دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

پایان‌نامه کارشناسی
مهندسی نرم‌افزار

تحلیلی بر عملکرد حافظه مشترک تحت بارهای کاری چند منظوره در پردازنده‌های گرافیکی

نگارش
پارسوآ خورسند رحیم‌زاده

استاد راهنما
دکتر حمید سربازی آزاد

۴ تیر ۱۳۹۶

@title

چکیده

در پردازنده‌های گرافیکی روی هر چندپردازنده جریانی یک واحد حافظه اختصاصی برای نگهداری داده‌های مربوط به فضای آدرس‌دهی مشترک ریسمان‌های در حال اجرا تعبیه شده‌است. فضای این حافظه به بلوک‌های هر چندپردازنده به طور جداگانه تخصیص می‌یابد و میان ریسمان‌های هر بلوک به طور مشترک استفاده می‌شود.

حافظه مشترک به برنامه‌نویس اجازه می‌دهد تا اطلاعات مورد نیاز برای اجرای هر بلوک از ریسه‌های برنامه را روی چندپردازنده جریانی شامل آن‌ها ذخیره کند. حافظه مشترک به دلیل ظرفیت و در نتیجه مساحت کم و نیز قرار گرفتن روی تراشه چندپردازنده زمان دسترسی بسیار پایینی (در حد چند کلاک) دارد. به این ترتیب برنامه‌نویس می‌تواند بخشی از داده را که احتمال می‌دهد قرار است در باز زمانی فعلی با نرخ بالا مورد دسترسی قرار گیرد را روی این حافظه بارگذاری کند تا دسترسی به آن با سربار کم امکان‌پذیر باشد.

در نسل‌های اولیه پردازنده‌های گرافیکی، حافظه‌مشترک به عنوان راه‌حلی برای مدیریت پیچیدگی‌های ناشی از زمان دسترسی غیرقابل پیش‌بینی حافظه اصلی مورد پیاده‌سازی قرار گرفت. چنین رویکردی در کاربردهای گرافیکی که نیاز به تضمین نرخ فریم ثابتی وجود دارد از توجیه‌پذیری بالایی برخوردار است.

در ادامه نتایج تحلیل روی عملکرد و میزان کاربرد حافظه مشترک در بارهای کاری محاسباتی ارائه می‌شود و در نهایت راهکارهایی برای بهبود کارایی این مدل حافظه پیشنهاد می‌گردد .

کلمات کلیدی: پردازنده‌های گرافیکی عام‌منظوره، حافظه مشترک، کارایی، زمان دسترسی .

فهرست مطالب

فهرست تصاویر

۱ مقدمه

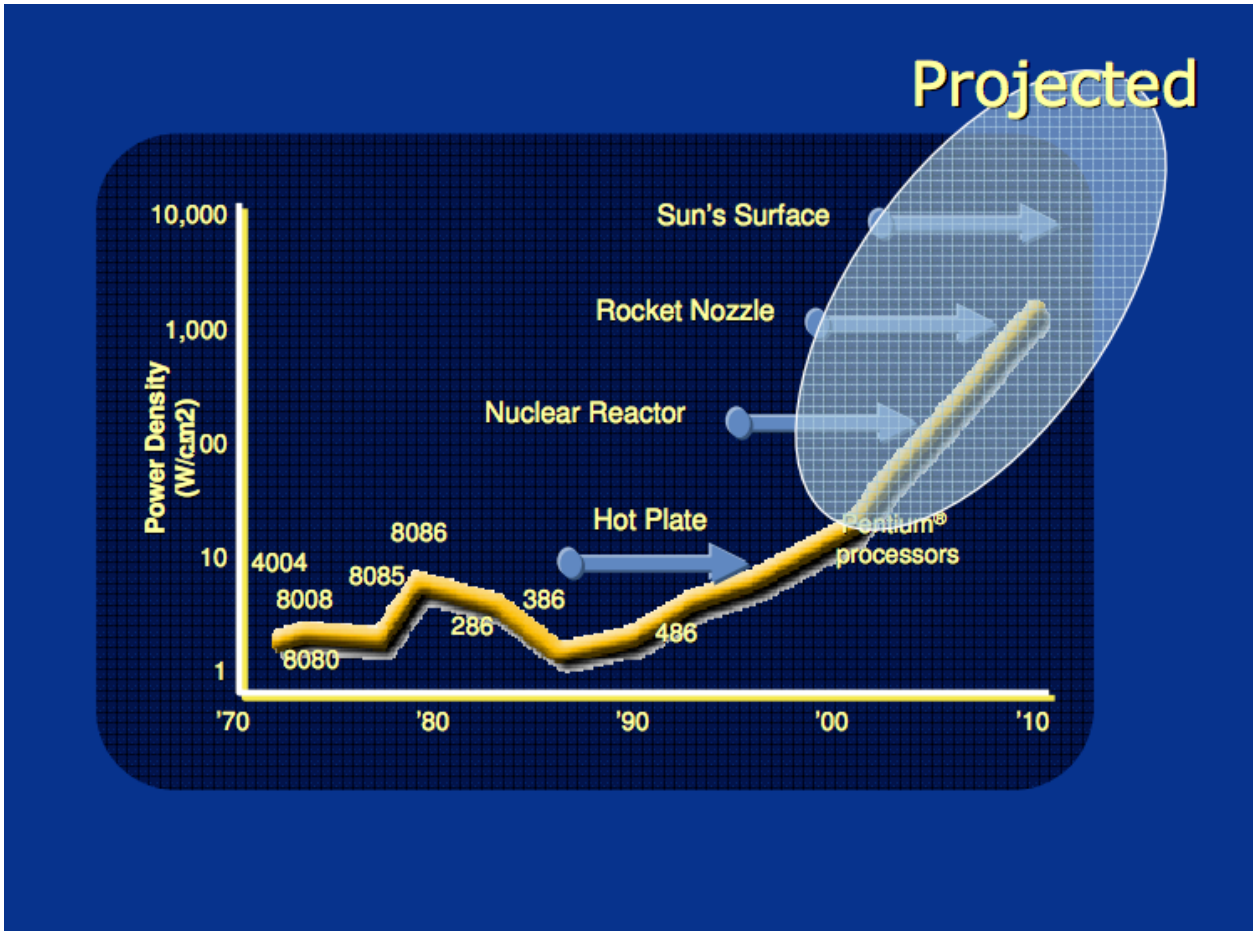
بر اساس قانون مور [۱] چگالی ترانزیستورهای تراشه‌های نیمه‌رسانا [۲] پس از گذشته به طور تقریبی هر هجدهماه، دو برابر می‌شود. در نتیجه می‌توان گفت که پردازنده‌ها کوچکتر، چگال‌تر و قدرتمندتر می‌شوند. بر اساس این قانون حداکثر فرکانس کاری پردازنده‌ها نیز قابل افزایش به نظر می‌رسد. اما به دلایل گوناگون روند افزایش فرکانس کاری پردازنده‌ها در سال‌های اخیر با کندی مواجه شده است. با ادامه روند افزایش فرکانس پردازنده‌ها، پیش‌بینی می‌شد که تا حوالی سال ۲۰۰۵ میلادی چگالی توان تراشه‌ها به سطح راکتورهای هسته‌ای برسد.

۱.۱ پردازنده‌های چندهسته‌ای

علی‌رغم پیشرفت چشمگیر قدرت محساباتی سخت‌افزارها و نیز نزدیک شدن به موانع عملی و فیزیکی برای افزایش فرکانس‌کاری آن‌ها، نیاز به بهبود عملکرد [۳] تراشه‌ها برای پشتیبانی از نیازمندی‌های جدید نرم‌افزار (به طور خاص رابط کاربری گرافیکی) و نیز انجام‌ها پردازش رو مجموعه‌داده‌های [۴] گسترده احساس می‌شد. در حدود سال ۲۰۰۰ میلادی اینتل با معرفی پردازنده‌های با معماری NetBurst انتظار دستیابی به فرکانس کاری 10Ghz را داشت اما در عمل به علت مشکلات گرما و توان مصرفی عملیات این چیپ‌ها فرکانس‌های بالای 4Ghz بدون استفاده از سیستم‌های خنک‌کننده بزرگ و پیچیده (معمولا مبتنی بر آب) امکان‌پذیر نبود.

در این میان ایده پردازنده‌های چندهسته‌ای به عنوان راه‌حلی برای افزایش کارایی ارائه شد. با توجه به اینکه در این فرکانس کاری پردازنده افزایش نمیابد، پارامترهای توان مصرفی و در نتیجه چگالی توان نیز تغییر شگرفی نمی‌کنند ولی چنانچه محسابات را بتوان به قسمت‌هایی با قابلیت موازی‌سازی

[۱] Moore's Law
[۲] Semiconductor
[۳] Performance
[۴] Dataset



شکل ۱.۱: پیش‌بینی رشد چگالی توان پردازنده‌ها

تقسیم کرد می‌توان زمان اجرا را به طور قابل توجهی کاهش داد. میزان ایده‌آل بهبود عملکرد محاسبه در این حالت از قانون امدال [۵] پیروی می‌کند.

لازم به ذکر است که تا قبل از ظهور پردازنده‌های چندهسته‌ای با تعریف امروزی گاها از چند پردازنده‌ی مرکزی مستقل برای انجام محاسبت استفاده می‌شد. در این معماری هر پردازنده عملا یک چیپ جداگانه بود که به واسطه یک باس [۶] مشترک به مادربورد و حافظه اصلی متصل می‌شد. این استقلال فیزیکی پردازنده‌ها مشکلات مختلفی ایجاد می‌کرد که در معماری‌های جدیدتر با قراردادن چند هسته [۷] پردازشی داخل یک تراشه تا حدی برطرف شده است.

[۵] Amdahl's law
[۶] Bus
[۷] Core

۲.۱ پردازنده‌های گرافیکی

تاریخچه پردازنده‌های گرافیکی به حدود سال‌های دهه ۷۰ میلادی بازمی‌گردد، زمانی که واحدهای سخت‌افزاری جداگانه برای بهبود عملکرد رایانه در اجرای بازی‌ها استفاده می‌شد. نسخه‌های اولیه چنین پردازنده‌هایی چیزی عملا گسترش جزیی معماری پردازنده‌های برداری [۸] برای کاربردهای گرافیکی بودند. در چنین پردازنده‌هایی که به طور خاص برای پردازش سیگنال و داده‌های در قالب ماتریس و آرایه طراحی شده بودند، تعداد زیادی واحد ریاضیاتی [۹] به طور همزمان دستورات یکسانی را روی قسمت‌های مختلف داده ورودی اجرا می‌کردند. این رویکرد که به اصطلاح مدل *دستور واحد و داده‌های متفاوت* [۱۰] نامیده می‌شود به طور خاص در محاسبات جبر خطی [۱۱] سودمند است.

پردازنده‌های گرافیکی در ابتدا سخت‌افزارهایی با عملکرد ثابت [۱۲] بودند و امکان برنامه‌پذیری [۱۳] نداشتند. با افزایش قدرت پردازنده‌های گرافیکی طی نسل‌های متمادی و آشکار شدن پتانسیل این روش پردازش برای کاربردهایی خارج از حوزه گرافیک و بازی‌های رایانه‌ای، به مرور زمان قابلیت برنامه‌پذیری نسبی به این سخت‌افزارها اضافه شد و امروزه واسط‌های نرم‌افزاری سطح بالای قدرتمندی مانند کودا [۱۴] توسعه داده توسط شرکت Nvidia و OpenCL برای منظور پیاده‌سازی برنامه‌های موازی عام‌منظوره روی این تراشه‌ها در دسترس هستند.

یک پردازنده گرافیکی به طور معمول متشکل است از تعدادی چندپردازنده [۱۵] که به صورت موازی دستورات (نه لزوما یکسان) را اجرا می‌کنند. به عنوان مثال پردازنده گرافیکی Nvidia Tesla K40 بر پایه معماری Kepler از ۱۵ چندپردازنده جریانی [۱۶] هر کدام با ۱۹۲ هسته پردازشی عدد طبیعی [۱۷] ، ۶۴ هسته پردازشی عدد ممیزدار [۱۸] و ۳۲ واحد انتقال داده [۱۹] تشکیل یافته است. این هسته‌ها به ۱۲ مجموعه *دستور واحد و داده‌های متفاوت* تقسیم می‌شود که هریک می‌توانند به طور مستقل دستور واحدی را روی داده ورودی خود اجرا کند . همچنین هر چندپردازنده جریانی دارای یک واحد

[۸] Vector Processor
[۹] Arithmetic Logic Unit
[۱۰] Single Instruction, Multiple Data (SIMD)
[۱۱] Linear Algebra
[۱۲] Fixed-Function
[۱۳] Programmability
[۱۴] CUDA: Compute Unified Device Architecture
[۱۵] Multiprocessor
[۱۶] Streaming Multiprocessor (SM)
[۱۷] Integer
[۱۸] Floating Point
[۱۹] Load/Store Unit

حافظه مشترک [۲۰] میان تمام هسته‌های پردازشی خود است که برای ذخیره داده‌های مورد نیاز در پردازش فعلی و عدم نیاز به بارگیری [۲۱] آن‌ها از حافظه اصلی در حین اجرا استفاده می‌شود.

در چنین معماری که با نام *دستور واحد و ریسه‌های متعدد* [۲۲] بازاریابی می‌شود تعداد زیادی ریسه به طور همزمان روندهای اجرایی متفاوتی را روی داده اعمال می‌کنند. یک پردازنده گرافیکی با معماری کپلر می‌تواند ۲۸۸۰ ریسه را به صورت موازی اجرا کند.

شکل ۲.۱: ساختار داخلی یک چندپردازنده جریانی در معماری Kepler

به لطف وجود چنین ظرفیت موازی‌سازی گسترده‌ای در پردازنده‌های گرافیکی زمان اجرای محاسبات روی آن‌ها به شکل قابل توجهی کاهش می‌یابد. اما به دلایل مختلف مانند وابستگی‌داده‌ای

[۲۰] Shared Memory
[۲۱] Fetch
[۲۲] Single Instruction, Multiple Threads (SIMT)

بین دستورات، عدم قابلیت موازی‌سازی همه بخش‌های پردازش، رفتارهای مختلف [۲۳] ریسه‌ها و در دستورات تصمیم‌گیری [۲۴] و همچنین تاخیرهای دسترسی به حافظه کارای پردازنده گرافیکی در شرایط واقعی بسیار پایین‌تر از مقدار آن روی کاغذ است.

۳.۱ ساختار پایان‌نامه

در ادامه و پس از پایان مقدمه، در فصل دوم مفاهیم پایه‌ای در طراحی پردازنده‌های چندهسته‌ای و به خصوص پردازنده‌های گرافیکی و به خصوص ساختاربندی حافظه در این تراشه‌ها را بررسی می‌کنیم. فصل سوم مروری بر کارهای پیشین در موضوع پایان‌نامه خواهد بود. در فصل چهارم شهود و انگیزه‌های اولیه برای این مطالعه را بررسی می‌کنم فصل پنجم را به معرفی شبیه‌ساز و بررسی نتایج حاصل از شبیه‌سازی اختصاص می‌دهیم. در نهایت در فصل ششم به جمع‌بندی و بررسی کارهای آتی می‌پردازیم.

[۲۳] Divergence
[۲۴] Decision Making

۲ مفاهیم پایه

در این فصل ابتدا مروری کلی بر مفاهیم پردازنده‌های چندهسته‌ای و دلایل روی‌آوردن به این پردازنده‌ها خواهیم داشت. در ادامه به بیان کلیاتی در مورد پردازنده‌های گرافیکی و معماری و مدل برنامه‌سازی آن‌ها خواهیم پرداخت.

۱.۲ دلایل روی آوردن به پردازنده‌های چندهسته‌ای

زمانی که اندازه مشخصه [۱] تزانزیستورها با فاکتور k کاهش می‌یابد، به دلیل کوتاه‌تر شدن سیم‌ها و کاهش اندازه خازن در گیت‌ها [۲] ، فرکانس کلاک [۳] نیز با فاکتور k قابل افزایش است. همچنین تعداد ترانزیستورهای موجود در واحد سطح با فاکتور x^2 و اندازه قالب [۴] ترازیستورها نیز با فاکتور k قابل افزایش می‌یابد. در چنین شرایطی قدرت پردازشی نیز به صورت تئوری با فاکتور k^4 افزایش می‌یاد. هرچند در عمل به دلیل مواردی مانن توازی پنهان [۵] یا رفتار غیرقابل پیش‌بینی حافظه نهان x^3 این فاکتور به طور عملی در مرتبه x^3 افزایش می‌یابد. به این نسبت‌ها قانون دانار گفته می‌شود [۶] . با این اوصاف به نظر می‌رسد که با معرفی هر نسل جدید پردازنده‌ها با اندازه مشخه کوچکتر باید شاهد بهبود شگرف در عملکرد نرم‌افزارها باشیم. اما در عمل این رشد با موانعی روبه‌روست که در ادامه به آن‌ها می‌پردازیم.

---

[۱] Feature Size
[۲] Gate
[۳] Clock
[۴] Die
[۵] $HiddenParallelism$
[۶] Dannar Scaling (MOSFET Scaling)

۱.۱.۲ چگالی توان

چگالی توان [۷] به شکل میزان توان (نرخ انتقال انرژی) بر واحد حجم تعریف می‌شود. میزان مصرف توان در تراشه‌ها به نرخ تغییر وضعیت گیت‌ها، یعنی نرخی که در آن خروجی یک گیت از صفر به یک تغییر می‌کند، بستگی‌دارد. به این دلیل به اصطلاح گفته می‌شود که تراشه‌ها نرخ توان مصرفی پویا دارند. با توجه به توضیح فوق انتظار می‌رود که نرخ توان مصرفی با افزایش فرکانس به طور خطی افزایش پیدا کند. با توجه به افزایش نمایی فرکان پردازنده‌ها در سال‌های پایانی دهه ۹۰ و اوایل قرن ۲۱ام، انتظار می‌رفت چگالی توان پردازنده‌ها در صورت حفظ این نرخ رشد تا سال ۲۰۱۰ میلادی به ۱۰۰۰۰ وات بر سانتی‌متر مربع یعنی چیزی در حدود چگالی توان در سطح خورشید برسد. واضح است که تراشه‌های نیمه رسانا در چنین وضعیتی تبخیر خواهند شد.

۲.۱.۲ دیوار حافظه

به طور معمول هر دسترسی به حافظه اصلی [۸] در حدود صدها سیکل کلاک زمان می‌برد. به طور مثال پردازنده ممکن است برای اجرای محاسبه‌ای که چند کلاک طول بکشد صدها کلاک منتظر دریافت داده و نوشتن مجدد آن در حافظه بماند. به طور میانگین در سال‌های گذشته فرکانس حافظه هر شش سال دوبرابر می‌شد در حالی‌که که در تبعیت از قانون مور فرکانس پردازنده هر دو سال دو برابر می‌شد. این تفاوت در نرخ رشد سبب ایجاد یک شکاف بزرگ میان عملکرد پردازنده و حافظه می‌شود و حافظه را به گلوگاهی [۹] برای عملکرد سیستم تبدیل می‌کند و تاثیر افزایش فرکانس پردازنده را به شدت کاهش می‌دهد. معماران سخت‌افزار با بهره‌گیری از ایده‌ها و روش‌های مختلف از جمله استفاده از چندین لایه حافظه نهان و بهینه سازی‌هایی مانند *بارگذاری با تاخیر* [۱۰] و *کپی هنگام نوشتن* [۱۱] سعی در مخفی کردن اثر سرعت حافظه از دید پردازنده دارند اما در نهایت مشکل همچنان باقی است. شکل ۲.۲ شکاف بین عملکرد پردازنده و حافظه اصلی را در سال‌های گذشته نشان می‌دهد.

[۷] Power
[۸] Random Access Memory (RAM)
[۹] Bottleneck
[۱۰] Lazy Writeback
[۱۱] Copy on Write

۲.۲ پردازنده‌های چندهسته‌ای به عنوان یک راه‌حل

در زمانی که افزایش فرکانس پردازنده‌ها دیگر ممکن به نظر نمی‌رسد، مهندسان ایده استفاده از چند پردازنده روی یک تراشه را برای بهبود عملکرد مطرح کردند. با توجه به اینکه کارایی یک پردازنده با فرکانس کاری و تعداد هسته‌های آن متناسب است، با افزایش تعداد هسته و ثابت نگه داشتن فرکانس می‌توانیم به عملکرد بهتری برسیم. با پذیرفته شدن این معماری توسط تولید کنندگان مطرح مانند Intel و AMD از آن به بعد:

- چگالی ترانزیستورها می‌تواند مانند قبل هر دو سال دو برابر شود
- فرکانس پردازنده‌ها افزایش نمیابد (بعضا شاهد کاهش فرکانس برای ملاحظات توان مصرفی هستیم)
- به جای دو برابر کردن فرکانس تمرکز روی دو برابر کردن تعداد هسته‌های پردازشی است

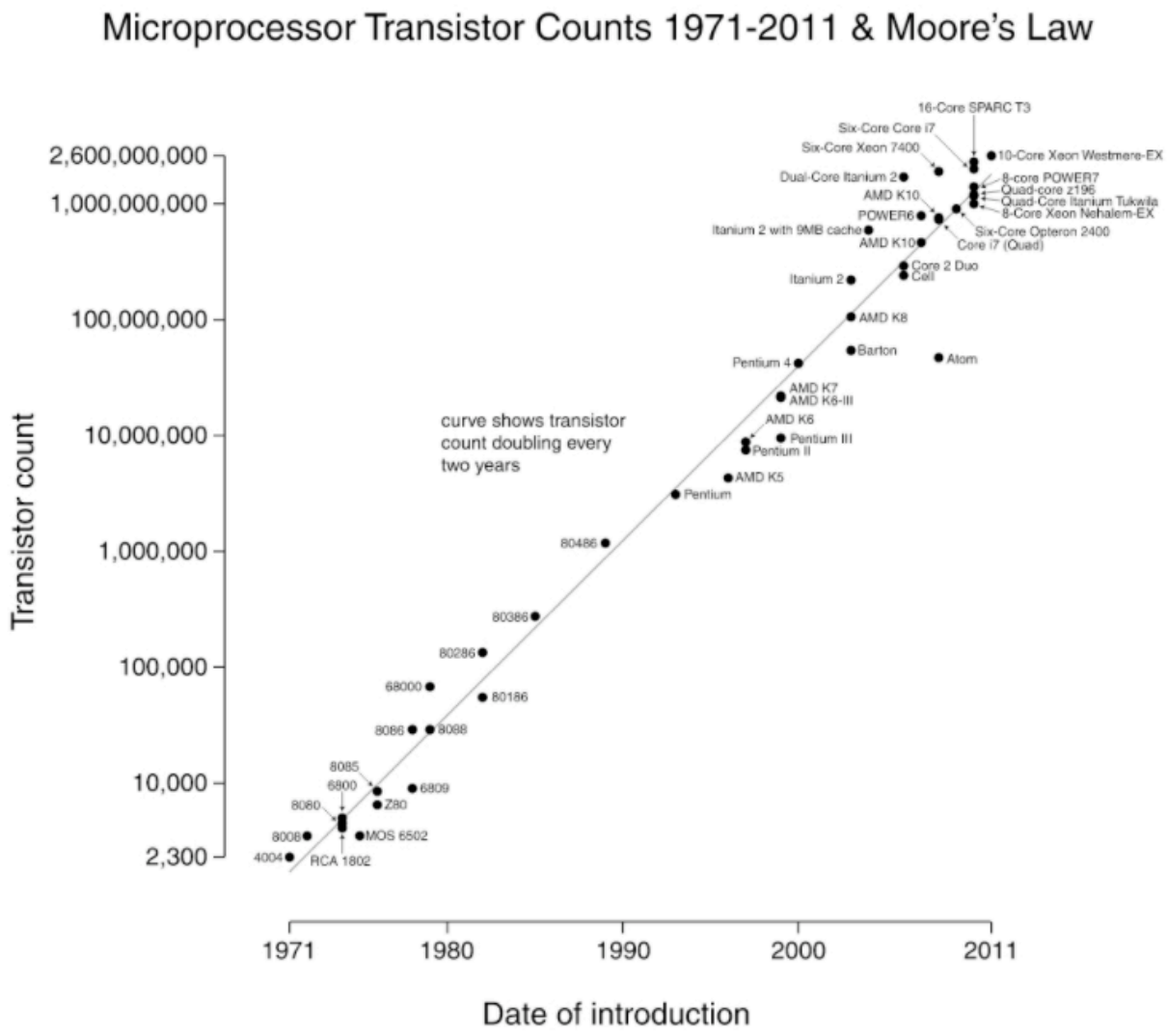
۳.۲ پردازنده‌های گرافیکی عام‌منظوره

به پردازنده‌های گرافیکی که قابلیت برنامه‌ریزی داشته باشد پردازنده گرافیکی عام‌منظوره [۱۲] گفته می‌شود. امروزه عمده کاربرد این پردازنده‌ها در محاسبات سنگین، شکستن رمزها، ارزهای رمزنگاری‌شده [۱۳] و شبیه‌سازی‌های علمی است.

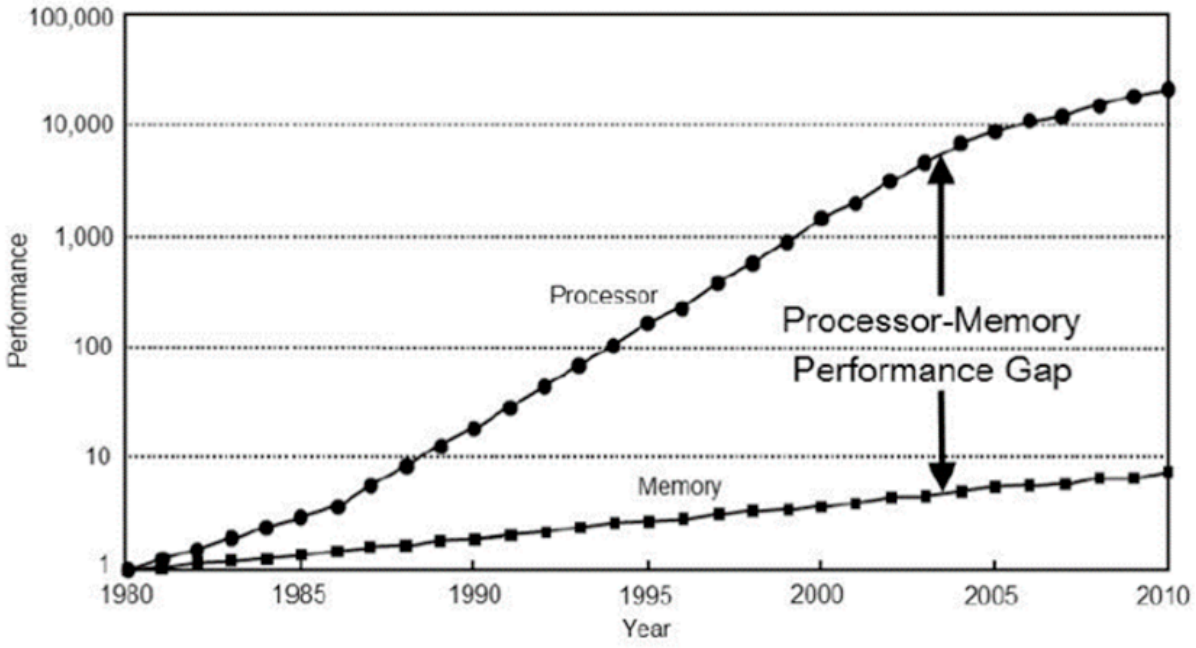
بر خلاف پردازنده‌های مرکزی که برای اجرای سیستم عامل و سوییچ کردن [۱۴] بین تعداد زیادی پردازه [۱۵] اجرا و پنهان کردن تاخیرهای حافظه برای حفظ پاسخگویی حداکثری طراحی شده‌اند، پردازنده‌های گرافیکی با هدف حداکثر سرعت در محاسبات تولید می‌شوند و بسیاری از پیچیدگی‌های داخلی پردازنده مرکزی را از معماری خود حذف می‌کنند. شکل ۴.۲ شکاف بین عملکرد این دو سخت‌افزار را در محاسبات روی اعداد ممیزدار نشان می‌دهد.

۴.۲ استفاده از پردازنده گرافیکی برای مسائل عام‌منظوره

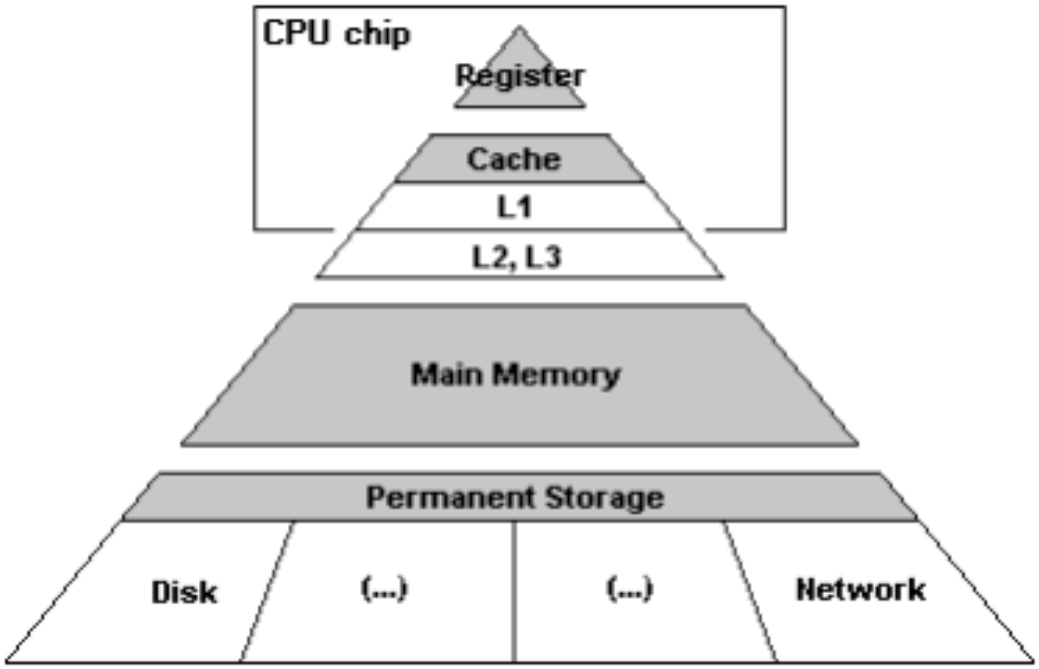
[۱۲] General Purpose Graphics Processing Unit
[۱۳] Cryptocurrency
[۱۴] Context Switching
[۱۵] Process



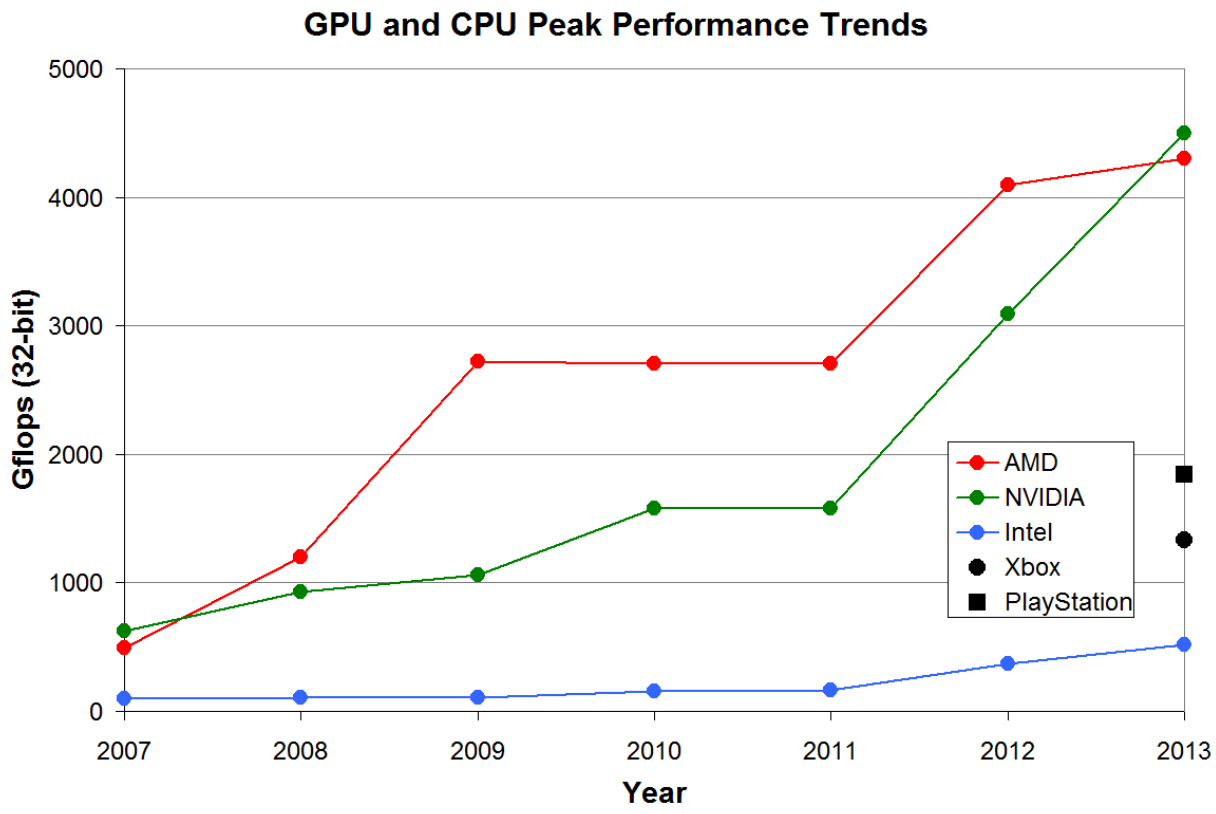
شکل ۱.۲: تبعیت رشد تعداد ترانزیستورها از قانون مور



شکل ۲.۲: شکاف بین عملکرد حافظه و پردازنده



شکل ۳.۲: سلسله مراتب حافظه در یک پردازنده امروزی



شکل ۴.۲: شکاف بین عملکرد پردازنده مرکزی و پردازنده گرافیکی در محاسبات عددی

۳ کارهای پیشین

۴ انگیزه و شهود

در این فصل ابتدا به بررسی وظایف حافظه مشترک و کاربردهای آن در پردازش‌های مختلف می‌پردازیم و سپس روش‌های پیشنهادی خود را برای اندازه‌گیری عملکرد آن تحت بارهای کاری علمی و نتایج به دست آمده را بررسی می‌کنیم.

در ادامه روشی برای بهبود عملکرد کلی تراشه گرافیک با تکیه بر تغییر معماری حافظه مشترک پیشنهاد می‌دهیم و شهودی برای تاثیرگذار بودن این رویکرد ارائه می‌کنیم.

۱.۴ حافظه چرک‌نویس

حافظه چرک‌نویس [۱] به حافظه‌ای اطلاق می‌شود که نتایج میانی محاسبات را نگهداری می‌کند. scratchpad معمولا نزدیک‌ترین واحد حافظه به ALU پس از رجیسترهاست و قادر به دسترسی مستقیم به حافظه اصلی [۲] است. از آنجا که عمده نتایج میانی در محاسبات سنگین در پایان دور ریخته می‌شوند استفاده از حافظه اصلی (و به تبع آن حافظه نهان) برای ذخیره‌سازی آن‌ها به علت سرعت کم و نیز احتمال تاثیر منفی بر سایر دستورات در حال اجرا (مصرف حافظه نهانی که می‌توانست به آن‌ها اختصاص پیدا کند) ضرورتی ندارد و در عوض از یک حافظه سریع‌تر داخلی به این منظور استفاده می‌شود.

مزیت دیگر این واحد حافظه زمان دسترسی قابل پیش بینی به آن است، زیرا داده قبل از رسیدن به پردازشگر از لایه‌های حافظه نهان عبور می‌کند و در زمان ثابتی قابل دسترسی است.

[۱] Scratchpad Memory
[۲] Direct Memory Access (DMA)