SynCron
Efficient Synchronization Support for Near-Data-Processing Architectures

Ramulator Course
06.05.2022

Christina Giannoula

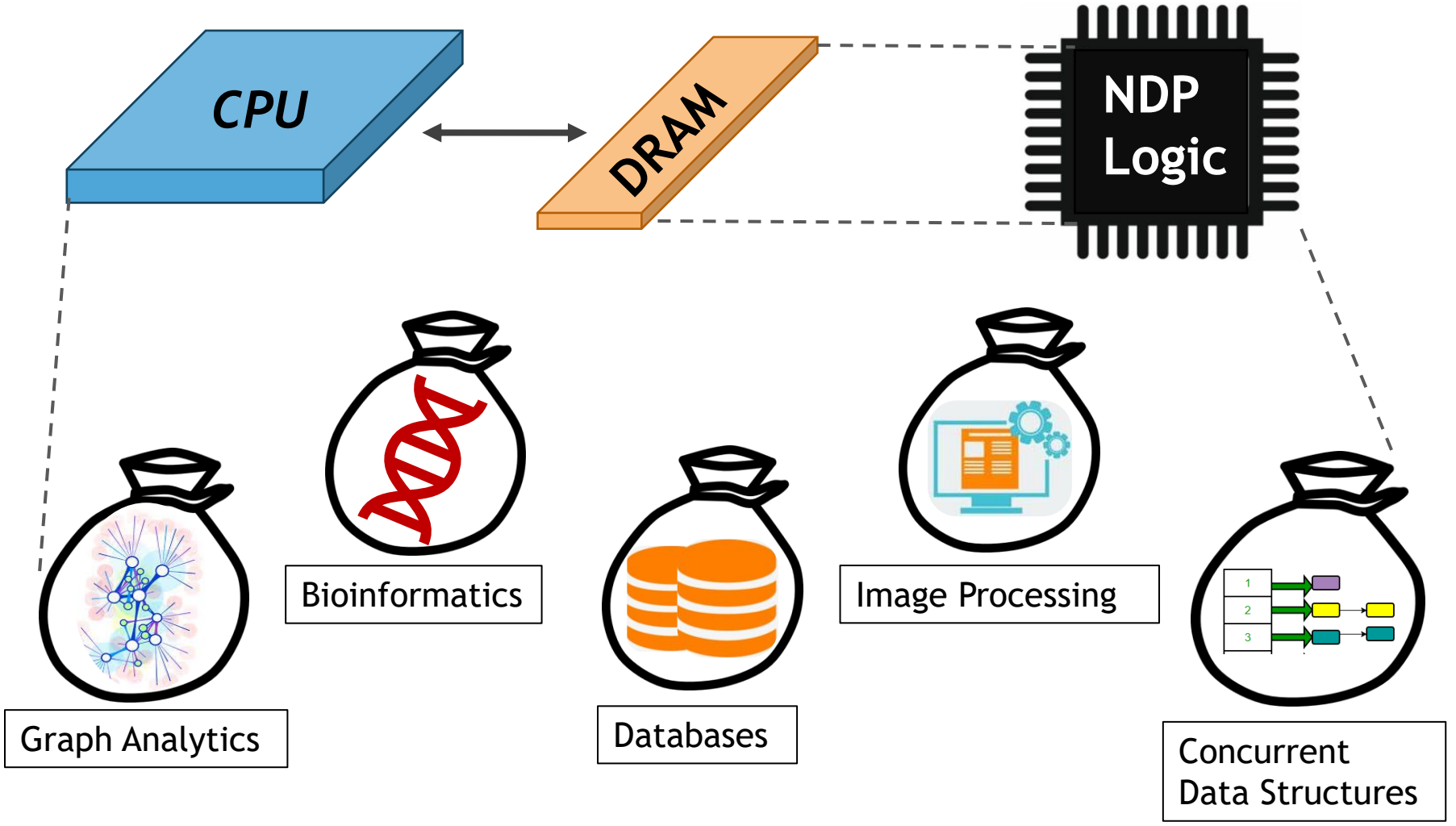
Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas
Ivan Fernandez, Juan Gómez Luna, Lois Orosa
Nectarios Koziris, Georgios Goumas, Onur Mutlu

SAFARI ETH zürich

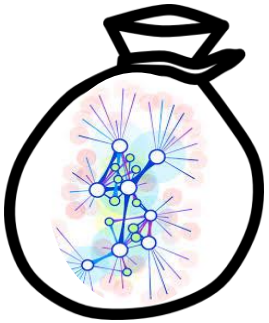
Agenda

- How is synchronization implemented in commodity architectures?
- Would existing hardware synchronization mechnanisms be efficient in Near-Data-Processing Architectures?
- How to design an efficient synchronization mechanism for Near-Data-Processing Architectures?

Near-Data-Processing (NDP) Systems



Synchronization is Necessary

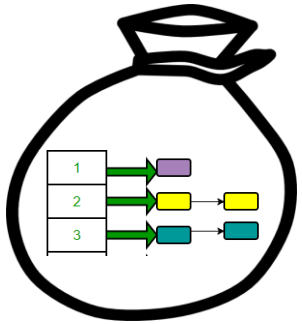


Graph Analytics

Bioinformatics

Databases

Image Processing



Concurrent Data Structures

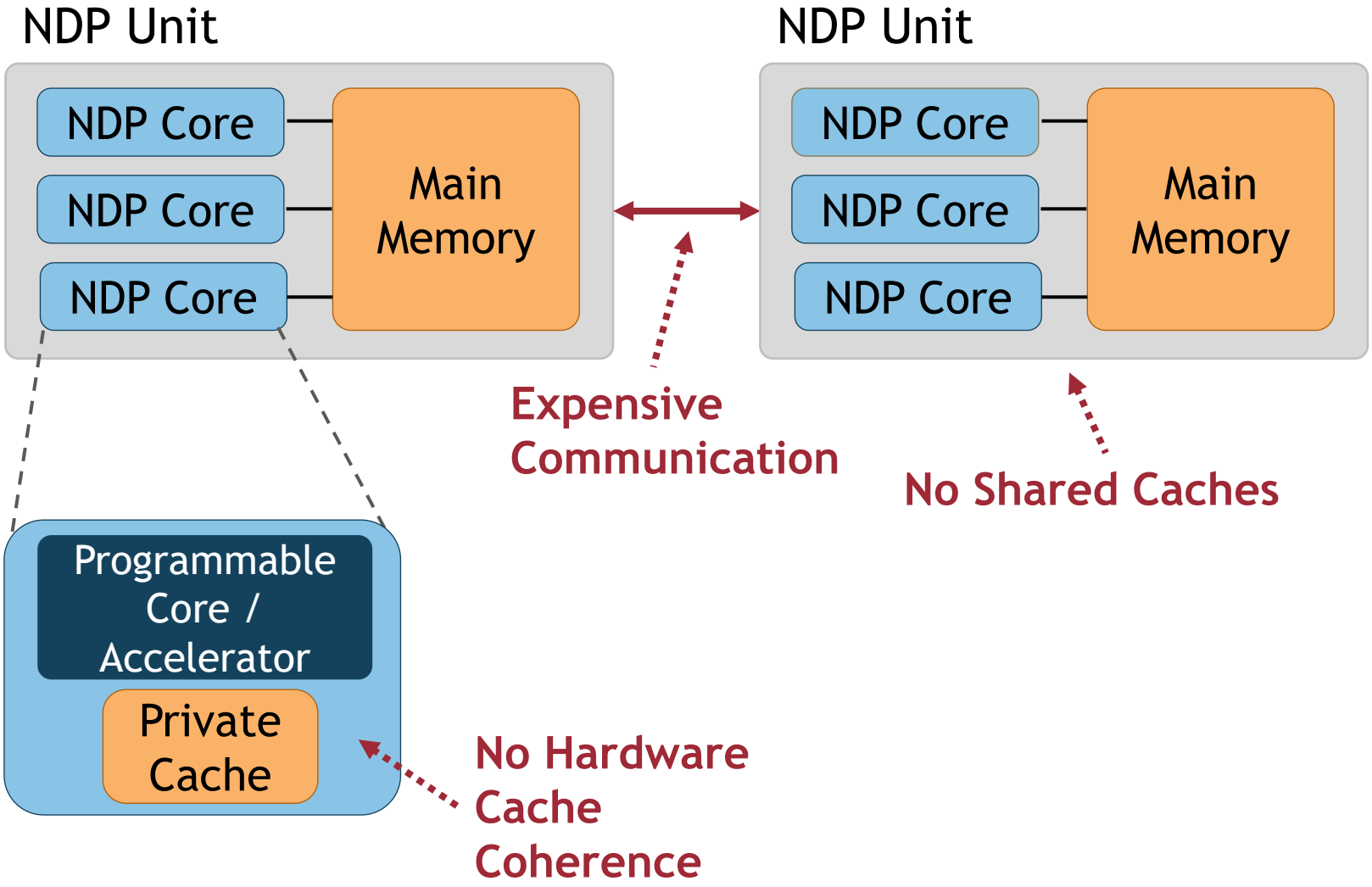
Single Source Shortest Path (SSSP)

Locks

Barriers

```
for v in Graph:
    for u in neighbors[v]:
        if distance[v] + edge_weight[v, u] < distance[u]
            lock_acquire(u)
            if distance[v] + edge_weight[v, u] < distance[u]
                distance[u] = distance[v] + edge_weight[v, u]
            lock_release(u)
```

Challenge: Efficient Synchronization



SynCron

The first end-to-end synchronization solution for NDP architectures

SynCron's Benefits:

1. High System **Performance**
2. Low Hardware **Cost**
3. Programming **Ease**
4. **Generality** to Cover a Wide Range of Synchronization Primitives

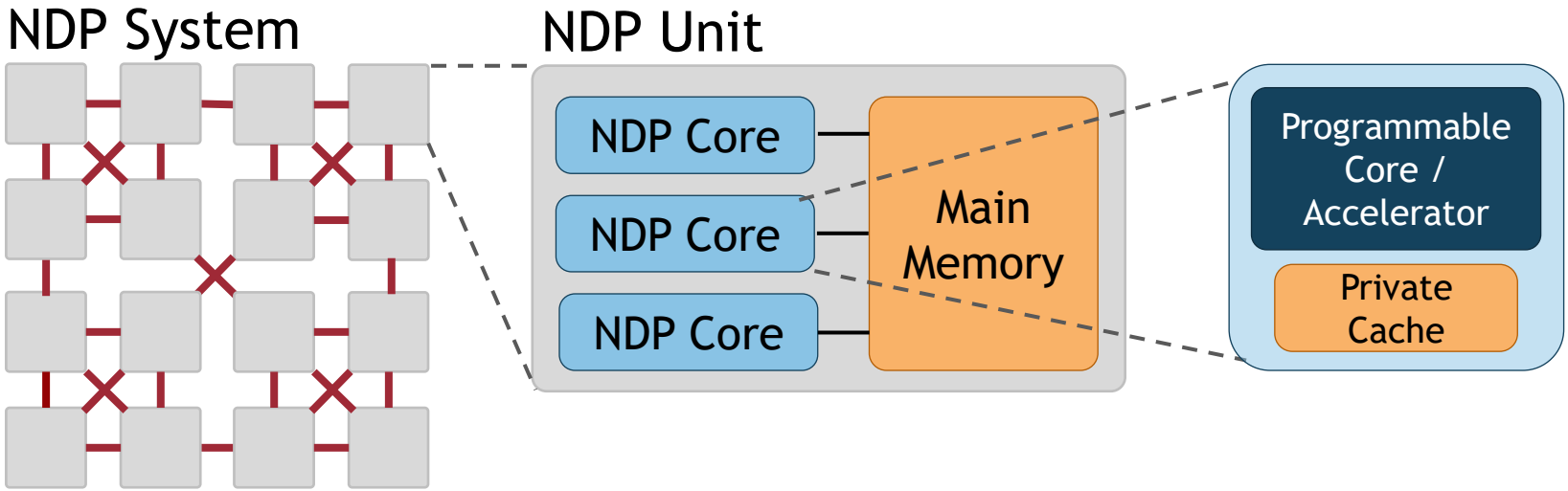
Outline

NDP Synchronization Solution Space

Our Mechanism: SynCron

Evaluation

Baseline NDP Architecture



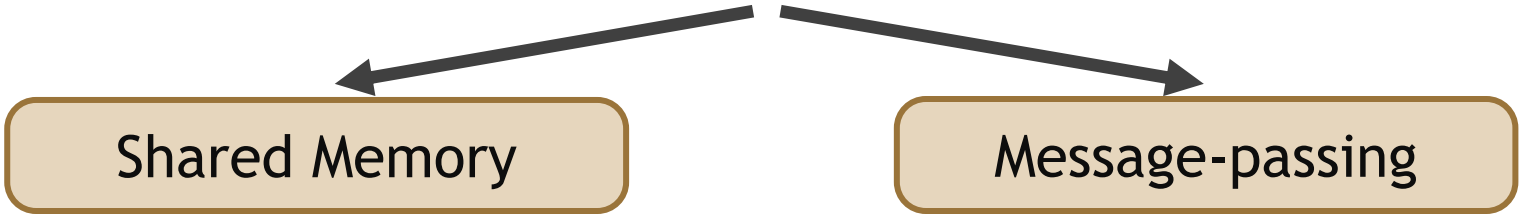
Synchronization **challenges in NDP** systems:

(1) Lack of a shared level of cache memory

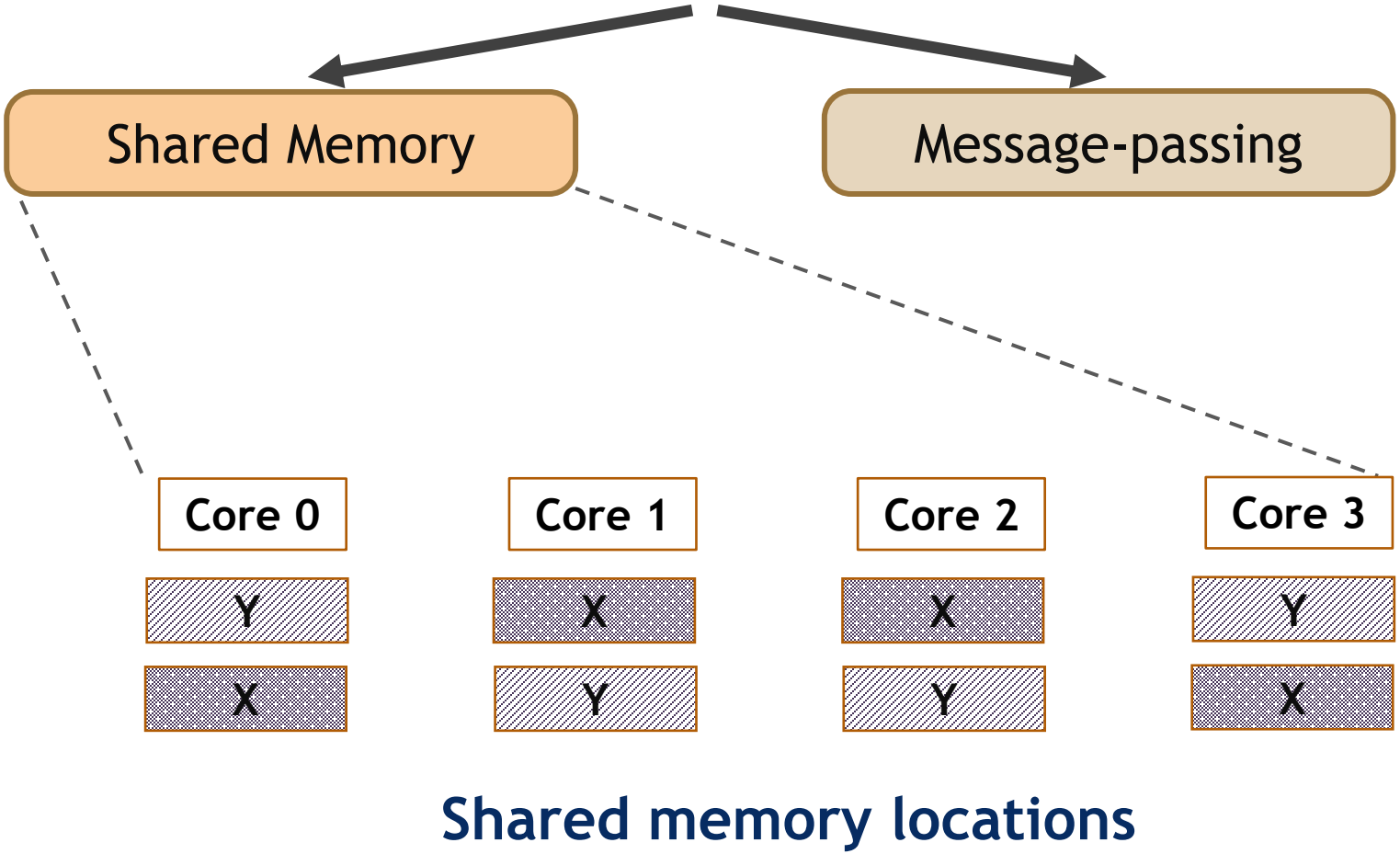
(2) Lack of hardware cache coherence support

(3) Expensive communication across NDP units

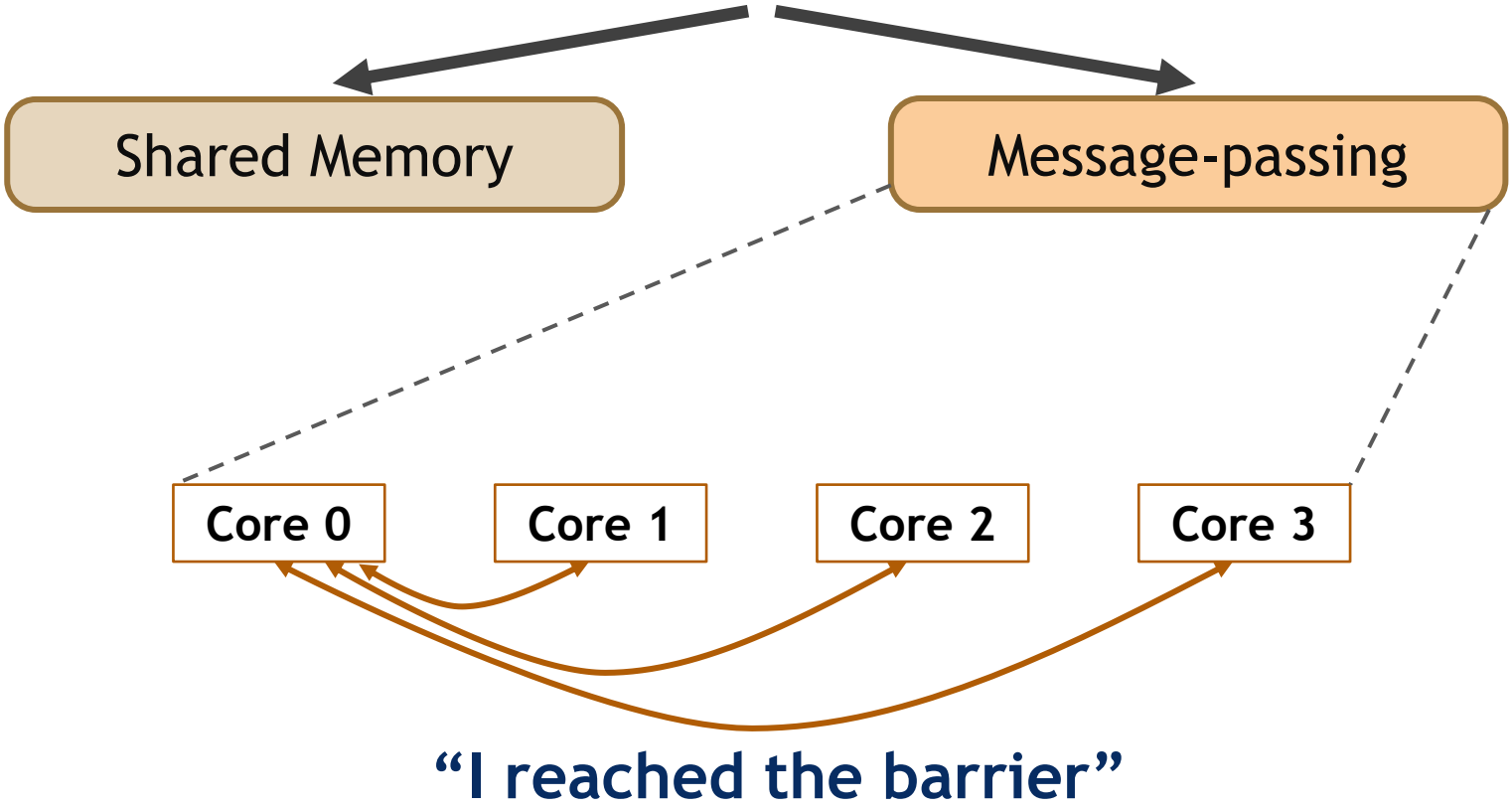
Synchronization Solution Space



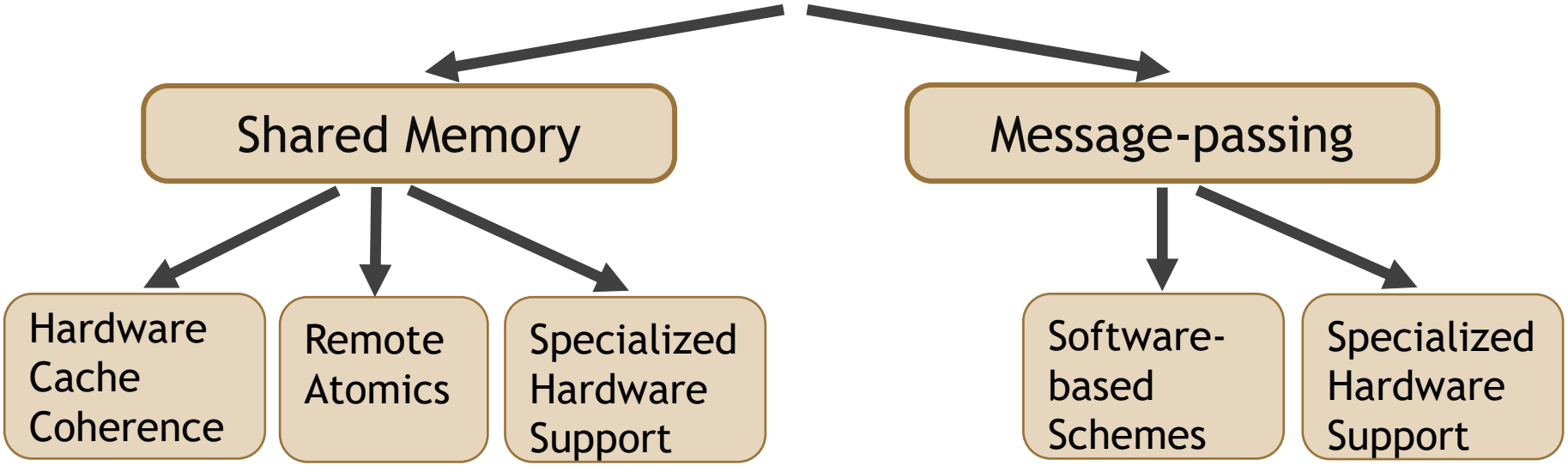
Synchronization Solution Space



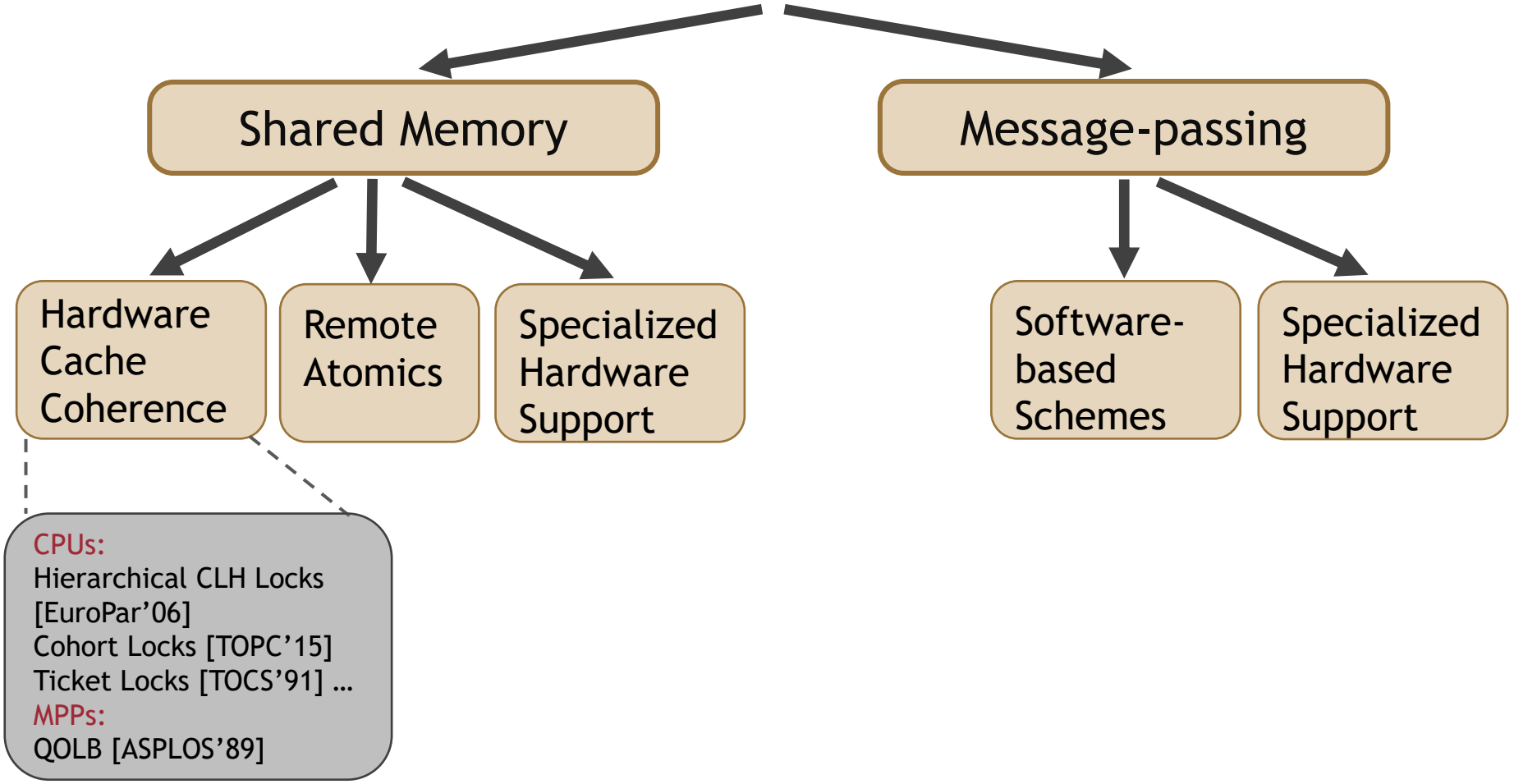
Synchronization Solution Space



NDP Synchronization Solution Space

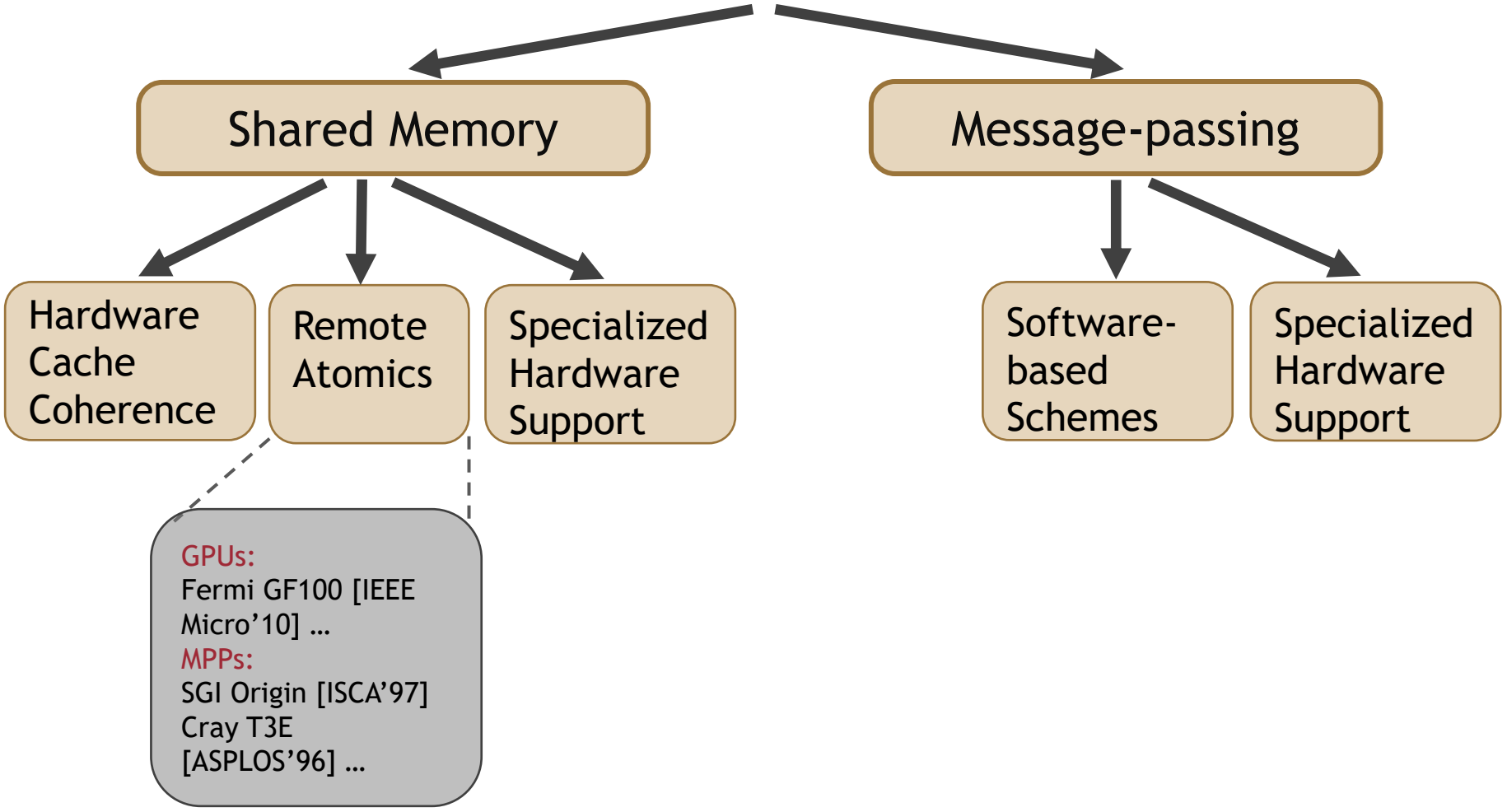


NDP Synchronization Solution Space



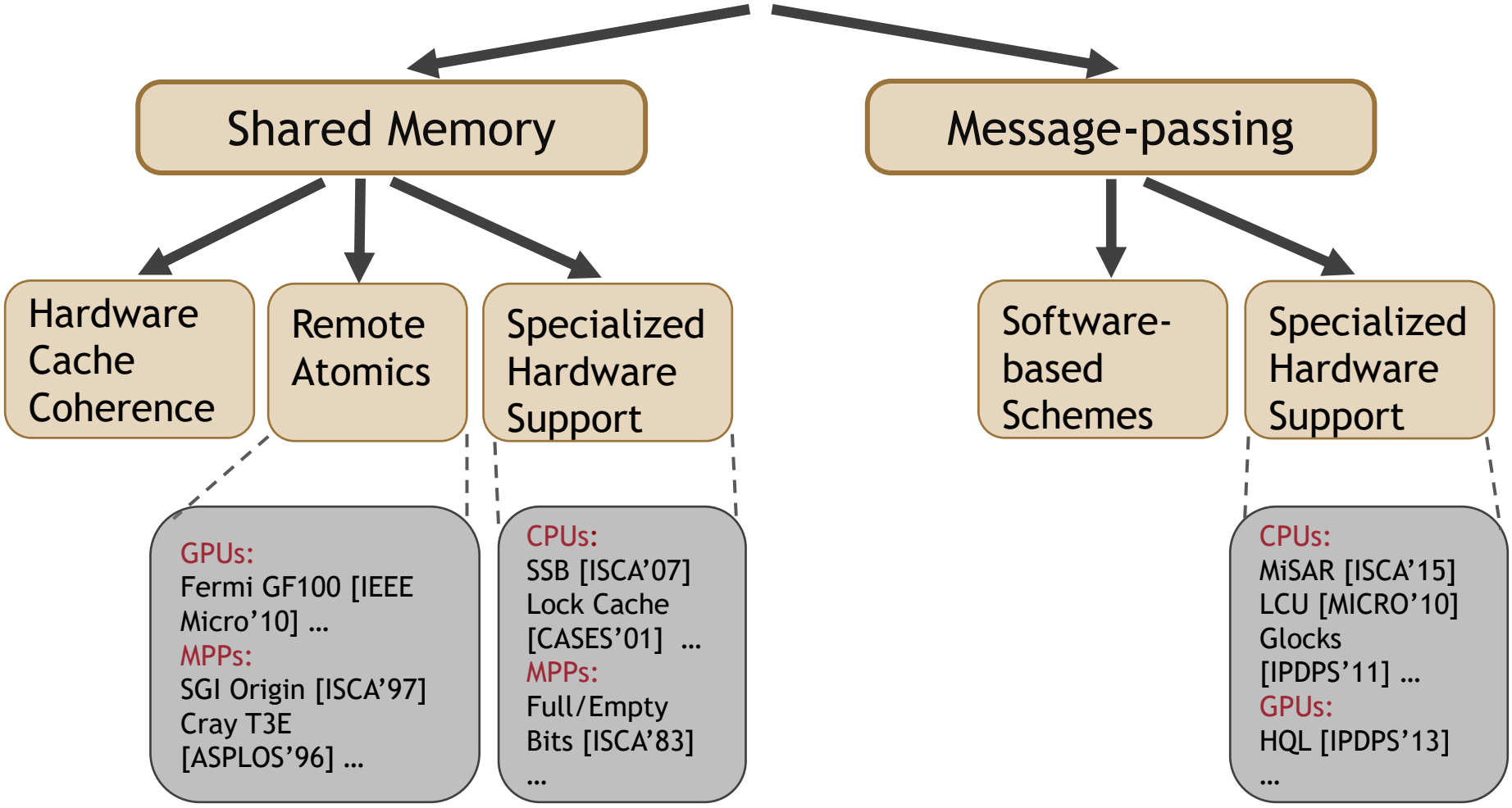
Lack of hardware cache coherence support

NDP Synchronization Solution Space



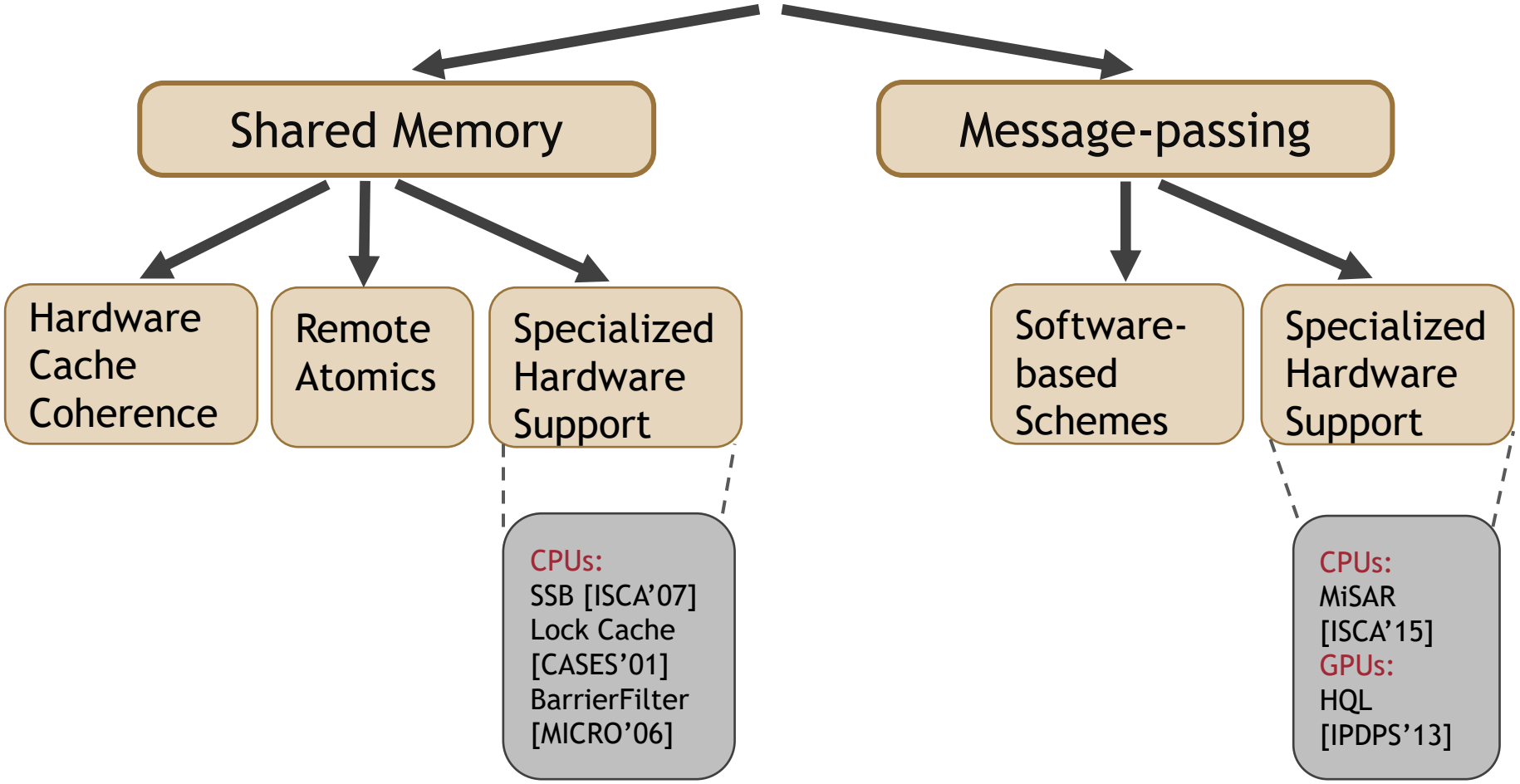
Expensive communication across NDP units

NDP Synchronization Solution Space



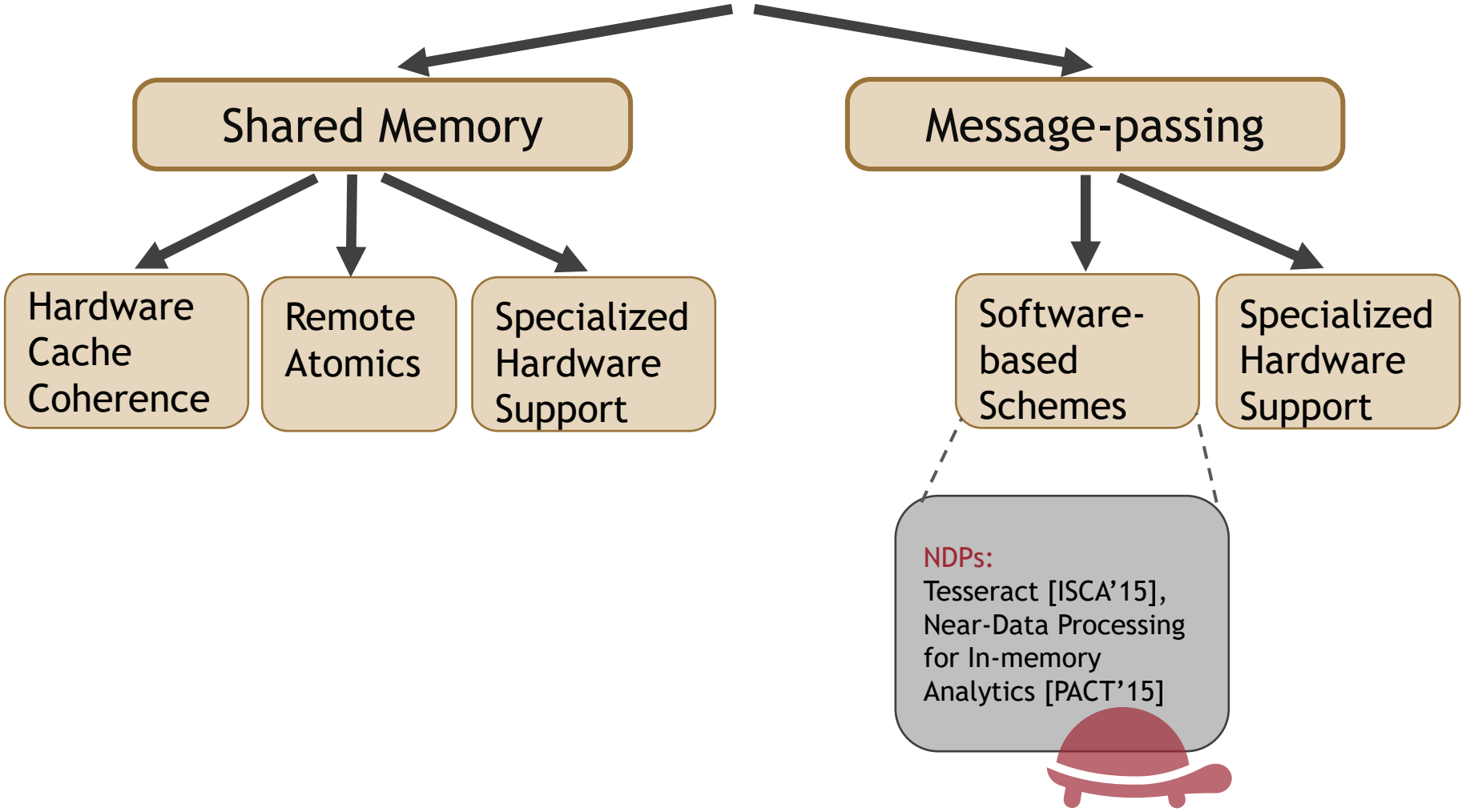
Expensive communication across NDP units

NDP Synchronization Solution Space

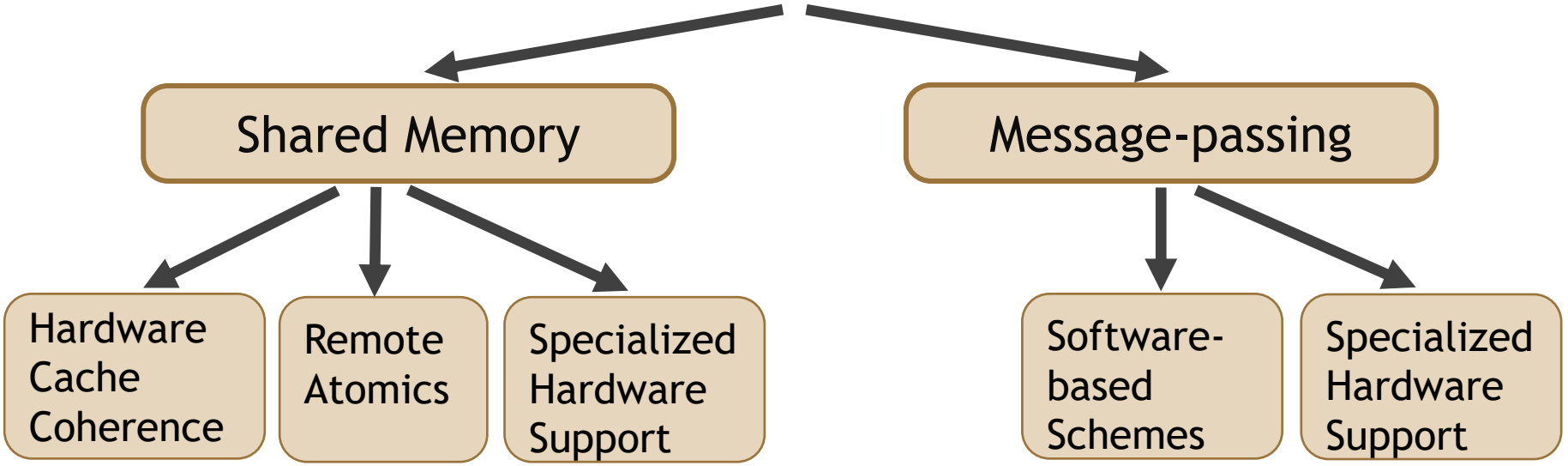


Lack of a shared level of cache memory

NDP Synchronization Solution Space

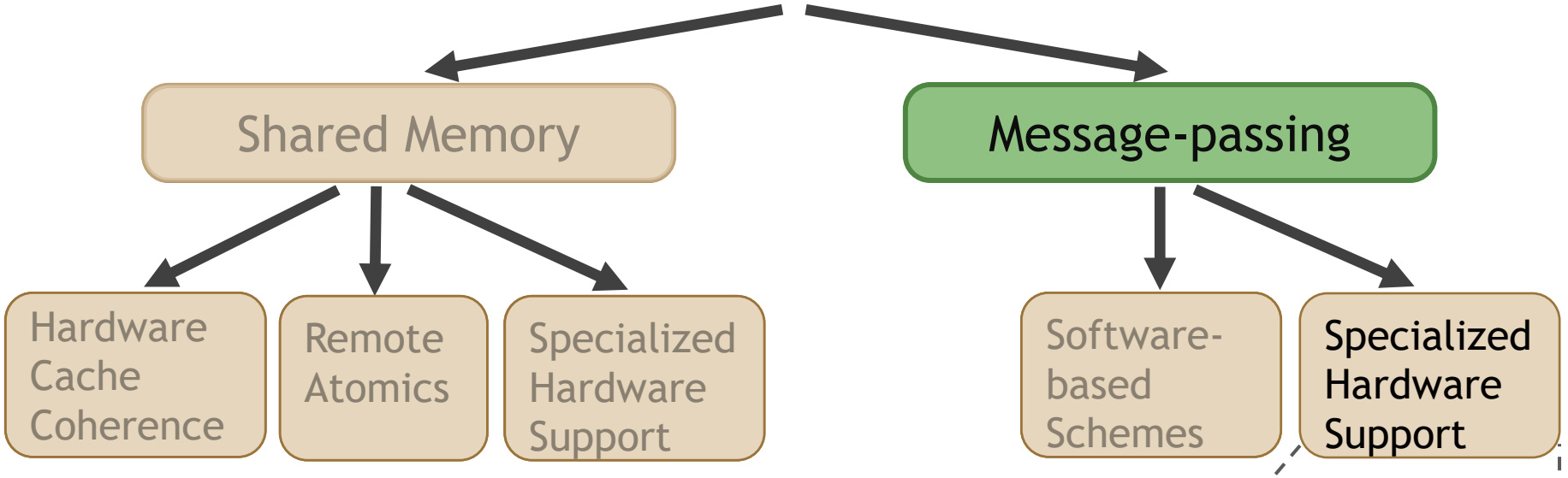


NDP Synchronization Solution Space



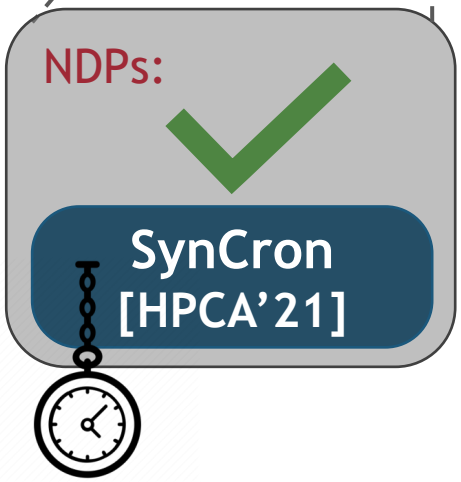
Prior schemes are **not suitable** or **efficient** for NDP systems

NDP Synchronization Solution Space



SynCron's Design Choices

- Hardware Message-passing to Avoid Synchronization via Shared Memory
- Hierarchical Communication to Eliminate Expensive Network Traffic
- Specialized Cache Structure to Minimize Latency Costs



Outline

NDP Synchronization Solution Space

Our Mechanism: SynCron

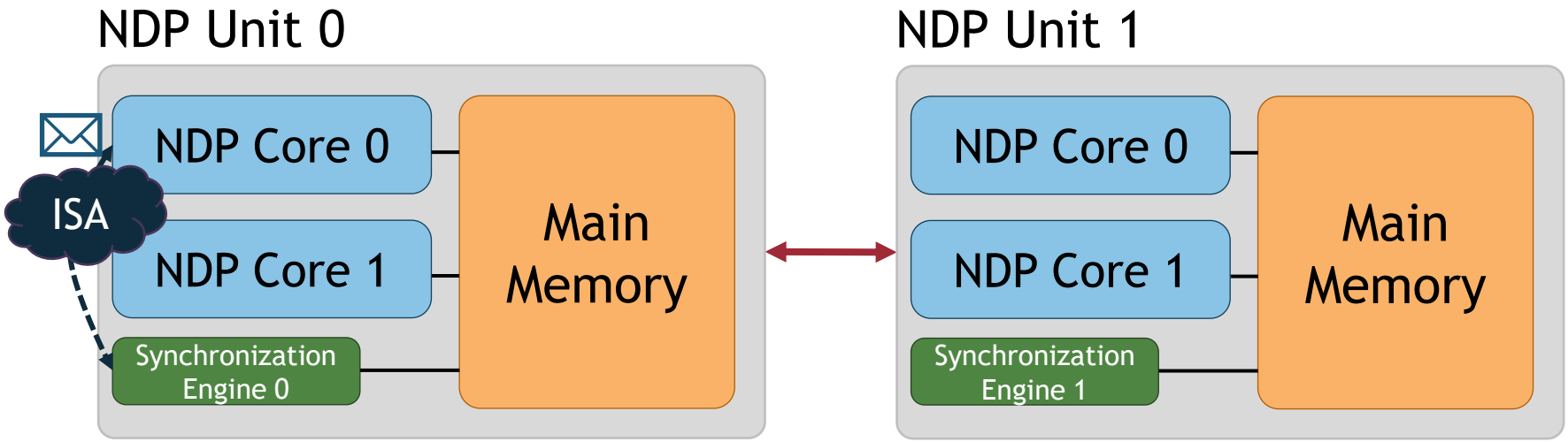
Evaluation

SynCron: Overview

SynCron consists of four key techniques:

1. **Hardware support** for synchronization acceleration
2. **Direct buffering** of synchronization variables
3. **Hierarchical** message-passing **communication**
4. Integrated hardware-only **overflow management**

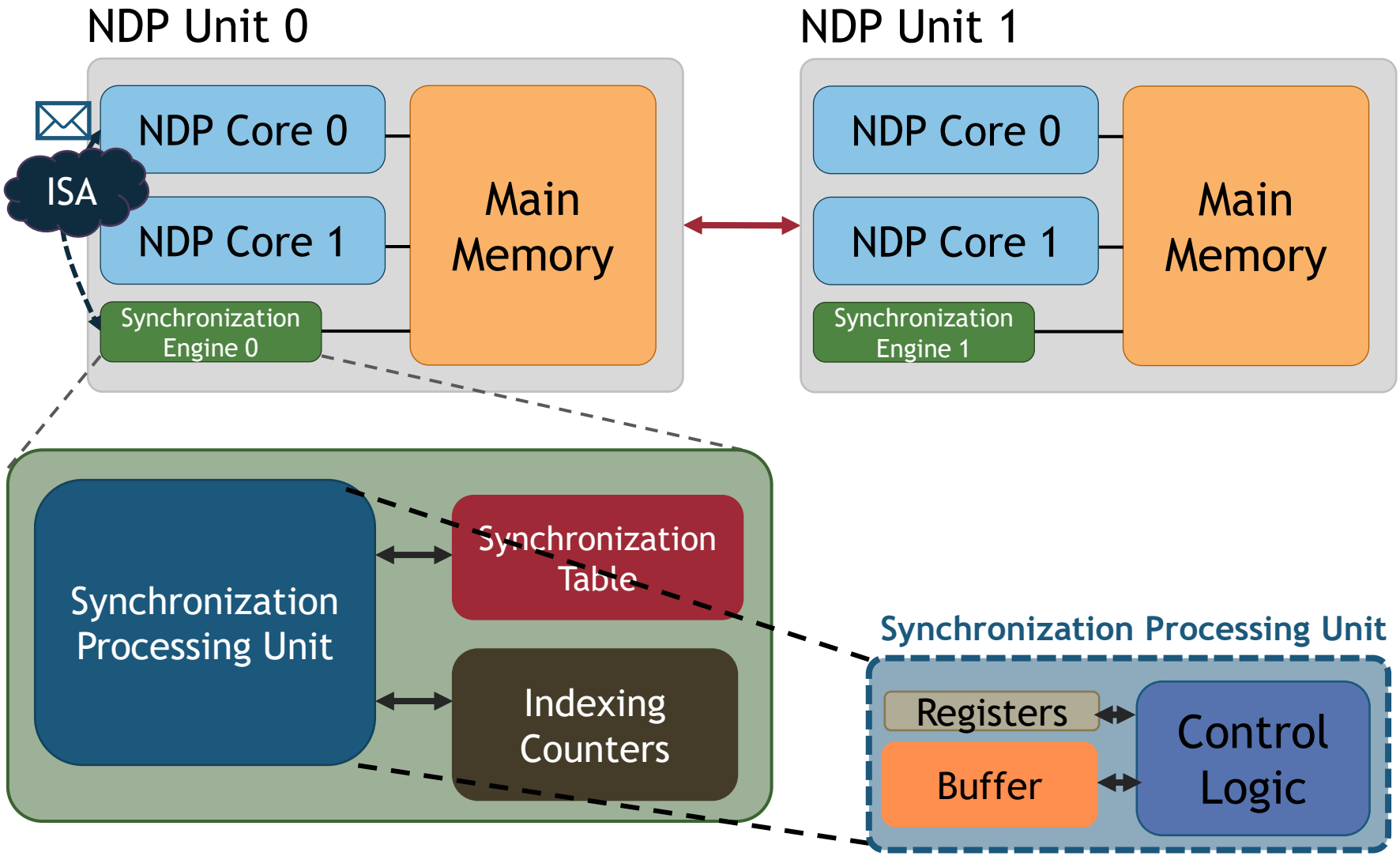
1. Hardware Synchronization Support



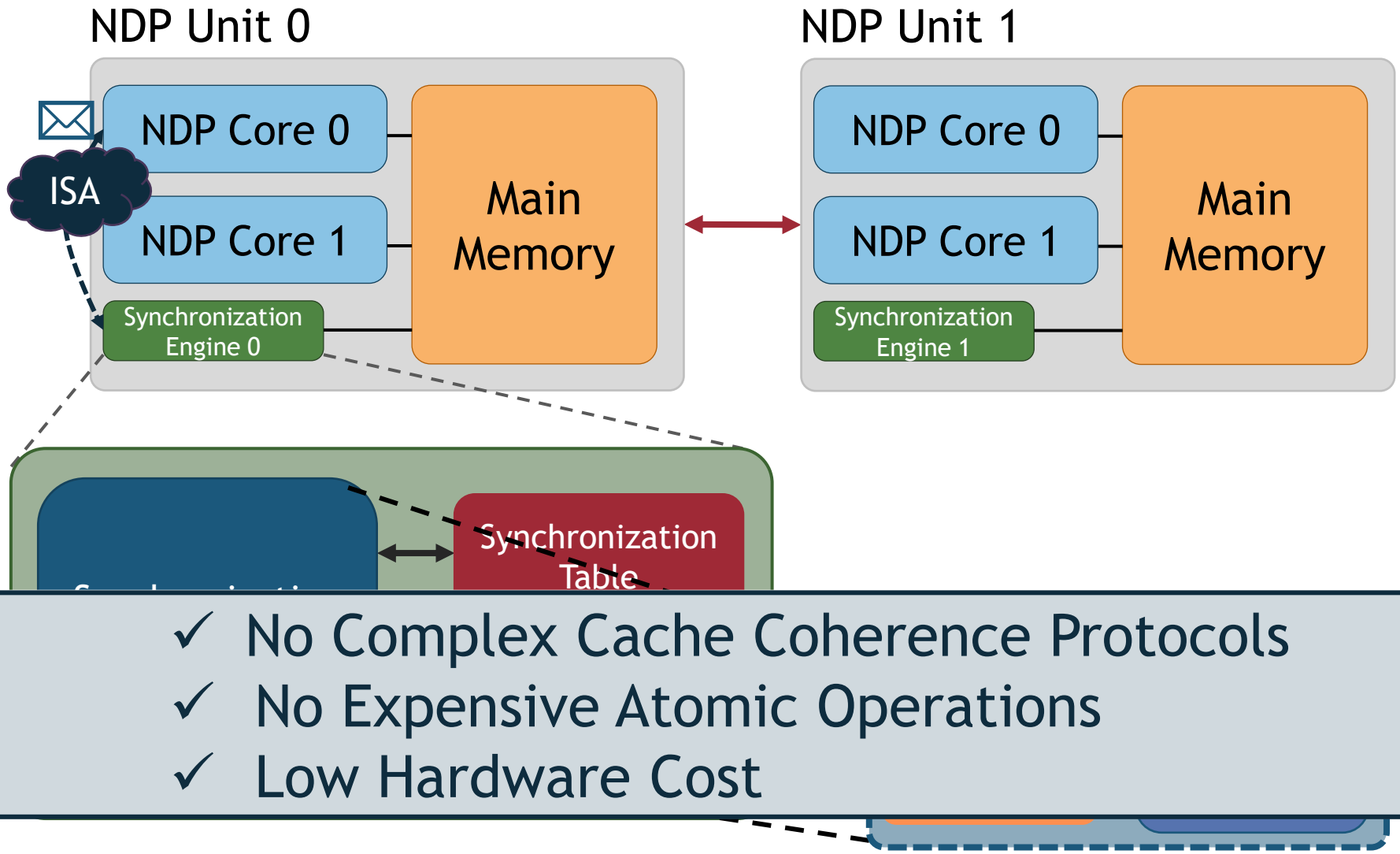
ISA
- req_sync
- req_async

Local lock acquire

1. Hardware Synchronization Support

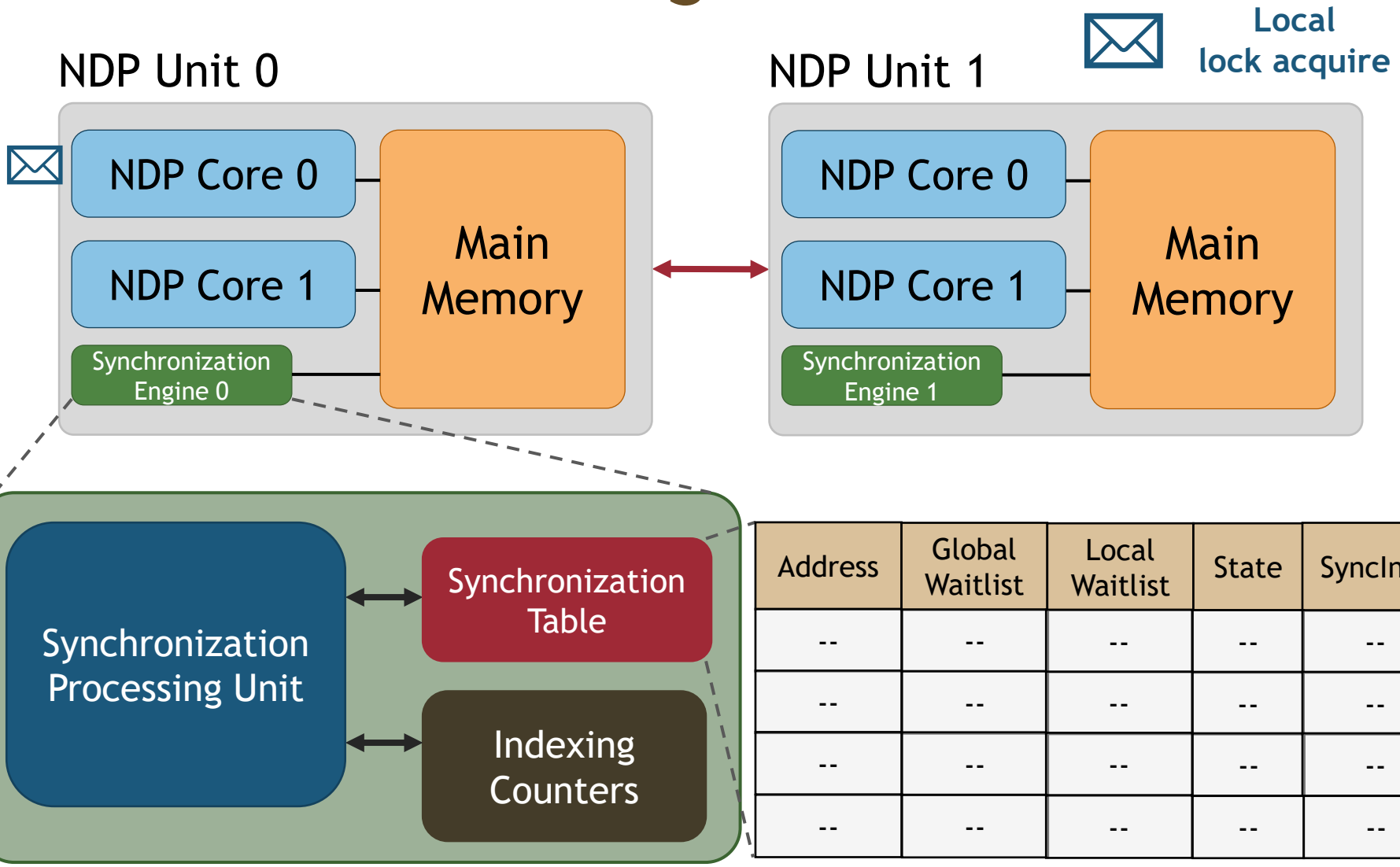


1. Hardware Synchronization Support



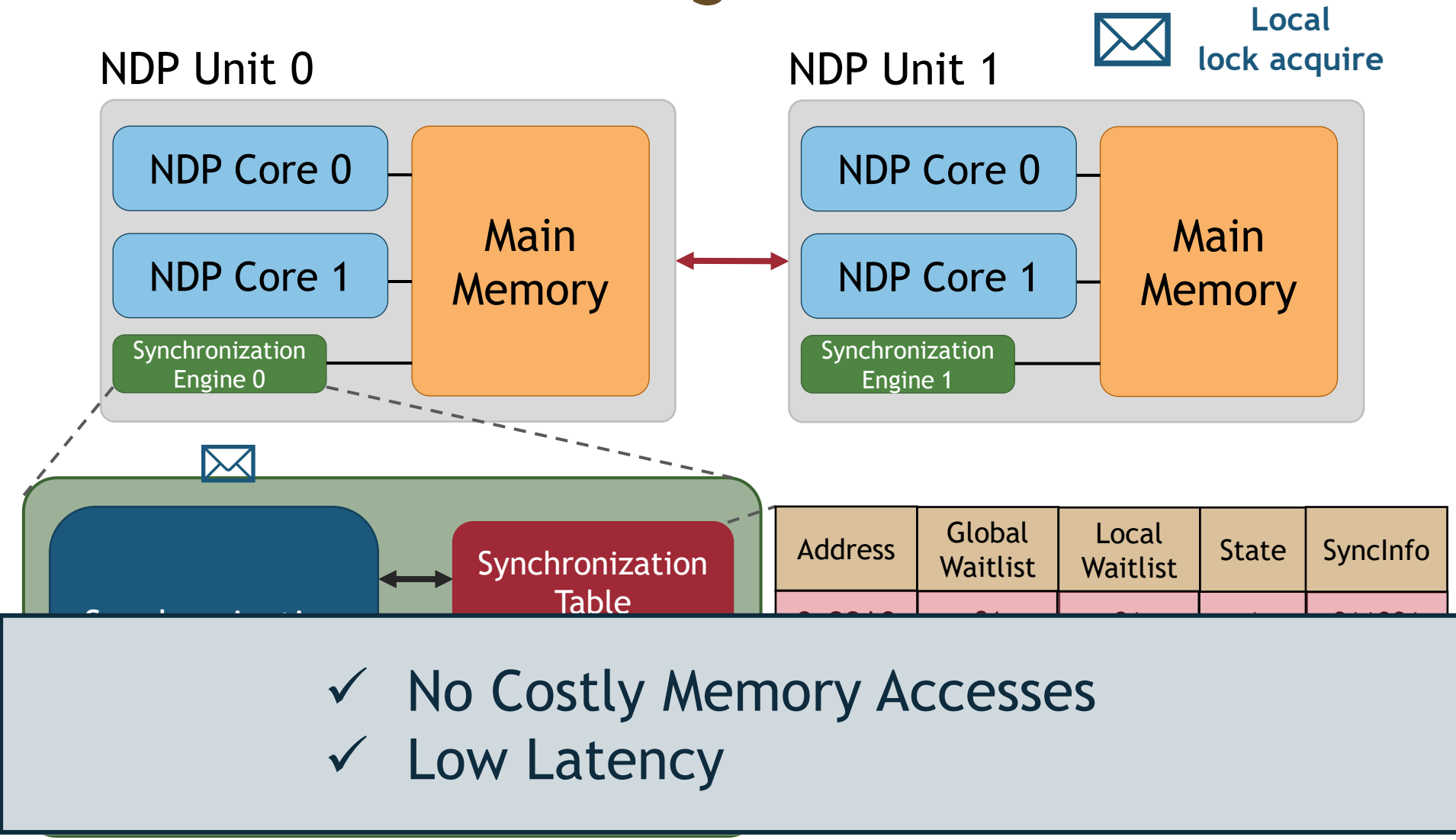
- ✓ No Complex Cache Coherence Protocols
- ✓ No Expensive Atomic Operations
- ✓ Low Hardware Cost

2. Direct Buffering of Variables



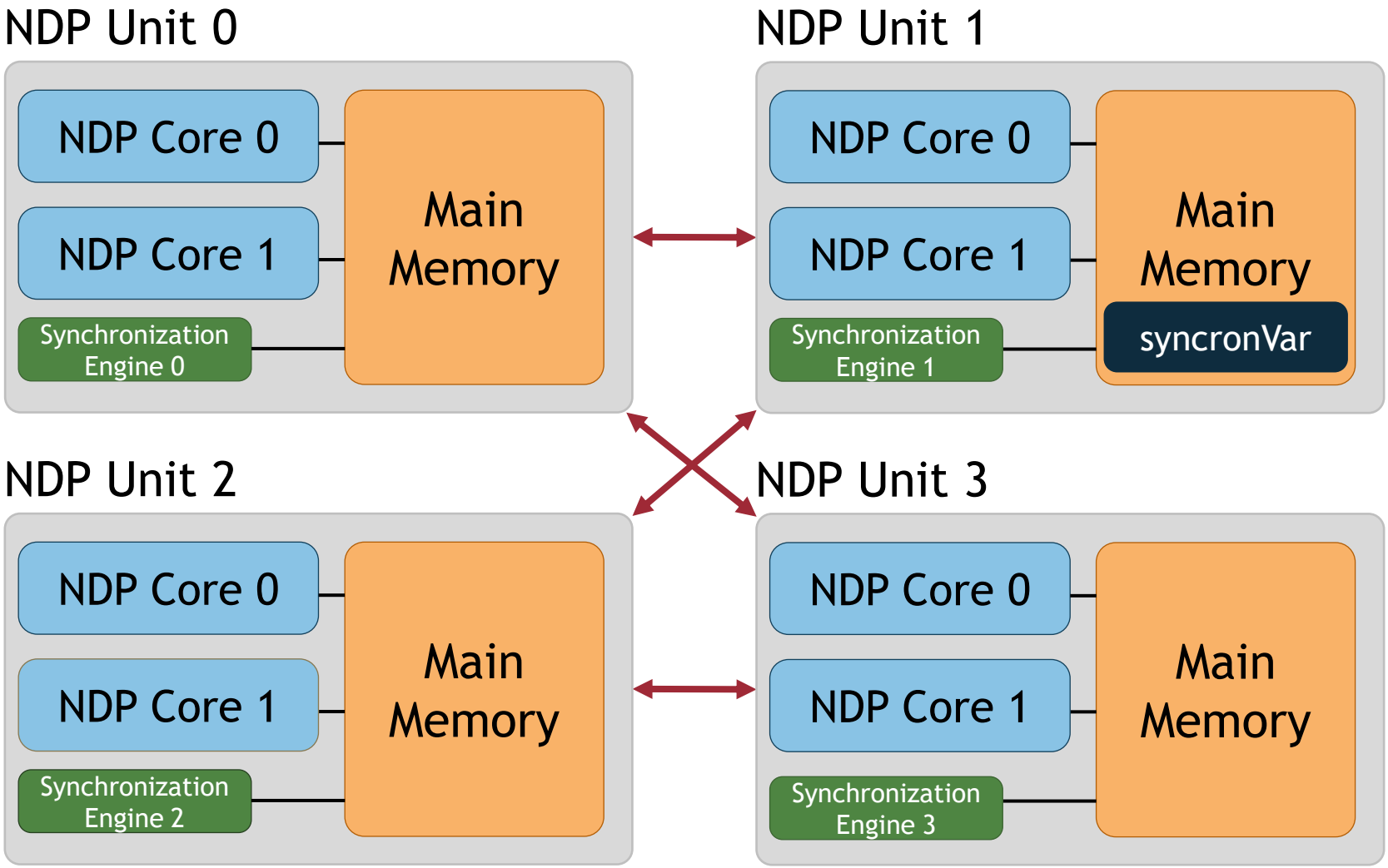
Address	Global Waitlist	Local Waitlist	State	SyncInfo
--	--	--	--	--
--	--	--	--	--
--	--	--	--	--
--	--	--	--	--

2. Direct Buffering of Variables

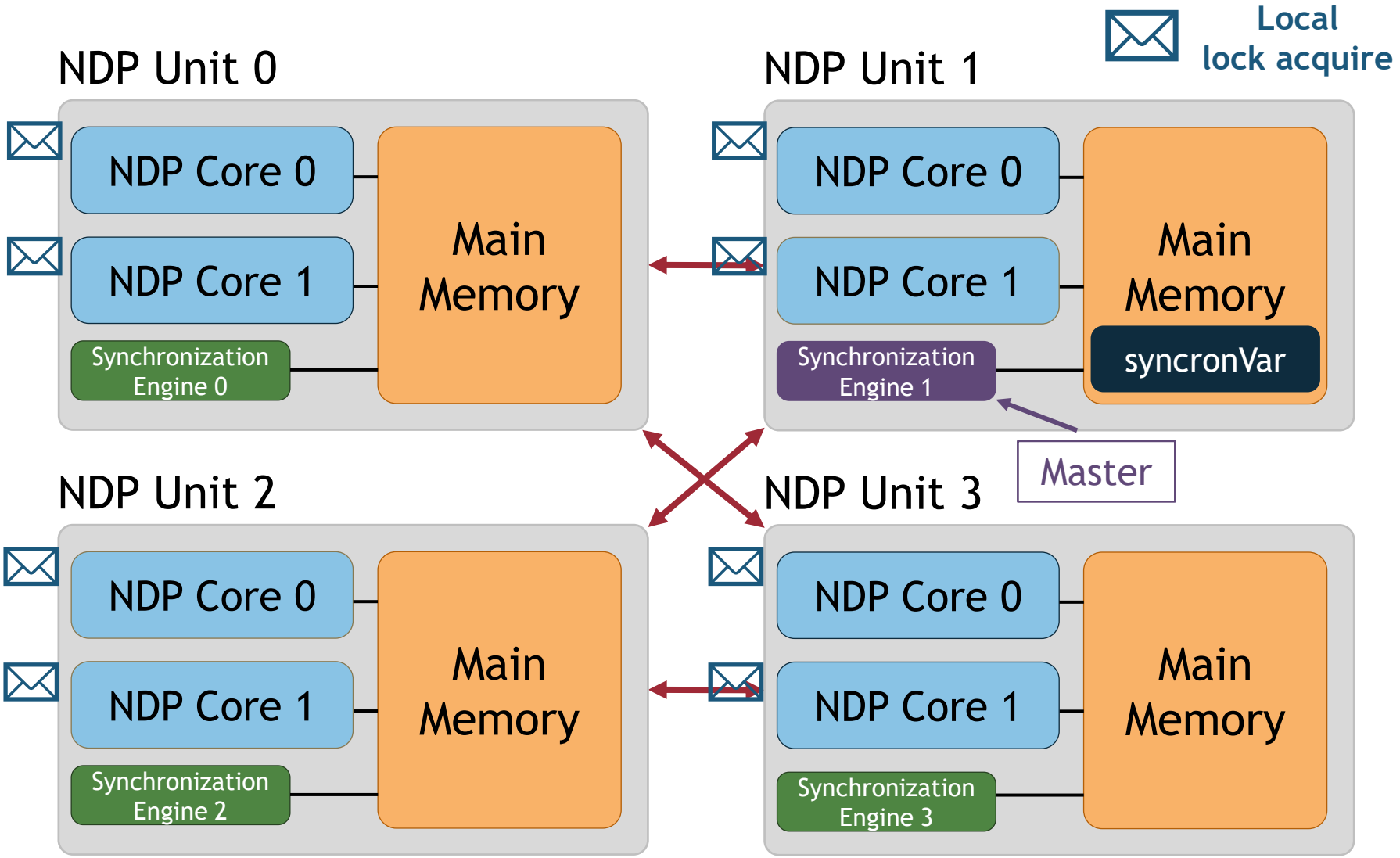


- ✓ No Costly Memory Accesses
- ✓ Low Latency

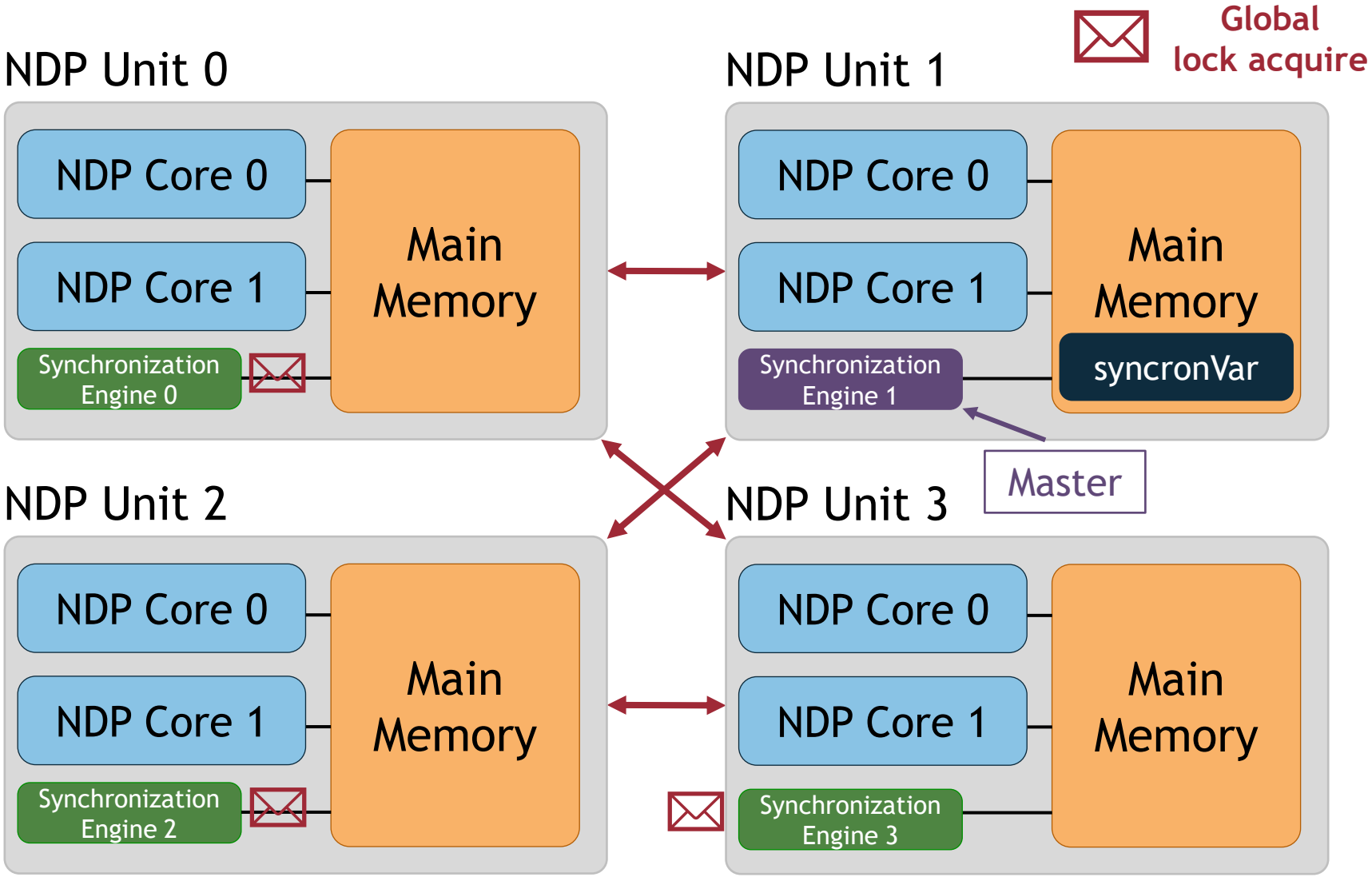
3. Hierarchical Communication



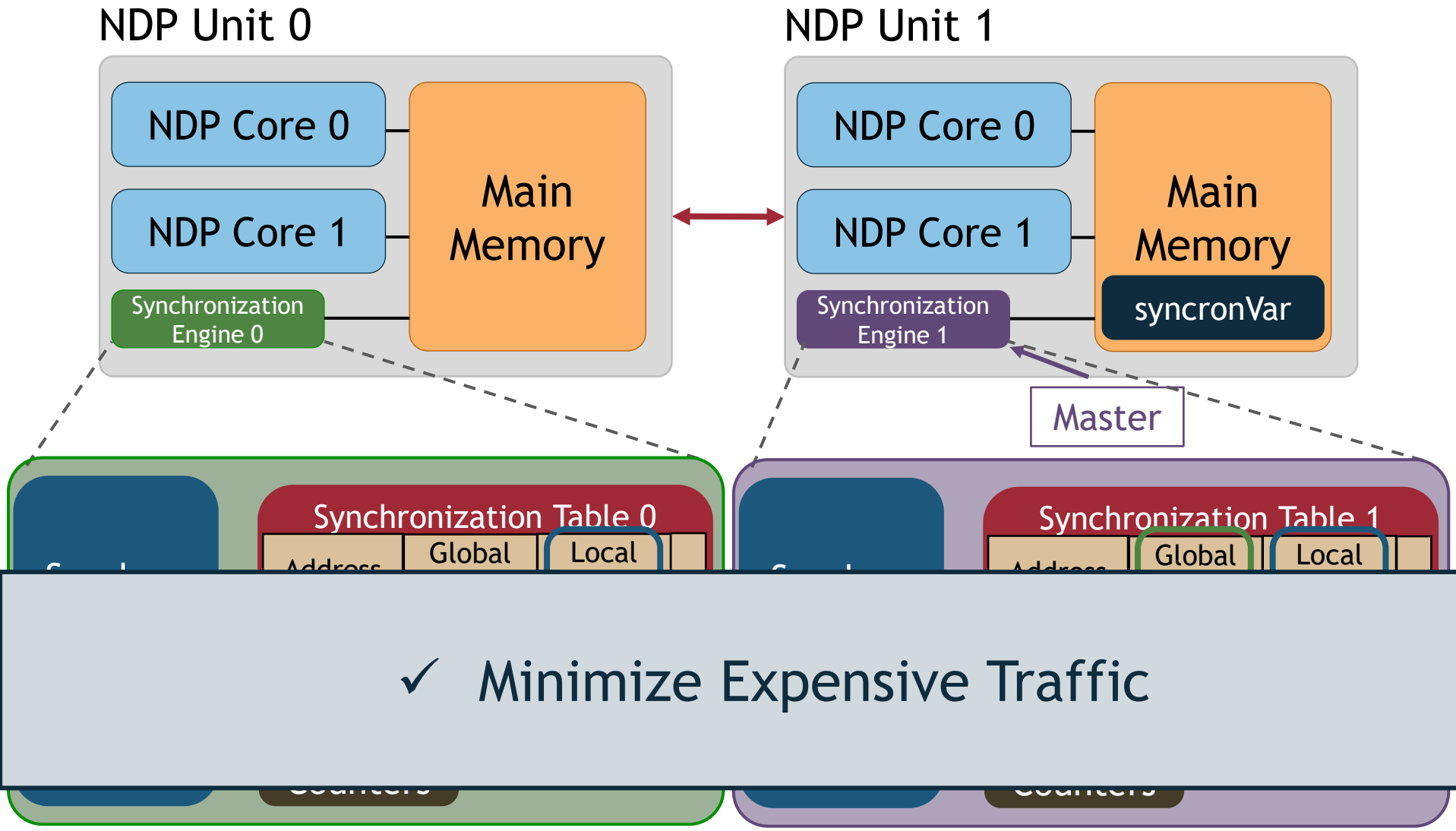
3. Hierarchical Communication



3. Hierarchical Communication

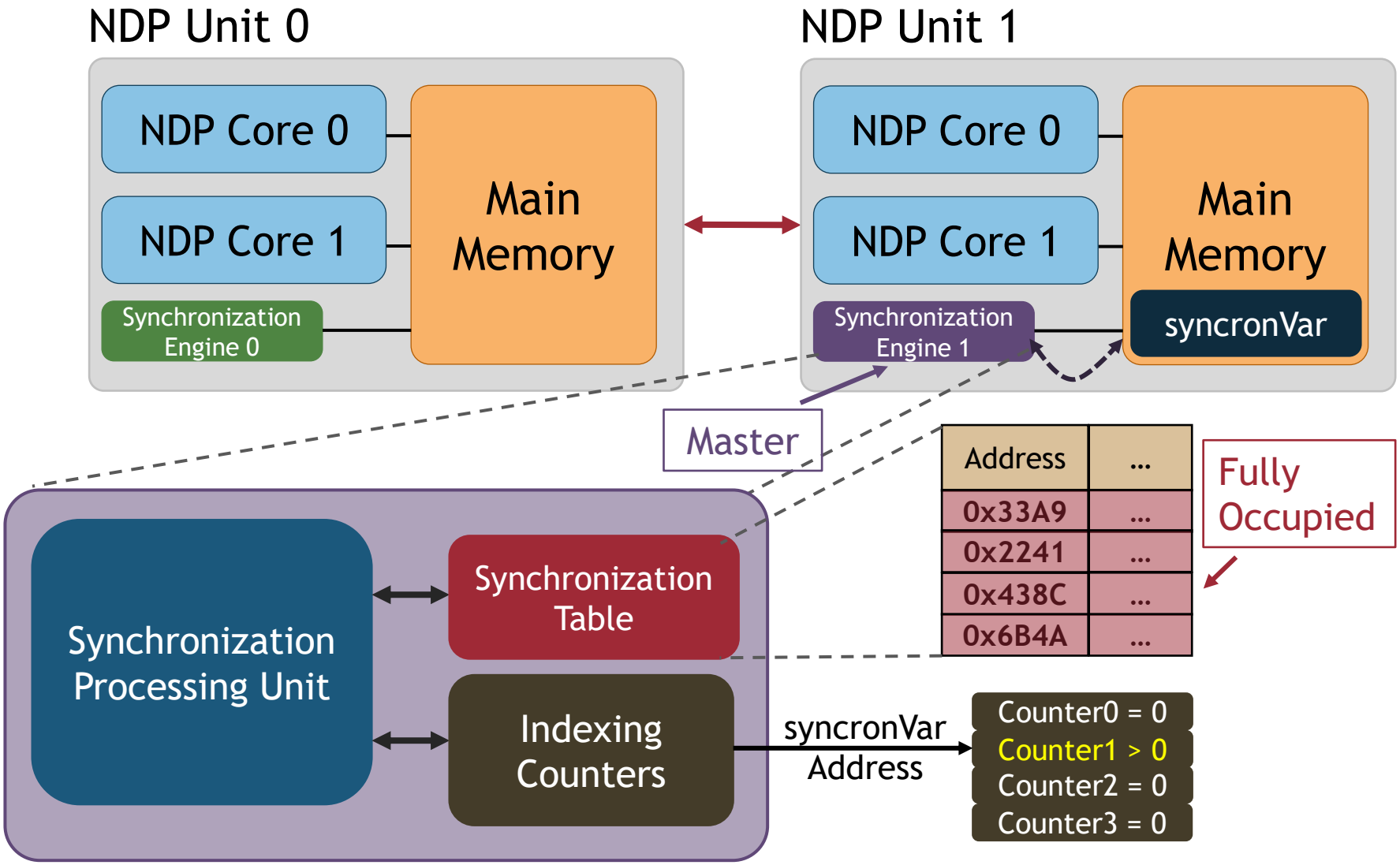


3. Hierarchical Communication

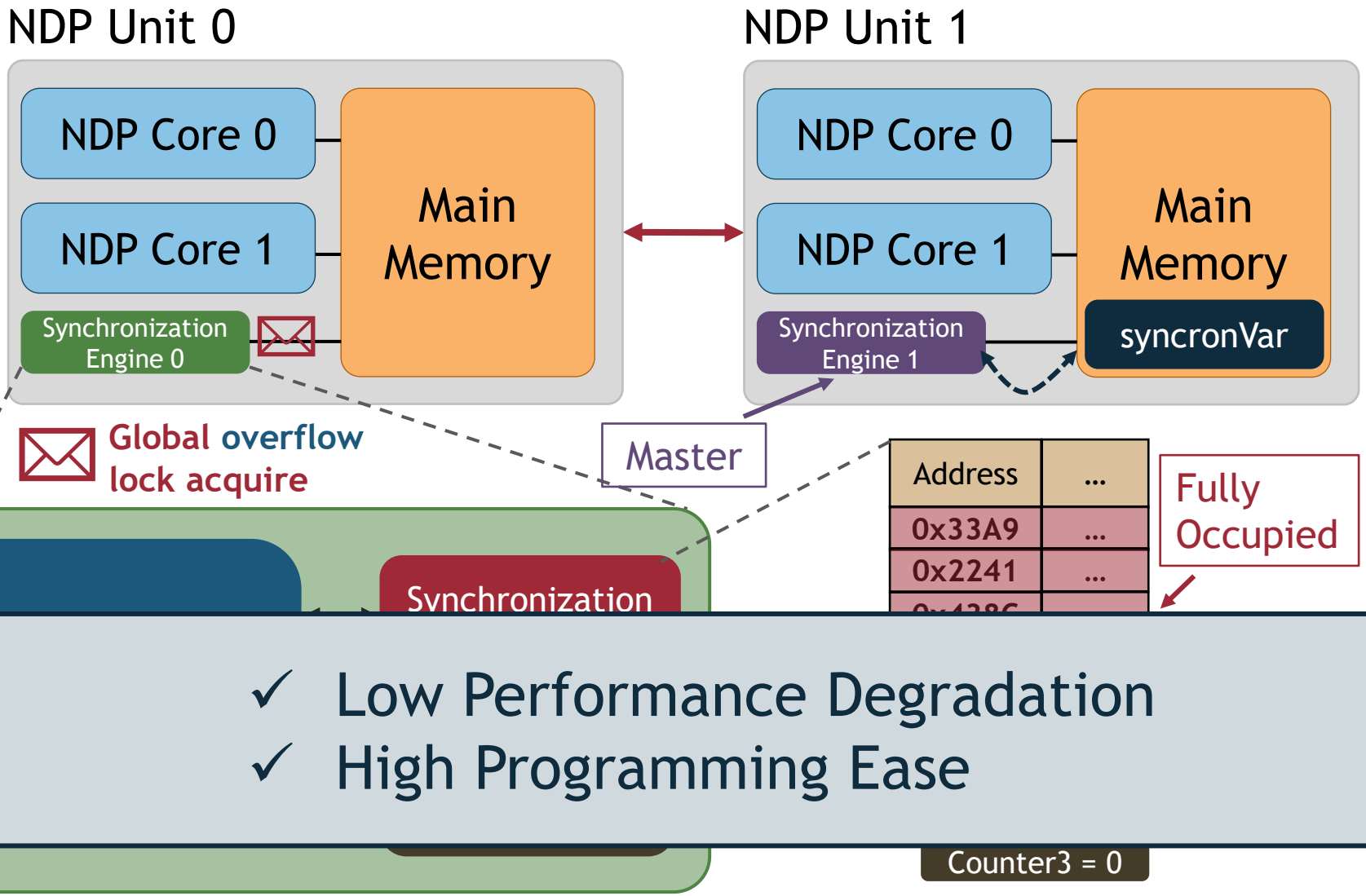


✓ Minimize Expensive Traffic

4. Integrated Overflow Management



4. Integrated Overflow Management



- ✓ Low Performance Degradation
- ✓ High Programming Ease

SynCron's Supported Primitives

Lock primitive
- lock_acquire()
- lock_release()

Barrier
- ba
- ba

Semaphore
- sem_wait()
- sem_post()

Condition variable primitive
- cond_wait()
- cond_signal()
- cond_broadcast()

Synchronization Metadata:
i. Queueing list
ii. Condition to be satisfied

SynCron's Supported Primitives

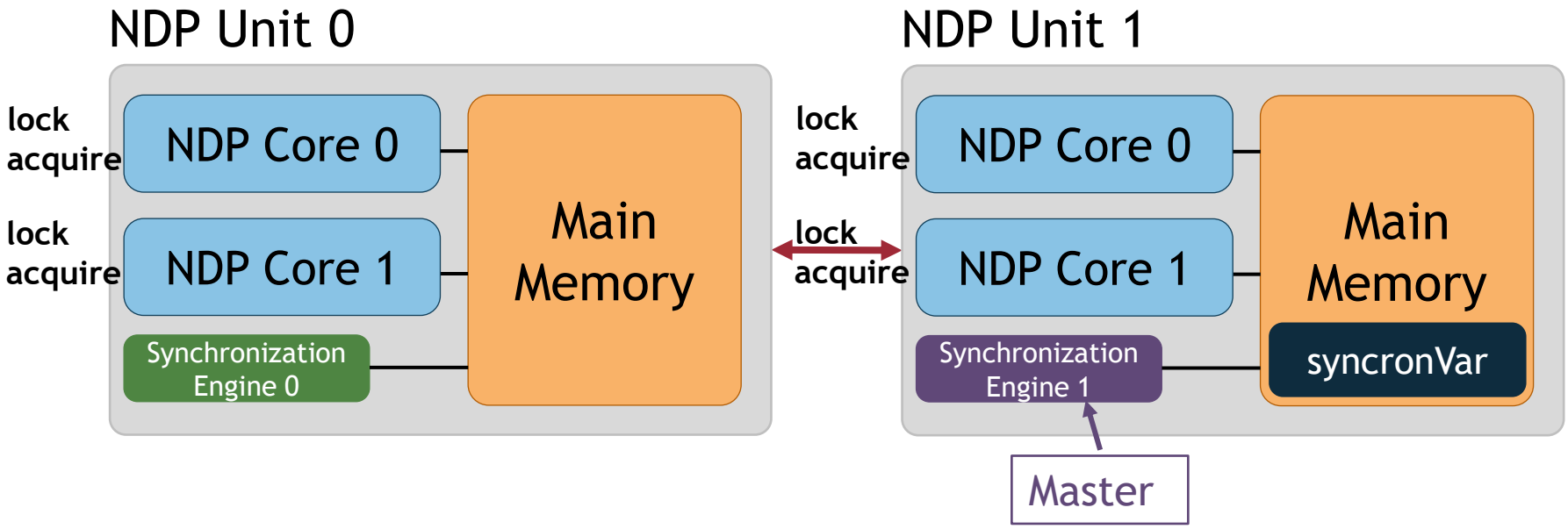
Lock primitive
- lock_acquire()
- lock_release()

Barrier primitive
- barrier_wait_within_NDP_unit()
- barrier_wait_across_NDP_units()

Semaphore primitive
- sem_wait()
- sem_post()

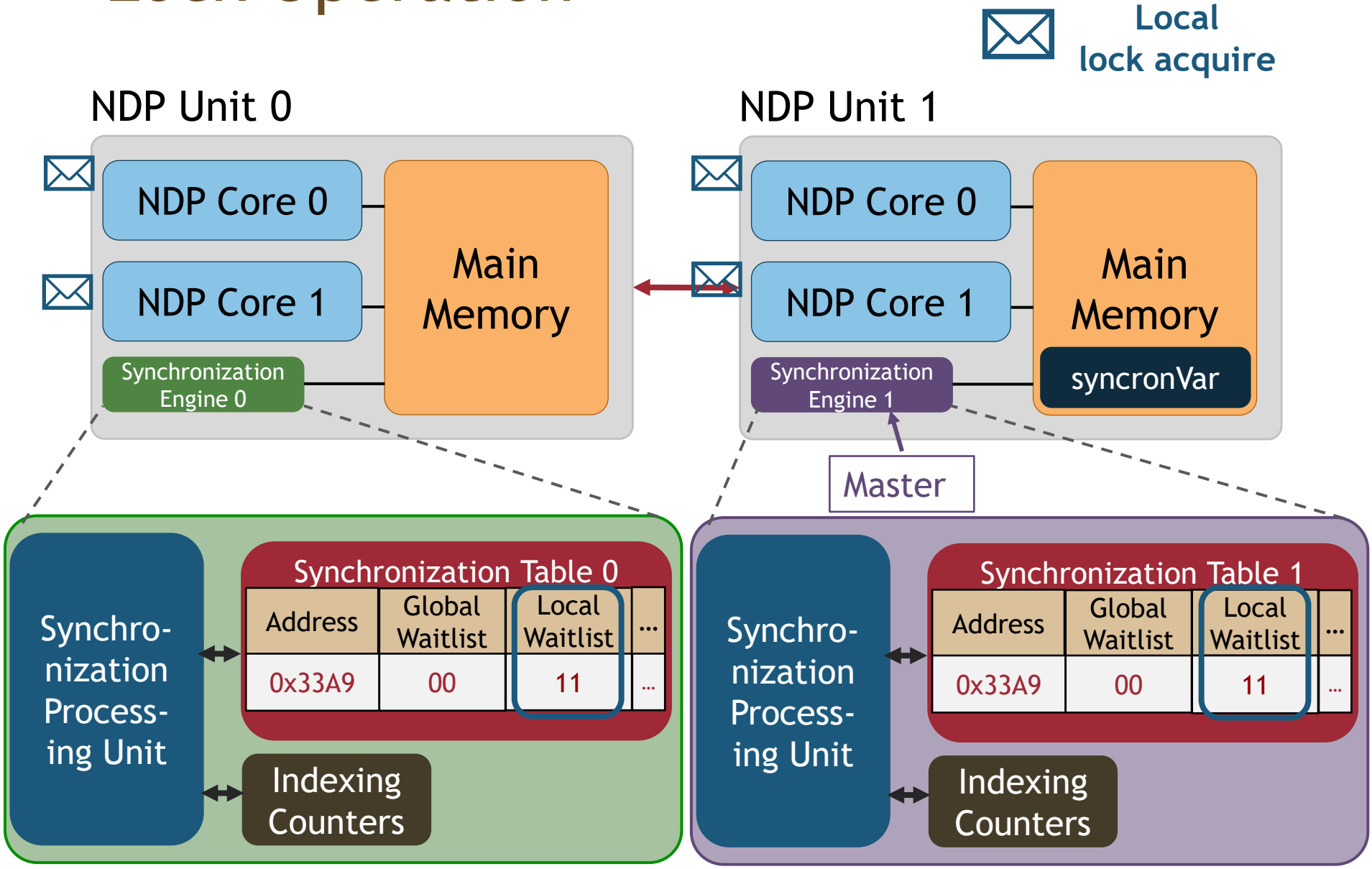
Condition variable primitive
- cond_wait()
- cond_signal()
- cond_broadcast()

Lock Operation

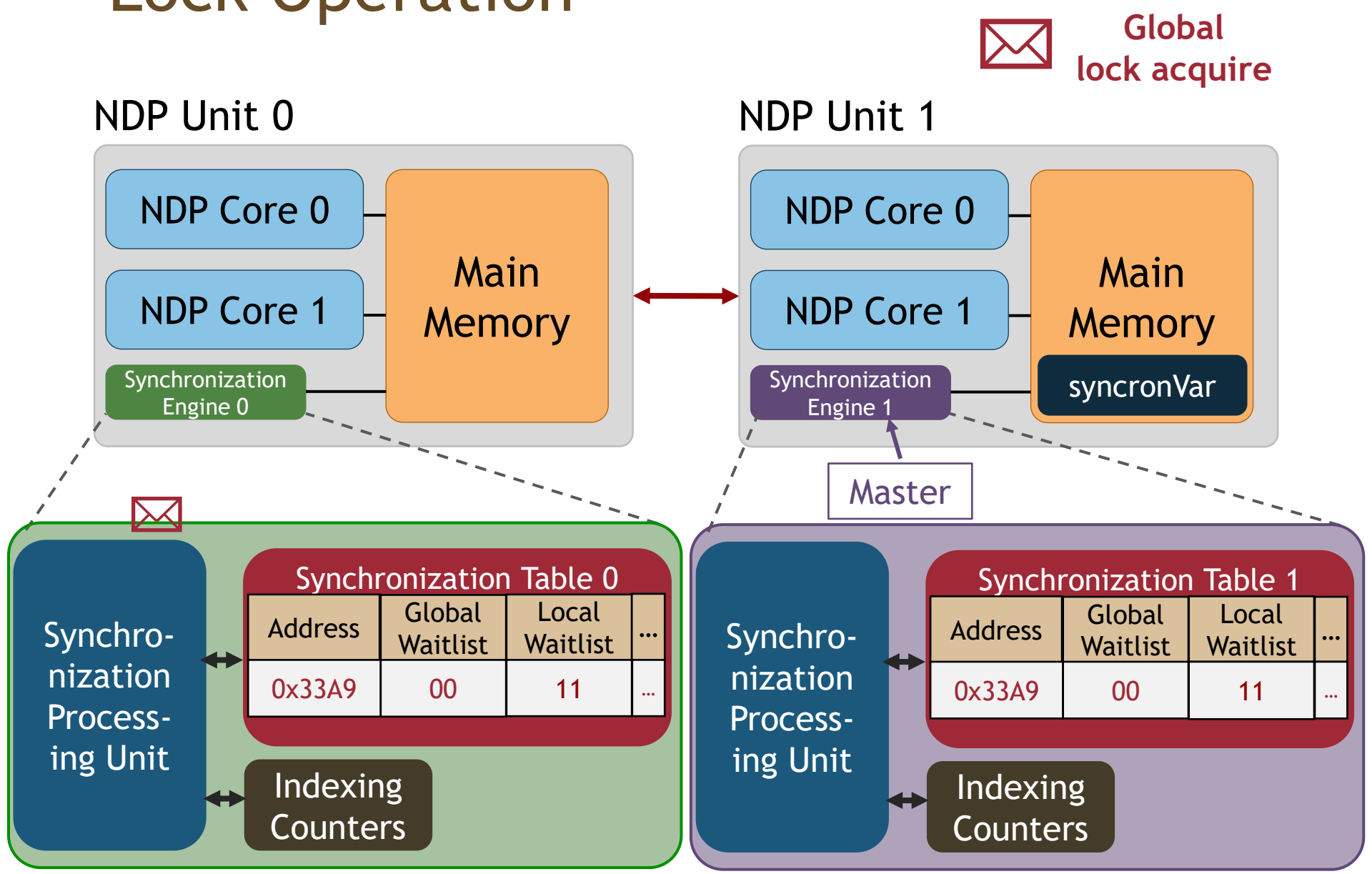


All NDP cores compete for the same lock variable

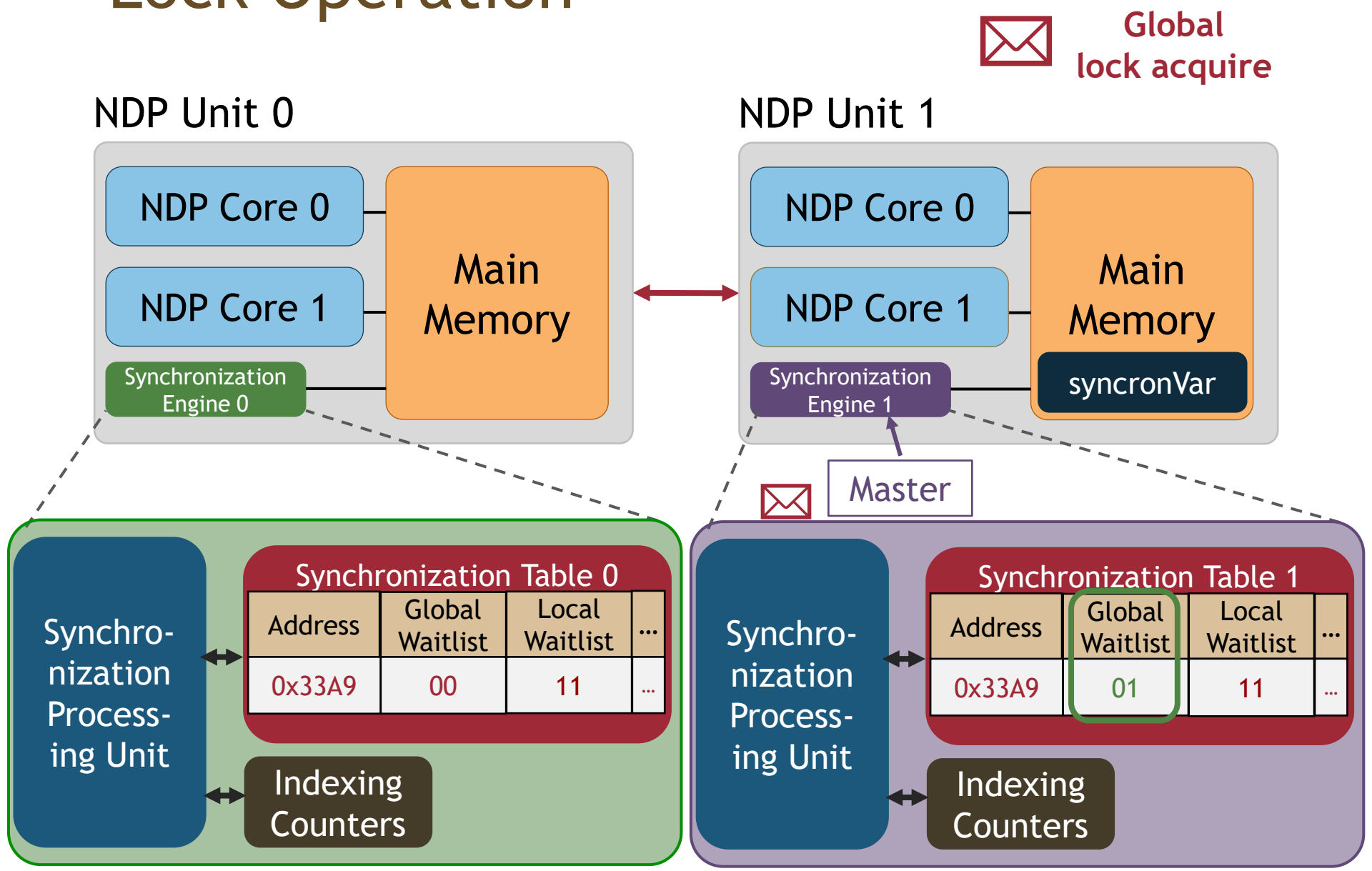
Lock Operation



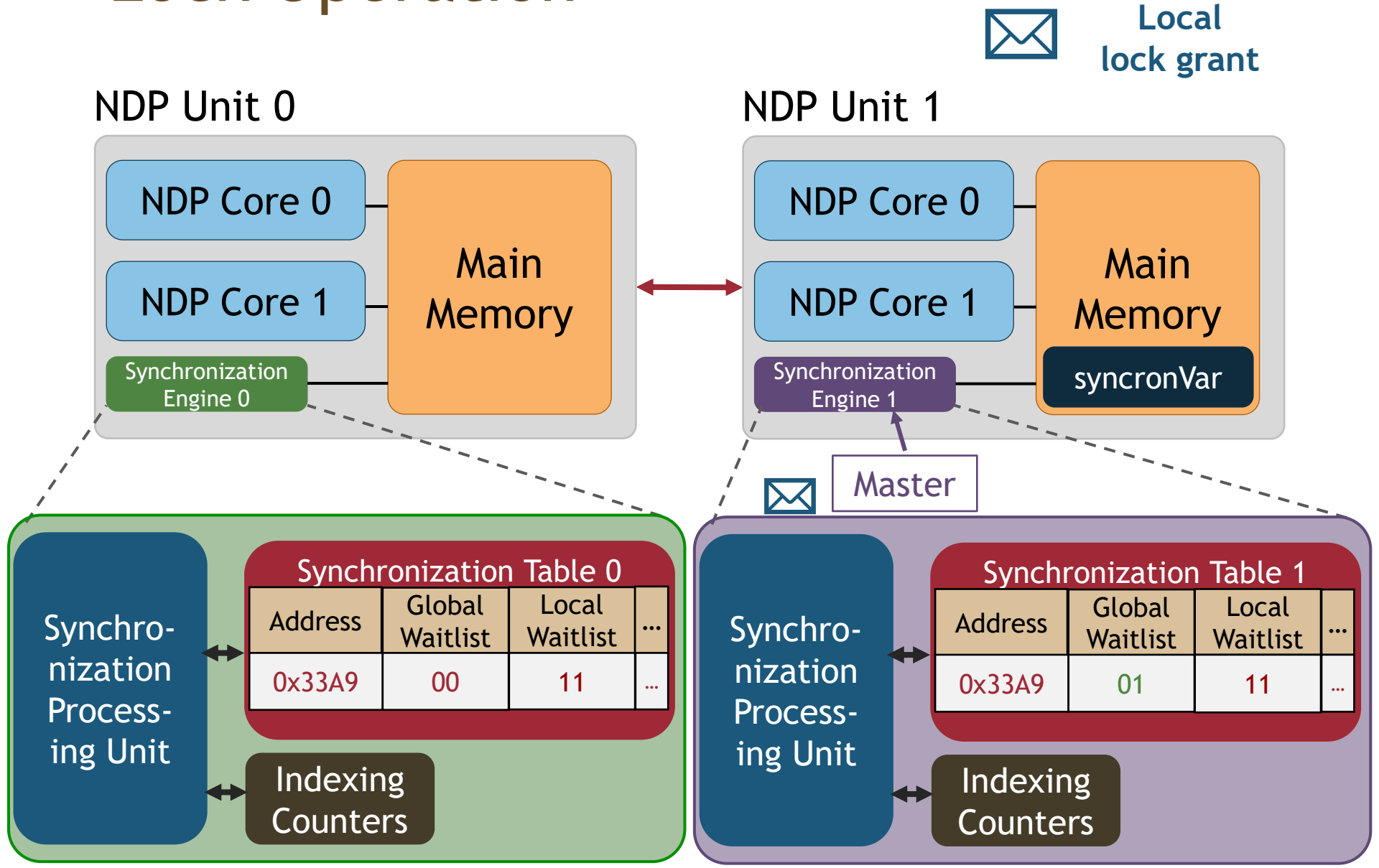
Lock Operation



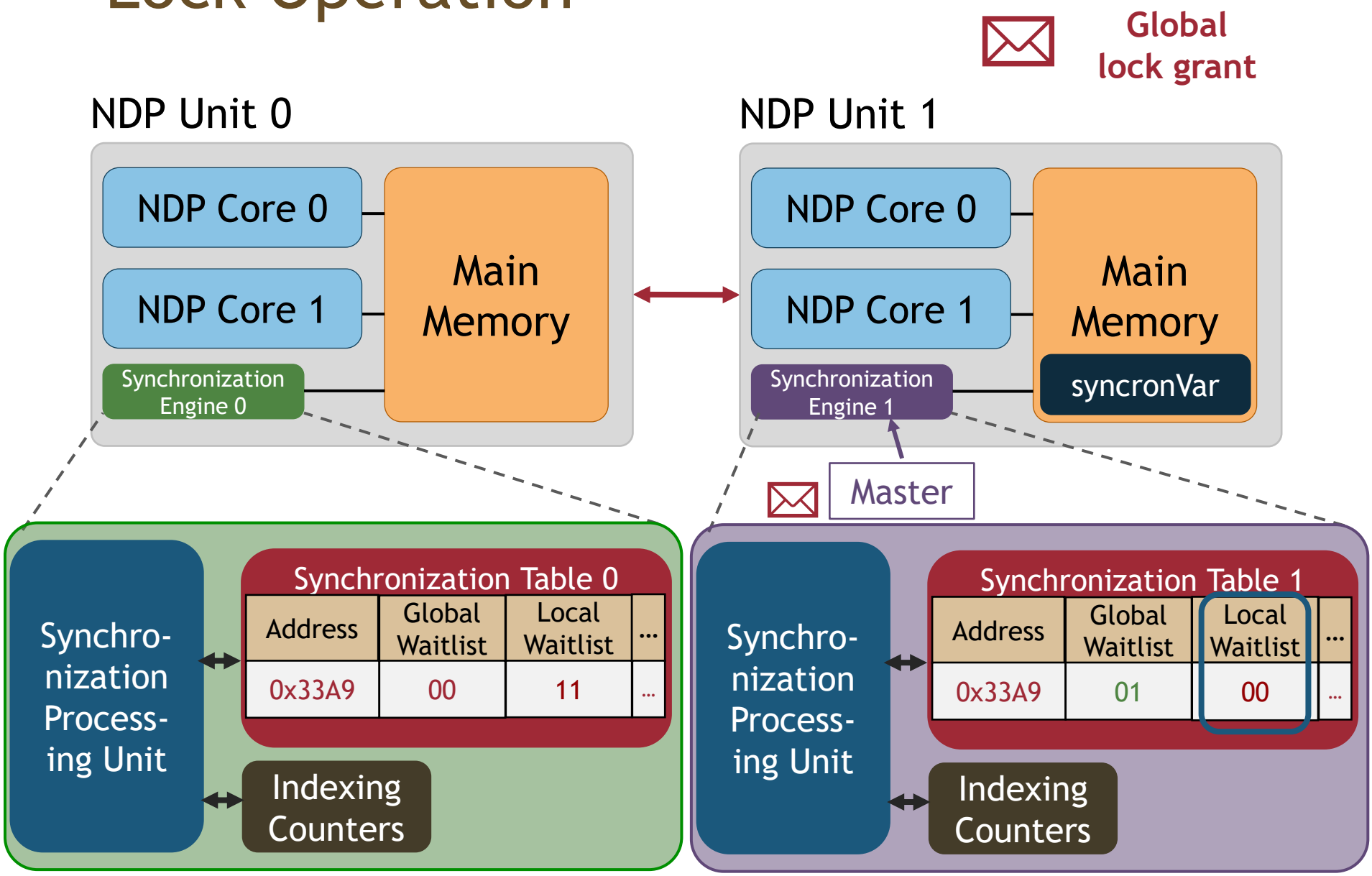
Lock Operation



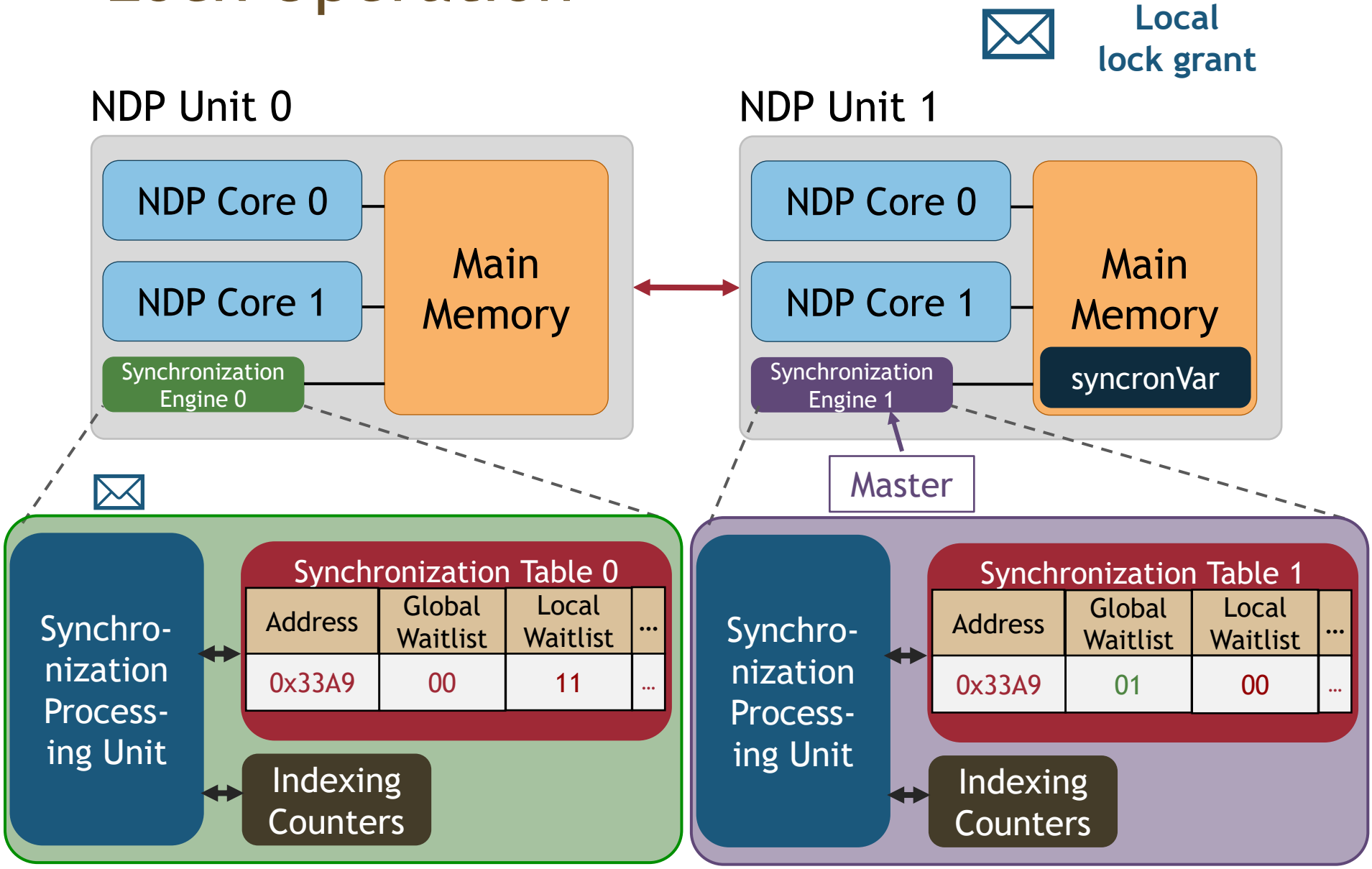
Lock Operation



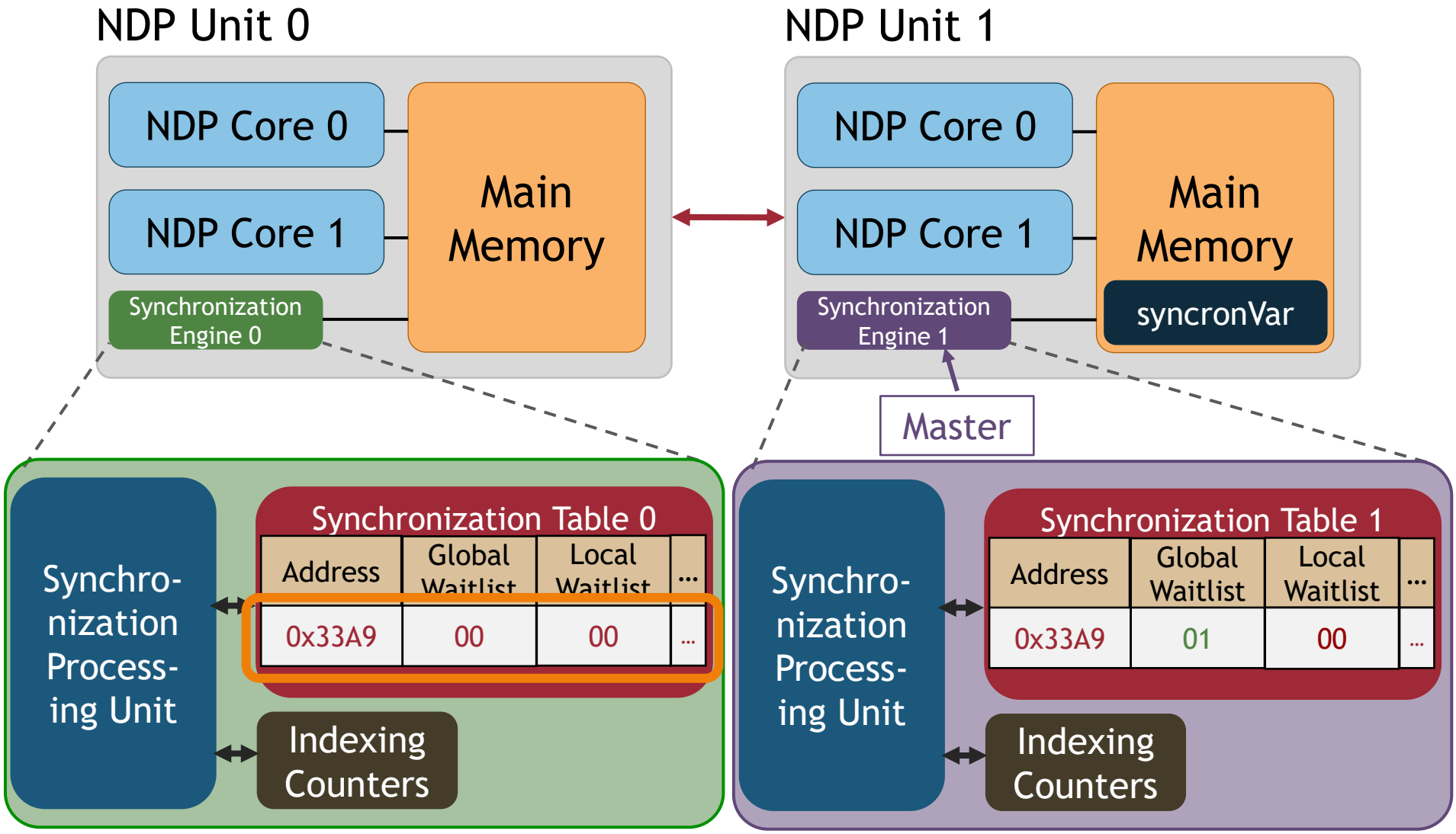
Lock Operation



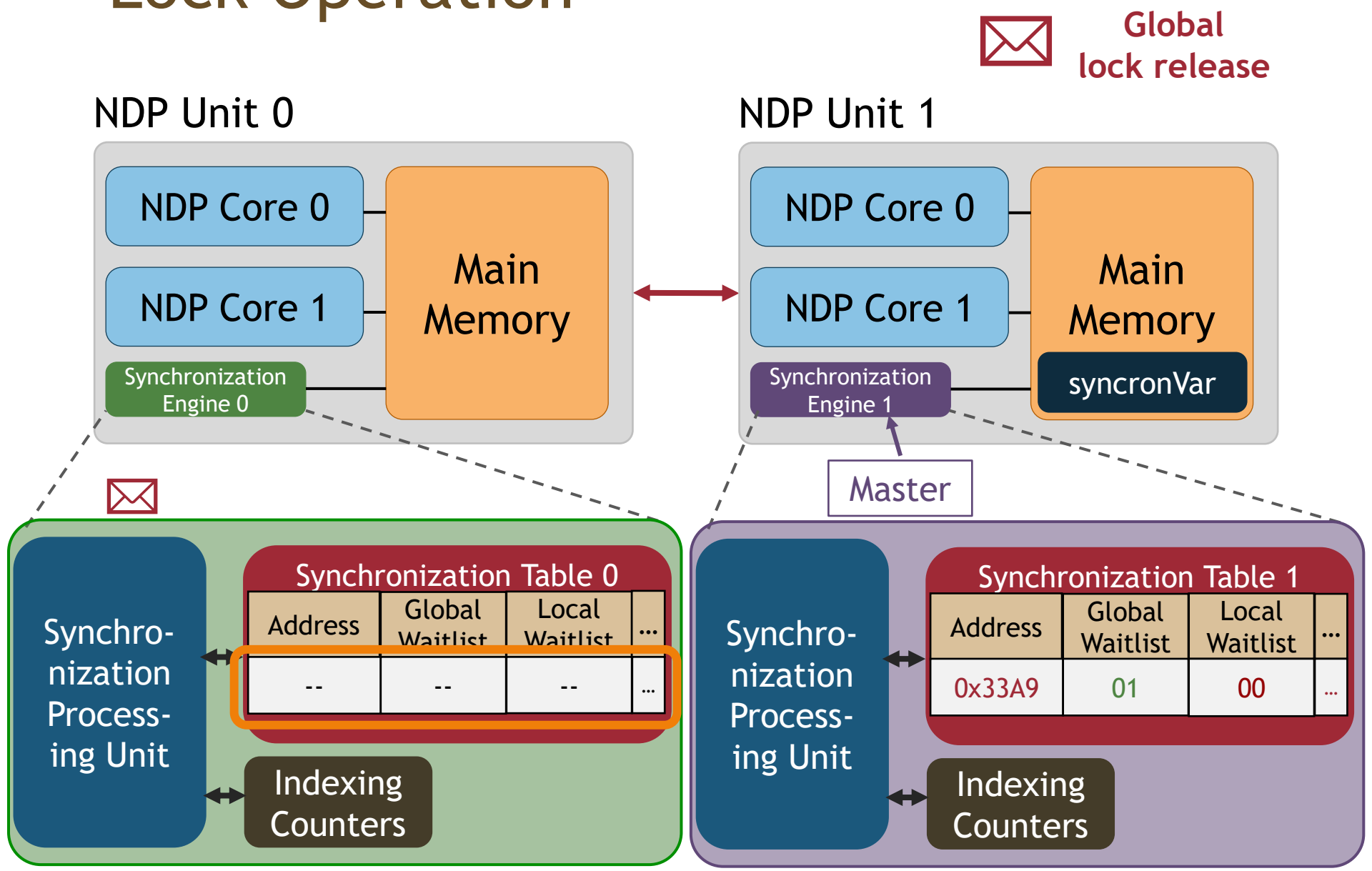
Lock Operation



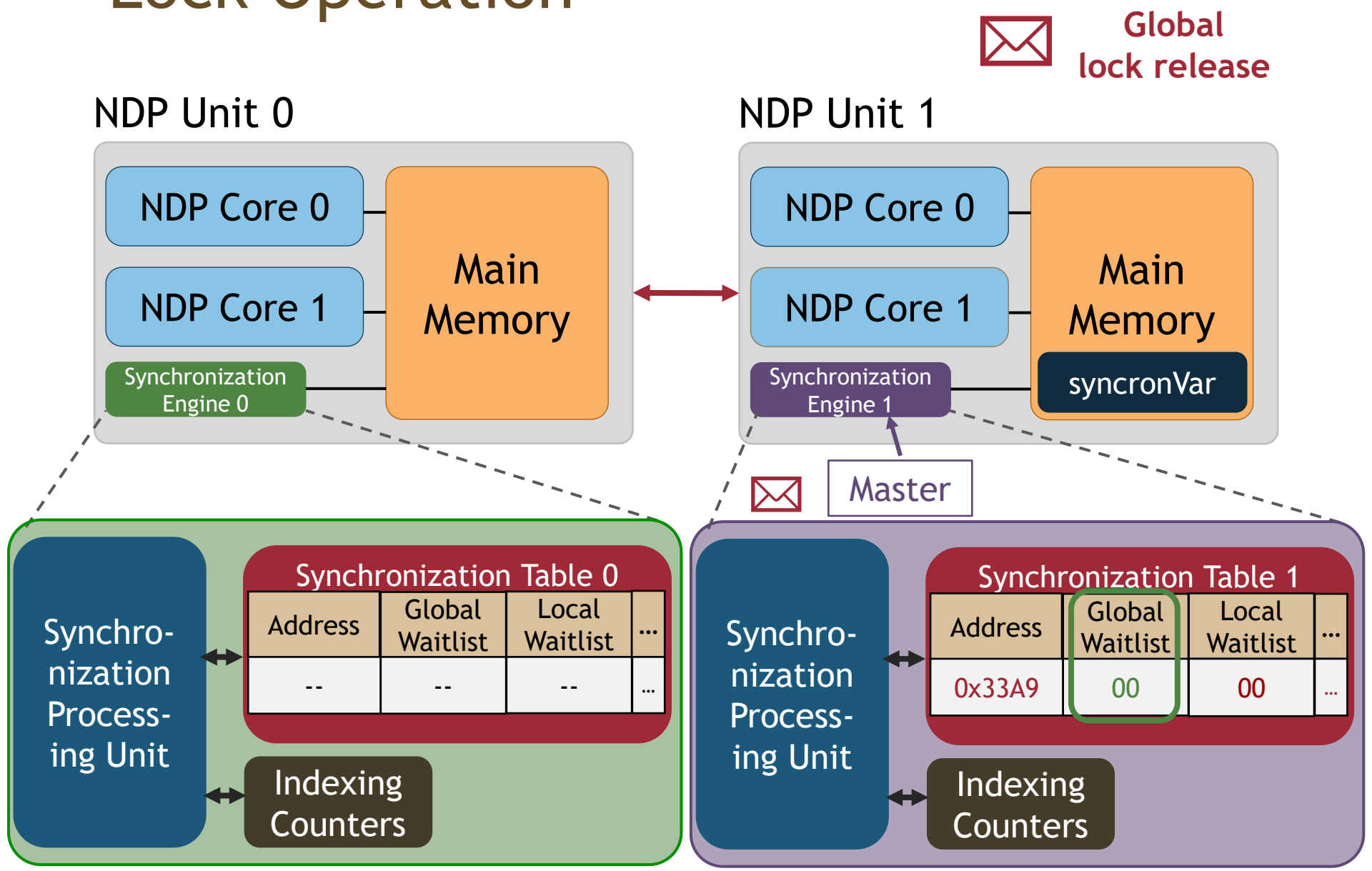
Lock Operation



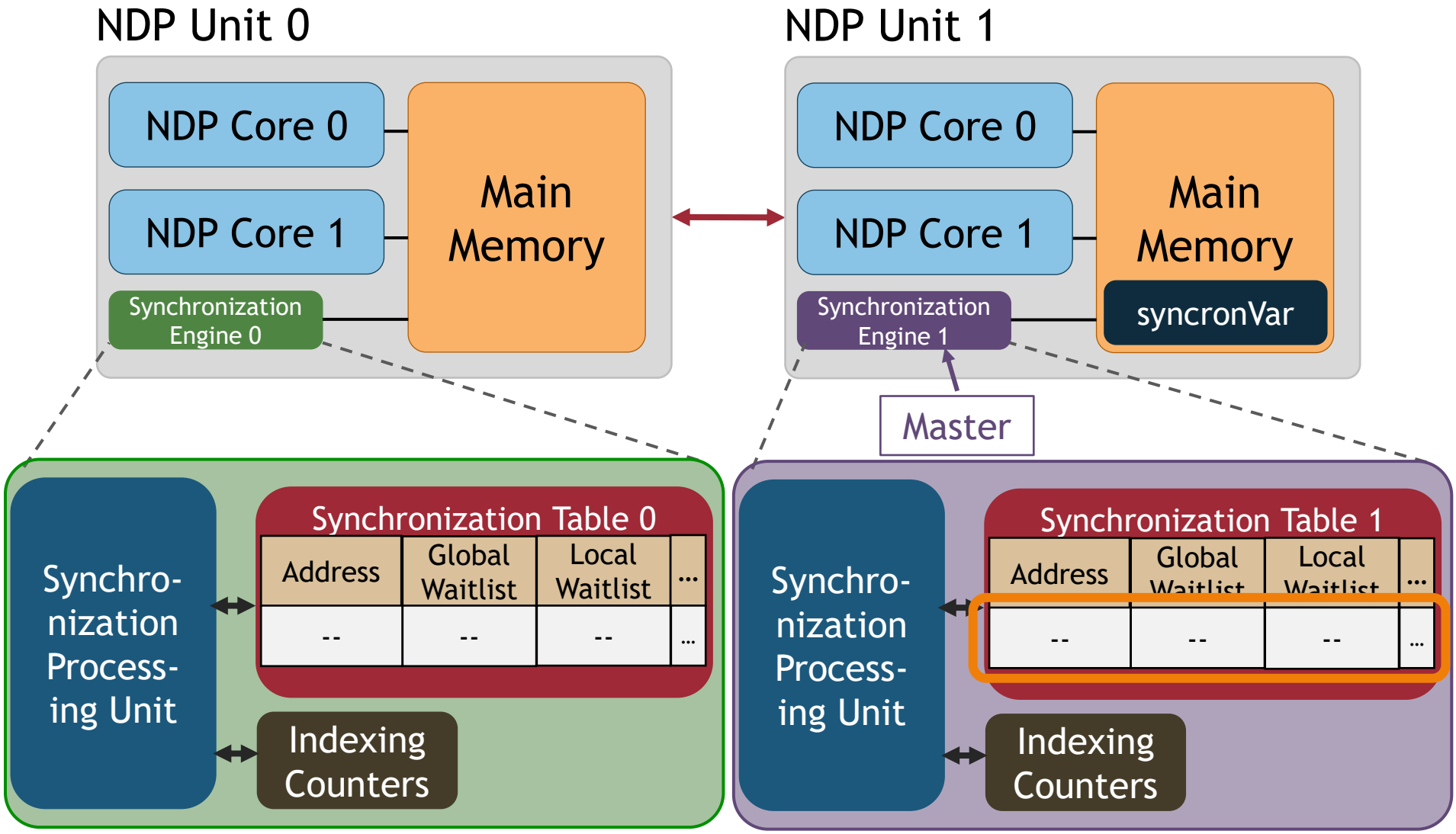
Lock Operation



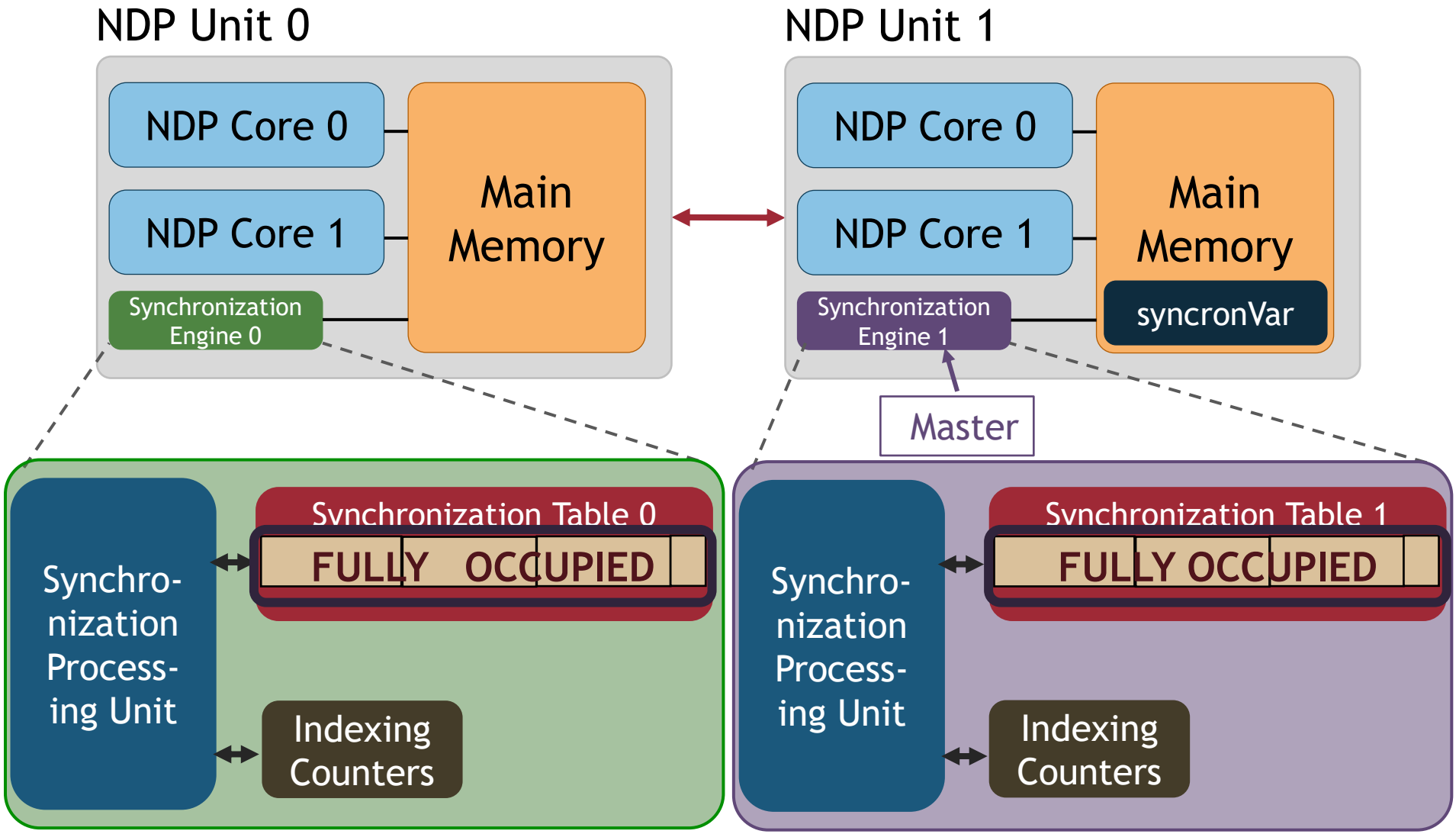
Lock Operation



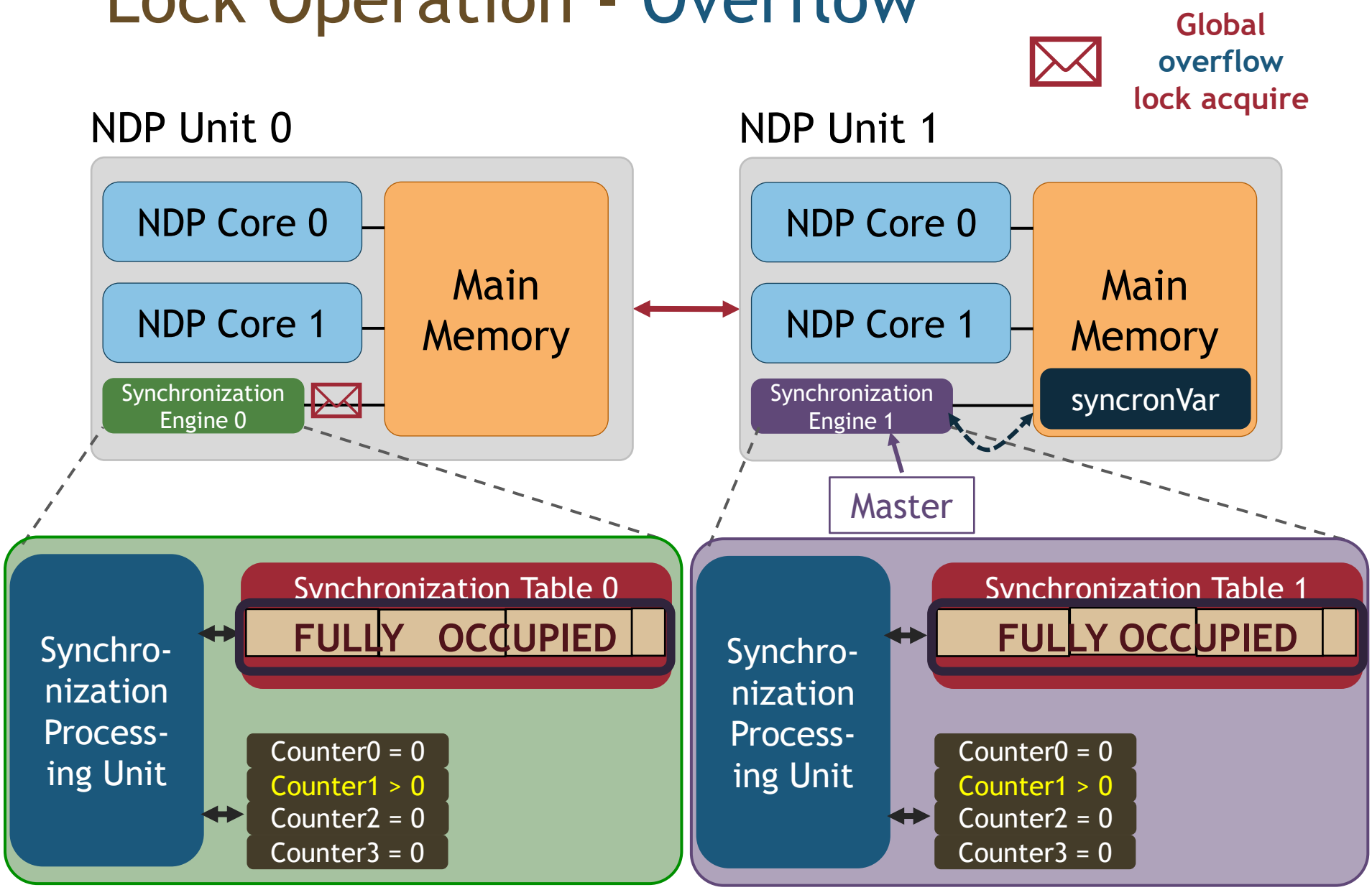
Lock Operation



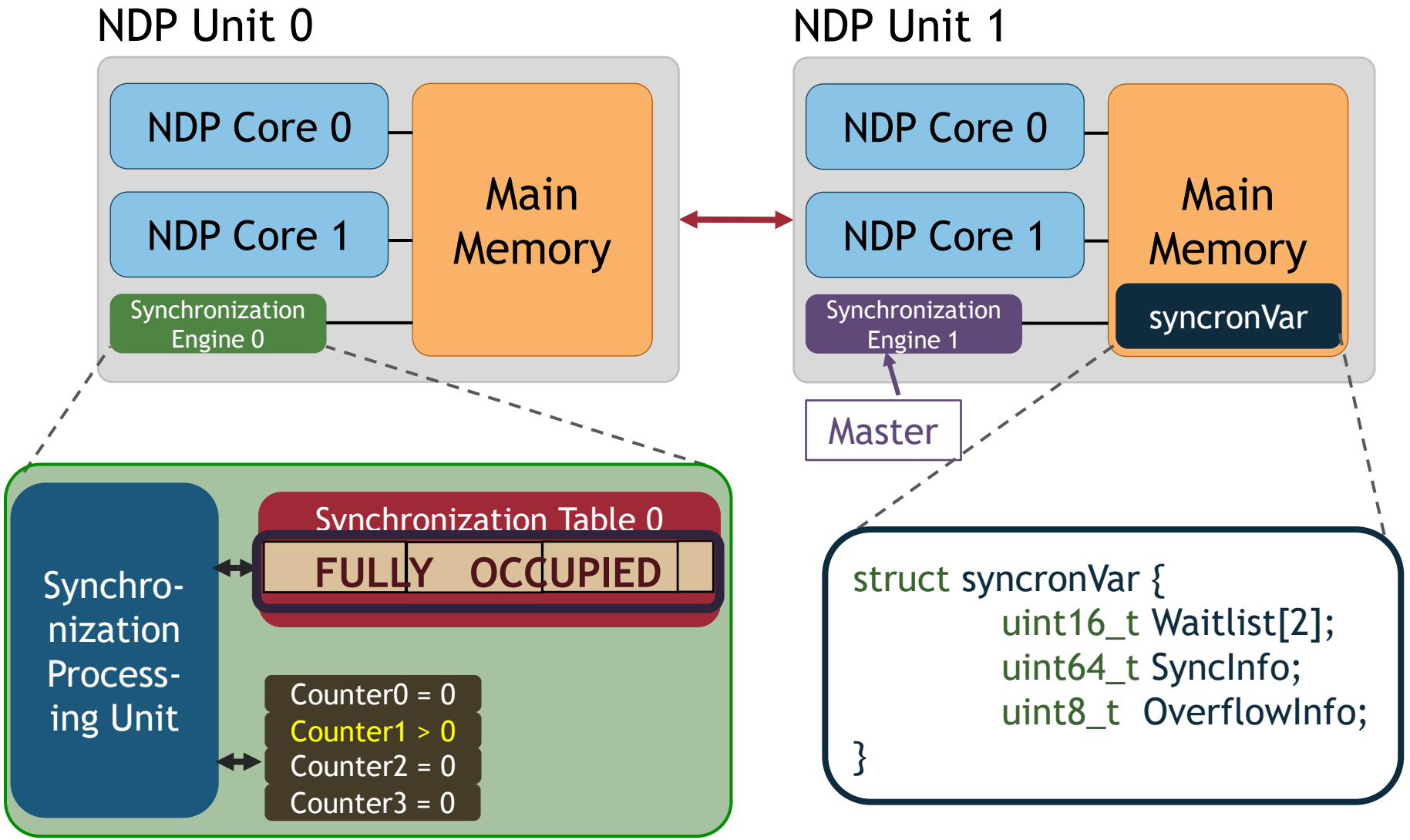
Lock Operation - Overflow



Lock Operation - Overflow



Lock Operation - Overflow



Outline

NDP Synchronization Solution Space

Our Mechanism: SynCron

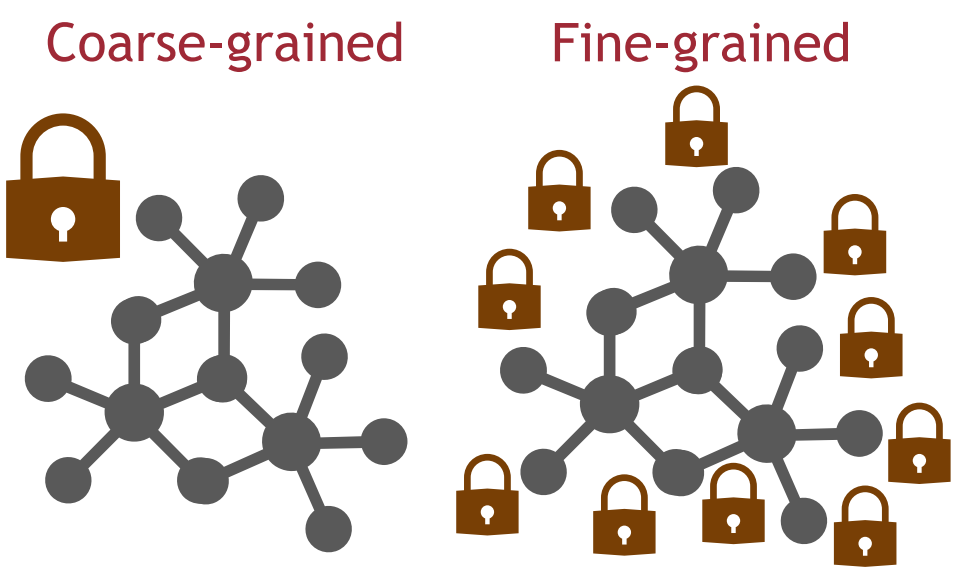
Evaluation

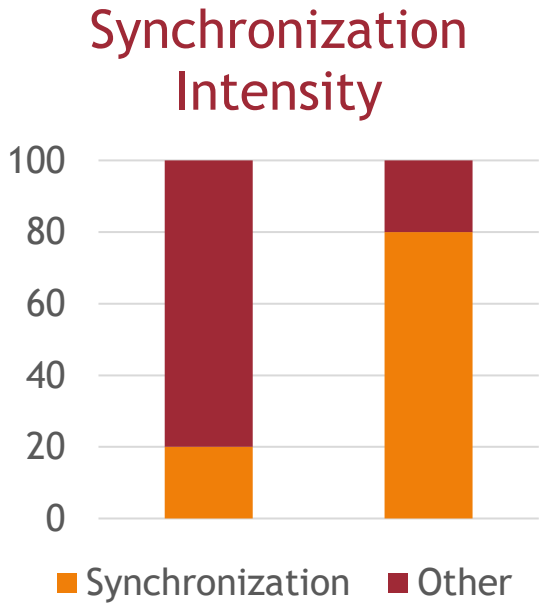
Evaluation Methodology

- Simulators:
  - **Zsim** [Sanchez+, ISCA'13]
  - **Ramulator** [Kim+, CAL'15]
- System Configuration:
  - **4x NDP units of 16 in-order cores**
  - 16KB L1 Data + Instr. Cache
  - **4GB HBM memory**
- SynCron's Default Parameters:
  - Synchronization Processing Unit @1GHz
  - 12-cycle worst-case latency for a message to be served [Aladdin]
  - 64 entries in Synchronization Table, 1-cycle latency [CACTI]
  - 256 entries in indexing counters 2-cycle latency [CACTI]

Workloads

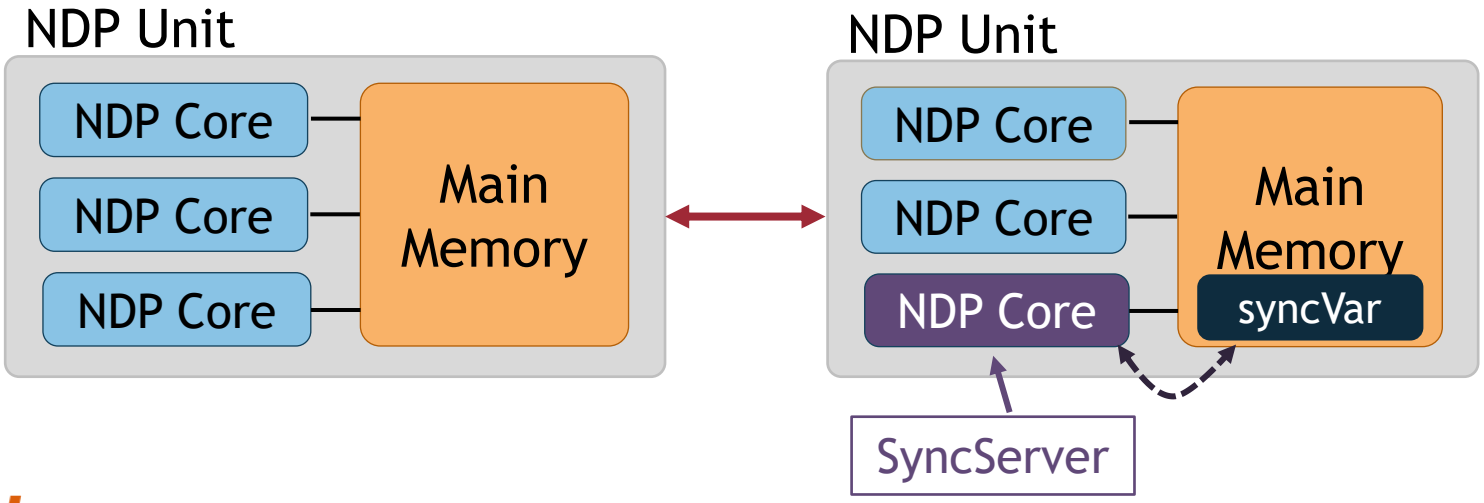
- 9x **Pointer-chasing** Data Structures from ASCYLIB [David+, ASPLOS'15]
- 6x **Graph Applications** from Crono [Ahmad+, IISWC'15]
- **Time Series Analysis** from Matrix Profile [Yeh+, ICDM'16]





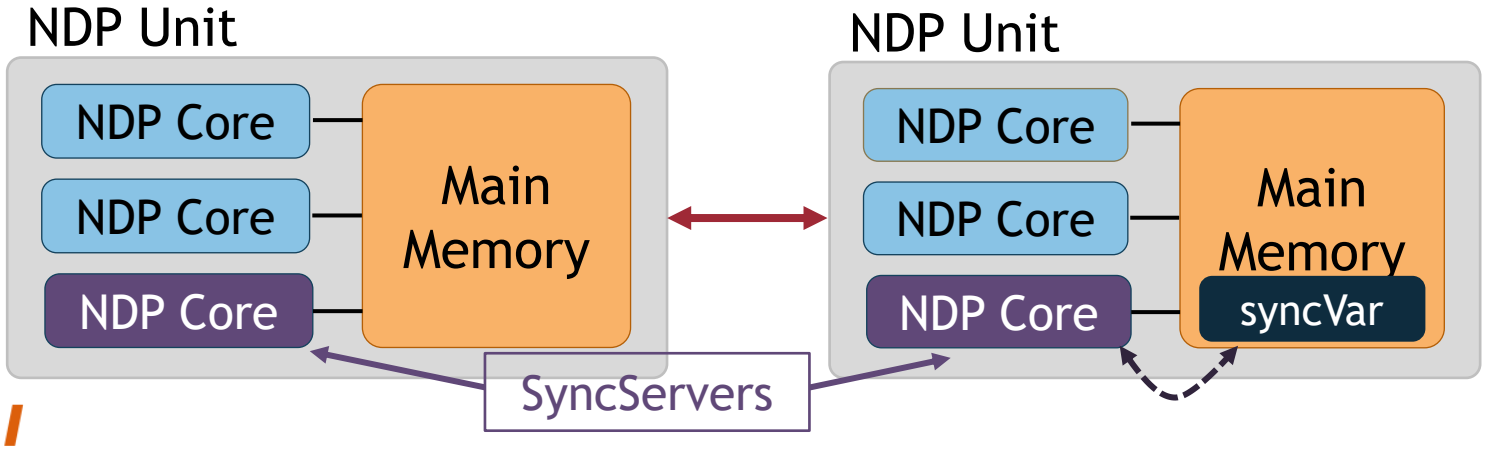
Comparison Points for SynCron

1. SynCron
2. Central [Ahn+, ISCA'15]:
   - Synchronization Server: One NDP core of the NDP system
   - Centralized hardware message-passing communication



Comparison Points for SynCron

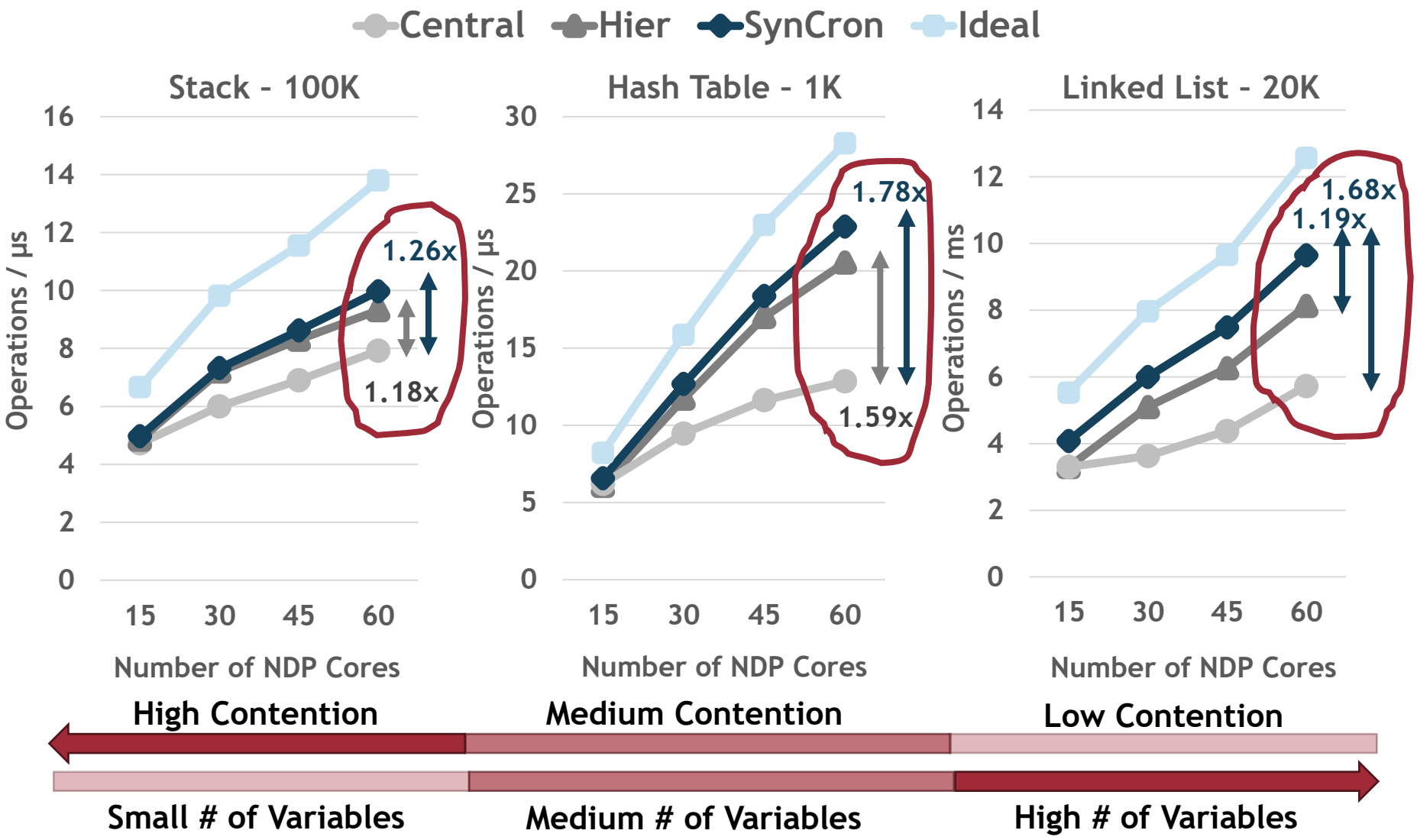
1. SynCron
2. Central [Ahn+, ISCA'15]:
   - Synchronization Server: One NDP core of the NDP system
   - Centralized message-passing communication
3. Hier [Gao+, PACT'15 / Tang+, ASPLOS'19]:
   - Synchronization Servers: One NDP core per NDP unit
   - Hierarchical message-passing communication



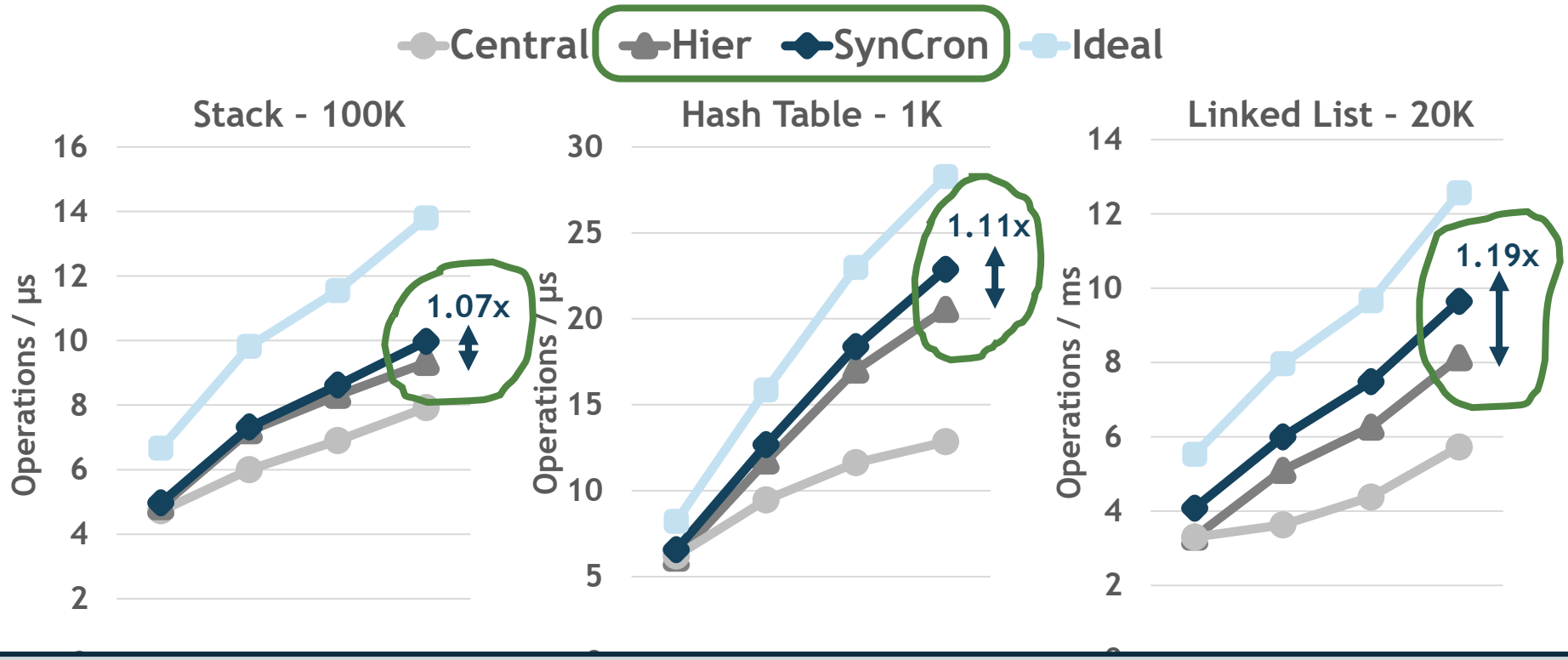
Comparison Points for SynCron

1. SynCron
2. Central [Ahn+, ISCA'15]:
   - Synchronization Server: One NDP core of the NDP system
   - Centralized hardware message-passing communication
3. Hier [Gao+, PACT'15 / Tang+, ASPLOS'19]:
   - Synchronization Servers: One NDP core per NDP unit
   - Hierarchical hardware message-passing communication
4. Ideal
   - Zero overhead for synchronization

Throughput of Pointer Chasing

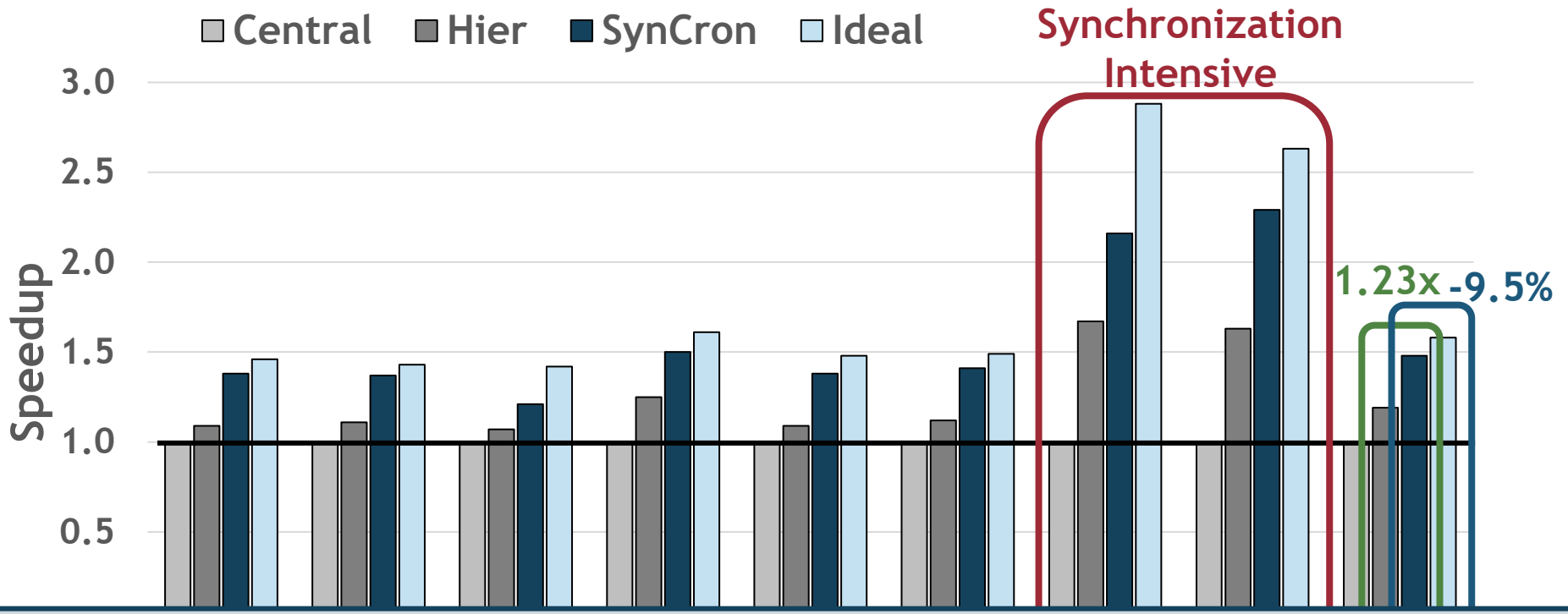


Throughput of Pointer Chasing



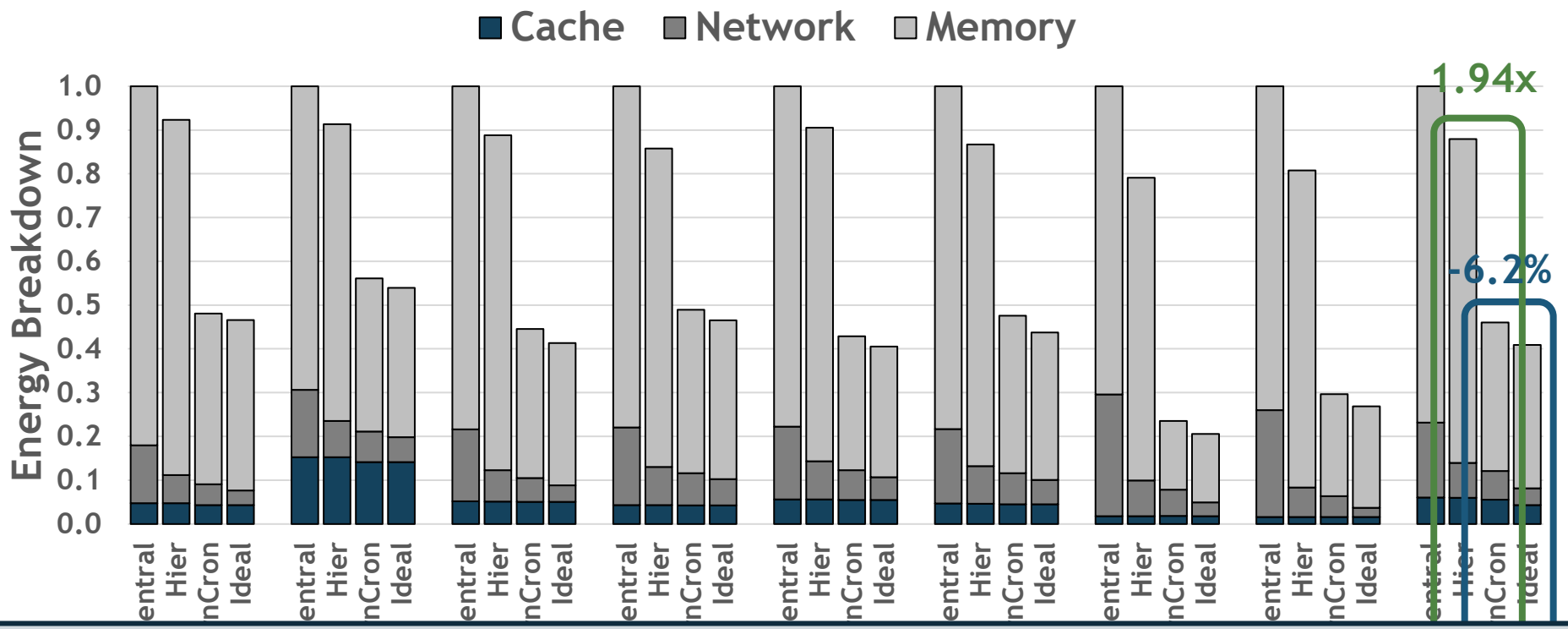
SynCron achieves the **highest throughput** under all contention scenarios

Speedup in Real Applications



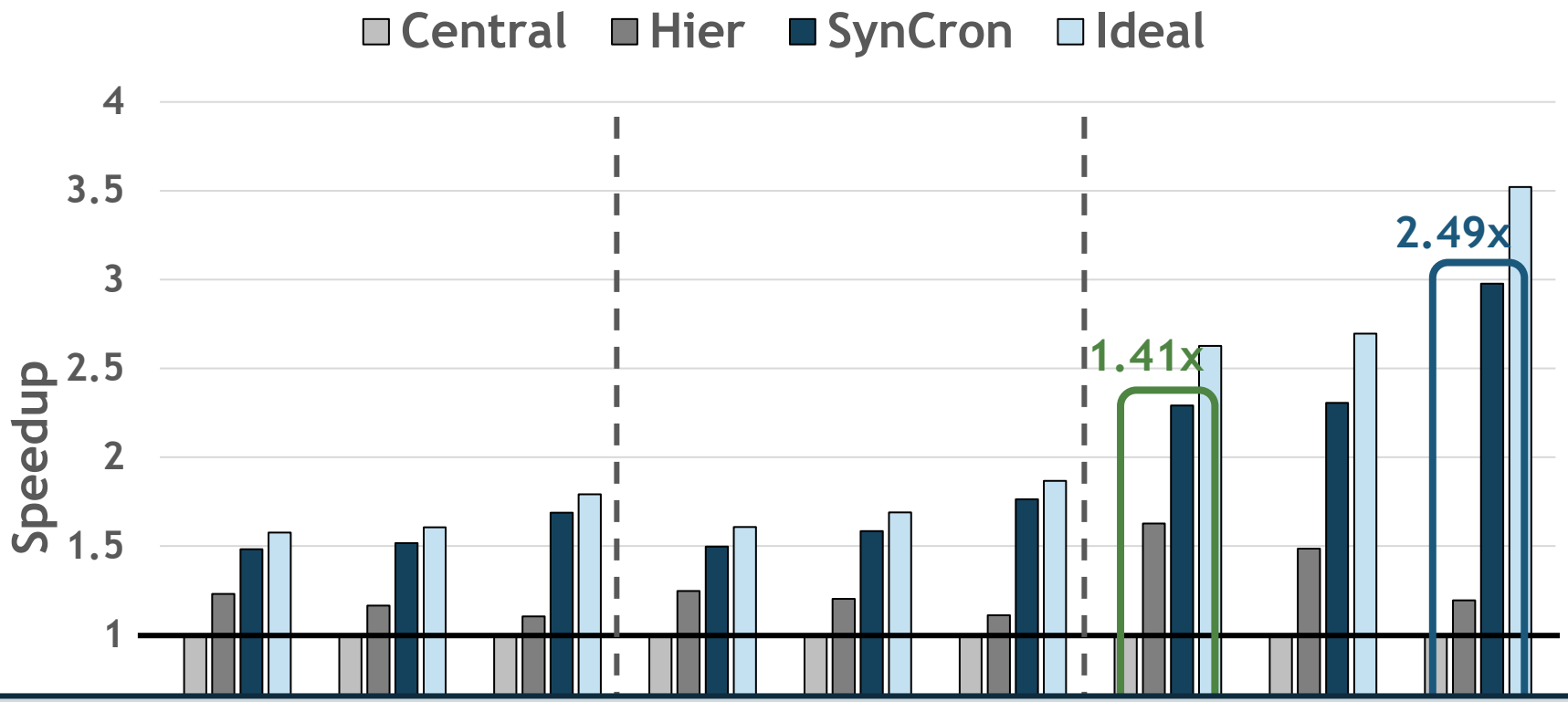
SynCron **performs best** across all **real** applications

System Energy in Real Applications



SynCron reduces system energy significantly

Memory technologies



SynCron is orthogonal
to the memory technology used

Area and Power Overheads

	Synchronization Engine	ARM Cortex A7
Technology	40nm	28nm
Area 9.78%	Total: 0.0461mm2	Total: 0.45mm2
Power 2.70%	2.7mW	100mW

SynCron has **low area** and **power** overheads

Sensitivity Studies

- Various data placement techniques
- Various transfer latencies on links across NDP units
- Overflow management cost
- Various sizes for the Synchronization Table

SynCron is **effective** for a **wide variety** of configurations

Summary & Conclusion

- Synchronization is a **major system challenge** for NDP systems
- **Prior** schemes are **not suitable** or **efficient** for NDP systems
- **SynCron** is the **first end-to-end** synchronization solution for NDP architectures
- Syncron consists of **four** key techniques:
  i. **Hardware support** for synchronization acceleration
  ii. **Direct buffering** of synchronization variables
  iii. **Hierarchical** message-passing **communication**
  iv. Integrated hardware-only **overflow management**
- SynCron's benefits: **90.5%** and **93.8%** of performance and energy of an **Ideal** zero-overhead scheme
- SynCron is **highly-efficient**, **low-cost**, **easy-to-use**, and **general** to support many synchronization primitives

SynCron
Efficient Synchronization Support for Near-Data-Processing Architectures

Ramulator Course
06.05.2022

Christina Giannoula

Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas
Ivan Fernandez, Juan Gómez Luna, Lois Orosa
Nectarios Koziris, Georgios Goumas, Onur Mutlu

SAFARI ETH zürich