



سیستم عامل

دکتر شریفی

پاپ و تاثیر نقش بور (دانشگاه فنی)

پیشخوان

۱. سسٹم عامل چیز ہے؟

۲. حرب میں سسٹم عامل ہے اور طریقہ؟

۳. جو ہر دو سسٹم عامل (OS) کا طرح و مساوی معاون ہے؟

پیشخوان

فالیت صدر و خفیر در کافر

۲ نمرہ Quiz +

اصطلاح ہے (سامل جان یا نرم افزار ہے)

۱ نمرہ ۸

۲ نرم افزار

۲۵ اور (باحت یا نرم افزار نہیں لود)

نرم افزار

۲۵ اور (باحت یا نرم افزار نہیں لود)

نرم افزار

### References:

1. Modern Operating System, Tanenbaum

2. Operating System Concepts Essentials, Silberschatz

3. Operating Systems: Three Easy Pieces

سیستم عامل : رابط سین بینانہ تکنوفلائرے و سخت افزار

کے ایجاد ایں رابطہ بین سیاست ( Politics ) و مکانیزم ( Mechanisms )

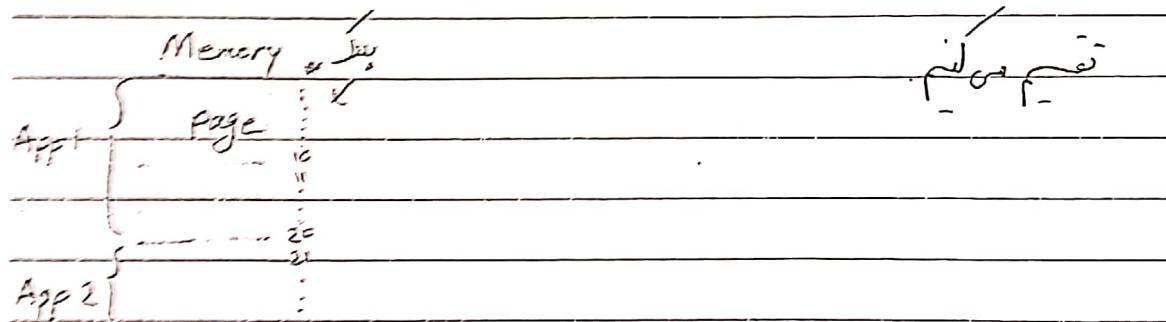
سیاست ( Mechanisms )

کے ایجاد مکانیزم ، بین سری

مال

Abstractness

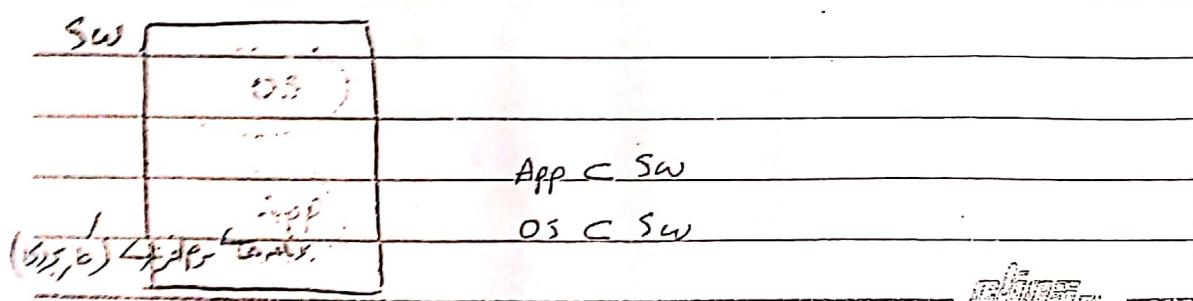
درست مکانیزم " درست حافظ " حافظ را بطور مجازی بین سرویز



دستوریں Policies

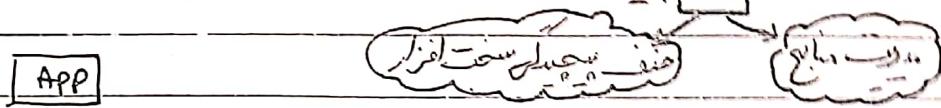
کیلئے جو حافظہ دوڑھا لے تو اس کی صورت میں یہیں اسے ادا کر دیں

جو حافظہ دوڑھا لے تو اس کی صورت میں یہیں اسے ادا کر دیں



اسسٰتھ عالیٰ جوست؟

نئو افزاں کے اس سے کہ سلسلہ من صفت افزاں و من افزاں کے طور کے اس سے



کیا ایسا ہوا موجود ہے میں OS, App, HW ایسا ہے؟

جو زبان میں APP و HW رابطے میں برقراہیم ترے لین طریقہ ہے

وہ سلسلہ باہر اہمیت این برقراہیکے راست کے اعمال کیم کہ سلسلہ رابطہ

اویسیم وجد وجد OS اور اپلائیشن App, HW رابطہ میں لکھ

OS این سلسلہ من صفت افزاں کے جنف میں لند کے مخبر پستے از

انواع لوگوں سخت افزاں کیں میں

کیک دیریت عرضیو یقاضا بیک میں میں سسٰتھ نیاز دیم

$$\text{سسٰتھ} = \text{عرضی} + \text{یقاضا} + \text{میں میں سسٰتھ}$$

$$\text{System} = \text{OS} + \text{App} + \text{HW}$$

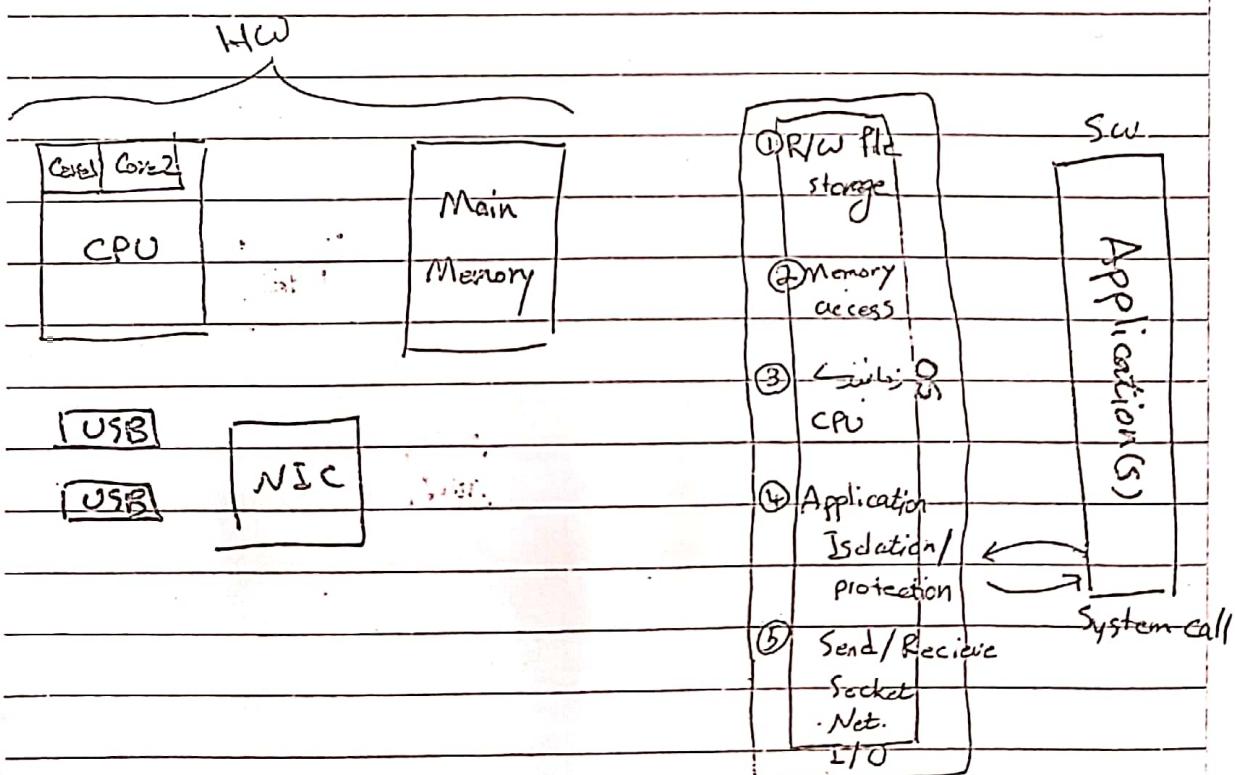
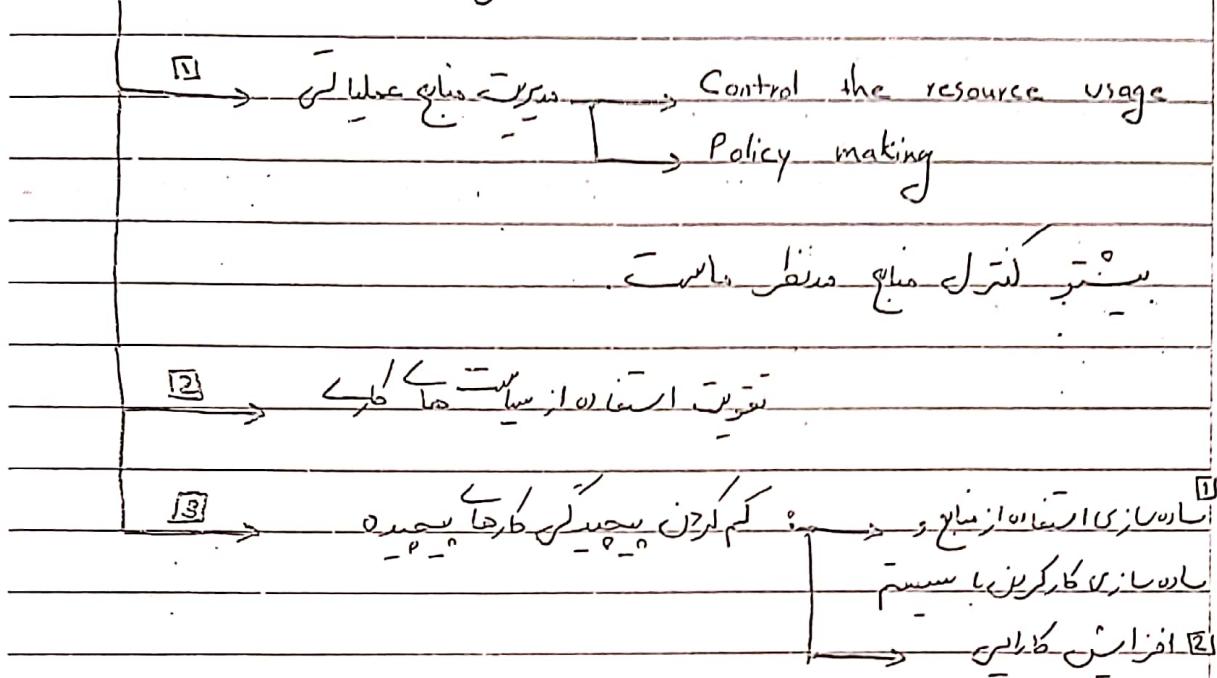
OS بہترانی System Manager ہے کہ طریقے احتمام من درج

III صرف نایع عملیات (لئے ہیوں بناہ، ریز، و سیستہ لذکر)

درائیں جا درست لئے ہیوں و سیستہ لذکر منظر اسے

CS

## OS : System Manager



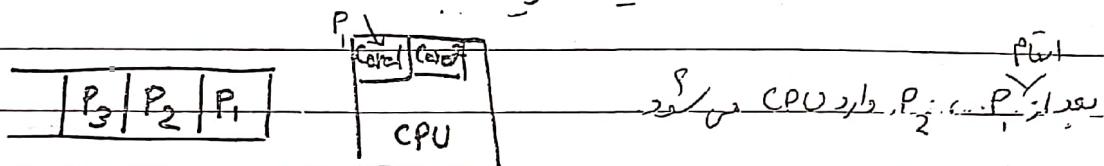
سُسِّيَرِ اَسَمَّاَنِ حَفَرَهَانِ قَابِلِ - اَجْرِيَ - مُتَّسِّرِ اَسَمَّاَنِ حَفَرَهَانِ قَابِلِ : Multi-task System

جَرِيَةِ اَسَمَّاَنِ حَفَرَهَانِ قَابِلِ

System Call

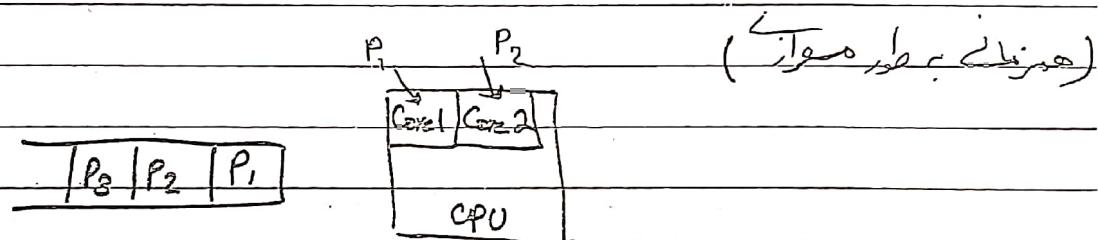
و هنرمان ۱ Concurrent  
۲ Simultaneous

: در آن واحد فریمی در CPU در حال اجراشون است Concurrent



(هنرمانی به طور سری)

: در آن واحد هم فریمی در CPU در حال اجراشون است Simultaneous



الدر ها اینکه نیز نهمان بود خشن از قبل از این بود

برنامه کاربری، برنامه کاربری از دسترسی داشتند

طريق

OS : فرآختن سیستم (عامل) System call

طاهر App های را با ارتباط با سخت افزار دارند که این طریق

OS با فرآخوانی می کنند (از طرف کارکننده)، از زوایه آن (زمانی)،

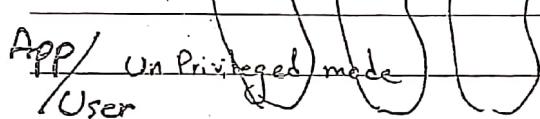
App از طرف کارکننده ارسال می کنند، از طرف کارکننده

لایه های سیستم

- کیا از این ۵ لایه استفاده خواهد شد؟

- طبق چه تعاریف از این ۵ لایه اخیر خواهد شد؟

- یارمینها کن راست نهایت workload



OS

HW.



USB

User-kernel switching :

لایه های

- System calls

- Trap instruction (کمک درخواست)

- Signal (رسانی ارسال میگردد)

## User-kernel Switching (transition) :

دستور ! لبرت دیر سخت اپلیکیشن تغییراتی ایجاد می کند

و سیمہ را بے App می فرستے

( HW Supported ) از طریق HW صدر می کرد ( ای جن Switching )

Overhead → ایجاد سریع → کاهش سرعت عمل

Switching locality → Affects HW Cache

## System calls :

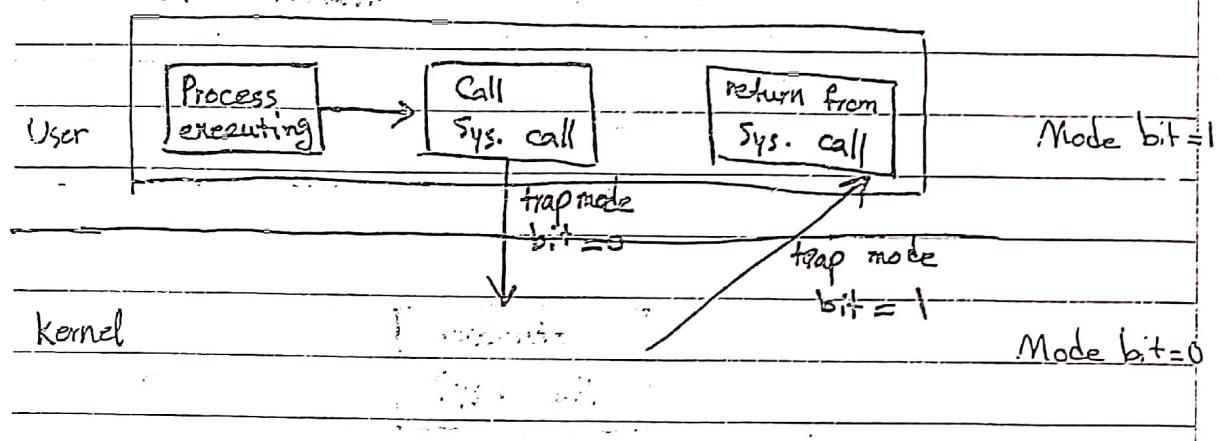
مرید پر سر لیسا خود System call کیل کوتے

پڑھنے نہیں، اجر فریس چوپ نہ کوئی

کسی System call ادا کئے فرائید وابستہ بے خروجی : Synch mode

" لزماً " : Asynch mode

مرید می خواهد بے خود دعاوی کرے



; System calls

او امراض کی اکھی نہیں

اپلیکیشن کے لئے App API, یعنی Functionality

Storage

Security

Process Management

Memory

File

### Windows Sys. call

Process Control

Create Process ( )

Exit Process ( )

Wait For Single Object ( )

### Unix sys. call

fork ( )

exit ( )

wait ( )

kill ( )

Windows Sys. callUnix Sys. call

<u>File Manipulation</u>	CreateFile( )	open( )
	ReadFile( )	read( )
	WriteFile( )	write( )
	CloseHandle( )	close( )
	SetConsoleMode( )	io_ctl( )
<u>Device Manipulation</u>	ReadConsole( )	read( )
	WriteConsole( )	write( )

برنامه در حال اجرا فرمه دارد

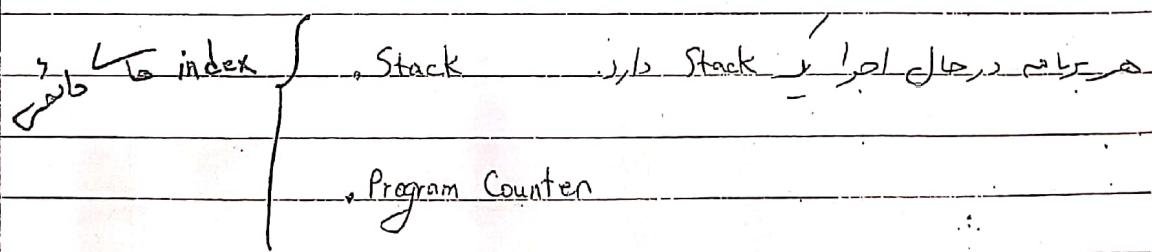
task / job

و مدرسه حسنه

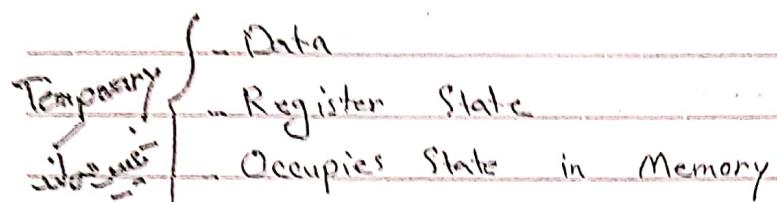
فرانسها جلوی ترکیب OS دارند

فرانسها جلوی ترکیب OS دارند (هر چنان) موارد

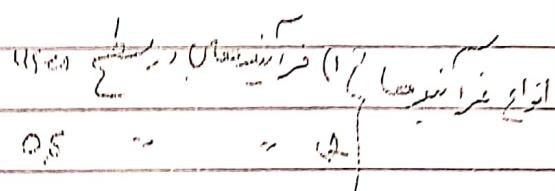
State of Execution :



## Parts and Temporary Holding Areas



### HCU State :



### I/O device Access

فرق بين فرائين و برامج : (فرق طرق فرائين و برامج في اجراءات على عنوان لروما و الحال اجراء شبيه)

برامح

فرائين

ـ حافظة جانبية خارجية.

ـ در حافظة داخلية (حافظة مجاورة).

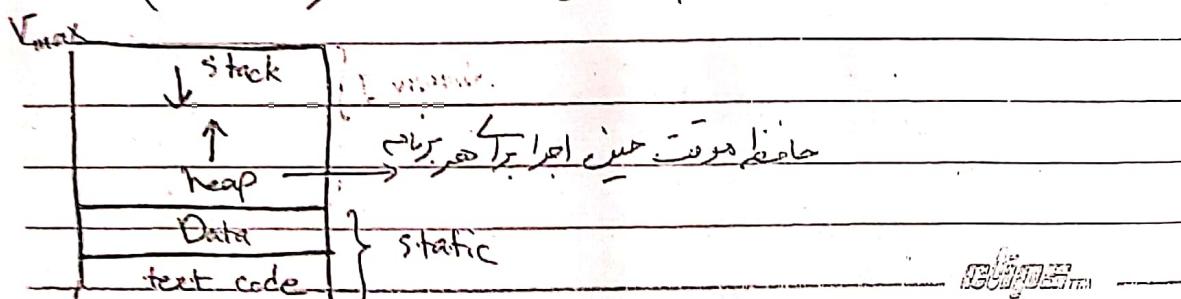
HDD, SSD, ...

ـ است Static

ـ است Dynamic

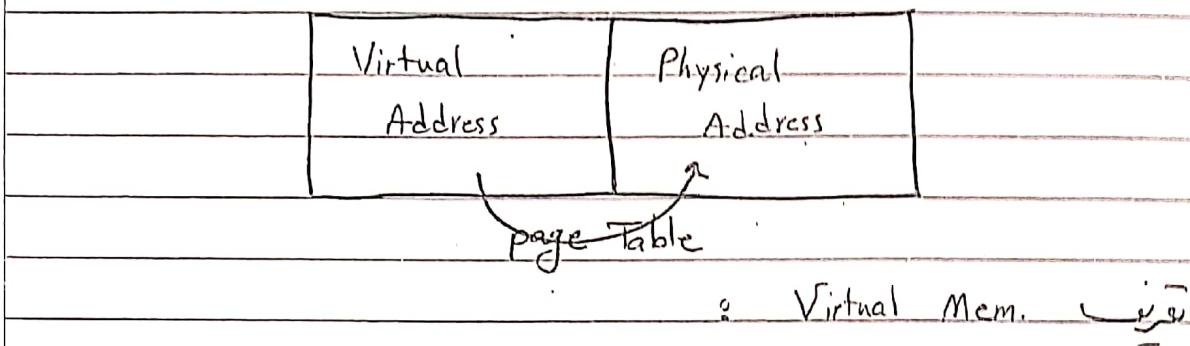
### Process States in Memory

(عمر)  $\rightarrow$  load  $\rightarrow$  برمجيات Data, text code



V<sub>min</sub> ..... Virtual Memory

بررسی To Data or heap , بررسی Instructions , بررسی Stack



عنوان از حافظه که برای نمایش داده شود است در حافظه

آن عنوان کوچکتر از حافظه ماست .

در فرآیند اجراهای زیر تابعیت داشته باشد :  
 $P_1 \rightarrow P_2$

فرآیند ها مربوط به این فرآیند ها در Virtual Add. space

$$P_1 \rightarrow P_2 : 0 - 64 KB (1)$$

$$P_1 : 0 - 32 KB, P_2 : 32 KB - 64 KB (2)$$

این دو فرآیند ها متعلق به Virtual Memory

این است که تفاوت دارند در موردی که حل احراز دارند

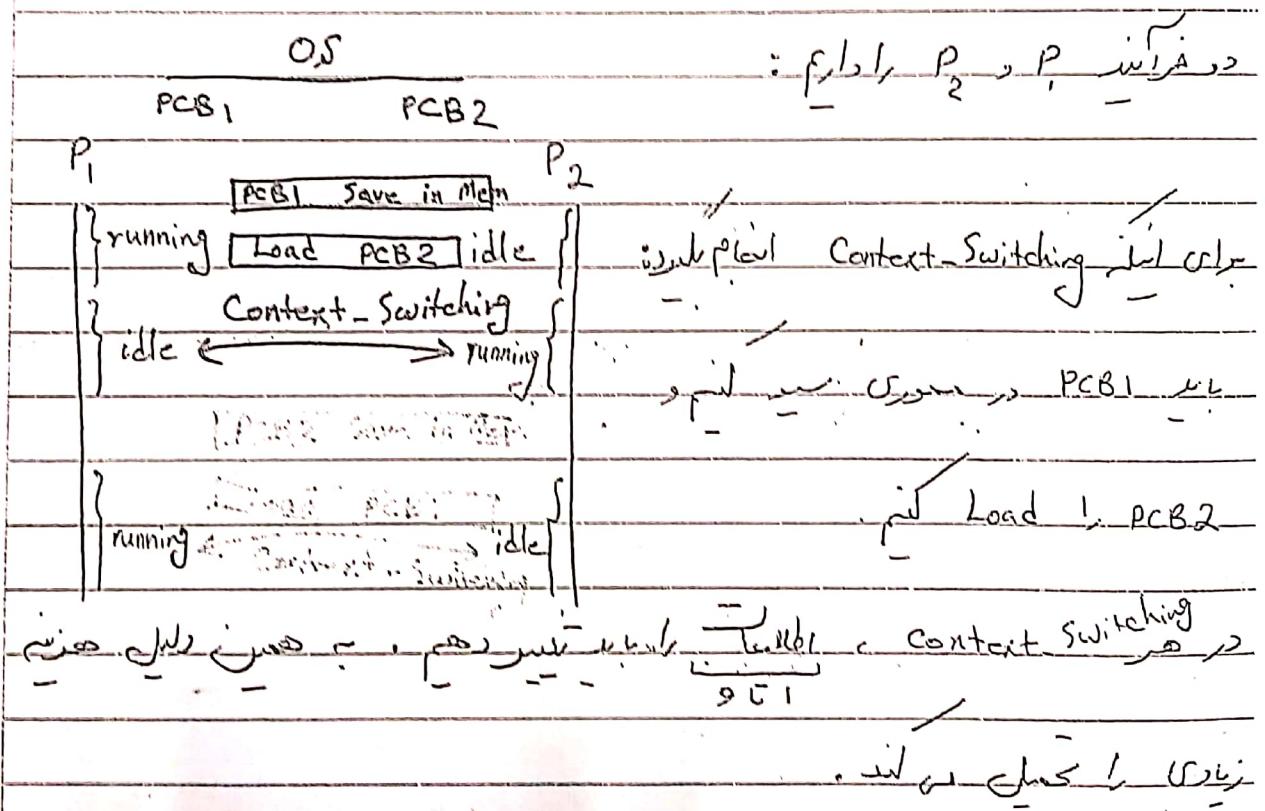
ثابت

این از نظر سرعان

## 2 CPU scheduling info

PCB لایو جیسے ہے اور فرائیں دھنے کے لئے OS کا کام کرے گا  
جسکے لئے وہ اپنے بارے میں PCB کا ایجاد کرے گا

## Two Context Switching PCB کے بینہ میں

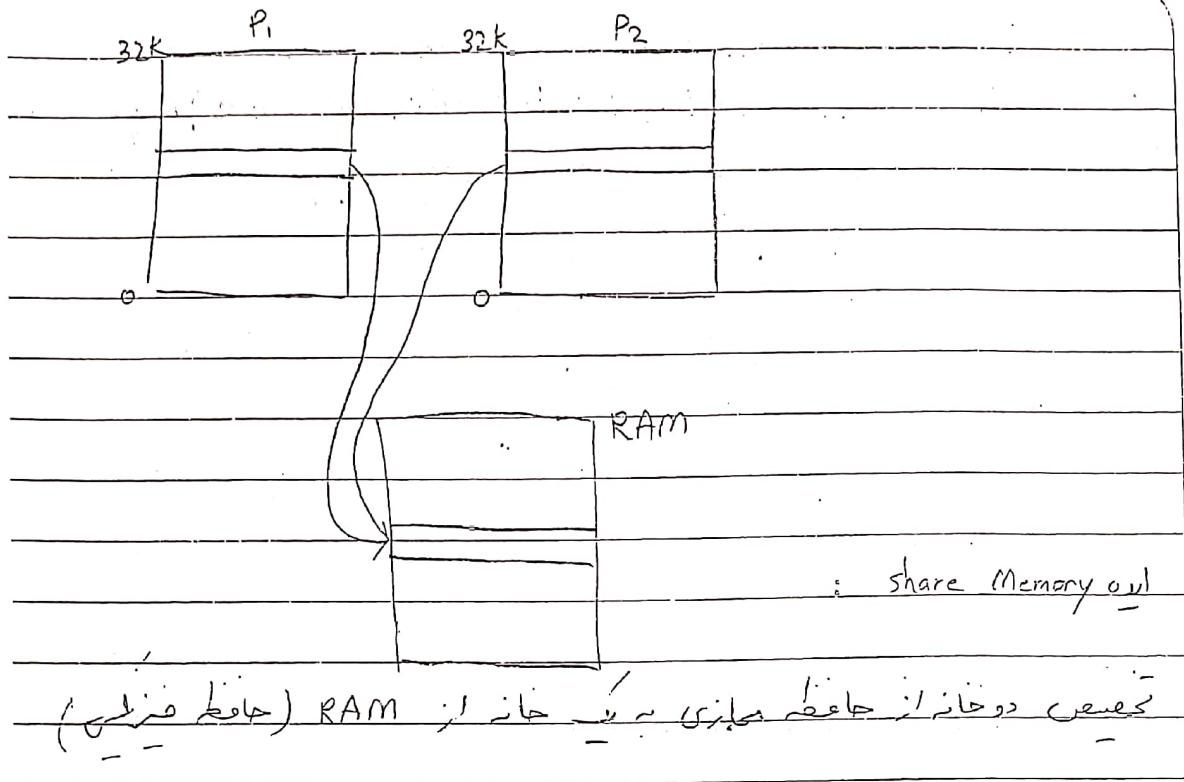


(To Register کے لئے مستقر) (حصہ کے مستقر)

(Cache کے لئے مستقر) (حصہ کے مستقر)

miss cache کے لئے مستقر (حصہ درخواست میں لیتیں)

Subject ..... Date .....



OS از دو حافظه خود برای فرآنش ایجاد می کند

با هم از فرآنش ایجاد می کنند

Program Counter (PC)

CPU Registers

CPU

Stack pointer

حافظه مشارک

PCB (Process Control Block)

1 Process State

5 Memory limits

2 Number

6 List of Open Files

3 PC

7 Priority

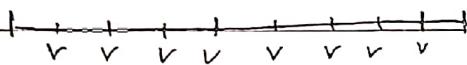
4 Register's

8 Signal Mask

لیٹریا، اسے time slot کہا جاتا ہے

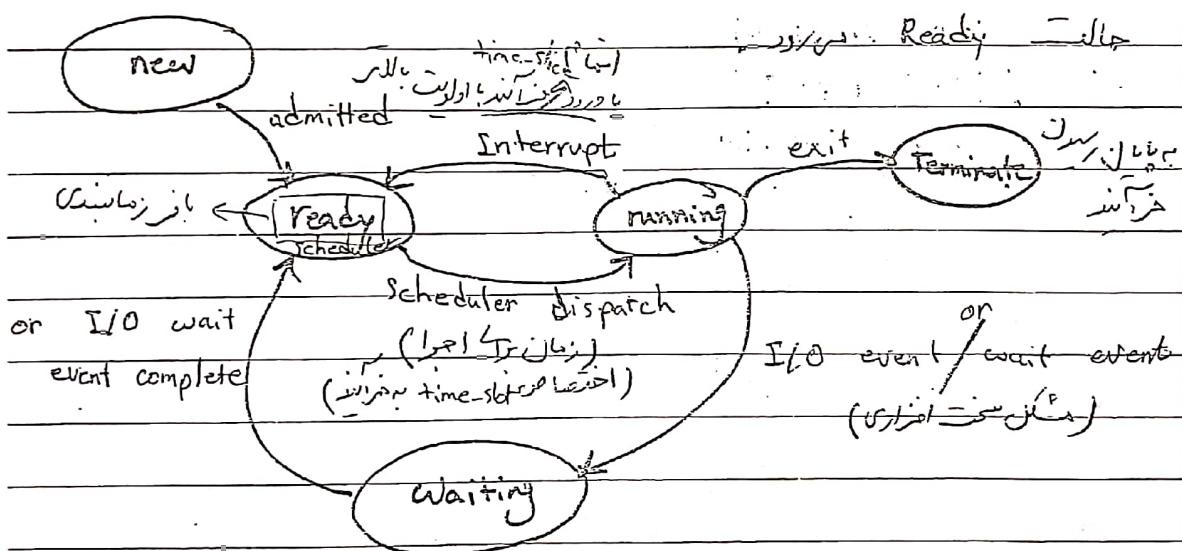


L2 context-switching

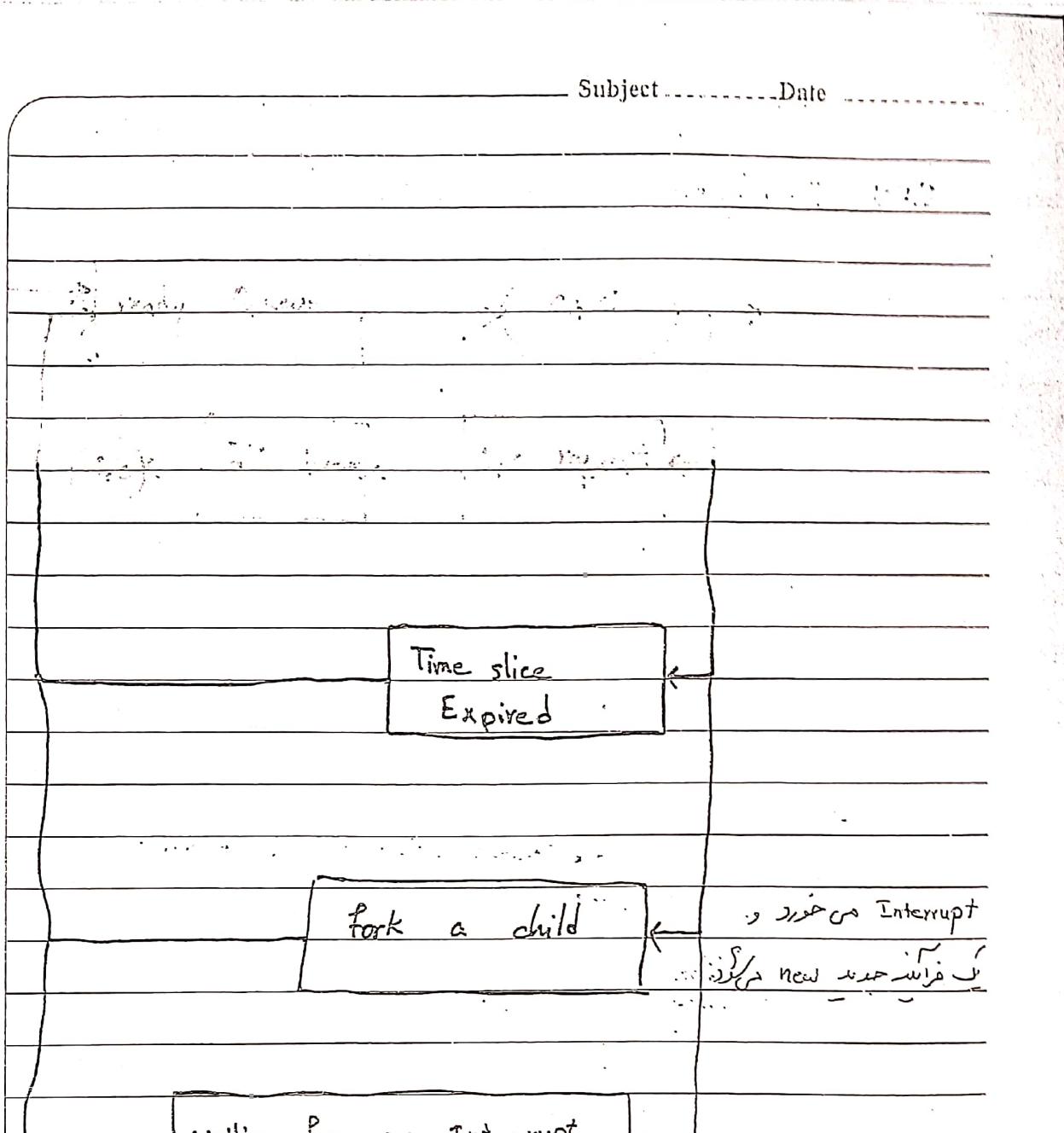


L8 context-switching

PCB میں تکمیل کی جانے والے OS کے مرضیوں کا لیٹریا



Process life cycle



صُبْحٌ هَلْسٌ لِّزْ إِسْتَهَادْ سِرْكَرْدْ مِنْ احْتَالَ الْنَّ وَجَدْ دَارِرْ

running ready

100٪ احْتَال اسْتَهَاد

CPU از CPU وَجَدْ

اسْتَهَاد دَارِرْ

شِنْ لَنْدْ

CPU Scheduler

نهضه مولن کلایم از فرآیندها به درحالات

هم این زمان باشد در برقراری سوچت

برود running حالت ①

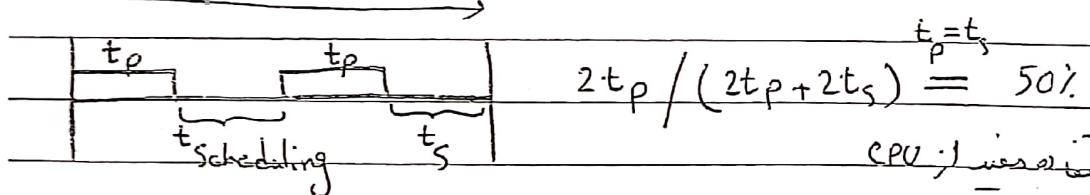
اول هم مدتی در حالت run باشد



Context switching + Scheduling

Time  
Slice

CPU



برای افزایش راندمان باشد  $t_p$  را ظاهراً بزرگتر از  $t_s$  افزایش دهیم

$T_p \uparrow$

$\uparrow T_p + T_s \downarrow \rightarrow$  در مرحله طراحی و پیاده‌سازی

Scheduler

ولنی این کار جملن است باید ۹۰٪ از فرآیند هاتر (تکریه) داشت

 $t_p \uparrow$ 

بسته در صفت انتظار پرسکنند

(مثل این امر را که نظر در صفت ناواریش نیز هم زیادی می خواهد)

بنی افراد باید می توانند تازیت داشتن برخوبی

نمایران باید مقادیر  $t_s$  را مشخص کنند

$t_s$  بین  $t_p$  و ۹۰٪ CPU می باشد که بخواهیم

$$\frac{t_p}{t_s} = 9$$

$$\frac{90}{100} = \frac{t_p}{t_p + t_s} = \frac{x t_s}{x t_s + t_s} = \frac{x}{x+1} \rightarrow x = 9$$

### Design Decision

1. What are appropriate time slice values?

2. Metrics to choose next process?

باید این دوره های بینی (کمترین) را داشتم

Refugee

## context switching ماحل

① preempt

P - تحرير الملف

② schedule

P گلچینی زمان

③ dispatch

P بایان زمان و حافظه

P - تحریر الملف = Scheduling : کامپیوٹر حزیر وظائف

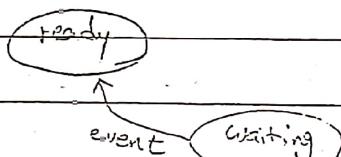
1. Maintaining the I/O Queue X I/O Device

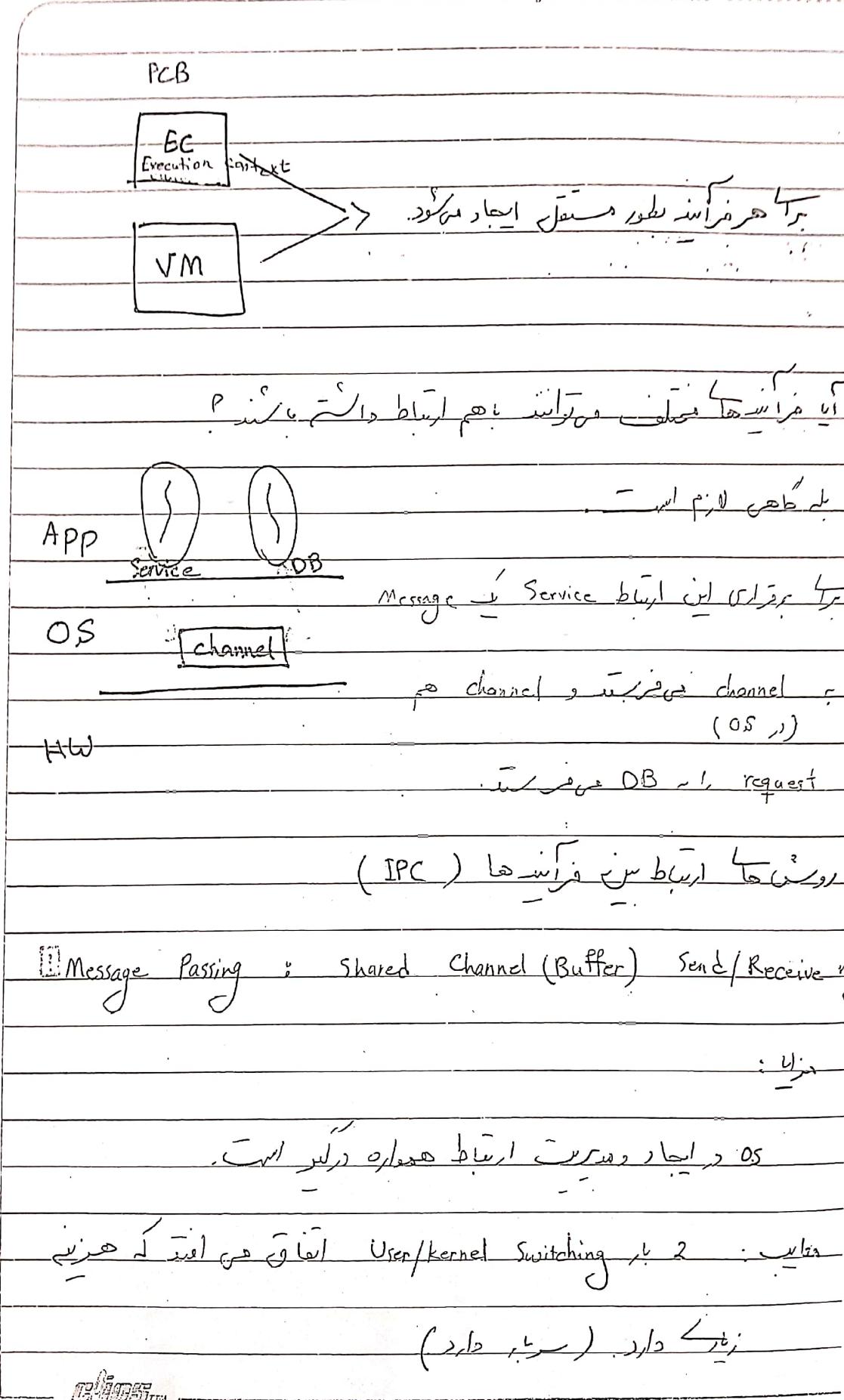
2. " = Ready Queue ✓

3 Decision on when to context switch ✓

4. " = " generate an event that a

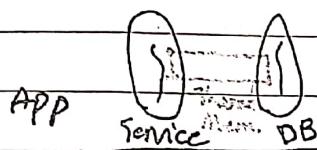
process is waiting on . X





Subject ..... Date .....

## Shared Memory



OS

HW

هذا :

برای اینجا . Msg Passing طریق است بروز رسانی

اتصال در اینجا Mem. Mapping.

های :

سیستم پاره سازی (نقطه ها)

OS دلخواه دارد و اینکه لذت برداری توسط خود App ها نماید

فرمود که باعث بالا رفتن سیستم می شود

لذا این چیز است

آنچه اند رفتار ارتباط سیستم App لفڑای (۲۳۰۰)

شیوه ایجاد رفتار Shared Mem. بر اساس دفعه ای این اینست مترجم داده می شود

BLUES

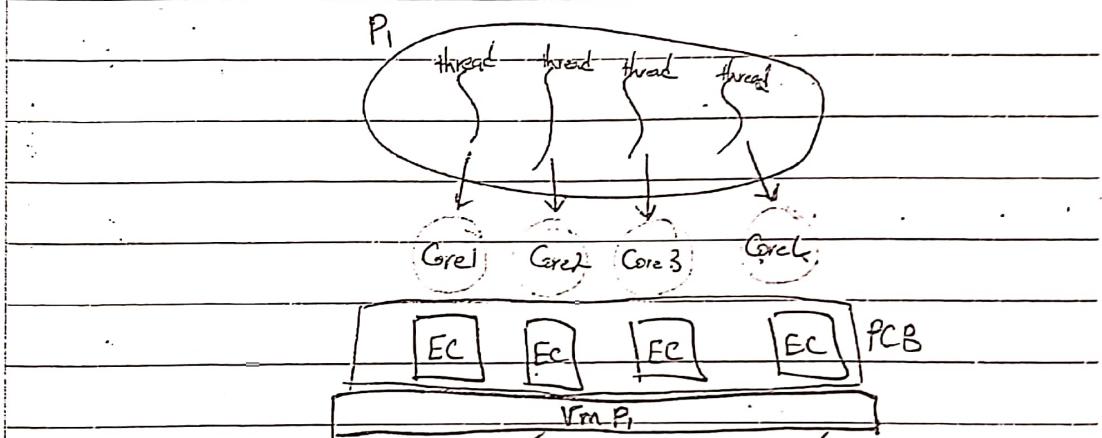
Thread

در multi-core CPU که از چند Thread است

هر یک Thread باید از یک Core است

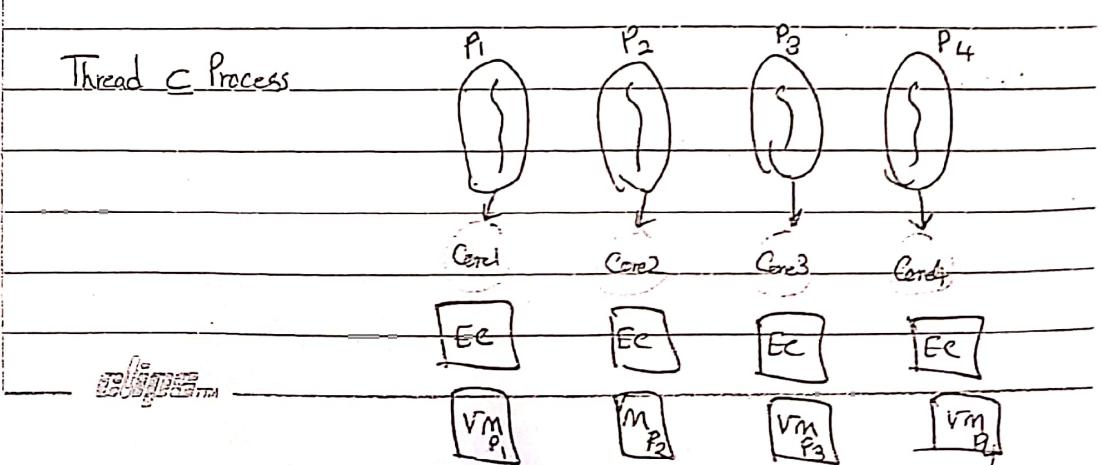
لایه را بالاتر فرمود

درین حالت برای هر Thread حداهم داشت



این خدمت اینهاست که یک Thread در یک Core دارند

برای یک Thread یک Core دارد

Thread C Process

Process ، Thread فریق سر

اچرا من کسی Process از واحد از Thread

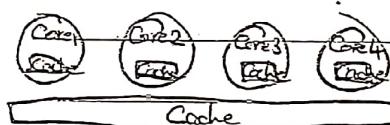
Thread می تواند در یک VM بین Process های

فرانسی خواهد داشت

Multi-Threading

عوایض از این سرعت اجرا

کاهش زمان اجرا



Hot Cache ← Hit rate افزایش

افزایش طبقه

بر دلیل این دو، نکل سرعت حافظه و ارتباطین thread

T<sub>1</sub>

البروگرایر از T<sub>1</sub> در میان اجرا در مرحله اجرا

Time-slice  
I/O Request

حالات T<sub>1</sub> باید I/O request

Single - Core

اصحاق در T<sub>2</sub> Time-slice می بود، اوضاع waiting

\* \* \* این طریق چیزی بهتر نخواهد بود؟ (فکر کن)

Chips

I/O مہر لے کر (T)

request -

کوئی Multi-Threading ایسے ilj میں OS کا (T)

اپنے اسی سیس کال میں ہے " در حال ت

thread کے core کے ilj میں Sys. call میں Multi-core

کام کرنے کے لئے

: Context Switching →

( منہج کے ) Thread , Process , EC  
Switching      Switching

اجمیں Process میں ( ) Physical Mem ← VM  
Switching      ↓  
Mem. Mapping

Process Switching باریں

: الگ

? f1, f2, f3 in thread ہے جسے چھوڑ دیا جائے ہے

An Introduction to Programming with threads, Birrell, 1989

EDITION

Subject ..... Date .....

## 1 Thread Data Structure

2

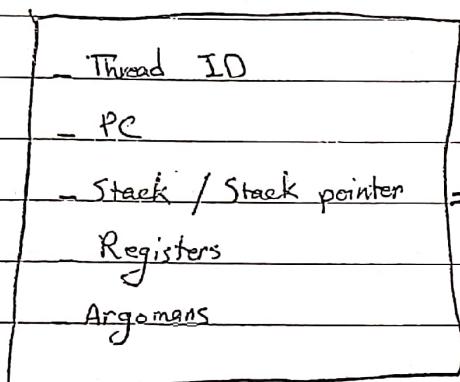
Thread میں ہے جو کام و ایجاد کرے گا /

3

کو thread سے امن سے ارتقا کرے گا / میں میں ہے جو کام کرے گا

جسون از حافظہ متذکر است یہ میں لے لیں گے

Thread  $t_j$



الآن ہم تھیں درستھان دیں

یہیں ہم اپنے وحیور دلائے گے thread

## Concurrency control & Coordination (متزوج سازی)

اور دو راه اسٹریم منزد :

1. Mutual Exclusion (ذترس اکسچن) : (اصحیح مطالعہ)

بے آئندہ مخفجہ VM

Mutex

2. Waiting

Condition Variables : ہمیں ہم کریں از طریقے

جو یہیں میں میں Condition Variables ہیں

Thread  $t_i$  ;  
DS

Fork ( proc, args )

دیگر child thread ی

Create Thread ( \_\_\_\_\_ )

Join ( Thread )  $\rightarrow$  child-thread  $T_1$  ی  $\rightarrow$  parent-thread  
 دیگر child-thread  $T_1$  را اجرا کر

: جملہ

$T_0$

{

$t_1 = \text{Fork} (\text{proc}, \text{args})$

{

$T_1$

{

child-result = join( $t_1$ )

return result;

end

$\rightarrow$   $T_1$   $\rightarrow$   $T_0$

: جملہ

Thread  $t_i$  ;

$T_0$

Shared-list list ;

{

$i = \text{Fork} (\text{safe\_insert}, 4)$

$\rightarrow T_1$

safe\\_insert(4)

دیگر child thread ی

$\rightarrow$  safe\\_insert( $i$ )

دیگر child

```
void safe-insert (int n) { insert-into-list (x); ... }
```

Thread t <sub>1</sub> ;	To
Shared-list list;	{
Safe-insert (5);	Safe-insert (5)
	}

Thread t <sub>1</sub> ;	To	T <sub>1</sub>
Shared-list list;	{	{
t <sub>1</sub> = fork (safe-insert, 4);	core1	core2
Safe-insert (5);	5 4 4 5	5 4 4 5

نحو ب سرعت ملائى لـ core 1  
اس ترتيب ادى الى

List → 4 5 : من اى دارتمان order اى

t<sub>1</sub> = fork (safe-insert, 4);

join (t<sub>1</sub>);

Safe-insert (5);

: من اى دارتمان 5 4 ترتيب اى

1) Safe-insert (5);

Safe-insert (4);

2) t<sub>1</sub> = fork (safe-insert, 5); join(t<sub>1</sub>);

Safe-insert (4);

لذلك ملائى لـ core 1

Eclipse

کہریں را حل ایجاد مساحت میں

### Mutual Exclusion

Mutex Construct

lock (mutex) {

critical Section

lock (mutex);

critical Section

unlock(mutex);

} unlock;

Birrell's lock API

(لئے ان خطای زندگی میں  
لئے ان خطای زندگی میں  
lock threads کو کہہ دیں)

lock (mutex) دیتے ہوئے  
lock (mutex) دیتے ہوئے

ذکر نہیں کرو

(4 قفل 5 حافظہ 5 حافظہ 5 قفل 4)

list <int> my\_list;

Mutex m;

Void Safe\_insert (int i) {

lock (m);

my\_list.insert (i);

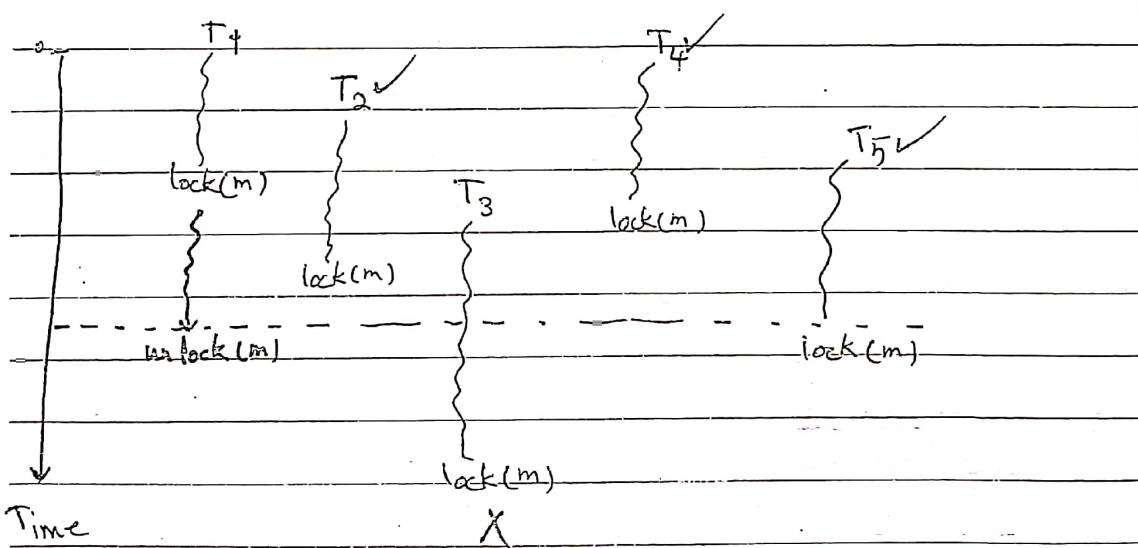
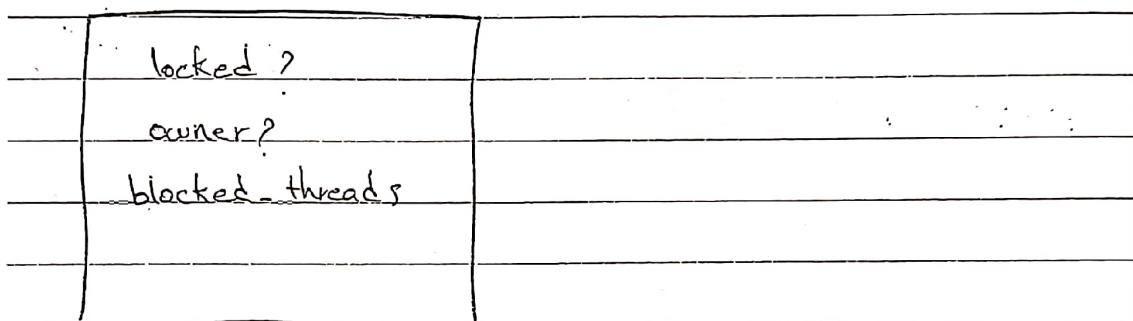
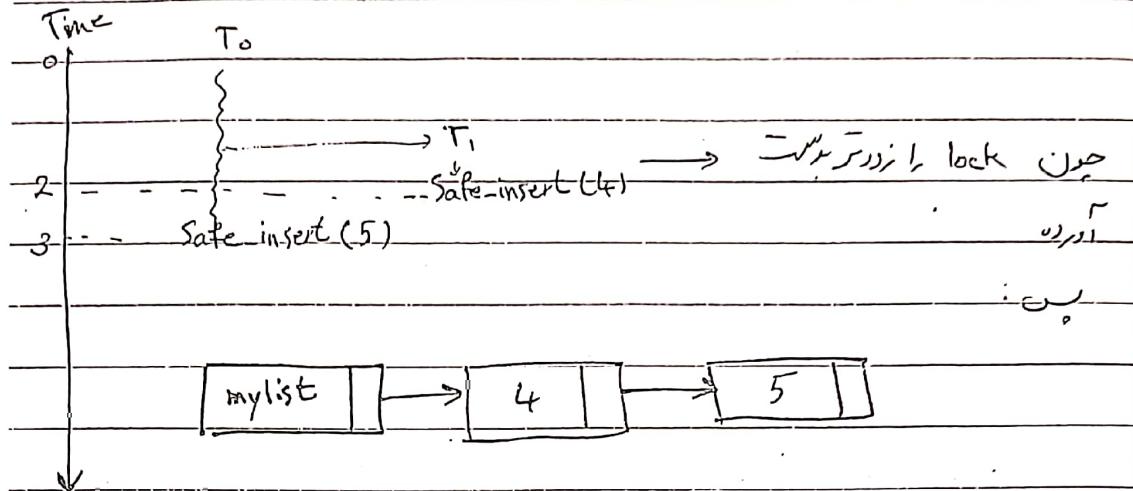
} // unlock

}

→ اسی طرح، lock اور unlock thread کو کہا جائے گا

eclipse

Subject ..... Date .....



S/critical Section w/  $T_2, T_4, T_5$  Random

SCENE

waiting → condition vars. (C.V.)

- جل

تولیم میزند از مردم بخواهند Producer/Consumer را

Many  
Producers



..... اینجا از اینجا

اینها کسی نیست

نیز اینجا اینها نیستند اما اینجا از اینجا

اینها نیستند اما اینجا از اینجا

// main

for i=0 ... 10

producers[i] = fork (safe-insert, Null);

consumer = fork (print-and-clear, my-list)

// producer : safe-insert

lock(m) {

list → insert (my-thread-id) !

? if unlock

// consumer : print-and-clear

lock(m) {

while (my-list-not-full(l))

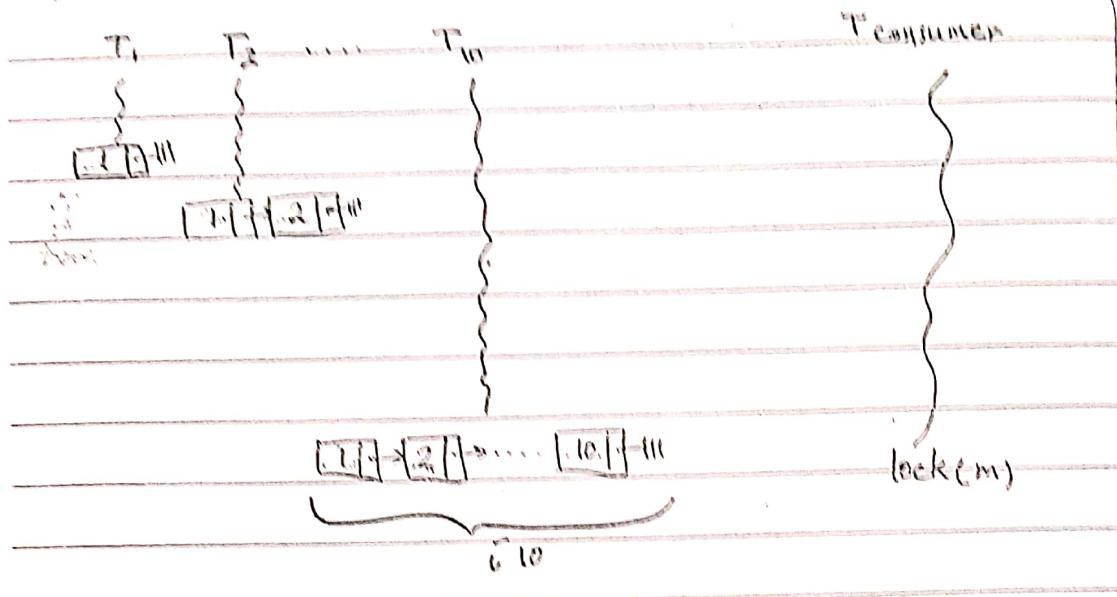
[wait(m, list-full())] C.V.

my-list.print-and-remove-all()

? if unlock

clip

Subject ..... Date .....



فی حالا  $m = \text{wait}(m, \text{list full})$  اینجا

برای  $T_i$  که آنرا  $lock$  نموده بودند  $\text{list}$  نباید  $lock$  شود

لیست باشد  $lock$  شود

Consumer داشت بلطف شرکت

// Consumer

lock(m) {

if (my-list.full)

print-list();

clear-list();

else

release-lock-and-try-again-later()

? // unlock

aliquan

P Consumer کامپیوٹر سائنس اسٹ کامپیوٹر سائنس اسٹ

کامپیوٹر سائنس اسٹ

## C.V. APIs

Condition type

ارسال کرنے والے thread کا لئے

one one transmission to threads =  
one one condition or

wait (mutex, C.V.) {

// releases mutex automatically

// goes on wait queue

wait

wait

// removes the thread from queue

// re-acquires mutex

// exit the wait operation

}

clip

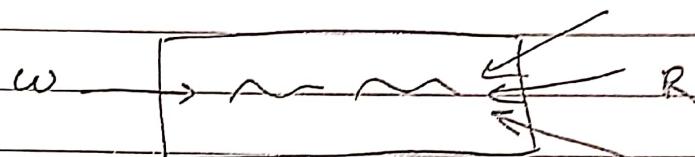
## Condition DS

- Condition Variables
- mutex references
- List of waiting threads

## Readers/ writer Problem

که میتواند ویرایش دارد که دو Reader که میخواهند از آن استفاده کنند.

لکن : که این میتواند بسیار ریسید و حذف نمایند برای ایند.



زمانی که حیزک زسته میشوند Reader ها نمیتوانند حیزک بوزیرند

ویرایش

با خاصیت Reader از خاطری مسترد میتوانند کرایند . الی حداقل

بر خواسته بسته خواندن باشد اینان زستن وجود شاره و نویسه

باید تایپ ایجاد شود . سیمی باید نویسه و خواندن در خصوصی  
خواندن (ها)

مسترد نمیتواند رفع نویسه مسترد نوشت است ، هم خواسته ای

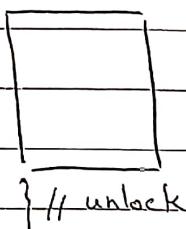
حق دسترس به معنای مسترد را ندارد





ساده ترین راه :

lock (m) { : mutex لک (l



لک رین : حمزه مان چیز نمیتواند Reader چیز را

درست داشته باشد

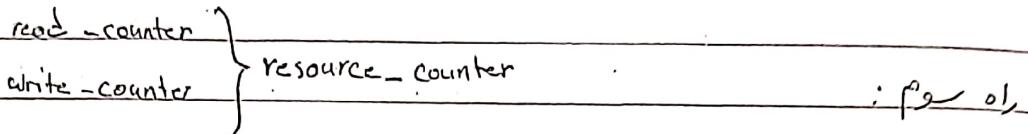
دوین راه :

Reader { if (read-counter > 0 ) R → OK  
free } if (read-counter == 0 && write-counter == 0 ) R → OK  
W → X W → OK

Writing { if (write-counter == 1 ) R → X  
W → X

این

Subject ..... Date .....



if (resource - counter == 0) // free - resource

R → OK

W → OK

if (resource - counter > 0) // Reading

R → OK

W → X

if (resource - counter == -1) // Writing

R → X

W → X Reader ينوي القراءة - بمعنى أن resource - counter = 0

Writer ينوي الكتابة - بمعنى أن resource - counter ≠ 0

(C.V) : فواید

حالت طرز :

lock(m)

while (! predicate - OK)

wait ( → )

update state

Signal / broadcast

} // unlock

clipse

// Reader(s)

```

lock(m) {
    if (resource_counter == -1)
        while (true)
            wait(m, read-phase)
    resource_counter++;
}
} // unlock

```

Reading

```

lock(m) {
    resource_counter--;
    if (resource_counter == 0)
        signal(write-phase); Readable default abc
}
} // unlock

```

Broadcast to Reader

// Writer

```

lock(m) {
    while (resource_counter != 0)
        wait(m, write-phase)
    resource_counter = -1;
}
} // unlock

```

Writing

```

lock(m) {
    resource_counter = 0;
    broadcast(read-phase);
    signal(write-phase);
}
} // unlock

```

Clip

Subject ..... Date .....

: signal or broadcast

از نشاط قفسی رفع کند ملک سمت لامورسی  
کاراپت

: broadcast or Signal

خط بار دارد Read

این امکان جلوگیری کند . باعث میکنی  
unnecessary wakeup ;

در این مرد

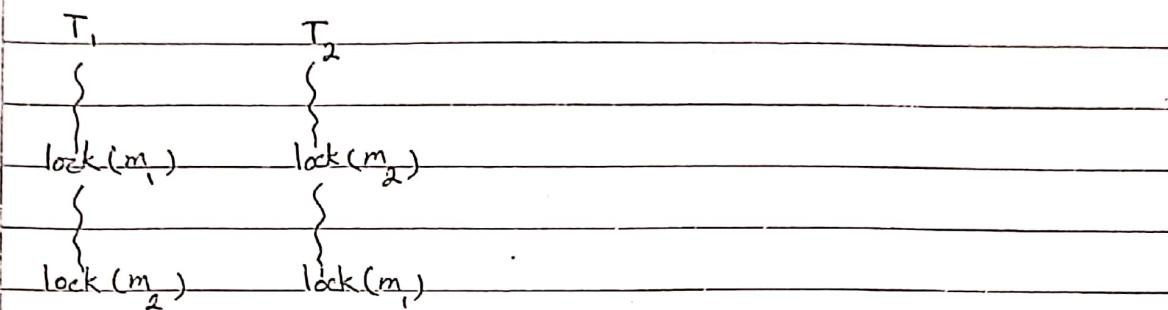
Deadlock ( بسته )

$T_1$   
lock(1)       $T_2$   
lock(2)

$T_2$  را آزاد نمود ،  $T_1$  بهترین لذت

آزاد نمود ، lock :  $T_1$

Le Bridge



Deadlock!

از اینجا به بعد اینجا بینست مهر

چنانچه Deadlock نیز اینجا پیش آمد

Avoiding Deadlock

1) fine-grained locking

همانند بیانم که از تاکنون lockها برای یک thread باید رویکردی را (حصتی)

که قابل حاصله در دلترین حالتی بوده باشد

2) Mega-lock

3) Maintain lock order

البرگراف مربوط به thread های ماهوی، حلقة ای و خود را نهاده شده است

برآن امر است که حلقة را از سری برم

دله چون هر سه دلت سری، زیرا (۱) دارد، اجرا و مفهوم

آنکه بقید و بعد از کاربریت کنم

CS

Dead-lock دلخواه

(برلت) roll-back

(الگوریتم ترجیح) Do-nothing

reboot 1, to thread

Multi-threading Models

(User, kernel سیستم)

Web Server

User

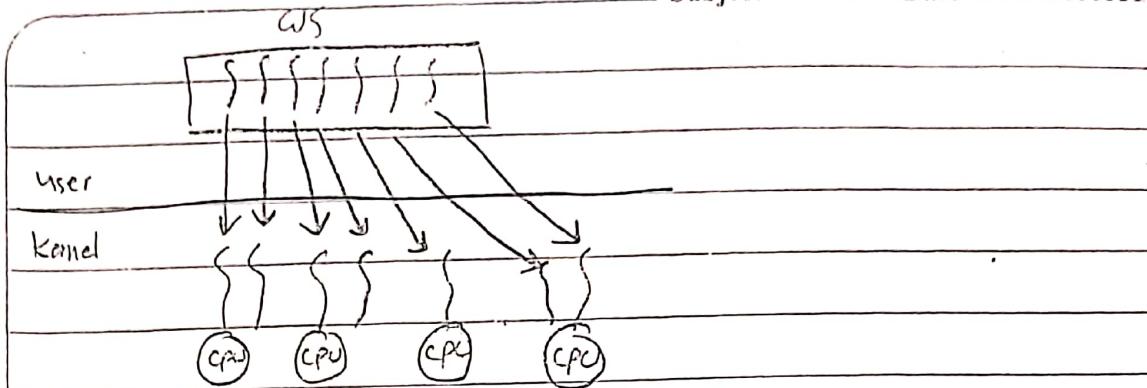
Kernel

کامپیوشن و مکانیزم multi-thread OS گیری از این اینستیتیو

اینستیتیو، داده ها را

One-to-One Model

user will kernel یک یک thread بر user یک یک thread



هزنا و دلایل:

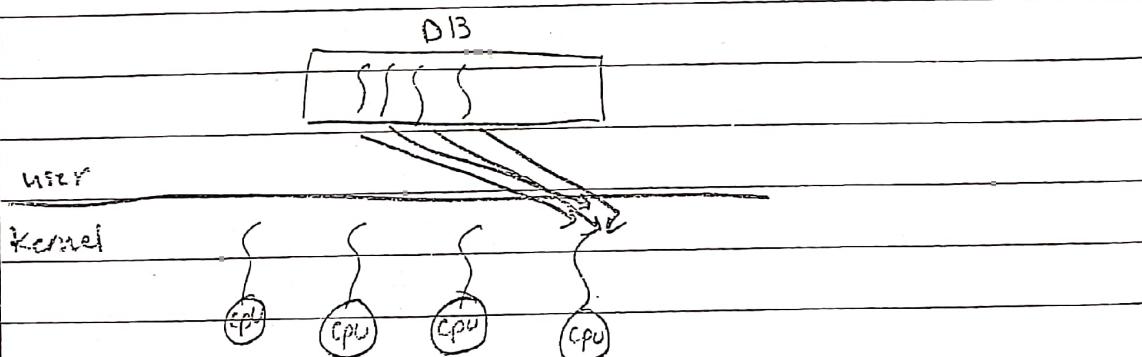
دوست اخراجی thread (OS + بین خود طبل) می‌داند.

برای طبقه کارهای app مراجعه نماید.

با این طبقه سازندهای OS در اخراج سرویس‌های ویندوز و مک‌وسی‌ای اند.

آخر OS نیز کرد باز app خود را در OS می‌خواهد اما دستورات اخراجی، thread (OS + بین خود درین دستورات و قدر) محدود نماید.

### Many-to-One Model



بین thread می‌باشد App فرایند می‌باشد که از thread می‌باشد

OS می‌باشد

Group

جزءاً و معايير :

+ لزوم ندارد برای طبقه OS را app. های که اجراء می‌کنند

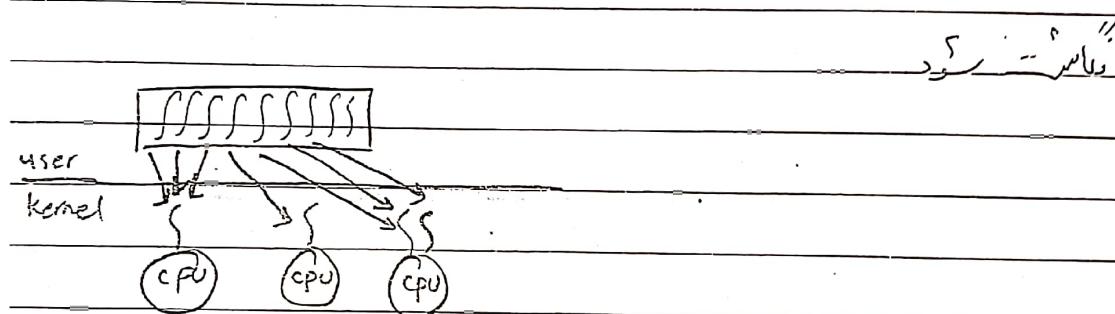
+ لزوم ندارد از حلیم سیاست های OS در اجراء می‌کنند

معایر اجراء احرازی thread بودن

طابعه پایه تر من

### Many-to-Many Model

kernel بین thread بین user بین thread بین



جزءاً و معايير :

معایر دو روئین دیگر را ندارد

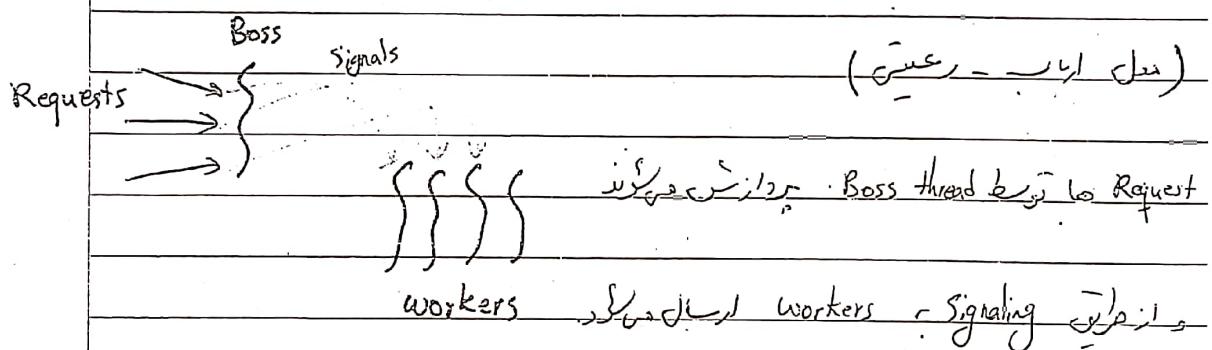
Coordination between User level threads and kernel level threads

is required

(Thread mgr در سایر طاری های که درین مدت می‌گذرد library داری خواهد کرد)

Thread mgr (cpu Scheduling , ?)

resource

Multi-threading Patterns( kernel job )Boss PatternPipelineLayeredBoss PatternBoss will manage workers using threads, Boss will threadmap to worker using job schedulingPerformance is limited to the performance of Boss thread.Throughput = 1Boss\_time\_per\_orderBoss thread → Bottle neckWipes

How many workers?

- On demand (dynamic)  $\rightarrow$  Delay!

(مساوا)

جدا عالی مسافر اینجا worker را کویی نماید

بخارا، موافقانه دستور دارد

Pool of workers:

Static Size

Dynamic Size

اگر idle worker نیست همان حالت است

که worker بتوان سفارت و خدمت را بخواهد

client Server می باشد

Pipeline: احجام همکار را جنسی آن که نرم افزار

Task

1	2	3	4
1	2	3	4
5	6	7	8

T<sub>1</sub> T<sub>2</sub>

T<sub>3</sub> T<sub>4</sub>

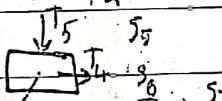
هر thread یک طبقه ایجاد می کند

که در آن همه این تابعیت ندارند

Sequence of sub-tasks

Stage

Shared Buffer



T<sub>2</sub> T<sub>1</sub>

elipses

Through <sup>put</sup> = توان پردازش قدرتی ترین قسم

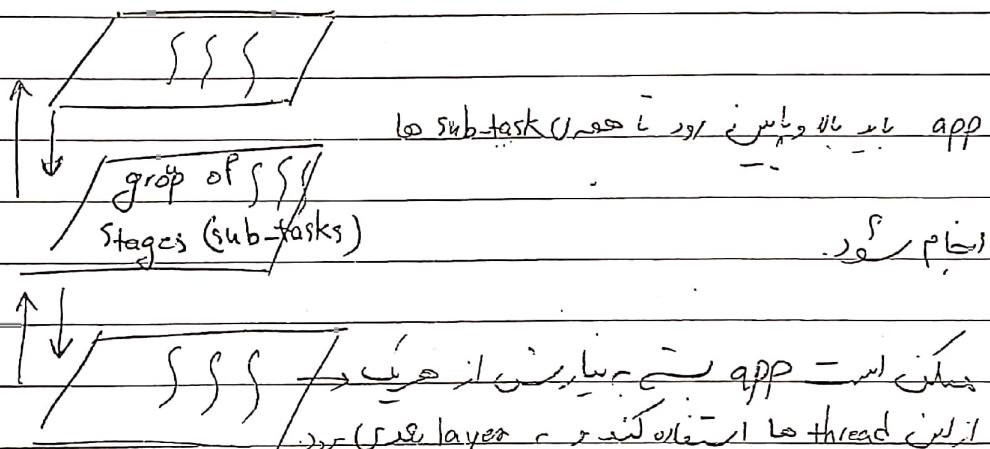
(Local Task) Specialization  
اچرا محدود

balancing & Synchron Overhead

جهت افزایش دسترسی به Locality +

متوجه شوند

Layered :

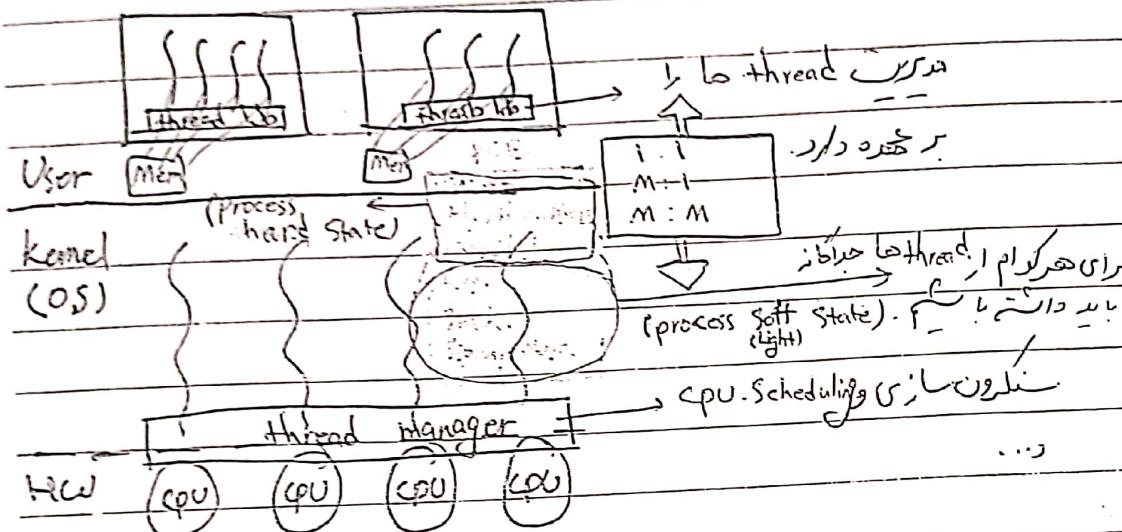


Specialization +

Course grained sub-task division

Not stable for all apps

متوجه شوند



نحوه خرائط (soft thread state) ← process hard state  
نحوه خرائط (soft thread) ← process soft state

### Signal PCB

### Multiple Data Structures

large continuous DS ✗ Scalability ✓ Smaller DS's

Private for each entity ✗ Overhead ✓ Easier to Share

Saved/restored on each Context Switch	✗ Performance	✓ User level library only needs update portion of the state.
	✗ Flexibility	✓

نحوه خرائط (soft state) ←

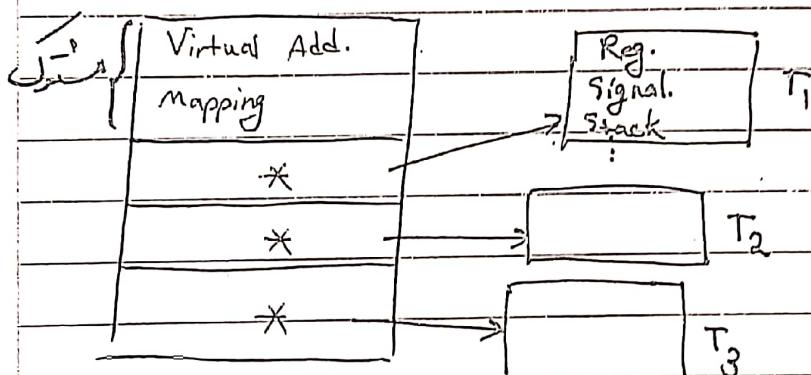
پیویس لیبیری (user level library)

User Level Thread (ULT)

- ULT ID
- UL Reg.s
- UL Stack
:

Kernel Level Thread (KLT)

- Stack
- Reg.
:
:

Signal PCBMulti Threading

Beyond Multi processing : Multi threading the sun OS kernel  
Ey Kholt

Implementing Lightweight Threads Stein & Shah

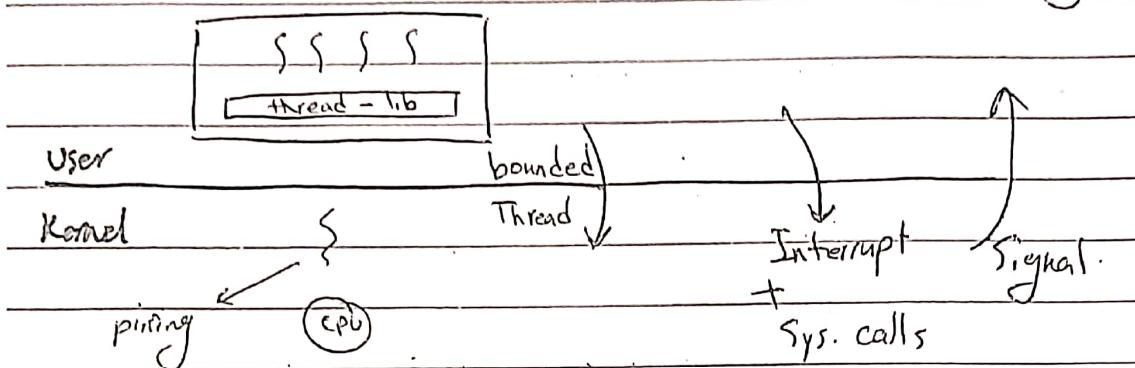
Clipper

• pin on kernel side, thread  $\rightarrow$  user side thread

بَلْنِ مَلِنْ كُورِنْ bound

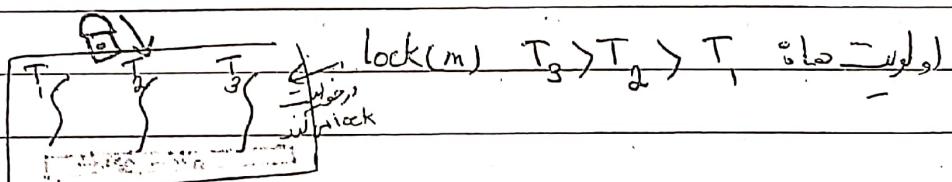
pin side  $\rightarrow$  bind side CPU  $\rightarrow$  kernel side thread

كُورِنْ



User side  $\rightarrow$  kernel side  $\rightarrow$  User side (OLT, KLT !!)

user side  $\rightarrow$  signal  $\rightarrow$  interrupt  $\rightarrow$  kernel side



user

kernel

T<sub>2</sub> { notification } T<sub>1</sub>



خارج من المدى  $\rightarrow$  user  $\rightarrow$  تفريغ المدى

CS

$T_1$  عملی را آزاد کر لند و بررسیتے تے  $T_2$  اور بعد جب حال  $T_2$  چلے

$T_1$  کی kernel notification دار،  $T_2$  کی timer دار

بعد تے  $T_2$  سرکاری بحاجت  $T_1$  دار  $CPU$  کو درستینج بکار لند

سوال: ایسا از دفعہ ہم اعلان نہیں کرتے بلکہ انہیں ہم

اعلان بخواہیں؟

نہ کھر نہیں، اگر ایک لئے بیدری، باسی مددگاریت ہے اسی ایں اعلان بخواہیں

برعوہ thread lib کے ساتھ کہ ممکن ہے کہ میریت آئیں نہیں برعوہ طریقے

لے کر ایسے کیمپلی کے ساتھ سروے کرنا ہے

جنہاں میں کارکرداں کا لام:

درسترنیک تعداد context switches از تعداد cycle ہے

متاثر میں میں ایسے Content Switches معمولی ہیں کہ کسی کو واحد بخواہیں۔

لیکن اسی کو ماہست دیکھنے میں دھم کر کرنا ہے

تعداد ہے cycle میں سرکار لئے کہ  $CPU \rightarrow$  thread

دھم دلست

clippings

## Interrupts & Signals

## Interrupts, Signals تاویرت سیگنال

### Interrupts vs Signals

① رویدادهای وقوع صورت  
خارجی ترطیب افزایشی بجز  
CPU ایجاد نمودند.  
I/O device / timer /  
other cpus

cpu به Signal ① رویدادهای وقوع صورت  
یا حوزه زمان افزایشی نمودند  
آنکه در حال اجرا صورت  
ایجاد نمودند.

② صورت آسنکرون ظاهر  
نمودند.

فروشنده صورت سندرن ②  
آنکه آسنکرون ظاهر نمودند.

HW پلٹ فرم نفع ③ OS پلٹ فرم نفع ③  
(Physical Platform) دستی دستگیری  
مشخص کردند.

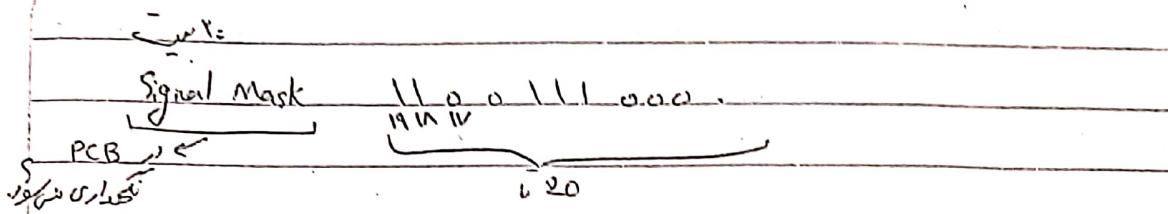
## Interrupt, Signal تاویرت سیگنال

1) Unique ID

2) قابل enable & disable تواند در سیستم

CPU ۴ تر

- برای فرستن سیگنال



الـ Signal Mask لـ وارد نسیم کرن ایسا نوع ۱۷ داره نجف

کنترلر حین Signal ignore عذرخواه است و همچو

درینتین مررت ایسا فراخوانی نیست

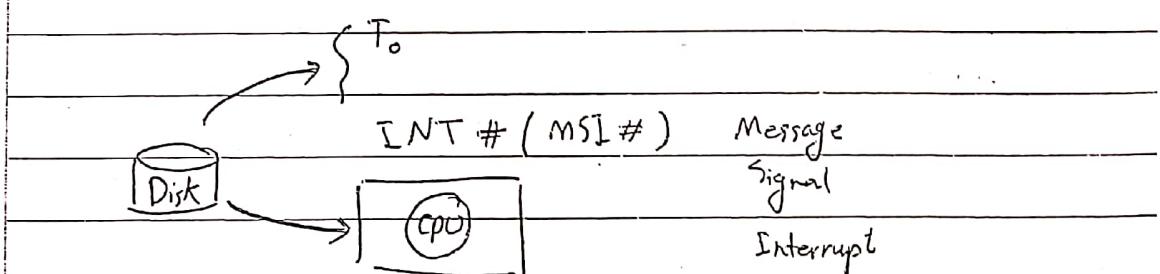
هم بـ Interrupt Mask

دستگاه رجور دارد این ایسا دستگاه اجرای میکند

کنترلر

## Interrupt Handling

ای رحال احرا ماند صورتی thread

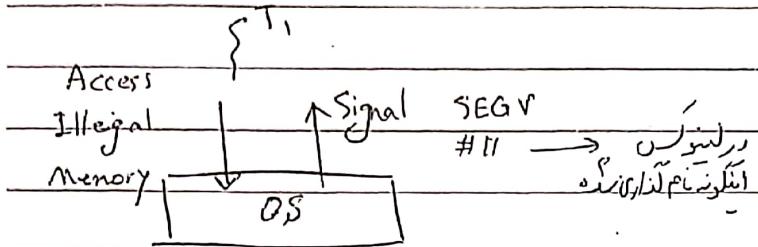


و خوددار (Interrupt Handler Table) IHT

INT1	handler 1
INT2	handler 2
INT3	handler 3
INTN	handler N
	INT i احرا بـ Handler

لـ Signal mask

## Signal Handling



در خواست دسترسی غیر مجاز به یعنی از حافظه باشد که این

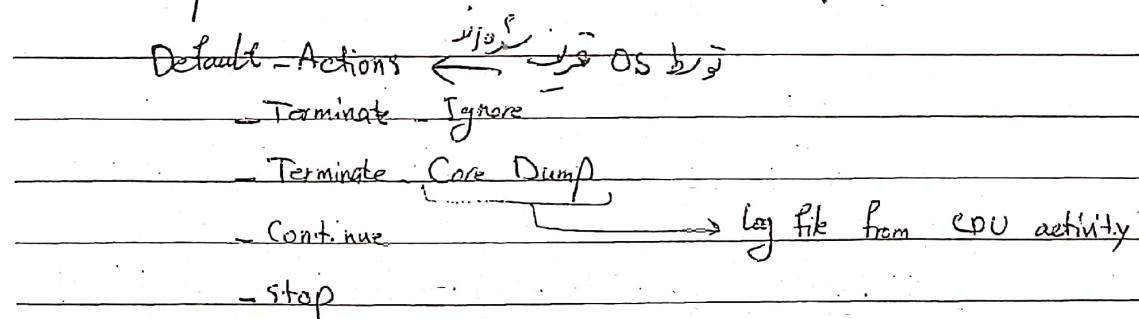
Signal ترکیبی است OS و بحاط دهن دسترسی مجاز

سیگنال نام مجاز نیست

## Signal Handling Table P.

Signal 1	handler 1 - start-add
Signal 2	handler 2 - start-add
Signal 3	
:	:
Signal N	handler N - start-add

## Handler / Actions



## Synchronous Signals

Signal Segment  
SIG SEGV

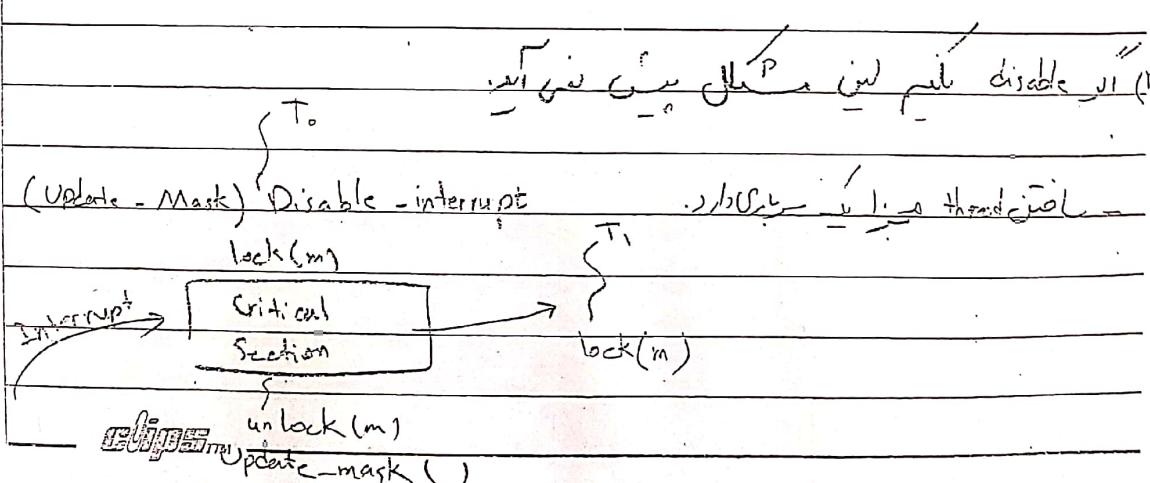
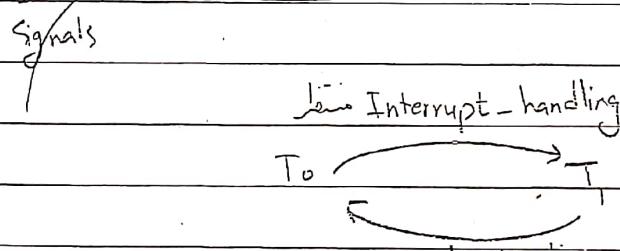
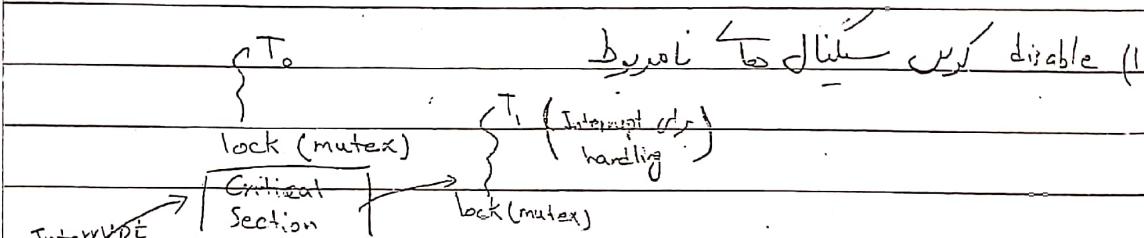
- SIG FPE هذا ينادي بـ SIGFPE
- SIG KILL (kill, id) يقتل thread

## Asynchronous Signals

SIG KILL (kill)

- SIG ALARM  $\rightarrow$  Timer expiring

Q : Why signal / interrupt disabling is useful ?



Subject ..... Date .....

لے کر اسے سامنے کریں  
lock ہجھ اور نہ اسے Interrupt handle داخل کریں

دست مارا ہو لیں

کبکا Interrupt handling فیکھے core یا Multi-core

اگر ignore کر دیں Int-hand. ہماری بدلی کو core کا

انی ہو تو یہ core

Types of Signals

1) one-shot-Signal

Signal 1 → handler 1  
Signal 2 → handler 2  
Signal 3 → handler 3

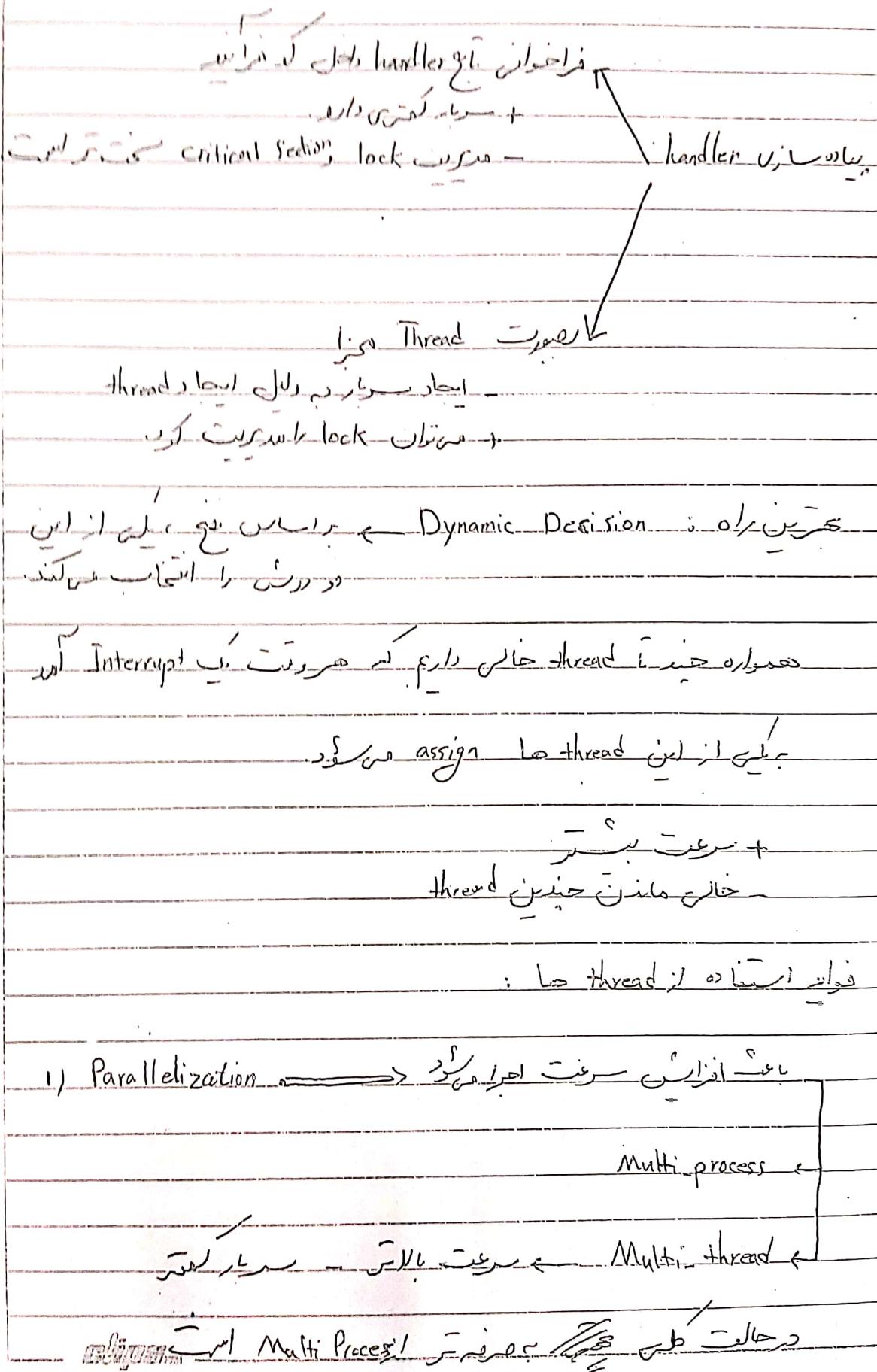
2) Real-time-Signal

Signal 1 → handler 1

Signal 1 → handler i

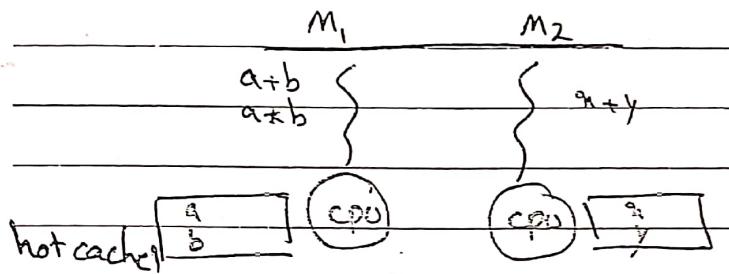
Signal 1 → handler 1

ANSWER



Subject ..... Date .....

2) Specialization  $\Rightarrow$  hot cache!



هر یاره هر راست میلیاره روی a, b احتماً رود به سراغ cache

فهرودک هدایت ریان و خواره است.

3) طاریه استعداد از حافظه ناچار

٤) Single Thread - در حال دسترسی به هر دو دستگاه I/O Device آخر هر مرد هم های با پسند دارد.

## Hide I/O Latency

## Performance Analysis :

1

معلمات سريري

## WorkLoad وظائف

لے ورکلے هائی سری نامہ در جاں احرا را تقویت من لئند

## Workload Types

CPU intensive

I/O

Communication intensive

Memory intensive

~~البيانات المترتبة على الأجهزة ( Measurements )~~

Metric ( مترقب )  $\rightarrow$  ~~بيانات المترقب~~

throughput ( تدفق )

( تقدار حجم البيانات في الثانية )

Service time ( Execution time + Waiting time )

$\downarrow$  SLA Violation Rate  
( Service level Agreement )

~~نسبة انتهاك اتفاقية مستوى الخدمة~~

SLA Violation Rate ( هي نسبة انتهاك اتفاقية مستوى الخدمة )

Real time ( الفعلية )

real time usage

purpose

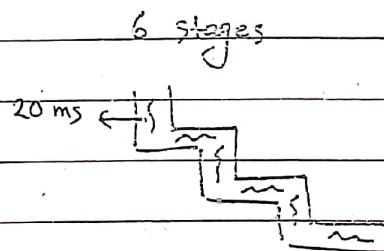
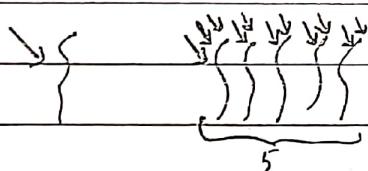
Subject ..... Date .....

پردازش سازهای را در آن نسبت به میانه: test bed

نسبت بینهای کرده است یا سیستم واحدی باشد

Boss / workers

Pipeline



if 11 orders

Exec. time per order = 120 ms

Total Completion Time

5 order  $\rightarrow$  120 ms

5 order  $\rightarrow$  120 ms +

1 order  $\rightarrow$  120 ms

360 ms (11 orders completed.)

In Pipeline: 1st order completed  $\rightarrow$  120 ms

2nd " "  $\rightarrow$  20 ms

11th " "  $\rightarrow$  20 ms

320 ms

(11 order completed) 320 ms

if 15 orders :

In Boss/Workers :

5 orders  $\rightarrow$  120 ms

5 orders  $\rightarrow$  120 ms

5 orders  $\rightarrow$  120 ms

360 ms (15 orders completed.)

In Pipeline :

1st order completed  $\rightarrow$  120 ms

2nd " "  $\rightarrow$  20 ms

15th " "  $\rightarrow$  20 ms

400 ms

(15 orders completed.)

Exec. time per order  $\leftarrow$  orders  $\rightarrow$

Total comp. Time =  $T \lceil \frac{x}{w} \rceil$

in Boss / workers

$\downarrow$  workers  $\rightarrow$

Total comp. Time =  $T + T_s (x-1)$

in pipeline

$\downarrow$   
Stage - 1 (1st)

digipage

$$T \left\lceil \frac{x}{w} \right\rceil \leq T + T_s(x-1)$$

Boss/worker لام : ما حل این معادله خواهیم یافت

برای pipeline از

Boss/worker  $\rightarrow$  order  $i \in [1, n]$  میانگین وقت

P.S.

$$\text{Avg Completion time} = \frac{5 \times 120 + 5 \times 240 + 360}{11} = 196 \text{ ms}$$

برای pipeline  $\rightarrow$  میانگین وقت طول مکالمه

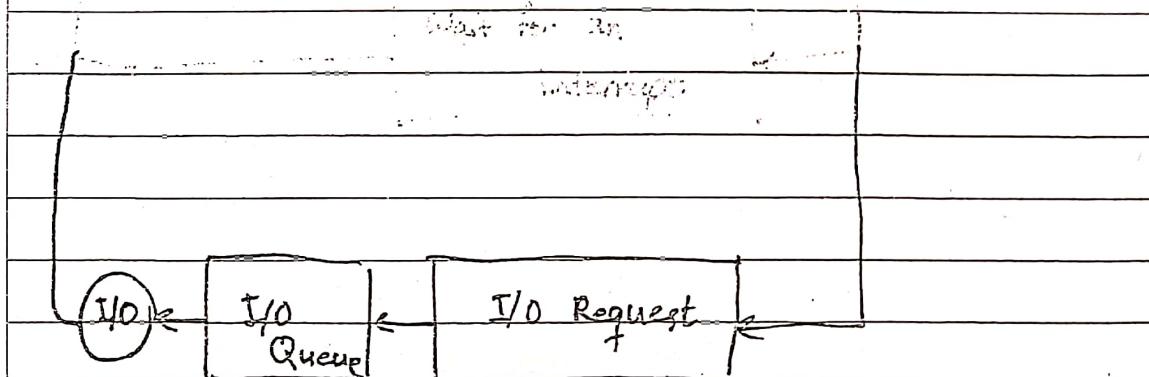
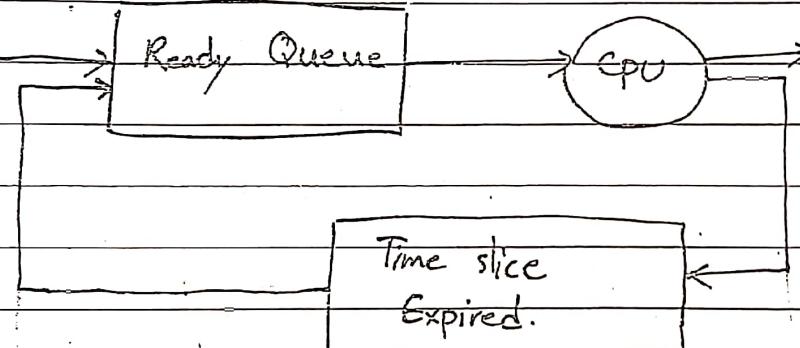
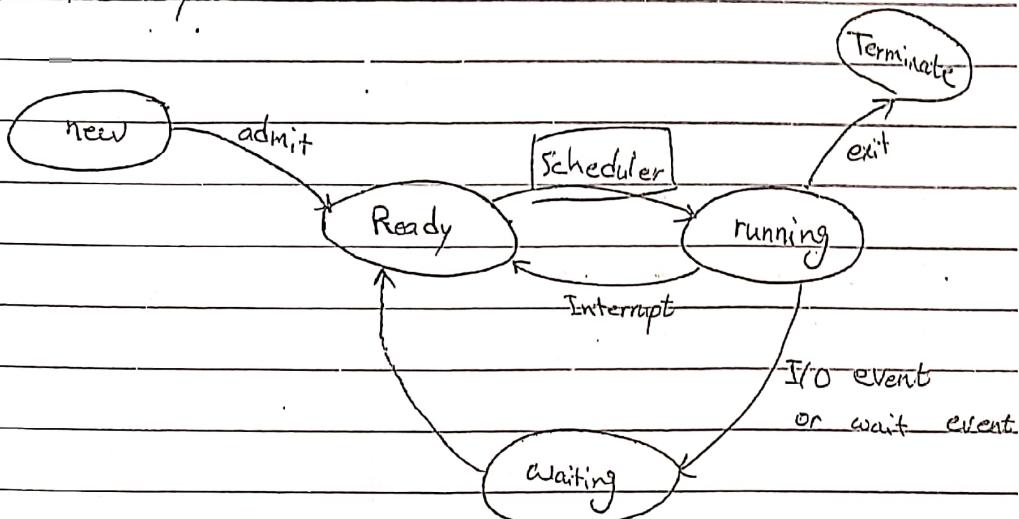
P.S.

$$\text{Avg Completion time} = \frac{120 + 140 + 160 + \dots + 320}{11} = 220 \text{ ms}$$

حتماً هر کاری انجام می‌دهید هم می‌باید هم می‌باید

edges

## Process Life cycle



Until CPU is free it is not available

*ANSWER*

Subject ..... Date .....

Thread  $\rightarrow$  Light Weight Process

Process  $\rightarrow$  Task / job

مهمة حالي

Scheduling :

1) Single Core (CPU)

2) Multi Core (Multi CPU)

: Non Preemptive

عوائق تعيق القدرة على توزيع المهام بين المكونات

أولاً، اختيار المكون

: Preemptive

الثانية، اختيار المكون الذي يحصل على المكون

第三次选择任务

د. احتساب دورة حركة المكون

(First-In-First-Out)

= FCFS

(First-Come-First-Service)

FCFS

بررسی مقایسه الگوریتم های زمانبندی بر اساس معیارهای این درست

Metrics :

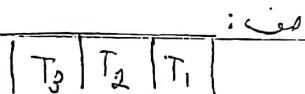
1) throughput مقدار طبقه انجام شده در واحد زمان (لذت بخش)

2) Avg. Completion Time : میانگین زمان تمام شدن کار

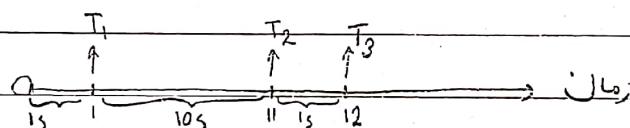
3) Avg. Waiting Time : انتظار کار تا اجرای آن

مثال :

$$T_1 = 1 \text{ s} \quad T_2 = 10 \text{ s} \quad T_3 = 1 \text{ s}$$



: FIFO = سریع



$$\frac{3}{3} = \frac{12}{12} \rightarrow q = \frac{3}{12} = 0.25 \rightarrow \text{throughput}$$

$$\text{Avg. Completion Time} = \frac{1 + 11 + 12}{3} = 8 \text{ s}$$

$$\text{Avg. Waiting Time} = \frac{0 + 1 + 11}{3} = 4 \text{ s}$$

مشکل

$$\text{throughput} = \frac{\text{قیاد کارها}}{\text{زمان پایان آخرین کار}}$$

$$\text{Avg. Completion Time} = \frac{C_1 + C_2 + C_3}{\text{قیاد کارها}} = \frac{1}{n} \sum_{j=1}^n C_j$$

$$C_n = \sum_{i=1}^n T_i = \frac{1}{n} \sum_{j=1}^n \sum_{i=1}^j T_i$$

کمترین زمان ملزم حفوب از ایسا کار که بیافسر کارها از آن دارد

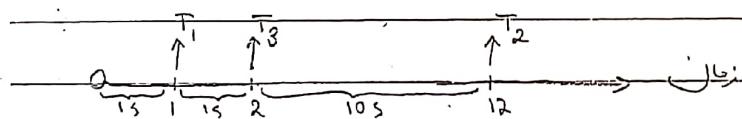
$$S / 60$$

$$\text{Avg. Waiting Time} = \frac{0 + T_1 + T_2}{\text{قیاد کارها}} = \frac{1}{n} \sum_{j=1}^n W_j$$

$$W = \frac{n-1}{n} \sum_{i=0}^{n-1} T_i = \frac{1}{n} \sum_{j=1}^n \sum_{i=0}^{j-1} T_i$$

( Shortest Job First )  $\therefore$  (SJF) انتخاب کریں

ترتیب زمان اجرا  $T_1 \rightarrow T_3 \rightarrow T_2$



$$\text{throughput} = \frac{3}{12} = 0.25$$

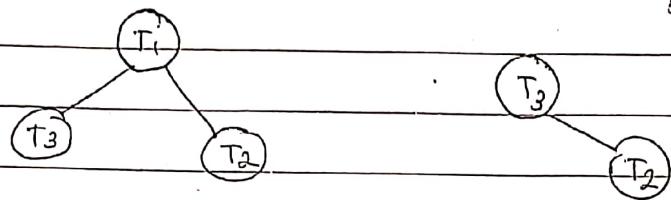
$$\text{Avg. Completion Time} = \frac{0 + 1 + 2}{3} = \frac{15}{3} = 5 \text{ s}$$

$$\text{Avg. Waiting Time} = \frac{0 + 1 + 2}{3} = 1 \text{ s}$$

ABC

: SJF سارہ سارہ

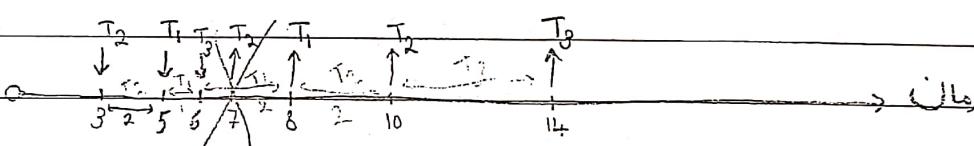
Min Heap اینٹیل



in Non-Preemptive Scheduling کے لئے اس کو

: Fibonacci Preemptive Scheduling حال

Task	Arrival Time	Exe Time	Priority
T <sub>1</sub>	5	3	P <sub>1</sub>
T <sub>2</sub>	3	4 - 2 = 2	P <sub>2</sub>
T <sub>3</sub>	6	4	P <sub>3</sub>



↓  
جنر اولویت ت بالا کرنا  
کم سروش کرنا

$$\text{throughput} = \frac{3}{11}$$

$$\text{Avg. Completion Time} = \frac{\frac{7}{(10-3)} + \frac{3}{(8-5)} + \frac{8}{(14-6)}}{3} = \frac{18}{3} = 6.5$$

$$\text{Avg. Waiting Time} = \frac{0}{3} + \frac{3}{3} + \frac{4}{3} = \frac{7}{3}$$

۱) سریع کر فرایند کدام است؟

I/O Request (۱)

Interrupt (۲)

Wait Event (۳)

(۴) دسترسی غیر مجاز خواهد بود  
حالاً دسترسی به حافظه رخواهد داشت  
در این (درگاه)

۵) اگر خارجی من در thread thread ۱ فرایند هستند اطمینان حاصل کن

thread control block

TCB<sub>1</sub>

TCB<sub>2</sub>

لئے چه طریق بلنم؟

{  
CPU

وقتی حال نیز فرایند نباشد:

۱) Update TCB<sub>1</sub>

۲) load TCB<sub>2</sub>

درست رکورد حال نو فرایند میزباند حل مسأله سه هم دارد:

۳) Map VM<sub>2</sub> to the physical Memory

: process  $\rightarrow$  job multi-threading اسے ایک دوسرے کا سلسلہ کرنے کا map بھائی تھا۔

سائز کو map بھائی تھا۔

لے کر hit rate کا سامنہ اٹھائیں۔ اسے بالآخر میرا۔ Hit Rate (%)

پیک ریت دا جراہا رینڈا جانے والے اسے۔

: ( Round Robin ) RR

time-slice دیتے ہیں اسے FIFO ہیں جو کوئی

کوئی نہ کوئی thread time slice میں از اسے کرے۔

Time-sharing (چالے چالے) ہے اسے زمانہ بھار کرے۔

Non-preemptive (باقی نہ کرے) اور Preemptive (باقی کرے)

Process	Arrival Time	Exec. Time	FIFO	SJF	RR
P <sub>1</sub>	0	5			
P <sub>2</sub>	1	5			
P <sub>3</sub>	5	3			
P <sub>4</sub>	6	2			
FIFO	$\downarrow P_1$	$\downarrow P_2$	$\downarrow P_1$	$\downarrow P_1$	$\downarrow P_2$
	0	1	5	5	6
			6	8	10
			10	10	13
			13	13	15
SJF	$\downarrow P_1$	$\downarrow P_2$	$\downarrow P_3$	$\downarrow P_4$	$\downarrow P_2$
	0	1	5	6	8
			6	8	10
			10	10	15
RR	$\downarrow P_1$	$\downarrow P_2$	$\downarrow P_3$	$\downarrow P_4$	$\downarrow P_1$
	0	1	5	6	6
			6	8	8
			8	10	10
			10	10	11
			11	11	R
			R	13	13
			13	13	15
			15	15	17
			17	17	TB
			TB	TB	
					TB

(٢) دلایل و نتایج حملات ماهیتی شده

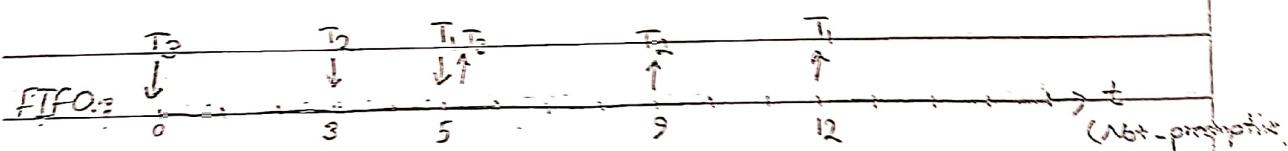
اصناف اولیه حریفان را معرفی کردند ~~بیش از ٥۰~~ Scheduler

اصناف اولیه حریفان را معرفی کردند ~~بیش از ٥۰~~ Scheduler

حدود ٤٨٠ تغییر می‌کند این باید

نمایل:

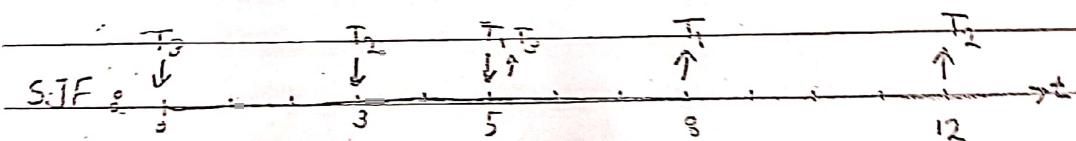
Task	Arrival	Priority	Exec. Time
T <sub>1</sub>	5	P <sub>1</sub>	3
T <sub>2</sub>	3	P <sub>2</sub>	4
T <sub>3</sub>	0	P <sub>3</sub>	5
$P_1 > P_2 > P_3$			



$$\text{Avg. Completion Time} = \frac{(5-0)+(9-3)+(12-5)}{3} = 7$$

$$\text{Avg. Waiting Time} = \frac{0+(5-3)+(9-5)}{3} = 2$$

$$\text{Throughput} = \frac{3}{12} = 0.25$$

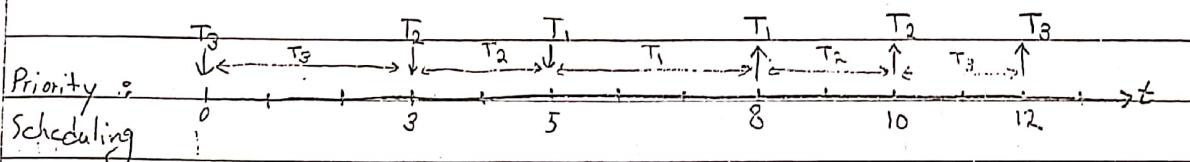


(Not-preemptive)

$$\text{Avg. Completion Time} = \frac{(5-0) + (12-3) + (8-5)}{3} = \frac{17}{3}$$

$$\text{Avg. Waiting Time} = \frac{0 + 0 + (8-3)}{3} = \frac{5}{3}$$

$$\text{Throughput} = 0.25$$



$$\text{Avg. Completion Time} = \frac{(12-0) + (10-3) + (8-5)}{3} = \frac{22}{3}$$

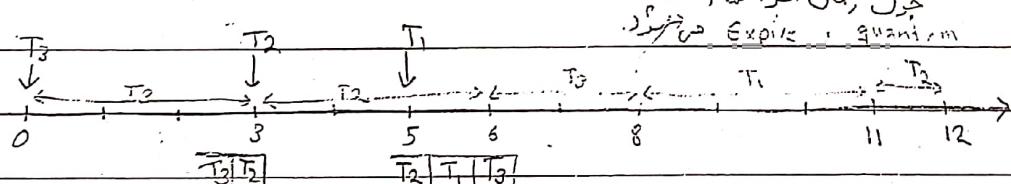
$$\text{Avg. Waiting Time} = \frac{(12-0-5) + (10-3-4) + (8-5-3)}{3} = \frac{10}{3}$$

$$\text{Throughput} = \frac{3}{12} = 0.25$$

\* اگر اولیت های میانی که ایجاد شد بود باشد که اینجا صدق نمی کند

فراز طاری

RR : if quantum = 3 s



و میاده سازی نمی کند time-sharing

کارهای بزرگ را که نیاز به زمان زیادی دارند

لذا task یک task ایجاد کردن که زمان اسکرین ایجاد کردن

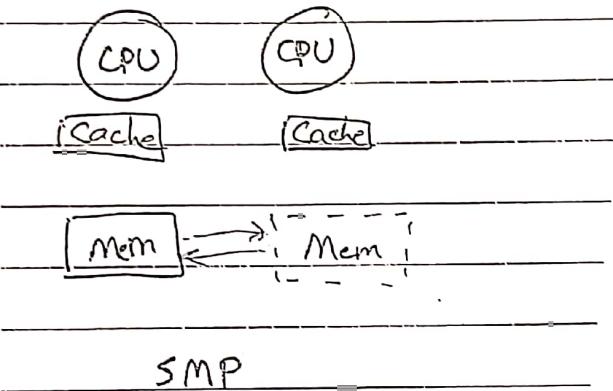
ایجاد task یک task ایجاد کردن که زمان اسکرین ایجاد کردن

Subject ..... Date .....

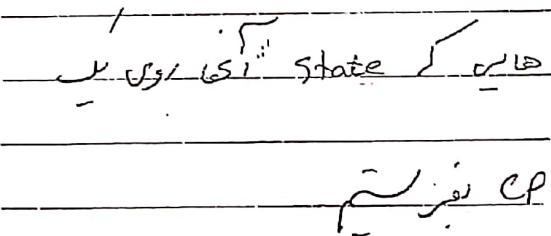
Idle quantum

## Scheduling on Multi-Core

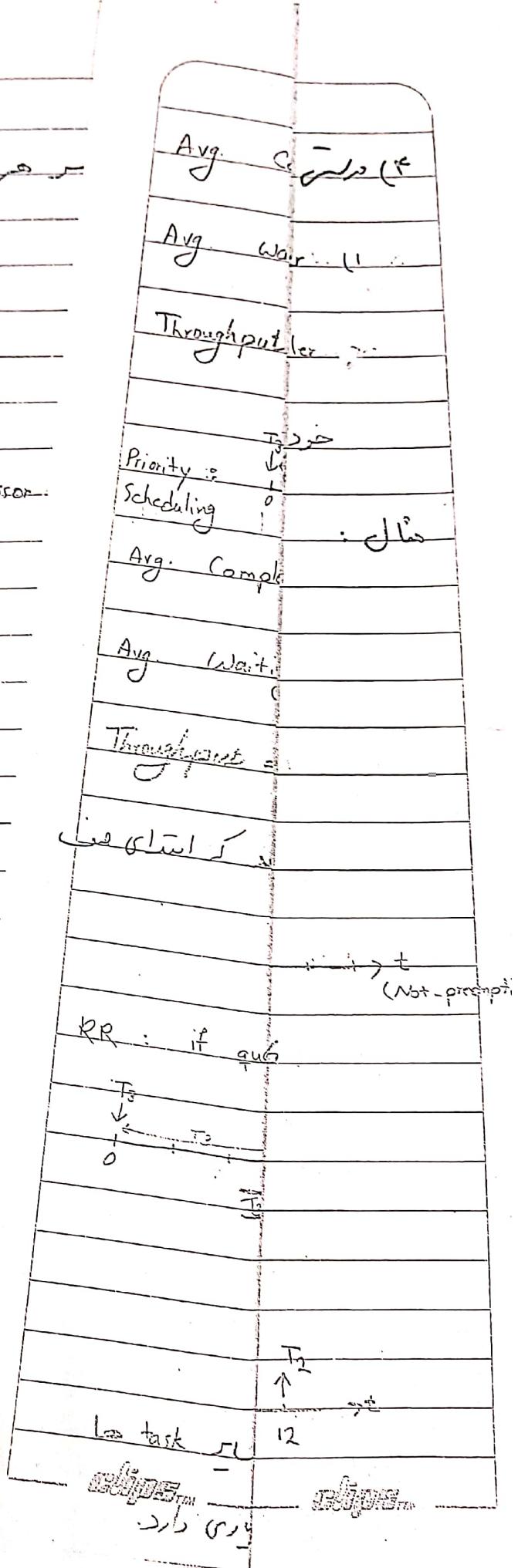
Shared-Memory Multi-Processor



SMP



initial Core

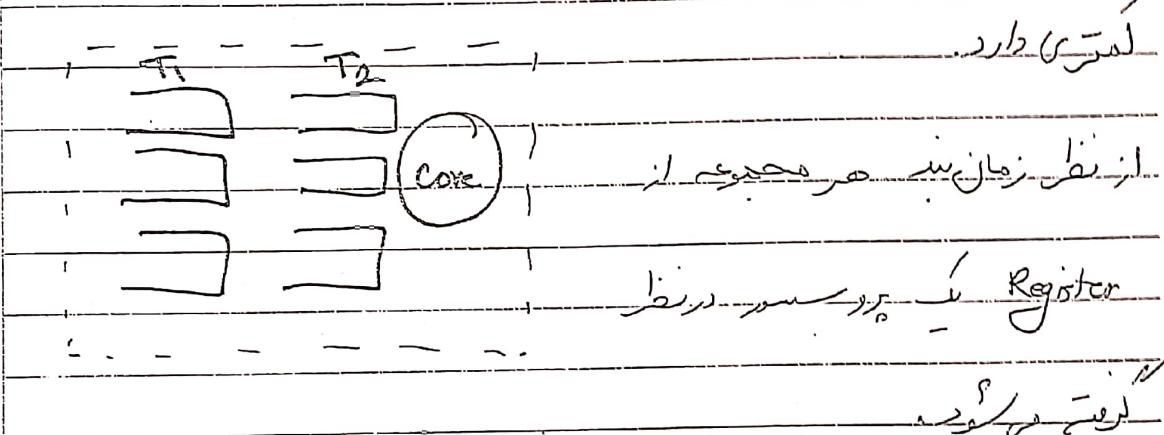


Hyperthreading

Hyperthreading  $\rightarrow$  Register مجموعه کی Core پر

$\rightarrow$  Register مجموعه کی مجموعہ کی Core پر

نظام کیمی از طریق خود این سوچیں



آخر دسترسی حاصل ! ہم برسن ہو رہد

مثال برے مطابق

Chip Multi-thread Processors need a new OS scheduler,

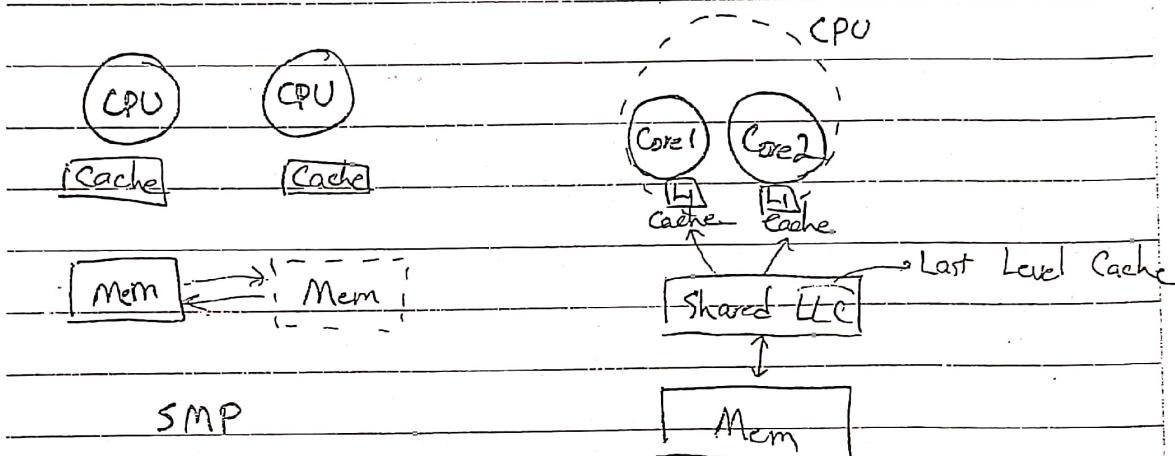
Fedorova et al.

نحوه انتشار quantum  $\rightarrow$  scheduling

X time X

## Scheduling on Multi-Core

### Shared Memory Multi-Processor (SMP)



Multi-Core

نحوه ایجاد task  $\rightarrow$  Numa Scheduling

ایجاد task  $\rightarrow$  CPU  $\rightarrow$  نزدیکی memory

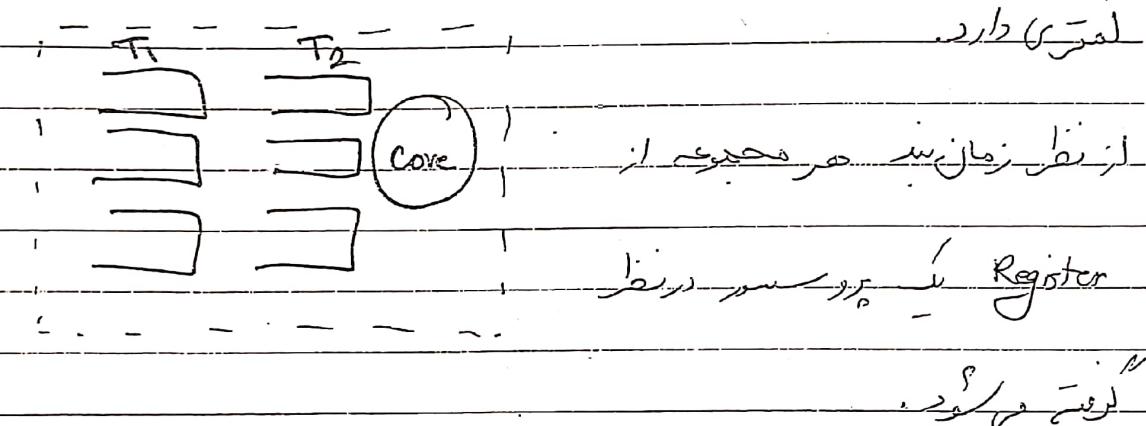
ایجاد task  $\rightarrow$  CPU  $\rightarrow$  نزدیکی thread

CS CamScanner

Hyperthreading  $\rightarrow$  وحدة دار Register مخصوصة لـ Core

$\rightarrow$  Register مخصوصة لـ بحاجة Core

نظام تغير المهام كودي اخر : ازمه



آخر دسترسی حاصل نمایند

فعال بودن

Chip Multi-thread Processors need a new OS scheduler,

Fedorova et al.

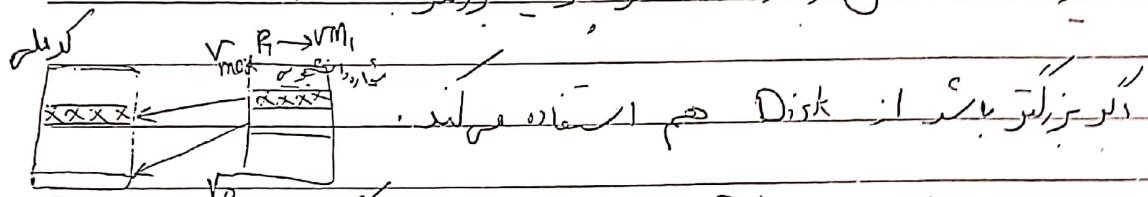
clips...

## Memory Management

الذاكرة الافتراضية : Virtual Memory

هزاره ندارد و تنها از سهین فرانسه تعیین می‌گردند.

RAM: میان ارزش کوچکتر یا بزرگتر باشد VM میز



دسترسی حافظه از دو طبقه در ترانزیستورات صورت نماید.

pages

## Segments

دراجهها، معلمینش هم لست و هر یار فرخانی در آنها  $\pm 500$  Load حسنه

• Segment → page تابع

1 KB "the - and - up page will

1 KB

## Segment ٦٦

, 2 KB ~~for~~<sup>1</sup>, 1 KB ~~for~~<sup>1</sup> ~~is~~<sup>is</sup> the

: لیست حافظه

1) Allocate لیست حافظه میز کن

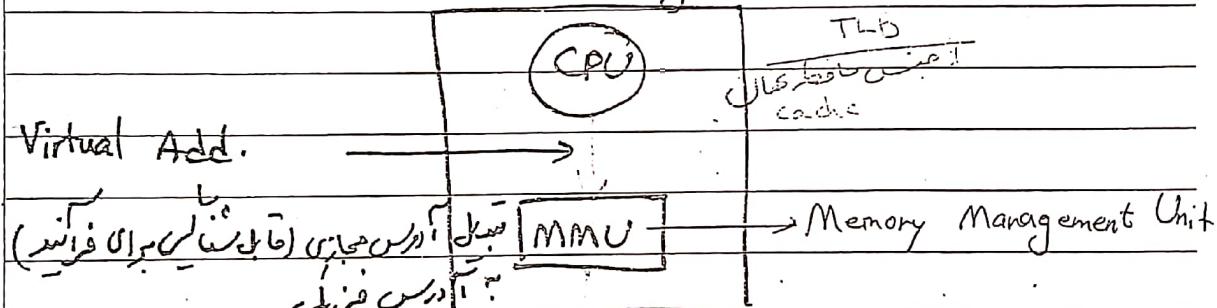
2) Replace لیست حافظه از بین

3) Arbitration لیست حافظه از بین → Address Mapping (Translation)

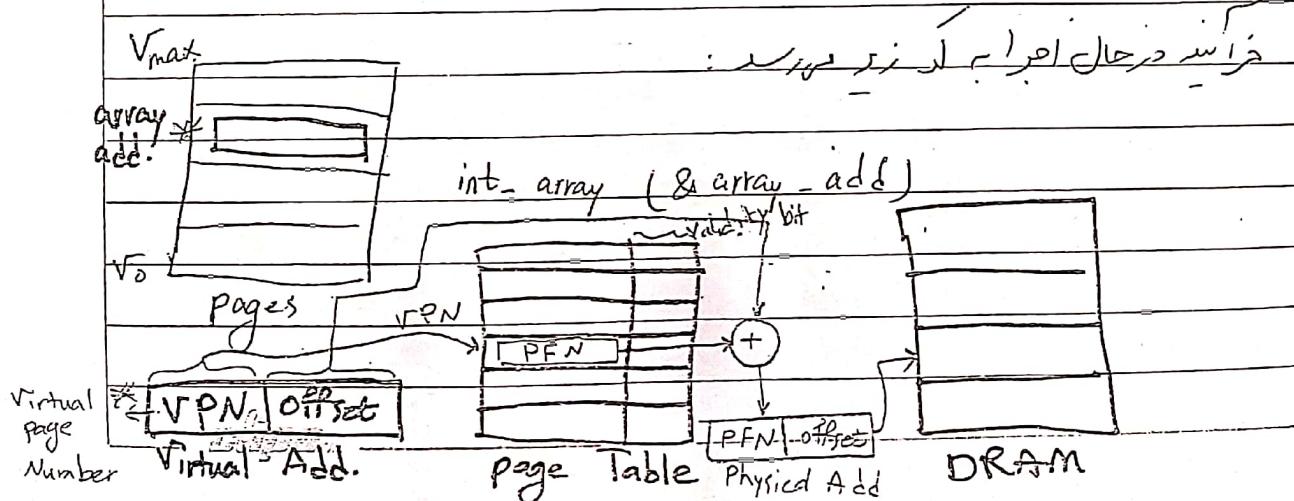
→ Validation (امان نمودن)

HW Support :

CPU Package



: فقط در حالت اصراری, Add. Translator [سازنده ترجمه]



حافظه داشتے ہیں : 32KB پر

$$32 \times 2^{10} \times 8 = \log_2 2^{18} = 18 \quad 18 \text{ بیت بر ٹکسٹ صفحہ}$$

الرہر page 32 جمیع 1KB، page داشت

$$\log_2 32 = 5 \rightarrow \text{VPN}$$

$$18 - 5 = 13 \rightarrow \text{offset}$$

رسانی کے امراض :

: MMU ، کارڈ

..... ←  
اگر مخفی - فریم (لیسٹ) -  
→ Reports faults .

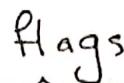
CR3 ← x86 → : to Register 16 bit  
pointer

mapping (page table) ، size ، address ... و (اپنے کو ہے )

TLB ( [Cache] - Translation Lookaside Buffer ) : (

. . . . . Validate کرنے اسے .

Validate translation of VA  $\rightarrow$  PA



Subject ..... Date .....

PFN	Validity bit	dirty bit	R	W	X
-----	--------------	-----------	---	---	---

## page Table Entry

حافظ اخرين تفسير رادر حافظ زخيره مولندا

Access bit

مخطط خواص بینیم page Table ایجاد کنیم

با فرض اینکه  $\text{page size} = 1 \text{ KB}$

page1

page2

offsets (10 بت)

page number (1 بت)

64-bit      32-bit

page Table Size?

## Page Table Entry

8 Bytes

4 Bytes

## نکات اور صورتی مختلف

The diagram illustrates a memory location structure. It consists of three fields: 'PN' (Physical Address Number), 'offset', and 'base'. The 'offset' field is explicitly labeled as being 10 bits wide. Above the structure, a callout bubble indicates that the sum of the 'base' and 'offset' fields totals 1 KB (1024 bytes). The total width of the address is shown as 54 bits.

Subject ..... Date .....

مکتبہ میرزا

## میراث حافظہ:

## Memory

## Allocation :

کے صفحے و صفحائی حامی اور نزدیک

۲- (دوره ایم) کنوارتے عنوان (:

kernel level 61

User level

نے App میں User Level پر اجازت دیکھ لی

رسی داده های مخصوص آپلود کردن صور از AP ها محذف است.

malloc()

آزاد کردن معتبر حافظه Free( )

11

1

malec (4)

malloc(3)

malloc(1)

二

free (1)

free (2)

malloc (3)

حاجة شخصی داشته در ذرا سخته زد

مطالعه نیست

(External) Memory Fragmentation

(کشیده)

حاله اعلان میگیرد حافظه

برای شخصی مطالعه، بینین اکلیوی رخواست ها را نمایم

با شرط اینکه غیر مطالعه شخصی داشم و قادر در خواسته های

شخصی تا حد امکان نزدیک شویم

Buddy

Linux kernel Allocator

Slab

64M

: Buddy, 64M

32

: f/b Malloc(8M) = درخواست

32

8M میگیرد

② malloc(8M)

③ malloc(4M)

④ free(8M)

⑤ free(8M)

جنب دفعه ایجاد شوند

سیکل یک 16 متر

نام دلار  $2^x$  allocator ، Buddy allocator

زیرا الرخصایی لـ مدیر حافظه یعنی ریم ترانزیشن ۲ برد ، بر این اساس

کوچکترین بلکه کوچکترین ۲ است بر این حافظه اطمینان می دهد .

برای مثال : `malloc(7m)` (است) باشد

۸m حافظه اطمینان می دهد

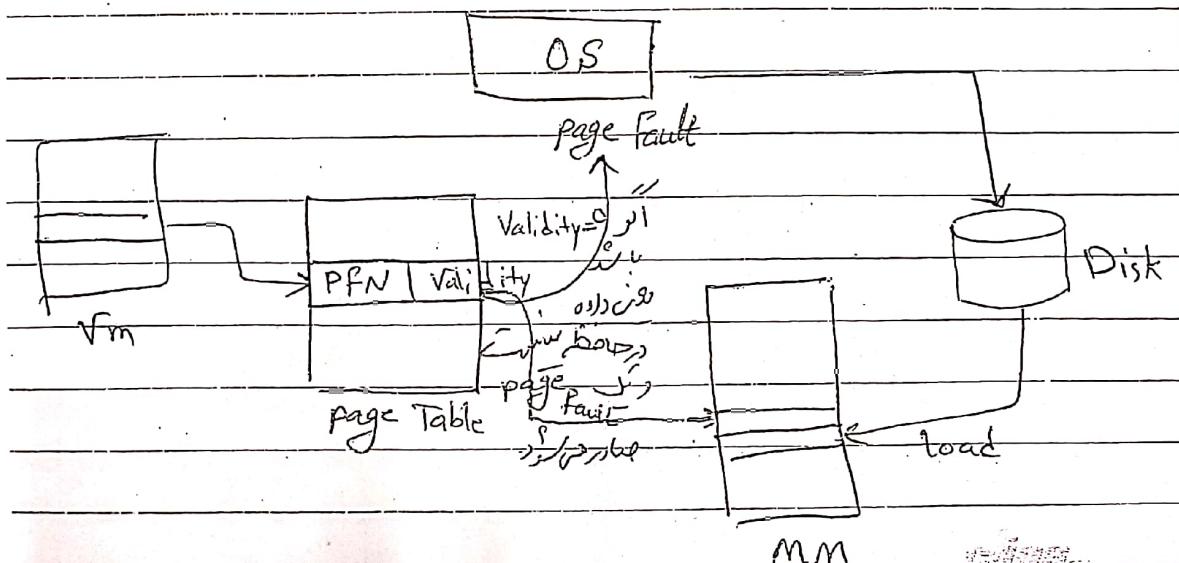
کلّ بلکه هر کسی نیز تواند حافظه ۲ است

قطع آن می خواهد بهم میگویند که اندازه آنها باشند

کلّ است

Disk سیستم (paging) در اطلاعات در حافظه های خود میباشد

page Demand



آخر page ها، حافظه pin

Disk

آخر حافظه pin، لذم تبادل

pin load حافظه Disk

1) LRU

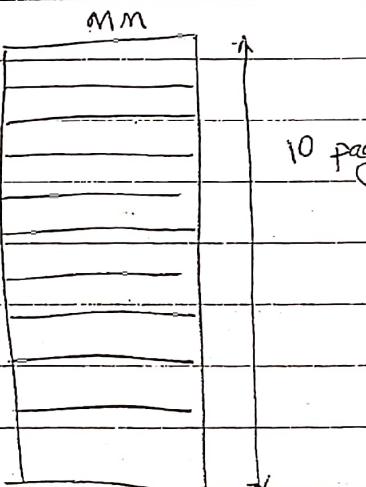
١٨ صفحه

for i=1 to 11 do

A[i] = 3 ;

for i=1 to 11 do

A[i] = 5 ;



10 pages

Second chance ←

## سناردون سائز (SMP size)

( mutex , C.V ) سائز سناردون سائز

Semaphore

Atomic Instruction ← test-8-set -

سناردون سائز : اسٹیکر برائی آن لارڈن دیو ترک سائز

در حال اچڑا درست .

روئی برائی سناردون سائز :

spinlock (m) unlock (m)

دو ٹکیں کوئی فرائید نہیں ملے m را جد

فرائید کو آئی آن لارڈن ایک lock (زندگی) میں لئے دیں سب سب

حکمت ملک حملہ میں لئے

نیوں ایک ایک دی سیوں ایک ایک lock (m) کے ساتھ ایک ایک

حکمت فریم نویسی ها کی برائی نویسی سیاہ سائز سناردون

حکمت ایسا ہے کاٹے ایک ایک ایک

روز خرد کو مکمل بان اسار یعنی CPU Spinning

(CPU Spinning چیزی است) (HCU Level) کیا افراز است.

Binary Semaphore یعنی اکثر افراد سازی کرده

(OS level)

Binary Semaphore ایسا یعنی (Integer Value) یعنی

(Set max, min values) Set boundary Values

init

try (wait)

Critical Section

post (unit)

binary Semaphore یعنی اگر کوئی بلبری max کی

نباشد سپس کوئی

#include <Semaphore.h>

Sem\_t : Sem;

Sem\_init (Sem\_t \* m, int pshared, int count)

int + read

مکانیزم دیگر

از این سایر انتهاه کیا

از سایر انتهاه

Sem\_wait (Sem\_t \* m)

روز کام سایر نظر بعل

CS CamScanner

Subject ..... Date .....

Sem-post (Sem\_t \* m) يُمسك بـ unlock لـ exit (جهاز)

Readers / writer (جهاز)

spinlock (جهاز)

# include <linux/spinlock.h>

read\_lock(m);

read\_unlock(m);

// critical Section

write\_lock(m);

write\_unlock(m);

// critical Section

write\_unlock(m);

نحوين (احصار)

Semaphore (جهاز) Reader / Writer (جهاز)

مهم (جهاز) spinlock (جهاز)

to, ٤٤

(جهاز) (جهاز) (جهاز)

CS

## Monitor

برای unlock, lock می‌توان از مراکز مخصوصی مانند

استفاده کنیم و قبلاً بعنوان library نوشته شده است

اگر مانند را برای برخاستن فرآم داشتیم

و درین را برای نزدیک فرآوری کردیم

که مانند مذکور شده می‌باشد critical section

می‌باشد

The Performance of spinlock Alternatives for Shared Memory

in Multi-processors (Anderson)

spinlock\_init (lock) : : spinlock via

: lock\_free ;

spinlock\_lock (lock) :

spin :

if (lock == free) {

else { goto spin; }

spinlock\_unlock (lock) :

lock = free;

اگر کسی از مراکز مذکور شده ایست همراه باشد و در

spinloop

داستنی

برای رفع مثال از Test-and-set

ک درست، اگر دیگر کارهای دیگر نداشته باشد

 $\text{if} (\text{lock} == \text{free}) \text{, } \text{lock} = \text{busy} \rightarrow \text{test-and-set(lock);}$ 

اگر test &amp; set خواهد بود اینها را thread می‌خوانیم

اگر lock را اجازه نمایند Set آنها را بخواهند

آنها می‌توانند

cache coherency with validity  $\rightarrow$  The bestcache coherency ~~with~~ invalid  $\rightarrow$  Horrible!

از زیرین کاری داشت همان مسئله نداشت

[1] Latency (رسان)

[2] Waiting time (Delay)

[3] Contention (رقابت)



spinlock\_init(lock);

spinlock\_init(lock);

lock = free;

spinlock\_lock(lock);

while (test\_and\_set(lock) == busy);

spinlock\_unlock(lock);

lock = free;

• delay

+ Latency

+ Delay

contention

GOOEY

Subject ..... Date .....

I/O Devices

I/O devices : I/O System (I/O)

File System (I/O) System (I/O)

: I/O System (I/O)

بروتوكول ها

Device handler

جهاز و نیکان سازی حسایت Application : I/O

, I/O (نیاع)

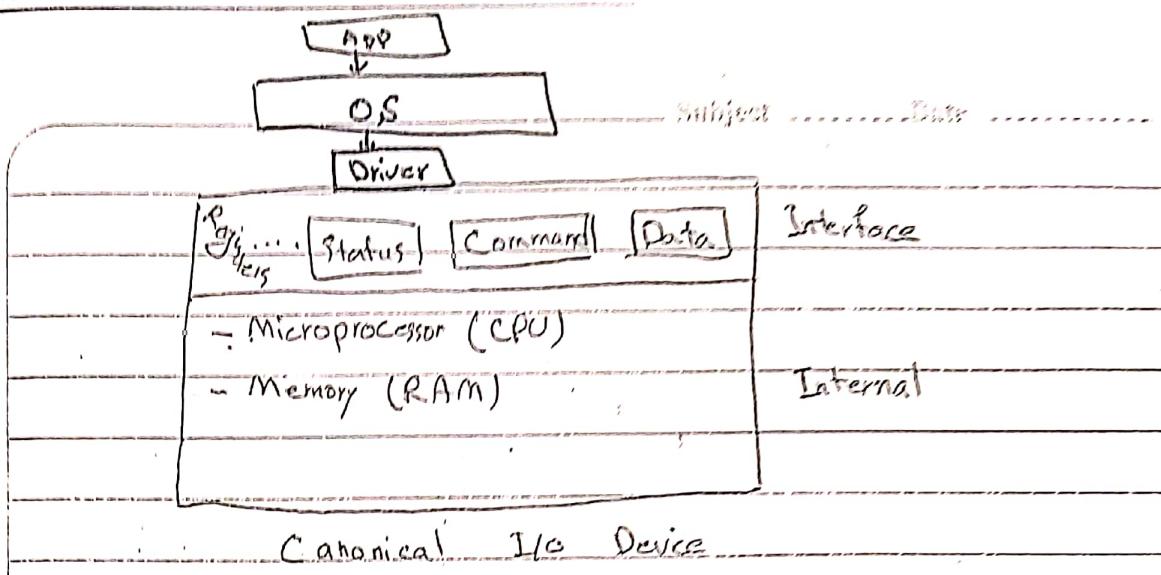
mouse , keyboard : ورود

پرنسپر , speaker , میکروفون : خروج

(NIC) برد کنترلر , HDD , CD/DVD ROM : اتصالات / ذخیره سازی  
SSD

(I/O storage و ذخیره سازی)

Clipper



اسناد داره ها :

: Block -

یک حجم از داده ها را هم خواندن کردن، استدیل می شود

کارهایی که این لیست می خواهد، نویسن دریل

: Character -

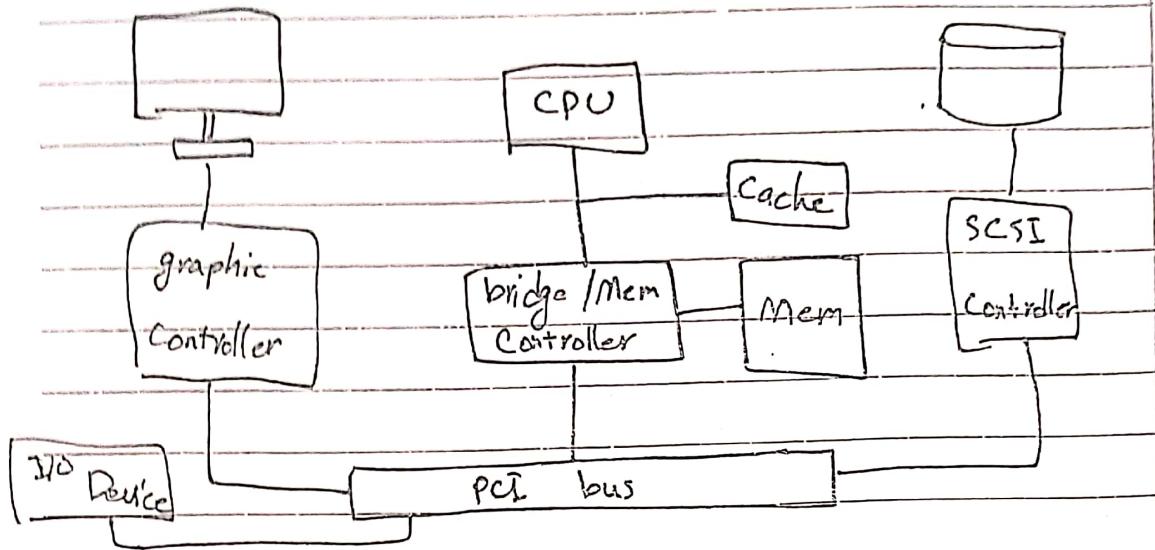
اسناد تصریح شده سری میانی

Nkt Devices

اسناد داره ها

دستگاه

## PCI (Peripheral Component Interconnect)



PCI Express  $\rightarrow$  PCI-X  $\rightarrow$  PCI

PCI-e امراض سرت

CPU ، I/O controller در پرینت کارهای حاسوبه

## II Programmable I/O (PIO)

Programmable I/O

کمپیوٹر کے امراض ایک فنار

I/O controller کو CPU کو Interface کیں

I/O controller کو CPU کو command بیان کیں

امراض کا Data Controller نہیں بلکہ اسے

Bridge

نیکون میکروسوفت پرینٹر کا CPU

فیس

NIC : 1500 packet , 8 byte register

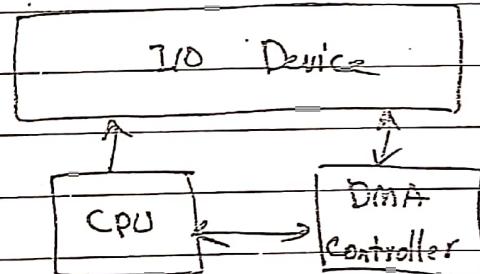
1 cycle

command from CPU → 1 cycle

$$= \lceil \frac{1500}{8} \rceil = 189 \text{ cycles}$$

$$189 + 1 = 190 \text{ cycles}$$

## E) Direct Memory Access (DMA)



وہ فیس جسے فیس

command from CPU

DMA config → Transmit instruction

وہ فیس

Subject ..... Date .....

فرق الورقة ببروكسل :

اللوريم فعطاً بـ Device اجراءاته

بروكسل < ارتباط من رويا حيد طرف

\* ناتج : الاردستورات تعيّن طلب استدعاء DMA بحسب ابرات

وله اردوستورات يابه بالذرة تعدادي زياداً منه يمكن

لسنة تعدادي cycles هامٍ كدر CPU صدره بقدر لغير از تعداد

DMA config بـ cycles هامٍ بـ cycles هامٍ

هـ این اسبرت بـ اول هـ اسـ

clipse

Subject ..... Date .....

clips

Subject ..... Date .....

## خُرَقُ الْأَدَارِسِمْ خَرَقُ دَرَالِ

اللوريم عطوري Device اجهزة

بروتوكول ← لارتباط من روایت حیر مرت

نکتہ: اگر دستورات میں اسی دلیل کے لئے DMA کا نام نہیں آئے تو اسی کے لئے اسی دلیل کے لئے DMA کا نام نہیں آئے تو اسی کے لئے

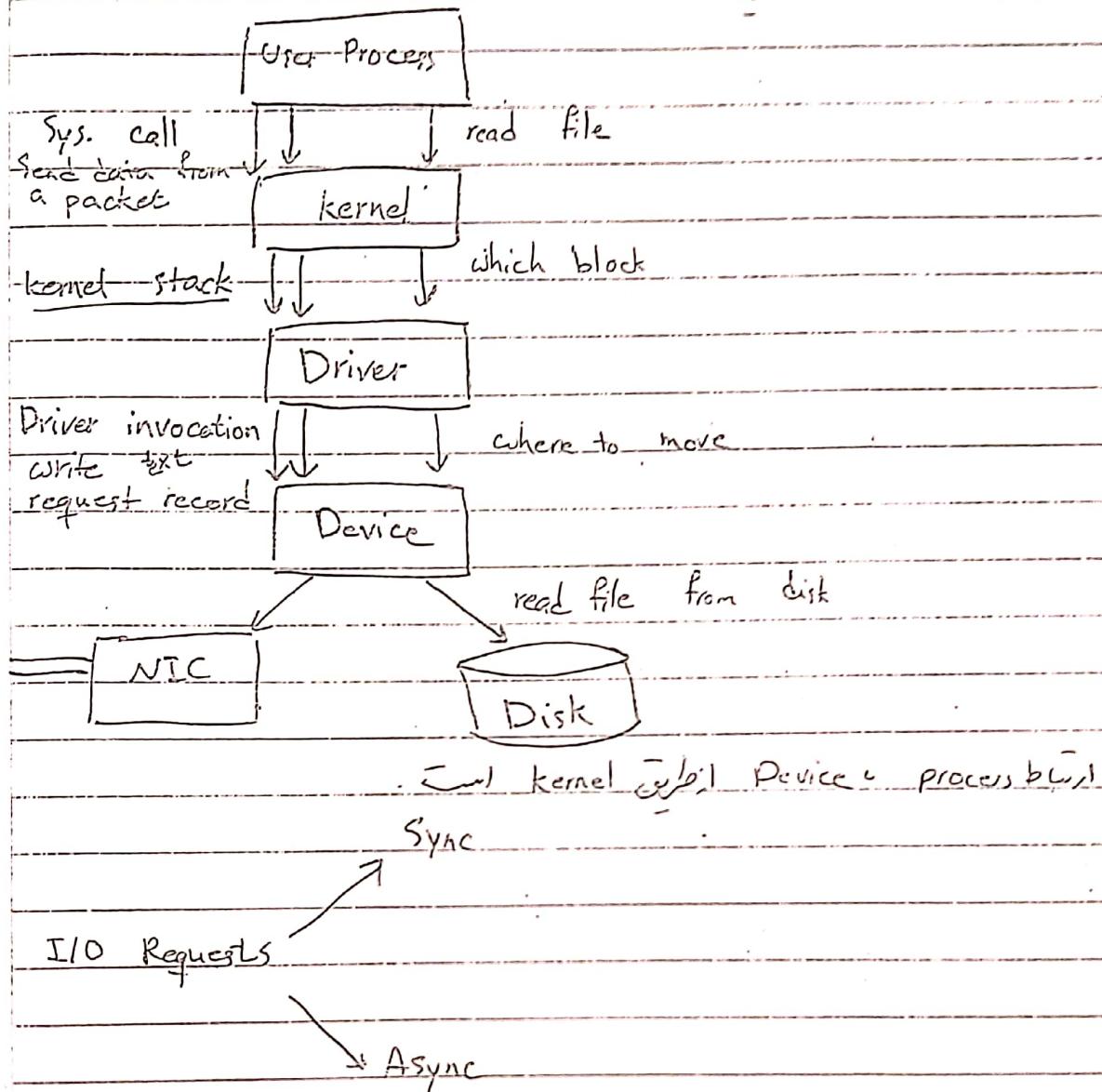
دلیل اور دستورات سارہ مانگنڈل تعداد کی ریکارڈ بائیوگرافی میکن

لیست نویلر cycle هایی که در CPU هنر دستور لکھ را زیر نگار

DMA config براي چند cycle

## در لین اسپریت روئن اول هنر لست

## 210 - خوارزمی و زیرساخت ایمنی



ماضی ارتباط من بین Driver و Device، Proc

درین روشن نیز (kernel bypassing) میگویند

که در همین هنگام همیشه آن، این است که سرتاسر User-level driver هست که در **lib** قرار دارد.

لذا در اینجا مقادیر لایه ها کمتر نموده شده اند.

clips

## Inter Process Communication (IPC) : ~~to process, you b̄iñj~~

### ① Message Passing

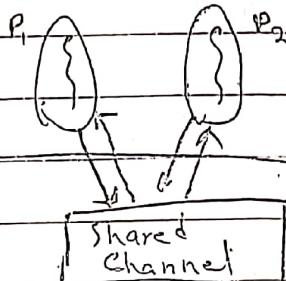
→ Sockets

Pipe

message queue

APP(User)

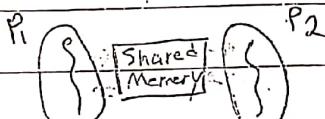
OS



### ② Shared Memory

App

OS



### ③ High level Semantics (~~fuller perspective~~)

RPC (Remote Procedure Call)

→ file

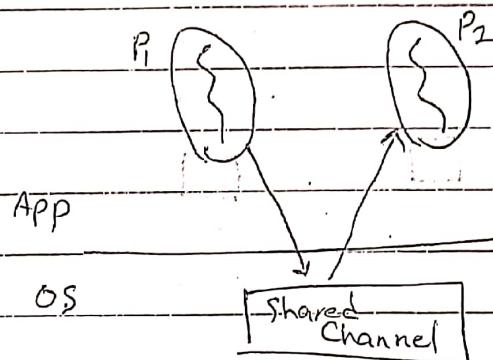
### ④ Synchronization

→ Socket :

نحوه ارسال اطلاعات از port یکی

: firs APJ to socket via

send()  
recv() ] Pass to/from msg buffer



در فرایند ها در دستگاهین مخازن این داده ها ذخیره شوند

سپس از آن داده ها برای انتقال از یک فرایند به دیگر فرایند استفاده شود

→ Pipe :

Stream

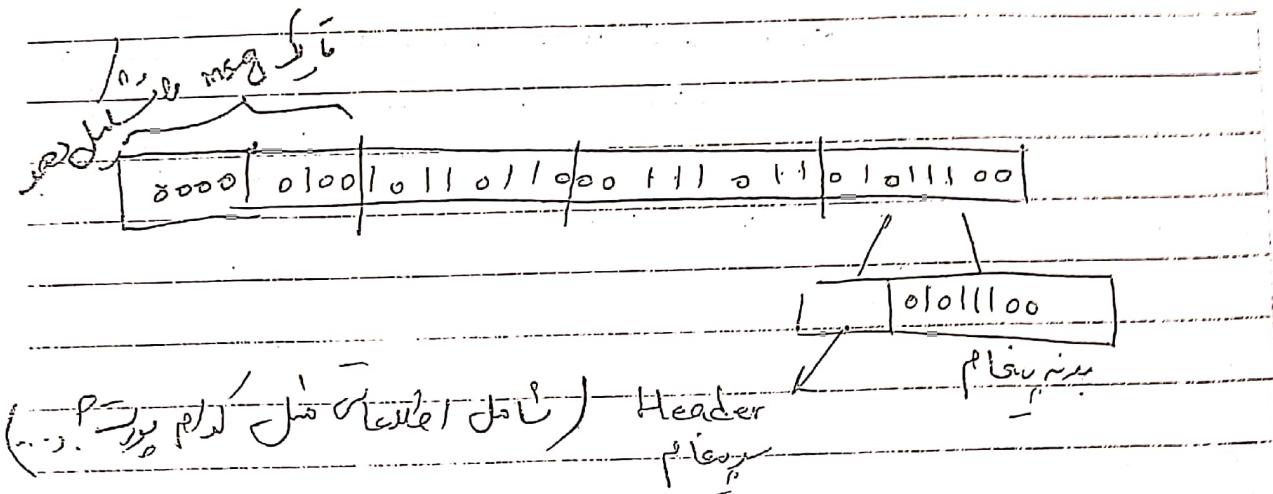
ین داده های از یک فرایند به دیگر فرایند انتقال می شوند

( Data Stream File )

msg queue :

لیست انتظار

Message



در pipe طردها بصورت توالی ارسال می شوند لیکن در msg queue داره

بصورت بیعام ارسال می شوند

msg queue پر توکلن سیریت اولویت داره بلک ولی در pipe

آخر بیعام ذرا فی اولویت بسته برآن بیعام را ارسال می کند.

در زمان بینی با اولویت درایل msg queue

\* دوست socket هم از زمان استفاده نمی کند.

(ج) بیعام (بر)

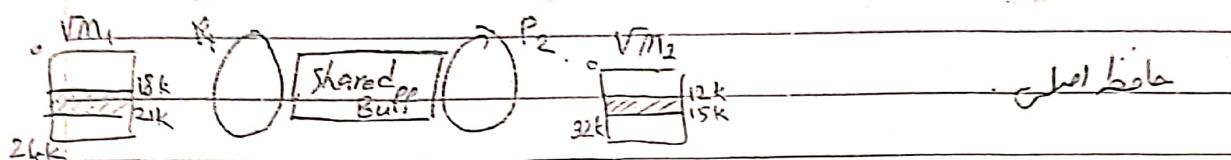
غایل و بروز شناسی POSIX, Sys V, Socket

- Portable OS Interface

استفاده می شوند

## Shared Memory

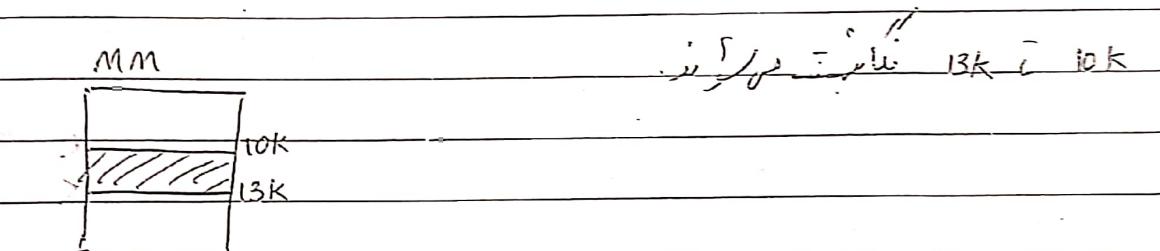
نحوه OS: Shared Buffer فرآیند حافظه خارجی از حافظه



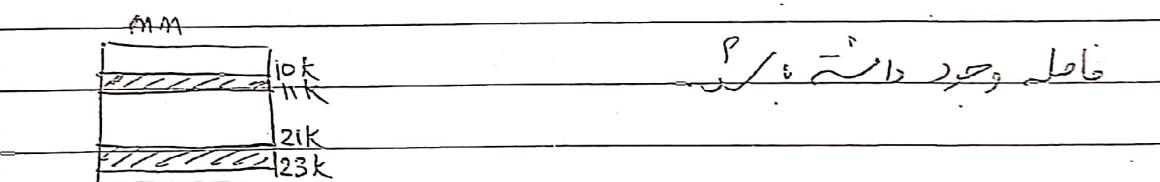
OS

هر دوست بدل مسند از حافظه

اصلی شد نوشت هر دوی خانهای حافظه ای از



حین این مشارک حافظه تخصیص یافته را



(Ctx. Sw) Message passing سیگنالاتور نسبت

- هب کران وحدت مادر (ایجاد سربر)

منکران سازی

Clipart

مداخلہ ایجاد کرنا

Sys. call مداخلہ ایجاد کرنے کا

لکھنؤ عصمن

لکھنؤ اور سیگنال مداخلہ ایجاد کرنے کا باصرہ رابطہ استوں کیا جائے۔  
ارتباط دستیق بین شرکتیں از طریق دسترسی تجارتی استوں۔

Msg passing vs Shared Memory

+ OS بھٹک دار مدخلہ ایجاد و نلاسٹ  
- مخفی طبیعت میتوں دھیل لئے  
- دار

+ تعداد دفعات کم کریں جوں

Context switching

کم کریں

- More data copies

+ تعداد دفعات کم کریں جوں Ctx.Sw

کم کریں

کم کریں

Shared Buffer

(msg passing)

(Shared memory)

Copy (msg)

map

CPU cycles for

CPU cycles to map

copying to/from  
port

into address space

t(cpy)

CPU cycles to copy

N \* t(cpy)

data to buffer

↓

t(map)

↓

↓

 $N \times t(\text{cpy}) \gg t(\text{map})$ 

التيجيه Shared memory

دعا

التيجيه msg passing

التيجيه المهمة توزيع الملفات

التيجيه Shared Mem.

تعريف Shared Mem. في Sys. V

لـ

Segments of shared memory

shared memory get

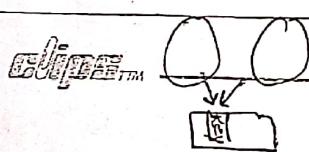
1) Create

OS allocates Memory

shmget( )

f\_tok( )

F token



OS بـ key إلى ...

2) Attach

shmatt()

map VM  $\rightarrow$  Physical Mem

3) Detach

shmctl()

حذف مappings (حذف مappings) invalidate address mapping  
("سترس دار")

4) Destroy

shmctl()

جهاز در ترانس. key بفرماسی محتوى ارسال

P دفع

فرماسی در تردد OS انجام فرمود که فرآیند مستمر است که هنگان

این اینtra process communication برقرار نماید

لین پر میتوان از طریق Shared Mem

Linux IPC Commands:

- ipcs: در kernel از IPC مربوط شده میباشد  
- آنرا مشاهده میکنیم

ipcrm -m [shmid]

از سین بدل کن وارد شوید id

del

اگر  $\Delta$  سیگنال داریم  $\Delta$  کو  $\Delta$  سیگنال کہا جاتا ہے Segment کہا جاتا ہے

خوبی اسکے لئے مفہومیت کیا ہے؟

اگر  $\Delta$  کو Segment کہا جائے تو اس کا معنی چیزیں ہے۔

اگر اندازہ داری کا بڑا جیسا Max Segment Size ہے تو

اگر  $\Delta$  کو Segment کہا جائے تو اس کی محدودیت اس کا انتظام ہے۔

ولیکن درست داری کا سیکھلہ کیتھی تری مراجم ہے۔

اگر  $\Delta$  کو Segment کر جائے تو اس کی محدودیت اس کا انتظام ہے۔

ابنط منیر ب حینا مراجیہ Segment کا خاصیت ہے (ایو)

خوبی اسکے لئے اس کی مفہومیت کیا ہے؟

اگر اندازہ داری کا بڑا جیسا Max Segment Size ہے تو اس کی محدودیت اس کا انتظام ہے۔

میتوانیں داری کو Segment کر رہا ہیں اور اس کی محدودیت اس کا انتظام ہے۔

داری کی محدودیت اس کا انتظام ہے۔