Embedded systems: Assignment 3

Sjoerd van der Heijden
10336001

November 11, 2018

Contents

1 Introduction

In this document I will show and explain my solutions to the tasks detailed in Assignment 3 of the course Embedded Systems 2018, so buckle up.

2 Assignment 3.1

For the first part of this assignment I was tasked to create a Petri net model of a four-way intersection for cars. On this intersection cars are only allowed to drive straight, and they have to give priority to cars on their right. Fig. 1 shows my solution for this task, with an example initial token distribution. I use inhibitors to prevent cars from driving as long as there's a car coming from their right, or when a car from the left or right is already on the intersection. Cars from opposite directions may cross the intersection simultaniously.

It is possible for the cars to come into deadlock, when a car from each direction is waiting to cross. I did not solve this, as this is not disallowed by the assignment. In order to solve it, one could add another level of priority, e.g. by adding another event that can be fired when cars are waiting from all directions and no cars are crossing, and results in one or two cars crossing the intersection. In the case of two cars crossing they would have to come from opposite directions to prevent collisions.

I also came up with a solution without needing inhibitors, but since it requires a more deliberate setup and is less intuitive I far prefer my current solution.

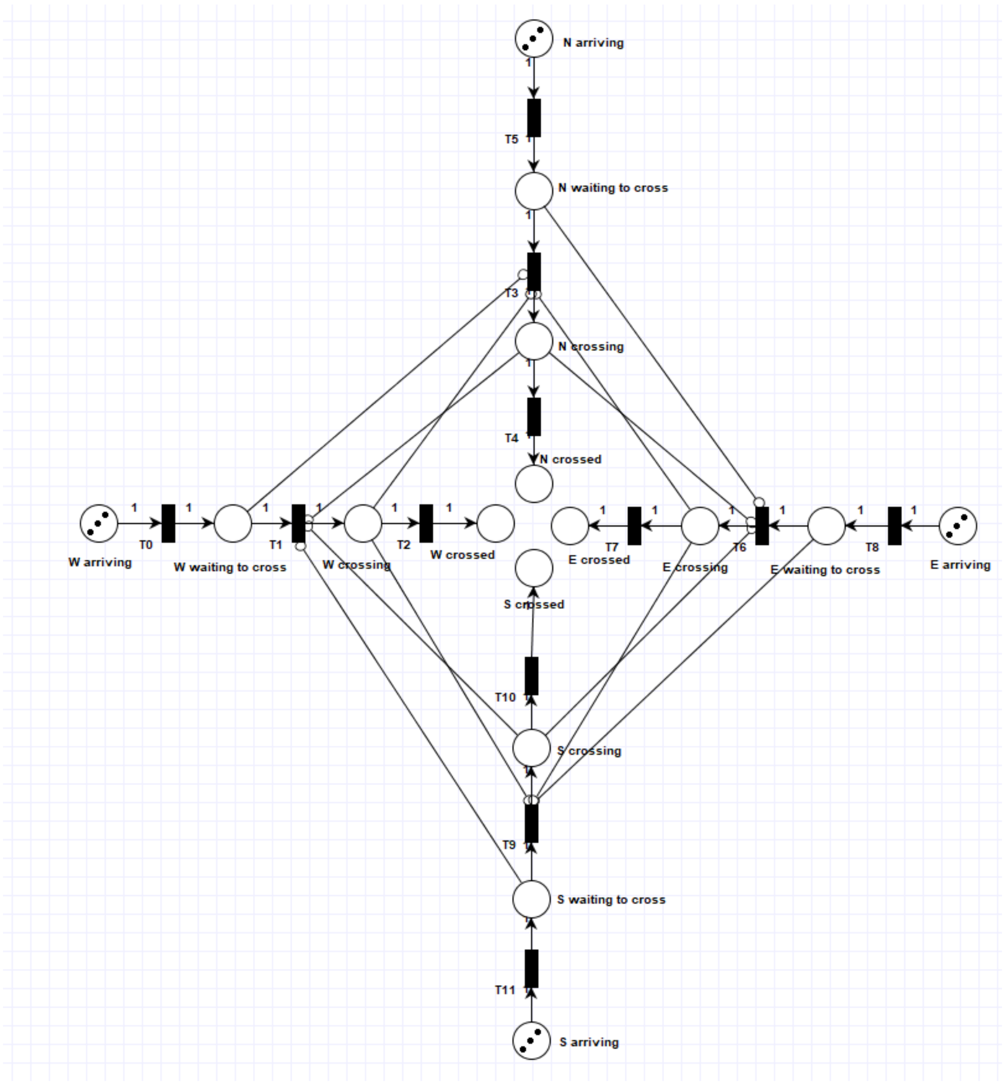

Figure 1: *My solution to assignment 3.1, this image was created using PIPE 5 [1, 2]. The N, E, S and W stand for the cardinal directions. For each direction, a car is first driving towards the intersection (arriving), then reaching the intersection and waiting to cross (waiting), then driving on the intersection (crossing), and finally they leave the intersection (crossed).*

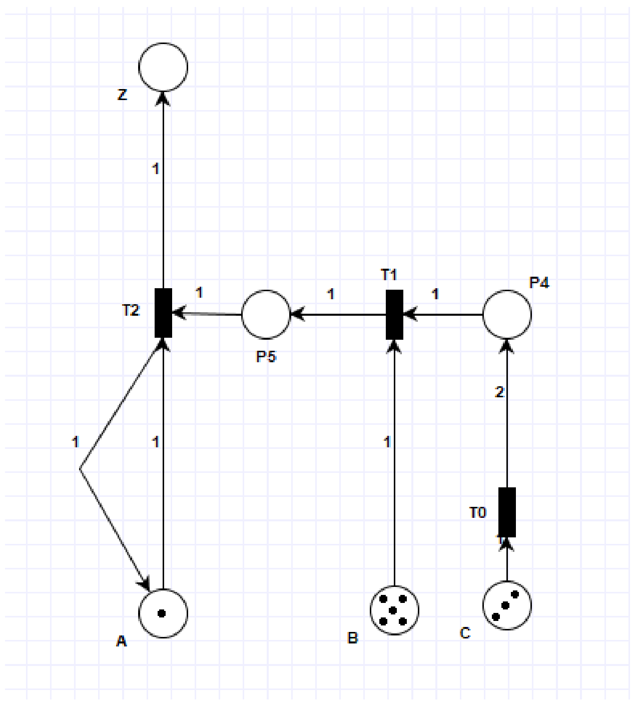

Figure 2: *My solution to assignment 3.2, this image was created using PIPE 5 [1, 2].*

3 Assignment 3.2

The assignment was to compute $Z = A \times \min(B, 2 \times C)$ using Petri nets. My solution is shown in Fig. 2. For the sake of being an example, there are tokens present in the net, so that it shows a possible initial state of the aforementioned equation.

4 Assignment 3.3

I made a chessclock library in Simulink (see Fig. 3 and Fig. 4), since it had to be the same for both players. How it works is that the players can tell the Simulink chart whose turn it is by toggling a switch. Depending on whose turn it is, one or the other clock starts ticking. When the switch is toggled that clock stops, the local time is reset and global time is incremented, and the other clock starts. After 40 turns both players' clocks are incremented by 15 minutes.

The only downside of having the chessclock implemented as library is that the number of moves taken is measured twice (once in the clock of each player). I set the initial global time to $(60 * 30) - 15$ in the chart, because I add 15 seconds to the global time before each turn instead of after. In this way they still start with 30 minutes.

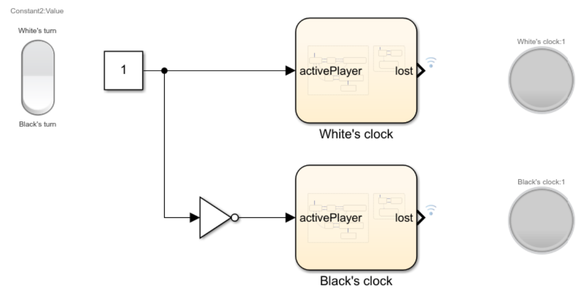

Figure 3: *The outside view of my solution to assignment 3.3: there is one chess clock for each player, the switch tells which player's turn it is, and the lights are green if a player is still playing and red if a player loses. This image was created using PIPE 5 [1, 2].*

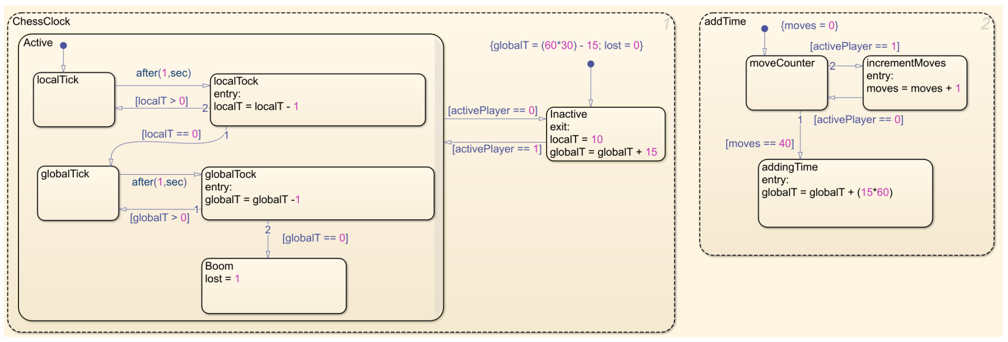

Figure 4: *A peek under the hood of my chessclock for assignment 3.3. On the left is the chess clock itself, on the right the number of turns taken is logged, and time is added to the clock if 40 turns have passed. The clock itself is somewhat simple: you start with 30 minutes on the global clock, which starts ticking after 10 seconds once it is your turn. After every turn you gain 15 seconds and when the global clock reaches 0 you lose. The clock does not tick during the opponent's turn. This image was created using PIPE 5 [1, 2].*

References

[1] Nicholas J Dingle, William J Knottenbelt, and Tamas Suto. Pipe2: a tool for the performance evaluation of generalised stochastic petri nets. *ACM SIGMETRICS Performance Evaluation Review*, 36(4):34–39, 2009.

[2] Pere Bonet, Catalina M Lladó, Ramon Puijaner, and William J Knottenbelt. Pipe v2. 5: A petri net tool for performance modelling. In *Proc. 23rd Latin American Conference on Informatics (CLEI 2007)*, 2007.