LUND UNIVERSITY

Department of
AUTOMATIC CONTROL

Real-Time Systems

Exam June 3, 2015, hours: 14.00–19.00

Points and grades

All answers must include a clear motivation and a well-formulated answer. Answers may be given in **English or Swedish**. The total number of points is 25. The maximum number of points is specified for each subproblem.

Accepted aid

The textbooks Real-Time Control Systems and Computer Control: An Overview - Educational Version. Standard mathematical tables and authorized "Real-Time Systems Formula Sheet". Pocket calculator.

Results

The result of the exam will become accessible through LADOK. The solutions will be available on WWW:
http://www.control.lth.se/course/FRTN01/

1. Given the following discrete-time system:

$$x(k+1) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u(k)$$
$$y(k) = [1 \quad 0]\, x(k)$$

a. Design a deadbeat controller (i.e. state-feedback with all poles at the origin) such that the closed loop system has unit static gain from r to y. (2 p)

b. Design an observer on predictor form with both poles at $z = 1/2$. To get full points you should also write down the equations for the observer. (2 p)

2. A system is given by

$$\frac{dx(t)}{dt} = \begin{bmatrix} -1 & 1 \\ 0 & 2 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 2 \end{bmatrix} u(t)$$
$$y(t) = [1.5 \quad 0]\, x(t).$$

Sample the system using zero-order hold. Give the answer as a pulse-transfer function. (1.5 p)

3. Consider the following discrete-time system

$$x(k+1) = \begin{bmatrix} 0.9 & 0.8 \\ 0.5 & 0.0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(k)$$

$$y(k) = [1 \quad 1]\, x(k).$$

The input to this system is constant, $u(k) = 1, \forall k$ s.t. $k \geq 0$.

a. Calculate $y(3)$ (the output at time $k = 3$) assuming initial state $x(0) = [0 \quad 0]^T$. (1.5 p)

b. Will the system converge, and in that case, to what output value? Solve this on paper, without using a calculator. (1 p)

4. Linda is writing a program for a small micro-controller, using EDF scheduling. Table 1 shows the task set she will use, along with the required periods, the relative deadlines, and the execution times that she has currently been able to achieve. There is only one problem: the task set is not schedulable!

Table 1: Task set in problem 4

Task	T	C	D
A	10 ms	7 ms	10 ms
B	20 ms	4 ms	20 ms
C	1 ms	0.4 ms	1 ms

a. To make the task set schedulable, Linda will try to optimize her code. For each of the tasks, how many times faster would she have to make it if that is the only code she will optimize? (2 p)

b. For the first prototype, there is no time to optimize any code, so Linda decides to lengthen the period of task C instead. How long does it need to be? (1 p)

5. The following system is to be implemented as a part of a process simulator on a microprocessor that does not has hardware-supported floating point arithmetics. Therefore it was decided to use fixed point arithmetics. It was decided to use signed 8-bit word length for both the variables (states, input, output) and the parameters.

$$x(k+1) = \begin{bmatrix} 0.35 & 4.25 \\ 0.05 & 0.65 \end{bmatrix} x(k) + \begin{bmatrix} 1.00 \\ 2.00 \end{bmatrix} u(k)$$

$$y(k) = [\, 1.25 \quad 2.05 \,]\, x(k).$$

Since the largest coefficient has integer part 4, we need $m = 3$ integer bits, which leaves $n = 8 - m - 1 = 4$ fractional bits.

a. What is the Q(3,4) representation of the parameters in the Φ, Γ and C matrices? (1 p)

b. Looking at the poles of the original and the approximated system, is this choice of system representation a good choice? (1.5 p)

6. Consider the ordinary PID controller given below:

$$U(s) = K(\beta Y_r(s) - Y(s) + \frac{1}{T_i s} E(s) - \frac{sT_d Y(s)}{1 + sT_d/N})$$

a. Discretize the I-part using a backward difference approximation (backward Euler). (1 p)

b. Discretize the D-part using Tustin approximation. (1 p)

c. Write the pseudo-code for the controller discretized in the above way by filling in the blanks in the code snippet below. The code does not have to include any anti-windup or any manual control mode. The code should be written so that the input-output latency is minimized. (1.5 p)

```
y = yIn.get();
e = yref - y;
...
...
uOut.put(u);
...
...
```

7. A metal workshop has, due to a decreasing demand for gasoline tanks, started to produce lighters instead. The lighters consist of two parts, the fuel unit and the fire unit. The two units are produced separately in production lines. In order to facilitate the high-level control the company uses a JGrafchart program to coordinate the two production lines (shown in Fig.1) It is supposed to work in the following way:

- When the operator presses the start-button the production of 10 batches should start.
- Every batch consists of 25 fuel units and 50 fire units.
- The production of fuel and fire units should execute in parallel since the time for producing the specified units might differ.
- When 10 batches are produced the system should return to the initial state and wait for the operator to press start again.
- The digital input `Start` is true when the operator presses the start button.
- Fuel and fire units are produced when the digital outputs `ProduceFuel` and `ProduceFire` are true.
- The analog inputs `FuelUnitsDone` and `FireUnitsDone` count the number of units produced since the respective production line last started to produce units. You can assume that the inputs only take integer values.
- The integer `Batches` should count how many batches that has been produced since the operator pressed the start button.

The best engineer in the workshop has put a lot of time and effort into the program but it still doesn't work the way it is supposed to. The engineer has made sure that the inputs and outputs works the way they should. However, the engineer has made four mistakes in the implementation of procedure. Find and correct his misstakes without making structural changes to the program, that is, you are not allowed to add or remove blocks or transitions or make changes in how the blocks and transitions are connected.

In JGrafchart (the Grafcet system used in Lab 2) a N action is the same as a standard action (level action) and an S action is the same as a stored action (impulse action).

(2 p)

8. Does the PI-controller implementation below support bumpless parameters changes when the control loop is in stationarity (controller error equal to 0)? If your answer is Yes then explain how. If you answer is No then modify the code so that it does. (Hint: Consider all the controller parameters)

(2 p)

```
public class PI {
  private PIParameters p;
  private double I; // Integrator state
  private double v; // Desired control signal
  private double e; // Current control error

  //Constructor
```

```
public PI(String name) {
  PIParameters p = new PIParameters();
  p.Beta = 0.8;
  p.H = 0.1;
  p.integratorOn = false;
  p.K = 1.0;
  p.Ti = 0.0;
  p.Tr = 10.0;
  new PIGUI(this, p, name);
  setParameters(p);

  this.I = 0.0;
  this.v = 0.0;
  this.e = 0.0;
}

public synchronized double calculateOutput(double y, double yref) {
  this.e = yref - y;
  this.v = p.K * (p.Beta * yref - y) + I;
  return this.v;
}
```

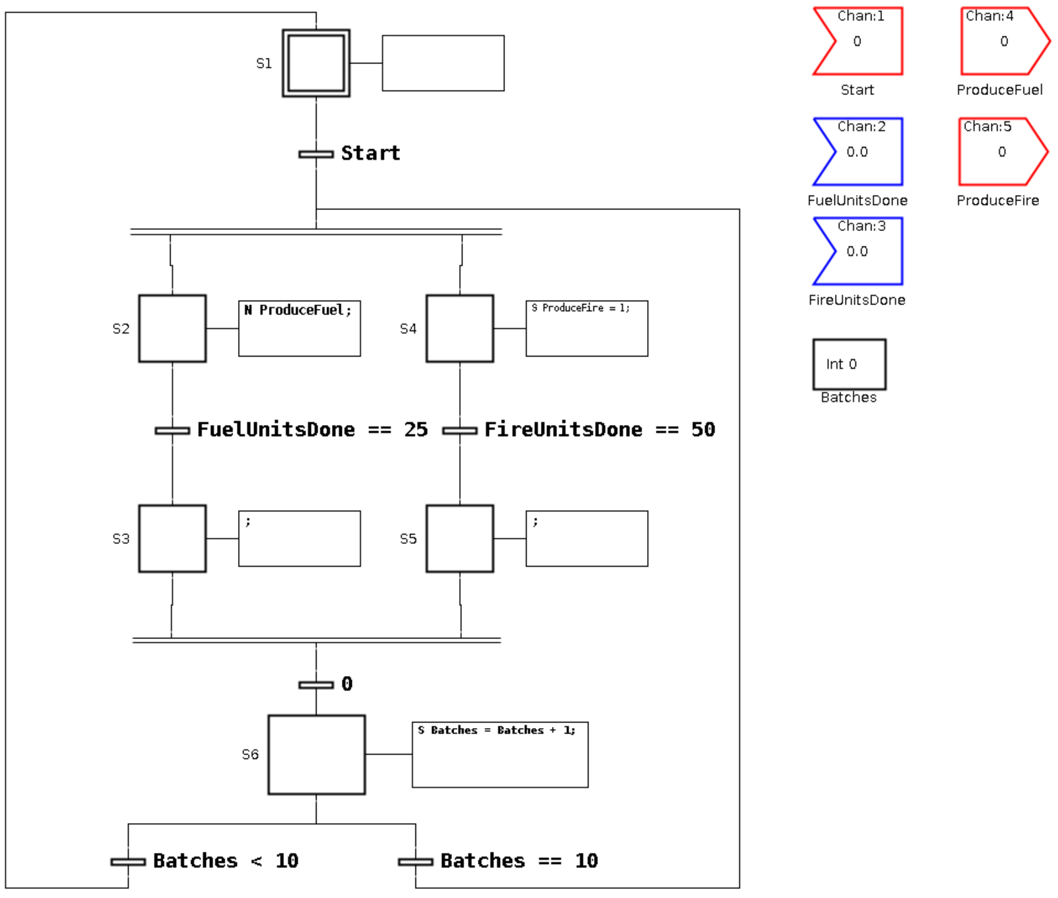

Figure 1: JGrafchart structure for problem 7.

```
  public synchronized void updateState(double u) {
    if (p.integratorOn) {
      I = I + (p.K * p.H / p.Ti) * e + (p.H / p.Tr) * (u - v);
    } else {
      I = 0.0;
    }
  }

  public synchronized long getHMillis() {
    return (long)(p.H * 1000.0);
  }

  public synchronized void setParameters(PIParameters newP) {
    p = (PIParameters)newP.clone();
    if (!p.integratorOn) {
      I = 0.0;
    }
  }
}
```

9. Consider the following simplified Beam control application from Computer Exercise 3 and Lab 1. The system consists of a Opcom class implemented in Swing that permits the user to change the controller parameters and shows the usual signal plots, a Regul thread that implements a PI controller for controlling the beam, and a ReferenceGenerator thread for updating the reference signal. The Opcom GUI also contains a button that the user can press in order to shut down the application.

When closing down the application it is important that the beam is left in a state that is not dangerous to the user. Since the beam dynamics is an integrator it is enough to make sure that the control signal is set to 0 before the system shuts down. In this way the beam will remain at its current position.

A Java application is shutdown by calling the *System.exit*(0) method. This method shuts down the Java Virtual Machine (JVM) by terminating all its threads. The method never returns and it is undefined in which order the threads are terminated.

The call to `System.exit(0)` is performed in the `actionPerformed` method of a `Listener` that listens to the shutdown (stop) button as shown below.

```
  stopButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
      regul.shutDown(); // Method in the Regul class
      measPanel.stopThread(); // Stops a PlotterPanel thread
      ctrlPanel.stopThread(); // Stops a PlotterPanel thread
      System.exit(0);
  }
});
```

a. Consider the following implementation of the regul thread below. Methods that are not of relevance to the questions are omitted from the code. Why does this implementation not work? The problem is not that `weShouldRun` is accessed without mutual exclusion. (1 p)

```
public class Regul extends Thread {

  private PI inner = new PI("PI");

  private AnalogIn analogInAngle;
  private AnalogOut analogOut;

  private ReferenceGenerator referenceGenerator;
  private OpCom opcom;

  private int priority;
  private volatile boolean weShouldRun = true;
  private long starttime;


  public Regul(int pri) {
    priority = pri;
    try {
      analogInAngle = new AnalogIn(0);
      analogOut = new AnalogOut(0);
    } catch (IOChannelException e) {
      System.out.print("Error: IOChannelException: ");
      System.out.println(e.getMessage());
    }
  }

  // Called in every sample in order to send plot data to OpCom
  private void sendDataToOpCom(double yref, double y, double u)
  // Omitted
  }

  // Called from OpCom when shutting down
  public synchronized void shutDown() {
    weShouldRun = false;
  }


  public void run() {
    long duration;
    long t = System.currentTimeMillis();
    starttime = t;

    setPriority(priority);
    while (weShouldRun) {
      double u = 0.0;
      double y = 0.0;
      double yRef = 0.0;

      yRef = referenceGenerator.getRef();
```

```
        y = getIn(analogInAngle);
        synchronized (inner) {
          u = limit(inner.calculateOutput(y, yRef));
          setOut(u);
          inner.updateState(u);
        }
        sendDataToOpCom(yRef, y, u);

        // sleep
        t = t + inner.getHMillis();
        duration = t - System.currentTimeMillis();
        if (duration > 0) {
          try {
            sleep(duration);
          } catch (InterruptedException x) {
          }
        }
      }
      setOut(0);
    }

    private double getIn(AnalogIn ain) {
    // Omitted
    }

    private void setOut(double u) {
    // Omitted
    }
  }
```

b. Provide a solution that is guaranteed to work. In the solution you may use the Java synchronization (synchronized, wait(), notify(), notifyAll()) and/or the explicit Semaphore class provided in the se.lth.control package. You do not need to repeat the code from the first subproblem, it is enough if you indicate which changes that are needed to the code. (3 p)