

## Re: Newbie question on C library system() function

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2007-05/msg01217.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2007-05/msg01217.html)

---

- *From:* [roberson@xxxxxxxxxxxxxxxxxxxxx](mailto:roberson@xxxxxxxxxxxxxxxxxxxxx) (Walter Roberson)
  - *Date:* Sun, 13 May 2007 19:40:32 +0000 (UTC)
- 

In article <1179083185.637754.170360@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>, philbo30 <masferfc@xxxxxxxx> wrote:

I need to send control data (rewind by 3 lines) to a kiosk printer via a 115200 tty and then print out information associated with a transaction. The problem is that my control command is executed via a bash shell system() call which is apparently much slower than my c data processing. The end result is a printer trying to back up and move forward at the same time. Unpretty.

Here' s the code for the system call:

```
char backup[32]="echo -ne \033K7 > /dev/ttyS0 \n"
int rewindsome(void)
{
    system(backup);
    system(backup);
    system(backup);
    return 0;
}
```

I need to make sure that the function above fully completes before the rest of my app executes. Ideas?

You probably don't need the \n in there.

You could do the same thing in C by using

```
#include <stdio.h>
int rewindsome(void) {
    FILE *printer = fopen("/dev/ttyS0", "a");
    if (!printer) {
```

Re: Newbie question on C library system() function

```
perror("Problem opening the printer");  
return 1;  
}  
fprintf( printer, "\033K7\033K7\033K7" );  
fclose( printer );  
return 0;  
}
```

However, this –probably– won't solve the problem. In my experience, your problem likely is not with system() being slow, but rather with the printer being slow and with there being no way to check with the printer to see whether it completed a command yet. You are sending the characters to the printer, but the characters are getting buffered in the printer memory and executed in the printer's own sweet time.

Lacking a way to check completion, the best you would be able to do would be to wait for a period of time, hoping the command completes (which would depend on what else the printer is doing); unfortunately, there is no way in standard C to wait for a given amount of time. Most operating systems provide ways outside of C to wait gracefully; chances are that your operating system supports a sleep() function for example.

—

Okay, buzzwords only. Two syllables, tops. — Laurie Anderson

.