بسمه تعالی

درس طراحی سیستم‌های نهفته مبتنی بر FPGA

آزمایش ۱: طراحی و پیاده‌سازی یک سیستم برای انتقال و پردازش اطلاعات

پردیس دانشکده‌های فنی دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
دکتر بیژن علیزاده

دستیاران آموزشی:
نگار آقاپور n.aghapour.s@gmail.com
فرید انجیدنی farid.anjidani@gmail.com
مهدی بحرینی mahdib25@gmail.com

پاییز ۱۳۹۸

مدت آزمایش: سه جلسه

اهداف آزمایش:

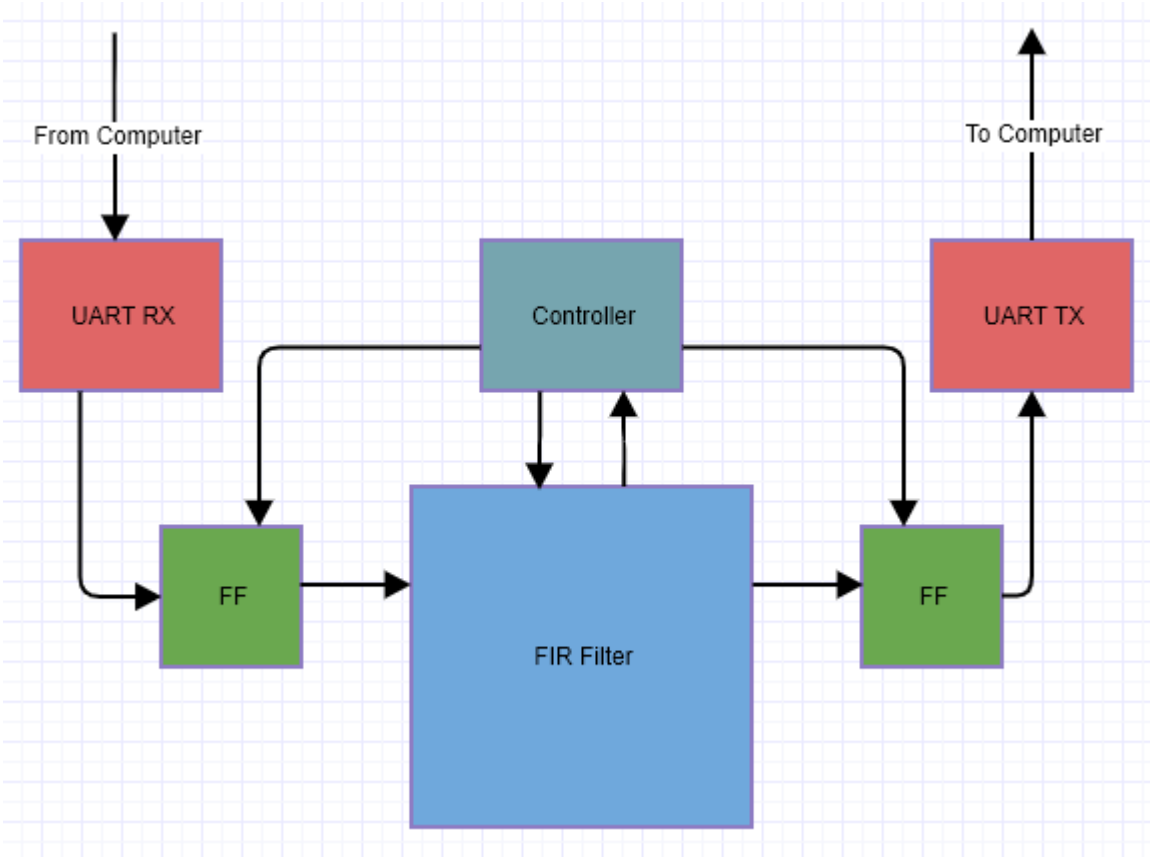
- ✓ آشنایی با برد DE2
- ✓ آشنایی با پورت سریال RS-232
- ✓ ارسال و دریافت سخت‌افزاری اطلاعات با PC
- ✓ ایجاد یک سیستم کامل دیجیتال با اتصال و کنترل چند ماژول مستقل
- ✓ آشنایی با سیگنال تپ
- ✓ آشنایی با PLL

مقدمه

در این آزمایش به طراحی یک سیستم کامل شامل واحد ارسال و دریافت داده، واحد پردازشی و واحد نمایش اطلاعات می‌پردازیم. اطلاعات از طریق PC و پورت سریال به سیستم منتقل می‌شود و پس از پردازش، نتیجه به PC ارسال می‌گردد. پردازش مورد نظر اعمال فیلتر دیجیتال بر روی نمونه‌های داده است.

شرح آزمایش

شمای کلی سیستم مورد نظر در این آزمایش در شکل ۱ آمده است. در این سیستم داده از طریق پورت سریال وارد شده و پس از پردازش (اعمال فیلتر FIR) از طریق پورت سریال به کامپیوتر داده می‌شود. یک کد C شامل توابع ارسال و دریافت داده به همراه گزارش آپلود شده است. هدف نهایی این آزمایش خواندن یک فایل صوتی، حذف نویز از آن به کمک فیلتر FIR و در نهایت ذخیره فایل خروجی است.



شکل ۱ شمای کلی آزمایش اول

مراحل زیر را به ترتیب انجام دهید:

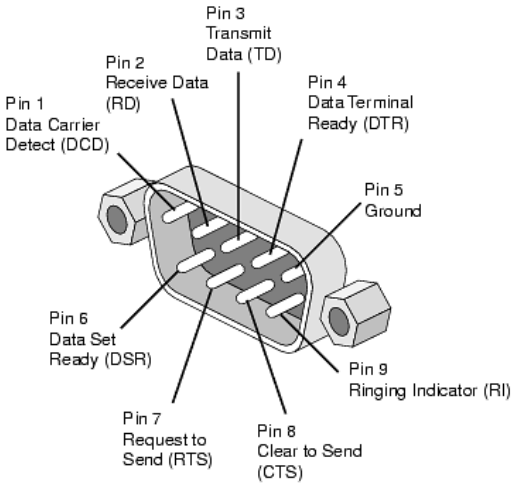
۱ـ ارتباط با کامپیوتر از طریق پورت سریال

پورت سریال یکی از ارتباطات خوب و مناسب کامپیوتر با دنیای بیرون به شمار می‌رود که اجازه می‌دهد اطلاعات به صورت دو طرفه انتقال یابد. این پورت با استاندارد RS-232 کار می‌کند. اغلب کامپیوترهای شخصی یک یا دو پورت COM برای واسط RS-232 دارند که معمولاً از یک کانکتور DE9 استفاده کرده و توانایی یک انتقال تمام دوطرفه (full-duplex) را دارند. لازم به ذکر است امروزه با توجه به نیاز به سرعت‌های انتقال بالاتر، پورت‌های سریال RS-232 جای خود را به سایر استانداردهای ارتباط سریال و خصوصاً USB داده‌اند. با این وجود نه‌تنها استفاده از پورت RS-232 هنوز منسوخ نشده است، بلکه در بسیاری از مواقع به دلایل اقتصادی و فنی استفاده از این پورت ترجیح داده می‌شود.

استاندارد RS-232 و به طور صحیح‌تر TIA/EIA-232، مشخصات الکتریکی، مکانیکی و عملکردی سیگنال‌های ارتباطی را مشخص می‌کند. این استاندارد اولین بار در سال ۱۹۶۲ برای استانداردسازی ارتباط میان کامپیوترها (DTE[1]ها)، مودم‌ها (DCE[2]ها) و ایجاد امکان اتصال دستگاه‌های تولید شده توسط

[1] Data Terminal Equipment
[2] Data Communicating Equipment

شرکت‌های مختلف مخابراتی وضع شد، اما کاربردهای گسترده‌تری خارج از این حوزه نیز یافت. در سال ۱۹۸۳ با انتشار کامپیوترهای شخصی توسط IBM، استاندارد RS-232 در این کامپیوترها در قالب کانکتور DE9 به کار گرفته شد. شکل ۲ کانکتور DE9 و شماره‌ی پین‌های آن را نشان می‌دهد.



شکل ۲ کانکتور DE9، پین‌ها و سیگنال‌های متناظر آن

پایه‌های نشان داده شده در شکل ۲ برای ارتباط با مودم استفاده می‌شود و در این آزمایش فقط از ۳ پایه‌ی مهم زیر استفاده می‌کنیم:

۱- پایه شماره ۲، RxD که همان داده سری دریافتی است.

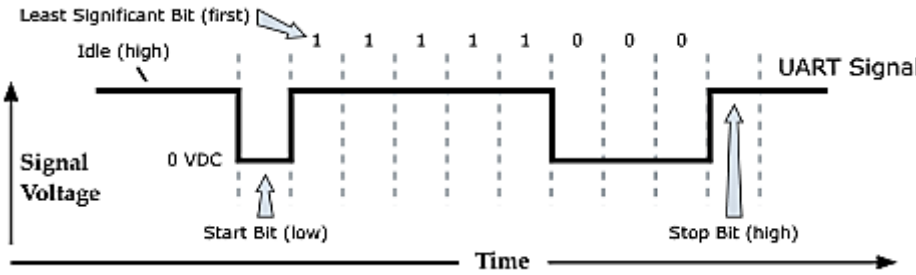
۲- پایه شماره ۳، TxD که همان داده سری ارسالی است.

۳- پایه شماره ۵، GND که پایه زمین می باشد.

در ارتباط سریال یک بیت داده در هر لحظه ارسال می‌گردد، پس یک خط برای انتقال در هر جهت کافی است، اما استاندارد RS-232 پروتکل ارتباطی مشخصی را تعیین نمی‌کند. با این وجود در این استاندارد معمولاً از پروتکل UART[3] استفاده می‌شود. در این پروتکل اطلاعات به صورت بسته‌های داده ۸ بیتی انتقال داده می‌شود به این صورت که نیز ابتدا بیت صفرام، سپس بیت ۱ ام و ... ارسال می‌گردند. توجه شود که این ارتباط به صورت آسنکرون می‌باشد، به این معنی که سیگنال کلاک به همراه داده ارسال نمی‌شود و قبل از شروع ارسال یا دریافت، فرستنده و گیرنده روی پارامترهای ارتباطی نظیر سرعت انتقال داده، فرمت داده و غیره بطور یکسان تنظیم می‌شوند. زمانی که انتقال داده‌ای روی خط ارتباطی نداریم، فرستنده روی خط مقدار ۱ قرار می‌دهد. فرستنده قبل از شروع ارسال هر بایت داده، یک بیت start (با مقدار ۰) روی خط قرار می‌دهد. بعد از بیت شروع، هشت بیت داده با سرعت و فرمتی که قبلاً برای هر دو طرف

[3] Universal Asynchronous Receiver Transmitter

تنظیم شده است، ارسال و دریافت می‌گردد. در انتها نیز فرستنده یک یا دو بیت stop (با مقدار ۱) به معنای اتمام ارسال ۸ بیت، روی خط قرار می دهد. به عنوان مثال شکل ۳ ارسال 0x1F را نشان می دهد.



شکل ۳ فریم داده‌ی 0x1F در پروتکل UART.

سرعت انتقال اطلاعات با baud rate مشخص می‌شود که نشان می‌دهد چند نمونه داده (در انتقال UART هر بیت یک نمونه محسوب می‌شود) در ثانیه ارسال شده است. برای مثال baud ۱۰۰۰ یعنی در ثانیه ۱۰۰۰ بیت انتقال می‌یابد. به عبارت دیگر انتقال هر بیت ۱ میلی‌ثانیه طول می‌کشد. کامپیوترهای شخصی معمولاً فقط سرعت‌های مشخصی از انتقال داده را پشتیبانی می‌کنند که برابر ۱۲۰۰، ۹۶۰۰، ۳۸۴۰۰ و ۱۱۵۲۰۰ baud است. در baud ۱۱۵۲۰۰ هر بیت ۸.۶ میکروثانیه طول می‌کشد و ارسال ۱۱ بیت (شامل یک بیت شروع، ۸ بیت داده و دو بیت خاتمه) حدود ۹۵.۵ میکروثانیه طول خواهد کشید.

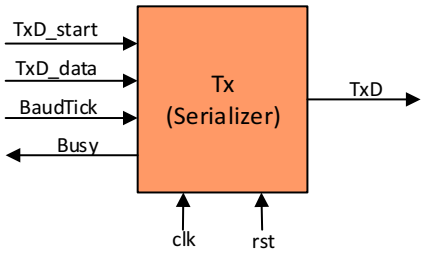
کد فرستنده‌ی UART در سایت درس بارگذاری شده است. شما در این آزمایش باید یک module نوشته که بتواند از طریق پورت RS_232 با کامپیوتر در ارتباط باشد. برای این کار مراحل زیر را انجام دهید:

گام ۱

ماژول BaudTickGen برای تولید نرخ baud ۱۱۵۲۰۰ را طراحی کنید. این ماژول با دریافت پالس ساعت ۵۰ مگاهرتز و شمارش آن به تعداد مورد نیاز، در هر ثانیه باید ۱۱۵۲۰۰ پالس تیک تولید کند. توجه نمایید خروجی ماژول، پالس ساعت نیست، بلکه در هر ثانیه از میان ۵۰ میلیون پالس ساعت در ۱۱۵۲۰۰ پالس مقدار خروجی این ماژول یک و در بقیه پالس‌ها صفر است.

گام ۲

ماژول async_transmitter، مطابق با شکل ۴ داده‌ی ۸ بیتی ورودی را با فعال شدن سیگنال TxD_start به صورت سریال از طریق خط TxD ارسال می‌کند. پارامترهای UART به صورت ۸ بیت داده، ۱ بیت خاتمه و بدون بیت parity انتخاب شده است (این تنظیمات معمولاً به صورت 8n1 نمایش داده می‌شود. بیت parity یک تک بیت است که می‌تواند جهت بررسی صحت دریافت داده در سمت گیرنده استفاده شود). فرستنده در حالتی که در حال ارسال داده است از TxD_start صرف نظر می‌کند، اما خروجی busy را در این مدت ۱ نگه می‌دارد.

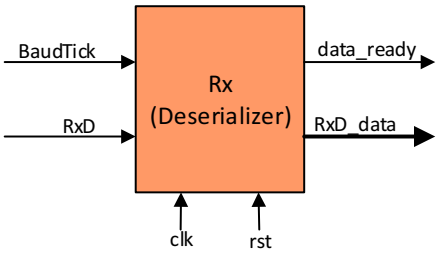


شکل ۴ ماژول فرستنده

صحت عملکرد ماژول Tx را با شبیه‌سازی تحقیق کنید. سپس TxD_data را بر روی سوئیچ‌های SW[7:0] تنظیم نمایید. در این بخش و بخش‌های بعدی این آزمایش از سیگنال کلاک 50 مگاهرتز استفاده نمایید. TxD_start را به KEY[0] متصل کنید. پس از مشخص کردن تخصیص پین‌ها[4]، عملکرد صحیح ماژول Tx را بر روی برد DE2 تحقیق کنید. پس از مشخص کردن تخصیص پین‌ها با استفاده از نرم افزار TeraTerm، RealTerm یا سایر نرم افزارهای ارتباط سریال، عملکرد صحیح ماژول TX را بر روی برد DE2 تحقیق کنید.

گام ۳

دیاگرام کلی ماژول گیرنده در شکل ۵ آمده است. در این ماژول داده از طریق خط RxD دریافت شده و به صورت بایت تبدیل می‌شود و روی باس RxD_data قرار می‌گیرد. هنگام آماده‌شدن داده، data_ready به مدت یک سیکل کلاک یک می‌شود.



شکل ۵ ماژول گیرنده

مسأله‌ای که می‌تواند مشکل‌ساز شود، دریافت آسنکرون داده‌ها است. گیرنده‌ی UART به منظور سنکرون شدن با داده ورودی آن را با نرخی بالاتر از نرخ انتقال داده نمونه‌برداری می‌کند (مثلاً چهار برابر نرخ baud) و پس از آن با نرخ baud از سیگنال ورودی نمونه‌برداری می‌کند. ماژول async_receiver وظیفه دریافت UART را بر عهده دارد. کلاک ورودی این ماژول دارای فرکانس ۴ برابر فرستنده است. به عبارت دیگر پارامتر oversampling در گیرنده ۴ می‌باشد. این پالس را با ماژول BaudTickGen با پارامتر oversampling = 4 تولید کنید. این کار منجر به شمارش متفاوت نسبت به حالت قبلی می‌گردد.

گام ۴

[4] Pin Assignment

در نهایت ماژولی با نام UART بنویسید که شامل ماژول‌های گیرنده، فرستنده باشد. علاوه بر تخصیص پین‌ها مشابه گام ۲، این ماژول هنگام دریافت داده آن را بر روی LEDهای قرمز رنگ نشان می‌دهد. مشابه بالا با استفاده از نرم افزار داده‌ای را بفرستید و صحت آن را چک کنید.

خواسته‌ها:

- توضیحات مربوط به عملکرد هر یک از ماژول‌ها گزارش شود.
- کد پیاده‌سازی سخت‌افزاری ماژول UART ارائه شود.

۲ – راه‌اندازی فیلتر FIR با استفاده از ضرایب ذخیره شده در حافظه (امتیازی)

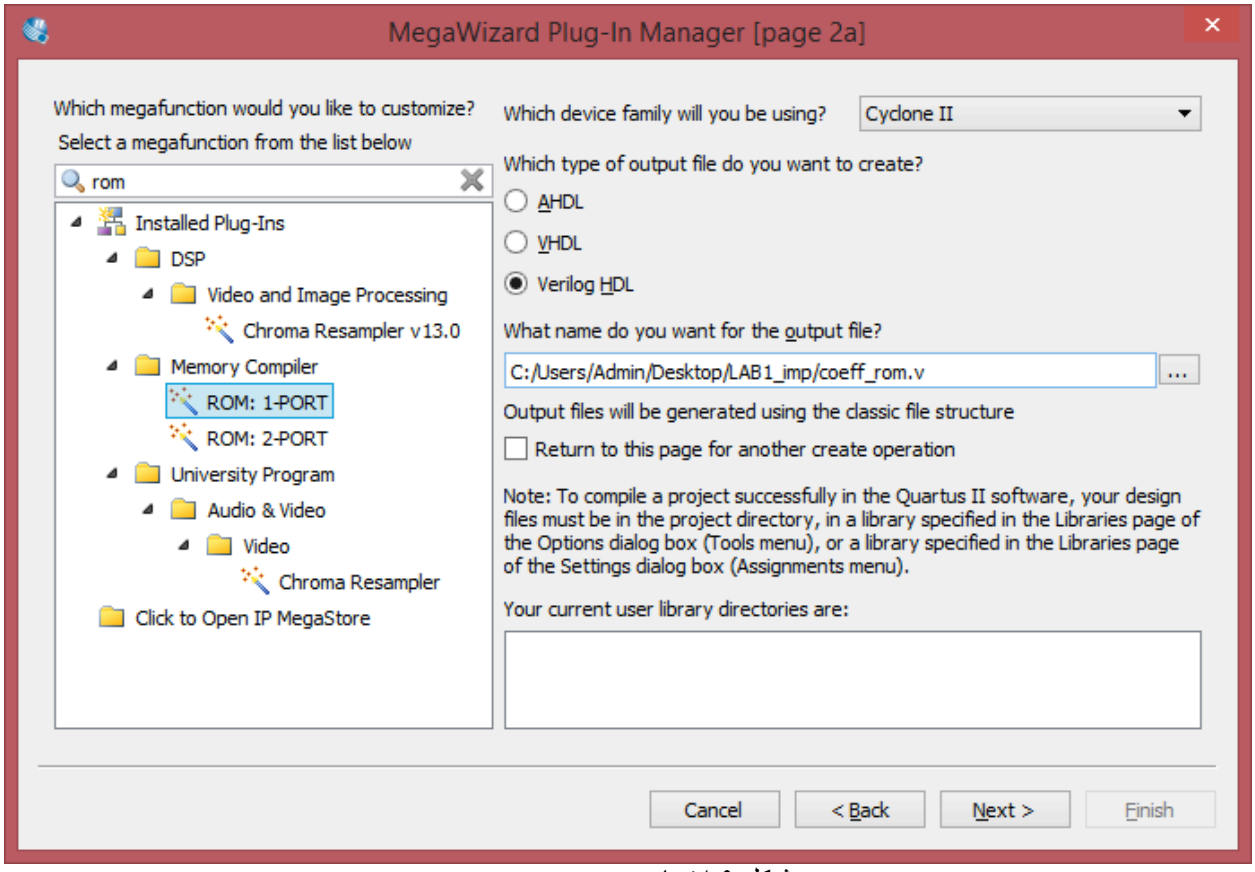
در این بخش فیلتر FIR طراحی شده در تکلیف کامپیوتری ۱ را با پارامترهای عرض بیت ۱۶ و طول فیلتر ۶۴ استفاده می‌کنیم. برای ذخیره سازی ضرایب میتوانید آن ها به صورت مستقیم در داخل ماژول FIR تعریف کنید. این بدین معنا است که بعد تعداد ضرایب رجیستر در نظر میگیرید. روش دیگر که نمره امتیازی دارد استفاده از بلوک‌های حافظه اختصاصی می‌باشد. برای پیاده‌سازی حافظه در یک FPGA هم می‌توان از LUT[5]های مورد استفاده در پیاده‌سازی لاجیک (روش اول) و هم از بلوک‌های حافظه‌ی تخصیص داده شده در FPGA استفاده کرد. در Cyclone II حافظه‌های اختصاصی از نوع بلوک‌های M4K هستند که هر کدام شامل ۴۰۹۶ بیت حافظه هستند. در این حافظه‌ها امکان پیکربندی با اندازه‌های مختلف (به صورت تعداد کلمات و عرض هر کلمه) وجود دارد که اصطلاحاً به آن Aspect Ratio گفته می‌شود. برای این کار گام‌های زیر را به ترتیب انجام دهید. (توجه برای روش ۱ نیازی به انجام کاری نیست صرفا به تعداد ضرایب و عرض بیت مشخص رجیستر در داخل ماژول FIR تعریف میکنید و آن ها مقدار دهی اولیه میکنید.)

گام ۱

در گام اول یک حافظه‌ی ROM برای ذخیره‌سازی ضرایب ایجاد می‌کنیم و آن را تست می‌کنیم. بدین منظور مراحل زیر را انجام دهید:

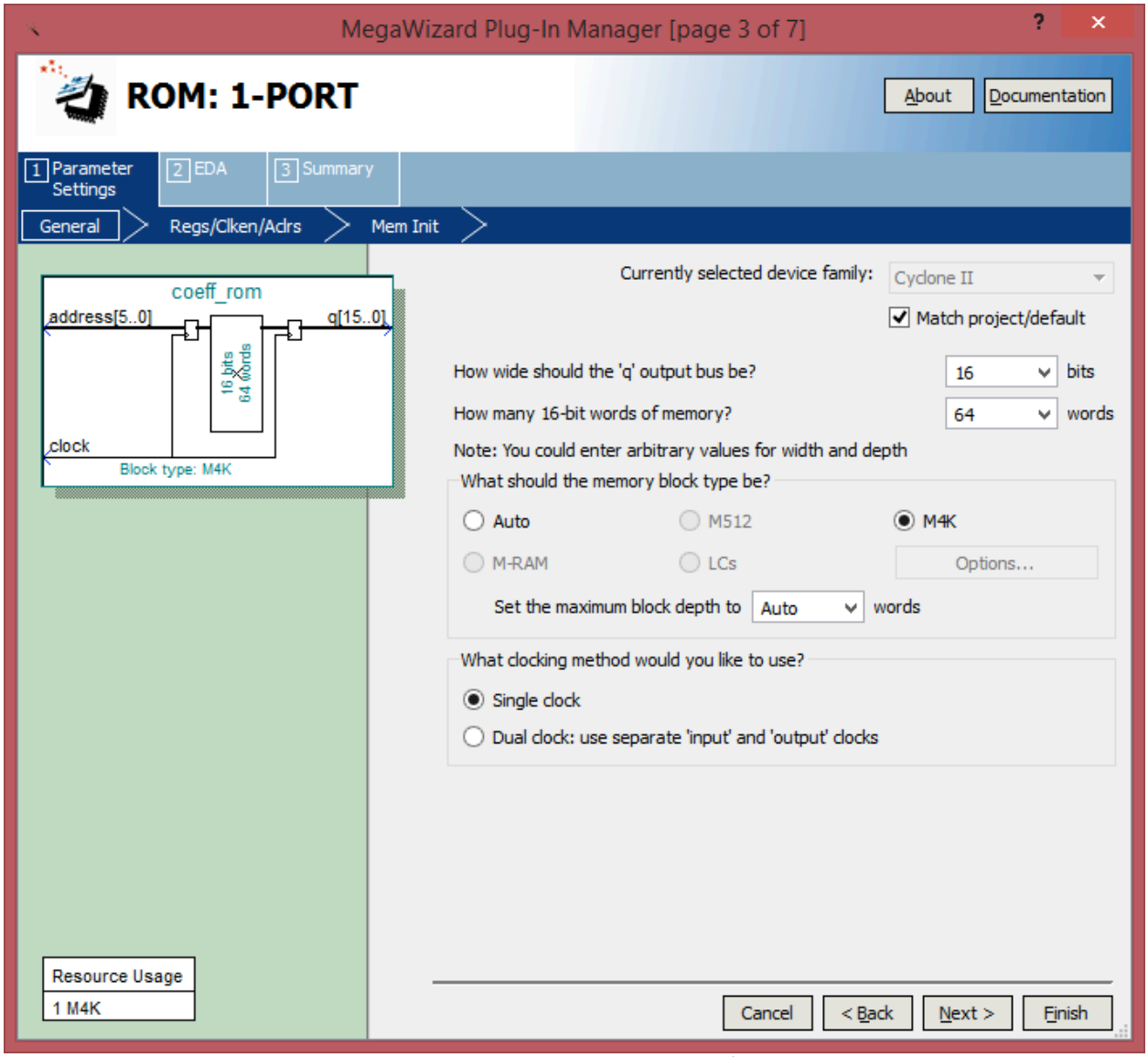
۱) در قسمت Tools > MegaWizard Plug-In Manager ایجاد یک megafunction جدید را انتخاب کنید. در صفحه‌ی دوم در قسمت Select a megafunction from the list below، از گروه Memory Compiler، حافظه‌ی ROM: 1-PORT را انتخاب کنید (شکل ۶). مطمئن شوید فایل خروجی با فرمت Verilog انتخاب شده است.

[5] Look-Up Table



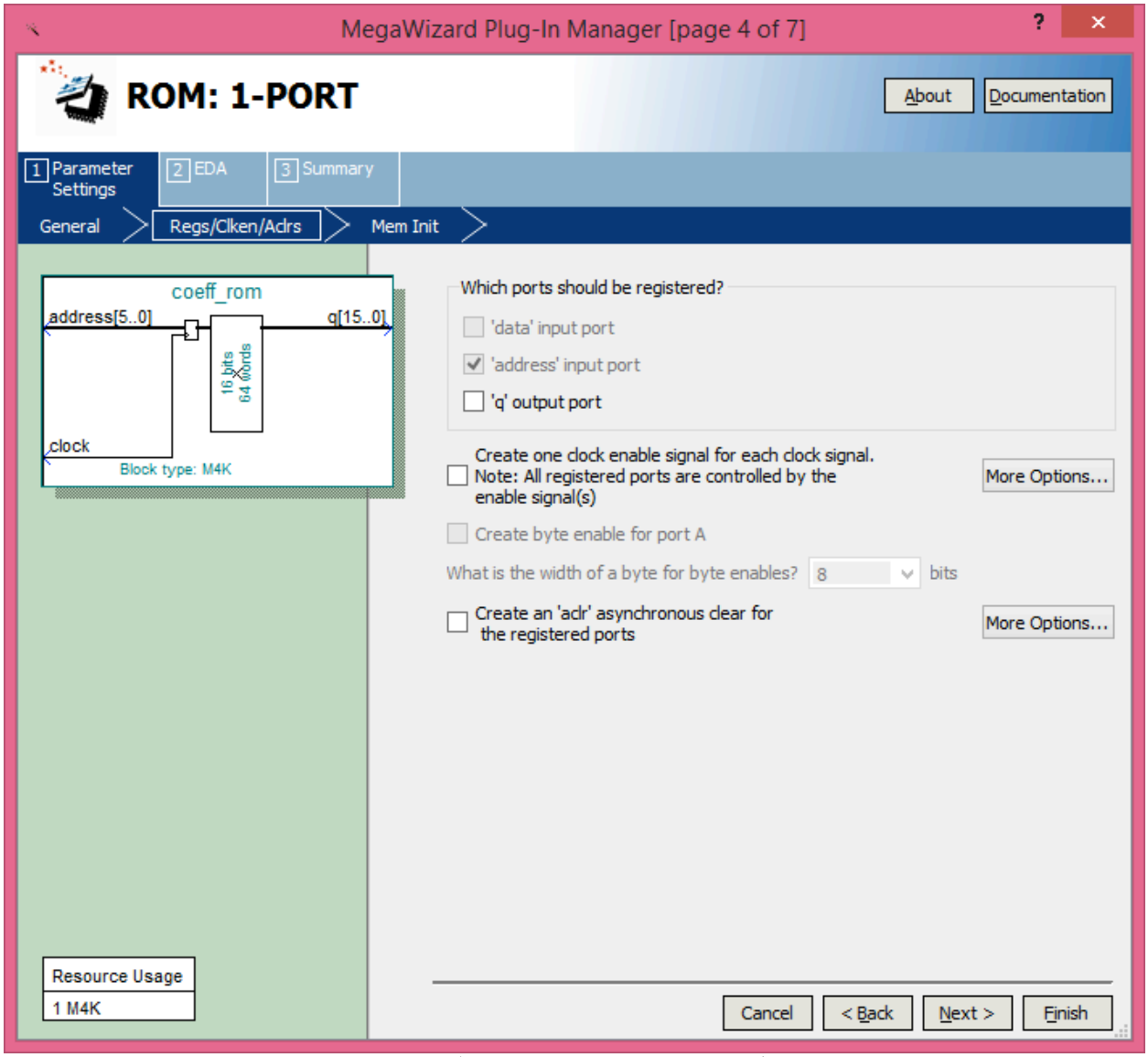
شکل ۶ انتخاب ROM: 1-PORT.

۲) در صفحه‌ی بعد ساختار حافظه را به صورت ۶۴ کلمه‌ی ۱۶ بیتی انتخاب کنید. نوع حافظه را نیز M4K قرار دهید (شکل ۷).



شکل ۷ انتخاب اندازه‌ی حافظه‌ی ضرایب.

۳) در صفحه‌ی چهارم ‘q’ output port را از حالت انتخاب خارج سازید تا رجیستر خروجی حذف گردد. استفاده از رجیستر در ورودی بلوک M4K به دلیل سنکرون بودن این بلوک‌ها اجباری است.



شکل ۸ پیکربندی ورودی/خروجی بلوک حافظه.

۴) ضرایب فیلتر FIR در قالب یک فایل MIF[6]. (و فایل مشابهی با فرمت hex) به همراه دستور کار در سایت قرار گرفته است. در صفحه‌ی بعد این فایل را انتخاب کنید.

۵) برای بقیه‌ی تنظیمات از مقادیر پیش‌فرض استفاده کنید.

۶) به منظور تست فایل ایجاد شده، آن را در یک ماژول نمونه‌گیری کنید. پورت address را به سوئیچ‌ها و پورت q (خروجی) را به LEDها متصل کنید. پس از پروگرام کردن FPGA فایل MIF (یا فایل معادل hex) را باز کرده و از تطبیق مقادیر بر این فایل اطمینان حاصل کنید.

[6] Memory Initialization File

گام ۲

با اعمال تغییراتي در کد تکلیف کامپیوتری ۱ خود، حافظه تولیدی را با حافظه موجود در کد تکلیف کامپیوتری ۱ جایگزین کنید. با استفاده از تست‌بنچ مورد استفاده در تکلیف کامپیوتری ۱ از صحت عملکرد فیلتر مطمئن شوید.

نکته مهم: حافظه تولید شده داده را یک کلاک بعد از درخواست آن به شما تحویل مي‌دهد. در صورت نیاز، حتماً طرح فیلتر خود را متناسب با این موضوع تغییر دهید.

۳ـ راه‌اندازی و تست سیستم کلی

در این بخش سیستم کلی پیاده‌سازی و تست خواهد شد. در حال حاضر تمامی اجزای شکل ۱ را به غیر از واحد Controller در اختیار داریم. مراحل زیر را انجام دهید تا واحد کنترلر نیز آماده گردد:

گام ۱

واحد کنترلر را با یک ماشین حالت پیاده‌سازی کنید به طوری که فرآیند زیر را کنترل کند:

۱) در حالت بیکاری، کنترلر منتظر می‌ماند تا داده‌ای از ورودی دریافت شود. هر وقت واحد گیرنده (Rx) سیگنال data_ready را منتشر کرد، کنترلر باید داده را به فیلتر FIR انتقال داده و سیگنال کنترلی input_valid را به مدت یک سیکل فعال کند. چون محاسبات به صورت ۱۶ بیتی انجام می‌شود، باید ساختاری پیاده کنید که گیرنده پس از دو بار دریافت داده ۸ بیتی، ورودی فیلتر را فعال کند.

۲) سپس کنترلر منتظر می‌ماند تا محاسبات فیلتر FIR تمام شده و output_valid فعال گردد. کنترلر باید داده خروجی ۱۶ بیتی را در قالب دو داده ۸ بیتی بفرستد و برای این کار باید از طریق سیگنال‌های کنترلی TxD_start و busy با واحد فرستنده (Tx) در ارتباط باشد.

۳) پس از اتمام ارسال دو داده‌ی ۸ بیتی کنترلر به حالت اولیه باز می‌گردد و منتظر دریافت ورودی بعدی می‌شود.

گام ۲

حال تمامی اجزای سیستم آماده است. در ابتدا برای یک داده از صحت کار مدار مطمئن شوید. این بدان معنی است که دوتا داده ۸ بیتی به وسیله نرم‌افزار (نرم‌افزار استفاده شده برای UART) فرستاده میشود و سپس باید جواب درست به نرم افزار برگردد. پس از آن کد متلبی که در اختیارتان قرار گرفته است را به

دقت مطالعه کنید و عملکرد آن باید در گزارش شرح داده شود. سپس با استفاده از کد مطلب درستی سیستم کد را تحقیق کنید. باید خروجی فیلتر کاملا بدون نویز باشد.

برای تست سیستم می‌توانید از signal tap استفاده کنید. زمانی که سیستم دچار مشکل است و خروجی درستی ندارد، یکی از راه‌های رفع مشکل، استفاده از برنامه‌ای است که بتوان به وسیله آن، سیگنال‌های داخل FPGA را مشاهده کرد. به طور کلی برای مشاهده مقدار سیگنال‌های داخلی، از نرم افزار های متفاوتی (analyzer) استفاده می‌شود. نرم‌افزار مورد استفاده در Quartus II ، نرم‌افزار signal tap است. شیوه کار بدین صورت است که یک سیگنال توسط کاربر به عنوان مبنا معرفی می‌شود و با توجه به این سیگنال، از وضعیت سایر سیگنال‌های مشخص‌شده نمونه‌برداری می‌شود. این کار برای پیدا کردن محل خطا بسیار مناسب است. نحوه کار با signaltap در سایت درس بارگذاری شده‌است. در صورت نیاز به اطلاعات بیشتر می‌توانید از منابع موجود در سایت شرکت اینتل استفاده کنید.

خواسته‌ها:

- کد پیاده‌سازی سخت‌افزاری ماژول تولیدکننده baud را ارسال نمایید.
- کد پیاده‌سازی سخت‌افزاری ماژول گیرنده را ارسال نمایید.
- کد پیاده‌سازی سخت‌افزاری سیستم کلی را ارسال نمایید.

۴ ـ نکات پیشرفته‌تر اما اساسی

گام ۱

برای آنکه ابزار سنتز بتواند زمان‌بندی مسیرهای ترکیبي بین فلیپ‌فلاپ‌های مدار را به درستي در هنگام سنتز و جایابي مدار انجام دهد، نیاز به دانستن فرکانس کلاک دارد. کلاک ورودی به مدار شما فرکانس ۵۰ مگاهرتز دارد. (یک constraint file با پسوند sdc. اضافه کنید که در آن ذکر کنید که فرکانس کلاک ورودی ۵۰ مگاهرتز مي‌باشد.) برای این کار مي‌توانید از رابط گرافیکي Quartus کمک بگیرید. بدین منظور از منوی Assignments گزینه ...TimeQuest Timing Analyzer Wizard را انتخاب کنید. در بخش Clock مي‌توانید محدودیت مد نظر خود را اعمال کنید. محدودیت اعمال شده را در فایل sdc. تولید شده مشاهده و گزارش نمایید و آن را توضیح دهید .چه محدودیتهای زماني دیگری را مي‌توان توسط این Wizard مشخص کرد؟

گام ۲

علاوه بر بلوک‌های ضرب‌کننده و حافظه، FPGA ها جهت تسریع الگوریتم‌ها ساختارهای دیگری نیز به کار مي‌برند از جمله این ساختارها PLL ها (Phased Locked Loop) هستند. به کمک این ساختارها مي‌توان با داشتن یک کلاک با فرکانس مشخص، کلاکي با فرکانس دیگری تولید کرد. یک PLL به مدار خود اضافه کنید و از کلاک ۵۰ مگا هرتز ورودی به FPGA کلاک مورد نظر خود را تولید کنید. مشابه حافظه‌ها از مسیر Tools و زیرمنوی MegaWizard Plug-In Manager میتوانید PLL را به صورت یک ماژول اضافه کنید. مدار خود را به کلاک جدید تولید شده متصل کنید. تلاش کنید بیشترین فرکانس ممکن را از مدار به دست آورید. (با روش‌هایي که در تمرین کامپیوتری اول گفته شد.)

فراموش نکنید که ماژولهای فرستنده و گیرنده UART به فرکانس کلاک ورودی حساس هستند. دقت نماییدکه شمارنده‌های این ماژول را با تغییر پارامتر ماژولهای Buad_Rate_Generator به درستي تغییر دهید. همچنین جهت اطمینان از قفل شدن حلقه PLL سیگنال lock خروجي آن را به یک LED متصل نمایید. بررسي کنید که آیا نیازی به اضافه کردن فرکانس جدید مدار(فرکانس خروجي) PLL به constraint فایل میباشد یا خیر؟(چرا؟)

گام 3

Routing کلاک معمولاً با بقیه سیگنال ها متفاوت است (چرا؟). بنابراین باید به ابزار سنتز گفته شود که کدام سیگنال کلاک می باشد که آن را بر روی مسیرهای ازپیش مشخص کلاک در FPGA سنتز نماید.

این کار در Quartus II به صورت خودکار انجام می شود (این ابزار سنتز چگونه کلاک را شناسایی می‌کند و این کار را انجام می‌دهد؟ اگر کلاک های شناسایی شده در مدار بیشتر از مسیرهای از پیش مشخص شده در FPGA باشد با چه منطقی مناسب‌ترین سیگنال‌ها را برای انتقال روی شبکه‌های کلاک شناسایی می‌کند؟)

حال شما باید این کار را دستی انجام دهید. برای این کار باید خودکار Quartus II را خاموش کنید. (چگونه؟)

سپس کلاک مورد نظر (کلاک خروجی PLL) را به عنوان کلاک معرفی کنید. (آیا کلاک ورودی ۵۰ مگاهرتز باید روی مسیر مشخص کلاک Route شود؟ چرا؟). جهت راهنمایی به مرجع [1] و [2] مراجعه کنید. در بعضی از FPGA ها با توجه به محدودیت‌های موجود در طراحی باید این کار را انجام داد. مثلاً اگر شما کلاک خود را به ۵ PLL متصل کنید شاید نیاز باشد که این کار را بکنید. چرا این کار لازم است؟

نکات مهم:

۱) رعایت استایل کدنویسی ارائه شده در کلاس ضروری است.

۲) نیازی به گزارش تمامی مراحل انجام آزمایش نیست اما اهداف، کلیات، مقدار پارامترهای مختلف، شرایط درستی‌سنجی، نتیجه‌ی درستی‌سنجی و سایر نکات مهم را حتماً در گزارش خود قید نمایید.
۳) پیروی از قالب خاصی در گزارش مد نظر نیست، اما ترجیحاً می‌توانید از قالب ارائه شده برای تکالیف کامپیوتری استفاده نمایید.
۳) آپلود فایل‌های شبیه‌سازی به همراه فایل گزارش ضروری است.

مراجع

[1] “How can I assign a PLL output clock to a Global Clock Network?” Online: https://www.altera.com/support/support-resources/knowledge-base/solutions/rd03182011_985.html

[2] “How do I assign a clock in my design to use specific global, regional, dual-regional, or periphery clock networks?” Online: https://www.altera.com/support/support-resources/knowledge-base/solutions/rd09282011_171.html

موفق باشید
۹۸/۷/۱۹