امیر محمّد آقاپور ۹۶۳۱۰۰۸ تمرین ۲

.۱

الف) حالتی که در آن پردازنده بی کار(idle) است و دستوری انجام (execute) نمی دهد و تنها بخش مربوط به تشخیص وقفه ها و دیکود از الگوریتم فون نایمن انجام می شود)

ب) با اجرای دستور مربوطه در ISA توسط سیستم عامل، چون این دستور در kernel mode اجرا می شود. با وقفه خارجی (external interrupt) نیز از آن خارج می شود که میتواند توسط سخت افزار خارجی تولید شود.

ج) به طور کلی وقتی scheduler سیستم عامل، پروسسی برای انجام نداشته باشد به این حالت می رود. مانند وقتی که پردازنده منتظر I/O است و کار دیگری برای انجام ندارد، برای کاهش توان مصرفی و دمای پردازنده به این حالت میرود.

الف)

استفاده از interrupt-chaining به این صورت که برای هر چند I/O یک generic interrupt routine در نظر بگیریم(مشابه برداری) که در صورتی که یکی از آن ها درخواست وقفه داد، تک تک آن ها را بررسی کند.(مشابه سر کشی)

ب)

سرکشی: وقتی دستگاه I/O در فاصله زمانی بسیار کم آماده باشد (حالت busy کوتاه یا به طور مکرر درخواست داشته باشد) یا رسیدگی به در خواست I/O ها مهم(critical) باشد؛ مانند از دست رفتن داده(پر شدن بافر) یا صدمه به دستگاه(مشکل در برق یا دما).

وقفه: وقتی فاصله زمانی میان مواقعی که I/O آماده است زیاد باشد، همچنین I/O ها حیاتی نباشند.

روش بیان شده: وقتی تعداد I/O ها زیاد و تأثیر سربار برداری (که فضای اشغالی است) زیاد می شود.

.۲

سخت افزاری: ایجاد مشکل در منبع تغذیه، ارور خواندن از حافظه(ارور parity به طور مثال)، درخواست I/O (کار با کیبورد و موس)

نرم افزاری: فراخوانی سیستمی(system call) برای درخواست انجام کاری توسط سیستم عامل از طرف برنامه کاربر، تولید ارور (trap یا exception) در برنامه (مانند تقسیم بر صفر یا ورودی غیر منتظره)

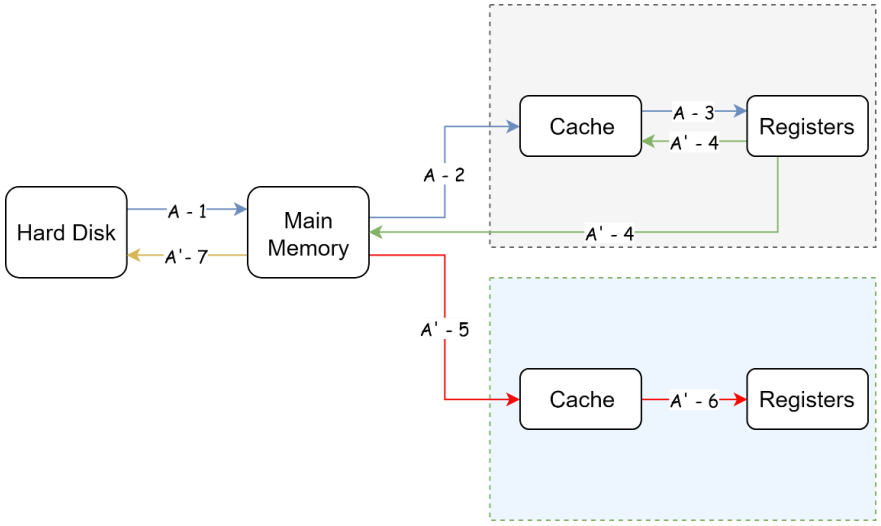
.۳

الف) ایجاد مکانسیمی برا اولویت دهی به وقفه ها (interrupt priority leveling) در سیستم عامل. این سیستم میتواند این گونه باشد که رسیدگی به درخواست کاربر ۱ یا ۲ حین رسیدگی به درخواست کاربر ۳ انجام شود و همچنین رسیدگی به درخواست کاربر ۱ حین رسیدگی به درخواست کاربر ۲ نیز انجام شود.(nested interrupt)

ب) در صورتی که کاربر با اولویت بالاتر مدام قبل از اتمام رسیدگی کامل به کاربر با اولویت پایین تر در خواست دهد. در این صورت درخواست کاربر اولویت پایین تر هیچ وقت انجام نمی شود.

۴.

بهتر است حافظه اصلی write-back و کش write-through باشد(با فرض UMA و اینکه کش بین پردازنده ها مشترک نیست)؛ یعنی همزمان با اعمال تغییر در کش پردازنده اوّل، A' در حافظه اصلی هم نوشته شود تا پردازنده دوم بتواند از مقدار آن استفاده کند. چون در دیسک سخت نوشته نشد، سرعت چندان کاهش نمیابد (نوشتن در دیسک سخت بعد از اتمام برنامه یا نیاز به جایگزین شدن داده آپدیت شده در رم انجام می شود). لازم به توضیح است، پردازنده اوّل بعد از خواندن A باید بیت dirty آن را در رم set کند تا پردازنده دوم بداند داده تغییر کرده ولی آپدیت نشده و در صورت لزوم منتظر بماند که نسخه آپدیت شده (یعنی A') در حافظه اصلی نوشته شود و همچنین در پایان داده در هارد آپدیت شود.



(لیبل های روی فلش ها نشان دهنده داده و زمان جابجایی هستند)

۵.

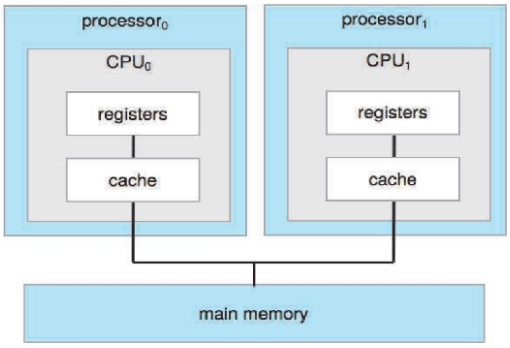
در مواقع انجام اعمال I/O با داده های حجیم بین حافظه اصلی و جانبی، به جای استفاده از وقفه برای جابجایی هر بلوک داده از حافظه جانبی به پردازنده و از آن جا به رم، با مشخص کردن بافر و نشانگر(pointer) و شمارنده برای I/O، کنترل کننده دستگاه می تواند داده ها را مستقیم و بدون دخالت پردازنده به رم منتقل کند(البته چون رابطِ رم مشترک با پردازنده است باید برای جاجایی هر بسته چند کلاک باس را قرض بگیرد (cycle stealing) که در این مدّت اگر پردازنده رم را بخواهد باید منتظر بماند) و در پایان نیز برا مطّلع کردن پردازنده از اتمام فرآید، یک وقفه تولید کند. چون همزمان با این جابجایی داده، پردازنده می تواند به اجرای دستورات دیگر بپردازد و با داده های کش خود کار کند، استفاده از DMA باعث افزایش همروندی می شود.

۶.

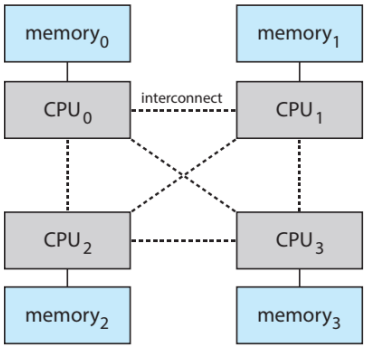
برای افزایش بهره وری در سیستم های چندپردازنده ای از NUMA استفاده می شود.

در UMA تمام پردازنده ها دسترسی یکنواخت به حافظه اصلی دارند. بدلیل وجود باس مشترک برای دسترسی به این حافظه، پردازنده ها باید منتظر هم بمانند. پس با افزایش تعداد پردازنده ها کارایی پایین می آید.

در NUMA، برای رفع این مشکل و افزایش بهره وری در این سیستم ها، از حافظه های جدا گانه(محلّی) برای هر پردازنده استفاده شده و پردازنده ها به هم متّصل می شوند. سرعت دسترسی هر پردازنده به حافظه محلّی خود بیشتر و بدون انتظار است. مشکل NUMA آن است که سرعت دسترسی یک پردازنده به حافظه محلی پردازنده دیگر کم است و باعث کاهش بهره وری کلّ سیستم میشود، که البته با مدیریت دستورات اجرا شده در هر پردازنده توسط سیستم عامل تأثیر این مشکل را می توان کم کرد.



UMA multiprocessing architecture.



NUMA multiprocessing architecture.

۷.

(با فرض واحد های SI)

تعداد بلوک در یک ثانیه $\frac{10\ MB/s}{4\ KB} = 2.5\ K/s$

تعداد کلاک انجام تنظیمات DMA در ثانیه $2.5\ K/s \times 5000 = 12.5\ M/s$

درصد زمان سپری شده توسط پردازنده برای انتقال $\frac{12.5\ M/s}{500\ MHz} = 2.5\%$

(اگر واحد های داده ها بر حسب کیبی بایت و مبی باید باشد، پاسخ نهایی $\frac{2.5\times2^{10}\times5000}{500\times10^{6}} = 2.56\%$ خواهد شد)