به نام خدا

تکلیف سری دوم درس زبان‌های توصیف سخت‌افزار و مدارات

برای ارسال تکالیف، حتما به نکات زیر توجه کنید:

- برای هر سوال در نرم افزار ModelSim ، فایل جداگانه‌ای ایجاد کنید.
- **تمامی سوالات می‌بایست شبیه سازی شوند؛ بنابراین لازم است برای هر سوال Test Bench مناسب نوشته و آن را ضمیمه کنید.**
- علاوه بر ارسال فایل جواب سوال و فایل شبیه سازی، می‌بایست از شکل موج‌های موجود در شبیه سازی Screenshot گرفته و آن‌ها را با کیفیت مناسب (به طوری که اسامی سیگنال‌ها و شکل موج‌ها واضح باشند) ارسال کنید.
- لازم نیست تمامی فایل‌های موجود در پوشه پروژه را ارسال کنید!! تنها فایل v. جواب، فایل v. شبیه سازی و تصاویر شکل موج‌های شبیه سازی شده را ارسال کنید.
- در صورت برخی سوالات قید شده است که برای سوال، گزارش کوتاهی بنویسید؛ این گزارش را به صورت مختصر و در فرمت pdf به همراه بقیه فایل‌هایی که در مورد قبل اشاره شد، ارسال کنید.
- حتی الامکان اسامی سیگنال‌ها و متغیرها را با مسمّی انتخاب کنید و همچنین با نظم و ترتیب برنامه بنویسید.
- **توجه کنید که برنامه‌ها باید تماما قابل سنتز باشند.**
- برای نام‌گذاری فایل‌های ارسالی به شکل زیر عمل کنید:

  فرض کنید می‌خواهید فایل‌های سوال دوم و قسمت (ج) آن را نام‌گذاری کنید.

  فایل اصلی جواب : Q2_3.v

  فایل Test Bench: TestQ2_3.v

  تصاویر شکل موج‌های شبیه سازی: ScrQ2_3_1.jpg, ScrQ2_3_2.jpg, ScrQ2_3_3.jpg, ...

  فایل گزارش (در صورت لزوم) : ReportQ_2_3.pdf
- تمامی فایل‌های خود را در یک فایل zip قرار دهید. نام این پوشه باید به فرمت Hwx_Student ID.zip باشد. برای مثال : Hw2_9595959.zip
- در نهایت این فایل را در قسمت مربوطه در سامانه Yekta آپلود کنید.

سوال اول:

الف) یک رجیستر ۸ بیتی با مدهای عملکرد شیفت به چپ و شیفت به راست و با امکان بار موازی (PL) طراحی کرده و برای آن تست بنچی بنویسید. ورودی های کنترلی از نوع active high هستند. همچنین رجیستر با لبه بالارونده کلاک کار می کند به این صورت که همه سیگنال های کنترلی ورودی از جمله reset به صورت سنکرون با لبه مثبت کلاک اعمال می شوند. توجه کنید که اگر بیش از یک سیگنال کنترلی ورودی در لبه بالارونده سیگنال پالس ساعت فعال باشد اولویت به این ترتیب است: ۱)reset ۲) load ۳) shift_right ۳) shift_left

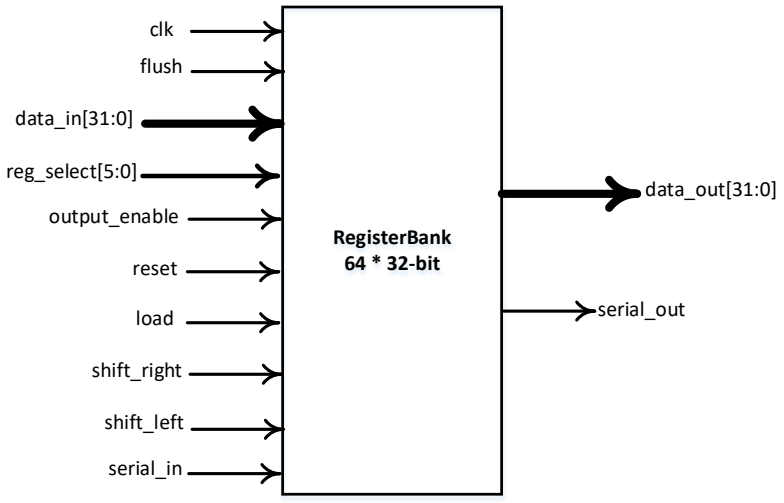
```
module myReg(clk, reset, pdata, qdata, load, shift_right, shift_left, serial_in, serial_out);

    input clk, reset, load, serial_in, shift_right, shift_left;
    input [7:0] pdata;
    output [7:0] qdata;
    output serial_out;

    //module body…

Endmodule
```

ب) فقط با استفاده از نمونه گیری از تعدادی ماژول طراحی شده در قسمت الف یک بانک رجیستری شامل ۶۴ عدد رجیستر ۳۲ بیتی طراحی کنید. ورودی و خروجی های این ماژول در شکل زیر نمایش داده شده اند



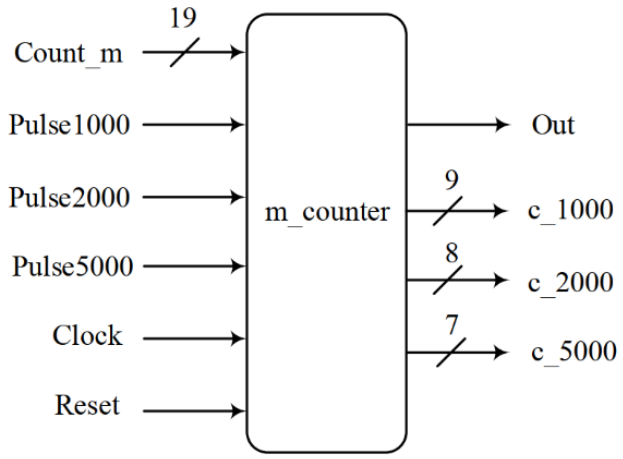
توجه کنید که اگر سیگنال output_enable صفر باشد خروجی data_out باید به صورت امپدانس بالا باشد ولی عملکرد رجیستر در مقابل سایر ورودی های کنترلی به طور معمول انجام می شود. سیگنال ورودی reg_select رجیستر مورد نظر از بین ۶۴ رجیستر ۳۲ بیتی موجود را برای اعمال فرمان انتخاب می کند. مثلا فعال بودن سیگنال reset در لبه بالارونده کلاک باعث صفر شدن محتوای رجیستر انتخاب شده می شود. سیگنال ورودی flush مقدار همه رجیستر های موجود در بانک رجیستری را به یکباره صفر می کند.

طراحی دوم:

فرض می‌کنیم که قرار است یک دستگاه پول‌شمار با امکان شمارش سه نوع اسکناس طراحی شود. این اسکناس‌ها مبالغ ۱۰۰۰، ۲۰۰۰ و ۵۰۰۰ تومانی را شامل می‌شوند. شمارش هر اسکناس با تولید یک لبه پالس بالارونده برای دستگاه مشخص می‌شود. ماژول مورد نظر برای طراحی این دستگاه به شکل زیر است که علاوه بر سیگنال‌های ورودی مختص هر نوع اسکناس (Pulse1000، Pulse2000 و Pulse5000)، شامل سیگنالی است که مبلغ مورد نظر برای شمارش را مشخص می‌کند. این سیگنال تحت عنوان count_m نامگذاری شده است.

در این طراحی قرار است بررسی شود که آیا حاصلجمع کل اسکناس‌های شمرده شده از عدد ورودی count_m بزرگتر یا مساوی است. بنابراین در خروجی یک سیگنال تک بیتی خواهیم داشت که در صورتی که مقدار مجموع اسکناس‌های شمارش ده از مبلغ تعیین شده بیشتر یا مساوی شد، این سیگنال ۱ شود و در غیر این صورت مقدار صفر دارد. از سوی دیگر، **تعداد اسکناس‌های شمارش شده** مربوط به هر نوع اسکناس هم قرار است در خروجی مشخص شود. بنابراین سه خروجی متناظر با هر نوع اسکناس خواهیم داشت که تعداد هر اسکناس را مشخص میکند. حداکثر عددی که قرار است این دستگاه بشمرد ۵۰۰۰۰۰ تومان است. بنابراین ورودی count_m یک سیگنال ۱۹ بیتی خواهد بود.

فرض کنید که ماژول دارای یک ورودی ریست آسنکرون و Active low باشد که با فعال شدن آن، خروجی و تمامی رجیسترهای داخلی مدار، مقدار صفر خواهند گرفت. ماژول مورد نظر دارای یک کلاک با فرکانسی بسیار بالاتر از نرخ تغییرات سایر سیگنال های ورودی است. **توجه کنید که تنها سیگنال پالس ساعت طرح شما فقط سیگنال Clock است و هیچ سیگنال دیگری نباید به عنوان پالس ساعت مورد استفاده قرار بگیرد.**

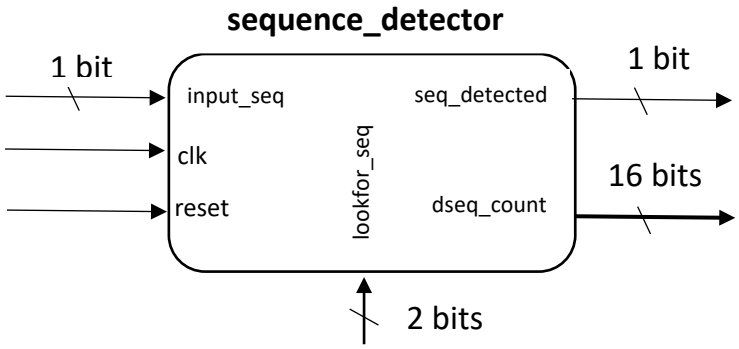


کد وریلاگی برای این مدار نوشته و عملکرد آن را با یک تست بنچ مناسبی بررسی کنید.

طراحی سوم:

مطابق شکل زیر یک ماژول sequence detector درسطح Behavioral تعریف کنید. عملکرد ورودی ها و خروجی های این ماژول به شرح زیر است:

- input_se: این ورودی تک بیتی در واقع رشته ای از ۰ و۱ هاست که به صورت سریال و با هر لبه پایین رونده پالس ساعت clk وارد ماژول میشود.
- lookfor_seq: این ورودی ۲ بیتی یکی از رشته های ۱۰۱۱۱، ۰۱۰۱۰، ۱۰۱۰۱ و یا ۱۰۱۰۰ را برای جستجو انتخاب می‌کند به این معنی که مثلا اگر seq_select=2 باشد، ماژول به دنبال رشته ۱۰۱۰۰ در رشته اعداد ورودی میگردد.
- seq_detected: این پایه یک بیتی پس از هر بار شناسایی رشته مورد نظر برای یک پالس ساعت یک میشود و سپس مجددا صفر می شود.
- seq_count: این خروجی ۱۶ بیتی تعداد رشته های تشخیص داده شده را نشان میدهد. با تغییر ورودی lookfor_seq این خروجی ریست می شود.



طراحی چهارم:

ماژولی برای محاسبه باقی‌مانده تقسیم رشته باینری ورودی بر 7 و تعیین بخش پذیری آن طراحی کنید. به عنوان مثال باقی‌مانده رشته 10101 - که معادل عدد 21 دسیمال است - بر 7، برابر 0 است.

فرض کنید که قبل از شروع به کار ماژول، باقی‌مانده برابر 0 است.

پورت‌های ماژول عبارت‌اند از:

- ✓ ورودی تک بیتی String که رشته ورودی را می‌سازد و باقیمانده آن بر 7 محاسبه می‌شود. (در هر پالس ساعت، یک بیت به سمت راست رشته اضافه می‌شود. به عنوان مثال اگر دو پالس ساعت بیت 1 و پس از آن یک پالس ساعت بیت 0 خوانده شود، رشته 110 حاصل می‌شود.)

- ✓ ورودی تک بیتی Clock که پالس ساعت ماژول است.
- ✓ ورودی تک بیتی Reset که به صورت Synchronous عمل می‌کند.
- ✓ خروجی سه بیتی Remainder که در هر لحظه باقیمانده تقسیم رشته ورودی بر 7 را مشخص می‌کند.

خروجی تک بیتی Divisible که هنگامی که رشته ورودی بر 7 بخش‌پذیر باشد به مدت یک پالس ساعت 1 می‌شود. (اگر با پالس ساعت بعدی باقی‌مانده تغییر کرد و 0 نماند، این بیت باید 0 شود.)