Stratix 10 Serial Flash Mailbox Client
Intel FPGA IP Core User Guide

Updated for Intel® Quartus® Prime Design Suite: 19.1
Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide.............................3
Signals..................................................................................................................................4
Register Map.....................................................................................................................5
Response Codes...............................................................................................................7
Using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP..................................................8
  Control Status Register Operation...............................................................................8
  Write Operation.............................................................................................................8
  Read Operation..........................................................................................................9
Design Example..............................................................................................................11
  Prerequisites.............................................................................................................12
  Creating Flash Image Containing Bitstreams.............................................................12
  Programming Flash Memory with Image Containing Stratix 10 Serial Flash
    Mailbox Client Intel FPGA IP..................................................................................13
  Reading Status Register of Flash Memory Device....................................................14
  Reading ID of Flash Memory Device.........................................................................15
  Reading ID of Flash Memory Device Using Control Command...............................15
  Performing Erase Operation on Flash Memory........................................................16
  Performing Read Operation on Flash Memory.........................................................16
  Performing Write Operation on Flash Memory.......................................................17
Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide Archives.............17
Document Revision History for the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP
  Core User Guide......................................................................................................18
Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide

The Stratix 10 Serial Flash Mailbox Client Intel FPGA IP core provides access to the low-voltage quad-serial configuration (EPCQ-L) device via Intel® Stratix® 10 devices.

Note: Contact your local Intel sales representative for more information about flash device support other than the EPCQ-L devices.

The Stratix 10 Serial Flash Mailbox Client Intel FPGA IP core supports:

- Direct flash access (write and read) through the Avalon-Memory Map (Avalon-MM) interface
- Control register for other operations through the control status register (CSR) interface in Avalon-MM
- 4 KB data for each write or read operation
- Supported operations (refer to the respective flash device datasheet for a complete list of supported operations):
  - Open
  - Close
  - Set chip select
  - Read data from flash
  - Write data to flash
  - Erase sector
  - Read device register
  - Write device register
  - Send device opcode

Related Information

Introduction to Intel FPGA IP Cores
Provides general information about all Intel FPGA IP cores, including parameterizing, generating, upgrading, and simulating IP cores.
Signals

Figure 1. Signal Block Diagram

Table 1. Signal Description

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Avalon®-MM Control Status Register Signals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>csr_address</td>
<td>7</td>
<td>Input</td>
<td>Avalon-MM address bus. The address bus is in word addressing.</td>
</tr>
<tr>
<td>csr_read</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM read control to the CSR.</td>
</tr>
<tr>
<td>csr_readdata</td>
<td>32</td>
<td>Output</td>
<td>Avalon-MM read data bus from the CSR.</td>
</tr>
<tr>
<td>csr_write</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM write control to the CSR.</td>
</tr>
<tr>
<td>csr_writedata</td>
<td>32</td>
<td>Input</td>
<td>Avalon-MM write data bus to CSR.</td>
</tr>
<tr>
<td>csr_waitrequest</td>
<td>1</td>
<td>Output</td>
<td>Avalon-MM waitrequest control from the CSR</td>
</tr>
<tr>
<td>csr_readdata_valid</td>
<td>1</td>
<td>Output</td>
<td>Avalon-MM read data valid that indicates the CSR read data is available.</td>
</tr>
<tr>
<td><strong>Avalon-MM Write Data Signals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wr_mem_write</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM write control to the memory</td>
</tr>
<tr>
<td>wr_mem_address</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM address.</td>
</tr>
<tr>
<td>wr_mem_writedata</td>
<td>32</td>
<td>Input</td>
<td>Avalon-MM write data bus to the memory.</td>
</tr>
<tr>
<td>wr_mem_waitrequest</td>
<td>1</td>
<td>Output</td>
<td>Avalon-MM waitrequest control from the memory.</td>
</tr>
<tr>
<td><strong>Avalon-MM Read Data Signals</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>continued...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Signal

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rd_mem_read</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM read control to the memory</td>
</tr>
<tr>
<td>rd_mem_readdata</td>
<td>32</td>
<td>Output</td>
<td>Avalon-MM read data bus from the memory.</td>
</tr>
<tr>
<td>rd_mem_readdata_valid</td>
<td>1</td>
<td>Output</td>
<td>Avalon-MM read data valid that indicates the memory read data is available.</td>
</tr>
<tr>
<td>rd_mem_address</td>
<td>1</td>
<td>Input</td>
<td>Avalon-MM address.</td>
</tr>
</tbody>
</table>

### Clock and Reset

<table>
<thead>
<tr>
<th>Signal</th>
<th>Width</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>1</td>
<td>Input</td>
<td>Input clock to clock the IP core. The maximum frequency supported for this clock is 250 MHz.</td>
</tr>
<tr>
<td>reset</td>
<td>1</td>
<td>Input</td>
<td>Synchronous reset to reset the IP core. Note: For IP core instantiation guidelines, you must refer to the Intel Stratix 10 Reset Release IP section in the Intel Stratix 10 Configuration User Guide.</td>
</tr>
</tbody>
</table>

### Related Information

- Using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP on page 8
- Intel Stratix 10 Configuration User Guide Provides more information about Intel Stratix 10 Reset Release IP.

### Register Map

**Table 2. Register Map and Definitions**

- Each address offset in the table represents one word of memory address space.
- All registers have a default value of 0x0 unless otherwise stated.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
<th>Field Name</th>
<th>R/W</th>
<th>Width</th>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>STATUS</td>
<td>Rsp_status</td>
<td>R</td>
<td>11</td>
<td>10:0</td>
<td>The status of executed commands.</td>
</tr>
<tr>
<td>1</td>
<td>ISR</td>
<td>Rddata_valid</td>
<td>R</td>
<td>1</td>
<td>1</td>
<td>Interrupt status register. Value 1 indicates that readdata is in FIFO. You can read the fill level to check how many data are available and start reading out the data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cmd_err</td>
<td>R</td>
<td>1</td>
<td>0</td>
<td>Value 1 indicate that the write transaction is not completed successfully.</td>
</tr>
<tr>
<td>2</td>
<td>IEN(1)</td>
<td>Rdat_valid_en</td>
<td>R/W</td>
<td>1</td>
<td>1</td>
<td>Interrupt status register. Write 1 to enable read data valid.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cmd_err_en</td>
<td>R/W</td>
<td>1</td>
<td>0</td>
<td>Write 1 to enable command error response.</td>
</tr>
<tr>
<td>3</td>
<td>CHIP_SELECT</td>
<td>Chip_select</td>
<td>R/W</td>
<td>4</td>
<td>3:0</td>
<td>Write the value of the EPCQ-L device you want to select.</td>
</tr>
<tr>
<td>4</td>
<td>OPEN</td>
<td>Open</td>
<td>W</td>
<td>1</td>
<td>0</td>
<td>Request exclusive access to the EPCQ-L device. Write 1 to request exclusive access. The IP returns OK if SDM accepts request.</td>
</tr>
</tbody>
</table>

(1) Default value is 0x1.
<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
<th>Field Name</th>
<th>R/W</th>
<th>Width</th>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>CLOSE</td>
<td>Close</td>
<td>W</td>
<td>1</td>
<td>0</td>
<td>Close exclusive access to the EPCQ-L device. Write 1 to close exclusive access. The IP returns OK if SDM accepts request.</td>
</tr>
<tr>
<td>6</td>
<td>WR_ENABLE</td>
<td>Wr_enable</td>
<td>W</td>
<td>1</td>
<td>0</td>
<td>Write 1 to perform write enable operation to the device.</td>
</tr>
<tr>
<td>7</td>
<td>WR_STATUS</td>
<td>Wr_status</td>
<td>W</td>
<td>8</td>
<td>7:0</td>
<td>Write a value to the status register of the device.</td>
</tr>
<tr>
<td>8</td>
<td>Rd_STATUS</td>
<td>Rd_status</td>
<td>R</td>
<td>8</td>
<td>7:0</td>
<td>Read the status register of the device. This field contains the information from read status register operation.²</td>
</tr>
<tr>
<td>9</td>
<td>SECTOR_ERASE</td>
<td>Sector_address</td>
<td>W</td>
<td>32</td>
<td>0:31</td>
<td>Erases one sector in the flash. The address of sector to be erase.</td>
</tr>
<tr>
<td>10</td>
<td>RD_DEVICE_ID</td>
<td>Device_id</td>
<td>R</td>
<td>32</td>
<td>0:31</td>
<td>Read this address to obtain the device ID.</td>
</tr>
<tr>
<td>11 - 12</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>CONTROL</td>
<td>Opcode</td>
<td>R/W</td>
<td>8</td>
<td>31:24</td>
<td>Opcode of the EPCQ-L device operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read_data_en</td>
<td>R/W</td>
<td>1</td>
<td>6</td>
<td>Value 1 indicates the command has read data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Write_data_en</td>
<td>R/W</td>
<td>1</td>
<td>5</td>
<td>Value 1 indicates the command has write data.</td>
</tr>
<tr>
<td></td>
<td>Execute</td>
<td>W</td>
<td>1</td>
<td>0</td>
<td></td>
<td>Write 1 to execute the command.</td>
</tr>
<tr>
<td>14</td>
<td>NUMB_BYTES</td>
<td>Number_bytes</td>
<td>R/W</td>
<td>4</td>
<td>3:0</td>
<td>The number of bytes to write or read to device register (maximum 8 bytes).</td>
</tr>
<tr>
<td>15</td>
<td>WRITEDATA_0</td>
<td>Writedata_0</td>
<td>W</td>
<td>32</td>
<td>31:0</td>
<td>The lower 4 bytes of write data.</td>
</tr>
<tr>
<td>16</td>
<td>WRITEDATA_1</td>
<td>Writedata_1</td>
<td>W</td>
<td>32</td>
<td>31:0</td>
<td>The upper 4 bytes of write data.</td>
</tr>
<tr>
<td>17</td>
<td>READDATA_0</td>
<td>Readdata_0</td>
<td>R</td>
<td>32</td>
<td>31:0</td>
<td>The lower 4 bytes of read data.</td>
</tr>
<tr>
<td>18</td>
<td>READDATA_1</td>
<td>Readdata_1</td>
<td>R</td>
<td>32</td>
<td>31:0</td>
<td>The upper 4 bytes of read data.</td>
</tr>
<tr>
<td>19</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>WRITE_OP</td>
<td>Write_op</td>
<td>W</td>
<td>2</td>
<td>1:0</td>
<td>Write 2'b01 to perform write operation with address provided in offset 21 and write data in FIFO. Write 2'b10 to flush out data inside write FIFO.</td>
</tr>
<tr>
<td>21</td>
<td>WRITE_ADDR</td>
<td>Write_addr</td>
<td>W</td>
<td>32</td>
<td>31:0</td>
<td>The device address for write operation.</td>
</tr>
<tr>
<td>22</td>
<td>WRITE_FIFO_LEVEL</td>
<td>Wr_fifo_level</td>
<td>R</td>
<td>32</td>
<td>31:0</td>
<td>Returns the fill level of the internal write data FIFO.</td>
</tr>
<tr>
<td>23</td>
<td>READ_OP</td>
<td>Read_op</td>
<td>W</td>
<td>2</td>
<td>1:0</td>
<td>Write 2'b01 to perform read operation with address provided in offset 24 Write 2'b10 to flush out data inside read FIFO.</td>
</tr>
</tbody>
</table>

² To check the status of stacked devices, refer to the flag status register. You can access the flag status register using the CONTROL command.
### Related Information

- [Using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP on page 8](#)
- [Response Codes on page 7](#)

### Response Codes

The Stratix 10 Serial Flash Mailbox Client Intel FPGA IP core returns response code for each command you execute together with read data, if applicable.

#### Table 3. Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Response Codes

You must check the response to ensure that the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP core receives the command.

<table>
<thead>
<tr>
<th>Value (Hex)</th>
<th>Error Code Response</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OK</td>
<td>Indicates that the command completed successfully. Depending on the command delivered to the Mailbox Client, the response error code may not be sufficient to ensure that the operation completed successfully.</td>
</tr>
<tr>
<td>1</td>
<td>INVALID_COMMAND</td>
<td>Indicates that the command is in an incorrect format.</td>
</tr>
<tr>
<td>2</td>
<td>UNKNOWN_BR</td>
<td>Indicates that the command code is not understood. This error may occur if you have deselected the <em>Use the factory default helper image</em> on the Programmer Tools -&gt; Options menu.</td>
</tr>
<tr>
<td>3</td>
<td>UNKNOWN</td>
<td>Indicates that the command code is not understood by the currently loaded firmware.</td>
</tr>
<tr>
<td>100</td>
<td>NOT_CONFIGURED</td>
<td>Indicates that the device is not configured.</td>
</tr>
<tr>
<td>1FF</td>
<td>ALT_SDM_MBOX_RESP_DEVICE_BUSY</td>
<td>Indicates that the device is busy.</td>
</tr>
<tr>
<td>2FF</td>
<td>ALT_SDM_MBOX_RESP_NO_VALID_RESP_AVAILABLE</td>
<td>Indicates that there is no valid response available.</td>
</tr>
<tr>
<td>3FF</td>
<td>ALT_SDM_MBOX_RESP_ERROR</td>
<td>General Error</td>
</tr>
</tbody>
</table>

### Related Information

- [Using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP on page 8](#)
- [Summary of Operation Codes in EPCQ-A Serial Configuration Device Datasheet](#)
- [Register Map on page 5](#)
Using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP

The following guidelines list the flow of operations you must execute for CSR, write, and read operations using the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP core. All interfaces are Avalon-MM compliant. Refer to the Avalon Interface Specification for more information.

Related Information
- Signals on page 4
- Register Map on page 5
- Response Codes on page 7

Control Status Register Operation

The following steps are guidelines to perform a read or write to a specific address offset using the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP CSR.

1. Assert the csr_write or csr_read signals while the csr_waitrequest signal is low. If the csr_waitrequest signal is high, the csr_write or csr_read signals must be kept high until the csr_waitrequest signal goes low.

2. Depending on the operation, perform the following steps:
   - For read operations, set the address value on the csr_address bus.
   - For write operations, set the address value on the csr_address bus and the value data on the csr_writedata bus.

3. For read operations, you can retrieve the data after the csr_readdatavalid signal is high.

Write Operation

The following steps are guidelines to perform a write operation using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP IP core. The maximum size of data for each write operation is limited to 4K byte.

1. Note: This step is required only if the flash device is filled.
   Erase the flash device using the SECTOR_ERASE command.

2. Flush out all the write data FIFO by writing 2b’10 to the WRITE_OP command.

3. Pre-store the data you want to write to the flash device in the write data FIFO via write data interfaces. Write the write data FIFO:
   - Assert the wr_mem_write signal while the wr_mem_waitrequest signal is low. If the wr_mem_waitrequest signal is high, the wr_mem_write signal must be kept high until the wr_mem_waitrequest signal goes low.
   - Write the address value at the wr_mem_address bus and write the data value to wr_mem_writedata bus.
   - Repeat step a to b to continuously pre-store the data into the write data FIFO.

Note: Refer to the base address assigned to wr_mem bus for Stratix 10 Serial Flash Mailbox Client Intel FPGA IP in the Intel Quartus® Prime Platform Designer for list of address values that you can write into.
d. De-assert the `wr_mem_write` signal once you have completed writing the data into the write data FIFO.

e. Optional: You can read the fill level of the internal write data FIFO using the `WRITE_FIFO_LEVEL` command to know if the write data FIFO is filled up.

4. Specify the start address of the location you want to write the data into the flash by using the `WRITE_ADDR` command.

5. Start the write operation by transferring the data from the write data FIFO into the flash device by writing `2'b01` to the `WRITE_OP` command.

6. Poll the `ISR` status to check the status of the write transaction. The `ISR` is triggered if the transaction is not successful.

7. Repeat step 1 to 6 to continue to perform the subsequent write operation.

   *Note:* You can poll the `CMD_STATUS` register each time you send a command to ensure that the command is successfully executed.

---

**Figure 2. Write Operation Example Timing Diagram**

```
clk

write

waitrequest

address A0 A1 A2

writedata D0 D1 D2
```

*Note:* You can use the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP IP to write the raw programming data (.rpD) file into the EPCQ-L device. Refer to the *Generating Programming Files using Convert Programming Files* in the *Intel Stratix 10 Configuration User Guide* for more information about programming the EPCQ-L device.

**Related Information**

- Summary of Operation Codes in EPCQ-A Serial Configuration Device Datasheet
- Generating Programming Files using Convert Programming Files of the Intel Stratix 10 Configuration User Guide

## Read Operation

The following steps are guidelines to perform a read operation using Stratix 10 Serial Flash Mailbox Client Intel FPGA IP IP core. The maximum size of data for each read operation is limited to 4K byte.
1. Specify the flash device address to read the data by using the `READ_ADDR` command.

2. Specify the number of words to read by using the `READ_WORDS` command. The maximum number of words can be read per transaction is 1024 (32'h00000400).

3. Flush out all the read data FIFO before you perform any read operations by writing 2b'10 to the `READ_OP` command.

4. Start the read operation by transferring data from flash to read data FIFO by writing 2'b01 to the `READ_OP` command.

5. Poll the `ISR` register to check if the data stored in read data FIFO is ready to read. You can also read the fill level of the internal read data FIFO using the `READ_FIFO_LEVEL` command to know if there is any data stored inside the read data FIFO.

6. Read the data stored in read data FIFO via read data interfaces. Refer to the following steps to read the read data FIFO:
   a. Assert the `rd_mem_read` signal while the `rd_mem_waitrequest` signal is low. If the `rd_mem_waitrequest` signal is high, the `rd_mem_read` signal must be kept high until the `wr_mem_waitrequest` signal goes low.
   b. Set the address value at the `rd_mem_address` bus.
      
      Note: Refer to the base address assigned to `rd_mem` bus for Stratix 10 Serial Flash Mailbox Client Intel FPGA IP in the Intel Quartus Prime Platform Designer for list of address values that you can read into.
   c. Read the `rd_mem_readdata` bus if `rd_mem_readdatavalid` signal is asserted high.
   d. Repeat step a to c to continuously read the data from the read data FIFO.
   e. De-assert the `rd_mem_read` signal once you have completed reading the data from the read data FIFO.
   f. Optional: You can read the fill level of the internal read data FIFO using the `READ_FIFO_LEVEL` command.

7. Repeat step 1 to 6 to continue to perform subsequent read operations.

Note: You can poll the `CMD_STATUS` register each time you send a command to ensure that the command is successfully executed.
**Figure 3. Read Operation Example Timing Diagram**

- clk
- read
- waitrequest
- address: A0, A1, A2
- readdatavalid
- readdata: D0, D1, D2

**Related Information**

*Summary of Operation Codes in EPCQ-A Serial Configuration Device Datasheet*

**Design Example**

The Stratix 10 Serial Flash Mailbox Client Intel FPGA IP example design includes the following functions:

- Creates flash Image containing bitstream.
- Programs the flash memory with image containing Stratix 10 Serial Flash Mailbox Client Intel FPGA IP.
- Reads the status register of the flash memory device.
- Reads the ID of the flash memory device.
- Reads the ID of the flash memory device using control command.
- Performs erase operation on the flash memory.
- Performs read operation on the flash memory.
- Performs write operation on the flash memory.
Prerequisites

To run Serial Flash Mailbox Client Intel FPGA IP design example, your system must meet the following hardware and software requirements:

  - To create a hardware project with JTAG master as host, instantiate a Serial Flash Mailbox Client Intel FPGA IP and connect to JTAG to Avalon Master Bridge.
  - Set base address for csr, rd_mem, and wr_mem.

Figure 4. Required Communication and Host Components for the Serial Flash Mailbox Client Intel FPGA IP Design Example

- Intel Stratix 10 SoC Development Kit

Creating Flash Image Containing Bitstreams

1. On the File menu, click Convert Programming Files.
2. Select JTAG Indirect Configuration File (.jic) from Programming file type drop-down list.
3. Select the configuration device from Configuration device drop-down list.
   Note: The configuration device depends on the type of configuration device used in the board.
4. Select Active Serial from Mode drop-down list.
5. Add SOF programming file into SOF Data.
6. Add device used into Flash Loader.
   Note: The device used must be same as the device used in Intel Quartus Prime Pro Edition software.
7. Click **Generate** to generate the flash image containing bitstreams.

**Programming Flash Memory with Image Containing Stratix 10 Serial Flash Mailbox Client Intel FPGA IP**

1. Open **Programmer**, click **Add File**, select the generated JIC programming file (.jic) and click **Open**.
2. Check the **Program/Configure** check box for the attached .jic file.
3. To begin programming the flash memory with the image containing Stratix 10 Serial Flash Mailbox Client Intel FPGA IP, click **Start**.
4. Configuration is complete when the progress bar reaches 100%. Power cycle the board and the Intel Stratix 10 device is automatically configured with the image file via the Active Serial configuration scheme.
Reading Status Register of Flash Memory Device

This guideline describes the sequences and commands to read the status register of the flash memory device.

```bash
# Set base address according to Platform Designer system
set CSR 0x00000000
set WR_MEM 0x00000240
set RD_MEM 0x00000248
# Set the variables to their respective offset
set AsmiCmdStatus          [expr $CSR + [expr 0x0  << 2]]
set AsmiIsr                    [expr $CSR + [expr 0x1  << 2]]
set AsmiIer                   [expr $CSR + [expr 0x2  << 2]]
set AsmiChipSelect             [expr $CSR + [expr 0x3  << 2]]
set AsmiOpen               [expr $CSR + [expr 0x4  << 2]]
set AsmiClose              [expr $CSR + [expr 0x5  << 2]]
set AsmiWrEnable               [expr $CSR + [expr 0x6  << 2]]
set AsmiWrStatus               [expr $CSR + [expr 0x7  << 2]]
set AsmiRdStatus              [expr $CSR + [expr 0x8  << 2]]
set AsmiSectorErase        [expr $CSR + [expr 0x9  << 2]]
set AsmiDeviceId              [expr $CSR + [expr 0xa  << 2]]
set AsmiControl              [expr $CSR + [expr 0xd  << 2]]
set AsmiNumbByte          [expr $CSR + [expr 0xe  << 2]]
set AsmiWriteData0         [expr $CSR + [expr 0xf  << 2]]
set AsmiWriteData1         [expr $CSR + [expr 0x10  << 2]]
set AsmiReadData0          [expr $CSR + [expr 0x11  << 2]]
set AsmiReadData1          [expr $CSR + [expr 0x12  << 2]]
set AsmiWriteOp              [expr $CSR + [expr 0x14  << 2]]
set AsmiWriteAddr            [expr $CSR + [expr 0x15  << 2]]
set AsmiWriteFifoLevel       [expr $CSR + [expr 0x16  << 2]]
set AsmiReadOp                [expr $CSR + [expr 0x17  << 2]]
set AsmiReadAddr             [expr $CSR + [expr 0x18  << 2]]
set AsmiReadNumbWords        [expr $CSR + [expr 0x19  << 2]]
set AsmiReadFifoLevel        [expr $CSR + [expr 0x1A  << 2]]
# Assign variable "m" to the string that is the 0th element in the list returned by get_service_paths master
set m [ lindex [ get_service_paths master ] 0 ]
# Open the connection to the master module
```
open_service master $m

# Assign variable "$m" to the string that is the 0th element in the list
# returned by get_service_paths master
set m [ lindex [ get_service_paths master ] 0 ]

# Open the connection to the master module
open_service master $m

# Write Offset 4 to request access to the flash memory device.
master_write_32 $m $AsmiOpen 0x1

# Write Offset 3 to select the 1st flash memory device attached to the IP.
master_write_32 $m $AsmiChipSelect 0x0

# Read Offset 8 for status register of the device.
master_read_32 $m $AsmiRdStatus 1

# Write Offset 5 to close access to the flash memory device.
master_write_32 $m $AsmiClose 0x1

---

### Reading ID of Flash Memory Device

This guideline describes the sequences and commands to read the ID of the flash memory device.

# Write Offset 4 to request access to the flash memory device.
master_write_32 $m $AsmiOpen 0x1

# Write Offset 3 to select the 1st flash memory device attached to the IP.
master_write_32 $m $AsmiChipSelect 0x0

# Read Offset 10 to obtain the device ID.
master_read_32 $m $AsmiDeviceId 1

# Write Offset 5 to close access to the flash memory device.
master_write_32 $m $AsmiClose 0x1

---

### Reading ID of Flash Memory Device Using Control Command

This guideline describes the sequences and commands to read the ID of the flash memory device using the control command.

# Write Offset 4 to request access to the flash memory device.
master_write_32 $m $AsmiOpen 0x1

# Write Offset 3 to select the 1st flash memory device attached to the IP.
master_write_32 $m $AsmiChipSelect 0x0

# Writing the command argument to Offset 14 (specify the number bytes to 0x4)
master_write_32 $m $AsmiNumbByte 0x4

# Writing the command argument to Offset 13 (the opcode to read device ID is 0xAF000041)
master_write_32 $m $AsmiControl 0xAF000041

# Read Offset 17 to determine the lower 4 bytes of read data where the device ID obtained from control command is stored at.
master_read_32 $m $AsmiReadData0 1

# Write Offset 5 to close access to the flash memory device.
master_write_32 $m $AsmiClose 0x1

---
Performing Erase Operation on Flash Memory

This guideline describes the sequences and commands to perform sector erase operation on the flash memory.

```c
# Write Offset 4 to request access to the flash memory device.
master_write_32 $m $AsmiOpen 0x1

# Write Offset 3 to select the 1st flash memory device attached to the IP.
master_write_32 $m $AsmiChipSelect 0x0

# Write Offset 6 to perform write enable operation to the device.
master_write_32 $m $AsmiWrEnable 0x1

# Writing the address argument to Offset 9 (Perform Sector Erase on address 0x03FF0000)
master_write_32 $m $AsmiSectorErase 0x03FF0000

# Write Offset 5 to close access to the flash memory device.
master_write_32 $m $AsmiClose 0x1
```

Performing Read Operation on Flash Memory

This guideline describes the sequences and commands to read one word of data from flash memory.

```c
# Write Offset 4 to request access to the flash memory device.
master_write_32 $m $AsmiOpen 0x1

# Write Offset 3 to select the 1st flash memory device attached to the IP.
master_write_32 $m $AsmiChipSelect 0x0

# Writing the address argument to Offset 24 (specify the device address for read operation to 0x03FF0000)
master_write_32 $m $AsmiReadAddr 0x03FF0000

# Writing the command argument to Offset 25 (specify the number of words to read from device is 1)
master_write_32 $m $AsmiReadNumbWords 0x1

# Writing the command argument to Offset 23 (Specify 0x2 to flush out data inside read FIFO)
master_write_32 $m $AsmiReadOp 0x2

# Writing the command argument to Offset 23 (Specify 0x1 to perform read operation)
master_write_32 $m $AsmiReadOp 0x1

# Read Offset 26 to determine the fill level of the internal read data FIFO.
master_read_32 $m $AsmiReadFifoLevel 1

# Read the data stored in read data FIFO via the base address of rd_mem in the IP.
master_read_32 $m $RD_MEM 1

# Write Offset 5 to close access to the flash memory device.
master_write_32 $m $AsmiClose 0x1
```
Performing Write Operation on Flash Memory

This guideline describes the sequences and commands to write one word of data into flash memory.

- Write Offset 4 to request access to the flash memory device.
  ```c
  master_write_32 $m $AsmiOpen 0x1
  ```
- Write Offset 3 to select the 1st flash memory device attached to the IP.
  ```c
  master_write_32 $m $AsmiChipSelect 0x0
  ```
- Write Offset 6 to perform write enable operation to the device.
  ```c
  master_write_32 $m $AsmiWrEnable 0x1
  ```
- Writing the command argument to Offset 20 (Specify 0x2 to flush out data inside write FIFO)
  ```c
  master_write_32 $m $AsmiWriteOp 0x2
  ```
- Pre-store the data that you want to write into flash memory in write data FIFO via the base address of wr_mem in the IP (Specify 0x11223344 to write into write data FIFO)
  ```c
  master_write_32 $m $WR_MEM 0x11223344
  ```
- Read Offset 22 to determine the fill level of the internal write data FIFO.
  ```c
  master_read_32 $m $AsmiWriteFifoLevel 1
  ```
- Writing the address argument to Offset 21 (specify the device address for write operation to 0x03FF0000)
  ```c
  master_write_32 $m $AsmiWriteAddr 0x03FF0000
  ```
- Writing the command argument to Offset 20 (Specify 0x1 to perform write operation)
  ```c
  master_write_32 $m $AsmiWriteOp 0x1
  ```
- Write Offset 5 to close access to the flash memory device.
  ```c
  master_write_32 $m $AsmiClose 0x1
  ```

### Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide Archives

If an IP core version is not listed, the user guide for the previous IP core version applies.

<table>
<thead>
<tr>
<th>IP Core Version</th>
<th>User Guide</th>
</tr>
</thead>
<tbody>
<tr>
<td>18.0</td>
<td>Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide</td>
</tr>
</tbody>
</table>
# Document Revision History for the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Core User Guide

<table>
<thead>
<tr>
<th>Document Version</th>
<th>Intel Quartus Prime Version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| 2019.05.17       | 19.1                       | • Updated Table: Signal Description to add a note regarding IP core instantiation guidelines to the reset signal.  
• Added the Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Archives topic. |
| 2019.04.01       | 19.1                       | Updated Table: Register Map and Definitions to update the field name for ISR from Cmd_err1 to Cmd_err. |
| 2018.12.24       | 18.0                       | • Added a design example section.  
• Updated Table: Register Map and Definitions.  
• Updated Table: Stratix 10 Serial Flash Mailbox Client Intel FPGA IP Response Codes.  
• Corrected minor typographical errors. |
| 2018.05.07       | 18.0                       | Initial release. |