An N Way Associative Cache with M Lines that have P Bytes per line and Z Bytes of Storage with a Y bit long address

The entire point of the Ways in an associative cache is to gain the ability to store multiple lines of data that all have the same Cache Index, with each line with the same Cache Index being stored as part of the same Set.

There are N Lines per Set meaning there are S=M/N=Z/(N\*P) Sets

Number of Index bits are an Associative Cache is based on the number of Sets, rather than the number of Lines like it is in a Direct Cache. The number of Index bits is log2(S). These are also known as set bits. These are used to determine which set the data from an address goes into. Each line of Data within a set have the same Index bits.

Number of Byte Offset in the address is address are log2(P)

The number of Cache Tag bits in the address is X=Y- log2(S)- log2(P) and those are the MSB of the address. Each Line of cache has X additional bits for storing the Tag bits for the Line

Each line has an additional log2(N) bits that are the LRU bits.

There is a total of S\*N\*log2(N)= M\*log2(N) LRU bits for the cache.

These bits are used to keep track of how old the data on each line is. Each line has log2(N) LRU bits because there are N possible order positions for it to have.

Each line has an additional 2 bits that are the MESI bits.

There is a total of S\*N\*2= M\*2 MESI bits for the cache.

**Write Through Cache:** Write Command causes data to be written to both cache and to Memory

**Write Back Cache:** Write command causes data to be written only to cache. Writing from Cache to Memory is then triggered by other events, like an event that would cause the edited cache line to be replaced by a different cache line.

NOTES AND PROBLEMS

We are not yet sure what controls the value of the share bus.

MEANING OF Acronyms and Enums

The Most Recent Used code (MRUc) will be the largest possible value of the LRU bits.

The Least Recent Used code (LRUc) will be the largest possible value of the LRU bits.

0 read data request to L1 data cache

1 write data request to L1 data cache

2 instruction fetch (a read request to L1 instruction cache)

3 invalidate command from L2

This represents snooping the other L1 caches and seeing that they have modified an address

We shall call this SNOOP\_WRITE

4 data request from L2 (in response to snoop)

This represents snooping the other L1 caches and seeing that they are trying to reach an address that you may have modified

We shall call this SNOOP\_READ

8 clear the cache and reset all state (and statistics)

9 print contents and state of the cache (allow subsequent trace activity)

Main

Read command

Case (Command)

(1) WRITE()

(2) READ()

(3) SNOOP\_WRITE()

(4) SNOOP\_READ()

SNOOP\_WRITE

If Address X is in Cache:

If MESI bits for Address X are set to M

Send retry Signal.

Write Data on Line for Address X to Main Memory

//Other Cache will then Read that data from Main Memory

Set MESI bits for Line of Address X to I

If MESI bits for Line of Address X are set to E

Set MESI bits for Line of address X to I

SNOOP\_READ

If Address X is in Cache:

If MESI bits for Address X are set to M

Send retry Signal.

Write Data on Line for Address X to Main Memory

Set MESI bits for Line of Address X to S

If MESI bits for Line of Address X are set to E

Set MESI bits for Line of address X to S

READ

Processor sends Cache using LRU a Read Command for Address X.

Use Cache Index X1 bits to find corresponding Cache Set XA.

If Cache Tag X2 is in Set XA of Cache and the MESI bits are set to E or S or M

\*Cache Hit NO CPU STALL\*

Set LRU bits for the filled line to MRUc from previous value X3.

For Each other Line who LRU bits were farther from LRUc than X3.

Increment the LRU bits towards LRUc by 1.

Deliver Data from Cache to CPU.

Else

Stall CPU

Read Data for Line of Address X from lower Cache or Memory.

If there is a Line with MESI bits set to I in Set XA

Select Line with MESI bits set to I in Set XA

Else

Select Line with LRU value at LRUc in Set XA

Cast out Selected Line on that cache line as Victim

Write retrieved Data to Selected Line.

Write Cache Tag X2 to that Line.

Set LRU bits for the used line to MRUc.

Increment the LRU bits for all other lines towards LRUc by 1.

If Shared Bus is active

Set MESI bits to S

Else

Set MESI bits to E

READ OUT NOTIFICATION OF READ TO OTHER CACHS

Deliver Data from Cache to CPU.

WRITE

Processor sends Cache using LRU a Write V Command for Address X

Use Cache Index X1 bits to find corresponding Cache Set XA.

If Cache Tag X2 is in Set XA of Cache and the MESI bits are set to E or S or M

Select Line with Cache Tag X2 in Set XA

Write V to Selected Line

Set LRU bits for Selected Line to MRUc from previous value X3.

For Each other Line who LRU bits were farther from LRUc than X3.

Increment the LRU bits towards LRUc by 1

IF MESI bits of Selected line are M or E

Set MESI bits of Selected Line to M

IF MESI bits of Selected line are S

Write Data on Line for Address X to Main Memory

Set MESI bits of Selected Line to E

Send out Signal telling other Caches their values for Address X is invalid,

Else

Read Data for Line of Address X from lower Cache or Memory.

If there is a Line with MESI bits set to I in Set XA

Select Line with MESI bits set to I in Set XA

Else

Select Line with LRU value at LRUc in Set XA

Cast out Selected Line on that cache line as Victim

Write retrieved Data to Selected Line.

Write Cache Tag X2 to that Line.

Write Data V to Select Line

Set LRU bits for the used line to MRUc.

Increment the LRU bits for all other lines towards LRUc by 1.

If Shared Bus is active

Set MESI bits to E

Else

Set MESI bits to M