ModusToolbox™

QSPI Configurator guide

About this document

Version

3.0

Scope and purpose

The Quad Serial Peripheral Interface (QSPI) Configurator is part of a collection of tools included with ModusToolbox™ software. The QSPI Configurator is a stand-alone tool used to open or create configuration files, configure memory slots, and generate code for your application.

Intended audience

This document helps application developers understand how to use the QSPI Configurator as part of creating a ModusToolbox™ application.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus
Italics	Denotes file names and paths.
`Courier New`	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
File > New	Indicates that a cascading sub-menu opens when you select a menu item

Abbreviations and definitions

The following define the abbreviations and terms used in this document:

- MPN – memory part number
- QSPI – quad serial peripheral interface
- SFDP – serial flash discoverable parameter
- SMIF – serial memory interface

Reference documents

Refer to the following documents for more information as needed:

- Device Configurator guide
- Eclipse IDE for ModusToolbox™ user guide
- ModusToolbox™ tools package release notes
- PDL reference guide
- Device datasheets
- Device technical reference manuals

Table of contents

1 Overview 3
1.1 Supported software 3
2 Launch the QSPI Configurator 4
2.1 make command 4
2.2 Eclipse IDE 4
2.3 From the Device Configurator 4
2.4 Executable (CLI) 5
2.5 Executable (GUI) 5
3 Quick start 6
4 Code generation 7
4.1 Files used by the configuration tool 7
5 GUI description 8
5.1 Menus 8
5.2 Toolbar 9
5.3 Target device family 9
5.4 Flashloader Directory 9
5.5 QSPI configuration fields 10
5.6 Edit memory file fields 10
6 Create new memory file 13
6.1 Memory database 13
7 XML schemas 14
7.1 *.cyqspi schema 14
7.2 *.cymem schema 14
8 Version changes 16

1 Overview

PSoC 6™ MCUs have both volatile RAM and non-volatile flash built in; however, you might want access to more memory than is available. You can use the SPI to connect one to four additional memory chips (Flash, non-volatile FRAM, or RAM) to the PSoC 6™ MCU. Every such memory uses at least two SPI Data wires, and some memories can use up to eight SPI Data wires to transmit data.

You can use the Serial Flash Discoverable Parameter (SFDP) standard to query the memory or memories attached to the device at run-time, or you can specify the memory part numbers (MPNs) for the Serial Memory InterFace (SMIF) driver functions to access the memory. If you choose not to Memory Map the memory, then use the SMIF driver functions (or abstraction layer functions written on top of it) to access it.

You can also choose to disable writing to the flash memory in your final product, as well as choose to encrypt the contents of the memory to prevent unwanted access.

1.1 Supported software

Name	Version	Link
CAT1 Peripheral Driver Library	2.3.0	**GitHub repo**: https://github.com/Infineon/mtb-pdl-cat1 **SMIF Driver Documentation**: https://Infineon.github.io/mtb-pdl-cat1/pdl_api_reference_manual/html/group__group__smif.html

2 Launch the QSPI Configurator

There are numerous ways to launch the QSPI Configurator, and those ways depend on how you use the various tools in ModusToolbox™ software.

2.1 make command

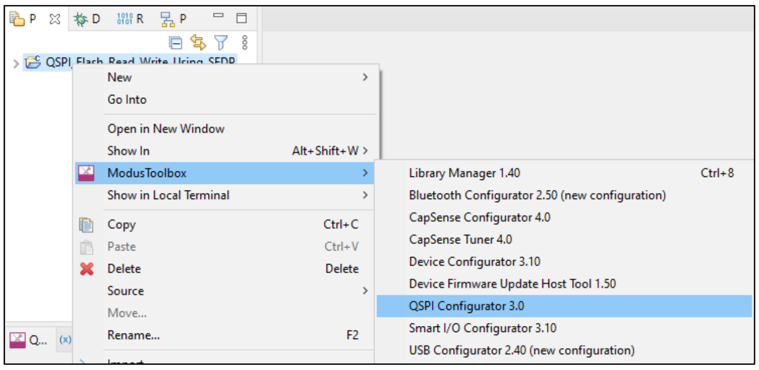
As described in the ModusToolbox™ user guide build system chapter, you can run numerous make commands in the application directory, such as launching the QSPI Configurator. After you have created a ModusToolbox™ application, navigate to the application directory and type the following command in the appropriate bash terminal window:

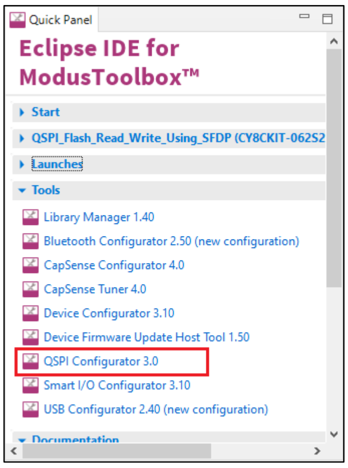
```
    make open CY_OPEN_TYPE=qspi-configurator
```

This command opens the QSPI Configurator GUI for the specific application in which you are working.

2.2 Eclipse IDE

If you use the Eclipse IDE for ModusToolbox™, you can launch the QSPI Configurator for the selected application. In the Project Explorer, right-click on the project and select **ModusToolbox > QSPI Configurator <version>**. You can also click the QSPI Configurator link in the IDE Quick Panel.


QSPI_Flash_Read_Write_Using_SFDP
New
Go Into
Open in New Window
Show In Alt+Shift+W
ModusToolbox
Show in Local Terminal
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Source
Move...
Rename... F2
Library Manager 1.40 Ctrl+8
Bluetooth Configurator 2.50 (new configuration)
CapSense Configurator 4.0
CapSense Tuner 4.0
Device Configurator 3.10
Device Firmware Update Host Tool 1.50
QSPI Configurator 3.0
Smart I/O Configurator 3.10
USB Configurator 2.40 (new configuration)

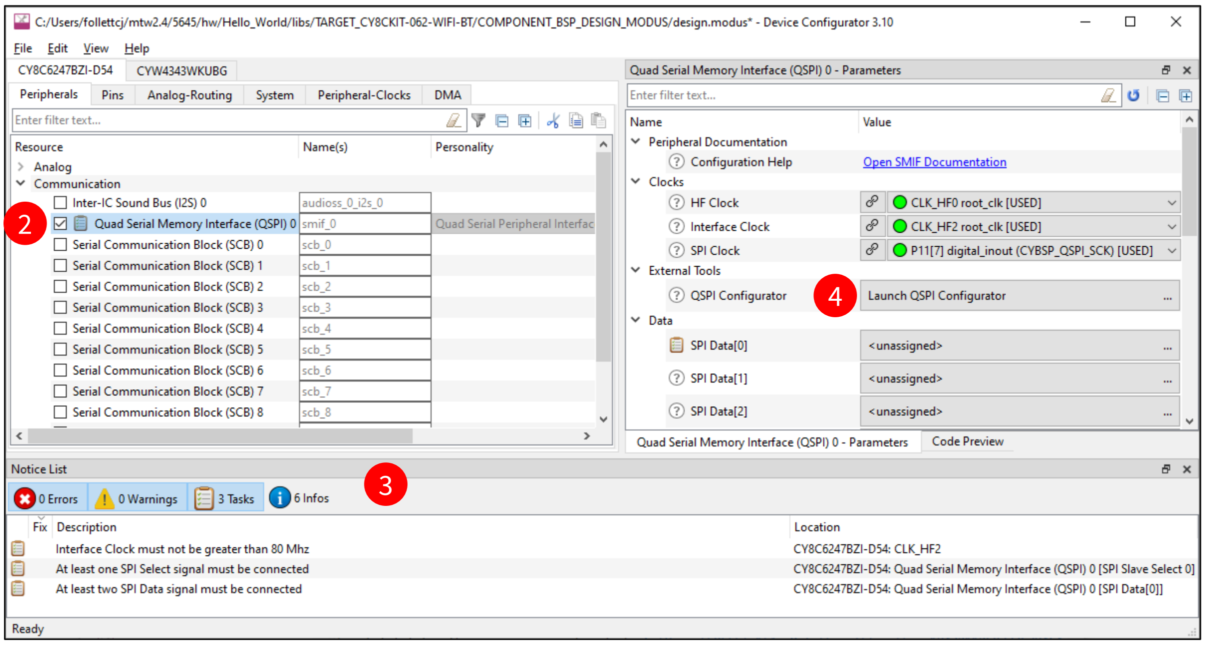

Quick Panel
Eclipse IDE for ModusToolbox™
Start
QSPI_Flash_Read_Write_Using_SFDP (CY8CKIT-062S2
Launches
Tools
Library Manager 1.40
Bluetooth Configurator 2.50 (new configuration)
CapSense Configurator 4.0
CapSense Tuner 4.0
Device Configurator 3.10
Device Firmware Update Host Tool 1.50
QSPI Configurator 3.0
Smart I/O Configurator 3.10
USB Configurator 2.40 (new configuration)
Documentation

Similar to the make command method, launching the QSPI Configurator using the Eclipse IDE opens the tool for the selected application. Refer to the Eclipse IDE for ModusToolbox™ User Guide for details about the IDE.

2.3 From the Device Configurator

You can also launch the QSPI Configurator from the Device Configurator:

1. Open the Device Configurator. See the Device Configurator guide for details.
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.



C:/Users/follettcj/mtw2.4/5645/hw/Hello_World/libs/TARGET_CY8CKIT-062-WIFI-BT/COMPONENT_BSP_DESIGN_MODUS/design.modus* - Device Configurator 3.10

Resource	Name(s)	Personality
Analog		
Communication		
Inter-IC Sound Bus (I2S) 0	audioss_0_i2s_0	
✓ Quad Serial Memory Interface (QSPI) 0	smif_0	Quad Serial Peripheral Interfac
Serial Communication Block (SCB) 0	scb_0	
Serial Communication Block (SCB) 1	scb_1	
Serial Communication Block (SCB) 2	scb_2	
Serial Communication Block (SCB) 3	scb_3	
Serial Communication Block (SCB) 4	scb_4	
Serial Communication Block (SCB) 5	scb_5	
Serial Communication Block (SCB) 6	scb_6	
Serial Communication Block (SCB) 7	scb_7	
Serial Communication Block (SCB) 8	scb_8	

Name	Value
Peripheral Documentation	
Configuration Help	Open SMIF Documentation
Clocks	
HF Clock	CLK_HF0 root_clk [USED]
Interface Clock	CLK_HF2 root_clk [USED]
SPI Clock	P11[7] digital_inout (CYBSP_QSPI_SCK) [USED]
External Tools	
QSPI Configurator	Launch QSPI Configurator
Data	
SPI Data[0]	<unassigned>
SPI Data[1]	<unassigned>
SPI Data[2]	<unassigned>

Fix	Description	Location
	Interface Clock must not be greater than 80 Mhz	CY8C6247BZI-D54: CLK_HF2
	At least one SPI Select signal must be connected	CY8C6247BZI-D54: Quad Serial Memory Interface (QSPI) 0 [SPI Slave Select 0]
	At least two SPI Data signal must be connected	CY8C6247BZI-D54: Quad Serial Memory Interface (QSPI) 0 [SPI Data[0]]

The QSPI Configurator saves configuration information in a *.*cyqspi* file, which contains all the required information about the device and the application. When you save changes, the QSPI Configurator generates/updates firmware in the BSP’s "GeneratedSource" folder.

2.4 Executable (CLI)

The QSPI Configurator executable can be run from the command line, and it also has a "cli" version of the executable as well. Running configurator executables from the command line can be useful as part of batch files or shell scripts to re-generate the source code based on the latest configuration settings. The exit code for the executable is zero if the operation is successful, or non-zero if the operation encounters an error. For more information about the command-line options, run the executable using the `-h` option.

2.5 Executable (GUI)

If you don't have an application or if you just want to see what the configurator looks like, you can launch the QSPI Configurator GUI by running its executable as appropriate for your operating system (for example, double-click it or select it using the Windows **Start** menu). By default, it is installed here:

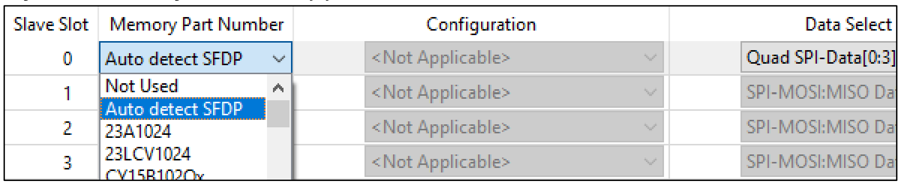
<Install_dir>/ModusToolbox/tools_<version>/qspi-configurator-<version>

When launched this way, 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.

3 Quick start

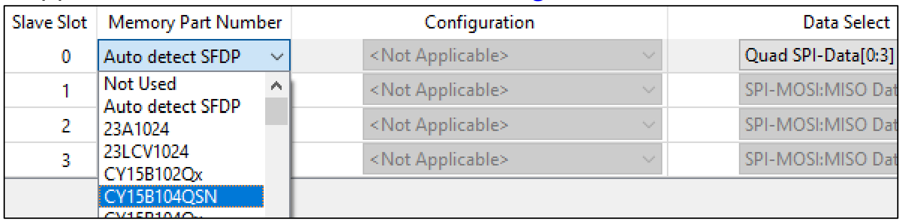
This section provides a simple workflow for how to use the QSPI Configurator.

1. Launch the configurator.
2. Select a memory from the **Memory Part Number** list.
   a. If your memory device supports SFDP, select **Auto detect SFDP**.



Slave Slot	Memory Part Number	Configuration	Data Select
0	Auto detect SFDP	<Not Applicable>	Quad SPI-Data[0:3]
1	Not Used	<Not Applicable>	SPI-MOSI:MISO Da
2	Auto detect SFDP	<Not Applicable>	SPI-MOSI:MISO Da
3	23A1024	<Not Applicable>	SPI-MOSI:MISO Da
	23LCV1024		
	CY15B102Qx		

   b. If your memory device does not support SFDP, or for manual configuration, select a memory from the **Memory Part Number** list, and specify the configuration parameters as required (for example, memory mapped, write enable, etc.). See QSPI Configuration Fields.



Slave Slot	Memory Part Number	Configuration	Data Select
0	Auto detect SFDP	<Not Applicable>	Quad SPI-Data[0:3]
1	Not Used	<Not Applicable>	SPI-MOSI:MISO Dat
2	Auto detect SFDP	<Not Applicable>	SPI-MOSI:MISO Dat
3	23A1024	<Not Applicable>	SPI-MOSI:MISO Dat
	23LCV1024		
	CY15B102Qx		
	CY15B104QSN		
	CY15B104Q…		

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

4 Code generation

The QSPI Configurator generates code into a "GeneratedSource" directory in your BSP, or in the same location you saved the Configuration file for non-Eclipse 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.

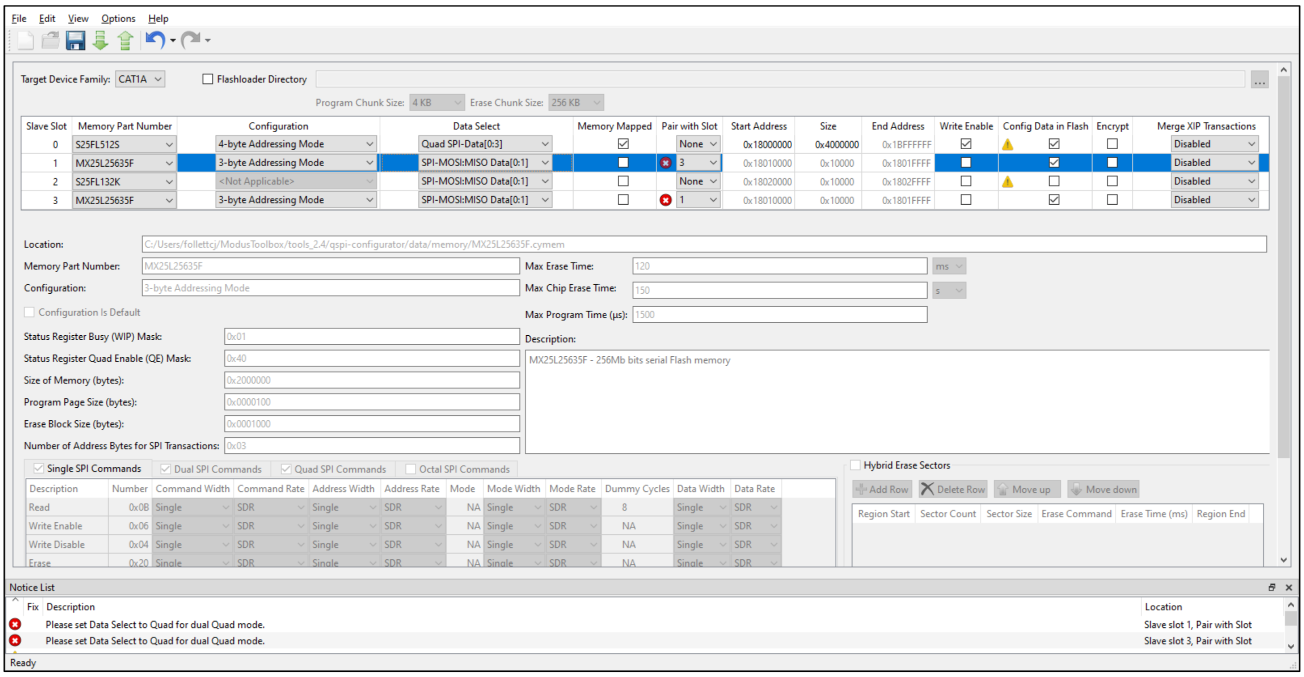
4.1 Files used by the configuration tool

The QSPI Configurator uses and provides write access to two types of files: the QSPI configuration file (**.cyqspi*), and the memory configuration files: (*.cymem*). The QSPI Configurator uses the information from these files to generate *cycfg_qspi_memslot.c*, *cycfg_qspi_memslot.h*, and *qspi_config.cfg* files.

- **.cyqspi* – This file controls the application-level settings used to provide access to the serial memory device. In the GUI, these settings are at the top of the main window. For the format and contents of the **.cyqspi* file, see *.cyqspi schema.
- **.cymem* – These files contain parameters of a specific serial memory device, including command formats, memory access sizes, and memory access timings. Unless you are creating a new **.cymem* file, these settings are typically read-only and are shown in the lower part of the main window. For the format and contents of **.cymem* files, see *.cymem schema.
- *cy_qspi_memslot.c* – This file is generated by the QSPI Configurator and contains structures populated with parameters specified by the **.cyqspi* file and the **.cymem* file or files.
- *cy_qspi_memslot.h* – This file is generated by the QSPI Configurator and contains the declarations of the structures defined in *cy_qspi_memslot.c*.
- *qspi_config.cfg* – This file contains a SMIF Bank layout for use with OpenOCD. It can be ignored if OpenOCD is not in use.

5 GUI description

The top area of the QSPI Configurator is used for configuring memories; the read-only area below it displays information about the selected memory part number.



Target Device Family: CAT1A
☐ Flashloader Directory
Program Chunk Size: 4 KB
Erase Chunk Size: 256 KB

Slave Slot	Memory Part Number	Configuration	Data Select	Memory Mapped	Pair with Slot	Start Address	Size	End Address	Write Enable	Config Data in Flash	Encrypt	Merge XIP Transactions
0	S25FL512S	4-byte Addressing Mode	Quad SPI-Data[0:3]	☑	None	0x18000000	0x4000000	0x1BFFFFFF	☑	⚠ ☑	☐	Disabled
1	MX25L25635F	3-byte Addressing Mode	SPI-MOSI:MISO Data[0:1]	☐	✖ 3	0x18010000	0x10000	0x1801FFFF	☐	☑	☐	Disabled
2	S25FL132K	<Not Applicable>	SPI-MOSI:MISO Data[0:1]	☐	None	0x18020000	0x10000	0x1802FFFF	☐	⚠ ☐	☐	Disabled
3	MX25L25635F	3-byte Addressing Mode	SPI-MOSI:MISO Data[0:1]	☐	✖ 1	0x18010000	0x10000	0x1801FFFF	☐	☑	☐	Disabled

Location: C:/Users/follettcj/ModusToolbox/tools_2.4/qspi-configurator/data/memory/MX25L25635F.cymem
Memory Part Number: MX25L25635F
Configuration: 3-byte Addressing Mode
☐ Configuration Is Default
Max Erase Time: 120 ms
Max Chip Erase Time: 150 s
Max Program Time (µs): 1500
Status Register Busy (WIP) Mask: 0x01
Status Register Quad Enable (QE) Mask: 0x40
Size of Memory (bytes): 0x2000000
Program Page Size (bytes): 0x0000100
Erase Block Size (bytes): 0x0001000
Number of Address Bytes for SPI Transactions: 0x03
Description: MX25L25635F - 256Mb bits serial Flash memory

☑ Single SPI Commands ☑ Dual SPI Commands ☑ Quad SPI Commands ☐ Octal SPI Commands

Description	Number	Command Width	Command Rate	Address Width	Address Rate	Mode	Mode Width	Mode Rate	Dummy Cycles	Data Width	Data Rate
Read	0x0B	Single	SDR	Single	SDR	NA	Single	SDR	8	Single	SDR
Write Enable	0x06	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Write Disable	0x04	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Erase	0x20	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR

☐ Hybrid Erase Sectors

Region Start	Sector Count	Sector Size	Erase Command	Erase Time (ms)	Region End

Notice List

Fix	Description	Location
✖	Please set Data Select to Quad for dual Quad mode.	Slave slot 1, Pair with Slot
✖	Please set Data Select to Quad for dual Quad mode.	Slave slot 3, Pair with Slot

Ready

5.1 Menus

The QSPI Configurator contains the following menus.

5.1.1 File

- **New** – Creates a new Configuration file.
- **Open…** – Opens and loads an existing Configuration file, in either the current *.cyqspi* format or the obsolete formats (*.cysmif* or embedded inside a .h file).
- **Save** – Saves changes to the *.cyqspi* file. If a file has not been opened or is not in the *.cyqspi* format, the Save file dialog will open.
- **Save As…** – Saves changes to a new file.
- **Open in System Explorer** – This opens your computer's file explorer tool to the folder that contains the Configuration file.
- **Import…** – Imports a specified configuration file.
- **Export…** – Exports the current configuration file into a specified file.
- **New *.cymem File…** – Creates a new memory file with default parameters. See Create memory file.
- **Open *.cymem File…** – Opens an existing memory file.
- **Recent Files** – Shows up to five recent files that you can open directly.
- **Exit** – Closes the configurator.

5.1.2 Edit

- **Undo** – Undoes the last action or sequence of actions.
- **Redo** – Redoes the last undone action or sequence of undone actions.

5.1.3 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.
- **Reset View** – Restores the Notice List and Toolbar to the default state.

5.1.4 Options

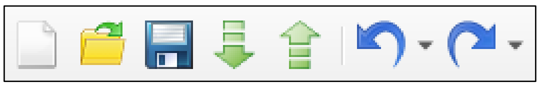
- **Settings…** – Opens the Settings dialog to set the default path for the memory file to be saved.

5.1.5 Help

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

5.2 Toolbar

The toolbar contains a few of the same commands included on the File and Edit menus.



5.3 Target device family

This pull-down menu allows you to select the device family for which the memory settings apply:

- CAT1A = PSoC™ 6 MCU
- CAT1B = Future use
- CAT1C = Future use

5.4 Flashloader Directory

A flashloader is an ELF file containing the algorithms and data structures required for flash memory programming. The QSPI Configurator can patch flashloaders so that their data structures contain data about the external flash memory that will be connected to your design, as well as how the memory is configured. The patched files are placed in the *GeneratedSource/* directory alongside the source code and header files generated by the QSPI Configurator.

To enable flashloader patching, select the **Flashloader Directory** check box and use the [**…**] button to find and select the appropriate directory (or type the directory location in the text field). When you enable the **Flashloader Directory** check box, the following fields also become active to set appropriate chunk sizes:

- Program Chuck Size
- Erase Chunk Size

These settings allow you to control the granularity of the operations done while programming the flash memory. Larger values allow for more efficient data transfer, while smaller values can use memory more efficiently. When a memory slot is configured for Auto detect SFDP, the QSPI Configurator cannot know what values are legal, so ensure that these values are at least as large as the largest size of the program/erase operations of all the memories in use. Also, neither value can be larger than the smallest memory in use.

Note: *You can also run `qspi-configurator` or `qspi-configurator-cli` via the command line using the `--flashloader-dir` option. If so, the next argument must be the directory path containing the flashloader(s) to patch. If you provide this argument on the command line, the flashloader directory cannot be changed in the GUI.*

5.5 QSPI configuration fields

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

Parameter	**Description**
Slave 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.
Configuration	For certain Memory Part Numbers, there is additional configuration information, such as 3-byte or 4-byte Addressing Mode.
Data Select	Allows you to 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™ MCU'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 in bytes of the memory device to be mapped in the PSoC™ memory map.
End Address	Represents the end address of the memory device as mapped in the PSoC™ MCU's memory map. This is a read-only field that is calculated automatically from "Start Address" and "Size" cells values.
Write Enable	This 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. Refer to AN228740 – PSoC™ 6 MCU Guide to Using Serial Memory Interface (QSPI).
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.
Merge XIP Transactions	Specifies how many cycles can pass between memory accesses while still skipping the overhead of re-sending the read command. For CAT1A (PSoC™ 6) devices, this field must be Disabled.

5.6 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.

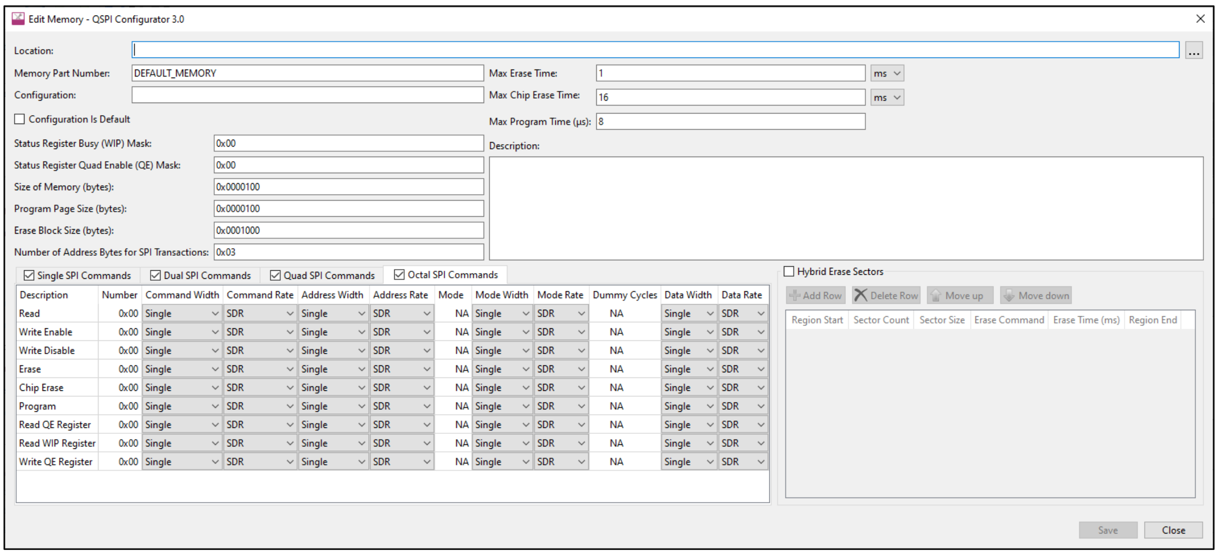
Field	**Description**
Location	The path and file name of the current memory file.
Memory Part Number	The is the name of the memory chip for which this configuration file is designed. This field will be displayed in the main QSPI Configurator window in the **Memory Part Number** drop-down menu.

Field	Description
Configuration	Some memory parts can be configured in more than one way, and each configuration is contained in a separate memory file. When creating more than one memory file for a part, use this field to describe how this file differs from the others. For example, "3-byte Addressing Mode" or "Hybrid Sectors at Bottom".
Configuration Is Default	For memory parts with more than one configuration, this check box specifies whether this configuration is the one that should be automatically selected when this memory part is chosen in the Memory Part Number drop-down menu.
Status Register Busy (WIP) Mask	Mask for the busy bit in the status register.
Status Register Quad Enable (QE) Mask	Mask for the quad enable bit in the status register.
Size of Memory (bytes)	Denotes the actual size of the memory device.
Program Page Size (bytes)	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 (bytes)	Provides the erase block size.
Number of Address Bytes for SPI Transactions	Sets the number of bytes that are expected for the address field in the QSPI transaction.
Max 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.
Max 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.
Max Program Time (µs)	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.
Commands Table (Single SPI, Dual SPI, Quad SPI, Octal SPI) Tabs show the SPI data widths supported by the selected memory.	
Description	List of commands: • Read • Write Enable • Write Disable • Erase Command • Chip Erase • Program • Read QE Register • Read WIP Register • Write QE Register
Number	Byte command word.
Command Width	Width of the command transfer.
Command Rate	Determines whether the command byte should be sent in Single Data Rate (SDR) or Double Data Rate (DDR) mode. CAT1A (PSoC™ 6) devices only support SDR.
Address Width	Width of the address transfer.

Field	**Description**
Address Rate	Determines whether the address bytes should be sent in SDR or DDR mode. CAT1A (PSoC™ 6) devices only support SDR.
Mode	Provides the mode word for the command.
Mode Width	Provides the width of the mode word transfer.
Mode Rate	Determines whether the mode byte should be sent in SDR or DDR mode. CAT1A (PSoC™ 6) devices only support SDR.
Dummy Cycles	Provides the number of dummy cycles in the transfer.
Data Width	Provides the width of data bytes in the transfer.
Data Rate	Determines whether the data bytes should be sent in SDR or DDR mode. CAT1A (PSoC™ 6) devices only support SDR.
Hybrid Erase Sectors Section	
Hybrid Erase Sectors	This check box specifies whether sectors of different sizes are present in this configuration. The table below contains details about those sectors. If this check box is not enabled, then there are no sectors with different sizes, and none of the other fields are active.
Row Buttons	Use these buttons to add and delete rows, as well as move rows up and down.
Region Table	This table contains information about the various erase sector regions: • The Region Start column specifies the start of the erase sector region. • The Sector Count column specifies how many erase sectors this region contains. • The Sector Size column specifies the size of the erase sectors in this region. • The Erase Command column specifies what command should be used to erase sectors in this region. • The Erase Time column specifies the maximum time that it can take to erase a sector in this region.

6 Create new memory file

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



Edit Memory - QSPI Configurator 3.0

Location:	
Memory Part Number:	DEFAULT_MEMORY
Configuration:	
Configuration Is Default	☐
Status Register Busy (WIP) Mask:	0x00
Status Register Quad Enable (QE) Mask:	0x00
Size of Memory (bytes):	0x0000100
Program Page Size (bytes):	0x0000100
Erase Block Size (bytes):	0x0001000
Number of Address Bytes for SPI Transactions:	0x03
Max Erase Time:	1 ms
Max Chip Erase Time:	16 ms
Max Program Time (µs):	8
Description:	

☑ Single SPI Commands ☑ Dual SPI Commands ☑ Quad SPI Commands ☑ Octal SPI Commands

Description	Number	Command Width	Command Rate	Address Width	Address Rate	Mode	Mode Width	Mode Rate	Dummy Cycles	Data Width	Data Rate
Read	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Write Enable	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Write Disable	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Erase	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Chip Erase	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Program	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Read QE Register	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Read WIP Register	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR
Write QE Register	0x00	Single	SDR	Single	SDR	NA	Single	SDR	NA	Single	SDR

☐ Hybrid Erase Sectors

Region Start	Sector Count	Sector Size	Erase Command	Erase Time (ms)	Region End

2. Click [**. . .**] button next to the **Location** field to specify the file name and location of the memory configuration file (*.cymem). If you prefer, you can ignore the **Location** field for now, and then specify the file name and location when saving the file.
3. Enter a desired **Memory Part Number**. When selected, this field will be displayed in the main QSPI Configurator window in the **Memory Part Number**.
4. Complete the information for the remaining fields, as appropriate. See Edit memory file fields.
5. Click **Save** to save the configured memory. If **Location** was not specified previously, this will open a save dialog to navigate to the appropriate location, type a file name, and click **Save**.
6. Clock **Close** to return to the QSPI Configurator.

6.1 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.

Note: *By default, some memory parts are configured with protected regions, which prevents the successful execution of program/erase memory commands.*

Note: *Dummy cycles may vary based on memory part configuration.*

Note: *The list of supported commands may vary between memory parts.*

7 XML schemas

7.1 *.cyqspi schema

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XQSPISchema">
  <xs:element name="Configuration">
    <xs:complexType>
      <xs:sequence>
        <xs:element type="xs:string" name="DevicePath"/>
        <xs:element type="xs:string" name="FlashloaderDir"/>
        <xs:element type="xs:int" name="FlashloaderProgramChunkSize"/>
        <xs:element type="xs:int" name="FlashloaderEraseChunkSize"/>
        <xs:element name="SlotConfigs">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="SlotConfig" maxOccurs="4" minOccurs="0">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element type="xs:byte" name="SlaveSlot"/>
                    <xs:element type="xs:string" name="PartNumber"/>
                    <xs:element type="xs:boolean" name="MemoryMapped"/>
                    <xs:element type="xs:string" name="DualQuad"/>
                    <xs:element type="xs:string" name="StartAddress"/>
                    <xs:element type="xs:string" name="Size"/>
                    <xs:element type="xs:string" name="EndAddress"/>
                    <xs:element type="xs:boolean" name="WriteEnable"/>
                    <xs:element type="xs:boolean" name="Encrypt"/>
                    <xs:element type="xs:string" name="DataSelect"/>
                    <xs:element type="xs:string" name="MemoryConfigsPath"/>
                    <xs:element type="xs:boolean" name="ConfigDataInFlash"/>
                    <xs:element type="xs:string" name="MergeTimeout"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute type="xs:string" name="app" fixed="QSPI"/>
      <xs:attribute type="xs:int" name="major"/>
      <xs:attribute type="xs:int" name="minor"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

7.2 *.cymem schema

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XQSPISchema">
  <xs:element name="CyMemoryConfiguration">
    <xs:complexType>
      <xs:sequence>
        <xs:element type="xs:string" name="PartNumber"/>
        <xs:element type="xs:string" name="Configuration" minOccurs="0"/>
        <xs:element type="xs:string" name="DisplayName" minOccurs="0"/>
        <xs:element type="xs:string" name="Description"/>
        <xs:element type="xs:string" name="NumberOfAddress"/>
        <xs:element type="xs:string" name="SizeOfMemory"/>
        <xs:element type="xs:string" name="EraseBlockSize"/>
        <xs:element type="xs:string" name="ProgramPageSize"/>
        <xs:element type="xs:string" name="StatusRegisterBusyMask"/>
        <xs:element type="xs:string" name="StatusRegisterQuadEnableMask"/>
        <xs:element type="xs:int" name="EraseTime"/>
        <xs:element type="xs:int" name="ChipEraseTime"/>
        <xs:element type="xs:int" name="ProgramTime"/>
        <xs:element name="HybridInfo" minOccurs="0" maxOccurs="1">
          <xs:complexType>
            <xs:sequence>
              <xs:element maxOccurs="unbounded" name="Region">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="RegionIndex" type="xs:int" />
                    <xs:element name="SectorCount" type="xs:int" />
                    <xs:element name="SectorSize" type="xs:string" />
                    <xs:element name="EraseCommand" type="xs:string" />
                    <xs:element name="EraseTime" type="xs:int" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Commands" maxOccurs="4" minOccurs="1">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Command" maxOccurs="unbounded" minOccurs="0">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element type="xs:string" name="CommandDescription"/>
                    <xs:element type="xs:string" name="CommandName"/>
                    <xs:element type="xs:string" name="CommandNumber"/>
                    <xs:element type="xs:string" name="CmdWidth"/>
                    <xs:element type="xs:string" name="CmdRate"/>
                    <xs:element type="xs:string" name="AddrWidth"/>
                    <xs:element type="xs:string" name="AddrRate"/>
                    <xs:element type="xs:string" name="Mode"/>
                    <xs:element type="xs:string" name="ModeWidth"/>
                    <xs:element type="xs:string" name="ModeRate"/>
                    <xs:element type="xs:byte" name="DummyCycles"/>
                    <xs:element type="xs:string" name="DataWidth"/>
                    <xs:element type="xs:string" name="DataRate"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute type="xs:string" name="mode" use="optional" default="Quad"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute type="xs:string" name="version" use="optional" default="1"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

8 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.
2.0	Added Import/Export functionality. Moved configuration data from being embedded in the .h file to a new .cyqspi file. Set EraseTime, ChipEraseTime and ProgramTime to max. Allowed memory mapping for any Data Select selections. Added warning when "Config Data in Flash" check box isn't selected, and the device part number is not in "auto detect mode." Disabled the "Config Data in Flash" check box when the "Memory Part Number" is set to "Auto detect SFDP." Added handling of invalid command line arguments. Made it impossible for paired slots to have different memory part numbers. Fixed the Pair with Slot Memory Overlap Error.
2.1	Separated the Memory Part Number combo box into Memory Part Number and Configuration. Added support for hybrid erase sectors.
2.20	Added Undo/Redo commands. Added Copy feature to the Notice List.
2.30	Removed the command-line generate options: `-g` and `--generate`.
	Removed version 1.1 personality support.
3.0	Added support for flashloader patching, and added the Flashloader directory field.
	Removed support for loading configuration data from .h files. Configuration information is now always located in the *design.cyqspi* file.

Revision history

Date	Revision	Description
11/27/2018	**	New document.
02/27/2019	*A	Added Notice List and toolbar. Updated to accommodate back end changes.
10/16/2019	*B	Updated to version 2.0.
03/27/2020	*C	Updated to version 2.1.
04/21/2020	*D	Updated text for hybrid memory descriptions.
08/28/2020	*E	Updated to version 2.20.
09/07/2020	*F	Updated to mention the UNITS (bytes/word) of size explicitly. Updated description for "End Address" field. Updated description for "Start address" field. Added a link to App Note AN228740 for the "Config Data in Flash" field.
03/12/2021	*G	Updated to version 2.30.
09/20/2021	*H	Updated to version 3.0.

Edition 2021-09-20

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-24376 Rev. *H