STM32L562 Dual Flash Bank Firmware Update Application Example

Gintaras Drukteinis,
RUTRONIK Electronics Worldwide, Kaunas, Lithuania

Abstract — **The capability to update firmware after a product is released to the market has become a common procedure in most applications. Microcontroller manufacturers are adopting new technologies and methods to offer augmented flexibility and safety during On-the-fly firmware updates. The dual flash bank programming and swapping techniques using the STM32L562 Cortex-M33 microcontroller are shown in this application note.**

Index Terms* — *Microcontroller (MCU), Random Access Memory (RAM), Read-Only Memory (ROM), Option Byte (OB), Trust Zone Enable option (TZEN), Interrupt Vector Table (IVT), Command Line Interface (CLI)

I. Introduction

STM32L5 MCUs might have up to 512KB of Flash memory with a dual-flash bank capability. The most important advantage of the dual-flash bank is that firmware can be updated while MCU is executing its regular tasks and if the update procedure fails, the MCU will stay functional.

For time deterministic applications the On-the-fly firmware procedure might decrease the performance to system failure. To avoid such malfunction the interrupt routines and other most vulnerable code should be executed from RAM.

II. Flash bank swapping

Flash banks are configured using Option Bytes (OB) located in system memory of the MCU. To be able to use flash banks separately the option bit DBANK must be set to logical '1'. If OB SWAP_BANK is set to logical '1', then Bank A addresses are swapped with Bank B and the MCU will now load the application from Bank B every time on system boot.

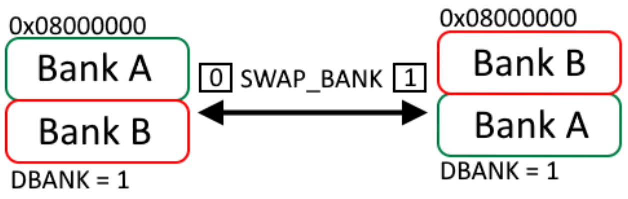


Fig. 1 Bank swapping representation in STM32L562.

Flash memory begins at address 0x08000000 in STM32L5 devices if the TZEN option is disabled[1]. This is also the beginning of the 256KB Bank A in MCUs with 512KB of Flash memory. Bank A is located at 0x08040000 accordingly.

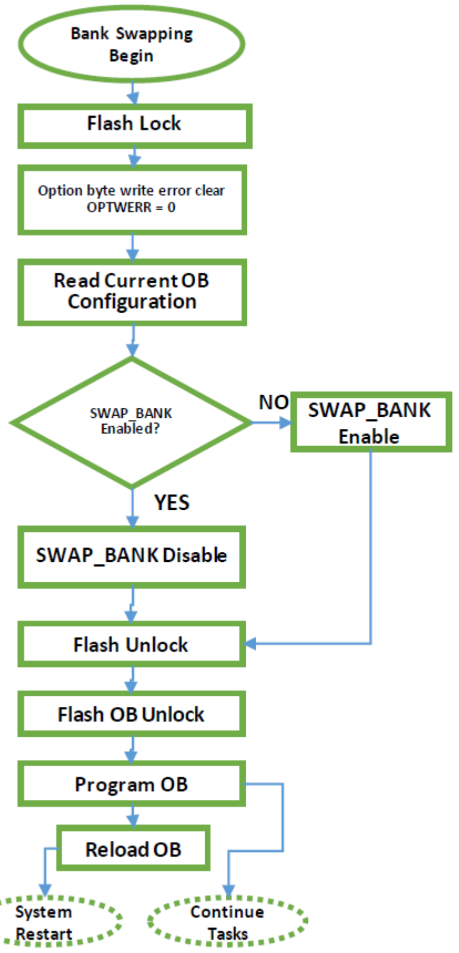


Fig. 2 Bank swapping program algorithm.

After both banks are configured as swapped, the application will continue to run normally as long as it is needed. From this point, since the banks swapping cannot be done on-the-fly the application may jump to a freshly programmed bank with all the actual RAM content unchanged and continue its operation

[1] Boot address is configurable with NSBOOTADDx and SECBOOTADD0 option bytes.

or simply reboot the system by reloading the OB – it is completely application dependent. The advantage of a proposed bank swapping technique is that another bank is always at 0x08040000, hence it simplifies application development.

III. Application Example

The firmware example is based on the STMicroelectronics "X-CUBE-DBFU" expansion package, though the basic workflow differs. The STM32L5 specifics needed to be taken into account while modifying the original code. For simplicity matters, the transition to a new code without going through system reset is not executed. The single RAM area is reserved using a linker script to execute all interrupts from there. The LED will blink every second from RTC interrupt notifying the users about the device status.

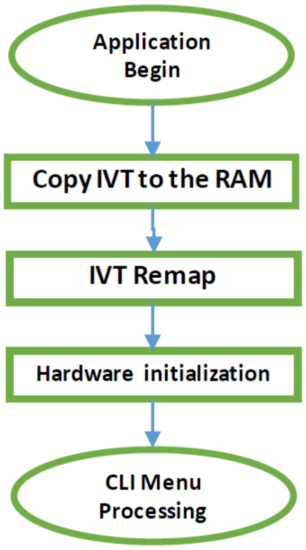
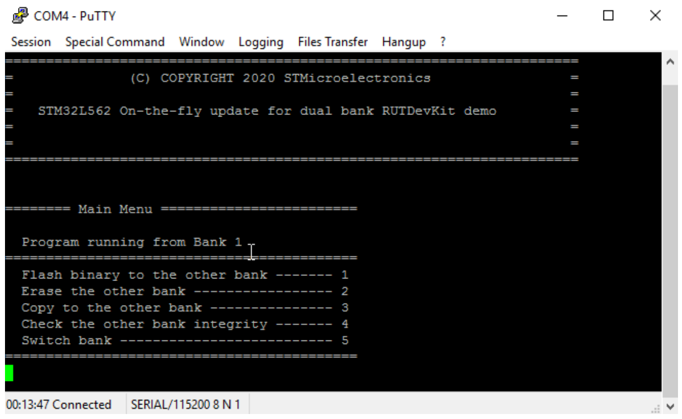


Fig. 3 Initialization algorithm.

The application runs on the RUTDevKit development platform and it is controlled using a PuTTY terminal. Embedded STM32L562ZET6Q MCU is interfaced with ST-LINK COM port, hence it is accessible as a virtual COM port using Micro USB connection. All available options are shown in the terminal window after MCU has completed all initialization tasks. The option '5' will switch to the other bank using the algorithm shown in (Fig. 2). The application must be present in other banks before swapping banks. It can be done with option '1' where application binary image will be transferred using YMODEM protocol via USB connection.



```
======================================================================
=              (C) COPYRIGHT 2020 STMicroelectronics                 =
=                                                                    =
=   STM32L562 On-the-fly update for dual bank RUTDevKit demo         =
=                                                                    =
=                                                                    =
======================================================================


======== Main Menu ========================

  Program running from Bank 1
============================================
  Flash binary to the other bank ------- 1
  Erase the other bank ----------------- 2
  Copy to the other bank --------------- 3
  Check the other bank integrity ------- 4
  Switch bank -------------------------- 5
============================================
```

00:13:47 Connected | SERIAL/115200 8 N 1

Fig. 4 Application user interface.

A straightforward solution is available using option '3' as the firmware simply will be copied from one currently used bank to the other, hence users might test the same application but running it on the other bank.

A linker script is written to reserve a RAM sector from 0x20038000 to 0x2003FFFF (32KB) for interrupt routines. Also the ROM sector from 0x08000000 to 0x08002FFF is used for IVT initialization. The rest of the initialization is done by running the application as it copies the IVT to the reserved RAM location and remaps the IVT to run from there.

IV. Summary

The application example demonstrates how to use dual bank features in STM32L562 MCU. No performance issues were detected while testing on-the-fly bank programming. Attention has to be paid with a linker script when starting with a new project as well as with interrupt service routines.

References

[1] "On-the-fly firmware update for dual bank STM32 microcontrollers" Application Note AN4767, by STMicroelectronics (May 2019).

[2] "X-CUBE-DBFU" STM32Cube Expansion Package, by STMicroelectronics. Available: www.st.com

[3] "RUTDevKit User Manual" user manual, by Rutronik. (May 2020). Available: www.rutronik.com

Contact:

Gintaras Drukteinis
Technical Support Engineer
RUTRONIK Elektronische Bauelemente GmbH
Jonavos g. 30
44262 Kaunas
Lithuania
gdr@rutronik.com
www.rutronik.com