دانشکده مهندسی
کامپیوتر و فناوری اطلاعات

پروژه - فاز اول

دکتر جوادی

سیستم‌های عامل

پروژه - فاز اول

مقدمه:

همانطور که در کلاس درس بیان شد، پروژه درس سیستم های عامل مربوط به شناخت کامل سیستم عامل آموزشی xv6 و افزودن قابلیت های جدید به آن است. با انجام دقیق این پروژه و تطبیق مفاهیمی که در کلاس درس معرفی شده است به یادگیری عمیق و دائمی شما کمک خواهد کرد. برنامه نویسی در سطح سیستم عامل به شما کمک میکند تا تجربه ای منحصر به فرد داشته باشید و دید بهتری نسبت به آنچه که در زبان های سطح بالاتر می نویسید بدست آورید. هدف از دو فاز ابتدایی این پروژه آشنایی شما دانشجویان با این سیستم عامل آزمایشی است و روند کار به این صورت است که در فاز اول به صورت فردی و در فاز دوم و سوم به صورت گروهی باید تغییراتی در این سیستم عامل ایجاد کنید تا بتوانید درک خوبی از این سیستم عامل را بدست آورید. در فاز اول پروژه، بایستی درک مناسبی از برخی مفاهیم پایه ای از جمله پردازه ها، سیستم کال ها و عملکرد کلی سیستم عامل را پیدا کنید و در نهایت دو سیستم کال ساده را به سیستم عامل خود اضافه کنید.

در این ترم، ما با آخرین نسخه از سیستم عامل xv6 که بر اساس معماری پردازنده RISC-V توسعه یافته است کار خواهیم کرد. برای آشنایی با این سیستم عامل، میتوانید از لینک زیر استفاده کنید:
https://pdos.csail.mit.edu/6.828/2022/xv6.html
برای اجرای این سیستم عامل می توانید مراحل و پیش نیازهای build کردن آن را از لینک زیر دریافت کنید.
https://pdos.csail.mit.edu/6.828/2022/tools.html

توصیه ما استفاده از Docker Container میباشد به این صورت که image محیط اجرایی پردازنده xv6 را در قالب یک container اجرا میکنیم. در ادامه به نحوه اجرای آن می پردازیم.

مراحل اجرای xv6 به کمک Docker:

1. نصب و اجرای
   1.1. با مراجعه به این آدرس برنامه Desktop Docker را برای سیستم عامل خود نصب کنید.
      1.1.1. در صورتی که هنگام مراجعه به وب سایت و یا مراحل دانلود با خطاهای مرتبط با تحریم برخورد کردید از شکن برای عبور از تحریم استفاده کنید. در صورتی که همچنان در این مراحل مشکل داشتید با تیم تدریس یاری در میان بگذارید.
   1.2. برنامه Docker Desktop را اجرا و از فعال شدن Docker Engine داخل آن اطمینان حاصل کنید.
2. دریافت Docker Image
   2.1. با استفاده از دستور زیر کانتینر wtakuo/xv6-env را دریافت کنید:
   ```
   docker pull wtakuo/xv6-env
   ```
3. اجرای محیط Container جهت اجرای سیستم عامل
   3.1. 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
   ```
   3.2. چنانچه بر روی ویندوز کار میکنید دستور زیر را اجرا کنید:
   ```
   docker run -it --rm -v "%cd%:/home/xv6/xv6-riscv" wtakuo/xv6-env
   ```
   3.3. با استفاده از دستورات زیر میتوانید کار خود را با سیستم عامل شروع کنید:
      3.3.1. با استفاده از دستور `make clean` فایل های کامپایل شده را پاک کنید.
      3.3.2. با استفاده از دستور `make qemu` پروژه را کامپایل و اجرا کنید تا سیستم عامل مربوط به کدی که نوشته اید اجرا شود.

پیش از شروع:

پیش از شروع خوب است که اطلاعات کافی راجب نحوه اجرا شدن سیستم عامل آزمایشیمون و نحوه اجرا شدن اولین پردازه اطلاعات کافی بدست آوریم. همچنین میتوانید از کتاب xv6: a simple, Unix-like teaching operating system به عنوان منبع استفاده کنید.

مراحل اضافه کردن سیستم کال به سیستم عامل XV6:

در فاز اول پروژه می خواهیم که دو سیستم کال جدید به این سیستم عامل آموزشی اضافه کنیم. خوب است که قبل از هرچیز مروری بر مراحل اضافه کردن سیستم کال داشته باشیم:

1. در صورت نیاز اضافه کردن struct جدید در هر فایل مربوطه
2. در صورت تعریف یک تایپ جدید یا یک تابع جدید که قرار است در فایل های دیگر مورد استفاده قرار گیرد، تعریف آن را در فایل defs.h مانند نمونه های قبلی قرار دهید. در کل هر تایپ یا تابعی که در فایل دیگری تعریف شده و لازم است در فایل های دیگر مورد استفاده قرار گیرد باید پروتوتایپ آن در این فایل قرار بگیرد.
3. در فایل syscall.h سیستم کال جدید را با یک شماره ی جدید اضافه کنید.
4. در فایل syscall.c نام تابع سیستم کال مورد نظر را به نامی که در فایل syscall.h قرار دادید مپ کنید.
5. پروتوتایپ تابع مورد نظر را نیز در ابتدای فایل syscall.c اضافه کنید. (extern "function" definition)
6. پیاده سازی تابع سیستم کال مورد نظر را با نامی که در sysproc.c قرار دهید. در این بخش اصولاً بخش انتقال و تبدیل پارامتر ها انجام می شود و در فایل proc.c پیاده سازی اصلی سیستم کال انجام می شود.
7. برای اینکه برنامه های کاربر بتوانند از این سیستم کال ها استفاده کنند لازم است در فایل usys.pl سیستم کال مربوطه را اضافه کنید.
8. پروتوتایپ تابع مورد نظر را در فایل user.h اضافه کنید.
9. در نهایت Makefile را تغییر دهید و اسم فایل برنامه ی کاربری که نوشتید را در بخش UPROGS اضافه کنید.

سیستم کال Child Processes:

در بخش اول قرار است که یک سیستم کال جدید به اسم child_processes به سیستم عامل آموزشی XV6 اضافه کنیم. این سیستم کال لیستی از اطلاعات پردازه های فرزند (پردازه هایی که با fork به وجود آمدند) را بر میگرداند. توجه شود که تمام فرزندان پردازه های فرزند نیز شامل خروجی می شوند.
سیستم کال Child Processes به یک پردازه اجازه می‌دهد تا لیستی از پردازه‌های فرزند و نوادگان خود را مشاهده کند. این ویژگی برای مدیریت، نظارت و دیباگ سلسله‌مراتب پردازه‌ها بسیار کاربردی است و به شناسایی و تحلیل رفتار پردازه‌ها در سیستم کمک می‌کند.

```
int child_processes(struct child_processes*);
```

به طور کلی یک همچین سیستم کالی خواهیم داشت که ورودی پوینتر به یک struct میگیرد تا آن را پر کند. اگر با موفقیت انجام شد خروجی ۰ میدهد.
برای struct child_processes همچین چیزی پیشنهاد می شود: (اسم و اطلاعات آن را میتوانید با توجه به نیاز های خود تغییر دهید.)

```
struct proc_info {
   char name[16];
   int pid;
   int ppid;
   enum procstate state;
};
```

```
struct child_processes {
   int count;
   struct proc_info processes[NPROC];
};
```

خروجی مورد انتظار چیزی مشابه زیر خواهد بود:

```
$ child_processes
number of child: 16
PID     PPID    STATE   NAME
6       5       sleep   child_processes
7       6       sleep   child_processes
8       6       sleep   child_processes
9       6       sleep   child_processes
10      6       sleep   child_processes
11      7       sleep   child_processes
12      7       sleep   child_processes
13      7       sleep   child_processes
14      8       sleep   child_processes
15      8       sleep   child_processes
16      9       sleep   child_processes
17      11      sleep   child_processes
18      11      sleep   child_processes
19      12      sleep   child_processes
20      14      sleep   child_processes
21      17      sleep   child_processes
```

در پردازه های تستی فرزند می توانید از سیستم کال sleep استفاده کنید تا فورا به اتمام نرسند و بتوانید در نهایت در خروجی خود آنها را مشاهده کنید.

سیستم کال Report Traps:

در بخش دوم قرار است که یک سیستم کال جدید دیگر به اسم trap_report به سیستم عامل آموزشی XV6 اضافه کنیم. این سیستم کال یک ریپورتی از کرش هایی که برنامه ما کرده است می دهد. (شامل خود پردازه - فرزندان و نوادگان پردازه ای که سیستم کال را فرا می خواند)
سیستم کال Report Traps برای جمع‌آوری و نمایش اطلاعات مربوط به کرش پردازه‌ها در xv6 کاربردی است. این سیستم کال اطلاعاتی مانند PID، نام پردازه، sepc، scause و stval را که هنگام رخ دادن یک trap ثبت می‌شوند، نمایش می‌دهد. این ویژگی به توسعه‌دهندگان کمک می‌کند تا مشکلات نرم‌افزاری را به‌راحتی شناسایی و دیباگ کرده و به بهبود پایداری و امنیت سیستم کمک کنند.
زمانی که یک پردازه کرش میکند (مثلا به فضای نامعتبری میخواهد دسترسی داشته باشد) در سیستم عامل یک trap ایجاد می شود. (کد آن را در فایل trap.c موجود در کرنل میتوانید مشاهده کنید.)
به صورت پیشفرض وقتی کرشی رخ دهد در کرنل همچین پیغامی پرینت می شود:

```
usertrap(): unexpected scause 0x000000000000000c pid=3
            sepc=0x0000000012345678 stval=0x0000000012345678
```

در مرحله اول باید اطلاعات این پیغام که شامل pid, pname, scause, sepc و stval می شود را ذخیره کنیم تا بتوانیم بعدا در سیستم کال جدیدمان این اطلاعات را نمایش دهیم.

در اینجا scause علت خطا را نشان میدهد. برای مثال در تصویر بالا عدد ۱۲ را نشان میدهد که به معنی Instruction page fault می باشد.
sepc نیز آدرس خط برنامه ای که کرش کرده است را نشان میدهد.

در نهایت سیستم کال زیر اضافه خواهد شد:

```
int report_traps(struct report_traps*);
```

در این سیستم کال ورودی یک استراکت از تایپ report_traps میگیرد تا کرش های مربوط به پردازه خود را در آن پر کند. **دقت کنید که تنها کرش های مربوط به خود پردازه و فرزندان و نوادگان پردازه ای که این سیستم کال را اجرا میکند باید در خروجی قرار بگیرد!**

```
#define MAX_REPORT_BUFFER_SIZE 10

struct report {
   char pname[16]; // Process Name
   int pid;        // Process ID
   uint64 scause;  // Supervisor Trap Cause
   uint64 sepc;    // Supervisor Exception Program Counter
   uint64 stval;   // Supervisor Trap Value
};

struct {
   struct report reports[MAX_REPORT_BUFFER_SIZE];
   int numberOfReports;
   int writeIndex;       // Circle loop
} _internal_report_list;

struct report_traps {
   struct report reports[MAX_REPORT_BUFFER_SIZE];
   int count;
}
```

خروجی مورد انتظار چیزی مشابه زیر خواهد بود:

```
$ test
number of exceptions: 2
PID     PNAME   scause                  sepc                    stval
3       test    0x000000000000000c      0x0000000012345678      0x0000000012345678
4       test    0x000000000000000c      0x0000000023457312      0x0000000023457312
```

بخش امتیازی:

تمامی این ریپورت ها را در یک فایل نیز ذخیره کنید و در هنگام لود شدن سیستم عامل آنها را دوباره بخوانید تا با ری استارت کردن سیستم عامل همچنان این سیستم کال بتواند لیست خطا های گذشته را پرینت کند. همچنین در فایل ذخیره شده نباید محدودیتی در سایز بافر وجود داشته باشد اما هنگام خواندن فقط چند مورد آخر را لود کنید.

توضیحات پایانی:

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 می‌توانید هر یک از تدریس یاران را به پروژه 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 آنلاین خواهد داشت بنابراین از استفاده از کدهای دیگران 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_P1_Sid.zip** (که Sid را با شماره دانشجویی خود جایگزین کنید) که شامل دو مورد زیر است:

1. **گزارش خیلی مختصر** از آنچه که انجام داده‌اید تا دو فراخوانی سیستمی خواسته شده را به xv6 اضافه کنید.
2. پوشه ای 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 تدریس یاری درس سیستم های عامل