SFWR ENG Assignment 3: Digital Control Implementation

Due: 1600 Monday December 2, 2002

For this assignment you will hand in one copy of the assignment for each lab group.

1. **Control Systems Implementation I** (20 marks)

   a) (5 marks) The code fragment shown in Figure 1 is taken from the RTAI real-time Linux source for the file `/usr/src/rtai/include/asm-i386/rtai_sched.h`.

```
#define __STR(x) #x
#define STR(x) __STR(x)

#define rt_switch_to(tsk) \
        __asm__ __volatile__( \
        "pushl %%eax\n\t" \
        "pushl %%ebp\n\t" \
        "pushl %%edi\n\t" \
        "pushl %%esi\n\t" \
        "pushl %%edx\n\t" \
        "pushl %%ecx\n\t" \
        "pushl %%ebx\n\t" \
        "movl "SYMBOL_NAME_STR(rt_current)", %%edx\n\t" \
        "pushl $1f\n\t" \
        "movl %%esp, (%%edx)\n\t" \
        "movl (%%ecx), %%esp\n\t" \
        "movl %%ecx, "SYMBOL_NAME_STR(rt_current)"\n\t" \
        "ret\n\t" \
"1:     popl %%ebx\n\t \
        popl %%ecx\n\t \
        popl %%edx\n\t \
        popl %%esi\n\t \
        popl %%edi\n\t \
        popl %%ebp\n\t \
        popl %%eax\n\t" \
        : \
        : "c" (tsk));
```

Figure 1: Excerpt from `rtai_sched.h`

   i) What is does the `rt_switch_to(tsk)` macro do?

   ii) Typically real-time schedulers make use of macros and inline functions. Why?

   b) (3 marks) The RTAI-Linux system call `rt_task_init( . . . )` is used to create a real-time task. One of its arguments is defined as `int stack_size`, the size of the task's stack. How does the real-time kernel use the stack space of each real-time task when running multiple tasks?

   c) (6 marks) Suppose a system has three levels of interrupts with level 1 being the highest priority and level 3 being the lowest priority. Use a diagram to explain what happens to the system at each step when the following sequence of events occurs.

(a) At time $t = 0\mu s$ the processors is initially executing the low priority interruptable process Task 1.

(b) At time $t = 5\mu s$ a level 2 interrupt occurs.

(c) At time $t = 8\mu s$ a level 1 interrupt occurs.

(d) At time $t = 9\mu s$ a level 3 interrupt occurs.

(e) No further interrupts occur.

Assume that a context switch takes negligible time and that the interrupt service routines (ISRs) and process require the following amount of execution time:

Process	CPU execution time (μs)
ISR 1	3
ISR 2	4
ISR 3	2
Task 1	10

d) (6 marks) What type of H/W & Software would you recommend for the proposed systems shown in Table 1?

Match each application in the column on the left with a method of implementation from the column on the right as has been done for System A, the programmable theromstat. The reasons for choices are to be given in Table 2. Complete the tables by making similar well thought out choices for the remaining 6 systems and record a brief explanation for each of your choices.

2. **Control Systems Implementation II** (15 marks)

Your hard work in 4A03 has paid off and landed you a consulting job. You have to implement an embedded digital position control system for a robot that has a built in DSP (digital signal processing) chip. The DSP chip does not have built in A/D converters but it does have 16 digital (i.e. 0/1) inputs.

a) (3 marks) What type of position sensor might you recommend and why? Justify any additional external hardware that might be used.

b) (3 marks) The sensor for the system has recently been upgraded so that it now provides 32-bit digital position readings by taking the current position and truncating it to a 32-bit quantity. What is the maximum quantization error associate with this sensor?

c) (5 marks) While the DSP chip only has 16 digital inputs, it also has an additional digital output that is currently unused in the system design. Sketch a simple circuit that would allow you to use this digital output with minimal additional external hardware to interface with the new 32-bit sensor to obtain more accurate measurements. (HINT: Remember 2D04? Think MUXes - that's MUltipleXers).

d) (2 marks) For this scheme to work, what assumption must the robot system satisfy relative to the digital controller?

e) (2 marks) What are two possible sources of quantization error that could occur in any computations that take place in the DSP?

3. **Scheduling** (25 marks)

a) (4 marks) The standard Linux fix priority scheduler can be replaced with an alternative scheduler. In Figure 2 and Figure 3 we illustrate the scheduling results two different alternative RT-Linux schedulers studied in class.

	System		Implementation
A	Programmable Thermostat		Single loop program running directly on widely used 486 controller board with EPROM and error correcting memory
B	Missile guidance control system		16-bit microcontroller with integrated A/D, RAM & ROM running a C program
C	Distributed networked process control		C++ application on a Linux PC
D	Robotic control in a University Lab		Custom harware with highend DSP
E	Garage door opener		PCs with A/D cards running RT-Linux
F	Video Conferencing System		x86 industrial PC with integrated network, I/O & RTOS
G	Reactor shutdown system	**A**	8-bit microcontroller programmed in assembler

Table 1: System implementation choice table to be filled in for question 4(c)

System	Reasons for choice of implementation
A	- its a simple task requiring little processor power - high volume product - very cost sensitive, code in assembler to fit program in onboard ROM
B	
C	
D	
E	
F	
G	

Table 2: Reasons for choices in question 4(c)

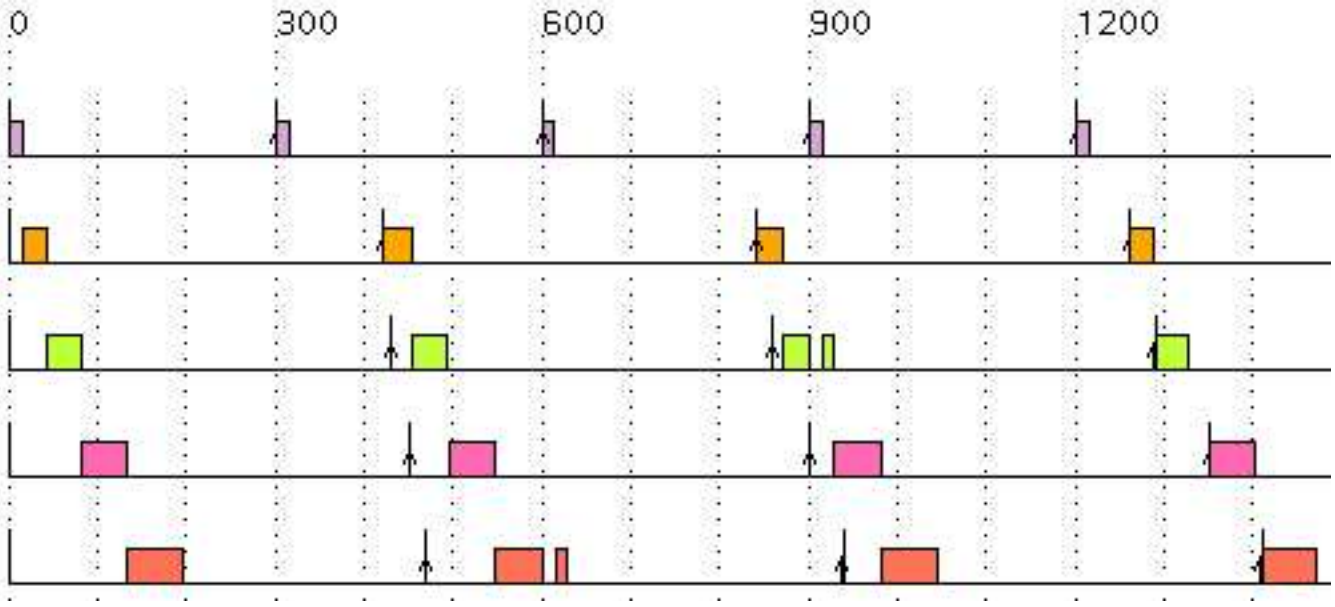

Figure 2: Scheduler 1

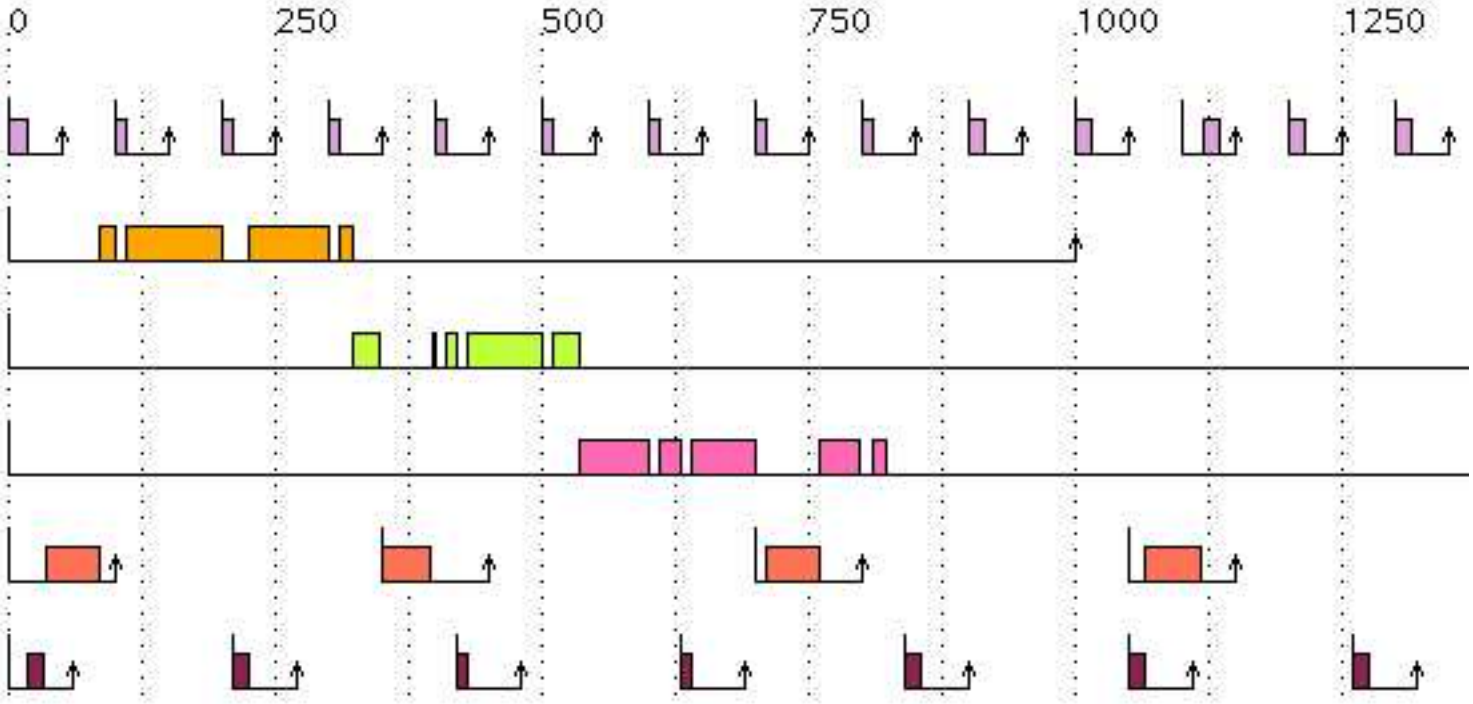

Figure 3: Scheduler 2

i) What scheduler do you believe has been used in Figure 2? Why do you believe this?

ii) What scheduler do you believe has been used in Figure 3? Why do you believe this?

b) (10 marks) Given the following 5 independent tasks with the specified periods, determine if it is possible to schedule the tasks if:

Task	Period	Max. Execution Time
A	30	5
B	4	1
C	10	1.5
D	8	1
E	20	1

i) The tasks are NOT pre-emptable? Justify your answer.

ii) All tasks except Task B are preemptable? Justify your answer.

iii) Use a well known scheduling algorithm to assign priorities between 1-5 to the tasks so that the tasks will be properly executed by the RT-Linux fixed priority scheduler with no interruptions occurring to Task B. What scheduling policy did you use to choose your task priorities?

c) (4 marks) Assume now that some of the tasks share a resource. Explain how "Priority Inversion", where a lower priority task prevents a higher priority task from runing, could arise.

d) (4 marks) Assume a fixed priority scheduler is used. For some well thought out reason, the priorities are assigned to the tasks in alphabetical order (i.e., Task A is assigned priority 1, Task B is assigned priority 2, ... Task E is assigned priority 5).

Assume that two of the tasks share a semaphore but the remaining tasks are all independent. Which tasks could share the semaphore without the risk of priority inversion occurring?

e) (3 marks) Explain the differences between Fixed priority, RM and EDF scheduling. (NOTE: This is only for 3 marks, so keep it short!)

BONUS: In the lab create and briefly document a "Sprinker" module that makes the servo track a waveform mimicing a lawn sprinkler shown in Figure 4.

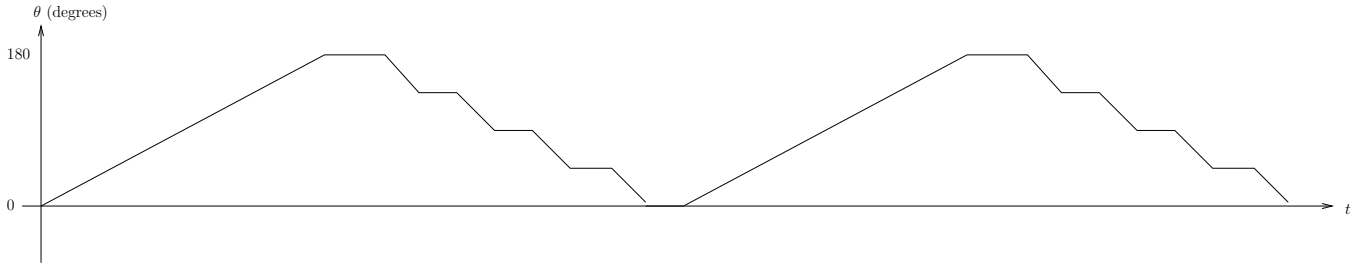

Figure 4: Sprinkler reference siginal for Bonus Question