دانشکده مهندسی
کامپیوتر و فناوری اطلاعات

پروژه‌ی دوم

دکتر جوادی

سیستم‌های عامل

پروژه‌ی دوم

مقدمه

در این بخش قرار است با هم‌گروهی خود دو بخش جدید به سیستم عامل xv6 که در فاز اول توسعه دادید اضافه کنید. دقت کنید که لازم است از فاز بخش اول استفاده کنید. در ادامه لازم است تا به هر صورتی که برای دو نفر امکان دارد برای پیاده سازی بخش های لازم تقسیم کار کنید. در هر بخش نکات لازم برای پیاده سازی توضیح داده شده است و سپس نحوه‌ی پیاده‌سازی و اجرای فایل تست نیز مشخص شده است. برای تحویل کامل پروژه لازم است که فایل های تست به درستی اجرا شوند. در صورتی که فاز اول را هیچکدام از اعضا پیاده سازی نکرده اند آن را پیاده سازی کنید و یا میتوانید از کد فاز اول هم‌کلاسی های خود کمک بگیرید.

اضافه کردن time و cpu usage به دستور top:

در این قسمت می خواهیم به دستور تاپی که در فاز قبلی پروژه پیاده سازی کرده ایم مدت زمانی که هر پردازه ایجاد شده و درصد cup مورد استفاده هر پردازه را به ستون های این دستور اضافه کنیم.
محاسبه ی مدت زمان ایجاد پردازه و مدت زمانی که پردازه در حالت اجرا است:
برای این کار لازم است که با متغیر ticks درxv6 آشنا شوید. همانطور که میدانید هر سیستم عاملی برای جلو رفتن کارش در قسمت زمان بندی و ... به یک کالک نیاز دارد، متغیر ticks تعداد تیک هایی که تا کنون گذشته است را ذخیره میکند. در xv6 یک تیک مقدار زمانی است که زمانبند به هر پردازه اختصاص میدهد و بعد از تمام شدن آن یک برنامه دیگر اجرا میشود. با کاربرد های بیشتر این متغیر در فاز سوم پروژه آشنا خواهیم شد. این متغیر در فای trap.c با آمدن وقفه تایمر آپدیت میشود.
برای پیاده سازی این بخش لازم است در ساختار proc struct یک فیلد جدید اضافه کرده و برای هر پردازه زمان ایجاد شدن آن را بر اساس ticks ذخیره کنید. توجه کنید از آنجایی که ممکن است چند پردازه به طور همزمان به این متغیر دسترسی پیدا کنند، حتما عملیات مربوط به این متغیر را به صورت اتمیک انجام دهید. سپس یک فیلد دیگر اضافه کنید که مدت زمانی که پردازه در حال اجرا است را ذخیره میکند. برای بروز رسانی این فیلد ها می توانید همه را در یک تابع پیاده سازی کنید.

تکمیل دستور top:

در این بخش لازم است به دستور top دو ستون اضافه کنید. ستون time که فیلد زمان شروع هر پردازه را پیدا کرده و اختلاف مقدار زمان ذخیره شده برای آن پردازه با مقدار متغیر ticks را نشان می دهد. دقت کنید که این عدد باید بر حسب ثانیه باشد.
ستون cpu٪ که درصد استفاده از cpu را تا دو رقم اعشار به فرمول مدت زمان اجرا به کل uptime سیستم نشان میدهد.

پروژه‌ی دوم

امتیازی:

در این بخش دستور top را به گونه ای پیاده سازی کنید که بعد از یک بار چاپ خارج نشود و تا وقتی دستور ctrl+c وارد نشده در یک دوره زمانی کوتاه صفحه را پاک کرده و اطلاعات جدید را چاپ کند (مانند سیستم عامل لینوکس).

تغییر الگوریتم زمانبندی:

ابتدا نحوه ی کارکرد تابع scheduler در فایل proc.c را بررسی کنید و سپس به بررسی توابع مربوط دیگر مانند sched و yield بپردازید و پیدا کنید که چگونه و کجا timer interrupt ایجاد میشود و handle میشود. در بررسی خود به نحوه کارکرد توابع و موارد استفاده و جایی که صدا می شوند بپردازید. سپس بررسی کنید که زمانبند پیش فرض xv6 از چه نوع است و چگونه کار می کند. در ادامه مشخص کنید که چگونه یک پردازه به پایان می رسد و از کدام توابع برای این بخش استفاده میشود. در آخر روش کارکرد یک زمانبند چند صفی را مطالعه کنید.
دقت کنید موارد مورد بررسی در هنگام تحویل پروژه مورد سوال قرار میگیرند.

در این بخش قرار است تا زمانبند پیش فرض را تغییر داده و یک زمانبند چند صفی پیاده سازی کنید.

cpu :

در این بخش لازم است برای درک بهتر کارکرد پیاده سازی خود تعداد پردازنده ها را به ۱ تغییر دهید. برای این کار به فایل Makefile مراجعه کنید.

Multi-level queue :

در این بخش که اکثر پیاده سازی این فاز را شامل میشود، باید ابتدا یک مکانیزم پیاده سازی کنید که بتواند از الگوریتم چند صفی پشتیبانی کند. شما لازم دارید که صف پردازه ها را طوری تعریف کنید که پردازه ها بتوانند در ۳ صف ذخیره شوند. سپس هر جا که لازم شد استراکت proc را طوری تغییر دهید که اگر لازم است اطلاعات مربوط به صفی که پردازه داخل آن هست را ذخیره کند.
هر صف از الگوریتم Round Robin استفاده میکند. صفی که بالا تر است دارای اولویت بیشتری است. یعنی برای مثال صف شماره یک اولویت بیشتری نسبت به صف شماره دو دارد. صف اول دارای کوانتوم زمانی ۵ و صف دوم دارای کوانتوم زمانی ۱۰، و صف سوم دارای کوانتوم زمانی ۲۰ است. پردازه ها در ابتدای ساخته شدن وارد صف اول می شوند.

پروژه‌ی دوم

در این بخش لازم است تا اگر فیلدی به استراکت proc اضافه کردید آن را در تابع مناسب مقداردهی اولیه کنید.

Multi-level queue scheduler :

در این بخش لازم است تا تابع scheduler را که قبلا بررسی کرده اید طوری تغییر دهید که بصورت چند صفی عمل کند و اولویت صف ها رعایت شوند.

Test :

در این بخش باید یک فایل تست در بخش user آماده کنید و ۱۰ پردازه در آن ایجاد کنید که هر کدام از ۱ تا ۱۰۰۰۰۰۰۰۰ شمارش میکند. سپس در پایان اجرا یک پیام چاپ کند که پردازه با شماره ی) nبین یک تا ده) تمام شد.

همچنین لازم است در هر بار اجرای پردازه که cpu به آن اختصاص داده میشود یک پیام مانند پیام قبل با متن مناسب چاپ کنید و هنگامی که از روی cpu برداشته شده و به صف منتقل میشود نیز یک پیام مناسب چاپ کنید. دقت کنید که در متن پیام ها از pid استفاده کنید. پیام هایی که هنگام اجرا و خروج توسط scheduler چاپ می کنید حتما حاوی اطلاعاتی که پردازه در کدام صف است نیز باشد.

پروژه‌ی دوم

از شما درخواست داریم که یک **private repository** در گیت هاب درست کنید و تغییرات کد خود را مرحله به مرحله **Commit** کنید و در صورت تمایل می توانید هر یک از تدریس یاران را به پروژه ی خود اضافه کنید. دقت کنید که شما نبایستی برنام ههای خود را با دیگر دانشجویان به اشتراک بگذارید.

توضیحات

- پروژه شما تحویل آنلاین خواهد داشت بنابراین از استفاده از کدهای یکدیگر یا کدهای موجود در وب که قادر به توضیح دادن عملکرد آنها نیستید، بپرهیزید.
- ابهامات خود را در گروه درس در تلگرام مطرح کنید و ما در سریعترین زمان ممکن به آنها پاسخ خواهیم داد .

آنچه که باید ارسال کنید:

یک فایل زیپ با نام OS_P2_Sid1_Sid2.zip (که Sid1 را با شماره دانشجویی خود و دیگری را با شماره دانشجویی هم‌گروهی خود جایگزین کنید) که شامل مورد زیر است:

- پوشه ای که در آن کدهای شما وجود دارد. دقت کنید که تنها و تنها فایل هایی را که تغییر داده اید یا اضافه کرده اید را برای ما بفرستید. آچلود یک نفر از اعضا کافی است.

موفق باشید

تیم تدریس یاری درس سیستم های عامل