پروژه چهار

اعضای گروه:

پارسا علیزاده - 810101572

نیلوفر مرتضوی - 220701096

محمدرضا خالصی - 810101580

Repository: https://github.com/pars1383/OS-xv6-public

Last commit: 5d5dbad7809b25b47db5fc5ea9cb0f8bfa41f067

1. **علت غیر فعال کردن وقفه چیست؟ توابع pushcli و popcli به چه منظور استفاده شده و چه تفاوتی با cli و sti دارند؟**

- **علت غیر فعال کردن وقفه:** وقفه ها در هنگام اجرای بخش های حساس کد (Critical Sections) غیرفعال می شوند تا از بروز شرایط رقابتی (Race Conditions) جلوگیری شود. اگر یک وقفه در حین دستکاری یک داده مشترک توسط کد اصلی رخ دهد و روال سرویس وقفه (Interrupt Service Routine) نیز سعی در خواندن یا تغییر همان داده داشته باشد، ممکن است داده دچار ناسازگاری یا خرابی شود. غیرفعال کردن وقفه ها تضمین می کند که بخش حساس کد به صورت اتمیک نسبت به سایر وقفه ها اجرا شود و تداخلی پیش نیاید.

- **توابع pushcli و popcli در مقابل cli و sti:**

  - cli (Clear Interrupt Flag): این دستور وقفه ها را بر روی پردازنده فعلی غیرفعال می کند.

- sti (Set Interrupt Flag): این دستور وقفه ها را بر روی پردازنده فعلی فعال می کند.
- pushcli: این تابع ابتدا وضعیت فعلی وقفه ها (فعال یا غیرفعال بودنشان) را ذخیره می کند و سپس وقفه ها را غیرفعال می نماید. این کار برای مدیریت صحیح وقفه ها در بخش های حساس تودرتو (Nested Critical Sections) ضروری است. اگر وقفه ها از قبل غیرفعال بوده باشند، pushcli این وضعیت را ثبت کرده و همچنان آنها را غیرفعال نگه می دارد (یا به بیان دقیق تر، یک شمارنده را افزایش می دهد که نشان دهنده نیاز به غیرفعال ماندن وقفه ها است).
- popcli: این تابع وضعیت وقفه ها را به حالتی که قبل از pushcli متناظرش بوده اند، بازمی گرداند. popcli به سادگی وقفه ها را فعال نمی کند، بلکه تنها در صورتی آنها را فعال می کند که قبل از pushcli فعال بوده باشند و این popcli بیرونی ترین فراخوانی در یک زوج تودرتو باشد. این مکانیزم امکان استفاده ایمن از بخش های حساس تودرتو را فراهم می کند. اگر وقفه ها قبل از pushcli فعال بوده باشند، popcli آنها را فعال می کند. اگر غیرفعال بوده باشند، تا زمانی که تعداد فراخوانی های pushcli با popcli برابر نشده و وضعیت اولیه "فعال" نبوده باشد، وقفه ها غیرفعال باقی می مانند.

2. **حالات مختلف پردازه ها در XV6 را توضیح دهید. تابع sched چه وظیفه ای دارد؟**

    - **حالات مختلف پردازه ها در XV6:**

- UNUSED: این مدخل در جدول پردازه ها استفاده نشده و خالی است.
- EMBRYO: پردازه در حال ایجاد شدن است (معمولاً پس از فراخوانی fork).
- SLEEPING: پردازه منتظر وقوع یک رویداد خاص است (مانند اتمام عملیات ورودی/خروجی، آزاد شدن یک قفل، یا فراخوانی wait توسط والد). در این حالت، پردازه قابل اجرا نیست.
- RUNNABLE: پردازه آماده اجرا است اما منتظر تخصیص CPU می باشد.
- RUNNING: پردازه در حال حاضر بر روی یک CPU در حال اجرا است.

- ZOMBIE: پردازه اجرای خود را به اتمام رسانده است (exit کرده)، اما والد آن هنوز با فراخوانی wait وضعیت خروج و منابع آن را جمع آوری نکرده است.

- **وظیفه تابع sched:** تابع sched قلب زمانبند (Scheduler) در XV6 است. وظیفه اصلی آن انتخاب یک پردازه از میان پردازه های در حالت RUNNABLE از جدول پردازه ها و سپس تعویض زمینه (Context Switch) پردازنده به آن پردازه است تا اجرای آن آغاز شود (و به حالت RUNNING درآید). این تابع زمانی فراخوانی می شود که یک پردازه CPU را رها می کند (مثلاً با به خواب رفتن، خروج، یا فراخوانی yield) یا زمانی که یک وقفه تایمر نشان می دهد که زمان تعویض پردازه فرا رسیده است. هر CPU حلقه زمانبند خود را اجرا می کند که این حلقه تابع sched را فراخوانی می نماید.

3. **در چه مواردی این قفل (خوانندگان-نویسندگان) نسبت به قفل بلیط مزیت دارد؟**

قفل خوانندگان-نویسندگان (Readers-Writers Lock) در موارد زیر نسبت به قفل بلیط (Ticket Lock) مزیت دارد:

- **سناریوهای با تعداد خواننده زیاد (Read-Heavy Scenarios):** مزیت اصلی قفل خوانندگان-نویسندگان این است که به چندین خواننده اجازه می دهد به طور همزمان به منبع مشترک دسترسی داشته باشند. اگر بار کاری سیستم عمدتاً شامل عملیات خواندن باشد، این نوع قفل می تواند به طور قابل توجهی کارایی و توان عملیاتی سیستم را در مقایسه با قفل بلیط بهبود بخشد. قفل بلیط تمامی دسترسی ها (چه خواندن و چه نوشتن) را سریال می کند و همه باید به نوبت منتظر بمانند.
- **کاهش رقابت برای خوانندگان:** در یک قفل خوانندگان-نویسندگان (بسته به سیاست اولویت دهی آن)، خوانندگان معمولاً مجبور نیستند منتظر سایر خوانندگان بمانند، مادامی که هیچ نویسنده ای قفل را در اختیار نداشته باشد یا منتظر گرفتن قفل نباشد. این امر منجر به کاهش زمان انتظار برای خوانندگان می شود.

4. در مقابل، قفل بلیط ترتیب FIFO (خروج به ترتیب ورود) را به شدت رعایت می کند و از قحطی (Starvation) برای هر پردازه جلوگیری می کند، اما این کار را به قیمت عدم تشخیص بین عملیات خواندن و نوشتن انجام می دهد. قفل خوانندگان-نویسندگان برای بهینه سازی دسترسی های همزمان خواندن طراحی شده است.