Alveo U280 Data Center Accelerator Card

User Guide

UG1314 (v1.2.1) November 20, 2019

XILINX®

Revision History

The following table shows the revision history for this document.

Section	Revision Summary
11/20/2019 Version 1.2.1	
General updates.	Editorial updates only. No technical content updates.
10/31/2019 Version 1.2	
All sections.	Updated to Vitis™ taxonomy.
06/28/2019 Version 1.1	
Chapter 2: Vitis Design Flow	New section.
02/11/2019 Version 1.0	
Initial Xilinx release.	N/A

Table of Contents

Chapter 1

Introduction

The Xilinx® Alveo™ U280 Data Center accelerator cards are Peripheral Component Interconnect express (PCIe®) Gen3 x16 compliant and Gen4 x8 compatible cards featuring the Xilinx 16 nm UltraScale+™ technology. The Alveo U280 card offers 8 GB of HBM2 at 460 GB/s bandwidth to provide high-performance, adaptable acceleration for memory-bound, compute-intensive applications including database, analytics, and machine learning inference.

The Alveo U280 Data Center accelerator cards are available in passive and active cooling configurations. Except where noted, this user guide applies to both the active and passive versions of the U280 card. The following figure shows a passively cooled Alveo U280 accelerator card.

Figure 1: **Alveo U280 Data Center Accelerator Card (Passive Cooling)**

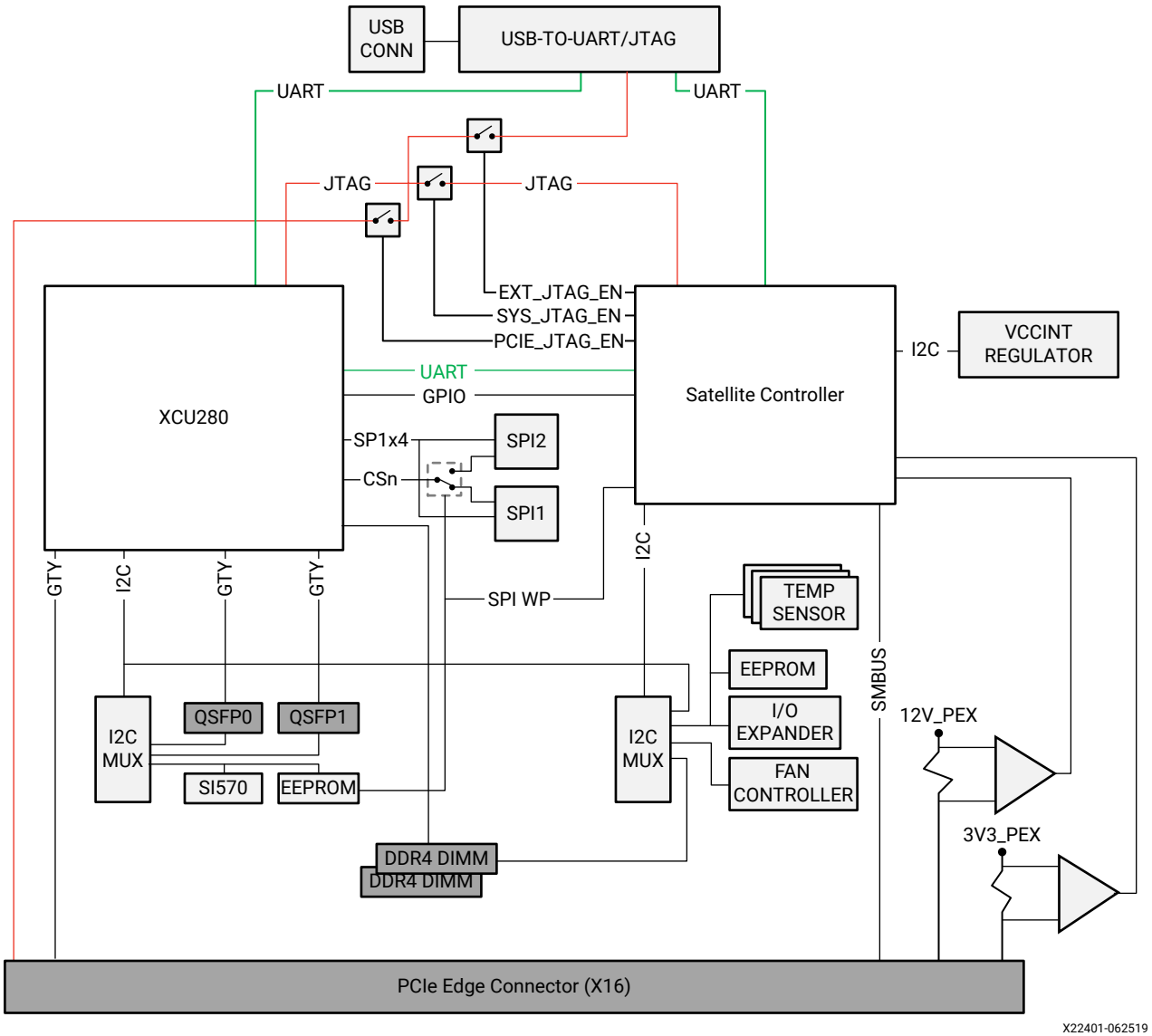
CAUTION! *The Alveo U280 accelerator card with passive cooling is designed to be installed into a data center server, where controlled air flow provides direct cooling. Due to the card enclosure, switches and LEDs are not accessible or visible. The card details in this user guide are provided to aid understanding of the card features. If the cooling enclosure is removed from the card and the card is powered-up, external fan cooling airflow MUST be applied to prevent over-temperature shut-down and possible damage to the card electronics. Removing the cooling enclosure voids the board warranty.*

See Appendix B: Additional Resources and Legal Notices for references to documents, files, and resources relevant to the Alveo U280 accelerator cards.

Block Diagram

The block diagram of the Alveo U280 accelerator card is shown in the following figure.

Figure 2: **U280 Block Diagram**



Card Features

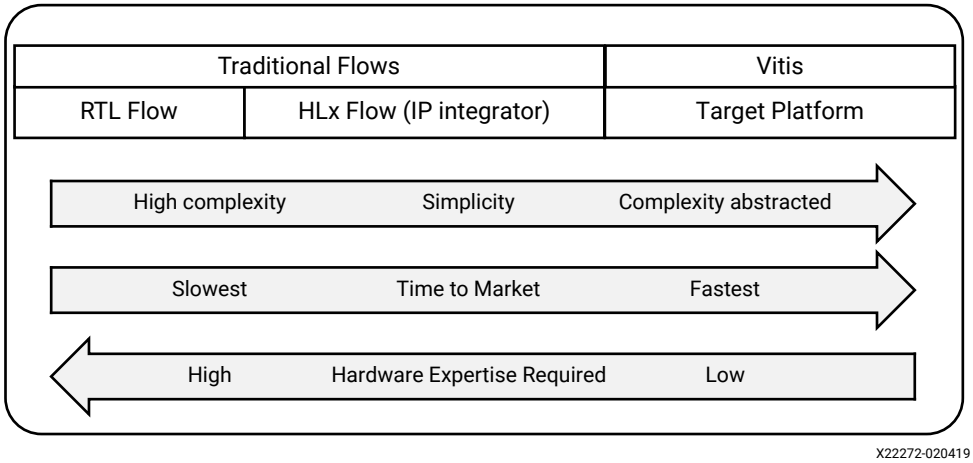
The Alveo U280 accelerator card features are listed in this section. Detailed information for each feature is provided in Chapter 5: Card Component Description.

- Alveo U280 accelerator card:
  - XCU280 FPGA
- Memory (two independent dual-rank DDR4 interfaces)
  - 32 gigabyte (GB) DDR4 memory
  - 2x DDR4 16 GB, 2400 mega-transfers per second (MT/s), 64-bit with error correcting code (ECC) DIMM
  - x4/x8 unregistered dual inline memory module (UDIMM) support
- Configuration options
  - 1 gigabit (Gb) Quad Serial Peripheral Interface (SPI) flash memory
  - Micro-AB universal serial bus (USB) JTAG configuration port
- 16-lane PCI Express
- Two QSFP28 connectors
- USB-to-UART FT4232HQ bridge with Micro-AB USB connector
- PCIe Integrated Endpoint block connectivity
  - Gen1, 2, or 3 up to x16
  - Gen4 x8
- I2C bus
- Status LEDs
- Power management with system management bus (SMBus) voltage, current, and temperature monitoring
- Dynamic power sourcing based on external power supplied
- 65W PCIe slot functional with PCIe slot power only
- 150 W PCIe slot functional with PCIe slot power and 6-pin PCIe AUX power cable connected
- 225 W PCIe slot functional with PCIe slot power and 8-pin PCIe AUX power cable connected
- Onboard reprogrammable flash configuration memory
- Front panel JTAG and universal asynchronous receiver-transmitter (UART) access through the USB port
- FPGA configurable over USB/JTAG and Quad SPI configuration flash memory

Design Flows

The preferred optimal design flow for targeting the Alveo Data Center accelerator card uses the Vitis™ unified software platform. However, traditional design flows, such as RTL or HLx are also supported using the Vivado® Design Suite tools. The following figure shows a summary of the design flows.

Figure 3: **Alveo Data Center Accelerator Card Design Flows**



Requirements for the different design flows are listed in the following table.

Table 1: **Requirements to Get Started with Alveo Data Center Accelerator Card Design Flows**

	RTL Flow	HLx Flow	Vitis
Flow documentation	UG949[1]	UG895[2]	UG1416[3]
Hardware documentation	UG1314	UG1314	N/A
Vivado tools support	Board support XDC	Board support XDC	N/A
Programming the FPGA	Vivado Hardware Manager	Vivado Hardware Manager	UG1301[4]

Notes:

1. *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949).
2. *Vivado Design Suite User Guide: System-Level Design Entry* (UG895). See "Using the Vivado Design Suite Platform Board Flow" in Chapter 2 and Appendix A.
3. Vitis Accelerated Flow in the *Vitis Unified Software Platform Documentation* (UG1416).
4. *Getting Started with Alveo Data Center Accelerator Cards* (UG1301).

Vitis Design Flow

The following section is aimed at designers, and is intended as a quick start guide to the U280 target platform. It describes how to link a kernel into the target platform and then run this kernel in hardware on the platform. Background information on the structure of the platform is also provided.

For more details on the Vitis™ and hardware platforms, refer to the Vitis Accelerated Flow in the *Vitis Unified Software Platform Documentation* (UG1416) and Embedded Processor Platform Development in the *Vitis Unified Software Platform Documentation* (UG1416).

Target Platform Overview

The target platform consists of a Vivado® IP integrator block design with all of the required board interfaces configured and connected to the device I/Os.

The Vitis environment platforms require the device to remain up and running on a remote host while applications are downloaded to it. Vitis target platforms use partial reconfiguration (PR) technology to enable compiled binary downloads to the accelerator device while the device remains online and linked to the PCIe® bus of the host. There is a fixed static logic partition that contains the board interface logic. The static partition is not reimplemented when the Vitis applications are run. This is known as a target platform because it describes the boundary of the physical board with the kernel logic. The target platform is intended to keep the device alive and to control reconfiguration and debugging of the device.

The dynamically programmable region defines the programmable device logic partition that integrates the user kernels from the Vitis environment. It contains memories, memory interfaces, and AXI interconnect logic. This logic is dynamically configured and implemented along with the Vitis kernel logic each time an application is run. Each time a kernel application is run, partial reconfiguration technology is applied to reconfigure the dynamic region.

The dynamically programmable region has several DDR/HBM/PLRAM memory interfaces coupled with interconnect logic. The term PLRAM refers to internal UltraRAM/block RAM that can be accessed by host and user kernels. The dynamically programmable region uses the Memory Subsystem (MSS) IP (for DDR/PLRAM) and the HBM Memory Subsystem (HMSS) IP (for HBM). These subsystems are unique to the Vitis platforms. They contain multiple memory interfaces, coupled with the appropriate interconnect IP. When the dynamic region is being built

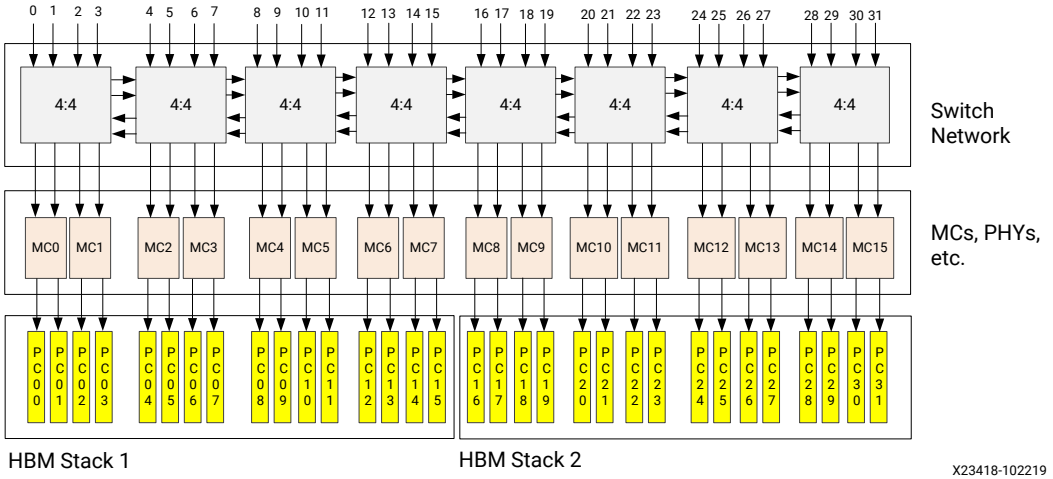
(that is, when kernels are applied and connected to memory resources), the IP automatically trim away any of the unused DDR/PLRAM interfaces and disable the unused HBM interfaces which reside in the dynamic region. This allows for more of the device to be available for kernel logic, and reduces run time. Dynamic memory enablement is also optimal for power, because unused memories consume minimal power.

Target Platform Detail

U280 High Bandwidth Memory (HBM) devices incorporate 4 GB HBM stacks. Using stacked silicon interconnect technology, the programmable logic communicates to the HBM stacks through memory controllers. The U280 has access to two stacks of 4 GB HBM, each consisting of 16 pseudo channels with direct access to 256 MB. A high-level diagram of the two HBM stacks is shown below.

The programmable logic has 32 HBM AXI interfaces. HBM AXI interfaces can access any memory location in any of the 32 HBM PCs on either of the HBM stacks through a built-in switch providing access to the full 8 GB memory space. For more detailed information on the HBM, refer to *AXI High Bandwidth Controller LogiCORE IP Product Guide* (PG276). The flexible connection between the programmable logic and the HBM stacks results in easy floorplanning and timing closure as well as offering flexibility for kernel implementation.

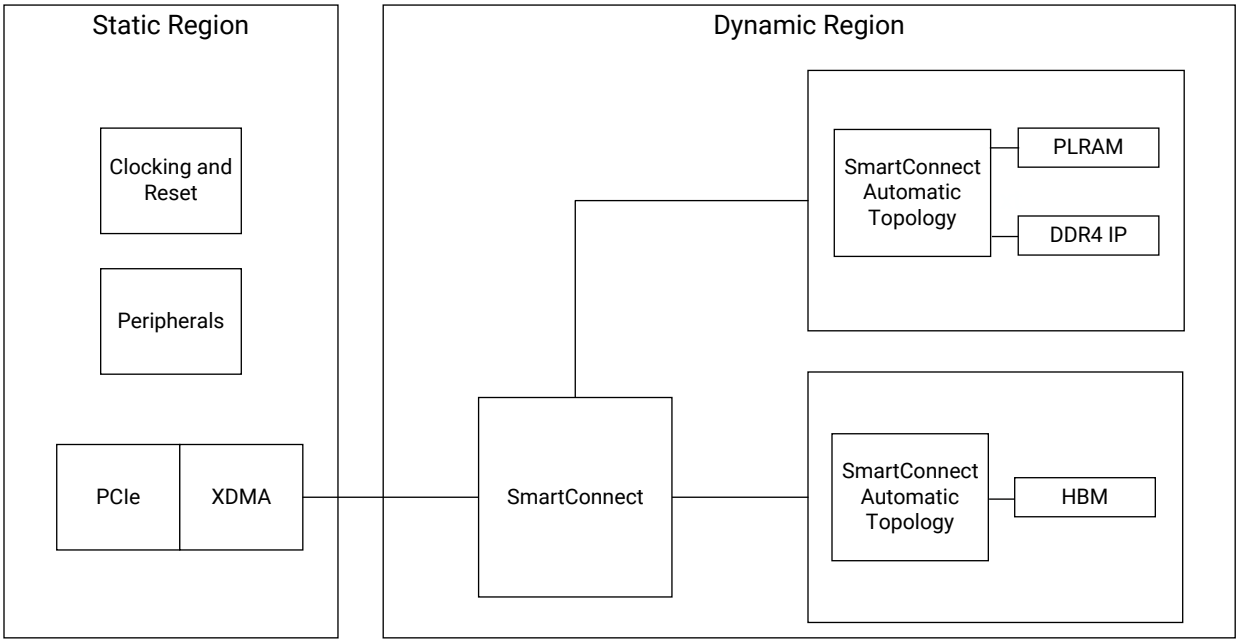
Figure 4: **High-Level Diagram of Two HBM Stacks**



In addition to HBM, two Memory IP instances enable access to on-board DDR4 SDRAM. The U280 accelerator card contains two channels of DDR4-2400 SDRAM, at 16 GB per channel for a total of 40 GB global memory between internal HBM and external DDR4.

The following figure shows a high-level view of the target platform and dynamic region in the U280 card.

Figure 5: **Static Region (Target Platform) and Dynamic Region of U280 Card**



Vitis HBM Memory Subsystem

The complexity of the HBM and required ancillary soft logic are abstracted by the HBM Memory Subsystem (HMSS) IP. When creating the target platform, the HMSS instantiates the HBM, enables all HBM PCs, and automatically connects the XDMA to the HBM for host access to global memory.

When the built target platform is used with the Vitis compiler, the HMSS is automatically re-customized by the flow to activate only the necessary memory controllers and ports as defined by kernel connectivity to HBM PCs, and to connect both the user kernels and the XDMA to those memory controllers for optimal bandwidth and latency.

It is important to specify the HBM PC mapping for each kernel in the system. Specifying this enables the optimal connection to be made from the kernel to the HBM within the HMSS.

The HMSS automatically handles the placement and timing complexities of AXI interfaces crossing super logic regions (SLRs) in SSI technology devices. You must specify the SLR location of the user kernels (this defaults to SLR0), which instructs the HMSS as to the location of each.

The HBM subsystem in U280 devices performs well in applications where sequential data access is required. However, for applications requiring random data access, performance can vary significantly depending on the application requirements (for example, the ratio of read and write operations, minimum transaction size, and size of the memory space being addressed). The Random Access Memory Attachment (RAMA) IP helps to address such problems by significantly improving memory access efficiency in cases where the required memory exceeds 256 MB (one HBM pseudo channel). Refer to *RAMA LogiCORE IP Product Guide* (PG310) for more information.

For using the RAMA IP to be considered, a kernel master should meet the following criteria:

- The master randomly accesses locations across more than one 256 MB HBM PC.
- The master uses a static, single ID on the AXI transaction ID ports (AxID), *or* the master uses slowly changing (pseudo-static) AXI transaction IDs.

If these conditions are not met, the thread creation used in the RAMA IP to improve performance has little effect, and consumes programmable logic for no purpose.

For details of how to specify the options for HBM PC mapping, SLR assignment, and RAMA insertion, refer to the Kernel Insertion section.

Related Information
Kernel Insertion

Memory Subsystem

When creating the target platform, the Memory Subsystem (MSS) instantiates all available DDR4 memory controllers and automatically constructs an optimized SmartConnect-based network to connect the XDMA to the memory controllers (two memory controllers in the case of the U280 card) for host access to global memory. PLRAM memory (internal SRAM – UltraRAM and block RAM) can also be accessed and is also created coupled to a similar optimal switch network.

When the built target platform is used with the Vitis compiler, the MSS is automatically re-customized by the flow to instantiate only the necessary memory as defined by kernel connectivity to memory resources, and to automatically construct an optimized SmartConnect-based network to connect both the user kernels and the XDMA to those memory resources. Unmapped memory resources are not instantiated, and the SmartConnect network is optimized accordingly.

The MSS automatically handles the placement and timing complexities of AXI interfaces crossing SLRs in SSI technology devices, in the event that user kernels in a given SLR are mapped to one or more memory controllers in a different SLR. Accordingly, you must specify the DDR/PLRAM resource requirements for each kernel in the system.

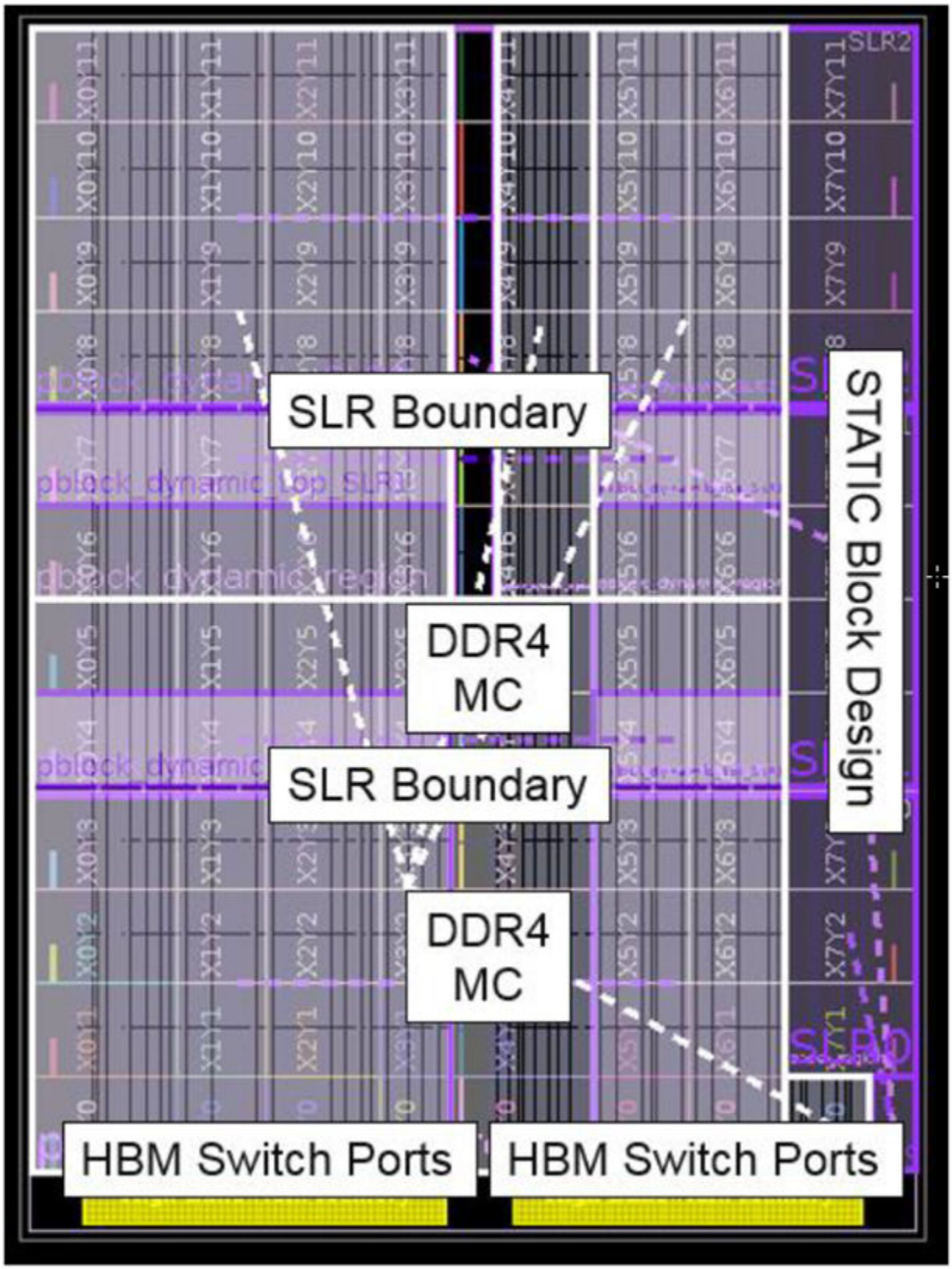
Key User Information

Target Platform Floorplan

The U280 Data Center accelerator card uses SSI technology and contains three super logic regions (SLRs). The static base region of the design is floorplanned to occupy a narrow region on the right of each SLR to allow a consistent large area in each SLR for acceleration components:

- The HBM ports are located in the bottom SLR of the device.
- One of the two DDR4 SDRAM memory controller IP instances is floorplanned to the bottom SLR (SLR0), in regions around the high performance I/O banks it uses.
- The other DDR4 SDRAM memory controller IP instance is floorplanned to the middle SLR (SLR1), in regions around the high performance I/O banks it uses.

Figure 6: **U280 Target Platform Floorplan**



The floorplan is arranged to maximize the number of resources available, and to ensure that a similar amount of logic is available per SLR. The following table shows an approximate resource count for the U280 platform per SLR.

Table 2: **U280 Resource Availability per SLR**

Area	SLR0	SLR1	SLR2
General Information			
SLR Description	Shared by dynamic and static region resources.	Shared by dynamic and static region resources.	Shared by dynamic and static region resources.
Dynamic Region Pblock Name	pfm_top_i_dynamic_region_pblock_dynamic_SLR0	pfm_top_i_dynamic_region_pblock_dynamic_SLR1	pfm_top_i_dynamic_region_pblock_dynamic_SLR2
Global Memory Resources Available in Dynamic Region			
System Port Name and Description	DDR[0]; (16 GB DDR4)	DDR[1]; (16 GB DDR4)	PLRAM[4:5]; (2 x up to 4 MB SRAM) Default of 128 KB
	HBM[0:31]; (32 x 256 MB HBM2 PC)	PLRAM[2:3]; (2 x up to 4 MB SRAM) Default of 128 KB	
	PLRAM[0:1]; (2x up to 4 MB SRAM) Default of 128 KB		
Approximate Available Fabric Resources in Dynamic Region			
CLB LUT	355K	340K	370K
CLB Register	725K	675K	734K
Block RAM Tile	490	490	510
UltraRAM	320	320	320
DSP	2733	2877	2880

You can also use the `platforminfo` utility to query the platform directly for usage information:

```
>platforminfo <path-to>/xilinx_u280_xdma_201920_1.xpfm
Basic Platform Information
…
Resource Availability...
  Total...
    LUTs:  1205303
    FFs:   2478166
    BRAMs: 1816
    DSPs:  9020
  Per SLR...
    SLR0:
      LUTs:  354687
      FFs:   724566
      BRAMs: 488
      DSPs:  2733
    SLR1:
      LUTs:  340382
      FFs:   673856
      BRAMs: 487
      DSPs:  2877
    SLR2:
```

```
      LUTs: 369885
      FFs: 733640
      BRAMs: 512
      DSPs: 2880
…
```

The available resources are what remains after the following reductions are taken into account:

- The target platform consumes a certain number of resources (11 out of 96 clock regions; 11.5% of the device plus some additional I/O).
- The dynamic region has additional non-kernel logic to allow access to memories for kernels and the target platform.

Kernel Insertion

The Vitis compiler (`v++`) is a standalone command line utility for compiling kernel accelerator functions and linking them with platforms supported by the Vitis environment. It supports kernels expressed in OpenCL, C, C++, and RTL (SystemVerilog, Verilog, or VHDL).

The `v++` utility has two distinct steps; compile and link. Compilation takes the kernel sources and returns a Xilinx Object (XO) file. The resulting XO files can then be linked into the target platform using the link step to produce the binary container (.xclbin) required by the host code. For full details of compiling a kernel using `v++` and for details of all options for compile and link, refer to the Vitis Accelerated Flow in the *Vitis Unified Software Platform Documentation* (UG1416).

This section details the options specific to the HBM when using `v++` in link mode.

When you are using the Xilinx target platform for the U280 card to connect to HBM, 32 ports are available for kernels to connect with each port able to access any region of both HBM stacks. Xilinx recommends using a maximum of 31 port connections, because this allows the host to have a dedicated connection to the HBM. The use of 32 ports is possible, but this puts host and kernel transactions in competition for bandwidth on one of the 32 ports. This has no effect unless host transactions are ongoing while the kernel is enabled, and kernel transactions are also ongoing.

RECOMMENDED: *Minimize the memory footprint of each kernel and partition memory access between kernels to avoid corruption of memory contents by competing kernels. Minimizing the memory footprint also allows the HMSS to make good decisions on the optimal connection. Accordingly, it is important to specify the HBM PC mapping for each kernel in the system.*

The HMSS automatically handles the placement and timing complexities of AXI interfaces crossing SLRs in SSI technology devices. You are required to specify the SLR location of the user kernels, which instructs the HMSS as to the location of each.

Basic Vitis Compiler Options

Table 3: **Basic Vitis Compiler Config File Options**

Option	Valid Values	Description
`--connectivity.sp` on command line or `[connectivity]` `sp =` in config file	<kernel_cu_name>.<kernel_arg>:<sptag>[min:max] The <kernel_cu_name> is specified in `v++ --nk` option. See the Vitis Accelerated Flow in the *Vitis Unified Software Platform Documentation* (UG1416) for more information on this option. The <kernel_arg> arguments can be obtained from the host code. The <sptag> specifies either a single HBM PC or a range of HBM PCs from 0 to 31, for example HBM[0], HBM[0:4].	Specifies HBM PC mapping for kernel.
`--connectivity.slr` on command line or `[connectivity]` `slr =` in config file	<kernel_cu_name>:<SLR> The <kernel_cu_name> is specified in the `v++ --nk` option. The <SLR> specifies an SLR in which the kernel will be located. Valid values are SLR0, SLR1 and SLR2.	Specifies an SLR in which the kernel will be located.

--sp Example

The kernel `dummy` reads input buffers `in1` and `in2` from HBM PCs 0 and 1 respectively, and writes output buffer `out` to HBM PCs 3–4. Each HBM PC is 256 MB, giving a total of 1 GB of memory access for this kernel.

```
v++ --connectivity.sp dummy.in1:HBM[0]
--connectivity.sp dummy.in2:HBM[1]
--connectivity.sp dummy.out:HBM[3:4] …

or in config file

[connectivity]
sp=dummy.in1:HBM[0]
sp=dummy.in2:HBM[2]
sp=dummy.out:HBM[3:4]
```

***Note*:** Only the mapping to the HBM PC is defined. The HMSS chooses the appropriate port to access the HBM to maximize bandwidth and minimize latency.

***Note*:** The example above targets only HBM. The memory resources in the platform are shown in the Memory Resource Availability per SLR table.

Memory Resource Availability per SLR

Table 4: **Memory Resource Availability per SLR**

Area	SLR0	SLR1	SLR2
Global Memory Resources Available in Dynamic Region			
System Port Name and Description	DDR[0]; (16 GB DDR4)	DDR[1]; (16 GB DDR4)	PLRAM[4:5]; (Two instances of up to 4 MB SRAM) 128 KB default
	HBM[0:31]; (32 instances of 256 MB HBM2 PC)	PLRAM[2:3]; (Two instances of up to 4 MB SRAM) 128 KB default	
	PLRAM[0:1]; (Two instances of up to 4 MB SRAM) 128 KB default		

The `platforminfo` utility can also be used to query the platform directly for memory resource information. The output below is shortened for brevity. The first emphasis section shows that HBM memory resource 31 (that is, HBM[31]) is in SLR0 and can have a maximum of 32 masters. The second emphasis section shows that PLRAM resource 5 (that is, PLRAM[5]) is in SLR2 and can have a maximum of 15 masters.

```
>platforminfo <path-to>/xilinx_u280_xdma_201920_1.xpfm
Basic Platform Information
...
Memory Info...
    Bus SP Tag: HBM
        Segment Index: 0
            Consumption: default
            SLR: SLR0
            Max Masters: 32

...

         Segment Index: 31
            Consumption: automatic
            SLR: SLR0
            Max Masters: 32
    Bus SP Tag: DDR
        Segment Index: 0
            Consumption: automatic
            SLR: SLR0
            Max Masters: 15
        Segment Index: 1
            Consumption: automatic
            SLR: SLR1
            Max Masters: 15
    Bus SP Tag: PLRAM
        Segment Index: 0
            Consumption: automatic
            SLR: SLR0
            Max Masters: 15
```

```
...

      Segment Index: 5
            Consumption: automatic
            SLR: SLR2
            Max Masters: 15

...
```

--slr Example

Given two instances of kernel dummy called `dummy_0` and `dummy_1`, place `dummy_0` in SLR0 and `dummy_1` in SLR1:

```
v++ --connectivity.slr dummy_0:SLR0 --connectivity.slr dummy_1:SLR1

or in configuration file

[connectivity]
slr=dummy_0:SLR0
slr=dummy_1:SLR1
```

HMSS RAMA IP Insertion

To insert RAMA IP for a kernel master (as might be desired for multi-PC random access kernels), create a user Tcl file. The master ports of interest need to be defined in this file. To point to the user Tcl file, add the following option on the `v++` command line:

```
v++ --xp param:compiler.userPostSysLinkTcl=<full_path_to>/user_tcl_file.tcl
```

The user Tcl file must identify the ports of interest. For more information on the use of Tcl in Vivado, refer to *Vivado Design Suite User Guide: Using Tcl Scripting* (UG894).

RAMA Example

Using a kernel called `dummy` whose two master ports are random access (`M00_AXI`/`M01_AXI`), the format is as follows:

```
hbm_memory_subsystem::ra_master_interface <Endpoint AXI master interface>
[get_bd_cells hmss_0]
```

The specific Tcl for the dummy kernel example is as follows:

```
hbm_memory_subsystem::ra_master_interface [get_bd_intf_pins dummy/M00_AXI]
[get_bd_cells hmss_0]
hbm_memory_subsystem::ra_master_interface [get_bd_intf_pins dummy/M01_AXI]
[get_bd_cells hmss_0]
validate_bd_design -force
```

It is important for the user Tcl file to contain a `validate_bd_design` command at the end. This allows the information to be collected correctly by the HBM subsystem:

```
validate_bd_design -force
```

Note: Both HLS and RTL kernels should ensure at least 64 OTs (outstanding transactions) for high performance random access. 128 OTs might be beneficial in some cases.

HBM Performance

The unique feature of the U280 platform is the hardened HBM subsystem, which is encapsulated by the Vitis HBM Memory Subsystem (HMSS). Important performance figures, recommendations, and limitations are covered in this section.

Performance Figures

A summary of the most common and least complex use case performances is shown below:

- Linear access point-to-point: single kernel master per HBM PC.
- Small random accesses point-to-point: single kernel master per HBM PC.

For linear accesses, the efficiency for all transaction sizes (read-only and write-only transaction types) is 91% when the transaction size is larger than 32 bytes.

Table 5: **Linear Accesses (Single Kernel Master per HBM PC Performance)**

Transaction Size	Type	Bandwidth[1]	Efficiency	Total Bandwidth[2]
64 B+	WO	13.1	91%	419
64 B+	RO	13.1	91%	419
32 B	WO	4.5	32%	144
32 B	RO	4.5	31%	144

Notes:
1. One master (Gb/s).
2. 32 masters (Gb/s).

The achievable bandwidth (read-only and write-only) for small random accesses is shown below.

Table 6: **Small Random Accesses (Single Kernel Master per HBM PC performance)**

Transaction Size	Type	Bandwidth[1]	Efficiency	Total Bandwidth[2]
64 B	WO	4.9	32%	147
64 B	RO	5.6	35%	160
32 B	WO	4.5	31%	144

Table 6: **Small Random Accesses (Single Kernel Master per HBM PC performance)** *(cont'd)*

Transaction Size	Type	Bandwidth[1]	Efficiency	Total Bandwidth[2]
32 B	RO	4.4	31%	141

Notes:
1. One master (Gb/s).
2. 32 masters (Gb/s).

***Note*:** Because of the complexity and flexibility of the switch, there are many combinations that result in congestion at a particular memory location or in the switch itself. It is important to plan memory accesses so that kernels access limited memory where possible, and to isolate the memory accesses for different kernels into different HBM PCs.

Interleaved read and write transactions cause a drop in efficiency with respect to read-only or write-only due to memory controller timing parameters (bus turnaround). The exact performance drop is dictated by the read/write ratio and timing.

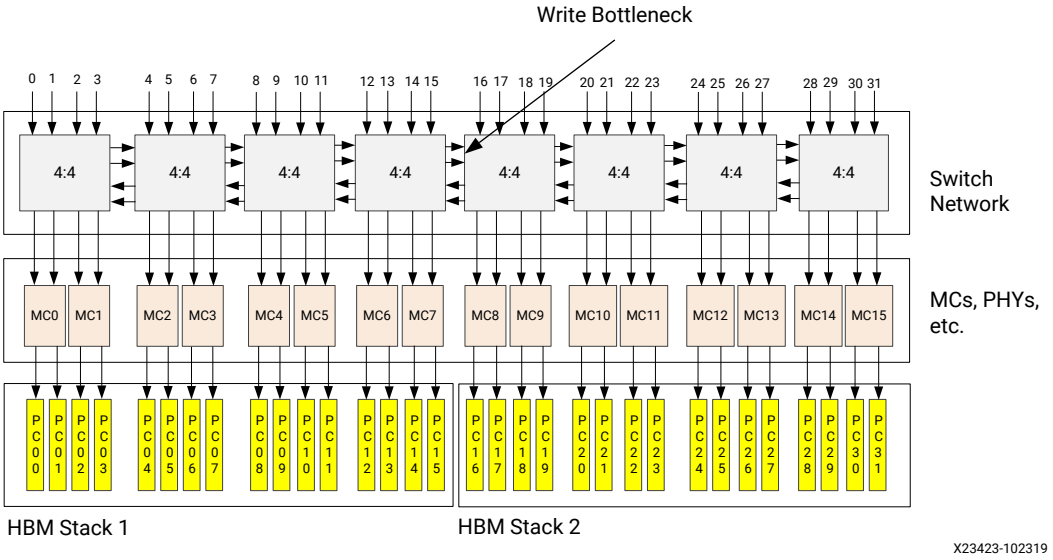
Performance Limitations

There is a bottleneck when a write transaction passes from HBM Stack 1 to HBM Stack 2. This bottleneck has the following effects:

- 33% of expected performance for 32B write transactions.
- 66% of expected performance for 64B write transactions.
- Other transaction sizes are not affected.
- Read transactions are not affected.

When planning memory resources, kernel masters planning to write at sizes of 32B/64B should not access HBM memory resources on both stacks.

Figure 7: **Write Bottleneck for Small Transactions**



Performance Notes

The following should be noted with respect to the platform:

- The host is connected to one of the HBM ports on the platform.
  - Up to 32 kernel masters can access HBM.
  - The HBM switch provides for 32 masters to be connected. The host therefore shares a switch port with one of the 32 kernel masters if required (this might not be optimal).
  - Xilinx recommends that no more than 31 kernel ports should be used if host and kernel interference is expected to cause a problem.
- A kernel requiring access to HBM and DDR4 must use separate masters.
  - This is a platform simplification.
  - DDR4/PLRAM access can be shared on a single master.
- A kernel master should only access a contiguous subset of the 32 HBM pseudo channels.
  - This makes optimization of the HBM connection point more straightforward.
- By default, without user memory planning (`--connectivity.sp` option on `v++`), all kernel masters access a single HBM PC.
- By default, all kernels are assumed in SLR0.
  - This might lead to device congestion for complex designs, leading to a need for kernel floorplanning.

Recommendations

Recommendations to get best performance from the platform are as follows:

- Select the kernel master data width and kernel frequency to match the following requirements:
  - The hardened HBM core clock runs at 450 MHz.
  - The hardened HBM AXI ports are 256 bits wide.
  - For linear transactions, the kernel should be configured to match the following bandwidth:
    - 256 bits at 450 MHz (this might be a challenging design).
    - 512 bits at 225 MHz (a larger but less challenging design).
  - It might be possible to achieve the same bandwidth with lower kernel frequency/bus width combinations. This depends on the traffic profile/switch usage.
  - For random transactions, it might be possible to scale the kernel master bandwidth down; a kernel clock of 300 MHz and a width of 256 bits, for example. This must be assessed on a case-by-case basis.
- Use only HBM memory resources where possible.
  - DDR interfaces consume a large number of resources in the middle of SLR0 and SLR1, and might in some cases cause congestion when routing to the HBM.
  - PLRAM resources also consume resources, but are more flexible in placement and are therefore preferable to DDR if HBM is not sufficient.
- The platform offers a large amount of flexibility in terms of memory access patterns.
  - Measure memory subsystem performance to validate assumptions when not using simple access (single master to single memory resource).
- Use as many HBM PCs as possible in parallel.
  - High bandwidth can only be achieved with many kernel ports and many HBM PCs being used in parallel.
- A kernel master should be capable of at least 64 outstanding read transactions and 32 outstanding write transactions for random access.
  - More than this might add some performance at the expense of increased logic (up to 128 outstanding transactions supported).
- Smaller transactions offer less efficiency than larger transactions. Use large transactions where possible.
- Avoid routing small write transactions across the boundary between HBM stacks 1 and 2.
  - When using 32B or 64B write transactions, ensure that a kernel accesses HBM[0:15] (stack 1) *or* HBM[16:31] (stack 2); not both.

PLRAM Use and Modification

The U280 platform contains HBM DRAM and DDR DRAM memory resources. An additional memory resource in the platform is internal FPGA SRAM (UltraRAM and block RAM). The platform initially contains six instances of 128 KB of PLRAM (block RAM). There are two PLRAM instances in each SLR. The size and type (UltraRAM/block RAM) of each PLRAM can be changed before kernels are inserted. A pre-system-linkage Tcl file is used to change the PLRAM. Usage of the pre-system-linkage Tcl file can be enabled as follows on the `v++` command line:

```
v++ --xp param:compiler.userPreSysLinkTcl=<full_path_to>/
pre_user_tcl_file.tcl
```

An API is provided to change attributes of the PLRAM instance or memory resource:

```
sdx_memory_subsystem::update_plram_specification <memory_subsystem_bdcell>
<plram_resource_name> <plram_specification>
```

In this API, `<plram_specification>` is a Tcl dictionary consisting of the following entries (entries below are the defaulted values for each instance in the platform):

```
 {
    SIZE 128K # Up to 4M
    AXI_DATA_WIDTH 512 # Up to 512
    SLR_ASSIGNMENT SLR0 # SLR0 / SLR1 / SLR2
    MEMORY_PRIMITIVE BRAM # BRAM or URAM
    READ_LATENCY 1 # To optimise timing path
}
```

In the example below, `PLRAM_MEM00` is changed to be 2 MB in size and composed of UltraRAM; `PLRAM_MEM01` is changed to be 4 MB in size and composed of UltraRAM. These resources correspond to the `v++` command line memory resources PLRAM[0:1].

```
# Setup PLRAM
sdx_memory_subsystem::update_plram_specification
[get_bd_cells /memory_subsystem] PLRAM_MEM00 { SIZE 2M AXI_DATA_WIDTH 512
SLR_ASSIGNMENT SLR0 READ_LATENCY 10 MEMORY_PRIMITIVE URAM}

sdx_memory_subsystem::update_plram_specification
[get_bd_cells /memory_subsystem] PLRAM_MEM01 { SIZE 4M AXI_DATA_WIDTH 512
SLR_ASSIGNMENT SLR0 READ_LATENCY 10 MEMORY_PRIMITIVE URAM}

validate_bd_design -force
save_bd_design
```

The `READ_LATENCY` is an important attribute, because it sets the number of pipeline stages between memories cascaded in depth. This varies by design, and affects the timing QoR of the platform and the eventual kernel clock rate. In the example above for `PLRAM_MEM01 (PLRAM[1])`:

- 4 MB of memory are required in total.

- Each UltraRAM is 32 KB (64 bits wide). 4 MB × 32 KB → 128 UltraRAMs in total.
- Each PLRAM instance is 512 bits wide → 8 UltraRAMs are required in width.
- 128 total UltraRAMs with 8 UltraRAMs in width → 16 UltraRAMs in depth.
- A good rule of thumb is to pick a read latency of depth/2 + 2 → in this case, `READ_LATENCY` = 10.

This allows a pipeline on every second UltraRAM, resulting in the following:

- Good timing performance between UltraRAMs.
- Placement flexibility; not all UltraRAMs need to be placed in the same UltraRAM column for cascade.

XILINX

Chapter 3

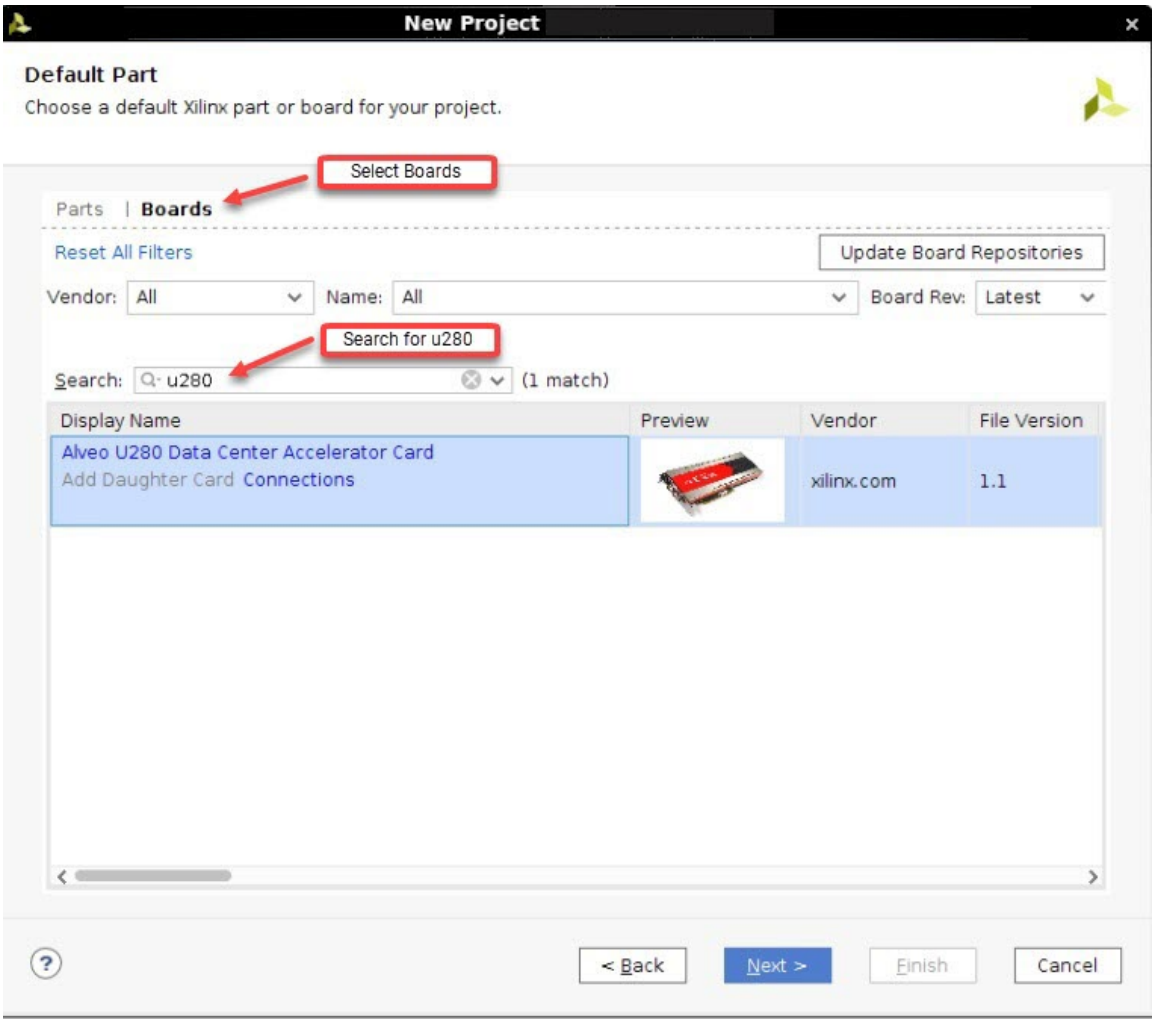
Vivado Design Flow

This section provides a starting point for expert HDL developers using the RTL flows, or developers who want to customize in HLx beyond the standard support in the Vivado® tools.

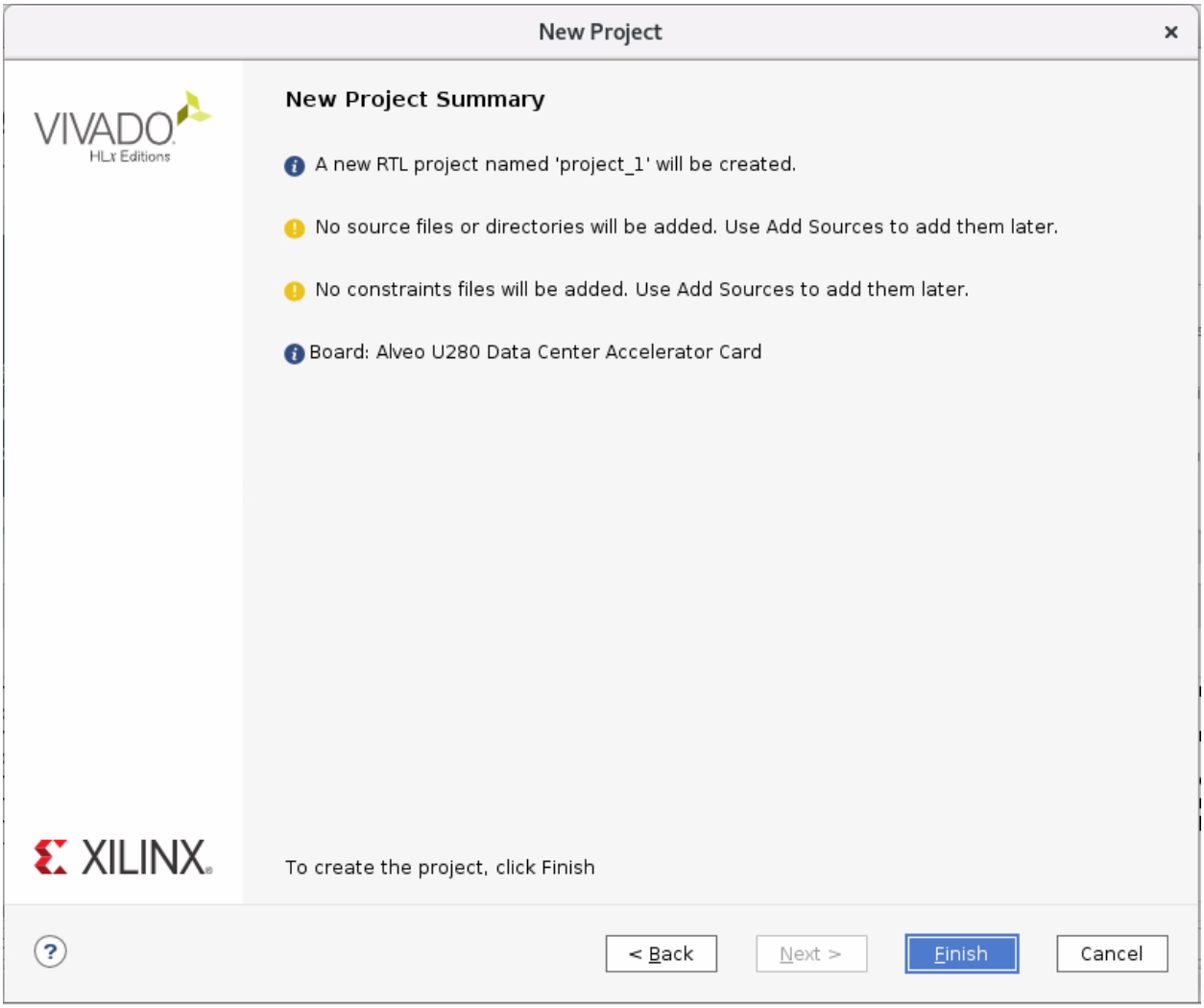
Downloading the Alveo U280 Card

For either the RTL or HLx flow, designers can start by downloading the U280 accelerator card files (zip).

1. In Vivado, select **Create New Project→RTL Project** to open the New Project dialog box.
2. Click **Update Board Repositories** for the latest board files, and then select the U280 card:

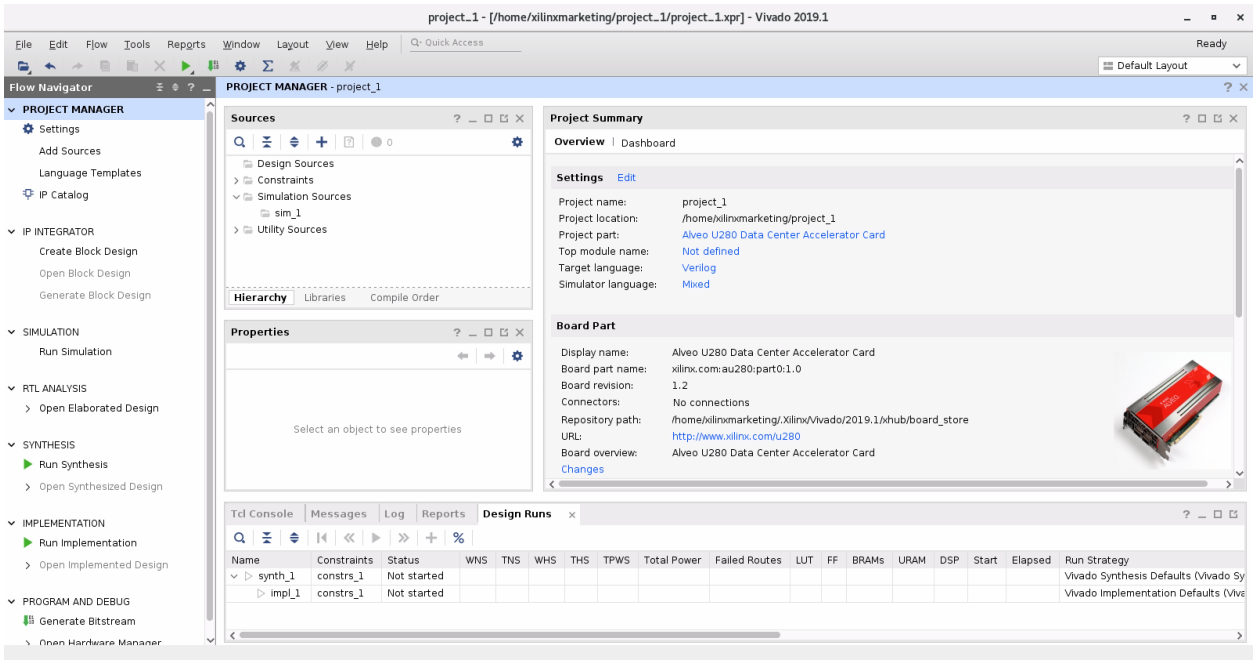


The New Project Summary dialog box appears.



3. Click **Finish**.

   The Project Summary Overview page appears. You can now create the RTL-based project.



Creating an MCS File and Programming the Alveo Card

For custom RTL flow, this section outlines the procedures to do the following:

- Create an MCS file (PROM image)
- Flash programming through the USB-JTAG (Micro USB) interface

Create an MCS File (PROM Image)

The Alveo accelerator card contains a Quad SPI configuration flash memory part that can be configured over USB-JTAG. This part contains a protected region, with the factory base image at the `0x00000000` address space. This base image points to the customer programmable region at a `0x01002000` address space offset.

To ensure that the PROM image is successfully loaded onto the Alveo accelerator card at power on, the starting address must be set to `0x01002000` and the interface set to spix4 when creating the MCS file. Details on adding this to the MCS file can be found in the *UltraScale Architecture Configuration User Guide* (UG570).

In addition, the following code must be placed in the project XDC file to correctly configure the MCS file.

```
# Bitstream Configuration
# ----------------------------------------------------------------------
set_property CONFIG_VOLTAGE 1.8                        [current_design]
set_property BITSTREAM.CONFIG.CONFIGFALLBACK Enable    [current_design]
set_property BITSTREAM.GENERAL.COMPRESS TRUE           [current_design]
set_property CONFIG_MODE SPIx4                         [current_design]
set_property BITSTREAM.CONFIG.SPI_BUSWIDTH 4           [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 85.0          [current_design]
set_property BITSTREAM.CONFIG.EXTMASTERCCLK_EN disable [current_design]
set_property BITSTREAM.CONFIG.SPI_FALL_EDGE YES        [current_design]
set_property BITSTREAM.CONFIG.UNUSEDPIN Pullup         [current_design]
set_property BITSTREAM.CONFIG.SPI_32BIT_ADDR Yes       [current_design]
# ----------------------------------------------------------------------
```

Program the Alveo Card

After the MCS file is created, see the procedure in the "Programming the FPGA Device" chapter in the *Vivado Design Suite User Guide: Programming and Debugging* (UG908) to connect to the Alveo Data Center accelerator card using the hardware manager.

1. Select **Add Configuration Device** and select the mt25qu01g-spi-x1_x2_x4 part.
2. Right-click the target to select **Program the Configuration Memory Device**.
   a. Select the MCS file target.
   b. Select **Configuration File Only**.
   c. Click **OK**.
3. After programming has completed, disconnect the card in the hardware manager, and disconnect the USB cable from the Alveo accelerator card.
4. Perform a cold reboot on the host machine to complete the card update.

IMPORTANT! *If you are switching between an Alveo Data Center accelerator card target platform and a custom design, revert the card to the golden image before loading an alternate image into the PROM. See Getting Started with Alveo Data Center Accelerator Cards (UG1301) for more information.*

Xilinx Design Constraints (XDC) File

RTL users can reference the *Vivado Design Suite User Guide: Using Constraints* (UG903) for more information. The Alveo accelerator card XDC files are available for download from their respective websites along with this user guide.

Chapter 4

Card Installation and Configuration

Electrostatic Discharge Caution

CAUTION! *ESD can damage electronic components when they are improperly handled, and can result in total or intermittent failures. Always follow ESD-prevention procedures when removing and replacing components.*

To prevent ESD damage:

- Use an ESD wrist or ankle strap and ensure that it makes skin contact. Connect the equipment end of the strap to an unpainted metal surface on the chassis.
- Avoid touching the adapter against your clothing. The wrist strap protects components from ESD on the body only.
- Handle the adapter by its bracket or edges only. Avoid touching the printed circuit board or the connectors.
- Put the adapter down only on an antistatic surface such as the bag supplied in your kit.
- If you are returning the adapter to Xilinx Product Support, place it back in its antistatic bag immediately.

Installing Alveo Data Center Accelerator Cards in Server Chassis

Because each server or PC vendor's hardware is different, for physical board installation guidance, see the manufacturer's PCI Express® board installation instructions.

For programming and start-up details, see *Getting Started with Alveo Data Center Accelerator Cards* (UG1301).

FPGA Configuration

The Alveo U280 accelerator card supports two UltraScale+™ FPGA configuration modes:

- Quad SPI flash memory
- JTAG using USB JTAG configuration port

The FPGA bank 0 mode pins are hardwired to M[2:0] = `001` master SPI mode with pull-up/down resistors.

At power up, the FPGA is configured by the Quad SPI NOR flash device (Micron MT25QU01GBBA8E12-0SIT) with the FPGA_CCLK operating at clock rate of 105 MHz (EMCCLK) using the master serial configuration mode. The Quad SPI flash memory NOR device has a capacity of 1 Gb.

If the JTAG cable is plugged in, QSPI configuration might not occur. JTAG mode is always available independent of the mode pin settings.

For complete details on configuring the FPGA, see the *UltraScale Architecture Configuration User Guide* (UG570).

Table 7: **Configuration Modes**

Configuration Mode	M[2:0]	Bus Width	CCLK Direction
Master SPI	001	x1, x2, x4	FPGA output
JTAG	Not applicable – JTAG overrides	x1	Not applicable

Card Component Description

This chapter provides a functional description of the components of the Alveo™ U280 Data Center accelerator card.

UltraScale+ FPGA

The Alveo U280 accelerator card is populated with the 16 nm UltraScale+™ XCU280 FPGA. This device incorporates two 4 GB High Bandwidth Memory (HBM) stacks adjacent to the device die. Using SSI technology, the device communicates to the HBM stacks through memory controllers that connect through the silicon interposer at the bottom of the device. Each XCU280 FPGA contains two 4 GB HBM stacks, resulting in up to 8 GB of HBM per device. The device includes 32 HBM AXI interfaces used to communicate with the HBM. The flexible addressing feature that is provided by a built-in switch allows for any of the 32 HBM AXI interfaces to access any memory address on either one or both of the HBM stacks. This flexible connection between the device and the HBM stacks is helpful for floorplanning and timing closure.

Related Information

Known Issues and Limitations

DDR4 DIMM Memory

Two independent dual-rank DDR4 interfaces are available. The card is populated with two socketed single-rank Micron MTA18ASF2G72PZ-2G3B1IG 16 GB DDR4 RDIMMs. Each DDR4 DIMM is 72 bits wide (64 bits plus support for ECC).

The detailed FPGA and DIMM pin connections for the feature described in this section are documented in the Alveo U280 accelerator card XDC file.

For more details about the Micron DDR4 DIMM, see the Micron MTA18ASF2G72PZ-2G3B1IG data sheet at the Micron website: http://www.micron.com.

Related Information

Xilinx Design Constraints (XDC) File

Quad SPI Flash Memory

The Quad SPI device provides 1 Gb of nonvolatile storage.

- Part number: MT25QU01GBBB8E12-0AAT (Micron)
- Supply voltage: 1.8V
- Datapath width: 4 bits
- Data rate: 100 MHz

For more flash memory details, see the Micron MT25QU01GBBB8E12-0AAT data sheet at the Micron website.

For configuration details, see the *UltraScale Architecture Configuration User Guide* (UG570). The detailed FPGA and Flash pin connections for the feature described in this section are documented in the Alveo U280 accelerator card XDC file, referenced in Xilinx Design Constraints (XDC) File.

Related Information
Xilinx Design Constraints (XDC) File

USB JTAG Interface

The Alveo accelerator card provides access to the FPGA device via the JTAG interface.

FPGA configuration is available through the Vivado® hardware manager, which accesses the on-board USB-to-JTAG FT4232HQ bridge device. The micro-AB USB connector on the Alveo U280 accelerator card PCIe® panel/bracket provides external device programming access.

Note: JTAG configuration is allowed at any time regardless of the FPGA mode pin settings consistent with the *UltraScale Architecture Configuration User Guide* (UG570).

For more details about the FT4232HQ device, see the FTDI website: https://www.ftdichip.com/.

FT4232HQ USB-UART Interface

The FT4232HQ Quad USB-UART provides a UART connection through the micro-AB USB connector. Channel BD implements a 2-wire level-shifted TX/RX UART connection to the FPGA. The FTDI FT4232HQ data sheet is available on the FTDI website: https://www.ftdichip.com/.

PCI Express Endpoint

The Alveo U280 accelerator card implements a 16-lane PCI Express® edge connector that performs data transfers at the rate of 2.5 giga-transfers per second (GT/s) for Gen1, 5.0 GT/s for Gen2, 8.0 GT/s for Gen3 applications, and 16.0 GT/s for Gen4 applications.

The detailed FPGA and PCIe pin connections for the feature described in this section are documented in the accelerator card design constraints (XDC) file. The endpoint is compliant to the v3.0 specification, and compatible with the specification (specifications are available here). For additional information on Gen4 compatible features, see *UltraScale+ Devices Integrated Block for PCI Express LogiCORE IP Product Guide* (PG213).

Related Information

Xilinx Design Constraints (XDC) File

QSFP28 Module Connectors

The Alveo accelerator cards host two 4-lane small form-factor pluggable (QSFP) connectors that accept an array of optical modules. Each connector is housed within a single QSFP cage assembly.

The QSFP+ connectors are accessible via the I2C interface on the Alveo U280 accelerator cards. The QSFP connector's sideband signals are accessible directly from the FPGA. The MODSELL, RESETL, MODPRSL, INTL, and LPMODE sideband signals are defined in the small form factor (SFF) specifications listed below. The components visible through the card PCIe panel/bracket top to bottom are:

- DONE, POWER GOOD, and 2x status LEDs
- QSFP0
- QSFP1
- USB

For additional information about the quad SFF pluggable (28 Gb/s QSFP+) module, see the SFF-8663 and SFF-8679 specifications for the 28 Gb/s QSFP+ at the SNIA Technology Affiliates website: https://www.snia.org/sff/specifications2.

Each QSFP connector has its own clock generator.

- QSFP0 clock
  - Clock generator: Silicon Labs SI5335A-B06201-GM

  - Output CLK1A/1B: the QSFP0_CLOCK_P/N clock is an AC-coupled LVDS 156.25 MHz clock wired to the QSFP0 GTY interface

- QSFP1 clock
  - Clock generator: Silicon Labs SI5335A-B06201-GM
  - Output CLK1A/1B: the QSFP1_CLOCK_P/N clock is an AC-coupled LVDS 156.25 MHz clock wired to the QSFP1 GTY interface

The detailed FPGA and QSFP pin connections for the feature described in this section are documented in the Xilinx Design Constraints (XDC) File.

Related Information
Xilinx Design Constraints (XDC) File

I2C Bus

The Alveo U280 accelerator cards implement an I2C bus network.

Card Power System

Limited power system telemetry is available through the I2C IP. I2C IP is instantiated during the FPGA design process which begins after the Alveo Data Center accelerator card is selected from the Vivado Design Suite Boards tab. Refer to Design Flows for more information.

Clocks

The FPGA reference clocks are supplied by a local oscillator and the PCIe edge connector. The local oscillator clock is distributed to four banks, providing that a 300 MHz reference clock is used. The DDR4 memories are recommended for driving FPGA system logic. The PCIe common clock is distributed to two banks providing 100 MHz providing the reference to the PCIe transceivers. The detailed FPGA and clock pin connections for the feature described in this section are documented in the U280 accelerator card XDC file.

Related Information
Xilinx Design Constraints (XDC) File

Chapter 6

Known Issues and Limitations

AR 71752 is the master answer record for the Alveo Data Center accelerator cards. For Technical Support, open a Support Service Request.

XILINX

Appendix A

Regulatory and Compliance Information

This product is designed and tested to conform to the European Union directives and standards described in this section.

CE Directives

2014/35/EC, *Low Voltage Directive (LVD)*

2014/30/EC, *Electromagnetic Compatibility (EMC) Directive*

CE Standards

EN standards are maintained by the European Committee for Electrotechnical Standardization (CENELEC). IEC standards are maintained by the International Electrotechnical Commission (IEC).

Electromagnetic Compatibility

EN:55032:2015, *Information Technology Equipment Radio Disturbance Characteristics – Limits and Methods of Measurement*

EN:55024:2015, *Information Technology Equipment Immunity Characteristics – Limits and Methods of Measurement*

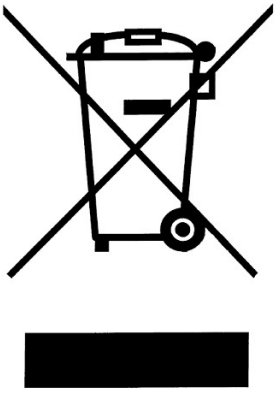
This is a Class A product. In a domestic environment, this product can cause radio interference, in which case the user might be required to take adequate measures.

Safety

IEC 60950-1, 2nd Edition, 2014, *Information technology equipment – Safety, Part 1: General requirements*

EN 60950-1, 2nd Edition, 2014, *Information technology equipment – Safety, Part 1: General requirements*

Compliance Markings



In August of 2005, the European Union (EU) implemented the EU Waste Electrical and Electronic Equipment (WEEE) Directive 2002/96/EC and later the WEEE Recast Directive 2012/19/EU. These directives require Producers of electronic and electrical equipment (EEE) to manage and finance the collection, reuse, recycling and to appropriately treat WEEE that the Producer places on the EU market after August 13, 2005. The goal of this directive is to minimize the volume of electrical and electronic waste disposal and to encourage re-use and recycling at the end of life.

Xilinx has met its national obligations to the EU WEEE Directive by registering in those countries to which Xilinx is an importer. Xilinx has also elected to join WEEE Compliance Schemes in some countries to help manage customer returns at end-of-life.

If you have purchased Xilinx-branded electrical or electronic products in the EU and are intending to discard these products at the end of their useful life, please do not dispose of them with your other household or municipal waste. Xilinx has labeled its branded electronic products with the WEEE Symbol to alert our customers that products bearing this label should not be disposed of in a landfill or with municipal or household waste in the EU.

This product complies with Directive 2002/95/EC on the restriction of hazardous substances (RoHS) in electrical and electronic equipment.

This product complies with CE Directives 2006/95/EC, *Low Voltage Directive (LVD)* and 2004/108/EC, *Electromagnetic Compatibility (EMC) Directive*.

Appendix B

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

Note: For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

References

These documents provide supplemental material useful with this guide:

Product Websites

The most up-to-date information related to the Alveo™ U280 card and documentation is available on the Alveo U280 Data Center Accelerator Card.

Supplemental Documents

The following Xilinx document provide supplemental material useful with this guide.

- *AXI High Bandwidth Controller LogiCORE IP Product Guide* (PG276)
- *UltraFast Design Methodology Guide for the Vivado Design Suite* (UG949)
- *Vivado Design Suite User Guide: System-Level Design Entry* (UG895)
- *Getting Started with Alveo Data Center Accelerator Cards* (UG1301)
- *Alveo Programming Cable User Guide* (UG1377)
- *UltraScale Architecture Configuration User Guide* (UG570)
- *Vivado Design Suite User Guide: Programming and Debugging* (UG908)
- *Virtex UltraScale+ FPGA Data Sheet: DC and AC Switching Characteristics* (DS923)
- *UltraScale Architecture-Based FPGAs Memory IP LogiCORE IP Product Guide* (PG150)
- *Vivado Design Suite User Guide: Using Constraints* (UG903)
- *UltraScale Architecture PCB Design User Guide* (UG583)

Additional Links

The following links provide supplemental material useful with this guide.

- Xilinx, Inc: https://www.xilinx.com
- Micron Technology: http://www.micron.com

  (MTA18ASF2G72PZ-2G3B1IG, MT25QU01GBB8E12-OAAT)
- *Si5394 Data Sheet*: https://www.silabs.com/documents/public/data-sheets/si5395-94-92-a-datasheet.pdf
- Future Technology Devices International, Ltd.: http://www.ftdichip.com

  (FT4232HQ)
- SFP-DD module: SFP-DD Specification

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