گزارش پروژه درس معماری کامپیوتر

رشته: مهندسی کامپیوتر

گرایش: سخت افزار

عنوان :

SAYEH CACHE

استاد:

دکتر سعید شیری قیداری

تدریس یاران:

فرزان دهباشی – پرهام الوانی

نگارش:

علی ایزدی

تیرماه ۱۳۹۶

چکیده:

هدف از این پروژه طراحی کش برای کامپیوتر پایه سایه است.
ماژولی که بین حافظه اصلی و cpu قرار میگیرد تا سرعت دسترسی به داده و دستور العمل را افزایش دهد.
پیاده سازی کش با استفاده از زبان vhdl انجام شده و تست های مختلفی به تنهایی و متصل به سایه روی آن انجام گرفته است.

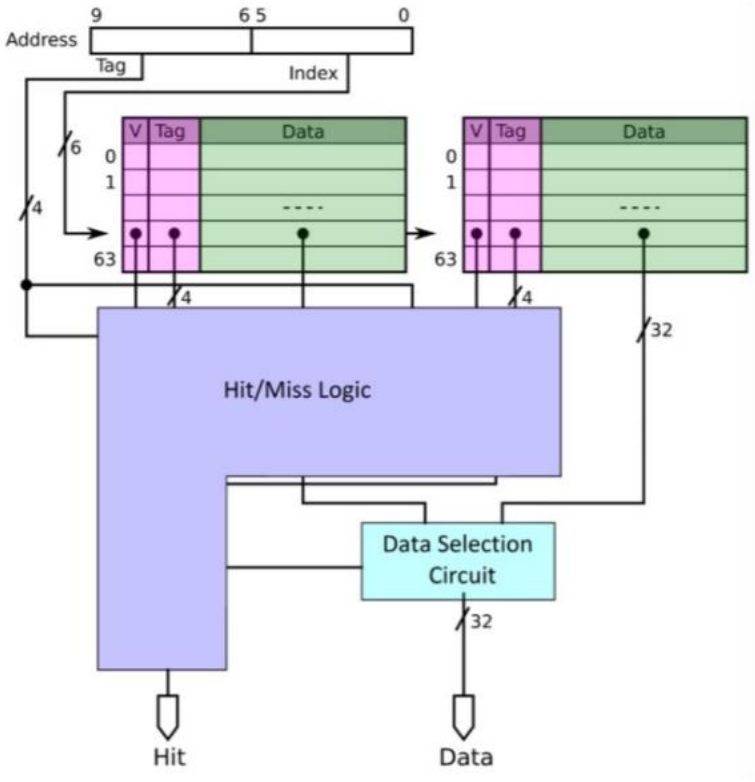
ماژول ها :

ماژول های cache عبارتند از:

۱- Data Array
۲- Tag valid array
۳- Most recently used array
۴- Miss hit logic
۵- Cache
۶- Memory
۷- Ram(memory-cache)

این نوع کش از نوع 2-way set associative است.

ساختار کش طراحی شده مطابق شکل زیر است:



ماژول Data array :

این ماژول برای نگه داری داده ها است که شامل یک way ۶۴ تایی میباشد.

هر داده دارای طول ۱۶ بیت است.

سیگنال آدرس مشخص کننده ردیفی است که داده باید در آن نوشته شود.

ماژول tag valid array :

این ماژول برای نگه داری ۴ بیت تگ که ۴ بیت آخر آدرس هستند و هم چنین برای نگه داری بیت valid استفاده میشود.

بیت valid ابتدای کار و زمانی که داده ای در memory نوشته میشود چون آن داده دیگر در کش اعتبار ندارد invalid یا صفر میشود.

ماژول most recently used array :

از این ماژول زمانی استفاده میشود که هر دو way یک ردیف data array پر شده باشد و داده ای بخواهد در یکی از این دو way ها نوشته شود، آن گاه این ماژول در خروجی خود شماره way که باید داده ی جدید در آن جایگزین شود را به data path میدهد.

در کش طراحی شده از policy زیر استفاده شده است:

داده ای که تاکنون بیشترین بار استفاده شده است را با داده جدید جایگزین میکنیم.

برای این کار از یک شمارنده برای هر way مانند data array استفاده میکنیم و

هر زمانی که داده ای در کش hit شد خانه مربوط به آن را یک واحد افزایش میدهیم.

و هر زمانی که داده ی جدیدی در کش نوشته میشود خانه شمارنده مربوط به آن ریست میشود.

ماژول miss hit logic :

این ماژول برای مقایسه تگ آدرس جدید با تگ داده های موجود در کش استفاده میشود.

سه ورودی آن عبارتند از تگ آدرس داده شده و و تگ های ذخیره شده در ۲ way با index آدرس داده شده.

این سه ورودی با هم مقایسه میشوند و خروجی های hit , w0, w1 را تولید میکنند تا مشخص کنند

اولاً داده hit شده است و در کش وجود دارد یا خیر

دوماً اگر hit شده است با کدام یک از way ها در index فعلی hit خورده است.

ماژول cache :

این ماژول به عنوان data path برای وصل کردن ماژول های ذکر شده در بالا استفاده میشود.

از هر کدام از ماژول های data array و tag valid array دو بار instantiate میشود زیرا کش مورد نظر 2 way set associative میباشد.

سیگنال های enable این چهار ماژول نیز به صورت gate level در همین ماژول مشخص میشوند.

کد data path ماژول کش به طور خلاصه برای فهم بهتر در زیر آورده شده است.

```
invalidate0 <= w0_valid and write;
invalidate1 <= w1_valid and write;

tagWren0 <= wren0;
tagWren1 <= wren1;

wren0 <= (not tagValid0(4) and write_to_cache) or (not ouput_MRU and write_to_cache and ready_MRU);
wren1 <= ((not tagValid1(4) and write_to_cache and tagValid0(4)) or ( ouput_MRU and write_to_cache and ready_MRU)) and (not tagWren0);


dataArray0 : dataArray port map (clk, address(5 downto 0), wren0, wrdata, data0);
dataArray1 : dataArray port map (clk, address(5 downto 0), wren1, wrdata, data1);
-----------------
tagValidArray0 : tagValidArray port map (clk, reset_n, address(5 downto 0), tagWren0, invalidate0, address(9 downto 6), tagValid0);
tagValidArray1 : tagValidArray port map (clk, reset_n, address(5 downto 0), tagWren1, invalidate1, address(9 downto 6), tagValid1);
-----------------
miss_hit_logic : missHitLogic port map (address(9 downto 6), tagValid0, tagValid1, hit_temp, w0_valid => w0_valid, w1_valid => w1_valid );
hit <= hit_temp;
-----------------
MRU : MRU_array port map (address(5 downto 0), write_to_cache, clk,  output => ouput_MRU, ready => ready_MRU);
-----------------
muxOut : mux port map (w1_valid, data0, data1, outdata);
-----------------
```

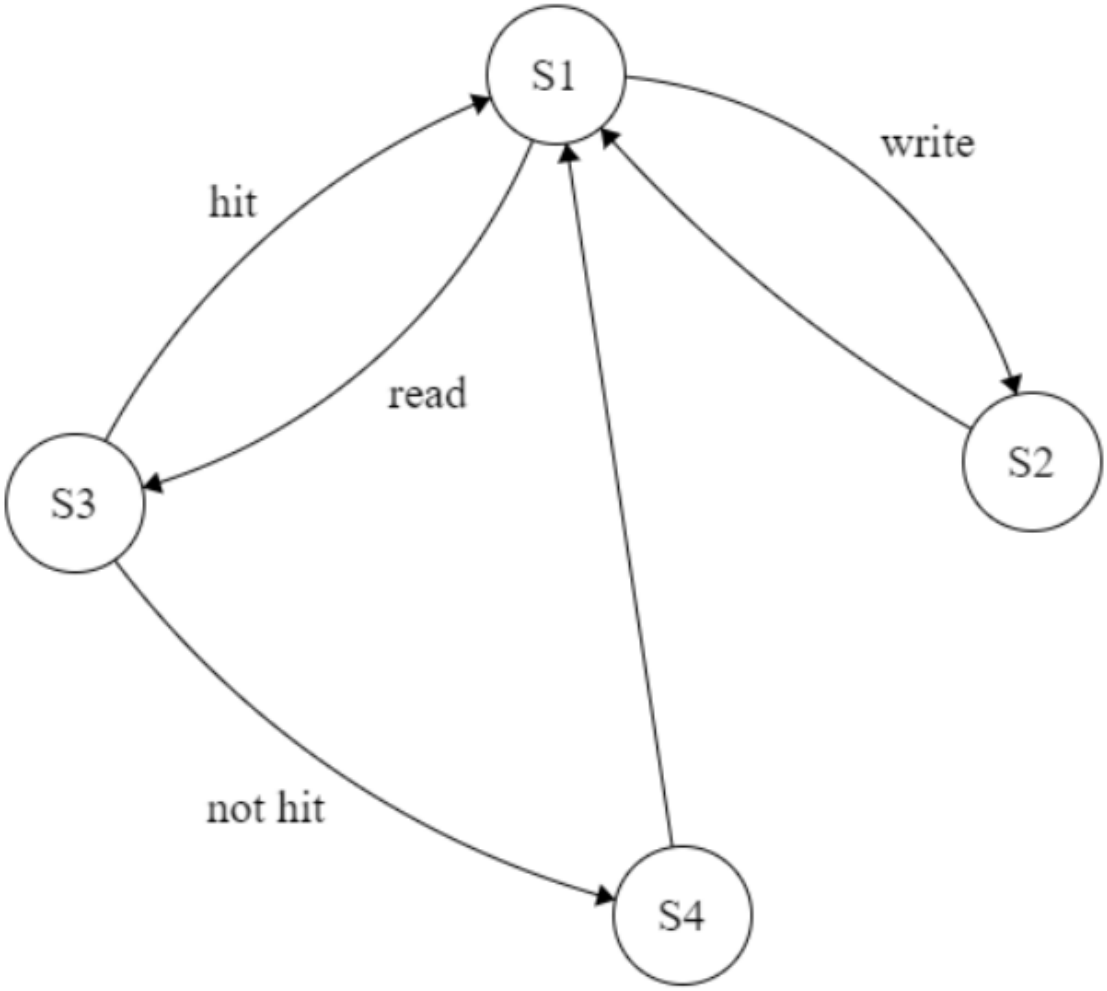
ماژول ram:

این ماژول به عنوان ماژول جایگزین مموری sayeh قرار میگیرد که در آن دو ماژول مموری اصلی و cache به هم متصل میشوند.

```
mem : memory port map (clk, writemem, address, ram_Data_in ,ram_Data_out, memdataready);
ch : cache port map (clk, write_to_cache_temp, write_temp, reset_n, address,  cache_wrdata, outdata, hit_temp);
```

کنترلر کش نیز در این ماژول قرار دارد.

Fsm طراحی شده مطابق شکل صفحه بعد است.



initial : S0

writemem<='1'; ram_Data_in<=indata; : S1

nothing to be done : S2

write_to_cache <='1'; cache_wrdata<=ram_Data_out; : S4