به نام خدا

تکلیف سری پنجم درس زبان‌های توصیف سخت‌افزار و مدارات

آخرین زمان تحویل: 9 خرداد ماه 1399

نکاتی در مورد انجام و ارسال تکالیف:

1- برای هر ماژول که می‌نویسید، یک فایل v. جداگانه درنظر بگیرید. همه‌ی ماژول‌ها را در یک فایل v. ننویسید.
2- ماژول Testbench هر سوال را حتما به همراه سایر ماژول‌ها ارسال فرمایید.
3- برای تحویل تکلیف یک گزارش مختصر از نحوه‌ی انجام تکالیف یا پاسخ به سوالات مطرح شده نیاز است. این گزارش در قالب PDF خواهد بود. دقت کنید که نیاز به کپی کردن کدها در داخل گزارش نیست؛ بلکه فایل کدها بصورت جداگانه باید ارسال شود.
4- توجه فرمایید که نیازی به ارسال همه‌ی فایل‌های پروژه‌ی نرم افزار ISE نیست! فقط فایل‌های v. را در کنار گزارش ارسال فرمایید.
5- در نهایت یک فایل زیپ به شکل HW5_StudentID.zip حاوی گزارش و فایل‌های v. را قبل از موعد تحویل در سامانه دروس آپلود فرمایید.
6- زدن دکمه‌ی Hand In پس از آپلود فراموش نشود.

سوال اول)

مابین یک فرستنده و گیرنده، داده‌ها با پروتکلی مشابه اینترفیس UART رد و بدل می شوند با این تفاوت که داده ها به جای 8 بیتی، 7 بیتی می باشند. از این هفت بیت، 4 بیت آن داده خواهد بود و 3 بیت دیگر نقش چک کردن خطا را ایفا می‌کنند. وظیفه‌ی شما این است که ماژولی طراحی کنید که 4 بیت داده را به شکل موازی از کاربر دریافت کرده، 3 بیت چک کردن خطا (parity) را به آن اضافه کرده و آن را به صورت سریال ارسال کند. همچنین لازم است که در سمت گیرنده نیز ماژول طراحی شود که با دریافت این هفت بیت (به شکل سریال)، اقدام به ساختن سه بیت Check Bits کرده تا در صورت بروز خطا آن را اصلاح کند. نحوه‌ی ساخته شدن این بیت‌ها به طور کلی در زیر بیان شده است:

داده‌های آماده به ارسال در ماژول فرستنده: x2, x4, x5, x6

p0 = [x2] xor [x4] xor [x6]
p1 = [x2] xor [x5] xor [x6]
p3 = [x4] xor [x5] xor [x6]

بیت‌های چک کردن خطا (parity Check): p0, p1, p3 به شکل روبرو محاسبه می‌شوند.

داده‌ها در یک قالب هفت بیتی به شکل {x6, x5, x4, p3, x2, p1, p0} (x6 پر ارزش است) از فرستنده به سمت گیرنده ارسال می‌شوند. در سمت گیرنده، پس از دریافت این هفت بیت (که مشخص نیست تحت تاثیر نویز درست دریافت

شده است یا خیر)، لازم است که سه بیت از روی آن‌ها ساخته شود. در این رابط فرض شده است که حداکثر یک بیت خطا در هر بسته‌ی هفت تایی ممکن است رخ دهد. بنابراین با تعیین مکان آن خطا، می‌توان آن را اصلاح کرده و به داده‌ی معتبر دست یافت. نام این سه بیت، **c0, c1, c2** خواهد بود که نحوه‌ی محاسبه‌ی آن‌ها با فرض اینکه هفت داده‌ی دریافتی به شکل **{r6, r5, r4, r3, r2, r1, r0}** (**r6** پر ارزش است) باشد، مطابق روابط روبرو است:

```
c0 = [r0] xor [r2] xor [r4] xor [r6]
c1 = [r1] xor [r2] xor [r5] xor [r6]
c2 = [r3] xor [r4] xor [r5] xor [r6]
```

در نهایت عدد سه بیتی **{c2, c1, c0}** نشان‌دهنده‌ی شماره بیتی است که در آن خطا رخ داده است. به طور مثال اگر مقدار آن **3** شود به این معنی است که داده‌ی **x2** از سمت فرستنده، اشتباه دریافت شده است و باید معکوس شود (**NOT**) تا به داده‌ی معتبر برسیم.

موارد خواسته شده به طور خلاصه:

1- ماژول فرستنده
   - 1-1- **Task** مربوط به محاسبه **parity**
   - 1-2- چیدن داده‌ها در کنار هم و ارسال آن
2- ماژول گیرنده
   - 2-1- **Function** مربوط به ساختن بیت‌های **Check**
   - 2-2- عملیات تصحیح خطا با توجه به بیت‌های **Check**
   - 2-3- نشان دادن داده‌ی معتبر
3- ماژول تست این رابط، به نحوی که ارسال و دریافت با و بدون خطا هر دو شبیه‌سازی شوند.

ماژول‌ها نوشته شده را شبیه‌سازی کرده و نتایج شبیه‌سازی را در گزارش ضمیمه کنید. برای شبیه سازی یک **testbench** طراحی کنید که هر دو ماژول فوق مستقیما در آن **instant** شده باشد. حداقل **10** داده **4** بیتی را به فرستنده تحویل دهید و خروجی گیرنده را چک کنید. حداقل در **3** مورد از **10** داده فوق در انتقال سیگنال سریال از فرستنده به گیرنده یک بیت خطا ایجاد کنید و نتیجه را مشاهده کنید. حداقل در یک مورد از **10** مورد فوق، در مسیر انتقال داده از فرستنده به گیرنده **2** بیت خطا ایجاد کنید و نتیجه را مشاهده کنید.

برای انتقال داده از فرستنده به گیرنده به جای اتصال مستقیم پورت خروجی فرستنده به گیرنده از دستور **assign** استفاده کنید تا در لحاتی که دوست دارید بتوانید خطا ایجاد کنید.

سوال دوم)

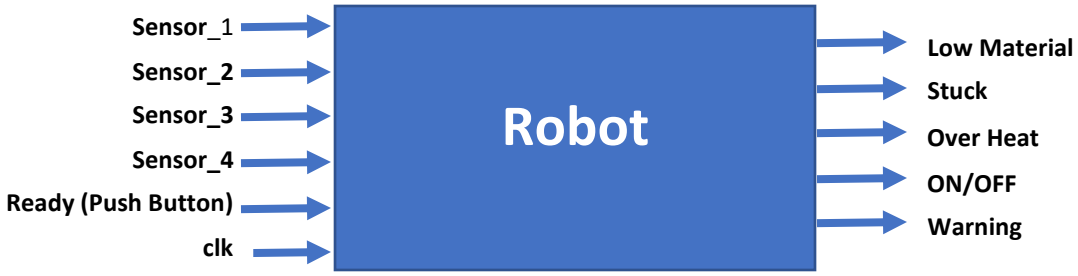
روباتی در یک کارخانه وظیفه ي مدیریت خط تولید را بر عهده دارد. این روبات با استفاده از دادههایی که از سنسورهاي مختلف دریافت میکند، تصمیم به انجام کارهاي مختلف میگیرد. فرکانس کاري روبات **4** مگاهرتز است. در هر پالس ساعت، اطلاعات **4** سنسور از خط تولید خوانده میشود که به شرح زیر هستند:

1- سنسور دماي دستگاه: اگر دماي دستگاه از حد مجاز بیشتر شود، این سنسور مقدار یک را به روبات میفرستد و در غیر این صورت مقدارش صفر است. روبات باید در صورت تشخیص دماي زیاد دستگاه را خاموش کند تا زمانی که مجددا شرایط براي روشن شدن فراهم باشد.

2-سنسور گیرکردن دستگاه: این سنسور هم مانند سنسور بالا عمل میکند اما با این تفاوت که وقتی اشکال رخ میدهد، براي راه اندازي مجدد دستگاه لازم است که تکنسین مشکل را حل کند و توسط یک کلید فشاري به روبات اعلام کند که مشکل برطرف شده و میتواند دستگاه را مجددا راه اندازي کند.

3- سنسور کم شدن مواد اولیه: این سنسور هم مانند سنسور اول عمل میکند. براي پاسخ به این سنسور روبات باید خروجی Low_Material را فعال کند و به مدت 2 دقیقه منتظر بارگزاري مواد اولیه شود (منتظر خاموش شدن سیگنال سنسور شود). اگر ظرف مدت دو دقیقه مواد اولیه تامین نشوند، خط تولید خاموش میشود و فقط در صورت اعلام تکنسین توسط همان کلید فشاري که در سنسور دوم وجود داشت و بارگزاري مجدد مواد اولیه (خاموش شدن سیگنال سنسور) مجددا روشن میشود.

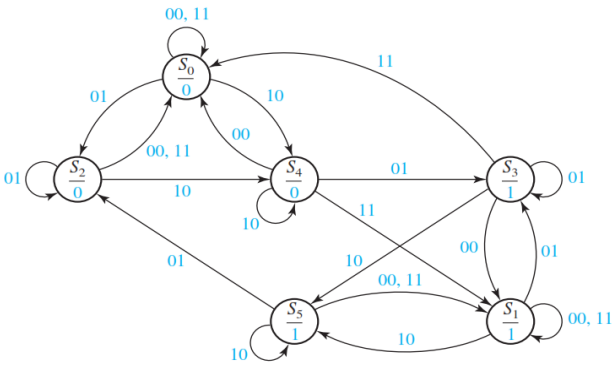
4-سنسور خطای مرگبار:این سنسور هم مانند سنسور اول عمل می کند .در صورت بروز خطای مرگباربابد خط تولید کاملا خاموش شدهو اژیر خطر که به خروجی warning متصل است فعال شود. برای خروج از این حالت وبازگشت به حالت عادی روبات ریست شود.



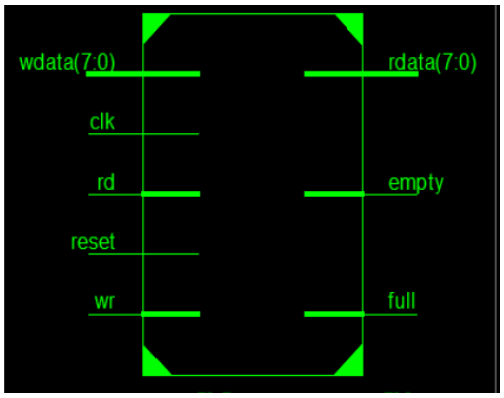
روبات را به نحوي شبیه سازي کنید که وقوع هر چهار اتفاق شبیه سازي شود. طراحی شما باید با ماشین حالت صورت گیرد (نتایج شبیهسازي در گزارش آورده شود).

سوال سوم)

یک سیستم کنترل کنندهی فرآیند ماشین حالتی به شکل رو به رو دارد. کد وریلاگ این سیستم کنترل کننده را با در نظر گرفتن ورودیها و خروجیهای مناسب بنویسید. نتایج شبیه سازی را به همراه نتایج RTL ضمیمه کرده و به همراه کد نوشته شده ارسال نمایید. (یک سناریوی شبیه سازی بگویید. مثلا از فلان حالت شروع شود و به فلان حالت برود)



سوال چهارم)



با استفاده از ماشین حالت یک **FIFO** سنکرون به عمق **1kByte** که عرض بیت داده ی ورودی و خروجی ان **8** باشد. برای طراحی این فیفو ابتدا یک حافظه **RAM** دو پورت با استفاده از **core generator** تولید کنید و سپس با تعریف دو اشاره گر **write_addr** و **read_addr** و استفاده از ماشین حالت سعی کنید بسته به درخواست کاربر داده را در آدرس مناسب داخل رم نوشته یا از آدرس مناسب در داخل حافظه بخوانید. در صورت پر بودن فیفو داده جدیدی در داخل فیفو ثبت نمی شود و در صورت خالی بودن فیفو داده ای را فیفو قابل خواندن نیست.(در ضمن این فیفو یک پورت **reset** و **count** هم دارد که در شکل مشخص نشده)

ماژول **Testbench** را هم به گونه ای طراحی کنید که شرایط پر بودن و خالی بودن فیو حتما تست شود. (شبیه سازي را انجام داده ونتایج رابه فایلهاي تکلیف ضمیمه کنید)