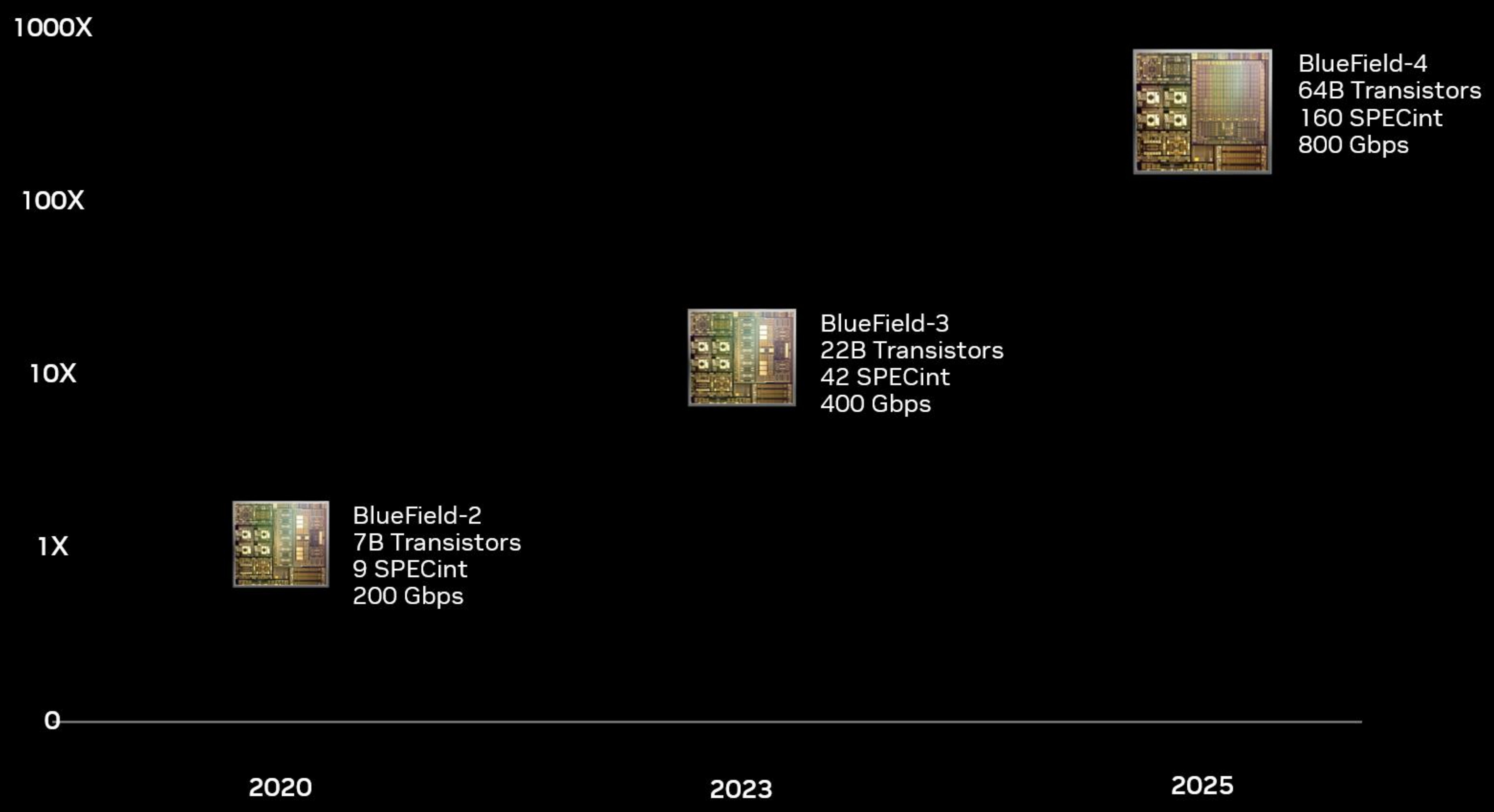
Offloading Tag Matching Logic to the Data Path Accelerator (DPA)

Salvatore Di Girolamo, Rami Nudelman, Jerónimo Sánchez García, Gil Bloch

UCF 2023

NVIDIA BlueField DPU Roadmap

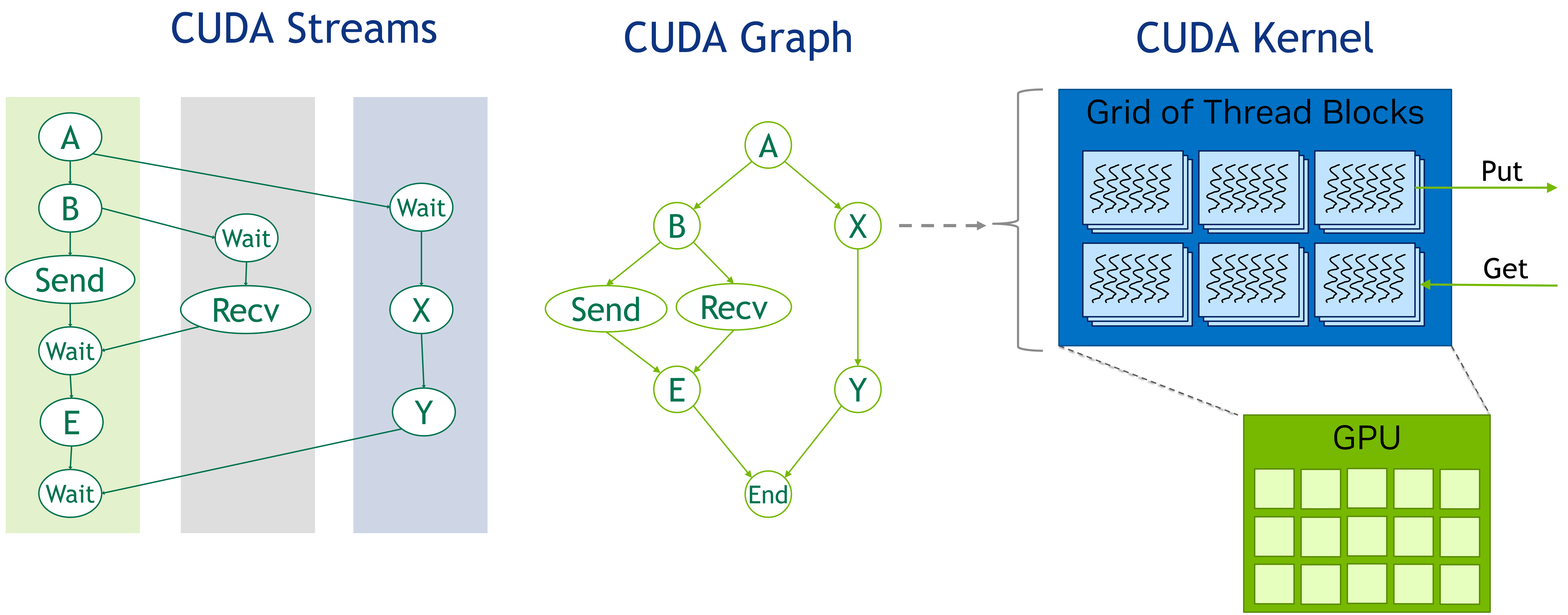
Exponential growth in data center infrastructure processing



NVIDIA

“GPU Centric” Communication

Communication Integrated with the GPU Work Scheduling Model



Can we offload MPI tag matching to SmartNICs?

Full offload of tag-matching and handling of unexpected messages.

Removing CPU from the data-path between the network and the GPU.

MPI tag matching

- Tag matching is a fundamental operation needed to enable MPI point-to-point semantic.

INTRODUCING NVIDIA BLUEFIELD-3 DPU

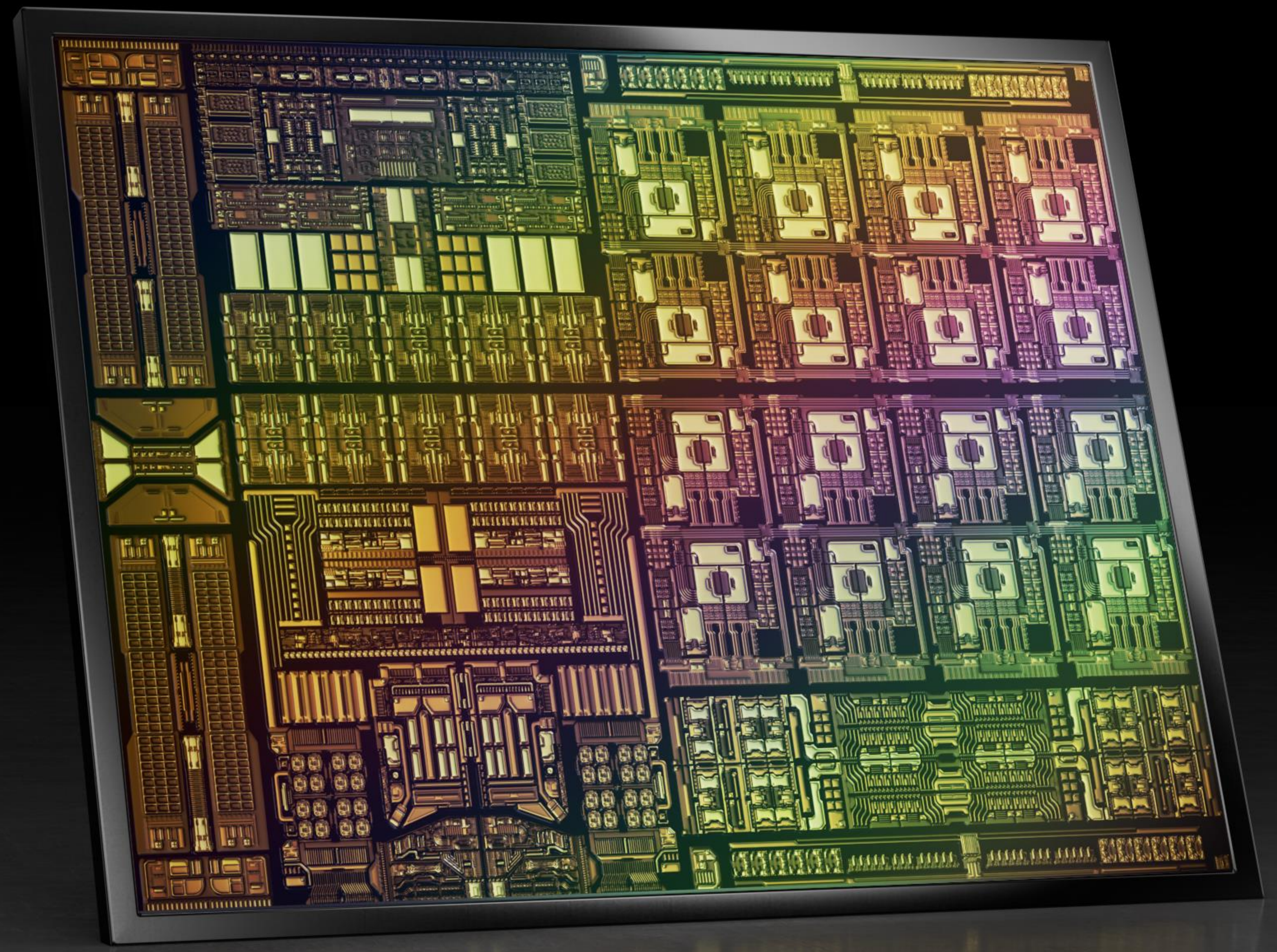
400Gb/s Data Processing Unit

Offloads and accelerates data center infrastructure

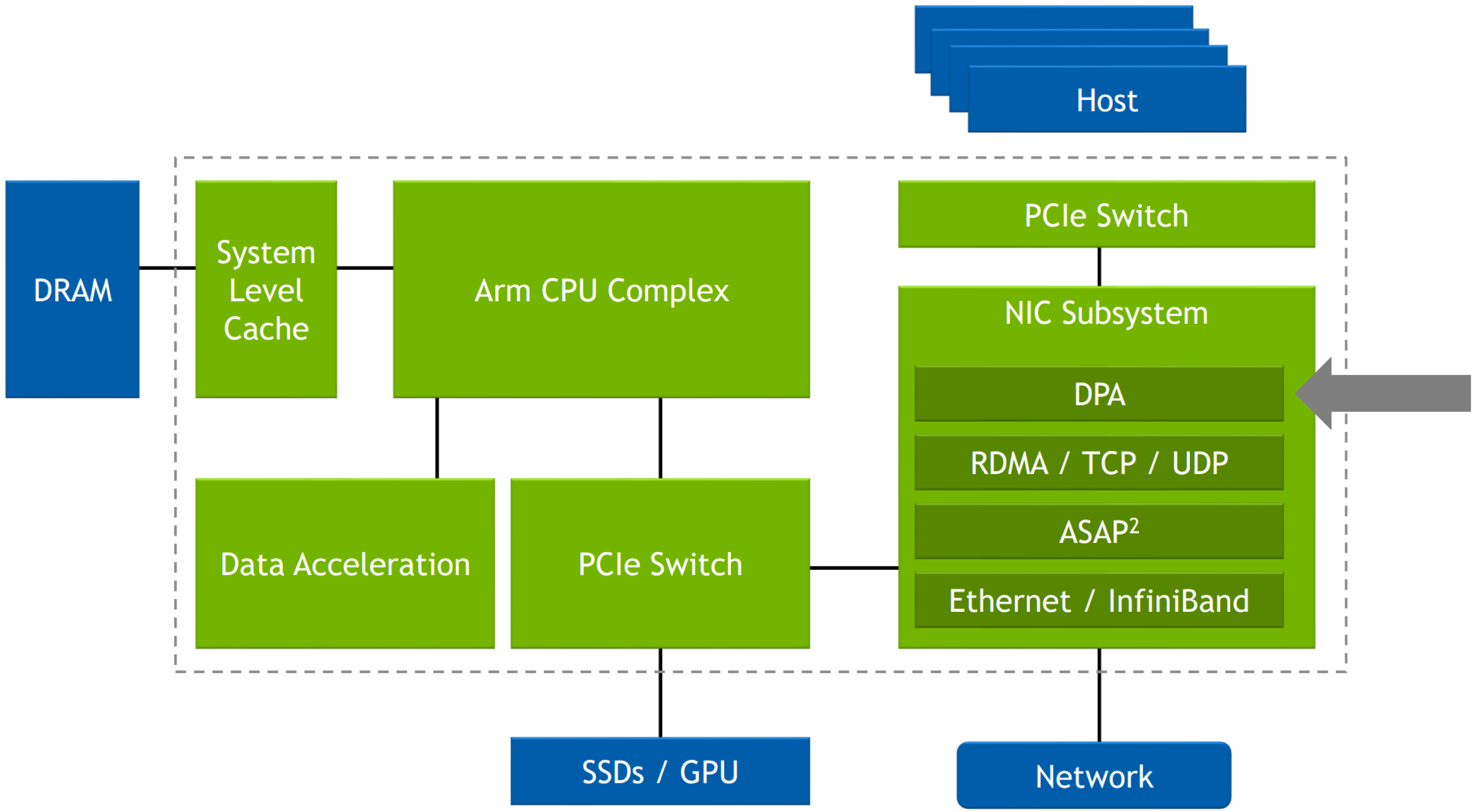
Isolates applications from control and management plane

16 x Arm A78 cores

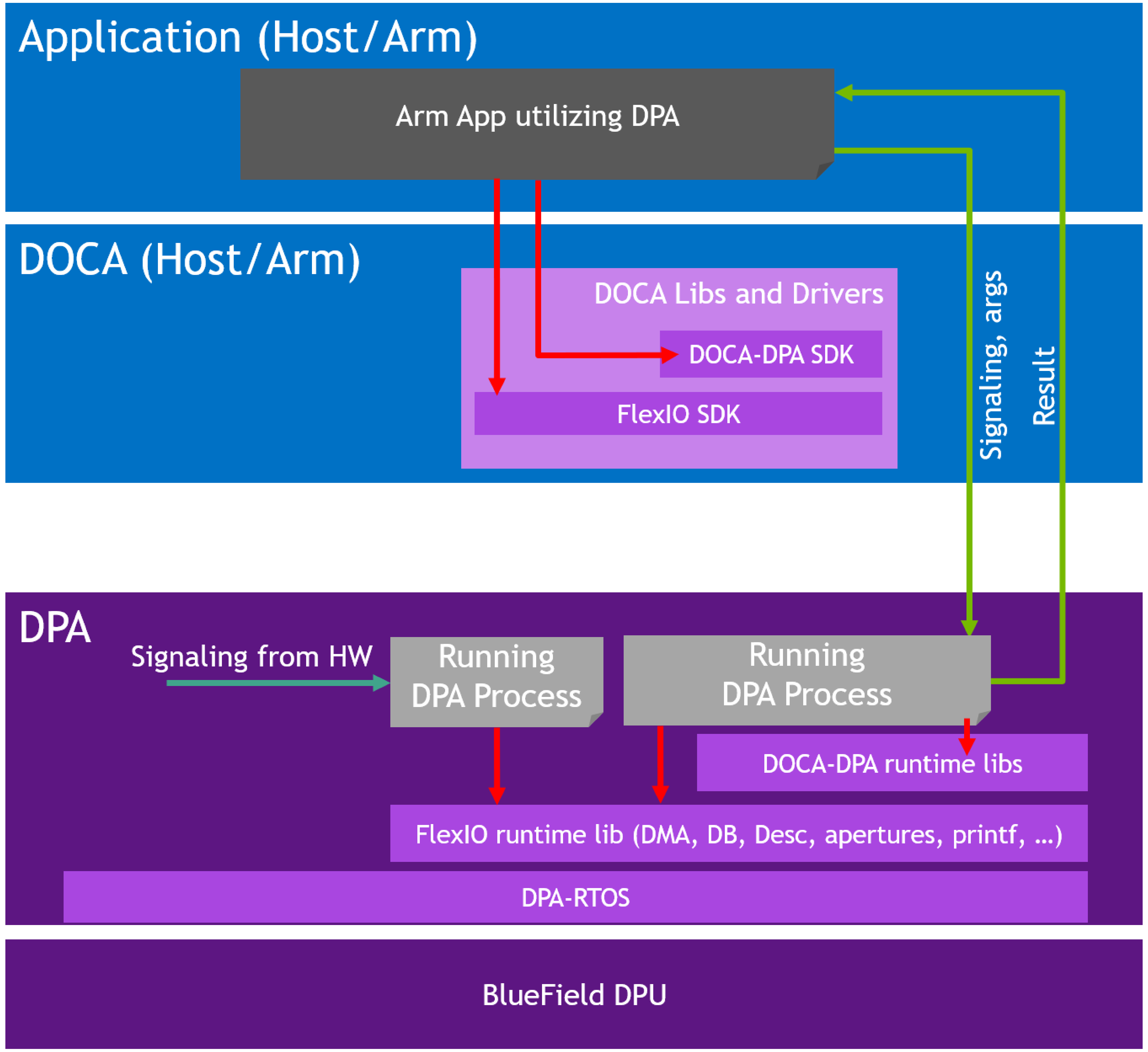
Datapath accelerator (DPA) – 16x Cores, 256 Threads



NVIDIA DPU System Architecture

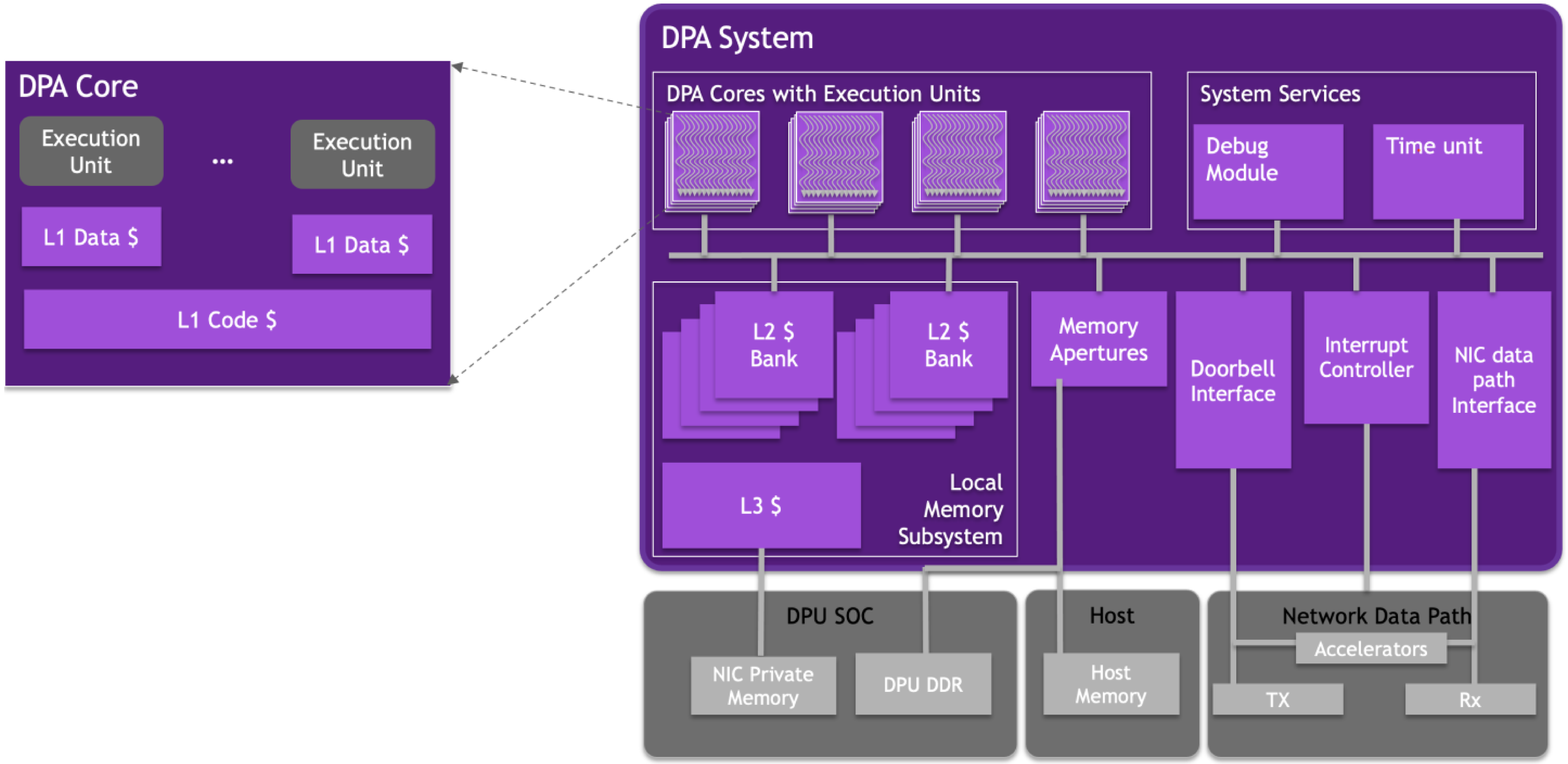


DPA Subsystem Overview



DPA Subsystem Overview

- High degree of programmability.
- 16 cores, each with16 threads.
- NIC has direct access to DPA caches.
- DPA has direct access to NIC interrupts and doorbells.
- Full access to on-NIC accelerators.
- Hardware-accelerated scheduling.
- Run-to-completion scheduling model.
- Load/store access to host memory via memory windows.
- RTOS enforces privileges and isolation between different processes.



DOCA Software Stack

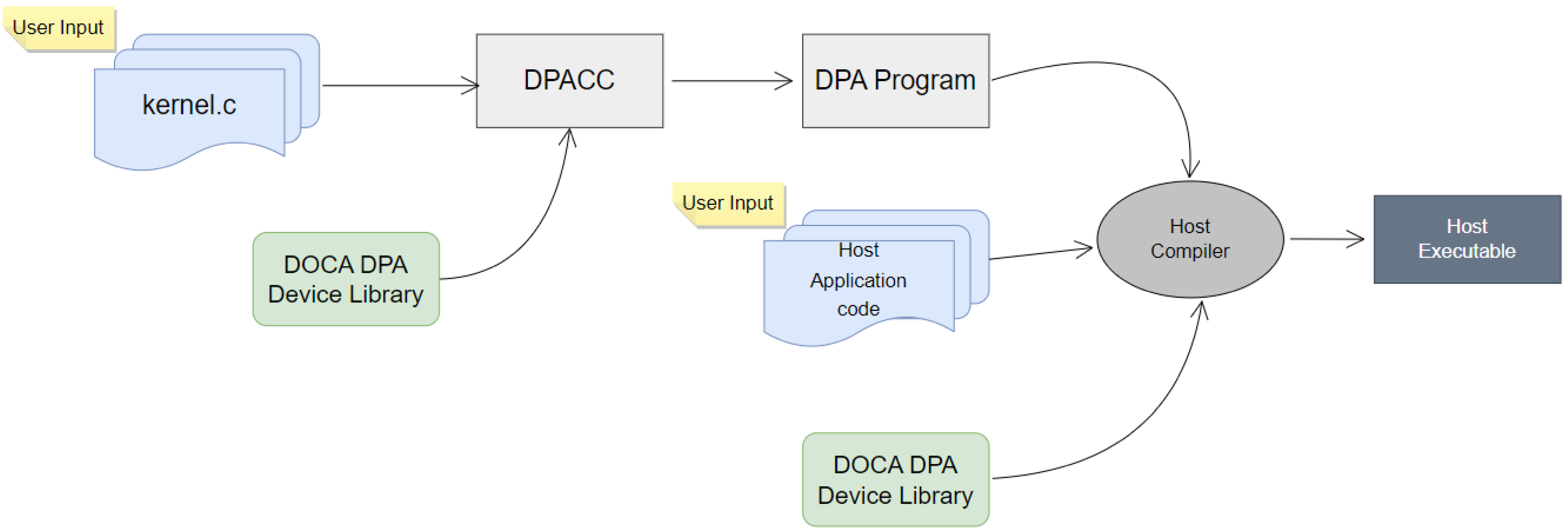
DPA Programming Flow

1. User writes DPA device code on the host (C language).

2. Use DPACC to build a DPA program. Inputs:
    1. Device code.
    2. DOCA DPA device library.

3. Build host executable using host compiler. Inputs:
    1. Compiled DPA program.
    2. Host-side user application.
    3. DOCA DPA host library.

4. DPA code can be triggered by:
    1. RPC calls from the host-side application.
    2. Network events



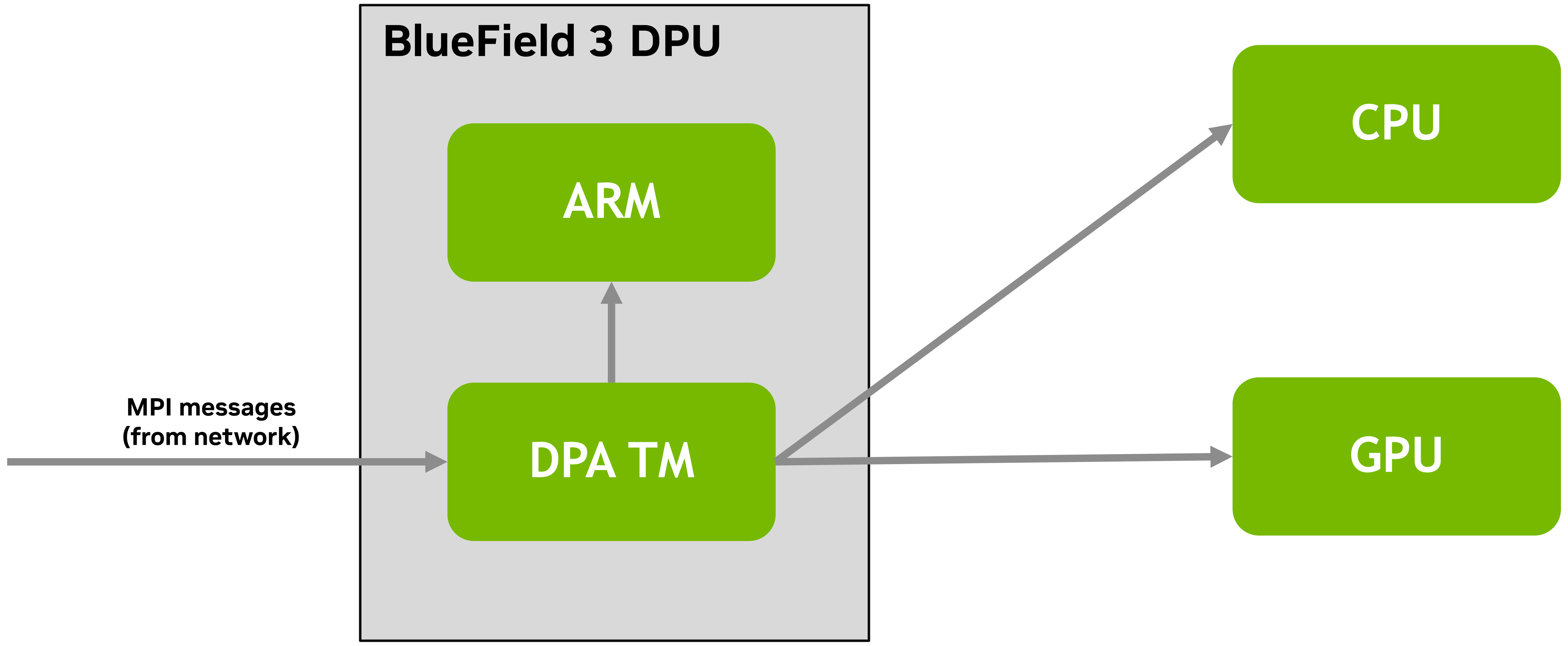
FlexIO intro

Event handlers

- The event handler executes its function each time an event occurs.
- **Event:** CQE received when the CQ is in the armed state.
- The event triggers an internal DPA interrupt that activates the event handler.
- When the event handler is activated, it is provided with a user-defined argument (e.g., pointer to user-defined sw context).

```
// Device code
__dpa_global__ myFunc(void *myArg){
   struct my_db *db = (struct my_db *)myArg;
   get_completion(db->myCq)
   work();
   arm_cq(myCq);
   return;
}

// Host code
main() {

   /* Load the application code into the DPA */
   flexio_process_create(device, application, &myProcess);

   /* create event handler to run my_func with my_arg */
   flexio_event_handler_create(myProcess, myFunc, myArg, &myEventHandler);

   /* Associate the event hanlder with a specific CQ */
   create_cq(&myCQ, ... , myEventHandler)


   /* start the event handler */
   flexio_event_handler_run(myEventHandler)
   ...
}
```

DPA-offloaded MPI tag matching



Goal: fully offloading MPI tag matching semantic, including support for unexpected messages.

Can we offload MPI tag matching to the DPA?

↓

Can we extract parallelism from MPI tag matching?

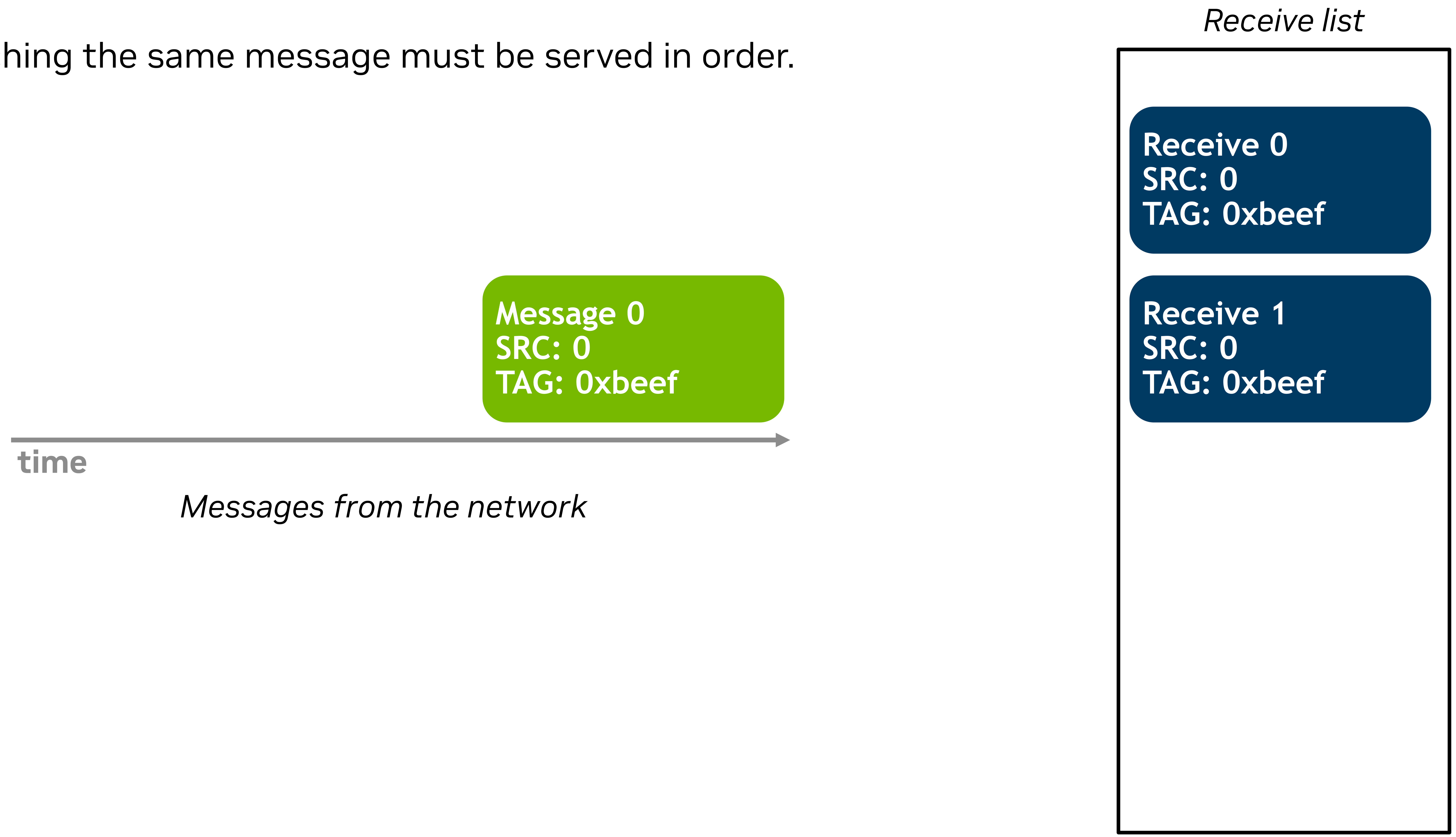
DPA-based MPI tag matching

Overview: Receiving an expected eager message

MPI tag matching constraints

Receive order

Constraint 1.
Receives matching the same message must be served in order.



MPI tag matching constraints

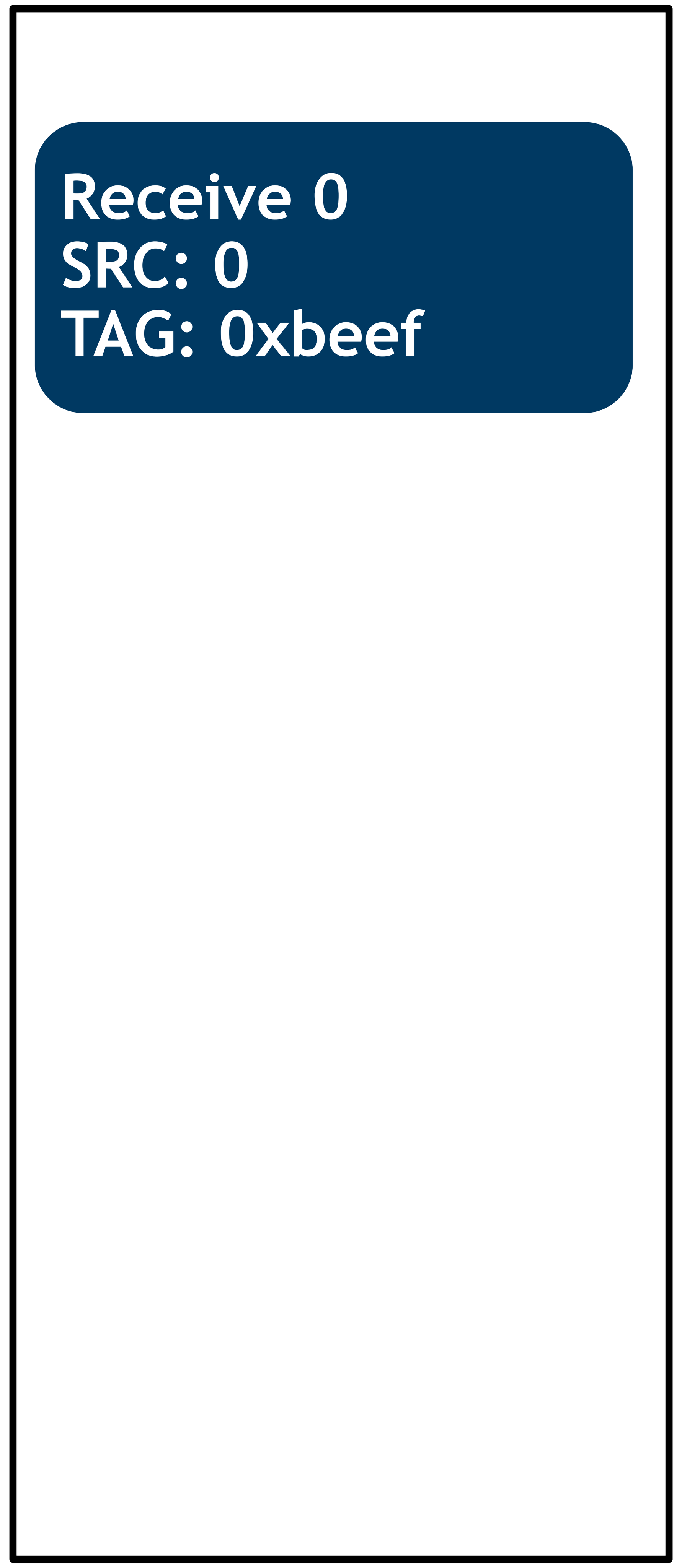
Non-overtaking messages

Constraint 2.

Messages from the same sender cannot overtake each other.

Messages from the network

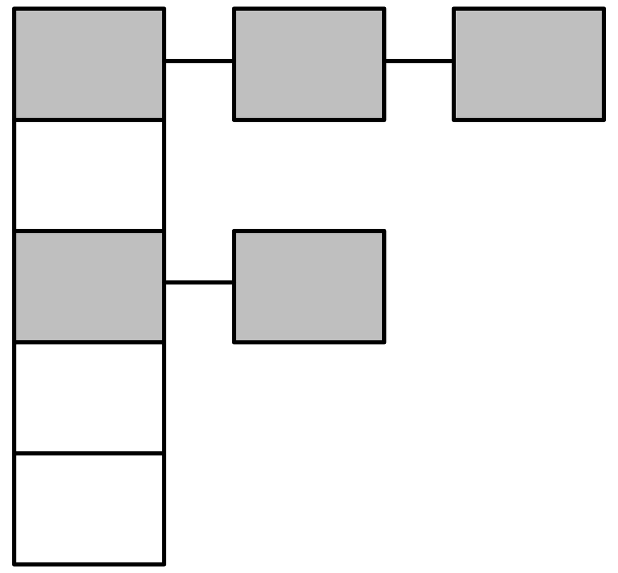
Receive list



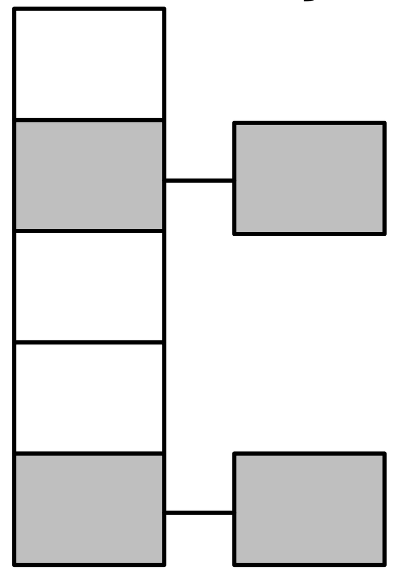
DPA-based MPI tag matching

Indexing receives

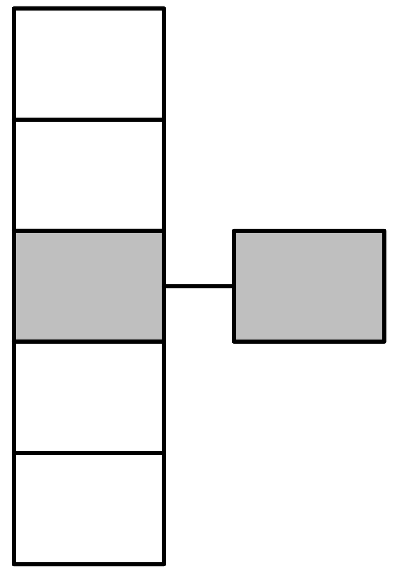
Index: No wildcards [Hash table]



Index: Only SRC wildcard [Hash table]



Index: Only TAG wildcard [Hash table]



Index: Both SRC and TAG wildcards [Linked list]

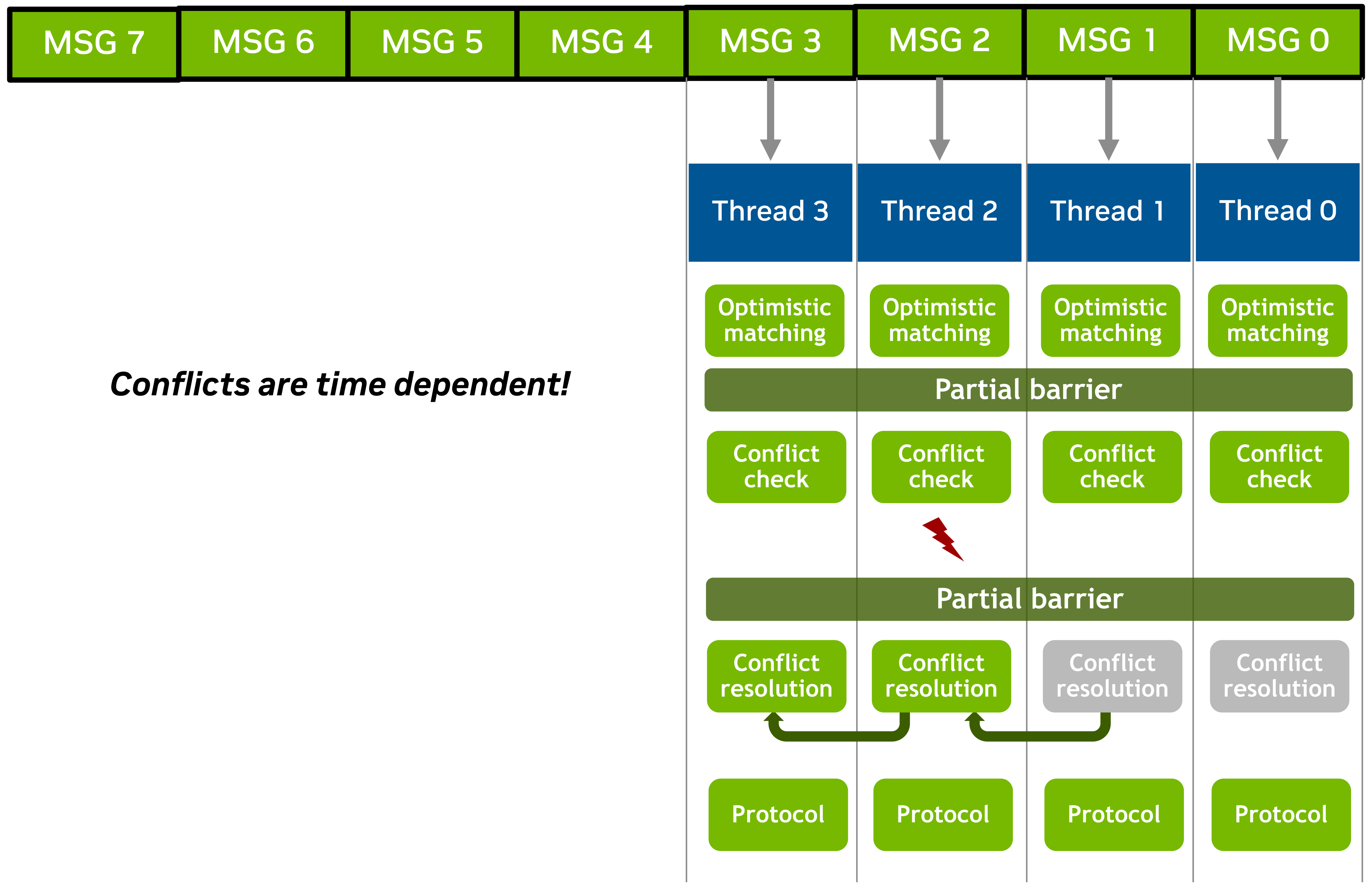
(1) Receive order
Within an index, constraint (1) must be enforced only between receives in the same bucket.

To enforce constraint (1) globally, we can match independently against the four indices and take the receive candidate that has been posted the earliest.

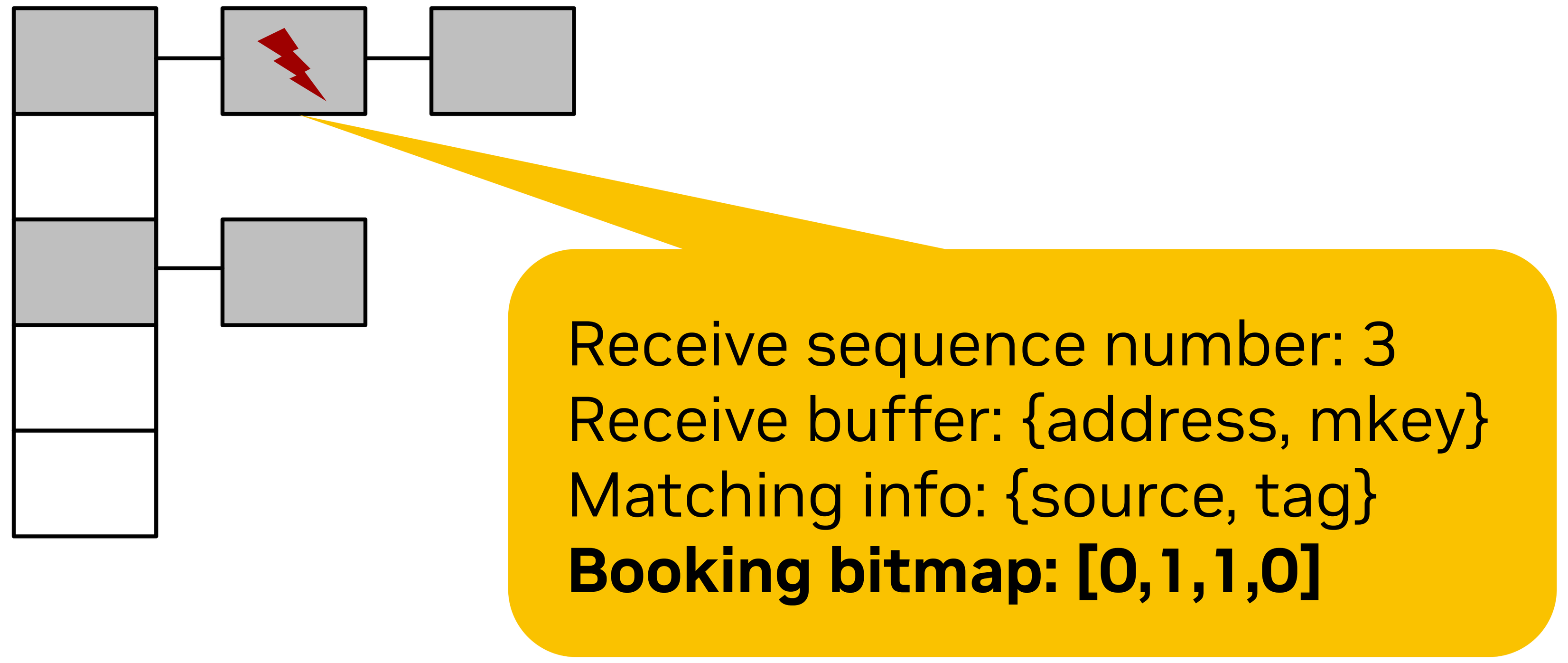
DPA-based MPI tag matching

Optimistic matching

Conflicts are time dependent!

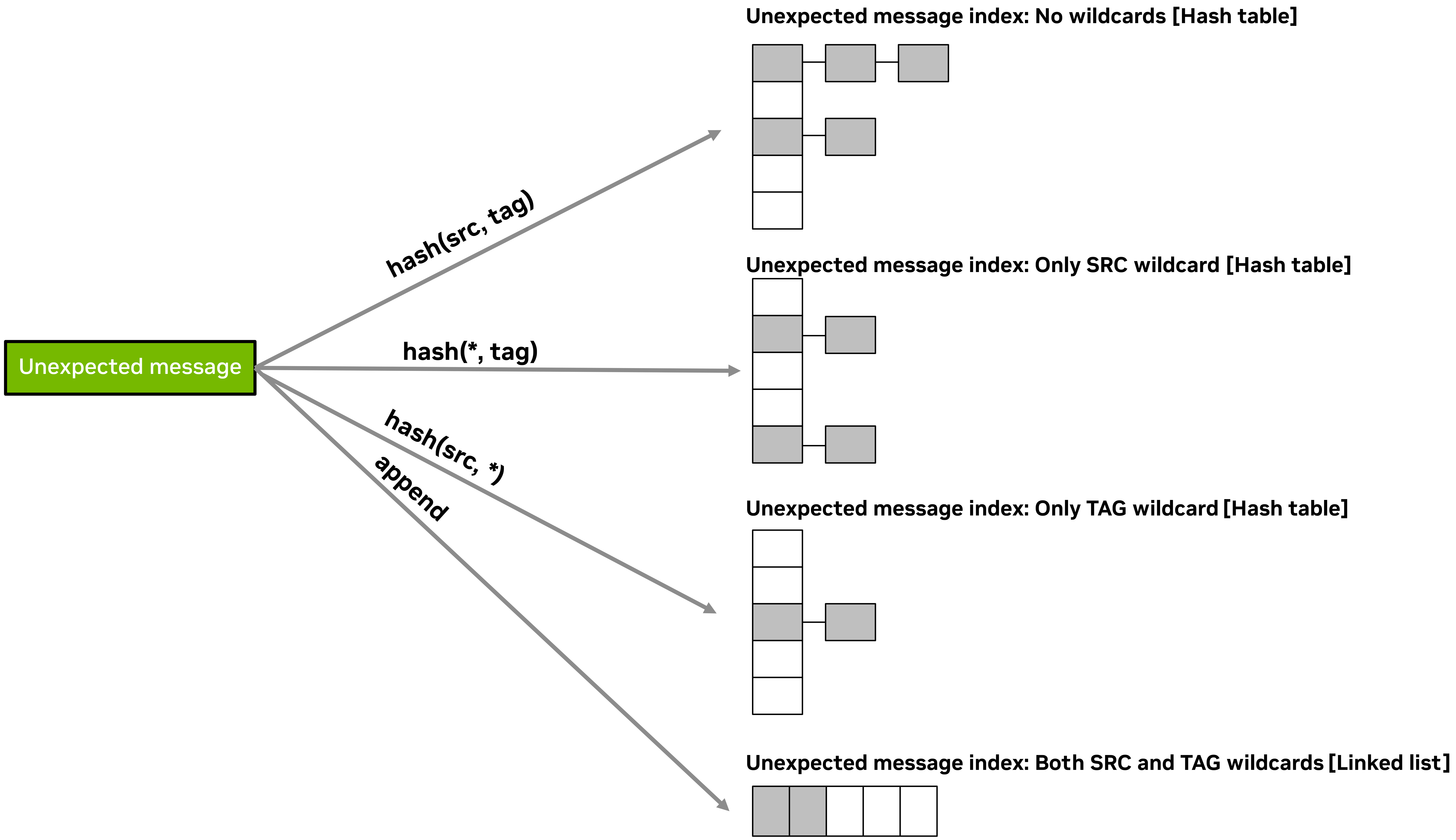


Index: No wildcards [Hash table]



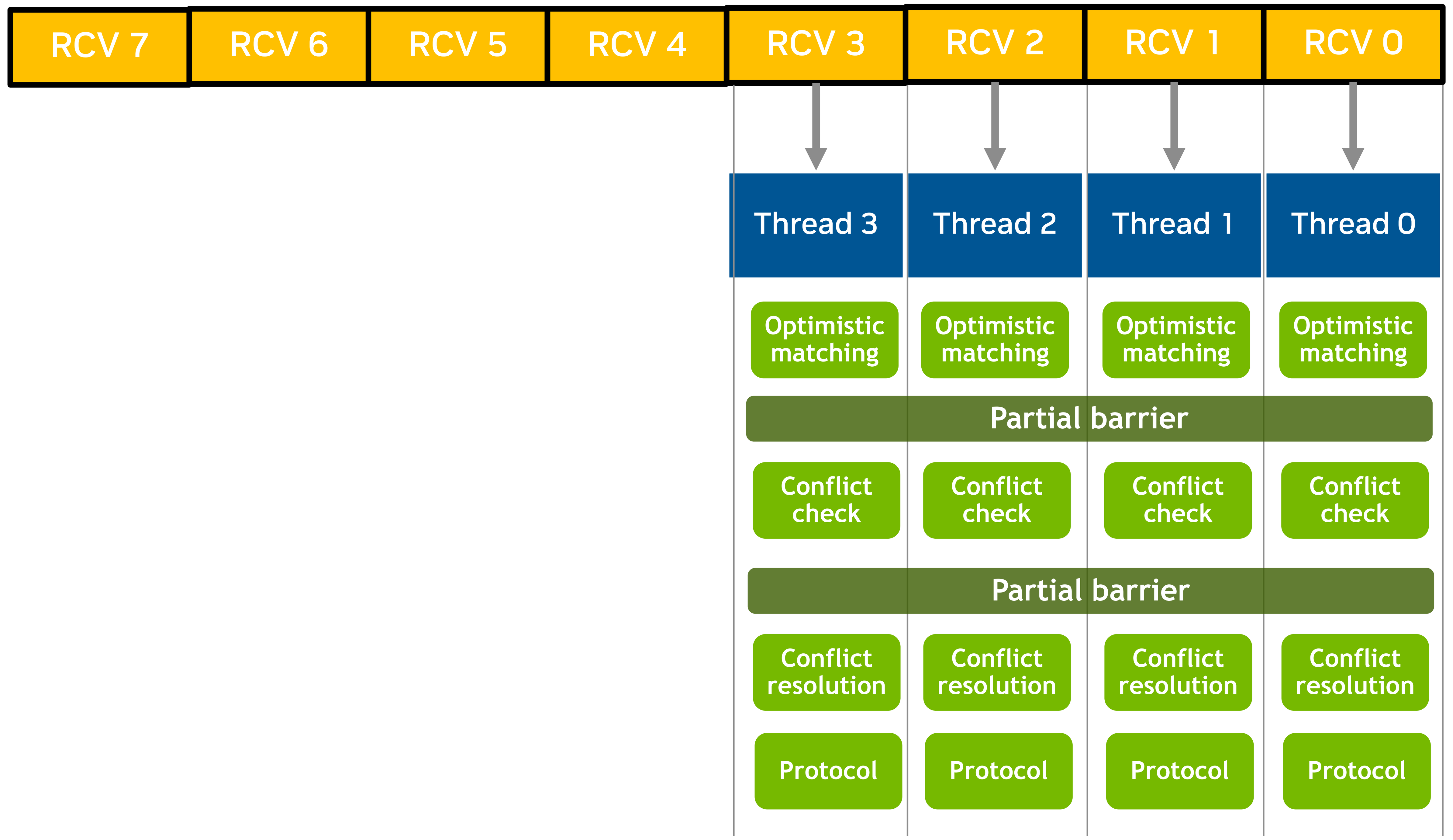
DPA-based MPI tag matching

Indexing unexpected messages



DPA-based MPI tag matching

Checking for unexpected messages

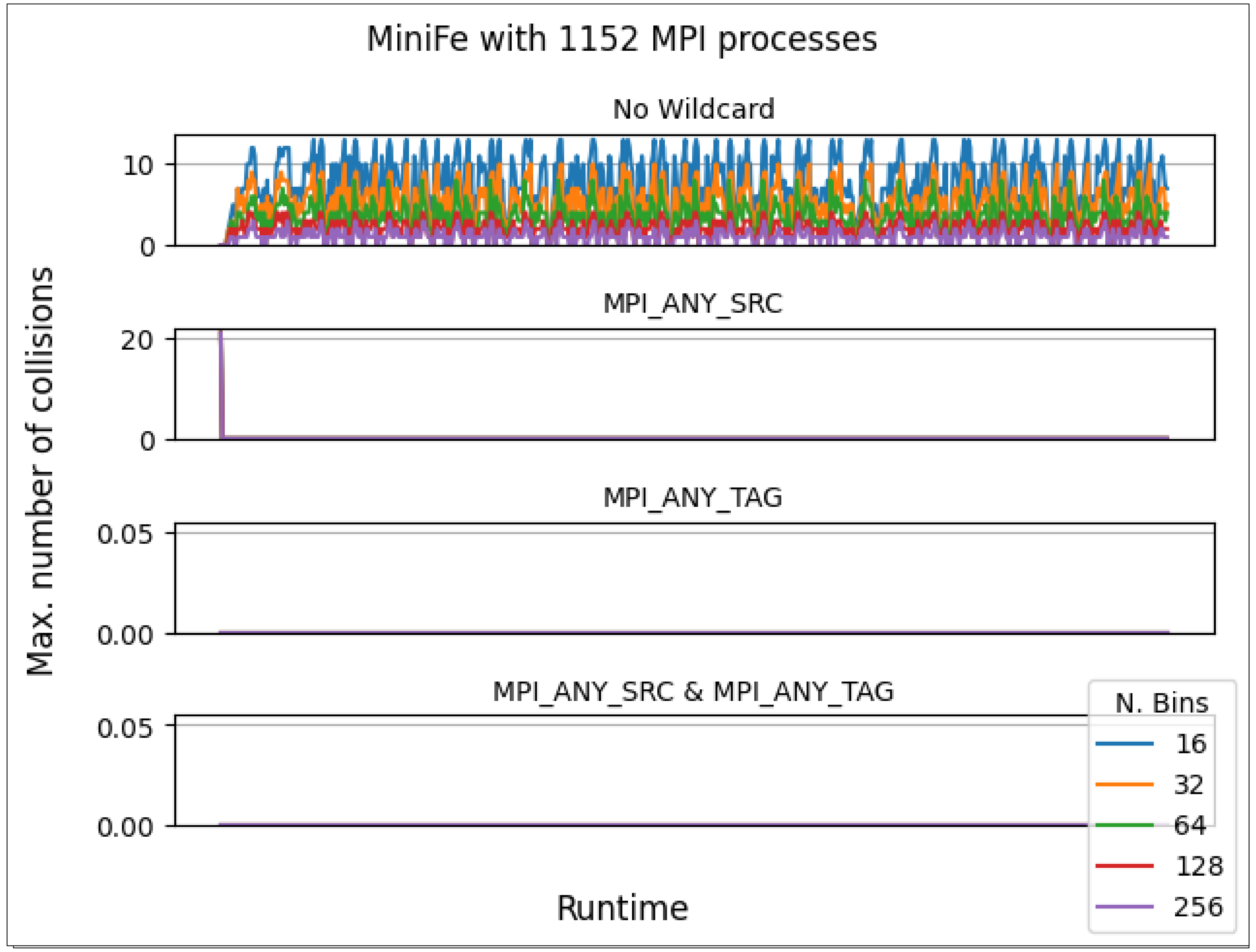


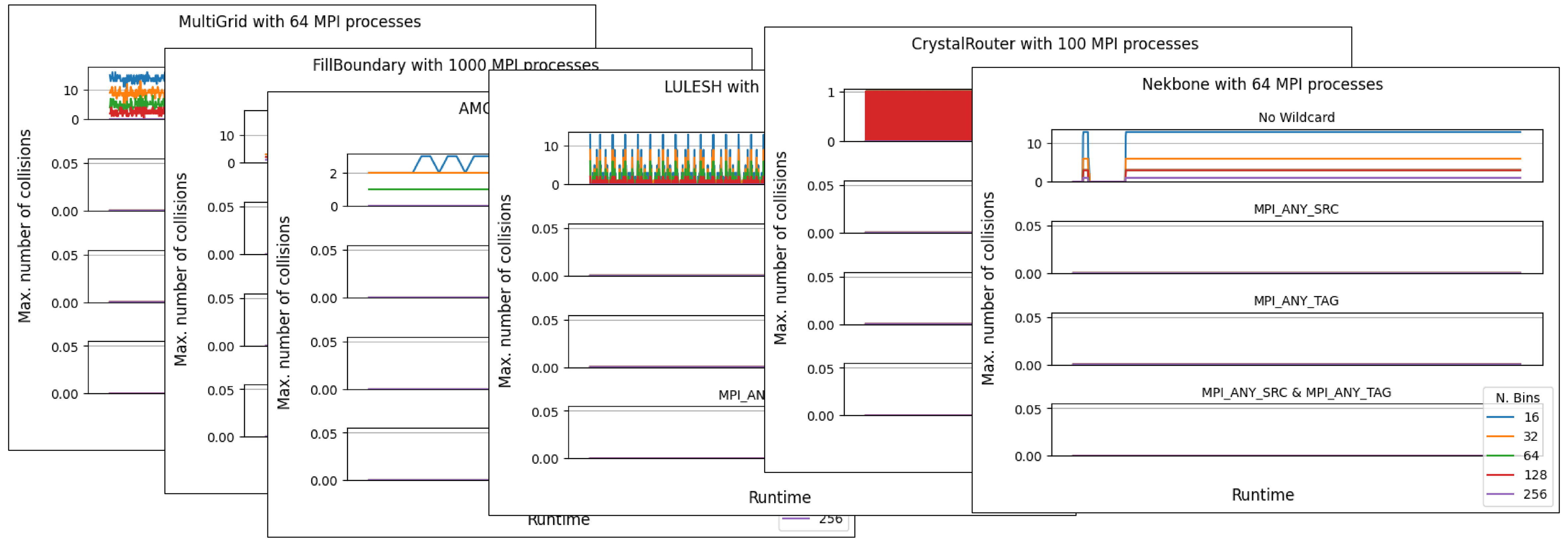
New receives can optimistically match unexpected messages similarly to the way new messages match receives.

If the optimistic matching phase concludes that there are no matching unexpected messages, the thread can move to the indexing of the receive.

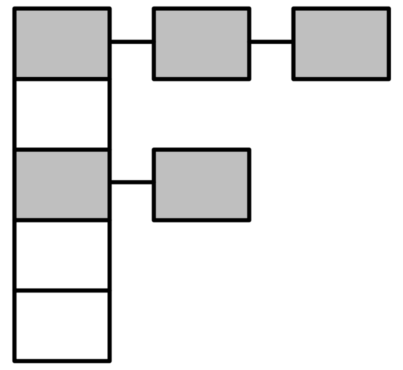
DPA-based MPI tag matching

Application analysis

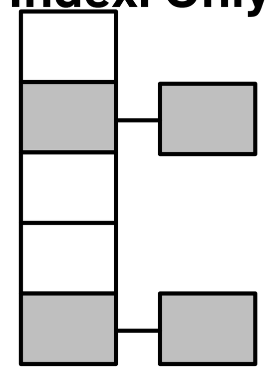




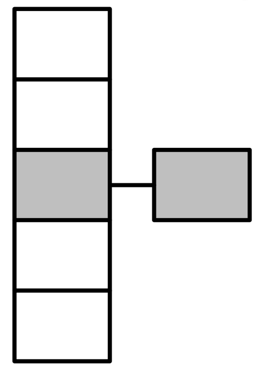
Index: No wildcards [Hash table]



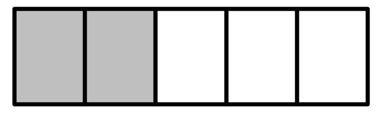
Index: Only SRC wildcard [Hash table]



Index: Only TAG wildcard [Hash table]

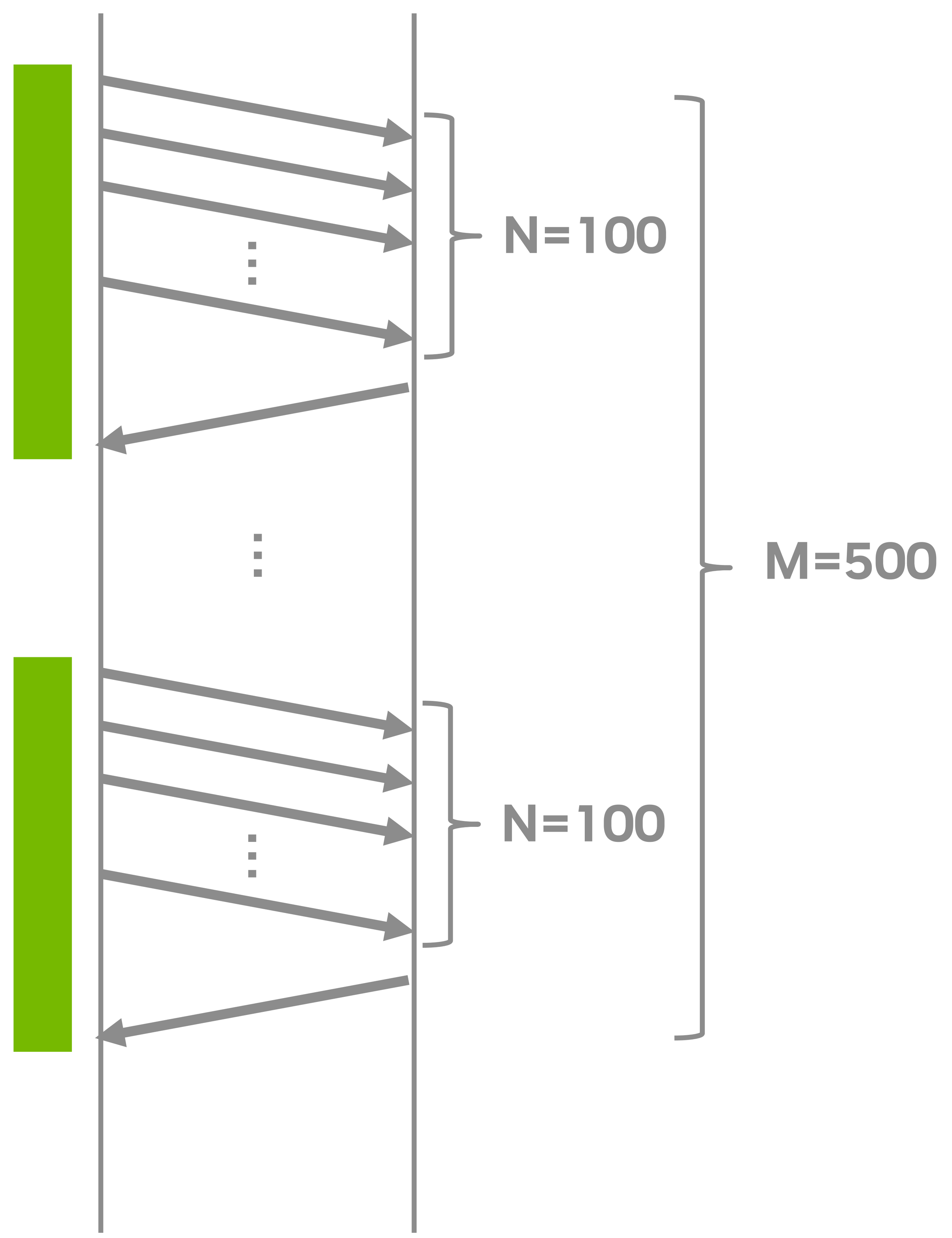


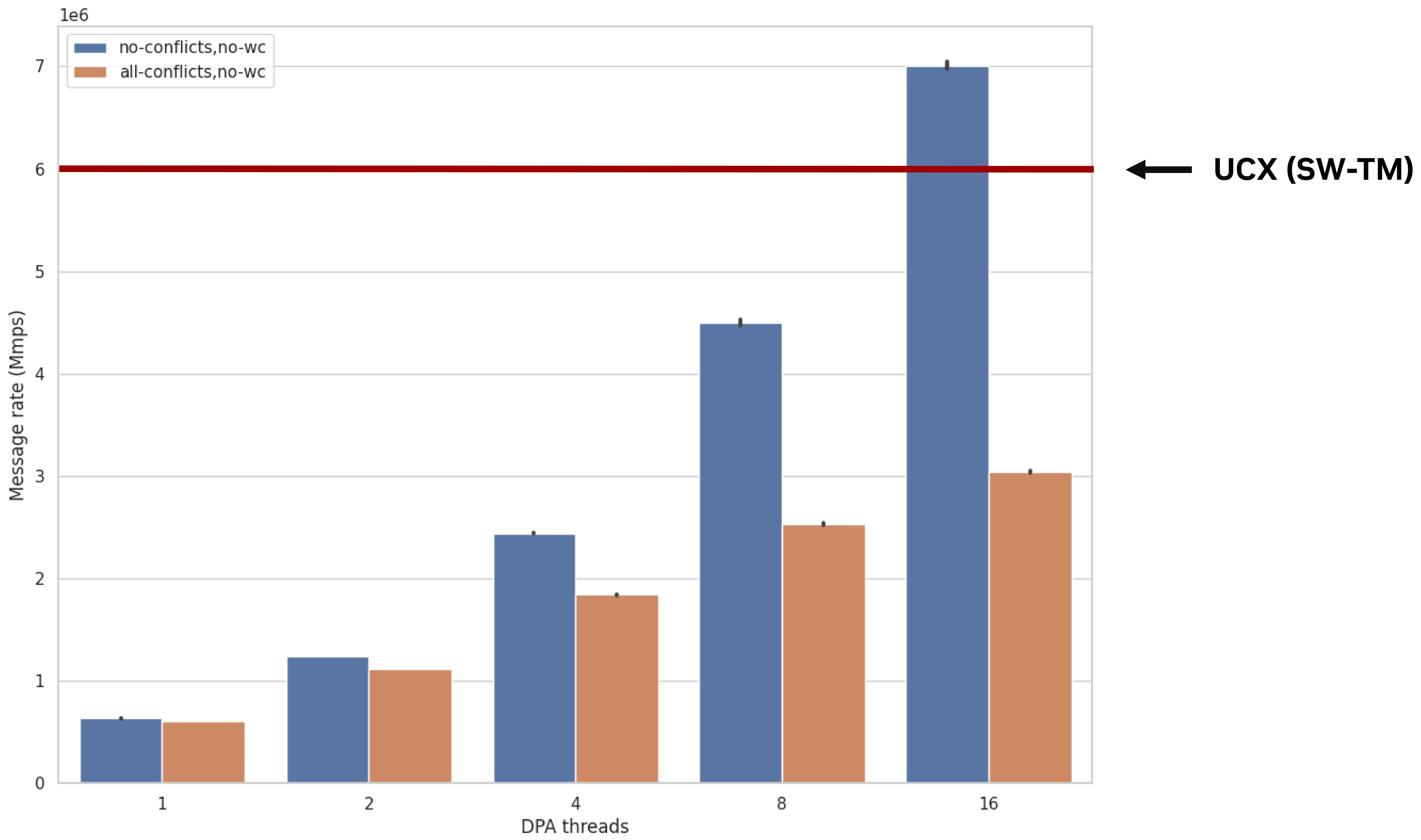
Index: Both SRC and TAG wildcards [Linked list]



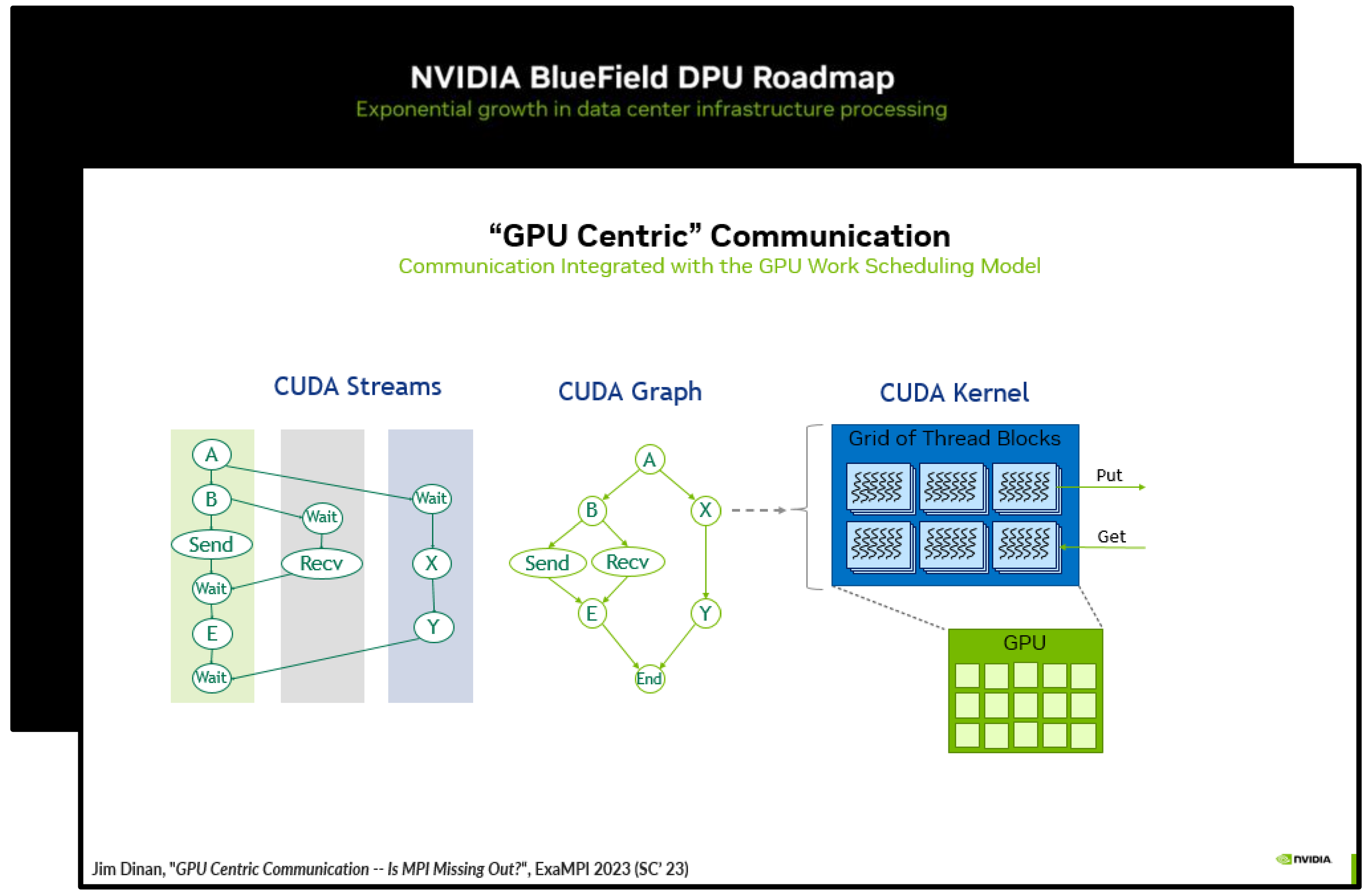
DPA-based MPI tag matching

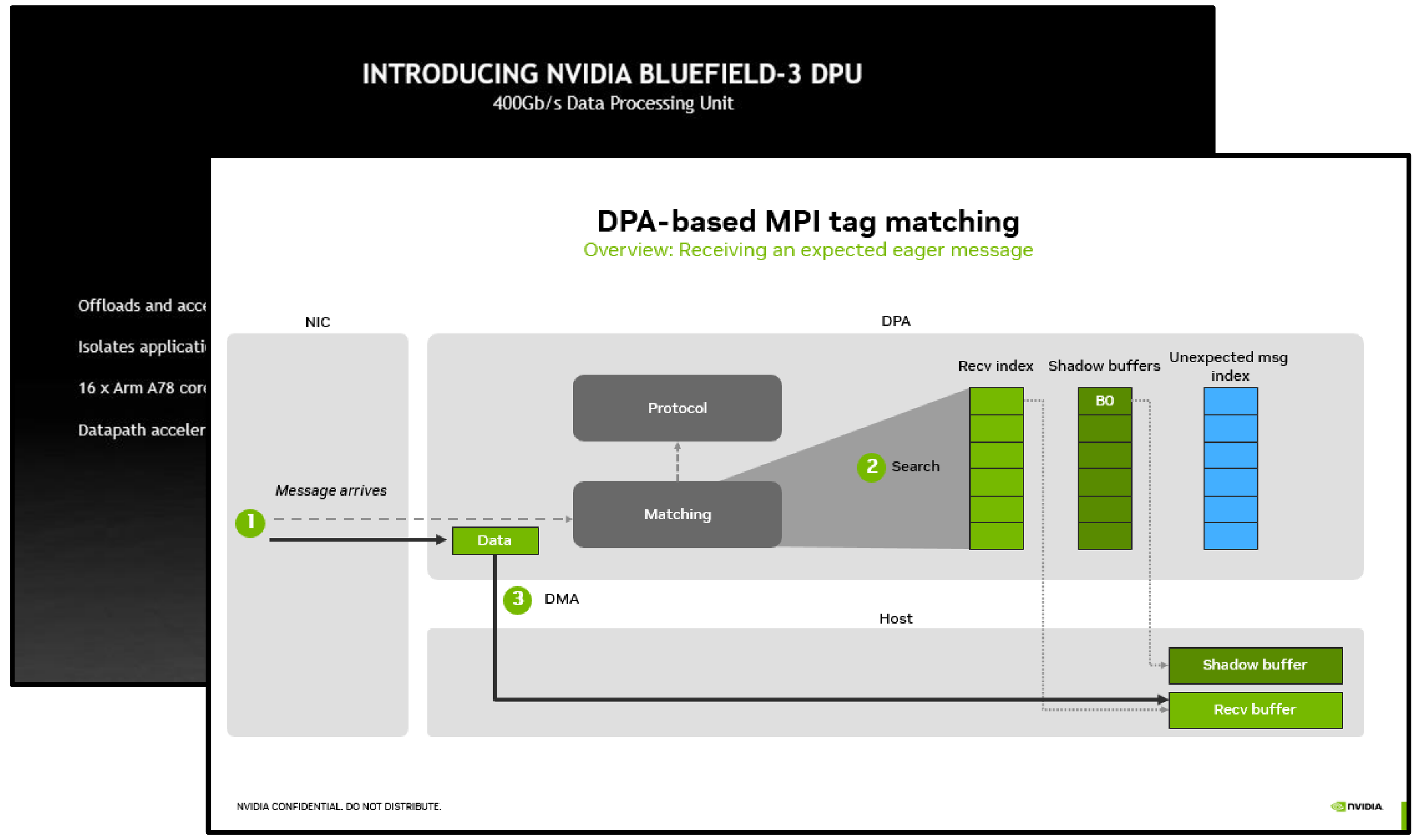
Preliminary results

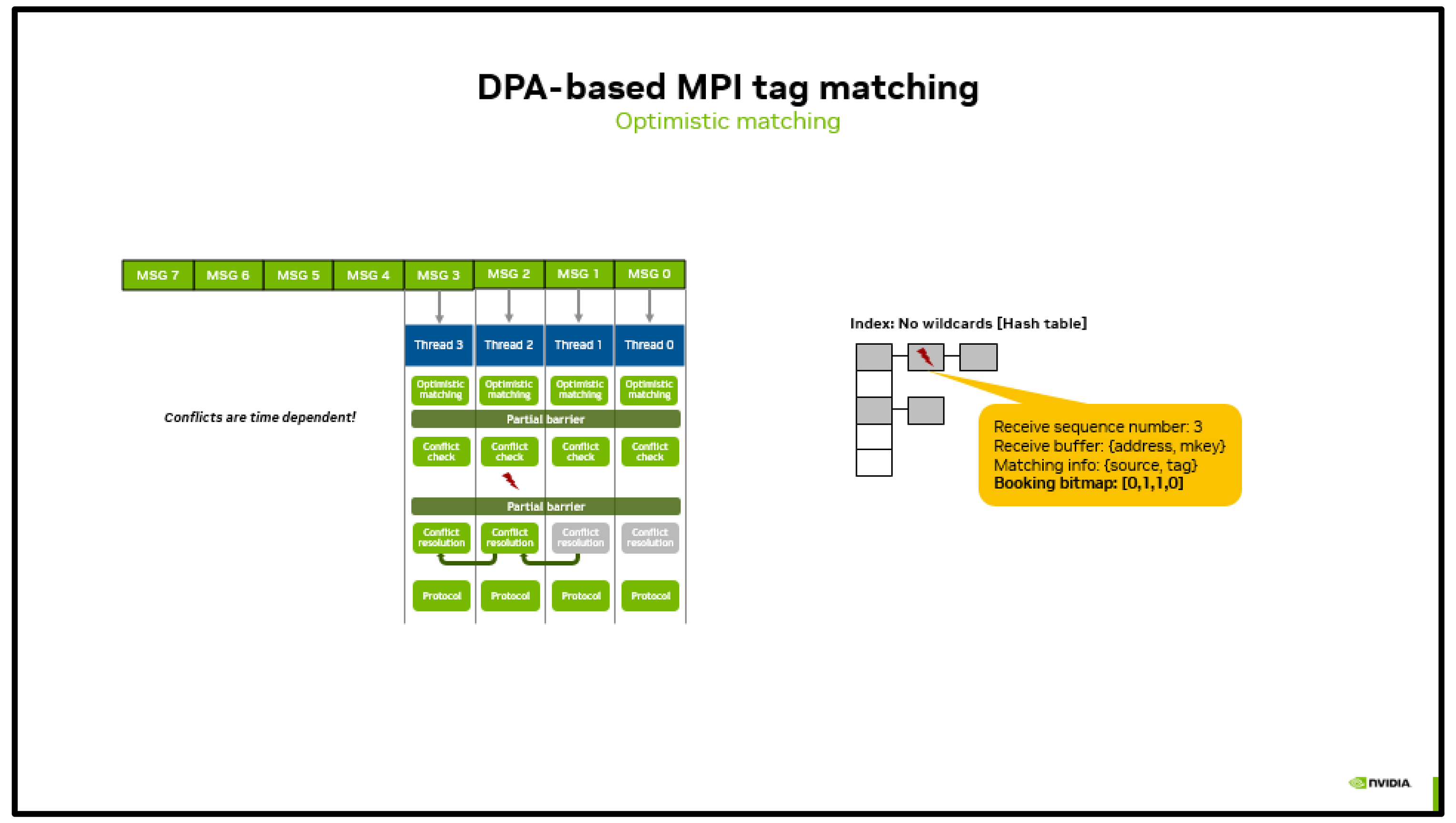


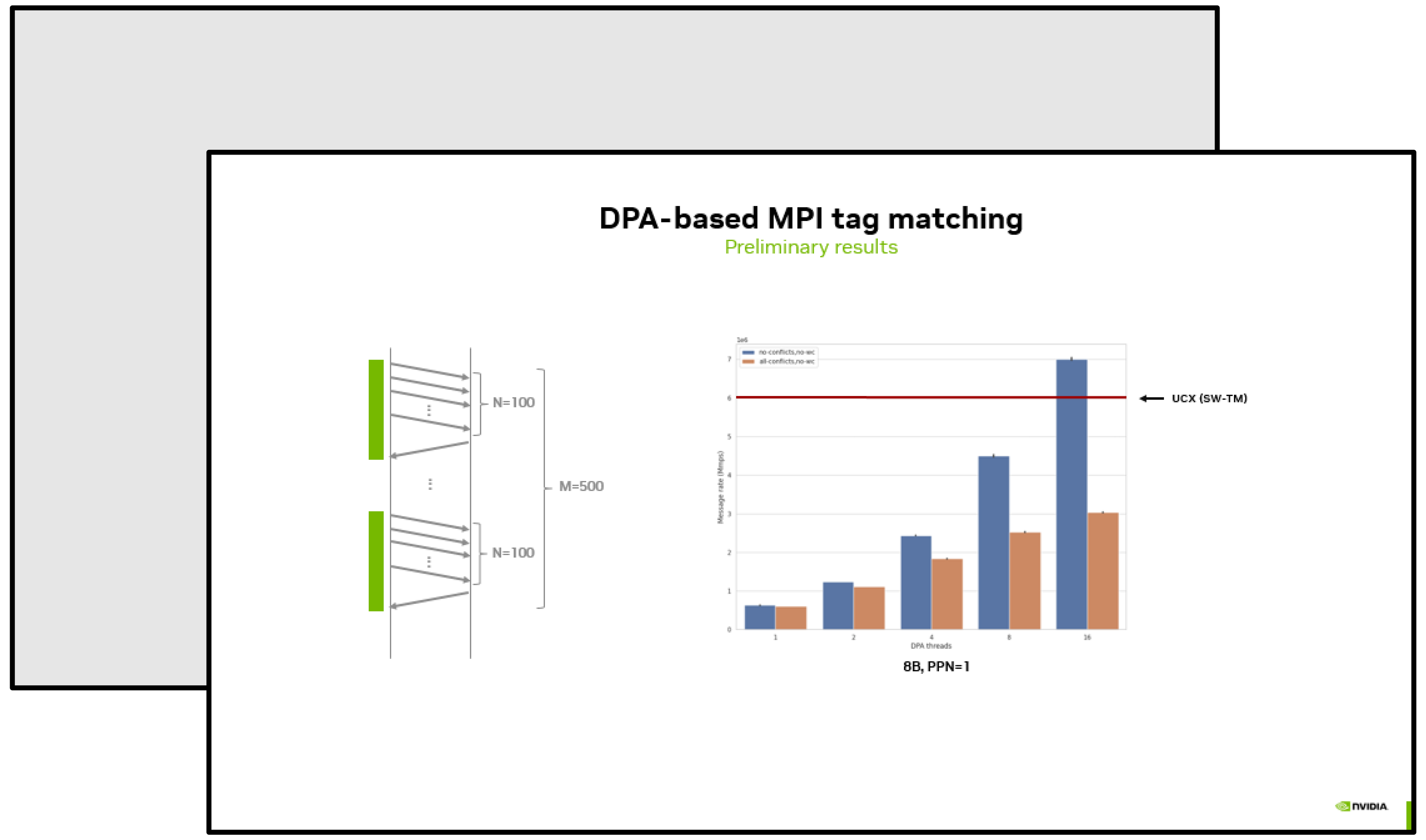


8B, PPN=1


NVIDIA BlueField DPU Roadmap
Exponential growth in data center infrastructure processing
“GPU Centric” Communication
Communication Integrated with the GPU Work Scheduling Model
CUDA Streams
CUDA Graph
CUDA Kernel
Grid of Thread Blocks
Put
Get
GPU
Send
Recv
Wait
End
Jim Dinan, "GPU Centric Communication -- Is MPI Missing Out?", ExaMPI 2023 (SC' 23)


INTRODUCING NVIDIA BLUEFIELD-3 DPU
400Gb/s Data Processing Unit
DPA-based MPI tag matching
Overview: Receiving an expected eager message
NIC
DPA
Message arrives
Protocol
Matching
Data
Search
DMA
Recv index
Shadow buffers
Unexpected msg index
Host
Shadow buffer
Recv buffer
NVIDIA CONFIDENTIAL. DO NOT DISTRIBUTE.


DPA-based MPI tag matching
Optimistic matching
MSG 7
MSG 6
MSG 5
MSG 4
MSG 3
MSG 2
MSG 1
MSG 0
Thread 3
Thread 2
Thread 1
Thread 0
Partial barrier
Conflicts are time dependent!
Index: No wildcards [Hash table]
Receive sequence number: 3
Receive buffer: (address, mkey)
Matching info: (source, tag)
Booking bitmap: [0,1,1,0]


DPA-based MPI tag matching
Preliminary results
N=100
M=500
N=100
UCX (SW-TM)
8B, PPN=1

NVIDIA