In the name of God

Computer Architecture Project’s Report
Cache Implementation

Nima Tavassoli - Summer 1396

The goal of the project is to design and implement a ram-cache memory for SAYEH basic computer.
Requested cache module is a 2-way set associative cache with 64 rows and 16-bit word.
It has valid bit , therefore it has write back method , we can store data's with the same index and different tags .

List of modules :

Data Array

Miss-Hit Logic

Most Recently Array

Tag-Valid Array

Data Selection

Cache Controller

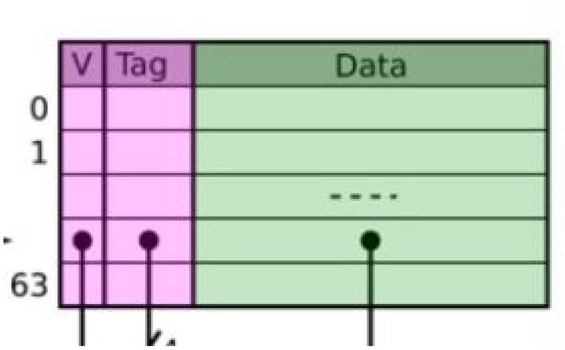
Main Cache

Top Memory Cache

Data Array :

According to cache system in this project we have needed 2 way cache, then i created two modules for data array ,to store section of data .
An array with 64 rows and has space for 16 bit to store .
We can load or write data (the case that write enable is active = 1) to this module with specific input address .

(The green part of this picture)



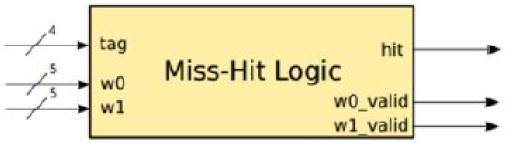
Tag-Valid Array :

Each data in a way of cache has specific index , tag and valid bit .
This module is responsible for storing tag bits with valid bit of each row (5 bits data).
It has invalidate bit as an input : if invalidate = 1 , valid bit in that row became 0 .
Valid-bit 0 means , the cell of cache is empty and we can write data to that place or the data in that cell has not updated .
I created 2 modules of Tag Valid Array for 2 ways like data array module
And we have write enable and address inputs too.
We store 2 datas that have same index with different tags that we need section of (9 to 6) of address to store tag.

It has 64 row of 5 bit , size (64 * 5).

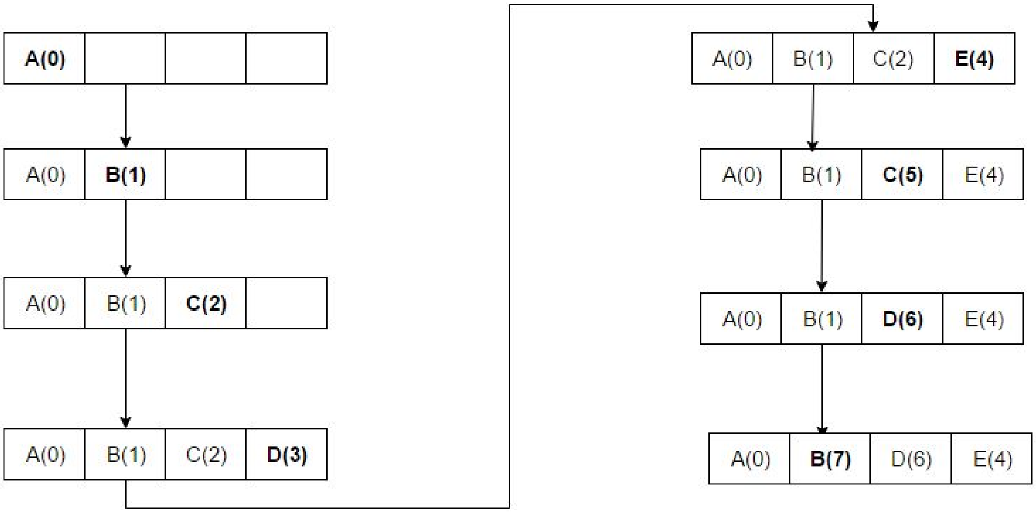
Miss Hit Logic :

This module is the core Logic of the cache and decides the accommodation .
As the below block diagram demonstrates , i created a module with a tag input that we want to search and two 5 bits (tag-valid bits of 2 ways) , we set hit ‘1’ when input tag = each of way’s tag.
(That means data we search for is find in the cache).
We have 2 other outputs w0_valid , w1_valid we set each of them ‘1’ when that data stored in specific way and the valid bit equals 1.
With miss-hit logic module we can find out that the data stored in cache or not stored until now.
Then we should bring the data to the cache from memory that other modules do the replacement.



Most Recently Used Array :

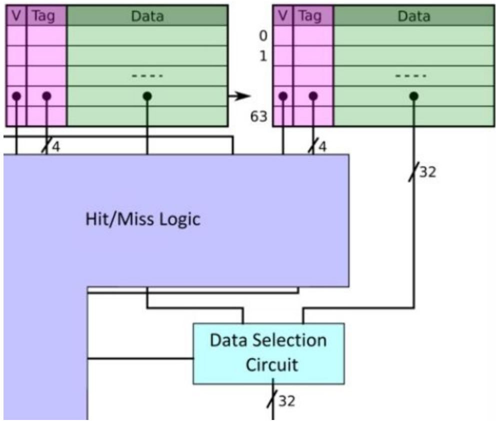
We bring data from memory to cache when miss = 1 , if one of ways on the specific address (index) was empty we place data to the empty cell , else if both of the ways are empty i store the data in the way-0 and we use MRU method for the state that we don’t have empty space at index .
MRU selects the way that we refer to recently.
I found out the method from this wikipedia example :
The access sequence for the below example is A B C D E C D B.



Explain of Implementation :
I have “mruwaybitprevios” as an input to calculate the number of references .
For calculating , i used an 64 bit array of integer and when dataReady signal = ‘1’ increment the specific cell of that array .
I have 2 arrays for 2ways.
The output of module “wayoutput” tells us which way is right to store data according to comparison between way-1 counter and way-0 counter.

Data Selection :

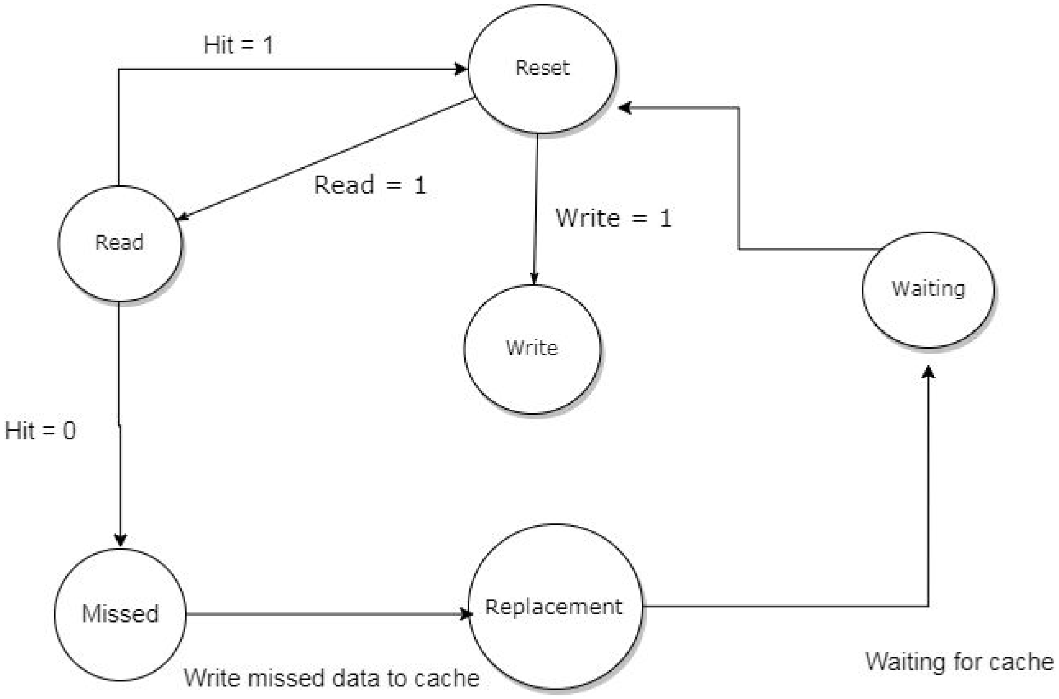
This module selects the cache output by w0_valid and w1_valid parameters from tag valid array.
It choose output data between way1 and way0 data .



Cache Controller :

We need controller set control signals to whole system works properly .
I created a FSM for cache module to transit among the states and do the specific jobs for the states .

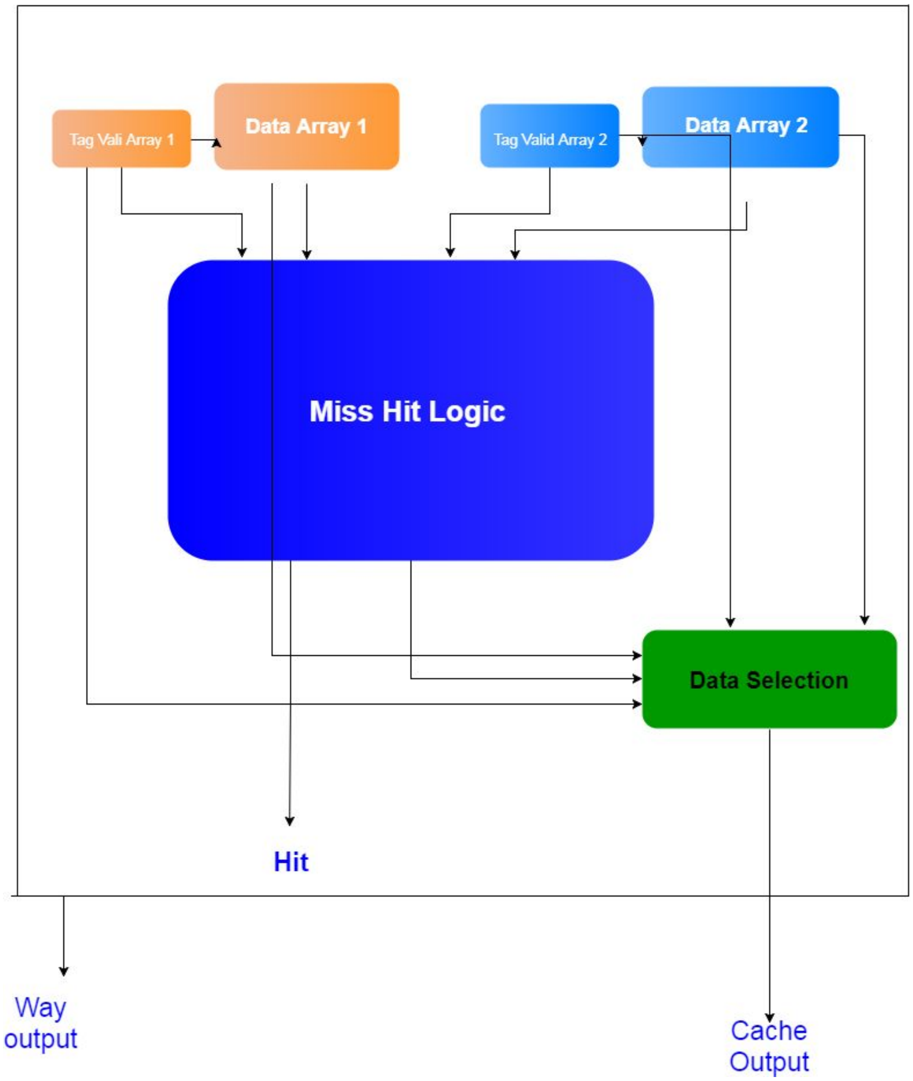
Cache State Machine :



The fsm briefly works this way :
Initial state : reset ,
When read = 1 we go to state read from cache if it was there we get hit= 1 .
Otherwise (hit = 0) we should go to replacement state to bring the data from memory to cache .
Then we should take appropriate time to cache for writing data (waiting state) finally we set the current state = reset .

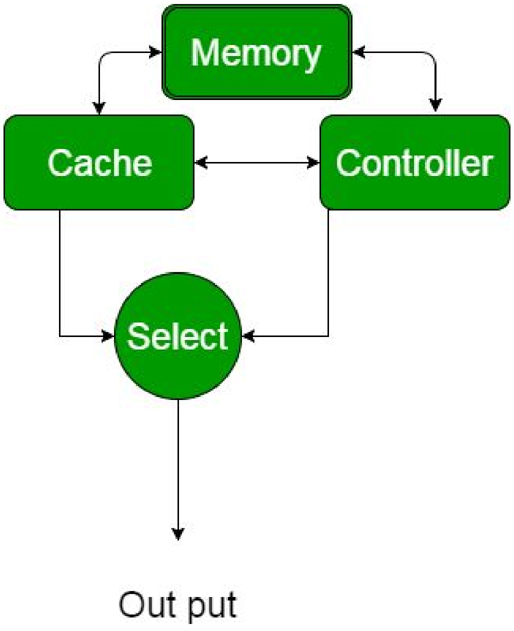
Main Cache :

I just connected the modules together to create a cache with storage and logic .



Top Memory Cache :

We connect cache , controller and memory together and use the system as a cache-ram memory that can store and load data s with the help of cache faster .
We can write to this system or read from memory or cache .


Memory
Cache
Controller
Select
Out put