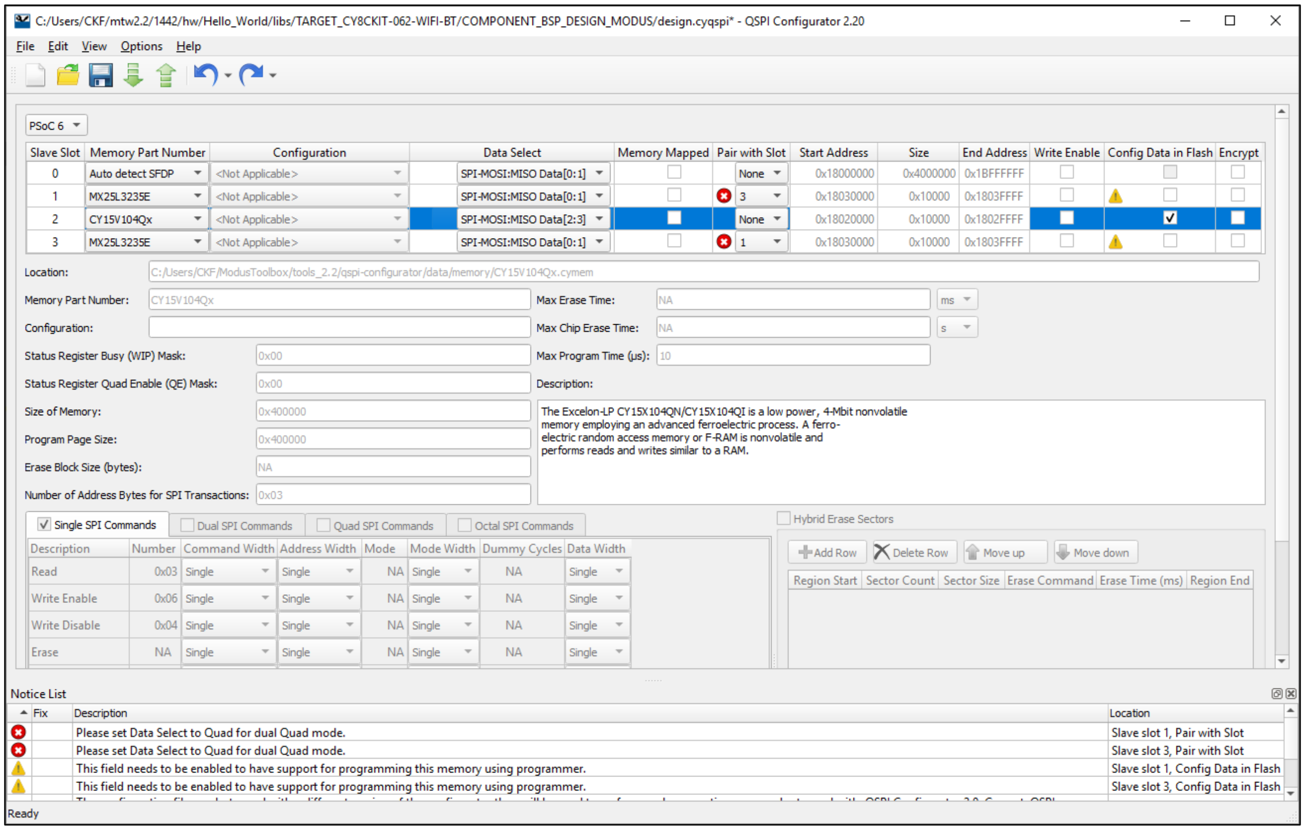
CYPRESS
EMBEDDED IN TOMORROW™

ModusToolbox®
QSPI Configurator Guide

Version 2.20

Overview

The Quad Serial Peripheral Interface (QSPI) Configurator is part of a collection of tools included with 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.


C:/Users/CKF/mtw2.2/1442/hw/Hello_World/libs/TARGET_CY8CKIT-062-WIFI-BT/COMPONENT_BSP_DESIGN_MODUS/design.cyqspi* - QSPI Configurator 2.20
File Edit View Options Help
PSoC 6

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
0	Auto detect SFDP	<Not Applicable>	SPI-MOSI:MISO Data[0:1]	☐	None	0x18000000	0x4000000	0x1BFFFFFF	☐	☐	☐
1	MX25L3235E	<Not Applicable>	SPI-MOSI:MISO Data[0:1]	☐	3	0x18030000	0x10000	0x1803FFFF	☐	☐	☐
2	CY15V104Qx	<Not Applicable>	SPI-MOSI:MISO Data[2:3]	☐	None	0x18020000	0x10000	0x1802FFFF	☐	✓	☐
3	MX25L3235E	<Not Applicable>	SPI-MOSI:MISO Data[0:1]	☐	1	0x18030000	0x10000	0x1803FFFF	☐	☐	☐

Location: C:/Users/CKF/ModusToolbox/tools_2.2/qspi-configurator/data/memory/CY15V104Qx.cymem

Memory Part Number:	CY15V104Qx	Max Erase Time:	NA ms
Configuration:		Max Chip Erase Time:	NA s
Status Register Busy (WIP) Mask:	0x00	Max Program Time (µs):	10
Status Register Quad Enable (QE) Mask:	0x00		
Size of Memory:	0x400000		
Program Page Size:	0x400000		
Erase Block Size (bytes):	NA		
Number of Address Bytes for SPI Transactions:	0x03		

Description:
The Excelon-LP CY15X104QN/CY15X104QI is a low power, 4-Mbit nonvolatile memory employing an advanced ferroelectric process. A ferro-electric random access memory or F-RAM is nonvolatile and performs reads and writes similar to a RAM.

[✓] Single SPI Commands ☐ Dual SPI Commands ☐ Quad SPI Commands ☐ Octal SPI Commands

Description	Number	Command Width	Address Width	Mode	Mode Width	Dummy Cycles	Data Width
Read	0x03	Single	Single	NA	Single	NA	Single
Write Enable	0x06	Single	Single	NA	Single	NA	Single
Write Disable	0x04	Single	Single	NA	Single	NA	Single
Erase	NA	Single	Single	NA	Single	NA	Single

☐ 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
	This field needs to be enabled to have support for programming this memory using programmer.	Slave slot 1, Config Data in Flash
	This field needs to be enabled to have support for programming this memory using programmer.	Slave slot 3, Config Data in Flash

Ready

Launch the QSPI Configurator

You can launch the QSPI Configurator various ways: as a stand-alone tool, from the Eclipse IDE for ModusToolbox, from the Device Configurator or from the command line. Then, you can either use the generated source with an Eclipse IDE application, or use it in any software environment you choose.

As a Stand-Alone Tool

You can launch the QSPI Configurator as a stand-alone tool without the Eclipse IDE. By default, it is installed here:

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

On Windows, you can launch the tool from the **Start** menu.

For other operating systems, the installation directory will vary, based on how the software was installed.

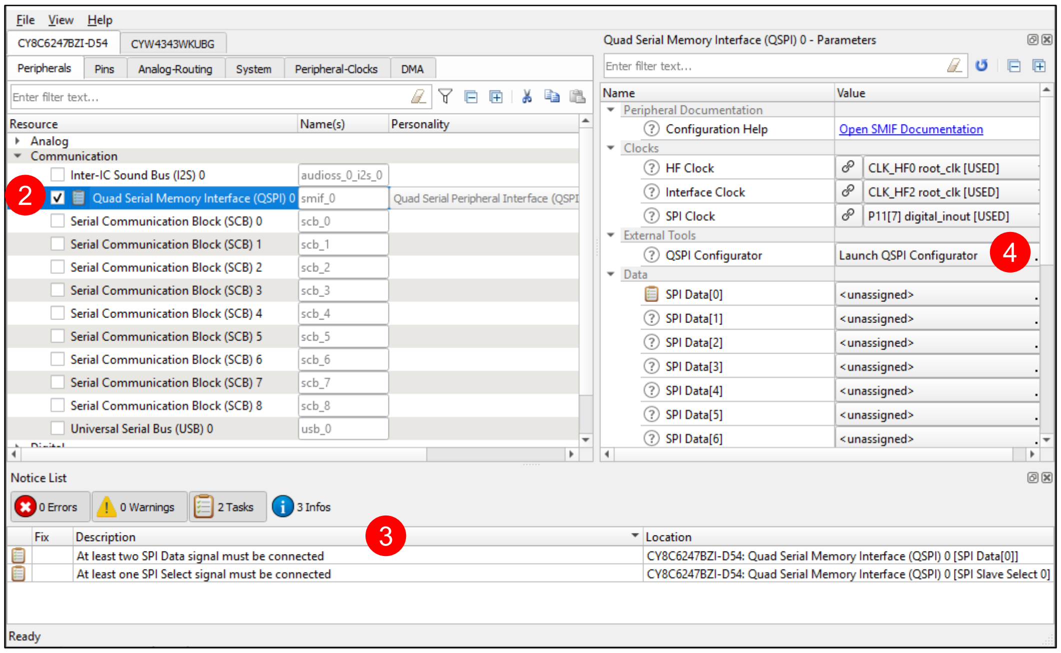
When run independently, the QSPI Configurator opens without any memories configured. You can either open a specific Configuration file or create a new one. See Menus for more information.

- If you open a Configuration file from an Eclipse IDE application, it will be the same flow as if you opened it from the Eclipse IDE.
- If you open a Configuration file from a non-Eclipse IDE application, the flow will be for your preferred working environment.
- If you create a new Configuration file, specify the file name and location to store the file.

From the Device Configurator

If your Eclipse IDE application does **not** include a *design.cyqspi* file:

1. Open the Device Configurator. See the Device Configurator User 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.



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 Interface (QSPI
☐ 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	
☐ Universal Serial Bus (USB) 0	usb_0	

Quad Serial Memory Interface (QSPI) 0 - Parameters

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 [USED]
External Tools	
QSPI Configurator	Launch QSPI Configurator
Data	
SPI Data[0]	<unassigned>
SPI Data[1]	<unassigned>
SPI Data[2]	<unassigned>
SPI Data[3]	<unassigned>
SPI Data[4]	<unassigned>
SPI Data[5]	<unassigned>
SPI Data[6]	<unassigned>

Fix	Description	Location
	At least two SPI Data signal must be connected	CY8C6247BZI-D54: Quad Serial Memory Interface (QSPI) 0 [SPI Data[0]]
	At least one SPI Select signal must be connected	CY8C6247BZI-D54: Quad Serial Memory Interface (QSPI) 0 [SPI Slave Select 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.

From the Eclipse IDE

If your Eclipse IDE application includes a *design.cyqspi* file in the Board Support Package (BSP), right-click on a project in the IDE Project Explorer, and select **ModusToolbox > QSPI Configurator** to open the tool.

Note You can also launch the QSPI Configurator from a link in the Quick Panel.

From the Command Line

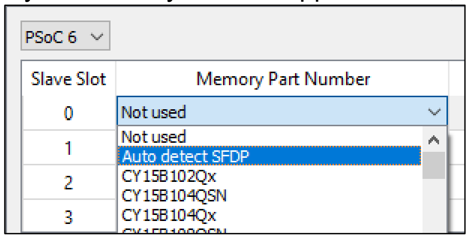
You can run the qspi-configurator executable from the command line. There is also a qspi-configurator-cli executable, which regenerates source code based on the latest configuration settings from a command-line prompt or from within batch files or shell scripts. The exit code for the qspi-configurator-cli executable is zero if the operation is successful, or non-zero if the operation encounters an error. In order to use the qspi-configurator-cli executable, you must provide at least the `--config` argument with a path to the configuration file.

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

Quick Start

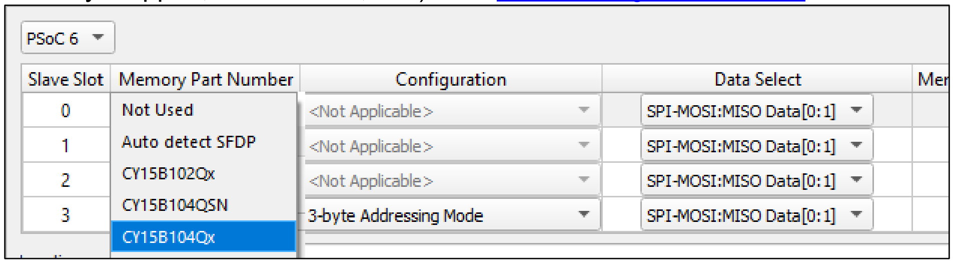
This section provides a simple workflow for how to use the QSPI Configurator.

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
0	Not used
1	
2	
3	

   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		<Not Applicable>	SPI-MOSI:MISO Data[0:1]
1		<Not Applicable>	SPI-MOSI:MISO Data[0:1]
2		<Not Applicable>	SPI-MOSI:MISO Data[0:1]
3		3-byte Addressing Mode	SPI-MOSI:MISO Data[0:1]

   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.

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.

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.

GUI Description

Menus

The QSPI Configurator contains the following menus.

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.
- **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.

Edit

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

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.

Options

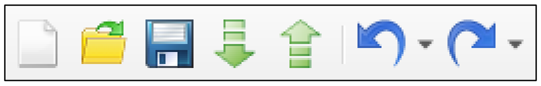
- **Settings…** – Opens the Settings dialog to set the default path for the memory file to be saved.

Help

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

Toolbar

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



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).

Parameter	Description
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	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.
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	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 (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.

Field	Description
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.
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.
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.

Memory Database

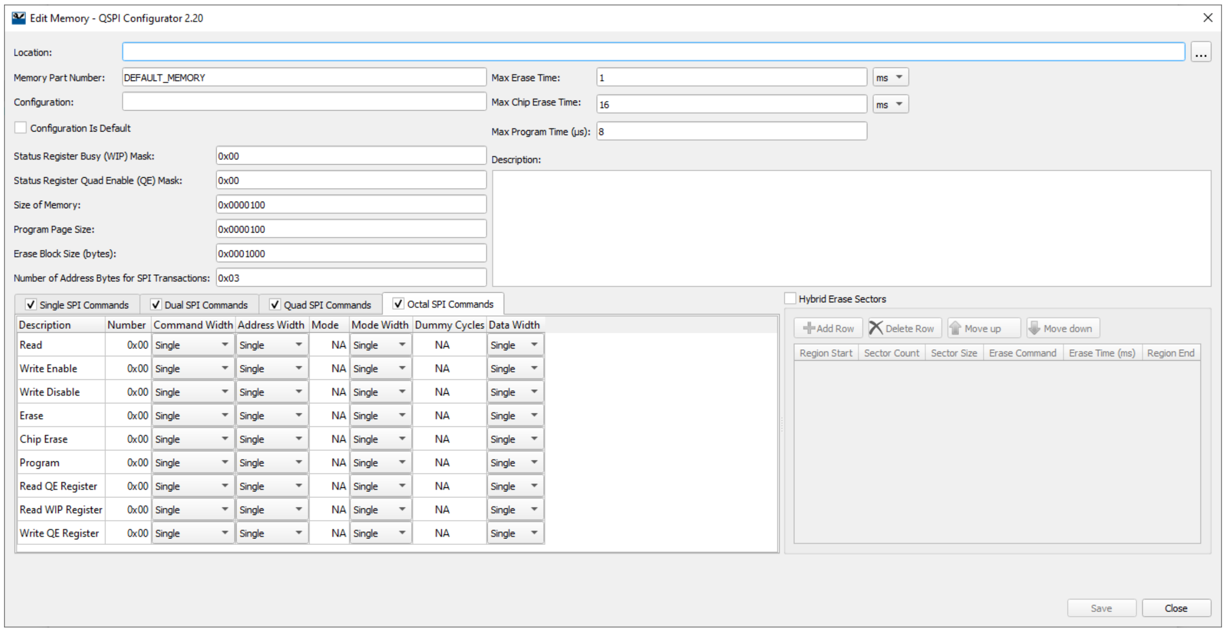
The QSPI Configurator memory database is a set of default memory configurations, based on values from each memory's datasheet. Check that the selected memory configuration is aligned with a particular part number.

Notes:

- By default, 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:



Edit Memory - QSPI Configurator 2.20

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:	0x0000100
Program Page Size:	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	Address Width	Mode	Mode Width	Dummy Cycles	Data Width
Read	0x00	Single	Single	NA	Single	NA	Single
Write Enable	0x00	Single	Single	NA	Single	NA	Single
Write Disable	0x00	Single	Single	NA	Single	NA	Single
Erase	0x00	Single	Single	NA	Single	NA	Single
Chip Erase	0x00	Single	Single	NA	Single	NA	Single
Program	0x00	Single	Single	NA	Single	NA	Single
Read QE Register	0x00	Single	Single	NA	Single	NA	Single
Read WIP Register	0x00	Single	Single	NA	Single	NA	Single
Write QE Register	0x00	Single	Single	NA	Single	NA	Single

☐ 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.

Migration of Configuration File Format

Versions of the QSPI Configurator prior to 2.0 used a C header file to store its configuration as a comment in XML format. In version 2.0 and beyond, the configuration is stored in a separate *.cyqspi* file in the XML format. Use the following instructions to migrate to the *.cyqspi* file as appropriate:

From the Eclipse IDE

1. Launch the QSPI Configurator. If there is no *.cyqspi* file, and if the .h file contains an XML configuration, it will be loaded automatically.
2. Click **Save** to create the *.cyqspi* file and update the C file.

Without the Eclipse IDE

1. Launch the QSPI Configurator.
2. Click **Open**.
3. Next to the **File name**, select the **Obsolete configurator files (*.h)** option and choose a header file.

4. After the configuration loads, click **Save** to create the .cyqspi file and update the C file.

Notes

- The .cyqspi file is located one directory up related to the header file.
- The command-line argument `--config` does not accept obsolete configuration files (*.h).

XML Schemas

*.cyqspi Schema

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="Configuration">
    <xs:complexType>
      <xs:sequence>
        <xs:element type="xs:string" name="DevicePath"/>
        <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: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>
```

*.cymem Schema

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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="AddrWidth"/>
                    <xs:element type="xs:string" name="Mode"/>
                    <xs:element type="xs:string" name="ModeWidth"/>
                    <xs:element type="xs:byte" name="DummyCycles"/>
                    <xs:element type="xs:string" name="DataWidth"/>
                  </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>
```

References

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

- Device Configurator Guide
- Eclipse IDE for ModusToolbox User Guide
- ModusToolbox Release Notes
- 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.
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.