به نام خدا

محمدفرحان بهرامی

401105729

پروژه امتیازی DSD

سوال هفتم میانترم

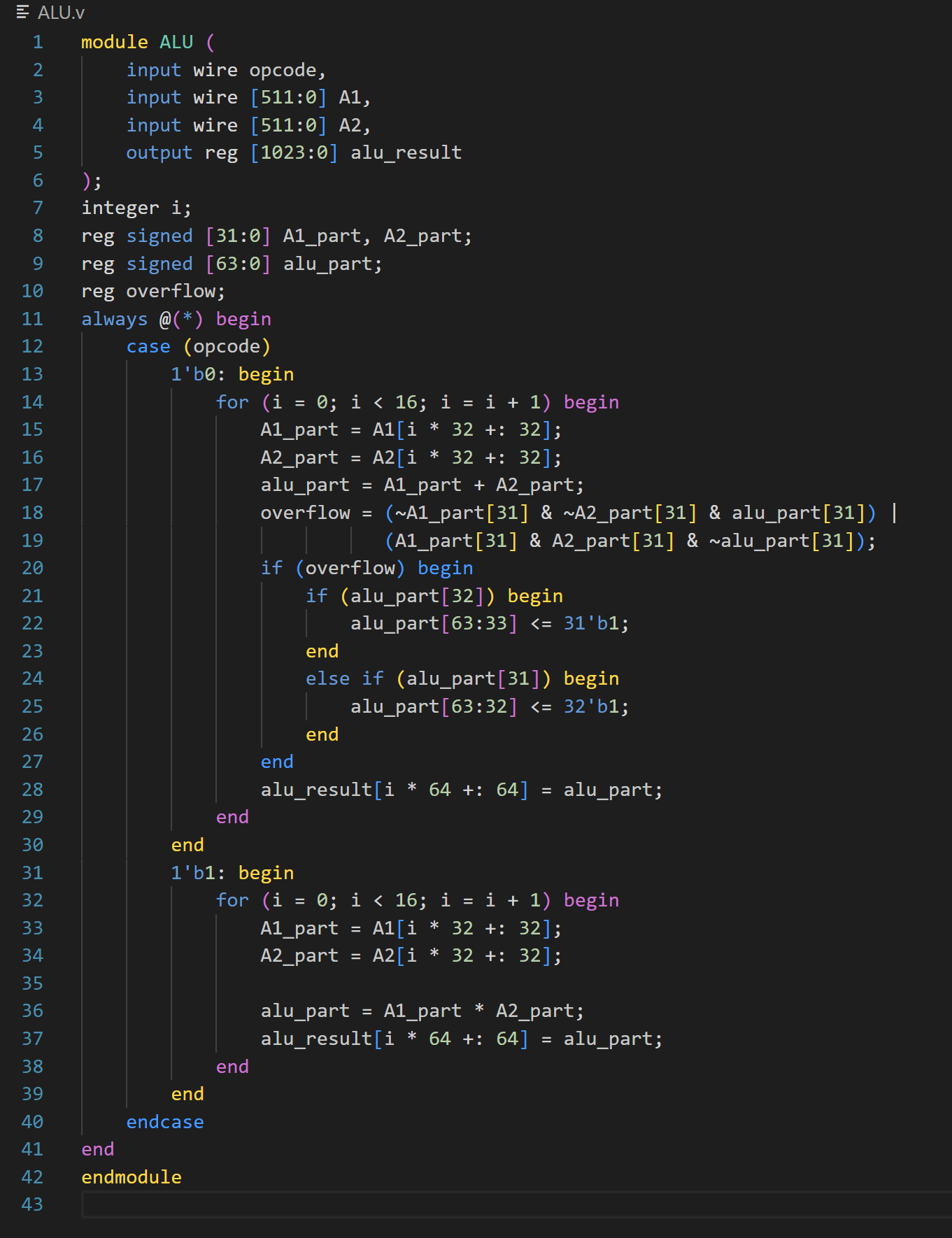
# ماژول های مورد نیاز:

ماژول ALU بر اساس مقدار opcode ورودی، دو عملیات جمع و ضرب را انجام می‌دهد. ورودی‌های ماژول شامل opcode سیگنال کنترلی برای تعیین نوع عملیات، A1 و A2 داده‌های ورودی 512 بیتی هستند و خروجی آن alu\_result نتیجه عملیات 1024 بیتی است. در داخل ماژول، چندین متغیر داخلی از جمله i شمارنده حلقه، A1\_part و A2\_part بخش‌های 32 بیتی از ورودی‌ها، alu\_part نتیجه موقتی عملیات 64 بیتی و overflow نشانگر سرریز تعریف شده‌اند.

در بلوک always حساس به تغییرات هر ورودی، بر اساس مقدار opcode، دو عملیات متفاوت انجام می‌شود. اگر opcode برابر با 0 باشد، عملیات جمع اجرا می‌شود. برای هر 16 بخش 32 بیتی از ورودی‌ها، جمع این بخش‌ها انجام شده و نتیجه در alu\_part ذخیره می‌شود. سپس، بررسی سرریز انجام می‌شود. اگر سرریز رخ دهد، بیت‌های بالایی alu\_part تنظیم می‌شوند تا سرریز را نشان دهند. در نهایت، نتیجه هر بخش 64 بیتی در alu\_result ذخیره می‌شود.

اگر opcode برابر با 1 باشد، عملیات ضرب اجرا می‌شود. برای هر 16 بخش 32 بیتی از ورودی‌ها، ضرب این بخش‌ها انجام شده و نتیجه در alu\_part ذخیره می‌شود. سپس، نتیجه هر بخش 64 بیتی در alu\_result ذخیره می‌شود.

کد مربوطه در صفحه بعد

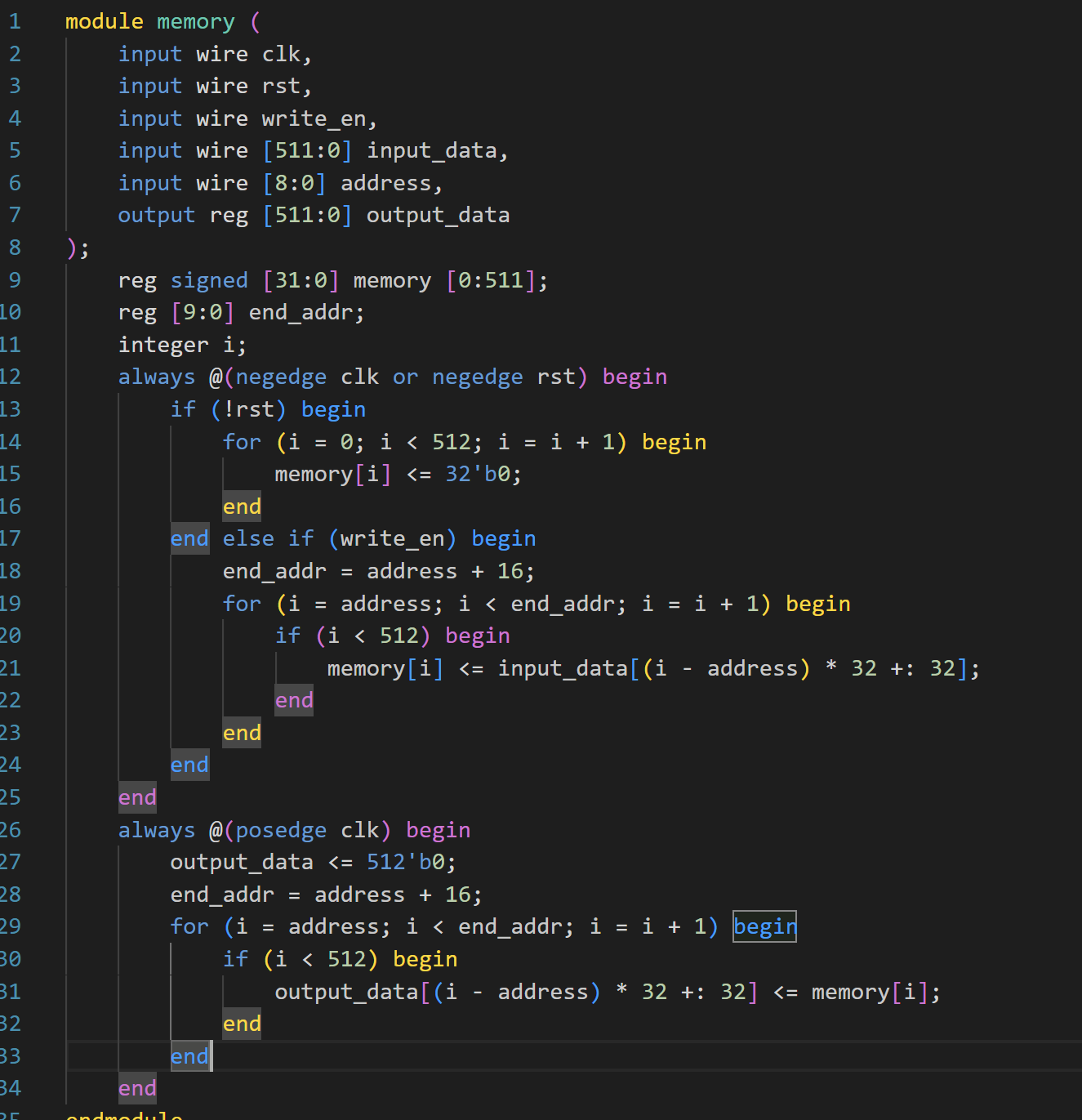


ماژول مموری:

ماژول memory یک حافظه 512 کلمه‌ای با عرض داده 32 بیت را پیاده‌سازی می‌کند. ورودی‌های ماژول شامل clk سیگنال ساعت، rst سیگنال ریست، write\_en فعال‌سازی نوشتن، input\_data داده‌های ورودی 512 بیتی و address آدرس 9 بیتی هستند و خروجی ماژول output\_data داده‌های خروجی 512 بیتی است.

در بلوک always حساس به لبه نزولی سیگنال ساعت یا ریست، اگر سیگنال rst غیر فعال باشد، یعنی در حالت ریست، یک حلقه برای مقداردهی صفر به تمام مکان‌های حافظه اجرا می‌شود. در صورت فعال بودن سیگنال write\_en، آدرس پایانی end\_addr به اندازه 16 کلمه بعد از آدرس ورودی تنظیم می‌شود و حلقه‌ای برای نوشتن داده‌ها از input\_data به مکان‌های مشخص شده در حافظه اجرا می‌شود. هر بخش 32 بیتی از input\_data به مکان مربوطه در حافظه نوشته می‌شود.

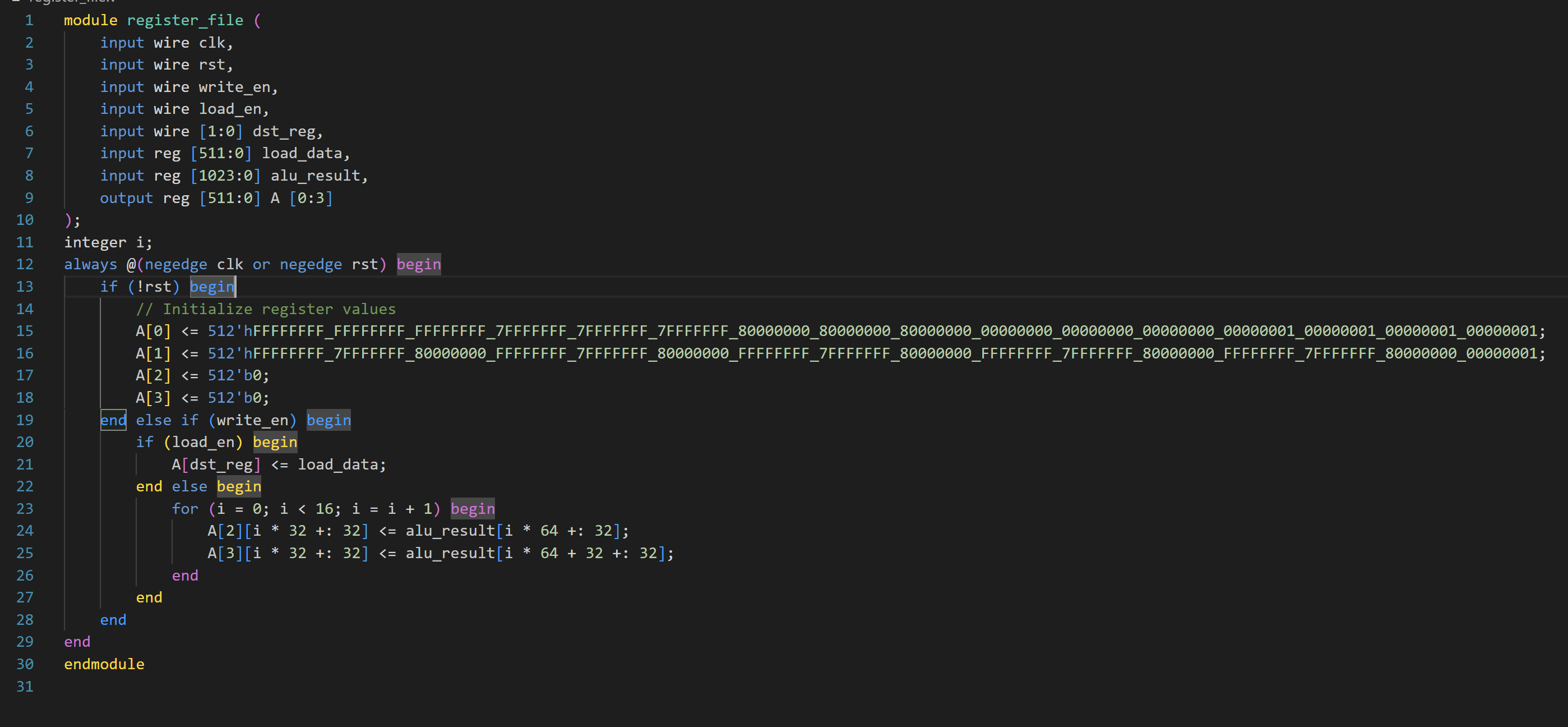
در بلوک always حساس به لبه بالارونده سیگنال ساعت، ابتدا output\_data به صفر تنظیم می‌شود. سپس آدرس پایانی end\_addr به اندازه 16 کلمه بعد از آدرس ورودی تنظیم می‌شود و حلقه‌ای برای خواندن داده‌ها از حافظه و انتقال آن‌ها به output\_data اجرا می‌شود. هر بخش 32 بیتی از حافظه به مکان مربوطه در output\_data منتقل می‌شود.



ماژول رجیستر فایل:

ماژول register\_file یک فایل رجیستر با چهار رجیستر 512 بیتی را پیاده‌سازی می‌کند که قابلیت نوشتن و بارگذاری داده‌ها را دارد. ورودی‌های ماژول شامل clk ساعت، rst ریست، write\_en فعال‌سازی نوشتن، load\_en فعال‌سازی بارگذاری داده، dst\_reg رجیستر مقصد، load\_data داده‌های بارگذاری و alu\_result نتیجه واحد ALU هستند. خروجی ماژول یک آرایه A از چهار رجیستر 512 بیتی است.

در بلوک always حساس به لبه نزولی ساعت و ریست، ابتدا مقادیر رجیسترها در صورت فعال بودن سیگنال ریست تنظیم می‌شوند. در این حالت، رجیسترهای A[0] و A[1] با مقادیر از پیش تعیین‌شده مقداردهی می‌شوند و رجیسترهای A[2] و A[3] به صفر تنظیم می‌شوند. اگر سیگنال write\_en فعال باشد، عملیات نوشتن انجام می‌شود. در صورتی که load\_en نیز فعال باشد، داده‌های ورودی load\_data در رجیستر مقصد dst\_reg نوشته می‌شوند. در غیر این صورت، داده‌های alu\_result در رجیسترهای A[2] و A[3] توزیع می‌شوند. برای این منظور، داده‌های 32 بیتی از alu\_result به صورت تکه‌های 64 بیتی به A[2] و A[3] منتقل می‌شوند.

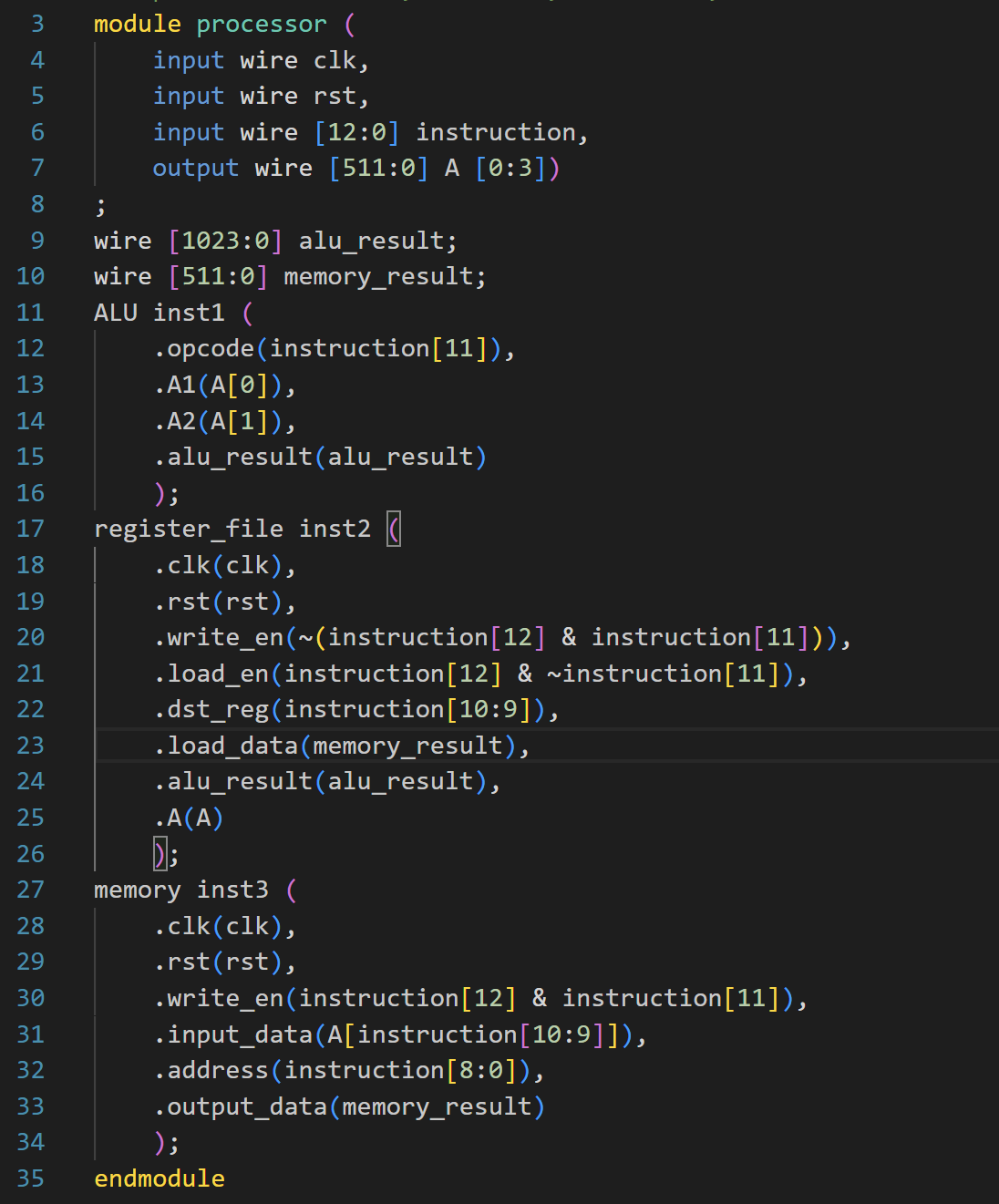


ماژول پردازنده:

ورودی‌های ماژول شامل clk سیگنال ساعت، rst سیگنال بازنشانی و instruction دستورالعمل 13 بیتی هستند. خروجی ماژول یک آرایه A از چهار رجیستر 512 بیتی است.

دستورالعمل 13 بیتی شامل سه بخش است: opcode بیت‌های 12 و 11، reg\_no بیت‌های 10 و 9 و address بیت‌های 8 تا 0 opcode مشخص‌کننده نوع عملیات است:

* 00 برای جمع
* 01 برای ضرب
* 10 برای بارگذاری
* 11 برای ذخیره



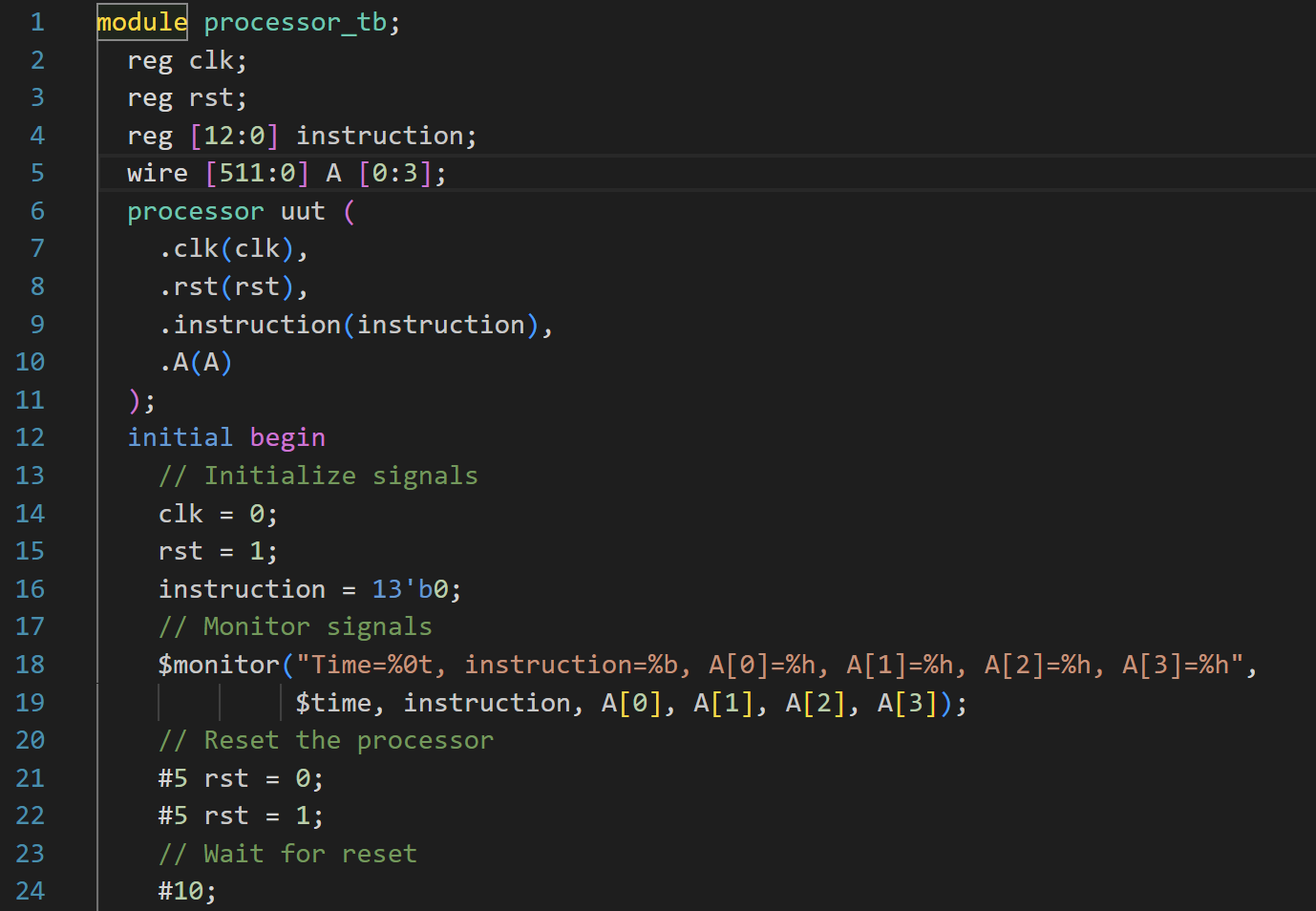
ماژول تست:

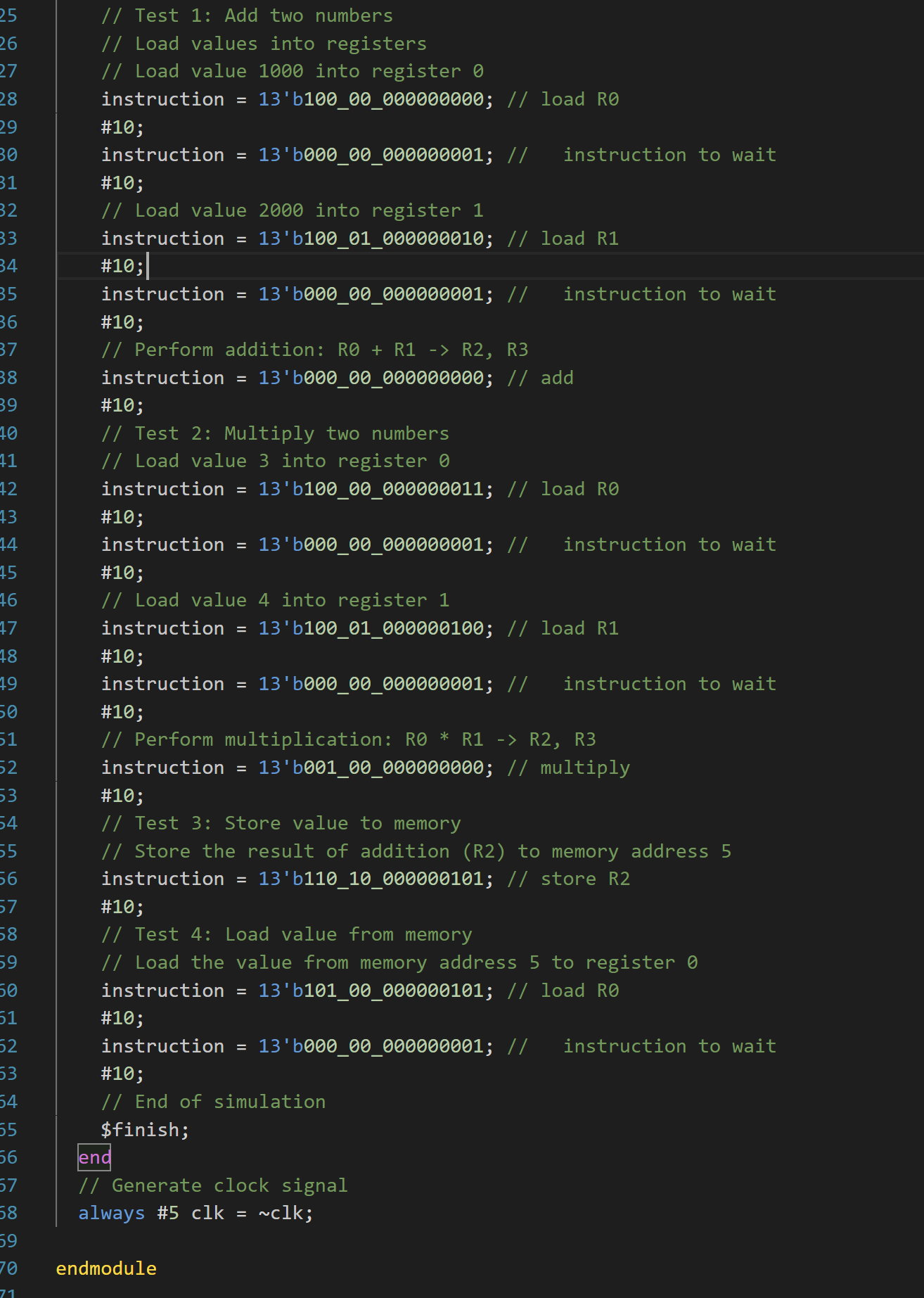
در تست اول، عملیات جمع دو عدد تست می‌شود. ابتدا مقادیر 1000 و 2000 به ترتیب به رجیسترهای 0 و 1 بارگذاری می‌شوند. این کار با استفاده از دستورالعمل‌های load انجام می‌شود. سپس، با استفاده از دستورالعمل add، مقدارهای موجود در رجیسترهای 0 و 1 جمع شده و نتیجه در رجیسترهای 2 و 3 ذخیره می‌شود.

در تست دوم، عملیات ضرب دو عدد تست می‌شود. ابتدا مقادیر 3 و 4 به ترتیب به رجیسترهای 0 و 1 بارگذاری می‌شوند. سپس، با استفاده از دستورالعمل multiply، مقدارهای موجود در رجیسترهای 0 و 1 ضرب شده و نتیجه در رجیسترهای 2 و 3 ذخیره می‌شود.

در تست سوم، عملیات ذخیره مقدار به حافظه تست می‌شود. مقدار موجود در رجیستر 2 (نتیجه جمع) به آدرس حافظه 5 ذخیره می‌شود.

در تست چهارم، عملیات بارگذاری مقدار از حافظه تست می‌شود. مقدار موجود در آدرس حافظه 5 به رجیستر 0 بارگذاری می‌شود.





که در نهایت بعد از شبیه سازی آن و خروجی گرفتن در مدل سیم به نتایج زیر دست پیدا میکنیم:

