مجله علمی- پژوهشی "پدافند الکترونیکی و سایبری"
سال پنجم، شماره ۱، بهار ۱۳۹۶، ص ۳۵-۲۷

طراحی وپیاده‌سازی کارآمد فیلتر دیجیتال وفقی LMS بر روی تراشه FPGA

احسان صابری[۱]، مسعود معصومی[۲*]

۱- کارشناس ارشد الکترونیک، ۲- استادیار، دانشکده فنی، دانشگاه آزاد اسلامی واحد اسلامشهر

(دریافت: ۹۴/۱۱/۲۲، پذیرش: ۹۵/۰۸/۱۰)

چکیده

فیلترهای وفقی بخش مهمی در بسیاری از سامانه‌های پردازش سیگنال دیجیتال هستند و در کاربردهای متنوعی از جمله حذف اکو، حذف نویز، سیستم‌های رادار، سونار و ... مورد استفاده قرار می‌گیرند. تحقق سخت‌افزاری سیستم‌های پردازش سیگنال دارای مزایایی از قبیل سرعت و بازدهی بالاتر، امکان مجتمع‌سازی و قابلیت پردازش موازی در مقایسه با تحقق نرم‌افزاری آن می‌باشد. امروزه تراشه‌های FPGA به‌دلیل دارا بودن ویژگی‌هایی از قبیل پردازش موازی اطلاعات، انعطاف معماری ... به طور عمده برای تحقق سخت‌افزاری سیستم‌های دیجیتال مورد استفاده قرار می‌گیرند. پیاده‌سازی کارآمد فیلترهای وفقی بر روی تراشه‌های FPGA امری مهم و در عین حال چالش برانگیز است زیرا این فیلترها بر خلاف فیلترهای غیرتطبیقی نیازمند تکرار محاسبات برای رسیدن به وزن‌های بهینه هستند. در این مقاله یک تحقق سخت‌افزاری کارآمد الگوریتم حداقل میانگین مربعات موسوم به LMS ارائه شده که در مقایسه با پیاده‌سازی گزارش‌شده در ادبیات مربوطه، دارای فرکانس کاری بالاتر و سطح اشغالی کمتر بر روی تراشه می‌باشد. صحت نتایج به‌دست‌آمده از طریق مقایسه نتایج پیاده‌سازی با نتایج به‌دست‌آمده از شبیه‌سازی یک فیلتر تطبیقی LMS حذف نویز تصدیق شده است. از آنجا که جمع‌آوری و پردازش دائمی اطلاعات و علائم محیطی و معنی دار از جمله ارکان مهم در چرخه مدیریت و جلوگیری از بحران از قبیل سامانه‌های هشدار دهنده و نیز پدافند غیرعامل می‌باشد لذا طراحی ارائه‌شده، می‌تواند به‌خوبی در ابزار و ادوات سخت‌افزاری مرتبط با این مقوله به‌کار گرفته شود.

واژه‌های کلیدی: شفافیت، فیلتر تطبیقی، الگوریتم LMS، تحقق سخت‌افزاری، تراشه برنامه‌پذیر FPGA

۱- مقدمه

مطابق با تعریف، پدافند غیر عامل مجموعه اقدامات بنیادی و زیربنایی است که در صورت به‌کارگیری می‌توان به اهدافی از قبیل تقلیل خسارات و صدمات، کاهش قابلیت و توانایی سامانه شناسائی دشمن، کاهش دقت هدف‌گیری تسلیحات آفندی دشمن و تحمیل هزینه بیشتر به وی نائل گردید. این مجموعه اقدامات عمدتا مستلزم به‌کارگیری جنگ‌افراز نبوده و با اجرای آن می‌توان از وارد شدن خسارات مالی به تجهیزات و تاسیسات حیاتی و حساس نظامی و غیرنظامی و تلفات انسانی جلوگیری نموده و یا میزان این خسارات و تلفات را به حداقل ممکن کاهش داد. جمع‌آوری و پردازش دائمی اطلاعات و علائم محیطی و معنی‌دار از جمله ارکان مهم در جلوگیری از بحران و نیز پدافند غیرعامل با اهداف اشاره شده می‌باشد. سامانه‌های پردازش سیگنال دیجیتال بدلیل دارا بودن مزایایی چون کیفیت بالاتر، انعطاف‌پذیری بیشتر، قابلیت فشرده‌سازی و رمز شدن و ... در مقایسه با سامانه‌های آنالوگ بسیار مورد توجه مهندسین، محققین و صاحب‌نظران مهندسی برق هستند و از جمله پایه‌های مهم در علم مدرن امروز ارتباطات بشمار می‌روند. به‌کارگیری کارآمد سامانه‌هایی مبتنی بر چنین پردازش‌هایی باعث افزایش چشمگیر قابلیت آفند و هم پدافند خواهد شد. یکی از مهمترین ساختارها در انواع پردازش‌های گسسته، فیلترهای دیجیتال هستند [۱]. از میان این نوع فیلترها، فیلتر دیجیتال وفقی بسیار موردتوجه قرار گرفته‌اند. این نوع فیلترها برای کاربردهایی مناسب است که پارامترهای آماری سیگنال ورودی آن در طول زمان تغییر می‌کند یا این‌که سیگنال ورودی دارای ویژگی‌های غیرخطی باشد [۲]. به عنوان نمونه از این فیلترها در سامانه‌های مخابراتی که از کانال‌های محوشونده[۲] برای ارتباط استفاده می‌کنند برای متعادل کردن کانال و افزایش کیفیت ارتباط به‌طور گسترده‌ای استفاده می‌گردد. همچنین می‌توان به کاربردهایی نظیر مدل‌سازی سامانه، حذف اکو و حذف نویز در سیگنال‌هایی با فرکانس پایین مانند نویزهای نوار قلب اشاره کرد [۳]. در مباحث

* رایانامه نویسنده مسئول: m_mamsoumi@iiau.ac.ir

2- Fading

نظامی همچون رادار و سونار که شناسایی یک سامانه اهمیت ویژه‌ای دارد با کمک فیلتر تطبیقی می توان تابع انتقال سامانه ناشناخته را با استفاده از ورودی و خروجی‌های آن تخمین زد و به‌دست آورد [۴]. علاوه بر آن از فیلترها و سامانه‌های پردازش سیگنال دیجیتال به‌طور گسترده‌ای در سامانه‌های هشداردهنده اولیه[1] که از جمله ابزارهای مهم دفاع غیرنظامی می‌باشند استفاده می‌شود.

ویژگی مهم فیلترهای تطبیقی این است که می‌توانند تابع انتقال خود را به‌منظور رسیدن به بهترین عملکرد تغییر داده و به‌هنگام سازند. روند انطباق مبتنی بر استفاده از یک تابع هزینه است که به کمک الگوریتم وفقی در یک فرآیند تکراری این تابع هزینه به حداقل مقدار ممکن می‌رسد [۵]. فیلترهای وفقی را می‌توان به‌وسیله فیلترهای FIR و یا IIR پیاده‌سازی کرد ولی با توجه به عدم وجود فیدبک در فیلترهای FIR، این نوع فیلتر تمام قطب و ذاتا پایدار بوده و از این رو پیاده‌سازی سخت‌افزاری این نوع فیلترها متداول‌تر از فیلترهای IIR است [۶]. برای پیاده‌سازی سخت‌افزاری این فیلتر همانند دیگر الگوریتم‌های پردازش سیگنال‌های دیجیتال، با محدودیت‌هایی از نظر سطح اشغالی روی تراشه، توان تلفاتی، سرعت و هزینه مواجه هستیم. با وجود آن‌که پردازشگرهای سیگنال دیجیتال[2] دارای قدرت بالایی در پردازش و انجام عملیات‌های پیچیده ریاضی هستند ولی به دلیل آن‌که پردازش اطلاعات در آنها به صورت متوالی انجام می‌گیرد نرخ نمونه‌برداری پایین‌تری در مقایسه با تراشه‌های ASIC[3] یا FPGA دارند. تراشه‌های ASIC با آن‌که به‌طور اختصاصی برای یک کاربرد خاص طراحی می‌شوند و با توجه به این ویژگی سرعت بالایی دارند ولی معماری غیرمنعطف و نیز طولانی بودن فرایند ساخت این تراشه‌ها عاملی است که انگیزه را برای استفاده از آنها به منظور پیاده‌سازی الگوریتم‌های پردازش سیگنال کاهش می‌دهد. تراشه‌های FPGA معایب ذکر شده این دو نوع تراشه را نداشته و قابلیت دریافت اطلاعات و پردازش آنها به صورت موازی را دارا بوده به‌دلیل انعطاف‌پذیری بالای معماری آن، سرعت انجام محاسبات در آنها بالاست [۷]. در این مقاله پیاده‌سازی سخت‌افزاری یک فیلتر دیجیتال وفقی جهت حذف نویز از یک سیگنال را ارائه می‌دهیم که این کار روی تراشه‌هایی از خانواده Spartan3 و Virtex محصول شرکت Xilinx، به کمک نرم‌افزار ISE انجام شده است. از نرم‌افزار MATLAB نیز جهت یافتن طول کلمه مناسب به‌منظور استفاده از اعداد در حالت ممیز ثابت بهره گرفته شده است. نتایج به‌دست‌آمده از پیاده‌سازی نشان‌دهنده کارآمد بودن این پیاده‌سازی در مقایسه با آخرین کارهای گزارش‌شده مشابه در منابع معتبر مربوطه است. بخش‌های مختلف این مقاله بدین شرح است: بخش دوم آشنایی مختصر با فیلتر وفقی و الگوریتم LMS، بخش سوم تشریح پیاده‌سازی سخت‌افزاری فیلتر، بخش چهارم بیان نتایج و مقایسه با کارهای انجام شده در این رابطه و در بخش پنجم نتیجه‌گیری نهایی از انجام این مقاله انجام شده است.

۲- مختصری در مورد فیلتر وفقی و الگوریتم LMS

فیلتر وفقی یا به عبارت دیگر فیلتر سازگار نوعی از فیلتر دیجیتال است که مقادیر ضرایب خود یا به‌عبارت دیگر وزن فیلتر را با توجه به تغییر آمار و شرایط محیطی اصلاح نموده و انطباق دهد. آنچه که در فرآیند انجام فیلتر رخ می‌دهد، تغییر دامنه ضرایب می‌باشد که این عمل توسط الگوریتم‌های تطبیقی صورت می‌پذیرد. بر اساس نظریه Wiener–Hopf، با یک رویکرد احتمالی می‌توان تخمین زد که وزن‌های بهینه برای فیلتر زمانی به‌دست می‌آیند که میانگین مربع خطا به حداقل مقدار خود برسد. در این حالت فیلتر همگرا شده است. زمان روند تطبیق به اندازه گام‌های حرکت به سمت نقطه بهینه قابل تنظیم است. اندازه گام کوچک باعث افزایش دقت و کاهش خطا می شود و در عین حال سرعت اجرای الگوریتم را نیز کاهش می‌دهد. انتخاب اندازه گام بزرگ در حالی که سرعت اجرای الگوریتم را زیاد می کند، به همان نسبت نیز خطای همگرایی را افزایش خواهد داد. از این‌رو انتخاب اندازه گام مناسب در فیلترهای سازگار امری مهم و اساسی است [۸]. با توجه به همین نظریه الگوریتم حداقل میانگین مربعات شکل گرفت که بر اساس آن با توجه به خصوصیات آماری داده های ورودی اقدام به یافتن نقطه بهینه پیدا شده و نهایتا وزن‌های فیلتر اصلاح می‌گردد. این الگوریتم یکی از کلی‌ترین و اساسی‌ترین روش‌های اصلاح وزن است که به‌دلیل سادگی در مفهوم و اجرا کاربرد بسیار زیادی در شاخه‌های گوناگون دیگر از جمله الگوریتم‌های اصلاح وزن‌ها در شبکه‌های عصبی نیز دارد.

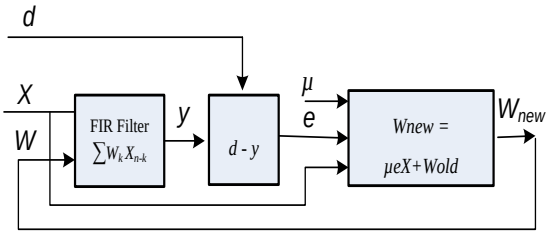
۲-۱- مدل ریاضی و روابط لازم برای فیلتر وفقی

عمدتا فیلترهای تطبیقی دارای دو بلوک سازنده اساسی هستند؛ ساختار فیلتر و نیز الگوریتم تعیین ضرایب به شرح زیر می‌باشد: ۲-۱-

۲-۱-۲- الگوریتم تطبیقی: وجود این الگوریتم برای تنظیم ضرایب فیلتر وفقی لازم است تا خروجی y(n) و سیگنال مطلوب d(n) با یکدیگر برابر شوند. سیگنال خطای e(n) از تفاضل دو سیگنال خروجی و مطلوب به‌دست می‌آید یعنی d(n) – y(n). از این سیگنال جهت تنظیم ضرایب سیستم وفقی

1- Early Warning Systems
2- Digital Signal Processors (DSPs)
3- Application Specific Integrated Circuits

در لحظات بعد استفاده می‌شود [۲]. شکل (۱) دو بلوک مزبور را نشان می دهد.



شکل (۱). بلوک دیاگرام فیلتر وفقی.

۲-۲- الگوریتم LMS

الگوریتم حداقل میانگین مربعات[1]، توسط Hoff Widrow و در سال ۱۹۵۹ معرفی گردید که یک الگوریتم تطبیقی بوده و مبتنی بر روش گرادیان است. الگوریتم LMS با استفاده از روش تخمینی از بردار گرادیان داده ها عمل می‌کند. LMS روشی است که تکرار و اصلاح مداوم بردار وزن در جهت منفی بردار گرادیان، نهایتاً منجر به حداقل رسیدن میانگین مربعات خطا می‌شود. این الگوریتم نسبت به دیگر الگوریتم‌ها ساده تر بوده و برای محاسبه ماتریس وزن نیاز به محاسبه تابع همبستگی و ماتریس معکوس ندارد [۹]. بر طبق روش Steepest Descent بردار وزن در این الگوریتم از رابطه زیر به‌دست می‌آید.

$$w(n+1) = w(n) + \frac{1}{2}\mu[-\nabla(E\{e2(n)\})] \qquad (۱)$$

و مبین آن است که بردار ضریب آینده w(n+1) حاصل جمع بردار حاضر w(n) و یک مقدار متغیر متناسب با منفی گرادیان خطاست. در این رابطه μ پارامتر اندازه گام است و مشخصات همگرایی الگوریتم LMS را کنترل می‌کند. $e(n)^2$ میانگین مربعات خطای بین خروجی y(n) و سیگنال مرجع که از رابطه (۲) به‌دست می‌آید [۱۰].

$$e(n)^2 = [d^*(n) - w^h\, x(n)]^2 \qquad (۲)$$

برای به‌هنگام‌سازی گرادیان بردار وزن، می‌توان از رابطه زیر استفاده کرد.

$$\nabla_w(E\{e^2(n)\}) = -2r + 2Rw(n) \qquad (۳)$$

البته در روش Steepest Descent به‌دست آوردن مقادیر ماتریس‌های R و r در زمان واقعی کار مشکلی است. به همین دلیل به‌جای آن از ماتریس کوواریانس r و R به‌جای ارزش‌های واقعی خود در الگوریتم LMS استفاده می‌شود. روابط زیر بیانگر آن است [۱۰].

$$R(n) = x(n).x^h(n) \quad \text{و} \quad r(n) = d^*(n).x(n) \qquad (۴)$$

پس رابطه (۱) را می‌توان بصورت رابطه (۵) بازنویسی کرد.

$$w(n+1) = w(n) + \mu x(n)\, e^*(n) \qquad (۵)$$

الگوریتم LMS به ازاء n=0 و محاسبه بردار وزن w(0) شروع می شود و با انجام محاسبات متوالی و اصلاحات لازم به کمترین میانگین مربع خطا دست می یابد. روابط ریاضی نهایی الگوریتم LMS را می‌توان در شکل (۲) مشاهده نمود [۱۰].

$$Output, y(n) = W^h x(n)$$
$$Error, e(n) = d^*(n) - y(n)$$
$$Weight, W(n+1) = w(n) + \mu x(n) e^*(n)$$

شکل (۲). روابط ریاضی الگوریتم LMS.

۳-۲- همگرایی و پایداری الگوریتم LMS

الگوریتم LMS با مقادیر دلخواهی از بردار وزن شروع می‌شود و جهت پایدار شدن و رسیدن به یک همگرایی باید رابطه (۶) برقرار باشد.

$$0 < \mu < \lambda max \qquad (۶)$$

که λ_{max} بزرگترین مقدار ویژه ماتریس همبستگی R است. همگرایی الگوریتم نسبت عکس با گسترده شدن مقادیر ویژه ماتریس همبستگی R دارد. وقتی که مقادیر ویژه R خیلی گسترده باشند همگرایی به آهستگی انجام می‌گیرد. اگر μ خیلی کوچک انتخاب شود همگرایی به آهستگی رخ می‌دهد. و چنانچه مقدار μ را بزرگ انتخاب کنیم موجب تسریع روند همگرایی ولی در عین حال کاهش پایداری خواهد شد [۱۰].

۳- پیاده‌سازی سخت‌افزاری فیلتر

برای پیاده‌سازی سخت‌افزاری فیلتر از دو روش کد نویسی با زبان Verilog و همچنین نرم‌افزار System Generator که از نرم-افزارهای الحاقی به نرم‌افزار ISE است و برای سنتز بلوک‌های پردازش سیگنال مورد استفاده قرار می‌گیرد استفاده شده است که به ترتیب شرح داده خواهد شد.

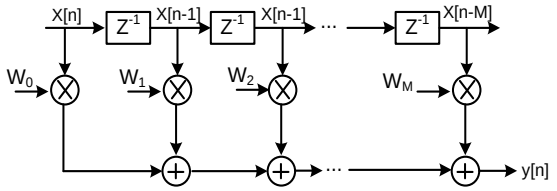
1- Least Mean Square

۳-۱- پیاده‌سازی با استفاده از زبان توصیف سخت‌افزار Verilog

در این مرحله بلوک‌های مورد نیاز که توسط زبان توصیف سخت‌افزاری Verilog نوشته و پس ازسنتز شده بود و نیز شبیه‌سازی نتایج مورد بررسی قرار می‌گیرد. در ادامه با ارائه توضیح مختصری در مورد فیلتر FIR ساختار مستقیم نحوه پیاده‌سازی به این روش شرح داده خواهد شد. لازم به‌ذکر است به‌منظور حفظ کارآمدی بیشتر پیاده‌سازی، کدها به‌صورت ساختاری نوشته شدند. همان‌طور که می‌دانیم پیاده‌سازی ساختاری غالباً در مقایسه با پیاده‌سازی رفتاری ازنظر مساحت اشغالی روی تراشه، توان مصرفی و نیز سرعت به‌مراتب بهینه‌تر است.

۳-۱-۱- فیلتر FIR با ساختار مستقیم

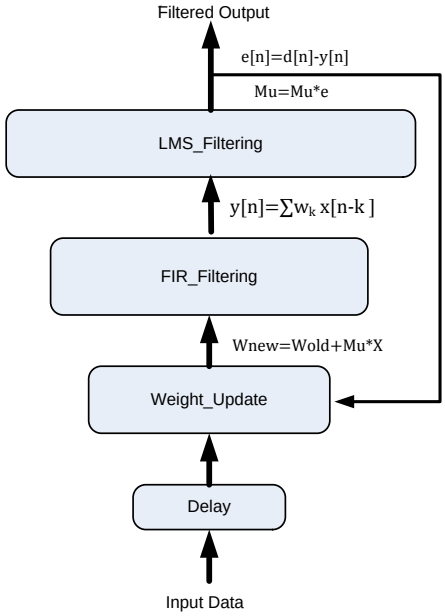
قلب مدار یک فیلتر FIR است که باید رابطه $y_{FIR}[n]=\sum_{k=0}^{M} w_k . x[n-k]$ را پیاده‌سازی نماید. از این نوع فیلتر به این دلیل استفاده شده است که اولا در ساختار آن فیدبک وجود ندارد و در نتیجه همیشه پایدار بوده و ثانیا نداشتن فیدبک اثر نویز و خطای کوانتیزاسیون را کم می‌نماید. از مستقیم ساختار فیلتر FIR جهت پیاده‌سازی در این بخش مورد استفاده قرار گرفته است. در شکل (۳) بلوک دیاگرام این ساختار مشاهده می‌گردد. برای طراحی مدار مذکور به این صورت عمل شده که هر ورودی x به ترتیب در ضرایب w ضرب شده و حاصل به خروجی منتقل می‌شود و همچنین وجود یک بلوک تاخیر نیز ضروری می‌باشد.



شکل (۳). بلوک دیاگرام فیلتر FIR همراه با ضرایب مربوطه.

۳-۱-۲- طراحی سخت‌افزاری فیلتر

هدف کلی در این قسمت پیاده‌سازی سخت‌افزاری روابط ریاضی بیان شده در شکل (۲) می‌باشد. مدار مورد نظر در این تحقیق یک فیلتر درجه ۹ بوده و نحوه پیاده‌سازی آن به صورت پایین به بالا[1] می‌باشد به این معنی که از پایین‌ترین سطح، شروع به طراحی بلوک‌ها نموده و سپس با اتصال آنها به یکدیگر مدار نهایی طراحی می‌گردد. بلوک‌های طراحی‌شده شامل بلوک تاخیر، بلوک به‌هنگام‌سازی وزن، بلوک محاسبه خروجی فیلتر FIR و سرانجام بلوک خروجی فیلتر وقفی می‌باشد. شبیه‌سازی و بررسی نتایج از همان پایه ترین بلوک انجام گرفته و پس از تایید صحت عملکرد در بلوک‌های سطح بالاتر مورد استفاده قرار می‌گیرد. در شکل (۴) بلوک دیاگرام طراحی این فیلتر را مشاهده می‌کنید. در ادامه برای هر بلوک توضیحات مختصری بیان می‌گردد.



شکل (۴). بلوک دیاگرام پیاده‌سازی فیلتر طراحی‌شده در این مقاله.

۳-۱-۲-۱- تاخیر[2]

از مهمترین چالش‌ها در جهت کاهش سطح اشغالی تراشه، کم نمودن تعداد ثبات‌ها جهت ذخیره و نگه‌داشتن نتایج عملیات ریاضی می‌باشد. با توجه به تعداد زیاد محاسبات ریاضی و نیاز متناسب با آن برای ثبت اطلاعات، در صورت عدم استفاده از روش مناسب به‌منظور کنترل این ثبات‌ها سطح اشغالی روی تراشه افزایش می‌یابد. در ادامه روش کاهش ثبات‌ها در این مقاله توضیح داده خواهد شد. زیر بلوک پایه مدار بلوک تاخیر می‌باشد که وظیفه آن علاوه بر ایجاد تاخیر در موارد مورد لزوم، پذیرفتن نقش ثبات به منظور ذخیره نتایج میانی انجام محاسبات ضرب می‌باشد. عملیات ضرب با استفاده از مدارات ترکیبی انجام گرفته است بدین صورت که نیازی به پالس ساعت نداشته و در لحظه اعمال ورودی حاصل‌ضرب دو عدد پس از زمان بسیار ناچیز در یک سیم[3] قرار می‌گیرد و سپس به یک بلوک تاخیر (به عنوان ثبات) انتقال می‌یابد. این روش باعث می‌شود اولا چون برای محاسبه عملیات از پالس ساعت استفاده نمی‌گردد نتیجه

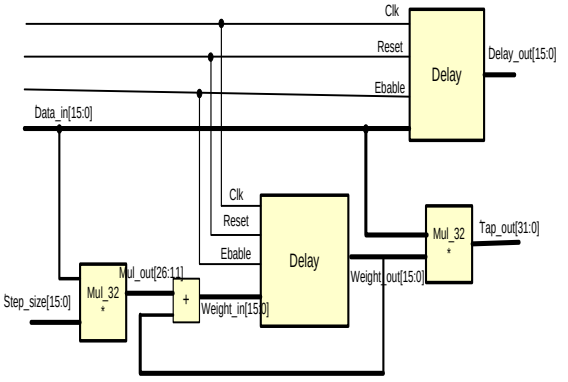
1- Bottom-Up

2- Delay
3- Wire

حاصل‌ضرب بلافاصله به‌دست آید و تاخیر را کاهش دهد و سرعت نهایی مدار را با توجه تعداد زیاد ضرب‌کننده‌ها در نهایت افزایش دهد (البته این مطلب در مورد جمع‌کننده‌ها و تفریق‌کننده مدار نیز صادق است) و ثانیا چون در مدارات ترکیبی حافظه وجود ندارد از بلوک تاخیر به عنوان ثبات استفاده می‌گردد که این امر باعث می شود به جای آن‌که یک ثبات به‌ازاء هر عملیات ریاضی نیاز باشد جواب نهایی آن در یک رجیستر ذخیره می‌گردد و این امر باعث کاهش مناسب سطح اشغالی تراشه می‌شود.

۳-۱-۲-۲- بلوک بهنگام سازی وزن

این بلوک دو کار مهم را انجام می‌دهد اول آن‌که حاصل‌ضرب y=w*x را محاسبه می‌کند که رابطه مذکور ضرب نویز ورودی در اولین ضریب فیلتر است و البته با توجه به آن‌که درجه فیلتر ۹ می‌باشد نیاز به همین تعداد ضرب برای محاسبه نهایی خروجی فیلتر FIR می‌باشد. ثانیا ضریب را به‌هنگام‌سازی می‌نماید که این فرآیند پس از انجام عملیات جمع و ضرب مورد نیاز برای ذخیره دیتای خروجی از یک بلوک تاخیر، که نقش ثبات را برعهده دارد، استفاده می‌کند به همین دلیل تمام آن با یک پالس ساعت انجام می‌پذیرد. شکل (۵) بلوک دیاگرام فرآیند به‌هنگام‌سازی ضرایب را نشان می‌دهد.



شکل (۵). بلوک دیاگرام فرآیند به‌هنگام‌سازی ضرایب.

۳-۱-۲-۳- محاسبه خروجی فیلتر FIR

در بلوک قبلی خروجی اصلی حاصل‌ضرب y=w*x است که برای پیاده کردن رابطه $y_{FIR}[n]=\sum_{k=0}^{M} w_k . x[n-k]$ به عنوان خروجی فیلتر FIR نیاز به استفاده از ۹ بلوک مذکور با توجه به درجه فیلتر می‌باشد. این بلوک عملا شکل (۳) را از نظر سخت‌افزاری پیاده می‌کند و در حقیقت خروجی آن کانولوشن $W*X$ می‌باشد.

۳-۱-۲-۴- خروجی نهایی فیلتر وفقی

در این بلوک به‌منظور هم‌زمان‌سازی انجام پردازش روی دیتاهای ورودی (نویز x و سیگنال آلوده به نویز d)، از دو بلوک تاخیر استفاده شده است. محاسبه سیگنال خطای به‌دست‌آمده به عنوان خروجی نهایی مدار در این بلوک صورت می‌پذیرد که در حقیقت تفاضل سیگنال آلوده به نویز و حاصل کانولوشن w*x محاسبه شده از بلوک قبل می‌باشد. همچنین با توجه به شکل (۴)، رابطه $W_{new} = W_{old} + \mu * x * e$ در بلوک به‌هنگام‌سازی وزن به‌طور کامل پیاده نمی‌شود $W_{new} = W_{old} + \mu * x * e$ یعنی سیگنال خطا e، در $\mu * x$ ضرب نمی‌گردد و در نتیجه برای پیاده‌سازی صحیح رابطه مذکور، سیگنال خطای محاسبه شده به‌ازاء هر دیتای ورودی در این بلوک به برنامه اضافه می‌شود و با توجه به ساختار موازی طراحی به طور هم‌زمان این مقدار به تمام ۹ بلوک جهت به‌هنگام‌سازی ضرایب اعمال می‌گردد.

۳-۲- پیاده‌سازی توسط System Generator

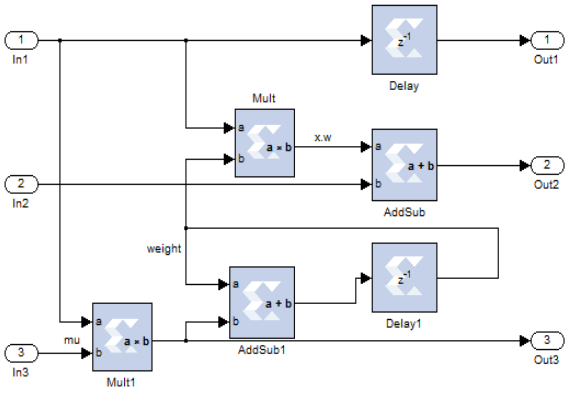
یکی دیگر از ابزارهای طراحی کارآمد شرکت Xilinx نرم‌افزار System Generator است که با توجه به مدل گرافیکی سامانه و الگوریتم توصیف شده در آن اقدام به تولید خودکار کد اجرایی می‌نماید. در ادامه روند این طراحی را برای فیلتر مورد نظر تشریح می‌کنیم.

۳-۲-۱- ایجاد زیرسامانه فیلتر

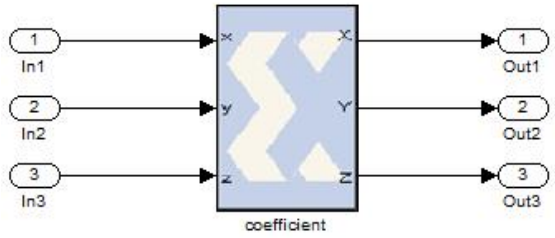
در انجام شبیه‌سازی با کمک این نرم‌افزار، نیاز به ایجاد یک زیر سامانه به‌منظور تسهیل در مراحل طراحی است. در حقیقت کاری که بایستی انجام گیرد ایجاد بلوکی است که بتواند موارد ذیل را به انجام برساند.

- سیگنال ورودی را با یک تاخیر انتقال دهد.
- ضریب اعمالی را اصلاح کند.
- رابطه $y[n]=\sum_{k=0}^{M} w_k . x[n-k]$ را پیاده‌سازی نماید.

برای انجام موارد فوق از بلوک شکل (۶) استفاده شده است. همان‌طور که از شکل پیداست این بلوک دارای ۳ ورودی و ۳ خروجی می باشد. ورودی ۱جهت انتقال ورودی به خروجی با یک تاخیر، ورودی ۲ جهت جمع حاصلضرب ورودی x در ضریب w و نهایتا ورودی ۳ به منظور اعمال اندازه گام جهت اصلاح وزن می‌باشد. خروجی این بلوک هم به ترتیب خروجی بلوک تاخیر، حاصل‌جمع به‌دست‌آمده از ضرب w و x و اعمال آن به بلوک بعدی و نهایتا اندازه گام اعمالی به طبقه بعدی می‌باشد. تمام مراحل ذکر شده فوق را می‌توان به کمک فرآیند ایجاد ماسک در نرم‌افزار MATLAB به صورت یک بلوک که در شکل (۷) نشان داده شده است می‌توان بیان نمود. در حقیقت این بلوک‌ها ضرایب فیلتر را تشکیل می‌دهند که با اتصال آنها مدار نهایی به‌دست می‌آید. همچنین برای انجام محاسبات تمام اعداد ورودی و خروجی در بلوک‌ها به صورت مکمل دو علامت‌دار می‌باشند.



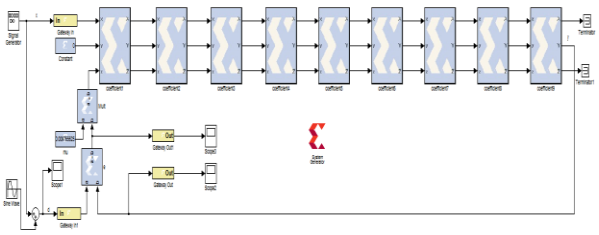
شکل (۶). زیر سامانه طراحی‌شده به منظور انجام عملیات شبیه‌سازی.



شکل (۷). بلوک ایجاد شده جهت انجام شبیه‌سازی.

۳-۲-۲- طراحی نهایی مدار

با توجه به مطالب بیان‌شده، در این قسمت یک فیلتر وفقی به طول ۹ که نویز به همراه یک سیگنال آلوده به نویز به آن اعمال شده طراحی شده و به کمک اسکوپ شکل موج‌های ورودی و خروجی نشان داده شده است (شکل ۸). از ویژگی‌های نرم‌افزار System Generator ایجاد کد HDL یک طرح می‌باشد که برای این پروژه کد Verilog ایجاد گردید و با کمک نرم‌افزار ISE این کد مورد سنتز قرار گرفته و نتایج نهایی آن در ادامه بیان می‌گردد.



شکل (۸). طراحی مدار نهایی به وسیله نرم‌افزار System Generator.

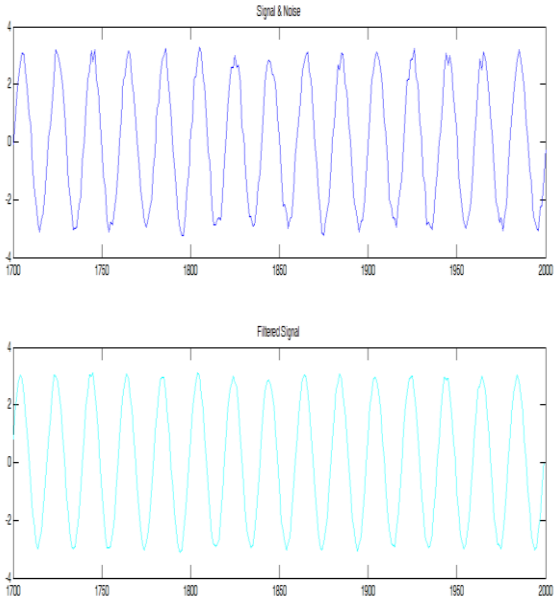
۴- نتایج پیاده‌سازی

در این قسمت با توجه به طراحی فیلتر به دو روش بیان شده نتایج هر کدام ارائه خواهد شد.

۴-۱- نتایج حاصل از پیاده‌سازی با زبان توصیف سخت‌افزار Verilog

برای انجام شبیه‌سازی و پیاده‌سازی از نرم‌افزارهای MATLAB 7.11، ISE 14.4 استفاده شده است تراشه مورد نظر برای پیاده‌سازی طرح XC4VFX12 محصول شرکت Xilinx می‌باشد. در یکی از شبیه‌سازی‌های انجام شده، نویز سفیدی با میانگین صفر و واریانس ۰.۰۴ و یک سیگنال نویزی (موج سینوسی با دامنه ۳ و نویز ایجاد شده با مشخصات ذکر شده) به عنوان ورودی ها ایجاد گردیده و نهایتا یک موج فیلتر شده به عنوان خروجی به‌دست‌آمده است. اندازه گام برابر 0.01 در نظر گرفته شده است. شکل (۹) سیگنال آلوده به نویز با مشخصات مذکور به همراه سیگنال فیلتر شده را نشان می‌دهد.

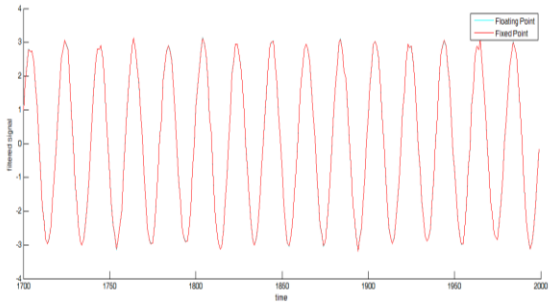
از آنجا که پیاده‌سازی محاسبات به صورت ممیز شناور بر روی تراشه FPGA مساحت زیادی را اشغال کرده و توان زیادی مصرف می‌کند لذا محاسبات را به صورت ممیز ثابت به انجام می‌رسانیم.



شکل (۹). شبیه‌سازی فیلتر وفقی با اعمال سیگنال آلوده به نویز به فیلتر و دریافت سیگنال فیلتر شده توسط نرم‌افزار MATLAB.

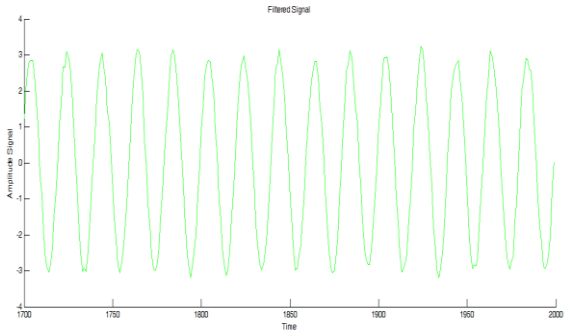
از طرفی چنانچه طول کلمه نامناسب برای حالت ممیز ثابت در نظر بگیریم علاوه‌بر این که بر عملکرد فیلتر تاثیر منفی داشته نویز کوانتیزاسیون را نیز تشدید می‌کند. برای یافتن طول کلمه مناسب جهت اعمال داده های ورودی و انجام محاسبات ریاضی و دریافت خروجی از نرم‌افزار MATLAB استفاده شده است. به این ترتیب که برنامه الگوریتم LMS را در این نرم‌افزار برای دو حالت ممیز ثابت و شناور نوشته و با بررسی انطباق کامل شکل موج‌های

خروجی و نیز به‌دست آوردن حداقل اختلاف میان سیگنال‌های فیلتر شده در این دو حالت طول کلمه مطلوب مورد تایید قرار می‌گیرد. در این طراحی فرمت ۱۶:۱۱ برای پیاده‌سازی ممیز ثابت در نظر گرفته شده است که در شکل (۱۰) انطباق نتایج شبیه‌سازی عملکرد فیلتر برای دو نوع تحقق ممیز ثابت و ممیز شناور نشان داده شده است. حداکثر اختلاف دامنه سیگنال‌های ممیز ثابت و شناور در این دو حالت برابر ۸- ۱۰× ۳,۳ اندازه‌گیری شد.



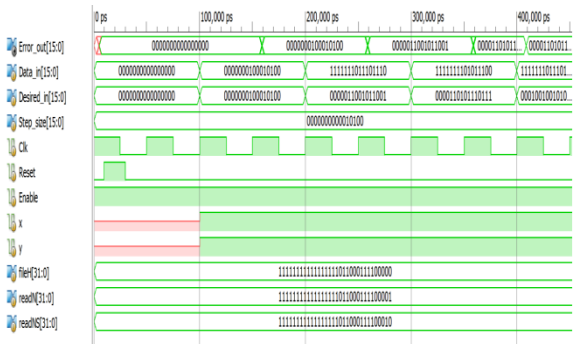
شکل (۱۰). انطباق سیگنال فیلترشده در دو حالت ممیز شناور و ثابت در محیط نرم‌افزار MATLAB.

برای انجام شبیه‌سازی سخت‌افزاری، ابتدا تمام دیتاهای ورودی‌ها را به اعداد هگزادسیمال ممیز ثابت تبدیل کرده و سپس آن را به مدار اعمال می‌نماییم. پس از پایان شبیه‌سازی در نرم‌افزار ISE مجددا با کمک نرم‌افزار MATLAB دیتای خروجی را به صورت ممیز شناور تبدیل کرده و نمایش می‌دهیم. همچنین برای آن‌که پردازش ما منطبق بر پیاده‌سازی عملی بر روی سخت‌افزار واقعی باشد از شبیه‌سازی Post-Pace&Route استفاده شده است. در این تحقیق عملکرد فیلتر به‌ازای ورودی‌های مختلف مورد بررسی و تحلیل قرار گرفت که شکل (۱۱) یکی از این سیگنال‌های فیلترشده را پس از تحقق سخت‌افزاری نشان می‌دهد.

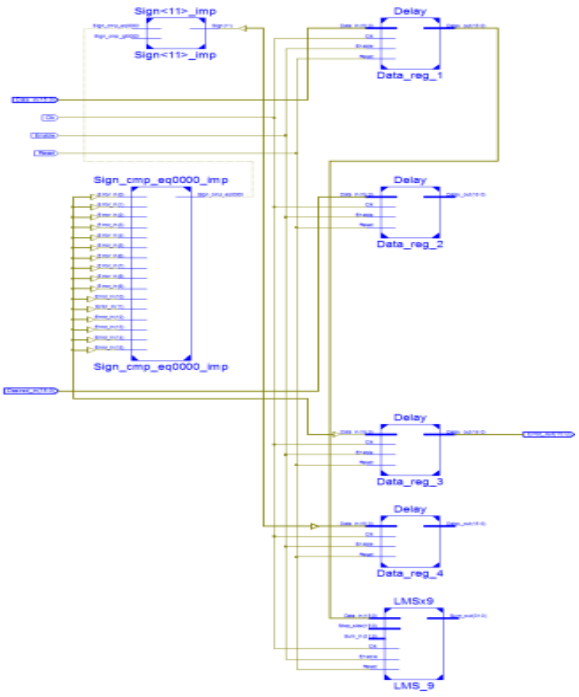


شکل (۱۱). سیگنال فیلترشده خروجی نرم‌افزار ISE.

چنانچه از شکل (۱۱) پیداست عملکرد فیلتر در پیاده‌سازی سخت‌افزاری کاملاً مناسب بوده و بسیار نزدیک به شبیه‌سازی در MATLAB می‌باشد. با وجود آن‌که در پیاده‌سازی سخت‌افزاری خطای برش رخ می‌دهد ولی بر روی شکل موج خروجی نهایی تاثیر بسیار اندکی داشته و نشان‌دهنده آن است که خطای مذکور بسیار ناچیز بوده است. در شکل (۱۲) شاهد نتایج شبیه‌سازی به صورت داده‌های باینری در شبیه‌ساز نرم‌افزار ISE و در شکل (۱۳) نیز بلوک دیاگرام طراحی شده توسط نرم‌افزار ISE هستیم.



شکل (۱۱). داده‌های سیگنال حاصل از شبیه‌سازی توسط نرم‌افزار ISE

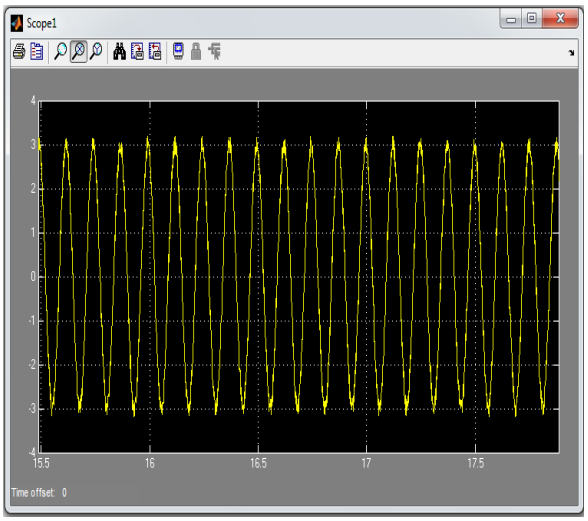


شکل (۱۲). بلوک دیاگرام فیلتر طراحی‌شده توسط نرم‌افزار ISE.

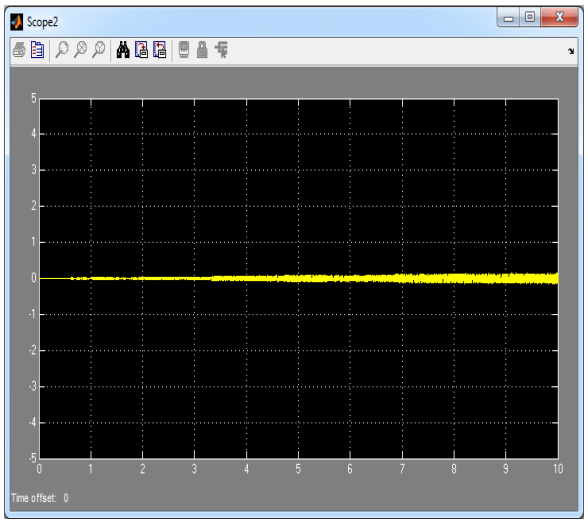
۴-۲- نتایج حاصل از کد نویسی در محیط System Generator

در این قسمت شکل موج‌های به‌دست‌آمده توسط نرم‌افزار System Generator نشان داده می‌شود. در شکل (۱۳) سیگنال آلوده به نویز با مشخصاتی همانند حالت قبل، که به فیلتر اعمال شده است را روی اسکوپ مشاهده می‌شود. در شکل (۱۴) نویز حذف شده از روی سیگنال قبل مشاهده می‌شود. چنانکه مشهود

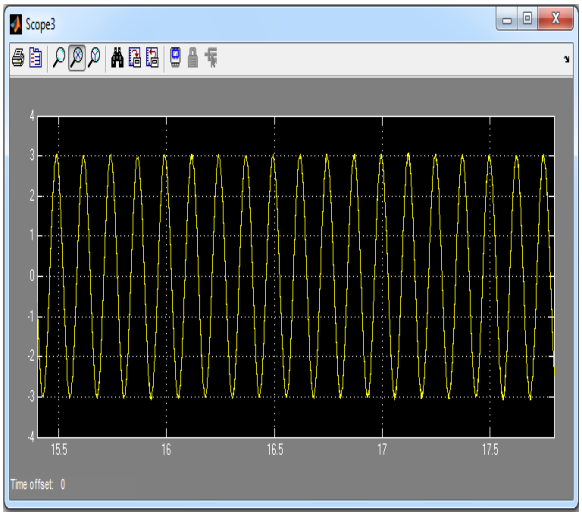
است با گذشت زمان و در اثر بهینه شدن ضرایب فیلتر نویز به‌طور موثرتری حذف می‌گردد. شکل (۱۵) شکل موج فیلتر شده را نشان می‌دهد.



شکل (۱۳). موج آلوده به نویز اعمال شده به فیلتر.



شکل (۱۴). نویز حذف شده از روی سیگنال آلوده به نویز



شکل (۱۵). شکل موج فیلترشده

۵- مقایسه نتایج

نتایج حاصل از پیاده‌سازی سخت‌افزاری توسط نرم‌افزار ISE و نیز نتایج سنتز کدهای Verilog ایجادشده توسط نرم‌افزار System Generator جهت انجام مقایسه از حیث حداکثر فرکانس کار و استفاده از منابع تراشه در جدول (۱) ارائه شده است.

جدول (۱). مقایسه فیلتر طراحی‌شده با طول فیلتر ۹ از نظر سرعت و مساحت اشغالی بر روی تراشه.

مرجع	نویسندگان	نوع تراشه	Slices used	4 Input LUTs	Multipliers	Max. Frequency (MHz)
[۲]	Rosado-Muñoz & Bataller-Mompeán	Xc3s1000	۲۴۲۹	۴۵۲۵	۹	۹/۴۸
		Xc4vfx12	۲۵۸۶	۴۷۷۷	۹	۱۸/۹۷
		Xc5vlx30	۳۹۰۶	۳۵۱۶	۹	۲۵/۲۷
System Generator		Xc4vfx12	۷۳۸	۱۲۲۴	۱۹	۱۴/۳۵
این پروژه		Xc3s1000	۳۵۵	۶۴۸	۱۱	۲۹/۳۰
		Xcfvfx12	۲۳۶	۳۰۴	۱۱	۳۲/۵۳
		Xc5vlx30	۲۷۲	۳۰۴	۱۱	۴۱/۸۹

۶- نتیجه‌گیری

در این مقاله یک فیلتر تطبیقی LMS کارآمد به‌صورت سخت‌افزاری پیاده‌سازی و نتایج آن ارائه شد. نتایج به‌دست‌آمده نشان‌دهنده افزایش فرکانس کاری و کاهش سطح اشغالی فیلتر مزبور بر روی تراشه هدف در مقایسه با سایر کارهای گزارش‌شده در منابع معتبر مربوطه می‌باشد. پیاده‌سازی انواع دیگر الگوریتم‌های LMS همچون NLMS، SLMS و ... و نیز پیاده‌سازی کارآمد الگوریتم حداقل مربعات بازگشتی (RLS) از جمله زمینه‌های تحقیقاتی مهم و جذاب در ادامه این کار می‌باشد. پیاده‌سازی مزبور می‌تواند زمینه‌ساز تحقق سریع و کم حجم فیلترهای دیجیتال همراه با سایر سامانه‌های پردازش سیگنال دیجیتال به‌صورت مجتمع و یک‌پارچه باشد.

۷- مراجع

[1] E. Ifeachor and B. Jervis, "Digital Signal Processing," A Practical Approach, Prentice Hall, 2002.

[2] A. Rosado-Muñoz and M. Bataller-Mompeán, "FPGA Implementation of an Adaptive Filter Robust to Impulsive Noise: Two Approaches," IEEE Trans. Industrial Electronics, vol. 58, no. 3, pp. 860-870, March 2011.

[3] J. Benesty, T. Gänsleraand, and D. Morgan, "Advances in Network and Acoustic Echo Cancellation," Berlin, Springer-Verlag, 2001.

[4] E. Nejevenko and A. Sotnikov, "Adaptive Modeling for Hydroacoustic Signal Processing," Pattern Recognit. Image Anal., vol. 16, no. 1, pp. 5–8, Jan. 2006.

[5] S. Haykin, "Adaptive Filter Theory," Prentice Hall, Upper Saddle River, NJ, First Edition, 2004.

[6] M. Salah, A. Zekry, and M. kamal, "FPGA Implementation of LMS Adaptive Filter," 28th National Raduo Scienec Conf., April 2011.

[7] A. Diggikar and S. Ardhapurkar, "Design and Implementation of Adaptive Filtering Algorithm for Noise Cancellation in Speech Signal on FPGA," Int. Conf. on Computing, Electronics and Electrical Technologies, 2012.

[8] U. Meyer-Baese, "Digital Signal Processing with Field Programmable Gate Arrays," Springer, Berlin, Heidelberg New York, Third Ed., 2007.

[9] A. Elhossini, S. Areibi, and R. Dony, "An FPGA Implementation of the LMS Adaptive Filter for Audio Processing," IEEE Int. Conf. on Reconfigurable Computing and FPGA's, pp. 1-8, Sep. 2006.

[10] H. Simon, "Introduction to Adaptive Filters," Macmillan Publishing Company New York, First Ed., 1985.

Journal Of Electronical & Cyber Defence
Vol. 5, No. 1, 2017, Serial No. 17

Efficient Hardware Implementation of LMS Adaptive Filter on FPGA

E. Saberi, M. Masoumi*

*Islamshahr Azad University

(Received: 11/02/2016, Accepted: 31/10/2016)

ABSTRACT

Adaptive filters are one of the most important building blocks of digital signal processing (DSP) systems which are used in a wide variety of applications such as echo and noise cancellation, channel equalizers, radar and sonar systems. Compared to software implementation, hardware implementation of DSP systems has some inherent advantages including higher speed and throughput, integratability and parallel processing. In recent years, Field Programmable Gate Arrays (FPGAs) have received a lot of attention due to their architecture flexibility, low cost and providing the possibility of parallel processing of DSP algorithms. Efficient realization of adaptive filters on FPGAs has always been a motivational and challenging research topic. In this article, we have presented an efficient hardware implementation of a 9-tap LMS adaptive filter that is much faster and consumes fewer resources compared to similar published works. The results have been verified by comparing those obtained from hardware implementation of an LMS adaptive noise canceller filter with the results obtained from simulation of a similar filter. Since the permanent collection and processing of meaningful signals are of the fundamentals of management cycle and crisis prevention, the presented design can be well used in hardware equipment related to this area.

Keywords: Adaptive filters, LMS algorithm, Hardware implementation, FPGA

* Corresponding Author Email: m_masoumi@iiau.ac.ir