Supervisors are typically implemented using PLCs (Programmable Logic Controllers), and the older version of Supremica can generate code that is understood by PLCs.

1. Instruction List (IL)
2. Structured Text (ST)
3. Function Block Diagrams (FBD)
4. Java
5. C
Potential problems

1. Event detection. A PLC or any computer operates periodically, and it is possible that we miss the event completely, or do not know in which order events take place.

2. Conflicts. You can have several enabled controllable events that can take place at the same time, but you have to choose which one should be allowed first.

3. Inexact synchronization. If an uncontrollable event takes place directly after input reading, the supervisor might think we are in another state.

4. Modular and distributed supervisor implementation. This is highly desirable from a practical point of view, but it causes coordination problems.
Event detection

Uncontrollable events are obtained from input signals.

Input signals can be boolean valued, or if not, one can generate events by using comparison against thresholds.

Requirement 1: Each signal cannot generate more than one event per scan cycle, otherwise we can miss events.

During a scan cycle more than one event can be detected, but they have to be from different inputs. But in that case we cannot detect in which order the events did occur.

Requirement 2: In order to guarantee correct implementation we need *interleaving insensitivity*.

This means that if we do not know the order of two events, it should not matter
in which order we execute them.

Is not interleave insensitive.

Is interleave insensitive.
Conflicts

If we are at a state $s$ where more than one controllable event is enabled, a conflict is possible.

If the occurrence of one of these enabled events lead to a state where some of the other enabled events no longer are enabled, we have a conflict at $s$ between the occurred event and the disabled events.

Here the behavior will depend on if we choose to execute a or b at initial state.

Here we do not have the same problem.
Inexact Synchronization

When the supervisor is running, the plant may generate events that are not seen before the next input reading.

This might mean that we supervise according to an incorrect state.

We call this problem delay sensitivity.

This is delay sensitive, if the uncontrollable event $uc$ takes place directly after reading inputs, we will allow $a$ although we should allow $b$.

Here we do not have the same problem.
Modular or distributed supervisor implementation

It is neither always possible nor desirable to implement all parts of the supervisor on one PLC.

Controllable events common to more than one supervisor module can be enabled only when it is enabled by all modules involved.

One need to differentiate between forcing and enabling supervisors.

The enabling supervisors need to send enabling signals to the forcing supervisor.

The product operation is useful for finding common events.