AN2606
Application note

STM32™ microcontroller system memory boot mode

Introduction

The bootloader is stored in the internal boot ROM memory (system memory) of STM32 devices. It is programmed by ST during production. Its main task is to download the application program to the internal Flash memory through one of the available serial peripherals (USART, CAN, USB, I^2C, SPI, etc.). A communication protocol is defined for each serial interface, with a compatible command set and sequences.

This document applies to the products listed in *Table 1*. They are referred to as STM32 throughout the document.

Table 1. Applicable products

Type	Part number or product series
Microcontrollers	STM32F1 series STM32F2 series STM32F05 (Entry-level) products: – STM32F050x4 and STM32F050x6 – STM32F051x4, STM32F051x6 and STM32F051x8 STM32L1 series: – STM32L151xx, STM32L152xx and STM32L162xx – STM32L100 Value Line STM32F3 series: – STM32F302xx, STM32F303xx, STM32F313xx, STM32F372xx, STM32F373xx, STM32F383xx STM32F4 series: – STM32F405xx, STM32F407xx, STM32F415xx, STM32F417xx, STM32F427xx, STM32F437xx, STM32F429xx, STM32F439xx

The main features of the bootloader are the following:

- It uses an embedded serial interface to download the code with a predefined communication protocol
- It transfers and updates the Flash memory code, the data, and the vector table sections

This application note presents the general concept of the bootloader. It describes the supported peripherals and hardware requirements to be considered when using the bootloader of STM32 devices listed in *Table 1*. However the specifications of the low-level communication protocol for each supported serial peripheral are documented in separate documents. For specifications of the USART protocol used in the bootloader, refer to AN3155. For the specification of the CAN protocol used in the bootloader, refer to AN3154. For the specification of the DFU (USB device) protocol used in the bootloader, refer to AN3156. For the specification of the I2C protocol used in the bootloader, refer to AN4221.

Contents

List of tables

List of figures

1 Related documents

For each supported product (listed in *Table 1*), please refer to the following documents available from *http://www.st.com*:

- Datasheet or databrief
- Reference manual and/or flash programming manual

2 Glossary

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, STM32F102xx and STM32F103xx microcontrollers where the Flash memory density ranges between 64 and 128 Kbytes.

High-density devices are STM32F101xx and STM32F103xx microcontrollers where the Flash memory density ranges between 256 and 512 Kbytes.

Connectivity line devices are STM32F105xx and STM32F107xx microcontrollers.

Low-density value line devices are STM32F100xx microcontrollers where the Flash memory density ranges between 16 and 32 Kbytes.

Medium-density value line devices are STM32F100xx microcontrollers where the Flash memory density ranges between 64 and 128 Kbytes.

High-density value line devices are STM32F100xx microcontrollers where the Flash memory density ranges between 256 and 5128 Kbytes.

XL-density devices are STM32F101xx and STM32F103xx microcontrollers where the Flash memory density ranges between 768 Kbytes and 1 Mbyte.

Medium-density ultralow power devices are STM32L151xx and STM32L152xx microcontrollers where the program memory density ranges between 64 and 128 Kbytes.

Medium-density plus ultralow power devices are STM32L151xx and STM32L152xx microcontrollers where the program memory size is 256Kbytes.

High-density ultralow power devices are STM32L151xx, STM32L152xx and STM32L162xx microcontrollers where the program memory density size is 384 Kbytes.

STM32F051xx devices are STM32F051x4, STM32F051x6 and STM32F051x8 microcontrollers where the Flash memory density ranges between 32 and 64 Kbytes.

STM32F050xx devices are STM32F050x4 and STM32F050x6 microcontrollers where the Flash memory density ranges between 16 and 32 Kbytes.

STM32F2xxxx devices are STM32F215xx, STM32F205xx, STM32F207xx and SMT32F217xx microcontrollers with a Flash memory density ranging from 128 to 1024 Kbytes.

STM32F31xxx devices are STM32F313xx microcontrollers with the Flash memory density ranges between 128 and 256 Kbytes.

STM32F30xxx devices are STM32F302xx and STM32F303xx microcontrollers with the Flash memory density ranges between 128 and 256 Kbytes.

STM32F37xxx devices are STM32F372xx and STM32F373xx microcontrollers with the Flash memory density ranges between 128 and 256 Kbytes.

STM32F38xxx devices are STM32F383xx microcontrollers with the Flash memory density ranges between 128 and 256 Kbytes.

STM32F40xxx/41xxx devices are STM32F405xx, STM32F407xx, STM32F415xx and SMT32F417xx microcontrollers with a Flash memory density ranging from 512 to 1024 Kbytes.

STM32F427xx/437xx devices are STM32F427xx and STM32F437xx microcontrollers with the Flash memory density ranges between 1024 and 2048 Kbytes.

STM32F429xx/439xx devices are STM32F429xx and STM32F439xx microcontrollers with the Flash memory density ranges between 1024 and 2048 Kbytes.

Note: BL_USART_Loop refers to the USART Bootloader execution loop.
BL_CAN_Loop refers to the CAN Bootloader execution loop.
BL_I2C_Loop refers to the I2C Bootloader execution loop."

3 General bootloader description

3.1 Bootloader activation

The bootloader is automatically activated by configuring the BOOT0 and BOOT1 pins in the specific “System memory” configuration (see *Table 2*) and then by applying a reset.

Depending on the used pin configuration, the Flash memory, system memory or SRAM is selected as the boot space, as shown in *Table 2* below.

In some products, BOOT1 is not an I/O but a bit in the option byte area. This is the case for the STM32F05x and STM32F3xx devices where BOOT1 is configured through nBoot1 bit in the option bytes.

- When nBoot1 bit is set to 1, it corresponds to BOOT1 reset to 0 in *Table 2*
- When nBoot1 bit is reset to 0, it corresponds to BOOT1 set to 1 n *Table 2*.

Table 2. Boot pin configuration

Boot mode selection pins		Boot mode	Aliasing
BOOT1	BOOT0		
X	0	User Flash memory	User Flash memory is selected as the boot space
0	1	System memory	System memory is selected as the boot space
1	1	Embedded SRAM	Embedded SRAM is selected as the boot space

Table 2 shows that the STM32 microcontrollers enter System memory boot mode if the BOOT pins are configured as follows:

- BOOT0 = 1
- BOOT1 = 0

The values on the BOOT pins are latched on the fourth rising edge of SYSCLK after a reset.

Note: In some products, you may enter to bootloader with (BOOT0 = 0 and BOOT1 = x) when the dual bank boot feature capability is available in STM32 products. Refer to section Dual bank boot feature in product section for more information.

Temperature outside ambient range might cause Bootloader to behave incorrectly due to internal clock (HSI) variation vs. temperature, inducing corruption of serial communication protocol clock.

3.2 Exiting System memory boot mode

System memory boot mode must be exited in order to start execution of the application program. This can be done by applying a hardware reset. During reset, the BOOT pins/bits (BOOT0 and BOOT1) must be set at the proper levels to select the desired boot mode (see *Table 2*). Following the reset, the CPU starts code execution from the boot memory located at the bottom of the memory address space starting from 0x0000 0000.

3.3 Bootloader identification

Depending on the STM32 device used, the bootloader may support one or more embedded serial peripherals used to download the code to the internal Flash memory. The bootloader identifier (ID) provides information about the supported serial peripherals.

For a given STM32 device, the bootloader is identified by means of the:

1. **Bootloader (protocol) version**: version of the serial peripheral (USART, CAN, USB, etc.) communication protocol used in the bootloader. This version can be retrieved using the bootloader Get Version command.
2. **Bootloader identifier (ID)**: version of the STM32 device bootloader, coded on one byte in the **0xXY** format**,** where:
   - **X** specifies the embedded serial peripheral(s) used by the device bootloader:
     X = 1: only one USART is used
     X = 2: two USARTs are used
     X = 3: two USARTs, one CAN and DFU are used
     X = 4: two USARTs and DFU are used
     X = 5: two USARTs and I^2C are used
     X = 6: only one I^2C is used
     X = 7: two USARTs, one CAN, DFU and one I^2C are used
     X = 8: one I^2C and one SPI are used
     X = 9: two USARTs, one CAN, DFU, one I^2C and one SPI are used
     X = 10: two USARTs, DFU and one I^2C are used
     X = 11: two USARTs, one I^2C and one SPI are used
     X = 12: two USARTs and one SPI are used
     X = 13: two USARTs, DFU, one I^2C and one SPI are used
   - **Y** specifies the device bootloader version

     Let us take the example of a bootloader ID equal to 0x10. This means that it is the first version of the device bootloader that uses only one USART.
     The bootloader ID is programmed in the last byte address - 1 of the device system memory and can be read by using the bootloader “Read memory” command or by direct access to the system memory via JTAG/SWD.

The table below provides identification information about the bootloader embedded in STM32 devices.

Table 3. Embedded bootloaders

STM32 family	Device	Supported serial peripherals	Bootloader ID		Bootloader (protocol) version
			ID	Memory location	
F1	Low-density	USART1	NA	NA	USART (V2.2)
	Medium-density	USART1	NA	NA	USART (V2.2)
	High-density	USART1	NA	NA	USART (V2.2)
	Connectivity line	USART1 / USART2 (remapped) / CAN2 (remapped) / DFU (USB Device)	NA	NA	USART (V2.2[1]) CAN (V2.0) DFU(V2.2)
	Medium-density value line	USART1	V1.0	0x1FFFF7D6	USART (V2.2)
	High-density value line	USART1	V1.0	0x1FFFF7D6	USART (V2.2)
	XL-density	USART1/USART2 (remapped)	V2.1	0x1FFFF7D6	USART (V3.0)
L1	Medium-density ultralow power	USART1/USART2	V2.0	0x1FF00FFE	USART (V3.0)
	High-density ultralow power	USART1/USART2/DFU (USB Device FS)	V4.5	0x1FF01FFE	USART (V3.1)/ DFU (V2.2)
	Medium-density plus ultralow power	USART1/USART2/DFU (USB Device FS)	V4.0	0x1FF01FFE	USART (V3.1)/ DFU (V2.2)
F2	STM32F2xxxx	USART1/USART3	V2.0	0x1FFF77DE	USART (V3.0)
		USART1/USART3/CAN2/ DFU (USB Device FS)	V3.3	0x1FFF77DE	USART (V3.1)/ CAN (V2.0)/ DFU (V2.2)
F0	STM32F051xx	USART1/USART2	V2.1	0x1FFFF7A6	USART (V3.1)
	STM32F050xx	USART1	V1.0	0x1FFFF7A6	USART (V3.1)
F4	STM32F40xxx/ 41xxx	USART1/USART3/CAN2/ DFU (USB Device FS)	V3.1	0x1FFF77DE	USART (V3.1)/ CAN (V2.0)/ DFU (V2.2)
	STM32F427xx/ 437xx	USART1/USART3/CAN2 /DFU (USB Device FS)	V3.0	0x1FFF76DE	USART (V3.1)/ CAN (V2.0)/ DFU (V2.2)
	STM32F429xx/ 439xx	USART1/USART3/CAN2 /DFU (USB Device FS)/I2C	V7.0	0x1FFF76DE	USART (V3.1)/ CAN (V2.0)/ DFU (V2.2)/ I2C (V1.0)

Table 3. Embedded bootloaders (continued)

STM32 family	Device	Supported serial peripherals	Bootloader ID		Bootloader (protocol) version
			ID	Memory location	
F3	STM32F37xxx	USART1/USART2/DFU (USB Device FS)	V4.1	0x1FFFF7A6	USART (V3.1)/ DFU (V2.2)
	STM32F30xxx	USART1/USART2/DFU (USB Device FS)	V4.1	0x1FFFF796	USART (V3.1)/ DFU (V2.2)
	STM32F38xxx	USART1/USART2/I2C1	V5.0	0x1FFFF7A6	USART (V3.1)/ I2C (V1.0)
	STM32F31xxx	USART1/USART2/I2C1	V5.0	0x1FFFF796	USART (V3.1)/ I2C (V1.0)

1. For connectivity line devices, the USART bootloader returns V2.0 instead of V2.2 for the protocol version. For more details please refer to the “STM32F105xx and STM32F107xx revision Z” errata sheet available from *http://www.st.com*.

4 STM32F100xx, STM32F101xx, STM32F102xx, STM32F103xx medium-density and high-density value line bootloader

Throughout this section, STM32F10xxx is used to refer to low-density, medium-density, high-density STM32F101xx and STM32F103xx devices, to low- and medium-density STM32F102xx devices, to low-, medium-, and high-density STM32F100xx, and to medium and high-density value line devices.

4.1 Bootloader configuration

The bootloader embedded in STM32F10xxx devices supports only one interface: the USART1.

The following table shows the required STM32F10xxx hardware resources used by the bootloader in System memory boot mode.

Table 4. STM32F10xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	Clock source	HSI enabled	The system clock is equal to 24 MHz using the PLL.
	RAM	-	512 bytes starting from address 0x20000000 are used by the bootloader firmware.
	System memory	-	2 Kbytes starting from address 0x1FFFF000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 receives.
	USART1_TX pin	Output push-pull	PA9 pin: USART1 transmits.
	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in the above table) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. So, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

4.2 Bootloader hardware requirements

The hardware required to put the STM32 into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

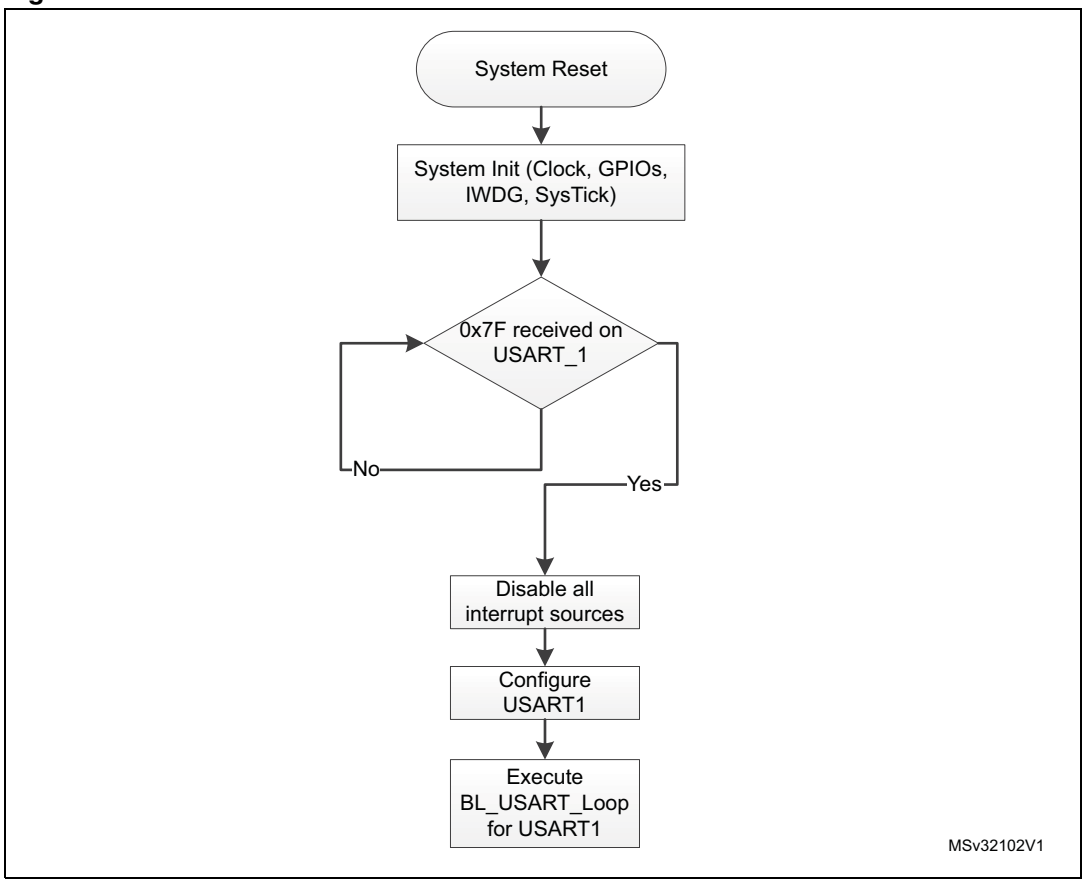
To connect to the STM32 during System memory boot mode, an RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly linked to the USART1_RX (PA10) and USART1_TX (PA9) pins.

Note: USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore user can use these pins for other peripherals or GPIOs.

For more details about hardware recommendations, refer to application note AN2586: "STM32 hardware development: getting started", available from the STMicroelectronics website: http://www.st.com.

4.3 Bootloader selection

Figure 1. Bootloader for STM32F10xxx with USART1



Once System memory boot mode is entered and the microcontroller has been configured as described above, the bootloader code begins to scan the USART1_RX line pin, waiting to receive the 0x7F data frame: one start bit, 0x7F data bits, even parity bit and one stop bit.

The duration of this data frame is measured using the Systick timer. The count value of the timer is then used to calculate the corresponding baud rate factor with respect to the current system clock.

Next, the code initializes the serial interface accordingly. Using this calculated baud rate, an acknowledge byte (0x79) is returned to the host, which signals that the STM32F10xxx is ready to receive user commands.

4.4 Bootloader version

Table 5 lists the bootloader versions of the STM32F10xxx devices.

Table 5. STM32F10xxx bootloader versions

Bootloader version number	Description
V2.0	Initial bootloader version.
V2.1	– Updated Go Command to initialize the main stack pointer – Updated Go command to return NACK when jump address is in the Option byte area or System memory area – Updated Get ID command to return the device ID on two bytes – Update the bootloader version to V2.1
V2.2	– Updated Read Memory, Write Memory and Go commands to deny access with a NACK response to the first 0x200 bytes of RAM memory used by the bootloader – Updated Readout Unprotect command to initialize the whole RAM content to 0x0 before ROP disable operation

5 STM32F105xx and STM32F107xx device bootloader

5.1 Bootloader configuration

The bootloader embedded in the STM32F105xx and STM32F107xx devices supports four serial peripherals: USART1, USART2, CAN2, and DFU (USB). This means that four serial peripherals are supported: USART1, USART2, CAN2 and DFU (USB).

The following table shows the hardware resources required by STM32F105xx and STM32F107xx devices used by the bootloader in System memory boot mode.

Table 6. STM32F105xx/107xx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock frequency is 24 MHz using the PLL. This is used only for USART1 and USART2 bootloaders and during CAN2, USB detection for CAN and DFU bootloaders (Once CAN or DFU bootloader is selected, the clock source will be derived from external crystal).
		HSE enabled	The external clock is mandatory only for DFU and CAN bootloaders and it must provide one of the following frequencies: 8 MHz, 14.7456 MHz or 25 MHz. For CAN bootloader, the PLL is used only to generate 48 MHz when 14.7456 MHz is used as HSE. For DFU bootloader, the PLL is used to generate a 48 MHz system clock from all supported external clock frequencies.
		-	The clock security system (CSS) interrupt is enabled for the CAN and DFU bootloaders. Any failure (or removal) of the external clock will generate system reset.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	System memory	-	18 Kbytes starting from address 0x1FFF B000 contain the bootloader firmware.
	RAM	-	4 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 receives.
	USART1_TX pin	Output push-pull	PA9 pin: USART1 transmits.
	USART2_RX (PD6), CAN2_RX (PB5), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		

Table 6. STM32F105xx/107xx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 receive (remapped pin)
	USART2_TX pin	Output push-pull	PD5 pin: USART2 transmit (remapped pin)
	USART1_RX (PA10), CAN2_RX (PB5), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
CAN2 bootloader	CAN2	Enabled	Once initialized, the CAN2 configuration is: Baudrate 125 kbps, 11-bit identifier. **Note:** CAN1 is clocked during the CAN bootloader execution because in STM32F105xx and STM32F107xx devices, CAN1 manages the communication between CAN2 and SRAM.
	CAN2_RX pin	Input	PB5 pin: CAN2 receives (remapped pin).
	CAN2_TX pin	Output push-pull	PB6 pin: CAN2 transmits (remapped pin).
	USART1_RX (PA10), USART2_RX (PD6), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB OTG FS	Enabled	USB OTG FS configured in Forced Device mode
	OTG_FS_VBUS pin	Input or alternate function, automatically controlled by the USB OTG FS controller	PA9: Power supply voltage line
	OTG_FS_DM pin		PA11: USB Send-Receive data line
	OTG_FS_DP pin		PA12: USB Send-Receive data line
	Interrupts	Enabled	USB_OTG_FS interrupt vector is enabled and used for USB DFU communication.
	USART1_RX (PA10), USART2_RX (PD6) and CAN2_RX (PB5) pins must be kept at a high or low level during the detection phase.		

The system clock is derived from the embedded internal high-speed RC for USARTx bootloader. This internal clock is used also for DFU and CAN bootloaders but only for the selection phase. An external clock (8 MHz, 14.7456 MHz or 25 MHz.) is required for DFU and CAN bootloader execution after the selection phase.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) will be initialized to their default reset values before jumping to the user application.
If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

5.2 Bootloader hardware requirements

The hardware required to put the STM32F105xx and STM32F107xx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F105xx and STM32F107xx during System memory boot mode, the following conditions have to be verified:

- The RX pins of the unused peripherals in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6), CAN2_RX (PB5), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10), CAN2_RX (PB5), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If CAN2 is used to connect to the bootloader: the USART1_RX (PA10), USART2_RX (PD6), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU is used to connect to the bootloader: the USART1_RX (PA10), USART2_RX (PD6) and CAN2_RX (PB5) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral to be performed through:
  - an RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used
  - a CAN interface (CAN transceiver) has to be directly connected to the CAN2_RX (PB5) and CAN2_TX (PB6) pins
  - a certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used)

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART2.

Once the USB Device is enabled, all its related pins are dedicated to USB communication only, and cannot be used for other application purposes.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM3210C-EVAL board. For more details about this, refer to document: “STM3210C-EVAL board user manual”, available from the STMicroelectronics website: *http://www.st.com*.

5.3 Bootloader selection

The STM32F105xx and STM32F107xx embedded bootloader supports four peripherals interfaces: USART1, USART2, CAN2 and DFU (USB). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 5.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baud rate sequence and then enters a loop, waiting for any USART bootloader command.

To use the CAN2 interface, connect the CAN cable to CAN2. Once the bootloader detects a frame on the CAN2_RX pin (PB5), the bootloader firmware enters a CAN loop and starts to check the external clock frequency value, if the HSE is 8 MHz, 14.7456 MHz or 25 MHz CAN bootloader firmware enters an infinite loop and waits until it receives a message, otherwise a system reset is generated.

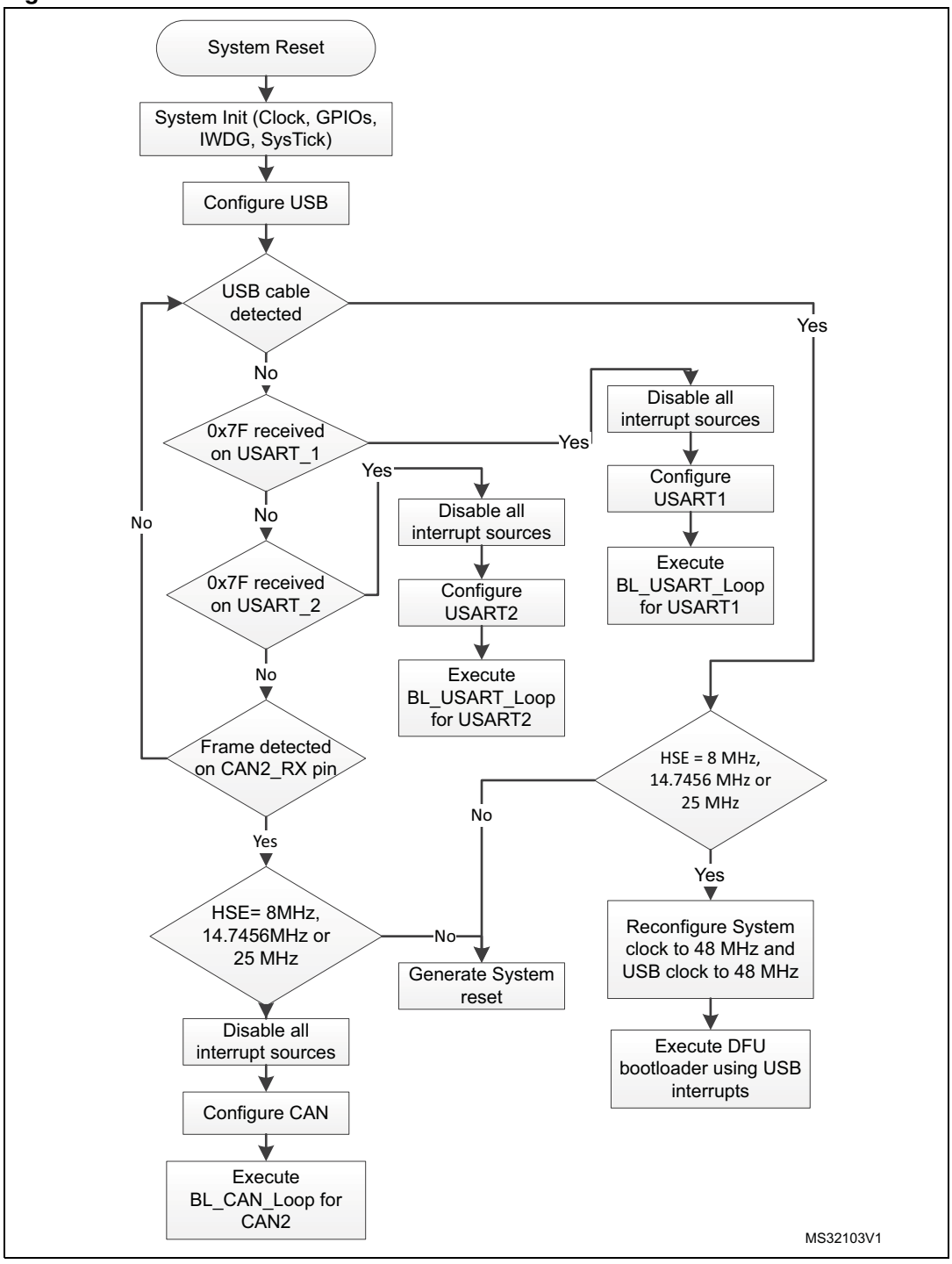
If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader then enters the DFU bootloader loop waiting for any DFU bootloader command.

To use the USART or the CAN bootloader, it is mandatory that no USB cable is connected to the USB peripheral during the selection phase. Once the USART or CAN bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except commands which generate a system reset.

Once one interface is selected for the bootloader, all other interfaces are disabled.

The figure below shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 2. Bootloader selection for STM32F105xx and STM32F107xx devices



5.4 Bootloader version

The table below lists the bootloader versions and the changes between all versions of the STM32F105xx and STM32F107xx devices.

Table 7. STM32F105xx and STM32F107xx bootloader versions

Bootloader version number	Description
V1.0	Initial bootloader version.
V2.0	– Bootloader detection mechanism updated to fix the issue when GPIOs of unused peripherals in this bootloader are connected to low level or left floating during the detection phase. For more details please refer to *Section 5.4.2*. – Vector table set to 0x1FFF B000 instead of 0x0000 0000 – Go command updated (for all bootloaders): USART1, USART2, CAN2, GPIOA, GPIOB, GPIOD and SysTick peripheral registers are set to their default reset values – DFU bootloader: USB pending interrupt cleared before executing the Leave DFU command – DFU subprotocol version changed from V1.0 to V1.2 – Bootloader version updated to V2.0
V2.1	– Fixed PA9 excessive consumption described in *Section 5.4.4*. – Get-Version command (defined in AN3155) corrected. It returns 0x22 instead of 0x20 in bootloader V2.0. Refer to *Section 5.4.3* for more details. – Bootloader version updated to V2.1
V2.2	– Fixed DFU option bytes descriptor (set to ‘e’ instead of ‘g’ because it is read/write and not erasable). – Fixed DFU polling timings for Flash Read/Write/Erase operations. – Robustness enhancements for DFU bootloader interface. – Updated bootloader version to V2.2.

5.4.1 How to identify STM32F105xx/107xx bootloader versions

Bootloader V1.0 is implemented on devices which date code is below 937 (refer to STM32F105xx and STM32F107xx datasheet for where to find the date code on the device marking). Bootloader V2.0 and V2.1 are implemented on devices with a date code higher or equal to 937.

There are two ways to distinguish between bootloader versions:

- When using the USART bootloader, the Get-Version command defined in AN2606 and AN3155 has been corrected in V2.1 version. It returns 0x22 instead of 0x20 as in bootloader V2.0.

- The values of the vector table at the beginning of the bootloader code are different. The user software (or via JTAG/SWD) reads 0x1FFFE945 at address 0x1FFFB004 for bootloader V2.0 0x1FFFE9A1 for bootloader V2.1, and 0x1FFFE9C1 for bootloader V2.2.
- The DFU version is the following:
  - V2.1 in bootloader V2.1
  - V2.2 in bootloader V2.2.

  It can be read through the bcdDevice field of the DFU Device Descriptor.

5.4.2 Bootloader unavailability on STM32F105xx/STM32F107xx devices with a date code below 937

Description

The bootloader cannot be used if the USART1_RX (PA10), USART2_RX (PD6, remapped), CAN2_Rx (PB5, remapped), OTG_FS_DM (PA11), and/or OTG_FS_DP (PA12) pin(s) are held low or left floating during the bootloader activation phase.

The bootloader cannot be connected through CAN2 (remapped), DFU (OTG FS in Device mode), USART1 or USART2 (remapped).

On 64-pin packages, the USART2_RX signal remapped PD6 pin is not available and it is internally grounded. In this case, the bootloader cannot be used at all.

Workaround

- For 64-pin packages

  None. The bootloader cannot be used.
- For 100-pin packages

  Depending on the used peripheral, the pins for the unused peripherals have to be kept at a high level during the bootloader activation phase as described below:
  - If USART1 is used to connect to the bootloader, PD6 and PB5 have to be kept at a high level.
  - If USART2 is used to connect to the bootloader, PA10, PB5, PA11 and PA12 have to be kept at a high level.
  - If CAN2 is used to connect to the bootloader, PA10, PD6, PA11 and PA12 have to be kept at a high level.
  - If DFU is used to connect to the bootloader, PA10, PB5 and PD6 have to be kept at a high level.

Note: This limitation applies only to STM32F105xx and STM32F107xx devices with a date code below 937. STM32F105xx and STM32F107xx devices with a date code higher or equal to 937 are not impacted. See STM32F105xx and STM32F107xx datasheet for where to find the date code on the device marking.

5.4.3 USART bootloader Get-Version command returns 0x20 instead of 0x22

Description

In USART mode, the Get-Version command (defined in AN3155) returns 0x20 instead of 0x20.

This limitation is present on bootloader versions V1.0 and V2.0, while it is fixed in bootloader version 2.1.

Workaround

None.

5.4.4 PA9 excessive power consumption when USB cable is plugged in bootloader V2.0

Description

When connecting an USB cable after booting from System-Memory mode, PA9 pin (connected to V_{BUS}=5 V) is also shared with USART TX pin which is configured as alternate push-pull and forced to 0 since the USART peripheral is not yet clocked. As a consequence, a current higher than 25 mA is drained by PA9 I/O and may affect the I/O pad reliability.

This limitation is fixed in bootloader version 2.1 by configuring PA9 as alternate function push-pull when a correct 0x7F is received on RX pin and the USART is clocked. Otherwise, PA9 is configured as alternate input floating.

Workaround

None.

6 STM32F101xx and STM32F103xx XL-density device bootloader

Throughout this section, STM32F10xxx XL-density is used to refer to XL-density STM32F101xx and STM32F103xx devices.

6.1 Dual bank boot feature

For STM32F101xx and STM32F103xx XL-density devices (these devices have two Flash memory banks: Bank 1 and Bank 2), an additional boot mechanism is available which allows booting from Bank 2 or Bank 1 (depending on the BFB2 bit status (bit 19 in the user option bytes @ 0x1FFFF800)).

1. When the BFB2 bit is reset, and the boot pins are configured to boot from the Flash memory (BOOT0 = 0 and BOOT1 = x) then, after reset, the device boots from the System memory and executes the embedded bootloader code which implements the dual bank Boot mode:
   a) First, the code checks Bank 2. If it contains a valid code (see *Note:* below), it jumps to application located in Bank 2 and leaves the bootloader.
   b) If the Bank 2 code is not valid, it checks Bank 1 code. If it is valid (see "*note*" below), it jumps to the application located in Bank 1.
   c) If both Bank 2 and Bank 1 do not contain valid code (see "*note*" below), the normal bootloader operations are executed as described in the following sections (no jump to Flash banks is executed). Refer to *Figure 3: Bootloader selection for STM32F10xxx XL-density devices* for more details.
2. When the bit BFB2 is set (default state), the dual bank boot mechanism is not performed.

Note: The code is considered as valid when the first data (at the bank start address, which should be the stack pointer) points to a valid address into the internal SRAM memory (stack top address). If the first address points to any other location (out of the internal SRAM) the code is considered not valid.

A dual bank boot mode example (FLASH\Dual_Boot) is provided within the STM32F10x Standard Peripheral Library available on http://www.st.com.

For the STM32F101xx and STM32F103xx XL-density devices, the Flash memory, system memory or SRAM is selected as the boot space, as shown in *Table 8* below.

Table 8. Boot pin and BFB2 bit configuration

BFB2 bit	Boot mode selection pins		Boot mode	Aliasing
	BOOT1	BOOT0		
1	X	0	User Flash memory	User Flash memory is selected as the boot space
	0	1	System memory	System memory is selected as the boot space
	1	1	Embedded SRAM	Embedded SRAM is selected as the boot space
0	X	0	System memory	System memory is selected as the boot space then dual bank mechanism is executed
	0	1	System memory	System memory is selected as the boot space then dual bank mechanism is executed
	1	1	Embedded SRAM	Embedded SRAM is selected as the boot space

Table 8 shows that the XL-density devices enter System memory boot mode in two cases:

1. If the BOOT pins are configured as follows: BOOT0 = 1 and BOOT1 = 0
2. Or if:
   a) the BFB2 bit is reset and
   b) boot pins are configured as follows: BOOT0 = 0 and BOOT1 = x

Note: When conditions a, b, and c below are fulfilled, it is equivalent to configuring boot pins for system memory boot (BOOT0 = 1 and BOOT1 = 0). In this case normal bootloader operations are executed.

a) BFB2 bit is reset

b) Both banks don't contain valid code

c) Boot pins configured as follows: BOOT0 = 0 and BOOT1 = x

When the BFB2 bit is cleared, and Bank 2 and/or Bank 1 contain valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code and never continues normal operations).
Consequently, if you have cleared the BFB2 bit (to boot from Bank 2) then, to be able to execute the bootloader code, you have to either:

- program the content of address 0x0808 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0, or

- set the BFB2 bit to 1, BOOT0 = 1 and BOOT1 = 0.

6.2 Bootloader configuration

The bootloader embedded in STM32F10xxx XL-density supports two serial interfaces: USART1 and USART2.

The following table shows the required hardware resources of STM32F10xxx XL-density devices used by the bootloader in System memory boot mode.

Table 9. STM32F10xxx XL-density configuration in System memory boot mode

Bootloader	Feature/peripheral	State	Comment
Common to all bootloaders	Clock source	HSI enabled	The system clock is equal to 24 MHz using the PLL.
	RAM	-	2 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.
	System memory	-	6 Kbytes starting from address 0x1FFF E000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 receives.
	USART1_TX pin	Output push-pull	PA9 pin: USART1 transmits.
	USART2_RX (PD6) pin must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit.
	USART2_RX pin	Input	PD6 pin: USART2 receives (remapped pins).
	USART2_TX pin	Output push-pull	PD5 pin: USART2 transmits (remapped pins).
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in *Table 9*) are initialized to their default reset values before jumping to the user application.
If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

6.3 Bootloader hardware requirements

The hardware required to put the STM32F10xxx XL-density devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

Note: As explained in Section 6.1: Dual bank boot feature, the System memory boot mode can also be executed by software when the BFB2 bit is reset, both banks start addresses are erased, and boot pins are configured to boot from Flash memory.

To connect to the STM32F10xxx XL-density devices during System memory boot mode, the following conditions have to be verified:

- The RX pin of the peripherals unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If the USART1 is used to connect to the bootloader: the USART2_RX (PD6) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If the USART2 is used to connect to the bootloader: the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- When the BFB2 bit is cleared, and Bank 2 and/or Bank 1 contain a valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code and never continues normal operations). Consequently, if you have cleared the BFB2 bit (to boot from Bank 2), then to be able to execute the bootloader code, you have to either:
  - program the content of address 0x0808 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0, or
  - set the BFB2 bit to 1, BOOT0 = 1 and BOOT1 = 0.
- Connection to the peripheral to be performed through:
  - an RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. This is also applicable for USART2.

6.4 Bootloader selection

The STM32F10xxx XL-density embedded bootloader supports two peripheral interfaces: USART1 and USART2. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

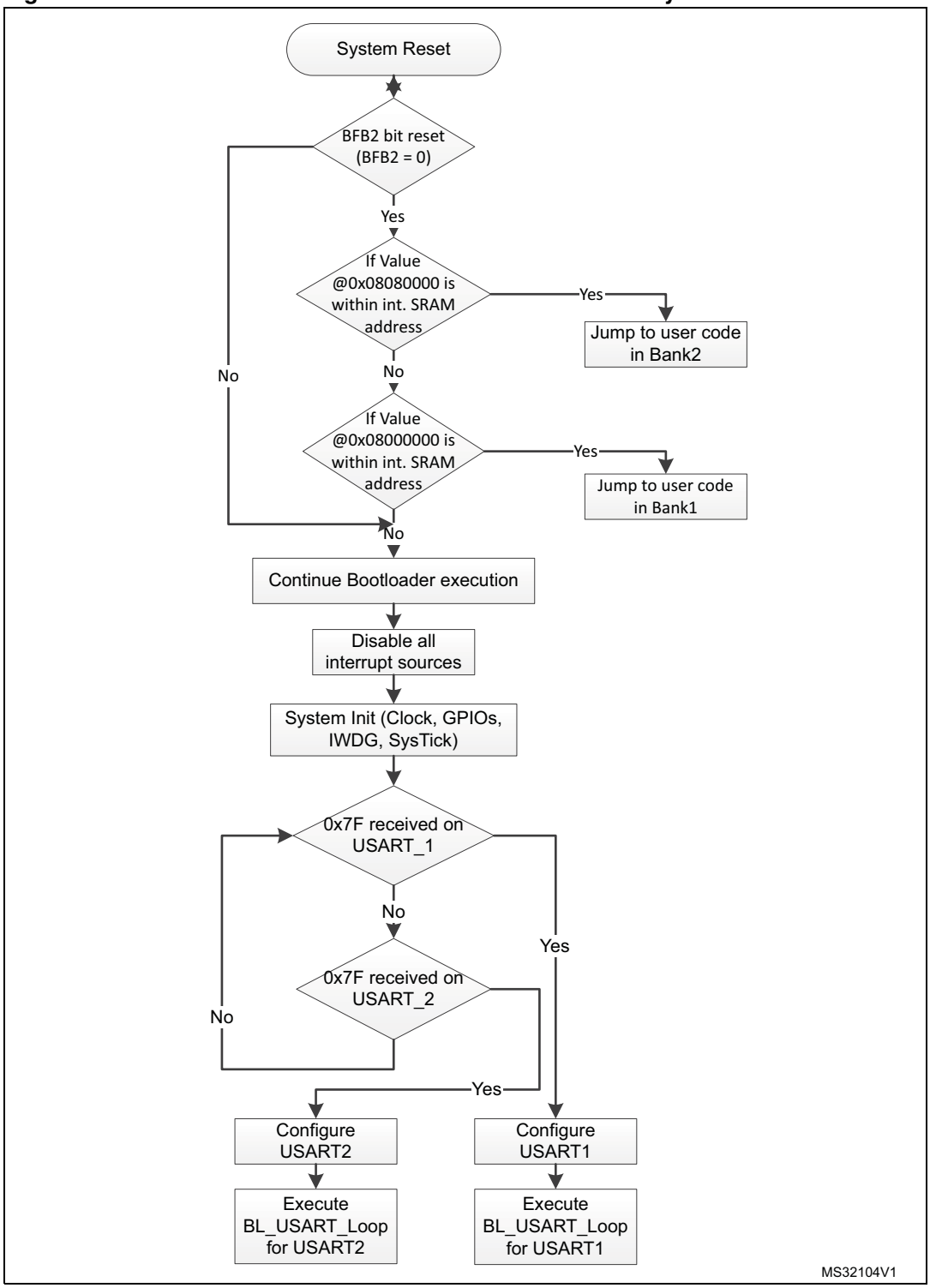
Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 6.3: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baudrate sequence and then enters a loop, waiting for any USART bootloader command.

Once one interface is selected for the bootloader, the other interface is disabled.

Figure 3 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 3. Bootloader selection for STM32F10xxx XL-density devices



6.5 Bootloader version

Table 10 lists the bootloader versions for the STM32F101xx and STM32F103xx XL-density devices.

Table 10. XL-density bootloader versions

Bootloader version number	Description
V2.1	Initial bootloader version

7 STM32L151xx, STM32L152xx and STM32L100xx medium-density ultralow power device bootloader

Throughout this section, STM32L15xxx medium-density is used to refer to medium-density STM32L151xx and STM32L152xx ultralow power devices.

7.1 Bootloader configuration

The bootloader embedded in STM32L100xx value line and STM32L15xxx medium-density devices supports two serial interfaces: USART1 and USART2 peripherals.

The following table shows the required hardware resources of STM32L100xx value line and STM32L15xxx medium-density devices used by the bootloader in System memory boot mode.

Table 11. STM32L100xx and STM32L15xxx configuration in System memory boot mode

Bootloader	Feature/peripheral	State	Comment
Common to all bootloaders	Clock source	HSI enabled	The system clock is equal to 16 MHz.
	RAM	-	2 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
	System memory	-	4 Kbytes starting from address 0x1FF00000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to Voltage Range 1.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 receives.
	USART1_TX pin	Output	PA9 pin: USART1 transmits.
	USART2_RX (PD6) pin must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit.
	USART2_RX pin	Input	PD06 pin: USART2 receives.
	USART2_TX pin	Output	PD05 pin: USART2 transmits.
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application. If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

7.2 Bootloader hardware requirements

The hardware required to put the STM32L15xxx medium-density devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32L15xxx medium-density devices during System memory boot mode, the following conditions have to be verified:

- The RX pins of the peripherals unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- The peripheral to be used has to be connected through an RS-232 serial interface (example, ST3232 RS-232 transceiver) which must be:
  - Directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used
  - Directly connected to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART2.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM32L152-EVAL board. For more details about this, refer to the “STM32L152-EVAL board user manual” (UM1018), available from the STMicroelectronics website: *http://www.st.com*.

7.3 Bootloader selection

The STM32L100xx value line and STM32L15xxx medium-density devices embedded bootloader supports two peripherals interfaces: USART1 and USART2. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 7.2: Bootloader hardware requirements for more information.

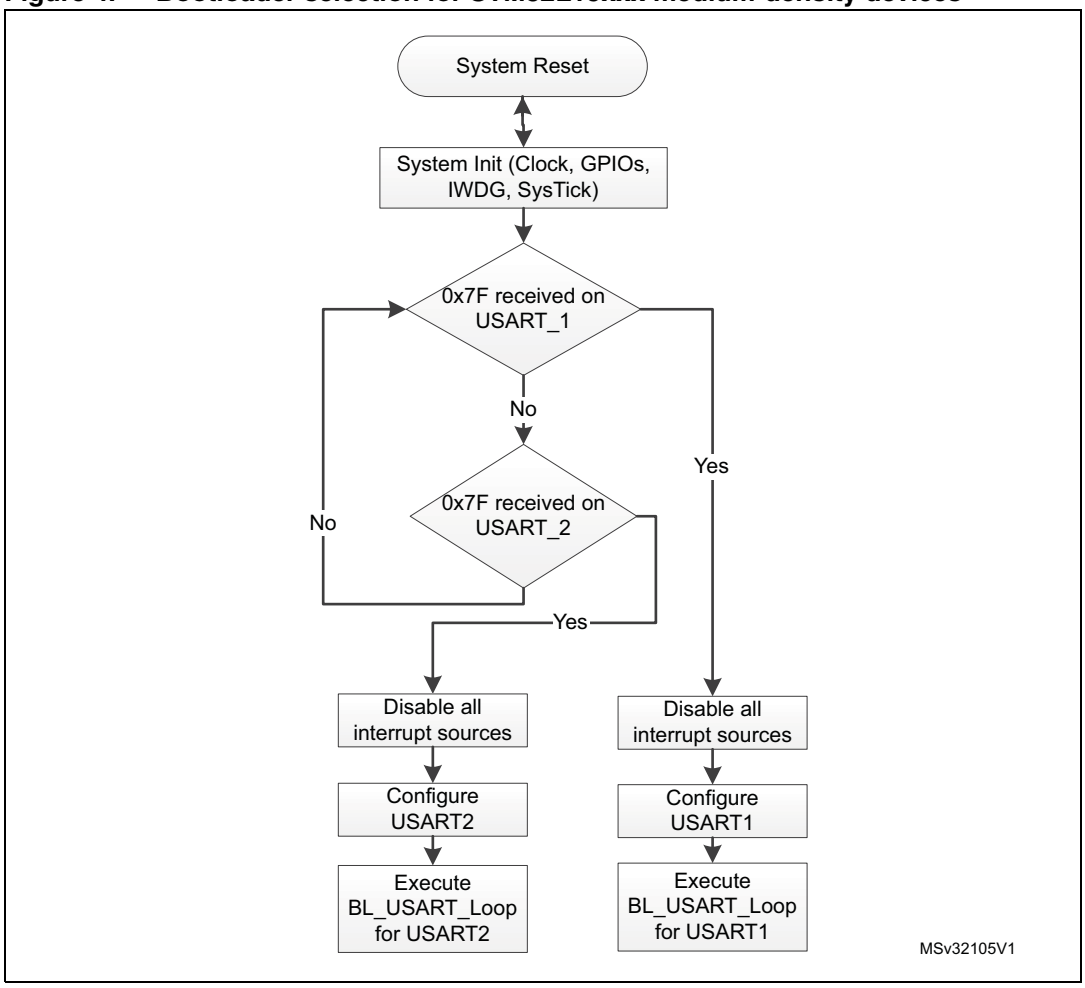
To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface.

Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the autobaudrate sequence and then enters a loop, waiting for any USART bootloader command.

Once one interface is selected for the bootloader, the other interface is disabled.

The figure below shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 4. Bootloader selection for STM32L15xxx medium-density devices



7.4 Important considerations

The bootloader of the medium-density ultralow power devices has some specific features that should be taken into consideration, as described below:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), the STM32L100xx value line and STM32L15xxx medium-density device bootloader firmware supports Data Memory (4 Kbytes from 0x08080000 to 0x08080FFF). Refer to the PM0062 Programming manual for more information.
- **Flash memory write** operations are performed through a program memory half page write operation. The bootloader firmware manages half page write operations at non-aligned addresses. Consequently, all write operations must only be Word-aligned (the address should be a multiple of 4). The number of data to be written must also be a multiple of 4 (non-aligned half page write addresses are accepted). Be aware of the duration needed for a write operation by referring to the product datasheet.
- **Data memory** can be read and written but cannot be erased using the Erase Command. When writing in a Data memory location, the bootloader firmware manages the erase operation of this location before any write. A write to Data memory must be Word-aligned (address to be written should be a multiple of 4) and the number of data must also be a multiple of 4. To erase a Data memory location, you can write zeros at this location.
- **Option byte**

  Address is 0x1FF80000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to PM0062 programming manual for more details about protection levels.
- **Read protect** command corresponds to the Level 1 protection.
- **Read unprotect** command corresponds to the Level 0 protection.
- **Mass erase** command is not supported by STM32L15xxx medium-density device bootloader firmware. To perform a mass erase operation, two options are available:
  – Erase all sectors one by one using the Erase command
  – Set protection level to Level 1. Then, set it to Level 0 (using the Read protect command and then the Read Unprotect command). This operation results in a mass erase of the internal Flash memory (refer to Programming Manual PM0062 for more details).

7.5 Bootloader version

The following table lists the STM32L100xx value line and STM32L15xxx medium-density bootloader versions.

Table 12. STM32L100xx value line and STM32L15xxx medium-density bootloader versions

Bootloader version number	Description	Known limitations
V2.0	Initial bootloader version.	When a Read Memory command or Write Memory command is issued with an unsupported memory address and a correct address checksum (ie. address 0x6000 0000), the command is aborted by the bootloader device, but the NACK (0x1F) is not sent to the host. As a result, the next 2 bytes (which are the number of bytes to be read/written and its checksum) are considered as a new command and its checksum.(1)

1. If the “number of data - 1” (N-1) to be read/written is not equal to a valid command code (0x00, 0x01, 0x02, 0x11, 0x21, 0x31, 0x43, 0x44, 0x63, 0x73, 0x82 or 0x92), then the limitation is not perceived from the host since the command is NACKed anyway (as an unsupported new command).

8 STM32L151xx and STM32L152xx medium-density plus ultralow power device bootloader

Throughout this section, STM32L15xxx medium-density plus is used to refer to the STM32L151xx and STM32L152xx medium-density plus ultralow power devices.

8.1 Bootloader configuration

The bootloader embedded in the STM32L15xxx medium-density plus devices supports three serial interfaces: USART1, USART2 and DFU (USB).

Table 13 shows the required hardware resources of STM32L15xxx medium-density plus devices used by the bootloader in System memory boot mode.

Table 13. STM32L15xxx medium-density plus configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock frequency is 16 MHz using the HSI. This is used only for USART1 and USART2 bootloaders and during USB detection for DFU bootloader (once the DFU bootloader is selected, the clock source is derived from the external crystal).
		HSE enabled	The external clock is mandatory only for the DFU bootloader and must be in the following range: [24, 16, 12, 8, 6, 4, 3, 2] MHz. The PLL is used to generate the USB 48 MHz clock and the 32 MHz clock for the system clock.
		-	The clock security system (CSS) interrupt is enabled for the DFU bootloader. Any failure (or removal) of the external clock generates a system reset.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog resets (in case the hardware IWDG option was previously enabled by the user).
	Power		Voltage range is set to Voltage Range 1.
	System memory	-	8 Kbytes starting from address 0x1FF0 0000. This area contains the bootloader firmware.
	RAM	-	4 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.

Table 13. STM32L15xxx medium-density plus configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is 8 bits, even parity and 1 stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for the USARTx bootloader.
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is 8 bits, even parity and 1 stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode
	USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB_DM pin	Input or alternate function, automatically controlled by the USB	PA11: USB send-receive data line
	USB_DP pin		PA12: USB send-receive data line
	Interrupts	Enabled	USB low priority interrupt vector is enabled and used for USB DFU communication.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

Note: For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI, MSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around the theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for the USARTx bootloader. This internal clock is also used the for DFU bootloader but only for the selection phase. An external clock in the range of [24, 16, 12, 8, 6, 4, 3, 2] MHz is required for the execution of the DFU bootloader after the selection phase.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

8.2 Bootloader hardware requirements

The hardware required to put the STM32L15xxx medium-density plus devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32L15xxx medium-density plus devices during System memory boot mode, the following conditions have to be verified.

- The RX pin of the peripherals that are not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB) is used to connect to the bootloader: the USART1_RX (PA10) and USART2_RX (PD6) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral.
  - An RS-232 serial interface (example, ST3232 RS-232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
  - A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used. As a result, the application can use these pins for other peripherals or GPIOs. This is also applicable for USART2.

8.3 Bootloader selection

The STM32L15xxx medium-density plus embedded bootloader supports three serial interfaces: USART1, USART2 and DFU (USB). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 8.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baudrate sequence and then enters a loop, waiting for any USART bootloader command.

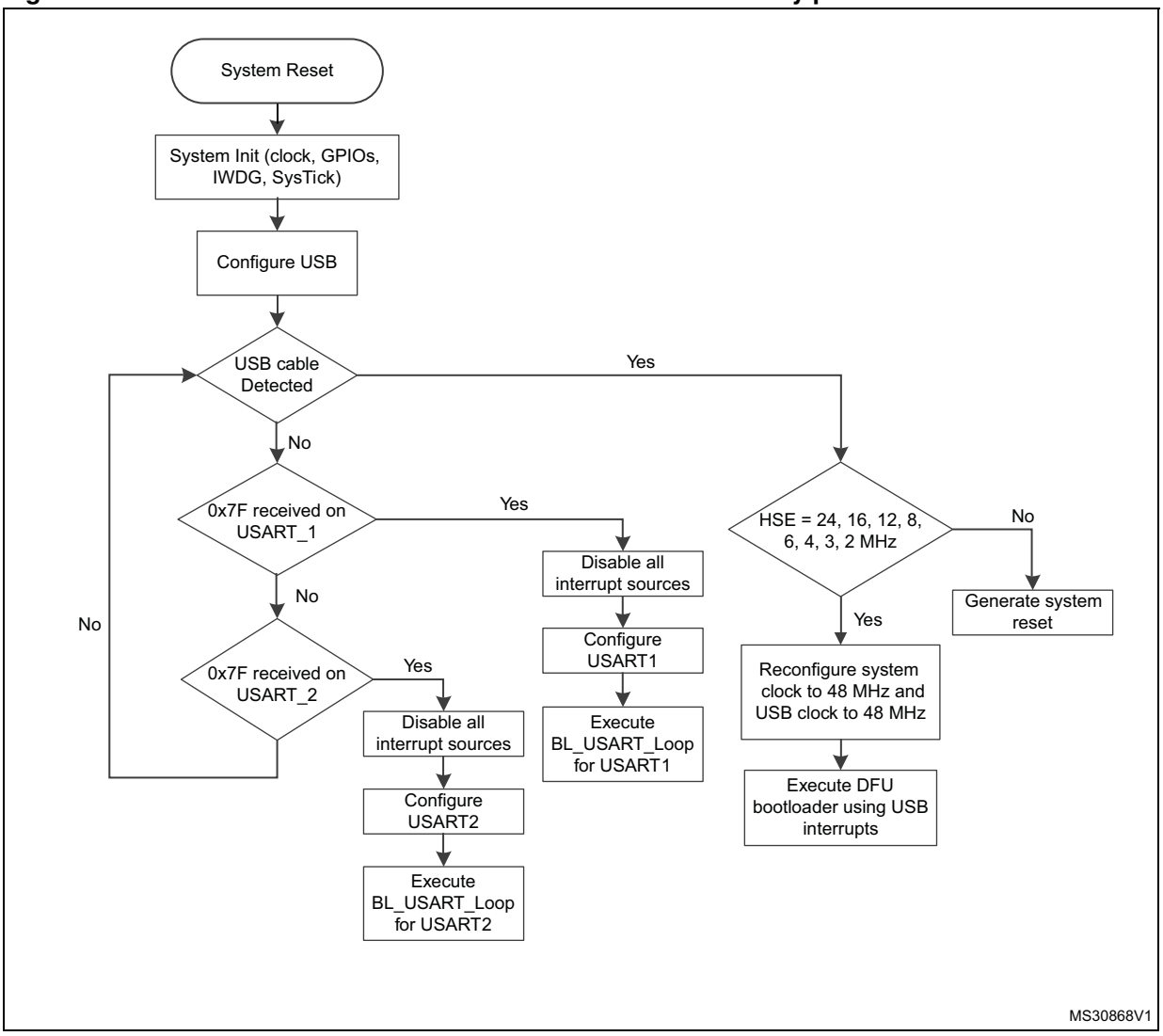
If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters DFU bootloader loop waiting for any DFU bootloader command.

To use the USART bootloader, it is mandatory that no USB Host be connected to the USB peripheral during the selection phase. Once the USART bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except for the commands which generate a system reset.

Once an interface is selected for the bootloader, the other interface is disabled.

Figure 5 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 5. Bootloader selection for STM32L15xxx medium-density plus devices



8.4 Important considerations

The bootloader of the STM32L15xxx medium-density plus devices has some specific features that should be taken into consideration, as described below.

- In addition to standard memories (internal Flash, internal SRAM, option bytes and system memory), the STM32L15xxx medium-density plus devices bootloader firmware supports Data Memory (8 Kbytes from 0x08080000 to 0x08081FFF). Refer to the PM0062 Programming manual for more information.
- Flash memory write operations are performed through a program memory half page write operation. The bootloader firmware manages half page write operations at non-aligned addresses. Consequently, all write operations must only be word-aligned (the address should be a multiple of 4). The number of data to be written must also be a multiple of 4 (non-aligned half page write addresses are accepted). Be aware of the duration needed for a write operation by referring to the product datasheet.
- Data memory can be read and written but cannot be erased using the Erase Command. When writing in a Data memory location, the bootloader firmware manages the erase operation of this location before any write. A write to Data memory must be word-aligned (address to be written should be a multiple of 4) and the number of data must also be a multiple of 4. To erase a Data memory location, you can write zeros at this location.
- Option byte

  Address is 0x1FF80000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the PM0062 programming manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands corresponds to the Level 0 protection.
- Mass erase commands are not supported by the STM32L15xxx medium-density plus devices bootloader firmware. To perform a mass erase operation, two options are available.
  – Erase all sectors one by one using the Erase command.
  – Set the protection level to Level 1. Then, set it to Level 0 (using the Read protect command and then the Read Unprotect command). This operation results in a mass erase of the internal Flash memory (refer to the Programming Manual PM0062 for more details).

If the application is loaded into the Flash memory at an address different to 0x08000000, the vector table has to be relocated to start from the address where the application is loaded.

8.5 Bootloader version

Table 14 lists the bootloader versions for the STM32L15xxx medium-density plus devices.

Table 14. STM32L15xxx medium-density plus bootloader versions

Bootloader version number	Description	Known limitations
V4.0	Initial bootloader version	For the USART interface, two consecutive NACKs instead of 1 NACK are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

9 STM32L151xx, STM32L152xx and STM32L162xx high-density ultralow power device bootloader

Throughout this section, STM32L1xxxx high-density is used to refer to STM32L151xx, STM32L152xx and STM32L162xx high-density ultralow power devices.

9.1 Dual bank boot feature

The STM32L1xxxx high-density devices have two Flash memory banks: Bank 1 and Bank 2. They feature an additional boot mechanism which allows booting from Bank 2 or Bank 1 depending on BFB2 bit status (bit 7 in the user option bytes located at 0x1FF8 0004).

- When the BFB2 bit is reset and the boot pins are configured to boot from Flash memory (BOOT0 = 0 and BOOT1 = x), after reset the device boots from the System memory and executes the embedded bootloader code which implements the dual bank Boot mode:
  a) The code first checks Bank 2. If it contains a valid code (see note below), it jumps to the application code located in Bank 2 and leaves the bootloader.
  b) If the Bank 2 code is not valid, it checks Bank 1 code. If it is valid (see note below), it jumps to the application located in Bank 1.
  c) If both Bank 2 and Bank 1 do not contain valid code (see note below), the normal bootloader operations are executed as described in the following sections and no jump to Flash banks is performed. Refer to *Figure 6: Bootloader selection for STM32L1xxxx high-density devices* for more details.

3. When BFB2 bit is set (default state), the dual bank boot mechanism is not performed.

Note: The code is considered as valid when the first data (at the bank start address, which should be the stack pointer) points to a valid address into the internal SRAM memory (stack top address). If the first address points to any other location (out of the internal SRAM) the code is considered not valid.

A dual bank Boot mode example (FLASH\Dual_Boot) is provided within the STM32L1xxxx Standard Peripheral Library available from http://www.st.Com.

For the STM32L1xxxx high-density devices, the Flash memory, system memory or SRAM is selected as the boot space, as shown in *Table 15* below.

Table 15. Boot pin and BFB2 bit configuration

BFB2 bit	Boot mode selection pins		Protection level2	Bank2 Valid	Bank1 Valid	Boot mode	Aliasing
	BOOT1	BOOT0					
1	X	0	X	X	X	User Flash memory	User Flash memory Bank1 is selected as the boot space
	0	1	No	X	X	System memory	System memory is selected as the boot space
	0	1	Yes	X	X	User Flash memory	User Flash memory Bank1 is selected as the boot space
	1	1	No	X	X	Embedded SRAM	Embedded SRAM is selected as the boot space
	1	1	Yes	X	X	User Flash memory	User Flash memory Bank1 is selected as the boot space
0	X	0	X	Yes	X	System memory	User Flash memory Bank2 is selected as the boot space
				No	Yes	System memory	User Flash memory Bank1 is selected as the boot space
			No	No	No	System memory	System memory is selected as the boot space
			Yes	No	No	System memory	CPU blocked (halted)
	0	1	No	X	X	System memory	System memory is selected as the boot space
	1	1	No	X	X	Embedded SRAM	Embedded SRAM is selected as the boot space
	X	1	Yes	Yes	X	System memory	User Flash memory Bank2 is selected as the boot space
				No	Yes	System memory	User Flash memory Bank1 is selected as the boot space
				No	No	System memory	CPU blocked (halted)

Table 15 shows that the STM32L1xxxx high-density devices enter System memory boot mode in three cases:

- If the BOOT pins are configured as follows:
  BOOT0 = 1 and BOOT1 = 0
- If the BFB2 bit is reset and protection Level2 is enabled
- If the BFB2 bit is reset and boot pins are configured as follows:
  BOOT0 = 0 and BOOT1 = x

Note: When the conditions a, b, and c described below are fulfilled, it is equivalent to configuring boot pins for system memory boot (BOOT0 = 1 and BOOT1 = 0). In this case normal bootloader operations are executed.

a) BFB2 bit is reset

b) Both banks don't contain valid code

c) Boot pins configured as follows: BOOT0 = 0 and BOOT1 = x

When the BFB2 bit is cleared, and Bank 2 and/or Bank 1 contain valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code and never continues normal operations).

Consequently, if you have cleared the BFB2 bit (to boot from Bank 2) then, to be able to execute the bootloader code, you have to either:

- program the content of address 0x0808 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0, or

- set the BFB2 bit to 1, BOOT0 = 1 and BOOT1 = 0.

9.2 Bootloader configuration

The bootloader embedded in STM32L1xxxx high-density devices supports three serial interfaces: USART1, USART2 and DFU (USB)

The following table shows the required hardware resources of STM32L1xxxx high-density devices used by the bootloader in System memory boot mode.

Table 16. STM32L1xxxx high-density configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock frequency is 16 MHz using the HSI. This is used only for USART1 and USART2 bootloaders and during USB detection for DFU bootloader (once the DFU bootloader is selected, the clock source will be derived from the external crystal).
		HSE enabled	The external clock is mandatory only for DFU bootloader and it must be in the following range: [24, 16, 12, 8, 6, 4, 3, 2] MHz. The PLL is used to generate the USB 48 MHz clock and the 32 MHz clock for the system clock.
		-	The clock security system (CSS) interrupt is enabled for the DFU bootloader. Any failure (or removal) of the external clock generates system reset.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power		Voltage range is set to Voltage Range 1.
	System memory	-	8 Kbytes starting from address 0x1FF0 0000. This area contains the bootloader firmware.
	RAM	-	4 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.

Table 16. STM32L1xxxx high-density configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode
	USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB_DM pin	Input or alternate function, automatically controlled by the USB	PA11: USB Send-Receive data line
	USB_DP pin		PA12: USB Send-Receive data line
	Interrupts	Enabled	USB Low Priority interrupt vector is enabled and used for USB DFU communication.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

Note: For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI, MSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for USARTx bootloader. This internal clock is used also for DFU bootloader but only for the selection phase. An external clock in the range of [24, 16, 12, 8, 6, 4, 3, 2] MHz is required for DFU bootloader execution after the selection phase.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

9.3 Bootloader hardware requirements

The hardware required to put the STM32L1xxxx high-density devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

Note: As explained in Section 9.1: Dual bank boot feature, the System memory boot mode can also be executed by software when the BFB2 bit is reset, both banks start addresses are erased, and boot pins are configured to boot from Flash memory.

To connect to the STM32L1xxxx high-density devices during System memory boot mode, the following conditions have to be verified:

- The RX pin of the peripherals that are not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB) is used to connect to the bootloader: the USART1_RX (PA10) and USART2_RX (PD6) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- When the BFB2 bit is cleared, and Bank 2 and/or Bank 1 contain a valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code and never continues normal operations). Consequently, if you have cleared the BFB2 bit (to boot from Bank 2), then to be able to execute the bootloader code, you have to either:
  - program the content of address 0x0808 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0, or
  - set the BFB2 bit to 1, BOOT0 = 1 and BOOT1 = 0.
- Connection to the peripheral to be performed through:
  - an RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used
  - A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used. as a result, the application can use these pins for other peripherals or GPIOs. This is also applicable for USART2.

9.4 Bootloader selection

STM32L1xxxx high-density embedded bootloader supports three serial interfaces: USART1, USART2 and DFU (USB). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 9.3: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baudrate sequence and then enters a loop, waiting for any USART bootloader command.

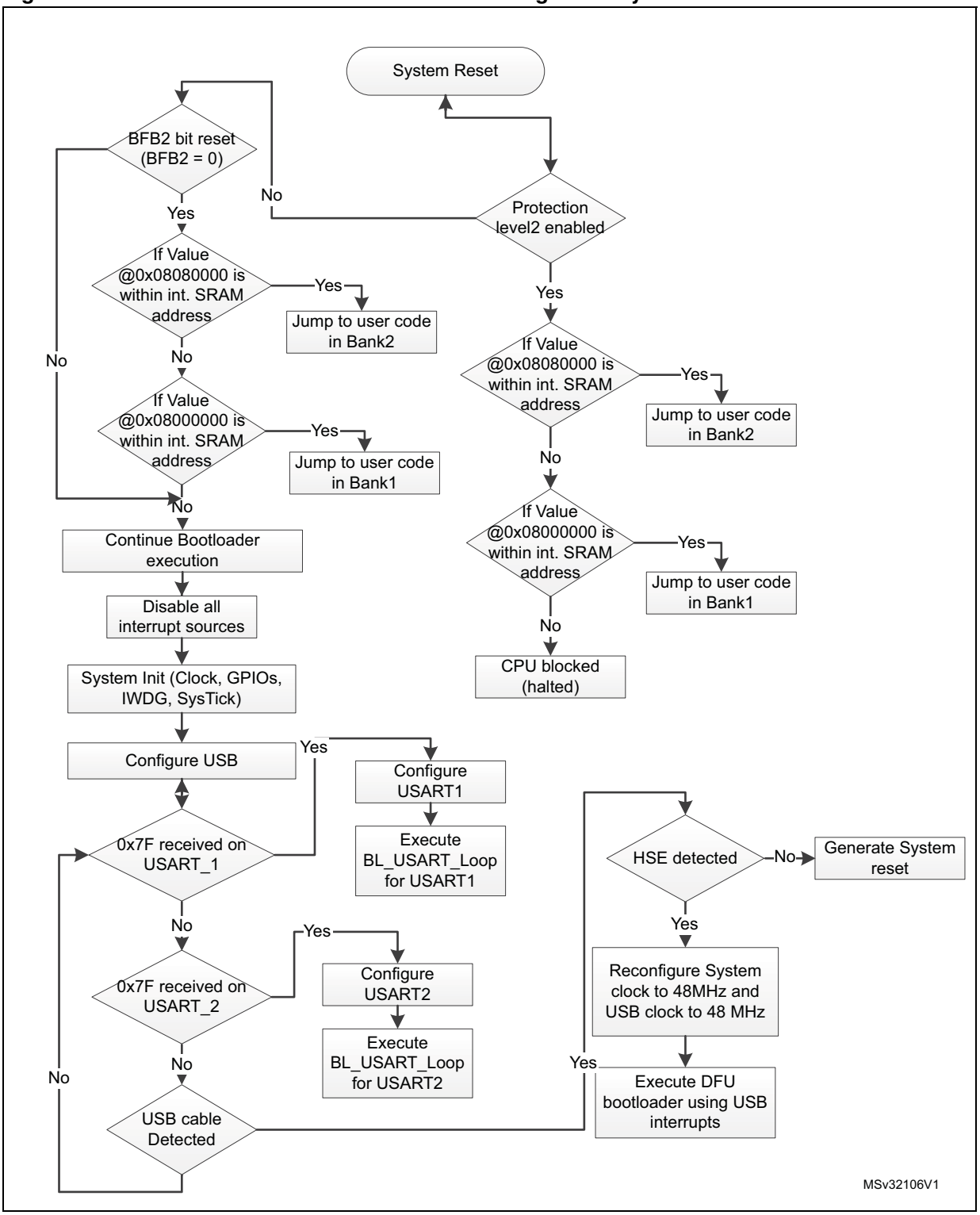
If a USB cable is plugged into the microcontroller into USB interface at any time during the bootloader firmware selection sequence, the bootloader enters DFU bootloader loop waiting for any DFU bootloader command.

To use the USART bootloader, it is mandatory that no USB Host is connected to the USB peripheral during the selection phase. Once the USART bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except for the commands which generate a system reset.

Once an interface is selected for the bootloader, the other interface is disabled.

Figure 6: Bootloader selection for STM32L1xxxx high-density devices shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 6. Bootloader selection for STM32L1xxxx high-density devices



9.5 Important considerations

The bootloader of the STM32L1xxxx high-density devices has some specific features that should be taken into consideration, as described below:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), the STM32L1xxxx high-density devices bootloader firmware supports Data Memory (12 Kbytes from 0x08080000 to 0x08082FFF). Refer to the PM0062 Programming manual for more information.
- **Flash memory write** operations are performed through a program memory half page write operation. The bootloader firmware manages half page write operations at non-aligned addresses. Consequently, all write operations must only be Word-aligned (the address should be a multiple of 4). The number of data to be written must also be a multiple of 4 (non-aligned half page write addresses are accepted). Be aware of the duration needed for a write operation by referring to the product datasheet.
- **Data memory** can be read and written but cannot be erased using the Erase Command. When writing in a Data memory location, the bootloader firmware manages the erase operation of this location before any write. A write to Data memory must be Word-aligned (address to be written should be a multiple of 4) and the number of data must also be a multiple of 4. To erase a Data memory location, you can write zeros at this location.
- **Option byte**

  Address is 0x1FF80000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to PM0062 programming manual for more details about protection levels.
- **Read protect** command corresponds to the Level 1 protection.
- **Read unprotect** command corresponds to the Level 0 protection.
- **Mass erase** command is not supported by STM32L1xxxx high-density devices bootloader firmware. To perform a mass erase operation, two options are available:
  – Erase all sectors one by one using the Erase command
  – Set protection level to Level 1. Then, set it to Level 0 (using the Read protect command and then the Read Unprotect command). This operation results in a mass erase of the internal Flash memory (refer to Programming Manual PM0062 for more details).

9.6 Bootloader version

Table 17 lists the bootloader versions for the STM32L1xxxx high-density devices.

Table 17. STM32L1xxxx high-density bootloader versions

Bootloader version number	Description	Known limitations
V4.1	Initial bootloader version	– In the bootloader code the PA13 (JTMS/SWDIO) I/O output speed is configured to 400 KHz, as consequence some debugger can not connect to the device in Serial Wire mode when the bootloader is running. – When the DFU bootloader is selected, the RTC is reset and thus all RTC information (calendar, alarm, ...) will be lost including backup registers. Note: When the USART bootloader is selected there is no change on the RTC configuration (including backup registers).
V4.2	Fix V4.1 limitations (available on Rev.Z devices only.)	– Stack overflow by 8 bytes when jumping to Bank1/Bank2 if BFB2=0 or when Read Protection level is set to 2. Workaround: the user code should force in the startup file the top of stack address before to jump to the main program. This can be done in the “Reset_Handler” routine. – When the Stack of the user code is placed outside the SRAM (ie. @ 0x2000C000) the bootloader cannot jump to that user code which is considered invalid. This might happen when using compilers which place the stack at a non-physical address at the top of the SRAM (ie. @ 0x2000C000). Workaround: place manually the stack at a physical address.
V4.5	Fix V4.2 limitations. DFU interface robustness enhancements (available on Rev.Y devices only).	– For the USART interface, two consecutive NACKs (instead of 1 NACK) are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

10 STM32F205/215xx and STM32F207/217xx bootloader

Throughout this section, STM32F2xxxx is used to refer to STM32F205xx, STM32F207xx, STM32F215xx and STM32F217xx devices.

Two bootloader versions are available on STM32F2xxxx devices:

- V2.0 supporting USART1 and USART3
  This version is embedded in STM32F2xxxx devices revision B.
- V3.2 supporting USART1, USART3, CAN2 and DFU (USB FS Device)
  This version is embedded in STM32F2xxxx devices revision X and Y.

10.1 Bootloader V2.x

10.1.1 Bootloader configuration

The bootloader V2.x embedded in STM32F2xxxx devices support two serial interfaces: USART1 and USART3 peripherals.

The following table shows the required hardware resources of STM32 devices used by the bootloader V2.x in System memory boot mode.

Table 18. STM32F2xxxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	Clock source	HSI enabled	The system clock is equal to 24 MHz.
	RAM	-	8 Kbytes starting from address 0x2000 0000.
	System memory	-	30688 bytes starting from address 0x1FFF 0000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to Voltage Range: [1.8V, 2.1V] (voltage range can be configured in run time using bootloader commands. Note that in this range internal Flash write operations are allowed only in byte format (Half-Word, Word and Double-Word operations are not allowed).
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 receives.
	USART1_TX pin	Output	PA9 pin: USART1 transmits.
	USART3_RX (PB11), USART3_RX (PC11) pins must be kept at a high or low level during the detection phase.		

Table 18. STM32F2xxxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART3 bootloader (on PC10/PC11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PC11 pin: USART3 receives.
	USART3_TX pin	Output	PC10pin: USART3 transmits.
	USART1_RX (PA10) and USART3_RX (PB11) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PB10/PB11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit
	USART3_RX pin	Input	PB11 pin: USART3 receives.
	USART3_TX pin	Output	PB10 pin: USART3 transmits.
	USART1_RX (PA10) and USART3_RX (PC11) pins must be kept at a high or low level during the detection phase.		
USART1 and USART3 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

The system clock is derived from the embedded internal high-speed RC. No external quartz is required for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in the above table) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. So, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

10.1.2 Bootloader hardware requirements

The hardware required to put the STM32F2xxxx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F2xxxx during System memory boot mode, the following conditions have to be verified:

- The RX pins of the peripherals unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART3_RX (PC11 and PB11) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PB10/PB11) is used to connect to the bootloader: the USART1_RX (PA10) and the other USART3_RX pin (PC11) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PC10/PC11) is used to connect to the bootloader: the USART1_RX (PA10) and the other USART3_RX pin (PB11) have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral to be performed through:
  - An RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART3_RX (PB11 or PC11) and USART3_TX (PB10 or PC10) pins or when USART3 is used.

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART3.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM322xG-EVAL board. For more details, refer to the STM322xG-EVAL board user manual, available from the STMicroelectronics website: *http://www.st.com*.

10.1.3 Bootloader selection

The STM32F2xxxx embedded bootloader V2.x supports two peripheral interfaces: USART1 and USART3 (on PB10/PB11 and PC10/PC11). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash memory.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 10.1.2: Bootloader hardware requirements for more information.

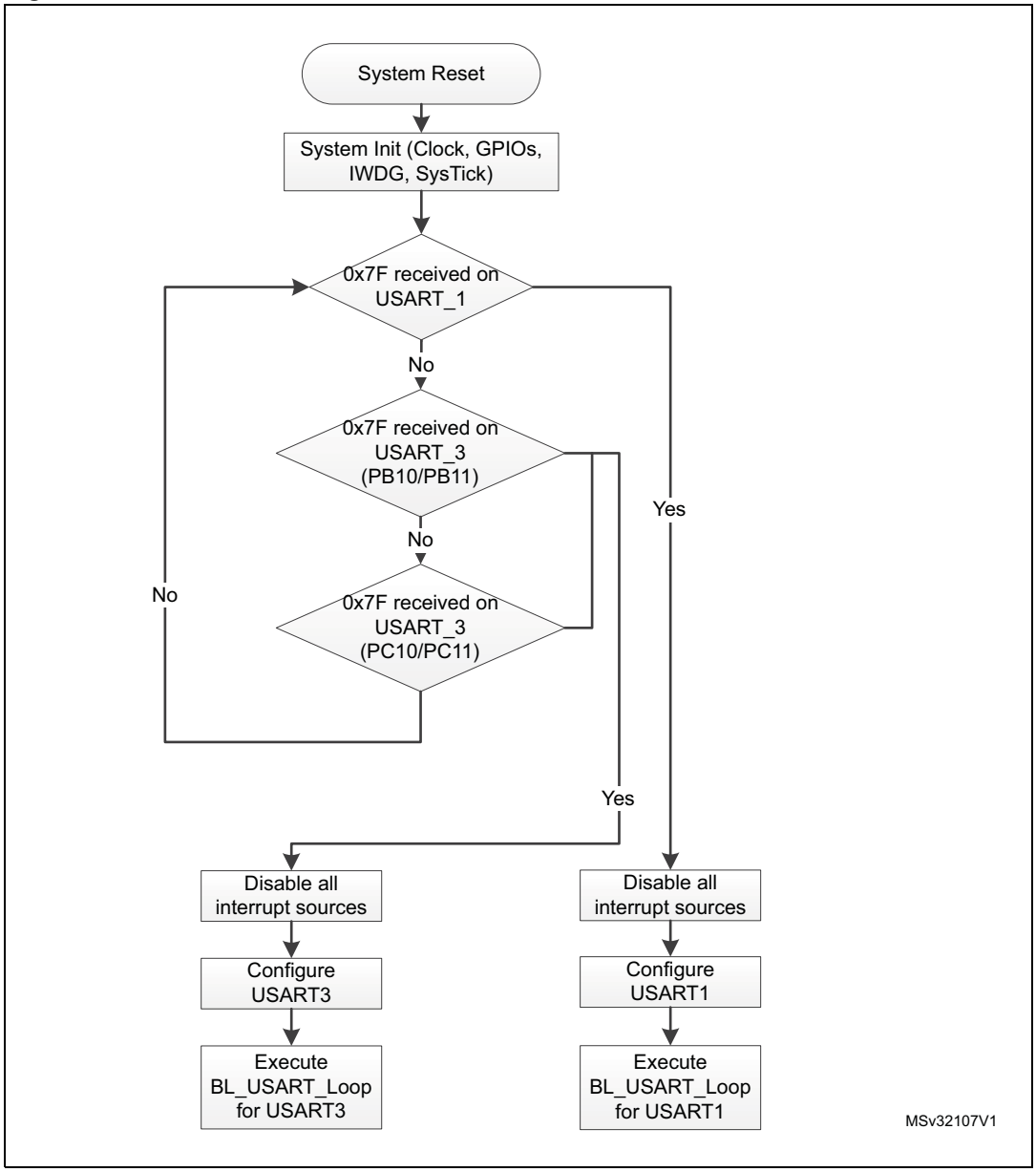
To use the USART bootloader on USART1 or USART3, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the

bootloader firmware executes the autobaudrate detection sequence and enters a loop, waiting for any USART bootloader command.

Once an interface is selected for the bootloader, the other interfaces are disabled.

Figure 7 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 7. Bootloader V2.x selection for STM32F2xxxx devices



10.1.4 Important considerations

The STM32F2xxxx bootloader has some specific features that should be taken into consideration:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), STM32F2xxxx bootloader firmware supports OTP memory (512 bytes from 0x1FFF 7800 to 0x1FFF 7A00). Refer to PM0059 Flash programming manual for more information.
- **OTP memory** can be read and written but cannot be erased using the Erase command. When writing in an OTP memory location, make sure that the relative protection bit (in the last 16 bytes of the OTP memory) is not reset.
- **Option bytes**

  Address is 0x1FFFC000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to PM0059 programming manual for more details about protection levels.
- **Read protect** command corresponds to Level 1 protection.
- **Read unprotect** command corresponds to Level 0 protection.
- **Mass erase** command on STM32F2xxxx takes longer than on other STM32F devices due to their memory density. Make sure that the timeout used by your host interface to wait for an acknowledge event after sending a Mass erase command is sufficient.
- **Voltage Range configuration**

  The Voltage Range can be updated on the fly by the bootloader software. The Voltage Range is set to its default value at each bootloader software startup (after system reset or jump to the bootloader code). The bootloader software allows modifying this parameter through a virtual memory location. This memory location is not physical but can be read and written using usual bootloader read/write operations according to the protocol in use (USART,CAN or DFU). This memory location contains 4 bytes which are described in *Table 19*. It can be accessed by 1, 2, 3 or 4 bytes. However, reserved bytes should remain at their default values (0xFF), otherwise the request will be NACKed.

Table 19. STM32F2xxxx Voltage Range configuration using bootloader V2.x

Address	Size	Description
0xFFFF0000	1 byte	This byte controls the current value of Voltage Range: 0x00: Voltage Range [1.8V, 2.1V] 0x01: Voltage Range [2.1V, 2.4V] 0x02: Voltage Range [2.4V, 2.7V] 0x03: Voltage Range [2.7V, 3.6V] 0x04: Voltage Range [2.7V, 3.6V] and Double Word write/erase operation is used. In this case it is mandatory to supply 9 V through VPP pin (refer to PM0059 for more details about Double-Word write operation). Other: all other values are not supported and will be NACKed.
0xFFFF0001	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0002	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0003	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.

10.1.5 Bootloader V2.x versions

Table 20 lists the V2.x versions of STM32F2xxxx bootloader.

Table 20. STM32F2xxxx bootloader V2.x versions

Bootloader version number	Description	Known limitations
V2.0	Initial V2.x bootloader version	When a Read Memory command or Write Memory command is issued with an unsupported memory address and a correct address checksum (ie. address 0x6000 0000), the command is aborted by the bootloader device, but the NACK (0x1F) is not sent to the host. As a result, the next 2 bytes (which are the number of bytes to be read/written and its checksum) are considered as a new command and its checksum.(1)

1. If the “number of data - 1” (N-1) to be read/written is not equal to a valid command code (0x00, 0x01, 0x02, 0x11, 0x21, 0x31, 0x43, 0x44, 0x63, 0x73, 0x82 or 0x92), then the limitation is not perceived from the host since the command is NACKed anyway (as an unsupported new command).

10.2 Bootloader V3.x

10.2.1 Bootloader configuration

The bootloader V3.x embedded in STM32F2xxxx devices support four serial peripherals: USART1, USART3, CAN2, and DFU (USB FS Device).

Table 21 shows the required hardware resources of STM32F2xxxx devices used by the bootloader V3.x in System memory boot mode.

Table 21. STM32F2xxxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock is equal to 24 MHz using the PLL. The HSI clock source is used at startup (interface detection phase) and when USARTx interfaces are selected (once CAN or DFU bootloader is selected, the clock source will be derived from external crystal).
		HSE enabled	The system clock is equal to 60 MHz. The HSE clock source is used only when the CAN or the DFU (USB FS Device) interfaces are selected. The external clock must provide a frequency multiple of 1 MHz and ranging from 4 MHz to 26 MHz.
		-	The Clock Security System (CSS) interrupt is enabled for the CAN and DFU bootloaders. Any failure (or removal) of the external clock generates system reset.
	RAM	-	8 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.
	System memory	-	30688 bytes starting from address 0x1FF0 0000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to [1.8V, 2.1V]. The voltage range can be configured in run time using bootloader commands. Note that in this range internal Flash write operations are allowed only in byte format (half-word, word and double-word operations are not allowed).

Table 21. STM32F2xxxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART3_RX (PB11), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PB10/PB11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PB11 pin: USART3 in reception mode
	USART3_TX pin	Output	PB10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PC10/PC11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PC11 pin: USART3 in reception mode
	USART3_TX pin	Output	PC10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART3 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloaders.
CAN2 bootloader	CAN2	Enabled	Once initialized, the CAN2 configuration is: Baudrate 125 kbps, 11-bit identifier. **Note:** CAN1 is clocked during CAN2 bootloader execution because STM32F2xxxx CAN1 manages the communication between CAN2 and SRAM.
	CAN2_RX pin	Input	PB05 pin: CAN2 in reception mode
	CAN2_TX pin	Output	PB13pin: CAN2 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB_OTG_FS	Enabled	USB OTG FS configured in Forced Device mode. USB_OTG_FS interrupt vector is enabled and used for USB DFU communications.
	USB_OTG_FS_DM pin	Input	PA11 pin: USB OTG FS DM line
	USB_OTG_FS_DP pin	Output	PA12pin: USB OTG FS DP line
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11) and CAN2_RX (PB05) pins must be kept at a high or low level during the detection phase.		
CAN2 and DFU bootloaders	TIM11	Enabled	This timer is used to determine the value of the external clock frequency. Once the external clock frequency is determined, the RCC system is configured to operate at 60 MHz system clock (using PLL).

Note: For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for USARTx bootloaders. No external quartz is required in this case for the bootloader code. This internal clock is also used for CAN and DFU (USB FS Device) but only for the selection phase. An external clock multiple of 1 MHz (between 4 and 26 MHz) is required for CAN and DFU bootloader execution after the selection phase.

The CAN and DFU bootloaders implement an external clock detection mechanism allowing to determine the value of the external clock using the internal high-speed RC and TIM11 timer. The accuracy of this mechanism allows to detect only frequencies multiple of 1 MHz and ranging from 4 to 26 MHz. Any other value is not supported and will result in unexpected behavior of the bootloader.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in the above table) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. So, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

10.2.2 Bootloader hardware requirements

The hardware required to put the STM32F2xxxx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F2xxxx during System memory boot mode, the following conditions have to be verified:

- The RX pins of the peripheral unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART3_RX (PC11 and PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PB10/PB11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PC10/PC11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX pin (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.

  - If CAN2 is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB FS Device) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11) and CAN2_RX (PB05) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral to be performed through:
  - An RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used
  - A CAN interface (CAN transceiver) has to be directly connected to the CAN2_RX (PB5) and CAN2_TX (PB13) pins.
  - A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART3.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM322xG-EVAL board. For more details about this, refer to the STM322xG-EVAL board user manual, available from the STMicroelectronics website: *http://www.st.com*.

10.2.3 Bootloader selection

The STM32F2xxxx embedded bootloader V3.x supports three peripheral interfaces: USART1, USART3 (on PB10/PB11 and PC10/PC11), CAN2 and DFU (USB FS Device). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 10.2.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART3, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baud rate sequence and then enters a loop, waiting for any USART bootloader command.

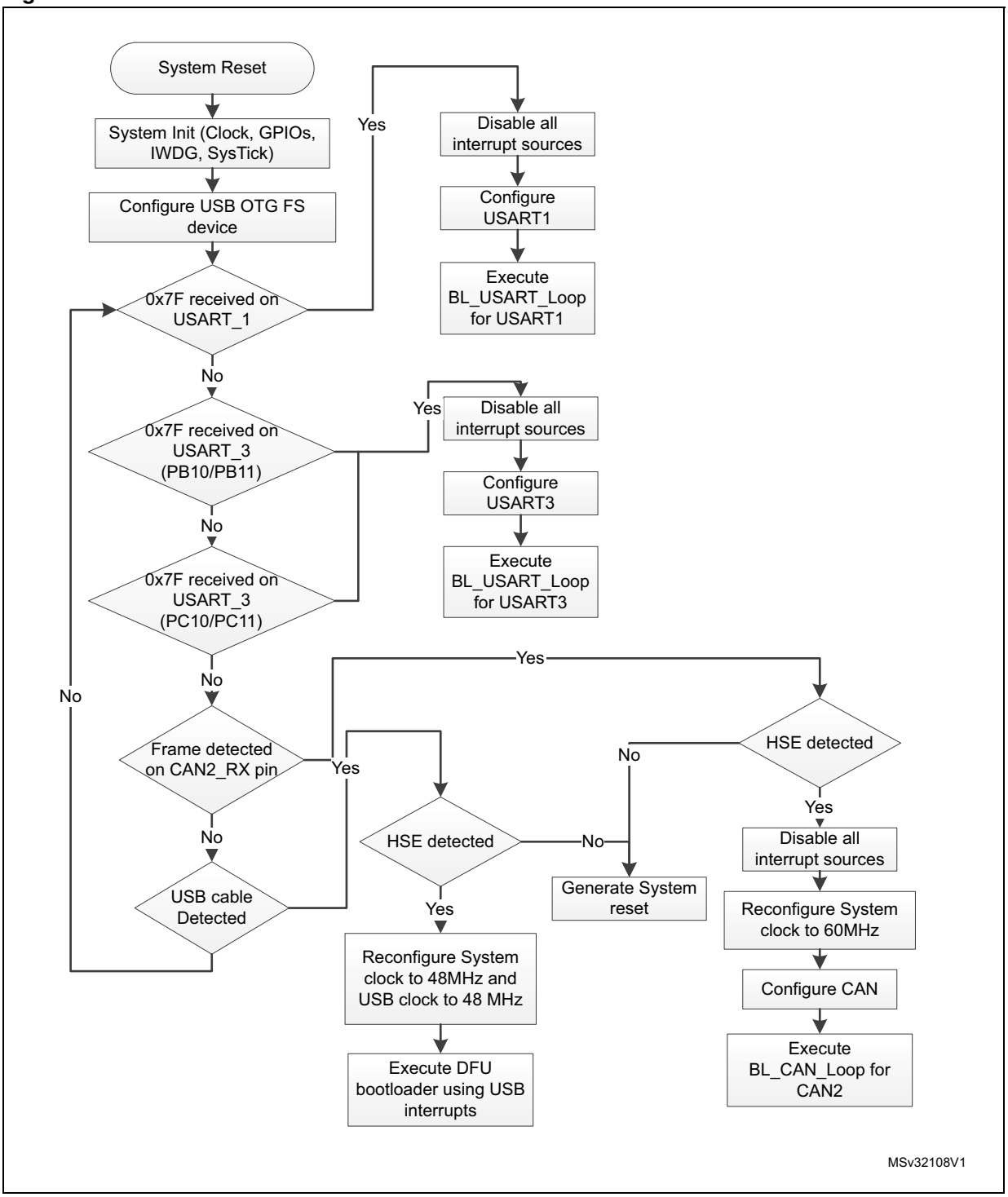
To use the CAN2 interface, connect the CAN cable to CAN2. Once the bootloader detects a frame on the CAN2_RX pin (PB5), the bootloader firmware enters a CAN loop and starts to determine the external clock frequency value. The supported HSE frequencies are multiple of 1 MHz ranging from 4 to 26 MHz. Any other values leads to an unexpected behavior, CAN bootloader firmware enters an infinite loop and waits until it receives a message. If the external clock is not present, a system reset is generated.

If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters the DFU bootloader loop waiting for any DFU bootloader command.

To use the USART or the CAN bootloader, it is mandatory that no USB Host is connected to the USB peripheral during the selection phase. Once the USART or CAN bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except commands which generate a system reset.

Once one interface is selected for the bootloader, all other interfaces are disabled. The figure below shows the bootloader selection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 8. Bootloader V3.x selection for STM32F2xxxx devices



10.2.4 Important considerations

STM32F2xxxx bootloader has some specific features that should be taken into consideration:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), STM32F2xxxx device bootloader firmware supports OTP memory (512 bytes from 0x1FFF 7800 to 0x1FFF 7A00, refer to PM0059 programming manual for more information).
- **OTP memory** can be read and written but cannot be erased using Erase command. When writing in an OTP memory location, make sure that the relative protection bit (in the last 16 bytes of the OTP memory) is not reset.
- **Option bytes**

  Address is 0x1FFFC000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to PM0059 programming manual for more details about protection levels.
- **Read protect** command corresponds to Level 1 protection.
- **Read unprotect** command corresponds to Level 0 protection.
- **Mass erase** command on STM32F2xxxx takes longer than on other STM32 devices due to their memory density. Make sure that the timeout used by your host interface to wait for an acknowledge event after sending a Mass erase command is sufficient.
- **Voltage Range configuration**

  The Voltage Range can be updated on the fly by the bootloader software. The Voltage Range is set to its default value at each bootloader software startup (after system reset or jump to the bootloader code). The bootloader software allows modifying this parameter through a virtual memory location. This memory location is not physical but can be read and written using usual bootloader read/write operations according to the protocol in use (USART,CAN or DFU). This memory location contains 4 bytes which are described in *Table 22*. It can be accessed by 1, 2, 3 or 4 bytes. However, reserved bytes should remain at their default values (0xFF), otherwise the request will be NACKed.

Table 22. STM32F2xxxx Voltage Range configuration using bootloader V3.x

Address	Size	Description
0xFFFF0000	1 byte	This byte controls the current value of Voltage Range: 0x00: Voltage Range [1.8V, 2.1V] 0x01: Voltage Range [2.1V, 2.4V] 0x02: Voltage Range [2.4V, 2.7V] 0x03: Voltage Range [2.7V, 3.6V] 0x04: Voltage Range [2.7V, 3.6V] and Double Word write/erase operation is used. In this case it is mandatory to supply 9 V through VPP pin (refer to PM0059 for more details about Double-Word write procedure). Other: All other values are not supported and will be NACKed.
0xFFFF0001	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0002	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0003	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.

10.2.5 Bootloader version V3.x

The following table lists the V3.x versions of STM32F2xxxx bootloader.

Table 23. STM32F2xxxx bootloader V3.x versions

Bootloader version number	Description	Known limitations
V3.2	Initial V3.x bootloader version.	– When a Read Memory command or Write Memory command is issued with an unsupported memory address and a correct address checksum (ie. address 0x6000 0000), the command is aborted by the bootloader device, but the NACK (0x1F) is not sent to the host. As a result, the next 2 bytes (which are the number of bytes to be read/written and its checksum) are considered as a new command and its checksum(1). – Option bytes, OTP and Device Feature descriptors (in DFU interface) are set to “g” instead of “e” (not erasable memory areas).
V3.3	Fix V3.2 limitations. DFU interface robustness enhancement.	– For the USART interface, two consecutive NACKs (instead of 1 NACK) are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

1. If the “number of data - 1” (N-1) to be read/written is not equal to a valid command code (0x00, 0x01, 0x02, 0x11, 0x21, 0x31, 0x43, 0x44, 0x63, 0x73, 0x82 or 0x92), then the limitation is not perceived from the host since the command is NACKed anyway (as an unsupported new command).

11 STM32F405/415xx and STM32F407/417xx bootloader

Throughout this section, STM32F40xxx/41xxx is used to refer to STM32F405xx, STM32F407xx, STM32F415xx and STM32F417xx devices.

11.1 Bootloader configuration

The bootloader embedded in STM32F40xxx/41xxx devices support four serial peripherals: USART1, USART3, CAN2, and DFU (USB FS Device).

Table 24 shows the required hardware resources of STM32F40xxx/41xxx devices used by the bootloader in System memory boot mode.

Table 24. STM32F40xxx/41xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock is equal to 24 MHz using the PLL. The HSI clock source is used at startup (interface detection phase) and when USARTx interfaces are selected (once CAN or DFU bootloader is selected, the clock source will be derived from external crystal).
		HSE enabled	The system clock is equal to 60 MHz. The HSE clock source is used only when the CAN or the DFU (USB FS Device) interfaces are selected. The external clock must provide a frequency multiple of 1 MHz and ranging from 4 MHz to 26 MHz.
		-	The Clock Security System (CSS) interrupt is enabled for the CAN and DFU bootloaders. Any failure (or removal) of the external clock generates system reset.
	RAM	-	8 Kbytes starting from address 0x2000 0000 are used by the bootloader firmware.
	System memory	-	30688 bytes starting from address 0x1FFF 0000 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to [1.8V, 2.1V]. The voltage range can be configured in run time using bootloader commands. Note that in this range internal Flash write operations are allowed only in byte format (Half-Word, Word and Double-Word operations are not allowed).

Table 24. STM32F40xxx/41xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART3_RX (PB11), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PB10/PB11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PB11 pin: USART3 in reception mode
	USART3_TX pin	Output	PB10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PC10/PC11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PC11 pin: USART3 in reception mode
	USART3_TX pin	Output	PC10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART3 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloaders.
CAN2 bootloader	CAN2	Enabled	Once initialized, the CAN2 configuration is: Baudrate 125 kbps, 11-bit identifier. **Note:** CAN1 is clocked during CAN2 bootloader execution because STM32F40xxx/41xxx CAN1 manages the communication between CAN2 and SRAM.
	CAN2_RX pin	Input	PB05 pin: CAN2 in reception mode
	CAN2_TX pin	Output	PB13pin: CAN2 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		

Table 24. STM32F40xxx/41xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
DFU bootloader	USB_OTG_FS	Enabled	USB OTG FS configured in Forced Device mode. USB_OTG_FS interrupt vector is enabled and used for USB DFU communications.
	USB_OTG_FS_DM pin	Input	PA11 pin: USB OTG FS DM line
	USB_OTG_FS_DP pin	Output	PA12pin: USB OTG FS DP line
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11) and CAN2_RX (PB05) pins must be kept at a high or low level during the detection phase.		
CAN2 and DFU bootloaders	TIM11	Enabled	This timer is used to determine the value of the external clock frequency. Once the external clock frequency is determined, the RCC system is configured to operate at 60 MHz system clock (using PLL).

Note: For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for USARTx bootloaders. No external quartz is required in this case for the bootloader code. This internal clock is also used for CAN and DFU (USB FS Device) but only for the selection phase. An external clock multiple of 1 MHz (between 4 and 26 MHz) is required for CAN and DFU bootloader execution after the selection phase.

The CAN and DFU bootloaders implement an external clock detection mechanism allowing to determine the value of the external clock using the internal high-speed RC and TIM11 timer. The accuracy of this mechanism allows to detect only frequencies multiple of 1 MHz and ranging from 4 to 26 MHz. Any other value is not supported and will result in unexpected behavior of the bootloader.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in the above table) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. So, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

11.2 Bootloader hardware requirements

The hardware required to put the STM32F40xxx/41xxx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F40xxx/41xxx during System memory boot mode, the following conditions have to be verified:

- The RX pins of the peripheral unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader: the USART3_RX (PC11 and PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PB10/PB11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PC10/PC11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX pin (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If CAN2 is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB FS Device) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11) and CAN2_RX (PB05) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral to be performed through:
  - An RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used
  - A CAN interface (CAN transceiver) has to be directly connected to the CAN2_RX (PB5) and CAN2_TX (PB13) pins.
  - A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART3.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM324xG-EVAL board. For more details about this, refer to the STM324xG-EVAL board user manual, available from the STMicroelectronics website: *http://www.st.com*.

11.3 Bootloader selection

The STM32F40xxx/41xxx embedded bootloader supports three peripheral interfaces: USART1, USART3 (on PB10/PB11 and PC10/PC11), CAN2 and DFU (USB FS Device). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 11.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART3, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baud rate sequence and then enters a loop, waiting for any USART bootloader command.

To use the CAN2 interface, connect the CAN cable to CAN2. Once the bootloader detects a frame on the CAN2_RX pin (PB5), the bootloader firmware enters a CAN loop and starts to determine the external clock frequency value.

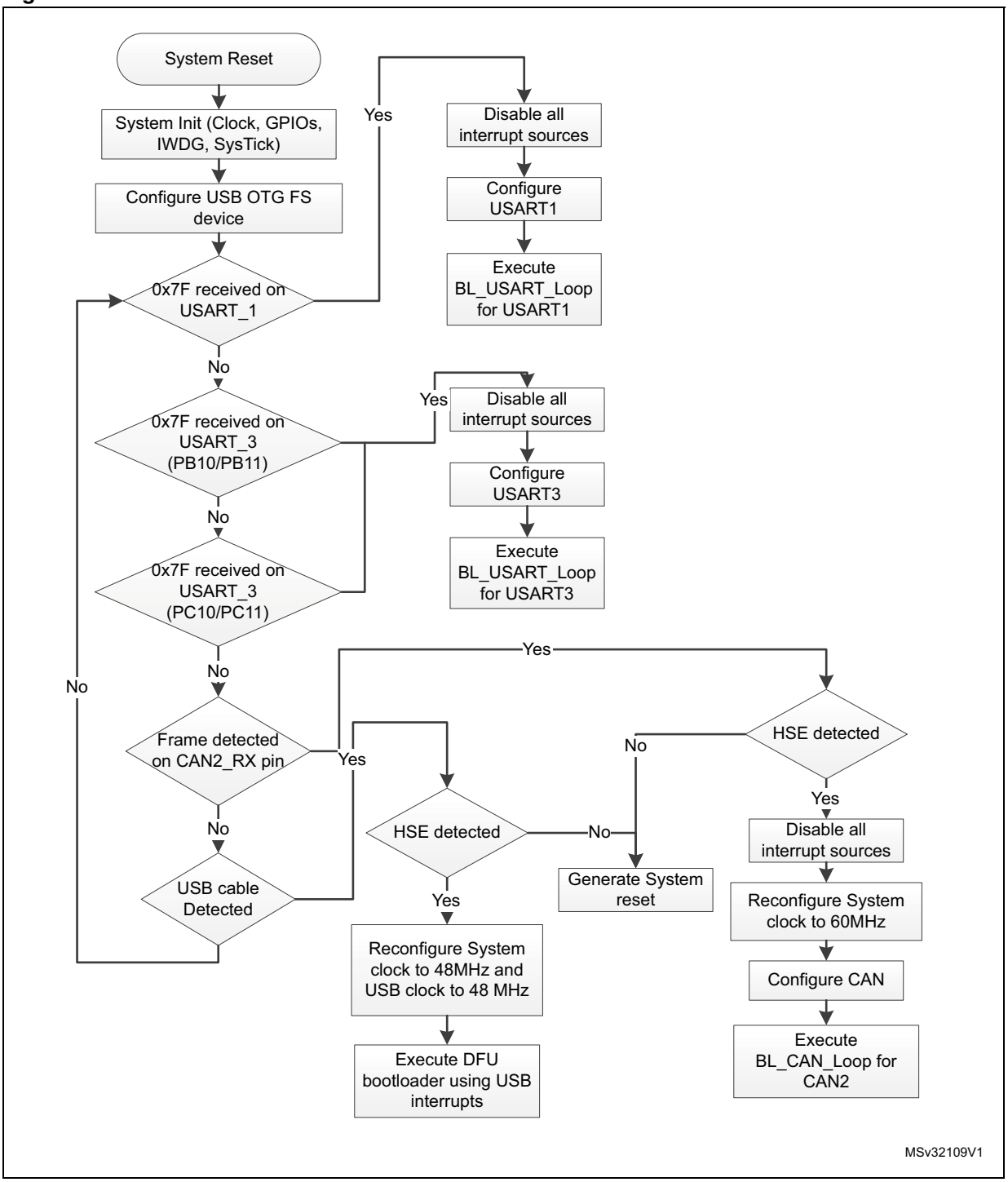
The supported HSE frequencies are multiple of 1 MHz ranging from 4 to 26 MHz. Any other values leads to an unexpected behavior, CAN bootloader firmware enters an infinite loop and waits until it receives a message. If the external clock is not present, a system reset is generated.

If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters the DFU bootloader loop waiting for any DFU bootloader command.

To use the USART or the CAN bootloader, it is mandatory that no USB Host is connected to the USB peripheral during the selection phase. Once the USART or CAN bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except commands which generate a system reset.

Once one interface is selected for the bootloader, all other interfaces are disabled. The figure below shows the bootloader selection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 9. Bootloader selection for STM32F40xxx/41xxx devices



11.4 Important considerations

STM32F40xxx/41xxx bootloader has some specific features that should be taken into consideration:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), STM32F40xxx/41xxx device bootloader firmware supports OTP memory (512 bytes from 0x1FFF 7800 to 0x1FFF 7A00, refer to RM0090 reference manual for more information).
- **OTP memory** can be read and written but cannot be erased using Erase command. When writing in an OTP memory location, make sure that the relative protection bit (in the last 16 bytes of the OTP memory) is not reset.
- **Option bytes**

  Address is 0x1FFFC000. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to PM0081 programming manual for more details about protection levels.
- **Read protect** command corresponds to Level 1 protection.
- **Read unprotect** command corresponds to Level 0 protection.
- **Mass erase** command on STM32F40xxx/41xxx takes longer than on other STM32 devices due to their memory density. Make sure that the timeout used by your host interface to wait for an acknowledge event after sending a Mass erase command is sufficient.
- **Voltage range configuration**

  The Voltage Range can be updated on the fly by the bootloader software. The Voltage Range is set to its default value at each bootloader software startup (after system reset or jump to the bootloader code). The bootloader software allows modifying this parameter through a virtual memory location. This memory location is not physical but can be read and written using usual bootloader read/write operations according to the protocol in use (USART,CAN or DFU). This memory location contains 4 bytes which are described in *Table 22*. It can be accessed by 1, 2, 3 or 4 bytes. However, reserved bytes should remain at their default values (0xFF), otherwise the request will be NACKed.

Table 25. STM32F40xxx/41xxx Voltage Range configuration using bootloader

Address	Size	Description
0xFFFF0000	1 byte	This byte controls the current value of Voltage Range: 0x00: Voltage Range [1.8V, 2.1V] 0x01: Voltage Range [2.1V, 2.4V] 0x02: Voltage Range [2.4V, 2.7V] 0x03: Voltage Range [2.7V, 3.6V] 0x04: Voltage Range [2.7V, 3.6V] and Double Word write/erase operation is used. In this case it is mandatory to supply 9 V through VPP pin (refer to PM0081 for more details about Double-Word write procedure). Other: All other values are not supported and will be NACKed.
0xFFFF0001	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0002	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0003	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.

11.5 Bootloader version

The following table shows the STM32F40xxx/41xxx bootloader version.

Table 26. STM32F40xxx/41xxx bootloader version

Bootloader version number	Description	Known limitations
V3.0	Initial bootloader version.	– When a Read Memory command or Write Memory command is issued with an unsupported memory address and a correct address checksum (ie. address 0x6000 0000), the command is aborted by the bootloader device, but the NACK (0x1F) is not sent to the host. As a result, the next 2 bytes (which are the number of bytes to be read/written and its checksum) are considered as a new command and its checksum(1). – Option bytes, OTP and Device Feature descriptors (in DFU interface) are set to “g” instead of “e” (not erasable memory areas).
V3.1	Fix V3.0 limitations. DFU interface robustness enhancement.	– For the USART interface, two consecutive NACKs (instead of 1 NACK) are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

1. If the “number of data - 1” (N-1) to be read/written is not equal to a valid command code (0x00, 0x01, 0x02, 0x11, 0x21, 0x31, 0x43, 0x44, 0x63, 0x73, 0x82 or 0x92), then the limitation is not perceived from the host since the command is NACKed anyway (as an unsupported new command).

12 STM32F051x4, STM32F051x6 and STM32F051x8 device bootloader

Throughout this section, STM32F051xx is used to refer to STM32F051x4, STM32F051x6 and STM32F051x8 devices.

12.1 Bootloader configuration

The bootloader embedded in STM32F051xx devices supports two serial interfaces: USART1 and USART2 peripherals.

The following table shows the required hardware resources of STM32F051xx devices used by the bootloader in System memory boot mode.

Table 27. STM32F051xx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	Clock Source	HSI Enabled	The System clock is equal to 24 MHz (using PLL clocked by HSI). 1 Flash Wait State.
	RAM	-	2 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
	System memory	-	3 Kbytes starting from address 0x1FFFEC00, contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset in case the hardware IWDG option was previously enabled by the user.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode.
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode.
	USART2_RX (PA15) pin must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is 8 bits, even parity and 1 Stop bit.
	USART2_RX pin	Input	PA15 pin: USART2 in reception mode.
	USART2_TX pin	Output	PA14 pin: USART2 in transmission mode.
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

Note: After the STM32F051x6 and STM32F051x8 devices have booted in bootloader mode, the serial wire debug (SWD) communication is no more possible until the system is reset,

because SWD uses PA14 pin (SWCLK) which is already used by the bootloader (USART2_RX).

12.2 Bootloader hardware requirements

The hardware required to put the STM32F051xx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to value 1. The setting of this bit can be done through STLINK utility or an equivalent tool.

To connect to the STM32F051xx devices during System memory boot mode, the following conditions have to be verified:

- The peripheral RX pins not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader, the USART2_RX (PA15) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader, the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- The peripheral to be used has to be connected through an RS232 serial interface (example, ST3232 RS232 transceiver) which must be:
  - directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used
  - directly connected to the USART2_RX (PA15) and USART2_TX (PA14) pins when USART2 is used.
- The USART1_CK, USART1_CTS and USART1_RTS pins are not used. As a result, the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART2.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM320518-EVAL board. For more details about this, refer to the “STM320518-EVAL board user manual” (UM1537), available from the STMicroelectronics website: *http://www.st.com*.

12.3 Bootloader selection

The STM32F051xx devices embedded bootloader supports two peripheral interfaces: USART1 and USART2. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash memory.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals which are not used by this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as

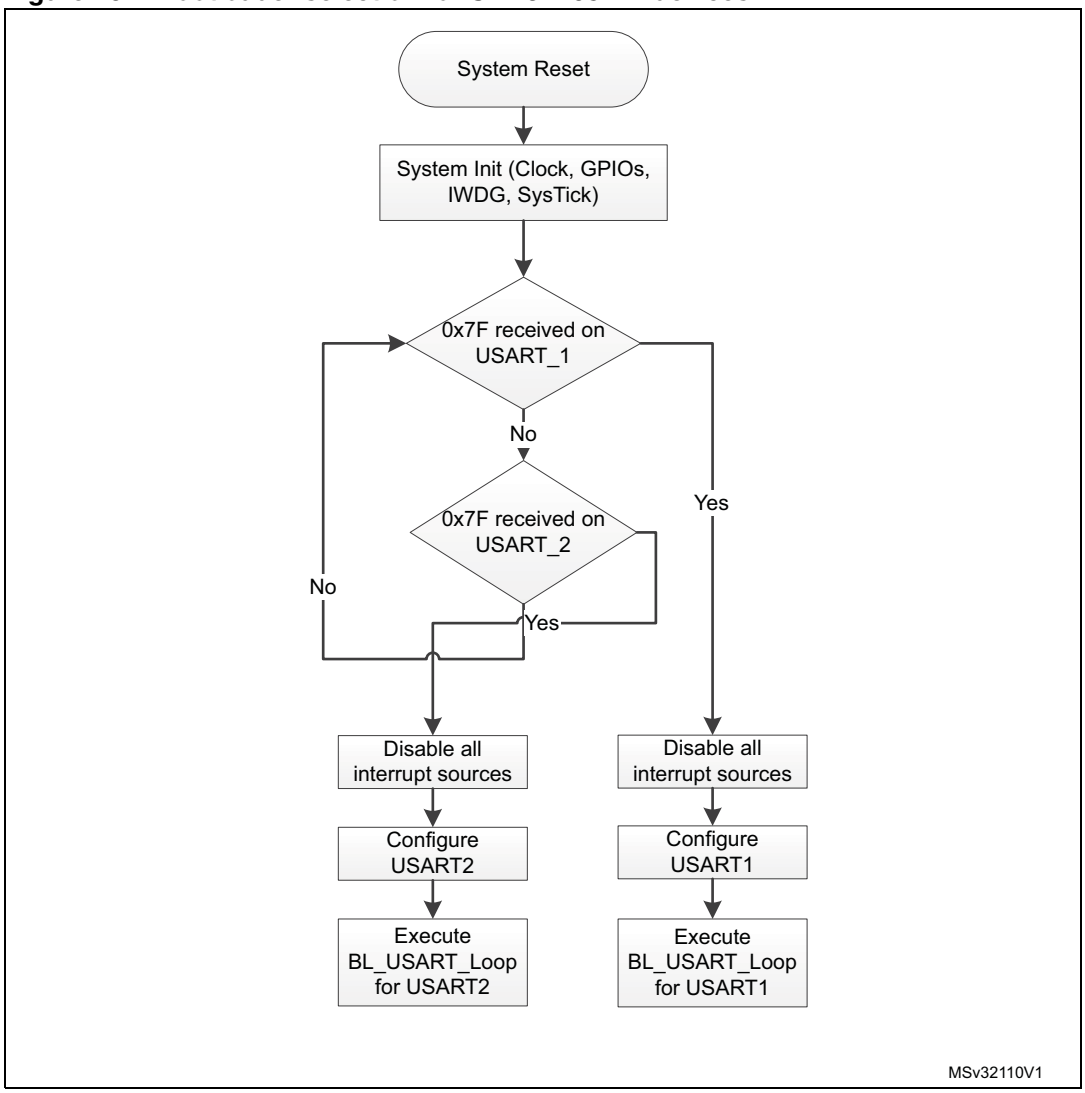
described below. Refer to Section 12.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the autobaudrate sequence and then enters a loop, waiting for any USART bootloader command.

Once one interface is selected for the bootloader, and the other interface is disabled.

The figure below shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 10. Bootloader selection for STM32F051xx devices



12.4 Important considerations

The bootloader of the STM32F051xx devices has some specific features that should be taken into consideration, as described below:

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to RM0091 reference manual for more details about protection levels.
- Read protect command corresponds to the Level 1 protection.
- Read unprotect command corresponds to the Level 0 protection.
- Jump to user application

  If an application that uses interrupts has to be downloaded to Flash memory address 0x0800 0000 by the bootloader firmware, then that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x0000 0000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to RM0091 reference manual for more details about software memory mapping).

  If the application is loaded into Flash memory at an address different from 0x08000000, then the vector table has to be mapped to SRAM and the SRAM should in this case be mapped by software to address 0x0000 0000.

12.5 Bootloader version

The following table lists the STM32F051xx bootloader versions.

Table 28. STM32F051xx bootloader versions

Bootloader version number	Description	Known limitations
V2.1	Initial bootloader version	When the user application configures a value of HSI TRIM bits (in RCC_CR register) and then jumps to the bootloader, the HSITRIM value is reset to its default value (0) at bootloader startup. For the USART interface, two consecutive NACKs instead of 1 NACK are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

13 STM32F050x4 and STM32F050x6 device bootloader

Throughout this chapter, STM32F050xx refers to the STM32F050x4 and STM32F050x6 devices.

13.1 Bootloader configuration

The bootloader embedded in the STM32F050xx devices supports one serial interface: USART1 peripheral.

Table 29 shows the required hardware resources of the STM32F050xx devices used by the bootloader in System memory boot mode.

Table 29. STM32F050xx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	Clock Source	HSI Enabled	The System clock is equal to 24 MHz (using PLL clocked by HSI). 1 Flash Wait State.
	RAM	-	2 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
	System memory	-	3 Kbytes starting from address 0x1FFFEC00 contain the bootloader firmware.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset in case the hardware IWDG option was previously enabled by the user.
USART1 bootloader (on PA10/PA9)	USART1	Enabled	Once initialized, the USART1 configuration is 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode.
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode.
	USART1_RX (PA15) pin must be kept at a high or low level during the detection phase.		
USART1 bootloader (on PA14/PA15)	USART1	Enabled	Once initialized, the USART1 configuration is 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA15 pin: USART1 in reception mode.
	USART1_TX pin	Output	PA14 pin: USART1 in transmission mode.
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host.

Note: After the STM32F050x4 and STM32F050x6 devices have booted in bootloader mode, serial wire debug (SWD) communication is no longer possible until the system is reset. This is because the SWD uses the PA14 pin (SWCLK) which is already used by the bootloader (USART2_RX).

13.2 Bootloader hardware requirements

The hardware required to put the STM32F050xx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to value 1. The setting of this bit can be done through the STLINK utility or an equivalent tool.

To connect to the STM32F050xx devices during System memory boot mode, the following conditions have to be verified.

- The peripheral RX pins not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 (PA10/PA9) is used to connect to the bootloader, the USART1_RX (PA15) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART1 (PA14/PA15) is used to connect to the bootloader, the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- The peripheral to be used has to be connected through an RS-232 serial interface (example, ST3232 RS-232 transceiver) which must be:
  - directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 (PA10/PA9) is used
  - directly connected to the USART1_RX (PA15) and USART1_TX (PA14) pins when USART1 (PA14/PA15) is used.
- The USART1_CK, USART1_CTS and USART1_RTS pins are not used. As a result, the application can use these pins for other peripherals or GPIOs.

13.3 Bootloader selection

The embedded bootloader of the STM32F050xx devices supports one peripheral interface, USART1. This peripheral interface mapping can be used to communicate with the bootloader and download the application code to the internal Flash memory.

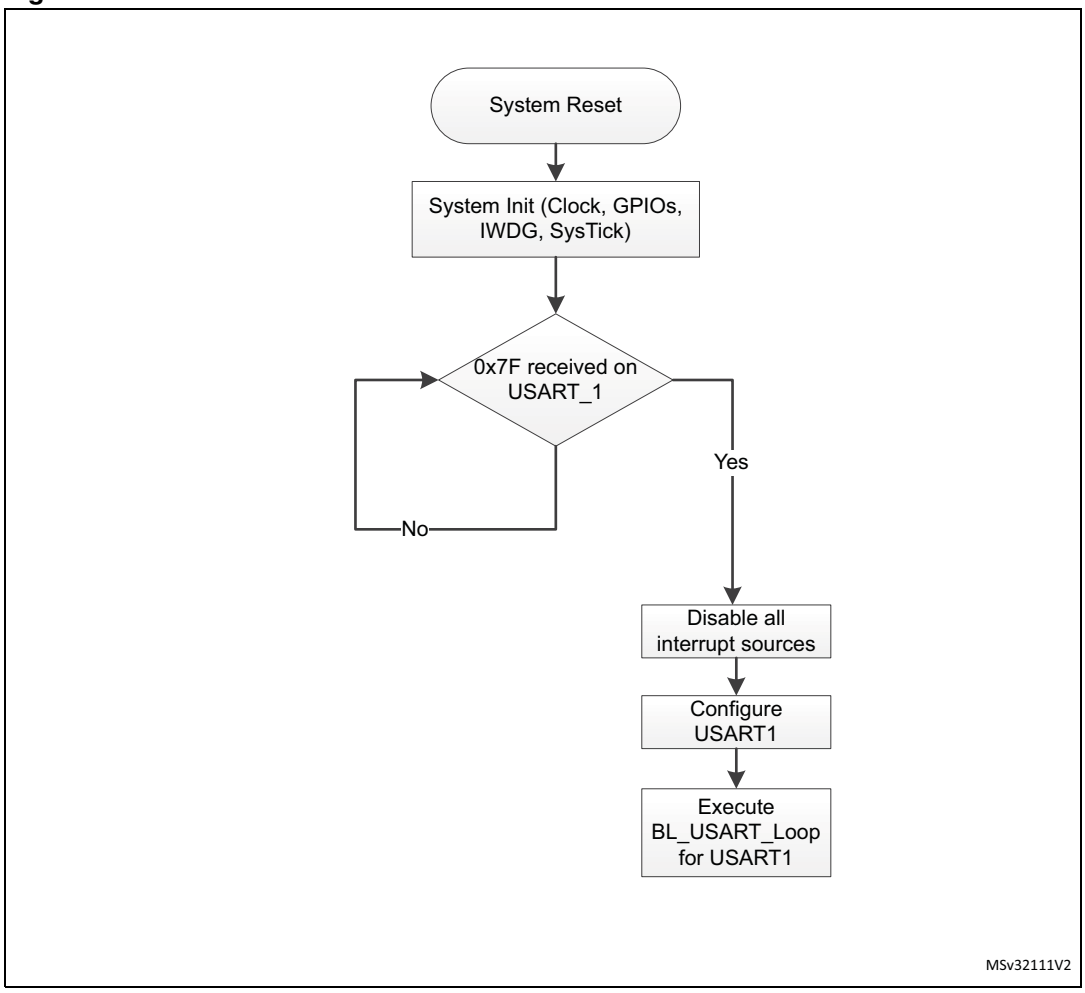
The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripheral which are not used by this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 13.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1, connect the serial cable to the desired mapping. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto baudrate sequence and then enters a loop, waiting for any USART bootloader command.

Figure 11 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 11. Bootloader selection for STM32F050xx devices



13.4 Important considerations

The bootloader of the STM32F050xx devices has some specific features that should be taken into consideration, as described below.

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the RM0091 reference manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands correspond to the Level 0 protection.
- Jump to user application.

  If an application that uses interrupts has to be downloaded to the Flash memory address 0x08000000 by the bootloader firmware, then that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x00000000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to the RM0091 reference manual for more details about software memory mapping).

  If the application is loaded into the Flash memory at an address different to 0x08000000, the vector table has to be relocated to start from the address where the application is loaded.

13.5 Bootloader version

Table 30 lists the STM32F050xx bootloader versions.

Table 30. STM32F050xx bootloader versions

Bootloader version number	Description	Known limitations
V1.0	Initial bootloader version	For the USART interface, two consecutive NACKs instead of 1 NACK are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

14 STM32F372xx and STM32F373xx device bootloader

Throughout this section, STM32F37xxx is used to refer to the STM32F372xx and STM32F373xx devices.

14.1 Bootloader configuration

The bootloader embedded in the STM32F37xxx devices supports three serial interfaces: USART1, USART2 and DFU (USB).

Table 31 shows the required hardware resources of the STM32F37xxx devices used by the bootloader in System memory boot mode.

Table 31. STM32F37xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	At startup, the system clock frequency is configured to 48 MHz using the HSI. If an external clock (HSE) is not present, the system is kept clocked from the HSI.
		HSE enabled	The external clock can be used for all bootloader interfaces and should have one the following values 24, 18,16, 12, 9, 8, 6, 4, 3 MHz. The PLL is used to generate the USB 48 MHz clock and the 48 MHz clock for the system clock.
		-	The clock security system (CSS) interrupt is enabled for the DFU bootloader. Any failure (or removal) of the external clock generates system reset.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	System memory	-	8 Kbytes starting from address 0x1FFFD800. This area contains the bootloader firmware
	RAM	-	5 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.

Table 31. STM32F37xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode
	USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB_DM pin	Alternate function, automatically controlled by the USB	PA11: USB Send-Receive data line
	USB_DP pin		PA12: USB Send-Receive data line
	Interrupts	Enabled	USB Low Priority interrupt vector is enabled and used for USB DFU communication.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

There are two cases of operation:

- If HSE is present and has a value of 24, 18, 16, 12, 9, 8, 6, 4, or 3 MHz, the system clock is configured to 48 Mhz with HSE as clock source. The DFU interface, USART1 and USART2 are functional and can be used to communicate with the bootloader device.
- If HSE is not present, the HSI is kept as default clock source and only USART1 and USART2 are functional.

The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around the theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

14.2 Bootloader hardware requirements

The hardware required to put the STM32F37xxx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to value 1. The setting of this bit can be done through the STLINK utility or an equivalent tool.

To connect to the STM32F37xxx devices during System memory boot mode, the following conditions have to be verified.

- The RX pin of the peripherals that are not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB) is used to connect to the bootloader: the USART1_RX (PA10) and USART2_RX (PD6) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- For the DFU interface, a 1.5 K(+/- 5%) pull-up resistor must be connected to USB_DP (PA12). For more information, refer to Universal Serial Bus Specification Revision 2.0, section 7.1.5.1.
- Connection to the peripheral should be done through:
  - an RS-232 serial interface (example, ST3232 RS-232 transceiver) which has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
  - a certified USB cable which has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used. As a result, the application can use these pins for other peripherals or GPIOs. This is also applicable for USART2.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM32373C-EVAL board. For more details about this, refer to the "STM32373C-EVAL board user manual", available from the STMicroelectronics website (www.st.com).

14.3 Bootloader selection

The STM32F37xxx embedded bootloader supports three serial interfaces: USART1, USART2 and DFU (USB). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

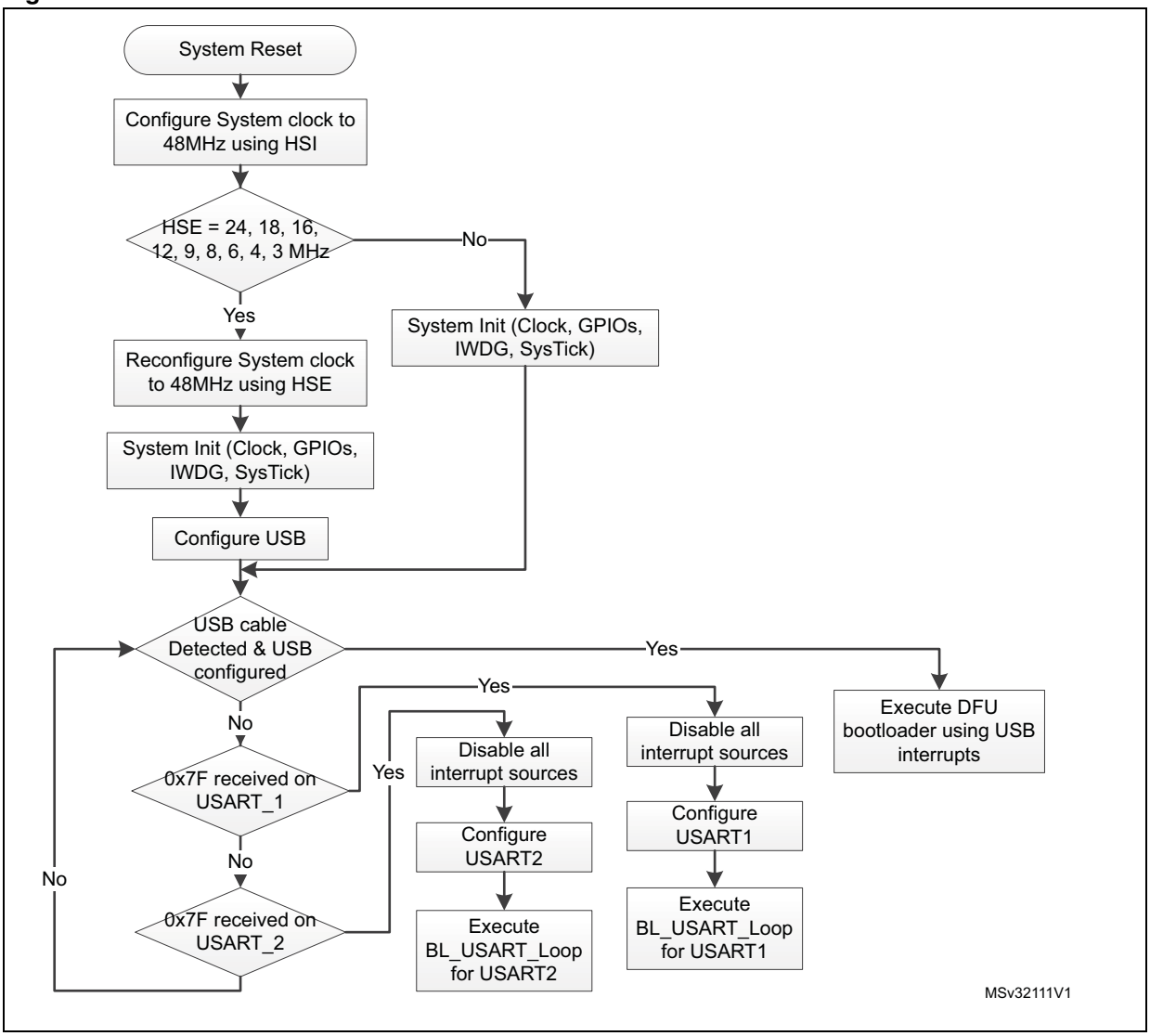
Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 14.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto baudrate sequence and then enters a loop, waiting for any USART bootloader command.

If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters a DFU bootloader loop waiting for any DFU bootloader command.

To use the USART bootloader, it is mandatory that no USB Host be connected to the USB peripheral during the selection phase (or in the case where the external clock is not present, even if a host is connected, the USB will never start). Once the USART bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except for the commands which generate a system reset. Once an interface is selected for the bootloader, the other interface is disabled. *Figure 12* shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 12. Bootloader selection for STM32F37xxx devices



14.4 Important considerations

The bootloader of the STM32F37xxx devices has some specific features that should be taken into consideration, as described below.

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the RM0313 reference manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands correspond to the Level 0 protection.
- Jump to user application

  If an application that uses interrupts has to be downloaded to the Flash memory address 0x08000000 by the bootloader firmware, then that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x00000000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to the RM0313 reference manual for more details about software memory mapping).

  If the application is loaded into the Flash memory at an address different from 0x08000000, then the vector table has to be relocated to start from the address where the application is loaded.

14.5 Bootloader version

Table 32 lists the bootloader versions for the STM32F37xxx devices.

Table 32. STM32F37xxx bootloader versions

Bootloader version number	Description	Known limitations
V4.1	Initial bootloader version	None

15 STM32F302xx and STM32F303xx device bootloader

Throughout this section, STM32F30xxx is used to refer to the STM32F302xx and STM32F303xx devices.

15.1 Bootloader configuration

The bootloader embedded in the STM32F30xxx devices supports three serial interfaces: USART1, USART2 and DFU (USB)

Table 33 shows the required hardware resources of the STM32F30xxx devices used by the bootloader in System memory boot mode.

Table 33. STM32F30xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	At startup, the system clock frequency is configured to 48 MHz using the HSI. If an external clock (HSE) is not present, the system is kept clocked from the HSI.
		HSE enabled	The external clock can be used for all bootloader interfaces and should have one the following values 24, 18,16, 12, 9, 8, 6, 4, 3 MHz. The PLL is used to generate the USB 48 MHz clock and the 48 MHz clock for the system clock.
		-	The clock security system (CSS) interrupt is enabled for the DFU bootloader. Any failure (or removal) of the external clock generates system reset.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	System memory	-	8 Kbytes starting from address 0x1FFFD800. This area contains the bootloader firmware
	RAM	-	5 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.

Table 33. STM32F30xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode
	USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins must be kept at a high or low level during the detection phase.		
DFU bootloader	USB_DM pin	Alternate function, automatically controlled by the USB	PA11: USB Send-Receive data line
	USB_DP pin		PA12: USB Send-Receive data line
	Interrupts	Enabled	USB Low Priority interrupt vector is enabled and used for USB DFU communication.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

There are two cases of operation:

- If HSE is present and has a value of 24, 18, 16, 12, 9, 8, 6, 4, or 3 MHz, the system clock is configured to 48 Mhz with HSE as clock source. The DFU interface, USART1 and USART2 are functional and can be used to communicate with the bootloader device.
- If HSE is not present, the HSI is kept as default clock source and only USART1 and USART2 are functional.

The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around the theoretical value), the bootloader might calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler was set to its maximum value by the bootloader).

15.2 Bootloader hardware requirements

The hardware required to put the STM32F30xxx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to value 1. The setting of this bit can be done through the STLINK utility or an equivalent tool.

To connect to the STM32F30xxx devices during System memory boot mode, the following conditions have to be verified.

- The RX pin of the peripherals that are not used in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10), USB_DM (PA11) and USB_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB) is used to connect to the bootloader: the USART1_RX (PA10) and USART2_RX (PD6) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- For the DFU interface, a 1.5 K(+/- 5%) pull-up resistor must be connected to USB_DP (PA12). For more information, refer to the Universal Serial Bus Specification Revision 2.0, section "7.1.5.1".
- Connection to the peripheral is to be performed through:
  - an RS-232 serial interface (example, ST3232 RS-232 transceiver) directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
  - a certified USB cable connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used. As a result, the application can use these pins for other peripherals or GPIOs. This is also applicable for USART2.

The user can control the BOOT0 and reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM32303C-EVAL board. For more details about this, refer to the "STM32303C-EVAL board user manual", available from the STMicroelectronics website (www.st.com).

15.3 Bootloader selection

The STM32F30xxx embedded bootloader supports three serial interfaces: USART1, USART2 and DFU (USB). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

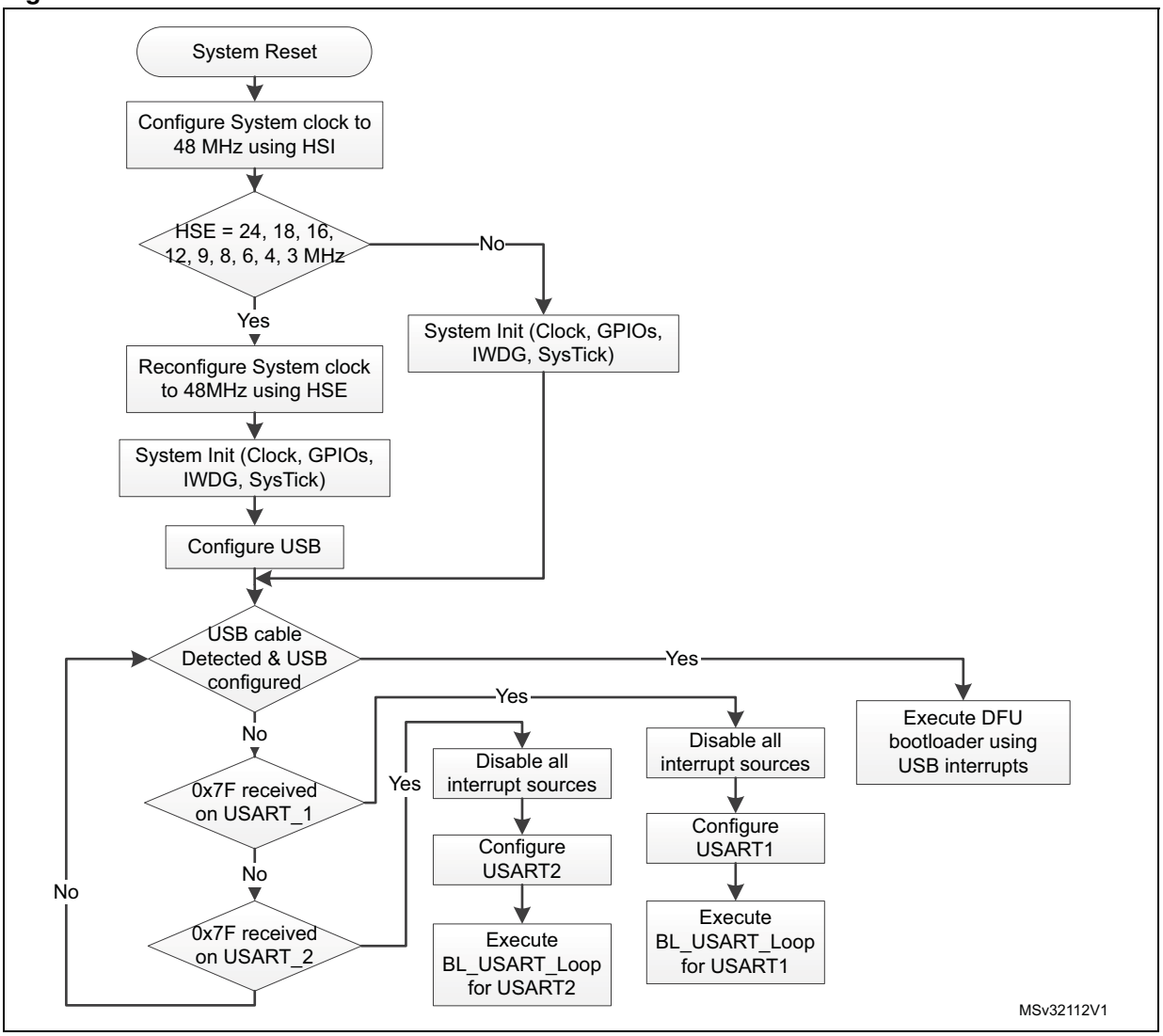
The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 15.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baudrate sequence and then enters a loop, waiting for any USART bootloader command. If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters into a DFU bootloader loop waiting for any DFU bootloader command.

To use the USART bootloader, it is mandatory that no USB Host be connected to the USB peripheral during the selection phase (or in the case where the external clock is not present, even if a host is connected, the USB will never start). Once the USART bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except for the commands which generate a system reset. Once an interface is selected for the bootloader, the other interface is disabled. *Figure 13* shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 13. Bootloader selection for STM32F30xxx devices



15.4 Important considerations

The bootloader of the STM32F30xxx devices has some specific features that should be taken into consideration, as described below.

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the RM0316 reference manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands correspond to the Level 0 protection.
- Jump to user application

  If an application that uses interrupts has to be downloaded to the Flash memory address 0x08000000 by the bootloader firmware, that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x00000000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to the RM0316 reference manual for more details about software memory mapping).

  If the application is loaded into the Flash memory at an address different to 0x08000000, the vector table has to be relocated to start from the address where the application is loaded.

15.5 Bootloader version

Table 34 lists the bootloader versions for the STM32F30xxx devices.

Table 34. STM32F30xxx bootloader versions

Bootloader version number	Description	Known limitations
V4.1	Initial bootloader version	None

16 STM32F383xx device bootloader

Throughout this section, STM32F38xxx is used to refer to the STM32F383xx devices.

16.1 Bootloader configuration

The bootloader embedded in the STM32F38xxx devices supports three serial interfaces: USART1, USART2 and I2C1.

Table 35 shows the required hardware resources of the STM32F38xxx devices used by the bootloader in System memory boot mode.

Table 35. STM32F38xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock frequency is 8 MHz using the HSI.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user). Window feature is disabled.
	System memory	-	8 Kbytes starting from address 0x1FFFD800. This area contains the bootloader firmware
	RAM	-	4 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode.
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode.
	USART2_RX (PD6) pin must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode.
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode.
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.

Table 35. STM32F38xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
I2C1 bootloader	I2C1	Enabled	I2C clock source is HSI. Once initialized, the I2C1 configuration is: I2C speed: 400 KHz, 7-bit address, slave mode, slave address: 0x6E, analog filter ON.
	I2C1_SCL pin	Input/ Output	PB6 pin: I2C1 clock line is used in open-drain mode.
	I2C1_SDA pin	Input/ Output	PB7 pin: I2C1data line is used in open-drain mode.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

The system clock is derived from the embedded internal high-speed RC for the I2C1, USART1 and USART2 bootloaders. No external quartz is required in this case for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in *Table 35*) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler has been previously set to its maximum value by the bootloader).

16.2 Bootloader hardware requirements

The hardware required to put the STM32F38xxx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while the nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to 1. The setting of this bit can be done through the STLink utility or an equivalent tool.

To connect to the STM32F38xxx devices during System memory boot mode, the following conditions have to be verified.

- The RX pins of the peripherals unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral.
  - If the I2C1 is used to connect to the bootloader: an I2C interface (I2C Master) has to be directly connected to the I2C1_SCL (PB6) and I2C1_SDA (PB7) pins.
  - If the USART is used to connect to the bootloader: an RS-232 serial interface (example, ST3232 RS-232 transceiver) which must be:

    –directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used.

    –directly connected to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
- The USART1_CK, USART1_CTS and USART1_RTS pins are not used. Therefore, the application can use these pins for other peripherals or GPIOs. The same is applicable to USART2.
- A 1.8 Kohm pull-up resistor has to be connected to both the SDA and SCL lines. This value is used to fix the timing register value.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM32373C-EVAL board. For more details about this, refer to the "STM32373C-EVAL board user manual" , available from the STMicroelectronics website (www.st.com).

16.3 Bootloader selection

The embedded bootloader of the STM32F38xxx devices supports three peripheral interfaces: I2C1, USART1 and USART2. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

To use the I2C bootloader on the I2C1 interface, connect the host master and the STM32 I2C1 slave target. The master and the STM32 target are connected together via the data (SDA) and clock (SCL) pins. Once the STM32 I2C interface recognizes its own address (0x6E) on the bus, the bootloader firmware enters an infinite loop and waits until it receives an I2C bootloader command.

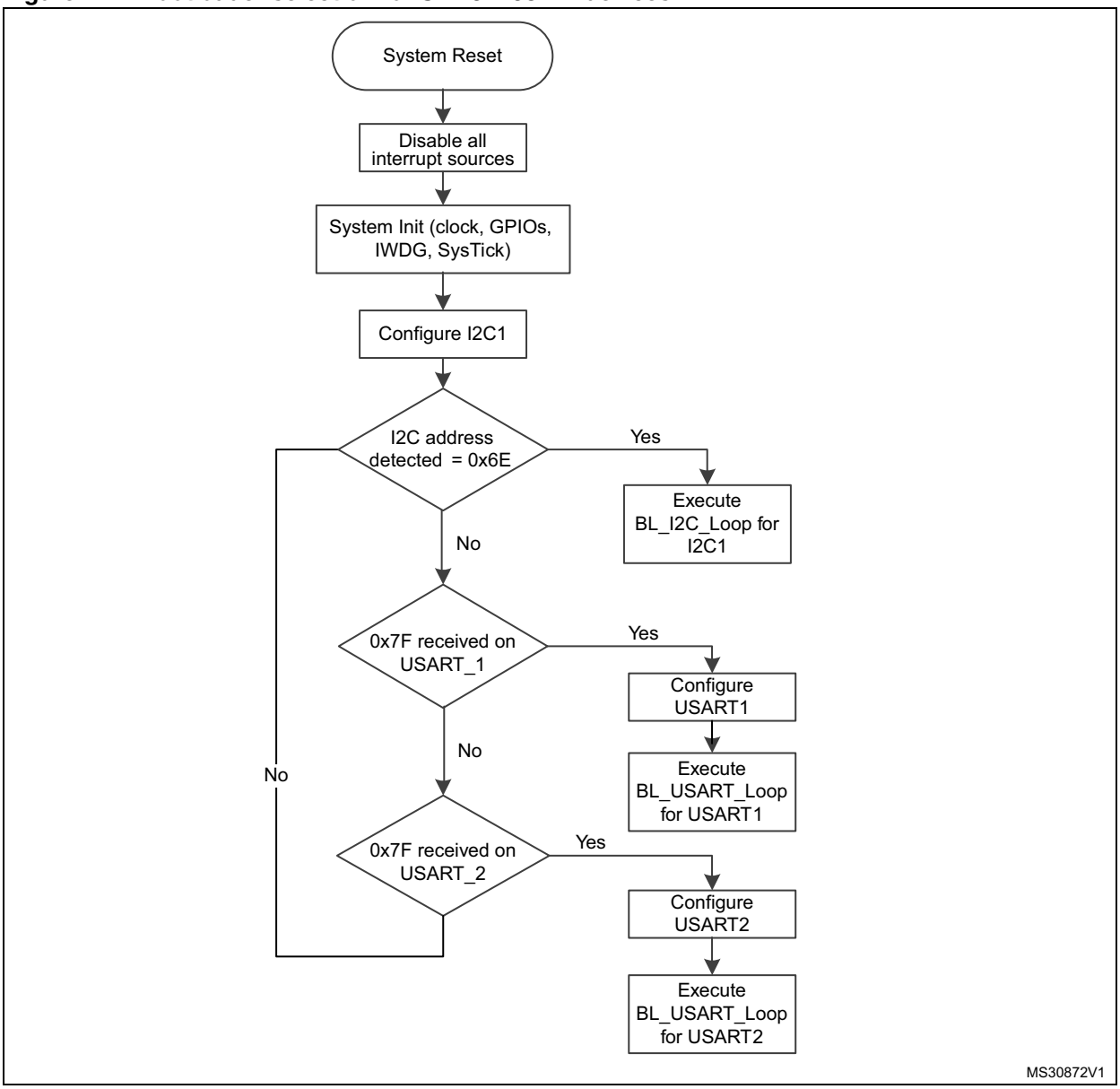
To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto baudrate sequence and then enters into an infinite loop and waits until it receives a USART bootloader command.

Note that for the USART1 and USART2 interfaces, the maximum baudrate supported by the bootloader is 57600 baud.

Once one interface is selected for the bootloader, the other interface is disabled.

Figure 14 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 14. Bootloader selection for STM32F38xxx devices



16.4 Important considerations

The bootloader of the STM32F38xxx devices has some specific features that should be taken into consideration, as described below.

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the RM0313 reference manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands correspond to the Level 0 protection.
- Jump to user application

  If an application that uses interrupts has to be downloaded to the Flash memory address 0x08000000 by the bootloader firmware, then that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x00000000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to the RM0313 reference manual for more details about software memory mapping).

  If the application is loaded into the Flash memory at an address different to 0x08000000, the vector table has to be relocated to start from the address where the application is loaded.

16.5 Bootloader version

Table 36 lists the bootloader versions for the STM32F38xxx devices.

Table 36. STM32F38xxx bootloader versions

Bootloader version number	Description	Known limitations
V5.0	Initial bootloader version	For USART1 and USART2 interfaces, the maximum baudrate supported by the bootloader is 57600 baud.

17 STM32F313xx device bootloader

Throughout this section, STM32F31xxx is used to refer to the STM32F313xx devices.

17.1 Bootloader configuration

The bootloader embedded in the STM32F31xxx devices supports three serial interfaces: USART1, USART2 and I2C1.

Table 37 shows the required hardware resources of the STM32F31xxx devices used by the bootloader in System memory boot mode.

Table 37. STM32F31xxx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock frequency is 8 MHz using the HSI.
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value and is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user). Window feature is disabled.
	System memory	-	8 Kbytes starting from address 0x1FFFD800. This area contains the bootloader firmware.
	RAM	-	5 Kbytes starting from address 0x20000000 are used by the bootloader firmware.
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode.
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode.
	USART2_RX (PD6) pin must be kept at a high or low level during the detection phase.		
USART2 bootloader	USART2	Enabled	Once initialized, the USART2 configuration is: 8 bits, even parity and 1 Stop bit. The USART2 uses its remapped pins.
	USART2_RX pin	Input	PD6 pin: USART2 in reception mode.
	USART2_TX pin	Output	PD5 pin: USART2 in transmission mode.
	USART1_RX (PA10) pin must be kept at a high or low level during the detection phase.		
USART1 and USART2 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloader.

Table 37. STM32F31xxx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
I2C1 bootloader	I2C1	Enabled	I2C clock source is HSI. Once initialized, the I2C1 configuration is: I2C speed: 400 KHz, 7-bit address, slave mode, slave address: 0x6E, analog filter ON.
	I2C1_SCL pin	Input/ Output	PB6 pin: I2C1 clock line is used in open-drain mode.
	I2C1_SDA pin	Input/ Output	PB7 pin: I2C1data line is used in open-drain mode.
	USART1_RX (PA10) and USART2_RX (PD6) pins must be kept at a high or low level during the detection phase.		

The system clock is derived from the embedded internal high-speed RC for I2C1, USART1 and USART2 bootloaders. No external quartz is required in this case for the bootloader code.

After downloading the application binary, if you choose to execute the Go command, all peripheral registers used by the bootloader (shown in the above table) are initialized to their default reset values before jumping to the user application.

If the user application uses the IWDG, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler has been previously set to its maximum value by the bootloader).

17.2 Bootloader hardware requirements

The hardware required to put the STM32F31xxx devices into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high while the nBOOT1 bit in the option bytes (starting at address 0x1FFFF800) is set to 1. The setting of this bit can be done through the STLink utility or an equivalent tool.

To connect to the STM32F31xxx devices during System memory boot mode, the following conditions have to be verified.

- The RX pins of the peripherals unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART2_RX (PD6) pin has to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART2 is used to connect to the bootloader: the USART1_RX (PA10) pin has to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral.
  - If I2C1 is used to connect to the bootloader: an I2C interface (I2C master) has to be directly connected to the I2C1_SCL (PB6) and I2C1_SDA (PB7) pins.
  - If the USART is used to connect to the bootloader: an RS-232 serial interface (for example, ST3232 RS-232 transceiver) which must be:

    –directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used.

    –directly connected to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
- The USART1_CK, USART1_CTS and USART1_RTS pins are not used; therefore the application can use these pins for other peripherals or GPIOs. The same is applicable for USART2.
- A 1.8 Kohm pull-up resistor has to be connected to both the SDA and SCL lines. This value is used to fix the timing register value.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM32303C-EVAL board. For more details about this, refer to the "STM32303C-EVAL board user manual", available from the STMicroelectronics website (www.st.com).

17.3 Bootloader selection

The embedded bootloader of the STM32F31xxx devices supports three peripheral interfaces: I2C1, USART1 and USART2. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

To use the I2C bootloader on I2C1 interface, connect the host master and the STM32 I2C1 slave target. The master and the STM32 target are connected together via the data (SDA) and clock (SCL) pins. Once the STM32 I2C interface recognizes its own address (0x6E) on the bus, the bootloader firmware enters an infinite loop and waits until it receives an I2C bootloader command.

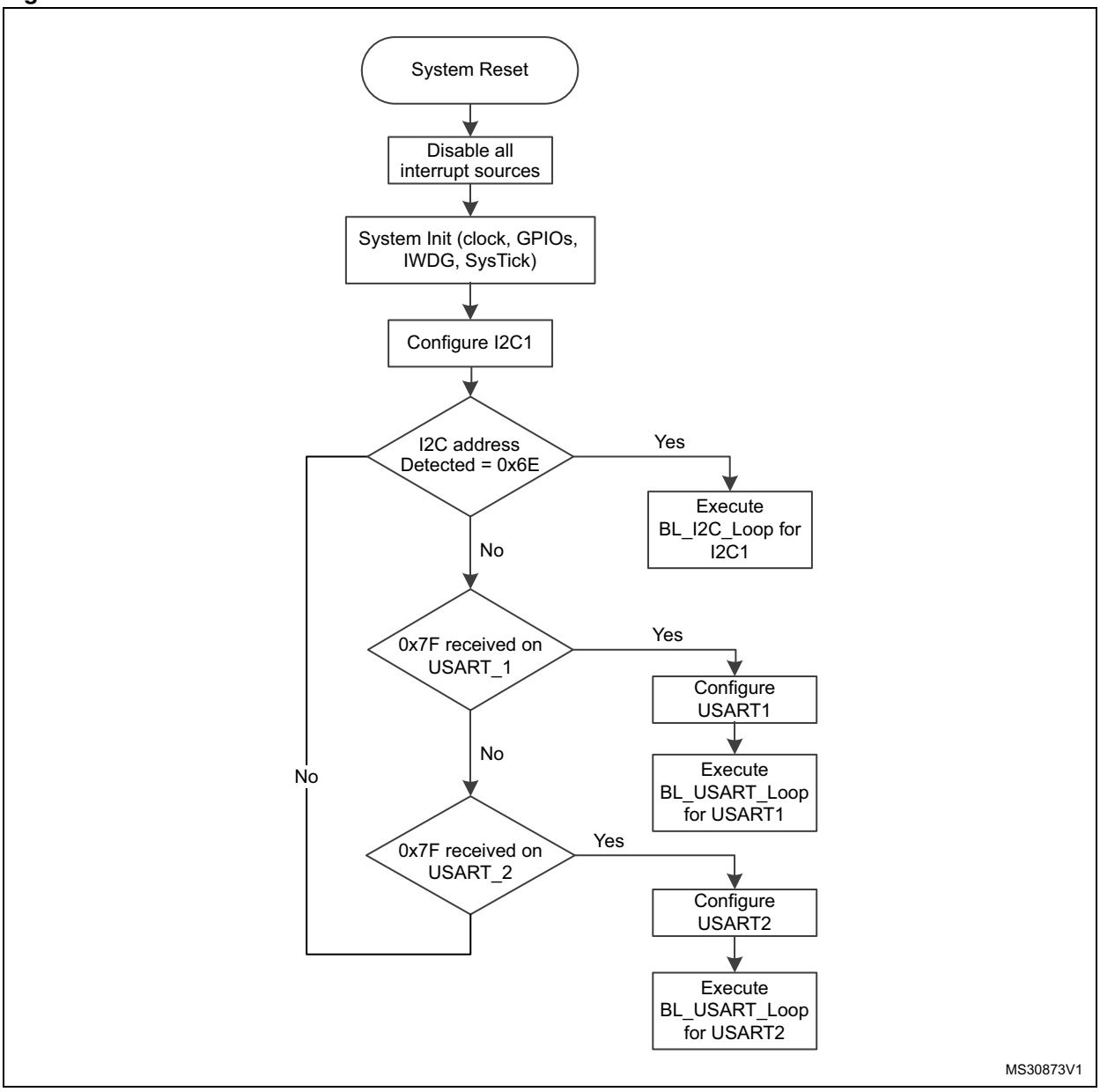
To use the USART bootloader on USART1 or USART2, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto baudrate sequence and then enters an infinite loop and waits until it receives a USART bootloader command.

Note that for the USART1 and USART2 interfaces, the maximum baudrate supported by the bootloader is 57600 baud.

Once one interface is selected for the bootloader, the other interface is disabled.

Figure 15 shows the bootloader detection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 15. Bootloader selection for STM32F31xxx devices



17.4 Important considerations

The bootloader of the STM32F31xxx devices has some specific features that should be taken into consideration.

- Option byte

  Address is 0x1FFFF800. They allow three levels of protection:

  – Level 0
  – Level 1
  – Level 2

  Refer to the RM0313 reference manual for more details about protection levels.
- Read protect commands correspond to the Level 1 protection.
- Read unprotect commands correspond to the Level 0 protection.
- Jump to user application

  If an application that uses interrupts has to be downloaded to the Flash memory address 0x08000000 by the bootloader firmware, then that application must at startup and before any interrupt is enabled, map the Flash memory by software to the address 0x00000000. This can be done by programming the MEM_MODE bits of the SYSCFG_CFGR1 register (refer to the RM0313 reference manual for more details about software memory mapping).

  If the application is loaded into the Flash memory at an address different to 0x08000000, the vector table has to be relocated to start from the address where the application is loaded.

17.5 Bootloader version

Table 38 lists the bootloader versions for the STM32F31xxx devices.

Table 38. STM32F31xxx bootloader versions

Bootloader version number	Description	Known limitations
V5.0	Initial bootloader version	For USART1 and USART2 interfaces, the maximum baudrate supported by the bootloader is 57600 baud.

18 STM32F427xx and STM32F437xx device bootloader

Throughout this section, STM32F427xx/437xx is used to refer to the STM32F427xx and STM32F437xx devices.

18.1 Bootloader configuration

The bootloader embedded in the STM32F427xx/437xx devices supports four serial peripherals: USART1, USART3, CAN2 and DFU (USB FS device).

Table 39 shows the required hardware resources of the STM32F427xx/437xx devices used by the bootloader in System memory boot mode.

Table 39. STM32F427xx/437xx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock is equal to 24 MHz using the PLL. The HSI clock source is used at startup (interface detection phase) and when USARTx interfaces are selected (once CAN or DFU bootloader is selected, the clock source will be derived from external crystal).
		HSE enabled	The system clock is equal to 60 MHz. The HSE clock source is used only when the CAN or the DFU (USB FS Device) interfaces are selected. The external clock must provide a frequency multiple of 1 MHz and ranging from 4 MHz to 26 MHz.
		-	The Clock Security System (CSS) interrupt is enabled for the CAN and DFU bootloaders. Any failure (or removal) of the external clock generates system reset.
	RAM	-	8 Kbytes starting from address 0x20000000 are used by the bootloader firmware
	System memory	-	30424 bytes starting from address 0x1FFF0000, contain the bootloader firmware
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to [1.8V, 2.1V]. The voltage range can be configured in run time using bootloader commands. Note that in this range internal Flash write operations are allowed only in byte format (Half-Word, Word and Double-Word operations are not allowed).

Table 39. STM32F427xx/437xx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART1 bootloader	USART1	Enabled	Once initialized, the USART1 configuration is: 8 bits, even parity and 1 Stop bit.
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART3_RX (PB11), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PB10/PB11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PB11 pin: USART3 in reception mode
	USART3_TX pin	Output	PB10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PC10/PC11)	USART3	Enabled	Once initialized, the USART3 configuration is: 8 bits, even parity and 1 Stop bit.
	USART3_RX pin	Input	PC11 pin: USART3 in reception mode
	USART3_TX pin	Output	PC10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART3 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloaders.
CAN2 bootloader	CAN2	Enabled	Once initialized, the CAN2 configuration is: Baudrate 125 kbps, 11-bit identifier. **Note:** CAN1 is clocked during CAN2 bootloader execution because STM32F4xx CAN1 manages the communication between CAN2 and SRAM.
	CAN2_RX pin	Input	PB05 pin: CAN2 in reception mode
	CAN2_TX pin	Output	PB13pin: CAN2 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		

Table 39. STM32F427xx/437xx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
DFU bootloader	USB_OTG_FS	Enabled	USB OTG FS configured in Forced Device mode. USB_OTG_FS interrupt vector is enabled and used for USB DFU communications.
	USB_OTG_FS_DM pin	Input	PA11 pin: USB OTG FS DM line
	USB_OTG_FS_DP pin	Output	PA12pin: USB OTG FS DP line
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11) and CAN2_RX (PB05) pins must be kept at a high or low level during the detection phase.		
	TIM11	Enabled	This timer is used to determine the value of the external clock frequency. Once the external clock frequency is determined, the RCC system is configured to operate at 60 MHz system clock (using PLL).

Note: For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions, the internal oscillator precision is altered above the tolerance band (1% around the theoretical value), the bootloader may calculate a wrong HSE frequency value. In this case, the bootloader DFU interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for USARTx bootloaders. No external quartz is required in this case for the bootloader code. This internal clock is also used for the CAN and DFU (USB FS device) but only for the selection phase. An external clock multiple of 1 MHz (between 4 and 26 MHz) is required for CAN and DFU bootloader execution after the selection phase.

The CAN and DFU bootloaders implement an external clock detection mechanism allowing to determine the value of the external clock using the internal high-speed RC and TIM11 timer. The accuracy of this mechanism allows to detect only those frequencies that are multiples of 1 MHz and ranging from 4 to 26 MHz. Any other value is not supported and will result in unexpected behavior of the bootloader.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in the above table) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. Therefore, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler has been previously set to its maximum value by the bootloader).

18.2 Bootloader hardware requirements

The hardware required to put the STM32F427xx/437xx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F427xx/437xx during System memory boot mode, the following conditions have to be verified.

- The RX pins of the peripheral unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below.
  - If USART1 is used to connect to the bootloader: the USART3_RX (PC11 and PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PB10/PB11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PC10/PC11) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX pin (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If CAN2 is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB FS device) is used to connect to the bootloader: the USART1_RX (PA10), USART3_RX (PC11 and PB11) and CAN2_RX (PB05) pins have to be kept at a high or low level and must not be left floating during the detection phase.
- Connection to the peripheral.
  - An RS-232 serial interface (for example, ST3232 RS-232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used.
  - A CAN interface (CAN transceiver) has to be directly connected to the CAN2_RX (PB5) and CAN2_TX (PB13) pins.
  - A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same note is applicable for USART3.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS-232 serial interface which controls BOOT0 through the CTS line and Reset through the DCD line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM324x7I_EVAL board. For more details about this, refer to the STM324x7I_EVAL board user manual, available from the STMicroelectronics website (www.st.com).

18.3 Bootloader selection

The STM32F427xx/437xx embedded bootloader supports three peripheral interfaces: USART1, USART3 (on PB10/PB11 and PC10/PC11), CAN2 and DFU (USB FS device). Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 18.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART3, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baud rate sequence and then enters a loop, waiting for any USART bootloader command.

To use the CAN2 interface, connect the CAN cable to CAN2. Once the bootloader detects a frame on the CAN2_RX pin (PB5), the bootloader firmware enters a CAN loop and starts to determine the external clock frequency value.

The supported HSE frequencies are multiples of 1 MHz ranging from 4 to 26 MHz. Any other values lead to unexpected behavior, the CAN bootloader firmware enters an infinite loop and waits until it receives a message. If the external clock is not present, a system reset is generated.

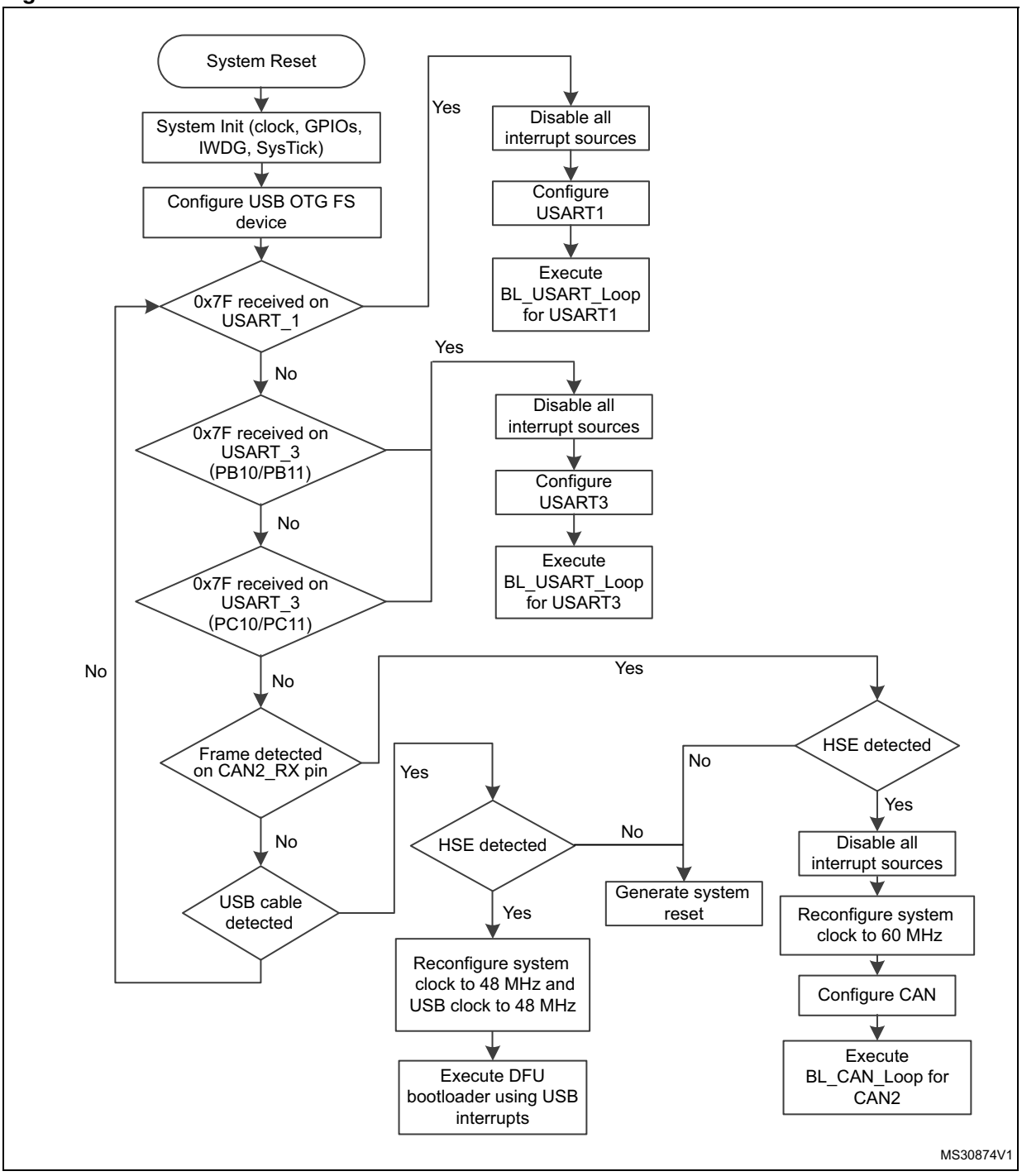
If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters the DFU bootloader loop waiting for any DFU bootloader command.

To use the USART or the CAN bootloader, it is mandatory that no USB Host be connected to the USB peripheral during the selection phase. Once the USART or CAN bootloader is selected, the user can plug a USB cable without impacting the selected bootloader execution except commands which generate a system reset.

Once one interface is selected for the bootloader, all other interfaces are disabled.

Figure 16 shows the bootloader selection mechanism. More details are provided in the sections corresponding to each peripheral bootloader.

Figure 16. Bootloader selection for STM32F427xx/437xx devices



18.4 Important considerations

The STM32F427xx/437xx bootloader has some specific features that should be taken into consideration.

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), the STM32F427xx/437xx device bootloader firmware supports OTP memory (528 bytes from 0x1FFF7800 to 0x1FFF7A0F).
- The OTP memory can be read and written but cannot be erased using the Erase command. When writing to an OTP memory location, make sure that the relative protection bit (in the last 16 bytes of the OTP memory) is not reset.
- Option bytes

  Their address is 0x1FFFC000 and 0x1FFEC000. They allow three levels of protection:
  - Level 0
  - Level 1
  - Level 2
- Read protect commands correspond to Level 1 protection.
- Read unprotect commands correspond to Level 0 protection.
- A mass erase command on the STM32F427xx/437xx takes longer than on other STM32 devices due to their memory density. Make sure that the timeout used by your host interface to wait for an acknowledge event after sending a Mass erase command is sufficient.
- Voltage range configuration

  The voltage range can be updated on the fly by the bootloader software. The voltage range is set to its default value at each bootloader software startup (after a system reset or jump to the bootloader code). The bootloader software allows modifying this parameter through a virtual memory location. This memory location is not physical but can be read and written using usual bootloader read/write operations according to the protocol in use (USART,CAN or DFU). This memory location contains 4 bytes which are described in *Table 40*. It can be accessed by 1, 2, 3 or 4 bytes. However, the reserved bytes should remain at their default values (0xFF), otherwise the request will be NACKed.

Table 40. STM32F427xx/437xx voltage range configuration using the bootloader

Address	Size	Description
0xFFFF0000	1 byte	This byte controls the current value of the voltage range. 0x00: voltage range [1.8 V, 2.1 V] 0x01: voltage range [2.1 V, 2.4 V] 0x02: voltage range [2.4 V, 2.7 V] 0x03: voltage range [2.7 V, 3.6 V] 0x04: voltage range [2.7 V, 3.6 V] and double word write/erase operation is used. In this case it is mandatory to supply 9 V through the VPP pin (refer to the PM0081 for more details about the double-word write procedure). Other: all other values are not supported and will be NACKed.
0xFFFF0001	1 byte	Reserved. 0xFF: default value. Other: all other values are not supported and will be NACKed.
0xFFFF0002	1 byte	Reserved. 0xFF: default value. Other: all other values are not supported and will be NACKed.
0xFFFF0003	1 byte	Reserved. 0xFF: default value. Other: all other values are not supported and will be NACKed.

If the application is loaded into the Flash memory at an address different to 0x08000000, then the vector table has to be relocated to start from the address where the application is loaded.

18.5 Bootloader version

Table 41 shows the STM32F427xx/437xx bootloader version.

Table 41. STM32F427xx/437xx bootloader version

Bootloader version number	Description	Known limitations
V3.0	Initial bootloader version.	For the USART interface, two consecutive NACKs instead of 1 NACK are sent when a Read Memory or Write Memory command is sent and the RDP level is active.

19 STM32F429xx and STM32F439xx device bootloader

Through all this section STM32F429xx/439xx will be used as reference to STM32F429xx and STM32F439xx devices.

19.1 Dual bank boot feature

The STM32F429xx/439xx devices have two Flash memory banks: Bank 1 and Bank 2. They feature an additional boot mechanism which allows booting from Bank 2 or Bank 1 depending on BFB2 bit status **(bit 4 in the user option bytes located at 0x1FFF C000).**

- When the BFB2 bit is set and the boot pins are configured to boot from Flash memory (BOOT0 = 0 and BOOT1 = x), after reset the device boots from the System memory and executes the embedded bootloader code which implements the dual bank Boot mode:
  a) The code first checks Bank 2. If it contains a valid code (see note below), it jumps to the application code located in Bank 2 and leaves the Bootloader.
  b) If the Bank 2 code is not valid, it checks Bank 1 code. If it is valid (see note below), it jumps to the application located in Bank 1.
  c) If both Bank 2 and Bank 1 do not contain valid code (see note below), when the protection level2 is disabled, the normal Bootloader operations are executed. Otherwise, it jumps to Bank 1 regardless of its validity. Refer to *Table 42* for more details.
- When BFB2 bit is reset (default state), the dual bank boot mechanism is not performed.

Note: The code is considered as valid when the first data (at the bank start address, which should be the stack pointer) points to a valid address (stack top address) into the internal or external SRAM memory (NOR/SRAM or SDRAM interfaced by the FMC). If the first address points to any other location (out of the SRAM memory) the code is considered not valid.

For the STM32F429xx/439xx devices, the Flash memory, system memory or SRAM is selected as the boot space, as shown in *Table 42* below.

Table 42. Boot pin and BFB2 bit configuration

Protection level2	BFB2 bit	Boot mode selection pins		Boot mode	Aliasing
		BOOT1	BOOT0		
0 or 1	0	X	0	User Flash memory	User Flash memory Bank1 is selected as the boot space.
		0	1	System memory	Boot on System memory to execute Bootloader.
		1	1	Embedded SRAM	Boot on Embedded SRAM
	1	X	0	System memory	Boot on System memory to execute dual bank boot mechanism. If Bank 2 and Bank 1are not valid, Bootloader is executed for Flash update.
		0	1	System memory	Boot on System memory to execute Bootloader.
		1	1	Embedded SRAM	Boot on Embedded SRAM.
2	0	X	0	User Flash memory	User Flash memory Bank1 is selected as the boot space.
		0	1	User Flash memory	
		1	1	User Flash memory	
	1	X	0	System memory	Boot on System memory to execute dual bank boot mechanism. If Bank 2 isn't valid, it jumps to Bank 1.
		0	1	System memory	
		1	1	System memory	

When entering System Memory there are two options: execute Bootloader (for Flash update) or execute Dual Bank Jump (see *Table 42*).

When protection level2 is enabled, bootloader is never executed for Flash update.

When the conditions a, b, and c described below are fulfilled, it is equivalent to configuring boot pins for system memory boot (BOOT0 = 1 and BOOT1 = 0). In this case when protection level2 is disabled, normal Bootloader operations are executed.

a) BFB2 bit is set

b) Both banks do not contain valid code

c) Boot pins configured as follows: BOOT0 = 0 and BOOT1 = x

When the BFB2 bit is set, and Bank 2 and/or Bank 1 contain valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code).

Consequently, if you have set the BFB2 bit (to boot from Bank 2) then, to be able to execute the Bootloader code for Flash update when protection level2 is disabled, you have to:

– set the BFB2 bit to 0, BOOT0 = 1 and BOOT1 = 0 or,

– program the content of address 0x0808 0000/0x0810 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0.

19.2 Bootloader configuration

The bootloader embedded in STM32F429xx/439xx devices support four serial peripherals: USART1, USART3, CAN2, DFU (USB FS Device) and I2C.

Table 43 shows the required hardware resources of STM32F429xx/439xx devices used by the bootloader in System memory boot mode.

Table 43. STM32F429xx/439xx configuration in System memory boot mode

Bootloader	Feature/Peripheral	State	Comment
Common to all bootloaders	RCC	HSI enabled	The system clock is equal to 24 MHz using the PLL. The HSI clock source is used at startup (interface detection phase) and when USART or I2C interfaces are selected (once CAN or DFU bootloader is selected, the clock source will be derived from external crystal).
		HSE enabled	The system clock is equal to 60 MHz. The HSE clock source is used only when the CAN or the DFU (USB FS Device) interfaces are selected. The external clock must provide a frequency multiple of 1 MHz and ranging from 4 MHz to 26 MHz.
		-	The Clock Security System (CSS) interrupt is enabled for the CAN and DFU bootloaders. Any failure (or removal) of the external clock generates system reset.
	RAM	-	8 Kbytes starting from address 0x20000000 are used by the bootloader firmware
	System memory	-	30424 bytes starting from address 0x1FFF0000, contain the bootloader firmware
	IWDG	-	The independent watchdog (IWDG) prescaler is configured to its maximum value. It is periodically refreshed to prevent watchdog reset (in case the hardware IWDG option was previously enabled by the user).
	Power	-	Voltage range is set to [1.62 V, 2.1 V]. The voltage range can be configured in run time using bootloader commands. Note that in this range internal Flash write operations are allowed only in byte format (Half-Word, Word and Double-Word operations are not allowed).
USART1 bootloader	USART1	Enabled	Once initialized the USART1 configuration is: 8 bits, even parity and 1 Stop bit
	USART1_RX pin	Input	PA10 pin: USART1 in reception mode
	USART1_TX pin	Output	PA9 pin: USART1 in transmission mode
	USART3_RX (PB11), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		

Table 43. STM32F429xx/439xx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
USART3 bootloader (on PB10/PB11)	USART3	Enabled	Once initialized the USART3 configuration is: 8 bits, even parity and 1 Stop bit
	USART3_RX pin	Input	PB11 pin: USART3 in reception mode
	USART3_TX pin	Output	PB10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART3 bootloader (on PC10/PC11)	USART3	Enabled	Once initialized the USART3 configuration is: 8 bits, even parity and 1 Stop bit
	USART3_RX pin	Input	PC11 pin: USART3 in reception mode
	USART3_TX pin	Output	PC10pin: USART3 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
USART1 and USART3 bootloaders	SysTick timer	Enabled	Used to automatically detect the serial baud rate from the host for USARTx bootloaders.
CAN2 bootloader	CAN2	Enabled	Once initialized the CAN2 configuration is: Baudrate 125 kbps, 11-bit identifier. *Note: CAN1 is clocked during CAN2 bootloader execution because STM32F429xx/439xx CAN1 manages the communication between CAN2 and SRAM.*
	CAN2_RX pin	Input	PB05 pin: CAN2 in reception mode
	CAN2_TX pin	Output	PB13pin: CAN2 in transmission mode
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		
I2C1 bootloader	I2C1	Enabled	I2C clock source is APB1.Once initialized, the I2C1 configuration is: I2C speed: 400 KHz, 7-bit address, slave mode, slave address: 0x6E, analog filter ON.
	I2C1_SCL pin	Input/output	PB6 pin: I2C1 clock line is used in open-drain mode.
	I2C1_SDA pin	Input/output	PB9 pin: I2C1 data line is used in open-drain mode.
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins must be kept at a high or low level during the detection phase.		

Table 43. STM32F429xx/439xx configuration in System memory boot mode (continued)

Bootloader	Feature/Peripheral	State	Comment
DFU bootloader	USB_OTG_FS	Enabled	USB OTG FS configured in Forced Device mode. USB_OTG_FS interrupt vector is enabled and used for USB DFU communications.
	USB_OTG_FS_DM pin	Input	PA11 pin: USB OTG FS DM line
	USB_OTG_FS_DP pin	Output	PA12pin: USB OTG FS DP line
	USART1_RX (PA10), USART3_RX (PB11), USART3_RX (PC11) and CAN2_RX (PB05) pins must be kept at a high or low level during the detection phase.		
	TIM11	Enabled	This timer is used to determine the value of the external clock frequency. Once the external clock frequency is determined, the RCC system is configured to operate at 60 MHz system clock (using PLL).

For the DFU interface, the external clock source (HSE) is required for USB operations. The detection of the HSE value is done by the bootloader firmware and is based on the internal oscillator clock (HSI). Thus, when due to temperature or other conditions the internal oscillator precision is altered above the tolerance band (1 % around theoretical value), the Bootloader might calculate a wrong HSE frequency value. In this case, the bootloader of DFU or CAN interface might dysfunction or might not work at all.

The system clock is derived from the embedded internal high-speed RC for USARTx. No external quartz is required in this case for the bootloader code. This internal clock is also used for CAN and DFU (USB FS Device), but only for the selection phase. An external clock multiple of 1 MHz (between 4 and 26 MHz) is required for CAN and DFU bootloader execution after the selection phase.

The CAN and DFU bootloaders implement an external clock detection mechanism allowing to determine the value of the external clock using the internal high-speed RC and TIM11 timer. The accuracy of this mechanism allows to detect only frequencies multiple of 1 MHz and ranging from 4 to 26 MHz. Any other value is not supported and will result in unexpected behavior of the bootloader.

After downloading the application binary, if you choose to execute the Go command, the peripheral registers used by the bootloader (shown in *Table 43*) are not initialized to their default reset values before jumping to the user application. They should be reconfigured in the user application if they are used. So, if the IWDG is being used in the application, the IWDG prescaler value has to be adapted to meet the requirements of the application (since the prescaler has been previously set to its maximum value by the bootloader).

19.3 Bootloader hardware requirements

The hardware required to put the STM32F429xx/439xx into System memory boot mode consists of any circuitry, switch or jumper, capable of holding the BOOT0 pin high and the BOOT1 pin low during reset.

To connect to the STM32F429xx/439xx during System memory boot mode, the following conditions have to be verified:

- The RX pins of the peripheral unused in this bootloader have to be kept at a known (low or high) level, and should not be left floating during the detection phase as described below:
  - If USART1 is used to connect to the bootloader:
    the USART3_RX (PC11 and PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PB10/PB11) is used to connect to the bootloader:
    the USART1_RX (PA10), USART3_RX (PC11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If USART3 (on PC10/PC11) is used to connect to the bootloader:
    the USART1_RX (PA10), USART3_RX pin (PB11), CAN2_RX (PB05), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) have to be kept at a high or low level and must not be left floating during the detection phase.
  - If CAN2 is used to connect to the bootloader:
    the USART1_RX (PA10), USART3_RX (PC11 and PB11), OTG_FS_DM (PA11) and OTG_FS_DP (PA12) pins have to be kept at a high or low level and must not be left floating during the detection phase.
  - If DFU (USB FS Device) is used to connect to the bootloader:
    the USART1_RX (PA10), USART3_RX (PC11 and PB11) and CAN2_RX (PB05) pins have to be kept at a high or low level and must not be left floating during the detection phase

Note: The I2C1 interface pins I2C1_SCL (PB6) and I2C1_SDA (PB9) should be left floating during the detection phase of any interface.

- When the BFB2 bit is set, and Bank 2 and/or Bank 1 contain a valid user application code, the Dual Bank Boot is always performed (bootloader always jumps to the user code and never continues normal operations). Consequently, if you have set the BFB2 bit (to boot from Bank 2), then to be able to execute the bootloader code when protection level2 is disabled, you have to:
  - set the BFB2 bit to 0, BOOT0 = 1 and BOOT1 = 0 or,
  - program the content of address 0x0808 0000/0x0810 0000 (base address of Bank2) and 0x0800 0000 (base address of Bank1) to 0x0.
- Connection to the peripheral to be performed through:
  - An RS232 serial interface (example, ST3232 RS232 transceiver) has to be directly connected to the USART1_RX (PA10) and USART1_TX (PA9) pins when

USART1 is used, or to the USART2_RX (PD6) and USART2_TX (PD5) pins when USART2 is used

– A CAN interface (CAN transceiver) has to be directly connected to the CAN2_RX (PB5) and CAN2_TX (PB13) pins.
– A certified USB cable has to be connected to the microcontroller (optionally an ESD protection circuitry can be used).
– If the I2C1 is used to connect to the bootloader: an I2C interface (I2C Master) has to be directly connected to the I2C1_SCL (PB6) and I2C1_SDA (PB9) pins.

The USART1_CK, USART1_CTS and USART1_RTS pins are not used, therefore the application can use these pins for other peripherals or GPIOs. The same is applicable for USART3.

An 1.8 Kohm pull-up resistor has to be connected to both SDA and SCL lines.

The user can control the BOOT0 and Reset pins from a PC serial applet using the RS232 serial interface which controls BOOT0 through the DSR line and Reset through the CTS line. The user must use a full null modem cable. The necessary hardware to implement for this control exists in the STM324x9I-EVAL board. For more details about this, refer to the STM324x9I-EVAL board user manual, available from STMicroelectronics website: *http://www.st.com*.

19.4 Bootloader selection

The STM32F429xx/439xx embedded bootloader supports six peripheral interfaces: USART1, USART3 (on PB10/PB11 and PC10/PC11), CAN2, DFU (USB FS Device) and I2C1. Any one of these peripheral interfaces can be used to communicate with the bootloader and download the application code to the internal Flash.

The embedded bootloader firmware is able to auto-detect the peripheral interface to be used. In an infinite loop, it detects any communication on the supported bootloader interfaces.

Note: The RX pins of the peripherals not used in this bootloader must be kept at a known (low or high) level and should not be left floating during the detection phase as described below. Refer to Section 18.2: Bootloader hardware requirements for more information.

To use the USART bootloader on USART1 or USART3, connect the serial cable to the desired interface. Once the bootloader detects the data byte 0x7F on this interface, the bootloader firmware executes the auto-baud rate sequence and then enters a loop, waiting for any USART bootloader command.

To use the CAN2 interface, connect the CAN cable to CAN2. Once the bootloader detects a frame on the CAN2_RX pin (PB5), the bootloader firmware enters a CAN loop and starts to determine the external clock frequency value. The supported HSE frequencies are multiple of 1 MHz ranging from 4 to 26 MHz. Any other values lead to an unexpected behavior, CAN bootloader firmware enters an infinite loop and waits until it receives a message. If the external clock is not present, a system reset is generated.

If a USB cable is plugged into the microcontroller's USB interface at any time during the bootloader firmware selection sequence, the bootloader enters the DFU bootloader loop waiting for any DFU bootloader command.

To use the USART or the CAN bootloader, it is mandatory that no USB Host is connected to the USB peripheral during the selection phase. Once the USART or CAN bootloader is

selected, the user can plug a USB cable without impacting the selected bootloader execution except commands which generate a system reset.

To use the I2C bootloader on the I2C1 interface, connect the host master and the STM32 I2C1 slave target. The master and the STM32 target are connected together via the data (SDA) and clock (SCL) pins. Once the STM32 I2C interface recognizes its own address (0x70) on the bus, the bootloader firmware enters an infinite loop and waits until it receives an I2C bootloader command.

Once one interface is selected for the bootloader, all other interfaces are disabled.

Figure 17 and *Figure 18* show the bootloader selection mechanism. More details are provided in the sections dedicated to each peripheral bootloader.

Figure 17. Dual Bank Boot Implementation for STM32F429xx/439xx

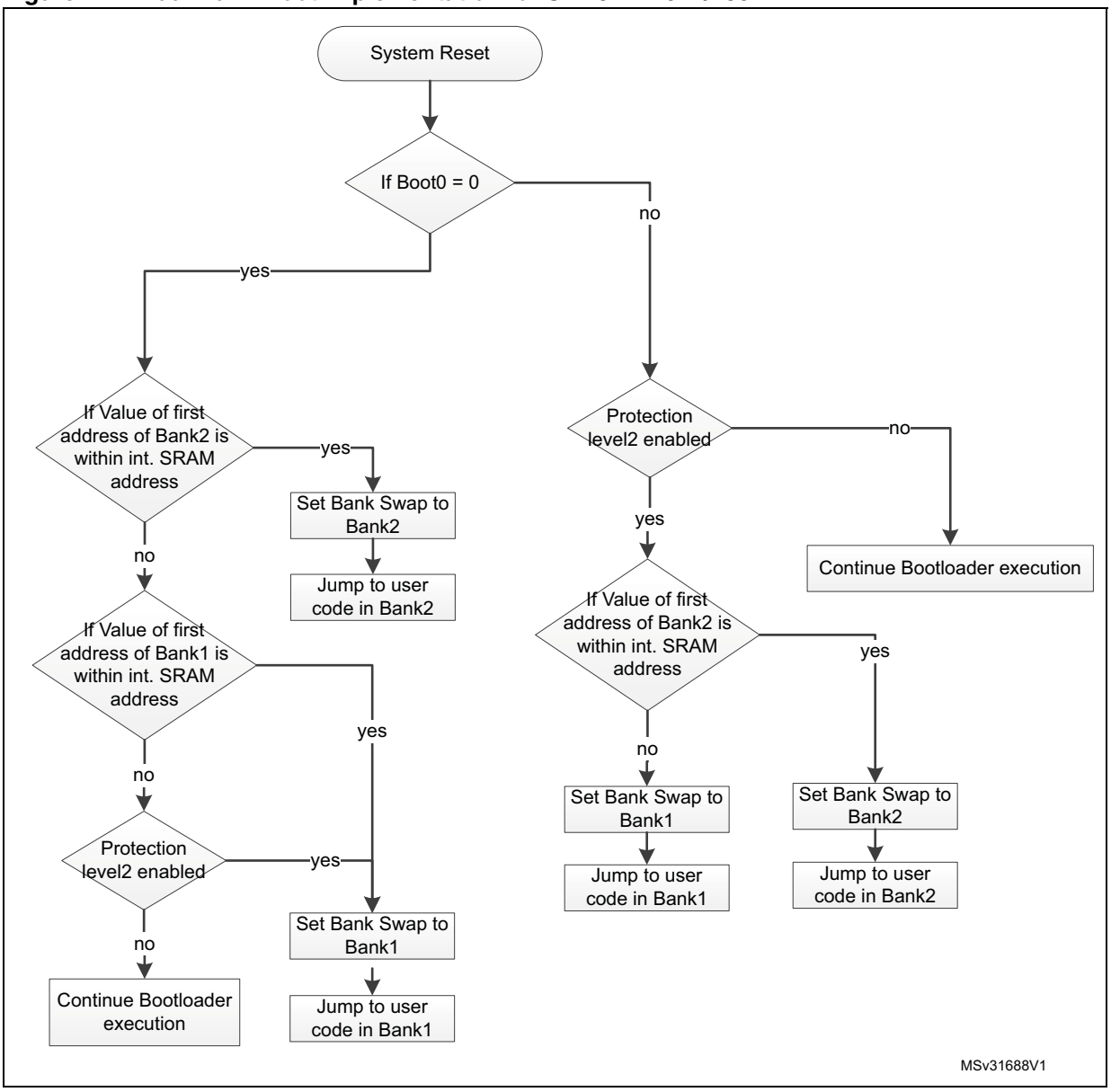
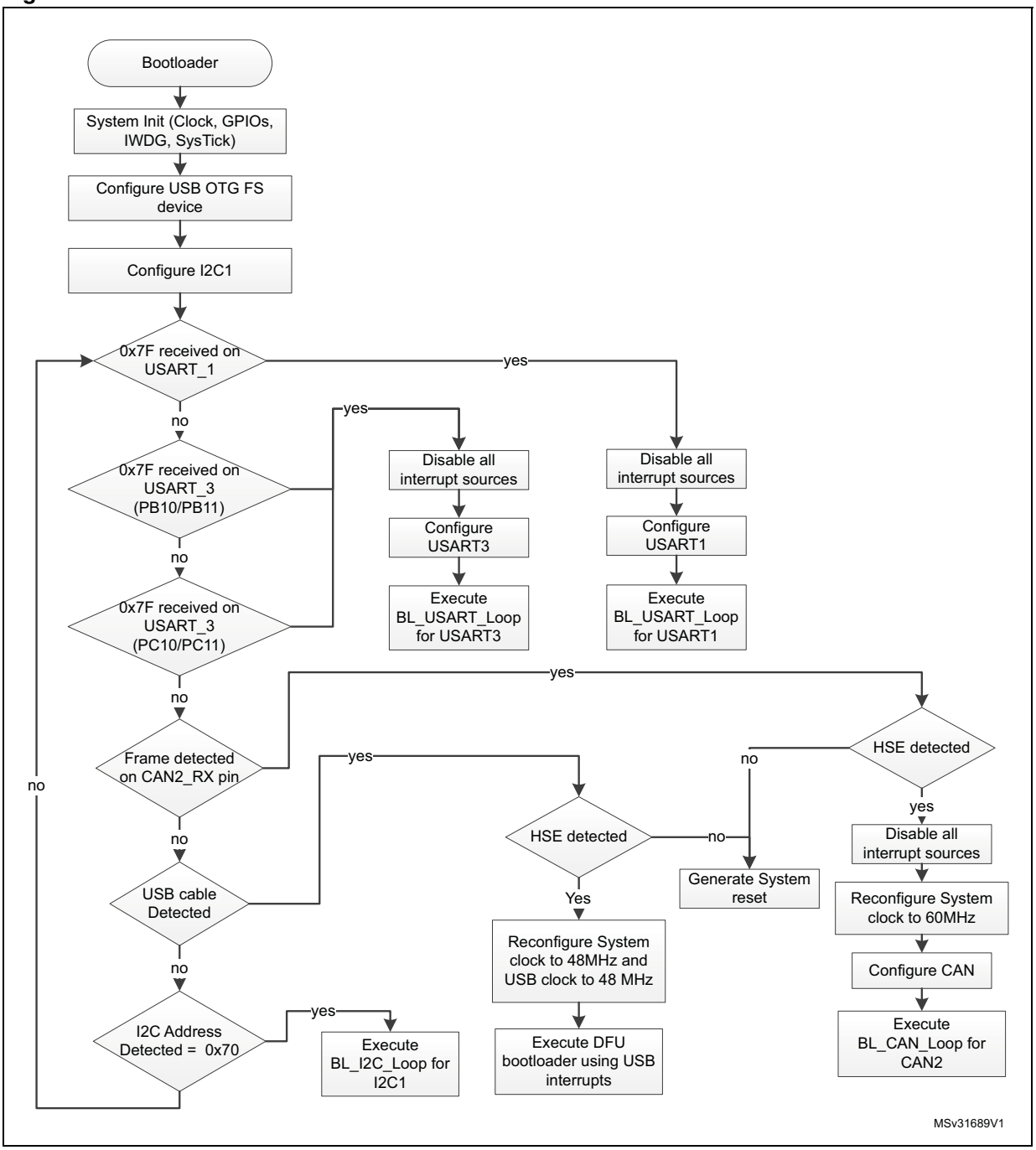


Figure 18. Bootloader selection for STM32F429xx/439xx



19.5 Important considerations

STM32F429xx/439xx bootloader has some specific features that should be taken into consideration:

- In addition to standard memories (internal Flash, internal SRAM, option bytes and System memory), STM32F429xx/439xx device bootloader firmware supports OTP memory (528 bytes from 0x1FFF7800 to 0x1FFF7A0F).
- **OTP memory** can be read and written, but cannot be erased using the Erase command. When writing in an OTP memory location, make sure that the relative protection bit (in the last 16 bytes of the OTP memory) is not reset.
- **Option bytes**

  The address is 0x1FFFC000 and 0x1FFEC000. Three levels of protection are allowed:
  - Level 0
  - Level 1
  - Level 2

  Refer to RM0216 reference manual for more details about protection levels.
- **Read protect** command corresponds to Level 1 protection.
- **Read unprotect** command corresponds to Level 0 protection.
- **Mass erase** command on STM32F429xx/439xx takes longer than on other STM32 devices due to their memory density. Make sure that the timeout used by your host interface to wait for an acknowledge event after sending a Mass erase command is sufficient.
- **Voltage Range configuration**

  The Voltage Range can be updated on the fly by the bootloader software. The Voltage Range is set to its default value at each bootloader software startup (after system reset or jump to the bootloader code). The bootloader software allows modifying this parameter through a virtual memory location. This memory location is not physical but can be read and written using usual bootloader read/write operations according to the protocol in use (USART, CAN or DFU). This memory location contains 4 bytes which are described in *Table 40*. It can be accessed by 1, 2, 3 or 4 bytes. However, reserved bytes should remain at their default values (0xFF), otherwise the request will be NACKed.

Table 44. STM32F429xx/439xx Voltage Range configuration using bootloader

Address	Size	Description
0xFFFF0000	1 byte	This byte controls the current value of Voltage Range: 0x00: Voltage Range [1.62V, 2.1V] 0x01: Voltage Range [2.1V, 2.4V] 0x02: Voltage Range [2.4V, 2.7V] 0x03: Voltage Range [2.7V, 3.6V] 0x04: Voltage Range [2.7V, 3.6V] and Double Word write/erase operation is used. In this case it is mandatory to supply 9 V through VPP pin (refer to PM0081 for more details about Double-Word write procedure). Other: All other values are not supported and will be NACKed.
0xFFFF0001	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0002	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.
0xFFFF0003	1 byte	Reserved. 0xFF: Default value. Other: all other values are not supported and will be NACKed.

If the application is loaded into Flash memory at an address different from 0x08000000, then the vector table has to be relocated to start from the address where the application is loaded.

19.6 Bootloader version

Table 45 shows the STM32F429xx/439xx bootloader version.

Table 45. STM32F429xx/439xx bootloader version

Bootloader version number	Description	Known limitations
V7.0	Initial bootloader version.	

20 Device-dependent bootloader parameters

The bootloader protocol's command set and sequences for each serial peripheral (USART, CAN, USB and I^2C) are the same for all STM32 devices. Some parameters, however, are device-dependent. For a few commands, the value of some parameters may depend on the device used. These parameters are listed below:

- PID (product ID), which changes with the device
- Valid memory addresses (RAM, Flash memory, System memory, option byte area) accepted by the bootloader when the Read Memory, Go and Write Memory commands are accepted.
- Size of the Flash memory sector used when executing the Write Protect command.

The table below shows the values of these parameters for each STM32 device bootloader in production.

Table 46. Bootloader device-dependent parameters

STM32 family	Device	Product (device) ID	RAM memory	Flash memory	Flash sector size	Option byte area	System memory
F1	Low-density	0x412	0x20000200 up to 0x20002800	0x08000000 up to 0x08008000	4 KB (4 pages of 1 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFF000 - 0x1FFFF800
	Medium-density	0x410	0x20000200 up to 0x20005000	0x08000000 up to 0x08020000	4 KB (4 pages of 1 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFF000 - 0x1FFFF800
	High-density	0x414	0x20000200 up to 0x20010000	0x08000000 up to 0x08080000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFF000 - 0x1FFFF800
	Connectivity line	0x418	0x20001000 up to 0x20010000	0x08000000 up to 0x08040000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFB000 - 0x1FFFF800
	Medium-density value line	0x420	0x20000200 up to 0x20002000	0x08000000 up to 0x08020000	4 KB (4 pages of 1 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFF000 - 0x1FFFF800
	High-density value line	0x428	0x20000200 up to 0x20008000	0x08000000 up to 0x08080000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFF000 - 0x1FFFF800
	XL-density	0x430	0x20000800 up to 0x20018000	0x08000000 up to 0x08100000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFE000 - 0x1FFFF800

Table 46. Bootloader device-dependent parameters (continued)

STM32 family	Device	Product (device) ID	RAM memory	Flash memory	Flash sector size	Option byte area	System memory
L1	Medium-density ultralow power	0x416	0x20000800 up to 0x20004000	0x08000000 up to 0x08020000	4 KB (16 pages of 256 bytes each)	0x1FF80000 - 0x1FF80010	0x1FF00000 - 0x1FF01000
	High-density ultralow power	0x436	0x20001000 up to 0x2000C000	0x08000000 up to 0x08060000	4 KB (16 pages of 256 bytes each)	0x1FF80000- 0x1FF80020	0x1FF00000- 0x1FF02000
	Medium-density plus ultralow power	0x427	0x20001000 up to 0x20008000	0x08000000 up to 0x08040000	4 KB (16 pages of 256 bytes each)	0x1FF80000 - 0x1FF80020	0x1FF00000 - 0x1FF02000
F2	STM32F2xxxx	0x411	0x20002000 up to 0x20020000	0x08000000 up to 0x08100000	12 sectors (4x16 KB, 1x64 KB, 7x128 KB)	0x1FFFC000- 0x1FFFC00F	0x1FFF0000 - 0x1FFF77FF
F0	STM32F051xx	0x440	0x20000800 up to 0x20002000	0x08000000 up to 0x08010000	4 KB (4 pages of 1 KB each)	0x1FFFF800 - 0x1FFFF80B	0x1FFFEC00 - 0x1FFFF800
	STM32F050xx	0x444	0x20000800 up to 0x20001000	0x08000000 up to 0x08008000	4 KB (2 pages of 1 KB each)	0x1FFFF800- 0x1FFFF80B	0x1FFFEC00 - 0x1FFFF800
F4	STM32F40xxx/ 41xxx	0x413	0x20002000 up to 0x20020000	0x08000000 up to 0x08100000	12 sectors (4x16 KB, 1x64 KB, 7x128 KB)	0x1FFFC000- 0x1FFFC00F	0x1FFF0000 - 0x1FFF77FF
	STM32F427xx/ 437xx	0x419	0x20002000 up to 0x20030000	0x08000000 up to 0x08200000	24 sectors (8x16 KB, 2x64 KB, 14x128 KB)	0x1FFFC000- 0x1FFFC00F 0x1FFEC000- 0x1FFEC00F	0x1FFF0000 - 0x1FFF77FF
	STM32F429xx/ 439xx	0x419	0x20002000 up to 0x20030000	0x08000000 up to 0x08200000	24 sectors (8x16 KB, 2x64 KB, 14x128 KB)	0x1FFFC000- 0x1FFFC00F 0x1FFEC000- 0x1FFEC00F	0x1FFF0000 - 0x1FFF77FF

Table 46. Bootloader device-dependent parameters (continued)

STM32 family	Device	Product (device) ID	RAM memory	Flash memory	Flash sector size	Option byte area	System memory
F3	STM32F37xxx	0x432	0x20001400 up to 0x20008000	0x08000000 up to 0x08040000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFD800 - 0x1FFFF7FF
	STM32F30xxx	0x422	0x20001400 up to 0x2000A000	0x08000000 up to 0x08040000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFD800 - 0x1FFFF7FF
	STM32F38xxx	0x432	0x20001000 up to 0x20008000	0x08000000 up to 0x08040000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFD800 - 0x1FFFF7FF
	STM32F31xxx	0x422	0x20001400 up to 0x2000A000	0x08000000 up to 0x08040000	4 KB (2 pages of 2 KB each)	0x1FFFF800 - 0x1FFFF80F	0x1FFFD800 - 0x1FFFF7FF

21 Bootloader timing characteristics

This section presents the main startup timings of the bootloader firmware depending on products. They can be used to set up the connection timeout, that is how long the host waits before synchronization with the bootloader is established.

Three types of timings will be described herein:

- Hardware-dependent timings relative to product and directly extracted from the product datasheet.
- Communication-dependent timings relative to the baudrate and data traffic on the bus. These timings depend only on the communication interface configuration and on the host behavior.
- Bootloader software-dependent timings relative to bootloader software operations.

All the timings described in his section are expressed in milliseconds (ms) except when otherwise specified.

21.1 USART bootloader timing characteristics

Two main timings need to be considered for host operations when using the USART bootloader:

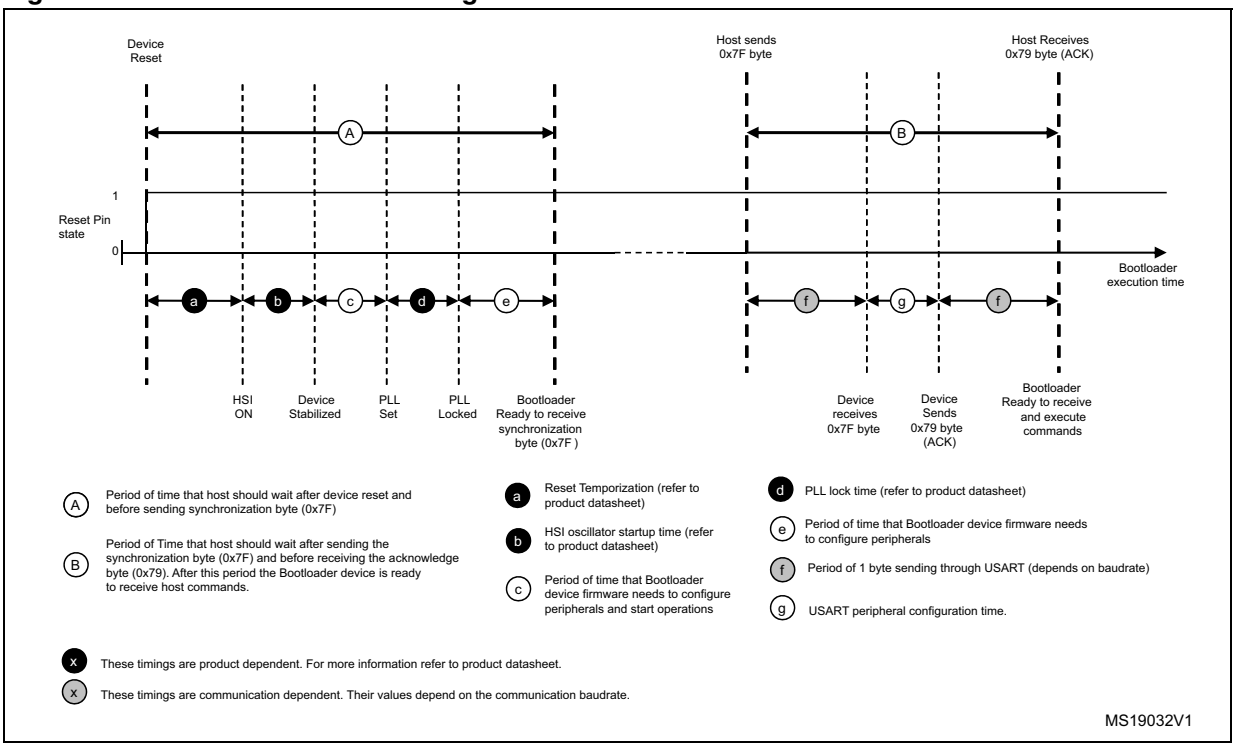
- Timing A

  After bootloader reset, this timing corresponds to the time during which the host waits before sending the synchronization data (0x7F) to properly configure the bootloader baudrate detection. This timing will be referred to as **A** throughout this section.
- Timing B

  After sending the synchronization data (0x7F), this timing corresponds to the time during which the host waits before receiving the first acknowledge response (meaning that the bootloader is ready to receive and execute host commands). This timing will be referred to as **B** throughout this section.

A and **B** timings are composed of different sub-timings as described in *Figure 19*.

Figure 19. USART bootloader timing waveforms



The timing values for each product are listed in *Table 47*, *Table 48*, *Table 49*, *Table 50*, *Table 51*, *Table 52*, *Table 53*, *Table 54*, and *Table 55*.

Table 47. USART bootloader timings for low/medium/high-density and value line devices

Time	Description	Min	Max	Unit
a	Reset temporization	1	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.004	-	ms
d	PLL Lock time	0.2	-	ms
e	Bootloader firmware operations	0.002	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.002	-	ms
A	Time = a + b + c + d + e	1.207	4.708	ms
B	Time = (2 x f) + g	0.15825	15.002	ms

Table 48. USART bootloader timings for XL-density line devices

Time	Description	Min	Max	Unit
a	Reset temporization	1	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.02	-	ms
d	PLL Lock time	0.2	-	ms
e	Bootloader firmware operations	0.006	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.006	-	ms
A	Time = a + b + c + d + e	1.227	4.728	ms
B	Time = (2 x f) + g	0.16225	15.006	ms

Table 49. USART bootloader timings for connectivity line devices (PA9 pin low)

Time	Description	Min	Max	Unit
a	Reset temporization	1	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.025	-	ms
d	PLL Lock time	0.35	-	ms
e	Bootloader firmware operations	0.02	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.007	-	ms
A	Time = a + b + c + d + e	1.396	4.897	ms
B	Time = (2 x f) + g	0.16325	15.007	ms

For connectivity line devices, PA9 pin (USB_VBUS) is used to detect the USB host connection. The initialization of USB peripheral is performed only if PA9 is high at detection phase which means that a host is connected to the port and delivering 5 V on the USB bus. When PA9 level is high at detection phase, more time is required to initialize and shutdown the USB peripheral.

To minimize bootloader detection time for connectivity line devices when PA9 pin is not used, keep PA9 state low during detection phase from the moment the device is reset till a device ACK is sent.

Table 50. USART bootloader timings for connectivity line devices (PA9 high)

Time	Description	Min	Max	Unit
a	Reset temporization	1	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.025	-	ms
d	PLL Lock time	0.35	-	ms
e	Bootloader firmware operations	523	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	105	-	ms
A	Time = a + b + c + d + e	524.376	527.877	ms
B	Time = (2 x f) + g	105.1563	120	ms

Table 51. USART bootloader timings for medium-density ultralow power devices

Time	Description	Min	Max	Unit
a	Reset temporization	0.4	1.6	ms
b	MSI oscillator stabilization time	-	0.04	ms
c	Bootloader firmware operations	0.064	-	ms
d	HSI oscillator startup time	0.0037	0.006	ms
e	Bootloader firmware operations	0.034	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.008	-	ms
A	Time = a + b + c + d + e	0.5417	1.744	ms
B	Time = (2 x f) + g	0.16425	15.008	ms

Table 52. USART bootloader timings for high-density ultralow power devices

Time	Description	Min	Max	Unit
a	Reset Temporization	0.4	1.6	ms
b	MSI oscillator stabilization time		0.07	ms
c	Bootloader firmware operations	0.058		ms
d	HSI oscillator startup time		0.006	ms
e	Bootloader firmware operations	0.174		ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0078		ms
A	Time = a + b + c + d + e	0.708	1.908	ms
B	Time = (2 x f) + g	0.16405	15.0078	ms

Table 53. USART bootloader timings for STM32F2xxxx devices

Time	Description	Min	Max	Unit
a	Reset temporization	0.5	3.0	ms
b	HSI oscillator startup time	0.0022	0.004	ms
c	Bootloader firmware operations	0.01	-	ms
d	PLL Lock time	0.075	0.2	ms
e	Bootloader firmware operations	84	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.009	-	ms
A	Time = a + b + c + d + e	84.5872	87.214	ms
B	Time = (2 x f) + g	0.16525	15.009	ms

Table 54. USART bootloader timings for STM32F40xxx/41xxx devices

Time	Description	Min	Max	Unit
a	Reset temporization	0.5	3.0	ms
b	HSI oscillator startup time	0.0022	0.004	ms
c	Bootloader firmware operations	0.01	-	ms
d	PLL Lock time	0.075	0.2	ms
e	Bootloader firmware operations	84	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.009	-	ms
A	Time = a + b + c + d + e	84.5872	87.214	ms
B	Time = (2 x f) + g	0.16525	15.009	ms

Table 55. USART bootloader timings for STM32F051xx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c + d + e	Bootloader firmware operations	0.111		ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0095		ms
A	Time = a + b + c + d + e	1.612	4.613	ms
B	Time = (2 x f) + g	0.16575	15.0095	ms

Table 56. USART bootloader timings for medium-density plus devices

Time	Description	Min	Max	Unit
a	Reset Temporization	0.4	1.6	ms
b	MSI oscillator stabilization time	0.07	0.07	ms
c	Bootloader firmware operations	0.0558	-	ms
d	HSI oscillator startup time	0.0037	0.006	ms
e	Bootloader firmware operations	0.157	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0078	-	ms
A	Time = a + b + c + d + e	0.6865	1.8888	ms
B	Time = (2 x f) + g	0.16405	15.0078	ms

Table 57. USART bootloader timings for STM32F050xx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c + d + e	Bootloader firmware operations	0.09	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0064	-	ms
A	Time = a + b + c + d + e	1.591	4.592	ms
B	Time = (2 x f) + g	0.16265	15.0064	ms

Table 58. USART bootloader timings for STM32F37xxx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.015	-	ms
d	PLL Lock time	0.2	0.2	ms
e	Bootloader firmware operations	43.2	-	ms
c + d(1)	Bootloader firmware operations	-	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.00245	-	ms
A	Time = a + b + c + d + e	44.916	47.917	ms
B	Time = (2 x f) + g	0.1587	15.0025	ms

1. This Timing replace «d» and «e» timing when external clock (HSE) is not present.

Table 59. USART bootloader timings for STM32F30xxx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c	Bootloader firmware operations	0.6	-	ms
d	PLL Lock time	0.2	0.2	ms
e	Bootloader firmware operations	41	-	ms
c + d(1)	Bootloader firmware operations	-	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0022		ms
A	Time = a + b + c + d + e	43.301	46.302	ms
B	Time = (2 x f) + g	0.15845	15.0022	ms

1. This Timing replace «d» and «e» timing when external clock (HSE) is not present.

Table 60. USART bootloader timings for STM32F38xxx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c + d + e	Bootloader firmware operations	0.041		ms
f	One USART byte sending period	0.15625	7.5	ms
g	Bootloader firmware operations	0.0099		ms
A	Time = a + b + c + d + e	1.542	4.543	ms
B	Time = (2 x f) + g	0.3224	15.0099	ms

Table 61. USART bootloader timings for STM32F31xxx devices

Time	Description	Min	Max	Unit
a	Reset Temporization	1.5	4.5	ms
b	HSI oscillator startup time	0.001	0.002	ms
c + d + e	Bootloader firmware operations	0.0401		ms
f	One USART byte sending period	0.15625	7.5	ms
g	Bootloader firmware operations	0.0095		ms
A	Time = a + b + c + d + e	1.5411	4.5421	ms
B	Time = (2 x f) + g	0.322	15.0095	ms

Table 62. USART bootloader timings for STM32F427xx and STM32F437xx devices

Time	Description	Min	Max	Unit
a	Reset temporization	0.5	3.0	ms
b	HSI oscillator startup time	0.0022	0.004	ms
c	Bootloader firmware operations	0.0089	-	ms
d	PLL Lock time	0.075	0.2	ms
e	Bootloader firmware operations	81.047	-	ms
f	One USART byte sending period	0.078125	7.5	ms
g	Bootloader firmware operations	0.0058	-	ms
A	Time = a + b + c + d + e	81.6331	84.2599	ms
B	Time = (2 x f) + g	0.16205	15.0058	ms

Table 63. USART bootloader timings for STM32F429xx and STM32F439xx devices

Time	Description	Min	Max	Unit
a	Reset temporization	TBD	TBD	ms
b	HSI oscillator startup time	TBD	TBD	ms
c	Bootloader firmware operations	TBD	-	ms
d	PLL Lock time	TBD	TBD	ms
e	Bootloader firmware operations	TBD	-	ms
f	One USART byte sending period	TBD	TBD	ms
g	Bootloader firmware operations	TBD	-	ms
A	Time = a + b + c + d + e	TBD	TBD	ms
B	Time = (2 x f) + g	TBD	TBD	ms

21.2 USB bootloader timing characteristics

The main timings that need to be considered for host operations when using the USB bootloader are the following:

- Timing A

  After bootloader reset, this timing corresponds to the time during which the host waits before starting the connection sequence with the device. It is similar to the USART connection timeout described in *Section 21.1*. It will be referred to as **A** throughout this section.

- Timing B

  When the connection sequence has started, this timing corresponds to the time required by the device to establish a correct connection with the host (meaning that the

bootloader is ready to receive and execute host commands). This timing includes enumerations and DFU components configuration (e.g. internal Flash memory). This timing will be referred to as **B** throughout this section.

For connectivity line devices, if the external HSE crystal frequency is different from 25 MHz (14.7456 MHz or 8 MHz), the device performs several unsuccessful enumerations (with connect – disconnect sequences) before being able to establish a correct connection with the host. This is due to the HSE automatic detection mechanism based on SOF detection.

A and **B** timings are composed of different sub-timings as described in *Figure 20*. Refer to *Table 47*, *Table 48*, *Table 49*, *Table 50*, *Table 51*, *Table 52*, *Table 53*, *Table 54*, and *Table 55* for the values of timing A (identical to USART bootloader), and to *Table 64*, *Table 65*, *Table 66*, and *Table 67* for the values of timing B.

Note: For USB interface, only minimum timings are provided since the connection timing depends on environment and host configuration (number of nodes (hubs), host speed, traffic on the USB bus, host loading …).

Figure 20. USB bootloader timing waveforms

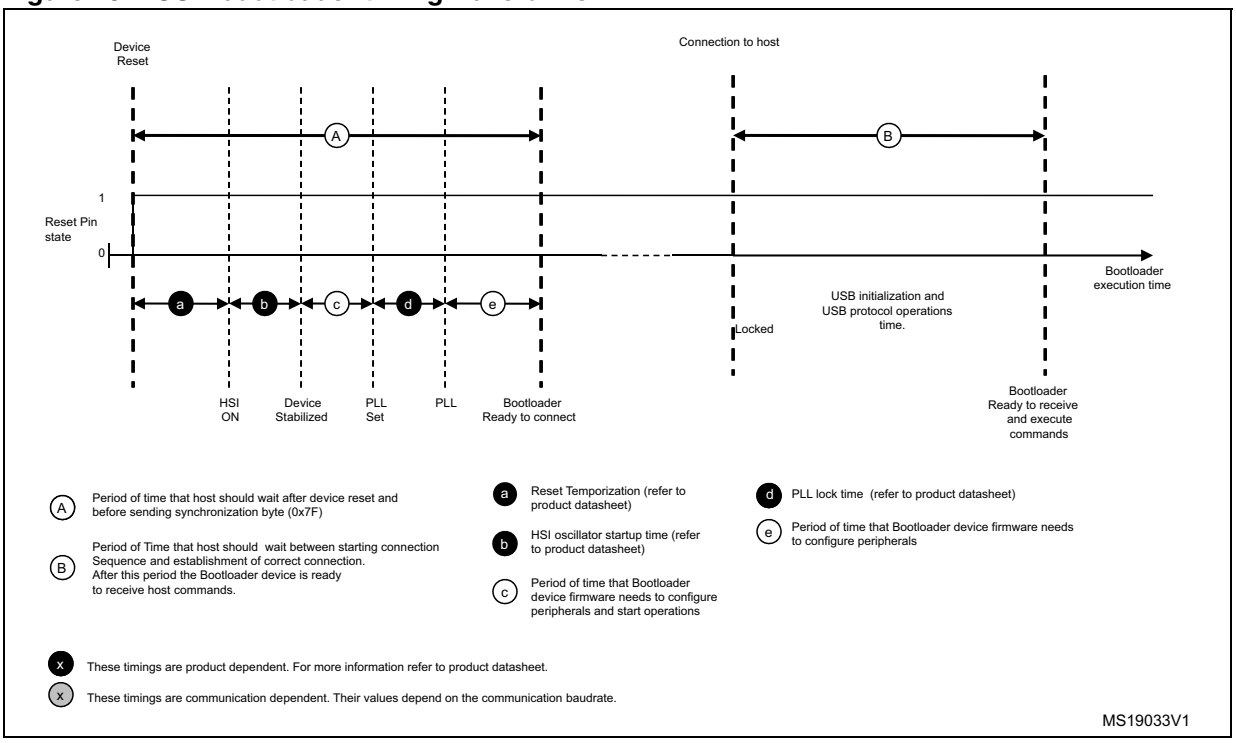


Table 64. USB minimum timings for connectivity line devices

Time	Description	25MHz	14.7456MHz	8MHz	Unit
a	Reset temporization	1	1	1	ms
b	HSI oscillator startup time	0.001	0.001	0.001	ms
c	Bootloader firmware operations	0.025	0.025	0.025	ms
d	PLL Lock time	0.35	0.35	0.35	ms
e	Bootloader firmware operations	523	523	523	ms
A	Time = a + b + c + d + e	524.376	524.376	524.376	ms
B	Connection establishment	460	4500	13700	ms

Table 65. USB minimum timings for high-density ultralow power devices

Time	Description	Min	Unit
a	Reset Temporization	0.4	ms
b	MSI oscillator stabilization time	0.07	ms
c	Bootloader firmware operations	0.058	ms
d	HSI oscillator startup time	0.0037	ms
e	Bootloader firmware operations	0.174	ms
A	Time = a + b + c + d + e	0.7057	ms
B	Connection establishment	849	ms

Table 66. USB minimum timings for STM32F2xxxx devices

Time	Description	Min	Unit
a	Reset temporization	0.5	ms
b	HSI oscillator startup time	0.0022	ms
c	Bootloader firmware operations	0.01	ms
d	PLL Lock time	0.075	ms
e	Bootloader firmware operations	84	ms
A	Time = a + b + c + d + e	84.5872	ms
B	Connection establishment	54	ms

Table 67. USB minimum timings for STM32F40xxx/41xxx devices

Time	Description	Min	Unit
a	Reset temporization	0.5	ms
b	HSI oscillator startup time	0.0022	ms
c	Bootloader firmware operations	0.01	ms
d	PLL Lock time	0.075	ms
e	Bootloader firmware operations	84	ms
A	Time = a + b + c + d + e	84.5872	ms
B	Connection establishment	54	ms

Table 68. USB minimum timings for medium-density plus devices

Time	Description	Min	Unit
a	Reset temporization	0.4	ms
b	MSI oscillator stabilization time	0.07	ms
c	Bootloader firmware operations	0.0558	ms
d	HSI oscillator startup time	0.0037	ms
e	Bootloader firmware operations	0.157	ms
A	Time = a + b + c + d + e	0.6865	ms
B	Connection establishment	849	ms

Table 69. USB minimum timings for STM32F37xxx devices

Time	Description	Min	Unit
a	Reset temporization	1.5	ms
b	HSI oscillator startup time	0.001	ms
c	Bootloader firmware operations	0.015	ms
d	PLL Lock time	0.2	ms
c + d(1)	Bootloader firmware operations (if HSE not present)		
e	Bootloader firmware operations	43.2	ms
A	Time = a + b + c + d + e	44.916	ms
B	Connection establishment	560	ms

Table 70. USB minimum timings for STM32F30xxx devices

Time	Description	Min	Unit
a	Reset temporization	1.5	ms
b	HSI oscillator startup time	0.001	ms
c	Bootloader firmware operations	0.6	ms
d	PLL Lock time	0.2	ms
e	Bootloader firmware operations	41	ms
A	Time = a + b + c + d + e	43.301	ms
B	Connection establishment	620	ms

Table 71. USB minimum timings for STM32F427xx/437xx devices

Time	Description	Min	Unit
a	Reset temporization	0.5	ms
b	HSI oscillator startup time	0.0022	ms
c	Bootloader firmware operations	0.0089	ms
d	PLL Lock time	0.075	ms
e	Bootloader firmware operations	81.047	ms
A	Time = a + b + c + d + e	81.6331	ms
B	Connection establishment	773.13	ms

Table 72. USB minimum timings for STM32F429xx/439xx devices

Time	Description	Min	Unit
a	Reset temporization	TBD	ms
b	HSI oscillator startup time	TBD	ms
c	Bootloader firmware operations	TBD	ms
d	PLL Lock time	TBD	ms
e	Bootloader firmware operations	TBD	ms
A	Time = a + b + c + d + e	TBD	ms
B	Connection establishment	TBD	ms

For the STM32F2xxxx, STM32F40xxx/41xxx, STM32F427xx/437xx and STM32F429xx/439xx devices bootloader, the timing values are independent from the HSE crystal frequency. The detection of the HSE crystal frequency value is performed through period measurement using TIM11 timer and HSI internal oscillator.

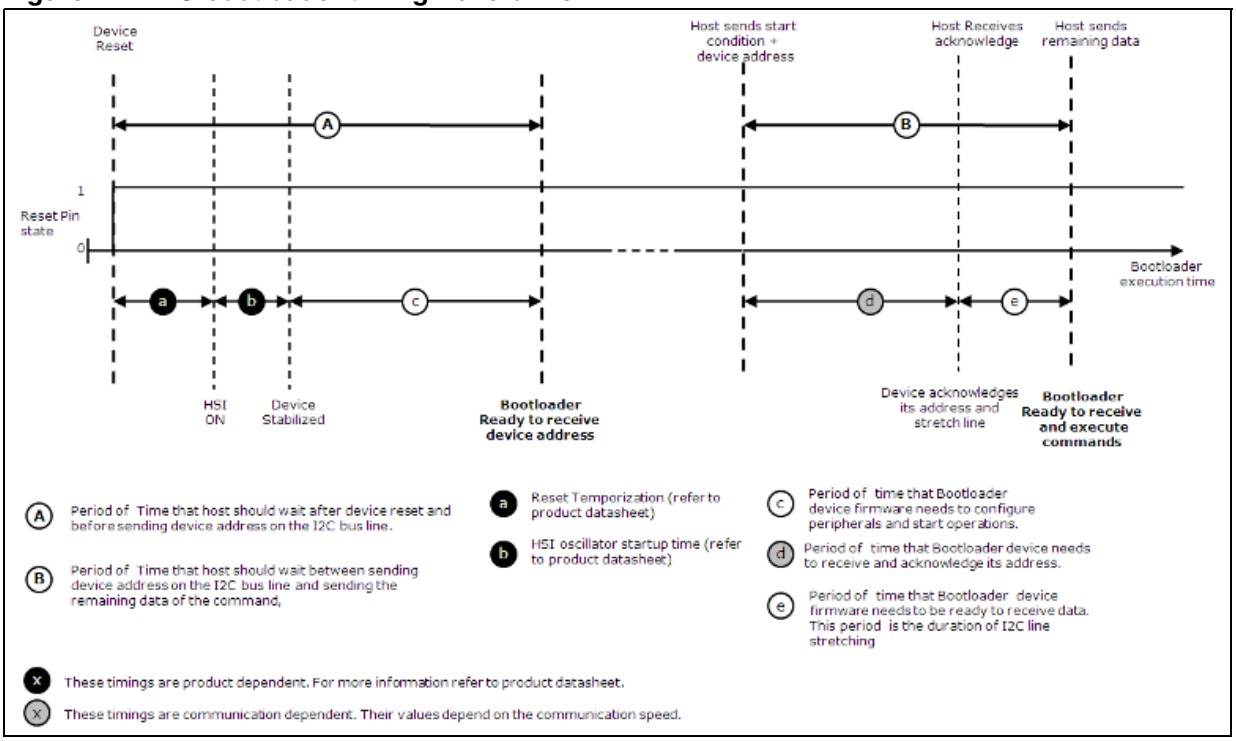
21.3 I2C bootloader timing characteristics

For the I2C bootloader, there are two main timings that need to be determined:

- After bootloader device reset, how much time host should wait before starting connection sequence with the device. This timing will be referred to as A through this section.
- After sending the device address, how much time is needed for the device to be ready to receive remaining data after detecting the start condition. This timing includes the period of line stretching between acknowledgement and the reception of the first data on I2C line. This timing will be referred to as B through this section.

Each of these two timings (A and B) is composed of different sub-timings as described in figure:

Figure 21. I2C bootloader timing waveforms



Note: For I2C communication, a timeout mechanism is implemented and it must be respected to execute bootloader commands correctly. This timeout is implemented between two I2C frame in the same command (eg: for Write memory command a timeout is inserted between command sending frame and address memory sending frame). Also the same timeout period is inserted between two successive data reception or transmission in the same I2C frame. If the timeout period is elapsed a system reset is generated to avoid bootloader crash. Please refer to the following tables to get the I2C timeout value of each product.

In erase memory command and read-out unprotect command, the duration of page erasing should be taken into consideration when implementing the host side. After sending the code of pages to be erased, the host should wait until the bootloader device performs page erasing to complete the remaining steps of erase command.

Table 73. I2C minimum timings for STM32F38xxx devices

Time	Description	Min	Unit
a	Reset temporization	1.5	ms
b	HSI oscillator startup time	0.001	ms
c	Bootloader firmware operations	0.041	ms
d	Start condition + one byte sending	0.0225	ms
e	Duration of I2C line stretching	0.0055	ms
A	Time = a + b + c	1.542	ms
B	Time = d + e	0.028	ms
-	I2C Timeout	10	ms

Table 74. I2C minimum timings for STM32F31xxx devices

Time	Description	Min	Unit
a	Reset temporization	1.5	ms
b	HSI oscillator startup time	0.001	ms
c	Bootloader firmware operations	0.0401	ms
d	Start condition + one byte sending	0.0225	ms
e	Duration of I2C line stretching	0.0054	ms
A	Time = a + b + c	1.5411	ms
B	Time = d + e	0.0279	ms
-	I2C Timeout	10	ms

Table 75. I2C minimum timings for STM32F429xx and STM32F439xx devices

Time	Description	Min	Unit
a	Reset temporization	TBD	ms
b	HSI oscillator startup time	TBD	ms
c	Bootloader firmware operations	TBD	ms
d	PLL Lock time	TBD	ms
e	Bootloader firmware operations	TBD	ms
f	Start condition + one byte sending	TBD	ms
g	Duration of I2C line stretching	TBD	ms
A	Time = a + b + c + d + e	TBD	ms
B	Time = f + g	TBD	ms
-	I2C Timeout	TBD	ms

22 Revision history

Table 76. Document revision history

Date	Revision	Changes
22-Oct-2007	1	Initial release.
22-Jan-2008	2	All STM32 in production (rev. B and rev. Z) include the bootloader described in this application note. Modified: *Section 3.1: Bootloader activation* and *Section 1.4: Bootloader code sequence*. Added: *Section 1.3: Hardware requirements*, *Section 1.5: Choosing the USART baud rate*, *Section 1.6: Using the bootloader* and *Section 3.2: Exiting System memory boot mode*. *Note 2* linked to Get, Get Version & Read Protection Status and Get ID commands in *Table 3: Bootloader commands*, *Note 3* added. Notion of “permanent” (Permanent Write Unprotect/Readout Protect/Unprotect) removed from document. Small text changes. Bootloader version upgraded to 2.0.
26-May-2008	3	Small text changes. RAM and System memory added to *Table : The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.*. *Section 1.6: Using the bootloader on page 8* removed. Erase modified, *Note 3* modified and *Note 1* added in *Table 3: Bootloader commands on page 9*. *Byte 3: on page 11* modified. *Byte 2: on page 13* modified. *Byte 2:*, *Bytes 3-4:* and *Byte 5: on page 15* modified, *Note 3* modified. *Byte 8: on page 18* modified. Notes added to *Section 2.5: Go command on page 18*. *Figure 11: Go command: device side on page 20* modified. Note added in *Section 2.6: Write Memory command on page 21*. *Byte 8: on page 24* modified. *Figure 14: Erase Memory command: host side* and *Figure 15: Erase Memory command: device side* modified. *Byte 3: on page 26* modified. *Table 3: Bootloader commands on page 9*. Note modified and note added in *Section 2.8: Write Protect command on page 27*. *Figure 16: Write Protect command: host side*, *Figure 17: Write Protect command: device side*, *Figure 19: Write Unprotect command: device side*, *Figure 21: Readout Protect command: device side* and *Figure 23: Readout Unprotect command: device side* modified.
29-Jan-2009	4	This application note also applies to the STM32F102xx microcontrollers. Bootloader version updated to V2.2 (see *Table 4: Bootloader versions*).

Table 76. Document revision history (continued)

Date	Revision	Changes
19-Nov-2009	5	IWDG added to *Table : The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.. Note* added. BL changed bootloader in the entire document. Go command description modified in *Table : The system clock is derived from the embedded internal high-speed RC, no external quartz is required for the bootloader code.*. Number of bytes awaited by the bootloader corrected in *Section 2.4: Read Memory command*. Note modified below *Figure 10: Go command: host side*. Note removed in *Section 2.5: Go command* and note added. Start RAM address specified and note added in *Section 2.6: Write Memory command*. All options are erased when a Write Memory command is issued to the Option byte area. *Figure 11: Go command: device side* modified. *Figure 13: Write Memory command: device side* modified. Note added and bytes 3 and 4 sent by the host modified in *Section 2.7: Erase Memory command*. Note added to *Section 2.8: Write Protect command*.
09-Mar-2010	6	Application note restructured. Value line and connectivity line device bootloader added (Replaces AN2662). *Introduction* changed. *Glossary* added.
20-Apr-2010	7	*Related documents*: added XL-density line datasheets and programming manual. *Glossary*: added XL-density line devices. *Table 3*: added information for XL-density line devices. *Section 4.1: Bootloader configuration*: updated first sentence. *Section 5.1: Bootloader configuration*: updated first sentence. Added *Section 6: STM32F101xx and STM32F103xx XL-density device bootloader*. *Table 46*: added information for XL-density line devices.
08-Oct-2010	8	Added information for high-density value line devices in *Table 3* and *Table 46*.
14-Oct-2010	9	Removed references to obsolete devices.
26-Nov-2010	10	Added information on ultralow power devices.
13-Apr-2011	11	Added information related to STM32F205/215xx and STM32F207/217xx devices. Added *Section 21: Bootloader timing characteristics*
06-Jun-2011	12	Updated: – *Table 12: STM32L100xx value line and STM32L15xxx medium-density bootloader versions* – *Table 18: STM32F2xxxx configuration in System memory boot mode* – *Table 20: STM32F2xxxx bootloader V2.x versions* – *Table 23: STM32F2xxxx bootloader V3.x versions*

Table 76. Document revision history (continued)

Date	Revision	Changes
28-Nov-2011	13	Added information related to STM32F405/415xx and STM32F407/417xx bootloader, and STM32F105xx/107xx bootloader V2.1. Added value line devices in *Section 4: STM32F100xx, STM32F101xx, STM32F102xx, STM32F103xx medium-density and high-density value line bootloader* title and overview.
30-Jul-2012	14	Added information related to STM32F051x6/STM32F051x8 and to High-density ultralow power STM32L151xx, STM32L152xx bootloader. Added case of BOOT1 bit in *Section 3.1: Bootloader activation*. Updated Connectivity line, High-density ultralow power line, STM32F2xx and STM32F4xx in *Table 3: Embedded bootloaders*. Added bootloader version V2.2 in *Table 7: STM32F105xx and STM32F107xx bootloader versions*. Added bootloader V2.2 in *Section 5.4.1: How to identify STM32F105xx/107xx bootloader versions*. Added note related to DFU interface below *Table 16: STM32L1xxxx high-density configuration in System memory boot mode*. Added V4.2 bootloader know limitations and updated description, and added V4.5 bootloader in *Table 17: STM32L1xxxx high-density bootloader versions*. Added note related to DFU interface below *Table 21: STM32F2xxxx configuration in System memory boot mode*. Added V3.2 bootloader know limitations, and added V3.3 bootloader in *Table 23: STM32F2xxxx bootloader V3.x versions*. Updated STM32F2xx and STM32F4xx system memory end address in *Table 24: STM32F40xxx/41xxx configuration in System memory boot mode*. Added note related to DFU interface below *Table 24: STM32F40xxx/41xxx configuration in System memory boot mode*. Added V3.0 bootloader know limitations, and added V3.1 bootloader in *Table 26: STM32F40xxx/41xxx bootloader version*. Added bootloader V2.1 know limitations in *Table 28: STM32F051xx bootloader versions*. Updated STM32F051x6/x8 system memory end address in *Table 46: Bootloader device-dependent parameters*. Added *Table 52: USART bootloader timings for high-density ultralow power devices*, and *Table 55: USART bootloader timings for STM32F051xx devices*. Added *Table 65: USB minimum timings for high-density ultralow power devices*.

Table 76. Document revision history (continued)

Date	Revision	Changes
24-Jan-2013	15	Updated generic product names throughout the document (see *Glossary*). Added the following new sections: – *Section 8: STM32L151xx and STM32L152xx medium-density plus ultralow power device bootloader*. – *Section 13: STM32F050x4 and STM32F050x6 device bootloader*. – *Section 14: STM32F372xx and STM32F373xx device bootloader*. – *Section 15: STM32F302xx and STM32F303xx device bootloader*. – *Section 16: STM32F383xx device bootloader*. – *Section 17: STM32F313xx device bootloader*. – *Section 18: STM32F427xx and STM32F437xx device bootloader*. – *Section 21.3: I2C bootloader timing characteristics*. Updated *Section 1: Related documents* and *Section 2: Glossary*. Added *Table 56* to *Table 62* (USART bootloader timings). Replaced *Figure 1* to *Figure 10*, and Figures *12*, *13* and *21*. Modified Tables *3*, *4*, *9*, *11*, *18*, *23*, *24*, *26* to *13*, *29*, *31*, *33*, *35*, *37*, *39* and *46*. Removed “X = 6: one USART is used” in *Section 3.3: Bootloader identification*. Replaced address 0x1FFFF 8002 with address 0x1FFF F802 in *Section 12.1: Bootloader configuration*. Modified procedure related to execution of the bootloader code in *Note: on page 28*, in *Section 6.3: Bootloader hardware requirements* and in *Section 9.3: Bootloader hardware requirements*.
06-Feb-2013	16	Added information related to I^2C throughout the document. Streamlined *Table 1: Applicable products* and *Section 1: Related documents*. Modified *Table 3: Embedded bootloaders* as follows: – Replaced "V6.0" with "V1.0" – Replaced "0x1FFFF7A6" with "0x1FFFF796" in row STM32F31xx – Replaced "0x1FFF7FA6" with "0x1FFFF7A6" in row STM32F051xx Updated figures *1*, *4* and *6*. Added *Note:* in *Glossary* and *Note:* in *Section 3.1: Bootloader activation*. ../...
		(Continued)

Table 76. Document revision history (continued)

Date	Revision	Changes
06-Feb-2013	16	Replaced: – "1.62 V" with "1.8 V" in tables *18*, *19*, *21*, *22*, *24*, *25*, *39* and *40* – "5 Kbytes" with "4 Kbytes" in row RAM of *Table 35* – "127 pages (2 KB each)" with "4 KB (2 pages of 2 KB each)" in rows F3 of *Table 46* – "The bootloader ID is programmed in the last two bytes of the device system memory" with "The bootloader ID is programmed in the last byte address - 1 of the device system memory" in *Section 3.3: Bootloader identification*. – "STM32F2xxxx devices revision Y" by "STM32F2xxxx devices revision X and Y" in *Section 10: STM32F205/215xx and STM32F207/217xx bootloader* – “Voltage Range 2” with “Voltage Range 1” in tables *11*, *16* and *28*.
21-May-2013	17	Updated: – *Introduction* – *Section 2: Glossary* – *Section 3.3: Bootloader identification* – *Section 7: STM32L151xx, STM32L152xx and STM32L100xx medium-density ultralow power device bootloader* to include STM32L100 value line – *Section 21.1: USART bootloader timing characteristics* – *Section 21.2: USB bootloader timing characteristics* – *Section 21.3: I2C bootloader timing characteristics* – *Table 1: Applicable products* – *Table 3: Embedded bootloaders* – *Table 27: STM32F051xx configuration in System memory boot mode* – *Table 29: STM32F050xx configuration in System memory boot mode* – *Table 46: Bootloader device-dependent parameters* – *Figure 11: Bootloader selection for STM32F050xx devices* Added *Section 19: STM32F429xx and STM32F439xx device bootloader*.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

ST PRODUCTS ARE NOT AUTHORIZED FOR USE IN WEAPONS. NOR ARE ST PRODUCTS DESIGNED OR AUTHORIZED FOR USE IN: (A) SAFETY CRITICAL APPLICATIONS SUCH AS LIFE SUPPORTING, ACTIVE IMPLANTED DEVICES OR SYSTEMS WITH PRODUCT FUNCTIONAL SAFETY REQUIREMENTS; (B) AERONAUTIC APPLICATIONS; (C) AUTOMOTIVE APPLICATIONS OR ENVIRONMENTS, AND/OR (D) AEROSPACE APPLICATIONS OR ENVIRONMENTS. WHERE ST PRODUCTS ARE NOT DESIGNED FOR SUCH USE, THE PURCHASER SHALL USE PRODUCTS AT PURCHASER’S SOLE RISK, EVEN IF ST HAS BEEN INFORMED IN WRITING OF SUCH USAGE, UNLESS A PRODUCT IS EXPRESSLY DESIGNATED BY ST AS BEING INTENDED FOR “AUTOMOTIVE, AUTOMOTIVE SAFETY OR MEDICAL” INDUSTRY DOMAINS ACCORDING TO ST PRODUCT DESIGN SPECIFICATIONS. PRODUCTS FORMALLY ESCC, QML OR JAN QUALIFIED ARE DEEMED SUITABLE FOR USE IN AEROSPACE BY THE CORRESPONDING GOVERNMENTAL AGENCY.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.
Information in this document supersedes and replaces all information previously supplied.
The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com