1. 修改 launch.json 文件如下所示

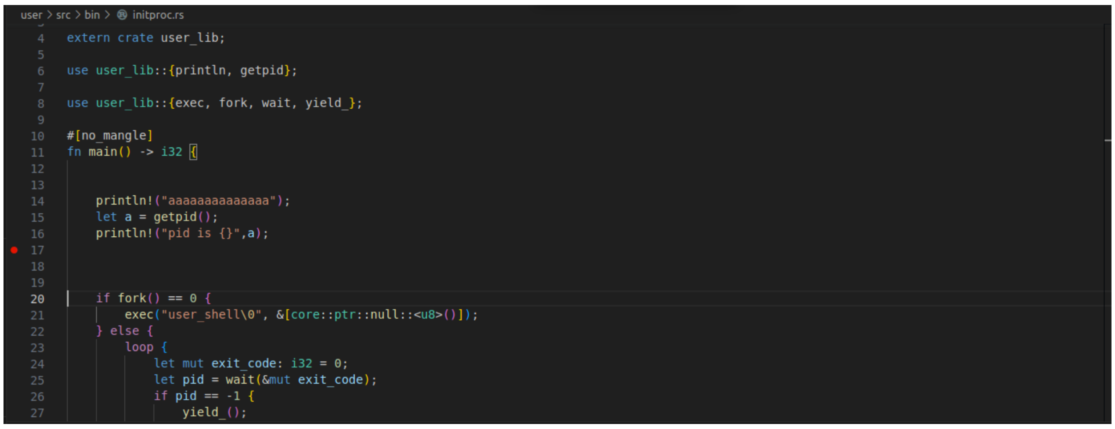
```
{
        "version": "0.2.0",
        "configurations": [
                {
                        "type": "gdb",
                        "request": "launch",
                        "name": "Attach to Qemu",
                        "executable":
"${userHome}/rCore-Tutorial-v3/os/target/riscv64gc-unknown-none-elf/release/os",
                        "target": ":1234",
                        "remote": true,
                        "cwd": "${workspaceRoot}",
                        "valuesFormatting": "parseText",
                        "gdbpath": "riscv64-unknown-elf-gdb",
                        "showDevDebugOutput":true,
                        "internalConsoleOptions": "openOnSessionStart",
                        "printCalls": true,
                        "stopAtConnect": true,
                        "qemuPath": "qemu-system-riscv64",
                        "qemuArgs": [
                                "-M",
                                "128m",
                                "-machine",
                                "virt",
                                "-bios",
                                "${userHome}/rCore-Tutorial-v3/bootloader/rustsbi-qemu.bin",
                                "-display",
                                "none",
                                "-device",

"loader,file=${userHome}/rCore-Tutorial-v3/os/target/riscv64gc-unknown-none-elf/release/os.bi
n,addr=0x80200000",
                                "-drive",

"file=${userHome}/rCore-Tutorial-v3/user/target/riscv64gc-unknown-none-elf/release/fs.img,if=
none,format=raw,id=x0",
                                "-device",
                                "virtio-blk-device,drive=x0",
                                "-device",
                                "virtio-gpu-device",
                                "-device",
                                "virtio-keyboard-device",
                                "-device",
                                "virtio-mouse-device",
                                "-serial",
                                "stdio",
                                "-s",
                                "-S"
                        ],
                                    "KERNEL_IN_BREAKPOINTS_LINE":65,
                                    "KERNEL_OUT_BREAKPOINTS_LINE":124,
                                    "GO_TO_KERNEL_LINE":30,
                },
        ]
}
```

其中"KERNEL_OUT_BREAKPOINTS_LINE":124,一栏中，124 可以更改为 125，126，127。

2. 目前用户态中仅支持在 initproc 的第 17 行打断点。

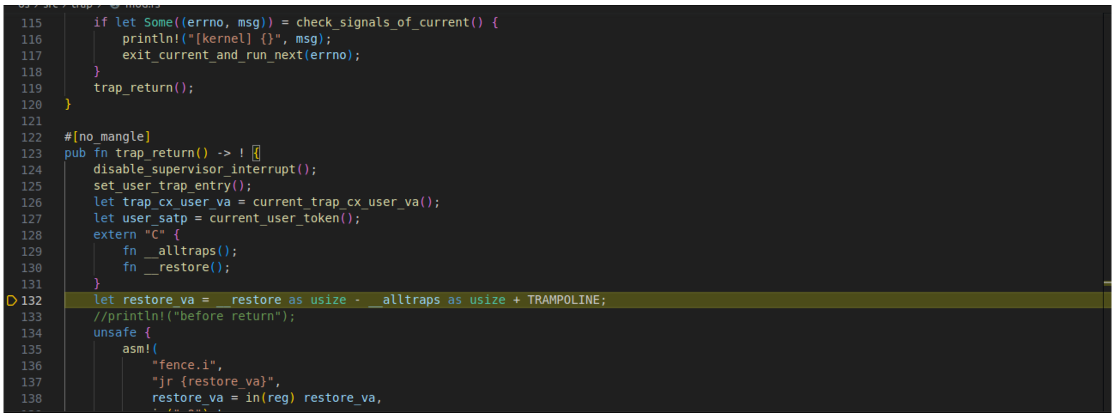


user > src > bin > initproc.rs

```
 4  extern crate user_lib;
 5
 6  use user_lib::{println, getpid};
 7
 8  use user_lib::{exec, fork, wait, yield_};
 9
10  #[no_mangle]
11  fn main() -> i32 {
12
13
14      println!("aaaaaaaaaaaaaa");
15      let a = getpid();
16      println!("pid is {}",a);
17
18
19
20      if fork() == 0 {
21          exec("user_shell\0", &[core::ptr::null::<u8>()]);
22      } else {
23          loop {
24              let mut exit_code: i32 = 0;
25              let pid = wait(&mut exit_code);
26              if pid == -1 {
27                  yield_();
```

错误样例：

1.若将进用户态断点打在 128~134 行，会一直卡在第 132 行不动



```
115     if let Some((errno, msg)) = check_signals_of_current() {
116         println!("[kernel] {}", msg);
117         exit_current_and_run_next(errno);
118     }
119     trap_return();
120 }
121 
122 #[no_mangle]
123 pub fn trap_return() -> ! {
124     disable_supervisor_interrupt();
125     set_user_trap_entry();
126     let trap_cx_user_va = current_trap_cx_user_va();
127     let user_satp = current_user_token();
128     extern "C" {
129         fn __alltraps();
130         fn __restore();
131     }
132     let restore_va = __restore as usize - __alltraps as usize + TRAMPOLINE;
133     //println!("before return");
134     unsafe {
135         asm!(
136             "fence.i",
137             "jr {restore_va}",
138             restore_va = in(reg) restore_va,
```

若打在其他非 124~135 行，则用户态断点失效

2.若 initproc 中断点不在第 17 行，则用户态断点失效。