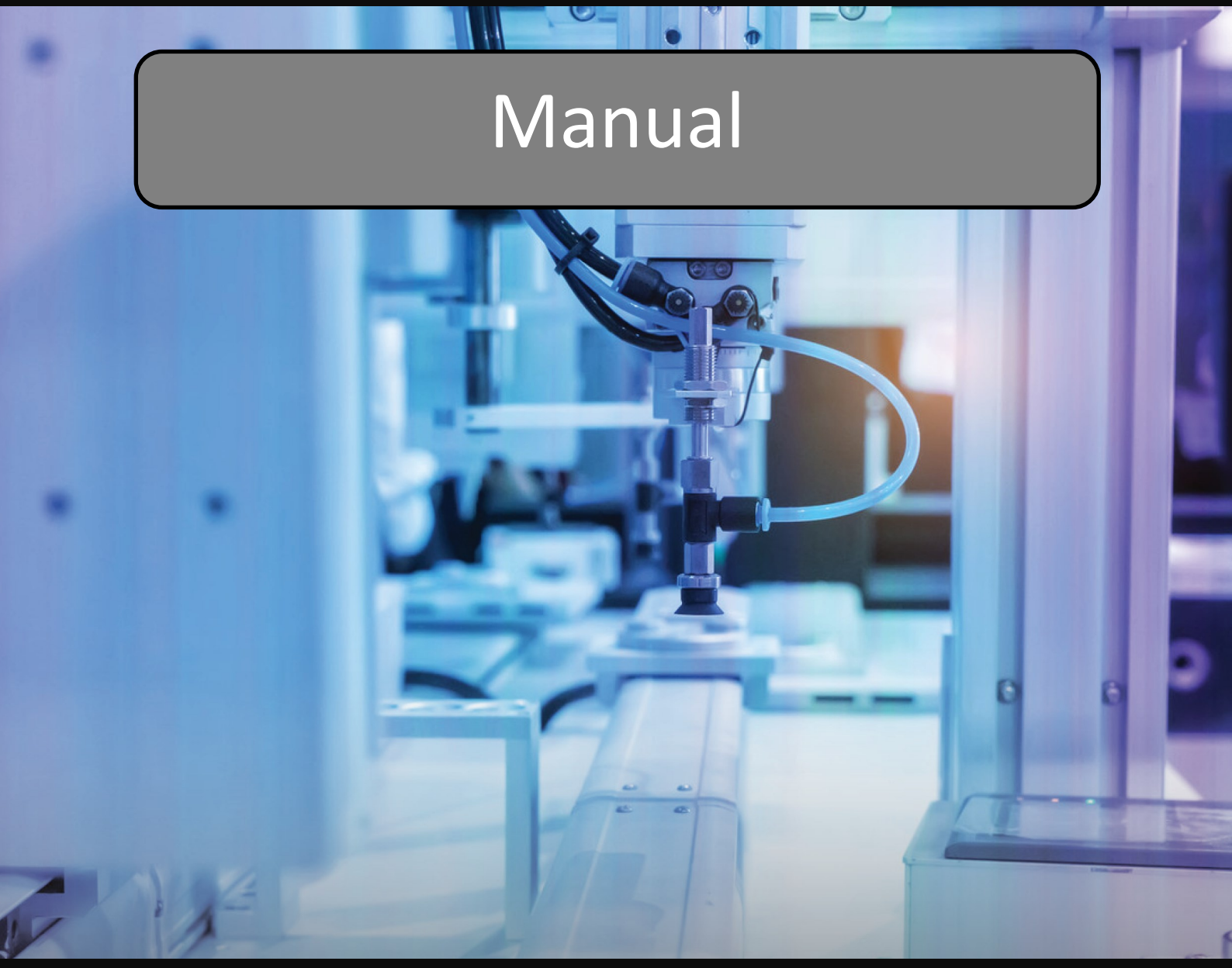
MATRIX | SMART FACTORY

Smart Factory



Manual

MATRIX

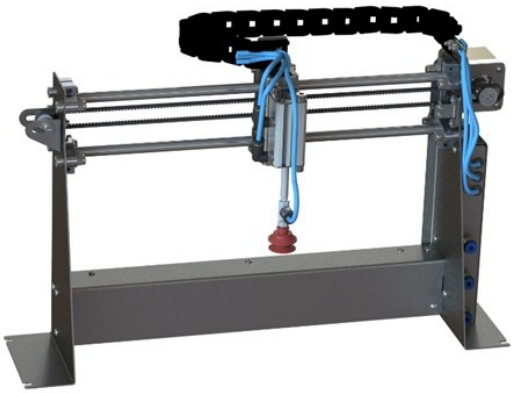
CP5279

www.matrixtsl.com

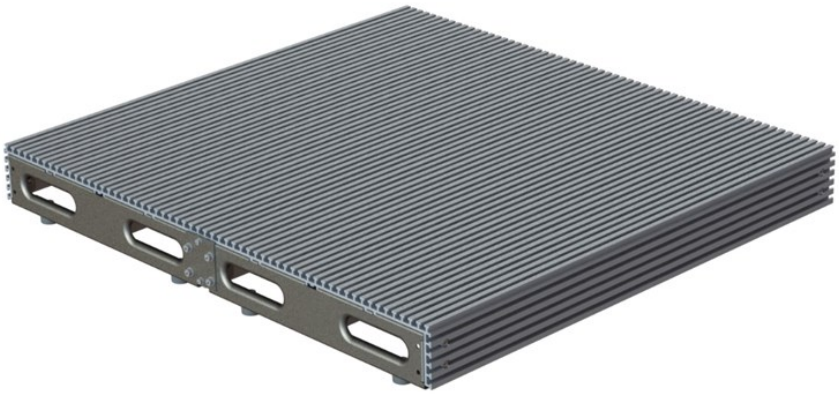
Contents

Contents of Product

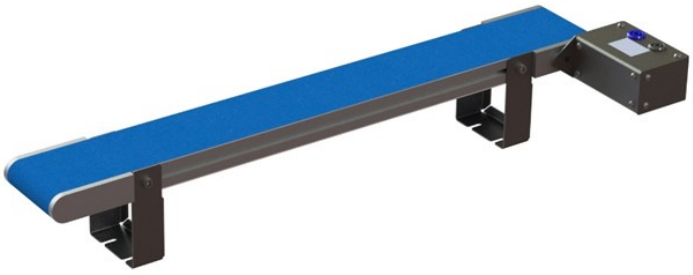
Modules



AU0696 – Gantry x1

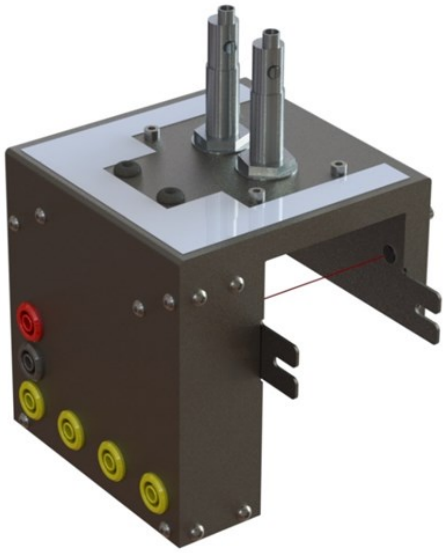


AU9318 – Base x1



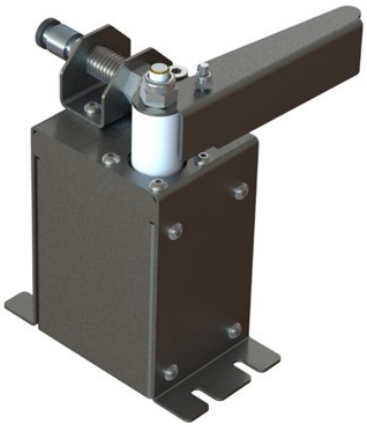
AU4353 – Conveyor x1

Contents of Product



AU6707 – Sensor Unit x1

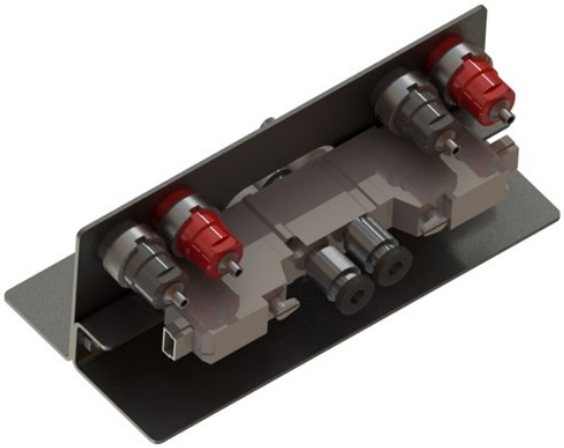
If you have earlier version of the Smart factory, you will have receiver 4 individual units instead of one. The wiring and the placement is the same.



AU1437 – Paddle x2

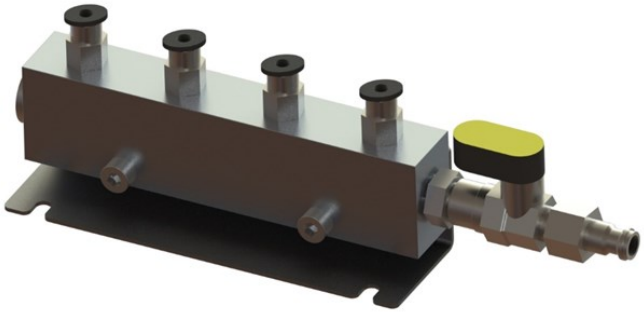
AU1443 – Vacuum Generator x1

AU9633 – 3/2 valve x3



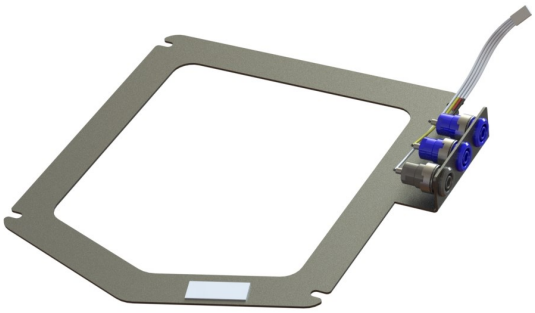
AU2834 – 5/2 valve x1

Contents of Product

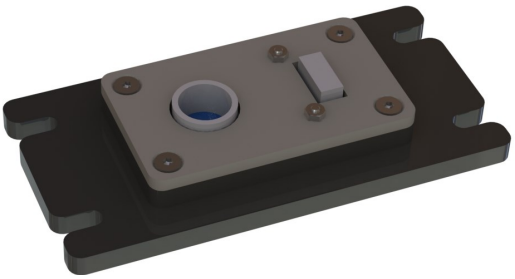


AU6004 – Manifold x1

AU0358 – Counter Rack x1



AU5775– Registration Plate x1



AU4655– Light colour sensor x1

Items

4x Steel counters

4x Aluminium counters

4x White Plastic counters

4x Black Plastic counters

5x AU7654 red bin

1x AU4655 Light sensor PCB holder

5m of pneumatic tubing

10x cable management clips

2x Lead, red. 500mm, 4mm to 4mm stackable

5x Lead, red, 250mm, 4mm to 4mm stackable

5x Lead, red, 1000mm, 4mm to 4mm Stackable

1x Lead black 500mm 4mm to 4mm stackable

3x Lead, black, 1000mm, 4mm to 4mm stackable

4x Lead, black, 250mm, 4mm to 4mm stackable

1x Lead, blue, 500mm, 4mm to 4mm stackable

7x Lead, blue, 1000mm, 4mm to 4mm stackable

9x Lead, yellow, 1000mm 4mm to 4mm stackable

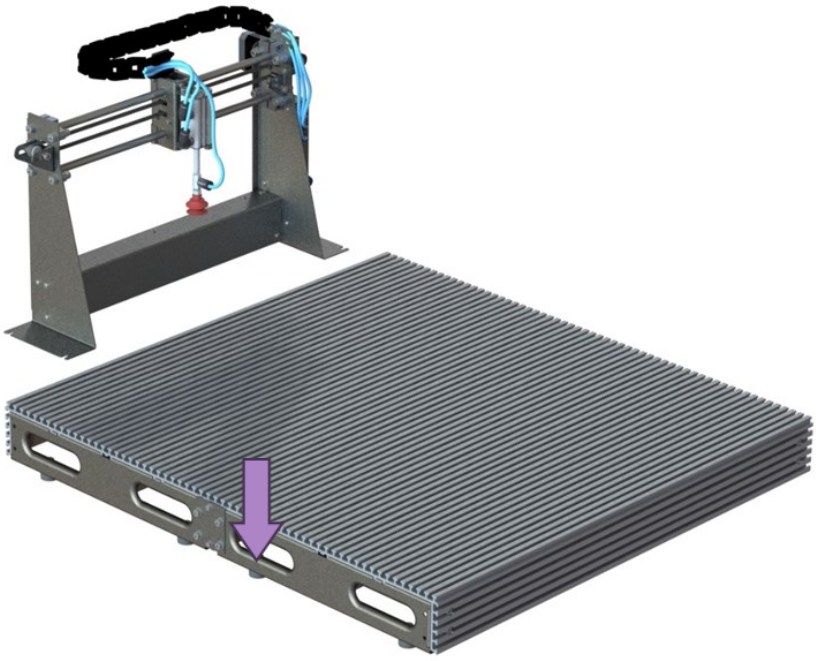
3x Lead, yellow, 250mm 4mm to 4mm stackable

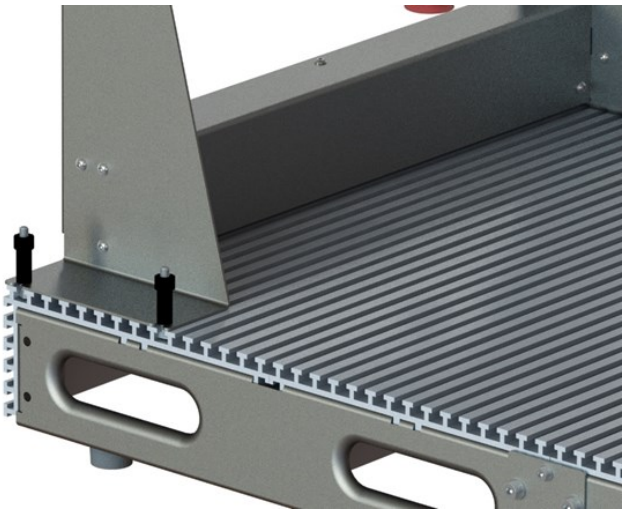
100x tee bolts

1x AU1090 permanent fixing kit

Step 1. Adding the Gantry

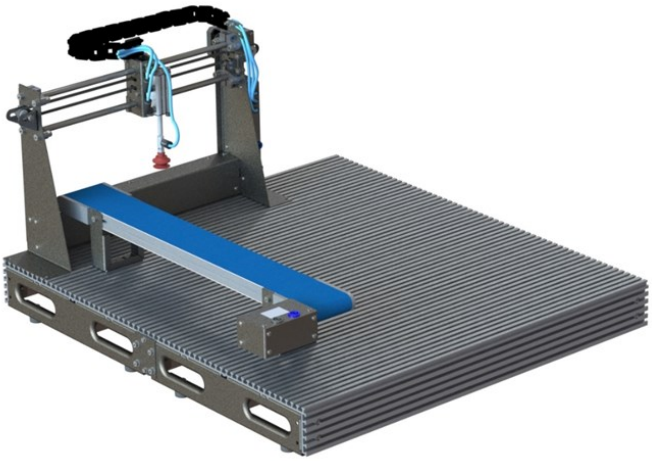
Add the gantry module to the base board in the top left corner. Use the tee nuts in the four corners of the module to fix it in place.



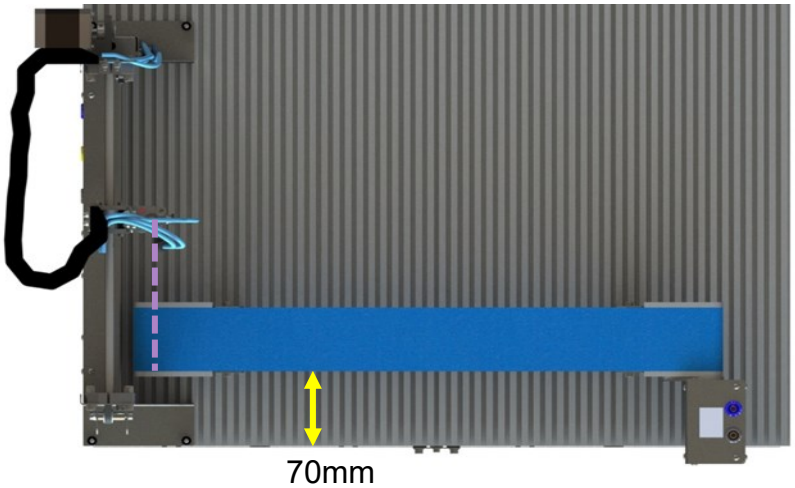


Step 2. Adding the Conveyor

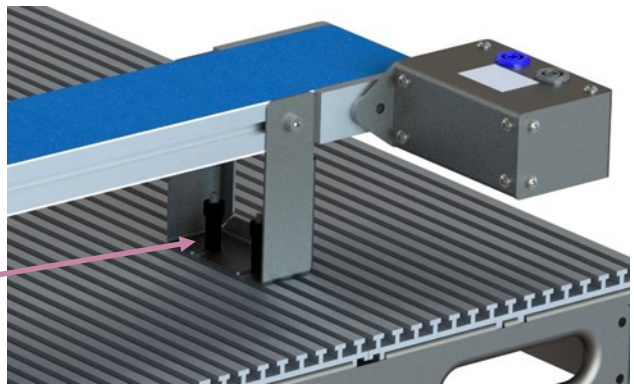
Place the conveyor module to the left side of the base unit. The motor should be in the opposite corner to the gantry unit. The 4 Tee bolts which fix the conveyor down to the base are easier to fix in place, if you place them in the right rung on the base board, and slide the conveyor into them to tighten them in place.



The final position should be that the conveyor is over the line of sight for the pick and place head. While being 70mm from the edge.



Use 2x Tee bolts in each

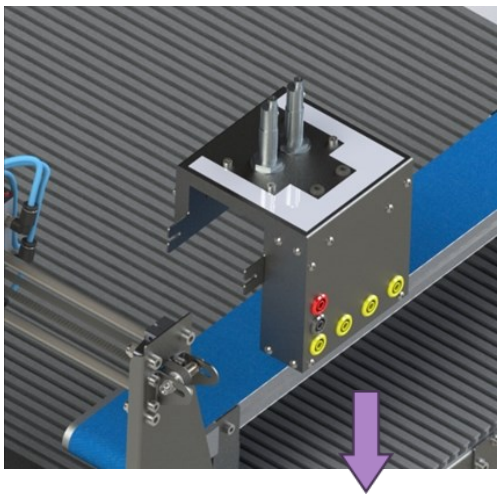


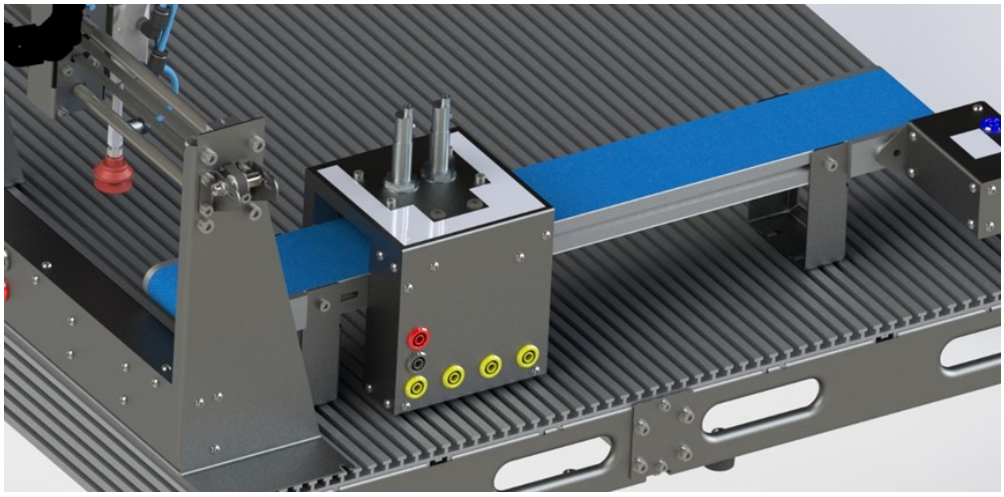
Step 3. Adding the Sensor Unit

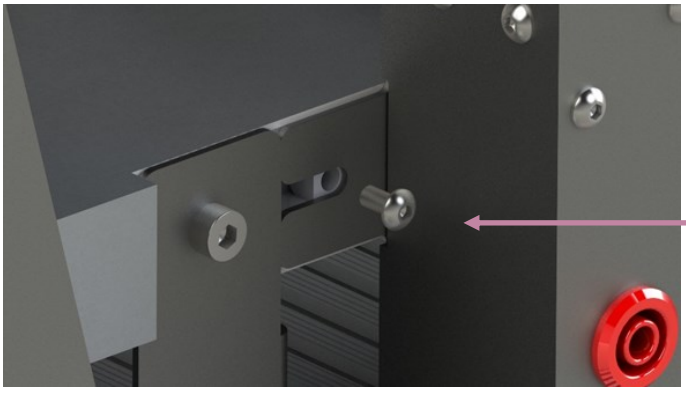
The sensor unit sits around the conveyor unit, towards the gantry end.

Insert the Tee nuts into the conveyor extrude slot first.

Slide the sensor unit as far left as possible toward the gantry, and fix down using the fixing tee nuts and M4 bolts. There are 4 positions to fix down.

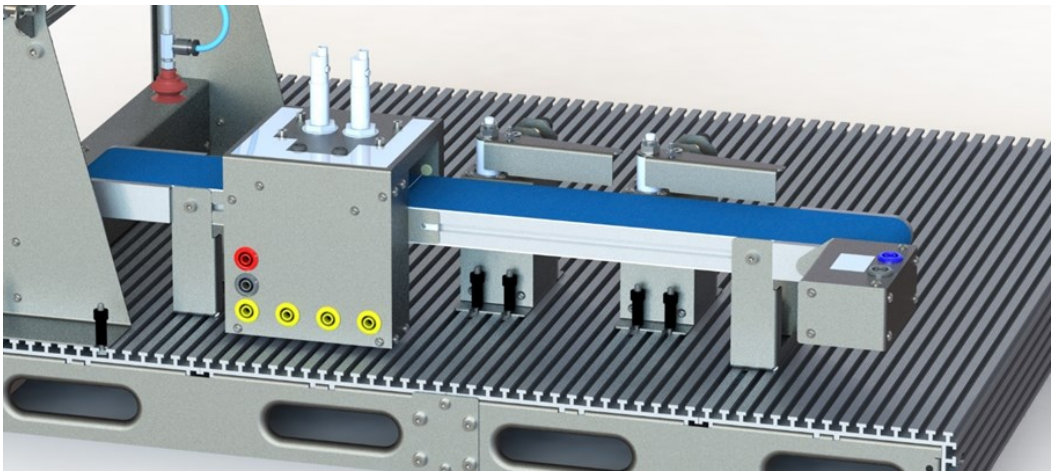


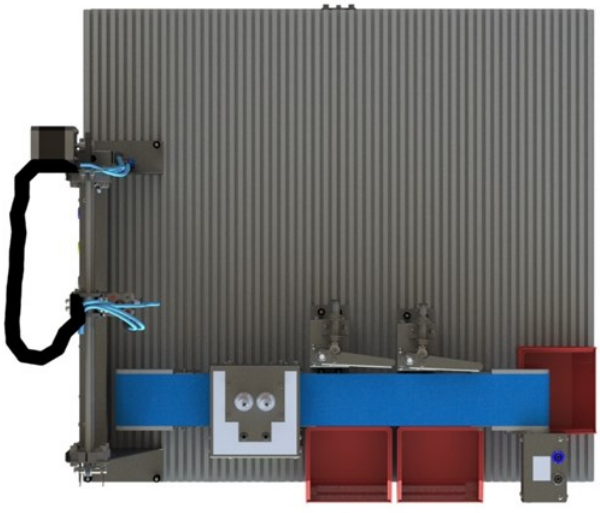


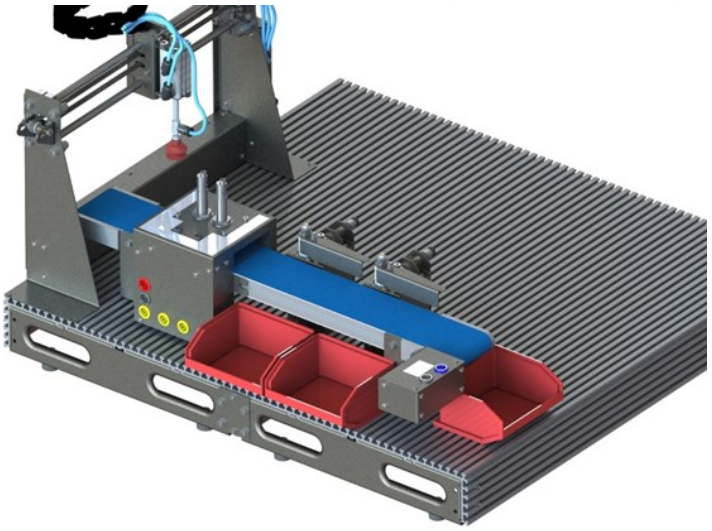


Step 4. Adding the Paddles

Add the 2 paddles to the side of the conveyor. Equally space them out, so there’s is clearance for the swing of the paddle. Fix down each paddle with 4x tee bolts.







The Paddles should only be 3mm from the edge of the conveyor.

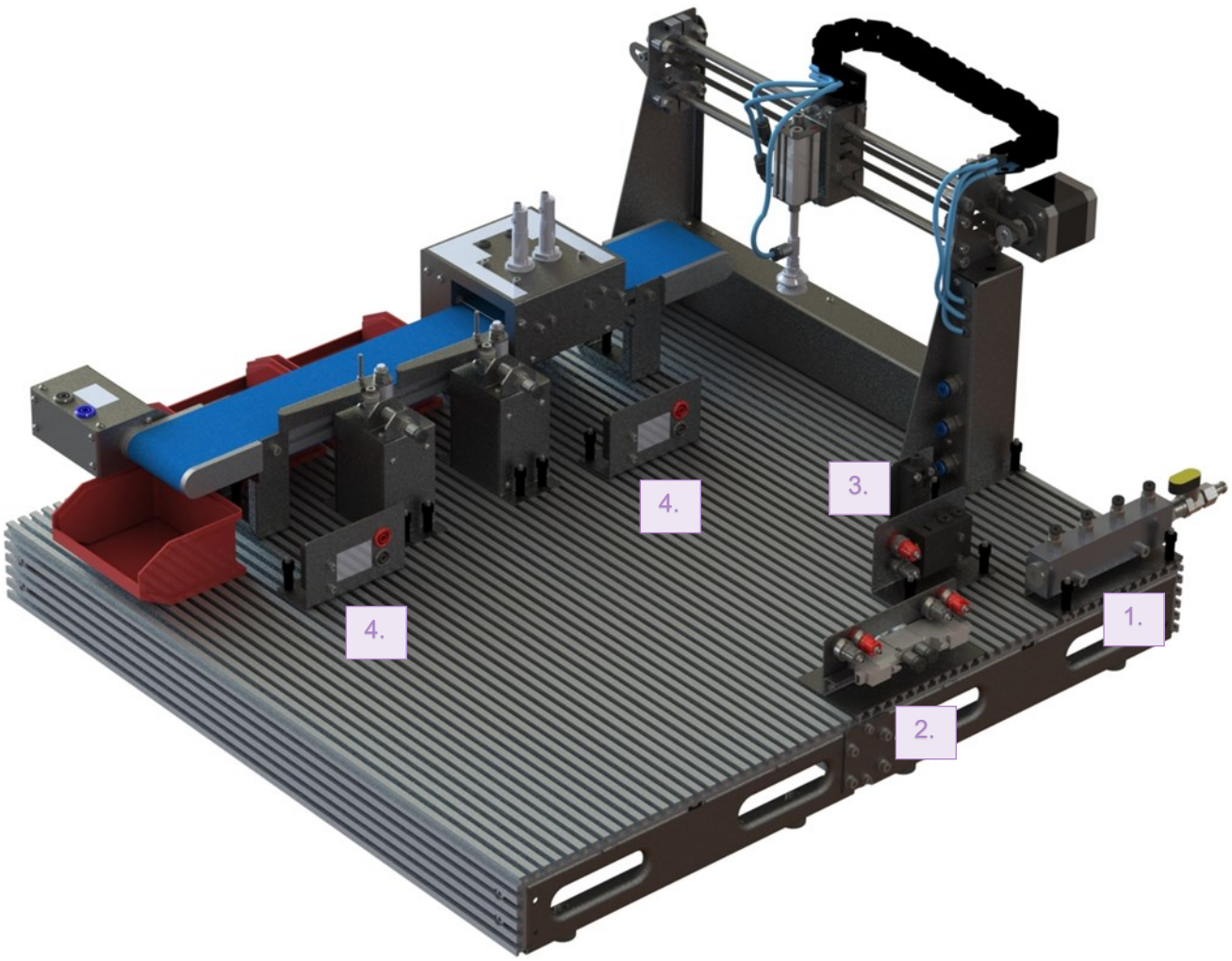
Add 3 red bins in the positions shown, the can overlap the edge of the base, as long as the base of the bin is fully on. These don’t connect down.

Step 5. Adding the Pneumatics

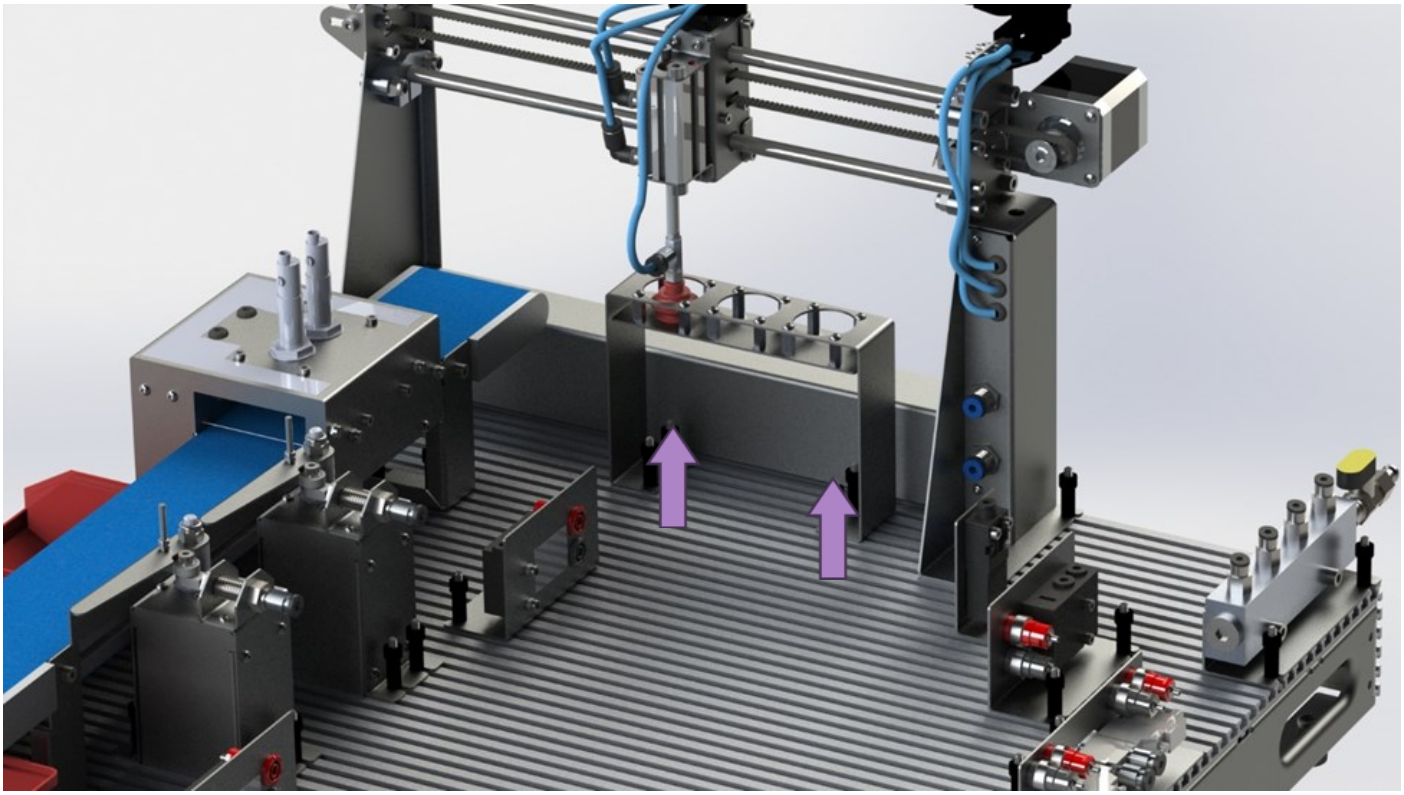
Now to add the pneumatic modules onto the base board. All these modules use tee bolts for fixtures.

Manifold (x1), 5/2 valve (x1), 3/2 valve (x3) and vacuum generator (x1).

1. Start with the manifold in the top right corner, placing it as far into the corner as possible.
2. Next place the 5/2 valve alongside it. With the valve facing outwards.
3. Next place one 3/2 valve and the vacuum generator in front of these, next to side of the gantry.
4. Place two 3/2 valves next to either paddle, one on each side. Face these in opposite direction.



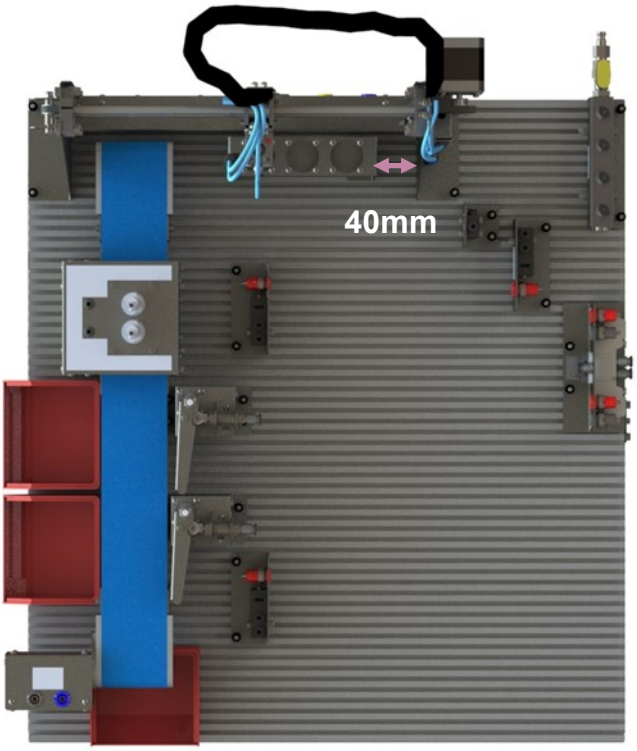
Step 6. Adding the Counter Rack



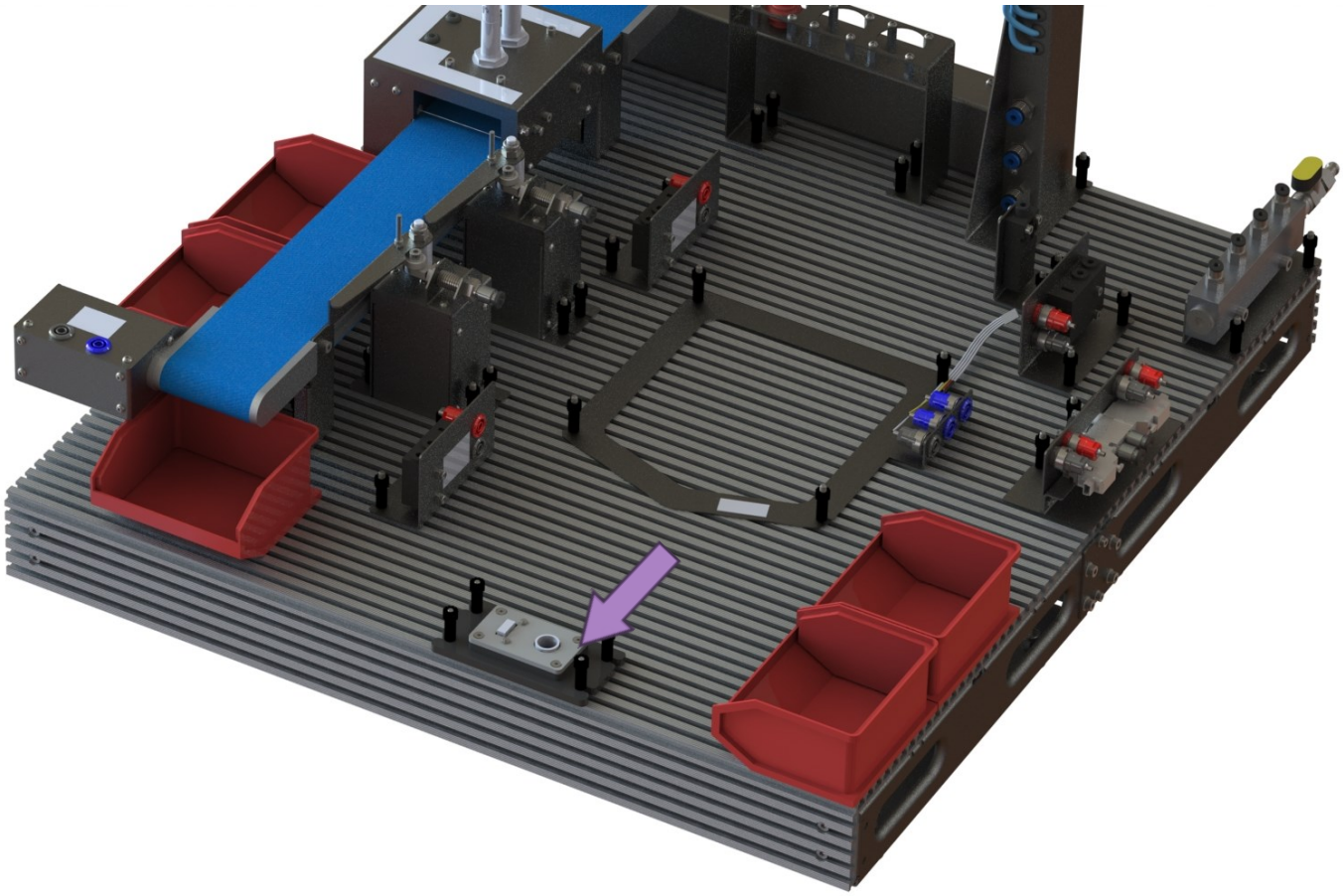
Add the counter rack under the gantry unit. Line up the 3 counter hoppers with the plunger head. The base should have a series of channels that line up for the 4 tee bolts fixings which hold the rack in place.

Place the rack about 40mm from the stepper motor side of the gantry.

Add the counters into the 3 hoppers in any order you'd like. However, by material is sensible.



Step 7. Adding the Robot Arm modules



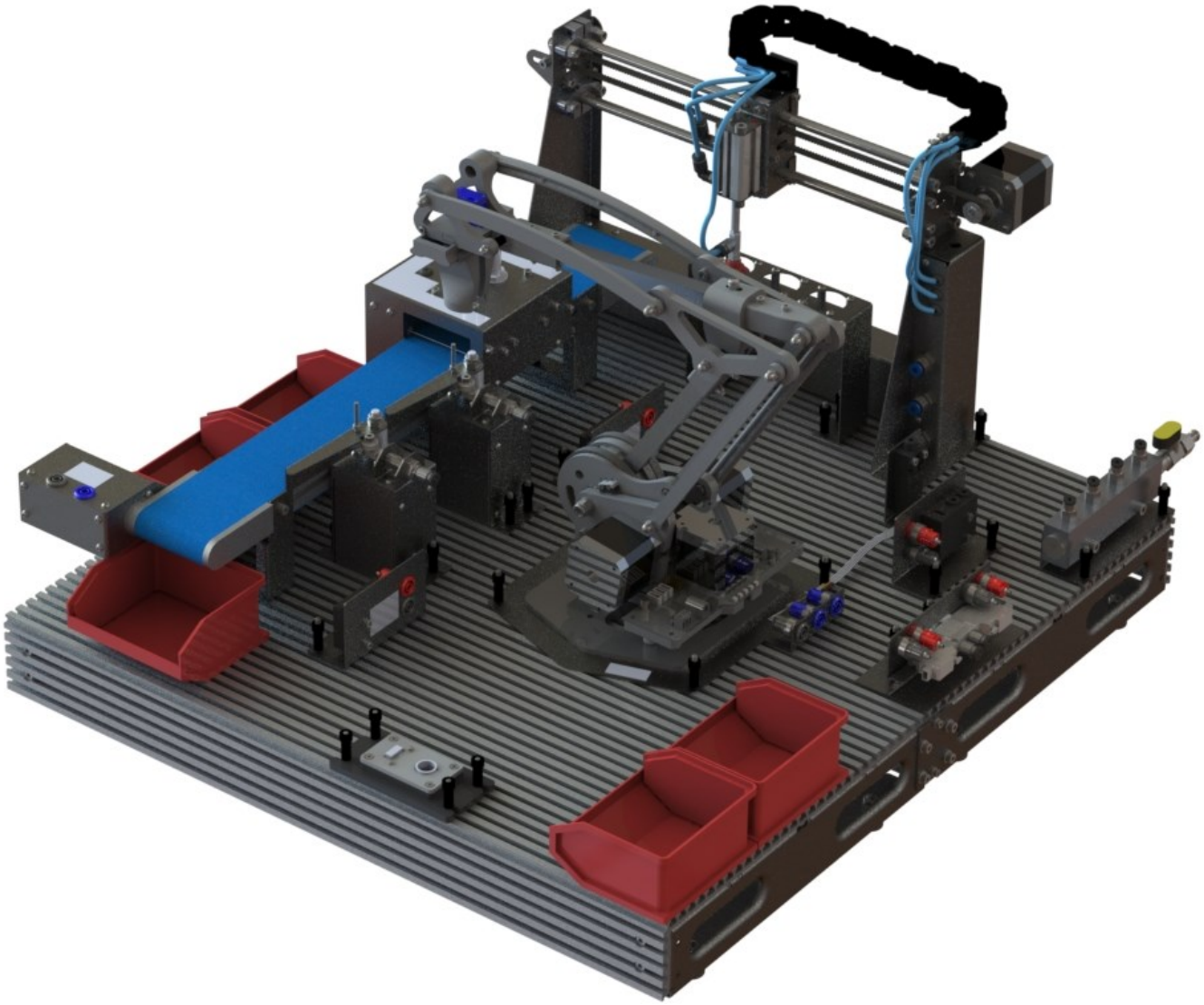
This step is optional if you didn’t purchase Matrix’s Robot Arm

Add the registration plate for the Robot arm in the middle of the base, with the pointed end facing away from the gantry module.

Add the light colour sensor module at the end of the base, with the middle of the unit lining up with the point of the registration plate.

Use 4 tee bolts to fix down both modules to the base.

Step 8. Adding the Robot Arm

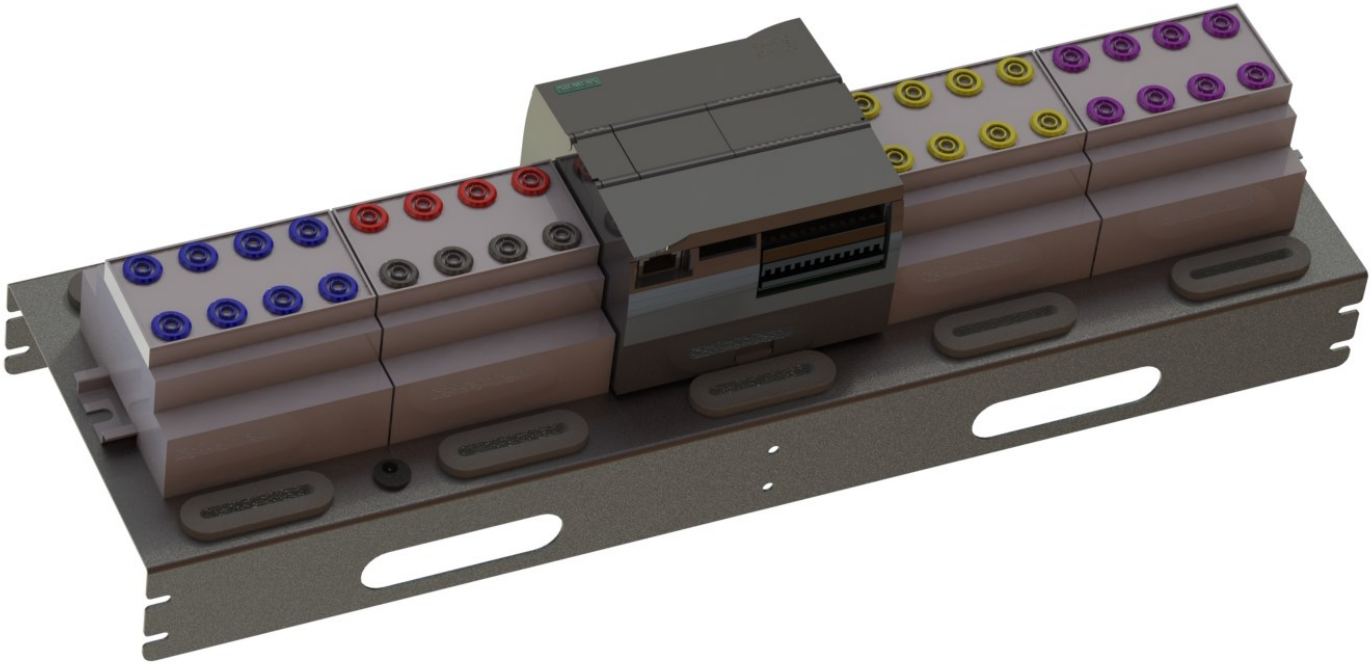


This step is optional if you didn’t purchase Matrix’s Robot Arm

Add the Robot Arm to the centre of the smart factory, using the shape of the registration plate to guide the orientation of the arm.

PLC Options

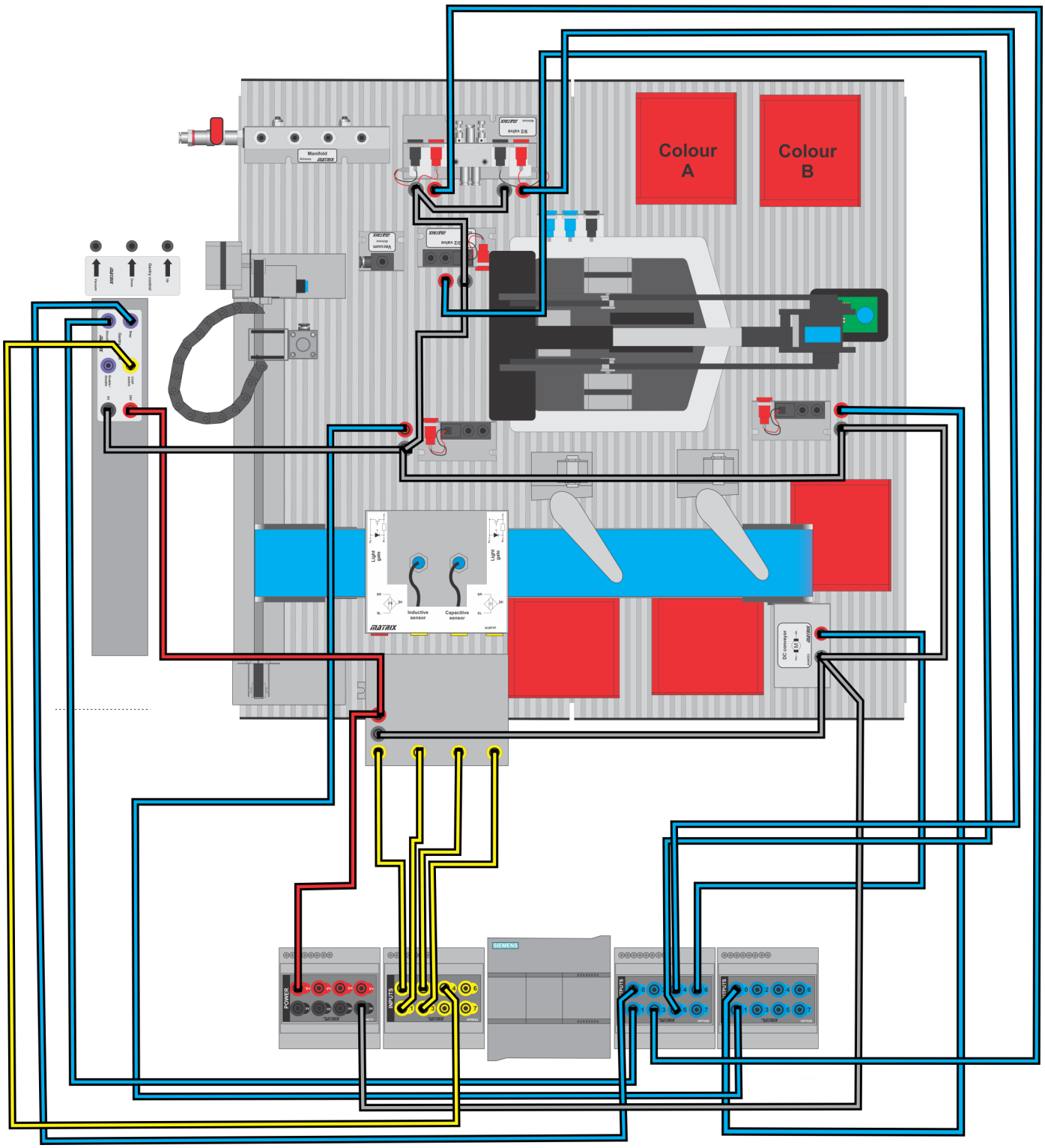
Expandable PLC (AU3686)



Siemens HMI PLC (AU0205)

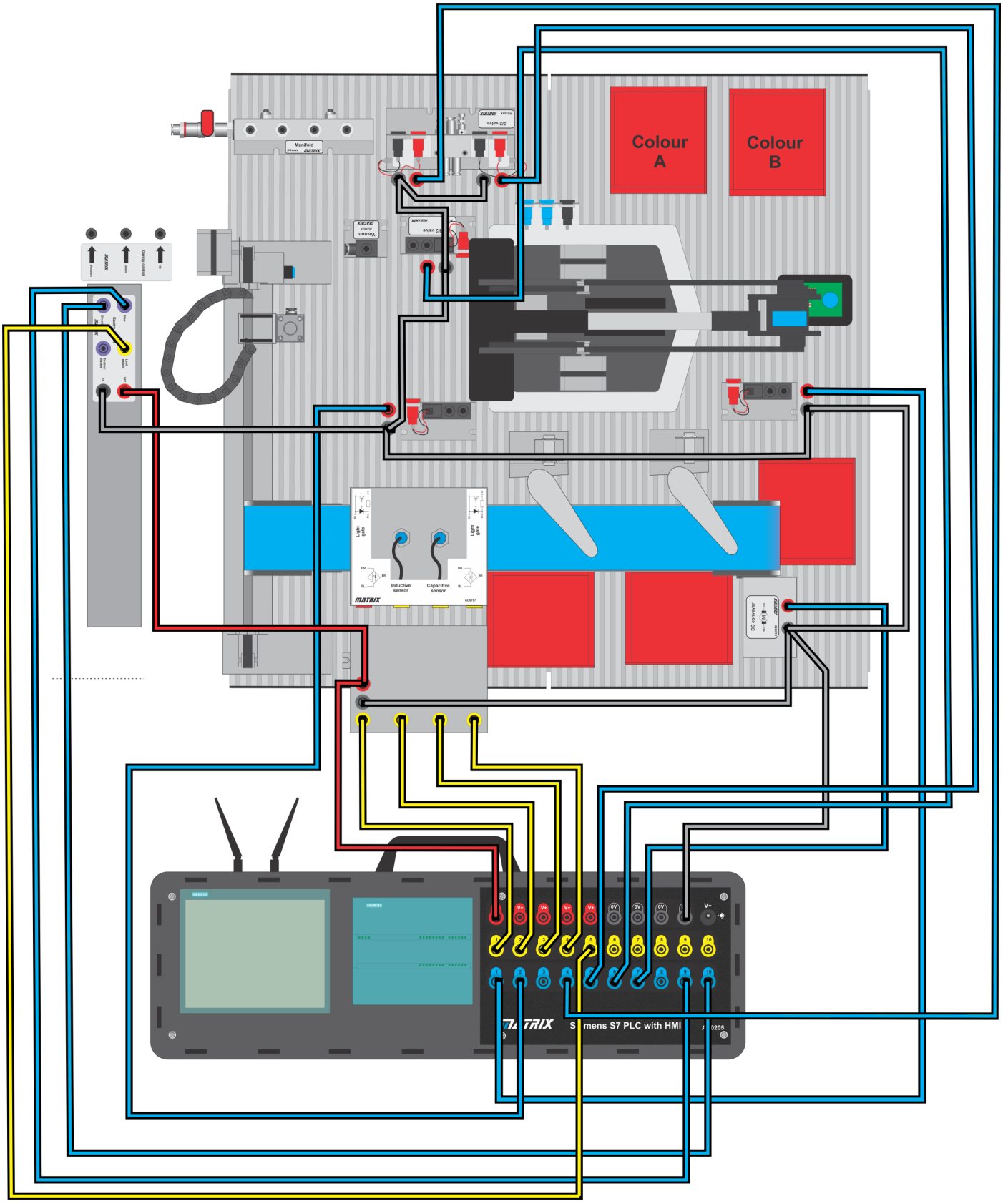
Wiring Diagrams

AU3686 Wiring Diagram

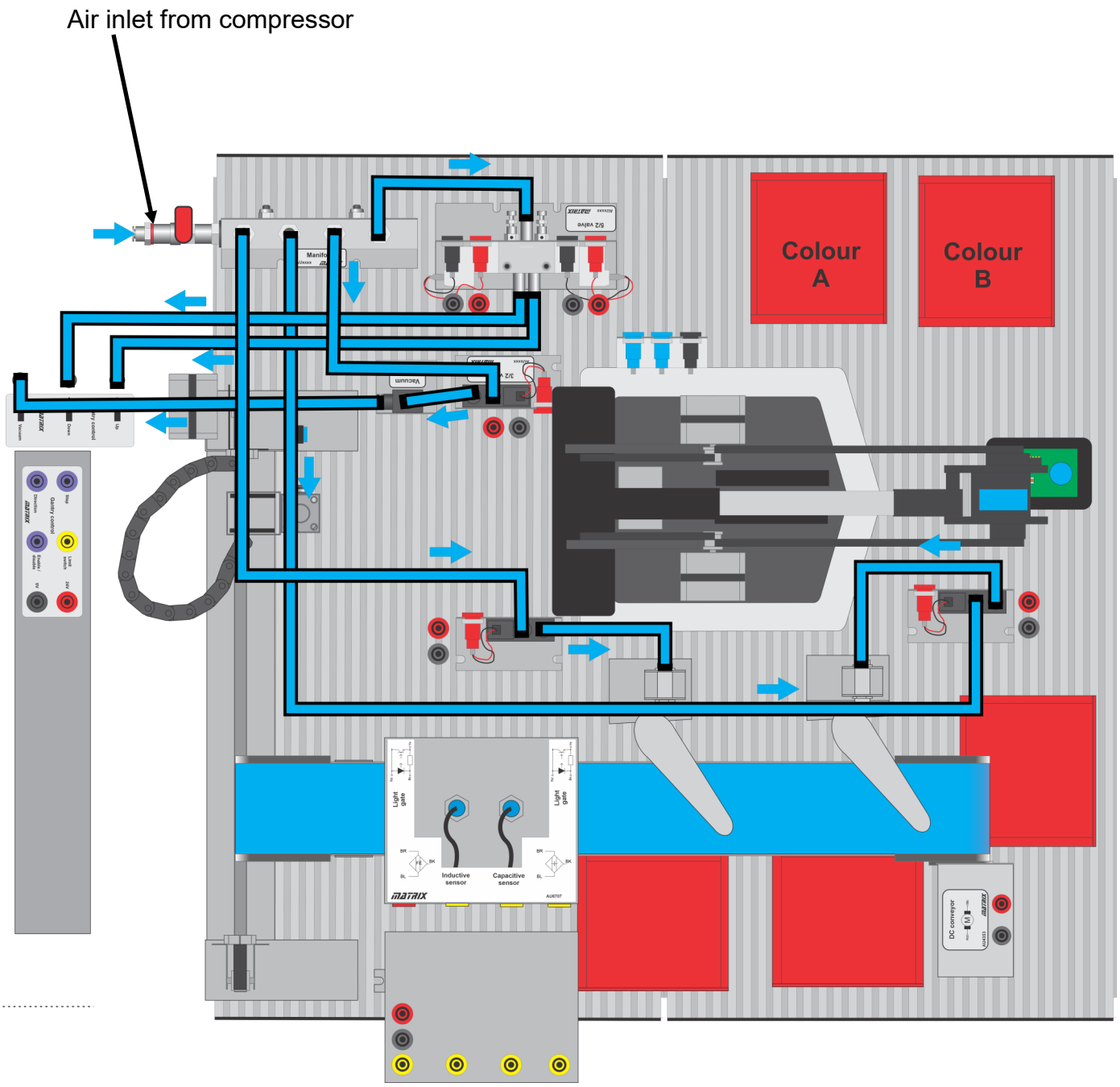


Wiring Diagrams

AU0205 Wiring Diagram

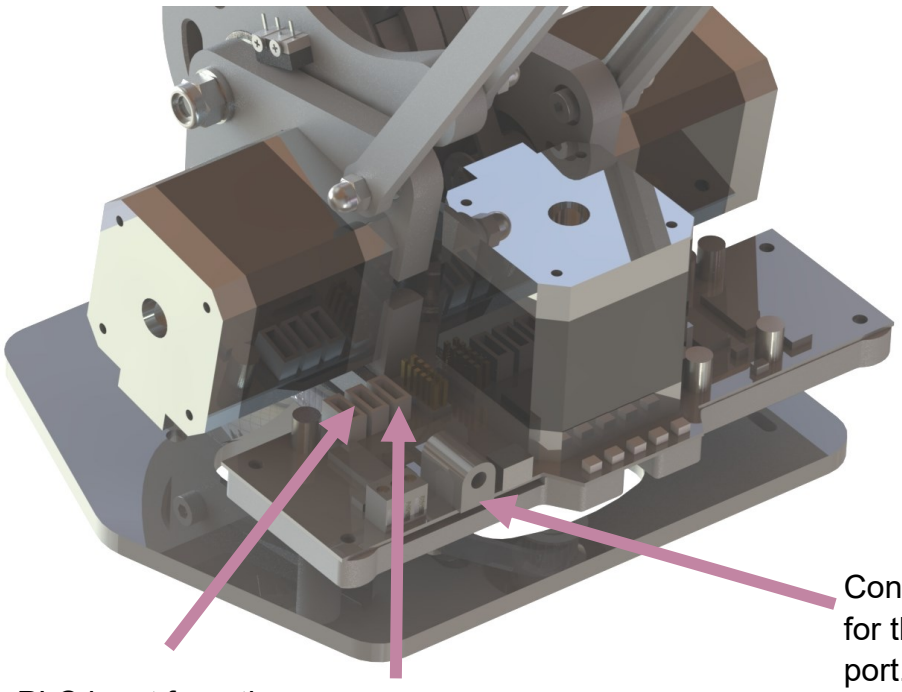


Pneumatic Diagram



Robot Arm Setup

Wiring for the Robot Arm



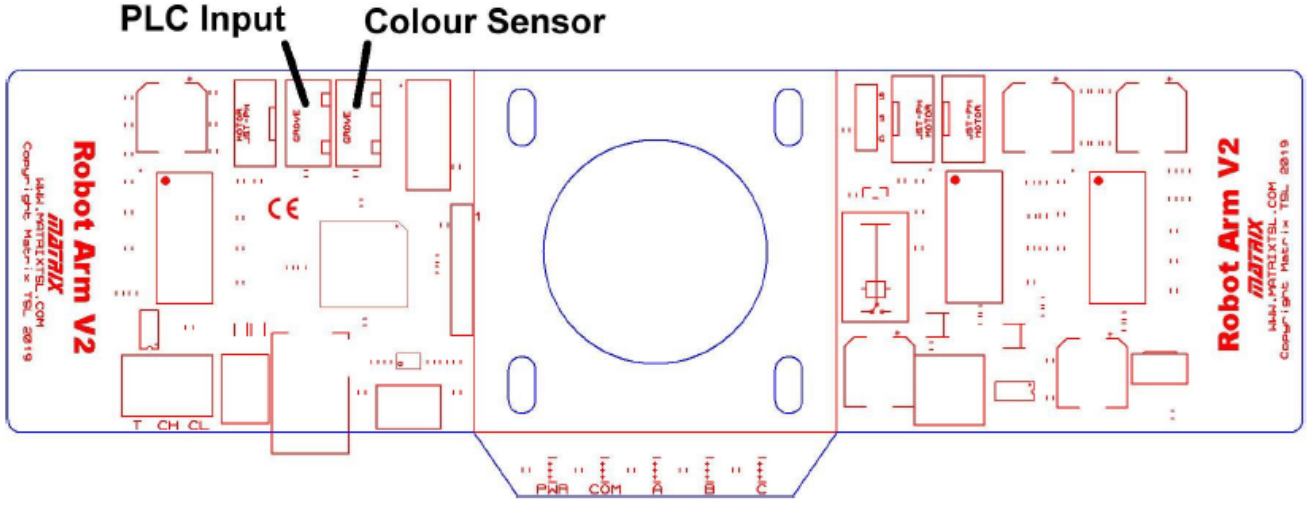
Connect the PLC input from the registration plate to the 2nd port of the 3 on the Robot Arm PCB.

Connect the light sensor ribbon cable from the 4 way JST connector on the light sensor to the 1st port on the Robot Arm PCB.

Connect the power supply for the robot arm to this port.

The USB port is next to this, if you are going to use a wired connection.

NOTE: Remember that the Robot arm runs of 12v system and has its own power supply



Functionality Test

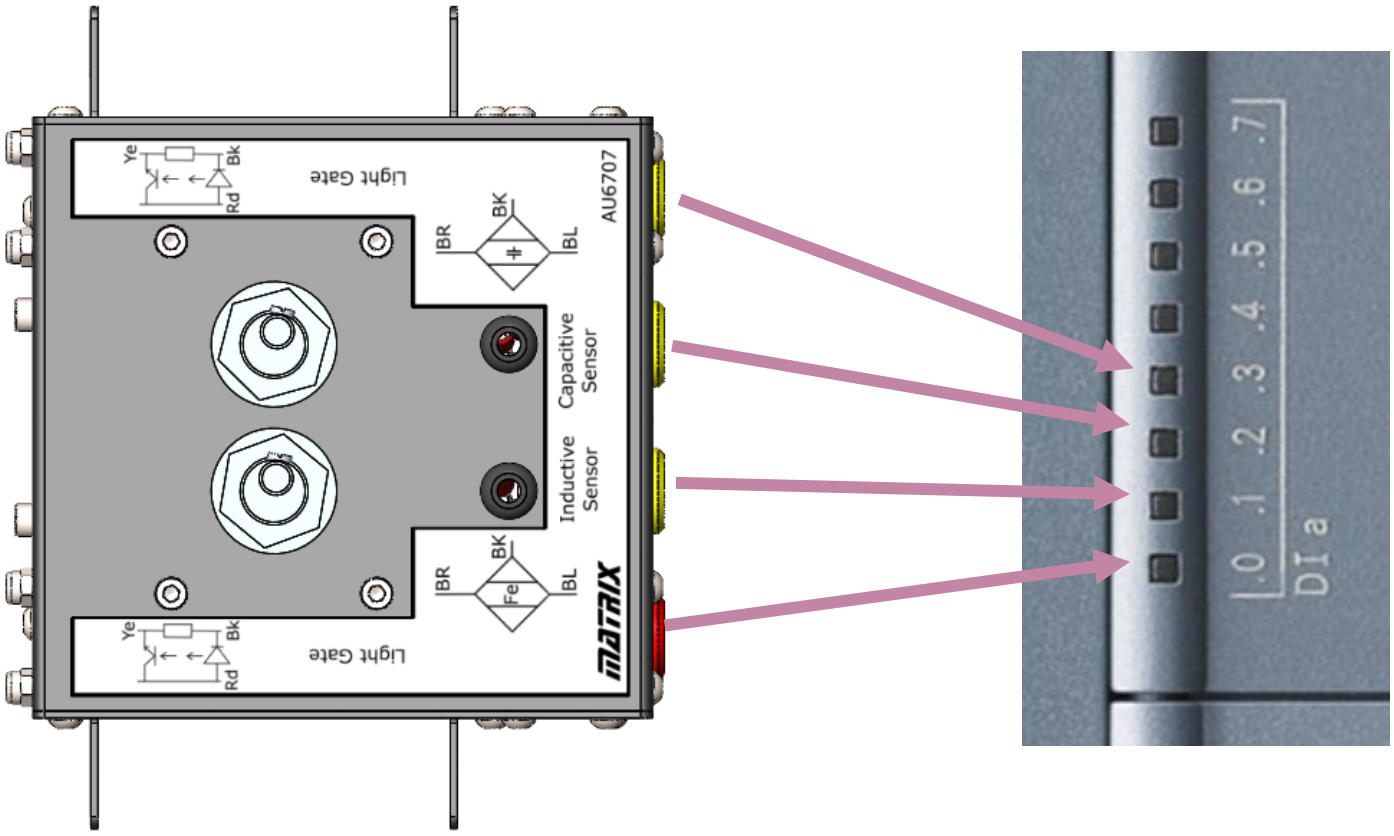
Step 1. Sensors

There are four sensors mounted above the conveyor, two light gates, an inductive sensor and a capacitive sensor. The PLC has a row of LEDs on the front that will light when each sensor is active. On the top row of LEDs, locate the group of eight that are labelled “DI a” With no counters on the conveyor, all these LEDs should be off. Place a counter on the conveyor and slide it under the first sensor. When the counter breaks the light beam, the LED labelled “.0” should light.

The next sensor is the inductive sensor. The “.1” LED should light when a steel counter passes underneath it. Usually, an LED on the body of the sensor lights at the same time. The sensor should not activate when either aluminium or plastic counters are under the sensor. The sensor is fixed to the mounting bracket with nuts above and below. The sensitivity can be adjusted by moving the sensor closer to the conveyor belt or further away.

Next the capacitive sensor has a tiny pot dial on the barrel, which affects sensitivity. The “.2” LED should light only when a steel or aluminium counter passes underneath it. This sensor can also be adjusted by changing the height of the sensor above the conveyor.

The final sensor is another light gate. The “.3” LED should light as the counter passes through the light beam.



Step 2. Stepper Homing Switch

A small micro-switch is mounted on the gantry close to the stepper motor. The PLC uses this switch to detect the home position of the gantry plunger mechanism. Pushing this switch by hand should cause the “0.4” LED to light.

Step 3. Reject Mechanisms

Before testing the reject mechanisms, check that all the pneumatic tubing is correctly installed and pushed fully into each connection point. Set the air compressor to a pressure of 3 bar and turn on the inlet tap of the manifold.

For each reject mechanism in turn, unplug its cable from the PLC output and connect it instead to one of the red +24VDC outputs. As soon as the cable is plugged in, the paddle should extend across the conveyor. Unplug the cable from the +24VDC output and the paddle should return to its home position. After this test, replace the cable back into the PLC output.

Step 4. Plunger

Check the pneumatic tubing and turn on the air supply to test the plunger. The plunger that is mounted on the gantry is controlled by a 3/2 valve and a 5/2 valve.

To test the vacuum, unplug the cable of the 3/2 valve from the PLC output and connect it to +24VDC. The 3/2 valve should operate and a flow of air will be heard exhausting from the vacuum generator. A counter placed under the rubber cup of the plunger should be held in place by the vacuum. When the cable is removed from +24VDC and replaced in the PLC output, this counter should fall.

The 5/2 valve has restrictors on its exhaust ports. As air is admitted into one side of the cylinder, air from the other side is expelled through the restrictor. To begin with these should be adjusted almost fully in, this will slow down the operation of the cylinder.

Slide the plunger along the gantry so that it is above the conveyor. It will be necessary to remove the +24VDC connection from the stepper input of the gantry so that the gantry can be moved by hand. Replace the +24VDC connection once it is positioned.

Remove the cable of the ‘down’ of the 5/2 valve from the PLC output to +24VDC and then back to the PLC output. The plunger should move down. Repeat this with the cable of the ‘up’ side and check that the plunger moves up.

With the restrictors wound fully in, the movement of the plunger will be sluggish and may not come down far enough to reach the counter rack. Adjust the restrictors by small increments and repeat the test until the movement is smooth but not too fast. There are lock nuts on the restrictors, tighten these once adjustment is complete.

Step 5. Conveyor

To test the conveyor, unplug its cable from the PLC output and connect it to one of the +24VDC outputs. The belt should run smoothly, traveling away from the gantry towards the reject mechanisms. After this test, replace the cable back into the PLC output.

Step 6. Gantry

The stepper driver requires a pulse to move the motor. This is not easy to do without using the PLC. For this reason, testing of the gantry is limited to visually checking the connections. Check that the plunger can be moved by hand when the power is off and that it moves smoothly.

Loading Software on AU3686

PLC I/O	Connector	Function
%I0.0	Yellow 0	First Light Gate
%I0.1	Yellow 1	Inductive Sensor (Detect Steel)
%I0.2	Yellow 2	Capacitive Sensor (Detect Aluminium)
%I0.3	Yellow 3	Final Light Gate
%I0.4	Yellow 4	Gantry Home Switch
%I0.5	Yellow 5	Trigger program, Bin 1
%I0.6	Yellow 6	Trigger program, Bin 2
%I0.7	Yellow 7	Trigger program, Bin 3
%Q0.0	Left-Hand Blue 0	Gantry Step
%Q0.1	Left-Hand Blue 1	Gantry Direction
%Q0.3	Left-Hand Blue 3	Plunger Down
%Q0.4	Left-Hand Blue 4	Plunger Up
%Q0.5	Left-Hand Blue 5	Plunger Vacuum
%Q0.6	Left-Hand Blue 6	Conveyor Motor
%Q1.0	Right-Hand Blue 0	Steel Reject Paddle
%Q1.1	Right-Hand Blue 1	Aluminium Reject Paddle

Use an Ethernet cable to connect the PLC to the computer that will be used for programming. Either connect the PLC directly to the Ethernet port of the computer or connect both devices to an Ethernet hub. It is best to use an authentic Ethernet port on the computer rather than a USB-to-Ethernet convertor because many convertors do not fully implement all the features that are needed for connecting to the PLC.

Programming the PLC is performed using the TIA portal software. This is available from Matrix or from a local Siemens distributer. The demo programs are written using TIA Portal V15 and are compatible with any version above this.

Demonstration programs are available from the Matrix website: -

https://www.matrixtsl.com/automatics/resources/

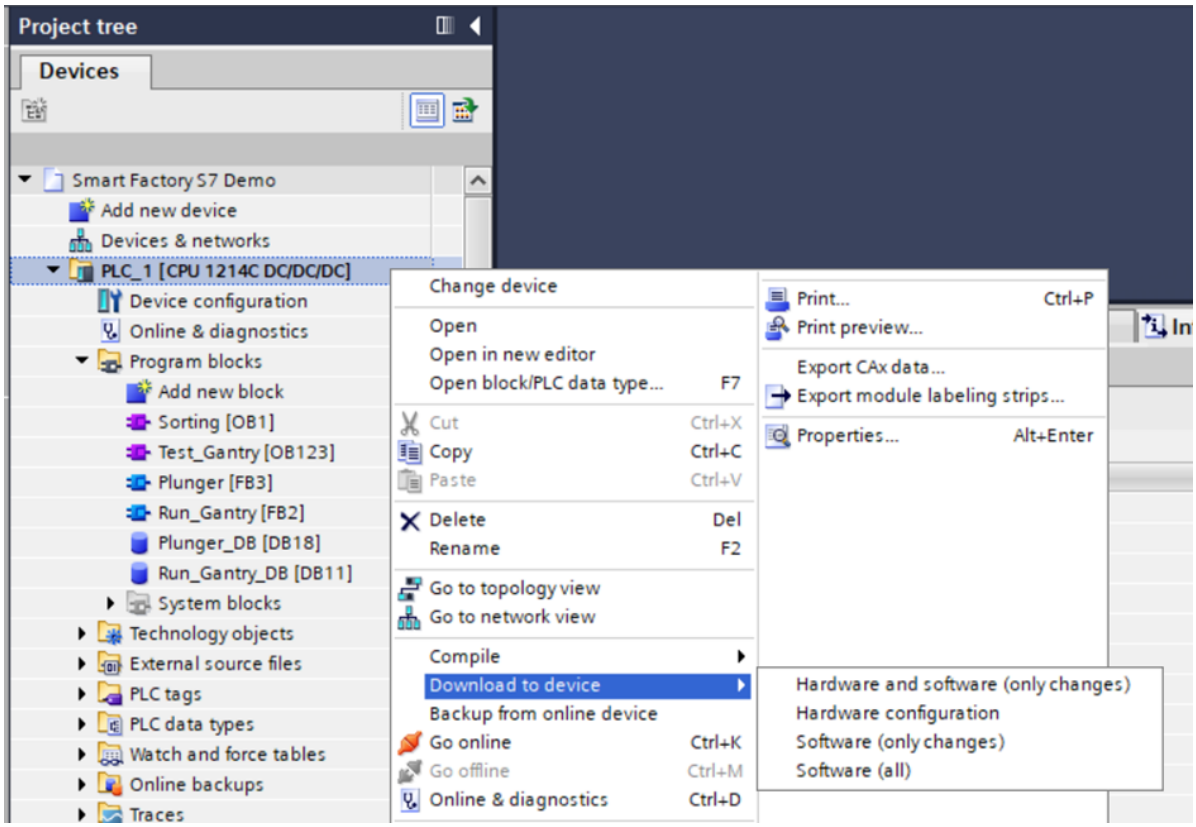
The appropriate program for the stand alone PLC is “Smart Factory Siemens S7-1200.zap15” Download the programs from the website and unzip them into a folder on the local computer.

Start TIA Portal, it will open in “Portal View”, click on the label “Project View” at the bottom left of the window.

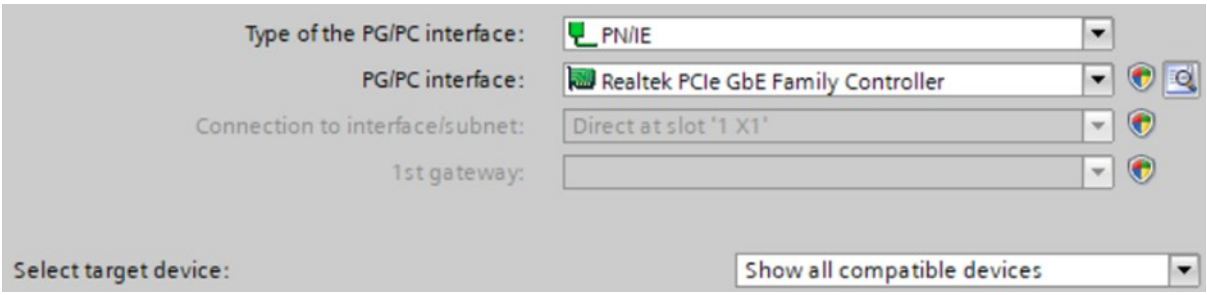
Select “Retrieve…” from the Project menu and browse to the location of the project archive. Once the file is selected, you will be prompted to choose a folder for the archive to be expanded into. Any folder on the local hard disk will work.

Loading Software on AU3686

Find the device “PLC_1” in the Project tree. Expand the device to examine the Program blocks. Right-click to download the project to the device. Both hardware configuration and software should be downloaded.



In the download window, check that the type of interface is “PN/IE”, the selected interface is the Ethernet interface of the local computer and the search option is “Show all compatible devices. Click “Start Search” to begin connecting to the PLC.



Once the PLC is found, click “Load” to download the configuration and software to the device. A dialog box will give options for what the PLC will do on completion of download. Change the option from ‘No Action’ to ‘Start’.

Operation with AU3686

The program is triggered by applying 24VDC to one of yellow inputs 5 thru 7. Briefly connect a cable from +24VDC to the input. It is also possible to install a momentary push button with one terminal connected to the input and the other to +24VDC.

When the input is triggered, the stepper motor will drive the plunger to the home position of the gantry. The PLC will hit the limit switch, back off and then hit the switch again to provide a precise home position.

The plunger will then be positioned over one of the bins in the hopper depending upon which input triggered the sequence. 5 being bin 1, 6 being bin 2 and 7 being bin 3. The plunger will extend, pick up a counter by vacuum and then retract. The counter will then be carried to the conveyor. The plunger will extend and drop the counter onto the conveyor by switching off the vacuum. The plunger then returns to the other end of the gantry.

The conveyor carries the counter under the rack of four sensors. All counters will trigger the first light gate. A steel counter will trigger the inductive sensor. Either steel or aluminium will trigger the second sensor. As the counters are identified, steel and aluminium counters are sorted into one of the two red bins by the reject paddles. Plastic counters are allowed to fall off the end of the conveyor into a red bin positioned at the end.

Loading Software on AU0205

PLC I/O	Connector	Function
%I0.0	Yellow 0	First Light Gate
%I0.1	Yellow 1	Inductive Sensor (Detect Steel)
%I0.2	Yellow 2	Capacitive Sensor (Detect Aluminium)
%I0.3	Yellow 3	Final Light Gate
%I0.4	Yellow 4	Gantry Home Switch
%I0.5	Yellow 5	Trigger pick from each bin in turn
%Q0.0	Blue 1	Steel Reject Paddle
%Q0.1	Blue 2	Aluminium Reject Paddle
%Q0.3	Blue 4	Plunger Down
%Q0.4	Blue 5	Plunger Up
%Q0.5	Blue 6	Plunger Vacuum
%Q0.6	Blue 7	Conveyor Motor
%Q1.0	Blue 9	Gantry Step
%Q1.1	Blue 10	Gantry Direction

The AU0205 product is supplied with demonstration software preloaded. If it is necessary to restore the demonstration software then it can be downloaded from: -

https://www.matrixtsl.com/automatics/resources/

The appropriate program for AU0205 is “AU0205 Smart Factory Demo_1.zap15” Download the programs from the website and unzip them into a folder on the local computer.

Start TIA Portal, it will open in “Portal View”, click on the label “Project View” at the bottom left of the window.

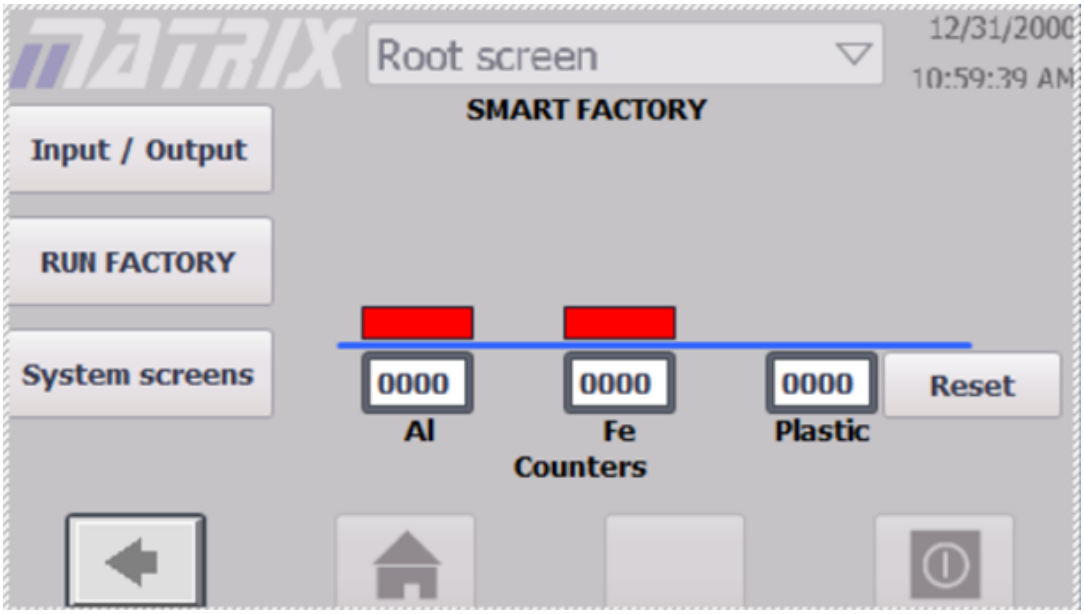
Select “Retrieve…” from the Project menu and browse to the location of the project archive. Once the file is selected, you will be prompted to choose a folder for the archive to be expanded into. Any folder on the local hard disk will work.

Connect the Ethernet cable of the controller to the computer that will be used for programming. Either connect the cable directly to the Ethernet port of the computer or connect both devices to an Ethernet hub. It is best to use an authentic Ethernet port on the computer rather than a USB-to-Ethernet convertor because many convertors do not fully implement all the features that are needed for connecting to the PLC.

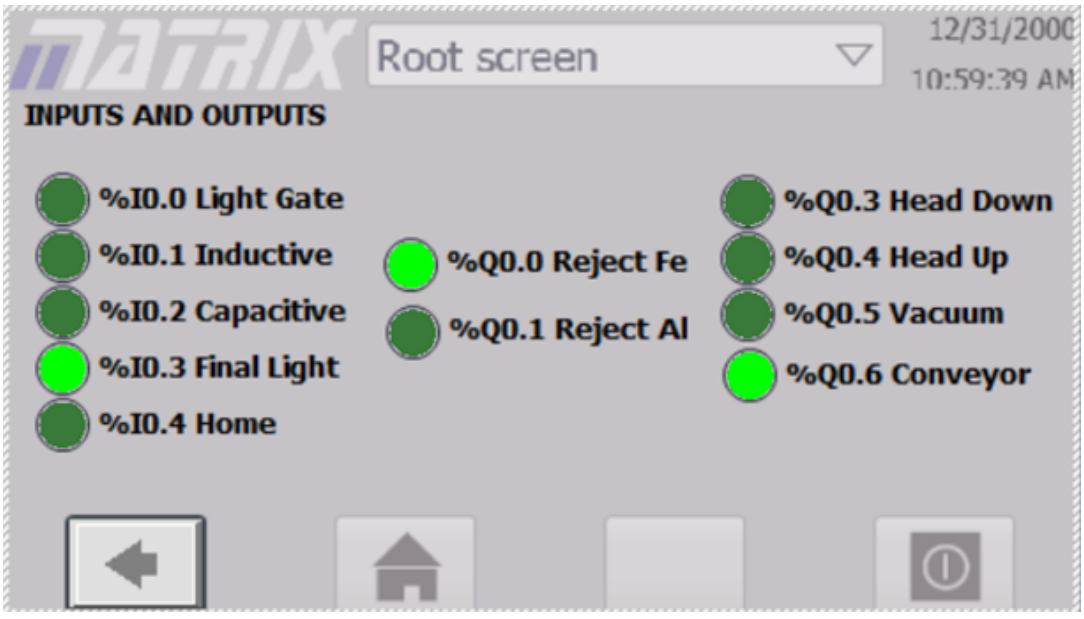
Find the devices “PLC_1” and “HMI_1”in the Project tree. Expand them to examine the demonstration program. Right-click PLC_1 to download the project to the device. Both hardware configuration and software should be downloaded. Right-click HMI_1 to download the software to the device. The instructions for downloading are described in the section on AU3686 above.

Operation with AU0205

When the device powers on the touchscreen HMI will show a start menu; touch the “Start” button to begin. At the next menu touch the “Smart Factory” button to start the demonstration. The first screen of the Smart Factory program shows tallies for the three categories of counters: Al = Aluminium, Fe = Steel and Plastic. These will normally show the numbers of counters since power on but they can be zeroed by touching the “Reset” button.



Touch the “Input/Output” button to see a display of the states of the PLC inputs and outputs. On this screen, the inputs and outputs of the PLC are represented as animations of LEDs. They will be coloured bright green when they are in the on state and dull green when in the off state.



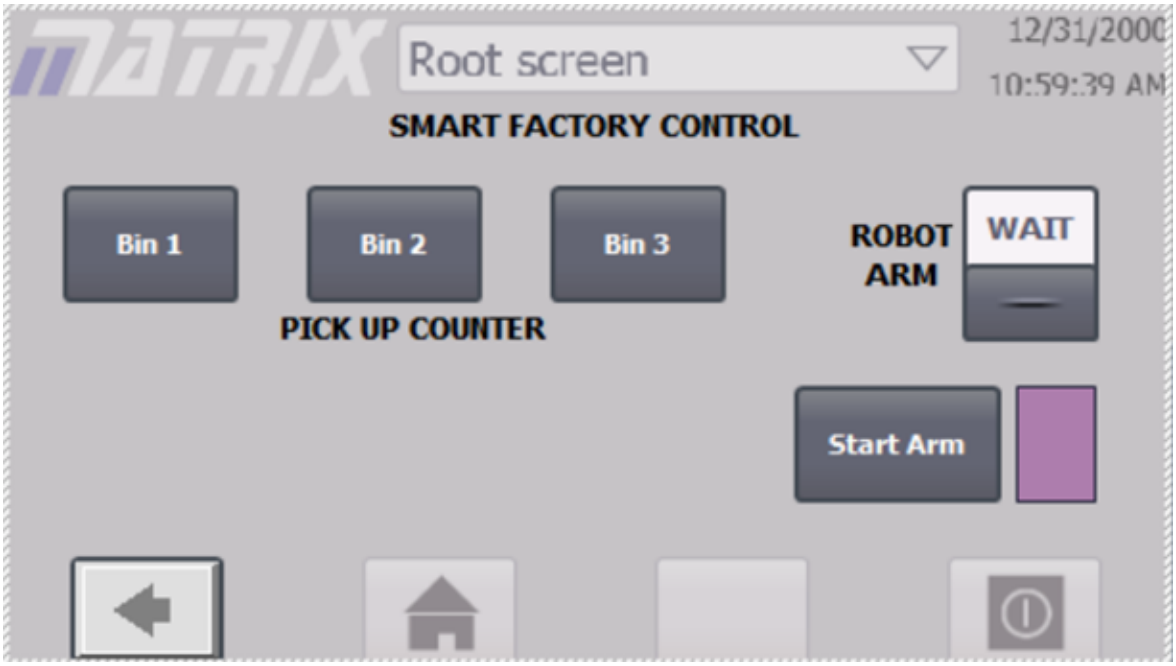
Operation with AU0205

Touch the back arrow at the bottom-left of the screen to return to the main screen and touch the “RUN FACTORY” button to load a screen that provides control over the Smart Factory. The buttons labelled Bin 1 thru Bin 3 will trigger the PLC to pick up a counter from the appropriate bin of the hopper and then deposit it on the conveyor for sorting.

On the right hand side of the screen is a switch that controls the behaviour of the PLC with respect to the robot arm. If a robot arm is installed with the factory, the switch should be in the “WAIT” state. In this state, the PLC will trigger the robot am to pick up any plastic counters, hold them to the colour sensor and then place them into the black and white sorting bins. If another counter is picked up by the gantry before the arm has completed its operation then the gantry will wait with the counter above the conveyor until the robot returns to its home position.

As well as being triggered automatically when a plastic counter is detected, the robot arm can be triggered by touching the “Start Arm” button.

If no robot arm is present, then the switch should be in the ‘IGNORE’ state.



Operation with AU0205

Webserver Connection

The S7 PLC contains a built in webserver that can be used for maintenance, diagnostics and user defined pages. The demonstration software includes webpages that can be used to see tallies of the counters and to trigger the pick and sort routine. The router in the AU0205 provides a wireless network that can be connected to a mobile phone or laptop computer. Find the network named “Matrix_AU0205” and log on with the password “Matrix123” Alternatively, a computer can be connected using the Ethernet cable.

To connect to the PLC, start a web browser and type in the address: -

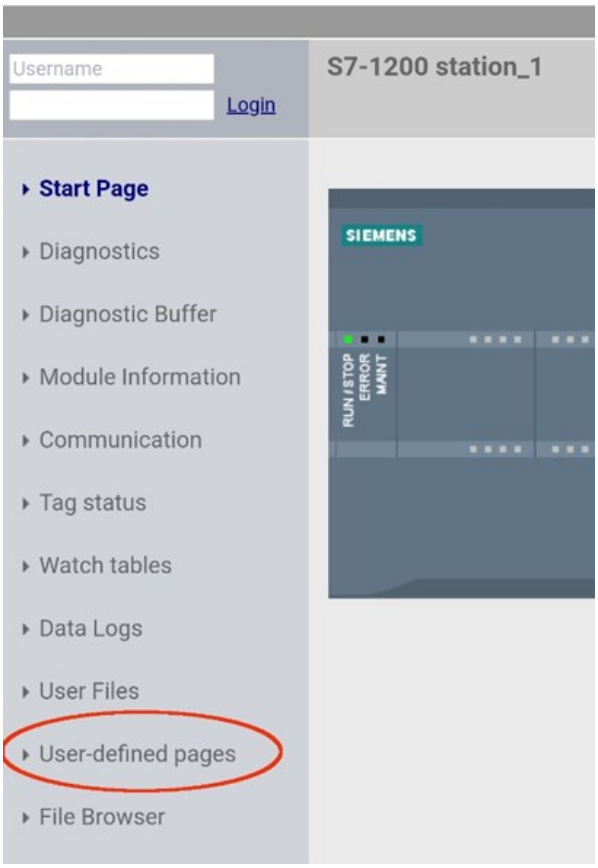
http://192.168.7.2

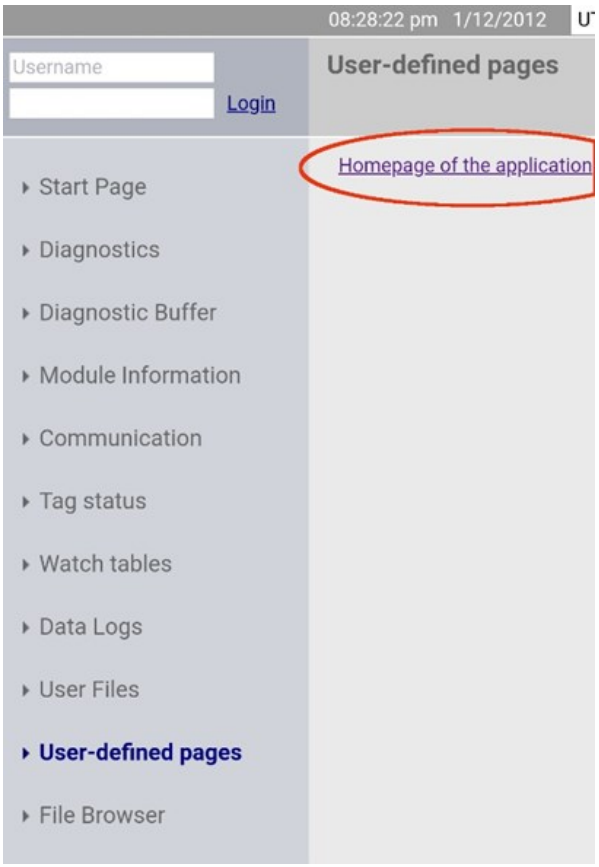
This is the IP address of the PLC on the local network that the router is producing. Note that not all web browsers will recognise an IP address. In some cases the web browser will misinterpret the address as a search term. If this happens, try a different browser application.

When the front page of the PLC appears, click or touch the “ENTER” link.

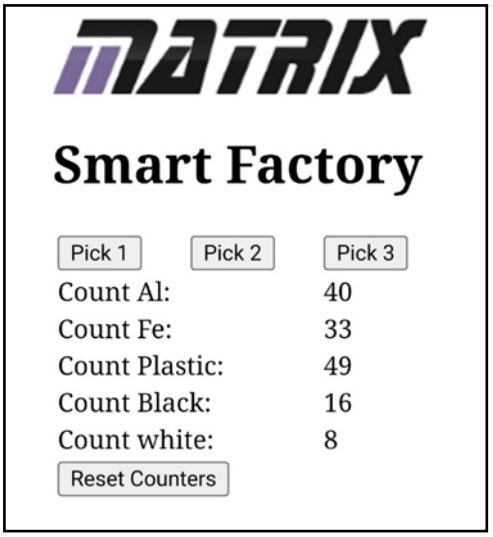
Operation with AU0205

On the next page click on the “User-defined-pages” and then “Homepage of the application” links.





The user defined page for the Smart Factory application shows tallies of the three categories of counters. It also splits the plastic category into black and white. The “Pick 1” thru “Pick 3” buttons will trigger the pick and sort routine in the same way as the HMI buttons and the “Reset Counters” button will zero the tallies.



Getting Started with Robot Arm

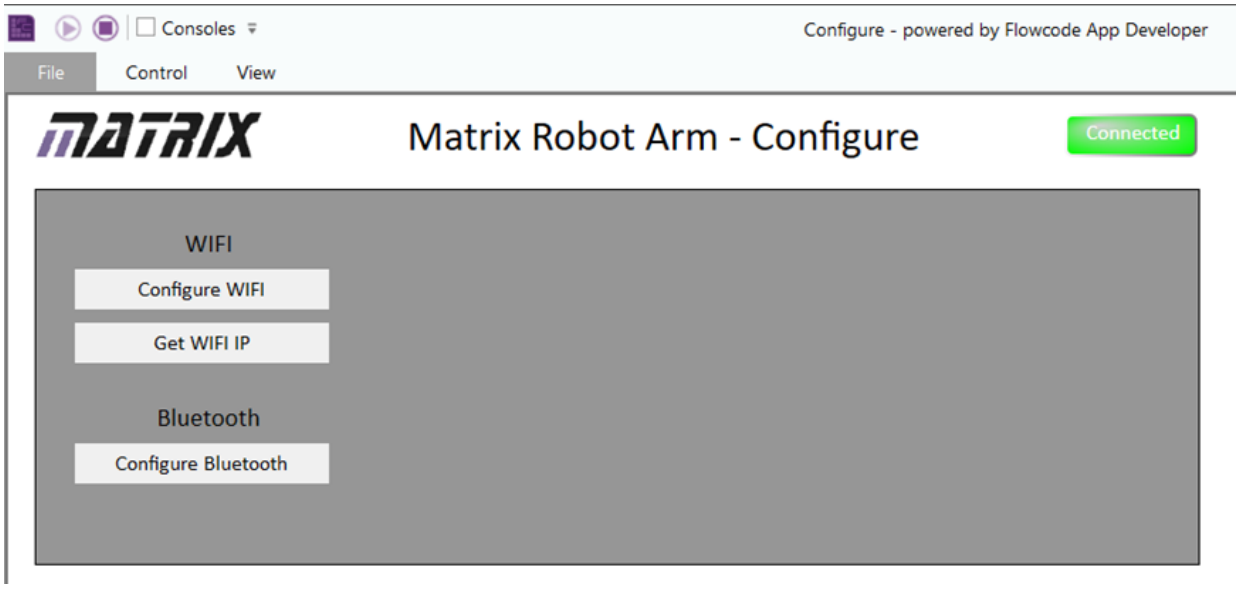
The AU0205 product contains a built in router that connects to the PLC by Ethernet and can connect to the robot arm by Wi-Fi. There are two steps to creating a connection. The first is to configure the robot to connect to the router and the second is to configure the router to assign a fixed address to the robot.

Configure Robot Arm Wi-Fi

To configure the robot, download the Robot Arm V2 Control Apps applications from:

https://www.matrixtsl.com/allcode/resources/

Connect the robot arm to the computer using a USB cable and apply power to the robot. Run the Configure.exe program. Press F5 or click on the run button at the top left of the screen and the “Connected” LED should turn green. Click on the “Configure WiFi” button to begin the configuration.



The application will prompt with a series of dialog boxes. Choose the following responses: -

WiFi Enabled? – Yes

Host Network? – No

WIFI SSID – “Matrix_AU0205”

WIFI Password – “Matrix123”

WIFI Port – 1245

After entering the responses, the software will ask you to power cycle the robot arm for the changes to take effect.

Configure Robot Arm IP Address

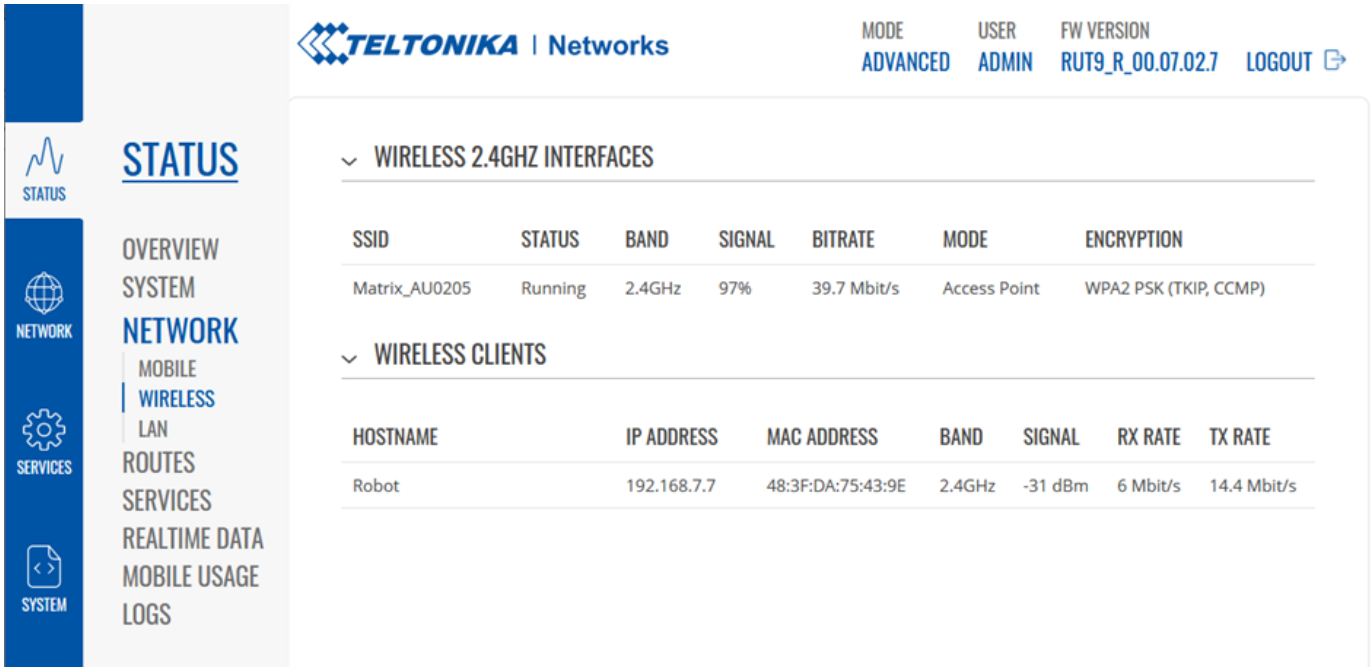
The router in the AU0205 contains a DHCP server that assigns addresses to components on the network.

Connect a computer to the router using either the Ethernet cable or the Wi Fi connection. Open a web browser and enter the address: -

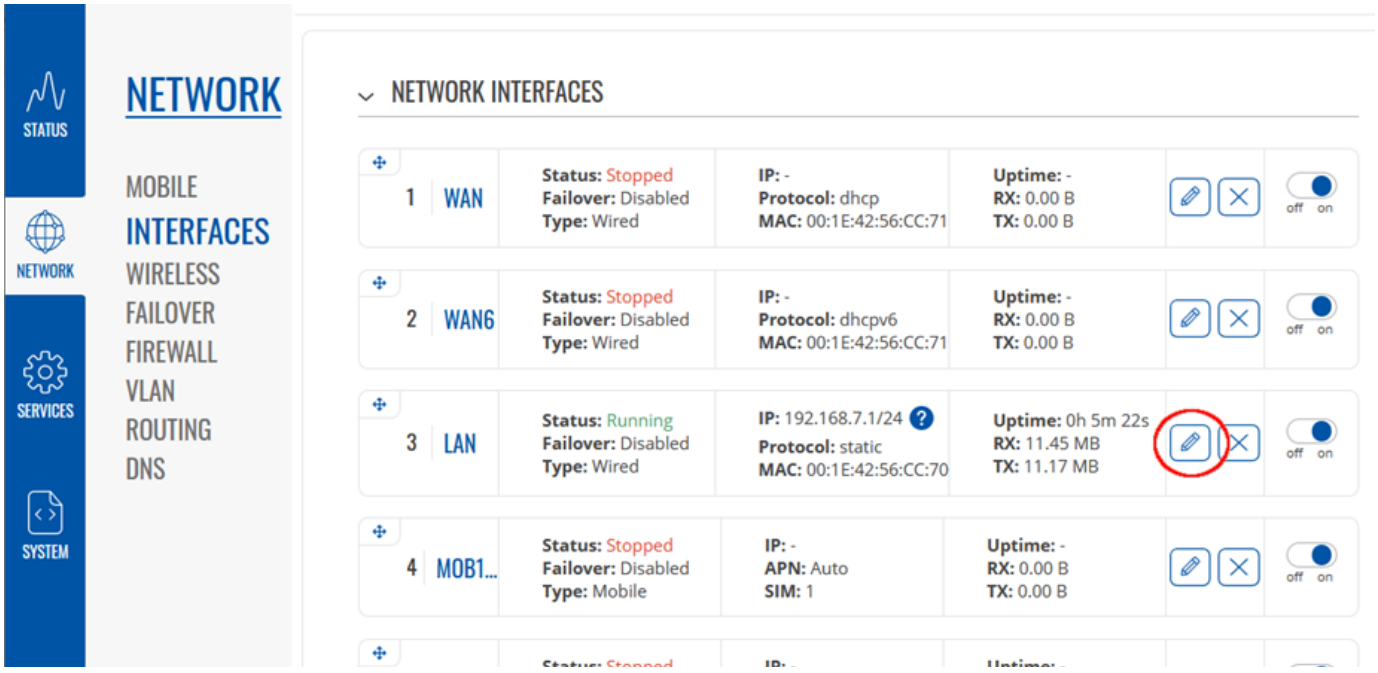
http://192.168.7.1

Login to the router using the username “admin” and the password “Admin123”

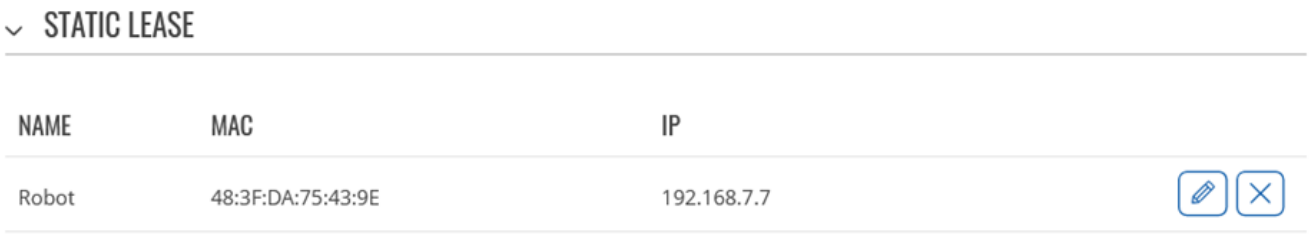
Look at the Wireless Clients section of the status page. If the robot has been configured correctly according to the previous section, it should appear in the list of clients. Make a note of the MAC address as this will be needed later.



Select the Network page by clicking on the icon on the left hand side of the screen. Select interfaces and click on the edit icon for the LAN interface.



In the properties for the LAN interface, edit the DHCP server and add a new static lease. Name the new instance “Robot”, enter the MAC address that was noted earlier and give it the IP address “192.168.7.7” This is the address that the demonstration software in the PLC uses to communicate with the robot arm.



Once the robot arm and the router are configured, power cycle the robot arm in order for the robot to obtain the new IP address from the router.

Getting Started with Robot Arm

Connecting the Robot Arm to AU3686

There is a demonstration program available for the AU3686 that connects to and controls the robot arm. Since the AU3686 product is not supplied with a router, it is necessary for customers to have a locally available router connecting to the PLC by Ethernet and the robot arm by Wi-Fi. Some networking expertise may be required to set this up. Follow the same steps as for configuring AU0205 but substitute the details of the local network.

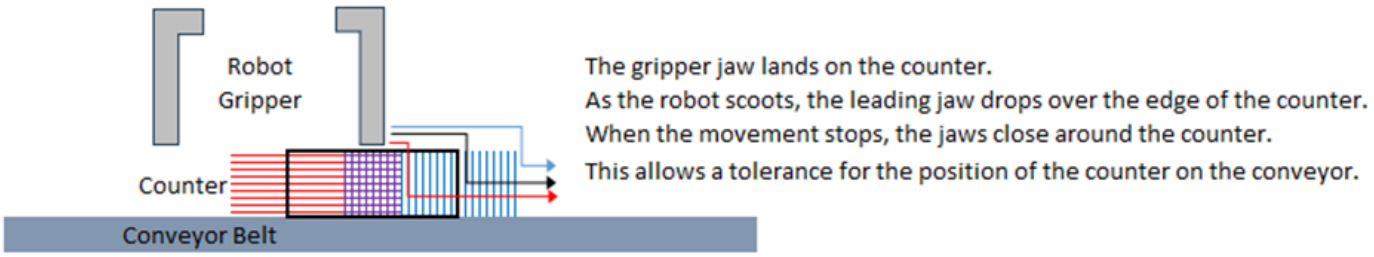
The demonstration software is “Smart Factory with Robot Demo.zap15” The software can be downloaded using TIA Portal using the procedure described in the section on AU3686. The IP address is coded in the function block “Robot_Arm_Comms” The default is 192.168.1.63 but it can be changed to suit the local network. The program will begin a sequence of picking three counters, one from each bin of the hopper, when input 5 is connected to +24VDC.

Getting Started with Robot Arm

In the demonstration programs, the robot arm is commanded to progress through a sequence of moves. These carry the plastic counter from the conveyor to the colour sensor. After reading the colour sensor, the counter is placed in either the white or black sorting bin and the robot returns to its home position.

The exact positions that the robot arm moves through are dependent on the layout of the Smart Factory. For the robot to work as intended in the demonstration programs, it will be necessary to make fine adjustments. There are two approaches that can be used. One is to carefully move the robot arm base, the colour sensor and the sorting bins until the program works as written. The better approach is the Robot Arm V2 Control Apps that were downloaded earlier and use the BasicControl application to find the appropriate positions. The step counts for each motor can then be edited in the demonstration programs before re-downloading them to the PLC.

Stopping the plastic counter on the conveyor relies on timing how long the conveyor motor is run. This leads to small variations in the exact stopping position. The program is designed to bring the robot arm down so that one jaw of the gripper lands over the centre of the counter. The robot then scoots sideways so that the jaw of the gripper slips of the edge of the counter before closing the gripper.



Getting Started with Robot Arm

The positions of the robot that are programmed into the demonstration programs are listed in the following table.

Instruction Number	Description	Step Counts		
		Motor 0	Motor 1	Motor 2
		Base	Shoulder	Elbow
0	Home all motors			
1	Position above conveyor	260	700	200
2	Land one jaw on counter	260	860	900
3	Scoot across counter	360	860	900
4	Close gripper			
5	Lift above conveyor	360	700	200
6	Rotate above sensor	2625	700	200
7	Lower onto sensor	2625	1360	1520
8	Read colour sensor			
9	Lift above sensor	2625	700	200
10-A	In mouth of black bin	3120	1200	1000
10-B	In mouth of white bin	3600	900	1000
11	Open gripper to drop			
12	Position above bins	3200	700	200
13	Home all motors			

Introduction to Programming

In order for the PLC to interact with the robot arm, they must both be part of a local area network with the robot connected by Wi-Fi and the PLC by Ethernet. An alternative way to control the robot arm is to write a FlowCode program to control the robot. Documentation for FlowCode and Robot Arm programming can be found on the Matrix website.

The blue sockets on the base of the robot arm can be connected to outputs of the PLC. The PLC program could be modified to use one of its outputs to signal that a plastic counter is ready to be collected. A FlowCode program could be written to wait for an input before running through the sequence of picking up and sorting a counter.

Getting Started with TIA Portal

A major goal of the Smart Factory is to provide a platform for teaching PLC programming. The PLC in the AU3686 and AU0205 products is made by Siemens and can be programmed using the TIA Portal software. This package incudes all the tools needed to configure and program the PLC and to design screens for the HMI if included.

Framework

The demonstration programs available on the Matrix website include “Smart Factory Framework.zap15” This is a project framework that includes the target PLC and has most of the low level configuration completed along with some low level functions for communicating with the robot arm. The user can add higher level control functions to the framework without needing to spend time on the lower level parts.

To use the framework, start TIA Portal, it will open in “Portal View”, click on the label “Project View” at the bottom left of the window. Select “Retrieve…” from the Project menu and browse to the location of the project archive. Once the file is selected, you will be prompted to choose a folder for the archive to be expanded into. Any folder on the local hard disk will work. Save the retrieved framework as the project name of your choice.

Introduction to Programming

Project

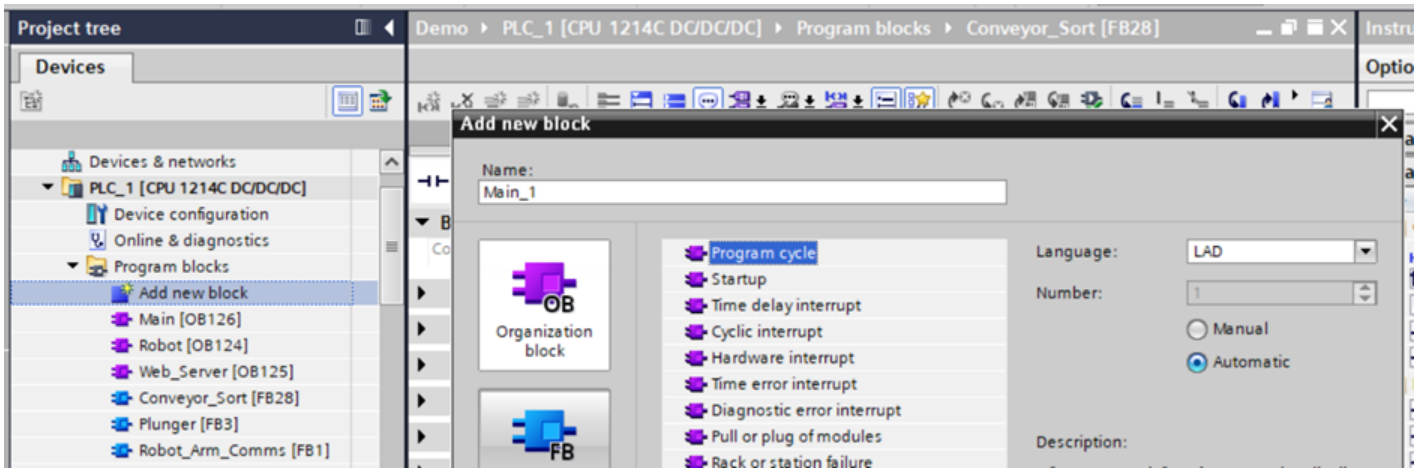
If not using the framework, create a new project in TIA portal. When the software opens in the portal view, click on “Create New Project.” Fill in details of the project and choose a location on the local hard drive to save the files, then click “Create.” Click on “Configure a Device” and then “Add new Device.”

Add a controller from the SIMATIC S7-1200 family. The device is CPU 1214C DC/DC/DC and its article number is 6ES7 214-1AG40-0XB0. If using the AU0205 then return to the portal view and add an HMI from the SIMATIC Basic family. The device is a 4” Display, type KTP400 Basic and its article number is 6AV2 123-2DB03-0AX0.

The added devices will appear in the Project tree on the left hand side of the Project View screen.

Basic I/O

The PLC in the Project tree contains a folder named “Program Blocks.” Click on “Add new block” to create a new program. The simplest type of program is an “Organization block” that runs on the program cycle. Such a program will run continuously.



Set the language to “LAD” to write the program in ladder logic.

With the new program open, click on the “Instructions” tab to open a toolbox of ladder logic instructions. The toolbox is context specific so it will contain items that are relevant to whatever editor window is active.

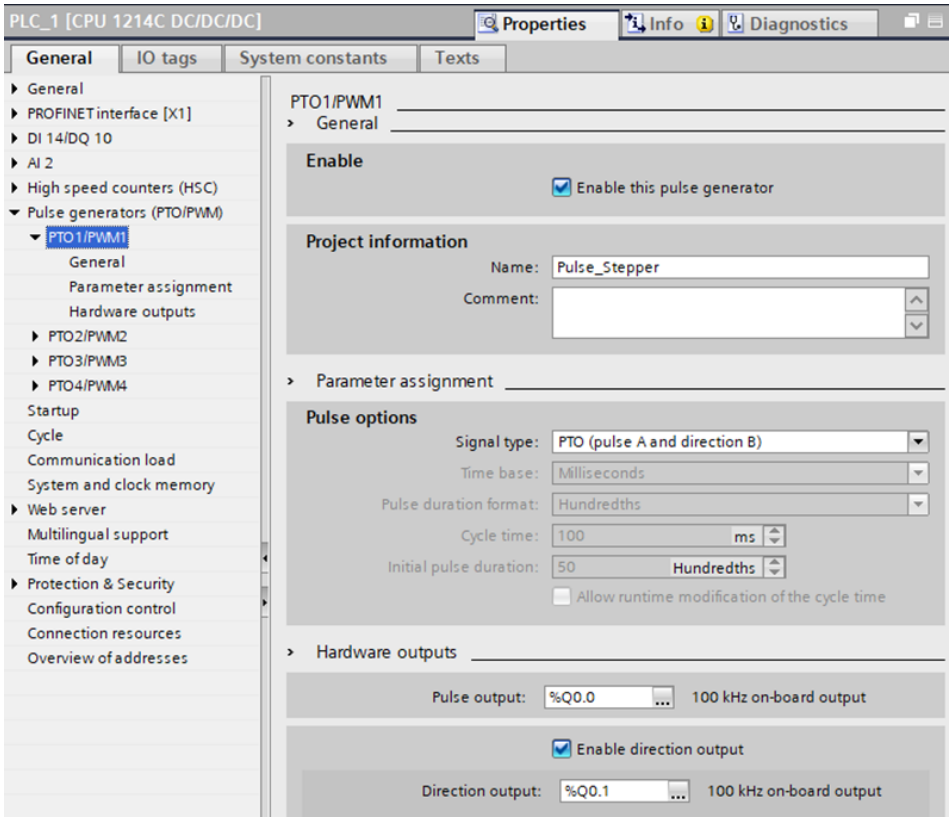
The PLC inputs are connected to yellow 4 mm sockets. They can be used as contacts in a program by using the addresses %I0.0 thru %I0.7. The AU0205 product has two more inputs that are accessed by addresses %I1.0 and%I1.1.

The PLC outputs are connected to blue 4 mm sockets. They can be accessed in a program by using the addresses %Q0.0 thru %Q0.7, %Q1.0 and %Q1.1. For example; to cause output number six to switch on whenever input number six is asserted, connect a normally open contact to an assignment. Give the two elements the addresses %I0.6 and %Q0.6.

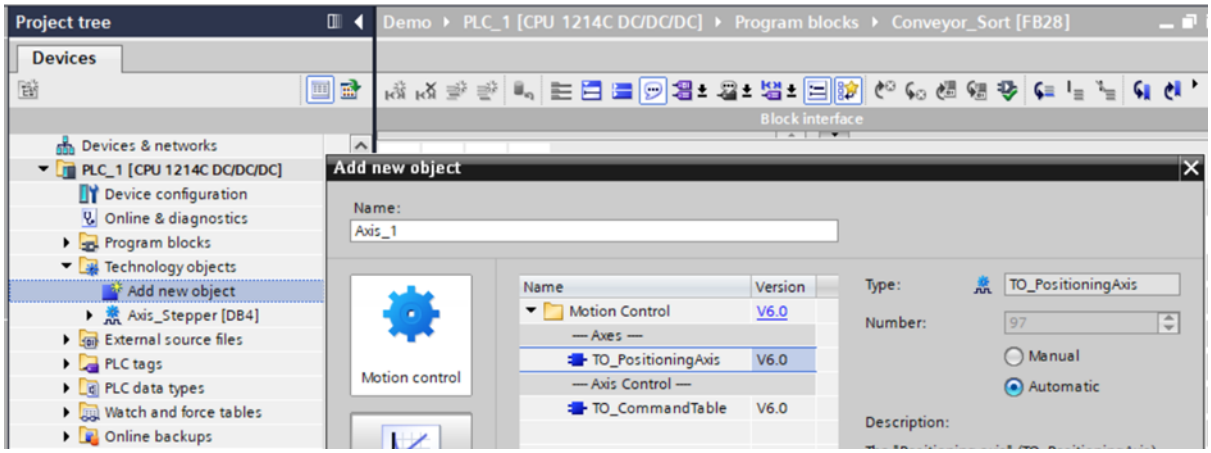
Motion Control

The stepper driver in the gantry is controlled by a pulse input and a level input. The stepper motor advances one step every time the “Step” input pulses. It moves towards its home position when the “Direction” input is not asserted and moves away from home the input is asserted.

While it would be possible to write a program that switched the step output on and off a number of times, there is a better way. The PLC contains a pulse generator that can be configured in its properties. Outputs of the PLC can be dedicated as pulse and direction outputs. Refer to the example projects and framework to see how to configure this.



Once configured, the pulse generator can be assigned to a Motion Control Object. In the Project tree, the PLC contains a folder named Technology objects. Click on “Add new object” and choose “TO_PositioningAxis”



Refer to the example projects and framework to see how to configure the object. Once configured, motion control commands will be available in the Instructions toolbox under the heading Technology.

Robot Arm

The robot arm is commanded using TCP communication. The instruction blocks that are used for this are found in the Communication section of the instruction toolbox under the Open user communication section. Refer to the example programs to see how these blocks are used. The robot arm has an application programming interface. Refer to the documentation of the robot arm product for details.

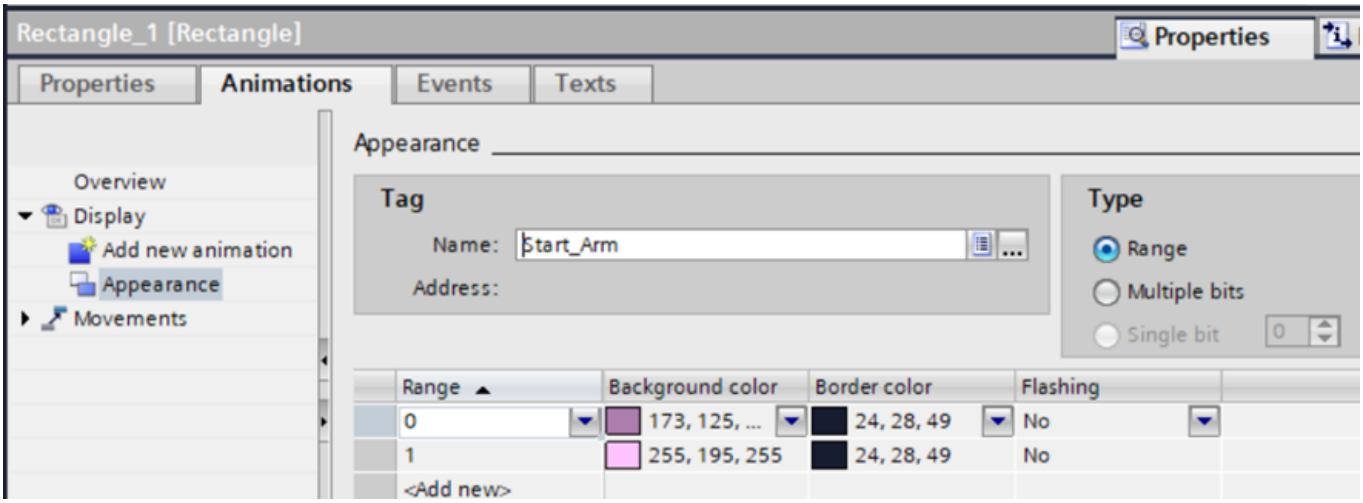
HMI Screens

Once added to the project, the HMI screen will appear in the Project tree. It will contain a folder named Screens and this folder includes the action “Add new screen.” When editing a screen, the toolbox will contain Basic objects and elements that can be added to the screen. The first screen to be shown when the system starts up can be chosen by right clicking that screen and choosing “Define as start screen.” The start screen is highlighted in the project tree by a small green arrow overlayed on its icon.

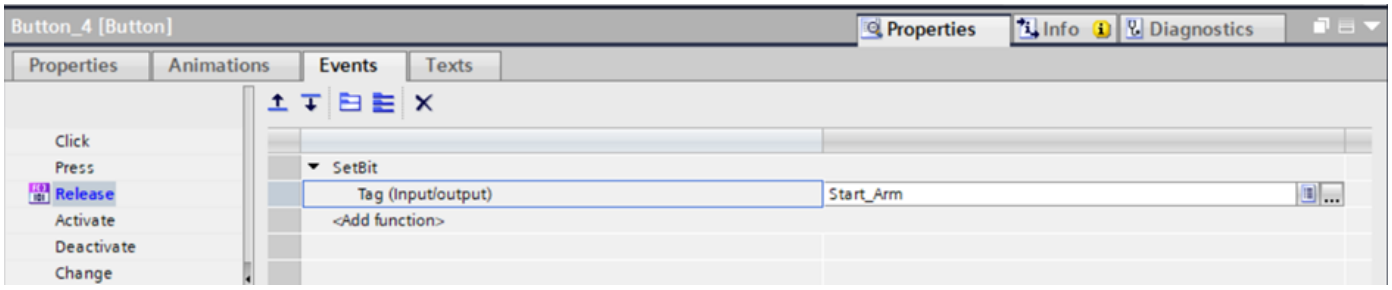
The properties of an object can include “Animations” that indicate the conditions of tags. These tags can be linked to tags in the PLC so that information about the running program is conveyed to the user.

The properties of an element can include events that change tags when the user interacts with the element on the screen. These tags can be linked to tags in the PLC so that the user can control the running program.

To add an indicator, drag a shape from the Basic objects toolbox onto a screen. Adjust the position of the object and resize it as required. In the properties for the shape, select the Animations tab. Click on “Add new animation” and select Appearance. Click on the ellipsis next to the tag name box to choose a tag to base the animation on. If a tag from the PLC is chosen then TIA Portal will automatically set up the required communication between the HMI and the PLC so that the tag is available in the HMI. A binary tag can be represented as a simple range type with different Background color for the zero and one states. If it is dull in the zero state and bright in the one state then it will appear to behave like an LED.



To add a control button, drag a button element from the toolbox onto a screen. Adjust position and size as required. In the properties for the element, select the Events tab. Choose the Release event and click on Add function. The drop down list that appears lists all the functions that can be configured to occur when the user touches and then releases the button. SetBit can be found under the Edit bits subheading. When the function is added, choose the tag that is to be set. TIA Portal will automatically set up the necessary communication with the PLC. If a switch element were chosen instead of a button then functions can be assigned to both Switch ON and Switch OFF events. SetBit and ResetBit would be logical choices here.



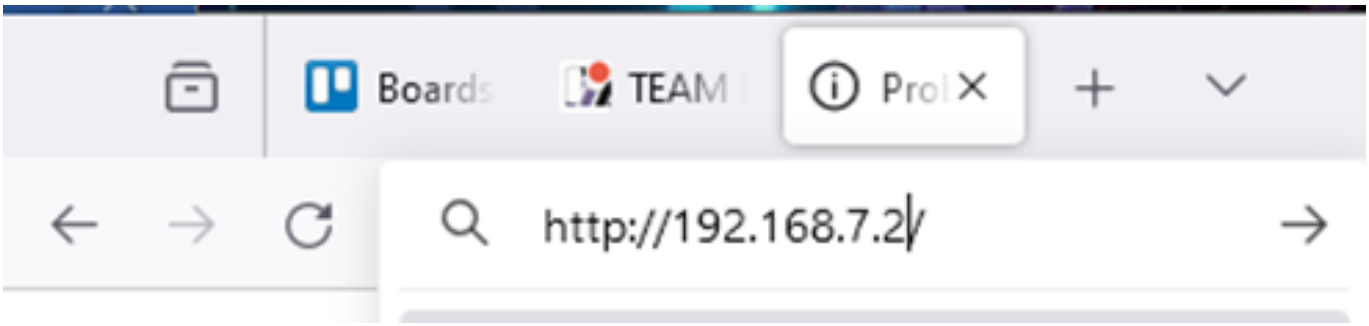
To add a navigation button, begin in the same way as adding a control button. Rather than setting a bit, add the ActivateScreen function to the button's Release event. When the user touches and then releases the button, the specified screen will be loaded.

Web server

The PLC contains a built-in webserver that can be invoked by adding a ”WWW” object from the toolbox to a program block. The WWW object can be found in the Communication section of the toolbox.

Any files used by the webserver are included in the “Web Files” subfolder in the folder where the project is saved. These will include HTML pages, images, scripting libraries such as jQuerry and tables defining the tags that are accessible to the web server. The online documentation for TIA Portal outlines the process of compiling the web pages into data blocks that can be used by the web server. The example projects demonstrate how to control programs and report data using web pages.

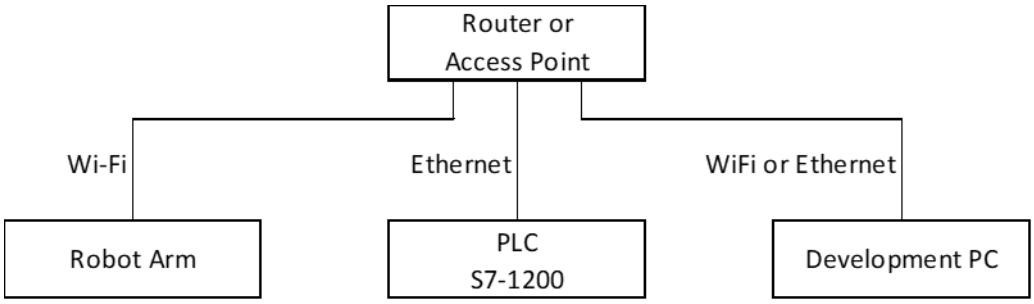
To access the web server, a computer or mobile phone must be connected to the same network as the PLC. The web pages can be viewed by typing “http://” into the address bar of a web browser followed by the IP address of the PLC.



Connecting the Robot Arm to the S7 using Wi-Fi

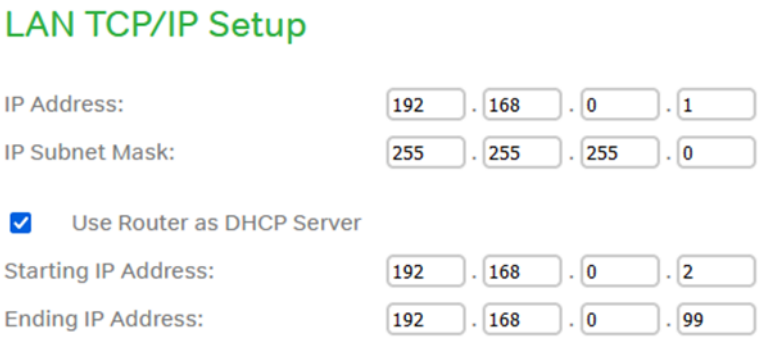
To get the Siemens PLC to work with the Robot Arm they must be on the same network. The Robot Arm has a Wi-Fi connection and the S7-1200 PLC has an Ethernet connection. To connect the two we need a wireless router. It is convenient if this is the same office or home router that is being used with the PC that will be used for programming.

In the absence of a home or office network an ad-hoc network could be created using a compact router such as TP-Link TL-WR802N



Static IP Address

The PLC requires a static IP address. This can be configured in your router or provided by your network administrator. Generally this is just a case of restricting the range of addresses available for DHCP. The image below shows the LAN setup page a typical home router. The ending IP address of the DHCP server has been set to 192.168.0.99. Any addresses above that value are available for use as static addresses.

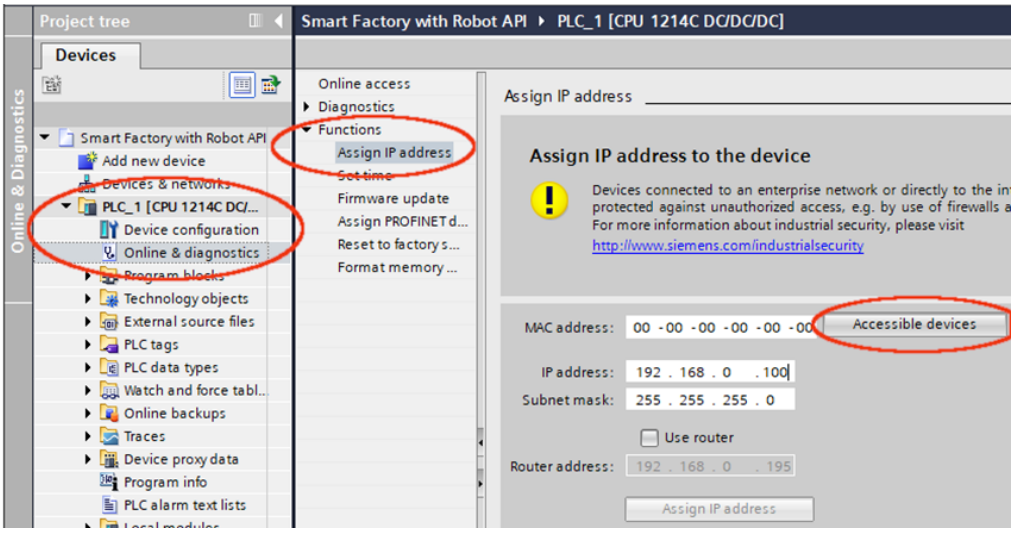


Network Address of PLC

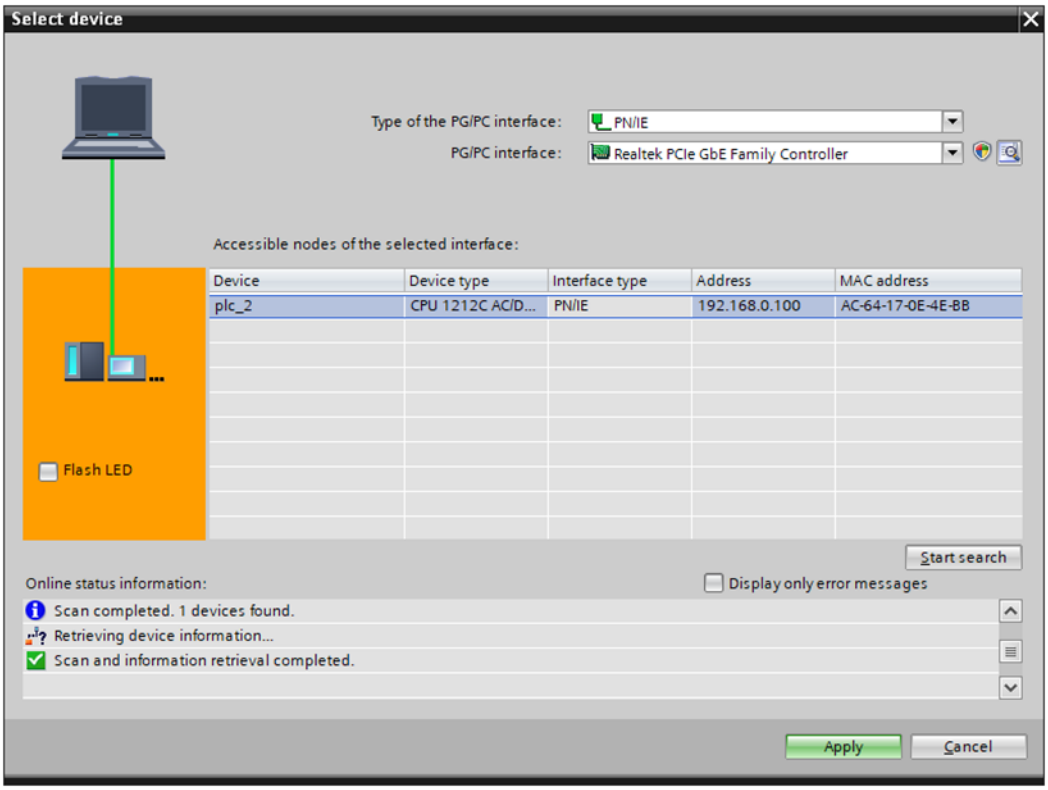
Begin by downloading the project framework or one of the example projects and opening it in the Siemens TIA Portal software.

To assign an address to the PLC, locate the PLC in the Devices tree and select Online and diagnostics. From the online panel choose Assign IP address. Before the IP address can be assigned, the software must find the PLC. Click on the Accessible devices button to scan the local network for the PLC.

Connecting the Robot Arm to the S7 using Wi-Fi



While scanning for accessible devices check that the Type of the interface is “PN/IE” and that the interface is the one on the PC that is physically connected to the network. Click on the Start Search button to find the PLC. Once it is found, clicking the Flash LED check box will make the RUN/STOP ERROR and MAINT LEDs on the PLC flash. This can be helpful if there is more than one PLC on the network as it ensures that we are connecting to the right one.

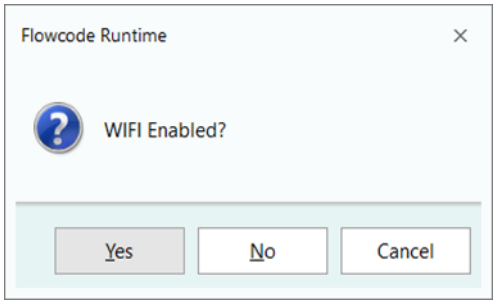


After finding the PLC, click on the Apply button to go back to the Assign IP address screen and click on the Assign IP address button to set the address.

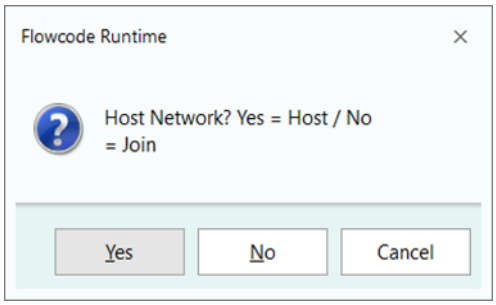
Connecting the Robot Arm to the S7 using Wi-Fi

Network Address of Robot Arm

The Wi-Fi networking of the robot arm can be configured using the Robot Arm V2 Control Apps that are available from the Matrix website. First connect the robot arm to a PC using a USB cable. Open the Configure App and select USB as the Comms Method. Click the ‘Go’ button in the top-left corner to start the app and click the Configure WIFI button. A series of dialog boxes guide the user through the configuration process.



Choose ‘Yes’.



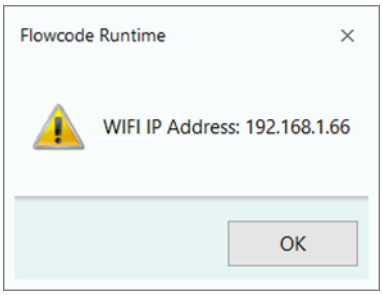
Choose ‘No’ to join the existing Wi-Fi network.

Enter the name of the Wi-Fi network that you wish to join.

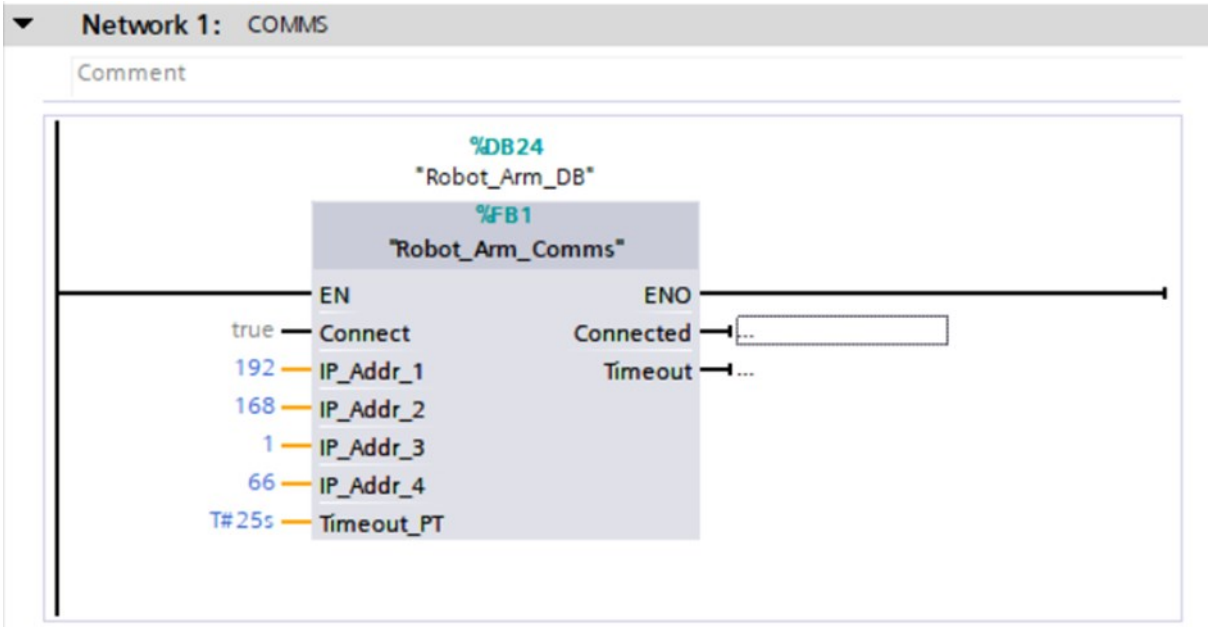
Enter 1245 as the port number.

Connecting the Robot Arm to the S7 using Wi-Fi

After entering these settings, stop the program and power cycle the arm. When the arm powers up, start the program again. Click on the WIFI Get IP button to learn the IP address that the Wi-Fi router assigned to the robot arm.



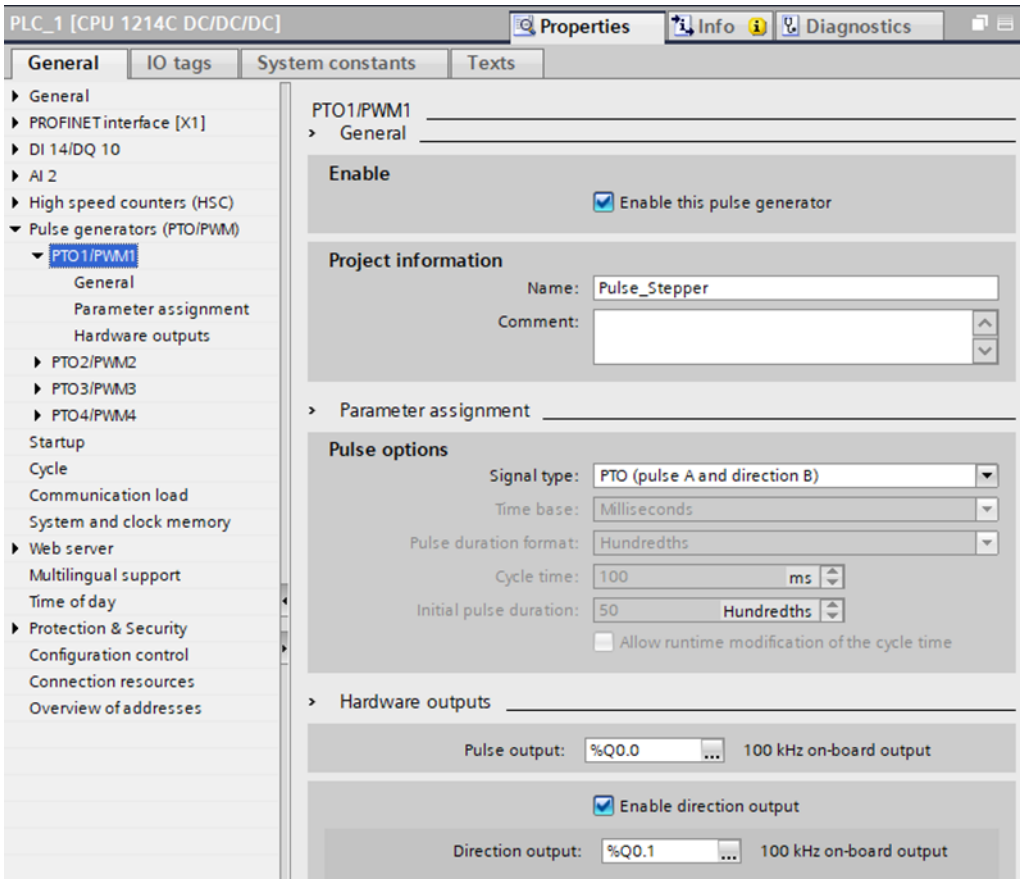
Make a note of this address. The function block Robot_Arm_Comms is described in the section on function blocks for the robot arm. This block maintains the connection with the robot and will require its IP_Addr inputs to match the address of the robot arm. The image below shows where the octets 192,168,1 and 66 are input to this block. These values should be replaced according to the address that is found in the step above.



Motion Control Technology Object

The project framework encapsulates the stepper motor in the Smart Factory gantry as a Motion Control Technology Object. The pulse output for driving the stepper is assigned to output %Q0.0 (DQa.0) and the direction control is assigned to output %Q0.1 (DQa.1). The home position limit switch is assigned to input %I0.4 (DIa.0). When starting from the project framework this object has already been configured and the function blocks in the Motion Control portion of the instruction set can be used directly.

The fast output pulse is configured in the ‘Device configuration’ tab of the PLC. The pulse generator PTO1 is assigned to output %Q0.0 with %Q0.1 as a direction output. This configuration has already been done in the example programs.



The Motion Control Technology object has the type ‘TO_PositioningAxis’. It builds on the pulse generator and encapsulates many of the operations that are needed for driving the gantry.

The parameters can be input using the configuration screen. The Function view screen guides the process of configuring the object. It is assigned to the pulse generator described above and millimeters are chosen as the unit of measurement. Mechanical parameters and the homing procedure are configured in this section.

Motion Control Technology Object

Basic parameters

Axis name: Axis_Stepper

Drive: PTO (Pulse Train Output)

Position Unit: mm

Pulse Output: %Q0.0

Direction Output: %Q0.1

Extended parameters

Pulses per revolution: 200

Load movement per revolution: 40mm

Dynamics

Maximum velocity: 2000 mm/s

Start/stop velocity: 1.0 mm/s

Acceleration & Deceleration: 2000 mm/s^2

Active Homing

In active homing, the program calls a home function and the axis will hunt for the homing microswitch automatically. If passive homing were selected then the program must start the axis moving, stop when it detects the switch and then set the home position.

Input homing switch: %I0.4

Homing direction: Negative

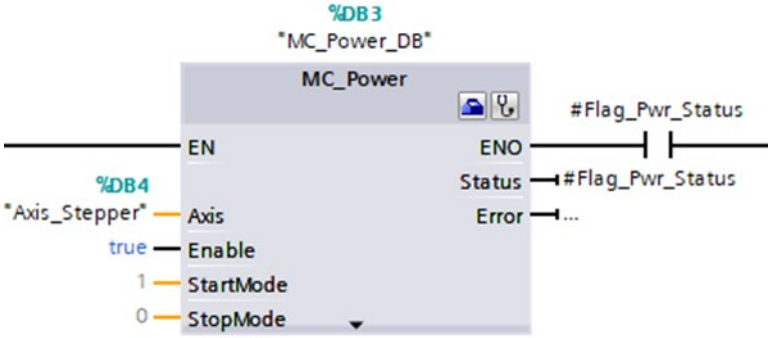
Side of homing switch: Top side

Approach velocity: 20.0 mm/s

Homing velocity: 10.0 mm/s

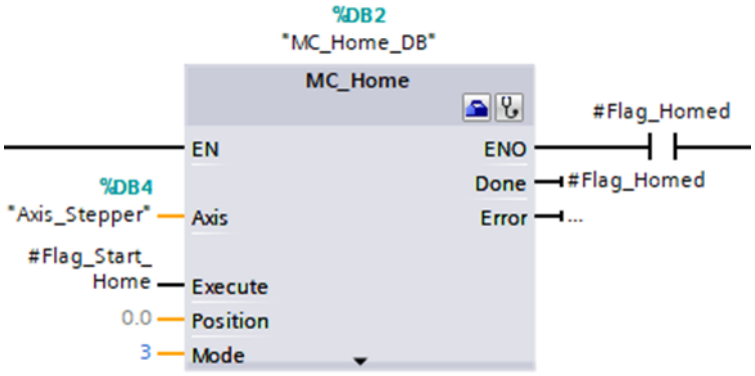
Once configured, a Commissioning panel is available that allows the programmer to test the axis to determine if the motion is correct before writing the program.

The MC_Power block must be called before any other function block. The Axis input is “Axis_Stepper”, Enable is true, StartMode = 1 and StopMode = 0. Once the object is ready the Status output becomes true and the rest of the program can continue.

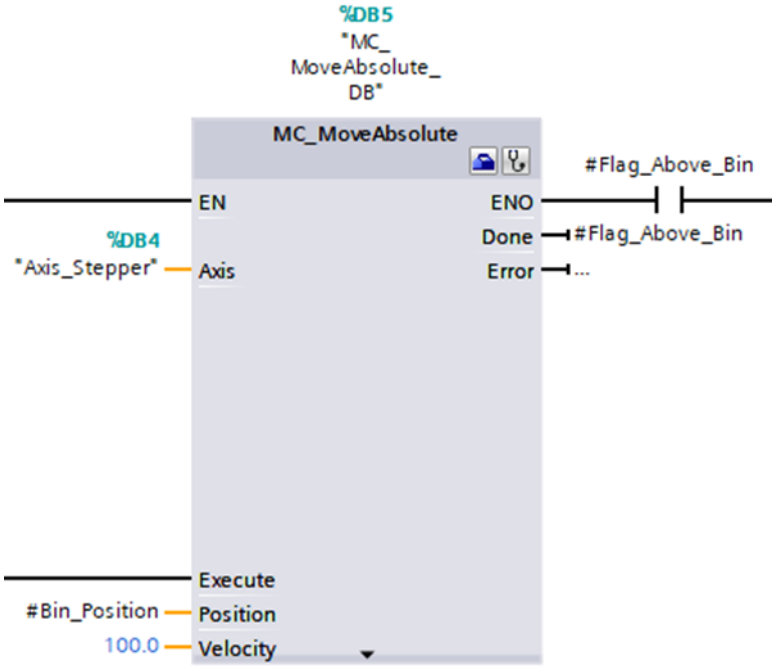


Motion Control Technology Object

At the beginning of an operation, it is a good idea to home the stepper. Calling the MC_Home block will cause the gantry to automatically drive into the limit switch and learn its home position. The homing procedure starts the Execute input changes from false to true. When the Done output becomes true, the rest of the operation can continue. The Position input should be set as 0.0 and the Mode input as 3.



The MC_MoveAbsolute block can be used to send the gantry tool to any position relative to its home position. The move will start when the Execute input goes from false to true. The Position input is the desired position measured in millimetres from the home position. The Velocity is measured in mm per second, 100 mm/s is a suitable velocity to use. The technology object will automatically generate a smooth acceleration ramp up to the target velocity and decelerate as the end of the move approaches. When the move is completed, the Done output will become true to signal that the next part of the program can continue.



Function Blocks for Robot Arm

The robot arm is shipped with an Application Programming Interface that allows it to be controlled via Wi-Fi.

The project framework contains function blocks that package the API of the AllCode Robot Arm. This makes it easy to use the Robot Arm with an S7 PLC.

Robot_Arm_Comms

This block attempts to establish a connection with a robot arm at the specified IP address. It must be run in a cycle before any of the API function blocks are used. If the connection to the arm is lost then it will attempt to re-establish the connection as soon as it becomes possible.

Connection	Direction	Data Type	Description
Connect	Input	Bool	Set to true to attempt to establish and maintain a con-
IP_Addr_1 ~ 4	Input	Byte	The IP address of the robot arm, this must be on the same subnet as the PLC.
Timeout_PT	Input	Time	This is the maximum time that the API function blocks
Connected	Output	Bool	Becomes true to indicate that the connection is estab-
Timeout	Output	Bool	Becomes true when one of the function blocks has waited

RA2_Get_API_Version

This block sends a query to the robot arm to check the API version in the arm firmware. This library is compatible with version 1. This block is useful as a check that the communication has been established and is working correctly.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the query.
Done	Output	Bool	This output is true for one cycle at the successful com-
Busy	Output	Bool	True while the query is sent or the block is waiting for a
Error	Output	Bool	True for one cycle if the query fails.
API_Version	Output	Word	If successful, this output will hold the API version number.

RA2_Home

This block sends the robot arm to its home position. This is necessary at least once because all the movements are relative to the homing microswitches. Depending upon where the arm is when this block is called, it can take up to 20 seconds to complete the operation.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Done	Output	Bool	This output is true for one cycle at the successful completion of the operation.
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Home_Axis

This block sends one single axis of the arm to its home position.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Axis	Input	Byte	Identifies the axis to be homed: 0 = Base, 1 = Shoulder, 2 =
Done	Output	Bool	This output is true for one cycle at the successful completion
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Get_Position

This block gets the current position in steps of each axis of the arm. It should give zero for each axis after a home operation and a positive number of steps if the arm is away from the home position.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the query.
Done	Output	Bool	This output is true for one cycle at the successful com-
Busy	Output	Bool	True while the query is sent or the block is waiting for a
Error	Output	Bool	True for one cycle if the query fails.
Axis_A, Axis_B,	Output	int	If successful, these outputs will hold the step counts for

RA2_Goto_Position

This block will cause the arm to move to a previously stored position. These positions would typically have been stored while controlling the arm with a pendant. The arm will take the shortest path from its current position. In the factory there are likely to be obstacles around the arm. The programmer should allow for this by programming a series of positions that navigate around the obstacles.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Pos_Index	Input	Byte	Identifies the index of the stored location.
Done	Output	Bool	This output is true for one cycle at the successful completion of
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Set_Motor

This block sets the position in steps of a single motor.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Motor	Input	Byte	Identifies the motor to be set: 0 = Base, 1 = Shoulder, 2 = El-
Coord	Input	int	This is the required position of the motor in steps.
Done	Output	Bool	This output is true for one cycle at the successful completion of
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Set_All_Motors

This block sets the positions of all three motors in steps.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Coord_0 ~ 2	Input	int	These are the required positions of the motors in steps.
Done	Output	Bool	This output is true for one cycle at the successful comple-
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Get_ToolXYZ

This block gets the current position in x,y,z coordinates of the tool at the end of the arm. The units are in millimetres with the pivot point of the base as the zero position.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the query.
Done	Output	Bool	This output is true for one cycle at the successful com-
Busy	Output	Bool	True while the query is sent or the block is waiting for a re-
Error	Output	Bool	True for one cycle if the query fails.
Tool_X, Tool_Y, Tool_Z	Output	int	If successful, these outputs will hold x,y,z coordinates of the tool. If it fails then the outputs will all be -1.

RA2_MoveToXYZ

This block moves the arm to position the tool at the specified x,y,z coordinates. The units are in millimetres with the pivot point of the base as the zero position.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Tool_X, Tool_Y,	Input	int	These are the required hold x,y,z coordinates of the tool.
Done	Output	Bool	This output is true for one cycle at the successful com-
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Set_Gripper

This block sets the position of the gripper tool at the end of the arm. The position can be set between 0 = fully open and 255 = fully closed. The jaw are sprung so that the servo motor can be driven to the fully closed position even if an object is between the jaws.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Gripper	Input	Byte	The required position of the gripper tool. 0 = fully open and
Done	Output	Bool	This output is true for one cycle at the successful comple-
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Get_Colour

This block reads an RGB colour sensor connected to the arm. It is necessary to ensure that the colour sensor is attached before calling this block since attempting to read a sensor that is not there will cause the arm to stop responding until the next power cycle. The output is a structure of type Colour that contains one byte for each of the three colours detected by the sensor.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Done	Output	Bool	This output is true for one cycle at the successful completion of
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.
Colour	Output	Struct	A structure of three bytes: Red, Green and Blue that contain the

RA2_Disable_Motors

This block cuts power to all three motors allowing the arm to be moved by hand. If this is used then a home instruction must be used to re-enable the motors and allow the arm to learn its position.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Done	Output	Bool	This output is true for one cycle at the successful comple-
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Set_LED

This block sets state of one of the five LEDs on the arm to on or off. This overrides the standard functions of the LEDs.

Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
LED	Input	Byte	Selects which LED to set. 0 ~4
LED_On	Input	Bool	True to switch the LED on, false to switch it off.
Done	Output	Bool	This output is true for one cycle at the successful completion of
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

RA2_Auto_LEDs

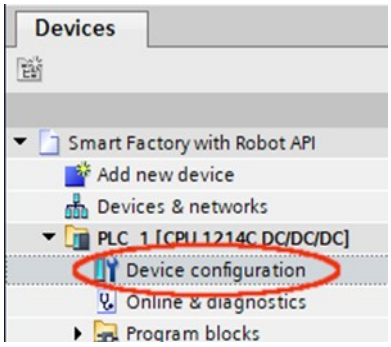
This block cancels the effect of RA2_Set_LED and returns LEDs to their standard functions.

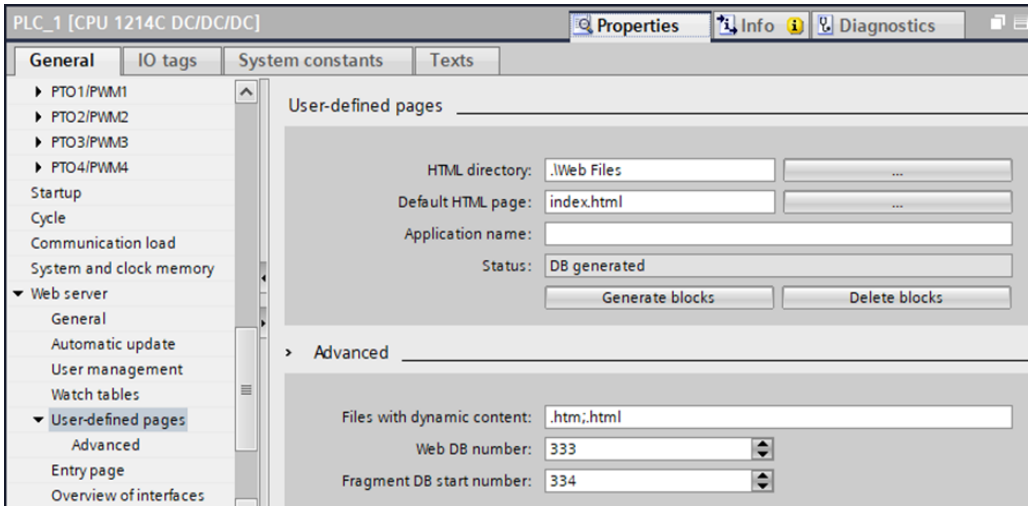
Connection	Direction	Data Type	Description
Req	Input	Bool	A rising edge on this input starts the operation.
Done	Output	Bool	This output is true for one cycle at the successful completion of the query.
Busy	Output	Bool	True while the operation is in progress.
Error	Output	Bool	True for one cycle if the query fails.

Control via the Webserver

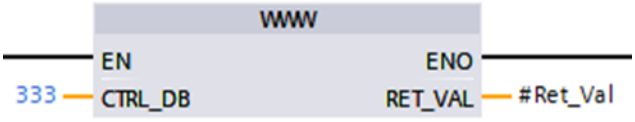
The S7 PLC has a built in web server that can be used to control and application and to report its status. The webserver can in effect be used as an HMI. The project framework has been configured to include user defined pages within its webserver. The pages to be included are contained in a folder named ‘Web Files’ within the project folder and the first page to be displayed will be ‘index.html’.

If these pages are edited then the data block in the PLC must be updated. Locate the PLC in the devices tree and select Device configuration. In the Properties panel for the PLC, locate the User-defined pages part of the Web server section and click the Generate blocks button. This will overwrite the existing data block with the newly edited pages.





Once the blocks are generated, the WWW function block handles updating and serving the web pages. This block is already included in the project framework. There is nothing for the programmer to do with this block apart from including it in the program so that it runs in the main cycle. The CTRL_DB input is set to 333, the default block number that is generated in the step above.



Once the webserver is enabled and configured, make a note of the IP address of the PLC (see section 1) and type it into the address bar of a web-browser. It may be necessary to prefix the address with https:// for example:

https://192.168.1.150

Control via the Webserver

Live Variables in Web Pages

The easiest way to include a live variable in a web page is to write the variable name in the html file as it would appear in a program and put ':=' in front of it and ':' after it. For example; if the program includes a data block named "Web_Data" with a variable inside it named "Count_Fe" then this variable would appear in the program as:

"Web_Data".Count_Fe

If the web page includes a piece of html code of the form:

:="Web_Data".Count_Fe:

Then, when the page is served, the variable name will be replaced with the current value of the variable.

The variable is only updated when the page is served, if the program writes a new value to the variable it will not be updated on the page. The simplest way to fix this is to include the line

<meta http-equiv="refresh" content="1">

at the top of the page to force the page to periodically reload. This is easy to do but the page will be seen to flicker as it re-loads. A more elegant approach is to place the variables in a separate page and use jQuery and AJAX to update the variables on the main page. An example of this approach is included in the project framework. For more information on jQuery and AJAX see online resources such as https://www.w3schools.com/js/js_ajax_intro.asp

If the web page is expected to write to the variable as well as read it then an html comment declaring AWP_In_Variable must be included at the top of each page that references that variable. If the program includes a variable named "Web_Data".Trigger_1 then the comment

<!-- AWP_In_Variable Name='"Web_Data".Trigger_1'-->

must be included. The variable can then be updated by posting a new value. The web pages included in the project framework contain an example of this.

Example Programs

The examples are product code AU6011 which can be downloaded from the Matrix web site.

The following example programs are provided to demonstrate the use of Smart Factory with the Siemens S7-1214 PLC: -

“Smart Factory Siemens S7-1200.zap15”

This program runs the gantry and the conveyor only. Aluminium and steel counters are sorted by the reject booms and plastic counters run off the end of the conveyor.

“Smart Factory with Robot Demo.zap15”

This program builds on the previous one. When a plastic counter is detected, the conveyor stops and a robot arm is triggered to pick up the counter. The robot holds the counter against a colour sensor and places it in a bin depending upon its colour. This program includes a web interface that presents the user with a count of the different types of counter that have been sorted and enables picking from any of the three hoppers.

“Smart Factory Framework.zap15”

The project includes all the necessary configuration for the various technologies that have been used in the previous programs. It also includes the function blocks for control of the robot arm. It does not include the higher level programming of the components. Use this project as a starting point for developing your own programs.

Opening the Example Program

1. Download the project archive.
2. Open TIA Portal version 15 or later.
3. In the Project View, select “Retrieve…” from the Project menu.
4. Select the project archive file and choose a directory to store the project.

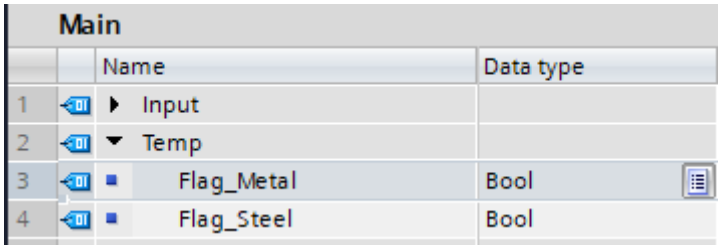
Example Programs

Sorting Counters – Sensor Gates and Reject Booms
The smart factory is built as shown in CP7329, Worksheet 4. The S7-1214 PLC is connected as follows: -

Connection	Device	PLC Tag Name
Inputs Block: 0	First light gate	%I0.0 Sensor_First_Light
Inputs Block: 1	Inductive sensor	%I0.1 Sensor_Inductive
Inputs Block: 2	Capacitive sensor	%I0.2 Sensor Capacitive
Inputs Block: 3	Final light gate	%I0.3 Sensor_Final_Light
Motor Outputs: 0	Steel reject boom	%Q1.0 Reject_Steel
Motor Outputs: 1	Aluminium	%Q1.1 Reject_Al

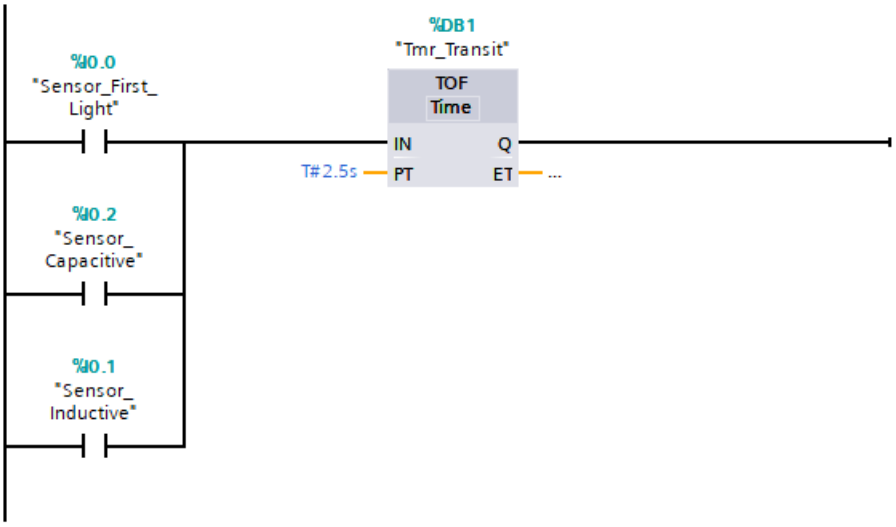
The PLC Program is named Sorting and runs as the Program cycle. It will run continuously, starting again as soon as it finishes.

Two local variables are declared, Flag_Metal and Flag_Steel, both of type Bool.



Network 1: TRANSIT TIMER

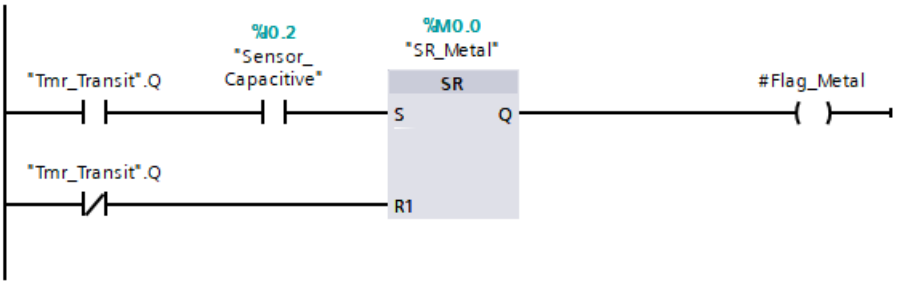
The first light gate or either of the following sensors will trigger the transit timer. Once triggered, the timer will stay on for the length of time taken to transit from one end of the conveyor to the other. The purpose of this timer is to keep the reject booms out until the counter has completed its transit.



Example Programs

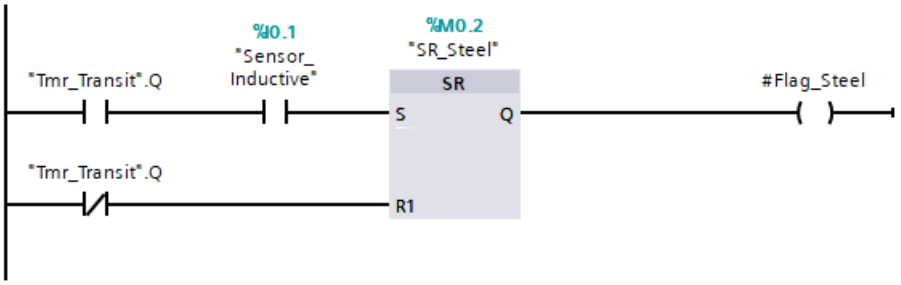
Network 2: LATCH METAL FLAG

This flag is latched when the capacitive sensor tells us that we have a metal counter. It will stay latched until the transit timer times out (the counter has completed its transit).



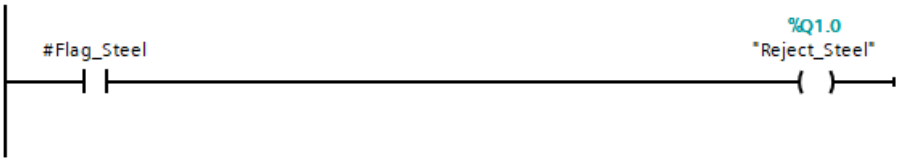
Network 3: LATCH STEEL FLAG

This flag is latched when the inductive sensor tells us that we have a steel counter. It will stay latched until the transit timer times out.



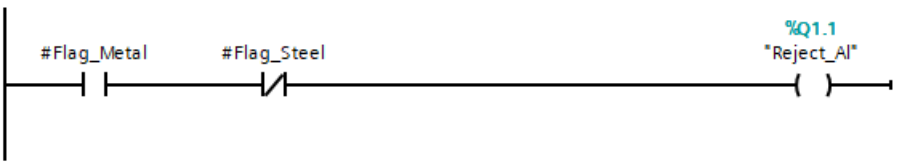
Network 4: SORT STEEL

As long as we have a steel counter, the reject boom is operated. This will be from the inductive sensor pulse until the transit timer times out.



Network 5: SORT ALUMINIUM

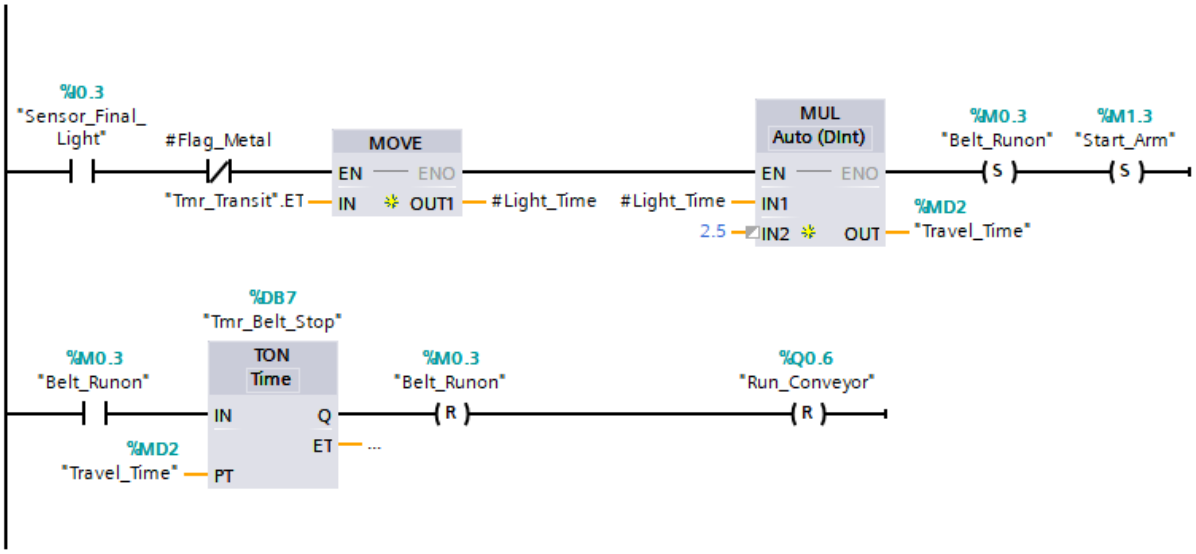
As long as we have a metal counter that is not steel, the reject boom is operated. This will be from the inductive sensor pulse until the transit timer times out.



Example Programs

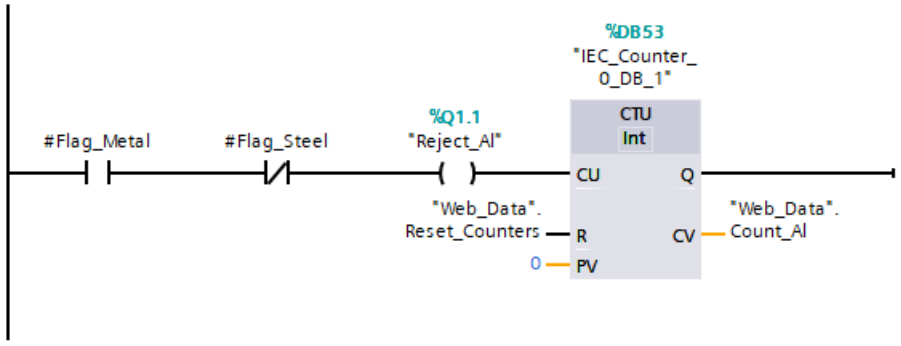
Network 6: TRIGGER ROBOT ARM

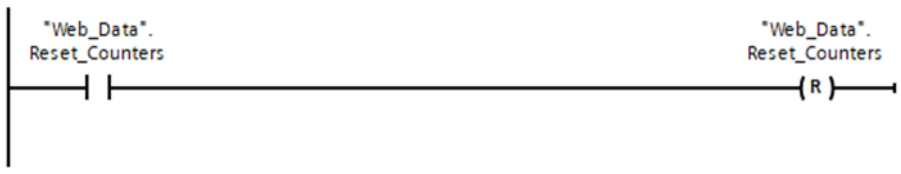
This network is only included in the “Smart Factory with Robot Demo” program. If the counter is not metal then the final light gate is used to get the conveyor speed. The conveyor is stopped after 2.5x the time between the two light gates and then the robot arm is triggered.



Network 7: RESET COUNTERS

This network is only included in the “Smart Factory with Robot Demo” program. Networks 4, 5 and 6 include CTU blocks that count up every time their CU input changes from false to true. The current values of these counters are written the count variables that are displayed on the web interface. The web interface incudes a flag that is used to reset the counters. This flag is read by each of the counters, triggering a reset and then it is itself reset in network 7 so that the counters are only reset once.





Picking Counters – Gantry and Plunger

With the sorting routine working, the program is extended to deliver counters to the conveyor for sorting. The following connections are added: -

Connection	Device	PLC Tag Name
Inputs Block: 4	Gantry home microswitch	%I0.4 Axis_Home_Position
Inputs Block: 5	Trigger pick from bin 1	%I0.5 Pick_Bin_1
Inputs Block: 6	Trigger pick from bin 2	%I0.6 Pick_Bin_2
Inputs Block: 7	Trigger pick from bin 3	%I0.7 Pick_Bin_7
Outputs: 0	Stepper motor pulse	%Q0.0 Stepper_Pulse
Outputs: 1	Stepper motor direction	%Q0.1 Stepper_Direction
Outputs: 3	Plunger down valve	%Q0.3 Pneu_Plunger_Down
Outputs: 4	Plunger up valve	%Q0.4 Pneu_Plunger_Up
Outputs: 5	Power vacuum generator	%Q0.5 Pneu_Vacuum
Outputs: 6	Power conveyor	%Q0.6 Run_Conveyor

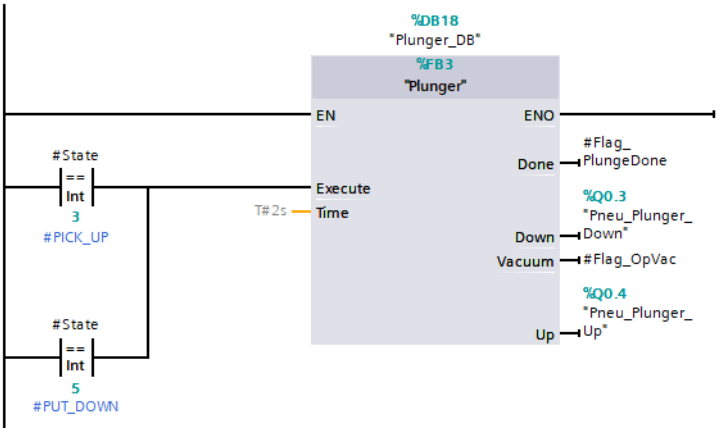
GANTRY PROGRAM

The Gantry program is added to the main cycle and calls the “Run_Gantry” function block. Along with the sorting program, this will be run to completion and then restarted.

The function is implemented as a state machine. The variable #State runs through a sequence with different parts of the program being run depending upon the sequence. The state is incremented when each stage is completed.

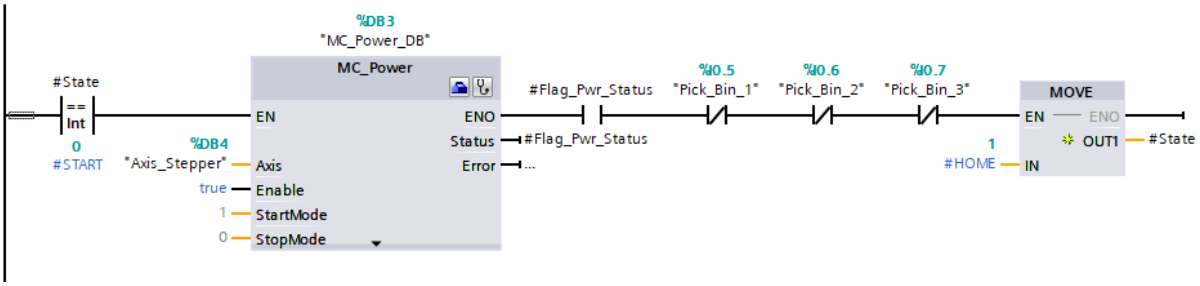
Network 1: Plunger

Two states can trigger the plunger function block; picking up from the bins or putting down on the conveyor. The plunger begins to operate as soon as either state is reached. The up and down valves are driven directly but the vacuum output is a flag. This flag is used to switch the vacuum on or off depending upon the state.



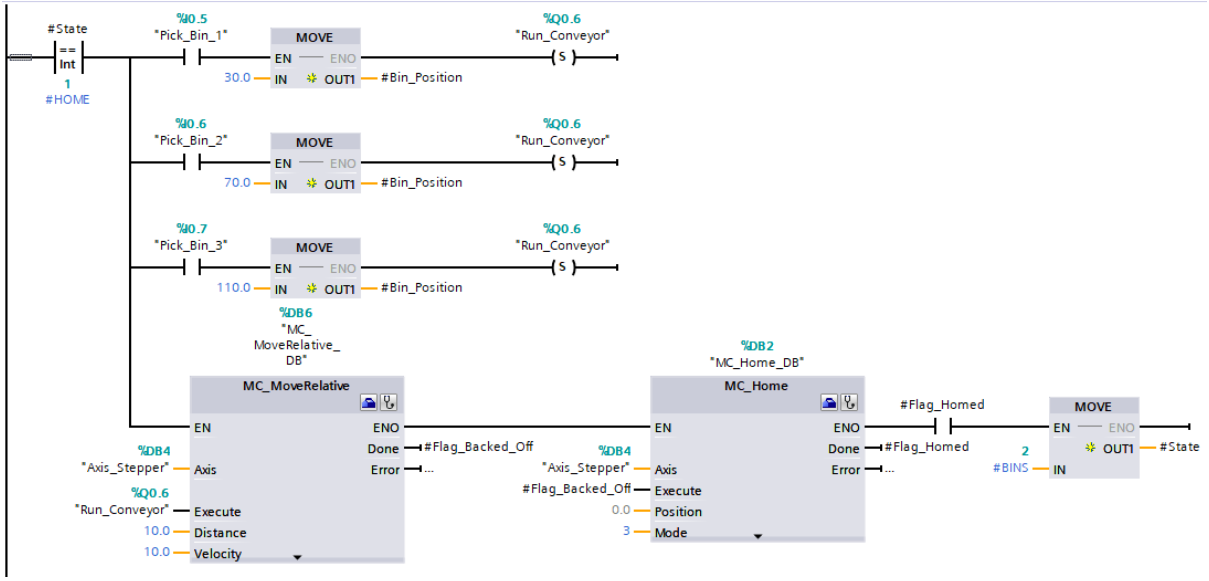
Network 2: Start

When the PLC starts up the MC_Power block is called to initialize the motion control object. Once the axis is ready and no pick inputs are energized then the state progresses to the #HOME state.



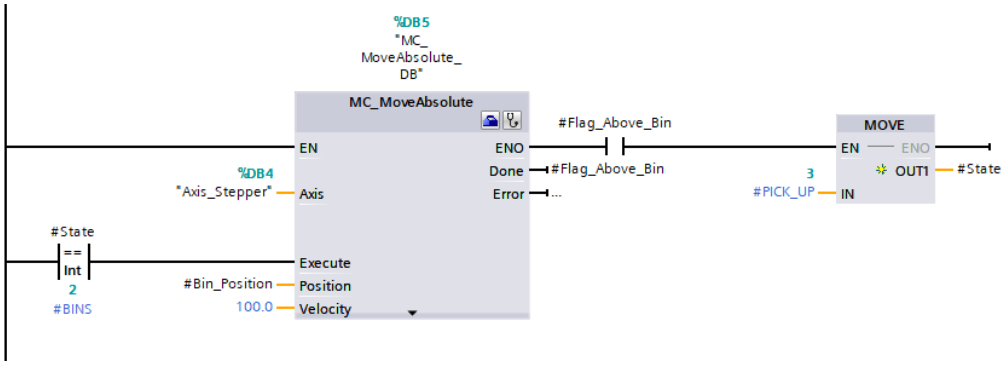
Network 3: Home

In this state the function waits for one of the pick inputs to be energized. A constant is copied to the #Bin_Position variable depending upon which input and the conveyor is started. Starting the conveyor triggers the stepper to back of by 10mm and then run its homing procedure. Once the homing procedure is complete, the state progresses to the #BINS state.



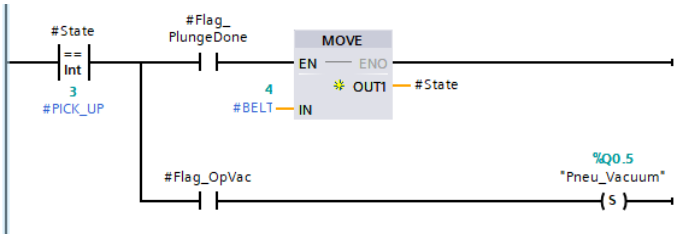
Network 4: Move to counter bins

In this state, the stepper motor is moved to position the plunger above one of the counter bins. The position at the end of this move depends on the bin that was chosen in the previous state.



Network 5: Pick up counter

In this state the program waits for two signals from the plunger function block that was called in network 1. When the signal is given to operate the vacuum, the vacuum generator is switched on. When the cycle of the plunger is complete, the state progresses.

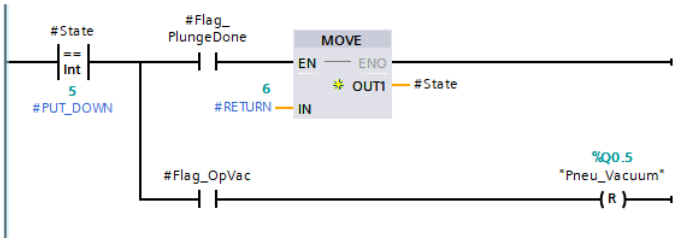


Network 6: Move to the belt

In this state the stepper moves to position the plunger above the conveyor belt.

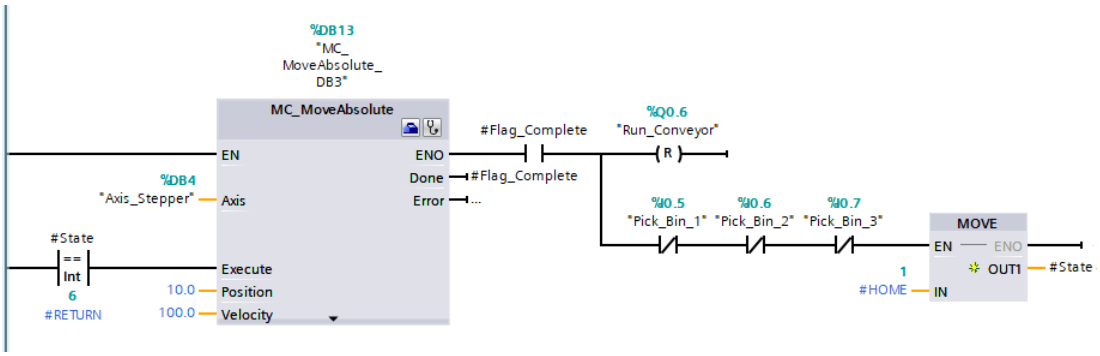
Network 7: Put down the counter

This state is very similar to network 5 where the counter was picked up. In this state however, the signal to operate the vacuum results in the vacuum generator being switched off and the counter dropped onto the conveyor.



Network 8: Returning to the park position

In this state, the stepper returns the gantry to its home position and stops the conveyor. The state becomes ready for the next cycle when all the pick inputs are off. This means that a new input signal is needed to trigger the next operation.

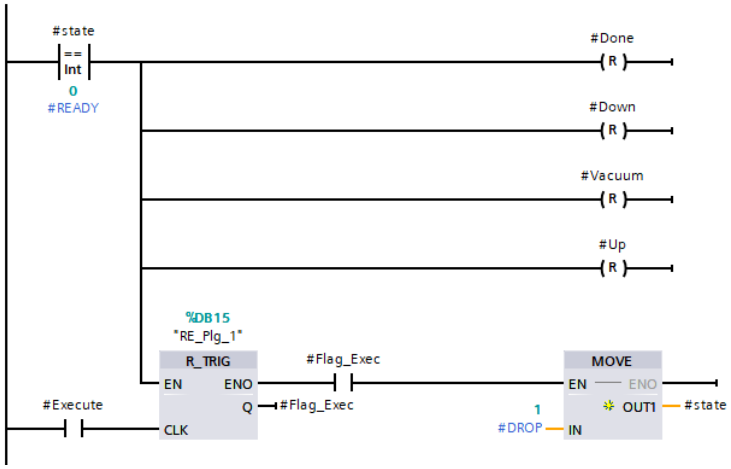


PLUNGER FUNCTION BLOCK

The plunger function block is implemented as a state machine and is called by the gantry program when it is in a position to operate the plunger.

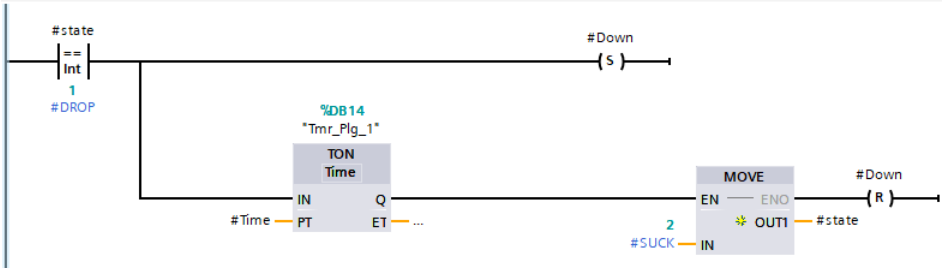
Network 1: Ready

In this state the function switches off all the outputs and waits for the rising edge of the execute input.



Network 2: Driving down

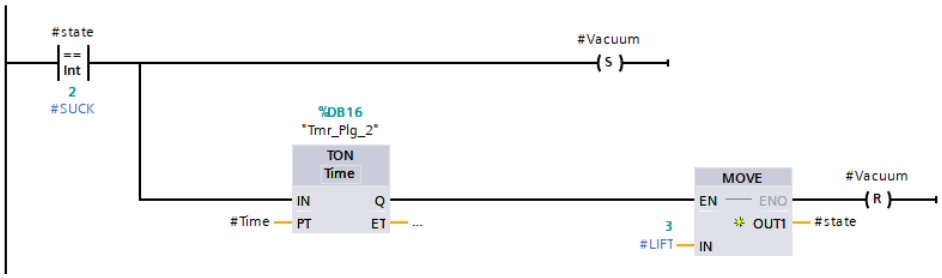
The valve is set to drive the plunger down and a timer is started. When the timer completes, the valve is switched off and the state progresses.



Example Programs

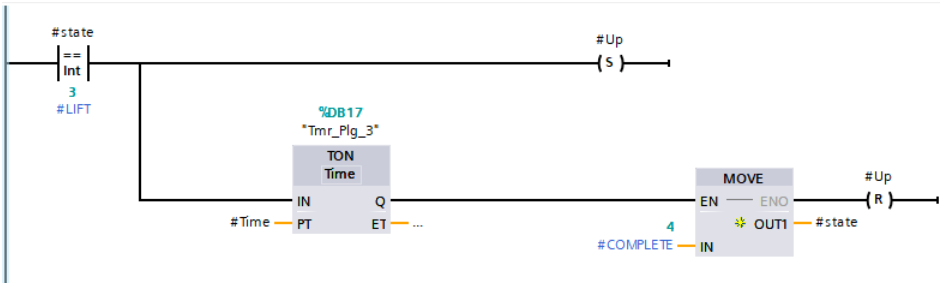
Network 3: Operating the Vacuum

In this state, the vacuum flag is signaled until a timer completes and the state progresses. The gantry program that calls this function will use the vacuum signal to turn the vacuum on or off depending upon where in the cycle it is.



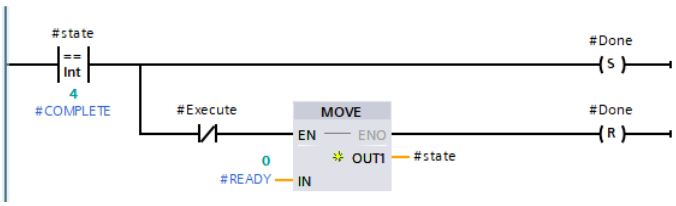
Network 4: Driving up

In this state the plunger is driven up until a timer completes.



Network 5: Complete

In this state a flag is set to signal to the calling program that the sequence is complete. When the execute signal is removed, the state resets to the beginning.

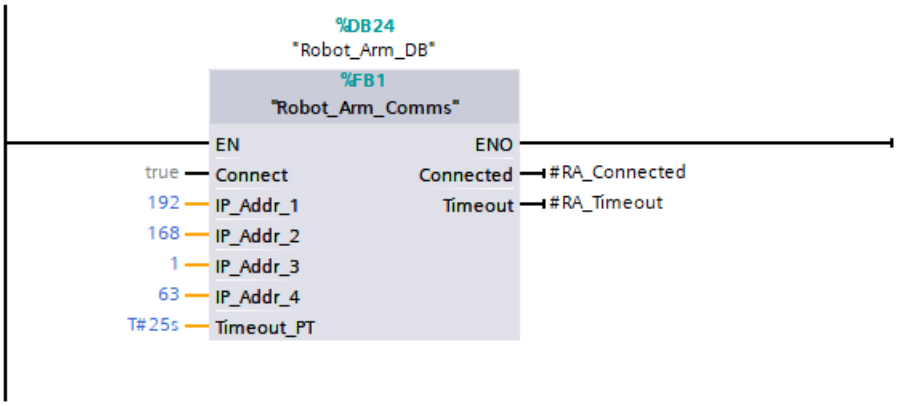


Control of the Robot Arm

Two networks are included in the Robot program. The first handles the communication to the robot arm and the second contains the entire sequence of picking up, checking and sorting a counter.

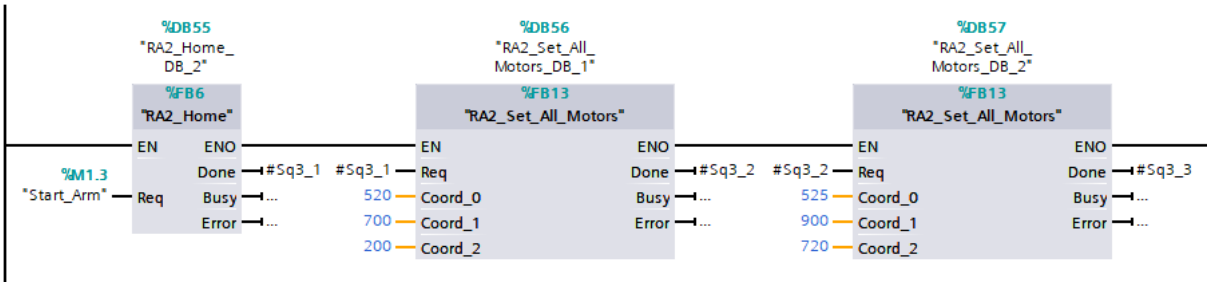
Network 1 COMMS

This network contains the block that handles communication. It must be run at the beginning of each program cycle. The inputs and outputs are described in section 3.

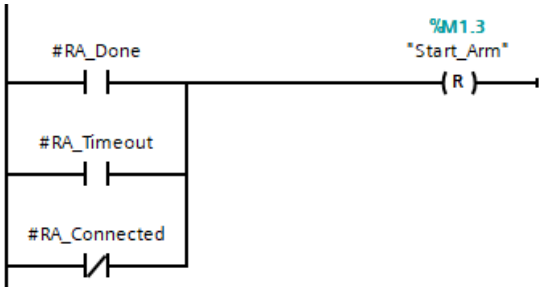


Network 2 Complete pick, check colour and sort routine

Each of the blocks is triggered by a transition from false to true on its Req input and sets its Done output to true when the operation is completed. This means that blocks can be chained together with the Done output of one operation triggering the Req input of the next.

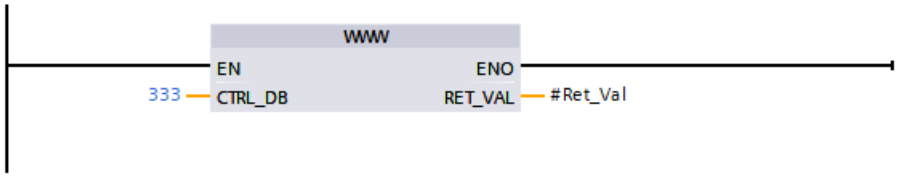


The whole sequence when the Sorting program detects a plastic counter and sets the “Start_Arm” flag. This flag is reset ant the end of the sequence. The Run_Gantry function block uses this flag to hold off dropping a new counter until the robot arm has finished its operation. If the robot arm is not connected or times out due to an error then this flag must also be reset so that the gantry is not left hanging.



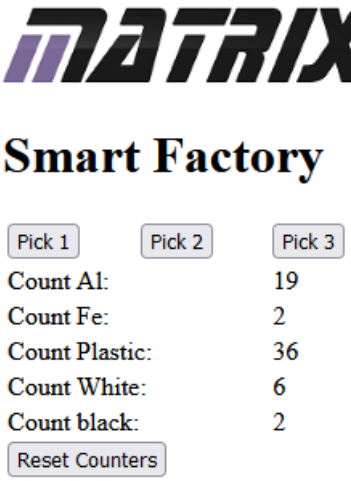
Web Server Interface

The “Smart Factory with Robot Demo” program includes a web server interface. The Web_Server program simply calls the WWW function block.

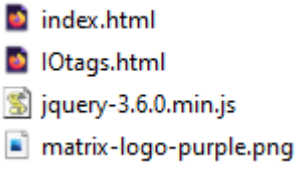


This block handles serving the user web pages and the live variables. The variables that are referenced by the web pages are all included in the “Web_Data” data block. This is done for convenience, the web pages could access any global variables in the PLC.

The web page includes buttons to trigger the gantry to pick from one of the three hoppers, counts of each type of counter sorted and a button to reset the counters.



The Web Files directory includes four files that are required for the web interface. These are the jQuery library, a logo and two html pages.



Example Programs

IOtags.html

This file contains declarations of all the variables that can be written and a JSON structure containing all the variables in the web interface. The names of the variables are replaced by the current values when the page is served.

```
<!-- AWP_In_Variable Name='"Web_Data".Trigger_1'-->
<!-- AWP_In_Variable Name='"Web_Data".Trigger_2'-->
<!-- AWP_In_Variable Name='"Web_Data".Trigger_3'-->
<!-- AWP_In_Variable Name='"Web_Data".Reset_Counters'-->
<!-- AWP_In_Variable Name='"Web_Data".Robot_Present'-->
{
"tg1":":="Web_Data".Trigger_1:",
"tg2":":="Web_Data".Trigger_2:",
"tg3":":="Web_Data".Trigger_3:",
"rct":":="Web_Data".Reset_Counters:",
"rap":":="Web_Data".Robot_Present:",
"cal":":="Web_Data".Count_Al:",
"cfe":":="Web_Data".Count_Fe:",
"cps":":="Web_Data".Count_Pls:",
"cbk":":="Web_Data".Count_Bk:",
"cwt":":="Web_Data".Count_Wt:",
"sts":":="Web_Data".Status_Str:"}
```

Index.html

This file contains the html code to display the web page and the JavaScript code to continually load IOtags.html and use its data to update the live variables displayed in the web page. It contains JavaScript functions for each of the buttons that posts new values for selected variables back to the PLC.

```
<!-- AWP_In_Variable Name='"Web_Data".Trigger_1'-->
<!-- AWP_In_Variable Name='"Web_Data".Trigger_2'-->
<!-- AWP_In_Variable Name='"Web_Data".Trigger_3'-->
<!-- AWP_In_Variable Name='"Web_Data".Reset_Counters'-->
<!-- AWP_In_Variable Name='"Web_Data".Robot_Present'-->
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>Matrix Smart Factory</title>
    <script src="jquery-3.6.0.min.js"></script>
    </head>
    <body>
      <img src="matrix-logo-purple.png">
      <h1>Smart Factory</h1>
      <table style="width:250px">
      <tr><td><button id='pk1'>Pick 1</button></td>
          <td><button id='pk2'>Pick 2</button></td>
          <td><button id='pk3'>Pick 3</button></td>
```

Example Programs

```
    </tr>
    <tr><td colspan="2">Count Al:</td><td id='Ct_Al'></td></tr>
    <tr><td colspan="2">Count Fe:</td><td id='Ct_Fe'></td></tr>
    <tr><td colspan="2">Count Plastic:</td><td id='Ct_Pls'></td></tr>
    <tr><td colspan="2">Count Black:</td><td id='Ct_Bk'></td></tr>
    <tr><td colspan="2">Count white:</td><td id='Ct_Wt'></td></tr>
    <tr><td colspan="3"><button id='rst_cnt'>Reset Counters</button></td></tr>
    </table>

</body>

<script type="text/javascript">
    $(document).ready(function(){
        $.ajaxSetup({ cache: false });
       setInterval(function() {
        $.getJSON("IOtags.html", function(result){
            $('#Ct_Al').text(result.cal);
            $('#Ct_Fe').text(result.cfe);
            $('#Ct_Pls').text(result.cps);
            $('#Ct_Bk').text(result.cbk);
            $('#Ct_Wt').text(result.cwt);
        });
       },1000);
       // Operate the buttons
       $('#rst_cnt').click(function(){
       url="IOtags.html";
       name='"Web_Data".Reset_Counters';
       val=1;
       sdata=escape(name)+'='+val;
       $.post(url,sdata,function(result){});
       });
       $('#pk1').click(function(){
       url="IOtags.html";
       name='"Web_Data".Trigger_1';
       val=1;
       sdata=escape(name)+'='+val;
       $.post(url,sdata,function(result){});
       });
       $('#pk2').click(function(){
       url="IOtags.html";
       name='"Web_Data".Trigger_2';
       val=1;
       sdata=escape(name)+'='+val;
       $.post(url,sdata,function(result){});
       });
       $('#pk3').click(function(){
       url="IOtags.html";
       name='"Web_Data".Trigger_3';
       val=1;
       sdata=escape(name)+'='+val;
       $.post(url,sdata,function(result){});
       });
    });
</script>

</html>
```