EE8205: Embedded Computer System

Multitasking and Real-time Operating System -- Problem Set

Problem 1: What is the difference between turnaround time and response time.

Problem 2: What is the difference between Nonpreemptive and Preemptive scheduling..

Problem 3: Explain how a preemptive priority scheduling system would work.

Problem 4: Consider the following C program for execution on a Linux system.

```
/* #define NOSYSCALL */
#ifdef NOSYSCALL
int getpid() { return 55; }
#endif
int main( int argc, char * argv[] ) {
    int i, a, limit = atoi(argv[1]);
    for( i = 0; i < limit; ++i ) a = getpid();
}
```

The system call ***getpid*** performs almost no processing. It looks up your pid and returns it. All the time it takes is system call overhead that is present in a system call. Likewise, the procedure call ***getpid*** does nothing but return a value. All the time it takes is procedure call overhead that is present in every procedure call.
Run this program for 2-3 million iterations and see how long it takes. Then un-comment out the #define and run it for 30-40 million iterations and see how long it takes. What do you conclude about the relative speed of a system call and a procedure call?

Problem 5: Suppose we run each of the following scheduling algorithms in a system that is very heavily overloaded. Describe how each of these algorithms act in the face of overloading. Discuss how this overloading affects the average waiting time of short jobs, medium jobs and long jobs (if they are affected differently). That is, discuss how the average waiting time changes (for short, medium and long jobs) when going from a lightly loaded system to a heavily loaded system. Be sure to discuss the overhead of extra context switches caused by the scheduling algorithm (if any).

Problem 6: Suppose a new process in s system arrives at an average of four processes per minute and each such process requires an average of 12 seconds of service time. Estimate the fraction of time the CPU is busy in a single processor system.

Problem 7: Consider the following processes are to be scheduled using, FCFS, Round Robin with time quantum 1 and 4.

	A	**B**	**C**	**D**	**E**
$\mathbf{T_a}$	0	1	3	9	12
$\mathbf{T_s}$	3	5	2	5	5

Problem 8: Assume you have the following processes to execute with one CPU.

Process	Arrival Time	Execution Time
0	0	75
1	10	40
2	10	25
3	80	20
4	85	45

Suppose a system uses RR scheduling with a time quantum of 15 and context switch time is five time units with RR scheduling.
Create a Gantt chart illustrating the execution of these processes.
What is the turn around time for process 3.

Problem 9: Consider two jobs, A and B, in a deadline scheduling system. The deadline for A is before the deadline for B. Explain why we should run A before B, that is, show that if running A then B fails to meet some deadline then running B before A will also fail to meet some deadline.

Problem 10: Consider a set of 5 aperiodic tasks with their execution profiles given below. Develop the scheduling diagram of these processes employing EDF and FCFS.

Process	Arrival Time	Execution Time	Starting Deadline
A	10	20	100
B	20	20	30
C	40	20	60
D	50	20	80
E	60	20	70

Problem 11:

a. Consider a periodic task set with the following independent tasks.
Task P1: C1 = 20 T1 = 100
Task P2: C2 = 30 T2 = 145

b.
Now add the following task to the set
Task P3: C3 = 68 T3 = 150
Verify the utilization-based analysis for all the three tasks.

c.
Suppose that the first instance of the preceding three tasks arrives at time t = 0.
Assume that the first deadline for each task is the following:
D1 = 100; D2 = 145; D3 = 150;
(i) Using Rate Monotonic Scheduling, will all three deadlines be met?
(ii) What about deadlines for future repetitions of each task?

Problem 12:
Consider three processes P, Q and S. P has a period of 100msec in which it requires 30msecs of processing. The corresponding values for Q and S are (6, 1) and (25, 5) respectively. Assume that P is the most important process in the system, followed by Q and then S.

(1) What is the behavior of the scheduler if priority is based on importance?
(2) What is the process utilization of P, Q and S.
(3) How should the process be scheduled so that all deadlines are met.
(4) Illustrate one of the schemes that allows these processes to be scheduled.

Problem 13:
Add a fourth process R, to the set of processes given in Problem 12. Failure of this process will not lead to safety being undermined. R has a period of 50msecs, but has a processing requirement that is data dependent and varies from 5-to-25 msec. Discuss how this process should be integrated with P, Q and S.

Problem 14:
Is the process set of Table below schedulable using the simple utilization based test? Is the process set schedulable using the response time analysis.

Process	Period	Execution Time
a	50	10
b	40	10
c	30	9

Problem 15:
The process set shown in Table below is not schedulable using the utilization criterion because process ***a*** must be given the top priority due to its criticality. How can the process set be transformed so that it is schedulable. The computations represented by ***a*** must still be given top priority.

Process	Period	Execution Time	Criticality
a	60	10	HIGH
b	10	3	LOW
c	8	2	LOW

Problem 16:
In a safety-critical real-time system, a collection of processes can be used to monitor key environment events. Typically, there will be a deadline defined between the event occurring and some output (which is in response to the event) being produced. Describe how periodic processes can be used to monitor such events.

Problem 17:
How can the process set shown in the Table below be optimally scheduled (using fixed priority scheduling)? Is this task set schedulable?

Process	T	C	B	D
a	8	4	2	8
b	10	2	2	5
c	30	5	2	30

Problem 18:

Real-time system designers wish to run a mixture of safety-critical, mission-critical and non-critical periodic and sporadic tasks on the same processor. They are using preemptive priority-based scheduling and have used the response-time analysis equation to predict that all tasks meet their deadlines.

Give reasons why the system might nevertheless fail to meet its deadlines at run-time.

What enhancement could be provided to the runtime support system to help eliminate the problem?