Direct Memory Access using CDMA

In the Zynq, multiple interconnections are available between the PS and PL sections with different performance levels for data transfer between the two subsystems. The General Purpose (GP) Master and Slave AXI interconnect are intended for peripherals that do not have high bandwidth requirements. There are four High Performance PS slave to PL master AXI interfaces available for peripherals that need higher bandwidth such as video and image processing applications. In this demo, we will enable a High Performance AXI slave port in the PS, add an AXI central DMA (CDMA) controller, and perform Direct Memory Access (DMA) operations between various memories.

Design Description

In this demo, we will enable the HP port of the PS and add an instance of the Central DMA (CDMA) controller in the PL. We will add two instances of an AXI-BRAM controller, one will access the second port of the BRAM via the processor. We will connect the interrupt request line from the CDMA to the input of the GIC of the PS. The following diagram represents the completed design (**Figure 1**).

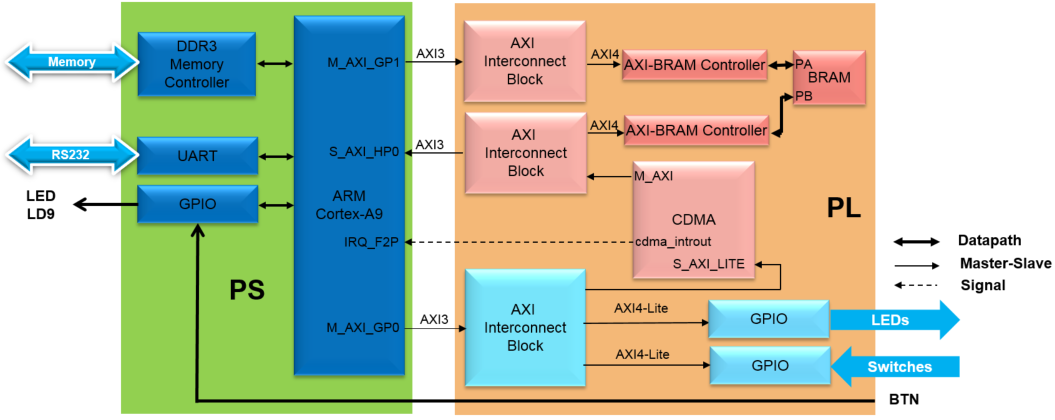


Figure 1. Completed Design

1. Create the Project Step
   1. Open the Vivado program. Create a project with GPIO and one BRAM.
      1. Start Vivado.
      2. Create a block diagram with a Zynq processing system and select *Run Block Automation*.
      3. Double-click on the *Zynq processing system* instance to open its configuration form.
      4. Click on the MIO configuration panel to open its configuration. Expand the I/O Peripherals (and GPIO). Deselect ENET 0, USB 0, SD 0, and expand GPIO and deselect USB Reset and I2C Reset. These are things we are not going to use, so we can disable them to keep the Zynq system simpler.
      5. Create 2 AXI GPIO ports, one for switches and one for LEDs. Use *AXI\_GPIO\_0* for the switches and *AXI\_GPIO\_1* for the LEDs. Connect them to the board interface. Run *Connection Automation* selecting both GPIO components. The block design is shown below.

A screenshot of a computer

Description automatically generated

**Figure 2. Initial Block Design with GPIO**

1. Configure the Processor to Enable M\_AXI\_GP1
   1. Open the Block Design and enable the M\_AXI\_GP1 interface.
      1. Double-click on the *Zynq processing system* instance to open its configuration form.
      2. Select *PS-PL Configuration* in the Page Navigator window in the left pane, expand *AXI Non Secure Enablement>GP Master AXI Interface*, and click on the check-box of the **M\_AXI GP1 Interface** to enable it.
      3. Select *Clock Configuration* in the Page Navigator window in the left pane, expand *PL Fabric Clocks* on the right, and click on the check-box of the **FCLK\_CLK1** to enable it.
      4. Enter the*Requested Frequency*for the **FCLK\_CLK1** as **140.00000** MHz.
      5. Click **OK** to accept the settings and close the configuration form.

A screenshot of a computer

Description automatically generated

Figure 3. M\_AXI\_GP1 interface enabled

1. Extend with BRAM Step 3
   1. Add an AXI BRAM Controller instance with BRAM.
      1. Click the  button and search for **BRAM** in the catalog.
      2. Double-click the AXI BRAM Controller to add an instance to the design.
      3. Click on Run Connection Automation, and select axi\_bram\_ctrl\_0
      4. Click on **BRAM\_PORTA** and **BRAM\_PORTB** check boxes.
      5. Click **S\_AXI**, and change the *Master* option to **/processing\_system7\_0/M\_AXI\_GP1**, change the *Crossbar clock source of interconnect IP, Clock source for Master interface,* and *Clock source for Salve interface* to **/processing\_system7\_0/FCLK\_CLK1 (140 MHz)** as they all run in the same clock domain, and click **OK**

A screenshot of a computer

Description automatically generated

Figure 4. Connecting AXI BRAM Controller to M\_AXI\_GP1 to run at faster clock speed

Notice that an instance of AXI SmartConnect and Processor System Reset are added, and the M\_AXI\_GP1\_ ACLK is connected to FCLK\_CLK1.

A screenshot of a computer

Description automatically generated

Figure 5. Clocking network connections

* + 1. Double-click on the **axi\_bram\_ctrl\_0** instance to open the configuration form.
    2. Set the *Data Width* to **64**.

A white background with black text

Description automatically generated

Figure 6. Setting the BRAM controller data width to 64

* + 1. Click **OK**.
  1. Using the Address Editor tab, set the BRAM controller size to 64KB. Validate the design.
     1. Select the Address Editor tab and notice that the BRAM controller memory space is **8K**.
     2. Click in the *Range* column of the *axi\_bram\_ctrl\_0* instance and set the size as **64K**.

A screenshot of a computer

Description automatically generated

Figure 7. AXI BRAM space assignment

Notice that the address range changed to 0x80000000-0x8000FFFF. This is in the M\_GP1 addressing space.

* + 1. Select the *Diagram* tab, and click on the  (Validate Design) button to make sure that there are no errors.

1. Configure the Processor to Enable S\_AXI\_HP0 Step 2
   1. Open the Block Design and enable the S\_AXI\_HP0 interface
      1. Click the **Diagram** tab.
      2. Double-click on the *Zynq processing system* instance to open its configuration form.
      3. Select *PS-PL Configuration* in the Page Navigator window in the left pane, expand *HP Slave AXI Interface* on the right, and click on the check-box of the **S AXI HP0 Interface** to enable it, and click **OK** to close the Configuration window.
2. Add CDMA and BRAM Step 3
   1. Instantiate the AXI central DMA controller.
      1. Click the  button and search for **Central** in the catalog.
      2. Double-click the AXI Central Direct Memory Access to add an instance to the design.
      3. Double-click on the *axi\_cdma\_0* instance and uncheck the *Enable Scatter Gather* option.
      4. Change the *Write/Read Data Width* to **64** and click **OK**.

Note the burst size changes from 16 to 8. You can increase this up to 256 to improve the performance. Here we are using smallest number since the application allows small number of words transfer.

* 1. Run connection automation

Connection automation could be run on all unconnected ports simultaneously. For the purposes of this demo, each port will be connected separately so that the changes made by the automation process are easier to follow.

* + 1. Click on **Run Connection Automation** and select **processing\_system7\_0/S\_AXI\_HP0** only.
    2. Check that this port will be connected to the */axi\_cdma\_0/M\_AXI* port and click **OK**.

A screenshot of a computer

Description automatically generated

Figure 8. Connection automation

* + 1. Verify the CDMA connection through the AXI SmartConnect to the HP0 port

A diagram of a computer

Description automatically generated

Figure 9. Connecting AXI Central DMA controller to S\_AXI\_HP0

Notice that an instance of AXI Memory Interconnect (axi\_mic\_1) is added, S\_AXI\_HP0 of the processing\_system7\_0 is connected to M00\_AXI of the axi\_mic\_1, S00\_AXI of the axi\_mic\_1 is connected to the m\_axi of the axi\_cdm\_0 instance. Also, m\_axi\_aclk of the axi\_cdma\_0 is connected to the net originating from FCLK\_CLK0 of the processing\_system7\_0.

* + 1. Click on **Run Connection Automation** again, and select **/axi\_cdma\_0** (which includes **S\_AXI\_LITE**).

Notice that the axi\_cdma\_0/M\_AXI port is no longer available to select. This is because it was connected to the processing system in the previous step.

* + 1. Ensure /processing\_system7\_0/*M\_AXI\_GP0* is selected in the drop-down button and click **OK**.

A diagram of a computer

Description automatically generated

Figure 10. CDMA connected

* 1. Instantiate another BRAM Controller and a BRAM.
     1. Click the  button and search for **BRAM** in the catalog.
     2. Double-click the AXI BRAM Controller to add an instance to the design.
     3. Click on **Run Connection Automation**, and select **/axi\_bram\_ctrl\_1/S\_AXI** only.
     4. For the *Master* connection, *s*elect **axi\_cdma\_0/M\_AXI** from the dropdown box.

A screenshot of a computer

Description automatically generated

Figure 11. BRAM connection automation

* + 1. Click **OK** toconnect to make the connection.

Notice that another axi interface (M01\_AXI) is added to the axi\_smc\_1 instance and is connected to the S\_AXI interface of the axi\_bram\_ctrl\_1 instance.

A computer diagram with blue lines

Description automatically generated with medium confidence

Figure 12. Connection between the new BRAM controller to the CDMA

* + 1. Double-click the *axi\_bram\_ctrl\_1* instance and change the *Number of BRAM Interface* to **1**.
    2. Change the *Data Width* to **64** and click **OK**.
    3. Double-click the *axi\_bram\_ctrl\_0* instance and also change the *Number of BRAM Interface* to **1**. Click **OK**.
    4. Using the wire tool, connect the **BRAM\_PORTA** of the *axi\_bram\_ctrl\_1* instance to the **BRAM\_PORTB** of the Block Memory Generator *axi\_bram\_ctrl\_0\_bram* instance.

A computer diagram of a computer

Description automatically generated with medium confidence

Figure 13. Connect the second BRAM controller

* 1. Connect the CDMA interrupt out port to the port of the processor.
     1. Double-click on the *processing\_system7\_0 instance* to open its configuration form.
     2. Select *Interrupts* in the Page Navigator window in the left pane, check the *Fabric Interrupts* box.
     3. Expand *Fabric Interrupts > PL-PS Interrupts Ports*, and click on the check-box of the **IRQ\_F2P**.

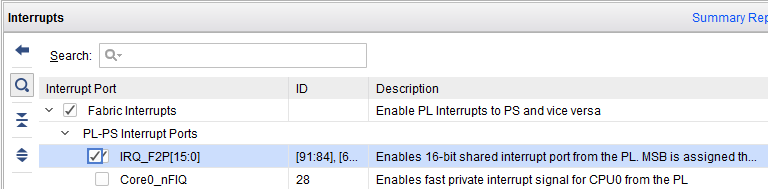


Figure 14. Enabling the processor interrupt

* + 1. Click **OK**.
    2. Using wiring tool, connect the **cdma\_introut** to the **IRQ\_F2P** port. (Click on the *cdma\_introut* port and drag to the *IRQ\_F2P* port)
  1. Using the Address Editor tab, set the BRAM controller size to 64KB. Validate the design.
     1. Select the Address Editor tab.
     2. Expand the *axi\_cdma\_0> Data* section, and change the memory size of *axi\_bram\_ctrl\_1* to **64K**.

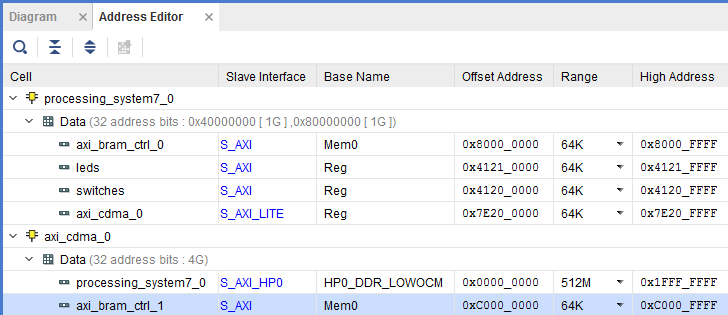


Figure 15. Address space

* + 1. The design should look similar to the figure below.

A diagram of a computer flowchart

Description automatically generated

Figure 16. Completed design

* + 1. Select the *Diagram* tab, and click on the  (Validate Design) button to make sure that there are no errors.

1. Create and HDL Wrapper and Generate the Bitstream Step 4
   * 1. Create an HDL Wrapper by right-clicking on design\_1.bd in the design sources window under the Sources tab and select **Create HDL Wrapper**. Click **OK** to the default selection to let Vivado manage the wrapper.
     2. Click on the **Generate Bitstream** to run the synthesis, implementation, and bit generation processes.
     3. Click **Save** to save the project, and **Yes** if prompted to run the processes. Click **OK** to launch the runs.
     4. When the bitstream generation process has completed successfully, click **Cancel**.
2. Generate an Application in Vitis Step 5
   1. Export the implemented design, and start Vitis
      1. Export the hardware configuration by clicking **File > Export > Export Hardware…**
      2. Click the box to *Include Bitstream*and click **OK**
      3. Open Vitis
      4. To clean the workspace, right-click on each open project except *system\_wrapper\_hw\_platform\_2* and select close project.
   2. Create an empty application project, named demoCDMA, and import the provided demoCDMA.c file.
      1. Create a new Workspace by entering workspace\_**demoCDMA**.
      2. Select **Create Application Project.**
      3. In the *Project Name* field, enter **demoCDMA** as the project name, leave all other settings to their default’s and click **Next.**
      4. For the Platform, select the **Create a new platform from Hardware (XSA)** tab. Then click the plus sign and navigate to the directory containing your Vivado project. Select **design\_1\_wrapper.xsa**. This is the export of your hardware description. Then click **Next**.
      5. For the Domain, leave the CPU, OS, Language and boot components as defaulted, click **Next**.
      6. Select the **Empty Application** template and click **Finish.**
      7. Select **demoCDMA > src** in the project view, right-click, and select **Import Sources.**
      8. Click **Browse** next to the From Directory**.**
      9. Browse to the folder containing the demoCDMA.c file that you downloaded with this demo and click **Select Folder**.
      10. In the panel on the right, check the box next to **demoCDMA.c** and click **Finish.**
      11. In the Explorer tab, select the demo\_CDMA\_system and click **Project > Build Project**.
3. Test in Hardware Step 6
   1. Connect and power up the board. Download the bitstream and program the FPGA.
      1. Connect and power up the board.
      2. In Vitis, select **Xilinx Tools > Program FPGA.**
      3. Click the **Program** button to program the FPGA.
   2. Establish serial communication, and run the demoCDMA application from the DDR3 memory.
      1. Right click on *demoCDMA*, and select **Debug As > Debug Configurations**
      2. Double click on Single Application Debug (GDB) to create a new configuration (*Debugger\_demoCDMA-GDB will be created*)**.**
      3. Select the **demoCDMA** project in *Project Explorer*, right-click and select **Debug As > Debug Configurations**. From the debug configuration options, select **Debugger\_demoCDMA-GDB** that we just created and configured. Then click **Debug** to download the application and execute ps7\_init. (If prompted, click **Yes** to switch to the Debug perspective.) The program execution starts and suspends at the entry point.
      4. Select the A close up of a word

         Description automatically generated tab. If it is not visible then select **Window > Show view > Terminal**.
      5. Click on A green cross on a white background

         Description automatically generated and select the appropriate COM port (depending on your computer), and configure it.
      6. Click the resume button in the debug window. The program will give you options in the Vitis Serial Terminal**.**

Follow the menu in the terminal emulator window and test transfers between various memories.

In order to send input through the Vitis Serial Terminal, use the input box near the bottom and the *Send* button.

A screenshot of a computer

Description automatically generated

Figure 17. Vitis Serial Terminal Program Run

* + 1. Select option 4 in the menu to complete the execution.
    2. Close Vitis and Vivado programs by selecting **File > Exit** in each program.
    3. Turn OFF the power on the board.

This demo led you through adding a CDMA controller to the PS so that we can perform DMA transfers between various memories. We used the high-performance port so DMA could be done between the BRAM residing in PL and DDR3 connected to the PS. We verified the design functionality by creating an application and executing it from the DDR3 memory.