LUND
UNIVERSITY

Department of
AUTOMATIC CONTROL

Real Time Systems, FRTN01 & FRTN60

Exam 30 May 2020 kl 08–13

Points and Grades

All answers must include a clear motivation, including the derivation of how the solution has been obtained, 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.

The number of points needed to pass the exam will not exceed 12. The number of points needed to get grade 4 will not exceed 17 and the number of points needed to get grade 5 will not exceed 22.

Accepted Aids

On this distance exam all available material is allowed, such as books, old exams, exercise manuals, Google, using Matlab, etc. You may of course not communicate with or ask someone else for help. You will have contact with the teacher using zoom during the exam.

Results

The result of the exam will become accessible through LADOK. The solutions will be available on the course home page.

1. You are working at a company designing controllers. As you have taken the real-time systems course at LTH you are brave enough to suggest to your employer that you should change the SISO controller structure from Figure 1 to Figure 2.

a. How do you motivate this change? (1 p)

b. How should $G_{ff}(s)$ be chosen in order to get $Y(s) = G_m(s)R(s)$? (1 p)

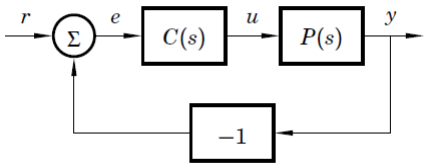

Figure 1 The old structure in Problem 1.

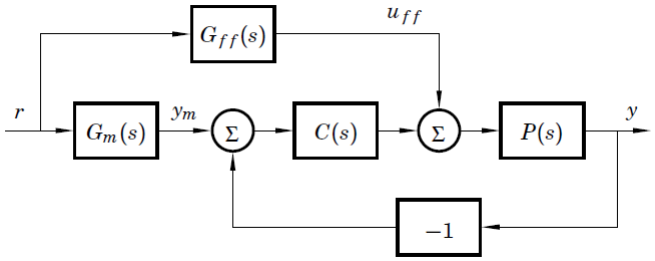

Figure 2 The new structure in Problem 1.

Solution

a. In the original structure, C is responsible for handling both set-point changes and disturbances. This means that there is a trade off in the performance. In the new structure, the problems have been separated. C now handles disturbances and G_m and G_{ff} handles set-point changes. This gives one more degree of freedom and therefore often better performance.

b. The transfer function from r to y is

$$Y(s) = \frac{P(G_{ff} + CG_m)}{1 + PC} R(s).$$

If $G_{ff} = G_m/P$ the transfer function from r to y becomes equal to G_m, i.e., perfect model following.

2.

a. Consider the discrete-time difference equation

$$y(k+2) - y(k) = u(k)$$

Show that using a discrete-time P-controller it is possible to achieve dead beat control. Assume that $y_{ref} = 0$. (1 p)

b. A continuous-time transfer function is

$$G(s) = \frac{K_p}{s+2}e^{-0.2s} \tag{1}$$

Show that despite this being a stable continuous-time system, some choices of discretization methods can give an unstable discrete-time system. (1 p)

c. Sample the system (1) using zero-order-hold with the sampling interval $h = 0.1$. Give the answer on pulse transfer function form ($H(z) = \ldots$). (1 p)

Solution

a. Using the Z-transform, the difference equation is transformed to

$$Y(z) = \frac{1}{z^2-1}U(z)$$

For dead-beat control, we want the closed loop characteristic polynomial to have all roots in 0. Assuming a P-controller is used, the loop transfer function becomes

$$\frac{K}{z^2-1}$$

giving the closed loop characteristic polynomial

$$z^2 - 1 + K$$

Deadbeat control can thus be achieved with a P-controller with $K = 1$.

b. When using forward Euler ($s' = \frac{z-1}{h}$), some stable continuous poles will map to unstable discrete time poles. We can disregard the time delay and just see what happens to the pole.

$$\frac{K_p}{s+2} \rightarrow \frac{K_p}{\frac{z-1}{h}+2}$$

The discrete time pole is in

$$z = 1 - 2h$$

i.e. unstable for $h > 1$

c. Since the time delay is a multiple of the sampling period, the solution is straightforward. The pulse transfer function for the system without delay is (more or less directly from the formula sheet)

$$H(z) = \frac{K_p 0.0906}{z - 0.8187}$$

The two samples input delay simply adds two pure time delays, i.e., two poles at the origin. Hence

$$H(z) = \frac{K_p 0.0906}{z^2(z - 0.8187)}.$$

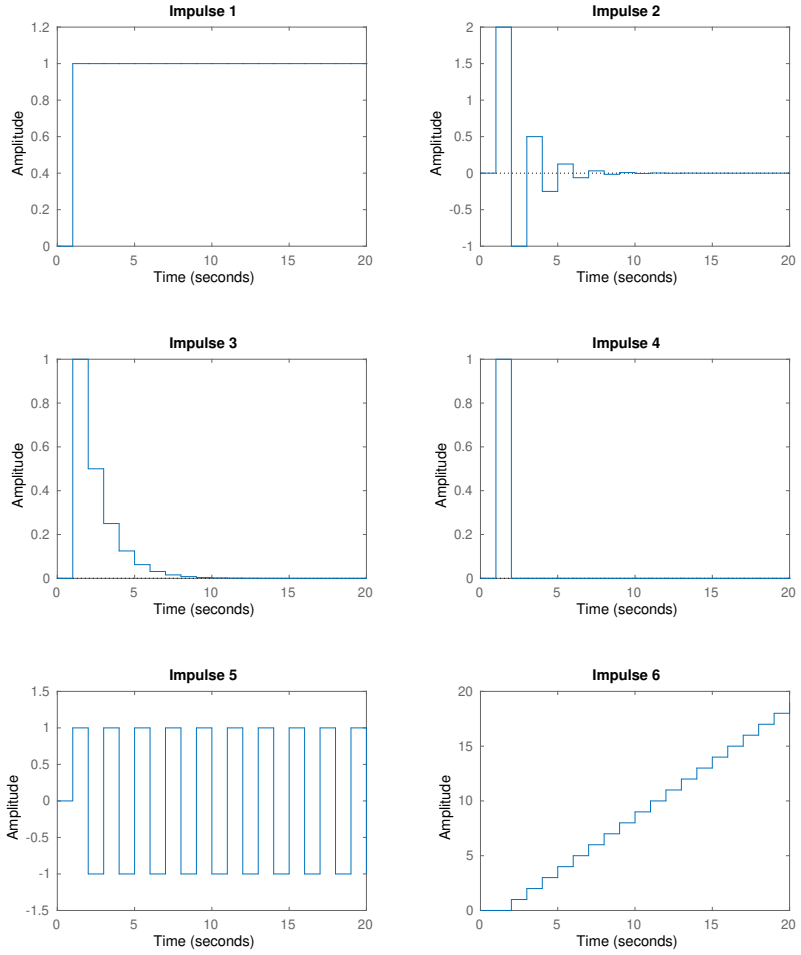

Figure 3 Impulse responses for Question X

3. Match the impulse responses in Figure 3 to the pole zero plots from Figure 4. Do not forget to motivate your answers. (2 p)

Solution

- Impulse1 = pzplot5. Marginally stable system with pole on the positive real axis
- Impulse2 = pzplot1. Stable system with oscialltory response compared to Impulse 3 and 4. This is due to a negative pole
- Impulse3 = pzplot3. Stable system with a positive pole and slower response compared to Impulse 4
- Impulse4 = pzplot6. Stable system with a pole at origin thus fast response
- Impulse5 = pzplot2. Marginally stable with a pole on the negative real axis

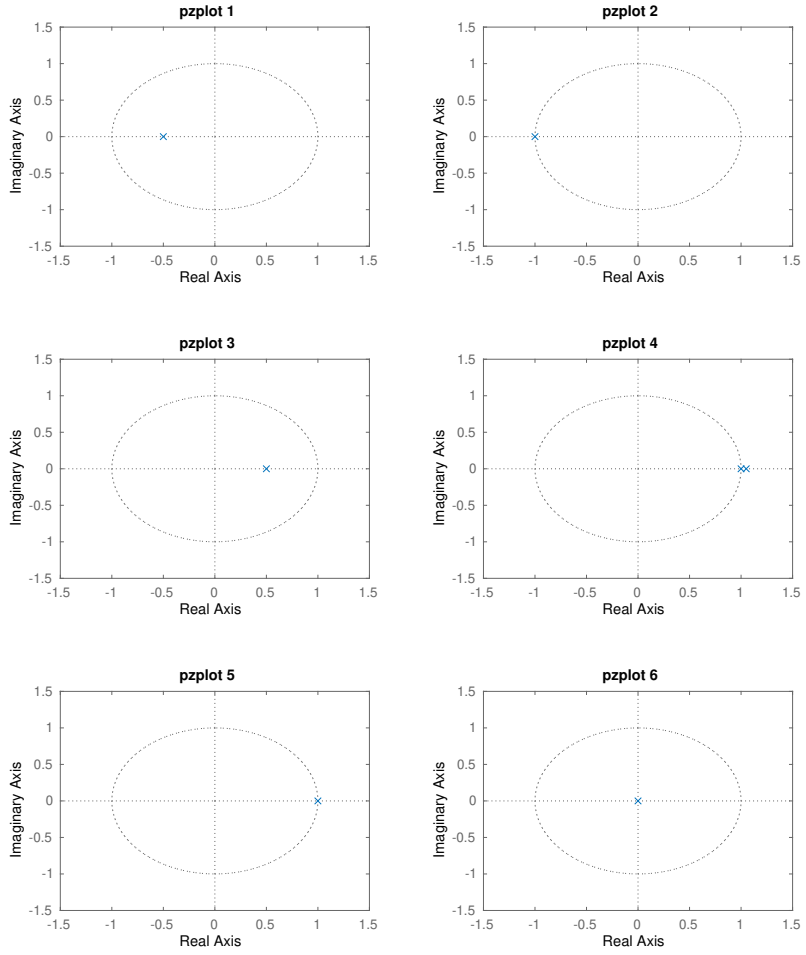

Figure 4 Pole zero plots for Question X

- Impulse6 = pzplot4. Unstable system, i.e., poles outside the unit circle

4. You are implementing the position control for a robot arm performing robot-assisted surgery. The application requires at the same time high precision and the use of simple enough hardware and software. The latter is needed for the instrument certification – the simpler the system the lower the cost of the certification. For this reason you have to write the code using fixed-point implementation and a 32-bits words processor.

The required position resolution is of the order of tens of micrometer (i.e. $10^{-5}m$). The control system has a position sensor with a range (i.e. maximum and minimum measurements) of $\pm 0.5m$. The sensor outputs a signal in Volt in the range $\pm 2V$ where $-2V$ corresponds to $-0.5m$ and $2V$ corresponds to $0.5m$. The volt signal is converted to digital with a A/D converter.

a. What is the variation in voltage that we need to be able to detect in order to achieve the desired precision? (1 p)

b. How many bits are needed in the A/D converter to achieve the desired resolution? (give the lower bound) (1 p)

c. Assuming that you use the minimum number of bits in the A/D converter, what is the measure resolution that is actually achieved? (1 p)

Solution

a. We want to detect a variation of at most 10 micrometers. This is equivalent to 0.00001/0.5 of the maximum span. Therefore the variation of volt that we want to be able to detect is $2 * 0.00001/0.5 = 0.00004V$

b. To detect the desired variation we need to be able to detect a $0.5/0.00001 = 50000th$ of the maximum span (calculated from $0.5/0.00001 = 50000$). Therefore at least 16 bits are needed since they will allow us to detect a $65536th$ of the maximum span (calculated from 2^{16}). Finally also the sign bit has to be taken into account so at least 17 bits are required in the A/D converter.

c. With 17 bits we are able to detect a movement of $0.5/65536 = 7.63\mu m$ which (as expected) is compliant with the specification.

5. The young engineer Carl was given the task to implement a PID controller on a small microcontroller that supports fixed priority scheduling. The controller was to be implemented as a single periodic task. On the same microcontroller also another multi-threaded application was executing. This application was developed by a third party company and Carl did not have complete information about the application.

In the first document he found about the application the only information he found was the following:

- The application consist of a number of periodic tasks (threads).
- The tasks fulfill the usual assumptions, i.e. no interprocess communication, no self-suspension, zero context switch overhead, deadline equal to period, etc.
- The priorities of the tasks are assigned using rate monotonic priority assignment.
- The total utilization of the tasks in the application is 50.0%.
- The period of the highest rate task is 60 ms
- The priority of the highest rate task is 90 (assume that the priority range is 0 (lowest priority) to 100 (highest priority))

a. Assume that the worst case execution time of the PID controller is 10 ms. What is the shortest period that the Carl can use for his controller in order for the entire task set, including the PID task, to be schedulable. (0.5 p)

b. After a while Carl found out that the application consisted of two tasks. Given this information what is the shortest period that Carl may use? (0.5 p)

c. After additional investigations Carl found out that the utilization of the two tasks in the application was divided as 0.3 and 0.2. Using this information what is the shortest period that Carl may use? (1 p)

d. Assume that there was some way for Carl to influence how the utilization of the two tasks in the application was divided (e.g. 0.3-0.2, 0.4-0.1, 0.45 - 0.05 etc). What would be the best division from Carl's point of view and what would then the shortest period be for the PID task? (0.5 p)

e. EDF scheduling and fixed priority scheduling have different overrun behaviour (the behaviour when a task executes longer than its worst-case execution time). Describe how each of the two scheduling policies behaves with respect to overruns. (Note: This subproblem is completely independent from the previous subproblems.) (1 p)

Solution

a. With the information available the only possibility that Carl has to check the schedulability is to use the 69% rule of thumb on the utilization. Given that the application utilizes 50% there is only 19% remaining for the PID task. Hence using this argumentation the shortest period that Carl can use is given by

$$\frac{10}{T_{PID}} < 0.19,$$

i.e. $T_{PID} > 52.6$ ms

b. With this knowledge we can use the approximative schedulability test, i.e.

$$0.5 + \frac{10}{T_{PID}} \leq 3(2^{1/3} - 1) = 0.7798$$

From this follows that $T_{PID} > 35.74$ ms

c. Now Carl may use the hyperbolic schedulability test, i.e.

$$(0.3 + 1)(0.2 + 1)(\frac{10}{T_{PID}} + 1) \leq 2$$

from which follows that $T_{PID} > 35.45$ ms

d. The best situation is if almost all the utilization comes from one of the tasks. This would give

$$\frac{10}{T_{PID}} < 2/1.5 - 1$$

from which follows that $T_{PID} > 30$ms

e. Using fixed priority scheduling a task that overruns will only disturb tasks of equal or lower priority. These tasks may miss their deadlines or be completely starved. Using EDF all tasks will be influenced, i.e. will miss deadlines.

6. The ball and beam process can be described by a system of the form

$$\frac{dx}{dt} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u.$$

Sample the system using ZOH and sampling interval h, i.e. for arbitrary value of h. (1 p)

Solution

$$\Phi(h) = e^{Ah} = \sum_{k\geq 0} \frac{1}{k!} A^k h^k = I + Ah + \frac{1}{2} A^2 h^2 = \begin{bmatrix} 1 & h & h^2/2 \\ 0 & 1 & h \\ 0 & 0 & 1 \end{bmatrix}$$

$$\Gamma(h) = \int_0^h e^{At} B \, dt = \int_0^h \begin{bmatrix} t^2/2 \\ t \\ 1 \end{bmatrix} dt = \begin{bmatrix} t^3/6 \\ t^2/2 \\ t \end{bmatrix} \Bigg|_{t=0}^{h} = \begin{bmatrix} h^3/6 \\ h^2/2 \\ h \end{bmatrix}$$

7. Assume that several independent, periodic control loops should be implemented in a single Java program. We will study two different implementations of the controllers.

In the first implementation, each controller is implemented as a single thread according to the following principle:

```
class Regul extends Thread {
  ...
  public void run() {
    setPriority(Thread.MAX_PRIORITY);
    while (true) {
      y = analogIn.get();
      r = refGen.get();
      u = ctrl.calculateOutput(r,y);
      analogOut.set(u);
      ctrl.updateState();
      opCom.setPlotData(r,y,u);
      t = t + h;
      duration = t - System.currentTimeMillis();
      if (duration > 0) {
        try {
          sleep(duration);
        } catch (InterruptedException e) {}
      }
    }
  }
}
```

In the second implementation, each controller is implemented as two threads: one I/O thread and one control thread, according to the following principle:

```
class Regul extends Thread {
  ...
  public void run() {
    setPriority(Thread.NORM_PRIORITY);
    io = new IO(h);
    while (true) {
      y = io.gety();
      r = refGen.get();
      u = ctrl.calculateOutput(r,y);
      io.setu(u);
```

```
      ctrl.updateState();
      opCom.setPlotData(r,y,u);
    }
  }
  ...
}

class IO extends Thread {
  private double y, u;
  private long h;
  ...

  public IO(long h) {
    this.h = h;
    ...
    start();
  }

  public double gety() {
    synchronized (this) {
      wait();
    }
    return y;
  }

  public void setu(double u) {
    this.u = u;
  }

  public void run() {
    setPriority(Thread.MAX_PRIORITY);
    while (true) {
      analogOut.set(u);
      y = analogIn.get();
      synchronized (this) {
        notify();
      }
      t = t + h;
      duration = t - System.currentTimeMillis();
      if (duration > 0) {
        try {
          sleep(duration);
        } catch (InterruptedException e) {}
      }
    }
  }
}
```

a. In which implementation will the input-output latency (control delay) be the longest? Explain! (1 p)

b. In which implementation will the input-output latency jitter be the largest? Explain! (1 p)

c. In which implementation will the sampling interval jitter be the largest? Explain! (1 p)

Solution

a. The input-output latency will be the longest in the second implementation. There, the D/A and the A/D conversions are synchronized which gives a latency of one sample on average. In the first implementation, the output is actuated as soon as possible, and the latency is always shorter than one sample.

b. The jitter in input/output latency will be the largest in the first implementation. There, all actions (A/D, computations, D/A) of all the controllers execute at the same priority. This means that an A/D or D/A conversion of one controller may have to wait for another controller to finish its computations. In the second implementation, A/D and D/A conversions execute at higher priority than the control computations and will therefore be less prone to jitter.

c. In the second implementation it is only the AD and D/A conversion that executes at highest priority. The amount of time that an IO thread can be preempted by another IO thread is smaller than in the first implementation. Hence, the jitter in sampling interval will be larger in the first implementation.

8. A simplified flow of the initial steps of chocolate manufacturing is shown below. Draw a Grafcet for this. (2 p)

- Fill a tank with beans upto 50 percent of the tank volume and separately another tank with 30 percent sugar
- Fill the tank with beans with water upto 70 percent of the tank volume
- Stir the beans at a low velocity for 10 minutes
- Empty the dirty water from the beans tankreduce the level in the tank to 50 percent
- During the stirring and emptying, the sugar has to be maintained at a high temperature of 190
- Pour the sugar into the beans tank
- Heat the sugar and bean mixture at a low temperature for 15 minutes while stirring at high velocity.
- Remove the beans and sugar mixture from the tank

You have access to two tanks. Use tank 1 for chocolate beans and tank 2 for sugar. All of the sensors and signals below can be accessed using subscripts. (LEVEL_1 = Level of tank 1. LEVEL_2 = level of tank 2)

Use the following signals:

- **SENSOR READINGS:**
  - LEVEL: The current volume filled in the tank
  - TEMP: Gives temperature of the tank in C
- **INPUTS TO THE TANKS:**

- FILLBEANS: To pour beans into the tank
- DRAINBEANS: To remove beans from the tank
- WATERIN: To pour water into the tank
- WATEROUT: To drain water from the tank
- AGITATOR: Stirrer. Set to 0.5 for low velocity and 1 for high velocity
- HEAT: Heats the tank. Set to 0.5 for low temperature and 1 for high temperature

SPECIAL INPUT ONLY TO TANK 2:

- MIXSUGAR: To pour sugar from tank 2 into the tank 1

Solution

One possible solution is given in Figure 5

9. You friend Carl owns a forest which is populated by foxes and rabbits. He likes very much the animals that live in the forest and wants to have both species. For this reason he started looking at them and made some observations that repeated on a day-to-day basis:

- every couple of rabbits gives birth to ("procreates") a new rabbit every day,
- every fox eats two rabbits per day,
- the number of rabbits does not affect the number of foxes,
- one pair of foxes procreates every second day.

Carl is afraid that the foxes will eat the rabbits quicker than they can procreate or that the rabbits will overpopulate the forest. For this reason he plans to take away some foxes every day to keep the animal population in the forest balanced. But he does not know how many to remove.

You think a little bit about it and you convince yourself that you can help him using control theory. You therefore make the following assumptions:

- you neglect the age of the animals (i.e. all the new rabbits will be immediatly able to procreate and the new foxes will be able to hunt the rabbits by themselves),
- you neglect the sex of the animals (i.e. no need to distiguish males and females: if there are 10 rabbits then it is like having 5 couples of rabbits),
- you neglect the possibility that one of the species completely disappears (in fact if you control everything properly this wont happen),
- you neglect the presence of other animals or environmental factors that might interact with rabbits and foxes.

a. Show how to model the number of rabbits and foxes as a discrete-time state-space model. The model should also include the possibility of removing the foxes. You are asked to define the state variables, the control action, the matrices Φ and Γ and the sampling time. (3 p)

b. For the obtained model design a state-feedback controller $u(k) = K(x_{ref} - x(k))$ such that the number of foxes and rabbits converges to the desired value x_{ref}. Place both the poles of the controller in 0.5 (1 p)

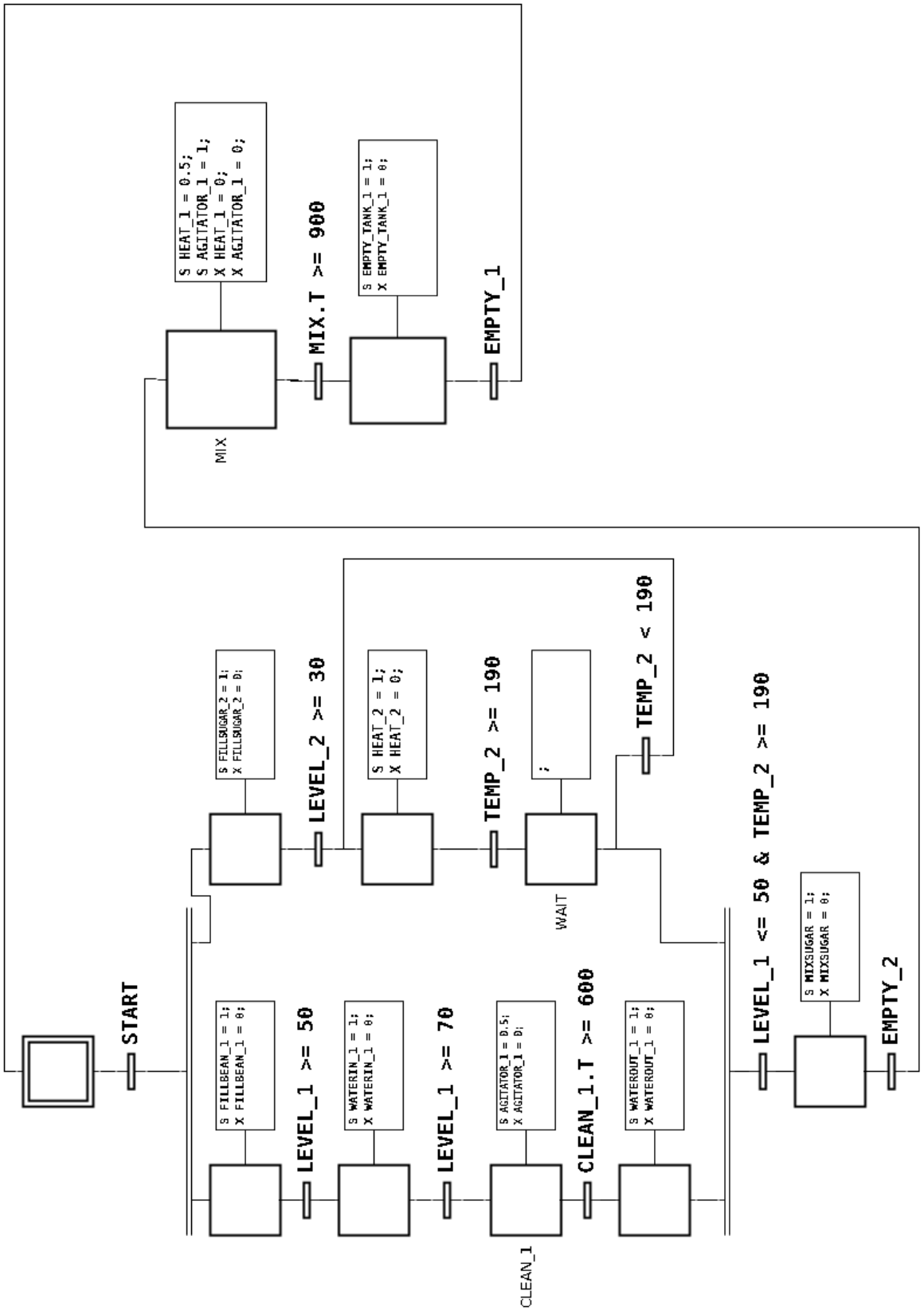

Figure 5 Solution for Chocolate Factory

c. Despite all the assumptions, what is still **not** captured in the model (and solution) you have retrieved in terms of the values that the variables can take? (To get full points you should define this in control terms). (1.5 p)

Solution

a. The sampling time of the system is one day. x_1 is the number of rabbits, x_2 is the number of foxes. The reasoning behind the model is as follows: the number of rabbits at $k+1$ is the number of rabbits plus half of it (since every couple procreates), minus 2 times the number of foxes (since each fox eats two rabbits). The number of foxes at $k+1$ is the number of foxes at k plus a quarter of the number of foxes (since one couple procreates every second day). The control action u is the possiblity of removing foxes so it is $\Gamma = [0, -1]'$. The state space model then looks like this:

$$x_1(k+1) = 1.5x_1(k) - 2x_2(k)$$
$$x_2(k+1) = 1.25x_2(k) - u(k)$$

b. Given the control law $u(k) = -k_1 * x_1(x) - k_2 * x_2(k)$ we get the following closed loop dynamics matrix:

$$\Phi_{cl} = \begin{pmatrix} 1,5 & -2 \\ 1,25 + k_1 & +k_2 \end{pmatrix} \tag{2}$$

with charactaristic polynomial:

$$det(zI - \Phi_{cl}) = z^2 + (-1.5 - k_2) * z + 2k_1 + 1.5k_2 + 2.5 \tag{3}$$

we impose two poles in 0.5, hence the characteristic polynomial $z^2 + 0.25 * z + 1$. Obtaining $k_2 = 1,75$ and $k_1 = 0,1875$.

c. The model and solution retrieved do not capture the fact that the states and the control action have to be integers, hence they are quantized. Also the control action saturates and can be only positive (or negative if gamma is defined as $[0, 1]'$)