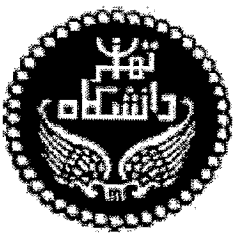
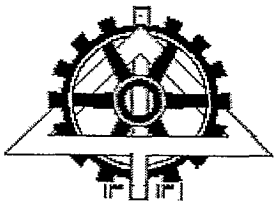
٤٦٧٣٢





دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

بهینه‌سازی معماری سیستم‌های نهفته‌ی چندپردازنده‌ای برای کاربردهای چند رسانه‌ای

نگارش:

محسن یوسف‌پور

استاد راهنما:

دکتر محمودرضا هاشمی

استاد مشاور:

دکتر سید امید فاطمی

پایان‌نامه برای دریافت درجه کارشناسی ارشد در رشته مهندسی کامپیوتر گرایش معماری کامپیوتر

بهمن ۱۳۸۶

۱۳۸۷/۲/۱۴

٤٦٧٣٣

دانشگاه تهران

پردیس دانشکده‌های فنی

دانشکده مهندسی برق و کامپیوتر

پایان‌نامه برای دریافت درجه‌ی کارشناسی ارشد در رشته‌ی معماری کامپیوتر

عنوان: بهینه‌سازی معماری سیستم‌های نهفته‌ی چندپردازنده‌ای برای کاربردهای چند رسانه‌ای

نگارش: محسن یوسف‌پور

این پایان‌نامه در تاریخ ۸۶/۱۱/۲۷ در مقابل هیات داوران دفاع گردید و مورد تصویب قرار گرفت.

معاون آموزشی و تحصیلات تکمیلی پردیس دانشکده‌های فنی: دکتر جواد فیض

رئیس دانشکده‌ی مهندسی برق و کامپیوتر: دکتر پرویز جبه‌دار مارالانی

معاون پژوهشی و تحصیلات تکمیلی دانشکده‌ی مهندسی برق و کامپیوتر: دکتر سعید نگار اصفهانی

استاد راهنما: دکتر محمودرضا هاشمی

استاد مشاور: دکتر سید امید فاطمی

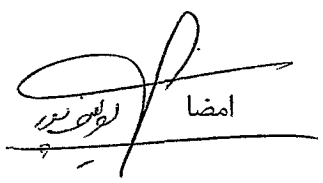
عضو هیات داوران: دکتر شاهین حسابی

عضو هیات داوران: دکتر سید مهدی فخرایی

عضو هیات داوران: دکتر سیامک محمدی

تعهدنامه اصالت اثر

اینجانب محسن یوسف پور تایید می کنم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب است و به دستاوردهای پژوهشی دیگران که در این نوشته از آنها استفاده شده است طبق مقررات ارجاع گردیده است. این پایان نامه قبلا برای احراز هیچ مدرک هم سطح یا بالاتر ارائه نشده است. کلیه حقوق مادی و معنوی این اثر متعلق به دانشکده مهندسی برق و کامپیوتر دانشگاه تهران می باشد.



تقدیم‌نامه:

تقدیم به آنان که «مَثَل‌شان به ستارگان آسمان نماید،
اگر ستاره‌ای فرو شُد، ستاره‌ی دیگر برآید.»

برگرفته از نهج‌البلاغه، خطبه‌ی یک‌صدم
ترجمه‌ی مرحوم سید جعفر شهیدی

تشکر و قدردانی:

انجام این پژوهش و به ثمر رساندن آن بدون یاری جناب آقای **دکتر محمود رضا هاشمی**، استاد محترم راهنمای پروژه و جناب آقای **دکتر سید امید فاطمی**، استاد محترم مشاور پروژه، هرگز میسر نبود. در آغاز گزارشِ نتایج این پژوهش، بر خود لازم می‌بینم که مراتب تشکر و قدرانی‌ام را نسبت به ایشان ابراز دارم.

از پدر، مادر و خواهران بزرگوارم عمیقا سپاسگزارم که در سرتاسر زندگی با حمایت‌های خود، یاور و پشتیبان من بودند.

هم‌چنین همواره ممنون و قدردان دوستانم جناب آقای **مهندس علی ربانی رانکوهی**، جناب آقای **مهندس محمدعلی توتونچیان**، جناب آقای **مهندس امیر نقدی نژاد**، سرکار خانم **مهندس پریسا رزاقی**، سرکار خانم **مهندس مهسان رفوئی**، سرکار خانم **مهندس هما عالم زاده**، جناب آقای **مهندس امین فرمهینی**، جناب آقای **مهندس محمدرضا بینش مروستی** و سایر دوستان بزرگوارم در آزمایش‌گاه‌های پردازش‌های چند رسانه‌ای و TLM، به سبب کمک‌های فراوان و بی‌دریغ‌شان خواهم بود.

این رساله از حمایت مالی **مرکز تحقیقات مخابرات ایران** برخوردار بوده است که به این ترتیب مراتب قدردانی و سپاسگزاری خود را از آن مرکز اعلام می دارم.

چکیده

سیستم‌های بر روی یک‌تراشه راه‌حل نوینی هستند که برای پاسخ‌گویی به نیازمندی‌های سیستم‌های جدید و کاهش زمان رسیدن محصولات به بازار پیش روی طراحان سیستم‌های دیجیتال قرار گرفته‌اند. ورود این پدیده به دنیای طراحی سیستم‌های دیجیتال مشکلات و چالش‌های دیگری را هم‌چون میزان مصرف توان، نحوه‌ی افراز سیستم به عناصر سازنده و چه‌گونگی انتقال داده میان عناصر سازنده وارد چرخه‌ی طراحی سیستم نمود. شبکه‌های بر روی تراشه امروزه یکی از راه‌حل‌های کارا و مناسبی هستند که برای سیستم‌های بر روی تراشه پیش‌نهاد می گردند.

طراحی یک شبکه‌ی ویژه‌ی کاربرد نقش موثری در افزایش کارایی سیستم بر عهده دارد. این شیوه‌ی طراحی زمانی می‌تواند به طور موثر و مفید به حل مساله‌ی انتقال داده بر روی تراشه کمک کند که یک روش و چرخه‌ی مناسب برای طراحی شبکه‌های بر روی تراشه که بر اساس کاربرد ورودی آماده شده‌اند، وجود داشته باشد. در این روش طراحی باید بتوان از یک توصیف اولیه‌ی کاربرد که حاوی اطلاعات ترافیکی آن می‌باشد، شبکه‌ای مناسب برای انتقال داده طراحی نمود.

در این پژوهش تلاش شده است تا یک مسیر طراحی سیستم انتقال داده بر روی تراشه، مبتنی بر شبکه‌های بر روی تراشه طراحی و معرفی گردد. در این مسیر سنتز معماری شبکه‌های بر روی تراشه، برای سه مساله‌ی اساسی نگاشت کاربرد ورودی بر روی شبکه، بهینه‌سازی رفتار شبکه متناسب با نیاز کاربرد و معماری مناسب مسیریاب‌های موجود در شبکه راه‌حل‌هایی پیش‌نهاد شده است. این مسیر طراحی که شامل سه قسمت ذکر شده می‌باشد برای درستی یابی ادعاها و راه‌حل‌ها بر روی شش کاربرد بی‌درنگ که عمدتا در حوزه‌ی کاربردهای چند رسانه‌ای بوده‌اند آزموده شده است. این کاربردها کدگذار MPEG-4، کدگذار JPEG2000، کدگشای شئ‌گرای ویدئو[1] (VOPD)، کاربرد تصویر در تصویر[2] (PIP)، کاربرد نمایش‌گر چندصفحه‌ای[3] (MWD) و یک پردازنده‌ی شبکه‌های کامپیوتری می‌باشند. بررسی و ارزیابی شبکه‌هایی که به وسیله‌ی این مسیر طراحی برای این کاربردها تولید شده‌اند نشان می‌دهد که به کمک این مسیر طراحی می‌توان شبکه‌هایی تولید کرد که از نظر میانگین تعداد جهش هر بسته در قیاس با روش‌های متداول یا پیش‌تر آزموده شده تا حدود نود درصد و از نظر مساحت اشغال شده بر روی تراشه در حدود سی و چهار درصد بهینه‌تر باشند.

1 Video Object Plane Decoder
2 Picture In Picture Application
3 Multi-Window Display

فهرست مطالب

فصل۱ - مقدمه ۱

۱-۱. سیستم‌های بر روی تراشه ۲

۱-۲. سیستم‌های چندپردازنده بر روی یک تراشه ۳

۱-۳. روش‌های انتقال داده ۷

۱-۳-۱. اتصال مستقیم دو عنصر ۷

۱-۳-۲. باس‌های بر روی تراشه ۸

۱-۳-۳. شبکه‌های بر روی تراشه ۱۱

۱-۴. مقایسه‌ی عمل‌کرد باس‌ها و شبکه‌های بر روی تراشه ۱۵

۱-۵. متدولوژی طراحی و سنتز شبکه‌های بر روی تراشه ۱۷

فصل۲ - شبکه های بر روی تراشه و پژوهش‌های پیرامون آن ۲۱

۲-۱. کاربردهای بی‌درنگ و سیستم‌های انتقال داده ۲۲

۲-۲. ساختار و توپولوژی شبکه‌های بر روی تراشه ۲۶

۲-۳. متدولوژی طراحی شبکه‌های بر روی تراشه ۳۰

۲-۴. نگاشت کاربرد و بهینه‌سازی شبکه‌های بر روی تراشه ۳۳

۲-۵. معماری شبکه‌های بر روی تراشه و ادوات مورد استفاده در آن‌ها ۳۸

فصل۳ - نگاشت کاربرد بر روی شبکه؛ گام اول سنتز شبکه‌ی بر روی تراشه ۴۶

۳-۱. نگاشت کاربرد بر روی شبکه ۴۷

۳-۲. نگاشت اتفاقی ۴۹

۳-۳. روش اکتشافی نخست: ترافیک ورودی بیش‌تر و کاربستن ظرفیت بیش‌تر ۵۰

۳-۴. روش اکتشافی دوم: ترافیک خروجی بیش‌تر و کاربستن ظرفیت بیش‌تر ۵۳

۳-۵. روش اکتشافی سوم: ترافیک ورودی بیش‌تر بر روی ترافیک ورودی بیش‌تر ۵۳

۳-۶. روش اکتشافی چهارم: ترافیک خروجی بیش‌تر بر روی ترافیک خروجی بیش‌تر ۵۵

۳-۷. روش‌های اکتشافی پنجم و ششم: نگاشت ناپیوسته بر اساس ترافیک ورودی و خروجی ۵۵

۳-۸. نگاشت به کمک بهینه‌سازی به روش کوچ زنبورها ۵۶

۳-۹. نگاشت به کمک روش الگوریتم ژنتیک ۶۰

فصل۴ - بهینه‌سازی شبکه بر اساس مدل توسعه‌یافته‌ی ترافیک؛ گام دوم سنتز شبکه‌ی بر روی تراشه ۶۳

۴-۱. توصیف توسعه‌یافته‌ی ترافیک ۶۴

۴-۲. بهینه‌سازی برای چند هدف در شبکه‌های بر روی تراشه ۶۹

۴-۳. جست و جو در تمام فضای پاسخ ۷۱

۴-۴. بهینه‌سازی برای چند هدف به روش الگوریتم ژنتیک ۷۲
۴-۵. بهینه‌سازی برای چند هدف به روش کوچ زنبورها ۷۴

فصل۵ – معماری مسیریاب‌های شبکه؛ گام سوم سنتز شبکه‌ی بر روی تراشه ۷۶
۵-۱. مسیریاب‌ها و سوئیچ‌ها در شبکه‌های بر روی تراشه ۷۷
۵-۲. ایده‌ی مسیریاب پویا با پشتیبانی از چند مسیر ۷۸
۵-۳. مسیریاب پویا با پشتیبانی از چند مسیر؛ معماری متمرکز ۷۹
۵-۴. مسیریاب پویا با پشتیبانی از چند مسیر؛ معماری توزیع‌یافته ۸۲

فصل۶ – نتایج عمل‌کرد مسیرسنتز طراحی شده بر روی نمونه‌های مورد بررسی ۸۴
۶-۱. معرفی نمونه‌های مورد بررسی ۸۵
۶-۲. نگاشت کاربردها بر روی شبکه‌های بر روی تراشه ۹۲
۶-۳. ویژگی‌های مسیریاب پویا با پشتیبانی از چند مسیر ۱۰۴
۶-۴. بهینه‌سازی شبکه‌ها با استفاده از مدل ترافیک توسعه‌یافته ۱۱۰
۶-۵. جمع‌بندی نتایج اعمال مسیر سنتز بر روی نمونه‌های مورد بررسی ۱۲۱

فصل۷ – نتیجه‌گیری و گام‌های در پیش رو ۱۲۴
۷-۱. نتیجه‌گیری ۱۲۵
۷-۲. گام‌های در پیش‌رو ۱۲۷

فصل۸ – فهرست منابع ۱۲۹

ضمیمه ۱۳۵

فهرست شکل‌ها

شکل۱-۱. نحوه‌ی اتصال مستقیم عناصر موجود در سیستم ۸
شکل۱-۲. استفاده از باس برای اتصال اجزای سیستم به یک‌دیگر ۸
شکل۱-۳. نحوه‌ی اتصال عناصر سیستم به یک‌دیگر به روش شبکه بر روی یک تراشه ۱۱
شکل۱-۴. میزان مصرف انرژی در سیستم های انتقال داده [۷۳] ۱۶
شکل۱-۵. میزان اثر خازنی اتصالات در اثر افزایش ابعاد سیستم [۷۴] ۱۶
شکل۱-۶. میزان افزایش تاخیر در شبکه‌های انتقال داده به ازای افزایش ابعاد سیستم [۷۴] ۱۷
شکل۱-۷. نمودار افزایش تاخیر خطوط انتقال داده به ازای توسعه‌ی تکنولوژی [۳۹] ۱۷
شکل۲-۱. نمای کلی کدگشای دو استاندارد H.264/MPEG-4 [۱۲] ۲۳
شکل۲-۲. نمای کلی پیاده‌سازی MPEG-4 بر روی FPGA [۷۵] ۲۴
شکل۲-۳. معماری RAW در پیاده‌سازی MPEG-4 [۷۶] ۲۵
شکل۲-۴. طراحی ویژه‌ی کاربرد برای استاندارد MPEG-4 [۷۶] ۲۵
شکل۲-۵. شبکه‌ی طراحی شده برای استاندارد MPEG-4 [۱] ۲۶
شکل۲-۶ برخی از توپولوژی‌های منظم مورد استفاده در شبکه‌های بر روی تراشه [۵۳] ۲۷
شکل۲-۷. نتایج مقایسه‌ی توپولوژی‌های منظم و نامنظم در شبکه‌های بر روی تراشه [۵۳] ۲۸
شکل۲-۸. نمایی از توپولوژی ترکیبی مش و حلقه [۷۷] ۲۸
شکل۲-۹. مقایسه‌ی عمل‌کرد ساختار مرکب مش/حلقه و مش ساده [۷۷] ۲۸
شکل۲-۱۰. شبکه‌ی تولید شده به صورت خودکار و شبکه‌ی طراحی شده‌ی دستی به همراه تصویر سطح تراشه در آن‌دو [۴۱] ۳۰
شکل۲-۱۱. نمودار مقایسه‌ی ویژگی‌های شبکه‌های مبتنی بر بسته و مبتنی بر اتصال [۴۵] ۳۰
شکل۲-۱۲. جریان طراحی شبکه‌های بر روی تراشه [۴۱] ۳۱
شکل۲-۱۳. مسیر طراحی شبکه‌های بر روی تراشه [۴۲] ۳۳
شکل۲-۱۴. تحلیل کاربرد MPEG-4 و جریانات موازی موجود در آن [۶۶] ۳۴
شکل۲-۱۵. شبکه‌ی طراحی شده برای کاربرد MPEG-4 [۶۶] ۳۴
شکل۲-۱۶. دو کاربرد نمایش‌گر چند صفحه‌ای و MPEG-4 که بر روی شبکه‌ی بر روی تراشه نگاشته شده‌اند [۶۷] ۳۵
شکل۲-۱۷. نمودار CDCG برای توصیف ترافیک ورودی [۶۸] ۳۵
شکل۲-۱۸. مسیر سنتز شبکه‌های بر روی تراشه [۷۲] ۳۶
شکل۲-۱۹. مسیریاب طراحی شده برای مدل شبکه‌ی Æthereal [۴۸] ۳۹
شکل۲-۲۰. معماری کلی مسیریاب مورد استفاده در مدل شبکه‌ی MANGO [۷۸] ۴۰
شکل۲-۲۱. معماری مسیریاب مبتنی بر اتصال و از پیش زمان‌بندی شده [۷۹] ۴۱
شکل۲-۲۲. معماری مسیریاب طراحی شده برای شبکه‌های مبتنی بر مدار [۸۰] ۴۲
شکل۲-۲۳. معماری مسیریاب به‌کار رفته در شبکه‌ی عمل‌وند پردازنده [۴۶] ۴۳
شکل۲-۲۴. معماری مسیریاب با پشتیبانی از چند مسیر [۸۲] ۴۴

شکل۲-۲۵. معماری مسیریاب پویای ترکیبی آنالوگ/دیجیتال [۸۳] ۴۵
شکل۴-۱. نمودار فرضی برای نیاز یک اتصال کاربرد به پهنای باند ۶۵
شکل۴-۲. نمودار فرضی دیگر برای نیاز یک اتصال کاربرد به پهنای باند ۶۵
شکل۴-۳. قسمتی از یک شبکه‌ی مثالی که به انتقال داده می‌پردازد ۶۶
شکل۴-۴. نمودار توصیف ترافیک خط A در شبکه‌ی مثالی ۶۷
شکل۴-۵. نمودار توصیف ترافیک خط B در شبکه‌ی مثالی ۶۷
شکل۴-۶. نمودار توصیف ترافیک خط C در شبکه‌ی مثالی ۶۸
شکل۴-۷. نمودار توصیف مجموع ترافیک خطوط A و B در شبکه‌ی مثالی ۶۸
شکل۴-۸. نمودار توصیف مجموع ترافیک خطوط A و C در شبکه‌ی مثالی ۶۸
شکل۵-۱. ساختار جدول مسیریابی در مسیریاب پویا ۸۰
شکل۵-۲. مسیریاب پویا با پشتیبانی از چند مسیر؛ معماری متمرکز ۸۱
شکل۵-۳. مسیریاب پویا با پشتیبانی از چند مسیر؛ معماری توزیع‌یافته ۸۳
شکل۶-۱. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد کدگذار MPEG-4 ..... ۸۶
شکل۶-۲. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد کدگشای شئ‌گرای ویدئو ۸۶
شکل۶-۳. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد تصویر در تصویر ۸۷
شکل۶-۴. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد نمایش‌گر چند صفحه‌ای ۸۷
شکل۶-۵. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد کدگذار JPEG2000 ..... ۸۸
شکل۶-۶. گراف هسته‌های عملیاتی، توصیف‌کننده‌ی کاربرد پردازش‌گر شبکه ۸۸
شکل۶-۷. شبکه‌ی حاصل از نگاشت کاربرد نمایش‌گر چند صفحه‌ای به کمک الگوریتم ترافیک خروجی – کاربستن ظرفیت ۹۴
شکل۶-۸. شبکه‌ی حاصل از نگاشت کاربرد نمایش‌گر چند صفحه‌ای به کمک الگوریتم ترافیک ورودی – ترافیک ورودی ۹۴
شکل۶-۹. شبکه‌ی حاصل از نگاشت کاربرد تصویر در تصویر به کمک الگوریتم ترافیک خروجی – ترافیک خروجی ۹۶
شکل۶-۱۰. شبکه‌ی حاصل از نگاشت کاربرد تصویر در تصویر به کمک الگوریتم ترافیک ورودی – کاربستن ظرفیت ۹۶
شکل۶-۱۱. شبکه‌ی حاصل از نگاشت کاربرد کدگشای شئ‌گرای ویدئو به کمک الگوریتم ترافیک خروجی – ترافیک خروجی ۹۷
شکل۶-۱۲. شبکه‌ی حاصل از نگاشت کاربرد کدگشای شئ‌گرای ویدئو به کمک الگوریتم ژنتیک. ۹۸
شکل۶-۱۳. شبکه‌ی حاصل از نگاشت کاربرد کدگذار MPEG-4 به کمک الگوریتم ترافیک ورودی – ترافیک ورودی ۹۹
شکل۶-۱۴. شبکه‌ی حاصل از نگاشت کاربرد کدگذار MPEG-4 به کمک الگوریتم کوچ زنبورها.. ۹۹
شکل۶-۱۵. شبکه‌ی حاصل از نگاشت کاربرد کدگذار JPEG2000 (توصیف میانگین) به کمک الگوریتم ترافیک ورودی – ترافیک ورودی ۱۰۱
شکل۶-۱۶. شبکه‌ی حاصل از نگاشت کاربرد کدگذار JPEG2000 (توصیف میانگین) به کمک

الگوریتم کوچ زنبورها ۱۰۱
شکل۶-۱۷. شبکه‌ی حاصل از نگاشت کاربرد کدگذار JPEG2000 (توصیف بیشینه) به کمک الگوریتم ترافیک خروجی – کاربستن ظرفیت ۱۰۱
شکل۶-۱۸. شبکه‌ی حاصل از نگاشت کاربرد کدگذار JPEG2000 (توصیف بیشینه) به کمک الگوریتم کوچ زنبورها ۱۰۲
شکل۶-۱۹. شبکه‌ی حاصل از نگاشت کاربرد پردازش‌گر شبکه (توصیف میانگین) به کمک الگوریتم ترافیک خروجی – ترافیک خروجی ۱۰۳
شکل۶-۲۰. شبکه‌ی حاصل از نگاشت کاربرد پردازش‌گر شبکه (توصیف میانگین) به کمک الگوریتم کوچ زنبورها ۱۰۳
شکل۶-۲۱. شبکه‌ی حاصل از نگاشت کاربرد پردازش‌گر شبکه (توصیف بیشینه) به کمک الگوریتم ترافیک خروجی – ترافیک خروجی ۱۰۴
شکل۶-۲۲. شبکه‌ی حاصل از نگاشت کاربرد پردازش‌گر شبکه (توصیف بیشینه) به کمک الگوریتم کوچ زنبورها ۱۰۴
شکل۶-۲۳. شبکه‌ی بستر ارزیابی عمل‌کرد مسیریاب‌ها ۱۰۸
شکل۶-۲۴. نمودار مقایسه‌ی قدرت انتقال داده‌ی مسیریاب‌ها در روی‌کردهای مسیریابی گوناگون ۱۰۸
شکل۶-۲۵. نمودار مقایسه‌ی سرعت انتقال داده در مسیریاب پویا با چند مسیر و مسیریاب تک‌مسیر ۱۰۹
شکل۶-۲۶. نمودار مقایسه‌ی سرعت انتقال داده در شبکه‌های با بارهای مختلف ۱۰۹

فهرست جداول

جدول۶-۱. داده‌های مربوط به سه گونه‌ی توصیف ترافیک برای کاربرد کدگذار JPEG2000..... ۹۰
جدول۶-۲. داده‌های مربوط به سه گونه‌ی توصیف ترافیک برای کاربرد پردازش‌گر شبکه..... ۹۱
جدول۶-۳. داده‌های مربوط به سه گونه‌ی توصیف ترافیک برای کاربرد پردازش‌گر شبکه (ادامه) .. ۹۲
جدول۶-۴. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد نمایش‌گر چند صفحه‌ای بر روی شبکه..... ۹۳
جدول۶-۵. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد تصویر در تصویر بر روی شبکه..... ۹۵
جدول۶-۶. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد کدگشای شئ‌گرای ویدئو بر روی شبکه..... ۹۷
جدول۶-۷. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد کدگذار MPEG-4 بر روی شبکه..... ۹۸
جدول۶-۸. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد کدگذار JPEG2000 بر روی شبکه..... ۱۰۰
جدول۶-۹. ارزیابی شبکه‌ی حاصل از نگاشت کاربرد پردازش‌گر موازی بر روی شبکه..... ۱۰۲
جدول۶-۱۰. اطلاعات مربوط به مساحت اشغال شده توسط مسیریاب‌های بر روی تراشه‌ی FPGA ۱۰۷
جدول۶-۱۱. اطلاعات مربوط به مساحت اشغال شده توسط مسیریاب‌های تک‌مسیر بر روی تراشه‌ی FPGA..... ۱۰۷
جدول۶-۱۲. بررسی عمل‌کرد بهینه‌سازی با چند هدف بر روی توصیف سنتی ترافیک کاربردها. ۱۱۲
جدول۶-۱۳. بررسی عمل‌کرد بهینه‌سازی با چند هدف بر روی شبکه‌ی سنتز شده برای کاربرد کدگذار JPEG2000 حاصل از ترافیک مبتنی بر میانگین پهنای باند..... ۱۱۵
جدول۶-۱۴. بررسی عمل‌کرد بهینه‌سازی با چند هدف بر روی شبکه‌ی سنتز شده برای کاربرد کدگذار JPEG2000 حاصل از ترافیک مبتنی بر بیشینه‌ی پهنای باند..... ۱۱۶
جدول۶-۱۵. بررسی عمل‌کرد بهینه‌سازی با چند هدف بر روی شبکه‌ی سنتز شده برای کاربرد پردازش‌گر شبکه حاصل از ترافیک مبتنی بر میانگین پهنای باند..... ۱۱۸
جدول۶-۱۶. بررسی عمل‌کرد بهینه‌سازی با چند هدف بر روی شبکه‌ی سنتز شده برای کاربرد پردازش‌گر شبکه حاصل از ترافیک مبتنی بر بیشینه‌ی پهنای باند..... ۱۲۰
جدول۶-۱۷. برخی از نتایج بهینه‌سازی شبکه‌های بر روی تراشه برای معیارهایی جز تعداد خطوط شبکه..... ۱۲۱
جدول۶-۱۸. به‌بود مساحت اشغال شده بر روی تراشه در اثر بهینه‌سازی شبکه..... ۱۲۱
جدول۶-۱۹. مقایسه‌ی تعداد جهش در شبکه‌های طراحی شده، با پژوهش‌های پیشین..... ۱۲۳

فصل۱ - مقدمه

۱-۱. سیستم‌های بر روی تراشه

طراحی سیستم‌ها و کاربردهای گوناگون به روش سیستم بر روی یک تراشه[1] قریب یک دهه است که تبدیل به روشی پرطرفدار برای طراحی و پیاده‌سازی سیستم‌های دیجیتال شده است. ویژگی عمده‌ی این‌گونه از سیستم‌ها قرار دادن پردازنده، حافظه و سایر عناصر مورد نیاز سخت‌افزاری بر روی یک تراشه و در کنار یک‌دیگر است. این تراشه می‌تواند یک تراشه‌ی برنامه‌پذیر هم‌چون FPGA و یا تراشه‌ی طراحی شده برای کاربرد خاص یا ASIC باشد. در این‌گونه سیستم‌ها عموماً هر عنصر موجود بر روی تراشه قسمتی از وظایف کلی کاربرد را بر عهده می‌گیرد و تمام عناصر در کنار یک‌دیگر وظیفه‌ی در نظر گرفته شده را به انجام می‌رسانند.

رشد سریع کاربردهای صنعت میکروالکترونیک سبب گشت که تولیدکنندگان صنعتی این‌گونه محصولات ناچار به کاهش زمان میان تولید ایده تا ارائه‌ی محصول به بازار گردند. این رقابت سبب گشت تا تولیدکنندگان در جست و جوی روش‌های طراحی‌ای باشند که نه تنها زمان رسیدن به بازار[2] در آن‌ها اندک است، بلکه تغییر محصول و هماهنگ‌سازی سریع آن با نیازهای نوظهور مشتری نیز آسان‌تر گردد. شاید بتوان ادعا نمود که یکی از مهم‌ترین دلایل اقبال صنعت میکروالکترونیک به سیستم‌های بر روی یک تراشه تسهیل فرآیند طراحی و سپس اعمال تغییرات در سیستم است [۱، ۲].

حضور پردازنده‌های نهفته در سیستم‌های بر روی یک تراشه و قابلیت برنامه‌پذیری آن‌ها باعث می‌شود ایجاد برخی تغییرات در عمل‌کرد سیستم منوط به اعمال تغییرات در تراشه نماند و سرعت هماهنگ‌سازی محصول با بازار بسیار افزایش یابد. این روش طراحی چالش‌های ویژه‌ای را نیز به همراه خود وارد فضای صنعت و آکادمی نموده است. کنار هم قرار دادن پردازنده‌ها، شتاب‌دهنده‌های سخت‌افزاری[3] و واحدهای حافظه بر روی یک تراشه و در کنار یک‌دیگر، و ایجاد ارتباط کارا میان آن‌ها در عین رعایت محدودیت‌های موجود در فضای تراشه، میزان مصرف توان، زمان اجرای عملیات و محدودیت‌های دیگری از این دست ملاحظاتی را وارد فرآیند طراحی سیستم‌های بر روی تراشه می‌نماید [۲]. یکی از این ملاحظات که در طراحی سیستم‌های بر روی یک تراشه باید مورد توجه جدی قرار گیرد، نحوه‌ی ارتباط اجزاء و عناصر موجود بر روی تراشه با یک‌دیگر و انتقال داده میان آن‌هاست. بدون شک باید مکانیزمی در معماری سیستم گنجاند که به وسیله‌ی آن بتوان داده‌های تولید شده در هر عنصر را به عنصر دیگری که به آن داده‌ها نیاز دارد منتقل نمود و جریان داده و کنترل را در سیستم برقرار نمود.

1 System on a Chip
2 Time to Market
3 Hardware Accelerators

۱-۲. سیستم‌های چندپردازنده بر روی یک تراشه

توسعه‌ی کاربردهای مورد نیاز صنعت میکروالکترونیک و توسعه‌ی روزافزون آن، و در کنار آن محدودیت‌هایی که در سیستم‌های طراحی شده بر روی تراشه وجود دارد باعث افزایش عناصر موجود بر روی تراشه شد. در این روند مسیر نویی در طراحی سیستم‌های بر روی تراشه ایجاد گردید که سیستم‌های چندپردازنده بر روی یک تراشه[1] نام‌گرفت. در این‌گونه از سیستم‌ها عملیاتی که در یک کاربرد خاص باید انجام پذیرد، به سبب حجم زیاد داده یا عملیات و از سوی دیگر زمان اندکی که برای اتمام عملیات در آن کاربرد خاص در نظر گرفته شده است، بر روی یک پردازنده‌ی نهفته قابل انجام نمی‌باشد [۲، ۳، ۴، ۵].

یکی از عمده‌ترین مسائلی که سبب می‌شود هنگام پیاده‌سازی یک سیستم به سراغ استفاده از روش چند پردازنده‌ای برویم، زمان اندک برای انجام عملیات است. در شرایطی که منابع پردازشی موجود بر روی یک پردازنده، توان پردازش داده‌های ورودی در زمان اندک را نداشته باشد؛ و در صورتی که ویژگی‌های کاربرد یا داده‌های ورودی چنین اجازه‌ای را به طراح بدهد، می‌توان با استفاده از بیش از یک پردازنده، زمان پردازش را کاهش داده و شرط زمانی نهاده شده بر مساله را برآورده نمود. در کاربردهای چندرسانه‌ای بی‌درنگ مثال‌های فراوانی برای این وضعیت می‌توان یافت. در تلفن‌های همراه، کامپیوترهای جیبی، دستگاه‌های گیرنده‌ی صوت و تصویر، تلویزیون‌های دیجیتال و بسیاری از کاربرهای مشابه دیگر، از زمان دریافت سیگنال‌های ورودی حاوی اطلاعات چندرسانه‌ای، تا زمان ارائه‌ی آن محتوا به کاربرِ دستگاه زمان فراوانی وجود ندارد. سیستم‌هایی که برای این‌گونه کاربردهای بی‌درنگ طراحی می‌گردند باید قابلیت آن‌را داشته باشند که هم‌زمان با دریافت بسته‌های داده که حاوی فریم‌های متوالی صوت یا تصویر هستند آن‌ها را به کاربر عرضه کنند. با افزایش تقاضاهای کاربران و به‌ازای افزایش کیفیت داده‌های چندرسانه‌ای که باید به این ادوات منتقل گردند، حجم داده و حجم عملیات پردازشی نیز افزایش می‌یابد. اما با وجود تمام این افزایش‌ها، فرصت موجود برای پردازش و ارائه‌ی محتوا به کاربر اگر کاهش نیابد، قطعا افزایش نیز نخواهد یافت. در چنین وضعیتی افزایش توان پردازشی سیستم در دست طراحی امری اجتناب ناپذیر است [۳]. استفاده از سیستم‌های چند پردازنده بر روی یک تراشه در چنین سیستم‌هایی می‌تواند راه‌حل مناسبی باشد.

در این شرایط کاربرد به شاخه‌های موازی قسمت شده و هر شاخه مستقلاً بر روی تعدادی از پردازنده‌ها اجرا می‌گردد. این مکانیزم که با الهام از روش پردازش موازی در سیستم‌های کامپیوتری، وارد سیستم‌های بر روی یک‌تراشه شده است، مطابق رابطه‌ی زیر که رابطه‌ی امدال[2] نام دارد، سبب

1 Multiprocessor System on a Chip (MPSoC)
2 Amdahl

افزایش سرعت اجرای سیستم خواهد گشت. در این فرمول f معادل سهمی از کاربرد است که قابل موازی‌سازی نیست و حتماً کل عملیات باید به صورت سریال انجام پذیرد و p برابر است با تعداد پردازنده‌هایی که در طراحی سیستم مورد استفاده قرار گرفته‌اند [۶].

$$Speedup \leq \frac{1}{f + \frac{1-f}{p}} \qquad (۱-۱)$$

آن‌چنان که از رابطه‌ی امدال به‌دست می‌آید، استفاده از چند پردازنده در طراحی یک سیستم تنها آن‌زمانی موثر خواهد افتاد که امکان یافتن تکه‌های موازی در کاربرد و سپردن هریک به یک مجموعه‌ی پردازش‌گر مجزا ممکن و میسر باشد. در غیر این‌صورت به سبب بزرگی f و نزدیک شدن آن به عدد یک، ضریب افزایش سرعت نیز در رقمی حدود یک باقی خواهد ماند و افزودن بر تعداد پردازنده‌ها جز اتلاف منابع ثمر دیگری نخواهد داشت. اما زمانی که استخراج فرآیندهای موازی از کاربرد مورد نظر برای پیاده‌سازی مقدور باشد، با انتخاب درست این فرآیندها و انتقال هریک بر روی پردازنده‌ای مستقل می‌توان مطابق این قاعده نرخ افزایش سرعت را بیش‌تر نمود.

یک از موضوعات فعال در حوزه‌ی تحقیقات، بررسی این دسته از کاربردها و تعیین قسمت‌های مستقل و موازی در کاربرد است. استفاده از چندپردازنده بر روی یک تراشه زمانی به افزایش کارایی منجر خواهد گشت که کاربرد به درستی بر روی چند پردازنده توزیع گردد. در حقیقت رسیدن به بیش‌ترین میزان افزایش کارایی وابستگی فراوانی به افراز درست مساله دارد.

برای کاربردهایی که باید بر روی سیستم‌های بر روی تراشه پیاده‌سازی شوند، سه گونه‌ی متفاوت موازی‌سازی تعریف می‌گردد. برای هرکاربرد تلاش می‌گردد بیش‌ترین موازی‌سازی با عنایت به این سه روی‌کرد تعیین شود و مساله بر اساس آن‌ها به زیر مساله‌های مستقل افراز گردد. این سه روی‌کرد اساسی، موازی سازی در لایه‌ی داده‌ها[1]، موازی سازی در لایه‌ی دستورالعمل‌ها[2] و موازی سازی در لایه‌ی وظایف[3] می‌باشند [۷، ۸].

موازی سازی در لایه‌ی داده‌ها، یکی از سه گونه‌ی موازی سازی است. در این روی‌کرد به مساله تلاش می‌شود جریان‌های موازی در میان داده‌ای که در مساله به جریان می‌افتد یافته شود. در برخی موارد جریان داده‌ی ورودی به گونه‌ای است که می‌توان آن‌را به بسته‌هایی قسمت نمود که پردازش آن‌ها و تولید داده‌ی خروجی بر مبنای آن‌ها وابستگی به داده‌های موجود در سایر بسته‌ها ندارد. با دریافت یکی از این بسته‌ها سیستم قادر خواهد بود خروجی متناظر آن‌را بدون هیچ کاستی تولید نماید. بنابراین می‌توان به جای یک واحد پردازشی، از چند واحد پردازشی مستقل استفاده نمود و به

1 Data Level Parallelism
2 Instruction Level Parallelism
3 Task Level Parallelism

نوبت بسته‌ها را پس از دریافت به واحدی که در حال پردازش نیست تحویل نمود. تنها در نهایت باید واحد پردازش‌گر دیگری در سیستم تعبیه نمود که وظیفه‌ی دریافت خروجی هر یک از این سلول‌های پردازش‌گر و ارائه‌ی آن‌ها با ترتیب مناسب به کاربر را بر عهده داشته باشد. ایده‌ی اصلی موازی سازی در لایه‌ی داده در همین توصیف نهفته است. در حقیقت این موازی سازی عبارت است از یافتن تکه‌هایی از جریان داده‌ی موجود در سیستم که بتوانند به صورت یک اتم منفرد و مستقل از سایر اتم‌ها مورد پردازش قرار گیرند. شکل‌های گوناگونی از این نوع توازی در کاربردهای چند رسانه‌ای قابل مشاهده است. وجود macroblockها در MPEG-2، sliceها در H.264، tileها در JPEG2000 و ساختارهای داده‌ی مستقل و مشابه دیگر در سایر استانداردهای چندرسانه‌ای همگی موید این نکته هستند که توان استخراج این شکل از توازی از کاربردهای چندرسانه‌ای به صورت بالقوه وجود دارد [۹، ۱۰]. در کاربردهای بی‌درنگ دیگر هم با بررسی نوع داده‌ی ورودی و کیفیت ورود آن به سیستم ممکن است بتوان اشکال دیگری از این نوع توازی را یافت. اگرچه استخراج موثر آن از کاربرد مستلزم انجام بررسی‌ها دقیق‌تر در کاربرد است.

شاید تذکر این نکته مفید باشد که مراد از واحد پردازش‌گر تنها پردازنده‌های نهفته‌ی عام منظوره نیست. بلکه هر کدام از عناصر سیستم که ورودی را دریافت نموده و پس از پرداختن و فرآوری داده، خروجی متناظر آن‌را تولید کرده و در اختیار واحد دیگر قرار می‌دهند، در فضای طراحی واحد پردازش‌گر تلقی می‌شوند. این پردازش‌گرها می‌توانند تنها مسئول انجام یک وظیفه‌ی مشخص و معین باشند و یا می‌توانند بسته به شرایط و نوع ورودی وظایف گوناگونی را بر عهده گیرند. اما آن‌چه اهمیت دارد آن است که عنوان پردازش‌گر نه به سبب تنوع وظایف بلکه تنها به علت نقشی که در پرداختن داده و آماده سازی آن در مسیر تولید خروجی نهایی دارند بر آن‌ها بار می‌گردد [۲].

گونه‌ی دیگری از موازی سازی، موازی سازی در لایه‌ی دستورالعمل است. این شکل از موازی سازی شاید ریزدانه‌ترین نگاه به مساله باشد که پیش از این نیز گونه‌هایی از آن در روش‌های افزایش کارایی معماری‌های کامپیوتری مورد استفاده قرار گرفته است. در این روش آن‌چه برای کشف و استخراج عملیات موازی مورد توجه اساسی قرار می‌گیرد، سلسله‌ی دستورالعمل‌هایی است که در کاربرد مورد نظر وجود دارد و باید برای تولید خروجی بر روی واحدهای پردازش‌گر انجام پذیرد. استقلال داده و عملیات در این گونه از کشف توازی، در سطح دستورالعمل‌ها صورت می‌پذیرد. این بدان معنی است که اگر دو دستورالعمل متوالی هر کدام بر روی دو دسته ثبات متفاوت عمل کنند، در استفاده از منابع اشتراک نداشته باشند و خروجی خود را نیز در مقاصد گوناگونی ذخیره نمایند، می‌توان تمهیدی اندیشید که آن‌دو به صورت هم‌زمان اجرا شوند. برای استفاده از این شکل توازی باید ویژگی‌ها خاصی را هم به سخت‌افزار پردازش‌گر و هم به توصیف نرم‌افزاری کاربرد افزود. روش‌هایی هم‌چون گشایش حلقه‌ها، وارد کردن دستورالعمل‌ها بدون در نظر گرفتن ترتیب قرار گیری آن‌ها، انجام عملیات بر روی بیش از یک دستور در شرایط استقلال عملیات و سایر روش‌های مشابه برای استفاده‌ی بهینه از این گونه‌ی موازی سازی مورد بهره‌برداری قرار می‌گیرند. این شکل از توازی به

سبب ریزدانگی تنها ویژه‌ی کابردهای خاص نیست. در هر کاربردی با جستجوی دقیق در توصیف نرم‌افزاری کاربرد می‌توان کمابیش آشکالی از توازی را یافت که با استفاده از معماری درست سخت‌افزار و چینش صحیح دستورالعمل‌ها در کنار یک‌دیگر به سطحی از افزایش سرعت دست‌یافت [۷، ۸، ۱۱].

گونه‌ی سوم روی‌کرد به موازی‌سازی کاربردها، موازی سازی در سطح وظایف است. این شیوه اگرچه در مفهوم مشابه توازی در سطح دستورالعمل هاست، اما درشت‌دانگی موجود در این روش سبب می‌شود که تفاوت‌هایی در روش به کارگیری این‌دو وجود داشته باشد. در برخی کاربردها، جریان داده وارد سیستم می‌گردد و پس از عبور از تعدادی مرحله و انجام پردازش‌های متعدد بر روی آن داده‌ی خروجی آماده می‌گردد. در برخی موارد سلسله‌ی عملیاتی که باید بر روی یک داده صورت پذیرد نسبت به یک‌دیگر دارای استقلال می‌باشند. استقلال دو وظیفه نسبت به یک‌دیگر به این معناست که زمانی که یک وظیفه بر روی یک داده در حال انجام است، وظیفه‌ی مستقل دیگری می‌تواند بر روی داده‌ی دیگری انجام شود. در چنین شرایطی این امکان وجود دارد که با تشخیص درست واحدهای پردازش‌گر مستقل و تامین داده‌ی مناسب برای آن‌ها در هر لحظه کل سیستم در حال پردازش چند بسته‌ی داده‌ی مختلف و در مراحل گوناگون باشد. این شکل از توازی نیز در کاربردهای چند رسانه‌ای کاملاً آشکار است. به عنوان مثال در یک سیستم فشرده سازی ویدئو، عملیات فشرده سازی، جبران سازی تصویر، تخمین حرکت و غیره نسبت به یک‌دیگر دارای استقلال می‌باشند و می‌توانند در هسته‌های پردازش‌گر گوناگون پیاده‌سازی شوند. داده‌ی ورودی به سیستم در هر لحظه در یکی از این واحدهای پردازش‌گر قرار می‌گیرد، در حالی‌که بسته‌های پیشین و پسین داده، به ترتیب در واحدهای پیشین و پسین در حال پردازش می‌باشند [۱۲، ۱۳].

گزیدن یکی از این روی‌کردها و افراز سیستم به زیر سیستم‌های سازنده، تحقیقات وسیعی را به خود مشغول داشته است. انتخاب مناسب‌ترین روش افراز سیستم با توجه به نوع عمل‌کرد سیستم، انتخاب مناسب‌ترین مقصد برای هر وظیفه با توجه به منابع سیستم و در نهایت طراحی ساختار هر هسته‌ی پردازش‌گر و قرار دادن واحدهای حافظه‌ی مورد نیاز در کنار آن، همگی قسمتی از وظایف طراح سیستم است که در چارچوب طراحی ساختار پردازشی سیستم باید مورد توجه قرار گیرند [۱۴، ۱۵، ۱۶، ۸۷].

یکی دیگر از مهم‌ترین وظایف طراح سیستم انتخاب مناسب‌ترین روش برای برقراری ارتباط میان اجزای پردازشی سیستم است که توسط مهندس طراح در گام پیشین، طراحی شده است. در این قسمت طراح سیستم باید با توجه به نیاز هر واحد، و حجم داده‌ای که میان واحدها مبادله می‌شود، سیستم انتقال داده‌ای را برگزیند که بتواند در زمان مناسب، با کیفیت مطلوب و با استفاده‌ی بهینه از منابع موجود، داده را در سیستم به جریان بیندازد. آن‌چه امروز در روش‌های طراحی سیستم مورد توجه قرار می‌گیرد، تلاش طراحان سیستم در مستقل ساختن این دو گام طراحی از یک‌دیگر است. استقلال این دو واحد به این معنی است که ضرورتی ندارد طراحی معماری واحدهای پردازشی و

معماری سیستم انتقال داده در یک چرخه‌ی طراحی صورت پذیرد، بلکه می‌توان به‌ترین گزینه برای هر دو را در دو فضای جست و جوی مختلف یافت. تنها باید در نظر داشت که برخی اطلاعات، هم‌چون حجم داده‌ی مورد نیاز یا زمان انتقال داده، بدون توجه به جزئیات طراحی واحدهای پردازش‌گر باید بین این دو چرخه‌ی طراحی به جریان بیافتد [۱۷، ۱۸، ۱۹، ۲۰].

فضای طراحی سیستم‌های انتقال داده، فضای بسیار وسیعی است. راه‌حل‌های متعددی برای حل این مساله وجود دارد. پارامترهای متنوعی نیز در انتخاب به‌ترین گزینه موثر می‌باشند. انتخاب یک متدولوژی طراحی مناسب برای طراحی سیستم انتقال داده، مستقل از چگونگی افراز سیستم و جزئیات واحدهای پردازش‌گر، از نکات مهمی است که هنگام طراحی یک مجموعه‌ی کامل شامل تعدادی عنصر پردازش‌گر بر روی تراشه باید مورد عنایت جدی قرار بگیرد [۱۷، ۱۸، ۲۱].

۱-۳. روش‌های انتقال داده

۱-۳-۱. اتصال مستقیم دو عنصر

از دیرباز روش‌های گوناگونی برای انتقال داده در سیستم‌ها مورد استفاده قرار می‌گرفت. یکی از روش‌های اولیه که برای این امر در نظر گرفته می‌شد، اتصال عناصر مرتبط به یک‌دیگر به صورت مستقیم بود. اتصال مستقیم دو عنصر به یک‌دیگر و برقراری ارتباط میان آن‌ها نیز به روش‌های گوناگونی صورت می‌پذیرد. برخی از این روش‌ها، تبدیل به استانداردهای انتقال داده به روش موازی یا سریال گشته‌اند و برخی نیز به صورت خاص منظوره طراحی شده و استفاده از آن‌ها رایج و متداول نیست.

یکی از سریع‌ترین و کاراترین روش‌های ممکن برای جابه‌جایی داده استفاده از همین نحوه‌ی انتقال داده است، اما معایب این روش در شرایطی که تعداد عناصر مرتبط با یک‌دیگر زیاد باشد، هم‌چون نیاز به اتصالات فراوان، افزایش مساحت اشغال شده، ایجاد محدودیت در اتصال یک عنصر با تعداد عناصر زیاد و به عبارت دقیق‌تر توپولوژی اتصالات و خاص منظوره شدن بیش از حد طراحی و ایجاد مانع بر سر راه استفاده‌ی مجدد و یا توسعه‌ی سیستم باعث شد تا دامنه‌ی استفاده از این روش برای سیستم‌های بزرگ چندان گسترده نگردد. الگوی کلی این روش جابه‌جایی داده در شکل۱-۱ ترسیم شده است.