

2.4-inch SPI Module MSP2402

User Manual - English Translation

Manufacturer: LiangHaoCai

Original document revision: Rev. 1.0

This is an English translation of the supplied Chinese-language manual. Product names, signal names, library names and code identifiers have been preserved.

1. Product Overview

This LCD module uses a 4-wire SPI interface. It is based on the ILI9341 controller, has a resolution of 240 × 320 pixels, and is optionally available with a touch panel. The module includes the LCD, a backlight-control circuit, and a touch-controller circuit.

Product Features

- 2.4-inch color display supporting 16-bit RGB and 65K colors.
- 240 × 320 resolution; optional touch function.
- SPI serial bus - only a few I/O pins are required.
- Built-in SD-card slot for easy expansion and experimentation.
- A wide range of example programs is provided.
- Industrial-grade manufacturing process for stable long-term operation.
- Low-level driver support is available.

Product Specifications

Item	Description
Display colors	RGB, 65K colors
SKU	With touch: MSP2402; without touch: MSP2401
Display size	2.4 inch
Display type	TFT
Controller	ILI9341
Resolution	320 × 240 pixels
Module interface	4-wire SPI
Active display area	36.72 × 48.96 mm
Module dimensions	77.18 × 42.72 mm
Viewing angle	> 60°
Operating temperature	-20°C to +70°C
Storage temperature	-30°C to +80°C
Operating voltage	3.3 V / 5 V
Current consumption	Approx. 90 mA
Weight	Approx. 25 g

2. Interface Description

No.	Module pin	Description
1	VCC	LCD positive supply, 3.3-5 V
2	GND	LCD ground
3	CS	LCD chip-select signal
4	RESET	LCD reset signal
5	DC/RS	LCD register/data selection signal
6	SDI (MOSI)	LCD SPI write-data signal
7	SCK	LCD SPI clock signal
8	LED	Backlight control; high level turns it on. Connect to 3.3 V if control is not required.
9	SDO (MISO)	LCD SPI read-data signal; may be left unconnected if reading is not required.
10	T_CLK	Touch-controller SPI clock
11	T_CS	Touch-controller chip select
12	T_DIN	Touch-controller SPI input
13	T_DO	Touch-controller SPI output
14	T_IRQ	Touch interrupt output

Pins 10-14 are used by the touch panel. They may be left unconnected on modules without touch support or when touch functionality is not required.

3. Hardware Configuration

The module hardware consists of three main sections: the LCD control circuit, the touch-panel control circuit, and the backlight-control circuit.

- The LCD control circuit drives both the LCD control pins and its data-transfer pins.
- The touch-control circuit detects touch events and reads touch coordinates; the touch panel is optional.
- The backlight circuit switches the backlight on and off. If backlight control is not needed, connect the LED pin directly to 3.3 V.

4. Operating Principle

ILI9341 Controller

The ILI9341 supports a maximum resolution of 240×320 pixels and contains 172,800 bytes of GRAM. It supports 8-, 9-, 16- and 18-bit parallel buses as well as 3-wire and 4-wire SPI. Because a parallel interface requires many I/O pins, SPI is generally the preferred option.

The controller supports 65K- and 262K-color RGB modes, display rotation, scrolling and video display. In the mode used here, each pixel is represented by a 16-bit RGB565 value. Pixel addressing follows row and column order; the increment/decrement direction depends on the selected scan mode. To display data, first set the target address and then write the color value.

SPI Communication Protocol

- CSX is the slave-select signal. The controller is enabled only while CSX is low.
- D/CX selects command or data mode: low = command, high = data.
- SCL is the SPI clock. One bit is transferred on each rising edge.
- SDA carries 8-bit SPI data, most-significant bit first.
- The examples use SPI mode 0: CPOL = 0 and CPHA = 0.

5. Arduino Wiring

Arduino UNO

Module pin	Arduino UNO pin
SDO (MISO)	12
LED	A0
SCK	13
SDI (MOSI)	11
DC/RS	A3
RESET	A4
CS	A5
GND	GND
VCC	5 V / 3.3 V
T_IRQ	6
T_DO	4
T_DIN	5
T_CS	2
T_CLK	3

Arduino Mega 2560

Module pin	Arduino Mega 2560 pin
SDO (MISO)	50
LED	A0
SCK	52
SDI (MOSI)	51
DC/RS	A3
RESET	A4
CS	A5
GND	GND
VCC	5 V / 3.3 V
T_IRQ	49
T_DO	47
T_DIN	48
T_CS	45
T_CLK	46

6. C51 Wiring

STC89C52RC and STC12C5A60S2 test-program wiring

Module pin	STC89 / STC12 board pin
VCC	5 V / 3.3 V
GND	GND
CS	P1.3
RESET	P3.3
DC/RS	P1.2
SDI (MOSI)	P1.5
SCK	P1.7
LED	P3.2
SDO (MISO)	P1.6
T_CLK	P3.6
T_CS	P3.7
T_DIN	P3.4
T_DO	P3.5
T_IRQ	P4.0

7. STM32 Wiring

Pin locations and available expansion headers differ between development boards. For convenient wiring, the example pin assignments therefore vary by board.

STM32F103RCT6

Module pin	MiniSTM32 / STM32F103RCT6
VCC	5 V / 3.3 V
GND	GND
CS	PB11
RESET	PB12
DC/RS	PB10
SDI (MOSI)	PB15
SCK	PB13
LED	PB9
SDO (MISO)	PB14
T_CLK	PC0
T_CS	PC13
T_DIN	PC3
T_DO	PC2
T_IRQ	PC10

STM32F103ZET6

Module pin	Elite STM32 / STM32F103ZET6
VCC	5 V / 3.3 V
GND	GND
CS	PB11
RESET	PB12

Module pin	Elite STM32 / STM32F103ZET6
DC/RS	PB10
SDI (MOSI)	PB15
SCK	PB13
LED	PB9
SDO (MISO)	PB14
T_CLK	PC0
T_CS	PC13
T_DIN	PC3
T_DO	PC2
T_IRQ	PC10

STM32F407ZGT6

Module pin	Explorer STM32F4 / STM32F407ZGT6
VCC	5 V / 3.3 V
GND	GND
CS	PB15
RESET	PB12
DC/RS	PB14
SDI (MOSI)	PB5
SCK	PB3
LED	PB13
SDO (MISO)	PB4
T_CLK	PB0
T_CS	PC5
T_DIN	PF11
T_DO	PB2
T_IRQ	PB1

STM32F429IGT6

Module pin	Apollo STM32F4/F7 / STM32F429IGT6
VCC	5 V / 3.3 V
GND	GND
CS	PD11
RESET	PD12
DC/RS	PD5
SDI (MOSI)	PF9
SCK	PF7
LED	PD6
SDO (MISO)	PF8
T_CLK	PH6
T_CS	PI8
T_DIN	PI3
T_DO	PG3
T_IRQ	PH11

8. Software Description

Arduino Code Architecture

The Arduino test software consists of the LCDWIKI libraries and application code. The library set contains LCDWIKI_SPI, LCDWIKI_GUI and LCDWIKI_TOUCH. The application contains several examples, each testing different functions.

- LCDWIKI_SPI is the low-level, hardware-dependent layer. It initializes the hardware, transfers commands and data, sets pixel coordinates and colors, and configures display modes.
- LCDWIKI_GUI is the middle layer. It uses the low-level API to draw graphics and display characters.
- LCDWIKI_TOUCH is the low-level touch library. It handles touch interrupts, data sampling, A/D conversion and touch-data transfer.
- The application examples use the APIs provided by the LCDWIKI libraries.

C51 and STM32 Code Architecture

- The demo API called by the main program is located in the test code.
- LCD initialization and related operations are located in the LCD code.
- Drawing points, lines, shapes, and displaying Chinese/English characters are handled by the GUI code.
- The main function starts and controls the application.
- Platform code differs between target platforms.
- Touch operations are located in the touch code.
- SPI initialization and configuration are located in the SPI code.
- Key handling is located in the key code; the C51 platform has no key-handling code.
- LED configuration is located in the LED code.

Software SPI vs. Hardware SPI

Example code is provided for both software SPI and hardware SPI, except for the STC89C52RC, which does not provide hardware SPI. The displayed content is the same in both versions, but:

- Hardware SPI is significantly faster.
- With software SPI, all control pins must be defined and may be assigned to any free GPIO.
- With hardware SPI, the clock and data pins are fixed by the platform; the remaining control pins may be assigned to free GPIO pins.
- Software SPI initialization only requires GPIO initialization. Hardware SPI additionally requires configuration of control and data registers.

9. Modifying GPIO Definitions

Arduino

The LCD and touch GPIO definitions are placed separately in each application example, so every example can define its own pins. Touch GPIO definitions are present only in examples that use the touch function.

C51

LCD-related GPIO definitions are located in `lcd.h`. With software SPI, all pin definitions may be changed. With hardware SPI, `LCD_BL`, `LCD_RS`, `LCD_CS` and `LCD_RST` may be reassigned, while `LCD_CLK` and `LCD_SDI` do not need to be defined. Touch-related GPIO definitions are located in `touch.h` and may be assigned to any free GPIO. If the MCU has no P4 GPIO bank, assign `penirq` to another GPIO.

STM32

Non-SPI LCD GPIO definitions are located in `lcd.h` and may be changed to any free GPIO. SPI GPIO definitions are located in `spi.h`. With software SPI, all pins may be reassigned. With hardware SPI, these definitions are not required; also remove SCLK, MISO and MOSI initialization from the `LCD_GPIOInit` function in `lcd.c`.

10. SPI Communication Implementation

Arduino

Hardware SPI is provided by the platform. The application only needs to configure registers and call the relevant functions. Refer to the MCU documentation for details. Software SPI is implemented in LCDWIKI_SPI.cpp in the LCDWIKI_SPI library.

For each transmitted bit, the data pin is driven high for 1 or low for 0. One byte is transferred at a time, most-significant bit first, with one bit transferred on each rising edge of the clock.

C51 and STM32

Hardware SPI is likewise provided by the MCU peripheral. Software SPI is implemented in lcd.c and spi.c. Its operating principle is the same as in the Arduino version: one byte at a time, MSB first, one bit per rising clock edge.

11. Touch-Screen Calibration

Arduino

Run the touch_screen_calibration example and follow the on-screen instructions. After successful calibration, copy the displayed calibration parameters into cali_para.h in the LCDWIKI_TOUCH library.

C51

Run the Touch_Adjust test item. This option is available only in the STC12C5A60S2 test program. After successful calibration, save the displayed parameters in touch.c.

STM32

The STM32 touch-test program can automatically detect whether calibration is required, or calibration may be started manually with a key. The calibration flag and parameters are stored in AT24C02 memory and loaded when needed.

Calibration flow

- Enter the touch test.
- Read the calibration flag from AT24C02.
- If no valid calibration exists, start automatic calibration.
- If valid calibration exists, enter the touch test directly.
- If touch accuracy is insufficient, press KEY0 to recalibrate.
- Repeat calibration until it passes, then save the parameters to AT24C02.

12. Commonly Used Software

The examples display Chinese and English characters, symbols and images, so bitmap-conversion software is required. The manual refers to two tools: Image2Lcd and PCtoLCD2002.

PCtoLCD2002 settings

- Dot-matrix format: negative code.
- Extraction mode: row by row.
- Bit order: forward, most-significant bit first.
- Output format: hexadecimal.
- Custom format: C51.

Image2Lcd settings

Configure horizontal scanning from left to right and top to bottom, with the least-significant bit first.

End of translated manual.