SynCron
Efficient Synchronization Support for Near-Data-Processing Architectures

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

SAFARI  
ETH zürich
Near-Data-Processing (NDP) Systems

- Graph Analytics
- Neural Networks
- Recommendation Systems
- Bioinformatics
Synchronization is Necessary

Single Source Shortest Path (SSSP)

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

NDP Unit

- NDP Core
- NDP Core
- NDP Core

Main Memory

Expensive Communication

No Shared Caches

Programmable Core / Accelerator

Private Cache

No Hardware Cache Coherence
SynCron

SynCron’s Benefits:

1. High System Performance
2. Low Hardware Cost
3. Programming Ease
4. General Synchronization Support

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

NDP Synchronization Solution Space

Our Mechanism: SynCron

Evaluation
Baseline NDP Architecture

Synchronization challenges in NDP systems:

1. Lack of hardware cache coherence support
2. Expensive communication across NDP units
3. Lack of a shared level of cache memory
NDP Synchronization Solution Space

(1) Shared Memory
- Hardware Cache Coherence
- Remote Atomics
- Specialized Hardware Support

(2) Message-passing
- Software-based Schemes
- Specialized Hardware Support
NDP Synchronization Solution Space

(1) Shared Memory
- Hardware Cache Coherence
- Remote Atomics
- Specialized Hardware Support

(2) Message-passing
- Software-based Schemes
- Specialized Hardware Support

Lack of hardware cache coherence support

CPUs:
- Hierarchical CLH Locks [EuroPar’06]
- Cohort Locks [TOPC’15]
- Ticket Locks [TOCS’91] …

MPPs:
- QOLB [ASPLOS’89]
NDP Synchronization Solution Space

(1) Shared Memory
- Hardware Cache Coherence
- Remote Atomics
- Specialized Hardware Support
  - GPUs: Fermi GF100 [IEEE Micro’10] ...
  - MPPs: SGI Origin [ISCA’97] Cray T3E [ASPLOS’96] ...
  - CPUs: SSB [ISCA’07] Lock Cache [CASES’01] ...
  - MPPs: Full/Empty Bits [ISCA’83] ...

(2) Message-passing
- Software-based Schemes
- Specialized Hardware Support
  - GPUs: HQL [IPDPS’13] ...
  - NDPs: Tesseract [ISCA’15]

Expensive communication across NDP units
Lack of a shared level of cache memory
NDP Synchronization Solution Space

(1) Shared Memory
- Hardware Cache Coherence
- Remote Atomics
- Specialized Hardware Support

(2) Message-passing
- Software-based Schemes
- Specialized Hardware Support

Prior schemes are not suitable or efficient for NDP systems
NDP Synchronization Solution Space

(1) Shared Memory
- Hardware Cache Coherence
- Remote Atomics
- Specialized Hardware Support

(2) Message-passing
- Software-based Schemes
- Specialized Hardware Support

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

NDPs:
- SynCron [HPCA’21]
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

- No Complex Cache Coherence Protocols
- No Expensive Atomic Operations
- Low Hardware Cost
2. Direct Buffering of Variables

<table>
<thead>
<tr>
<th>Address</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>...</td>
</tr>
<tr>
<td>--</td>
<td>...</td>
</tr>
<tr>
<td>--</td>
<td>...</td>
</tr>
<tr>
<td>--</td>
<td>...</td>
</tr>
</tbody>
</table>

Local lock acquire
2. Direct Buffering of Variables

- No Costly Memory Accesses
- Low Latency
3. Hierarchical Communication

NDP Unit 0

- NDP Core 0
- NDP Core 1
- Synchronization Engine 0
- Main Memory

NDP Unit 1

- NDP Core 0
- NDP Core 1
- Synchronization Engine 1
- Main Memory

- syncronVar

NDP Unit 2

- NDP Core 0
- NDP Core 1
- Synchronization Engine 2
- Main Memory

NDP Unit 3

- NDP Core 0
- NDP Core 1
- Synchronization Engine 3
- Main Memory
3. Hierarchical Communication

NDP Unit 0

NDP Core 0

NDP Core 1

Synchronization Engine 0

Main Memory

NDP Unit 1

NDP Core 0

NDP Core 1

Synchronization Engine 1

 syncronVar

Main Memory

NDP Unit 2

NDP Core 0

NDP Core 1

Synchronization Engine 2

Main Memory

NDP Unit 3

NDP Core 0

NDP Core 1

Synchronization Engine 3

Main Memory

Local lock acquire

Master
3. Hierarchical Communication
3. Hierarchical Communication

- Minimize Expensive Traffic
4. Integrated Overflow Management

NDP Unit 0

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 0

NDP Unit 1

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 1
- SyncronVar

Synchronization Processing Unit

- Synchronization Table
- Indexing Counters

Synchronization Table:

<table>
<thead>
<tr>
<th>Address</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>...</td>
</tr>
<tr>
<td>0x2241</td>
<td>...</td>
</tr>
<tr>
<td>0x438C</td>
<td>...</td>
</tr>
<tr>
<td>0x6B4A</td>
<td>...</td>
</tr>
</tbody>
</table>

Counter Values:

- Counter0 = 0
- Counter1 > 0
- Counter2 = 0
- Counter3 = 0

Fully Occupied

Address: 0x33A9
4. Integrated Overflow Management

- Low Performance Degradation
- High Programming Ease
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()
All NDP cores compete for the same lock variable.
Lock Operation

NDP Unit 0

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 0

Synchro-nization Process-ing Unit

Indexing Counters

Synchronization Table 0

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>11</td>
</tr>
</tbody>
</table>

NDP Unit 1

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 1

Synchronization Table 1

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>11</td>
</tr>
</tbody>
</table>

Local lock acquire

Master
Lock Operation

NDP Unit 0
- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 0

Synchro-
Process-
Table 0
Address | Global Waitlist | Local Waitlist | ...
------- | ------------- | ------------- | 
0x33A9  | 00            | 11            | ...

Indexing Counters

NDP Unit 1
- NDP Core 0
- NDP Core 1
- Main Memory
- SyncronVar

Synchronization Engine 1

Master

Synchro-
Process-
Table 1
Address | Global Waitlist | Local Waitlist | ...
------- | ------------- | ------------- | 
0x33A9  | 01            | 11            | ...

Indexing Counters
Lock Operation

NDP Unit 0

- NDP Core 0
- NDP Core 1
- Main Memory

- Synchronization Engine 0

- Synchro-nization Process- ing Unit
- Indexing Counters

- Synchronization Table 0
  - Address: 0x33A9
  - Global Waitlist: 00
  - Local Waitlist: 11

NDP Unit 1

- NDP Core 0
- NDP Core 1
- Main Memory

- Synchronization Engine 1

- syncronVar

- Master

- Synchro-nization Process- ing Unit
- Indexing Counters

- Synchronization Table 1
  - Address: 0x33A9
  - Global Waitlist: 01
  - Local Waitlist: 11
Lock Operation

**NDP Unit 0**
- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 0

**Synchronization Processing Unit**
- Indexing Counters

---

**Synchronization Table 0**
<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>11</td>
</tr>
</tbody>
</table>

---

**NDP Unit 1**
- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 1

**Synchronization Processing Unit**
- Indexing Counters

---

**Synchronization Table 1**
<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>01</td>
<td>00</td>
</tr>
</tbody>
</table>

---

*Global lock grant*
Lock Operation

NDP Unit 0

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 0

Synchronization Processing Unit

Indexing Counters

Synchronization Table 0

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>11</td>
</tr>
</tbody>
</table>

NDP Unit 1

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 1
- syncronVar

Synchronization Processing Unit

Indexing Counters

Synchronization Table 1

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>01</td>
<td>00</td>
</tr>
</tbody>
</table>

Master

Local lock grant
Lock Operation

NDP Unit 0

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 0

Synchronization Processing Unit

Indexing Counters

Synchronization Table 0

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>00</td>
</tr>
</tbody>
</table>

NDP Unit 1

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 1

syncronVar

Master

Synchronization Processing Unit

Indexing Counters

Synchronization Table 1

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>01</td>
<td>00</td>
</tr>
</tbody>
</table>
Lock Operation

NPD Unit 0
- NDP Core 0
- NDP Core 1
- Main Memory
  - Synchronization Engine 0

NPD Unit 1
- NDP Core 0
- NDP Core 1
- Main Memory
  - Synchronization Engine 1

Synchronization Table 0
- Address
- Global Waitlist
- Local Waitlist
- ... (empty)

Synchronization Table 1
- Address: 0x33A9
- Global Waitlist: 01
- Local Waitlist: 00
- ... (empty)
Lock Operation

NDP Unit 0

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 0

Synchro-

nization Process-

ing Unit

Indexing Counters

Synchronization Table 0

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>--</td>
<td>--</td>
<td>--</td>
<td></td>
</tr>
</tbody>
</table>

NDP Unit 1

NDP Core 0

NDP Core 1

Main Memory

Synchronization Engine 1

syncronVar

Synchro-

nization Process-

ing Unit

Indexing Counters

Synchronization Table 1

<table>
<thead>
<tr>
<th>Address</th>
<th>Global Waitlist</th>
<th>Local Waitlist</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x33A9</td>
<td>00</td>
<td>00</td>
<td></td>
</tr>
</tbody>
</table>

Global lock release

Master
Lock Operation

NDP Unit 0

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 0

Synchro-

Synchronization Processing Unit

Indexing Counters

Synchronization Table 0

| Address | Global Waitlist | Local Waitlist | ...
|---------|-----------------|----------------|----
| --      | --              | --             | ...

NDP Unit 1

- NDP Core 0
- NDP Core 1
- Main Memory
- Synchronization Engine 1
- syncronVar

Master

Synchronization Processing Unit

Indexing Counters

Synchronization Table 1

| Address | Global Waitlist | Local Waitlist | ...
|---------|-----------------|----------------|----
| --      | --              | --             | ...

36
Lock Operation

More details in the paper
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

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

- **Stack – 100K**:
  - Central
  - Hier
  - SynCron
  - Ideal

- **Hash Table – 1K**:
  - Central
  - Hier
  - SynCron
  - Ideal

- **Linked List – 20K**:
  - Central
  - Hier
  - SynCron
  - Ideal

**Number of NDP Cores**

- **Operations / μs**
  - Stack – 100K
  - Hash Table – 1K
  - Linked List – 20K

- **Throughput of Pointer Chasing**
  - High Contention
  - Medium Contention
  - Low Contention

- **Small # of Variables**
- **Medium # of Variables**
- **High # of Variables**
SynCron achieves the highest throughput under all scenarios.
SynCron performs best across all real applications
System Energy in Real Applications

SynCron reduces system energy significantly
# Area and Power Overheads

<table>
<thead>
<tr>
<th></th>
<th>Synchronization Engine</th>
<th>ARM Cortex A7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>40nm</td>
<td>28nm</td>
</tr>
<tr>
<td>Area</td>
<td>9.78%</td>
<td>Total: 0.45mm²</td>
</tr>
<tr>
<td></td>
<td>Total: 0.0461mm²</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>2.70%</td>
<td>100mW</td>
</tr>
<tr>
<td></td>
<td>2.7mW</td>
<td></td>
</tr>
</tbody>
</table>

SynCron has low area and power overheads.
Sensitivity Studies

• Different memory technologies (HBM, HMC, DDR4)
• 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

Christina Giannoula
christina.giann@gmail.com
Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas
Ivan Fernandez, Juan Gómez Luna, Lois Orosa
Nectarios Koziris, Georgios Goumas, Onur Mutlu

SAFARI
ETH zürich
UNIVERSITY OF TORONTO
UNIVERSIDAD DE MÁLAGA