
بسم الله الرحمن الرحيم

دانشکده : برق و رباتیک

گروه : الکترونیک

موضوع:

طراحی، شبیه سازی و پیاده سازی سخت افزاری الگوریتمی جهت فشرده سازی بلادرنگ ویدیو در بستر FPGA

دانشجو : میلاد بزرگمهر

اساتید راهنما : دکتر هادی گرایلو

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

بهمن ۹۲

دانشگاه صنعتی شاهرود

دانشکده : برق و رباتیک

گروه : الکترونیک

پایان نامه کارشناسی ارشد آقای میلاد بزرگمهر

تحت عنوان: طراحی، شبیه سازی و پیاده سازی سخت افزاری الگوریتمی جهت فشرده سازی بلادرنگ ویدیو در بستر FPGA

در تاریخ .... توسط کمیته تخصصی زیر جهت اخذ مدرک کارشناسی ارشد مورد ارزیابی و با درجه ..... مورد پذیرش قرار گرفت.

اساتید راهنما	امضاء
نام و نام خانوادگی :	

اساتید داور	امضاء	نماینده تحصیلات تکمیلی	امضاء
نام و نام خانوادگی :		نام و نام خانوادگی :	
نام و نام خانوادگی :			
نام و نام خانوادگی :			

خداوند بزرگ را شاکرم که لطف خود را شامل حال من نمود تا بتوانم تحقیق خود را به پایان برسانم و گامی هر چند اندک، در راه توسعه

علمی ایران عزیز بردارم.

همچنین از زحمات جناب آقای دکتر گرایلو استاد محترم راهنما که با هدایت و حمایت های بی دریغشان یاری ام نمودن بسیار

سپاسگزارم.

و در پایان از زحمات خانواده خوبم و دوستان عزیزم و سایر کسانی که در تدوین این تحقیق مرا یاری نمودند تشکرم.

تعهد نامه

اینجانب میلاد بزرگمهر دانشجوی دوره کارشناسی ارشد رشته مهندسی برق – گرایش الکترونیک (دیجیتال) دانشکده برق و رباتیک دانشگاه صنعتی شاهرود نویسنده پایان نامه **طراحی، شبیه سازی و پیاده سازی سخت افزاری الگوریتمی جهت فشرده سازی بلادرنگ ویدیو در بستر FPGA** تحت راهنمائی دکتر هادی گرایلو متعهد می شوم.

- **تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .**
- **در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است .**
- **مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است .**
- **کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « دانشگاه صنعتی شاهرود» و یا « Shahrood University of Technology » به چاپ خواهد رسید .**
- **حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه رعایت می گردد.**
- **در کلیه مراحل انجام این پایان نامه ، در مواردی که از موجود زنده (یا بافتهای آن ها) استفاده شده است ضوابط و اصول اخلاقی رعایت شده است .**
- **در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل راز داری ، ضوابط و اصول اخلاق انسانی رعایت شده است .**

تاریخ

امضای دانشجو

مالکیت نتایج و حق نشر

- **کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .**
- **استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.**

چکیده

امروزه نیاز روزافزون به ارسال و ذخیره‌سازی اطلاعات، فشرده‌سازی را به امری بسیار ضروری تبدیل کرده است داده‌های ویدیویی بسیاری به صورت روزافزون درکاربردهای متنوع و متفاوتی در حال تولید، ذخیره، و ارسال هستند که نقش بسیار مهمی در زندگی روزمره انسان‌ها و کاربردهای اکتشافی، فضایی و نظامی ایفا می‌کنند. در بسیاری از این کاربردها نیاز است که تصاویر ویدیویی به صورت بلادرنگ ارسال و ذخیره شوند. در کاربردهای بلادرنگ موضوع بسیار مهم سرعت پردازش است تا بتوان حجم داده‌های ورودی را به صورت بلادرنگ فشرده کرد، همچنین قابلیت انعطاف سیستم و توانایی به‌روزرسانی آن با هزینه کم نیز بسیار مورد توجه می‌باشند. برای افزایش سرعت یکی از بهترین راهکارها استفاده از پردازش موازی می‌باشد. یک راه‌کار مناسب برای دستیابی به پردازش موازی استفاده از FPGA ها می‌باشد؛ که توانایی موازی‌سازی و قابلیت انعطاف بالایی دارند. قابلیت موازی‌سازی راه‌کار مناسبی جهت بالا بردن سرعت است که با استفاده از آن می‌توان با یک طراحی مناسب سرعت و کارایی بالایی برای طرح مورد نظر فراهم کرد. در این پایان‌نامه، به دنبال ارائه روشی جهت فشرده‌سازی برخط ویدیو در بستر FPGA هستیم. در این راستا ابتدا یک الگوریتم فشرده‌سازی بلوکی مبتنی بر موجک با پیچیدگی کم و مناسب با بستر سخت‌افزاری طراحی میکنیم. الگریتم طراحی شده قابلیت موازی‌سازی بالایی داشته و برداشته شده از الگوریتم‌های فشرده‌سازی بلوکی موجود مانند EBCOT و SBHP، است، و سپس سخت‌افزار مطلوب را بر اساس نیاز، طراحی و مونتاژ می‌کنیم. در پایان تبدیل موجک دو بعدی به روش طرح بالابری در بستر سخت افزار طراحی شده پیاده‌سازی شد. زمان انجام یک سطح تبدیل موجک بر رویFPGA SPARTAN 3 XC3S400 با فرکانس کاری 78Mhz برابر 3.65ms است و تعداد برش‌های مورد استفاده برای پیاده سازی تبدیل موجک ۱۱۱۷ است که در مقایسه با دو روش پیاده‌سازی شده تبدیل موجک به روش طرح بالابری نتایج شبیه‌سازی نشان دهنده‌ی بهبود طرح پیشنهادی از نظر استفاده از منابع موجود در FPGA است.

کلمات کلیدی: فشرده سازی برخط ویدیو– موازی سازی– تبدیل موجک- طرح بالابری- FPGA

فهرست مطالب

۱ مقدمه

۲ مروری بر کارهای انجام شده

۳ مبانی تبدیل موجک

۴ الگوریتم مورد استفاده

۵ سخت افزار طراحی شده

پیوست ب روند کلی ایجاد پروژه و برنامه ریزی FPGA در محیط نرم افزار ISE

فهرست شکل‌ها

فهرست جداول

فصل اول

مقدمه

۱-۱ مقدمه

۱-۱-۱ فشرده‌سازی ویدیو

امروزه نیاز روزافزون به ارسال و ذخیره‌سازی اطلاعات، فشرده‌سازی را به امری بسیار ضروری تبدیل کرده است داده‌های ویدیویی بسیاری به صورت روزافزون درکاربردهای متنوع و متفاوتی در حال تولید، ذخیره، و ارسال هستند که نقش بسیار مهمی در زندگی روزمره انسان‌ها و کاربردهای اکتشافی، فضایی و نظامی ایفا می‌کنند. در بسیاری از این کاربردها نیاز است که تصاویر ویدیویی به صورت بلادرنگ ارسال و ذخیره شوند، کاربردهایی مانند اجلاس ویدیویی[1]، تصویربرداری نظامی و همچنین در عملیات فضایی. فشرده‌سازی می‌تواند دو نقش اساسی را در این کاربردها بازی کند. نقش اول کاهش حجم اطلاعات ارسالی و نقش دوم کاهش فضای مورد نیاز برای ذخیره‌سازی و مدیریت موثرتر آن‌ها است. برای هر کاربرد بسته به نیاز می‌توان از روش‌های فشرده‌سازی متفاوتی برای رسیدن به میزان کیفیت مطلوب استفاده کرد. انتخاب روش مطلوب بسته به کیفیت مورد نیاز و همچنین حجم پردازش است.

۱-۱-۲ فشرده‌سازی بلادرنگ ویدیو در بستر FPGA

در کاربردهای بلادرنگ موضوع بسیار مهم سرعت پردازش است تا بتوان حجم داده‌های ورودی را به صورت بلادرنگ فشرده کرد، همچنین قابلیت انعطاف سیستم و توانایی به‌روزرسانی آن با هزینه کم نیز بسیار مورد توجه می‌باشند. برای افزایش سرعت یکی از بهترین راهکارها استفاده از پردازش موازی می‌باشد. برای تحقق پردازش موازی می‌توان از مدارهای ASIC[2] استفاده کرد که استفاده از آن‌ها محدودیت‌هایی را در پیش دارد، از جمله این محدودیت‌ها می‌توان به هزینه اولیه طراحی و عدم انعطاف‌پذیری برای بروز رسانی و یا اصلاح مدار اشاره کرد. برای ساخت یک مدار ASIC باید طرح مورد نظر را به کارخانه‌های موجود ارسال کرد و بر اساس حجم مدار هزینه پرداخت کرد و اگر طرح مورد

[1] Video conferencing
[2] Application specific integrated circuit

نظر به تولید انبوه نرسد هزینه پرداخت شده قابل توجه است، از طرفی برای تغییر در طراحی مدار نیاز به طراحی و ساخت یک IC جدید است که هزینه و زمان زیادی را در پی دارد.

راه‌کار دیگری که می‌تواند کارایی بسیار بالایی برای هدف مورد نظر داشته باشد استفاده از FPGA[1] ها می‌باشد؛ که توانایی موازی‌سازی و قابلیت انعطاف بالایی دارند. به عنوان مثال در ماهواره‌های ارسالی به فضا، تصویربرداری و ارسال تصاویر به زمین یک وظیفه مهم تلقی می‌شود. در این ماهواره‌ها به علت محدودیت شدید در میزان حمل بار، ذخیره‌سازی و یا پردازش تصاویر در فضا مقدور نیست، در نتیجه تصاویر گرفته شده باید به صورت بلادرنگ به زمین ارسال شوند. به دلیل محدودیت در پهنای باند ارسال اطلاعات، تصاویر گرفته شده قبل از ارسال باید فشرده‌سازی شوند. همچنین عامل اساسی دیگر در ماهواره‌ها قابلیت انعطاف و ایجاد تغییر و اصلاح مدار به صورت نرم‌افزاری می‌باشد زیرا اصلاح سخت‌افزاری آن‌ها هزینه بسیار زیادی دارد، برای نمونه تعمیر آیینه اولیه ۱۹ اینچی شکسته شده تلسکوپ هابل نزدیک به ۵۰ میلیون دلار هزینه برداشت [1] که نشان دهنده هزینه بسیار بالای اصلاح سخت افزاری است در نتیجه با استفاده از مدارات قابل تعریف برای پیاده‌سازی سیستم فشرده‌سازی می‌توان هزینه اصلاحات سخت افزاری مربوط به مدارها را کاهش داد و بدون نیاز به ارسال روبات یا انسان تغییرات مورد نظر را در سخت‌افزار اعمال کرد.

قابلیت موازی‌سازی راه‌کار مناسبی جهت بالا بردن سرعت است که با استفاده از آن می‌توان با یک طراحی مناسب سرعت و کارایی بالایی برای طرح مورد نظر فراهم کرد؛ به عنوان مثال در زمینه فشرده‌سازی ویدیو فریم ورودی را می‌توان به چند بلوک مجزا تقسیم کرد و هر بلوک را به طور موازی پردازش نمود، در نتیجه با استفاده از سرعت بالای فراهم شده پردازش بلادرنگ ممکن می‌شود.

[1] Field Programmable Gate Array

۱-۱-۳ هدف و ساختار پایان‌نامه

در این پایان‌نامه، به دنبال ارائه روشی جهت فشرده‌سازی ویدیو در بستر FPGA هستیم. در این راستا ابتدا یک الگوریتم فشرده‌سازی بلوکی مبتنی بر موجک با پیچیدگی کم مناسب با بستر سخت‌افزاری طراحی میکنیم. الگریتم طراحی شده قابلیت موازی‌سازی بالایی دارد و برداشته شده از الگوریتم‌های فشرده‌سازی بلوکی موجود مانند EBCOT و SBHP، است، سپس سخت‌افزار مطلوب را بر اساس نیاز، طراحی و مونتاژ می‌کنیم.

ساختار این پایان‌نامه به این صورت است که در فصل دوم به بررسی الگوریتم‌های پیاده‌سازی شده موجود بر روی FPGA می‌پردازیم. در فصل سوم مفاهیم پایه، در فصل چهارم الگوریتم پیشنهادی را توضیح می‌دهیم و در فصل پنجم به برسی و توضیح کامل سخت‌افزار می‌پردازیم، در فصل ششم نتایج پیاده‌سازی سخت‌افزاری را بررسی میکنیم و در فصل هفتم به نتیجه گیری و پیشنهاد برای بهبود کار می‌پردازیم.

فصل دوم

مروری بر کارهای انجام شده

مقدمه

کارهای صورت گرفته در زمینه فشرده‌سازی بلادرنگ در بستر FPGA معمولا محدود به پیاده سازی الگوریتم‌های موجود فشرده‌سازی می‌شود. که ساده سازی‌های بر روی آن‌ها صورت گرفته تا بتوان با توجه به محدودیت‌های سخت افزاری قابل پیاده سازی باشند. و تقریبا در همه موارد پیاده سازی بر روی بردهای آماده و از پیش طراحی شده صورت گرفته است. در نتیجه در طول این فصل ابتدا به توضیح فشرده‌سازی ویدیو و استانداردهای فشرده‌سازی بلادرنگ می‌پردازیم، و در ادامه به بررسی چند مقاله موجود در این زمینه که از همین الگوریتم‌ها استفاده کرده‌اند می‌پردازیم.

۲-۱ فشرده‌سازی ویدیو

تحلیل آماری سیگنال‌های ویدیویی نشان می‌دهد که همبستگی و شباهت زیادی بین فریم‌های متوالی و نیز بین عناصر داخلی هر فریم وجود دارد. حذف این همبستگی‌ها منجر به فشرده‌سازی در پهنای باند می‌شود. بدون اینکه تأثیر چندان نامطلوبی در درجه تفکیک[1] تصاویر برجای بگذارد؛ به علاوه سیستم بینایی انسان (HVS[2]) به حذف برخی اطلاعات مکانی-زمانی[3] سیگنال ویدیویی غیرحساس است و از این رو می‌توان از آن برای افزایش میزان فشرده‌سازی سیگنال ویدیویی استفاده کرد. هرگاه فقط همبستگی‌های داخلی (یعنی همبستگی‌های مکانی) یک تصویر حذف شود، در اصطلاح به آن کدگذاری درون-فریمی[4] اطلاق می‌شود. مانند استاندارد JPEG، اما اگر همبستگی‌های بین فریمی (یعنی همبستگی‌های زمانی) نیز حذف شود، به آن کدگذاری بین-فریمی[5] اطلاق می‌گردد. نوع متداولی از این نوع کدگذاری، کدگذاری بین- فریمی[6] پیش بین می‌باشد که در استانداردهایی نظیر H.261،

[1] Resolution
[2] Human Visual System
[3] Spatial-Temporal
[4] Intra-Frame Coding
[5] Inter-Frame Coding
[6] Inter-Frame Predictive Coding

H.263، H.264، MPEG1، MPEG2، MPEG4 مورد استفاده قرار می‌گیرد. این نوع از کدگذاری مبتنی بر سه اصل زیر به منظور کاهش تزاید[1] می‌باشد :

۱- کاهش تزاید مکانی: به منظور کاهش تزاید و یا شباهت بین پیکسل‌های هر فریم.

۲- کاهش تزاید زمانی: به منظور کاهش شباهت بین فریم‌های متوالی از طریق کدگذاری تفاضل آن‌ها.

۳- کدگذاری آنتروپی: به منظور کاهش تزاید بین نمادهای حاصل از مراحل فشرده‌سازی فوق با استفاده از تکنیک‌های کدگذاری با طول متغیر (VLC[2]).

در ادامه به توضیح چند استاندارد کدگذاری بلادرنگ می‌پردازیم.

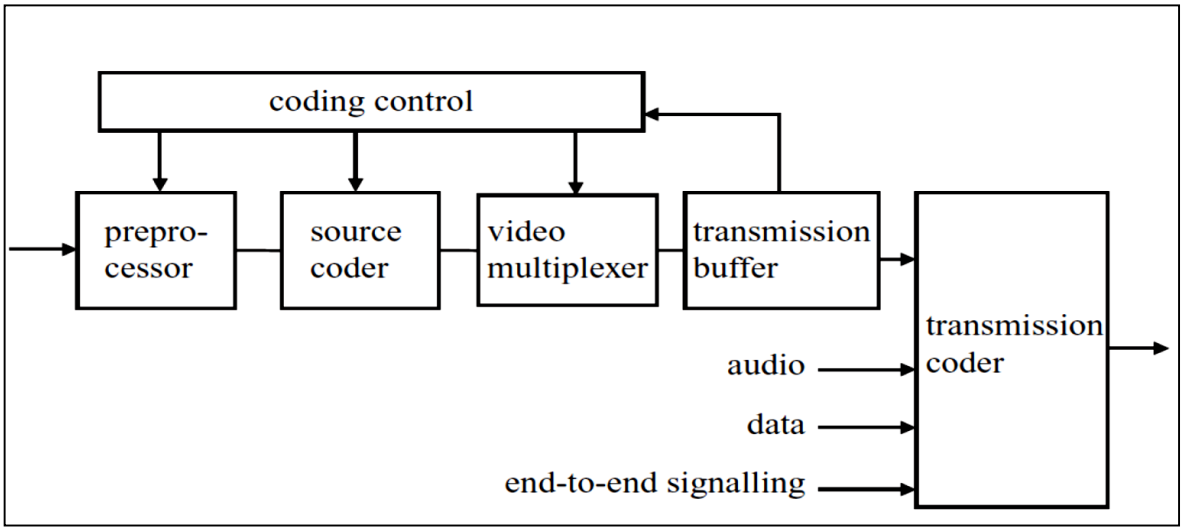
۲-۲ کدگذاری به منظور کنفرانس ویدیویی (H.261)

استاندارد H.261 به منظور ارائه روش‌هایی جهت کدگذاری، کدگشایی و انتقال سیگنال‌های ویدیویی روی شبکه های ISDN و با یکی از نرخ‌های انتقال P×64Kbit/s به وجود آمد. پارامتر p عددی بین ۱ تا ۳۰ می‌باشد. این کدگذار یک کدگذار بین- فریمی مبتنی بر DCT است. در این کدگذار ابتدا از پیش بینی بین- فریمی در حوزه پیکسلی استفاده می‌شود. در ادامه خطای پیش بینی به حوزه فرکانسی تبدیل شده و سپس به منظور کاهش پهنای باند از کوانتیزاسیون استفاده می‌گردد. در مرحله‌ی پیش بینی می‌توان از جبران حرکت (MC) نیز استفاده کرد گرچه این کار اختیاری است. بنابراین، در این کدگذار تزاید زمانی از طریق پیش بینی بین- فریمی و تزاید مکانی از طریق گرفتن تبدیل کاهش داده می‌شود. این نکته قابل ذکر است که در طراحی واحدهای رمزگذار و رمزگشا آزادی‌ها و اختیار عمل‌هایی به طراح داده‌شده است. اما به شرطی که همچنان از گرامر داده‌شده برای دنباله‌ی بیتی تبعیت کنند. گاهاً به جای نام H.261 از نام RM8 نیز استفاده می‌شود که اشاره به آخرین نسخه‌ی تصحیح شده (نسخه‌ی هشتم) از دست نوشته‌ی مرجع استاندارد[3] دارد که در سال ۱۹۸۹ تهیه گردیده است.

[1] Redundancy
[2] Variable Length Coding
[3] Reference Manual (RM)

۲-۲-۱ قالب و ساختار ویدیو

شکل (۲-۱) بلوک دیاگرام مربوط به سیستم صوتی-تصویری مبتنی بر H.261 را نشان می‌دهد. در این بلوک دیاگرام واحد پیش پردازنده سیگنال ویدیویی را که به قالب CCIR-601 (قالب خروجی یک دوربین) است به قالب جدیدی تبدیل می‌کند. پارامترهای مختلف کدگذاری با دیگر داده‌ها (شامل داده‌های صوتی، داده‌های سیگنالینگ، و داده‌های سیگنال ویدیویی فشرده شده) مالتی پلکس و ترکیب می‌شوند. وظیفه بافر انتقال، کنترل نرخ بیت داده‌های ارسالی است.



شکل (۲-۱) دیاگرام بلوکی یک رمزگذار صوتی-تصویری H.261 [2]

در استاندارد H.261 امکان انجام درونیابی سه تصویر بین هر دو تصویر کدگذاری و ارسال شده وجود دارد. این کار باعث کاهش نرخ فریم از مقدار ۳۰ فریم بر ثانیه به مقادیر به ترتیب ۱۵ (اگر فقط یک تصویر درونیابی شود)، ۱۰ (اگر دو تصویر درونیابی شود)، و ۷/۵ فریم بر ثانیه (اگر سه تصویر درونیابی شود) می‌گردد. برای کاهش بیشتر نرخ بیت داده‌های ارسالی از قالب تصویری QCIF[1] برای کاهش درجه تفکیک فریم‌ها استفاده می‌شود.

---

[1] *Quarter Common Intermediate Forma*

در قالب‌های تصویری CIF و QCIF بلوک‌های تصویر به گروه‌هایی به نام ماکروبلوک (MB) گروه‌بندی می‌شوند که هر ماکروبلوک شامل چهار بلوک شدت روشنایی[1] و دو بلوک رنگ[2] متناظر با هم می‌باشد. ماکروبلوک‌ها به نوبه خود به لایه‌هایی به نام گروه بلوک‌ها یا GOB[3] تقسیم بندی می‌شوند. همان طور که در شکل (۲-۲) نشان داده‌شده است هر فریم به قالب CIF شامل ۱۲ عدد GOB و هر فریم به قالب QCIF شامل ۳ عدد GOB می‌باشد.

هدف از ساختاربندی فریم‌ها بر حسب ماکروبلوک‌ها و GOBها عبارت است از :

• استفاده از کدگذاری‌های مشابه در حالت‌های بین/درون-فریمی برای بلوک‌های رنگ و شدت روشنایی مربوط به ناحیه ی مشترک

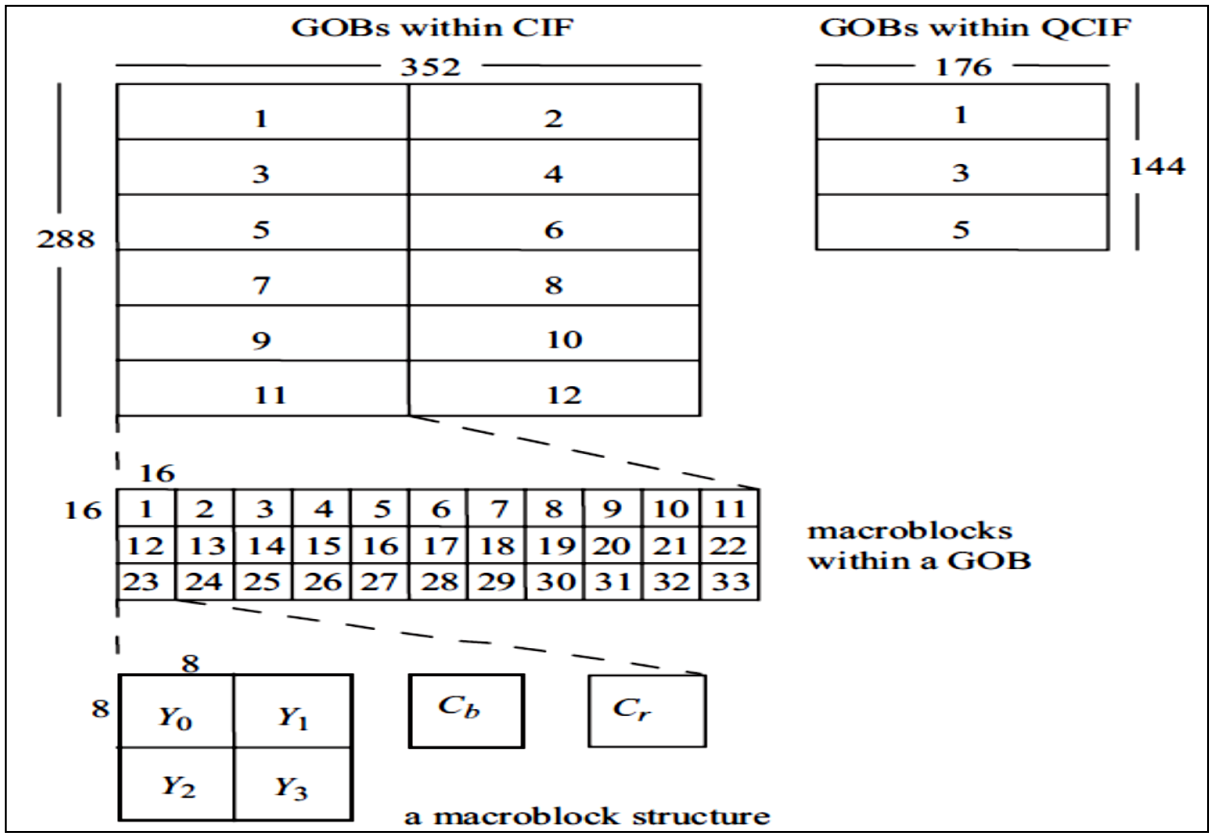
• استفاده از یک بردار حرکت برای هر دو بلوک رنگ و شدت روشنایی

• کدگذاری موثر بلوکهای۸ ×۸ تبدیل DCT

• کمک به از سرگیری ارسال داده‌ها در هنگام خرابی بیت‌های ارسالی

• حمل اطلاعات جانبی مربوط به GOB، MB یا لایه‌های بالاتر. این اطلاعات شامل قالب تصویر، مراجع زمانی، نوع MB ، اندیس کوانتایزر و غیره می‌شود.

[1] Luminance
[2] Chrominance
[3] Group of Blocks



شکل (۲-۲) مفاهیم بلوک، مایکروبلوک و GOB در تصاویر با قالب CIF و QCIF [2]

۲-۲-۱-۱ پیش‌بینی

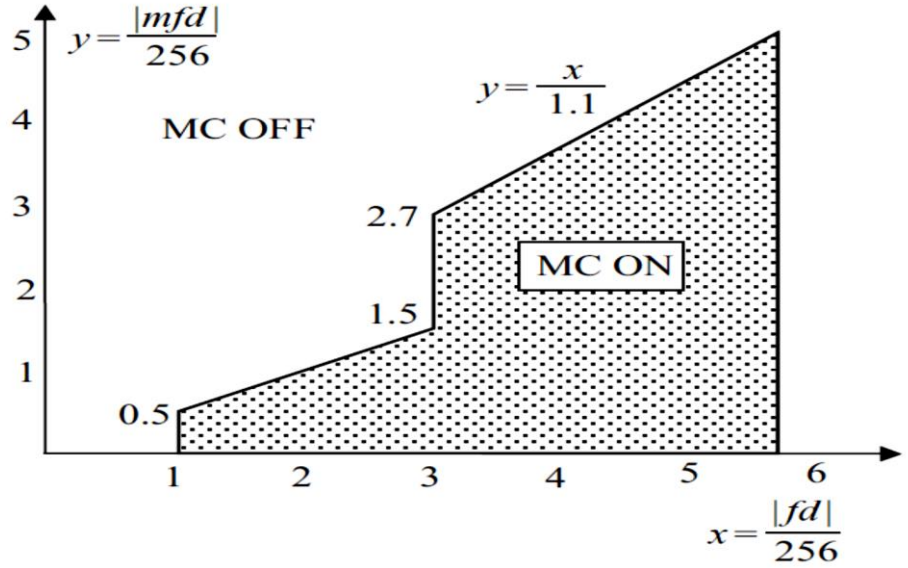
عمل پیش بینی به صورت بین- فریمی انجام شده و می‌تواند شامل جبران حرکت نیز باشد زیرا جبران حرکت (MC) در استاندارد H.261 اختیاری است. واحد رمزگشا به ازاء هر یک MB یک بردار جابجایی دریافت می‌کند.

مولفه‌های این بردار جابجایی اعداد صحیح بوده و از ۱۵-/+ پیکسل بر فریم فراتر نمی‌روند. تخمین حرکت فقط روی مؤلفه شدت روشنایی انجام می‌شود اما از همین بردار جابجایی محاسبه شده برای جبران حرکت همه‌ی چهار بلوک شدت روشنایی استفاده می‌شود. برای جبران حرکت دو بلوک رنگ، ابتدا بردار جابجایی مذکور را نصف و سپس به سمت صفر گرد می‌کنند.

برای انتقال بردارهای جابجایی محاسبه شده، تفاضل آن‌ها از یکدیگر را محاسبه و از کدگذاری با طول متغیر استفاده می‌کنند.

۲-۲-۱-۲ تصمیم در مورد انجام جبران حرکت

تصمیم در مورد این که برای یک MB موجود نیاز به انجام جبران حرکت می‌باشد یا خیر بستگی به این دارد که آیا جبران حرکت باعث کاهش قابل توجه خطای پیش بینی می‌شود یا نه؟ برای انجام این تصمیم گیری از نمودار شکل (۲-۳) استفاده می‌شود. محور افقی نشان دهنده‌ی خطای پیش بینی نرمالیزه شده بدون جبران حرکت و محور عمودی نشان دهنده‌ی خطای پیش بینی نرمالیزه شده با انجام جبران حرکت می‌باشد. اگر مقادیر خطای پیش بینی با و بدون جبران حرکت به گونه‌ای باشند که در ناحیه‌ی هاشور زده شده قرار بگیریم، باید از جبران حرکت استفاده کنیم .



شکل (۲-۳) تصمیم گیری در مورد استفاده/عدم استفاده از جبران حرکت [2]

۲-۳ کدگذاری اطلاعات ویدیویی در کاربرد مخابرات نرخ بیت پایین (H.263)

استاندارد H.263 از نوعی نمایش رمزگذاری شده استفاده می‌کند که به منظور پشتیبانی از نرخ انتقال‌های پایین در کاربردهایی نظیر شبکه‌های موبایل، شبکه‌های PSTN[1]، و شبکه‌های ISDN[2] طراحی شده است. نسخه آزمایشی این استاندارد در سال ۱۹۹۵ آماده گردید. این استاندارد، توسعه یافته‌ی

[1] Public Switched Telephone Network
[2] Integrated Services Digital Network

استانداردهای H.261 و MPEG-1.2 است. برای این منظور ابعاد تصاویر مورد نظر کوچک انتخاب گردید. در استاندارد H.263 استفاده از پنج قالب sub-QCIF،QCIF،CIF،4CIF،16CIF (مطابق با جدول ۲-۱) توصیه گردیده است.

جدول ۲-۱ پنج قالب تصویری مورد پشتیبانی H.263

Picture fomat	Number of pixel for luminance per line	Number of line for luminance per pixle	Number of pixel for chrominance per line	Number of line for chrominance per pixle
Sub-QCIF	128	96	64	48
QCIF	176	144	88	72
CIF	352	288	176	144
4CIF	704	576	352	288
16CIF	1408	1152	704	576

البته بعد از ارائه H.263 تلاش‌هایی در جهت رسیدن به کاربردهای مخابرات بلادرنگ نیز صورت گرفت، که منجر به پیشنهاد H.263+ گردید. این تلاش‌ها ادامه یافت تا اینکه در سال ۲۰۰۰ نسخه توسعه یافته‌ی بعدی نیز با نام H.263++ نیز ارائه گردید [3]. فعالیت دیگری که در سازمان استاندارد ITU-T به موازات صورت گرفت شامل همکاری با گروه مربوط به استاندارد MPEG-4 و تهیه استانداردی به نام H.263L بود. [4] در همه‌ی این استانداردها هدف اصلی رسیدن به مخابرات بلادرنگ بود که مستلزم رعایت دو محدودیت مهم زمان تاخیر و پیچیدگی واحدهای رمز گذار و رمزگشا است.

۲-۴ کدگذار ویدیویی پیشرفته (H264)

استاندارد H.264 محصول شرکت سازمان‌های ITU-T و ISO/IEC (گروه کاری MPEG-4) است که در سال ۲۰۰۳ توسط سازمان ITU-T نهایی و اعلام گردید. این استاندارد در سازمان ISO به نام

14496-10 و یا MPEG4-part 10 شناخته شده است. البته این پروژه در شروع کار خود با نام AVC[1] شروع به کار کرده و به همین دلیل به طور غیر رسمی به همین نام خوانده می‌شود. کارایی فشرده‌سازی این استاندارد حداقل دو برار استاندارد MPEG-2 است. به همین دلیل امروزه (در سال ۲۰۱۰) عمده کشورها از این استاندارد در کاربردهای متنوعی استفاده می‌کنند.

برخی ویژگی‌های برجسته این کدک به قرار زیر است:

- کاهش نرخ بیت تا ۵۰ درصد نسبت به استانداردهای H.263+ و MPEG-4 visual
- پشتیبانی از حالت تأخیر کم برای سازگاری با برخی کاربردهای مخابراتی و پشتیبانی از حالت حجم پردازش بالا در کاربردهایی که تاخیر زمانی مساله مهمی برای آن‌ها محسوب نمی‌شود، مانند ذخیره سازی ویدیو
- مقاوم بودن نسبت به خطای کانال
- سازگاری و انعطاف با ساختار شبکه‌های انتقال داده

برخی نقاط تمایز این استاندارد از نظر کاری به قرار زیر است:

- استفاده از دو کدگذار انتروپی: کدگذار با طول متغییر و وابسته به فحوا (CAVLC[2])
- انجام جبران حرکتی به کمک چندین تصویر مرجع
- انجام تخمین حرکت با بلوک‌هایی با ابعاد متغییر ۴×۴ تا ۱۶×۱۶
- انجام جبران حرکت با دقت یک چهارم پیکسل، بهبود دقت پیش بینی، و استفاده از درونیابی با پیچیدگی محاسباتی کمتر
- استفاده از تبدیل معکوس پذیر عدد صحیح به منظور جلوگیری از عدم تطابق تبدیل و عکس آن
- استفاده از بلوک‌هایی با ابعاد متغییر در تبدیل عدد صحیح ۸×۸ و ۴ ×۴

[1] Advance Video Coding
[2] Context-Adaptive Variable Length Coding

- استفاده از فیلتر حذف اثر بلوکی ایجاد شده به دلیل جبران سازی حرکت و کوانتیزاسیون و مقاوم سازی بیشتر نسبت به خطا
- بخش بندی داده‌ها (DP) و بسته‌بندی قطعه‌های ویدیویی به سه سطح اولویت به منظور محافظت در حین انتقال
- انتقال با تزاید برخی نواحی به منظور بهبود مقاوم سازی نسبت به گم شدن داده‌ها

۲-۵ مسائل مربوط به پیاده سازی و بررسی چند مقاله پیاده سازی شده

روش‌های موازی سازی به طور کلی میتوانند به سه دسته کلی تقسیم شوند.

۱-روش مبتنی بر موازی سازی عملیات و انجام همزمان آن‌ها

۲-روش مبتنی بر موازی سازی داده، و تقسیم داده ورودی به چندین بخش و انجام پردازش بر روی بخش‌های مختلف

۳-روش مبتنی بر عملیات و داده که ترکیب دو روش بالا می‌باشد

روش‌هایی که تا کنون از آن‌ها استفاده شده به طور عمده مربوط به استانداردهای فشرده‌سازی موجود می‌باشند. که برای پیاده سازی آن‌ها بر روی FGPA با استفاده در اکثر موارد نیاز به استفاده از چند FPGA می‌باشد. همچنین در روش‌های قدیمی تر برای انجام یک سری عملیات ثابت مانند ضرب و جمع یک مدار جدا استفاده کرده و با استفاده از قابلیت تعریف مجدد درون سیستم[1] موجود در FPGA عملیاتی را که نیاز به انعطاف پذیری بیشتری می‌باشند را بر روی FPGA پیاده سازی کرده.

همچنین روش استفاده شده دیگر کاهش پیچیدگی محاسباتی و ساده سازی الگوریتم استفاده شده می‌باشد. یعنی با توجه به حجم محاسباتی که می‌توان در بستر FPGA موجود انجام داد. الگوریتم مورد نظر را برای قابل اجرا بودن بر روی آن ساده کرد [5].

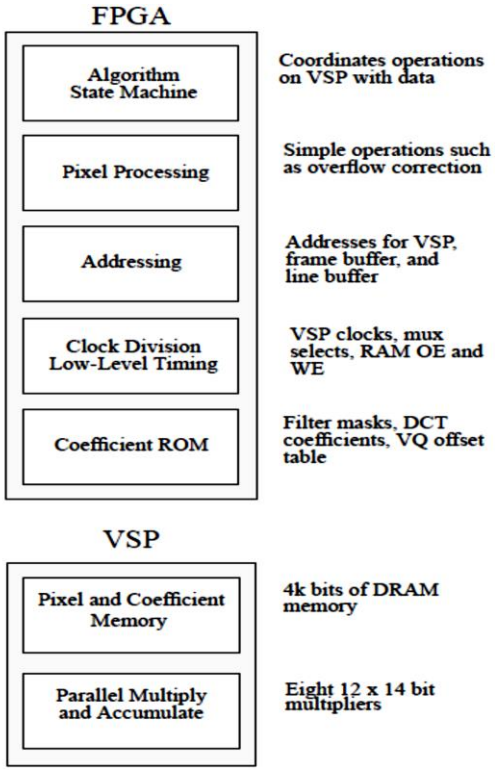
[1] In system reconfigrability

۲-۵-۱ روش پیاده سازی فشرده‌سازی ویدیو در بستر FPGA [5]

در [5] از الگوریتم فشرده‌سازی با پیچیدگی کم بر پایه موجک استفاده شده. همچنین تبدیل موجک استفاده شده از ضرایب صحیح و جمع و شیفت استفاده می‌کند. اگرچه این ساده سازی‌ها کارایی را کاهش می‌دهد ولی باعث پایین آمدن پیچیدگی طراحی نیز می‌شود.

طرح پیاده سازی شده دارای نرخ ۲۰ فریم بر ثانیه با سایز فریم ۲۵۶×۲۵۶ و به صورت ۸ بیتی با مصرف توان ۱ وات است. که بر روی FPGA Xilinx Xc4008 پیاده سازی شده، به علت پیچیدگی محاسباتی و میزان گیت مورد نیاز بالا از یک پردازنده سیگنال ویدیویی (VSP)[1] ASIC طراحی شده در کنار FPGA برای انجام کارهای محاسباتی مانند ضرب و جمع استفاده شده. در نتیجه مسائل مربوط به الگوریتم فشرده‌سازی بر روی FPGA و کارهای محاسباتی که تقریبا در همه الگوریتم‌ها مشترک هستند بر روی VSP پیاده سازی شده‌اند. در نتیجه طرح انعطاف و قابلیت تغییر خود را حفظ کرده و در صورت نیاز می‌توان از الگوریتم دیگر استفاده کرد. در شکل (۲-۴) نحوه تقسیم محاسبات بین FPGA و VSP نشان داده‌شده.

[1] Video Signal Processor



شکل (۲-۴) نحوه تقسیم محاسبات بین FPGA و VSP [5]

در انتها در جدول ۲-۲ مشخصات زمانی و آماری طرح نشان داده‌شده است.

جدول ۲-۲ مشخصات زمانی و آماری

Frame Rates for FPGA-VSP Processor:	
(All operations on 256×256× 8 bit pictures)	
7×7 Mask 2D Filter	13.3 frames/sec
8×8 Block DCT	55 frames/sec
4× 4 Block VQ at 1/2 bpp	7.4 frames/sec
One level wavelet transform	35.7 frames/sec
Max FPGA Clock Rate	20 MHz
(except VQ with minimize on FPGA at 5MHz)	
Max VSP Clock 50MHz (400M Multiply/sec.)	

۲-۵-۲ پیاده‌سازی سخت‌افزاری نرم‌افزاری کدگذار H.263 بر روی FPGA [6]

در [6] به بررسی پیاده سازی الگوریتم فشرده‌سازی H.263 بر روی FPGA پرداخته می‌شود همانطور که در بالا توضیح داده شد، الگوریتم H.263 به منظور پشتیبانی از نرخ انتقال‌های پایین استفاده می‌شود و شامل بلوک‌های تخمین حرکت، تبدیل کسینوسی گسسته‌DCT، چندی ساز و کدگذار با طول متغییر است.

بستر سخت افزاری که پیاده سازی بر روی آن صورت گرفته برد Stratix II شرکت ALTERA می‌باشد که در آن از Stratix II EP2S60 FPGA استفاده شده است. این FPGA شامل یک پردازنده نرم‌افزاری به نام NIOS II است که در طراحی مورد نظر از آن استفاده شده است برای فهمیدن اینکه کدام روش برای پیاده‌سازی در بستر FPGA و پردازش موازی مناسبتر است و کدام روش برای پیاده‌سازی نرم افزاری، تمام بلوک‌های عملیاتی بطور کامل بررسی شده و بلوک‌های مناسب انتخاب شده‌اند.

ایده اصلی این است که بلوک‌ها با ساختار موازی در بستر سخت‌افزار طراحی شوند تا سرعت بالاتری بدست آید.

در آخر الگوریتم مورد نظر بر روی برد پیاده سازی شده و اطلاعات را از دوربین دریافت کرده و پردازش را بر روی آن اعمال میکند و به کامپیوتر ارسال می‌کند.

نتایج پیاده سازی و میزان منابع استفاده شده در جدول آورده شده است. فرکانس کاری طرح 120MHz است.

جدول ۲-۳ نتایج پیاده سازی

	NIOS II (FAST) (%)	2-D DCT coprocessor (%)	2-D IDCT coprocessor (%)
ALUTs	11	3	3
ESBs	44	1	1
DSPs	3	8	8
IOBs	41	15	15
Fmax(MHz)	227	133	139

نتایج تقابل بین کارایی و سرعت را نشان می‌دهد. برای همه نمونه ها با توالی QCIF و میزان ۱۰ فریم بر ثانیه و پارامتر چندی سازی ۱۶ بدست آمده است.

فصل سوم

مبانی تبدیل موجک

۳-۱ چرا تبدیل موجک ؟ [2]

قبل از توصیف تبدیل موجک و بیان کاربرد آن در فشرده‌سازی تصویر لازم است به دو سؤال زیر پاسخ داده شود:

۱- مشکل تبدیل DCT چیست و چرا ما باید از تبدیل موجک استفاده کنیم؟

۲- اگر تبدیل موجک بهتر از DCT است چرا استاندارد JPEG از آن استفاده نکرد؟

پاسخ به اولین سؤال به صورت زیر است: تبدیل DCT و دیگر تبدیلاتی که مبتنی بر بلوک هستند، به این شکل عمل می‌کنند که تصویر را به تعدادی بلوک ناهمپوشان تقسیم کرده و هر بلوک را جداگانه پردازش می‌کنند. در نرخ بیت‌های بسیار کوچک لازم است که ضرایب تبدیل با گام بزرگی کوانتیزه شوند و بنابراین در حین کدگشایی، خطای بازسازی بزرگی تولید می‌شود. این خطا در مرز بلوک‌ها واضح‌تر و مشهودتر از بقیه نواحی بوده و تولید یک اثر نامطلوب به نام اثر بلوکی[1] می‌کند. یک راه برای حذف یا کاهش این اثر نامطلوب این است که به توابع پایه اجازه دهیم در این نقاط به صفر میل کنند یا اینکه بلوک‌ها را به صورت همپوشان انتخاب کنیم. تکنیک دوم، تبدیل متعامد همپوشان (LOT[2]) [7] است. تبدیل موجک حالت خاصی از این تبدیل است و بنابراین انتظار می رود اثرات بلوکی از خود نشان ندهد.

پاسخ به سؤال دوم مربوط به روش‌های جدیدی موجود در زمینه کدگذاری تصویر در اواسط دهه ۱۹۸۰ می‌شود یعنی زمانی که استاندارد JPEG در حال بررسی و توسعه بود. گرچه در این زمان، تبدیل موجک و نیاکان آن، کدگذاری زیر باندی، شناخته‌شده بودند، هنوز روش کارآمدی برای کدگذاری ضرایب تبدیل موجک در مقایسه با تبدیل DCT وجود نداشت. در عمل تمام پیشنهادهای ارسال‌شده برای کمیته JPEG مبتنی بر DCT بوده و هیچ یک استفاده‌ای از تبدیل موجک نکرده بود. به علاوه در همان زمان از بین ۱۵ پیشنهاد ارسال‌شده برای کمیته H.261، ۱۴ پیشنهاد مبتنی بر DCT و ۱ پیشنهاد

[1] Blocking Artifact
[2] Lapped Orthogonal Transform

مبتنی بر چندی سازی برداری[1] بوده و هیچ یک اشاره‌ای به تبدیل موجک نکرده بودند. به همین دلیل بنابر وضعیت زمانی آن سال‌ها، از تبدیل DCT در استاندارد JPEG استفاده شد. اما در JPEG2000 از روش تبدیل موجک استفاده شده است.

با این حال، وضعیت روش‌های فشرده‌سازی تصویر مبتنی بر تبدیل موجک از زمان پیشنهاد استاندارد JPEG تا کنون رشد و پیشرفت فزاینده و چشمگیری داشته است. بیش‌ترین سهم در این زمینه را باید به آقای ژوزف شاپیرو[2] داد که روش موجک درخت صفر جاسازی‌شده EZW[3] را پیشنهاد داد و پیشرفت بزرگی در زمینه کدگذاری ضرایب موجک ایجاد کرد[8] .

۲-۳ کدگذاری زیر باندی [2]

قبل از پرداختن به تبدیل موجک، نگاهی به نیاکان آن یعنی کدگذاری زیر باندی می‌اندازیم که گاهی به آن تبدیل موجک اولیه نیز گفته می‌شود [9]. البته همان طور که بعداً خواهیم دید این دو اصطلاح، یکی هستند. اصطلاح زیر باند حاصل کار مهندسین [10] و تبدیل موجک حاصل کار ریاضیدانان [11] است. بنابراین، قبل از پرداختن به ریاضیات که گاهاً تعقیب و ادامه آن خسته‌کننده می‌شود، نگاهی مهندسی به پردازش چند درجه تفکیک[4] سیگنال می‌کنیم تا درک موضوع برای ما آسان تر گردد. کدگذاری زیر باندی اولین بار توسط کروچر و دیگران در سال ۱۹۷۶[12] پیشنهاد گردید و تاکنون قدرت و کارایی آن در کدگذاری صوت و تصویر به عنوان یک ابزار ساده و قدرتمند به اثبات رسیده است. ایده اصلی این تکنیک، بخش‌بندی طیف سیگنال به چندین محدوده یا باند فرکانسی[5] و سپس کدگذاری و انتقال هر باند به طور مجزا است. این کار مختص کدگذاری تصویر ارائه و استفاده گردید. در توجیه این کار سه نکته قابل تأمل است. اول این که طیف تصاویر طبیعی اغلب طیفی غیرخطی است و بیشتر انرژی این‌گونه تصاویر در محدوده فرکانس‌های پایین متمرکز است. نکته دوم این که

[1] Vector Quantization
[2] Josef Shapiro
[3] Embedded Zero tree Wavelet
[4] Multi resolution
[5] Frequency Band

میزان درک انسان از نویز در هر دو باند فرکانسی بالا و پایین افت می‌کند؛ بنابراین طراح می‌تواند سیستم فشرده‌سازی را به گونه‌ای طراحی کند که مقدار اعوجاج حاصل از فشرده‌سازی بر طبق یک معیار ادراکی تنظیم شود. بالاخره، سوم این که از آنجا که در این روش، تصاویر به صورت کلی و یکجا پردازش می‌شوند نه بصورت بلوکی، بنابراین مساله‌ای به نام اعوجاج بلوکی مشابه با آن چه در مورد روش‌های مبتنی بر تبدیل بلوکی، مانند DCT مطرح بود وجود نخواهد داشت.

بنابراین اگرچه ایده زیر باندی مشابه با تبدیل فوریه، مبتنی بر تحلیل فرکانسی تصویر است اما فیلتربانک‌های آن خاصیت ناهمبسته سازی بهتری دارند که این خاصیت در مورد تصاویر طبیعی مناسب‌تر و مطلوب‌تر است. برای توضیح بیشتر به این نکته توجه کنید که توابع پایه فوریه در حوزه فرکانسی کاملاً دقیق اما در حوزه مکانی دارای دقت نمی‌باشند؛ به عبارت دیگر انرژی آن‌ها در سرتاسر بازه مکانی گسترده شده است. این امر در صورتی که پیکسل‌های تصویر همواره و همیشه باهم همبستگی داشته باشند، مشکلی محسوب نمی‌شود اما در عمل چنین نبوده و پیکسل‌های تصویر در بازه بسیار محدودی باهم همبستگی دارند. به ویژه در نقاط ناپیوستگی تصاویر مانند لبه‌ها، بازه همبستگی مذکور بسیار کوتاه است. بر خلاف توابع پایه فوریه، توابع پایه‌ی زیرباندی نه تنها متمرکز (یا دقت) فرکانسی[1] نسبتاً خوبی دارند، بلکه در حوزه مکانی نیز فشرده و متمرکز می‌باشند. اگر لبه‌های تصویر بیش از حد به هم نزدیک نباشند، اغلب توابع یا عناصر پایه‌ی زیرباندی با این نواحی مرز مشترک نداشته و بنابراین به طور متوسط خاصیت نا همبسته سازی خوبی از خود نشان می‌دهند (توجه شود که انجام فشرده‌سازی مستلزم حذف یا کاهش تزاید بوده و کاهش تزاید نیز مستلزم ناهمبسته سازی است).

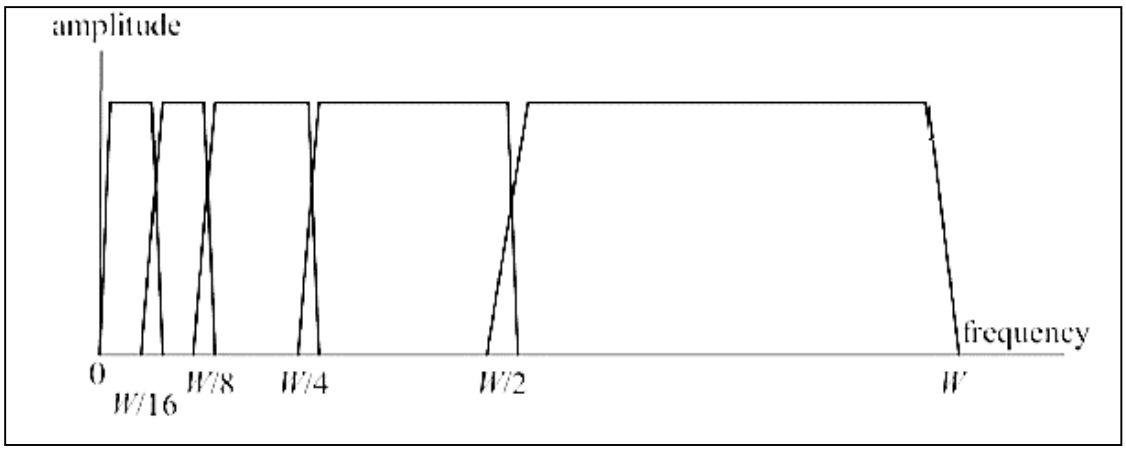
در کدگذاری زیرباندی، جداسازی باندها از یکدیگر توسط فیلتر کردن تصویر ورودی به کمک بانکی از فیلترهای تحلیل میان‌گذر، مطابق با شکل (۳-۱)، انجام می‌شود. در این شکل ملاحظه می‌کنید که جهت هماهنگ کردن طیف تصاویر تجزیه‌شده‌ی خروجی با مشخصات سیستم بینایی انسان، فیلترها

[1] Frequency Concentration

به صورت باندهای اکتاو[1] (یعنی افزایش عرض باندها به صورت توان صحیحی از ۲) طراحی و تنظیم شده‌اند.

از آنجا که پهنای باند هر نسخه از تصویر فیلتر شده کاهش یافته است، از نظر تئوری می‌توان با نرخ کمتری (بر طبق قضیه نایکوئیست) نمونه برداری کرد و تعدادی زیرتصویر[2] کاهش بعد یافته به دست آورد. سپس این زیرتصویرها کوانتیزه، کدگذاری و ارسال می‌شوند. در واحد کدگشا، زیرتصویرهای دریافتی به ابعاد اولیه بازگردانده شده و از مجموعه‌ای از فیلترها به نام فیلتربانک ترکیب عبور داده می‌شوند. این فیلتربانک عمل درونیابی و ترکیب زیرتصویرها به منظور بازسازی تصویر اولیه را انجام می‌دهد.

فیلترهایی که در عمل در بانک فیلتر استفاده می‌شوند، مانند شکل (۳-۱)، ناحیه یا باند گذر محدود و غیر صفر دارند لذا در این فیلتربانک‌ها در فرآیندهای زیرنمونه برداری/ بالا نمونه برداری مقداری اعوجاج همپوشانی[3] در تصویر بازسازی شده وجود خواهد داشت. برای رفع مشکل اعوجاج، فیلترهای مورد استفاده باید رابطه خاصی با هم داشته باشند طوری که مؤلفه‌های همپوشان اثر یکدیگر را خنثی کرده و تصویری بدون اعوجاج همپوشانی حاصل شود.



شکل (۳-۱) یک نمونه بانک فیلترهای میان‌گذر

[1] Octave
[2] Sub image
[3] Aliasing Distortion

۳-۳ تبدیل موجک[2]

تبدیل موجک حالت خاصی از کدگذاری زیرباندی است که در زمینه کدگذاری تصویر و ویدیو بسیار فراگیر و متداول شده است. کدگذاری زیرباندی تصاویر مبتنی بر تحلیل فرکانسی است حال آنکه تبدیل موجک بر تئوری تقریب[1] استوار است. با این حال از آنجا که تصاویر طبیعی به طور محلی هموار بوده و می‌توان آن را با تقریب تکه‌ای - چندجمله‌ای مدل کرد، اگر از تابع چند جمله‌ای مناسب استفاده کنیم، عمل تقریب متناظر با نوعی تحلیل فرکانسی خواهد شد که مشابه با ایده زیرباندی است. در حقیقت، تحلیل موجک راه و ابزار بسیار موثری برای تقریب چنین توابعی در اختیار ما قرار می‌دهد طوری که از تعداد بسیار کمی عناصر پایه (برای انجام تقریب) استفاده می‌کند.

از نظر ریاضی، تبدیل موجک یک تابع مجذور- انتگرال پذیر $x(t)$ معادل با تجزیه این تابع بر حسب مجموعه‌ای از توابع پایه به صورت زیر است:

$$X_{\mathrm{w}}(\mathrm{a},\mathrm{b}) = \int_{-\infty}^{\infty} \mathrm{x}(\mathrm{t})\, \Psi_{a,b}(t)dt \qquad (۱-۳)$$

که در آن $\Psi_{a,b}(\mathrm{t})$ به نام تابع پایه شناخته می‌شود. این تابع نسخه‌ای کشیده شده و جابجا شده از یک سیگنال میان گذر مانند $\Psi(\mathrm{t})$ است که تابع موجک مادر[2] نامیده شده و به صورت زیر تعریف می‌شود:

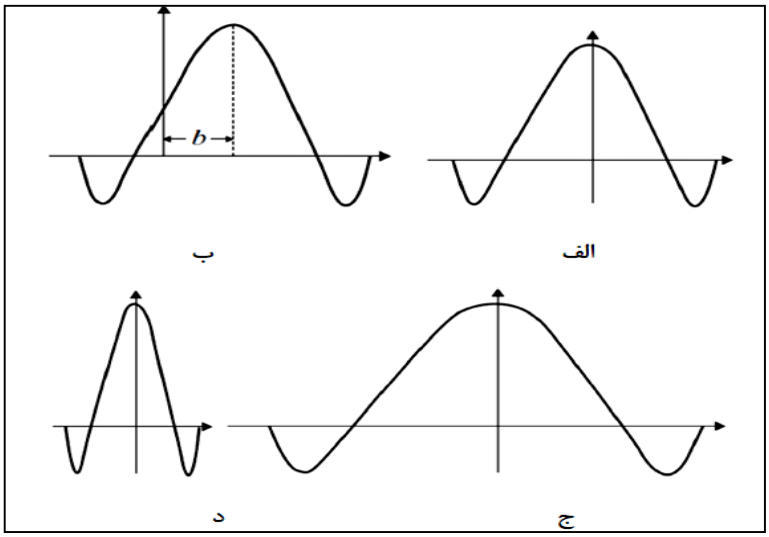
$$\Psi_{a,b} = \frac{1}{\sqrt{a}}\Psi(\frac{t-b}{a}) \qquad (۲-۳)$$

که در آن aوb به ترتیب پارامترهای اتساع[3] و جابجایی[4] نامیده می‌شوند. تأثیر این پارامترها در شکل (۳-۲) نشان داده‌شده است.

[1] Approximation Theory
[2] Mother Wavelet
[3] Dilation
[4] Translation

عرض تابع پایه با تغییر فاکتور اتساع (یا مقیاس) α به این شکل تغییر می‌کند که هرچه این فاکتور بزرگتر شود، عرض زمانی تابع پایه نیز بزرگتر می‌شود و بنابراین، عرض (یا پهنای باند) فرکانسی آن کمتر می‌شود؛ بنابراین به کمک این پارامتر می‌توانیم درجه تفکیک زمانی و فرکانسی را (با مصالحه‌ای که بین یکدیگر دارند) در تبدیل موجک تغییر دهیم و همین ویژگی جالب تبدیل موجک است که باعث می‌شود این تبدیل در تحلیل سیگنال‌هایی که ویژگی‌هایی با اندازه‌های مختلف دارند، مانند تصاویر طبیعی[1]، مناسب باشد. متناظر با هر اندازه‌ی ویژگی‌ای یک تابع پایه $\Psi_{a,b}(t)$ ، جود دارد که آن ویژگی به بهترین وجه توسط این تابع تحلیل می‌شود. برای مثال، در تصویری از یک خانه که شخصی از پنجره بیرون را نگاه می‌کند، تابع پایه‌ی متناظر با مقدار α بزرگ به راحتی کل خانه را تحلیل می‌کند اما برای تحلیل شخص و پنجره‌ی کنار آن باید از تابع مقیاسی با اندازه‌ی متوسط α استفاده کرد؛ همچنین، برای تحلیل نواحی مربوط به چشم شخص باید از تابع مقیاسی با اندازه کوچک α استفاده نمود. بنابراین، تبدیل موجک معادل با تحلیل سیگنال با فیلترهای میان گذری است که فرکانس میانی آن‌ها متغیر و وابسته به پارامتر α اما فاکتور کیفیت[2] آن‌ها ثابت است. توجه کنید که فاکتور کیفیت یک فیلتر میان‌گذر، نسبت فرکانس مرکزی به پهنای باند آن فیلتر تعریف می‌شود.

[1] Natural Images
[2] Quality Factor



شکل (۳-۲) تأثیر اتساع و جابجایی زمانی روی موجک مادر؛ (الف) موجک مادر $\Psi(t) = \Psi_{1,0}(t)$ با a=1 و b=0 (ب) حالت a=1 و $b \neq 1$ (ج) حالت a=2 و b=0 و (د) حالت a=1/2 و b=0 [2]

۳-۳-۱ تبدیل موجک گسسته

تبدیل موجک تعریف شده در رابطه ۳-۱ سیگنال یک بعدی X(t) را به تابعی دوبعدی مانند $X_w(a,b)$ نگاشت می‌کند و بنابراین موجب تولید تزاید زیادی می‌شود. سیگنال اولیه را می‌توان از روی تبدیل موجک و به ازای مقادیر گسسته‌ای از پارامترهای a و b به دست آورد[13] . پارامتر a را می‌توان با انتخاب $a = a_0^m$ (که a_0>۱ و m عددی صحیح است) گسسته کرد. با افزایش مقدار a پهنای باند (یا همان درجه تفکیک فرکانسی) تابع پایه کاهش می‌یابد و بنابراین برای پوشش یک ناحیه‌ی فرکانسی مشخص، نیاز به گام‌های بیشتری (به نام سلول‌های درجه تفکیک) است. به طور مشابه، گسسته سازی پارامتر b نیز معادل با نمونه‌برداری زمانی است که فرکانس نمونه‌برداری وابسته به پهنای باند سیگنال نمونه‌برداری شده دارد که این پهنای باند نیز به نوبه خود به طور معکوس متناسب با پارامتر a است. پارامتر b را می‌توان به صورت $b = nb_0a_0^m$ انتخاب کرد. اگر $a_0 = 1$ و $b_0 = 1$ انتخاب شوند، برخی از درجات آزادی انتخاب تابع $\Psi(t)$ وجود دارند که موجب می‌شوند توابع $\Psi_{m,n}(t)$ تشکیل یک پایه‌ی

یکه-متعامد[1] در فضای توابع مجذور- انتگرال پذیر دهند. این امر به این معنا است که هر تابع مجذور-انتگرال پذیر X(t) را می‌توان به صورت یک ترکیب خطی از توابع پایه به صورت زیر توصیف کرد :

$$x(t) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} a_{m,n} \Psi_{m,n}(t) \qquad (۳-۳)$$

که در آن ، ضرایب $a_{m,n}$ به نام ضرایب تبدیل موجک تابع X(t) نامیده شده و از روی رابطه ۳-۱ چنین محاسبه می‌شوند :

$$a_{m,n} = \int_{-\infty}^{\infty} x(t)\, \Psi_{m,n}(t) \qquad (۴-۳)$$

جالب است توجه شود که با هر افزایش مقدار m مقدار a دو برابر می‌شود (یعنی دو برابر شدن عرض یا پهنای باند زمانی و نصف شدن عرض یا پهنای باند فرکانسی). این ویژگی متناظر با تحلیل سیگنال به کمک تجزیه‌ی اکتاو باندهای فرکانسی و نیز تبدیل موجک دوتایی[2] است. با توجه به شکل (۳-۱) (که برای توضیح اصول عملکرد کدگذاری زیرباندی استفاده شد)، می‌توان نتیجه گرفت که تبدیل موجک در واقع نوعی کدگذاری زیرباندی است.

۳-۳-۲ نمایش چند درجه تفکیک

با استفاده از مفهوم تحلیل چند درجه تفکیک سیگنال[3] می‌توان کاربرد تبدیل موجک در کدگذاری تصویر را بهتر درک کرد. فرض کنید تابعی مانند $\Phi(t)$ وجود دارد طوری که مجموع توابع $\Phi(t-n), n \in Z$ یکه - متعامد باشند. همچنین فرض کنید که تابع $\Phi(t)$ جواب یک معادله تفاضلی دو مقیاسی به صورت زیر باشد :

$$\Phi_{(t)} = \sum_{n=-\infty}^{\infty} c_n \sqrt{2} \Phi(2t-n) \qquad (۵-۳)$$

که در آن

[1] Orthonormal
[2] Dyadic Wavelet Transform
[3] Multiresolution Signal Analysis

$$c_n = \int_{-\infty}^{\infty} \Phi(t)\sqrt{2}\Phi(2t-n)dt \qquad (۶-۳)$$

است.

فرض کنید که X(t) یک سیگنال مجذور- انتگرال پذیر باشد که بتوان آن را به صورت یک ترکیب خطی از توابع $\Phi\ (t-n)$ توصیف کرد:

$$x(t) = \sum_{n=-\infty}^{\infty} c_n \Phi(t-n) \qquad (۷-۳)$$

که در آن c_n ضرایب بسط بوده و حاصل افکنش[1] سیگنال X(t) بر روی توابع $\Phi(t-n)$ می‌باشند. از آنجا که اتساع[2] یک تابع، درجه تفکیک آن را تغییر می‌دهد، می‌توان سیگنال X(t) را بر حسب درجات تفکیک مختلف (که با اتساع و فشردن تابع $\Phi(t)$ ایجاد می‌شوند) نمایش داد. بنابراین می‌توان سیگنال X(t) را در هر درجه‌ی تفکیک m به صورت زیر نمایش داد :

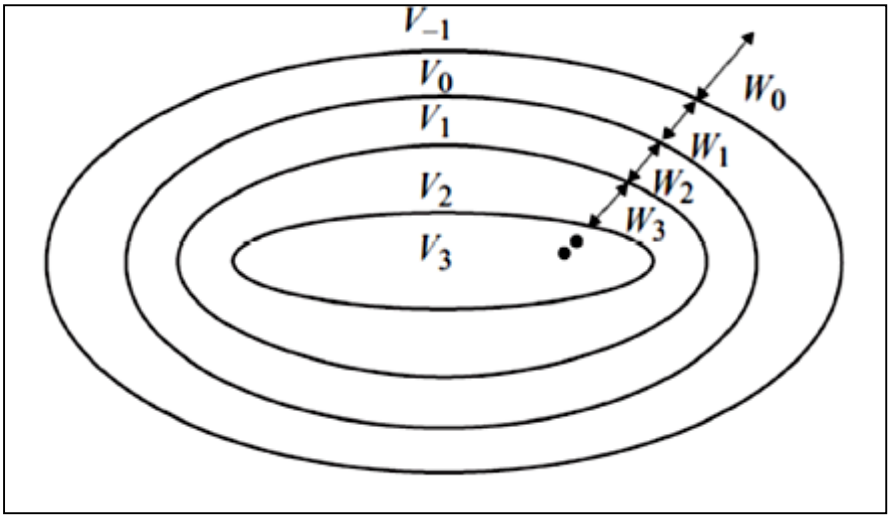
$$x_m(t) = 2^{-m/2} \sum_n c_n{}^m \phi(2^{-m}t - n) \qquad (۸-۳)$$

اگر فضای تشکیل شده از توابع $2^{-\frac{m}{2}}\Phi(2^{-m}t-n)$ را V_m بنامیم، با توجه به رابطه ۳-۵ می‌توان دید که تابع $\Phi_{(t)}$ به گونه‌ای است که به ازا هر i>j ، هر تابعی که متعلق به فضای V_i می‌باشد متعلق به فضای V_j نیز است و بدین ترتیب برای i>j داریم $V_i \subset V_j$ بنابراین، فضاهای متناظر با مقیاس‌های متوالی تودرتو بوده و هر فضای V_m (با m در حال افزایش) را می‌توان به صورت فضایی که درجه‌ی تفکیک آن رو به کاهش است، نگاه کرد.

در نتیجه، فضای متناظر با درجه تفکیک درشت تر[3] V_{j-1} را به دو زیر فضا می‌توان تجزیه کرد: یک فضا متناظر با درجه‌ی تفکیک ریزتر[4] و فضای دیگر، فضای W_j که فضای مکمل متعامد فضای V_j بوده و به گونه‌ای است که $V_j + W_j = V_{j-1}$ گردد. در این رابطه (به دلیل متعامد بودن دو فضای W_j و V_j)

[1] Projection
[2] Dilation
[3] Coarser Resolution
[4] Finer Resolution

داریم $W_j \perp V_j$. فضای W_j در واقع فضای ناشی از اختلافات بین درجات تفکیک نرم و درشت مذکور است و می‌توان از آن به صورت جزئیات لازم برای حرکت از فضای با درجه تفکیک کوچک‌تر، V_j به فضای با درجه تفکیک بزرگ‌تر، V_{j-1} یادکرد. سلسه مراتب فضاها از دیدگاه درجه تفکیک در شکل (۳-۳)نشان داده‌شده است.



شکل (۳-۳) فضاهای چند درجه‌ی تفکیک [2]

آقای مَلَت[1][14] نشان داد که در حالت کلی، پایه‌ی متناظر با فضای W_j را می‌توان به صورت جابه‌جایی‌ها و اتساع‌های یک تابع نمونه[2] به نام موجک[3] توصیف کرد. بنابراین، فضای W_m در واقع فضای تشکیل شده از توابع $\Psi_{m,n}(t) = 2^{-m/2}\Psi(2^{-m}t - n)$ می‌باشد. موجک $\Psi_t \in V_{-1}$ را می‌توان به صورت زیر از روی تابع $\Phi(t)$ به دست آورد:

$$\Psi_{(t)} = \sum_{\mathrm{n}=-\infty}^{\infty} (-1)^{\mathrm{n}} \mathrm{c}_{1-\mathrm{n}} \sqrt{2} \Phi(2\mathrm{t} - \mathrm{n}) \qquad (۹-۳)$$

[1] Mallat

[2] Prototype

[3] Wavelet

تابع $\Phi(t)$ را تابع مقیاس[1] مربوط به نمایش چند درجه‌ی تفکیک می‌نامند. بنابراین ضرایب تبدیل موجک رابطه۳-۴ متناظر با حاصل افکنش سیگنال X(t) بر روی فضای جزئیات متناظر با درجه‌ی تفکیک m، یعنی W_m می‌باشند. بنابراین، تبدیل موجک اساساً یک سیگنال را به فضاهای با درجات تفکیک مختلفی تجزیه می‌کند. در متون علمی، به این نوع تجزیه در حالت کلی، تجزیه‌ی چنددرجه‌ی تفکیک[2] اطلاق می‌گردد.

در ادامه مثالی از نحوه‌ی محاسبه‌ی موجک هار[3] به کمک تکنیک فوق، آورده می‌شود.

مثال (موجک هار) [2]

تابع مقیاس متناظر با موجک هار همان تابع معروف مستطیلی است :

$$\Phi(t) = \begin{cases} 1 & 0 \le t \le 1 \\ 0 & otherwise \end{cases}$$

تابع فوق در رابطه۳-۵ صدق کرده و ضرایب C_n مربوطه را می‌توان از رابطه۳-۶ صورت

$$c_n = \begin{cases} \frac{1}{\sqrt{2}} & n = 0,1 \\ 0 & otherwise \end{cases}$$

محاسبه کرد. بنابراین با استفاده از رابطه۳-۹ می‌توان موجک هار را به صورت زیر به دست آورد:

$$\Psi(t) = \Phi(2t) - \Phi(2t-1)$$

$$\Psi(t) = \begin{cases} 1 & 0 \le t \le 1/2 \\ -1 & 1/2 \le t \le 1 \end{cases}$$

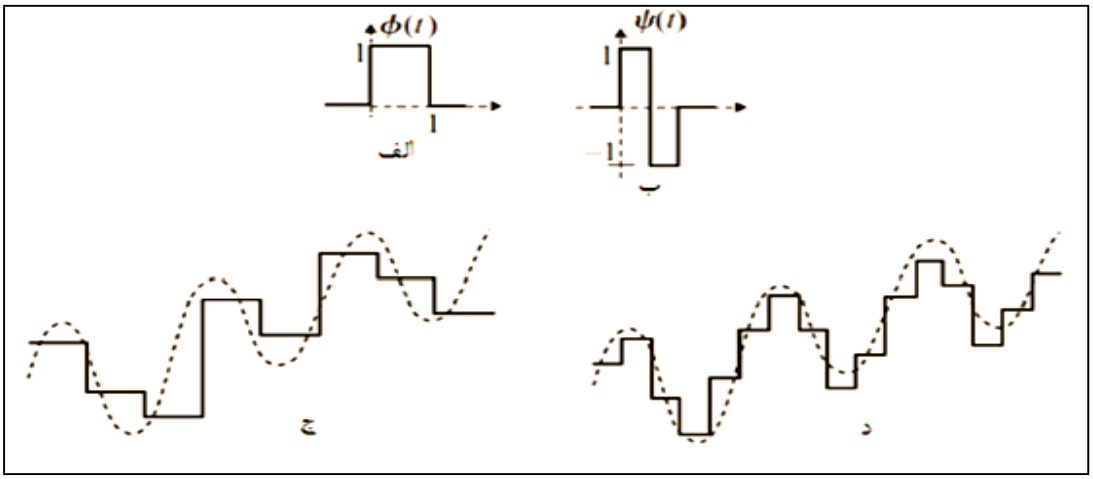
تابع مقیاس $\Phi(t)$ (تابع مستطیلی) و موجک هار متناظر با آن به ترتیب شکل (۳-۴) الف و ب نشان داده شده‌اند.

---

[1] Scaling Function

[2] Multi resolution Decomposition

[3] Haar Wavelet



شکل (۳-۴) (الف) تابع مقیاس هار .(ب) موجک هار . (ج) تقریب یک تابع پیوسته $\mathrm{X(t)}$ در مقیاس درشت تر $A_0\mathrm{x(t)}$ و (د) مقیاس نرم‌تر (یا بزرگ‌تر) $A_1\mathrm{x(t)}$ [2]

از دیدگاه تقریب (سیگنال) می‌توان تجزیه‌ی چند درجه‌ی تفکیکی را چنین توضیح داد. فرض کنید سیگنال $\mathrm{x(t)}$ در درجه تفکیک j توسط تابع $A_1\mathrm{x(t)}$ و از طریق بسط دادن به کمک توابع پایه‌ی متعامد تقریب زده شود. فضای W_j فضای اختلافات بین مقیاس درشت تر V_{j-1} و مقیاس نرم‌تر V_j می‌باشد. سیگنال $D_jX(t) \in W_j$ نیز اختلاف بین دو تقریب از سیگنال $\mathrm{x(t)}$ در دو درجه‌ی تفکیک j و j-۱ است. در واقع داریم : $D_\mathrm{j}\mathrm{x(t)} = \mathrm{A_{j-1}x(t)} - \mathrm{A_j x(t)}$. بنابراین سیگنال $\mathrm{x(t)}$ را می‌توان به دو قسمت تقسیم کرد:

$$x(t) = A_{-1}x(t) = A_0x(t) + D_0x(t)$$

شکل (۳-۴)—ج وشکل (۳-۴)—د نشان دهنده‌ی دو تقریب از یک تابع پیوسته در دو درجه‌ی تفکیک متوالی و به کمک تابع مقیاس مستطیلی می‌باشند. تقریب درشت تر$A_0x(t)$ در شکل (۳-۴)—ج و تقریب انجام‌شده در درجه‌ی تفکیک بزرگ‌تر، یعنی $A_1x(t)$ نیز در شکل (۳-۴)—د نشان داده‌شده اند. در یک تابع هموار و با تغییرات آهسته‌ی $\mathrm{x(t)}$ عمده‌ی تغییرات (که همان انرژی سیگنال می‌باشند) در جزء $A_0x(t)$ قرار گرفته و جزء $D_0x(t)$ تقریباً صفر می‌باشد. با تکرار فرآیند جداسازی فوق‌الذکر و انجام یک بخش‌بندی دیگر به صورت $A_0x(t) = A_1x(t) + D_1x(t)$ می‌توان تبدیل موجک

سیگنال x(t) را محاسبه کرده و تابع اولیه x(t) را می‌توان برحسب موجک‌های آن به صورت زیر بسط داد:

$$x(t) = D_0\, x(t) + D_1 x(t) + D_2 x(t) + \cdots + + D_n x(t) + A_n x(t) \quad (۱۰-۳)$$

که در آن پارامتر n بیانگر تعداد سطوح تجزیه است. از آنجا که دامنه‌ی تغییرات[۱] علامت‌های جزئیات $D_j\, x(t)$ بسیار کمتر از دامنه‌ی تغییرات تابع اولیه‌ی x(t) است، کدگذاری آن‌ها راحت تر از کدگذاری ضرایب بسط رابطه۳-۳ است.

۳-۳-۳ تبدیل موجک و فیلتر بانک

برای اینکه فرآیند جداسازی مکرر گفته شده در بخش قبل در عمل قابل پیاده‌سازی باشد باید الگوریتم موثری برای محاسبه‌ی $D_j\, x(t)$ از روی ضرایب بسط اولیه‌ی سیگنال x(t) در دست باشد. یکی از نتایج بخش‌بندی چنددرجه‌ی تفکیک فضا این است که تابع مقیاس $\Phi_{(t)}$ دارای خاصیت خود تشابهی است. اگر Φ(t) و $\overline{\Phi}$(t) به ترتیب توابع مقیاس تحلیل (یعنی مربوط به مسیر تجزیه) و ترکیب (یعنی مربوط به مسیر بازسازی) باشند و همچنین توابع $\Psi(t)$ و $\overline{\Psi}(t)$ به ترتیب موجک‌های تحلیل و ترکیب باشد، آن گاه از آن جا که $V_j \subset V_{j-1}$ می‌باشد، این توابع را می‌توان به صورت بازگشتی به صورت زیر تعریف کرد:

$$\Phi(\mathrm{t}) = \sum_{\mathrm{n}=-\infty}^{\infty} \mathrm{c_n}\sqrt{2}\Phi(2\mathrm{t}-\mathrm{n}) \quad (۱۱-۳)$$

$$\overline{\Phi}(\mathrm{t}) = \sum_{\mathrm{n}=-\infty}^{\infty} \overline{\mathrm{c}}_\mathrm{n}\sqrt{2}\overline{\Phi}(2\mathrm{t}-\mathrm{n})$$

$$\Psi(t) = \textstyle\sum_{\mathrm{n}=-\infty}^{\infty} \mathrm{d_n}\sqrt{2}\Phi(2\mathrm{t}-\mathrm{n})$$

[۱] Dynamic Range

$$\bar{\Psi}(t) = \sum_{n=-\infty}^{\infty} \bar{d}_n \sqrt{2} \Phi(2t - n)$$

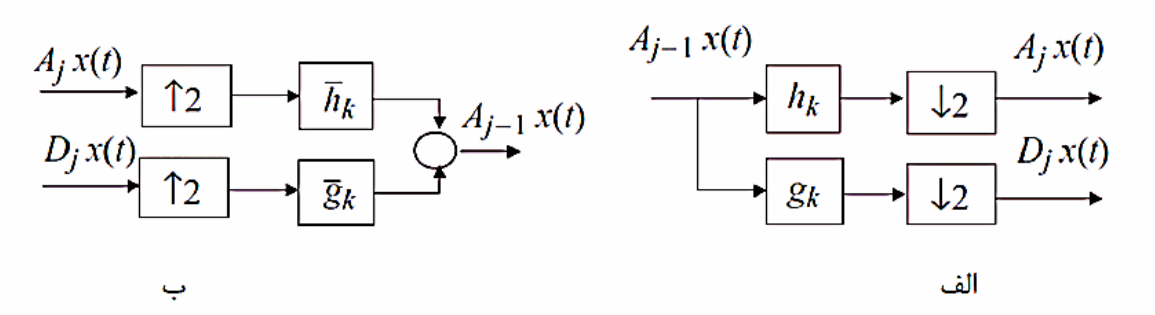
این روابط بازگشتی راهی را برای محاسبه ضرایب تقریب x(t) در درجه تفکیک j، یعنی $A_j x(t)$، و ضرایب سیگنال جزئیات، یعنی $D_j x(t)$ از روی ضرایب تقریب سیگنال در مقیاس بالاتر، یعنی $A_{j-1} x(t)$ در اختیار قرار می‌دهند. به کمک برخی روابط و دستکاری‌های ریاضی می‌توان نشان داد که هر دو گروه ضرایب تقریب و جزئیات مربوط به یک درجه تفکیک ریزتر را می‌توان از طریق کانولوشن کردن ضرایب تقریب سیگنال در درجه‌ی تفکیک درشت تر با یک فیلتر خاص و سپس زیرنمونه برداری سیگنال حاصل شده با نرخ ۲ به دست آورد. برای این کار، در مورد ضرایب تقریب مربوط به درجه تفکیک ریزتر (یا پایین تر[۱]) از یک فیلتر پایین گذر با ضرایب $h_k = c_{-k}$ و برای جزئیات نیز از یک فیلتر بالاگذر با ضرایب $g_k = d_{-k}$ استفاده می‌شود. به عکس، می‌توان سیگنال مربوط به درجه تفکیک بالاتر (یا درشت تر) را می‌توان از روی سیگنال جزئیات متناظر با آن درجه‌ی تفکیک و سیگنال تقریب مربوط به درجه‌ی تفکیک پایین تر بازیابی کرد. برای انجام این کار ابتدا ضرایب سیگنال‌های تقریب و جزئیات گفته شده را با نرخ ۲ بالانمونه برداری[۲] کرده، سپس به ترتیب فیلترهای ترکیب[۳] با ضرایب $\bar{h}_k = \bar{c}_{-k}$ و $\bar{g}_k = \bar{d}_{-k}$ کانولوشن کرده و در پایان حاصله‌ای به دست آمده را با هم جمع می‌کنیم. یک مرحله از هر کدام از فرآیندهای تفکیک (یا تجزیه) و ترکیب در شکل (۳-۵) نشان داده‌شده است. این شکل در واقع مشابه با بلوک دیاگرام مربوط به کدگذاری زیرباندی می‌باشد. در فرآیند تجزیه، سیگنال ورودی به دو مؤلفه باند بالا گذر و باند پایین گذر تفکیک شده و در نتیجه درجه‌ی تفکیک فرکانسی با ضرایب ۲ افزایش می‌یابد گرچه به دلیل استفاده از عمل زیرنمونه برداری (با نرخ ۲) درجه تفکیک زمانی با ضریب ۲ کاهش می‌یابد. در نتیجه، در مجموع هر مرحله از فرآیند تجزیه موجب بهتر / بیشتر شدن درجه‌ی تفکیک فرکانسی و کمتر شدن درجه‌ی تفکیک زمانی می‌گردد.

---

[۱] Lower Resolution

[۲] Up-Sampling

[۳] Synthesis

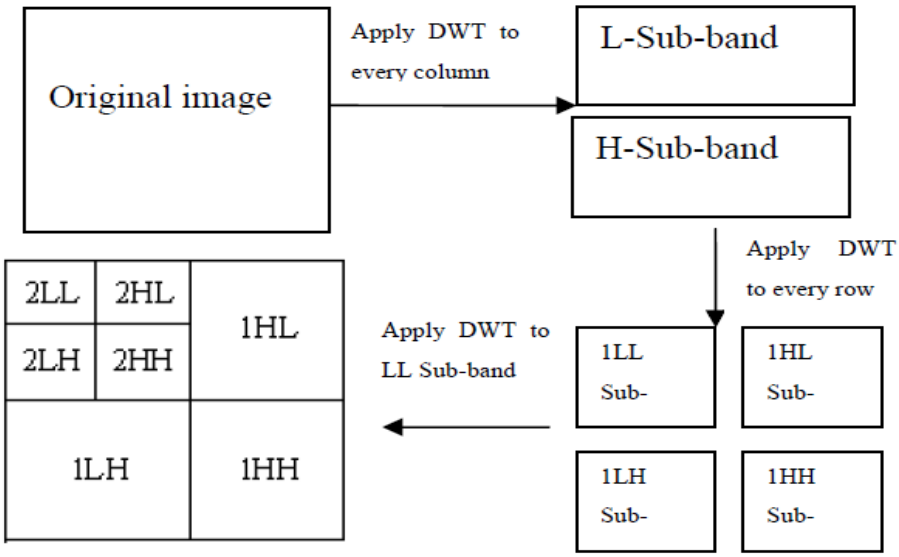


شکل (۳-۵) یک مرحله تبدیل موجک (الف) تحلیل (تجزیه) (ب) ترکیب

۳-۴ اعمال تبدیل موجک بر روی تصویر [2]

در واقع تصویر را می‌توان به عنوان یک سیگنال دو بعدی در نظر گرفت و با تعمیم ساختار فیلتر کردن دو باندی نشان داده‌شده در شکل (۳-۵) در هر بعد از ابعاد سیگنال مورد بررسی، می‌توان به تبدیل موجک چندبعدی[1] دست یافت. برای تجزیه‌ی یک تصویر (که سیگنالی دوبعدی محسوب می‌شود)، می‌توان فرآیند تجزیه‌ی یک بعدی را در هر کدام از ابعاد سطری و ستونی اعمال کرد. یک نمونه از چنین فرآیندی شکل (۳-۶) نشان داده‌شده است. این شکل فرآیند تجزیه تصویر با استفاده از تبدیل موجک را نشان می‌دهد. در ابتدا فیلتر پایین گذر به هر یک از سطرهای تصویر اعمال می‌شود، که در نتیجه آن اجزای فرکانس پایین سطر به دست می‌آید ولی از آنجایی که فیلتر پایین گذر یک فیلتر نیم باند می‌باشد داده‌های خروجی شامل فرکانس‌هایی فقط در نیمه ابتدایی بازه فرکانسی اصلی می‌باشد. نمونه‌های خروجی زیرنمونه برداری با نرخ ۲ می‌شوند. بنابراین اطلاعات خروجی شامل نصف تعداد نمونه‌های اصلی می‌باشد. سپس فیلتر بالا گذر نیز به همان ترتیب به سطرها اعمال و اجزای فرکانس بالا بدست می‌آید.

[1] Multidimensional Wavelet Transform



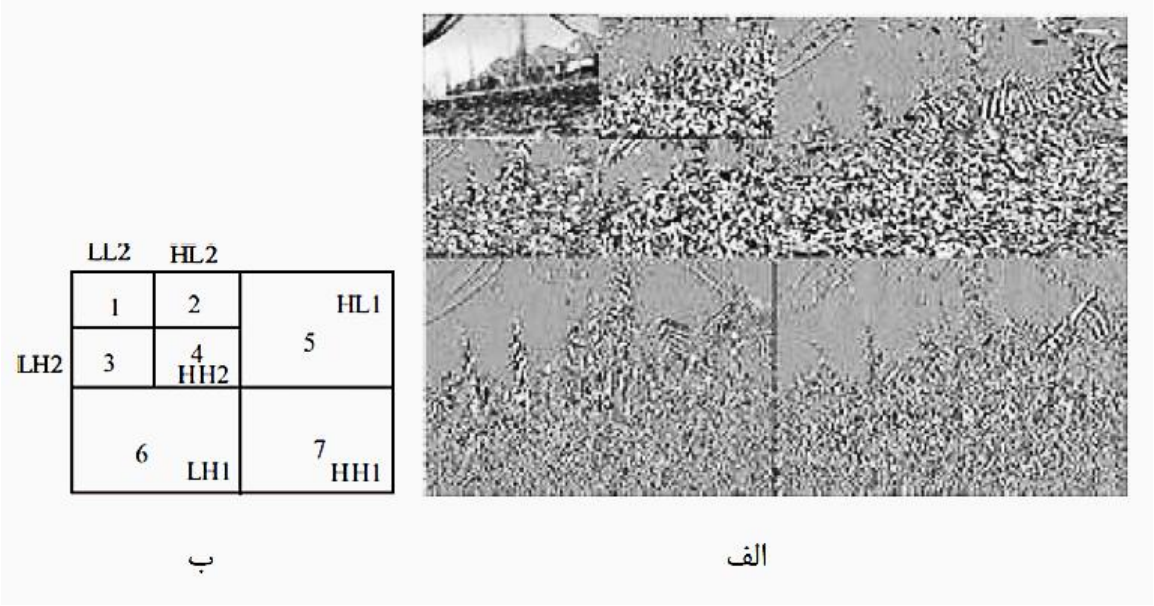
شکل (۳-۶) تبدیل موجک ۲ مرحله‌ای بر روی تصویر

در مرحله بعد فیلتر کردن به طریق گفته شده بر روی ستون‌های داده‌های خروجی مرحله قبل انجام می‌شود. این روند می‌تواند تا هر تعداد مرحله‌ای تکرار شود.

در شکل (۳-۶) نمادهای L و H به ترتیب بیانگر فیلترهای تجزیه‌ی پایین گذر و بالاگذر بوده که در آن‌ها از زیرنمونه برداری با نرخ ۲ نیز استفاده می‌شود. در اولین مرحله از این فرآیند تجزیه‌ی دوتایی[1] سه زیرتصویر با محتویات فرکانس بالا تولید می‌شوند. برای مثال، زیرتصویر 1LH عمدتاً دارای جزئیات فرکانس بالا (در راستای) عمودی و جزئیات فرکانس پایین (در راستای) افقی می‌باشد. این ویژگی‌ها در مورد زیرتصویر 1HL برعکس می‌باشند. زیر تصویر 1HL در هر دو راستای افقی و عمودی دارای جزئیات فرکانس بالا می‌باشد. جزئیات تصویر در درجه‌ی تفکیک پایین تر (یعنی مرحله‌ی دوم از فرآیند تجزیه‌ی نشان داده‌شده شکل (۳-۶) با نمادهای 2LH، 2HL و 2HH نمایش داده می‌شوند. باند 2LL یک تصویر زیرنمونه برداری شده‌ی پایین گذر است که یک نسخه‌ی رونوشت برداری (کپی) شده از روی تصویر اصلی در ابعاد کوچکتر می‌باشد. حاصل اعمال فرآیند نشان داده‌شده که در شکل (۳-۶) روی یک نمونه تصویر شکل (۳-۷) الف نشان داده‌شده است (تصویر اصلی در این شکل نشان داده

[1] Dyadic Decomposition

نشده است). در این فرآیند ، هفت زیر تصویر تولید می‌شود که نام و جایگاه هر کدام در شکل (۳-۷)ب مشخص شده است.

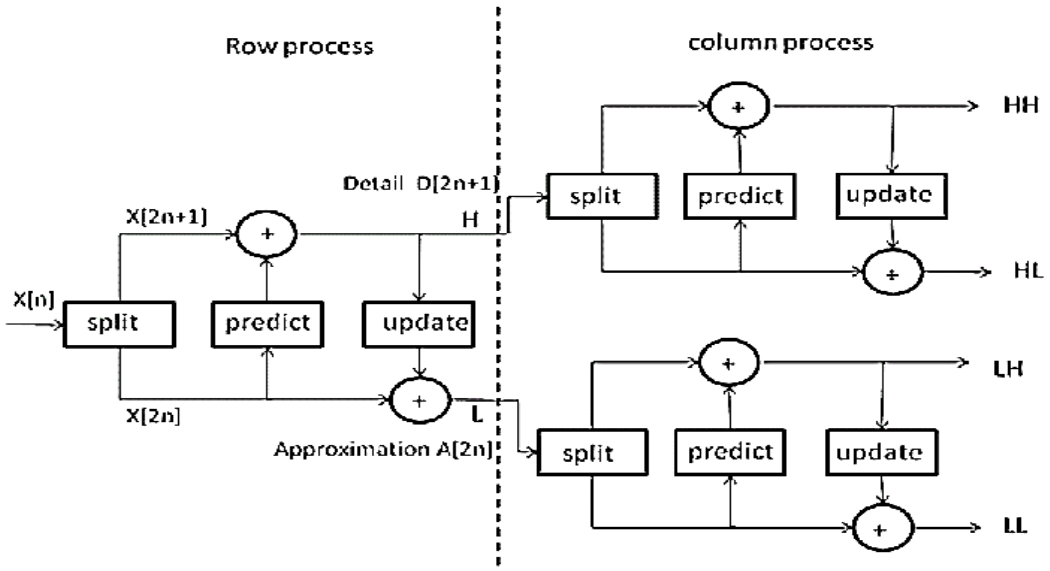


شکل (۳-۷) (الف) هفت زیرتصویر تولید شده توسط فرآیند نشان داده‌شده درشکل (۳-۶) (ب) چیدمان باندهای مختلف متناظر با فرآیند شکل (۳-۶)

۳-۵ تبدیل موجک به روش طرح بالابری[1]

روش جایگزین دیگر برای تبدیل موجک روش طرح بالابری می‌باشد، که ابتدا توسط ویم سویلدنز[2] معرفی شد [15] . همانطور که در بخش‌های قبل ذکر شد تبدیل موجک با استفاده از کانولوشن نیاز به حافظه و محاسبات زیادی دارد، بنابراین روش مناسبی برای پیاده‌سازی سخت‌افزاری در مواقعی که میزان حافظه و حجم محاسبات قابل انجام به شدت محدود است نمی‌باشد. در این مواقع طرح بالابری روش مناسبی به حساب می‌آید. در شکل مراحل طرح بالا بر دو بعدی که شامل پیش‌بینی و بروز رسانی است دیده می‌شود.

[1] Lifting Scheme
[2] Wim Sweldens



شکل (۳-۸) تبدیل به روش طرح بالابری دوبعدی

مزیت دیگر طرح بالابری این است که تبدیل معکوس آن نیز از همین روش پیروی میکند. این روش فقط نیاز به جمع و شیفت اعداد صحیح دارد که پیاده‌سازی آن را بسیار آسان می‌کند. برای موجک 5/3 مرحله پیش‌بینی و بروزرسانی در رابطه ۳-۱۲ تا ۳-۱۵ آورده شده‌اند [16].

$$D_0 = D_0 + D_0 - S_0 - S_1 \quad (۱۲-۳)$$

$$S_0 = S_0 + \left(2 * \frac{D_0}{8}\right) \quad (۱۳-۳)$$

$$D_i = D_i + D_i - S_i - S_{i+1} \quad (۱۴-۳)$$

$$S_i = S_i + ((D_{i-1} + D_i)/8) \quad (۱۵-۳)$$

در معادلات مذکور D_i پیکسل‌های فرد و S_i پیکسل‌های زوج که از یک سطر یا ستون گرفته شده‌اند می‌باشند. هر سطر یا ستون به صورت زیر می‌باشند.

$S_0\ D_0 S_1\ D_1 S_2\ D_2$ ....

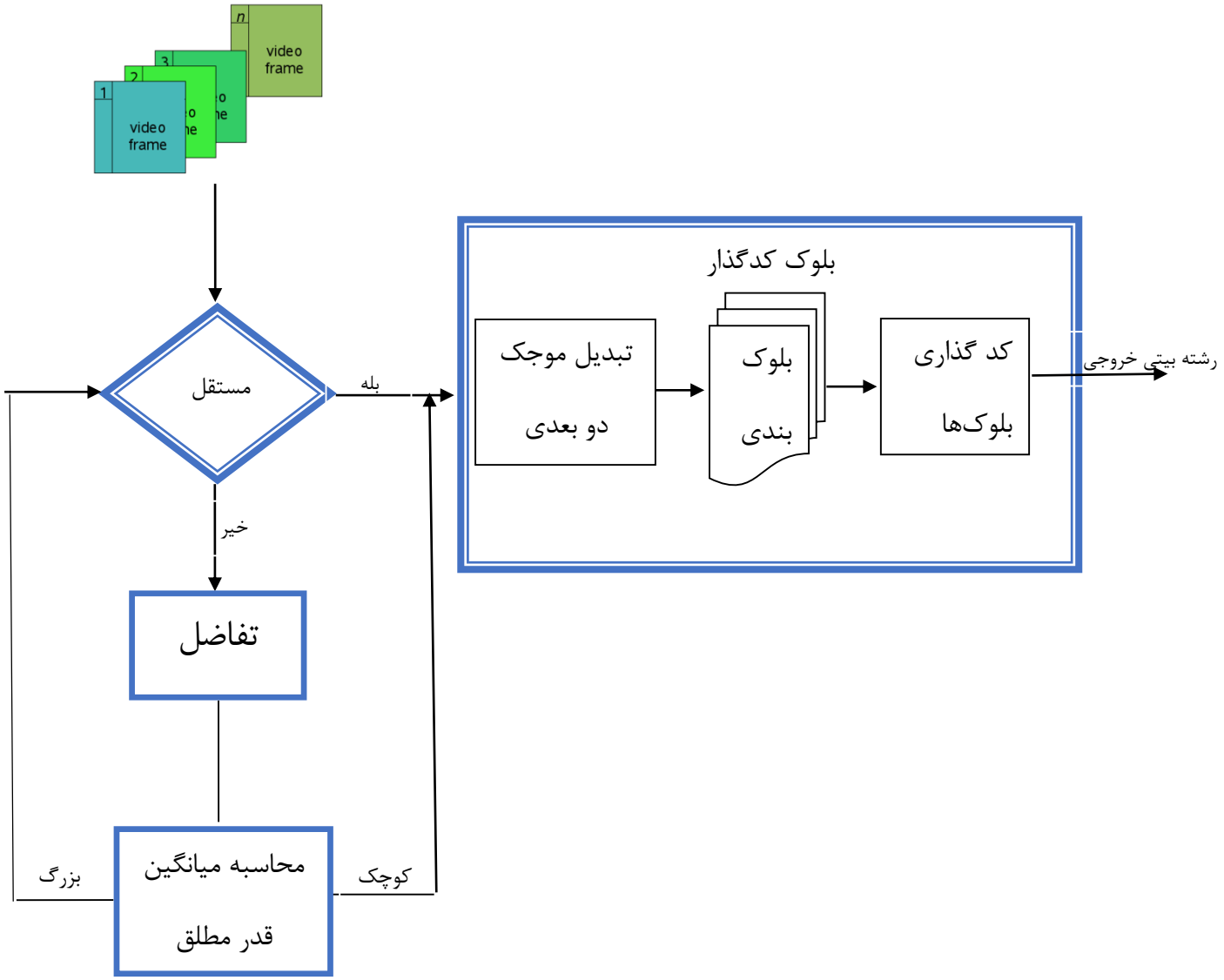
برای محاسبه تبدیل موجک نیاز است که پیکسل‌های سطرها یا ستون‌ها در یک زمان گرفته شوند در رابطه(۳-۱۴)و رابطه (۳-۱۵) نیاز است که D_i قبل از S_i محاسبه شود، بنابراین پیکسل‌های فرد باید ابتدا پردازش شوند و سپس پیکسل‌های زوج. در هر مرحله تبدیل موجک ابتدا سطر و سپس ستون‌های تصویر پردازش می‌شوند و در هر مرحله طول سیگنال نسبت به مرحله قبل نصف می‌شود.

فصل چهارم

الگوریتم مورد استفاده

مقدمه

در این فصل روش پیشنهادی به منظور فشرده‌سازی برخط ویدیو معرفی می‌شود. همانطور که در شکل (۴-۱) نشان داده شده، روش پیشنهادی دارای یک سری اجزاء و تکنیک‌هاست که مهم ترین تکنیک‌ها، ابتدا تعیین مستقل یا وابسته بودن فریم ورودی، مشخص کردن تعلق داشتن یا نداشتن فریم‌ها به یک نما، استفاده از تبدیل موجک طرح بالابری، بلوک بندی تصویر و کدگذاری پیشنهادی که در ادامه هر بخش به‌صورت کامل توضیح داده می‌شود.



شکل(۴-۱) بلوک دیاگرام روش پیشنهادی

۴-۱ تعیین مستقل یا وابسته بودن فریم‌ها

در روش مورد استفاده از هر n فریم یک فریم مستقل و مابقی وابسته در نظر گرفته می‌شوند فریم‌های وابسته بر اساس اطلاعات فریم‌های مستقل کدگذاری می‌شوند. زمانی که از دو فریم مشابه تفاضل بگیریم حاصل بدست آمده کوچک می‌شود در نتیجه بودجه بیتی کمتری برای کدگذاری آن مورد نیاز است، پس مقدار بزرگتر n می‌تواند از این نظر مفید باشد، از طرفی میزان بزرگ n موجب تَسَری خطا شده و خطای کلی کار را بالا می‌برد، زیرا اگر در کدگذاری یک فریم مستقل خطا وجود داشته باشد با انتخاب n بزرگ این خطا به‌صورت زنجیری به همه فریم‌های وابسته منتقل می‌شود. در نتیجه برای انتخاب n باید یک مصالحه بین میزان خطا و میزان فشرده‌سازی برقرار کرد. در روش پیشنهادی مقدار n به‌صورت تجربی بدست می‌آید.

۴-۲ تفاضل گیری

در روش‌های تخصصی تر فشرده‌سازی ویدیو قبل از تفاضل گیری ابتدا از تخمین بردارهای جابجایی[1] و جبران‌سازی حرکتی استفاده می‌شود، علاوه بر این‌ها از روش تشخیص نما نیز استفاده می‌شود چون اگر بدون در نظر گرفتن تغییر نما تفاضل بگیریم اگر دو فریم مربوط به دو نمای مختلف باشند تفاضل خیلی بزرگ می‌شود و کارایی به شدت پایین می‌آید. در الگوریتم پیشنهادی به دلیل مسائل مربوط به پیاده سازی سخت افزاری و استفاده از FPGA ارزان قیمت برای پایین آوردن حجم محاسبات از روش‌های گفته شده استفاده نمی‌شود و فقط از هر n فریم متوالی یک فریم مستقل و از فریم‌های وابسته نسبت به فریم مستقل تفاضل گرفته شده و میانگین قدرمطلق ماتریس بدست آمده را محاسبه می‌کنیم. اگر این میانگین از حد مشخصی بیشتر باشد فریم مورد نظر را مستقل در نظر می‌گیریم. همچنین در هر فریم در دنباله بیتی یک بیت برای مشخص کردن مستقل یا وابسته بودن در نظر گرفته می‌شود.

[1] Motion Vector Estimation

۳-۴ الگوریتم موجک مورد استفاده

فیلتر مورد استفاده یک مدل اصلاح شده از موجک دو متعامدی[1] (۲،۲) از نوع CDF[2] می‌باشد. اگر تعداد 2k عنصر در هر سطر و ستون موجود باشد، بنابراین به تعداد k ضریب موجک پایین گذر و k ضریب بالا گذر به وجود می‌آید. معادلات تجزیه فیلترها به صورت زیر می‌باشند.

ضرایب بالا گذر $g(k) = x(2k+1) - \frac{1}{2}(x(2k) + x(2k+2))$

ضرایب پایین گذر $f(k) = x(2k) + \frac{g(k+1)+g(k)}{4}$

که در آن g(k) ، k امین ضریب بالاگذر می‌باشد. و f(k)، k امین ضریب پایین می‌باشد و x(k) مکان k امین پیکسل ورودی را نشان می‌دهد. شکل(۴-۱) چینش ضرایب برای نمونه های سطری ورودی با ۵۱۲ عنصر را نشان میدهد. نمونه های مرزی نیز به وسیله گسترش متقارن درست می‌شوند.

0 1	.....	254 255	256 257	.....	510 511

Pixel Data

f0 f1	.....	f254 f255	g0 g1	.....	g254 g255

Coefficient Data

شکل(۴-۲) ترتیب قرارگیری ضرایب در سطر

ضرایب باید به طریقی باز نویسی شوند که همه ضرایب با فرکانس پایین (f) از ضرایب فرکانس بالا (g) جدا شوند. که در شکل(۴-۲) نشان داده‌شده است. در آن داده‌های میانگین و تفاضل دسته بندی شده‌اند. بعد از اعمال فیلتر به تمام سطرها، همان فیلتر مجدداً به ستون‌ها نیز اعمال می‌شود. و سطرها نیز به همان ترتیب چیده می‌شوند.

[1] Biorthogonal
[2] Cohen-Debuchies-Feaveu

۴-۴ الگوریتم پیاده سازی شده

با توجه به پیچیدگی محاسباتی کم این روش انتخاب مناسبی برای پیاده‌سازی در بستر سخت افزار ارزان قیمت انتخاب شده می‌باشد. از آنجاکه میزان بالا تراکم انرژی در ذات تبدیل موجک است. ضرایب بزرگتر در زیرباندهای پایین قرار میگیرند، درحالیکه منطقه بزرگی که ضرایب در زیرباندهای بالا میباشند به نسبت کوچک هستند. کد گذاری این مناطق بر پایه تقسیم بلوکی صفحه بیتی کارآمد تر می‌باشد.

در الگوریتم بلوکی موجود هر زیر باند به صورت جدا با استفاده از کدگذاری صفحه بیتی با شروع از با ارزش ترین بیت تا کم ارزش ترین بیت کد گذاری می‌شود. هر صفحه بیتی با استفاده از دو مرحله کد گذاری می‌شود مرحله مهم[1] و مرحله پالایش[2] بعد از کد گذاری تمام زیر باندها، رشته بیتی به صورت بهینه مرتب می‌شود. نقش هریک از بسته‌های رشته بیتی متعلق به هر زیر باند در میانگین مربع خطای[3] تصویر مقایسه و بسته بیتی متعلق به زیر باند با بزرگترین نسبت میانگین مربع خطای برای اولویت خروج مشخص می‌شوند.

در شکل(۴-۳) که سه مرحله موجک بر روی تصویر اعمال شده می‌توان مشاهده کرد بیشتر انرژی در کوچک‌ترین زیر باند متمرکز شده. و بیشتر ضرایب در زیر باندهای بالا کوچک می‌باشند

[1] Significant pass
[2] Refinement pass
[3] Mean square error

Original image

ب

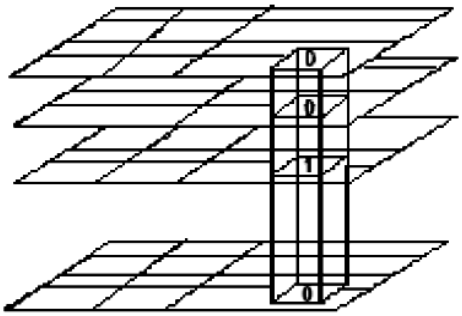
Wavelet coefficients after transformation

الف

شکل(۴-۳) یک مثال از پراکندگی ضرایب الف: ضرایب موجک بعد از تبدیل ب: تصویر اصلی

شکل(۴-۴) ساختار صفحه بیتی را نشان می‌دهد.

0	5	3	0	-1	-6	-8	-2
0	6	37	12	13	6	-16	7
4	-3	65	34	23	28	-23	5
2	0	45	48	64	7	12	1
0	0	14	22	34	-4	11	0
0	0	-23	12	7	0	-5	0
24	33	-6	-9	7	0	12	0
0	0	0	0	0	0	0	16



شکل(۴-۴) مثال از صفحه بیتی[17]

هدف کد گذاری بلوکی، کد کردن مناطق بزرگ ضرایب که احتمال وجود صفحه های بیتی صفر در آن‌ها زیاد می‌باشد به روش کارآمد و سریع است. برای مثال می‌توان یک بلوک کامل صفر ۱۶*۱۶ را با فقط یک صفر که نماینده ۲۵۶(۱۶*۱۶) ضریب می‌باشد کد کرد. این روش از روش‌های کد گذاری

SPIHT[1],SBHP[2] و EBCOT[3] کمک گرفته ولی تفاوت‌های نیز با آن‌ها دارد. ما روش‌های تقسیم مختلفی برای کدگذاری بلوک‌های کوچک غیر صفر در نظر گرفته ایم. که کارایی بالا و پیچیدگی محاسباتی کم داشته باشد. روش کد گذاری به صورت زیر خلاصه می‌شود.

۱- انتخاب یکی از سه روش برای تقسیم یک بلوک به زیر بلوک‌ها با اندازه های یکسان، بر اساس تعداد و پراکندگی بیت‌های پرارزش در بلوک؛

۲- هر زیر بلوک باید کد گذاری شود. هر زیر بلوک صفر با ۰ کد می‌شود و هر زیر بلوک غیر صفر با ۱.

۳- این رویه را به ترتیب به هر زیر بلوک اعمال کرده تا به بلوک غیر صفر با سایز ۲*۲ برسیم، سپس یکی از ۱۶ علامت را برای مشخص کردن بلوک ۲*۲ استفاده می‌کنیم

۴-۴-۱ سه روش تقسیم بلوک

در این قسمت می خواهیم سه روش تقسیم بلوک را توضیح دهیم. فرض کنید تصویر x به وسیله چند سطح موجک تجزیه شود. تعداد صفحه های بیتی برای هر زیر بلوک به وسیله بیشینه مقدار ضریب c(i,j) در آن بلوک مشخص می‌شود. هر ضریب اگر در رابطه زیر صدق کند می‌توان گفت ضریب مهم است.

$$2^n \leq c(i,j) < 2^n$$

در غیر اینصورت می‌توان گفت ضریب مورد نظر غیر مهم می‌باشد.

شکل(۴-۵) نمونه یک صفحه بیتی می‌باشد. در این مثال فقط سه بیت مهم وجود دارد. و همه بیت‌های دیگر صفر می‌باشند.

[1] Set Partitioning in Hierarchical Trees
[2] Subband Block Hierarchical Partitioning
[3] Embedded Block Coding with Optimal Truncation Points

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1+	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1-	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1+	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

شکل(۴-۵) نمونه یک صفحه بیتی

مراحل زیر برای تقسیم صفحه بیتی به زیر بلوک‌ها می‌باشد. تقسیم صفحه بیتی به شکل غیر متعارف ناکارآمد خواهد بود، زیرا چنین کاری پیچیدگی کار را بالا می‌برد. هدف جدا کردن بیت‌های پر ارزش از کم ارزش به صورت کارآمد در حین داشتن بزرگترین زیر بلوک‌های صفر ممکن است.

۱-اگر تمام یک بلوک صفر باشد [0] برای نشان دادن آن فرستاده می‌شود

۲-اگر فقط یک بیت مهم در بلوک باشد. از مختصات آن نقطه استفاده می‌شود. و [10] را برای آن می‌فرستیم.

۳-اگر بیشتر از یک بیت مهم باشد بلوک را به زیر بلوک‌های کوچکتر تقسیم کرده و این را با [11] نشان می‌دهیم.

آمار های استخراج شده از تصاویر مختلف نشان می‌دهد که سه روش بالا کارآمد و منطقی می‌باشند.

حال با ذکر یک نمونه به مقایسه این روش، با روش‌های درخت چهارتایی[1] می‌پردازیم. شکل(۴-۶) یک نمونه بلوک ۸*۸ را نشان می‌دهد که در آن فقط یک بیت مهم می‌باشد و به روش درخت چهارتایی تقسیم شده است. در این نمونه از روش مختصات برای کد کردن بیت مهم استفاده می‌کنیم. این بلوک، یک بلوک ۸*۸ است و برای نشان دادن مختصات x و y آن سه بیت برای هر کدام و در کل ۶ بیت نیاز است. و یک بیت برای علامت. در نتیجه تعداد کل بیت‌ها به صورت زیر است:

۱ که نشان دهنده این است که بلوک مذکور بلوک مهم است، ۱۰ که نشان دهنده روش مختصات است، ۰۰۱۰۱۰ برای مشخص کردن مختصات بیت پر ارزش، ۱ برای مشخص کردن علامت، در نتیجه بیت‌های خروجی برابر [1100010101] می‌باشند که تعداد آن‌ها ۱۰ عدد است. ولی اگر میخواستیم از روش درخت چهارتایی برای کدگذاری این بلوک استفاده کنیم نتیجه خروجی [11000010000101] می‌شد که تعداد بیت‌های آن ۱۴ می‌باشد. در نتیجه روش مورد استفاده چهار بیت کمتر مصرف کرده است. که اگر سایز بلوک بزرگتر از ۸ میبود تعداد بیت‌های ذخیره شده بیشتر می‌شد.

	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0

شکل(۴-۶) یک نمونه بلوک ۸*۸

در نمونه بلوک‌های دیگر نیز نسبت به روش درخت چهارتایی بهبود دیده می‌شود.

[1] Quad-tree

۴-۵ یک مثال ساده

در این بخش الگوریتم را با یک مثال ساده بطور عملی توضیح می‌دهیم. در شکل(۴-۷) یک بلوک ۱۶*۱۶ از ضرایب یک تصویر نمونه بعد از تبدیل موجک نشان داده‌شده، که در ادامه به توضیح الگوریتم بر روی آن می‌پردازیم.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	6	18	-5	4	1	0	0	2	0	0	0	0
1	0	0	0	0	0	12	-5	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	-2	0	0	4	0	0	-1	0	0	0
3	0	0	5	0	0	0	0	0	24	0	0	0	0	2	3	1
4	0	2	23	16	0	0	0	0	0	12	-5	0	0	0	-1	0
5	0	0	11	0	0	0	0	23	0	-2	23	-6	0	0	0	0
6	0	0	0	0	0	6	18	0	0	0	-36	22	-8	0	0	0
7	0	0	0	0	23	12	5	0	0	-12	-31	1	8	3	0	0
8	0	0	-6	-12	-56	32	3	0	0	-6	0	4	4	-20	5	0
9	0	0	1	0	-23	33	-6	0	12	48	3	16	63	-36	12	0
10	0	0	0	0	0	20	-4	0	6	50	12	0	14	22	8	0
11	0	1	-1	0	0	6	-16	0	0	21	13	53	37	11	3	0
12	0	0	0	-3	-13	-48	-42	12	0	8	0	12	0	4	0	1
13	0	3	0	0	0	-10	34	8	13	45	51	22	0	12	-11	0
14	0	0	5	0	3	0	7	1	0	12	23	0	-3	-38	-46	6
15	0	-2	0	0	0	0	0	0	0	0	0	0	0	0	8	0

شکل(۴-۷) بلوک نمونه ۱۶*۱۶ ضرایب

بیشترین ضریب در این بلوک ۶۳ است. در نتیجه نیاز به ۶ صفحه بیتی داریم. کد گذاری از بیت پر ارزش شروع و تا آخرین صفحه بیتی ادامه پیدا میکند. در شکل(۴-۸) صفحه بیتی پر ارزش بلوک ۱۶*۱۶ ذکر شده نشان داده شده است. در ابتدا چون این بلوک تمام صفر نیست و همچنین فقط یک بیت مهم ندارد، آن را به چهار زیر بلوک تقسیم کرده و در هر زیر بلوک روند کار را ادامه می‌دهیم.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1-	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1-	1+	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1+	0	0	0	1+	0	0	1+	1-	0	0
10	0	0	0	0	0	0	0	0	0	1+	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1+	1-	0	0	0
12	0	0	0	0	0	1-	1-	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	1+	0	0	1+	1+	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1-	1-	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

شکل(۴-۸) صفحه بیتی پر ارزش ضرایب شکل(۴-۷)

در ابتدا عدد ۱۱ را برای نشان دادن شیوه تقسیم بلوک را در خروجی قرار می‌دهیم. و در ادامه عدد صفر را برای نشان دادن بلوک تمام صفر در نتیجه خروجی برابر [110] است.

در بلوک ۸*۸ بعدی که در شکل نشان داده‌شده فقط یک بیت مهم وجود دارد. در نتیجه باید در ابتدا ۱۰ را به عنوان شیوه کد گذاری و سپس مختصات بیت مهم را در خروجی قرار می‌دهیم. و خروجی بعد از این مرحله به صورت [-11010110010] است.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	1-	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1-	1+	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1+	0	0	0	1+	0	0	1+	1-	0	0
10	0	0	0	0	0	0	0	0	0	1+	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1+	1-	0	0	0
12	0	0	0	0	0	1-	1-	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	1+	0	0	1+	1+	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	1-	1-	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

شکل(۴-۹) زیر بلوک دوم

در ادامه برای زیر بلوک بعدی، زیر بلوک را به چهار بلوک ۴*۴ تقسیم و برای هر بلوک مراحل فوق را تکرار می‌کنیم. تا در نهایت به یک بلوک ۲*۲ برسیم. در این مرحله اعداد موجود آن بلوک ۲*۲ را در خروجی قرار می‌دهیم. در نهایت با اعمال یک کد گذاری هافمن بر روی رشته بیتی خروجی اطلاعات کد گذاری شده و برای ارسال آماده می‌شوند.

فصل پنجم

سخت افزار طراحی شده

مقدمه

در این فصل مدار طراحی و ساخته شده را بطور بلوکی مجزا کرده، در ابتدا توضیح کلی در رابطه با اهداف طراحی، نحوه انتخاب قطعات و پارامترهای موثر در ساخت داده می‌شود و سپس هر بلوک بطور مجزا توضیح داده‌شده و نحوه کار آن شرح داده می‌شود.

۵-۱ اهداف و روند طراحی

هدف از طراحی برد مورد نظر، ساختن یک بستر سخت‌افزاری است که بتوان از طریق آن به طور مستقل ویدیو را دریافت و سپس به طور بلادرنگ فشرده‌سازی کرده و داده‌ها را به رایانه ارسال کرد. برای برآورده کردن نیازهای محاسباتی بالای فشرده‌سازی ویدیو پردازش موازی راهکار مناسبی برای بالا بردن سرعت و رسیدن به پردازش بلادرنگ می‌باشد.

پس از تعریف کامل نیازهای طراحی، در مرحله بعد باید به دنبال قطعات موجود برای برآورده کردن نیازها رفت و طراحی اولیه را بر اساس قطعات موجود در بازار کامل کرد. پس از انتخاب کامل قطعات و مشخص کردن تمام تراشه‌ها از جمله پردازنده، حافظه‌ها و مدارات ارتباطی، اقدام به رسم مدار طراحی‌شده در نرم‌افزار Altium Designer می‌کنیم. برای انجام یک طراحی دقیق و مناسب نیاز است تا برگه اطلاعات[1] تمام تراشه‌ها به دقت مطالعه شود و ملاحظات طراحی مربوطه از قبیل تغذیه و دیکوپلینگ، فوت‌پرینت[2] و نحوه ارتباط با دیگر قطعات مدارات و همچنین قرارگیری در ساختار کلی مدار مد نظر قرار گیرد.

پس از طراحی شماتیک[3] مدار، بر اساس اطلاعات فنی برای پیاده‌سازی طرح مورد نظر باید اقدام به طراحی برد مدار چاپی (PCB[4]) نمود. برای طراحی PCB طرح شماتیک مورد نظر را که در نرم‌افزار Altium Designer طراحی‌شده، به قسمت PCB انتقال می‌دهیم. اگرچه نرم‌افزار Altium Designer

[1] Data Sheet
[2] Foot Print
[3] Schematic
[4] Printed Circuit Board

امکاناتی برای مسیریابی خودکار[1] PCB از روی مدار شماتیک دارا می‌باشد ولی با توجه به اینکه طرح مورد نظر پیچیده است، مسیریابی خودکار خطای بالایی دارد. در نتیجه برای حصول اطمینان از صحت کارکرد برد مدارچاپی طرح مورد نظر به صورت دستی مسیریابی می‌شود. در مسیریابی و طراحی PCB باید چندین پارامتر مهم را مدنظر قرار داد از جمله این قوانین می‌توان به موارد زیر اشاره کرد:

- در نظر گرفتن فاصله بین مسیرها و رعایت محدودیت‌های ساخت بر اساس هزینه و امکانات موجود و همچنین فرکانس کاری مدار.
- تعیین تعداد لایه‌های برد بر اساس بودجه، نیازها و توصیه‌های موجود در برگه اطلاعات.
- چینش مناسب قطعات و در نظر گرفتن محدودیت‌های مونتاژ.

از آنجا که فرکانس کاری مدار نسبتاً بالا می‌باشد باید مسائل مربوط به طراحی مدارات فرکانس بالا در نظر گرفته شود. (در ادامه در مورد ملاحظات فرکانسی توضیح داده می‌شود.) فاصله بین خطوط علاوه بر تأثیر بر پارامترهای القایی و اهمی در روند ساخت برد و هزینه نیز تأثیرگذار است. از آنجا که برد مورد نظر به صورت تکی تولید می‌شود و بودجه ساخت آن بسیار محدود است باید محدودیت‌های موجود در بازار برای ساخت آن در شرایط موجود را در نظر گرفت که از جمله آن‌ها رعایت فاصله مناسب بین خطوط و همچنین تعداد لایه‌های برد است. برد به صورت دولایه متالیزه و در ابعاد ۱۰*۱۰ سانتیمتر به صورت چاپ سبز با مارکاژ ساخته شده است. بعد از تحویل برد ساخته شده تمام اتصالات آن چک شده و از صحت اتصالات برد اطمینان حاصل شد.

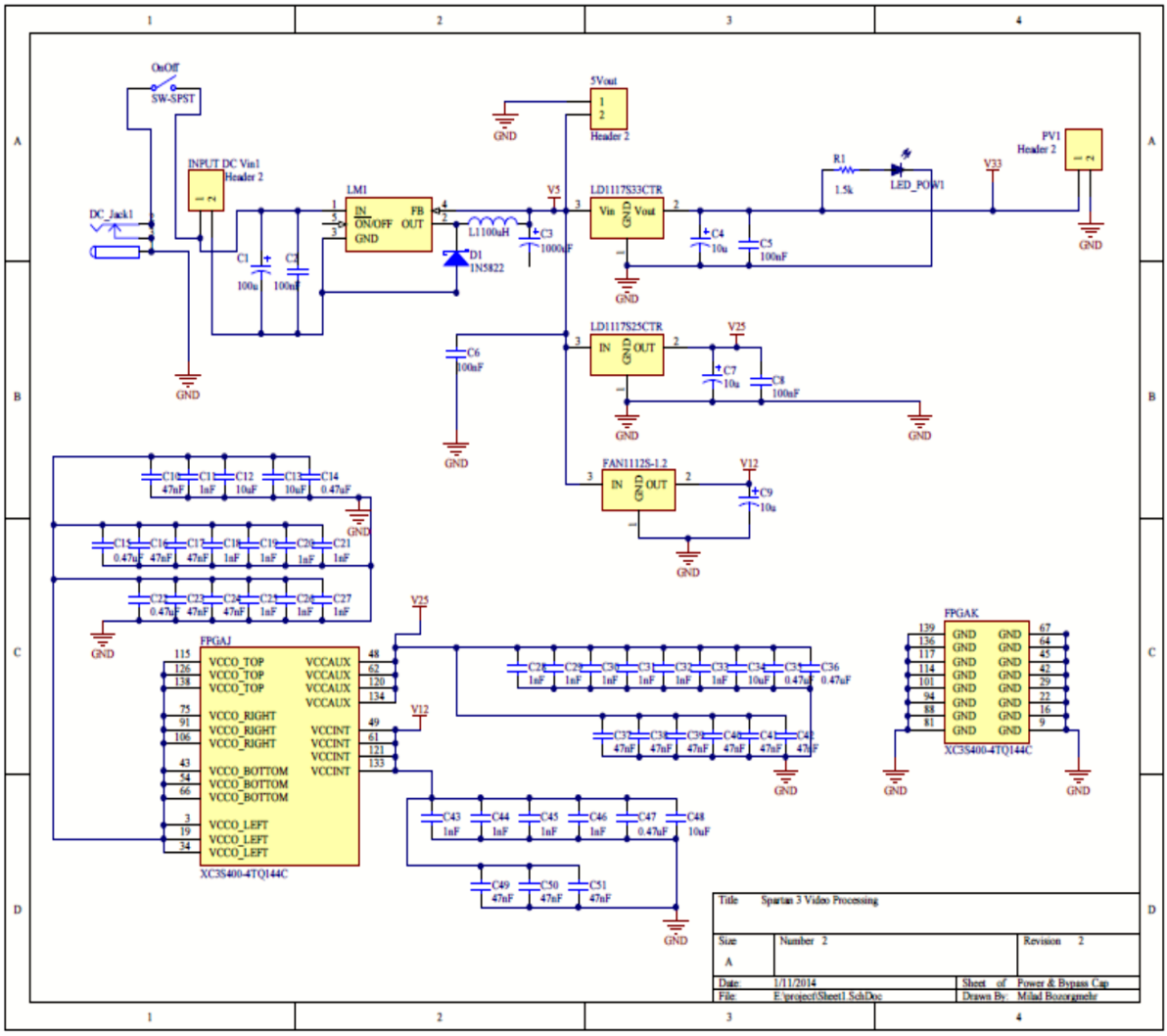
مونتاژ قطعات برد به صورت دستی انجام شده. که کار بسیار وقت‌گیر و نیازمند دقّت فراوان است. بعد از اتمام مونتاژ، اتصالات مجدّداً چک شده تا در صورت بروز اتصال کوتاه یا قطعی در اثر مونتاژ مشکل مرتفع گردد. در پایان چند برنامه نمونه برای آزمون همه قسمت‌های مدار و ساختار کلی طرح نوشته‌شده و برنامه راه اندازی همه عناصر مدار از جمله دوربین، رابط USB و رَم نوشته شده و صحت کارایی قطعات مدار آزموده می‌شوند.

[1] Auto Route

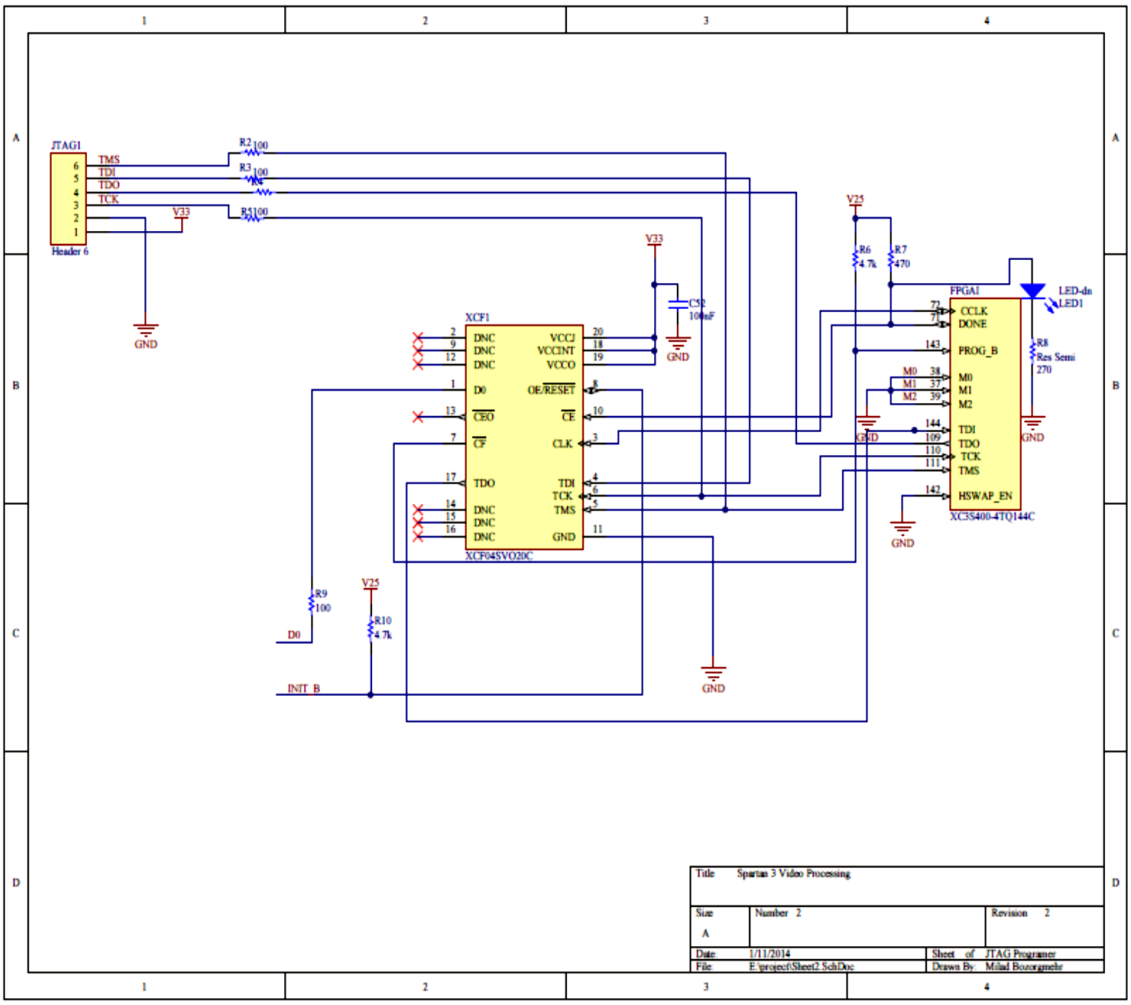
۵-۲ ساختار کلی و جزئی برد طراحی شده

شماتیک مدار طراحی شده در چهار صفحه[1] مجزا ترسیم شده است. درشکل (۵-۱) تا شکل (۵-۴) چهار صفحه شماتیک نشان داده شده‌اند. همانطور که در شکل‌ها قابل مشاهده است هر صفحه مربوط به یک یا چند قسمت طرح است که از طریق نام گذاری با صفحه‌های دیگر مرتبط شده است. در شکل (۵-۵) طرح PCB نهایی آورده شده‌است؛ همچنین ساختار کلی برد طراحی‌شده به صورت دیاگرام بلوکی بر روی تصویر از برد نهایی نشان داده‌شده است. همان طور که در شکل قابل مشاهده است، تمام اجزای برد به طور کلی مشخص شده‌اند. المان‌های مانند سوئیچ‌ها و LEDها برای آزمون طراحی و همچنین برای استفاده‌های آینده در برنامه در نظر گرفته شده‌اند. همچنین پورت USB[2] برای ارتباط با خارج از برد در نظر گرفته شده. مدارات دیکوپلینگ که به صورت بلوکی نشان داده‌شده به‌صورت پراکنده در برد اصلی قرار گرفته‌اند. بلوک‌های آورده شده در زیر بخش‌های مربوطه به طور کامل توضیح داده‌شده‌اند.

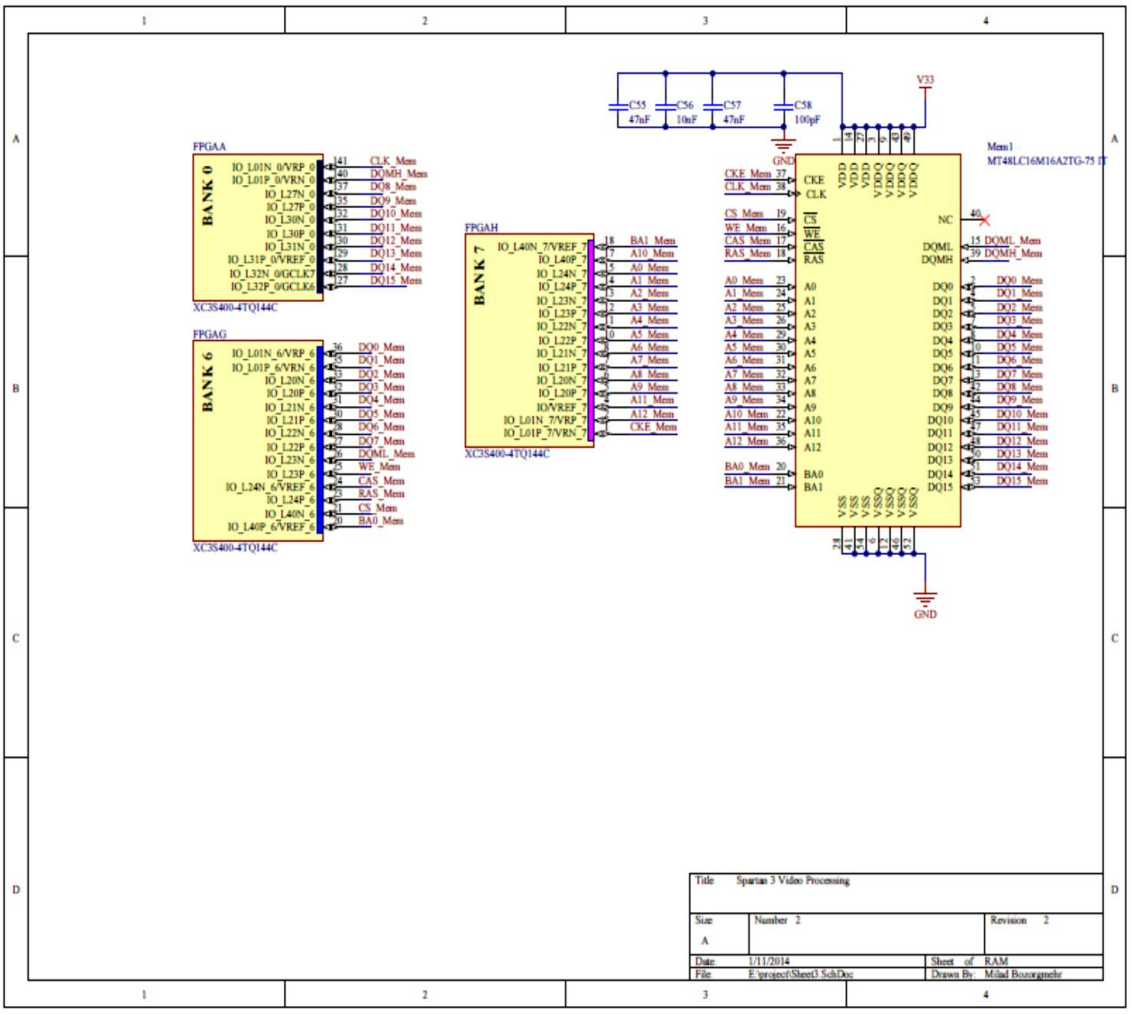
[1] Sheet
[2] Universal Serial Bus



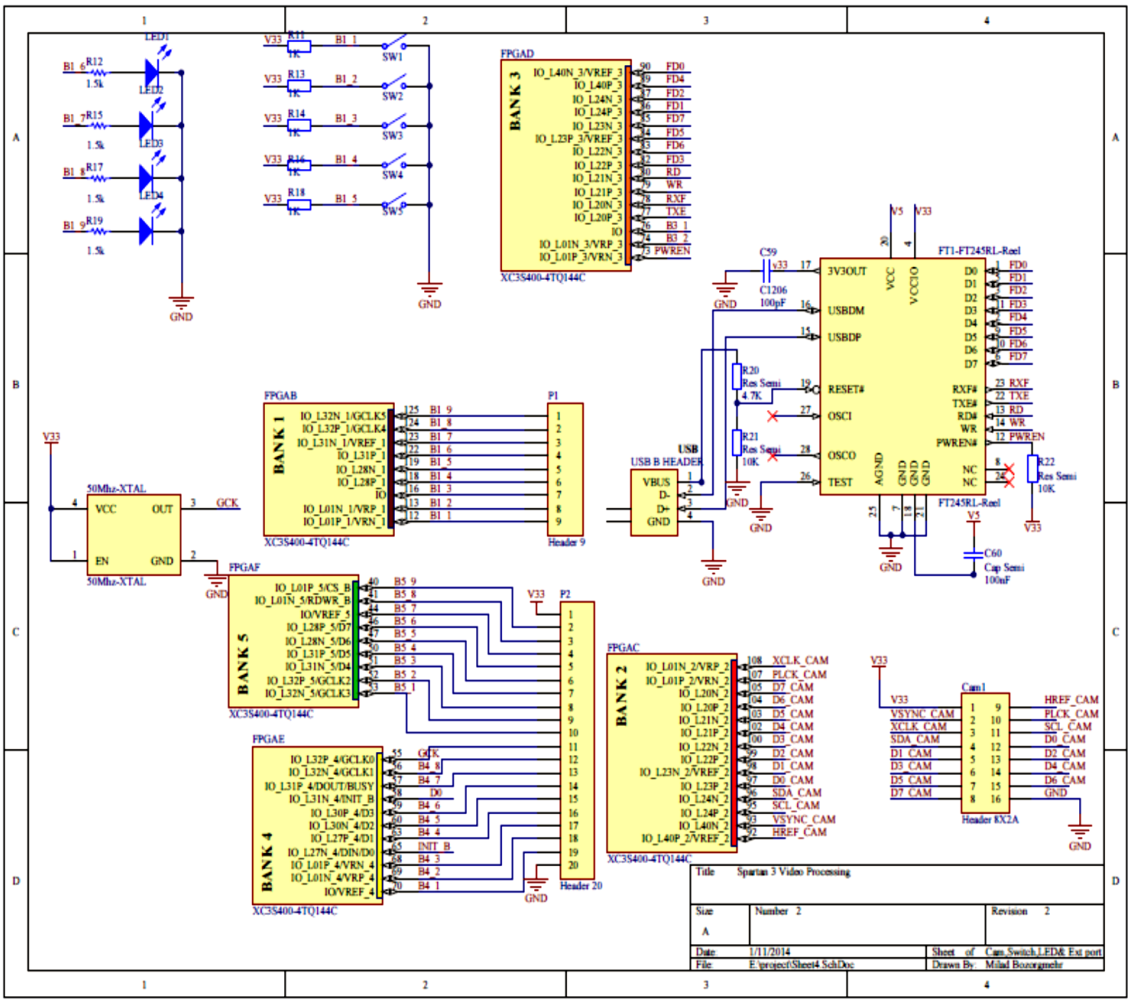
شکل (۵-۱) صفحه اول شماتیک مربوط به بخش تغذیه



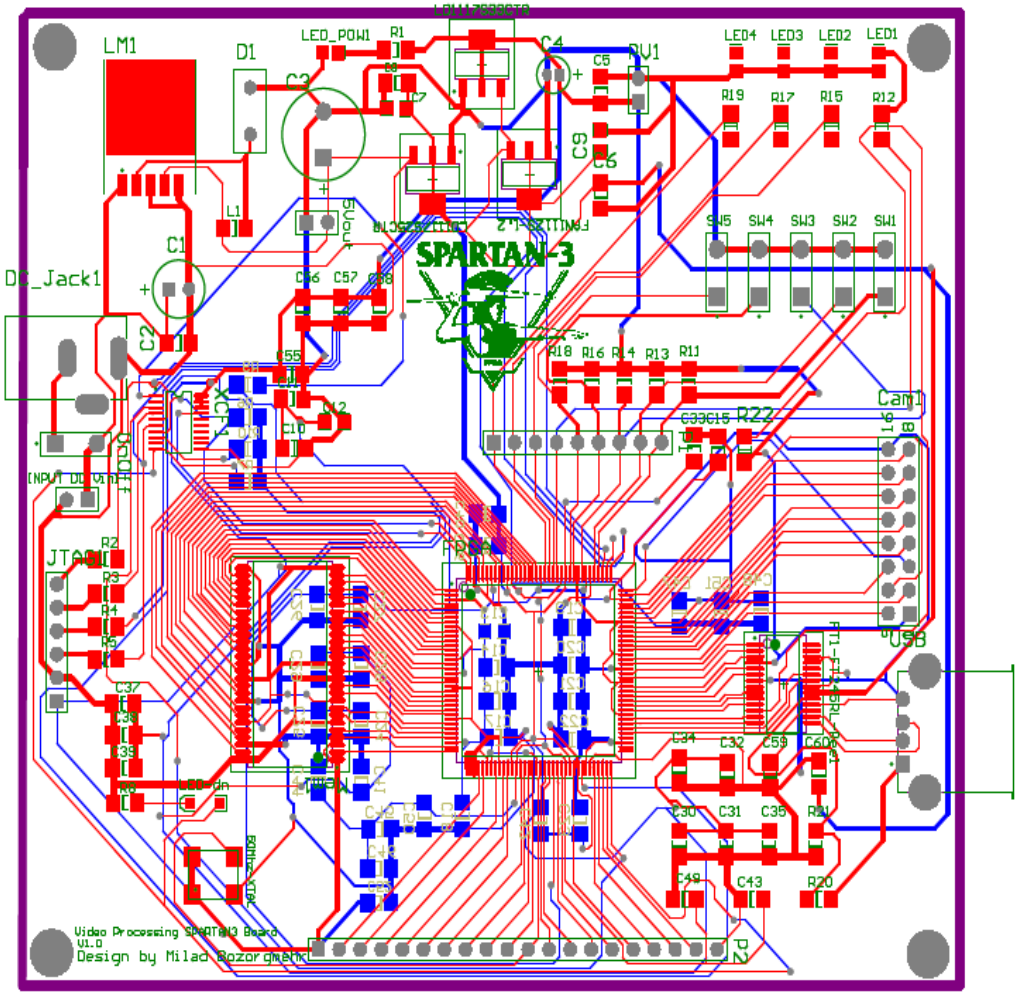
شکل (۵-۲) صفحه دوم شماتیک مربوط به ROM و پروگرمر JTAG



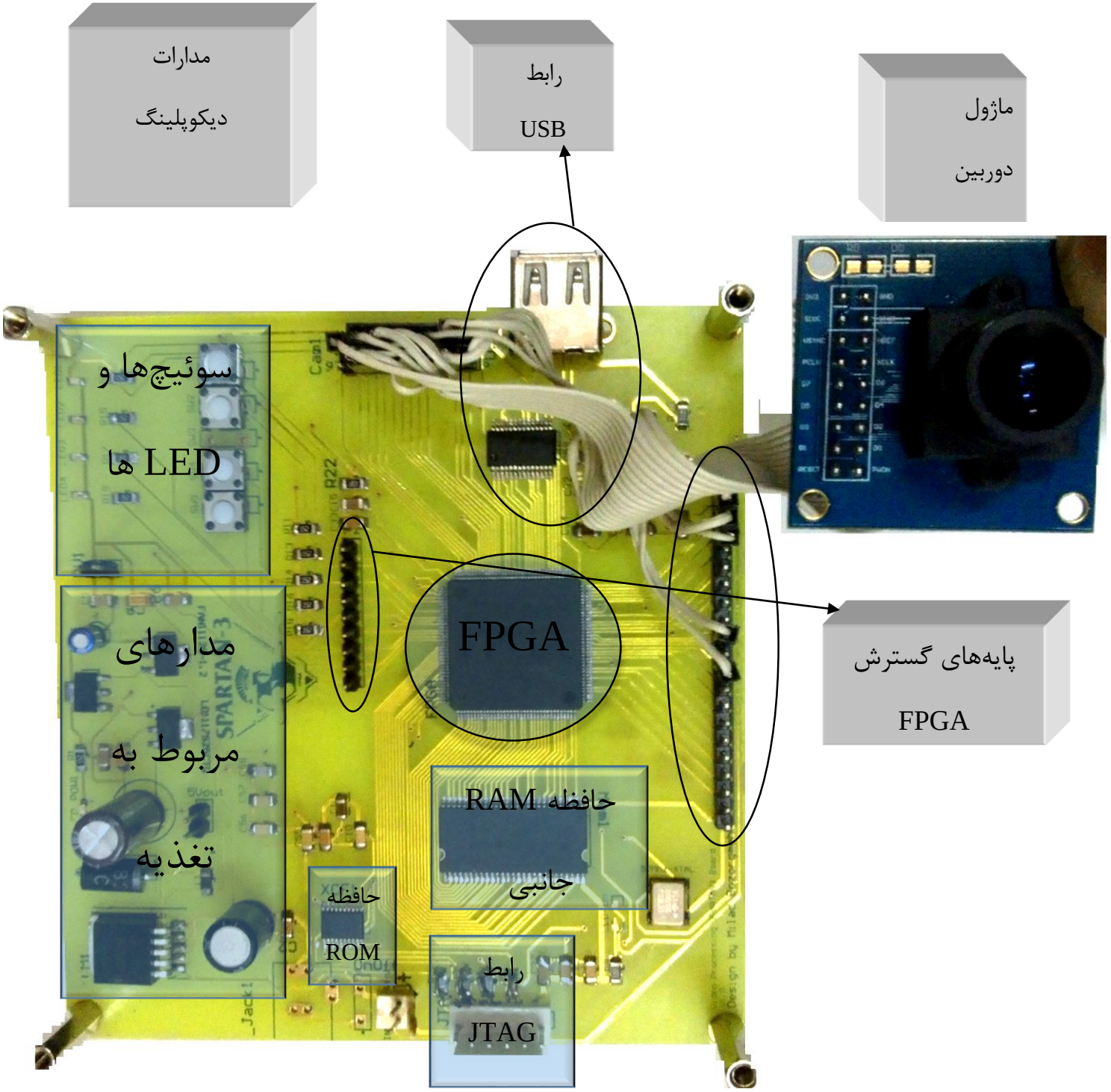
شکل (۵-۳) صفحه سوم شماتیک مربوط به RAM



شکل (۵-۴) صفحه چهارم شماتیک مربوط به سوئیچ‌ها، LEDها، پورت دوربین و رابط USB



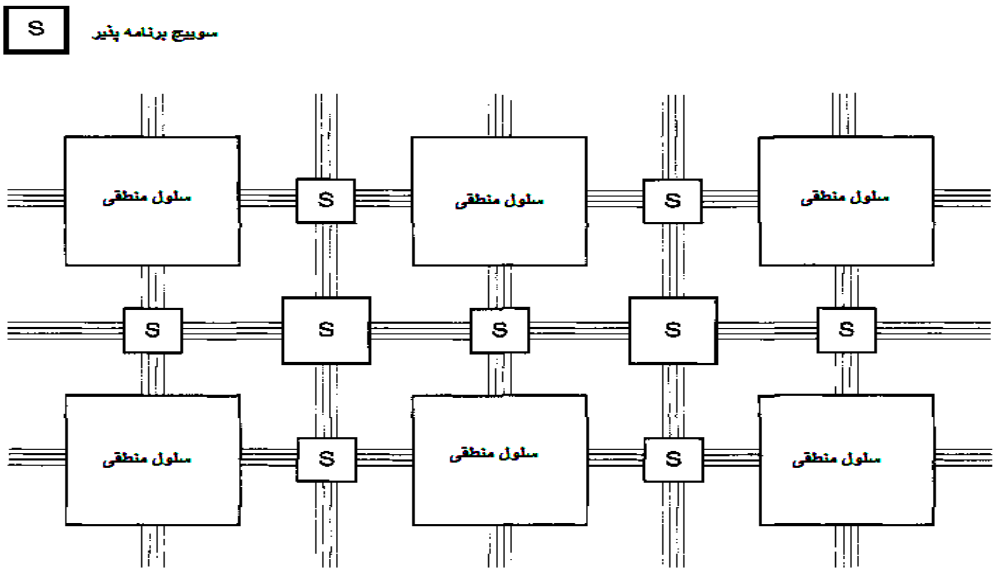
شکل (۵-۵) PCB نهایی



شکل (۵-۶) شمای کلی طرح

۳-۵ تراشه (FPGA)

یک آرایه گیتی برنامه‌پذیر در محل[1] (FPGA) یک قطعه منطقی است که حاوی یک آرایه دو بعدی از سلول‌های منطقی کلی[2] و سوییچ‌های برنامه پذیر بوده که در شکل (۵-۷) نشان داده‌شده است. یک سلول منطقی می‌تواند برنامه‌ریزی شود تا اتصال میان سلول‌های منطقی به وجود آید. یک طرح سفارشی می‌تواند با مشخص نمودن عمل هر سلول منطقی و ایجاد انتخابی اتصال هر سوئیچ برنامه پذیر انجام شود. پس از تکمیل طراحی و ساخت می‌توان آرایش را به FPGA منتقل نمود و مدار مورد نظر را به دست آورد. چون این فرآیند می‌تواند به جای ایجاد در یک دستگاه جداگانه ساخت (Fabrication) در محل استفاده (field) تولید گردد. قطعه را به نام برنامه پذیر در محل نام گذاشته‌اند [18].



شکل (۵-۷) ساختار ادراکی FPGA

ساختار یک FPGA به طور کلی شامل ماتریس قابل برنامه‌ریزی بسیار پیچیده‌ای از مدارهای مجتمع منطقی، ثبات‌ها، RAM و منابع قابل مسیریابی است که این منابع می‌توانند برای اجرای عملیات منطقی و حسابی، ذخیره‌سازی متغیرها و انتقال داده بین اجزاء مختلف سیستم برنامه‌ریزی و استفاده شوند. به

[1] Field Programmable Gate Array
[2] generic

علاوه هیچ کنترل مرکزی و توالی اجرای دستورات وجود ندارد. و در عمل هزاران عملیات به طور موازی بر روی یک FPGA در یک سیکل کلاک قابل اجراست. در نتیجه اگرچه فرکانس کاری FPGA ممکن است کمتر از پردازنده‌های سری باشد ولی کارایی آن در بسیاری از کاربردها از قبیل پردازش تصویر و ویدیو با آن‌ها قابل‌مقایسه و در بعضی موارد بیشتر است.

۵-۳-۱ FPGA مورد استفاده در طرح پیشنهادی

امروزه FPGA های بسیار پیچیده و با قابلیت‌های فراوانی موجود می‌باشد که امکانات پردازشی بالایی از جمله فرکانس کاری بسیار بالا، میزان حافظه بالا، تعداد گیت‌های سیستمی بالا و ... را دارا می‌باشند. ولی در این طرح با توجه به بودجه محدود و FPGA های موجود در بازار FPGA مدل SPARTAN 3 از شرکت Xilinx مورد استفاده قرار گرفت که در ادامه توضیح مختصری در مورد آن داده می‌شود.

۵-۳-۲ مروری بر معماری SPARTAN 3

معماری خانواده اسپارتان ۳ شامل پنج عنصر اصلی برنامه پذیر می‌باشد[19] :

• بلوک‌های منطقی آرایش‌پذیر (CLBs): این بلوک‌ها، شامل جداول LUT مبتنی بر رَم هستند که می‌توان برای پیاده‌سازی عناصر ذخیره پذیر و منطقی که به صورت فلیپ فلاپ یا لچ[1] مورد استفاده قرار گیرند. بلوک‌های CLB توانایی برنامه‌ریزی شدن برای اجرای گستره وسیعی از توابع منطقی همانند ذخیره کننده اطلاعات را دارند. هر CLB دارای چهار برش[2] به هم متصل است که به صورت جفت به هم متصل هستند.

• بلوک‌های ورودی / خروجی (IOB): این بلوک‌ها جریان اطلاعات بین پین‌های O/I و منطق درونی قطعه را کنترل می‌کنند. هر O/I از مسیر داده دو طرفه به علاوه توان عملیاتی سه حالته پشتیبانی می‌کند.۲۴ استاندارد سیگنال مختلف، شامل ۸ استاندارد تفاضلی با کارایی بالا موجود می‌باشند.

[1] Latch
[2] Slice

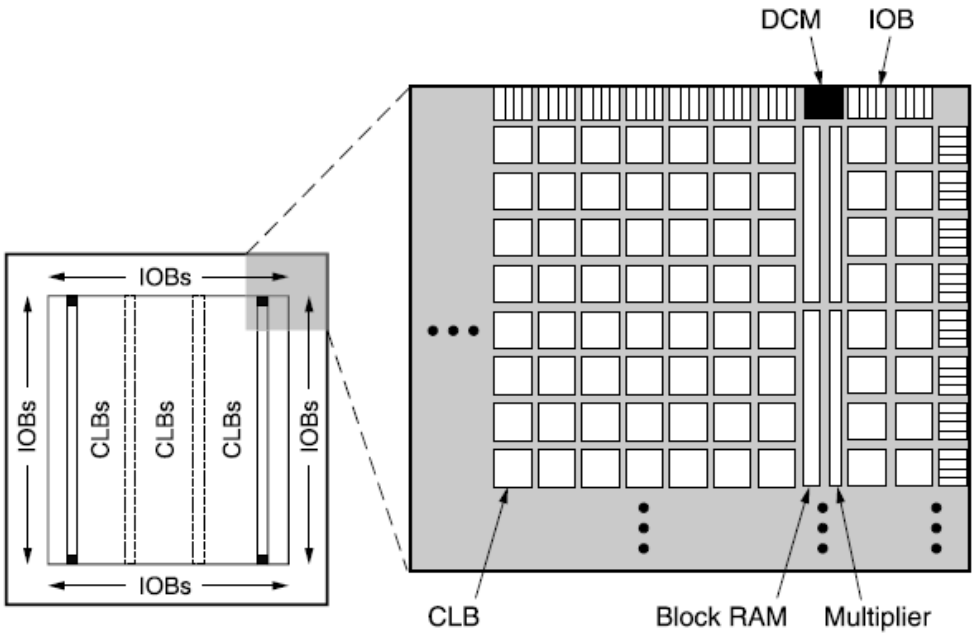
همچنین دارای ثبات‌های نرخ دو برابر داده (DDR2) نیز موجودند. ویژگی کنترل‌کننده امپدانس دیجیتالی (DCI[1])، فراهم‌کننده تطبیق پایانه‌های طرح مدار است تا اینکه به صورت درون تراشه عمل تطبیق امپدانس صورت پذیرد. این امر باعث ساده‌سازی طراحی مدار می‌شود.

• بلوک‌های رم : ذخیره‌سازی اطلاعات را در فرم ۱۸kbit به صورت بلوک‌های دو خروجی.

• بلوک‌های ضرب کننده (ultiplier Blocks M): دو عدد ۱۸ بیتی باینری را به عنوان ورودی دریافت و نتیجه آن را محاسبه می‌کند.

• بلوک‌های مدیریت کلاک دیجیتال(DCM[2]) برای تقسیم، توزیع، چند برابر کردن و شیفت فاز سیگنال کلاک

ساختار قرارگیری عناصر که در بالا به آن‌ها اشاره شد در شکل (۵-۸) نشان داده‌شده است.



شکل (۵-۸) معماری خانواده اسپارتان ۳

یک حلقه از IOB ها، آرایه‌ای از CLB ها را احاطه کرده است. تراشه‌های Xc3s50 تنها یک ستون از بلوک رم جاسازی‌شده را دارا هستند. تمام تراشه‌ها موجود در محدوده Xc3s200 الی Xc3s200

[1] Digitally Controlled Impedance
[2] Digilall Clock Manager

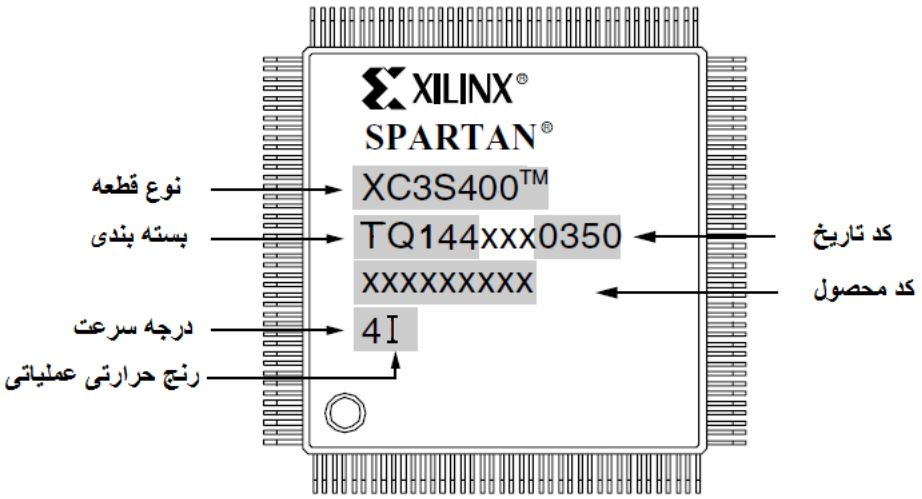
دربردارنده دو ستون از بلوک‌های رم هستند. تراشه‌های سری ۴۰۰۰ و ۵۰۰۰ دارای چهار ستون بلوک رم هستند. هر ستون از چندین بلوک رم ۱۸Kbit تشکیل‌شده و با یک ضرب کننده اختصاص‌یافته در ارتباط است. واحدهای DCMs در قسمت انتهایی بلوک‌های رم قرارگرفته‌اند. خانواده اسپارتان ۳ شامل یک شبکه قویی از سوئیچ‌ها و مسیرهایی است که تمام عناصر کاربردی پنج گانه را به یکدیگر متصل کرده و سیگنال‌ها را از طریق این مسیرها انتقال می‌دهد. هر یک از عناصر عملیاتی یک سوئیچ ماتریسی دارد که اجازه ارتباط برای مسیردهی چندگانه را به آن می‌دهد.

تراشه FPGA مورد استفاده در طرح انجام شده در این پایان‌نامه خانواده XC3S400 است که مشخصات آن در جدول ۵-۱ آورده شده است.

جدول ۵-۱ مشخصات FPGA مورد استفاده

تعداد ورودی/ خروجی تفاضلی	تعداد ورودی/ خروجی	DCM	ضرب کننده	رم بلوکی	رم توزیع شده	CLB	سلول منطقی	گیت سیستمی	نام قطعه
۴۶	۹۶	۴	۱۶	k۲۸۸ bit	۵۶ kbit	۸۹۶	۸۰۶۴	۴۰۰۰۰۰	XC3S400

در شکل (۵-۹) نام کامل قطعه و مشخصات بسته‌بندی نشان داده‌شده است. همانطور که در شکل مشخص است قطعه از نوع XC3S400 -4 TQ144 I می‌باشد که قسمت اول در این نام نوع قطعه را مشخص می‌کند. نماد4- مشخص‌کننده درجه سرعت و میزان تأخیر است. بخش TQ144 مشخص‌کننده تعداد پایه‌های قطعه است و در آخر حرف I ابتدای کلمه Industrial است که مشخص‌کننده رنج حرارتی بین -40°C تا 100°C می‌باشد.



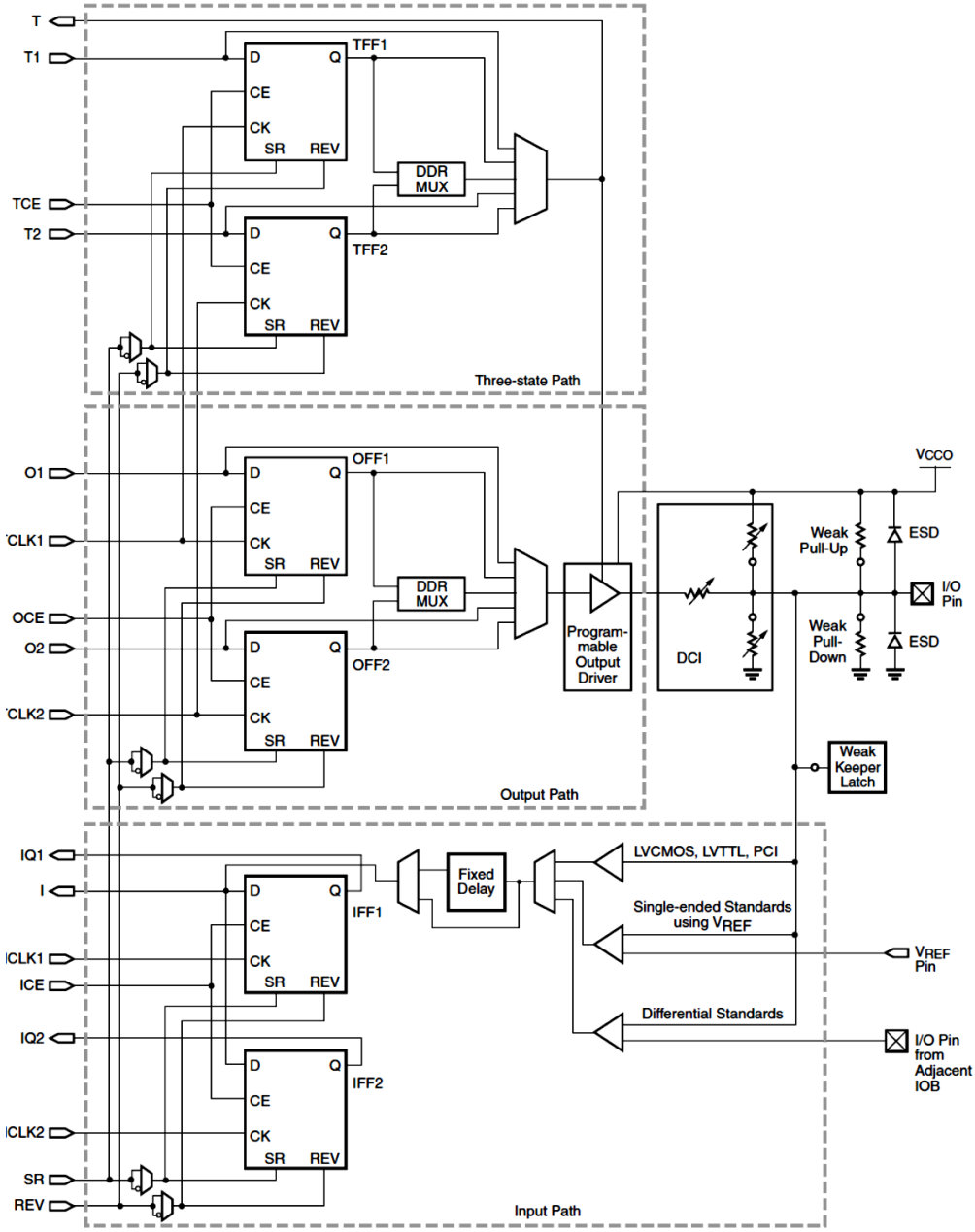
شکل (۵-۹) فرم بسته‌بندی

۵-۳-۲-۱ ساختار کلی پایه‌های ورودی خروجی

بلوک‌های ورودی خروجی یک واسط دو طرفه قابل برنامه‌ریزی را بین پین‌های O/I و منطق درونی FPGA فراهم می‌کنند. نموداری از ساختار درونی IOB در شکل (۵-۱۰) نشان داده‌شده است. سه مسیر اصلی سیگنال در IOB وجود دارد: مسیر خروجی، مسیر ورودی و مسیر سه حالته[1]. هر مسیر دارای جفت‌های عناصر ذخیره کننده خود است که می‌تواند به عنوان لچ یا ثبات عمل کند.

بلوک‌های ورودی خروجی در۷ بانک مجزا قرارگرفته‌اند به گونه‌ای که هر طرف قطعه ۲ بانک قرار می‌گیرد. هر بانک می‌تواند با یک ولتاژ مرجع خاص و یا یک ولتاژ خروجی خاص کار کند. هر کدام از سه حالت فوق را می‌توان به صورت نرم‌افزاری برای تمام پایه‌های I/O تعریف نمود.

[1] State



شکل (۵-۱۰) بلوک دیاگرام ساده‌شده IOB

۵-۲-۳-۲ مقاومت‌های Pull up/Pull down

مقاومت‌های انتخابی Pull up/Pull down برای تعیین سطح ولتاژ پایه‌های بدون استفاده مورد استفاده قرار می‌گیرند. مقاومت‌های Pull up، پایه مورد نظر را به VCCO متصل کرده و مقاومت Pull down

پایه مورد نظر را به زمین وصل می‌کند. این مقاومت‌ها را می‌توان برای تمام IO های بدون کارکرد با استفاده از گزینه مولد جریان بیت[1] انتخاب کرد.

سطح ولتاژ پایین بر روی پایه EN-HSWAP ، مقاومت‌های Pull-up را بر روی تمام O/I قبل از پیکره‌بندی و در حالت شروع به کار مدار فعال می‌کند. در نتیجه در ابتدای روشن کردن برد همه پایه‌های در حالت ولتاژ بالا هستند.

۵-۳-۲-۳ استانداردهای قابل انتخاب برای علامت‌های ورودی / خروجی

همانطور که در ابتدا اشاره شد، ورودی‌ها خورجی‌ها، ۱۷ استاندارد متفاوت تکی[2] را پشتیبانی می‌کنند. این استانداردها در جدول ۵-۲ نشان داده شده‌اند. به علاوه اکثر ورودی‌ها خورجی‌ها می‌توانند در جفت‌های مشخصی به کار روند که از ۶ استاندارد تفاضلی نشان داده‌شده در جدول ۵-۳پشتیبانی می‌کنند.

دو سطح ولتاژ (VCCO ، REF) تعیین کننده استاندارد سیگنال مطلوب می‌باشند. خطوط VCCO، فراهم‌کننده جریان برای راه‌اندازی خروجی است. سطح ولتاژ در این خطوط، تعیین‌کننده بازه تغییرات ولتاژ خروجی برای تمامی استانداردها به جز GTLP و GTL است. تمام استانداردهای تکی به جز LVCMOS، نیازمند یک ولتاژ مرجع (Vref) برای تعیین سطح سوییچ ورودی است. زمانی‌که که اطلاعات پیکره‌بندی به درون FPGA بارگذاری می‌شود، پین ورودی/خروجی مربوط به بانکی از FPGA را برای یک استاندارد مشخص پیکره‌بندی می‌کند. تعداد کمی از پین‌های مشخص رزرو شده ورودی خروجی بانک تعریف‌شده برای استانداردهای مذکور به ورودی‌های Vref تبدیل می‌شوند. زمانی که یکی از استانداردهای LVCMOS به‌کاربرده شود، این پین‌ها ورودی/خروجی باقی می‌مانند. این امر به علت این است که ولتاژ VCCO آستانه سوئیچ ورودی را بایاس کرده است. در این صورت به ولتاژ

[1] Bit stream generator
[2] Single ended

مرجع نیازی نیست. سطوح VCCO،VREF برای به دست آوردن استاندارد تکی مطلوب با توجه به جدول ۵-۲ انتخاب می‌شوند.

استانداردهای تفاضلی، یک جفت از سیگنال‌ها را که قطبیت یکی بر عکس دیگری می‌باشد به کار می‌گیرند. توانایی حذف نویز در این استاندارد، اجازه می‌دهد انتقال اطلاعات با سرعت بسیار بالا صورت گیرد. یک umberN-L منحصربه‌فرد (بخشی از نام پین قطعه) معرف جفت خطوط مرتبط با هر بانک است. هر جفت، حروف P و N به ترتیب معرف خط اصلی و خط وارونه می‌باشند. به عنوان مثال پین IO-L43P-7 وIO-L43N-7 معرف خطوط اصلی و وارونه هستند که دربرگیرنده جفت خط L43 در بانک ۷ می‌باشند.

خطوط VCCO، فراهم‌کننده جریان برای خروجی هستند. خطوط VccAux تأمین‌کننده توان برای ورودی‌های تفاضلی است که باعث مستقل شدن آن‌ها از ولتاژ VCCO برای یک بانک O/I می‌شود. انتخاب سطح VCCO برای رسیدن به استاندارد تفاضلی مطلوب بر طبق جدول ۵-۳ صورت می‌پذیرد.

جدول ۵-۲ استانداردهای ورودی خروجی تک پایه

Signal Standard	V_{CCO}		V_{REF} for Inputs(1)	Board Termination Voltage (V_{TT})
	For Outputs	For Inputs		
GTL	Note 2	Note 2	0.8	1.2
GTLP	Note 2	Note 2	1	1.5
HSTL_I	1.5	-	0.75	0.75
HSTL_III	1.5	-	0.9	1.5
HSTL_I_18	1.8	-	0.9	0.9
HSTL_II_18	1.8	-	0.9	0.9
HSTL_III_18	1.8	-	1.1	1.8
LVCMOS12	1.2	1.2	-	-
LVCMOS15	1.5	1.5	-	-
LVCMOS18	1.8	1.8	-	-
LVCMOS25	2.5	2.5	-	-
LVCMOS33	3.3	3.3	-	-
LVTTL	3.3	3.3	-	-
PCI33_3	3.0	3.0	-	-
SSTL18_I	1.8	-	0.9	0.9
SSTL2_I	2.5	-	1.25	1.25
SSTL2_II	2.5	-	1.25	1.25

جدول ۵-۳ استانداردهای ورودی خروجی تفاضلی

Signal Standard	V_{CCO} (Volts)		V_{REF} for Inputs (Volts)	V_{OD}(1) (mV)	
	For Outputs	For Inputs		Min.	Max.
LDT_25	2.5	-	-	430	670
LVDS_25	2.5	-	-	250	400
BLVDS_25	2.5	-	-	250	450
LVDSEXT_25	2.5	-	-	330	700
ULVDS_25	2.5	-	-	430	670
RSDS_25	2.5	-	-	100	400

۵-۳-۲-۴ پیکره‌بندی[1]

تراشه‌های اسپارتان۳ توسط بارگذاری داده‌های مختص خود در حافظه داخلی پیکره‌بندی می‌شوند. تنظیمات با استفاده از زیرمجموعه‌ای از پین‌های تراشه صورت می‌پذیرد که برخی از آن‌ها " اختصاصی " فقط با یک عملکرد می‌باشند و درحالی‌که مابقی، با "هدفی دوگانه" هستند و می‌توانند مورد استفاده دوگانه به عنوان کارکرد عمومی O/I[2] و همچنین عامل پیکره‌بندی قرار بگیرند. بسته به نوع طراحی سیستم، از چندین حالت پیکره‌بندی پشتیبانی می‌شود. پین‌های M0، M1، M2 پین‌های اختصاصی پیکره‌بندی هستند. تنظیمات حالت پین‌ها جدول ۵-۴ نشان داده‌شده است.

جدول ۵-۴ تنظیمات مدهای پیکره‌بندی

Configuration Mode(1)	M0	M1	M2	Synchronizing Clock	Data Width	Serial DOUT(2)
Master Serial	0	0	0	CCLK Output	1	Yes
Slave Serial	1	1	1	CCLK Input	1	Yes
Master Parallel	1	1	0	CCLK Output	8	No
Slave Parallel	0	1	1	CCLK Input	8	No
JTAG	1	0	1	TCK Input	1	No

پین ورودی HSWAP_EN چگونگی حالت پین‌های O/I را که در هنگام پیکره‌بندی استفاده نمی‌شوند را تعریف می‌کند. به طور پیش‌فرض، HSWAP_EN با استفاده از مقاومت‌های pull-up داخلی در حالت High قرار دارد. در حالت پیش‌فرض مقاومت‌های pull-up مربوط به O/I را در مدت پیکره‌بندی غیرفعال هستند. زمانی که پایه HSWAP_EN را به زمین متصل کنیم مقاومت‌های pull-up ورودی خروجی فعال شده و خروجی‌ها در وضعیت High قرار می‌گیرند. (در این برد همانطور که قبلاً هم ذکر شد این پایه زمین شده است).

گزینه‌ای نیز در دسترس است که می‌تواند پین‌ها را در حالت عملکرد پیکره‌بندی خود حتی پس از اینکه پیکره‌بندی دستگاه کامل شد، حفظ کند. اگر این گزینه انتخاب نشده باقی بماند، پین‌های پیکره‌بندی به استثنای CCLK، PROG_B و DONE می‌توانند به عنوان O/I با عملکردی معمول مورد استفاده قرار گیرند.

[1] Configuration
[2] General purpose input/output

۵-۳-۲-۵ حالت‌های پیکره‌بندی

اسپارتان ۳ از پنج حالت پیکره‌بندی زیر پشتیبانی می‌کند:

- حالت Slave سریال
- حالت Master سریال
- حالت Slave موازی (SelectMAP)
- حالت Master موازی (SelectMAP)
- حالت اسکن مرزی (JTAG) (IEEE1532 ، IEEE1.1194.)

از آنجاییکه در برد طراحی و ساخته شده این پایان‌نامه از حالت اسکن مرزی استفاده شده فقط به توضیح این حالت پیکره‌بندی می‌پردازیم.

۵-۳-۱-۵-۱ حالت اسکن مرزی[1] (JTAG)

در حالت اسکن مرزی پین‌های اختصاص داده‌شده برای پیکره‌بندی FPGA مورد استفاده قرار می‌گیرند. پیکره‌بندی به طور کامل از طریق IEEE1.1194 (TAP[2]) انجام می‌شوند. پیکره‌بندی FPGA با استفاده از حالت اسکن مرزی سازگار با استاندارد IEEE1.1194 -۱۹۹۳ و استاندارد IEEE1532 برای تراشه‌های قابل برنامه‌ریزی مجدد درون سیستمی انجام می‌شود. پیکره‌بندی از طریق پورت اسکن مرزی صرفنظر از حالت پیکره‌بندی انتخاب شده، همیشه ممکن است. با انتخاب حالت اسکن مرزی حالت‌های دیگر غیرفعال می‌شوند. حالت اسکن مرزی انتخاب شده حالت‌های دیگر را غیرفعال می‌کند. قابل اعتماد ترین حالت در زمان برنامه‌نویسی از طریق JTAG می‌باشد.

فرآیند پیکربندی با تعریف پین PROG_B آغاز می‌شود. پایان عملیات پاک‌سازی حافظه توسط یک شدن INIT_B مشخص می‌شود. در این مرحله، اطلاعات پیکره‌بندی بر روی FPGA نوشته می‌شود. تراشه FPGA سیگنال‌های تنظیم / بازیابی عمومی (GSR[3]) را در طول پیکره‌بندی فعال نگه می‌دارد.

[1] Boundary-Scan
[2] Test Access Point
[3] Global Set/Reset

و همه فلیپ فلاپ‌ها نیز در حالت (Reset) هستند. اتمام تمامی مراحل بالا توسط یک شدن پین DONE مشخص می‌شود. این پین به یک LED متصل است که روشن شدن آن نشاندهنده پایان موفق عملیات پیکره‌بندی است. زمان نسبی رویدادهای پیکره‌بندی می‌تواند از طریق گزینه‌های موجود در BitGen و نرم‌افزار IMPACT ، تغییر کند.

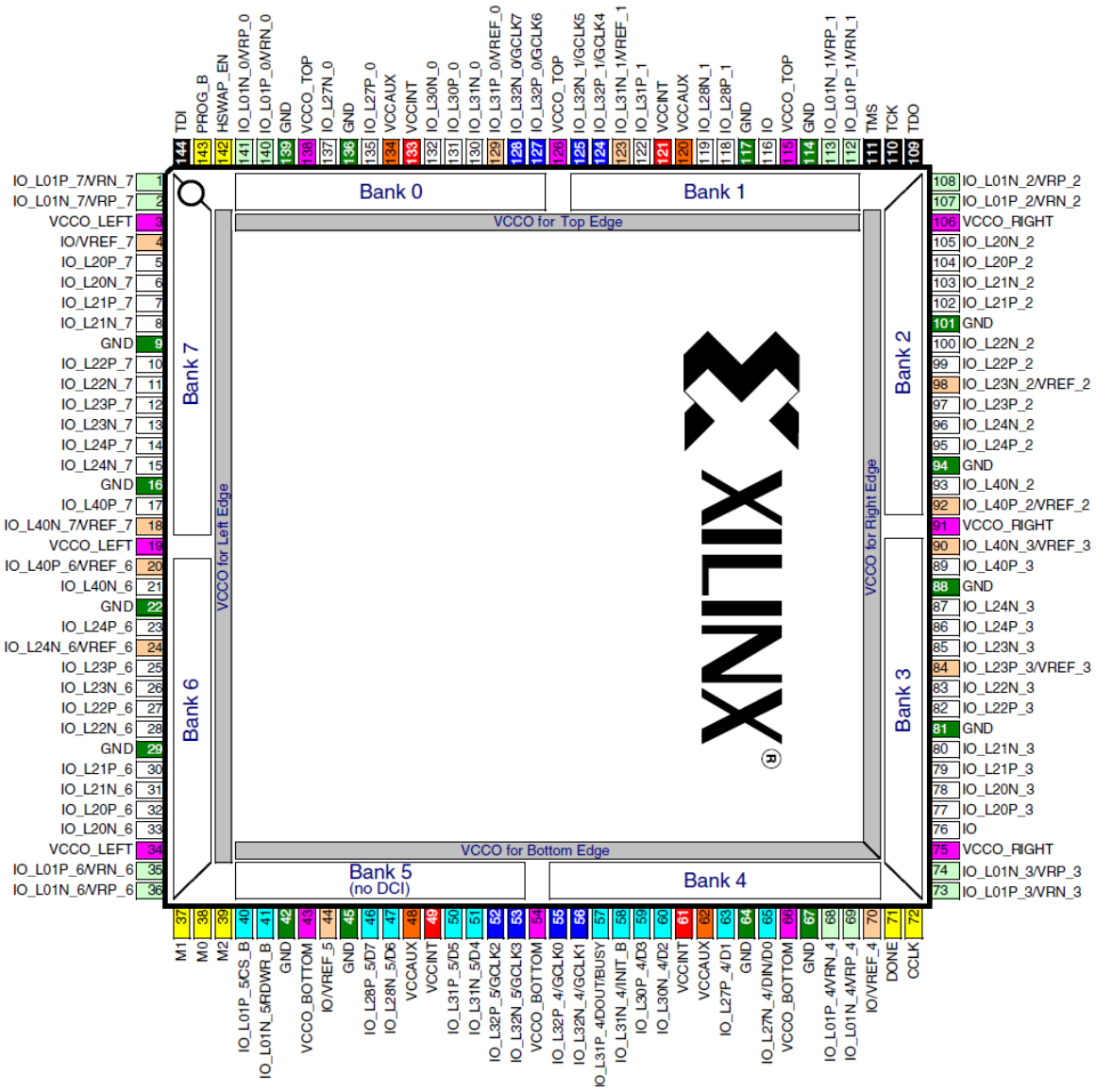
برای پیکره‌بندی برد پک پورت JTAG بر روی آن تعبیه شده است، که می‌توان با اتصال پروگرامر شرکت Xilinx به برد، عملیات پیکره‌بندی را انجام داد و همانطور که در بالا ذکر شد یک LED برای نشان دادن پایان عملیات نیز در نظر گرفته شده است.

۵-۳-۲-۶ ولتاژهای مورد نیاز برای تغذیه تراشه FPGA

برای تغذیه FPGA در طرح مورد نظر به سه سطح ولتاژ مختلف نیاز است، شامل سطح 1.2v برای Vccint که تأمین‌کننده اصلی ولتاژ مدارات منطقی داخلی می‌باشد. سطح 2.5v برای Vccaux یک ولتاژ کمکی برای عملکرد بهتر و بالا بردن کارایی FPGA در کاربردهایی مثل سوییچینگ خروجی، تغذیه سیستم مدیریت کلاک و تغذیه پین‌های JTAG مورد استفاده قرار می‌گیرد و در آخر سطح 3.3v برای Vcco که برای هر بانک ورودی خروجی به صورت جدا می‌باشد و برای تغذیه راه اندازه‌ای ورودی خروجی مورد استفاده قرار می‌گیرد. برای تأمین این سه سطح ولتاژ از سه تراشه رگولاتور استفاده شده که در بخش تغذیه مدار بطور کامل توضیح داده می‌شود.

۵-۳-۲-۷ ساختار کلی پایه‌ها

در شکل (۵-۱۱) یک شمای کلی فوت‌پرینت تراشه FPGA مورد استفاده نشان داده‌شده است. نام و کاربرد تمام ۱۴۴ پایه آن نیز ذکر شده است.



شکل (۵-۱۱) شکل ظاهری FPGA مورد استفاده

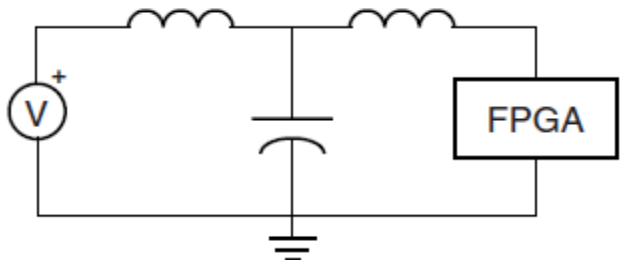
۵-۴ طراحی سیستم توزیع توان و خازن‌های دیکوپلینگ

موضوع طراحی سیستم توزیع توان در طرح‌های مبتنی بر FPGA بسیار چالش برانگیز است. اکثر تراشه‌های بزرگ مثل میکروپروسسورها دارای فرکانس و شرایط کاری مشخص می‌باشند. در نتیجه مشخصات خازن‌های بایپس آن‌ها معلوم است. در حالیکه FPGA دارای چنین شرایطی نیستند و برای کاربردهای مختلفی مورد استفاده قرار می‌گیرند که هر کاربرد بسته به شرایط خود فرکانس و یا دامنه‌ای

از فرکانس‌های خاص دارد. در نتیجه نیاز جریان‌های گذرا و تغییرات آن‌ها به راحتی قابل پیش‌بینی نمی‌باشد؛ در چنین شرایطی برای یک طراحی مناسب باید بدترین شرایط ممکن را در نظر گرفت.

همانطور که می‌دانیم جریان‌های گذرای مورد نیاز در مدارات دیجیتال عامل جهش[1] سیگنال زمین و تغذیه می‌باشد و محدود کننده سرعت مدار است. در شکل (۵-۱۲) یک مدار ساده شده سیستم توزیع توان نشان داده‌شده است.

هدف اصلی طراحی سیستم توزیع توان، تأمین مناسب توان مورد نیاز برای کارکرد مطلوب FPGA و دیگر تراشه‌های موجود در مدار است. در اکثر تراشه‌ها از جمله FPGA نیاز است تا ولتاژ تغذیه بیشتر از ٪۵ و کمتر از ٪۵ ولتاژ نامی تغذیه نشود. یعنی نوسان ولتاژ تغذیه نباید بیشتر از ٪۱۰ باشد. تغییرات جریان مورد نیاز و در نتیجه‌ی آن ولتاژ، باعث به وجود آمدن فرکانس‌های مختلفی در تمام بازه‌ها در تغذیه می‌شود. خازن‌ها در نقش ذخیره سازهای انرژی کوچک عمل می‌کنند و می‌توانند در مواقعی که جریان شارژ مورد نیاز است به سرعت عمل کنند و از تغییر ولتاژ جلوگیری کنند.

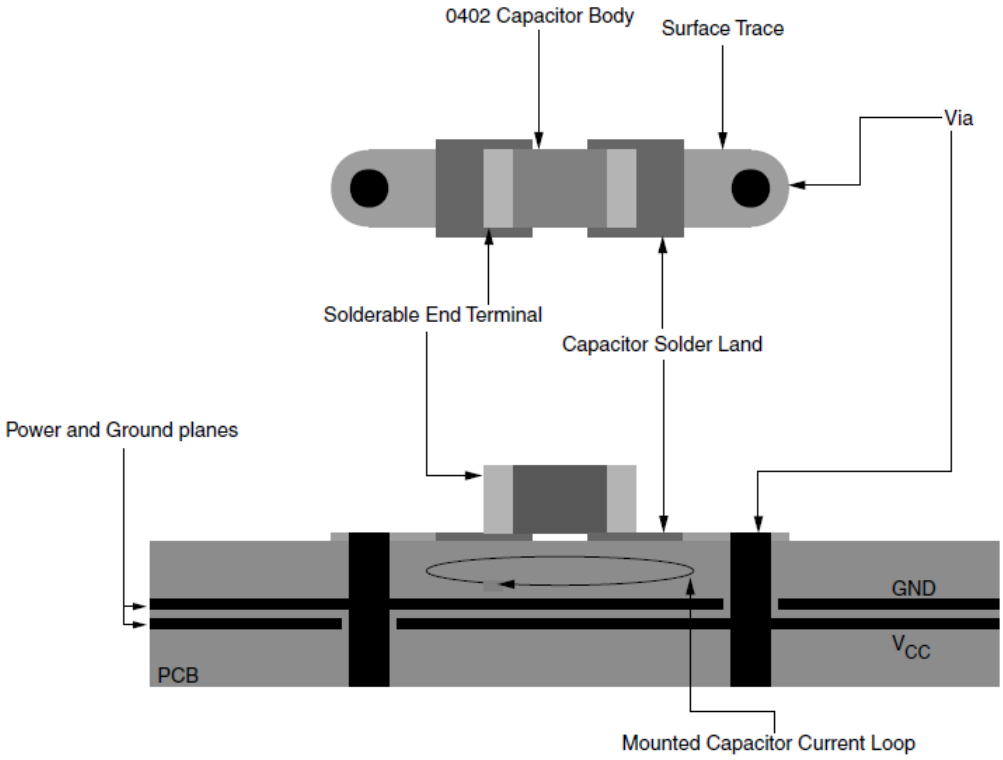


شکل (۵-۱۲) مدار توزیع توان ساده شده

وجود اندوکتانس[2] در مدار باعث جلوگیری از پاسخ سریع به نیاز تغییر جریان می‌شود. در نتیجه برای داشتن ولتاژ تغذیه بدون نوسان باید اندوکتانس مدار را به حداقل رساند و خازن‌های مورد نیاز را تأمین کرد. اندوکتانس مدار ناشی از عوامل مختلفی است. یکی از عوامل اندوکتانس نویزی موجود مربوط به اندازه بسته‌بندی خازن‌ها است. در نتیجه باید برای انتخاب خازن کوچک‌ترین بسته‌بندی موجود با حداکثر ظرفیت را انتخاب نمود تا کمترین اندوکتانس نویزی ایجاد شود. در نتیجه در این طرح تمام

[1] Debounce
[2] inductance

خازن‌ها به صورت SMD با کوچک‌ترین بسته‌بندی موجود در دسترس استفاده شده است. همانطور که در شکل (۵-۱۳) مشاهده می‌شود طراحی PCB می‌تواند باعث به وجود آمدن یک مسیر جریان و در نتیجه اندوکتانس پارازیتی ناشی از مسیر جریان در PCB شود که باید در حین طراحی با رعایت نکات مهم آن را به حداقل رساند.



شکل (۵-۱۳) مسیر جریان به وجود آمده در PCB

هر خازن یک باند فرکانسی باریک دارد که بیشترین تأثیر را به عنوان خازن بایپس در آن بازه دارا می‌باشد. بسته به نوع خازن باند موثر آن متفاوت است. خازن‌های تانتالیوم باند موثر بهتری دارند. باند موثر وابسته به فرکانس رزونانس خازن است. همانطور که می‌دانیم خازن واقعی غیر ایده‌آل مقداری اندوکتانس و مقاومت نیز دارند. که تشکیل یک مدار RLC را می‌دهند، فرکانس رزونانس آن از رابطه زیر به دست می‌آید[20].

$$F = \frac{1}{2\pi\sqrt{LC}} \qquad (۵-۱)$$

موقعیت قرارگیری خازن‌ها در برد باید در نزدیک‌ترین نقطه به FPGA باشد. چون در این صورت جریان کمترین مسیر ممکن را طی می‌کند و از به وجود آمدن جریان گردشی جلوگیری می‌شود. و همچنین نویز به وجود آمده در آن منطقه بایبس شده. برای طراحی خازن‌های دیکوپلینگ باید به تعداد پین‌های تغذیه خازن دیکوپلینگ در نظر گرفت. که شامل پین‌های تغذیه Vcco,Vccint,Vccaux می‌شوند. همچنین مقدار خازن‌ها باید به گونه‌ای انتخاب شوند تا تمام بازه‌های فرکانسی را پوشش دهد. و از هر دهه رنج مقادیر خازن باید در مقادیر خازن‌ها موجود باشد. مقادیر خازن‌ها در جدول ۵-۵ آورده شده، همانطور که در جدول هم مشخص شده مقدار دقیق خازن مهم نیست. ولی محدوده مقداری آن‌ها دارای اهمیت است.

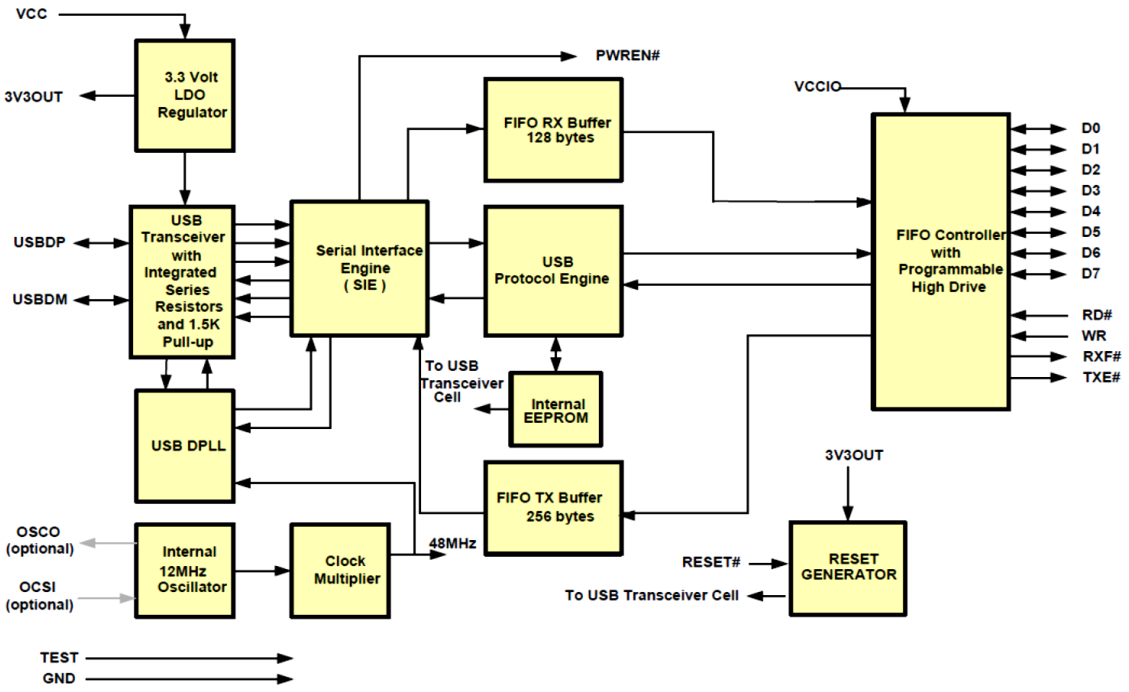
جدول ۵-۵ در صد مقادیر خازن‌های مورد استفاده برای دیکوپلینگ مناسب

نوع خازن	تعداد بر اساس درصد	مقدار خازن
Tantalum	4%	470µF-1000µF
CDR32 1206	14%	1µF-4.7µF
CDR32 1206	27%	0.1µF-0.47µF
CDR32 1206	55%	0.01µF-0.047µF

۵-۵ رابط USB

برای ارتباط مدار با کامپیوتر راه‌های زیادی وجود دارد و می‌توان از پورت‌های موازی یا سریال (RS232) نیز استفاده کرد. ولی ارتباط از طریق این پورت‌ها از سرعت پایینی برخوردار است همچنین این پورت‌ها در کامپیوترهای جدید در حال از بین رفتن هستند. پورت USB علاوه بر سرعت بالا از قابلیت شناسایی خودکار نیز برخوردار است. برای سهولت ارتباط از طریق پورت USB و پایین آمدن پیچیدگی سخت‌افزاری و نرم‌افزاری باید از یکی از تراشه‌های واسط موجود در بازار استفاده نمود. یکی از شرکت‌های معروف در این زمینه شرکت FTDI CHIP است. که تراشه‌های مختلفی در زمینه ارتباط از طریق USB دارد. تراشه مورد استفاده در این طرح FT245R است[21] .

این تراشه توانایی ارتباط موازی را دارا می‌باشد. یعنی اطلاعات به صورت ۸ بیتی از FPGA دریافت می‌شود. و با تبدیلات انجام شده در تراشه برای ارسال به USB آماده می‌شوند. در شکل (۵-۱۴) ساختار داخلی تراشه نشان داده‌شده است. همانطور که در بلوک دیاگرام قابل مشاهده است تراشه در این مدل دارای یک نوسان ساز داخلی است که نیاز به مدارات جانبی را کاهش می‌دهد. همچنین قابلیت کار کردن با ولتاژ 3.3v را دارد که دیگر نیاز به تبدیل سطح ولتاژ نیست و می‌تواند به صورت مستقیم به FPGA متصل شود. همچنین دارای دو بافر برای ارسال و در یافت داده است که می‌تواند داده‌های ارسالی از سمت FPGA را در بافر 256byts خود قرار داده و به USB ارسال کند.



شکل (۵-۱۴) ساختار داخلی تراشه FT245

۵-۵-۱ عملکرد پایه‌های FT245

در این قسمت به شرح مختصری از پین‌های FT245 می‌پردازیم.

D0-D7: به عنوان پورت ارسال و دریافت استفاده می‌شود. به این صورت که در هنگام دریافت این پین‌ها به عنوان ورودی و در زمان ارسال به عنوان خروجی محسوب می‌شوند.

WR: برای ارسال داده به USB باید بعد از قرار دادن اطلاعات بر روی پورت داده این پایه را برای زمان مشخص یک و سپس صفر کنیم.

RD: برای خواندن اطلاعات این پایه را برای مشخص صفر و سپس یک می‌کنیم.

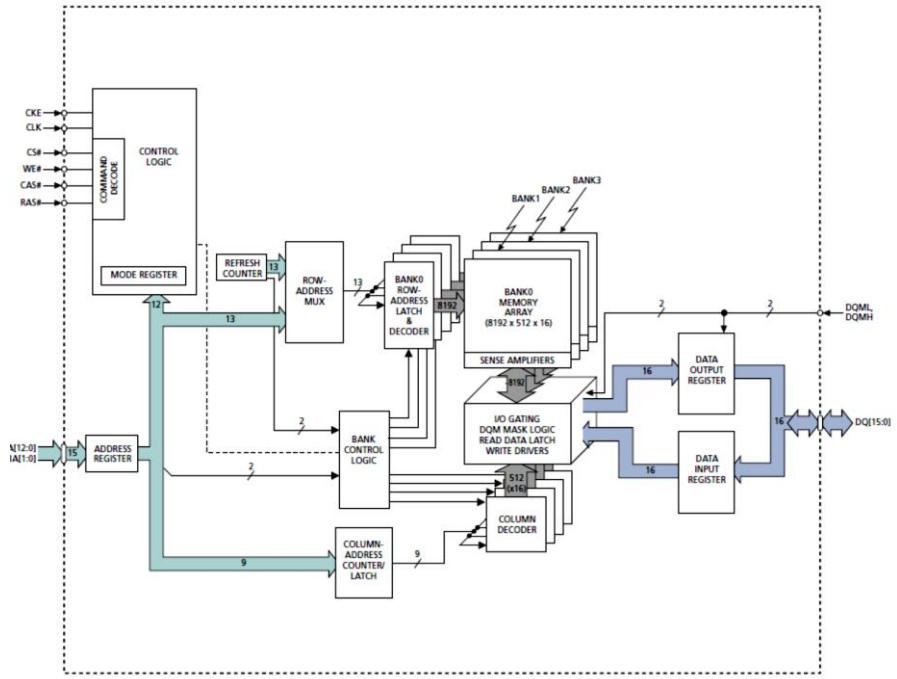
RXF: نشان‌دهنده وضعیت خواندن است در صورت یک بودن به معنی آمادگی برای خواندن است

TXE: نشان‌دهنده وضعیت نوشتن است. در صورت صفر بودن به معنی آمادگی تراشه برای دریافت اطلاعات است

۶-۵ حافظه RAM

رم استفاده شده در این پروژه MT48LC64M4A2 از نوع SDRAM با سرعت بالا و عملکرد به صورت همزمان[1] با ظرفیت 256Mbit است. همانطور که در شکل (۵-۱۵) مشخص شده رم مورد استفاده دارای ۴ بانک حافظه با دسترسی همزمان است. تأخیر کم این رم اجازه خواندن و نوشتن سریع را می‌دهد. همچنین سطح ولتاژ که این رم در آن کار می‌کند 3.3v است که انتخاب مناسبی برای ارتباط با FPGA می‌باشد.

[1] Synchronous



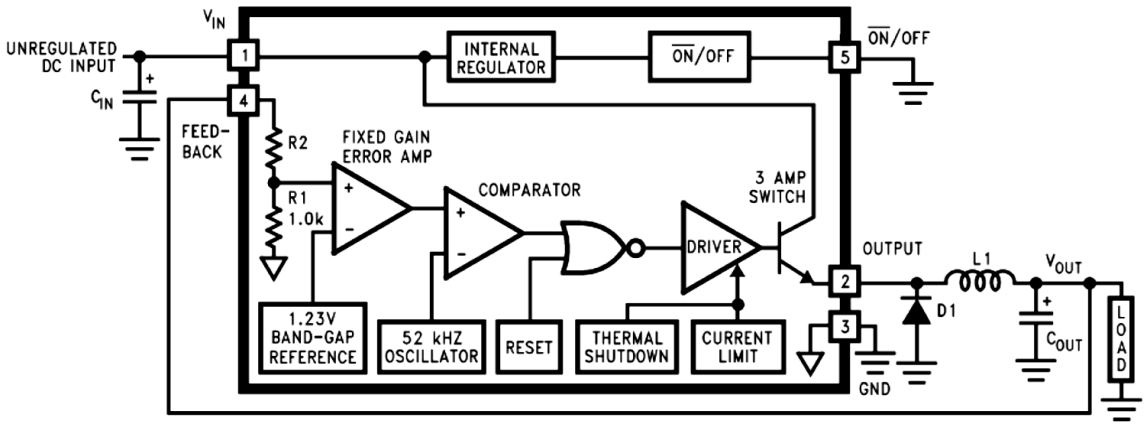
شکل (۵-۱۵) بلوک دیاگرام داخلی SDRAM

۵-۷ تغذیه

در این مدار چهار سطح ولتاژ وجود دارد که عبارتند از 3.3V,1.2V,2.5V,5V . که سطح ولتاژ 3.3V برای تغذیه FPGA و تراشه‌های دیگر مورد استفاده قرار می‌گیرد. همانطور که در بخش FPGA توضیح داده شد سطح 1.2V و2.5V نیز برای تغذیه FPGA مورد استفاده قرار می‌گیرد. سطح 5V برای ارتباط USB و همچنین تغذیه در دسترس برای پورت‌های گسترش برای مدارهای آینده مورد استفاده قرار می‌گیرد.

۴-۶-۱ سطح ولتاژ 5V

تراشه رگولاتور LM2576 یک رگولاتور 5V است. که بلوک دیاگرام آن به همراه مدارهای مورد نیاز برای راه‌اندازی در شکل (۵-۱۶) نشان داده‌شده است. تلورانس ولتاژ خروجی این رگولاتور 4%± است که برای این طرح و با توجه به رگولاتورهای موجود در مرحله بعد میزان قابل قبولی است.



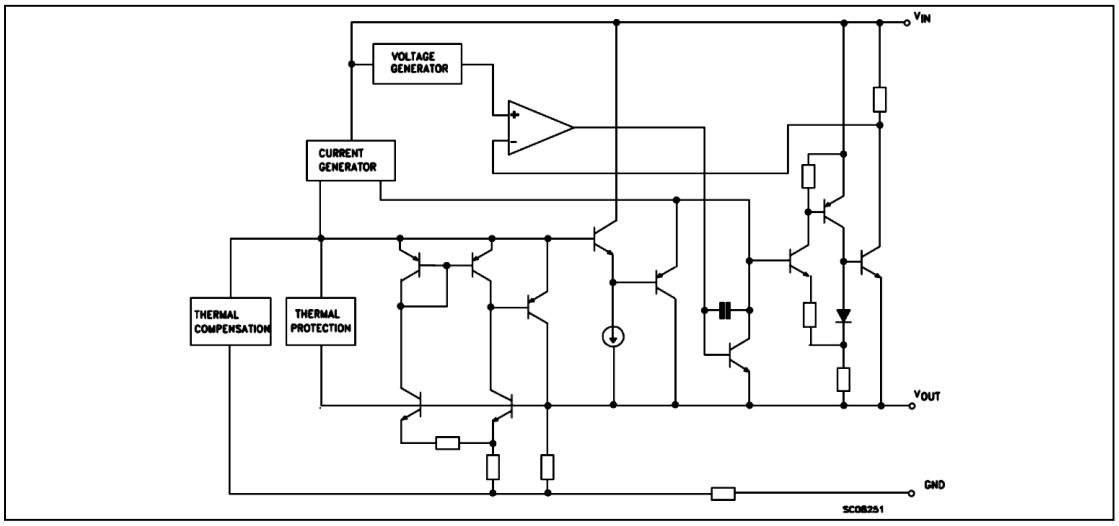
شکل (۵-۱۶) بلوک دیاگرام LM2576

جریان خروجی این رگولاتور 3A است. همچنین ولتاژ ورودی آن می‌تواند از ۷ تا ۴۰ ولت باشد که با توجه به مشخص نبودن ولتاژ ورودی در این طراحی بازه بسیار مناسبی است. همانطور که در بلوک دیاگرام می‌توان دید یکی از مزایای این رگولاتور دارا بودن محدودکننده جریان سریع است. در نتیجه می‌تواند نقش محافظ را برای مدار ایفا کند و در مواقع اضافه جریان مانند یک فیوز سریع عمل کند و از آسیب رسیدن به مدار جلوگیری کند.

۴-۶-۲ سطح ولتاژ 3.3V

برای تأمین ولتاژ 3.3V خروجی 5V را به رگولاتور LD1117S33CTR که ولتاژ خروجی آن 3.3V است می‌دهیم.

بلوک دیاگرام این رگولاتور در شکل (۵-۱۷) نشان داده‌شده است.



شکل (۵-۱۷) بلوک دیاگرام رگولاتور LD1117

رگولاتور LD1117S33CTR توانای فراهم کردن جریان تا سقف 800mA را دارا می‌باشد.

۴-۶-۳ سطح ولتاژ 2.5V

برای تأمین ولتاژ 2.5V نیز مانند حالت 3.3V خروجی 5V را به رگولاتور LD1117S25CTR که ولتاژ خروجی آن 2.5V است می‌دهیم. مشخصات و مدار داخلی این رگولاتور نیز دقیقاً مثل نوع نشان داده‌شده در شکل (۵-۱۷) است و فقط دارای خروجی ۲/۵ ولتی است.

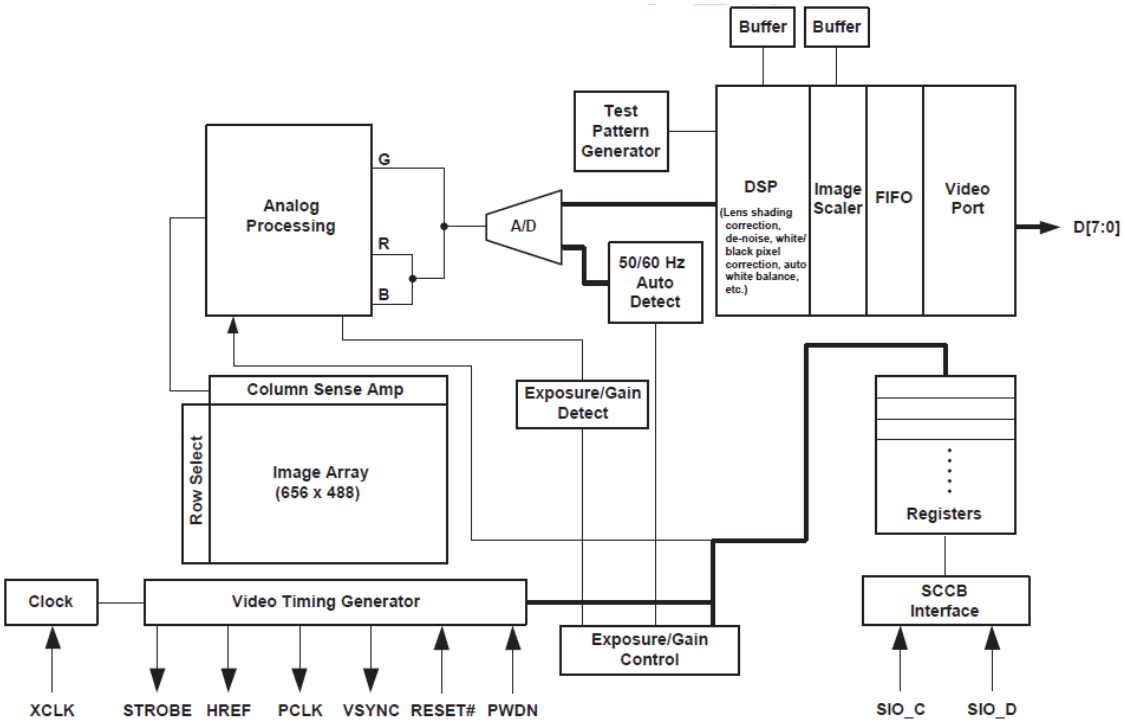
۴-۶-۴ سطح ولتاژ 1.2V

برای تأمین ولتاژ 1.2V از خروجی 5V به عنوان ورودی رگولاتور FAN1112S-1.2 که خروجی آن 1.2V است استفاده می‌کنیم. نحوه عملکرد این رگولاتور نیز مانند موارد گفته شده است و توانایی تأمین جریان تا 1A را دارا می‌باشد.

۵-۸ ماژول دوربین OV7670

ماژول دوربین OV7670 یک دوربین CMOS است و دارای بخش‌های مختلف برای پردازش اولیه روی تصویر (میزان رنگ ، کنترانس و....) می‌باشد. همچنین توانایی تنظیم خودکار بهره AGC ، تنظیم تراز سفیدی AWB را دارا می‌باشد، مدارات بایاسینگ و رگولاتور۱/۸ ولت بر روی خود برد قرارگرفته است.

بلوک دیاگرام این ماژول در شکل (۵-۱۸) نشان داده‌شده. برای برنامه‌ریزی ابتدایی این ماژول باید فرمان‌های مورد نیاز را از طریق پین‌های SIO-C و SIO-D و توسط پروتکول I2C ارسال می‌کنیم و بعد از انجام تعاریف اولیه داده‌های تصویر را از طریق پورت داده دریافت می‌کنیم. همچنین چند پایه کنترلی برای تعیین موقعیت پیکسل وجود دارد. که در برنامه مورد نظر برای استفاده از این ماژول از آن‌ها استفاده شده است.



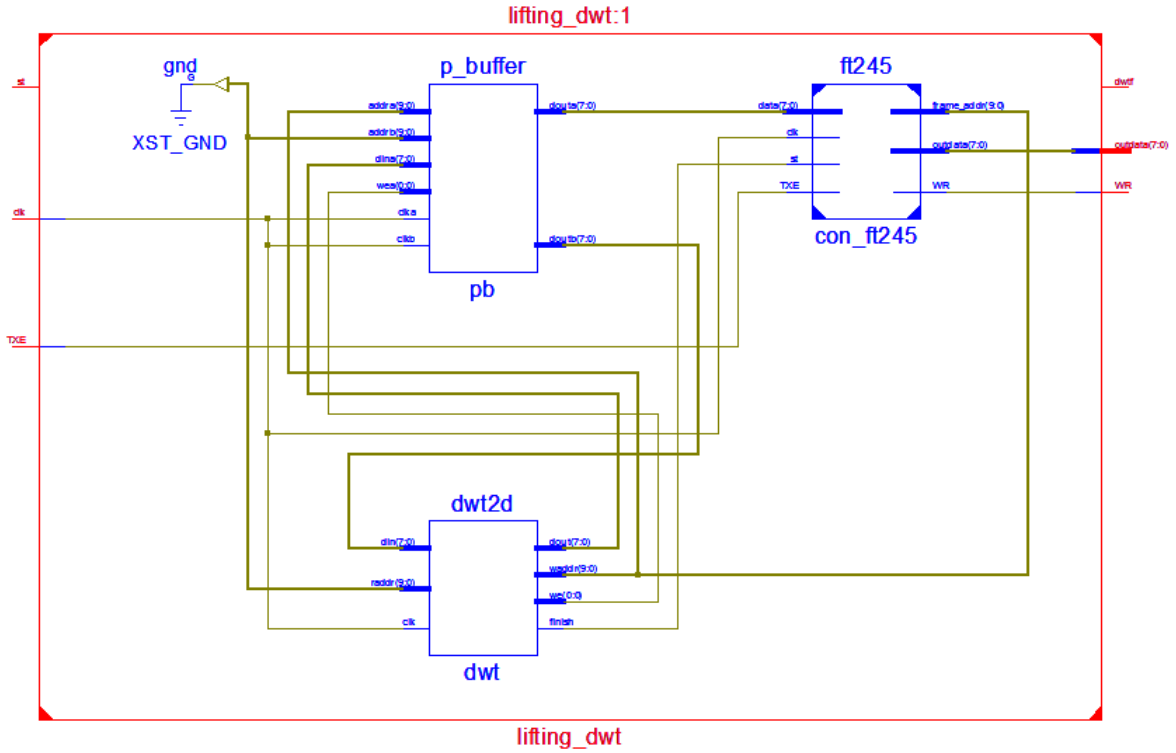
شکل (۵-۱۸) بلوک دیاگرام دوربین OV7670

فصل ششم

بررسی نتایج شبیه سازی و پیاده سازی سخت افزاری

۶-۱ شبیه‌سازی و پیاده‌سازی سخت‌افزاری

برای پیاده سازی سخت افزاری ابتدا کد VHDL تبدیل موجک به روش طرح بالابری نوشته می‌شود. پس از نوشتن کد در محیط نرم‌افزار ISE برای ارزیابی کد نوشته شده از نرم‌افزار Isim[۱] استفاده می‌کنیم. پس از شبیه‌سازی با استفاده از نرم‌افزار IMPACT کد نوشته شده را به FPGA منتقل می‌کنیم. کد نوشته شده شامل سه ماژول اصلی است که میتوان این ماژول‌ها را به صورت شماتیک با استفاده از شماتیک RTL[۲] نرم افزار ISE مشاهده کرد. در شکل (۶-۱) بلوک‌های استفاده شده (که از نرم‌افزار ISE بدست آمده) در کد VHDL نشان داده شده است.



شکل (۶-۱) شماتیک RTL تبدیل موجک دو بعدی

همان‌طور که در شکل (۶-۱) دیده می‌شود. طرح دارای ۳ بلوک اصلی است. بلوک P_buffer که یک RAM دو پورت بوده و قابلیت خواندن و نوشتن از دو پورت جداگانه در یک حافظه را به صورت همزمان

---

[۱] Isim مخفف ISE Simulator، شبیه‌سازی ISE است. که به صورت مجتمع با نرم افزار ISE Design Suite نسب می‌شود.

[۲] Regester Transfer Level

دارا می‌باشد. تصویر اولیه با ابعاد فریم ۱۰۰×۱۰۰ و پیکسل‌های ۸ بیتی است. برای برقراری امکان استفاده از حافظه درونی FPGA ابعاد تصویر را کاهش داده‌ایم. تصویر مورد نظر در محیط نرم‌افزار ISE در رَم بازگذاری می‌شود و به FPGA منتقل می‌شود. استفاده از حافظه درونی باعث بالا رفتن سرعت می‌شود. بلوک تبدیل موجک (dwt2d) پیکسل‌های تصویر را از طریق پورت B خوانده و همانطور که در فصل چهار توضیح داده شد، معادلات تجزیه فیلترها مربوط به بخش ۴-۳ را بر روی سطرهای تصویر اعمال می‌کند. همانطور که مشاهده می‌شود معادلات تجزیه فقط شامل جمع و شیفت می‌باشند:

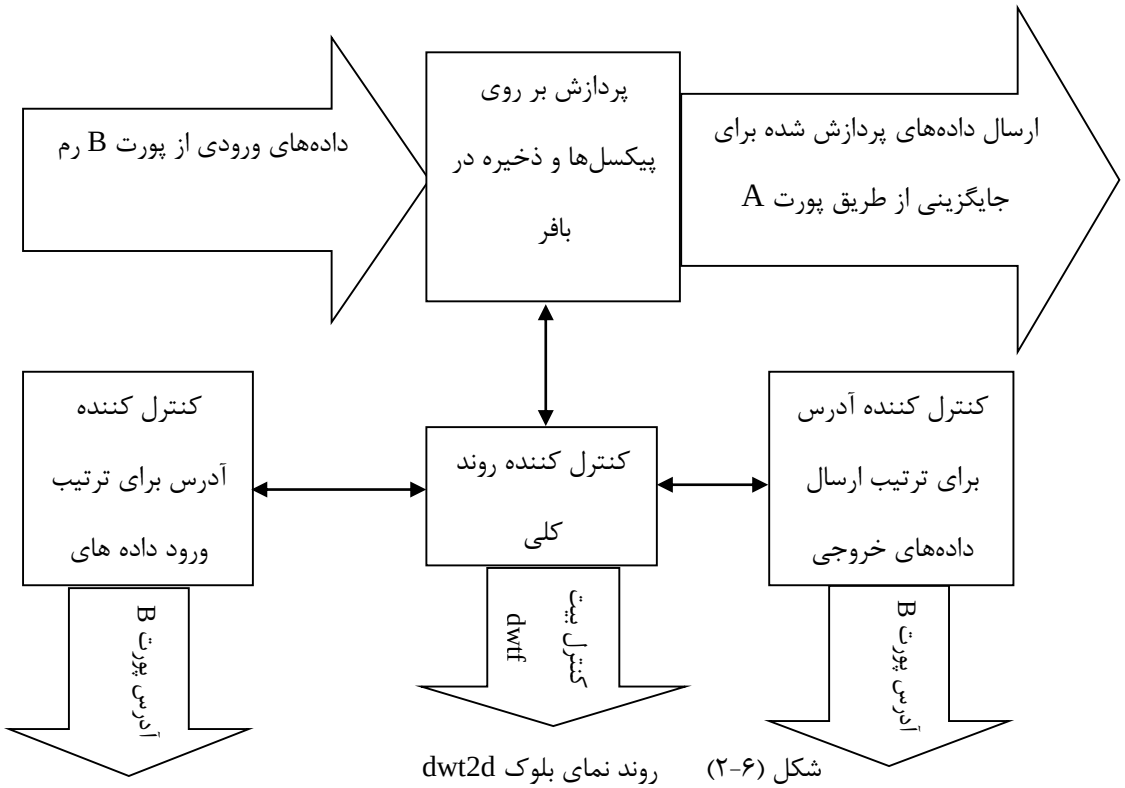
(۶-۱)

ضرایب بالا گذر $g(k) = x(2k+1) - \frac{1}{2}(x(2k) + x(2k+2))$

ضرایب پایین گذر $f(k) = x(2k) + \frac{g(k+1)+g(k)}{4}$

بنابراین میزان محاسبات و همچنین تاخیر محاسبات برای اعمال ضرایب پایین است.

در روند نمای زیر ترتیب کلی کارکرد بلوک اعمال تبدیل موجک نشان داده‌شده است و در ادامه به مراحل مختلف کار شرح داده می‌شود.



شکل (۶-۲) روند نمای بلوک dwt2d

هر پیکسل با یک سیکل کلاک از حافظه از طریق پورت B خوانده می‌شود و با توجه به رابطه (۶-۱) بعد از رسیدن سه پیکسل میتوان اولین ضریب بالاگذر را محاسبه کرد و سپس برای محاسبه ضریب پایین‌گذر نیاز به پیکسل زوج و همچنین مقدار قبلی و فعلی ضریب پایین گذر است؛ در نتیجه در ادامه کار، با دو سیکل کلاک دو ضریب بالاگذر و پایین‌گذر محاسبه می‌شوند. در حین اعمال فیلتر بر روی یک سطر، اطلاعات مربوط به آن سطر با ترتیب مناسب در یک بافر ذخیره می‌شود و بعد از اتمام فیلتر، یک سطر از نتایج موجود در بافر از طریق پورت A در حافظه و در محل تصویر اصلی با ترتیب مناسب بازنویسی می‌شود، پس از اتمام عملیات بر روی تمام سطرها و باز نویسی نتایج با ترتیب مناسب در حافظه، مراحل توضیح داده شده در بالا را بر روی ستون‌ها اعمال کرده و نتایج مجدّداً در حافظه باز نویسی می‌شود. همانطور که در روند نما نیز مشاهده می‌شود واحد اعمال فیلتر برای سطرها و ستون‌ها مشترک است و از طریق ترتیب خواندن پیکسل‌ها از حافظه می‌توان روند اعمال فیلترها را بر سطرها و ستون‌ها کنترل کرد. بلوک کنترل کننده روند کلی برای کنترل روند اعمال فیلترها بر روی سطرها و ستون‌ها و همچنین روند دریافت داده‌ها از ورودی و ارسال داده به خروجی و همچنین اعلام زمان اتمام تبدیل مورد استفاده قرار می‌گیرد. بعد از اتمام مراحل فوق و انجام کامل یک سطح موجک بر روی تصویر و ذخیره آن در حافظه بیت خروجی dwtf یک شده و با ارسال سیگنال اتمام کار به بلوک ft245 اطلاعات موجود در حافظه از طریق آن بلوک به تراشه FT245R ارسال شده و پس از تبدیل‌های انجام شده توسط تراشه به USB ارسال می‌شوند. در آخر اطلاعات توسط نرم‌افزار متلب دریافت شده و نمایش داده می‌شود.

۶-۲ نتایج و مقایسه‌ها

کد نوشته شده توسط نرم‌افزار Xilinx ISE 14.2 به برد طراحی شده در این پایان نامه انتقال داده شده است. تصویر انتقال داده شده به برد برای اعمال موجک، تصویر سیاه و سفید Cameraman[۱] با ابعاد

[۱] تصویر موجود در نرم‌افزار Matlab

۱۰۰×۱۰۰ پیکسل می‌باشد. نتایج گزارش دریافت شده از نرم‌افزار که شامل تعداد سلول‌ها و برش‌های استفاده شده و همچنین فرکانسی که مدار قابلیت کار در آن را دارد در جدول ۶-۱ آورده شده است.

جدول ۶-۱ نتایج FPGA

پارامتر	مقادیر بدست آمده	موجود
تعداد برش‌ها[1]	1117	3584
تعداد فلیپ فلاپ‌های برش	823	7168
تعداد LUT	2453	7168
فرکانس کاری	78 Mhz	

همانطور که در جدول هم دیده می‌شود طرح مورد نظر توانایی پردازش با فرکانس 78Mhz را دارا می‌باشد. کل زمان مورد نیاز برای انجام یک سطح تبدیل موجک بر روی تصویر بدون در نظر گرفتن زمان ارسال اطلاعات برابر 3.65 ms بدست آمد.

بعد از بازسازی تصویر دریافتی(۸ بیتی) و بازسازی آن مربع میانگین خطا (MSE) و همچنین حداکثر نسبت سیگنال به نویز (PSNR) با استفاده از رابطه ۶-۲ محاسبه شدند.

رابطه ۶-۲

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(pi - qi)^2$$

$$PNSR = 10log\frac{(255)^2}{MSE}$$

n= تعداد پیکسل‌ها

Pi= پیکسل تصویر اصلی

qi=پیکسل تصویر بازسازی شده

[1] برش یا Slice، هر CLB شامل چهار برش است

حداکثر نسبت سیگنال به نویز (PSNR) بدست آمده برای تصویر Cameraman به صورت ۸ بیتی برابر ۱۷/۰۷ می‌باشد.

۶-۳ مقایسه و بررسی کارایی

اکثر کارهای مشابه بر روی بردهای از پیش طراحی شده با FPGAهای با امکانات بالاتر و گرانتر و همچنین امکانات بیشترِ قرار داده شده بر روی برد انجام شده‌اند. در این پایان نامه به دلیل استفاده از FPGA با امکانات کمتر، برای صرفه‌جویی در حافظه از ابعاد فریم کوچکتر استفاده شده است. اگرچه فرکانس کاری و همچنین منابع موجود در FPGA نیز در مدل استفاده شده کمتر است، ولی در ادامه به مقایسه طرح پیاده سازی شده با چند مقاله مشابه می‌پردازیم.

در جدول ۶-۲ نتایج دو مقاله [23] [22] که به پیاده سازی موجک دو بعدی با استفاده از طرح بالابری پرداخته‌اند آورده شده است.

جدول ۶-۲ چند روش پیاده سازی شده

فیلتر موجک	5/3	5/3
ابعاد تصویر	256*256	256*256
تعداد بیت مورد استفاده برای هر پیکسل	8 bit	8 bit
تراشه FPGA مورد استفاده	XCV600E	XC4VLX15
تعداد برش‌ها	1835	2646
فرکانس کاری	108MHz	117.6MHz
زمان مورد نیاز	2.36 ms	-

در هر دو مقاله آورده شده FPGA مورد استفاده قویتر از FPGA مورد استفاده در این پایان نامه است. تعداد برش‌های استفاده شده در طرح پیاده سازی شده در این پایان نامه کمتر از هر دو مقاله است که دلیل اصلی آن تفاوت در طراحی و استفاده از فقط یک واحد پردازش برای اعمال بر سطرها و ستون‌ها

است. در [23] با توجه به بالاتر بودن فرکانس کاری مدار (که دلیل اصلی آن مدل FPGA مورد استفاده است) زمان مورد نیاز برای انجام یک سطح موجک نزدیک به زمان بدست آمده در این پایان نامه است. البته همانطور که در جدول نیز مشاهده می‌شود ابعاد تصویر اعمالی بزرگتر از تصویر استفاده شده در این پایان نامه است.

۴-۶ نتیجه گیری

با توجه به زمان بدست آمده برای انجام یک سطح موجک رسیدن به پردازش بلادرنگ برای انجام فشرده‌سازی رشته ویدیویی ممکن می‌باشد(البته با توجه به حافظه داخلی با ظرفیت پایین برای پیاده سازی بهینه نیاز به رم ناهمزمان با تاخیر پایین است). در این طرح برای مقایسه روش پیاده سازی شده با مقالات دیگر از تصویر بارگذاری شده در حافظه استفاده شده است، اما با توجه به راه اندازی دوربین متصل شده به بورد طراحی شده و دریافت تصویر از آن، می‌توان تصویر ورودی را از ماژول دوربین دریافت کرد، سپس تبدیل موجک و مراحل دیگر کار را بر روی آن انجام داد. اما دریافت تصویر از دوربین به علت پایین بودن فرکانس کاری دوربین موجب بالاتر رفتن زمان مورد نیاز برای انجام تبدیل موجک می‌شود و در نتیجه مقایسه کار با مقالات دیگر مشکل می‌شود. مرحله پیاده سازی کامل الگریتم فشرده‌سازی پیشنهادی به دلیل پیچیدگی بالای پیاده سازی در بستر FPGA ارزان قیمت مورد استفاده و کمبود وقت بطور کامل انجام نشد. در نهایت کد مربوط به راه‌اندازی و استفاده از تمام سخت افزارهای موجود بر روی برد شامل حافظه رَم همزمان، ماژول دوربین و رابط USB نوشته شده و از صحت عملکرد آن‌ها اطمینان حاصل شد تا برای پیاده‌سازی الگریتم پیشنهادی بطور کامل بتوان از ماژول‌های نوشته شده استفاده کرد.

فصل هفتم

نتیجه‌گیری و پیشنهاد راهکارهای آینده

۷-۱ نتیجه گیری

در این پایان نامه ابتدا یک بُرد سخت افزاری مبتنی بر FPGA برای فشرده‌سازی بلادرنگ و ارسال آن به کامپیوتر طراحی شد. در طراحی برد سعی شد با توجه به بودجه محدود امکانات اولیه مورد نیاز برای دریافت و پردازش ویدیو در برد آورده شود. امکانات حداقل بکار برده شده شامل ماژول دوربین ارزان قیمت، حافظه RAM همزمان، رابط USB و همچنین در دسترس گذاشتن پایه‌های FPGA برای اتصال قطعات دیگر و گسترش احتمالی در آینده است. در ادامه یک الگوریتم فشرده‌سازی بلوکی ساده سازی شده برای فشرده‌سازی ویدیو معرفی شد که در آن سعی شده بود از امکان موازی سازی FPGA استفاده شود و با پردازش بلوکی تصویر به صورت موازی سرعت پردازش را افزایش داد و با افزایش سرعت، فشرده سازی بلادرنگ ویدیو و ارسال رشته بیتی فشرده شده به پورت USB مقدور شود. از آنجا که اساس الگوریتم مورد استفاده بر پایه تبدیل موجک است، از روش تبدیل موجک با استفاده از طرح بالابریی استفاده شد زیرا این روش حجم محاسبات و حافظه کمتری نیاز دارد. در ادامه پیاده‌سازی سخت‌افزاری تبدیل موجک طرح بالابری بر روی یک تصویر نمونه انجام شد و زمان مورد نیاز برای انجام یک سطح موجک محاسبه شد. زمان بدست آمده نشان می‌داد که با استفاده از این روش می‌توان به فشرده‌سازی بلادرنگ ویدیو دست پیدا کرد.

۷-۲ پیشنهاد راهکارهای آینده

یکی از اصلی‌ترین راهکارهای افزایش کارایی استفاده از FPGA قویتر و در نتیجه داشتن منابع در دسترس بیشتر و فرکانس کاری بالاتر است. همچنین استفاده از دوربین با سرعت و کیفیت بالاتر و در نظر گرفتن یک سخت‌افزار جداگانه برای دریافت و پردازش ابتدایی اطلاعات دریافتی از دوربین برای بالاتر بردن سرعت پردازش داده‌های ورودی، همچنین استفاده از پروتکل‌های ارسال اطلاعات با سرعت بالاتر مانند USB3 که برای این کار می‌توان از تراشه‌های گرانتر و با کارایی بالاتر شرکت FTDI کمک گرفت. با استفاده از FPGA قویتر قدرت پردازش بالاتر رفته و می‌توان استانداردهای فشرده‌سازی

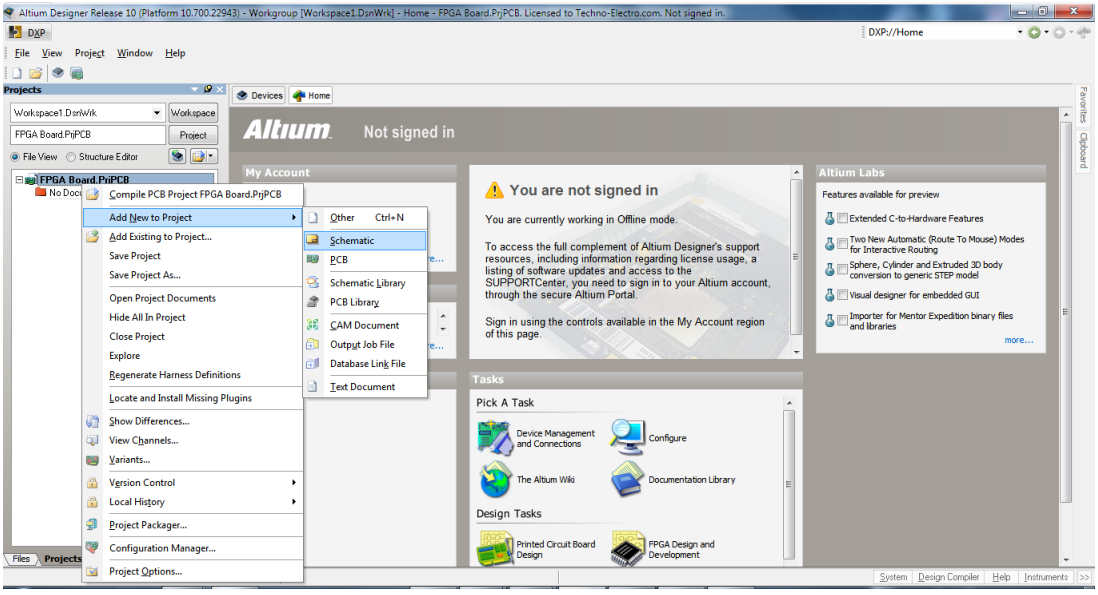
بلادرنگ جدید را پیاده سازی نمود که کارایی بالاتری نسبت به الگوریتم ساده شده مورد استفاده خواهند داشت.

پیوست آ

روند کلی طراحی برد PCB در محیط نرم‌افزار Altium Designer

آ.۱ ایجاد پروژه در محیط Altium Designer

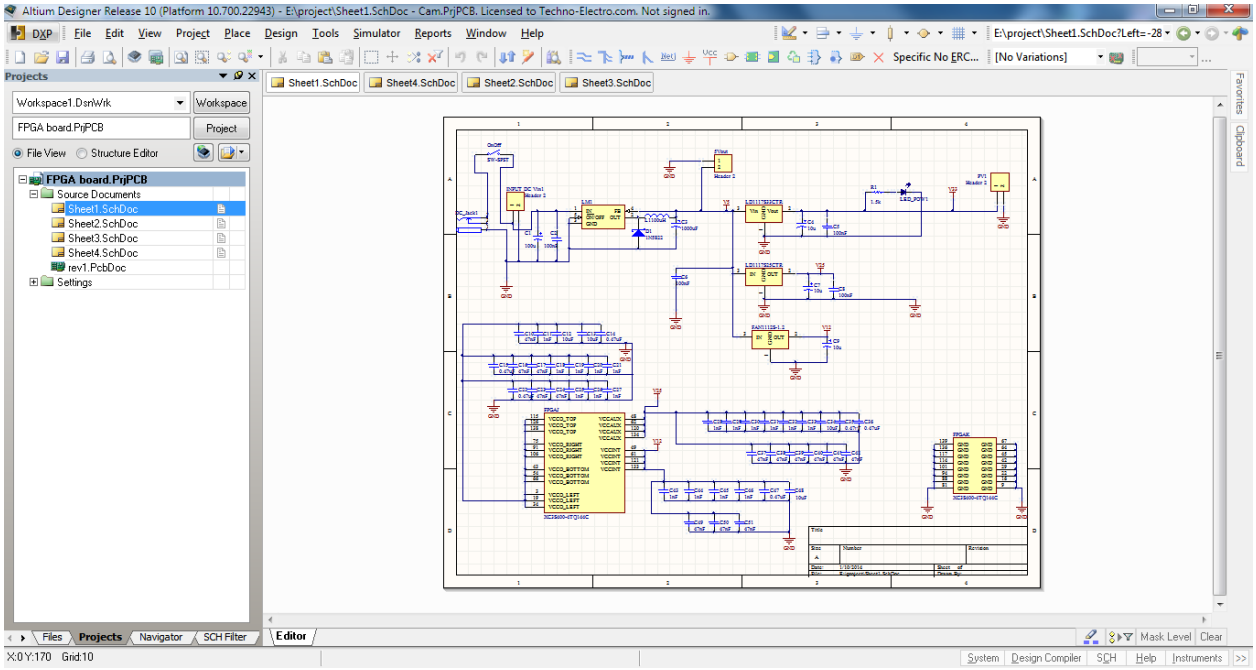
ابتدا نرم‌افزار Altium Designer را از منوی استارت باز کرده. برای ایجاد یک طرح PCB ابتدا باید یک پروژه (PCB project) تعریف کرد. برای ایجاد یک پروژه از مسیر File > New > Project > PCB Projectیک پروژه جدید ایجاد میکنیم. بعد از ایجاد پروژه جدید باید آن را ذخیره کرده و برای آن یک نام انتخاب کنیم. برای این منظور از مسیر File > Save Project را انتخاب کرده و برای پروژه خود یک نام دلخواه می‌گذاریم. (در این فایل نام FPGA Board را انتخاب میکنیم). بعد از ذخیره پروژه ابتدا باید طرح شماتیک مدار را در محیط شماتیک رسم کرده و فوت‌پرینت مناسب بر اساس قطعه موجود در بازار برای هر قطعه را انتخاب کنیم، سپس به PCB انتقال دهیم. برای ایجاد شماتیک مانند شکل آ-۱ بر روی پروژه خالی ایجاد شده راست کلیک کرده و یک شماتیک جدید ایجاد می‌کنیم.



شکل آ-۱ ایجاد شماتیک در پروژه

بعد از ایجاد یک صفحه شماتیک جدید از قسمت Place > part برای آورده قطعه مورد نظر اقدام می‌کنیم. قبل از جایگذاری قطعات ابتدا باید کتابخانه مربوط به قطعه مورد نظر را اضافه کرده سپس قطعه مورد نظر را از بین قطعات موجود در آن کتابخانه پیدا و جایگذاری کنیم. بعد از گذاشتن تمام قطعات برای سیم کشی بین آن‌ها Place > Wire را انتخاب و مدار را ترسیم میکنیم.

تراشه FPGA مورد استفاده در این پایان نامه (XC3S400-4TQ144C) دارای ۱۱ قسمت است که در طرح مورد نظر در ۴ صفحه شماتیک جدا قرار گرفته است. برای ارتباط بین صفحه‌های مختلف از اسم گذاری استفاده می‌شود و به هر پایه یک نام منحصربفرد تخصیص داده می‌شود. صفحه شماتیک اول مربوط به تغذیه مدار، صفحه دوم مربوط به قسمت JTAG، صفحه سوم مدار مربوط به رَم و صفحه چهارم باقی مداراها می‌باشد. در شکل آ-۲ صفحه اول مدار شماتیک که مربوط به تغذیه برد طراحی شده در پایان‌نامه است نشان داده شده.

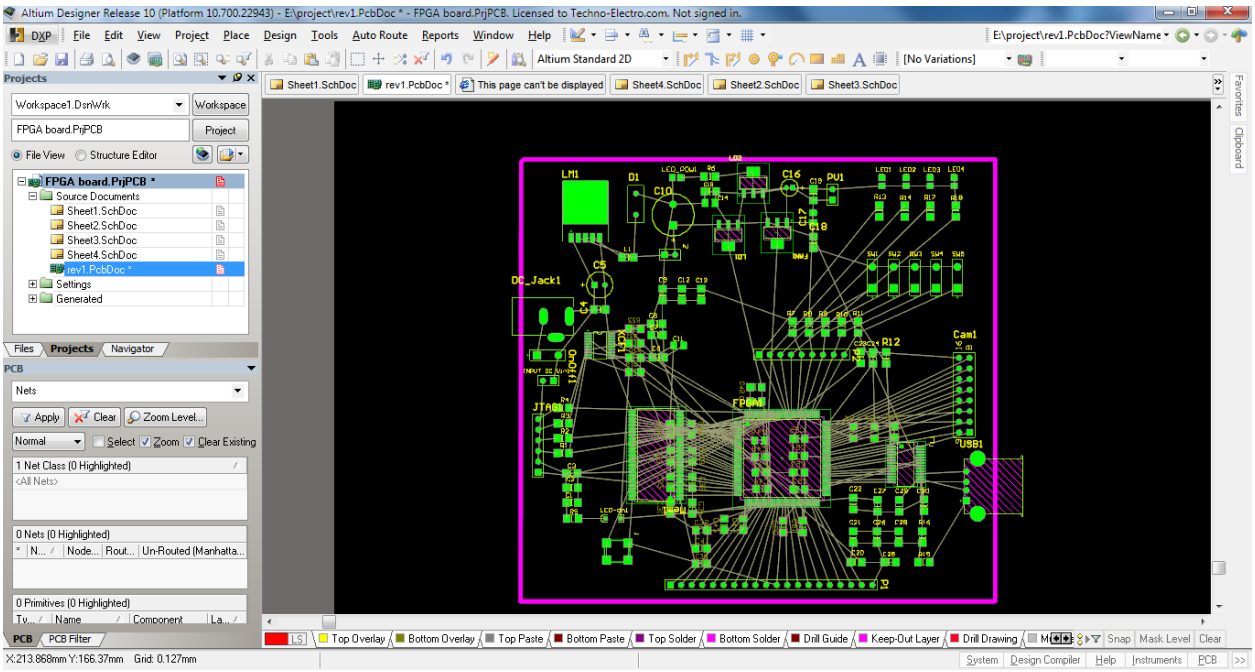


شکل آ-۲ صفحه اول مدار طراحی شده (تغذیه)

بعد از ترسیم کامل مدار شماتیک، بر روی پروژه ایجاد شده راست کلیک کرده از قسمت Add New To Project > PCB را انتخاب کرده و یک صفحه PCB خالی به پروژه اضافه می‌کنیم، سپس باید طرح شماتیک ترسیم شده را به PCB انتقال دهیم. برای این کار از قسمت Design > Update PCB Document را انتخاب می‌کنیم و طرح شماتیک مورد نظر را به PCB منتقل می‌کنیم.

بعد از انتقال طرح به PCB ، قطعات به‌صورت نا مرتب در PCB ظاهر می‌شوند. ابتدا باید یک کادر با اندازه مناسب برای برد PCB رسم کرد، برای رسم کادر از قسمت پایین صفحه PCB لایه را به Keep-Out Layer تغییر داده و از قسمت Place > Interactive Routing را انتخاب سپس کادر محافظ را

ترسیم میکنیم. بعد از ترسیم کادر باید قطعات را به‌صورت مناسب چیدمان کرد. برای این کار دو روش وجود دارد. روش چیدمان خودکار (Auto Placer) و روش چیدمان دستی، برای چیدمان خودکار از قسمت Tools > Component > Auto placer را انتخاب می‌کنیم. ولی در این طرح برای بالا بردن دقت از چیدمان دستی استفاده می‌کنیم. پس از چیدمان مناسب نتیجه مانند شکل آ-۳ می‌شود. در این مرحله باید مدار را مسیر یابی (Route) نمود، برای این کار برای بالا بردن دقت طراحی از مسیریابی دستی استفاده می‌کنیم.



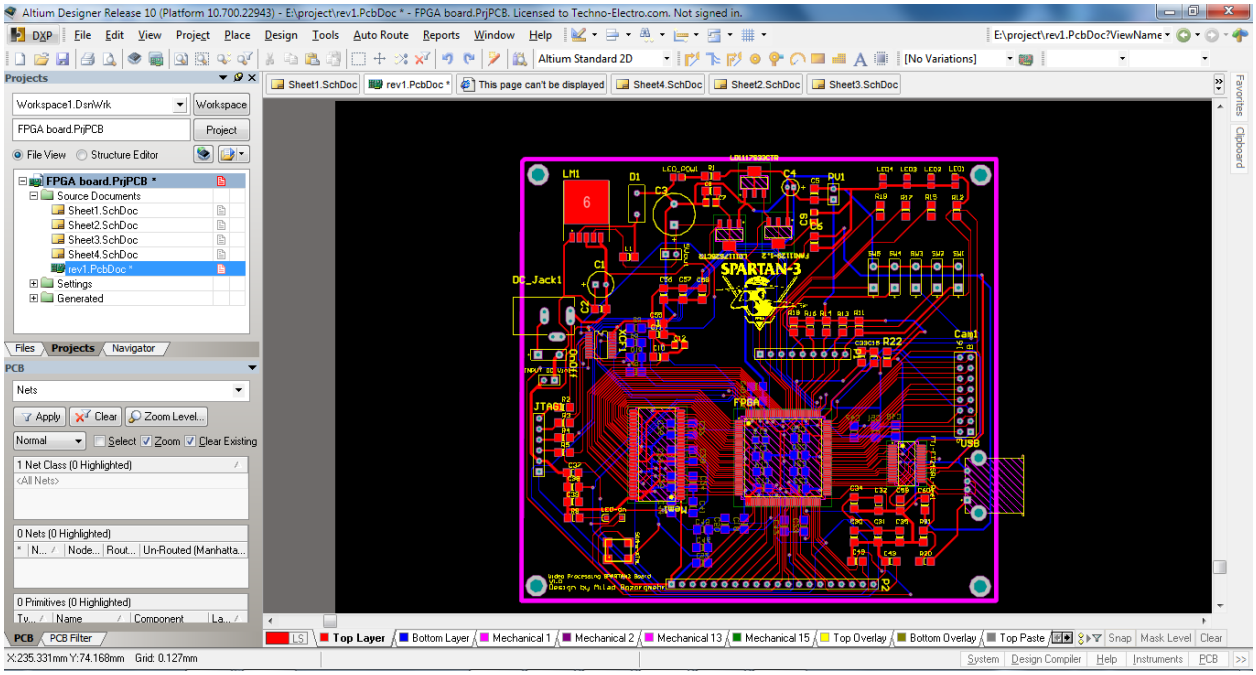
شکل آ-۳ طرح PCB بعد از چیدمان قطعات

قبل از مسیر یابی باید از قسمت Tools > Design Rule Check را انتخاب و قوانین مورد نظر را بر اساس محدودیت‌های موجود در ساخت برد و همچنین مونتاژ قطعات و ملاحظات فرکانسی تنظیم می‌نماییم.

برای مسیر یابی دستی از قسمت Place > Interactive Routing را انتخاب کرده سپس با انتخاب لایه مناسب بر روی پایه مورد نظر کلیک کرده و مسیر را رسم میکنیم. در مسیریابی دستی باید به نکات از

جمله طول خطوط، نزدیکی خطوط گذاشتن Via برای اتصال دو طرف برد در محل مناسب، در نظر گرفتن مسائل مربوط به سلف‌های پراکنده و ... توجه داشت.

بعد از کامل شدن مسیر یابی می‌توان نوشته‌ها از قبیل نام طراح نام برد و توضیح مربوط به نقاط مختلف برد را قرار داد. نتیجه نهایی در شکل آ-۴ نشان داده‌شده است.



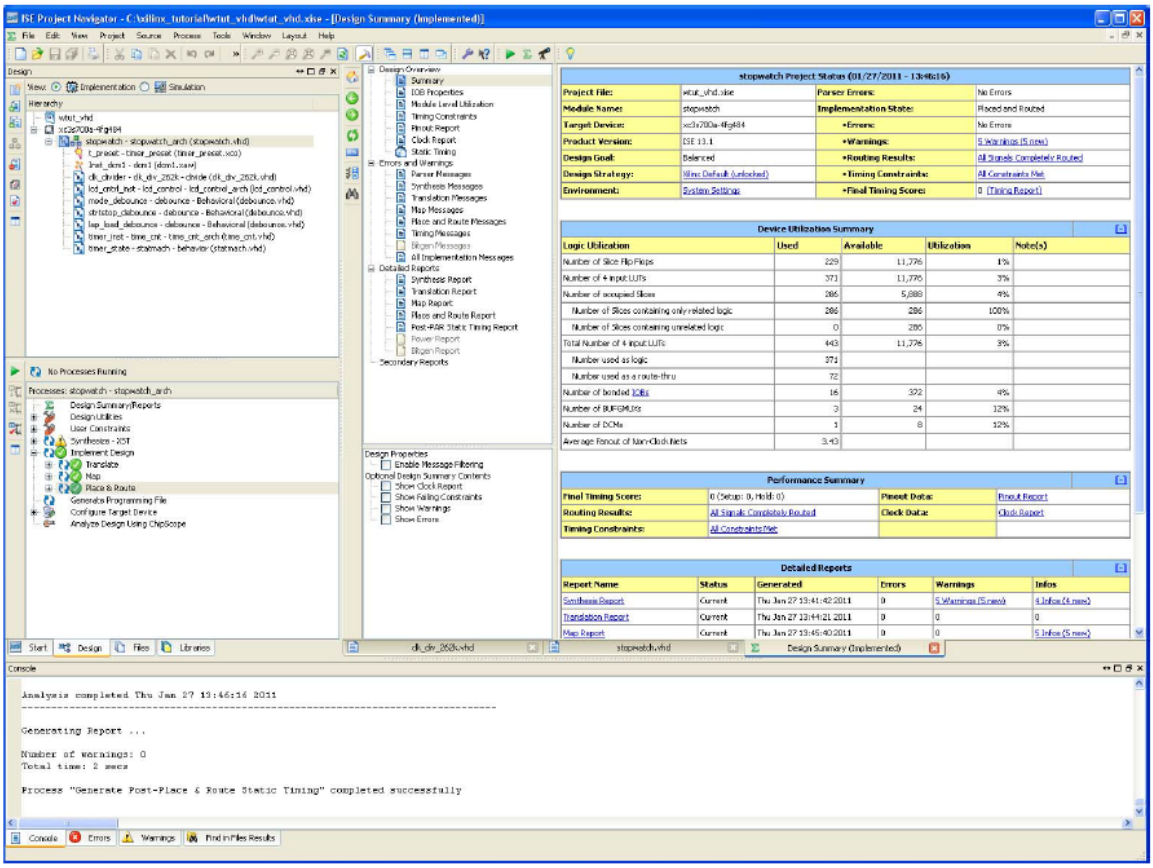
شکل آ-۴ شکل نهایی مدار

پیوست ب

روند کلی ایجاد پروژه و برنامه ریزی FPGA در محیط نرم افزار ISE

ب.۱محیط نرم‌افزار ISE

نرم افزار ISE همه جوانب روند طراحی را کنترل می‌کند. از طریق محیط هدایت‌گر پروژه (Project Navigator)، قادر خواهید بود به تمام روند طراحی و ابزار پیاده‌سازی و همچنین فایل‌های مرتبط با طرح دسترسی داشته باشید. در شکل (ب-۱) محیط پیش فرض هدایت‌گر پروژه مشاهده میشود که شامل چهار زیر پنجره است.

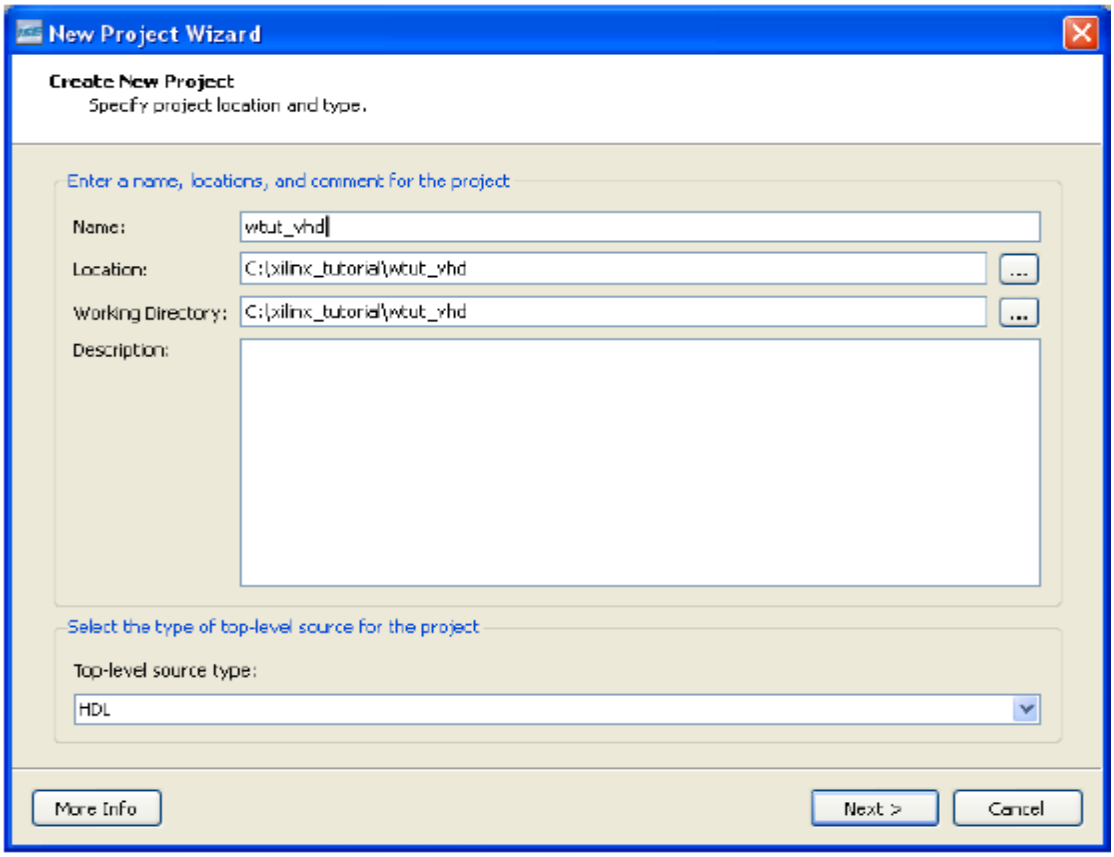


شکل (ب-۱) محیط پیش‌فرض هدایت‌گر پروژه

درشکل (ب-۱) در قسمت بالا سمت چپ پنجره برای دسترسی به فایل‌های پروژه ایجاد شده، و همچنین روند اجرای پروژه قرار دارد. پنجره قرار گرفته در قسمت پایین برای نشان دادن خطاها و هشدارهای موجود در پروژه است. و پنجره بزرگ سمت راست محیط کار است (work space) که برای نوشتن کد، طراحی شماتیک، مشاهده گزارش و مشاهده شکل موج شبیه سازی است.

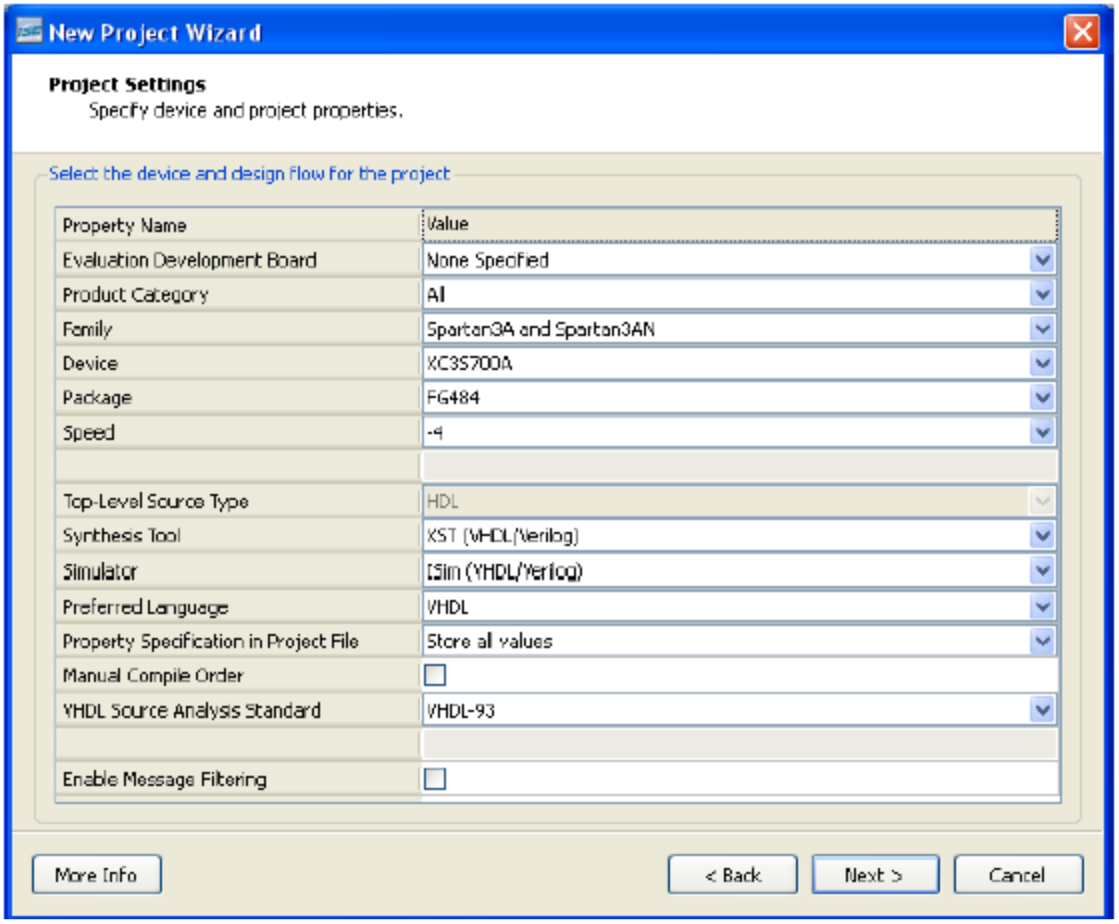
ب.۲ ایجاد پروژه

برای ایجاد پروژه جدید بعد از باز کردن نرم افزار از قسمت File > New Project را انتخاب می‌کنیم که سپس پنجره ایجاد پروژه (شکل (ب-۲)) ظاهر می‌شود.



شکل (ب-۲) پنجره ایجاد پروژه جدید مرحله ۱

پس از پر کردن قسمت‌های مربوط به نام و آدرس ذخیره‌سازی پروژه و انتخاب HDL ، بر روی NEXT> کلیک کرده و به مرحله بعد(شکل (ب-۳)) می‌رویم.



شکل (ب-۳) پنجره ایجاد پروژه جدید مرحله ۲

در مرحله بعد در ۶ قسمت اول نوع تراشه را معرفی کنیم. در قسمت بعد ابزار سنتز (Synthesis Tool) و شبیه ساز (Simulator) مورد نظر را انتخاب می‌کنیم. ابزار سنتز مورد استفاده در این پایان نامه XST است، همچنین شبیه ساز مورد استفاده ISim است . که این ابزار سنتز و شبیه ساز بصورت پیش فرض با نرم‌افزار ISE نسب می‌شوند. برای انتخاب دیگر ابزارهای سنتز و شبیه سازها باید آن‌ها را بصورت جداگانه نسب کنیم. در آخر زبان مورد نظر را انتخاب میکنیم و باقی مقادیر در حالت پیش فرض رها می‌شوند. مقادیر انتخاب شده برای این پایان نامه بصورت زیر هستند.

Product Category: **All**
Family: **Spartan3**
Device: **XC3S400**
Package: **TQ144**

speed: **-4**

Synthesis Tool: **XST (VHDL/Verilog)**

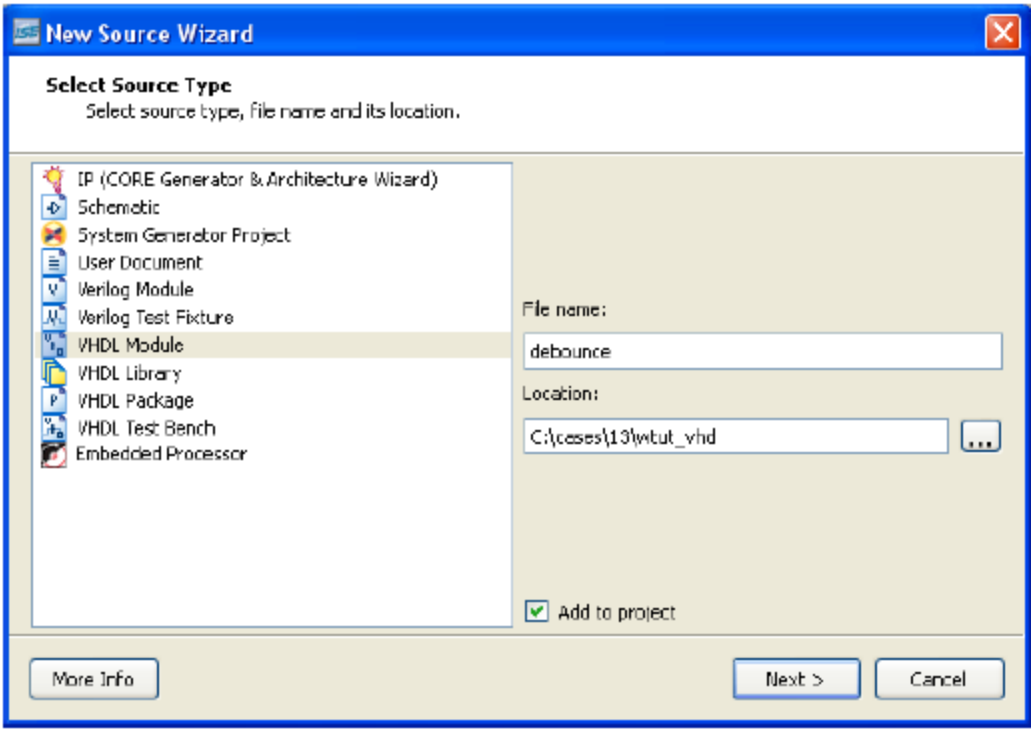
Simulator: **ISim (VHDL/Verilog**

Preferred Language: **VHDL**

بعد از پر کردن قسمت‌های ذکر شده بر روی NEXT> کلیک کرده و به محله بعد رفته و یک گزارش کلی از مقادیر انتخاب شده را مشاهده می‌کنیم، در آخر با کلیک بر روی Finish پروژه جدید ایجاد می‌شود.

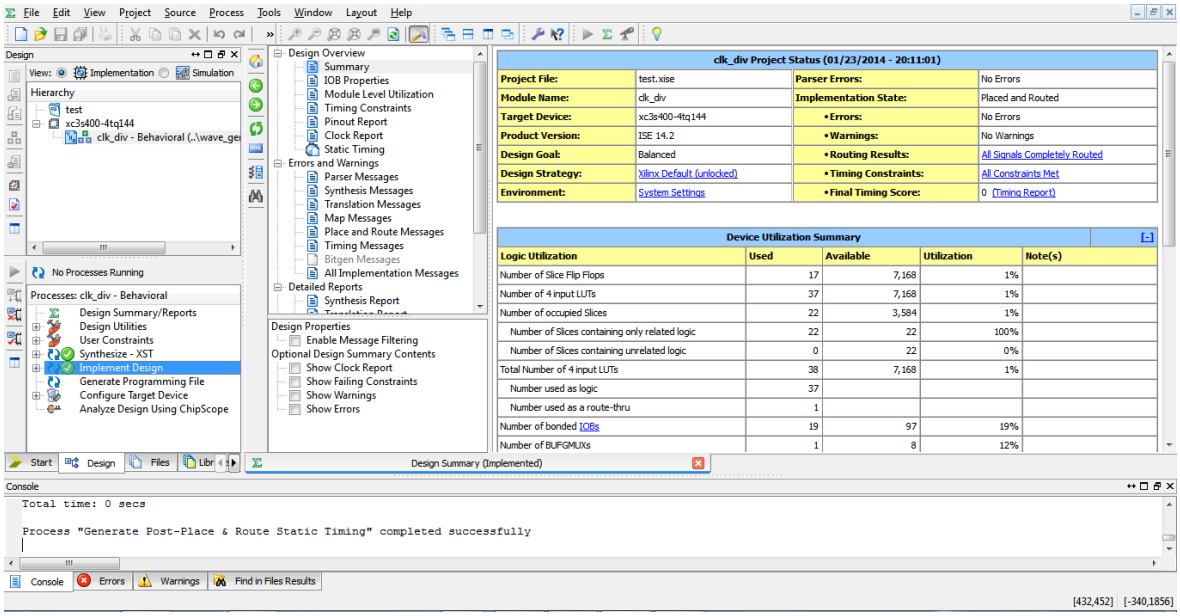
بعد از ایجاد یک پروژه خالی Project>New Source را انتخاب می‌کنیم سپس یک پنجره مانند شکل (ب-۴) ظاهر می‌شود. در این پنجره می‌توان نوع فایل مورد نظر برای اضافه کردن به پروژه را انتخاب کرد.

در این قسمت برای نوشتن کد VHDL از پنجره ایجاد شده VHDL Module را انتخاب می‌کنیم. در مرحله بعد یک پنجره برای مشخص کردن ورودی و خروجی‌های پروژه ایجاد شده ظاهر می‌شود. که پر کردن آن اختیاری است. ورودی و خروجی‌ها در محیط کد نویسی مشخص می‌شود ولی اگر از ابتدا آن‌ها را مشخص کنیم بعد از ظاهر شدن محیط برنامه نویسی قسمت مربوط به ورودی و خروجی ها مشخص شده است.



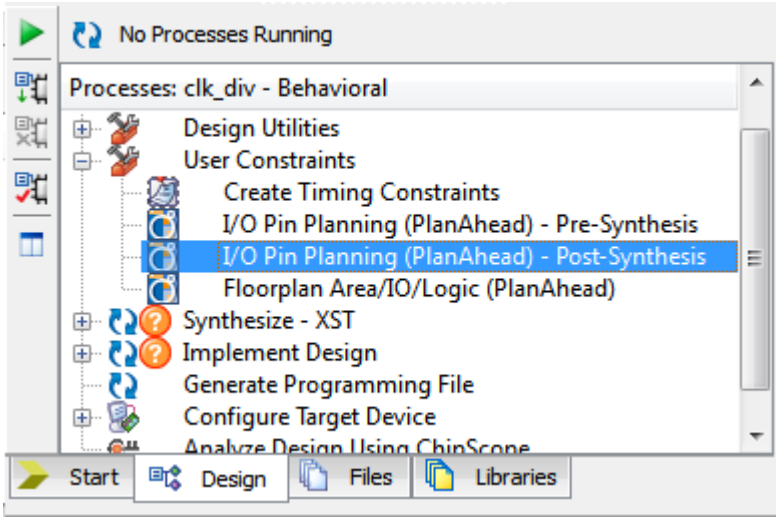
شکل (ب-۴) ایجاد یک منبع جدید

بعد از نوشتن کد VHDL مورد نظر نوبت به سنتز کردن آن می‌رسد. ابزار سنتز کد HDL مورد نظر را برای پیاده سازی آماده می‌کند و در صورت وجود خطا در برنامه آن را گزارش می‌کند. برای این کار در پنجره process (پایین سمت چپ) بر روی Implement Design کلیک کرده و منتظر می‌مانیم تا مراحل کار انجام شود. اگر کد نوشته شده خطا نداشته باشد نتیجه کار مانند شکل شده و Implement Design به رنگ سبز در می‌آید.



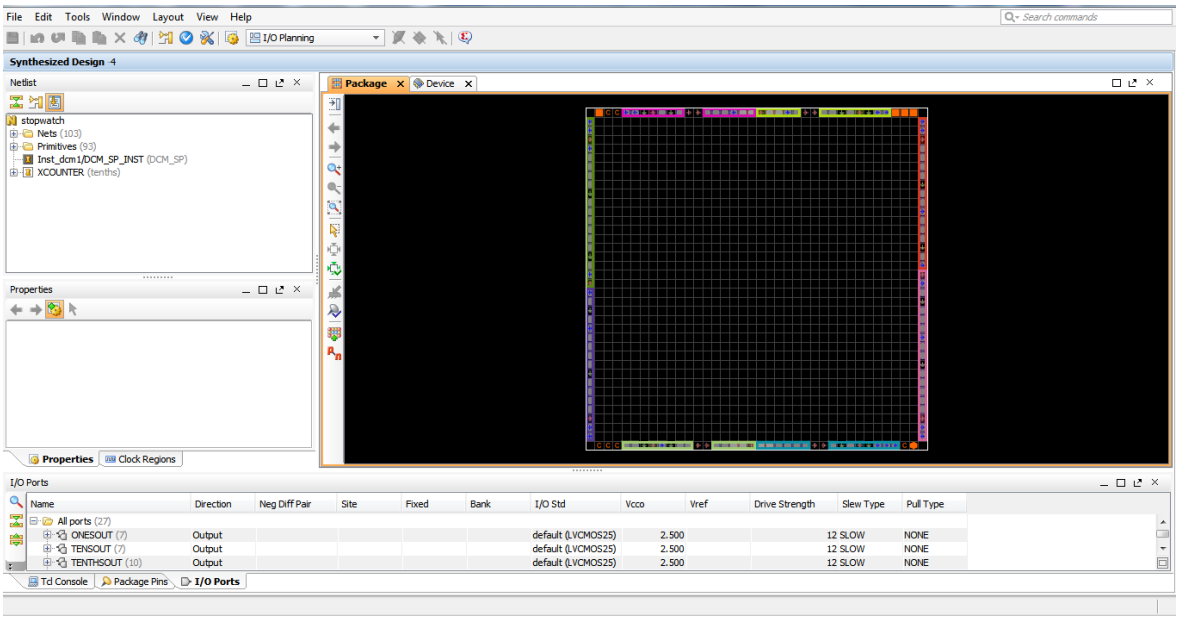
شکل (ب-۵) سنتز کردن موفق کد نوشته شده

بعد از سنتز کردن کد نوشته شده حال باید پایه‌های ورودی خروجی کد نوشته شده را بر روی FPGA مشخص کرد. برای این کار باید یک فایل با پسوند ucf به پروژه اضافه نمود و در آن فایل با دستورات خاص مربوط به آن پایه‌های مورد نظر را مشخص نمود. راه دیگر برای تعریف فایل ucf استفاده از محیط گرافیکی نرم‌افزار PlanAhead است که به صورت پیش فرض با مجموعه نرم افزار ISE نسب می‌شود. برای این کار در پنجره process بر روی علامت + کنار User Constrains کلیک کرده و سپس قسمت I/O Pin Planning post-synthesize را انتخاب می‌کنیم.(شکل (ب-۶))



شکل (ب-۶) انتخاب نرم‌افزار PlanAhead بای ایجاد فایل ucf

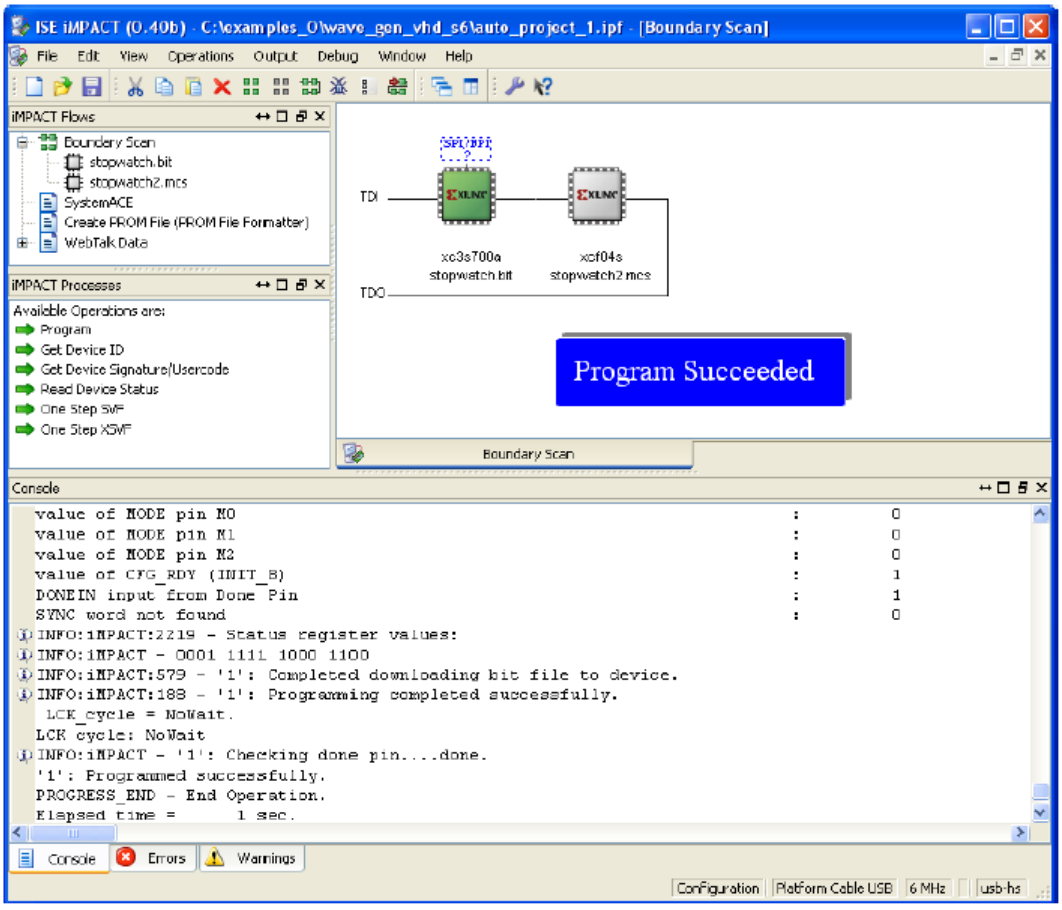
با انتخاب I/O Pin Planning نرم افزار PlanAhead باز می‌شود و در محیط گرافیکی آن شکل پایه‌های FPGA مشخص شده در اول پروژه، نشان داده شده است. همانطور که در شکل (ب-۷(دیده می‌شود شکل ظاهری FPGA مورد نظر نشان داده شده است و نام پایه‌های ورودی خروجی در پنجره پایین آورده شده است. برای مشخص کردن پایه‌ها می‌توان نام پایه مورد نظر را انتخاب کرده و آن را کشده (drag) و در پایه مورد نظر رها (drop) کنیم. با این کار ورودی یا خروجی انتخاب شده به پایه مورد نظر تخصیص می‌یابد.



شکل (ب-۷) محیط نرم افزار PlanAhead

بعد از مشخص کردن تمام ورودی خروجی‌ها اطلاعات را ذخیره کرده و از برنامه خارج می‌شویم.

در مرحله آخر نوبت به انتقال برنامه نوشته شده، به FPGA میرسد. برای انتقال برنامه به FPGA از پنجره Process، قسمت Configure Target Device را انتخاب می‌کنیم، سپس همه مراحل سنتز و کامپایل طی شده، و فایل قابل انتقال به FPGA با پسوند bin ایجاد شده و نرم افزار iMPACT باز می‌شود. در صفحه باز شده File>New را انتخاب کرده و-Configure Devices using Boundary Scan (JTAG) را انتخاب می‌کنیم. در صفحه باز شده راست کلیک کرده و Initialize Chain را انتخاب می‌کنیم، سپس نرم‌افزار بصورت خودکار FPGA متصل شده را شناسایی می‌کند. در آخر بر روی نام FPGA کلیک کرده و Program را انتخاب کرده و فایل bin موجود را انتخاب می‌کنیم. سپس با کلیک بر روی ok مرحله انتقال برنامه به FPGA آغاز می‌شود. در نهایت پیغامی مبنی بر برنامه ریزی موفق نشان داده می‌شود(شکل (ب-۸)).



شکل (ب-۸) برنامه ریزی موفق FPGA

مراجع

[1] M. Carraeu, "Hubble is fitted with a new ‘Eye’," Houston Chronicle, December 7, 1993.

[2] m. ghanbari, Standard Codecs Image compression to advanced video coding, The Institution of Engineering and Technology, 2011.

[3] T. SAVATIER, "Difference between MPEG-1 and MPEG-2 video," pp. 94-37, 1994.

[4] ITU-T recommendation H.263þþ, "‘Video coding for low bit rate communication," in *ITU-T SG16*, 2000.

[5] J. V. S. M. ,. R. J. Brian Schoner, "Techniques for FPGA Implementation of Video Compression System," *FPGA '95 Proceedings of the 1995 ACM third international symposium on Field-programmable gate arrays,* pp. 154-159, 1995.

[6] P. K. F. G. P. N. M. H. L. Ahmed Ben Atitallaha, "An FPGA implementation of HW/SW codesign architecture for H.263 video coding," *Int. J. Electron. Commun. (AEÜ),* pp. 605-620, 2007.

[7] H. a. S. D. MALAVER, "The LOT: transform coding without blocking effects," *IEEE Trans. Acoust. Speech Signal Process,* vol. 4, no. 37, pp. 553-559, 1989.

[8] J. SHAPIRO, "Embedded image coding using zero trees of wavelet coefficients," *IEEE Trans. Signal Process,* vol. 4, no. 12, p. 3445–3462, 1993.

[9] B. USEVITCH, "A tutorial on modern lossy wavelet image compression:foundation of JPEG 2000," *IEEE Signal Process,* vol. 5, no. 18, p. 22–35, 2001.

[10] R. W. S. a. F. J. CROCHIERE, "Digital coding of speech in sub bands," *Bell Syst. Tech,* p. 1069–1085, 1967.

[11] I. DAUBECHIES, "Orthonormal bases of compactly supported wavelets," *Pure Appl. Math,* p. 909–996, 1988.

[12] D. a. T. A. LE GALL, "Subband coding of images using symmetric short kernel filters and arithmetic coding techniques," *IEEE International Conference on Acoustics, Speech and Signal Processing,* p. 761–764, 1988.

[13] I. DAUBECHIES, "The wavelet transform, time frequency localization and signal analysis," *IEEE Trans. Inf. Theory*, vol. 5, no. 36, p. 961–1005, 1990.

[14] S. MALLAT, "A theory of multiresolution signal decomposition: the wavelet representation," *IEEE Trans. Pattern Anal*, vol. 7, no. 11, p. 674–693, 1989.

[15] W. Sweldens, "The lifting scheme: A custom-design construction of biorthogonal wavelets," 1996.

[16] W. Sweldens, "The lifting scheme: A construction of second generation wavelets," in *Industrial Mathematics Initiative*, University of South Carolina, 1995.

[17] J. Zhu, "Low-complexity Block dividing Coding Methode for Image Compression using Wavelets," Massey University, Palmerstone North, 2007.

[۱۸] P. P.chu, "نمونه سازی FPGA با مثال هایی ازVHDL," در *FPGA Prototyping By VHDL Examples: Xilinx Spartan -3 version*, بابل, علوم رایانه, ۱۳۹۰, p. 21.

[19] Xilinx, "Spartan 3 family complete Data Sheet," Xilinx, July 13,2004.

[20] Xilinx, "Power Distribution System (PDS) Design: Using Bypass/Decoupling Capacitors," XAPP623 (v2.1), 2005.

[21] FTDI, "FT245R USB FIFO I.C.," FTDI , 2005.

[22] P. C. K. M. a. Y. M.E. Angelopoulou, "Implementation and comparison of 5/3 Lifring 2D Discrete Wavelet Transform Computation Schedules on FPGAs," *Journal of VLSI Signal Processing 2007*, pp. 3-21, 2008.

[23] M. Z. T. S. M. A. B. B. M. a. R. T. Dhaha Dia, "Multi-level Discrete Wavelet Transform Architecture Design," *Proceedings of the World Congress on Engineering*, vol. 1, 2009.

Abstract

Today, the growing need to transmit and store information makes data compression an important issue. Variety of video data are increasingly produced, stored and transmitted in different application that make an important role in everyday life, exploration applications, space industry and military industry. In many of these applications real-time video streaming and storage is required. In real-time application, processing speed is critical issue, because high volume of input data needs to be compressed in real-time. Furthermore, the flexibility and the ability to update the system with low cost are also highly regarded. One of the best ways to increasing the speed is parallel processing. An appropriate strategy to achieve parallel processing is using FPGAs for their Parallelization capabilities and high flexibility. Parallelization ability is an appropriate strategy for increasing processing speed; high performance and speed are achievable with an appropriate design. In this thesis, we propose a method for real-time compressing in FPGA-based hardware.

In this regard, a low-complexity high parallelism block-based wavelet video compression algorithm with appropriate hardware compatibility, which has extracted from block compression algorithms such as EBCOT and SBHP, is purposed and designed. Then we design a hardware based on software needs. At the end, execution time and speed are compared with similar works. Result shows optimization in terms of use of available hardware resources compare with two other methods.

Key words: real time video compression – parallelization – wavelet transform – lifting scheme - FPGA

Shahrood University of Technology

Faculty of Electrical Engineering

Design, Simulation and Hardware Implementation of a Real-Time Video Compression Algorithm Based on FPGA

(M.S.C)

Milad Bozorgmehr

Supervisor(s):

Dr. Hadi Grailu

Date: **Jan 2014**