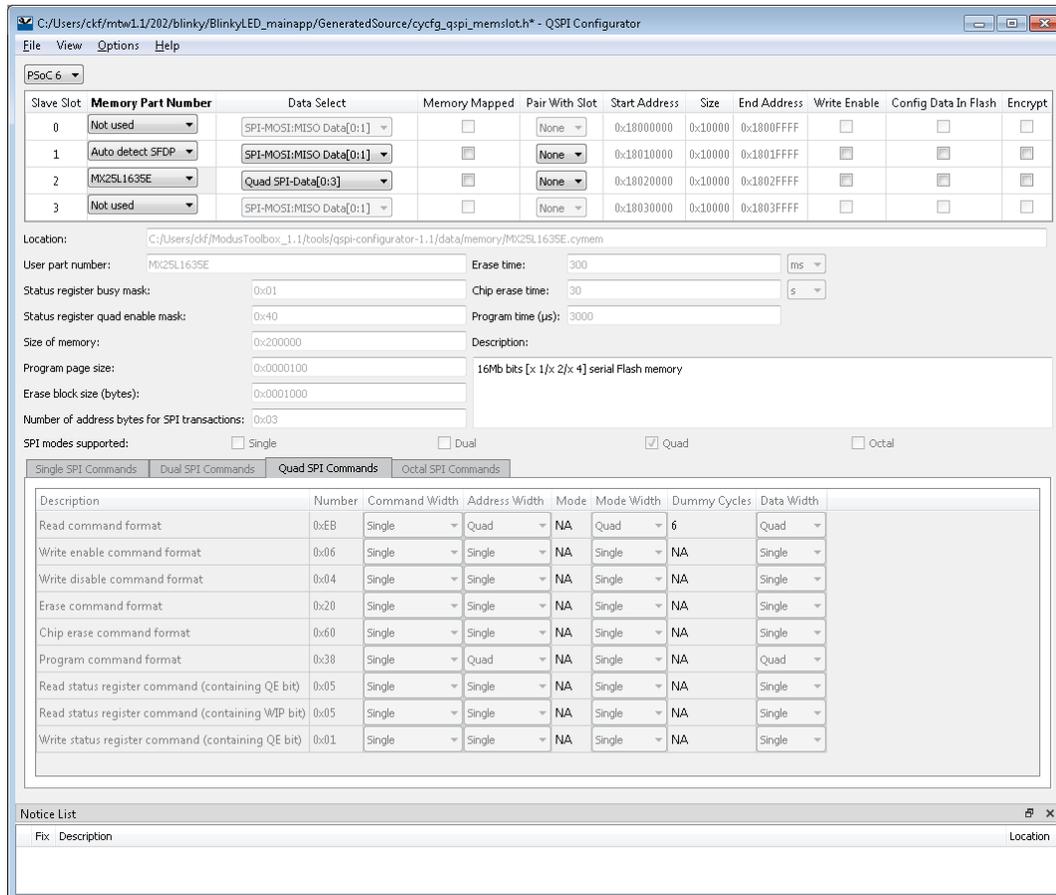


Overview

The Quad Serial Peripheral Interface (QSPI) Configurator is part of a collection of tools included in the ModusToolbox software. Use the QSPI Configurator to open or create configuration files, configure memory slots, and generate code for your application. The QSPI Configurator is a stand-alone tool. Use the top area for configuring memories; the read-only area below it displays information about the selected memory part number.



Slave Slot	Memory Part Number	Data Select	Memory Mapped	Pair With Slot	Start Address	Size	End Address	Write Enable	Config Data In Flash	Encrypt
0	Not used	SPI-MOSI:MISO Data[0:1]	<input type="checkbox"/>	None	0x18000000	0x10000	0x1800FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
1	Auto detect SFDP	SPI-MOSI:MISO Data[0:1]	<input checked="" type="checkbox"/>	None	0x18010000	0x10000	0x1801FFFF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
2	MX25L1635E	Quad SPI-Data[0:3]	<input checked="" type="checkbox"/>	None	0x18020000	0x10000	0x1802FFFF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
3	Not used	SPI-MOSI:MISO Data[0:1]	<input type="checkbox"/>	None	0x18030000	0x10000	0x1803FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Location: C:/Users/ckf/ModusToolbox_1.1/tools/qspi-configurator-1.1/data/memory/MX25L1635E_cyem

User part number: MX25L1635E Erase time: 300 ms

Status register busy mask: 0x01 Chip erase time: 30 s

Status register quad enable mask: 0x40 Program time (µs): 3000

Size of memory: 0x200000 Description: 16Mb bits [x 1/x 2/x 4] serial Flash memory

Program page size: 0x0000100

Erase block size (bytes): 0x0001000

Number of address bytes for SPI transactions: 0x03

SPI modes supported: Single Dual Quad Octal

Description	Number	Command Width	Address Width	Mode	Mode Width	Dummy Cycles	Data Width
Read command format	0xEB	Single	Quad	NA	Quad	6	Quad
Write enable command format	0x06	Single	Single	NA	Single	NA	Single
Write disable command format	0x04	Single	Single	NA	Single	NA	Single
Erase command format	0x20	Single	Single	NA	Single	NA	Single
Chip erase command format	0x60	Single	Single	NA	Single	NA	Single
Program command format	0x38	Single	Quad	NA	Single	NA	Quad
Read status register command (containing QE bit)	0x05	Single	Single	NA	Single	NA	Single
Read status register command (containing WIP bit)	0x05	Single	Single	NA	Single	NA	Single
Write status register command (containing QE bit)	0x01	Single	Single	NA	Single	NA	Single

Notice List

Fix	Description	Location

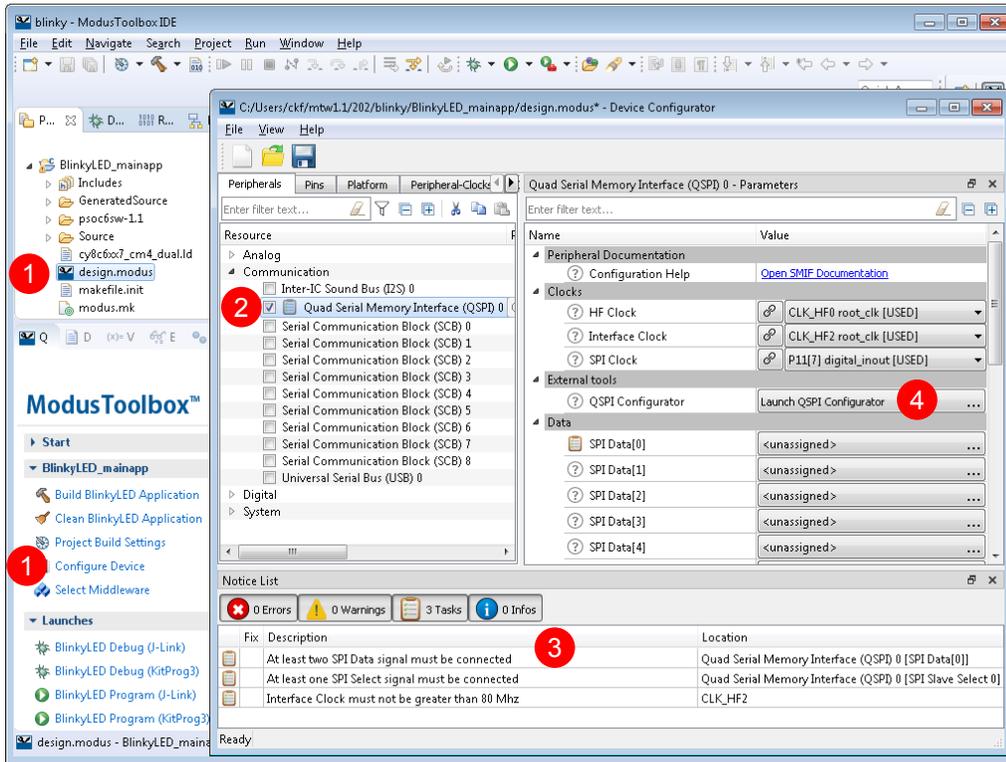
Launch the Configurator

You can run the QSPI Configurator from, and use it with, a ModusToolbox IDE application. You can also run it independently of the IDE. Then, you can either use the generated source with a ModusToolbox IDE application, or use it in any software environment you choose.

From a ModusToolbox IDE Application

1. Launch the Device Configurator from the ModusToolbox IDE.
2. On the Peripherals tab, enable the **QSPI** resource.
3. Optionally, review and resolve the tasks in the Notice List pane. These can be resolved later.
4. On the Parameters tab, click the **Launch QSPI Configurator** button.

Refer to the *Device Configurator Guide* for more information.



The QSPI Configurator is based on the *design.modus* file, which contains all the required information about the device and the application. This information is passed to the QSPI Configurator. When you save changes, it generates/updates firmware in the ModusToolbox IDE application's "GeneratedSource" folder.

Independent of the ModusToolbox IDE

To run the QSPI Configurator independently, navigate to the install location and run the executable. The default install location for the QSPI Configurator on Windows is:

```
<install_dir>\tools\qspi-configurator-<version>
```

For other operating systems, the installation directory will vary, based on how the software was installed. When run independently, the QSPI Configurator opens without any memories configured. You can either open a specific Configuration file or create a new one. See [Menus](#) for more information.

- If you open a Configuration file from a ModusToolbox IDE application, it will be the same flow as if you opened it [from the ModusToolbox IDE](#).
- If you open a Configuration file from a non-ModusToolbox IDE application, the flow will be your preferred working environment.
- If you create a new Configuration file, specify the file name and location to store the file.

From the Command Line

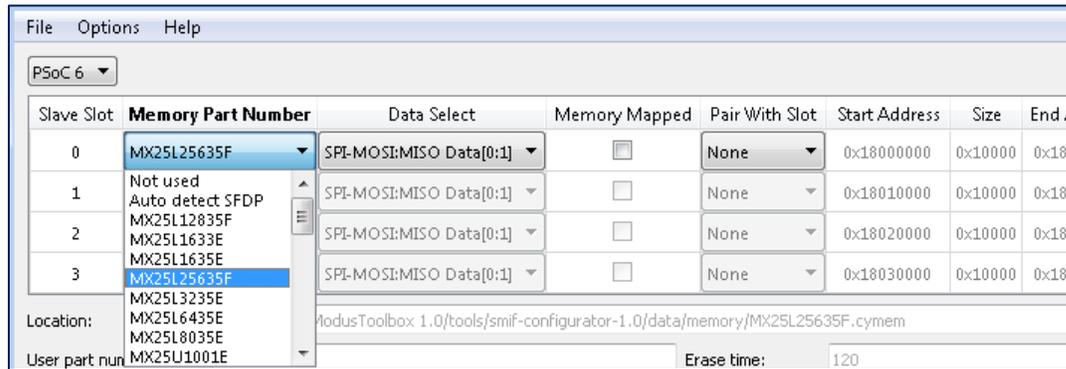
You can run the configurator from the command line. However, there are only a few reasons to do this in practice. The primary use case would be to re-generate source code based on the latest configuration settings. This would often be part of an overall build script for the entire application.

For information about command line options, run the configurator using the `-h` option.

Quick Start

This section provides a simple workflow for how to use the QSPI Configurator.

1. [Launch the configurator.](#)
2. On the QSPI Configurator, select a memory from the **Memory Part Number** list, and specify the configuration parameters if required (for example, memory mapped, write enable). See [Memory Configuration Fields](#).



Note If the required memory part number is not present in the list, you can add it. Select **<Browse>...**, navigate to the memory file location, and select it.

3. Save the Configuration file to generate source code.

The QSPI Configurator generates code into a “GeneratedSource” directory in your ModusToolbox IDE application, or in the same location you saved the Configuration file for non-ModusToolbox IDE applications. That directory contains the necessary source (.c) and header (.h) files for the generated firmware, which uses the relevant driver APIs to configure the hardware.

4. Use the generated structures as input parameters for QSPI functions in your application.

QSPI Configurator Menus

The QSPI Configurator contains the following menus.

File

- **Open Configuration:** Open and load an existing Configuration file.
- **Save Configuration:** Save changes to the file. If the file does not exist, the Save file dialog will open.
- **Save Configuration As...:** Save changes to a new file.
- **New *.cymem File:** Create a new memory file with default parameters. See [Create Memory File](#).
- **Open *.cymem File:** Open an existing memory file.
- **Exit:** Close the configurator.

View

- **Notice List:** Shows/hides the Notice List pane, which contains any errors, warnings, tasks, and information notes. See the *Device Configurator Guide* for more details.
- **Toolbar:** Shows/hides the Toolbar.

Options

- **Settings:** Opens the Settings dialog to set the default path for the memory file to be saved.

Help

- **View Help:** Opens this document.
- **About QSPI Configurator:** Open the About box for version information.

QSPI Configuration Fields

The top part of the QSPI Configurator contains the following parameters:

Parameter	Description
Slave Select Slot	Specifies the slave slot being configured. This number corresponds to the slave select line that will be connected to the memory device.
Memory Part Number	Device part number represents the memory device that is being interfaced to the corresponding slave slot. You can select a memory device from the list, or select the option to auto detect the device. Based on the memory device selected, the corresponding *.cymem file is linked into the slave slot
Data Select	Lets you select the data line options for the corresponding memory slot.
Memory Mapped	When this option is enabled, the configured memory device is mapped to the PSoC device's memory map. If disabled, access to memory must be done through the QSPI API.
Pair with Slot	Determines the paired slot for dual Quad operation
Start Address	Determines the starting address where the memory device is going to be mapped in the PSoC memory map.
Size	Determines size of the memory device to be mapped in the PSoC memory map. This field is a representation of the size of the memory being mapped.
End Address	Represents the end address of the memory device as mapped in the PSoC device's memory map. This field is a representation of the size of the memory being mapped.
Write Enable	Lets you enable or disable writes to the external memory in a memory mapped mode of operation.
Config Data in Flash	Determines whether a specific memory slot's config structures are to be placed in Flash or SRAM. When chosen to be placed in SRAM, the support for the programmer is not provided.
Encrypt	Determines whether to treat the memory device in the corresponding slave slot as an encrypted device. If the memory is mapped, all access to this memory will be decrypted on-the-fly. Setting this field does expect that the right encryption key is loaded as a part of the secure image.

Edit Memory File Fields

The Edit Memory dialog contains the following fields. These fields also display as read-only in the lower part of the QSPI Configurator.

Field	Description
Location	Path and file name of the current memory file.
User part number	User-defined field for the part. This field will be displayed in the main QSPI Configurator window in the Memory part number , when it is selected.
Status register busy mask	Mask for the busy bit in the status register.

Field	Description
Status register quad enable mask	Mask for the quad enable bit in the status register.
Size of memory	Denotes the actual size of the memory device.
Program page size	Denotes the page size for a program operation. This size provides the granularity with which program operations can be committed in the memory device.
Erase block size	Provides the erase block size.
Number of address bytes for QSPI transactions	Sets the number of bytes that are expected for the address field in the QSPI transaction.
Erase time	Time the device typically takes to erase a Erase Type 1 size. You must poll the device's busy status to determine whether the operation has completed. This field has no meaning if the corresponding Erase Type size is 00h.
Chip erase time	Typical time to erase one chip (die). You must poll the device's busy status to determine whether the operation has completed. For a device consisting of multiple dies that are individually accessed, the time is for each die to which a chip erase command is applied.
Program time	Typical time the device takes to write a full page. You must poll the device's busy status to determine whether the operation has completed. You may scale this by half or a quarter to determine approximate times for half and quarter page program operations.
Description	Blank field to type a description for the memory.
SPI modes supported	This shows the SPI data widths supported by the selected memory.
Lower Table	
Description	List of commands: <ul style="list-style-type: none"> • Read command format • Write enable command format • Write disable command format • Erase command format • Chip erase command format • Program command format • Read status register command (containing QE bit) • Read status register command (containing WIP bit) • Write status register command (containing QE bit)
Number	Byte command word.
Command width	Width of the command transfer
Address width	Width of the address transfer
Mode	Provides the mode word for the command
Mode width	Provides the width of the mode word transfer
Dummy cycles	Provides the number of dummy cycles in the transfer
Data width	Provides the width of data bytes in the transfer

Memory Database

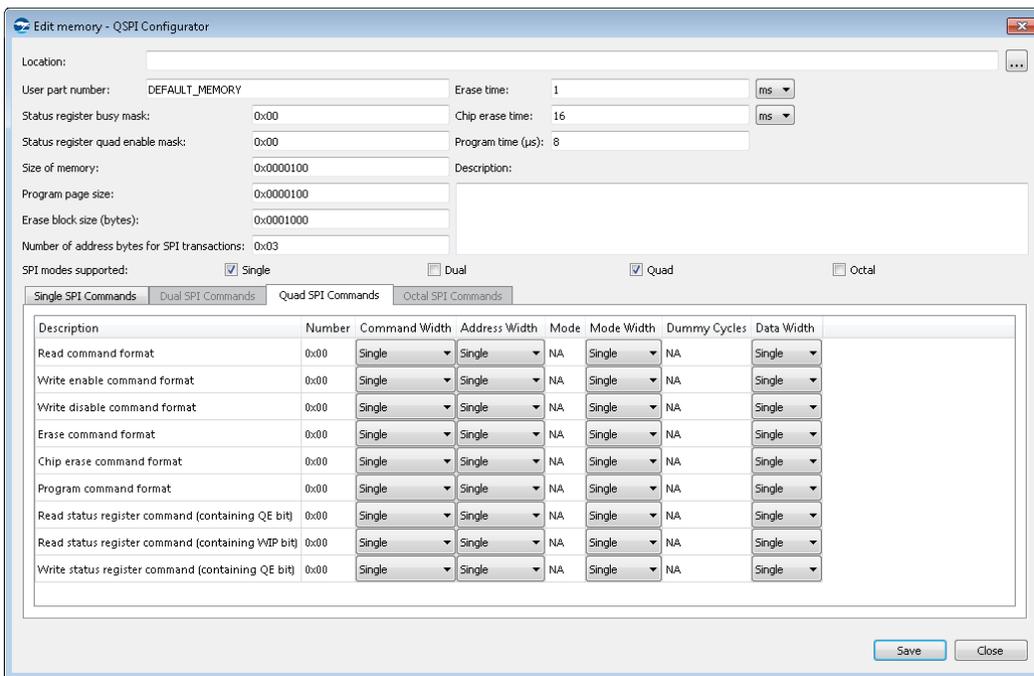
The QSPI Configurator memory database is a set of default memory configurations, based on values from each memory's datasheet. Check that the selected memory configuration is aligned with a particular part number.

Notes:

- By default, the memory database is configured for QuadSPI mode only.
- By default, some memory parts are configured with protected regions, which prevents the successful execution of program/erase memory commands.
- Dummy cycles may vary based on memory part configuration.
- The list of supported commands may vary between memory parts.

Create New Memory File

1. Select **New *.cymem File** to open the Edit memory window from the main QSPI Configurator:



Description	Number	Command Width	Address Width	Mode	Mode Width	Dummy Cycles	Data Width
Read command format	0x00	Single	Single	NA	Single	NA	Single
Write enable command format	0x00	Single	Single	NA	Single	NA	Single
Write disable command format	0x00	Single	Single	NA	Single	NA	Single
Erase command format	0x00	Single	Single	NA	Single	NA	Single
Chip erase command format	0x00	Single	Single	NA	Single	NA	Single
Program command format	0x00	Single	Single	NA	Single	NA	Single
Read status register command (containing QE bit)	0x00	Single	Single	NA	Single	NA	Single
Read status register command (containing WIP bit)	0x00	Single	Single	NA	Single	NA	Single
Write status register command (containing QE bit)	0x00	Single	Single	NA	Single	NA	Single

2. Ignore the **Location** field for a new memory file. This field can be used to select an existing memory file instead of creating a new one.
3. Enter a desired **User part number**. This field will be displayed in the main QSPI Configurator window in the **Memory part number**, when it is selected.
4. Complete the information for the remaining fields, as appropriate. See [Edit Memory File Fields](#).
5. Click **Save** to open a save dialog. Then on the dialog, navigate to the appropriate location, type a file name, and click **Save**.
6. Click **Close** to return to the QSPI Configurator.

References

Refer to the following documents for more information, as needed:

- Device Configurator Guide
- ModusToolbox IDE User Guide
- PDL Reference Guide
- Device Datasheets
- Device Technical Reference Manuals

Version Changes

This section lists and describes the changes for each version of this tool.

Version	Change Descriptions
1.0	New tool.
1.1	Added Notice List and toolbar. Updated to accommodate back end changes.

© Cypress Semiconductor Corporation, 2018-2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, ModusToolbox, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.