Learn-/Training Document

Siemens Automation Cooperates with Education (SCE) | From NX MCD V12/TIA Portal V15.0

DigitalTwin@Education Module 150-002
Configuration of the Automation Program of a Dynamic 3D Model in the TIA Portal

siemens.com/sce

SIEMENS

Global Industry Partner of WorldSkills International

worldskills

Matching SCE trainer packages for this Learn-/Training Document

SIMATIC STEP 7 Software for Training (including PLCSIM Advanced)

- **SIMATIC STEP 7 Professional V15.0 - Single License**
  Order no.: 6ES7822-1AA05-4YA5
- **SIMATIC STEP 7 Professional V15.0 - Classroom License for 6 Users**
  Order no.: 6ES7822-1BA05-4YA5
- **SIMATIC STEP 7 Professional V15.0 - Upgrade License for 6 Users**
  Order no.: 6ES7822-1AA05-4YE5
- **SIMATIC STEP 7 Professional V15.0 - Student License for 20 Users**
  Order no.: 6ES7822-1AC05-4YA5

SIMATIC WinCC Engineering/Runtime Advanced software in the TIA Portal

- **SIMATIC WinCC Advanced V15.0 - Classroom License for 6 Users**
  6AV2102-0AA05-0AS5
- **Upgrade SIMATIC WinCC Advanced V15.0 - Classroom License for 6 Users**
  6AV2102-4AA05-0AS5
- **SIMATIC WinCC Advanced V15.0 - Student License for 20 Users**
  6AV2102-0AA05-0AS7

NX V12.0 Educational Bundle (Schools, universities, not for in-company training centers)

- **Contact**: academics.plm@siemens.com

Additional information regarding SCE

siemens.com/sce

Information regarding use

The SCE Learn-/Training Document for the integrated automation solution Totally Integrated Automation (TIA) was prepared for the program "Siemens Automation Cooperates with Education (SCE)" specifically for training purposes for public educational facilities and R&D institutions. Siemens does not guarantee the contents.

This document is only to be used for initial training on Siemens products/systems. This means it can be copied in whole or in part and given to trainees/students for use within the scope of their training/course of study. Disseminating or duplicating this document and sharing its content is permitted within public training and advanced training facilities for training purposes or as part of a course of study.

Exceptions require written consent from Siemens. Send all related requests to scesupportfinder.i-ia@siemens.com.

Offenders will be held liable. All rights including translation are reserved, particularly if a patent is granted or a utility model or design is registered.

Use for industrial customer courses is explicitly not permitted. We do not consent to commercial use of the Learn-/Training Document.

We wish to thank the HS Darmstadt, particularly Mr. Heiko Webert and Mr. Prof. Dr.-Ing. Stephan Simons, and all other persons involved for their support in the preparation of this SCE learning/training document.

Table of contents

List of figures

List of tables

Configuration of an Automation Program for a Dynamic 3D Model in the TIA Portal

1 Goal

The following pages show a detailed description of the dynamic 3D model from Module 1 "Virtual commissioning of a production plant using a 3D model". The in-depth explanation of a proposed solution for an automation program, both for the PLC and for the HMI, rounds off this module.

2 Requirement

The requirements from Module 1 are still required for editing this module.

You should have basic knowledge of the basics of **PLC programming in the TIA Portal**, the **SCL** programming language. An understanding of visualization from module **SCE_EN_042_ 201_WinCC Advanced with TP700 Comfort and SIMATIC S7-1500**" is assumed.

Since the PLC is simulated in this workshop using S7-PLCSIM Advanced, there are no hardware components for the controller in this module.

You should also have acquired the theoretical basics from the first module of the training series.

A thorough study of the first module of this training series is recommended in order to be familiar with the basic functionality of the dynamic 3D model.

3 Required hardware and software

The following components are required for this module:

1 **Engineering Station**: Requirements include hardware and operating system (for additional information: See Readme on the TIA Portal installation DVDs, as well as in the NX software package)

2 **SIMATIC STEP 7 Professional software in TIA Portal** – V15.0 or higher

3 **SIMATIC WinCC Runtime Advanced software in TIA Portal** – V15.0 or higher

4 **SIMATIC S7-PLCSIM Advanced software** – V2.0 or higher

5 **NX software with Mechatronics Concept Designer add-on** – V12.0 or higher

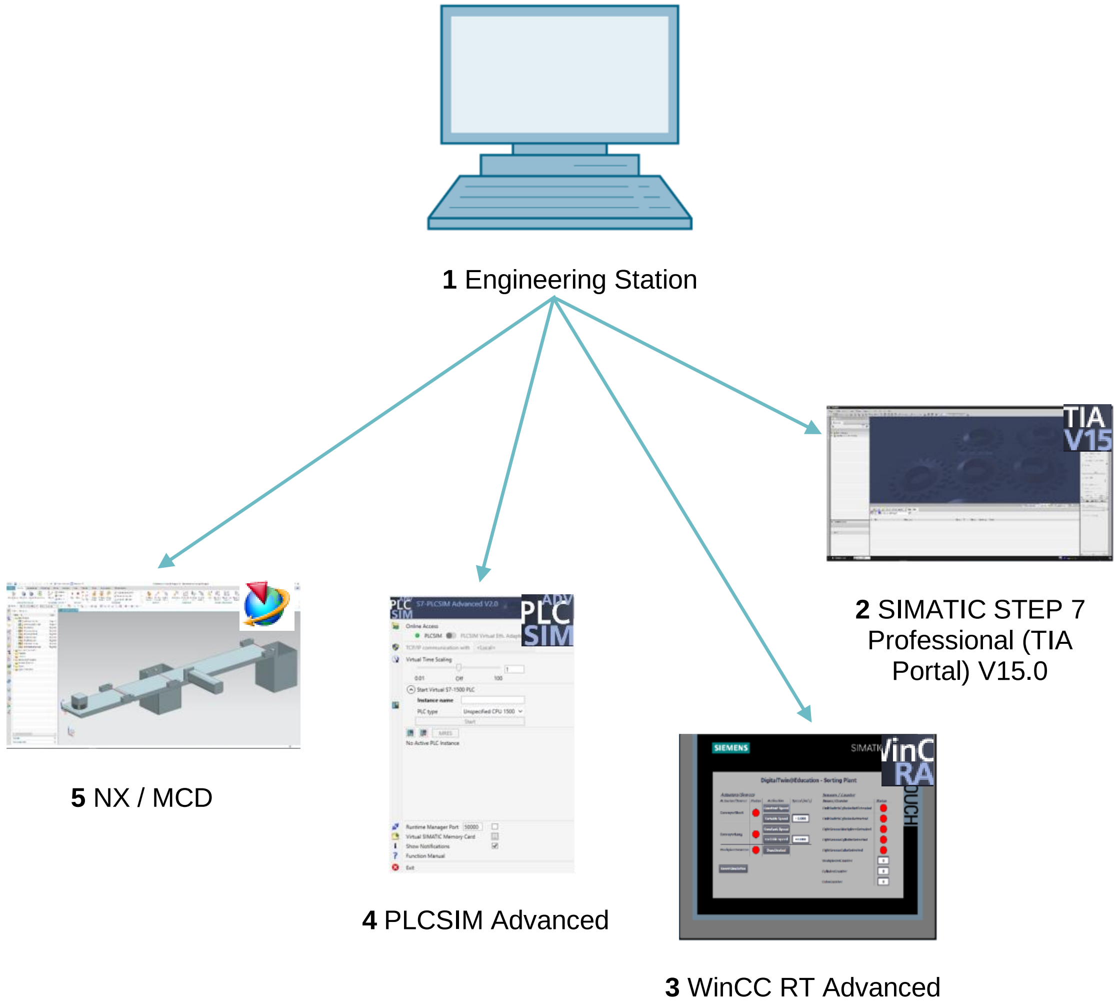


Figure 1: Overview of required software and hardware components in this module

Figure 1 clearly shows that the engineering station is the only hardware component of the system. The remaining components are based exclusively on software.

4 Theory

An introduction to the dynamic 3D model "SortingPlant" is already part of Module 1 of the DigitalTwin@Education Workshop series. However, this is limited to the information required for basic understanding and commissioning.

The 3D model is explained in detail in this chapter. This serves as a basis for creating an automation program, as described in Chapter 5 and Chapter 7.

The "SortingPlant" consists of:

- The workpieces to be sorted
- Two conveyor belts ("ConveyorShort" and "ConveyorLong")
- An ejector, which ejects the cylindrical workpieces
- A light sensor that detects all workpieces, regardless of their shape, before they leave the first conveyor belt ("Workpiece light sensor ")
- A light sensor that detects the cylindrical workpieces shortly before the ejector ("Cylinder light sensor ")
- A light sensor that detects the remaining cuboid workpieces at the end of the second conveyor belt ("Cube light sensor")

The individual components and their associated signals are presented below.

4.1 Workpieces

Workpieces can be generated during the simulation within the NX CAE add-on Mechatronics Concept Designer (MCD). Figure 2 shows the 3D model "SortingPlant". The source for generating the workpieces is highlighted in orange. There are two types of workpieces for this model:

- A cylindrical body, "Cylinder"
- A cuboid body, "Cube".

The "Cube" body is higher than the "Cylinder" body, which is highly relevant for the light sensors described in Chapter 4.5, Chapter 4.6 and Chapter 4.7.

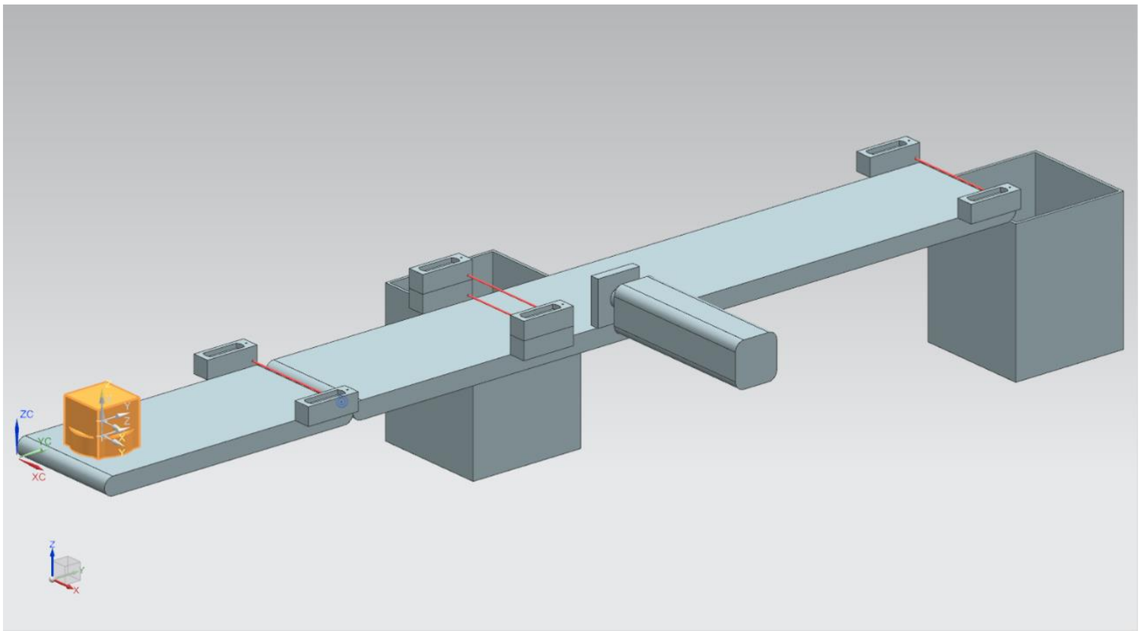


Figure 2: "SortingPlant" model with highlighted workpieces "Cylinder" and "Cube".

The workpieces are generated according to the following principle:

- A cylindrical body is generated at the start of the simulation and then every 10 seconds.
- The first cuboid body is generated after 5 seconds.
- Then, another cuboid body is generated every 10 seconds.
- The timing is controlled by counters within MCD.

A Boolean signal controls the activation and deactivation of the generation process for each type of workpiece. "**osWorkpieceCylinder_SetActive**" is responsible for the generation of cylindrical bodies and "**osWorkpieceCube_SetActive**" for the generation of cuboid bodies.

If the logical value "**1**" is assigned to one of these signals, the generation of the respective workpiece starts as described above. In addition, an internal, signal-specific counter counts up. With a logical value "**0**", this counter stops. This also means that no further workpiece of this type is produced. The internal counter retains its value, so that the last counter value is continued when the object source is activated again. The internal counter can only be reset within MCD.

NOTE

To reset the simulation in MCD, you must select "**Home**" menu in the menu bar (see Figure 3, step 1). Finally, select the "Restart" ⏮ icon in the NX MCD simulation control icons (see Figure 3, step 2).

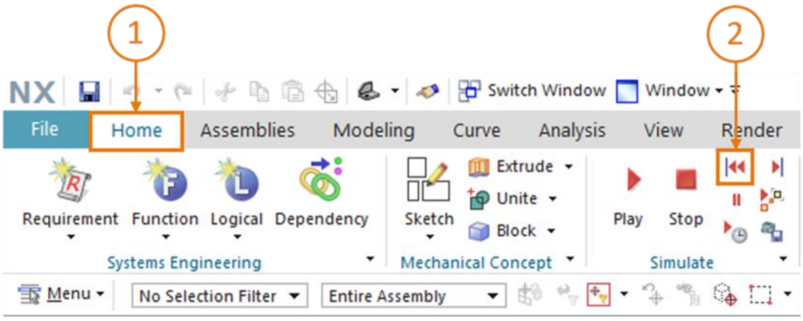


Figure 3: Restarting the NX MCD simulation

4.2 ConveyorShort

There are two different transport surfaces in the "SortingPlant" 3D model. In Figure 4, the first and shorter conveyor, "ConveyorShort", is highlighted. It can only move in one possible direction, which is also shown in Figure 4.

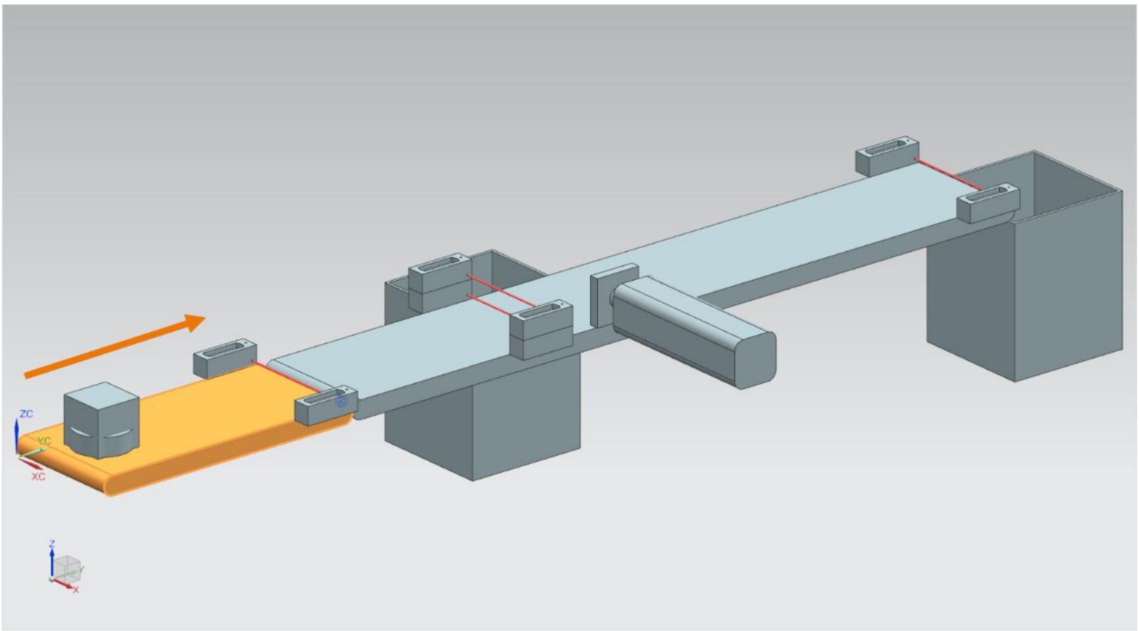


Figure 4: "SortingPlant" model with highlighted "ConveyorShort" conveyor belt and "Travel direction" (orange)

"ConveyorShort" serves as a transport element of the system, which introduces newly produced workpieces into the sorting process. The "Cylinder" and "Cube" bodies, which have already been explained in Chapter 4.1, are used as workpieces.

The conveyor belt moves either at a constant speed or a speed selected by the user. This is solved in MCD by two different transport speed controllers. A controller for constant speed and another controller for variable speed are available for this purpose.

In the dynamic 3D model, three signals are defined for this conveyor belt:

- ***scConveyorShortConstSpeed_SetActive*** is a Boolean signal that activates or deactivates the constant speed controller. A constant speed of 0.05 m/s has been set in MCD.
- The Boolean signal ***scConveyorShortVarSpeed_SetActive*** activates or deactivates the controller for variable speed.
- The variable speed ***scConveyorShortVarSpeed_SetSpeed*** is a signal in floating point value format that specifies a speed for the system in the unit m/s. This is only considered if *scConveyorShortVarSpeed_SetActive* is activated.

4.3 ConveyorLong

The second transport surface of the 3D model, "ConveyorLong", is shown in Figure 5. As already described for "ConveyorShort" in Chapter 4.2, this conveyor also moves in only one direction.

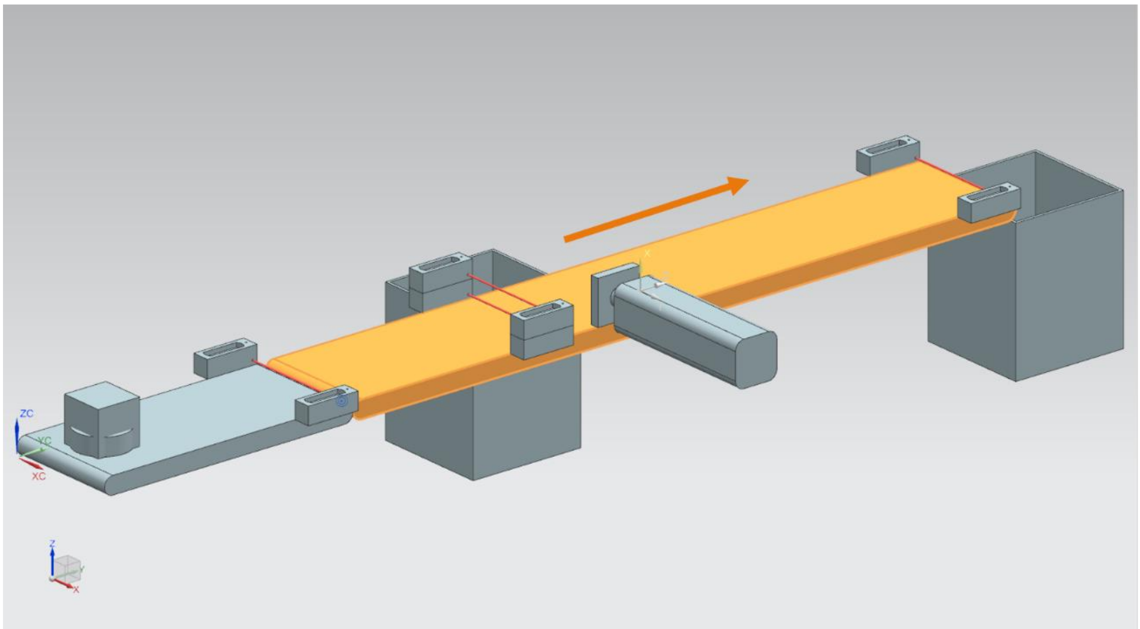


Figure 5: "SortingPlant" model with highlighted "ConveyorLong" conveyor belt and travel direction (orange)

The longer "ConveyorLong" conveyor belt transports the workpieces as a central component of the sorting process. During transport, cylindrical workpieces are sorted out into a container using an ejector (see Chapter 4.4). The cuboid workpieces travel to the end of the belt, where they fall into another container.

This conveyor belt can also be moved at a constant speed or a user-definable variable speed. Two controllers are also available in MCD for this.

As for the shorter conveyor belt (see Chapter 4.2), there are three defined signals for "ConveyorLong" in the dynamic 3D model:

- ***scConveyorLongConstSpeed_SetActive***, to activate or deactivate the controller for constant belt speed. Here, too, a speed of 0.05 m/s has been defined in the MCD model.
- ***scConveyorLongVarSpeed_SetActive***, to activate or deactivate the controller for variable belt speed.
- ***scConveyorLongVarSpeed_SetSpeed***, as default of the variable speed in m/s in floating point value format.

4.4 Cylinder ejector

As already mentioned in Chapter 4.3, cylindrical bodies are sorted out by an ejector, which is shown in orange in Figure 6.

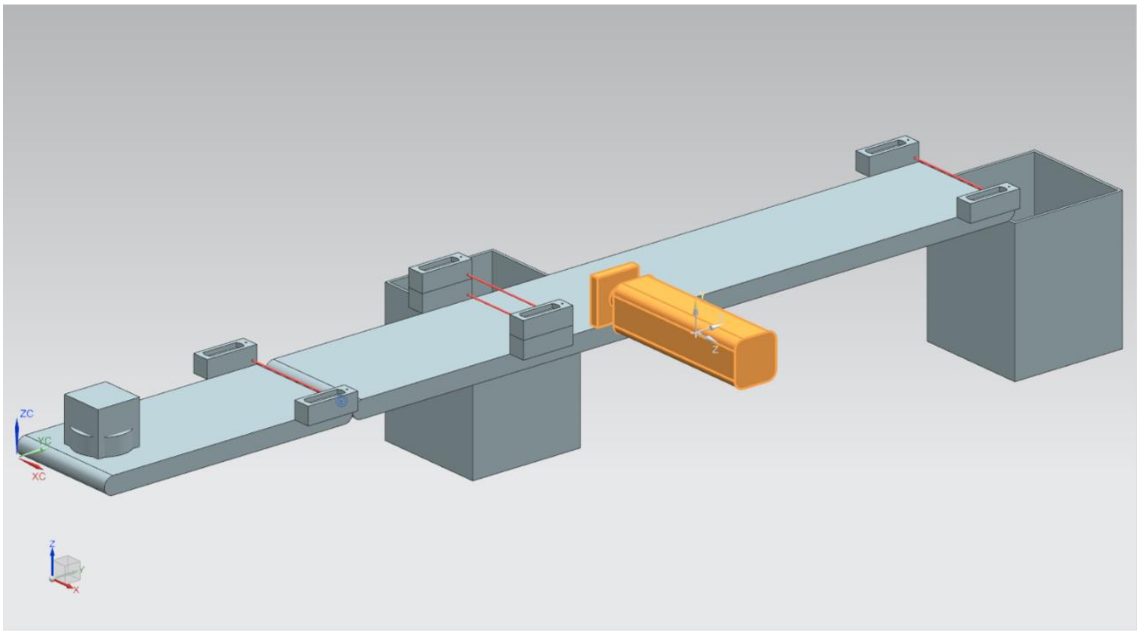


Figure 6: "SortingPlant" model with highlighted ejector

The "Cylinder" ejector is responsible for ejecting the "Cylinder" type workpieces from the "ConveyorLong" conveyor belt. As shown in Figure 7, the ejection arm can be extended and retracted.

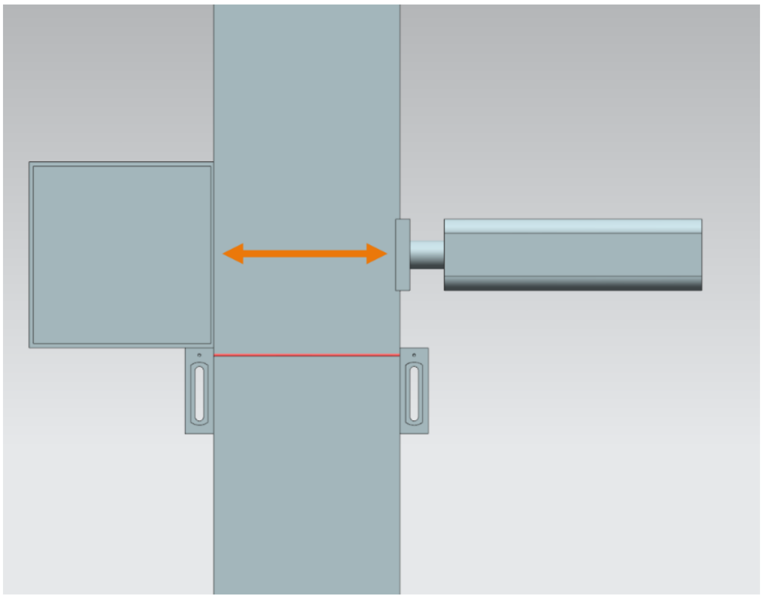


Figure 7: Travel direction of the ejector (orange)

The ejector is to act as a bidirectional actuator, i.e. there is one signal each for the extension and retraction of the ejector arm. Two sensors detect whether the cylinder is fully extended or fully retracted.

This leads to the following signals:

- ***pcCylinderHeadExtend_SetActive***: With a value of logical "**1**" at this signal, the ejector arm extends as far the limit position.
- ***pcCylinderHeadRetract_SetActive***: Setting this signal to logical "**1**" retracts the ejector arm as far as the limit position.
- ***csLimitSwitchCylinderNotExtended***: This Boolean signal indicates whether the ejector arm has not yet been fully extended. This signal is not set to logical value "**0**" until it has been fully extended, otherwise the signal returns the logical value "**1**".
- ***csLimitSwitchCylinderRetracted***: This Boolean signal indicates whether the ejector arm has been completely retracted. This state is indicated by logical value "**1**", otherwise logical value "**0**" is set.

4.5 Workpiece light sensor

Figure 8 shows the "Workpiece" light sensor highlighted. In the 3D model, it consists of a sensor head with counterpart and the light beam. This light sensor is responsible for detecting any workpiece in the process at the end of the "ConveyorShort".

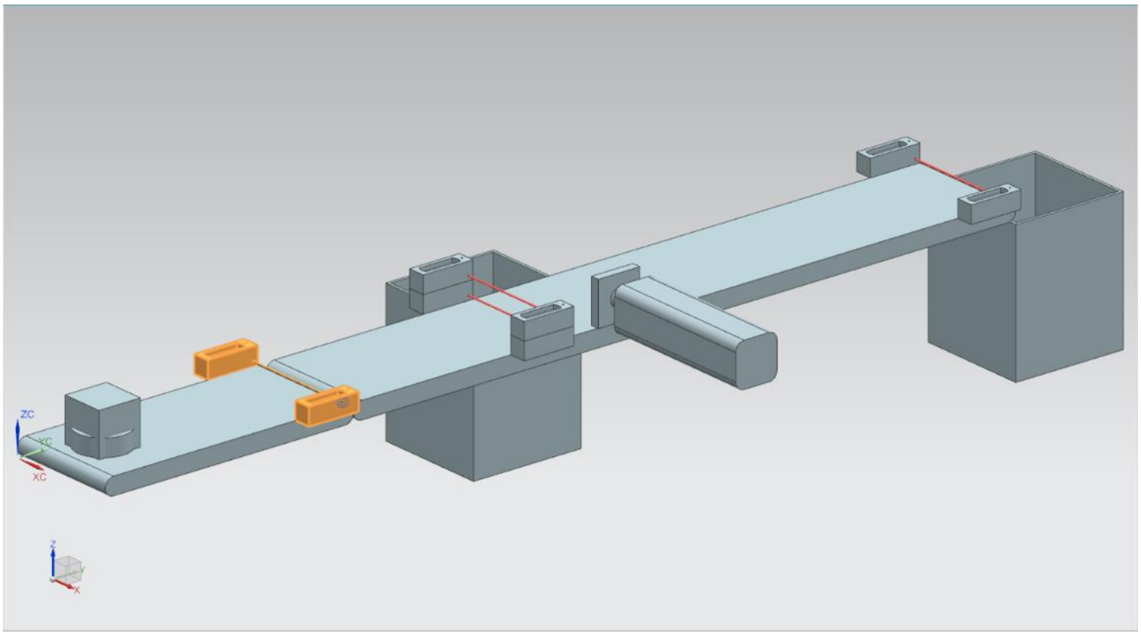


Figure 8: "SortingPlant" model with highlighted "Workpiece" light sensor

The following Boolean signal is assigned to the workpiece light sensor:

csLightSensorWorkpieces_Detected.

If any workpiece breaks through the light beam, the sensor is triggered. This sets the value of the signal to logical "**1**". Otherwise, the light sensor returns logical value "**0**".

4.6 Cylinder light sensor

A system of two light sensors has been implemented in the 3D model to recognize cylindrical bodies. As shown in Figure 9, these two light sensors are arranged one above the other.

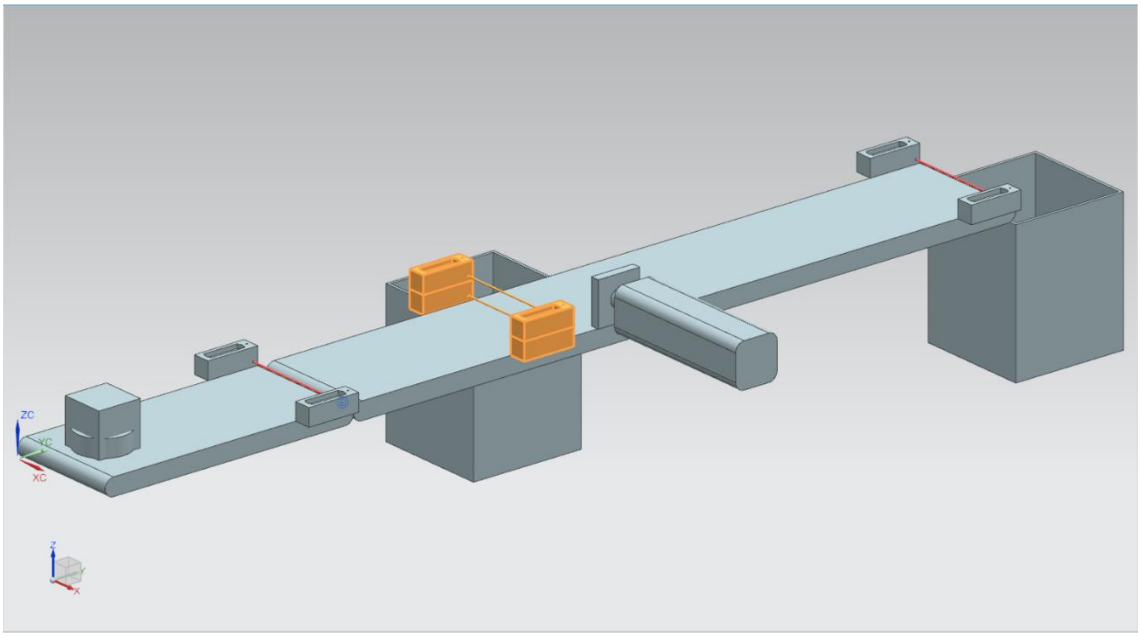


Figure 9: "SortingPlant" model with highlighted "Cylinder" light sensor system

Figure 10 shows how the two light sensors trigger for the various workpieces:

- In the case of the "Cube" body, both light sensors trigger because both light beams are interrupted.
- In the case of the smaller "Cylinder" body, because it has a different size than the "Cube", only the lower light beam is interrupted and thus only the lower light sensor is triggered.

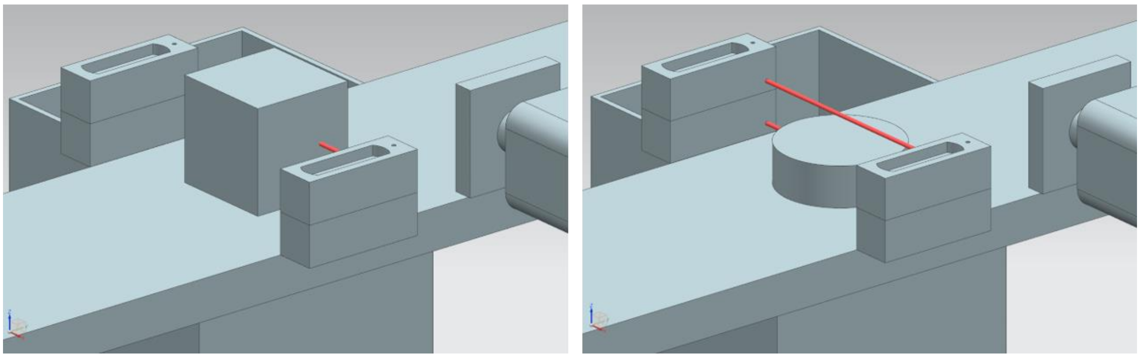


Figure 10: Triggering the light sensors: Comparison of "Cube" body (left) and "Cylinder" body (right)

- If no body interrupts the light rays, neither of the two light sensors is triggered.
- The situation in which only the upper light sensor triggers is only possible if the upper light sensor were defective and were to trigger continuously.

A "cylinder" is thus detected if the lower light sensor triggers but the upper one does not. This logic has been implemented in the 3D model in MCD itself.

The result is assigned to the Boolean signal ***csLightSensorCylinder_Detected***.

The logical value "**1**" describes that a cylindrical body has been detected by the light sensor system. Otherwise, the signal assumes the logical value "**0**".

4.7 Cube light sensor

The last light sensor of the 3D model is highlighted in Figure 11.

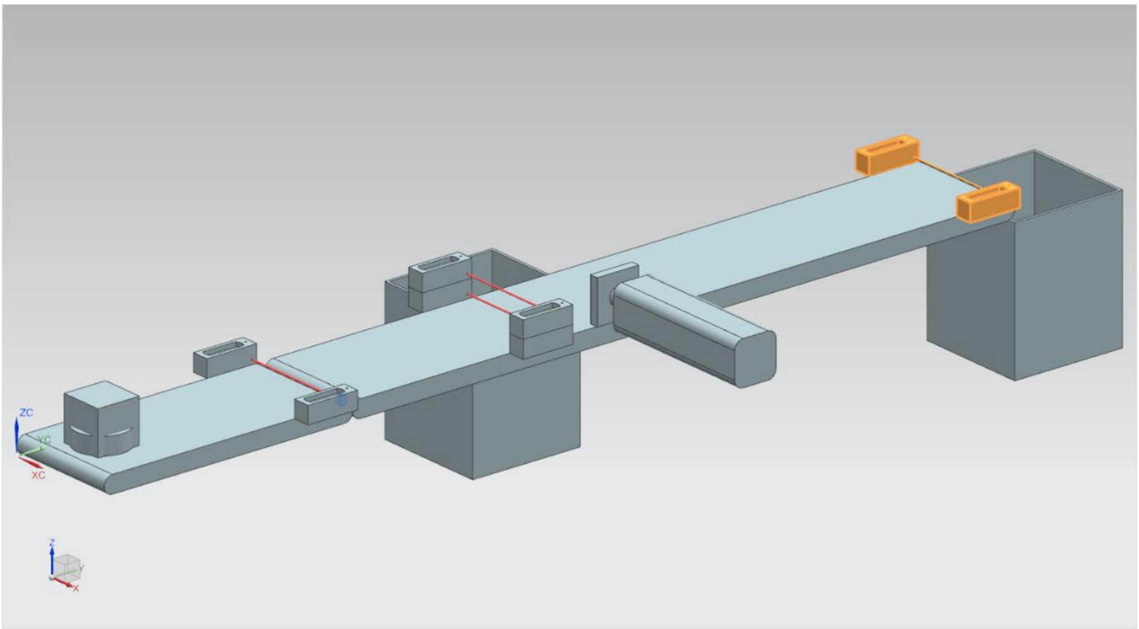


Figure 11: "SortingPlant" model with highlighted "Cube" light sensor

In comparison to the detection of cylindrical bodies, as described in Chapter 4.6, only a single light sensor is used here, since all cylindrical bodies should be sorted out using the ejector from Chapter 4.4. This leaves only the "Cube" workpieces to trigger the light sensor.

Triggering of the light sensor is described with the boolean signal ***csLightSensor Cube_Detected***.

When a body is detected, the signal will assume the logical value "**1**". Otherwise, the signal remains at logical value "**0**".

5 Task

After the more detailed description of the individual components with the associated signals in Chapter 4, this chapter describes the requirements for the automation program of the PLC and the visualization by the HMI.

5.1 Generating new workpieces

The generation process has already been explained in Chapter 4.1. However, the generation of new workpieces is linked to two conditions:

1. A new workpiece may only be generated if the corresponding signal has been set in the PLC by the HMI. For the cylindrical bodies, this is the signal **osWorkpieceCylinder_SetActive** and for the cuboid bodies the signal **osWorkpieceCube_SetActive**. Both signals are to be controlled simultaneously by a single input element in the HMI.
2. A new workpiece must not be generated when the ejector is in the process of sorting out a cylindrical workpiece, otherwise it could lead to a jam on the transport surfaces.

5.2 Controlling the transport surfaces

As already described in Chapter 4.2 and Chapter 4.3, the transport surfaces can be moved either at constant or variable speed. Note that only one of the two controllers, constant or variable, may be active at the same time. Otherwise, verifiable behavior would no longer be guaranteed. Mutual interlocking of the controllers must be implemented by the automation program.

When constant speed is activated, the current value for the variable speed must be kept at ZERO. The user may transfer the specified value in the HMI for the specified variable speed only after deactivating constant speed and activating variable speed.

The variable speed should be limited to a maximum of 0.15 m/s in the automation program.

The two "ConveyorShort" and "ConveyorLong" transport surfaces are to be able to be operated independently of each other.

5.3 Ejecting cylinder workpieces

If a cylindrical body is detected, as explained in Chapter 4.4, it must be transported slightly further before it can be ejected. This is due to the distance between the light beam of the sensor and the cylinder axis of the ejector. Assuming that the automation program detects the cylindrical workpiece when it leaves the light beam (negative edge of the light sensor system signal), this distance is 20 mm, as shown in Figure 12.

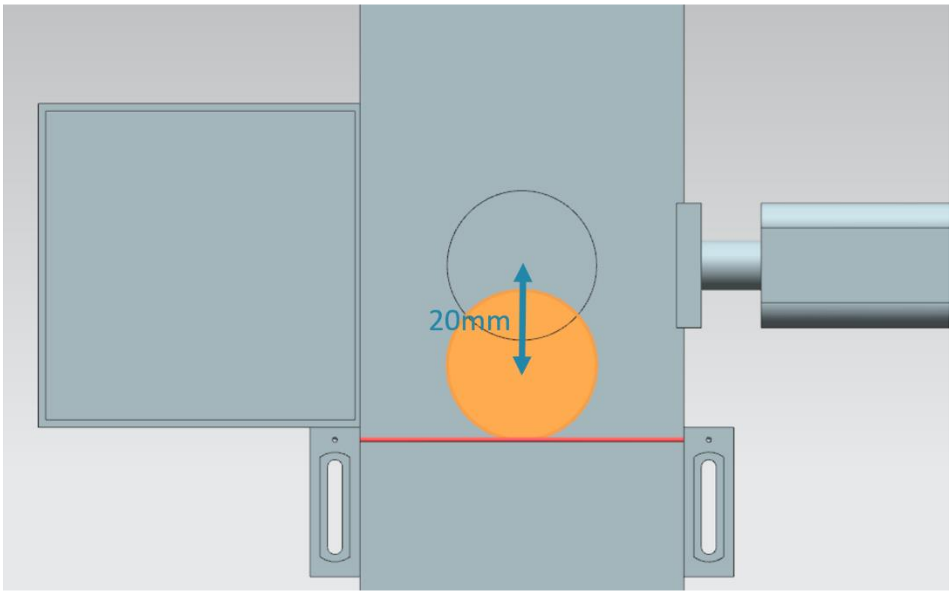


Figure 12: Distance of a cylindrical workpiece from the triggering of the sensor to the ejector arm.

Based on the specified speed of the "ConveyorLong" conveyor belt, a corresponding waiting time until ejection must be calculated during which the workpiece is transported to the center of the ejector. The waiting time must be implemented in the automation program.

After the waiting time, it is important to interrupt the running operation to prevent a backup on the conveyor belts. For this reason, you should deactivate the generation of new workpieces (see Chapter 5.1) and stop both conveyor belts at this time. Activating the object generation and the conveyor belts should then be locked. Only then can the ejection process begin.

Make sure that the ejector arm extends completely at first. This is required to ensure that the sorting of the cylindrical workpieces is successful.

The ejector arm should then be completely retracted again.

Finally, remove the blocking of the workpiece generation and transport surfaces and restore their state before the ejecting operation.

5.4 Counting the workpieces

To monitor the simulation process, the workpieces are to be counted during a simulation run. The current counter values are to be displayed in the HMI. It is recommended that the sensor signals of the light sensors be used for this purpose (see Chapter 4.5, Chapter 4.6 and Chapter 4.7).

5.5 Resetting the simulation data

It is also possible to reset the output signals in the automation program. This allows the simulation in MCD and the automation program to synchronize again before the simulation is restarted in the event of an interruption. This is required so that the MCD model always starts with the initial value when the simulation is started. The reset of the signals in the automation program is to be controlled via the HMI.

The reset should affect all output data of the automation program, i.e.:

- Generation of new workpieces
- Control of the conveyor belts
- Control of the ejector
- Counters for workpieces

It should be noted that resetting the simulation only applies to the PLC program. The simulation in the digital twin in MCD must be reset similar to Module 1 of the DigitalTwin@Education Workshop series.

You should now use this information to create a corresponding automation program with HMI. A possible solution is discussed in Chapter 7.

6 Planning

The detailed explanation of the automation program refers to the ready-made project "**150-001_DigitalTwinAtEducation_TIAP_Basic**", which was provided with Module 1. However, you are encouraged to use the discussion in Chapter 7 as a basis for creating your own approach to a solution.

The PLC program and HMI discussed were implemented with the **SIMATIC STEP 7 Professional V15.0** software. The **SIMATIC S7-PLCSIM Advanced V2.0** software is used to simulate a virtual PLC. The HMI is simulated with the optional package **SIMATIC WinCC Runtime Advanced V15.0** of the TIA Portal. Simulated Ethernet interfaces connect the virtual PLC to the virtual HMI.

To test your solution for functionality, it is recommended to once again use **Mechatronics Concept Designer V12.0**. You must also maintain the appropriately configured signals in your solution, otherwise the inputs and outputs will not be connected to each other. You can use the MCD model "**150-001_DigitalTwinAtEducation_MCD_dynModel_Signals**" from Module 1 again for this.

7 Structured step-by-step instructions

This chapter discusses the proposed solution of the automation program from Module 1 of the DigitalTwin@Education Workshop series. The discussion covers both the PLC program and the HMI design and its connection to the PLC.

Activity diagrams and state machines according to the Unified Modeling Language (UML) standard were used to represent processes. You can find further information on this under the links [3], [4] and [5] in Chapter 9.

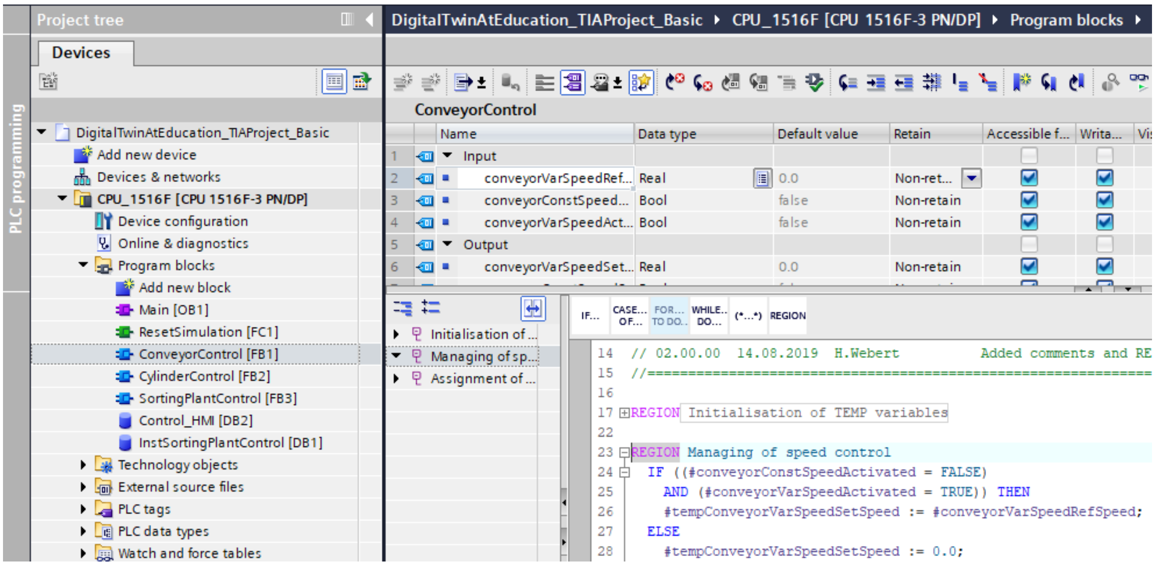
7.1 PLC program

7.1.1 General information about the PLC program

The automation program is based on the Programming Guideline for S7-1200/1500 [1] and the Guide to Standardization [2]. Note in particular:

- All identifiers use the camelCasing notation (all words are written together and begin with a capital letter). The input and output signals of the automation program are exceptions, since these signal designations should not deviate from those in the MCD model.
- The function blocks (FBs) and functions (FCs) have been created in the SCL programming language.
- Output tags are written only once per cycle. As a result, there is a temporary tag for each output signal. According to the programming guide, all temporary tags are initialized at the beginning of the FBs and FCs. During processing, only the temporary tags are accessed. At the end of each FB and FC, if applicable the temporary tag is assigned to the associated output in each case.

7.1.2 Structure of the TIA project



PLC programming

Project tree

Devices

- DigitalTwinAtEducation_TIAProject_Basic
  - Add new device
  - Devices & networks
  - CPU_1516F [CPU 1516F-3 PN/DP]
    - Device configuration
    - Online & diagnostics
    - Program blocks
      - Add new block
      - Main [OB1]
      - ResetSimulation [FC1]
      - ConveyorControl [FB1]
      - CylinderControl [FB2]
      - SortingPlantControl [FB3]
      - Control_HMI [DB2]
      - InstSortingPlantControl [DB1]
    - Technology objects
    - External source files
    - PLC tags
    - PLC data types
    - Watch and force tables

DigitalTwinAtEducation_TIAProject_Basic ▸ CPU_1516F [CPU 1516F-3 PN/DP] ▸ Program blocks ▸

ConveyorControl

	Name	Data type	Default value	Retain	Accessible f...	Writa...	Vi
1	▾ Input				□	□	
2	conveyorVarSpeedRef...	Real	0.0	Non-ret...	☑	☑	
3	conveyorConstSpeed...	Bool	false	Non-retain	☑	☑	
4	conveyorVarSpeedAct...	Bool	false	Non-retain	☑	☑	
5	▾ Output				□	□	
6	conveyorVarSpeedSet...	Real	0.0	Non-retain	☑	☑	

- Initialisation of ...
- Managing of sp...
- Assignment of ...

```
// 02.00.00  14.08.2019  H.Webert          Added comments and RE
//==================================================================

REGION Initialisation of TEMP variables

REGION Managing of speed control
  IF ((#conveyorConstSpeedActivated = FALSE)
    AND (#conveyorVarSpeedActivated = TRUE)) THEN
    #tempConveyorVarSpeedSetSpeed := #conveyorVarSpeedRefSpeed;
  ELSE
    #tempConveyorVarSpeedSetSpeed := 0.0;
```

Figure 13: Structure of the TIA project

The project structure in the TIA Portal for controlling the dynamic 3D model can be seen in Figure 13. Several program blocks have been created to solve this task.

- ConveyorControl: a function block for controlling a conveyor belt
- CylinderControl: a function module for controlling the ejector
- SortingPlantControl: a function module for mapping the functionality of the 3D model "SortingPlant"
- ResetSimulation: a function for resetting the output signals
- Control_HMI: a data block for data exchange between the PLC and the HMI
- As well as the Main (OB1) for the organized call of the FBs and FCs

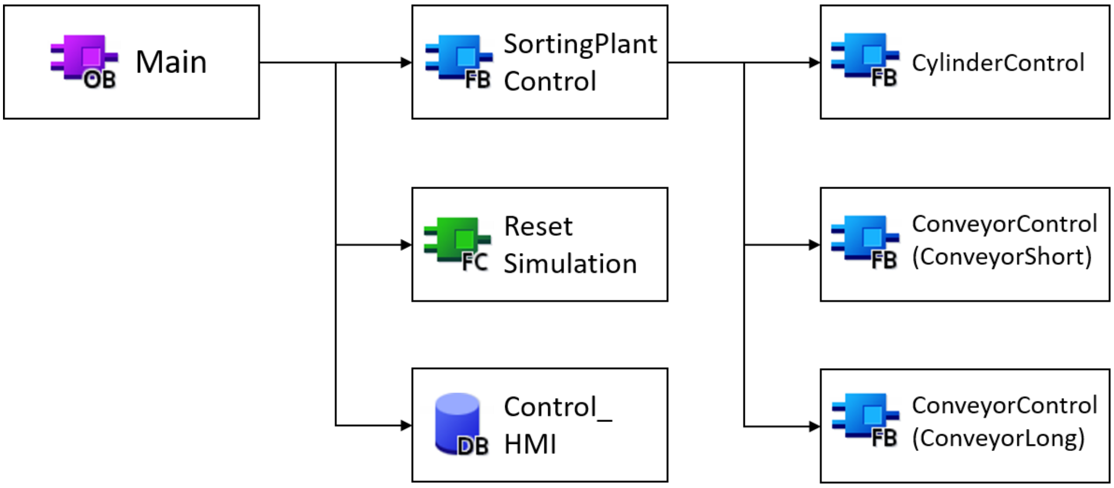


Figure 14: Call diagram of the TIA project

These blocks are called as shown in Figure 14 and explained in detail below.

7.1.3 FB ConveyorControl

The control of a conveyor belt was implemented in a function block.

Figure 15 shows the activity diagram of the FB "ConveyorControl".

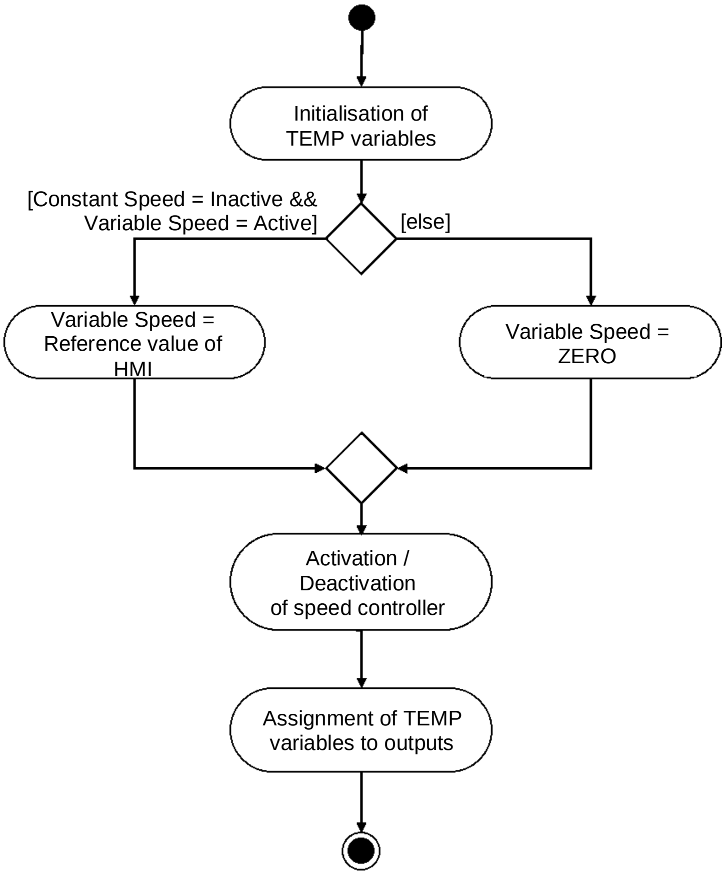

Initialisation of TEMP variables
[Constant Speed = Inactive && Variable Speed = Active]
[else]
Variable Speed = Reference value of HMI
Variable Speed = ZERO
Activation / Deactivation of speed controller
Assignment of TEMP variables to outputs

Figure 15: Activity diagram of the FB "ConveyorControl"

The FB starts initializing the temporary tags according to the programming guideline [1].

As described in Chapter 5.2, the variable speed should be ZERO when the conveyor moves at a constant speed. Otherwise, the speed should be taken from the setpoint set via the HMI.

The current status of the two speed controllers is determined by the two activity signals in the HMI (see Chapter 7.2.1).

The FB ends with the assignment of the temporary tags to the outputs.

The situation in which both the controller for the constant speed and the controller for the variable speed have been activated, has been intercepted accordingly by the HMI and will therefore not occur within this FB.

7.1.4 FB CylinderControl

The ejector is controlled by its own function block.

The activity diagram for this FB is shown in Figure 16.

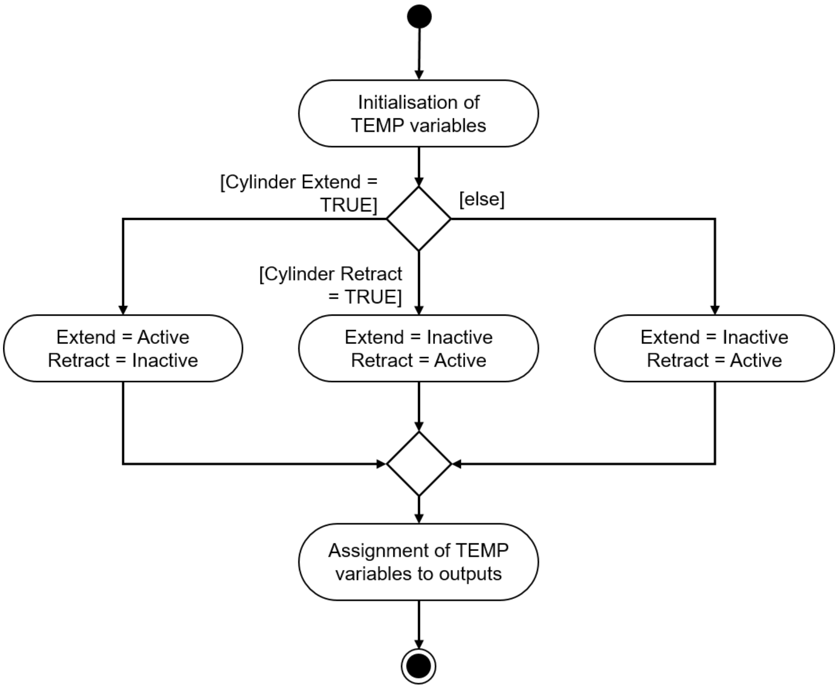


Figure 16: Activity diagram of the FB "CylinderControl"

This FB also begins with the initialization of the temporary tags and ends with their assignment to the corresponding outputs.

Since the ejector is a bidirectional actuator, it must always be ensured that only one of the two output signals are set to the logical value "**1**" at the same time.

If the ejector is to be extended, the signal for extension must be activated and the signal for retraction deactivated. The signal for extension must now assume the logical value "**1**" until the cylinder has been extended. The retract signal remains locked while the cylinder is extending.

The retraction of the ejector is to be realized like the extension with activation of the retraction signal as well as deactivation and blocking of the extension signal.

When the ejector is completely retracted, both signals should be deactivated in the optimal case. Since no inertia was defined for the ejector cylinder in the underlying MCD model, the ejector could move independently due to gravitational effects. To prevent this, the signals in this case should be controlled in the exact same way as during retraction.

There are still signals from the limit position sensors of the ejector. These have no further task in this FB. They serve only for future development of the system.

7.1.5 FB SortingPlantControl

The logic of the dynamic 3D model is mainly realized in this FB. It calculates the output signals for the MCD simulation as a result of the PLC input signals from the MCD simulation. An activity diagram with an overview of the various tasks of the FB is shown in Figure 17.

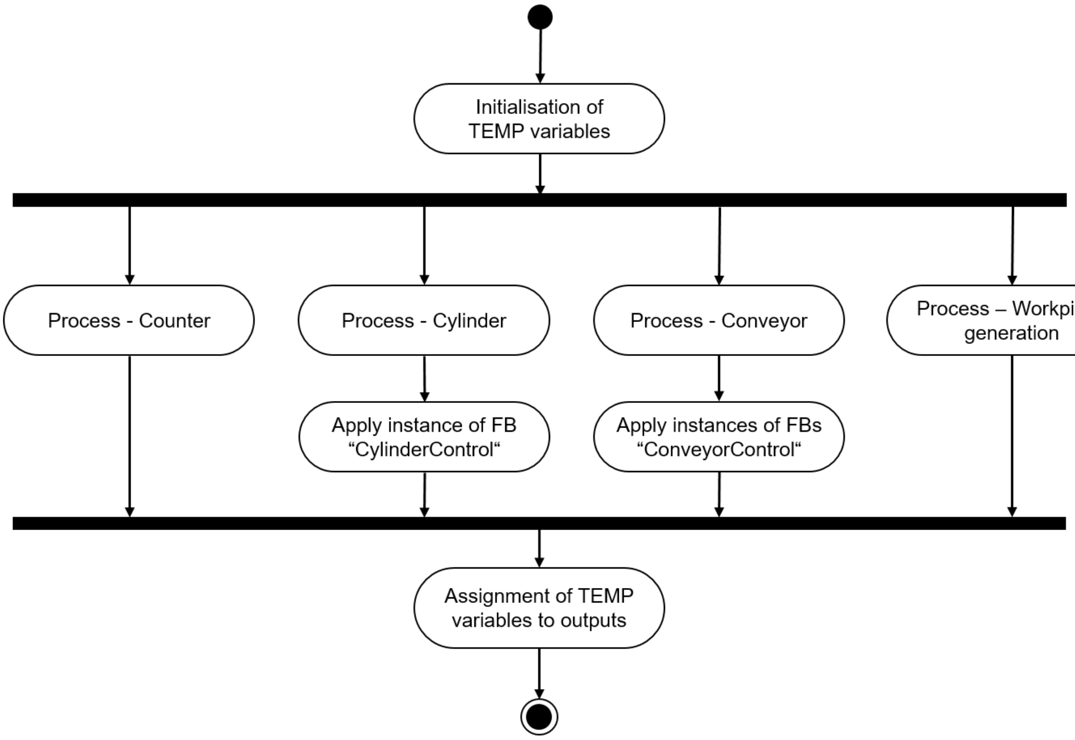

Initialisation of TEMP variables
Process - Counter
Process - Cylinder
Process - Conveyor
Process – Workpiece generation
Apply instance of FB “CylinderControl“
Apply instances of FBs “ConveyorControl“
Assignment of TEMP variables to outputs

Figure 17: Activity diagram of the FB "SortingPlantControl" in general

The initialization of the temporary tags is also the beginning in this FB. The final step is the assignment of these tags to the corresponding outputs.

The FB "SortingPlantControl" contains various processes that ensure the seamless operation of the plant.

Figure 18 shows the activity diagram for the counter functionality. There is a separate counter for each light sensor from Chapter 4.5, Chapter 4.6 and Chapter 4.7. This applies to each of these counters: If the respective light sensor is triggered, i.e. a positive edge occurs, the corresponding counter increments. Otherwise, it retains its original value and saves it for the next cycle.

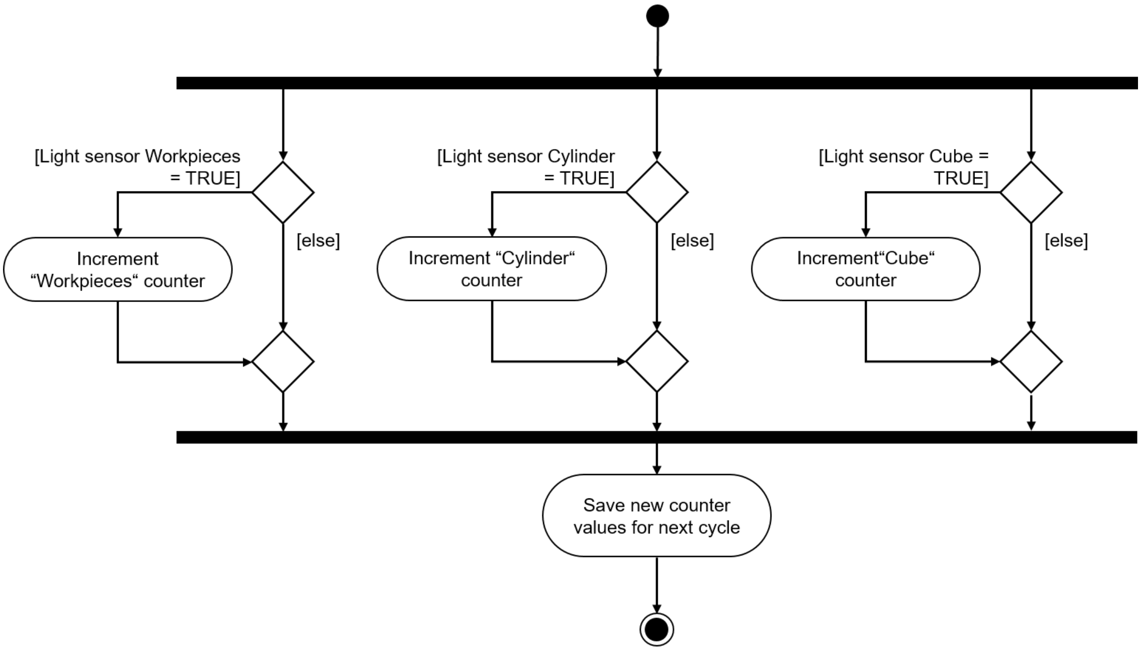

[Light sensor Workpieces = TRUE]
[else]
Increment "Workpieces" counter
[Light sensor Cylinder = TRUE]
[else]
Increment "Cylinder" counter
[Light sensor Cube = TRUE]
[else]
Increment"Cube" counter
Save new counter values for next cycle

Figure 18: Activity diagram for the counters within FB "SortingPlantControl"

The sequence for the ejector within FB "SortingPlantControl" is shown in Figure 19.

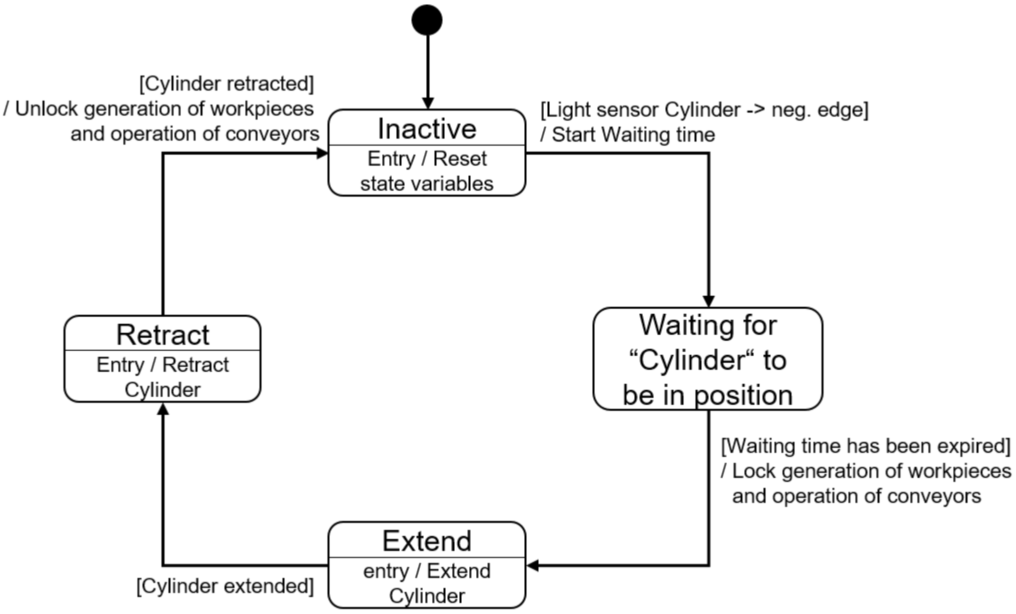

[Cylinder retracted]
/ Unlock generation of workpieces and operation of conveyors
Inactive
Entry / Reset state variables
[Light sensor Cylinder -> neg. edge]
/ Start Waiting time
Retract
Entry / Retract Cylinder
Waiting for "Cylinder" to be in position
[Waiting time has been expired]
/ Lock generation of workpieces and operation of conveyors
Extend
entry / Extend Cylinder
[Cylinder extended]

Figure 19: Status diagram for the ejector within FB "SortingPlantControl".

If the "Cylinder" light sensor has not yet triggered, the "Cylinder" ejector remains in the "Inactive" state. When this state is entered, the status tags of the ejection process are reset. According to Figure 12, the ejection process does not start until the "Cylinder" light sensor shows a negative edge. In this case, the waiting time starts and, with the change to the next state, the sequence waits for the cylindrical body to reach the ejection position. The following assumption has been made for the waiting time:

- Figure 12 shows that the workpiece must be transported 20 mm further for optimum ejection of the cylindrical body.
- The constant speed of 0.05 m/s (= 50 mm/s) was assumed as the speed reference.
- This results in a waiting time of $t_{Wait} = \frac{20\,mm}{50\,mm/s} = 400\,ms$.

The extension of the ejector arm can be prepared when the waiting time expires. From this point on, the long conveyor belt, "ConveyorLong", must not move any further, otherwise proper sorting is no longer guaranteed. To avoid a backup, both conveyor belts are stopped, and the generation of new workpieces is blocked. Only then can the ejection process begin.

In the "Extend" state, the FB "CylinderControl" instance is given the command to extend. As soon as the limit switch signals complete extension, the cylindrical body is sorted out. Now the cylinder must be completely retracted again before the conveyor belts can restart and new workpieces can be generated again.

The "Retract" state forwards the corresponding retract command to the FB "CylinderControl" instance. When the limit switch signals that the cylinder has been completely retracted, the block for workpiece generation is lifted and the conveyor belts can start running again. The cylinder then waits in the "Inactive" state until the next cylindrical body arrives.

The logic of the conveyor belts is part of FB "ConveyorControl" (see Chapter 7.1.2). Additional tasks within the "SortingPlantControl" FB are shown in Figure 20. As already noted in the description of the state diagram for the ejector, the conveyor belts should stop while the ejector is active. This information is then forwarded to the corresponding instance FBs of "ConveyorControl". If the ejector is inactive, both conveyor belts can be controlled without restrictions.

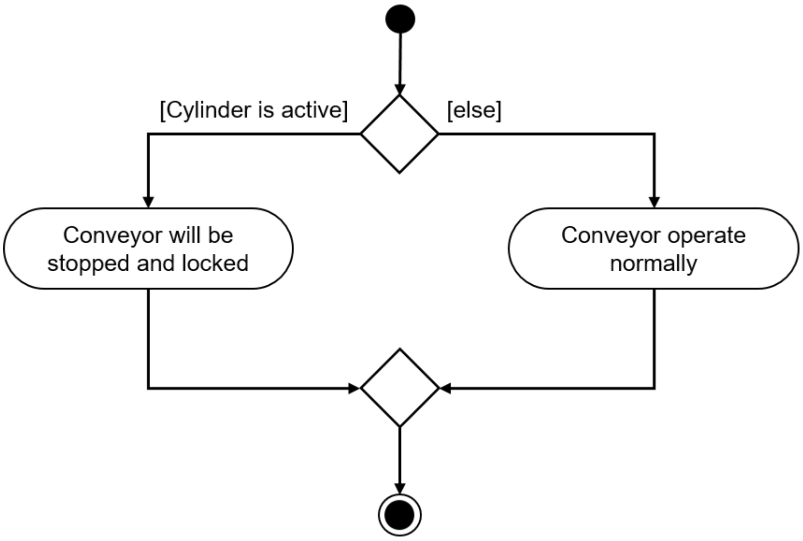

[Cylinder is active]
[else]
Conveyor will be stopped and locked
Conveyor operate normally

Figure 20: Activity diagram for the conveyor belts within FB "SortingPlantControl".

Figure 21 shows that the workpiece generation takes place simultaneously with the sequence of the conveyor belt. If the ejector is active, the generation of new workpieces is prevented. Otherwise, new workpieces can be created. It should be noted that both signals for generating new workpieces (generating cylindrical bodies and generating cuboid bodies) are controlled simultaneously by the FB.

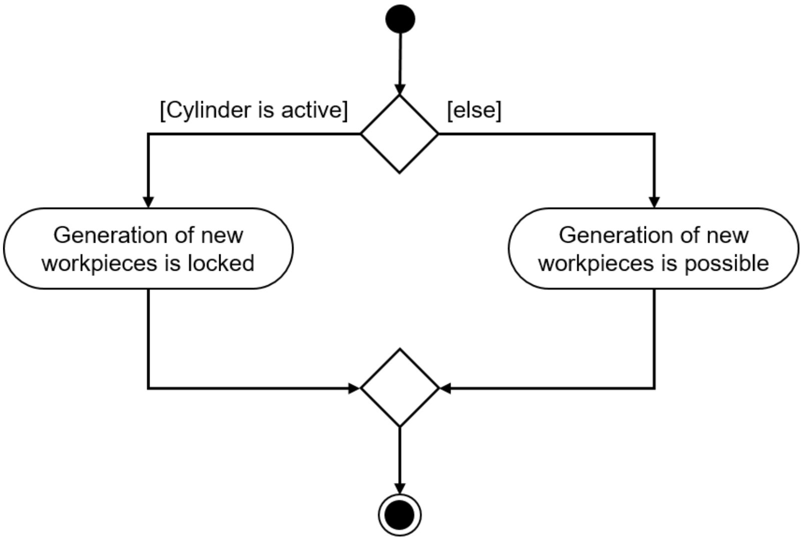

[Cylinder is active]
[else]
Generation of new workpieces is locked
Generation of new workpieces is possible

Figure 21: Activity diagram for workpiece generation within FB "SortingPlantControl".

7.1.6 FC ResetSimulation

During this function, there is a check to determine whether the user has triggered the "Reset Simulation" signal. If this is the case, all outputs of the automation program are reset. Otherwise, the outputs remain at the values assigned in FB "SortingPlantControl". This includes the output signals from the PLC for the 3D model and the output signals from the PLC for the HMI. The latter are explained in Chapter 7.2.

7.1.7 DB Control_HMI

A data block is used for data exchange between the PLC and the HMI. The following exchange is realized with this:

- The data from the HMI is transferred to FB "SortingPlantControl" via this DB so that the user can control the model as described in Chapter 7.1.5.
- The data from FB "SortingPlantControl" that is to be made available to the user, e.g. status messages such as "active/inactive" or counter values, is transferred to the HMI via this DB.

You can find a more detailed explanation of how the HMI works in Chapter 7.2.

7.1.8 Main (OB1)

There are two networks within OB1:

- In the first network, an instance of the FB "SortingPlantControl" is called. The instance is connected to the inputs and outputs of the 3D model and HMI.
- In the second network, FC "Reset Simulation" is called. This function is only activated if the user has activated the ResetSimulation signal.

The proposed solution of the PLC has therefore been thoroughly discussed. In the following chapter, you will learn more about the design of the HMI and its connection to the PLC.

7.2 HMI design

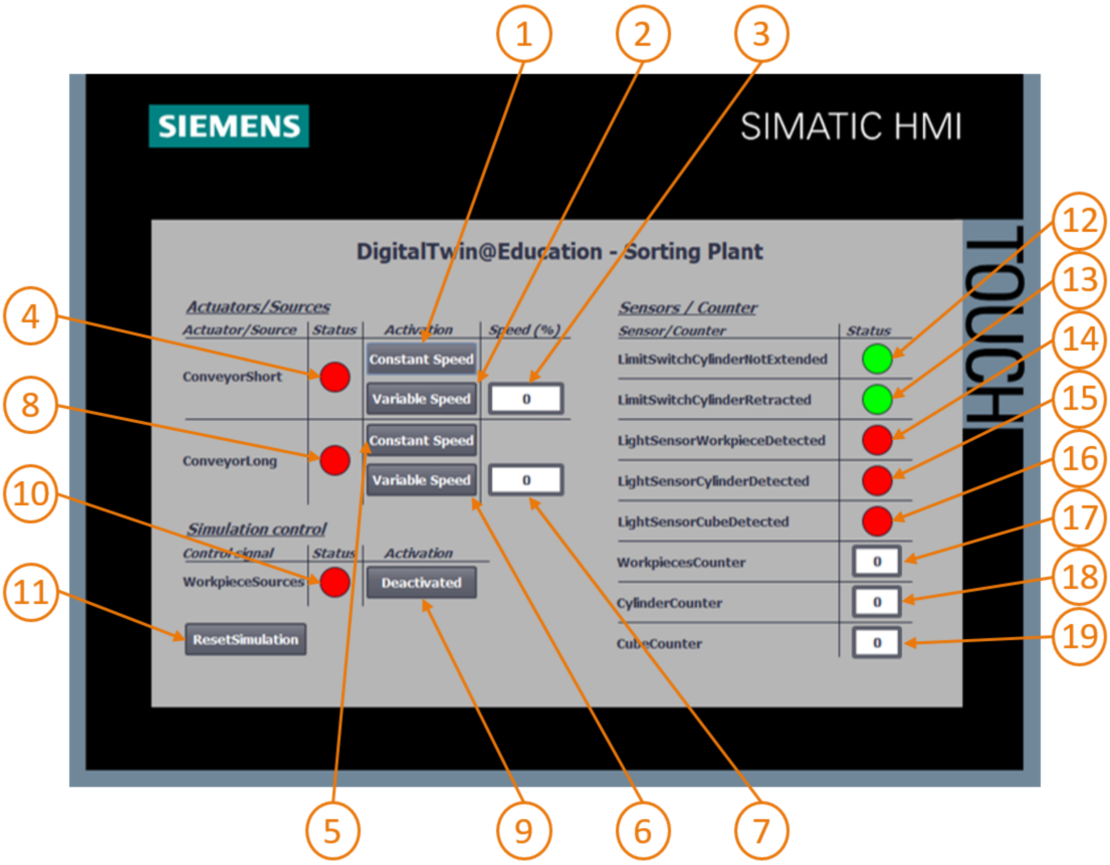


Figure 22: HMI implementation for controlling the "SortingPlant" model by the user

Figure 22 shows the implemented HMI with text boxes, buttons and input/output fields (I/O fields). The HMI design as well as the functionality and configuration of the HMI fields, marked with numbers in Figure 22, are described below.

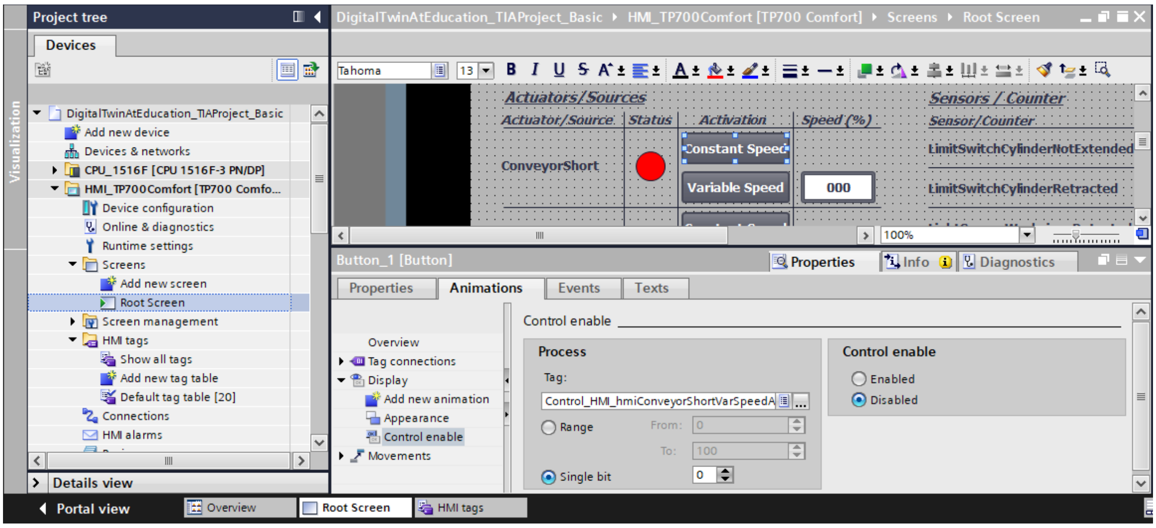
The explanation is divided into 3 chapters:

- *Actuators/Sources*: This chapter lists the externally controllable actuators of the system. These are the two conveyor belts, "ConveyorShort" and "ConveyorLong".
- *Sensors/Counter*: This chapter contains all sensors and counter values of the system. The sensor information contains the signals of the light sensors and the limit switches of the ejector. The workpiece counters described in Chapter 7.1.5 are also displayed in the HMI.
- *Simulation control*: This chapter summarizes specific commands of the simulation. In addition to the command for generating new workpieces, this also includes the button for resetting the simulation in the PLC and the HMI.

7.2.1 Actuators/Sources

The Constant Speed button for the "ConveyorShort" actuator (see Figure 22, element 1) enables the user to start the controller for moving the transport surface at constant speed. It has two different animations and one event.

- The first animation relates to the operability of the button. As noted in Chapter 5.2, it must be ensured that both controllers (constant speed and variable speed) are not active at the same time. Therefore, this button is disabled if the variable speed controller has already been activated. This is illustrated in Figure 23.



Control enable

Process

Tag: Control_HMI_hmiConveyorShortVarSpeedA

○ Range From: 0 To: 100

● Single bit 0

Control enable

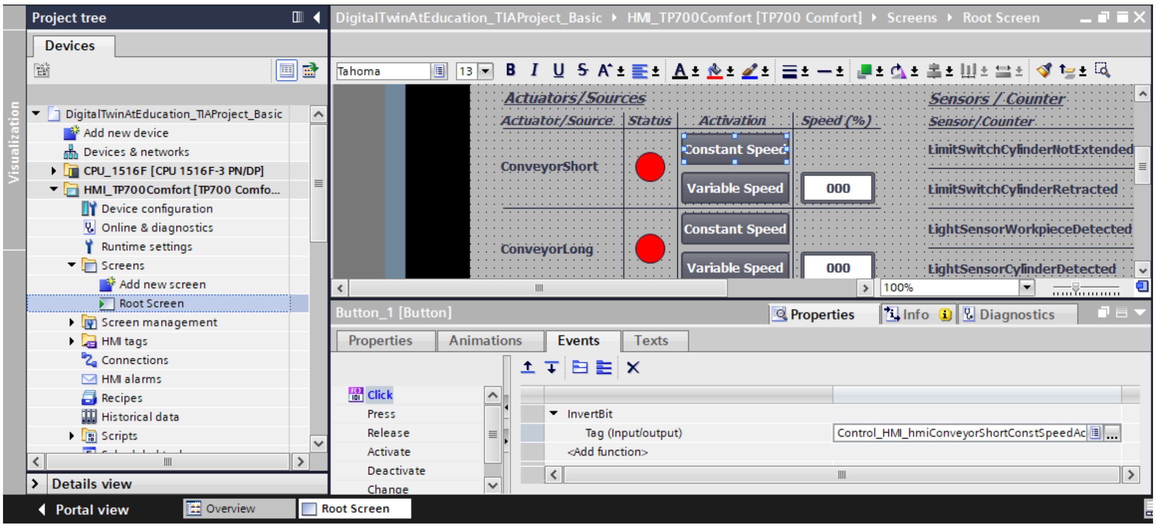
○ Enabled

● Disabled

Figure 23: Animation parameters of the HMI, in this case locking a button

- The second animation relates to the design of the button: When the Constant Speed controller is activated, the button turns blue. The button remains gray when deactivated.

- The event acts according to the following functionality: When the button is clicked, the tag for activating the constant speed controller is inverted (see Figure 24)



InvertBit
Tag (Input/output) Control_HMI_hmiConveyorShortConstSpeedAc
<Add function>

Figure 24: Event parameter of the HMI, here InvertBit when clicking a button

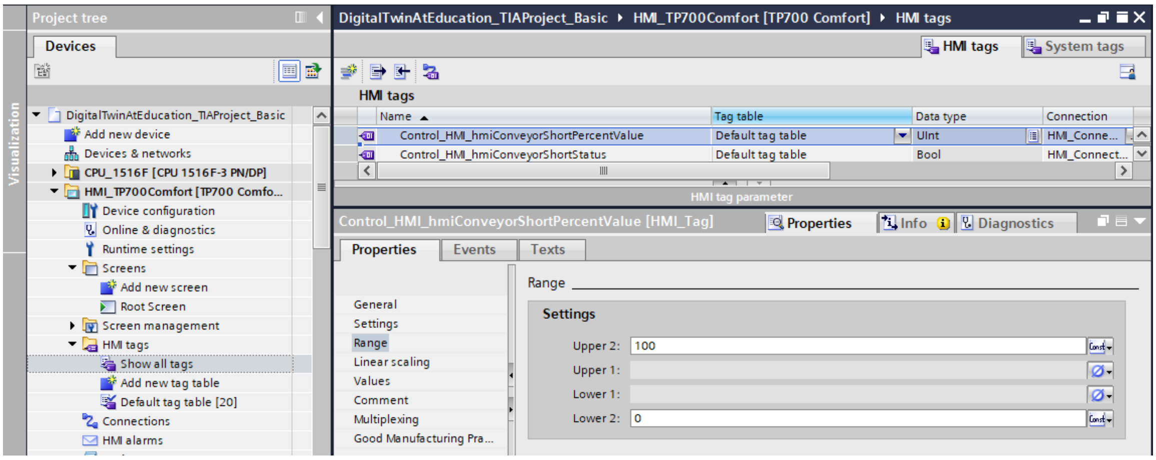
The "Variable Speed" button of the "ConveyorShort" actuator (see Figure 22, element 2) controls the controller for variable speed. Two animations and one event are also configured in this button.

- The operability of the button is opposite from that of the previously described button: The "Variable Speed" button may only be operated if the Constant Speed button of the "ConveyorShort" actuator has not been activated by the user.
- When the controller for variable speed is activated, the button turns blue. It remains gray in the deactivated state.
- By clicking the button, the tag for controlling the controller with variable speed is inverted as an event.

The input field box behind the "Variable Speed" button (see Figure 22, element 3) allows the user to specify a percentage travel speed for the conveyor belt. However, this is only transmitted to the model if the controller for the variable speed is activated. For this, a reference to the corresponding tag of the DB "Control_HMI" has been defined in the properties of the I/O field. Two animations have been defined, which correspond to those of the "Variable Speed" button:

- The I/O field may only be operable if the constant speed controller for the "ConveyorShort" conveyor belt has not yet been activated.
- Similar to the button, the I/O field is grayed out if the constant speed controller is active.

The input field must transmit the specified motor speed to the PLC in percent as an integer. The data type *UInt* was used for this. Furthermore, the value range from *0* to *100* is to apply. This was specified via the properties of the tag as seen in Figure 25.



HMI tags

Name ▲	Tag table	Data type	Connection
Control_HMI_hmiConveyorShortPercentValue	Default tag table	UInt	HMI_Conne...
Control_HMI_hmiConveyorShortStatus	Default tag table	Bool	HMI_Connect...

Control_HMI_hmiConveyorShortPercentValue [HMI_Tag]

Range

Settings

Upper 2: 100
Upper 1:
Lower 1:
Lower 2: 0

Figure 25: Defining the value range of the variable speed

The status signal of "ConveyorShort" (see Figure 22, element 4) assumes two states:

- If the conveyor belt does not move, i.e. no controller is active, the "Inactive" status is displayed in the form of a red circle. The same applies if the conveyor belt is locked because the ejector is sorting out a cylindrical body.
- If one of the two controllers have been activated and the ejector is not sorting out a cylindrical body, the "Active" status is indicated by a green circle.

The buttons (see Figure 22, elements 5+6), the I/O field (see Figure 22, element 7) and the status display (see Figure 22, element 8) of the "ConveyorLong" behave similarly to the buttons of the "ConveyorShort" described above. However, the tags used here no longer refer to the "ConveyorShort" conveyor belt, but instead to "ConveyorLong".

7.2.2 Sensors/Counter

This chapter explains all sensor signals displayed in the HMI (see Figure 22, elements 12 - 16). These include:

- Both limit switches ***csLimitSwitchCylinderNotExtended*** and ***csLimitSwitchCylinderRetracted*** of the ejector (see Chapter 4.4)
- The three light sensors for detecting the workpieces (see Chapter 4.5, Chapter 4.6 and Chapter 4.7)

The functionality of the status displays is the same for all sensor signals. If a light sensor or position sensor is actuated, a green circle indicates the "Active" status. In all other cases, a red circle indicates the "Inactive" status.

NOTE

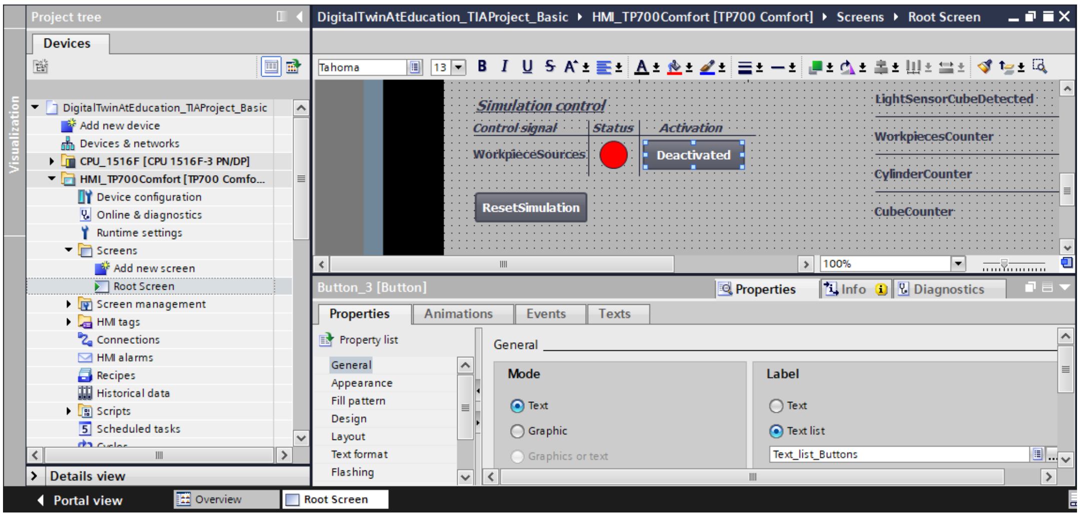
In contrast to all other tags, the sensor signals are picked up and displayed directly by the input tags of the PLC. Buffering for the HMI in DB "Control_HMI" is not intended.

The output fields (see Figure 22, elements 17 - 19) show the counter values from the PLC in the HMI as described in Chapter 5.4. Since these fields only serve the output, only the corresponding process tag needs to be assigned in each case. Therefore, no further animations or events need to be configured.

7.2.3 Simulation control

For control purposes, the button for activating the generation of new workpieces is provided in the HMI (see Figure 22, element 9). As described in Chapter 7.1.5, when this button is pressed, two signals for workpiece generation (generation of cylindrical bodies and generation of cuboid bodies) are controlled to logical value "**1**". The control of the generation of the two workpiece types independent of each other is not intended. The button has the following properties:

- This button uses a text list. When the button is activated, the text "Activated" is written on the button. Otherwise, the text "Deactivated" appears on the button. This is shown, for example, in Figure 26.



Mode: ● Text ○ Graphic

Label: ○ Text ● Text list

Text_list_Buttons

Figure 26: Button in the HMI with assigned text list

- As already used for the conveyor belts (see Chapter 7.2.1, the button turns blue in the active state, otherwise it remains gray.
- The "Click" event inverts the signal for generating new workpieces.

If generation of new workpieces is activated, a green circle becomes visible in the status element (see Figure 22, Element 10). If it is deactivated, a red circle appears.

To execute the "Reset Simulation" function from Chapter 7.1.6, an additional button has been introduced (see Figure 22, element 11). This has an animation and two events:

- If the signal for resetting the simulation is active, the button appears in blue. Otherwise, it remains gray.
- When the "Press" event occurs, an activated reset signal is transmitted to the PLC. In addition, the controllers of both conveyor belts are deactivated and the input fields for setting the variable speed in percent are reset to ZERO.
- When the "Release" event occurs, the reset signal to the PLC is deactivated again.

The proposed solution is now complete, and you should be able to set up a similar TIA project yourself.

At the end of this module, check your self-created automation program with the simulation in MCD according to the procedure described in Module 1. Use the two test scenarios from Module 1 of the DigitalTwin@Education Workshop series.

The next module examines possible errors in the existing PLC program and discusses additional optimizations and extensions.

8 Checklist – Step-by-step instructions

The following checklist helps trainees/students to independently check whether all steps of the step-by-step instructions have been carefully completed and enables them to successfully complete the module on their own.

No.	Description	Checked
1	Theory of Module 1 of the DigitalTwin@Education Workshop series internalized.	
2	*Recommended:* Module 1 of the DigitalTwin@Education Workshop series performed in full.	
3	Functionality of the 3D model understood in detail.	
4	Based on the previous description, a corresponding automation program including visualization was created.	
5	*Optional:* The PLC program and the HMI of the proposed solution were understood and successfully implemented.	
6	The testing of the simulated PLC program with the HMI and the 3D simulation in MCD with the test scenarios from Module 1 of the DigitalTwin@Education Workshop series was successful.	

Table 1: Checklist of the "Configuration of an automation program for a dynamic 3D model in the TIA Portal".

9 Additional information

You can find additional information as an orientation aid for initial and advanced training, for example: Getting Started, videos, tutorials, apps, manuals, programming guidelines and trial software/firmware, at the following link:

Preview "Additional information" – In preparation

Here are some interesting links:

[1] support.industry.siemens.com/cs/document/90885040/programming-guideline-for-s7-1200-s7-1500?dti=0&lc=en-US

[2] support.industry.siemens.com/cs/document/109756737/guide-to-standardization?dti=0&lc=en-US

[3] omg.org/spec/UML/2.5.1/PDF

[4] geeksforgeeks.org/unified-modeling-language-uml-activity-diagrams/

[5] geeksforgeeks.org/unified-modeling-language-uml-state-diagrams/

Further Information

Siemens Automation Cooperates with Education
siemens.com/sce

SCE Learn-/Training Documents
siemens.com/sce/documents

SCE Trainer Packages
siemens.com/sce/tp

SCE Contact Partners
siemens.com/sce/contact

Digital Enterprise
siemens.com/digital-enterprise

Totally Integrated Automation (TIA)
siemens.com/tia

TIA Portal
siemens.com/tia-portal

TIA Selection Tool
siemens.com/tia/tia-selection-tool

SIMATIC Controller
siemens.com/controller

SIMATIC Technical Documentation
siemens.com/simatic-docu

Industry Online Support
support.industry.siemens.com

Product catalogue and online ordering system Industry Mall
mall.industry.siemens.com

Siemens
Digital Industries, FA
P.O. Box 4848
90026 Nuremberg
Germany

Subject to change and errors

siemens.com/sce