دانیال مبینی دهکردی - ۴۰۱۳۰۰۲۳

۱.۱)

الف)

Application Programs

Purpose:

طراحی شده برای کمک به کاربران در انجام وظایف یا فعالیت های خاص.

Examples:

پردازشگرهای کلمه (به عنوان مثال، مایکروسافت ورد)

مرورگرهای وب (به عنوان مثال، گوگل کروم)

بازی ها (مانند Minecraft)

سرویس گیرندگان ایمیل (به عنوان مثال، Microsoft Outlook)

User Interaction:

برای ارائه عملکرد به طور مستقیم با کاربران تعامل کنید.

Dependency:

برای عملکرد به برنامه های سیستم و سیستم عامل بستگی دارد.

System Programs

Purpose:

مدیریت و کنترل سخت افزار کامپیوتر و فراهم کردن بستری برای اجرای برنامه های کاربردی.

Examples:

سیستم عامل ها (مانند ویندوز، لینوکس)

درایورهای دستگاه (به عنوان مثال، درایورهای چاپگر)

ابزارهای کمکی (به عنوان مثال، ابزار پاکسازی دیسک)

ابزارهای مدیریت فایل (مانند Windows Explorer)

User Interaction:

معمولاً در پس‌زمینه اجرا می‌شود و مستقیماً با کاربران در تعامل نیست.

Dependency:

برای عملکرد برنامه های کاربردی و عملکرد کلی سیستم ضروری است.

به طور خلاصه، Application Programs برای کاربران نهایی برای انجام وظایف خاص طراحی شده اند، در حالی که System Programs سخت افزار را مدیریت می کنند و محیط لازم را برای اجرای برنامه های کاربردی فراهم می کنند.

ب)

Device Controller

Shape:

یک جزء سخت افزاری فیزیکی، اغلب به شکل یک برد مدار یا تراشه.

Connection:

Internal Controllers:

یکپارچه شده به مادربرد یا از طریق اسلات های داخلی (مانند PCI، PCIe) متصل می شوند.

External Controllers:

از طریق پورت های خارجی (به عنوان مثال، USB، Thunderbolt) متصل می شوند.

Device Driver

Shape:

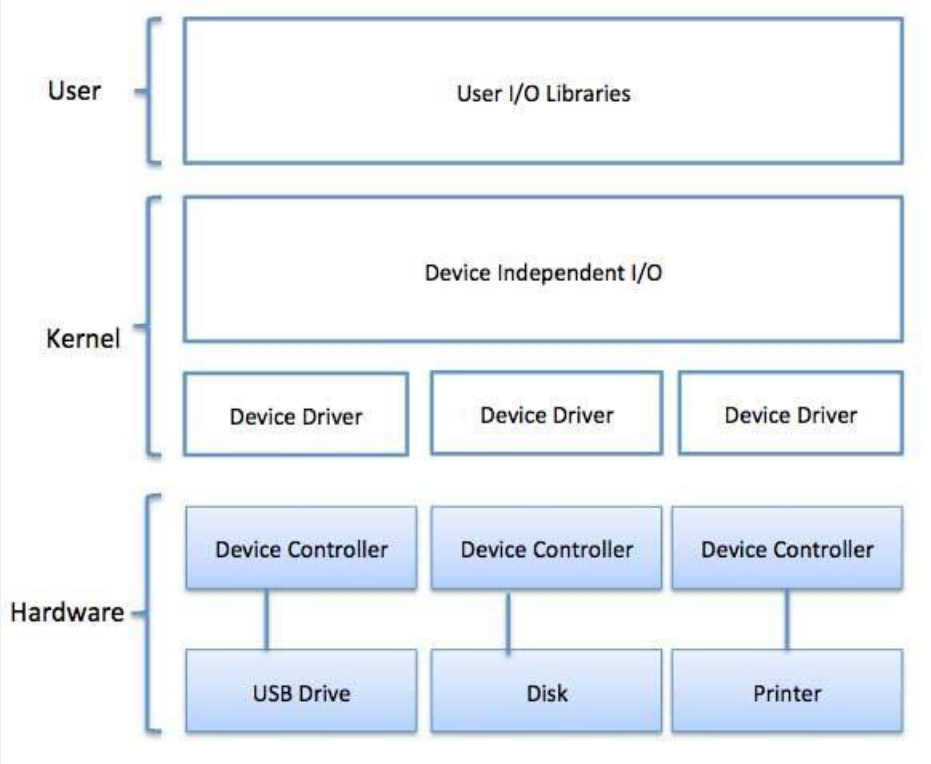
نرم افزار، بنابراین شکل فیزیکی ندارد.

Connection:

بر روی سیستم عامل نصب شده است.

از طریق هسته سیستم عامل با کنترل کننده دستگاه ارتباط برقرار می کند.

چگونگی قرار گیری و اتصال آن ها در سیستم:



ج)

Kernel Mode

Purpose:

دسترسی کامل به تمامی منابع سخت افزاری و سیستمی را فراهم می کند.

Privileges:

امتیازات سطح بالا که امکان تعامل مستقیم با سخت افزار و حافظه را فراهم می کند.

Components:

هسته سیستم عامل، درایورهای دستگاه و خدمات سیستم سطح پایین.

Operations:

می تواند هر دستور CPU را اجرا کند و به هر آدرس حافظه ارجاع دهد.

Stability:

خطا در حالت هسته می تواند کل سیستم را خراب کند.

Examples:

Operating system kernel

Device drivers

System calls

User Mode

Purpose:

محیطی محدود برای اجرای برنامه های کاربردی فراهم می کند.

Privileges:

امتیازات محدود، جلوگیری از دسترسی مستقیم به سخت افزار و منابع حیاتی سیستم.

Components:

برنامه های کاربردی و برخی از ابزارهای کمکی سیستم.

Operations:

فقط می تواند زیرمجموعه ای از دستورالعمل های CPU را اجرا کند و باید از تماس های سیستمی برای درخواست خدمات از هسته استفاده کند.

Stability:

خطاها در حالت کاربر مجزا هستند و معمولاً کل سیستم را تحت تأثیر قرار نمی دهند.

Examples:

Web browsers

Word processors

Games

Key Differences

Access Level:

Kernel Mode:

دسترسی کامل به سخت افزار و منابع سیستم.

User Mode:

دسترسی محدود، باید از تماس های سیستمی برای تعامل با سخت افزار استفاده کند.

Stability:

Kernel Mode:

خطاها می توانند کل سیستم را خراب کنند.

User Mode:

خطاها به برنامه جدا می شوند و ثبات سیستم را حفظ می کنند.

Execution:

Kernel Mode:

می تواند هر دستور CPU را اجرا کند و به هر آدرس حافظه دسترسی داشته باشد.

User Mode:

محدود به زیرمجموعه‌ای از دستورالعمل‌های CPU است و باید از تماس‌های سیستمی برای عملیات ممتاز استفاده کند.

دلایل استفاده از kernel mode و user mode :

Stability and Reliability

Security

Resource Management

Performance Optimization

Development and Maintenance

System Integrity

۲.۱)

الف)مراحل و مکانیسم های وقفه

مراحل:

درخواست وقفه: یک دستگاه سیگنال وقفه را به پردازنده می دهد.

تأیید وقفه: پردازنده وقفه را تأیید می کند و اجرای فعلی را متوقف می کند.

مدیریت وقفه: پردازنده یک کنترل کننده وقفه یا روال سرویس وقفه (ISR) را اجرا می کند.

تکمیل: ISR کامل می شود و پردازنده کار قطع شده را از سر می گیرد.

سیستم‌های مدرن از تکنیک‌های مدیریت وقفه پیشرفته مانند ادغام وقفه و اولویت‌بندی برای بهبود کارایی و کاهش تأخیر استفاده می‌کنند.

ب)

DMA برای انتقال داده ها بین حافظه و تجهیزات جانبی بدون دخالت CPU، کاهش سربار CPU و افزایش کارایی انتقال داده استفاده می شود.

ج)

پردازنده DMA Controller را با نوشتن در رجیسترهای آن، مشخص کردن آدرس مبدا و مقصد، جهت انتقال داده و مقدار داده برای انتقال تنظیم می کند.

DMA controller فرآیند انتقال داده را به عهده می گیرد و داده ها را مستقیماً بین دستگاه و حافظه جابجا می کند.

د)

DMA controller پس از تکمیل انتقال داده ها، وقفه ای را به پردازنده ارسال می کند که نشان دهنده پایان عملیات DMA است.

۳.۱)

الف)

مراحل اسکن یک عکس با حجم کم

شروع اسکن:

CPU دستوری را برای شروع اسکن به اسکنر ارسال می کند.

انتقال داده:

اسکنر تصویر را می گیرد و داده ها را به CPU ارسال می کند.

پردازش CPU:

CPU داده های دریافتی را پردازش کرده و در حافظه ذخیره می کند.

تکمیل:

CPU سیگنال تکمیل فرآیند اسکن را می دهد.

مراحل اسکن یک عکس بزرگ با استفاده از DMA

شروع اسکن:

CPU دستوری را برای شروع اسکن به اسکنر ارسال می کند.

تنظیم DMA:

CPU کنترل کننده DMA را با منبع (بافر اسکنر) ، مقصد (حافظه حافظه) و اندازه داده هایی که قرار است منتقل شوند، پیکربندی می کند.

شروع انتقال DMA:

CPU به کنترل کننده DMA دستور می دهد تا انتقال را آغاز کند.

عملیات DMA:

کنترلر DMA انتقال داده را به عهده می گیرد و داده ها را مستقیماً از اسکنر به حافظه منتقل می کند بدون اینکه CPU درگیر شود.

وقفه در تکمیل:

هنگامی که انتقال کامل شد، کنترلر DMA یک وقفه به CPU ارسال می کند تا پایان عملیات را سیگنال دهد.

چگونه DMA تعداد دستورالعمل های اجرا را کاهش می دهد

بدون DMA:

CPU باید داده ها را از اسکنر بخواند، آنها را پردازش کند و در حافظه بنویسد که شامل دستورالعمل های متعدد برای هر بایت داده است.

با DMA:

CPU فقط باید کنترلر DMA را راه اندازی کند و پس از اتمام انتقال، وقفه را مدیریت کند.

خلاصه

استفاده از DMA برای انتقال داده های بزرگ به طور قابل توجهی تعداد دستورالعمل های اجرایی مورد نیاز با بارگذاری وظیفه انتقال داده از CPU به کنترل کننده DMA را کاهش می دهد. این به CPU اجازه می دهد تا در حین انجام انتقال داده، وظایف دیگری را انجام دهد و کارایی و عملکرد کلی سیستم را بهبود بخشد.

ب)

بله

شرایطی که تحت آن تداخل ممکن است رخ دهد

Memory Contention:

ممکن است هر دو کنترلر CPU و DMA نیاز به دسترسی همزمان به حافظه داشته باشند که منجر به اختلاف می شود.

این می تواند باعث تاخیر شود زیرا یکی باید منتظر بماند تا دیگری دسترسی به حافظه خود را کامل کند.

Bus Contention:

هر دو کنترلر CPU و DMA گذرگاه سیستم را برای ارتباط با حافظه و تجهیزات جانبی به اشتراک می گذارند.

درخواست‌های دسترسی همزمان می‌تواند منجر به مشاجره اتوبوس شود و باعث تاخیر شود.

Cache Coherency:

اگر CPU به داده‌های کش دسترسی داشته باشد که توسط DMA نیز منتقل می‌شوند، مشکلات انسجام حافظه پنهان ممکن است ایجاد شود.

این می تواند منجر به استفاده از داده های قدیمی توسط CPU یا DMA شود.

اولویت بین DMA و CPU برای دسترسی به حافظه

اولویت DMA:

در بسیاری از سیستم ها، برای دسترسی به حافظه، کنترلر DMA نسبت به CPU اولویت بیشتری دارد.

دلیل: عملیات DMA معمولاً برای انتقال داده با سرعت بالا استفاده می شود که باید به سرعت تکمیل شود تا عملکرد سیستم حفظ شود و داده ها از دست نرود. اولویت دادن به DMA کارآمد و به موقع بودن این انتقال ها را تضمین می کند.

چرا DMA اولویت بالاتری دارد؟

کارایی:

DMA به گونه ای طراحی شده است که بلوک های بزرگ انتقال داده را کارآمدتر از CPU مدیریت کند.

اولویت بندی DMA زمان مورد نیاز برای این انتقال ها را به حداقل می رساند و گذرگاه و حافظه را برای سایر عملیات آزاد می کند.

تخلیه CPU:

با دادن اولویت بیشتر به DMA، CPU می‌تواند به اجرای وظایف دیگر بدون گرفتار شدن در عملیات انتقال داده ادامه دهد.

این عملکرد کلی سیستم و پاسخگویی را بهبود می بخشد.

الزامات Real-Time:

برخی از تجهیزات جانبی، مانند دستگاه های صوتی و تصویری، به انتقال داده ها در زمان واقعی نیاز دارند.

اولویت بندی DMA تضمین می کند که این الزامات بلادرنگ برآورده می شوند و از از دست رفتن داده ها جلوگیری می کند و کیفیت خدمات را حفظ می کند.

خلاصه

در حالی که عملیات همزمان توسط CPU و DMA می تواند به دلیل اختلاف حافظه و گذرگاه با یکدیگر تداخل داشته باشد، DMA اغلب برای دسترسی به حافظه اولویت بیشتری دارد. این اولویت‌بندی، انتقال کارآمد و به‌موقع داده‌ها را تضمین می‌کند، CPU را تخلیه می‌کند، و نیازهای بلادرنگ را برآورده می‌کند، و در نهایت عملکرد کلی سیستم را بهبود می‌بخشد.

۴.۱)

سیستم های چند پردازنده ای:

مزایا:

مقیاس پذیری، تحمل خطا، پردازش موازی.

معایب:

هزینه بیشتر، مصرف انرژی، تاخیر، پیچیدگی.

سیستم های چند هسته ای:

مزایا:

مقرون به صرفه، کم مصرف، تاخیر کم، طراحی فشرده.

معایب:

چالش های مدیریت حرارتی، اختلاف منابع، مقیاس پذیری محدود، پیچیدگی.

۱.۲)

الف)

System Calls

Process Control

مثال:

Fork

کاربرد: با کپی کردن فرآیند فراخوانی، یک فرآیند جدید ایجاد می کند. برای ایجاد پردازش های فرزند در سیستم عامل های مشابه یونیکس استفاده می شود.

File Management

مثال:

Open

کاربرد:یک فایل را باز می کند و یک توصیفگر فایل را برمی گرداند. برای دسترسی به فایل ها برای خواندن، نوشتن یا هر دو استفاده می شود.

Device Management

مثال:

ioctl

کاربرد:عملیات ورودی/خروجی خاص دستگاه را انجام می دهد. برای کنترل دستگاه های سخت افزاری استفاده می شود.

Information Maintenance

مثال: getpid

کاربرد:شناسه فرآیند فراخوانی را برمی‌گرداند. برای به دست آوردن اطلاعات مربوط به فرآیند استفاده می شود.

Communication

مثال: pipe

کاربرد:یک کانال داده یک طرفه ایجاد می کند که می تواند برای ارتباطات بین فرآیندی استفاده شود. برای انتقال داده بین فرآیندها استفاده می شود.

Protection

مثال: chmod

کاربرد: مجوزهای یک فایل را تغییر می دهد. برای تنظیم حقوق دسترسی برای فایل ها و دایرکتوری ها استفاده می شود.

ب)

fork (): یک فرآیند جدید با کپی کردن فرآیند فراخوانی ایجاد می کند.

exit (): فرآیند فراخوانی را خاتمه می دهد و وضعیت خروج را برمی گرداند.

chmod (): مجوزهای یک فایل یا دایرکتوری را تغییر می دهد.

ج)

scanf (): ورودی فرمت شده را با استفاده از فراخوانی سیستم read () می خواند.

printf (): خروجی فرمت شده را با استفاده از فراخوانی سیستم write () می نویسد.

malloc (): حافظه را به صورت پویا با استفاده از فراخوانی سیستم brk () یا mmap () تخصیص می دهد.

fopen (): فایلی را با استفاده از فراخوانی سیستمی open () باز می کند و ممکن است از fstat () برای اطلاعات فایل استفاده کند.

۲.۲)

الف)

مقایسه معماری های Microkernel, Modular, and Monolithic

معماری میکروکرنل: مینیمالیستی، انعطاف پذیری بالا، راندمان پایین به دلیل سربار IPC.

معماری مدولار: رویکرد متعادل، انعطاف پذیری خوب، کارایی متوسط.

معماری یکپارچه: راندمان بالا، انعطاف پذیری کمتر، ایزولاسیون ضعیف خطا.

ب)

در یک ساختار میکروکرنل، برنامه‌های کاربر از طریق ارسال پیام و مکانیسم‌های IPC با تماس‌های سیستم ارتباط برقرار می‌کنند که شامل مراحل متعددی برای اطمینان از ماژولار بودن و جداسازی خطا است.

۳.۲)

الف)

Emulator vs. VM: شبیه سازها محیط های سخت افزاری/نرم افزاری مختلف را تقلید می کنند، در حالی که ماشین های مجازی سیستم های کامپیوتری کامل را مجازی می کنند.

ب)

اجرای برنامه های ویندوز در لینوکس: از emulation (به عنوان مثال، Wine) برای ترجمه تماس های Windows API به لینوکس استفاده می کند.

ج)

Docker: یک پلتفرم کانتینری‌سازی که مزایایی را نسبت به VMها ارائه می‌کند، از جمله سبک وزن، عملکرد، کارایی منابع، قابلیت حمل و استقرار سریع‌تر.

۴.۲)

الف)

LKM: ماژول کرنل قابل بارگیری، عملکرد هسته را در زمان اجرا گسترش می دهد.

ب)

System Call vs. LKM: تماس‌های سیستمی نیاز به کامپایل مجدد هسته و راه‌اندازی مجدد دارند، در حالی که LKMها را می‌توان به صورت پویا و بدون راه‌اندازی مجدد بارگیری/دانلود کرد.

ج)

lsmod

د)

ioctl in LKM: به برنامه های فضای کاربر اجازه می دهد تا دستورات کنترلی را به درایورهای دستگاه ارسال کنند.

ه)

insmod and rmmod: به ترتیب ماژول های هسته را وارد و حذف کنید.

و)

printf vs. printk: برای خروجی فضای کاربر است، printk برای ثبت فضای هسته است.