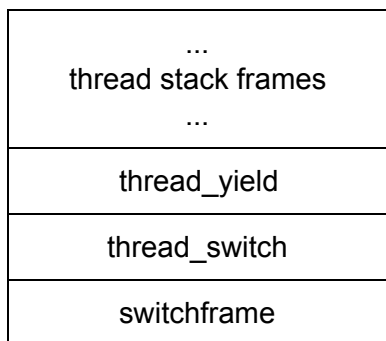


## Context Switches and Switchframe

A context switch occurs when a thread:

1. Calls `thread_yield`
2. Exits
3. Blocks
4. Is preempted

The assembly function **switchframe\_switch** performs the *low-level* context switch, and is called by **thread\_switch**, which is called by **thread\_yield**. On the stack, if a thread voluntarily calls `thread_yield`, this would look like:



**thread\_switch** (*high-level* context switching code):

1. Finds the next thread to run
2. Calls **switchframe\_switch** to perform the *low-level* context switch

The **ABI** (application binary interface) of OS/161 states that the calling function must save the t-registers. This will be done by **thread\_switch** prior to calling **switchframe\_switch**.

**switchframe\_switch** (*low-level* context switching code):

1. Saves the context of the old thread, whose stack pointer is stored in **a0**
2. Loads the context of the new thread, whose stack pointer is stored in **a1**
3. Jumps to the return address (**ra**)

But whose return address are we jumping to? The **NEW THREAD**.

WHERE does it return to? **thread\_switch in the new thread**.

So, who pops the switchframe stack frame off of the new threads stack? The **OLD THREAD**.

The final instruction of **switchframe\_switch**, is to pop the switchframe stack from off of the **NEW THREADS** stack.