



Lecture # 1.4

Exit Handlers

Course: Advanced Operating System

Instructor: Arif Butt

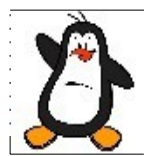
Punjab University College of Information Technology (PUCIT)
University of the Punjab



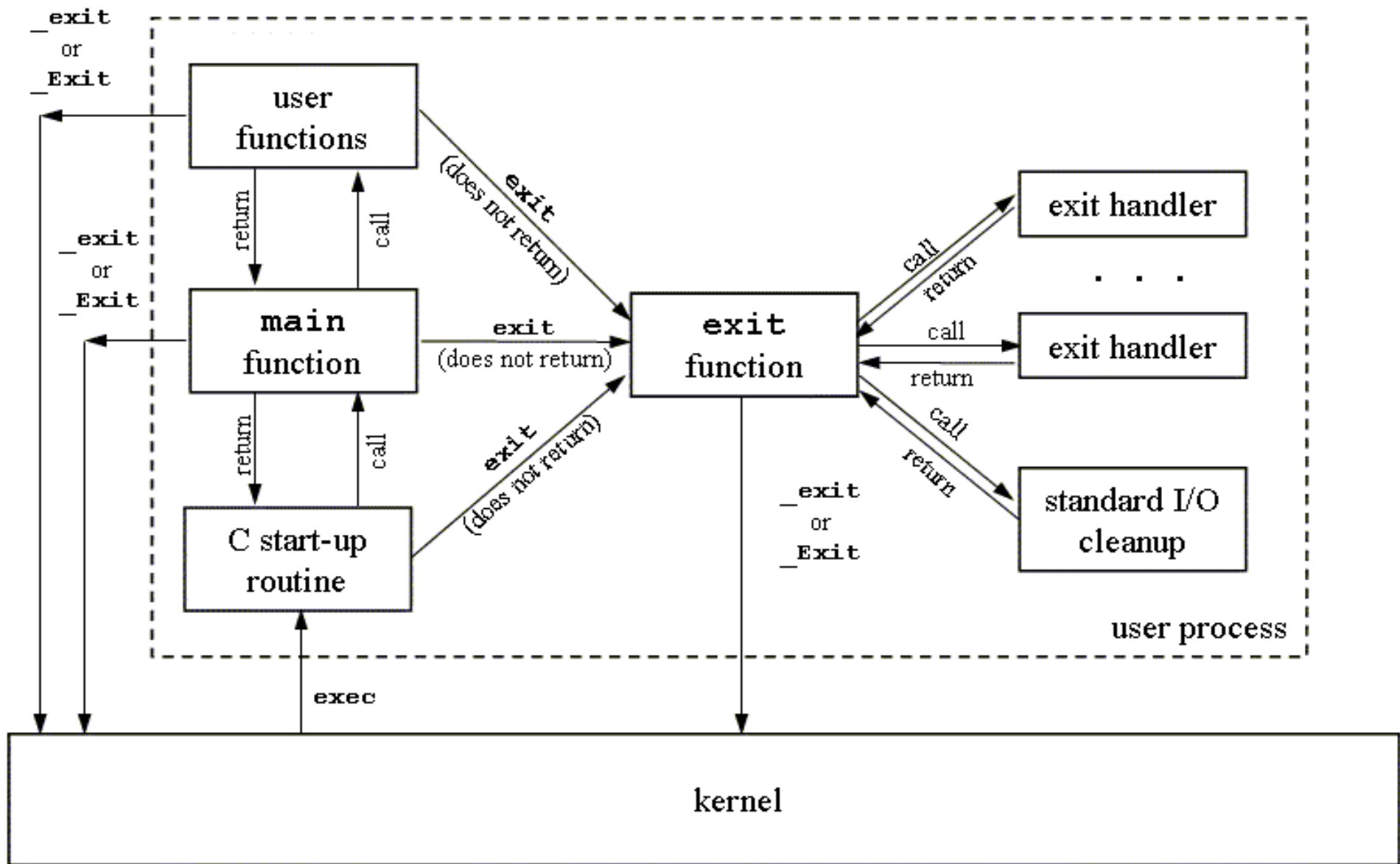
Agenda

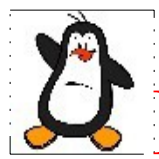
- How a C program start & terminates
- Normal & abnormal termination
- What are exit handlers
- Registering exit handler using `atexit()`
- Registering exit handlers using `on_exit()`
- Process Resource Limits





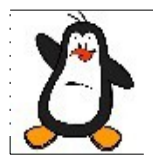
How a C Program Starts & Terminates





How a C Program Terminates

- Normal termination
 - The main function's **return** statement
 - Any function calling **exit()** library call
 - Any function calling **_exit()** system call
- Abnormal termination
 - Calling **abort()** function
 - Terminated by a signal



Limitations of `atexit()`

Limitations of exit handler registered via `atexit()`

- An exit handler doesn't know what exit status was passed to `exit()`; which may be useful. e.g., we may like to perform different actions depending on whether the process is exiting successfully or unsuccessfully
- We can't specify an argument to exit handler when called; which may be useful to define an exit handler that perform different actions depending on its argument



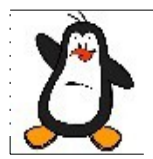
Library Call on `_exit()`

```
int on_exit(void(*func)(int,void*),void*arg);
```

- The `on_exit()` is also used to register exit handlers like `atexit()`, but is a more powerful than `atexit()`
- It accepts two arguments, a function pointer and a void pointer
- The `func` is a function pointer that is passed two arguments (an integer and a void*)
- The first argument to `func` is the integer value passed to `exit()`, and the second argument is the second argument to `on_exit()`

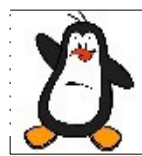


Process Resource Limits



Process Resource Limits

- Every process has a set of resource limits that can be used to restrict the amounts of various system resources that the process may consume
- We can set the resource limits of the shell (terminal) using the **ulimit** built-in command. These limits are inherited by the processes that the shell creates to execute user commands
- Since kernel 2.6.24, the Linux-specific **/proc/PID/limits** file can be used to view all of the resource limits of any process



Things To Do

O.k., and now you'll do exactly what I'm telling you !



If you have problems visit me in counseling hours. . . .
