Dresden University of Technology

- Bill Williams[1], Bert Wesarg[1+2], Jonathan Madsen[3], Ammar Elwazir[3]
- [1]IAK/ZIH/CIDS, TU Dresden, [2]GWT-TUD GmbH, [3]AMD

IPTW 2025, Stuttgart, 6./7. November 2025

Integrating ROCProfiler-SDK into Score-P: A Case Study

Dresden University of Technology

- What makes a tools interface good or bad?
- What’s interesting about rocprofiler-sdk?
- What’s new in OTF2, and why?
- How does it all come together in an adapter?
- Where does everything stand now?

Dresden University of Technology

- What makes a tools interface good?
  - Performance
  - Functionality
  - Maintainability
- ...and bad?
  - Limiting assumptions

Dresden University of Technology

- Performance: not just about the internal event measurement
  - Don’t make tools pay for what they don’t need
  - Don’t make tools pay for what can be done cheaper internally

Dresden University of Technology

- Functionality: what matters in 2025?
  - Multi-tool support, with flexible initialization
    - One or more “always-on” tools + tools like Score-P + LD_PRELOAD-based tools
    - Users want it to “just work”
  - Support everything reasonable
    - Different layers of the infrastructure
    - Flexible configuration of measurement
    - Best effort in-order delivery of events
    - API wrapping and introspection is sufficient for *anything*, but not always *necessary*

Dresden University of Technology

- Maintainability: the devil in the details
  - User-level APIs are churning rapidly these days
    - Memory management
    - Kernel launch
    - Data transfer
  - …but the fundamental abstractions stay unchanged
- Let tools specify that they’re interested in these higher-level abstractions!

Dresden University of Technology

- Ways to cause problems as a tools interface
  - Assume that you’re the one true source of all events, and no other event streams need to be integrated
  - Assume that only one tool is present
  - Do the bare minimum: “it’s technically possible, everything else is a tools problem”
  - Assume that data formats don’t matter
    - Everyone will be profiling so clocks don’t matter?
    - Scale doesn’t matter, so ordering and filtering don’t matter?

Dresden University of Technology

- rocprofiler-sdk: how does it measure up?
  - **Performance**
    - **Pay for what you use**
  - Functionality
    - Multi-tool support and isolation via sessions
    - Standard callback+buffer-based APIs
    - Correlation IDs as first-class citizen
  - Maintainability
    - Tools can register for semantic events
    - Tools can use reflection-style interface to examine API calls
    - Direct struct-style access and inspection still possible

Dresden University of Technology

- rocprofiler-sdk: how does it measure up?
  - Performance
    - Pay for what you use
  - **Functionality**
    - **Multi-tool support and isolation via sessions**
    - **Standard callback+buffer-based APIs**
    - **Correlation IDs as first-class citizen**
  - Maintainability
    - Tools can register for semantic events
    - Tools can use reflection-style interface to examine API calls
    - Direct struct-style access and inspection still possible

Dresden University of Technology

- rocprofiler-sdk: how does it measure up?
  - Performance
    - Pay for what you use
  - Functionality
    - Multi-tool support and isolation via sessions
    - Standard callback+buffer-based APIs
    - Correlation IDs as first-class citizen
  - **Maintainability**
    - **Tools can register for semantic events**
    - **Tools can use reflection-style interface to examine API calls**
    - **Direct struct-style access and inspection still possible**

Dresden University of Technology

- rocprofiler-sdk: what's new and interesting?
  - Code object load/unload events
    - Process kernel info once and only once
    - Reduce data volume in kernel launch events
  - Kernel dispatch, memory copy as fundamental abstractions
    - No checking parameters/specific function semantics unless you need them
  - Forbidden to call user-level APIs from tool code
    - Avoids many types of races and deadlocks
    - Forces tools interface to provide equivalent solutions
      - *stream tracking API* to notify tools of the active stream for a call
      - *Memory kinds* encoded in alloc/free events so tools don't need to query pointer properties

Dresden University of Technology

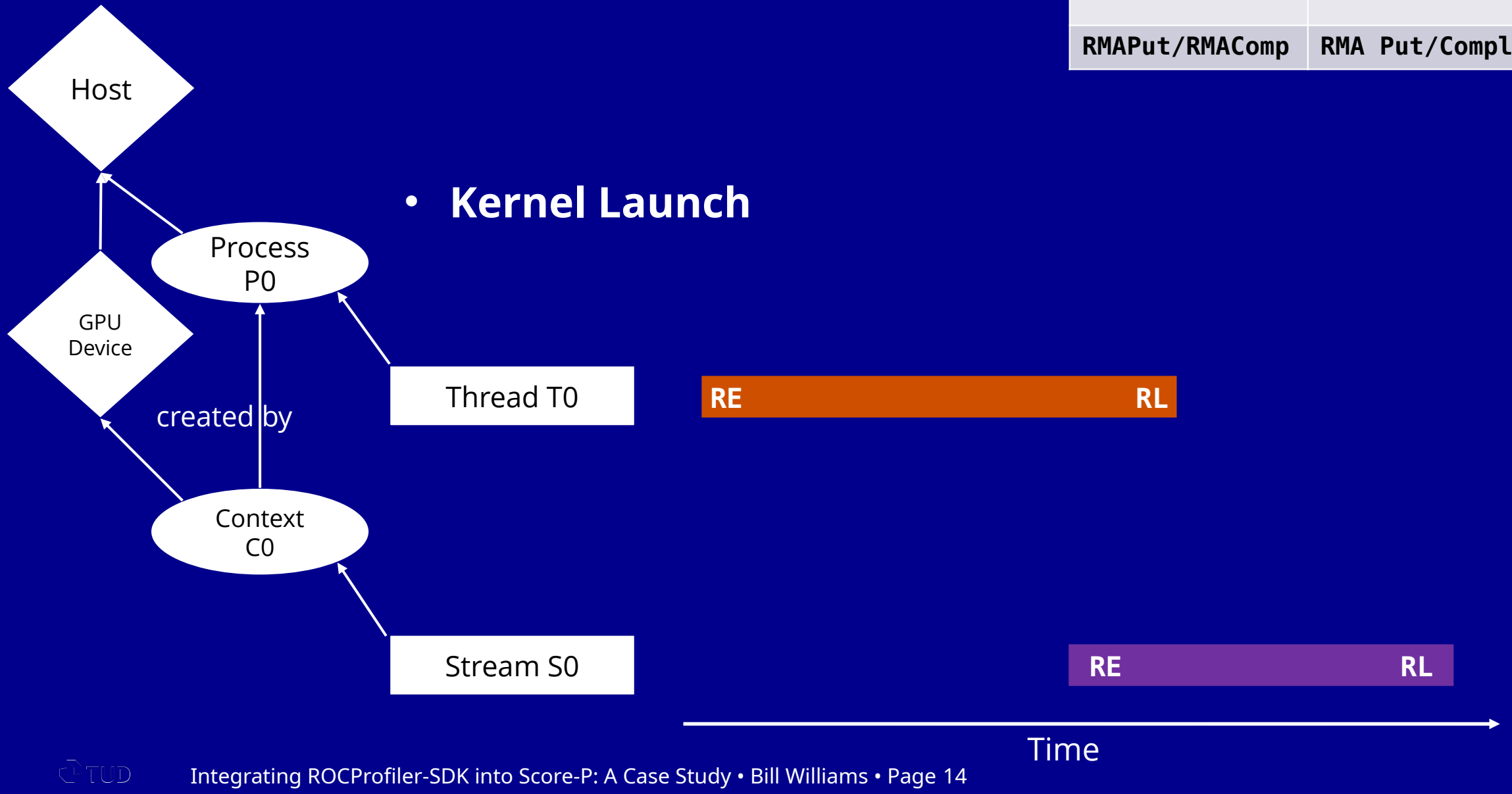
- What event models exists in OTF2
  - *Enter/leave* for functions, API, OpenMP constructs, and kernels
  - MPI *point-to-point* and *collectives*
  - Generic *fork/join* and *create/wait* threading events
  - Generic one-sided *communication/synchronization* used for MPI and offloading (*put/get* only)
- No specific events for offloading
  - ⇒Take opportunity to design one

Dresden University of Technology

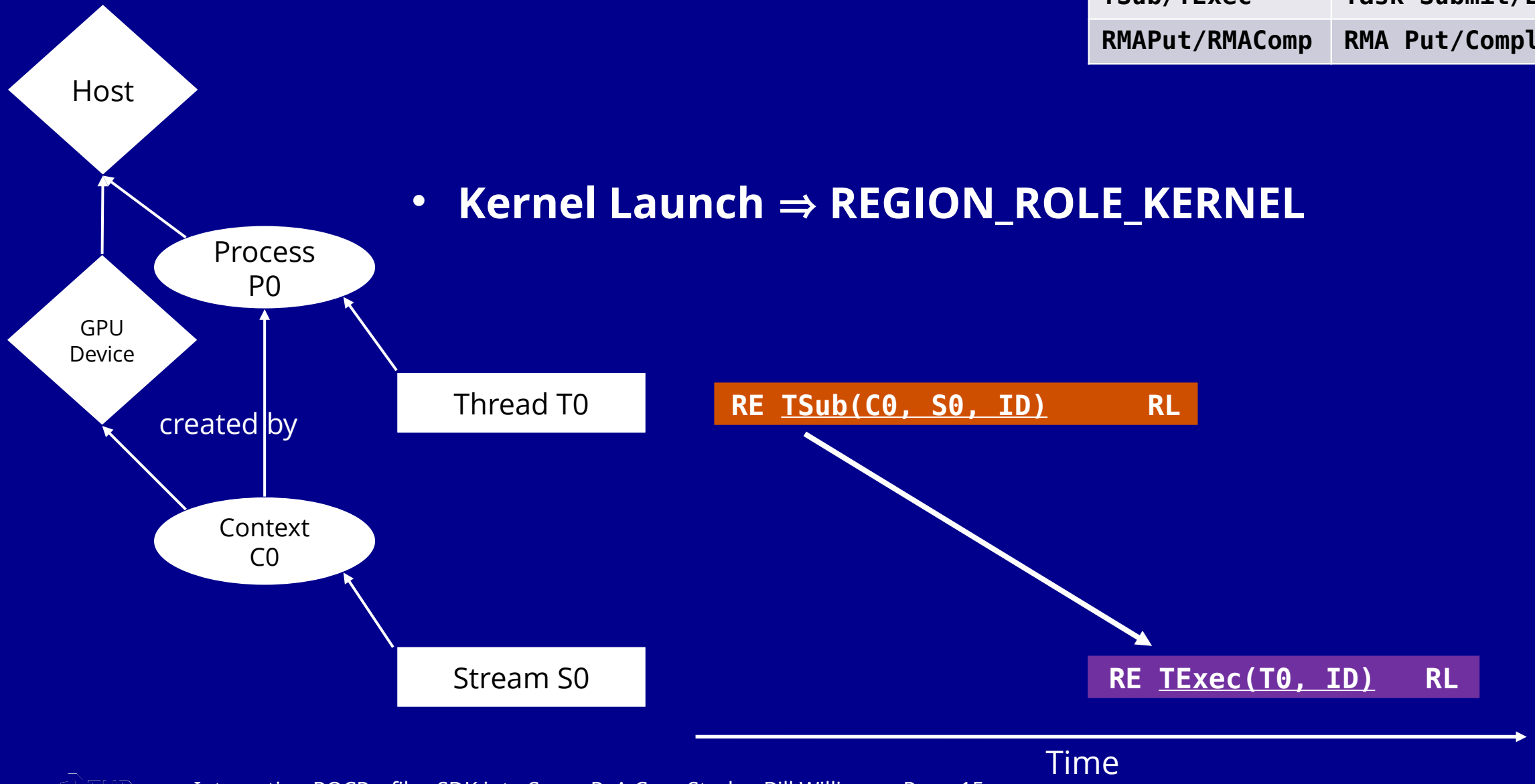
- What makes up the foundation of an offloading programming model?
  - Kernels
  - Data transfers
  - Synchronization
  - ⇒ Everything is a task submitted by the host to the device
  - ⇒ Focus on the correlation of host and device interaction
- What does it need to correlate events in different OTF2 locations?
  - Identifier: Correlation IDs provided by tools interface
  - Scope: One process and all of its threads and offloading streams

RE/RL	Region Enter/Leave
RMAPut/RMAComp	RMA Put/Complete

- **Kernel Launch**



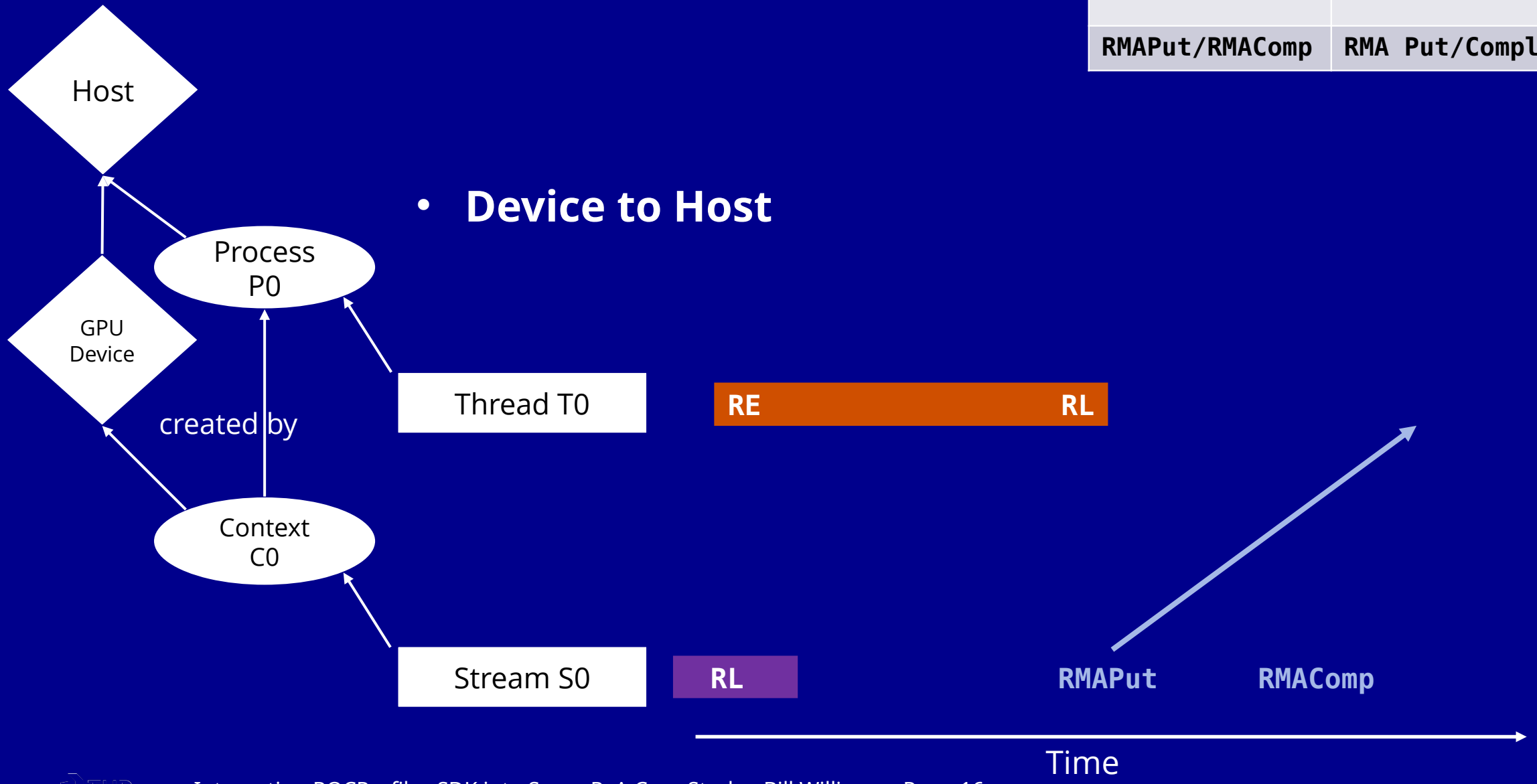
RE/RL	Region Enter/Leave
TSub/TExec	Task Submit/Execute
RMAPut/RMAComp	RMA Put/Complete



- **Kernel Launch ⇒ REGION_ROLE_KERNEL**

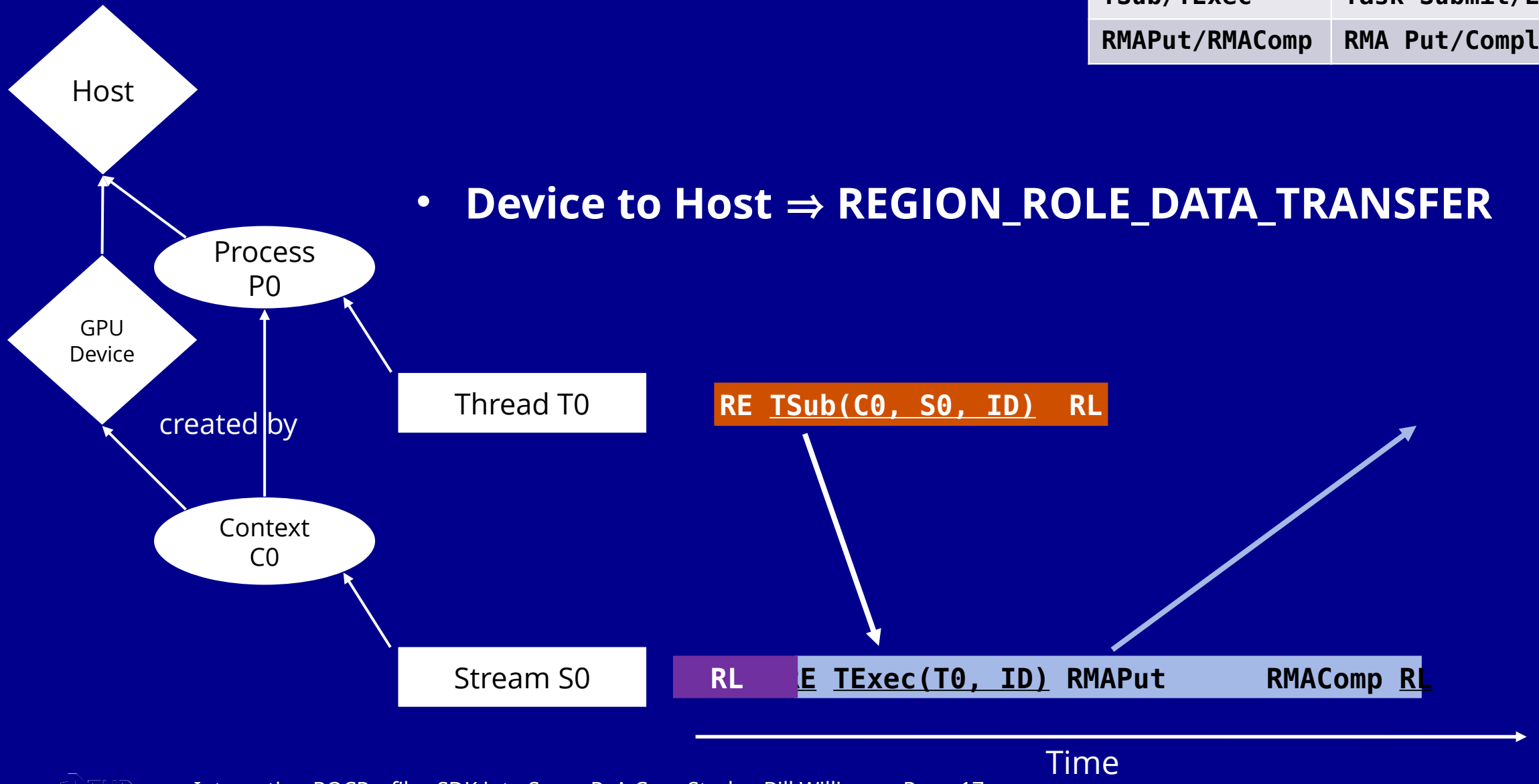
RE/RL	Region Enter/Leave
RMAPut/RMAComp	RMA Put/Complete

- **Device to Host**



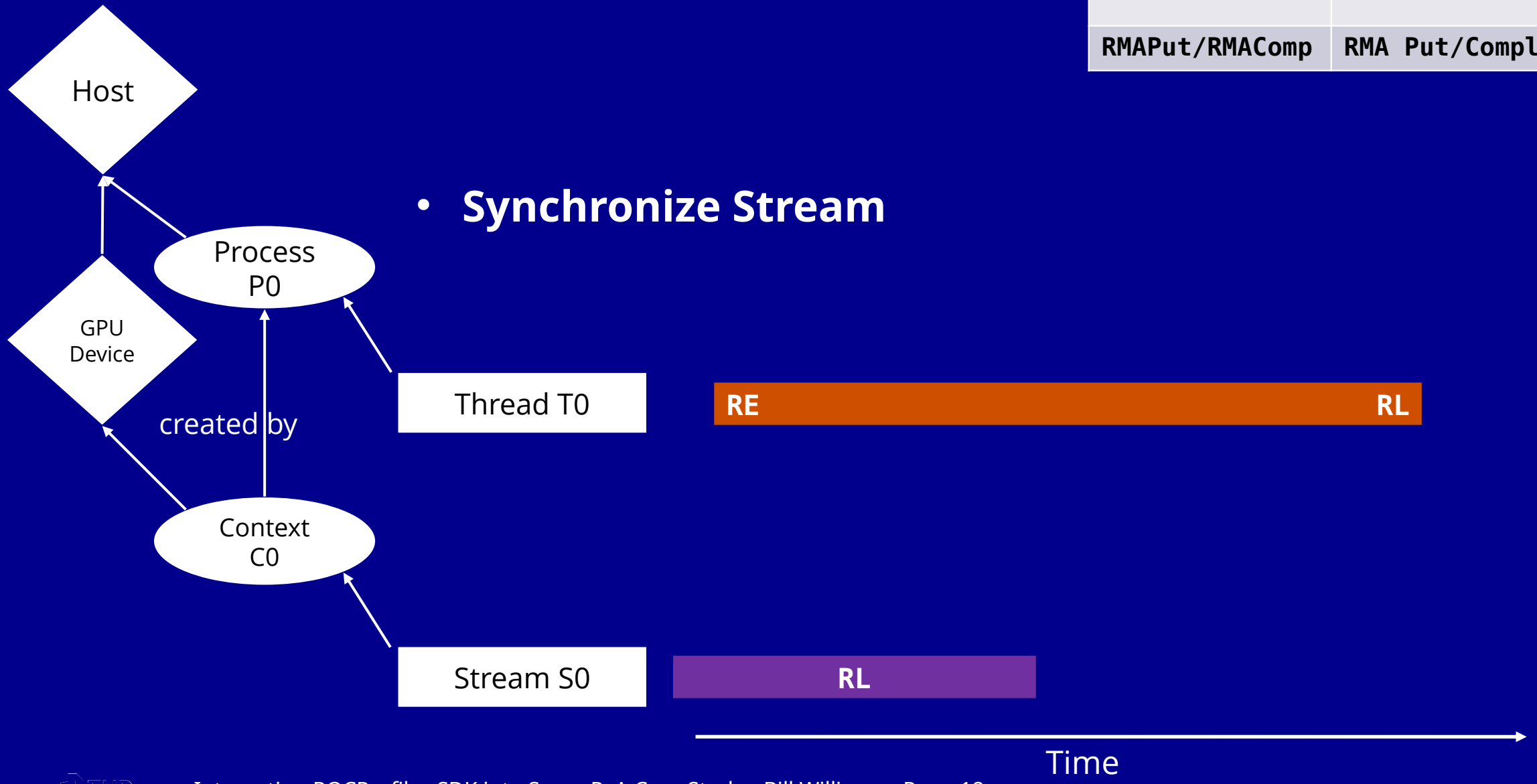
RE/RL	Region Enter/Leave
TSub/TExec	Task Submit/Execute
RMAPut/RMAComp	RMA Put/Complete

- **Device to Host ⇒ REGION_ROLE_DATA_TRANSFER**

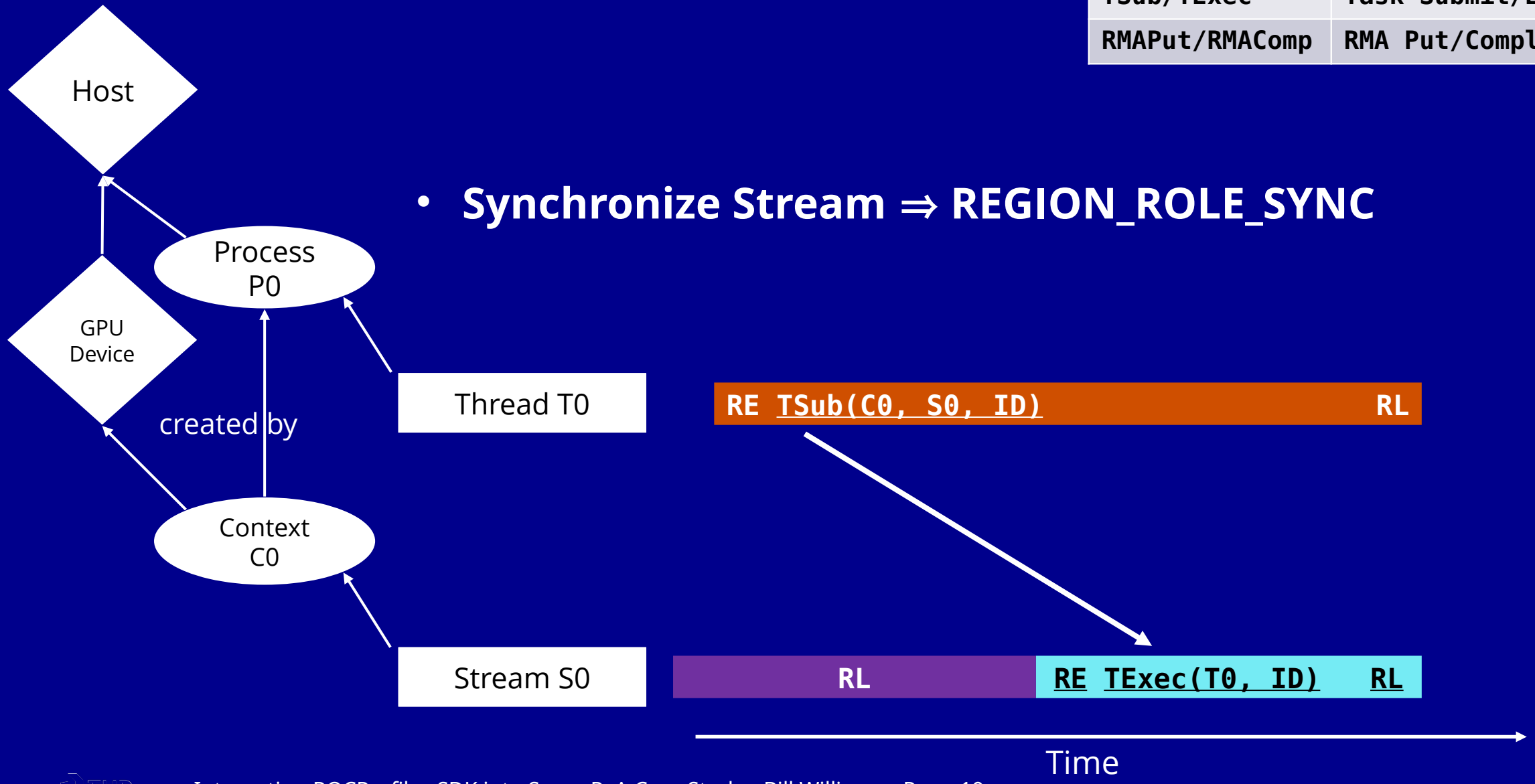


RE/RL	Region Enter/Leave
RMAPut/RMAComp	RMA Put/Complete

- **Synchronize Stream**



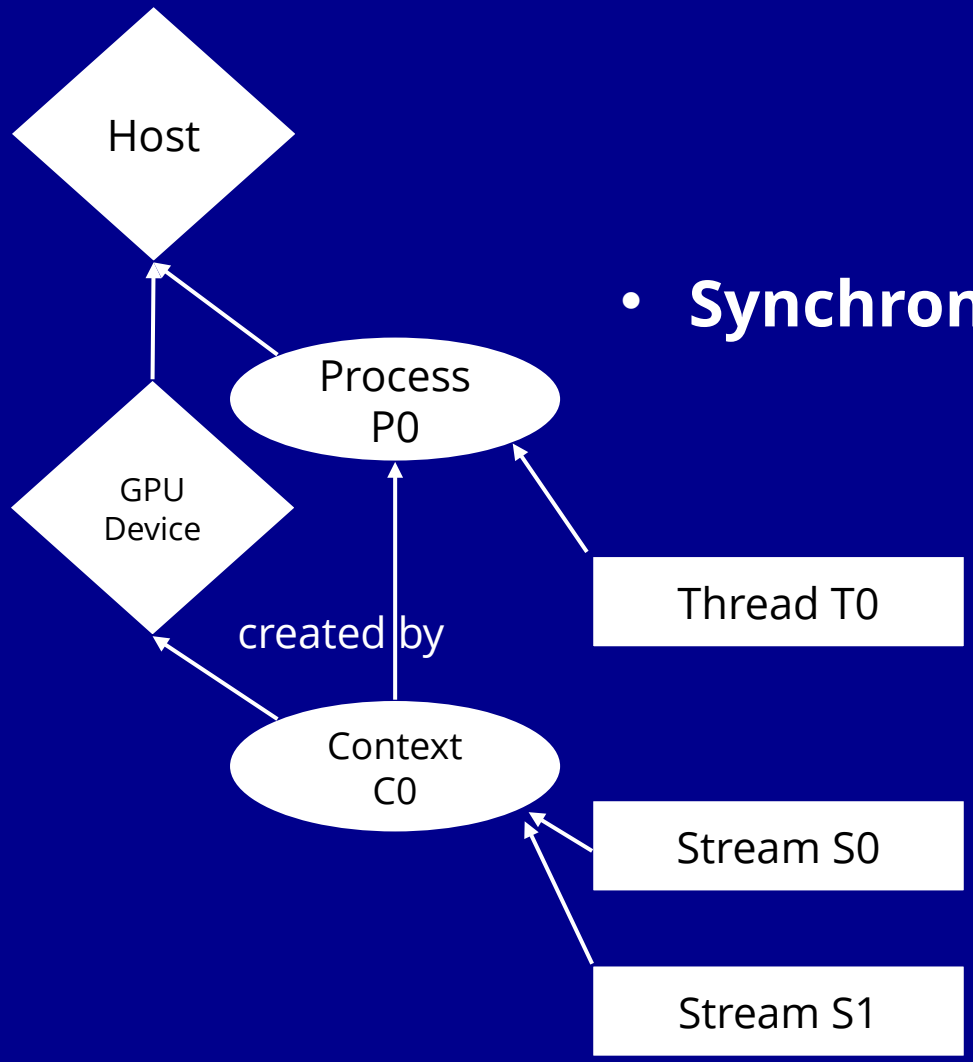
RE/RL	Region Enter/Leave
TSub/TExec	Task Submit/Execute
RMAPut/RMAComp	RMA Put/Complete

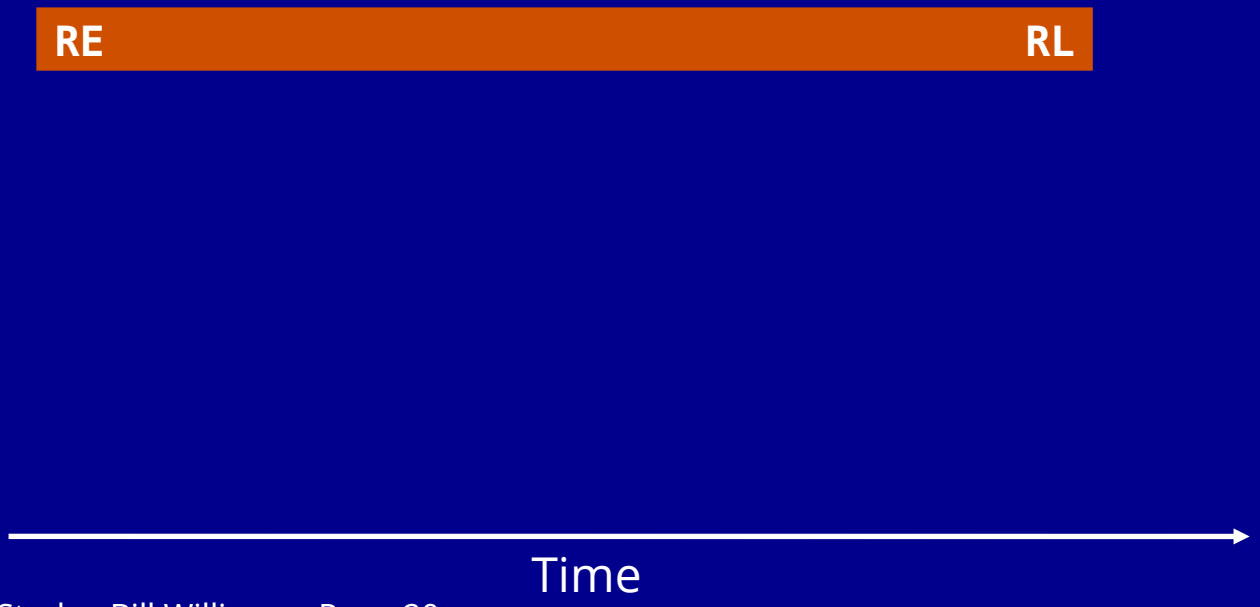


- **Synchronize Stream ⇒ REGION_ROLE_SYNC**

RE/RL	Region Enter/Leave
RMAPut/RMAComp	RMA Put/Complete

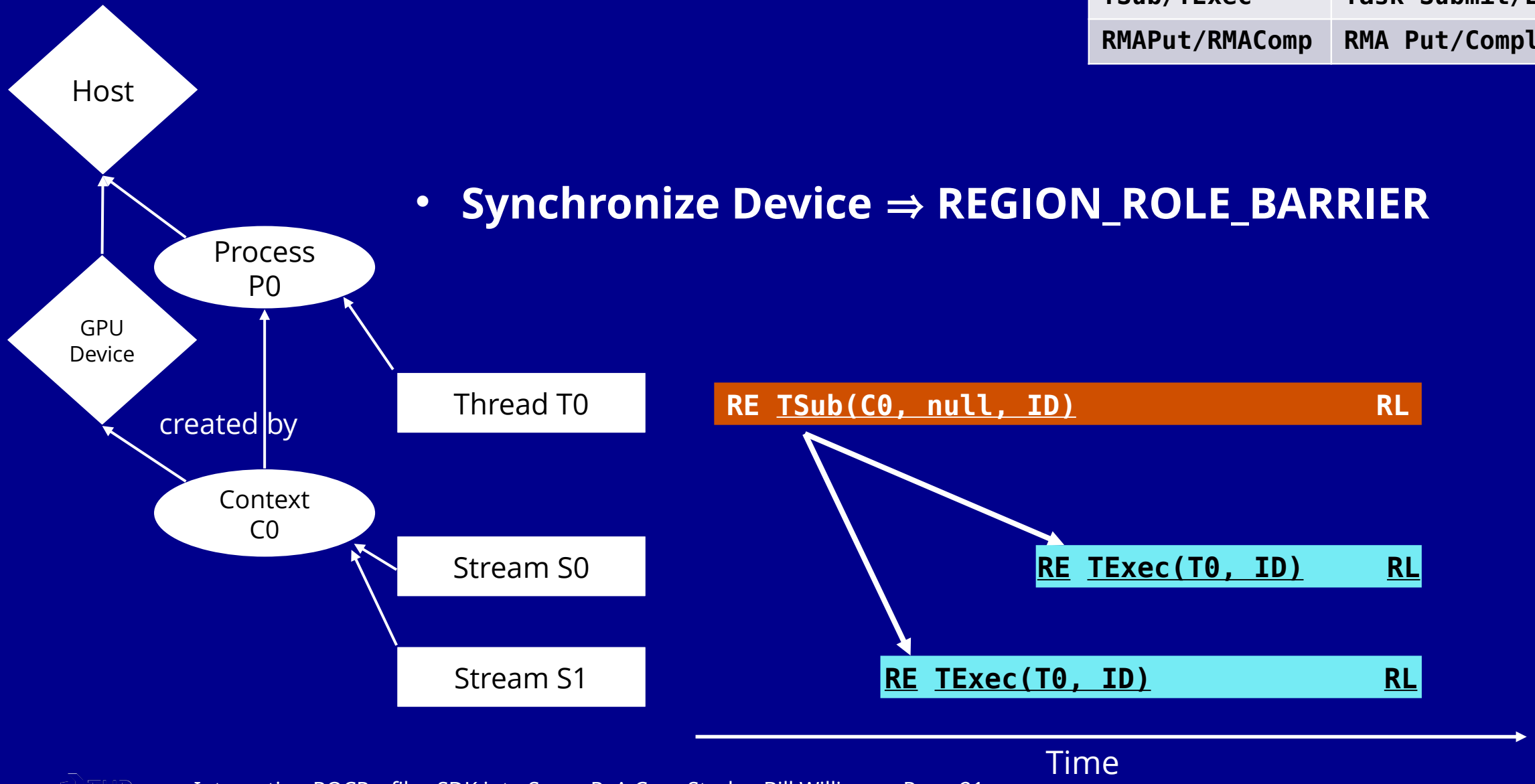
- **Synchronize Device**





RE/RL	Region Enter/Leave
TSub/TExec	Task Submit/Execute
RMAPut/RMAComp	RMA Put/Complete

- **Synchronize Device ⇒ REGION_ROLE_BARRIER**



Dresden University of Technology

- What changes in OTF2 and why?
  - Task launch/execution events
    - Incorporate correlation IDs—these are first-class elements of both ROCProfiler-SDK and CUPTI
    - Denote the submission mode: blocking/non-blocking
  - New region roles/flags
    - Capture host/device distinction
    - More precise description of API calls
  - Generate barriers corresponding to sync events
    - Explicit to start, can extend to synchronous API calls later

Dresden University of Technology

What does the adapter look like?

- API calls recorded on host threads as usual
- Activity buffers processed per-flush as usual
- Code object load/unload based interface requires adaptation to batch-create kernel regions up front

Dresden University of Technology

What does the adapter look like?

- “Many small callbacks” approach becomes possible
  - Kernel launch API calls record entry/exit, kernel launch *events* record the kernel details, correlation ID, etc.
- Stream changing callbacks + device changing API interception to keep track of Score-P *locations*

Dresden University of Technology

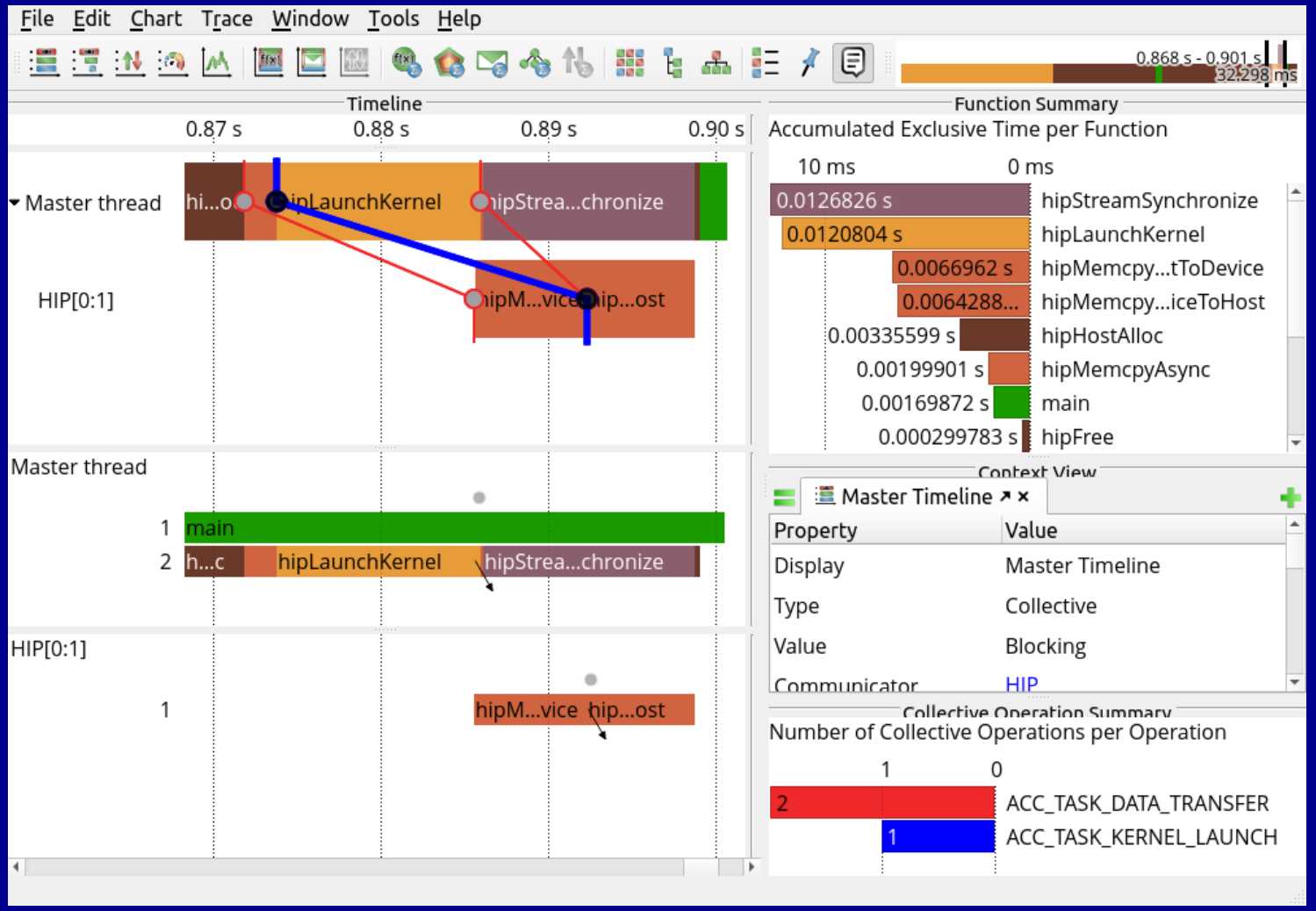
What does the adapter look like?

- New implementation to support synchronization/barriers
  - Record when and for what streams (device locations) a synchronizing call is started and ends on the host
  - Keep a list of pending sync operations for each location
  - When processing device events, write a sync from the *last* event pre-sync to the *first* event post-sync on the stream

Dresden University of Technology

- What can we do with all this?
  - Waiting time becomes easier to measure
    - Contrast with the previous purely “idle region”-based approach
  - Memory metrics can have kinds, not just host/device memory
  - Still early WIP: device-side PC sampling, RCCL, and hardware counters
  - Internal details become visible
    - Features implemented by library kernels
    - Timing differences between HIP level and MPI level for data transfers

Dresden University of Technology



Dresden University of Technology

- Status and outlook
  - Lots of feature integration still to be done
  - Feature parity with old adapter basically ready
  - ISC26 release cycle seems likely

Dresden University of Technology

Questions?