به نام پروردگار هدایت کننده به راه راست

پروژه اول سیستم عامل – فرآیندها

دکتر رمضانی

پاییز 1401

طراحان پروژه : فاطمه شفیعی، حسام محبی، سیدامین حسینی

نکات :

تحویل پروژه بصورت حضوری و توسط تی‌ای ها انجام خواهد شد.

این پروژه را باید با یکی از زبان‌های اصلی برنامه نویسی یعنی C++, Java, Python پیاده سازی کنید.

پروژه باید به‌صورت گروه های دونفره انجام شود. (امکان پروژه بصورت تکی در شرایط خاص وجود دارد.)

شرح پروژه

هدف از این پروژه آشنایی با مفهوم اولیه فرآیند و متن فرآیند (context) است. برای راحتی کار شما متن فرآیند ساده سازی شده‌است. متن هر فرآیند در این پروژه شامل موارد زیر است :

- **شناسه فرآیند (Process ID)**
- **حالت فرآیند (State)**
- **ثبات ها (Registers)**
- **پروگرم کانتر (PC)**

شناسه فرآیند (Process ID):

شناسه هر فرآیند عددی منحصربفرد است و دستورات ورودی برای هر فرآیند با شناسه آن مشخص می شوند.

حالت فرآیند(State):

در این پروژه هر فرآیند می تواند سه حالت زیر را داشته باشد :

1. **آماده اجرا (ready)**
2. **درحال اجرا (running)**
3. **بلوکه شده (blocked)**

ثبات ها (Registers):

در این پروژه تنها سه رجیستر temp و accumulator ,Instruction register داریم . در هر دستور اگر عددی داده شده باشد، قبل از اجرای دستور عدد در ریجستر temp ذخیره می شود . هر دستور قبل از اجرا در رجیستر IR (Instruction register) ذخیره می شود . هر دستورالعمل در این پروژه بر رجیستر accumulator اعمال می شود . این دستورات در ادامه توضیح داده می شوند .

پروگرم کانتر (PC):

با اجرای هر دستور از برنامه مربوط به یک فرآیند(Process) مقدار این رجیستر یکی اضافه می شود. پس از اجرای آخرین دستوربرنامه فرآیند، PC باید مجدد به اولین خط برنامه اشاره کند .

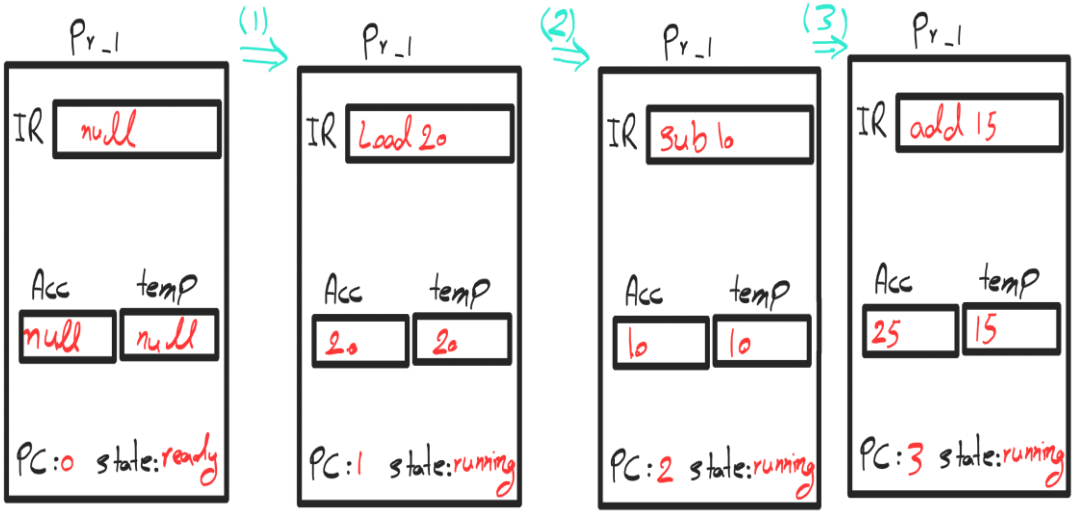
دستورات اجرایی توسط فرآیند

```
1   load 20
2   sub 10
3   add 15
4   mul 3
```

این دستورات در یک فایل و در هنگام ایجاد یک فرآیند به آن داده می شوند. پیش از اجرای هر دستور باید آن دستور به طور کامل در رجیستر IR بارگزاری شود.

این دستورات به شرح زیر می باشند :

1. دستور load که به دنباله آن یک عدد می آید به این صورت عمل می کند که این عدد باید ابتدا در رجیستر temp فرآیند و سپس در رجیستر accumulator بارگزاری شود.
2. دستور sub که به دنباله آن یک عدد می آید به این صورت عمل می کند که این عدد باید ابتدا در رجیستر temp فرآیند و سپس از مقدار رجیستر accumulator به اندازه temp کم می شود .
3. دستور add که به دنباله آن یک عدد می آید به این صورت عمل می کند که این عدد باید ابتدا در رجیستر temp فرآیند و سپس به مقدار رجیستر accumulator به اندازه temp اضافه می شود .
4. دستور mul که به دنباله آن یک عدد می آید به این صورت عمل می کند که این عدد باید ابتدا در رجیستر temp فرآیند و سپس مقدارررجیستر accumulator در temp ضرب می شود .



سیگنال ها

برنامه شما به عنوان ورودی دنباله ای از سیگنال ها را دریافت می کند و نسبت به هر نوع سیگنال باید فعالیت خاصی را روی فرآیند مورد نظر انجام دهد . لیست سیگنال ها به شرح زیر است :

```
1   create_process process_id process_file.txt
2   run_process process_id
3   block_process process_id
4   unblock_process process_id
5   kill_process process_id
6   show_context process_id
```

1. **سیگنال** create_process

در این سیگنال شناسه فرآیند و نام فایلی که دستورات فرآیند در آن وجود دارند، ورودی داده میشوند و باید فرآیند مربوط به آن ایجاد شود. در این حالت، state فرآیند باید به صورت ready مقدار دهی شود.

2. **سیگنال** run_process

شناسه فرآیند را میگیرد و خطی از برنامه که PC آنرا مشخص می کند اجرا می کند و پس از اتمام اجرا state فرآیند باید به صورت ready مقدار دهی شود.

3. **سیگنال** block_process

شناسه فرآیند را میگیرد و state فرآیند باید به صورت block مقدار دهی شود. اگر فرآیند block باشد نباید هیچ سیگنال دیگری را اجرا کند تا زمانی که unblock شود.

4. **سیگنال** unblock_process

شناسه فرآیند را میگیرد و state فرآیند باید به صورت ready مقدار دهی شود.

5. **سیگنال** show_context

در صورت دریافت این دستور باید محتوای فرآیند به شکل استاندارد داده شده چاپ شود.

```
Process ID : pr_1
Instruction Register : sub 10

Accumulator : 10.0              Temp : 10.0
Program Counter : 2             State : ready
```

6. **سیگنال** kill_process

در صورتی که یک فرآیند kill شود از لیست فرآیند ها حذف شده و context آن به طور کامل از بین می رود.

مثال زیر، برای درک بهتر شما داده شده است.

ورودی :

```
create_process pr_1 process1_instruction_file.txt
run_process pr_1
show_context pr_1
create_process pr_2 process2_instruction_file.txt
run_process pr_2
show_context pr_2
block_process pr_1
run_process pr_1
unblock_process pr_1
run_process pr_1
show_context pr_1
kill_process pr_1
show_context pr_1
run_process pr_2
show_context pr_2
kill_process pr_2
show_context pr_2
```

خروجی :

```
loop :  0
create_process pr_1 process1_instruction_file.txt
loop :  1
run_process pr_1
loop :  2
show_context pr_1
Process ID : pr_1
Instruction Register : load 20

Accumulator : 20.0                Temp : 20.0
Program Counter : 1               State : ready
loop :  3
create_process pr_2 process2_instruction_file.txt
loop :  4
run_process pr_2
loop :  5
show_context pr_2
Process ID : pr_2
Instruction Register : load 20

Accumulator : 20.0                Temp : 20.0
Program Counter : 1               State : ready
loop :  6
block_process pr_1
loop :  7
run_process pr_1
this process is blocked
loop :  8
unblock_process pr_1
```

```
loop :  9
run_process pr_1
loop :  10
show_context pr_1
Process ID : pr_1
Instruction Register : sub 10

Accumulator : 10.0                Temp : 10.0
Program Counter : 2               State : ready
loop :  11
kill_process pr_1
loop :  12
show_context pr_1
process doesn't exist
loop :  13
run_process pr_2
loop :  14
show_context pr_2
Process ID : pr_2
Instruction Register : add 15

Accumulator : 35.0                Temp : 15.0
Program Counter : 2               State : ready
loop :  15
kill_process pr_2
loop :  16
show_context pr_1
process doesn't exist
loop :  17
show_context pr_2
process doesn't exist
loop :  18
```