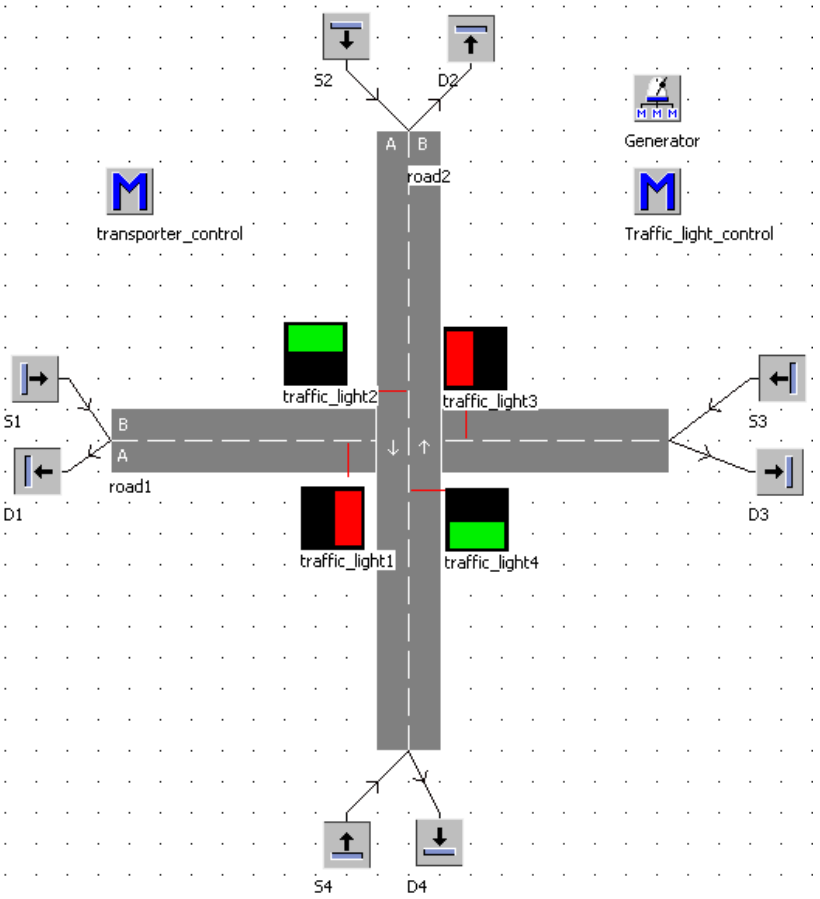
Part II: Intensive Programming

Simulation of Production Systems

Traffic Lights

Definition

You are to simulate a simple crossroads. The crossroads is regulated by traffic lights. In this example, the transporter decides directly at the crossroads, if it stops or goes on (without slow down). All transporters behind it drive against it. Insert a *traffic_light* in the class library (duplicate a class *SingleProc* and rename it). Create two (different) icons in the class *traffic_light* (icon1: green, icon 2: red). Insert in the class *traffic_light* a user-defined attribute "*go*" (data type *boolean*). Create a new frame. Set the scaling factor to 0.25 (Frame window – Tools – Scaling factor). Length of *road1*: 90 m, *road2*: 75.25 m. Set up the following frame:



Figrue1: Model

In this example, we also show how the *TwoLaneTrack* works. Settings: The sources *S1, S2*, and *S3* create each transporter. The interval of the creation should be randomly distributed. Make the following setting in the source *S1*:

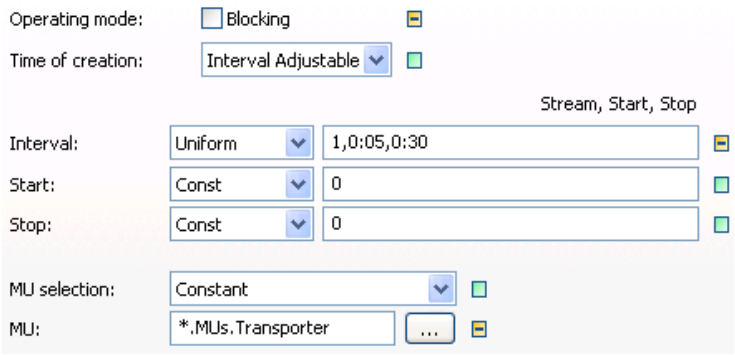


Figure 2: Source Settings

Make this setting also for *S2, S3*, and *S4*. Set the stream (first number in field interval for each source to another value). The transporters move at a speed of 10 meters per second and accelerate with 10 m/s². To ensure that the transporter can pass each other, set in the ways a *Track pitch* of *6* meters.
Create at the crossroads sensors on the lanes. You can specify for each lane its own sensors. For every track, you need to specify two sensors. One sensor in lane A and one sensor in lane B. All sensors will trigger the method transporter_control. In case of road1, it could look as follows:

ID	Position	Front	Rear	L.	Path	C.
1	32m	A			transporter_control	*
2	60m	B			transporter_control	*

In case of road 2:

ID	Position	Front	Rear	L.	Path	C.
1	42m	A			transporter_control	*
2	42m	B			transporter_control	*

Initialize the simulation, so that two traffic lights show the icon 2 (red) (right mouse button – next icon) and the attribute *go* is *false*; the other two traffic lights show the green icon and the attribute *go* has the value *true*.

For the traffic light control, we use in this example a method (*traffic_light_control*) and a generator. The method switches the lights and the generator repeatedly calls the method with an interval of 1:30 minutes.

Tasks:

- **Task 1) Program the method *Traffic light control*!**
  The Traffic light control should change the icon as well as the state (user defined attribute *go*) of the traffic light.
  You can test the method by starting this repeatedly. The lights should "switch". The method is called by the generator. Open the generator by double-click and make the following settings on the tab times:

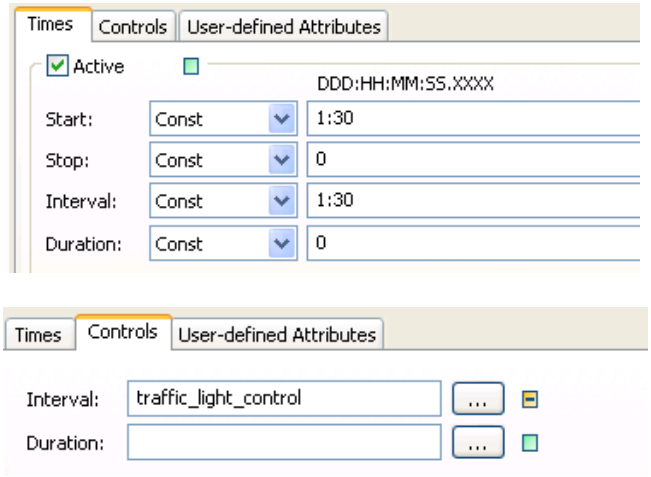


Figure 3: Generator setting

- **Task 2) Program the method *transporter_control*!**
  The transporter should stop when the relevant traffic light has the value *go=false* and wait until go=true. Then the transporter should start again.