به نام خدا


سیستم‌های عامل (بهار ۱۴۰۲)

# فاز اول پروژه

استاد درس:

دکتر جوادی

مهلت نهایی ارسال پروژه:

۱۸ فروردین ۱۴۰۲ ساعت ۲۳:۵۹

نکته مهم: دقت کنید که تمدید نخواهیم داشت و تحویل اسکایپی خواهید داشت و تنها دانشجویانی که فاز اول را به موقع انجام داده‌اند، خواهند توانست وارد فاز دوم شوند

# مقدمه

همانطور که در کلاس درس بیان شد، پروژه درس سیستم‌های عامل در مورد شناخت کامل سیستم عامل آموزشی xv6 و اضافه کردن قابلیت‌های جدید به آن است. با انجام دقیق این پروژه و نگاشت مفاهیم بیان شده در کلاس درس به معادل عملیاتی آنها، به یک یادگیری عمیق و ماندگار دست خواهید یافت. مهم‌تر اینکه برنامه‌نویسی در سطح سیستم عامل به شما کمک می‌کند تا یک تجربه بی‌نظیر ار برنامه‌نویسی سیستمی داشته باشید.

هدف از دو فاز ابتدایی پروژه آشنایی شما دانشجویان عزیز با این سیستم عامل آزمایشی است و روند کار به این صورت است که فاز اول به صورت انفرادی و فاز دوم به صورت گروهی  باید تغییراتی در این سیستم عامل ایجاد کنید تا بتوانید درک خوبی از این سیستم عامل بدست آورید. در فاز دوم و سوم از شما درخواست خواهیم کرد که به گروه‌های دو نفره تقسیم شوید و تغییراتی که در فاز سوم انجام خواهید داد بسیار مهم‌تر و جدی‌تر خواهند بود که موضوع این قسمت در زمان مناسب به شما اعلام خواهد شد. در فاز اول پروژه بایستی درک مناسبی از برخی مفاهیم از جمله پروسس‌ها، سیستم کال ها و عملکرد کلی سیستم عامل پیدا کنید و در نهایت دو سیستم کال ساده را به سیستم عامل خود اضافه کنید.

در این ترم با آخرین نسخه از سیستم عامل xv6 که بر پایه معماری پردازنده RISC-V توسعه داده شده است کار خواهیم کرد. برای آشنایی با این سیستم عامل از لینک زیر استفاده کنید.

https://pdos.csail.mit.edu/۶.۸۲۸/۲۰۲۲/xv۶.html

برای اجرای این سیستم عامل می‌توانید مراحل و پیش‌نیاز های build کردن را از لینک زیر دریافت کنید.

**https://pdos.csail.mit.edu/۶.۸۲۸/۲۰۲۲/tools.html**

توصیه ما پیروی از دستورات بالا جهت راه اندازی سیستم عامل می‌باشد. ولی در صورتی که به هر علتی مایل به استفاده از این روش نیستید. می‌توانید این سیستم عامل را در قالب یک docker container اجرا کنید.

## مراحل اجرای سیستم عامل از طریق **Docker**

۱. نصب و اجرای Docker

۱،۱. با مراجعه به این آدرس برنامه Docker Desktop را برای سیستم عامل خود نصب کنید.

۱،۱،۱. در صورتی که هنگام مراجعه به وبسایت و یا مراحل دانلود با خطاهای  مربوط به تحریم برخوردید می‌توانید از شکن برای گذر از تحریم استفاده کنید. در صورتی که همچنان در این مراحل مشکلی داشتید با تیم تدریسیاری در ارتباط باشید.

۱،۲. برنامه Docker Desktop را اجرا و از فعال شدن Docker Engine داخل آن اطمینان حاصل فرمایید.

## ۲. دریافت Docker Image

۲،۱. با استفاده از دستور `docker pull wtakuo/xv6-env` Image مربوطه را دانلود کنید.

۲،۱،۱. در صورتی که در اجرای دستور بالا با خطاهای مربوط به تحریم مواجه شدید در تمامی مراحل بعدی بجای عبارت wtakuo/xv6-env از عبارت زیر استفاده کنید.

m.docker-registry.ir/wtakuo/xv6-env

## ۳. اجرای محیط Container جهت کار با سیستم عامل

۳،۱. کد های مربوط به سیستم را در یک پوشه (فرضا xv6-riscv) ذخیره کنید. داخل این پوشه دستور زیر را اجرا کنید.

*docker run -it --rm -v $(pwd):/home/xv6/xv6-riscv wtakuo/xv6-env*

در صورتی که اجرای مراحل فوق همگی موفقیت آمیز باشد، باید با پیغامی مشابه زیر مواجه شوید:

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

xv6@0c765f60374a:~/xv6-riscv$
```

۳،۲. در این محیط می‌توانید دستورات لازم جهت بیلد و اجرای سیستم‌عامل را فراخوانی کنید.

# فاز اول

برای توضیحات تکمیلی درمورد کدها و قسمت های مختلف سیستم عامل می‌توانید از منابع زیر استفاده کنید:

- XV6 Book[1]
- Learn OS with me[2]
- Google

## بررسی فرایند اجرای سیستم عامل

در ابتدا می‌خواهیم نحوه عملکرد و اجرای سیستم عامل و لود شدن اولین پردازه را بررسی کنیم. برای این منظور ابتدا فایل main.c سیستم عامل را بررسی کنید. سپس مراحلی که برای ساخته شدن (نه اجرا) اولین پردازه سیستم عامل طی می‌شود را بررسی کنید. برای راحتی و درک بهتر. می‌توانید از ابزار GDB و قرار دادن Breakpoint برای تابع userinit استفاده کنید. (می‌توانید به ویدیوهای آموزشی قرار داده شده مراجعه کنید.)

در گزارش نهایی خود به سوالات زیر پاسخ دهید:

- مراحل بوت شدن سیستم عامل از تابع main تا اجرای اولین پردازه را به طور کلی شرح دهید
- مراحل فراخوانی یک سیستم کال، از برنامه‌ی یوزر تا کد کرنلی که نوشتید را شرح دهید. توضیح دهید به ترتیب کدام توابع و کدام قسمت از کد اجرا می‌شود.

سعی داشته باشید برای پاسخ به این سوالات را با ارجاع دادن به کدها و اسکرین شات کامل کنید.

---

[1] https://pdos.csail.mit.edu/6.828/2022/xv6/book-riscv-rev3.pdf

[2] https://xiayingp.gitbook.io/build_a_os/

# پیاده سازی دو سیستم کال

در این قسمت می خواهیم دو سیستم کال به سیستم عامل xv6 اضافه کنیم.

## int getProcTick(int pid)

در این فراخوانی سیستمی از شما می‌خواهیم که تعداد تیک هایی (Ticks) که در سیستم عامل xv6 از لحظه ایجاد یک پردازه با آیدی pid تا لحظه فراخوانی این سیستم کال، گذشته است را برگردانید. برای این فراخوانی سیستمی لازم است که با متغیر ticks در xv۶ آشنا شوید. همانطور که میدانید هر سیستم عاملی برای جلو رفتن کارش در قسمت زمان بندی و ... به یک کلاک 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 برنامه‌ی دیگر اجرا می‌شود. با کاربرد های بیشتر این متغیر در فاز سوم پروژه آشنا خواهیم شد. این متغیر در فایل trap.c با آمدن وقفه تایمر آپدیت میشود که در ویدیویی که برای شما خواهیم گذاشت درباره این فایل توضیح خواهیم داد. برای پیاده سازی این فراخوانی سیستمی نیازی به دانستن جزئیات پیاده سازی متغیر ticks نیست و صرفا پیاده سازی خود فراخوانی سیستمی مد نظر است. برای پیاده سازی این بخش لازم است در ساختار struct proc یک فیلد جدید اضافه کرده و برای هر پردازه زمان ایجاد شدن آن را بر اساس ticks ذخیره کنید و با صدا شدن این سیستم کال، آن پردازه را پیدا کرده و اختلاف مقدار زمان ذخیره شده برای آن پردازه با مقدار متغیر ticks را برگردانید. 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.

نتیجتا شما باید یک فراخوانی سیستمی پیاده سازی کنید که مقدار تیک های گذشته برای پردازه با آیدی pid از زمان ایجاد شدن آن تا آن لحظه را برگرداند. همچنین پس از ساخت این فراخوانی سیستمی از شما میخواهیم که یک برنامه تست به نام getProcTicksTest.c ایجاد کنید که با آن صحت کارکرد این فراخوانی سیستمی را چک کنید.

## int sysinfo(struct sysinfo *info)

سیستم کال sysinfo در سیستم عامل لینوکس برخی اطلاعات کلی در مورد سیستم را به ما می‌دهد. برای اطلاعات بیشتر در مورد آن می‌توانید به این لینک مراجعه کنید. در این قسمت می‌خواهیم یک نسخه‌ی ساده شده از این سیستم کال را به سیستم عامل xv6 اضافه کنیم. سیستم کال ما باید یک پوینتر به یک struct sysinfo بگیرد، اطلاعات درون این ساختار را کامل کند (مشخصا این قسمت به صورت call by reference است) و در صورت عدم خطا مقدار صفر و در صورت وجود خطا مقدار منفی یک را برگرداند. ساختار struct sysinfo به صورت زیر است و شما باید آن را در فایل مناسب تعریف کنید.

```
struct sysinfo {
              long uptime;             // Seconds since boot
              unsigned long totalram;  // Total usable main memory size
              unsigned long freeram;   // Available memory size
              unsigned short procs;    // Number of current processes
          };
```

برای پاس دادن یک پوینتر به یک سیستم کال به تابع argaddr در فایل syscall.c مراجعه کنید.

- Uptime

باید ثانیه هایی که از بوت سیستم گذشته را پیدا کنید. برای این کار از همان متغیر ticks که در سیستم کال قبلی استفاده کردید استفاده کنید، اما باید تحقیق کنید که در سیستم عامل xv6 هر تیک معادل چند ثانیه است و مقدار ticks را به ثانیه تبدیل کنید.

- Memory statistics

باید مقدار کل حافظه و مقدار حافظه‌ی استفاده نشده را پیدا کنید. در سیستم عامل xv6، عملیات های مربوط به حافظه همگی با واحد Memory Page انجام می‌شوند. این Memory Page ها در قالب یک لیست پیوندی در سیستم عامل نگهداری می‌شوند. برای دسترسی به این لینکد لیست بایستی به ساختار kmem مراجعه کنید. واحد خروجی شما باید به بایت باشد، پس نیاز است تحقیق کنید که در سیستم عامل xv6 هم Memory Page چند بایت است و آن را تبدیل کنید.

دقت: از آنجایی که این ساختار می‌تواند همزمان توسط چندین پردازه دچار دخل و تصرف شود، توصیه اکید می‌شود که عملیات های مربوط به سیستم کال خود را به صورت اتمیک انجام دهید.

- Processes

باید تعداد پردازه های فعال در سیستم را پیدا کنید. برای این کار به ساختار proc مراجعه کنید. برای استفاده از این ساختار نیز عملیات های مربوط به سیستم کال خود را به صورت اتمیک انجام دهید.

برای بررسی صحت پیاده سازی خود یک برنامه‌ی کاربر ایجاد کنید که از این سیستم کال استفاده می‌کند و نتایج را چاپ می‌کند.

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 برنامه‌های خود را با دیگر دانشجویان به اشتراک بگذارید.

## توضیحات

- این فاز پیش‌نیاز قطعی فازهای بعدی است و انجام ندادن آن باعث می‌شود که نتوانید فاز دوم را شروع کنید و همچنین نمی‌تواند برای انجام پروژه گروهی را تشکیل دهید.
- پروژه شما تحویل اسکایپی 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 بات سوالات 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 Sid_hw1.zip ( که Sid را با شماره دانشجویی خود جایگزین کنید) که شامل دو مورد زیر است:

- گزارش خیلی مختصر از آنچه که انجام داده‌اید  تا دو فراخوانی سیستمی خواسته شده را به ۶XV اضافه کنید.
- پوشه‌ای که در آن کدهای شما وجود دارد. دقت کنید که تنها و تنها فایل‌هایی را که تغییر داده‌اید یا اضافه کرده‌اید را برای ما بفرستید.

موفق باشید

تیم تدریسیاری درس سیستم‌های عامل