



2022

COMPUTER  
ARCHITECTURE  
PHASE 2

---

PROF.SARBAZI AZAD

MEMBERS:

HOSSEIN GOLI

ERFAN SADRAIYE

AMIRHOSSEIN BARATI

ALIREZA FOROODNIA

در این فاز همانگونه که خواسته شده بود یک واحد حافظه نهان در پردازنده تعبیه شد که به کمک آن پردازش هایی که دیتای آن درون این حافظه موجود هستند مانند گذشته در یک کلاک انجام می شوند و سایر پردازش ها ابتدا یک درخواست به حافظه اصلی برای داده هایشان می دهند و پس از رسیدن این داده ها به حافظه نهان ادامه فعالیت خود را پیش می گیرند.

## کلیات پیاده سازی

برای پیاده سازی موارد بالا ابتدا یک ماژول **Cache** (حافظه نهان) طراحی شد و سپس این ماژول به همراه ماژول حافظه ای که در فاز های قبلی نیز وجود داشت (و در این فاز تاخیر به آن افزوده شده بود) در یک ماژول **mem\_dp** تجمیع شد و برای کنترل این مجموعه نیز یک ماژول **cache\_CU (cache control unit)** به مجموعه اضافه شد که مسئولیت کنترل تاخیرهای حافظه و هماهنگ سازی واحد های **cache** و **memory** را بر عهده دارد. جزئیات پیاده سازی به صورت ماژول به ماژول در صفحات بعد شرح داده شده است.

## ماژول Cache :

این ماژول با استفاده از روش **direct mapping** , شیوه انتقال داده از نوع **write back** و در قالب بلوک های یک کلمه ای با حجم کلی 8kb (شامل  $2^{11}$  کلمه 32 بیتی) طراحی شده است.

این ماژول دارای ترمینال لیستی به شکل رو به رو است :

```
module cache
    data_out,
    hit,
    cache_miss_addr,
    dirty_bit,
    we,
    clk,
    rst_b,
    addr,
    is_byte,
    data_in
endmodule;
```

پورت های **clk** , **data\_out**, **data\_in**, **addr**(address), **we**(write enable), **reset\_b**(reset bar) همان مسئولیت هایی را بر عهده دارند که کمابیش در هر واحد حافظه ای موجود است.

سایر پورت ها مسئولیت های زیر را بر عهده دارند :

**is\_byte** : این پورت از ورودی های ماژول است که مشخص می کند فرمان خواندن یا نوشتنی که به حافظه نهان داده شده است از نوع **Byte** است یا **word**. مقدار این پورت توسط واحد **Cache\_cu** تعیین می شود.

**Hit** : همانگونه که از نام این پورت پیداست مسئولیت اعلام شرایط درخواست حافظه داده شده به حافظه نهان را بر عهده دارد. مقدار این بیت از حافظه نهان به سمت **Cache\_cu** خارج می شود تا مراحل انتقال داده از حافظه اصلی به حافظه نهان آغاز شود (این مراحل در ادامه در توضیحات ماژول **Cache\_cu** به طور کامل شرح داده می شوند).

**Dirty\_bit** : این پورت نیز مانند **hit** مسئولیت اعلام وضعیت خانه ای که درخواست دسترسی به آن داده شده است را دارد. خروجی این پورت نیز به **Cache\_cu** منتقل می شود تا در صورتی که این مقدار صفر باشد مرحله نوشتن در حافظه اصلی صورت نگیرد.

**Cache\_miss\_address** : این پورت در صورت رخ دادن یک **miss** در حافظه نهان آدرسی را تولید می کند که باید محتویات فعلی حافظه نهان به آن منتقل شوند تا مقادیر جدید با آنها جایگزین شوند.

کلیت شیوه کارکرد این واحد حافظه نهان دقیقاً مطابق با مطالبی است که استاد در بخش حافظه نهان تدریش کردند فلذا نیازی به توضیح آن احساس نمی شود.

## ماژول Cache\_cu :

همانطور که در ابتدا گفته شد این ماژول مسئولیت کنترل و مدیریت داده ها را در واحد مموری (که در ادامه شرح داده می شود) بر عهده دارد. در این ماژول به کمک سیگنالهایی که از حافظه نهان گرفته می شوند و همچنین دستوراتی که به ماشین داده می شود، عملیات انتقال داده به رجیستر ها و یا انتقال در میان حافظه اصلی و حافظه نهان را زمان بندی می کند.

ترمینال لیست این ماژول به شکل زیر است :

```
module cache_cu (  
    dirty,  
    cache_we,  
    mem_we,  
    mem_in_select,  
    clk,  
    rst_b,  
    hit,  
    opcode,  
    reg_write_enable,  
    cache_in_select  
);
```

در میان های بالا پورت های **clk**, **rst\_b** مسئولیت های عادی خود را بر عهده دارند.

پورت های **dirty**, **hit**, **opcode** به عنوان سیگنال های وضعیت به این ماژول وارد می شوند تا نوع عملیاتی را که این واحد باید مدیریت کند را مشخص کنند.

پورت های **cache\_we**, **mem\_we**, **mem\_in\_select**, **reg\_write\_enable**, **cache\_in\_select** پورت های کنترلی هستند که به کمک آنها این ماژول عملیات های مربوط با انتقال داده را انجام می دهد.

نحوه کلی عملکرد :

این ماژول هنگامی که دستور های **lw**, **sw**, **lb**, **sb** را تشخیص دهد شروع به عملیات انتقال داده می کند. با ارسال ادرس به حافظه نهان در صورت وقوع یک **hit** مقادیر را از حافظه نهان به خروجی هدایت می کند. در صورت وقوع یک **miss** ابتدا مقدار بیت **dirty** (که در کش توضیح داده شد) را چک می کند. یک بودن این بیت به معنی تغییر کردن داده در زمان حضورش در حافظه نهان است، فلذا این مقدار باید در حافظه اصلی به روز رسانی شود که این عملیات پنج کلاک به طول می انجامد و پس از آن مقدار اصلی طی چهار کلاک به حافظه نهان وارد شده و به خطوط خروجی حافظه هدایت می شود.

مکانیزم ایجاد تاخیر در مموری نیز به کمک یک شمارنده و یک نمودار **fsm** ایجاد شده تا واحد حافظه نظر به اینکه در چه شرایطی قرار دارد بتواند به میزان مورد نیاز حافظه اصلی تاخیر ایجاد کند تا داده های مورد نیاز همگی برای خواندن/نوشتن آماده شوند.

## ماژول mem\_dp :

این ماژول طبق طرح ارائه شده در زیر طراحی شده است که شامل یک واحد از حافظه نهان است و با واحد حافظه اصلی در mips\_machine نیز در ارتباط است و به کمک های خروجی اش وضعیت حافظه را به Cache\_cu اطلاع می دهد و پورت های ورودی اش را از Cache\_cu دریافت می کند (به جز پورت address که مستقیماً از control unit وارد می شود).

وظایف این واحد به شرح زیر است :

۱. نوشتن در کش با استفاده از دیتا ورودی / خروجی مموری
۲. نوشتن در مموری با استفاده از خروجی کش
۳. هدایت خروجی سیگنال های کترلی hit-dirty برای استفاده cache controller

