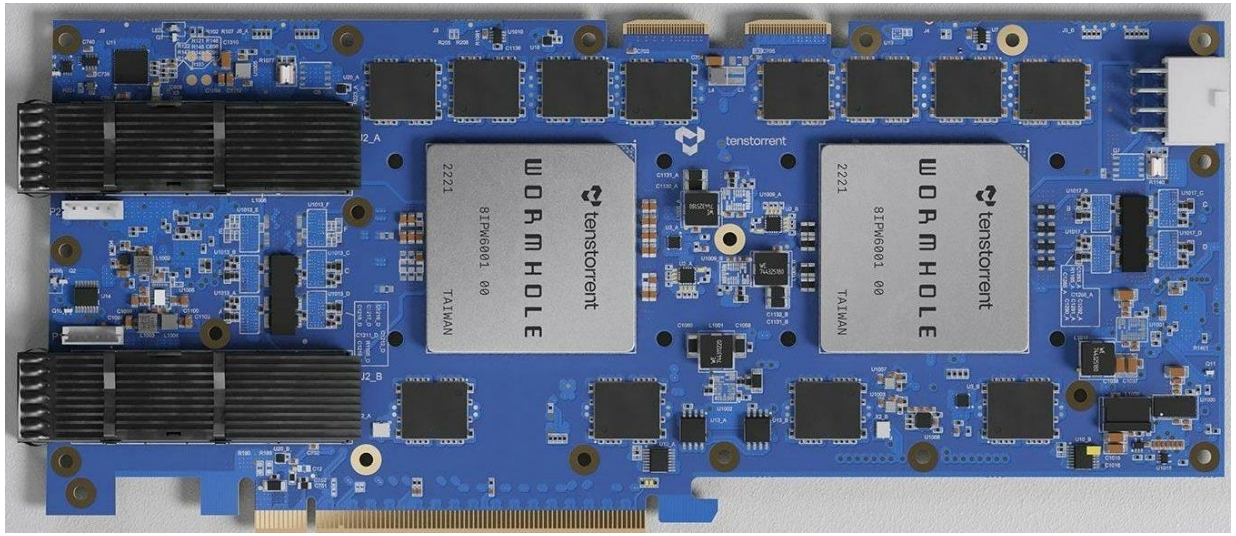
An overview of the Tenstorrent architecture



epcc

tenstorrent

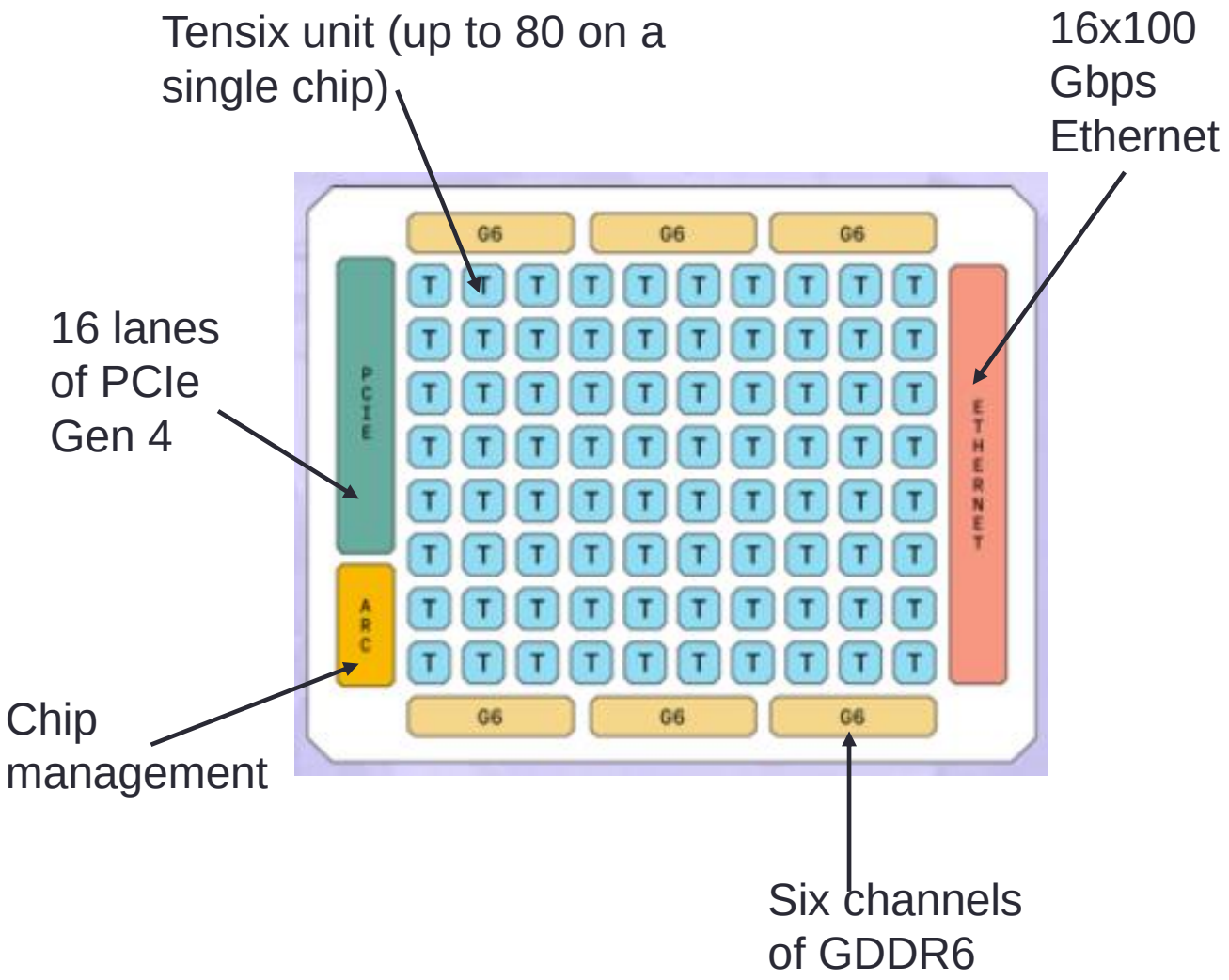
Scaling out rather than up

- The Tenstorrent approach is to scale out a (fairly) simple initial compute unit across a chip and multiple chips
  - This simple unit is known as a Tensix unit (more on this soon….)
  - PCIe accelerator cards contain one or more chips, each with many Tensix units

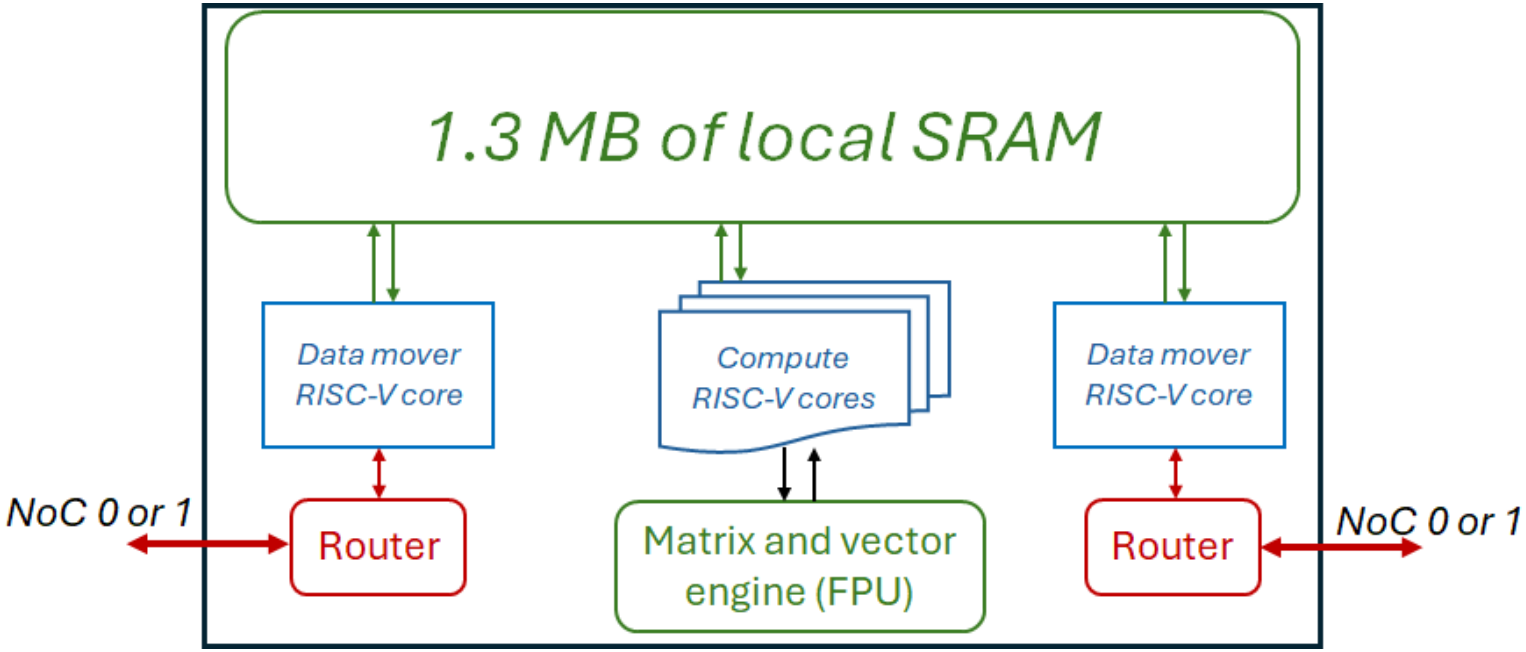
- Cards then scale by being interconnected together
  - These then all appear as a very large (virtual) chip
  - Can do this yourself with the correct cables and connectors
  - The Galaxy contains 32 Wormholes

A single Wormhole chip….

- Built on a 12 nm process
- 12GB GDDR6 on the board
- Draws up to 300 Watts
- Performance:
  - 292 TFLOPS (FP8)
  - 164 TFLOPS (BLOCKFP8)



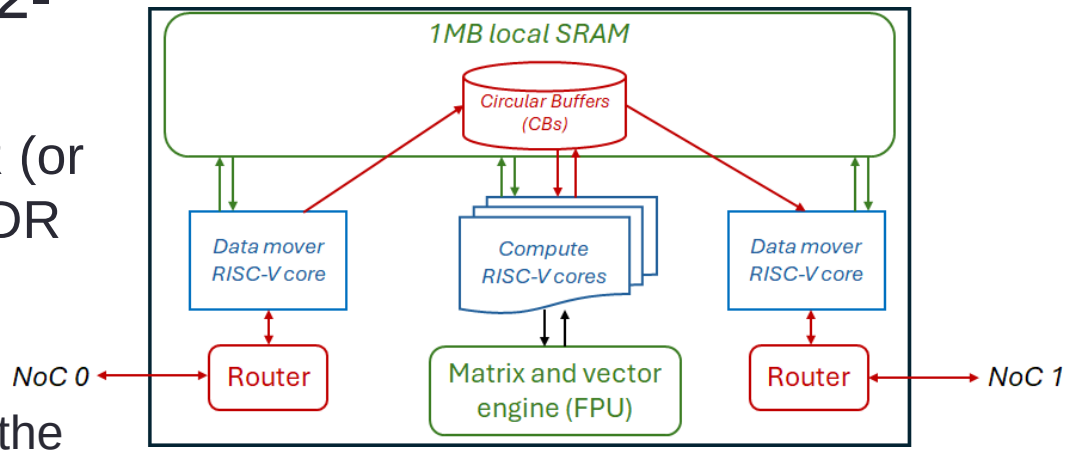
A Tensix unit



- Each Tensix unit contains:
  - Five “baby” RISC-V CPU cores
    - Two of these are for data movement, three drive the compute side by driving the matrix and vector engine
  - A matrix and vector engine (FPU)
  - 1.3MB of local fast SRAM (a bit like a cache)
  - Two routers (one connected to each data mover core)

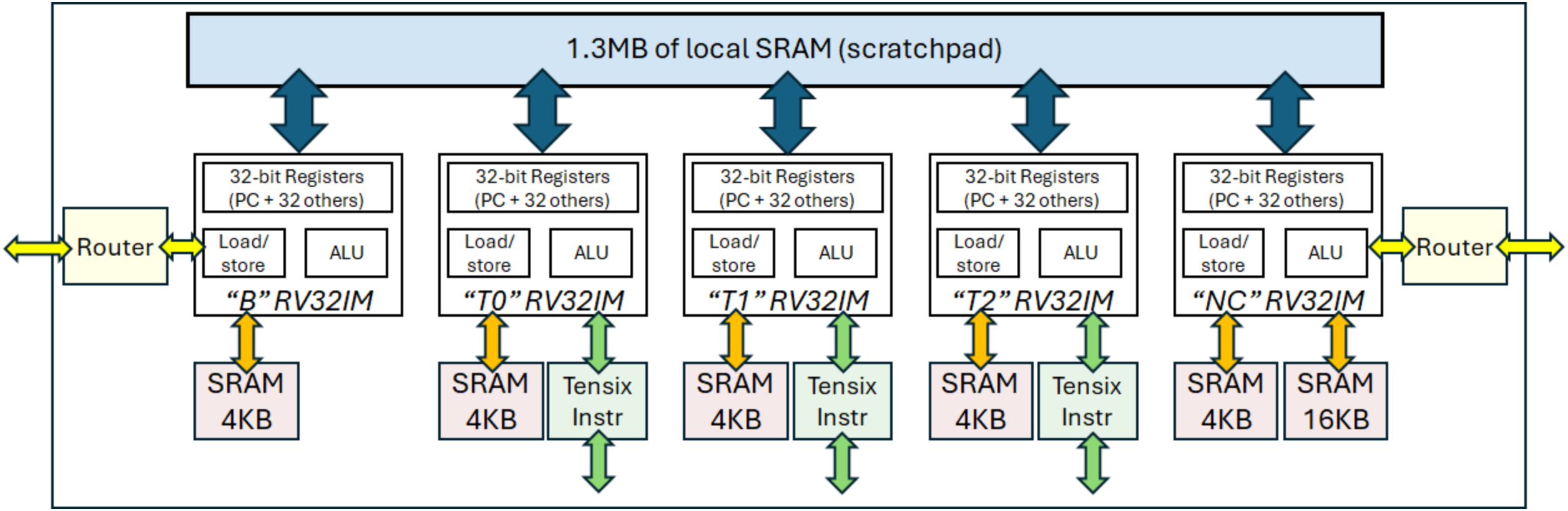
RISC-V CPU cores

- The five “baby” CPU cores are very simple (32-bit RISC-V with Integer support)
  - Two data movers one to get data from external DDR (or another Tensix unit) in, and one to write results to DDR or another Tensix
  - Three compute cores that interact with the FPU
    - One packs data into registers of the FPU, one controls the FPU compute, and the third unpacks from FPU result registers to SRAM.



- RISC-V cores communicate with each other via Circular Buffers (CBs)
  - CBs contain pages of memory, each is a configurable size and follows a producer-consumer approach.
    - Producers will wait until there is a free page, fill this and push to make it available
    - Consumers will block for a page to pushed and made available, read the data and then pop it

The RISC-V cores in more detail….

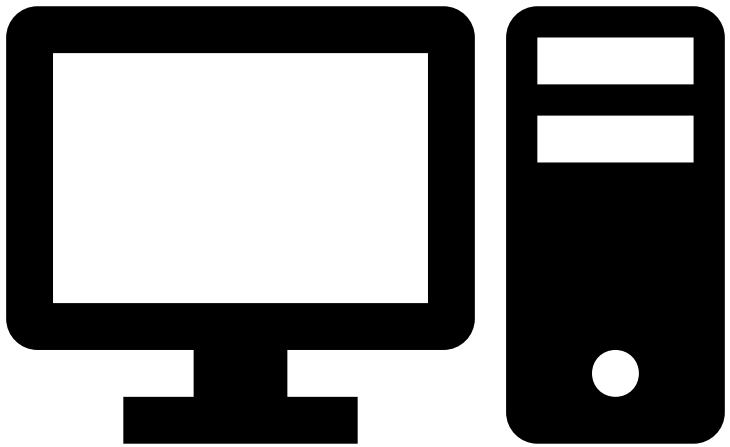


- Each core has some associated SRAM as a local memory for data (and instructions for NC)
- More details on the Tensix instructions and the compute later on….

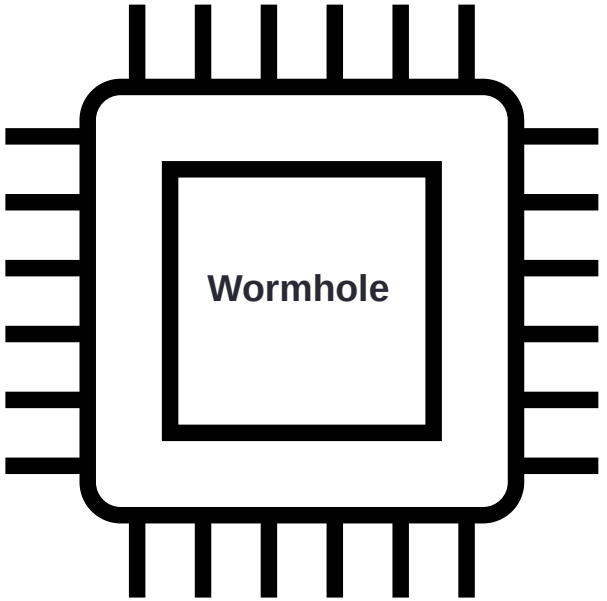
Programmer’s perspective

Host

PCIe accelerator



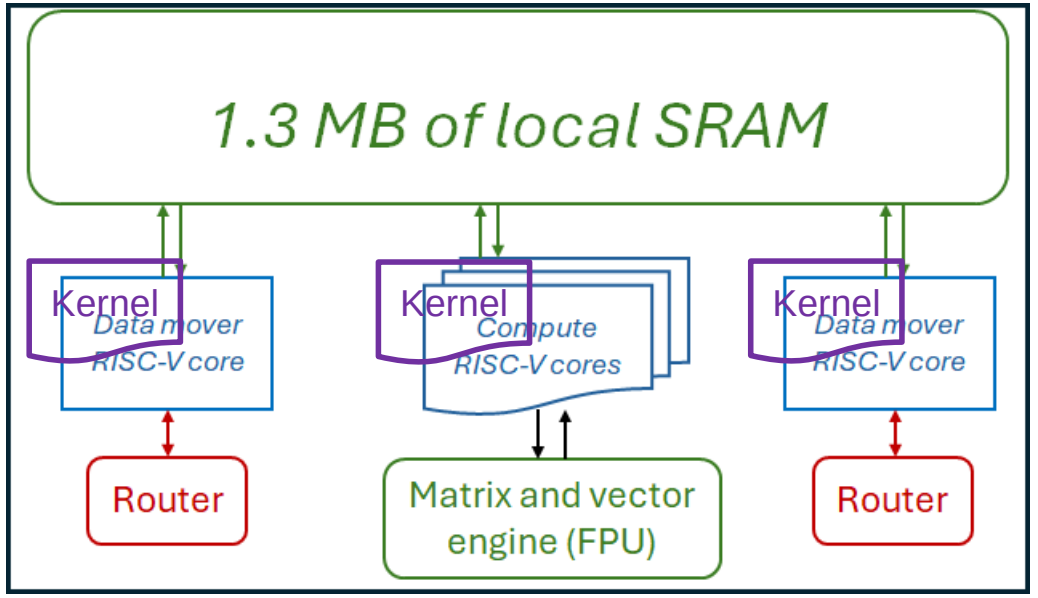
- Input data to DDR
- CB configuration
- Kernels



- Results from DDR

Programmer's perspective

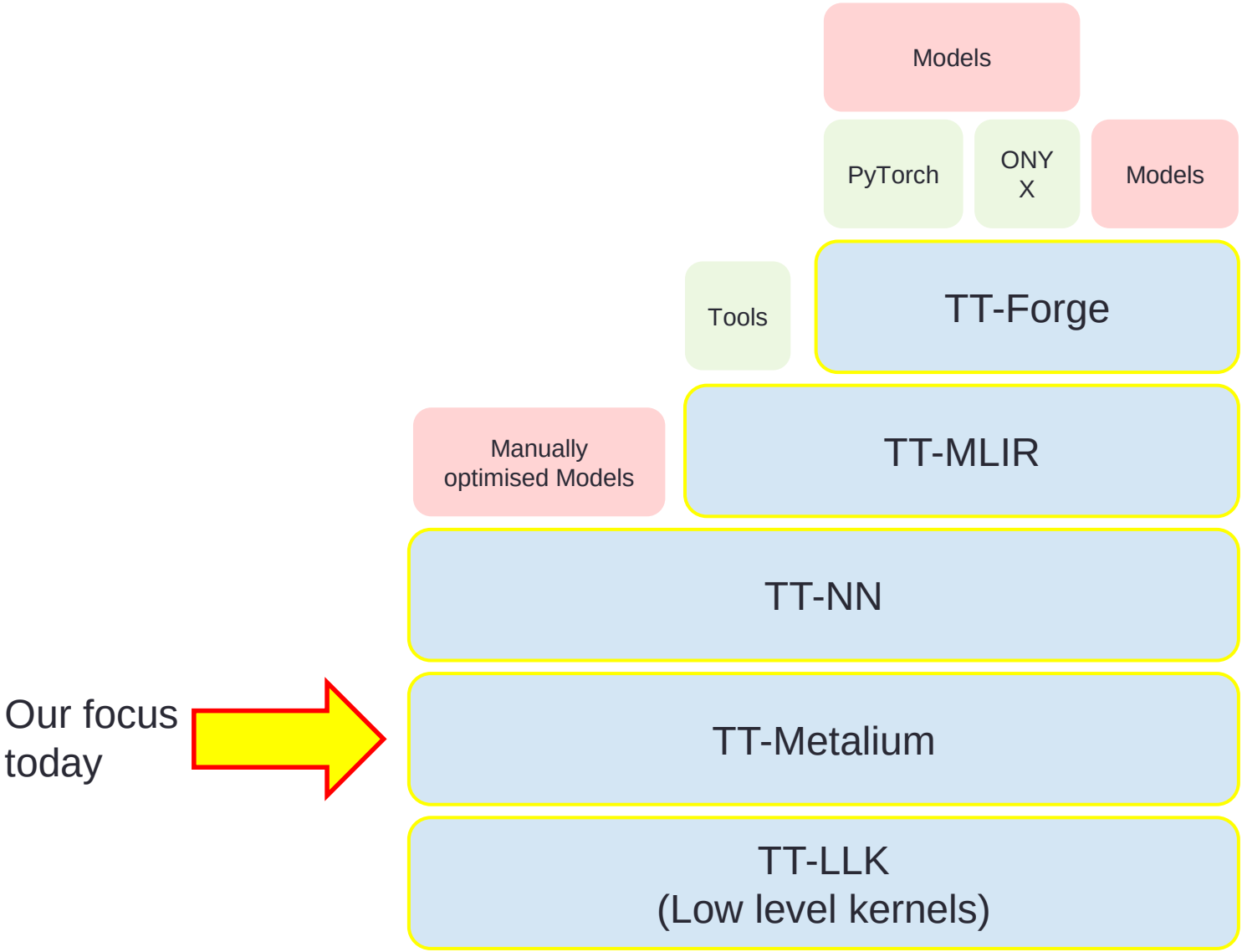
- Host code is written by the programmer
- Three kernels are written by the programmer:
  - Data movement in core
  - Compute cores
  - Data movement out core



- As we will discuss later, these can be replicated across Tensix units or individual kernels allocated on a unit by unit basis

- In the host code, each kernels path and name is provided
  - When the host code is launched then each kernel is first compiled and launched

TT-Metalium SDK



Our focus today

ML graph compiler
https://github.com/tenstorrent/tt-forge-fe

Compiler support for Tensix
https://github.com/tenstorrent/tt-mlir

Optimised ML building block operations
https://github.com/tenstorrent/tt-metal

Direct programming SDK
https://github.com/tenstorrent/tt-metal

Low level interaction with hardware
https://github.com/tenstorrent/tt-llk