

آزمایشگاه سیستم‌های عامل

آزمایش شماره‌ی سه: همگام‌سازی

تاریخ تحویل: ۱۷ آذر ۱۳۹۳

## اهداف آزمایش

- آشنایی با همگام‌سازی در لینوکس
- آشنایی با نمونه‌ای از پیاده‌سازی مکانیسم‌های همگام‌سازی
- پیاده‌سازی و اشکال‌زدایی یک مکانیسم جدید در هسته‌ی لینوکس

## چکیده

در این آزمایش ابتدا به بررسی نحوه‌ی پیاده‌سازی یکی از مکانیسم‌های همگام‌سازی در لینوکس خواهیم پرداخت. در ادامه با استفاده از دانش به‌دست‌آمده از بخش قبل، به پیاده‌سازی یک مکانیسم جدید همگام‌سازی خواهیم پرداخت و در انتها با طراحی یک سناریو، درستی عملکرد مکانیسم جدید پیاده‌سازی شده را نشان خواهیم داد.

## شرح آزمایش

این آزمایش شامل دو بخش است. در بخش اول به بررسی کد منبع مکانیسم همگام‌سازی سمافور در هسته‌ی لینوکس خواهید پرداخت. در بخش بعدی یک مکانیسم جدید همگام‌سازی توصیف‌شده که آن را پیاده‌سازی خواهید کرد.

### آشنایی با پیاده‌سازی سمافور در لینوکس

بخش عمده‌ای از تعریف‌ها و پیاده‌سازی‌های مرتبط با سمافور در فایل‌های `semaphore.h`<sup>۱</sup> و `semaphore.c`<sup>۲</sup> آمده است. برای آشنایی بیشتر با مقدمات پیاده‌سازی یک مکانیسم همگام‌سازی، به سؤالاتی از این فایل‌ها خواهیم پرداخت:

۱. ساختار داده‌ی اصلی سمافور با نام `semaphore` در فایل `semaphore.h` تعریف شده است. به‌طور کامل توضیح دهید<sup>۳</sup> که هریک از ۳ عضو این ساختار چه کاربردی دارند؟
۲. توابع و ماکروهایی که برای مقداردهی اولیه به سمافور تعریف‌شده‌اند را نام برده و به‌طور مختصر شرح دهید.
۳. به پیاده‌سازی توابع `up()` و `down()` در فایل `semaphore.c` مراجعه کنید. توضیح دهید که برای حفاظت از ناحیه‌ی بحرانی در این دو تابع چگونه عمل شده است.
۴. به‌طور کامل توضیح دهید که تابع `up()` در ناحیه‌ی بحرانی خود چه می‌کند؟
۵. به‌طور کامل توضیح دهید که تابع `down()` در ناحیه‌ی بحرانی خود چه می‌کند؟

پس از پاسخ به سؤالات بالا باید توان توضیح نقش اساسی مفاهیم زیر در پیاده‌سازی سمافور را داشته باشید.

- نحوه‌ی حفاظت از ناحیه‌ی بحرانی
- صف

نکته: مورد بالا تنها یکی از نمونه‌ها است و شما می‌توانید با مطالعه‌ی نمونه‌های دیگر، روش‌های دیگری برای پیاده‌سازی خود بیابید. در این راستا یک مستند نمونه در سایت درس قرار داده‌شده است.

---

<sup>۱</sup> در پوشه‌ی `include/linux/`

<sup>۲</sup> در پوشه‌ی `kernel/`

<sup>۳</sup> منظور از توضیح کامل، توضیحی است که گویای تمام مفاهیم بکار رفته در پیاده‌سازی باشد. برای مثال ممکن است لازم باشد که یک ساختار، تابع یا عضو پراهمیت، به‌طور جزئی شرح داده‌شود. در غالب این موارد، توضیحی فراتر از کامنت‌های موجود در کد منبع لازم است.

## پیاده‌سازی مکانیسم جدید همگام‌سازی

در این بخش شما یک مکانیسم جدید همگام‌سازی را پیاده‌سازی خواهید کرد که دارای پروتکل ارث‌بری اولویت<sup>۴</sup> است. برای کسب اطلاعات بیشتر در مورد این پروتکل می‌توانید معادل انگلیسی آن را در اینترنت جستجو کنید. پیاده‌سازی شما باید نوع<sup>۵</sup> newsem را به همراه ۳ فراخوان سیستمی زیر را در اختیار کاربر قرار دهد:

- `int newsem_init( newsem* instance)`: اشاره‌گر به یک newsem را دریافت کرده و مقداردهی اولیه‌ی آن را انجام می‌دهد.
- `int newsem_wait( newsem* instance)`: مقدار سمافور را کاهش می‌دهد و اگر مقدار منفی شود وظیفه‌ی درخواست دهنده را بلوکه می‌کند.
- `int newsem_signal( newsem* instance)`: مقدار سمافور را افزایش می‌دهد و در صورت وجود پردازنده‌ی منتظر، پر اولویت‌ترین آن‌ها را از حالت بلوکه خارج می‌کند.

مقدار بازگردانده‌شده توسط توابع بالا باید نشان‌دهنده‌ی موفقیت فراخوانی و یا کد خطایی که رخ داده باشد. تخصیص مقادیر کدها در این پروژه دلخواه است. البته معمولاً این کدها در سیستم‌های عامل مختلف دارای مقادیر استاندارد هستند. برای مثال غالباً مقدار صفر نشان‌دهنده‌ی موفقیت است و مقدار منفی عدم موفقیت را می‌رساند. ممکن است متغیر `errno` نیز در مواردی جزئیات نوع خطا را نشان دهد. در این پروژه از شما انتظار می‌رود که نوع خطای رخ داده را نیز بازگردانید.

تعریف newsem و فراخوانی توابع باید به‌صورت زیر قابل‌استفاده باشد:

```
newsem * ns1 = new newsem;
int ret_val;
ret_val = newsem_init( ns1);
ret_val = newsem_wait( ns1);
ret_val = newsem_signal( ns1);
```

---

<sup>۴</sup> Priority Inheritance

<sup>۵</sup> type

## طراحی سناریوی تست

برای اطمینان از عملکرد درست پیاده‌سازی شما، نیاز به یک برنامه‌ی C است که اجرای آن نشان‌دهنده‌ی این مطلب باشد. این برنامه باید درستی عملکرد به‌عنوان یک مکانیسم همگام‌سازی و همین‌طور درستی عملکرد پروتکل ارث‌بری اولویت را نیز نشان دهد.

## سؤال پایانی

توضیح دهید که کدام بخش‌های یک مکانیسم همگام‌سازی در لینوکس (ازجمله مکانیسم پیاده‌سازی شده در این پروژه) باید حتماً در سطح هسته اجرا شوند؟ چرا؟ (به عبارتی، اگر قرار بود newsem را به‌صورت یک سرویس سطح کاربر پیاده‌سازی کنید، مهم‌ترین فراخوانی‌های سیستمی که ناچار به استفاده از آن‌ها بودید چه بودند؟)

## سایر نکات

- پاسخ به سؤالات را در گزارش خود بیاورید. گزارش باید به‌صورت تایپ‌شده یا تصویر خوانا آپلود شود.
- تنها بخش‌های تغییریافته‌ی هسته و کدهای مربوط به سناریوی تست خود را به همراه گزارش آپلود کنید (کل هسته را آپلود نکنید).
- طبیعت پروژه‌های آزمایشگاه به‌گونه‌ای است که ممکن است با مشکلات پیش‌بینی‌نشده مواجه شوید. دستیاران آموزشی در رفع این مشکلات به شما کمک خواهند کرد، ولی مسئولیت انجام درست پروژه به عهده‌ی خود شما است. بنابراین توصیه می‌شود که پروژه را زود شروع کنید.
- پروژه‌های آزمایشگاه باید در گروه‌های سه‌نفره انجام شوند. همه‌ی اعضای گروه باید روی همه‌ی قسمت‌های پروژه تسلط داشته باشند و هریک از اعضا متناسب با میزان تسلط نمره دهی خواهد شد.
- سؤالات خود را در فروم مخصوص پروژه در CECM مطرح کنید و در صورت امکان به سؤالات دیگران پاسخ دهید. دستیاران آموزشی درس نیز به پرسش‌های مطرح‌شده در سایت پاسخ خواهند داد.

• **این پروژه به‌هیچ‌وجه تمدید نخواهد شد.**