PLC Homework

Industrial Automation mod. B

Claudia Vinci

Viviana Casale

Mary Aliberto

Index

Introduction

This project aims to create an industrial process for assembling and decorating cakes from pre-baked sponge cake bases. The line is divided into sequential stations: soaking the base, spreading the cream, positioning the second layer, masking, and final topping. The goal is to demonstrate, in a Siemens TIA Portal environment, the coordinated control of these phases via PLC and HMI, with state logic and recipes that can be set from the panel.

The operating process begins with the arrival of the base on the conveyor belt. Proximity sensors detect the product's position and synchronize the activation of the volumetric dosers: in the first station for the soaking, in the second for the cream. Next, a robotic arm picks and places the second layer; then the masking station covers the entire surface, and finally, the topping station applies the selected decorative element.

The HMI allows start/stop, recipe selection and initialization (type of soaking, cream, and topping), and setting the number of cakes to be produced.

The designed system represents an intermediate module in a larger production chain: dough preparation, baking, and eventual cutting of the discs are considered upstream in this work; packaging and shipping are planned downstream.

Demonstration video:

demo_PLC.mkv

Components

The project involves the use of the following components.

Central Control

- **PLC**: the entire system is controlled by a SIEMENS SIMATIC S7-1200 CPU 1212C AC/DC/RLY. The whole process is simulated using TIA Portal V16.

Interface

- **HMI display** (KTP400 Basic PN): used to start/stop the process, set the number of cakes to assemble, and select parameters such as the type of syrup, cream flavor, topping/grains, and so on.

Sensors and Actuators

- **Proximity sensors**: detect the presence of the cake at each station. They synchronize the dosing operations (soaking, cream, topping) and the pick-and-place of the second layer, preventing dispensing or movements when the product is not in position.

- **Volumetric dispenser**: used in most stations (base soaking, cream filling, and topping) to deliver the exact quantity.

- **Robotic arm**: used to pick and place the second sponge layer onto the cream filling.

A conveyor belt is used throughout the entire process to transfer the cake between the various stations, ensuring that it moves at the correct pace along the line.

Prices

Although the project focuses on simulation in TIA Portal, building the physical line requires a significant investment due to industrial components and, above all, food-grade machinery that meets hygienic standards.

The estimated total investment for a complete implementation falls within a broad range: ~€30,500 to €70,000.

We can break costs down as follows:

Automation and Control Costs

- PLC + HMI (SIEMENS SIMATIC S7-1200 + KTP400 Basic PN): approx. €500–€800 combined (basic lab/education configuration).
- Sensors + electrical panel: ~€2,000–€3,500 (photoelectric/proximity sensors, terminals, protection devices, power supplies, wiring, enclosure).

Specialized Mechanics (main cost drivers)

- Volumetric dispensers (3): ~€3,000–€7,500 each ⇒ ~€9,000–€22,500 total (soaking, filling, topping).
- Pick-and-place robotic arm: ~€15,000–€35,000 (hygienic compatibility, wash-down, food-safe end-effector, cables, and seals).
- Food-grade conveyor: ~€4,000–€8,000 (stainless-steel frame, compliant belt, guards, adjustments, and supports).

In summary, this is a mid-to-upper-range project in terms of budget, where hygienic requirements and the mechanical complexity of dispensers and specialized robotics are the primary cost drivers.

Variable tables

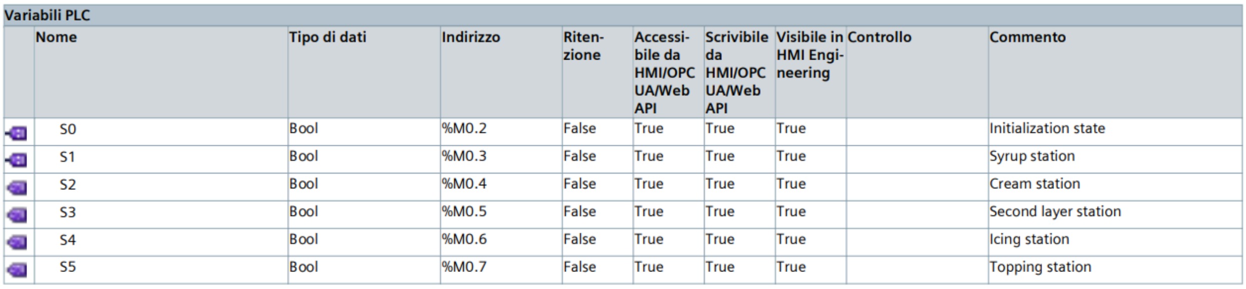
This section lists all the variables defined in order to implement the process. The context of usage of each variable will be explained later in the ladder program section.

Two main variable types are used:

- ***Memory variables (%M):*** act as internal flags that the controller can set or reset independently of physical inputs and outputs. They are essential in ladder logic because they allow the program to remember the status of a process between scan cycles, which is especially important for state-machine control. Using the SET instruction forces a memory bit to stay TRUE even after the original input signal is gone, while the RESET instruction forces it back to FALSE when the process or conditions demand it. This latching behavior provides the PLC a way to store and manage the active states of a system.

- ***DB variables:*** DB variables are elements stored inside *data blocks*, which provide structured and organized memory areas for more complex or persistent data: they can hold timers, counters, integers, or other parameters. For example, each timer (Timer_S1 to Timer_S5) has its own DB to store preset and elapsed times along with the done bit, the Cake_counter is stored in DB7 to track the number of completed products, and the DBSelection block holds recipe choices such as cream, syrup, and toppings, and also the counter variable to define the number of cakes to be produced.

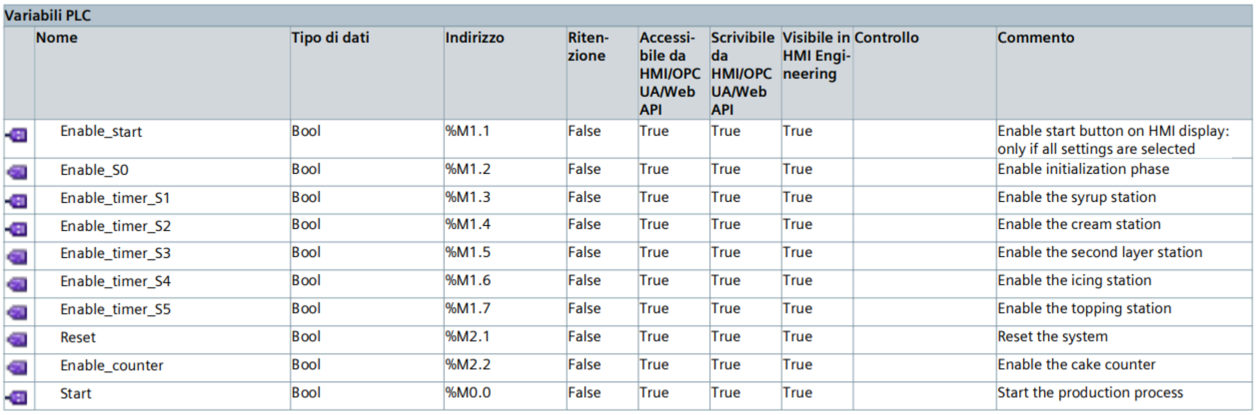
State variables



Variabili PLC

Nome	Tipo di dati	Indirizzo	Ritenzione	Accessibile da HMI/OPC UA/Web API	Scrivibile da HMI/OPC UA/Web API	Visibile in HMI Engineering	Controllo	Commento
S0	Bool	%M0.2	False	True	True	True		Initialization state
S1	Bool	%M0.3	False	True	True	True		Syrup station
S2	Bool	%M0.4	False	True	True	True		Cream station
S3	Bool	%M0.5	False	True	True	True		Second layer station
S4	Bool	%M0.6	False	True	True	True		Icing station
S5	Bool	%M0.7	False	True	True	True		Topping station

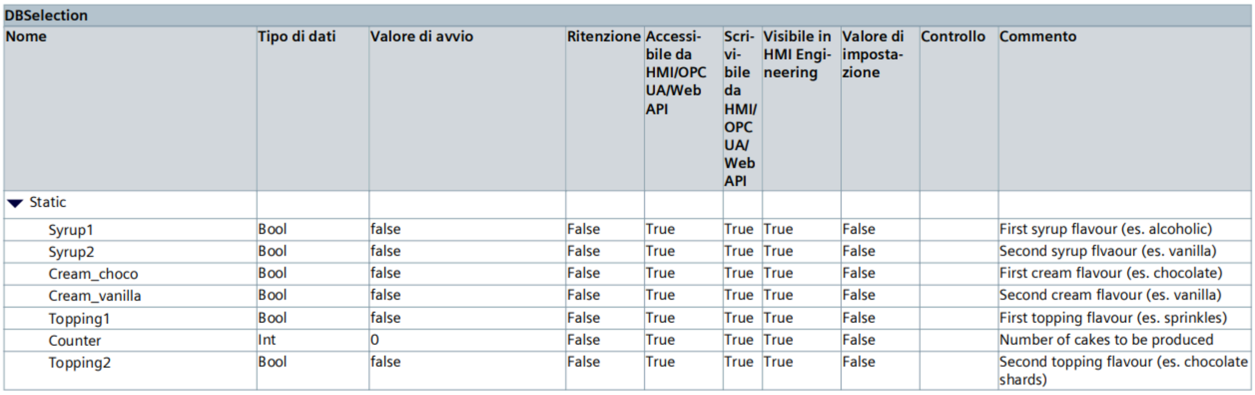
PLC variables



Variabili PLC

	Nome	Tipo di dati	Indirizzo	Riten-zione	Accessi-bile da HMI/OPC UA/Web API	Scrivibile da HMI/OPC UA/Web API	Visibile in HMI Engi-neering	Controllo	Commento
	Enable_start	Bool	%M1.1	False	True	True	True		Enable start button on HMI display: only if all settings are selected
	Enable_S0	Bool	%M1.2	False	True	True	True		Enable initialization phase
	Enable_timer_S1	Bool	%M1.3	False	True	True	True		Enable the syrup station
	Enable_timer_S2	Bool	%M1.4	False	True	True	True		Enable the cream station
	Enable_timer_S3	Bool	%M1.5	False	True	True	True		Enable the second layer station
	Enable_timer_S4	Bool	%M1.6	False	True	True	True		Enable the icing station
	Enable_timer_S5	Bool	%M1.7	False	True	True	True		Enable the topping station
	Reset	Bool	%M2.1	False	True	True	True		Reset the system
	Enable_counter	Bool	%M2.2	False	True	True	True		Enable the cake counter
	Start	Bool	%M0.0	False	True	True	True		Start the production process

DB Selection



DBSelection

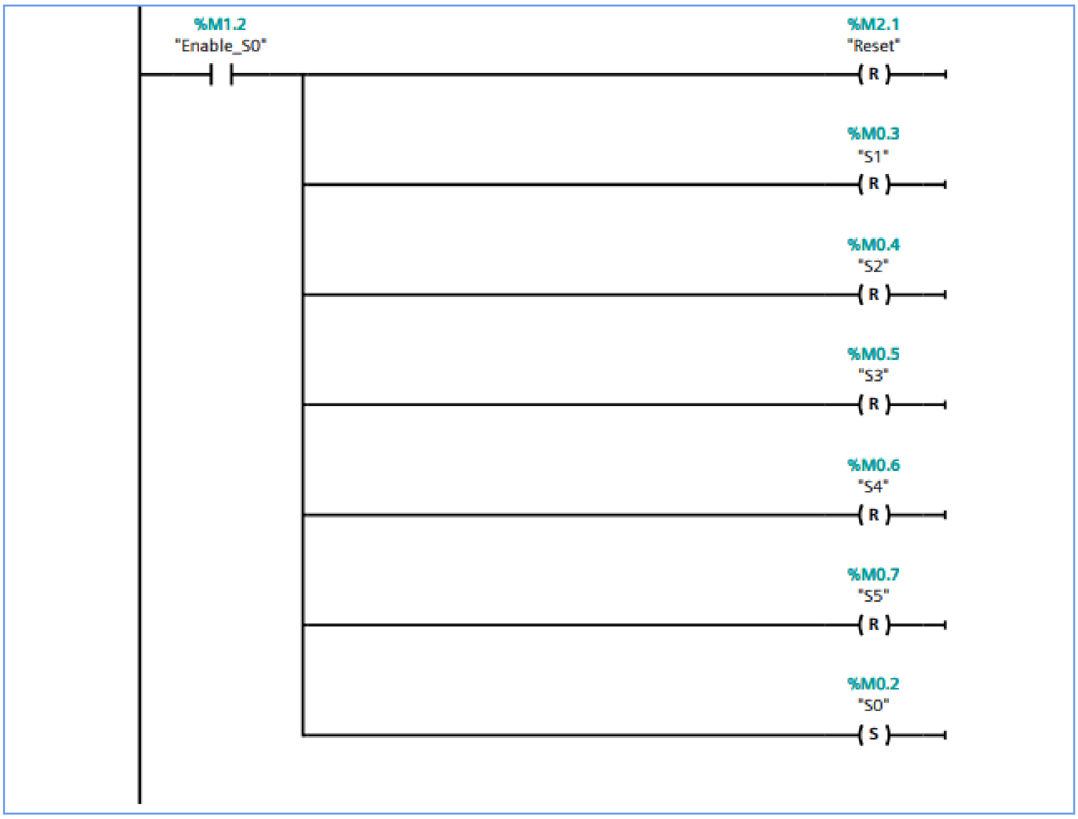
Nome	Tipo di dati	Valore di avvio	Ritenzione	Accessi-bile da HMI/OPC UA/Web API	Scri-vi-bile da HMI/OPC UA/Web API	Visibile in HMI Engi-neering	Valore di imposta-zione	Controllo	Commento
▼ Static									
Syrup1	Bool	false	False	True	True	True	False		First syrup flavour (es. alcoholic)
Syrup2	Bool	false	False	True	True	True	False		Second syrup flvaour (es. vanilla)
Cream_choco	Bool	false	False	True	True	True	False		First cream flavour (es. chocolate)
Cream_vanilla	Bool	false	False	True	True	True	False		Second cream flavour (es. vanilla)
Topping1	Bool	false	False	True	True	True	False		First topping flavour (es. sprinkles)
Counter	Int	0	False	True	True	True	False		Number of cakes to be produced
Topping2	Bool	false	False	True	True	True	False		Second topping flavour (es. chocolate shards)

Ladder Logic Program

The ladder logic program automates a cake assembly process through a sequence of well-defined states, each representing an operational phase of the production line. The ladder logic diagram uses a state-machine structure where **S0** represents the initial setup and recipe selection, followed by **S1** to **S5**, which each control a timed production step such as dispensing syrup, cream, or toppings.

The ladder scheme is organized into 9 segments: the first one initializes the system*;* in the next segment all the *state transitions* are defined; then, for each of the 6 system states, the functioning is implemented; in the last segment, the reset commands are handled.

Segment 1: System Initialization



The system is initialized by *resetting* all the states from *S1 to S5* and the *Reset state*, while *setting* the initial state S0, which is the initialization state where all the choices for the cakes' production are made. The *set* and *reset* operations are performed only once the ***Enable_S0*** variable is set to True; this happens when the "*Initialize a new process*" button is pressed on the HMI, thus each time we want to start a new production process.

Segment 2: State transitions

The second segment ensures that each cake passes through all processing stations in order, with the transition from one phase to the next determined by the logic implemented in the ladder diagram.

In fact, it defines all the state transitions that regulate the sequence of steps in the industrial process. Each step (S1–S5) is controlled by a timer and represents an operational phase of the production line. In particular, after the initialization phase:

1. Soaking the cake layer with syrup;
2. Spreading the cream;
3. Placing the second layer;
4. Covering with cream;
5. Adding the topping.

The timers were used in order to simulate the volumetric dispenser functioning. The completion of one step (through the timer signal) enables the next, thereby creating a sequential flow.

More in detail, the first transition from ***S0 to S1***, happens when:

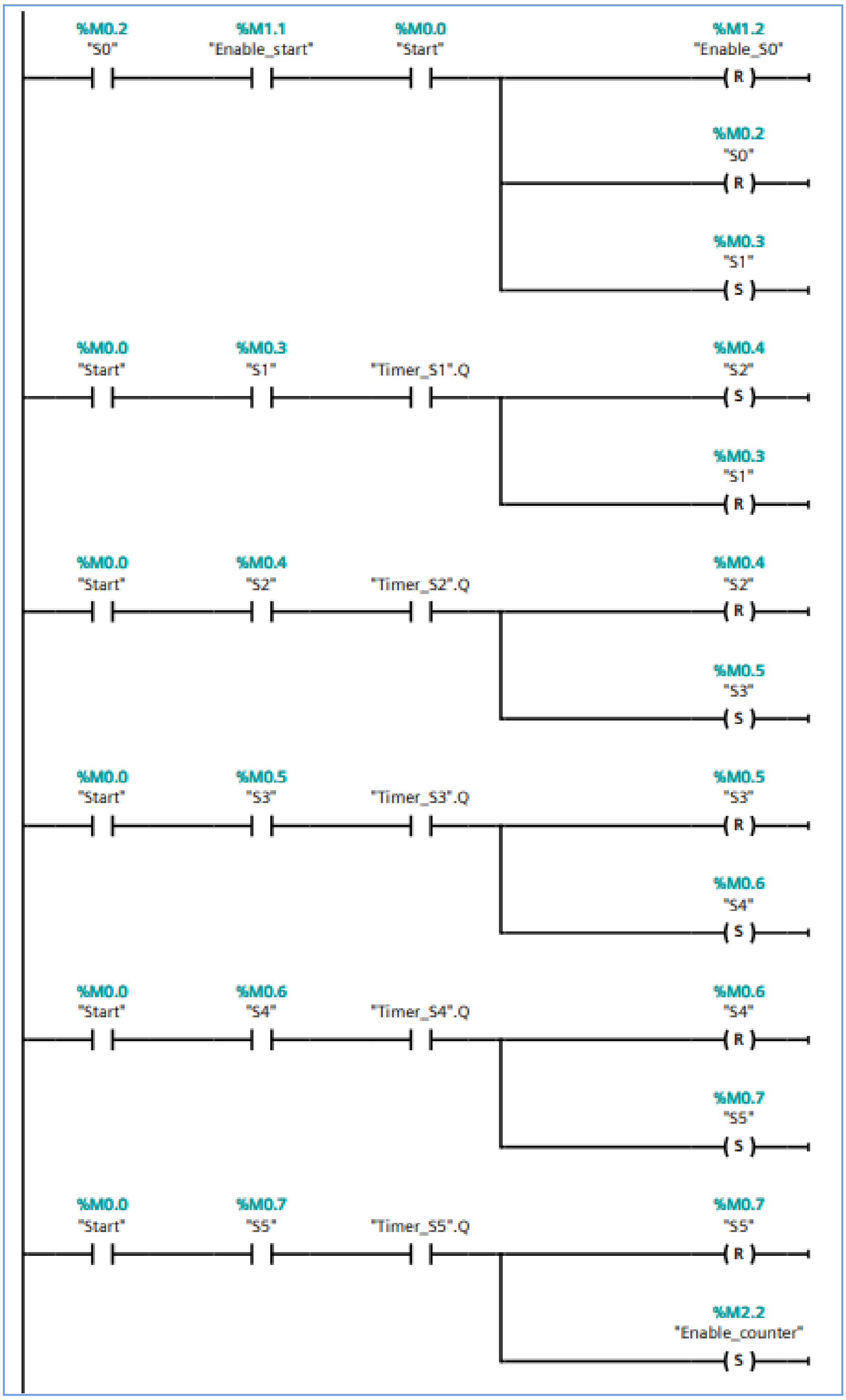
- ***S0*** is set, meaning that, from the HMI display, a new process was enabled by clicking on the "*initialize new process*" button and the system was then initialized (as illustrated in the previous segment).
- ***Enable_start*** is set, meaning that all the recipe choices about syrup, cream, topping and number of cakes have been selected from the HMI display; this allows the *Start* button to appear on HMI display.
- ***Start*** is set, meaning that the process has been started by clicking on the start button on the HMI display.

At this point, the variables *Enable_S0* and *S0* are reset, while the state *S1* is set, allowing the transition.

Each of the transitions, from ***S1 to S5***, follow the same scheme and happens if:

- ***Start*** is set, meaning the process is still running and has not been paused or stopped from the HMI display.
- The ***current state*** is set, meaning the process is running in that particular state: the cake is located in that station and recognized by the proximity sensor.
- The ***timer output*** is set, meaning that the timer corresponding to the current state has expired: the volumetric dispenser operation is done.

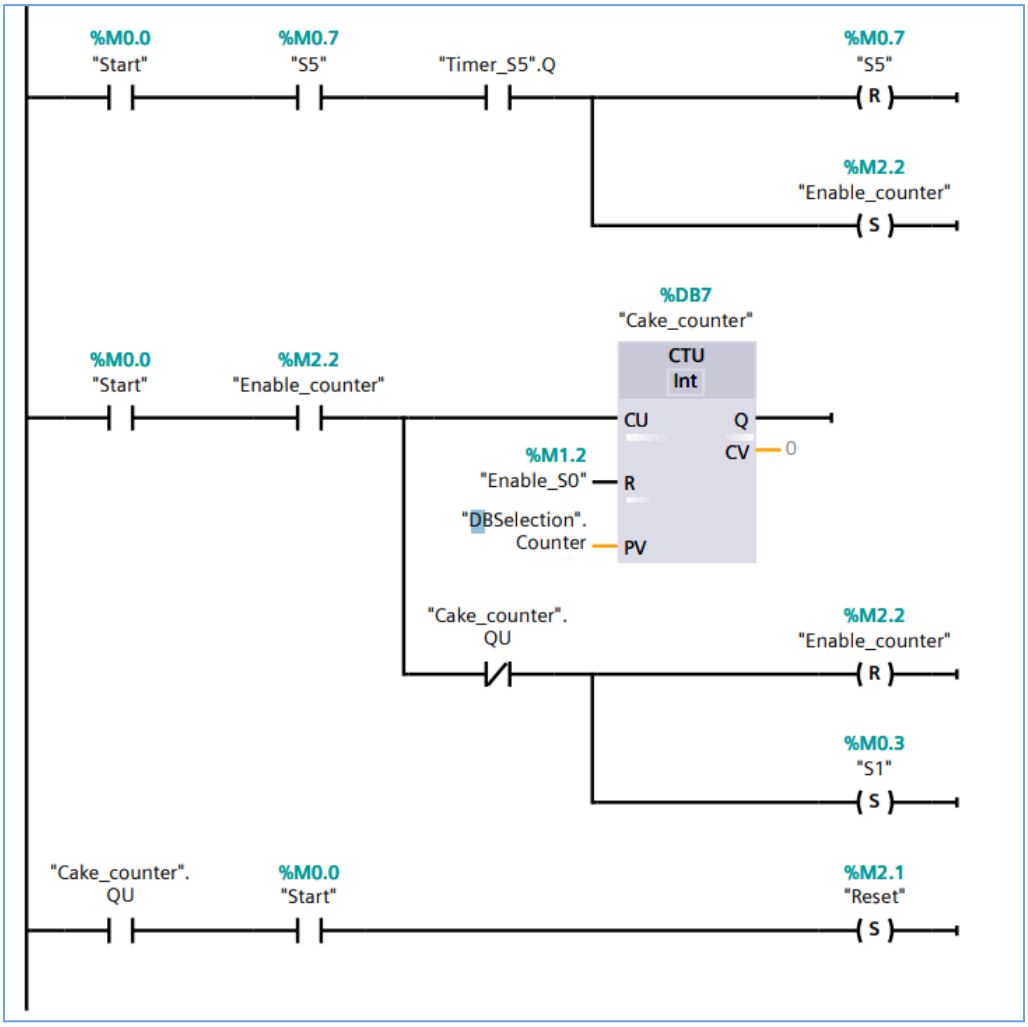
If so, the current state (which now becomes the previous) is *reset*, while the next one is *reset.*



When the last state (S5) is reset, ***Enable_counter*** is set: this allows the *Cake_counter* to count the completed cake. This happens if the process is still running, meaning that also ***Start*** is set, hence the process has not been paused or stopped).

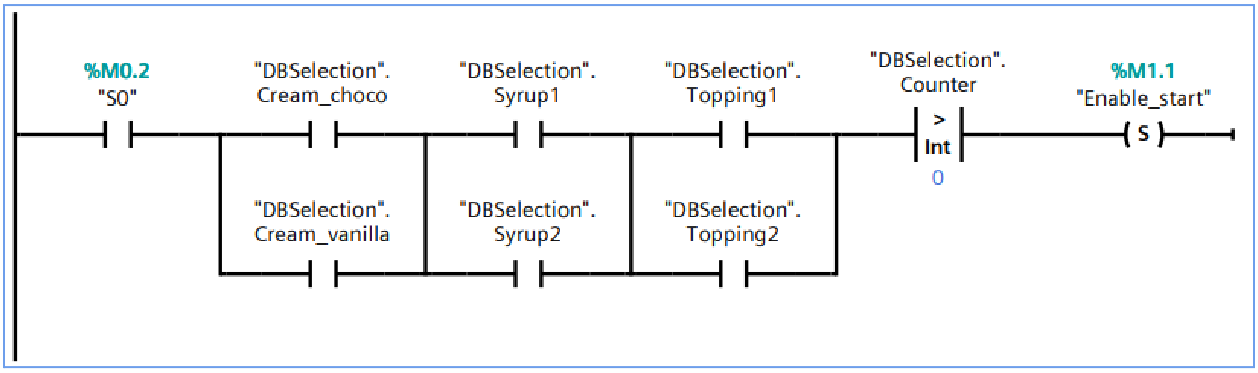
Now, there are two possibilities:

- If the ***Cake_couter output*** (Cake_output.QU) is set, it means that the number of completed cakes has reached the one defined from the HMI display at the initialization step, thus the process is completed and the state *Reset* is set.
- Otherwise, the process starts again by resetting the *Enable_counter* variable and setting the *S1* state to start assembling the next cake.



Note: the counter is reset each time *Enable_S0* is set, so each time a new process is initialized.

Segment 3: S0 state functioning

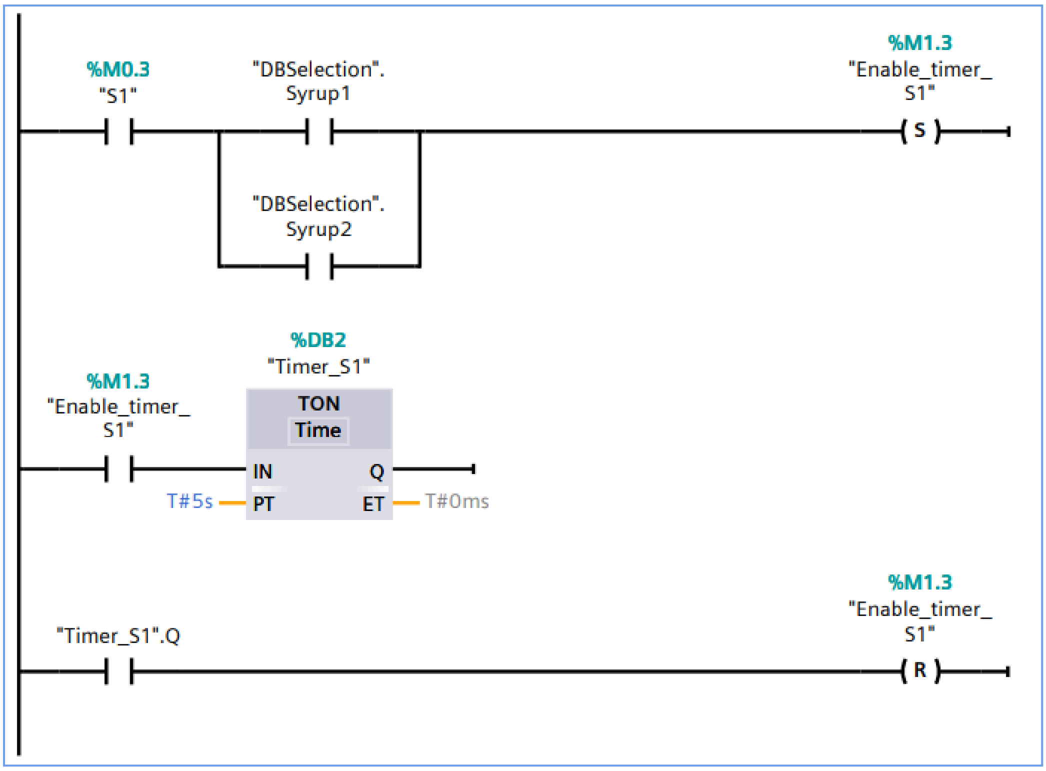


The state S0 is the initialization state, where the choices about the cakes to be produced are made before starting the process.

The ***Enable_start*** variable is *set* (allowing the process to start running once the *Start* button is clicked on HMI screen) only if:

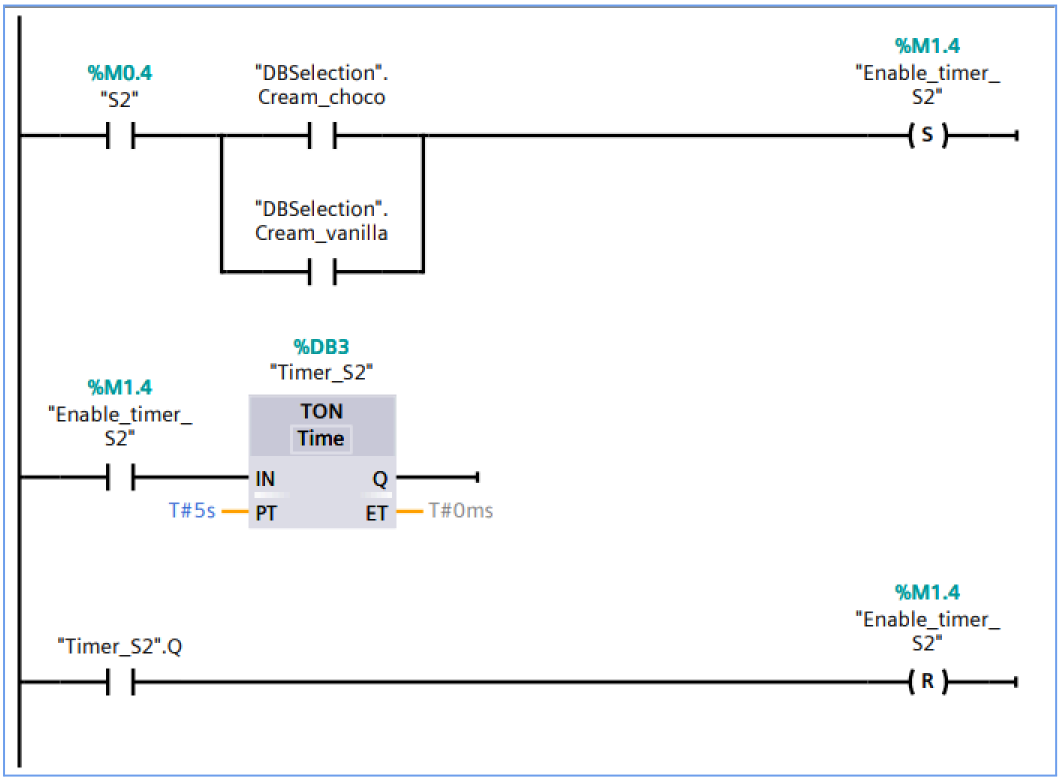
- ***S0*** is set, meaning the process is indeed in the initialization state;
- One of the two creams is selected, so one of the two variables *DBSelection.**Cream_choco*** or *DBSelection.**Cream_vanilla*** is set by clicking on the respective button on the HMI display;
- One of the two syrups is selected, so one of the two variables *DBSelection.**Syrup1*** or *DBSelection.**Syrup2*** is set by clicking on the respective button on the HMI display;
- One of the two toppings is selected, so one of the two variables *DBSelection.**Topping1*** or *DBSelection.**Topping2*** is set by clicking on the respective button on the HMI display;
- The variable *DBSelection.**Counter*** is set and greater than zero, meaning that the number of cakes to be produced is specified.

Segment 4: S1 state functioning

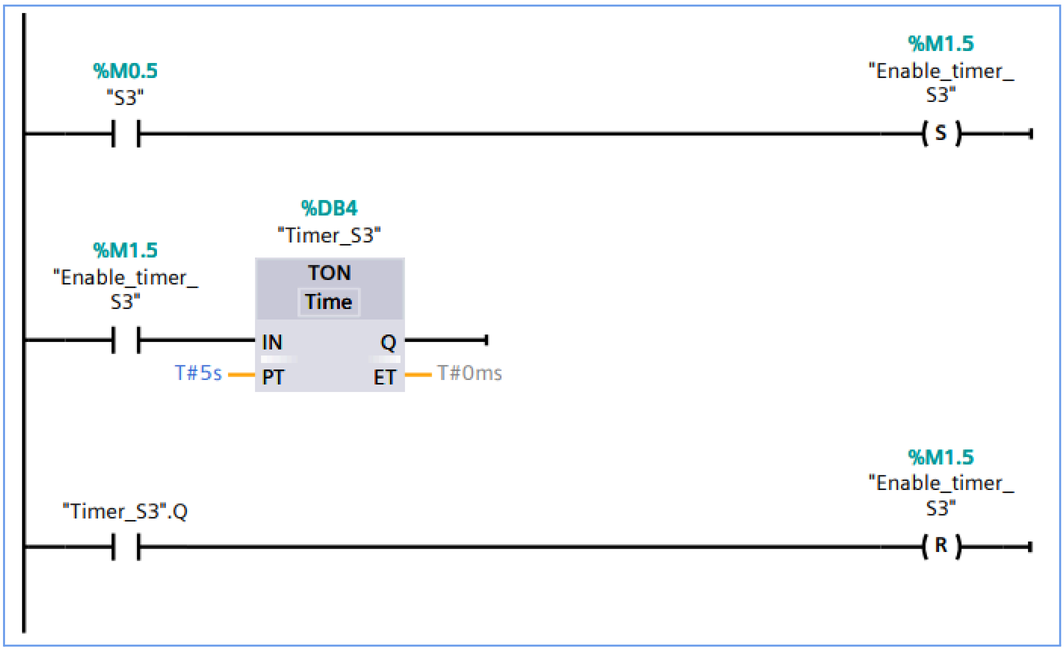


Once the system is in the S1 state, if a syrup has been selected, the volumetric dispenser starts operating. To simulate the syrup dispense, a timer is used; thus, the variable ***Enable_timer_S1*** is set. This allows the timer to start; once it is finished, the timer is reset and the state transition (as explained in the *State transition* segment) is allowed.

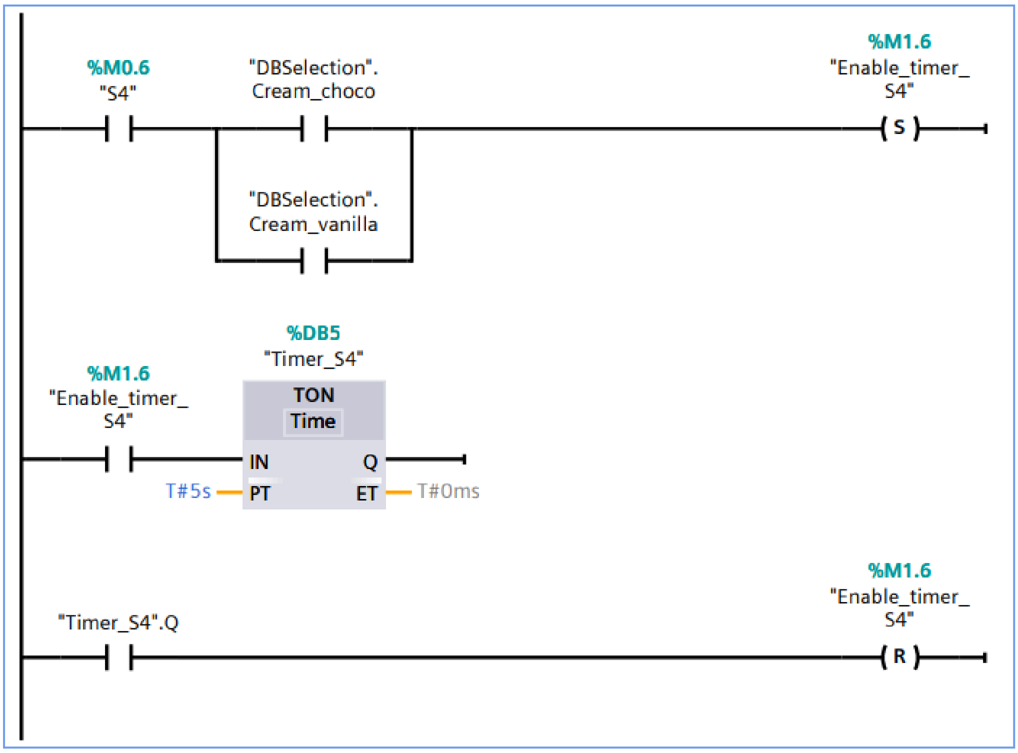
Segment 5: S2 state functioning



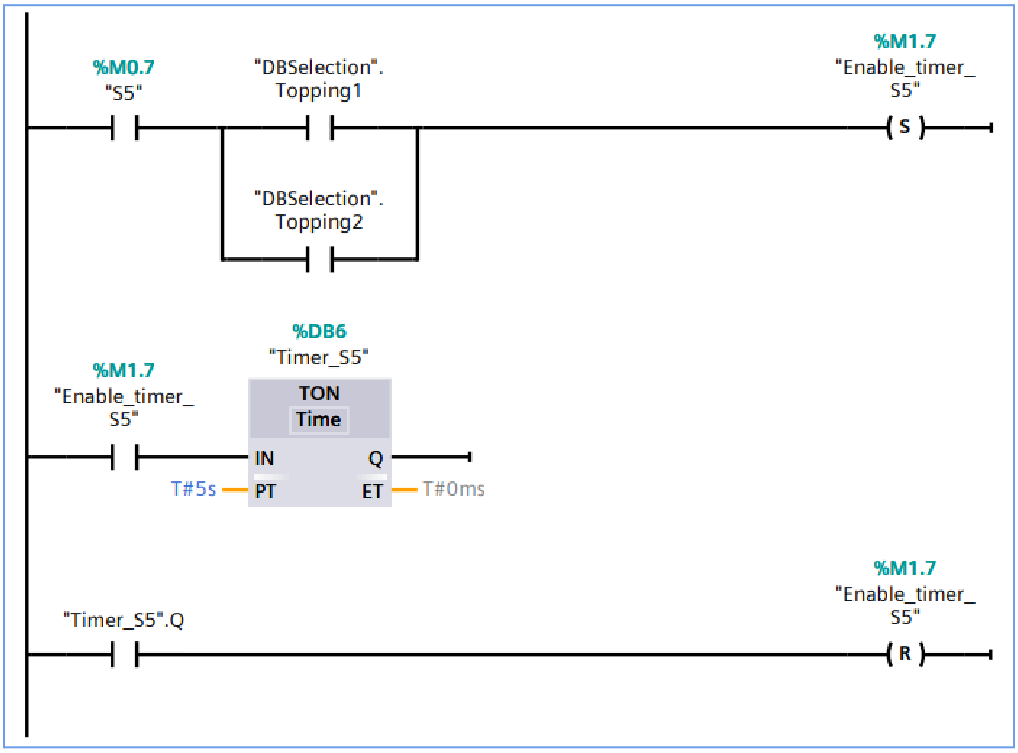
Segment 6: S3 state functioning



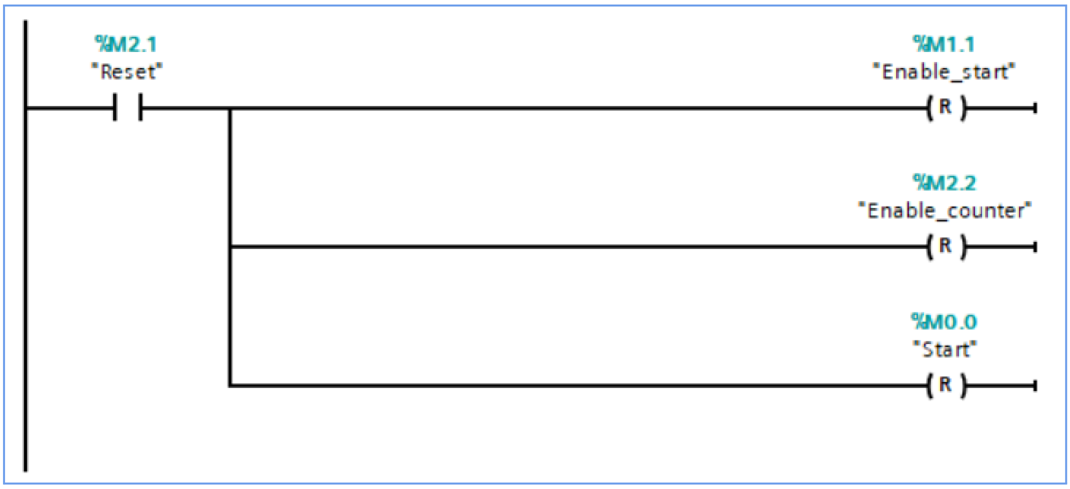
Segment 7: S4 state functioning



Segment 8: S5 state functioning



Segment 9: Reset state functioning



The ***Reset*** variable is set in two cases: rather the process is finished, or the process is stopped, by clicking on the *Stop* button on the HMI display while running, and then the reset through the *Reset* button.

When this happens:

- ***Enable_start*** is reset;
- ***Enable_counter*** is reset;
- ***Start*** is reset.

HMI

A ***Human-Machine Interface*** (HMI) is a system or device that enables interaction between people and machines, allowing operators to monitor, control, and manage processes effectively. It provides a graphical representation of data, often through touchscreens, control panels, or software dashboards, making complex systems easier to understand and operate.

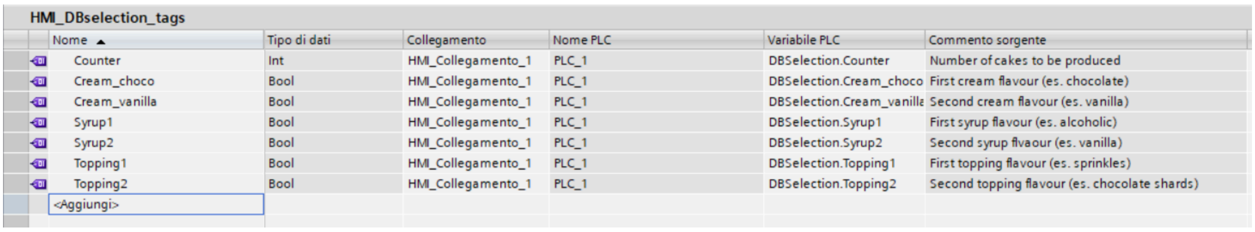
Here, the importance of the HMI display relies on the possibility to easily set all the choices for the production process (syrup flavour, cream and topping), and also on the possibility to start and stop the process when necessary, and also to reset the system. Finally, another important feature is the possibility to monitor the process, since on the HMI display it is possible to see if the system is running and its operational state.

HMI variables

In the automation system, the HMI variables are directly linked to the corresponding PLC variables in order to ensure real-time monitoring and control of the process.

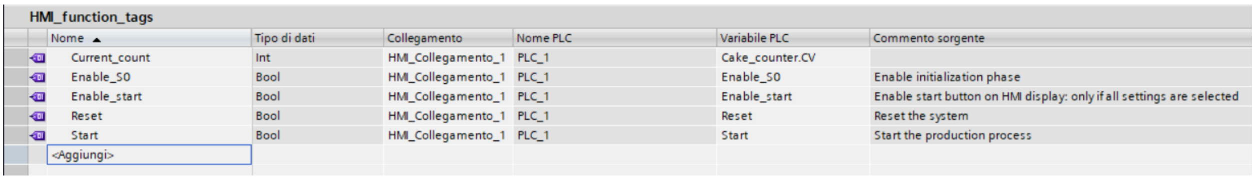
Each HMI tag is mapped to a specific PLC memory address, so that operator inputs on the HMI (such as start, stop, or reset commands) are written to the PLC, while process information (such as step status, timers, and the cake counter value) is read from the PLC and displayed on the HMI screen.

This bidirectional communication allows operators to supervise the sequence, follow the progression of each step, and check production data, while also enabling manual intervention when required.



HMI_DBselection_tags

Nome ▲	Tipo di dati	Collegamento	Nome PLC	Variabile PLC	Commento sorgente
Counter	Int	HMI_Collegamento_1	PLC_1	DBSelection.Counter	Number of cakes to be produced
Cream_choco	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Cream_choco	First cream flavour (es. chocolate)
Cream_vanilla	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Cream_vanilla	Second cream flavour (es. vanilla)
Syrup1	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Syrup1	First syrup flavour (es. alcoholic)
Syrup2	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Syrup2	Second syrup flvaour (es. vanilla)
Topping1	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Topping1	First topping flavour (es. sprinkles)
Topping2	Bool	HMI_Collegamento_1	PLC_1	DBSelection.Topping2	Second topping flavour (es. chocolate shards)
<Aggiungi>					

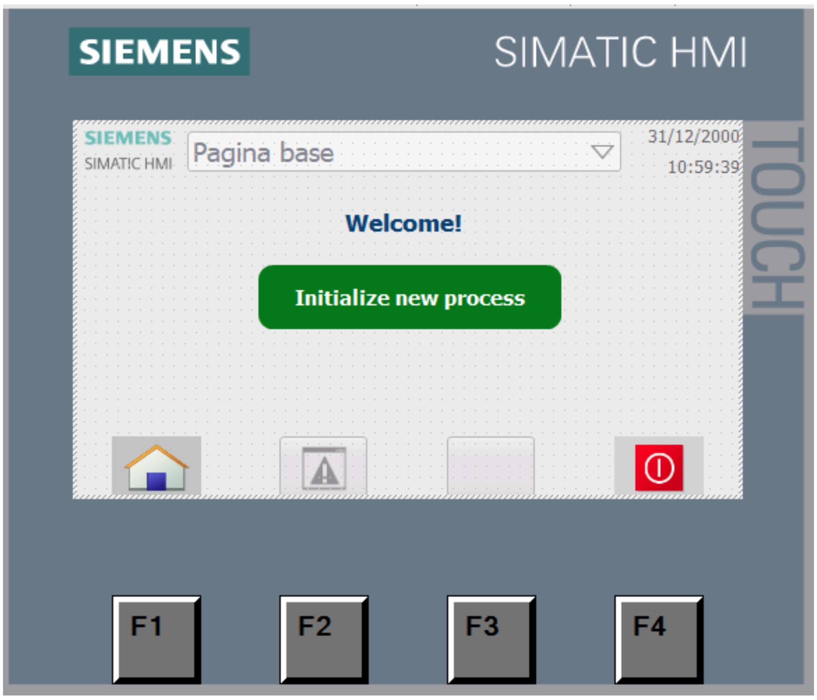


HMI_function_tags

Nome ▲	Tipo di dati	Collegamento	Nome PLC	Variabile PLC	Commento sorgente
Current_count	Int	HMI_Collegamento_1	PLC_1	Cake_counter.CV	
Enable_S0	Bool	HMI_Collegamento_1	PLC_1	Enable_S0	Enable initialization phase
Enable_start	Bool	HMI_Collegamento_1	PLC_1	Enable_start	Enable start button on HMI display: only if all settings are selected
Reset	Bool	HMI_Collegamento_1	PLC_1	Reset	Reset the system
Start	Bool	HMI_Collegamento_1	PLC_1	Start	Start the production process
<Aggiungi>					

Base Page

The Base page is the one that opens when the system is operating. It simply shows a button which allows to initialize a new process.

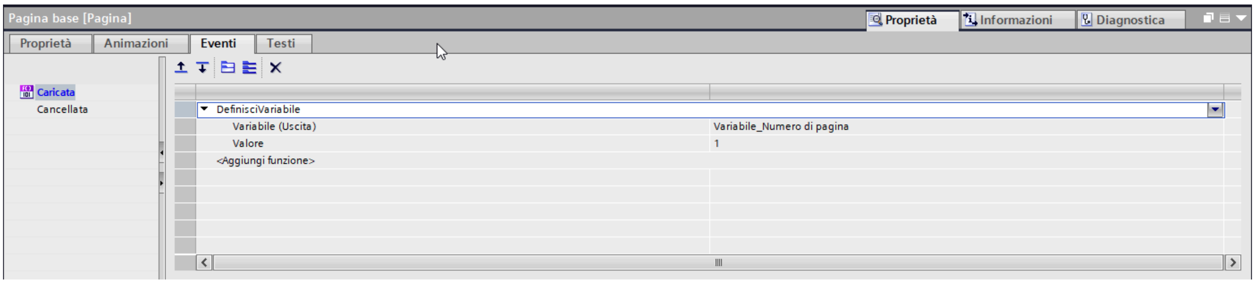


In this page, two main events are defined to control its behavior:

1. an event associated with page load;
2. an event associated with clicking the ***Initialize new process*** button.

Event 1: Page load

When this event occurs, the ***Page Number*** variable is set to 1.
This operation records the current state of the interface, marking this screen as the initial page.



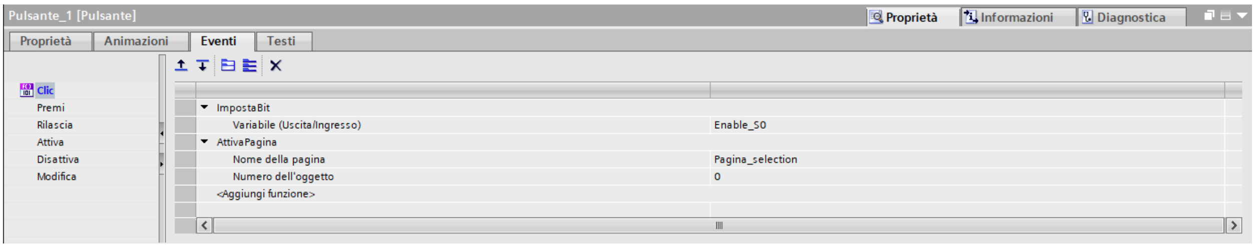
DefinisciVariabile
Variabile (Uscita) Variabile_Numero di pagina
Valore 1
<Aggiungi funzione>

Event 2: Click “Initialize new process”

This event occurs when the green button on the main page is clicked. During this event, the system performs two actions:

1. The ***Enable_S0*** variable is set, enabling the start of S0 state;

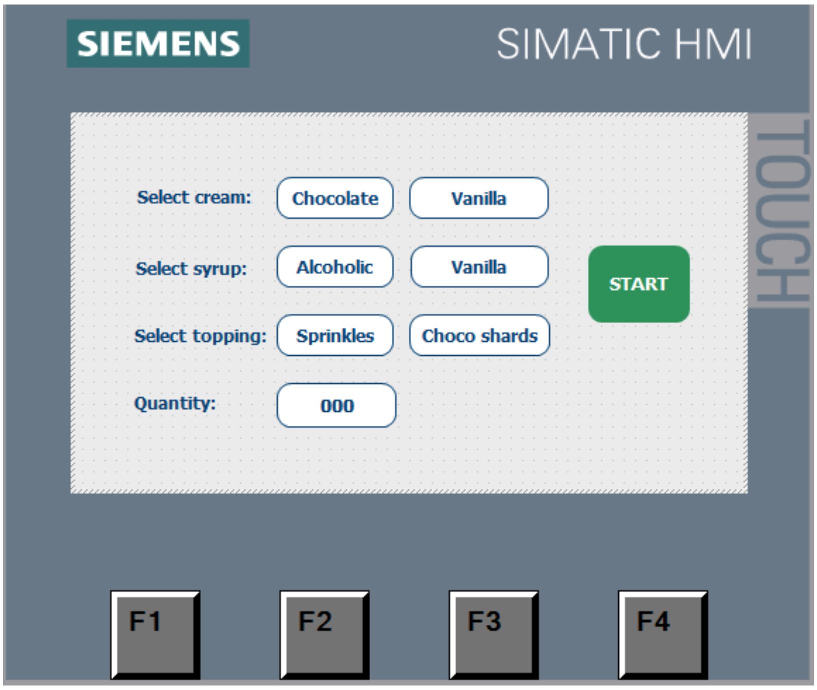
2. The ***Selection Page*** is activated, switching from the current screen to the next one, which is dedicated to the process settings selection.



ImpostaBit
Variabile (Uscita/Ingresso) Enable_S0
AttivaPagina
Nome della pagina Pagina_selection
Numero dell'oggetto 0

Selection Page

The Selection page is the one that appears once the S0 state is set; here the process is initialized by choosing the settings needed to start the production process, such as the syrup, cream and topping types and the number of cakes to be produced with those settings.

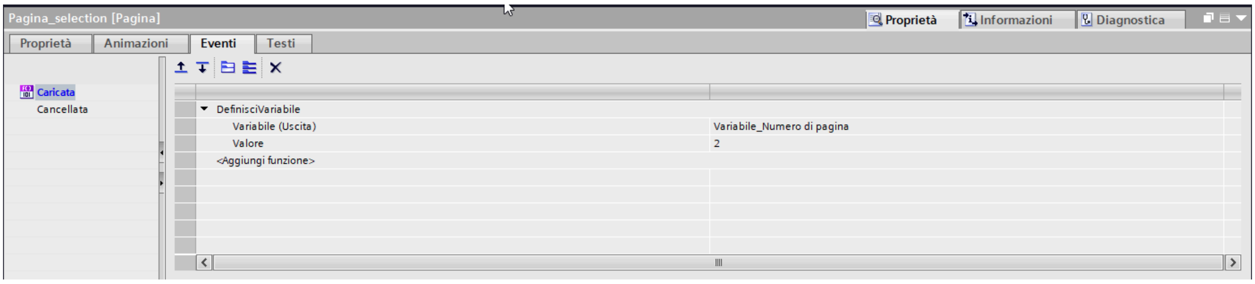


Here, several main events are defined to control this behavior:

1. an event associated with page load;
2. events associated with clicking the buttons to choose the cake's cream, syrup, topping and quantity;
3. an event associated with clicking the START button.

Event 1: Page load

When the Selection page loads, the ***Page Number*** variable is set to 2.
This operation records the current interface state, useful for tracking navigation across the different screens.



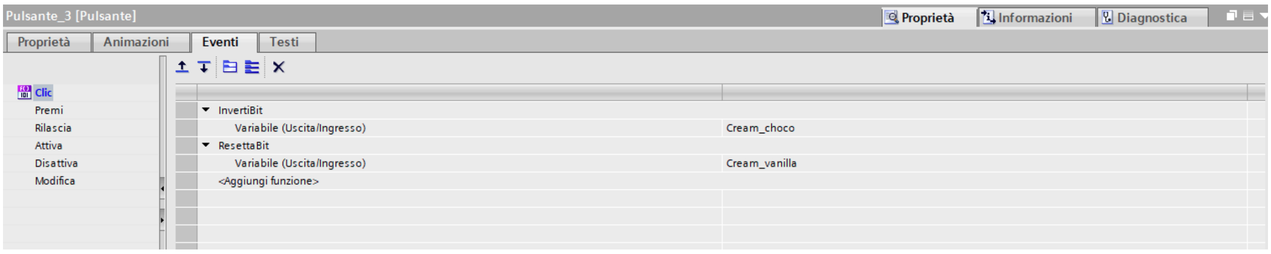
- DefinisciVariabile
  - Variabile (Uscita): Variabile_Numero di pagina
  - Valore: 2
- <Aggiungi funzione>

Event 2: Choosing cake production settings

When clicking the button dedicated to selecting the cream/syrup/topping, two actions are performed:

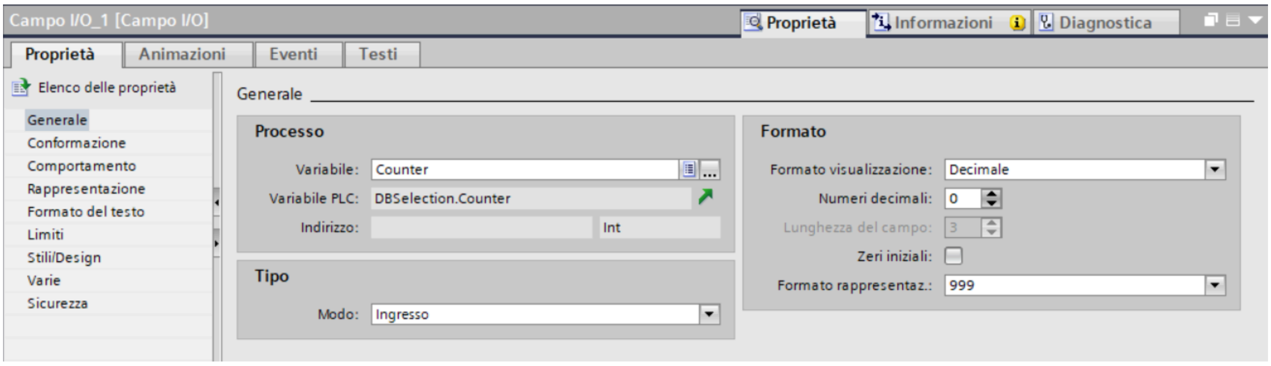
1. The corresponding variable is *toggled*, i.e., its state is inverted from 0 to 1 or vice versa.

2. The variable corresponding to the second option for that component (the unselected option) is *reset*, so that only one type of cream/syrup/topping can be selected at a time.



- InvertiBit
  - Variabile (Uscita/Ingresso): Cream_choco
- ResettaBit
  - Variabile (Uscita/Ingresso): Cream_vanilla
- <Aggiungi funzione>

Also, the number of cakes to be produced has to be specified by inserting the number into the corresponding I/O field. In particular, the *Quantity* field is mapped to the *Counter* variable.



Generale

Processo
Variabile: Counter
Variabile PLC: DBSelection.Counter
Indirizzo: Int

Tipo
Modo: Ingresso

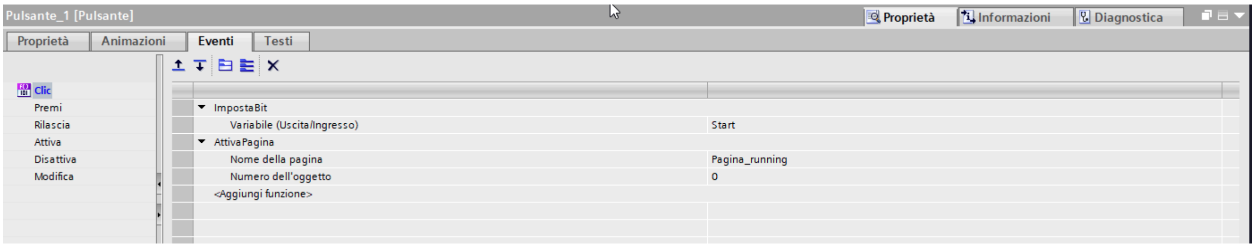
Formato
Formato visualizzazione: Decimale
Numeri decimali: 0
Lunghezza del campo: 3
Zeri iniziali: □
Formato rappresentaz.: 999

Event 3: Click the START button

When the START button is clicked, the following operations are executed:

1. The ***Start*** variable is set to start the process with the selected parameters.

2. The ***Running page*** is activated to switch to the screen where the process execution status can be monitored.

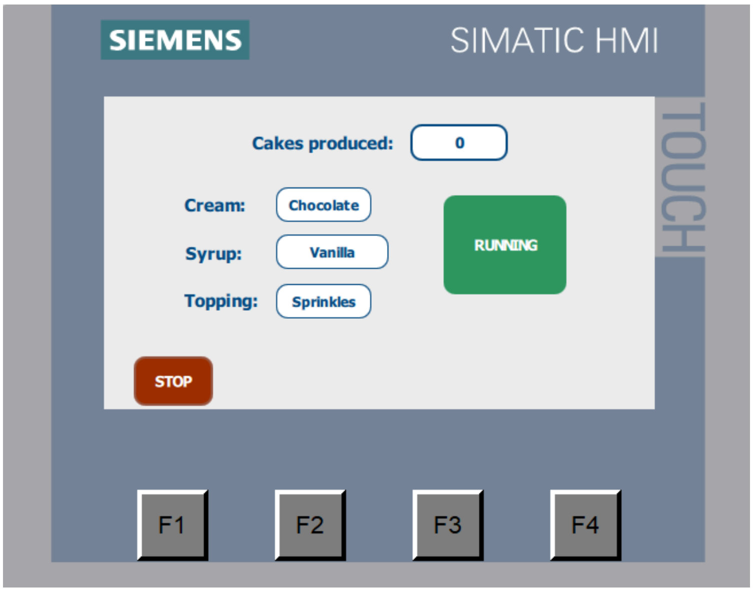


ImpostaBit	
Variabile (Uscita/Ingresso)	Start
AttivaPagina	
Nome della pagina	Pagina_running
Numero dell'oggetto	0
<Aggiungi funzione>	

Running Page

In the Running page the overall process can be monitored, stopped, resumed and reset. More specifically, the user can see the chosen settings for the current production process, and the number of produced cakes in real time.

If the process is running, the RUN button blinks, while if the process is paused by clicking on the STOP button, the process can be both resumed by clicking again on the RUN button, or reset by clicking on the corresponding RESET button, which appears only if the process is paused first.

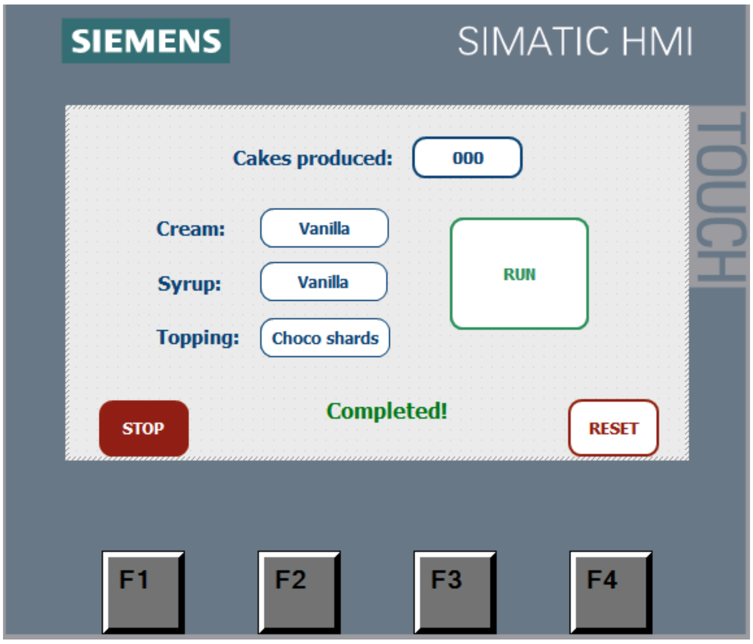


Cakes produced: 0

Cream: Chocolate

Syrup: Vanilla

Topping: Sprinkles



Cakes produced: 000

Cream: Vanilla

Syrup: Vanilla

Topping: Choco shards

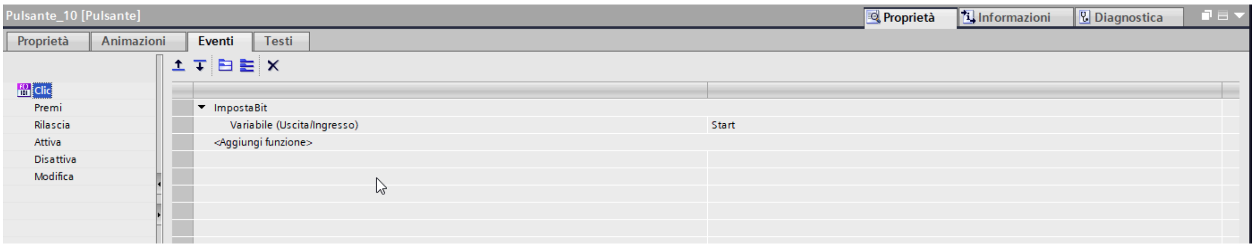
Completed!

To control this behaviour, three events are defined:

1. an event associated with clicking the STOP button;
2. an event associated with clicking the RUN button;
3. an event associated with clicking the RESET button.

Event 1: Click the STOP button

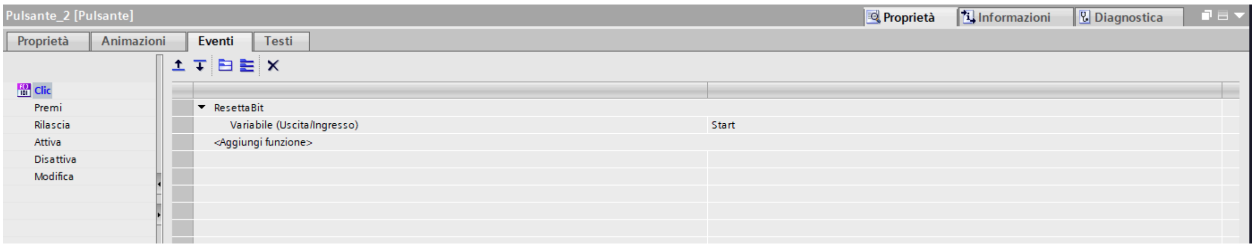
As the page is loaded, the *Start* variable related to the RUN button event is already set. But When the STOP button is clicked, the ***Start*** variable is reset, pausing the execution. The RUN button stops blinking, while the RESET button appears on the screen, giving the possibility to reset the process.



ImpostaBit
Variabile (Uscita/Ingresso) Start
<Aggiungi funzione>

Event 2: Click the RUN button

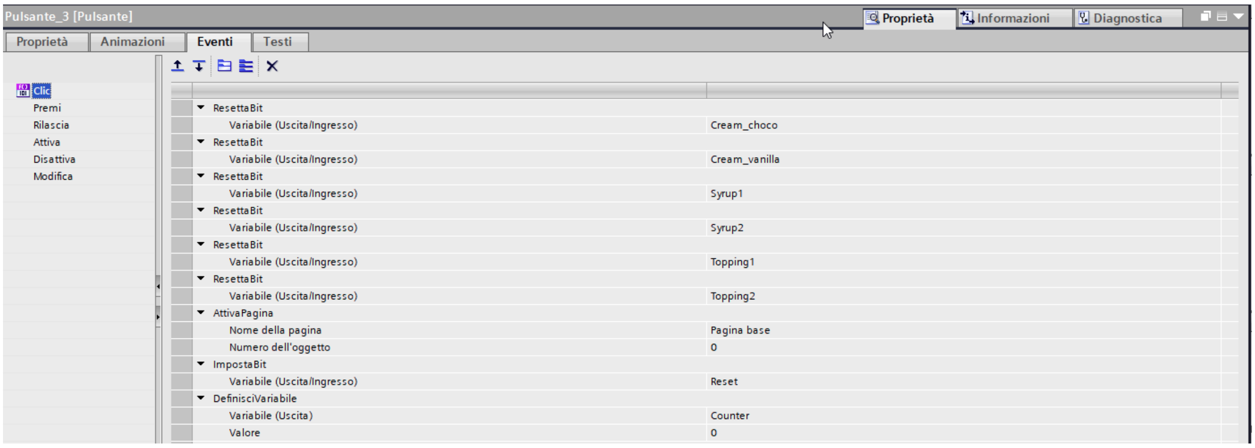
When the RUN button is clicked, the ***Start*** variable is set again, resuming the operating cycle continuing from the step at which it was interrupted. The RUN button starts to blink again, while the RESET button is hidden to prevent from accidentally resetting the process.



ResettaBit
Variabile (Uscita/Ingresso) Start
<Aggiungi funzione>

Event 3: Click the RESET button

When the RESET button is clicked, after pausing the execution, all variables related to process settings are reset. Finally, the ***Base Page*** is activated to return to the initial screen and enable a new execution of the production cycle.



ResettaBit	
Variabile (Uscita/Ingresso)	Cream_choco
ResettaBit	
Variabile (Uscita/Ingresso)	Cream_vanilla
ResettaBit	
Variabile (Uscita/Ingresso)	Syrup1
ResettaBit	
Variabile (Uscita/Ingresso)	Syrup2
ResettaBit	
Variabile (Uscita/Ingresso)	Topping1
ResettaBit	
Variabile (Uscita/Ingresso)	Topping2
AttivaPagina	
Nome della pagina	Pagina base
Numero dell'oggetto	0
ImpostaBit	
Variabile (Uscita/Ingresso)	Reset
DefinisciVariabile	
Variabile (Uscita)	Counter
Valore	0

Appendix

Complete Ladder Diagram

In the following pages, the complete Ladder diagram is shown as reported from TIA Portal.

CAKE_ASSEMBLY_def / PLC_1 [CPU 1212C AC/DC/Rly] / Blocchi di programma

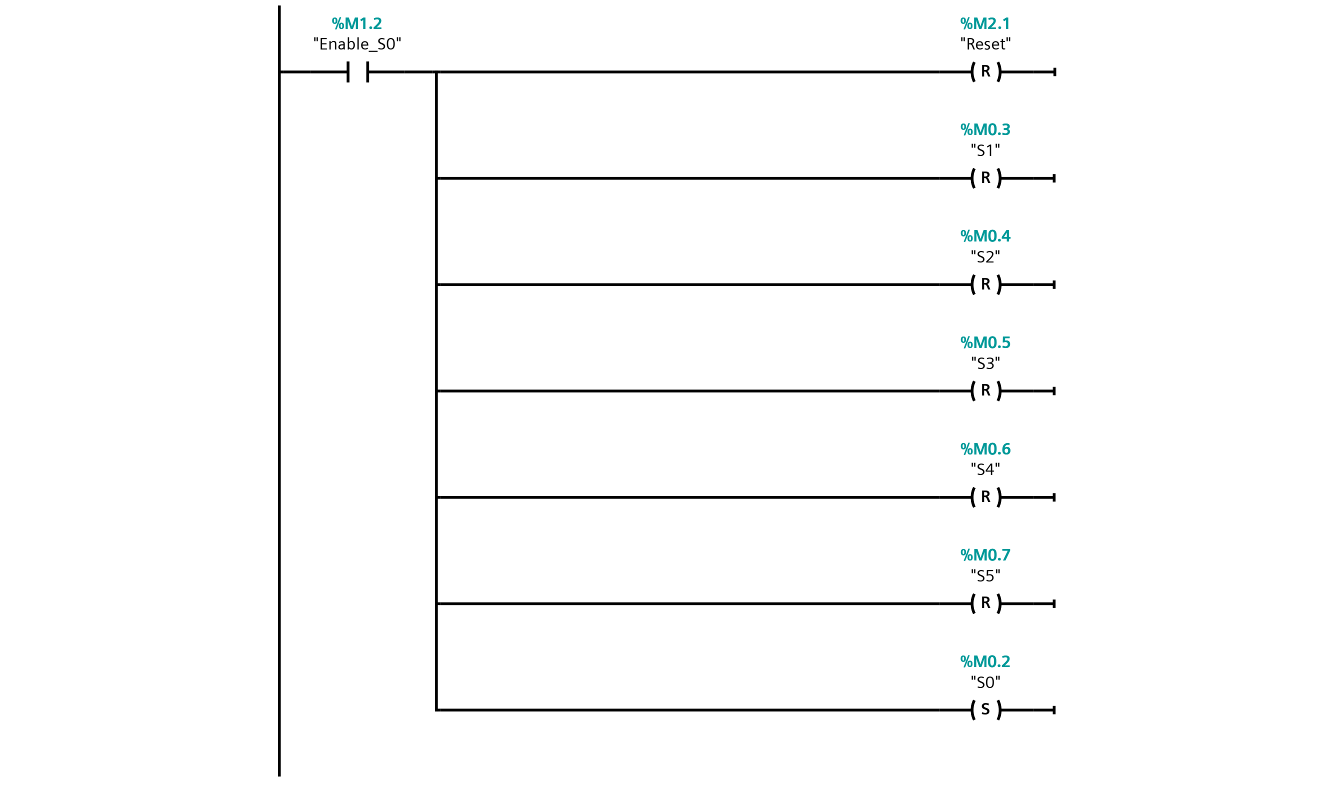
Main [OB1]

Main Proprietà					
Generale					
Nome	Main	**Numero**	1	**Tipo**	OB
Linguaggio	KOP	**Numerazione**	Automatico		
Informazioni					
Titolo	"Main Program Sweep (Cycle)"	**Autore**		**Commento**	
Famiglia		**Versione**	0.1	**ID definito dall'utente**	

Main			
Nome	**Tipo di dati**	**Valore di default**	**Commento**
▼ Input			
Initial_Call	Bool		Initial call of this OB
Remanence	Bool		=True, if remanent data are available
Temp			
Constant			

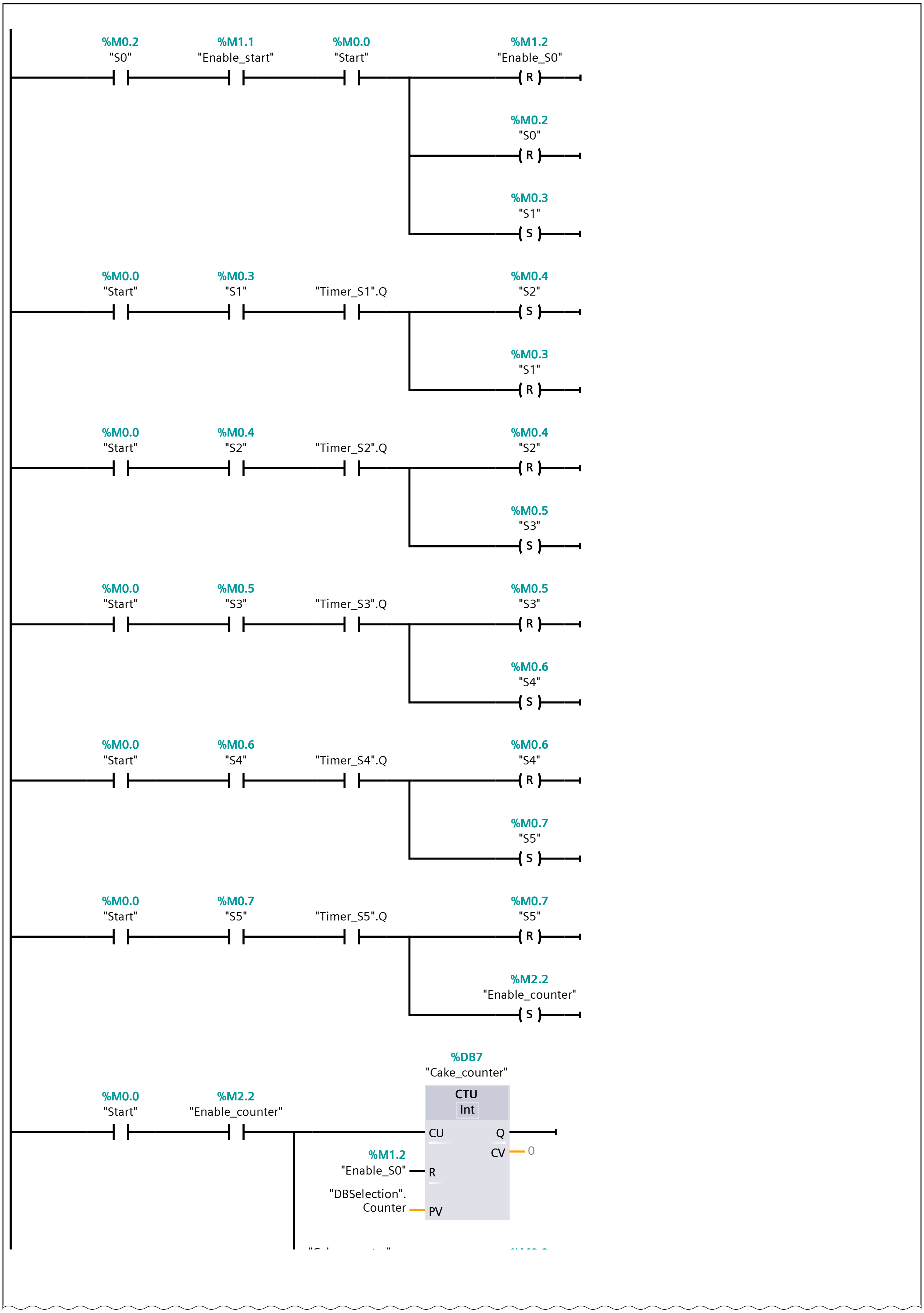
Segmento 1: System initialization

The system is initialized by setting the initial state S0 (which sets all the choices for the cakes to be made) and resetting all the other states.



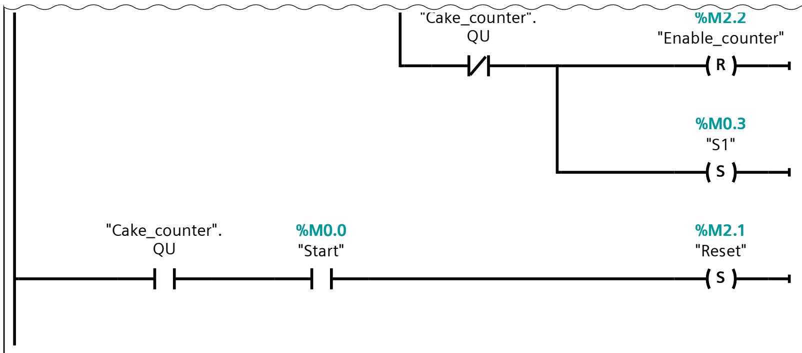
Segmento 2: State transitions

Segmento 2: State transitions (1.1 / 2.1)

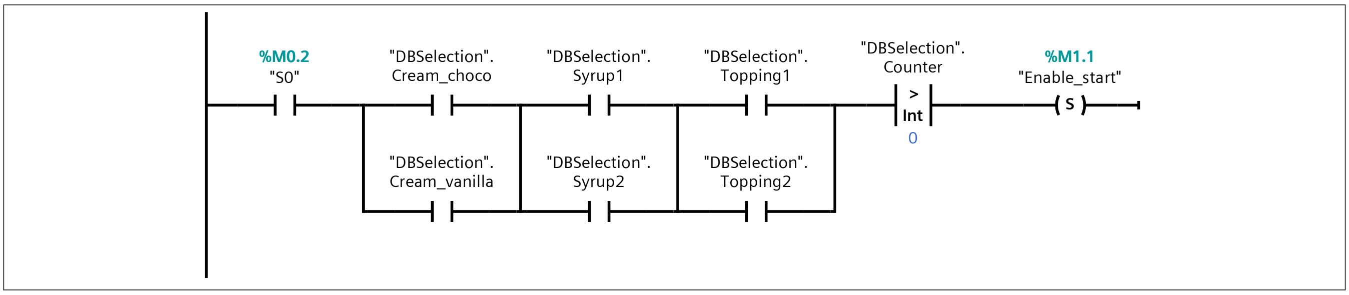


Segmento 2: State transitions (2.1 / 2.1)

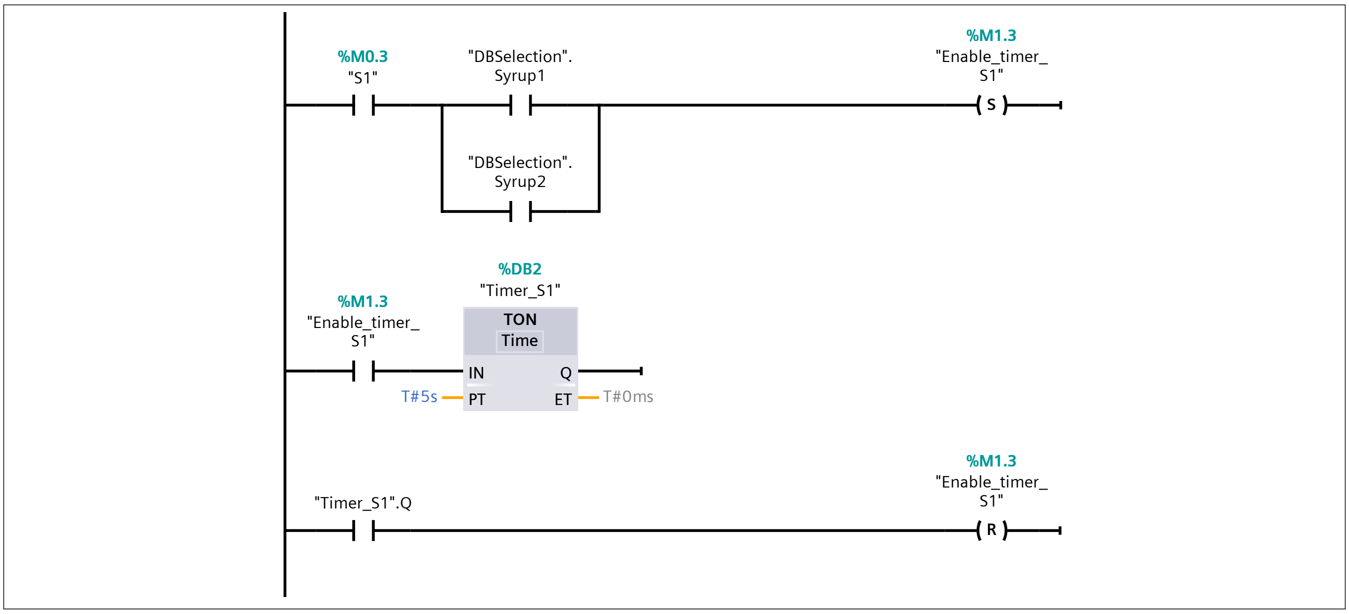
1.1 (Pagina1 - 2)



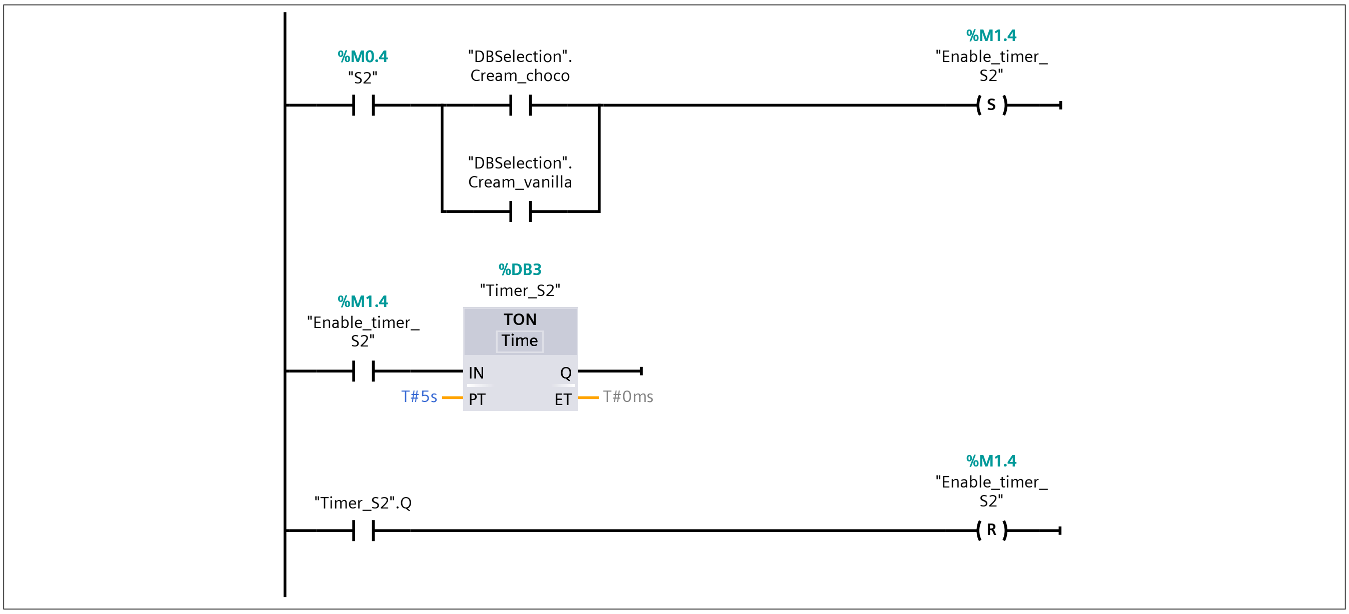
Segmento 3: S0 state functioning



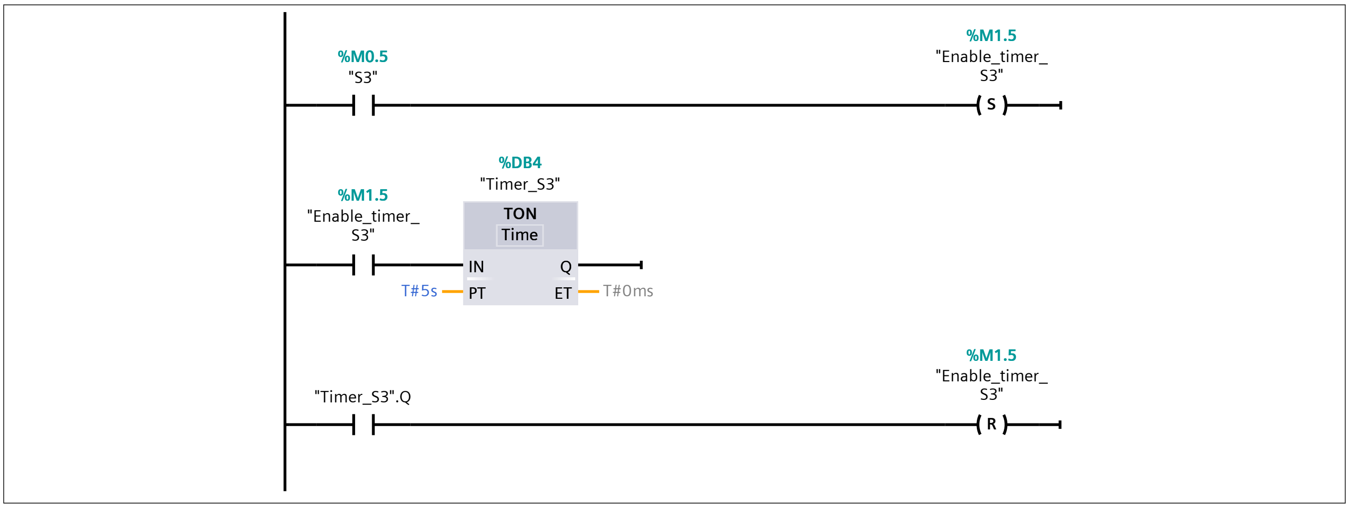
Segmento 4: S1 State commands



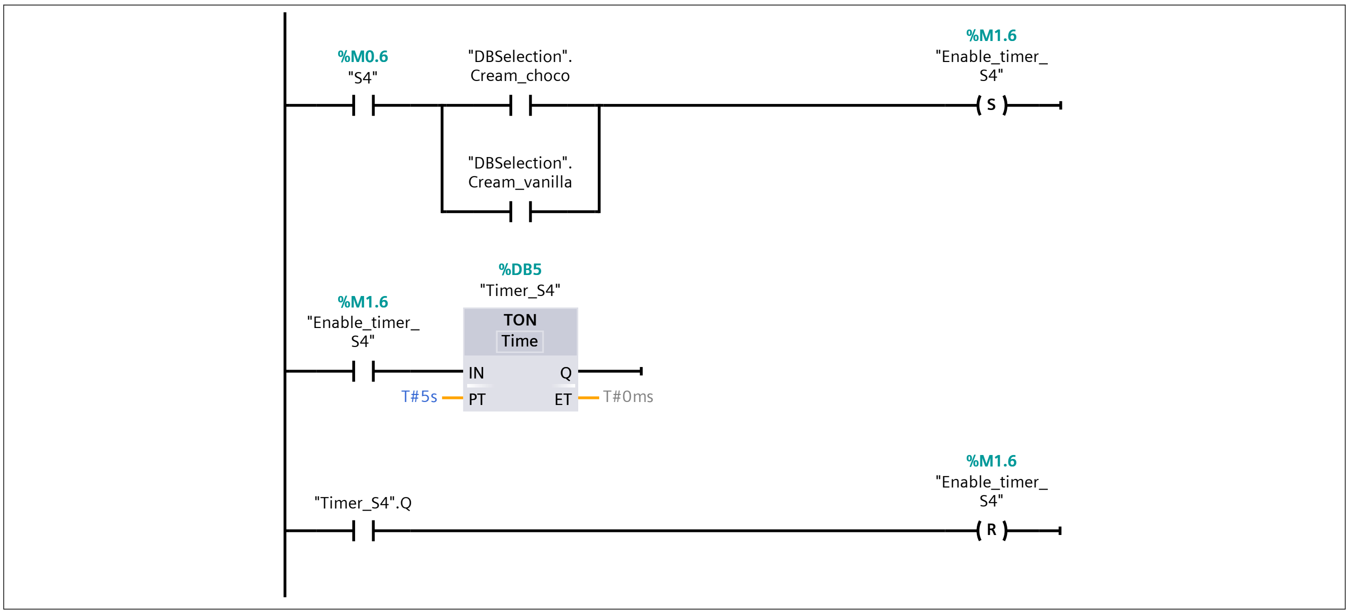
Segmento 5: S2 State commands



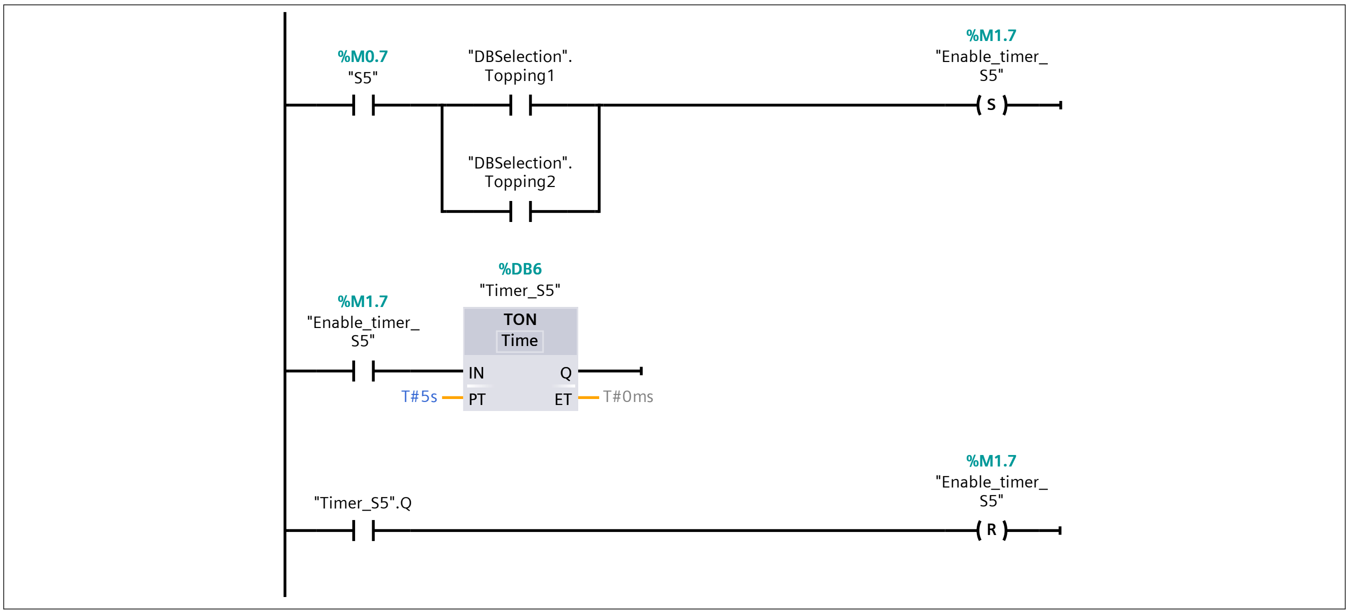
Segmento 6: S3 State commands



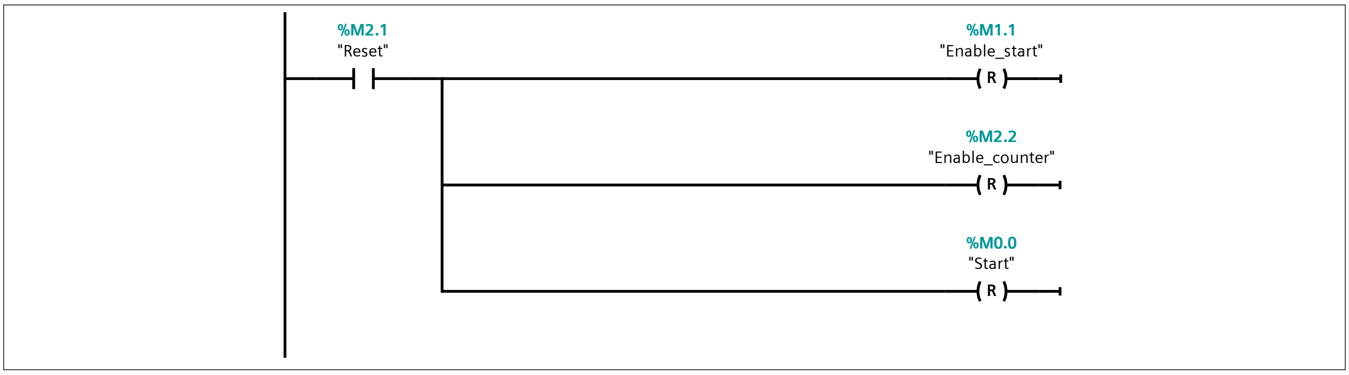
Segmento 7: S4 State commands



Segmento 8: S5 State commands



Segmento 9: Reset commands


%M2.1
"Reset"
%M1.1
"Enable_start"
R
%M2.2
"Enable_counter"
R
%M0.0
"Start"
R