**به نام خدا**

گزارشکار فاز اول پروژه درس معماری کامپیوتر

**طراحی و پیاده­سازی Data Path**

استاد

**دکتر حمید سربازی آزاد**

اعضای گروه

**محمدپارسا بشری 400104812**

**محسن قاسمی 400105166**

**امیرحسین رازلیقی 99102423**

بهار 1402

فهرست

[مقدمه و هدف فاز اول 2](#_Toc136006998)

[طراحی Register File 2](#_Toc136006999)

[طراحی Memory 2](#_Toc136007000)

[طراحی ALU 2](#_Toc136007001)

[منابع و مراجع 3](#_Toc136007002)

# مقدمه و هدف فاز اول

هدف کلی این پروژه، طراحی و پیاده­سازی یک پردازنده MIPS است. در فاز اول قصد داریم Datapath و Control Unit این پردازنده را به صورت Single Cycle طراحی و پیاده­سازی کنیم. همچنین برای اطمینان از صحت عملکرد Component های پردازنده به صورت خودکار، تعدادی تست نیز برای ماژول­هایمان می­نویسیم. ماژول­های Register File و Memory از قبل در اختیارمان قرار گرفته بود، بنابراین به توضیح مختصری درباره آن­ها اکتفا می­کنیم.

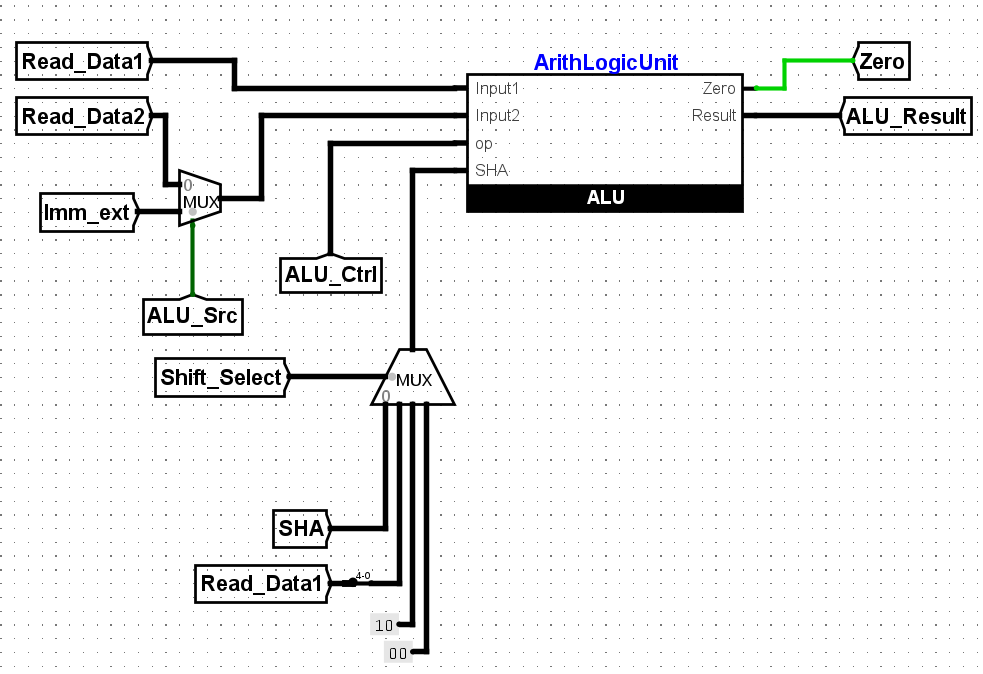
# طراحی Register File

پردازنده MIPS پردازنده­ای 32 بیتی با معماری RISC است که 32 عدد ثبات عمومی دارد. طراحی Register File به سادگی شامل 32 عدد ثبات می­شود که ورودی Write Enable آنها با استفاده از یک دیکودر 5×32 تولید می­شود (ثبات شماره 0 قابل نوشتن نیست و همواره مقدار 0 را نگه می­دارد). دو ورودی 5 بیتی ReadIndex1 و ReadIndex2 نیز به عنوان ورودی select به دو MUX 5×32×32 داده می­شوند تا محتویات ثبات­هایی که باید خوانده شوند را انتخاب کنند.

# طراحی Memory

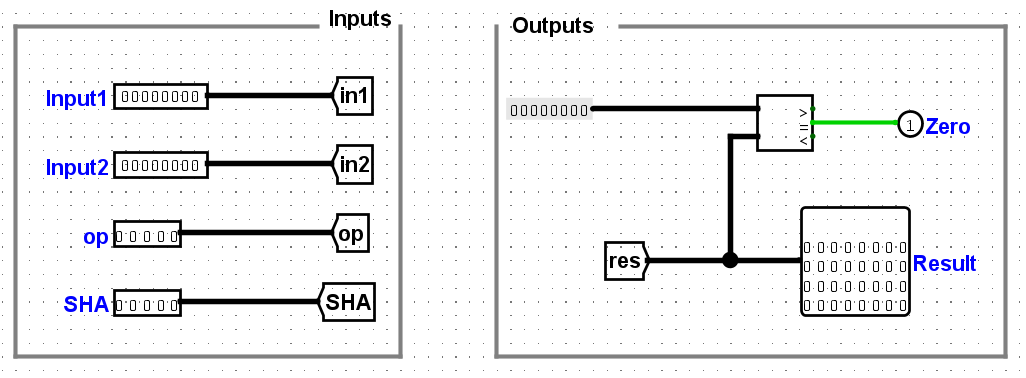
از دو حافظه مجزا برای داده­ها و دستورات استفاده می­کنیم (Data Memory و Instruction Memory) که هر کدام شامل 16K کلمه هستند و از آنجایی که هر کلمه 4 بایت است، هر ماژول حافظه 64KB ظرفیت دارد که با استفاده از 4 عدد RAM 16K×8 و ساختاری شبیه low-order interleaving ساخته می­شود. به این صورت که دو بیت سمت راست را از Address حذف می­کنیم و 14 بیت باقی­مانده را به هر چهار ماژول می­دهیم. سپس برای نوشتن یک (یا چند) بایت خاص داخل یک کلمه، از ورودی Mask استفاده می­کنیم.

# طراحی ALU

واحد محاسبات و منطق (ALU) وظیفه انجام عملیات­های اصلی روی داده­ها را دارد. این واحد دو ورودی دیتا (32 بیتی) دریافت می­کند و با توجه به ورودی operation (5 بیتی)، عملیاتی که باید روی این دو داده انجام شود را انتخاب می­کند. سپس خروجی عملیات انجام شده را از طریق خروجی Result (32 بیتی) اعلام می­کند. همچنین خروجی Zero (1 بیتی) صفر بودن حاصل محاسبه را نشان می­دهد.

شکل 1- نمای بیرونی ALU

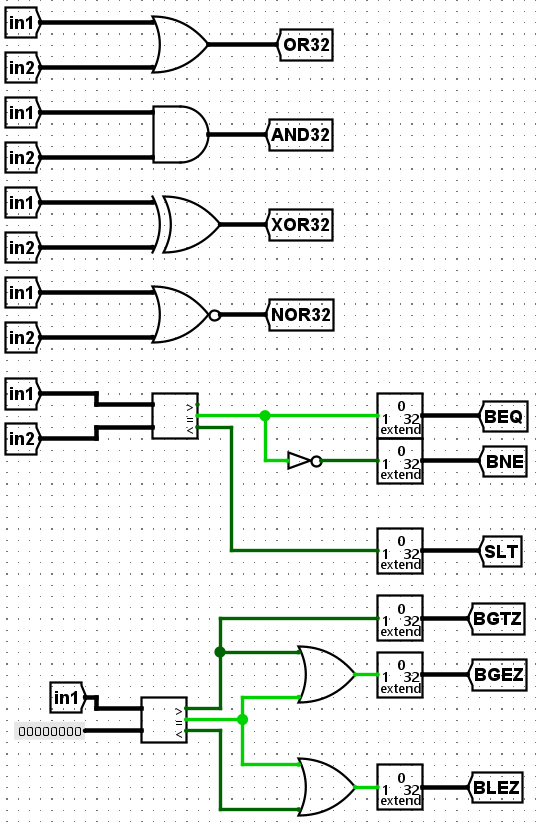
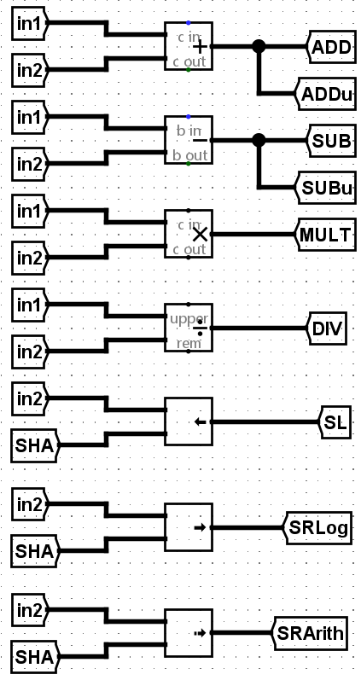
همچنین ورودی SHA مقدار شیفت را مشخص می­کند. توجه کنید که ورودی­های ALU توسط CU تولید می­شوند.

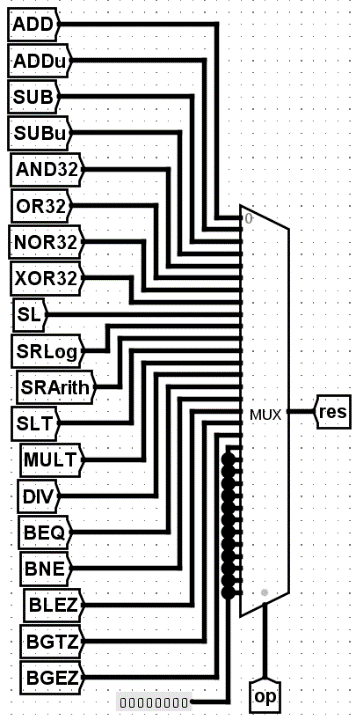
حالا به طراحی داخلی ALU می­پردازیم. ابتدا ترمینال­های ورودی و خروجی را می­سازیم. توجه کنید که خروجی Zero صرفا چک می­کند که آیا Result صفر است یا خیر.

شکل 2- ترمینال­های ورودی و خروجی ALU

بدنه اصلی ALU شامل کامپوننت­هایی­ است که محاسبات را انجام می­دهند. در این قسمت تمام خروجی­های ممکن را تولید می­کنیم.

شکل 3- تولید تمام خروجی­های ممکن در ALU



سپس با استفاده از یک MUX 5×32×32 و با توجه به ورودی operation خروجی مورد نظرمان را انتخاب می­کنیم.

شکل 4- انتخاب خروجی مورد نظر با توجه به بیت­های op

منستی

# منابع و مراجع

شسیب