Real-Time Systems — EDA222/DIT161

Final Exam, March 10, 2014 at 08:30 – 12:30 in the V building

---

Examiner:
Professor Jan Jonsson

Questions:
Risat Pathan, phone: 076 214 8509

Aids permitted during the exam:
J. Nordlander, *Programming with the TinyTimber kernel*
Chalmers-approved calculator

Content:
The written exam consists of 8 pages (including cover), containing 7 problems worth a total of 60 points.

Grading policy:
24–35 ⇒ grade 3
36–47 ⇒ grade 4
48–60 ⇒ grade 5

Solution:
Posted on the course home page on Tuesday, March 11, 2014 at 09:00.

Results:
Posted on the course home page on Monday, March 31, 2014 at 09:00.

Inspection:
Room 4128, Rännvägen 6 B, on Monday, March 31, 2014 at 10:00–11:00. Inspection at another occasion could be arranged by contacting the course examiner.

Language:
Your solutions should be written in English.

IMPORTANT ISSUES

1. Use separate sheets for each answered problem, and mark each sheet with the problem number.
2. Justify all answers. Lack of justification can lead to loss of credit even if the answer might be correct.
3. Explain all calculations thoroughly. If justification and method is correct then simple calculation mistakes do not necessarily lead to loss of credit.
4. If some assumptions in a problem are missing or you consider that the made assumptions are unclear, then please state explicitly which assumptions you make in order to find a solution.
5. Write clearly! If I cannot read your solution, I will assume that it is wrong.

Good Luck!

PROBLEM 1

State whether the following propositions are TRUE or FALSE. Each correct statement will give 0.5 points; each erroneous statement will give -0.5 points; an omitted statement gives 0 points. Although a motivation for a correct answer is not required, a convincing one gives another 0.5 points, while an erroneous/weak one gives another -0.5 points. **Quality guarantee**: The total result for this problem cannot be less than 0 points. (6 points)

a) A *necessary* schedulability condition for periodic tasks on a uniprocessor platform is that the total utilization of the tasks is not larger than 1.

b) Message *queuing delay* is bounded in network communication using controller area network (CAN).

c) An *exact* schedulability test for global fixed-priority scheduling of periodic tasks is already known.

d) Hard real-time guarantee cannot be provided for systems with *sporadic tasks* since the inter-arrival time of consecutive instances of the tasks is not strictly periodic.

e) Deadlock occurs only in *non-preemptive scheduling* since a task that holds a shared resource cannot be preempted.

f) Disabling processor interrupts is a machine-level technique that is generally used to implement *mutual exclusion* in multiprocessor systems.

PROBLEM 2

A fundamental prerequisite for correct concurrent execution of multiple tasks with shared resources is that the run-time system can guarantee *mutual exclusion* and *synchronization*.

a) Describe how mutual exclusion and synchronization are achieved among the object methods in TinyTimber? (4 points)

b) State one similarity and one difference between protected object and monitor. (2 points)

PROBLEM 3

Most scheduling analysis techniques assume the worst-case execution time (WCET) to model the computational demand of a real-time task. One of the earliest methods for WCET analysis was presented by Shaw in the end of the 1980s. Assume that the function `main` (see in next page) is used as part of a real-time program and that the function, when called, is allowed to take at most 185 μs to execute where.

- Each declaration statement costs 1 μs to execute.
- Each assignment statement costs 1 μs to execute. Assume that the assignment of initial values to elements of arrays also costs 1 μs regardless of the number of elements.
- A function call costs 2 μs plus WCET for the function in question.
- Each compare statement costs 2 μs.
- Each addition and subtraction operation costs 3 μs.
- Each multiplication and division operation costs 4 μs.
- Each return statement costs 2 μs.

- Each modulo operation costs 5 μs.
- The function `abs()` is predefined and costs 5 μs. This function computes and returns the absolute value of the number given as its parameter. For example, `abs(-5)` returns 5. Assume that the overhead to call function `abs()` is already included in its WCET estimation.
- Reading a value of an array element costs 0 μs.
- All other language constructs can be assumed to take 0 μs to execute.

```
int main(){

    char Flag;
    int result;
    int P;
    int Q;
    int find;
    int count;
    int data [6] = {1, 2, 3, 4, 5, 6};
    count = 6;
    P = -16;
    Q = 12;
    Flag = 'F';

    if((abs(P) % Q) != 0)
        find = FuncA(abs(P), Q);
    else
        find = Q;

    result = FuncC(data, find, 0, count-1);

    if (result == -1)
        return -1;

    else if (result <= 16)
    {
        Flag = 'T';
        return 1;
    }

    else
    {
        Flag = 'T';
        return 2;
    }
}


int FuncA(int x, int y){
    if (y == 0)
        return x;
    else
        return FuncA(y, x % y);
}
```

```
int FuncB(int y){
    int z;
    z = 2;
    if(y == 0)
        return 1;

   while(y > 1){
        z = z * z;
        y = y - 1;
                }
    return z;
}


int FuncC(int data[], int x, int y, int z){
    int start;
    int end;
    int mid;
    start = y;
    end = z;
    mid = start + (end - start)/2;
    if (start > end)
        return -1;
    else if (data[mid] == x)
        return FuncB(data[mid]);
    else if (data[mid] > x)
        return FuncC(data, x, start, mid-1);
    else
        return FuncC(data, x, mid+1, end);
}
```

a) Drive WCET for function `main` by using Shaws method and check whether the functions deadline will be met or not. (8 points)

b) Identify ***all*** the false paths in the program given in subproblem a). (2 points)

PROBLEM 4

Consider a real-time system with a sensor and a printer. The system is implemented using independent periodic tasks `T1` and `T2`. Each invocation of periodic task `T1` calls another job `J1`. Task `T1` reads sensor measurements and job `J1` processes them. Each invocation of periodic task `T2` buffers the processed sensor data during first 10ms and prints the buffered data during next 30ms. A timing specification for the tasks and job (collectively called processes) is given in the table below.

Process	Offset	Period	Deadline	Execution Time
`T1`	0	60	10	10
`J1`	10	–	20	20
`T2`	30	60	infinite	40

All numbers in milliseconds (ms)

On a separate sheet at the end of this exam paper you find a scheduling diagram template and a C-code template. Write your solutions of subproblems a) and b) directly on that sheet and hand it in for grading together with the rest of your solutions.

a) Draw the TinyTimber schedule for the processes from time 0ms to 600ms using the diagram template given at the end of this exam paper. The sensor value is incremented by `1` in every 50ms. Assume that at time instant 0, the sensor value is 0. Show the sensor values printed by task `T2`. (4 points)

b) Implement the processes in C code using the template at the end of this exam paper. (4 points)

PROBLEM 5

Consider a uniprocessor real-time system with three periodic tasks and a run-time system that employs static scheduling using a time table. The table below shows O_i (offset), C_i (WCET), D_i (deadline) and T_i (period) for the three tasks.

	O_i	C_i	D_i	T_i
τ_1	1	1	3	3
τ_2	0	1	3	4
τ_3	2	?	6	6

a) Find the *largest* value of C_3 by simulating the deadline-monotonic (DM) schedule to construct a time table for the execution of the three tasks such that all the tasks meet their deadlines. Assume that the tasks are allowed to preempt each other. Your solution should clearly indicate the start and stop times for each task. In addition, the total length of your time table (in time units) should be given. (5 points)

b) State one advantage and one disadvantage of static scheduling over dynamic scheduling. (2 points)

c) State two disadvantages of preemptive scheduling over non-preemptive scheduling. (2 points)

PROBLEM 6

Consider a real-time system with periodic tasks and a run-time system that employs scheduling on uniprocessor. All tasks arrive the first time at time 0.

a) Do you agree that if a periodic task set is schedulable (i.e., meet all the deadlines) using Deadline-Monotonic (DM) scheduling algorithm where the relative deadline of each task is equal to its period, then the task set is also schedulable (i.e., meet all the deadlines) using Earliest-Deadline First (EDF) scheduling algorithm? Why or why not? (2 points)

b) The table below shows C_i (WCET), D_i (deadline) and T_i (period) for the three periodic tasks. Can you guarantee that the task set is schedulable using Earliest-Deadline-First (EDF) scheduling algorithm for some C_3 where C_3 is an integer and $5 \leq C_3 \leq 7$? Why or why not? Show your calculation. (6 points)

	C_i	D_i	T_i
τ_1	2	5	5
τ_2	3	6	10
τ_3	?	$2C_3$	20

c) Consider a task set scheduled using uniprocessor deadline-monotonic (DM) scheduling algorithm where the immediate ceiling priority protocol (ICPP) is used for handling shared resources. State the steps to compute the blocking factor (B_i) of a given task τ_i. (3 points)

PROBLEM 7

There are two approaches for scheduling tasks on multiprocessor platform: the *partitioned* approach and the *global* approach. The table below shows C_i (WCET) and T_i (period) for six periodic tasks to be scheduled on $m = 3$ processors. The relative deadline of each periodic task is equal to its period.

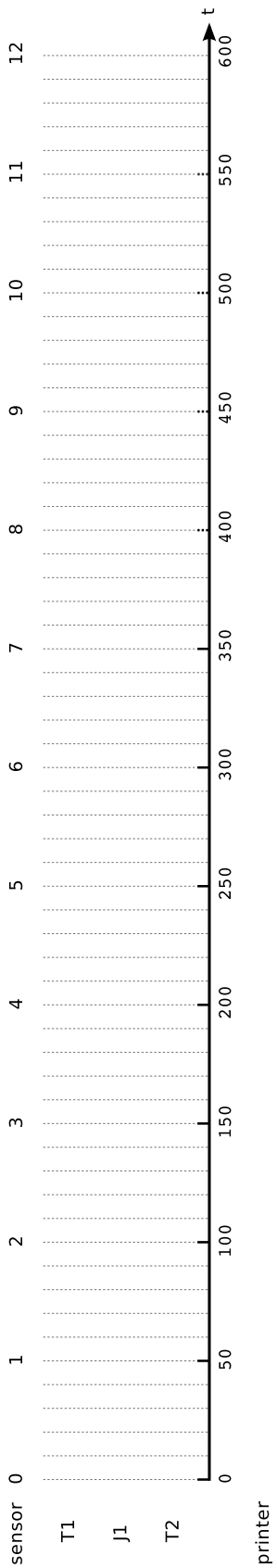
	C_i	T_i
τ_1	2	10
τ_2	10	25
τ_3	12	30
τ_4	5	10
τ_5	8	20
τ_6	7	100

a) The task set is schedulable using rate-monotonic first-fit (RMFF) partitioned scheduling algorithm. Show how the task set is partitioned on $m = 3$ processors so that all the deadlines are met using RMFF scheduling? (3 points)

b) Now consider that task τ_2 is removed from the task set and another task τ_7 with $C_7 = 3$ is added to the system. The task set now again has total six tasks. Determine the smallest possible period T_7 of task τ_7 so that all the six tasks are RMFF schedulable on $m = 3$ processors. Show your calculation. (4 points)

c) Describe how the RM-US priority-assignment policy avoids Dhall's effect. (3 points)

Scheduling Diagram Template Problem 4 (a)
(Submit this page with rest of the solutions)



Template Code for Problem 4 (b)
(Submit this page with rest of the solutions)

```
#include "TinyTimber.h"

typedef struct{
    Object super;
    char *id;
} RTprocess;

Object app = initObject();
RTprocess rtp1 = {initObject(), "T1"};
RTprocess rtp2 = {initObject(), "J1"};
RTprocess rtp3 = {initObject(), "T2"};


void exec1(RTprocess *self, int u) {



    work1(); // executes for 10ms




}

void exec2(RTprocess *self, int u) {


    work2(); // executes for 20ms


}

void exec3(RTprocess *self, int u) {



    work3(); // executes for 40ms




}


void kickoff(RTprocess *self, int u) {







}

main() {
     return TINYTIMBER(&app, kickoff, 0);
}
```