Therac-25

1. Software problem #1 (our best guess)

A. Three threads:

−−Hand: sets the collimator/turntable position

−−Treat: sets a bunch of other parameters. Part of its job takes

eight seconds, during which time it’s ignoring everything else.

−−Vtkbp (keyboard handler): invoked when user types. It parses

the input, and writes to a two-byte shared variable, "MEOS" (mode/energy

offset)

−−"Treat" reads top byte, sets current and energy

−−"Hand" reads bottom byte, sets the collimator/turntable position

B. Pseudocode:

Vtkbp (gets and parses keyboard input):

data_completion_flag = 0

while (1) {
    wait_for_keyboard_activity();
    /* there was some keyboard activity; let’s check it */
    if (cursor_in_bottom_right) {
        parse_the_input();
        set the MEOS variable
        set data_completion_flag = 1;
        signal hand thread
        signal treat thread
    } else {
        /* operator still typing */
        data_completion_flag = 0;
    }
    yield();
}

Hand (sets the turntable position):

while (1) {
    wait until signalled
    read bottom byte of MEOS variable
    /* next line executes quickly */
    set turntable position
    yield();
}

Treat (sets a bunch of parameters and delivers treatment):

dataent() { /* this is a subroutine that was called */
    while (1) {
        wait until signalled
        read top byte of MEOS variable
        set_energy_and_current(); /* this takes eight seconds */
        if (data_completion_flag == 1)
            break;
        /*
         * now we leave the subroutine and progress to a state in
         * which the machine will accept a "beam on" command
         */
        return;
    }
}

2. Software problem #2 (simplified)

[Simplifying here and condensing to one thread of control; in

reality, the functions below are spread over two different threads,

but that is not actually the problem, despite what the paper

sometimes says. The problem appears to be given by the following

simplified description.]

class3 = 0;

while (1) {
    if (in field light position) {
        increment class3;
    }
    if (operator pressed set) {
        if (class3 != 0) {
            move turntable out of field light mode;
        }
        break;
    }
}

What’s the issue here? (Hint: class3 is only one byte.)
- Turntable: rotates the turntable
- Treat: sets magnets, sets energy, sets current

Keyboard Handler (VkHzp)

Turntable Thread (Hard)

Parameter setting/Treatment (Treat)

MEOS (mode energy offset)
3. Implementing threads

Per-thread state in thread control block:

```c
typedef struct tcb {
    unsigned long esp;     /* Stack pointer of thread */
    char *t_stack;     /* Bottom of thread’s stack */
    /* ... */
};
```

Machine-dependent thread-switch function:

```c```
void swtch(tcb *current, tcb *next);
```

Machine-dependent thread initialization function:

```c```
void thread_init (tcb *t, void (*fn) (void *), void *arg);
```

Implementation of swtch(current, next):

```c```
pushl %ebp; movl %esp, %ebp # Save frame pointer
pushl %ebx; pushl %esi; pushl %edi # Save callee-saved regs
movl 8(%ebp),%edx # %edx = current
movl 12(%ebp),%eax # %eax = next
movl %esp,(%edx) # %edx->esp = %esp
movl (%eax),%esp # %esp = %eax->esp
popl %edi; popl %esi; popl %ebx # Restore callee-saved regs
popl %ebp # Restore frame pointer
ret # Resume execution
```

[thanks to David Mazieres]