SE350 W10 RTX Project Part One

Due at 11:59pm on January 23rd 2010

Submit code to Course Book System

a) Program the second serial port (i.e. UART1) on the MCF5307 board. Put the program source code under a directory/folder named “uart1”.

The program echoes the user input on the key board to UART1 until the user hits the “q” key. When the “Return” or “Enter” key is pressed, a new line should start on the second serial port.

b) Program the first timer (i.e. Timer0) and UART1 on the MCF5307 board. Put the program source code under a directory/folder named “timer0”.

When the program starts, it displays “Timer started” on the UART1 terminal screen. Then every 5 seconds, it displays “5 seconds passed” on UART1.

c) Write process_switch(int pid) function by using software interrupt. Put the program source code under a directory/folder named “process_switch”.

The caller of process_switch(int pid) is switched out of the processor and let the process whose pid is specified in the parameter field “pid” to have the processor and run.
You are provided with partially finished code where there are three processes proc1 and proc2. The proc1 calls process_switch() to switch to proc2 every time it prints 5 uppercase alphabets on janusROM terminal. The porc2 calls process_switch() to switch to proc1 every time it prints 5 lowercase alphabets on janusROM terminal. The process which the process_switch() switches to should continue to run from the place where it was last switched out of the processor. For a process which has never run, a call of process_switch() to switch to it will make the process to run from its entry point. You are free to create your own assembly file, data structure, c sub-routines to finish the process_switch() routine. The partially finished code provided to you is in p1_c.zip file on the lab web site. An example of expected output on janusROM terminal of this program is:

```
ABCDE
abcde
FGHIJ
fghij
KLMNO
klmno
……
```

Hint: when an interrupt happens (for example an software interrupt), an exception frame is automatically generated by the processor. The a7, sr and pc are pushed onto the exception frame. What is the assembly instruction to restore the exception frame?