بسم الله الرحمن الرحیم

طراحی مدارهای دیجیتال با System Verilog

مولف

Mark Zwolinski

مترجمان:

علی کارگرنژاد

عضو هیئت علمی دانشگاه آزاد اسلامی واحد تهران جنوب

محمد تقی فاتحی خواجه

عضو هیئت علمی دانشگاه آزاد اسلامی واحد تهران جنوب

ندا پرقیمت

نشر نوآور

نشر نوآور

تلفن: ۲ - ۶۶۴۸۴۱۹۰

سرشناسه	: زولینسکی، مارک Mark ,Zwolinski
عنوان و نام پدیدآور	: طراحی مدارهای دیجیتال با System Verilog /مولف[مارک زولینسکی] ; مترجمان علی کارگرنژاد ، محمدتقی فاتحی‌خواجه، ندا پرقیمت.
مشخصات نشر	: تهران : نوآور: پارسیا، ۱۳۹۰.
مشخصات ظاهری	: ۳۵۲ ص.
شابک	: ۹۷۸-۶۰۰-۱۶۸-۰۶۵-۶
وضعیت فهرست نویسی	: فیپا
یادداشت	: عنوان اصلی: Digital system design with SystemVerilog,C2010.
موضوع	: وریلاگ (زبان توصیفی سخت‌افزار کامپیوتر)
موضوع	: کامپیوترهای رقمی -- طرح و ساختمان
موضوع	: کامپیوترها -- شبیه‌سازی
شناسه افزوده	: فاتحی خواجه، محمدتقی، ۱۳۴۸ - مترجم
شناسه افزوده	: پرقیمت، ندا، ۱۳۶۵ - مترجم
شناسه افزوده	: کارگرنژاد، علی، ۱۳۵۵ - مترجم
رده بندی کنگره	: TK۷۸۸۵ /۷/ز۹ط۴ ۱۳۹۰
رده بندی دیویی	: ۳۹۰۲۸۵۵۳/۶۲۱
شماره کتابشناسی ملی	: ۲۷۰۰۵۷۸

طراحی مدارهای دیجیتال با System Verilog

مؤلف:	**مارک زولینسکی**
مترجمان:	**علی کارگرنژاد، محمدتقی فاتحی خواجه، ندا پرقیمت**
ناشر:	**نوآور**
ناشر همکار:	**پارسیا**
شمارگان:	**۵۰۰ نسخه**
حروف‌نگاری:	**واحد رایانه نوآور**
مدیر تولید:	**محمدرضا نصیرنیا**
نوبت چاپ:	**اول - ۱۳۹۰**
شابک:	**۹۷۸-۶۰۰-۱۶۸-۰۶۵-۶**

قیمت: تومان

نمایشگاه دائمی و مرکز فروش:

نوآور: تهران - خ انقلاب، خ فخررازی، خ شهدای ژاندارمری نرسیده به خ دانشگاه ساختمان ایرانیان، پلاک ۵۸، طبقه دوم، واحد ۶

تلفن مرکز پخش: ۹۲-۶۶۴۸۴۱۹۱-۰۹۱۲۶۰۶۲۳۸۳

www.noavarpub.com

فروشگاه ۱: تهران خ انقلاب، بین خ ۱۲ فروردین و اردیبهشت، پلاک ۱۳۱۲، کتابفروشی صانعی تلفن: ۶۶۴۰۹۹۲۴ - ۶۶۴۰۵۳۸۵

فروشگاه ۲: تهران خ انقلاب، نبش خ ۱۲ فروردین پلاک ۱۳۱۰، کتابفروشی الیاس تلفن: ۶۶۹۵۵۸۷۸ - ۶۶۴۰۵۰۸۴

فروشگاه ۳: تهران خ انقلاب، مقابل دانشگاه تهران، جنب بانک ملت، پلاک ۱۲۱۲، کتابفروشی گوتنبرگ تلفن: ۶۶۴۱۳۹۹۸-۶۶۴۰۲۵۷۹

حق چاپ و نشر برای ناشر محفوظ است

پیشگفتار

درباره این کتاب

بعد از این که "کتاب طراحی سیستم دیجیتال با VHDL" منتشر شد، ایده نوشتن یک کتاب طراحی دیجیتال بوسیله یک زبان توصیف سخت افزار به نظر جدید می‌آمد. درهمان زمان چندین کتاب مشابه با همین موضوع انتشار پیدا کرد. اکنون کتاب طراحی سیستم دیجیتال با VHDL توسط چندین دانشگاه به عنوان مرجع اصلی پذیرفته شده و به زبان‌های لهستانی، چینی، ژاپنی و ایتالیایی ترجمه شده است. من قبلا درباره نوشتن طراحی سیستم دیجیتال با زبان وریلاگ فکر کرده بودم اما در مورد استفاده از زبان وریلاگ به عنوان زبانی برای تدریس کمی تردید داشته و دارم، با وجود اینکه به طور گسترده‌ای مورد استفاده می‌باشد. طولی نکشید که بعد از انتشار ویرایش دوم طراحی سیستم دیجیتال با VHDL، System Verilog به عنوان یک زبان جدید توصیف سخت افزار پدیدار شد. این زبان جدید بسیاری از تردیدهای من درباره وریلاگ را از بین برد و حتی مزیت‌هایی را در برابر VHDL عرضه کرد. از این رو موفقیت کتاب اول و پیدایش زبان جدید مرا متقاعد کرد که زمان ویرایش جدید فرا رسیده است.

این کتاب به عنوان کتاب درسی مختص دانشجویان دوره لیسانس و هم فوق لیسانس می‌باشد. اکثر کتاب‌های وریلاگ و سیستم‌وریلاگ، بر اساس طرح‌های عملی برای مهندسین بنا نهاده شده‌اند. بنابراین بعضی از ویژگی‌های سیستم‌وریلاگ به هیچ عنوان در این کتب بیان نشده‌اند. در مقابل، جنبه‌هایی از طراحی دیجیتال در این کتاب پوشش داده شده که در کتاب‌های نمونه سیستم‌وریلاگ دیگر نخواهد بود.

سرفصل‌ها برای مهندسی برق، الکترونیک و کامپیوتر در میان کشورها، دانشگاه‌ها یا کالج‌ها متفاوت می‌باشد. مطالب مطرح در این کتاب برای دانشجویان سال دوم و سوم کارشناسی و همچنین دانشجویان کارشناسی ارشد در نظر گرفته شده است. این طور فرض می‌شود که دانشجویان با قواعد جبر بولی و طراحی منطق ترکیبی آشنا هستند. در دانشگاه ساوتمپتون انگلستان سر فصل‌های دانشجوی سال اول دوره لیسانس، شامل طراحی ترتیبی آسنکرون و منطق قابل برنامه‌ریزی نیز می‌باشد. بنابراین، این کتاب بر این اساس به وجود آمده است. اغلب این گونه تصور می‌شده که مباحثی مانند سیستم‌وریلاگ برای تدریس در سال دوم بسیار اختصاصی بوده و بهتر است به سال آخر یا دوره فوق لیسانس موکول شود. دلایل محکم و خوبی وجود دارد برای اینکه چرا باید سیستم‌وریلاگ هرچه زودتر در برنامه درسی گنجانده شود. با افزایش پیچیدگی مدارهای مجتمع، کسب دانش علم سیستم‌وریلاگ و ابزارهای طراحی مربوطه یک نیاز برای دانش آموختگان به شمار می‌آید. اگر این مباحث را به سال آخر موکول کنیم، زمان کمی و شاید هیچ زمانی برای اینکه دانشجویان این علم را در کار پروژه‌ای به کار ببندند وجود نداشته باشد. ثانیاً گفت و گو با همکاران در بسیاری از کشورها نشان داد که دانشجویان امروزی برگزیدن علوم

کامپیوتر یا مهندسی کامپیوتر را به مهندسی برق یا الکترونیک ترجیح می‌دهند. سیستم‌وریلاگ مفاهیم جذابی را برای دانشجویان کامپیوتر در طراحی سخت افزار عرضه کرده است. سرانجام ابزارهای شبیه‌سازی و سنتز و بسته طراحی FPGA ساخته شده و به طور نسبتاً ارزان در موسسه‌های آموزشی و روی سیستم عامل کامپیوترهای شخصی در دسترس می‌باشند.

ساختار این کتاب

فصل ۱ ایده‌های را که در بطن این کتاب است، معرفی می‌کند، به طور مثال استفاده از ابزارهای خودکارسازی طراحی الکترونیک و CMOS و تکنولوژی منطق قابل برنامه ریزی. ما همچنین بعضی از مشکلات مهندسین مثل محدوده نویز و گنجایش خروجی را در نظر گرفتیم.

در فصل ۲ قواعد جبر بولی و طراحی منطق ترکیبی مرور می‌شوند. مسئله مهم زمان‌بندی و مشکلات مربوط به هزارد مورد بررسی قرار می‌گیرند. برخی تکنیک‌های مقدماتی برای نشان دادن داده‌ها مطرح می‌شود.

در فصل ۳ سیستم‌وریلاگ به واسطه مدل‌های اولیه گیت منطقی معرفی می‌شود. در این فصل بر اهمیت کد مستند شده تاکید می‌شود. ما نشان می‌دهیم که چطور نت لیست‌های گیت‌های اولیه ساخته می‌شوند و چگونه تاخیرها به واسطه گیت‌ها مدل‌سازی می‌شوند. ما همچنین مدل‌های پارامتری را مطرح خواهیم کرد. ایده استفاده از سیستم‌وریلاگ برای تصدیق مدل‌ها، با استفاده از testbench تعریف میگردد.

در فصل ۴ تکنیک‌های متنوع مدل‌سازی شرح داده می‌شود. بلوک‌های ساختاری ترکیبی، دیکدرها، انکدرها، مالتی پلکسرها، جمع کننده‌ها و چک کننده‌های پریتی مدل‌سازی شده، با استفاده از یک رشته ساختار کد همزمان و ترتیبی سیستم‌وریلاگ، مدل سازی میشوند. مدل‌های سخت افزاری سیستم‌وریلاگ دراین فصل معرفی می‌شوند و در فصول ۵، ۶ و ۷ عملا مدل‌های سخت افزاری سنتزپذیر بیان میشوند. گرچه بحث اینکه چه چیزهایی دقیقا پشتیبانی می‌شوند تا فصل ۱۰ به عقب افتاده است. روش طراحی testbench دوباره در فصل ۴ مطرح می‌شود. به علاوه نماد وابستگی IEEE معرفی می‌شود.

فصل ۵ بلوک‌های ساختاری ترتیبی گوناگونی را نشان می‌دهد: لچ‌ها، فلیپ فلاپ‌ها، ثبات‌ها، شمارنده‌ها، حافظه و یک مالتی پلکسر ترتیبی. به همان سبکی که در فصل ۴ استفاده شد با نماد وابستگی IEEE، طراحی testbench و معرفی ساختار کدبندی سیستم‌وریلاگ.

فصل ۶ شاید مهمترین فصل این کتاب باشد و در مورد این بحث می‌کند که در اصول طراحی دیجیتال چه چیزهایی ممکن است مطرح شود:طراحی ماشین‌های حالت متناهی. علائم چارت ASM بیان می‌شود. فرایند طراحی از چارت ASM به فلیپ فلاپ‌های نوع D و منطق حالت بعدی و خروجی

شرح داده می‌شود. مدل‌های سیستم‌وریلاگ ماشین‌های حالت معرفی می‌شوند.

در فصل ۷ مفاهیم سه فصل قبل ترکیب می‌شوند. علائم چارت ASM برای بیان ماشین‌های حالت کوپله شده و خروجی‌های رجیستر شده، توسعه میابند و بنابراین مسیرداده-کنترل‌کننده، قسمت‌بندی می‌شود. از این رو ما ایده دستورالعمل در قالب سخت افزار را توضیح می‌دهیم و مدل‌سازی یک ریزپردازنده بسیار مقدماتی را در سیستم‌وریلاگ ادامه می‌دهیم. این وسیله‌ای را برای معرفی واسطها و پکیج‌ها تأمین می‌کند.

طراحی testbench در فصل ۸ با جزئیات بیشتر مورد بحث قرار می‌گیرد. پس از پوشش‌دهی تکنیک‌های فصل‌های قبل، ما بحث درباره معماری testbench، تولید آزمون تحمیلی تصادفی و تائید مبتنی بر اثبات را ادامه می‌دهیم.

سیستم‌وریلاگ اصولا یک زبان مدل‌سازی باقی میماند. فصل۹ عملیات یک شبیه ساز سیستم‌وریلاگ را توصیف می‌کند. ابتدا ایده شبیه‌سازی رویدادگرا توضیح داده می‌شود و سپس ویژگی‌های مخصوص سیستم‌وریلاگ مورد بحث قرار می‌گیرد. مسئله دیگری که به طور فزاینده مهم شمرده می‌شود نقش سیستم‌وریلاگ به عنوان یک زبان برای توصیف مدل‌های سنتز به همان صورت که در فصل ۱۰ توصیف شد، می‌باشد. امروزه نوع عمده ابزارهای سنتز که در دسترس است، برای سنتز RTL می‌باشد. این ابزارها می‌توانند به وجود فلیپ فلاپ‌ها و لچ‌ها در یک مدل سیستم‌وریلاگ پی ببرند. این ساختارها توصیف شده‌اند. متقابلاً فلیپ فلاپ‌ها میتوانند به غلط ایجاد شوند اگر توصیف ضعیف نوشته شده باشد و یا توصیف شامل خطاهای معمول باشد. فرایند سنتز می‌تواند توسط محدودیت‌هایی کنترل شود. به دلیل اینکه این محدودیت‌ها خارج از خود زبان هستند آنها در شرایط کلی مورد بحث قرار می‌گیرند. ساختارهای مناسب برای FPGA بیان شده است. و در نهایت سنتز رفتاری که انتظار می‌رود که یک تکنولوژی طراحی مهم شود، به طور خلاصه بررسی می‌شود.

فصل ۱۱ و ۱۲ به مباحث آزمون و طراحی برای آزمون اختصاص داده شده‌اند. اغلب این سطح از مطلب مورد اهمال قرار می‌گرفت. اما امروزه به عنوان یک بخش مهم از فرایند طراحی شناخته می‌شود. در فصل ۱۱ ایده مدل‌سازی خطا معرفی شده است. سپس شیوه‌های تولید آزمون بیان می‌شود. اثرات یک آزمون توسط شبیه‌سازی خطا تعیین می‌شود.

در فصل ۱۲ سه اصل مهم در طراحی برای آزمون شرح داده می‌شود: مسیر اسکن، آزمون خودساخته و اسکن مرزی. این همیشه یک موضوع بی‌نتیجه بوده است. اما یک شبیه ساز سیستم‌وریلاگ به طور مثال می‌تواند به منظور اینکه نشان دهد چگونه یک ساختار آزمون خودساخته می‌تواند اثرات متفاوتی برای مدارهای عاری از خطا و خطادار تولید کند، استفاده شود.

ما در فصل ۱۳ از سیستم‌وریلاگ به عنوان ابزاری برای کشف رفتارهای متناقض در مدارهای ترتیبی آسنکرون استفاده می‌کنیم. اگرچه روش غالب طراحی به طور رایج سنکرون است. محتملاً سیستم‌های دیجیتال به طور فزاینده شامل ارتباط آسنکرون مدارهای سنکرون با یکدیگر خواهند بود. ما مفاهیم مد

اساسی را معرفی می‌کنیم و نشان می‌دهیم چگونه مدارهای آسنکرون تجزیه و تحلیل و طراحی می‌شوند. ما از شبیه‌سازی سیستم‌وریلاگ به منظور توضیح مشکلات هزاردها، رقابت‌ها و نقض زمان راه اندازی و نگه داری استفاده می‌کنیم. ما همچنین مشکل شبه پایداری را مورد بررسی قرار می‌دهیم.

فصل آخر Verilog-AMS و مدل‌سازی سیگنال ترکیبی را معرفی می‌کند. توضیح خلاصه‌ای از مبدل‌های دیجیتال به آنالوگ و آنالوگ به دیجیتال آورده شده است. ساختارهای Verilog-AMS که به عنوان مبدل‌ها مدل شده‌اند بیان شده است. ما اینجا همچنین ایده حلقه قفل شده فاز را معرفی می‌کنیم و یک مدل ساده سیگنال ترکیبی را ارائه دادیم. پیوست به طور خلاصه تشریح می‌کند که چگونه سیستم‌وریلاگ با ورژن‌های قبلی وریلاگ متفاوت است. انتهای هر فصل شامل تعدادی تمرین می‌باشد. این تمرین‌ها همچنین بصورت کمکی هستند تا دستورات هر فصل را شبیه‌سازی و یا در صورت لزوم سنتز کنید. برای اجرای این اعمال شبیه‌سازی و سنتز ممکن است خواننده خودش مجبور به نوشتن testbenchها و فایل‌های محدودیت (قید) شود. مثال‌ها روی وب سایت zwolinski.org قابل دسترس است.

نشرنوآور
تلفن: ۲-۶۶۴۸۴۱۹۱

فهرست

پیشگفتار

فصل ۱ / مقدمه

۱-۱ طراحی دیجیتال امروزی(نوین)
۱-۲ طراحی با زبان‌های توصیف سخت افزار
۱-۲-۱ طراحی خودکار
۱-۲-۲ SystemVerilog چیست؟
۱-۲-۳ VHDL چیست؟
۱-۲-۴ شبیه سازی
۱-۲-۵ سنتز
۱-۲-۶ استفاده مجدد
۱-۲-۷ تأیید
۱-۲-۸ روند طراحی
۱-۳ تکنولوژی CMOS
۱-۳-۱ گیت‌های منطقی
۱-۳-۲ ASICها و FPGA
۱-۴ منطق قابل برنامه‌ریزی
۱-۵ مشخصات الکتریکی
۱-۵-۱ محدوده نویز
۱-۵-۲ گنجایش خروجی

فصل ۲ / طراحی منطق ترکیبی

۲-۱ جبر بولی
۲-۱-۱ مقادیر
۲-۱-۲ عملگرها
۲-۱-۳ جداول درستی
۲-۱-۴ قوانین جبر بولی
۲-۱-۵ قانون دمورگان
۲-۱-۶ قضیه بسط شانن
۲-۲ گیت‌های منطقی
۲-۳ طراحی منطق ترکیبی
۲-۳-۱ حداقل‌سازی منطق
۲-۳-۲ جدول‌های کارنو
۲-۴ زمان‌بندی
۲-۵ کدهای عددی
۲-۵-۱ اعداد صحیح
۲-۵-۲ اعداد با ممیز ثابت
۲-۵-۳ اعداد با ممیز شناور
۲-۵-۴ کاراکترهای الفبایی- عددی
۲-۵-۵ کدهای گری
۲-۵-۶ بیت‌های توازن

فصل ۳ / منطق ترکیبی با استفاده از مدل‌های گیت سیستم‌وریلاگ

۳-۱- فایل‌ها و ماژول‌ها
۳-۲ شناسه‌ها، فضاها و توضیحات
۳-۳ مدل‌های گیت‌پایه
۳-۴ یک نت لیست ساده
۳-۵ مقادیر منطقی
۳-۶ انتساب‌های پیوسته
۳-۶-۱ عملگرهای سیستم‌وریلاگ
۳-۷ تأخیرها
۳-۸ پارامترها
۳-۹ Testbench(بستر آزمون)

فصل ۴ / بلاک‌های ساختار ترکیبی

۴-۱ مالتی پلکسر (تسهیم‌کننده)
۴-۱-۱ مالتی پلکسر ۲ به ۱
۴-۱-۲ مالتی پلکسر ۴ به ۱
۴-۲ دیکدر (رمزگشا)
۴-۲-۱ دیکدر ۲ به ۴
۴-۲-۲ دیکدر پارامتری
۴-۲-۳ دیکدر سِوِن سگمنت (7-seg)
۴-۳ رمزگذار اولویت
۴-۳-۱ مقادیر یکتا و بی‌اهمیت
۴-۴ جمع‌کننده‌ها
۴-۴-۱ مدل تابعی
۴-۴-۲ جمع‌کننده موجی

۶-۲-۲ ثبات‌های حالت
۶-۲-۳ طراحی یک شمارنده سه بیتی
۶-۳ ماشین‌های حالت الگوریتمی
۶-۴ سنتز از روی چارتهای ASM
۶-۴-۱ پیاده‌سازی سخت افزار
۶-۴-۲ تخصیص حالت
۶-۴-۳ حداقل‌سازی حالت
۶-۵ ماشین‌های حالت در سیستم‌وریلاگ
۶-۵-۱ اولین مثال
۶-۵-۲ تشخیص‌دهنده بیت توازن متوالی
۶-۵-۳ ماشین Vending
۶-۵-۴ ذخیره‌سازی داده
۶-۶ test benchها برای ماشین حالت

فصل ۷/ سیستم‌های ترتیبی پیچیده

۷-۱ ماشین‌های حالت به هم پیوسته
۷-۲ تقسیم‌بندی مسیر داده- کنترل کننده
۷-۳ دستورالعمل‌ها
۷-۴ یک میکروپروسسور ساده
۷-۵ مدل سیستم‌وریلاگ یک میکروپروسسور ساده

فصل ۸ / نوشتن Testbench

۸-۱ Testbenchهای پایه
۸-۱-۱ تولید کلاک
۸-۱-۲ Reset و سایر سیگنال‌های قطعی
۸-۱-۳ نمایش پاسخ‌ها
۸-۱-۴ پاسخ‌های موقت
۸-۱-۵ بردارهای تست از یک فایل
۸-۲ ساختار Testbench
۸-۲-۱ برنامه‌ها
۸-۳ تولید محرک‌های تصادفی ساختگی
۸-۳-۱ برنامه‌نویسی شی‌گرا
۸-۳-۲ تولید عدد تصادفی (Randomization)
۸-۴ تأیید مبتنی بر بازبینی

فصل ۹ / شبیه‌سازی سیستم‌وریلاگ

۹-۱ شبیه‌سازی فعال شده با رخداد
۹-۲ شبیه‌سازی سیستم‌وریلاگ
۹-۳ رقابت‌ها
۴-۴-۳ Tasks (کارها)
۴-۵ چک‌کننده توازن
۴-۶ بافرهای سه حالته
۴-۶-۱ منطق چند مقداری
۴-۷ Testbench بلاک‌های ترکیبی

فصل ۵ / مدل‌های سیستم‌وریلاگ بلاک‌های منطقی ترتیبی

۵-۱ نگهدارنده‌ها(لچ‌ها)
۵-۱-۱ نگهدارنده SR
۵-۱-۲ نگهدارنده D
۵-۲ فلیپ فلاپ‌ها
۵-۲-۱ فلیپ فلاپ D تغییرکننده با لبه
۵-۲-۲ SET وRESET آسنکرون (نشاندن و بازنشاندن آسنکرون)
۵-۲-۳ Set و Reset سنکرون و فعال‌ساز کلاک
۵-۳ فلیپ فلاپ‌های JK و T
۵-۴ ثبات‌ها و ثبات‌های انتقالی
۵-۴-۱ ثبات چندبیتی
۵-۴-۲ ثبات‌های انتقالی (شیفت رجیسترها)
۵-۵ شمارنده‌ها
۵-۵-۱ شمارنده باینری
۵-۵-۲ شمارنده جانسون
۵-۵-۳ ثبات انتقال با فیدبک خطی
۵-۶ حافظه
۵-۶-۱ ROM
۵-۶-۲ SRAM
۵-۶-۳ RAM سنکرون
۵-۷ ضرب‌کننده ترتیبی
۵-۸ Testbench برای بلاک‌های ساختار ترتیبی
۵-۸-۱ تولید کلاک
۵-۸-۲ Resetها و سایر سیگنال‌های قطعی
۵-۸-۳ پاسخ‌های بررسی

فصل ۶ / طراحی ترتیبی سنکرون

۶-۱ سیستم‌های ترتیبی سنکرون
۶-۲ مدل‌های سیستم‌های ترتیبی سنکرون
۶-۲-۱ ماشین‌های مور و میلی

۱۱-۴-۱ شبیه‌سازی موازی خطا
۱۱-۴-۲ شبیه‌سازی همزمان خطا

فصل ۱۲ / طراحی برای قابلیت آزمون‌پذیر بودن

۱۲-۱ بهبود قابلیت آزمون پذیری تک منظوره
۱۲-۲ طراحی ساخت یافته برای آزمون
۱۲-۳ خودآزمایی درون ساخته شده
۱۲-۳-۱ مثال
۱۲-۳-۲ بررسی بلوک منطقی ساخته شده در داخل (BILBO)
۱۲-۴ اسکن مرزی (IEEE ۱۱۴۹/۱)

فصل ۱۳ / طراحی ترتیبی آسنکرون

۱۳-۱ مدارهای آسنکرون
۱۳-۲ تجزیه و تحلیل مدارهای آسنکرون
۱۳-۲-۱ تجزیه و تحلیل غیر رسمی
۱۳-۲-۲ تجزیه و تحلیل رسمی
۱۳-۳ طراحی مدارهای آسنکرون
۱۳-۴ ماشین‌های حالت آسنکرون
۱۳-۵ زمان‌های راه اندازی و نگهداری و ناپایداری
۱۳-۵-۱ محدودیت‌های مد اساسی و مدارهای سنکرون
۱۳-۵-۲ مدل‌سازی سیستم‌وریلاگ نقض زمان برپایی و نگهداری
۱۳-۵-۳ ناپایداری

فصل ۱۴ / مواجهه با دنیای آنالوگ

۱۴-۱ مبدل‌های دیجیتال به آنالوگ
۴-۲ مبدل‌های آنالوگ به دیجیتال
۱۴-۳ Verilog-AMS
۱۴-۳-۱ اصول وریلاگ_AMS
۱۴-۳-۲ دستورات کمکی
۱۴-۳-۳ مدلسازی سیگنال مختلط
۱۴-۴ حلقه‌های قفل فاز
۱۴-۵ شبیه‌سازهای AMS_وریلاگ
پیوست الف) پاسخ به سوالات انتخابی
۹-۳-۱ اجتناب از رقابت
۹-۴ مدل‌های تأخیر
۹-۵ ابزارهای شبیه‌سازی

فصل ۱۰ / سنتز سیستم‌وریلاگ

۱۰-۱ سنتز RTL
۱۰-۱-۱ سیستم‌وریلاگ سنتزناپذیر
۱۰-۱-۲ فلیپ فلاپ‌ها و نگهدارنده‌های استنتاج شده
۱۰-۱-۲-۱ نگهدارنده حساس به سطح
۱۰-۱-۲-۲ فلیپ فلاپ حساس به لبه
۱۰-۱-۳ منطق ترکیبی
۱۰-۱-۴ خلاصه‌ای از قوانین سنتز RTL
۱۰-۲ قیود
۱۰-۲-۱ صفات
۱۰-۲-۲ قیدهای مساحتی و ساختاری
۱۰-۲-۲-۱ کدگذاری حالت
۱۰-۲-۲-۲ قیدهای منبع
۱۰-۲-۲-۳- قیدهای زمانی
۱۰-۲-۳ صفات full_case و Parallel_case
۱۰-۳ سنتز FPGAها
۱۰-۴ سنتز رفتاری
۱۰-۵ بازبینی نتایج سنتز
۱۰-۵-۱- شبیه‌سازی زمان‌بندی

فصل ۱۱ / آزمون سیستم‌های دیجیتالی

۱۱-۱ ضرورت وجود آزمون
۱۱-۲ نمونه‌های خطا
۱۱-۲-۱ مدل خطای گیرکرده تکی
۱۱-۲-۲ خطاهای PLA
۱۱-۳ تولید الگوی آزمون مبتنی خطا
۱۱-۳-۱ الگوریتم مسیر حساس
۱۱-۳-۲ خطاهای غیرقابل تشخیص
۱۱-۳-۳ الگوریتم D
۱۱-۳-۴ PODEM
۱۱-۳-۵ از بین رفتن خطا
۱۱-۴ شبیه‌سازی خطا

تلفن: ۲-۶۶۴۸۱۹۱

فصل ۱

مقدمه

در این فصل با تأکید خاص روی طراحی سیستم‌های دیجیتال از طریق به کارگیری زبانهای توصیف سخت افزار[1] (HDLs) نظیر سیستم‌وریلاگ، مراحل طراحی را بررسی خواهیم کرد. نگاه مختصری به تکنولوژی CMOS (نیمه‌هادی اکسید فلزی مکمل[2]) مدارهای مجتمع خواهیم داشت و تکنولوژی‌های منطق قابل برنامه‌ریزی مورد بحث قرار می‌گیرند. در پایان خواص الکتریکی مربوط به CMOS و منطق قابل برنامه‌ریزی بررسی خواهند شد.

۱-۱ طراحی دیجیتال نوین

طراحی مدارهای الکترونیکی به طور سنتی به دو بخش اصلی آنالوگ و دیجیتال تقسیم می‌شود. این دو موضوع معمولاً جداگانه تدریس می‌شوند و مهندس‌های الکترونیک معمولاً در یک بخش متخصص می‌شوند. در این دو بخش تخصص‌های بالاتری هم وجود دارند مانند: طراحی آنالوگ در فرکانس‌های رادیویی، طراحی مدارهای مجتمع دیجیتال و یا تلاقی دو دسته طراحی سیگنال آمیخته. علاوه بر این‌ها مهندسی نرم افزار هم نقش روز به روز مهم تری در سیستم‌های جاسازی شده پیدا می‌کند.

الکترونیک دیجیتال در کالاهای مصرفی روز به روز چشمگیرتر می‌شود. اتومبیل‌ها سیستم‌های کنترل پیچیده و سطح بالایی دارند. در خیلی از خانه‌ها کامپیوترهای شخصی وجود دارد. محصولاتی که به طور معمول آنالوگ در نظر گرفته می‌شدند، مانند رادیو، تلویزیون و تلفن‌ها، دیجیتالی شده‌اند یا در حال دیجیتالی شدن هستند. دیسک‌های فشرده دیجیتال و mp3ها تقریباً جایگزین سیستم‌های آنالوگ

1- Hardware Description Languages
2- Complementary metal oxide semiconductor

ضبط و پخش صدا گردیده‌اند. با این تغییرات، طول عمر محصولات کم شده و در کمتر از یک سال، مدل‌های جدیدی احتمالاً جایگزین محصولات الکترونیکی دیجیتال قبلی در فروشگاه محل شما شده‌اند.

۱-۲ طراحی با زبان‌های توصیف سخت افزار

۱-۲-۱ طراحی خودکار

برای همگامی با این تغییرات سریع، باید محصولات الکترونیکی خیلی سریع طراحی شوند. طراحی آنالوگ هنوز هم یک حرفه تخصصی با حقوق خوب است. طراحی دیجیتال بسیار وابسته به طراحی به کمک یک کامپیوتر[1] (CAD) شده که تحت عنوان طراحی خودکار[2] (DA) یا طراحی خودکار الکترونیک[3] (EDA) هم شناخته می‌شود. ابزارهای EDA اجازه انجام دو کار را می‌دهند: *سنتز*[4] یا ترجمه یک مشخصه در جهت اجرای واقعی طرح و دیگری *شبیه سازی*[5] که توسط آن مشخصه یا جزئیات اجرا را می‌توان مورد آزمایش قرار داد تا مشخص شود که آیا درست کار می‌کند یا نه.

ابزارهای سنتز و شبیه‌سازی EDA خودشان مستلزم انتقال طرح از تصورات ذهن طراح به ابزارها می‌باشند. این کار را با ترسیم نموداری از طرح به کمک یک نرم افزار گرافیکی می‌توان انجام داد. این نرم افزار به گیرنده شماتیک[6] معروف است. روش دیگر ارائه طرح به صورت متنی مانند یک برنامه نرم افزاری است. توصیف متنی سخت افزار دیجیتالی را می‌توان به زبان برنامه‌ریزی اصلاح شده‌ای مانند C و یا یک زبان توصیف سخت افزار (HDL) نوشت. در طی سی سال گذشته تعداد زیادی HDL طراحی شده‌اند. امروزه دو تا از این HDLها پرکاربرد هستند. Verilog و VHDL. HDLهای استاندارد مهم هستند چرا که می‌توانند مورد استفاده ابزارهای CAD گوناگون از شرکت‌های گوناگون قرار گیرند. در دوران پیش از Verilog و VHDL هر ابزاری HDL خاص خودش را داشت که کار ترجمه بین HDLها دشوار و وقت گیر بود، این کار مثلاً موقع تائید خروجی‌های ابزار سنتز شرکتی با شبیه‌ساز شرکت دیگر لازم بود.

۱-۲-۲ SystemVerilog چیست؟

سیستم‌وریلاگ یک زبان توصیف سخت افزار است. در بسیاری از موارد، یک HDL همانند یک زبان برنامه نویسی در نظر گرفته می‌شود، اما HDLها مشخصه‌های بسیاری دارند که یک زبان برنامه نویسی نظیر C آن‌ها را ندارد.

1- Computer-Aided Design
2- Design Automation
3- Electronic Design Automation
4- Synthesis
5- Simulation
6- Schematic capture

نشر نوآور
تلفن: ۲-۶۶۴۱۴۲۹۱

Verilog ابتدا در اوایل سال 1980 توسعه یافت که مبتنی بر زبان (و شبیه ساز) Hilo-2 (متعلق به دانشگاه Brunel) بود. اولین کمپانی که Verilog را توسعه داد، کمپانی Cadence بود. در اوایل سال ۱۹۹۰، Cadence آن را بعنوان یک زبان عمومی معرفی و در سال ۱۹۹۵، استاندارد IEEE گردید (استاندارد ۱۳۶۴). در سال ۲۰۰۱ نسخه جدید استاندارد آن پذیرفته شد که شامل ویژگی‌های بیشتری بود و سرانجام در سال ۲۰۰۵ ویرایش‌های جزئی آن پذیرفته شد. کار ادامه یافت تا Verilog به مدل سطح سیستمی گسترش یافت. این زبان جدید به عنوان SystemVerilog (سیستم‌وریلاگ) شناخته شده که آخرین ورژن آن 3.1a میباشد (در سال ۱۹۹۵، ورژن Verilog برابر ۱ و در سال ۲۰۰۱ ورژن آن 2.0 بود). زبان سیستم‌وریلاگ در سال ۲۰۰۵ استاندارد IEEE-1800 شد.

زبان Verilog توسعه یافت و امکان مدل کردن مدارهای آنالوگ (Verilog-A) و مدل نمودن سیگنال آمیخته[1] (مختلط) را به وجود آورد (Verilog-AMS).

۱-۲-۳ VHDL چیست؟

در همان دوره، زبان توصیف سخت افزار دیگری- HDL-VHSIC[2](مدارات مجتمع با سرعت بسیار بالا) یا VHDL- توسط سازمان دفاعی آمریکا ایجاد شد و تحت عنوان استاندارد ۱۰۷۶بوسیله IEEE استاندارد گردید. ۴ نسخه IEEE 1076 در سال‌های ۱۹۸۷، ۱۹۹۳، ۲۰۰۲ و ۲۰۰۸ وجود دارد. زبان‌های توصیف سخت افزار دیگری نظیر ELLA و UDL/I وجود داشته‌اند اما امروزه Verilog و VHDL رایج تر هستند. هر زبانی موافقان و مخالفان خود را دارد. بطور واقعی (اگر امکان دید بدون غرض وجود داشته باشد)، هر دو زبان نقص داشته و تلاش برای یافتن بهترین بی‌فایده است.

۱-۲-٤ شبیه سازی[3]

با این وجود، HDLهای دیگر قابل ملاحظه هستند. زبان SystemC از ویژگی‌های زبان C++ برای مدل کردن سخت افزار استفاده می‌کند. در حال حاضر نمی‌توان تخمین زد که آیا SystemC می‌تواند جایگزین سیستم‌وریلاگ یا VHDL باشد یا نه. با این وجود، شایان ذکر است که بسیاری از راهبردهای مدل طراحی به این ۳ زبان رجوع میکنند.

زبان‌های HDL دارای ۳ ویژگی هستند که بسیار کم در زبان‌های برنامه نویسی وجود دارد: (۱) همزمانی[4]، (۲) نمایش زمان، و (۳) نمایش ساختار.

سخت افزار فی نفسه بطور همزمان و موازی است. بنابراین زبان HDL میبایست قادر باشد که

1- Mixed-signal
2- Very High Speed Integrated Circuit
3- Simulation
4- Concurrency

عملیاتی که بصورت همزمان اتفاق می‌افتند را توصیف نماید. بعنوان مثال زبان C (که بسیار مورد استفاده است) یک زبان ترتیبی است.

انجام عملیات در سخت افزار زمان محدودی را طی میکند. لذا مکانیزمی نیاز است که گذر زمان را توصیف نماید.

معماری سخت افزار بسیار مهم است. یک زبان C شامل توابعی است که فرا خوانده میشوند و وظایف خود را انجام می‌دهند اما دیگر، حالت‌های درونی را حفظ نمی‌کنند. از طرفی دیگر، گیت‌ها یا معماری‌های سخت افزاری دیگر مقدار داشته و حتی اگر کاری را انجام ندهند حالت خود را حفظ میکنند.

SystemC این ویژگی‌ها را در زبانی شبیه به C فراهم نموده است. سیستم‌وریلاگ (و VHDL) این ویژگی‌ها را درون خود دارند.

همزمانی، زمان و معماری از تفاوت‌های عمده بین زبان‌های HDL و زبان‌های برنامه نویسی هستند. یک زبان C میتواند کامپایل شده و بروی یک PC یا ایستگاه کاری اجرا شود. سیستم‌وریلاگ می‌تواند کامپایل شود، اما برای اجرا شدن نیازمند یک شبیه ساز است. شبیه ساز، عملیات فعل و انفعالی بین عناصر همزمان و مدل‌های گذار زمانی را مدیریت می‌کند. شبیه ساز همچنین حالت هر عنصر ساختاری را حفظ می‌کند. تعدادی شبیه ساز برای سیستم‌وریلاگ وجود دارد.

حامیان Verilog اغلب بر این باورند که یادگیری این زبان ساده تر از VHDL است. به این دلیل قابل بحث و مناظره است. VHDL مدل‌های شبیه‌سازی خوش تعریفی دارد. دو شبیه ساز متفاوت برای VHDL (اغلب) تضمین می‌کنند که شبیه سازی‌های دقیقا یکسانی را تولید نمایند. مدل‌های شبیه‌سازی سیستم‌وریلاگ بسیار ضعیف تر تعریف شده‌اند. اگر خیلی دقیق باشید، دو شبیه ساز سیستم‌وریلاگ ممکن است شبیه سازی‌های متفاوتی تولید کنند. هدف این کتاب این است که نشان دهد چگونه مدل‌های سخت افزار نوشته شود تا با رفتار قابل پیش‌بینی، شبیه‌سازی و سنتز گردد. به این دلیل، این کتاب تلاشی برای بیان همه جزئیات زبان سیستم‌وریلاگ ندارد.

۱-۲-۵ سنتز[1]

سیستم‌وریلاگ یک زبان *توصیف* سخت افزار است نه یک زبان *طراحی* سخت افزار. در سال ۱۹۸۰ شبیه‌سازی دیجیتالی یک تکنولوژی تکامل یافته بود اما سنتز اتوماتیک سخت افزار نه (این استدلال شامل VHDL نیز میشود). ممکن است در سیستم‌وریلاگ مدل‌هایی نوشته شوند که نتوان آنهارا بصورت سخت افزار فیزیکی قابل درک مشابه نمود. فقط یک زیر مجموعه از دستورات زبان با استفاده از ابزارهای سنتز سطح انتقال ثبات[2] (RTL)، میتواند سنتز شود. علاوه بر این ابزارهای سنتز RTL بوسیله تشخیص الگوهای ویژه در کد برنامه کار کرده و از آن الگوها برای استنباط وجود ثبات‌ها استفاده می‌کند

1- Synthesis
2- Register Transfer Level

(ابزارهای سنتز RTL همچنین مدارهای منطقی ترکیبی را تا حدودی بهینه میکنند). بنابراین شیوه برنامه نویسی در سیستم‌وریلاگ مهم است. یک استاندارد IEEE برای سنتز RTL –1364.1– در سال ۲۰۰۲ پذیرفته شد. این استاندارد، یک زیرمجموعه از دستورات Verilog و مفاهیم این زیرمجموعه را برای نسخه Verilog سال ۲۰۰۱، تعریف نمود.

مدل‌های سخت افزاری در این کتاب مطابق با استاندارد 1364.1 سال ۲۰۰۲ برای زبان سیستم‌وریلاگ خواهد بود. به عبارت دیگر، از مدل استاندارد سنتز استفاده خواهد شد.

۱-۲-۶ استفاده مجدد[1]

در حال حاضر در صنایع الکترونیکی، ایده استفاده مجدد، بطور گسترده ای، رایج است. مدارات مجتمع چنان بزرگ و پیچیده هستند که برای یک تیم تقریباً غیر ممکن است که طرحی را از روی یک چرکنویس بسازند. در عوض این انتظار می‌رود که در بیشتر طراحی‌ها، بخش‌هایی از طراحی‌های پروژه‌های قدیمی تر و یا ایجاد شده بوسیله کمپانی‌ها را مجددا استفاده کنند. روشن است که اگر یک طرح بتواند مورد استفاده مجدد قرار بگیرد، آن طرح تطبیق‌پذیر خواهد بود. طبیعی است که هر کسی بخواهد می‌تواند از این طرح استفاده نموده یا آنرا طوری تطبیق دهد که برای طراحی‌های مختلف مورد استفاده قرار گیرد.

در سطح ساده، فرض کنید می‌خواهید یک ضرب کننده ۴ بیتی طراحی کنید. این کار با تنظیم عرض ورودی‌ها و خروجی‌ها در مقدار ۴ انجام میپذیرد. همچنین نیاز است که عرض برخی از ثبات‌های داخلی را تنظیم نمایید. بعد از گذشت زمانی، به فرض شما می‌خواهید یک ضرب کننده ۱۳ بیتی طراحی کنید. در سطح عملیاتی (یا RTL برای طرح سنتز پذیر)، تفاوت بین دو طراحی، تعداد عرض ورودی، خروجی و ثبات می‌باشد. هر دو طرح قدیمی و جدید نیاز به شبیه‌سازی و سنتز دارند. این احتمال وجود دارد که در تغییر طراحی از ۴ به ۱۳ بیتی دچار اشتباه شوید. این اتفاق نیاز به تلاش زیادی جهت اشکال زدایی دارد. فرض کنید شما از ابتدا یک ضرب کننده n-بیتی طراحی می‌کردید. این برنامه فقط یک بار اشکال زدایی خواهد شد. زمانیکه میخواستید یک ضرب کننده ۱۳ بیتی تولید کنید کافی است که به سادگی پارامتر را برابر ۱۳ قرار دهید. ایده تولید طراحی‌های پارامترپذیر بسیار جالب است. ما تا جایی که امکان دارد، سازه‌های قابل استفاده مجدد پارامتر پذیر، طراحی می‌کنیم.

ما همچنین نشان خواهیم داد که چگونه مدل‌هایی را بنویسیم که در شبیه سازهای مختلف رفتار یکسانی داشته باشند و با ابزارهای سنتز مختلف نیز، نتیجه سنتز یکسانی داشته باشند. به علاوه باید اطمینان کسب کنیم که رفتار مدل قبل و بعد از سنتز یکسان است.

تلفن: ۶۶۴۸۴۱۹۱-۱۰

1- Reusability