CS 450 Module R4 Applied Dispatching

West Virginia University

Fall 2025

WestVirginiaUniversity.

Goal
s

Goal
s

- Apply dispatching concepts from R3 to Command Handler
- **R3 must be completely working before beginning R4**
- Implement additional processes to exercise dispatching code

Kernel Functions

`void  kmain(void);`

- Remove the call to your command handler
- Create two System processes:
  - Command Handler, running at the highest priority
  - System Idle Process, running at the lowest priority – the function is provided for you as `sys_idle_process()` in `<processes.h>`
- Uncomment the following line in kmain to begin running Command Handler:

```
//R4:__asm__ volatile (”int $0x60” : : ”a”(IDLE)) ;
```

User Commands

Command Handler

Command Handler must issue a sys_req IDLE request

- Must be treated like any other process
- Make Command Handler opportunistically multitask
    - Remove the Yield command from the user
    - Yield the CPU to other processes immediately before or after prompting for and reading user input
- Ensure shutdown still cleanly shuts down the system

Command Handler must perform a sys_req EXIT

Remove Command: Yield

- Remove the Yield command completely from your comhand

Alarm

- Command (which must spawn a separate ready, non-suspended process!) to display a message at a certain time with two parameters
  - Time – When to display the message
  - Message – The message to display
- Process should be idle before the specified time
- If the current time matches the requested time or later, display the message and exit the process – it is OK for alarms to trigger late, but **never** early
- It should be possible to have multiple Alarm processes running concurrently
- (Multiple pcbs connected to the same function)