LE17.2 SVC Handlers

Bookmark this page

LE17.2.1: Wait SVC Handler

0.0/1.0 point (ungraded)

The WAIT() supervisor call expects the address of a user's memory location in R0. The WAIT handler examines the value of that memory location and if it's positive, decrements the value in the memory location by 1 and resumes execution of the user's program at the instruction following the WAIT() SVC. If the memory location is not positive, the handler should arrange to re-execute the WAIT() SVC next time the user's program runs and then yield the remainder of the current time slice.

Your goal is to fill in the required code for Wait_h(), the kernel-mode handler for the WAIT SVC. You can use the kernel procedure GetUserLocn(addr) to get the values of a location in the user's memory and SetUserLocation(addr,v) to set the value of a memory location. The kernel subroutine Scheduler() is also available for your use. Select the correct choice of instructions to include in the **if** and **else** clauses of the Wait_h() stub shown below. Make sure to include all the required instructions. Assume that the selected instructions are executed in the order they appear.

```
Wait_h() {
  int locn = UserMState.Regs[0]; // user location to test
  int value = GetUserLocation(locn); // current value
  if (value > 0) {
    // if clause code

  } else {
    // else clause code

  }
}
```

if clause code (select all that apply):

- [ ] UserMState.Regs[XP] = UserMState.Regs[XP] - 4;
- [ ] SetUserLocation(locn,value)
- [ ] SetUserLocation(locn,value-1)
- [ ] GetUserLocation(locn)
- [ ] Scheduler();

else clause code (select all that apply):

- [ ] UserMState.Regs[XP] = UserMState.Regs[XP] - 4;
- [ ] SetUserLocation(locn,value)
- [ ] SetUserLocation(locn,value-1)
- [ ] GetUserLocation(locn)
- [ ] Scheduler();

Submit

Calculator

LE17.2.2: YieldN SVC Handler

0.0/1.0 point (ungraded)

The Yield() SVC can be used in user-mode programs on a time-sharing system to give up the remainder of their current time slice. The kernel implementation of the Yield simply calls the kernel Scheduler() routine to choose another process to execute. When the yielding process is next scheduled, user-mode execution resumes with the instruction following the Yield() SVC.

Complete the code for the handler for a new SVC, YieldN(), which expects a numeric value, N, in the user's R0 and behaves as if the user program had contained N consecutive Yield() SVCs. When execution resumes following the completion of YieldN(), R0 should contain 0. Select the correct choice of instructions to include in the **if** and **else** clauses of the YieldN_h() stub shown below. Make sure to include all the required instructions. Assume that the selected instructions are executed in the order they appear.

```
YieldN_h() {
  if (UserMState.Regs[0] > 0) {
    // if clause code

  } else {
    // else clause code

  }
}
```

if clause code (select all that apply):

- [ ] UserMState.Regs[0] = UserMState.Regs[0] - 1;
- [ ] UserMState.Regs[0] = UserMState.Regs[0] - N;
- [ ] UserMState.Regs[0] = 0;
- [ ] UserMState.Regs[XP] = UserMState.Regs[XP] - 4;
- [ ] UserMState.Regs[LP] = UserMState.Regs[LP] - 4;
- [ ] YieldN();
- [ ] Scheduler();

else clause code (select all that apply):

- [ ] UserMState.Regs[0] = UserMState.Regs[0] - 1;
- [ ] UserMState.Regs[0] = UserMState.Regs[0] - N;
- [ ] UserMState.Regs[0] = 0;
- [ ] UserMState.Regs[XP] = UserMState.Regs[XP] - 4;
- [ ] UserMState.Regs[LP] = UserMState.Regs[LP] - 4;
- [ ] YieldN();
- [ ] Scheduler();

Submit

Calculator

Discussion

Topic: 17. Virtualizing the Processor / LE17.2 SVC Handlers

Hide Discussion

Add a Post

Show all posts by recent activity

? How does wait SVC work? 3

What's the purpose of >...The WAIT handler examines the value of that memory location and if it's positive, decrements the value in t...

[Spoiler] LE17.2.2: YieldN SVC Handler, Zeroing? 3

Community TA

Previous

Next Up: Worked Examples
8 min + 1 activity

Calculator