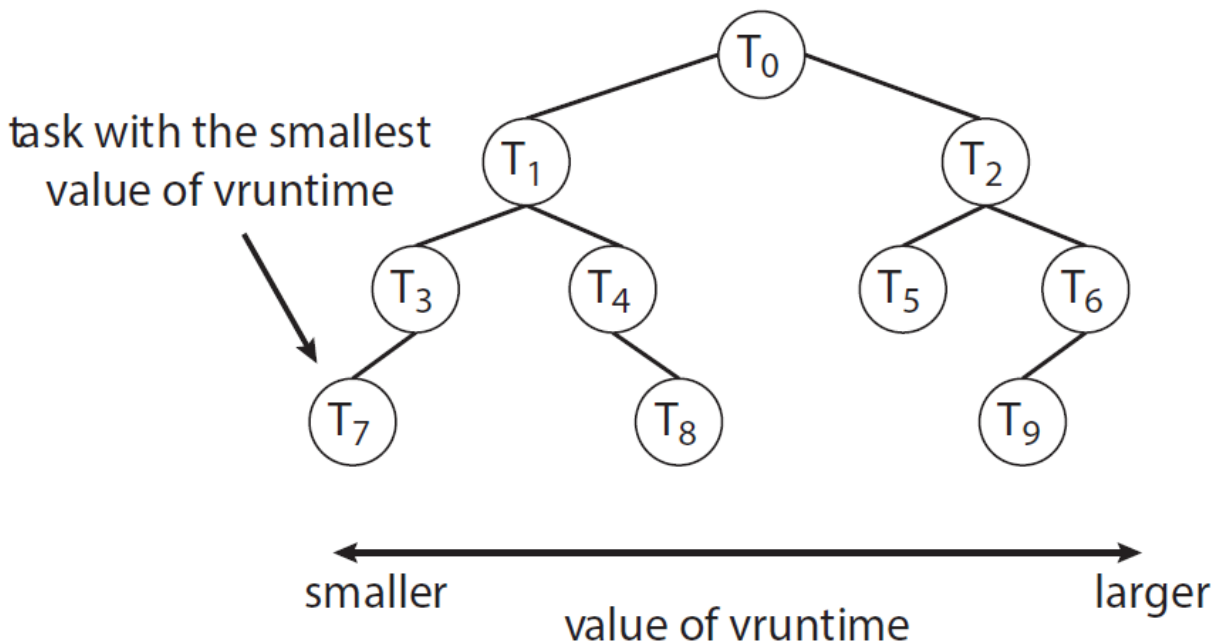


پروژه سوم آزمایشگاه سیستم عامل

مرتضی نوری (810198481)
شایان شاه محمدی (810198531)
سید محمد امین اطیابی (810198559)

پاسخ سوالات

- 1 - با توجه به بدنه تاب sched می بینیم که یک عمل context switch رخ داده است . وقتی تابع sched صدا زده می شود که یک پردازش در حالت runnable داشته باشیم . در این زمان scheduler جایگزین پردازش فعلی می شود تا پردازش را انتخاب کرده و به حالت running تبدیل کند .
- 2 - مطابق توضیحات فصل 5 کتاب در مورد زمانبندی کاملاً منصفانه لینوکس ، ما یک درخت red-black برای زمانبندی داریم .



در این درخت هرچه به سمت راست برویم زمان اجرا پردازش ها بیشتر می شوند . در نتیجه سمت چپ ترین گره دارای کمترین مقدار vruntime و بیشترین اولویت است ، پس پردازش بعدی که باید اجرا شود سمت چپ ترین گره است .

- 3 - با توجه به فایل proc.c ، تنها یک صف مشترک برای پردازش ها داریم .

```
2409 struct {  
2410     struct spinlock lock;  
2411     struct proc proc[NPROC];  
2412 } ptable;
```

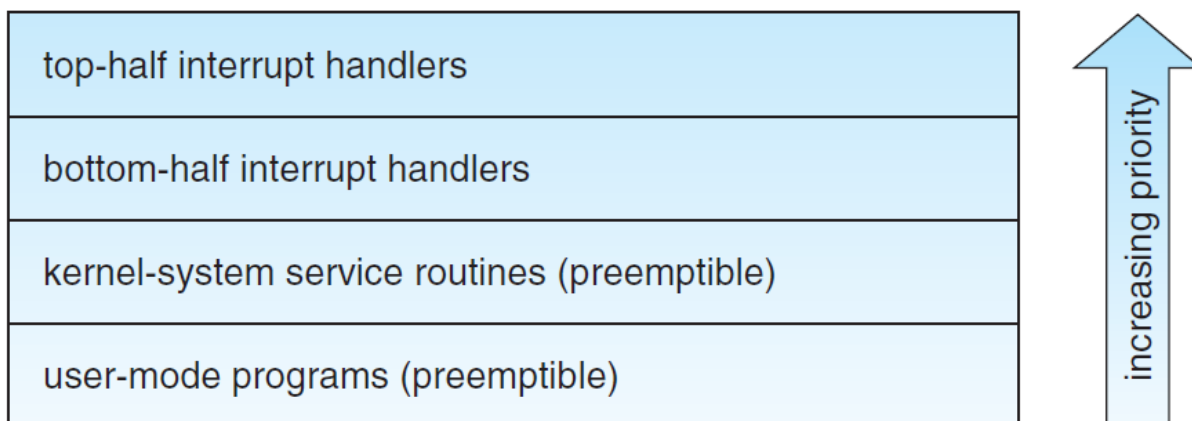
در این حالت وقتی پردازش ای در حال استفاده از پردازنده است باید ptable.lock را گفته باشد .

اما در لینوکس به ازای هر پردازنده یک صف وجود دارد که این صورت نیاز به مکانیزمی جهت load balancing بین پردازنده ها داریم . در مقابل وقتی همه پردازنده ها در صفی مشترک هستند ، نیاز به مدیریت کردن دسترسی های همزمان است که به کمک lock این بخش مدیریت شده است .

4 - این عمل به خصوص در زمانی که پردازنده در حالت idle بسیار مهم است (هیچ پردازنده runnable پیدا نمی شود) . اگر این یک زمانبند دائما با پردازنده هایی که lock شده اند رو به رو شود هیچ پردازنده ای که در حال اجرا پردازنده ای بوده است نمی تواند عمل context switch انجام دهد و هیچ پردازنده ای را به حالت runnable تغییر وضعیت دهد . دلیل اینکه این وقفه به صورت دوره ای فعال می شود این است که برای مثال پردازنده ای منتظر عمل I/O است و اگر این وقفه نباشد این عمل هیچگاه انجام نمی شود . در سیستم های تک پردازنده ای نیز مثال گفته شده کاملا ممکن است ، پس نیاز به وقفه دوره ای ست.

5 - در سیستم عامل لینوکس وقفه ها به دو دسته top-half و bottom-half تقسیم بندی می شوند .

- top-half : در این دسته وقفه غیر فعال می شوند و حداقل های تنها برقرار هستند . مانند ارتباط با سخت افزار و تغییر flag ها در هسته .
- bottom-half : در این دسته بندی وقفه های دیگر فعال هستند و فعالیت های مورد نیاز دیگر وقفه در این دسته اتفاق می افتد .



مطابق دیاگرام بالا ، وقفه های top-half اولویت بیشتری نسبت به bottom-half دارند . همچنین وقفه های bottom-half اولویت بیشتری نسبت به پردازنده های دیگر دارند .

زمانبندی بازخوردی چند سطحی

ابتدا قبل از اجرا برنامه تست پرازده ها را مشاهده می کنیم .

```
> ~/Desktop/os_lab_3/Source make qemu
qemu-system-i386 -serial mon:stdio -drive file=fs.img,index=1,media=disk,format=raw -drive file=xv6.img,index=0,media=disk,format=raw -smp 2 -m 512

(qemu-system-i386:20126): Glib-GObject-CRITICAL **: 18:59:53.708: g_value_set_boxed: assertion 'G_VALUE_HOLDS_BOXED (value)' failed
(qemu-system-i386:20126): dbind-WARNING **: 18:59:53.717: Error retrieving accessibility bus address: org.freedesktop.DBus.Error.ServiceUnknown: The name org.ally.Bus was not provided by any .service files

(qemu-system-i386:20126): Gtk-WARNING **: 18:59:53.756: Theme parsing error: gtk.css:3611:22: 'none' is not a valid color name
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
Morteza Nouri, Shayan Shahmohammadi, Seyed Mohammad Amin Atyabi
$ pproc
name      pid      state    queue_level  cycle      arrival    HRRN      MHRRN
.....
init       1        SLEEPING 1             7           0           0         19
sh         2        SLEEPING 1             3           4           0         46
pproc      3        RUNNING  1             5           284          0         0
$
```

مطابق دیگرام بالا ، وقفه های top-half اولویت بیشتری نسبت به bottom-half دارند .
همچنین وقفه های bottom-half اولویت بیشتری نسبت به پردازنده های دیگر دارند .
نکته دوم سوال رو بخونیم

زمانبندی بازخوردی چند سطحی

بلافاصله بعد از اجرا برنامه تست .

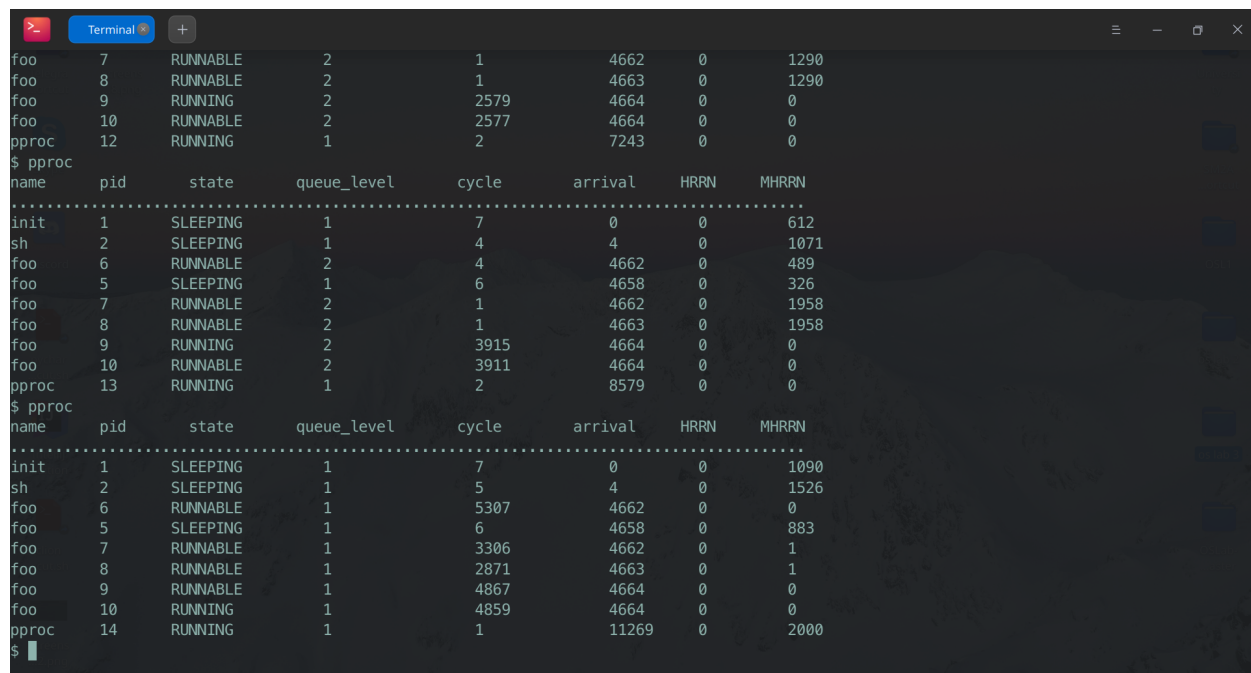
```
nown: The name org.ally.Bus was not provided by any .service files

(qemu-system-i386:20126): Gtk-WARNING **: 18:59:53.756: Theme parsing error: gtk.css:3611:22: 'none' is not a valid color name
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
Morteza Nouri, Shayan Shahmohammadi, Seyed Mohammad Amin Atyabi
$ pproc
name      pid      state    queue_level  cycle      arrival    HRRN      MHRRN
.....
init       1        SLEEPING 1             7           0           0         19
sh         2        SLEEPING 1             3           4           0         46
pproc      3        RUNNING  1             5           284          0         0
$ foo&
$ foo is runnig
pproc
name      pid      state    queue_level  cycle      arrival    HRRN      MHRRN
.....
init       1        SLEEPING 1             7           0           0         341
sh         2        SLEEPING 1             4           4           0         598
foo        6        RUNNABLE 2             4           4662          0         16
foo        5        SLEEPING 2             6           4658          0         10
foo        7        RUNNABLE 1             1           4662          0         65
foo        8        RUNNABLE 2             1           4663          0         65
foo        9        RUNNING  2             130          4664          0         0
foo       10        RUNNABLE 2             129          4664          0         0
pproc     11        RUNNING  1             2           4793          0         0
$
```

مطابق دیگرام بالا ، وقفه های top-half اولویت بیشتری نسبت به bottom-half دارند .
همچنین وقفه های bottom-half اولویت بیشتری نسبت به پردازنده های دیگر دارند .
نکته دوم سوال رو بخونیم

زمانبندی بازخوردی چند سطحی

بعد از مدت زمان زیادی که به بعضی پردازش ها منابع اجرایی اختصاص داده نشده است .



```
$ pproc
foo      7      RUNNABLE  2          1          4662      0          1290
foo      8      RUNNABLE  2          1          4663      0          1290
foo      9      RUNNING  2          2579       4664      0          0
foo     10      RUNNABLE  2          2577       4664      0          0
pproc    12      RUNNING  1          2          7243      0          0
$ pproc
name     pid     state   queue_level  cycle   arrival  HRRN  MHRRN
-----
init      1     SLEEPING      1          7          0        0        612
sh        2     SLEEPING      1          4          4        0       1071
foo        6     RUNNABLE      2          4         4662      0        489
foo        5     SLEEPING      1          6         4658      0        326
foo        7     RUNNABLE      2          1         4662      0       1958
foo        8     RUNNABLE      2          1         4663      0       1958
foo        9     RUNNING      2          3915       4664      0          0
foo       10     RUNNABLE      2          3911       4664      0          0
pproc     13     RUNNING      1          2         8579      0          0
$ pproc
name     pid     state   queue_level  cycle   arrival  HRRN  MHRRN
-----
init      1     SLEEPING      1          7          0        0       1090
sh        2     SLEEPING      1          5          4        0       1526
foo        6     RUNNABLE      1          5307       4662      0          0
foo        5     SLEEPING      1          6         4658      0        883
foo        7     RUNNABLE      1          3306       4662      0          1
foo        8     RUNNABLE      1          2871       4663      0          1
foo        9     RUNNABLE      1          4867       4664      0          0
foo       10     RUNNING      1          4859       4664      0          0
pproc     14     RUNNING      1          1        11269      0       2000
$
```