Summary

This application note outlines how to use a Zynq® UltraScale+™ MPSoC in conjunction with the LogiCORE™ IP UltraScale+ architecture Soft Error Mitigation (SEM) controller. The reference design provides an example of how the SEM controller is integrated with a processing system (PS). It targets the Xilinx® ZCU102 board but can be adjusted for different devices, family architectures, and boards. Configuration memory soft errors are detected and corrected by:

- Using the processing system to allow the SEM controller to initialize and scan for errors in the programmable logic (PL).
- Using a minimal set of the Zynq MPSoC processors’ extended multiplexed I/O (EMIO) general purpose I/O (GPIO) pins to control the SEM controller clock, internal configuration access port (ICAP) arbitration interface, and board LEDs.
- Reading status signals from the SEM controller and performing multiboot.

Alternatively, Integrating LogiCORE SEM IP with AXI in Zynq UltraScale+ Devices (XAPP1303) [Ref 1] outlines how to use a Zynq UltraScale+ MPSoC in conjunction with an AXI4-Lite interface and the LogiCORE IP UltraScale+ architecture SEM controller. The AXI4-Lite interface is used to communicate with and manage the SEM controller.

Reference Design

The reference design was created using Vivado® Design Suite 2016.2 IP integrator.

Download the reference design files for this application note from the Xilinx website.

This reference design uses the SEM controller, monitor interface, and UART to report information to the user. To demonstrate multiboot, there are actually two designs in this reference design. The designs are referred to as Design 1 and Design 2. Diagrams of the designs are provided in Figure 1, Figure 2, and Figure 3. Both designs consist of the following Vivado IP integrator blocks:

- Zynq UltraScale+ MPSoC
- Utility buffer - Buffer configured as an BUFGCE
- UltraScale architecture Soft Error Mitigation controller
- SEM UART - UART module for connection to an external device such as Tera Term
• Constant - Used to drive a constant value on a signal or bus
• Concat - Used to concatenate signals and smaller buses of varying widths into a larger bus
• Slice - Used to rip bits off a bus

In both designs, the MPSoC EMIO GPIO interface connects to the chip enable of a BUFGCE, the ICAP arbitration interface, and LEDs. Both designs use the SEM UART to receive status information from the SEM controller and to send commands to the SEM controller. Refer to UltraScale Architecture Soft Error Mitigation Controller LogiCORE IP Product Guide (PG187) [Ref 2] for information about using the UART interface with SEM.

The primary difference between the two designs is the behavior of the LEDs which is described below. The LED behavior difference is used to demonstrate multiboot using Quad Serial Peripheral Interface (QSPI) flash memory. After the software application for Design 1 completes, the BIT file for Design 2 is configured in the device. After the software application for Design 2 completes, the BIT file for Design 1 is configured in the device. This looping between Design 1 and Design 2 will continue until the user chooses to stop.

In Design 1, the PS is used to enable the ICAP clock from the BUFGCE, then to turn on the gpio_led_1 and gpio_led_2 LEDs. Then the PS uses the EMIO GPIO interface of the Zynq UltraScale+ MPSoC block to allow the SEM controller to arbitrate for the ICAP. The SEM controller initializes, and then Design 1 uses the PS to poll the status_correction and status_uncorrectable signals. Detection of each type of error advances the software application. After an uncorrectable error is injected and detected, the PS is used to put the SEM controller in the Idle state, disable the clock, and boot using the boot image for Design 2. GPIO LEDs are used to demonstrate the design is functioning properly. These signals are connected to LEDs:

bufgce_enabled, cap_gnt, cap_rel, cap_req, gpio_led_1, gpio_led_2, status_correction, and status_uncorrectable

When a signal connected to an LED is asserted High, the associated LED illuminates.

In Design 2, the PS is used to enable the ICAP clock from the BUFGCE. The PS waits for you to turn on DIP switch 1, which causes GPIO LED[1] to blink. After turning off DIP switch 1 and turning on DIP switch 2, four LEDs blink in succession. After turning off DIP switch 2, the PS uses the EMIO GPIO interface of the Zynq UltraScale+ MPSoC block to allow the SEM controller to arbitrate for the ICAP. The SEM controller initializes. You are instructed to verify the SEM controller is operational, then the PS is used to put the SEM controller in the Idle state, disable the clock, and boot using the boot image for Design 1.

GPIO LEDs are used to demonstrate the design is functioning properly. These signals are connected to LEDs:

bufgce_enabled, cap_gnt, cap_rel, cap_req, gpio_led_1, gpio_led_2, gpio_led_3, and gpio_led_4

When a signal connected to an LED is asserted High, the associated LED illuminates.
IMPORTANT: This application and all supporting files are for demonstration purposes only. All files are delivered as is. This reference design is not intended to be a drop-in solution. After integrating this reference design into a new design, thorough verification is required to ensure the resulting solution meets functional and reliability goals.

Hardware

Figure 1 shows a high-level block diagram of the reference design. A PC is used to connect to the evaluation board JTAG port and USB to Serial port. More detailed Vivado IP integrator diagrams are shown in Figure 2 and Figure 3, which show each design’s connection to the LEDs and DIP switches.

Figure 1: Design 1 and Design 2 High-Level Block Diagram
Design 1 uses the signals listed in Table 1 to provide visual indicators by connecting them to the evaluation board GPIO LEDs.

**Table 1: Design 1 GPIO LED Connections**

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>GPIO LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufgce_enabled</td>
<td>GPIO LED[0]</td>
</tr>
<tr>
<td>gpio_led_1</td>
<td>GPIO LED[1]</td>
</tr>
<tr>
<td>gpio_led_2</td>
<td>GPIO LED[2]</td>
</tr>
<tr>
<td>status_correction</td>
<td>GPIO LED[3]</td>
</tr>
<tr>
<td>status_uncorrectable</td>
<td>GPIO LED[4]</td>
</tr>
<tr>
<td>cap_req</td>
<td>GPIO LED[5]</td>
</tr>
<tr>
<td>cap_gnt</td>
<td>GPIO LED[6]</td>
</tr>
<tr>
<td>cap_rel</td>
<td>GPIO LED[7]</td>
</tr>
</tbody>
</table>

Design 1 does not use any DIP switches.

Design 2 uses the signals listed in Table 2 to provide visual indicators by connecting them to the evaluation board GPIO LEDs.

**Table 2: Design 2 GPIO LED Connections**

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>GPIO LED</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufgce_enabled</td>
<td>GPIO LED[0]</td>
</tr>
<tr>
<td>gpio_led_1</td>
<td>GPIO LED[1]</td>
</tr>
<tr>
<td>gpio_led_2</td>
<td>GPIO LED[2]</td>
</tr>
<tr>
<td>gpio_led_3</td>
<td>GPIO LED[3]</td>
</tr>
<tr>
<td>gpio_led_4</td>
<td>GPIO LED[4]</td>
</tr>
<tr>
<td>cap_req</td>
<td>GPIO LED[5]</td>
</tr>
<tr>
<td>cap_gnt</td>
<td>GPIO LED[6]</td>
</tr>
<tr>
<td>cap_rel</td>
<td>GPIO LED[7]</td>
</tr>
</tbody>
</table>

Design 2 uses DIP switch 1 to blink GPIO LED 1 on and off. It also uses DIP switch 2 to flash GPIO LEDs 1 through 4 in sequence.

*Figure 2 and Figure 3* show the Vivado IP integrator diagrams for the designs.
Figure 2: Design 1 Vivado IP Integrator Diagram
Figure 3: Design 2 Vivado IP Integrator Diagram
Software Application

Two software applications are provided with this application note—one for each hardware design. Each software application contains comments to help understand the steps involved in creating the software application:

Design 1—MPSoC_SEM_ICAP_MBoot/design1/SDK/source/sem_icap_mboot_1.c
Design 2—MPSoC_SEM_ICAP_MBoot/design2/SDK/source/sem_icap_mboot_2.c

RECOMMENDED: Become familiar with the Zynq UltraScale+ MPSoC Technical Reference Manual (UG1085) [Ref 3] and Zynq UltraScale+ MPSoC Register Reference (UG1087) [Ref 4], which were used to create the applications.

Full logs for the applications are provided in the application note ZIP file.

Design 1 and Design 2

MPSoC_SEM_ICAP_MBoot/teraterm_sem_icap_mboot.log
MPSoC_SEM_ICAP_MBoot/teraterm_sem_uart.log

Tool Flow and Verification

Table 3 lists the tool flow and verification procedures used for the provided reference design.

Table 3: Reference Design Checklist

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>General</strong></td>
<td></td>
</tr>
<tr>
<td>Developer Name</td>
<td>Xilinx</td>
</tr>
<tr>
<td>Target Devices</td>
<td>xczu9eg-ffvb1156</td>
</tr>
<tr>
<td>Source code provided?</td>
<td>Yes</td>
</tr>
<tr>
<td>Source code format (if provided)</td>
<td>VHDL, C</td>
</tr>
</tbody>
</table>
| Design uses code or IP from existing reference design, application note, 3rd party or Vivado software? If yes, list. | UltraScale Soft Error Mitigation (Beta) 
SEM UART 
Zynq UltraScale+ MPSoC 
Utility Buffer 
Constant 
Concatenate 
Slice |
| **Simulation**                                                          |                                                                             |
| Functional simulation performed                                         | No                                                                          |
| Timing simulation performed?                                             | No                                                                          |
| Test bench provided for functional and timing simulation?                | No                                                                          |
Requirements

The design was developed on the following hardware and software. Refer to Limitations and Considerations for additional information regarding the SEM controller and targeting other software versions.

Hardware

- HW-Z1-ZCU102 Rev B or newer
- ZU9EG silicon
- JTAG programming cable
- USB cable for the JTAG connection
- USB cable for the UART connection

Software

- Vivado Design Suite 2016.2
- Software Development Kit (SDK) 2016.2
- Tera Term 4.83 for serial UART connection

Reference Design Files

The top-level reference design file, xapp1298-integrating-sem-ip.zip, contains two reference designs. The directory structure for each reference design is as follows.

Table 3: Reference Design Checklist (Cont’d)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test bench format</td>
<td>N/A</td>
</tr>
<tr>
<td>Simulator software and version</td>
<td>N/A</td>
</tr>
<tr>
<td>SPICE/IBIS simulations</td>
<td>No</td>
</tr>
</tbody>
</table>

Implementation

- Synthesis software tools/versions used: Vivado design suite 2016.2
- Implementation software tool(s) and version: Vivado design suite 2016.2
- Static timing analysis performed?: Yes

Hardware Verification

- Hardware verified?: Yes
- Platform used for verification: Zynq UltraScale+ MPSoC ZCU102 Evaluation Board
Directory Structure

MPSoC_SEM_ICAP_MBoot
|-- teraterm_sem_icap_mboot.log
    - Full log file from running the SW Application
|-- teraterm_sem_uart.log.log
    - Full log file from SEM controller
|-- readme.txt
    - This file
|-- design1
    |-- SDK
    |   |-- boot_img
    |   |   |-- BOOT.bin
    |   |   |-- fsbl_a53_0.elf
    |   |   |-- output.bif
    |   |   |-- sem_icap_mboot_a53_0.elf
    |   |   |-- zu9eg_sem_wrapper.bit
    |   |   `-- source
    |   |       `-- sem_icap_mboot_1.c
    |   `-- Vivado
    |       `-- hdl
    |           `-- constraints
    |               `-- zu9eg_sem.xdc
|-- design2
    |-- SDK
    |   |-- boot_img
    |   |   |-- BOOT.bin
    |   |   |-- fsbl_a53_0.elf
    |   |   |-- output.bif
    |   |   |-- sem_icap_mboot_a53_0.elf
    |   |   |-- zu9eg_sem_wrapper.bit
    |   |   `-- source
    |   |       `-- sem_icap_mboot_2.c
    |   `-- Vivado
    |       `-- hdl
    |           `-- constraints
    |               `-- zu9eg_sem.xdc
|-- ip
    |-- sem_ultra_v3_1
        `-- Packaged IP required for ZU9EG support
|-- sem_uart_src
    `-- project_1
        |-- sem_ultra_0_example
        |   `-- sem_ultra_0_example.srsc
        `-- imports
            `-- example_design
                `-- support
                    |-- sem_ultra_0_uart.v
                    |-- sem_ultra_0_uart_fifo.v
                    |-- sem_ultra_0_uart_piso.v
                    |-- sem_ultra_0_uart_sipo.v
|-- Vivado
    `-- tcl
        `-- board_repository.tcl
        `-- ip_repository.tcl
            `-- uart.tcl
design 1

This directory contains all of the SDK image files and source running the SW application and generating the SW application from scratch. The boot_img directory contains all of the SDK generated files for running the application. The design’s constraints are in the Vivado directory.

- SDK/boot_img
  - BOOT.bin - Boot image file that contains the ELF and BIT files
  - fsbl_a53_0.elf - Contains the First Stage Boot Loader software application data
  - output.bif - SDK generated file that references the files necessary to create the BOOT image
  - sem_icap_mboot_a53_0.elf - Contains the SEM ICAP and multiboot software application data
  - zu9eg_sem_wrapper.bit - PL configuration file
- SDK/source - Software application source code
- Vivado/hdl/constraints - Contains the constraints file for the Vivado design

design 2

This directory contains all of the SDK image files and source running the SW application and generating the SW application from scratch. The boot_img directory contains all of the SDK generated files for running the application. The design’s constraints are in the Vivado directory.

- SDK/boot_img
  - BOOT.bin - Boot image file that contains the ELF and BIT files
  - fsbl_a53_0.elf - Contains the first stage boot loader software application data
  - output.bif - SDK generated file that references the files necessary to create the BOOT image
  - sem_icap_mboot_a53_0.elf - Contains the SEM ICAP and multiboot software application data
  - zu9eg_sem_wrapper.bit - PL configuration file
- SDK/source - Software application source code
- Vivado/hdl/constraints - Contains the constraints file for the Vivado design

ip

This directory contains the SEM UltraScale+ family packaged design (sem_ultra_v3_1) that supports the ZU9EG device on the ZCU102 board.
sem_uart_src

The source code for the SEM UART is contained in this directory structure. This source was
generated by targeting an UltraScale+ device that supports SEM. Refer to Generate the SEM
UART Source Code for more details.

project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/

- sem_ultra_0_uart.v
- sem_ultra_0_uart_fifo.v
- sem_ultra_0_uart_piso.v
- sem_ultra_0_uart_sipo.v

Vivado

Three Tcl files are provided for reference. These files can be modified to accommodate a
different directory structure, then sourced in the Vivado Tcl console when creating the
reference design from scratch.

- tcl/board_repository.tcl - Contains the command used to set the path for the
  board files downloaded from the Zynq UltraScale+ ZCU102 HeadStart website
- tcl/ip_repository.tcl - Contains the commands to set the SEM controller repository
  from the build to a local repository
- tcl/uart.tcl - Contains the commands to add and import the SEM UART files to design

Licensing

**SEM Controller Licensing**

This Xilinx LogiCORE IP module is provided at no additional cost with the Xilinx Vivado Design
Suite under the terms of the Xilinx End User License.

Information about this and other Xilinx LogiCORE IP modules is available at the Xilinx
Intellectual Property website. For information on pricing and availability of other Xilinx
LogiCORE IP modules and tools, contact your local Xilinx sales representative.

---

Reference Design Steps

Setup

Setup instructions are included in the sections below. If you are creating the reference design
from scratch, refer to Creating the Reference Design. However, to run the reference design with
the supplied ZIP files, refer to Running the Reference Design, page 49. This section also
contains any other setup information required to run the reference design.
Follow the steps in the remaining sections to create the reference design.

**Creating the Reference Design**

*Download ZCU102 Board Files for Vivado Design Suite 2016.2.*

*Note:* The ZCU102 board files are not released with Vivado Design Suite 2016.2, and they are not provided in this application note. The board files must be downloaded from the [Zynq UltraScale+ ZCU102 HeadStart website](https://www.xilinx.com/products/design-tools/vivado/design-suites.html).

**IMPORTANT:** It could take up to two days to access the files after you register.

1. Go to the [Zynq UltraScale+ ZCU102 HeadStart website](https://www.xilinx.com/products/design-tools/vivado/design-suites.html).
2. Click the **Documentations and Designs** tab.
3. Scroll down to the **ZCU102 ES1 Board Files**.
4. Click the link for the ZCU102 board files: **ZCU102 2016.2 board files**
   
   *Note:* Use only 2016.2 board files.
5. If you agree to the license agreement, select **I Accept**.
6. Download the design file.

**Generate the SEM UART Source Code**

Generate the SEM UART source code, hardware design, and software application for the reference design.

Design 1 and Design 2 use the same UART files. It is not necessary to generate separate UART files for Design 1 and Design 2.

*Note:* `<path>` = C:/Projects in the steps that follow.

1. The UART source files can be generated from Vivado tools by targeting a device in the design’s architecture family that supports SEM. In this application note, any UltraScale+ device that supports SEM could be chosen (such as the xcku9p-ffve900).
2. Select **UltraScale Soft Error Mitigation** from the **IP Catalog**.
3. Set **Mode** to **Mitigation and Testing** and **Controller Clock Period (ps)** to **10000**.
4. In Generate Output Products, click **Generate**.
5. In the Project Manager Hierarchy tab, right-click **sem_ultra_0**, select **Open IP Example Design**, and choose the output directory. A second Vivado interface opens and the example design is generated.
6. Close both Vivado windows. The design does not need to be synthesized or implemented.
7. In this application note, the four UART files are located at
<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
sem.Ultra_0_example/sem.Ultra_0_example.srcs/sources_1/imports/
example_design/support/

    sem.Ultra_0_uart

    sem.Ultra_0_uart_fifo.v

    sem.Ultra_0_uart_piso.v

    sem.Ultra_0_uart_sipo.v

**Design 1 - Generate PL Design**

These instructions describe how to generate the PL bitstream for Design 1 of the reference design from scratch.

Unzip the xapp1298-integrating-sem-ip.zip file. See the readme.txt file in the MPSoC_SEM_ICAP_MBoot folder.

**Create Project**

*Note:* <path> = C:/Projects in the steps that follow.

1. Change directory to MPSoC_SEM_ICAP_Mboot.


3. Before creating a project, in the Tcl console set the path to the ZCU102 board repository files. For example:

    set_param board.repoPaths C:/Projects/zcu102

4. Create a new project and configure it as follows:
   a. Project name: **zu9eg_sem**
   b. Project location: `<path>/MPSoC_SEM_ICAP_MBoot/design1`
   c. Select **RTL Project** and check **Do not specify sources at this time.**
d. Select **Boards**, set **Xilinx** as the vendor, and select the **Zynq UltraScale+ ZCU102 Evaluation Board**. See Figure 4.

![Image of board selection in Vivado](image)

**Figure 4:** Design 1 - Board Selection

e. Click **Next**, then **Finish**.

5. Set the IP repository path for the ZU3EG SEM controller included in the application note ZIP file.

   a. `set_property ip_repo_paths <path>/MPSoC_SEM_ICAP_MBoot/ip/sem Ultra_v3.1 [current_project]`

   For example, in the Vivado Tcl console enter:

   ```tcl
   set_property ip_repo_paths C:/Projects/MPSoC_SEM_ICAP_MBoot/ip/sem Ultra_v3.1 [current_project]
   ```

   b. `update_ip_catalog -rebuild`

   Ignore this duplicate IP warning:

   ```text
   WARNING: [IP_Flow 19-1663] Duplicate IP found for xilinx.com:ip:sem Ultra:3.1. The one found in IP location 'c:/Projects/MPSoC_SEM_ICAP_MBoot/ip/sem Ultra_v3.1' will take precedence over the same IP in the Xilinx installed IP.
   ```
Add Files

1. Add the constraints file via the Vivado Tcl console:
   
a. add_files -fileset constrs_1 -norecurse
   
   <path>/MPSoC_SEM_ICAP_MBoot/design1/Vivado/hdl/constraints/
   zu9eg_sem.xdc

   b. import_files -fileset constrs_1 -force
   
   <path>/MPSoC_SEM_ICAP_MBoot/design1/Vivado/hdl/constraints/
   zu9eg_sem.xdc

2. Add four SEM UART files to the project:
   
a. add_files -norecurse -scan_for_includes
   
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
   sem Ultra_0_example/sem Ultra_0_example.srcs/sources_1/imports/
   example_design/support/sem Ultra_0 uart.v}

   b. add_files -norecurse -scan_for_includes
   
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
   sem Ultra_0_example/sem Ultra_0_example.srcs/sources_1/imports/
   example_design/support/sem Ultra_0 uart fifo.v}

   c. add_files -norecurse -scan_for_includes
   
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
   sem Ultra_0_example/sem Ultra_0_example.srcs/sources_1/imports/
   example_design/support/sem Ultra_0 uart piso.v}

   d. add_files -norecurse -scan_for_includes
   
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
   sem Ultra_0_example/sem Ultra_0_example.srcs/sources_1/imports/
   example_design/support/sem Ultra_0 uart sipo.v}

   For example:

   add_files -norecurse -scan_for_includes
   
   (C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem Ultra_0_example/
   sem Ultra_0_example.srcs/sources_1/imports/example_design/support/sem Ultra_0 uart.v
   C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem Ultra_0_example/
   sem Ultra_0_example.srcs/sources_1/imports/example_design/support/sem Ultra_0 uart fifo.v
   C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem Ultra_0_example/
   sem Ultra_0_example.srcs/sources_1/imports/example_design/support/sem Ultra_0 uart piso.v
   C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem Ultra_0_example/
   sem Ultra_0_example.srcs/sources_1/imports/example_design/support/sem Ultra_0 uart sipo.v)

3. Import four SEM UART files to the project:
   
a. import_files -force -norecurse
   
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
   sem Ultra_0_example/sem Ultra_0_example.srcs/sources_1/imports/
   example_design/support/sem Ultra_0 uart.v}
b. import_files -force -norecurse
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
    sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/
    example_design/support/sem_ultra_0_uart_fifo.v}

c. import_files -force -norecurse
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
    sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/
    example_design/support/sem_ultra_0_uart_piso.v}

d. import_files -force -norecurse
   {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
    sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/
    example_design/support/sem_ultra_0_uart_sipo.v}

For example:

    import_files -force -norecurse
    {C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/
     sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0 UART.v
     C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/
     sem_ultra_0_example.srcs/sources_1/imports/example_design/support/
     sem_ultra_0_uart_fifo.v
     C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/
     sem_ultra_0_example.srcs/sources_1/imports/example_design/support/
     sem_ultra_0_uart_piso.v
     C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/
     sem_ultra_0_example.srcs/sources_1/imports/example_design/support/
     sem_ultra_0_uart_sipo.v}

4. Select Create Block Design and set the Design name to zu9eg_sem.

Add IP to the Diagram

1. In the Vivado Diagram tab, right-click Add IP and select ZYNQ UltraScale+ MPSoC.
   a. Select Run Block Automation. See Figure 5.

   ![Diagram with Run Block Automation selected](X18136-02217)

   Figure 5: Design 1 - Run Block Automation

   b. Verify Apply Board Presets is selected.
c. Configure the Zynq UltraScale+ MPSoC block to enable EMIO GPIO. Deselect UART 1. Click OK. See Figure 6.

2. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Utility Buffer in the search box, and select it.
   b. Double-click the utility buffer and select the BUFGCE type. Click OK.
3. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type UltraScale Soft Error Mitigation in the search box, and select it.
   b. In the Duplicate IP window, click Add Active IP.
   c. Double-click the icon. Select Mitigation and Testing, and make sure the Controller Clock Period is set to 10000ps (100MHz). Click OK.
   d. In the diagram, select the cap_req pin, right-click, and select Make External. Repeat this for the status_correction and status_uncorrectable pins.
4. Right-click on an empty area of the **Diagram** tab and select **Add Module**.
   a. Type *sem_ultra_0_uart.v* in the search box, and select it.
   b. Right-click the *uart_tx* and *uart_rx* pins and select **Make External**.

5. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **1-bit wide** and to output **0**. (Connects to UltraScale Soft Error Mitigation command_strobe and aux_error_cr_ne, aux_error_cr_es, and aux_error_uc signals.)

6. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **44-bits wide** and to output **0**. (Connects to UltraScale Soft Error Mitigation command_code signal.)

7. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **3-bits wide** and to output **0**. (Connects to In0[2:0] Concat.)

8. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **89-bits wide** and to output **0**. (Connects to In4[88:0] Concat.)

9. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Slice** in the search box, and select it.
   b. Configure as follows (Connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **0**
      - Din Down To: **0**
      - Click **OK**.
c. Right-click on Dout[0:0] pin, select Make External. A port named Dout is created.
d. Change the name of Dout. Select the Dout port in the Diagram tab. In the External Port Properties General Tab, change Dout to bufgce_enabled. See Figure 7.

10. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Slice in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: 95
      - Din From: 1
      - Din Down To: 1
c. Right-click on Dout[0:0] pin, select Make External. Change the name to gpio_led_1. Use the same process in step 9 step d to change the pin name.

11. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Slice in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: 95
- Din From: 2
- Din Down To: 2

c. Right-click on Dout[0:0] pin, select Make External. Change the name to gpio_led_2. Use the same process in step 9 step d to change the pin name.

12. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Slice in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: 95
      - Din From: 6
      - Din Down To: 6
   c. Right-click on Dout[0:0] pin, and select Make External. Change the name to cap_gnt. Use the same process in step 9 step d to change the pin name.

13. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Slice in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: 95
      - Din From: 7
      - Din Down To: 7
   c. Right-click on Dout[0:0] pin, and select Make External. Change the name to cap_rel. Use the same process in step 9 step d to change the pin name.

14. Right-click on an empty area of the Diagram tab and select Add IP.
   a. Type Concat in the search box, and select it.
      - Configure it to have 5 ports. See Figure 8.
      - Change Auto to Manual on all Ports.
      - Set In0 Width to 3, In1 Width to 1, In2 Width to 1, In3 Width to 1, and In4 Width to 89.
      - Click OK.
b. See Figure 2 to make the remaining connections to concat IP:

- Connect In0 to Const dout[2:0].
- Connect In1 to UltraScale Soft Error Mitigation status_correction.
- Connect In2 to UltraScale Soft Error Mitigation status_uncorrectable.
- Connect In3 to UltraScale Soft Error Mitigation cap_req.
- Connect In4 to Const dout[88:0].
- Connect dout[94:0] to ZYNQ UltraScale+ MPSoC emio_gpio_i[94:0].
Make Final Diagram Connections

1. Connect all signals as shown in Figure 2 (Design 1 Vivado IP Integrator Diagram).
2. Right-click zu9eg_sem.bd and select Create HDL Wrapper.
   a. Select Let Vivado manage wrapper and auto-update in the pop-up window.
3. Right-click zu9eg_sem_wrapper.v and select Set As Top.
4. Right-click in an open area of the Block Design and select Validate Design.

Generate Block Design and Bitstream

1. Select Generate Block Design in the Flow Navigator.
   a. Select Generate. See Figure 9.
2. In the Flow Navigator, select Generate Bitstream, open the implemented design, and verify all timing was met.
3. To export the bitstream, select File > Export > Export Hardware. Check Include bitstream.
4. Select File > Launch SDK.

This launches the SDK where the Software application can be created. Proceed to Design 1 - Generate the PS Software Application.
Design 1 - Generate the PS Software Application

These instructions describe how to generate the PS software application for Design 1 of the reference design from scratch.

1. Create the FSBL Application Project by selecting File > New > Application Project.
   a. Enter fsbl_a53_0 as the Project name. Click Next.
   b. Select Zynq MP FSBL. Click Finish. See Figure 10.

Figure 10: Design 1 - Zynq MP FSBL Template
2. Create the Application Project by selecting **File > New > Application Project**.
   a. Enter the Project name **sem_icap_mboot_a53_0**. Click **Next**.
   b. Select **Hello World** and click **Finish**.
   c. Delete **helloworld.c** by right-clicking it, then select **Delete** in the project explorer. See **Figure 11**.
   
   ![Figure 11: Design 1 - Delete helloworld.c](image)

   d. Copy the **sem_icap_mboot_1.c** file from
      
      `<path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/source/sem_icap_mboot_1.c`
      
      to
      
      `<path>/MPSoC_SEM_ICAP_MBoot/design1/zu9eg_sem/zu9eg_sem.sdk/
      sem_icap_mboot_a53_0/src/`
Right-click the *src* directory. Select **Refresh** so the copied file appears. See **Figure 12**.

**Design 1 - Create Boot Image**

1. Copy the FSBL executable and linkable format (ELF) file `fsbl_a53_0.elf` from `<path>/MPSoC_SEM_ICAP_MBoot/design1/zu9eg_sem/zu9eg_sem.sdk/fsbl_a53_0/Debug/` to the `boot_img` directory `<path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/`.

2. Copy the BIT file to the `boot_img` directory

   from `<path>/MPSoC_SEM_ICAP_MBoot/design1/zu9eg_sem/zu9eg_sem.sdk/zu9eg_sem_wrapper_hw_platform_0/zu9eg_sem_wrapper.bit`

   to `<path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/zu9eg_sem_wrapper.bit`

3. Copy the `sem_icap_mboot` ELF file to the `boot_img` directory

   from `<path>/MPSoC_SEM_ICAP_MBoot/design1/zu9eg_sem/zu9eg_sem.sdk/sem_icap_mboot_a53_0/Debug/sem_icap_mboot_a53_0.elf`

   to `<path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/sem_icap_mboot_a53_0.elf`
4. Select **Xilinx Tools > Create Boot Image**. Three partitions are added in the following order:
   a. `fsbl_a53_0.elf`
   b. `zu9eg_sem_wrapper.bit`
   c. `sem_icap_mboot_a53_0.elf`

5. Change Architecture to **Zynq MP**.

6. In the **Create Boot Image** window, set Output BIF file path to
   <path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/output.bif
   a. Verify the Output path is the same as the Output BIF file path. See *Figure 13*.

7. Add the partitions by selecting **Add**.
   a. Add the `fsbl_a53_0.elf` partition by selecting **Add** in the **Create Boot Image** window.
   b. In the **Add Partition** window, set File path to
      <path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/fsbl_a53_0.elf. Select **OK**.

*Figure 13:  Design 1 - Create Boot Image*
c. Verify the Partition Type is **bootloader**, the Destination Device is PS, and the Destination CPU is **A53 x64**. See **Figure 14**.

![Add new boot image partition window](image)

**Figure 14:** Design 1 - FSBL Boot Partition

8. Add the BIT file to the second partition by selecting **Add** in the **Create Boot Image** window.

a. In the **Add Partition** window, set File path to 
   <path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/zu9eg_sem_wrapper.bit. Select **OK**.
b. Verify the Partition type is **datafile** and the Destination Device is PL. See Figure 15.

![Add new boot image partition window](image)

**Figure 15**: Design 1 - BIT Partition

9. Add the `sem_icap_mboot_a53_0.elf` file to the third partition by selecting **Add** in the **Create Boot Image** window.

   a. In the Add Partition window, set File path to 
   
   `<path>/MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/sem_icap_mboot_a53_0.elf`. Select **OK**.
b. Verify the Partition type is **datafile**, the Destination Device is **PS**, and the Destination CPU is **A53 0**. See **Figure 16**.

![Figure 16: Design 1 - SEM ICAP MultiBoot Partition](image-url)
10. Verify the Boot image partitions paths. See Figure 17. Click Create Image.

![Design 1 - Create Boot Image](X18168-022217)

**Design 2 - Generate PL Design**

These sections contain instructions to generate the PL bitstream for Design 2 of the reference design from scratch.

**Create Project**

These instructions describe how to generate this portion of the reference design from scratch.

*Note:* `<path> = C:/Projects` in the following steps.

1. Change directory to `MPSoCSEM_ICAP_Mboot`.
2. Open the Vivado Design Suite.
3. Before creating a project, in the Tcl console set the path to the zcu102 board repository files:
   a. Set `set_param board.repoPaths` to the path to zcu102 board files. For example:
      ```bash
      set_param board.repoPaths C:/Projects/zcu102
      ```
4. Create a new project and configure it as follows:
   a. Project name: \texttt{zu9eg\_sem}
   b. Project location: \texttt{<path>/MPSoC\_SEM\_ICAP\_MBoot/design2}.
   c. Select \texttt{RTL Project} and \texttt{Do not specify sources at this time}.
   d. Select \texttt{Boards}, set \texttt{Xilinx} as the vendor, and select the \texttt{Zynq UltraScale+ ZCU102 Evaluation Board}. See Figure 18.

   ![New Project](image)

   \textbf{Figure 18:} Design 2 - Board Selection

   e. Click \texttt{Next}, then \texttt{Finish}.

5. Set the IP repository path for the ZU3EG SEM controller included in the application note ZIP file.
   a. \texttt{set\_property ip\_repo\_paths}
      \texttt{<path>/MPSoC\_SEM\_ICAP\_MBoot/ip/sem\_ultra\_v3\_1} [current\_project]

      For example, in the Vivado Tcl console enter:

      \begin{verbatim}
      set_property ip_repo_paths C:/Projects/MPSoC_SEM_ICAP_MBoot/ip/sem_ultra_v3_1 [current_project]
      \end{verbatim}

   b. \texttt{update\_ip\_catalog -rebuild}

      Ignore this duplicate IP warning:
WARNING: [IP_Flow 19-1663] Duplicate IP found for xilinx.com:ip:sem_ultra:3.1. The one found in IP location c:/Projects/MPSoC_SEM_ICAP_MBoot/ip/sem_ultra_v3_1 will take precedence over the same IP in the Xilinx installed IP.

Add Files

1. Add the constraints file via the Vivado Tcl console:
   a. add_files -fileset constrs_1 -norecurse
      <path>/MPSoC_SEM_ICAP_MBoot/design2/Vivado/hdl/constraints/zu9eg_sem.xdc
   b. import_files -fileset constrs_1 -force
      <path>/MPSoC_SEM_ICAP_MBoot/design2/Vivado/hdl/constraints/zu9eg_sem.xdc

2. Add four SEM UART files to the project:
   a. add_files -norecurse -scan_for_includes
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart.v}
   b. add_files -norecurse -scan_for_includes
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_fifo.v}
   c. add_files -norecurse -scan_for_includes
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_piso.v}
   d. add_files -norecurse -scan_for_includes
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_sipo.v}

For example:

add_files -norecurse -scan_for_includes
{C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart.v}
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_fifo.v
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_piso.v
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem_ultra_0_example/sem_ultra_0_example.srcs/sources_1/imports/example_design/support/sem_ultra_0_uart_sipo.v}
3. Import four SEM UART files to the project:
   a. `import_files -force -norecurse
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
       sem.Ultra_0_example/sem.Ultra_0_example.srcs/sources_1/imports/
       example_design/support/sem.Ultra_0_uart.v}
   b. `import_files -force -norecurse
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
       sem.Ultra_0_example/sem.Ultra_0_example.srcs/sources_1/imports/
       example_design/support/sem.Ultra_0_uart_fifo.v }
   c. `import_files -force -norecurse
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
       sem.Ultra_0_example/sem.Ultra_0_example.srcs/sources_1/imports/
       example_design/support/sem.Ultra_0_uart_piso.v }
   d. `import_files -force -norecurse
      {<path>/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/
       sem.Ultra_0_example/sem.Ultra_0_example.srcs/sources_1/imports/
       example_design/support/sem.Ultra_0_uart_sipo.v}

For example:

```
import_files -force -norecurse
{C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem.Ultra_0_example/
  sem.Ultra_0_example.srcs/sources_1/imports/example_design/support/sem.Ultra_0_uart.v
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem.Ultra_0_example/
  sem.Ultra_0_example.srcs/sources_1/imports/example_design/support/sem.Ultra_0_uart_fifo.v
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem.Ultra_0_example/
  sem.Ultra_0_example.srcs/sources_1/imports/example_design/support/sem.Ultra_0_uart_piso.v
C:/Projects/MPSoC_SEM_ICAP_MBoot/sem_uart_src/project_1/sem.Ultra_0_example/
  sem.Ultra_0_example.srcs/sources_1/imports/example_design/support/sem.Ultra_0_uart_sipo.v}
```
4. Select **Create Block Design** and set the Design name to **zu9eg_sem**.

**Add IP to the Diagram**

1. In the Vivado **Diagram** tab, right-click **Add IP** and select **ZYNQ UltraScale+ MPSoC**.
   a. Select **Run Block Automation**. See Figure 19.

   ![Figure 19: Design 2 - Run Block Automation](image1)

   b. Verify **Apply Board Presets** is selected.
   c. Configure the Zynq UltraScale+ MPSoC block to enable EMIO GPIO. Deselect **UART 1**. Click **OK**. See Figure 20.

   ![Figure 20: Design 2 - Enable EMIO GPIO](image2)
2. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Utility Buffer** in the search box, and select it.
   b. Double click the utility buffer and select the **BUFGCE** type. Click **OK**.
3. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **UltraScale Soft Error Mitigation** in the search box, and select it.
   b. In the **Duplicate IP** window, click **Add Active IP**.
   c. Double-click the icon. Select **Mitigation and Testing**, and make sure the Controller Clock Period is set to **10000ps (100MHz)**. Click **OK**.
   d. In the diagram, select the **cap_req** pin. Right-click and select **Make External**.
4. Right-click on an empty area of the **Diagram** tab and select **Add Module**.
   a. Type **sem_ultra_0_uart.v** in the search box, and select it.
   b. Right-click the **uart_tx** and **uart_rx** pins and select **Make External**.
5. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **1-bit wide** and to output **0** (connects to UltraScale Soft Error Mitigation **command_strobe** and **aux_error_cr_ne**, **aux_error_cr_es**, and **aux_error_uc** signals).
6. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **44-bits wide** and to output **0**. (Connects to UltraScale Soft Error Mitigation **command_code** signal.)
7. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **5-bits wide** and to output **0**. (Connects to **In0[4:0]** Concat.)
8. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **2-bits wide** and to output **0**. (Connects to **In2[1:0]** Concat.)
9. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Constant** in the search box, and select it.
   b. Configure it to be **85-bits wide** and to output **0**. (Connects to **In5[84:0]** Concat.)
10. Right-click on an empty area of the **Diagram** tab and select **Add IP**.

   a. Type **Slice** in the search box, and select it.

   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      
      - **Din Width**: 95
      - **Din From**: 0
      - **Din Down To**: 0
      - Click **OK**.

   c. Right-click on **Dout[0:0]** pin and select **Make External**. A port named Dout is created.

   d. Change the name of Dout. Select the **Dout** port in the Diagram tab. In the External Port Properties General Tab, change Dout to **bufgce_enabled**. See **Figure 21**.

---

**Figure 21:**  Design 2 - Configure External Port - **bufgce_enabled**
11. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Slice** in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **1**
      - Din Down To: **1**
   c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **gpio_led_1**. Use the same process in **step 10 step d** to change the pin name.

12. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Slice** in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **2**
      - Din Down To: **2**
   c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **gpio_led_2**. Use the same process in **step 10 step d** to change the pin name.

13. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Slice** in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **3**
      - Din Down To: **3**
   c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **gpio_led_3**. Use the same process in **step 10 step d** to change the pin name.

14. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   a. Type **Slice** in the search box, and select it.
   b. Configure as follows (connects to emio_gpio_o[94:0] on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **4**
      - Din Down To: **4**
c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **gpio_led_4**. Use the same process in step 10 step d to change the pin name.

15. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   
a. Type **Slice** in the search box, and select it.
   
b. Configure as follows (connects to `emio_gpio_o[94:0]` on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **6**
      - Din Down To: **6**
   
c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **cap_gnt**. Use the same process in step 10 step d to change the pin name.

16. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   
a. Type **Slice** in the search box, and select it.
   
b. Configure as follows (connects to `emio_gpio_o[94:0]` on the Zynq UltraScale+ MPSoC block):
      - Din Width: **95**
      - Din From: **7**
      - Din Down To: **7**
   
c. Right-click on **Dout[0:0]** pin and select **Make External**. Change the name to **cap_rel**. Use the same process in step 10 step d to change the pin name.

17. Right-click on an empty area of the **Diagram** tab and select **Add IP**.
   
a. Type **Concat** in the search box, and select it.
   
b. Configure it to have **6** ports. Change Auto to **Manual** on all Ports. Set In0 Width to **5**, In1 Width to **1**, In2 Width to **2**, In3 Width to **1**, In4 Width to **1**, and In5 Width to **85**. Click **OK**. See **Figure 22**.
c. See Figure 3 to make the remaining connections to Concat IP:

- Connect In0 to Const dout[4:0].
- Connect In1 to external pin cap_req.
- Connect In2 to Const dout[1:0].
- Connect In3 to SW13_DIP1. Use the same process in step 10 step d to right-click In3[0:0] pin, select Make External, then change name to SW13_DIP1.
- Connect In4 to SW13_DIP2. Use the same process in step 10 step d to right-click In4[0:0] pin, select Make External, then change name to SW13_DIP2.
- Connect In5 to Const dout[84:0].
- Connect Concat dout[94:0] to Zynq UltraScale+ MPSoC emio_gpio_i[94:0].

**Figure 22:** Design 2 - Configure Concat IP - Width 95
Make Final Diagram Connections

1. Connect all signals as shown in Figure 3, Design 2 Vivado IP Integrator Diagram.
2. Right-click zu9eg_sem.bd and select Create HDL Wrapper.
   a. Select Let Vivado manage wrapper and auto-update in the pop-up window.
3. Right-click zu9eg_sem_wrapper.v and select Set As Top.
4. Right-click in an open area of Block Design and select Validate Design.

Generate Block Design and Bitstream

1. Select Generate Block Design in the Flow Navigator. See Figure 23.
   a. Select Generate. See Figure 23.
2. In the Flow Navigator, select Generate Bitstream, open the implemented design, and verify all timing was met.
3. To export the bitstream, select File > Export > Export Hardware. Check Include bitstream.
4. Select File > Launch SDK.

This launches the SDK where the Software application can be created. Proceed to Design 2 - Generate the PS Software Application.
Design 2 - Generate the PS Software Application

These instructions describe how to generate the PS software application for Design 2 of the reference design from scratch.

1. Create the FSBL Application Project by selecting File > New > Application Project.
   a. Enter fsbl_a53_0 as the Project name. Click Next.
   b. Select Zynq MP FSBL. Click Finish. See Figure 24.

![Figure 24: Design 2 - Zynq MP FSBL Template](image-url)
2. Create the Application Project by selecting **File > New > Application Project**.
   a. Enter the Project name `sem_icap_mboot_a53_0`. Click **Next**.
   b. Select **Hello World** and click **Finish**.
   c. Delete `helloworld.c` by right-clicking it, then select **Delete** in the project explorer. See **Figure 25**.

![Figure 25: Design 2 - Delete helloworld.c](image)

   d. Copy the `sem_icap_mboot_2.c` file from
   `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/source/sem_icap_mboot_2.c`
   to `<path>/MPSoC_SEM_ICAP_MBoot/design2/zu9eg_sem/zu9eg_sem.sdk/
   sem_icap_mboot_a53_0/src/`
Reference Design Steps

e. Right-click the **src** directory. Select **Refresh** so the copied file appears. See Figure 26.

![Design 2 - Project Explorer](image)

**Design 2 - Create Boot Image**

1. Copy the FSBL executable and linkable format (ELF) file `fsbl_a53_0.elf` to the `boot_img` directory.

   Copy from
   `<path>/MPSoC_SEM_ICAP_MBoot/design2/zu9eg_sem/zu9eg_sem.sdk/fsbl_a53_0/Debug/`

   to the `boot_img` directory
   `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/`

2. Copy the BIT file to the `boot_img` directory.

   Copy from
   `<path>/MPSoC_SEM_ICAP_MBoot/design2/zu9eg_sem/zu9eg_sem.sdk/zu9eg_sem_wrapper_hw_platform_0/zu9eg_sem_wrapper.bit`

   to `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/zu9eg_sem_wrapper.bit>`
3. Copy the `sem_icap_mboot` ELF file to the `boot_img` directory.

   Copy from
   
   `<path>/MPSoC_SEM_ICAP_MBoot/design2/zu9eg_sem/zu9eg_sem.sdk/
    sem_icap_mboot_a53_0/Debug/sem_icap_mboot_a53_0.elf`

   to `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/
     sem_icap_mboot_a53_0.elf`

4. Select **Xilinx Tools > Create Boot Image**. Three partitions are added in the following order:
   a. `fsbl_a53_0.elf`
   b. `zu9eg_sem_wrapper.bit`
   c. `sem_icap_mboot_a53_0.elf`

5. Change Architecture to **Zynq MP**.
6. In the **Create Boot Image** window, set Output BIF file path to 
<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/output.bif 

a. Verify the Output path is the same as the Output BIF file path. See **Figure 27**.

![Figure 27: Design 2 - Create Boot Image](https://www.xilinx.com)
7. Add the partitions by selecting **Add**.
   a. Add the `fsbl_a53_0.elf` partition by selecting **Add** in the **Create Boot Image** window.
   b. In the **Add Partition** window, set File path to `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/fsbl_a53_0.elf`. Select **OK**.
   c. Verify the Partition Type is **bootloader**, the Destination Device is **PS**, and the Destination CPU is **A53 x64**. See **Figure 28**.

![Add new boot image partition](image)

**Figure 28:** Design 2 - FSBL Boot Partition
8. Add the BIT file to the second partition by selecting **Add** in the **Create Boot Image** window.

   a. In the **Add Partition** window, set File path to 
      `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/ 
      zu9eg_sem_wrapper.bit`. Select **OK**.

   b. Verify the Partition type is **datafile** and the Destination Device is **PL**. See **Figure 29**.

*Figure 29: Design 2 - BIT Partition*
9. Add the `sem_icap_mboot_a53_0.elf` file to the third partition by selecting **Add** in the **Create Boot Image** window.

   a. In the **Add Partition** window, set **File path** to
      
      `<path>/MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/sem_icap_mboot_a53_0.elf`. **Select OK**.

   b. Verify the Partition type is **datafile**, the Destination Device is **PS**, and the Destination CPU is **A53 0**. See **Figure 30**.

![Add new boot image partition](image)

**Figure 30:** Design 2 - SEM ICAP MultiBoot Partition
10. Verify the **Boot image partitions** paths. See Figure 31. Click **Create Image**.

![Figure 31: Design 2 - Create Boot Image](image)

The reference design can be run with the created design files. Proceed to **Running the Reference Design** step 2. Set up the board.

**Running the Reference Design**

This section provides a guide for downloading the image files through JTAG to the ZCU102 board. It also shows how to use the PS to initialize the SEM controller and reads status from the SEM controller via the PS EMIO GPIO interface. The SEM UART terminal is used to send commands to the SEM controller.

1. **Unzip the reference design**: Unzip the `xapp1298-integrating-sem-ip.zip`. See the readme.txt file in the `MPSoC_SEM_ICAP_MBoot` folder.
2. **Set up the board**: Connect the power, USB UART cable, and JTAG programming cable to the board. Set MODE[3:0] pins to ON ON ON ON (SW6). Note that for this DIP switch, in relation to the arrow, moving the switch toward the label ON is a 0. DIP switch labels 1 through 4 are equivalent to Mode pins 0 through 3. See Figure 32 and Figure 33. Apply power.

**IMPORTANT:** Ensure the ZCU102 board is powered on, the USB UART cable is connected, the JTAG cable is connected, and the JTAG cable drivers have been installed. See step 3 below.
3. Set up the UART driver and select the COM port:
   
a. Open two Tera Term terminals and set up a serial connection with these settings:
   
   - **Baud**: 115200
   - **Settings**: Data 8; Parity None; Stop 1
   - **Flow Control**: None
   - **Terminal ID**: VT100
   - **New-line Transmit**: CR (terminal transmits CR as end of line)
   - **New-line Receive**: CR + LF (terminal receives CR as end of line and expands to CR + LF)
   - **Local Echo**: No

   **RECOMMENDED**: The Silicon Labs USB UART drivers might need to be installed to get this connection to work for the first time. For more information, see the Silicon Labs CP210x USB-to-UART Installation Guide (UG1033) [Ref 3].

   b. In Tera Term, select the **COM XX** port associated with the **Silicon Labs Quad CP210x USB to UART Bridge: Interface 0 (COMXX)** for the PS UART and **Silicon Labs Quad CP210x USB to UART Bridge: Interface 2 (COMXX)** for the SEM UART in the **Device Manager**, where **COM XX** depends on your computer. See **Figure 34**.

   ![Device Manager](image)

   **Figure 34**: COM Port Selection
Program the Flash Memory

1. Open SDK.
2. Xilinx Tools > Program Flash (for Design 1). Settings follow:
   - Image File: MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/BOOT.bin
   - Offset: 0x00000000
   - Flash Type: qspi_quad_parallel
   - FSBL File: MPSoC_SEM_ICAP_MBoot/design1/SDK/boot_img/fsbl_a53_0.elf

3. Xilinx Tools > Program Flash (for Design 2). Settings follow:
   - Image File: MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/BOOT.bin
   - Offset: 0x02000000
   - Flash Type: qspi_quad_parallel
   - FSBL File: MPSoC_SEM_ICAP_MBoot/design2/SDK/boot_img/fsbl_a53_0.elf

4. After programming finishes, turn off power to the board.
5. Set MODE[3:0] pins (SW6) to ON ON OFF ON. Note that for this DIP switch, in relation to the arrow, moving the switch toward the label ON is a 0. DIP switch labels 1 through 4 are equivalent to Mode pins 0 through 3. See Figure 35.

6. Turn on power to the board.
7. The application starts. Follow the directions in the PS UART terminal and verify SEM controller is operational in the SEM UART terminal. Refer to Results for Design 1 Application and Results for Design 2 Application for more details on the software application.

Results

All UART logs are provided under the main directory of the application note ZIP file:

- teraterm_sem_icap_mboot.log - Log file that contains the UART output from the PS.
- teraterm_sem_uart.log - Log file that contains the UART output from the SEM controller.
Results for Design 1 Application

This application initializes the PS UART, controls the transfer of control of the PL configuration logic from PCAP to ICAP (ICAP is required by SEM), and uses the PS EMIO GPIO pins. The application requires your input. A snippet of the output from PS UART of this application is provided alongside of the SEM UART log in Figure 36.

The PS disables PCAP and enables ICAP. Then the BUFGCE is enabled which activates the clock to the SEM logic. This also turns on GPIO LED 0. After GPIO LED 1 and GPIO LED 2 on the evaluation board are turned on by the PS, the PS checks that the SEM controller has asserted ICAP Request. If it is asserted, the PS gives ICAP control to the SEM controller by asserting ICAP Grant. At this point, the SEM controller initializes and automatically transitions to the Observation state. You are instructed to inject a correctable error. This can be performed by entering the following case-sensitive commands in the SEM UART terminal:

```
I
N C000A098000
O
```

This is shown in Figure 36 by the top red arrow. After transitioning to the Observation state, the SEM controller detects and corrects the error. GPIO LED 3 flashes on very quickly. You are instructed to inject an uncorrectable error. This can be performed by entering the following case-sensitive commands in the SEM UART terminal:

```
I
N C000A098000
N C000A098004
O
```

This is shown in Figure 36 by the middle red arrow. After transitioning to the Observation state, the SEM controller detects the error, but is unable to correct it. GPIO LED 4 turns on and remains on until the device is reconfigured. To reconfigure the device, the PS needs to have control of the CAP interface via PCAP, so SEM must relinquish control of the ICAP interface.

The PS asserts ICAP Release (GPIO LED [7]) and waits for the SEM controller to deassert ICAP Request (GPIO LED [5]). After SEM deasserts ICAP Request, it automatically transitions to the Idle state. The PS disables the BUFGCE clock (GPIO LED [0]). The SEM controller is now ready for reconfiguration to occur.
The PS enables the PCAP and performs multiboot. This is represented by the bottom red arrow in Figure 36. Refer to the application (sem_icap_mboot_1.c) for more details.

Figure 36: Design 1 PS and SEM UART Output

Results for Design 2 Application

This application initializes the PS UART, controls the transfer of control of the PL configuration logic from PCAP to ICAP (ICAP is required by SEM), and uses the PS EMIO GPIO pins and user input from the DIP switches on the evaluation board. A snippet of the output from PS UART of this application is provided alongside of the SEM UART log in Figure 37.

The PS disables PCAP and enables ICAP. Then the BUFGCE is enabled which activates the clock to the SEM logic. This also turns on GPIO LED 0. After you turn on DIP switch 1, GPIO LED 1 blinks. Then you must turn off DIP switch 1, and turn on DIP switch 2. This causes GPIO LEDs 1
to 4 to flash on and off in succession of each other. You must turn off DIP switch 2. Then the PS checks that the SEM controller has asserted ICAP Request. If it is asserted, the PS gives ICAP control to the SEM controller by asserting ICAP Grant. At this point, the SEM controller initializes and automatically transitions to the Observation state.

You are instructed to perform a short manual test to verify the SEM controller is operating correctly. This is represented by the red arrow in Figure 37. After this, the PS performs multiboot again and the PL is configured with the boot image from Design 1. However, to reconfigure the device, the PS needs to have control of the CAP interface via PCAP, so SEM must relinquish control of the ICAP interface.

The PS asserts ICAP Release (GPIO LED [7]) and waits for the SEM controller to deassert ICAP Request (GPIO LED [5]). After SEM deasserts ICAP Request, it automatically transitions to the Idle state. The PS disables the BUFGCE clock (GPIO LED [0]). The SEM controller is now ready for reconfiguration to occur.

The PS enables the PCAP and performs multiboot. Refer to the application (sem_icap_mboot_2.c) for more details.

---

**Figure 37:** Design 2 PS and SEM UART Output
Debugging

Ensure all diagram connections match those in Figure 2 and Figure 3.

Limitations and Considerations

**IMPORTANT:** This application and all supporting files are for demonstration purposes only. All files are delivered as is. This reference design is not intended to be a drop-in solution. After integrating this reference design into a new design, thorough verification is required to ensure the resulting solution meets functional and reliability goals.

**SEM Controller IP**

For the latest information, see the *UltraScale Architecture Soft Error Mitigation Controller LogiCORE IP Product Guide* (PG187) [Ref 2]. Note the following:

- The SEM controller does not operate on soft errors in block memory, distributed memory, or flip-flops. Soft error mitigation in these memory resources must be addressed by the user logic through preventive measures such as redundancy or error detection and correction codes.

- The SEM controller initializes and manages the FPGA integrated silicon features for soft error mitigation and when included in a design, does not include any design constraints or options that would enable the built-in detection functions. For example, do not set POST_CRC, POST_CONFIG_CRC, or any other related constraints. Similarly, do not include options to modify GLUTMASK.

- Software-computed ECC and CRC values are not supported.

- Design simulations that instantiate the SEM controller are supported. However, it is not possible to observe the controller behaviors in simulation. Design simulation including the controller compiles, but the controller does not exit the Initialization state. Hardware-based evaluation of the controller behaviors is required.

- Use of bitstream security (encryption and authentication) is currently not supported by the controller.

- Use of SelectMAP persistence is not supported by the controller.

- Only a single ICAP instance is supported for each SEM controller and it must reside at the primary/top physical location.

SEM IP are not supported when targeting UltraScale+ ES1 devices where the GTH and GTY transceivers’ dynamic reconfiguration port (DRP) is connected to user logic or IP.
Targeting Other Software Versions

**SEM IP**

The ZCU102 board uses a ZU9EG device. SEM IP support of this device is not available in the Vivado IP integrator IP catalog before Vivado release 2016.3. The `MPSoC_SEM_ICAP_MBoot/ip` directory contains the SEM UltraScale+ packaged design (`sem_ultra_v3_1`) which is required only for designs before 2016.3. From 2016.3 onward, you can generate the SEM IP directly in the Vivado IP integrator using the IP catalog.

**ZCU102 Board Files**

Starting in Vivado release 2016.4, the ZCU102 board files are included with the Vivado tools. These board files are for board revision 1.0. For designs targeting release 2016.4 or later with board revision 1.0 or later, you do not need to download any board files.

Download the board files from the [Zynq UltraScale+ ZCU102 HeadStart](#) website:

- If you are using Vivado release 2016.4 with a board revision before 1.0 (such as Rev D).
- If you are using a Vivado release prior to 2016.4.

**IMPORTANT:** Be careful to download the board files for your specific Vivado release and board revision.

---

**References**

**RECOMMENDED:** This application note was developed using Vivado Design Suite 2016.2. It is preferable to use the 2016.2 versions of Zynq UltraScale + MPSoC Technical Reference Manual (UG1085) [Ref 3] and Zynq UltraScale+ MPSoC Register Reference (UG1087) [Ref 4].

1. Integrating LogiCORE SEM IP with AXI in Zynq UltraScale+ Devices ([XAPP1303](#))
2. UltraScale Architecture Soft Error Mitigation Controller LogiCORE IP Product Guide ([PG187](#))
4. Zynq UltraScale+ MPSoC Register Reference ([UG1087](#))
5. Silicon Labs CP210x USB-to-UART Installation Guide ([UG1033](#))
## Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>02/10/2017</td>
<td>1.0</td>
<td>Initial Xilinx release.</td>
</tr>
<tr>
<td>02/13/2017</td>
<td>1.0.1</td>
<td>Made typographical changes.</td>
</tr>
<tr>
<td>02/27/2017</td>
<td>1.0.2</td>
<td>An important note was added to the Reference Design and Limitations and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Considerations sections.</td>
</tr>
</tbody>
</table>

## Please Read: Important Legal Notices

The information disclosed to you hereunder (the “Materials”) is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available “AS IS” and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx’s limited warranty, please refer to Xilinx’s Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx’s Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos.

**AUTOMOTIVE APPLICATIONS DISCLAIMER**

AUTOMOTIVE PRODUCTS (IDENTIFIED AS “XA” IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE (“SAFETY APPLICATION”) UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD (“SAFETY DESIGN”). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

© Copyright 2017–2018 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. ARM is a registered trademark of ARM in the EU and other countries. All other trademarks are the property of their respective owners.