TECNOLÓGICO DE COSTA RICA

ARQUITECTURA DE COMPUTADORES II

PROYECTO 1

COHERENCIA DE CACHE

DOCUMENTO DE DISEÑO

REALIZADO POR:

EMANUEL ESQUIVEL LÓPEZ

PROFESOR:

LUIS BARBOZA

I SEMESTRE - 2021

CARTAGO, COSTA RICA

Índice

SECTION 1

Requerimientos del sistema

Numero	Nombre de requerimiento
1	El sistema debera tener 4 CPU
2	Los CPU deberan tener cache L1
3	Cada CPU estara conectado a una cache L2
4	El protocolo de comunicacion es con L1 debera ser MSI
5	El protocolo de comunicacion es con L2 debera ser directorio
6	El sistema debera tener interfaz grafica
7	La cache L1 debera tener 2 bloques
8	La chache L2 debera tener 4 bloques
9	La memoria principal debera tener 8 bloques
10	Los datos deberan ser Hexadecimal de 16 bits

Tabla 1: Requerimientos básicos del sistema

Como se ve en la tabla esos requerimientos son los básicos para poder realizar el proyecto, con esto se puede profundizar mas, lo cual se especificara mas en el detalle de cada implementación.

SECTION 2

Soluciones propuestas

2.1. Opción 1

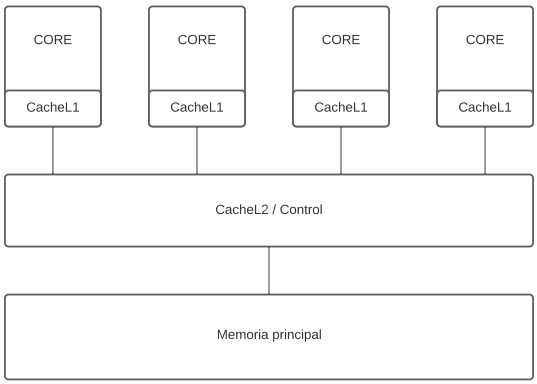

Figura 1: Primera solución propuesta

Para esta primera se realiza como se ve en el diagrama, ya que podemos obtener desde la misma L2 ya que esta interactúan todos los cores, como se ve no hay una entidad **aparte** para controlar las interacciones de los demás procesadores, solo existe las memorias básicas y los procesadores, esto se pensó ya que para el ahorro del numero de clases este se puede obviar y tener un control general en L2 pero esto conlleva a una clase mas grande.

2.2. Opción 2

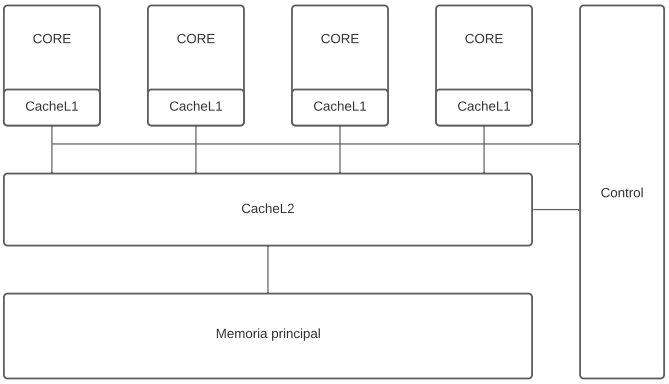

Figura 2: Segunda solución propuesta

En esta a diferencia de la opción 1 si tenemos un control de todas la memorias en una clase aparte, ya que en si se va a trabajar en comunicación con las memoria L1 y L2, así llevando a casa memoria los respectivos cambios que sufren las demás, informando los cambios hechos por los demás procesadores y así indicar el estado en sus memorias.
El principal funcionamiento del Control seria:

- Llevar el estado de cambios a la memoria L1
- Informar a los procesadores de cambios en L2.
- Llevar el control de los datos recientemente actualizado.

3

SECTION

Comparación de soluciones y selección final

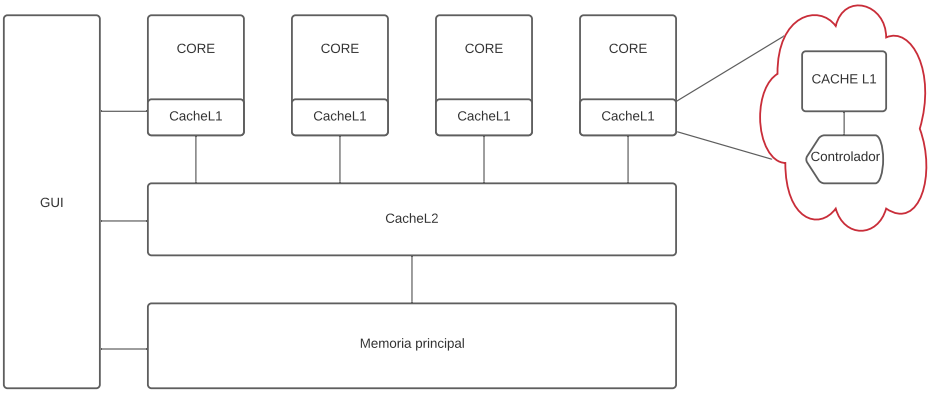

Figura 3: Solución 1 Elegida.

Se elige esta solución ya que se logra ver que cada core principal tiene su propia memoria cache L1, se ve en la parte derecha que esta en su interior tiene un controlador propio, el cual es el encargado de estar actualizando la memoria antes de cada operación.
El sistema de cache L1 se encarga de escribir a L2 y Memoria principal si es necesario, por lo que en general es la mente principal de todo el programa, cada vez que se realiza una acción en la memoria por parte del procesador, menos por *CALC*, L1 tiene ese método el cual es el encargado de consultar a L2 los estados de las direcciones de memoria, así como también los estados y propietarios, con esto podemos ver si el dato en L1 es o no valido para la lectura, o bien es valido y puede ser leído desde L2. En caso de que no sea valido este deberá ir a la memoria principal y actualizar.
La memoria principal no tienen nada mas direcciones y datos.
El sistema al inicio tiene todo invalido y en blanco los valores en memoria, esto para forzar a escribir o a leer directamente a memoria.
Los valores en L2 son actualizados con lecturas o con escrituras a la memoria principal.
Con esto nos ahorramos una clase extra llamada control, ya que el control lo lleva cada memoria L1 por separado, con lo cual tendríamos un mejor manejo de manera independiente, y no siempre que se actualice de manera innecesaria por ejemplo cuando se realiza CALC o operaciones de escritura.

4

SECTION

Implementación de diseño

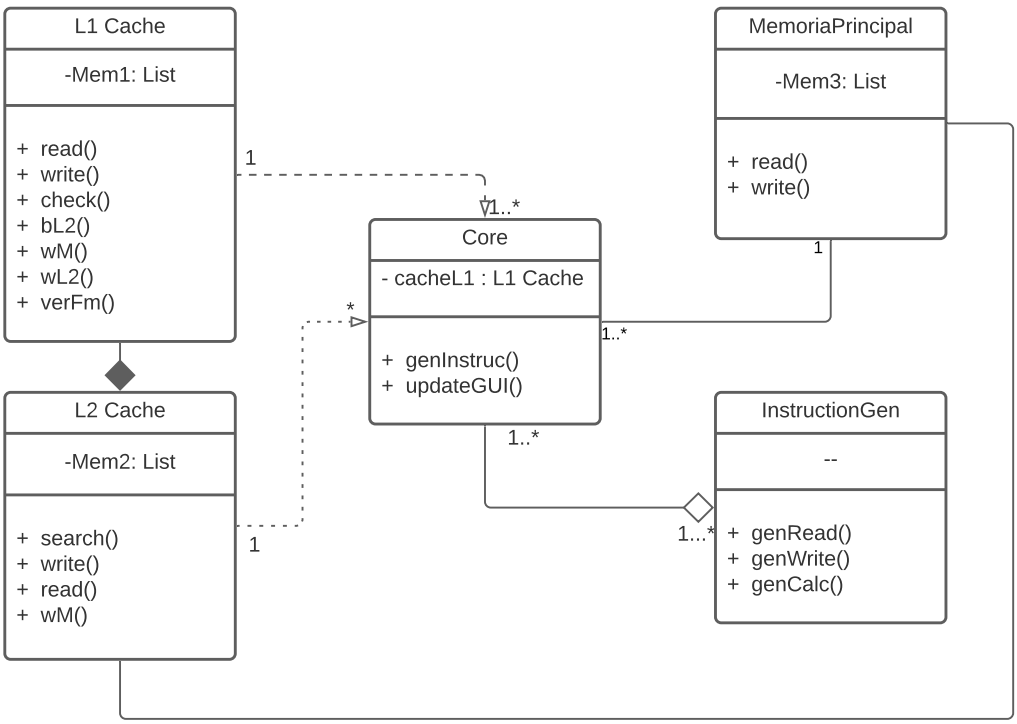

Figura 4: Diagrama de clases.

El diagrama de clases del sistema es bastante corto, ya que las clases solo son las necesarias para el funcionamiento del sistem.
Se divide las siguientes funcionalidades.

- Core: Se encarga de analizar directamente las instrucciones generadas y monitorear la memoria L1 principalmente pero teniendo entradas de memorias L2 y principal.
- L1 cache: Totalmente encargado de almacenar los datos mas recientes e interactuar con memoria principal y L2. El método **verFm** se encarga de actualizar la memoria L1 de acuerdo con los datos que haya en L2,
- L2: Solo guarda los datos usados o leídos por todos los procesadores.
- Memoria principal: Encargada de realizar el almacenamiento de los datos principales.
- Generador de instrucciones: Genera las instrucciones para cada procesador de manera independiente para cada uno.

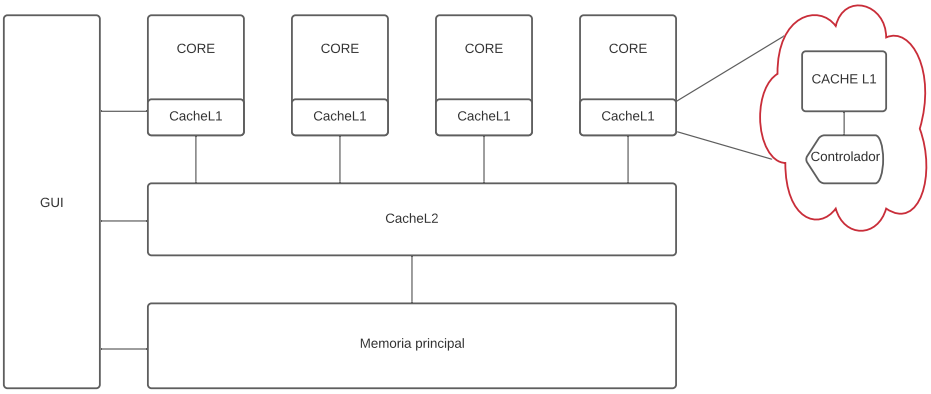

Figura 5: Diagrama de arquitectura.

Se elige esta solución ya que se logra ver que cada core principal tiene su propia memoria cache L1, se ve en la parte derecha que esta en su interior tiene un controlador propio, el cual es el encargado de estar actualizando la memoria antes de cada operación.
El sistema de cache L1 se encarga de escribir a L2 y Memoria principal si es necesario, por lo que en general es la mente principal de todo el programa, cada vez que se realiza una acción en la memoria por parte del procesador, menos por *CALC*, L1 tiene ese método el cual es el encargado de consultar a L2 los estados de las direcciones de memoria, así como también los estados y propietarios, con esto podemos ver si el dato en L1 es o no valido para la lectura, o bien es valido y puede ser leído desde L2. En caso de que no sea valido este deberá ir a la memoria principal y actualizar.
La memoria principal no tienen nada mas direcciones y datos.
El sistema al inicio tiene todo invalido y en blanco los valores en memoria, esto para forzar a escribir o a leer directamente a memoria.
Los valores en L2 son actualizados con lecturas o con escrituras a la memoria principal.