



Threads and Synchronization

1



Announcement

- csil-vmserve1 accounts and group permissions
 - TSG suggested that sharing account passwords violates their policy
 - Instead, TSG suggested to create groups such that files and directories in individual user accounts can be made read/write/executable by all group members, if so desired.
 - To give permissions for your group members to access a file simply use:
 \$ chmod g+[rwx] filename
- Please e-mail me who is in your group
 - Subject line should read: "CS423 GROUP"
 - Body should contain:
 "member last name, member firstname, member netIDs",
 one member per line.

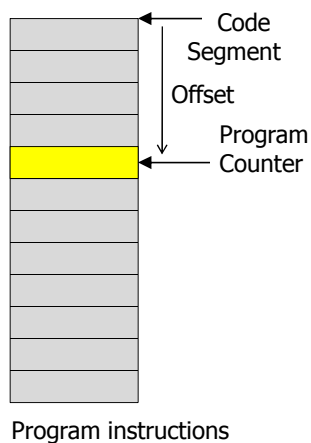
2

Implementing Multiple Threads (on One Processor)

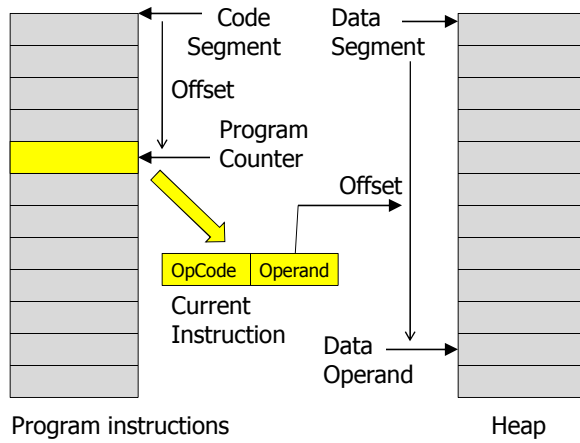
- Threads provide an illusion of concurrency
 - Given one "real CPU", give the illusion of multiple "virtual CPUs" each dedicated to a thread
 - The "virtual CPUs" transparently share the "real CPU"
- The abstraction is recursive
 - If the virtual CPU abstraction is perfect, a "virtual CPU" is indistinguishable from a "real CPU"
 - Therefore, a "virtual CPU" can play the role of a real CPU and can give the illusion of multiple (2nd-level) "virtual CPUs" transparently sharing the 1st-level virtual CPU.

3

What's a "Real CPU"?

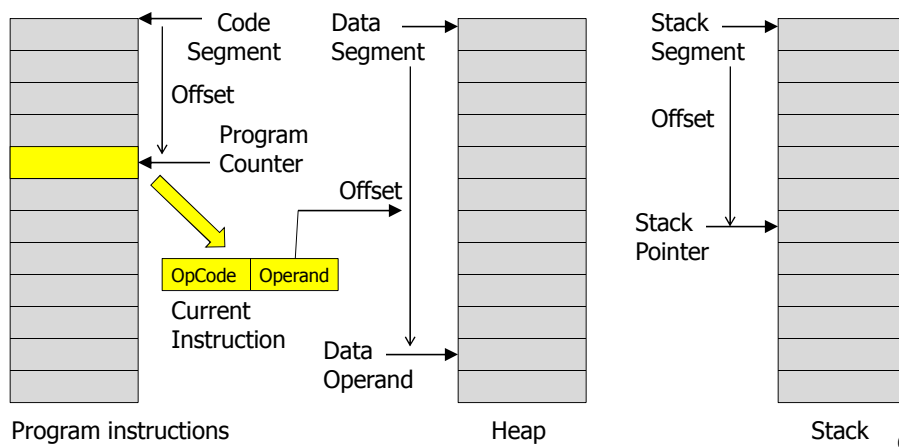


What's a "Real CPU"?

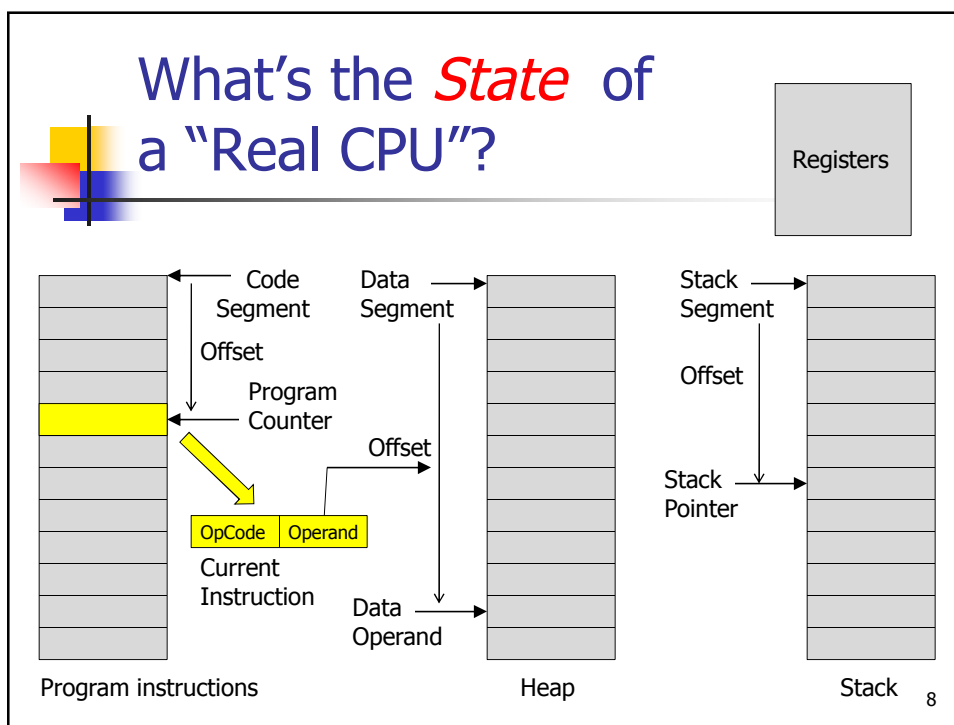
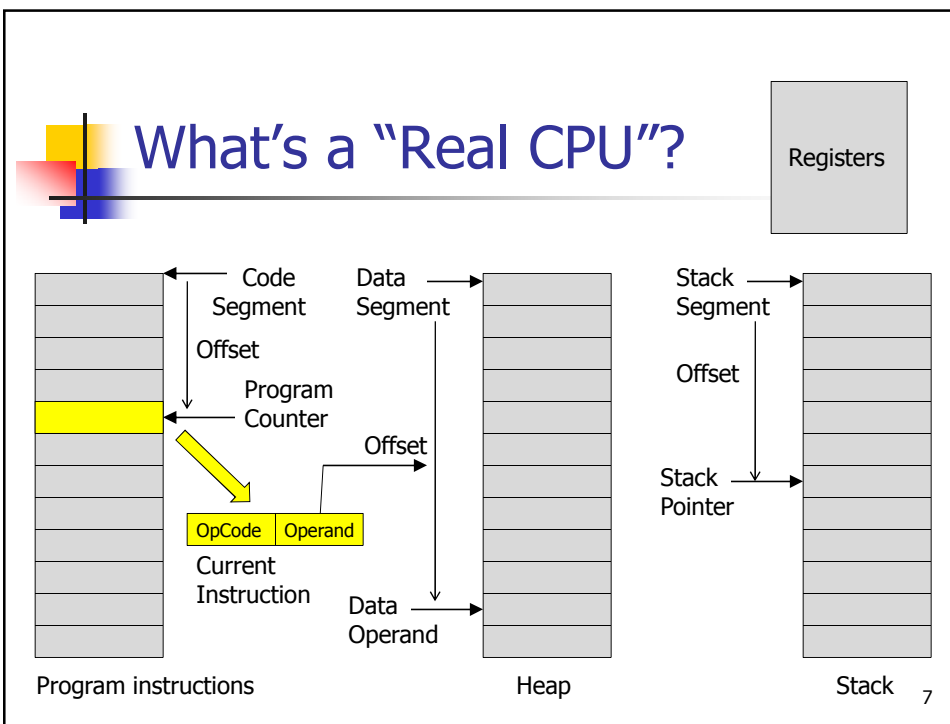


5

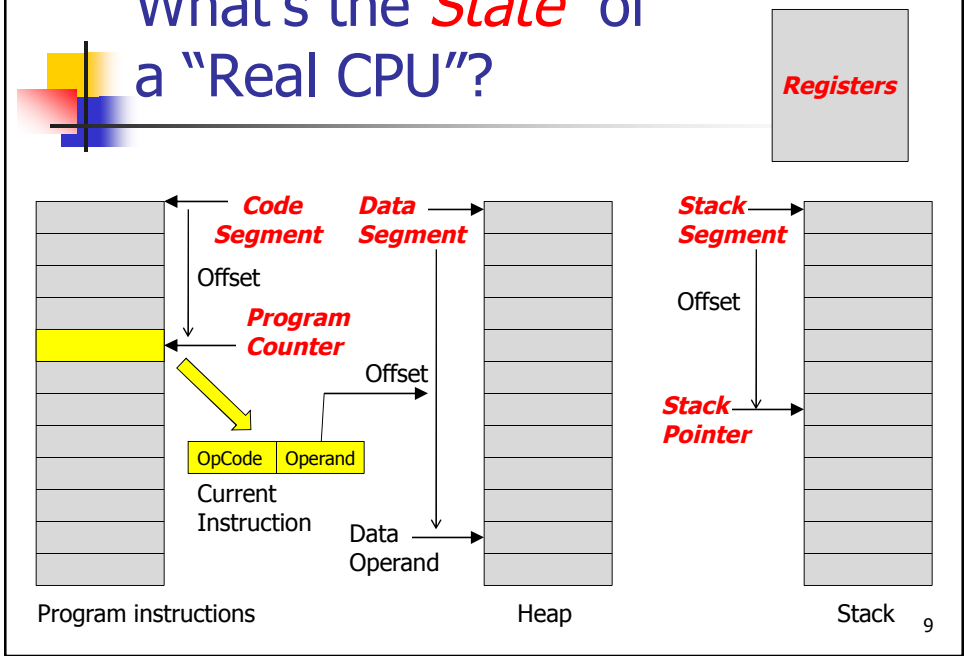
What's a "Real CPU"?



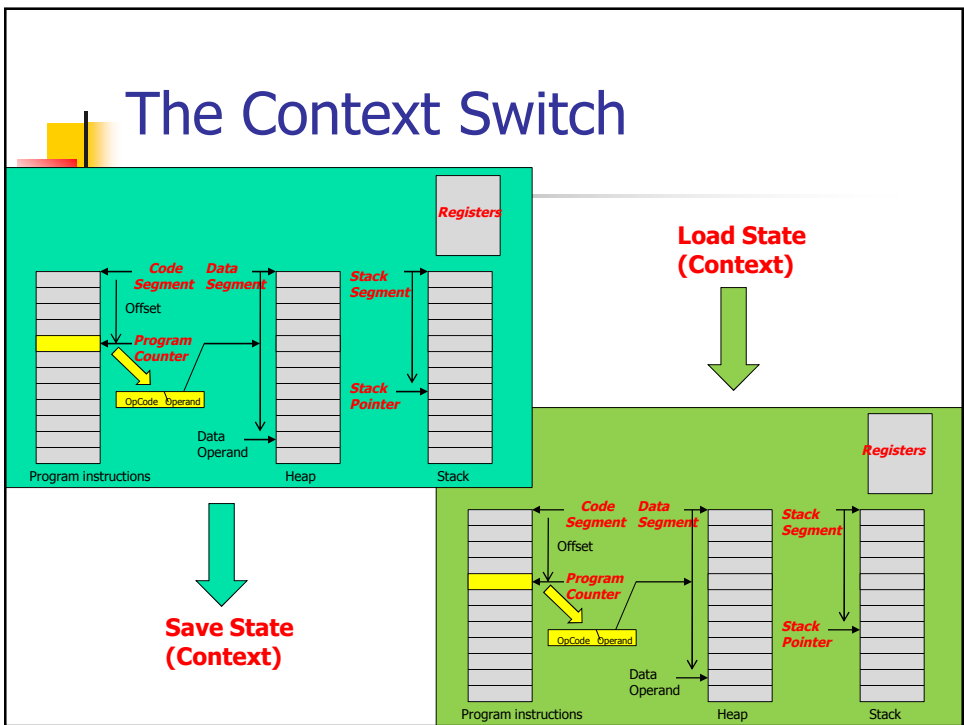
6



What's the *State* of a "Real CPU"?

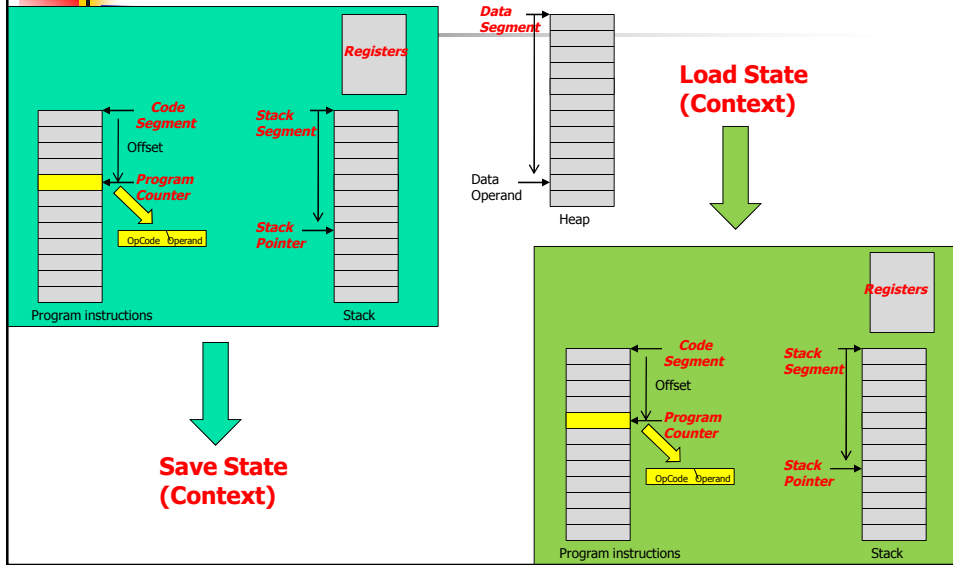


The Context Switch



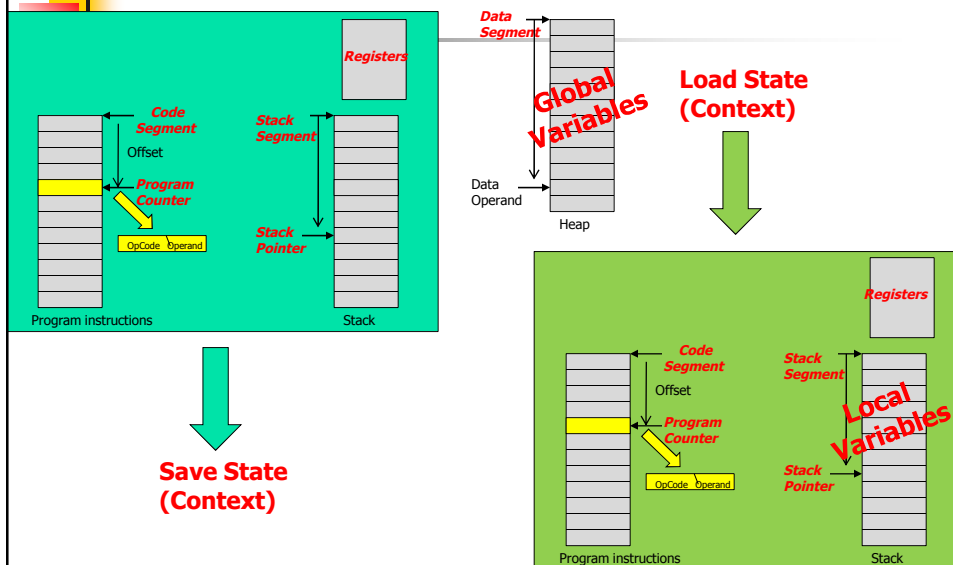
Side Note The Context Switch

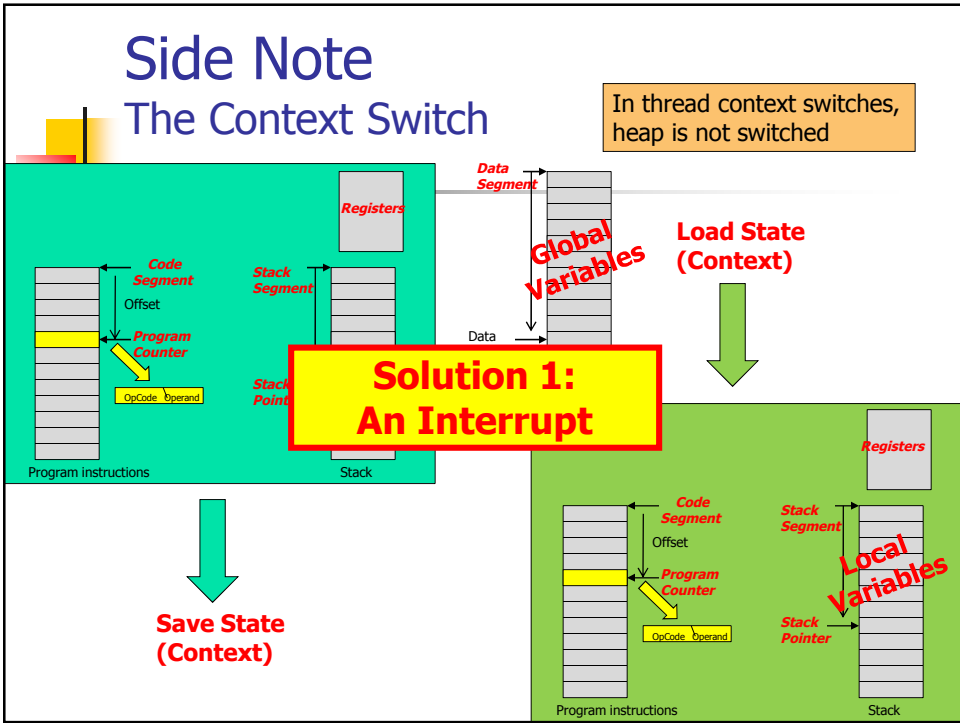
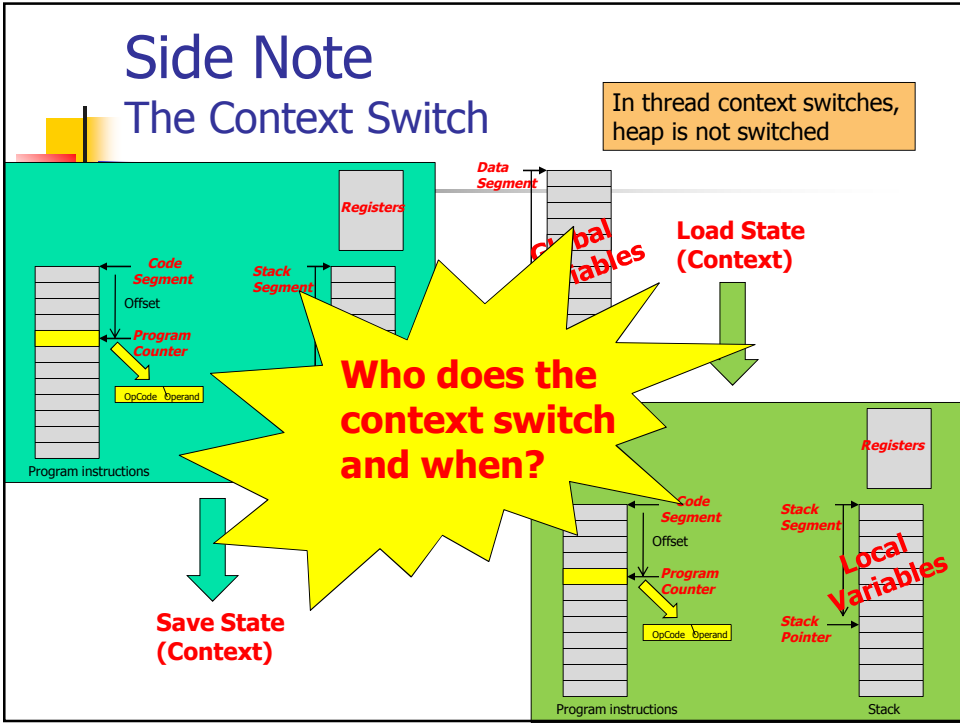
In thread context switches, heap is not switched



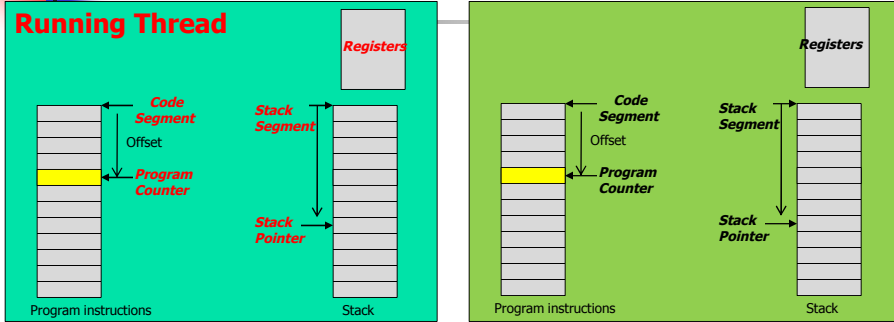
Side Note The Context Switch

In thread context switches, heap is not switched

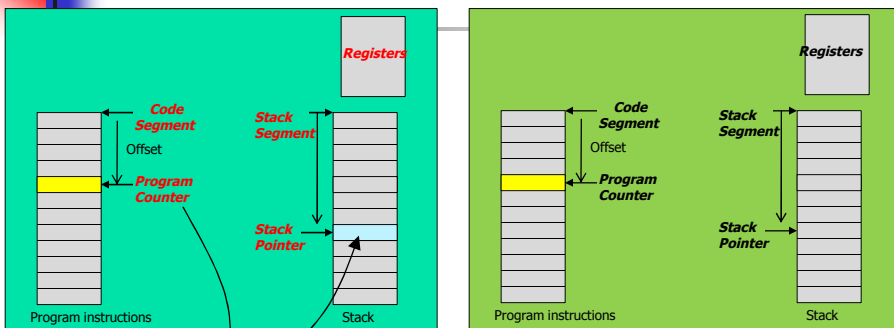




Context Switch (Interrupt)

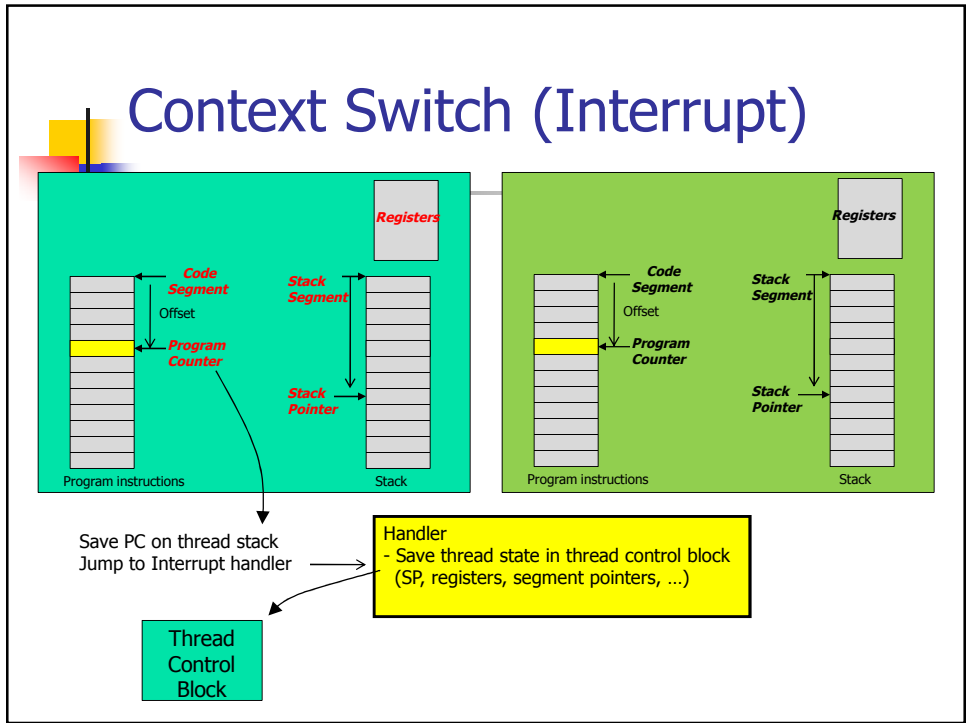


Context Switch (Interrupt)

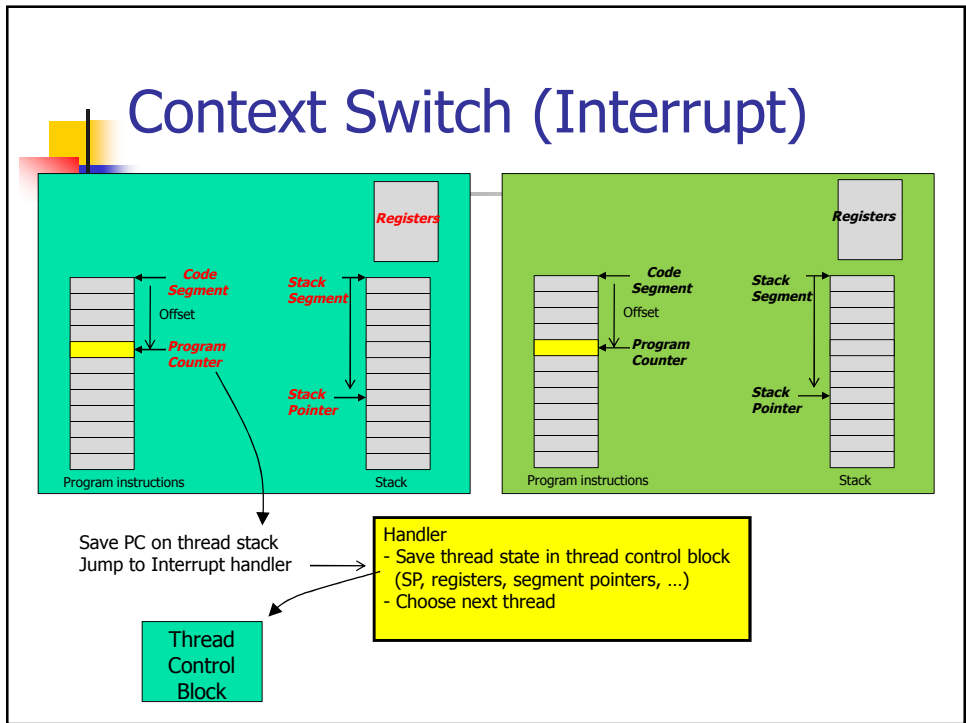


Interrupt
Save PC on thread stack
Jump to Interrupt handler

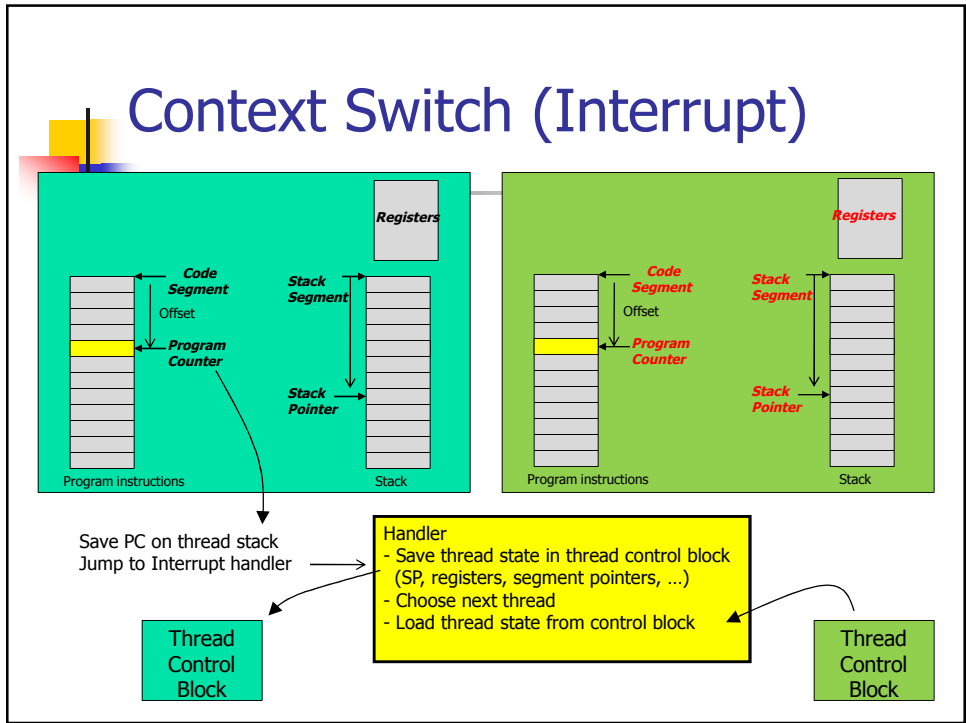
Context Switch (Interrupt)



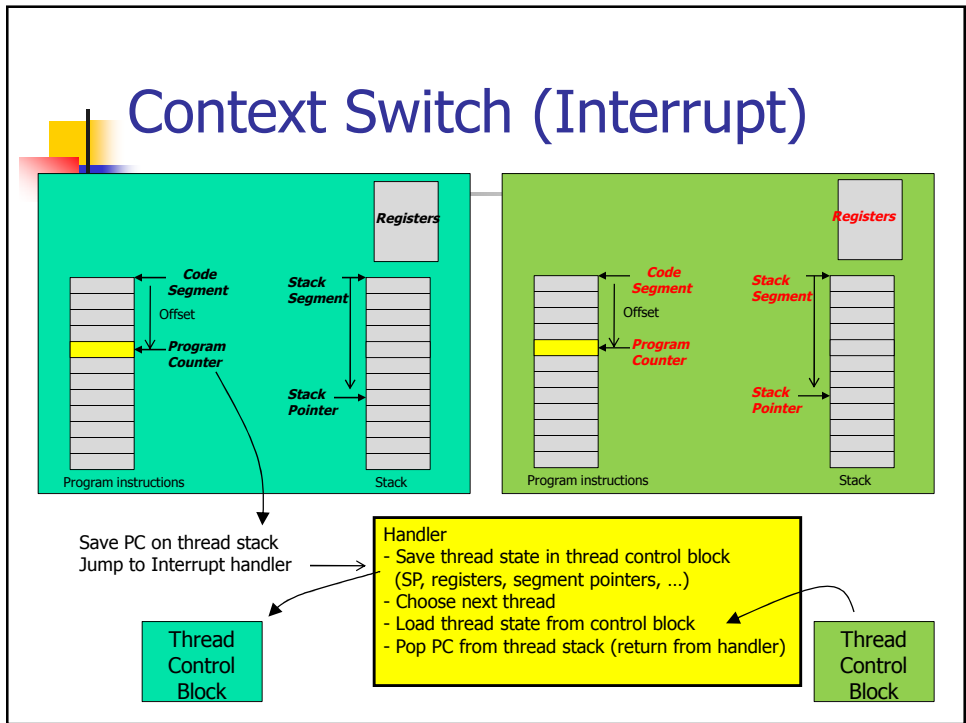
Context Switch (Interrupt)



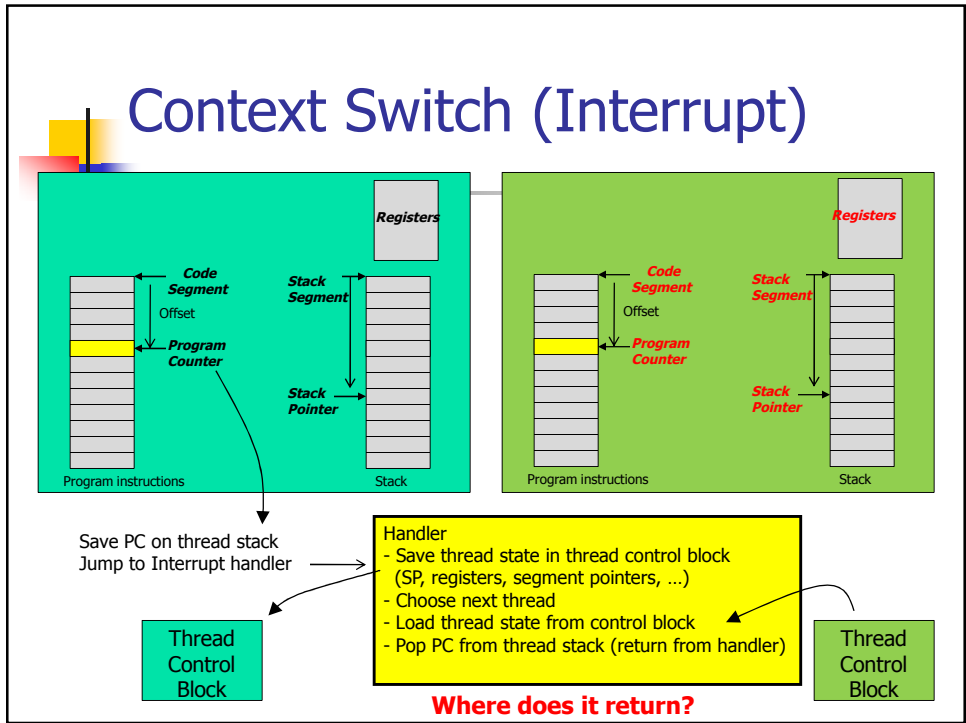
Context Switch (Interrupt)



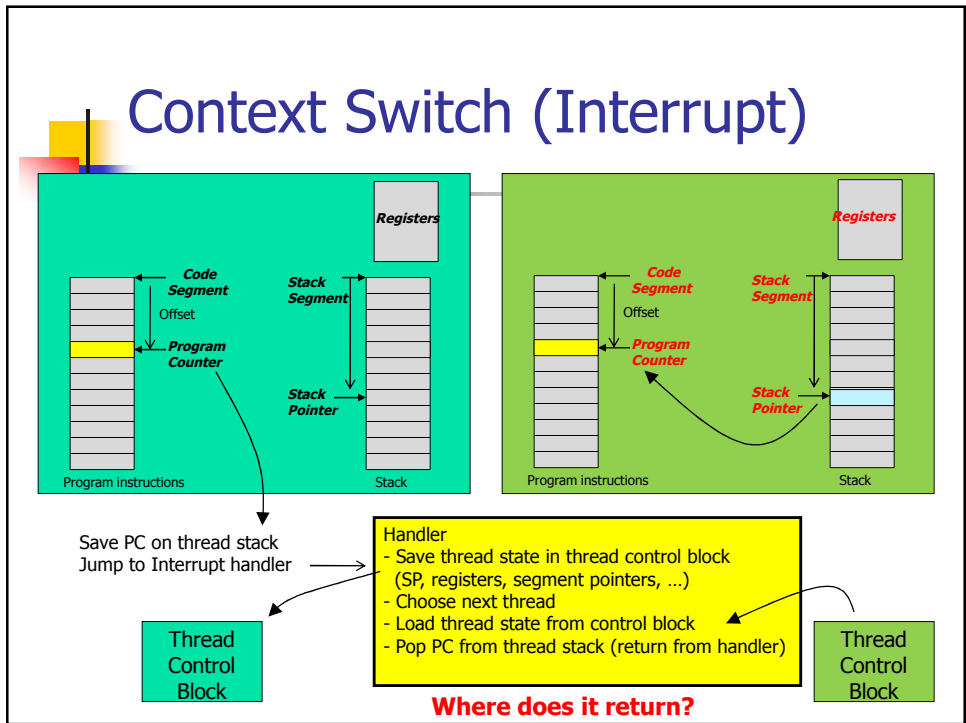
Context Switch (Interrupt)



Context Switch (Interrupt)

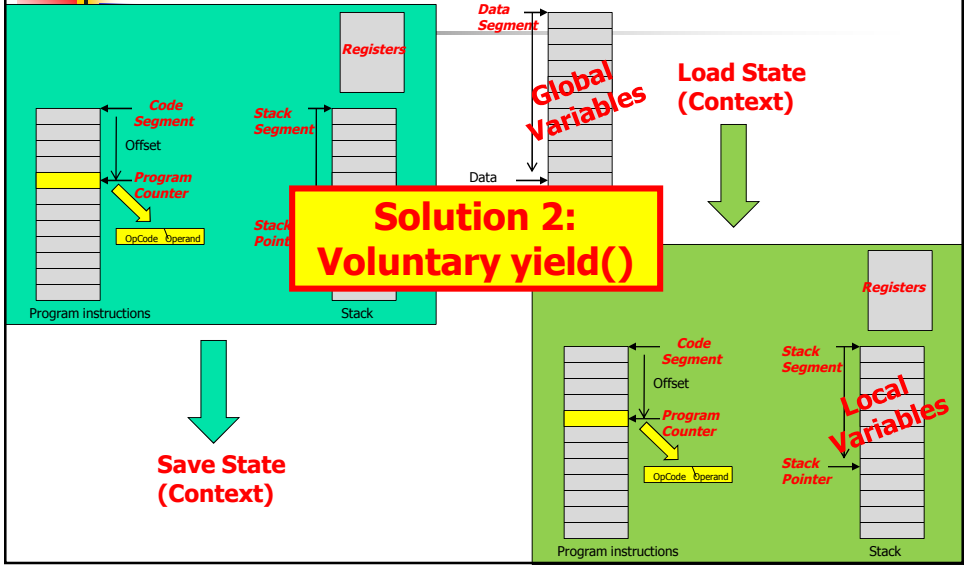


Context Switch (Interrupt)

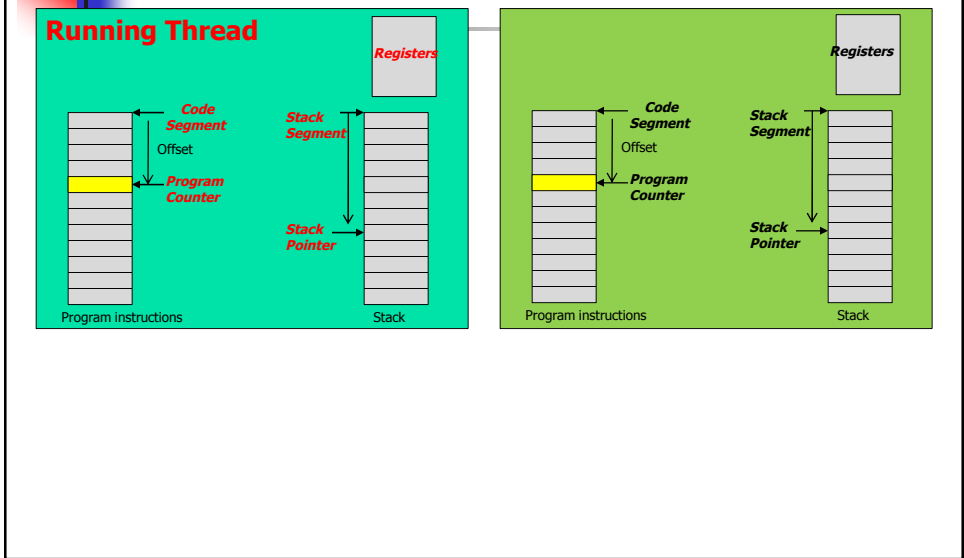


Context Switch

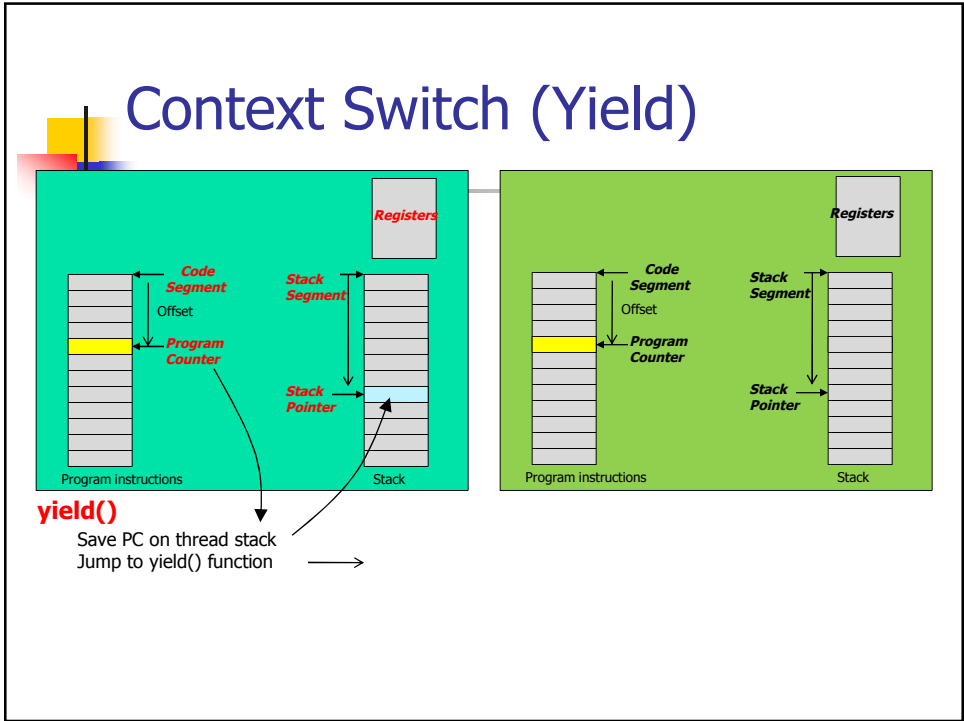
In thread context switches, heap is not switched



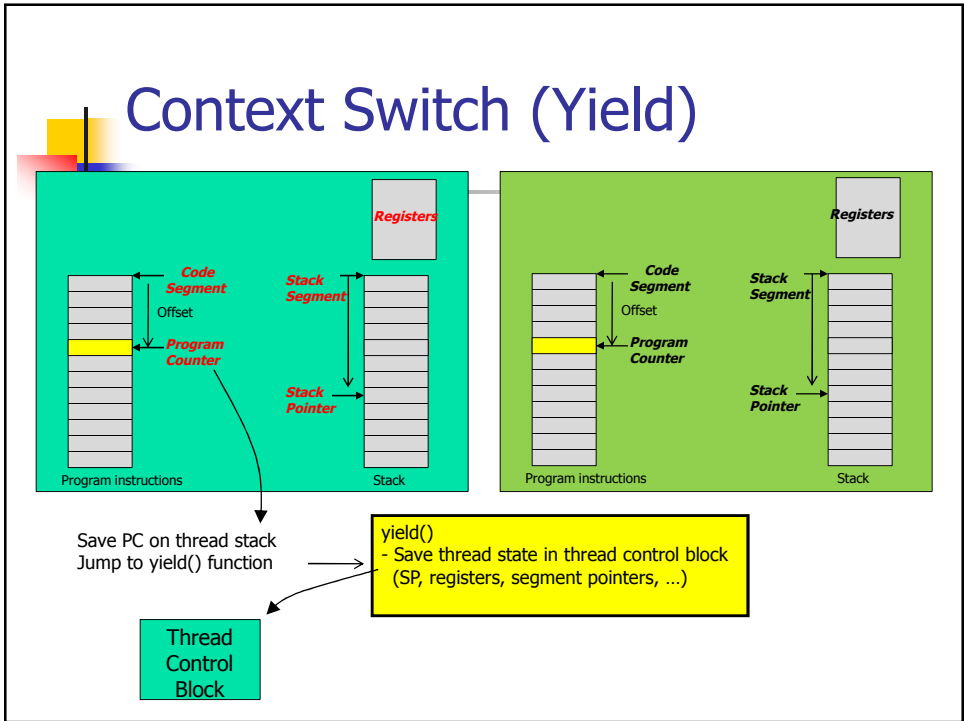
Context Switch (Yield)



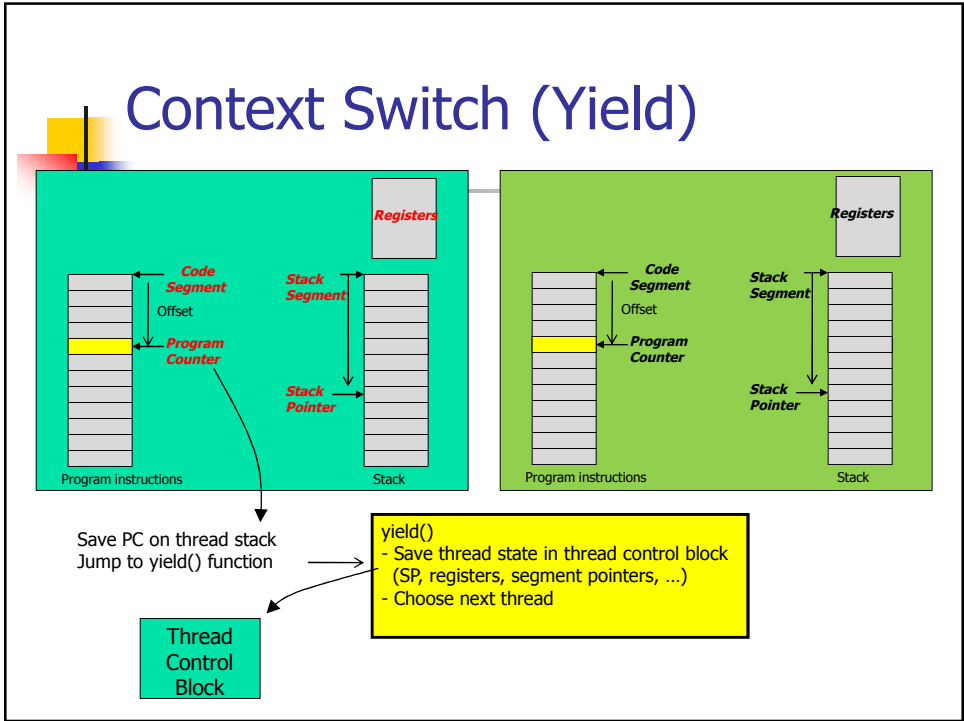
Context Switch (Yield)



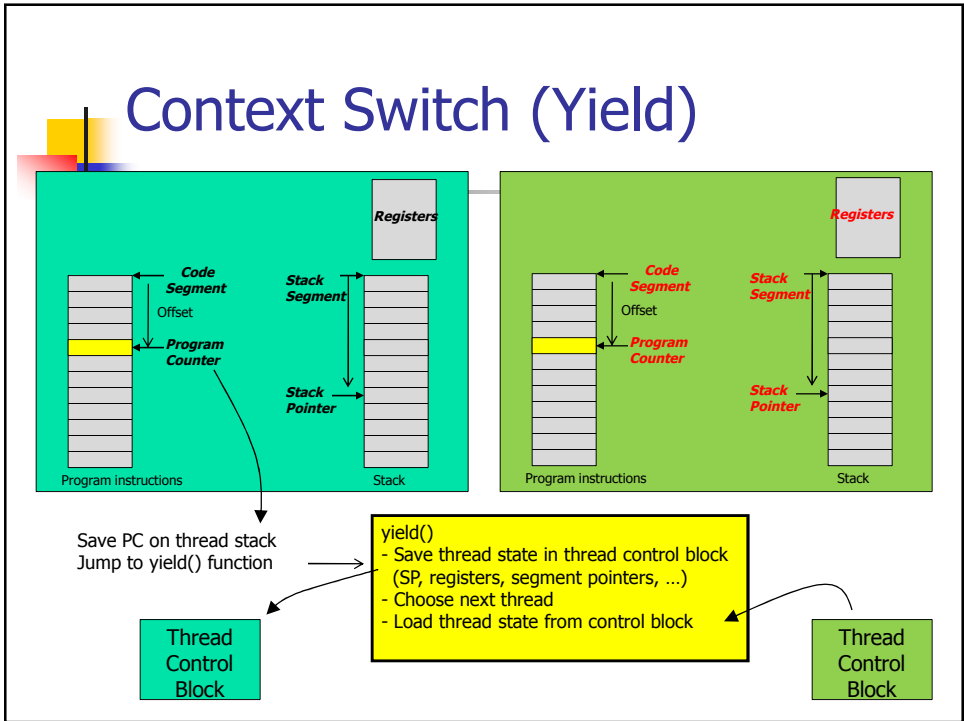
Context Switch (Yield)



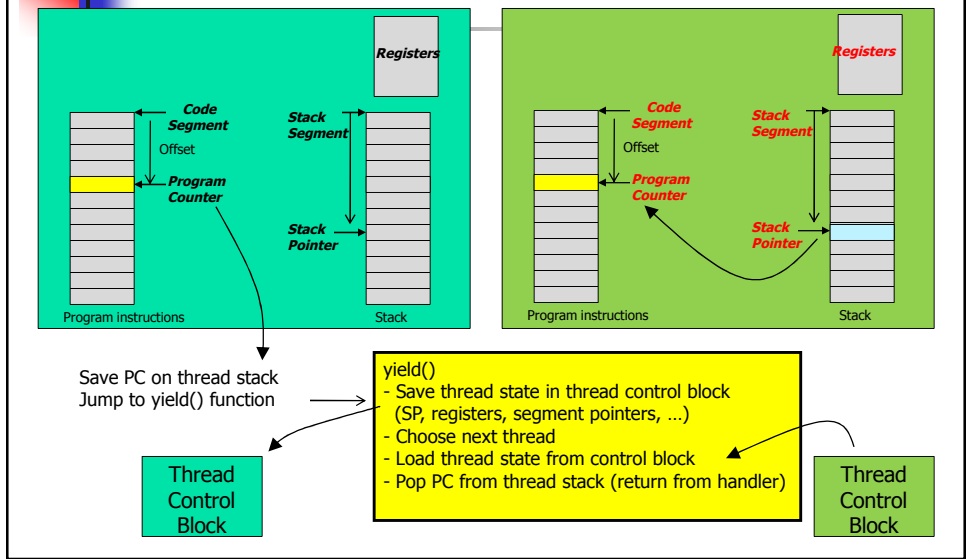
Context Switch (Yield)



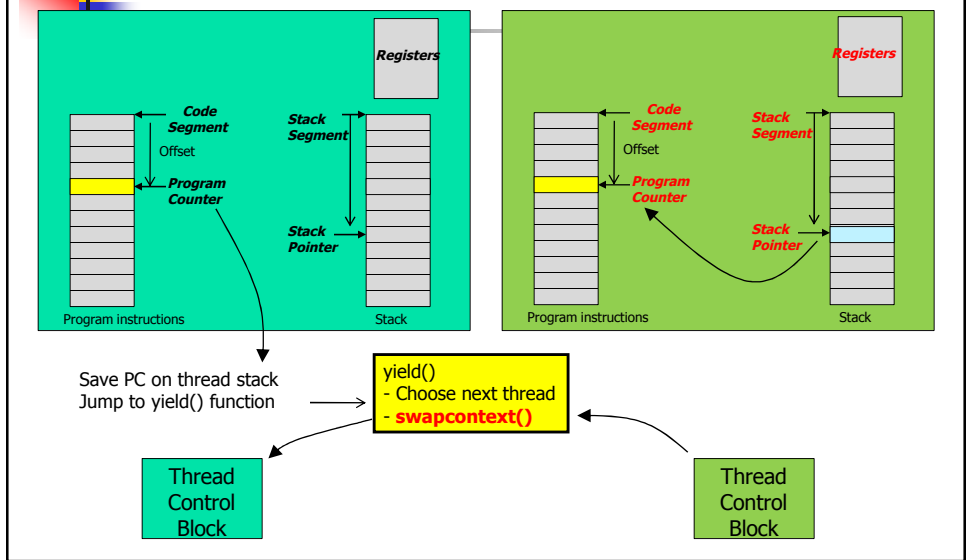
Context Switch (Yield)



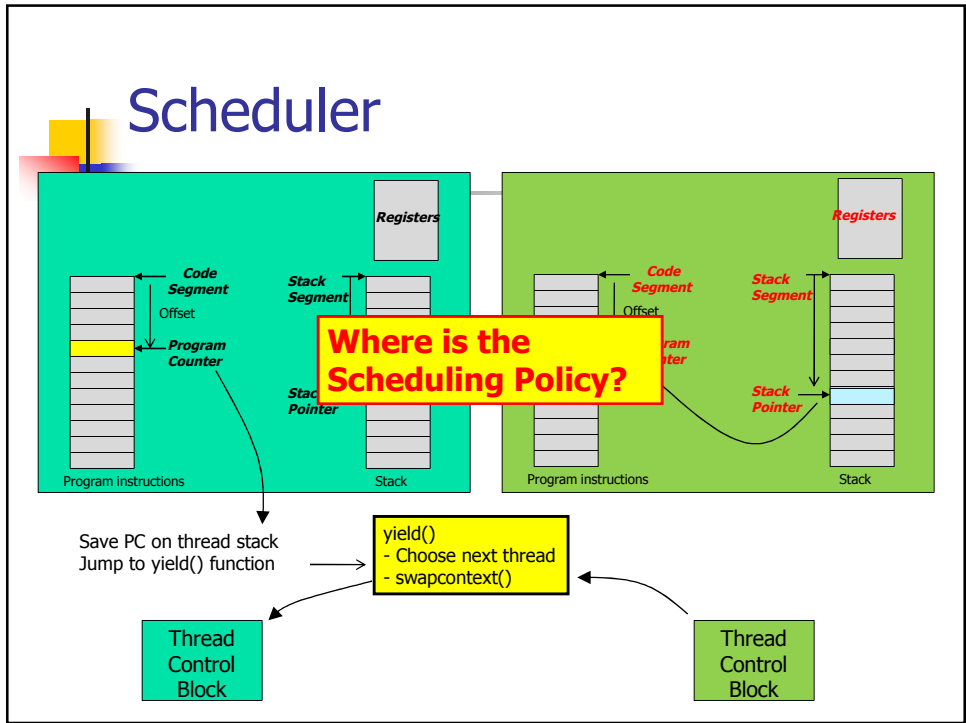
Context Switch (Yield)



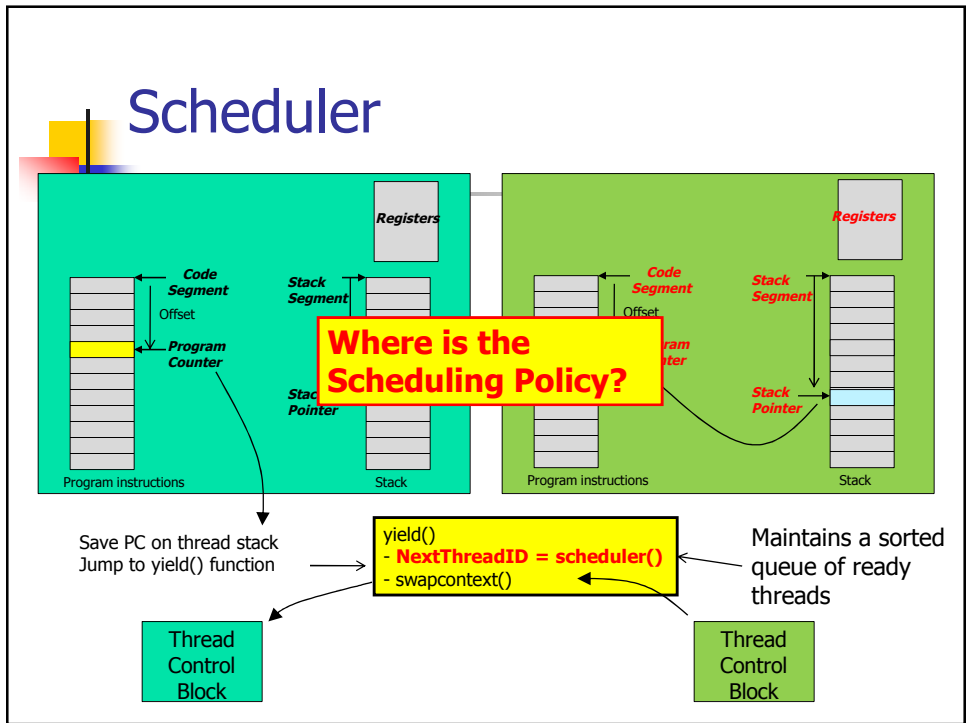
Linux User Level Implementation



Scheduler



Scheduler





Issues

- Initialization?
- Interrupt context switch versus yield?

33



Issues

- Initialization?
 - Stack for each thread needs to be allocated a priori → just malloc() the right sized chunk.
 - Thread control blocks must be created
- Interrupt context switch versus yield?

34



Issues

- Initialization?
 - Stack for each thread needs to be allocated a priori → just malloc() the right sized chunk.
 - Thread control blocks must be created
- Interrupt context switch versus yield?
 - Yield is voluntary. What if application did not yield?
 - Yield cannot do preemptive scheduling.

35



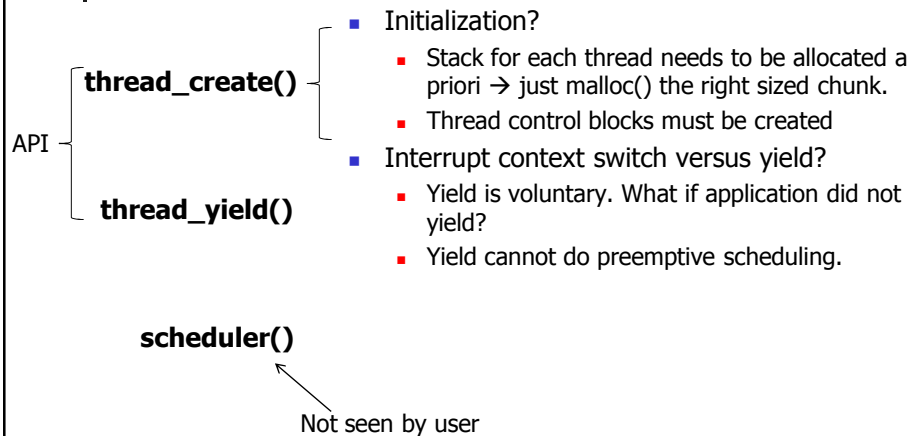
A User-level Thread Package

- Initialization?
 - Stack for each thread needs to be allocated a priori → just malloc() the right sized chunk.
 - Thread control blocks must be created
- Interrupt context switch versus yield?
 - Yield is voluntary. What if application did not yield?
 - Yield cannot do preemptive scheduling.

36



A User-level Thread Package



37



Review

- How do you implement a context switch?
 - In cooperative user-level threads?
 - In kernel threads?
- How do you implement a thread scheduler?
- Trade-offs between cooperative and non-cooperative thread models?

38