



Introduction

Interactive interfaces are more and more integrated into many applications such as medical devices, process control, mobile phones and other hand-held devices. These interfaces are based mainly on graphic HMIs (human machine interface) using color LCDs.

Worldwide, the desire for color support is growing dramatically. The purpose of this application note is to address this aspect by describing how to use the STM32F10xxx FSMC (flexible static memory controller) to drive a TFT color LCD. This document first describes how to connect a color LCD to the FSMC, then it provides a TFT LCD interfacing example.

The demonstration firmware delivered with this application note, the STM32F10xxx firmware library, and other such firmware are available for download from the STMicroelectronics website: www.st.com.

Contents

- 1 STM32F10xxx flexible static memory controller (FSMC) overview . . . 6**
 - 1.1 FSMC NOR Flash/SRAM bank description 6

- 2 TFT LCD (thin-film-transistor liquid crystal display) interfacing with FSMC 9**
 - 2.1 Common color LCD interfaces 9
 - 2.2 Typical use of the FSMC to interface with an LCD module 10
 - 2.3 Connecting the FSMC to the LCD Intel 8080-like (I80) interface 10
 - 2.4 Connecting the FSMC to the LCD Motorola 6800 (M68)-like interface . . 11
 - 2.5 Interfacing an LCD with the FSMC of a High-density STM32F10xxx device offered in a 100-pin package 12
 - 2.6 Timing computation 13
 - 2.6.1 LCD timings 13
 - 2.6.2 FSMC timings 14

- 3 Interfacing the Ampire AM-240320L8TNQW00H TFT LCD with the FSMC 15**
 - 3.1 Ampire TFT LCD interface 15
 - 3.2 Hardware connection 15
 - 3.3 Timing computation 16

- 4 Firmware description 18**
 - 4.1 LCD driver firmware description 18
 - 4.2 Running the demo 19
 - 4.2.1 Menu 19
 - 4.2.2 Demo startup 20
 - 4.2.3 Navigation 22
 - 4.3 LCD demo applications 23
 - 4.3.1 Internal Flash memory 24
 - 4.3.2 NOR Flash memory 26
 - 4.3.3 External SRAM 27
 - 4.3.4 NAND Flash memory 29
 - 4.3.5 SD card 32

5	Color TFT LCD interfacing with the FSMC: performance	34
6	LCD demo programming	35
6.1	Programming the M29W128 NOR Flash memory	36
6.2	LCD demo programming	39
7	Conclusion	43
8	Revision history	44

List of tables

Table 1.	Advantages/drawbacks of LCD controllers in MCUs	9
Table 2.	Parallel interface Read/Write status	10
Table 3.	LCD Intel 8080 interface pins	11
Table 4.	FSMC and LCD Intel 6800 pins	11
Table 5.	LCD 8080-like interface timing characteristics	13
Table 6.	FSMC and Ampire LCD pins	15
Table 7.	LCD 8080-like interface timing characteristics	16
Table 8.	Driver library description	18
Table 9.	High-level LCD driver functions	18
Table 10.	LCD display speeds	34
Table 11.	Document revision history	44

List of figures

Figure 1.	Asynchronous NOR Flash read access timing	7
Figure 2.	Asynchronous NOR Flash write access timing	8
Figure 3.	Connecting the FSMC to an LCD Intel 8080-like interface	11
Figure 4.	Connecting the FSMC to an LCD Motorola 6800-like interface	12
Figure 5.	Connecting the FSMC to an LCD Motorola 6800-like interface ⁽¹⁾	12
Figure 6.	Connecting the FSMC to an LCD Intel 8080-like interface in devices delivered in 100-pin packages	13
Figure 7.	TFT LCD hardware connection with FSMC	16
Figure 8.	Structure of the demonstration menus	19
Figure 9.	Warning message	20
Figure 10.	ST logo	20
Figure 11.	STM32 family	20
Figure 12.	Main menu	21
Figure 13.	Corresponding submenus.	21
Figure 14.	Navigating in the demonstration menus	22
Figure 15.	No bitmap files (see note 1)	23
Figure 16.	Copy images to the internal Flash memory	24
Figure 17.	Copy images to internal Flash memory complete.	24
Figure 18.	Animation showing the STM32F10xxx display speed	25
Figure 19.	ST logo	25
Figure 20.	Timing display submenu	25
Figure 21.	Timing display submenu	26
Figure 22.	Timing display submenu	26
Figure 23.	Timing display submenu	27
Figure 24.	Copy images to external SRAM	27
Figure 25.	Copy images to external SRAM complete	28
Figure 26.	Timing display submenu	28
Figure 27.	Timing display submenu	29
Figure 28.	Copy images to the NAND Flash memory	29
Figure 29.	Copy images to NAND Flash complete	30
Figure 30.	Timing display submenu	30
Figure 31.	NAND Erase operation	31
Figure 32.	NAND Erase operation complete	31
Figure 33.	Copy images to SD card	32
Figure 34.	Copy images to SD card complete	32
Figure 35.	Timing display submenu	33
Figure 36.	Programming the NOR Flash memory	36
Figure 37.	Selecting the STM32_FSMC-LCD_Demo_NORFlash.dfu file	37
Figure 38.	Click Yes to continue	37
Figure 39.	NOR Flash memory programming	38
Figure 40.	Programming the demo	39
Figure 41.	Selecting the STM32_FSMC-LCD_Demo.dfu file.	40
Figure 42.	Click Yes to continue	40
Figure 43.	Internal Flash memory programming	41
Figure 44.	DFU mode left.	42

1 STM32F10xxx flexible static memory controller (FSMC) overview

The STM32F10xxx flexible static memory controller (FSMC) is an embedded external memory controller that allows the STM32F10xxx microcontroller to interface with a wide range of memories, including SRAM, NOR Flash, NAND Flash and LCD modules.

The FSMC NOR Flash/SRAM bank is suitable for MCU parallel color LCD interfaces. It is described in this section.

1.1 FSMC NOR Flash/SRAM bank description

To control a NOR Flash/SRAM memory, the FSMC provides the following features:

- Select the bank to be used to map the NOR Flash/SRAM memory: there are four independent banks that can be used to interface with NOR Flash/SRAM/PSRAM memories, and each bank is selected using a separate Chip Select pin.
- Enable or disable the address/data multiplexing feature
- Select the memory type to be used: NOR Flash/SRAM/PSRAM
- Define the external memory databus width: 8/16 bits
- Enable or disable the burst access mode for NOR Flash synchronous memories
- Configure the use of the wait signal: enable/disable, polarity setting and timing configuration
- Enable or disable the extended mode: this mode is used to access the memory with different timing configurations for read and write operations.

As the NOR Flash/PSRAM controller supports both asynchronous and synchronous memories, the user should select only the useful parameters depending on the memory characteristics.

The FSMC also provides the possibility of programming several parameters to interface correctly with the external memory. Depending on the memory type, some parameters are not used.

If an external asynchronous memory is used, the user has to compute and set the following parameters according to the AC timing information specified in the memory datasheet:

- ADDSET: address setup time
- ADDHOLD: address hold time
- DATAST: data setup time
- ACCMOD: access mode

This parameter gives the FSMC the flexibility to access a wide variety of asynchronous static memories. There are four extended access modes that allow write access while reading the memory with different timings, if the memory supports this kind of feature.

When the extended mode is enabled, the FSMC_BTR register is used for read operations and the FSMC_BWR register is used for write operations.

In the case where a synchronous memory is used, the user has to compute and set the following parameters:

- CLKDIV: clock divide ratio
- DATLAT: data latency

Note that NOR Flash memory read operations can be synchronous if the memory supports this mode, while write operations usually remain asynchronous.

When programming a synchronous NOR Flash memory, the memory automatically switches between the synchronous and the asynchronous mode, so in this case, all parameters have to be set correctly.

Figure 1 and Figure 2 show the different timings during a typical NOR Flash memory access.

Figure 1. Asynchronous NOR Flash read access timing

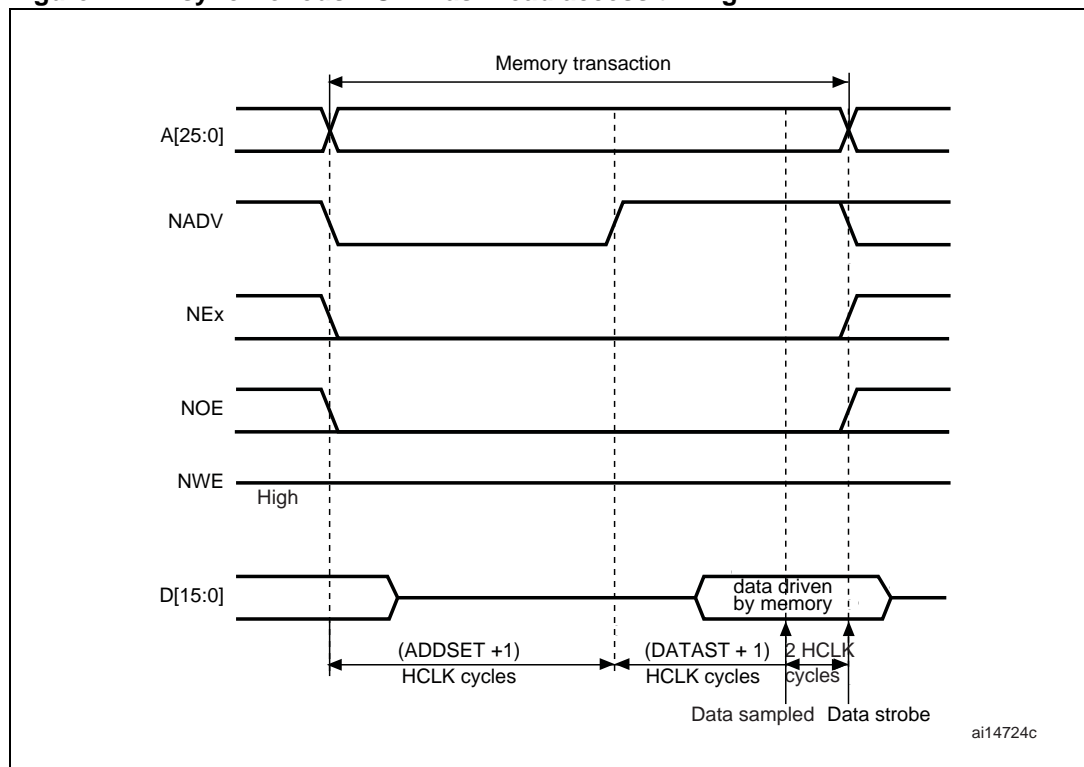
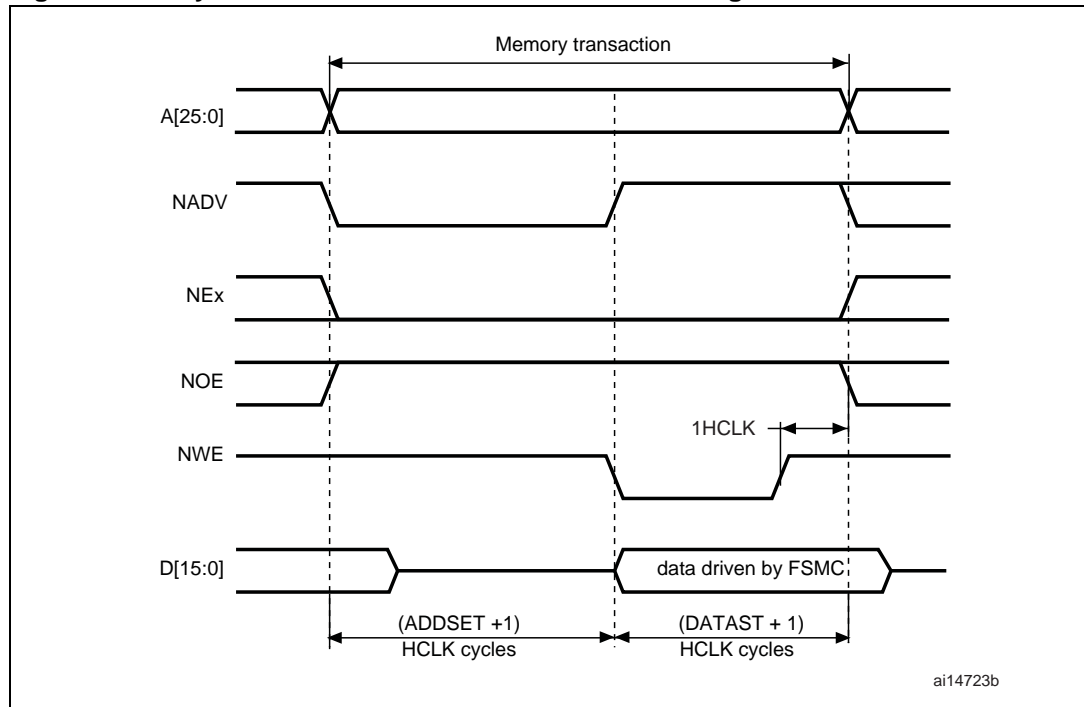


Figure 2. Asynchronous NOR Flash write access timing



2 TFT LCD (thin-film-transistor liquid crystal display) interfacing with FSMC

2.1 Common color LCD interfaces

Dot-matrix LCD units, which are usually controlled with a Silicon embedded glass LCD driver, can be interfaced with microcontrollers using a serial interface. This type of LCD unit has an embedded RAM for display and do not require any special microcontroller feature.

Unlike dot-matrix LCD units, color LCDs need a specific controller.

The use of color LCD controllers means that each pixel needs four lines: three analog voltage lines for red, green and blue (horizontal) and one voltage line for selection (vertical).

Typically, color LCDs can be interfaced in two ways:

- RGB interface with synchronization signals
- MPU (microprocessor unit) interface (parallel or serial)

All graphic LCDs come with built-in drivers in the form of a chip or silicon on glass. These drivers do the conversion between RGB signals and synchronization and pixel control.

Many LCDs also come with an LCD controller that does the conversion between the MPU interface and the RGB signals. Some chips are both drivers and controllers. The role of the controller is to constantly refresh the LCD.

[Table 1](#) lists the differences between LCDs that integrate controllers and LCDs that need an MPU LCD controller.

Table 1. Advantages/drawbacks of LCD controllers in MCUs

LCD controller in MCU	LCD controller in LCD
External RAM is needed (cost)	RAM is included in the LCD controller
Continuous refresh of LCD causes a high power consumption and a high MCU bandwidth usage	The MCU interface is active only when the LCD display changes.
Good for menus and interfaces, not suitable for movies, games, high-end OS (needs an MPU architecture).	Good for menus and interfaces, not suitable for movies, games, high-end OS.
Not proven with MCU (only MPU with LCD are designed in real products).	Cost-effective QVGA (240 × 320 pixels) and below. Widely used in the appliance and industrial market.

The most common LCD MPU parallel interfaces are of the Intel 8080 (I80) and Motorola 6800 (M68) type.

The next section focusses on describing how to connect LCD Intel 8080-like and Motorola 6800-like interfaces with an STM32F10xxx FSMC.

LCD controller signals are divided into two types: data signals and control signals.

The data signals are connected to the LCD databus and depend on the LCD color depth (8 bits, 9 bits, 16 bits, 18 bits or 24 bits (true colors)).

The control signals are used to define the operation type (read or write), and whether the operation consists in addressing (writing commands to) LCD registers or the display RAM.

[Table 2](#) describes the control signals for LCD Intel 8080-like and Motorola 6800-like interfaces.

Table 2. Parallel interface Read/Write status

Control	6800 processor		8080 processor		Function
	E	R/ \overline{W}	\overline{RD}	\overline{RW}	
0	1	1	0	1	Reads registers (status)
0	1	0	1	0	Writes commands to registers
1	1	1	0	1	Reads display data (RAM)
1	1	0	1	0	Writes display data (RAM)

2.2 Typical use of the FSMC to interface with an LCD module

The STM32F10xxx FSMC has four different banks of 64 Mbytes to support NOR Flash memories/PSRAMs and similar external memories.

The external memories share the addresses, data and control signals with the controller.

Each external device is accessed by means of a unique Chip Select signal, but the FSMC can gain access to only one external device at a time. Each bank is configured by means of dedicated registers including the different features and the timing parameters.

As we have seen above, the FSMC provides all the signals needed by the LCD controller. The FSMC signals used for LCD interfacing are described below:

- FSMC [D0:D15]: FSMC databus: 16-bit width
- FSMC NEx: FSMC Chip Select
- FSMC NOE: FSMC Output Enable
- FSMC NWE: FSMC Write Enable
- FSMC Ax: one address line used to select between LCD Registers and LCD Display RAM where x can be 0 to 25

Note: The prefix "N" in signal names specifies that the signal is active low.

The LCD address depends on the used FSMC NOR Flash/PSRAM bank (NEx) and the selected address (Ax) to drive the LCD RS pin.

Example: with NE2 and A4, the LCD base address will be 0x6400 0000 and 0x6400 0020; with NE4 and A0, the LCD base address will be 0x6C00 0000 and 0x6C00 0002.

2.3 Connecting the FSMC to the LCD Intel 8080-like (I80) interface

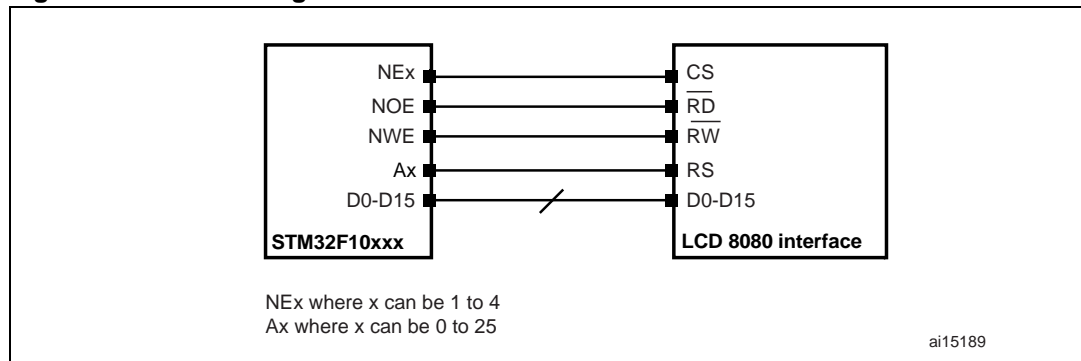
The LCD Intel 8080-like MPU interface is based on four control signals and a databus with a variable width depending on the MPU interface capability. [Table 3](#) describes these signals.

Table 3. LCD Intel 8080 interface pins

LCD signals	Signal description
RS	LCD Register Select
D0-D15	Data D0-D15
\overline{CS}	Chip Select
\overline{RD}	Read operation: active low
\overline{WR}	Write operation: active low

A typical connection between FSMC and LCD Intel 8080 is illustrated on [Figure 3](#).

Figure 3. Connecting the FSMC to an LCD Intel 8080-like interface



2.4 Connecting the FSMC to the LCD Motorola 6800 (M68)-like interface

The LCD Motorola 6800-like MPU interface is based on four control signals and a databus with a variable width depending on the MPU interface capability. [Figure 4](#) describes these signals.

Table 4. FSMC and LCD Intel 6800 pins

LCD signals	Signal description
RS	LCD Register Select
D0-D15	Data D0-D15
\overline{CS}	Chip Select
E	Read Write Enable/Disable
R/ \overline{W}	Read operation / Write operation

[Figure 4](#) illustrates a typical connection between the FSMC and an LCD Intel 6800-like interface.

Figure 4. Connecting the FSMC to an LCD Motorola 6800-like interface

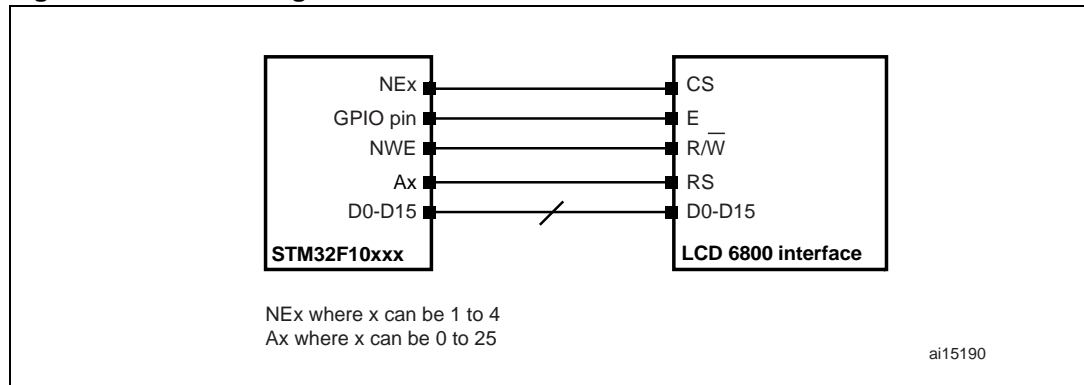
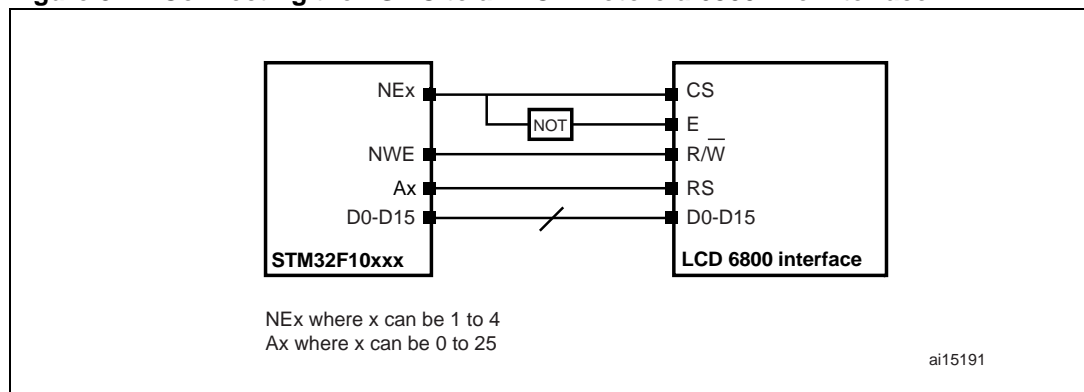


Figure 5 shows an alternative connection of an LCD Motorola 6800-like interface with the FSMC.

Figure 5. Connecting the FSMC to an LCD Motorola 6800-like interface⁽¹⁾



1. Many LCDs have two complementary Chip Select pins.

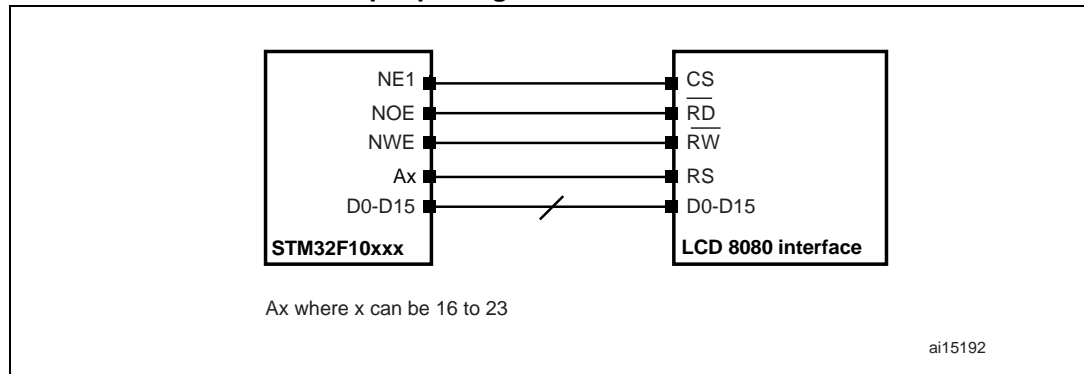
2.5 Interfacing an LCD with the FSMC of a High-density STM32F10xxx device offered in a 100-pin package

The STM32F101VC/D/E and STM32F103VC/D/E are High-density devices delivered in 100-pin packages. They have a reduced number of FSMC signals. Figure 6 give an overview of how to interface a color LCD with the FSMC on devices in 100-pin packages.

The NOR Flash/SRAM FSMC signals available in this package for LCD interfacing are listed below:

- FSMC D0-D15: FSMC databus: 16-bit width
- FSMC NE1: FSMC Chip Select: Only NOR Flash/SRAM Bank1 can be used
- FSMC NOE: FSMC Output Enable
- FSMC NWE: FSMC Write Enable
- FSMC A_x: one address line used to select between LCD Registers and LCD display RAM where x can be 16 to 23

Figure 6. Connecting the FSMC to an LCD Intel 8080-like interface in devices delivered in 100-pin packages



2.6 Timing computation

As described above, for asynchronous NOR Flash-like memories, there are different possible access protocols. It is first necessary to define what kind of protocol is to be used with the user LCD. The choice depends on the different control signals and on the behavior of the LCD during read and write transactions.

2.6.1 LCD timings

Table 5 gives the list of the main color LCD timings needed to compute the corresponding FSMC timings.

Table 5. LCD 8080-like interface timing characteristics

Symbol	Parameter	Condition
t_{AH}	Address hold time	RS
t_{AS}	Address setup time	RS
t_{CYC}	System cycle time	-
$t_{CYC(READ)}$	System cycle time (Read)	-
t_{WRLW}	Low pulse width for write	\overline{WR}
t_{WRLR}	Low pulse width for read	\overline{RD}
t_{WRHW}	High pulse width for write	\overline{WR}
t_{WRHR}	High pulse width for read	\overline{RD}
t_{DS}	Data setup time	D0-D15
t_{DH}	Data hold time	D0-D15
t_{ACC}	Data access time	D0-D15 (this timing depends on the load capacitance, C_L)
t_{OD}	Output disable time	-

2.6.2 FSMC timings

Usually, ModeB is used for LCD interfacing (so the extended mode is not used) and the same timings are used for read and write operations.

In this case, the FSMC needs three timing parameters: ADDSET, DATAST and ADDHOLD.

These parameters are computed according to the LCD characteristics and as a function of the STM32F10xxx AHB clock (HCLK).

Based on the NOR Flash/SRAM read and write access timing diagrams illustrated in [Figure 1](#) and [Figure 2](#), the following equations are found:

The write or read access time is the time between the falling edge and the rising edge of the LCD Chip Select signal. This timing is computed as a function of the FSMC timing parameter:

$$\text{Write/Read access} = ((\text{ADDSET} + 1) + (\text{DATAST} + 1)) \times \text{HCLK} = t_{\text{CYC}}$$

The DATAST parameter is measured between the falling edge and the rising edge of the write signal in a write operation:

$$\text{Write Enable signal low to high} = \text{DATAST} \times \text{HCLK}$$

To make sure of the correct timing configuration of the FSMC, the timings have to take into consideration:

- the maximum write/read access time
- The different internal memory delays.

Hence, we have the following equations:

$$((\text{ADDSET} + 1) + (\text{DATAST} + 1)) \times \text{HCLK} = \max(t_{\text{CYC}}, t_{\text{CYC(READ)}})$$

$$\text{DATAST} \times \text{HCLK} = t_{\text{WRLW}}$$

DATAST must verify:

$$\text{DATAST} = (((t_{\text{ACC}} + t_{\text{AS}}) + (t_{\text{su(Data_NE)}} + t_{\text{v(A_NE)}})) / \text{HCLK}) - \text{ADDSET} - 4$$

Where:

- $t_{\text{su(Data_NE)}}$: FSMC_NEx low to data valid
- $t_{\text{v(A_NE)}}$: FSMC_NEx low to FSMC_A valid
- $(t_{\text{su(Data_NE)}} + t_{\text{v(A_NE)}}) = 36 \text{ ns}$

Note: *When selecting the FSMC timing mode, make sure that the NOE behavior corresponds to the used LCD requirements. Please refer to the NOR Flash/SRAM FSMC timing diagram description in the STM32F10xxx reference manual.*

For more details on the flexible static memory controller (FSMC) timing values, please refer to the STM32F101xC/D/E and STM32F103xC/D/E High-density datasheets.

3 Interfacing the Ampire AM-240320L8TNQW00H TFT LCD with the FSMC

3.1 Ampire TFT LCD interface

The AM-240320L8TNQW00H TFT LCD integrates an embedded LCD controller, the ILI9320. It can be interfaced as Intel 8080 or Motorola 6800. It has a color depth of 18 bits or 16 bits and, a resolution of 320 × 240 pixels.

In this application note, the FSMC is configured to interface the Ampire TFT LCD as follows:

FSMC Bank1 NOR/PSRAM 4 is selected to support the LCD device:

- Memory type is SRAM
- Databus width 16 bits
- Non-multiplexed memory

All remaining parameters have to be kept cleared.

3.2 Hardware connection

The Ampire TFT LCD (ILI9320 controller) is connected to the STM32F10xxx FSMC through the FSMC bank1 NOR/PSRAM 4, which is used to control the LCD signals.

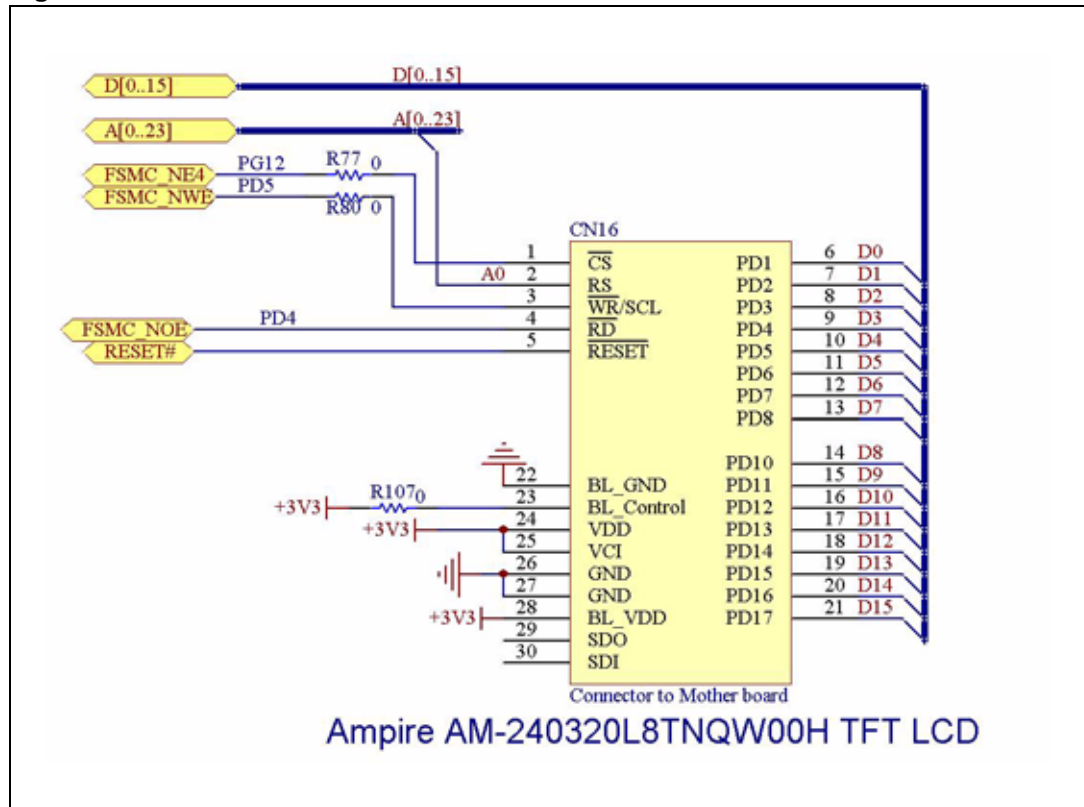
[Table 6](#) shows the correspondence between LCD module pins and FSMC pins. All FSMC GPIOs are configured in alternate function push-pull.

Table 6. FSMC and Ampire LCD pins

LCD signals	FSMC signals	Pin / Port assignement	Signal description
RS	A0	PF00	Address A0
D0-D15	D0-D15	GPIOD/GPIOE	Data D0-D15
\overline{CS}	NE4	PG12	Chip Enable
\overline{RD} (E)	NOE	PD04	Ouput Enable
\overline{WR} (R/W)	NWE	PD05	Write Enable

[Figure 7](#) shows a typical connection between the STM32F10xxx microcontroller and the Ampire LCD (ILI9320 controller). This connection is extracted from the schematic of the STM3210E-EVAL evaluation board.

Figure 7. TFT LCD hardware connection with FSMC



3.3 Timing computation

Table 7 gives the list of the Ampire AM-240320L8TNQW00H color LCD timings needed to compute the corresponding FSMC timings.

Table 7. LCD 8080-like interface timing characteristics

Symbol	Parameter	Value
t_{AH}	Address hold time	5 ns
t_{AS}	Address setup time	5 ns for Read 10 ns for Write
t_{CYC}	System cycle time	100 ns
$t_{CYC(READ)}$	System cycle time (Read)	300 ns
t_{WRLW}	Low pulse width for write	50 ns
t_{WRLR}	Low pulse width for read	150 ns
t_{WRHW}	High pulse width for write	50 ns
t_{WRHR}	High pulse width for read	150 ns
t_{DS}	Data setup time	10 ns
t_{DH}	Data hold time	15 ns
t_{ACC}	Data access time	100 ns maximum

Corresponding LCD FSMC timings

Using the formulas described above, we have:

- the HCLK frequency is 72 MHz
- the FSMC access mode is ModeB

So, we have the following equations:

$((ADDSET + 1) + (DATAST + 1)) \times HCLK = 100$ (for write cycle only as we only write to the LCD)

$$DATAST \times 13.8 = 50$$

The DATAST must verify:

$$DATAST = ((t_{ACC} + t_{AS}) + 36)/HCLK - ADDSET - 4$$

So, we will have the following FSMC timings:

- Address setup time: 0x1
- Address hold time: 0x0
- Data setup time: 0x5

4 Firmware description

This application note is based on:

- the STM32F10xxx firmware library
- the LCD driver firmware (offering the main functions required to control the LCD and the display application)
- a specific firmware to call the LCD driver functions, as well as other functions required for control and display (`main.c` and `stm32f10x_it.c` files).

The user may build any similar application using the same library and driver, and different interfacing firmware/hardware.

4.1 LCD driver firmware description

The user may interface the LCD directly through the driver application layer. The driver functions are summarized in the following sections. [Table 8](#) presents the general driver file organization.

Table 8. Driver library description

Files	Description
<code>fonts.h</code> , <code>lcd.h</code> , <code>lcd.c</code>	<ul style="list-style-type: none"> – LCD definitions, type definitions and function prototypes – Basic functions (init, read, write register, write LCD RAM, set display window, write bitmap pictures, display char, clear line, clear LCD, etc.)

High-level functions

These are the functions that can simply be called by the final application to execute all needed configurations and perform high-end functionalities (LCD initialization, text writing, display of bitmap images, configuration of all the hardware components, etc.). [Table 9](#) presents these functions.

Table 9. High-level LCD driver functions

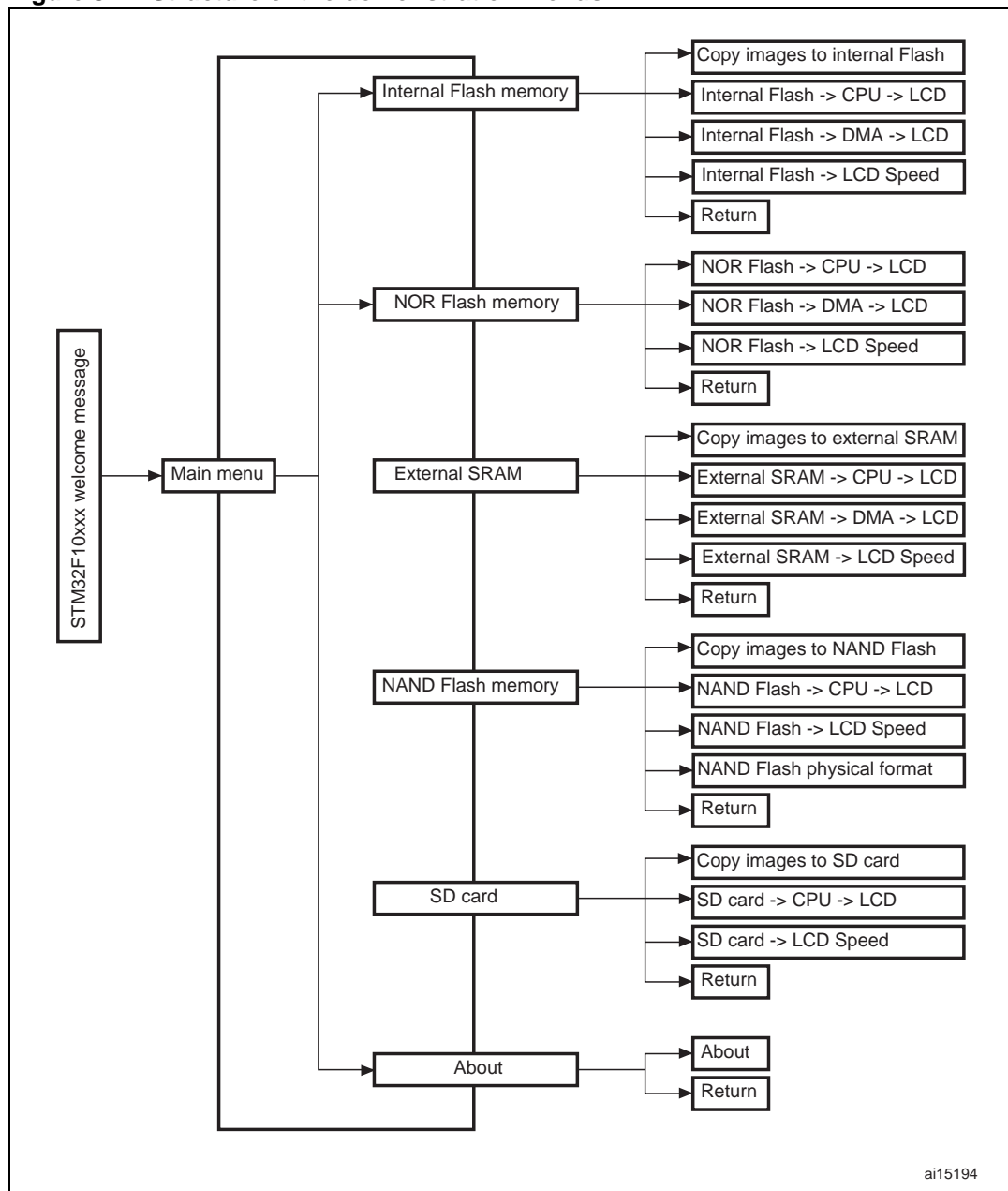
Function name	Description
<code>STM3210E_LCD_Init</code>	Initializes the LCD
<code>LCD_SetTextColor</code>	Sets the text color
<code>LCD_SetBackColor</code>	Sets the background color
<code>LCD_ClearLine</code>	Clears the selected line
<code>LCD_Clear</code>	Clears the whole LCD
<code>LCD_DisplayStringLine</code>	Displays a maximum of 20 char on the LCD
<code>LCD_SetDisplayWindow</code>	Sets a display window
<code>LCD_WriteBMP</code>	Displays a bitmap picture loaded in the internal Flash memory

4.2 Running the demo

4.2.1 Menu

Figure 8 shows the menu system of the STM32F10xxx LCD demonstration. The main menu is shown on the left hand side. The UP, DOWN, RIGHT and LEFT joystick directions allow the user to navigate between items in the main menu and the submenus. To enter a submenu, press the SEL push-button (SEL push-button is the switch closure that occurs when the joystick button is pushed). To exit a submenu select the **Return** menu and press SEL.

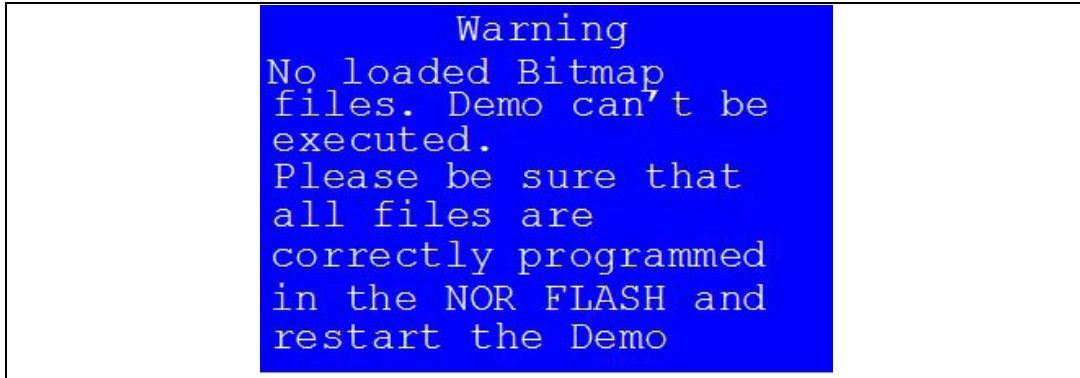
Figure 8. Structure of the demonstration menus



4.2.2 Demo startup

After a board reset, at demo startup, the icons and bitmap files are checked in the NOR Flash memory. All the icons have to be correctly programmed in the NOR Flash memory for the demo to start, so if an icon is missing, the demo does not start and the message shown in [Figure 9](#) is displayed on the LCD screen.

Figure 9. Warning message



However, if the icons are correctly loaded into the NOR Flash memory, the Welcome screen is displayed and the ST Logo appears on the LCD (see [Figure 10](#)).

Figure 10. ST logo



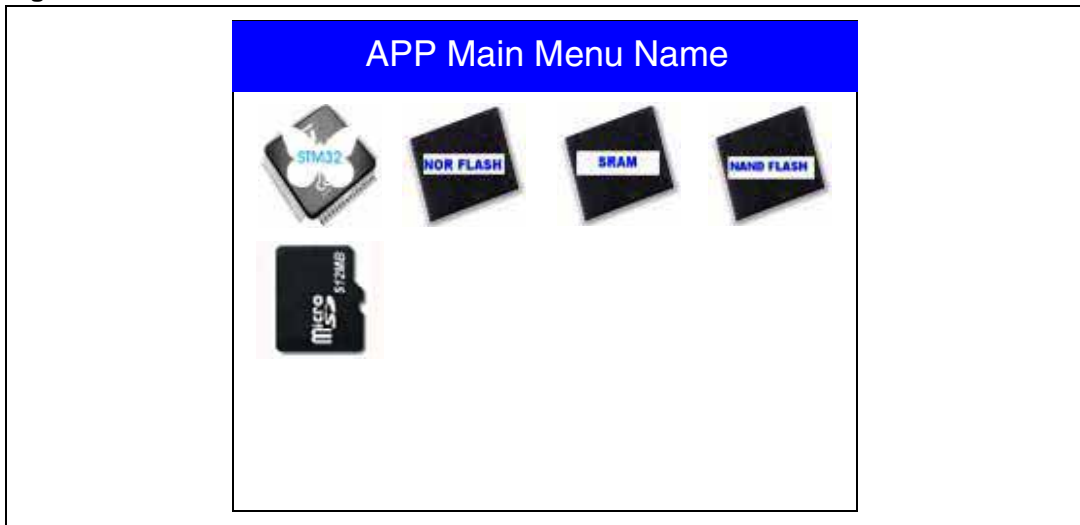
Then, after 1 second, the STM32F10xxx slide shown in [Figure 11](#) is displayed on the LCD screen.

Figure 11. STM32 family



After the STM32F10xxx welcome screen, the main menu appears. It is displayed in the form of a set of icons that shows all the submenus in the same screen. You can navigate using the UP, DOWN, RIGHT and LEFT joystick directions to select the desired submenu. To enter the desired submenu, press the SEL joystick push-button, and the new submenu corresponding to the selected icon is displayed.

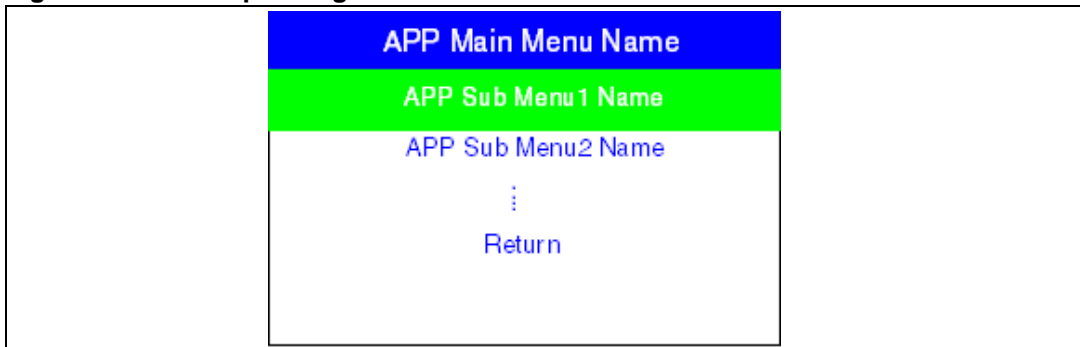
Figure 12. Main menu



1. The icons shown in *Figure 12* are taken from http://commons.wikimedia.org/wiki/Crystal_Clear.

Once a submenu has been selected, the name of the application is listed at the top of the display and all the corresponding submenus are listed below as shown in *Figure 13*.

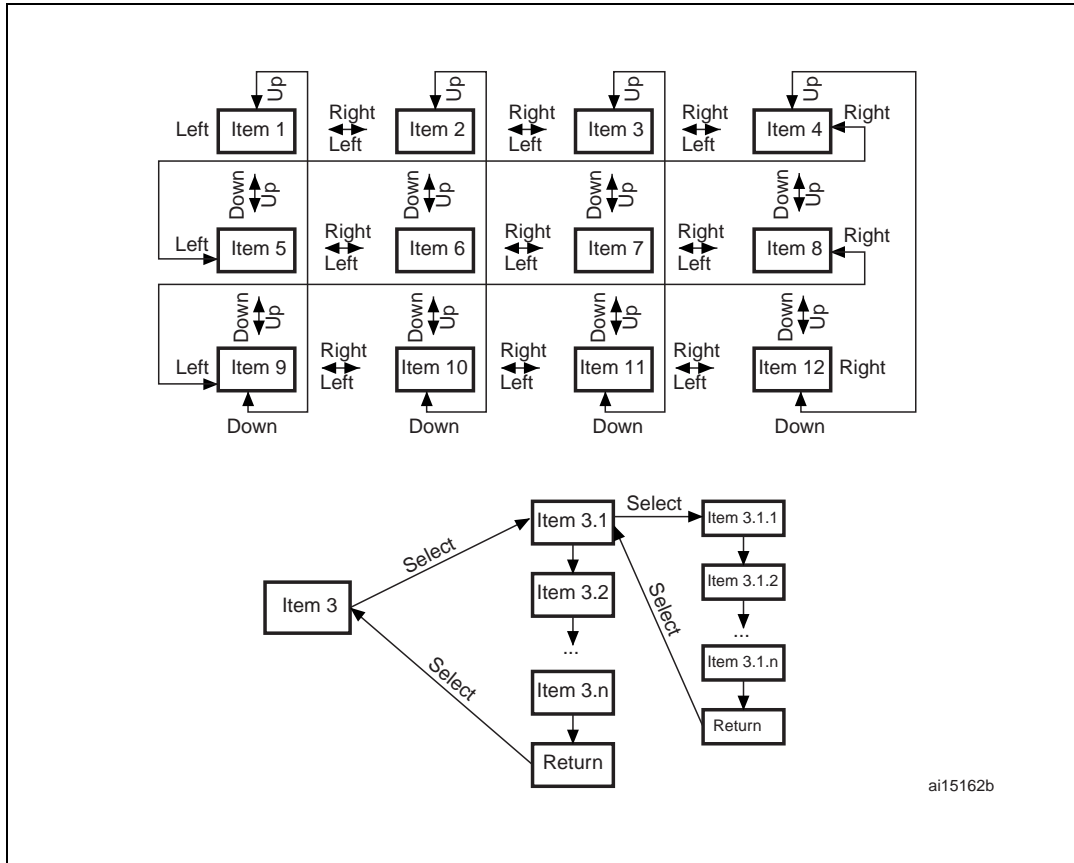
Figure 13. Corresponding submenus



4.2.3 Navigation

The demonstration menu is based on circular navigation, submenu selection, item selection and back capability as described in [Figure 14](#).

Figure 14. Navigating in the demonstration menus



The user navigates using the joystick push-buttons: RIGHT, LEFT, SEL, UP and DOWN located on the evaluation board.

- the UP, DOWN, RIGHT and LEFT push-buttons are used to perform circular navigation in the main menu and the current menu items
- the SEL push-button selects the current item
- the UP and DOWN push-buttons are used for vertical navigation in the submenus
- to return to the upper menu, go to the Return line and press SEL

4.3 LCD demo applications

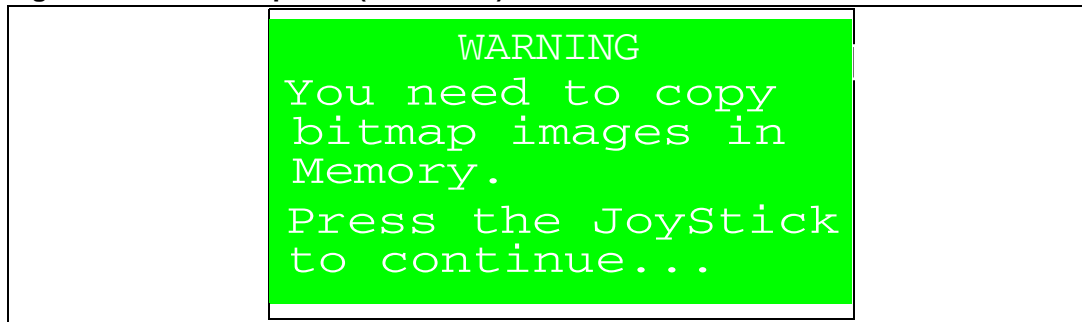
The purpose of the LCD demo is to provide full examples of how to read data from different memory locations, and display them on the LCD. For data transfer, we have used CPU transfer and DMA transfer. For each case, the display timing is computed and displayed on the LCD. This timing is the sum of the time taken to read from the selected memory and the time taken to write the image to the LCD.

This section provides a detailed description of each part of the demonstration.

- Note: 1 *In the demonstration, the core runs at HCLK = 72 MHz.*
- 2 *All timings used for this demo are related to the STM3210E-EVAL board memories. You can retrieve other values if you use different memories. The STM3210E-EVAL board has the following memories:*
- *NOR Flash: Numonyx M29W128F, Numonyx M29W128G, Spansion S29GL128P*
 - *External SRAM: ISSI IS61WV51216BLL*
 - *NAND Flash: Numonyx NAND512W3A2CNB*
 - *SD card™: Numonyx SMS128FF*
- 3 *The properties of the used bitmap image files are:*
- *encoding: bitmap*
 - *width: 320*
 - *height: 240*
 - *16 bits per pixel*
 - *image data size: 150 Kbytes*

Note: *In any LCD demo application, if the bitmap images are not loaded into the selected memory (Internal Flash memory, External SRAM) at the reserved address, the message shown in [Figure 15](#) is displayed on the LCD screen.*

Figure 15. No bitmap files (see note 1)



1. Memory can be the internal Flash memory or the external SRAM.

4.3.1 Internal Flash memory

This part of the LCD demo is dedicated to displaying the bitmap pictures stored in the internal Flash memory, as well as the computed one-frame display time.

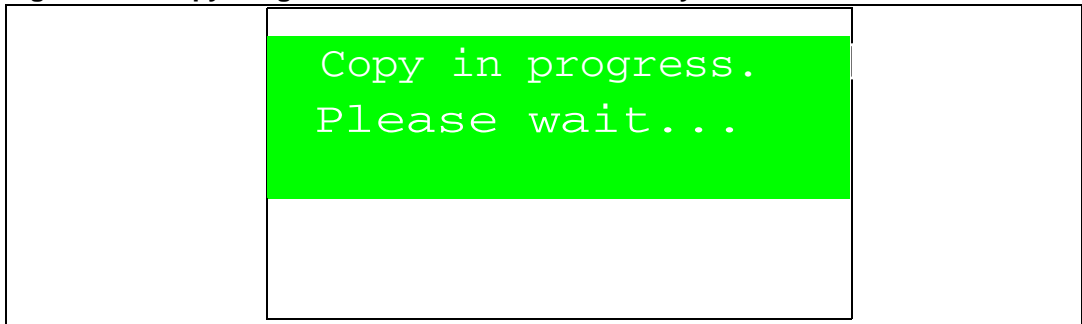
It is first necessary to load the corresponding images into the dedicated, internal Flash memory.

Copy images to internal Flash

This submenu is used to copy images from the NOR Flash memory to a specified address in internal Flash memory.

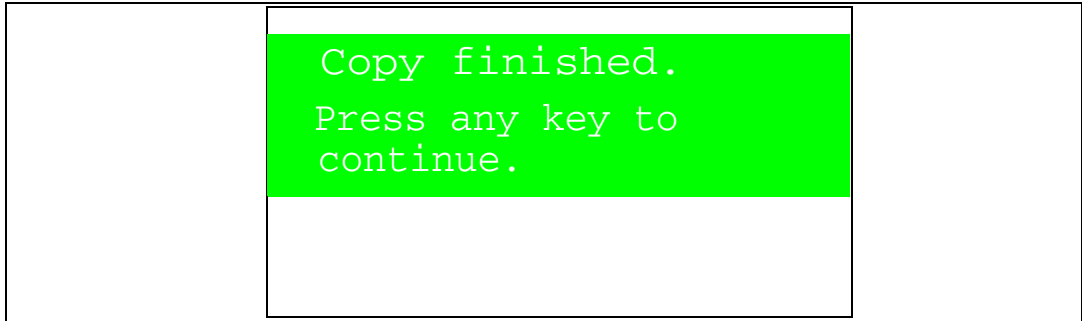
When this submenu is selected, the message shown in [Figure 16](#) is displayed on the LCD.

Figure 16. Copy images to the internal Flash memory



When the copy operation is complete, the message shown in [Figure 17](#) appears on the LCD screen.

Figure 17. Copy images to internal Flash memory complete



Internal Flash -> CPU -> LCD

This submenu uses the CPU to read the data images from the internal Flash and to write them onto the LCD. The two images are displayed at maximum speed (with no delay).

When the **Internal Flash -> CPU -> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

Figure 18. Animation showing the STM32F10xxx display speed

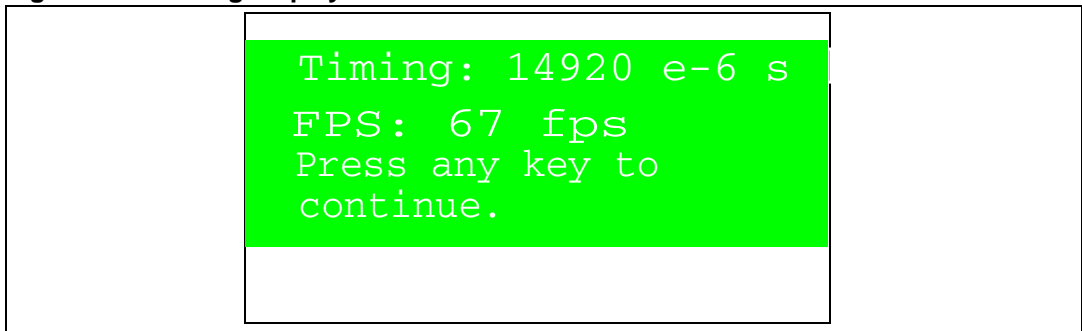


Figure 19. ST logo



If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 20](#) is displayed.

Figure 20. Timing display submenu



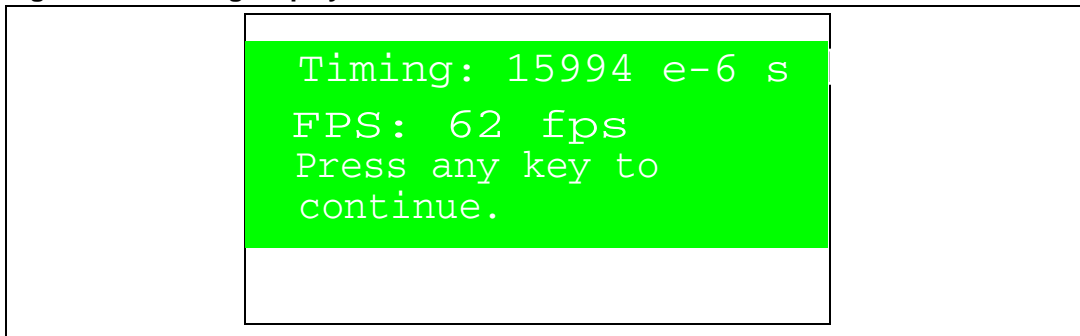
Internal Flash -> DMA -> LCD

This submenu uses the DMA to read the data images from the internal Flash and to write them onto the LCD (0 % CPU load). The two images are displayed at maximum speed (with no delay).

When the **Internal Flash -> DMA-> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 20](#) is displayed.

Figure 21. Timing display submenu



Internal Flash -> LCD Speed

This submenu is used to display, at a controlled speed, the set of two images read from the Internal Flash memory. You can control the display speed of the two images (or frames) through the RV1 trimmer. The maximum delay between the two frame is of about 1 s. It is obtained when RV1 is at its maximum position.

4.3.2 NOR Flash memory

This part of the LCD demo is dedicated to displaying the bitmap pictures stored in the NOR Flash memory, as well as the computed one-frame display time.

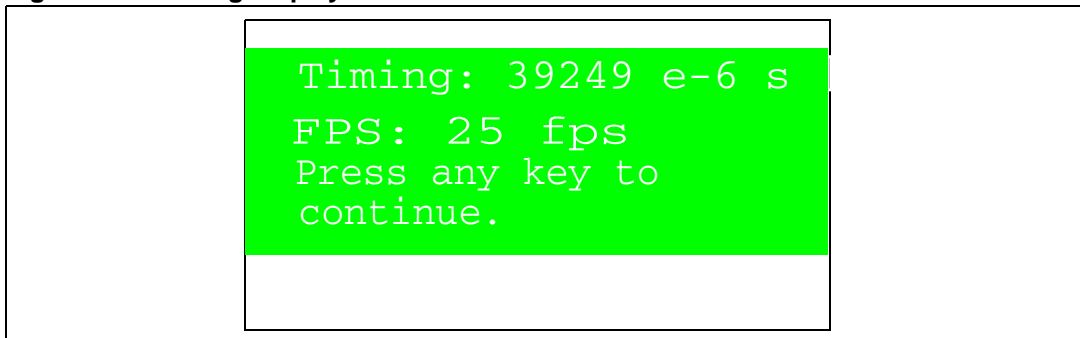
NOR Flash -> CPU -> LCD

This submenu uses the CPU to read the data images from the NOR Flash memory and, to write them onto the LCD. The two images are displayed at maximum speed (with no delay).

When the **NOR Flash -> CPU -> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 22](#) is displayed.

Figure 22. Timing display submenu



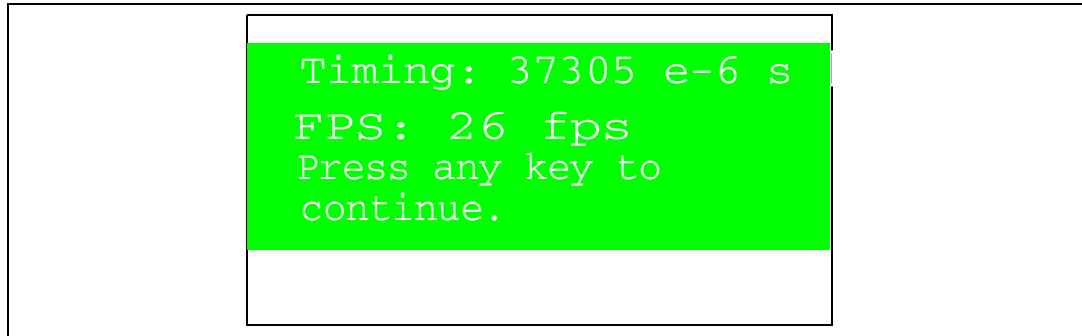
NOR Flash -> DMA -> LCD

This submenu uses the DMA to read the data images from the NOR Flash memory and, to write them onto the LCD (0 % CPU load). The two images are displayed at maximum speed (with no delay).

When the **NOR Flash -> DMA-> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 23](#) is displayed.

Figure 23. Timing display submenu



NOR Flash-> LCD Speed

This submenu is used to display, at a controlled speed, the set of two images read from the NOR Flash memory. You can control the display speed of the two images (or frames) through the RV1 trimmer. The maximum delay between the two frame is of about 1 s. It is obtained when RV1 is at its maximum position.

4.3.3 External SRAM

This part of the LCD demo is dedicated to displaying the bitmap pictures stored in the external SRAM as well as the computed one-frame display time.

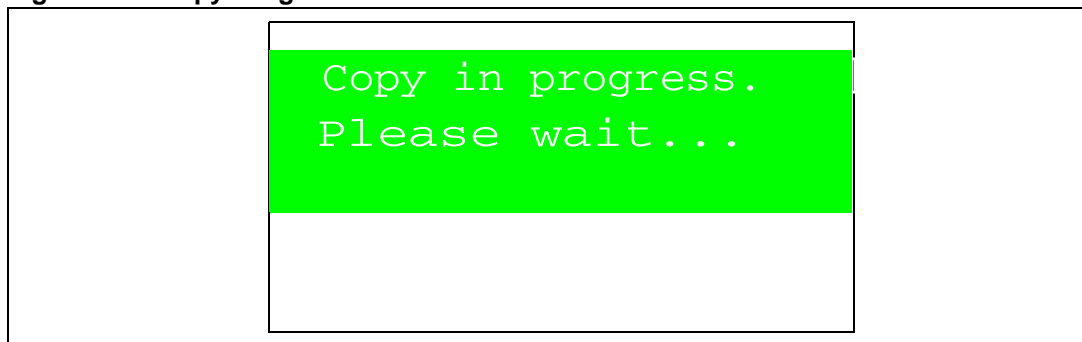
It is first necessary to load the corresponding images into the dedicated external SRAM memory.

Copy images to external SRAM

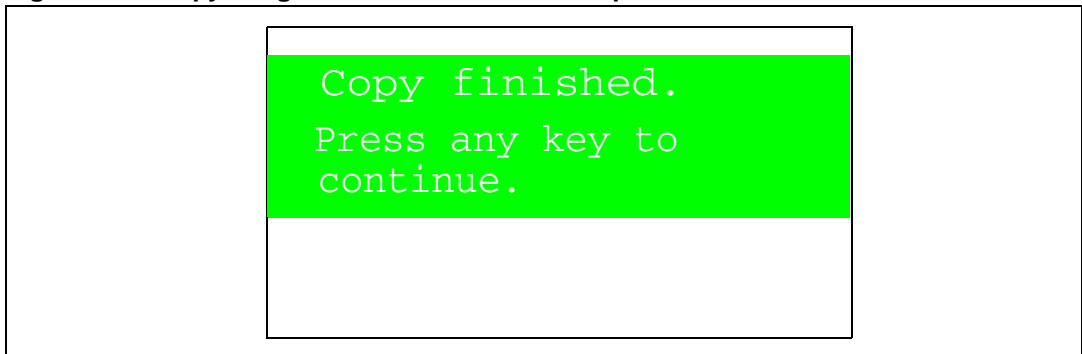
This submenu is used to copy images from the NOR Flash memory to a specified address in external SRAM.

When this submenu is selected the message shown in [Figure 24](#) is displayed on the LCD.

Figure 24. Copy images to external SRAM



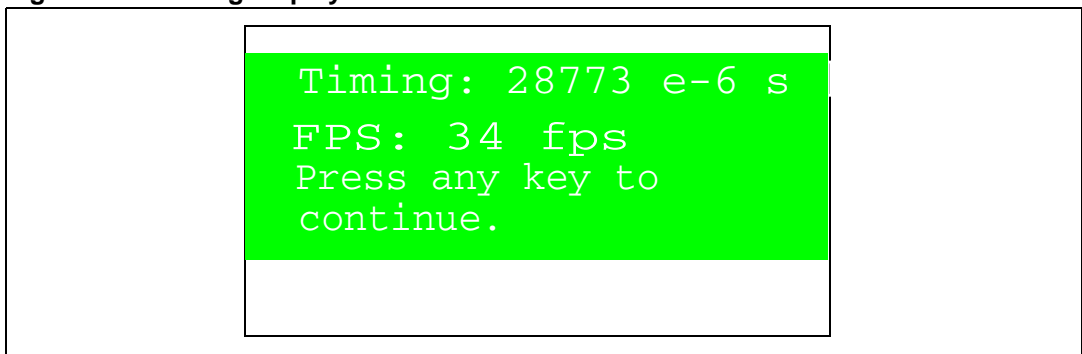
When the copy operation is complete, the message shown in [Figure 25](#) appears on the LCD screen.

Figure 25. Copy images to external SRAM complete**External SRAM-> CPU -> LCD**

This submenu uses the CPU to read the data images from the external SRAM and, to write them on the LCD. The two images are displayed at maximum speed (with no delay).

When the **External SRAM -> CPU -> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

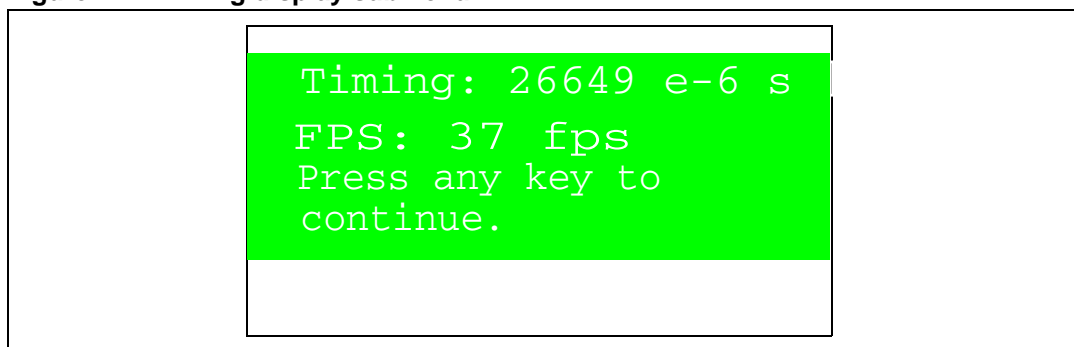
If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 26](#) is displayed.

Figure 26. Timing display submenu**External SRAM -> DMA -> LCD**

This submenu uses the DMA to read the data images from the external SRAM and, to write them onto the LCD (0 % CPU load). The two images are displayed at maximum speed (with no delay).

When the **External SRAM -> DMA-> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 27](#) is displayed.

Figure 27. Timing display submenu**External SRAM-> LCD Speed**

This submenu is used to display, at a controlled speed, the set of two images read from the external SRAM. You can control the display speed of the two images (or frames) through the RV1 trimmer. The maximum delay between the two frame is of about 1 s. It is obtained when RV1 is at its maximum position.

4.3.4 NAND Flash memory

This part of the LCD demo is dedicated to displaying the bitmap pictures stored in the NAND Flash memory, as well as the computed one-frame display time. An internal 16 Kbyte buffer is used for NAND Flash buffer reading.

Bitmap image is read and written to LCD by 16 Kbytes.

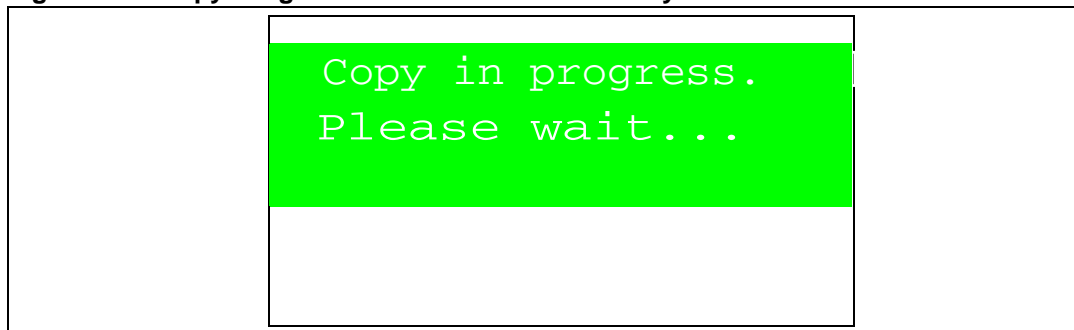
It is first necessary to load the corresponding images into the dedicated NAND Flash memory.

Note: When entering different NAND Flash display menus without first writing the images into the NAND Flash memory, blank images are displayed on the LCD screen if the NAND Flash memory has already been erased.

Copy images to NAND Flash

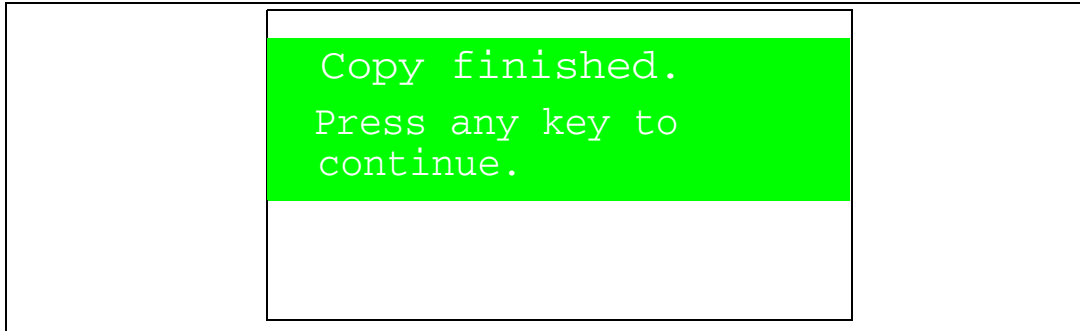
This submenu is used to copy images from the NOR Flash memory to a specified address in NAND Flash memory.

When this submenu is selected the message shown in [Figure 28](#) is displayed on the LCD.

Figure 28. Copy images to the NAND Flash memory

When the copy operation is complete, the message shown in [Figure 29](#) appears on the LCD screen.

Figure 29. Copy images to NAND Flash complete



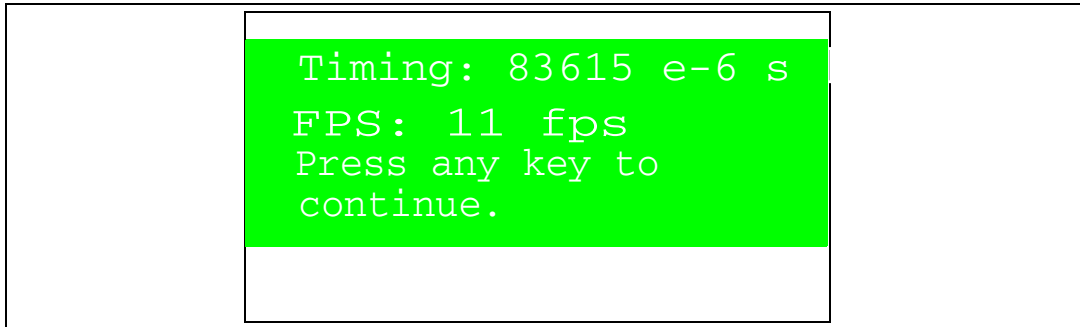
NAND Flash -> CPU -> LCD

This submenu uses the CPU to read the data images from the NAND Flash memory and, to write them onto the LCD. The two images are displayed at maximum speed (with no delay).

When the **NAND Flash -> CPU -> LCD** menu is selected, the animation shown in [Figure 18](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 20](#) and [Figure 19](#) is displayed.

Figure 30. Timing display submenu

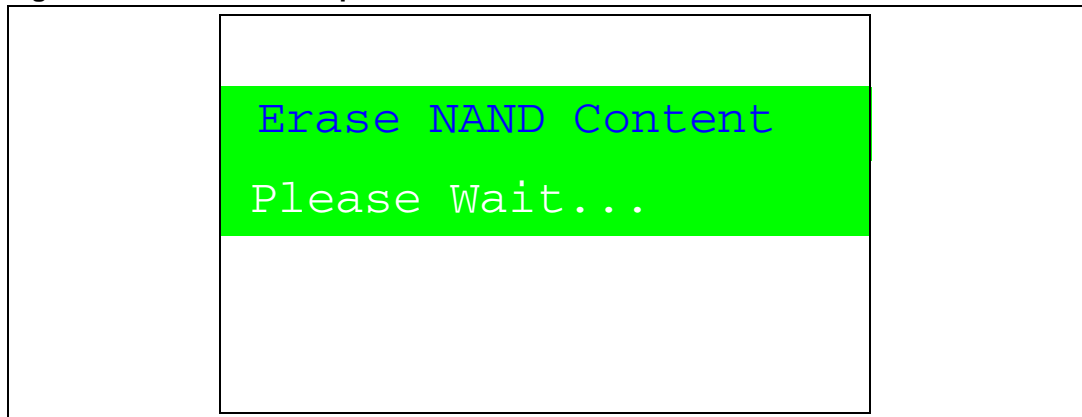


NAND Flash-> LCD Speed

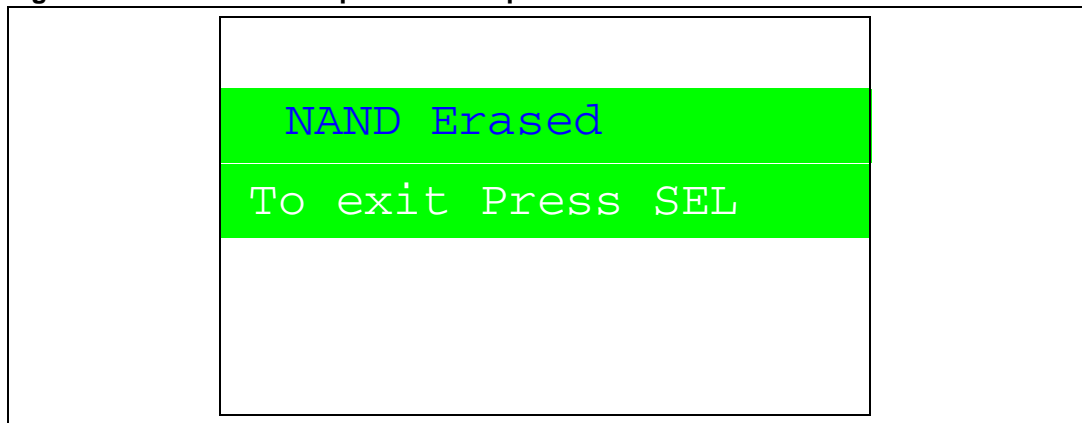
This submenu is used to display, at a controlled speed, the set of two images read from the NAND Flash memory. You can control the display speed of the two images (or frames) through the RV1 trimmer. The maximum delay between the two frame is of about 1 s. It is obtained when RV1 is at its maximum position.

NAND Flash physical format

The submenu is used to erase all the NAND Flash contents. When this submenu is selected, the message shown in [Figure 31](#) is displayed on the LCD.

Figure 31. NAND Erase operation

When the NAND Erase operation is complete, the message shown in [Figure 32](#) is displayed on the LCD.

Figure 32. NAND Erase operation complete

To exit this submenu and return to the **NAND physical format** submenu, press SEL.

4.3.5 SD card

This part of the LCD demo is dedicated to displaying the bitmap pictures stored in the SD card, as well as the computed one-frame display time. An internal 10 Kbyte buffer is used for SD card buffer reading. Bitmap images are read and written to the LCD 10 Kbytes at a time.

It is first necessary to load the corresponding images into the dedicated SD card.

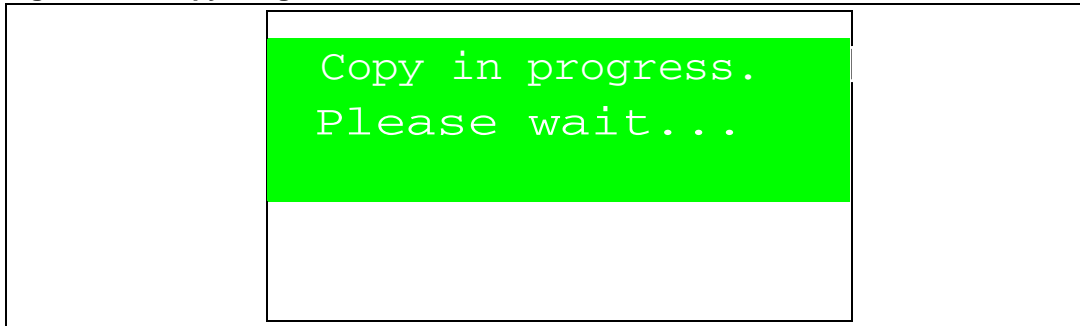
Note: When entering SD card display menus without first writing the images into the SD, blank images are displayed on the LCD screen if the SD card has already been erased.

Copy images to SD card

This submenu is used to copy images from the NOR Flash memory to a specified address in the SD card.

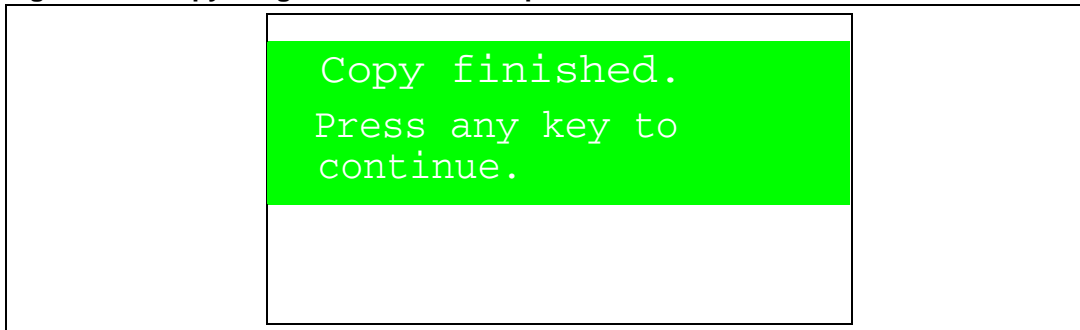
When this submenu is selected, the message shown in [Figure 33](#) is displayed on the LCD.

Figure 33. Copy images to SD card



When the copy operation is complete, the message shown in [Figure 34](#) is displayed on the LCD screen.

Figure 34. Copy images to SD card complete

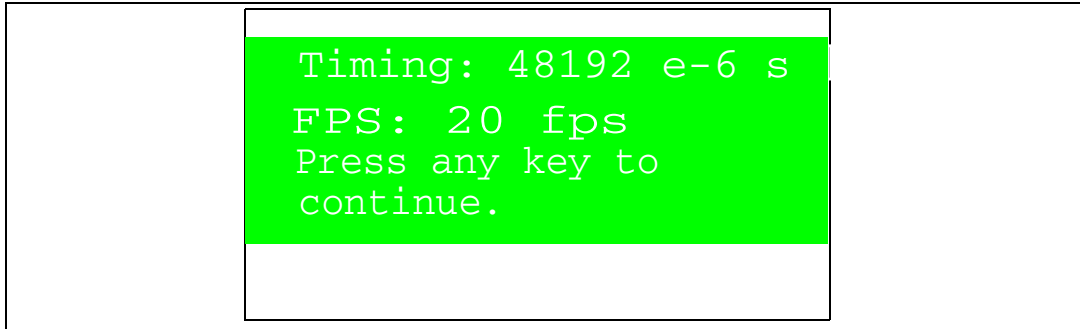


SD card -> CPU -> LCD

This submenu uses the CPU to read the data images from the SD card and, to write them onto the LCD. The two images are displayed at maximum speed (with no delay).

When the **SD card -> CPU -> LCD** menu is selected, the animation shown in [Figure 18](#) and [Figure 19](#) is displayed on the LCD screen.

If you press the SEL push-button while the animation is playing, the animation is stopped and the submenu shown in [Figure 35](#) is displayed.

Figure 35. Timing display submenu**SD card-> LCD Speed**

This submenu is used to display, at a controlled speed, the set of two images read from the SD card. You can control the display speed of the two images (or frames) through the RV1 trimmer. The maximum delay between the two frame is of about 1 s. It is obtained when RV1 is at its maximum position.

5 Color TFT LCD interfacing with the FSMC: performance

Table 10 gives the speeds at which bitmap pictures are displayed from the different memories.

Table 10. LCD display speeds^{(1) (2)}

Memory	Frame size	Resolution	Color depth	Display timing without DMA	Display timing with DMA	Frames per second (FPS)
Internal Flash	150 Kbytes	320 × 240	16 bits	14.920 ms	15.994 ms	67 fps / 62 fps
External SRAM	150 Kbytes	320 × 240	16 bits	28.773 ms	26.649 ms	34 fps / 37 fps
NOR Flash	150 Kbytes	320 × 240	16 bits	39.429 ms	37.305 ms	25 fps / 26 fps
NAND Flash	150 Kbytes	320 × 240	16 bits	83.615 ms	-	11 fps
SD card (1-bit)	150 Kbytes	320 × 240	16 bits	81.257 ms	-	12 fps
SD card (4-bit)	150 Kbytes	320 × 240	16 bits	48.192 ms	-	20 fps

1. The CPU frequency is 72 MHz, FSMC frequency is 36 MHz. The SDIO clock frequency is 24 MHz.
2. The SysTick timer was used to compute the timings, as well as Keil Compiler RVMDK V3.20 - O3, Thumb2 Mode.

In the first case of a transfer from the internal Flash memory, CPU transfer is faster than DMA transfer. This is due to the fact that the internal Flash memory is optimized for code execution. However, it is usually better to use the DMA for the transfer as the CPU is then free to do other jobs in parallel.

Note: *If the CPU is executing code from the internal Flash memory, DMA transfer from the internal Flash memory will be slowed down.*
For the other cases (bitmap picture in external memory), the multilayer bus matrix ensures seamless DMA transfer while the CPU is executing code from the internal Flash memory or the RAM.

6 LCD demo programming

This section explains how to use the DFU (device firmware upgrade) application to program the LCD demonstration application. It also describes the files needed for the STM3210E-EVAL board.

To save programming time using the DFU, proceed as follows:

- Use a certified USB Hub 2.0 high-speed between the STM3210E-EVAL board and your PC
- Minimize the used PC RAM size by closing unused applications
- Use an efficient PC

Note: When using the DFU application, make sure that in the STM3210E-EVAL board, jumper JP14 is in the position 2-3.

6.1 Programming the M29W128 NOR Flash memory

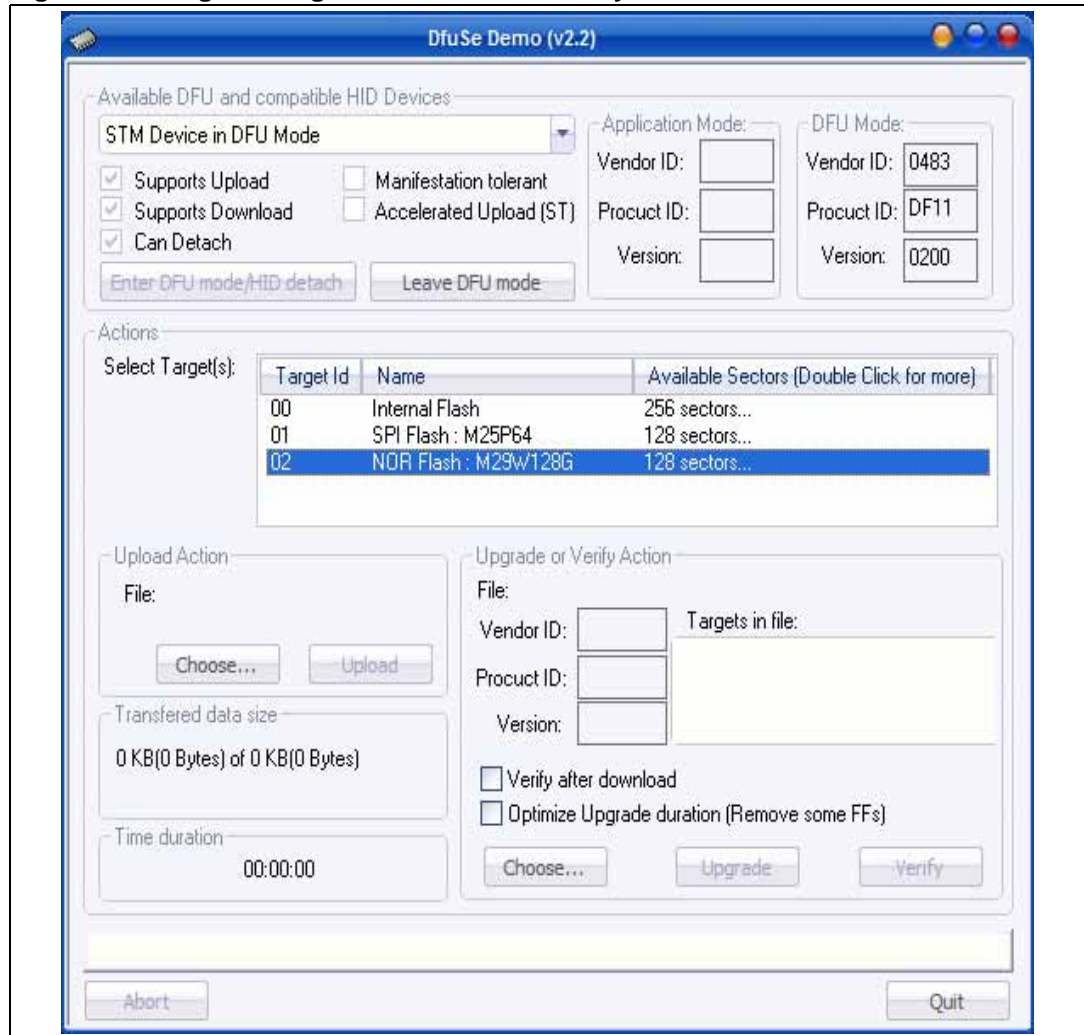
To program the NOR Flash memory, go through the following steps:

1. If you are doing this operation for the first time, load the DFU firmware project using your preferred toolchain and load the DFU image using JTAG.
2. Connect the STM3210E-EVAL board to the PC with a USB cable. If it is the first time you do this, you will be asked to install a driver for the board. The driver is already available in the installation directory.

To enter the DFU mode, hold down the Key push-button on the STM3210E-EVAL board.

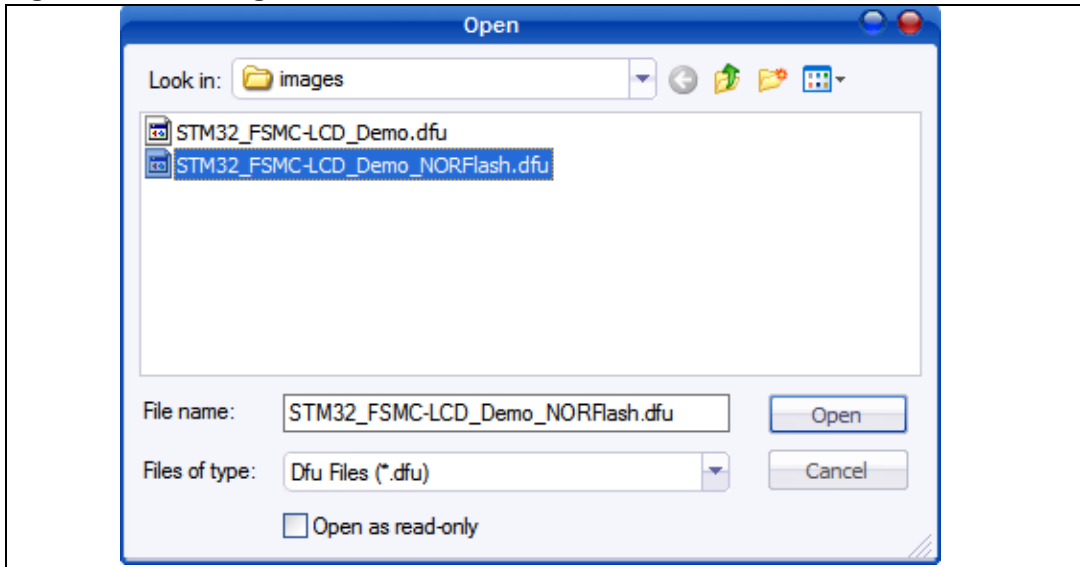
3. Run the *DfuSeDemo.exe* program
4. The window shown in [Figure 36](#) appears. In the **Actions** frame, **Select Target(s)** field, select **NOR Flash: M29W128** and in the **Upgrade or Verify Action** frame, click on the **Choose...** button.

Figure 36. Programming the NOR Flash memory



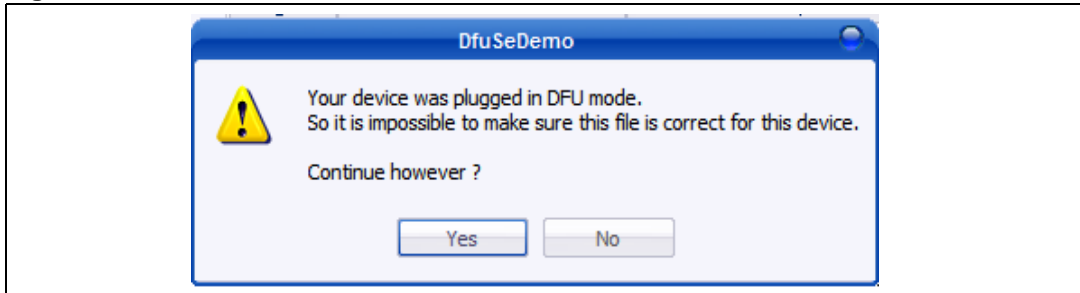
- 5. This causes the window shown in [Figure 37](#) to open. Select the *STM32_FSMC-LCD_Demo_NORFlash.dfu* file.

Figure 37. Selecting the *STM32_FSMC-LCD_Demo_NORFlash.dfu* file



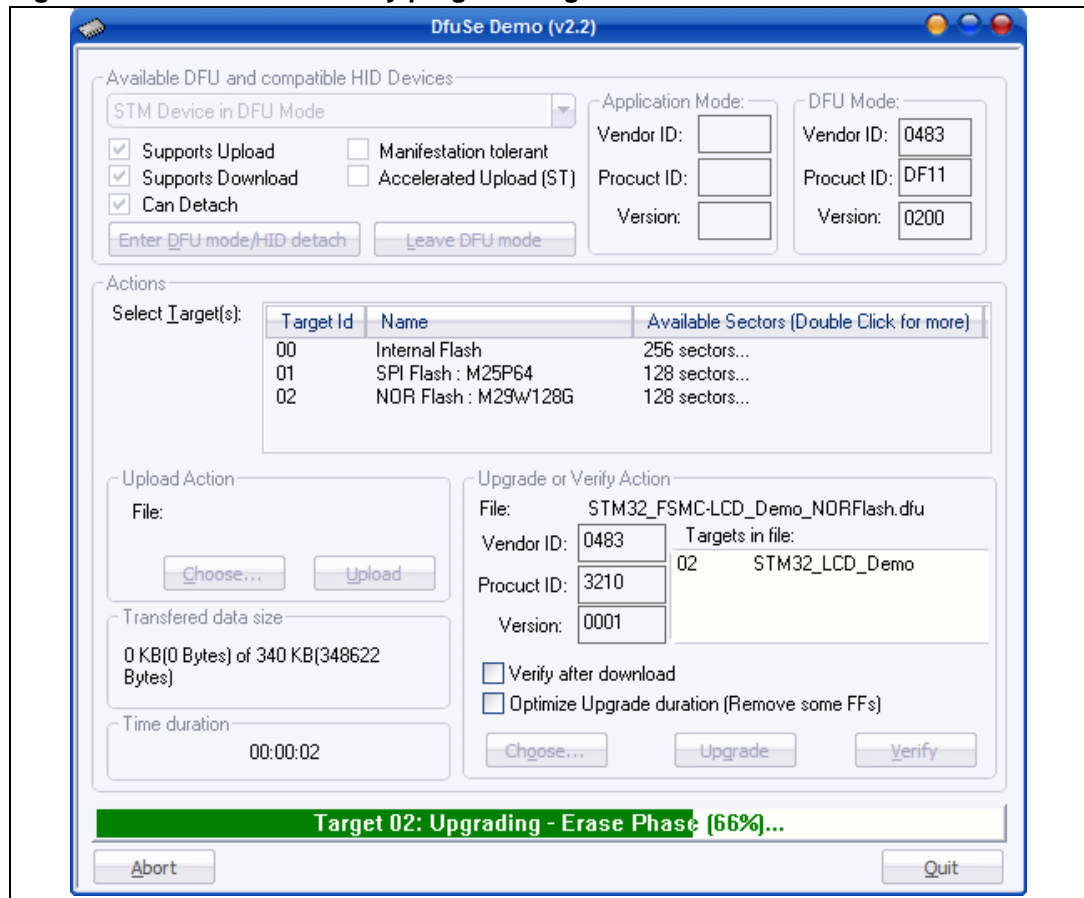
- 6. Then click on the **Open** button. The previous window appears again, click on the **Upgrade** button. A dialog window appears. Select **Yes** to continue (see [Figure 38](#)).

Figure 38. Click Yes to continue



7. The M29W128 NOR Flash programming operation starts as shown in *Figure 39*.

Figure 39. NOR Flash memory programming

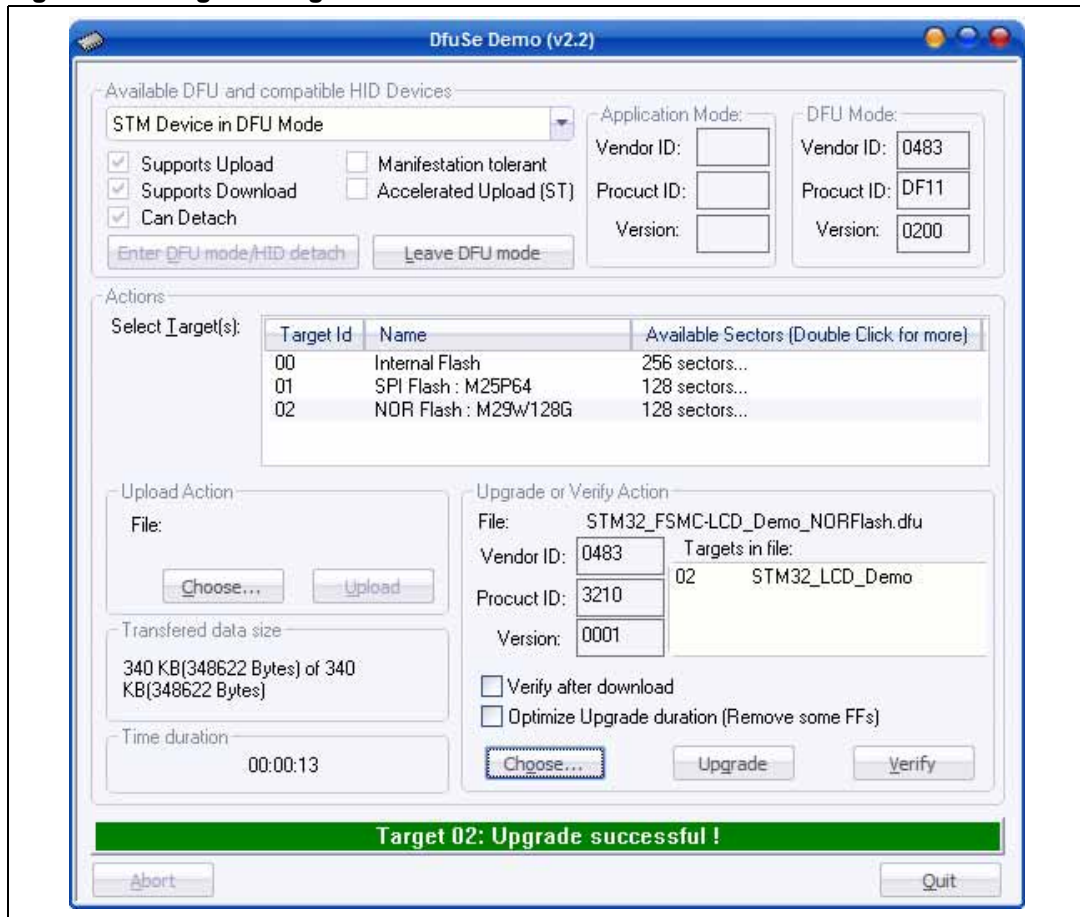


6.2 LCD demo programming

To program the *STM32_FSMC-LCD_Demo.dfu* file in the internal Flash memory, go through the following steps:

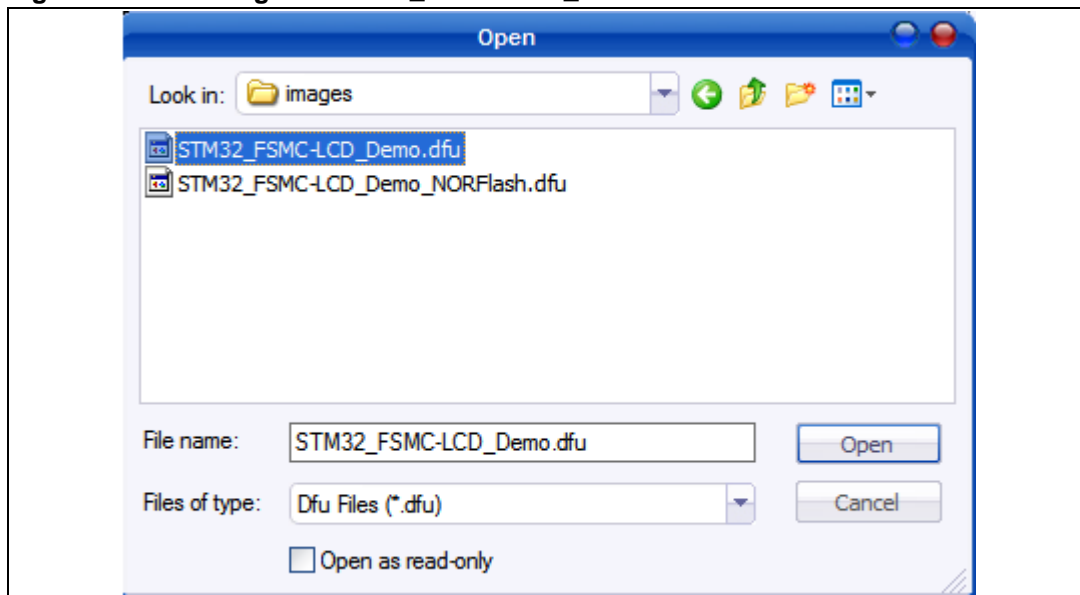
1. When the NOR Flash memory has finished programming, in the **Action** frame (**Select Target(s)**) field, select *Internal Flash* and in the **Upgrade or Verify Action** frame, click on the **Choose...** button.

Figure 40. Programming the demo



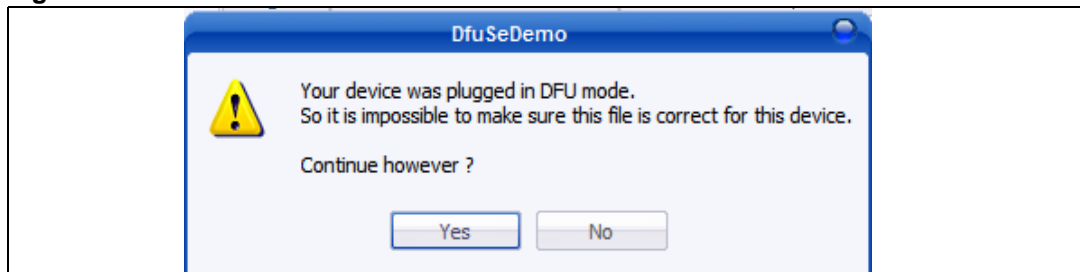
- This causes the window shown in [Figure 41](#) to open. Select the *STM32_FSMC-LCD_Demo.dfu* file, as shown in [Figure 41](#).

Figure 41. Selecting the *STM32_FSMC-LCD_Demo.dfu* file



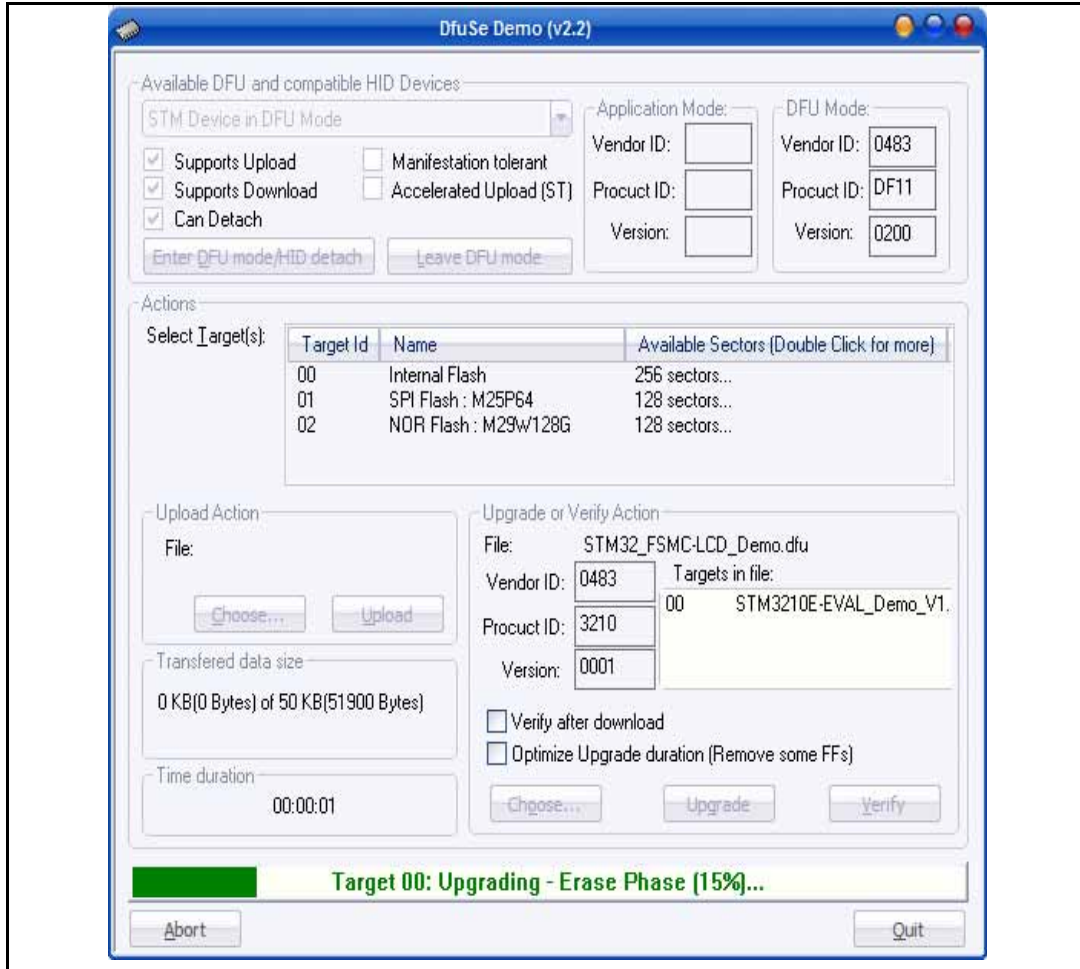
- Then click on the **Open** button. The previous window appears again, click on the **Upgrade** button. A dialog window appears. The dialog window shown in [Figure 42](#) then appears. Select **Yes** to continue.

Figure 42. Click Yes to continue



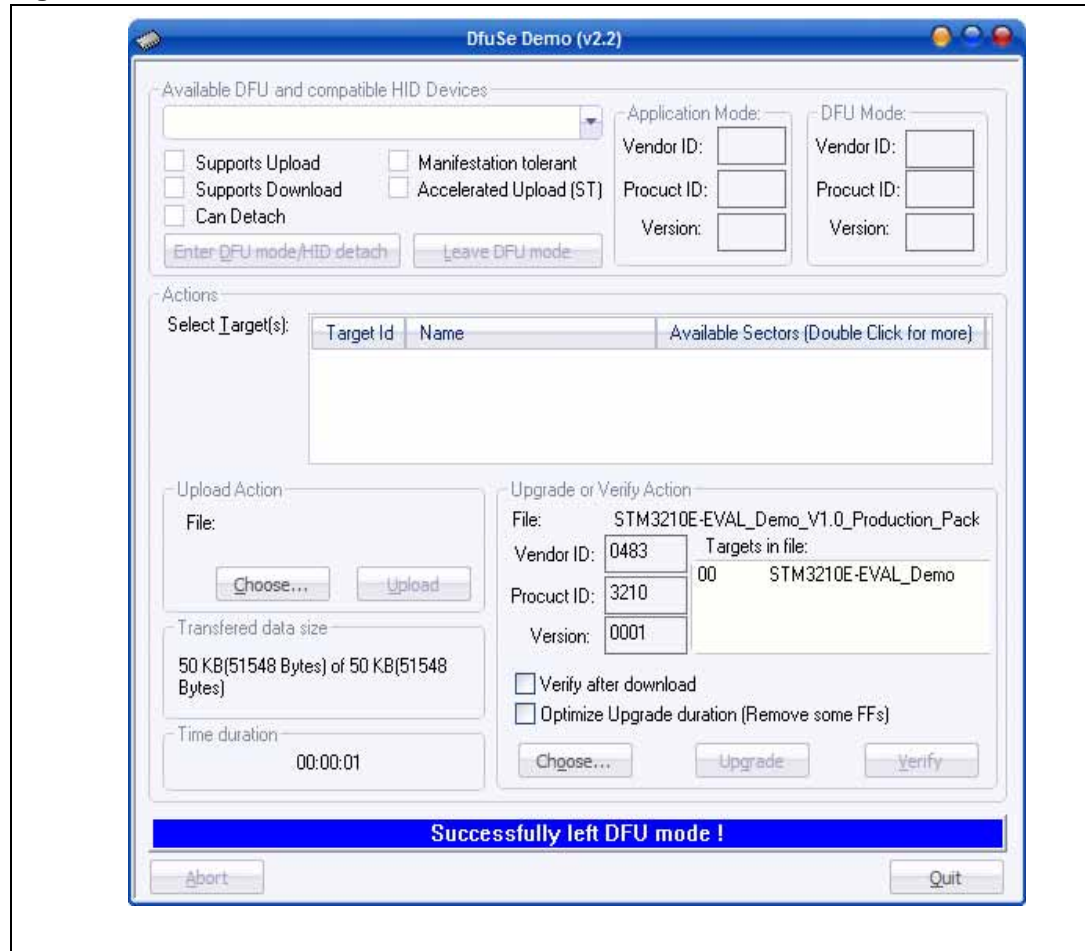
- 4. The STM32F10xxx's internal Flash memory starts programming as shown in [Figure 43](#).

Figure 43. Internal Flash memory programming



- When the programming operation is complete, click on **Leave DFU Mode**. This causes the target to switch from DFU mode to Application mode, as shown in [Figure 44](#). And the demonstration starts.

Figure 44. DFU mode left



7 Conclusion

Due to its performance and flexibility, the FSMC can easily interface color LCDs and serve to build multimedia applications. The FSMC is suitable for displays that require low-power and medium-resolution LCDs. Using the High-density STM32F10xxx FSMC, associated with the MCU performance, signal processing and control function capabilities, to directly interface LCD displays dramatically reduces the application system cost and complexity.

This application note has demonstrated how to create a display application by using the STM32F10xxx FSMC to interface an LCD while using only a small amount of the STM32F10xxx processing power.

8 Revision history

Table 11. Document revision history

Date	Revision	Changes
22-Jul-2008	1	Initial release.
19-Sep-2008	2	Reference to COSMORAM removed. NWAIT signal removed from Figure 1: Asynchronous NOR Flash read access timing and Figure 2: Asynchronous NOR Flash write access timing .

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.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED ST REPRESENTATIVE, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

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.

© 2008 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

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

www.st.com