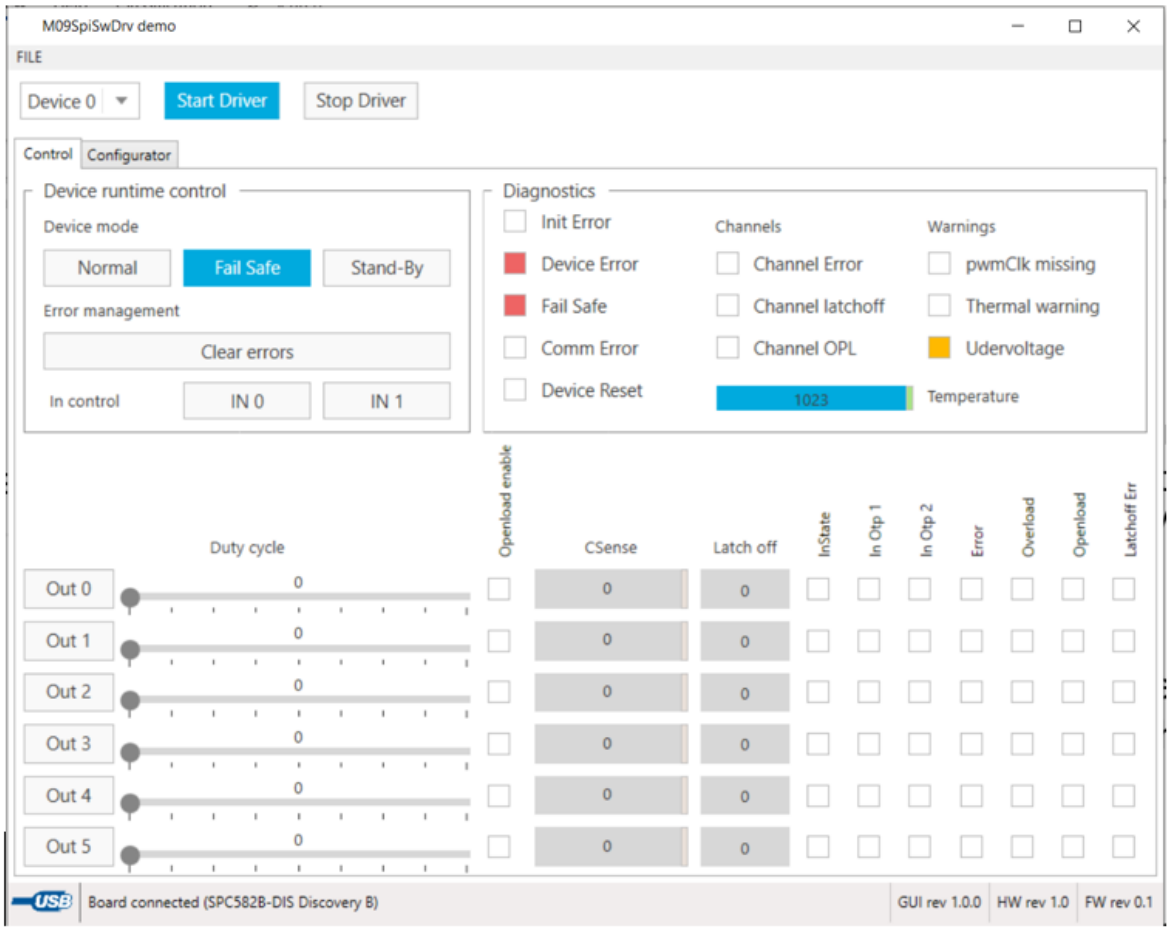
Graphical user interface (GUI) dedicated to set and control the M09SPI

Introduction

The STSW-EV-M09SPI is the graphical user interface (GUI) dedicated to set and control the 12 V M09SPI devices assembled in the corresponding evaluation EV-M09SPI device boards. The STSW-EV-M09SPI has been developed by using Visual Studio/C sharp and it works with the EV-SPC582B programmed with a dedicated firmware.

Figure 1. STSW-EV-M09SPI graphical user interface



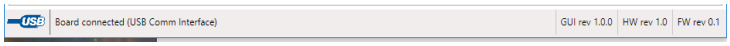
1 Get software

Search on www.st.com, STSW-EV-M09SPI and in the “Tools & Software” section. To get the software (GUI + Firmware) follow the procedure below.

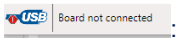
2 Status strip

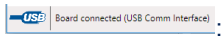
The status trip is shown in the below figure:

Figure 2. Status trip



The arrow icons show interface status between FTDI and GUI:

- 

: Board not connected or FTDI drivers not installed
- 

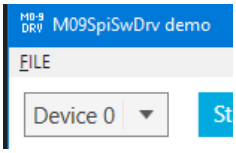
: Normal application operation (communication between MCU and GUI correctly established (configured FTDI descriptor or COM port shown)

3 Main control

3.1 Device selector

Count of devices visible in the combo box selector is stored in the XML formatted configuration file “configuration.xml”.

Figure 3. Device selector combo box

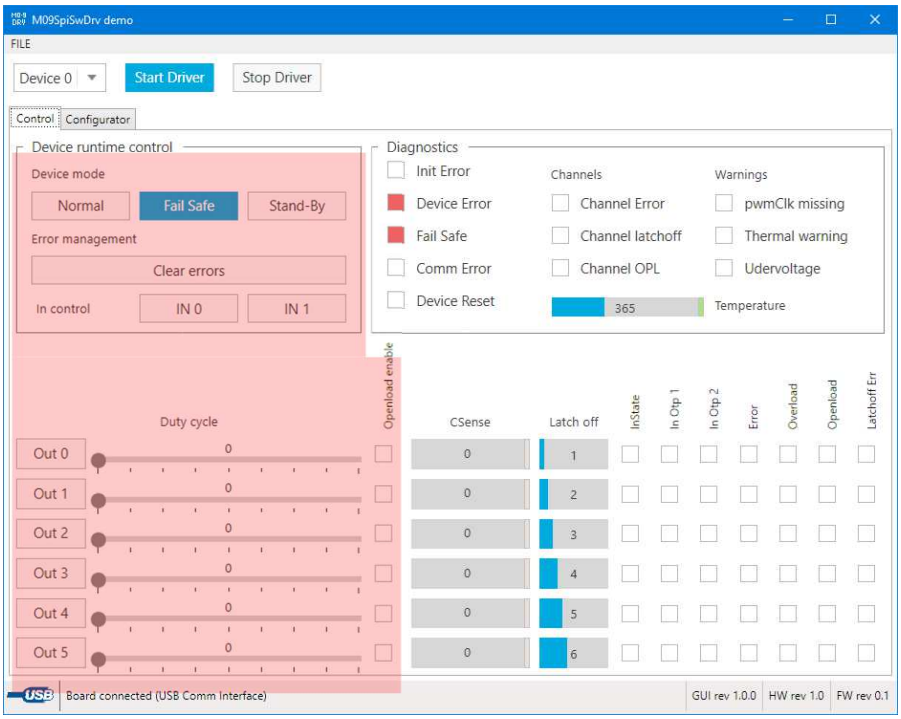


3.2 Tab control

This tab shows device features applicable when the software driver is started (applied by the GUI button “*Start Driver*”), initiates the action *M09Spi_Init()*, *M09Spi_WriteConfig()*. It gives possibility to apply changes of device mode initiates calling *M09Spi_SetDeviceMode()*, adjust channels runtime controls, for example On/Off state, duty cycle, control output “off-state diagnostic” feature by initiating action *M09Spi_SetOutputs()*. Last control of software driver applies clearing of software driver errors grabbed during its runtime operation, function call *M09Spi_ClearErrors()*.

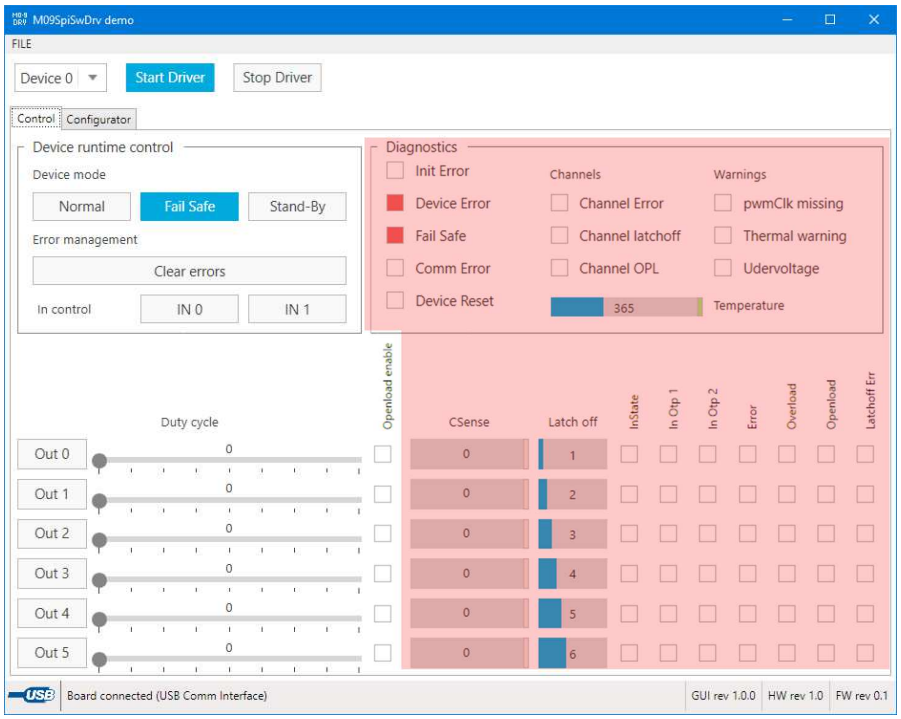
There are two buttons capable to control IN GPIOs connected to the device (these are not a part of the M09SPI software driver, these are driven by the DIO drivers).

Figure 4. Tab control - part 1



Second part of the tab shows errors reported by the software driver, grouped to device related and per channel particular status flags.

Figure 5. **Tab control - part 2**



All status flags are latched by the software driver until they are cleared by calling *M09Spi_ClearErrors()* done in the application layer of firmware triggered by the button *"Clear errors"*.

Similar (latched flag) approach is applied to update status flags of ADC data (ADCSRx) content. ADC updated value (UPDTSRx bit) is signaled in the GUI by the green stripe next to the bar with ADC value.

Status flag is cleared by calling *M09Spi_GetDiag()*, issued by GUI periodic readings of device and channels diagnostic.

Main M09SPI driver function *M09Spi_Run()* is running in the firmware autonomously, without user interaction.

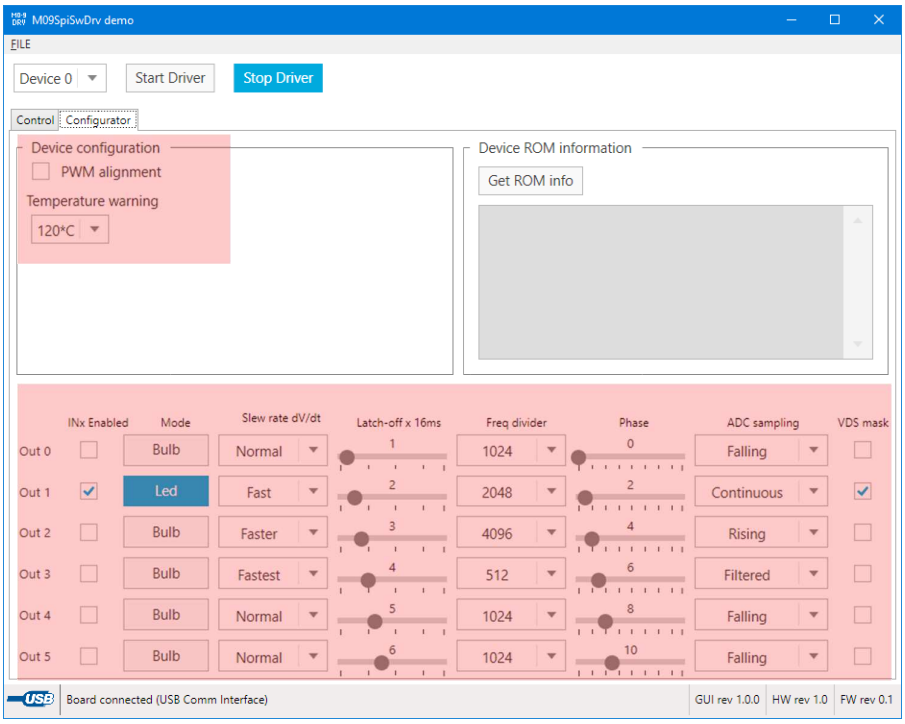
Triggering of timeout watchdog and diagnostic reading is driven by the firmware autonomously too, the timer periodically calls the API function *M09Spi_TriggerWdAndDiag()* (every 10 ms).

3.3 Tab configurator

This tab shows device features applicable when the software driver is stopped (applied by the GUI button "*Stop Driver*" - initiates the calling function *M09Spi_Deinit*. It gives a possibility to apply changes of device related configuration (for example temperature warning, PWM alignment), adjust channels related configuration parameters:

- Output controllability by INx during normal mode
- Output mode (CCR)
- Slew rate
- Latch-off timeout
- Output PWM frequency division factor (of PWMClk input)
- PWM phase shift
- ADC sampling configuration
- Masking of VDS error

Figure 6. **Tab configurator**

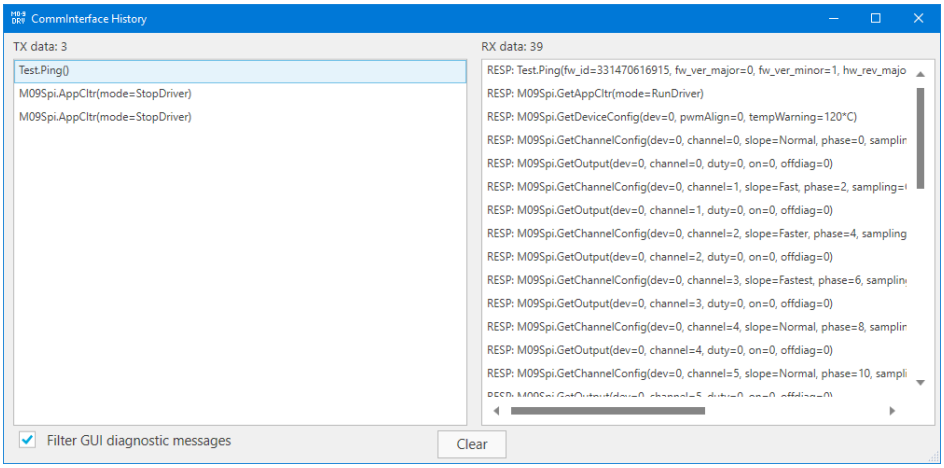


The “Get ROM info” button issues reading of device ROM content is applied by the action *M09Spi_GetDeviceInformation()*; this action should be applied after the driver is activated (“*Start driver*”), otherwise content is 0, reading of MCU RAM not filled yet with device ROM data.

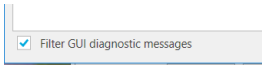
4 Communication history form

The communication history form displays communication frames applied over the GUI.

Figure 7. Communication history form



Available actions:

- Periodic GUI requests for updating status information are normally hidden (but can be enabled by unchecking the checkbox 

- The 

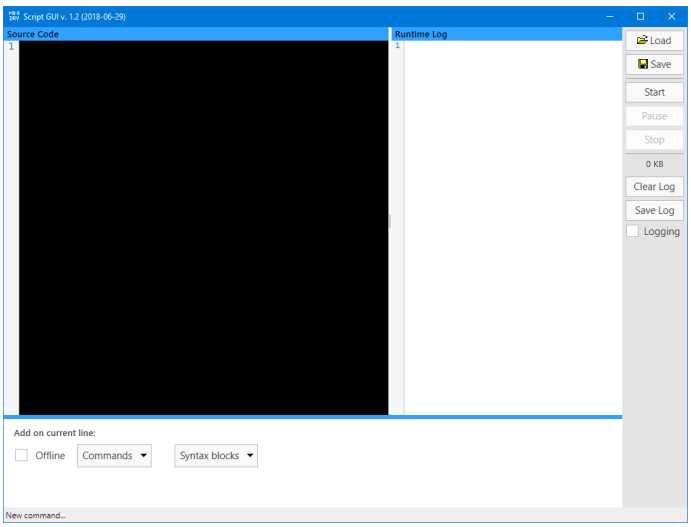
button erases all communication history from the log window.

Note: Due to speed reasons, only the last 100 messages are depicted kept once reaching this limit. The oldest communication messages are discarded.

5 Online command editor

This form can apply user defined sequence of user specified actions (Tx), eventually evaluation of application response messages (Rx).

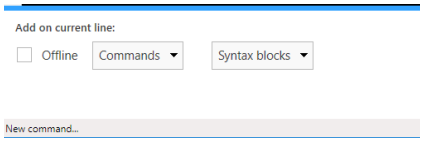
Figure 8. Online command editor



The steps to use the online command editor are the following:

1. Once the window is open user actions can be entered by selecting a particular action in the command selector part of the window:

Figure 9. Command selector



2. Once a requested command is applied, its interpretation is filled into the entry:

Figure 10. M09SPI driver

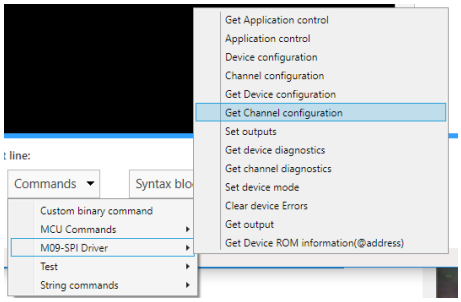
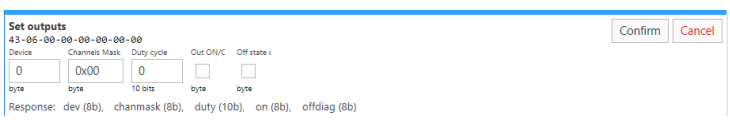


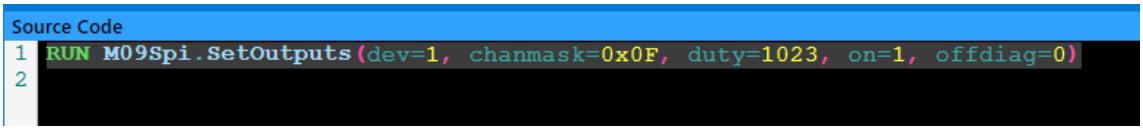
Figure 11. Set outputs



3. User applies command parameters (for example device ID, channels mask, etc.).

4. Filled command must be approved (button “Confirm”) to pass command to user defined sequence (black text box part):

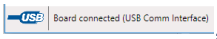
Figure 12. Source code



5. User can modify commands in the text box by his needs (parameters and values must be valid in order to later on successfully run the script).
6. User commands list can be reorganized according to needs cut/paste actions are implemented in the text box with commands.
7. Additional actions like “*Loop*”, “*Goto*”, “*Break*” can be also inserted by “*Syntax blocks*” items.
8. Finished script can be saved or reloaded (buttons “*Save*”, “*Load*”).
9. Script can be passed to the firmware by the action “*Start*”.
10. All communication applied to/from firmware is shown in “*Runtime Log*” part of the window.
11. Log content (after test sequence finish) can be stored or cleared applying the actions “*Save Log*”, or “*Clear Log*”.
12. If there is no need of communication data logging, the feature can be disabled by the checkbox “*Logging*”.

6 Troubleshooting

The following list shows all the requirements for the proper functioning of the GUI:

1. GUI
   - Verify all files related to GUI are present
   - Verify .NET v4.5 installed (if not installed, GUI should request to download it)
   - Verify that FTDI device is recognized by operating system (drivers are installed)
2. Microcontroller operation
   LED_A should blink in normal conditions on the universal board when powered. If not, verify the following points:
   - Board power supply active and connected–signaled by LED1 (5 V) and LED2 (3.3 V)
   - Jumper JP2 is present (3.3 or 5 V selection)
   - If LED_A still not blinking, there is an issue with the firmware in the microcontroller or with the power supply part
3. Communication with GUI
   - Correctly established connection between GUI and MCU application is shown in the GUI status 

, Rx, and Tx LEDs on the USB communication board are blinking together in the interval ~200 ms
   - When GUI signaling no connection it needs to verify that:
     - Board is powered and jumpers are at the correct positions
     - USB cable is connected
     - FTDI drivers are installed properly (in the “*Device manager*” window); the correctly installed driver is signaled by the “*USB ok*” LED on the USB communication board
     - If only Tx LED is blinking - no operation/response from MCU comes
       - Try to restart firmware with reset button (SP3)
       - If the LEDs are still not blinking together, there is an issue with UART communication (hardware issue)
4. M09 device operation
   Verify connected:
   - JP1 - “VDD”, JP22 position “REV_NET”, jumpers control device VDD power supply
   - SPI related jumpers JP11, JP12, JP13, JP15
   - INx control related jumpers JP8, JP9
   - Optional JP14 VBAT jumper connecting VCC power supply between motherboard and daughterboard

Revision history

Table 1. **Document revision history**

Date	Revision	Changes
20-Oct-2022	1	Initial release.

Contents

List of figures