

مدلسازی کنترلر USB توسط VHDL و پیاده سازی آن بر FPGA

حامد نعمتیان

دانشگاه زنجان

۰۲۱ - ۸۵۱۶۲۱۳

H_nematian_n@yahoo.com

مهندس علی آذرپیوند

دانشگاه زنجان

مدیریت پژوهشی دانشکده فنی و مهندسی

Azarpeyvand@mail.znu.ac.ir

چکیده: در این مقاله هدف نشان دادن طریقه نوشتن کد VHDL مربوط به کنترلر USB و سنتز آن بر روی FPGA میباشد. با توجه به روی آوردن تکنولوژی و فن آوری جدید در استفاده از FPGA به دلیل فرکانس کاری و قابلیت اطمینان بالای آن، و نظر به اینکه کارتهای USB Controller موجود در بازار از میکروکنترلرها استفاده می کنند که دارای محدودیت های کاربردی هستند، درصدد شبیه سازی کنترلر USB و پیاده سازی آن بر روی FPGA برای انعطاف پذیرتر کردن این استاندارد برآمدیم. در این راه موفق به استخراج پروتکل USB، نوشتن کدهای VHDL مربوطه، شبیه سازی، آماده سازی جهت سنتز بر FPGA و تست هریک از مراحل گشتیم و به نتایج مطلوبی نیز دست یافتیم.

کلمات کلیدی: کنترلر USB، کد VHDL، شبیه سازی، سنتز بر روی FPGA

۱- مقدمه

USB (Universal Serial Bus) یک مدار واسط سریع و قابل انعطاف برای اتصال وسایل جانبی به کامپیوتر می باشد. اختراع و استاندارد گذاری آن در سال ۱۹۹۵ توسط یک گروه کامپیوتری و کارخانجات وابسته به آن صورت گرفت. هدف از طراحی USB در آن زمان بکار گیری تمام فضای پورت و باس سریال و به روز رسانی آن با تکنولوژی قرن بیست و یکم بود. البته این مطلب صحت دارد که استانداردهای ارتباطی بسیاری بین کامپیوتر و پیرامون آن وجود دارد، اما هدف این گروه ساختن تکنولوژی ای بود که اکتیویته سرعت بالا و پایین باس را ترکیب کرده و دسترسی مشترکی برای هر دو سرعت فراهم کند - یک تکنولوژی با یک پروتکل قدرتمند برای پیکربندی اتوماتیک وسایل جانبی و یک باس سریال که ارتباط و وصل شدن به آن ساده و آسان بوده و در آن کاربر نیاز به پیکربندی های سخت

افزایی و نرم افزاری نداشته باشد- این اقدامات منجر به پیدایش USB گردید. USB مشخصا شامل سه بخش می باشد : Host ، Root Hub و Device . بخش اول به کامپیوتری اطلاق می شود که در حال برقراری ارتباط از طریق پورت USB به کمک Root Hub تعبیه شده بر مادربرد می باشد . بخش سوم و مورد نظر ما در این مقاله Device است و به وسیله ای گفته می شود که خواهان برقراری ارتباط با کامپیوتر از طریق پورت USB می باشد . Device برای این منظور نیاز به یک کنترلر به نام USB Controller دارد تا بتواند اطلاعات خود را به فرمت USB تبدیل و ارسال نماید و اطلاعات رسیده از کامپیوتر را نیز به فرمت قابل فهم برای خود درآورد ، همچنین قادر به پاسخگویی به کامپیوتر در جهت شناسایی Device مورد نظر باشد . امروزه در بازار، کارتهایی جهت تحقق این امر یافت می شود ، ولیکن استفاده از میکروکنترلرها و اتصالات خارجی در آنها مشکلاتی اعم از فرکانس کاری پایین ، نویزپذیری بالا ، اشغال فضای نسبتا زیاد و انعطاف پذیری کم را ایجاد کرده است . استفاده از FPGA این مزیت را دارد که از مشکلات فوق تا حد مطلوبی به گونه ای که در بخشهای زیر توضیح خواهیم داد، می کاهد و زمینه ارتقای آنها فراهم می آورد . در این راستا به تحقیق راجع به پروتکل USB پرداخته ایم و یکی از انتقالهای ضروری در این استاندارد را به نام انتقال کنترلی انتخاب کرده و مورد مطالعه قرار داده ایم . در این مقاله با اقدامات صورت گرفته در جهت نوشتن کدهای VHDL مربوط به پروتکل USB ، شبیه سازی ، تست و پیاده سازی آن بر FPGA آشنا می شویم .

۲- مشخصات کنترلر مورد نظر

هدف ما در این مقاله ، طراحی کنترلر برای وسایل جانبی است که از کلاس HID پیروی می کنند . سرعت ساپورت شده توسط این کنترلر مطابق با استاندارد سرعت پایین 1.5 Mbps است . از دیگر خصوصیات وسایل جانبی مرتبط با این کنترلر ویژگی Self Powered ، Remote-Wakeup آنها می باشد . انتقال ساپورت شده توسط این کنترلر از نوع انتقال کنترلی می باشد و یک بافر جهت ذخیره داده های ارسالی و دریافتی در نظر گرفته شده است . بسته به کاربرد قابلیت اضافه کردن چهار بافر دیگر مشابه با این بافر وجود دارد .

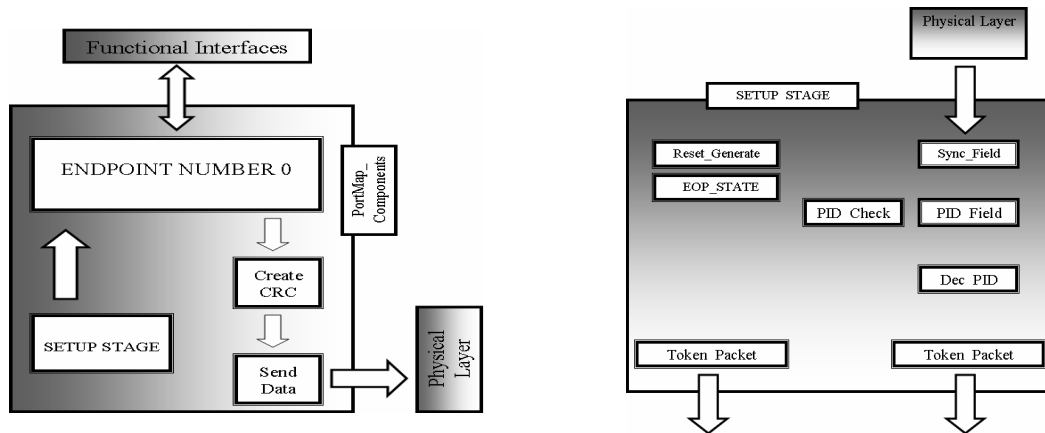
۳- از شبیه سازی تا سنتز

برای نیل به اهداف یاد شده قبل از نوشتن کدهای VHDL ، بایستی بر قوانین انتقال در استاندارد USB تسلط کامل پیدا کرد . ادامه کار را به کمک سه نرم افزار ModelSim ، Leonardo و MaxPlus بررسی می کنیم .

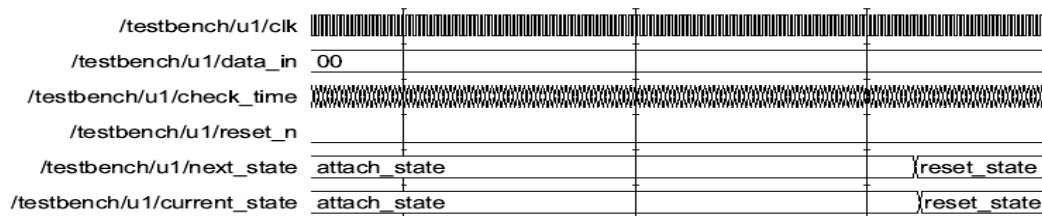
۳-۱ شبیه سازی در محیط ModelSim

در تصویر شماره ۱ ساختارهایی را که به منظور طبقه بندی ، برای مراحل انتقال در نظر گرفته ایم نشان داده شده است . برای هر فاز انتقال اطلاعات ، کدهای VHDL مربوط را به صورت یک کامپوننت در می آوریم . در ابتدا هر کامپوننت در محیط نرم افزار ModelSim شبیه سازی شده است . شکل موجهای بدست آمده از این شبیه سازی را بررسی می کنیم تا از صحت عملکرد هر کامپوننت مطمئن شویم . در ذیل تعدادی از شکل موجهای حاصل از شبیه سازی آورده شده است .

تصویر شماره ۲ مربوط به کامپوننت تشخیص اتصال و ریست شدن Device می باشد . بررسی در مدت زمان 10 ms انجام شده است و باعث فشردگی سیگنالهای نمایشی گردیده است .



تصویر ۱: ساختار کلی و طریقه ارتباط کامپوننت های موجود



تصویر ۲: شکل موج تست کامپوننت ریست

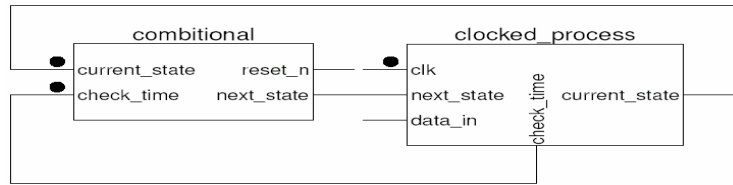
تصویر شماره ۳ دریافت ۸ بایت دیتا در مرحله Setup و پردازش CRC ورودی را نمایش می دهد. این عمل با موفقیت صورت پذیرفته و بایتهای استخراجی برای ارسال به واحد بعدی آماده گشته اند .



تصویر ۳: شکل موج دریافت ۸ بایت دیتا در مرحله Setup

۳ - ۲ سنتر در محیط Leonardo

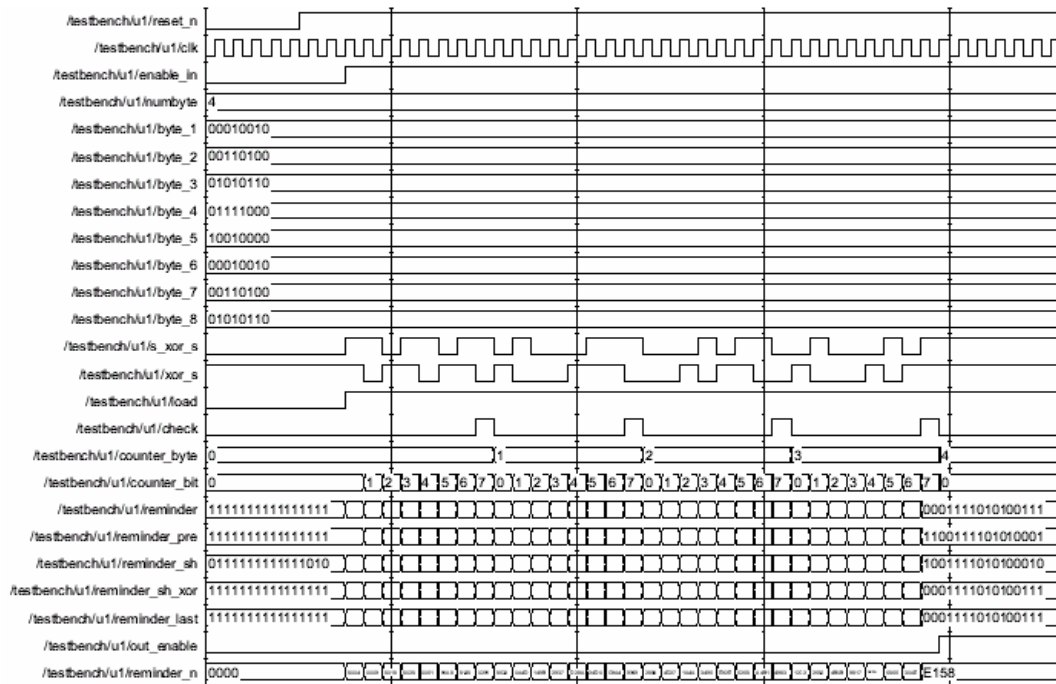
پس از شبیه سازی و بدست آوردن نتایج مورد نظر ، کدهای VHDL نوشته شده را برای سنتر در محیط نرم افزاری Leonardo فراخوانی می کنیم . عمل سنتر را بر FPGA نوعی EPLK30QC208-3 از شرکت Altera انجام می دهیم . با بررسی تعداد Logic cells مصرفی ، حداکثر فرکانس کاری و حداکثر تاخیر گیت مجاز که از مطالعه نتایج سنتر بدست می آید ، به ویرایش دوباره کدها به منظور بهینه سازی مصرف Logic cells و بالا بردن فرکانس کاری می پردازیم . پس از انجام بهینه سازی لازم ، مجدداً شبیه سازی و سنتر را تا حصول نتایج مطلوب ادامه می دهیم ، تا به کدهایی مناسب از بعد فرکانس کاری و تاخیر گیت مجاز و همچنین از لحاظ تعداد Logic cells مصرفی برسیم . در تصاویر زیر برخی از این کامپوننتها و اتصالات نشان داده شده است .



تصویر ۴ : استیت ماشین مربوط به حالت ریست

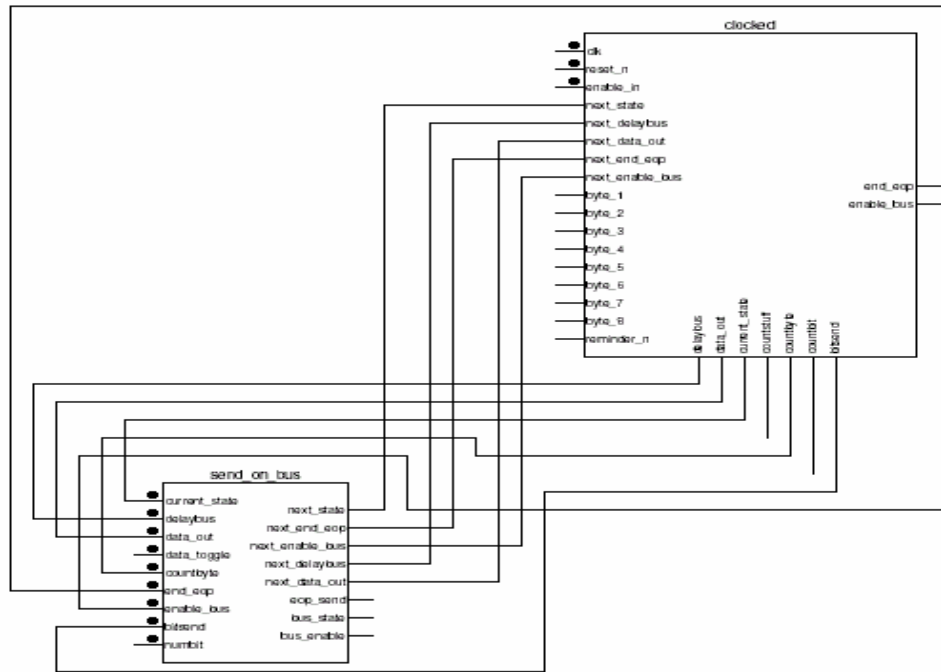
تصویر شماره ۴ استیت ماشین مربوط به کامپوننت ریست می باشد . در ساختار این کامپوننت ۲۹ عدد Logic cells معادل ۱/۰۸٪ کل Logic cells موجود بکار رفته است و فرکانس تا حد 85 MHz را پاسخگوست .

تصویر شماره ۵ مربوط به واحد CRC می باشد ، که تعداد ۱۰۳ Logic cells لازم دارد . این کامپوننت تا فرکانس 62.1 MHz را پاسخگوست .



تصویر ۵ : استیت ماشین مربوط به CRC

تصویر شماره ۶ مربوط به واحد ارسال دیتا می باشد که تعداد ۲۰۴ Logic cells مصرفی و فرکانس کاری تا 47 MHZ از مشخصه های آن می باشد .



تصویر ۶ : استیت ماشین مربوط به واحد ارسال دیتا

۳-۳ سنتز توسط MaxPlus

برای داشتن فایل‌هایی با پسوند VHO لازم است فایل‌های بدست آمده از Leonardo با پسوند edf در Maxplus فراخوانی کنیم . این فایل‌ها شامل گیت‌های بکار رفته در سنتز می باشد که تاخیر های واقعی گیت‌ها در آن اعمال شده است . این فایل‌ها را مجدداً در محیط ModelSim شبیه سازی می کنیم . با مشاهده تفاوت میان شکل موج‌های شبیه سازی فایل VHO و شکل موج‌های بدست آمده از فایل VHD به لزوم تصحیح کدهای VHDL اولیه پی می بریم . لازم است مراحل فوق را تا حصول نتایج مطلوب تکرار کنیم . پس از اینکه تست تمامی کامپوننت‌های موجود با موفقیت انجام شد ، مرحله انتهایی کار فرا می رسد . در این مرحله کامپوننت‌ها را در فایل جدیدی توسط یک کد VHDL ، Port Map می نماییم . فایل حاضر قادر به شناساندن Device به کامپیوتر ، دریافت و ارسال دیتا و همچنین مدیریت انتقال اطلاعات بین وسایل جانبی و کامپیوتر می باشد ؛ و می توان آنرا بر روی FPGA وسیله جانبی پیاده کرد .

تصویر شماره ۷ نمونه ای از شکل موج‌های مربوط به انتقال دیتا میان کامپیوتر و وسیله جانبی را نشان می دهد . بنا به درخواست کامپیوتر ، وسیله جانبی اطلاعاتی را به کامپیوتر فرستاده است . این اطلاعات بدون هیچ خطایی دریافت می گردد .



تصویر ۷: شکل موج حاصل از یک انتقال نمونه اطلاعات

۴ - نتیجه گیری و پیشنهادات

یکی از مزایای استفاده از FPGA که مهمترین برتری آن نسبت به میکروکنترلرهای موجود محسوب می شود ، عدم نیاز به سیم کشی و اتصالات خارجی در طراحی مدارها می باشد . از سوی دیگر با توجه به فرکانس کاری بالای آن در ضمن داشتن پالسهایی با لبه تیز ، قابلیت ارتقای سرعت ارسال و دریافت داده ها وجود دارد . تمامی این مزایا به کاهش حجم و هزینه ،افزایش قابلیت اطمینان ونسبت سیگنال به نویز وهمچنین انعطاف پذیری بیشتر آن می انجامد. با شبیه سازی های انجام شده کد VHDL ای که قابل پیاده سازی بر روی FPGA است بدست آمد .

با توجه به روند سریع توسعه تکنولوژی و نیاز به رابطهای سریعتر ، کدهای بدست آمده برای نسخه USB1.1 را در کاربردها ، با سرعت و دقت بالاتر به نسخه USB2.0 می توان ارتقا داد . همچنین بسته به کاربرد ، سه نوع دیگر انتقال - وقفه ای ، توده ای ، سنکرون - را می توان به مجموعه کدهای فوق افزود .

۶ - مراجع

[۱] ظریف ، پورت USB ، چاپ اول مرداد ۸۳ ، انتشارات نگین دانش ، ۱۳۸۳

[2] Jan Axelson , " USB Complete " , Second Edition , LakviewResearch 5310 Chinook Ln. , Madison , WI 53704 , USA

[3] Steven Mcdowell & Martin D.Seyer,"USB Explained" ,1999

[4] Intel , Compaq , Microsoft , NEC Corporation , "Universal Serial Bus Specification " , Revision1.1, SEP.23 1998