به نام خدا

سبحان رنجبر 401521264

سامان صیادفعال 401521408

در این سوال با سیستم عامل XV6 قرار است بتوانیم اگر پروسه CPU یا Memory بیشتر از حد نیاز خواست ان را کیل کنیم یا نتواند بگیرد که ما در ادامه از واژه های limit CPU و limit Memory استفاده می کنیم.

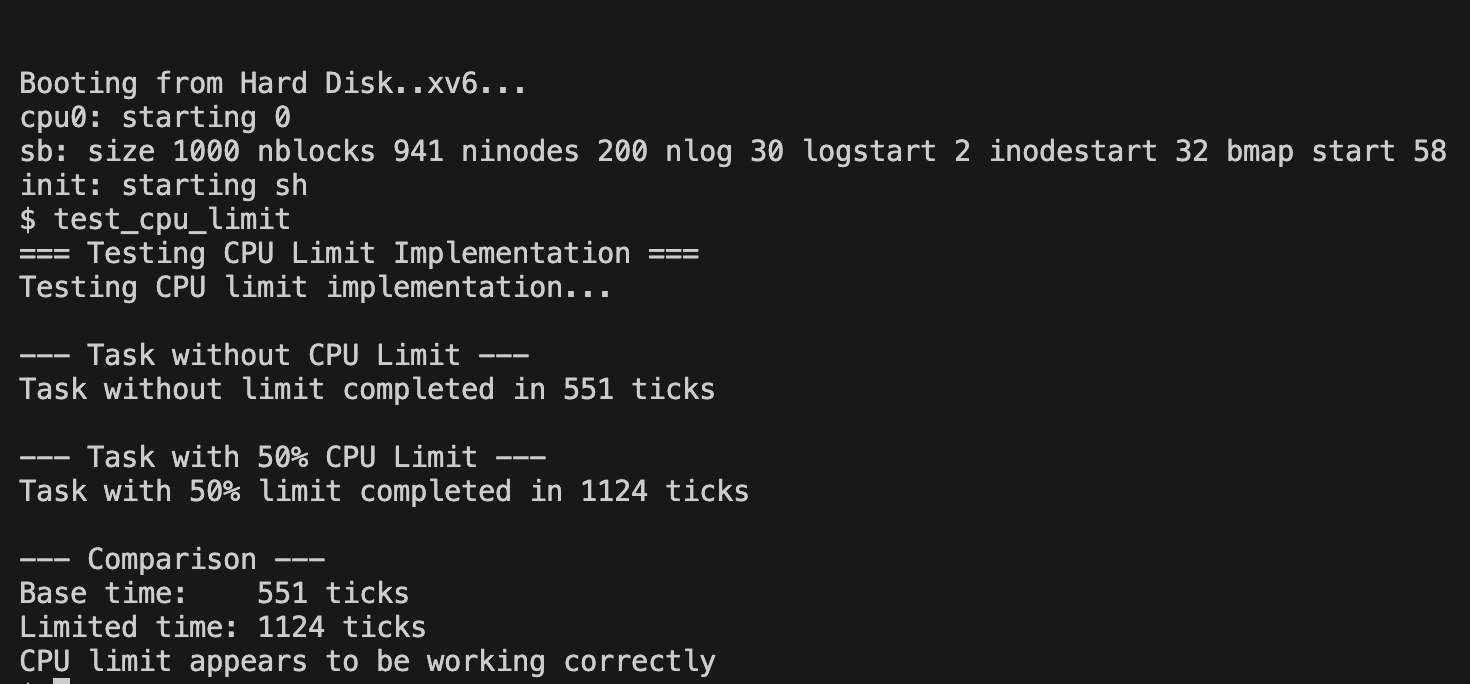
برای راحت تر خوندن کد های که نوشتیم جاهایی که تغییر دادیم بالاش از کامنت

//this is change

استفاده کردیم و در هر خط کامنت هم گذاشتیم و یه دید کلی هم از پروژه در این گزارش کار نوشتیم امیدوارم خوشتون بیاد و قسمت هایی که توضیحات مشترک بود در قسمت پ توضیح داده ایم.

الف) محدود کردن CPU

تستی که برای این بخش دادیم



ب) محدود کردن Memory

به خدا برای همه کد هایی که نوشتیم کامنت گذاشتم الان proc.h چیشو بگم دو متعیر از نوع int تعریف کردیم که مقدار مصرف شده و اجازه دارد مصرف کند گفته شده است.

در قسمت sysproc.c تابعی به نام sys\_change\_memory\_usage تعریف کردیم که برای ما محدودیت هایی که در زمینه مموری نیاز داریم را بودجود بیاورد و همچنین سعی کردیم چاپ کنیم که برای تست هایی که دادیم و در این فایل دو مورد قرار دادیم چگونه محدودیت مموری و حافظه را رفع می کنیم و همان طور که در کد قرار گرفته با شرط و پوینتر هر دفعه چک می کنیم که سایز مورد نظر بالاتر یا پایین تر از محدود گفته شده نرود.

در فایل umalloc.c از توابع free و infree و malloc و imalloc استفاده کردیم

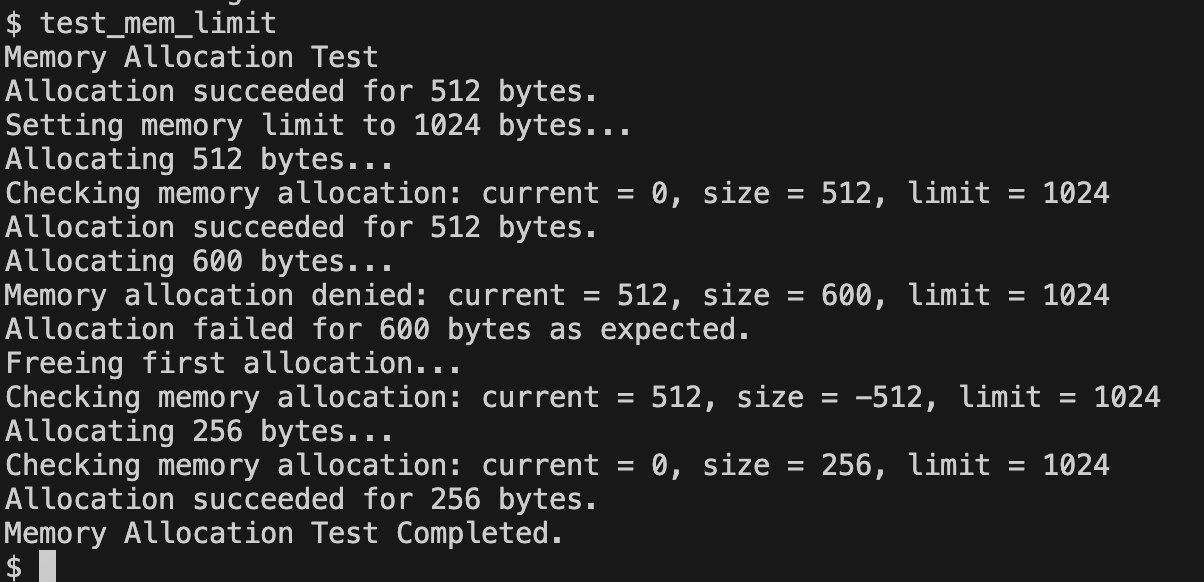
free => بروز نگه داری حافظه و محاسبه اندازه حافظه

infree => برای ازاد سازی حافظه و بلوک های کنار هم که ازاد هستند را ادغام می کند

malloc => محدودیت هایی که در استفاده از حافظه داریم را بررسی می کند

imalloc => اختصاص دادن حافظه و درخواست حافظه از morecore()

تستی که برای این بخش دادیم



پ) قسمت مشترک :

اضافه کردن Test\_mem\_limit و Test\_cpu\_limit به قسمت UPROGS در Makefile .

اضافه کردن دستورات limit کردن برای هم مموری و هم سی پی یو در قسمت defs.h این قسمت برای تعریف کردن توابعی است که زدیم برای او اس. (define کردن منظورمه فارسیش نمی دونم چی می شه 🫣)

سیستم کال ها نیز در در فایل syscall.c تعریف شده است.

شماره سیستم کال هم در فایل syscall.h گفته شده است.

در قسمت sysproc.c ما چندتا شرط گذاشتیم اگر برقرار نبود با -1 از توابع هایی که در مراحل قبل ساختیم خارج شود. برای مثال cpu کمتر از 0 یا بیشتر از 100 بود و در این فایل محدودیت هایی در خود xv6 وجود دارد برای مموری و سی پی یو ولی ما نیاز به متغییر ها و توابع خودمان در این قسمت داریم پس آن ها را حذف کرده و از توابع و متغییر هایی که اضافه کردیم ، استفاده کردیم سعی کردیم توی این فایل توضیح های مفیدی بدیم و از عکس و کد استفاده نکنیم که بسیار زیاد بشود و تی ای موقع خوندن حوصلش سر بره و در کد هم سعی کردیم کامنت بذاریم در جاهای مختلف بازهم اگر جایی کمی یا کاستی بود بگید خوشحال می شیم کامل ترش کنیم.

توضیحات کلی :

Makefile:

همان طور که می دونیم این فایل برای کامپایل کردن و ساخت برنامه است که این جا یک سیستم عامل است.دستورات اجرایی و کامپایل کرنل و برنامه های کاربردی در این جا قرار می گیرد.

main.c:

کرنل از این جا شروع می شود و مقداردهی اولیه (initialize)برای این قسمت می باشد.

syscall.c:

پیاده سازی سیستم کال ها در این قسمت می باشد و مدیریت فراخوانی برنامه های کاربردی توسط این فایل می باشد.

proc.c:

مدیریت پروسس ها / ایجاد و تعویض و برنامه ریزی پروسس و ... => در این قسمت قرار دارند.

موارد زیر کاربردی نداشتن زیاد در پروژه که زدیم فقط برای اشنایی با پروژه و همچنین چیزهایی که یادگرفتم از هرکدوم در کمتر از یه خط توضیح ریزی گفتم بدم.

bootsam.S:

مسئول راه اندازی BIOS و بوت لودر(راه انداری رایانه).

bootmain.c:

مسئول بارگذاری کردن کرنل از دیسک به حافظه

kalloc.c:

مدیریت حافظه فیزیکی

vm.c:

ویرچوال ماشین مدیریت حافظه مجازی

trap.c:

مدیریت وقفه ها

file.c:

هر کاری بافایل ها داشته باشیم در این جا قرار دارد ( مثل بازکردن یا خواندن و نوشتن فایل ها)

bio.c:

تنظیمات بافر I/o و کش کردن بلوک های دیسک در این جا قرار دارد.

ide.c:

مدیریت خواندن و نوشتن در بلوک های دیسک

console.c:

پردازش ورودی و خروجی صفحه کلید و نمایش ان در خروجی

به دلیل حجم بالا، فایل کامل پروژه در گیت هاب قرار دارد

لینک پروژه در گیت هاب

[Github.com/samansayad93/osproject](http://github.com/samansayad93/osproject)