

Industrial Automation

(Automação de Processos Industriais)

GRAFCET

(Sequential Function Chart) 3/3

<http://users.isr.ist.utl.pt/~jag/courses/api1819/api1819.html>

Prof. José Gaspar, rev. 2018/2019

GRAFCET vs Ladder

GRAFCET/SFC can be converted directly to ladder logic

Memory variables:

Assign one Boolean variable to each step (s_i) and transition (t_j)

Code to run once:

1. Initialize steps and transitions

Code to run at **every scan cycle:**

2. Check transitions and activate steps

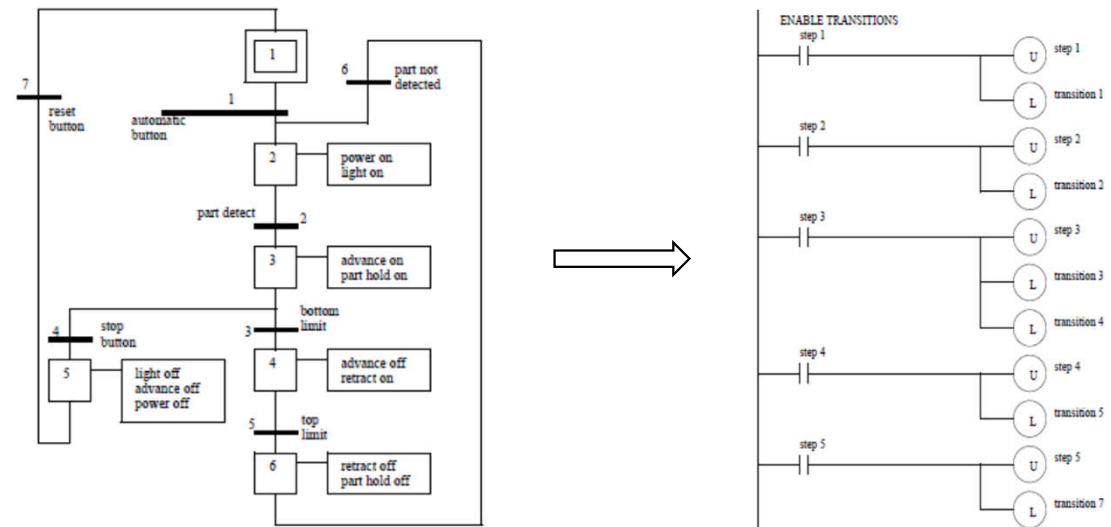
3. Perform activities for steps

4. Enable transitions (jump to 2.)

Ref: [Hugh Jack 2008]

GRAFCET/SFC can be converted directly to ladder logic

Ref: [Hugh Jack 2008]



Memory variables:

Assign one Boolean variable to each step (s_i) and transition (t_j).

Create memories to keep output values.

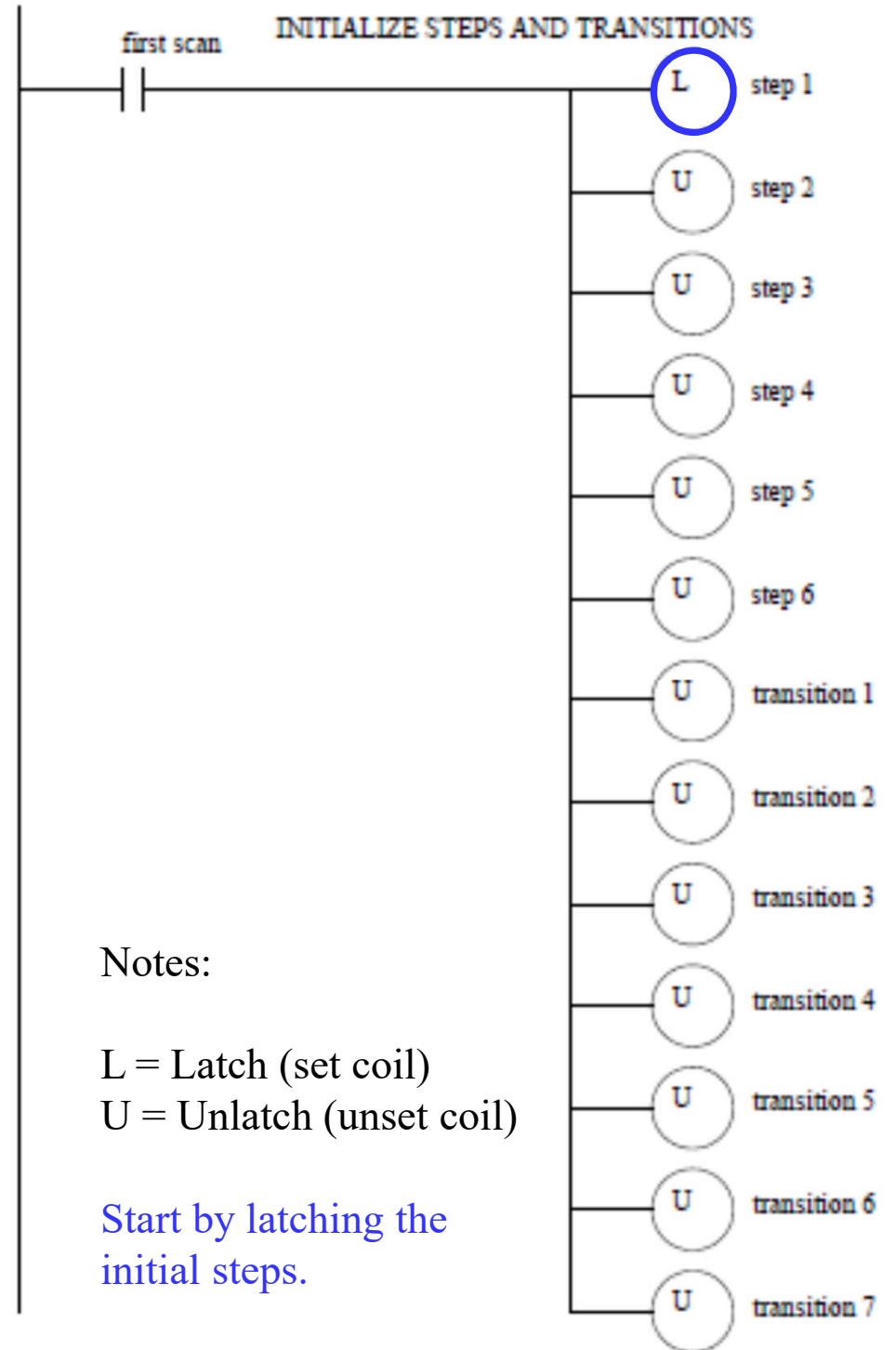
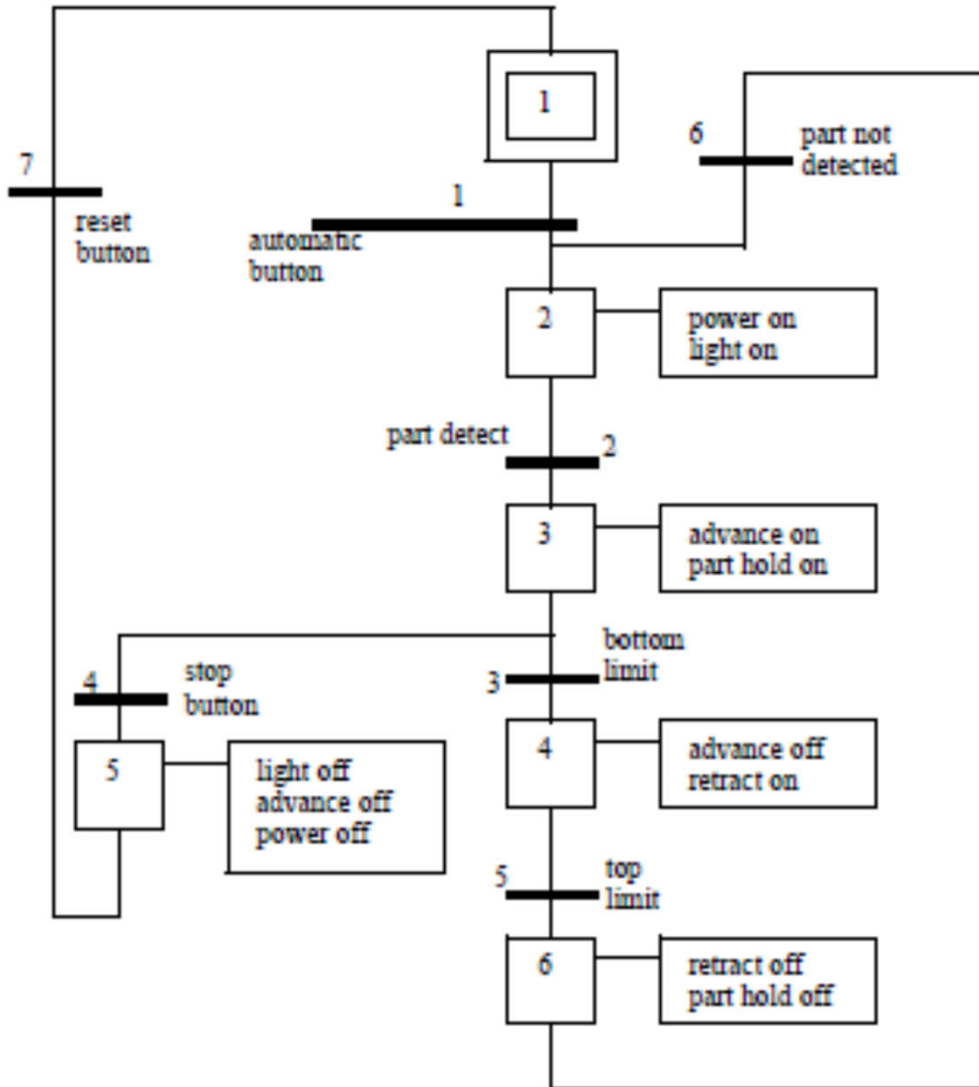
Code to run once:

1. Initialize steps and transitions

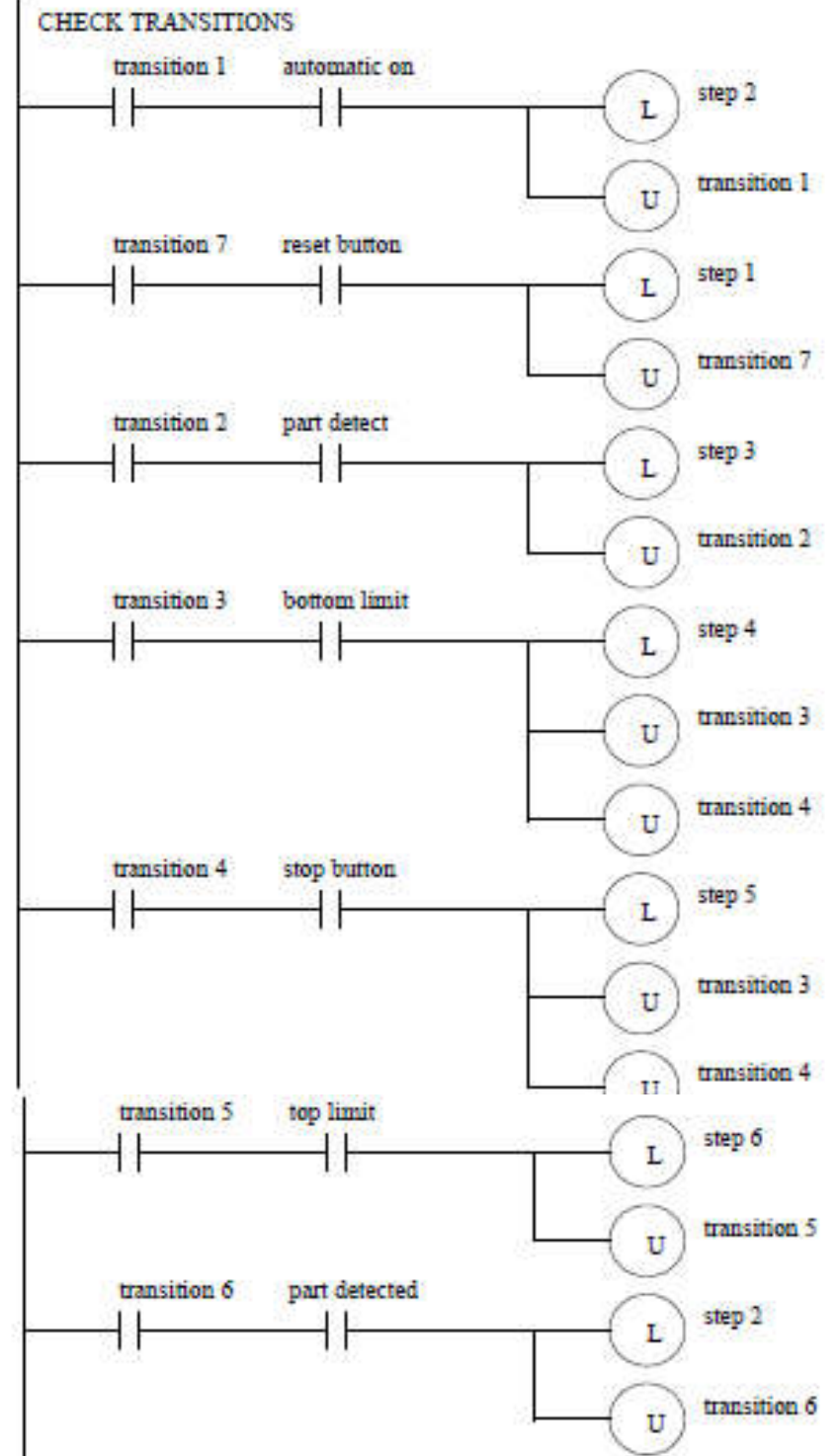
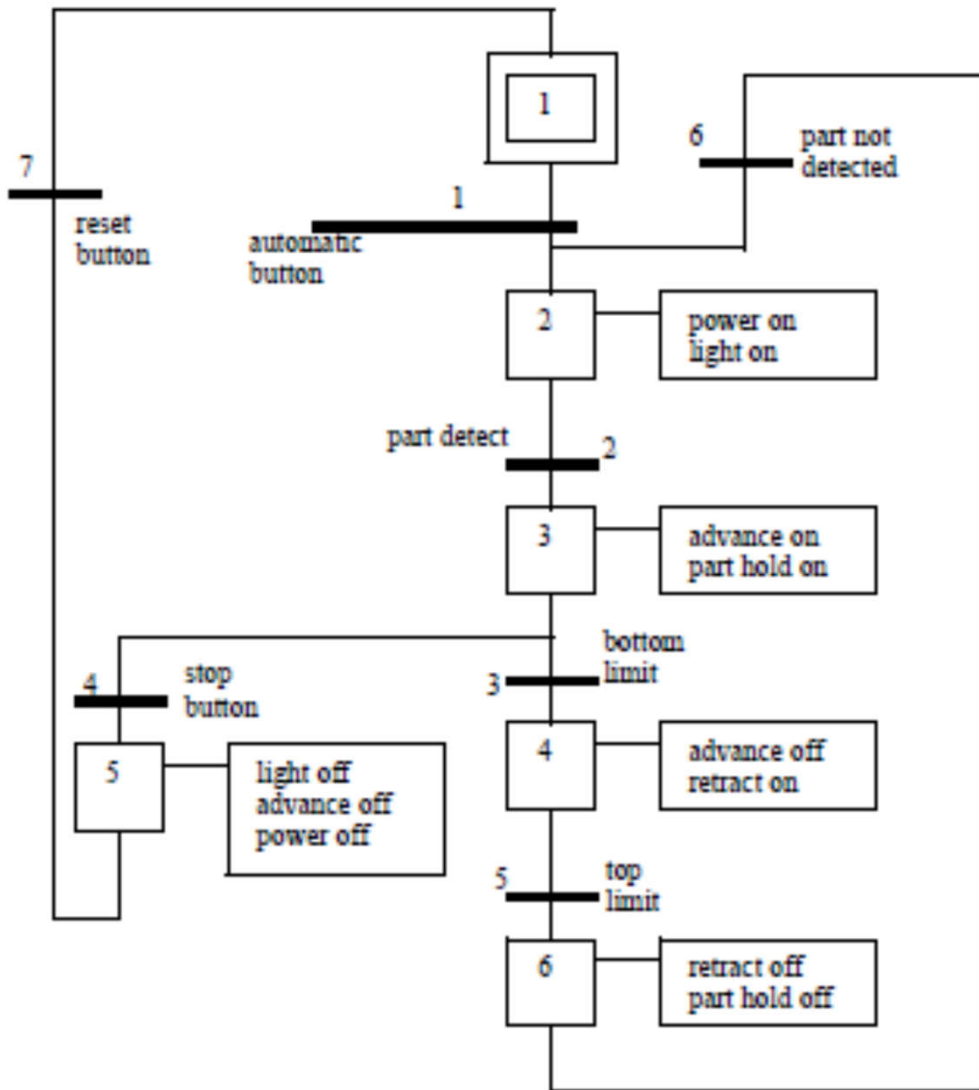
Code to run at every scan cycle:

2. Check transitions and activate steps
3. Perform activities for steps
4. Enable transitions (jump to 2.)

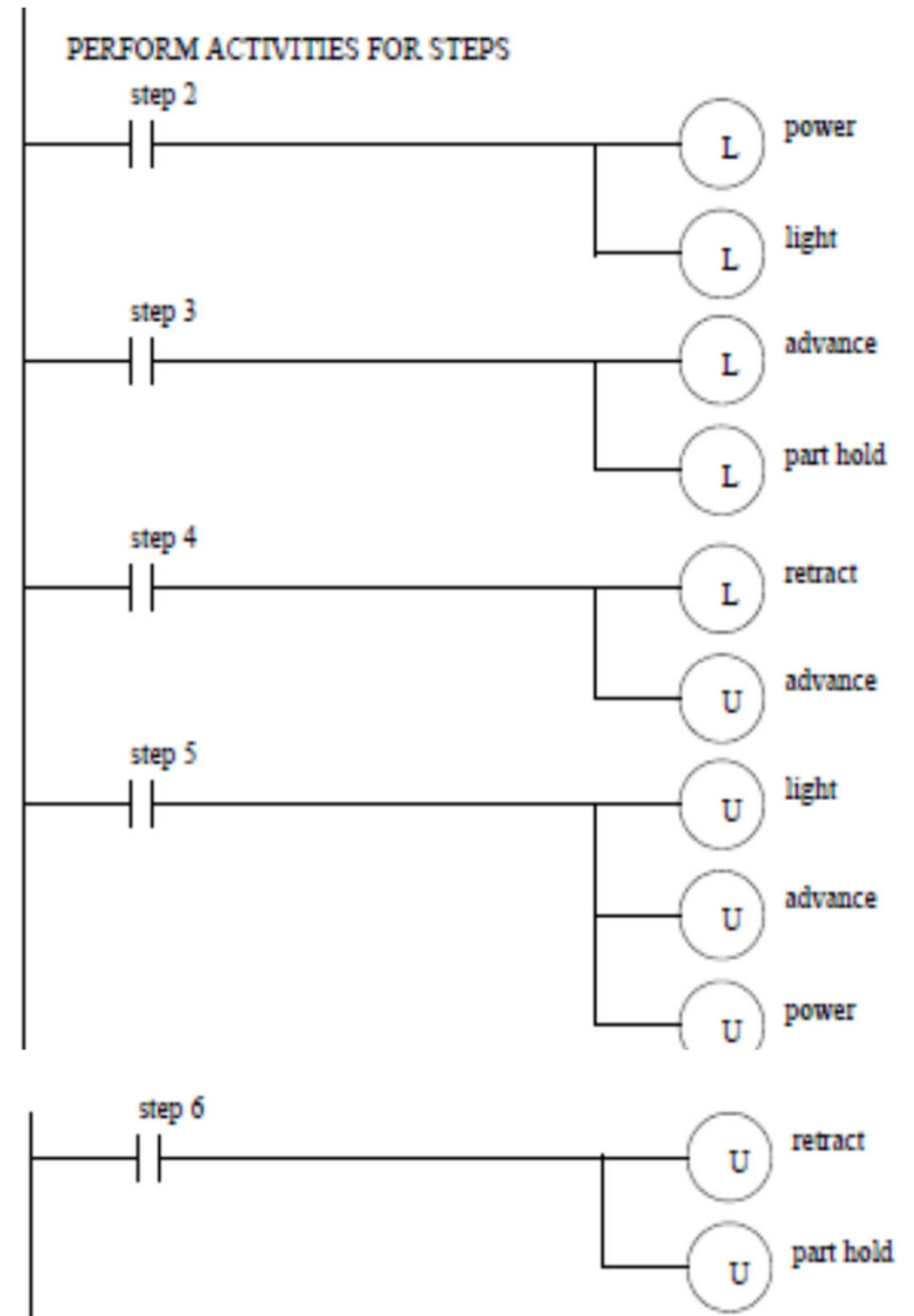
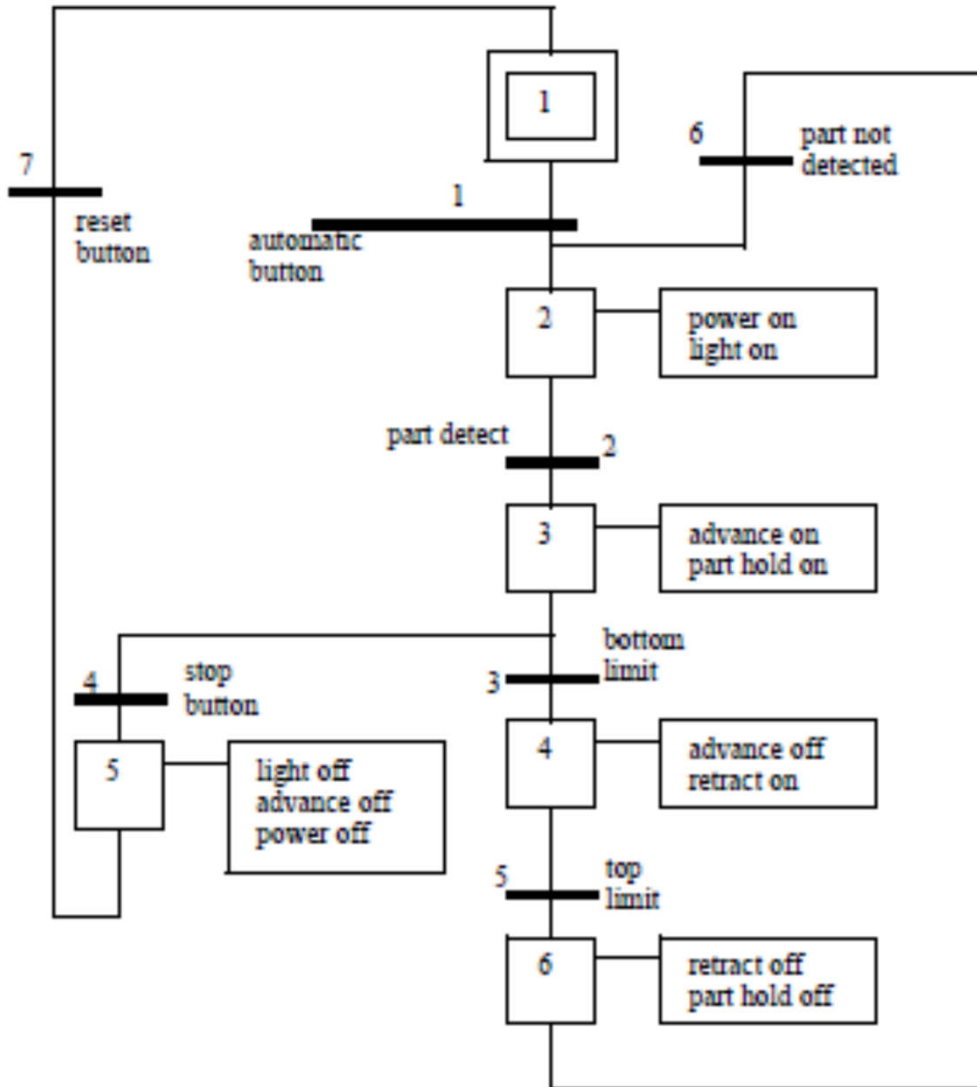
1. Initialize steps and transitions



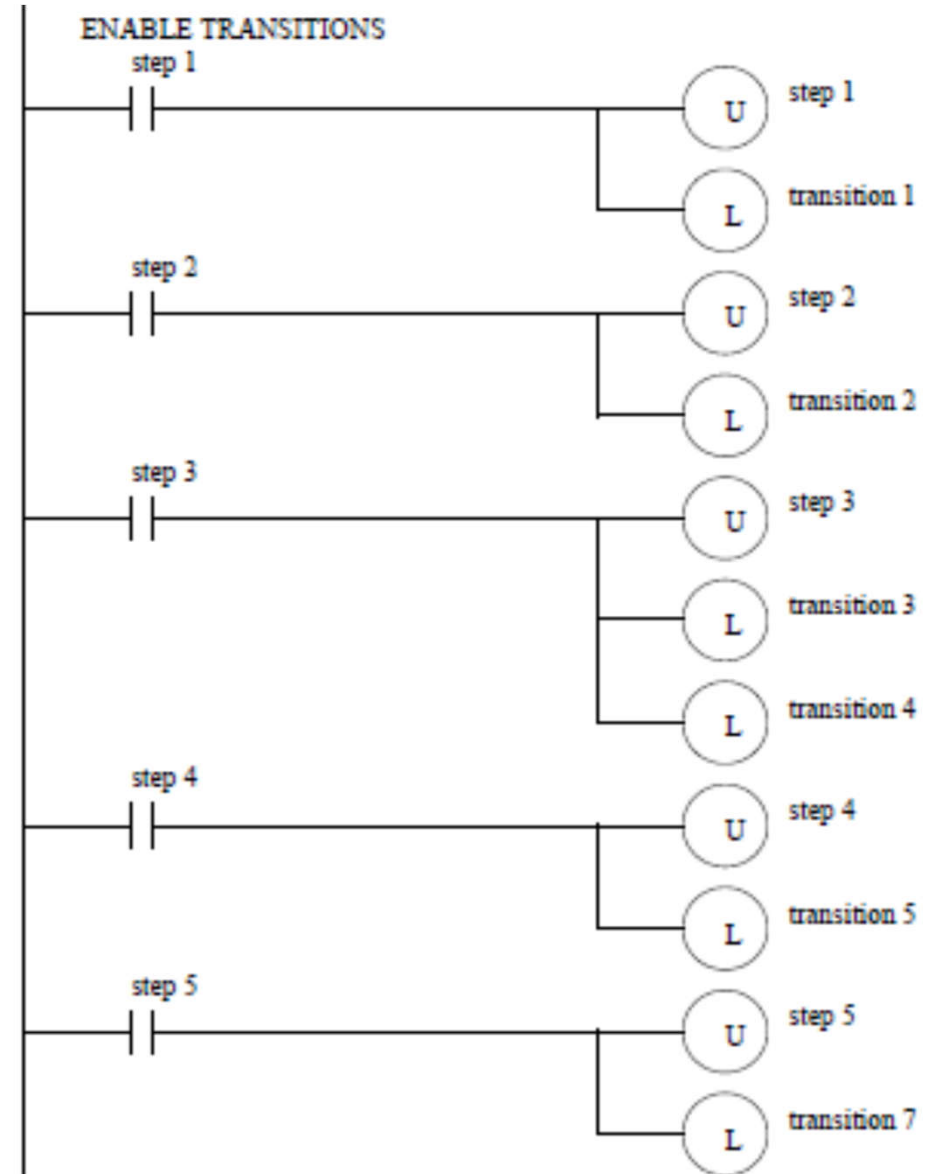
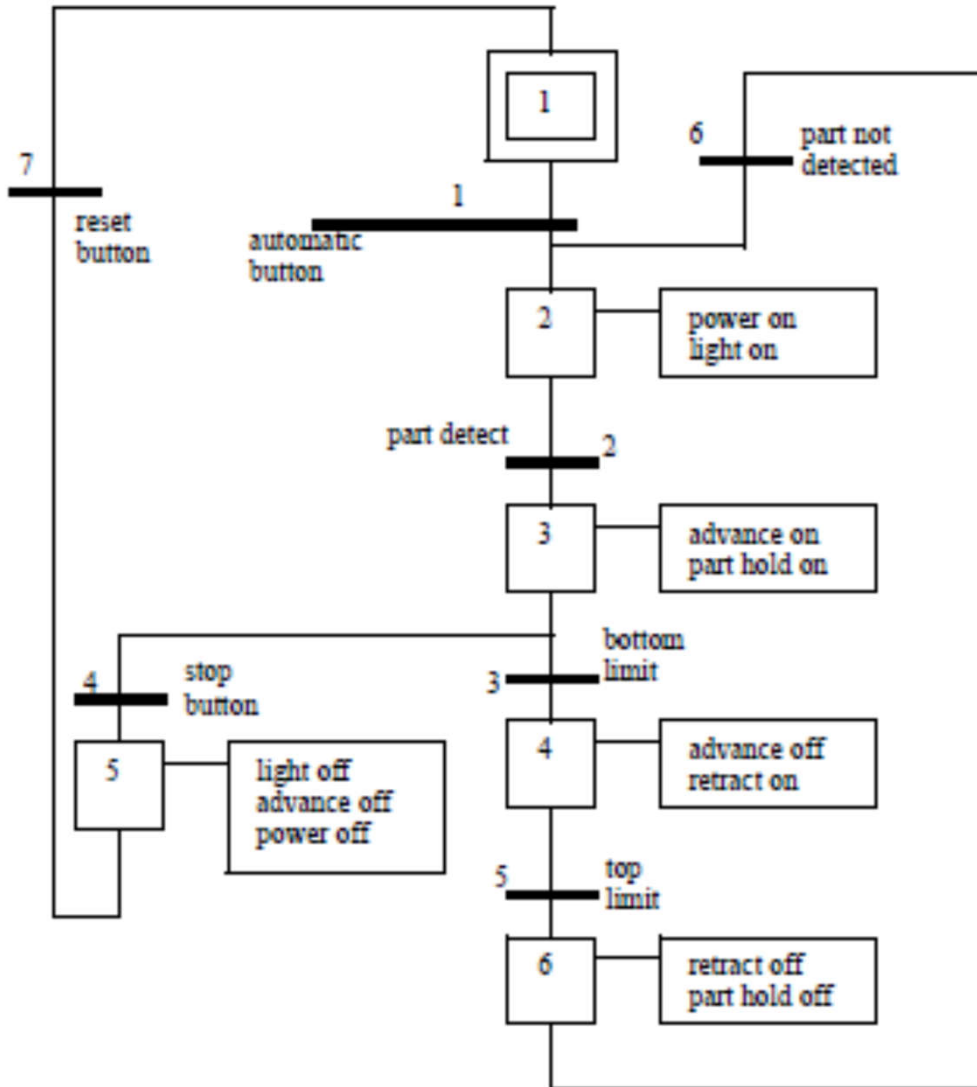
2. Check transitions & activate steps



3. Perform activities for steps

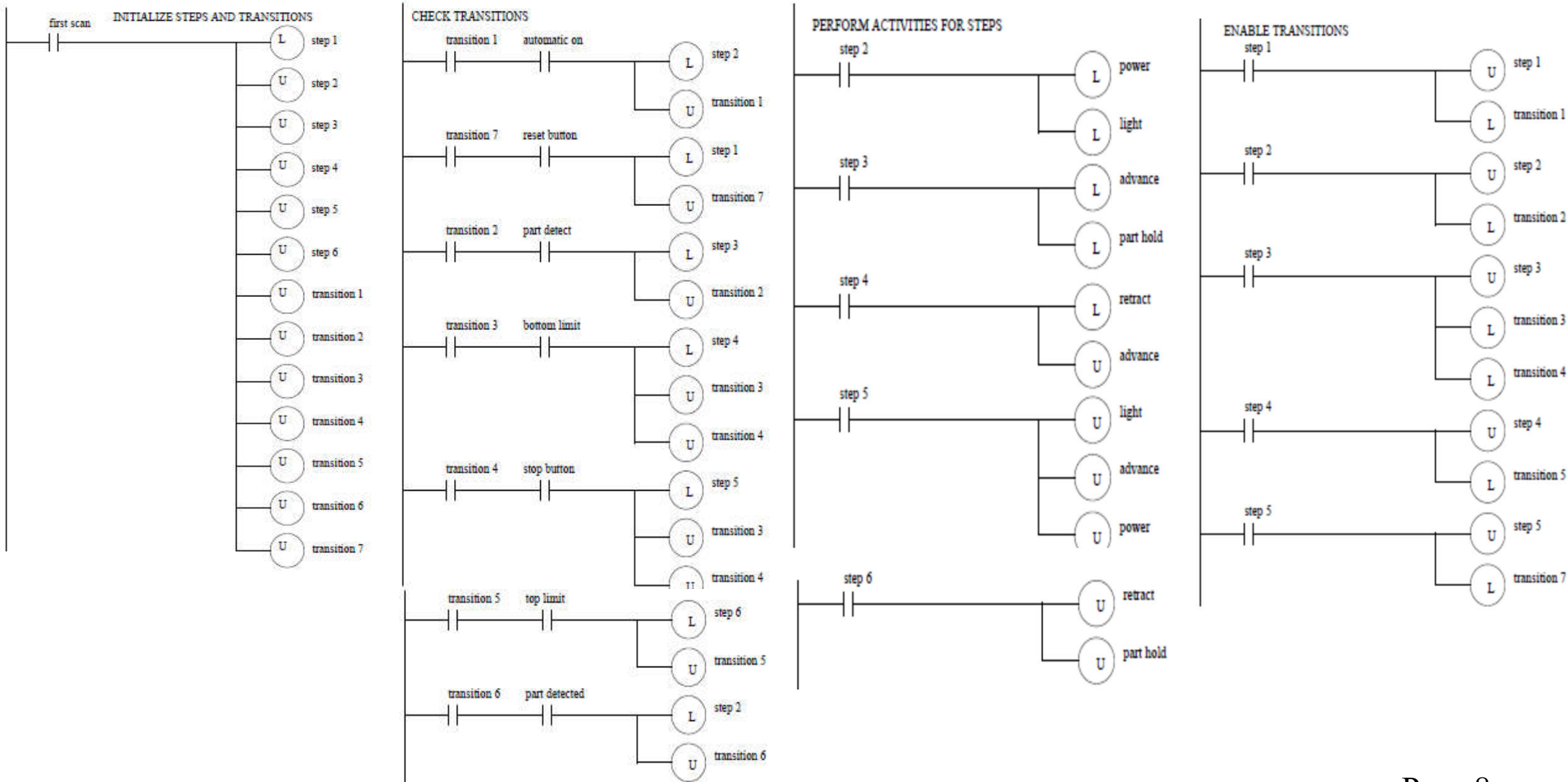


4. Enable transitions



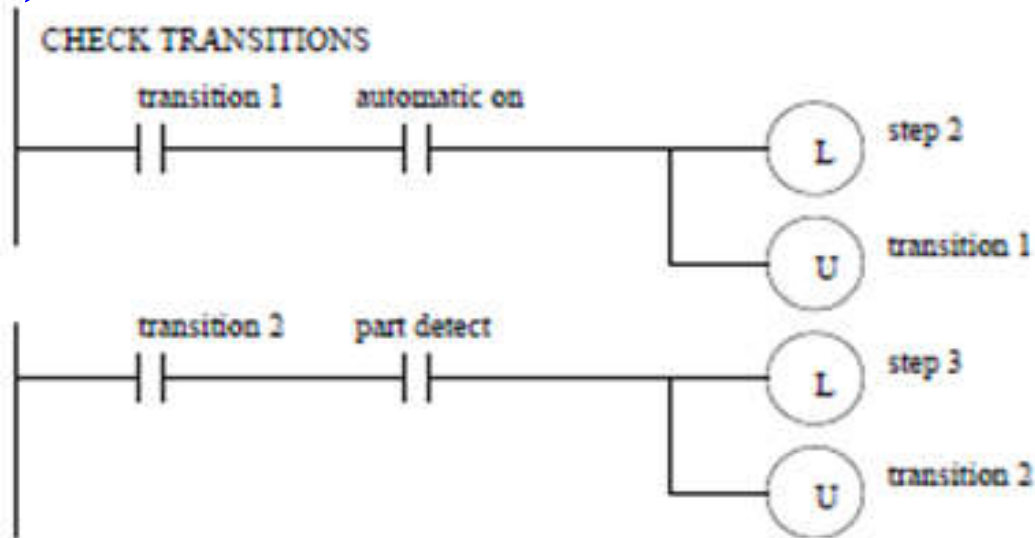
Note: all active steps are made inactive.

The complete program, four sections:

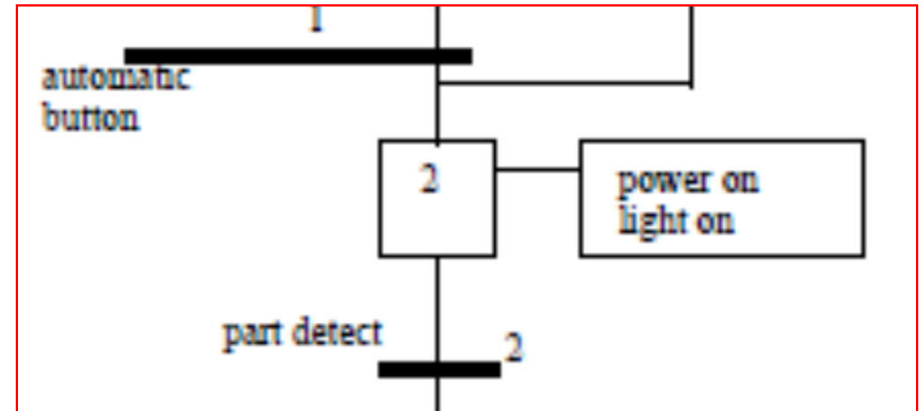
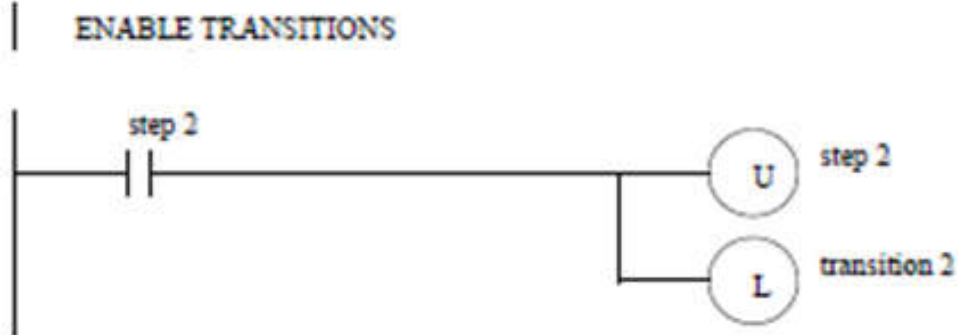


Discussion: not-keeping vs keeping steps active

1)

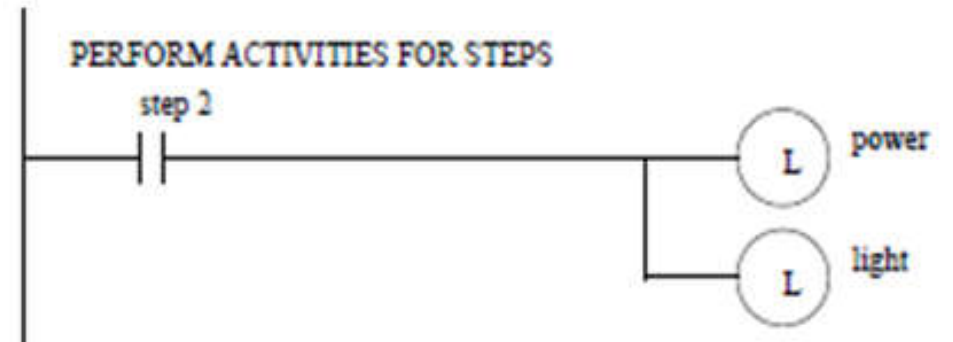


3)



Small part of the Grafset to analyse

2)



*If step2 is **active** in 1 then its work is done in 2 and it is going to be made **inactive (unlatched)** in 3.*

*Note: code parts 1, 2 and 3 run in a single scan cycle; **latched outputs imply they have no spikes.***

*Note2: **Unity Pro** is not like this, step2 gets inactive only after transposing transitions2.*

Homework challenges:

Convert the ladder code shown in the previous slides to a **structured text program**.

Consider simulating the ladder diagram, **ladder instructions one-by-one**, saving all variables:

- Steps (1..6)
- Transitions (1..6)
- Inputs (automatic button, part detect, ...)
- Outputs (power, light, advance, ...)

Confirm that:

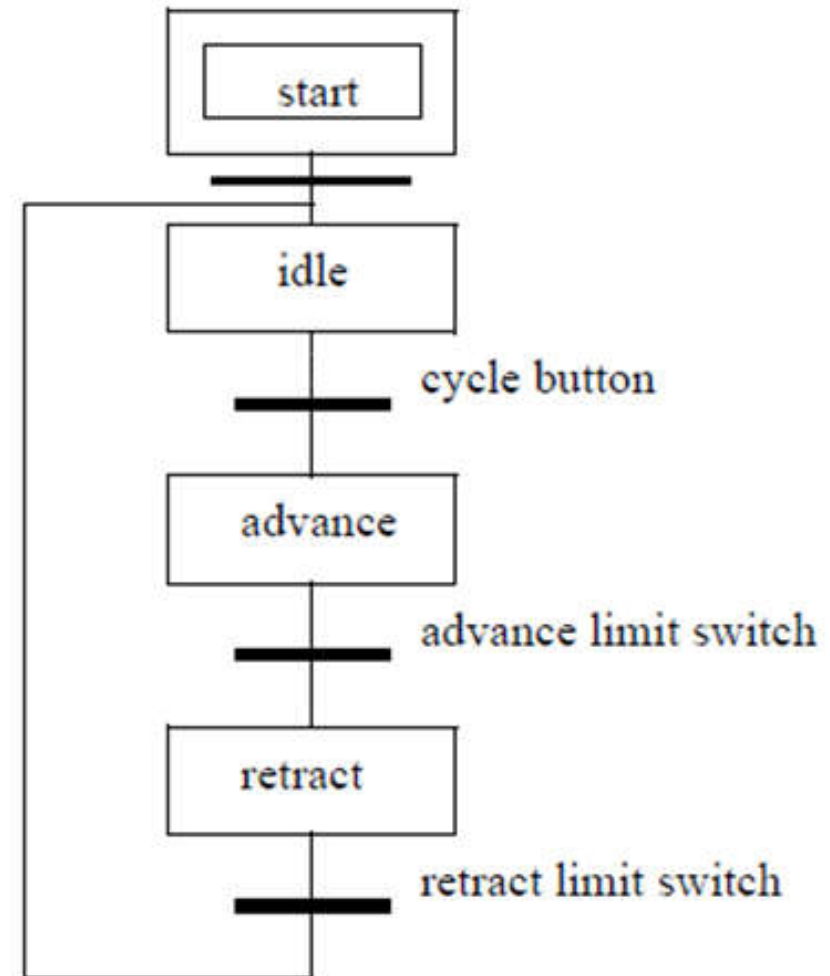
- Step variables are active *at most one scan cycle*
- Outputs are set/unset (latched/unlatched) and therefore *do not need the steps being active all the time*.

GRAFCET Practice Problem 1

Draw **one SFC** for one stamping press that can **advance and retract** when a **cycle button** is pushed, and then stop until the button is pushed again. The press has **limit switches** indicating stop advancing and stop retracting.

Further study: discuss the advantages of using SFC as compared with using Ladder in this problem.

From [Hugh Jack 2008]



GRAFCET Practice Problem 3

Design a garage door controller using an SFC. The behavior of the garage door controller is as follows:

- There is a **single button in the garage and a single button remote control**. When the button is pushed the door will move up or down.
- There are **top/bottom limit switches to stop** the motion of the door.
- If the button is pushed once while moving, the door will **stop**. A second push will start motion again in the **opposite direction**.
- There is a light beam across the bottom of the door. If the beam is cut while the door is closing the door will **stop and reverse**.
- There is a garage **light that will be on for 5 minutes** after the door opens or closes.

