectorError);
     if(halstatus != HAL_OK)
     {
          return 15;
     }
}
else
{
     pEraseInit.Banks = FLASH_BANK_1;
     pEraseInit.TypeErase = FLASH_TYPEERASE_MASSERASE;
     pEraseInit.VoltageRange  = VOLTAGE_RANGE_3;
     uint32_t SectorError = 0;
     halstatus = HAL_FLASHEx_Erase(&pEraseInit, &SectorError);
     if(halstatus != HAL_OK)
     {
          return 15;
     }
}??????????????????????????????????????????????????????
```

And use this code to actually program it

```
uint32_t BaseAddress = 0;
if ((SYSCFG->MEMRMP & SYSCFG_MEMRMP_SWP_FB) == RESET) //Test if Bank == 1
{
     BaseAddress = 0x08100000;
}
else
{
     BaseAddress = 0x08000000;
}
HAL_FLASH_Unlock();
HAL_FLASH_OB_Unlock();
__HAL_FLASH_CLEAR_FLAG(FLASH_FLAG_WRPERR);
__HAL_FLASH_CLEAR_FLAG(FLASH_FLAG_ALL_ERRORS);
for(int i = 0; i < 48; i++)
```

```
{
     halstatus = HAL_FLASH_Program(FLASH_TYPEPROGRAM_BYTE,BaseAddress + FirmwareAddressOffset + i,FirmwareData[i]);
     if(halstatus != HAL_OK)
     {
          return 4; //Error 4
     }
}
int verified = 0;
uint8_t val = 0;
for(int i = 0; i < 48; i++)
{
     val = *(__IO uint8_t *) (BaseAddress + FirmwareAddressOffset + i);
     if(val == FirmwareData[i])
     {
          verified++;
     }
     //verified++; //Added for Debug
}
if(verified == 48)
{
     FirmwareAddressOffset = FirmwareAddressOffset + 48;
     return 3;//OK
}
else
{
     return 4;//Error
}????????????????????????????????????????????????????????????????????????????????????
```

Once its all done i use the following code to actually switch the banks

```
HAL_FLASH_OB_Unlock();
FLASH_OBProgramInitTypeDef flashConfig;
if ((SYSCFG->MEMRMP & SYSCFG_MEMRMP_SWP_FB) == RESET) //Test if Bank == 1
{
     //Switch boot to bank 2, should be done at the end of update
     FLASH_WaitForLastOperation(9000);
     HAL_FLASHEx_OBGetConfig(&flashConfig);
     flashConfig.BootAddr0 = 0x2040;
     flashConfig.BootAddr1 = 0x2000;
     HAL_FLASHEx_OBProgram(&flashConfig);
}
else
{
     //Switch boot to bank 1, should be done at the end of update
     FLASH_WaitForLastOperation(9000);
     HAL_FLASHEx_OBGetConfig(&flashConfig);
     flashConfig.BootAddr0 = 0x2000;
     flashConfig.BootAddr1 = 0x2040;
     HAL_FLASHEx_OBProgram(&flashConfig);
}
HAL_FLASH_OB_Launch();
NVIC_SystemReset();????????????????????????????????????????????????????????????????????
```

So for Step 3 This works. It loads version 2 into Bank 2 and starts running it. I can see version 2 being output, and after i hit hard reset it again loads the new firmware from Bank 2. All is as expected here.

My problem is as follows:

When i try to program version 1 or version 2 into Bank 2 and run the upgrade process it never is able to program anything into bank 1. It can erase it but the actual program never happens even though no errors are reported except when it actually tries to verify the memory against the live data. When i look at the Bank 1 memory after programming from Bank 2 its all 0xFFFF.

I need help to figure out why it is able to program Bank 2 from Bank 1, but fails to program Bank 1 when running from Bank 2?

-Andriy