# Solutions to the exam in Real-Time Systems 180411

These solutions are available on WWW:
*http://www.control.lth.se/course/FRTN01/*

**1.**

**a.** The reachability matrix $W_C = \begin{pmatrix} \Gamma & \Phi\Gamma \end{pmatrix}$ is

$$W_C = \begin{pmatrix} 1 & -2 \\ 0 & 0 \end{pmatrix}$$

which has not full rank (since the second column vector is a scalar multiple of the first column vector), i.e., the system is not reachable and the open loop system has a pole in 0.5 that cannot be modified by the control signal.

**b.** However, since our objective is to place both poles in 0.5 it is indeed possible to do this using state feedback.

The closed loop system becomes

$$\begin{aligned} x(k+1) &= (\Phi - \Gamma L)x(k) + \Gamma l_r y_{ref}(k) \\ &= \begin{pmatrix} -2 - l_1 & 3 - l_2 \\ 0 & 0.5 \end{pmatrix} x(k) \end{aligned}$$

The characteristic polynomial of the closed loop system becomes

$$(z + (2 + l_1))(z - 0.5)$$

which is should be equal to

$$(z - 0.5)(z - 0.5)$$

From this follows that $l_1 = -2.5$ and $l_2$ could have any value, e.g., 0.
In order to have unit static gain ($H_{yy_{ref}} = 1$), $l_r$ should be chosen as

$$l_r = \frac{1}{C(I - \Phi + \Gamma L)^{-1}\Gamma} = 0.5$$

**2.**

**a.** The CPU utilization, $U = 2/4 + 1/3 = 0.833$

**b.** Since the deadlines are different from the periods and we are using EDF the only possibility to determine the scheduability (that is part of the course) is to draw the schedule over one hyperperiod and to check that all deadlines are met. The hyperperiod is $\text{lcm}(4, 3) = 12$ and the schedule is shown in Figure 1. As can be seen from the schedule all deadlines are met and, hence, the taskset is schedulable.

**3.**

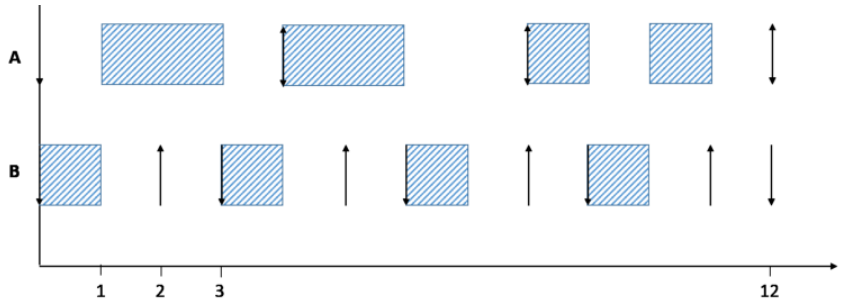



**Figure 1** Solution to Problem **??**. The down arrows indicate job arrivals and the up arrows indicate job deadlines.

**a.** The continuous time transfer function $G(s)$ can be found as

$$G(s) = C\left(sI - A\right)^{-1} B = \frac{1}{s(s+1)}.$$

Using the table from the formula sheet, the pulse transfer function $H(z)$ is determined as

$$H(z) = \frac{e^{-1}z + \left(1 - 2e^{-1}\right)}{z^2 - \left(1 + e^{-1}\right)z + e^{-1}}.$$

The zero is determined from the nominator of $H(z)$, i.e. $z_z = 2 - e = -0.718$. The poles are determined from the denominator of $H(z)$, i.e. $z_p = \frac{1 + e^{-1}}{2} \pm \frac{1 - e^{-1}}{2}$, giving $z_p = 1$ and $z_p = e^{-1} = 0.368$. Since one of the poles is precisely on the unit circle, the system is stable but not asymptotically stable.

**b.** From $H(z)$ one can derive the following expression.

$$y(k+2) - \left(1 + e^{-1}\right)y(k+1) + e^{-1}y(k) = e^{-1}u(k+1) + \left(1 - 2e^{-1}\right)u(k)$$

**4.**

**a.** Replacing $s$ with $\frac{z-1}{h}$ in $G(s) = \frac{s^2 + 2s + 100}{s^2 + 20s + 100}$ gives the following expression for $H(z)$.

$$H(z) = \frac{z^2 + (2h-2)z + 1 - 2h + 100h^2}{z^2 + (20h-2)z + 1 - 20h + 100h^2}$$

Checking the asymptotic stability conditions

$$\begin{aligned} 1 - 20h + 100h^2 &< 1 \\ 1 - 20h + 100h^2 &> -1 + 20h - 2 \\ 1 - 20h + 100h^2 &> -1 - 20h + 2 \end{aligned}$$

gives

$$\begin{aligned} 20h(5h - 1) &< 0 \\ h^2 - 0.4h + 0.04 &> 0 \\ h^2 &> 0 \end{aligned}$$

leading to the following inequalities

$$h < 1/5$$
$$h \neq 1/5$$
$$h \neq 0$$

The solution is thus $0 < h < 1/5$, with the discretized filter

$$H(z) = \frac{z^2 + (2h-2)z + 1 - 2h + 100h^2}{z^2 + (20h-2)z + 1 - 20h + 100h^2}.$$

**b.** The distorted frequency $\omega_1$ is given by (see page 43 in the Computer Control text book),

$$\omega_1 = \frac{2}{h}\tan^{-1}\left(\frac{\omega_d h}{2}\right),$$

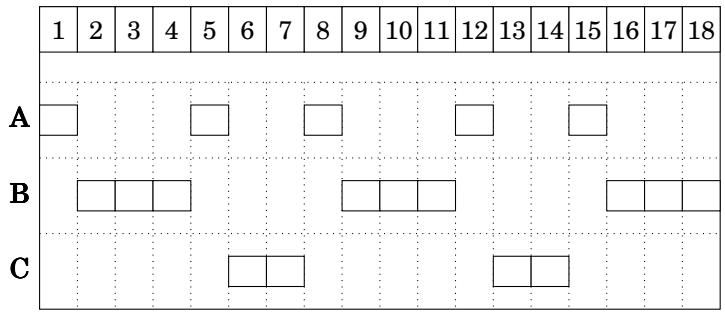
which (when substituting $h = 0.2$s) gives $\omega_1 = 10\tan^{-1}(1) = \frac{5\pi}{2}$ rad/s $\approx 7.85$ rad/s.

**5.** The error is in the `take()` method. The synchronized block is around a too small part of the code. This may result in that, if two consecutive calls to `take()` are made, the "wrong" thread may end up as owner of the semaphore. To fix the problem, put the two lines next to the right bracket inside the synchronized block.

**6.** *This problem was incorrectly formulated. The taskset, in fact, is not schedulable, so no schedule can be generated. We have taken that into account in the correction.*

**a.** The schedule lenght is the least common multiplier between the periods of the tasks, $LCM(3, 6, 9) = 18$.

**b.** The taskset is not schedulable, so a non-preemptive schedule cannot be generated. An example of non-preemptive schedule would have been the following, where A misses its deadline in the last activation period.




If we disregard the missed period, the worst case response time for Task A would be equal to 3 (both at time 15 and at time 12). The worst case response time for task B is equal to 6 (at time 18) and the worst case response time for task C is 7 (at time 7).

**c.** The pseudo-code for the dispatcher of the three tasks that uses the scheduler above is the following. Notice the time increment and the wait until.

```
CurrentTime(t);
LOOP
  A();
  IncTime(t, 1);
  WaitUntil(t);
  B();
  IncTime(t, 3);
  WaitUntil(t);
  A();
  IncTime(t, 1);
  WaitUntil(t);
  C();
  IncTime(t, 2);
  WaitUntil(t);
  A();
  IncTime(t, 1);
  WaitUntil(t);
  B();
  IncTime(t, 3);
  WaitUntil(t);
  A();
  IncTime(t, 1);
  WaitUntil(t);
  C();
  IncTime(t, 2);
  WaitUntil(t);
  A();
  IncTime(t, 1);
  WaitUntil(t);
  B();
  IncTime(t, 3);
  WaitUntil(t);
END;
```

**7.**

**a.** In fixed-point representation, a coefficient $k$ should be stored as an integer $K = \text{round}(k \cdot 2^N)$, where the integer $N$ is the number of fractional bits.

8-bit integers can store values in the range $[-128, 127]$, and the largest magnitude among the coefficients is 3.262. In order to represent the integer part of this we need two bits. That means that $8 - 1 - 2 = 5$ bits should be used for the fractional part in order to get maximal resolution.

The controller coefficients become

$$A = \text{round}(0.8967 \cdot 2^5) = 29$$
$$B = \text{round}(0.2332 \cdot 2^5) = 7$$
$$C = \text{round}(-3.262 \cdot 2^5) = -104$$
$$D = \text{round}(-0.8484 \cdot 2^5) = -27$$

**b.**
```
#define A 29
```

```
#define B 7
#define C -104
#define D -27
#define N 5

int8_t y,x,u;
int16_t x16 = 0, u16 = 0;

...

y = readInput();
/* Calculate output */
u16 = u16 + ((int16_t)D*(int16_t)y) + (1 << N-1) >> N; /* add D*y */
/* check for saturation */
if (u16 > 127) {
  u = 127;
} else if (u16 < -128) {
  u = -128;
} else {
  u = u16;
}

writeOutput(u);

/* Update state */
x16 = (((int16_t)A*(int16_t)x + (int16_t)B*(int16_t)y) + (1 << N-1)) >> N;
/* check for saturation */
if (x16 > 127) {
  x = 127;
} else if (x16 < -128) {
  x = -128;
} else {
  x = x16;
}
u16  = (((int16_t)C*(int16_t)x) + (1 << N-1)) >> N;
```

**8.** A possible solution is shown in Figure 2.

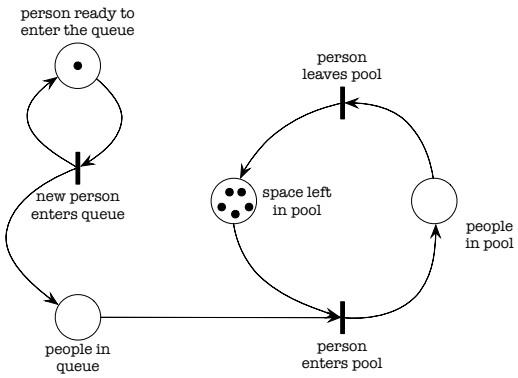



**Figure 2** Solution for problem 8.

**9.**

**a.** By looking at the step response we see that **B** has a stationary error while **A** does not. This means that only one of the controllers should have an integrator. Indeed, looking at $H_{C2}(z)$ we see that it contains $(z-1)$ in the denominator. Therefore we know that **A** correspond to $H_{C2}$ and **B** to $H_{C1}$.

**b.** By looking at the two step responses we see that **C** has a "ringing" control signal. This implies that the model corresponding to that one does not include the undamped process zero found at $z = 0.95$. We see that for step response **D** there is no ringing signal, implying that the undamped process zero is included in the model.

Looking at the two models, we see that $H_{m2}(z)$ include the undamped process zero. Hence, $H_{m1}$ correspond to **C** and $H_{m2}$ correspond to **D**.

**10.**

**a.** The first thing one should realize by inspecting the code is that the two controllers are running with different sampling periods. The controller for the first plant, $P_1(s)$, has a sampling period of $h_1 = h = 0.05$ seconds, while the second controller for $P_2(s)$ has a sampling period of $h_2 = 2 \cdot h = 0.1$ seconds.

When doing the stability analysis and discretizing the plants using ZOH we therefore have the following discretized plants:

$$H_1(z) = \frac{1 - e^{-h_1}}{z - e^{-h_1}}, \qquad H_2(z) = \frac{1 - e^{-10h_2}}{z - e^{-10h_2}}.$$

We should then also notice that both controllers are simple P-controllers with gains $K_1 = 10$ and $K_2 = 20$.

We can now derive the closed loop transfer functions for the two plants and controllers. The first one is given by

$$H_{\text{cl}_1}(z) = \frac{K_1 H_1(z)}{1 + K_1 H_1(z)} = \frac{K_1 \cdot (1 - e^{-h_1})}{z - e^{-h_1} + K_1 \cdot (1 - e^{-h_1})}$$

implying that the pole is located at

$$z_1 = e^{-h_1} - K_1 \cdot (1 - e^{-h1}) \approx 0.46$$

which is stable.

Similariliy, the pole for the second system is given by

$$z_2 = e^{-10h_2} - K_2 \cdot (1 - e^{-10h_2}) \approx -12.27$$

which is outside the unit circle, implying that the closed loop is unstable!

**b.** To remedy the problem of the second closed-loop being unstable it is tempting to change the sampling period of that node by removing the `if`-statement. However, this would move the pole of the second closed-loop system to

$$z_2' = e^{-10h_2/2} - K_2 \cdot (1 - e^{-10h_2/2}) \approx -7.26$$

which does not solve the problem.

A different way to move the close-loop pole of the unstable system would be to change the controller gain $K_2$. In fact, a controller gain of 1 would be sufficient to stabilize the plant since the pole will then be located in

$$z_2'' = e^{-10h_2} - (1 - e^{-10h_2}) \approx -0.26.$$