# EL2450 Homework 2

Andreas Fröderberg - 19880730-7577
Martin Favre - 19920130-0010

# 1 Rate Monotonic scheduling

**Task 1**
In Rate Monotonic scheduling, each task is given a fixed priority based on its task rate, shorter task rate equals higher priority. The task with the highest priority is then run at all times.

**Task 2**
Calculating the utilization factor U from

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} = \frac{6}{20} + \frac{6}{29} + \frac{6}{35} = 0.68 \tag{1}$$

To be schedulable, the condition

$$U \leq n(2^{\frac{1}{n}} - 1), \tag{2}$$

where $n$ is the number of processes. Calculating for this example with 3 tasks, the condition becomes

$$0.68 \leq 3(2^{\frac{1}{3}} - 1) = 0.78 \tag{3}$$

which holds, therefore the taskset is schedulable with RM scheduling.

**Task 3**
All pendulums are asymptotically stable and have similar control performance. The performance is shown in Figure 1.

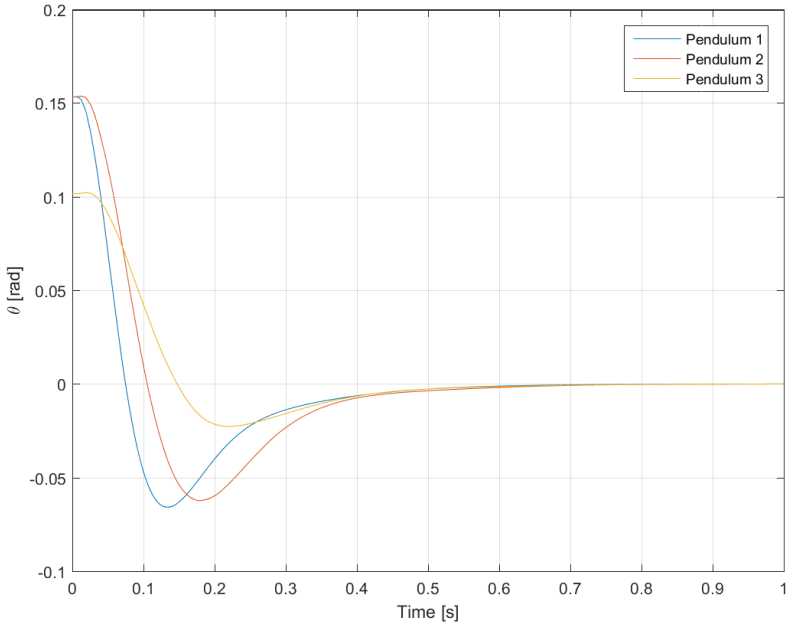


Figure 1: Performance of pendulums under rate monotonic scheduling.

**Task 4**

In Figure 2, the execution of tasks in Matlab is shown.

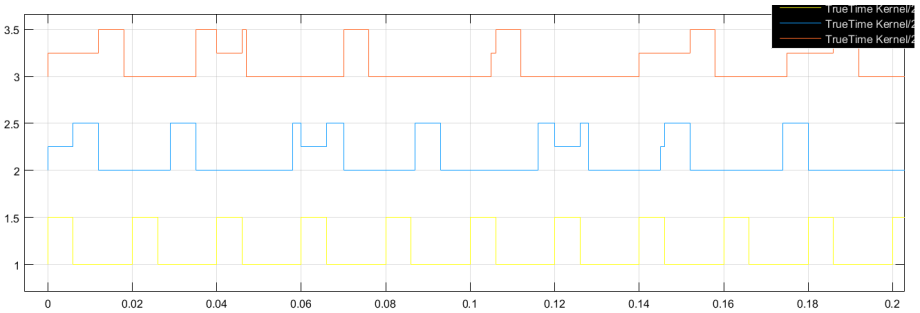


Figure 2: Schedule for pendulums when computation time of all is 6 ms. Yellow is small pendulum, blue is medium and red is big.

In the figure below, Figure 3, the execution order as calculated by hand is shown. It can be seen that the schedules both are consistent.

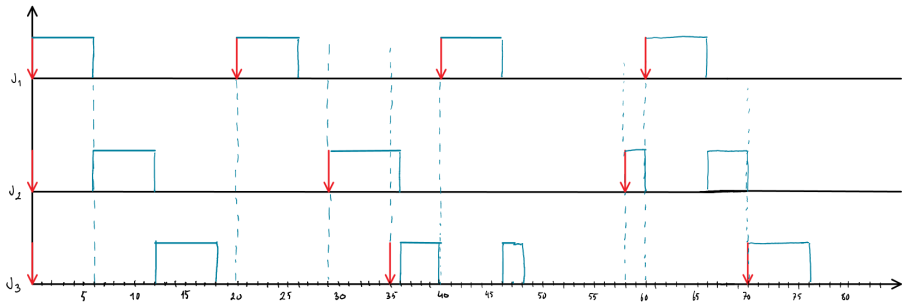


Figure 3: Hand drawn and calculated execution order with Rate Monotonic scheduling. Task $J_1$ represents the small pendulum and rising index represents bigger pendulums.

**Task 5**

For this execution time, the CPU utilization factor becomes

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} = \frac{10}{20} + \frac{10}{29} + \frac{10}{35} = 1.1. \tag{4}$$

The utilization factor is thus larger than Equation 3 which means that the processes is not schedulable. This is seen in Figure 4 where the small pendulum is no longer stable because of the fact that the deadlines are missed.

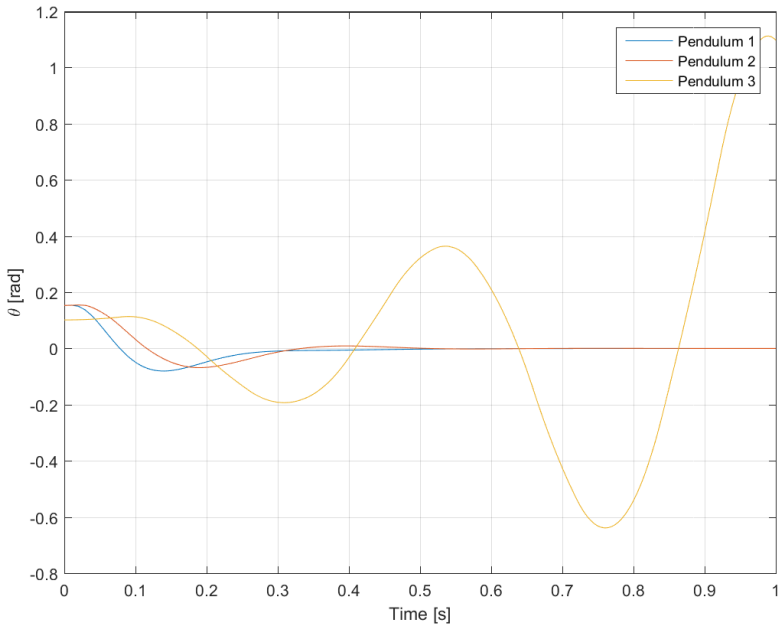


Figure 4: Performance of pendulums under rate monotonic schedulingi with computation time 10 ms for each task.

The schedule in the simulations is shown in Figure 5.

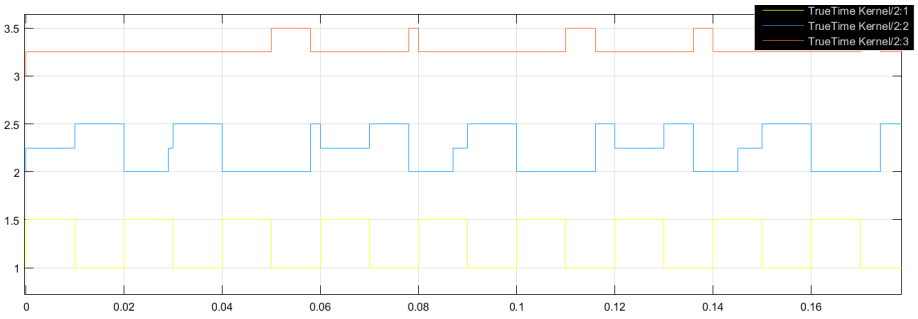


Figure 5: Schedule for Rate Monotonic scheduling when computation time is 10 ms

When viewing the hand-drawn and the real schedule, the following are shown.

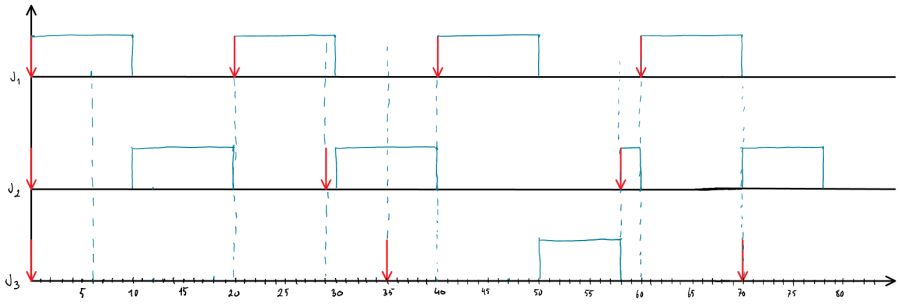

Figure 6: Schedule when computation time is 10 ms. Task $J_1$ is smallest pendulum, higher index represents larger pendulum.

## 2 Earliest Deadline First, EDF

**Task 1**

In EDF, the task that has the closest deadline to the current time is always run. The advantage of this is that the processor can be fully utilized if the the scheme is schedulable. Another advantage is that tasks will not get starved if the processor is overloaded and deadlines are missed since the dynamic prioritization will handle allow all tasks to get processing time. One disadvantage is that tasks can not be given predefined priorities which might be important depending on the implementation.

**Task 2**

EDF is schedulable as long as CPU utilization is lower than 100 %. The utilization factor is the same as the answer q2 from RM, 0.68. Since $U = 0.68 \leq 1$ holds, the task set is schedulable with EDF. The hand calculated schedule is shown below in Figure 7.

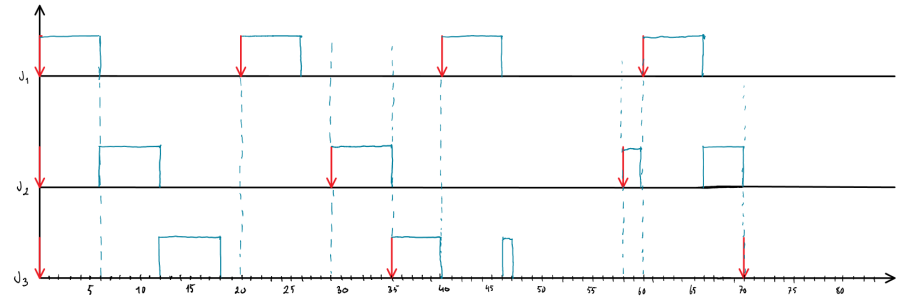

Figure 7: Schdule for EDF with 6 ms calculation time. Task $J_1$ represents the smallest pendulum, higher index represents larger pendulums.

**Task 3**

All pendulums are asymptotically stable and have similar control performance. The performance is shown in Figure 1.

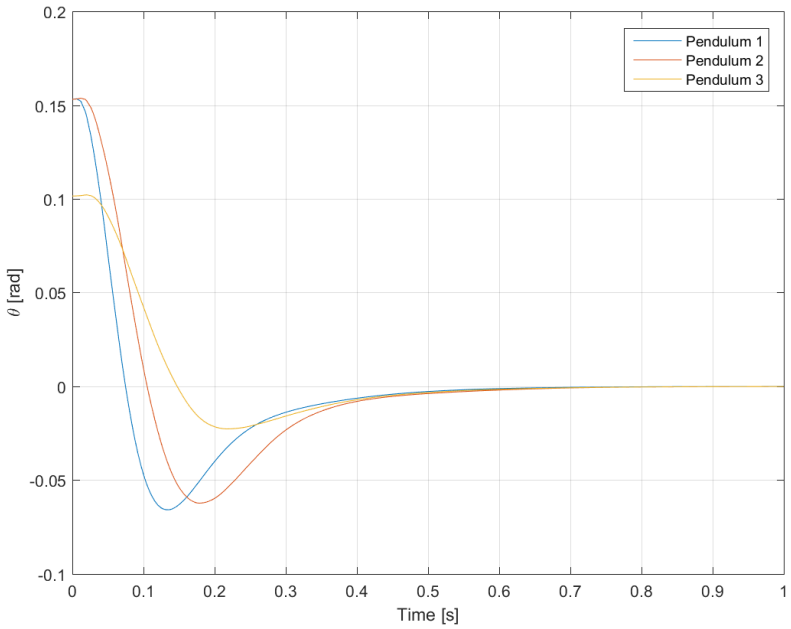


Figure 8: Performance of pendulums under earliest deadline first scheduling with computation time 6 ms for each task.

**Task 4**

The schedule in the model is shown in Figure 9.

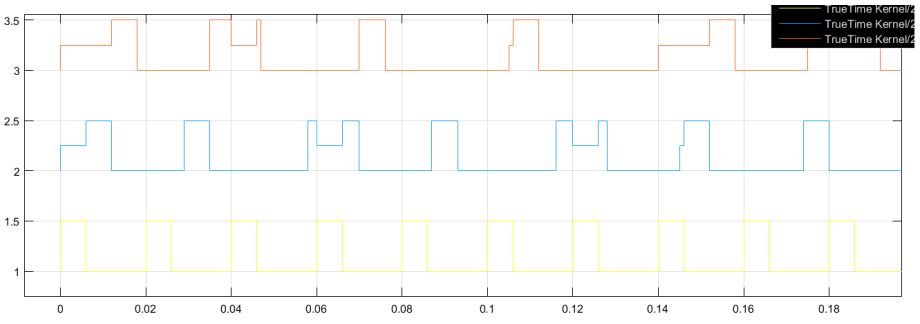


Figure 9: Performance of pendulums under earliest deadline first scheduling with computation time 6 ms for each task.

It is consistent with the schedule presented in task 2.

**Task 5**

For this execution time, the CPU utilization factor becomes

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} = \frac{10}{20} + \frac{10}{29} + \frac{10}{35} = 1.1. \tag{5}$$

The utilization factor is thus larger than one which means that the processes is not schedulable.

However looking at Figure 10 the system is still stable although with deteriorated control performace for all three pendulums. This is a result of the disability to meet all the deadlines but still meeting enough for the system to stabilize.

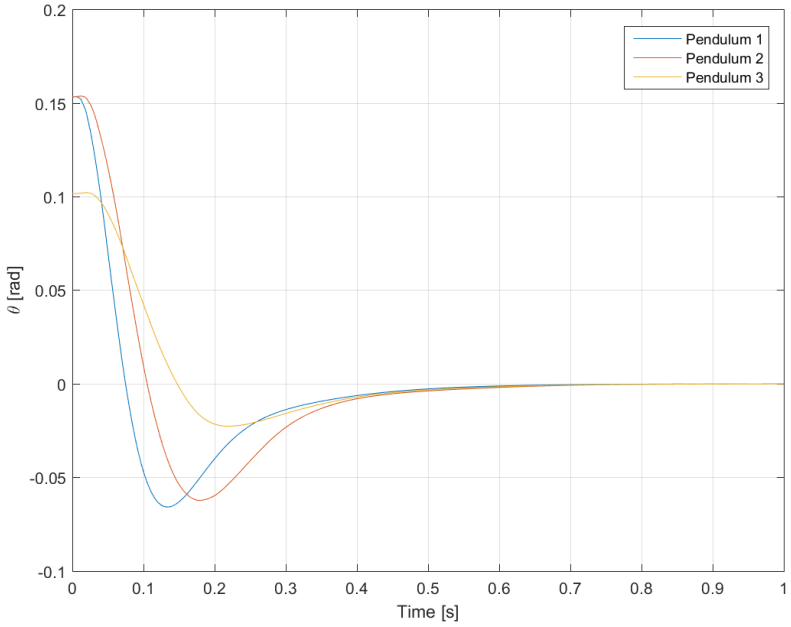


Figure 10: Performance of pendulums under earliest deadline first scheduling with computation time 10 ms for each task.

**Task 6**

Comparing the systems with 6 $ms$ execution time in Figure 1 and Figure 8 little difference can be seen. However with execution time of 10 $ms$ in Figure 4 and Figure 10 the control performance is much better with EDF-scheduling.

A relation can be drawn with the different utilization factors of EDF and RM, where RM is limited to a factor of 0.68 from Q2 whilst EDF can handle 1.0 and is therefore better.

# 3 Networked Control Systems

**Task 1**
With

$$\dot{x}(t) = Iu(t) \tag{6}$$

$$y(t) = Cx(t) \tag{7}$$

and

$$u(kh) = -Kx(kh). \tag{8}$$

Sampling the system with a period h gives:

$$\Phi = e^{Ah} \tag{9}$$

$$\Gamma_0(\tau_A) = I \int_0^{h-\tau_k} \mathrm{d}s = (h - \tau_k)I \tag{10}$$

$$\Gamma_1(\tau_A) = I \int_{h-\tau_k}^{h} \mathrm{d}s = \tau_k I \tag{11}$$

Defining $z(kh) = [x^T(kh), u^T((k-1)h]^T$ as the augumented state vector, the augmented closed-loop system is

$$z((k+1)h) = \widetilde{\phi}(k)z(kh) \tag{12}$$

where

$$\widetilde{\phi}(k) = \begin{bmatrix} \Phi - \Gamma_0(\tau_k)K & \Gamma_1(\tau_k) \\ -K & 0 \end{bmatrix} = \begin{bmatrix} 1 - (h - \tau_A)IK & \tau_A I \\ -K & 0 \end{bmatrix}. \tag{13}$$

Which results in:

$$z((k+1)h) = \begin{bmatrix} 1 - (h - \tau_A)I & \tau_A I \\ -K & 0 \end{bmatrix} z(kh). \tag{14}$$

**Task 2**
If the poles of the system are given by the equation

$$0 = z^2 + a_1 z + a_2, \tag{15}$$

where (15) is the denominator of the pulse-transfer function, then the sytem is stable if the poles $p_1$ and $p_2$ are confined within the unit circle. According to the Jury criterion, this is fulfilled when

$$a_2 < 1, \tag{16}$$

$$a_2 > -1 + a_1 \tag{17}$$

and

$$a_2 > -1 - a_1 \tag{18}$$

are fulfilled. The poles of (14) are the eigenvalues of $\widetilde{\phi}$, as given by the equation

$$0 = \det(\lambda I - \widetilde{\phi}). \tag{19}$$

Applying (19) for (14) yields the characteristic equation

$$0 = \lambda^2 + \lambda(KIh - KI\tau - 1) + K\tau I. \tag{20}$$

Comparing (15) and (19) one can identify the parameters

$$a_1 = KIh - KI\tau - 1 \tag{21}$$

and

$$a_2 = K\tau I. \tag{22}$$

Combining (17) and (18) yields the criterion

$$|a_1| < 1 + a_2. \tag{23}$$

Finally, the stability criterion becomes

$$\max\left\{\frac{1}{2} - \frac{1}{KIh}, 0\right\} < \frac{\tau}{h} < \min\left\{\frac{1}{KIh}, 1\right\}. \tag{24}$$

Plotting the region results in Figure 11

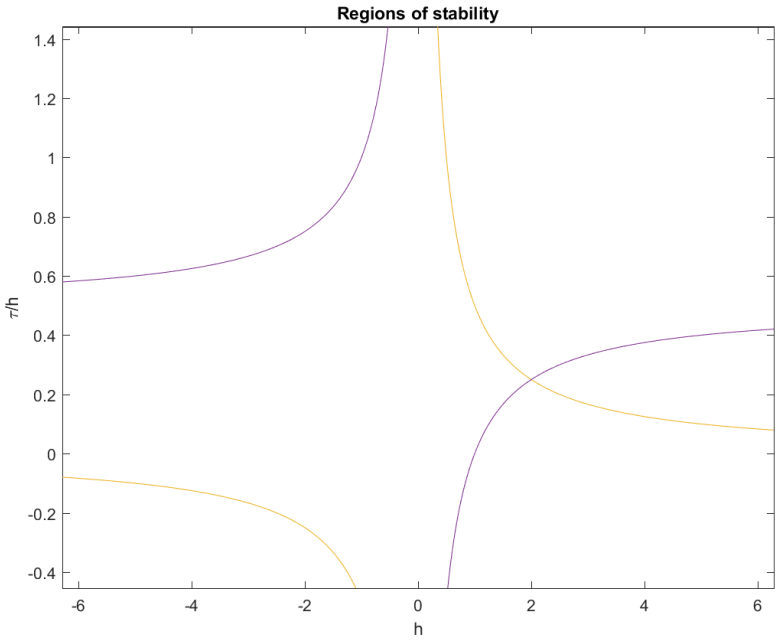


Figure 11: The stability regions.

**Task 3**

In Figure 12 the system is simulated with three different time delays. The system becomes unstable approximately above a delay of 0.04s.

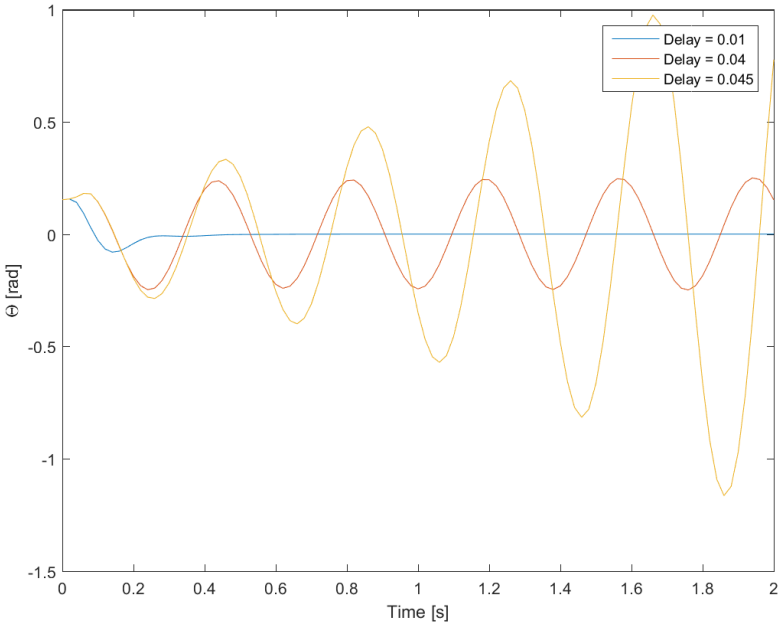


Figure 12: Statechart of the three parallel systems.

# 4 Discrete Event Systems

### Task 1

In Figure 13 the three systems can be seen.

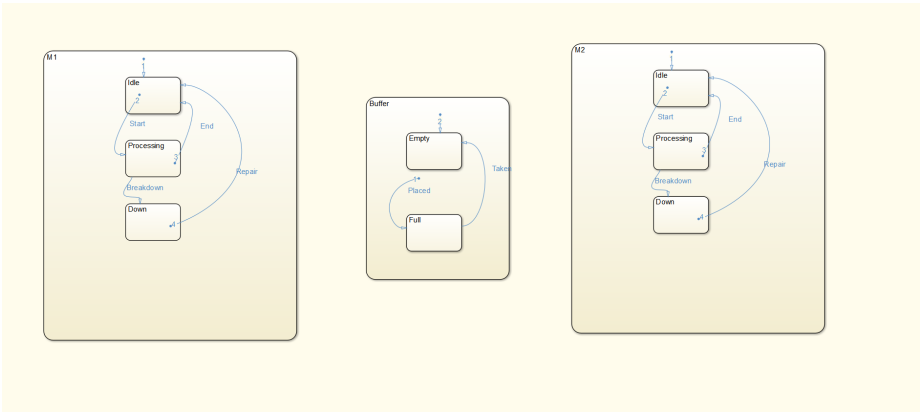


Figure 13: Statechart of the three parallel systems.

### Task 2

In Figure 14 the full statechart can be seen. As all behaviours were not defined assumptions were made that only one state may change at a time, and therefore $PROCESSING$ triggers $FULL$ without changing its state.

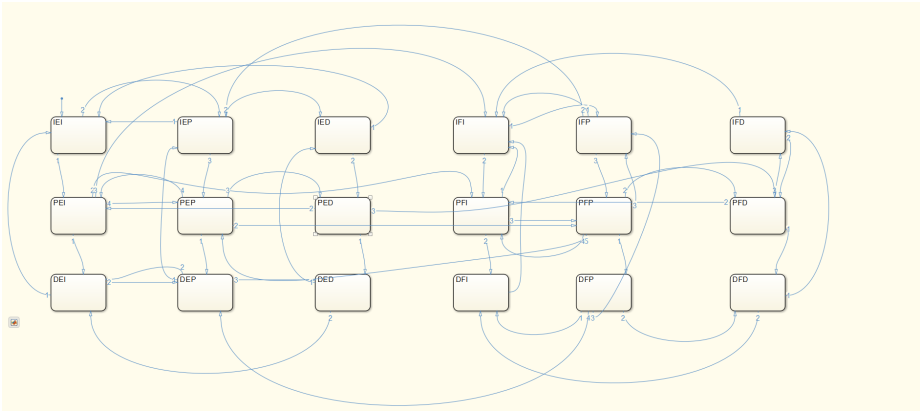


Figure 14: Full statechart of all possible events and transition.

### Task 3

In Figure 15 the full statechart with the added rules can be seen.

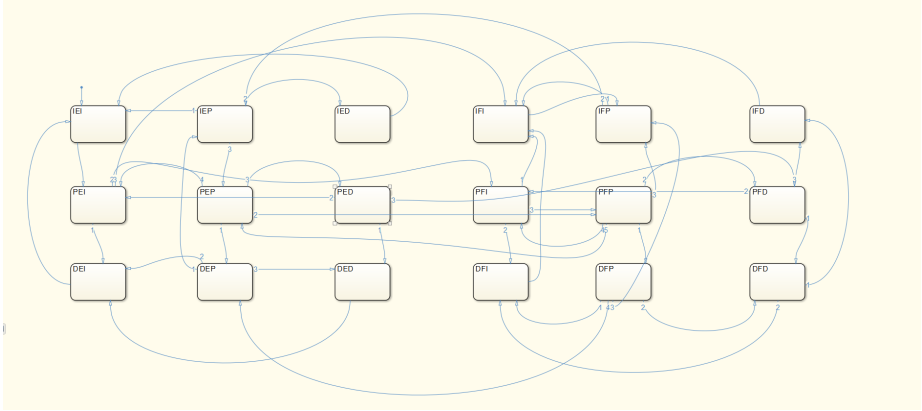


Figure 15: Statechart of all possible events and transition with added rules.

### Task 4

Add two additional parallel pair of states with the states $START\ ENABLED$ and $START\ DISABLED$ for the first and $REPAIR\ ENABLED$ and $REPAIR\ DISABLED$ for the second.

Add guards on all transitions initiated by the events $START$ and $REPAIR$ which requires the associated parallel statemachine to be in its enabled state.