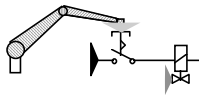

Liveness of Sequential Function Charts

The nightmare of an automation engineer is, that plant control is caught in a dead-lock or in a subsequence, from where other plant control function cannot be executed anymore. An important property of automata to handle this problems is called liveness. Roughly, an automata has the liveness property if it is possible to go through all possible passes of the automata and being able to reach the initial marking again. In plant control it is often necessary to have an automata with a sequence of initialization steps and some steps for stopping. This sequences are usually executed only once and lead to non-live automata. To be able to use the liveness property despite initialization or shut-down sequences it is reasonable to split automata into living and non-living parts.

Definition of liveness: A SFC has the liveness property for an initial marking, if and only if there exists a firing-sequence of enabled transitions that leads back to the initial marking and all of the transitions were fired at least once.

The liveness property can also be expressed in term of the SFC array representation M:

A SFC has the liveness property, if there exists a vector f with positive, nonzero integers for witch $M*f = 0$.

This rather abstract formula becomes evident if we study the meaning of multiplying a column of the array with an integer factor and the addition of two columns. Multiplying a column by an integer factor n leads to a row indicating how the number of tokens will change, when the transition, related to that column, fires n-times. If two columns are added element-by-element, the resulting column specifies the change of tokens when both transitions are fired together. Now, if there is a combination of the before mentioned array operations which leads to a column with all elements equal to zero, the pattern of tokens (marking) will not change. Therefore, the elements of the vector f indicate, how often each transition has to be fired to get back to the initial marking. If all elements of f are integers greater than zero, then each transitions has fired at least once.

For the alarm example of the succeeding lesson:

f^T	1	1	1	
step\ transition	T1	T2	T3	Sum
S1	-1	0	1	0
S2	1	-1	0	0
S3	0	1	-1	0
S4	-1	1	0	0
S5	1	-1	0	0

Table 1: Array representation of the alarm example

The table shows on the top row the vector f. Obviously each transition has to be fired once in order to get back to the initial marking.

The following example illustrates how to handle automata with non-live parts. There are to possibilities: either the initialization and stop part is removed (grayed part of Figure 1) of alternatively the stop part is connected to the initialization sequence (dashed line in Figure 1) .

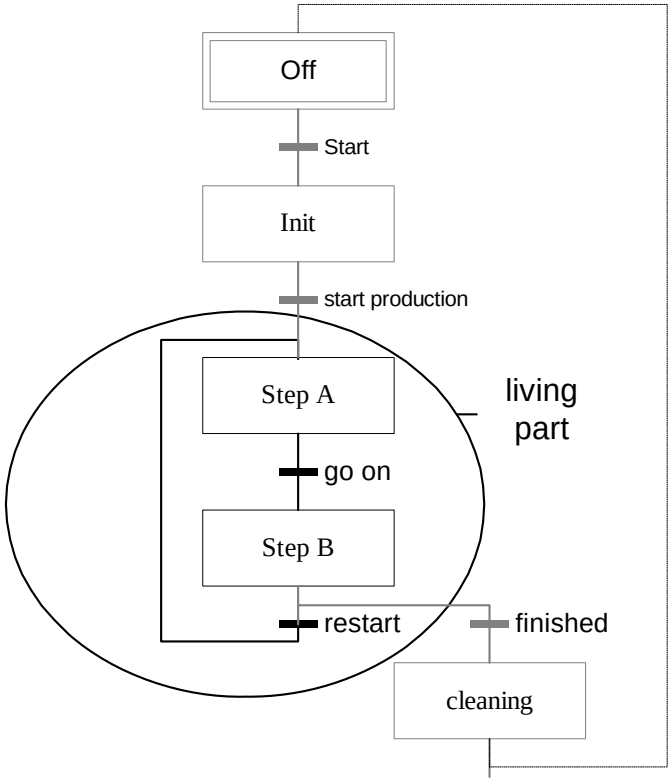

Figure 1: Living part of an SFC

Dead-Lock

What is a dead-lock? In the closed world of an SFC, this question can easily be answered. A SFC is in a dead-lock if there is a marking, where not transition is enabled. Dead-lock are the nightmare of engineers. We are therefore interested in method to analyze our automata to be able to exclude dead-locks. With the available computer power, this can be immediately checked for an SFC of reasonable size. But a dead-lock free SFC does not guarantee that plant control never goes into a dead-lock. This is because the SFC is only a part of the complete system. To have a complete check for dead-lock, the entire system has to be modeled and this is a task for genius people.

The following SFC has a dead-lock:

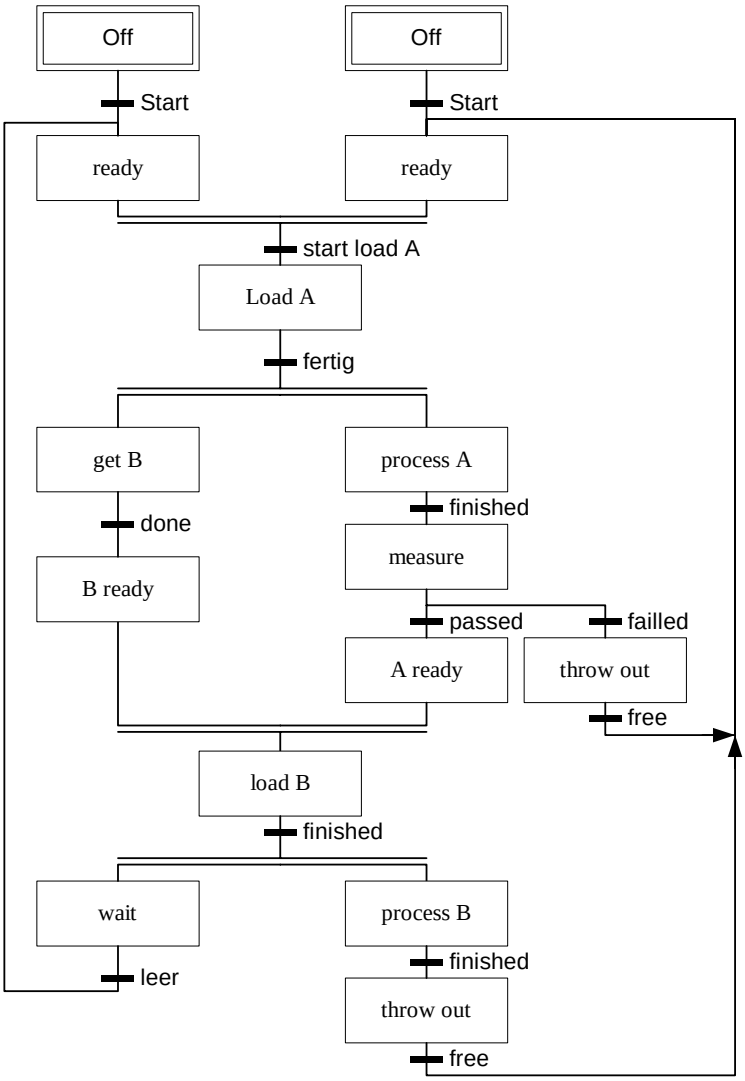

Figure 2: Example of an SFC with Dead-Lock

Simulate the SFC and draw the dead-lock marking! How can we avoid this kind of dead-locks?

Reachability and Reachability graph

A marking M2 is reachable from a marking M1 if there exists a sequence of firing transitions that change the SFC marking from M1 to M2.

A reachability graph is a formal representation, which shows the interconnection of all possible markings when only a single transition is fired. A reachability graph of a bounded SFC has a finite number of markings and can be graphically represented. For a reasonable SFC this graph can be drawn on a sheet of paper of reasonable size!

Figure 3 shows all possible marking of example SFC. It is your task to complete the graph. Draw arcs between the different markings of the SFC and label them with the name of the fired transition!

Initial marking

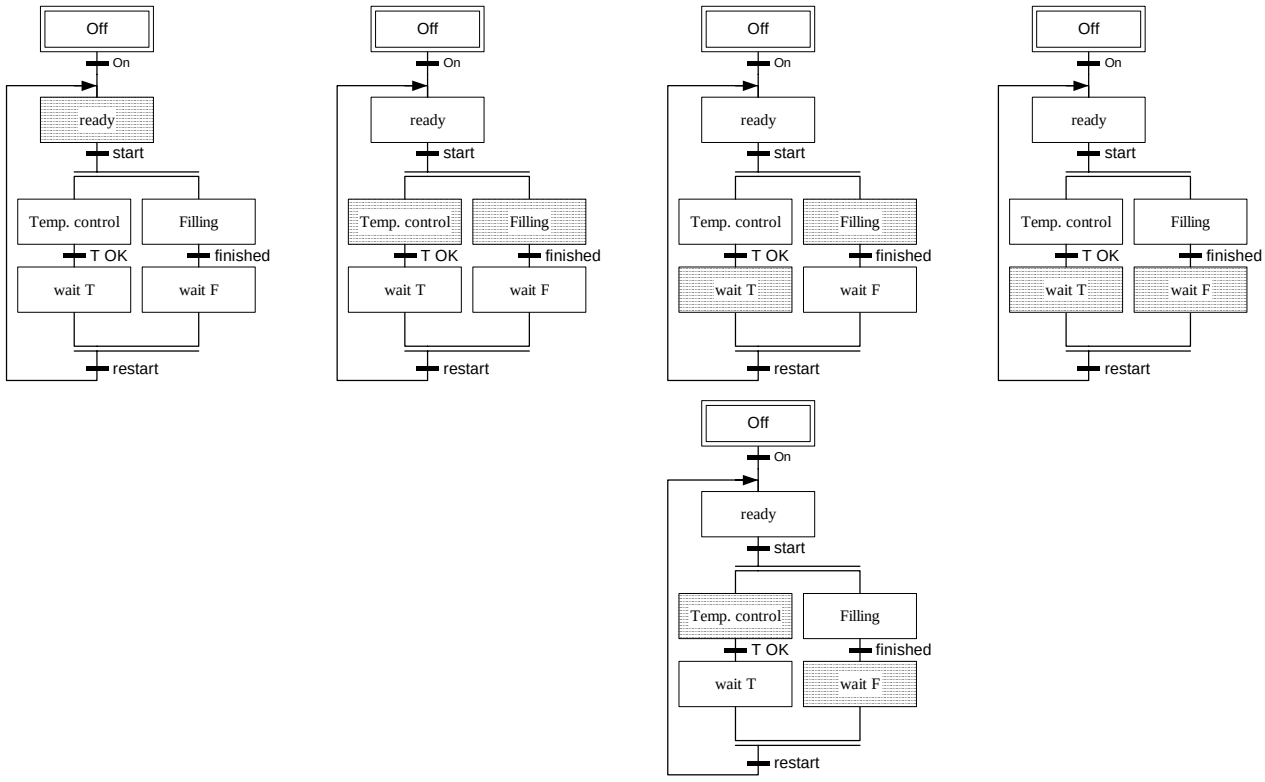

Figure 3: Reachability graph

Sketch an algorithm to create a reachability graph!

The state of a sequential function chart

The term state machine is well know. How to SFC relate to the concepts of a state machine? To answer this question, it is reasonable to spend some thoughts on the meaning of a 'state' and on the reason, why were are interested in state representations. When you buy a car, you are interested not interested in the complete history of the car if you have a complete knowledge of its actual state. The state is therefore an information that includes all necessary information about the past to be able to predict the actual and future behavior of the system.

In the context of SFC, there arises the question: what do we have to know about the state of the SFC to be able to predict its future behavior? Obviously we have to now the pattern of marked steps, i.e. the marking of the SFC. Indeed, the set of reasonable markings are the states of the SFC and as a consequence: the reachability graph is its state machine representation.