Compilers, APIs and Languages In Service of Parallelism

Máté Ferenc Nagy-Egri

GPU-Day – The Future of Many-Core Computing in Science
Wigner RCP
22-23 June 2017

Table of Contents

Lightning History

How we got to where we are?

The Dark Ages

- Roughly from the late 90s, the graphics brought to life a wonderful beast: GPUs
  - Still going strong
- From the early 2000s, people wanted to do more with GPUs than just graphics
- Before GPGPU was a thing, people violated GPUs in all sorts of ways to make them do their bidding
  - Still in stlye to do compute through graphics APIs
  - Opens the way to new platforms: OpenGL ES?

Enter CUDA

- Nvidia heard the pleads of many whom wished to do great things with GPUs
- CUDA was soon followed by a myriad of domain specific libraries
    - Still one of the most appealing properties of CUDA
- By far the most effort and assets put into any GPGPU API/library
    - Much of the pioneering can be accounted to Nvidia
    - Rock stable drivers and toolchain

On the shoulders of giants

- Many sought to dethrone the first-born of GPGPU APIs
- Stream SDK & Brook+
  - Sic transit gloria mundi
- Apple OpenCL
  - Handed over to Khronos for further development
- Google Renderscript
  - compute-only, then experimental graphics, then again compute-only
- Microsoft C++AMP
  - Took a shot at integrating GPGPU into the C++ language
- Khronos SYCL
  - GPGPU without language extensions

On the shoulders of giants

- Many sought to dethrone the first-born of GPGPU APIs
- OpenMP
  - GPU offload support from version 4.0
- OpenACC
  - PGI (Nvidia) initiative, much like OpenMP 4.0
- F#
  - Auto-parallelizing compiler
- Haskell Futhark
  - Purely functional GPU programming
- Ocaml SPOC
  - Library in multi-paradigm language

Colors of the Rainbow

All things „new” under the Sun

Risen from ashes to epic heights

AMD

Wigner GPU Lab

RYZEN EPYC

Risen from ashes to epic heights

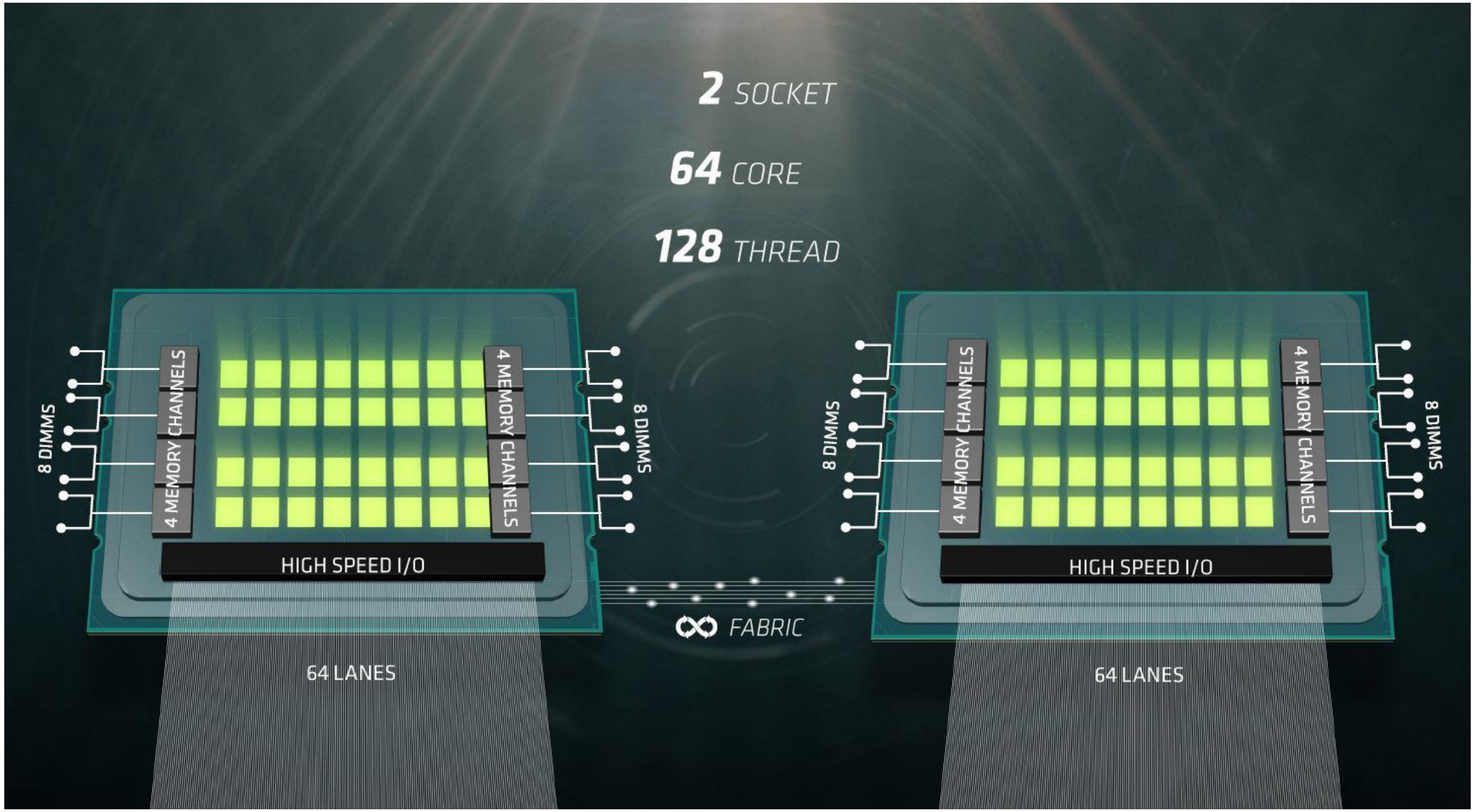
Consumer + Enthusiast

- Ryzen R3-5-7 series CPUs dueling Core i3-5-7
- Based on the brand new Zen architecture
  - Similar SMT design as Intel counterparts
- Threadripper (16C-32T) battles new Core-i9 processors

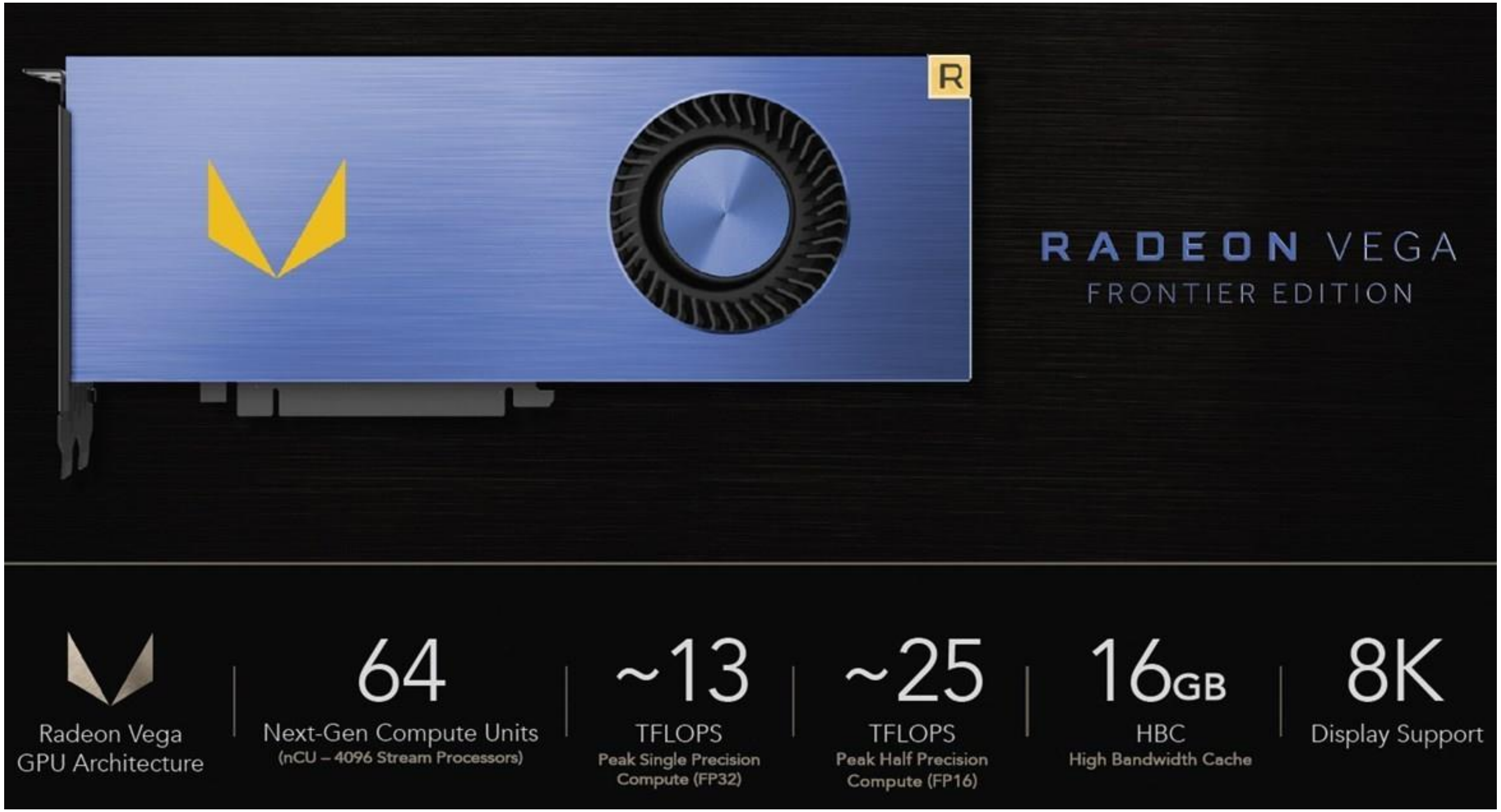
Workstation + Server

- EPYC series (formerly known as Naples)
- 2-4 Zen modules on one die
- New Infinity Fabric interconnect
- Heavily biased towards IO capabilities
  - Compute ain’t too shabby either

AMD EPYC for IO supremacy



Enter VEGA



Radeon Vega GPU Architecture

64 Next-Gen Compute Units (nCU – 4096 Stream Processors)

~13 TFLOPS Peak Single Precision Compute (FP32)

~25 TFLOPS Peak Half Precision Compute (FP16)

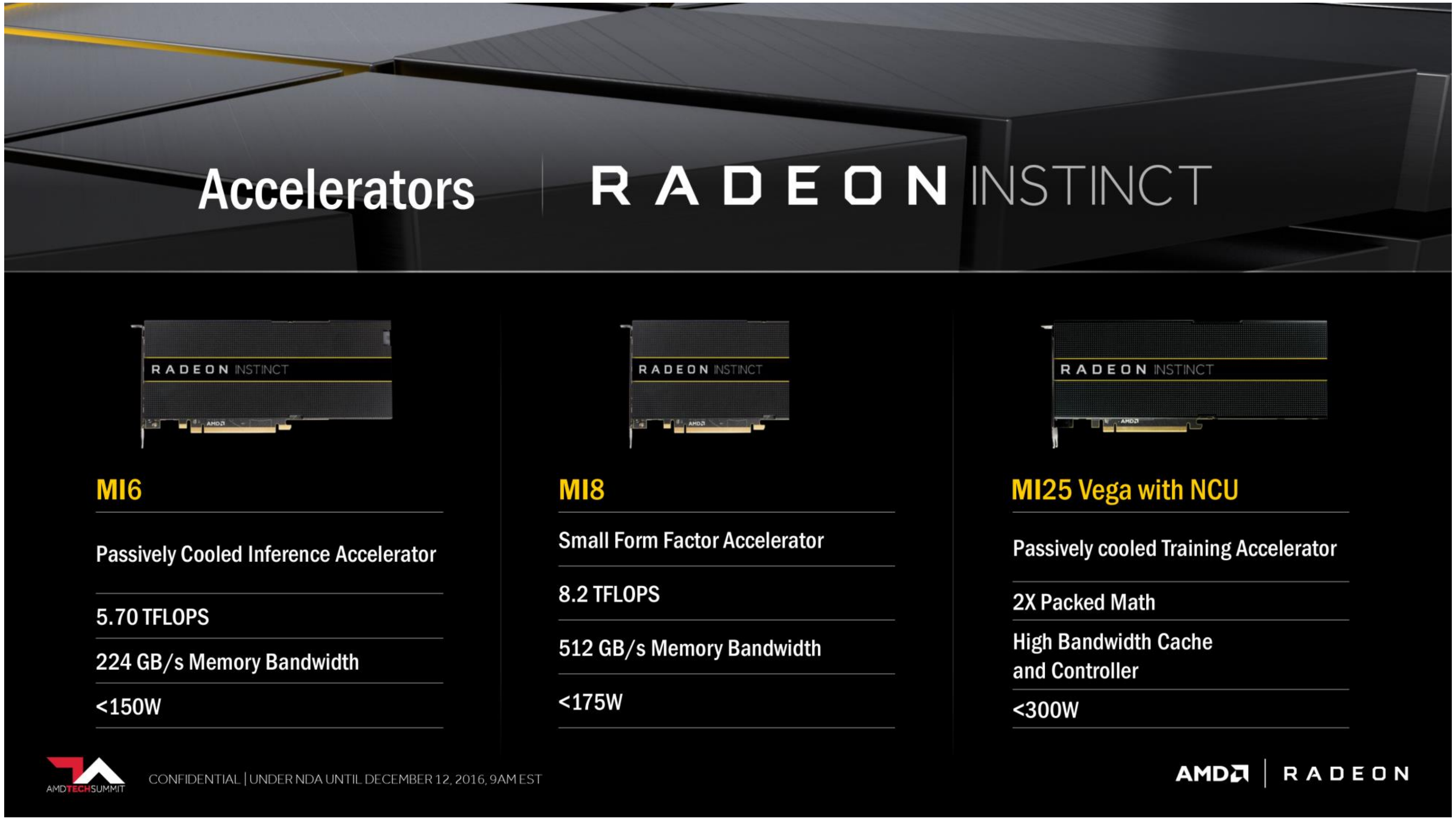
16GB HBC High Bandwidth Cache

8K Display Support

Enter VEGA

- Heavy bias towards graphics and AI workloads
- Two initial configurations
  - Aircooled 300 Watt ($1199)
  - Watercooled 375 Watt ($1799)
- Double precision ~675 Gflops

Head on AI



Accelerators | RADEON INSTINCT

MI6	MI8	MI25 Vega with NCU
Passively Cooled Inference Accelerator	Small Form Factor Accelerator	Passively cooled Training Accelerator
5.70 TFLOPS	8.2 TFLOPS	2X Packed Math
224 GB/s Memory Bandwidth	512 GB/s Memory Bandwidth	High Bandwidth Cache and Controller
<150W	<175W	<300W

AMD TECH SUMMIT

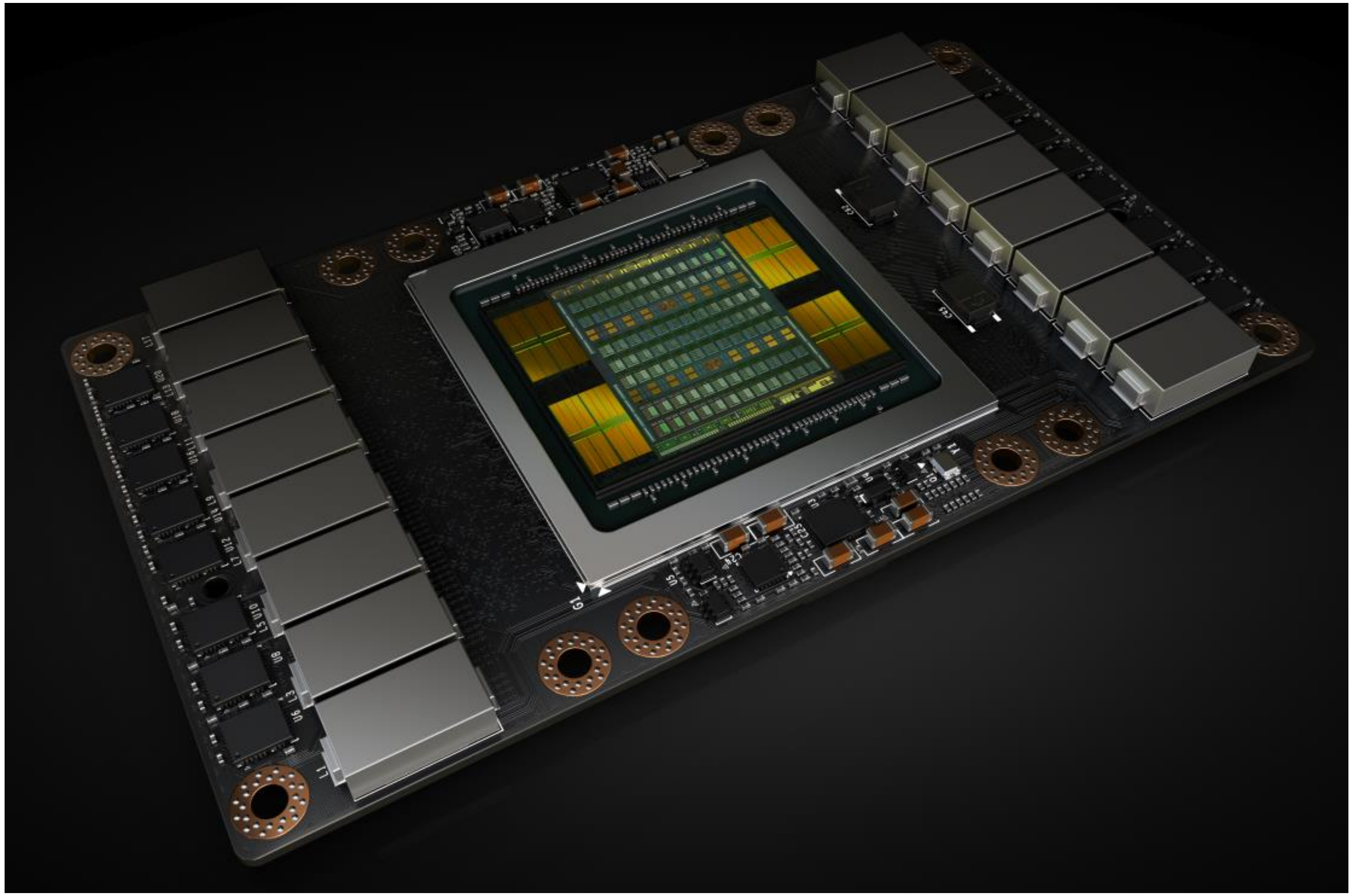
CONFIDENTIAL | UNDER NDA UNTIL DECEMBER 12, 2016, 9AM EST

AMD | RADEON

Boltzmann initiative

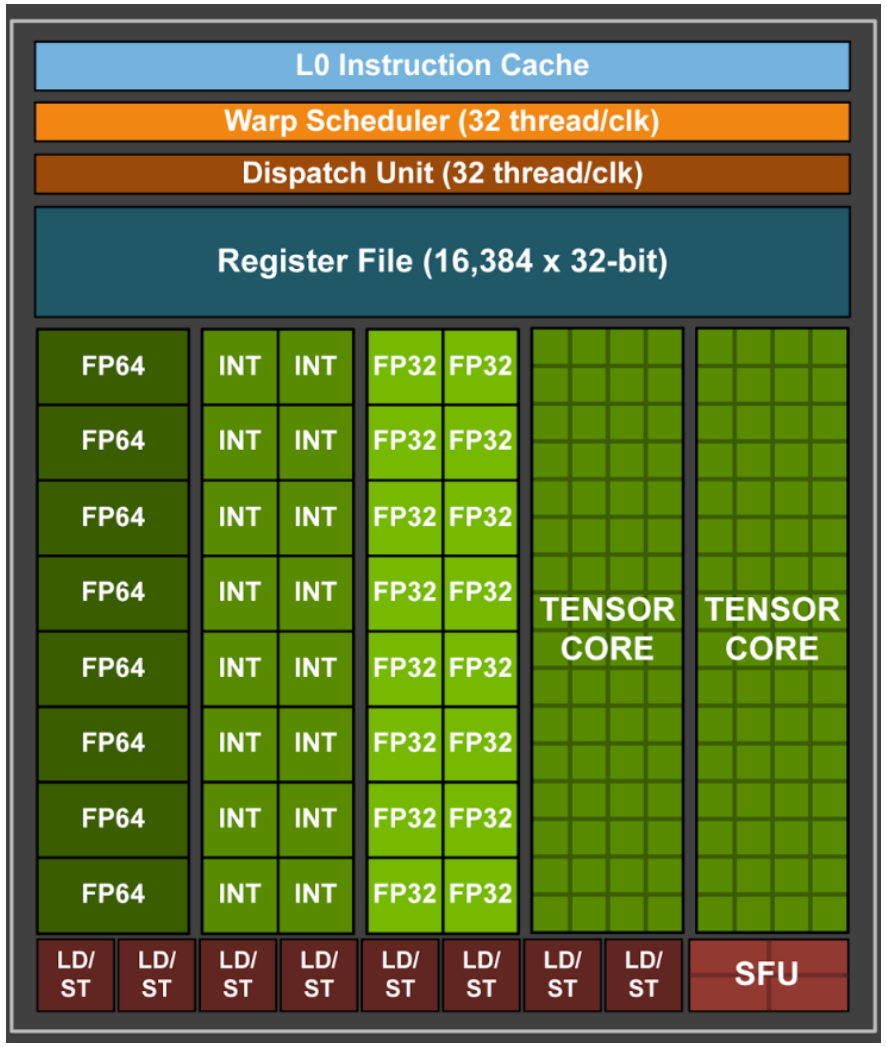
- AMD’s efforts to open-source much of their software
  - Drivers, runtimes, compilers, instruction set architectures, libraries, SDKs, etc.
    - Con: clear sign of lacking manpower
    - Pro: harness OSS toolchains, trust and effort
  - Bears fruit both short and long term
- ROCm, driver and runtime of the future (compute)
  - Minimum system requirement: PCI-E 3.0 atomics (!)
- HCC, compiler infrastructure for heterogenous programming
  - HC attribute-driven single-source C++ language
  - C++AMP 1.2 support
  - OpenCL 1.2/2.0 (host/device) support

Volta on stage



Inside Volta

- Bias towards AI
- Dedicated INT, FP32, FP64 and now FP16 tensor cores
- Futuristic specs
  - 16 GB HBM2
  - 5120 CUDA cores
  - 7,5 Tflops DP, 15 Tflops SP
  - 16 MB cache
  - 20 MB registers


L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
FP64
INT
FP32
TENSOR CORE
LD/ ST
SFU

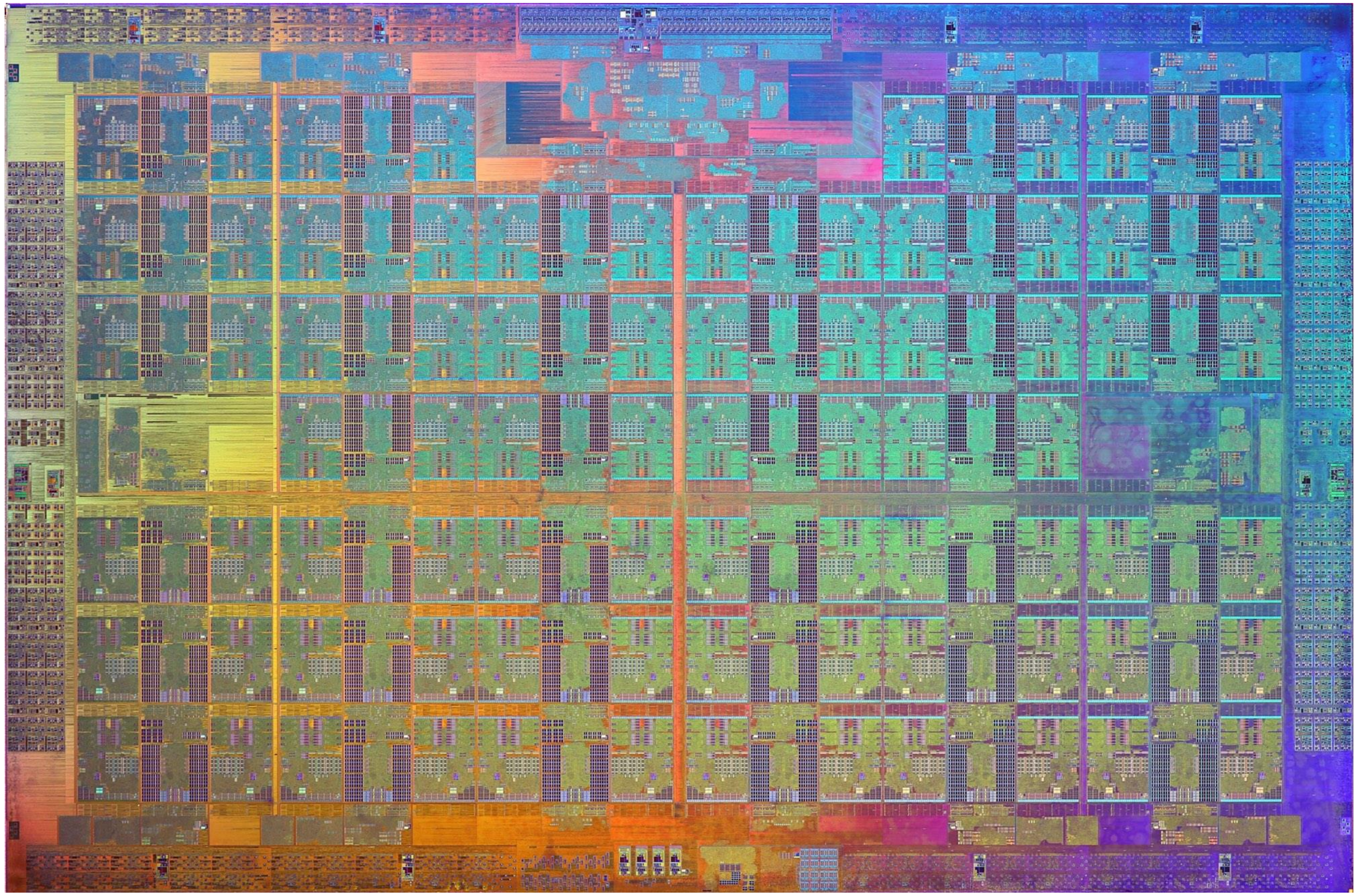
Deep learning at the forefront

- Nvidia Drive PX 2
  - Board designed for (semi-)autonomous car control
- Nvidia DGX-1
  - Custom designed 1U rack filled with Volta cards to be connected via NVLINK (proprietary interconnect)
- Nvidia Drive Works
  - Set of tools commonly associated with implementing autonomous driving/assistance control software

The grass is always greener

- Nvidia tackles hard both consumer and professional front-lines
- Nvidia GRID as their new, cloud-based, low-latency desktop/gaming streaming technology
- CUDA 8.5 goes further than all previous versions
  - Fairly complete C++14 support
  - Extends the range of domain specific libraries
    - cuBLAS (+device), cuFFT, cuRAND, cuSOLVER, nvBLAS, nvCUVID, nvGRAPH, nvRTC, Thrust
- The LLVM compiler toolchain make steady progress towards providing an open-source alternative to nvcc.

Intel Knights Landing



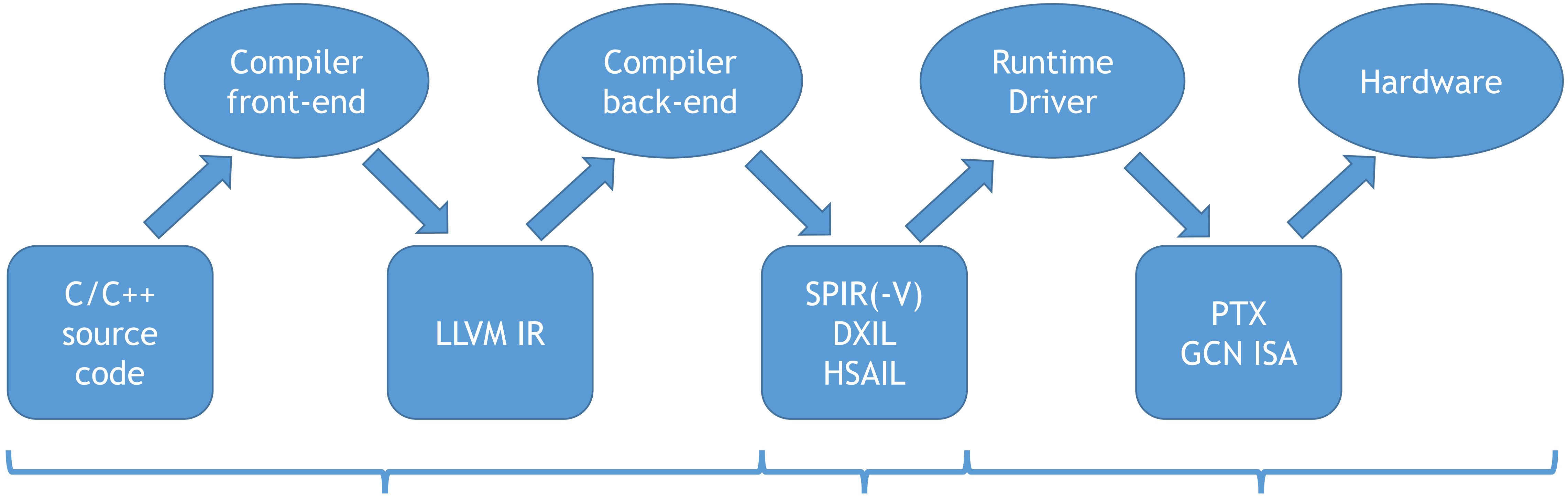
Intel Knights Landing

- Up to 72 x86 cores on one die
- Peak 1,5 GHz
- HyperThreading 4 threads onto same core
- Host processor socket

State of The Art

Where are we now?

A fairly simple GPU compilation model

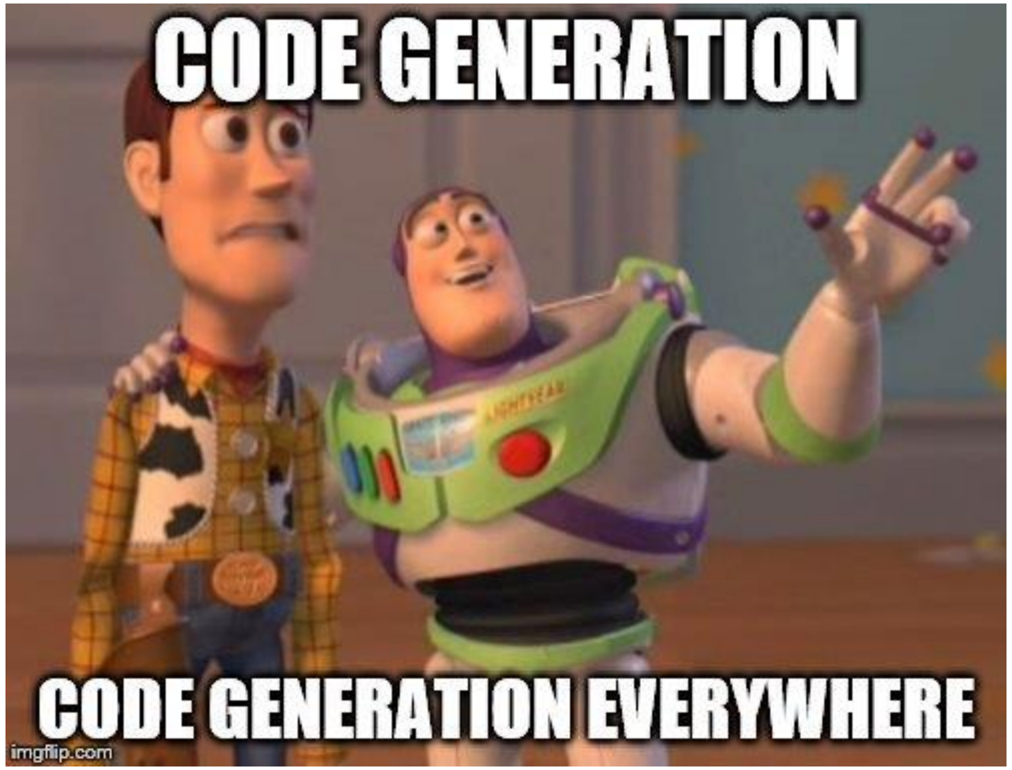


Compiler related

API related

Vendor related

Age of compilers



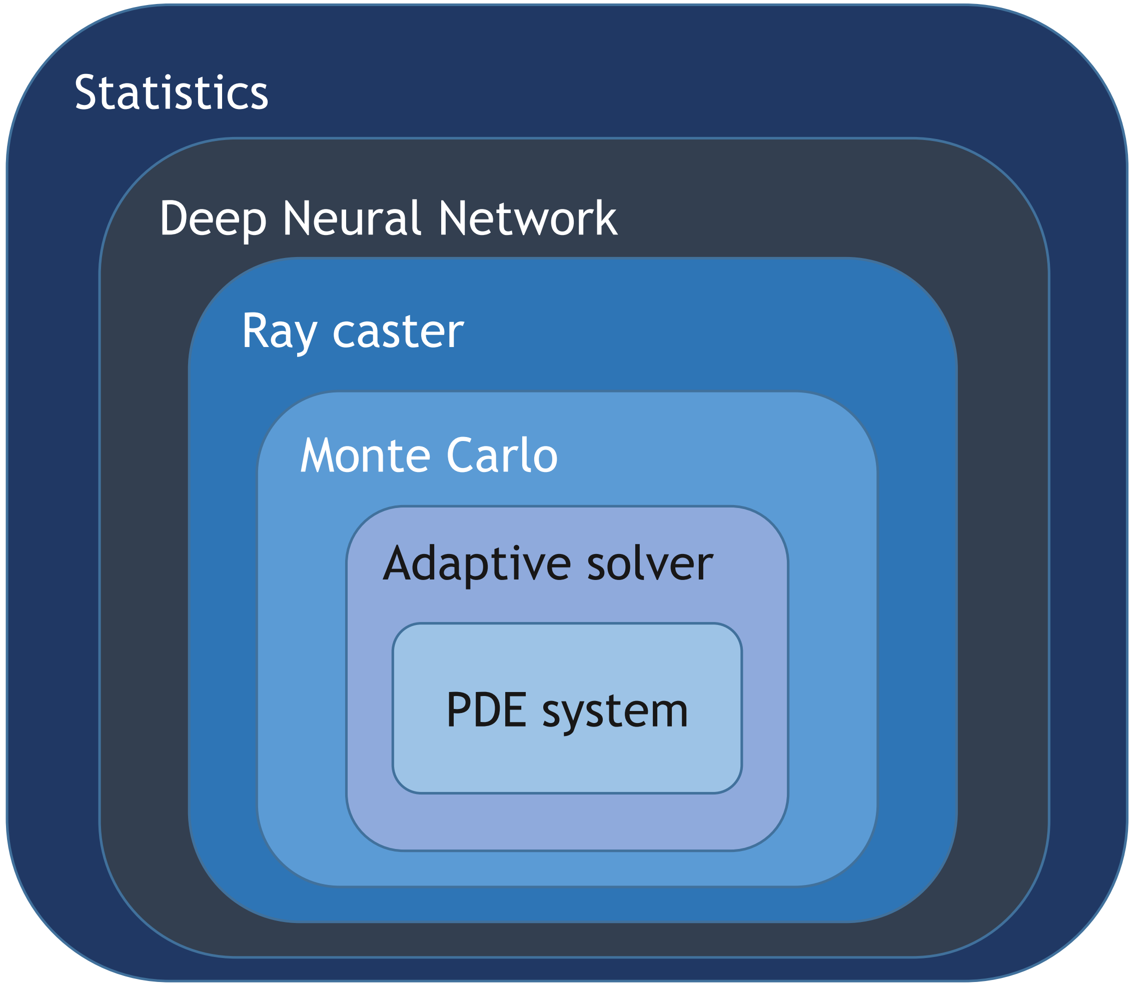
- As the complexity of leading graphics/compute increases, code complexity follows
- People make errors
- Inserting a tool in between humans and the actual code may help
  - And increases complexity in itself, but usually a net win

Exempli gratia

- C++/WinRT
  - Given an input .winmd file that contains all the valid types and functions of the Windows Runtime (OS API), the compiler generates ISO C++ header files with elaborate TMP that facilitate consuming/authoring COM objects
- Vulkan
  - C/C++ headers of various Khronos APIs are generated from the XML file defining all API functions
- QML/XAML
  - Declarative languages to define GUIs, which in turn will be compiled to C++ source code implementing GUI presentation and internal logic
- Qt3D
  - Declarative language for HW state switches required to render a scene

Typical computational science

- Handling the composition of algorithms is different than executing only a single one
  - Fusion, concurrency?
- What parallelisation should we employ on the various levels?
  - Human intuition may be sufficient
  - Usually is parameter sensitive
  - Heuristics, PGO?
- Rapid prototyping?



Why compilers matter?

- Without exploiting information on the levels they manifest, we place unneccessary burden on later stages of compilation
  - Recognizing identity matrix multiplication on assembly level?
- Just-in-time compilation may be suitable
  - Tensorflow XLA middleware
- DSL
  - Using many DSLs give birth to the ROX (Relational-ObjectOriented-XML) problem.
- EDSL
  - Requires a very good language to embed
    - Either an expressive functional/multi-paradigm language
    - An expressive imperative/low-level language (C++)
  - Stresses the compiler

Thank you for your attention!

Questions?