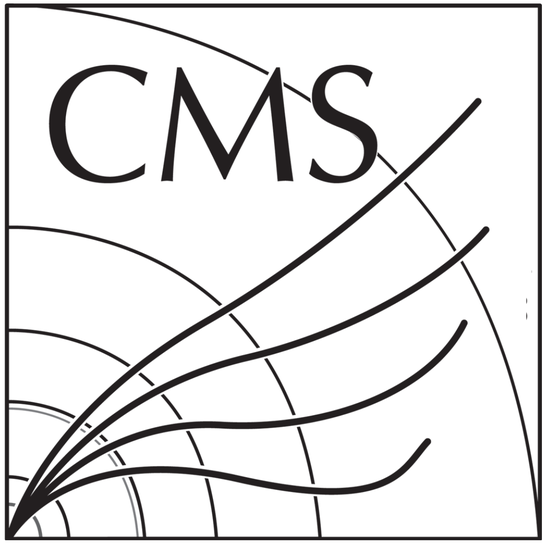
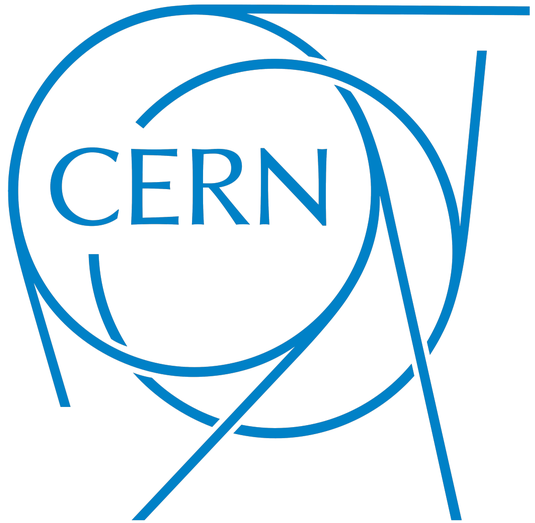
# FPGA Hardware

**CERN - Zenuity Meeting 8/10/19**
**Sioni Summers**

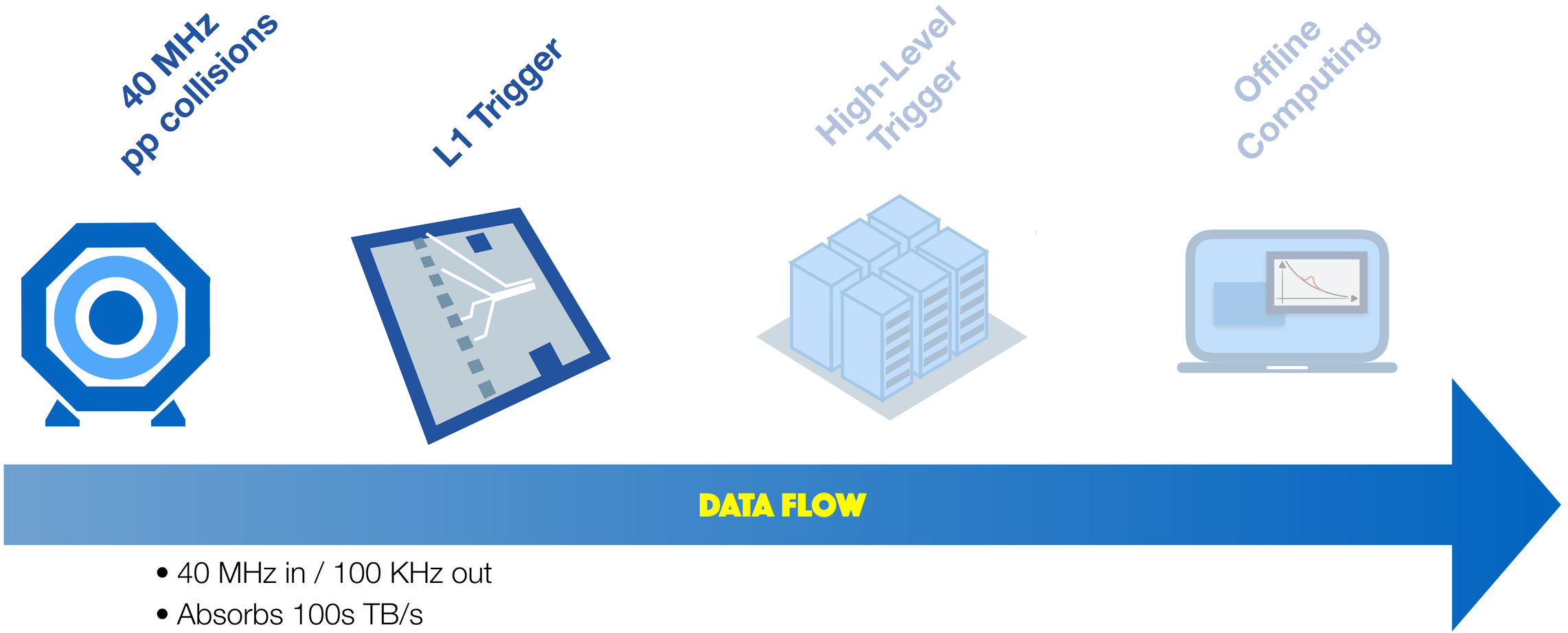


# Contents & Introduction

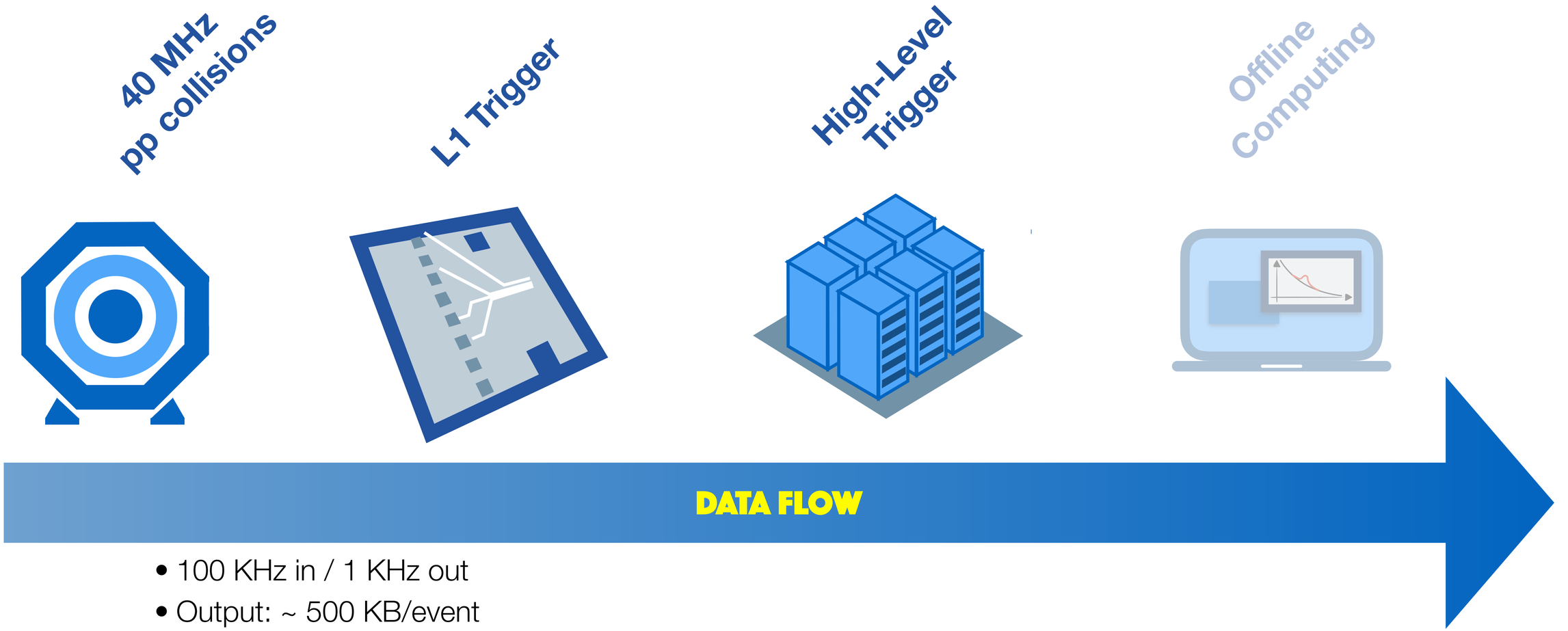
- Real time processing at the LHC
- Introduction to FPGAs
- FPGA boards

# The LHC big data problem



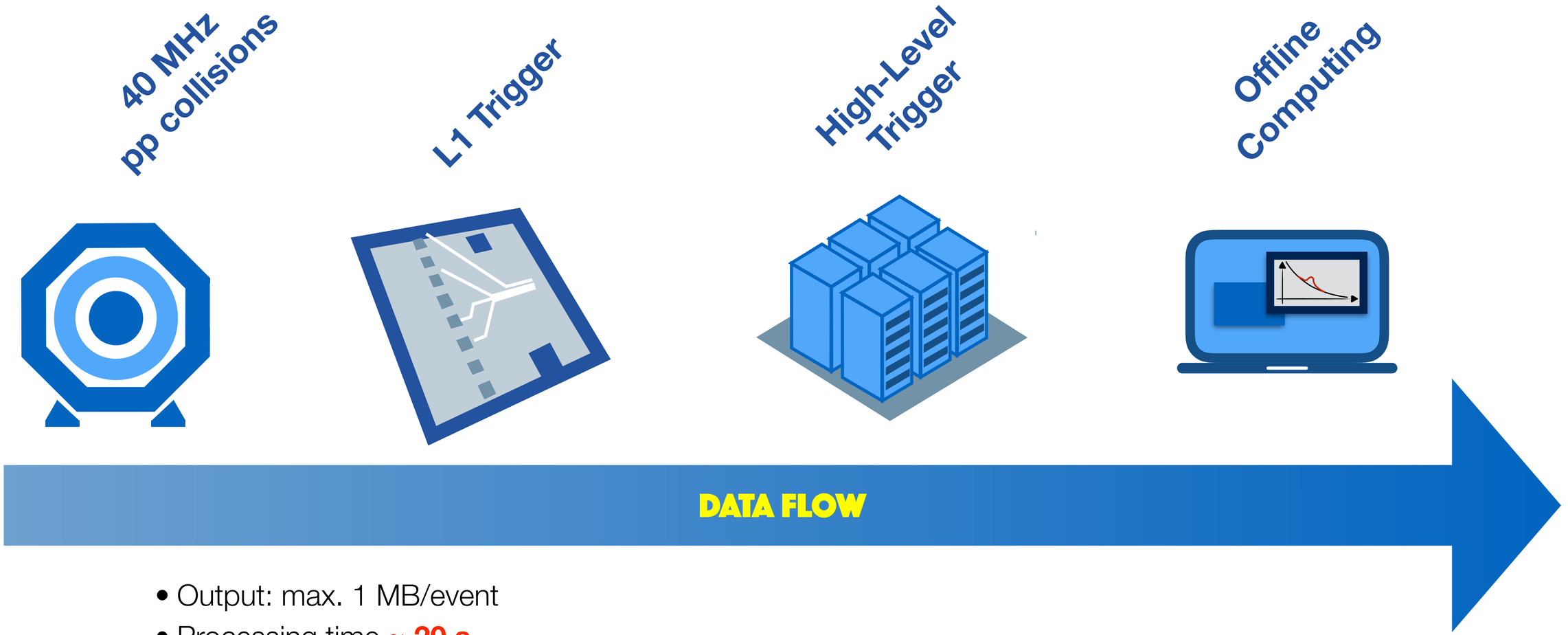

- 40 MHz in / 100 KHz out
- Absorbs 100s TB/s
- Trigger decision to be made in **~ 10 µs**
- Coarse local reconstruction
- FPGAs / Hardware implemented

# The LHC big data problem




- 100 KHz in / 1 KHz out
- Output: ~ 500 KB/event
- Processing time **~ 300 ms**
- Simplified global reconstruction
- Software implemented on CPUs

# The LHC big data problem




- Output: max. 1 MB/event
- Processing time **~ 20 s**
- Accurate global reconstruction
- Software implemented on CPUs

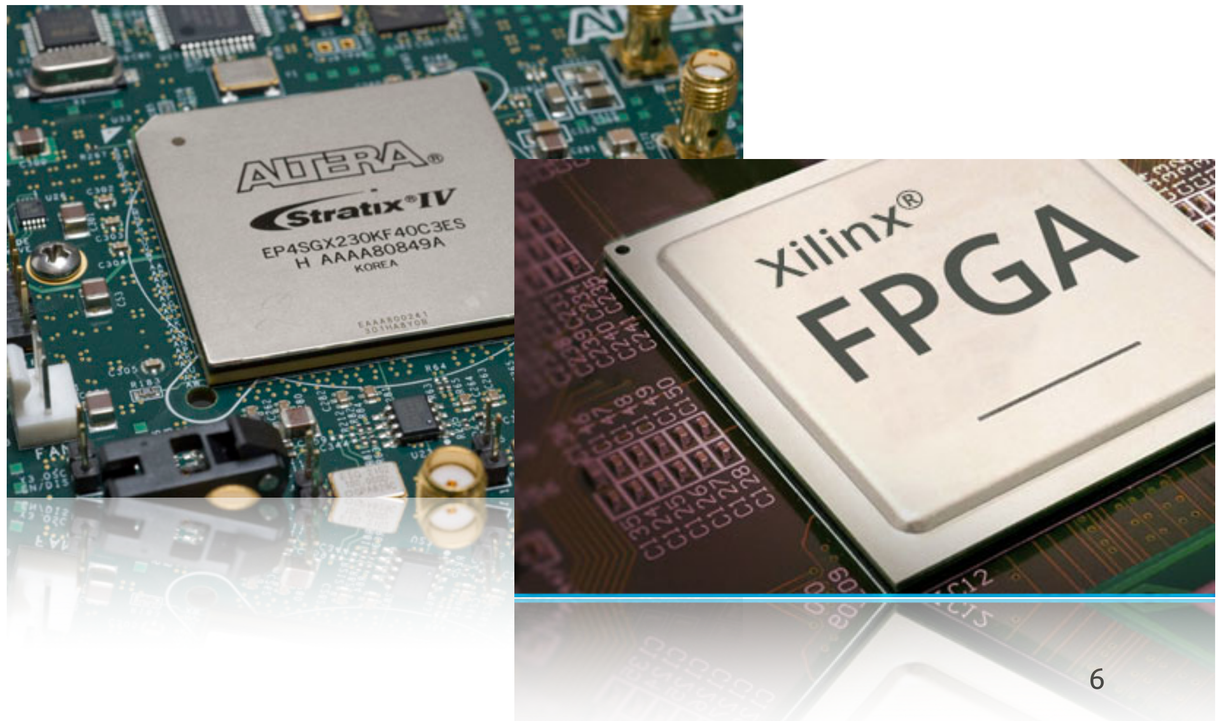
# What are FPGAs?

**Field Programmable Gate Arrays** are reprogrammable integrated circuits

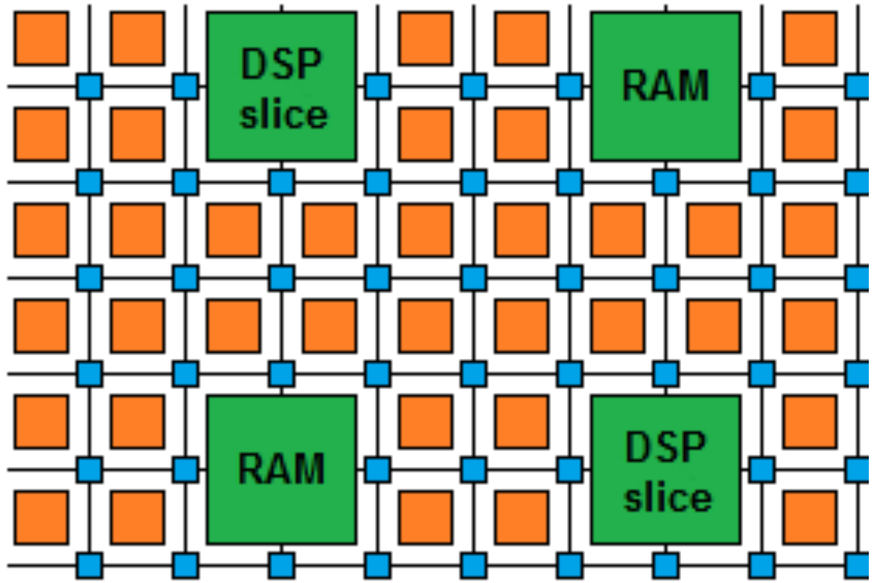
Contain many different building blocks (‘resources’) which are connected together as you desire

Originally popular for prototyping ASICs, but now also for high performance computing

‘Computing in space as well as time’




FPGA diagram




**LUTs -** generic logic

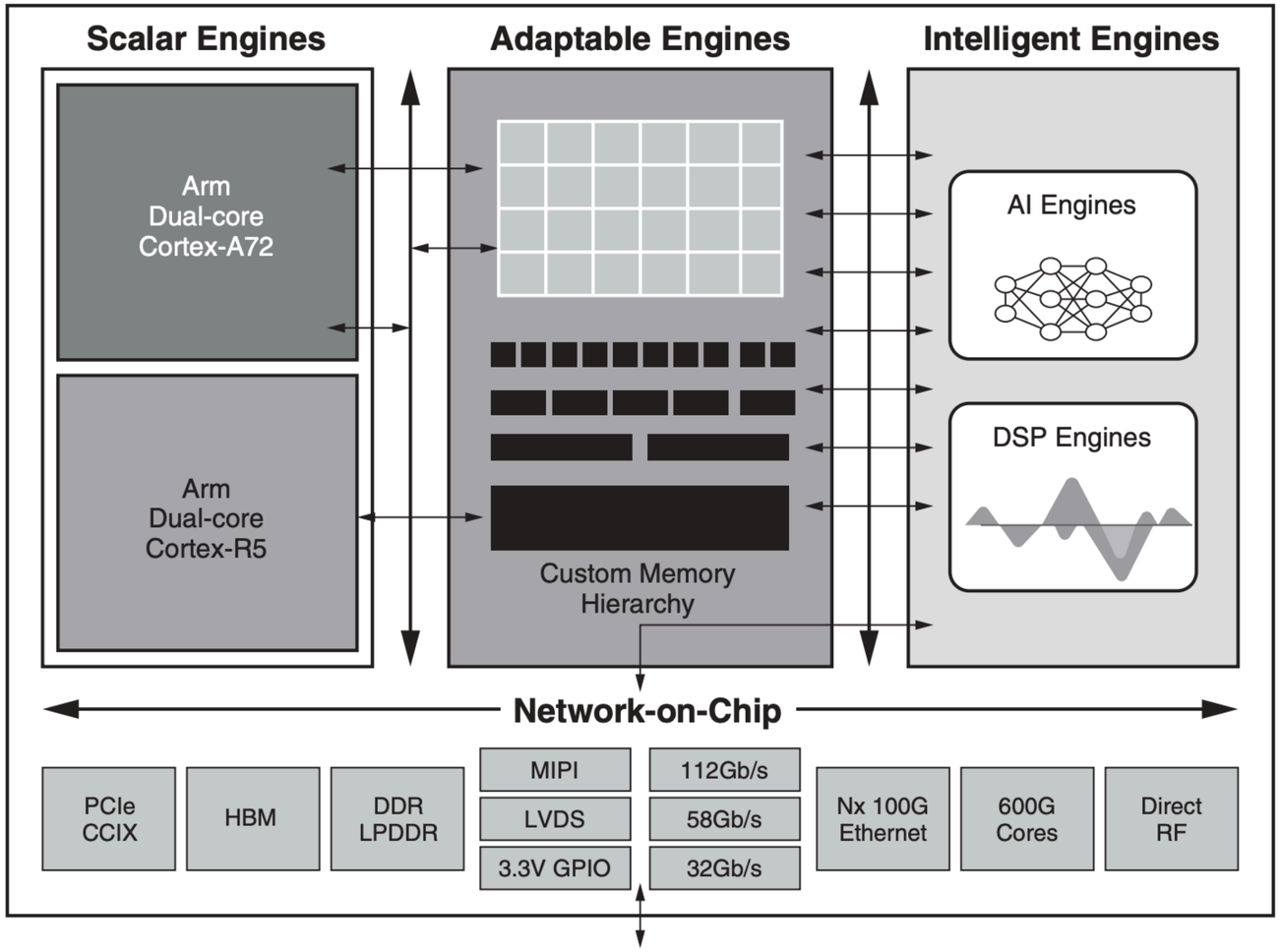
**DSPs -** for multiplication

**BRAM -** for local, high-throughput storage

# Hardened units

- Modern FPGAs have many ‘hardened’ (not-reprogrammable) components
  - High speed transceivers (for IO)
  - ARM cores (e.g. Zynq System on Chip devices)
  - Soon: Vector Processing Units
    - Pictured - Versal
- More efficient (area, frequency, power) than using general logic

https://www.xilinx.com/support/documentation/white_papers/wp505-versal-acap.pdf



# FPGA IO

- IP cores provided to utilise many common interfaces
- Many high speed transceivers, and GPIO pins make FPGAs powerful for embedded applications
- Potential to have sensor / controller IO in same package as compute

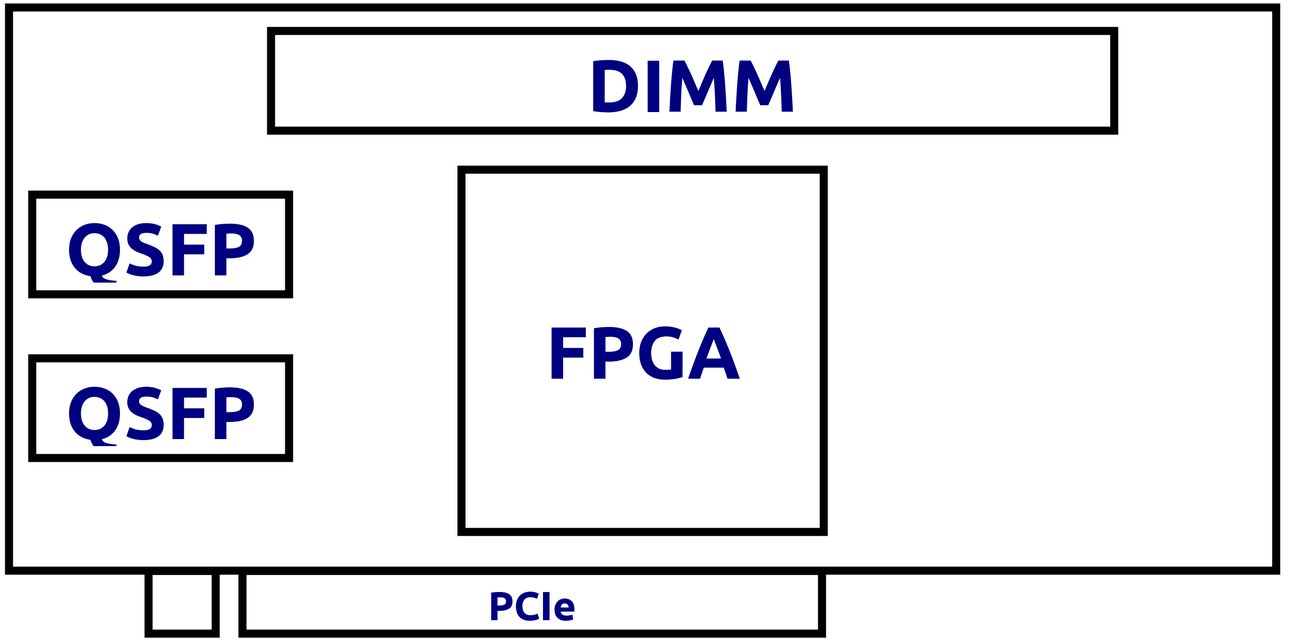
**Major IP Interface Blocks Supported by Xilinx**

| Communications | Compute | Video |
|---|---|---|
| Ethernet | USB | HDMI |
| PCI Express | SATA | DisplayPort |
| Interlaken | SAS | MIPI |
| Aurora | ONFI | SDI |
| Serial RapidIO | AXI Chip2Chip | Ethernet AVB |
| CPRI | IIC | |
| SONET | SPI | |
| Infiniband | UART | |
| OTN | | |

https://www.xilinx.com/products/technology/connectivity.html

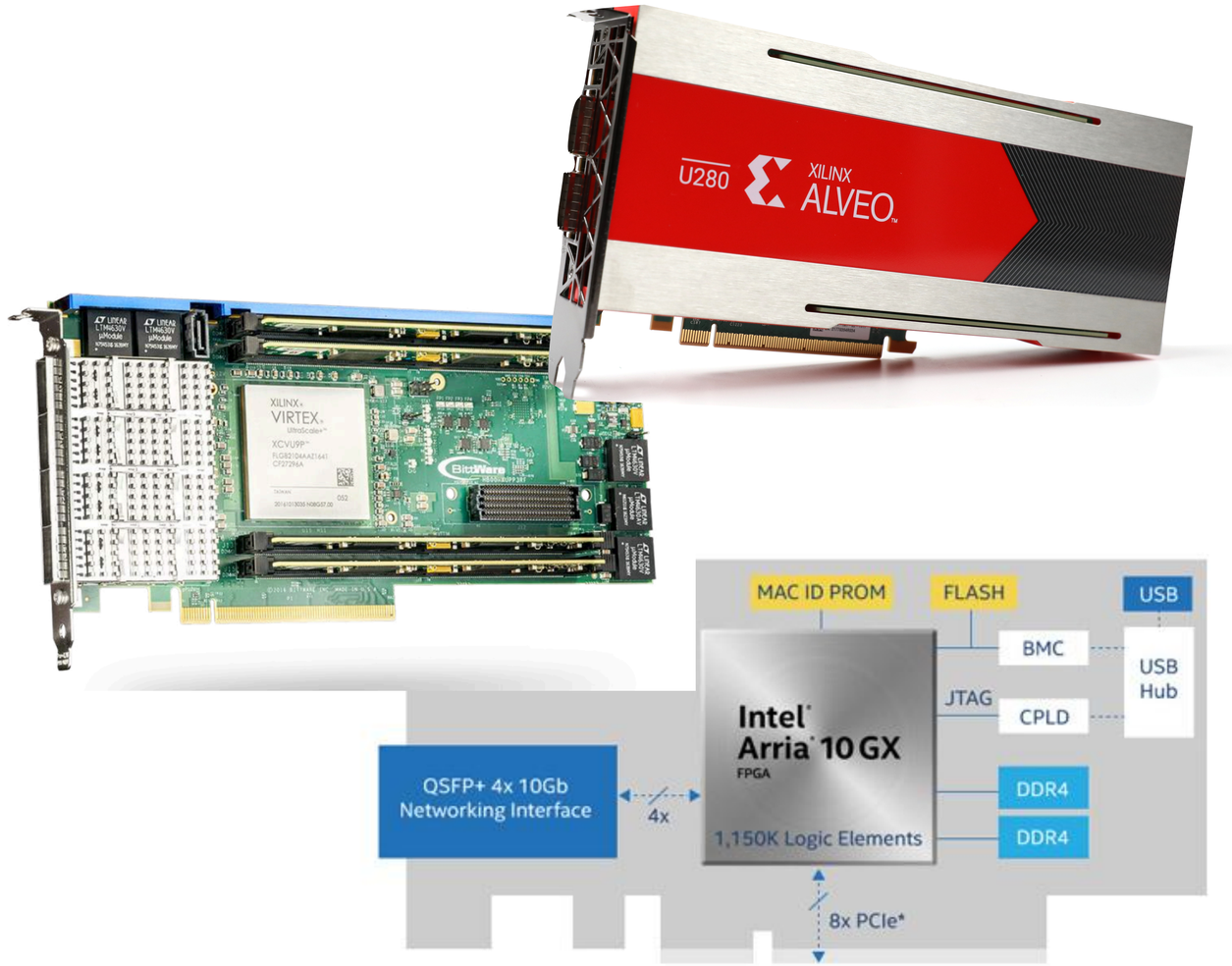
# Anatomy of an FPGA board

- The FPGA does not exist in isolation
- Cartoon of a quite common layout for compute acceleration / datacentre
  - PCIe form factor
  - DIMM for DDR / QDR
  - QSFP for networking
- Different boards have different balance of memory and networking



# Anatomy of an FPGA board

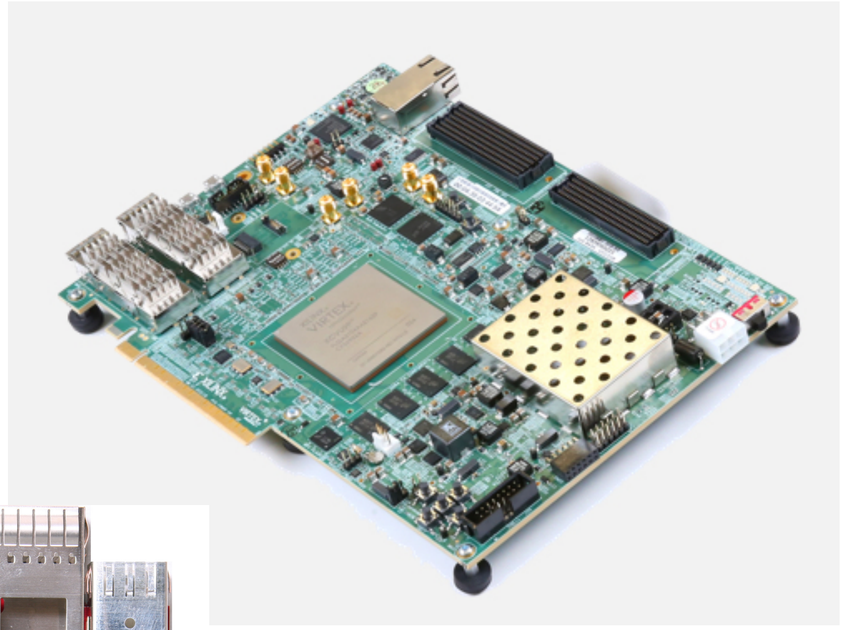
- The FPGA does not exist in isolation
- Cartoon of a quite common layout for compute acceleration / datacentre
  - PCIe form factor
  - DIMM for DDR / QDR
  - QSFP for networking
- Different boards have different FPGAs & balance of memory and networking

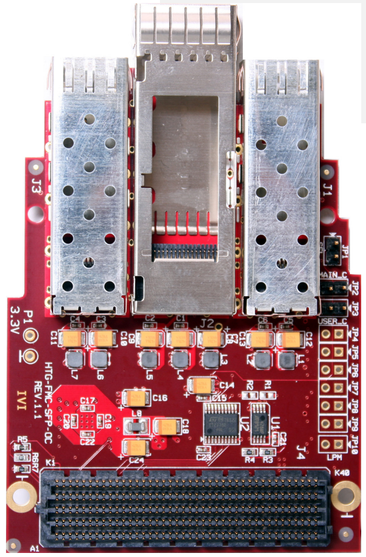



Xilinx, Bittware, Intel

# Anatomy of an FPGA board

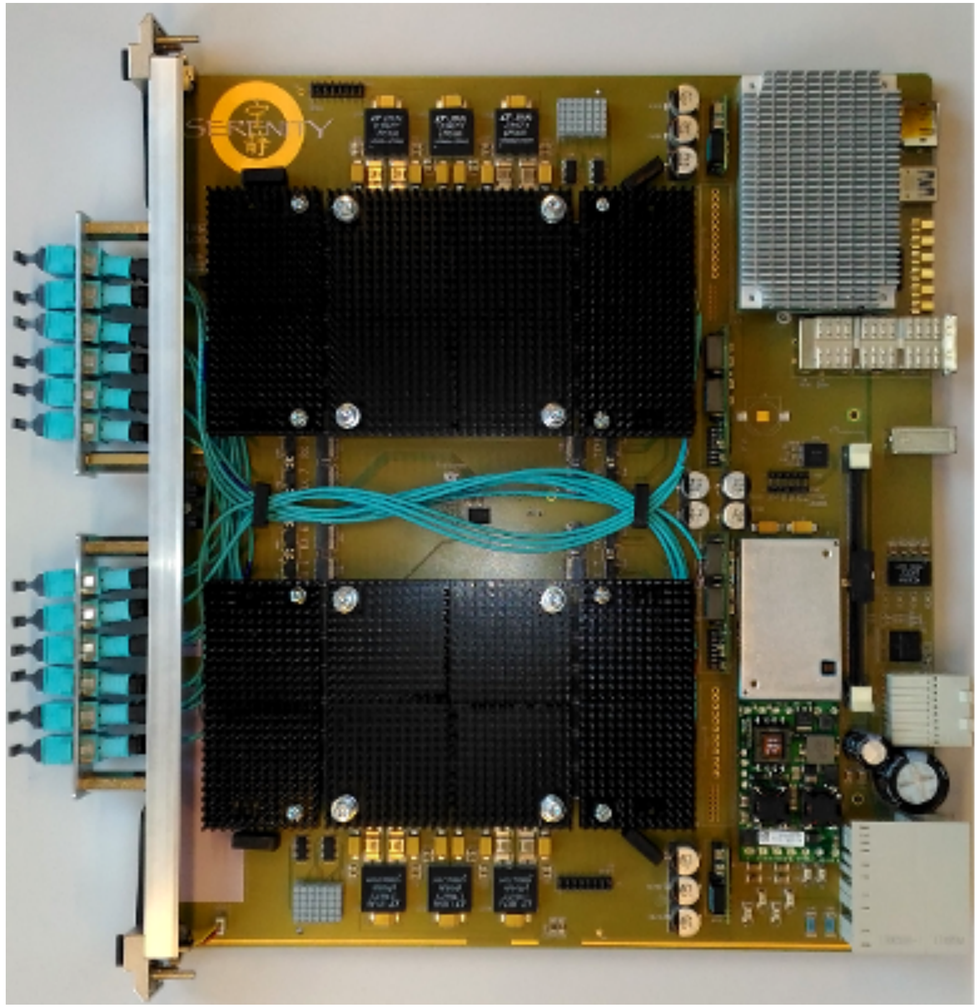
- Development / Evaluation kits
- For developers of end-products
  - FPGA boards or ASIC
- Provide opportunity to gain experience with components and software
- Generally more IO pins exposed than typical compute accelerator cards
- Expandable with FMCs (FPGA mezzanine card)
- Useful for developing custom boards for embedded applications





# LHC Experiment FPGA boards

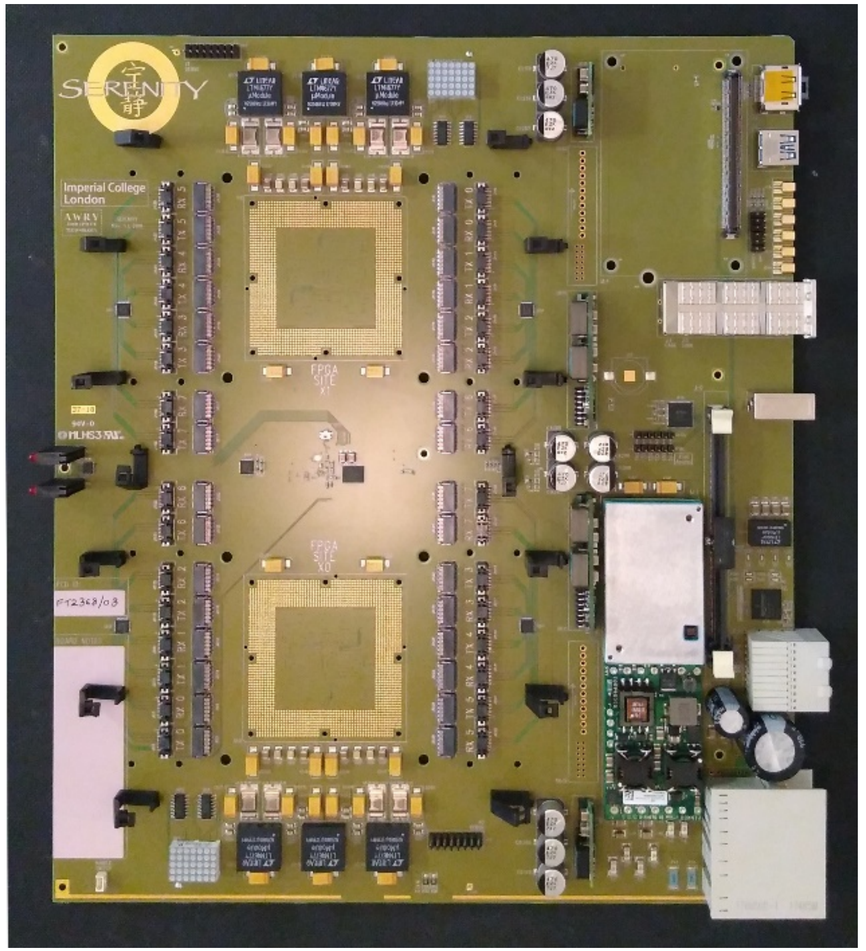
- LHC experiments typically use custom boards in the trigger
- Necessary due to constraints of huge-I/O-bandwidth and low-latency
- Example right - Serenity ATCA blade
  - 2 high-end Xilinx FPGAs
  - Up to 5 Tb/s throughput - optical fibres
  - No external memory
  - Artix service FPGA, COMExpress CPU



http://serenity.web.cern.ch/serenity/

# LHC Experiment FPGA boards

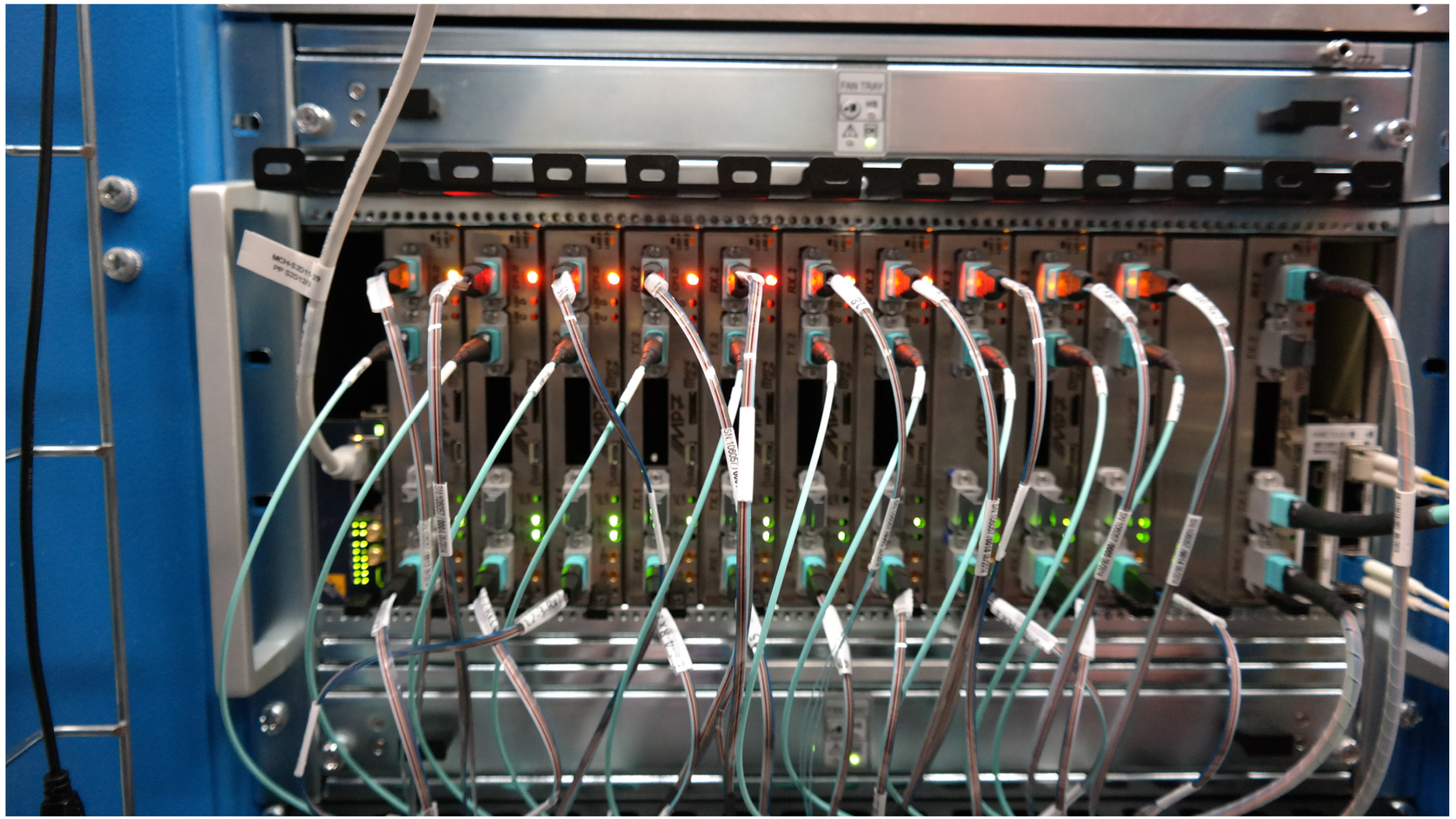
- LHC experiments typically use custom boards in the trigger
- Necessary due to constraints of huge-I/O-bandwidth and low-latency
- Example right - Serenity ATCA blade
  - 2 high-end Xilinx FPGAs
  - Up to 5 Tb/s throughput - optical fibres
  - No external memory
  - Artix service FPGA, COMExpress CPU




http://serenity.web.cern.ch/serenity/

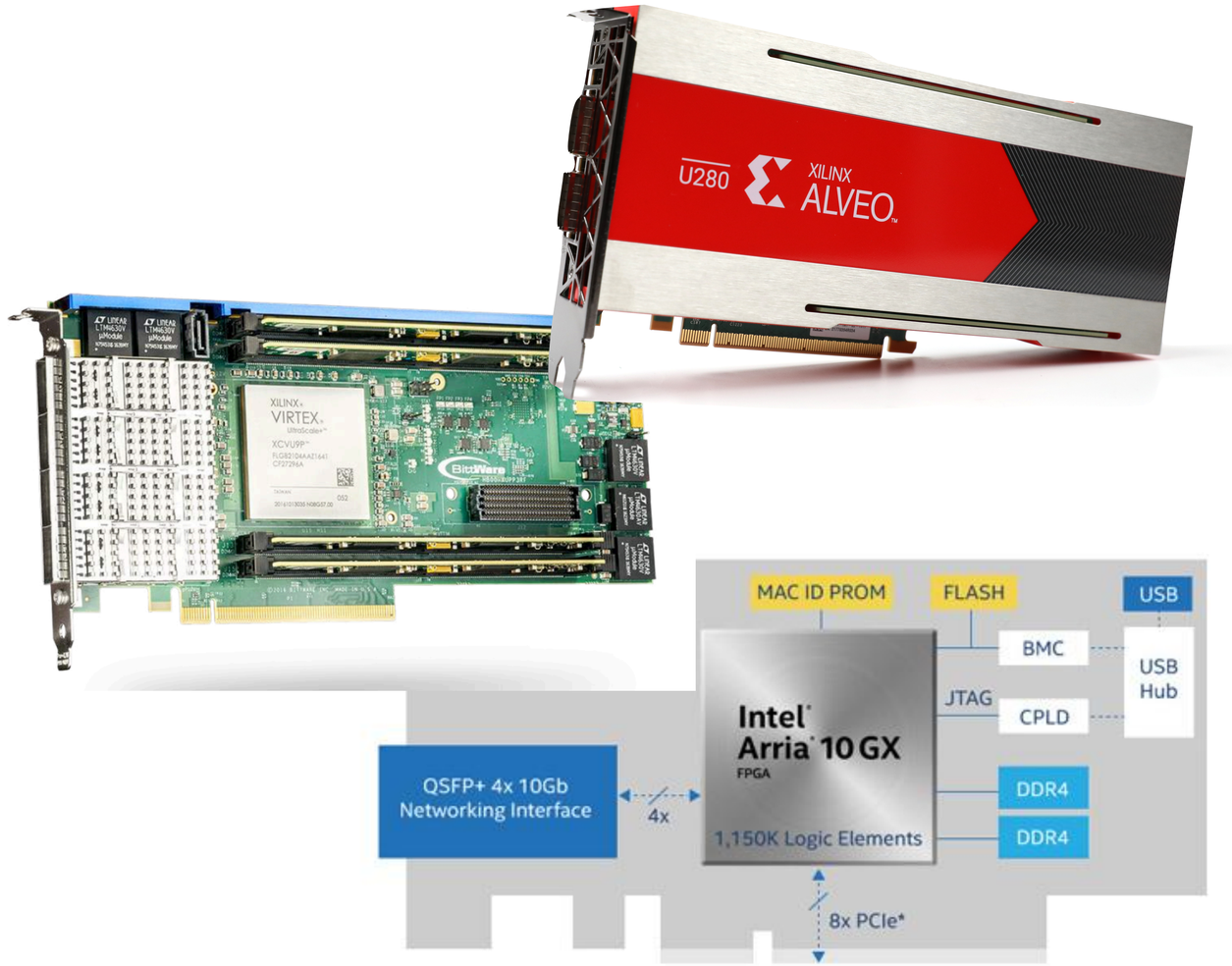
# LHC Experiment FPGA boards

- CMS trigger boards situated in crates underground in cavern next to detector
- CMS trigger @ Hi-Lumi upgrade with comprise hundreds of boards
- Pictured: MP7s in a crate from the current CMS trigger



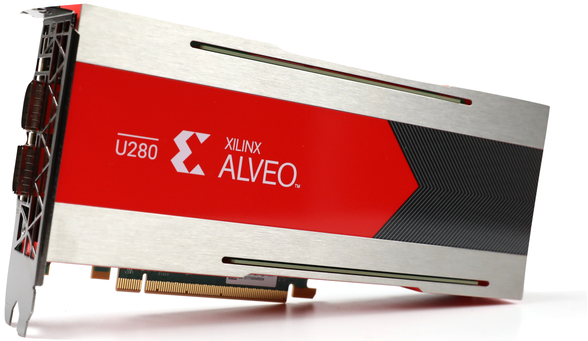
# LHC Experiment FPGA boards

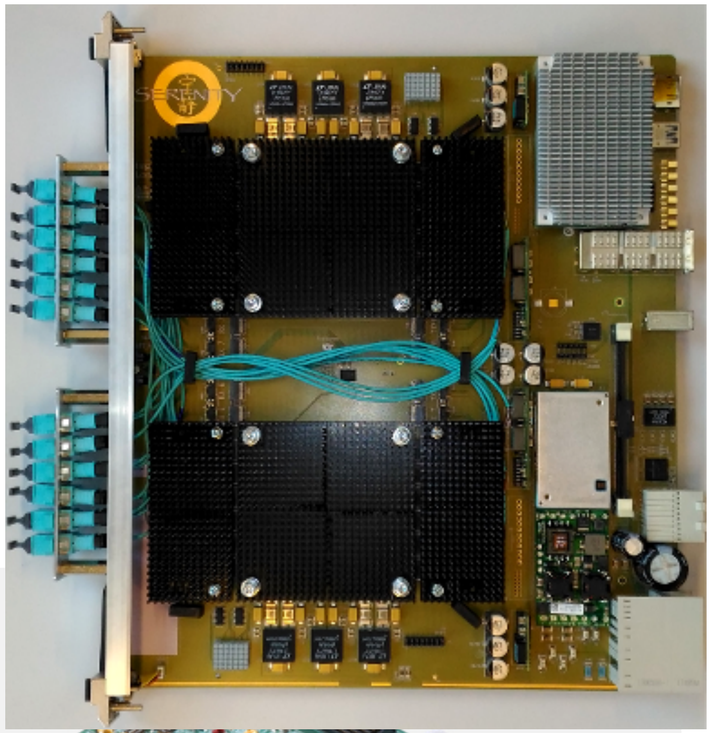
- Much more likely to use commodity hardware acceleration cards for High Level Trigger & offline computing
  - Currently using commodity CPUs
- Ease of use, warranty, tech support, standard server compatibility, associated drivers & compilers

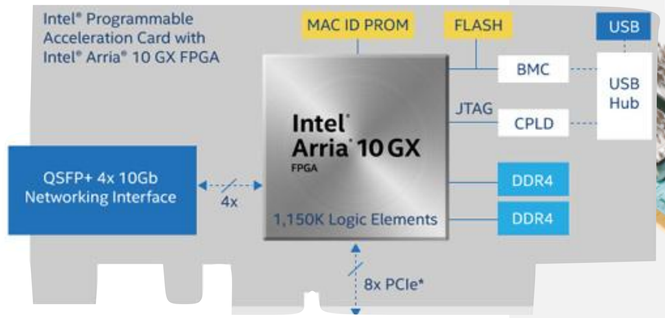


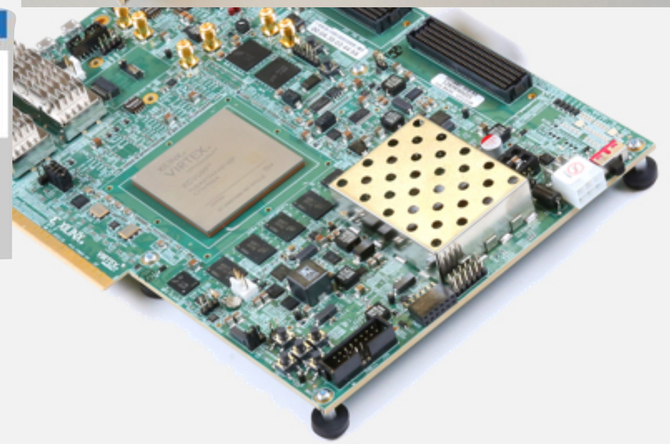
# What do we have?

- All of the above!
- In the lab:
  - Real LHC hardware prototyping / demonstrating upgrade work
  - Development kit for the same purpose
- In the CERN data centre
  - Xilinx Alveo U200 accelerator cards
  - Intel Arria 10 PAC





# FPGA Programming

- FPGAs are notoriously difficult to program…
  - Low level languages
    - Register Transfer Level: VHDL / Verilog
  - Configurable IP cores (transceivers)
  - IO protocols
  - Connect signals to specific pins
  - Hours to compile
- **WORM**
  - Write once, *Run* many

```
entity add is
port(
  clk : in  std_logic;
  a   : in  signed(31 downto 0);
  b   : in  signed(31 downto 0);
  c   : out signed(31 downto 0)
)
end add;

architecture rtl of add is
  if rising_edge(clk) then
    c <= a + b;
  end if;
end rtl;
```

```
set_property CONFIG_VOLTAGE 1.8 [current_design]
set_property CFGBVS GND [current_design]

# PCIe connections
set_property PACKAGE_PIN AK10 [get_ports pcie_sys_clk_p]
set_property PACKAGE_PIN AE19 [get_ports pcie_sys_rst]
set_property PULLUP true [get_ports pcie_sys_rst]
set_property IOSTANDARD LVCMOS18 [get_ports pcie_sys_rst]
set_false_path -from [get_ports pcie_sys_rst]

set_property PACKAGE_PIN AN8 [get_ports {pcie_rxp[0]}]
set_property PACKAGE_PIN AM10 [get_ports {pcie_txp[0]}]
```

# High Level Languages

- Recently we have **High Level Synthesis** C / C++ / SystemC / OpenCL compilers for Xilinx & Intel
  - Much lower time to deployment
  - Automated powerful features not possible with low level languages
    - Automatic pipelining
    - Loop (un)rolling
    - Execution control
- hls4ml built with Vivado HLS

```
int add (int a, int b){
  return a + b;
}
```

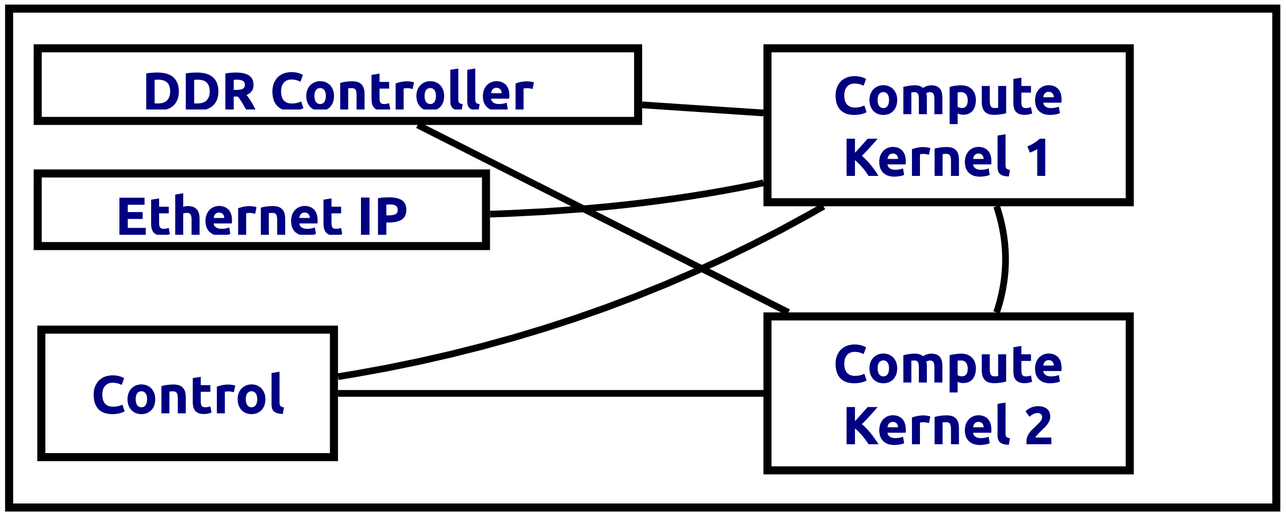
- HLS produces **IP Cores** which can be integrated with other processing or FPGA IO separately

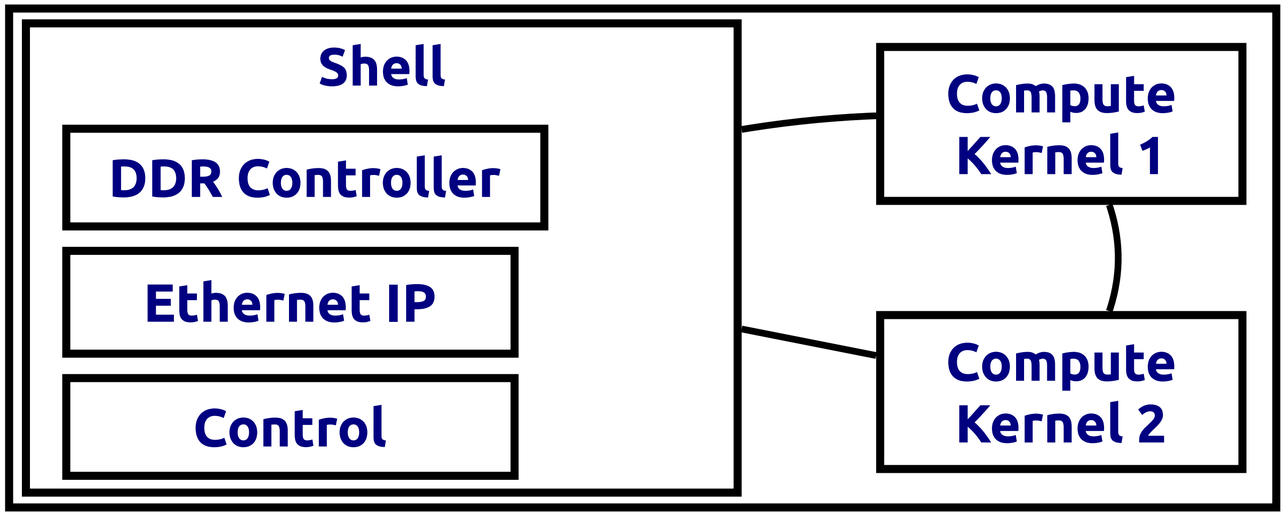
# Even Higher Level

- Domain specific languages / libraries
- Developers map domain specific description of problem to low level FPGA operations
- Like hls4ml!
  - Keras / PyTorch / ONNX description to HLS code
- Also e.g. Xilinx's Video suite
- User may still need to connect algorithms to board infrastructure, and run compile step
- With newer, bigger devices, manufacturers provide pre-compiled bitfiles
- Skip the compile step (~hours)
- But constrained to what bitfiles are available, and they are board specific
- For ML type applications normally means a large Matrix-Vector multiplier unit with caching using local memory
  - Xilinx ML suite, ChaiDNN
- New Xilinx Versal and Vitis range

# Board Firmware Shells

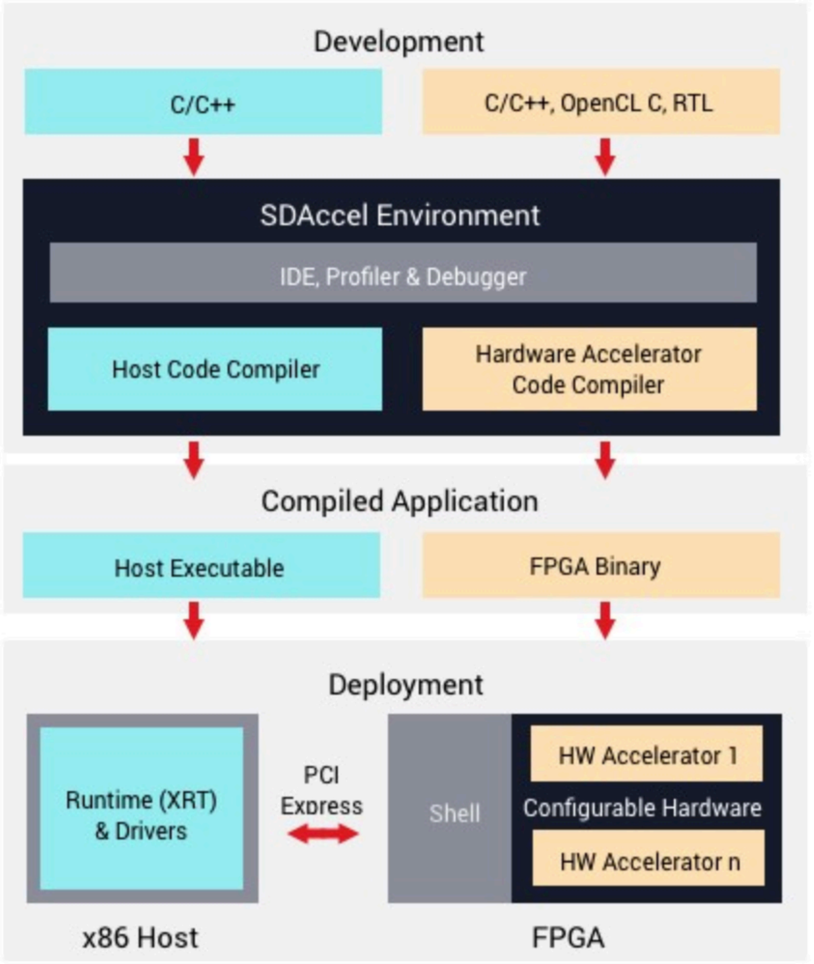
- The algorithm(s) must eventually interface with the pins
- The unsustainable way is to have the algorithms do this directly themselves
- The better way is to for the manufacturer to  provide *shells* which handle the low-level protocols and expose higher-level (typically streams) to the algorithm
- These are necessarily FPGA and board specific



# Board Shells

- Example - Xilinx SDAccel
  - Plugging HLS IP cores into specific boards
- Xilinx provide FPGA *shell* which connects user's AXI Streams to PCIe / DDR interfaces on the physical board
- C/C++ compute kernels use AXI stream objects as function arguments, and you're done
- Application programmer liberated from developing IO protocols



# Conclusions

- FPGA computing is a broad domain
- Which level you choose to enter depends on the end-goal:
- On the board level :
  - Custom boards (ASICs?); development kits; server acceleration cards
- On the programming level:
  - RTL; High Level Synthesis; Domain Specific Libraries; Pre-made binaries
- Boards with firmware shells and compilers are an advantage for getting started quickly
  - e.g. Xilinx Alveo cards and SDAccel
- Development kits and low-level programming needed if you want to go more embedded…