AN2557
Application note

STM32F10x in-application programming using the USART

Introduction

An important requirement for most Flash-memory-based systems is the ability to update firmware when installed in the end product. This ability is referred to as in-application programming (IAP). The purpose of this application note is to provide general guidelines for creating an IAP application. The STM3210B-EVAL, STM3210E-EVAL and STM3210C-EVAL boards were used to validate the IAP driver.

The STM32F10x microcontroller can run user-specific firmware to perform IAP of the microcontroller-embedded Flash memory. This feature allows the use of any type of communication protocol for the reprogramming process (such as CAN, USART, USB). USART is the example used in this application note.

Contents

List of tables

List of figures

1 IAP overview

Low-density devices are STM32F101xx, STM32F102xx and STM32F103xx microcontrollers where the Flash memory density ranges between 16 and 32 Kbytes.

Medium-density devices are STM32F101xx and STM32F103xx microcontrollers where the Flash memory density ranges between 32 and 128 Kbytes. Medium-density devices are implemented in the STMicroelectronics STM3210B-EVAL evaluation board.

High-density devices are STM32F101xx and STM32F103xx microcontrollers where the Flash memory density ranges between 256 and 512 Kbytes. High-density devices are implemented in the STMicroelectronics STM3210E-EVAL evaluation board.

Connectivity line devices are STM32F105xx and STM32F107xx microcontrollers.

1.1 Principle

You should program the IAP driver to the Flash memory base address via the JTAG/SWD interface using the development toolchain of your choice or the factory-embedded boot loader in the System memory area.
The IAP driver uses the USART to load a binary file from the HyperTerminal to the STM32F10x's internal Flash memory, and then executes it.

1.2 IAP driver description

The IAP driver contains the following set of source files:

- *main.c*: where the USART initialization and RCC configuration are set. A main menu is then executed from the *common.c* file.
- *common.c*: contains display functions and the main menu routine. The main menu gives the options of loading a new binary file, executing the binary file already loaded and disabling the write protection of the pages where the user loads his binary file (if they are write-protected).
- *ymodem.c* and *download.c*: they are used to receive the data from the HyperTerminal application (using the YMODEM protocol[a], and then to load them into the STM32F10x's internal RAM. In the event of a failure when receiving the data, the "Failed to receive the file" error message is displayed. If the data is received successfully, it is programmed into the internal Flash memory from the appropriate address. A comparison between internal RAM contents and internal Flash memory contents is performed to check the data integrity. If there is any data discrepancy, the "Verification failed" error message is displayed. Other error messages are also displayed when the image file size is greater than the allowed memory space and when the user aborts the task.
- STM32F10x Standard Peripherals Library.

a. The Ymodem protocol sends data in 1024-byte blocks. An error check is performed in data blocks transmitted to the STM32F10xxx's internal RAM to compare the transmitted and received data. Blocks unsuccessfully received are acknowledged with an NAK (Negative Acknowledgement). For more details about the Ymodem protocol, refer to existing documentation.

To select the STMicroelectronics evaluation board STM3210C-EVAL (STM32F10x Connectivity line), STM3210E-EVAL (STM32F10x High-Density) or STM3210B-EVAL (STM32F10x Medium-Density) used to run the IAP, uncomment the corresponding line in stm32_eval.h file (under Utilities\STM32_EVAL)

```
//#define USE_STM3210B_EVAL
//#define USE_STM3210E_EVAL
//#define USE_STM3210C_EVAL
```

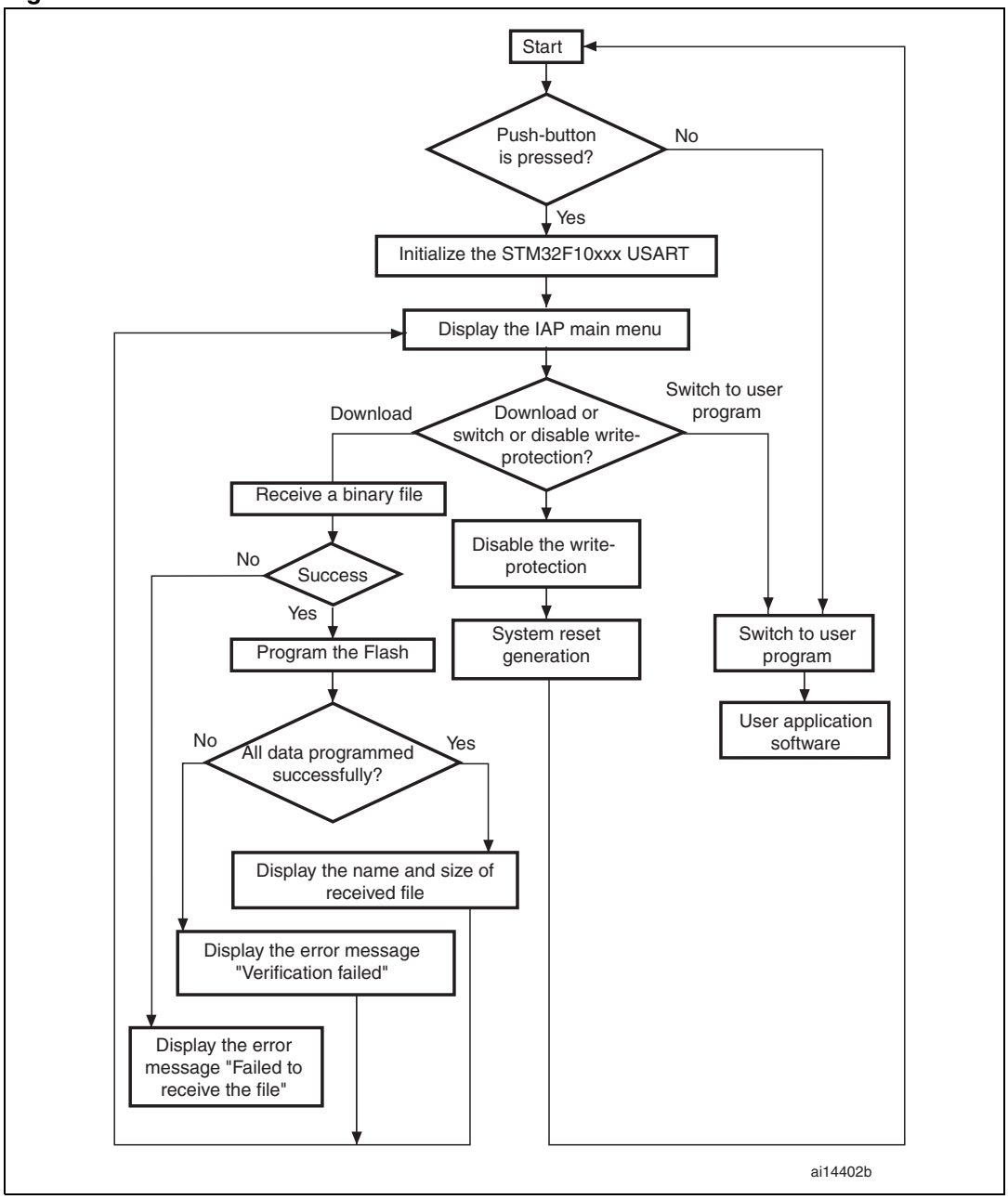
The user can choose to either go to the user application or execute the IAP for reprogramming purposes by pressing a push-button connected to a pin.

- Not pressing the push-button at reset switches to the user application
- Pressing the push-button at reset displays the IAP main menu

Refer to *Table 1.: STM32F10xxx IAP implementation* for more details about the STM3210B-EVAL, STM3210E-EVAL and STM3210C-EVAL board push-button used to enter the IAP mode.

The IAP flowchart is represented in *Figure 1*.

Figure 1. Flowchart of the IAP driver



2 Running the IAP driver

The IAP driver is programmed in Flash memory:

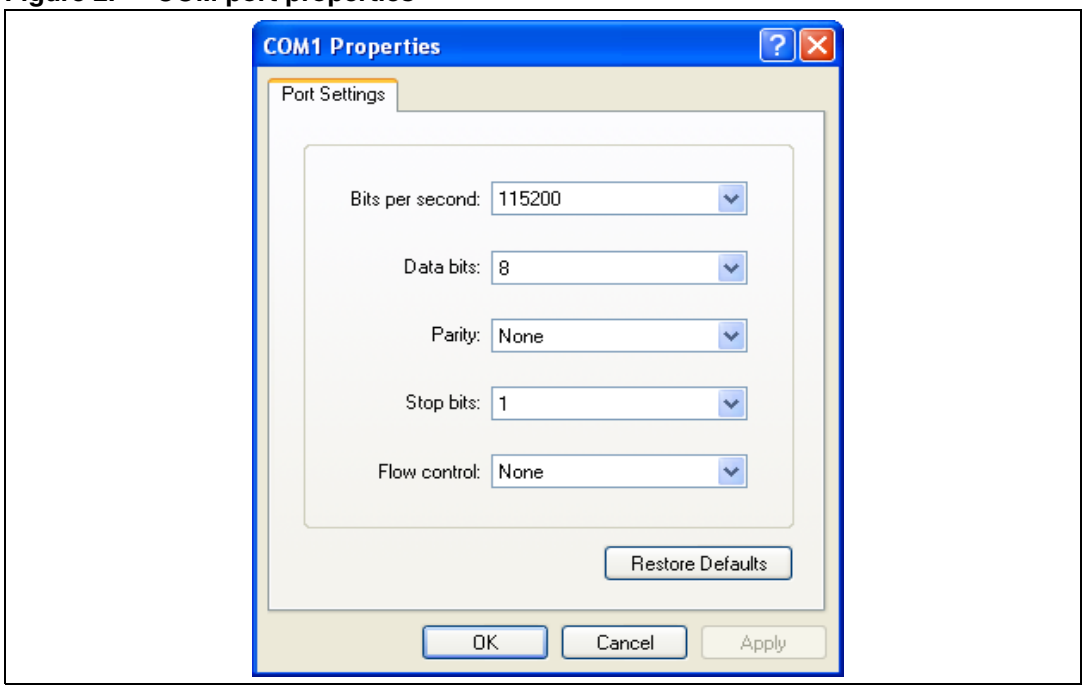
- from page 0 to page 7 on Medium-density devices
- from page 0 to page 3 on High-density and Connectivity Line devices

The user application occupies the remaining memory space.

2.1 HyperTerminal configuration

To use the IAP, the user must have a PC running HyperTerminal and configured as shown in *Figure 2*.

Figure 2. COM port properties



Note: The baud rate value of 115200 bps is used as an example.

Care must be taken when selecting the system clock frequency. To guarantee successful communication via the USART, the system clock frequency in the end application must be such that a baud rate equal to 115200 bps can be generated.

2.2 Executing the IAP driver

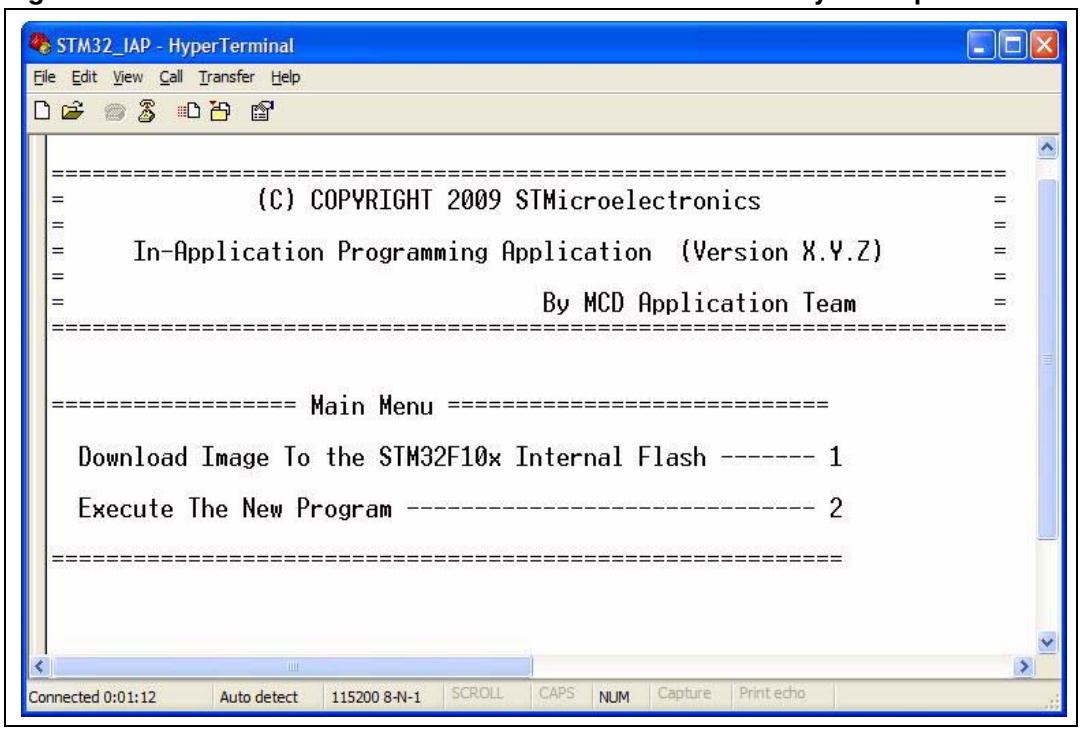
As an example in this application note, pressing the pin connected to the push-button allows the IAP driver to run.

By pressing the push-button at reset, the user can run the IAP driver to reprogram the STM32F10x's internal Flash memory. It is not mandatory to use the push-button; the user can apply a signal to this pin with respect to its active level. Refer to *Table 1.: STM32F10xxx IAP implementation*.

3 IAP driver menu

Running the IAP displays the following menu in the HyperTerminal window.

Figure 3. IAP Driver menu when the STM32F10x Flash memory is not protected



3.1 Download image to the internal Flash memory

To download a binary file via HyperTerminal to the STM32F10x's internal Flash memory, do as follows:

1. Press **1** on the keyboard to select the **Download Image To the STM32F10x Internal Flash** menu
2. Select **Send File** in the **Transfer** menu
3. In the **Filename** field, type the name and the path of the binary file you want to download
4. From the protocol list, select the **Ymodem** protocol
5. Click on the **Send** button

As a result, the IAP driver loads the binary file into the STM32F10x's internal Flash memory from the defined base address and displays the binary file name and size in the HyperTerminal window.

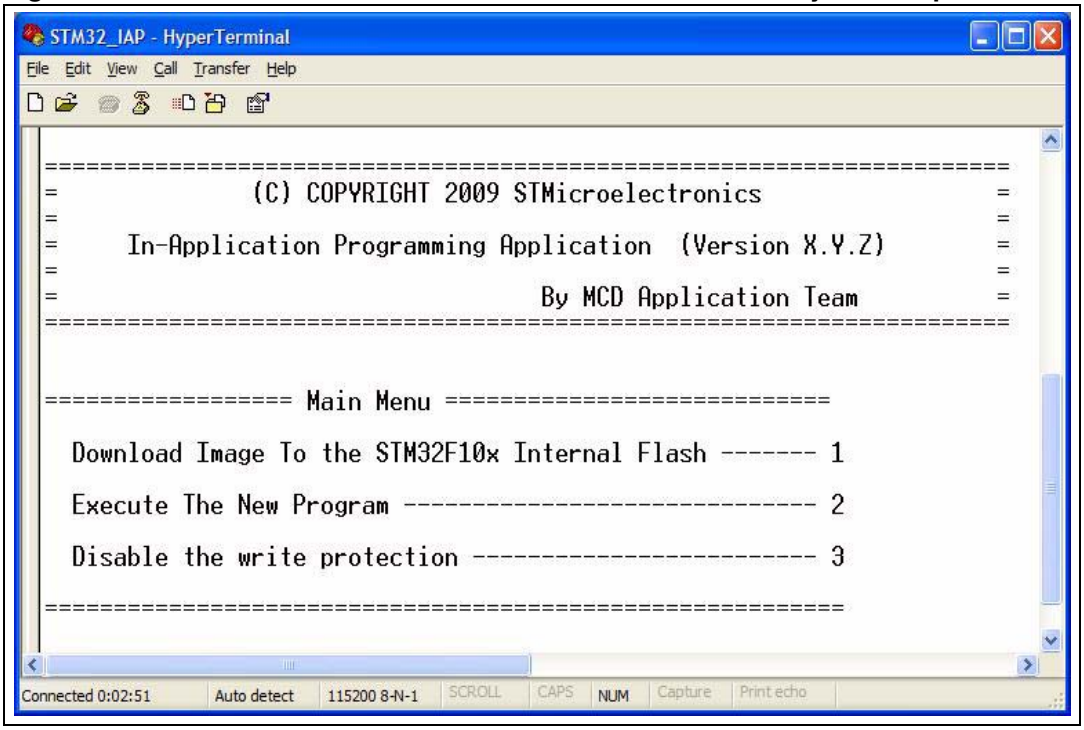
3.2 Execute the new program

Once the new program has been loaded, press **2** on the keyboard to select the **Execute The New Program** menu and execute the code.

3.3 Disabling the write protection

When the IAP starts, it checks the Flash memory pages where the user program is to be loaded to see if any are write-protected. If it is the case, the menu shown in *Figure 4* appears.

Figure 4. IAP driver menu when the STM32F10x Flash memory is write-protected



Prior to downloading the new program, the write protection must be disabled. To do so, press **3** (**Disable the write protection**) on the keyboard. The write protection is disabled and a system reset is generated to reload the new option byte values. After resuming from reset, the menu shown in *Figure 3* is displayed.

Note: In this application, the read protection is not supported, so the user has to verify that the Flash memory is not read-protected.

4 STM32F10x IAP implementation summary

Table 1 provides a summary of the STM32F10x IAP implementation.

Table 1. STM32F10xxx IAP implementation

	Firmware		Hardware	
	The IAP program is located at 0x8000000. The Flash routines (program/erase) are executed from the Flash memory. The size of this program is about 8 Kbytes and programmed on:	The user application (image to be loaded with the IAP) will be programmed starting from address 0x8002000(1) The maximum size of the image to be loaded is:	Push-button (active level: low)	USART used
Medium-density devices (STM3210B-EVAL)	page 0 to page 7	120 Kbytes (page 8 - page 127)	Key push-button connected to pin PB.09	USART1
High-density devices (STM3210E-EVAL)	page 0 to page 3	248 Kbytes (page 4 - page 127)	Key push-button connected to pin PG.08	USART1
Connectivity line devices (STM3210C-EVAL)	page 0 to page 3	504 Kbytes (page 4 - page 255)	Key push-button connected to pin PB.09	USART2 (Tx/Rx pins remapped)

1. User application location address is defined in the *common.h* file as: `#define ApplicationAddress 0x8002000`. To modify it, change the default value to the desired one.

The STM32F10xxx IAP package comes with:

- Source files and pre-configured projects for the IAP program (under Project\IAP directory)
- Source files and pre-configured projects that build the application to be loaded into Flash memory using the IAP (under Project\binary_template directory).

The readme.txt files provided within this package describes step by step how to execute this IAP application.

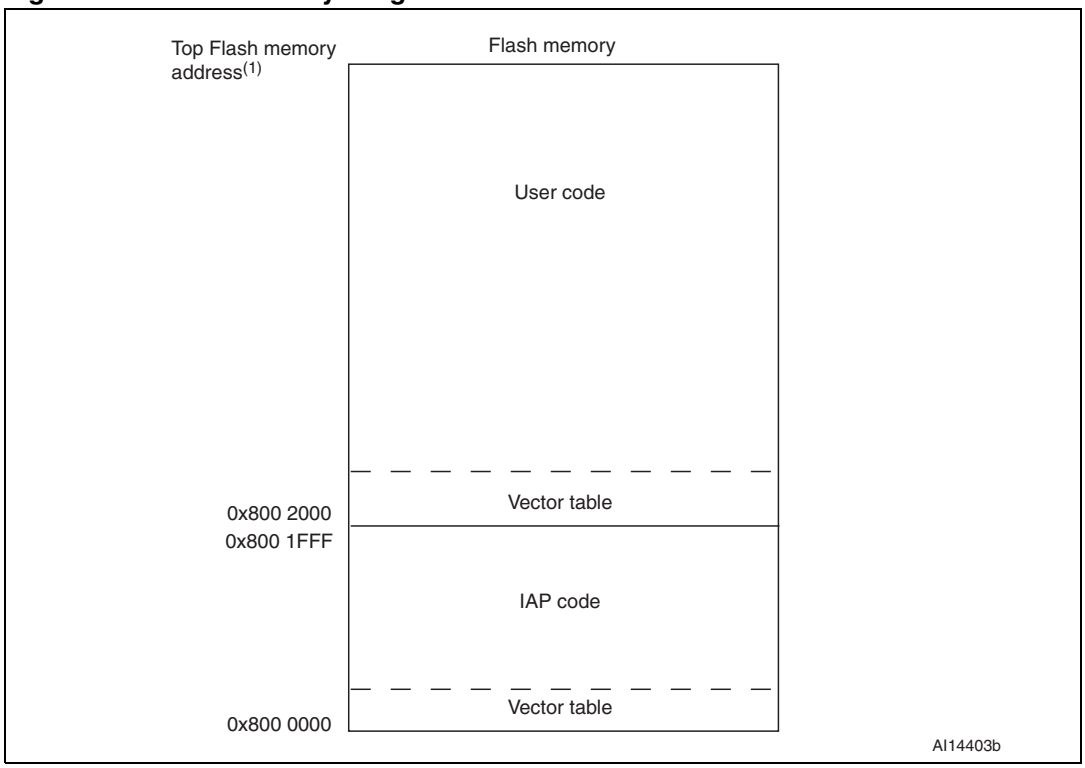
5 User program conditions

The user application to be loaded into the Flash memory using IAP should be built with these configuration settings:

1. Set the program load address at 0x08002000, using your toolchain linker file
2. Relocate the vector table at address 0x08002000, using the “NVIC_SetVectorTable” function.

An example application program to be loaded with the IAP application is provided with preconfigured projects.

Figure 5. Flash memory usage



1. Top Flash memory address is equal to 0x0801 FFFF for Medium-density devices, to 0x0803 FFFF for Connectivity Line devices and to 0x0807 FFFF for High-density devices.

6 Revision history

Table 2. Revision history

Date	Revision	Changes
05-Jun-2007	1	Initial release.
05-Oct-2007	2	The IAP driver can also be programmed from the SWD interface (see *Section 1.1: Principle*). Modified: – *Figure 3: IAP Driver menu when the STM32F10x Flash memory is not protected* – *Figure 4: IAP driver menu when the STM32F10x Flash memory is write-protected* – *Figure 6: IAP driver directory structure*. Flash routines modified in *Table 1.: STM32F10xxx IAP implementation*. *Section 5: User program conditions* updated. *Section 7: How to use the IAP driver* modified, RIDE project added.
30-May-2008	3	Application note updated to apply to High-density devices (implemented on the STM3210E-EVAL evaluation board). Small text changes. *Figure 1: Flowchart of the IAP driver on page 7* corrected.
13-Jun-2008	4	*Figure 6: IAP driver directory structure* modified.
31-Jul-2009	5	Updated for connectivity line devices.