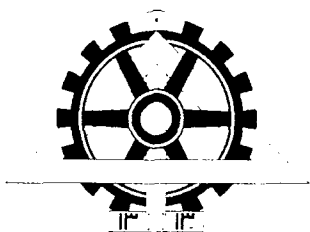


١٤/١٠/١٤

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

٤٢/١٢



دانشگاه تهران

دانشکده فنی

گروه مهندسی برق و کامپیوتر

گروه مهندسی برق و کامپیوتر
دانشکده فنی
دانشگاه تهران

عنوان:

پیاده سازی سخت افزاری یک توربو دیگدر

نگارش:

محمسن نبی پور

۴۲۱۸۲

استاد راهنما:

دکتر سید مهدی فخرایی

استاد مشاور:

دکتر حمید رضا جمالی

پایان نامه جهت دریافت درجه کارشناسی ارشد

در رشته مهندسی برق گرایش الکترونیک

شهریور ماه ۱۳۸۱

۴۲۱۸۲

دانشگاه تهران
دانشکده فنی
گروه مهندسی برق و کامپیوتر

عنوان:

پیاده سازی سخت افزاری یک توربو دیسک

توسط:

مهندس نبی پور

پایان نامه برای دریافت درجه کارشناسی ارشد
رشته مهندسی برق گرایش الکترونیک

از این پایان نامه در تاریخ ۱۳۸۱/۶/۱۳ در مقابل هیات داوران دفاع به عمل آمده و مورد
تصویب قرار گرفت.

سرپرست تحصیلات تکمیلی دانشکده فنی: دکتر ممدعلی بنی هاشمی

مدیر گروه آموزشی:

دکتر محمود کمره ای

سرپرست تحصیلات تکمیلی گروه:

دکتر جواد فیض

استاد راهنما:

دکتر سید مهدی ففرایی

استاد مشاور:

دکتر حمیدرضا جمالی

عضو هیات داوران:

دکتر عبدالرضا نبوی

عضو هیات داوران:

دکتر حمید شفیعی

عضو هیات داوران:

دکتر بهجت فروزنده

مکیده:

در این پایان نامه ساختارهای مختلف سخت افزاری برای پیاده سازی یک دیکدر توربو بر پایه دو الگوریتم اصلی MAP^1 و $SOVA^2$ بررسی شده اند. نمودارها و جداولی جهت ارزیابی و مقایسه ساختارهای مختلف ارائه گردیده است. این نمودارها و جداول به همراه نتایج شبیه سازیهای سیستمی که در ابتدای پایان نامه آمده است می تواند به عنوان یک مرجع کامل برای طراحان مورد استفاده قرار گیرد. کاهش پیچیدگی سخت افزاری، حفظ دقت و بالا بردن سرعت یک دیکدر توربو نیاز سیستمهای جدید مخابرات دیجیتال می باشد. ارائه یک ساختار جدید چند پنجره ای برای پیاده سازی الگوریتم Log-MAP امکان انتخاب بهینه ترین ساختار را از بین ساختارهای پنجره ای فراهم می کند. ساختارهای بر پایه الگوریتم MAP و ساختارهای بر پایه الگوریتم SOVA نیز با هم مقایسه شده اند. در نهایت یک ساختار بدون حافظه و غیر پنجره ای برای پیاده سازی الگوریتم MAP پیشنهاد شده است که پیش بینی می شود پیچیدگی و توان مصرفی آن از ساختارهای پنجره ای کمتر باشد. سرعت آن با ساختارهای پنجره ای یکی باشد و کارایی سیستمی آن نیز بالاتر از کارایی ساختارهای پنجره ای باشد.

فهرست مطالب:

۱	فصل اول : مقدمه.....
۵	فصل دوم : طرز کار توربو کدها.....
۷	۱-۲- ساختار داخلی یک توربو کد.....
۸	۲-۲- کد کننده توربو.....
۹	۳-۲- کد کننده کانوشنال.....
۱۱	۴-۲- دیکدر توربو.....
۱۳	۵-۲- الگوریتم MAP.....
۱۵	۱-۵-۲- ساده سازی الگوریتم MAP و آماده سازی آن برای پیاده سازی عملی.....
۱۷	۲-۵-۲- روش رشته کامل (Full String).....
۱۷	۳-۵-۲- روش پنجره لغزان (Sliding Window).....
۱۹	۴-۵-۲- روش چند شاخه ای (Multiple Branch).....
۱۹	۵-۵-۲- مقایسه پیچیدگی روش های فوق.....
۲۰	۶-۲- الگوریتم SOVA.....
۲۳	فصل سوم : بررسی پارامترهای سیستمی در پیاده سازی یک توربو دیکدر.....
۲۴	۱-۳- تعداد تکرارها.....
۲۵	۲-۳- پهنای بیت احتمالات ورودی (NBQ).....
۲۶	۳-۳- پهنای بیت محاسبات داخلی (NBD).....
۲۹	۴-۳- طول و نقشه ایتترلیور.....
۳۰	۵-۳- انتخاب طول LUD در الگوریتم SOVA.....
۳۲	فصل چهارم : پیاده سازی توربو دیکدر بر پایه الگوریتم MAP.....
۳۴	۱-۴- مروری بر کارهای انجام شده.....
۳۵	۲-۴- ساختار پنجره لغزان پر سرعت (FSSW).....
۳۸	۳-۴- ساختار دو پنجره ای (DW).....
۴۱	۴-۴- ساختار جدید چند پنجره ای (MW).....

۴۳مقایسه ساختارهای مختلف
۴۷ساختار بلوک ACS
۵۰ساختار ایترلیور
۵۲دیکدر کامل
۵۵فصل پنجم: پیاده سازی بلوک SISO بر پایه الگوریتم SOVA
۵۶۱-۵- الگوریتم ویتربی
۵۷۲-۵- الگوریتم SOVA
۵۸۳-۵- سخت افزار SOVA
۵۸۱-۳-۵- بلوک ACS
۶۰۲-۳-۵- رجیسترهای نگه دارنده مسیر
۶۱۳-۳-۵- LUD_Reg_File
۶۲۴-۳-۵- Update Register (UR)
۶۳۵-۳-۵- TD_Reg_File
۶۴۶-۳-۵- Output Unit
۶۴۷-۳-۵- کنترلر
۶۵۴-۵- نتایج سنتز SOVA
۶۵۵-۵- خلاصه فصل
۶۷فصل ششم: نتیجه گیری و پیشنهادات
۶۸۱-۶- بالا بردن سرعت بلوک ACS
۶۹۲-۶- پیاده سازی سخت افزار تکرار و فقی
۶۹۳-۶- طراحی یک ساختار شکل پذیر و قابل تغییر
۶۹۴-۶- ارائه یک روش جدید در پیاده سازی الگوریتم MAP
۷۲ضمائم
۷۹مراجع

فهرست جداول و اشکال:

- جدول (۱-۲): مقایسه روشهای دیکدینگ الگوریتم MAP ۱۹
- جدول (۱-۴): ورودی و خروجی هر بلوک در فازهای پی در پی ۳۷
- جدول (۲-۴): دیاگرام فاز ساختار DW در حالت کار عادی ۴۰
- جدول (۳-۴): سلسله کارهایی که برای توقف عملیات دیکدینگ انجام می گیرد ۴۰
- جدول (۴-۴): دیاگرام زمانی روش سه پنجره ای ۴۳
- جدول (۵-۴): نتایج پیاده سازی توربو دیکدر بر روی FPGA ۵۳
- جدول (۱-۵): نتایج پیاده سازی بیوکهای مسیر دیتای SOVA با $LUD=8$ و $TD=36$ ۶۵
- شکل (۱-۲): کد کننده و دیکدر تکراری متناظر با آن ۷
- شکل (۲-۲): کد کننده کانولوشنال (الف) غیر سیستماتیک و کلاسیک (ب) سیستماتیک و بازگشتی ۸
- شکل (۳-۲): یک کد کننده توربو با نرخ کد $1/3$ ۹
- شکل (۴-۲): دیاگرام حالت یک کد کننده کانولوشنال ۱۰
- شکل (۵-۲): دیاگرام ترلیس متناظر با دیاگرام حالت شکل (۴-۲) ۱۰
- شکل (۶-۲): دیکدر توربو متناظر با کد کننده شکل (۳-۲) ۱۱
- شکل (۷-۲): عملکرد توربو کد با دیکدینگ تکراری ۱۲
- شکل (۸-۲): بلوک کد کننده کانولوشنال و SISO متناظر با آن ۱۲
- شکل (۹-۲): یک شاخه از ساختمان ترلیس ۱۴
- شکل (۱۰-۲): دیاگرام فاز روش رشته کامل ۱۷
- شکل (۱۱-۲): دیاگرام فاز روش پنجره لغزان ۱۸
- شکل (۱۲-۲): دیاگرام فاز روش چند پنجره ای ۱۹
- شکل (۱۳-۲): به روز کردن مقادیر نرم در الگوریتم SOVA ۲۱
- شکل (۱-۳): رفتار الگوریتم تکراری بر حسب تعداد تکرار مختلف ۲۴
- شکل (۲-۳): تاثیر کوانتیزه کردن ضلعات ورودی به توربو دیکدر بر عملکرد آن ۲۶
- شکل (۳-۳): عملکرد یک توربو دیکدر بر حسب تعداد بیت عملیاتی مختلف و با $NBQ=4$ ۲۷

- شکل (۳-۴): تاثیر مقدار NBQ در کزکرد همان توربو دیکدر با تعداد بیت عملیاتی ثابت
 ۲۸ (NBD=7)
- شکل (۳-۵): نمونه یک کوانتایزر غیر خطی قابل استفاده در ورودی دیکدر.....
 ۲۸
- شکل (۳-۶): مقایسه دیکدر Log-MAP با یک دیکدر Threshold-Log-MAP.....
 ۲۹
- شکل (۳-۷): متوسط طول LUD مناسب برای ورودیهای متفاوت.....
 ۳۰
- شکل (۳-۸): انحراف از معیار طول LUD مناسب برای ورودیهای متفاوت.....
 ۳۰
- شکل (۴-۱): نمای کنی یک توربو دیکدر.....
 ۳۳
- شکل (۴-۲): مسیر داده در ساختار FSSW.....
 ۳۵
- شکل (۴-۳): وضعیت مکانی بردارهای رو به جلو (A)، بردارهای رو به عقب (B) و بردار
 مقادیر ورودی (P) برای یک پنجره به طول ۴ از یک تریس.....
 ۳۶
- شکل (۴-۴): ساختار FSSW با طول پنجره ۴.....
 ۳۶
- شکل (۴-۵): سیگنالهای ورودی و خروجی واحد کنترل در ساختار FSSW.....
 ۳۸
- شکل (۴-۶): مسیر داده در ساختار (DW).....
 ۳۹
- شکل (۴-۷): مسیر داده در ساختار TW.....
 ۴۲
- شکل (۴-۸): نحوه عملکرد روش سه پنجره ای.....
 ۴۲
- شکل (۴-۹): تریس استفاده شده در هر یک از کدهای کانوشنل.....
 ۴۳
- شکل (۴-۱۰): پیچیدگی سخت افزار ساختارهای مختلف بر حسب (الف) میزان حافظه و
 (ب) سلولهای منطقی (LC).....
 ۴۴
- شکل (۴-۱۱): مقایسه پیچیدگی هر ساختار بر حسب تعداد گیت منطقی.....
 ۴۵
- شکل (۴-۱۲): تاثیر پارامترهای NBQ و NBD در سخت افزار هر ساختار.....
 ۴۵
- شکل (۴-۱۳): Nd و Nc هر ساختار.....
 ۴۶
- شکل (۴-۱۴): تاخیر و سرعت بلوک SISO بر حسب تعداد ACS های رو به عقب.....
 ۴۷
- شکل (۴-۱۶): ACS استفاده شده برای نرخ کد ۱/۲.....
 ۴۸
- شکل (۴-۱۷): ساختار سریال برای ACS در کدهای با نرخ غیر ۱/n.....
 ۵۰
- شکل (۴-۱۸): ساختمان یک اینترلیور بنوکی.....
 ۵۱
- شکل (۴-۱۹): تحقق ساختار شبه تصادفی با حافظه های RAM و ROM.....
 ۵۱
- شکل (۴-۲۰): سخت افزار کامل یک توربو دیکدر.....
 ۵۲
- شکل (۵-۱): بلوک دیگرام الگوریتم SOVA.....
 ۵۸

- شکل (۲-۵): ساختار بلوک ACS و ترلیس متناظر با کد آن ۵۹
- شکل (۳-۵): ساختار سرینل برای پیاده سازی بلوک ACS ۶۰
- شکل (۴-۵): رجیسترهای نگه دارنده مسیرهای برنده ۶۱
- شکل (۵-۵): ساختار بلوک LUD_Reg_File ۶۲
- شکل (۶-۵): ساختار Update Register ۶۳
- شکل (۷-۵): ساختار بلوک خروجی در دیکدر SOVA ۶۴
- شکل (۸-۵): کنترلر SOVA ۶۵
- شکل (۹-۵): چنمایی (Layout) هسته مدار فوق بر روی کتابخانه سلولهای 0.6um
تعداد گیت: ۳۳۴۰، فرکانس: ۵۰MHz، اندازه: ۲/۰۷۰mm×۳/۲۴۲mm ۶۶
- شکل (۱-۶): یک ستون از ترلیس کد کنولوشنال ۷۰

فصل اول

مقدمه

کتابخانه دانشگاه تهران
موسسه مطالعات و تحقیقات اجتماعی
تهران

با پیشرفت صنعت مخابرات و گسترش اینترنت، نیازهای جدیدی به پردازش سیگنال بصورت Real Time بوجود آمد که پردازشگرهای عام منظوره جوابگوی سرعت و دقت مورد نیاز این کاربردها نبودند و یا حداقل برای این کاربردها بهینه نشده بودند. بنابراین پردازشگرهای جدیدی با قابلیتهای انجام کارهای پردازش سیگنال دیجیتال (DSP^۱) طراحی شد. اولین DSP در سال ۱۹۸۲ توسط شرکت Texas Instrument ساخته شد که با موفقیت روبرو شد. این پردازشگر برای اینکه امکاناتی جهت انجام عملیات DSP فراهم کرده باشد سخت افزاری برای انجام سریع عمل ضرب و یک معماری حافظه با باس دو قلو اضافه کرد که این ساختار تا چند سال در پردازشگرهای DSP تکرار می شد. در نسلهای تازه تر، سعی می شد که بر قابلیتهای نسل قبل افزوده شود تا کارایی بالا رود. اخیراً برای بالا بردن کارایی در DSPها از معماریهای جدید و متفاوت با ساختار سنتی سابق استفاده می شود. به طور کلی پردازشگرهای DSP که توسط شرکتهای بزرگ به بازار عرضه می شود برای کاربرد خاصی طراحی نشده اند و می تون به نوعی آنها را عام منظوره نامید. در کنار این پردازشگرها همواره طراحان مایل به پیاده سازی ساختارهای اختصاصی برای مصارف خاص بوده اند که از تغییر ساختار پردازشگرهای موجود استفاده کرده و سیستم خود را پیاده سازی می کردند. دلایل تمایل طراحان به طرف ساخت سیستمهای خاص منظوره را می توان چنین بیان کرد:

۱- با افزودن سخت افزار اختصاصی بهینه شده به پردازشگر، بار نرم افزاری سیستم را بر دوش سخت افزار می اندازند و به همین دلیل از سرعت بالاتری بهره می گیرند.

۲- برای بعضی از کاربردها، معماریهای موجود جوابگوی دقت و سرعت مورد نیاز نیستند و طراح مجبور است برای تحقق هدف خود از چند پردازشگر بصورت موازی استفاده کند، بنابراین طراح ترجیح می دهد از معماری بهینه شده برای کار خودش که احتمالاً ساده تر نیز هست استفاده کند.

۳- با پیشرفتی که در FPGA^۲ها صورت گرفته برای پیاده سازی طرحهای اختصاصی هزینه زیادی لازم نیست.

یکی از سیستمهای دیجیتال که همیشه بار پردازشی زیادی داشته، دیکدینگ کدهای کانال است. برای مثال الگوریتم ویتربی^۳ که برای دیکد کردن کدهای کانوشنال استفاده می شود ده ها بار مورد تحقیق و بازمینی محققان قرار گرفته و روشهای مختلف پیاده سازی این الگوریتم به صورت نرم افزاری و سخت افزاری گزارش شده است [۱ و ۲]. در سال ۱۹۹۳ دسته جدیدی از کدهای کانال

^۱ Digital Signal Processor^۲ Field Programmable Gate Array^۳ Viterbi Algorithm

به نام توربوکدها توسط سه محقق فرانسوی اختراع شدند [۴]. این کدها با عملکرد بسیار خوبی که دارند از همان ابتدا نظر طراحان سیستم و مدار را به خود جلب کرده‌اند. الگوریتمهای پیچیده با جمع و ضربهای زیاد در ابتدا مانع بزرگی بر سر راه پیاده‌سازی دیکدر این کدها بر روی سیستمهای عملی بودند [۵]. ساده‌سازی دیکدینگ کدهای توربو و حفظ دقت و کارایی این کدها بطور همزمان، همواره دغدغه محققان بوده است و در این راه تغییراتی در جهت کاهش پیچیدگی الگوریتمهای اولیه اعمال شده است [۶ و ۷ و ۸]. با وجود ساده‌سازیهای انجام شده، پیاده‌سازی این کدها بر روی پردازشگرهای معمولی و پردازشگرهای DSP که برای این کار بهینه نشده‌اند ممکن نمی‌باشد زیرا سرعت مدار به شدت پایین خواهد آمد. برای پیاده‌سازی عملی این کدها سه راه کلی وجود دارد:

۱- پیاده‌سازی بر روی DSPهایی که دارای سخت افزار جمع-مقایسه - انتخاب در درون واحد محاسباتی خود می‌باشند.

۲- پیاده‌سازی بر روی FPGA.

۳- پیاده‌سازی بر روی ASIC.

پیاده‌سازی بر روی DSPهای عام منظوره روش خوبی نیست زیرا با توجه به خصوصیات و کارایی پردازشگرهای DSP موجود در بازار اگر بخواهیم از سرعت معقول و مناسبی برخوردار باشیم باید دو، سه یا چهار پردازشگر را با هم موازی کنیم. پیاده‌سازی با مدار اختصاصی ASIC هم پروسه‌ای وقت گیر و پرهزینه می‌باشد. همچنین ایجاد تغییر در طرح بسیار مشکل می‌باشد و اگر تعداد تراشه‌های مورد نیاز خیلی زیاد نباشد این راه مقرون به صرفه نیست. پیاده‌سازی بر روی FPGA از مزایای زیادی برخوردار است. زمان کوتاه برای پیاده‌سازی طرح، امکان تست سریع مدار، امکان ایجاد تغییر در طرح بدون صرف وقت و هزینه زیاد را می‌توان از مزایای پیاده‌سازی سخت افزارهای اختصاصی با تعداد کم نام برد.

در این پایان نامه گزارش پیاده‌سازی یک بلوک توربو دیکدر با الگوریتمهای مختلف و روشهای متفاوت با زبان VHDL آمده است که بر روی کتابخانه های FPGA و ASIC نگاشته شده‌اند. در فصل دوم این رساله با توربوکدها، نحوه کارکرد آنها و اجزاء کد ساز و دیکدر آن آشنا خواهیم شد. فصل سوم به تجزیه و تحلیل مسائل مربوط به پیاده‌سازی این کدها می‌پردازد و پارامترهای مدار از جمله تعداد بیتهای کوانتیزاسیون و تعداد بیت محاسبات داخلی، با توجه به نتایج شبیه‌سازیهای سیستمی را انتخاب می‌کند. فصل چهارم ساختارهای سخت افزاری دیکدر را با الگوریتم Log-Maximum A Posteriori مرور می‌کند. در فصل پنجم ساختار یک دیکدر با الگوریتم Soft Output Viterbi Algorithm بررسی خواهد شد. در فصل ششم به مرور نتایج بدست آمده از

پیاده سازی‌های سخت افزاری خواهیم پرداخت و زمینه‌های تحقیق در این موضوع را معرفی خواهیم کرد.

دلایل انتخاب موضوع فوق بعنوان یک پروژه کارشناسی ارشد را می‌توان به ترتیب زیر بیان کرد:

۱- بحث توربوکدها و دیکدینگ آنها از حرارت، تازگی و جذابیت خاصی برخوردار است و بدلیل عملکرد خوبشان، این کدها در سیستمهای مختلف از جمله در لایه فیزیکی مخابرات سیار نسل سوم به کار می‌روند.

۲- در آزمایشگاه منخایرات سنولی دانشگاه تهران تحقیقات جامع و کاملی بر روی این کدها انجام گرفته بود و آنها را برای پیاده سازی سخت افزاری آماده کرده بود [۹] ولی بدون پیاده سازی سخت افزاری، بعضی نتایج و دعاهای مطرح شده این گروه قابل ارائه و اثبات نبود.

۳- در آزمایشگاه VLSI دانشگاه تهران یک پردازشگر DSP قوی طراحی شده که از ساختار پردازشگر TMSC60 که یکی از قویترین DSPهای موجود در بازار است استفاده کرده است. سخت افزار طراحی شده در این پروژه میتواند بعنوان یک بلوک در داخل این DSP اضافه شود و یا اینکه به عنوان یک پردازنده کمکی در کنار این پردازنده قرار گیرد و بر قابلیتها و قدرت این پردازشگر بیافزاید.

فصل دوه

طرز کار توريو كدها

یک سیستم نمونه مخابراتی از یک طرف به فرستنده و از طرف دیگر به گیرنده ختم می شود که بین فرستنده و گیرنده، کانال انتقال وجود دارد و معمولاً این کانال حاوی نویز است و اطلاعات عبوری را تحت تاثیر قرار می دهد. هدف یک سیستم مخابراتی انتقال داده، انتقال صحیح اطلاعات از فرستنده به گیرنده است. چنانچه اطلاعات خام از فرستنده به کانال نویز آلود وارد شود، به صحت اطلاعات دریافت شده در گیرنده نمی توان اعتماد کرد. بنابراین برای تشخیص اینکه آیا در اطلاعات دریافت شده خطایی رخ داده یا نه و برای تصحیح خطاهای آشکار شده، نیاز به ارسال اطلاعات اضافی علاوه بر اطلاعات خام داریم که به این کار کدینگ کانال^۱ می گوئیم. در بسیاری از سرویسهای مخابرات دیجیتال، پهنای باند و توان فرستنده جزو محدودیتهای سیستم می باشند و پر واضح است که انتخاب نوع کد تشخیص و تصحیح خطا، نقش مهمی در افزایش کارایی سیستم در عرض باند و توان مصرفی دارد.

شانون (Shannon) در [۱۰] ثابت کرد که بهره تکنیکهای تصحیح خطا نامحدود نمی باشد و یک حد تئوری دارد که با توجه به ظرفیت کانال بدست می آید. به این حد، حد شانون^۲ گفته می شود، که برای یک کانال با نویز سفید جمع شونده^۳ از رابطه (۱-۲) بدست می آید.

$$C = B \ln \left(1 + \frac{S}{N} \right) \quad (1-2)$$

که در آن، B پهنای باند سیگنال، S انرژی سیگنال و N انرژی نویز است. از آن زمان تا کنون طراحان کد، همواره در پی طراحی کدی بوده اند که بهره آن تا حد ممکن به حد شانون نزدیک باشد ولی افزایش بهره کدها، پیچیدگی آنها را زیاد می کند بطوریکه پیاده سازی عملی آنها باید همواره با توجه به فن آوریهای موجود سنجیده شود تا برای ساخت به صرفه باشند. توربو کدها^۴ دسته جدیدی از کدهای کانولوشنال^۵ هستند که کارایی خیلی نزدیک به حد شانون دارند. این کدها در سال ۱۹۹۳ توسط سه محقق فرانسوی^۶ معرفی شدند [۴].

^۱ Channel Coding

^۲ Shannon Limit

^۳ Additive White Gaussian Noise (AWGN)

^۴ Turbo Codes

^۵ Convolutional

^۶ Claude Berrou, Alain Glavieux and Punya Thitimajshima