پروژه درس ساختار کامپیوتر و میکروپروسسور

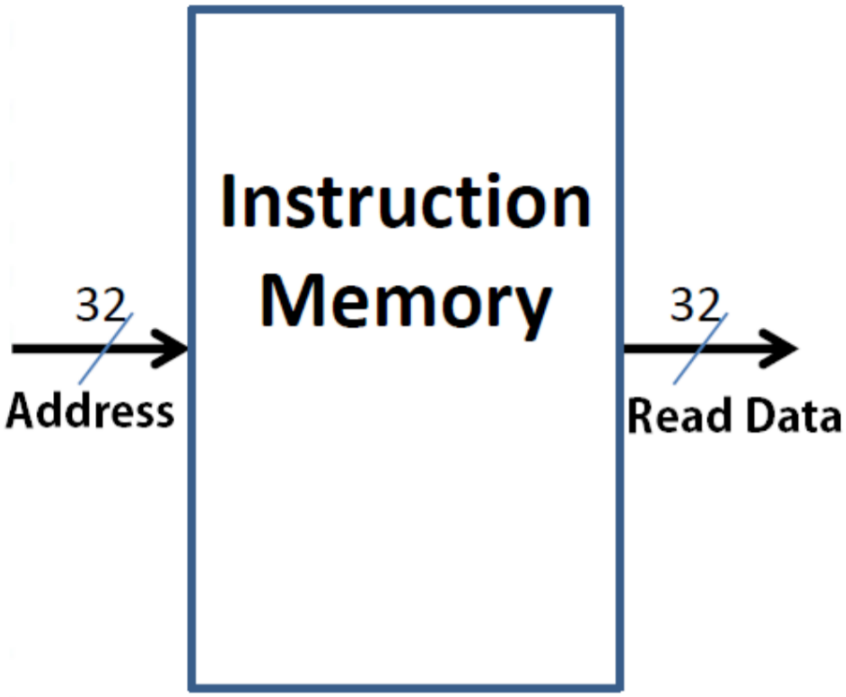
دکتر موحدیان

فاز اول

پیاده سازی ماژول های المان‌های حالت برای طراحی پردازنده ها

قسمت اول – پیاده سازی Instruction Memory Module

هدف در این قسمت، پیاده سازی ماژول Memory Instruction است.



ماژول مذکور باید شامل یک ورودی باشد:

ورودی 32 بیتی برای آدرس PC (Address)

ماژول مذکور باید شامل یک خروجی باشد:

خروجی 32 بیتی مربوط به دستور موجود در رجیستر با آدرس PC موجود در Instruction Memory (Read_Data)

نکته: به دلیل نبود حافظه فیزیکی برای Memory Instruction ،حافظه مذکور در یک فایل با نام memfile.dat در اختیار شما قرار گرفته است.

توضیحات پیرامون فایل memfile.dat

این فایل شامل تعدادی عدد به فرمت HEX در هر خط می باشد که معادل با دستور های 32 بیتی مذکور برای پردازنده ما می باشند.

نحوه خواندن داده ها از فایل memfile.dat

برای خواندن داده های موجود در فایلmemdfile.dat و ذخیره آن ها در یک رجیستر برای ارتباط با ماژول Instruction Memory همانند زیر عمل می کنیم:

ابتدا یکرجیستر 2 بعدی برای حافظه مورد نظر در ماژول خود به شکل زیر تعریف کنید:

reg [31:0] RAM [1023:0];

به این معنا که حافظه RAM مذکور شامل 1024 خانه حافظه است که هر کدام محتوی یک رجیستر 32 بیتی هستند. در مجموع یک حافظه 4 کیلوبایتی خواهیم داشت.

سپس به کمک دستور زیر:

initial

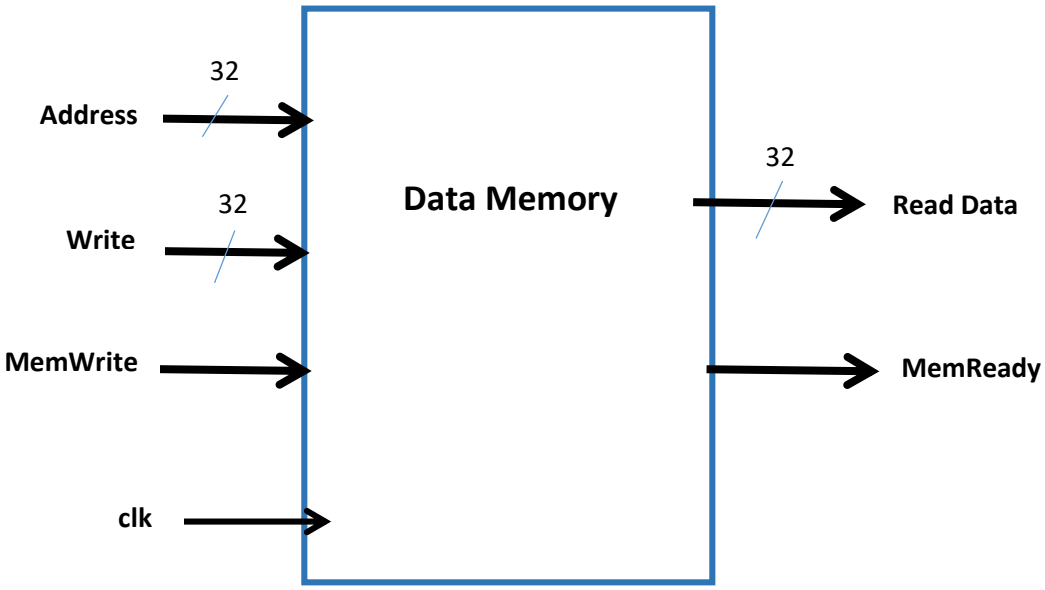
$readmemh (“memfile.dat”, RAM);

محتوای موجود در فایل memfile.dat در رجیستر 2 بعدی RAM به این صورت قرار می گیرد که از اولین خط از فایل مربوطه تا آخرین خط (خط n ام)، داده های HEX به ترتیب در رجیستر های [0]RAM تا [n-1]RAM که هر کدام 32 بیت هستند قرار می گیرند.

هدف طراحی ماژولی با ورودی و خروجی های تعریف شده به شکل بالا است که با دریافت یک آدرس 32 بیتی به عنوان ورودی (Address) رجیستر 32 بیتی خروجی را با مقدار RAM[Address] مقدار دهی کند.

قسمت دوم – پیاده سازی Data Memory

هدف در این قسمت، پیاده سازی ماژول Data Memory است.



ماژول مذکور باید شامل چهار ورودی باشد:

1. ورودی یک بیتی برای کلاک (**clk**)

2. ورودی یک بیتی برای بیت کنترلی MemWrite حافظه (**MemWrite**)

3. ورودی 32 بیتی برای آدرس خانه حافظه (**Address**)

4. ورودی 32 بیتی مربوط به داده مورد نظر برای ذخیره در رجیستر با آدرس ورودی موجود در Data Memory (**WriteData**)

ماژول مذکور باید شامل دو خروجی باشد:

خروجی 32 بیتی مربوط به داده موجود در رجیستر با آدرس ورودی موجود در Data Memory (**ReadData**)

خروجی یک بیتی بیانگر آماده بودن خروجی برای خواندن (MemReady)

هدف طراحی ماژولی با ورودی و خروجی های ذکر شده و یک حافظه RAM دو بعدی با 1024 خانه 32 بیتی می باشد که با دریافت یک آدرس 32 بیتی(word aligned) به عنوان ورودی (Address)، همواره مقدار موجود در خانه Address را برای خواندن در رجیستر خروجی قرار می دهد. (مدار ترکیبی)

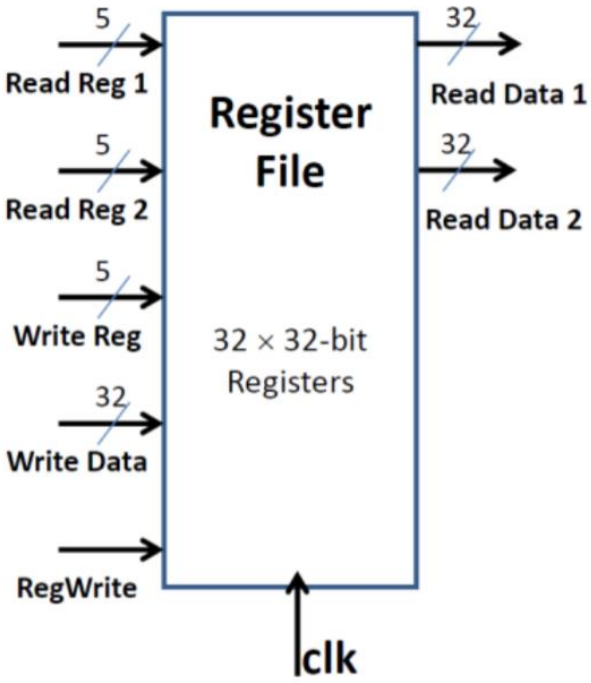
در واقعیت حافظه ی RAM به این صورت ایده آل نیست و خواندن از حافظه می‌تواند با مقداری تاخیر همراه باشد و حافظه تا کامل شدن عملیات خواندن به چند سیکل کلاک زمان نیاز داشته باشد. به همین دلیل نیز برای تسریع پردازش از حافظه دیگری به نام Cache استفاده می‌شود که بسیار سریع است. در این جا می-خواهیم این تاخیر را به طور ساده شبیه سازی کنیم تا در آینده که پردازنده را کامل می‌کنیم تاثیر آن را روی عملکرد الگوریتم‌های مختلف ببینیم.

برای این کار یک پارامتر MEM_DELAY در کد خود تعریف کنید که بیانگر تعداد کلاک لازم برای خواندن می‌باشد. بیت خروجی MemReady بیانگر طی شدن این زمان خواهد بود. زمانی که مقدار جدیدی در Address می‌آید در ابتدا خروجی Read Data قابل استفاده نبوده و مقدار MemReady صفر خواهد بود. پس از MEM_DELAY کلاک مقدار MemReady یک شده و خروجی حافظه در دسترس پردازنده قرار می‌گیرد. طراحی شما باید به گونه ای باشد که در صورت صفر قرار دادن مقدار MEM_DELAY همان مدار ترکیبی کار کند و در همان کلاک خروجی آماده باشد.

برای نوشتن در حافظه نیز ماژول به این صورت خواهد بود که در صورت فعال بودن بیت MemWrite ،در انتهای کلاک بعد مقدار موجود در ورودی (DataWrite) را در خانه حافظهRAM با آدرس 32 بیتی Addressقرار می دهد.

قسمت سوم – پیاده سازی Register File

هدف در این قسمت، پیاده سازی ماژول Register File است.



ماژول مذکور باید شامل شش ورودی باشد:

1. ورودی یک بیتی برای کلاک (**clk**)
2. ورودی یک بیتی برای بیت کنترلی Write Enable رجیستر فایل(**RegWrite**)
3. ورودی پنج بیتی برای اشاره به آدرس رجیستر اول (**Read Reg 1**)
4. ورودی پنج بیتی برای اشاره به آدرس رجیستر دوم (**Read Reg 2**)
5. ورودی پنج بیتی برای اشاره به آدرس رجیستر سوم (**Write Reg**)
6. ورودی 32 بیتی مربوط به داده مورد نظر برای ذخیره در رجیستر با آدرس Write Reg موجود در رجیستر فایل (**Write Data**)

ماژول مذکور باید شامل دو خروجی باشد:

1. خروجی 32 بیتی مربوط به داده موجود در رجیستر با آدرس Read Reg 1 موجود در رجیستر فایل (**Read Data 1**)

2. خروجی 32 بیتی مربوط به داده موجود در رجیستر با آدرس Read Reg 2 موجود در رجیستر فایل (**Read Data 2**)

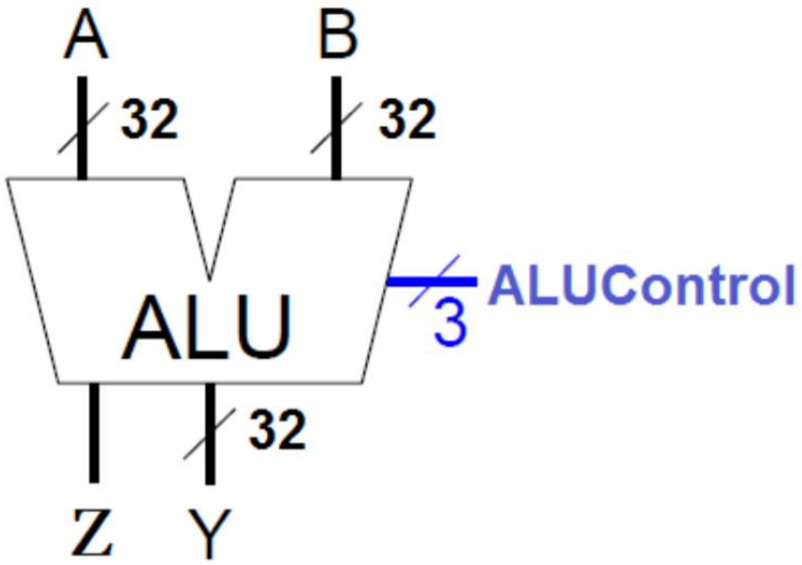
هدف Module :

هدف طراحی ماژولی با ورودی و خروجی های ذکر شده و یک رجیستر فایل دو بعدی با 32 رجیستر 32 بیتی می باشد که با دریافت سه آدرس 5 بیتی به عنوان ورودی، همواره رجیستر ها با آدرس های Read Reg 1 و Read Reg 2 را در خروجی هایRead Data 1 و Read Data 2 قرار می دهد. (این بخش به صورت ترکیبی و بدون نیاز به کلاک پیاده سازی می شود)

و هم چنین، در صورت فعال بودن بیت RegWrite ،در انتهای کالک بعد مقدار موجود در ورودی Write Dataرا در رجیستر با آدرس 5 بیتی Write Reg قرار می دهد.

قسمت چهارم – پیاده سازی ماژول ALU (Arithmetic Logic Unit)

هدف در این قسمت، پیاده سازی ماژول ALU است.



ماژول مذکور باید شامل سه ورودی باشد:

1. ورودی اول 32 بیتی (A)
2. ورودی دوم 32 بیتی (B)
3. سیگنال کنترلی 3 بیتی ورودی (ALU control)

ماژول مذکور باید شامل دو خروجی باشد:

1. خروجی 32 بیتی برای نتیجه ALU (Y.)
2. خروجی یک بیتی برای نشان دادن صفر بودن خروجی (Z.)

هدف Module :

هدف طراحی ماژولی با ورودی و خروجی های ذکر شده می باشد که با دریافت دو رجیستر 32 بیتی به عنوان ورودی، با توجه به سیگنال ALUControl همواره عملیات مورد نظر را که در زیر آمده اند بر روی دو رجیستر انجام داده و نتیجه خروجی را در رجیستر با نام Y قرار می دهد.

نکته: در صورت صفر شدن Y ،خروجی یک بیتی Z فعال می شود.

جدول عملیات های ALU با توجه به 3 بیت [2:0]ALUControl :

Operation	ALUControl[2:0]
ADD	**3'b000**
SUB	**3'b001**
AND	**3'b010**
OR	**3'b011**
XOR	**3'b100**
NOR	**3'b101**
SLT	**3'b110**
SLTU	**3'b111**

خواسته های پروژه:

1- طراحی کد وریلاگ واحد های ذکر شده و کارکرد صحیح این مدارها با تست بنچ هایی که در اختیارتان قرار داده شده است یا تست بنچی که خودتان زده اید.

2- گزارشی کوتاه در مورد نحوه نوشتن واحد ها

نکات مهم پروژه:

1- زمان تحویل حضوری در CW اعلام می شود.

2- هر واحد خود را در فولدری جداگانه و به همراه تست بنچ آن قرار دهید و در غالب فایلی با اسم Modules_Student#.rar در cw آپلود کنید که منظور از student# شماره دانشجویی شما می‌باشد.