Homework Assignment

Title: **Coffee Machine**

Supervisor: **Bence Graics**

The modeling task

Construct the model of the following problem using the extended timed automaton formalism of the UPPAAL tool!

Form Ltd. wants to release a new coffee machine to the market and created the following system design. The coffee machine consists of three distinct modules: a **controller**, a **cup dispenser** and a **liquid dispenser** (responsible for the coffee and milk). The three modules can be in off state (this is the initial state for all modules) or in operating state (consisting of additional substates) where they operate according to their functionality. All three modules get to idle state after getting turned on to operate. Each module can be turned off only in its idle state.

The **controller** controls the turning on and turning off of the other modules: when the coffee machine is turned on or off, the controller (while changing its state) notifies the other modules, which change their state between off and idle state accordingly. In the idle state, the controller can issue two kinds of commands: it can command the preparation of an *espresso* or a *cappuccino*; in both cases, it notifies the other modules. The other modules work in parallel. After issuing a command, the controller sets a variable to false, which denotes whether the ordered coffee is ready. The controller waits until another module notifies it that (from the other module's point of view) the coffee is ready. In this case, the controller sets the variable referring to the state of the ordered coffee to true and goes to the idle state. Furthermore, the controller is responsible for the "registration" of the number of cups stored in the coffee machine: if the controller and the cup dispenser are both in idle state (the controller verifies this), then cups can be placed into the coffee machine, 10 at a time. The number of available cups in the coffee machine is stored in a variable. At most 40 cups can be inside the coffee machine at the same time. Initially, there are 10 cups in the coffee machine.

The **cup dispenser** in idle state waits for the arrival of an espresso or a cappuccino command. When a command arrives, it examines whether there are enough (at least one) cups in the coffee machine and if so, it prepares a cup, which takes 3 time units. After the preparation of the cup, it subtracts one from the number of cups available in the coffee machine and indicates that the cup is ready using a variable. Then, it waits until another module indicates that the coffee is ready. After that, it sets the variable indicating the readiness of the cup to false and goes to the idle state.

The **liquid dispenser** in idle state waits for the arrival of an espresso or a cappuccino command. When an espresso command arrives, the module grinds the coffee beans (this takes 10 time units) and pours the coffee out through the opening (this takes 3 time units). Then, it notifies all the other modules that the coffee is ready and goes to the idle state. When a cappuccino command arrives, the module pours out milk through the opening (this takes 4 time units) after the grinding of the coffee beans (10 time units), but before pouring the coffee (3 time units).

According to an engineer, this design is not correct, and the system will not meet the requirements. He suggests that there should be a mechanism in the controller that *turns off the coffee machine after the last available cup is used up and does not allow the coffee machine to be turned on again until it is filled up with some cups.*

The verification task

Using temporal logic expressions and model checking, verify the fulfillment of the following requirements! Examine the requirements both in the case of the *original* and the *extended* system (proposed by the engineer)! If a requirement is not met, explain the reason of the violation and (using simple methods) fix or simplify the design, and based on that, the system model!

1. There is no deadlock in the system.
2. Every module is in off state and operating state exactly at the same time. (There is no such state in the system where one module is in off state and another one is in operating state.)
3. The liquid dispenser does not pour liquid (coffee or milk) out through the opening if there is no prepared cup.
4. Every espresso and cappuccino command is carried out; the system does not get stuck while a command is being carried out.
5. If the coffee machine runs out of cups, then there will eventually be a refill.

Hints

The notifications and commands between the modules are worth modeling by using synchronization channels. Consider broadcast channels too!

Note that it is allowed to submit the homework without the successful verification of all requirements, but this is not a solution of full value (i.e., the grade of the homework will be less than 5).