Pair & Swap : An Approach to Graceful Degradation for Dependable Chip Multiprocessors

Masashi Imai (miyabi@hal.rcast.u-tokyo.ac.jp)
Tomohide Nagai (nagai@hal.rcast.u-tokyo.ac.jp)
Takashi Nanya (takashi.nanya@canon.co.jp)

Agenda

- Introduction
  - Related works
- Pair & Swap
  - Concept
  - Hardware model
  - Execution steps
  - Comparison mechanism
  - Task management mechanism
- Evaluation
- Conclusion

Background & Motivation

- VLSI technology scaling
  - → The performance improvement of a single processor is limited due to clock skew, power dissipation, ILP, and complexity
- CMP (Chip Multi-Processor)
  - Integrates multiple processor cores in a single chip
  - CMP is a promising VLSI architecture, not only for high performance but also for reducing power dissipation
  - Even if a processor core becomes faulty, the remaining cores can continue to operate
  - → It is not efficient to replace the entire CMP chip immediately when a permanent fault occurs

Background & Motivation

- We consider CMP systems as non-repairable systems and present an approach to graceful degradation for dependable CMP

- Dual module redundancy (DMR)
  - Can detect faults by comparing the result of tasks
  - The number of tasks in *N*-cores CMP : *N*/2
- Triple module redundancy (TMR)
  - Can mask faults
  - Can identify a failure core
  - The number of tasks in *N*-cores CMP : *N*/3

→ Pair-based scheme for dependable CMP in order to achieve high-performance

Related works

- Single-processor SMT devices
  - RMT (Redundant MultiThreading) [Nirmal98]
  - AR-SMT (Active-stream/Redundant-stream Simultaneous MultiThreading) [Eric99]
    - A tme redundancy techniques which compares the results of a leading thread called A-thread with the results of a trailing thread called R-thread
  - SRT (Simultaneous and Redundantly Threaded) [Reinhardt00]
    - Executes two identical copies of the same program as independent threads and compares their results
  - SRTR (SRT with Recovery) [Vijaykumar02]

Related works

- Dual-processor devices which indicate both a dual-core CMP chip and different dies
  - Lockstep techniques [Nicholas93, Timothy99, Reorda09]
    - Assumes that an error in either processor will cause a difference between the states of the two processors
  - Watchdog processors [Mahmod88]
  - DIVA (Dynamic Implementation Verification Architecture) [Austin99]
    - Employs a high-performance processor core as a leading core and a low-performance core as a trailing checker core

Related works

- CMP devices
  - CRT (Chip-level Redundant Threading) [Mukherjee02]
    - Applies SRT's detection techniques to CMPs
  - CRTR (CRT with Recovery) [Mohamed03]
    - Extends the CRT for transient-fault detection
  - DCR (Dual Core Redundancy) [Gong08]
    - Extends the CRT by adding HW implemented context saving and recovery
  - TCR (Triple Core Redundancy) [Gong08]
    - Extends three copies of a given program on a leading thread, a middle thread, and a trailing thread
  - DCC (Dynamic Core Coupling) [Christopher07]
    - Allows arbitrary CMP cores to verify each other's execution while requiring no dedicated cross-core communication channels or buffers
    - The basic concept of our method is similar to DCC, while DCC employs a TMR using hot spares in order to isolate a failure core and recovery its task

Agenda

- Introduction
  - Related works
- Pair & Swap
  - Concept
  - Hardware model
  - Execution steps
  - Comparison mechanism
  - Task management mechanism
- Evaluation
- Conclusion

Fault model

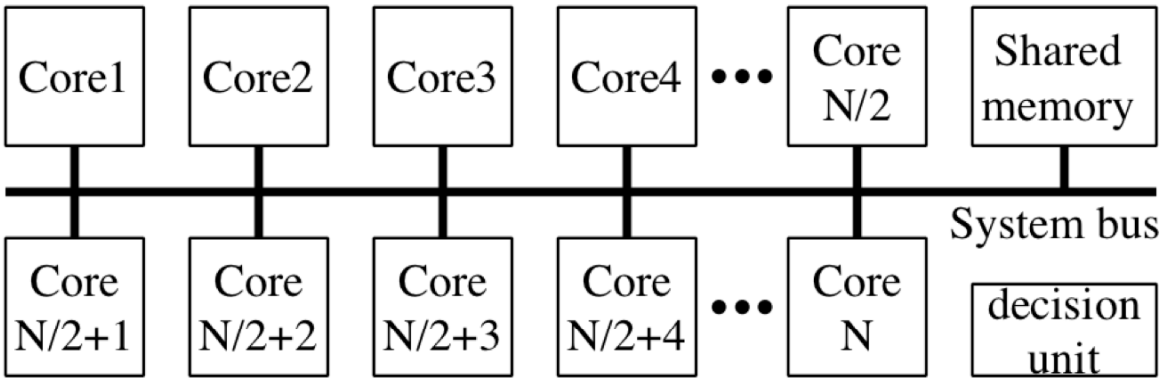
- Single-core fault
  - A fault can occur only in a single core at a time
- Permanent fault
  - We must identify the failure core and stop using it
- Transient fault
  - The core in which a transient fault occurs can be recovered by re-executing from the latest checkpoint
  - → We do not have to stop using it immediately
  - Generally, transient faults tend to occur much more frequently than permanent faults

Pair & Swap

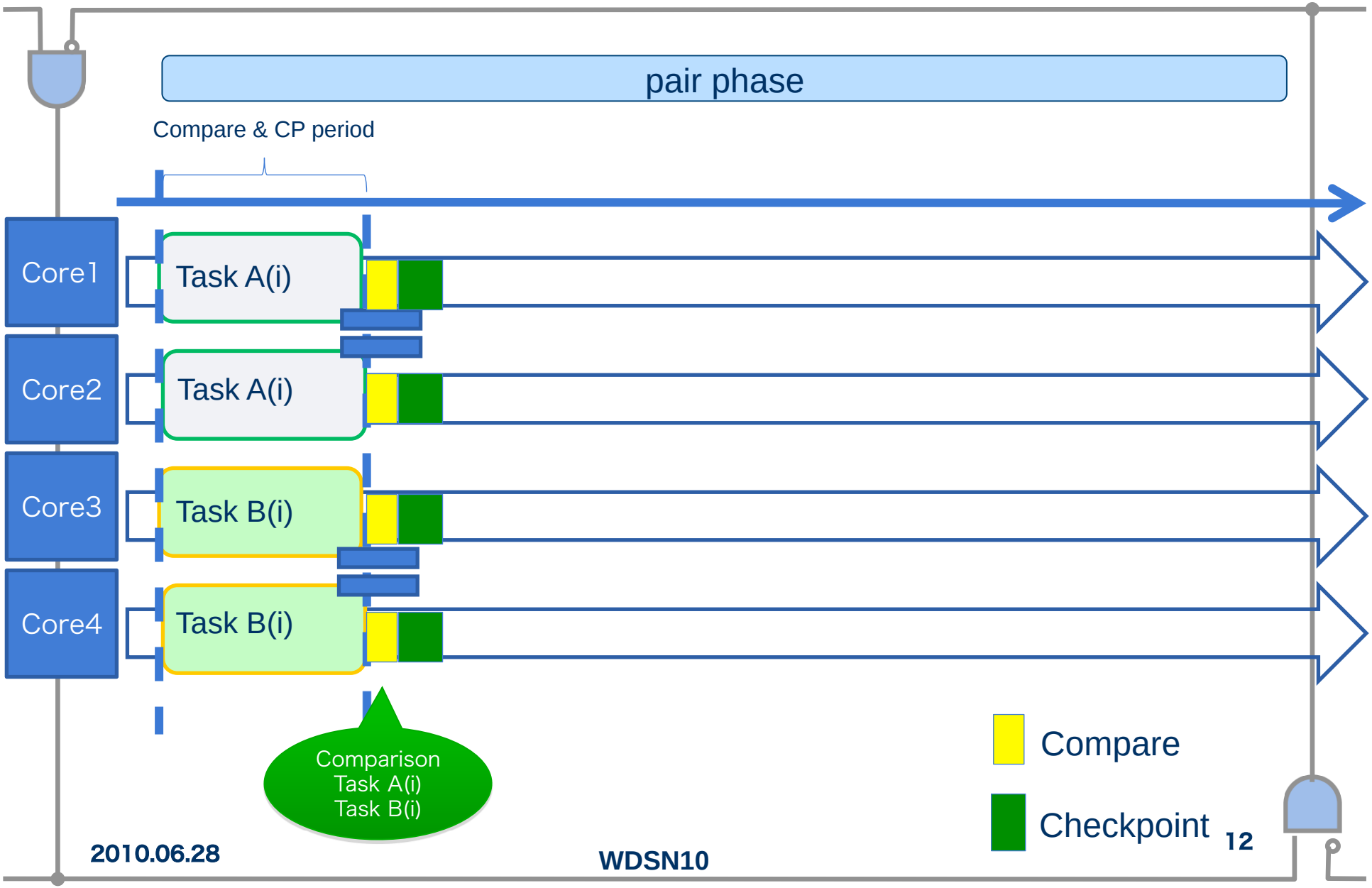
- Processor-level fault tolerance technique for CMPs which consists of two phases
- *Pair phase* : replication and comparison
  - Two identical copies of a given task are executed on a pair of two processor cores and the results are compared
  - If no fault is detected, each core repeats a period of execution and comparison
- *Swap phase* : swap and retry
  - Partners of the mismatched pair are swapped with another pair and mismatched task is re-executed from the latest checkpoint
  - It is decided whether the fault is transient or permanent in the end of the swap phase
    - Permanent fault: the failure core is identified and isolated to reconfigure the entire CMP system for continuous operation in a degraded mode
    - Transient fault: the swapped pairs continue their tasks without any reconfiguration in the next pair phase

Target model

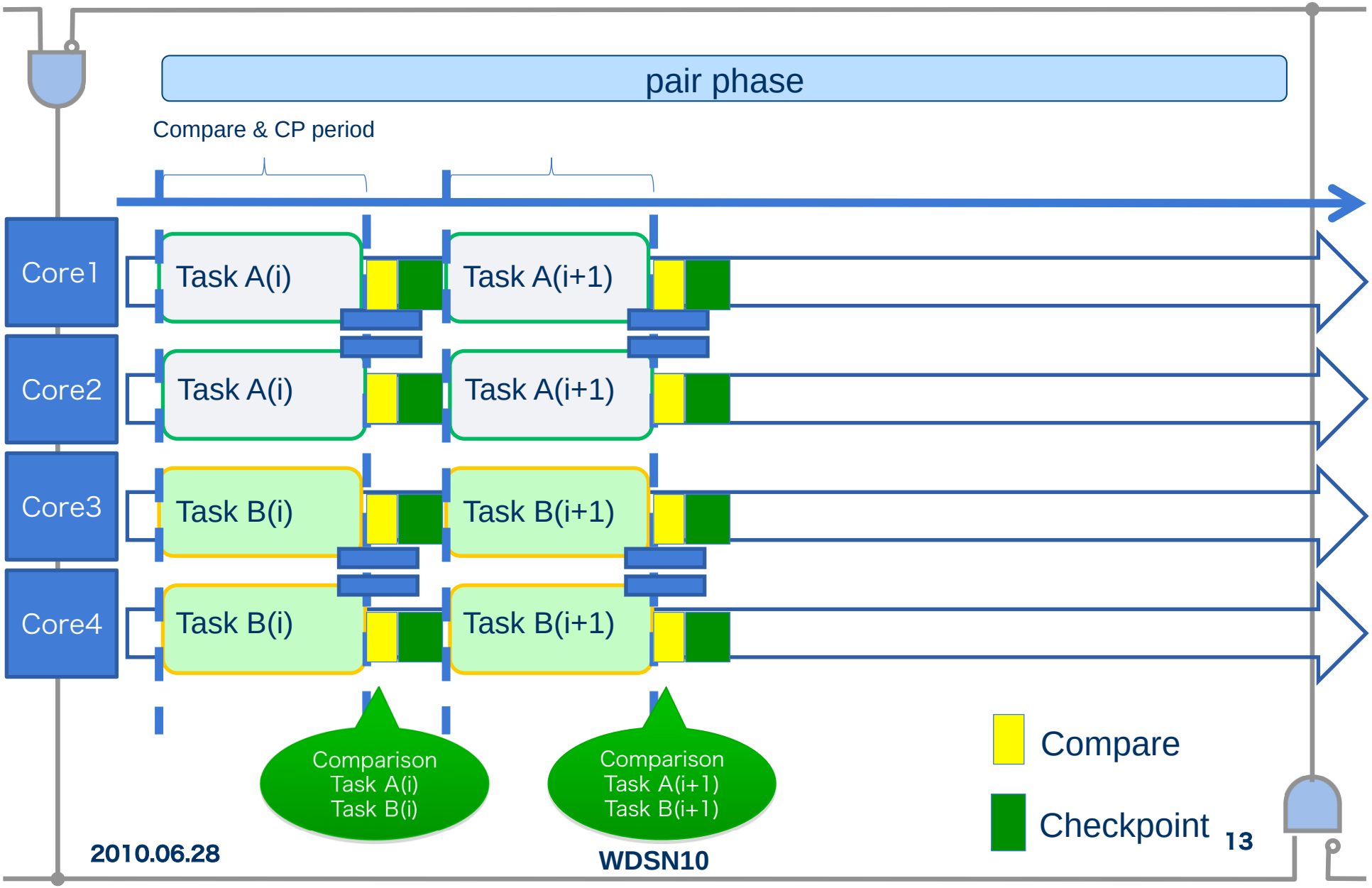
1. More than four cores in order to swap partners
2. A stable storage in order to retry the mismatched task from the latest correct checkpoint
   - A shared memory is used as the stable storage and the correct checkpoint data is stored in the shared memory
3. A non-faulty decision unit which decides the comparison results of all the pairs in order to generate consistent comparison results
   - It is needed because a pair of two cores in which a fault may occur cannot generate a consistent comparison result by themselves



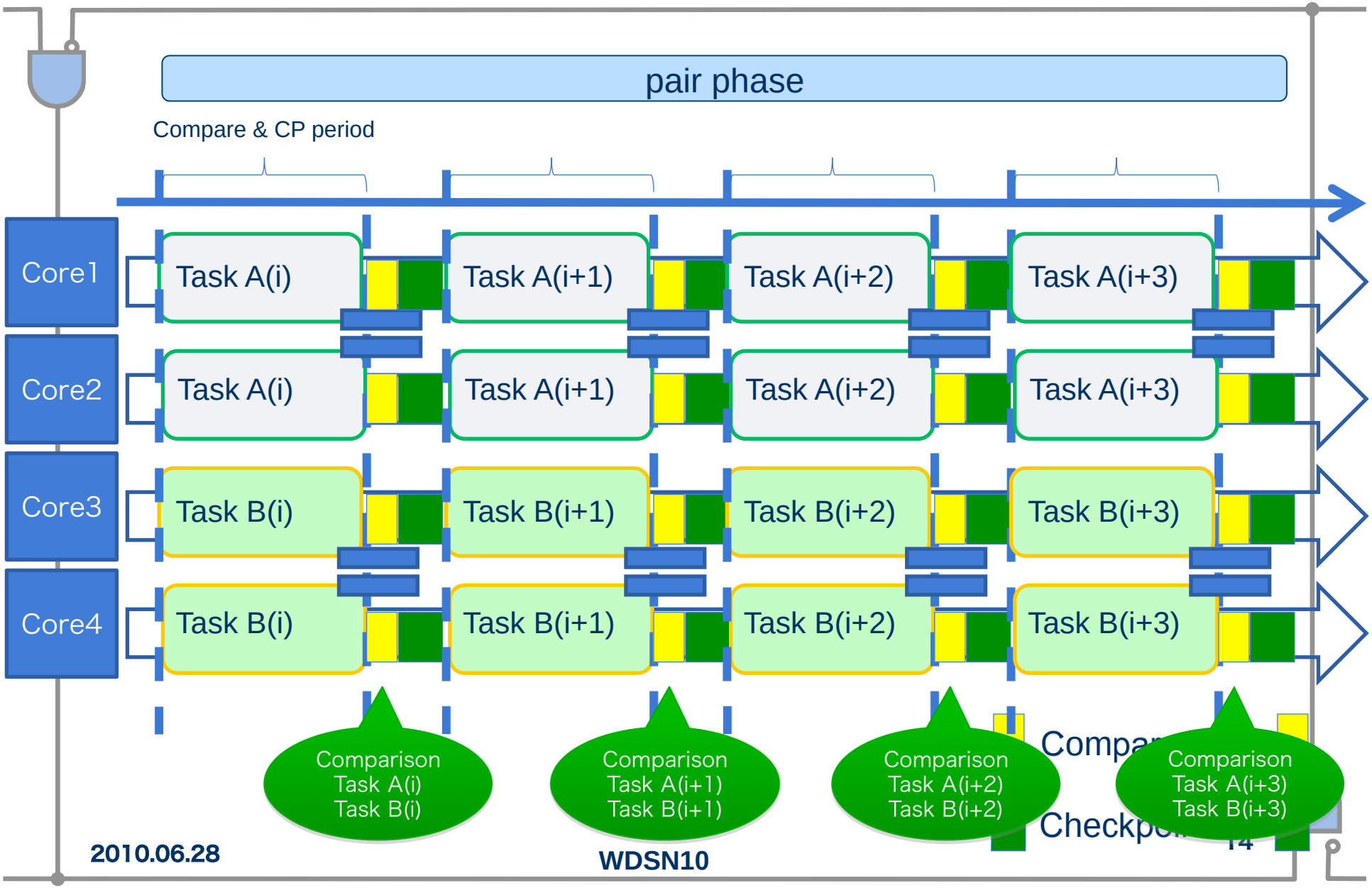
Pair & Swap: Pair phase



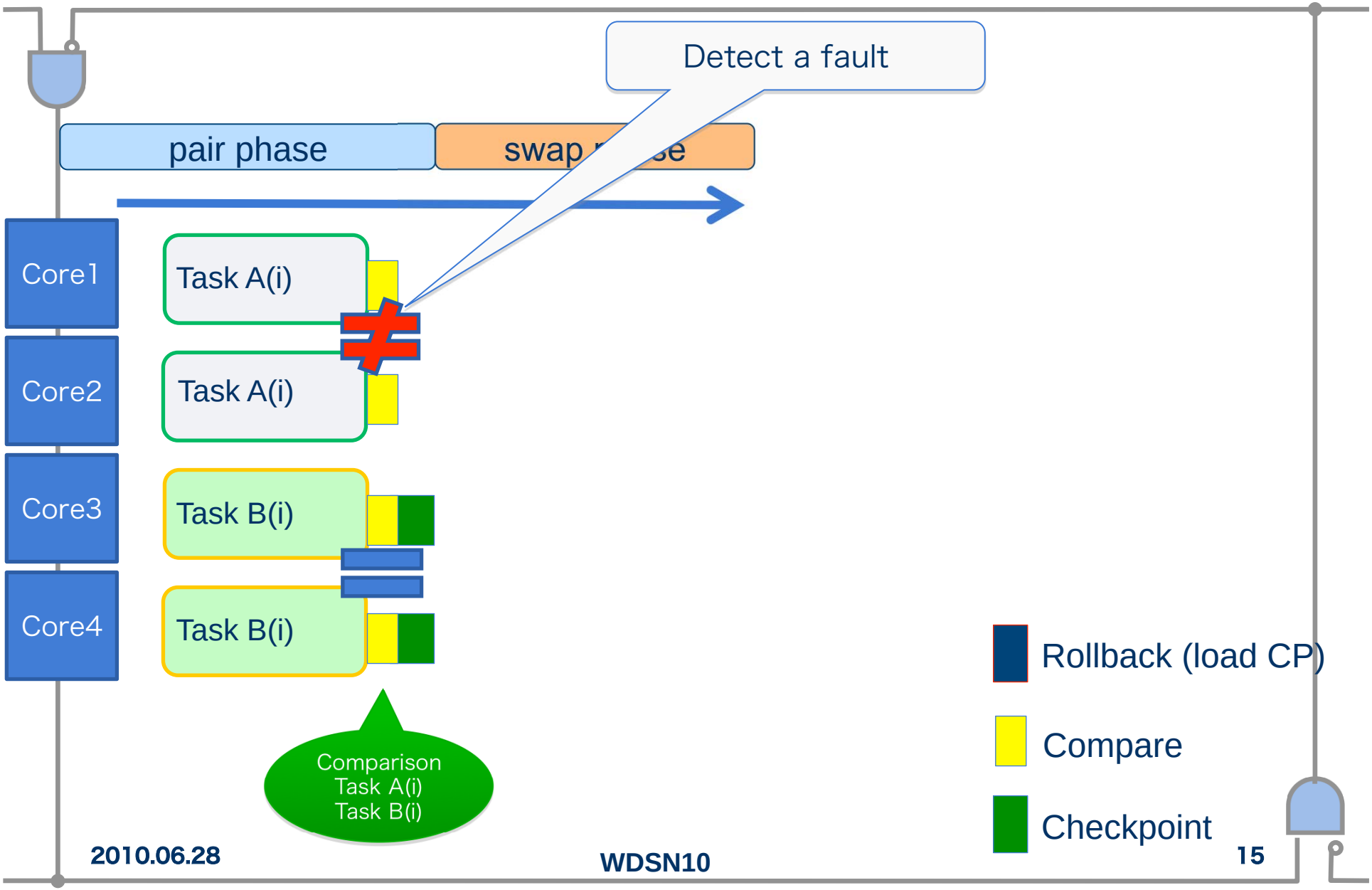
Pair & Swap: Pair phase



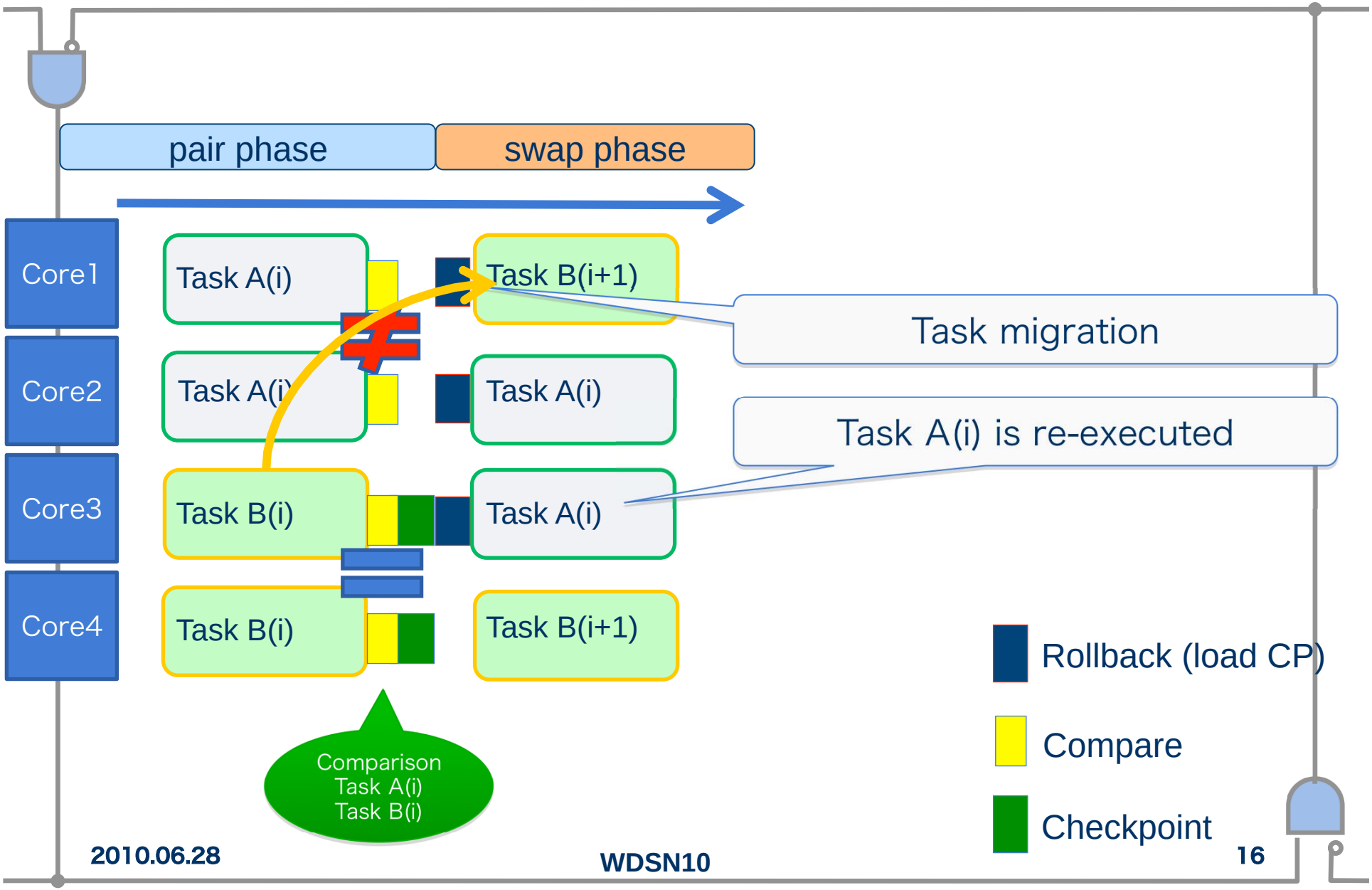
Pair & Swap: Pair phase



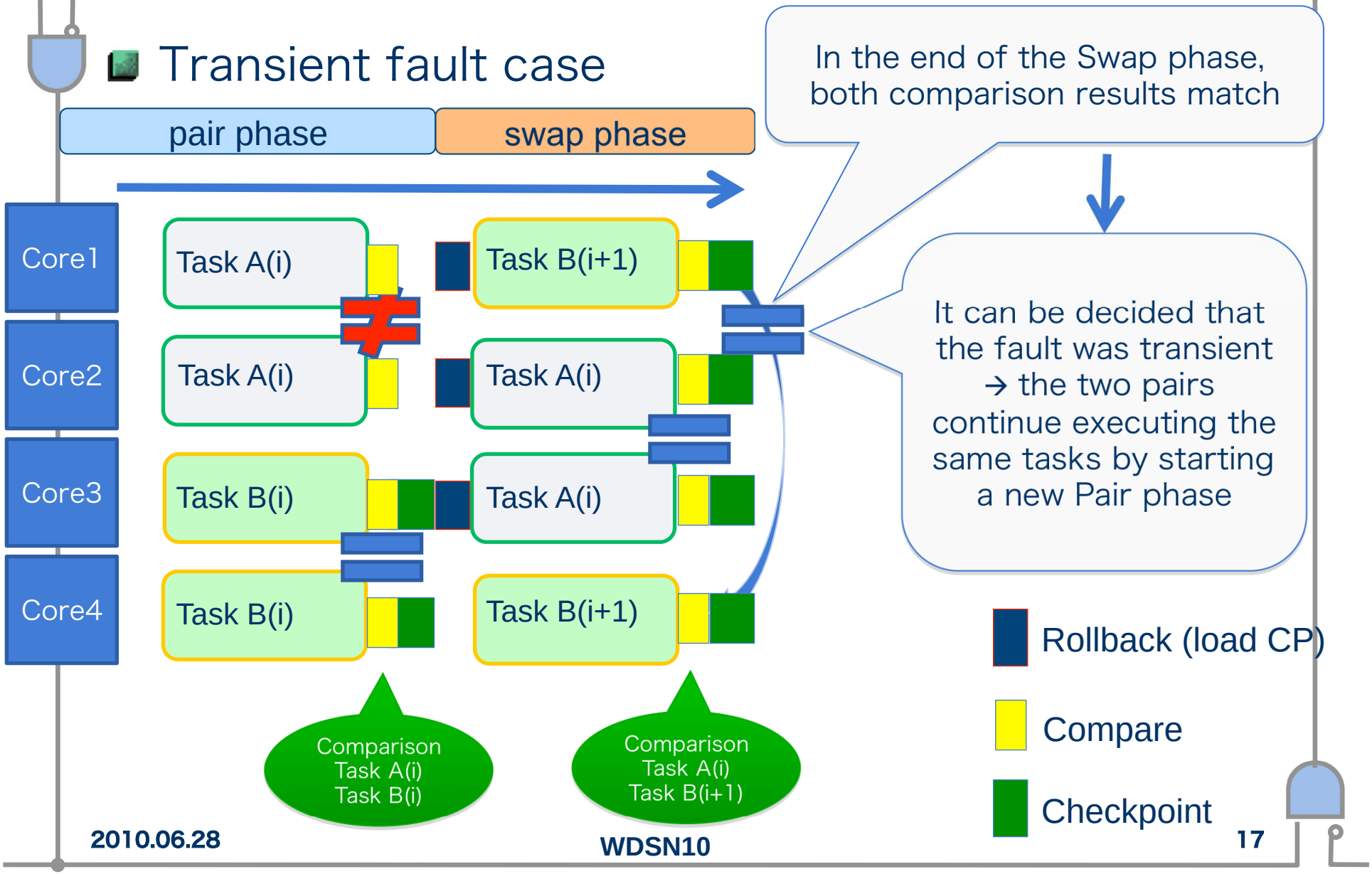
Pair & Swap: Swap phase



Pair & Swap: Swap phase

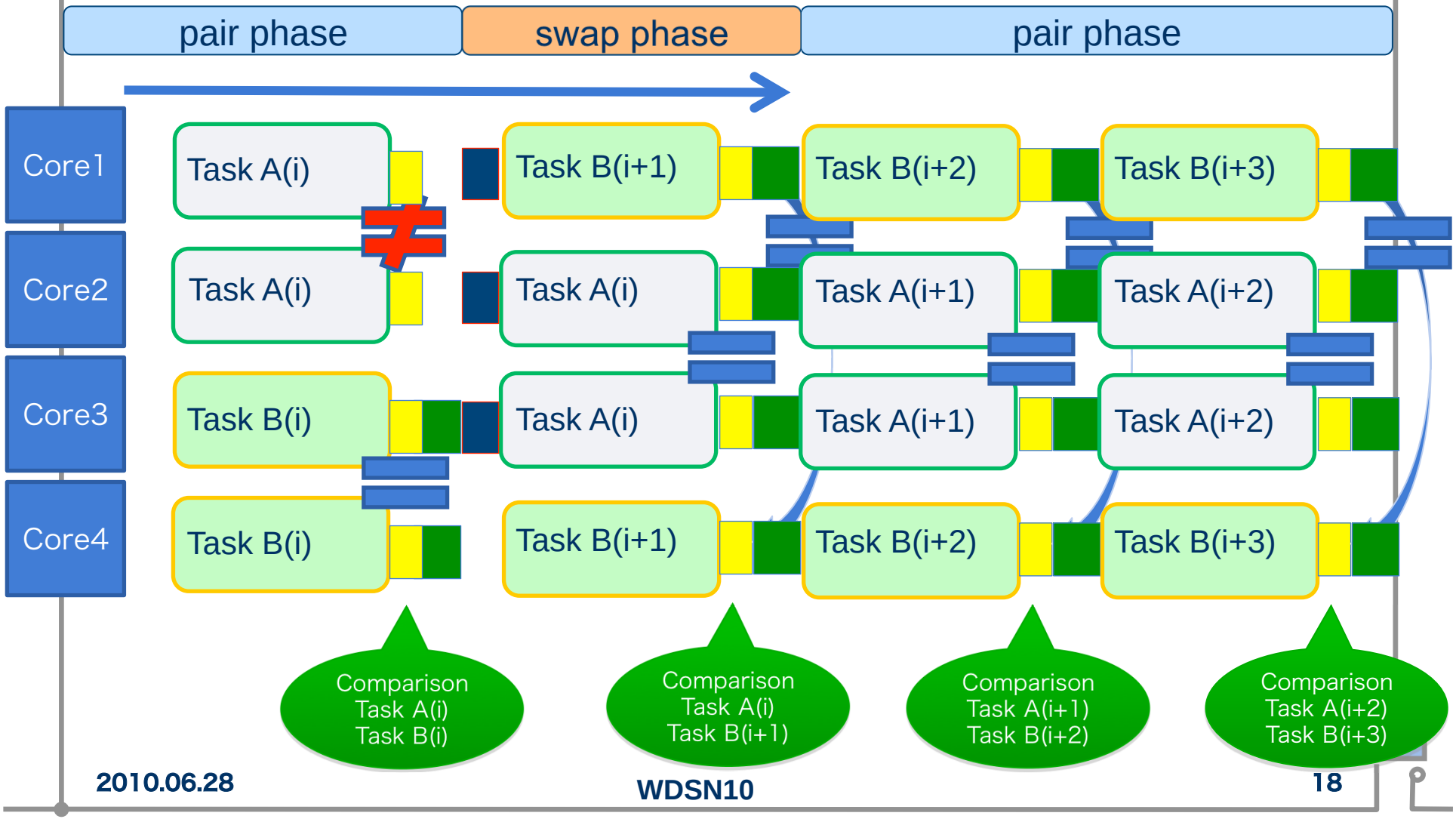


Pair & Swap: Fault location (1)



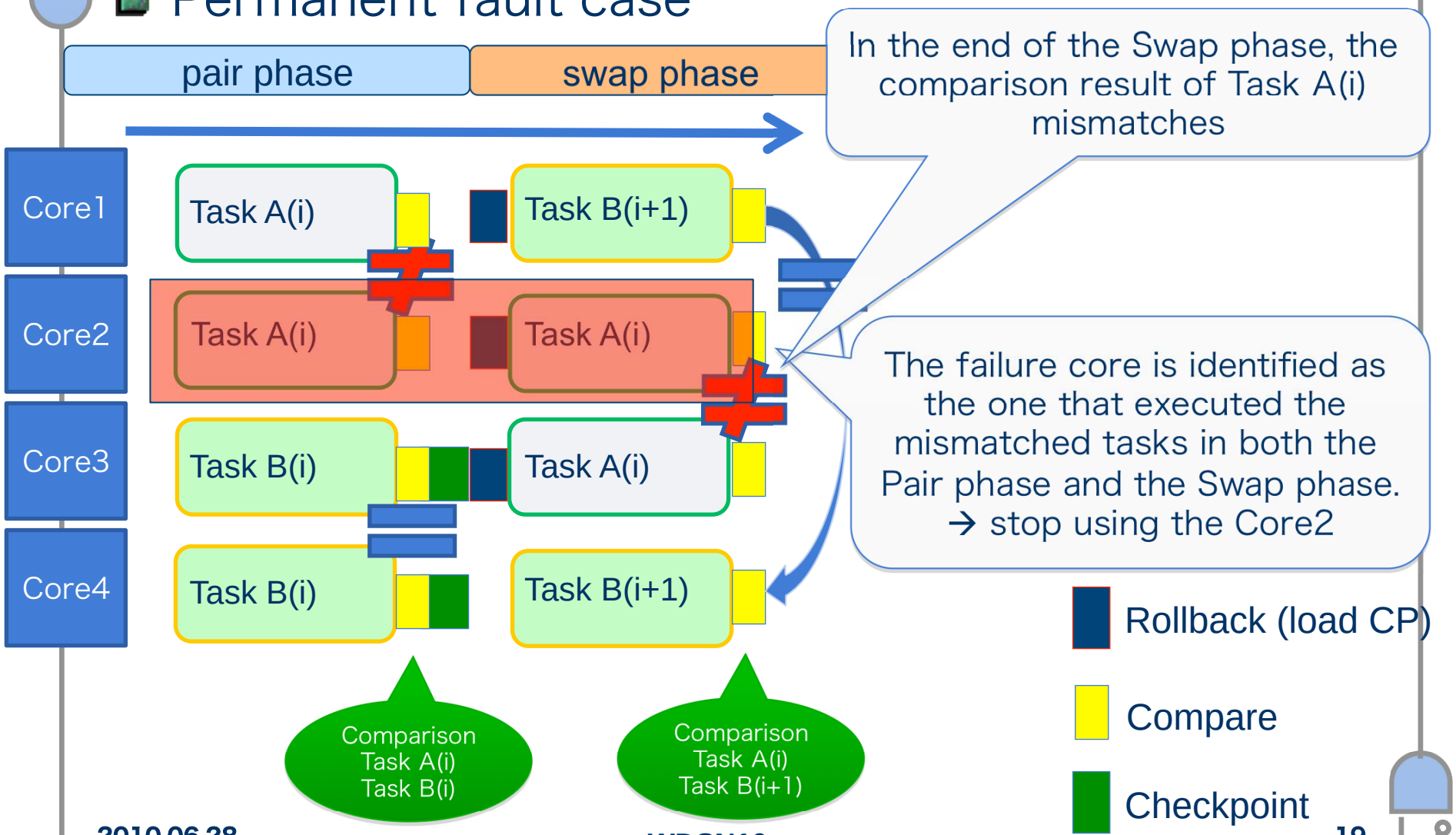
Pair & Swap: Fault location (1)

Transient fault case



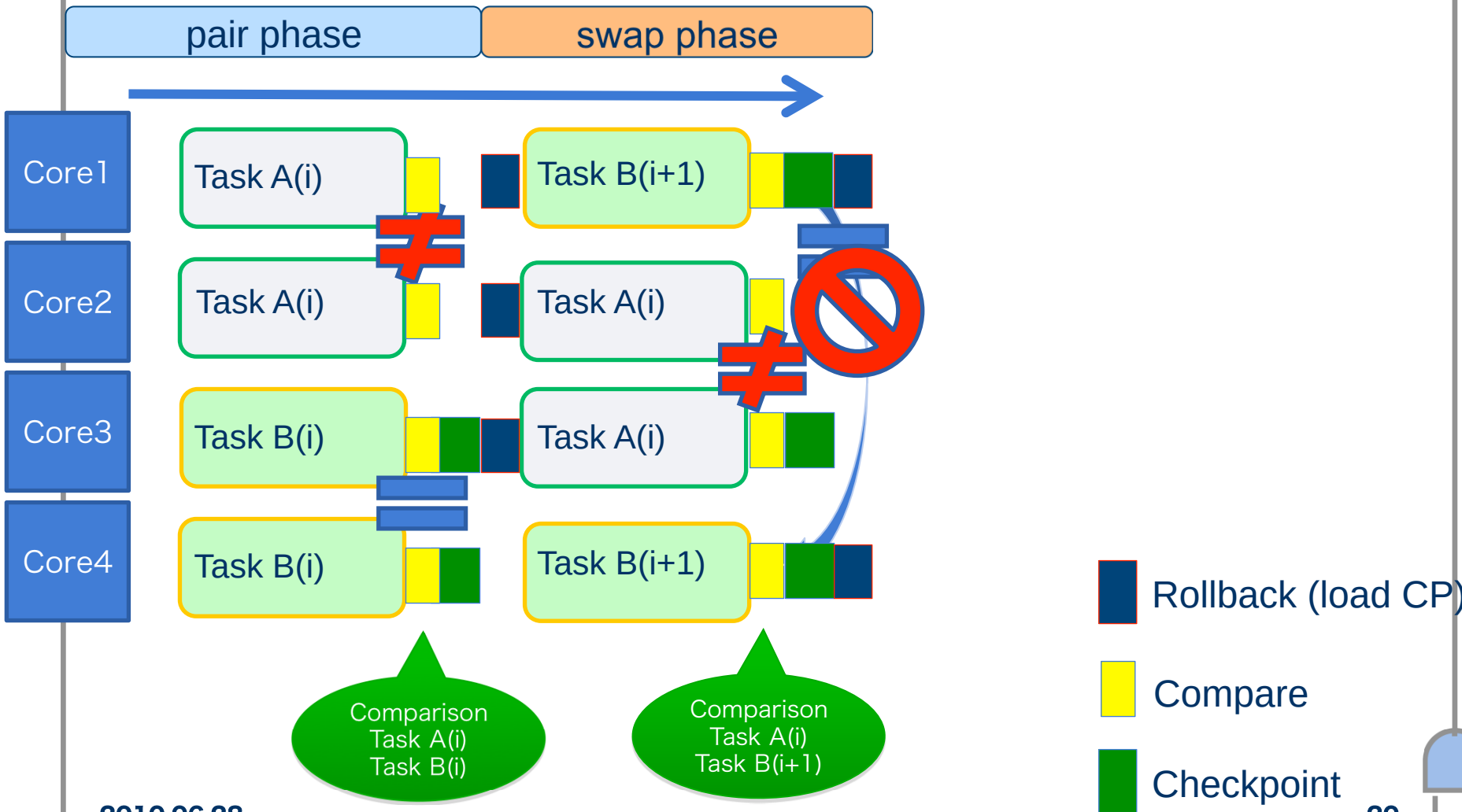
Pair & Swap: Fault location (2)

Permanent fault case



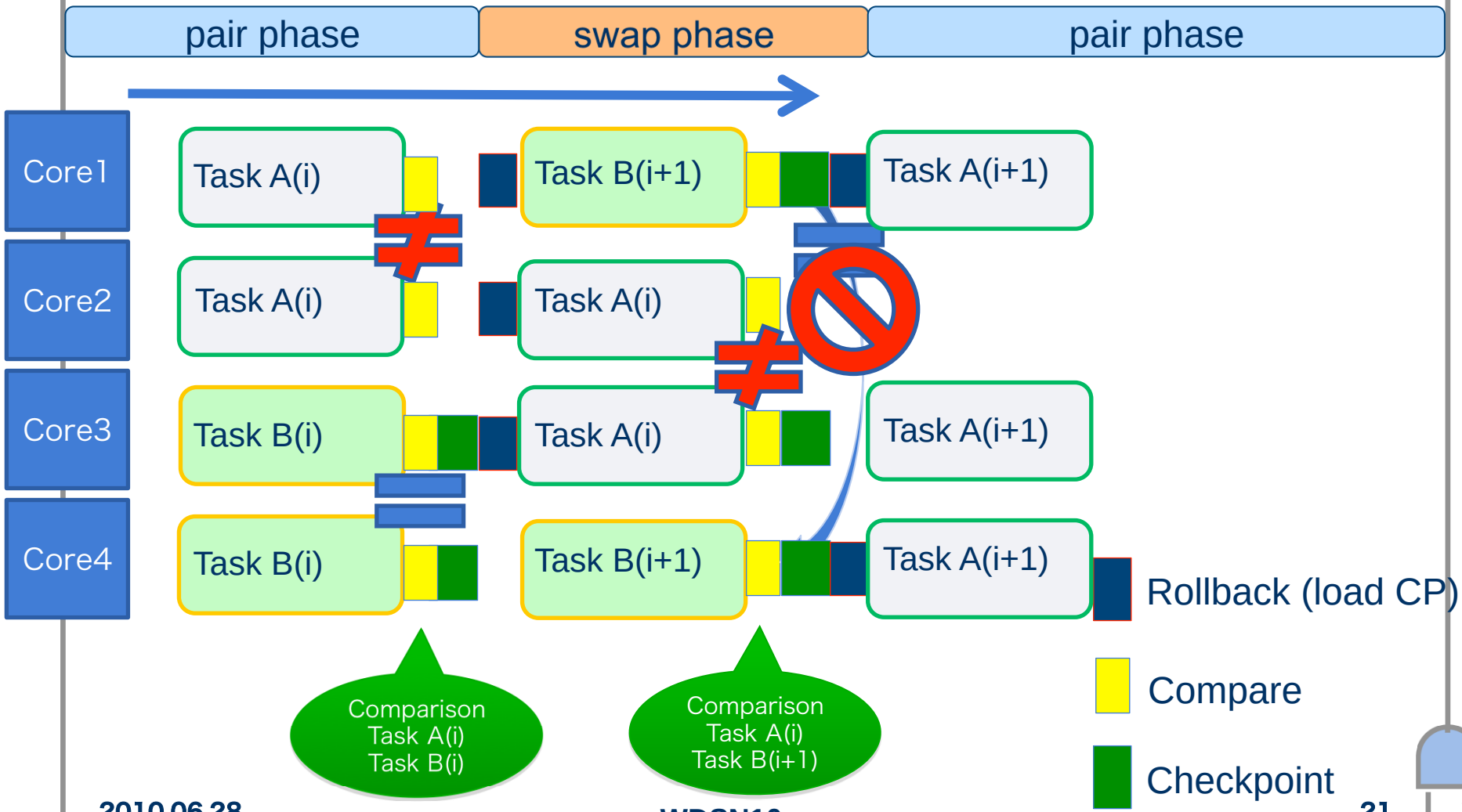
Pair & Swap: Fault location (2)

Permanent fault case



Pair & Swap: Fault location (2)

Permanent fault case

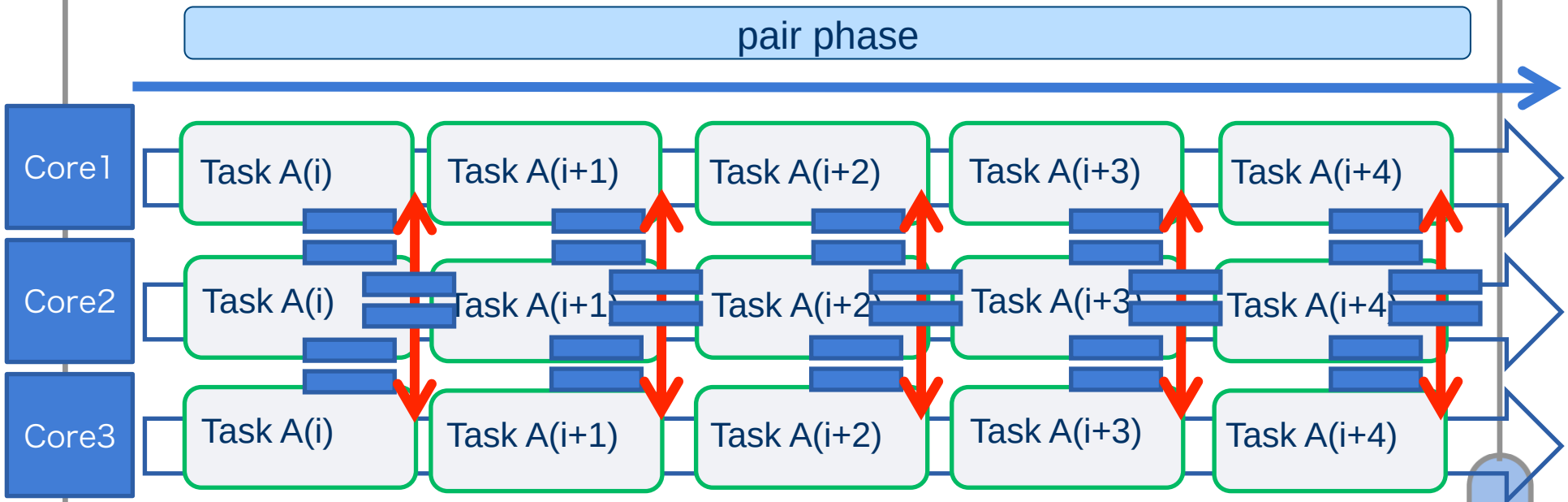


“Trio” configuration

- When a permanent fault occurs, the number of processor cores may become odd (*2m+1*)

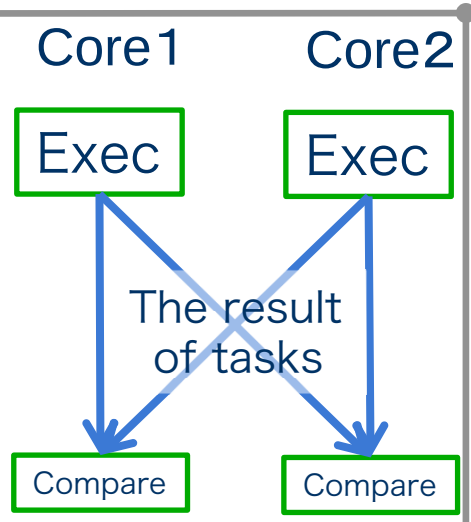
→ It can compose *m-1* pairs using *2(m-1)* cores and 3 processor cores remain

→ *3* cores execute the same task and compare their results each other: “Trio” configuration



Comparison mechanism

- What should be compared for replicated task execution depends on the application
  - all register file, status registers, and memory updates
  - the output value of the system may only be required
- Two processor cores in each pair exchange the compressed data over the system bus
  - MPI can be used
- Each core compares its data with partner's data each other



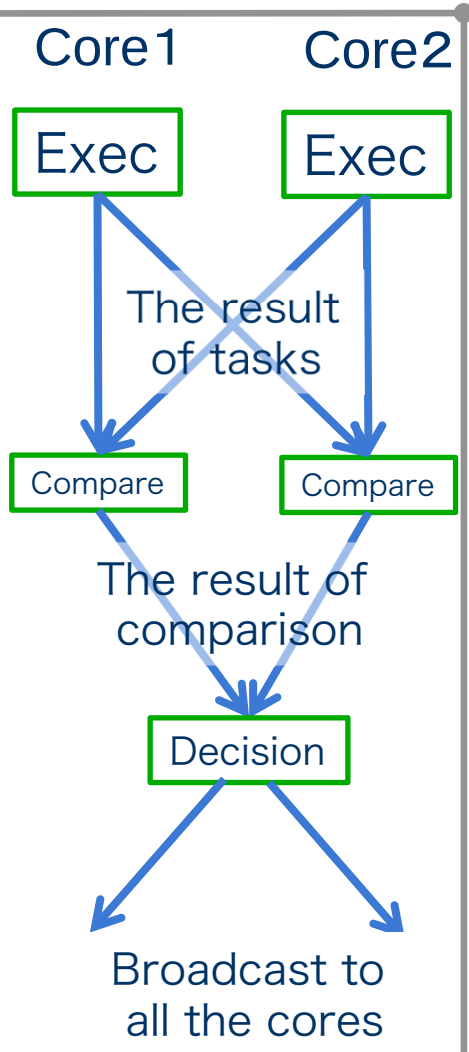
		Core1	Core2
No fault occurs	Task execution	No error	No error
	Comparison result	MATCH	MATCH
A fault occurs during task execution	Task execution	No error	Error
	Comparison result	MISMATCH	MATCH or MISMATCH
A fault occurs during comparison	Task execution	No error	No Error
	Comparison result	MATCH	MISMATCH

Comparison mechanism

- The decision unit
  - Gathers the comparison results from all the cores
  - Decides whether the results match or mismatch for all the pairs like the following table

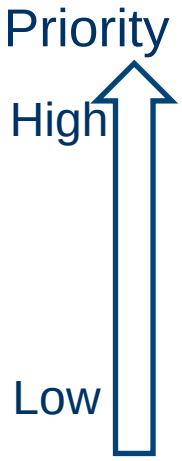
Core1 comparison result	Core2 comparison result	Decision unit output
Match	Match	Match
Match	Mismatch	Mismatch
Mismatch	Match	Mismatch
Mismatch	Mismatch	Mismatch

    - A Comparison result of each core can be represented by one bit.
  - Broadcasts its results to all the cores



Task assignment table

- Each processor core manages the core paring and the task assignment table
  - There is no special core which controls the entire system
- Tasks have priority and the list is ordered by the priority



Task	Assigned cores
A	0, 1
B	2, 3
C	4, 5
D	6, 7

- When a mismatched task is detected in the comparison results which are broadcasted by the decision unit in the Pair phase, each core updates the table for the following Swap phase
- The swapping pair is selected as follows;
  1. If there is a Trio in the table, the Trio is selected
  2. The pair which executes the lowest priority task except it own pair is selected
- Permanent fault → The lowest task in the table cannot be executed in the next Pair phase

Agenda

- Introduction
  - Related works
- Pair & Swap
  - Concept
  - Hardware model
  - Execution steps
  - Comparison mechanism
  - Task management mechanism
- Evaluation
- Conclusion

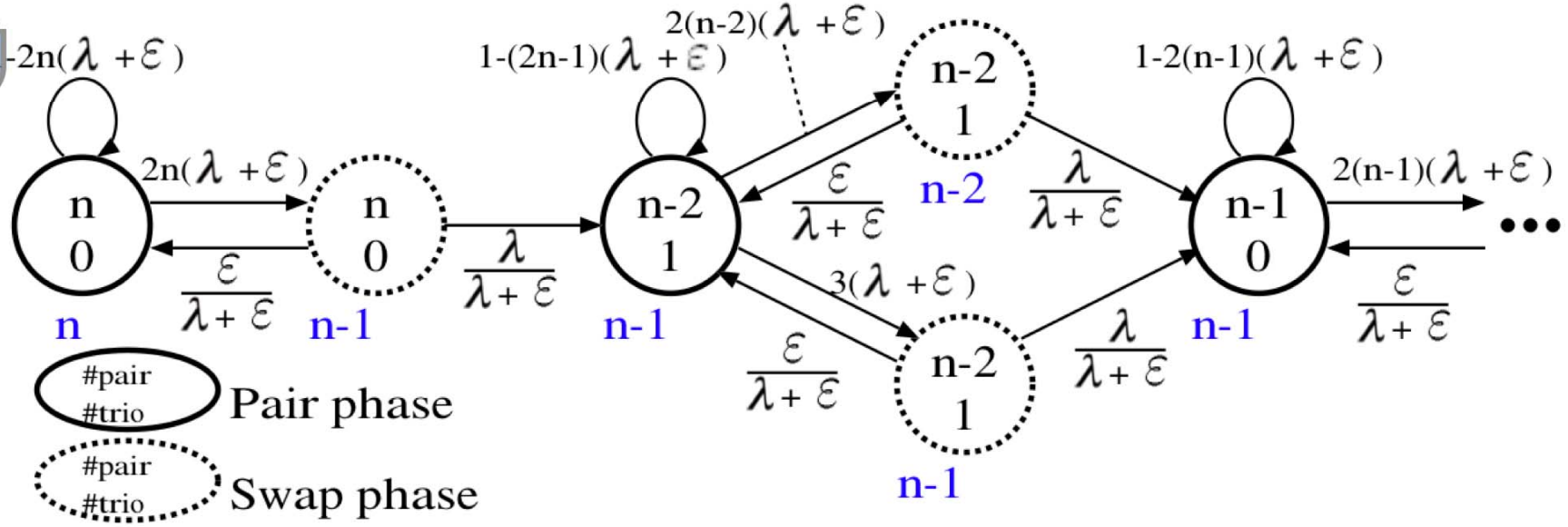
Evaluation

- Evaluate the expected value of the computation capability to failure called “MCTF (Mean Computation To Failure)” using the Markov chains in order to compare the performance

$$MCTF = \sum_{i \neq failure} (Performance(i) \times \int_0^\infty P_i(t)dt)$$

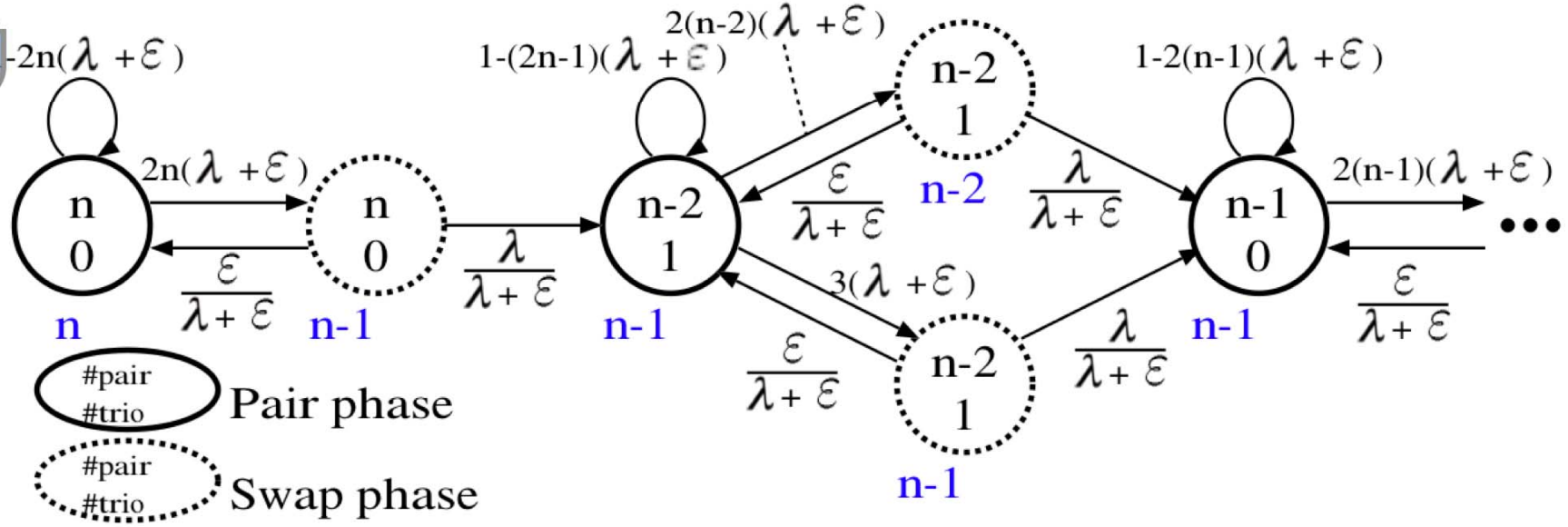
- Comparison targets
  1. Proposed Pair & Swap
  2. Dynamic TMR
  3. Static TMR
- Failure rate
  - Permanent : $\lambda = 1.0 \times 10^{-9}$
  - Transient : $\varepsilon = 1.0 \times 10^{-7}$
- Fault detection, fault location, and reconfiguration are successfully executed with a probability of 1

Markov chain of Pair & Swap



- The performance is defined as the mean number of tasks which can be executed at the state
  - If a fault is detected in any pairs, the mismatched task must be re-executed
  - → The mean number of tasks decreases in the Swap phase
  - If a fault is detected in a Trio, the task can be executed continuously
  - → The mean number of tasks does not change

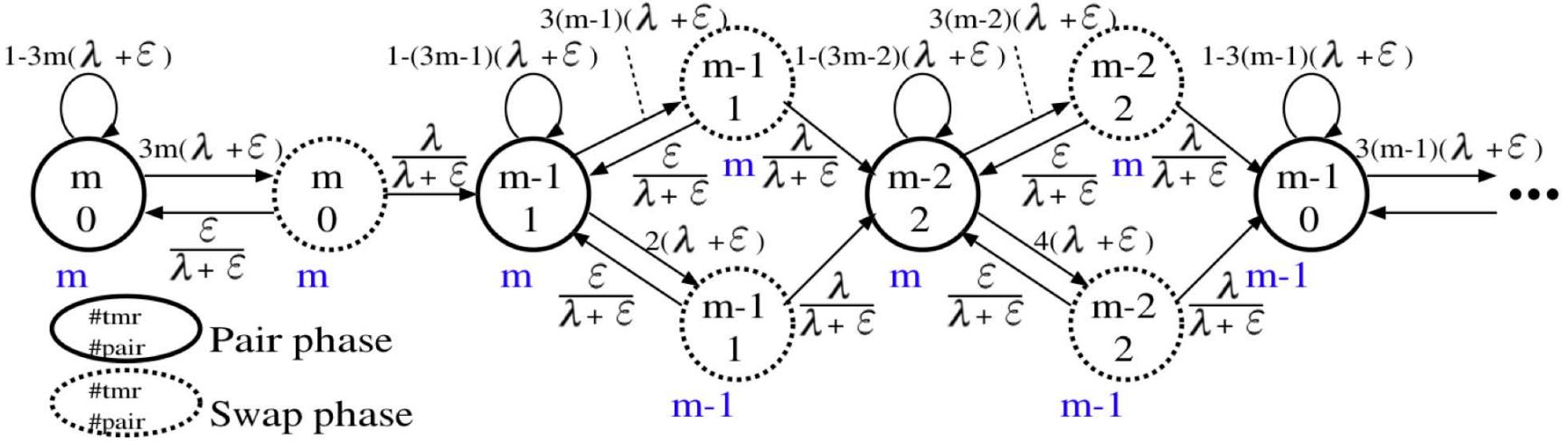
Markov chain of Pair & Swap



- The performance is defined as the mean number of tasks which can be executed at the state
  - If a fault is detected in any pairs, the mismatched task must be re-executed
  - → The mean number of tasks decreases in the Swap phase
  - If a fault is detected in a Trio, the task can be executed continuously
  - → The mean number of tasks does not change

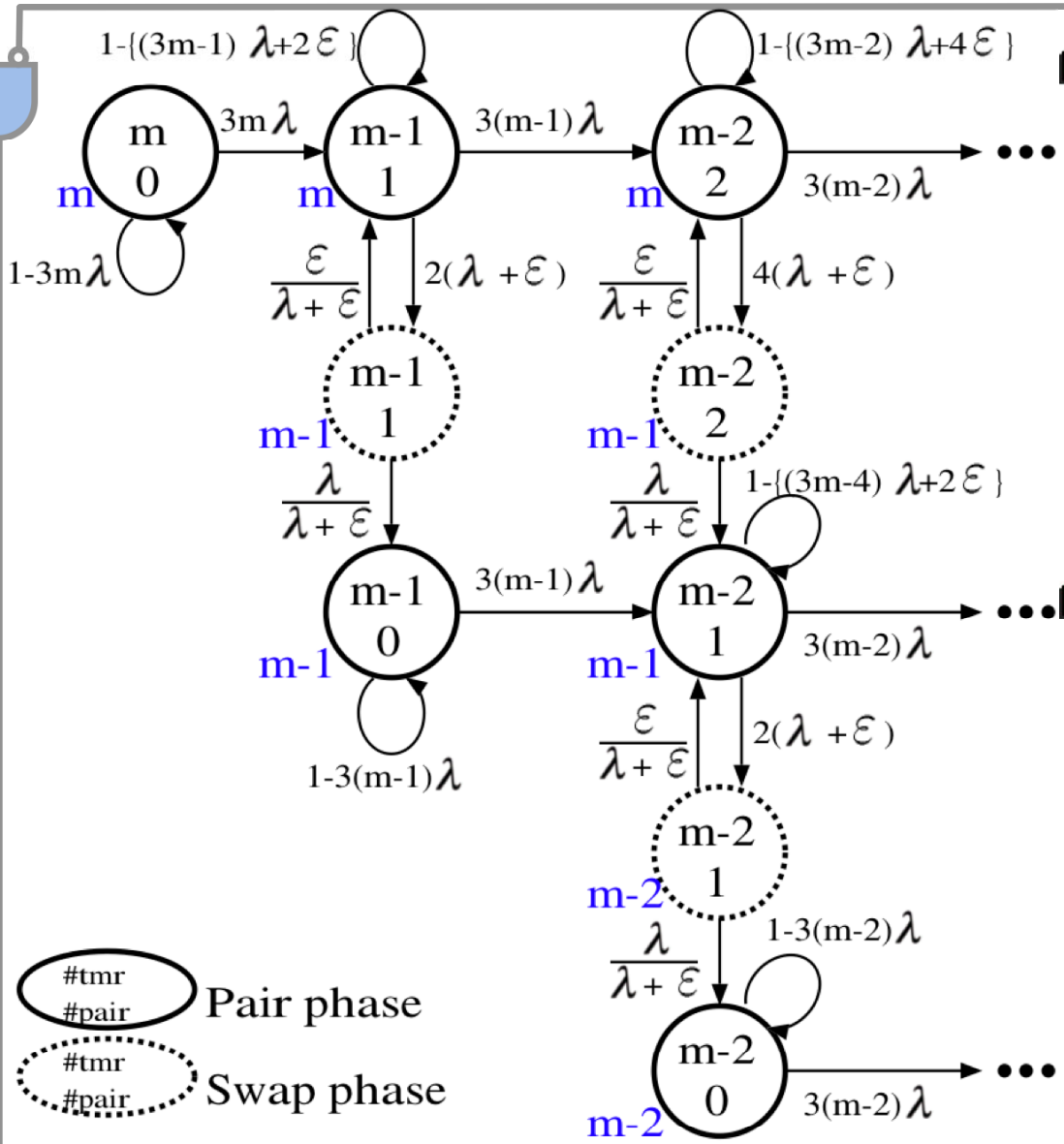
Markov chain of dynamic TMR

- Dynamic TMR in which three processor cores are dynamically coupled as the number of active cores decreases
  - When the number of active cores is 3m+2, the remaining 2 cores compose a pair
  - When the number of active cores is 3m+1, a TMR and the remaining 1 core compose 2 pairs



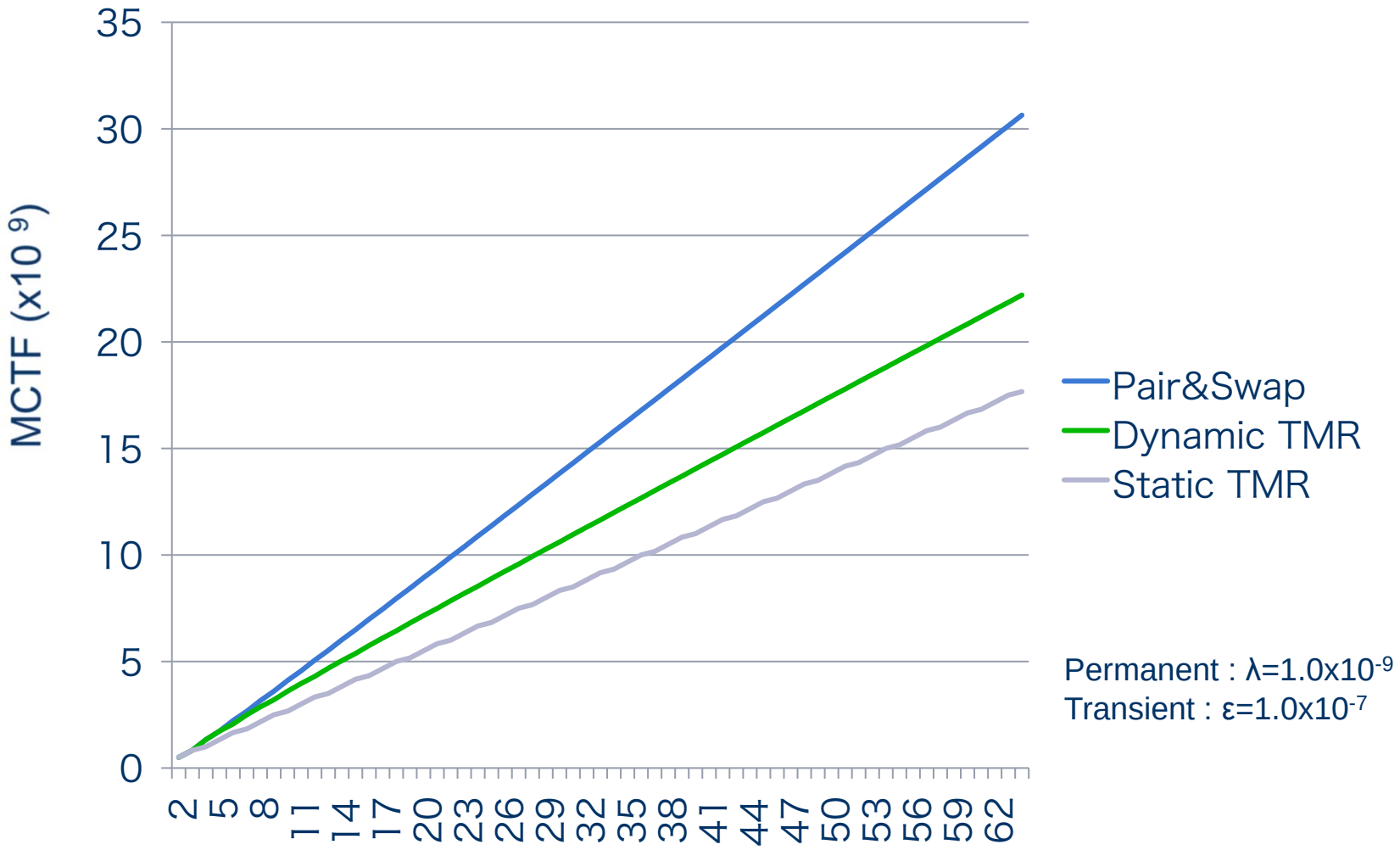
- If the number of cores is the same as 2n=3m at the initial state, the mean number of tasks of the proposed P&S is 1.5 times larger

Markov chain of static TMR

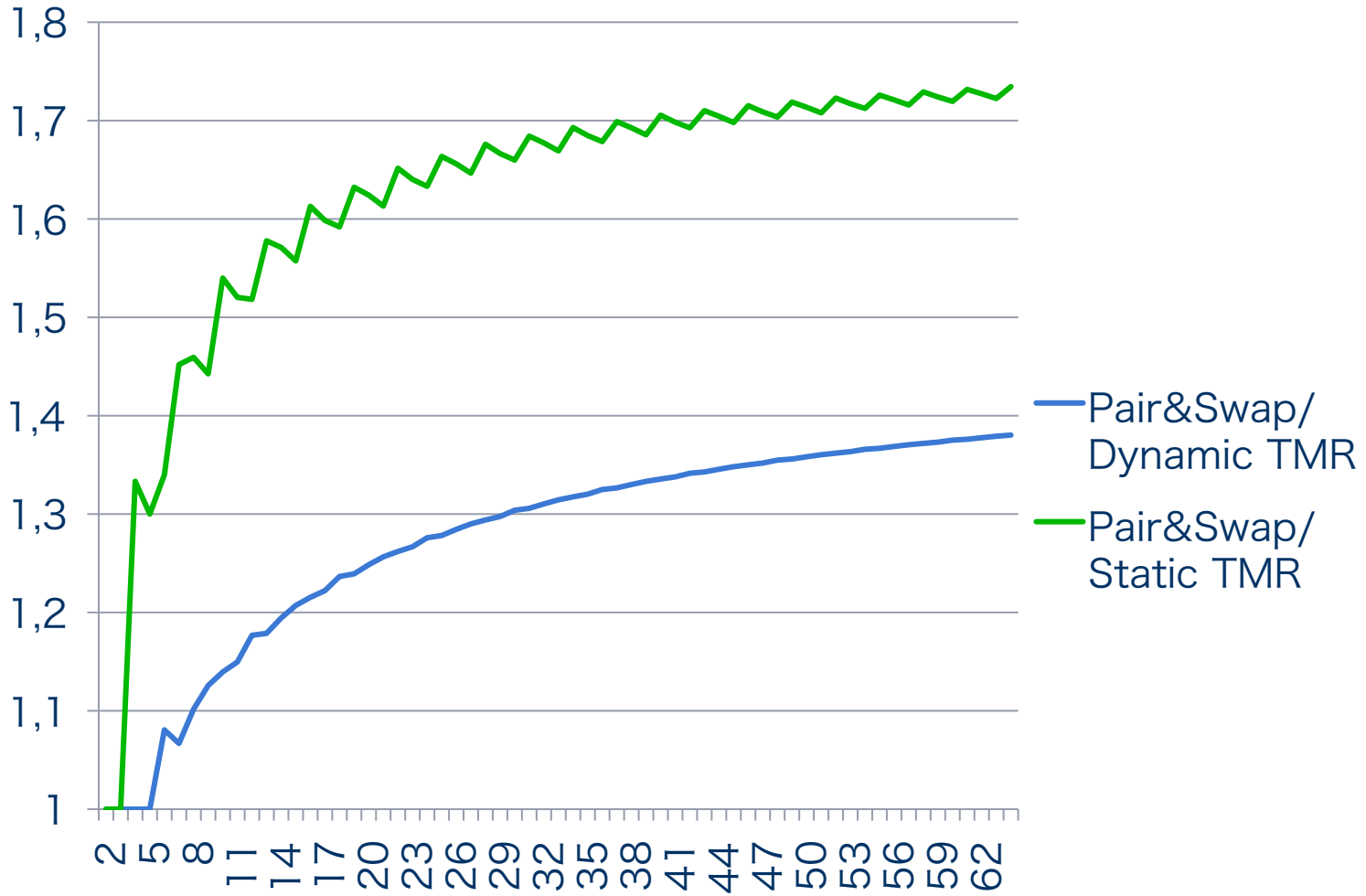


- When a permanent fault occurs in any TMR
  - → The remaining two cores compose a pair and compare their results each other
- When a permanent fault occurs in any pair
  - → Both two processor cores cannot be used

MCTF


35
30
25
20
15
10
5
0
MCTF (x10 9)
2 5 8 11 14 17 20 23 26 29 32 35 38 41 44 47 50 53 56 59 62
Initial number of cores
Pair&Swap
Dynamic TMR
Static TMR
Permanent : λ=1.0x10-9
Transient : ε=1.0x10-7

MCTF ratio


1,8
1,7
1,6
1,5
1,4
1,3
1,2
1,1
1
2 5 8 11 14 17 20 23 26 29 32 35 38 41 44 47 50 53 56 59 62
Pair&Swap/
Dynamic TMR
Pair&Swap/
Static TMR

Evaluation summary

- Advantage
  - Achieves about 1.4 times larger Mean Computation To Failure than dynamic TMR as the number of cores at the initial state increases
- Overhead
  - Comparison, check pointing, and task swapping use the system bus
    - → It might become a serious bottle-neck with an increasing number of processors

Conclusion

- Pair & Swap
  - Enables graceful degradation
  - Tolerates both transient faults and permanent faults
  - Requires only one extra task execution for the swap phase to decide whether the fault is transient or permanent and identify the failure core
  - Achieves about 1.4 times larger Mean Computation To Failure than dynamic TMR as the number of cores at the initial state increases

On going issues

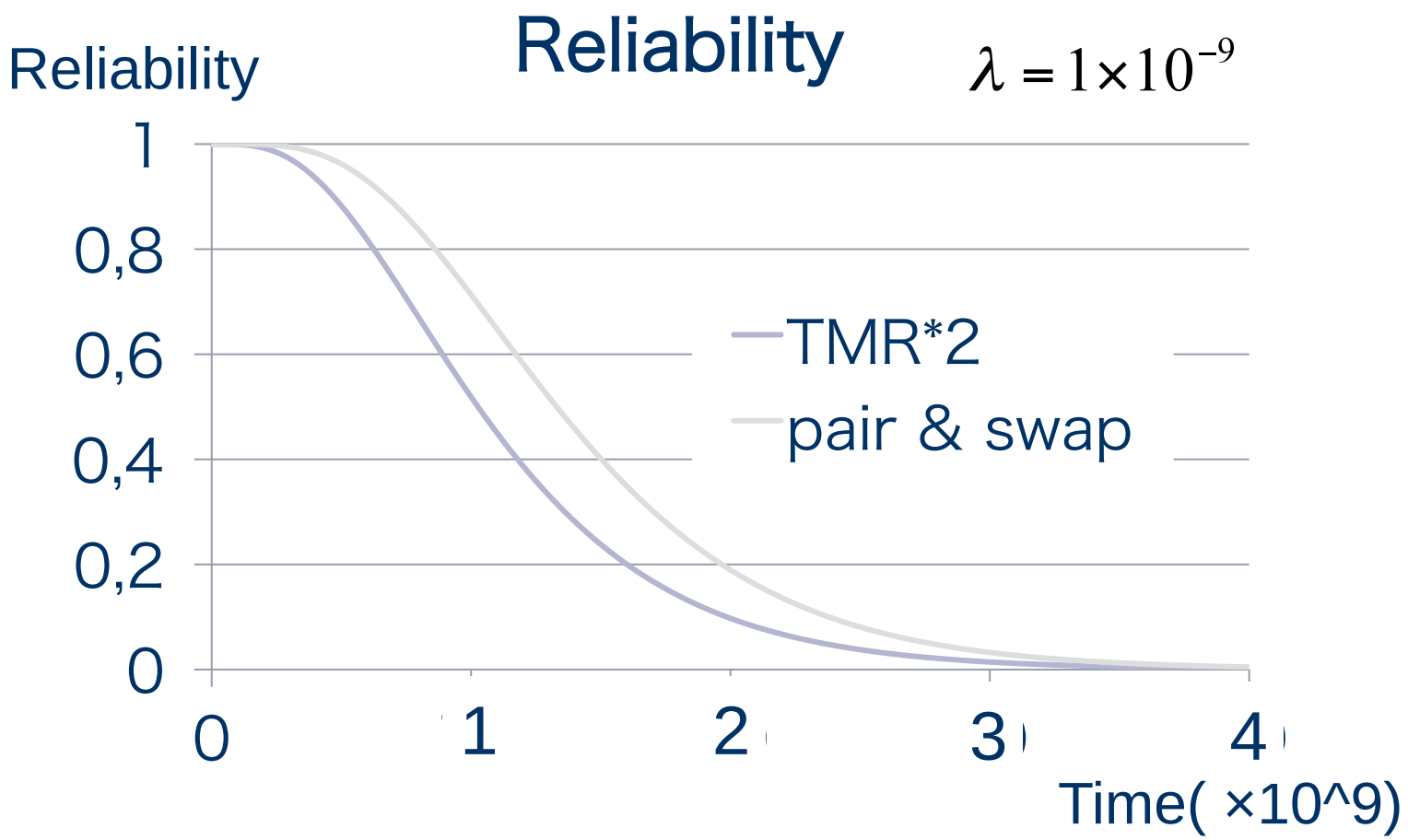
- Evaluate the overhead
  - Task migration
    - Data size, overhead time
    - Waiting time for synchronization
  - Checkpoint
- Implementation in real hardware
  - Use V850E* processor core and implement the proposed scheme based on the NoC (Network-on-Chip) architecture

Killer application

- Sensor – controller – actuator system
  - Each program size is small
  - What should be compared for replicated task execution is only output value
  - Home electronics, …
- The proposed scheme requires to execute tasks twice when a fault is detected

  → it is not suitable for hard-deadline-based application since the throughput is required to be twice larger than the normal execution

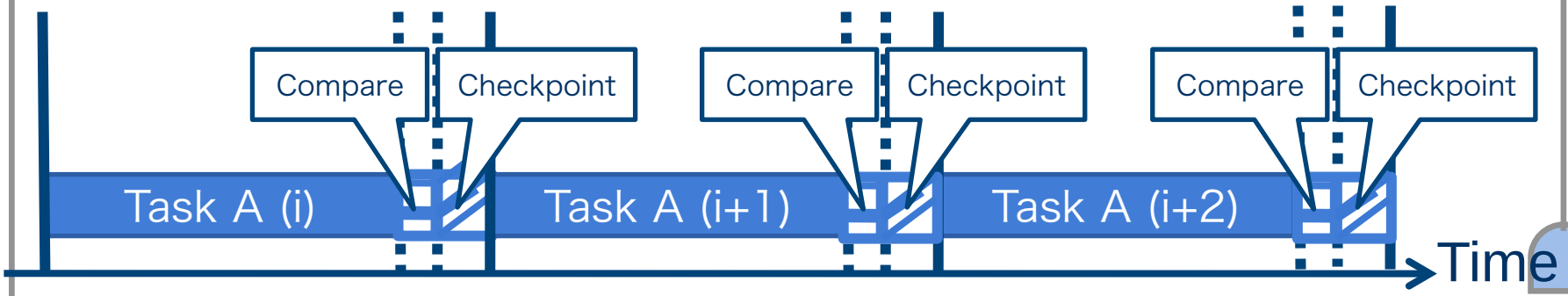
Reliability of 6-cores CMP



$$MTTF_{p\&s} = \frac{29}{20\lambda} \qquad MTTF_{2TMR} = \frac{69}{60\lambda}$$

Execution steps

1. Executes a given task based on the *task assignment table* which contains a list of all the tasks to be executed and the corresponding list of cores assigned to each task
2. Exchanges execution results between cores in each pair
3. Compares its execution results with the partner's results
4. Sends the comparison result to the decision unit
5. Receives comparison results of all the pairs which are broadcasted by the decision unit
6. Updates the task assignment table
7. Makes checkpoint data and stores it in the shared memory when its comparison result matches
8. Loads the corresponding checkpoint data from the shared memory when its comparison result mismatches or it belongs to the swapping pair



Fault analysis

- Digital relays in the power distribution network
  - The number of processor unit failure is high

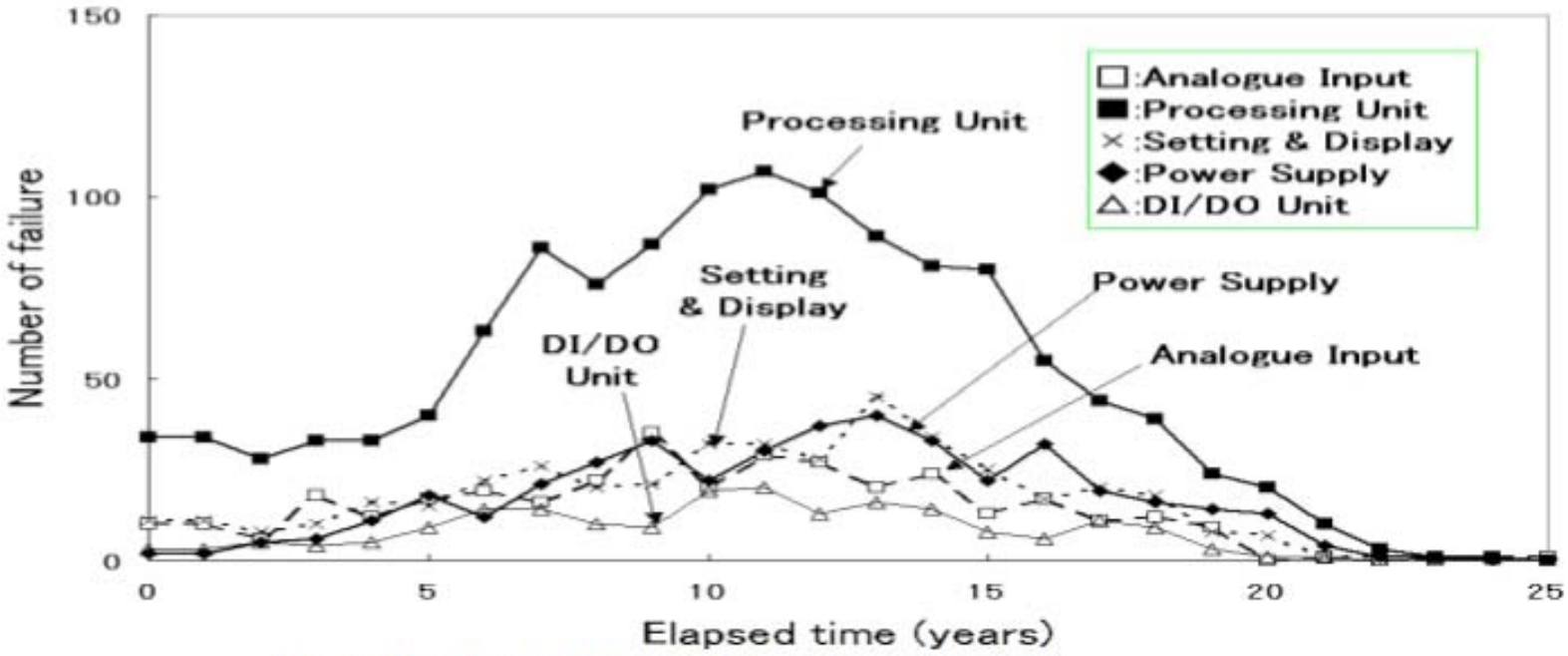
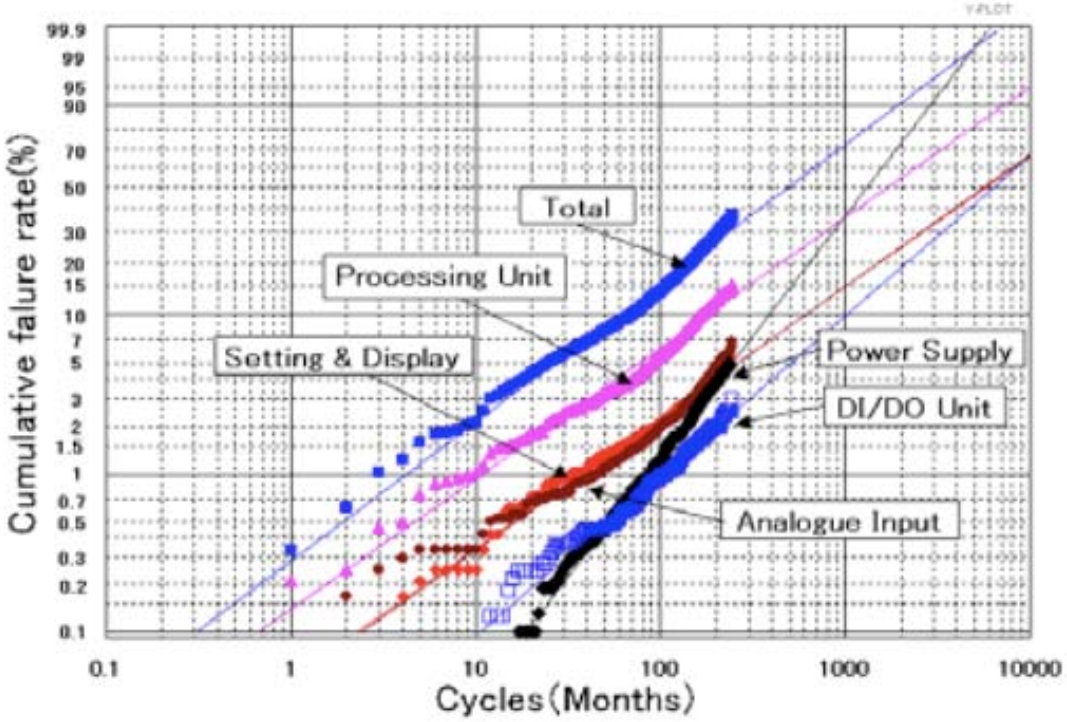


Fig. 5. Number of failures of main components.

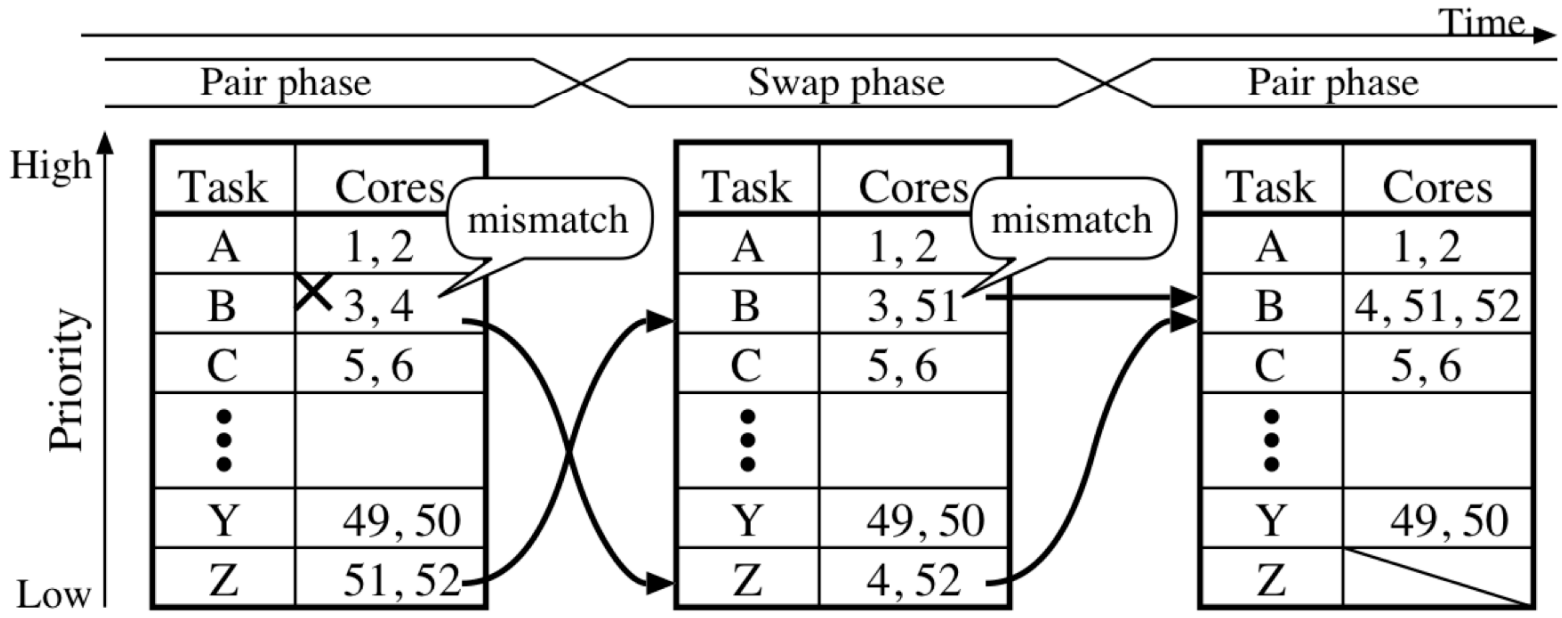
Fault analysis

- Digital relays in the power distribution network
  - The number of processor unit failure is high

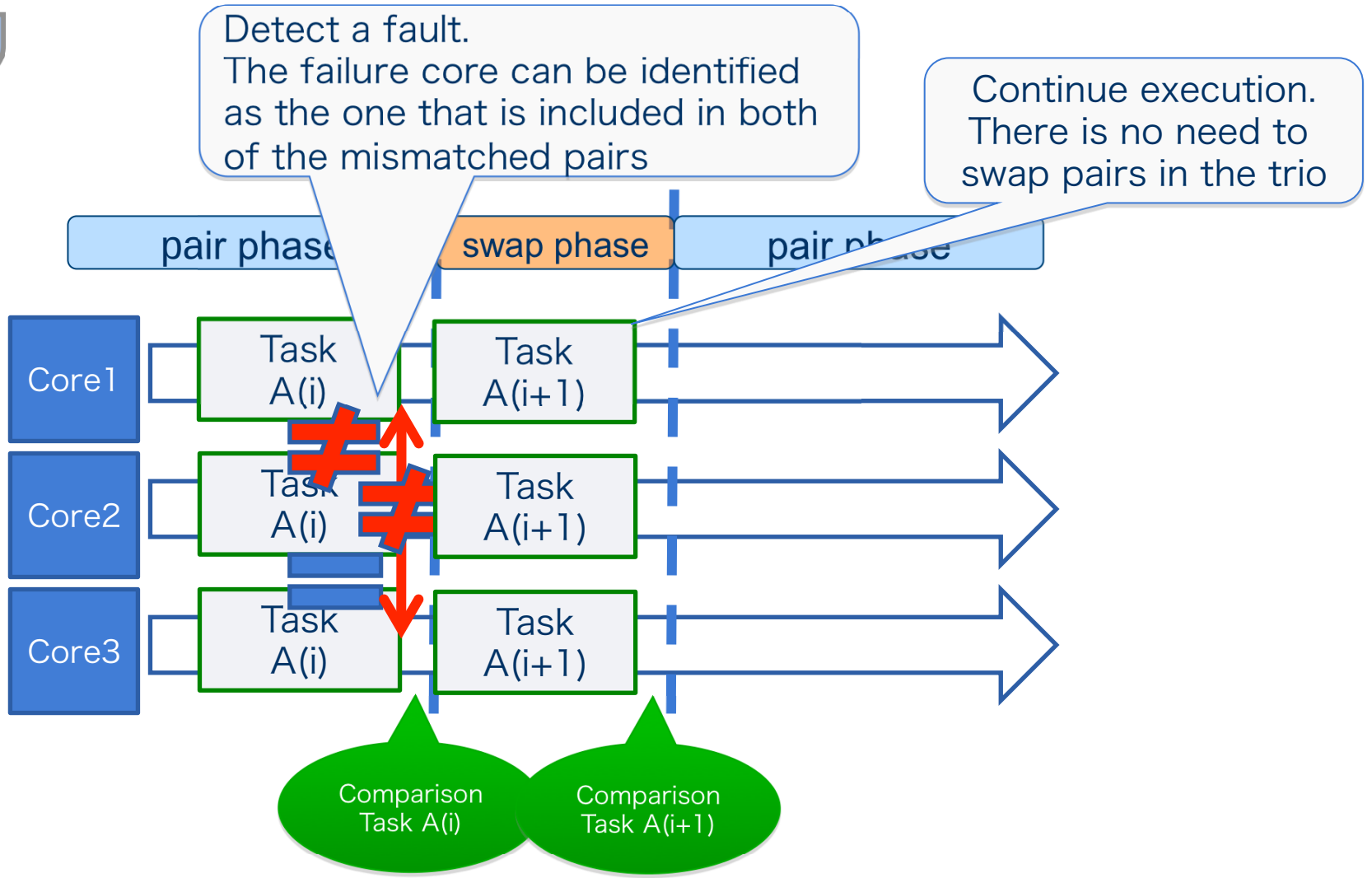


Failure Unit	m	η
Total	0.88	7.52E+02
Analogue Input	0.83	9.08E+03
Processing Unit	0.83	2.63E+03
Setting & Display	0.84	8.84E+03
Power Supply	1.54	1.67E+03
DI/DO Unit	1.02	9.21E+03

Task assignment table

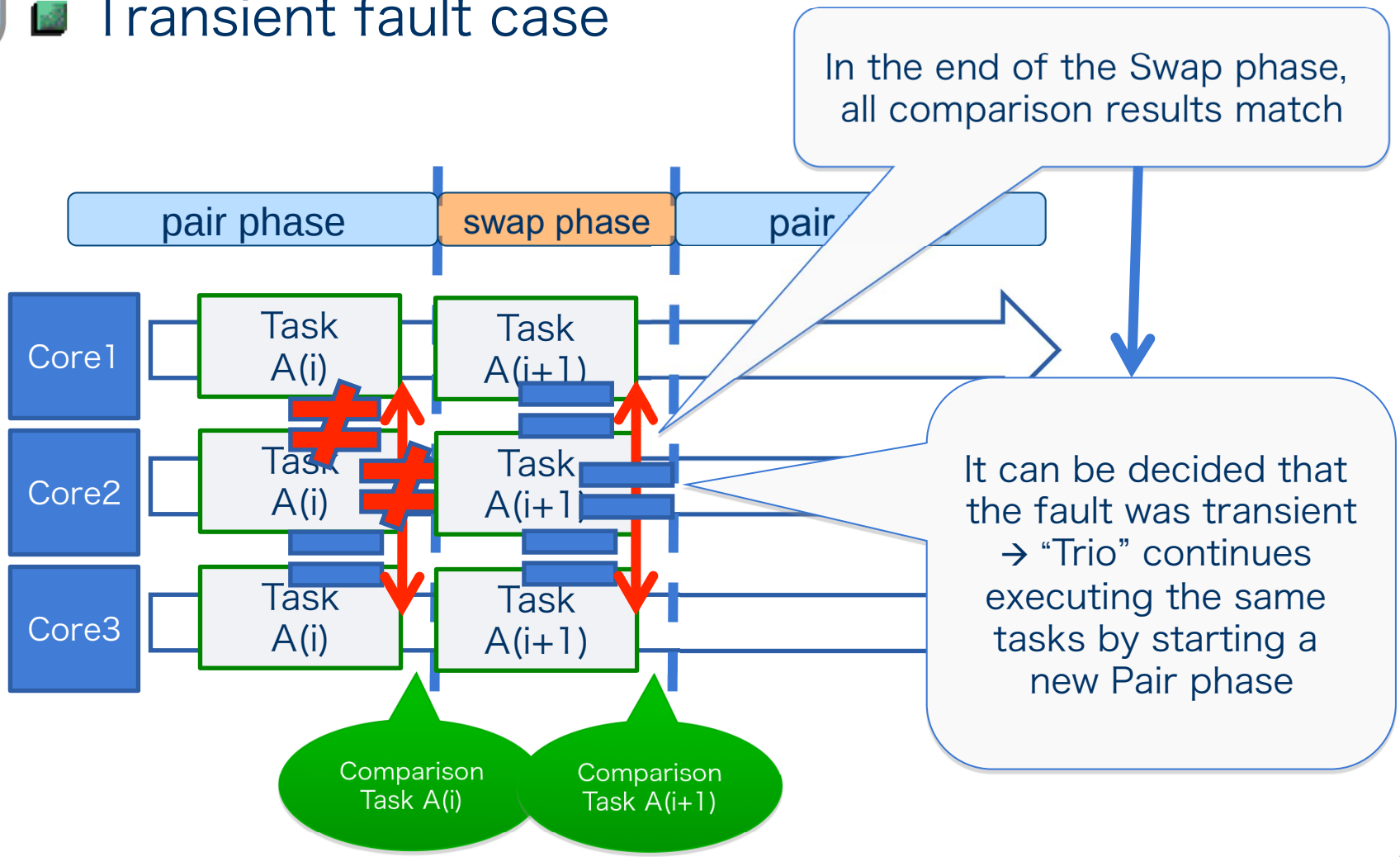


Trio: Swap phase



Trio: Fault location (1)

Transient fault case



Trio: Fault location (2)

- Permanent fault case