Solutions to the exam in Real-Time Systems 130108

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

1.

a. $G_1(z)$ has two poles in $z = 1$, which means that it should exhibit unstable behavior. $G_3(z)$ and $G_4(z)$ only differs in the time constant, with $G_4(z)$ faster than $G_3(z)$. $G_2(z)$ has a pole on the negative real axis. Further, its response is on the form $x(n) = -0.99x(n-1) = 0.99 * 0.99x(n-2) = \cdots = (-0.99)^n x(0)$. Therefore the matching is:

$$
\begin{aligned}
G_1(z) &\rightarrow 4 \\
G_2(z) &\rightarrow 1 \\
G_3(z) &\rightarrow 3 \\
G_4(z) &\rightarrow 2
\end{aligned}
$$

b. $G_2(z)$ is a first order process. Yet, it has an oscillatory response. Thus, it cannot be the result of zero-order hold sampling a first-order continuous system.

2.

a. The pulse-transfer function is given by

$$
\begin{aligned}
H(z) &= C(zI - \Phi)^{-1}\Gamma \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z-2 & -1 \\ 0 & z-1 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix} \frac{1}{(z-2)(z-1)} \begin{bmatrix} z-1 & 1 \\ 0 & z-2 \end{bmatrix} \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} \\
&= \frac{0.5}{(z-2)(z-1)}
\end{aligned}
$$

b. The system has a pole outside the unit circle in $z = 2$, so it is unstable.

c. The closed-loop system is given by

$$
\begin{aligned}
x(k+1) &= (\Phi - \Gamma L)x(k) + \Gamma l_r r(k) \\
y(k) &= Cx(k)
\end{aligned}
$$

with the pulse-transfer function

$$
\begin{aligned}
H_{cl}(z) &= C(zI - \Phi + \Gamma L)^{-1}\Gamma l_r \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} z-2 & -1 \\ 0.5l_1 & z-1+0.5l_2 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} l_r \\
&= \begin{bmatrix} 1 & 0 \end{bmatrix} \frac{1}{(z-2)(z-1+0.5l_2)+0.5l_1} \begin{bmatrix} z-1+0.5l_2 & 1 \\ -0.5l_1 & z-2 \end{bmatrix} \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} l_r
\end{aligned}
$$

The characteristic polynomial $(z-2)(z-1+0.5l_2)+0.5l_1 = z^2+(0.5l_2-3)z+2+0.5l_1-l_2$ should be equal to z^2 for deadbeat control, which gives

$$L = \begin{bmatrix} 8 & 6 \end{bmatrix}$$

The static gain is $H_{cl}(1) = 0.5l_r$, which gives $l_r = 2$.

3. Pulse Width Modulation. Used, e.g., in micro-controllers that have digital outouts but no analog outputs. The analog value that should be output is encoded through the duty cycle of of a square wave signal. For example, a square wave signal that is high for 10% of the time and low for 90% of the time will have a duty cycle of 10%. If this signal is sent through an analog low-pass filter the output of the filter will approximate the desired analog output, i.e, the average value of the square wave signal.

4.

a. Sampling the process using the table "Zero-order hold sampling of a continuous-time system with transfer function $G(s)$" gives

$$H(z) = \frac{0.6321}{z-0.3679}$$

The closed-loop system is given by

$$H_{cl}(z) = \frac{KH(z)}{1+KH(z)} = \frac{1.264}{z+0.8964}$$

The pole is located in -0.8964, inside the unit circle, so the closed-loop system is stable.

b. The sampled process, including a one sample delay, is now given by

$$H(z) = \frac{0.6321}{z(z-0.3679)}$$

The closed-loop system is given by

$$H_{cl}(z) = \frac{1.264}{z^2-0.3679z+1.264}$$

The poles are located in $0.1836 \pm 1.1092i$, i.e., outside the unit circle, so the closed-loop system in unstable.

5.

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 ints can store values in the range $[-128, +127]$, and the largest magnitude of any coefficient is 2.278. This means that no more than $\log_2(128/2.78) = 5.81$ fractional bits may be used. $N = 5$ gives the best resolution and should hence be used.

The controller coefficients become

$$A = \text{round}(0.1466 \cdot 2^5) = 5 \qquad B = \text{round}(1.050 \cdot 2^5) = 34$$
$$C = \text{round}(1.517 \cdot 2^5) = 49 \qquad D = \text{round}(-2.278 \cdot 2^5) = -73$$

b. The pole is located in $A/2^5 = 0.1562$, i.e., a 6% error in pole location.

c.
```
#define A 5
#define B 34
#define C 49
#define D -73
#define N 5

uint_8 y, x, u;
uint_16 x16 = 0, u16 = 0;

...

y = readInput();

/* calculate output */
u16 = (u16 + (int16_t)D*(int16_t)y)>>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)>>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;
```

6.

a. Replacing $dx_1(t)/dt$ with a backward difference approximation gives

$$x_1[k] = x_1[k-1] + hx_2[k]$$

Replacing $dx_2(t)/dt$ with a backward difference approximation gives

$$x_2[k] = x_2[k-1] - \frac{1.4h}{T_f}x_2[k] - \frac{h}{T_f^2}x_1[k] + \frac{h}{T_f^2}y[k]$$

Replacing $x_1[k]$ with the first expression gives

$$x_2[k] = x_2[k-1] - \frac{1.4h}{T_f}x_2[k] - \frac{h}{T_f^2}x_1[k-1] - \frac{h^2}{T_f^2}x_2[k] + \frac{h}{T_f^2}y[k]$$

Rearranging the terms leads to

$$(1+\frac{1.4h}{T_f}+\frac{h^2}{T_f^2})x_2[k] = x_2[k-1] - \frac{h}{T_f^2}x_1[k-1] + \frac{h}{T_f^2}y[k]$$

or

$$x_2[k] = \frac{1}{(T_f^2+1.4hT_f+h^2)}(T_f^2 x_2[k-1] - hx_1[k-1] + hy[k]) \qquad (1)$$

Inserting this into the equation for $x_1[k]$ then, finally, leads to

$$x_1[k] = (1-\frac{h^2}{den})x_1[k-1] + \frac{hT_f^2}{den}x_2[k-1] + \frac{h^2}{den}y[k] \qquad (2)$$

$$den = (T_f^2 + 1.4hT_f + h^2)$$

Hence, the solution to the problem is given by Equations 1 and 2.

b. Since $dy_f(t)/dt = x_2(t)$ the only remaining thing is to discretize the integral part with a forward approximation which gives

$$I[k+1] = I[k] + \frac{Kh}{T_I}(y_{ref}[k] - y_f[k])$$

which is the same as

$$I[k+1] = I[k] + \frac{Kh}{T_I}(y_{ref}[k] - x_1[k])$$

The pseudocode for the controller looks like

```
// CalculateOutput
x1 = p1*x1old + p2*x2old + p3*y;
x2 = p4*x2old + p5*(y - x1old);
v = K*(Beta*yref - x1) + I - K*Td*x2;
u = sat(v);
// output u
// UpdateState
I = I + (K*h/Ti)*(yref - x1);
x1old = x1;
x2old = x2;
```

with the precalculated parameters

```
den = Tf*Tf + 1.4*h*Tf + h*h;
p1 = 1 - h*h/den;
p2 = h*Tf*Tf/den;
p3 = h*h/den;
p4 = Tf*Tf/den // equals p2/h
p5 = h/den; // equals p3/h
```

7.

a. First try the condition

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

We get

$$\frac{1}{3} + \frac{7}{16} + \frac{2}{50} = 0.81 > 3(2^{1/3} - 1) = 0.78$$

from which we can't draw any conclusion.
Then try the condition

$$\prod_{i=1}^{n} \left(\frac{C_i^{max}}{T_i} + 1 \right) \leq 2$$

We get

$$\left(\frac{1}{3} + 1\right)\left(\frac{7}{16} + 1\right)\left(\frac{2}{50} + 1\right) = 1.99 \leq 2$$

Yes, all deadlines will be met.

b.

$$R_A = C_A^{max} = 1$$

$$\begin{aligned}
R_B^1 &= C_B^{max} = 7 \\
R_B^2 &= C_B^{max} + \left\lceil \frac{R_B^1}{T_A} \right\rceil \cdot C_A^{max} = 7 + \left\lceil \frac{7}{3} \right\rceil \cdot 1 = 10 \\
R_B^3 &= 7 + \left\lceil \frac{10}{3} \right\rceil \cdot 1 = 11 \\
R_B^4 &= 7 + \left\lceil \frac{11}{3} \right\rceil \cdot 1 = 11
\end{aligned}$$

$$\begin{aligned}
R_C^1 &= C_C^{max} = 2 \\
R_C^2 &= C_C^{max} + \left\lceil \frac{R_C^1}{T_A} \right\rceil \cdot C_A^{max} + \left\lceil \frac{R_C^1}{T_B} \right\rceil \cdot C_B^{max} = 2 + \left\lceil \frac{2}{3} \right\rceil \cdot 1 + \left\lceil \frac{2}{16} \right\rceil \cdot 7 = 10 \\
R_C^3 &= 2 + \left\lceil \frac{10}{3} \right\rceil \cdot 1 + \left\lceil \frac{10}{16} \right\rceil \cdot 7 = 13 \\
R_C^4 &= 2 + \left\lceil \frac{13}{3} \right\rceil \cdot 1 + \left\lceil \frac{13}{16} \right\rceil \cdot 7 = 14 \\
R_C^5 &= 2 + \left\lceil \frac{14}{3} \right\rceil \cdot 1 + \left\lceil \frac{14}{16} \right\rceil \cdot 7 = 14
\end{aligned}$$

c.

$$R_A = C_A^{min} = 0.5$$

$$\begin{aligned}
R_B^1 &= C_B^{min} = 4 \\
R_B^2 &= C_B^{min} + \left\lceil \frac{R_B^1 - T_A}{T_A} \right\rceil_0 \cdot C_A^{min} = 4 + \left\lceil \frac{4-3}{3} \right\rceil_0 \cdot 0.5 = 4 + 1 \cdot 0.5 = 4.5 \\
R_B^3 &= 4 + \left\lceil \frac{4.5-3}{3} \right\rceil_0 \cdot 0.5 = 4 + 1 \cdot 0.5 = 4.5
\end{aligned}$$

$$\begin{aligned}
R_C^1 &= C_C^{min} = 1.5 \\
R_C^2 &= C_C^{min} + \left\lceil \frac{R_C^1 - T_A}{T_A} \right\rceil_0 \cdot C_A^{min} + \left\lceil \frac{R_C^1 - T_B}{T_B} \right\rceil_0 \cdot C_B^{min} \\
&= 1.5 + \left\lceil \frac{1.5-3}{3} \right\rceil_0 \cdot 0.5 + \left\lceil \frac{1.5-16}{16} \right\rceil_0 \cdot 4 = 1.5 + 0 \cdot 0.5 + 0 \cdot 4 = 1.5
\end{aligned}$$

8.

a. The solution is

```
public class Writer extends Thread {

  MultiStepSemaphore sem;

  public Writer(MultiStepSemaphore s) {
    sem = s;
  }

  public void run() {
    while (true) {
      sem.take(3);
    // access critical section
      sem.give(3);
    }
  }
}

public class Reader extends Thread {

  MultiStepSemaphore sem;

  public Reader(MultiStepSemaphore s) {
    sem = s;
  }

  public void run() {
    while (true) {
      sem.take();
    // access critical section
```

```
        sem.give();
      }
    }
  }

  public class Main {

    public static void main(String[] args) {

    MultiStepSemaphore s;
    s = new MultiStepSemaphore(3);
    Writer w;
    Reader r;
    for (int i = 1; i==3; i++) {
      w = new Writer(s);
      w.start();
    }
    for (int i = 1; i==4; i++) {
      r = new Reader(s);
      r.start();
    }
  }
```

b. The class `ReadersWritersGuard` is given below (with all exception handling excluded):

```
public class ReadersWritersGuard {

    int maxWriters = 1;
    int maxReader = 1;

// the number of writer processes inside the section
    int writersCounter = 0;
// the number of reader processes inside the section
    int readersCounter = 0;

    public ReadersWritersGuard() {}

    public ReadersWritersGuard(int maxW, int maxR) {
        this();
        maxWriters = maxW;
        maxReaders = maxR;
    }

    public synchronized void writersTake() {
        while ((writersCounter == maxWriters) || (readersCounter > 0)) {
            wait();
        }
        writersCounter++;
    }

    public synchronized void readersTake() {
        while ((readersCounter == maxReaders) || (writersCounter > 0) ) {
            wait();
```

```
        }
        readersCounter++;
    }

    public synchronized void writersGive() {
        writersCounter--;
        notifyAll();
    }

    public synchronized void readersGive() {
        readersCounter--;
        notifyAll();
    }
}
```

It should be used as

```
public class Writer extends Thread {

  ReadersWritersGuard sem;

  public Writer(ReadersWritersGuard s) {
    sem = s;
  }

  public void run() {
    while (true) {
      sem.writersTake();
    // access critical section
      sem.writersGive();
    }
  }
}

public class Reader extends Thread {

  ReadersWritersGuard sem;

  public Reader(ReadersWritersGuard s) {
    sem = s;
  }

  public void run() {
    while (true) {
      sem.readersTake();
    // access critical section
      sem.readersGive();
    }
  }
}

public class Main {

  public static void main(String[] args) {

  ReadersWritersGuard s;
```

```
  s = new ReadersWritersGuard(2,3);
  Writer w;
  Reader r;
  for (int i = 1; i==3; i++) {
    w = new Writer(s);
    w.start();
  }
  for (int i = 1; i==4; i++) {
    r = new Reader(s);
    r.start();
  }
}
```