

UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE PATENT TRIAL AND APPEAL BOARD

ARM LTD., MEDIATEK INC,

Petitioners,

v.

DAEDALUS PRIME LLC,

Patent Owner.

Case No.: IPR2025-00207

U.S. Patent No. 8,984,228

PATENT OWNER'S DEMONSTRATIVE EXHIBITS

Daedalus Ex. 2008

This Case Turns on Architecture, Not Labels

The claims require one, logically unified caching agent that performs cache coherency for both the cores and the I/O module

- Coordination among **multiple autonomous controllers** is **not** a single caching agent
- **Agarwal** and **Sinharoy** disclose **peer-based, address-homed controllers**, not a unified agent
- Petitioner relies on **relabeling** (“collective,” “acting in concert”) rather than disclosure

If protocol coordination were enough, the word “single” would have no meaning

The '228 Patent

Claimed Architecture: A Single Caching Agent

700

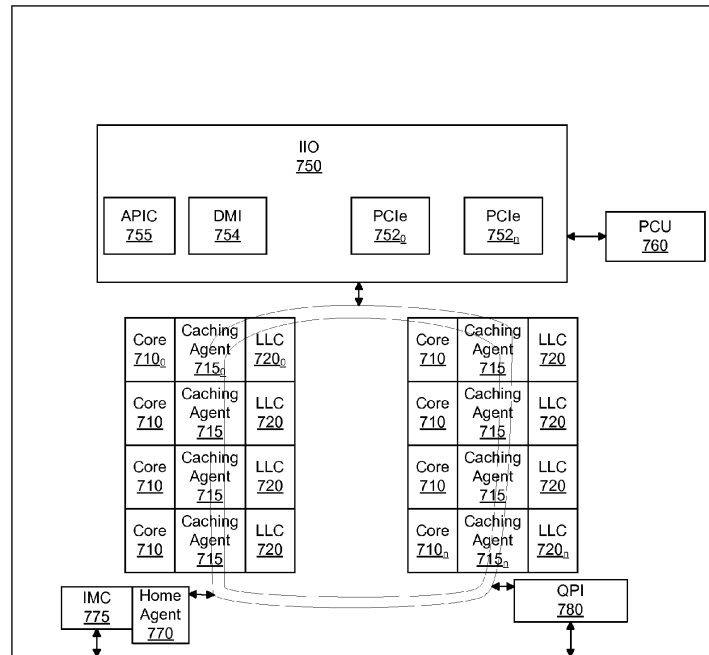


FIG. 6

“a caching agent ... the caching agent a single caching agent for the multicore processor and including a plurality of distributed portions each associated with a corresponding one of the plurality of cores”

— Claim 1

Claimed Architecture: A Single, Logically Unified Caching Agent

One caching agent, not multiple independent agents

The caching agent **may be physically distributed**, but:

- The distributed portions **together form the caching agent**

The caching agent performs cache coherency for:

- **All cores**, and
- **The integrated I/O module**

The caching agent includes **shared control logic**, including:

- **A table of requests** that stores pending requests for multiple sources

Unity is structural, not emergent from protocol behavior

The '228 Patent vs. Agarwal

What Agarwal Discloses

Cache coherence is managed by **directory controllers assigned on an address-homed basis**

For any given memory address:

- A **specific tile** is designated as the *home tile*
- The directory controller on that tile **alone** makes coherence decisions

Each directory controller:

- Maintains state **only for its assigned address range**
- Operates **independently** of other directory controllers
- Services requests from **any core** that accesses its addresses

What Agarwal Does Not Disclose

- ✗ A **single caching agent** that performs coherency for all cores
- ✗ A **logically unified agent** with shared control structures
- ✗ Any **per-core association** between a controller and a corresponding core
- ✗ A **shared table of requests** receiving pending requests from multiple sources

Why Agarwal Fails

Address-homed, autonomous directory controllers

≠

A single caching agent with distributed portions

Protocol Coordination Does Not Create a “Single Caching Agent”

Petitioner’s Reply Theory

- Multiple controllers that:
 - Exchange messages
 - Follow a coherence protocol
 - “Act collectively” or “in concert”
- ⇒ allegedly constitute **one caching agent**

Why That Theory Is Wrong

- **Directory and snooping systems always involve coordination**
 - Message passing is **how coherence protocols work**
- But coordination does **not** mean:
 - Shared control logic
 - Unified decision-making
 - A single ingress or arbitration point
- Each controller in Agarwal (and Sinharoy):
 - Maintains its **own state**
 - Makes **independent coherence decisions**
 - Acts without centralized authority

The '228 Patent vs. Sinharoy

What Sinharoy Discloses

Multiple cache and coherency controllers operating as **peers**

Coherence maintained through **broadcast snooping** on a fabric

Each controller:

- Maintains its **own control logic and state**
- **Independently evaluates** coherence transactions
- Takes action according to protocol rules

Independent controllers reacting to broadcast traffic remain independent controllers
Peer-level coherence $\neq$ a single, logically unified caching agent

Why Sinharoy Fails the Claim Requirement

- ✗ No **single caching agent** that performs coherency for all cores and I/O
- ✗ No **unified control structure** governing coherency decisions
- ✗ No **centralized decision-making**
- ✗ Controllers act **autonomously**

Broadcast Snooping is Communication

Broadcast snooping (in Sinharoy) is a **communication mechanism**

- It allows multiple agents to observe the same transaction

It does **not**:

- unified control
- centralize decision-making
- transform peers into portions of a single agent

Failures of the Prior Art

The Claims Require Per-Core Functional Association — Not Address-Homed or Shared Control

What the Claims Require

- Each **distributed portion of the caching agent** must be associated with a corresponding one of the plurality of cores
- This requires:
 - **Functional association**, not mere proximity
 - Responsibility for coherency **on behalf of that core**
 - As part of the **single, unified caching agent**

Why Agarwal Fails This Requirement

- Directory controllers are **address-homed**
 - Assigned to memory regions, not cores
- A directory controller:
 - Routinely performs coherency operations for **multiple cores**
 - Services whichever core accesses its addresses
- No one-to-one relationship between a controller (or agent portion), and a specific core

Why Sinharoy Fails This Requirement

- L3 cache regions are:
 - **Address-homed**
 - Accessible by **all cores**
- L3 region controllers:
 - Act as **shared resources**
 - Service requests from multiple cores depending on address
- No controller is functionally tied to a corresponding core

The TOR is a Shared Ingress Structure Not an Execution Resource

What the Claims Require

- The **single caching agent** includes:
 - “a **table of requests to store pending requests**”
- The table:
 - Receives requests from **multiple sources** (cores and I/O)
 - **Stores pending requests** before execution
 - Functions as a **front-end arbitration and control structure**

How the Specification Describes the TOR

- A **shared control structure** within the caching agent
- Downstream of ingress queues, upstream of execution
- Used to:
 - Hold pending transactions
 - Arbitrate among requests
 - Manage ordering and deadlock avoidance

What Petitioner Points To Instead

- RC machines
- Write machines
- Per-controller or per-unit buffers

Why That Fails (Architecturally)

- These are **execution resources**, not ingress control
- They act on transactions **after** scheduling
- They are:
 - Distributed across controllers
 - Independent of one another
 - Not shared tables storing pending requests from multiple sources

Allocation Transactions are Specific IO-Originated Transaction Flows

What the Claims Require

- An **allocation transaction** that:
 - **Originates from an IO device**
 - **Is received by an integrated IIO module**
 - **Is forwarded by the IIO module to the caching agent**
 - Results in **direct storage of data into shared cache**
 - Occurs **without communicating with system memory**

What Petitioner Relies On Instead

- “Cache injection”
- Internal coherence activity
- Data appearing in cache as an end result

Allocation Transactions Continued

Why That Does Not Satisfy the Claims

- Cache injection describes **where data ends up**, not:
 - **Who originated the transaction**
 - **Who received it**
 - **Who forwarded it**
- Neither Agarwal nor Sinharoy discloses:
 - An **IO-device-originated allocation transaction**
 - Receipt of that transaction by an **IIO module**
 - **Forwarding by the IIO module** to a shared caching agent
- Internal coherence handling is **not a proxy** for the claimed IO-to-IIO-to-agent sequence

“Without Communicating with System Memory” Is Not Shown

- Petitioner points to coherence mechanisms that:
 - Involve system-level coordination
 - Depend on directory or snoop traffic
- The claims require **direct cache storage without system memory communication**
- Petitioner relies on **inference**, not disclosure

End result ≠ claimed transaction semantics
Petitioner shows data in cache — not the claimed allocation transaction that puts it there

Expert Labels Do Not Substitute for Disclosure

Petitioner's Reliance on Expert Testimony

- Describes prior-art components as:
 - “acting collectively”
 - “functioning as a single caching agent”
 - “effectively unified”
- Uses **descriptive terminology** to bridge missing claim elements

Why That Is Legally Insufficient

- Whether a reference discloses a claim limitation turns on:
 - **What the reference teaches**, not
 - How an expert **characterizes the outcome**
- Expert testimony may:
 - Explain what is disclosed
- It may **not**:
 - Add missing structure
 - Rewrite the reference
 - Collapse architectural distinctions into functional labels

Application Here

- Neither Agarwal nor Sinharoy discloses:
 - A **single caching agent**
 - **Per-core associated portions**
 - A **shared table of requests**
 - The **claimed allocation-transaction sequence**
- The expert:
 - Relabels independent controllers
 - Describes protocol behavior as architectural unity
- Supplies conclusions untethered to disclosure

Conclusion:

Petitioner Has Not Met Its Burden

What Petitioner Was Required to Show

- That the asserted prior art discloses:
 - A **single caching agent**
 - With **distributed portions associated with corresponding cores**
 - Including a **shared table of requests**
 - Receiving **IO-originated allocation transactions**
 - **Without communicating with system memory**

What the Record Shows Instead

- **Agarwal:**
 - Address-homed, autonomous directory controllers
- **Sinharoy:**
 - Peer-based controllers using broadcast snooping
- **Both:**
 - Rely on protocol-level coordination
 - Lack unified control structures
 - Do not disclose the claimed transaction sequence

What Cannot Cure These Failures

- **✗** Relabeling independent controllers as “collective”
- **✗** Describing protocol outcomes instead of structure
- **✗** Expert testimony substituting for missing disclosure

POR 1-3, 9-15, 35-37
Ex. 1001 7:35-46, 47-49, 60-63
Ex. 2006 ¶¶ 29-33, 39-44, 63-66, 108-111, 114-117
PO Sur-reply 1-2, 5-9, 10-14

ARM LTD v. Daedalus Prime IPR2025-00207
Patent Owner's Demonstrative Exhibits -- Not Evidence
Daedalus Ex. 2008