



**Politecnico  
di Torino**

# **FLUID AUTOMATION**

# **TASK 2**

**Automatic conveyor system**

Students:

Muratore Luigi

Akbarov Iskandar

Muhammad Fatir Noshab

s333098

s329650

s331898

Professor:

Luigi Mazza

# CONTENT

- 01** Task requirements
- 02** FluidSIM
- 03** FluidSIM – Pneumatic circuit
- 04** FluidSIM – Programmable logic controller – PLC
- 05** CODESYS
- 06** CODESYS – Ladder code (LD) and Structured text (ST)
- 07** Human-Machine Interface – HMI
- 08** Animations

# 01 Requirements

Develop an automatic conveyor system to process and store steel plates coming from a steel mill.

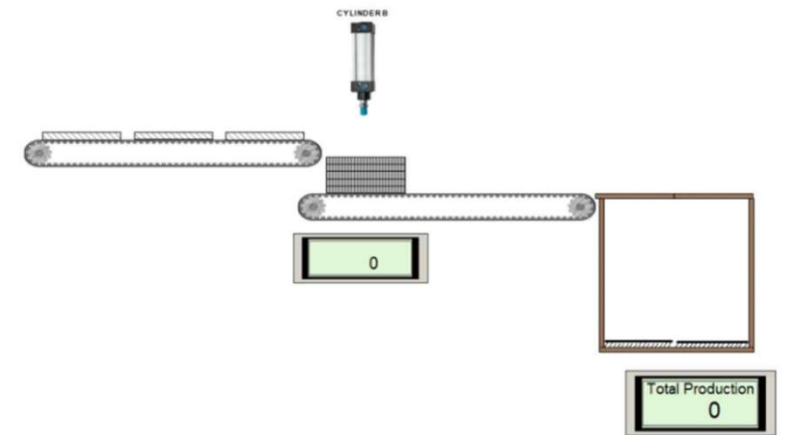
The top conveyor can transport three steel plates simultaneously, which are dropped onto a second conveyor to form a stack.

The operator shall be able to change the number of plates that each stack is made of.

As the stack is complete, the top conveyor stops to allow for the extension of a pneumatic cylinder.

Such cylinder presses on the stack for 3 seconds before retracting.

The stack is then transported to a wooden crate for shipping, while the top conveyor is reactivated to build the next stack.



The incoming plates must be loaded on the first conveyor according to a clock signal generated by a Pulse Timer.

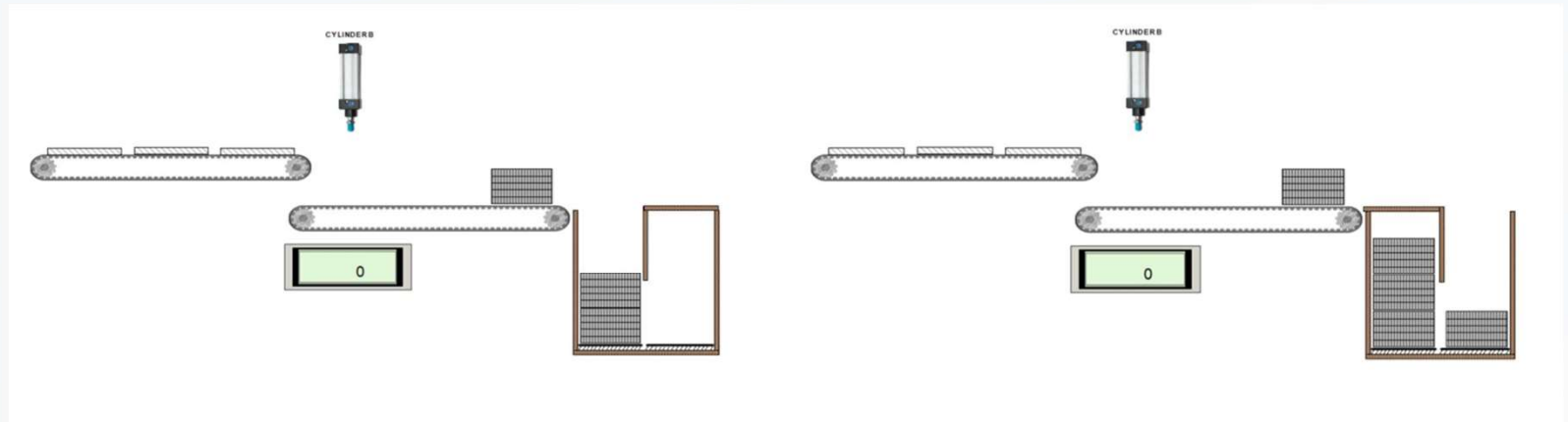
No more than three plates can be loaded at the same time due to weight limits. Develop an automatic conveyor system to process and store steel plates coming from a steel mill.

# 01 Requirements

The wooden crate at the end can store the stacks of plates in **two columns**, with a maximum of **three stacks each**.

Two openings at the top allow for the stacks to be lowered in the correct column: as soon as the first column is full, the second opening is activated to direct the incoming stacks to the second column, while the first opening is closed off.

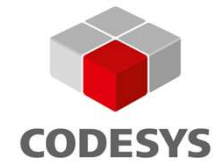
When the whole crate is full, it is ready to be shipped so it must be transported to the designated storage area of the plant.



# Goals and Objectives

Develop the automatic control to implement the task with:

- CODESYS (programming)
- Fluid Sim (plant)



Using:

- Ladder
- ST languages (in proper well-organized FB– ladder and ST, and FUN– in ST).

Moreover, develop a clear and understandable HMI interface CODESYS with:

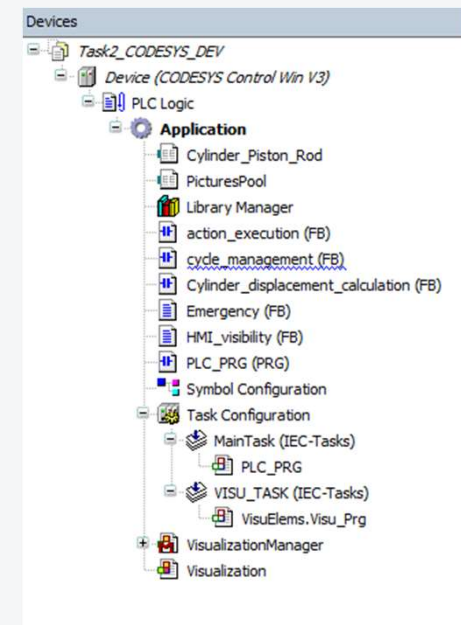
- Start
- Stop
- Emergency
- Alarms
- Conveyors
- Storage station
- Cylinders
- Sensors
- Pilot lights



# Goals and Objectives

In particular, develop a well-organized code that uses POU in LD (ladder) and structured text (ST) with batch method including:

- Main POU (Start, Stop, general commands)
- FBs with I/O communications signals for:
  - Cycle management (LD)
  - Actions execution (LD)
  - Emergency (ST)
  - HMI visibility (ST)
- FUN in ST for cylinders and objects animation



# 02 FluidSIM

*FluidSIM is a simulation software developed by Festo Didactic, designed for the creation and analysis of pneumatics, hydraulics, and electrical circuits.*

*It is widely used in education and training, as well as in industry, to simulate and study fluid and control systems.*



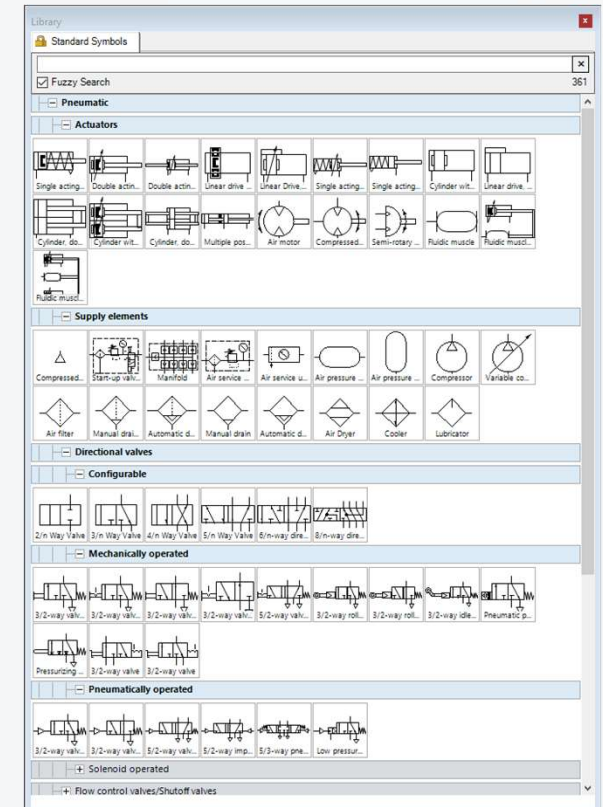
## Key Features:

- **Simulation Environment:** Allows users to design and simulate fluid power and electrical circuits.
- **Interactive Circuit Design:** Provides an intuitive drag-and-drop interface for building complex systems.
- **Real-Time Simulation:** Simulates the dynamic behavior of components in real-time, giving feedback on system performance.
- **Component Library:** Offers an extensive library of pre-built components, including valves, cylinders, sensors, and motors.
- **Detailed Analysis:** Allows users to analyze the behavior of individual components and overall system performance.
- **Reduced Testing Costs:** By simulating systems before physical testing, it minimizes the need for costly prototypes and hardware.
- **Integration with PLCs:** Can be connected to external PLCs for hardware-in-the-loop (HIL) simulations.



## 02 FluidSIM - Library

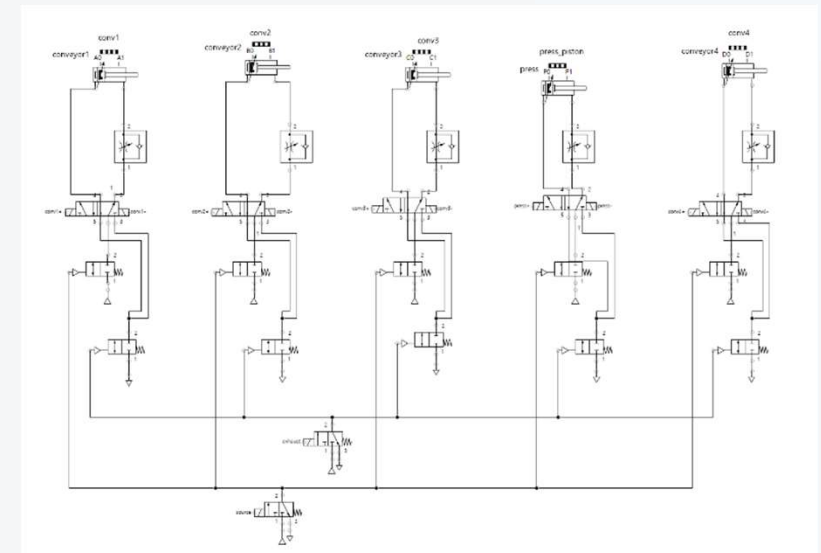
- We utilized the **Library Explorer** in FluidSIM to gather components and all necessary items, making use of different sections, in particular:
  - Pneumatic -> Actuators
  - Pneumatic -> Directional Valves
  - Pneumatic -> Flow control valves
  - Pneumatic -> Sensors
  - Electrical Controls (IEC Standard) -> Power supply
  - Electrical Controls (IEC Standard) -> Output components
  - Electrical Controls (IEC Standard) -> Switches and Contact
  - EasyPort/OPC/DDE





# 03 FluidSIM - Pneumatic

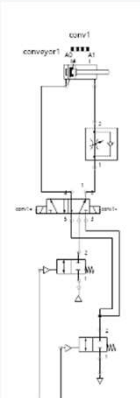
- We used pneumatic pistons to simulate the movements of the boxes, we linked each box to the linear movement of the respective piston:
  - The first piston simulates the first horizontal movement.
  - The second piston simulates the second horizontal movement.
  - The third piston simulates the third horizontal movement.
  - The fourth piston simulates the vertical movement of the press.
  - The fifth one simulate the fourth horizontal movement.
- For the pneumatic circuit we used:
  - 5x - **double acting cylinders**
  - 5x - **5/2 directional valve with 2 stable positions**
  - 10x - **2/2 directional valve with 1 stable position**
  - 2x - **3/2 directional valve with 1 stable position**
  - 10x - **sensors** (sensor ref. Bidirectional)
  - 5x - **restrictors**
- We used 5/2 directional valves electrically switched by **solenoids** on both sides to control each cylinder.
- We used 3/2 and 2/2 directional valves to manage the emergency and stop mode.



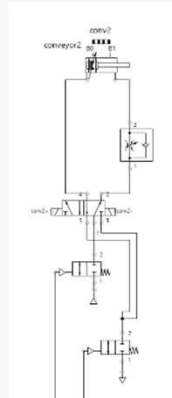
# 03 FluidSIM - Pneumatic

Since we needed to follow a specific path, composed of 5 linear movements, specifically 4 horizontal and 1 vertical, we could link the translation animations of each box to the linear position of the respective piston.

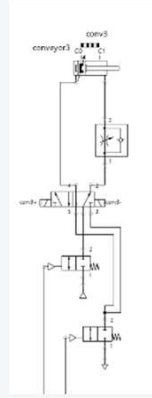
- 1 horizontal



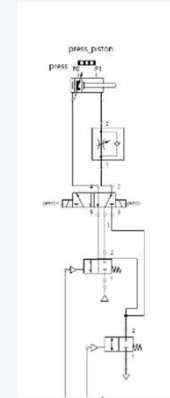
- 2 horizontal



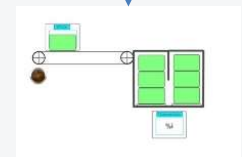
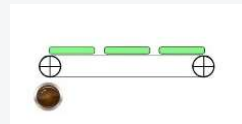
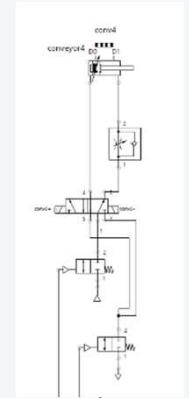
- 3 horizontal



- 4 Vertical

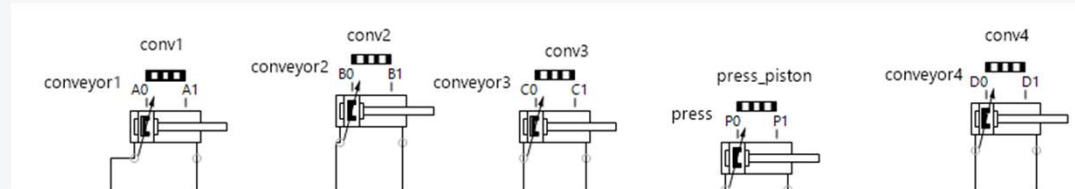


- 5 horizontal



We will handle the animation mixing the visibilities of the four boxes in CODESYS.

# 03 FluidSIM - Sensors



## Limit Switch:

- **A0:** detect the fully instroke piston *conveyor1*
- **A1:** detect the fully outstroke piston *conveyor1*
- **B0:** detect the fully instroke piston *conveyor2*
- **B1:** detect the fully outstroke piston *conveyor2*
- **C0:** detect the fully instroke piston *conveyor3*
- **C1:** detect the fully outstroke piston *conveyor3*
- **P0:** detect the fully instroke piston *press*
- **P1:** detect the fully outstroke piston *press*
- **D0:** detect the fully instroke piston *conveyor4*
- **D1:** detect the fully outstroke piston *conveyor4*

- > it is activated when the piston is in a rest position and fully retracted.
- > it is activated when the valve of the piston is switched and it is fully extended.
- > it is activated when the piston is in a rest position and fully retracted.
- > it is activated when the valve of the piston is switched and it is fully extended.
- > it is activated when the piston is in a rest position and fully retracted.
- > it is activated when the valve of the piston is switched and it is fully extended.
- > it is activated when the piston is in a rest position and fully retracted.
- > it is activated when the valve of the piston is switched and it is fully extended.
- > it is activated when the piston is in a rest position and fully retracted.
- > it is activated when the valve of the piston is switched and it is fully extended.

## Displacement encoders:

- **conv1:** provide motion feedback for linear measurement of the movement of piston *conveyor1*
- **conv2:** provide motion feedback for linear measurement of the movement of piston *conveyor2*
- **conv3:** provide motion feedback for linear measurement of the movement of piston *conveyor3*
- **press\_piston:** provide motion feedback for linear measurement of the movement of piston *press*
- **conv4:** provide motion feedback for linear measurement of the movement of piston *conveyor4*

# 04 FluidSIM - PLC

We used **Programmable logic controller (PLC)** to handle all the information regarding inputs and outputs and program actions related to a specific situation.

To complete the connections in FluidSIM, we used:

- **Electric connections and lines**
- **Make switch**
- **Push-Button**
- **Solenoids**
- **Displacement encoder**
- **FluidSIM Input Port**
- **FluidSIM Output Port**

The ports are structured in terms of **bytes, bits, and port names**:

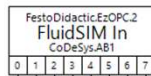
## Digital Inputs (AB)

### Addressing:

- Digital inputs are addressed by bytes and bits.
- Each byte consists of 8 bits.

### Port Naming:

- Starting with "AB" followed by the byte and bit number.
- Example: AB1.3 (Input byte 1, bit 3).



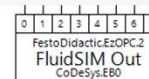
## Digital Outputs (EB)

### Addressing:

- Digital outputs are also addressed by bytes and bits.

### Port Naming:

- Starting with "EB" followed by the byte and bit number.
- Example: EB1.7 (Output byte 1, bit 7).



This structured approach allows for precise control and monitoring of various processes.

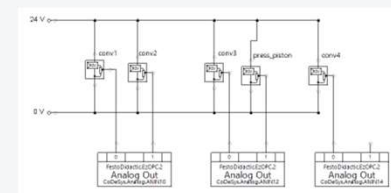
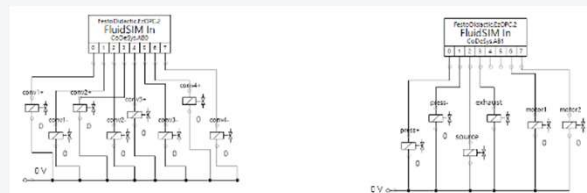
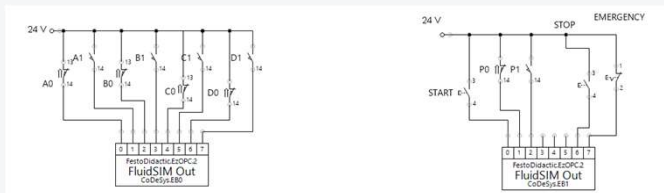
# 04 FluidSIM - PLC

We connected the normally-open make switches to the sensors mounted on the pistons' rod: *A0, A1, B0, B1, C0, C1, P0, P1, D0* and *D1*. So whenever one of the sensors is activated, an electrical signal comes to the respective PLC's port and we can handle a specific task knowing where the signal is.

The solenoids are the ones that take the signal to the valve to allow them to commute and handle the outstroke or the instroke of a cylinder. They allow us to also control the valve at the beginning of the circuit to let the air enter into the circuit or the block it in the emergency condition.

The displacement encoder take us data related to the position of the piston and so allows us to move the boxes in the simulated environment.

Once we have defined all the inputs and all the outputs, we passed these data to CODESYS to be able to manage all the system macro actions, routines and subroutines.



- **FluidSIM OUT:**
  - **Push button and limit switch**
    - A0/A1
    - B0/B1
    - C0/C1
    - D0/D1
    - P0/P1
    - START
    - STOP
    - EMERGENCY
- **FluidSIM IN:**
  - **Solenoids:**
    - conv1+ / conv1-
    - conv2+ / conv2-
    - conv3+ / conv3-
    - conv4+ / conv4-
    - press+ / press-
    - source
    - exhaust
    - Motor 1 / Motor 2
- **FluidSIM Analog OUT:**
  - **Displacement encoder:**
    - conv1
    - conv2
    - conv3
    - press\_piston
    - conv4

# 05 CODESYS

**CODESYS** (Controller Development System) is a powerful, IEC 61131-3 compliant software for industrial automation.

It is widely used in industries such as manufacturing, process automation, energy, and building automation.



**CODESYS**

## Key Features

- **Multi-language Support:**

- Ladder Diagram (LD)
- Structured Text (ST)
- Function Block Diagram (FBD)
- Sequential Function Chart (SFC)
- Instruction List (IL)

- **Cross-Platform Compatibility:**

Supports a wide range of hardware platforms, including PLCs, HMIs, and soft PLCs.

- **Real-Time Control:**

Enables real-time, deterministic control of industrial processes.

- **Integrated Development Environment (IDE):**

Offers tools for coding, simulation, and debugging in one unified environment.

- **Flexibility:**

Adaptable to various hardware and application types.

- **Scalability:**

Suitable for small to large-scale automation projects.

# 06 CODESYS-POUs

In CODESYS, a **POU (Program Organization Unit)** is a core element used to organize and structure the program. It refers to individual programming units that include Programs, Functions, and Function Blocks.

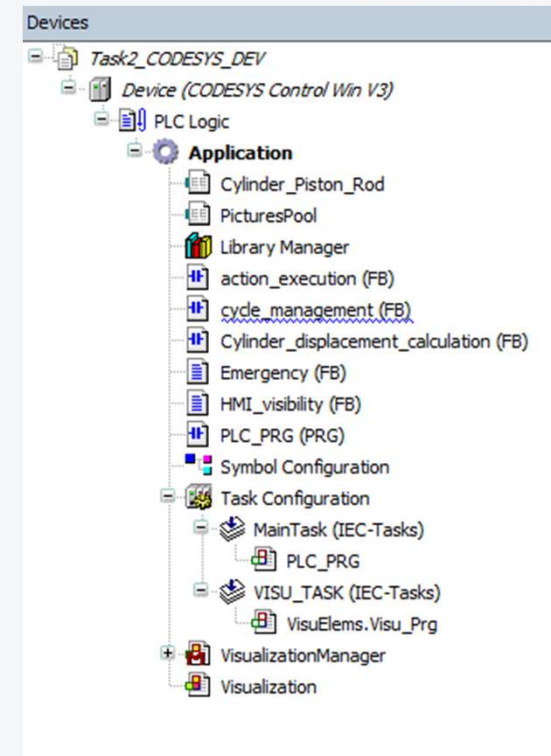
Each POU can be written in any of the IEC 61131-3 languages and serves different purposes:

- **Program:** Main unit where code execution starts; defines the primary control logic.
- **Function:** A reusable block that returns a single output based on given inputs, like a mathematical operation.
- **Function Block:** A reusable unit with internal memory, used for complex operations or control logic, such as managing timers or counters.

POUs help modularize the code, making it easier to develop, test, and maintain complex automation projects.

We created six different POUs based on the requirements of our projects:

- **Main PLC\_PRG**
- *Action execution*
- *Cycle management*
- *Cylinder displacement calculation*
- *Emergency*
- *HMI visibility*





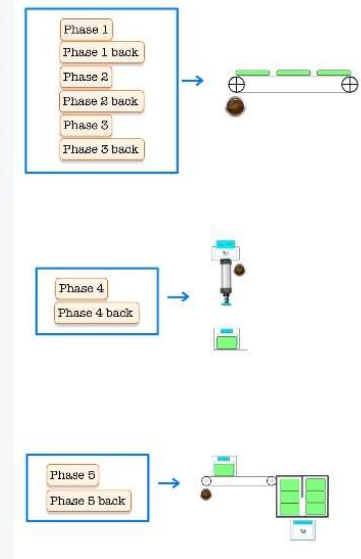
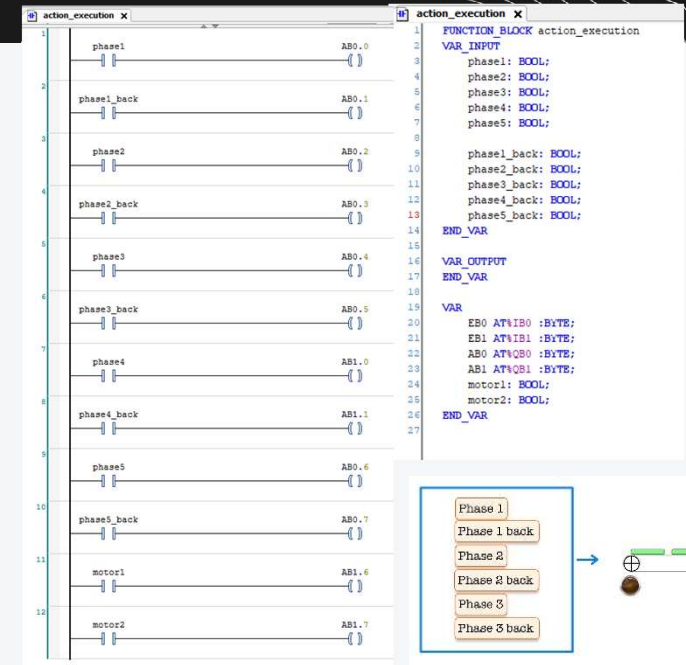
# 06 CODESYS – Action execution (LD)

We started creating the **Action execution POU in Ladder Code**, it is responsible of taking the data from the PLC on FLuidSIM and convert them into actions according to a sequence.

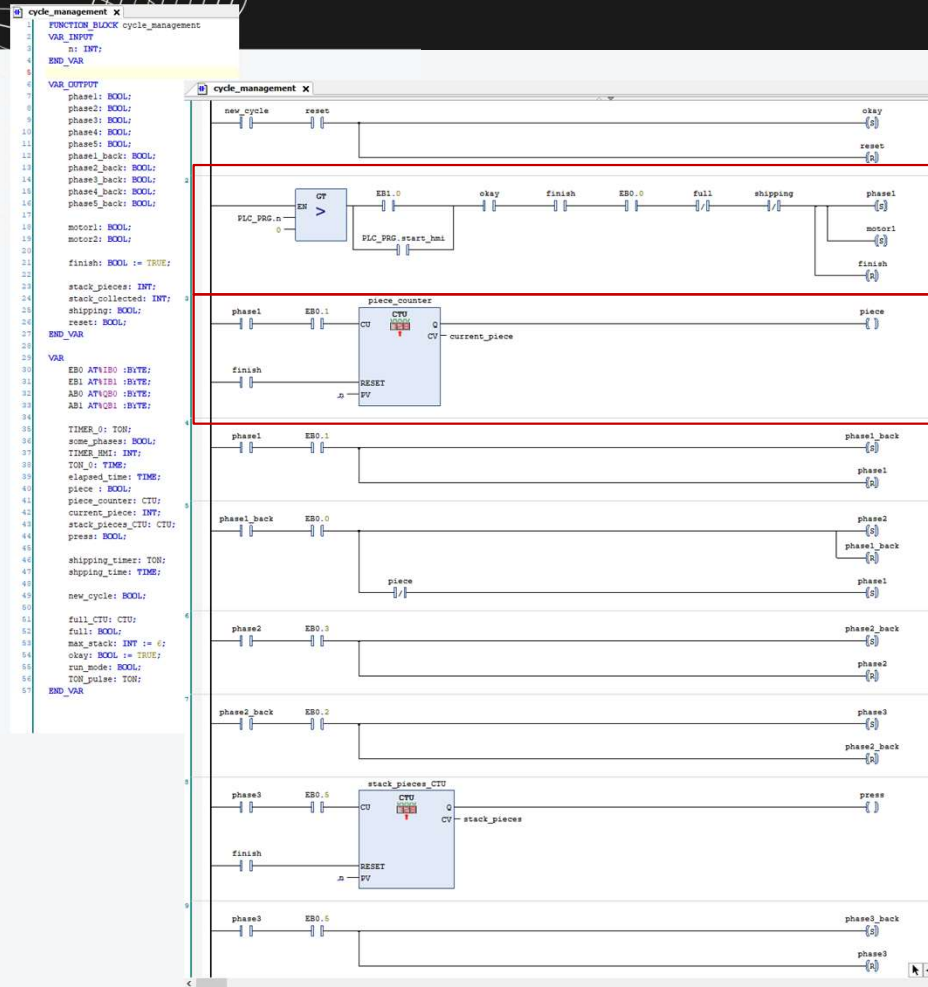
**Ladder Diagram** is a graphical programming language, it is used to develop software for programmable logic controllers (PLCs). It is one of the standard languages specifies for use with PLCs.

In CODESYS we first defined all the input and output variables and then we divided all the operations of our cycles into different phases to handle them in an easier way:

- |                                  |                       |                           |
|----------------------------------|-----------------------|---------------------------|
| • ABO.0 (conv1+) → Phase 1:      | conveyor1 outstroke → | first box goes forward    |
| • ABO.1 (conv1-) → Phase 1 back: | conveyor1 instroke →  | first box goes backward   |
| • ABO.2 (conv2+) → Phase 2:      | conveyor2 outstroke → | second box goes forward   |
| • ABO.3 (conv2-) → Phase 2 back: | conveyor2 instroke →  | second box goes backward  |
| • ABO.4 (conv3+) → Phase 3:      | conveyor3 outstroke → | third box goes forward    |
| • ABO.5 (conv3-) → Phase 3 back: | conveyor3 instroke →  | third box goes backward   |
| • AB1.0 (press+) → Phase 4:      | press outstroke →     | press goes forward        |
| • AB1.1 (press-) → Phase 4 back: | press instroke →      | press goes backward       |
| • ABO.6 (conv4+) → Phase 5:      | conveyor4 outstroke → | pressed box goes forward  |
| • ABO.7 (conv4-) → Phase 5 back: | conveyor4 instroke →  | pressed box goes backward |



# 06 CODESYS – Cycle management (LD)



Then we created the *Cycle management POU* in *Ladder Code*, it:

- is responsible for managing the performing of the loop cycles.
- defines the position and the timing of each phases
- performs the pulse signal for the loading of the plates on the first conveyor
- performs all the sequence phases
- manages all the counters and timers

We used all the information related to the limit switch mounted on the pistons on FLUIDSIM to better supervise all the operations. Here we handled all the variable:

- new cycle
- full
- shipping
- finish
- okay
- reset

The **second network** starts effectively the loop.

**If :**

- $n > 0$
- start is pressed (either from the HMI or from the PLC)
- all the variables are checked

**Then:**

- phase1 is performed
- motor1 is turned on
- reset of the variable finish

The **third network** is used to count the pieces loaded on the first conveyor.

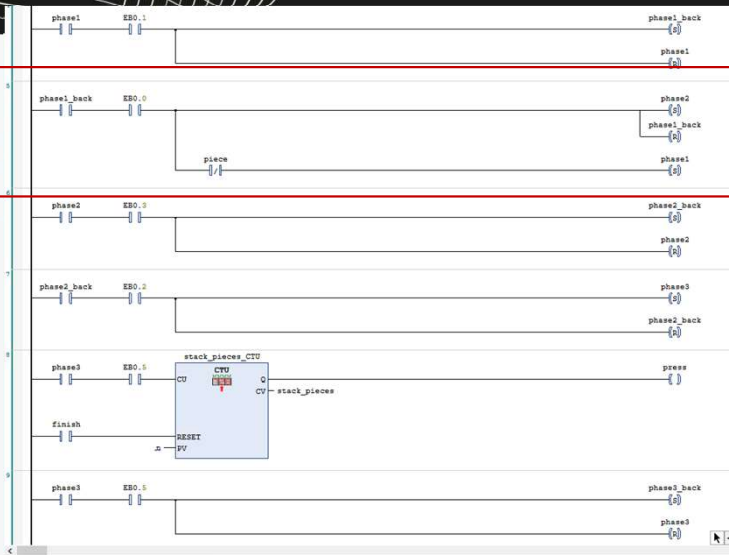
**If :**

- $n =$  the quantity of pieces loaded on the conveyor

**Then:**

- stop the loading

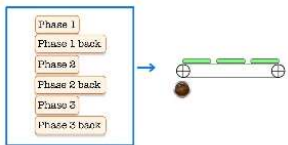
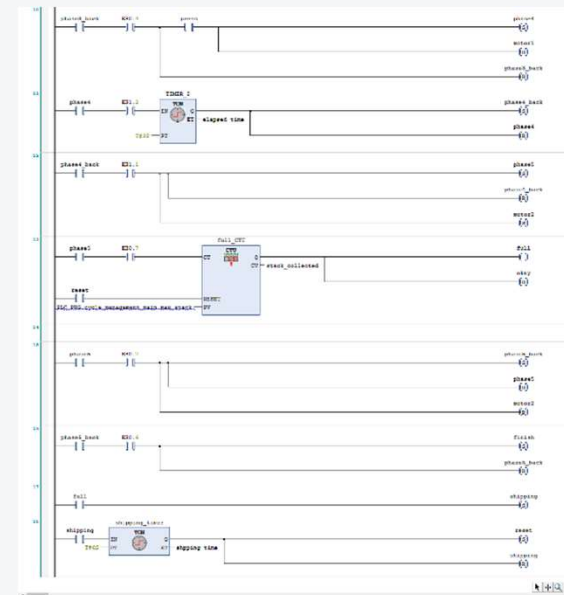
# 06 CODESYS – Cycle management (LD)



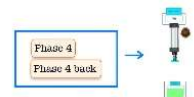
The most important condition is the **network 5**:

- the phase1-back set the execution of phase2
- in parallel it sets also the phase1 again if the variable piece is not set.

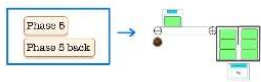
Finish variable → set at every ending loop of stack-cycle and reset at every start.



From phase1 to phase3-back = boxes on the first conveyor.

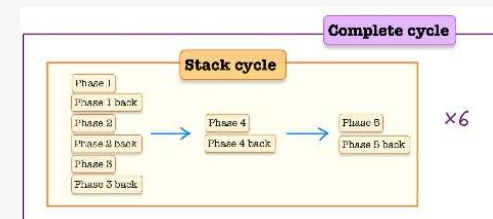


From phase4 to phase4-back = press → 3 seconds → timer



From phase5 to phase5-back = stack from the press station to the stacking area

Stacking area → max 6 stacks → counter  
If counter = 6 → full → shipping → delivering → new-cycle



# 06 CODESYS – Cylinder displacement (LD)

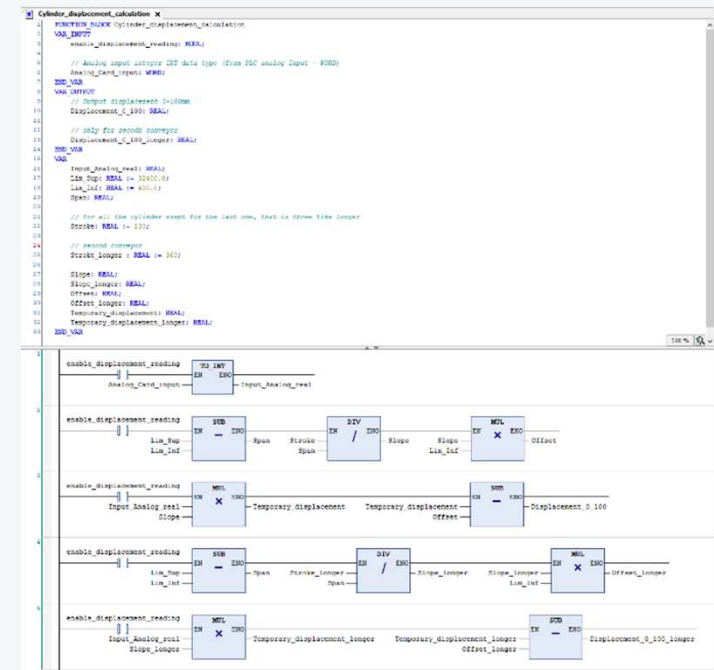
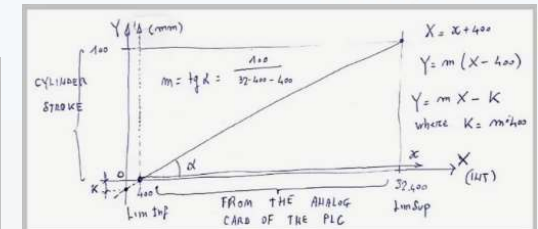
The **cylinder displacement calculation** is a POU written in **Ladder Code**, necessary to convert the linear position of the pistons in a linear function and then in a scale from 0 to 100 to better handle the animations.

The displacement encoder reads the value of the position of the piston. They send these data from the analog card of the PLC to CODESYS.

By means of some **simple formulas**, we are able to have a linear scale of the position. To then perform animations, as translation, in a better way.

We firstly convert the data from the analog card of the PLC to an integer number, then we use **math operators blocks** to perform calculations, as divider or subtractor and so on, to have the displacement as output.

$$m = \frac{\text{STROKE}}{\text{LIM\_SUP} - \text{LIM\_INF}} = \text{Slope}$$
$$K = m \cdot \text{LIM\_INF} = \text{OFFSET}$$
$$Y = m \cdot X - K$$



# 06 CODESYS – Emergency (ST)

```
Emergency X
1 FUNCTION_BLOCK Emergency
2
3 VAR_INPUT
4   start_hmi: BOOL;
5 END_VAR
6
7 VAR_OUTPUT
8   stop_state: BOOL;
9   emergency_state: BOOL;
10  normal_mode: BOOL;
11 END_VAR
12
13 VAR
14   EBO AT%IB0: BYTE;
15   EBI AT%IB1: BYTE;
16   ABO AT%QB0: BYTE;
17   ABI AT%QB1: BYTE;
18
19   emergency: BOOL;
20   stop: BOOL;
21 END_VAR
22
23 IF EBI.0 OR start_hmi THEN
24   emergency_state := 0;
25   stop_state := 0;
26 END_IF
27
28 IF emergency OR NOT EBI.7 THEN
29   emergency_state := 1;
30 END_IF
31
32 IF emergency_state THEN
33   ABI.2 := 0;
34   ABI.3 := 0;
35 END_IF
36
37 IF stop OR EBI.6 THEN
38   stop_state := 1;
39 END_IF
40
41 IF stop_state THEN
42   ABI.2 := 0;
43   ABI.3 := 1;
44 END_IF
```

The *Emergency* POU, written in *Structure Text (ST)*, is necessary to prevent dangerous situation and handle failures and unexpected results.

We defined three states:

- normal-mode
- stop-state
- emergency-state.

At the beginning of the cycle	During the cycles
<p><b>If:</b></p> <ul style="list-style-type: none"><li>• start button is clicked (either from the HMI or from the PLC)</li></ul> <p><b>Then:</b></p> <ul style="list-style-type: none"><li>• stop-state and emergency-state -&gt; set to FALSE or 0.</li></ul>	<p><b>If:</b></p> <ul style="list-style-type: none"><li>• emergency button is clicked (either from the HMI or from the PLC)</li></ul> <p><b>Then:</b></p> <ul style="list-style-type: none"><li>• emergency-state -&gt; set to TRUE or 1.</li></ul> <p><b>If:</b></p> <ul style="list-style-type: none"><li>• stop button is clicked (either from the HMI or from the PLC)</li></ul> <p><b>Then:</b></p> <ul style="list-style-type: none"><li>• stop-state -&gt; set to TRUE or 1.</li></ul>

Here we handled these two states directly on the pneumatic circuit, placing two 3/2 valve with 1 stable position:

- *Emergency-state* -> both of the valve are de-energized -> stop the system instantaneously.
- *Stop-state* -> blocks only the source of air -> let the exhaust open to unload the air -> end in the next phase.

We did it sending commands directly to the PLC port where the solenoids of the valves are connected.



# 06 CODESYS – HMI visibility (ST)

Here we handled all the possible **animations**, in terms of *translation and visibilities*.

We coded this POU in **Structured Text**.

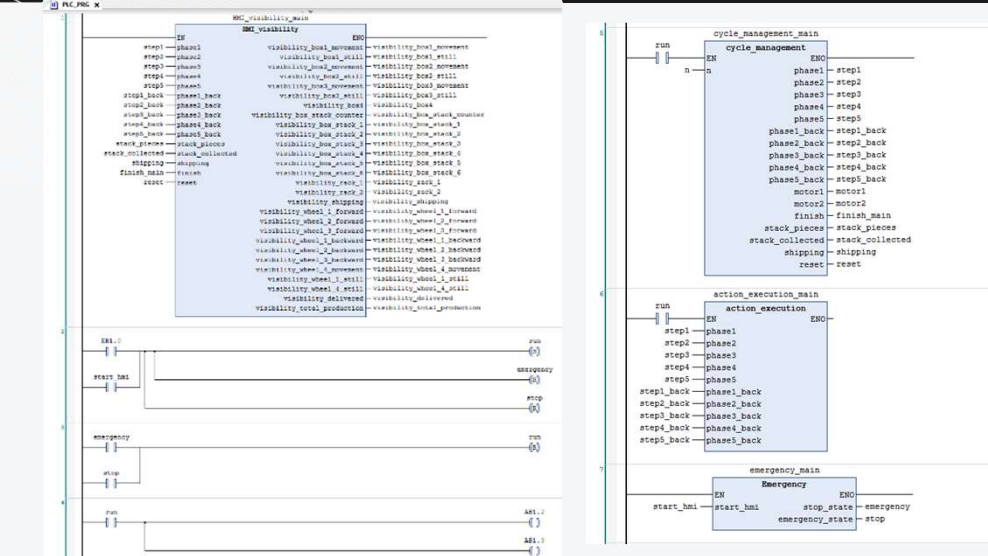
As for the other POUs we firstly set the input variables that were needed to activate a particular condition and then, we coded them in if-cycle.

We dealt with *visibility* of:

- three plates on the first conveyor
- the big box that contains the plates coming from the first conveyor
- the shipping
- the delivering station.

```
1 FUNCTION_BLOCK HMI_visibility
2
3 VAR_INPUT
4     phase1: BOOL;
5     phase2: BOOL;
6     phase3: BOOL;
7     phase4: BOOL;
8     phase5: BOOL;
9
10    phase1_back: BOOL;
11    phase2_back: BOOL;
12    phase3_back: BOOL;
13    phase4_back: BOOL;
14    phase5_back: BOOL;
15
16    stack_pieces: INT;
17    stack_collected: INT;
18    shipping: BOOL;
19    finish: BOOL;
20    reset: BOOL;
21
22 END_VAR
23
24 VAR_OUTPUT
25     visibility_box1_movement: BOOL := TRUE;
26     visibility_box1_still: BOOL := TRUE;
27
28     visibility_box2_movement: BOOL := TRUE;
29     visibility_box2_still: BOOL := TRUE;
30
31     visibility_box3_movement: BOOL := TRUE;
32     visibility_box3_still: BOOL := TRUE;
33
34     visibility_box4: BOOL := TRUE;
35
36     visibility_box_stack_counter: BOOL := TRUE;
37
38     visibility_box_stack_1: BOOL := TRUE;
39     visibility_box_stack_2: BOOL := TRUE;
40     visibility_box_stack_3: BOOL := TRUE;
41     visibility_box_stack_4: BOOL := TRUE;
42     visibility_box_stack_5: BOOL := TRUE;
43     visibility_box_stack_6: BOOL := TRUE;
44
45     visibility_rack_1: BOOL := FALSE;
46     visibility_rack_2: BOOL := TRUE;
47
48     visibility_shipping: BOOL := TRUE;
49
50     visibility_wheel_1_forward: BOOL := TRUE;
51     visibility_wheel_2_forward: BOOL := TRUE;
52     visibility_wheel_3_forward: BOOL := TRUE;
53     visibility_wheel_1_backward: BOOL := TRUE;
54     visibility_wheel_2_backward: BOOL := TRUE;
55     visibility_wheel_3_backward: BOOL := TRUE;
56
57     visibility_wheel_1_movement: BOOL := TRUE;
58
59     visibility_wheel_1_still: BOOL := FALSE;
60     visibility_wheel_2_still: BOOL := FALSE;
61
62     visibility_delivered: BOOL := TRUE;
63
64     visibility_total_production: BOOL := FALSE;
65
66 VAR
67 END_VAR
68
69 // Visibility of the wheels -> pulley conveyors
70 IF phase1 THEN
71     visibility_wheel_1_forward := FALSE;
72     visibility_wheel_2_forward := TRUE;
73     visibility_wheel_3_forward := TRUE;
74     visibility_wheel_1_still := TRUE;
75     visibility_wheel_2_backward := TRUE;
76     visibility_wheel_3_backward := TRUE;
77 END_IF
78
79 IF phase2 THEN
80     visibility_wheel_1_forward := TRUE;
81     visibility_wheel_2_forward := TRUE;
82     visibility_wheel_3_forward := FALSE;
83     visibility_wheel_1_still := TRUE;
84     visibility_wheel_2_backward := TRUE;
85     visibility_wheel_3_backward := TRUE;
86 END_IF
87
88 IF phase3 THEN
89     visibility_wheel_1_forward := TRUE;
90     visibility_wheel_2_forward := TRUE;
91     visibility_wheel_3_forward := FALSE;
92     visibility_wheel_1_still := TRUE;
93     visibility_wheel_2_backward := TRUE;
94     visibility_wheel_3_backward := TRUE;
95 END_IF
96
97 IF phase4 THEN
98     visibility_wheel_1_forward := TRUE;
99     visibility_wheel_2_forward := TRUE;
100    visibility_wheel_3_forward := FALSE;
101    visibility_wheel_1_still := TRUE;
102    visibility_wheel_2_backward := TRUE;
103    visibility_wheel_3_backward := TRUE;
104 END_IF
105
106 IF phase5 THEN
107     visibility_wheel_1_forward := TRUE;
108     visibility_wheel_2_forward := TRUE;
109     visibility_wheel_3_forward := FALSE;
110     visibility_wheel_1_still := TRUE;
111     visibility_wheel_2_backward := TRUE;
112     visibility_wheel_3_backward := TRUE;
113 END_IF
114
115 // Visibility of boxes of conveyor 1
116 IF phase1 THEN
117     visibility_box1_movement := FALSE;
118     visibility_box1_still := TRUE;
119 END_IF
120
121 IF phase2 THEN
122     visibility_box2_movement := FALSE;
123     visibility_box2_still := TRUE;
124 END_IF
125
126 IF phase3 THEN
127     visibility_box3_movement := FALSE;
128     visibility_box3_still := TRUE;
129 END_IF
130
131 IF phase4 THEN
132     visibility_box4 := FALSE;
133 END_IF
134
135 // Visibility of the stack on the rack
136 IF stack_collected=1 THEN
137     visibility_box_stack_1 := FALSE;
138 END_IF
139
140 IF stack_collected=2 THEN
141     visibility_box_stack_2 := FALSE;
142 END_IF
143
144 IF stack_collected=3 THEN
145     visibility_box_stack_3 := FALSE;
146 END_IF
147
148 IF stack_collected=4 THEN
149     visibility_box_stack_4 := FALSE;
150 END_IF
151
152 IF stack_collected=5 THEN
153     visibility_box_stack_5 := FALSE;
154 END_IF
155
156 IF stack_collected=6 THEN
157     visibility_box_stack_6 := FALSE;
158 END_IF
159
160 // Visibility box on conveyor2 and box with the stack
161 IF stack_pieces > 0 THEN
162     visibility_box_stack_counter := FALSE;
163 END_IF
164
165 IF phase1_back THEN
166     visibility_box1_movement := TRUE;
167     visibility_box1_still := FALSE;
168 END_IF
169
170 IF phase2_back THEN
171     visibility_box2_movement := TRUE;
172     visibility_box2_still := FALSE;
173 END_IF
174
175 IF phase3_back THEN
176     visibility_box3_movement := TRUE;
177     visibility_box3_still := FALSE;
178 END_IF
179
180 IF phase4_back THEN
181     visibility_box4 := TRUE;
182 END_IF
183
184 IF phase5_back THEN
185     visibility_box_stack_counter := TRUE;
186 END_IF
187
188 // Visibility of the rack 1 or 2
189 IF NOT shipping AND NOT reset THEN
190     visibility_rack_1 := TRUE;
191     visibility_rack_2 := FALSE;
192 END_IF
193
194 IF shipping THEN
195     visibility_rack_1 := FALSE;
196     visibility_rack_2 := TRUE;
197 END_IF
198
199 // Visibility of DELIVERED
200 IF reset THEN
201     visibility_delivered := FALSE;
202 END_IF
203
204 IF NOT reset THEN
205     visibility_delivered := TRUE;
206 END_IF
207
208 // Visibility of shipping section
209 IF shipping THEN
210     visibility_shipping := FALSE;
211     visibility_rack_1 := TRUE;
212     visibility_rack_2 := TRUE;
213     visibility_box_stack_1 := TRUE;
214     visibility_box_stack_2 := TRUE;
215     visibility_box_stack_3 := TRUE;
216     visibility_box_stack_4 := TRUE;
217     visibility_box_stack_5 := TRUE;
218     visibility_box_stack_6 := TRUE;
219     visibility_total_production := TRUE;
220 END_IF
221
222 IF NOT shipping THEN
223     visibility_shipping := TRUE;
224     visibility_rack_1 := FALSE;
225     visibility_rack_2 := FALSE;
226     visibility_box_stack_1 := FALSE;
227     visibility_box_stack_2 := FALSE;
228     visibility_box_stack_3 := FALSE;
229     visibility_box_stack_4 := FALSE;
230     visibility_box_stack_5 := FALSE;
231     visibility_box_stack_6 := FALSE;
232     visibility_total_production := FALSE;
233 END_IF
234
235 // Visibility of the rack 1 or 2
236 IF NOT shipping AND NOT reset THEN
237     visibility_rack_1 := TRUE;
238     visibility_rack_2 := FALSE;
239 END_IF
240
241 IF shipping THEN
242     visibility_rack_1 := FALSE;
243     visibility_rack_2 := TRUE;
244 END_IF
245
246 // Visibility of DELIVERED
247 IF reset THEN
248     visibility_delivered := FALSE;
249 END_IF
250
251 IF NOT reset THEN
252     visibility_delivered := TRUE;
253 END_IF
254
255 // Visibility of shipping section
256 IF shipping THEN
257     visibility_shipping := FALSE;
258     visibility_rack_1 := TRUE;
259     visibility_rack_2 := TRUE;
260     visibility_box_stack_1 := TRUE;
261     visibility_box_stack_2 := TRUE;
262     visibility_box_stack_3 := TRUE;
263     visibility_box_stack_4 := TRUE;
264     visibility_box_stack_5 := TRUE;
265     visibility_box_stack_6 := TRUE;
266     visibility_total_production := TRUE;
267 END_IF
268
269 IF NOT shipping THEN
270     visibility_shipping := TRUE;
271     visibility_rack_1 := FALSE;
272     visibility_rack_2 := FALSE;
273     visibility_box_stack_1 := FALSE;
274     visibility_box_stack_2 := FALSE;
275     visibility_box_stack_3 := FALSE;
276     visibility_box_stack_4 := FALSE;
277     visibility_box_stack_5 := FALSE;
278     visibility_box_stack_6 := FALSE;
279     visibility_total_production := FALSE;
280 END_IF
281
282 // Visibility of the rack 1 or 2
283 IF NOT shipping AND NOT reset THEN
284     visibility_rack_1 := TRUE;
285     visibility_rack_2 := FALSE;
286 END_IF
287
288 IF shipping THEN
289     visibility_rack_1 := FALSE;
290     visibility_rack_2 := TRUE;
291 END_IF
292
293 // Visibility of DELIVERED
294 IF reset THEN
295     visibility_delivered := FALSE;
296 END_IF
297
298 IF NOT reset THEN
299     visibility_delivered := TRUE;
300 END_IF
301
302 // Visibility of shipping section
303 IF shipping THEN
304     visibility_shipping := FALSE;
305     visibility_rack_1 := TRUE;
306     visibility_rack_2 := TRUE;
307     visibility_box_stack_1 := TRUE;
308     visibility_box_stack_2 := TRUE;
309     visibility_box_stack_3 := TRUE;
310     visibility_box_stack_4 := TRUE;
311     visibility_box_stack_5 := TRUE;
312     visibility_box_stack_6 := TRUE;
313     visibility_total_production := TRUE;
314 END_IF
315
316 IF NOT shipping THEN
317     visibility_shipping := TRUE;
318     visibility_rack_1 := FALSE;
319     visibility_rack_2 := FALSE;
320     visibility_box_stack_1 := FALSE;
321     visibility_box_stack_2 := FALSE;
322     visibility_box_stack_3 := FALSE;
323     visibility_box_stack_4 := FALSE;
324     visibility_box_stack_5 := FALSE;
325     visibility_box_stack_6 := FALSE;
326     visibility_total_production := FALSE;
327 END_IF
328
329 // Visibility of the rack 1 or 2
330 IF NOT shipping AND NOT reset THEN
331     visibility_rack_1 := TRUE;
332     visibility_rack_2 := FALSE;
333 END_IF
334
335 IF shipping THEN
336     visibility_rack_1 := FALSE;
337     visibility_rack_2 := TRUE;
338 END_IF
339
340 // Visibility of DELIVERED
341 IF reset THEN
342     visibility_delivered := FALSE;
343 END_IF
344
345 IF NOT reset THEN
346     visibility_delivered := TRUE;
347 END_IF
348
349 // Visibility of shipping section
350 IF shipping THEN
351     visibility_shipping := FALSE;
352     visibility_rack_1 := TRUE;
353     visibility_rack_2 := TRUE;
354     visibility_box_stack_1 := TRUE;
355     visibility_box_stack_2 := TRUE;
356     visibility_box_stack_3 := TRUE;
357     visibility_box_stack_4 := TRUE;
358     visibility_box_stack_5 := TRUE;
359     visibility_box_stack_6 := TRUE;
360     visibility_total_production := TRUE;
361 END_IF
362
363 IF NOT shipping THEN
364     visibility_shipping := TRUE;
365     visibility_rack_1 := FALSE;
366     visibility_rack_2 := FALSE;
367     visibility_box_stack_1 := FALSE;
368     visibility_box_stack_2 := FALSE;
369     visibility_box_stack_3 := FALSE;
370     visibility_box_stack_4 := FALSE;
371     visibility_box_stack_5 := FALSE;
372     visibility_box_stack_6 := FALSE;
373     visibility_total_production := FALSE;
374 END_IF
375
376 // Visibility of the rack 1 or 2
377 IF NOT shipping AND NOT reset THEN
378     visibility_rack_1 := TRUE;
379     visibility_rack_2 := FALSE;
380 END_IF
381
382 IF shipping THEN
383     visibility_rack_1 := FALSE;
384     visibility_rack_2 := TRUE;
385 END_IF
386
387 // Visibility of DELIVERED
388 IF reset THEN
389     visibility_delivered := FALSE;
390 END_IF
391
392 IF NOT reset THEN
393     visibility_delivered := TRUE;
394 END_IF
395
396 // Visibility of shipping section
397 IF shipping THEN
398     visibility_shipping := FALSE;
399     visibility_rack_1 := TRUE;
400     visibility_rack_2 := TRUE;
401     visibility_box_stack_1 := TRUE;
402     visibility_box_stack_2 := TRUE;
403     visibility_box_stack_3 := TRUE;
404     visibility_box_stack_4 := TRUE;
405     visibility_box_stack_5 := TRUE;
406     visibility_box_stack_6 := TRUE;
407     visibility_total_production := TRUE;
408 END_IF
409
410 IF NOT shipping THEN
411     visibility_shipping := TRUE;
412     visibility_rack_1 := FALSE;
413     visibility_rack_2 := FALSE;
414     visibility_box_stack_1 := FALSE;
415     visibility_box_stack_2 := FALSE;
416     visibility_box_stack_3 := FALSE;
417     visibility_box_stack_4 := FALSE;
418     visibility_box_stack_5 := FALSE;
419     visibility_box_stack_6 := FALSE;
420     visibility_total_production := FALSE;
421 END_IF
422
423 // Visibility of the rack 1 or 2
424 IF NOT shipping AND NOT reset THEN
425     visibility_rack_1 := TRUE;
426     visibility_rack_2 := FALSE;
427 END_IF
428
429 IF shipping THEN
430     visibility_rack_1 := FALSE;
431     visibility_rack_2 := TRUE;
432 END_IF
433
434 // Visibility of DELIVERED
435 IF reset THEN
436     visibility_delivered := FALSE;
437 END_IF
438
439 IF NOT reset THEN
440     visibility_delivered := TRUE;
441 END_IF
442
443 // Visibility of shipping section
444 IF shipping THEN
445     visibility_shipping := FALSE;
446     visibility_rack_1 := TRUE;
447     visibility_rack_2 := TRUE;
448     visibility_box_stack_1 := TRUE;
449     visibility_box_stack_2 := TRUE;
450     visibility_box_stack_3 := TRUE;
451     visibility_box_stack_4 := TRUE;
452     visibility_box_stack_5 := TRUE;
453     visibility_box_stack_6 := TRUE;
454     visibility_total_production := TRUE;
455 END_IF
456
457 IF NOT shipping THEN
458     visibility_shipping := TRUE;
459     visibility_rack_1 := FALSE;
460     visibility_rack_2 := FALSE;
461     visibility_box_stack_1 := FALSE;
462     visibility_box_stack_2 := FALSE;
463     visibility_box_stack_3 := FALSE;
464     visibility_box_stack_4 := FALSE;
465     visibility_box_stack_5 := FALSE;
466     visibility_box_stack_6 := FALSE;
467     visibility_total_production := FALSE;
468 END_IF
469
470 // Visibility of the rack 1 or 2
471 IF NOT shipping AND NOT reset THEN
472     visibility_rack_1 := TRUE;
473     visibility_rack_2 := FALSE;
474 END_IF
475
476 IF shipping THEN
477     visibility_rack_1 := FALSE;
478     visibility_rack_2 := TRUE;
479 END_IF
480
481 // Visibility of DELIVERED
482 IF reset THEN
483     visibility_delivered := FALSE;
484 END_IF
485
486 IF NOT reset THEN
487     visibility_delivered := TRUE;
488 END_IF
489
490 // Visibility of shipping section
491 IF shipping THEN
492     visibility_shipping := FALSE;
493     visibility_rack_1 := TRUE;
494     visibility_rack_2 := TRUE;
495     visibility_box_stack_1 := TRUE;
496     visibility_box_stack_2 := TRUE;
497     visibility_box_stack_3 := TRUE;
498     visibility_box_stack_4 := TRUE;
499     visibility_box_stack_5 := TRUE;
500     visibility_box_stack_6 := TRUE;
501     visibility_total_production := TRUE;
502 END_IF
503
504 IF NOT shipping THEN
505     visibility_shipping := TRUE;
506     visibility_rack_1 := FALSE;
507     visibility_rack_2 := FALSE;
508     visibility_box_stack_1 := FALSE;
509     visibility_box_stack_2 := FALSE;
510     visibility_box_stack_3 := FALSE;
511     visibility_box_stack_4 := FALSE;
512     visibility_box_stack_5 := FALSE;
513     visibility_box_stack_6 := FALSE;
514     visibility_total_production := FALSE;
515 END_IF
516
517 // Visibility of the rack 1 or 2
518 IF NOT shipping AND NOT reset THEN
519     visibility_rack_1 := TRUE;
520     visibility_rack_2 := FALSE;
521 END_IF
522
523 IF shipping THEN
524     visibility_rack_1 := FALSE;
525     visibility_rack_2 := TRUE;
526 END_IF
527
528 // Visibility of DELIVERED
529 IF reset THEN
530     visibility_delivered := FALSE;
531 END_IF
532
533 IF NOT reset THEN
534     visibility_delivered := TRUE;
535 END_IF
536
537 // Visibility of shipping section
538 IF shipping THEN
539     visibility_shipping := FALSE;
540     visibility_rack_1 := TRUE;
541     visibility_rack_2 := TRUE;
542     visibility_box_stack_1 := TRUE;
543     visibility_box_stack_2 := TRUE;
544     visibility_box_stack_3 := TRUE;
545     visibility_box_stack_4 := TRUE;
546     visibility_box_stack_5 := TRUE;
547     visibility_box_stack_6 := TRUE;
548     visibility_total_production := TRUE;
549 END_IF
550
551 IF NOT shipping THEN
552     visibility_shipping := TRUE;
553     visibility_rack_1 := FALSE;
554     visibility_rack_2 := FALSE;
555     visibility_box_stack_1 := FALSE;
556     visibility_box_stack_2 := FALSE;
557     visibility_box_stack_3 := FALSE;
558     visibility_box_stack_4 := FALSE;
559     visibility_box_stack_5 := FALSE;
560     visibility_box_stack_6 := FALSE;
561     visibility_total_production := FALSE;
562 END_IF
563
564 // Visibility of the rack 1 or 2
565 IF NOT shipping AND NOT reset THEN
566     visibility_rack_1 := TRUE;
567     visibility_rack_2 := FALSE;
568 END_IF
569
570 IF shipping THEN
571     visibility_rack_1 := FALSE;
572     visibility_rack_2 := TRUE;
573 END_IF
574
575 // Visibility of DELIVERED
576 IF reset THEN
577     visibility_delivered := FALSE;
578 END_IF
579
580 IF NOT reset THEN
581     visibility_delivered := TRUE;
582 END_IF
583
584 // Visibility of shipping section
585 IF shipping THEN
586     visibility_shipping := FALSE;
587     visibility_rack_1 := TRUE;
588     visibility_rack_2 := TRUE;
589     visibility_box_stack_1 := TRUE;
590     visibility_box_stack_2 := TRUE;
591     visibility_box_stack_3 := TRUE;
592     visibility_box_stack_4 := TRUE;
593     visibility_box_stack_5 := TRUE;
594     visibility_box_stack_6 := TRUE;
595     visibility_total_production := TRUE;
596 END_IF
597
598 IF NOT shipping THEN
599     visibility_shipping := TRUE;
600     visibility_rack_1 := FALSE;
601     visibility_rack_2 := FALSE;
602     visibility_box_stack_1 := FALSE;
603     visibility_box_stack_2 := FALSE;
604     visibility_box_stack_3 := FALSE;
605     visibility_box_stack_4 := FALSE;
606     visibility_box_stack_5 := FALSE;
607     visibility_box_stack_6 := FALSE;
608     visibility_total_production := FALSE;
609 END_IF
610
611 // Visibility of the rack 1 or 2
612 IF NOT shipping AND NOT reset THEN
613     visibility_rack_1 := TRUE;
614     visibility_rack_2 := FALSE;
615 END_IF
616
617 IF shipping THEN
618     visibility_rack_1 := FALSE;
619     visibility_rack_2 := TRUE;
620 END_IF
621
622 // Visibility of DELIVERED
623 IF reset THEN
624     visibility_delivered := FALSE;
625 END_IF
626
627 IF NOT reset THEN
628     visibility_delivered := TRUE;
629 END_IF
630
631 // Visibility of shipping section
632 IF shipping THEN
633     visibility_shipping := FALSE;
634     visibility_rack_1 := TRUE;
635     visibility_rack_2 := TRUE;
636     visibility_box_stack_1 := TRUE;
637     visibility_box_stack_2 := TRUE;
638     visibility_box_stack_3 := TRUE;
639     visibility_box_stack_4 := TRUE;
640     visibility_box_stack_5 := TRUE;
641     visibility_box_stack_6 := TRUE;
642     visibility_total_production := TRUE;
643 END_IF
644
645 IF NOT shipping THEN
646     visibility_shipping := TRUE;
647     visibility_rack_1 := FALSE;
648     visibility_rack_2 := FALSE;
649     visibility_box_stack_1 := FALSE;
650     visibility_box_stack_2 := FALSE;
651     visibility_box_stack_3 := FALSE;
652     visibility_box_stack_4 := FALSE;
653     visibility_box_stack_5 := FALSE;
654     visibility_box_stack_6 := FALSE;
655     visibility_total_production := FALSE;
656 END_IF
657
658 // Visibility of the rack 1 or 2
659 IF NOT shipping AND NOT reset THEN
660     visibility_rack_1 := TRUE;
661     visibility_rack_2 := FALSE;
662 END_IF
663
664 IF shipping THEN
665     visibility_rack_1 := FALSE;
666     visibility_rack_2 := TRUE;
667 END_IF
668
669 // Visibility of DELIVERED
670 IF reset THEN
671     visibility_delivered := FALSE;
672 END_IF
673
674 IF NOT reset THEN
675     visibility_delivered := TRUE;
676 END_IF
677
678 // Visibility of shipping section
679 IF shipping THEN
680     visibility_shipping := FALSE;
681     visibility_rack_1 := TRUE;
682     visibility_rack_2 := TRUE;
683     visibility_box_stack_1 := TRUE;
684     visibility_box_stack_2 := TRUE;
685     visibility_box_stack_3 := TRUE;
686     visibility_box_stack_4 := TRUE;
687     visibility_box_stack_5 := TRUE;
688     visibility_box_stack_6 := TRUE;
689     visibility_total_production := TRUE;
690 END_IF
691
692 IF NOT shipping THEN
693     visibility_shipping := TRUE;
694     visibility_rack_1 := FALSE;
695     visibility_rack_2 := FALSE;
696     visibility_box_stack_1 := FALSE;
697     visibility_box_stack_2 := FALSE;
698     visibility_box_stack_3 := FALSE;
699     visibility_box_stack_4 := FALSE;
700     visibility_box_stack_5 := FALSE;
701     visibility_box_stack_6 := FALSE;
702     visibility_total_production := FALSE;
703 END_IF
704
705 // Visibility of the rack 1 or 2
706 IF NOT shipping AND NOT reset THEN
707     visibility_rack_1 := TRUE;
708     visibility_rack_2 := FALSE;
709 END_IF
710
711 IF shipping THEN
712     visibility_rack_1 := FALSE;
713     visibility_rack_2 := TRUE;
714 END_IF
715
716 // Visibility of DELIVERED
717 IF reset THEN
718     visibility_delivered := FALSE;
719 END_IF
720
721 IF NOT reset THEN
722     visibility_delivered := TRUE;
723 END_IF
724
725 // Visibility of shipping section
726 IF shipping THEN
727     visibility_shipping := FALSE;
728     visibility_rack_1 := TRUE;
729     visibility_rack_2 := TRUE;
730     visibility_box_stack_1 := TRUE;
731     visibility_box_stack_2 := TRUE;
732     visibility_box_stack_3 := TRUE;
733     visibility_box_stack_4 := TRUE;
734     visibility_box_stack_5 := TRUE;
735     visibility_box_stack_6 := TRUE;
736     visibility_total_production := TRUE;
737 END_IF
738
739 IF NOT shipping THEN
740     visibility_shipping := TRUE;
741     visibility_rack_1 := FALSE;
742     visibility_rack_2 := FALSE;
743     visibility_box_stack_1 := FALSE;
744     visibility_box_stack_2 := FALSE;
745     visibility_box_stack_3 := FALSE;
746     visibility_box_stack_4 := FALSE;
747     visibility_box_stack_5 := FALSE;
748     visibility_box_stack_6 := FALSE;
749     visibility_total_production := FALSE;
750 END_IF
751
752 // Visibility of the rack 1 or 2
753 IF NOT shipping AND NOT reset THEN
754     visibility_rack_1 := TRUE;
755     visibility_rack_2 := FALSE;
756 END_IF
757
758 IF shipping THEN
759     visibility_rack_1 := FALSE;
760     visibility_rack_2 := TRUE;
761 END_IF
762
763 // Visibility of DELIVERED
764 IF reset THEN
765     visibility_delivered := FALSE;
766 END_IF
767
768 IF NOT reset THEN
769     visibility_delivered := TRUE;
770 END_IF
771
772 // Visibility of shipping section
773 IF shipping THEN
774     visibility_shipping := FALSE;
775     visibility_rack_1 := TRUE;
776     visibility_rack_2 := TRUE;
777     visibility_box_stack_1 := TRUE;
778     visibility_box_stack_2 := TRUE;
779     visibility_box_stack_3 := TRUE;
780     visibility_box_stack_4 := TRUE;
781     visibility_box_stack_5 := TRUE;
782     visibility_box_stack_6 := TRUE;
783     visibility_total_production := TRUE;
784 END_IF
785
786 IF NOT shipping THEN
787     visibility_shipping := TRUE;
788     visibility_rack_1 := FALSE;
789     visibility_rack_2 := FALSE;
790     visibility_box_stack_1 := FALSE;
791     visibility_box_stack_2 := FALSE;
792     visibility_box_stack_3 := FALSE;
793     visibility_box_stack_4 := FALSE;
794     visibility_box_stack_5 := FALSE;
795     visibility_box_stack_6 := FALSE;
796     visibility_total_production := FALSE;
797 END_IF
798
799 // Visibility of the rack 1 or 2
800 IF NOT shipping AND NOT reset THEN
801     visibility_rack_1 := TRUE;
802     visibility_rack_2 := FALSE;
803 END_IF
804
805 IF shipping THEN
806     visibility_rack_1 := FALSE;
807     visibility_rack_2 := TRUE;
808 END_IF
809
810 // Visibility of DELIVERED
811 IF reset THEN
812     visibility_delivered := FALSE;
813 END_IF
814
815 IF NOT reset THEN
816     visibility_delivered := TRUE;
817 END_IF
818
819 // Visibility of shipping section
820 IF shipping THEN
821     visibility_shipping := FALSE;
822     visibility_rack_1 := TRUE;
823     visibility_rack_2 := TRUE;
824     visibility_box_stack_1 := TRUE;
825     visibility_box_stack_2 := TRUE;
826     visibility_box_stack_3 := TRUE;
827     visibility_box_stack_4 := TRUE;
828     visibility_box_stack_5 := TRUE;
829     visibility_box_stack_6 := TRUE;
830     visibility_total_production := TRUE;
831 END_IF
832
833 IF NOT shipping THEN
834     visibility_shipping := TRUE;
835     visibility_rack_1 := FALSE;
836     visibility_rack_2 := FALSE;
837     visibility_box_stack_1 := FALSE;
838     visibility_box_stack_2 := FALSE;
839     visibility_box_stack_3 := FALSE;
840     visibility_box_stack_4 := FALSE;
841     visibility_box_stack_5 := FALSE;
842     visibility_box_stack_6 := FALSE;
843     visibility_total_production := FALSE;
844 END_IF
845
846 // Visibility of the rack 1 or 2
847 IF NOT shipping AND NOT reset THEN
848     visibility_rack_1 := TRUE;
849     visibility_rack_2 := FALSE;
850 END_IF
851
852 IF shipping THEN
853     visibility_rack_1 := FALSE;
854     visibility_rack_2 := TRUE;
855 END_IF
856
857 // Visibility of DELIVERED
858 IF reset THEN
859     visibility_delivered := FALSE;
860 END_IF
861
862 IF NOT reset THEN
863     visibility_delivered := TRUE;
864 END_IF
865
866 // Visibility of shipping section
867 IF shipping THEN
868     visibility_shipping := FALSE;
869     visibility_rack_1 := TRUE;
870     visibility_rack_2 := TRUE;
871     visibility_box_stack_1 := TRUE;
872     visibility_box_stack_2 := TRUE;
873     visibility_box_stack_3 := TRUE;
874     visibility_box_stack_4 := TRUE;
875     visibility_box_stack_5 := TRUE;
876     visibility_box_stack_6 := TRUE;
877     visibility_total_production := TRUE;
878 END_IF
879
880 IF NOT shipping THEN
881     visibility_shipping := TRUE;
882     visibility_rack_1 := FALSE;
883     visibility_rack_2 := FALSE;
884     visibility_box_stack_1 := FALSE;
885     visibility_box_stack_2 := FALSE;
886     visibility_box_stack_3 := FALSE;
887     visibility_box_stack_4 := FALSE;
888     visibility_box_stack_5 := FALSE;
889     visibility_box_stack_6 := FALSE;
890     visibility_total_production := FALSE;
891 END_IF
892
893 // Visibility of the rack 1 or 2
894 IF NOT shipping AND NOT reset THEN
895     visibility_rack_1 := TRUE;
896     visibility_rack_2 := FALSE;
897 END_IF
898
899 IF shipping THEN
900     visibility_rack_1 := FALSE;
901     visibility_rack_2 := TRUE;
902 END_IF
903
904 // Visibility of DELIVERED
905 IF reset THEN
906     visibility_delivered := FALSE;
907 END_IF
908
909 IF NOT reset THEN
910     visibility_delivered := TRUE;
911 END_IF
912
913 // Visibility of shipping section
914 IF shipping THEN
915     visibility_shipping := FALSE;
916     visibility_rack_1 := TRUE;
917     visibility_rack_2 := TRUE;
918     visibility_box_stack_1 := TRUE;
919     visibility_box_stack_2 := TRUE;
920     visibility_box_stack_3 := TRUE;
921     visibility_box_stack_4 := TRUE;
922     visibility_box_stack_5 := TRUE;
923     visibility_box_stack_6 := TRUE;
924     visibility_total_production := TRUE;
925 END_IF
926
927 IF NOT shipping THEN
928     visibility_shipping := TRUE;
929     visibility_rack_1 := FALSE;
930     visibility_rack_2 := FALSE;
931     visibility_box_stack_1 := FALSE;
932     visibility_box_stack_2 := FALSE;
933     visibility_box_stack_3 := FALSE;
934     visibility_box_stack_4 := FALSE;
935     visibility_box_stack_5 := FALSE;
936     visibility_box_stack_6 := FALSE;
937     visibility_total_production := FALSE;
938 END_IF
939
940 // Visibility of the rack 1 or 2
941 IF NOT shipping AND NOT reset THEN
942     visibility_rack_1 := TRUE;
943     visibility_rack_2 := FALSE;
944 END_IF
945
946 IF shipping THEN
947     visibility_rack_1 := FALSE;
948     visibility_rack_2 := TRUE;
949 END_IF
950
951 // Visibility of DELIVERED
952 IF reset THEN
953     visibility_delivered := FALSE;
954 END_IF
955
956 IF NOT reset THEN
957     visibility_delivered := TRUE;
958 END_IF
959
960 // Visibility of shipping section
961 IF shipping THEN
962     visibility_shipping := FALSE;
963     visibility_rack_1 := TRUE;
964     visibility_rack_2 := TRUE;
965     visibility_box_stack_1 := TRUE;
966     visibility_box_stack_2 := TRUE;
967     visibility_box_stack_3 := TRUE;
968     visibility_box_stack_4 := TRUE;
969     visibility_box_stack_5 := TRUE;
970     visibility_box_stack_6 := TRUE;
971     visibility_total_production := TRUE;
972 END_IF
973
974 IF NOT shipping THEN
975     visibility_shipping := TRUE;
976     visibility_rack_1 := FALSE;
977     visibility_rack_2 := FALSE;
978     visibility_box_stack_1 := FALSE;
979     visibility_box_stack_2 := FALSE;
980     visibility_box_stack_3 := FALSE;
981     visibility_box_stack_4 := FALSE;
982     visibility_box_stack_5 := FALSE;
983     visibility_box_stack_6 := FALSE;
984     visibility_total_production := FALSE;
985 END_IF
986
987 // Visibility of the rack 1 or 2
988 IF NOT shipping AND NOT reset THEN
989     visibility_rack_1 := TRUE;
990     visibility_rack_2 := FALSE;
991 END_IF
992
993 IF shipping THEN
994     visibility_rack_1 := FALSE;
995     visibility_rack_2 := TRUE;
996 END_IF
997
998 // Visibility of DELIVERED
999 IF reset THEN
1000     visibility_delivered := FALSE;
1001 END_IF
1002
1003 IF NOT reset THEN
1004     visibility_delivered := TRUE;
1005 END_IF
1006
1007 // Visibility of shipping section
1008 IF shipping THEN
1009     visibility_shipping := FALSE;
1010     visibility_rack_1 := TRUE;
1011     visibility_rack_2 := TRUE;
1012     visibility_box_stack_1 := TRUE;
1013     visibility_box_stack_2 := TRUE;
1014     visibility_box_stack_3 := TRUE;
1015     visibility_box_stack_4 := TRUE;
1016     visibility_box_stack_5 := TRUE;
1017     visibility_box_stack_6 := TRUE;
1018     visibility_total_production := TRUE;
1019 END_IF
1020
1021 IF NOT shipping THEN
1022     visibility_shipping := TRUE;
1023     visibility_rack_1 := FALSE;
1024     visibility_rack_2 := FALSE;
1025     visibility_box_stack_1 := FALSE;
1026     visibility_box_stack_2 := FALSE;
1027     visibility_box_stack_3 := FALSE;
1028     visibility_box_stack_4 := FALSE;
1029     visibility_box_stack_5 := FALSE;
1030     visibility_box_stack_6 := FALSE;
1031     visibility_total_production := FALSE;
1032 END_IF
1033
1034 // Visibility of the rack 1 or 2
1035 IF NOT shipping AND NOT reset THEN
1036     visibility_rack_1 := TRUE;
1037     visibility_rack_2 := FALSE;
1038 END_IF
1039
1040 IF shipping THEN
1041     visibility_rack_1 := FALSE;
1042     visibility_rack_2 := TRUE;
1043 END_IF
1044
1045 // Visibility of DELIVERED
1046 IF reset THEN
1047     visibility_delivered := FALSE;
1048 END_IF
1049
1050 IF NOT reset THEN
1051     visibility_delivered := TRUE;
1052 END_IF
1053
1054 // Visibility of shipping section
1055 IF shipping THEN
1056     visibility_shipping := FALSE;
1057     visibility_rack_1 := TRUE;
1058     visibility_rack_2 := TRUE;
1059     visibility_box_stack_1 := TRUE;
1060     visibility_box_stack_2 := TRUE;
1061     visibility_box_stack_3 := TRUE;
1062     visibility_box_stack_4 := TRUE;
1063     visibility_box_stack_5 := TRUE;
1064     visibility_box_stack_6 := TRUE;
1065     visibility_total_production := TRUE;
1066 END_IF
1067
1068 IF NOT shipping THEN
1069     visibility_shipping := TRUE;
1070     visibility_rack_1 := FALSE;
1071     visibility_rack_2 := FALSE;
1072     visibility_box_stack_1 := FALSE;
1073     visibility_box_stack_2 := FALSE;
1074     visibility_box_stack_3 := FALSE;
1075     visibility_box_stack_4 := FALSE;
1076     visibility_box_stack_5 := FALSE;
1077     visibility_box_stack_6 := FALSE;
1078     visibility_total_production := FALSE;
1079 END_IF
1080
1081 // Visibility of the rack 1 or 2
1082 IF NOT shipping AND NOT reset THEN
1083     visibility_rack_1 := TRUE;
1084     visibility_rack_2 := FALSE;
1085 END_IF
1086
1087 IF shipping THEN
1088     visibility_rack_1 := FALSE;
1089     visibility_rack_2 := TRUE;
1090 END_IF
1091
1092 // Visibility of DELIVERED
1093 IF reset THEN
1094     visibility_delivered := FALSE;
1095 END_IF
1096
1097 IF NOT reset THEN
1098     visibility_delivered := TRUE;
1099 END_IF
1100
1101 // Visibility of shipping section
1102 IF shipping THEN
1103     visibility_shipping := FALSE;
1104     visibility_rack_1 := TRUE;
1105     visibility_rack_2 := TRUE;
1106     visibility_box_stack_1 := TRUE;
1107     visibility_box_stack_2 := TRUE;
1108     visibility_box_stack_3 := TRUE;
1109     visibility_box_stack_4 := TRUE;
1110     visibility_box_stack_5 := TRUE;
1111     visibility_box_stack_6 := TRUE;
1112     visibility_total_production := TRUE;
1113 END_IF
1114
1115 IF NOT shipping THEN
1116     visibility_shipping := TRUE;
1117     visibility_rack_1 := FALSE;
1118     visibility_rack_2 := FALSE;
1119     visibility_box_stack_1 := FALSE;
1120     visibility_box_stack_2 := FALSE;
1121     visibility_box_stack_3 := FALSE;
1122     visibility_box_stack_4 := FALSE;
1123     visibility_box_stack_5 := FALSE;
1124     visibility_box_stack_6 := FALSE;
1125     visibility_total_production := FALSE;
1126 END_IF
1127
1128 // Visibility of the rack 1 or 2
1129 IF NOT shipping AND NOT reset THEN
1130     visibility_rack_1 := TRUE;
1131     visibility_rack_2 := FALSE;
1132 END_IF
1133
1134 IF shipping THEN
1135     visibility_rack_1 := FALSE;
1136     visibility_rack_2 := TRUE;
1137 END_IF
1138
1139 // Visibility of DELIVERED
1140 IF reset THEN
1141     visibility_delivered := FALSE;
1142 END_IF
1143
1144 IF NOT reset THEN
1145     visibility_delivered := TRUE;
1146 END_IF
1147
1148 // Visibility of shipping section
1149 IF shipping THEN
1150     visibility_shipping := FALSE;
1151     visibility_rack_1 := TRUE;
1152     visibility_rack_2 := TRUE;
1153     visibility_box_stack_1 := TRUE;
1154     visibility_box_stack_2 := TRUE;
1155     visibility_box_stack_3 := TRUE;
1156     visibility_box_stack_4 := TRUE;
1157     visibility_box_stack_5 := TRUE;
1158     visibility_box_stack_6 := TRUE;
1159     visibility_total_production := TRUE;
1160 END_IF
1161
1162 IF NOT shipping THEN
1163     visibility_shipping := TRUE;
1164     visibility_rack_1 := FALSE;
1165     visibility_rack_2 := FALSE;
1166     visibility_box_stack_1 := FALSE;
1167     visibility_box_stack_2 := FALSE;
1168     visibility_box_stack_3 := FALSE;
1169     visibility_box_stack_4 := FALSE;
1170     visibility_box_stack_5 := FALSE;
1171     visibility_box_stack_6 := FALSE;
1172     visibility_total_production := FALSE;
1173 END_IF
1174
1175 // Visibility of the rack 1 or 2
1176 IF NOT shipping AND NOT reset THEN
1177     visibility_rack_1 := TRUE;
1178     visibility_rack_2 := FALSE;
1179 END_IF
1180
1181 IF shipping THEN
1182     visibility_rack_1 := FALSE;
1183     visibility_rack_2 := TRUE;
1184 END_IF
1185
1186 // Visibility of DELIVERED
1187 IF reset THEN
1188     visibility_delivered := FALSE;
1189 END_IF
1190
1191 IF NOT reset THEN
1192     visibility_delivered := TRUE;
1193 END_IF
1194
1195 // Visibility of shipping section
1196 IF shipping THEN
1197     visibility_shipping := FALSE;
1198     visibility_rack_1 := TRUE;
1199     visibility_rack_2 := TRUE;
1200     visibility_box_stack_1 := TRUE;
1201     visibility_box_stack_2 := TRUE;
1202     visibility_box_stack_3 := TRUE;
1203     visibility_box_stack_4 := TRUE;
1204     visibility_box_stack_5 := TRUE;
1205     visibility_box_stack_6 := TRUE;
1206     visibility_total_production := TRUE;
1207 END_IF
1208
1209 IF NOT shipping THEN
1210     visibility_shipping := TRUE;
1211     visibility
```

# 06 CODESYS - MAIN PLC\_PRG



The **MAIN POU** is the most important POU of the whole project, because it takes all the information from the other POUs, and it organizes them to let them work together.

Firstly, we have that the **start button** (either from the HMI or from the PLC):

- set the variable run that -> open the valves source and exhaust of the pneumatic circuit.
- reset stop and emergency

Stop and emergency reset run



The main POU is responsible for linking the input and output of POU-blocks:

- HMI\_visibility
- cycle\_management
- action\_execution
- Emergency
- cylinder\_displacement\_calculation

On the left we have the **inputs** that enter to the block of the POU and on the right the **output** that exit from it.



# 07 HMI

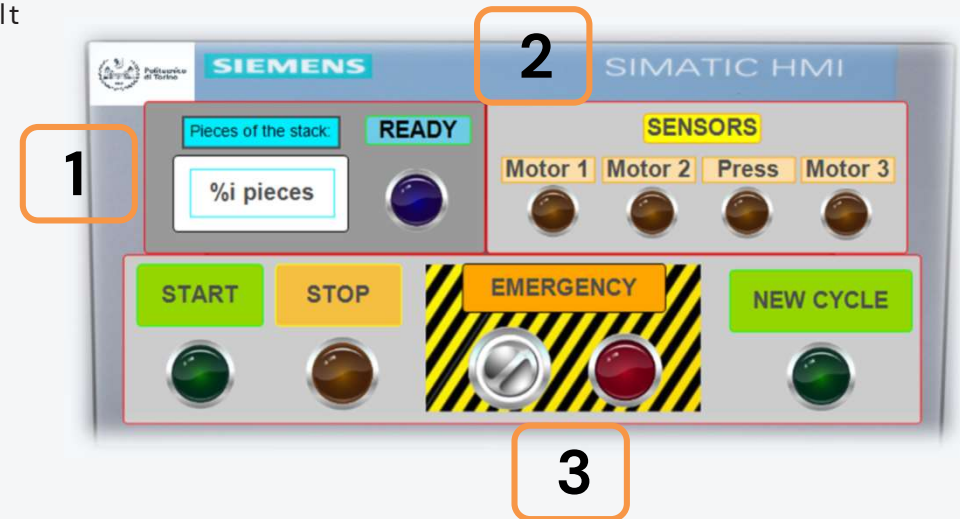
As far as the **Human-Machine Interface (HMI)** is concerned, we built a simple panel composed of three sections from where we can control the whole system.

The three sections are:

1. **Input**
2. **Sensors**
3. **Modes**

We used:

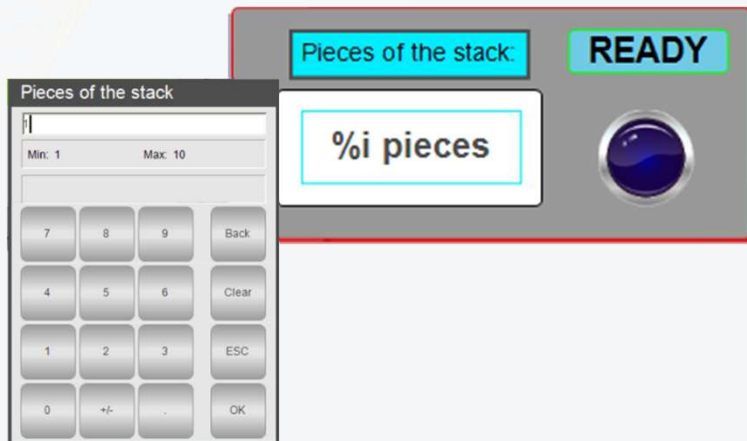
- **Push-buttons**
- **Selectors**
- **Lamps**
- **Input numpad**



Human-Machine Interface (HMI) is very important to make a connection between programming side and animations.

Thanks to that we can have a **representation**, physically simulated, to what the system should do in the real world if everything was in these conditions.

# 07 HMI – Input & Sensors



The first section is related to the **input**:

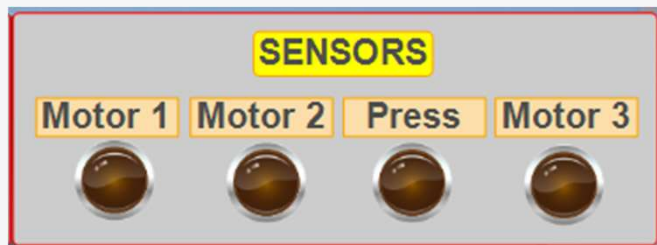
- at the beginning of each stack-cycle the operator is required to insert the number of pieces that the stack will be formed by.

A stack-cycle is made of the number of pieces selected by the operator that are going to be pressed and stocked.

Six stack-cycles compose a complete-cycle.

Clicking on the box *pieces* a **numeric pad** appears and the operator can insert a number (minimum 1 and maximum 10 to make the simulation more dynamic)

At the end of every stack-cycle the **blue lamp** will turn on indicating that the machine is ready to perform a *new cycle*.



The **sensors** are simulated by the **yellow lamps** and indicate the status of the motors, they turned on to indicates that motors are moving.

They are also present in correspondence of the motor in the plant.

# 07 HMI – Modes

According to the requirements the push-buttons are responsible to:

- **start** the cycle
- **stop** the cycle in the first available ending phase
- **stop** the cycle immediately in case of emergency and start a new cycle



## **START:**

- motor 1 is activated
- conveyor 1 starts to move
- the phases related to the boxes are activated

we will see as much boxes as we set on the input appear on the first conveyor once at time and max 3 at the same time.

## **STOP:**

- air in the pneumatics circuit is stopped
- the cycle will end in the next available phase.

## **EMERGENCY:**

- the whole system is stopped
- all the conveyors are stopped
- all the boxes are stopped

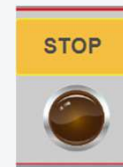
The emergency is essential, and it has the highest priority. There is a red lamp that allows us to check the emergency status.

We can then activate again the conveyors and start again the cycles with the boxes only if the switch is placed in the original position.

## **NEW CYCLE:**

- reset all the variables
- make available a new cycle

is used when we have 6 stacks ready, shipped and delivered.



# 08 Animations

As last step we built the **model graphically** to be able to create animations and test the system.

We created everything from scratch, with lines, round and square shapes, using the **Visualization Toolbox** in CODESYS, in particular, we drew:

- 10 boxes
- 2 conveyors
- 1 piston
- 1 shipping station
- 1 delivered station
- 2 timers
- 1 counter

We also added pictures to make the simulation more real:

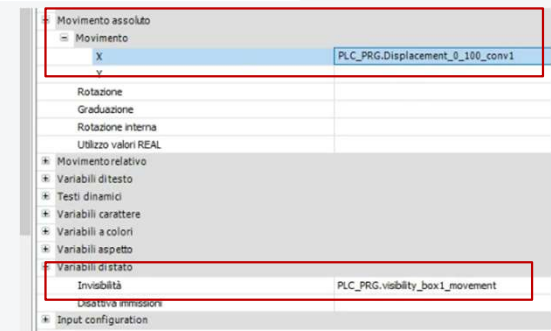
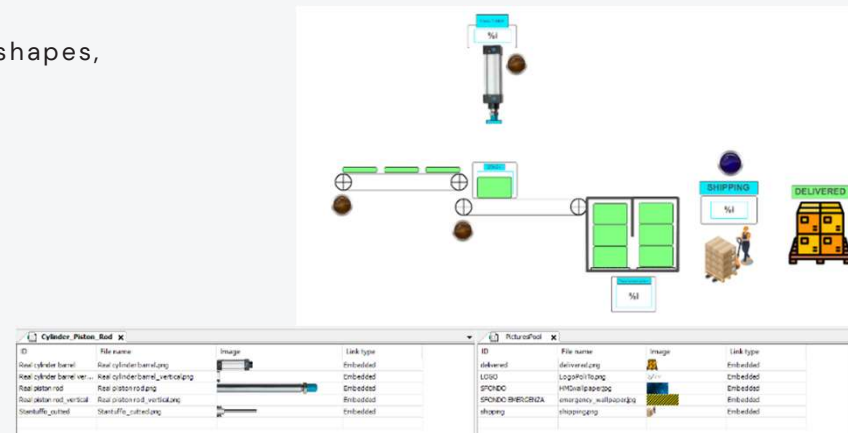
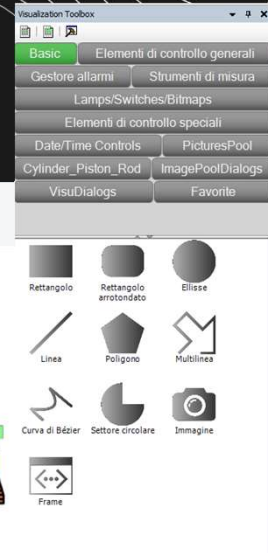
- for the emergency area
- for the HMI panel
- for the shipping and delivered station.

We embedded them into the projects thanks to the section *Pictures*

To be able to control the animations with the loops programmed with *Ladder code* and *Structured Text*, we had to **connect the objects with pre-defined variables**.

We worked with *two types of animations*:

- **Visibility**
- **Translation**

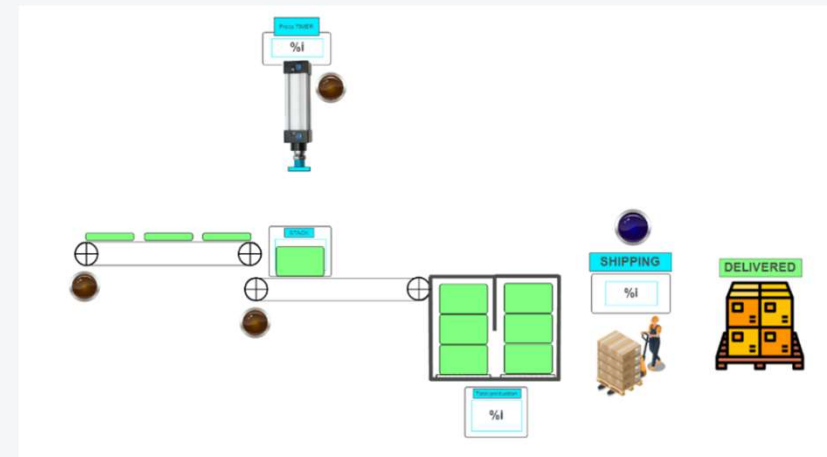


# 08 Animations - Pictures

To better visualize our plant, we built the whole system from scratch using the visualization toolbox of CODESYS.

We used rectangles, ellipsoids, lines, counters, lamps, buttons and so on.

To make a complete simulation we added animations, as for the boxes but also for the stoking, the shipping and the delivering stations. We had timers and counters, and we mixed everything programming visibilities and linking variables.



For example, as far the lamps are concerned, they are simply linked to a variables of the MAIN POU. While the plates are linked both to the visibility variable to be visible only when it is necessary and to the displacement variable to be able to move and simulate the conveyor.

# FINAL RESULT

Single stack-cycle with 4 pieces

TEST



# FINAL RESULT

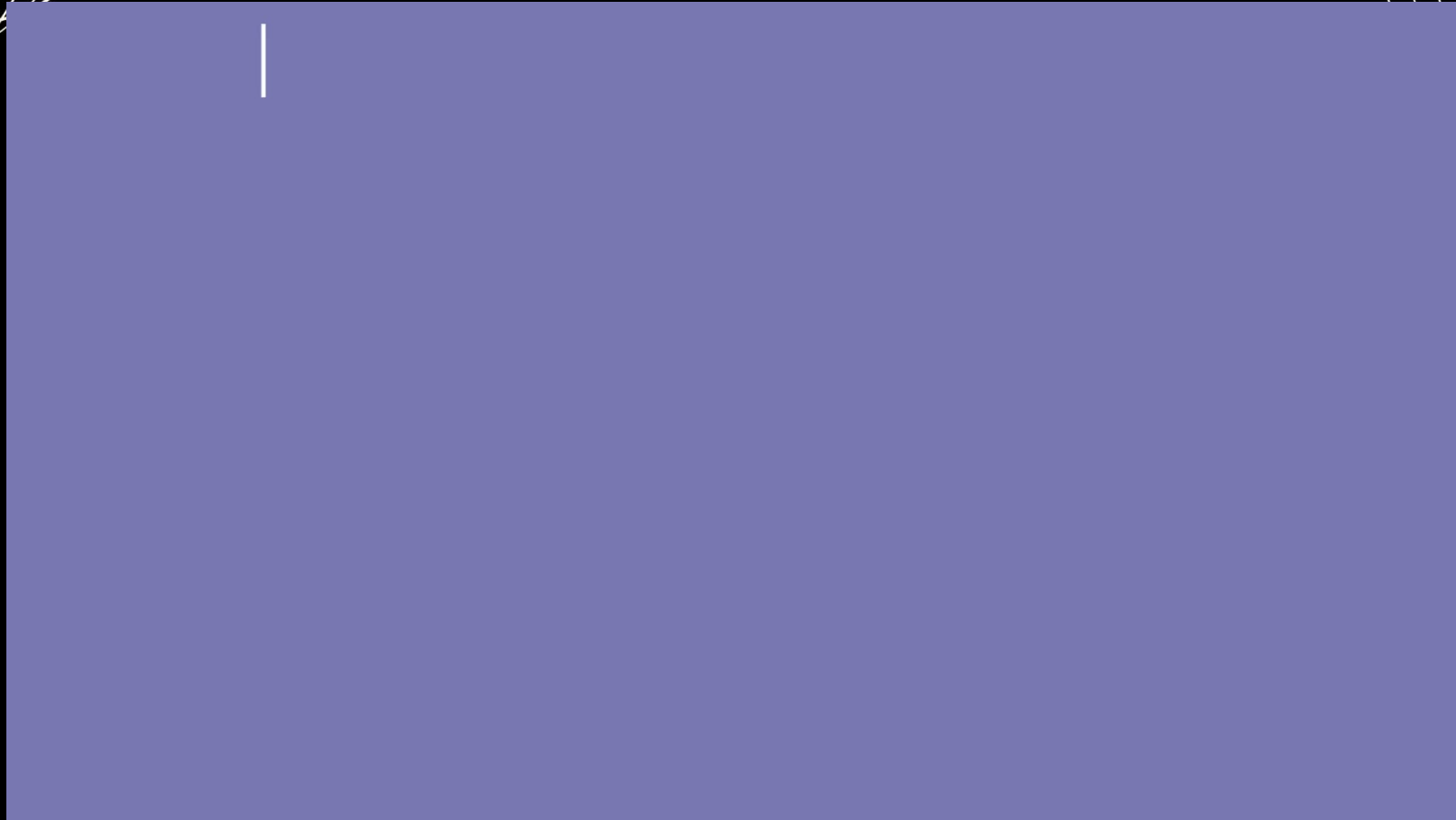
Complete cycle

I



# FINAL RESULT

Stop & Emergency



# OUR TEAM



Luigi  
Muratore  
s333098



Muhammad Fatir  
Noshab  
s331898

Iskandar  
Akbarov  
s329650

# THANKS FOR WATCHING



**Politecnico  
di Torino**