Industrial Automation
(Automação de Processos Industriais)

GRAFCET
(Sequential Function Chart)

http://users.isr.ist.utl.pt/~jag/courses/api1415/api1415.html

Slides 2010/2011 Prof. Paulo Jorge Oliveira
Rev. 2011-2015 Prof. José Gaspar
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]
Ladder Diagram

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

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)

```
  1
(1) m
  2 Right
  3  b
 (2) Load
  3  p
 (3)  Left
  4
(2) a
```
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)

[http://www-ipst.u-strasbg.fr/pat/autom/grafce_t.htm](http://www-ipst.u-strasbg.fr/pat/autom/grafce_t.htm)

[http://www.automationstudio.com](http://www.automationstudio.com) (See projects)

Bibliography:

- Petri Nets and GRAFCET: Tools for Modelling Discrete Event Systems  

- 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/](http://www.lurpa.ens-cachan.fr/grafcet/)
General Information about SFC Sequence Language

- Initial Step
- Transition Condition
- Parallel Branch
- Step
- Alternative Branch
- Jump
- Transition Condition
- Alternative Junction
- Parallel Junction
- Transition Condition (Boolean Value of a Transition Section)
- Transition Condition (Boolean Variable)
- Return Value (of a transition section)
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“, by I.E.C. 60848.
**GRAFCET**  **Basic Elements**

**Steps**
- Inactive: 1
- Active: 2
- Initial: 3

**Connections**
- Directed Arc

**Transitions**
- Simple
  - (1) \( R_1 \)
- Joint (parallel junction)
  - (2) \( R_2 \)
- Fork (parallel branch)
  - (3) \( R_3 \)
- Joint e fork
  - (3) \( R_3 \)

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

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.
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.

Example 1

Example 2

Example 3
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.

AND Divergences:

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

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

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

OR Convergences:

If Step 1 active and \( a \) TRUE then deactivate Step 1 and activate Step 3 (state of Step 2 remains unchanged).

(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 Convergences:

If Steps 1 and 2 active and \( a \) TRUE then deactivate Steps 1 and 2 and activate Step 3.
Example:

GRAFCET state evolution

Level activated Action. Actions can also be activated during transitions - see next.
Given 4 Steps (1 to 4) and 2 Transitions (t1 and t2) write a segment of GRAFCET to solve the following problem:

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

- if t1 is TRUE, activate Step 3 (and deactivate Steps 1 and 2);
- if t2 is TRUE, activate Step 4 (and deactivate Steps 1 and 2);
- otherwise, the state is maintained.
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:

- Transition 1 priority
- Three mutually exclusive hypotheses
Example 1: modeling a control/automation system
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.
To guarantee alternating A and B, modify the program, adding the following GRAFCET:

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

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.
Example 3: modeling and automation of a distribution system

Objective:
fill 1&2, empty 1&2
refill only after both empty

Sensors:
m = ON/OFF
b₁, h₁, b₂, h₂ = level

Actuators:
V₁, V₂, W₁ W₂ = admit/exhaust
Example 3: modeling and automation of a distribution system

GRAFCET

\[
\begin{align*}
(1) & \quad h_1 \\
(2) & \quad h_1 \\
(3) & \quad b'_1 \\
(4) & \quad h_2 \\
(5) & \quad b'_2 \\
(6) & \quad =1
\end{align*}
\]
Example 3: modeling and automation of a distribution system
Transitions can be conditions, events and conditions mixed with events

(a) Events ↑f and ↓f obtained from a condition f

(b) Event ↑a.b obtained from event ↑a and condition b

(c) Event (↑a . ↑b) obtained from events ↑a and ↑ b

(d) Event (↑a + ↑b) obtained from events ↑a and ↑ b
GRAFCET Transitions can be conditions, events and conditions mixed with events

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

\( \mathbf{6}a = \downarrow a' \)

\( \mathbf{6}a . a = \mathbf{6}a, \quad \mathbf{6}a . a' = 0, \quad \mathbf{8}a . a' = \mathbf{8}a, \quad \mathbf{8}a . a = 0 \)

\( \mathbf{6}a . \mathbf{6}a = \mathbf{6}a, \quad \mathbf{6}a . \mathbf{6}a' = 0 \)

\( \mathbf{6}(a . b) = \mathbf{6}a . b + \mathbf{6}a . a, \quad \mathbf{6}(a + b) = \mathbf{6}a . b' + \mathbf{6}a . a' \)

\( \mathbf{6}(a . b) . \mathbf{6}(a . c) = \mathbf{6}(a . b . c) \)

In general, if events a and b are independent

\( \mathbf{6}a . \mathbf{6}b = 0 \)
GRAFCET  Other auxiliary mechanisms

Macro-steps
Other auxiliary mechanisms

Pseudo Macro-steps

Macro Actions

• Force actions

• Enable actions

• Mask actions
GRAFCET Implementation in DOLOG80

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

At startup do:

<table>
<thead>
<tr>
<th>AM128</th>
<th>SLMx</th>
<th>...</th>
<th>AM128</th>
<th>SLMy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store R_k evaluation in M100</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AM1</th>
<th>AM2</th>
<th>AM3</th>
<th>AM4</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLM1</td>
<td>SLM2</td>
<td>SLM3</td>
<td>RLM1</td>
</tr>
<tr>
<td>(initial steps)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AM1</td>
<td>AM2</td>
<td>AM3</td>
<td>AM4</td>
</tr>
<tr>
<td>RLM128</td>
<td>AM100</td>
<td>RLM2</td>
<td>SLM4</td>
</tr>
</tbody>
</table>

Comment: implementing GRAFCET does not need a high level language!
GRAFCET Implementation in the TSX3722/TSX57

Steps

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial steps (</td>
<td><img src="image" alt="Symbol" /></td>
<td>symbolize the initial active steps at the beginning of the cycle after initialization or re-start from cold.</td>
</tr>
<tr>
<td>Simple steps (</td>
<td><img src="image" alt="Symbol" /></td>
<td>show that the automatic system is in a stable condition. The maximum number of steps (including the initial steps) can be configured from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 - 96 for a TSX 37-10,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 - 128 for a TSX 37-20,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1 - 250 for a TSX 57.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum number of active steps at the same time can be configured.</td>
</tr>
</tbody>
</table>
### Macro-steps

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macro steps</td>
<td>![Macro step symbol]</td>
<td>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.</td>
</tr>
<tr>
<td>Stage of Macro steps</td>
<td>![Stage of macro step symbol]</td>
<td>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. Each macro step includes an IN and OUT step.</td>
</tr>
</tbody>
</table>
## Implementation in the TSX3722/TSX57

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transitions</td>
<td>![Symbol]</td>
<td>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. The maximum number of transitions is 1024. It cannot be configured. The maximum number of valid transitions at the same time can be configured.</td>
</tr>
<tr>
<td>AND divergences</td>
<td>![Symbol]</td>
<td>Transition from one step to several steps: is used to activate a maximum of 11 steps at the same time.</td>
</tr>
<tr>
<td>AND convergences</td>
<td>![Symbol]</td>
<td>Transition of several steps to one: is used to deactivate a maximum of 11 steps at the same time.</td>
</tr>
<tr>
<td>OR divergences</td>
<td>![Symbol]</td>
<td>Transition from one step to several steps: is used to carry out a switch to a maximum of 11 steps.</td>
</tr>
<tr>
<td>OR convergences</td>
<td>![Symbol]</td>
<td>Transition of several steps to one: is used to end switching from a maximum of 11 steps.</td>
</tr>
</tbody>
</table>
### Implementation in the TSX3722/TSX57

#### Arcs/Connectors

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source connectors</td>
<td><img src="symbol" alt="n" /></td>
<td>&quot;n&quot; is the number of the step &quot;it comes from&quot; (source step).</td>
</tr>
<tr>
<td>Destination connector</td>
<td><img src="symbol" alt="n" /></td>
<td>&quot;n&quot; is the number of the step &quot;it's going to&quot; (target step).</td>
</tr>
<tr>
<td>Links directed towards:</td>
<td>![symbol]</td>
<td>These links are used for switching, jumping a step, restarting steps (sequence).</td>
</tr>
</tbody>
</table>
  * top
  * bottom
  * right or left
Information associated with Steps in the GRAFCET:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits associated with the steps (1 = active step)</td>
<td></td>
</tr>
<tr>
<td>%Xi</td>
<td>Status of the i step of the main Grafcet (i from 0 - n) (n depends on the processor)</td>
</tr>
<tr>
<td>%XMj</td>
<td>Status of the j macro step (j from 0 - 63 for TSX/PMX/PCX 57)</td>
</tr>
<tr>
<td>%Xj.i</td>
<td>Status of the i step of the j macro step</td>
</tr>
<tr>
<td>%Xj.IN</td>
<td>Status of the input step of the j macro step</td>
</tr>
<tr>
<td>%Xj.OUT</td>
<td>Status of the output step of the j macro step</td>
</tr>
<tr>
<td>System bits associated with Grafcet</td>
<td></td>
</tr>
<tr>
<td>%S21</td>
<td>Initializes Grafcet</td>
</tr>
<tr>
<td>%S22</td>
<td>Grafcet resets everything to zero</td>
</tr>
<tr>
<td>%S23</td>
<td>Freezes Grafcet</td>
</tr>
<tr>
<td>%S24</td>
<td>Resets macro steps to 0 according to the system words %SW22 - %SW25</td>
</tr>
<tr>
<td>%S25</td>
<td>Set to 1 when:</td>
</tr>
<tr>
<td></td>
<td>•  tables overflow (steps/transition),</td>
</tr>
<tr>
<td></td>
<td>•  an incorrect graph is run (destination connector on a step which does not belong to the graph).</td>
</tr>
</tbody>
</table>
Information associated with Steps in the GRAFCET (bis):

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

And where to find information related with Transitions?

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

GRAFCET Section Structure

1. Preliminary processing
2. Sequential processing
3. Subsequent processing

LD, IL, ST
LD, IL, ST
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.

<table>
<thead>
<tr>
<th>Set to 1</th>
<th>Reset to 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>- By setting %S0 to 1</td>
<td>- By the system at the beginning of the process</td>
</tr>
<tr>
<td>- By the user program</td>
<td>- By the user program</td>
</tr>
<tr>
<td>- By the terminal (in debugging or animation table)</td>
<td>- By the terminal (in debugging or animation table)</td>
</tr>
</tbody>
</table>
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 Reference Manual, Volume 2).

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

<table>
<thead>
<tr>
<th>Set to 1</th>
<th>Reset to 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>• By the user program</td>
<td>• By the system at the end of the sequential process</td>
</tr>
<tr>
<td>• By the terminal (in debugging or animation table)</td>
<td></td>
</tr>
</tbody>
</table>
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.