Real-Time Control Systems

Exam 2008-03-14

8:30 – 12:30: Halls at “Maskin”

Course code: SSY190

Teacher: Knut Åkesson, phone 0701-749525

The teacher will visit examination halls twice to answer questions. This will be done approximately one hour after the examination started and one hour before it ends.

The exam comprises 30 credits. For the grades 3, 4, and 5, is respectively required 15, 20 and 25 credits.

Solutions and answers should be complete, written in English and be unambiguously and well motivated. In the case of ambiguously formulated exam questions, the suggested solution with possible assumptions must be motivated. The examiner retains the right to accept or decline the rationality of assumptions and motivations.

Solutions will be announced on the course web-page on the first week-day after the exam date. *Exam results* will be announced on the department notice board on the latest 2008-03-28 at 12:30. *The results* are open for review 2008-03-28, 12:30-13:30 at the department.

Allowed aids:

- An A4 sheet with handwritten notes on both pages. The name of the student must be on the sheet. You should hand-in your notes together with your solutions.
- A pocket calculator with erased memory.
- Dictionary (paper and electronic) between English and the students native language.

1

a) What is priority inversion? Explain by giving a simple example.

(1p)

b) Explain shortly how the priority inheritance protocol works.

(1p)

c) Explain the OSI reference model of computer communication; describe its motivation and purpose.

(1p)

d) Explain the CSMA/CD medium access protocol.

(1p)

2

Determine whether each of the following statements are **true**, **false**, or if **more information is needed** in order to determine if the statement is true or false. A short motivation is required.

a) A given task set with CPU utilization below 100% is schedulable using EDF scheduling.

(1p)

b) In a real-time operating system it is desirable to wake up the process that has been waiting for the longest period for a semaphore when another process executes signal() on the same semaphore.

(1p)

3

A filter with the following transfer function

$$U(s) = \frac{100}{(s+5)(s+20)} Y(s)$$

will be implemented in a computer with the sampling interval h=150 ms. Two approximations of the continuous time design were obtained using the forward and backward Euler methods.

a) Determine which of the implementations i) and ii) below that corresponds to the backward Euler method.

i) $u(k) = -1.75u(k-1) + 0.5u(k-2) + 2.25y(k-2)$

ii) $u(k) = \frac{1}{7}(5.75u(k-1) - u(k-2) + 2.25y(k))$

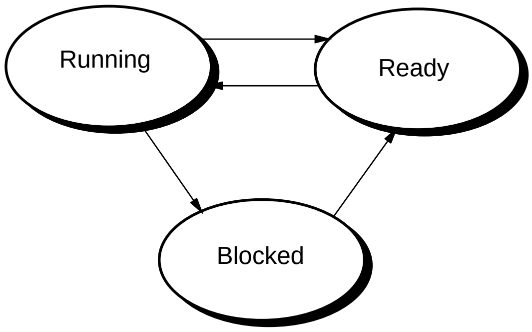
(2p)

b) Describe one disadvantage with the forward Euler method.

(1p)

4

Each process in a real-time operating system can be in the following three states.



a) Explain shortly what characterizes each of the three states?

(1p)

b) Explain for each of the four transitions, in the figure above, what could cause the process to switch state?

(2p)

5

A system consists of three processes, P1, P2, and P3. Each process executes the following sequences of semaphore operations. The system has five mutex semaphores A, B, C, D, and E. All semaphores are initialized to 1. Intermediate code that depends on the semaphore is represented by the function call "useXY" where "XY" denotes that semaphores X and Y must be taken when the code is executed.

P1	P2	P3
wait(A)	wait(D)	wait(C)
wait(C)	wait(E)	wait(D)
useAC()	useDE()	wait(B)
signal(C)	signal(E)	useBCD()
signal(A)	signal(D)	signal(B)
...	...	signal(D)
wait(E)	wait(B)	signal(C)
wait(A)	wait(A)	
useAE()	useAB()	
signal(A)	signal(A)	
signal(E)	signal(B)	

a) The system above may deadlock, explain why.

(2p)

b) Suggest a modification to the system such that the risk of deadlock is avoided without jeopardizing the synchronization needs of the system, i.e., when the function useXY() is called then corresponding semaphores must be taken.

(1p)

6

The following code describes an implementation of a PID controller that is not quite optimal.

```
y := AIIn(ychan)
e := yref-y
ad := Td/(Td-N*h)
bd := -K*N*ad
D := ad*D + bd*(y-yold)
v := K*(beta*yref-y) + I + D
yold := y
if mode = auto then u := sat(v,umax,umin)
else u := sat(uman,umax,umin)
I := I + (K*h/Ti)*e + (h/Tr)*(u-v);
DAOut(u,uchan)
if increment then uinc := 1
elseif decrement then
uinc := -1
else
uinc := 0
uman := uman + (h/Tm)*uinc +(h/Tr)*(u-uman)
```

The code is not time optimal. Explain how the code could be rewritten to minimize the computational delay before the control signal is sent to the physical process.

(3p)

7

In a real-time control system there are three threads, one control thread, one thread for reference generation, and a thread for user interaction. The control thread must execute at a sampling rate of 4 milliseconds, the reference generator should update the references every 12 milliseconds, and the user interaction thread should poll for user interaction and update the plotters 50 times per second. The total execution times are x milliseconds for the control thread, 4 milliseconds for the reference generator, and 9 milliseconds for the user interaction thread. You may assume that the real-time kernel is ideal.

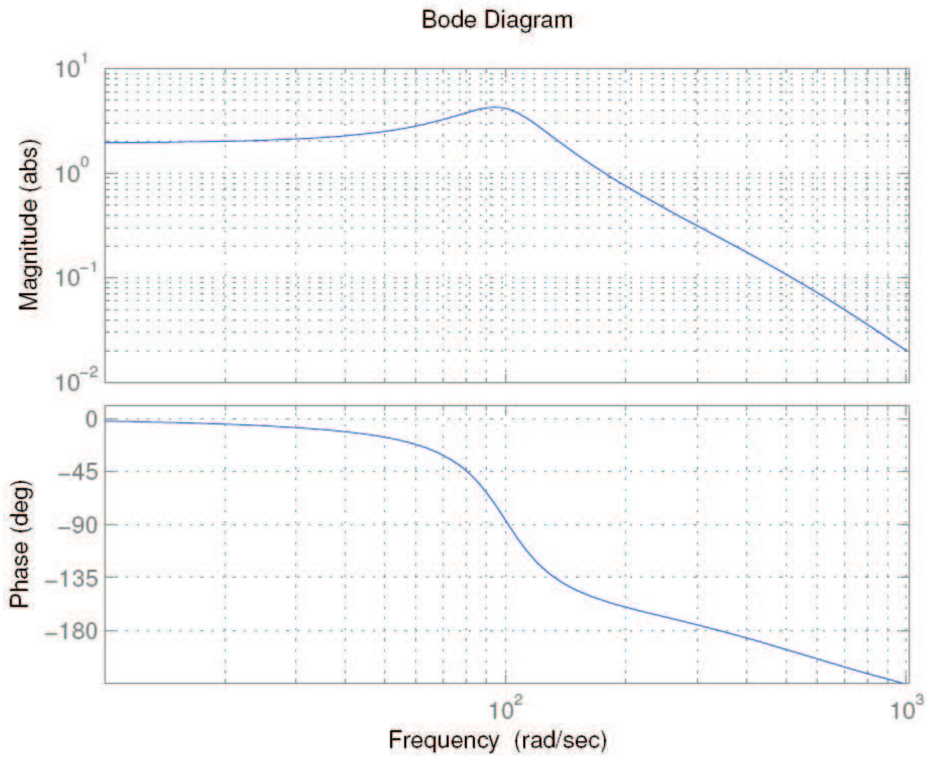
a) Assume that the deadline D_i for each thread is equal to the period T_i. Assume that $x = 0.5$, that is, the controller execution time is 0.5 milliseconds and that all blocking due to interprocess communication can be ignored. Will the task set be schedulable using rate monotonic priority assignments?

(3p)

b) What is the maximum controller execution time x allowed if the task set is to be schedulable using Earliest Deadline First scheduling?

(2p)

c) The open-loop Bode diagram of the system is shown below.



What is the maximum controller execution time x allowed if the closed-loop system should be stable?

(2p)

8

Consider the problem

$$\begin{array}{ll} \max & x_1 + 2x_2 \\ \text{s.t.} & -2x_1 + x_2 \leq 2 \\ & -x_1 + 2x_2 \leq 7 \\ & x_1 \leq 3 \\ & x_1, x_2 \geq 0 \end{array}$$

a) Find the optimal solution.

(3p)

b) Assume x_1 and x_2 can take only integer values. Is your solution also the optimal solution for this case?

(2p)

Good Luck!

Solutions

1

a) Priority inversion is the scenario where a low priority process holds a shared resource that is required by a high priority process. This causes the execution of the high priority task to be blocked until the low priority task has released the resource, effectively "inverting" the relative priorities of the two processes. A medium priority process that does not use the shared resource will preempt the low priority process and thus delay the high priority process as well. For an example see the course book.

b) The priority inversion protocol is a method for handling the priority inversion problem. When priority inversion is used the process scheduling algorithm will increase the priority of a process to the maximum priority of any process waiting for any resource on which the process has a resource lock. The process priority is set back the original priority when the process releases the resource.

c) *From Wikipedia*: The Open Systems Interconnection Basic Reference Model (OSI Reference Model or OSI Model for short) is a layered, abstract description for communications and computer network protocol design. It was developed as part of the Open Systems Interconnection (OSI) initiative and is sometimes known as the OSI seven layer model. From top to bottom, the OSI Model consists of the Application, Presentation, Session, Transport, Network, Data Link, and Physical layers. A layer is a collection of related functions that provides services to the layer above it and receives service from the layer below it. For example, a layer that provides error-free communications across a network provides the path needed by applications above it, while it calls the next lower layer to send and receive packets that make up the contents of the path.

d) *From Wikipedia*: Carrier Sense Multiple Access With Collision Detection (CSMA/CD), in computer networking, is a network control protocol in which:

   - a carrier sensing scheme is used.
   - a transmitting data station that detects another signal while transmitting a frame, stops transmitting that frame, transmits a jam

signal, and then waits for a random time interval (known as "back-off delay" and determined using the truncated binary exponential backoff algorithm) before trying to send that frame again.

2

a) More information is needed. It is schedulable if the deadline is equal to the period, otherwise we cannot tell if it is schedulable or not.

b) No, processes are sorted based on their priority. The waiting process with the highest priority should get a chance to execute.

3

a) Note, that the continous transfer function is stable. (i) has discrete time poles in -2 and 0.25. (ii) has discrete time poles in 0.57 and 0.25. Thus (i) is unstable and (ii) is stable. Since the backward Euler transformation always will map a stable continuous system to a stable discrete time system system (i) must be the one who corresponds to the Euler forward method.
Euler forward:

$$\frac{100}{\left(\frac{z-1}{h}+5\right)\left(\frac{z-1}{h}+20\right)}$$

$$\frac{100\,h^2}{z^2+(25\,h-2)\;z+100\,h^2-25\,h+1}$$

Euler backward:

$$\frac{100}{\left(\frac{z-1}{h\,z}+5\right)\left(\frac{z-1}{h\,z}+20\right)}$$

$$\frac{100\,h^2\,z^2}{(100\,h^2+25\,h+1)\;z^2+(-25\,h-2)\;z+1}$$

b) The Euler forward method does not always map a stable continuous system to a stable discrete time system.

4

a)
   - Ready – The process is ready to execute.
   - Running – The process is currently executing.
   - Blocked – The process is waiting for an event.

b) The reason for the different state transitions are the following:
   - Running $\Rightarrow$ Blocked
     The currently running process waits for an event.
   - Blocked $\Rightarrow$ Ready
     The event that a process is waiting for occurs.
   - Ready $\Rightarrow$ Running
     The currently running process becomes blocked or ready. One of the processes that is Ready is selected to be the next running process. The basis for this selection is determined by which scheduling strategy that is used.

5

a) The deadlock state is presented below. P1 holds A, waits for C. P2 holds B, waits for A. P3 holds C and D, waits for B.

P1	P2	P3
wait(A)	wait(D)	wait(C)
wait(C) $\Leftarrow$	wait(E)	wait(D)
useAC()	useDE()	wait(B) $\Leftarrow$
signal(C)	signal(E)	useBCD()
signal(A)	signal(D)	signal(B)
...	...	signal(D)
wait(E)	wait(B)	signal(C)
wait(A)	wait(A) $\Leftarrow$	
useAE()	useAB()	
signal(A)	signal(A)	
signal(E)	signal(B)	

b) Introduce and order in which the resources are allocated. Following this eliminates the possibility for circular-wait. For example, use the order $A < B < C < D < E$.

P1	P2	P3
wait(A)	wait(D)	wait(B)
wait(C)	wait(E)	wait(C)
useAC()	useDE()	wait(D)
signal(C)	signal(E)	useBCD()
signal(A)	signal(D)	signal(D)
...	...	signal(C)
wait(A)	wait(A)	signal(B)
wait(E)	wait(B)	
useAE()	useAB()	
signal(E)	signal(B)	
signal(A)	signal(A)	

6

It is not necessary to include the calculation of ad and bd. They should only be derived when the parameters changes. This can be done in another part of the program structure. There is no reason to delay the control signal, u, to be sent to the physical process by calculation of I and declaration of yold. They should be done afterwards. The code below shows how the PID Controller could have been implemented to minimize the time delay before the control signal is sent.

```
y := AIIn(ychan)
e := yref-y
D := ad*D + bd*(y-yold)
v := K*(beta*yref-y) + I + D
if mode = auto then u := sat(v,umax,umin)
else u := sat(uman,umax,umin)
DAOut(u,uchan)
I := I + (K*h/Ti)*e + (h/Tr)*(u-v);
if increment then uinc := 1
elseif decrement then
uinc := -1
else
uinc := 0
```

```
uman := uman + (h/Tm)*uinc +(h/Tr)*(u-uman)
yold := y
```

7

a) A sufficient schedulability criterion is if the utilization satisfies the condition

$$\sum_{i=1}^{i=n} \frac{C_i}{T_i} \leq n(2^{1/n} - 1).$$

For this setup the utilization is

$$\sum_{i=1}^{i=n} \frac{C_i}{T_i} = 0.5/4 + 4/12 + 9/20 = 0.9083 > 3(2^{1/3} - 1) \approx 0.78$$

and we can not conclude that the tasks are schedulable. We instead turn to the exact analysis, where the response time for process i can be calculated from iteration of the equation

$$R_i = C_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil C_j.$$

The highest priority task is the control task, which will trivially have a response time which we denote $R_1 = C_1 = x = 0.5$ ms. For the medium priority reference generator we get, with initial guess $R_2^0 = C_2 = 4$:

$$R_2^1 = C_2 + \left\lceil \frac{R_2^0}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{4}{4} \right\rceil 0.5 = 4.5$$

$$R_2^2 = C_2 + \left\lceil \frac{R_2^1}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{4.5}{4} \right\rceil 0.5 = 5.0$$

$$R_2^3 = C_2 + \left\lceil \frac{R_2^2}{T_1} \right\rceil C_1 = 4 + \left\lceil \frac{5.0}{4} \right\rceil 0.5 = 5.0$$

which gives $R_2 = 5.0$ ms. For the user I/O task, with initial guess $R_3^0 = C_3 = 9$ we get

$$R_3^1 = C_3 + \left\lceil \frac{R_3^0}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^0}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{9}{4} \right\rceil 0.5 + \left\lceil \frac{9}{12} \right\rceil 4 = 14.5$$

$$R_3^2 = C_3 + \left\lceil \frac{R_3^1}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^1}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{14.5}{4} \right\rceil 0.5 + \left\lceil \frac{14.5}{12} \right\rceil 4 = 19$$
$$R_3^3 = C_3 + \left\lceil \frac{R_3^2}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^2}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{19}{4} \right\rceil 0.5 + \left\lceil \frac{19}{12} \right\rceil 4 = 19.5$$
$$R_3^4 = C_3 + \left\lceil \frac{R_3^3}{T_1} \right\rceil C_1 + \left\lceil \frac{R_3^3}{T_2} \right\rceil C_2 = 9 + \left\lceil \frac{19.5}{4} \right\rceil 0.5 + \left\lceil \frac{19.5}{12} \right\rceil 4 = 19.5$$

which gives $R_3 = 19.5$ ms. All tasks will therefore meet their deadlines.

b) Utilization $U = x/4 + 4/12 + 9/20 < 1$ for schedulability, which gives $x < 0.867$ milliseconds.

c) From the Bode diagram we see that the phase margin is around 25 degrees, with a crossover frequency of around 180 rad/s. The total phase loss due to a delay of x seconds is then $180\frac{180}{\pi}x = 10300x$ degrees, and we have that x must satisfy $x < 25/10300 = 0.0025$ seconds or equivalently 2.5 ms.

8

a) The problem can be solved graphically or by using the Simplex method. The Simplex method is presented below. Start by writing the problem in standard form. This we do by knowing that $\max x = \min -x$, and by introducing the slack variables, x_3, x_4, x_5.

$$\begin{array}{ll} \min & -x_1 - 2x_2 \\ s.t. & -2x_1 + x_2 + x_3 = 2 \\ & -x_1 + 2x_2 + x_4 = 7 \\ & x_1 + x_5 = 3 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{array}$$

Note that this can be written in matrix form like

$$\begin{array}{ll} \min & c^T x \\ s.t. & Ax = b \\ & x \geq 0 \end{array}$$

with A a rectangular matrix contaning the identity matrix for the slack variables, that is

$$Ax = \begin{bmatrix} N & I_{m \times m} \end{bmatrix} \begin{bmatrix} x_N \\ x_B \end{bmatrix}$$

The variables of x corresponding to N (that is, x_N) are called non-basic, and the other variables are called basic.

Now we are ready to go through the simplex algorithm, that goes like this.

0. Letting the slacks = b, is one solution, this we know
1. Is this the optimal solution? If there are negative coefficients in c^T, then no.
2. Determine a non-basic variable to make basic (heuristically this is the one that decreases $c^T x$ the most, but this is not always the best choice).
3. Determine the basic variable to make non-basic (this is the one that becomes zero when the new basic variable is at its limit).
4. Express all non-basic variables in term of the basic variables.
5. rewrite the objective function in term of the non-basic variables.
6. Goto 1

Here we go!

With the slack variables as basics, is $(x_1, x_2) = (0, 0)$ optimal? No, because increasing any of them will decrease $c^T x$ from its current value (0). So, decide which one of the non-basics to take in. Let's go for x_1 even though x_2 decreases $c^T x$ more. Then we have to decide which one of the slacks (the current basics) to take out. Let x_2 remain at 0, and express the slacks in terms of x_1, then we get,

$$\begin{cases} x_3 = 2 + 2x_1 \\ x_4 = 7 + x_1 \\ x_5 = 3 - x_1 \end{cases}$$

Thus, $x_5 = 0$ when x_1 is at its limit 3, so we make x_5 a non-basic. Rewrite everything in terms of x_2 and x_5.

$$\begin{array}{rc} \min & -3 + x_5 - 2x_2 \\ s.t. & x_3 = 8 - 2x_5 - x_2 \\ & x_4 = 10 - x_5 - 2x_2 \\ & x_1 = 3 - x_5 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{array}$$

Is this now optimal? No, increasing x_2 still decreases $c^T x$, so we do the whole thing agian. This time we go for making x_2 basic (x_5 would be no idea). Which one to take out? With $x_5 = 0$ we have

$$\begin{cases} x_1 = 3 \\ x_3 = 8 - x_2 \\ x_4 = 10 - 2x_2 \end{cases}$$

Thus, $x_4 = 0$ when x_2 is at its limit 5, so we make x_4 non-basic and x_2 basic. Rewrite the everything in terms of the new basic variables x_1, x_2, x_3. We get

$$\begin{array}{ll} \min & -13 + x_4 + 2x_5 \\ s.t. & x_1 = 3 - x_5 \\ & x_2 = 5 - \frac{1}{4}x_4 - \frac{1}{2}x_5 \\ & x_3 = 3 - \frac{3}{2}x_5 + \frac{1}{2}x_4 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{array}$$

Are we now at the optimum? Yes, since there are no negative coefficients in c^T, increasing any variable cannot decrease teh objective function. Thus, $(x_1, x_2) = (3, 5)$ is the optimal extreme point, and the optimal value of the objective function is -13. Of course, this means that the optimal value for the original problem $\max x_1 + 2x_2$ is 13.

b) Assuming a requirement that x_1, x_2 can take only integer values, means that we have solved a relaxation of the problem, since we could have acquired non-integer values. However, since the optimal solution to the relaxed problem is also feasible for the problem with x_1, x_2 integer, the optimal solution $(x_1, x_2) = (3, 5)$ is the optimal solution in both cases.

Some of the problems in this exam is borrowed from exams to a similar course at Lund Institute of Technology.