## Contents

1 Revision History ................................................................. 1
   1.1 Revision 4.0 ................................................................. 1
   1.2 Revision 3.0 ................................................................. 1
   1.3 Revision 2.0 ................................................................. 1
   1.4 Revision 1.0 ................................................................. 1

2 SmartFusion2 and IGLOO2 - Accessing eNVM and eSRAM from FPGA Fabric . . . 2
   2.1 Accessing eNVM from FPGA Fabric ............................................. 2
   2.2 Accessing eSRAM from FPGA Fabric ............................................. 2
   2.3 Design Requirements ............................................................... 2
   2.4 Design Description ................................................................. 2
   2.5 Hardware Implementation ......................................................... 3
       2.5.1 SmartDesign Components .................................................. 3
   2.6 Simulation .............................................................................. 10
       2.6.1 eNVM Simulation ............................................................. 10
       2.6.2 eSRAM Simulation ............................................................. 11
   2.7 Setting Up the Design ............................................................... 11
   2.8 Programming the Demo Design ................................................... 13
   2.9 Configuring the Device ............................................................ 14
   2.10 Running the Design .............................................................. 16
       2.10.1 eNVM Write and Read Operations ....................................... 16
       2.10.2 eSRAM Write and Read Operation ....................................... 19
   2.11 Conclusion ............................................................................. 19

3 Appendix: Design Files ............................................................ 20

4 Appendix: eNVM and eSRAM Write/Read Operations ................................. 21
## Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Top-Level SmartDesign for IGLOO2</td>
<td>3</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Top-Level SmartDesign for SmartFusion2</td>
<td>3</td>
</tr>
<tr>
<td>Figure 3</td>
<td>IGLOO2—Device Features Page</td>
<td>4</td>
</tr>
<tr>
<td>Figure 4</td>
<td>IGLOO2—Memories Page</td>
<td>5</td>
</tr>
<tr>
<td>Figure 5</td>
<td>IGLOO2—Peripherals Page</td>
<td>6</td>
</tr>
<tr>
<td>Figure 6</td>
<td>IGLOO2—Clocks Page</td>
<td>6</td>
</tr>
<tr>
<td>Figure 7</td>
<td>SmartFusion2—Device Features Page</td>
<td>7</td>
</tr>
<tr>
<td>Figure 8</td>
<td>SmartFusion2—Memories Page</td>
<td>8</td>
</tr>
<tr>
<td>Figure 9</td>
<td>SmartFusion2—Peripherals Page</td>
<td>8</td>
</tr>
<tr>
<td>Figure 10</td>
<td>SmartFusion2—Clocks Page</td>
<td>9</td>
</tr>
<tr>
<td>Figure 11</td>
<td>TPSRAM Configuration</td>
<td>9</td>
</tr>
<tr>
<td>Figure 12</td>
<td>eNVM Write Command Sequence—1</td>
<td>10</td>
</tr>
<tr>
<td>Figure 13</td>
<td>eNVM Write Command Sequence—2</td>
<td>10</td>
</tr>
<tr>
<td>Figure 14</td>
<td>eNVM Write Command Sequence—3</td>
<td>10</td>
</tr>
<tr>
<td>Figure 15</td>
<td>eNVM Read Simulation</td>
<td>10</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Release Exclusive Access to eNVM</td>
<td>10</td>
</tr>
<tr>
<td>Figure 17</td>
<td>eSRAM Write Simulation</td>
<td>11</td>
</tr>
<tr>
<td>Figure 18</td>
<td>eSRAM Read Simulation</td>
<td>11</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Accessing eNVM and eSRAM Demo Setup</td>
<td>13</td>
</tr>
<tr>
<td>Figure 20</td>
<td>FlashPro—New Project</td>
<td>14</td>
</tr>
<tr>
<td>Figure 21</td>
<td>FlashPro—Project Configuration</td>
<td>14</td>
</tr>
<tr>
<td>Figure 22</td>
<td>FlashPro—RUN PASSED</td>
<td>15</td>
</tr>
<tr>
<td>Figure 23</td>
<td>SmartDebug Window</td>
<td>16</td>
</tr>
<tr>
<td>Figure 24</td>
<td>Debug FPGA Array</td>
<td>17</td>
</tr>
<tr>
<td>Figure 25</td>
<td>Select Debug FPGA Array</td>
<td>17</td>
</tr>
<tr>
<td>Figure 26</td>
<td>Debug FPGA Array Window</td>
<td>18</td>
</tr>
<tr>
<td>Figure 27</td>
<td>Debug FPGA Array - Memory Blocks</td>
<td>18</td>
</tr>
<tr>
<td>Figure 28</td>
<td>Debug FPGA Array—Memory Blocks</td>
<td>19</td>
</tr>
</tbody>
</table>
## Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>2</td>
</tr>
<tr>
<td>Table 2</td>
<td>Jumper Settings for IGLOO2 Evaluation Kit Board</td>
<td>11</td>
</tr>
<tr>
<td>Table 3</td>
<td>Jumper Settings SmartFusion2 Security Evaluation Kit Board</td>
<td>12</td>
</tr>
</tbody>
</table>
1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 4.0
The document was updated for Libero SoC v11.8 software release.

1.2 Revision 3.0
The document was updated for Libero SoC v11.7 software release (SAR 77431).

1.3 Revision 2.0
The following is a summary of the changes in revision 2.0 of this document.

- Updated the document for Libero SoC v11.6 software release (SAR 71803).
- Updated Simulation, page 10 (SAR 64030).

1.4 Revision 1.0
Revision 1.0 was the first publication of this document.
2 SmartFusion2 and IGLOO2 - Accessing eNVM and eSRAM from FPGA Fabric

This application note describes how to access the embedded non-volatile memory (eNVM) and embedded static random access memory (eSRAM) from the FPGA fabric in SmartFusion®2 system-on-chip (SoC) FPGA and IGLOO®2 FPGA devices.

2.1 Accessing eNVM from FPGA Fabric

SmartFusion2 SoC FPGA and IGLOO2 FPGA devices have a maximum of two on-chip 256 KB flash memories called eNVM. The eNVM stores the application code image or data required to be stored by the end application. The eNVM block is interfaced through the eNVM controller to the AHB bus matrix. The eNVM can be initialized by the custom logic in the FPGA fabric (fabric master).

In this application note, the fabric master writes to and reads from the 25th page (address starting from 0x60000C80 to 0x60000CFC) of the eNVM.

For more information about eNVM initialization methods, see AC391: SmartFusion2 SoC FPGA - eNVM Initialization Application Note.

2.2 Accessing eSRAM from FPGA Fabric

SmartFusion2 SoC FPGA and IGLOO2 FPGA devices have two eSRAM blocks, each of 32 KB, for data read and write operations. These eSRAM blocks are interfaced through eSRAM controllers to the AHB bus matrix.

In SmartFusion2 SoC FPGA and IGLOO2 FPGA devices, the eSRAM can be accessed by custom logic in the FPGA fabric (fabric master).

In this application note, the fabric master writes to and reads from 32 eSRAM locations (0x20000000 to 0x20000080).

2.3 Design Requirements

For this application, SoftConsole v4.0 is used. For more information see TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial. The following table lists the design requirements.

<table>
<thead>
<tr>
<th>Table 1 • Design Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Requirements</td>
</tr>
<tr>
<td><strong>Hardware Requirements</strong></td>
</tr>
<tr>
<td>SmartFusion2 Security Evaluation Kit</td>
</tr>
<tr>
<td>IGLOO2 Evaluation Kit:</td>
</tr>
<tr>
<td>- 12 V adapter</td>
</tr>
<tr>
<td>- FlashPro4 programmer</td>
</tr>
<tr>
<td>Host PC or Laptop</td>
</tr>
<tr>
<td><strong>Software Requirements</strong></td>
</tr>
<tr>
<td>Libero® System-on-Chip (SoC)</td>
</tr>
<tr>
<td>SoftConsole</td>
</tr>
</tbody>
</table>

2.4 Design Description

The design examples included with this application note uses the following DIP switches:
• SW5-1 DIP switch: used to start eNVM write and read operations. An incremental data pattern starting from 0x00000000 to 0x0000001F is written to the 25th page of the eNVM.
• SW5-2 DIP switch: used to start eSRAM write and read operations. An incremental data starting from 0xA1B2C300 to 0xA1B2C31F is written to eSRAM locations starting from 0x20000000 to 0x20000080.

During eSRAM or eNVM read operation, the read data from the eNVM or eSRAM is stored in the fabric SRAM. The SmartDebug tool in Libero SoC verifies the write and read operations performed on the eNVM and eSRAM.

The design example uses two RTL FSMs—one FSM provides the eNVM or eSRAM write and read commands. The other FSM is an AHB master that receives these commands and communicates with the selected memory using AHB bus matrix through the FIC_0 (fabric interface controller) interface.

The read operations of the eNVM and the read and write operations of the eSRAM are simple AHB transactions. eNVM write requires a separate set of command sequences. For more information about this, see Appendix: eNVM and eSRAM Write/Read Operations, page 21.

2.5 Hardware Implementation

The hardware implementation involves configuring the device features, memory, peripherals, and clocks pages using System Builder. Configuring the TPSRAM IP and adding fabric logic are done at the top-level using SmartDesign.

The following figure shows the top-level hardware design in SmartDesign for IGLOO2.

*Figure 1 • Top-Level SmartDesign for IGLOO2*

The following figure shows the top-level hardware design in SmartDesign for SmartFusion2.

*Figure 2 • Top-Level SmartDesign for SmartFusion2*

2.5.1 SmartDesign Components

The top-level SmartDesign has four components (as shown in preceding figures):

• eSRAM_eNVM_access_0: System Builder generated component.
• AHB_IF_0: user generated RTL FSM, which performs AHB master function. This FSM interacts with the eNVM and eSRAM controller using AHB switch matrix through the FIC_0 interface.
• eSRAM_eNVM_RW_0: user generated RTL FSM, which takes inputs from the user and provides the required commands to AHB_IF_0 (AHB master).
• TPSRAM_0: fabric IP core. Stores the data read from the eNVM or eSRAM.

2.5.1.1 System Builder Configuration for IGLOO2

For configuring IGLOO2 using System Builder, follow these steps:

1. In the Device Features page, ensure HPMS On-chip Flash Memory (eNVM) and On-chip SRAM (eSRAM) check boxes are checked, as shown in the following figure.

Figure 3 • IGLOO2—Device Features Page
2. In the **Memories** page, add **Zeros_client** to initialize the eNVM with zeros, as shown in the following figure.

**Figure 4 • IGLOO2—Memories Page**
3. In the **Peripherals** page, add **HPMS FIC_0_USER_MASTER** under **Subsystems** to provide the AHBL master interface to the user logic, as shown in the following figure.

**Figure 5 • IGLOO2—Peripherals Page**

![Peripherals Page](image)

4. In the **Clocks** page, select **On-chip 25/50 MHz RC Oscillator** as **System Clock** and **100 MHz** as **HPMS_CLK** frequency, as shown in the following figure.

**Figure 6 • IGLOO2—Clocks Page**

![Clocks Page](image)
For more information on how to generate a complete System Builder component for IGLOO2, see the IGLOO2 System Builder User Guide.

### 2.5.1.2 System Builder Configuration for SmartFusion2

For configuring SmartFusion2 using System Builder, follow these steps:

1. In the **Device Features** page, ensure **MSS On-chip Flash Memory (eNVM)** check box is selected, as shown in the following figure.

**Figure 7 • SmartFusion2—Device Features Page**

2. In the **Memories** page, add **Zeros_client** and **dummy_client**, as shown in the following figure.
3. In the **Peripherals** page, drag **Fabric AMBA Master** to **MSS FIC_0 - Fabric Master Subsystem**. It provides the AHBL master interface to the user logic, as shown in the following figure.

**Figure 9** • SmartFusion2—Peripherals Page
4. In the **Clocks** page, select **On-chip 25/50 MHz RC Oscillator** as **System Clock** and **100 MHz** as **M3_CLK** frequency, as shown in the following figure.

**Figure 10 • SmartFusion2—Clocks Page**

For more information on how to generate a complete System Builder component for SmartFusion2, see the **SmartFusion2 System Builder User Guide**.

**2.5.1.3 TPSRAM IP Configuration**

In SmartDesign, TPSRAM IP is configured as: write port 32 (depth) × 32 (width) and read port 32 (depth) × 32 (width). The following figure shows the TPSRAM IP configuration.

**Figure 11 • TPSRAM Configuration**
2.6 Simulation

To simulate the design, on the Design Flow tab, right-click Simulate under Verify Pre-Synthesized Design, and select Open Interactively.

2.6.1 eNVM Simulation

The following figures show the eNVM write command sequence. For more information, see Appendix: eNVM and eSRAM Write/Read Operations, page 21.

Figure 12 • eNVM Write Command Sequence—1

Figure 13 • eNVM Write Command Sequence—2

Figure 14 • eNVM Write Command Sequence—3

The following figure shows the eNVM read simulation (AHB read operation).

Figure 15 • eNVM Read Simulation

The following figure shows the release exclusive access to eNVM.

Figure 16 • Release Exclusive Access to eNVM
2.6.2 eSRAM Simulation

The following figure shows the eSRAM write simulation (AHB write operation).

*Figure 17 • eSRAM Write Simulation*

The following figure shows the eSRAM read simulation (AHB read operation).

*Figure 18 • eSRAM Read Simulation*

2.7 Setting Up the Design

The following steps describe how to setup the hardware demo for IGLOO2 Evaluation Kit board:

1. Connect the jumpers on the IGLOO2 Evaluation Kit board according to the following table.

*Table 2 • Jumper Settings for IGLOO2 Evaluation Kit Board*

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Pin (From)</th>
<th>Pin (To)</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>J22</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J23</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J24</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J8</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
<tr>
<td>J3</td>
<td>1</td>
<td>2</td>
<td>Default</td>
</tr>
</tbody>
</table>

*Note:* Ensure the power supply switch, SW7 is switched off while connecting the jumpers on the IGLOO2 Evaluation Kit.

2. Connect the power supply to the J6 connector.
3. Switch on the power supply switch, SW7.
4. Connect the FlashPro4 programmer to the J5 connector of the IGLOO2 Evaluation Kit board.
The following steps describe how to setup the hardware demo for SmartFusion2 Security Evaluation Kit board:

1. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board according to the following table.

### Table 3 • Jumper Settings SmartFusion2 Security Evaluation Kit Board

<table>
<thead>
<tr>
<th>Jumper</th>
<th>Function</th>
<th>Default Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>J23</td>
<td>Selects switch-side MUX inputs of A or B to the line side</td>
<td>Closed</td>
</tr>
<tr>
<td></td>
<td>Pin 1-2 (Input A to the line side) that is on board 125 MHz differential clock oscillator output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>is routed to the line side</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pin 2-3 (Input B to the line side) that is external clock required to source through SMA connectors to the line side</td>
<td>Open</td>
</tr>
<tr>
<td>J22</td>
<td>Selects the output enables control for the line side outputs</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>Pin 1-2 (line side output enabled)</td>
<td>Closed</td>
</tr>
<tr>
<td></td>
<td>Pin 2-3 (line side output disabled)</td>
<td>Open</td>
</tr>
<tr>
<td>J24</td>
<td>Provides VBUS supply to USB when using in Host mode</td>
<td>Open</td>
</tr>
<tr>
<td>J8</td>
<td>JTAG selection jumper to select between RVI header or FP4 header for application debug</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>Pin 1-2 FP4 for SoftConsole/FlashPro</td>
<td>Closed</td>
</tr>
<tr>
<td></td>
<td>Pin 2-3 RVI for Keil™ ULINK™/IAR J-Link®</td>
<td>Open</td>
</tr>
<tr>
<td></td>
<td>Pin 2-4 to remotely toggle JTAG_SEL signal using GPIO capability of FT4232 chip</td>
<td>Open</td>
</tr>
<tr>
<td>J3</td>
<td>Selects SW2 input or ENABLE_FT4232 signal from FT4232H chip</td>
<td>--</td>
</tr>
</tbody>
</table>

### Note:
Ensure the power supply switch, SW7 is switched off while connecting the jumpers on the SmartFusion2 Security Evaluation Kit.

2. Connect the power supply to the J6 connector.
3. Switch on the power supply switch, SW7.
4. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit board.
The following figure shows the demo setup for SmartFusion2 Security Evaluation Kit and IGLOO2 Evaluation Kit.

### Figure 19 • Accessing eNVM and eSRAM Demo Setup

#### 2.8 Programming the Demo Design

The following steps describe how to program the demo design:

1. Download the demo design from the following link:
   
   http://soc.microsemi.com/download/rsc/?f=m2s_m2gl_ac429_liberov11p8_df

2. Switch on the power supply switch, SW7.
3. Launch the FlashPro software.
4. Click **New Project**, as shown in the following figure.
5. In the **New Project** window, enter the project name as **eNVM_eSRAM_RW**.
6. Click **Browse** and navigate to the location where the project needs to be saved.
7. Select **Single device** as the programming mode.
8. Click **OK** to save the project.
2.9 Configuring the Device

The following steps describe how to configure the device:

1. On the FlashPro window, click **Configure Device**, as shown in the following figure.
2. Click **Browse** and navigate to the location where the eSRAM_eNVM_access_top.stp programming file is located, and select the file.
   The default location of the programming file is:
   - For SmartFusion2: `<download folder>eSRAM_eNVM_RW_Fabric\Programming_file\SF2\eSRAM_eNVM_access_top.stp`
   - For IGLOO2: `<download folder>eSRAM_eNVM_RW_Fabric\Programming_file\IGL2\eSRAM_eNVM_access_top.stp`
3. Select **PROGRAM** as **Action** and click the **PROGRAM** button, as shown in the following figure.

---

![Figure 20 • FlashPro—New Project](image1)

![Figure 21 • FlashPro—Project Configuration](image2)
4. Wait until the **Programmer Status** changes to **RUN PASSED**, as shown in the following figure.

**Figure 22** • FlashPro—**RUN PASSED**
2.10 Running the Design

The design can be run to perform the eNVM and eSRAM write/read operations.

2.10.1 eNVM Write and Read Operations

The following steps describe how to read from and write to the eNVM:

1. Make 1 to 0 transition using the SW5-1 DIP switch (FPGA pin number: L19) to write to and read from the eNVM. An incremental data starting from 0x00000000 to 0x0000001F is written to page 25 of the eNVM, and the data is read back from the eNVM and stored in the fabric SRAM. For more information about eNVM write operation, see Appendix: eNVM and eSRAM Write/Read Operations, page 21.

   The write and read operations can be verified using the SmartDebug tool in Libero SoC.

2. In Libero SoC, go to Design Flow > Program and Debug Design > Debug Design > SmartDebug Design. Right-click and select Open Interactively. The SmartDebug window opens up, as shown in the following figure.

3. Click Debug FPGA Array.

   The the Debug FPGA Array window opens, as shown in the following figure.
4. Click **Memory Blocks** tab to list the available memory blocks.
5. Select the desired memory block and click **Select**, as shown in the following figure.

**Figure 24 • Debug FPGA Array**

6. Click **Read Block**, as shown in the following figure.
Figure 26 • Debug FPGA Array Window

The fabric SRAM (TPSRAM) memory content is displayed, as shown in the following figure. If the initial data written to the eNVM matches the displayed data, eNVM write and read are successful.

Figure 27 • Debug FPGA Array - Memory Blocks

7. Close the Debug FPGA Array window and the SmartDebug window.

For more information on how to interpret the memory block data, go to Help > Help Topics > Debug Design > SmartFusion2 and IGLOO2 SmartDebug > Debug FPGA Array > Memory Blocks.
2.10.2 eSRAM Write and Read Operation

The following steps describe how to perform read and write operations from and to the eSRAM:

1. Make 1 to 0 transition using the SW5-2 DIP switch (FPGA pin number: L18) to write to and read from the eSRAM. An incremental data pattern starting from 0xA1B2C300 to 0xA1B2C31F is written to 32 locations in eSRAM address starting from 0x20000000 to 0x20000080, and the data is read back from the eSRAM and stored in the fabric SRAM.

2. To verify the write and read operations using the SmartDebug tool in Libero SoC, follow the steps 2 to 8 in eNVM Write and Read Operations, page 16. The fabric SRAM (TPSRAM) memory content is displayed, as shown in the following figure. If the initial data written to the eSRAM matches the displayed data, eSRAM write and read operations are successful.

Figure 28 • Debug FPGA Array—Memory Blocks

For more information on how to interpret the memory block data, go to Help > Help Topics > Debug Design > SmartFusion2 and IGLOO2 SmartDebug > Debug FPGA Array > Memory Blocks.

2.11 Conclusion

This application note described how to write to and read from the eNVM and eSRAM from the FPGA fabric. It also described the usage of the SmartDebug tool to verify the design functionality.
Appendix: Design Files

Download the design files from the Microsemi SoC products group website:
http://soc.microsemi.com/download/rsc/?f=m2s_m2gl_ac429_liberov11p8_df

The design file consists of Libero Verilog projects and programming files (*.stp) for SmartFusion2 Security Evaluation Kit and IGLOO2 FPGA Evaluation Kit. See the Readme.txt file included in the design file for the directory structure and description.
4 Appendix: eNVM and eSRAM Write/Read Operations

The following steps describe how eNVM write operation is performed:

1. Wait for Bit 0 of status register (address: 0x60080120) to become 1. If this bit is 0, it implies that the eNVM is busy.
2. Request exclusive access to the eNVM. This is required to ensure no two masters can write to the eNVM at the same time. This is done by writing 0x1 to the REQACCESS register (address: 0x600801FC).
3. Check if the request is granted, by reading back from the REQACCESS register. On read back, check for data bits [2:0], which must be equal to 6.
4. If data bits [2:0] = 0x6, it implies that the request is granted by the fabric.
5. If bit 2 is 0, the request for exclusive access is denied. The eNVM cannot be written at this time.
6. Write 0x00001FF1 to ENVM_CR register (address: 0x4003800C). This changes the FREQRNG register field to 15 decimals.
7. Writes to the eNVM are buffered. First, write data into the write data buffer (WDB), which is a byte addressable 1024-bit buffer. Its base address is 0x60080080 for eNVM_0 and 0x600C0080 for eNVM_1. Then, use a single command to commit (program) data into one page of the eNVM. Write the data into the WDB.
8. Compute the values of bits that need to be written into the eNVM command register.
   • Bits 31-20 must be 0x080.
   • Bit 19 must be 0x0.
   • Bits 18-7 corresponds to the number of page to be written (for 25th page, the bit field is 000 0000 1100 1).
   • Bits 6-0 must be 0x0.
9. Write the eNVM command register (address: 0x60080148) with the data computed in step 8. eNVM does not respond to further commands until the write is complete.
10. Release exclusive access to the eNVM by writing 0x0 to the REQACCESS register (address: 0x600801FC).

**Note:** Special command sequences are not required for eNVM read, eSRAM read, and eSRAM write operations. eNVM read and eSRAM read are performed using AHB read operation, and eSRAM writes are performed using AHB write operation.