This course provides a detailed description of the MIPS Coherent Processing System from a programming point of view.

Course Overview

- **Description of a MIPS Coherent Processing System**
- **Software visible changes**
  - Global Configuration Registers
  - Global Interrupt controller
  - Cluster Power Controller
  - Booting a CPS system
  - Segmentation Control and EVA
  - Additional Changes

MIPS

2

This class is designed for Software programmers. It will give you an understanding of the additional programming interfaces for a MIPS Coherent Processing system.

This introduction section covers an overall description of the Coherent Processing System as it pertains to a MIPS multi Core implementation.

Additional sections cover programming details on

Global Configuration Registers,

Global interrupt controller,

Cluster Power Controller

Booting a CPS system

Segmentation Control and EVA

and additional changes

CPU Cores

- **4 CPU cores are supported**
  - Multi threaded Cores that have been enhanced for a multi-core configuration.
  - Single Cores that have been enhanced for a multi-core configuration.
- **Core Enhancements:**
  - MESI coherence states have been added to the data cache, with proper cache controller handling.
  - The Data Cache Tag array is duplicated to allow coherence requests to access the cache in parallel with normal load/store traffic.
  - A read-only OCP Slave port, called the Intervention Port, has been added to receive coherent requests from the Coherence Manager.
  - Core grouping into Coherent Domains.

MIPS

There are 2 different types of products that are supported, one that contains multi threaded cores and one the contains non-multi threaded cores. There is no mixing of CPU cores within a CPS.

+ Each core has been enhanced to support a coherent processing system. These enhancements include:

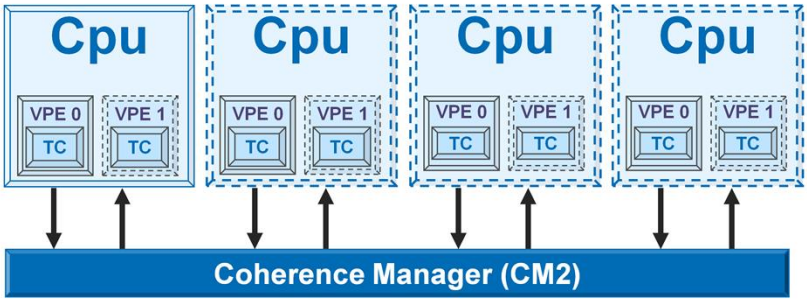
+ The MESI coherence protocol has been implemented to insure coherent caches for all cores in a coherent domain. You can control the coherent state of memory by selecting cache attributes, defining regions of memory that have different cache coherency attributes and by controlling which cores belong to a coherent domain.

+ For performance reasons the data cache tag array has been duplicated. This allows the normal operation of the data cache in the execution pipeline without interference by coherent requests from other cores or devices in the system.

+ Coherent requests are routed to each core through a separate port called the intervention port. This allows for the snooping of a processors cache.

+ Each core in the system can be treated as a independent processor or part of what's call a coherent Domain. As part of a Coherent Domain a processor will maintain cache coherency with other processors in that domain.

Coherence Manager (CM)



- High-performance coherent interconnect controls coherent traffic between processors, memory, I/O and interrupts.
- Supports direct cache to cache transfers
- Enables speculative memory reads to reduce latency
- Optimized 256-bit interface to optional MIPS® SOC-it® L2 cache controller

MIPS 4

Here is a diagram of a coherent Processing System with 4 processors. In terms of a MIPS Coherent Processing System, a CPS is a cluster of homogeneous processors that share memory, interrupts and I/O devices. All processors in the system can be a multi threaded or single threaded but not both.

The system can have just one processor. This adds cache snooping to a single processor system. However the real intention of the CPS is to connect multiple processors together in a totally coherent fashion ideally suited to run a symmetric multi processing OS such as SMP Linux.

+ To accomplish this the system contains a Coherence Manager that allows each processor in the system the same view of memory.

+ The Coherence Manager (CM) is responsible for establishing the global ordering of requests from all elements of the system and sending the correct data back to the requester.

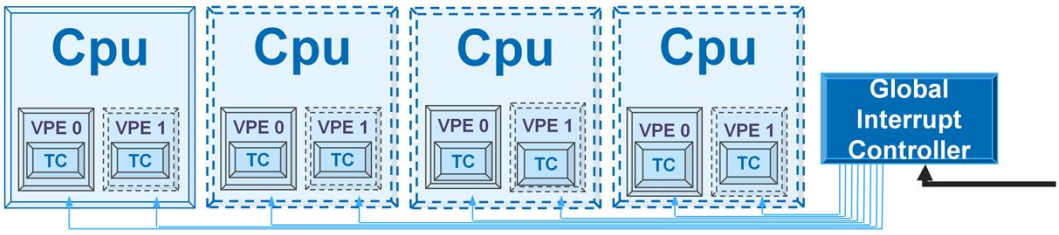
+ It supports Cache to Cache transfers so if a read request hits in another Processor's L1 cache and the cache line is in the Exclusive or Modified state, it will return the data to the CM and it will be forwarded to the requesting CPU, reducing latency on the miss.

+ To further improve performance it supports Speculative Reads. This means that Coherent read requests are forwarded to the memory interface in parallel to the lookup in the other L1 caches. This is speculating that the cache line will not be found in another Processor's L1

cache. If another cache was able to provide the data, the memory request is not needed. The CM will cancel the speculative request, dropping the request if it has not gone out, or dropping the memory response if it has gotten one.

+ And it has a 256 bit interface to the L2 Cache controller to allow full cache line transfers.

Global Interrupt Controller (GIC)



- Multiprocessor-aware interrupt controller
- Configurable number of system interrupts - 8 to 256
- Masking and mapping of vectored interrupts to a particular Processing Element
- Map local interrupt sources to local interrupt vectors
- Standardized mechanism for sending inter-processor interrupts

MIPS

5

Connected to the coherence manager is a global interrupt controller.

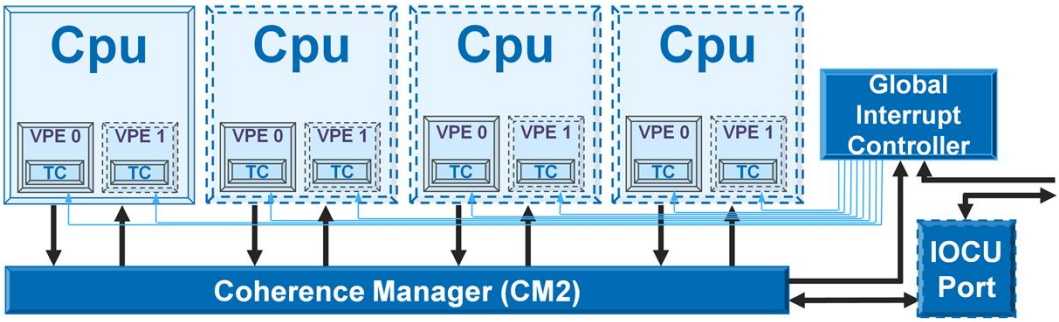
+ It can control from 8 to 256 interrupt sources configurable in groups of 8.

+ It can map any external interrupt source to any Processing Element.

+ It can also map local interrupts such as timer and performance counter interrupts to interrupt vectors

+ It also provides a mechanism for sending inter-processor interrupts

I/O Coherence Unit (IOCU)

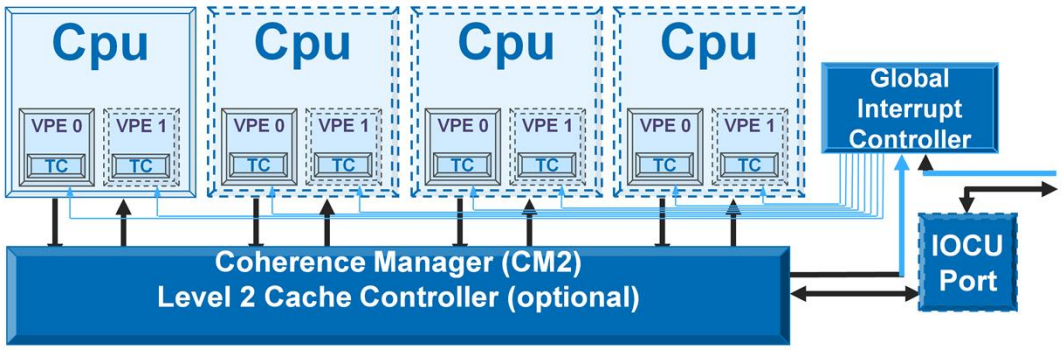


MIPS

6

The optional I/O coherency unit is a customizable block that connects I/O devices on a system interconnect to the Coherence Manager. With the IOCU transfers from devices maintains I/O coherence of the caches in all coherent CPUs in the cluster. Up to 2 IOCUs are supported in a interAptiv CPS The IOCU acts as an interface block between the Coherence Manager 2 (CM2) and coherent I/O devices. Coherent reads and writes of I/O devices generate interventions in other coherent CPUs that query the L1 cache. I/O reads access the latest data in caches or in memory, and I/O writes invalidate stale cache data and merge newer write data with existing data as required.

Level 2 Cache (L2)



MIPS

7

The L2 cache controller is included in the CM improve memory performance.

ITC and Power Management

- **Inter Thread Communication Unit (ITC)**
  - A customizable block that allows any of the processing units in the CPS to communicate with any other via shared hardware resources.
- **Manage Power Consumption**
  - Setup registers to control power states
    - Per-core control
    - CM-level control

MIPS

8

There are 2 additional blocks that can be added to the CPS:

+ The ITC unit allows the sharing of semaphore and messages between any execution unit with the CPS.

+ The Cluster Power Controller or CPC manages static leakage and dynamic power consumption based on system-level power states assigned to the individual components of the Coherent Processing System.

+ Each processor in the Coherent Processing System is its own power domain and consists of a power sequencer to control reset, isolation, and power rail enables. This sequencer is programmed by the Cluster Power Controller. The CPC provides global setup registers, as well as local core configuration registers.

+ The CPC controls the power mode of the Coherency Manager which it can power down when all Processors go to a power down state and all coherent activity has stopped.

Cache Coherence

- **Cache States**
  - MESI protocol
    - **M**odified – Dirty cache line
    - **E**xclusive – only L1 cache with line
    - **S**hared – cache line present in other L1 caches
    - **I**nvalid – invalid cache line

CCA	Description
0,1,6	Reserved
2	Uncached, non-coherent
3	Writeback, writ-allocate, non-coherent
4	Writeback; write-allocate, coherent, exclusive
5	Writeback; write-allocate, coherent, exclusive on write
7	Uncached Accelerated, non-coherent

MIPS

9

First it is important you understand the MESI cache line protocol. It is how the system determines which cache has the most up to date cache line for the data being requested.

+ M is for Modified: It means that this cache has a dirty copy of the cache line and the line is not present in any other L1 data cache, making this line the only up-to-date copy of the data in the system.

+ E is for Exclusive: It means that this cache has a copy of the line with the right to modify. The line is not present in other L1 data caches. The line is still clean meaning that it is consistent with the value in L2 cache or memory.

+ S is for Shared: It means that this cache has a read-only copy of the line. The line may be present in other L1 data caches, also in a Shared state. The line will have the same value it has in the L2 cache or memory.

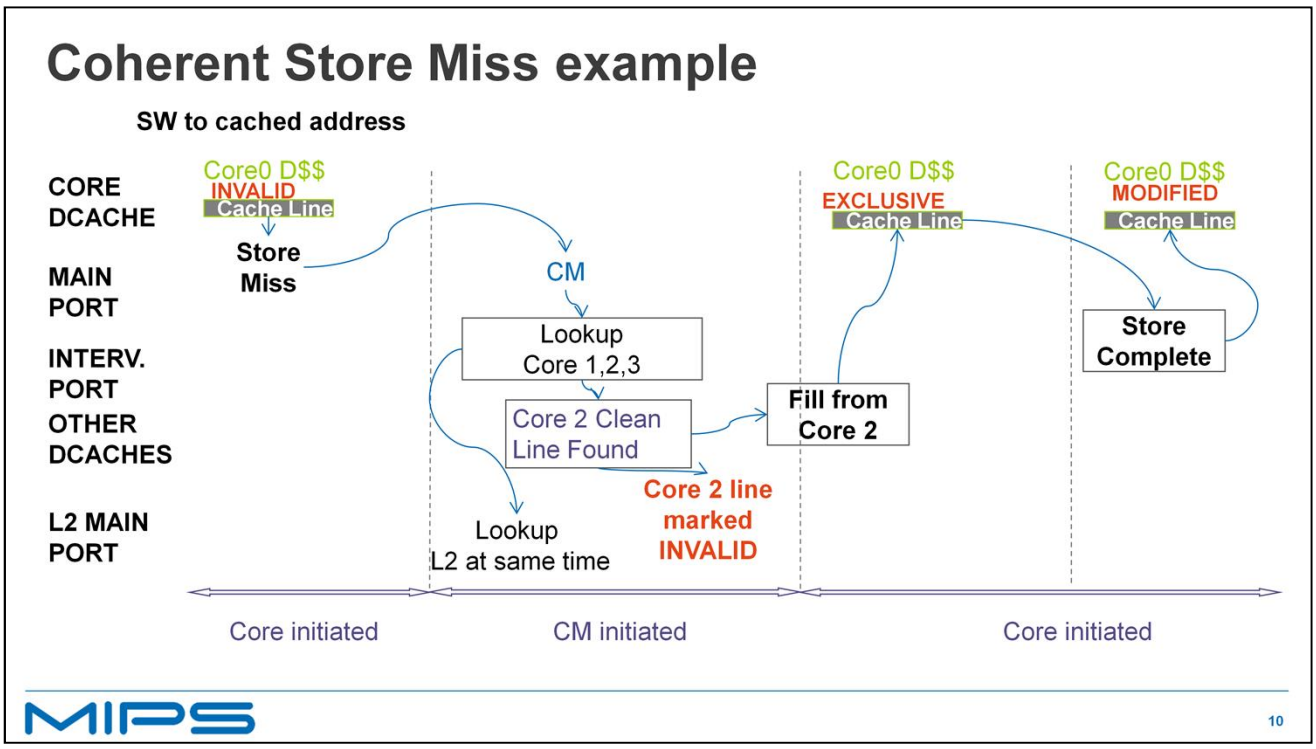
+ I is for Invalid: It means that the data in this cache line is not valid and replaceable.

When a Processor is part of a Coherent Domain, each L1 cache in the domain is connected through an intervention port. The intervention port is use to snoop the caches in the domain to see if other caches have a copy of the data being requested. If a requested cache line is in an Exclusive or Modified state, it has the most recently updated copy of the data so that data is returned with its response on a read type intervention.

+ The Coherent CCAs are number 4 and 5. These differ only in the handling of load misses. For CCA 4, a load miss will request Exclusive ownership. This can be chosen for data that is not expected to be shared. For CCA 5, a load miss will request Shared ownership; however, if there are no other Sharers, the CM will normally automatically upgrade the line and indicate that it should be installed as Exclusive.

Coherent requests from all CPUs and I/O devices are sent to each CPU's Intervention Port. Each CPU sees all of the interventions, including those from its own requests, called self-interventions to keep all requests in the same order. Each CPU updates the MESI state of its cache lines in response to these interventions.

As a reminder, the CCA is set for KSEG 0 by setting the K0 field bit of the CP0 Config Register. For systems without a TLB KUSEG CCA is set in the KU field and the K23 field sets the CCA for KSEG 2 and 3. For systems with a TLB all cacheable segments aside from KSEG 0 are set in the TLB entry.



Here is an illustrated example of a Store word the misses in the L1 cache with a write back CCA

+ Core 0 initiates a intent to modify message toward the coherence manager.

+ The coherence manager sends interventions toward all cores and the L2 cache at the same time

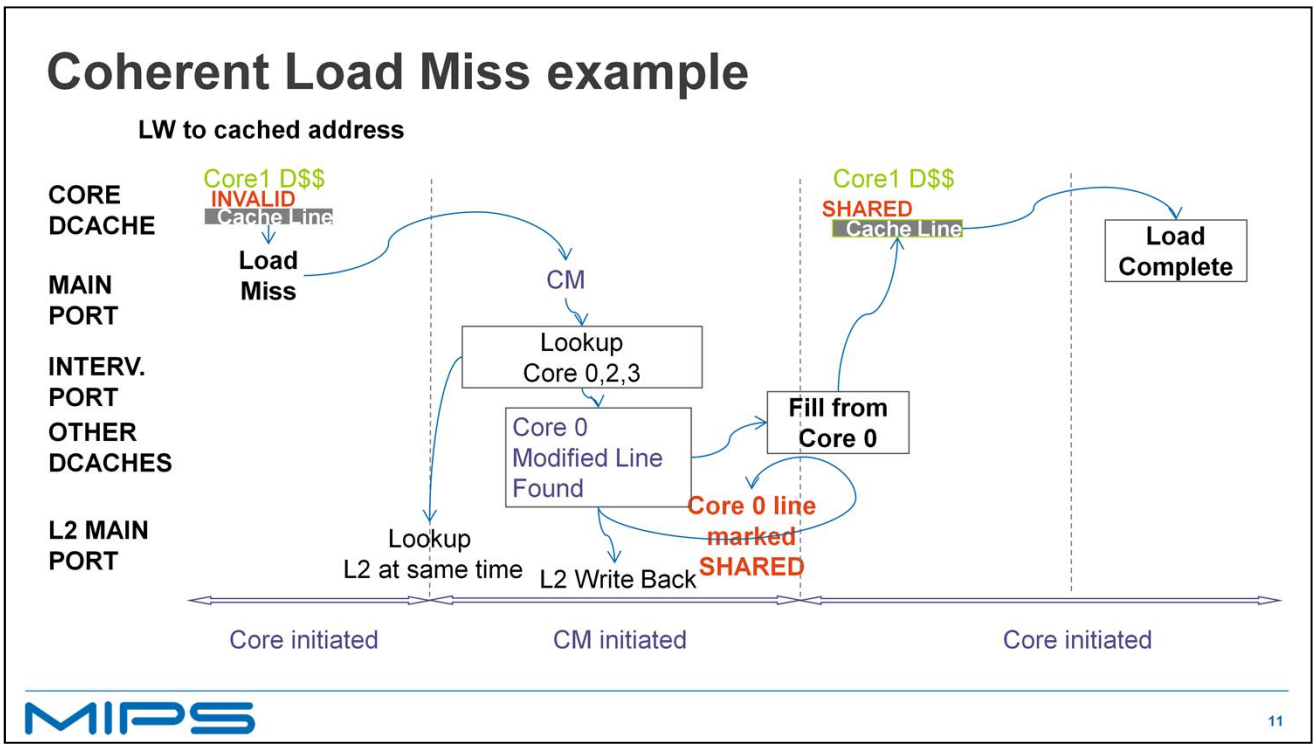
+ Core 2 responds with a hit with it's line in an Exclusive State meaning it has the only clean L1 copy and is consistent with the L2 cache

+ There is a cache to cache transfer from Core 2 to core 0 over the intervention Port

+ Core 2's cache line is marked invalid

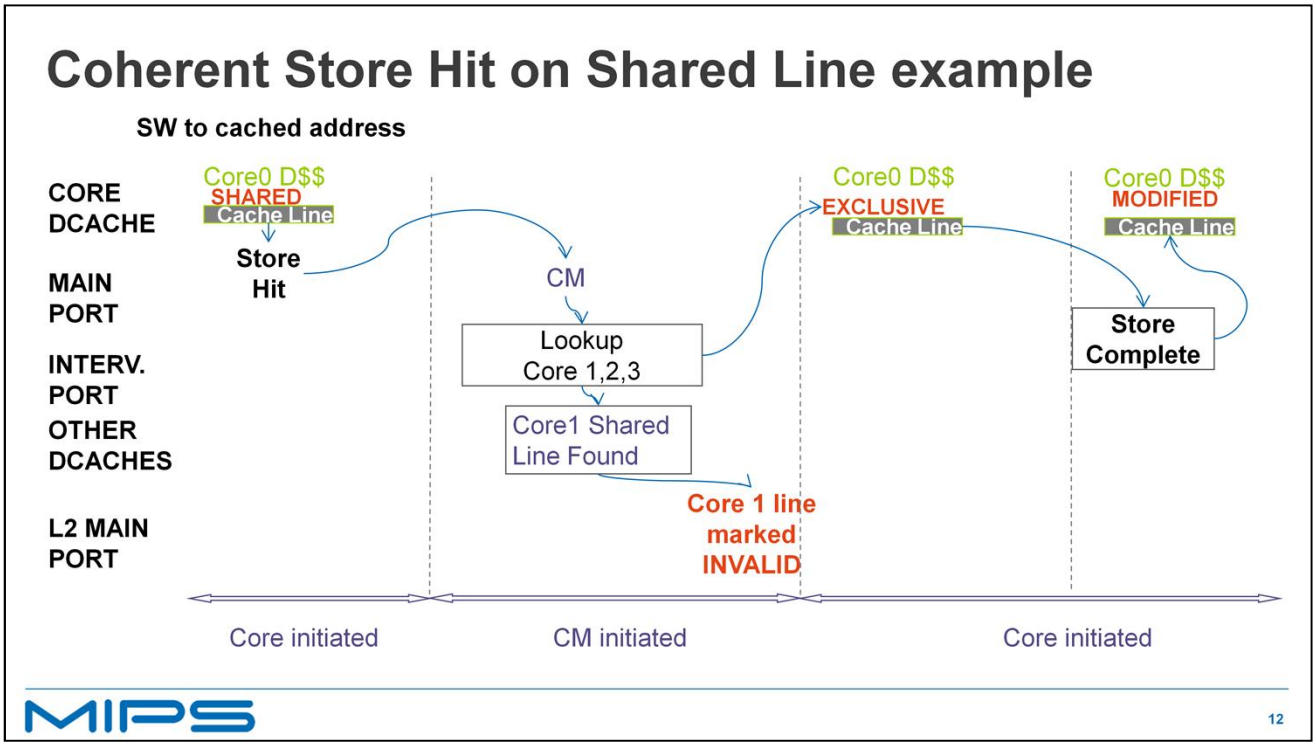
+ Core 0's cache line is marked Exclusive

+ After the Store Word Completes the Cache line state changes to modified.



Here is an illustrated example of a Load word the misses in the L1 cache of Core 1

+ Core initiates a no intent to modify message toward the coherence manager.
+ The coherence manager sends interventions to all cores
+ where core 0 responds with a hit – 'Modified.'
+ The coherence manager now initiates a write-back of the modified line, and moves line data from the core 0 intervention port to the L2 cache.
+ There is a cache to cache transfer from Core 0 to Core 1 over the intervention Port
+ Core 0 cache line then migrates to 'Shared' status.
+ Core 1 cache Line is marked as Shared
+ The Load Word then completes



Here is an illustrated example of a Store word that hits on a Shared L1 cache line of Core 0 with a write back CCA

+ Core 0 initiates a intent to modify message toward the coherence manager.
+ the coherence manager initiates interventions to all cores.
+ Core 1 responds with a hit 'Shared' and invalidates its line.
+ Core 0 is upgrades its cache line to 'Exclusive.'
+ After the store has completed, the cache line status migrates to 'Modified.' State