دانشگاه صنعتی امیرکبیر

دانشکده مهندسی برق-گروه الکترونیک

گزارش سمینار درس پردازش موازی

Reconfigurable Computing

ارائه دهنده:

علی اکبر سلطانلو

۸۶۱۲۳۰۳۰

استاد راهنما:

دکتر معتمدی

زمستان ۱۳۸۶

فهرست

مقدمه

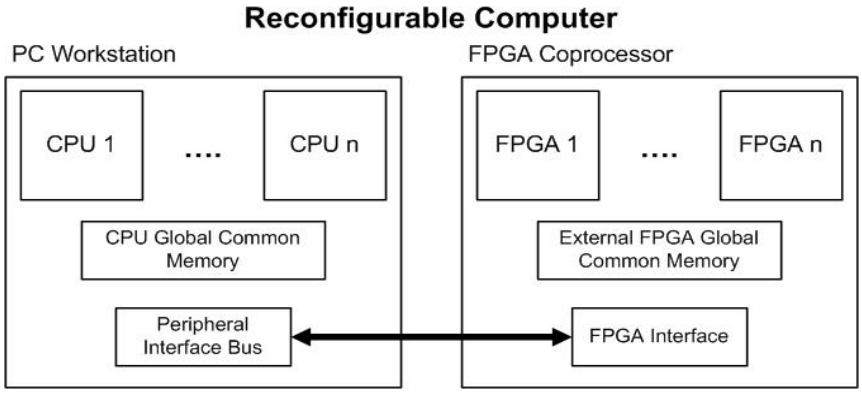
دو روش اصلی در محاسبات مورد نیاز در اجرای الگوریتم ها وجود دارد. روش اول استفاده از تکنولوژی سخت افزاری است یعنی استفاده از ASIC [1] ها. ASIC ها بطور خاص برای انجام محاسبات معینی طراحی و ساخته می شوند. آنها در اجرای محاسباتی که برای آنها ساخته شده اند بسیار کارا و سریع هستند. اما بعد از ساخته شدن قابل تغییر و اصلاح نیستند. و برای تغییرات باید از نو طراحی شوند و IC های جدید در سیستم ها تعبیه شوند که بعضا سخت افزار سیستم نیز باید از نو طراحی شود.

روش دوم استفاده از میکروپروسسور است که بصورت نرم افزاری برنامه ریزی می شود. پردازنده برای انجام محاسبات باید یک سری دستورالعمل را به ترتیب اجرا کند. با برنامه ریزی مجدد سیستم، کارکرد آن عوض می شود بدون آنکه نیاز باشد سخت افزار را تغییر دهیم. اما این انعطاف پذیری به قیمت کاهش سرعت و افزایش زمان اجرای محاسبات نسبت به ASIC تمام می شود.

محاسبات قابل پیکر بندی مجدد[2] می تواند این فاصله موجود بین سخت افزار و نرم افزار را برطرف کرده و بازده سیستم را بالا ببرد. وسایل قابل پیکر بندی مجدد [3] شامل تعدادی FPGA هستند که آنها نیز حاوی تعداد زیادی عناصر محاسباتی می باشند که کارکرد آنها توسط تعدادی بیت تعیین می شود. این عناصر محاسباتی به عنوان بلوک های منطقی [1] شناخته می شوند که اتصالات بین آنها نیز می تواند برنامه ریزی شود. بنابراین مدارهای دیجیتال می توانند در این سخت افزار قابل پیکر بندی نگاشت گردند.

---

[1] - Application Specific Integrated Circuit
[2]- Reconfigurable Computing
[3]- Reconfigurable Devices



شکل۱ – معماری کلی یک کامپیوتر قابل پیکربندی مجدد

FPGA ها به همراه محاسبات قابل پیکربندی برای سرعت بخشیدن به انواع مختلفی از برنامه های کاربردی[۲] بکار می روند. برای رسیدن به این مقاصد، سیستم های قابل پیکربندی مجدد معمولا از ترکیب اجزای قابل پیکربندی مثل FPGA به همراه یک میکروپروسسور همه منظوره استفاده می کنند. میکروپروسسور کارهایی را انجام می دهد که بطور کارا در FPGA نمی تواند انجام شود. در عوض FPGA هم محاسباتی را که در میکروپروسسور طول می کشد به سرعت انجام می دهد.

در حقیقت یک میکروپروسسور می تواند برای حل مسائل متفاوتی در یک زمان بکار رود، بر خلاف ASIC ها که برای کار خاصی طراحی می شوند.

یک کامپیوتر قابل پیکر بندی مجدد[۳] می تواند توسط FPGA کارهایی را که یک ASIC انجام می دهد پیاده سازی کند. کارهایی که اگر FPGA نبود میکروپروسسور مجبور بود با اجرای متوالی تعدادی دستورالعمل انجام دهد. در حقیقت کامپیوتر قابل پیکربندی مجدد با پیکربندی قطعات FPGA و سیم بندی آنها در زمان اجرای برنامه کاربردی این محاسبات را انجام می دهد. این یعنی می توان در هر

[1]- Logic Block
[2] -Application
[3] -Reconfigurable Computer

برنامه کاربردی یک ASIC متفاوت داشت متناسب با برنامه کاربردی مورد نظر- به همراه اینکه سیستم ما هنوز میکروپروسسور را به عنوان یک پردازنده همه منظوره در اختیار دارد.

در نتیجه کامپیوتر قابل پیکربندی مجدد می تواند به عنوان یک ابزار قدرتمند برای بسیاری کاربرد ها به کار رود. مثل پردازش سیگنال یک بعدی و دو بعدی- پردازش تصویر- سرعت بخشیدن به شبیه سازی- محاسبات علمی.

در حقیقت انعطاف پذیری میکرو پروسسور با قدرت FPGA به نحو عالی ترکیب شده است.

پروسه طراحی شامل تقسیم کردن برنامه به دو بخش، یکی برای پیاده سازی روی سخت افزار و دیگری برای پیاده سازی در نرم افزار روی میکروپروسسور است. پیاده سازی سخت افزار مستلزم پیکربندی FPGA است که ابتدا باید به شکل توصیف سخت افزار مشخص شود. اما در اینجا گاها ظرفیت FPGA ها نیز مورد اهمیت قرار می گیرد. FPGA فقط آن مقدار از برنامه را می تواند بپذیرد که در ساختار های آن جا شود. قسمت های باقیمانده از برنامه را می توان با استفاده دوباره از بعضی بخش های سخت افزار در حین اجرای برنامه پیاده سازی کرد. این پروسه تحت عنوان RTC [1] شناخته می شود.

اما همانطور که این روش دارای مزیت سرعت بخشیدن به اجزای بخش های بیشتری از برنامه است، اما یک سربار پیکربندی را نیز مطرح می کند که مقدار سرعت ممکن را محدود می کند، چون پیکربندی چندین میلی ثانیه طول می کشد. پیکربندی سریع و کارا موضوعی بسیار مهم وحیاتی است.

[1] -Run Time Reconfiguration

کامپیوتر SRC-6e

کامپیوتر SRC-6e توسط شرکت SRC Computer طراحی و ساخته شده است که در مقایسه با سایر کامپیوتر های این شرکت مثل SRC7 دارای کمترین تعداد پردازنده، کمترین تعداد FPGA و حداقل حافظه مورد نیاز است. البته معماری آن شبیه بقیه کامپیوتر های این شرکت است و نیز محیط نرم افزار آن با بقیه یکسان است.

محیط SRC-6e یکی از اولین ماشین های قابل پیکر بندی مجدد همه منظوره است که انعطاف پذیری میکروپروسسور های سنتی را با قدرت FPGA های امروزی ترکیب کرده است. در این محیط محاسبات به دو دسته می تواند تقسیم شود:

۱- آنهایی که در نرم افزار اجرا می شوند توسط دستورات میکروپروسسور

۲- آنهایی که با استفاده از توانمندی های FPGA های نوین در سخت افزار قابل پیکربندی مجدد اجرا می شوند.

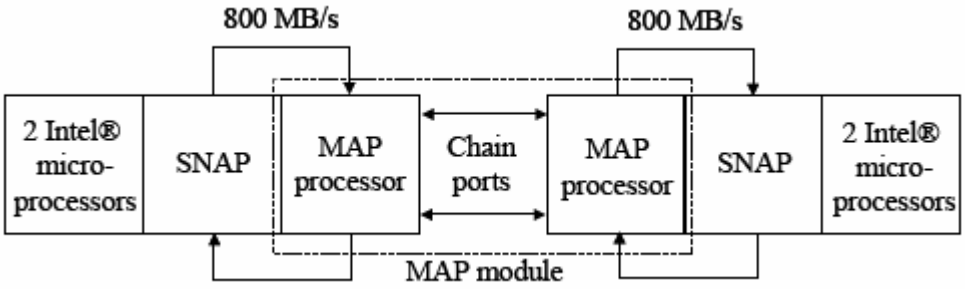
مدل برنامه ریزی که در اینجا مورد هدف قرار گرفته، این است که توجه برنامه ریز و طراح از جزییات توصیف سخت افزاری کنده شده و در عوض به تابعی که قرار است پیاده سازی شود معطوف گردد. این روند امکان استفاده از طراحان نرم افزاری و به کار بردن ریاضیات در جهت بهبود کد را فراهم می کند و بطور ذاتی زمان حل مسئله را کاهش می دهد.

با وجود این روند که برنامه نویس را از جزییات توصیفات سخت افزار جدا کنیم، محیط SRC یک برنامه ریز با انعطاف پذیری ضروری برای بکار بردن معماری های متفاوت را فراهم می کند که می تواند برای پیاده سازی توابع یکسان بکار رود. این مورد از طریق استفاده از یک زبان برنامه نویسی

سطح بالاتر نظیر Forteran یا C می تواند انجام شود. اما قبل از آن باید دید که چه توابعی را که می توان در سخت افزار پیاده سازی کرد.

معماری سخت افزار SRC-6e

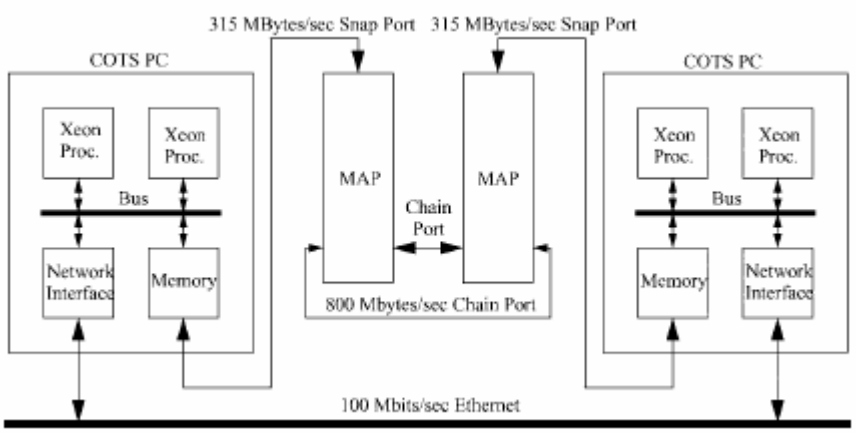
SRC-6e از دو برد پردازنده و یک برد MAP [1] تشکیل شده است. برد MAP دارای Xillinx FPGA (Virlex 2 xc2v 6000) است. برد های پردازنده ها از طریق کارت SNAP با نرخ انتقال داده 800 MB/S به بردMAP متصل شده است. کارت SNAP روی اسلات DIMM مادر بورد قرار می گیرد و اتصال میکروپروسسور و MAP را برقرار می کند.



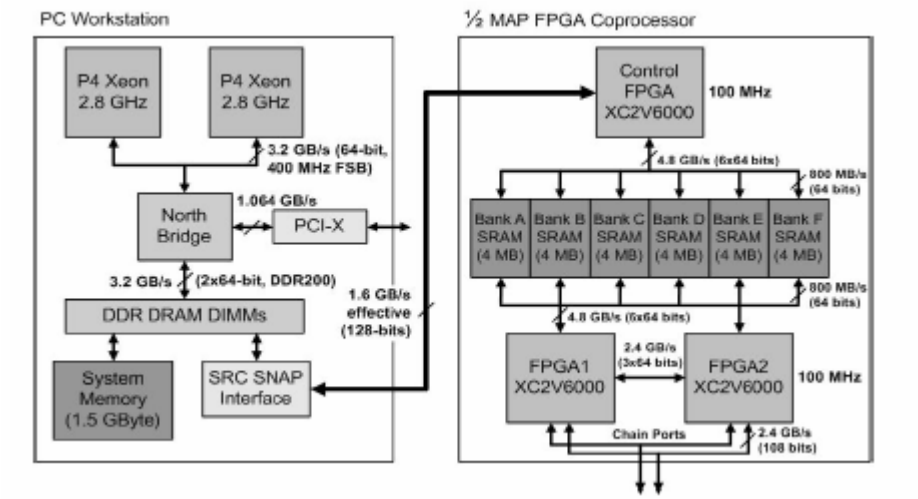
شکل۲ – معماری کلی کامپیوتر قابل پیکربندی مجدد SRC-6e

همانطور که در شکل دیده می شود هر دو PC [2] ماشین های دو پردازنده ای COTS [3] هستند که هر کامپیوتر دارای دو پردازنده Intel xeon با کلاک 1000 MHz، 1.5GB حافظه و رابط شبکه با سرعت 100MB/S است. هر کامپیوتر بطور مستقل عمل می کند اما از طریق شبکه با کامپیوتر دیگر در ارتباط است.

[1] - Multi-Adaptive Processor (پردازنده چند وفقی)
[2] - Personal Computer
[3] - Commercial-Off-The-Shelf



شکل۳ – معماری SRC-6e



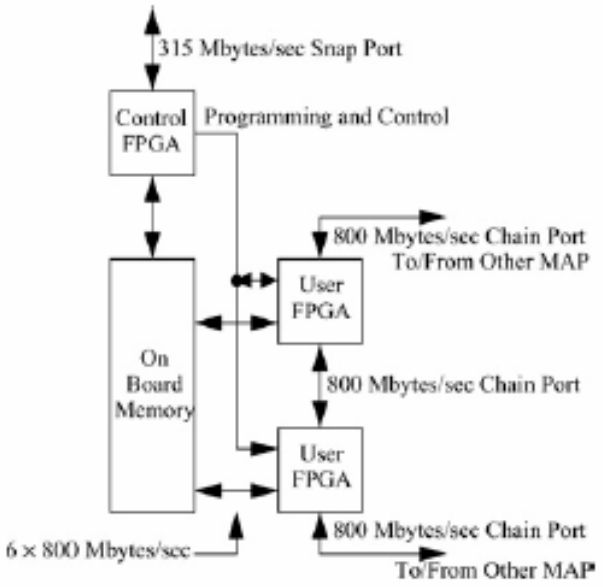
شکل٤ – جزئیات بیشتری از معماری SRC-6e

همچنین هر PC به MAP از طریق پورت SNAP نیز متصل است. MAP بخش قابل پیکر بندی مجدد این معماری است که مخفف Multi Adaptive Processor است. هر MAP شامل ۳ تا Xillinx virtex 2 FPGA است. دو تا از FPGA ها برای اعمال منطقی هستند که کاربرد تعریف می کند، در حالی که سومی به عنوان کنترل کننده MAP [1] بکار می رود. همچنین هر MAP شامل 24MB حافظه

[1] -MAP Controller

RAM است که به ٦ بانک 4MB تقسیم می شوند. پهنای باند بین هر بانک حافظه و FPGA های کاربر 800MB/S است.

پورت SNAP دارای ٦٤بیت عرض و پهنای باند 315MByte/S دارد. و MAP را به PC از طریق پورت حافظه PC متصل می کند. در حقیقت پورت SNAP روی اسلات حافظه موجود در PC نصب می شود. این معماری موجب می شود انتقال به MAP یا ازMAP در ماکزیمم سرعتی که باس حافظه می تواند پشتیبانی کند انجام شود و به این ترتیب نیاز به تغییر و اصلاح PC از بین می رود.



شکل٥ – معماری MAP

MAP ها از طریق پورت chain با حداکثر پهنای باند 800MByte/S به یکدیگر متصل می شوند. استفاده پورت chain توسط برنامه کاربردی کاربر که بر روی FPGA اجرا می شود تعریف می شود. برای عملیات های I/O با پهنای باند بالا، مانند کاربردهای جنگ الکترونیک، پورت های chain می

توانند در هر نقطه شکسته شوند و به سایر وسایل مانند ADC، DAC، DRFM [1] متصل شوند. هر PC در SRC-6e تحت سیستم عامل مستقل کار می کند.

مدل برنامه نویسی SRC-6e

کمپایل کردن برنامه های کاربردی برای سیستم SRC-6e شامل مراحل بیشتری نسبت به یک کامپیوتر معمولی است. تا جاییکه امکان پذیر بوده سعی شده کمترین اجزای محاسباتی برای کاهش هزینه و زمان توسعه در نظر گرفته شود.

کامپایلر های مورد نیاز برای MAP به شکل Forteran و C موجود هستند. وجود این کامپایلرها به همراه محیط پیشرفته نرم افزاری LINUX ، پیچیده ترین محیط نرم افزاری کامپیوتر ها که تا به امروز ایجاد شده است را باعث می شوند. بسیاری از کامپیوتر های قابل برنامه ریزی مجدد تجاری موجود با فراخوانی روال های آماده [2] برنامه ریزی می شوند. در حقیقت این روال ها از قبل توسط سازنده نوشته شده است. کامپایلر های C و Forteran موجود، تعداد زیادی بهینه سازی کد و تغییر چندین جمله مستقل به جملات قابل اجرا روی سخت افزار موازی را فراهم می کنند. همچنین کامپایلر همه رابط های سخت افزاری و نرم افزاری لازم را تولید می کند.

طراحی وتوسعه نرم افزار SRC-6e نیاز دارد که پروگرمر بطور صریح و شفاف معین کند که کدام بخش از الگوریتم باید روی MAP اجرا شود و کدام بخش روی پردازنده. در حقیقت کدی که قرار است روی MAP اجرا شود در فایل جداگانه ای قرار می گیرد و کدی که قرار است توسط

[1] -Digital Radio Frequency Memory
[2] -Canned Procedures

ریزپردازنده اجرا شود نیز در فایل دیگری قرار می گیرد. در واقع کدی که روی MAP اجرا می شود آن بخش از الگوریتم است که اگر روی ریزپردازنده انجام شود زمان زیادی صرف آن می شود.

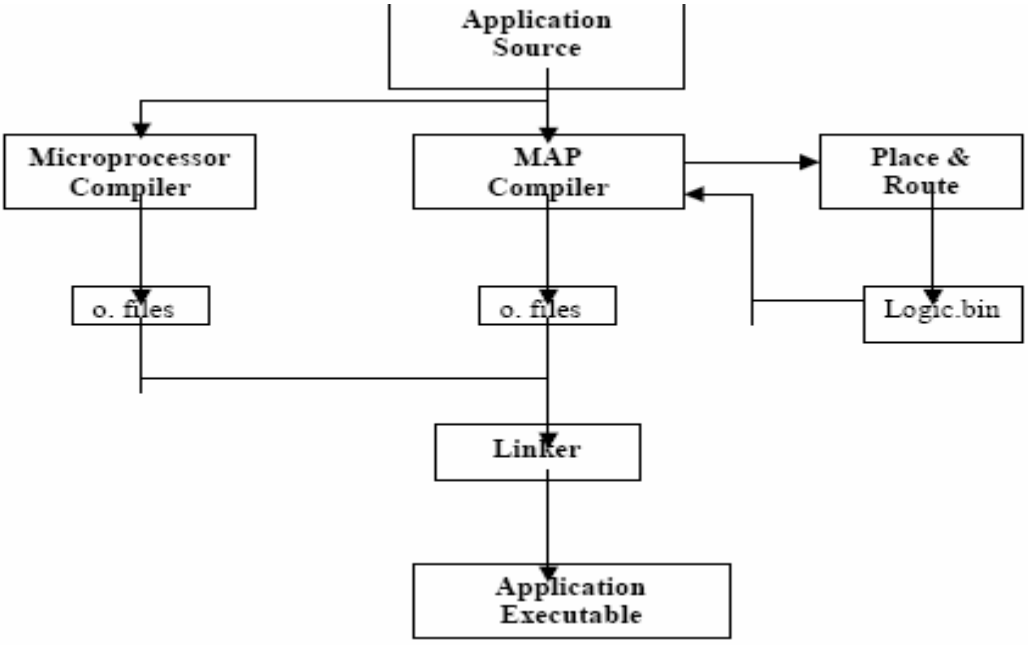
MAP و ریزپردازنده هر دو می توانند در C یا Forteran برنامه ریزی شوند. همچنین MAP می تواند توسط VHDL یا Verilog نیز برنامه ریزی شود.

اجرای برنامه SRC-6e در ابتدا روی یک یا تعداد بیشتری پردازنده آغاز می شود. زمانیکه یک روال که با استفاده از MAP می تواند سریعتر اجرا شود فراخوانی شود، MAP برنامه ریزی می شود و داده ورودی به روال از حافظه عمومی در PC به حافظه MAP کپی می شود و سپس جریان اجرا به MAP منتقل می شود. در این نقطه اگر PC برنامه ای برای اجرا داشته باشد یا کار دیگری را بتواند انجام دهد بطور موازی به اجرای آن می پردازد وگرنه به حالت معلق می ماند تا وقفه ای از طرف MAP مبنی بر اتمام محاسبات دریافت کند. بعد از اینکه MAP یا MAP ها کارشان تمام شد داده خروجی حاصل از روال MAP از حافظه MAP به حافظه عمومی PC کپی می شود و اجرا به PC باز می گردد.

با این روش چندین Task بطور همزمان می توانند در PC ها و MAP ها اجرا شوند.و در ضمن در یک MAP چندین برنامه متفاوت هم می توانند اجرا شوند اگر آنها را به FPGA های متفاوت اختصاص دهیم.

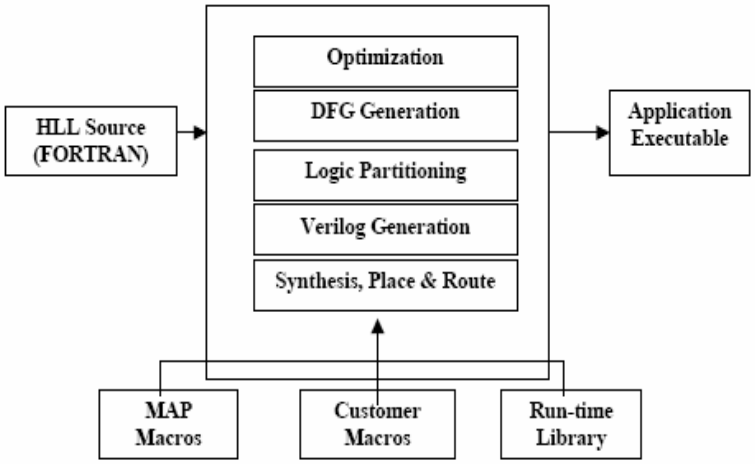
کامپایلر MAP

کامپایلر MAP روال ها را به Object file ترجمه می کند. فرایند ترجمه شامل چندین مرحله است که هر مرحله که توسط بخشی مجزا از کامپایلر MAP انجام می شود.



شکل٦ – نمای کلی فرآیند کامپایل کردن در SRC-6e

بخش بهینه سازی کامپایلر بخش های معنایی و گرامری زبان را بررسی می کند. بخش DFG generation تحلیل اضافه ای را انجام می دهد و روال های نوشته شده برای اجرا روی MAP را تشخیص می دهد. این بخش همچنین روابط بین بلوک های پایه یک روال را نیز نشان می دهد.

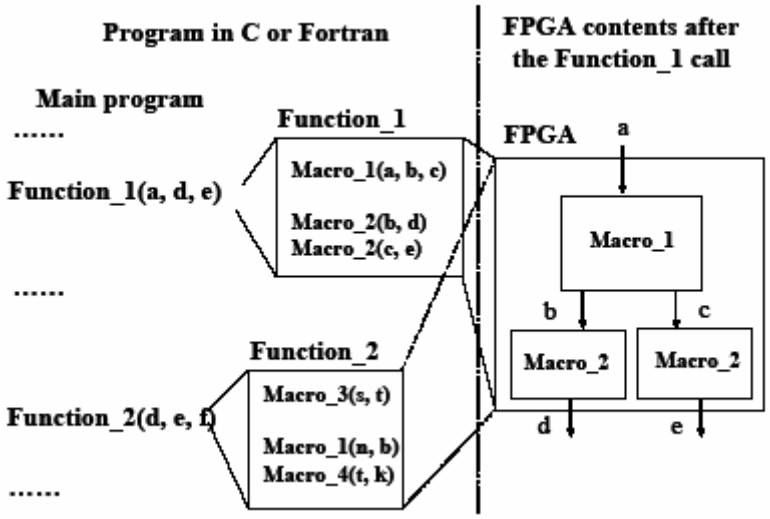


شکل٧ – فرآیند کامپایل کردن

اسمبلر Comlist یک تابع C ایجاد می کند که حاوی قالب های Comlist برای یک روال است. قالب در زمان اجرا کامل می شود. Comlist بر روی پردازنده کنترل MAP اجرا می شود و زیر برنامه ای که روی سخت افزار MAP اجرا خواهد شد را کنترل می کند. همچنین انتقال داده بین حافظه MAP و حافظه عمومی سیستم بر عهده Comlist است.

بخش تولید کننده Verilog کد های لازم را برای برنامه ایجاد می کند. تولید کننده Verilog ، Data flow graph تولید شده در واحد های قبلی را به کد های verilog ترجمه می کند.

کامپایلر MAP فایل منبع که شامل عملگر های متنوعی است را به ماکرو ها ترجمه می کند. ماکرو ها به عنوان قطعات منطقی از قبل طراحی شده که قابل پیاده سازی سخت افزار هستند شناخته می شوند و هر ماکرو یک تابع مشخص را پیاده سازی می کند. البته کاربر هم می تواند تعدادی ماکرو تعریف کند.



شکل۸ – مدل برنامه ریزی SRC-6e

شکل بالا نگاشت بین ماکرو و محتوای FPGA را نشان می دهد. توجه به این نکته ضروری است که محتوای هرتابع MAP بصورت نرم افزاری سخت افزار داخل FPGA را توصیف می کند.

زمانی که Macro-2 دو بار در تابع۱ فراخوانی می شود کامپایلر دو بار بلوک منطقی نشان دهنده Macro-2 را ایجاد می کند. مقادیر موجود در آرگومان ماکرو اتصالات موجود در سخت افزار را تعیین می کنند.

هر زمان که یک تابع MAP جدید فراخوانی شود محتوای داخل FPGA باید دوباره پیکربندی شود. اگر یک تابع یکسان چندین بار پشت سر هم فراخوانی شود پیکربندی فقط دفعه اول انجام می شود.

پیاده سازی Benchmark بر روی SRC-6e

به منظور مقایسه بازده [1] کامپیوتر قابل پیکر بندی مجدد SRC-6e و یک کامپیوتر معمولی با پردازنده Pentium 4 می توان یک الگوریتم یکسان را روی هر دو پیاده سازی کرد. در انجا دو الگوریتم Tripple DES و DES Breaker را مورد بررسی قرار می دهیم.

الگوریتم Tripple DES

الگوریتم Tripple DES یک استاندارد رمزنگاری امریکایی و یکی از معروفترین الگوریتم های رمزنگاری می باشد. Tripple DES به چند روش می تواند تعریف شود که در این جا از روش پیشنهاد شده توسط Tuchman استفاده شده است.

[1] -Preformance

رمز نگاری DES روی بلوک های داده ٦٤ بیتی (٨ بایتی) و توسط یک کلید ٦٤ بیتی انجام شده و یک داده رمز شده ٦٤ بیتی به عنوان خروجی تولید می شود. ما در اینجا از نحوه پیاده سازی الگوریتم صرف نظر کرده و فقط نتایج حاصله را بررسی می کنیم.

اندازه گیری زمان اجرا

١- اندازه گیری روی SRC-6e

برای اجرای الگوریتم ابتدا باید FPGA پیکربندی شود. در این الگوریتم نیازی نیست چندین بار FPGA را پیکر بندی کرد. سه نوع زمان اجرا روی SRC-6e اندازه گیری شده است:

١- زمان اجرای کلی، شامل زمان پیکربندی و زمان انتقال داده سربار (زمان انتقال داده ورودی و خروجی بین حافظه میکروپروسسور و حافظه MAP.

٢- زمان اجرای کلی، بدون زمان پیکربندی.

٣- زمان اجرا فقط روی MAP. این زمان شامل زمان پیکربندی و زمان انتقال داده سربار نیست.

Length (words)	Total time (sec)	Throughput (MB/s)	Total time w/o config (sec)	Throughput w/o config (sec)	MAP time (sec)	MAP Throughput (MB/sec)
1024	0.099	0.08	0.00050	16.29	1.12E-05	730.12
10,000	0.100	0.80	0.00133	60.33	0.000101	792.23
25,000	0.102	1.96	0.00266	75.19	0.000251	796.88
50,000	0.105	3.81	0.00492	81.30	0.000501	798.44
100,000	0.108	7.37	0.00932	85.84	0.001001	799.22
250,000	0.123	16.27	0.02228	89.77	0.002501	799.69
500,000	0.146	27.32	0.04421	90.48	0.005001	799.84

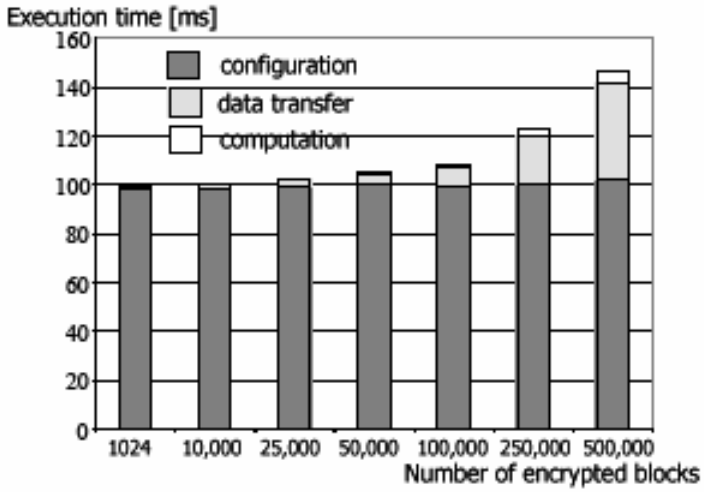
جدول١ – زمان اجرا و گذردهی برای سه نوع اندازه گیری زمانی مختلف

جدول بالا زمان اجرا و گذردهی برای هر سه نوع اندازه گیری زمانی گفته شده را نشان می دهد. تعداد بلوک های رمزگذاری شده از ١٠٢٤ تا ٥٠٠٠٠٠ تغییر می کند و هر بلوک شامل ٨ بایت داده (٦٤ بیت)

است. ستون دوم کل زمان اجرا شامل زمان پیکربندی و زمان انتقال را نشان می دهد. گذردهی بنابر نسبت تعداد بلوک های رمزگذاری برحسب MByte به کل زمان اجرا برحسب ثانیه است که در ستون ۳ نشان داده شده است.

ستون ٦ زمان اجرا بدون در نظر گرفتن زمان پیکربندی را نشان می دهد. با کم کردن ستون ٦ از ستون ۲ زمان پیکربندی FPGA روی برد MAP بدست می آید که حدود 100ms است. همانطور که در ستون ۵ ملاحظه می شود اگر از زمان پیکربندی جلوگیری کنیم گذردهی سیستم پیشرفت قابل ملاحظه ای می کند.

ستون ۷ گذردهی MAP را نشان می دهد که عدد قابل ملاحظه ای است. پیاده سازی این الگوریتم روی MAP بصورت Pipeline انجام شده و به همین علت با هر کلاک یک بلوک داده در خروجی آماده می شود.



شکل۹ – اجزای کل زمان اجرا برحسب تعداد بلوک های داده در الگوریتم Tripple DES

شکل بالا کل زمان اجرا بر حسب تعداد بلوک های پردازش شده را نشان می دهد. همانطور که ملاحظه می شود زمان پیکربندی برای همه یکسان است چون فقط یک بار انجام می شود.

۲- اندازه گیری زمان اجرا بر روی پردازنده Pentium 4

همین الگوریتم روی یک کامپیوتر شخصی با پردازنده 1.8GHz Pentium 4 و 512KByte حافظه پنهان و 1GByte حافظه اصلی پیاده سازی شد. البته دو روند در این پیاده سازی مد نظر گرفت. پیاده سازی توسط زبان C و توسط کدهای اسمبلی.

	P4 non-optimized		P4 optimized	
Length (words)	Total time (sec)	Throu-ghpu (MB/sec)	Total time (sec)	Throu-ghput (MB/s)
1024	0.00379	2.15920	0.00102	8.06299
10,000	0.03663	2.18400	0.01010	7.92354
25,000	0.09279	2.15540	0.02561	7.80969
50,000	0.18637	2.14627	0.05116	7.81937
100,000	0.37150	2.15343	0.09960	8.03253
250,000	0.91990	2.17415	0.25478	7.84985
500,000	1.83200	2.18341	0.49841	8.02546

جدول ۲ – کل زمان اجرای الگوریتم Tripple DES روی پردازنده Pentium 4

همانطور که ملاحظه می شود در پیاده سازی بصورت اسمبلی زمان اجرا ٤ برابر کاهش یافته است.

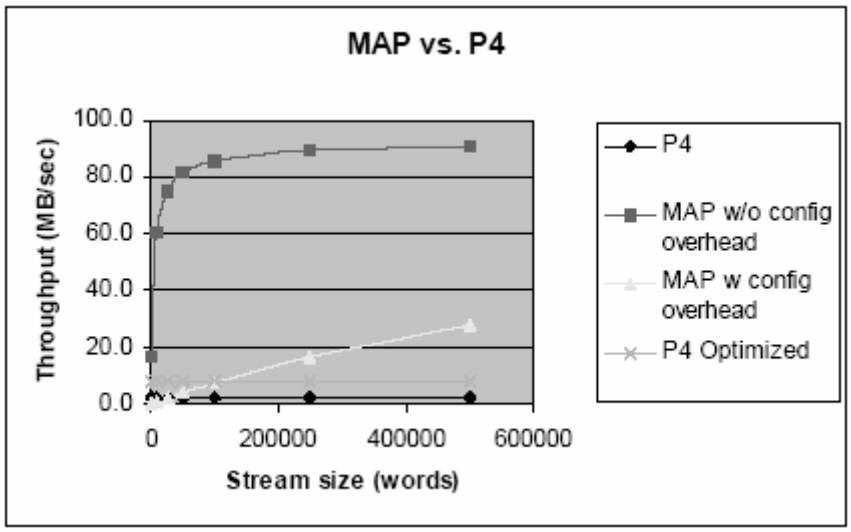
مقایسه

بر مبنای اندازه گیری های انجام شده افزایش سرعت در SRC-6e در مقابل Pentium 4 در جدول زیر آورده شده است. در پیاده سازی با زبان اسمبلی روی Pentium 4 ، SRC-6e حدود ۳,۵ برابر سریعتر از Pentium 4 است.

Length (words)	MAP vs. Non-optimized P4			MAP vs. Optimized P4		
	Speedup Total	Speedup Total w/o Config	Speedup MAP	Speedup Total	Speedup Total w/o Config	Speedup MAP
1024	0.04	7.5	338.2	0.01	2.0	90.6
10000	0.37	27.6	362.8	0.10	7.6	100.0
25000	0.91	34.9	369.7	0.25	9.6	102.0
50000	1.78	37.9	372.0	0.49	10.4	102.1
100000	3.42	39.9	371.1	0.92	10.7	99.5
250000	7.49	41.3	367.8	2.07	11.4	101.9
500000	12.51	41.4	366.3	3.40	11.3	99.7

جدول ۳– افزایش سرعت در SRC-6e در مقایسه با Pentium 4 در اجرای الگوریتم Tripple DES

بدون در نظر گرفتن زمان پیکربندی افزایش سرعت به ۱۱ برابر می رسد و همچنین با حذف زمان انتقال این عدد به ۱۰۰ افزایش پیدا می کند که این زمان ها در پیاده سازی با C حدود ٤ برابر می شوند.



شکل ۱۰ – منحنی گذردهی برای SRC-6e و Pentium 4

پیاده سازی الگوریتم DES Breaker

به عنوان دومین Benchmark برای مقایسه SRC-6e و Pentium 4 الگوریتم شکستن DES انتخاب شده است. در این الگوریتم کلید لازم برای انطباق بلوک داده کد نشده و بلوک داده رمزگذاری شده جستجو می شود. این الگوریتم برخلاف الگوریتم های قبلی کمترین انتقال و بیشترین پردازش را دارد. برای اجرای این الگوریتم از FPGA های دیگر در MAP نیز می توان استفاده کرد و بطور موازی جستجو را انجام داد.

Number of DES units	Search Size (keys)	Total Time (sec)	Total Time w/o Config. (sec)	MAP only (sec)
1 X	128,000	0.101	0.0016	0.00128
	1,000,000	0.109	0.0103	0.01001
	100,000,000	1.101	1.0006	1.00001
2 X	128,000	0.101	0.0009	0.00064
	1,000,000	0.104	0.0053	0.00500
	100,000,000	0.602	0.5006	0.50000
4 X	128,000	0.101	0.0006	0.00032
	1,000,000	0.102	0.0028	0.00250
	100,000,000	0.352	0.2503	0.25000
8 X	128,000	0.097	0.0005	0.00016
	1,000,000	0.098	0.0015	0.00125
	100,000,000	0.222	0.1253	0.12500

جدول٤ – زمان اجرا و گذردهی الگوریتم DES Breaker بر روی SRC-6e

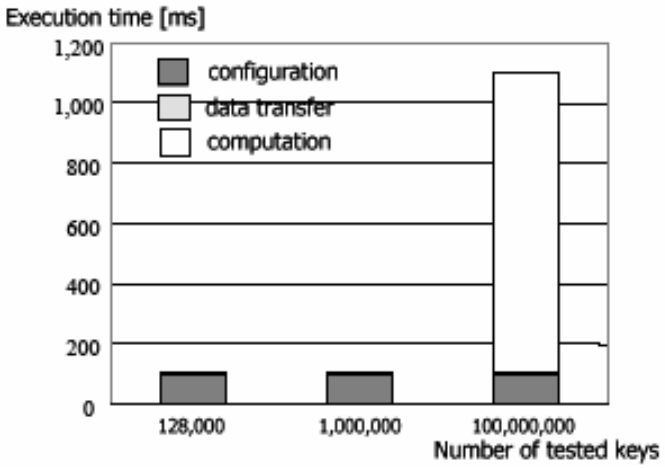
ستون اول در جدول بالا تعداد واحد های FPGA بکار رفته در MAP را نشان می دهد.

Search size (keys)	Time for non-optimized DES (sec)	Time for optimized DES (sec)
128,000	0.25	3.22
1,000,000	1.97	24.64
100,000,000	198.40	2394.51

جدول٥ – کل زمان اجرای الگوریتم DES Breaker در Pentium 4

با مقایسه جداول و می توان افزایش سرعت در SRC-6e نسبت به Pentium 4 بررسی کرد. ملاحظه می شود که الگوریتم مورئ نظر برای پیاده سازی بصورت اسمبلی برای این تعداد داده (کلید) بهینه نیست و پیاده سازی C بهتر جواب داده است.

همانطور که در شکل ۱۱ مشخص است زمان اجرا طولانی است درحالیکه زمان پیکربندی در مقایه با ان زمان کوچکی محسوب می شود و زمان انتقال هم صفر است زیرا کلید ها برای تست کردن در MAP تولید می شوند.



شکل۱۱ – بخش های مختلف زمان اجرا برحسب تعداد بلوک های پردازشی در الگوریتم DES Breaker

Number of DES units	Search Size (keys)	Speedup Total	Speedup w/o Config.	Speedup MAP only
1 X	128,000	2.5	157.4	194.0
	1,000,000	18.1	191.3	197.3
	100,000,000	180.2	198.3	198.4
2 X	128,000	2.5	265.1	387.8
	1,000,000	18.9	373.0	394.6
	100,000,000	329.4	396.3	396.8
4 X	128,000	2.5	406.7	774.6
	1,000,000	19.3	706.0	789.0
	100,000,000	563.0	792.6	793.6
8 X	128,000	2.6	500.0	1562.5
	1,000,000	20.1	1313.3	1576.0
	100,000,000	893.7	1583.4	1587.2

جدول ٦ – افزایش سرعت در SRC-6e نسبت به Pentium 4 در پیاده سازی الگوریتم DES Breaker

اگر ۸ واحد جستجو بصورت موازی پیاده سازی شوند (چون حجم الگوریتم کم است می توان تا ۸ واحد موازی را در یک FPGA پیاده سازی کرد) با در نظر گرفتن زمان پیکربندی و زمان انتقال افزایش سرعت به ۸۹٤ خواهد رسید. البته بدون در نظرگرفتن زمان پیکربندی افزایش سرعت ۱۵۸۳ خواهد بود.

نتیجه گیری:

دو Benchmark مورد استفاده Tripple DES و DES Breaker هر کدام در کلاس متفاوت از الگوریتم ها را نشان می دهند. هر دو حساس به محاسبه هستند اما در مشخصات انتقال داده متفاوت عمل می کنند. Tripple DES بر مبنای جریان انتقال داده زمان واقعی بنا شده درحالیکه DES Breaker کمترین ورودی و خروجی را نیاز دارد. همانطور که در نتایج دیده شد بازده SRC-6e به نوع Application بکار رفته بستگی دارد.

زمان پیکربندی در همه سیستم هایی که از FPGA استفاده می کنند وجود دارد. زمان پیکربندی در سیستم هایی که از پورت سریال برای پیکربندی استفاده می کنند بسیار با اهمیت است. در کاربردهایی که زمان اجرا و محاسبه زیاد است زمان پیکربندی می تواند تادیده گرفته شود و برعکس در کاربرد های کوتاه و ترتیبی باید سعی شود که این زمان مینیمم شود.

در پیاده سازی الگوریتم های پیچیده ممکن است نیاز باشد که یک FPGA چندین بار پیکربندی گردد که این امر موجب زیاد شدن زمان پیکربندی و ناکارآمدی SRC-6e خواهد شد.

مراجع:

[1]- William Stallings, Cryptography and Network Security, Prentice Hall, 1999
[2]- Performance and Overhead in a Hybrid Reconfigurable Computer, Osman Devrim Fidanci1, Dan Poznanovic2, Kris Gaj3, Tarek El-Ghazawi1, Nikitas Alexandridis1 1George Washington University, 2SRC Computers Inc., 3George Mason University 2003
[3]- ElectronicWarfare Digital Signal Processing on COTS, Computer Systems with Reconfigurable Architectures, journal of aerospace computing, information, and, communication vol. 2, October 2005
[4]- High-Level Language Abstraction for Reconfigurable Computing, Walid A. Najjar University of California, Riverside
[5]- Implementation of Elliptic Curve Cryptosystems on a Reconfigurable Computer
[6]- Configurable Computing: A Survey of Systems and Software, Katherine Compton Department of Electrical and Computer Engineering Northwestern University Evanston, IL USA 2005