LUND UNIVERSITY

Department of
AUTOMATIC CONTROL

Real-Time Systems

Exam May 12, 2016, 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 results will be reported via LADOK. Solutions will be available on the course homepage: `http://www.control.lth.se/course/FRTN01/`

1. An industrial process with a long transport delay is described by the continuous-time transfer function

$$G(s) = \frac{1}{(1+10s)^2} e^{-30s}$$

a. Discretize the process using zero-order hold with the sampling interval $h = 1$ and write down the resulting discrete-time transfer function $H(z)$. (2 p)

b. What is the order of the discrete-time system? Is the discrete-time system stable? (1 p)

2. Haeley Hacker has implemented a real-time application with three critical common resources R1, R2 and R3 protected by mutual exclusion semaphores accessed by three different processes P1, P2 and P3. The critical sections in the three processes are accessed through the following statements:

P1	P2	P3
Wait(R1);	Wait(R2);	Wait(R3);
Wait(R2);	Wait(R3);	Wait(R1);
// Using R1 and R2	// Using R2 and R3	//Using R1 and R3
Signal(R2);	Signal(R3);	Signal(R1);
Signal(R1);	Signal(R2);	Signal(R3);

Do you see a problem with this solution? If so, explain why and suggest an implementation avoiding the problem. (2 p)

3. The workers at Maggie's farm have become tired of sorting the cattle every day. With the aid of an engineering student from Lund, they have constructed an automatic sorting and milking pen (Swedish: *fålla*). The logic that controls the pen has been implemented in JGrafchart, see Figure 1.

The system sorts the cattle based on their age and gender. When an animal enters the pen, `AnimalInPen` is activated. The gender and age sensors are read, setting the variables `FEMALE` (Bool) and `AGE` (Real). The animal is then placed in a category and an appropriate action is taken.

The control system has the following inputs:

- `AnimalInPen` is active when there is an animal in the pen.
- `ReadGender` reads the gender of the animal.
- `ReadAge` reads the age of the animal.

The system has the following outputs:

- `Milking` activates the milking machine.
- `ReleaseGateOpen` opens the gate to the pasture.
- `AbattoirGateOpen` opens the gate leading to the abattoir.

a. Describe (in words) how the animals are sorted and what action is taken for each category. (1 p)

b. Maggie's brother, who is the supervisor at the farm, thinks that there is a flaw in the logic. Change the logic of the program so that female cattle older than 10 years are sent to the abattoir. No structural changes to the program should be performed. (1 p)

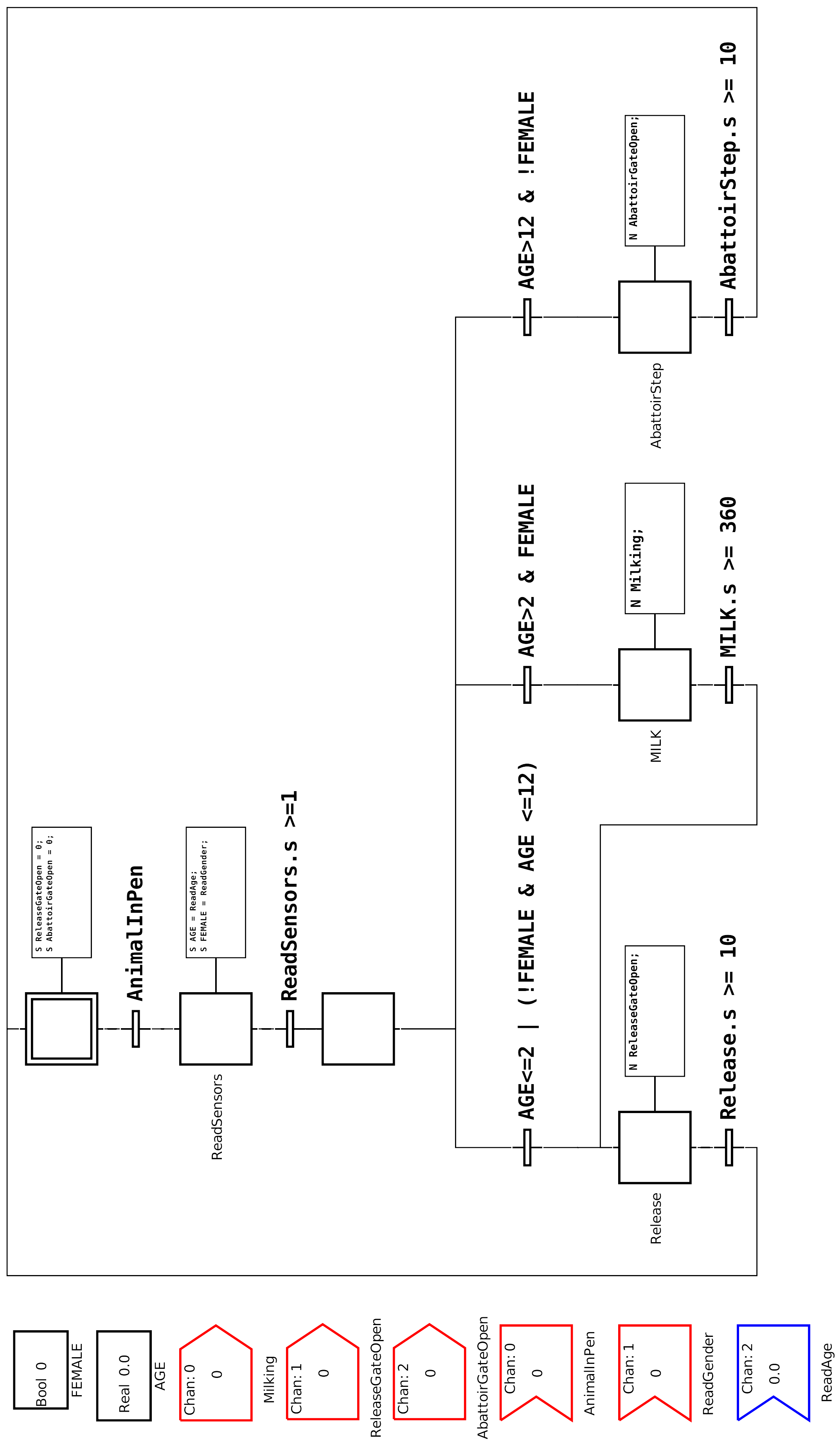

Figure 1 JGrafchart program in Problem 3

4. Consider the following control application. Three classes are involved: `Regul`, `Opcom`, and `Refgen`. `Regul` implements the controller, `Opcom` implements a Swing-based GUI, and `Refgen` calculates the reference signal. There are also three threads involved; the Regul thread, the Refgen thread, and the Swing event-dispatching thread within Opcom. Refgen needs to pass information concerning the current reference signal to Regul. Similarly, Opcom needs to pass new controller parameters to Regul.

Consider the following two implementations. Implementation 1 uses two monitors: `ReferenceMonitor` and `ParameterMonitor`. The two monitors are implemented as inner classes inside the `Regul` class. The ReferenceMonitor has two synchronized methods: `setRef` and `getRef`. The method `setRef` is called from the unsynchronized method `setRef` of `Regul`, which is called by the `Refgen` thread. The Regul thread calls `getRef` to extract the reference value. Similarly, the ParameterMonitor has the two synchronized methods `setParameters` and `getParameters`. The method `setparameters` is called from the unsynchronized method `setParameters` of `Regul`, which is called by the Swing event-dispatching thread. The Regul thread calls `getParameters` to extract the new parameters.

In Implementation 2 the `setRef`, `getRef`, `setParameters`, and `getParameters` are instead all implemented as synchronized methods of `Regul`. No separate inner classes are used to implement the monitors.

What is the difference between these two implementations from a synchronization/blocking point of view? Which solution is the best from a synchronization/blocking point of view? (2 p)

5. Consider the system

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

a. Design a state feedback controller $u(k) = -Lx(k) + l_r r(k)$ such that the poles of the closed-loop system are located in 0.5 and that the static gain from r to y is equal to 1. (2 p)

b. Design a predictor-form observer with both poles located at 1/4. (1.5 p)

6. Consider the set of tasks in Table 1, where T denotes the task period, D denotes the deadline and C denotes the execution time of the task.

a. Assign a priority to each task according to the Rate Monotonic Scheduling (RMS) principle, and check whether the tasks will meet their deadlines using the approximate, utilization-based schedulability condition. (1 p)

Table 1 The task set in Problem 6

Task name	T	D	C
A	3	3	1
B	5	5	2
C	2	2	0.5

b. Check whether the task set is schedulable using response time analysis and assuming RMS priority assignment. (2 p)

c. Draw the execution schedule for the task set assuming RMS and that all tasks are released simultaneously. Draw the schedule for the full hyperperiod. (2 p)

d. Is the task set schedulable using the Earliest Deadline First (EDF) scheduling algorithm? (1 p)

7. A stable continuous-time controller for a chemical plant is given (using the differentiation operator $p = \frac{d}{dt}$) by the equations

$$v(t) = \frac{s_1 p + s_0}{p^2 + r_1 p + r_0} y(t) \tag{1}$$

$$u(t) = \gamma(t) v(t) + K y(t) \tag{2}$$

where $\gamma(t)$ is an unknown and time-varying parameter. A simple estimator of γ has the form

$$\gamma(t) = \int_0^t \beta(y(\tau)) d\tau \tag{3}$$

where β is a known function of the output y.

a. Discretize the controller given by the equations (1)–(3) using the sampling interval h and the forward differences method.

(2.5 p)

b. The performance of the discretized controller was considered somewhat unsatisfactory, since the resulting controller became unstable for the chosen parameters. Someone suggested that you use Tustins's method instead. Explain why this is a better option than forward differences in this case.

(0.5 p)

8. A linear, sampled-data description of a DC servo process is given by

$$x(k+1) = \underbrace{\begin{pmatrix} 0.75 & 0 \\ 0.25 & 1 \end{pmatrix}}_{\Phi} x(k) + \underbrace{\begin{pmatrix} 0.25 \\ 0.0625 \end{pmatrix}}_{\Gamma} u(k)$$

You would like to simulate the process using a C program running on a small microcontroller without support for floating point arithmetics. Assume the following design specifications:

- The input u and the states x_1 and x_2 are represented by integers. It is known that the input can only take values in the range $[-256, 255]$.
- All calculations in the C program, including intermediate calculations, should be carried out using 16-bit integers (type `int16_t`).
- No variables, including intermediate results, should be allowed to overflow. Rather, they should be saturated.

a. What is the smallest (constant) number of fractional bits needed to represent the coefficients in the Φ and Γ matrices using fixed-point representation without round-off errors? Use this number and convert the coefficients to fixed-point representation. (1 p)

b. Sketch C code for simulating the process one step ahead, given current values of u, x_1 and x_2 stored in the `int16_t` variables `u`, `x1`, and `x2`. Saturate the variables `x1` and `x2` at suitable values so that it is guaranteed that overflow cannot occur. (To make the problem easier, you may assume the same saturation limits for both `x1` and `x2`.) (2.5 p)