Industrial Automation

(Automação de Processos Industriais)

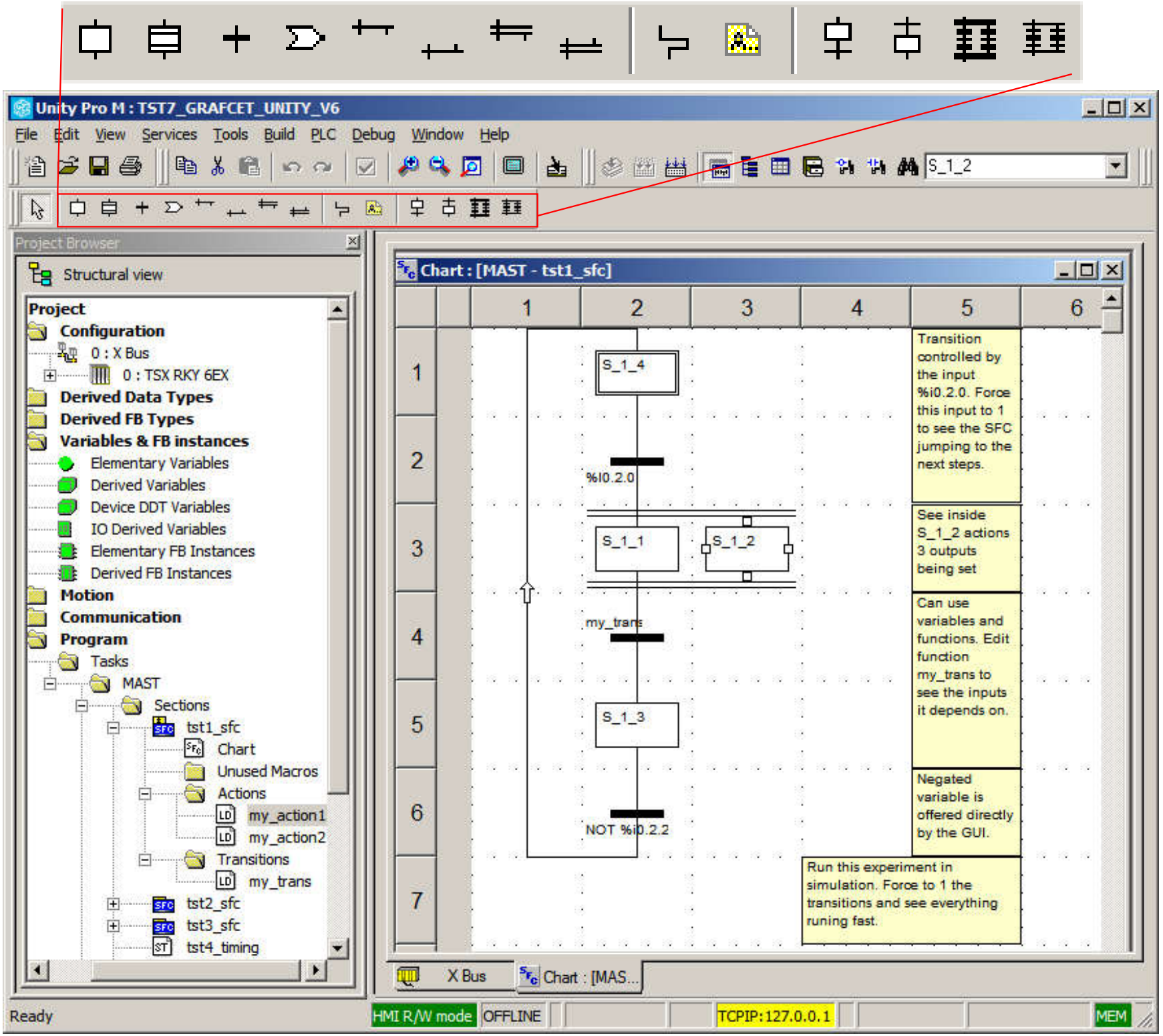
GRAFCET

(Sequential Function Chart) 2/3

http://users.isr.ist.utl.pt/~jag/courses/api19b/api1920.html

Prof. José Gaspar, 2019/2020

Creating a *Sequential Function Chart* in Unity



Operating Modes

Hide | Locate | Back | Forward | Stop | Refresh | Print | Options

Contents | Index | Search

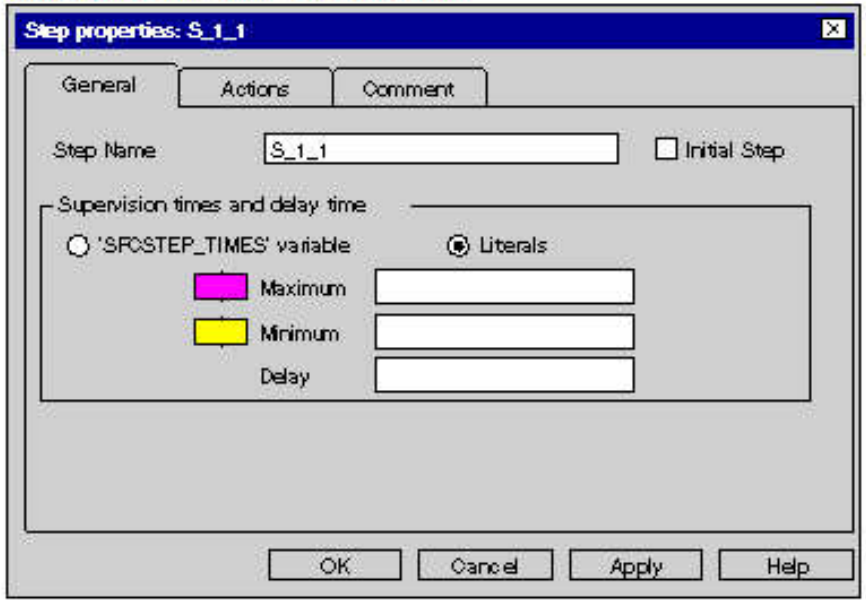
- Unity
  - Whats New
  - General Safety Instructions
  - Compatibility Rules
  - Addendum
  - Unity Pro Software
    - Languages Reference
      - Safety Information
      - About the Book
      - General Presentation of Unity Pro
      - Application Structure
      - Data Description
      - Programming Language
        - Function Block Language FBD
        - Ladder Diagram (LD)
        - SFC Sequence Language
          - General Information about SFC Sequence
          - Steps and Macro Steps
          - Actions and Action Sections
            - Action
            - Action Section
            - Qualifier
          - Transitions and Transition Sections
            - Transition
            - Transition Section
          - Jump
          - Link
          - Branches and Merges
            - Alternative Branches and Alternative J
            - Parallel Branch and Parallel Joint
          - Text Objects
          - Single-Token
          - Multi-Token
        - Instruction List (IL)

Steps properties dialog box

Submit Feedback

General tab

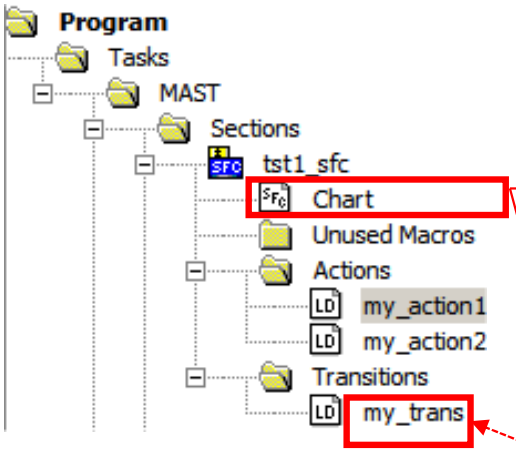
Representation of the **General** tab:

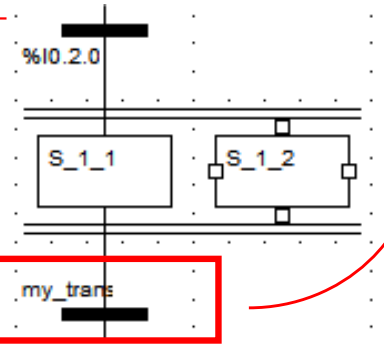


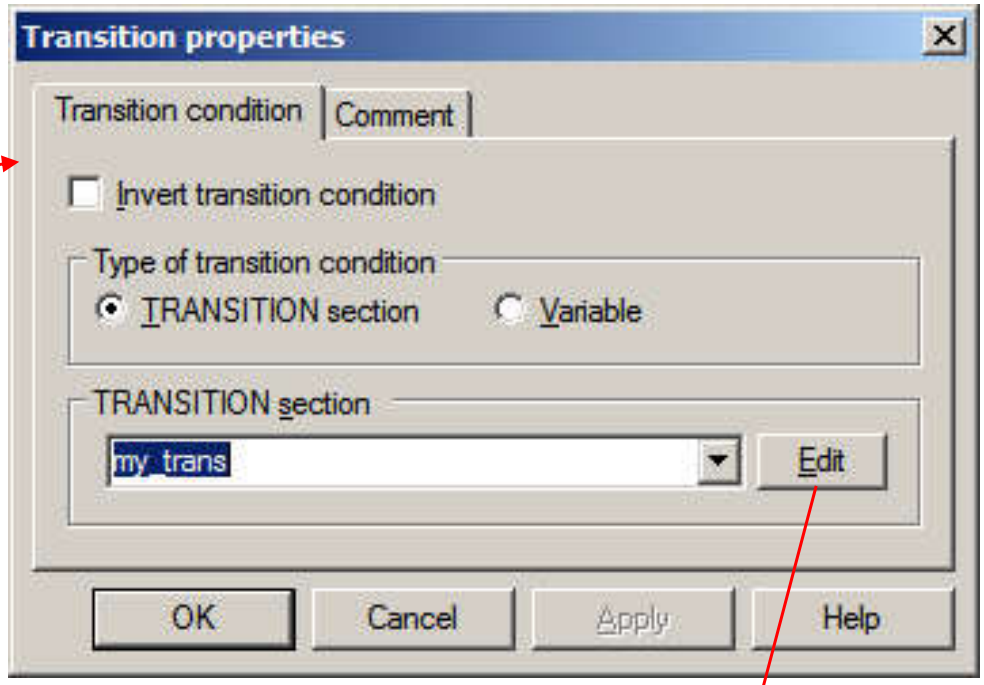
Elements of the **General** tab:

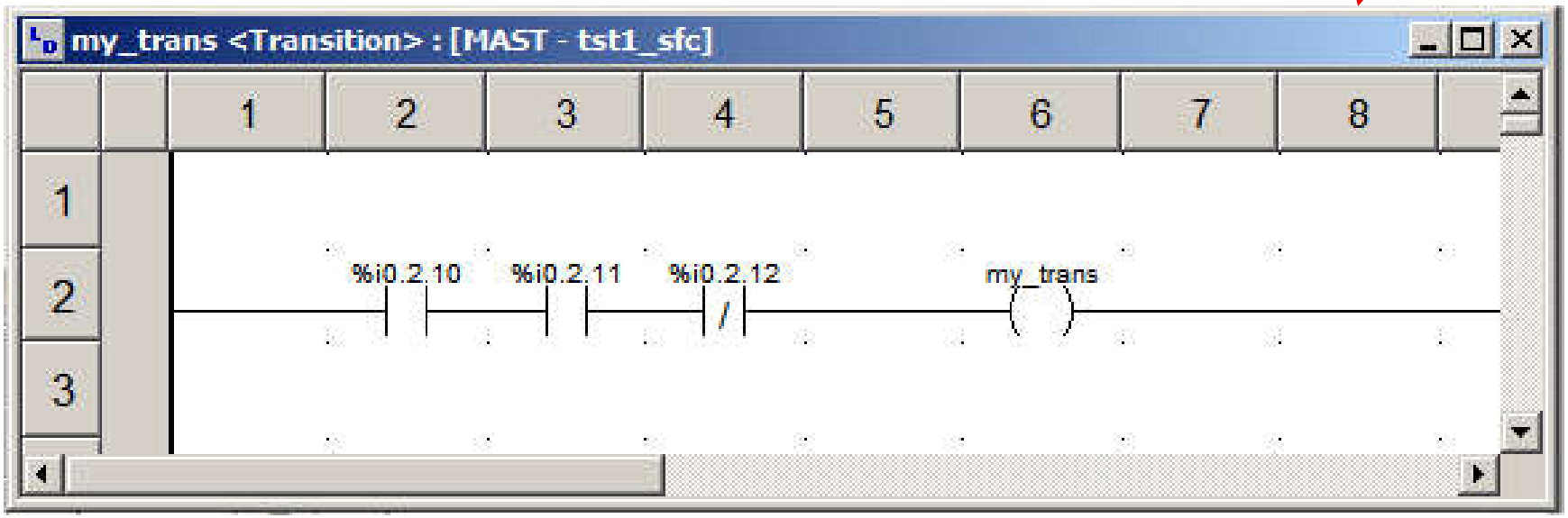
Element	Description
Step Name	Enter the step name in this text box.
Initial Step	When this check box is activated, the current step becomes an initial step.
Supervision times and	Define the step supervision times and delay time in this area, see also section *Step times* in the Reference manual

Transitions can be variables or **sections**









Properties of **Transition Sections** (Unity Pro)

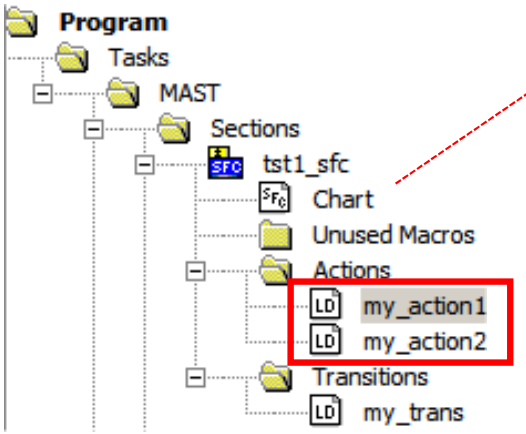
Transition sections have the following properties:

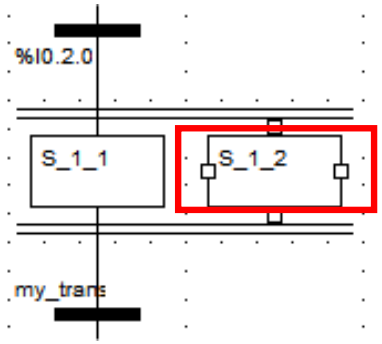
- Transition sections only have **one single output, *transition variable***, whose data type is BOOL. The name of these variables are identical to the names of the transition sections.
- The transition variable can only be used once in written form.
- The transition variable can be read in any position within the project.

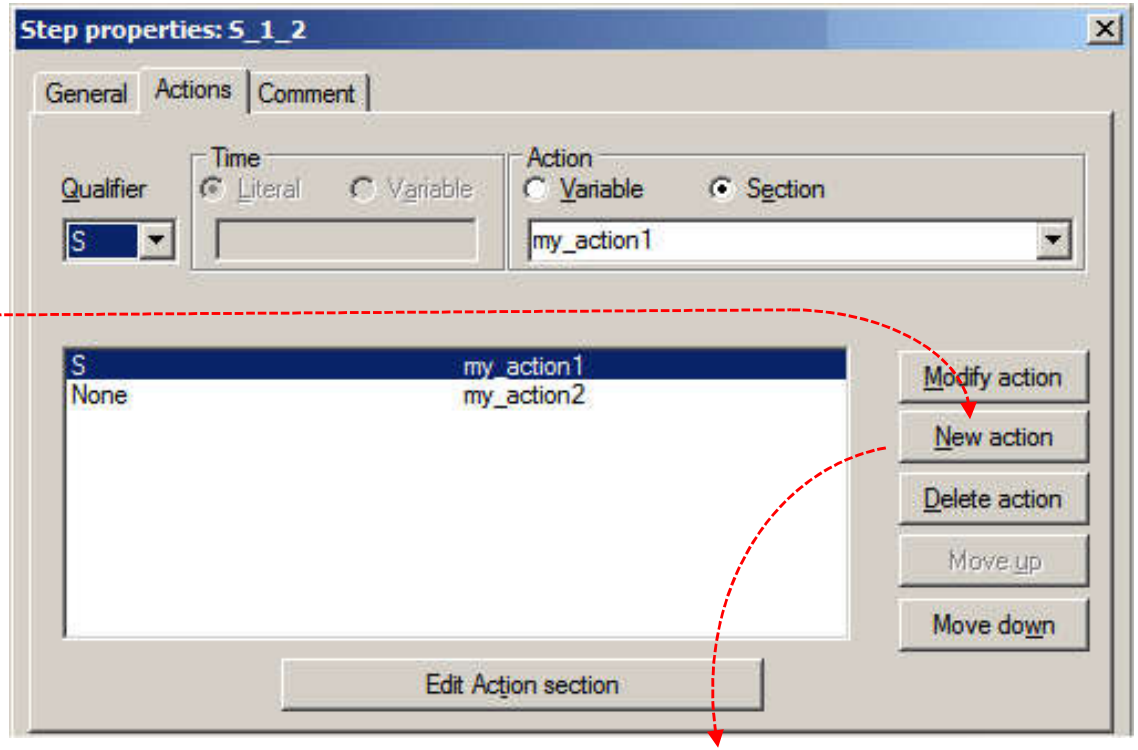
Alternatively, can use a ***transition function*** to define the transition logic:

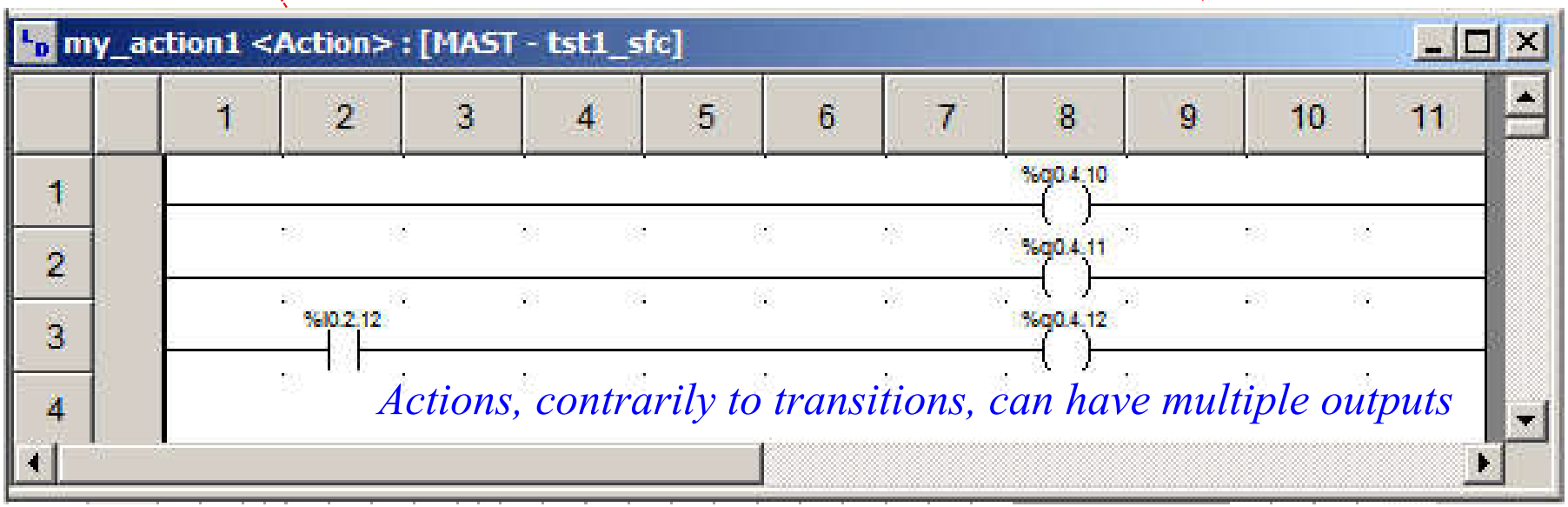
- Only functions can be used. Function blocks or procedures cannot be used.
- **Only one coil** may be used in LD.
- There is only one network, i.e. all functions used are linked with each other either directly or indirectly.
- Transition sections can only be used once.
- **Transition sections belong to the SFC section in which they were defined**. If the respective SFC section is deleted then all transition sections of this SFC section are also deleted automatically.
- Transition sections can be called exclusively from transitions.

Actions are associated to Steps



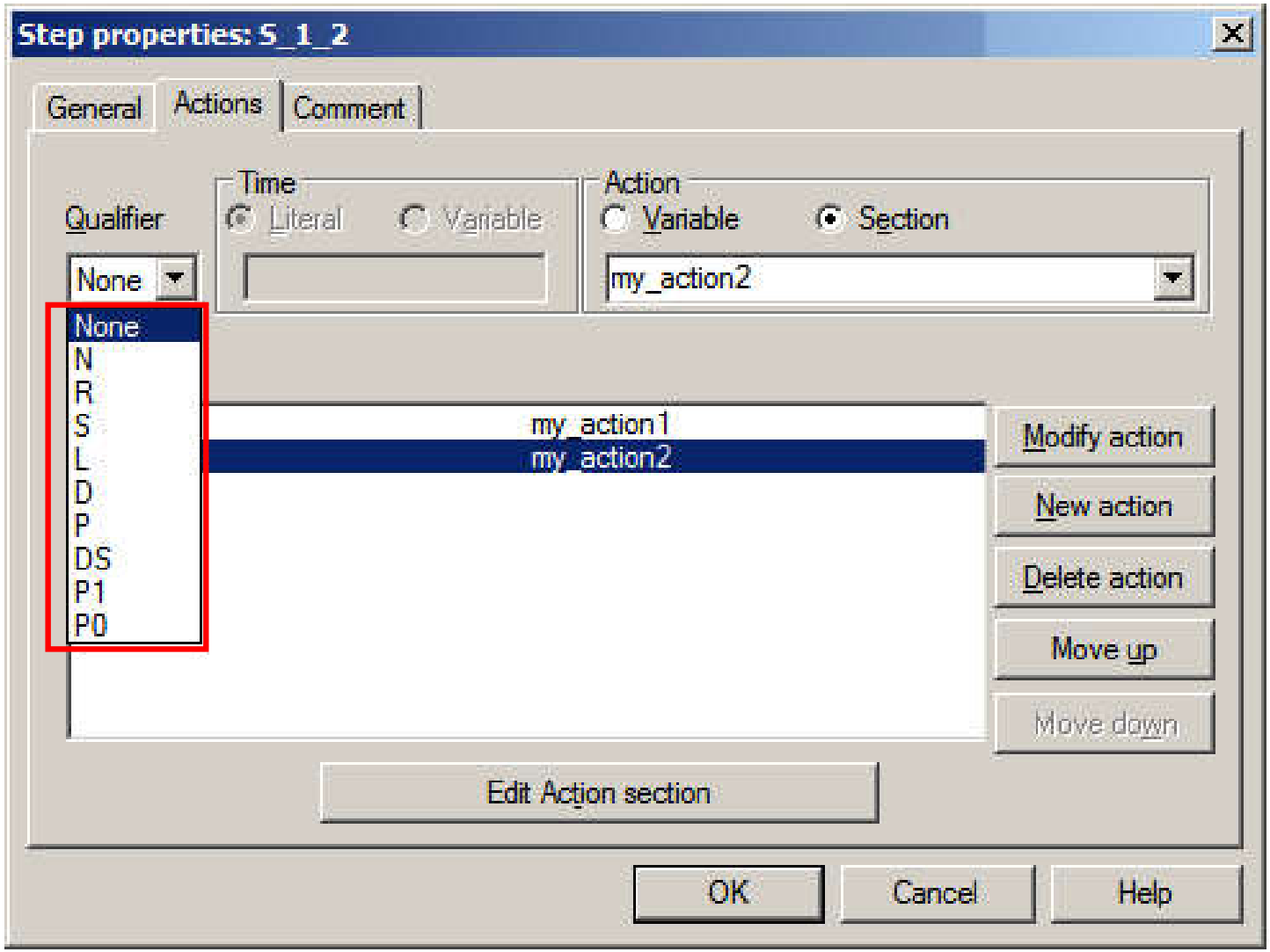






Actions, contrarily to transitions, can have multiple outputs

Actions can be of various types



Qualifier (Meaning)	Description
N / None (None)	If the step is active then action is 1 and if the step is inactive the action is 0.
R (Overriding)	The action, which is set in another step with the qualifier `S`, is reset. The activation of any action can also be prevented.
S (Set)	The set action remains active, even when the associated step becomes inactive. The action only becomes inactive, when it is reset in another step of the current SFC section, using the qualifier R. **Note:** If an action variable is modified outside of the current SFC section, it may no longer reflect the action's activation state.

P (Pulse)	If the step becomes active, the action becomes 1 and this remains for one program cycle, independent of whether or not the step remains active.
P1 (Pulse rising edge)	If the step becomes active (0->1-edge), the action becomes 1 and this remains for one program cycle, independent of whether or not the step remains active. **Note:** Independent of their position in the action list field, actions with the qualifier `P1` are always processed first. More information can be found in the Action of the SFC sequence language.
P0 (Pulse falling edge)	If the step becomes inactive (1->0-edge), the action becomes 1 and this remains for one program cycle. **Note:** Independent of their position in the action list field, actions with the qualifier `P0` are always processed last. More information can be found in the Action of the SFC sequence language.

L (Time limited)	If the step is active, the action is also active. After the process of the time duration, defined manually for the action, the action returns to 0, even if the step is still active. The action also becomes 0 if the step is inactive. **Note:** For this qualifier, an additional duration of data type `TIME` must be defined.
D (Delayed)	If the step is active, the internal timer is started and the action becomes 1 after the process of the time duration, which was defined manually for the action. If the step becomes inactive after that, the action becomes inactive as well. If the step becomes inactive before the internal time has elapsed then the action does not become active. **Note:** For this qualifier, an additional duration of data type `TIME` must be defined.
DS (Delayed and saved)	If the step becomes active, the internal timer is started and the action becomes active after the process of the manually defined time duration. The action first becomes inactive again when qualifier R is used for a reset in another step. If the step becomes inactive before the internal time has elapsed then the action does not become active. **Note:** For this qualifier, an additional duration of data type `TIME` must be defined.