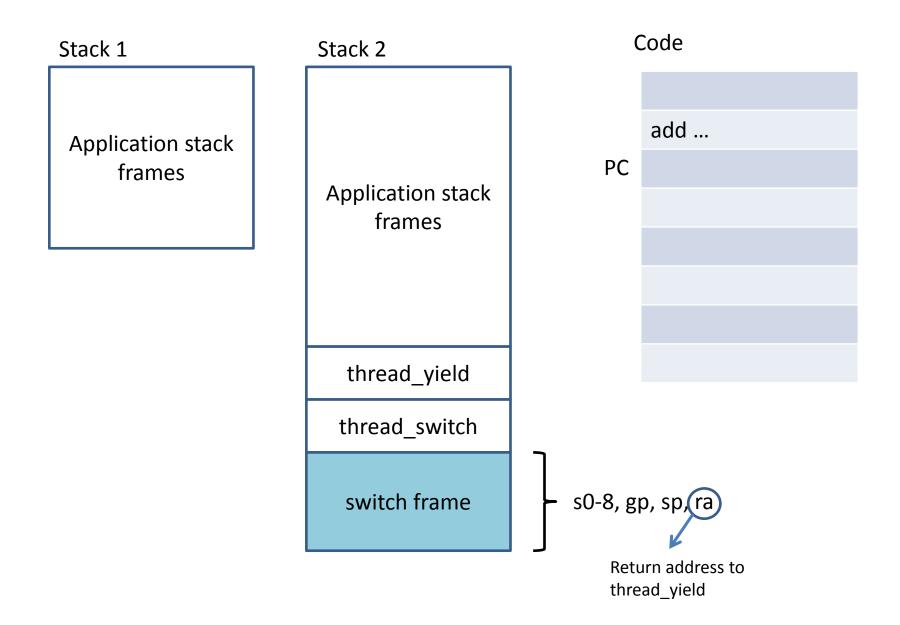
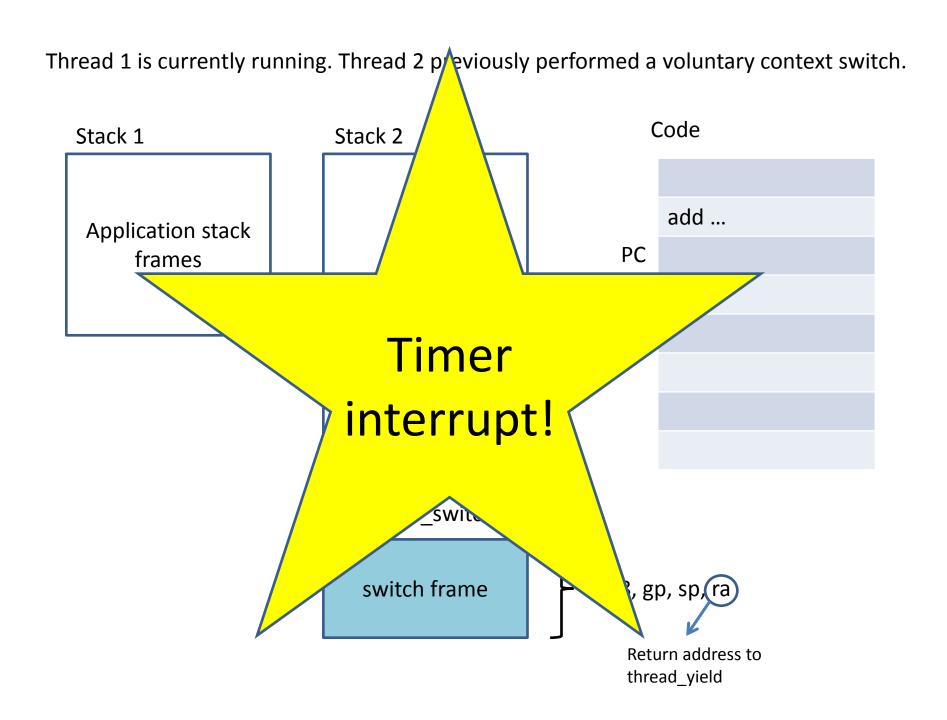
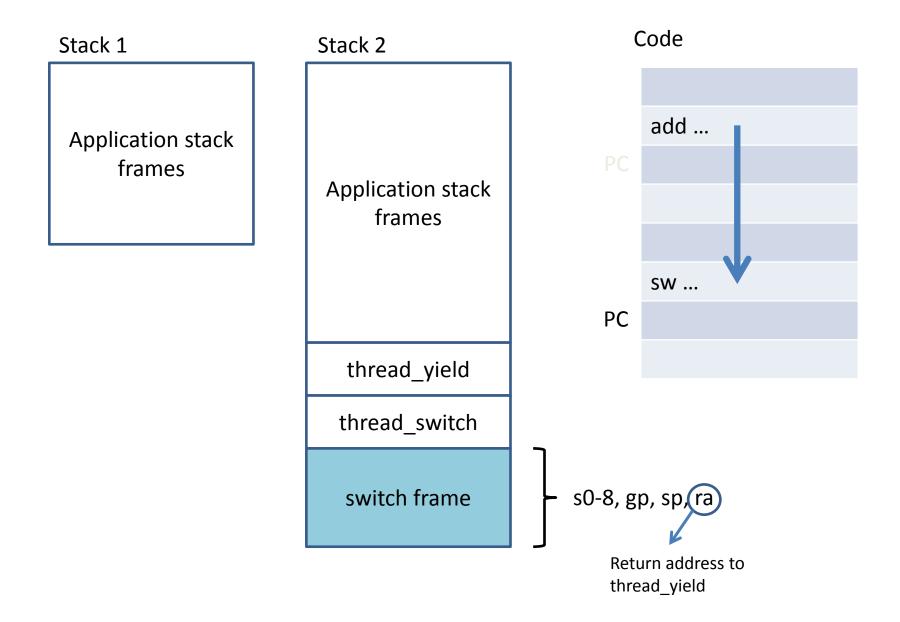
Thread 1 is currently running. Thread 2 previously performed a voluntary context switch.

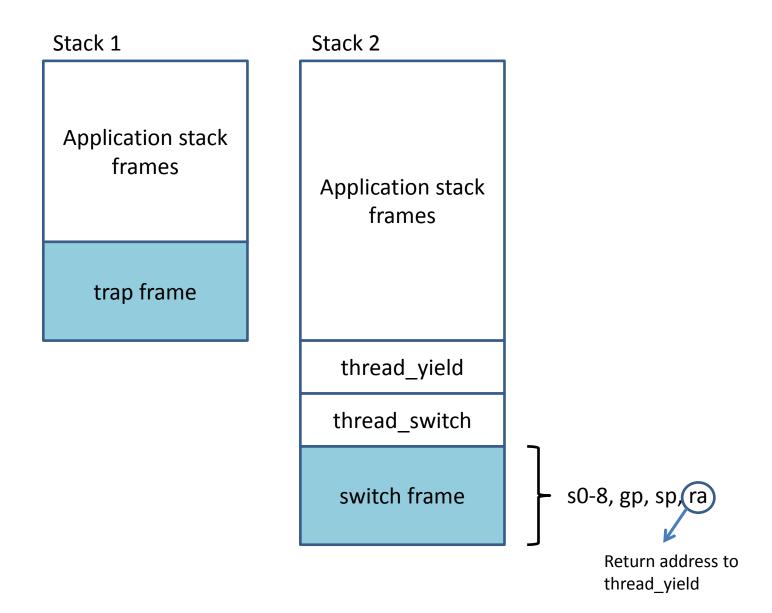




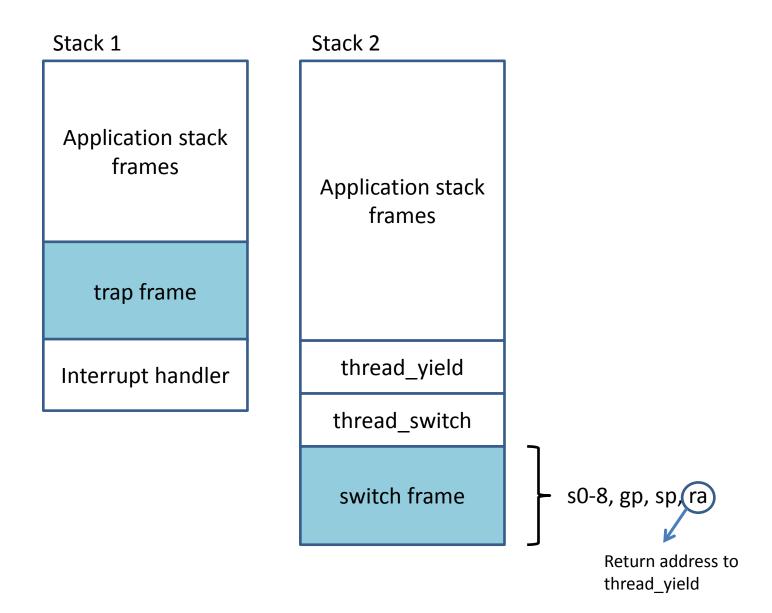
Transfers control to a fixed location in memory



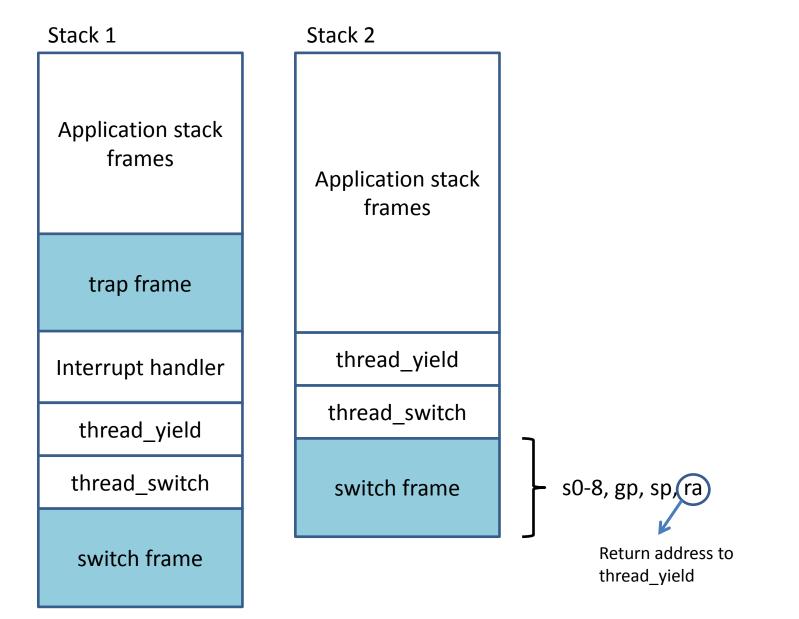
Save the current context into the trap frame. Must save every register.



Call interrupt handler. Identify the device that raised the interrupt.



Running total counter equals to the quantum. Schedule thread 2 for execution. Perform context switch.



Restore the context from the switch frame. Returns to thread_yield using the value in return address (ra) register.

Stack 1 Stack 2 Application stack frames Application stack frames trap frame thread_yield Interrupt handler thread_yield thread_switch switch frame

Return from thread_yield to continue executing the application.

Stack 1

Application stack frames

trap frame

Interrupt handler

thread_yield

thread_switch

switch frame

Stack 2

Application stack frames

Context switch back to thread 1.

Stack 1

Application stack frames

trap frame

Interrupt handler

thread_yield

thread_switch

switch frame

Stack 2

Application stack frames

thread_yield

thread_switch

switch frame

Restore context from the switch frame and return on the ra register to continue in thread_yield.

Stack 1 Stack 2 Application stack frames Application stack frames trap frame thread_yield Interrupt handler thread_switch thread_yield switch frame

After returning from the interrupt handler, restore the context in the trap frame and return to the application.

Stack 1

Application stack frames

Stack 2

Application stack frames

thread_yield

thread_switch

switch frame