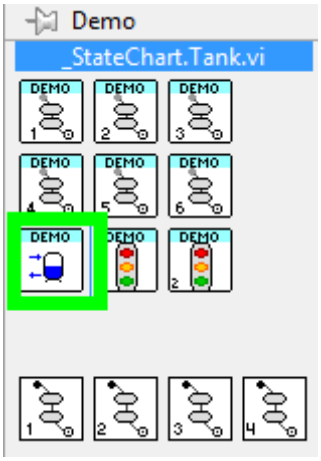
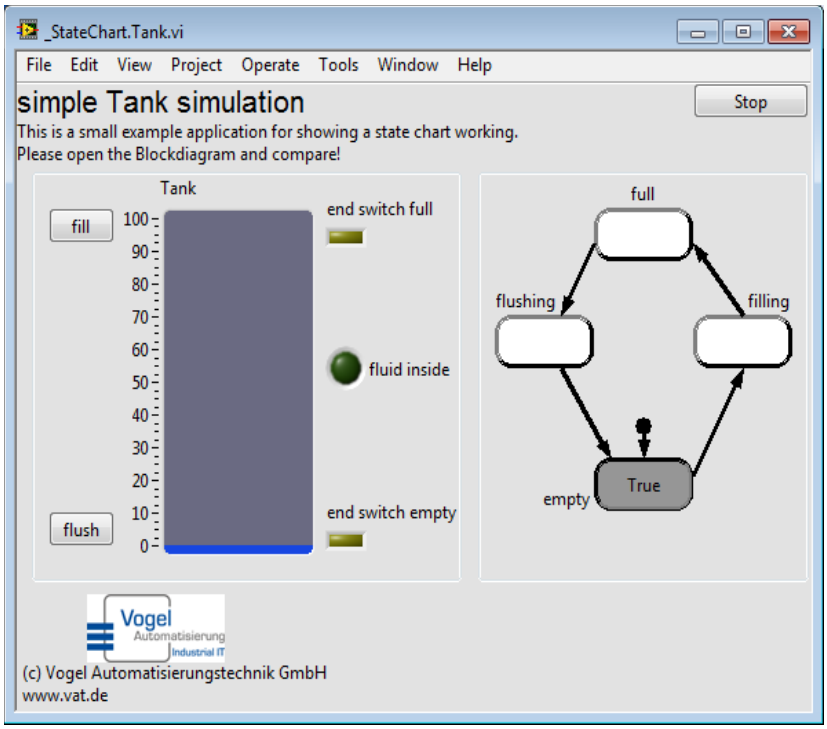
Tutorial openStateChart

This document is intended to show how the features of the openStateChart function library can be used.

The demonstration is based on the example tank. You can find the example on the palette „user libraries/StateChart.“

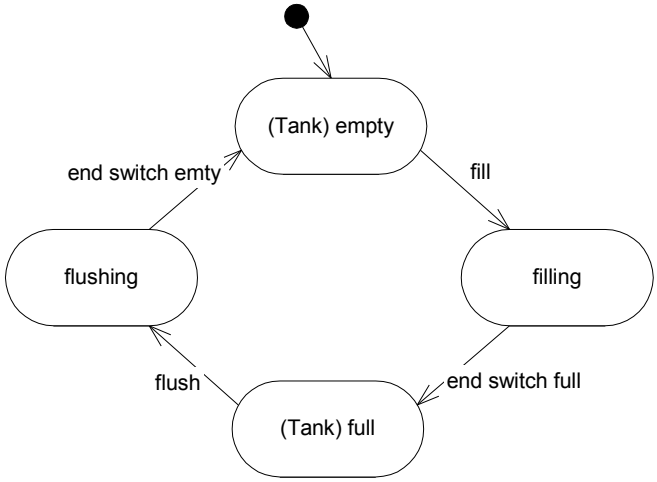




Preview

A very simple case should serve as an illustration.
A tank has an inlet and an outlet and one sensor each for empty and full.

The level of the tank is to commute between full and empty message. The tank should always be completely emptied or filled.
For this purpose, you can design a simple state chart:

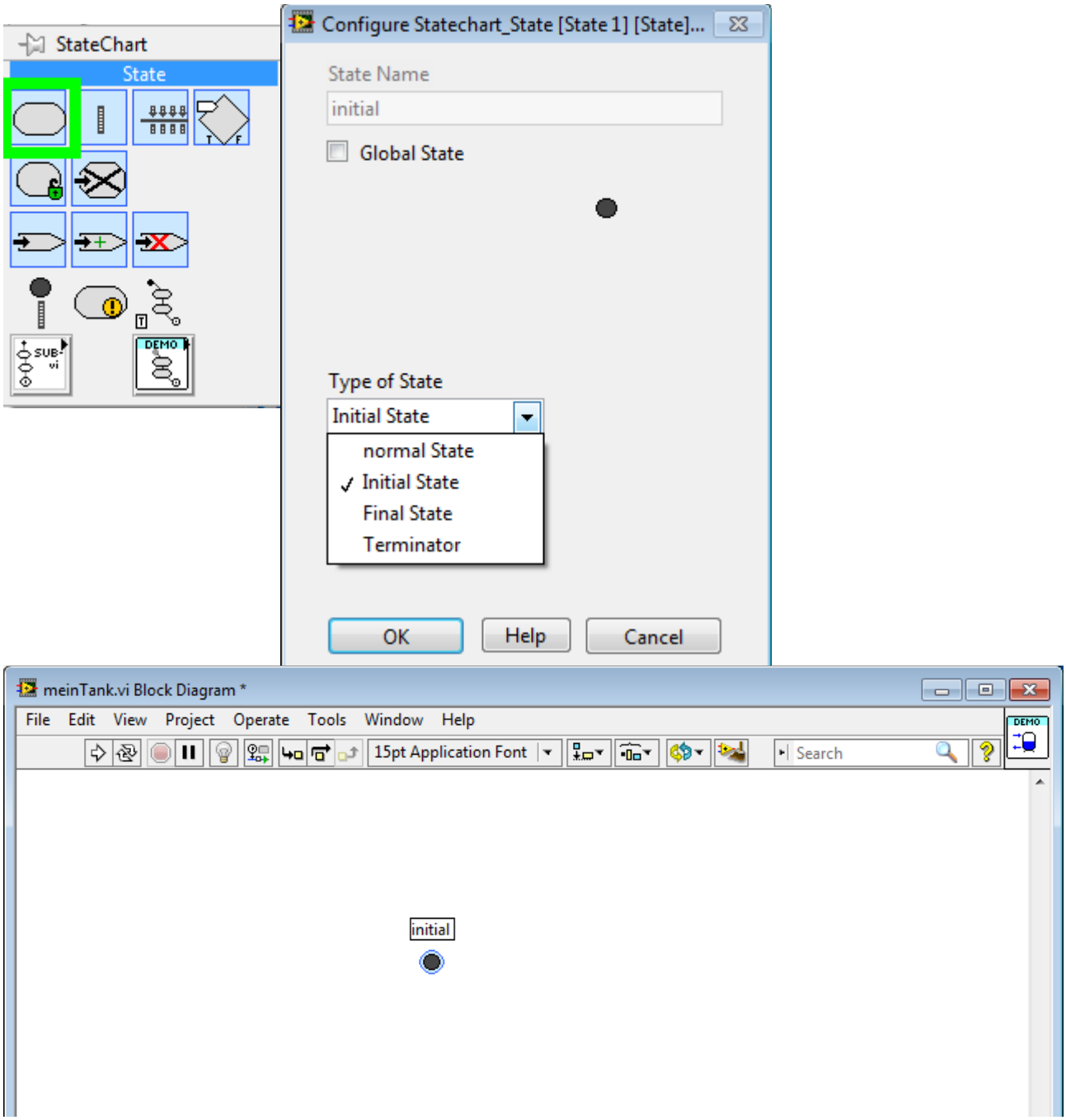


Implementation with openStateChart

First, we place the display elements and functions for the states of the state machine on the VI.

1. Step States

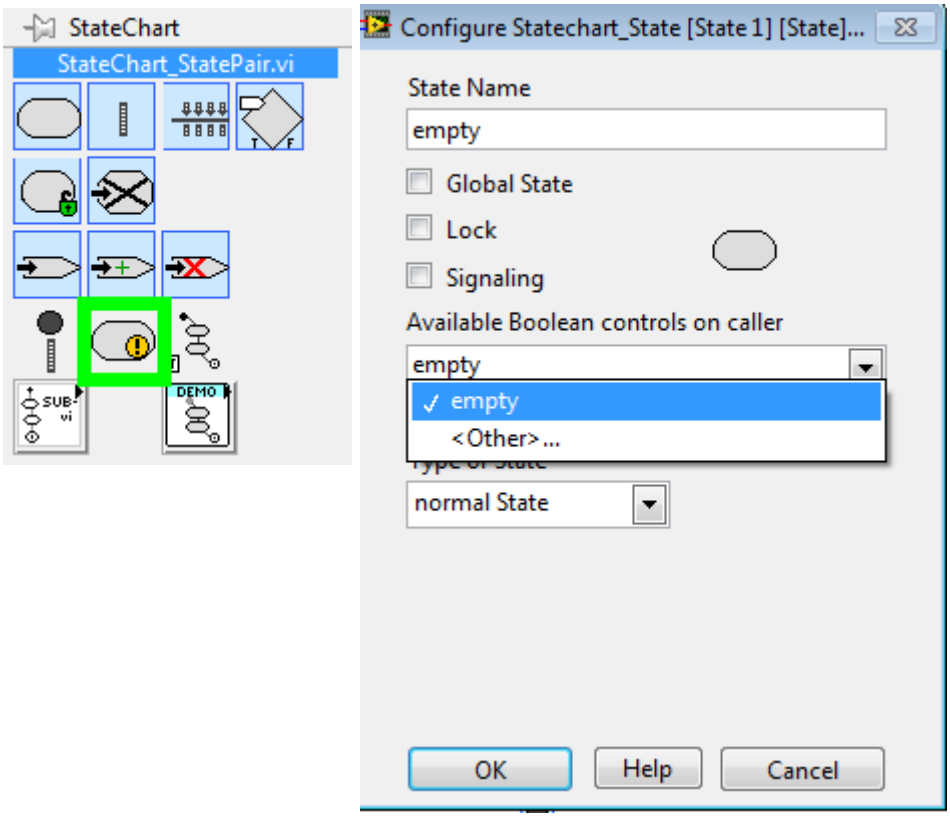
Let's begin with the start state. Place a State VI on the Block Diagram.

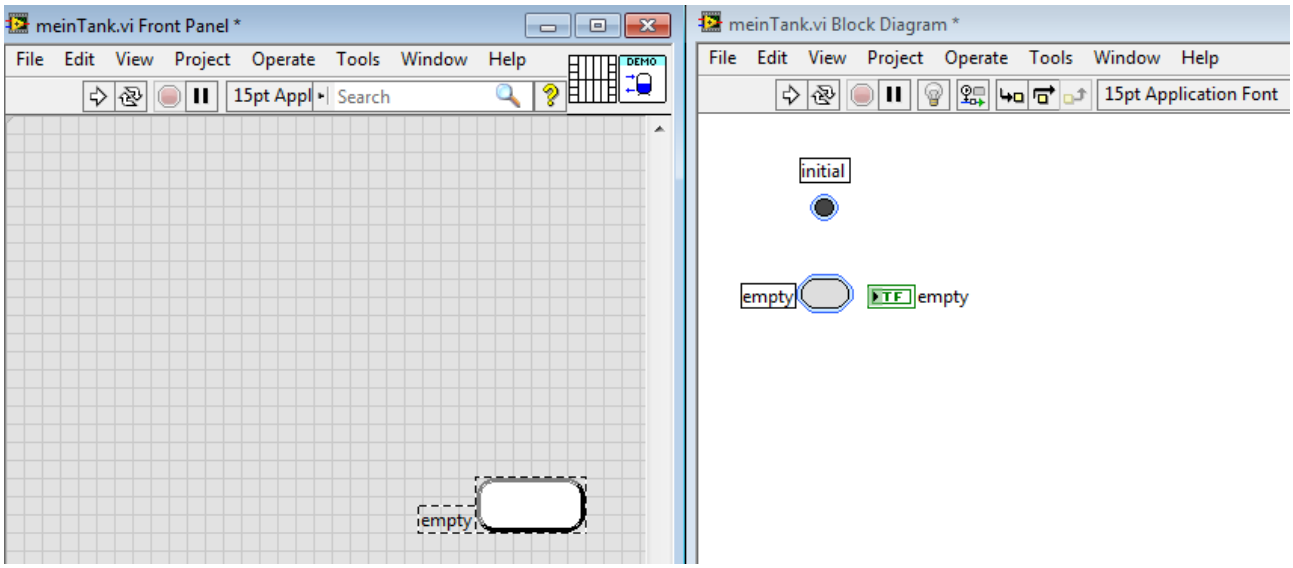


Configure the state as Initial State.

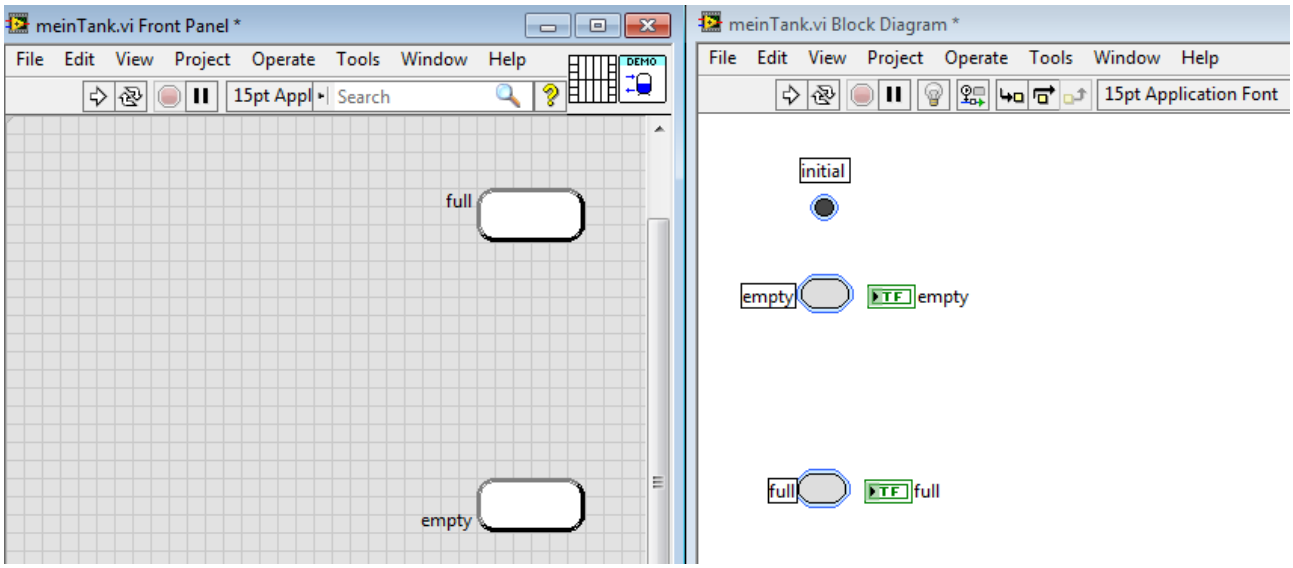
Now add the state "empty". Because we want to have an indicator for the state on the front

panel, we can insert a State Pair. A State Pair is the Combination State VI and Indicator. Name the Indicator to „empty“ and configure the State VI as normal State named „empty“.

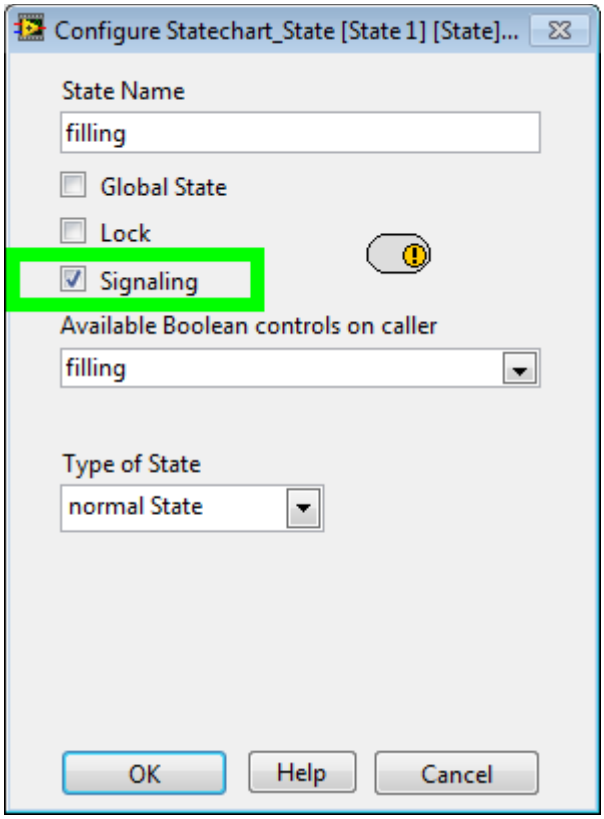


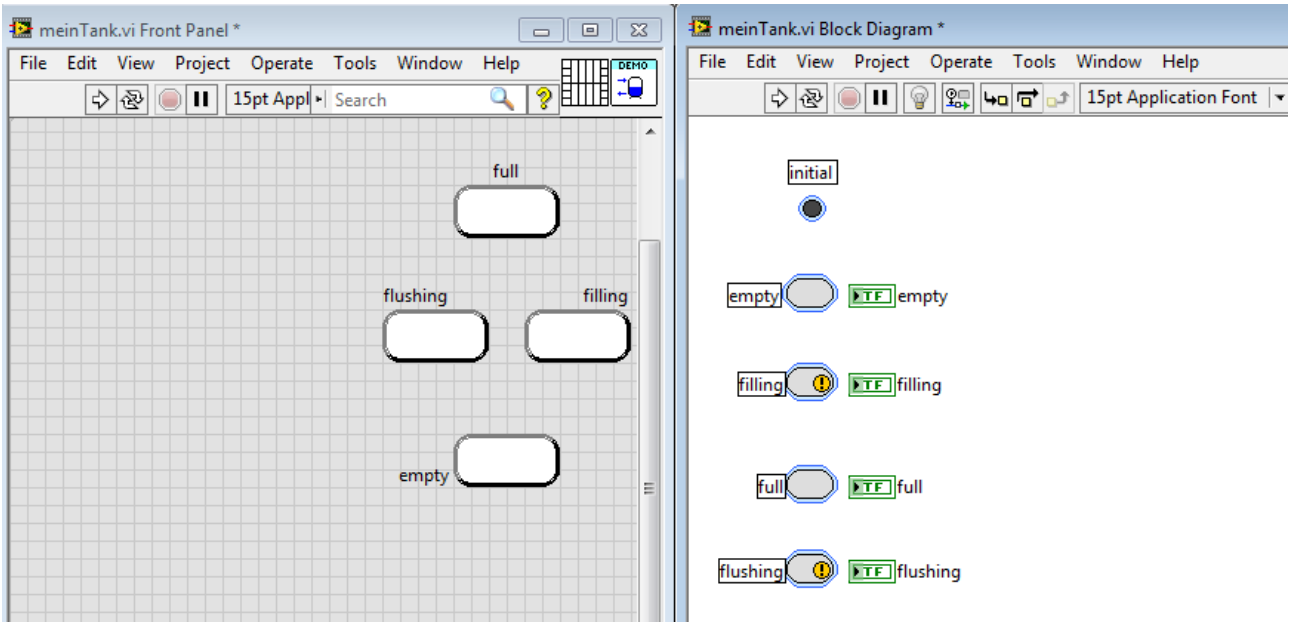


Repeat the process for the state "full".

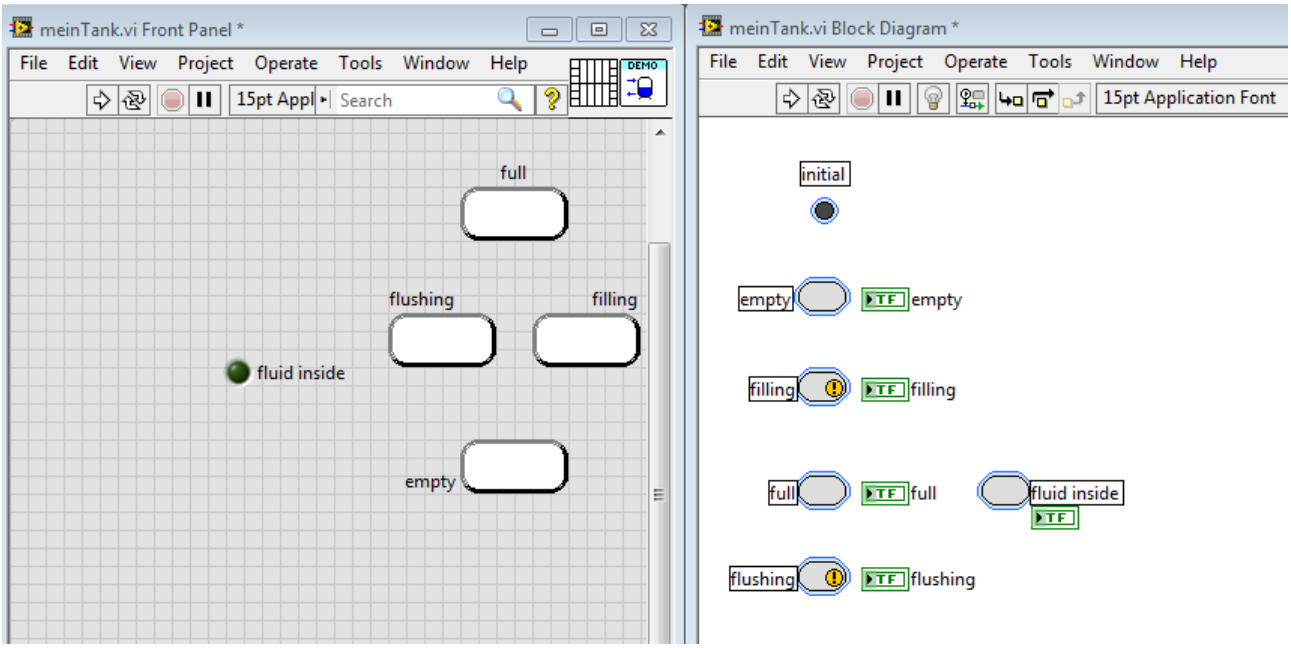


Next states are „filling“ and „flushing“. It should be noted that actions are to be programmed in case of state change for both states. So we want to get an event on state changes. To do this select the box "Signaling".





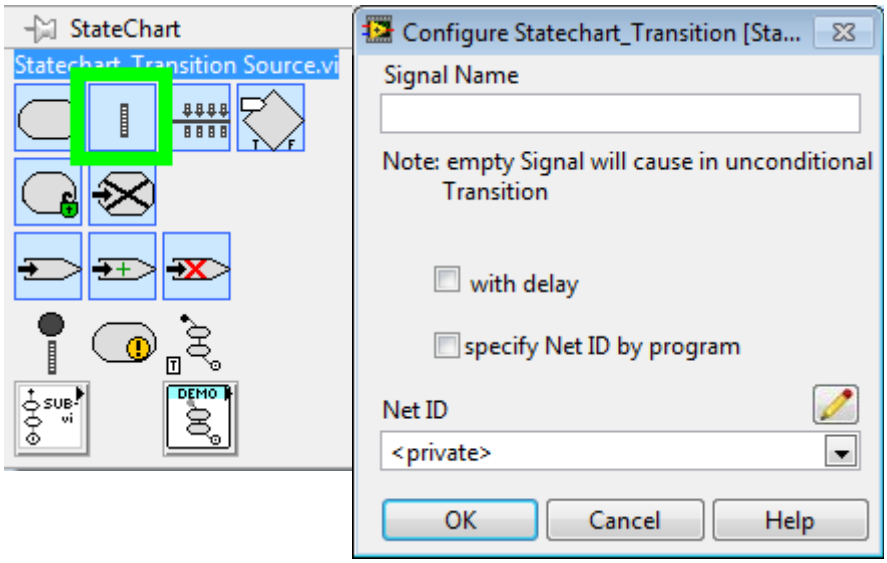
We want to know additionally whether the liquid is in the tank. This we want to display with an additional state "fluid inside".



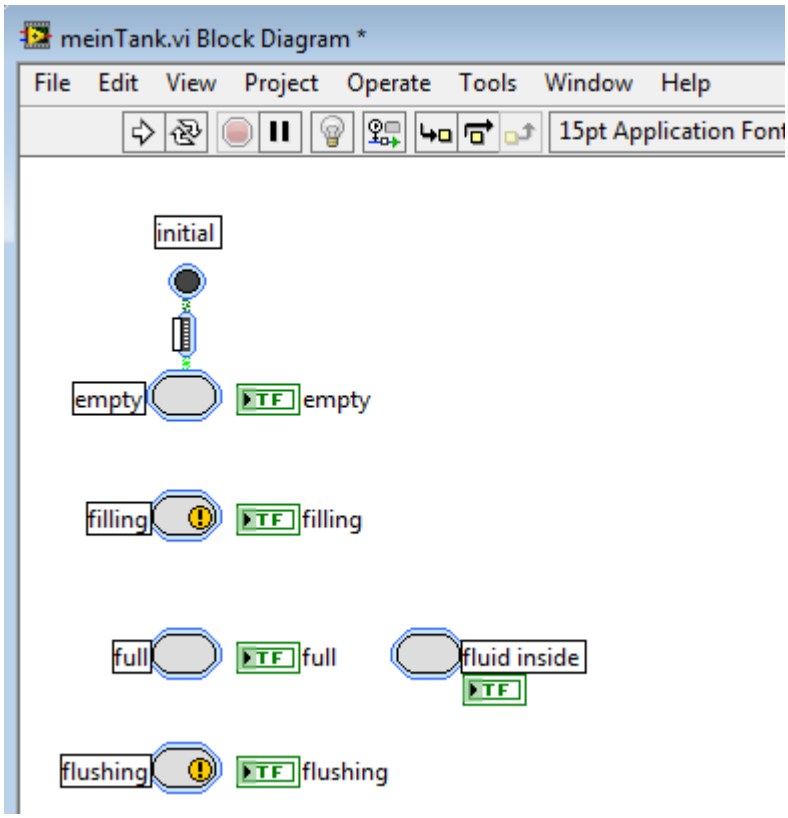
The states for our small application are done.

2. step Transitions

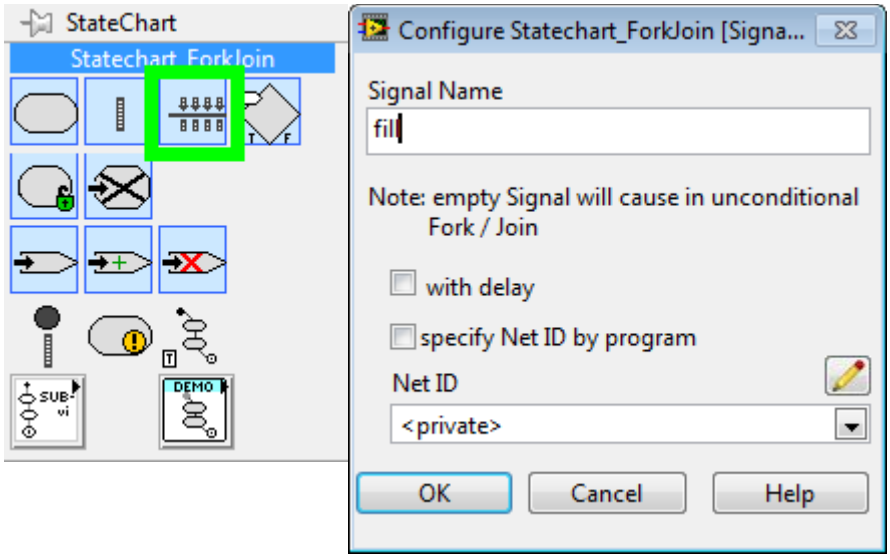
Next, we program the switching conditions for our net. We assume that the tank is empty at system start-up. That's why we connect our start state with the state „empty“, by using a unconditional, immediate switching transition.



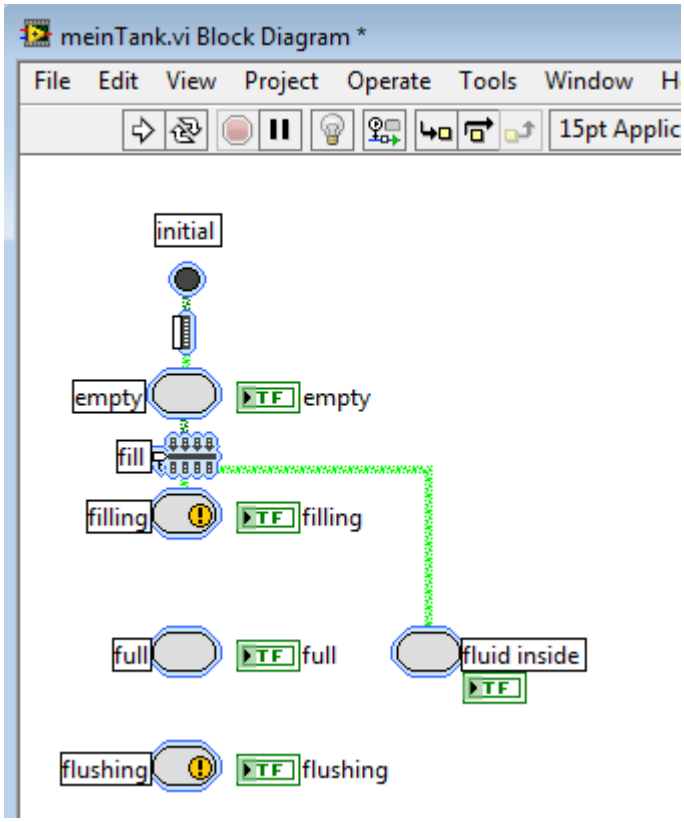
Leave the field „Signal Name“ empty. We want to create a private net. The network name for the global identification is set equal to the name of the instance of the calling VI. - In our case „mein Tank.vi“.
Connect the states „initial“ and „empty“ with the Transition.



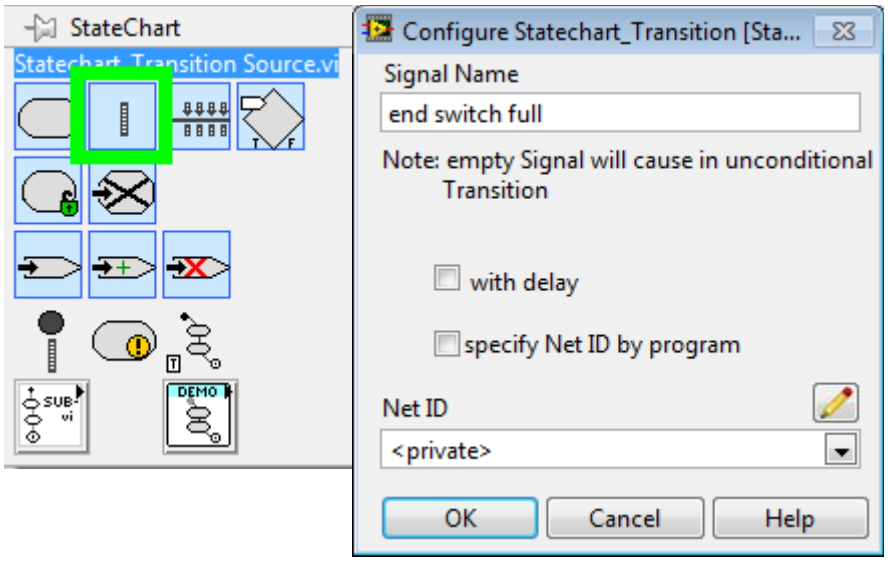
The transition from "empty" to "filling" should only take place when the command "fill" has the value "True". Additionally, the state "fluid inside" should be set. Since the state machine splits here, we need no transition, but a fork.



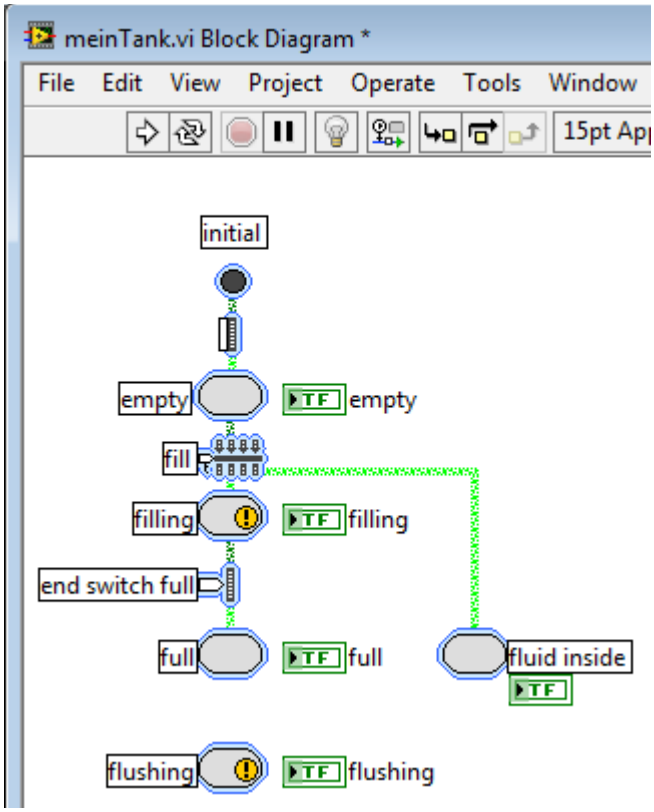
Enter at the configuration dialog the signal name "fill" as a transition condition.
Now you can connect the fork with the states "empty", "filling" and "fluid inside". The choice of the predecessor or successor connection is free.



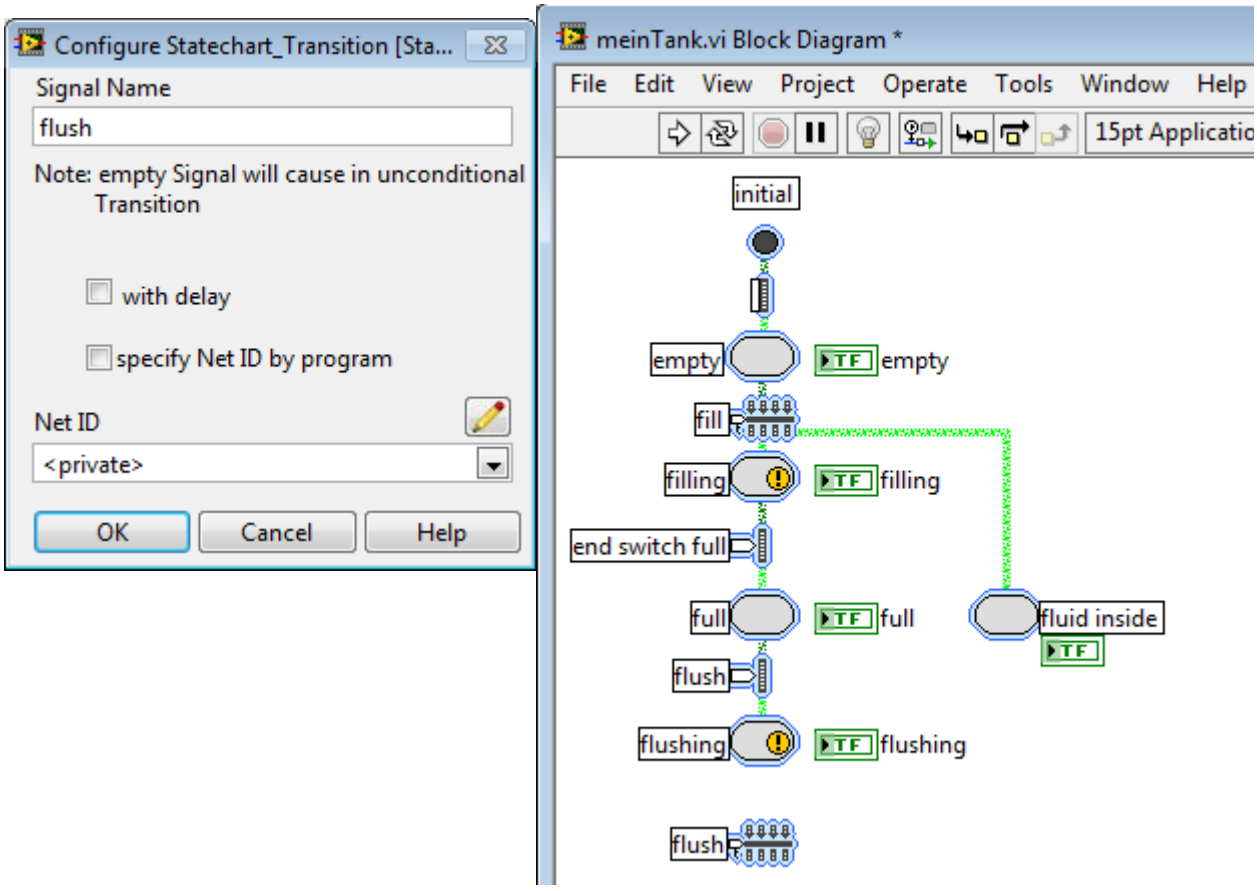
We want to stop filling the tank when the limit switch "end switch full" reaches the value "True". For this purpose we use a conditional transition.
Enter at the configuration dialog of the transition the signal name "end switch full" as transition condition.



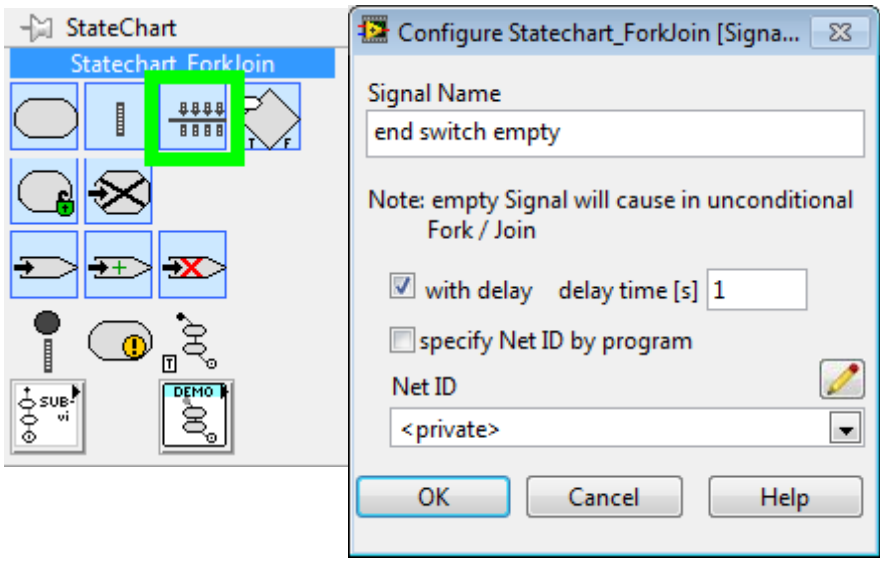
Now you can connect the transition to the states "filling" and "full".



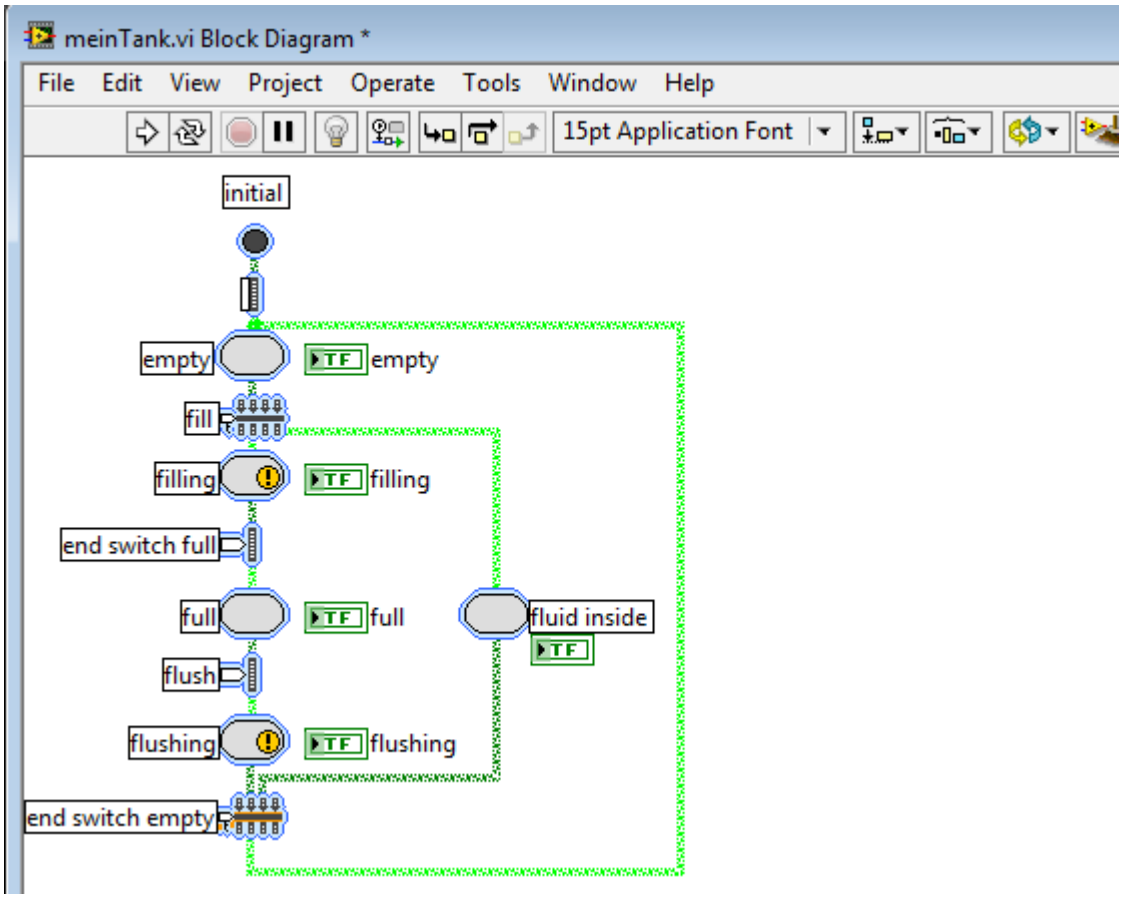
The transition from "full" to "flushing" will occur if the "flush" command has the value "True".



We want to stop emptying the tank when the limit switch "end switch empty" reaches the value "True". Additionally, the state "fluid inside" is to be reset. Because the state machine unites here, we will need no transition, but an join.



Enter at the configuration dialog the signal name "end switch empty" as a transition condition. In addition, you can insert a delay time for the transition condition to empty the tank completely. (Here for example 1s)
Now you can connect the Join with the states "flushing", "empty" and "fluid inside".



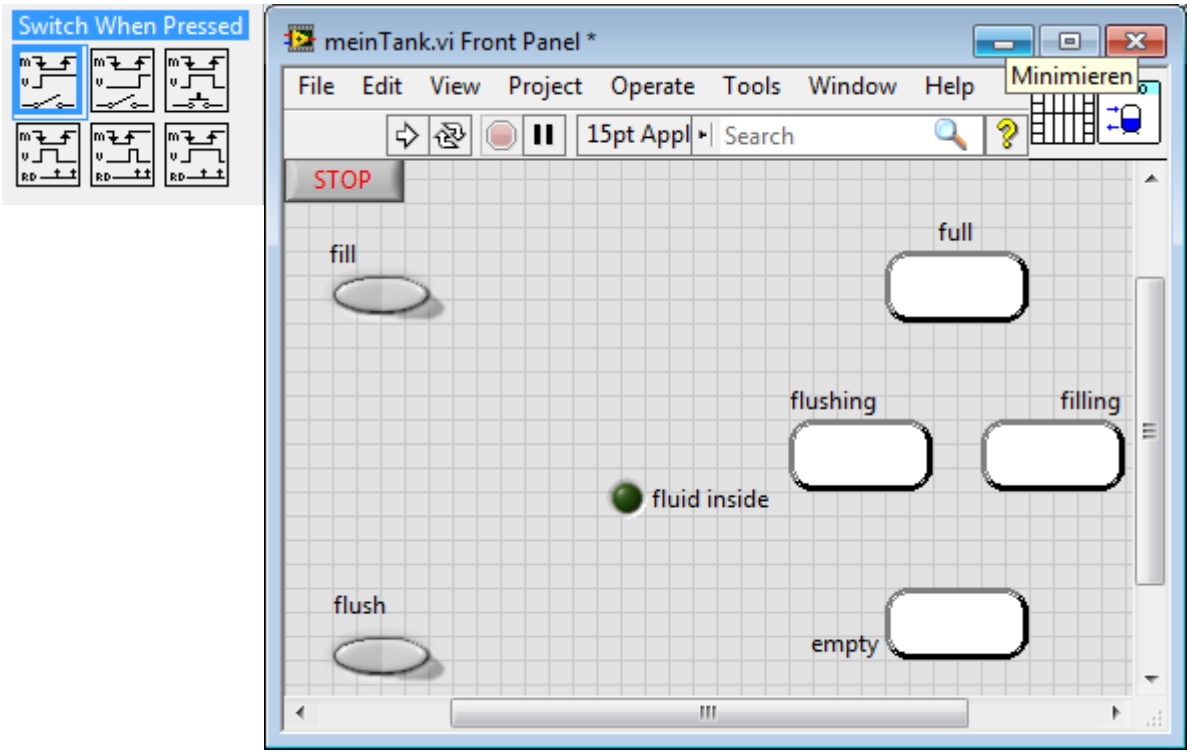
Our state machine is finished at this point.

Set the switching conditions

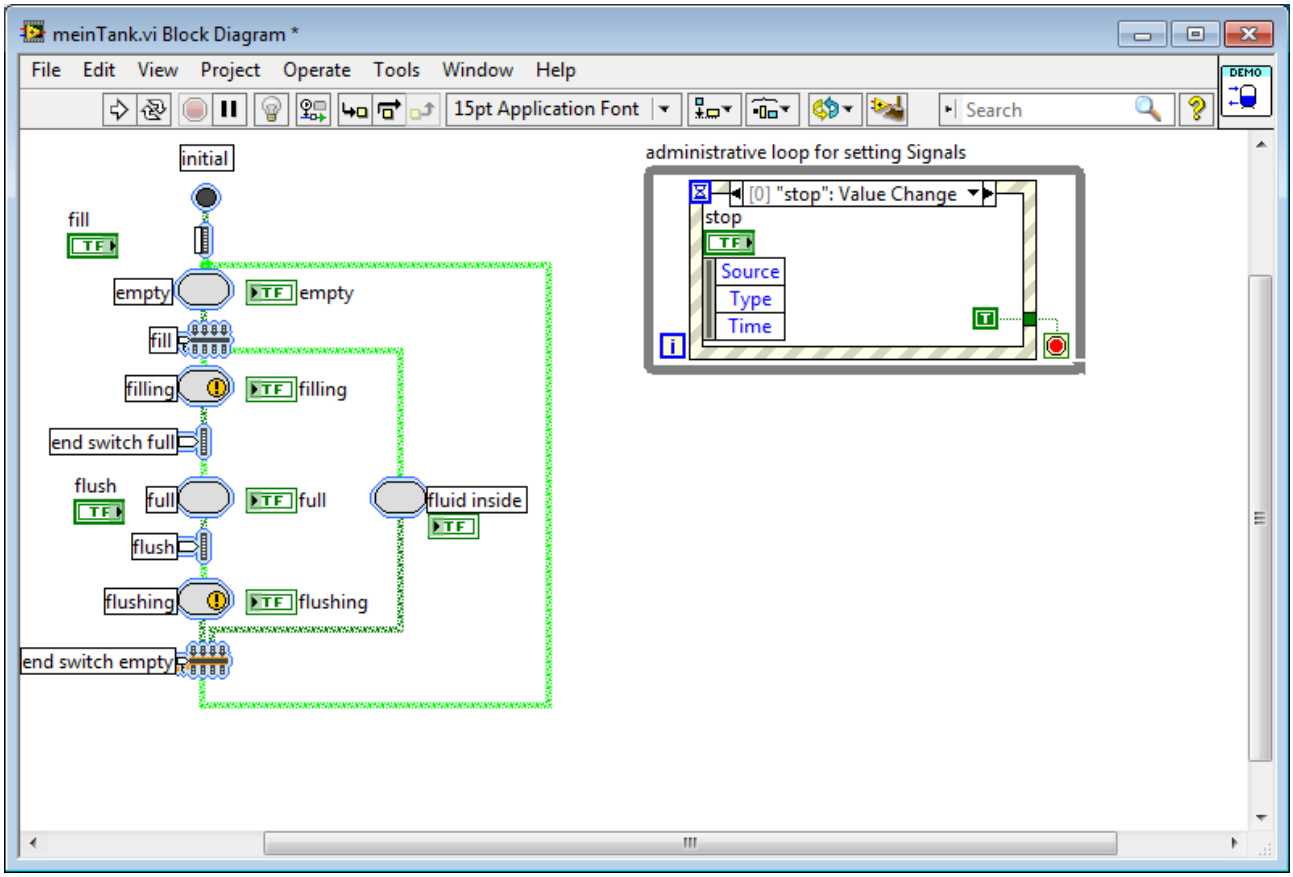
The state diagram is already ready to work. However, it still lacks the setting of the switching conditions and in the absence of a real tank, a simulation.

Now create Controls for switching commands "fill", "flush" and exit the sample program.

Change the Mechanical Action of all switching elements to "Switch When Pressed"

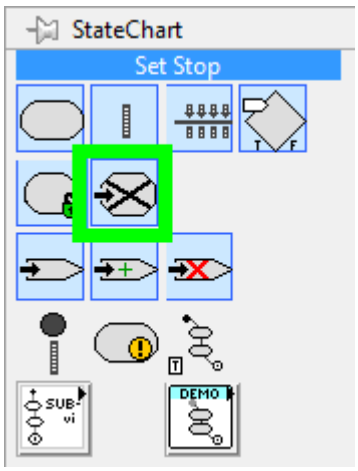


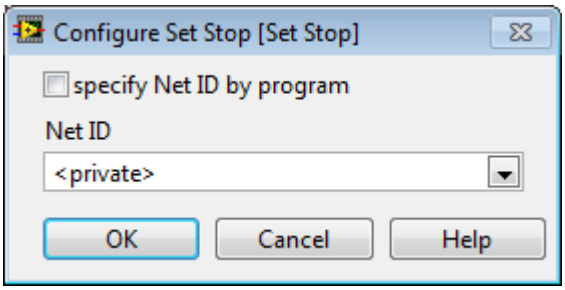
For the handling of the keys we need a "event structure"

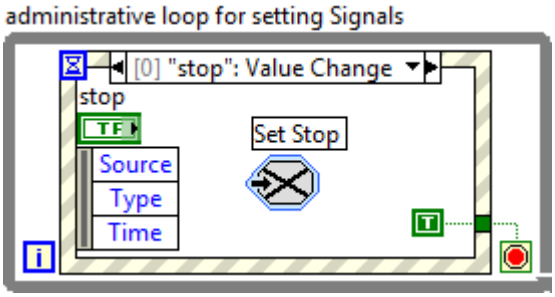


Create an "event case" for value change of "stop", "fill" and "flush".

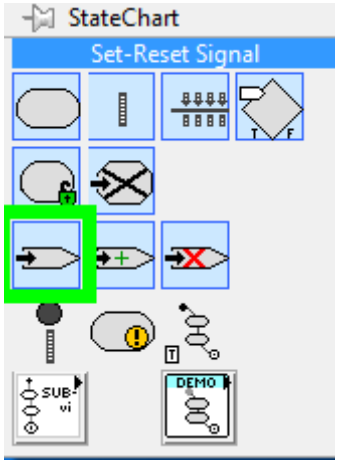
In the case of "stop" the program and the Net should close. You can close the net with a Terminator or the function SetStop. We will use the function SetStop.

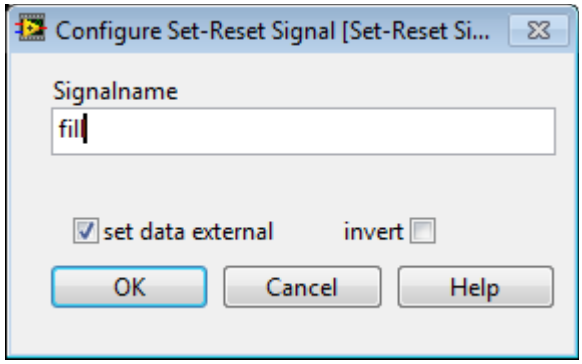


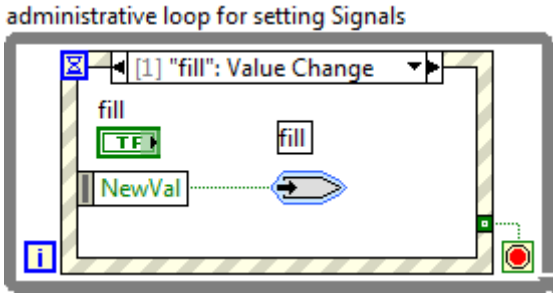




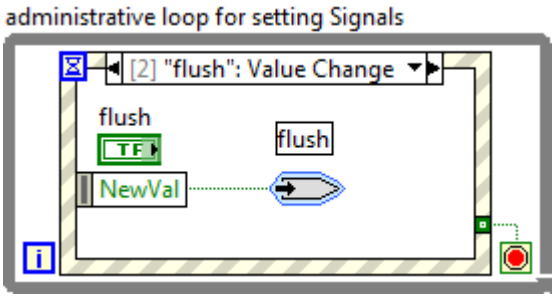
The enabling of the filling and emptying can be realized using the function "set-reset signal". Enter at the configuration dialog a signal name "fill". Set the check-box for „set data external“.





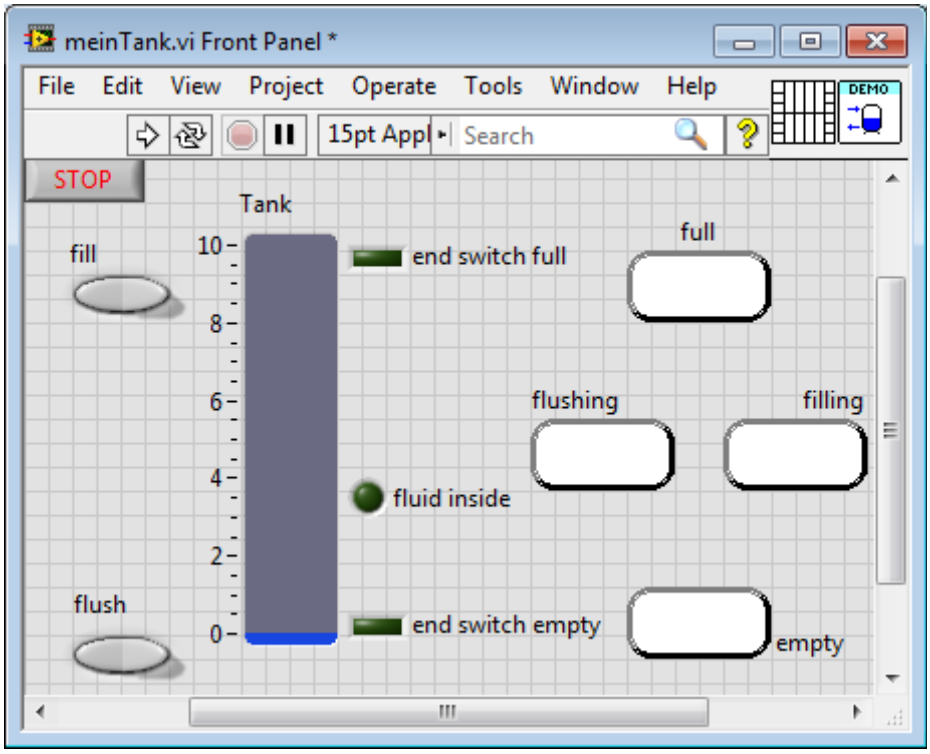


Repeat the same for the signal „flush“.

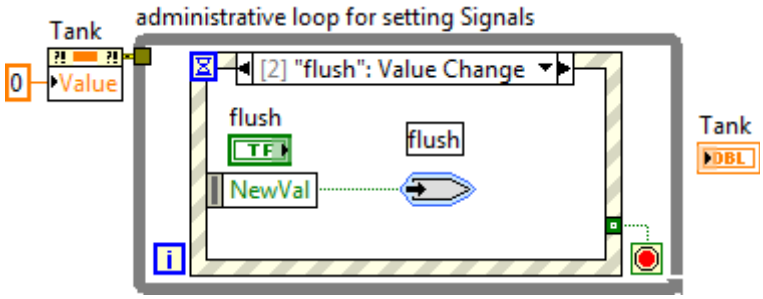


Simulation of the Tank

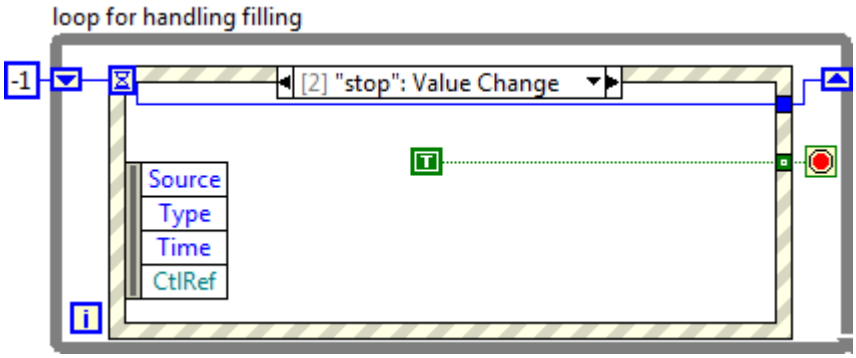
Create a numeric Indicator for the tank and for the end switches full and empty.



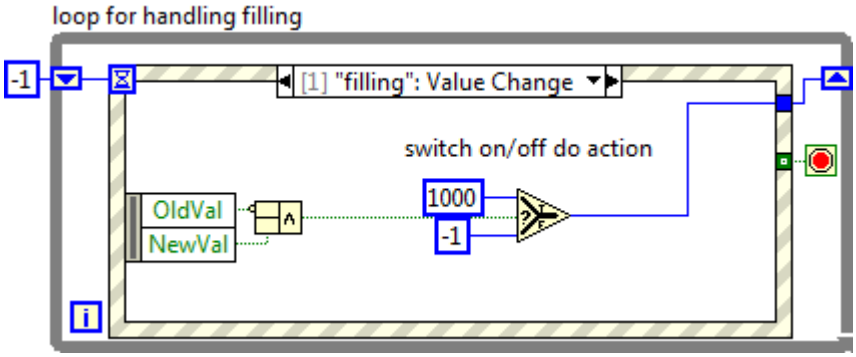
The value of the tank is set at program start to 0.



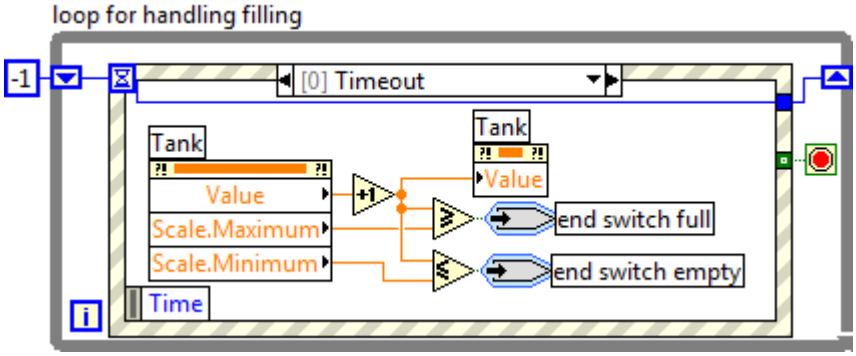
We now want to simulate the filling. Create a "Event Case" for the simulation of filling. The "Event Case" should end by „stop“.



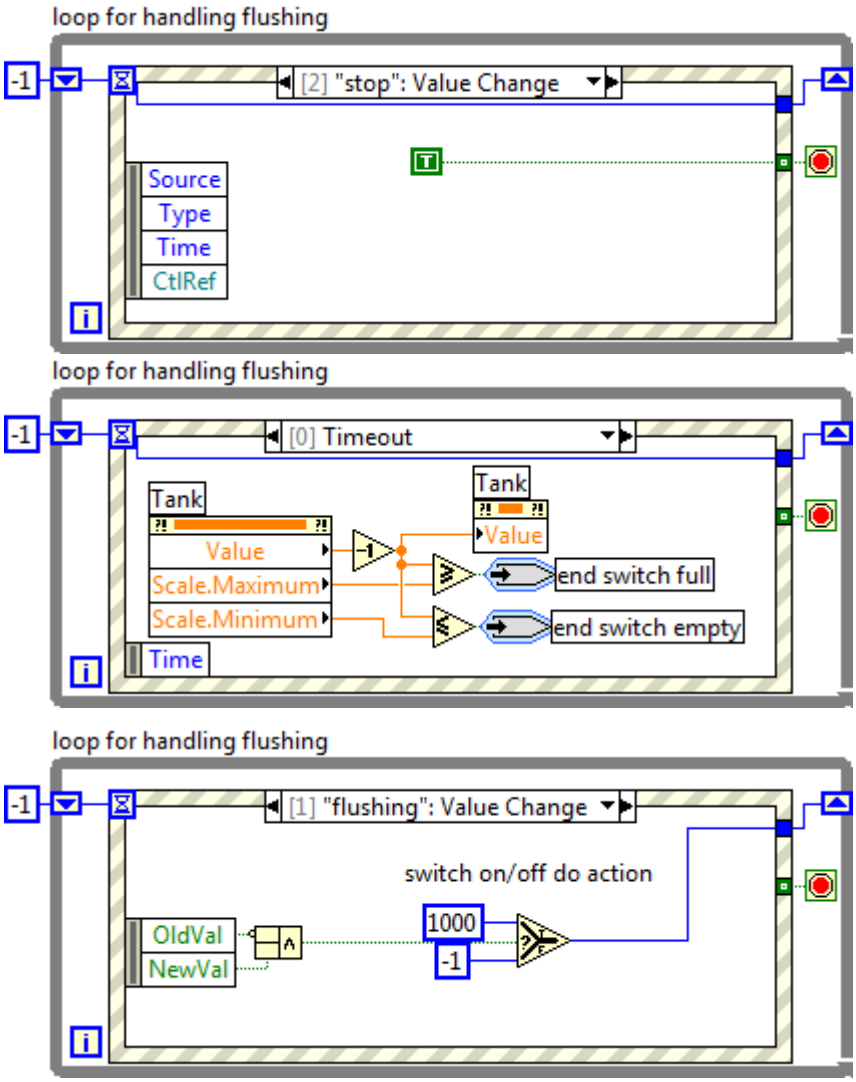
On state „filling“ = „True“, we want increment the content of the tank cyclic. Therefor we use the timeout case of the event structure. We switch on or off the case timeout, on state change of the state „filling“. Since we have enabled the "Signaling" option on the creation of the state of "filling", we get an event when the state "filling" is changed on the indicator "filling".



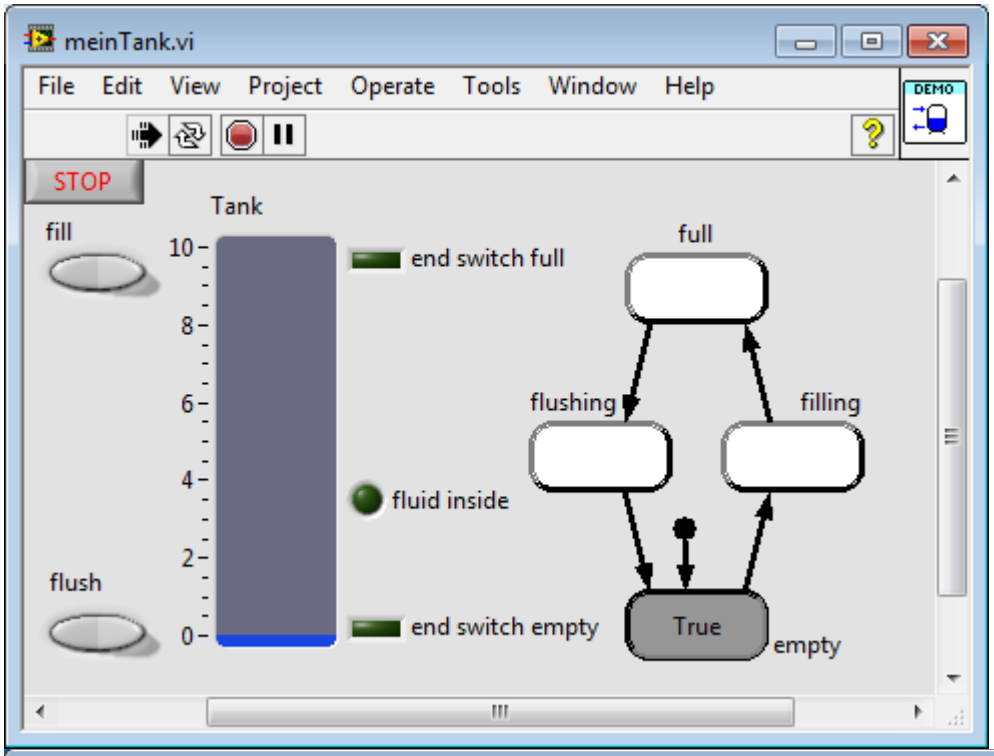
The timeout action increases the tank level and generates the signals of the limit switches.

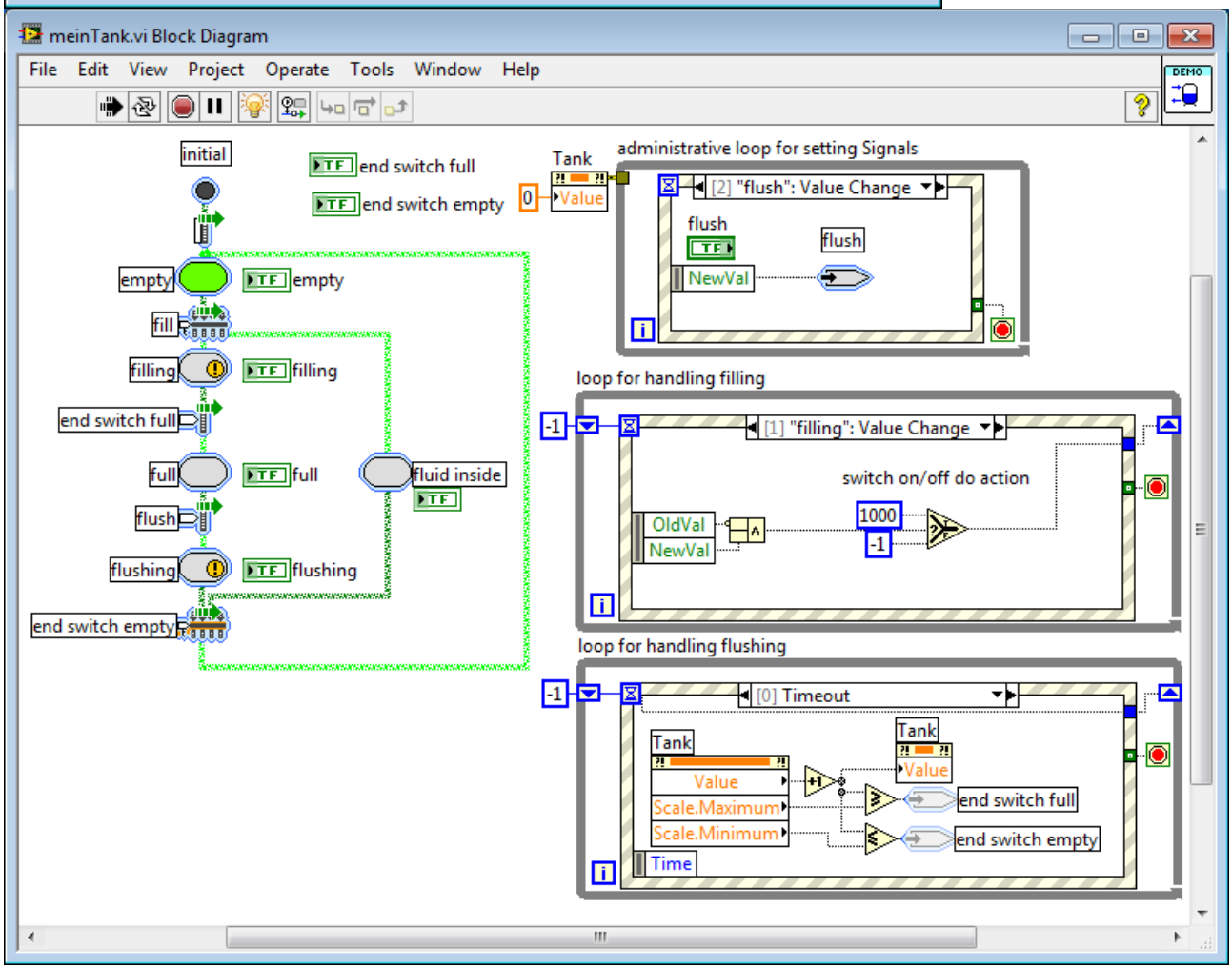


For the simulation of emptying we create another event structure analogous to the filling, but with the difference that this time the change "flushing" switches on or off the timeout action and the timeout action reduces the tank level.



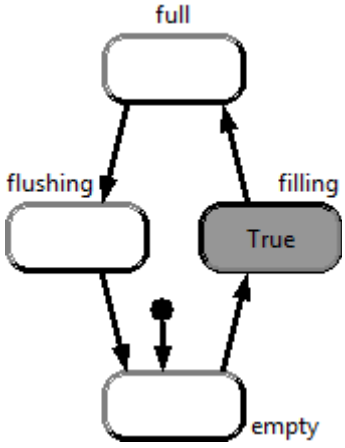
Now you can even embellish the front panel slightly and start your application for the first time.


meinTank.vi
File Edit View Project Operate Tools Window Help
DEMO
STOP
Tank
fill
flush
10
8
6
4
2
0
end switch full
fluid inside
end switch empty
full
flushing
filling
True
empty

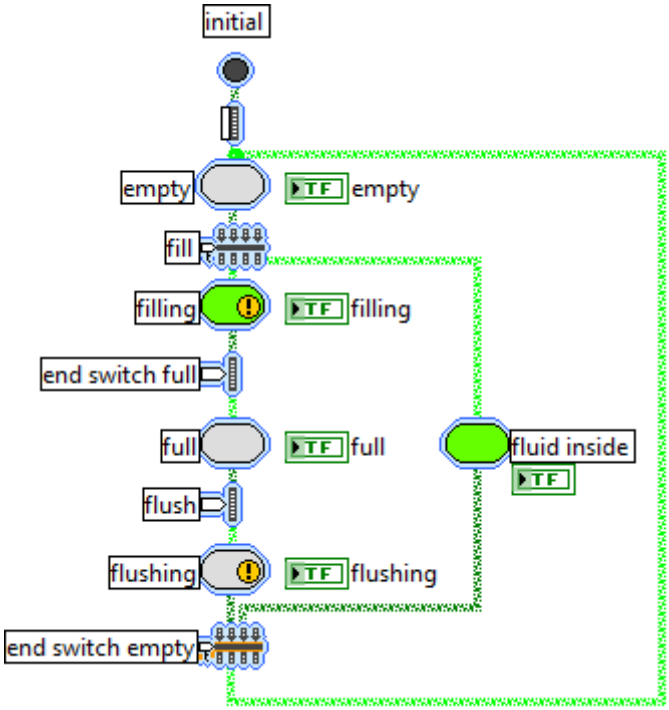

meinTank.vi Block Diagram
File Edit View Project Operate Tools Window Help
DEMO
initial
end switch full
end switch empty
Tank
0
Value
administrative loop for setting Signals
[2] "flush": Value Change
flush
NewVal
empty
fill
filling
full
flushing
fluid inside
loop for handling filling
[1] "filling": Value Change
switch on/off do action
OldVal
NewVal
1000
-1
loop for handling flushing
[0] Timeout
Scale.Maximum
Scale.Minimum
Time
+1

Observe the Net

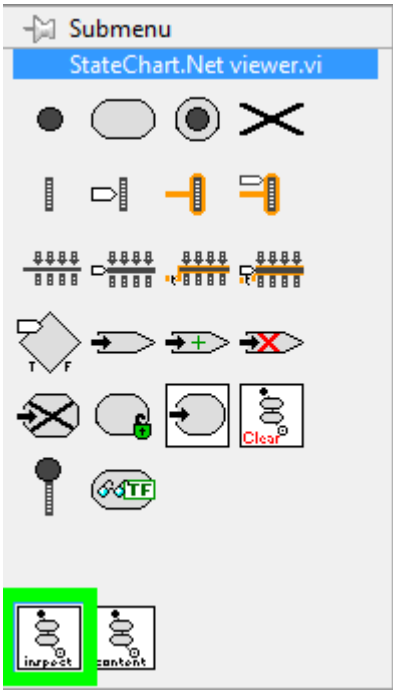
All states for which there exists a Boolean indicator are represented by these. (In our example, these are all, except the initial state.)



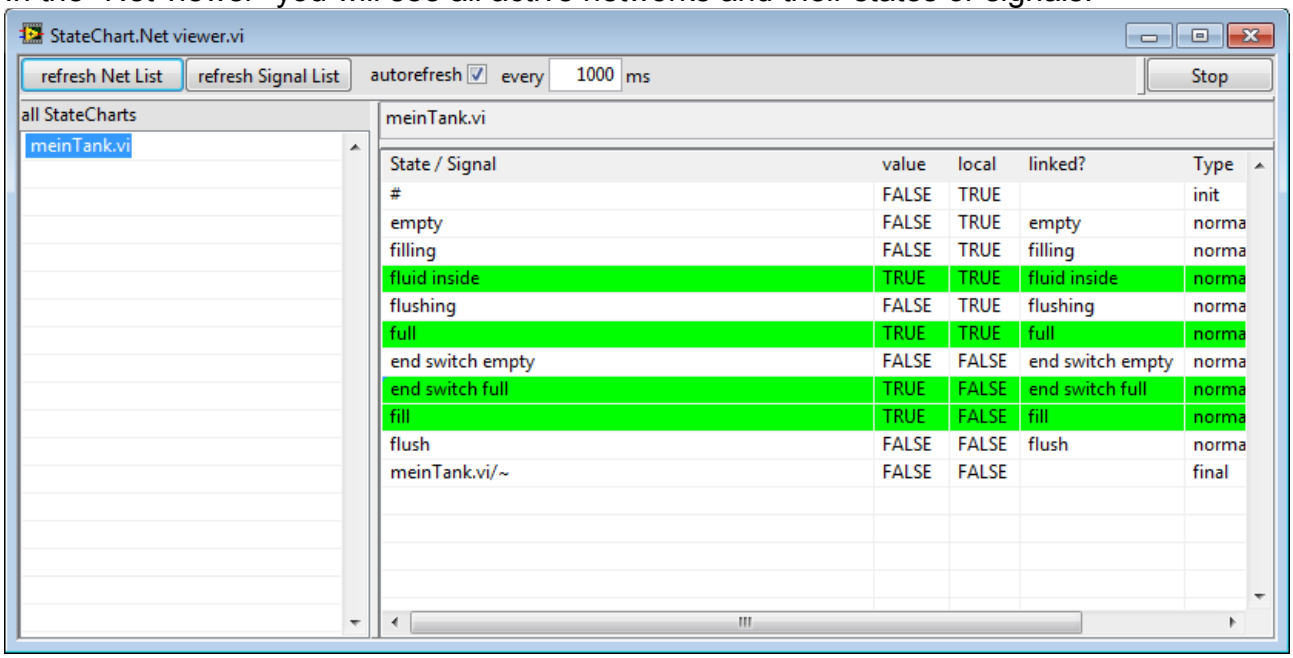
In addition, at the block diagram the icons of the active state VI's are color coded (only express VI's).



The network may also be observed by the "StateChart.Net viewer". You can find it on the Palette StateChart – Sub-menu or at the menu under „Tools → open StateChart Netinspector".



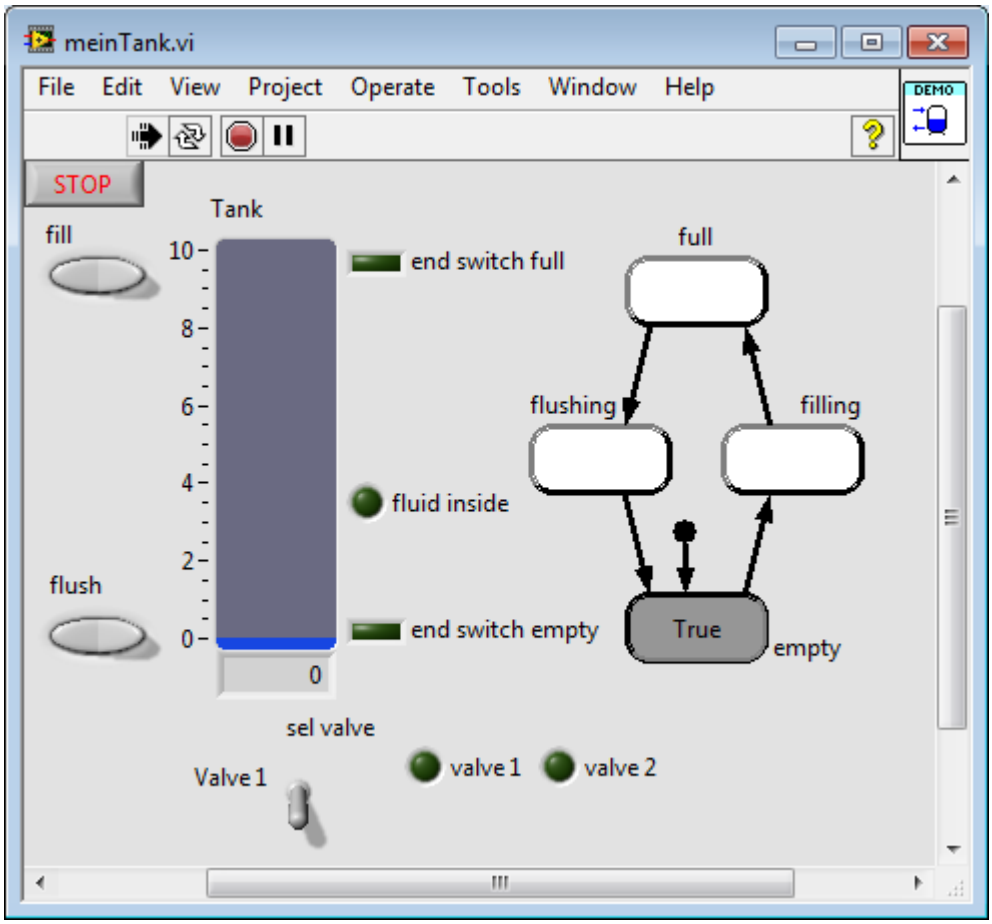
In the "Net viewer" you will see all active networks and their states or signals.



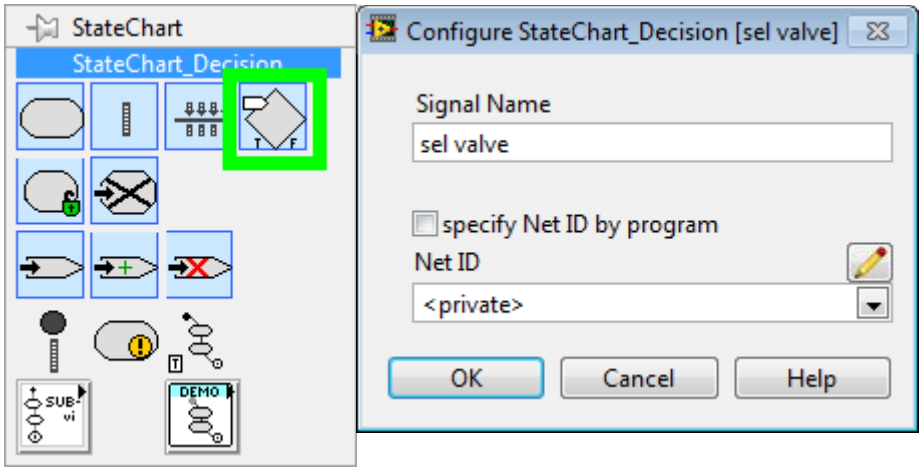
For our net, these are the programmed states "full", "empty", "filling", "flushing", "fluid inside" as well as the initial state ("#"). Furthermore, a final state is always generated here "meinTank.vi / ~". In addition, the net include the enabling conditions "fill", "flush", "end switch full" and "empty end switch".

Extension

One could imagine that you want to connect two exhaust valves to the tank. Of the valves only one to be active.



To implement this behavior, we need a decision. Create a Boolean control with the name "sel valve" and place a "Decision" on the block diagram. Configure the decision as shown below.



Create an event in the event structure for setting the signals.

Expand your net to the states "valve1" and "valve 2" and "?valve". Connect the elements.

administrative loop for setting Signals