

Project 2 Roadmap++

Awww! ugly slides.

Review

Process A

```
main() {  
    for(1000)  
        Print "A"  
    Kill(B, SIGUSR1)  
}
```

Process B

```
function handler() {  
    Print "HANDLING"  
}  
  
main() {  
    Signal(&handler, SIGUSR1)  
    for(;;)  
        Print "B"  
}
```

Output

AABBAABBAABB.....HANDLING BBBB.....

1000 A'S

System Calls

- **Sys_Signal**: register a signal handler
- **Sys_Kill**: send a signal
- **Sys_RegDeliver**: initialize signal handling for a process
- **Sys_WaitNoPID**: wait for any child process to die
- **Sys_ReturnSignal**: indicate completion of signal handler

System Calls

- **Sys_Signal:**
 - **Sys_Kill:**
 - Sys_RegDeliver:
 - Sys_WaitNoPID:
 - Sys_ReturnSignal:
- } Referenced in user code

Review

Process A

```
main() {  
    for(1000)  
        Print "A"  
    Kill(B, SIGUSR1)  
}
```

Process B

```
function handler() {  
    Print "HANDLING"  
}  
  
main() {  
    Signal(&handler, SIGUSR1)  
    for(;;)  
        Print "B"  
}
```

Output

AABBAABBAABB.....HANDLING BBBB.....
└──────────────────┘
1000 A'S

System Calls

- Sys_Signal:
- Sys_Kill:
- Sys_RegDeliver:
- Sys_WaitNoPID:
- Sys_ReturnSignal:

System Calls

- Sys_Signal:
- Sys_Kill:
- Sys_RegDeliver:
- Sys_WaitNoPID:
- Sys_ReturnSignal:

Executed by stub code
once a signal has been
handled

Helper Functions

- Send_Signal
- Set_Handler
- Check_Pending_Signal
- Setup_Frame
- Complete_Handler

Review

Process A

```
main() {  
    for(1000)  
        Print "A"  
    Kill(B, SIGUSR1)  
}
```

Process B

```
function handler() {  
    Print "HANDLING"  
}  
  
main() {  
    Signal(&handler, SIGUSR1)  
    for(;;)  
        Print "B"  
}
```

Overview

A

B

Overview

A

RegDeliver
Signal(SIGCHLD)

B

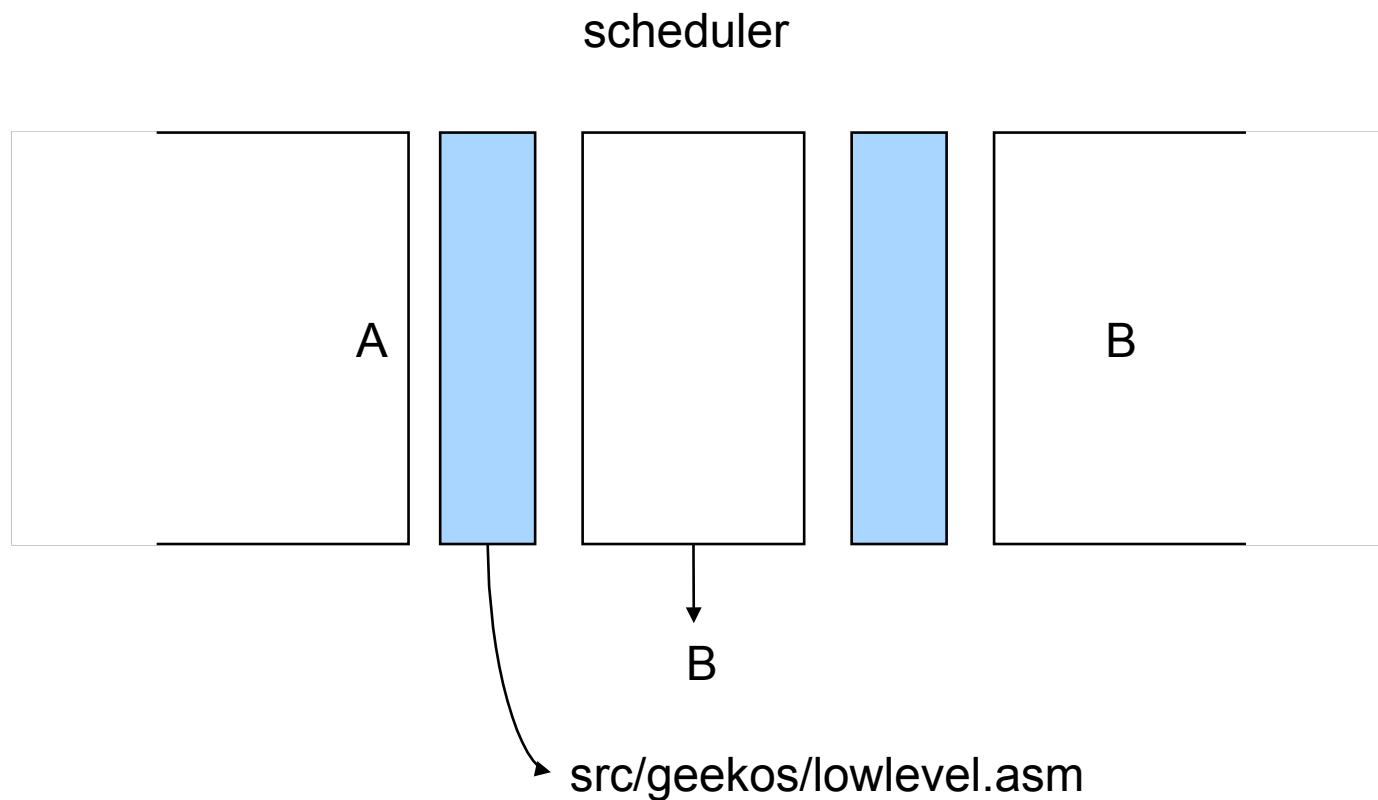
RegDeliver
Signal(SIGCHLD,)

Helper Functions

- Send_Signal
- Set_Handler
- Check_Pending_Signal
- Setup_Frame
- Complete_Handler

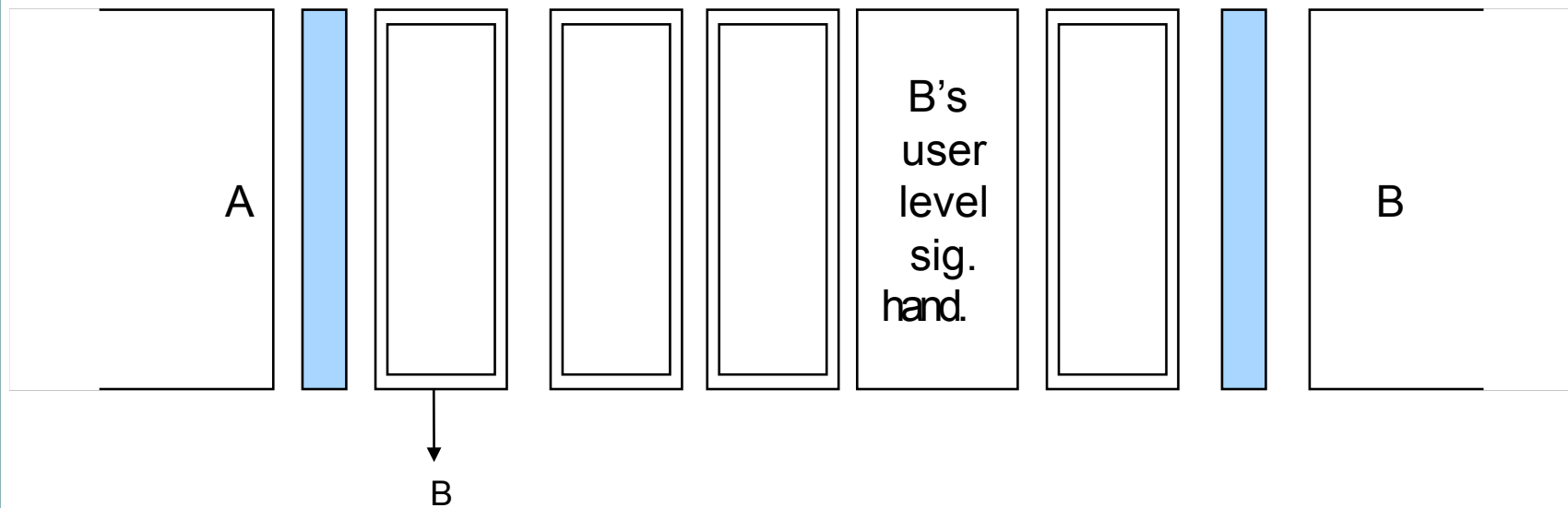
Look at Scheduler

Scheduler: w/o signals

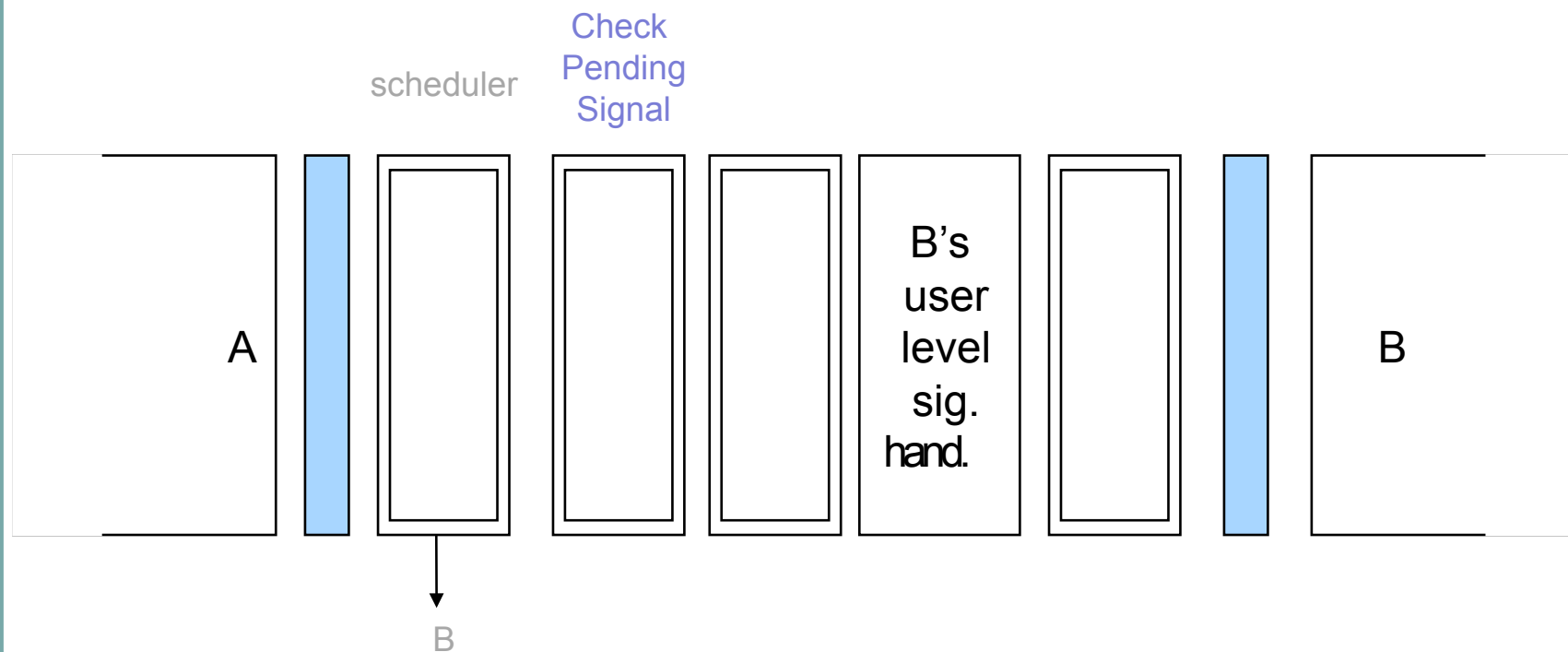


Scheduler: w/ signals

scheduler



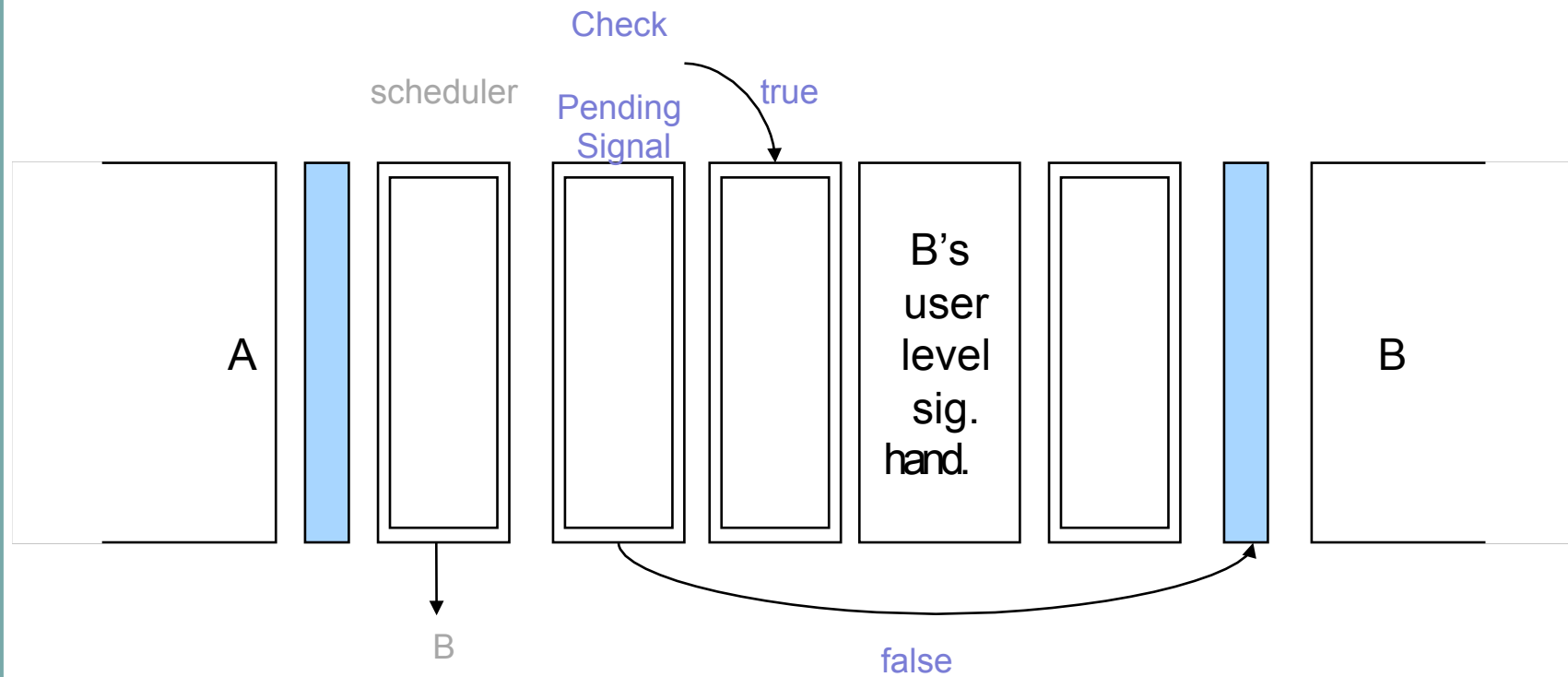
Scheduler: w/ signals



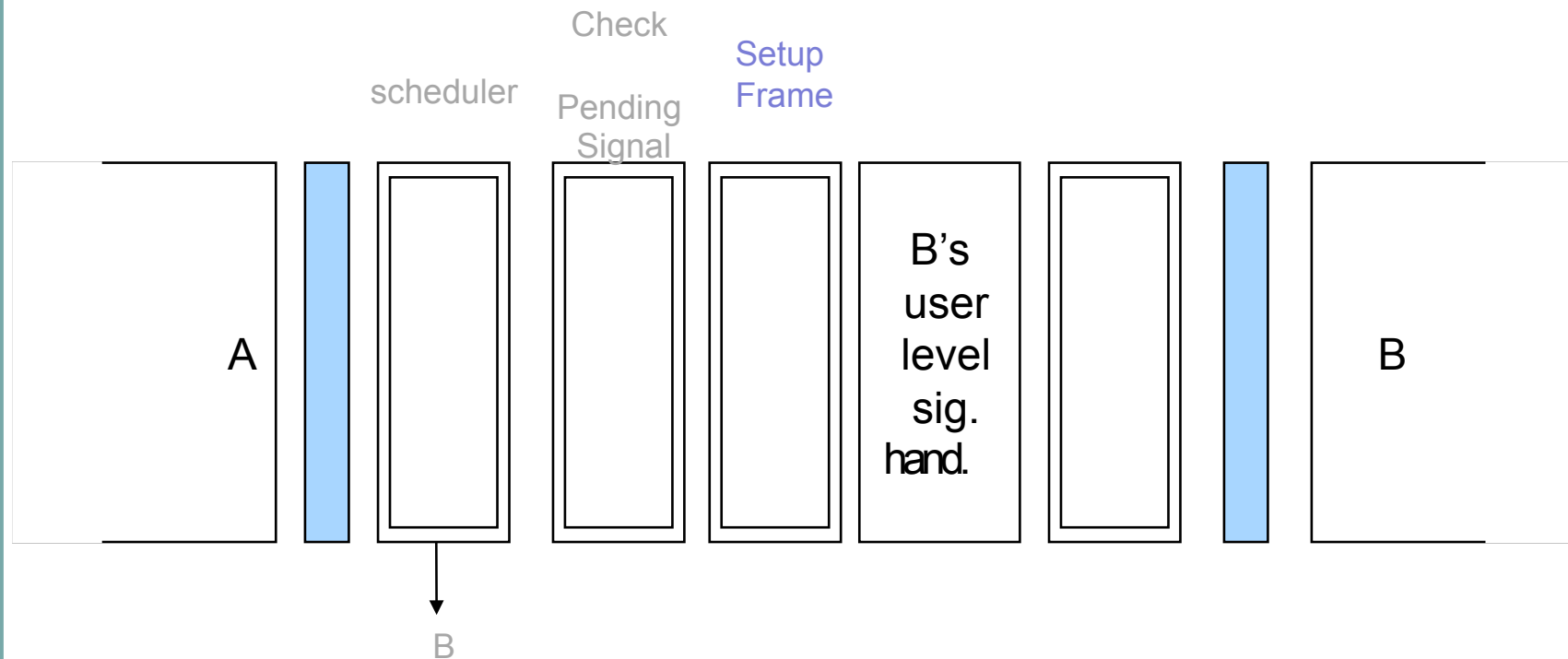
Check Pending Signal

- Boolean output
- Determines whether to proceed with signal handling

Scheduler: w/ signals



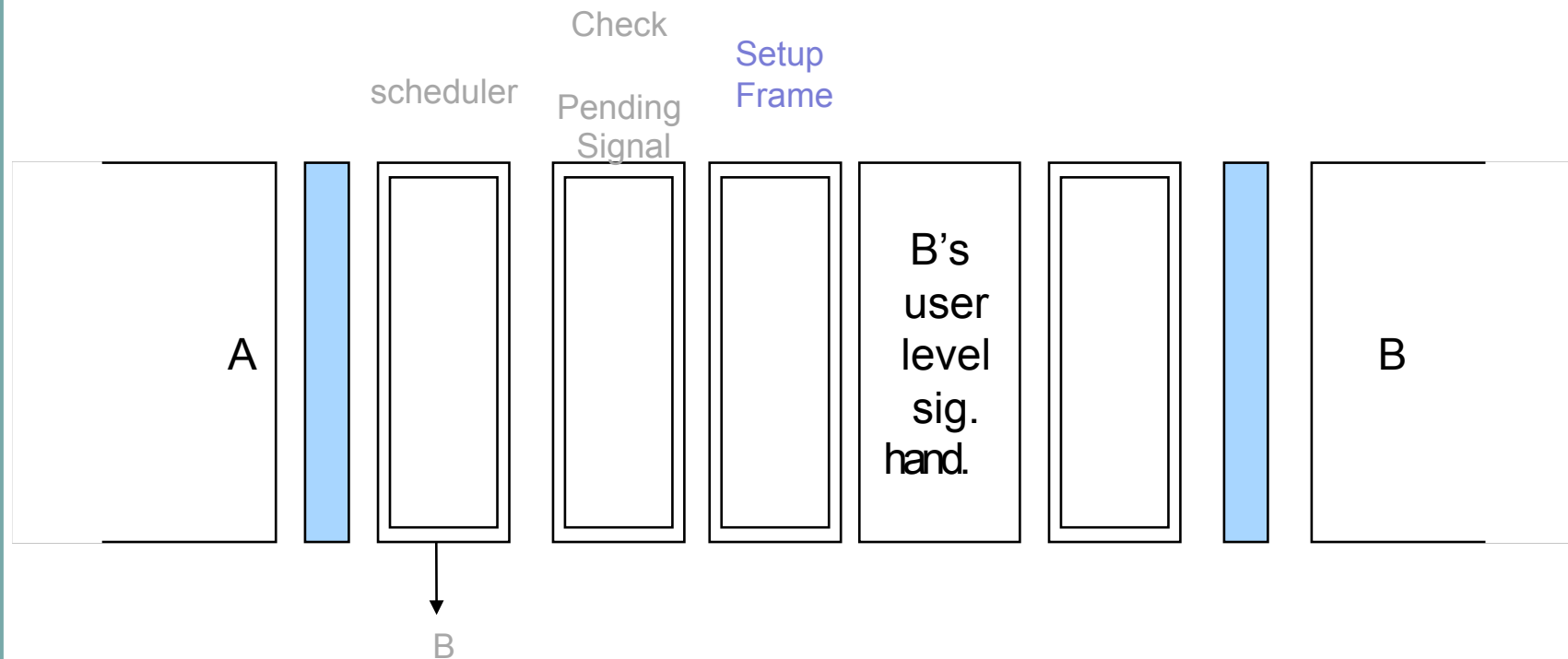
Scheduler: w/ signals



Setup Frame

- Sets up state to enable user-level handling code execution

Scheduler: w/ signals



Setup Frame

- Sets up state to enable user-level handling code execution
- How are functions called?

Function Calls

- Parameter of return address is stored on the stack so when finished
 - Pop off stack
 - Continue execution
- Setup Frame
 - Enables user stack to keep:
 - Interrupt_State Vector
 - Return address

Storing Return Address

- Want complete_handler to execute once user level handling done.
- Hack (well kind of)
 - Place address of return_signal as return address on stack
 - Now return_signal stored as function

Scheduler: w/ signals

