PCI Express Data Streaming Directly into the GPU

Thomas Zerrer

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

History:

- Founded in 2005
- Office in Hildrizhausen, near Stuttgart
- Focus on PCIe DMA IP cores
- Partner of Lattice, AMD, and Altera

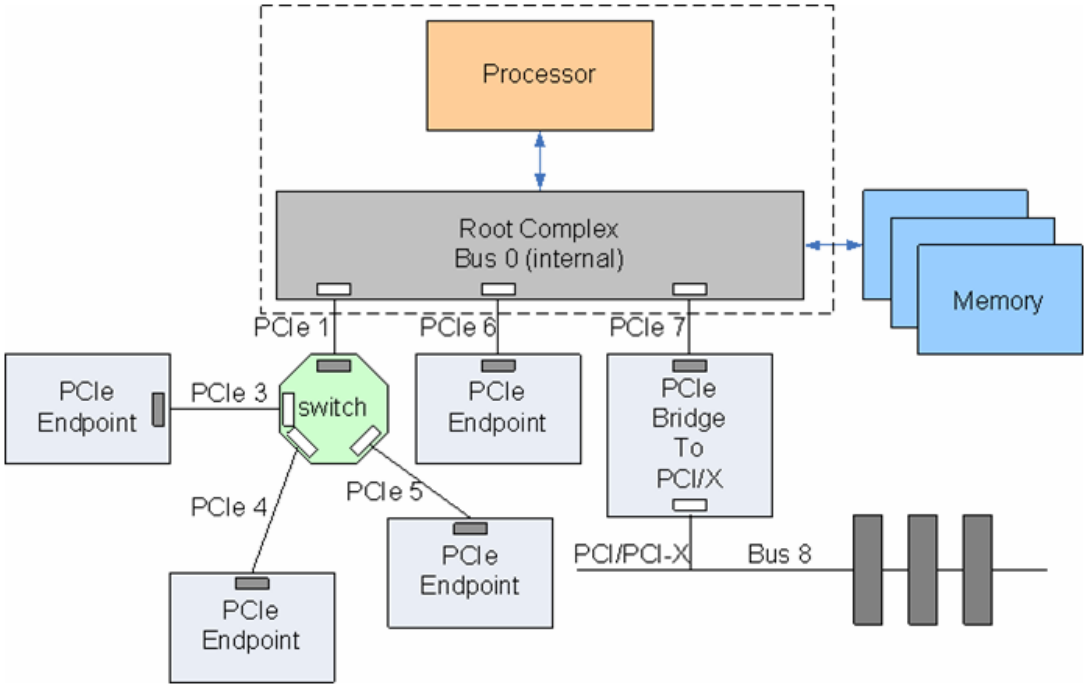
© 2005-2026 Smartlogic GmbH

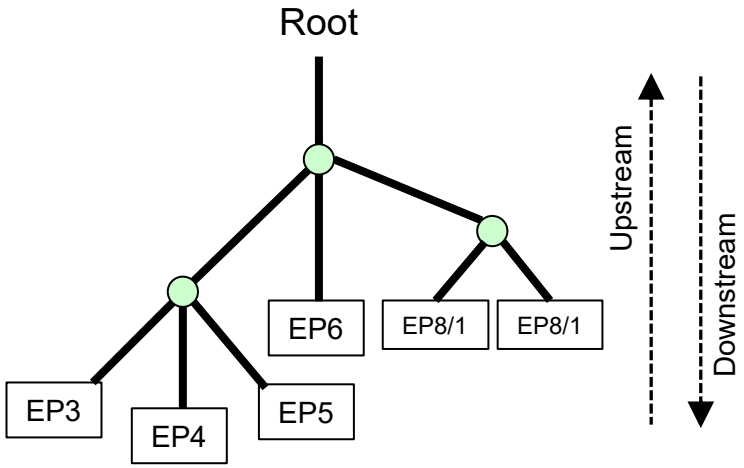
Objectives:

- Understand the structure and basic terms of a PCI Express system
- Peer-to-Peer access (P2P)
- Learn how PCI Express can be used to utilize high-performance and low latency GPU applications

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Typical PCI Express Architecture





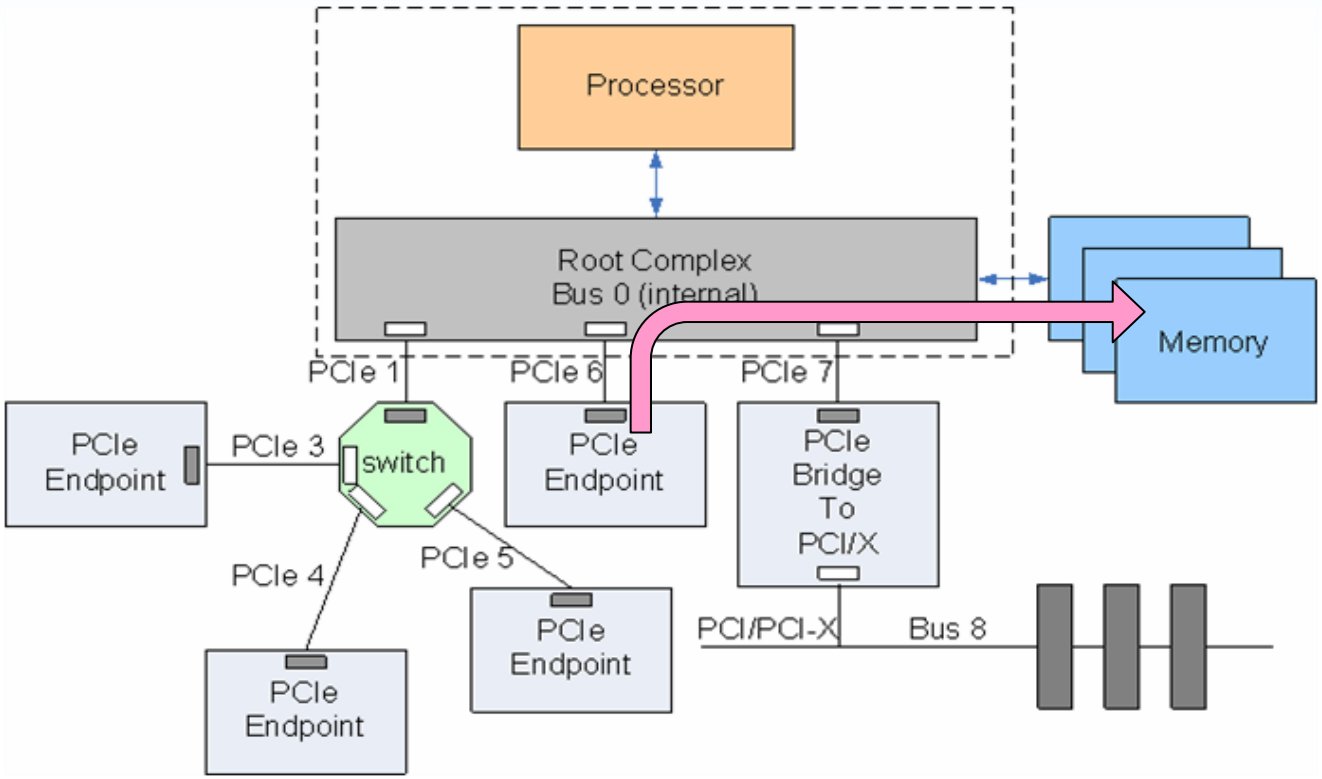
Features:

- PCI Express is always a point-to-point connection
- Switches allow several endpoints to be accessed through a single PCIe connection (PCIe1)
- Every endpoint can exchange data with the Root Complex **or** other PCIe endpoints

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

PCI Express: Bus Mastering

SMARTLOGIC

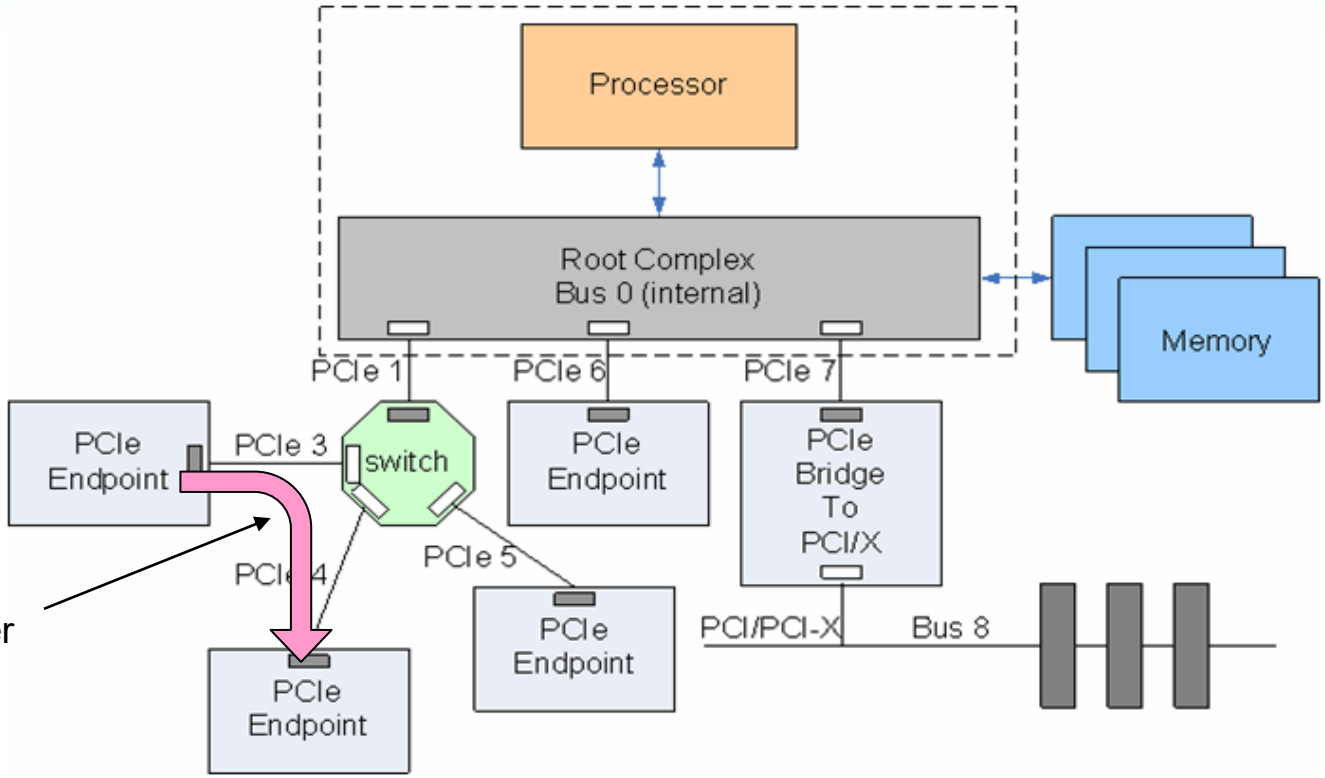


Bus Mastering Features:

- Every endpoint can act as a bus master and read or write data
- No processor involvement required for such direct accesses
- Data transfer can be to the Root Complex (e.g. direct memory access) or to any PCIe endpoint

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

PCI Express: Peer-to-Peer Transfers



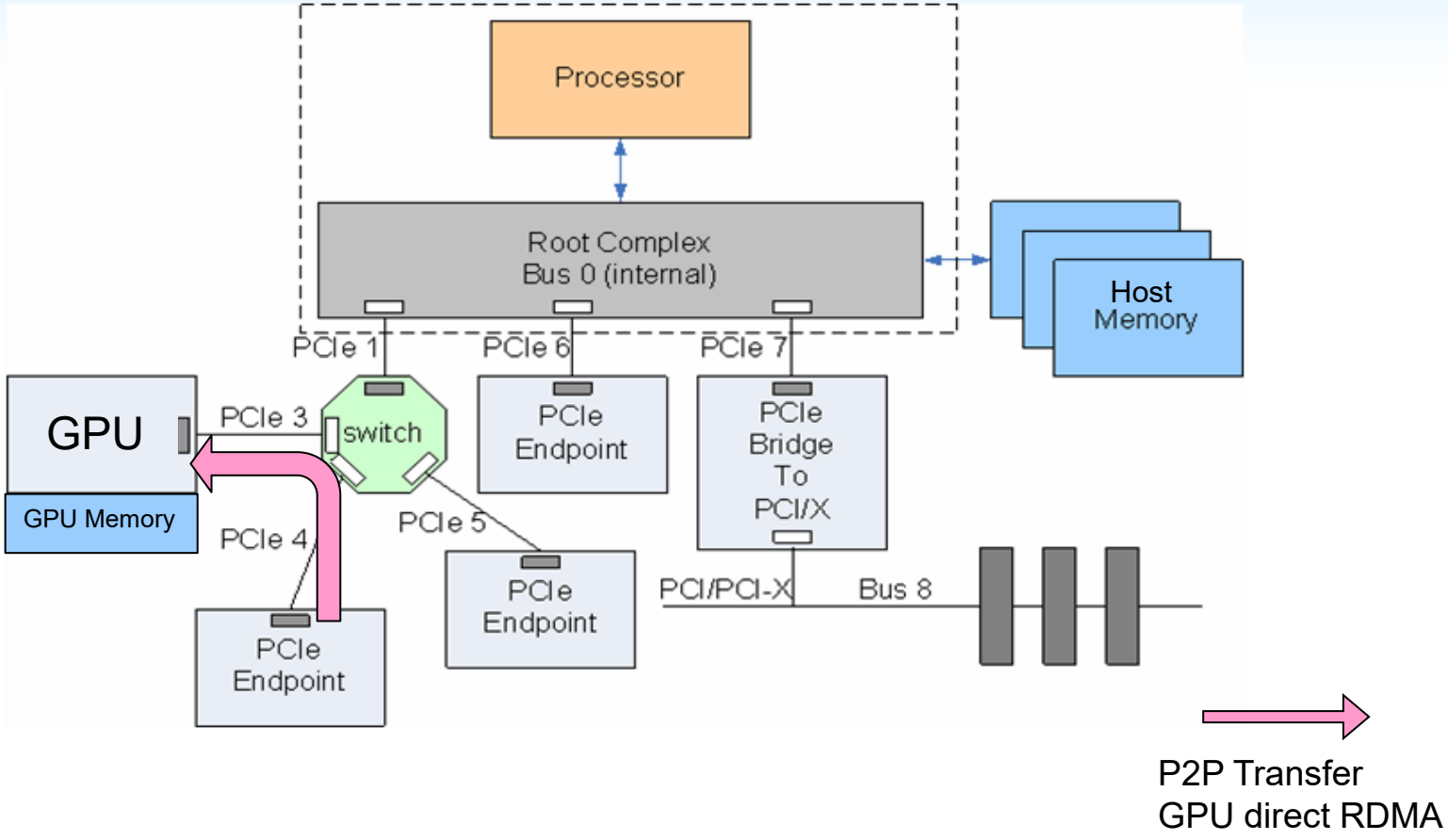
If an endpoint transfers data to another endpoint, this is called a peer-to-peer (P2P) transfer:

- Packets are routed based on the destination address in the packet header
- P2P requires that both endpoints are connected to the same switch
- P2P transfers can be reads or writes
- P2P via the root complex is supported by most chipsets, carefully check that this works

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

PCI Express P2P Example: GPU

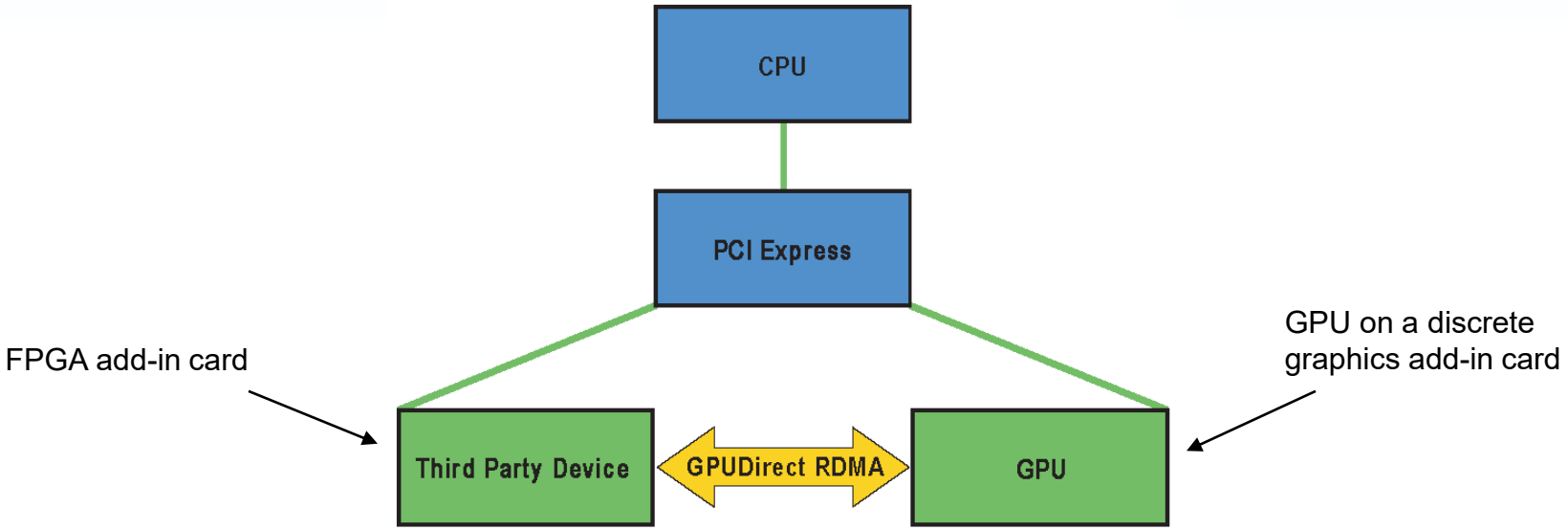
SMARTLOGIC



- The GPU can be accessed using P2P transfers
- The DMA engine kernel-mode driver needs access to the physical PCIe addresses, which have to be provided by the GPU driver
- Carefully select the graphics card based on the capabilities of its device driver

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

BIOS Preparation – required in case of a discrete GPU



If the GPU is located on a discrete graphics add-in card, the following BIOS settings are recommended:

- Enable Above 4G Decoding (enable 64-bit BARs)
- Enable RBAR support (RBAR = resizable BAR)

Both settings ensure that BAR1 (i.e. the visible amount of VRAM on BAR1) is significantly larger than 256 MB. Typically several gigabytes.

Note: NVIDIA calls peer-to-peer transfers to its GPUs „GPUDirect RDMA“

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

IOMMU – Pass-through

NVIDIA specifies that, when an IOMMU is present, address translation should be disabled

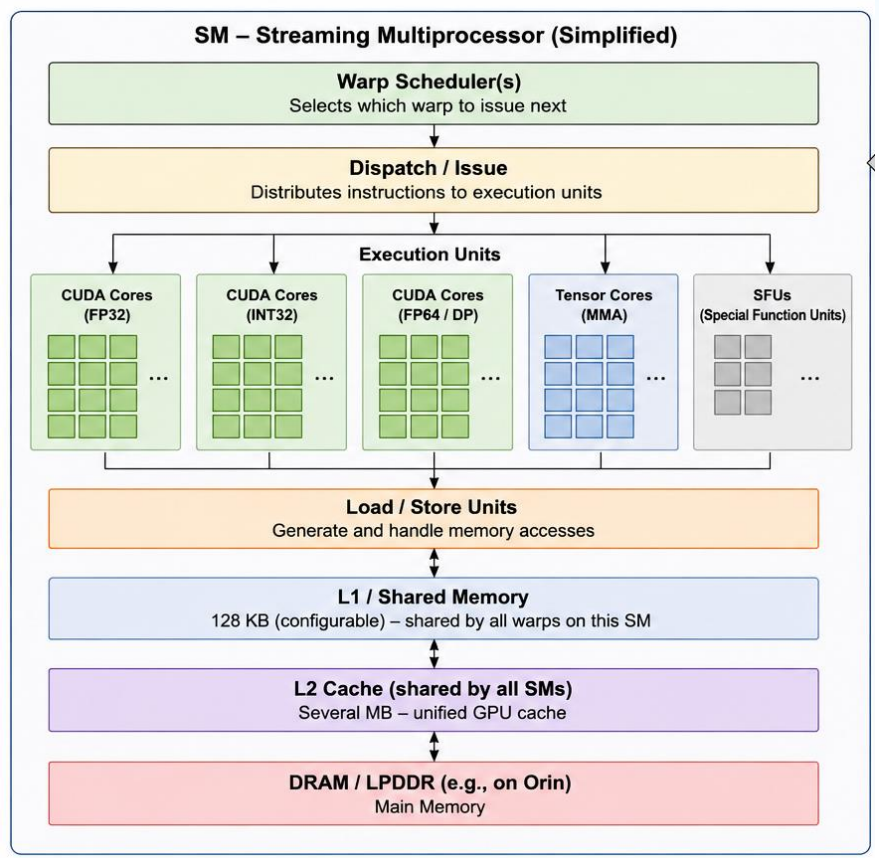
- This is done by setting the IOMMU to „pass-through“.
- Pass-through is configured in GRUB with the kernel boot parameter iommu=pt in /etc/default/grub
- Don‘t forget to issue sudo update-grub and a reboot of the system

This setting is required for discrete and internal GPUs

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Simplified Block Diagram of an NVIDIA GPU

SMARTLOGIC



Example Orin nano 4 GB:

SMs	INT32*	FP32*	FP64*	Tensor*	SFU*
4	64	512	64	16	16

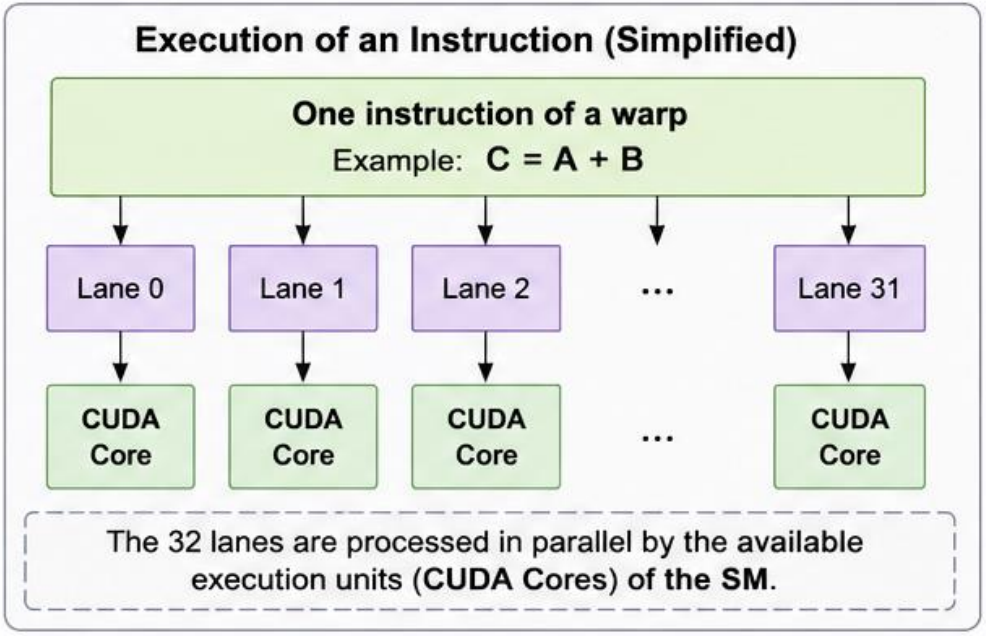
* per SM

- Each GPU has several Streaming Multiprocessors (SMs)
- CUDA Cores within the SM perform arithmetic operations of the active threads (lanes)
- An SM contains several types of specialized execution units, including FP32, FP64, INT32, Tensor Cores, load/store units, and SFUs
- Memory accesses go through the load / store units into the cache hierarchy (L1 -> L2 -> DRAM)

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Simplified Block Diagram of an NVIDIA GPU

SMARTLOGIC

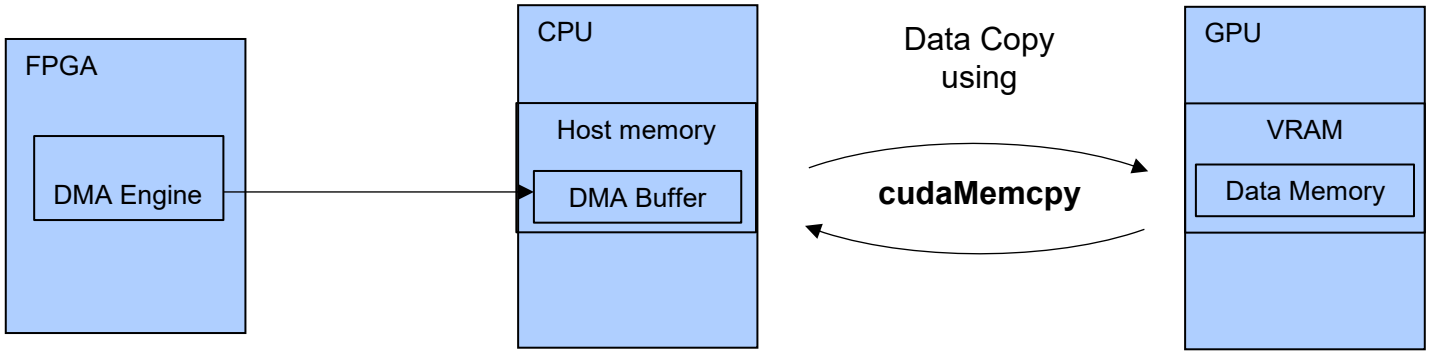


- A warp is the basic scheduling and execution unit and consists of 32 lanes
- SIMT : **S**ingle **I**nstruction **M**ultiple **L**anes :
  Each thread on each lane executes the same instruction to process vectors
- Multiple warps, each consisting of 32 lanes, reside on one SM
- CUDA cores perform arithmetic operations of the active threads (lanes)

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Non-optimized GPU Application : cudaMemcpy

SMARTLOGIC

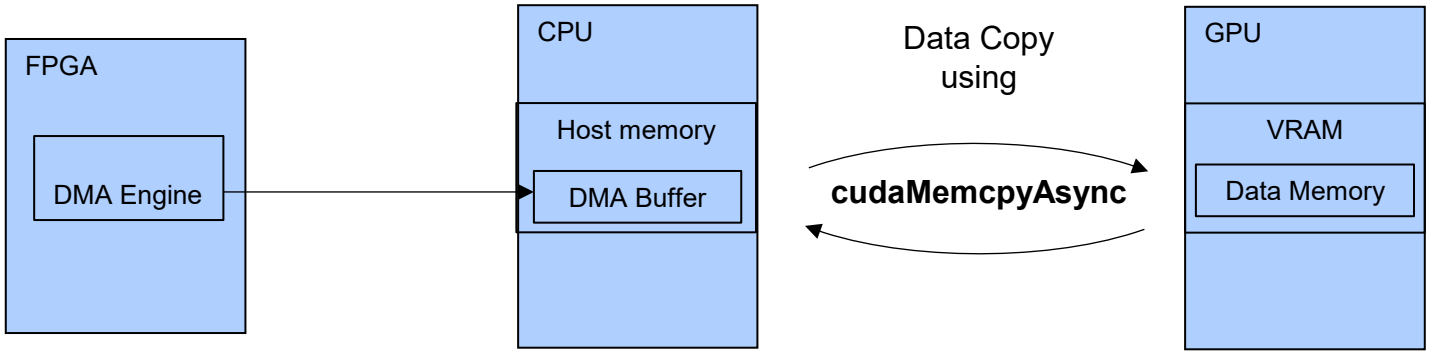


- Disadvantage: Data must be copied over PCIe, consuming PCIe and memory bandwidth
- Disadvantage: cudaMemcpy returns, when the transfer has finished
- Latency is introduced by the copy process, which might not be tolerable for some applications

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Non-optimized GPU Application : cudaMemcpyAsync

SMARTLOGIC

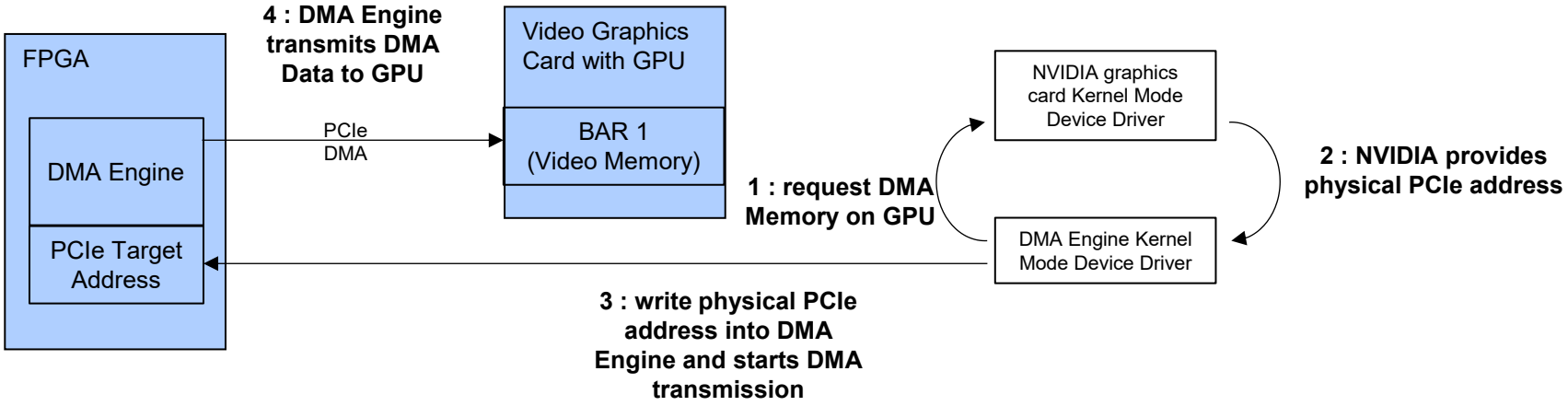


- cudaMemcpyAsync uses the DMA copy engines of the GPU under certain conditions
- If copy engines can be used, cudaMemcpyAsync returns before the transfer has finished
- Challenge: DMA buffer must have the right format (page-locked / pinned) for cudaMemcpyAsync
  i.e. the DMA engine and its kernel-mode driver must be compatible with CUDA
  If there is an incompatibility, an internal staging buffer is introduced and a further CPU copy is introduced.

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Case 1: Optimized GPU application with discrete GPU (dGPU)

SMARTLOGIC

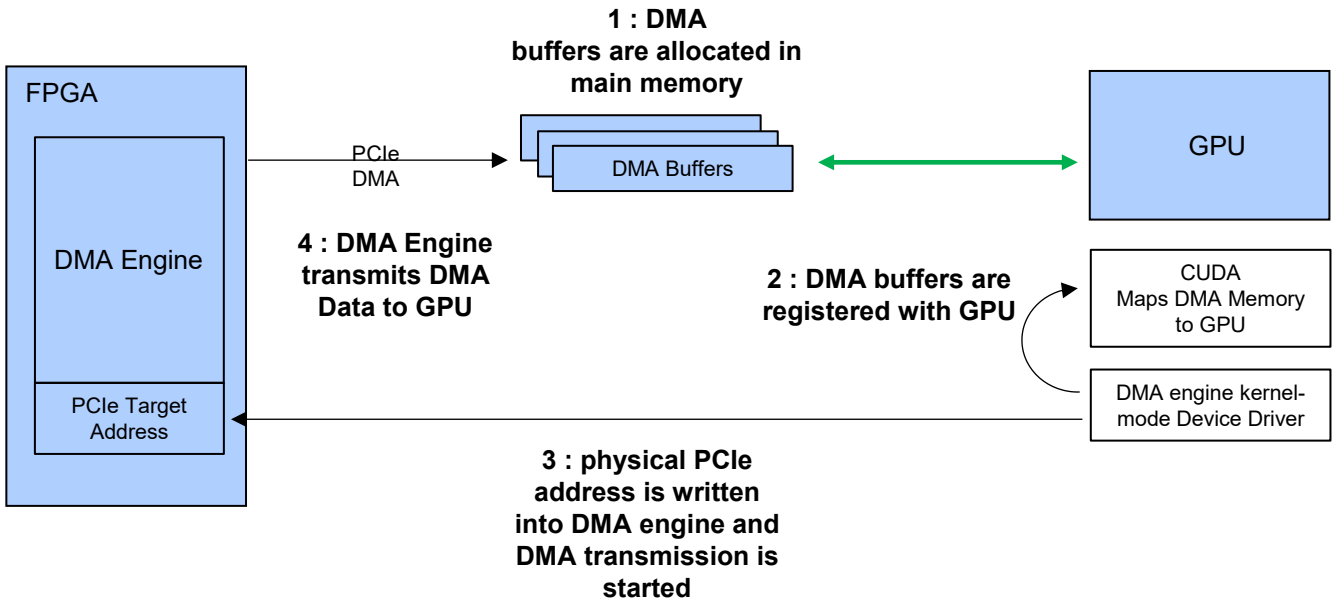


- DMA memory is part of the graphics card VRAM (i.e. it is not located in the host memory)
- DMA memory on GPU must be allocated via the graphics card device driver
  Never write DMA data to unreserved memory!
- Only the graphics card device driver can provide the physical PCIe destination address!
  The device driver of the DMA engine must be NVIDIA compatible

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Case 2: Optimized GPU application with integrated GPU (iGPU)

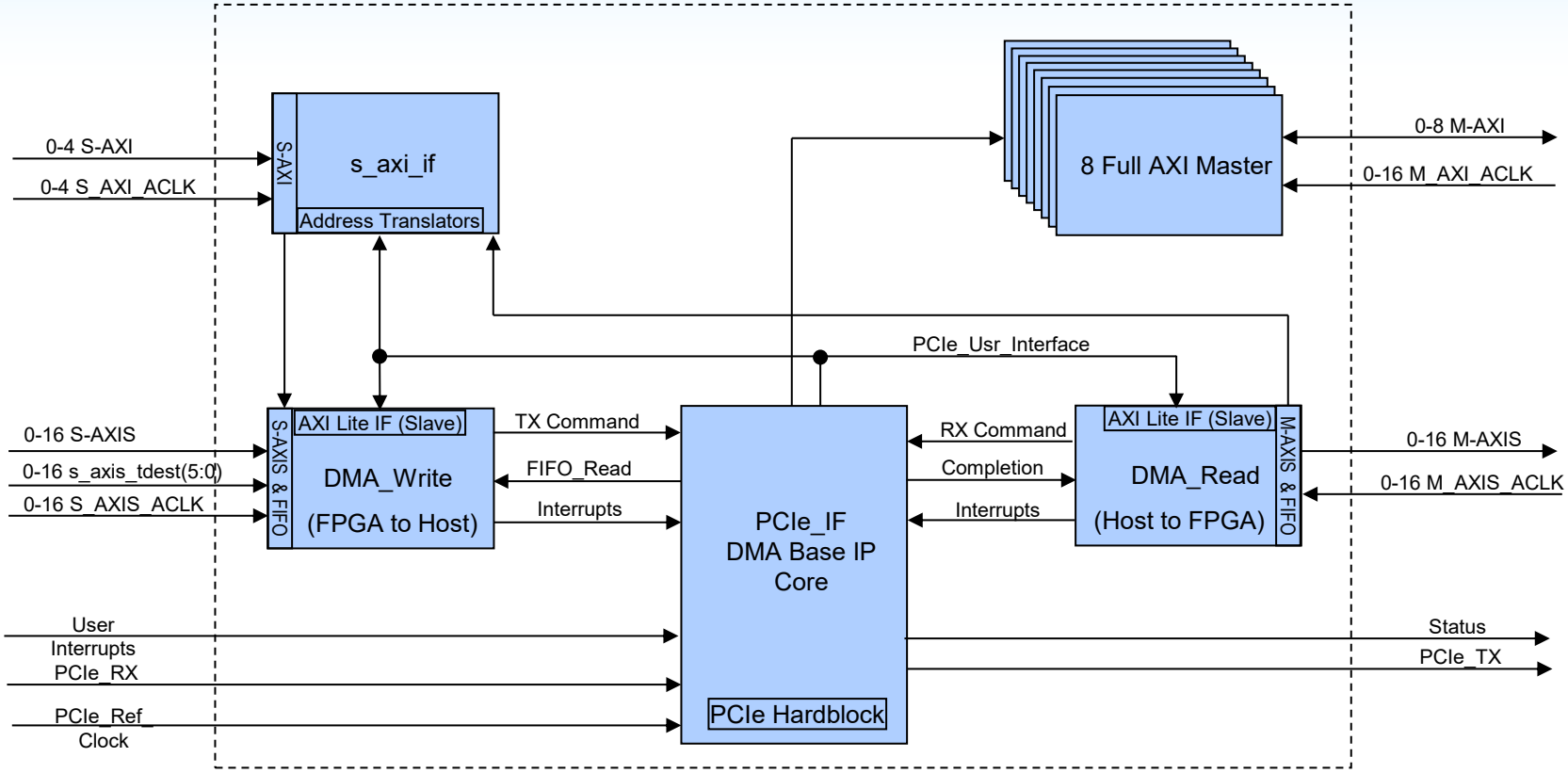
SMARTLOGIC



- In this case GPU memory and CPU memory are shared: DMA buffers are part of the main memory
- DMA buffers must be allocated by the DMA engine kernel-mode driver via Linux
- DMA buffers must pinned and registered with CUDA : Device drivers must be compatible
- Typical example of this platform type is NVIDIA Orin

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential

Smartlogic Multi-channel PCI Express DMA IP Core



Highlights:

- Smallest logic footprint in the industry, especially designed for low-density FPGA families with PCIe
- Multi-channel architecture: All interfaces can be used in parallel, non-blocking
- Kernel Mode Driver is NVIDIA compatible, NVIDIA GPU reference design available
- Designed for lowest latency and highest performance

Copyright Smartlogic 2005-2026, All Rights reserved. Confidential