OpenNet

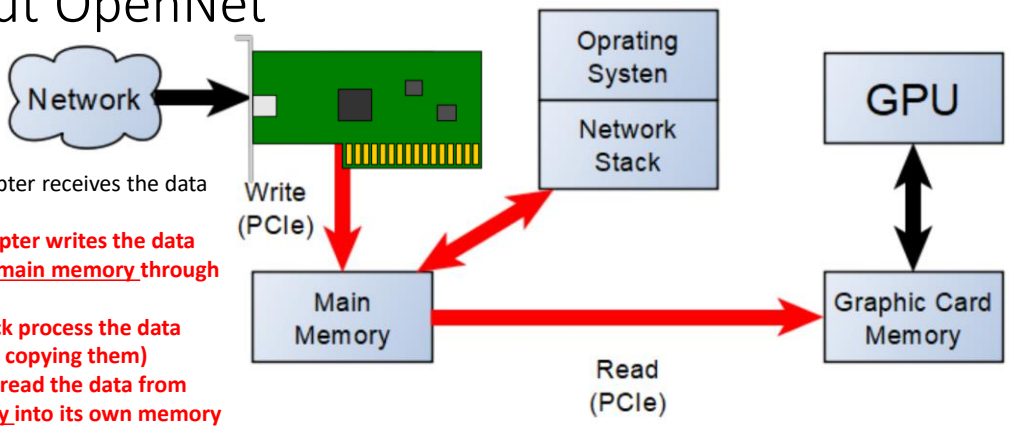
by kms

Martin Dubois, ing

Contents

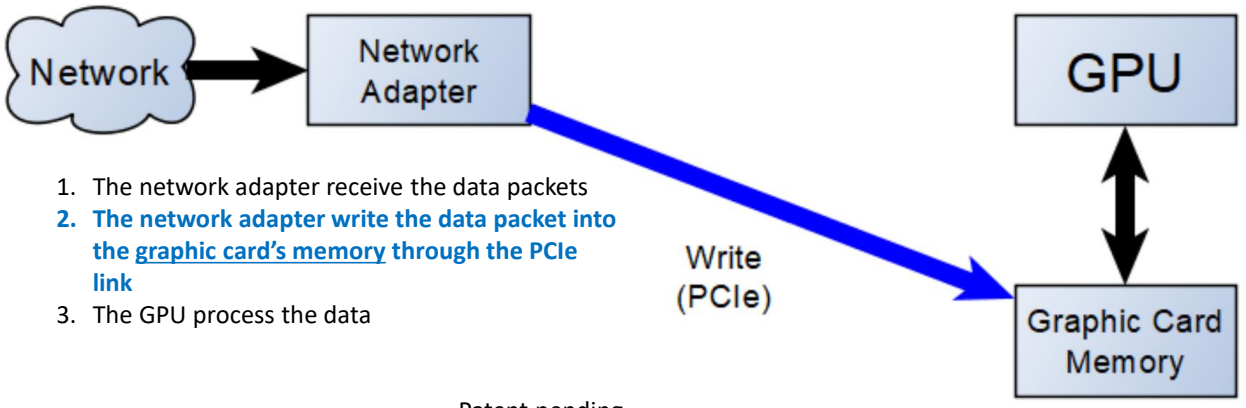
- Without OpenNet vs With OpenNet
- Technical information
- Possible applications
  - **A**rtificial **I**ntelligence
  - **D**eep **P**acket **I**nspection
  - Image and Video processing
  - Network equipment development and validation
  - …

Without OpenNet



1. The network adapter receives the data packets
2. **The network adapter writes the data packets into the main memory through the PCIe link**
3. **The network stack process the data packets (possibly copying them)**
4. **The graphic card read the data from the main memory into its own memory**
5. The GPU process the data

With OpenNet



1. The network adapter receive the data packets
2. **The network adapter write the data packet into the graphic card's memory through the PCIe link**
3. The GPU process the data

Patent pending

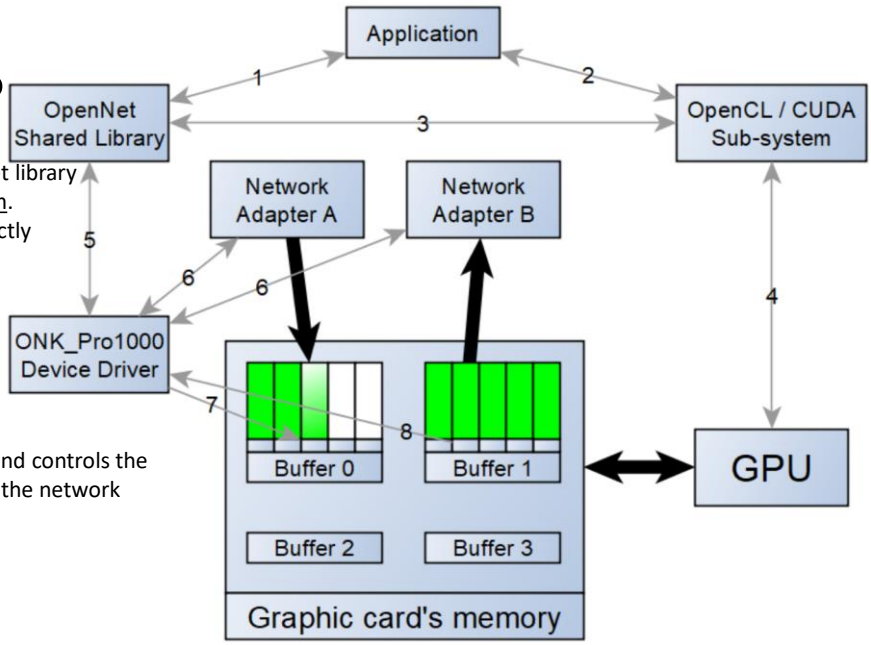
Receive and Send

- Once the GPU processed a data packet, and possibly modify it, it can
  - Forward it to the operating system network stack
  - Forward it using one of network adapter connected to the OpenNet system (including the one that received the data packet)
  - Drop it!

When the packet is forwarded using a network adapter, the network adapter read it directly from the graphic card's memory!

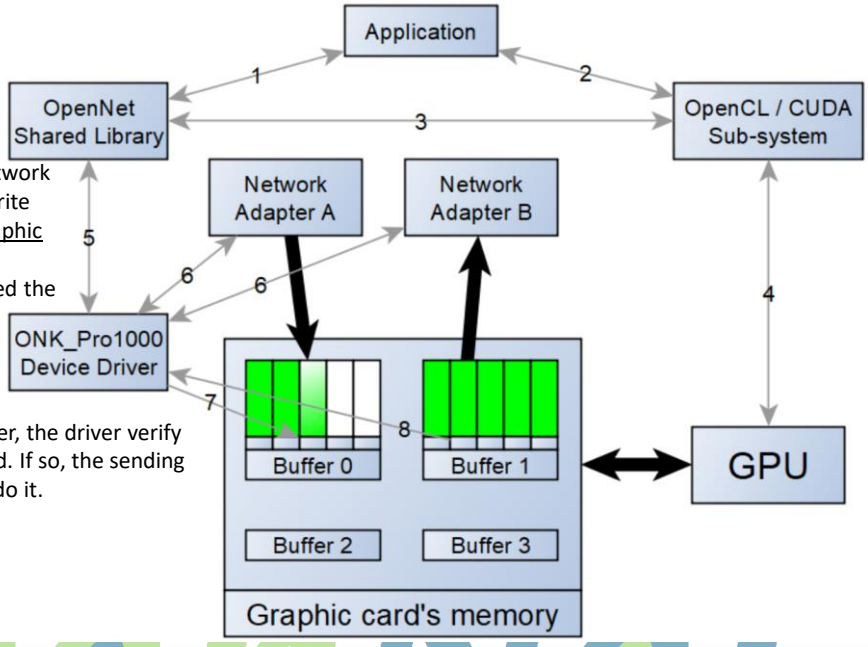
How it work?

1. The application uses the OpenNet library to configure the OpenNet system.
2. The application also interact directly with the OpenCL™ or CUDA® environment.
3. The OpenNet library controls the execution of kernels processing data packets
4. The OpenCL™ or CUDA® environment controls the GPU.
5. The OpenNet library configures and controls the special device drivers controlling the network adapters.



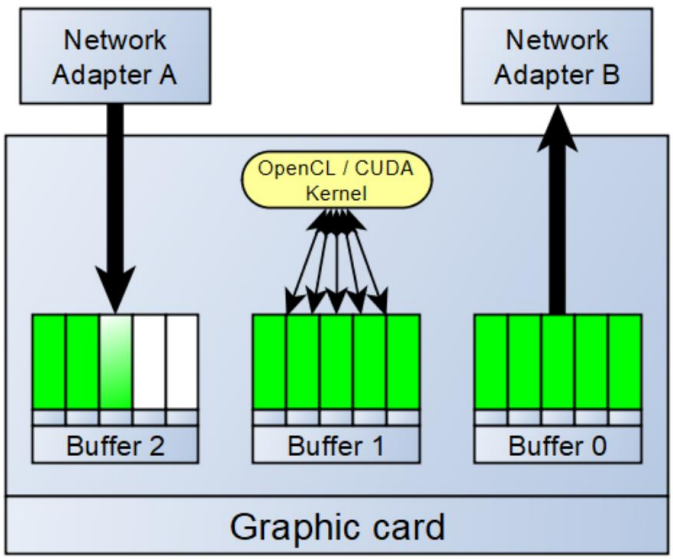
How it work?

6. The device driver controls the network adapters and instructs them to write received data directly into the graphic card's memory.
7. After the network adapter received the data, the driver writes the packet size into the graphic card's memory. When the buffer is full, the kernel processing it is launch.
8. After the GPU processed the buffer, the driver verify if the packets must be transmitted. If so, the sending network adapter is instructed to do it.



Little bit more details

1. The network adapter A receives data packets and write them into a buffer.
2. When the buffer is full, OpenNet launch the OpenCL™ or CUDA® kernel.
3. The kernel process all the data packets at the same time, possibly modifies them and indicate if the packets must be retransmitted. The kernel can uses other data buffers or interacts with other kernels.
4. Network adapter B (or any other network adapter in the OpenNet system) retransmit the data packets as requested.



Technical information

Operating systems

- Windows® 10
  - 64 bits
- Linux
  - 64 bits
  - Ubuntu 18.04

OpenNet

GPU programming language

Windows

- OpenCL™

Linux

- CUDA®

Graphic cards

Windows

- All cards supporting the AMD® DirectGMA™ technology
- Radeon™ Pro
  - WX5100, WX7100, WX8200 and WX9100

Linux

- All cards supporting the NVIDIA GPUDirect® technology
- Quadro™
- Tesla™

Network adapters

- Intel®
  - 1 Gb/s – 82576
  - 10 Gb/s – 82599

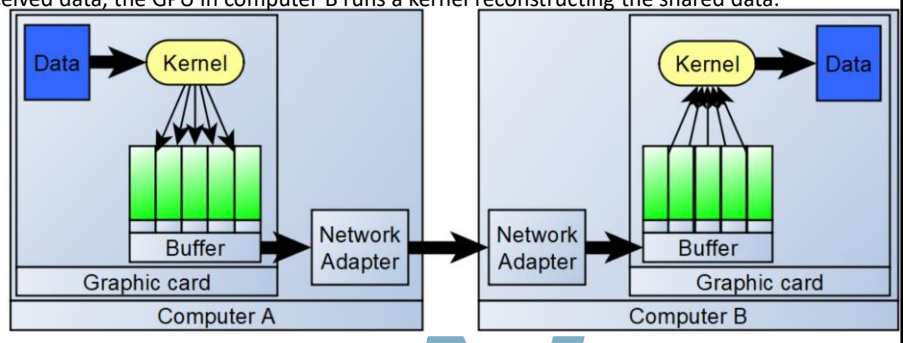
Possible applications

Artificial Intelligence

- Feed training data or data to process
- Share large amount of data between graphic cards installed in different computers

Share large amount of data between graphic cards

- When buffer is ready to be filled, the GPU in computer A runs a kernel that simply break shared data in packets and indicate which packets must be sent.
- When a buffer is filled with received data, the GPU in computer B runs a kernel reconstructing the shared data.
- The communication protocol is completely implemented in OpenCL™ or CUDA®. It can operate on top of:
  - Ethernet for efficiency
  - IP to be routable
  - UDP to be router friendly



Deep Packet Inspection

- **D**eep **P**acket **I**nspection
- Filtering and selecting

Filtering and selecting

- The OpenCL™ or CUDA® code inspects each data packet and decide which equipment or computer will do the final inspection or processing.
- The OpenCL™ or CUDA® code can access the full data packet.
- The next computer in the chain can also use OpenNet and GPU.

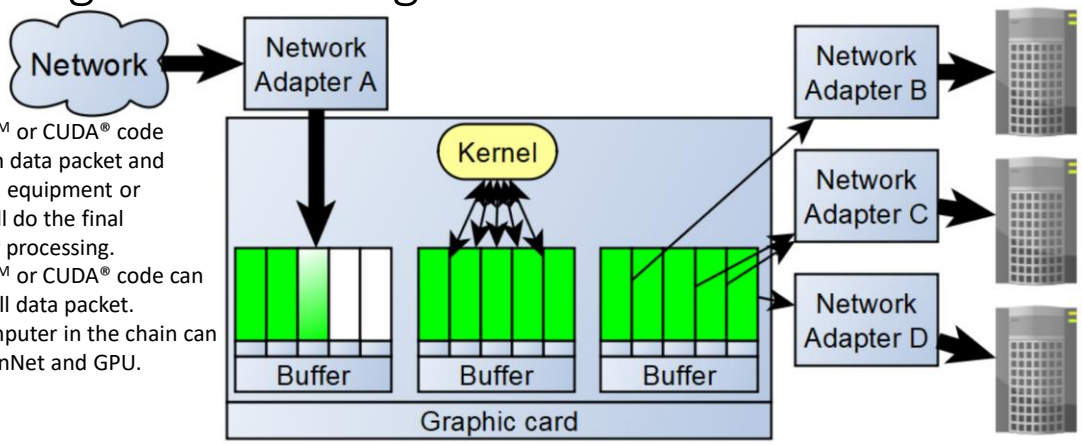
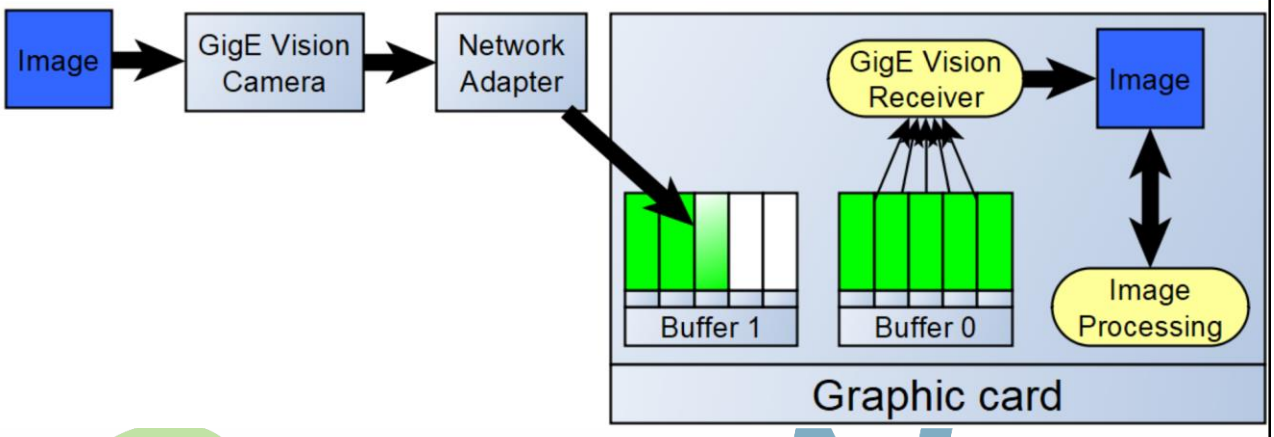


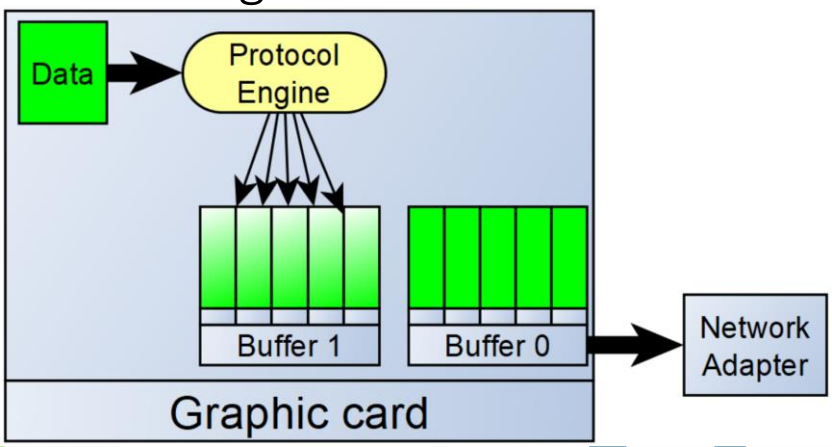
Image and Video processing

- GigE Vision receiver
- Video streaming

GigE Vision Receiver



Video streaming



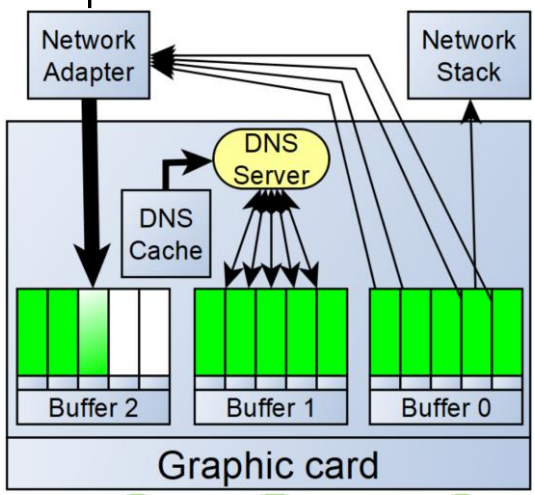
Network equipment development and validation

- Test of OpenCL™ code aimed to FPGA
- Traffic verification
- Traffic generation

Other possible applications

- Service implementation
- TCP or other protocols offload

Service implementation – DNS



The end !

Many other applications are possible!

Any question?
Thanks!

http://www.kms-quebec.com/en/opennet.htm