

# **Industrial Automation**

## **(Automação de Processos Industriais)**

### **GRAFCET**

### ***(Sequential Function Chart)***

<http://users.isr.ist.utl.pt/~jag/courses/api1718/api1718.html>

Prof. Paulo Jorge Oliveira, original slides  
Prof. José Gaspar, rev. 2017/2018

## **Syllabus:**

**Chap. 3 – PLC Programming languages [2 weeks]**

...

**Chap. 4 - GRAFCET (*Sequential Function Chart*) [1 week]**

The GRAFCET norm.

Elements of the language.

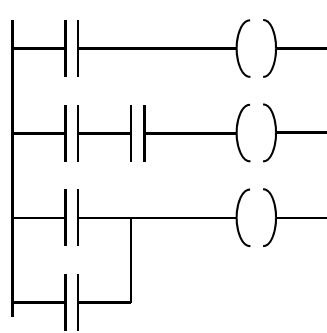
Modelling techniques using GRAFCET.

...

**Chap. 5 – CAD/CAM and CNC Machines [1 week]**

## PLC Programming Languages (IEC 61131-3)

### *Ladder Diagram*



### *Structured Text*

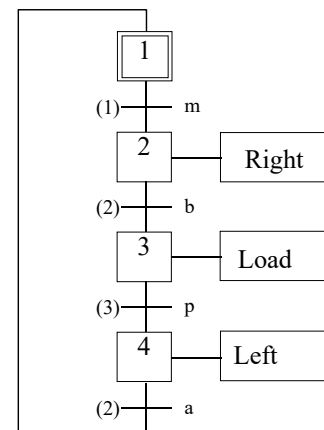
```

If %I1.0 THEN
  %Q2.1 := TRUE
ELSE
  %Q2.2 := FALSE
END_IF
    
```

### *Instruction List*

LD	%M12
AND	%I1.0
ANDN	%I1.1
OR	%M10
ST	%Q2.0

### *Sequential Function Chart (GRAFCET)*



---

## Some pointers to GRAFCETs (SFCs)

- History: [http://www.lurpa.ens-cachan.fr/grafcet/groupe/gen\\_g7\\_uk/geng7.html](http://www.lurpa.ens-cachan.fr/grafcet/groupe/gen_g7_uk/geng7.html)
- Tutorial: [http://asi.insa-rouen.fr/~amadisa/grafcet\\_homepage/tutorial/index.html](http://asi.insa-rouen.fr/~amadisa/grafcet_homepage/tutorial/index.html)  
[http://www-ipst.u-strasbg.fr/pat/autom/grafce\\_t.htm](http://www-ipst.u-strasbg.fr/pat/autom/grafce_t.htm)
- Simulator: [http://asi.insa-rouen.fr/~amadisa/grafcet\\_homepage/grafcet.html](http://asi.insa-rouen.fr/~amadisa/grafcet_homepage/grafcet.html)  
<http://www.automationstudio.com> (See projects)
- Bibliography:
- **Petri Nets and GRAFCET: Tools for Modelling Discrete Event Systems**  
R. David, H. Alla, New York : PRENTICE HALL Editions, 1992
  - **Grafcet: a powerful tool for specification of logic controllers**, R. David,  
IEEE Trans. on Control Systems Tech., 1995 v3n3 pp253-268 [\[online\]](#)
  - **Programação de Autómatos**, Método GRAFCET, José Novais,  
Fundação Calouste Gulbenkian
  - **Norme Française NF C 03-190 + R1 : Diagramme fonctionnel  
"GRAFCET" pour la description des systèmes logiques de commande**
- Homepage: <http://www.lurpa.ens-cachan.fr/grafcet/>

**Unity Pro Help**

Back Forward Print Options Help

Contents Index Search

General Information about SFC Sequence Language

See: [Related Topics](#)  [Submit Feedback](#)

Link

Initial Step

T1 Transition condition (Boolean Variable)

Parallel branch

Step (associated to an action)

T\_4.2 Transition condition (return value of a transition section)

%I0.1 %I0.2 %I0.3

Alternative branch

S\_4.7 Jump

T\_4.3

%I0.4 %I0.5

Transition condition (Topological Boolean address)

Alternative junction

Parallel junction

1 Transition condition (Boolean Literal)

MS\_4.1 Macro Step

Return\_Va r

© 2009 Schneider Electric. All rights reserved.

## GRAFCET History

- 1975 – Decision of the workgroup "Logical Systems" of AFCET (Association Française de Cybernétique Economique et Technique) on the creation of a committee to study a standard for the representation of logical systems and automation.
- 1977 – GRAFCET definition ([Graphe Fonctionnel de Commande Etape-Transition](#)).
- 1979 – Dissemination in schools and adopted as research area for the implementation of solutions of automation in the industry.
- 1988 - GRAFCET becomes an international standard denominated as [Sequential Function Chart \(SFC\)](#), by I.E.C. 60848.

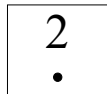
# GRAFCET Basic Elements

## Steps

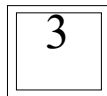
Inactive



Active



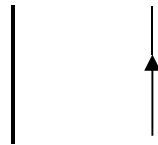
Initial



**Actions** can be associated with **Steps**.

## Connections

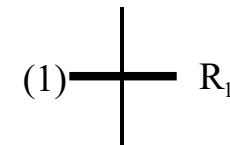
Undirected or Directed Arc



Direction of undirected **Arcs** is in context.

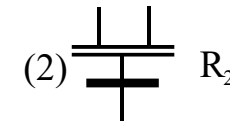
## Transitions

*Simple*



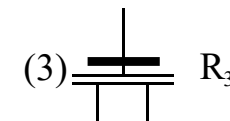
*Joint*

(parallel junction)

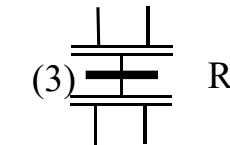


*Fork*

(parallel branch)



*Joint e fork*



A **logical receptivity** function can be associated with each **Transition**.

## GRAFCET Basic Elements

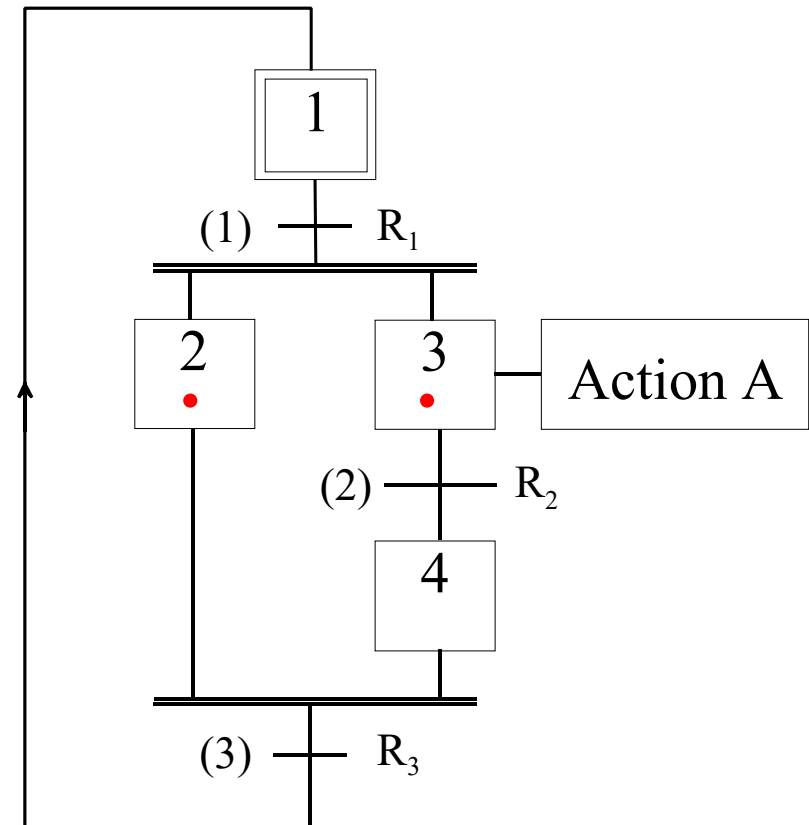
### Oriented connections (arcs)

In a GRAFCET:

An Arc can connect Steps to Transitions

An Arc can connect Transitions to Steps

*Arcs must be in-between:* A Step can not have Transitions directly as inputs (source); A Step can not have Transitions as direct outputs (drain); Similarly for the Transitions.



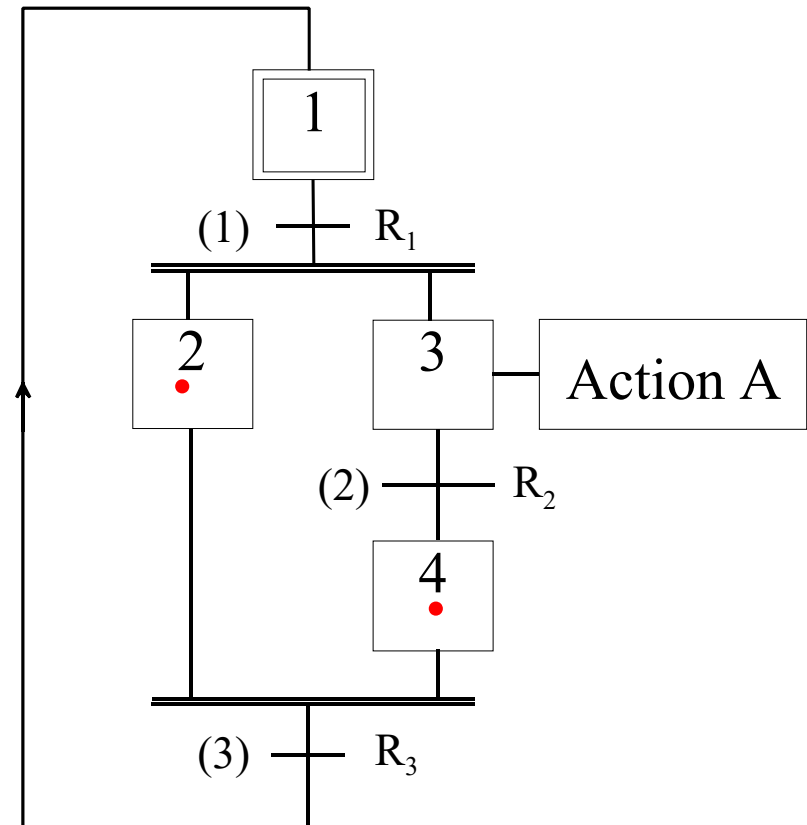


GRAFCET **State of a GRAFCET****Definition of State:**

The set of markings of a GRAFCET constitutes its state.

**Question:**

How does the state of a GRAFCET evolve?



## GRAFCET State Evolution:

- **Rule 1: Initial State**

State evolution requires active Steps at the beginning of operation (at least one).

- **Rule 2: Transposition of a Transition**

A Transition is active or enabled only if all the Steps at its input are active (if not it is inactive).

A Transition can only be transposed if it is active and is true the associated condition (receptivity function).

- **Rule 3: Evolution of active Steps**

The transposition of a Transition leads to the deactivation of all the Steps on its inputs and the activation of all Steps on its outputs.

- **Rule 4: Simultaneous transposition of Transitions**

All active Transitions are transposed simultaneously.

- **Rule 5: Simultaneous activation and deactivation of a Step**

In this case the activation has priority.

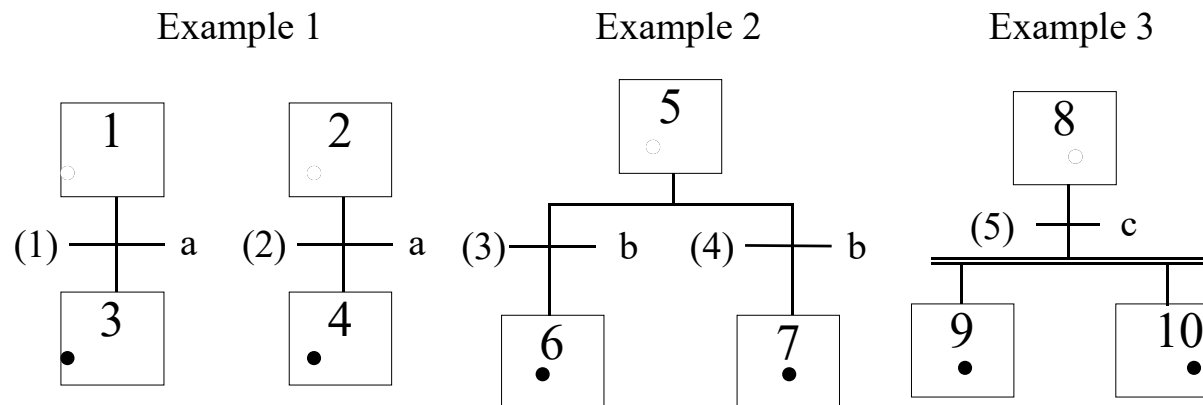
## GRAFCET State Evolution:

- **Rule 2a:**

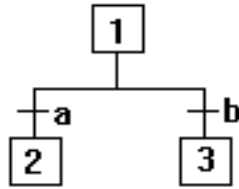
All active Transitions are transposed immediately.

- **Rule 4:**

Simultaneously active Transitions are transposed simultaneously.



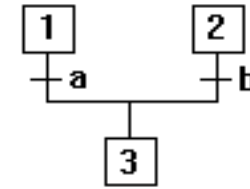
OR Divergences:



If Step 1 active and a TRUE  
 then deactivate Step 1 and activate Step 2.

If a and b TRUE and Step 1 active  
 (PL7) then deactivate Step 1 and activate Steps 2 & 3  
 (Unity) then deactivate Step 1 and activate Step 2

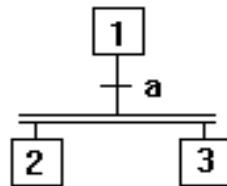
OR Convergences:



If Step 1 active and a TRUE then deactivate Step 1  
 and activate Step 3 (state of Step 2 remains unchanged).  
 The same happens for Step 2 and b.

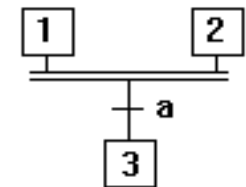
(PL7) If both Steps 1 and 2 are active and a and b are TRUE  
 then Steps 1 and 2 are deactivated and Step 3 is activated.

AND Divergences (fork):



If Step 1 active and a TRUE  
 then deactivate Step 1 and activate Steps 2 and 3.

AND Convergences (join):



If Steps 1 and 2 active and a TRUE  
 then deactivate Steps 1 and 2 and activate Step 3.

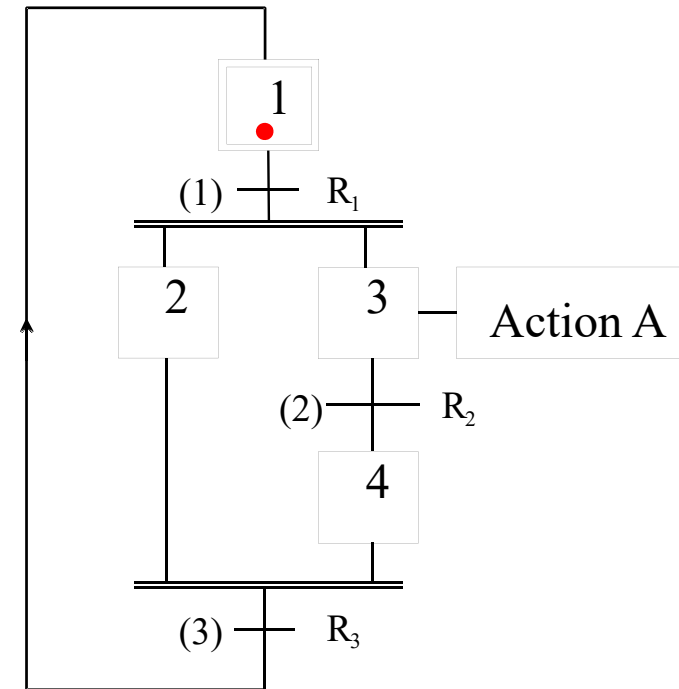
Note: to make Unity Pro similar to PL7 the option “allow multiple tokens” has to be enabled.

# GRAFCET

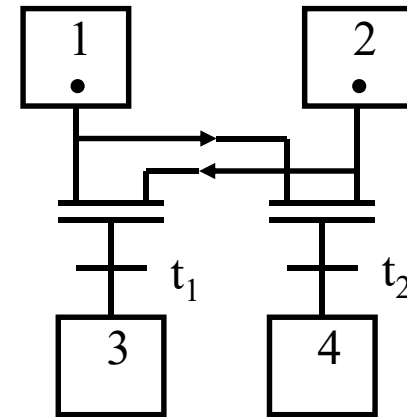
Example:

## GRAFCET state evolution

Level activated Action. Actions can also be activated during transitions - see next.



## GRAFCET

**Modelling problem:**

Given 4 Steps (1 to 4) and 2 Transitions ( $t_1$  and  $t_2$ ) write a segment of GRAFCET to solve the following problem:

In the case that the Steps 1 and 2 are active:

- if  $t_1$  is TRUE, activate Step 3 (and deactivate Steps 1 and 2);
- if  $t_2$  is TRUE, activate Step 4 (and deactivate Steps 1 and 2);
- otherwise, the state is maintained.

## GRAFCET

**Another modeling problem:**

Given 4 Steps (1 to 4) and 2 Transitions (t1 and t2) write a segment of GRAFCET to solve the following problem:

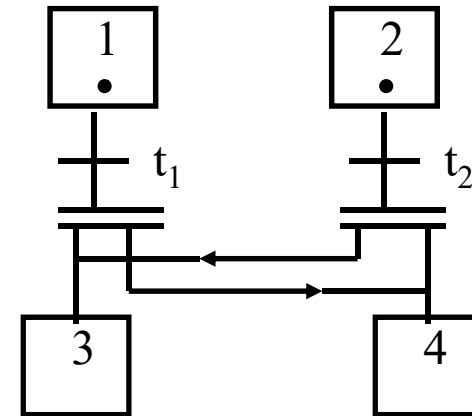
If Step 1 is active and t1 is TRUE

OR

If Step 2 is active and t2 is TRUE

THEN

Activate Steps 3 and 4.

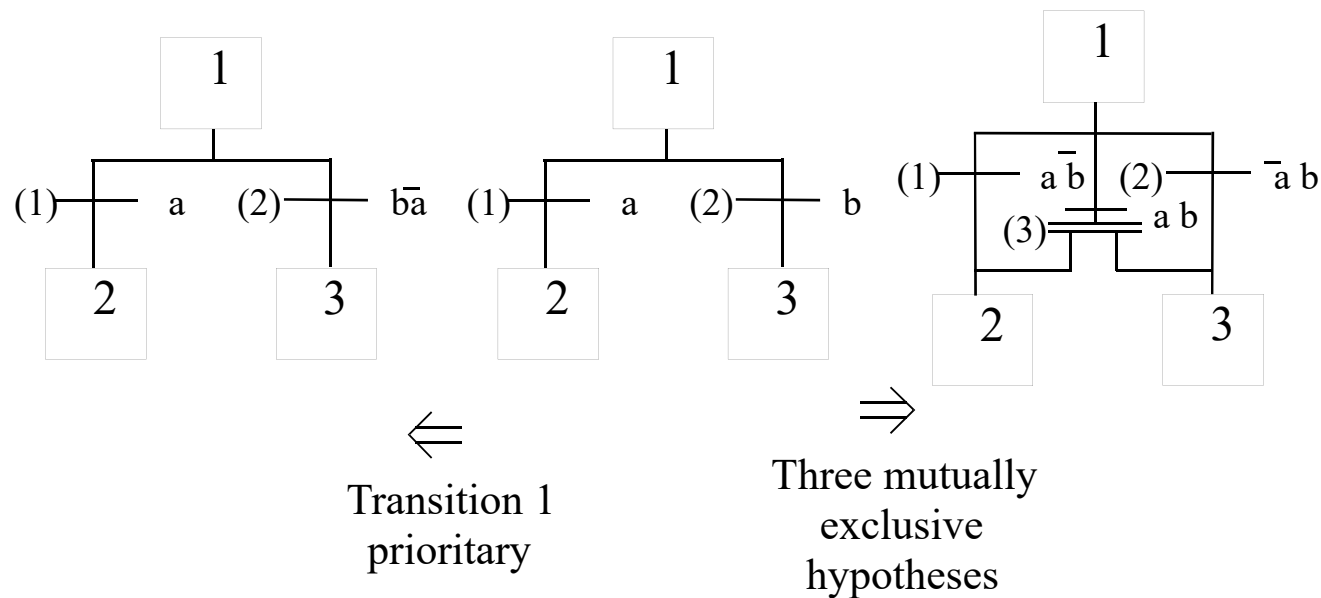


# GRAFCET

## GRAFCET state evolution, **Conflicts**:

There exist **Conflicts** when the validation of a Transition depends on the same Step or when more than one receptivity functions can become true **simultaneously**.

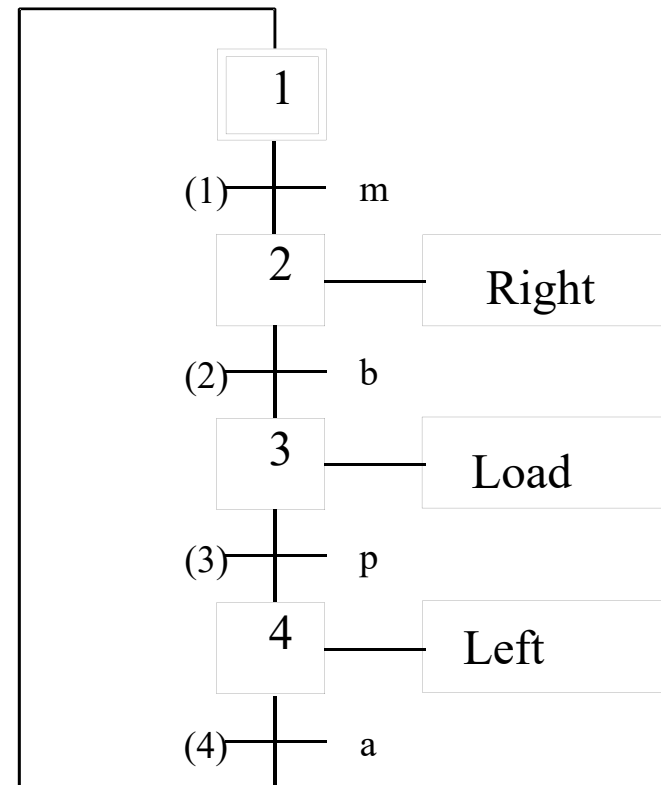
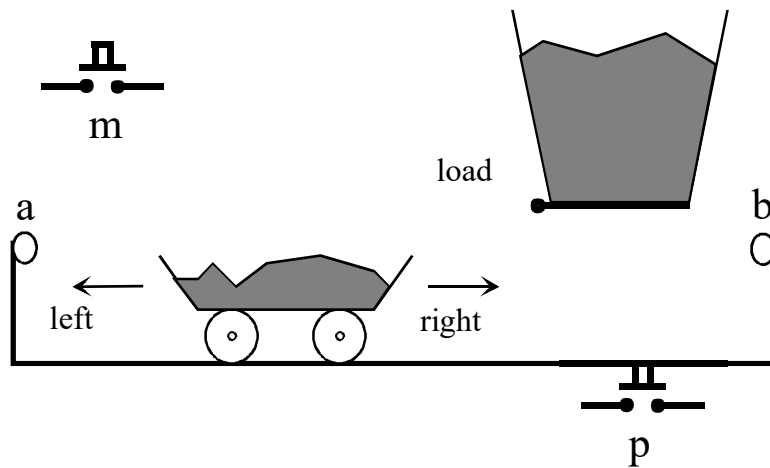
Solutions:





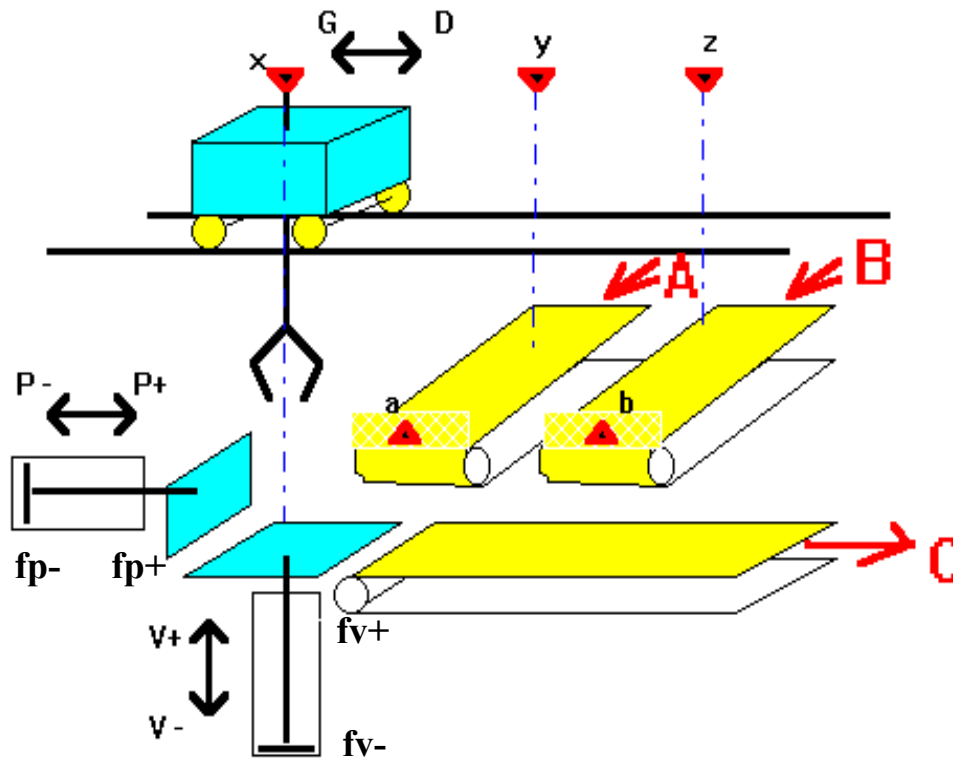
# GRAFCET

## Example 1: modeling a control/automation system



## GRAFCET

## Example 2: modeling a automated transport workcell



\* Conveyor **A** brings parts (sensor **a** detects part ready to lift)

\* Conveyor **B** brings parts (sensor **b** detects part ready to lift)

• Hanging crane, commanded with **D** (**droit**) e **G** (**gauche**), uses sensors **x**, **y** e **z** to detect crane over the base, over **A**, or over **B**, respectively.

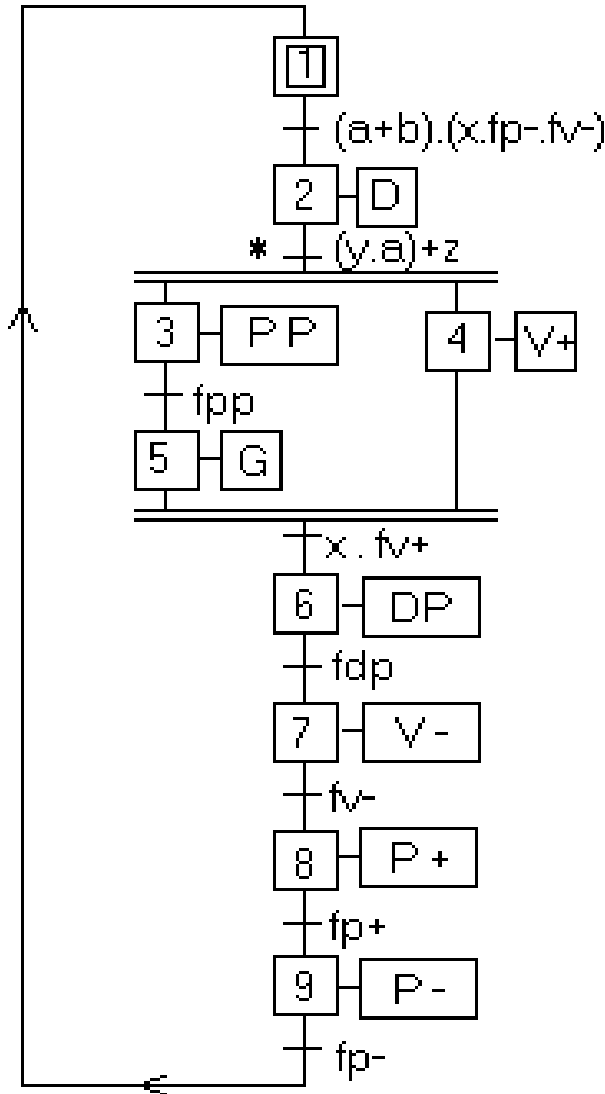
• Clamp of the crane grabs and releases parts with commands **PP** and **DP**. Limit switches **fpp** and **fdp** indicate grabbed and released part. A holding platform has two extreme positions, top and bottom, detected by switches **fv+** and **fv-**. Part release can only be done having the holding platform up.

\* Effector pushes parts with commands **P+** e **P-**. Limit switches **fp+** and **fp-** indicate max and min pushing positions.

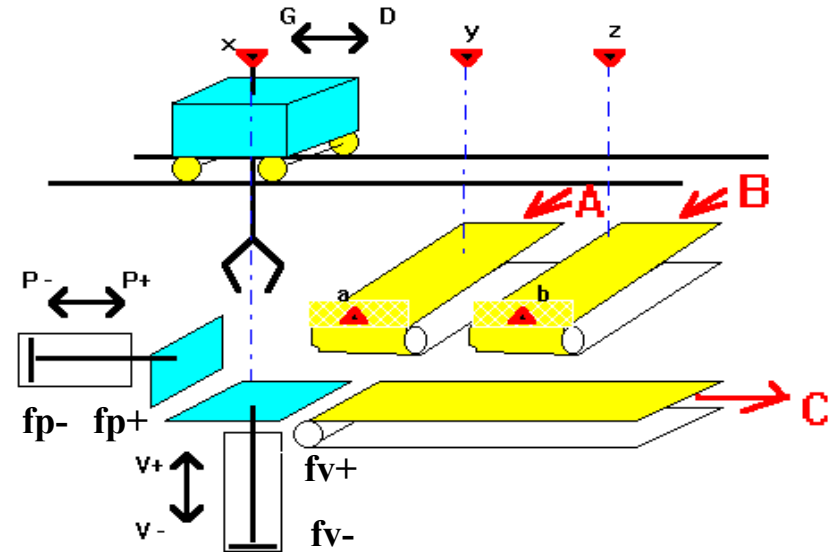
\* The output conveyor is always ON.

\* Conveyors **A** e **B** are commanded by other automata, independent of this workcell.

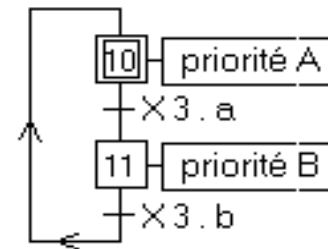
GRAFCET Example 2 (cont)



← *Solution*



To guarantee alternating A and B, modify the program, adding the following GRAFCET:



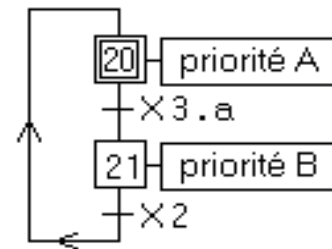
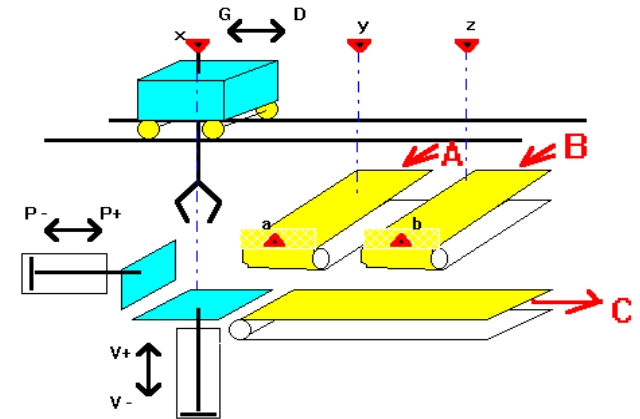
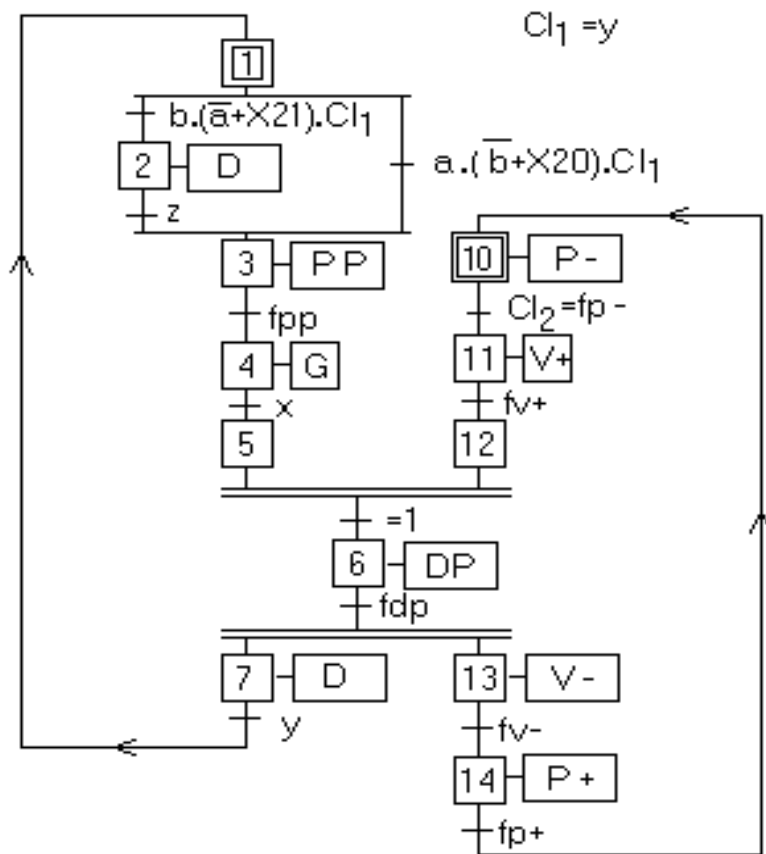
and changing the receptivity function \* to:  $y.a . (\bar{b} + X10) + z$

*Explanation: grab part in y, if there exists part in a and if b has not the priority; if b is true and has priority, then grab part in z.*

*Note: terminology X10 of PL7 changes to S\_1\_10 in Unity Pro*

GRAFCET Example 2 (cont)

*Improved solution:*



- a) After processing one part (P+) prepare immediately to receive the next one:  $fv+$ .
- b) Move crane (D) to an optimal waiting location (i.e. location that reduces delays):  $y$ .

## GRAFCET

## Example 3: modeling and automation of a distribution system

Objective:

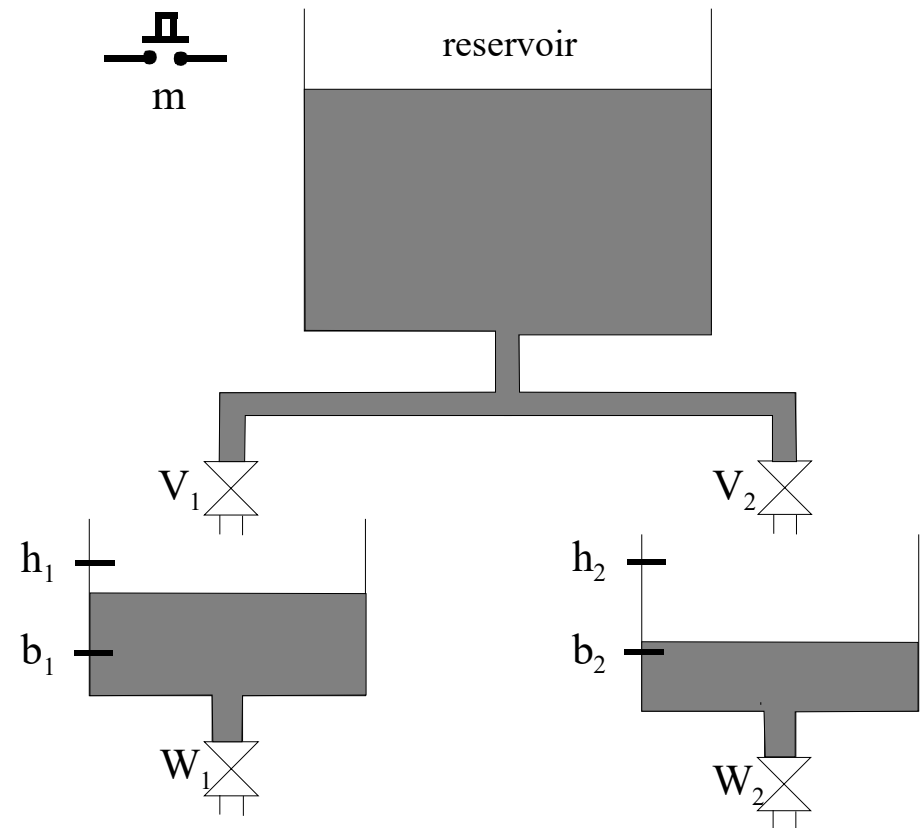
fill 1&amp;2, empty 1&amp;2

refill only after both empty

Sensors:

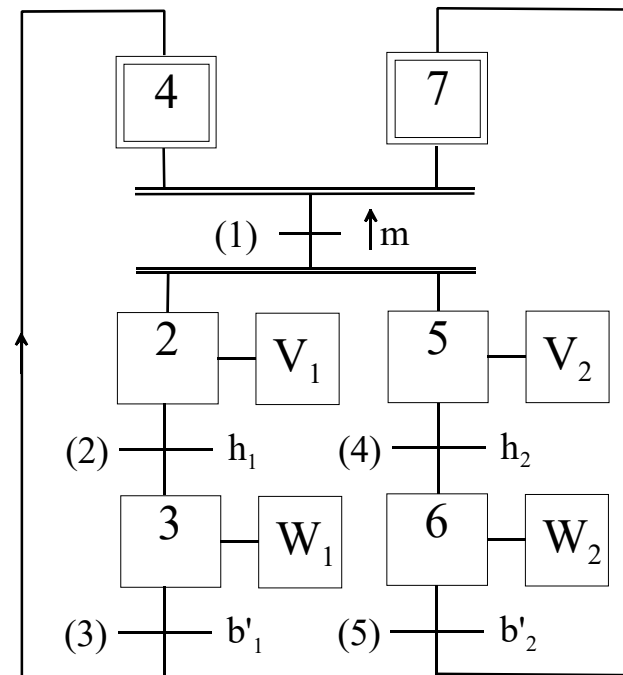
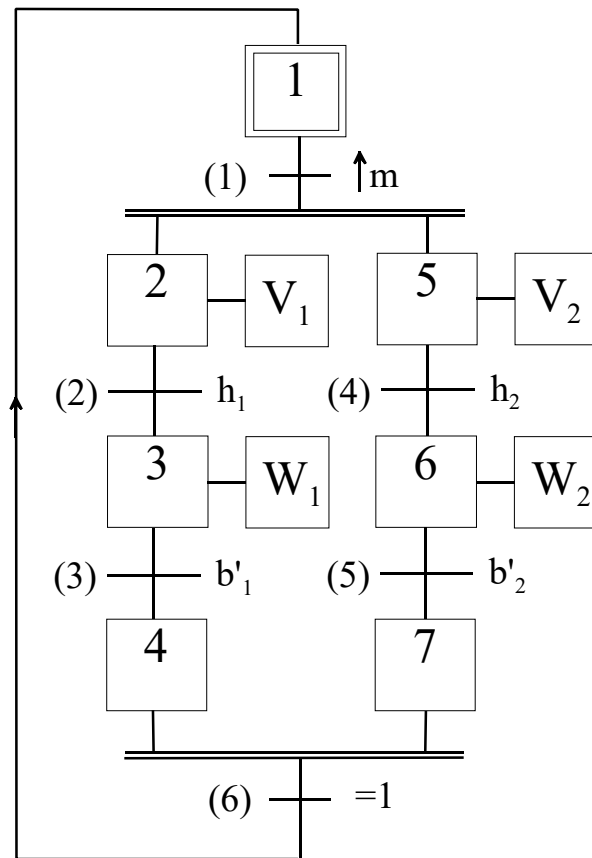
 $m = \text{ON/OFF}$  $b_1, h_1, b_2, h_2 = \text{level}$ 

Actuators:

 $V_1, V_2, W_1, W_2 = \text{admit/exhaust}$ 

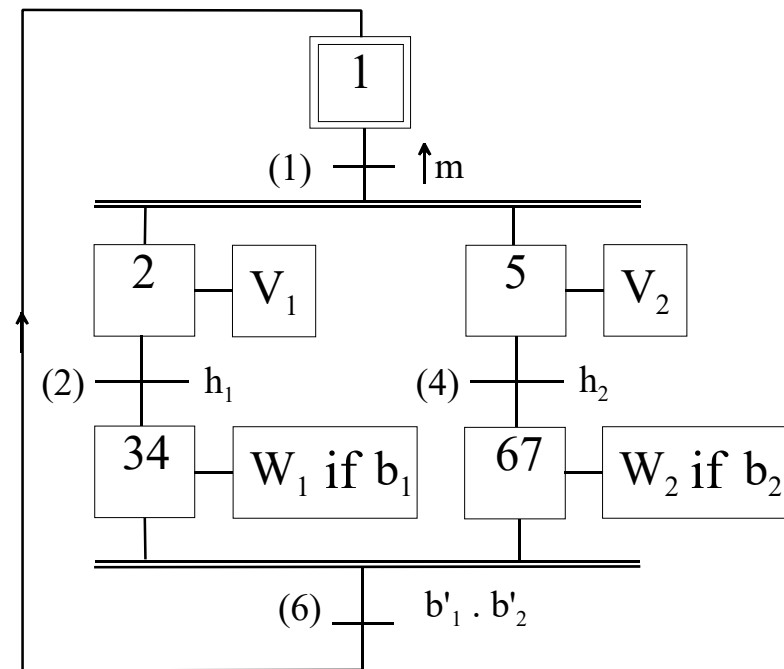
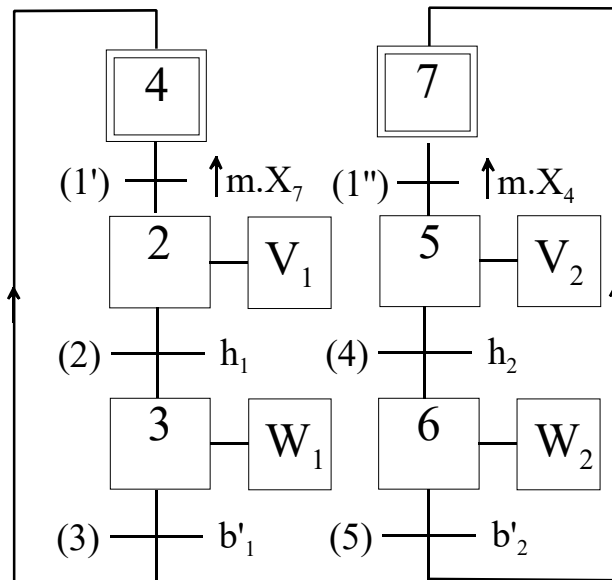
# GRAFCET

## Example 3: modeling and automation of a distribution system



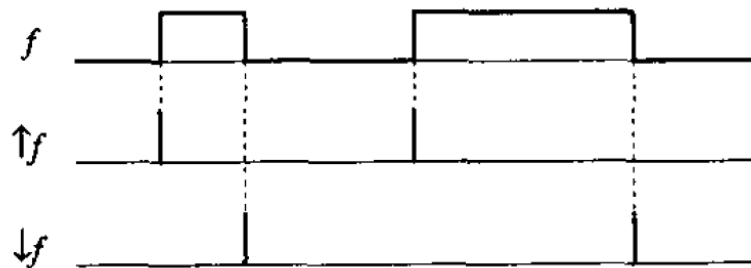
# GRAFCET

## Example 3: modeling and automation of a distribution system

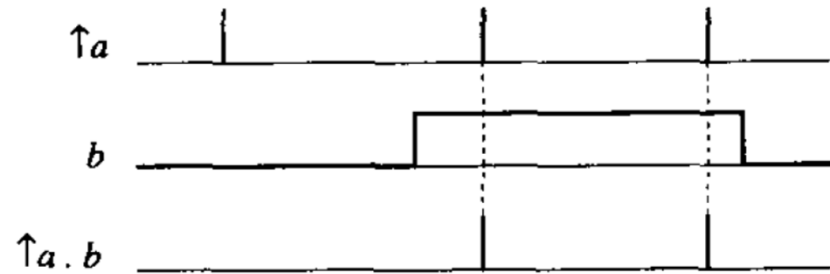


*Note logic including rising edges.*

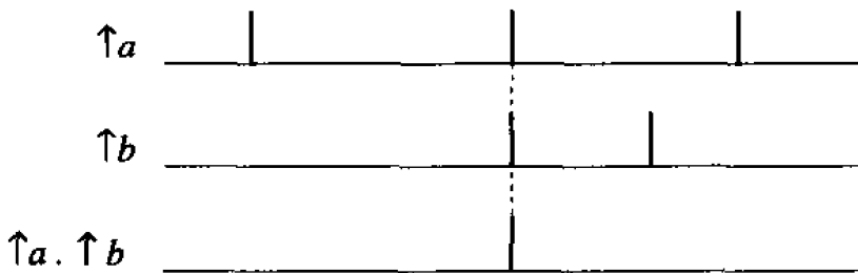
GRAFCET **Transitions can be conditions, events and conditions mixed with events**



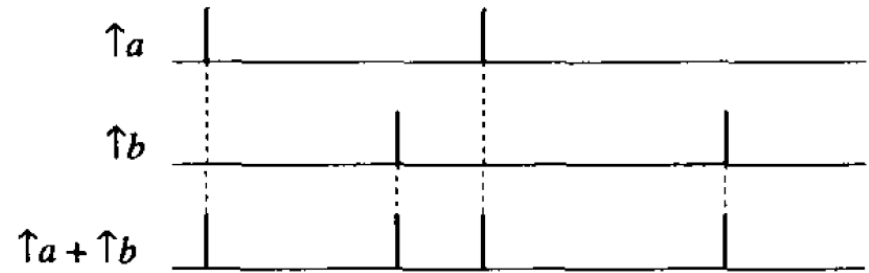
(a) Events  $\uparrow f$  and  $\downarrow f$  obtained from a condition  $f$



(b) Event  $\uparrow a.b$  obtained from event  $\uparrow a$  and condition  $b$



(c) Event  $(\uparrow a . \uparrow b)$  obtained from events  $\uparrow a$  and  $\uparrow b$



(d) Event  $(\uparrow a + \uparrow b)$  obtained from events  $\uparrow a$  and  $\uparrow b$



## GRAFCET **Transitions can be conditions, events and conditions mixed with events**

Properties of events (edge triggers) mixed with conditions (Boolean variables):

$$\uparrow a = \downarrow a'$$

$$\uparrow a . a = \uparrow a, \quad \uparrow a . a' = 0, \quad \downarrow a . a' = \downarrow a, \quad \downarrow a . a = 0$$

$$\uparrow a . \uparrow a = \uparrow a, \quad \uparrow a . \uparrow a' = 0$$

$$\uparrow (a . b) = \uparrow a . b + \uparrow b . a, \quad \uparrow (a + b) = \uparrow a . b' + \uparrow b . a'$$

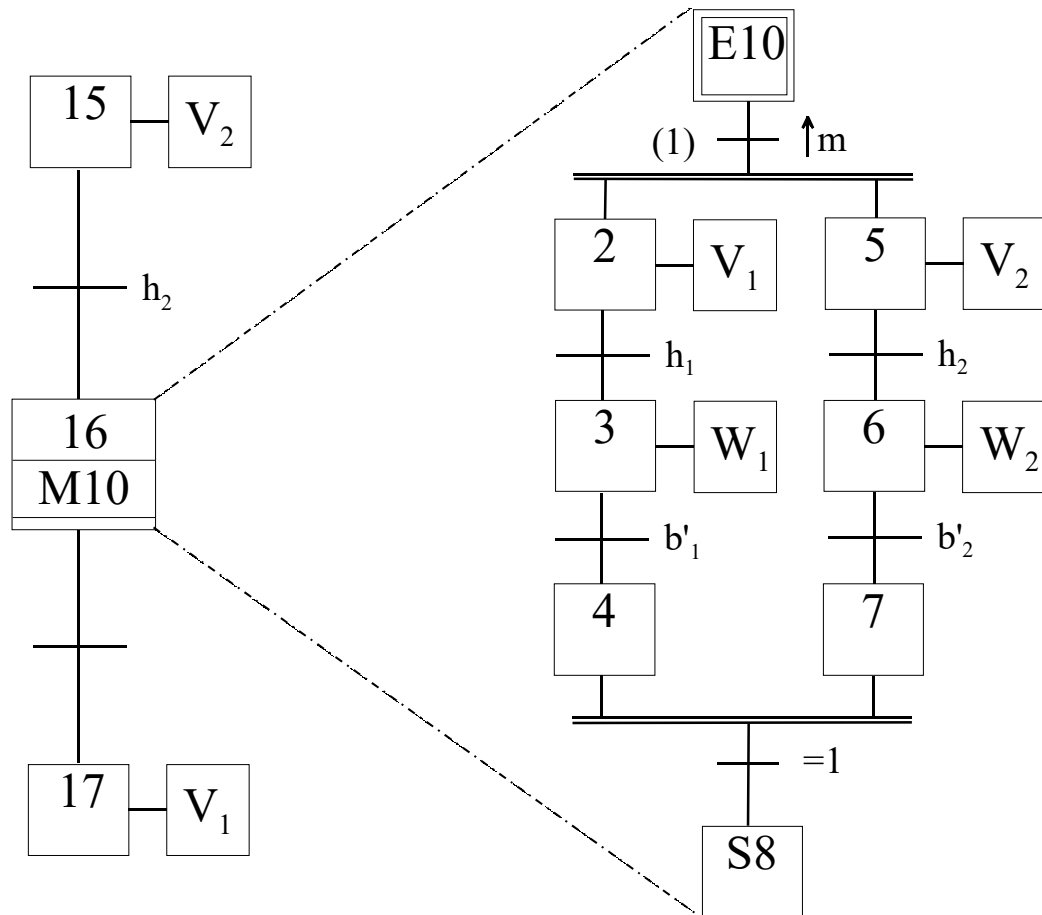
$$\uparrow (a . b) . \uparrow (a . c) = \uparrow (a . b . c)$$

In general, if events a and b are independent

$$\uparrow a . \uparrow b = 0$$

GRAFCET **Other auxiliary mechanisms**

**Macro-steps**



GRAFCET **Other auxiliary mechanisms**

**In PL7 one Step may contain various Macro Actions:**

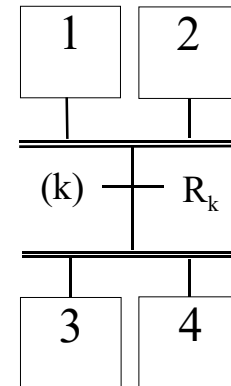
- **Force actions**
- **Enable actions**
- **Mask actions**

*Unity Pro has even more options.*

## GRAFCET Implementation in DOLOG80

**The activity of each Step is stored in an auxiliary memory.**

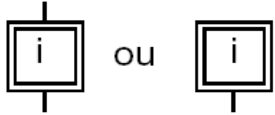
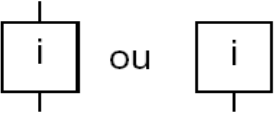
<b>At startup do:</b>	Store $R_k$ evaluation in M100	
AM128		
SLM <sub>x</sub>	AM1	
...	AM2	AM3
AM128	AM100	AM4
SLM <sub>y</sub>	SLM3	RLM1
(initial steps)	AM1	AM3
RLM128	AM2	AM4
	AM100	RLM2
	SLM4	



*Comment: implementing GRAFCET does not require a high level language!*

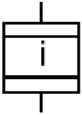
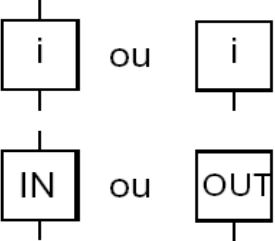
# GRAFCET Implementation in the TSX3722/TSX57

## Steps


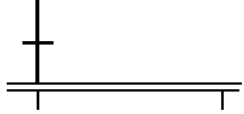
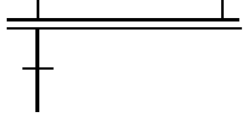
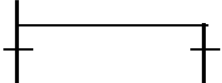
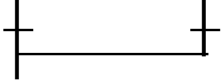
Name	Symbol	Functions
Initial steps (		symbolize the initial active steps at the beginning of the cycle after initialization or re-start from cold.
Simple steps (		show that the automatic system is in a stable condition. The maximum number of steps (including the initial steps) can be configured from: <ul style="list-style-type: none"> <li>● 1 - 96 for a TSX 37-10,</li> <li>● 1 - 128 for a TSX 37-20,</li> <li>● 1 - 250 for a TSX 57.</li> </ul> The maximum number of active steps at the same time can be configured.

# GRAFCET Implementation in the TSX3722/TSX57

## Macro-steps




Name	Symbol	Functions
Macro steps		<p>Symbolize a macro step: a single group of steps and transitions.                      The maximum number of macro steps can only be configured from 0 - 63 for the TSX 57.</p>
Stage of Macro steps		<p>Symbolizes the stages of a macro step.                      The maximum number of stages for each macro step can be configured from 0 - 250 for the TSX 57.</p> <p>Each macro step includes an IN and OUT step.</p>

## GRAFCET Implementation in the TSX3722/TSX57

Name	Symbol	Functions
Transitions		<p>allow the transfer from one step to another. A transition condition associated with this condition is used to define the logic conditions necessary to cross this transition.</p> <p>The maximum number of transitions is 1024. It cannot be configured. The maximum number of valid transitions at the same time can be configured.</p>
AND divergences		<p>Transition from one step to several steps: is used to activate a maximum of 11 steps at the same time.</p>
AND convergences		<p>Transition of several steps to one: is used to deactivate a maximum of 11 steps at the same time.</p>
OR divergences		<p>Transition from one step to several steps: is used to carry out a switch to a maximum of 11 steps.</p>
OR convergences		<p>Transition of several steps to one: is used to end switching from a maximum of 11 steps.</p>

# GRAFCET Implementation in the TSX3722/TSX57

## Arcs/Connectors

Name	Symbol	Functions
Source connectors		"n" is the number of the step "it comes from" (source step).
Destination connector		"n" is the number of the step "it's going to" (target step).
Links directed towards: <ul style="list-style-type: none"> <li>● top</li> <li>● bottom</li> <li>● right or left</li> </ul>		These links are used for switching, jumping a step, restarting steps (sequence).



Information associated with Steps in the GRAFCET:

Name		Description
Bits associated with the steps (1 = active step)	%Xi	Status of the i step of the main Grafcet
		(i from 0 - n) (n depends on the processor)
	%XMj	Status of the j macro step (j from 0 - 63 for TSX/PMX/PCX 57)
	%Xj.i	Status of the i step of the j macro step
	%Xj.IN	Status of the input step of the j macro step
	%Xj.OUT	Status of the output step of the j macro step
System bits associated with Grafcet	%S21	Initializes Grafcet
	%S22	Grafcet resets everything to zero
	%S23	Freezes Grafcet
	%S24	Resets macro steps to 0 according to the system words %SW22 - %SW25
	%S25	Set to 1 when: <ul style="list-style-type: none"> <li>● tables overflow (steps/transition),</li> <li>● an incorrect graph is run (destination connector on a step which does not belong to the graph).</li> </ul>

PL7  
(changed in Unity)

Information associated with Steps in the GRAFCET (bis):

Name		Description
Words associated with steps	%Xi.T	Activity time for main Grafcet step i.
	%Xj.i.T	Activity time for the i step of the j macro step
	%Xj.IN.T	Activity time for the input step of the j macro step
	%Xj.OUT.T	Activity time for the output step of the j macro step
System words associated with Grafcet	%SW20	Word which is used to inform the current cycle of the number of active steps, to be activated and deactivated.
	%SW21	Word which is used to inform the current cycle of the number of valid transitions to be validated or invalidated.
	%SW22 à %SW25	Group of 4 words which are used to indicate the macro steps to be reset to 0 when bit %S24 is set to 1.

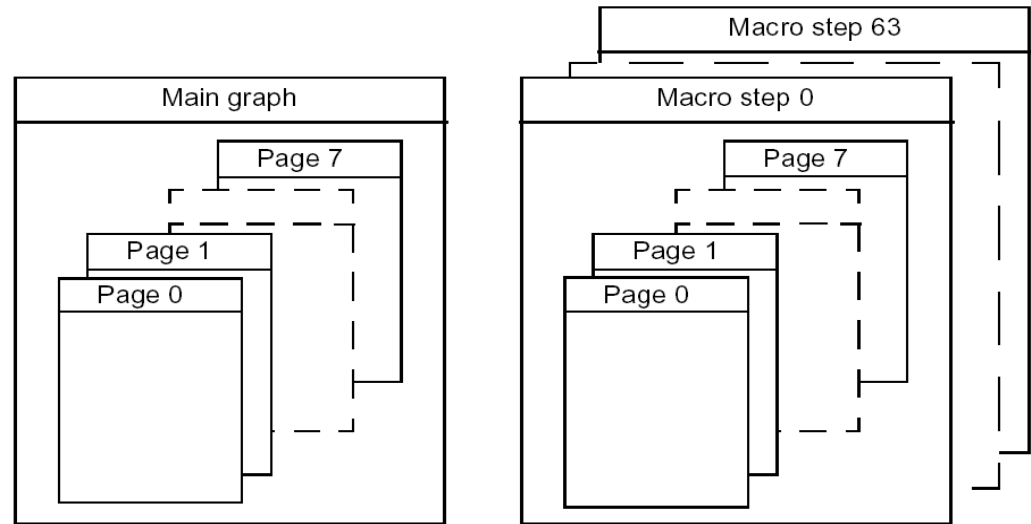
PL7  
(changed in Unity)

And where to find information related with Transitions?

Does not make sense state or activity nor timings  
(only number of occurrences).

# GRAFCET

## General structure:



## Characteristics:

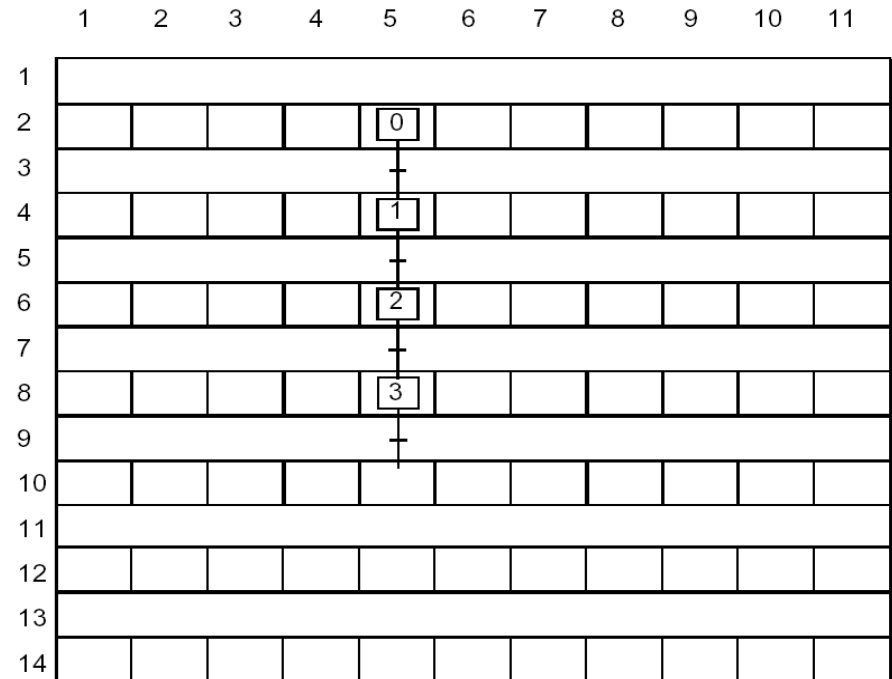
Number	TSX 37 -10		TSX 37 -20		TSX 57	
	Default settings	Maximum	Default settings	Maximum	Default settings	Maximum
Main graph steps	96	96	128	128	128	250
Macro steps	0	0	0	0	8	64
Macro step steps	0	0	0	0	64	250
Step total	96	96	128	128	640	1024
Steps active at the same time	16	96	20	128	40	250
Transitions valid at the same time	20	192	24	256	48	400

## GRAFCET

### Editor: 8 pages

- Pages 0 to 7
- 154 cells (14\*11)

### Characteristics:

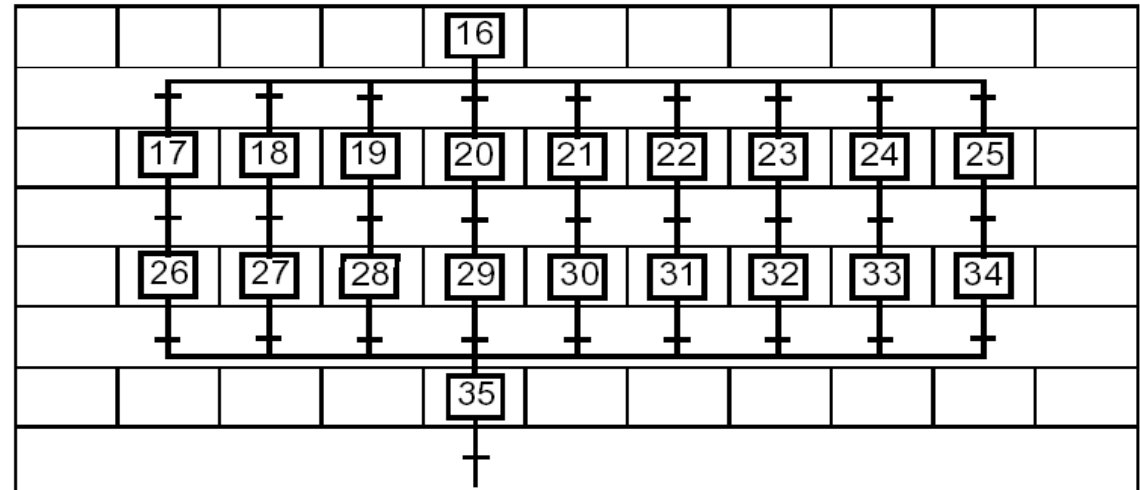


- The first line is used to enter the source connectors.
- The last line is used to enter the destination connectors.
- The even lines (from 2 - 12) are step lines (for destination connector steps),
- The odd lines (from 3 - 13) are transition lines (for transitions and source connectors).
- Each step is located by a different number (0 - 127) in any order.
- Different graphs can be displayed on one page.

# GRAFCET

## OR divergences

(OR convergences)



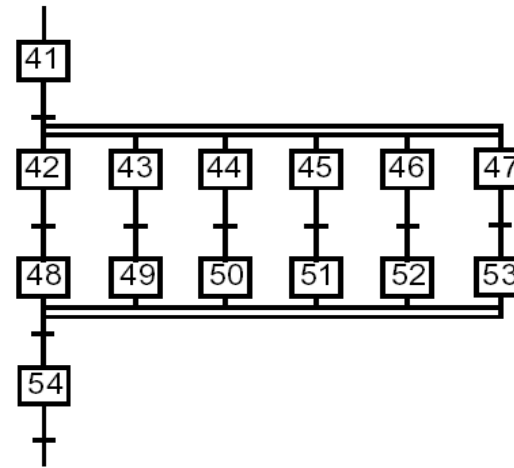
Characteristics:

- The number of transitions upstream of a switching end (OR convergence) or downstream of a switching (OR divergence) must not exceed 11.
- Switching can be to the left or to the right.
- Switching must general finish with switching end.
- To avoid crossing several transitions at the same time, the associated transition conditions must be exclusive.

## GRAFCET

### AND divergences

(AND Convergences)

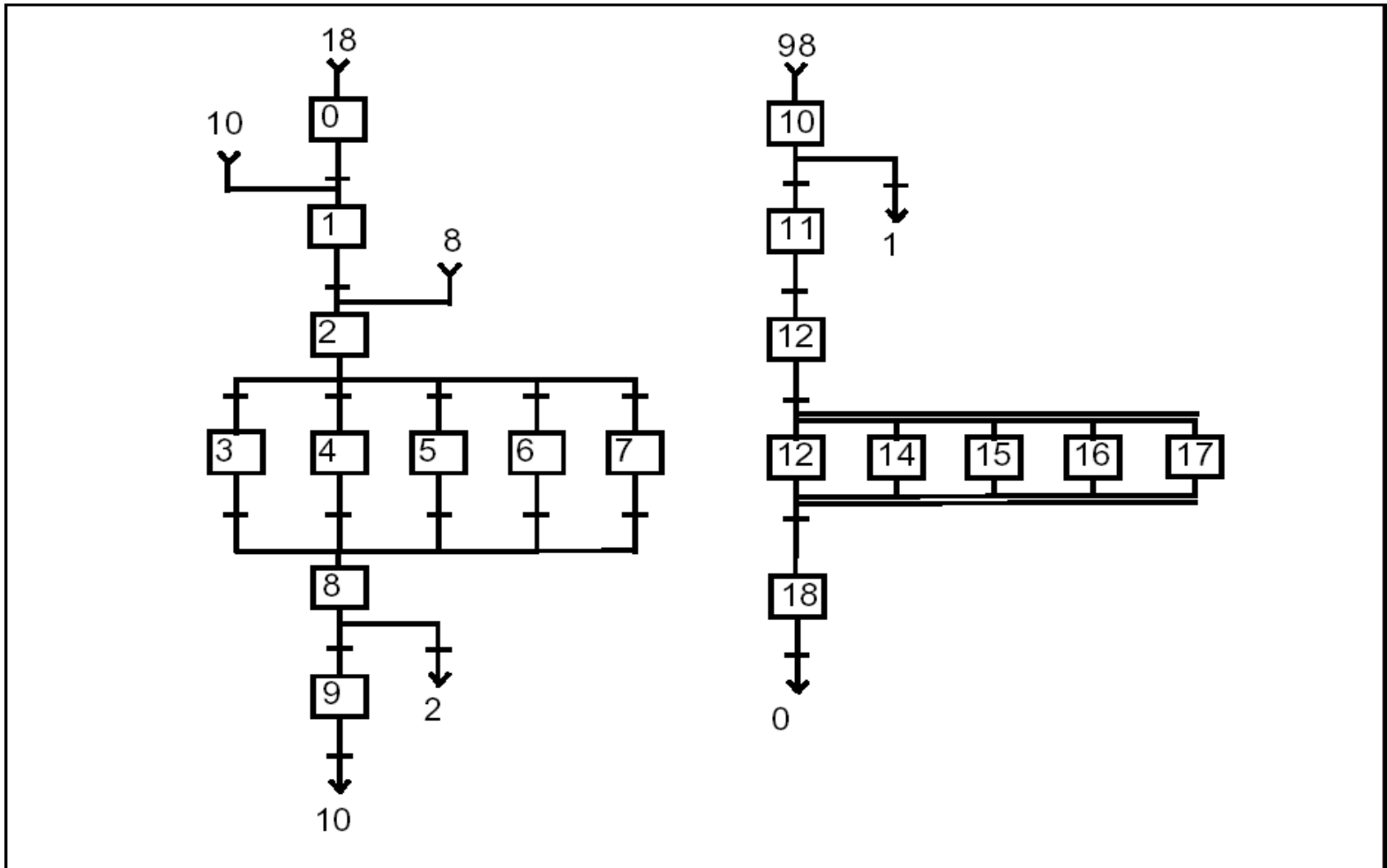


Characteristics:

- The number of steps downstream from a simultaneous activation (AND divergence) or upstream from a simultaneous deactivation (AND convergence) must not exceed 11.
- Simultaneous activation of steps must usually end with a simultaneous deactivation of steps.
- Simultaneous activation is always shown from left to right.
- Simultaneous deactivation is always shown from right to left.

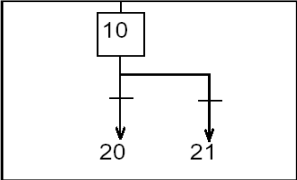
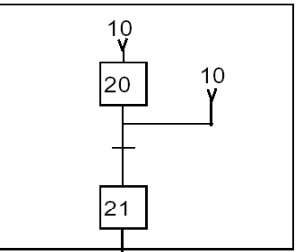
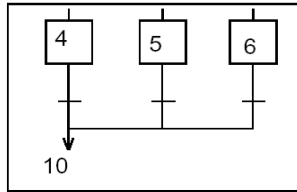
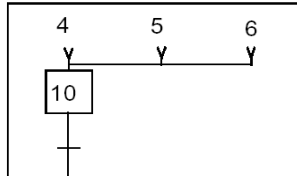
# GRAFCET

## Arcs/Connectors

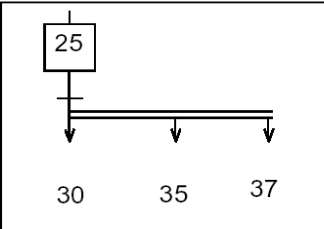
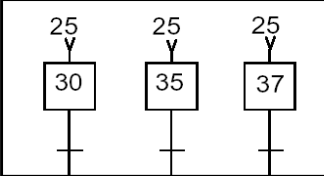
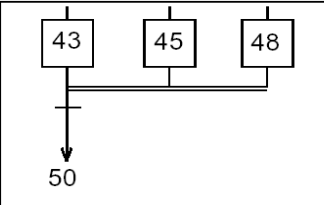
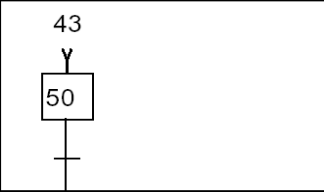


# Rules for divergences and convergences:

## OR

Rule	Illustration
For switching, transitions and destination connectors must be entered on the same page.	 <p style="text-align: right;">Page 1</p>
To end switching, the source connectors must be entered on the same page as the destination step.	 <p style="text-align: right;">Page 2</p>
For an end to switching followed by a return to destination, there must be as many source connectors as steps before the end of switching.	 <p style="text-align: right;">Page 1</p>  <p style="text-align: right;">Page 2</p>

## AND

Rule	Illustration
To activate steps simultaneously, the destination connectors must be on the same page as the divergence step and transition.	 <p style="text-align: right;">Page 2</p>
	 <p style="text-align: right;">Page 3</p>
To deactivate simultaneously, the convergence steps and transition must be on the same page as the destination connector.	 <p style="text-align: right;">Page 1</p>
When several steps converge onto one transition, the source connector has the number of the furthest upstream step on the left.	 <p style="text-align: right;">Page 2</p>



## GRAFCET

### Programming Actions

---

The PL7 software allows three types of action:

- **actions for activation** : actions carried out once when the step with which they are associated passes from the inactive to the active state.
- **actions for deactivation** : actions carried out once when the step with which they are associated passes from the active to the inactive state.
- **continuous actions** : these actions are carried out for as long as the step with which they are associated is active.

**Note:** One action can include several programming elements (sequences or contact networks).

---

These actions are located in the following manner:

MAST - <Grafcet section name> - CHART (or MACROk)- PAGE n %Xi x

with

x = P1 for Activation, x = N1 Continuous, x = P0 Deactivation

n = Page number

i = Step number

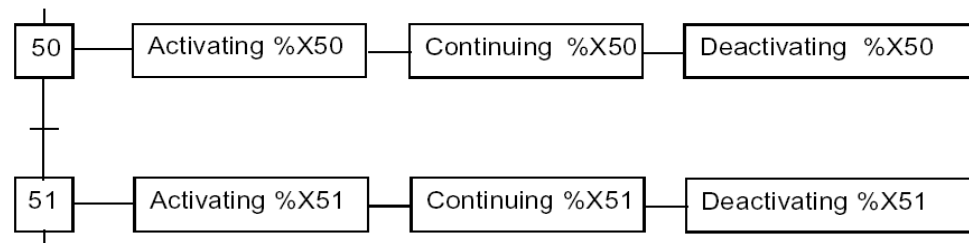
**Example:** MAST - Paint - CHART - PAGE 0 %X1 P1 Action for activating step 1 of page 0 of the Paint section

---

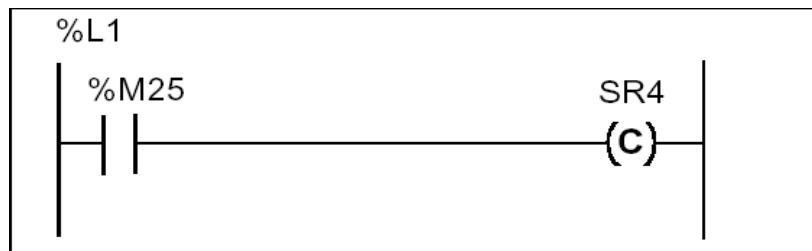
# GRAFCET

## Programming Actions

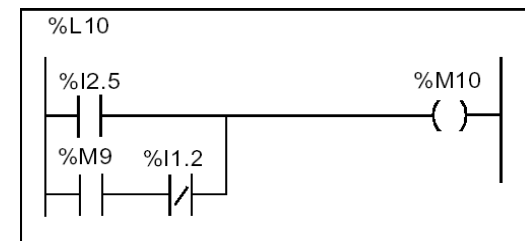
Example of execution of Actions



Example of Activation/deactivation

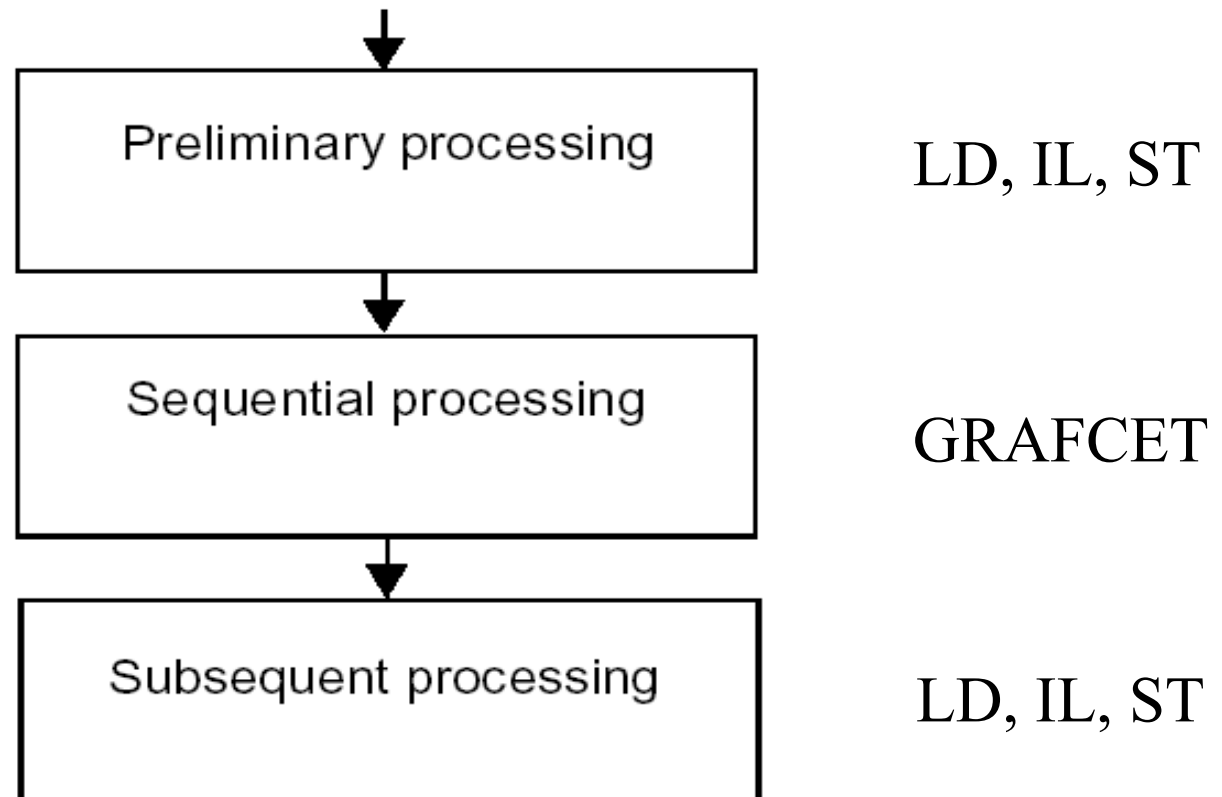


Example of continuous Action



## GRAFCET

### GRAFCET Section Structure



## GRAFCET

### GRAFCET Section Initialization

---

Initializing the Grafcet is done by the system bit %S21.  
Normally set at state 0, setting %S21 to 1 causes:

- active steps to deactivate,
  - initial steps to activate.
- 

The following table gives the different possibilities for setting to the system bit %S21 to 1 and 0.

Set to 1	Reset to 0
<ul style="list-style-type: none"><li>● By setting %S0 to 1</li><li>● By the user program</li><li>● By the terminal (in debugging or animation table)</li></ul>	<ul style="list-style-type: none"><li>● By the system at the beginning of the process</li><li>● By the user program</li><li>● By the terminal (in debugging or animation table)</li></ul>

---

## GRAFCET

### GRAFCET Section Reset

---

The system bit %S22 resets Grafcet to 0.

Normally set at 0, setting %S22 to 1 causes active steps in the whole of the sequential process to deactivate.

**Note:** The RESET\_XIT function used to reinitialize via the program the step activity time of all the steps of the sequential processing. (See (See Reference Manual, Volume 2)).

---

The following table gives the different possibilities for setting to the system bit %S22 to 1 and 0.

Set to 1	Reset to 0
<ul style="list-style-type: none"><li>● By the user program</li><li>● By the terminal (in debugging or animation table)</li></ul>	<ul style="list-style-type: none"><li>● By the system at the end of the sequential process</li></ul>

---