بسمه تعالی

درس طراحی سیستم‌های نهفته مبتنی بر FPGA

آزمایش ۴: طراحی یک شتاب‌دهنده‌ی سخت‌افزاری با رابط Avalon

پردیس دانشکده‌های فنی دانشگاه تهران

دانشکده مهندسی برق و کامپیوتر

دکتر بیژن علیزاده

دستیاران آموزشی:

سیامک بیگ محمدی siamackbm@yahoo.com

علیرضا خادم arkhadem@ut.ac.ir

رضا فرضی farzi.reza1994@gmail.com

پاییز ۱۳۹۷

مدت آزمایش: سه جلسه

اهداف آزمایش:

- ✓ آشنایی با رابط Avalon و مفاهیم پیشرفته در Nios II
- ✓ آشنایی با طراحی یک شتاب‌دهنده‌ی سخت‌افزاری

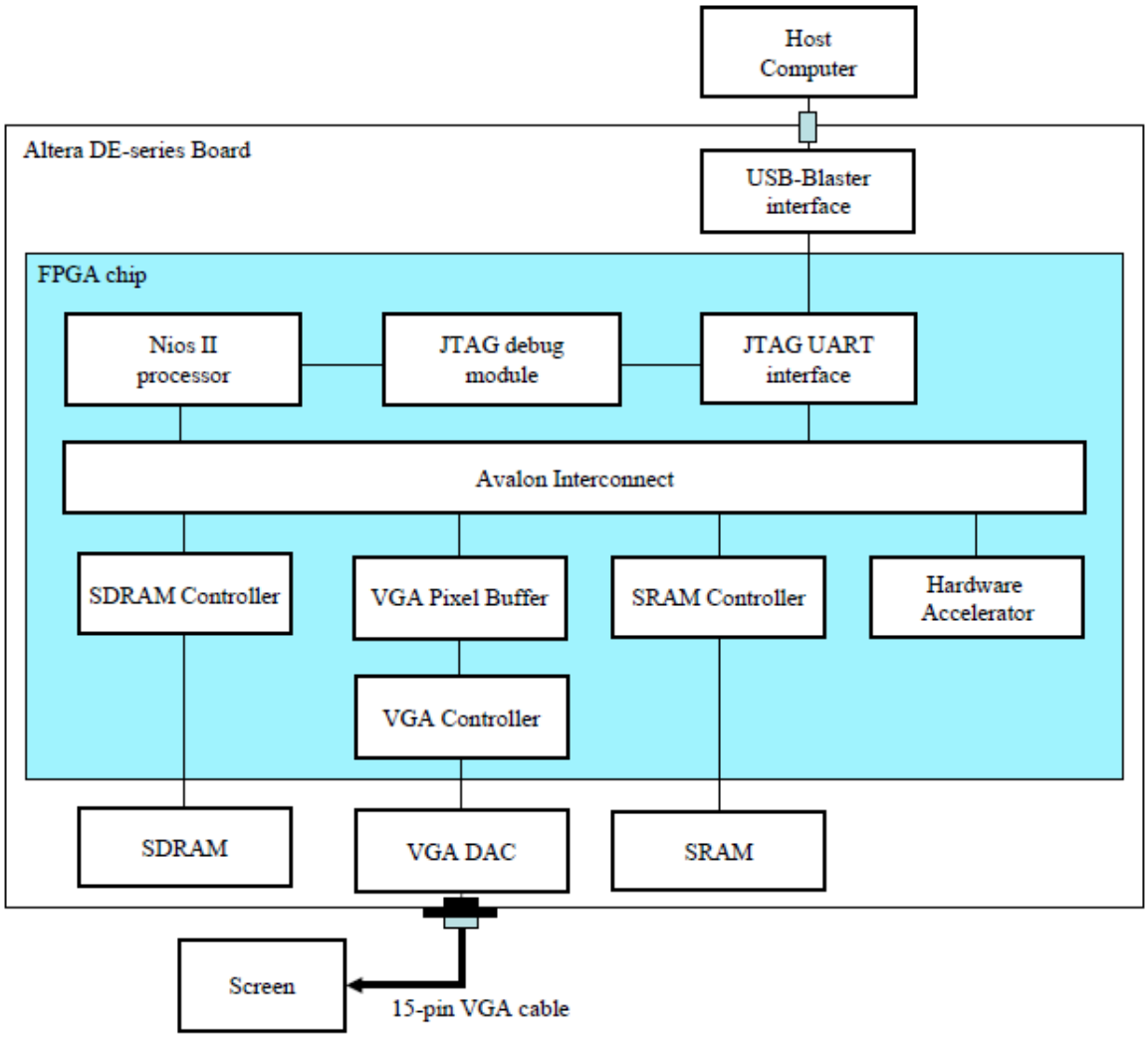
مقدمه

در این آزمایش با مفهوم شتاب‌دهی سخت‌افزاری[1] آ شنا خواهید شد. شتاب‌دهنده‌های سخت‌افزاری مدارهایی ه ستند که به منظور بردا شتن وظایف خا صی از دوش پردازنده طراحی شده‌اند. برای مثال الگوریتم محا سبه‌ی FFT را در اندازه‌های بزرگ توسط نرم‌افزار نمی‌توان به صورت بی‌درنگ انجام داد. با سپردن این کار به سخت‌افزار، اولاً سخت‌افزار می‌تواند همان کار را در زمان کم‌تری انجام دهد (مشابه روند مشاهده شده با دستورات اختصاصی) و ثانیاً نرم‌افزار همزمان به انجام سایر وظایف پردازشی خود می‌پردازد (بر خلاف روند دستورات اختصاصی).

[1] Hardware Acceleration

شرح آزمایش

شکل ۱ یک سیستم نمونه با شتاب‌دهنده‌ی سخت‌افزاری را نشان می‌دهد. تنها تفاوت این شکل با سیستم DE2 Media Computer در وجود واحد شتاب‌دهنده‌ی سخت‌افزاری ا ست. این شتاب‌دهنده پس از دریافت د ستور انجام عملیات مورد نظر، می‌تواند در صورت لزوم مستقیماً از طریق باس Avalon با سایر واحدهای سخت‌افزاری نیز در ارتباط باشد.



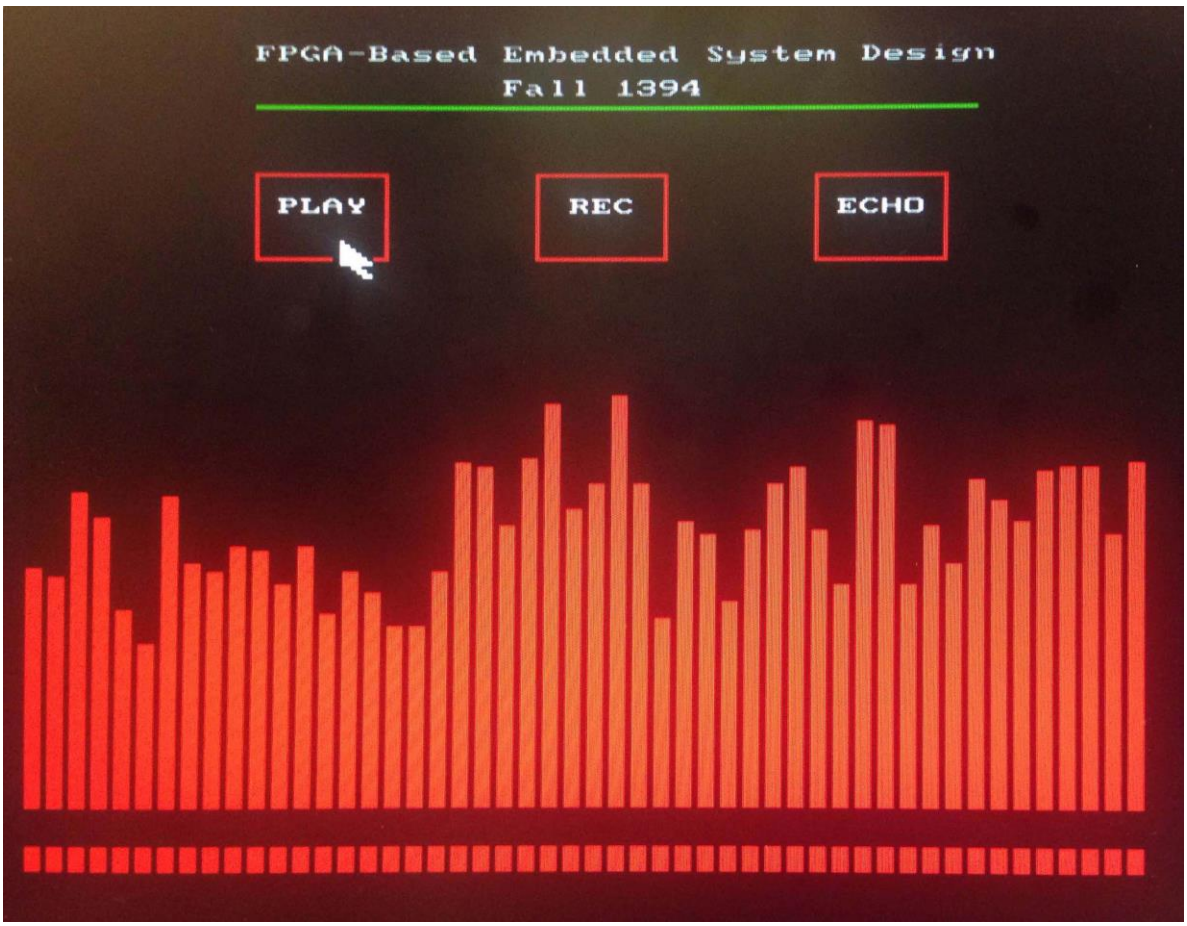
شکل ۱ سیستمی با شتاب‌دهنده‌ی سخت‌افزاری.

بخش‌های زیر را به ترتیب انجام دهید:

۱- طراحی سیستم مورد نظر به صورت نرم‌افزاری

در این بخش بر پایه‌ی سیستم DE2 Media Computer و نرم‌افزاری که در آزمایش دوم طراحی کرده‌اید، سیستمی پیاده کنید که:

- با کلیک روی دکمه‌ی .Rec، ۱۰ ثانیه از صدای ورودی ضبط شود. سپس متوسط اندازه‌ی دامنه‌ی سیگنال در N بازه با زمان ۱۰/N ثانیه، به صورت مستطیل‌های کنار هم رسم شود. برای رسم این نمودار تابع Plot_Audio را بنویسید و آن را پس از اتمام ضبط صدا فراخوانی کنید (باید با N دلخواه کار کند).
- با کلیک روی دکمه‌ی play، همراه با پخش صدای ضبط شده مستطیل‌هایی در زیر مستطیل‌های میانگین نمایش داده شود که نشانگر مکان فعلی پخش صدا باشد (شکل ۲).

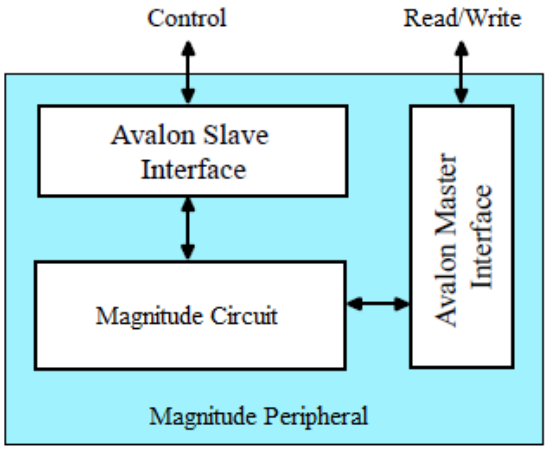


شکل ۲ رابط گرافیکی سیستم مورد نظر.

می‌توانید در بخش تنظیمات پروژه بهینه‌سازی کامپایلر را خاموش کنید تا کد شما به همان شکلی که نوشته‌اید اجرا شود. زمان اجرای الگوریتم محاسبه‌ی دامنه‌ی متوسط را برای مقایسه‌های بعدی اندازه بگیرید (گزارش شود).

۲- طراحی شتاب‌دهنده‌ی سخت‌افزاری

در این بخش به طراحی سخت‌افزار شتاب‌دهنده و انجام عملیات بخش ۱ با این سخت‌افزار خواهیم پرداخت. در شکل ۳ شمای کلی مدار جانبی سخت‌افزار محاسبه‌ی متوسط دامنه رسم شده است که باید به سیستم DE2 Media Computer اضافه شود. این شتاب‌دهنده شامل سه بخش است: مدار محاسبه‌ی دامنه، یک رابط Avalon Slave و یک رابط Avalon Master. پروسسور از طریق رابط Slave (با رابط Master خود) آدرس حافظه‌ی شروع بافر صدای راست و چپ، آدرس مکان ذخیره‌سازی جواب، تعداد بازه‌ها (Num) و تعداد نقاط در هر بازه (Size) را در اختیار مدار محاسبه‌ی دامنه قرار می‌دهد و با نوشتن در رجیستر Go مدار محاسبه‌ی دامنه کار خود را شروع می‌کند و از طریق رابط Master خود، داده‌ها را از مموری خوانده و پردازش می‌کند و نتیجه را در آدرس مشخص‌شده از مموری توسط پردازنده ذخیره می‌کند. همچنین سیگنال Done را فعال می‌کند. پردازنده روی این سیگنال polling انجام می‌دهد و با ۱ شدن آن مستطیل‌های بیانگر دامنه‌ی سیگنال را رسم می‌کند. همانطور که مشاهده می‌شود این روند مشابه عملیات DMA[۲] است با این تفاوت که جهت سهولت کار از اینتراپت استفاده نشده است و به جای آن polling انجام می‌شود.



شکل ۳ شمای کلی مدار جانبی محاسبه‌ی متوسط دامنه.

[2] Direct Memory Access

مراحل زیر را به ترتیب انجام دهید:

۱-۲ طراحی رابط Avalon Memory-Mapped Slave

همانطور که در شکل ۱ نشان داده شده است، در سیستم مبتنی بر Nios II، برای ارتباط میان پروسسور و سایر اجزا از گذرگاه Avalon استفاده می‌شود که در آن پردازنده‌ی Nios II نقش Master (پایه) را دارد. برای آنکه دستورات لازم را توسط پردازنده به شتاب‌دهنده‌ی سخت‌افزاری ارسال کنیم، لازم است شتاب‌دهنده شامل یک رابط Slave (پیرو) باشد. در این بخش لازم است یک رابط Avalon Memory-Mapped Slave طراحی کنید که قابلیت خواندن از رجیسترهای شکل ۴ و نوشتن در آن‌ها را داشته باشد. برخی از این رجیسترها را در بخش بعد استفاده خواهیم کرد.

Slave Address	31	30..12	11..1	0	
00	Done	Size	Num	Go	Config. Reg.
01					Right Addr.
10					Left Addr.
11					Out Addr.

شکل ۴ رجیسترهای مورد استفاده در رابط Slave.

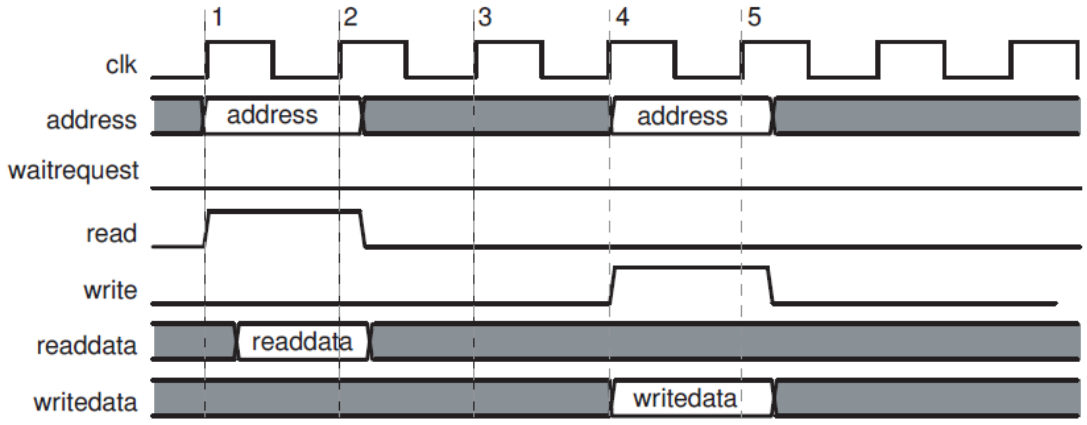
به منظور طراحی رابط Slave لازم است با پروتکل مورد استفاده در گذرگاه Avalon آشنایی داشته باشید. گذرگاه Avalon یک رابط استاندارد و راحت برای اتصال واحدهای سخت‌افزاری درون FPGA می‌باشد. این استاندارد انواع متفاوتی از جمله High Speed و Memory Mapped تعریف می‌کند. در این آزمایش از رابط Avalon Memory-Mapped استفاده خواهیم کرد. رابط معمول Avalon Memory-Mapped نوشتن و خواندن را توسط سیگنال waitrequest کنترل شده با Slave اجرا می‌کند. برای شروع یک انتقال، Master سیگنال‌های address، byteenable، read/write و writedata را در هنگام پالس کلاک منتشر می‌کند. زمانی که یک Slave این سیگنال‌ها را دریافت می‌کند و chipselect آن فعال است، اطلاعات را ذخیره می‌کند یا اطلاعات درخواست شده را در خروجی (readdata) می‌گذارد. اگر Slave تا لبه‌ی کلاک بعدی قادر به پاسخ‌گویی نباشد، می‌تواند سیگنال waitrequest را قبل از لبه‌ی بالارونده‌ی کلاک منتشر کند. بدین ترتیب انتقال اطلاعات به تأخیر می‌افتد و Master، آدرس و سیگنال‌های کنترلی را ثابت نگه می‌دارد.

هنگام پیاده‌سازی رابط Slave اسامی تمامی سیگنال‌های آن را با پیشوند _avs_avalonslave مانند avs_avalonslave_read انتخاب کنید. در این قسمت پروژه، رابط Slave شما نیازی به انتشار waitrequest

ندارد زیرا فرض بر این اســت که می‌تواند در یک کلاک یا تعداد کلاک ثابتی پاســخ دهد (و باید این کار را انجام دهد). شــکل ۵ زمان‌بندی مورد نیاز در گذرگاه Avalon را (با عملکرد بدون waitrequest) نشــان می‌دهد. دقت کنید سیگنال chipselect که در طول انتقال خواندن/نوشتن فعال می‌شود، در این شکل نشان داده نشده است. روند شکل ۵ بدین صورت است:

۱) Master سیگنال‌های read و address را روی لبه‌ی بالارونده‌ی کلاک منتشر می‌کند. در همان سیکل، Slave سیگنال‌های Master را دیکد می‌کند و readdata را ارائه می‌کند.
۲) Master، سیگنال readdata را روی لبه‌ی بالارونده‌ی کلاک می‌خواند و آدرس و سیگنال‌های کنترلی را غیر فعال می‌کند. در این لحظه انتقال پایان می‌یابد.
۳) هیچ سیگنال کنترلی منتشر نمی‌شود.
۴) Master، ســـیگنال‌های address، write و writedata را روی لبه‌ی بالارونده‌ی کلاک منتشـــر می‌کند. سیگنال‌های Master ثابت باقی مانده و Slave آن‌ها را دیکد می‌کند.
۵) Slave، writedata را روی لبه‌ی بالارونده‌ی کلاک می‌گیرد. Master سیگنال‌های writedata، address و write را غیر فعال می‌کند و انتقال پایان می‌یابد.

برای آشنایی بیشتر با رابط Avalon به اسلایدهای درس و در صورت نیاز به فصل ۳ از مرجع [1] مراجعه کنید.



شکل ۵ زمانبندی Avalon برای انتقال خواندن و نوشتن بدون سیگنال waitrequest.

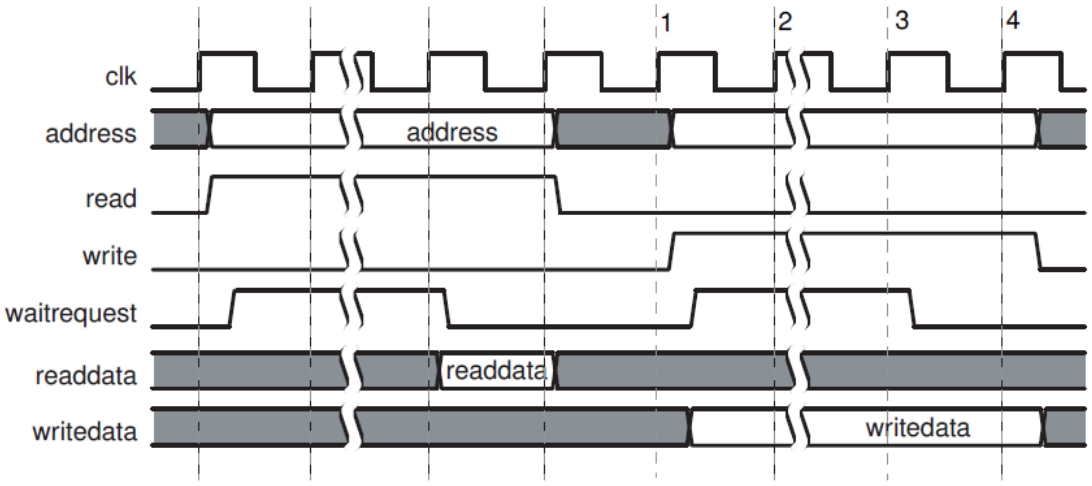
برای آنکه رابط Slave خود را تست کنید، بیت‌های 0 تا 17 رجیسترها را به LEDR وصل کنید (با استفاده از یک مالتی‌پلکسر که با سوئیچ‌های SW[1:0] کنترل می‌شود). پس از اضافه کردن این سخت‌افزار در محیط Qsys، کامپایل کردن پروژه در Quartus و پروگرام کردن FPGA، نرم‌افزاری بنویسید که در رجیسترهای مذکور

مقادیری را بنویسد و آن را خوانده و با مقدار نوشته شده مقایسه کند. توجه کنید که در Qsys باید PIOهای مربوط به LEDهای قرمز و سوئیچ‌ها را غیر فعال کنید (تیک کنار آن‌ها را بردارید) و در ماژول خود پورت‌های مربوطه را export کنید. همچنین تغییراتی نیز در پروژه‌ی Quartus و Wrapper مربوط به سیستم اعمال کنید تا LEDها و سوئیچ‌ها به درستی متصل شوند.

۲-۲ طراحی رابط Avalon Memory-Mapped Master

در این بخش رابط Master را برای ارتباط با کنترلر SDRAM پیاده‌سازی خواهیم کرد. مدار بخش قبل را به گونه‌ای تغییر دهید که با نوشتن در رجیستر Go (توسط پروسسور) به تعداد مشخص شده (Num) از آدرس راست و چپ را خوانده و جمع کند و نتیجه را در آدرس خروجی بنویسد. پس از این کار سیگنال Done فعال می‌شود. پس از ساخت و انتقال سیستم به FPGA، با نوشتن نرم‌افزار مناسب، صحت عملکرد آن را تست کنید.

در طراحی رابط Master در نظر داشته باشید که سیگنال waitrequest می‌تواند توسط کنترلر SDRAM منتشر شود. در این حالت رابط Master باید تمامی سیگنال‌های کنترلی خود را ثابت نگه دارد تا زمانی که waitrequest غیر فعال گردد. مشابه بخش قبل، اسامی سیگنال های مربوط به رابط Master را با پیشوند avm_avalonmaster_[signal] وارد نمایید. شکل ۶ زمانبندی Avalon را برای انتقال خواندن و نوشتن به همراه سیگنال waitrequest نشان می‌دهد.



شکل ۶ زمانبندی Avalon برای انتقال خواندن و نوشتن به همراه سیگنال waitrequest.

۳-۲ طراحی مدار محاسبه‌ی دامنه

مطابق توضـــیحات بخش‌های قبل، مداری طراحی کنید که با دریافت آدرس بافر راســـت و چپ صـــدا، آدرس قرارگیری نتیجه‌ی عملیات، تعداد بازه‌های محاسبه‌ی دامنه و تعداد نقاط هر بازه، پس از نوشتن در رجیستر Go شـــروع به کار کند و مجموع قدر مطلق داده‌ها را در هر بازه محاســـبه کند و در مکان مربوطه در آدرس نتایج بنویسد. در انتهای کار سیگنال Done، ۱ می‌شود. عملکرد این سیستم باید دقیقاً مشابه بخش ۱ (بخش طراحی نرم‌افزاری) باشد. با انتقال سیستم طراحی شده به FPGA، سرعت عملکرد سخت‌افزار خود را نسبت به نرم‌افزار بسنجید و زمان اجرا را در هر دو حالت گزارش کنید. توجه نمایید که عملکرد و اعداد بدست آمده توسط نرم‌افزار و شتاب‌دهنده باید یکسان باشند.

بخش امتیازی (۱۰٪): رابط Master را به گونه‌ای طراحی کنید که امکان Burst داشـــته باشـــد. از Burstهای حداکثر ۱۶ تایی برای انتقال داده استفاده کنید. فاصله بین گرفتن سیگنال Go تا صدور سیگنال Done را در هر دو حالت با و بدون Burst با اضافه کردن یک کانتر اندازه بگیرید و باهم مقایسه نمایید.

نکات مهم:

۱) نیازی به گزارش تمامی مراحل انجام آزمایش نیســـت اما اهداف، کلیات، مقدار پارامترهای مختلف، مقایسه نتایج و سایر نکات مهم را حتماً در گزارش خود قید نمایید.
۲) پیروی از قالب خاصـــی در گزارش مد نظر نیســـت، اما ترجیحاً می‌توانید از قالب ارائه شـــده برای تکالیف کامپیوتری استفاده نمایید.
۳) آپلود فایل‌های شبیه‌سازی به همراه فایل گزارش ضروری است.

مراجع

[1] Altera, “Avalon Interface Specification”, Chapter 3: Avalon Memory-Mapped Interfaces, May 2007.

Available: http://www.altera.com/literature/manual/mnl_avalon_spec_1_3.pdf

موفق باشید

۹۷/۰۹/۲۴