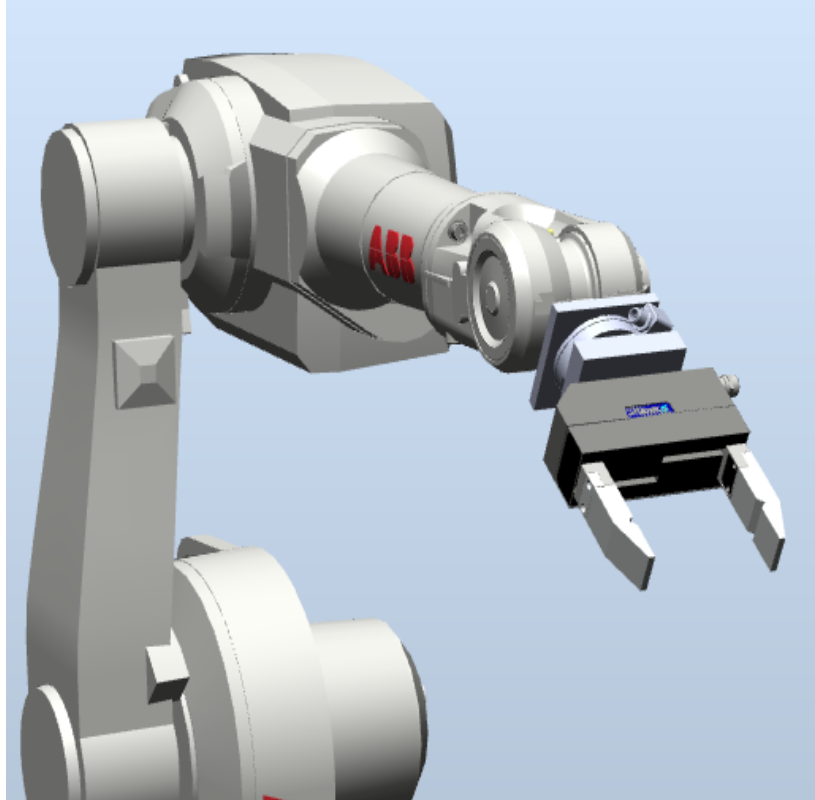


Guide for ABB RobotStudio

This guide explains step-by-step the process of creating a tool in RobotStudio



by

Michael Natapon Hansson

Department of Mechanical and Manufacturing Engineering, Aalborg University

January 2015



AALBORG UNIVERSITY
DENMARK

Content

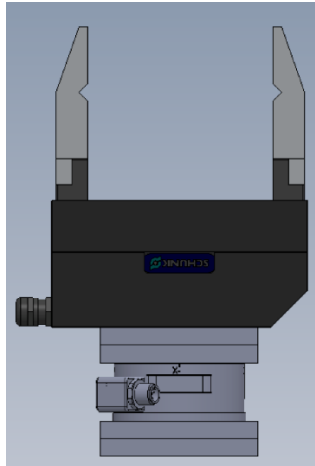
Creating a Tool	3
Creating a CAD Model	3
Importing the models in RobotStudio	4
Altering the reference frame of your geometries	5
Creating a tool mechanism of your tool geometries.....	16
Creating a Smart Component.....	25
Creating Virtual Controller I/O's to control gripper	49

Creating a Tool

Before any tools can be created, you will need a geometry, a CAD file (*.SAT) or a library (*.rslib), to act as the virtual representation of the physical tool. Since there exist many different types of tools that can be used in robot assignments, this guide will show the basic steps of creating a tool in RobotStudio. The outset of this guide is to create a parallel gripper from a CAD file, but the steps explained in this guide can also be used to create other types of grippers (e.g. vacuum grippers).

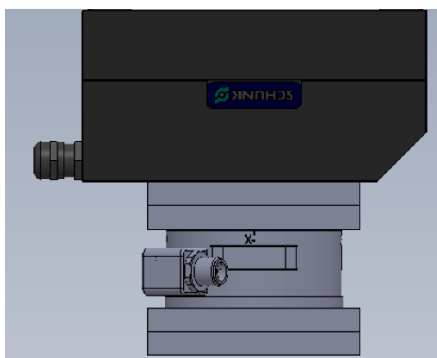
Creating a CAD Model

Create your model of the gripper in a CAD modelling software (e.g. SolidWorks).

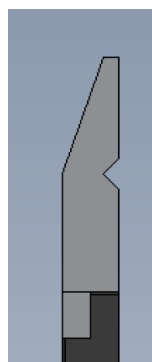


Example of gripper

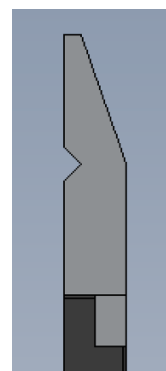
Since we are going to make a “mechanisme” in RobotStudio, we need to save all the moveable parts (the jaws) in separate files. REMEMBER to save the files in the *.SAT format, otherwise there may be problems importing the models in RobotStudio.



body.sat



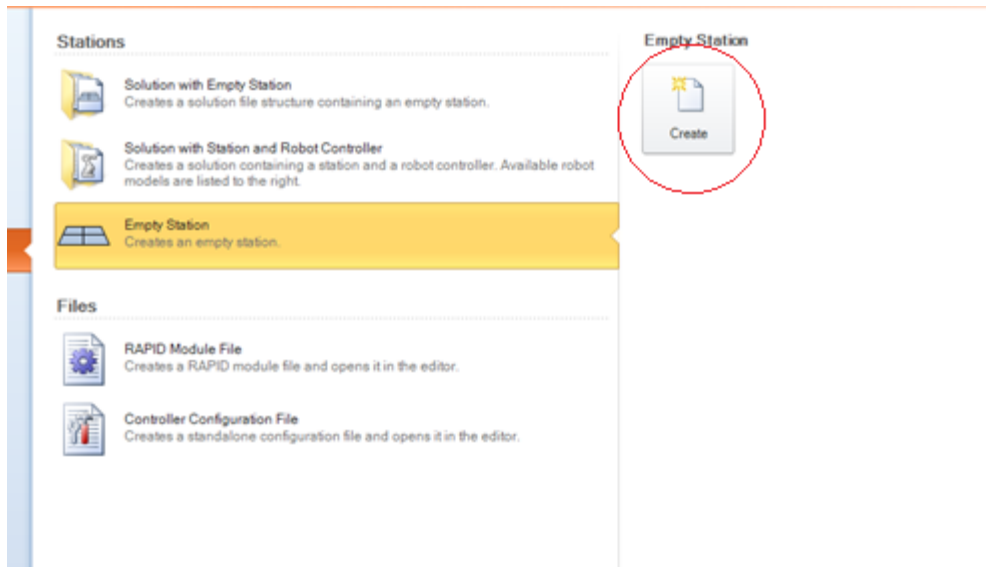
fingerA.sat



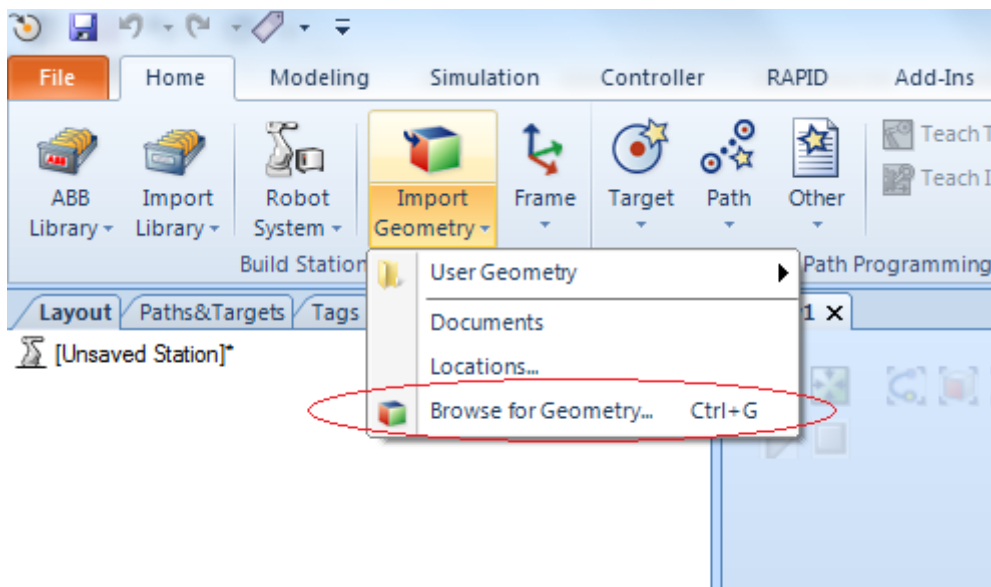
fingerb.sat

Importing Models in RobotStudio

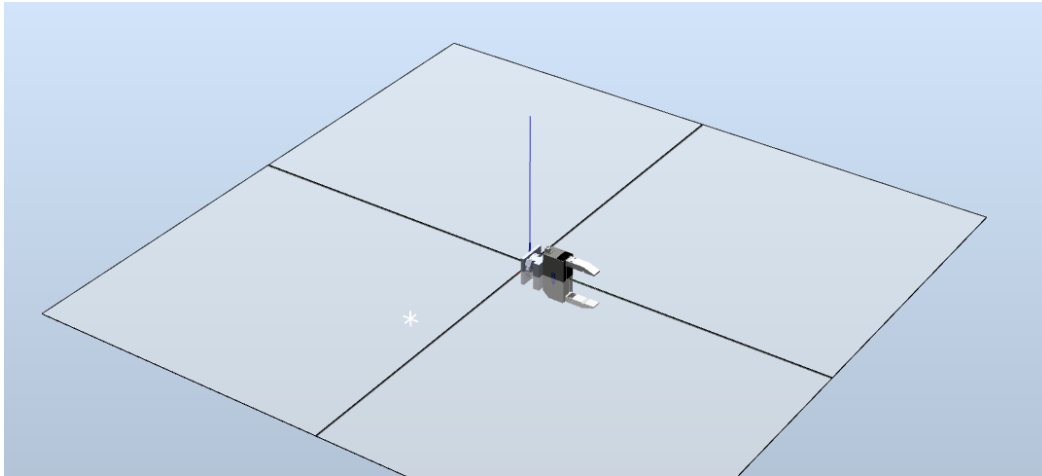
When you have made the necessary files to represent your gripper, we need to import them to RobotStudio. A good advice is to create a new station, dedicated entirely to make your tool.



Next, click on the dropdown of the “Import Geometry” button. Find all the .SAT files that you have created to represent your tool and import them to RobotStudio.

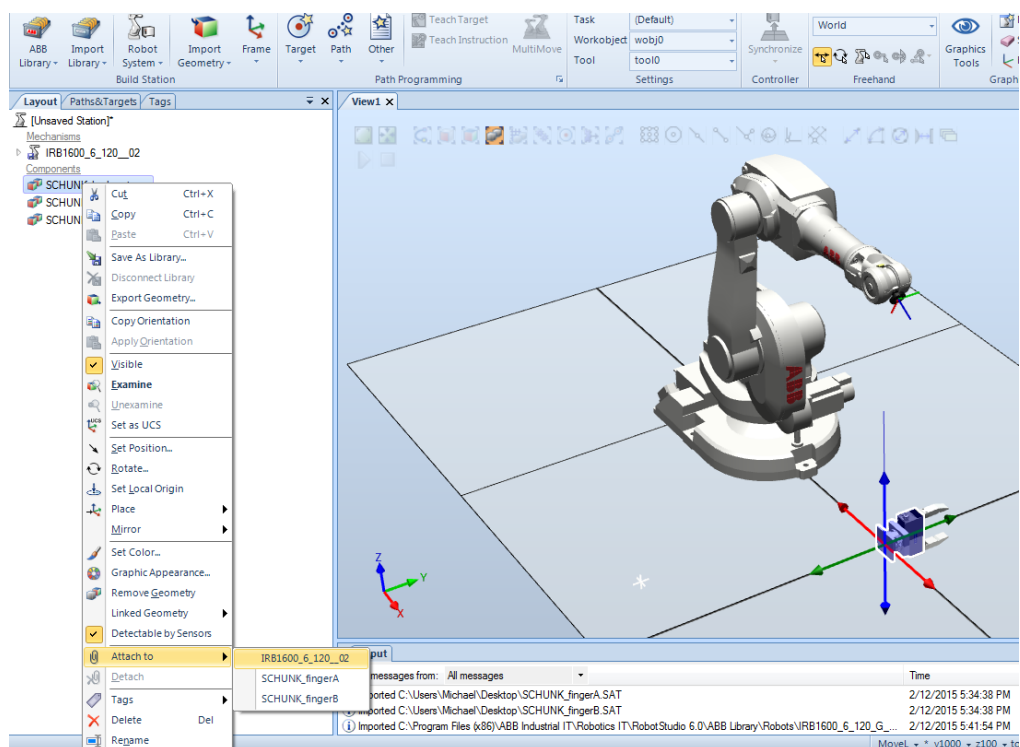


Your scene should look something like this.

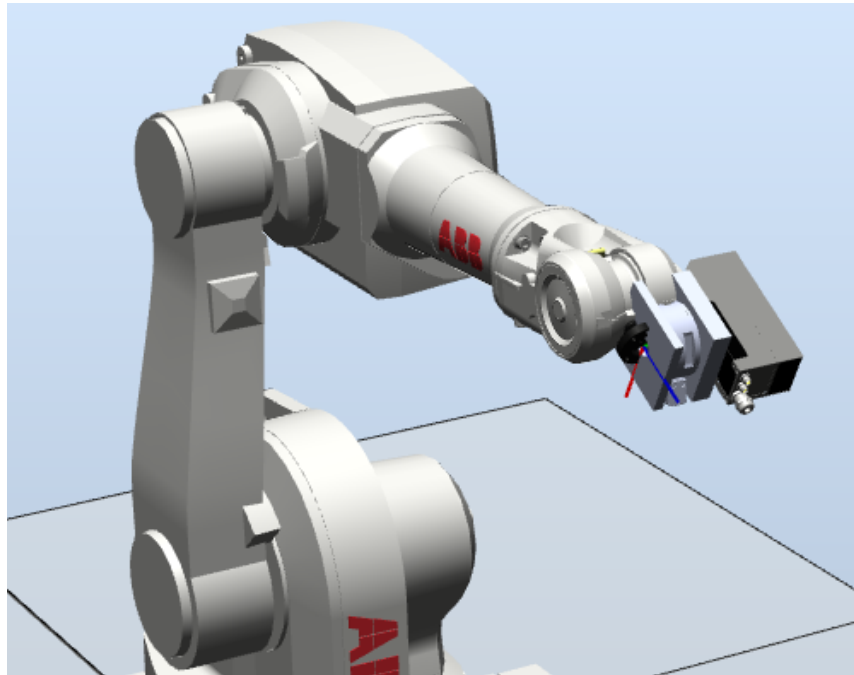


Altering the reference frame of your geometries

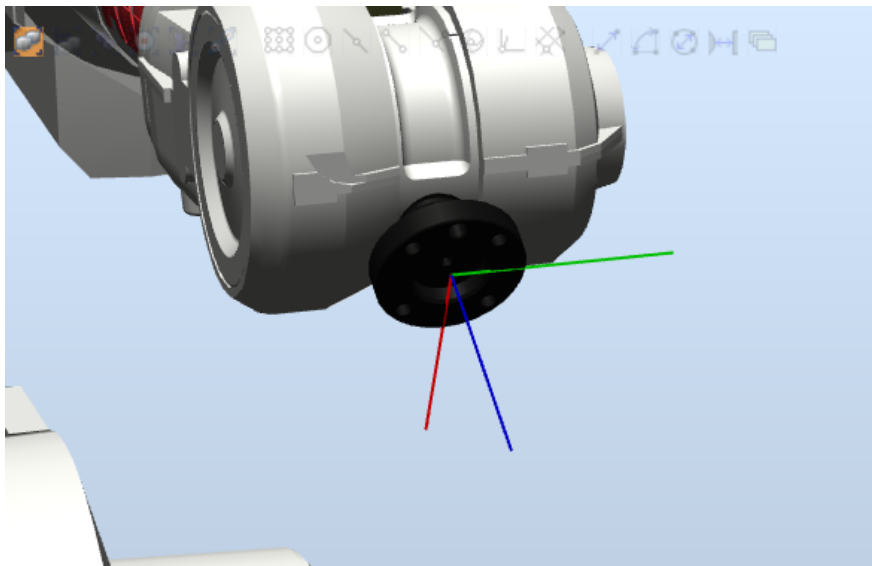
Depending on how you have modelled your tool in the CAD modelling software (the reference frame you are using), your geometries may not have the desired position and orientation, when you attach it to the robot. A fast way to check is to import a robot and attach the “body” part of your gripper to the robot.



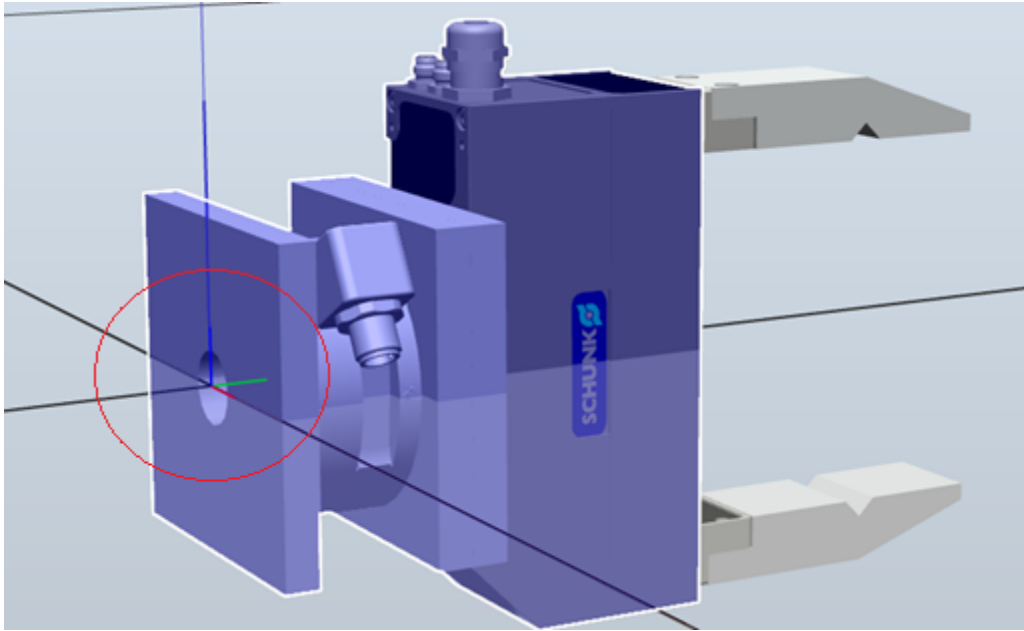
In this example, the geometries of the gripper is not modelled in a way that gives a correct attachment to the robot.



Detach the gripper from the robot, and take a closer look on the robots tool attachment point. When attaching a tool to the robot, the gripper will be attached to the robot according to this frame.



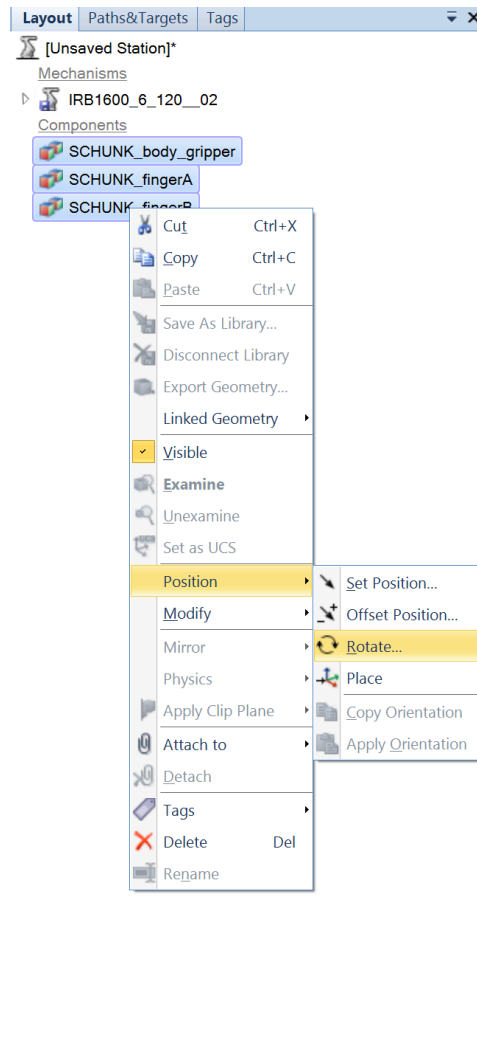
Next, take a look at the frame of your gripper.



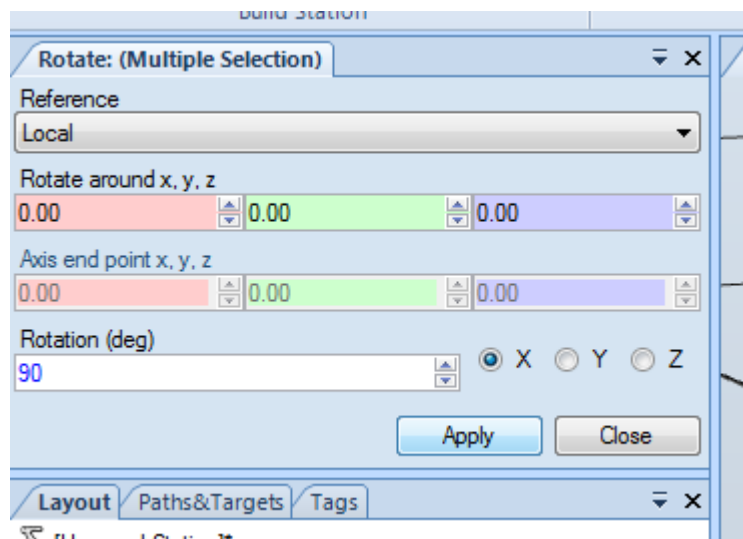
The reason that the gripper was not correctly attached to the robot is that the frame of the gripper and the frame of the attachment point of the robot DO NOT MATCH. So now there is two options, either you can correct the frame of the gripper trough the CAD modelling software, or you can do it through RobotStudio. For this example, we are going to do it through RobotStudio.

If you look back on the image where the gripper was attach to the robot, you will see that we have to alter the grippers frame, so that the jaws/fingers are pointing up along the z-axis. Additionally, it would also be nice if the “SCHUNK” logo were pointing away from the robot, so that it is visible.

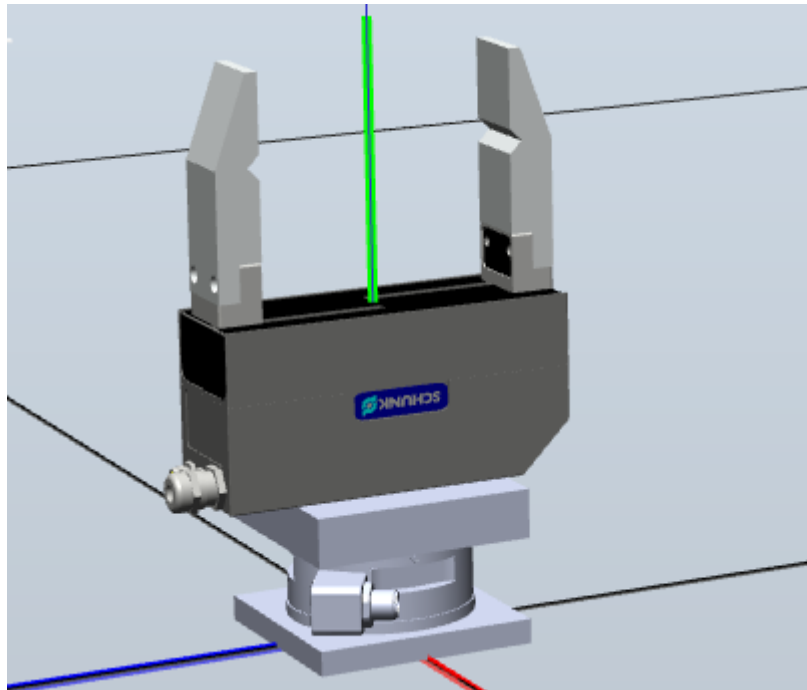
Looking at the image above, we will have to make the jaws point upwards in the z-direction by rotating the gripper around the x-axis. Start by marking all of the geometries of the gripper, right click on the mouse and select “Rotate” under “Position”.



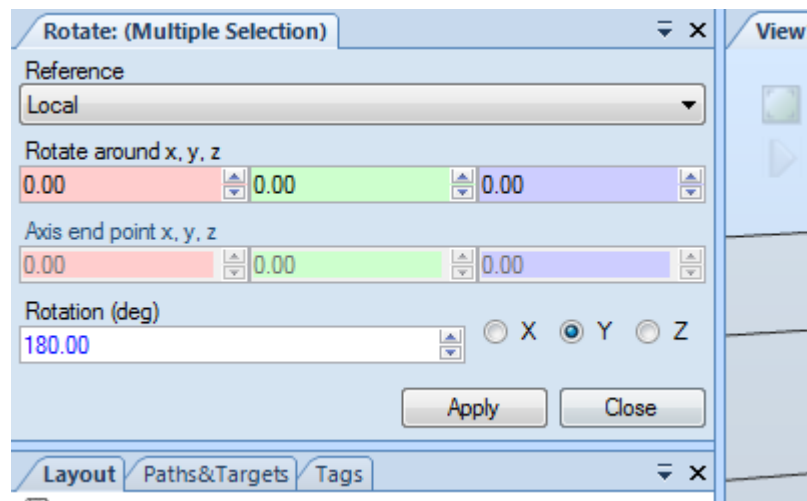
In the “Rotate” window, select the “Local” reference, and rotate 90 degrees around the x-axis.



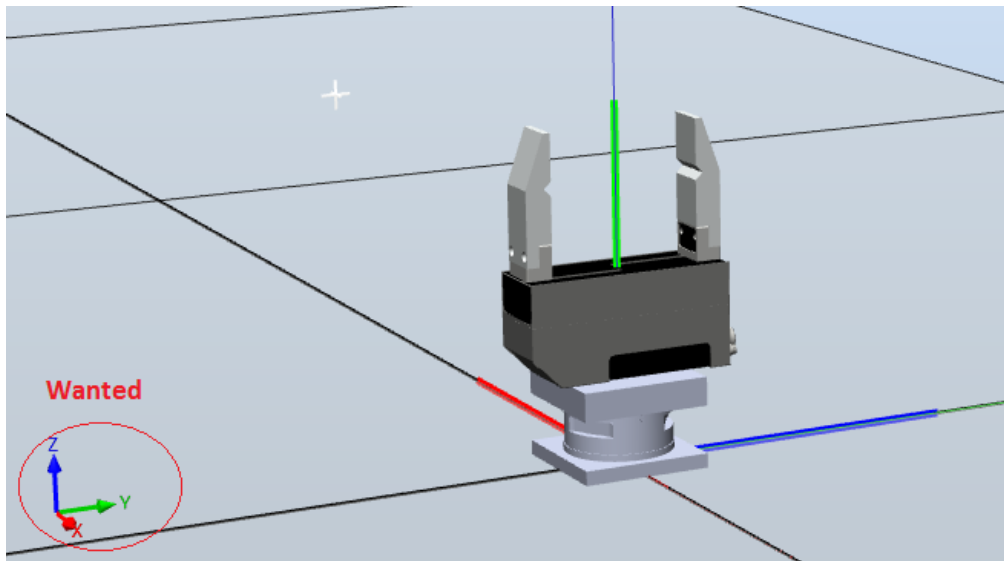
Your gripper should now look like this.



To make the “SCHUNK” logo visible when attach to the robot, rotate the gripper 180 degrees around the y-axis.

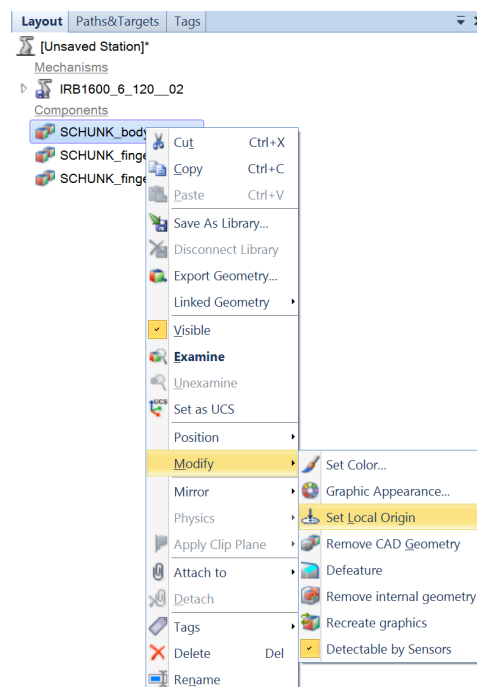


Now we have rotated the gripper so that it has the desired configuration, and should look like this.

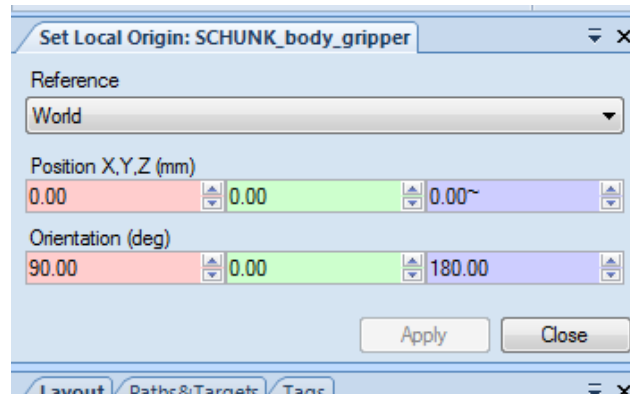


But as you can see on the image above, we have actually not changed the reference frame of the gripper, since we wanted the jaws to point towards the z-axis. The reason why we have done the rotation of the gripper in this matter is that it becomes much easier to actually change the frame of the gripper, now that we have the correct configuration according to the world coordinate.

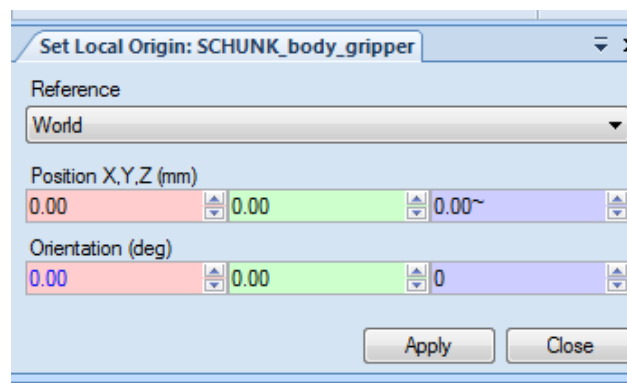
To change the frame of the gripper, start by marking the “body” geometry of the gripper, and click on “Set Local Origin” under “Modify”.



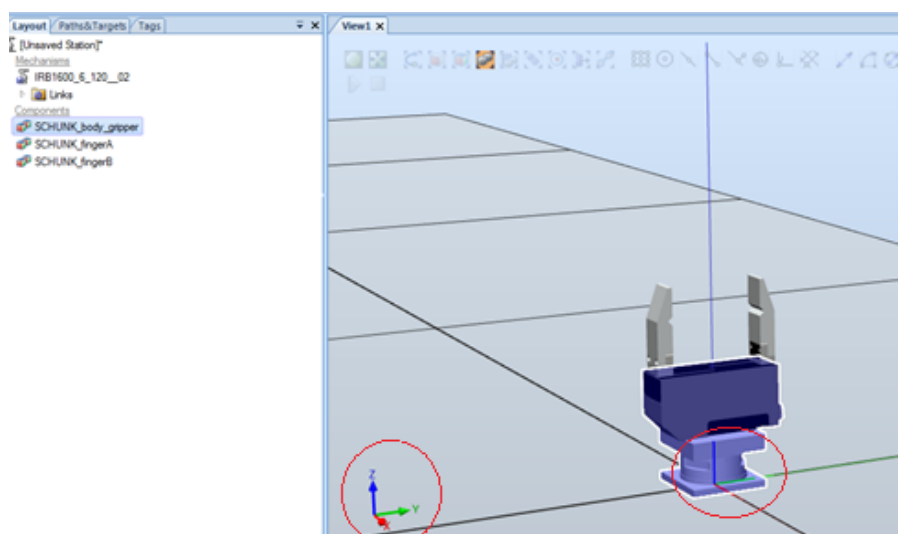
If you select the “World” reference, you will see that the frame of the gripper is oriented with 90 degrees in the x- and 180 degrees in the z-axis.



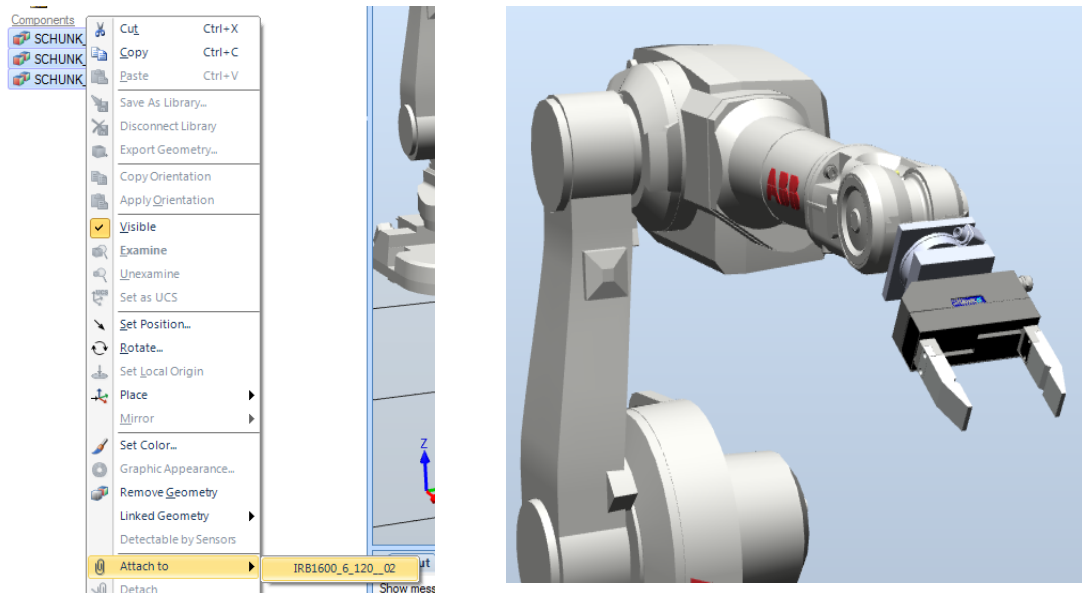
Just change these values to zero and apply.



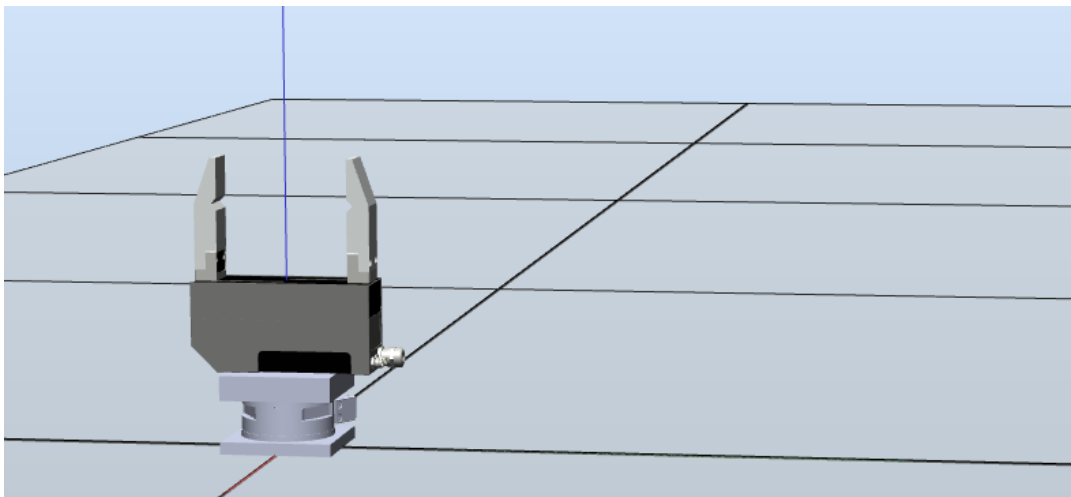
You can now see that the frame of the gripper has been altered and that the jaws are pointing upwards in the z-axis. Now do the exact same procedure for the rest of the gripper's geometries.



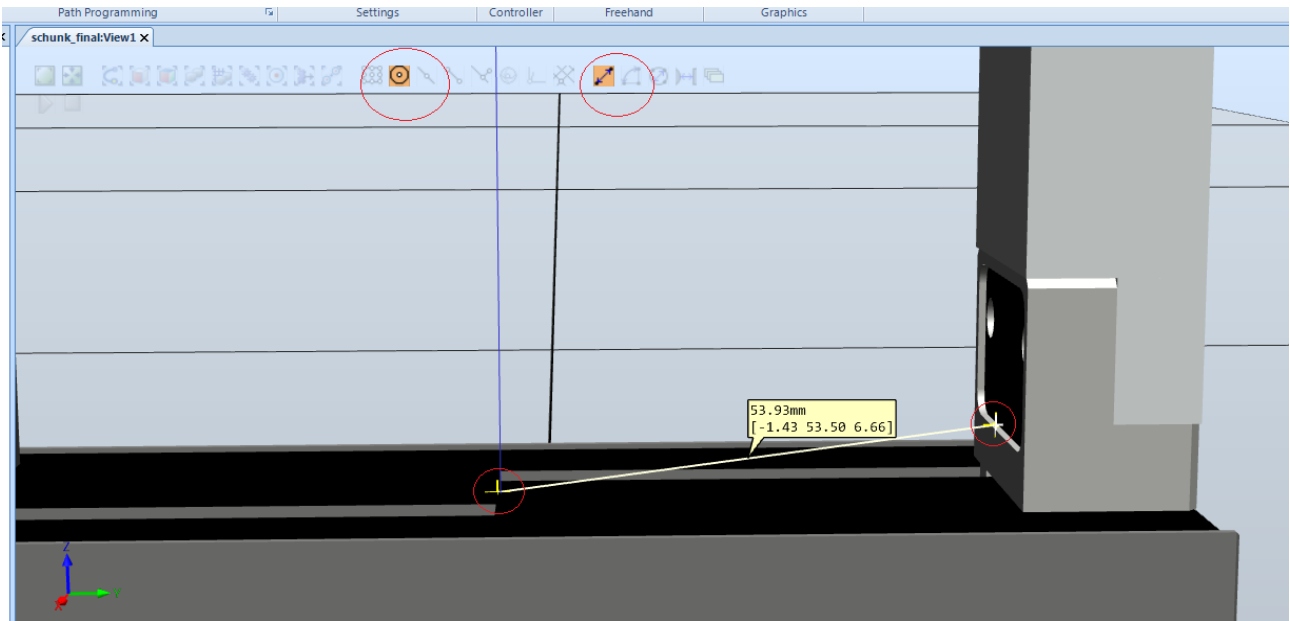
If we mark all of the grippers geometries and attach them to the robot, we will now see that the gripper is correctly attach to the robot (or at least how we want it to be attached).



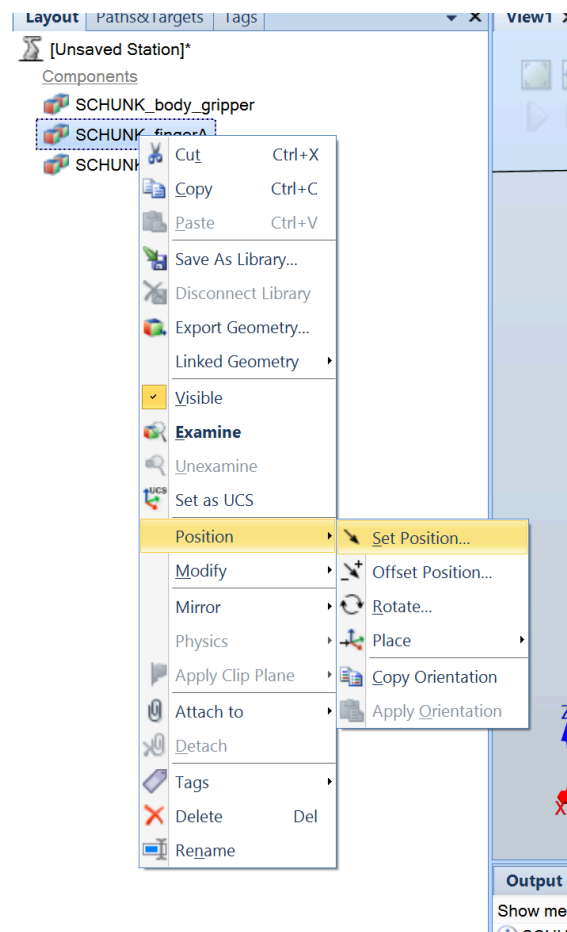
Next, detach all the geometries of the gripper from the robot, and delete your robot, so only the tool is represented in your station.



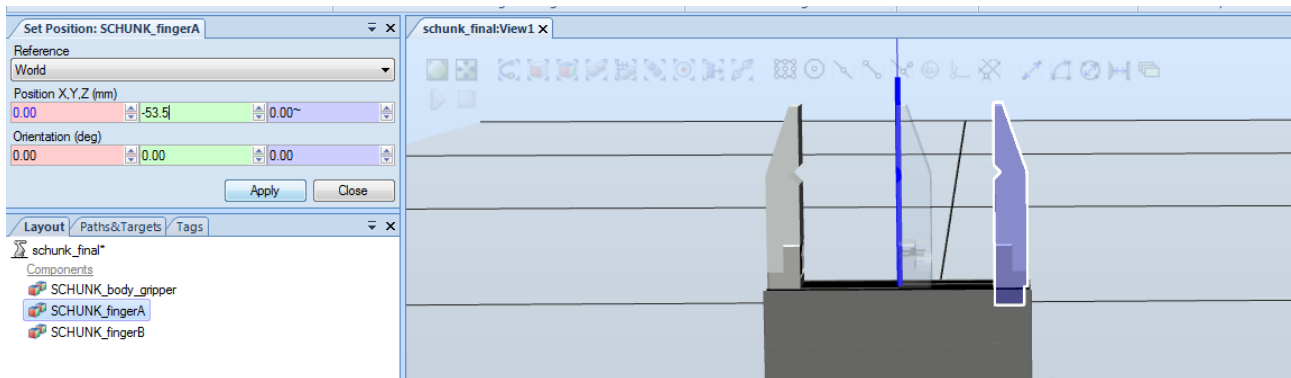
The last thing to do is to close the jaws of the gripper. The reason for this is that it will make life so much easier when we are going to make the jaws move later on. The first thing we have to do is to get an indication of how far we are going to move the jaws towards the center. Click on the “Snap Center” and “Point to Point” buttons, and measure the distance from the center of the gripper towards the closest part of a jaw. We can see that the distance that we have to move the jaw is 53.5 mm in the y-direction.



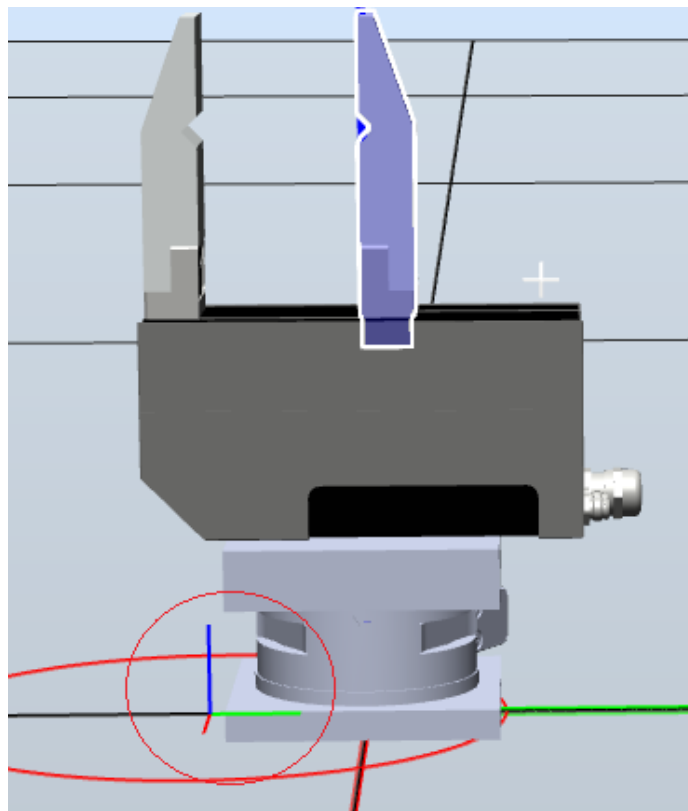
Mark one of the geometries for the tools jaw and click “Set Position” under “Position”.



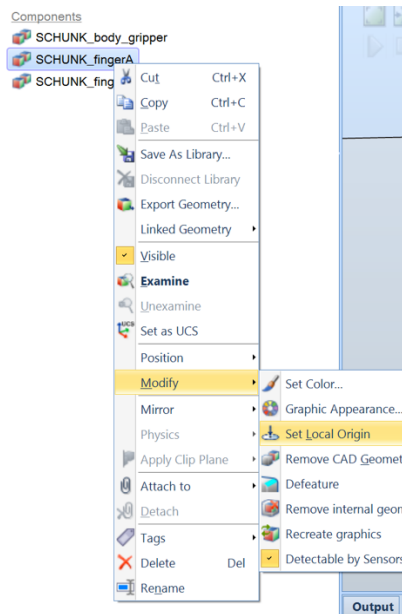
Move the position of the jaw so that it reaches the center.



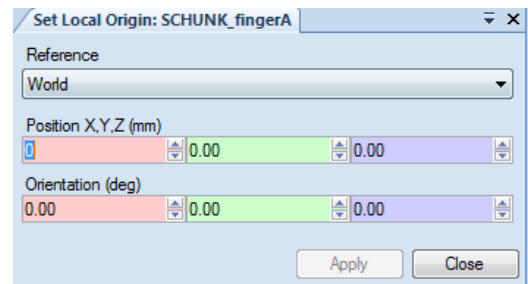
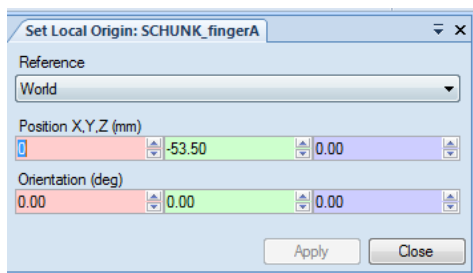
However, we still want the reference frame of the jaw to be in the center of the gripper. Right now we have just shifted the jaw geometry, as seen in the image below.



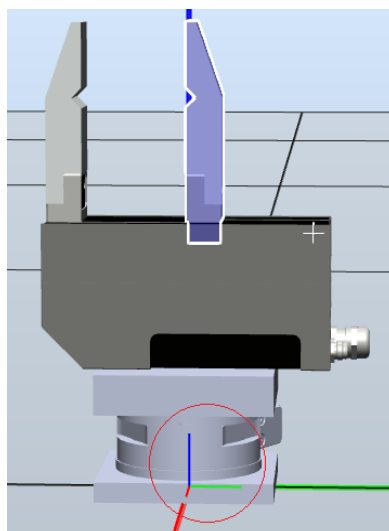
Go to “Set Local Origin”



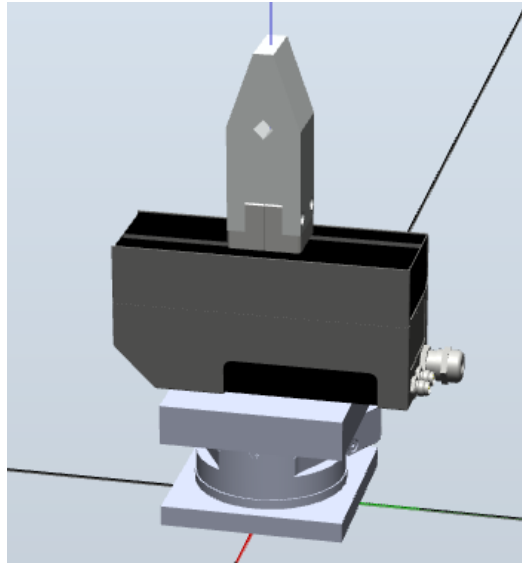
And change the value to zero



The reference frame will then move back to the centre of the gripper. Do the exact same procedure for the other jaw.

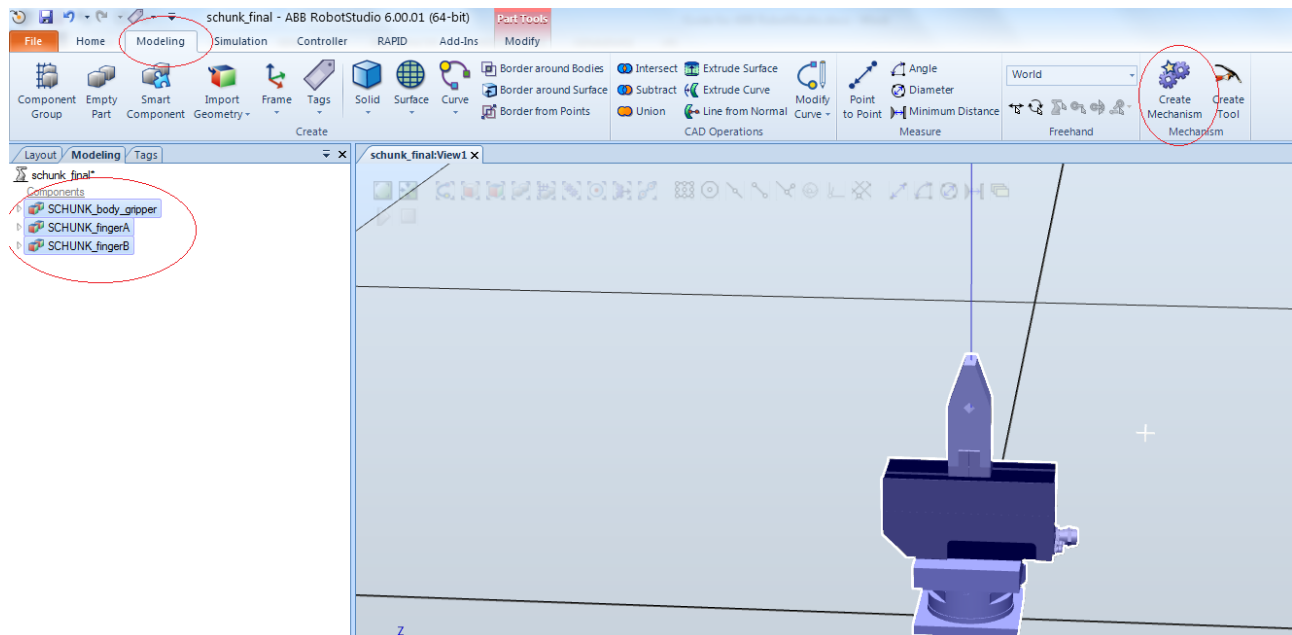


You will now have a representation of your tool that looks like this.

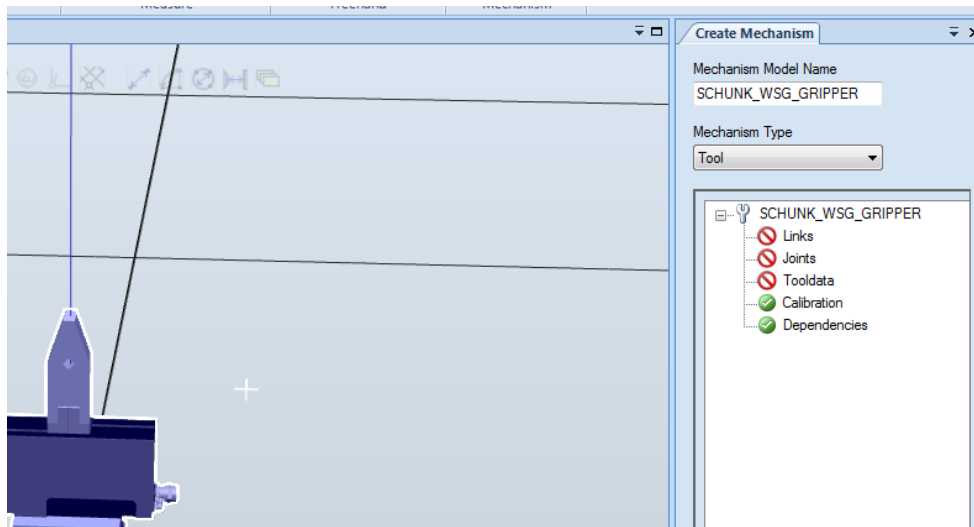


Creating a tool mechanism of your tool geometries

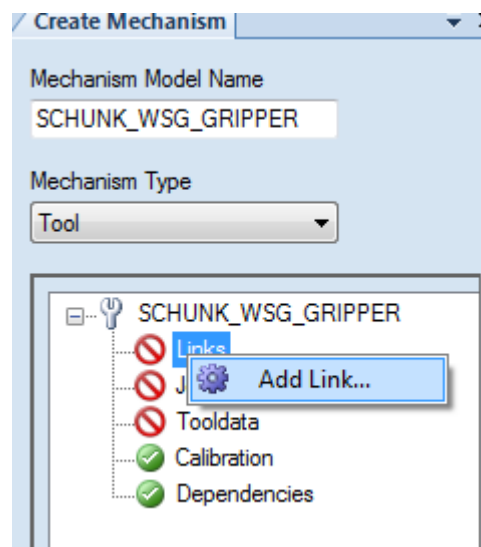
Right now, your tool is actually nothing but a graphical representation. Therefore, the next step is to create a tool mechanism of the tool geometries. Click on the “Modelling” tab in the top of your screen, mark all of your tool geometries, and click “Create Mechanism”.



Type what you want the tool to be called and select that you want to create a tool mechanism.



Right click on “Links” and select “Add Link”



Type what you want the link to be named, select the body part, set it as Baselink, press the arrow, and press apply.

Create Link

Link Name
L_Body

Selected Part:
SCHUNK_body_gripper

☒ Set as BaseLink

Added Parts
SCHUNK_body_gripper

Remove Part

Selected Part

Part Position (mm)
0.00 0.00 0.00

Part Orientation (deg)
0.00 0.00 0.00

Apply to Part

OK Cancel Apply

Continue to do the same for the jaws. You will see that you cannot set them as Baselink, but then again, you are not supposed to.

Create Link

Link Name
L_Finger_A

Selected Part:
SCHUNK_fingerA

☐ Set as BaseLink

Added Parts

Remove Part

Selected Part

Part Position (mm)
0.00 0.00 0.00

Part Orientation (deg)
0.00 0.00 0.00

Apply to Part

OK Cancel Apply

Create Link

Link Name
L_Finger_B

Selected Part:
SCHUNK_fingerB

☐ Set as BaseLink

Added Parts

Remove Part

Selected Part

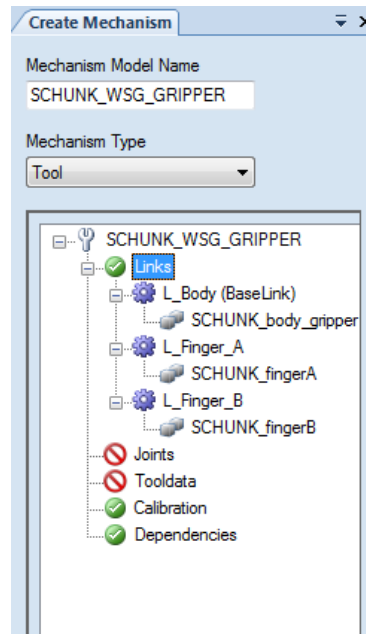
Part Position (mm)
0.00 0.00 0.00

Part Orientation (deg)
0.00 0.00 0.00

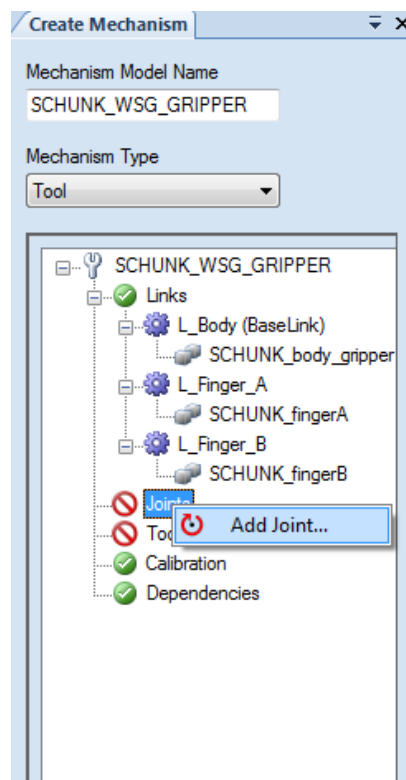
Apply to Part

OK Cancel Apply

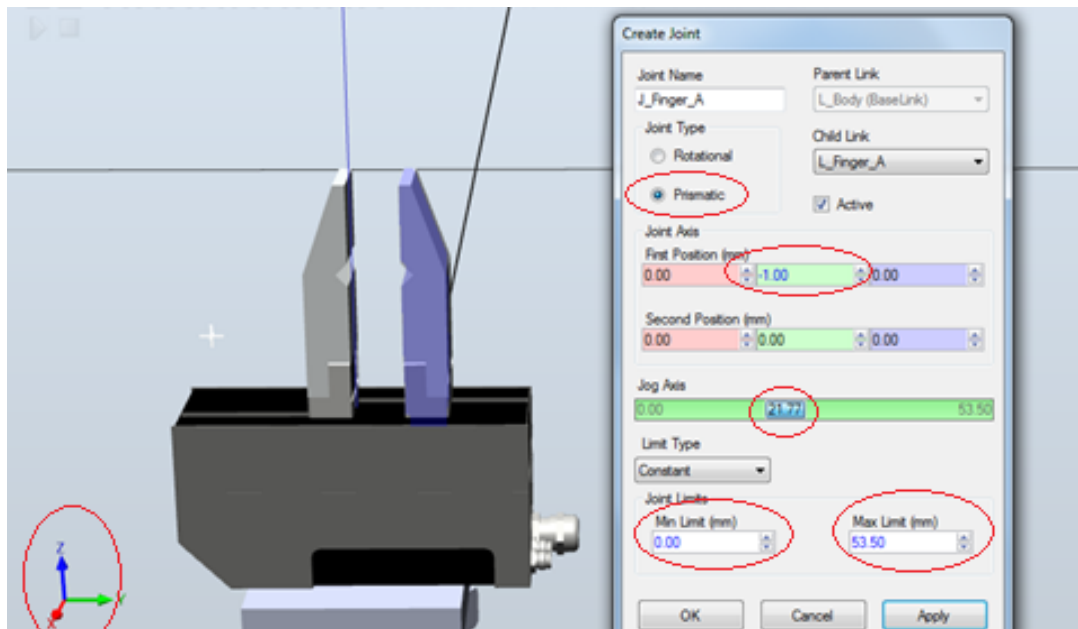
Your links should now be defined, and it should look like this.



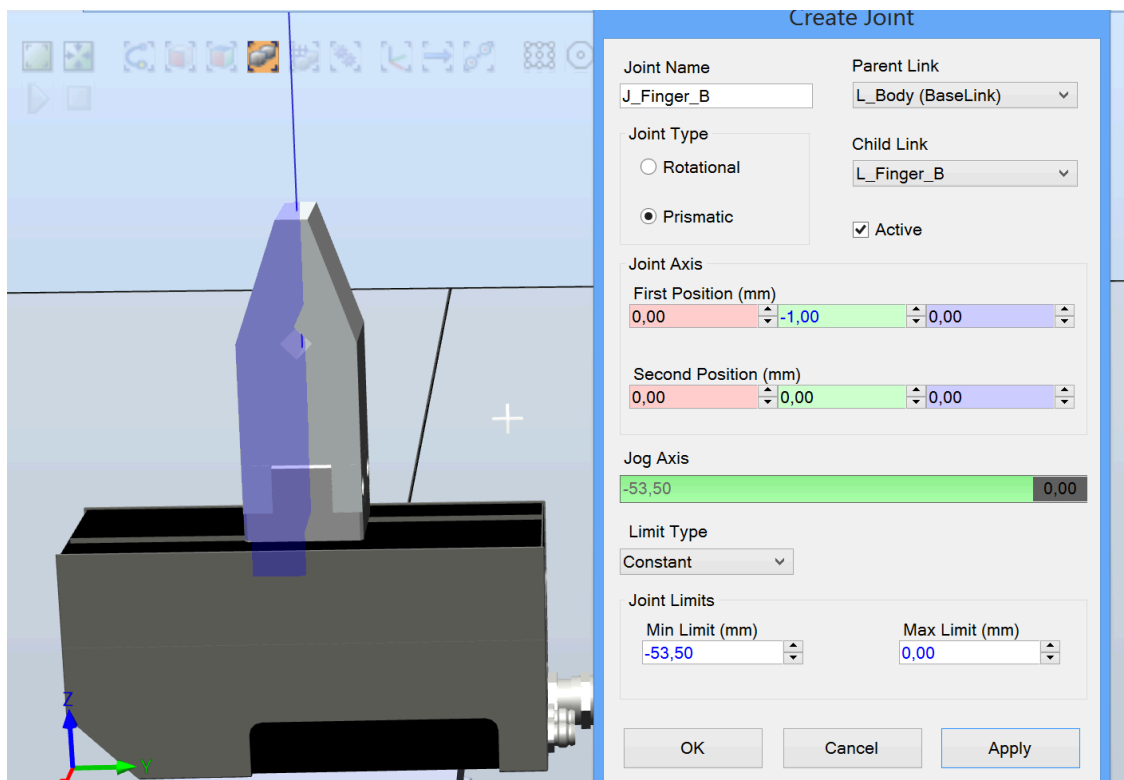
Next, right click "Joints" and select "Add Joint"



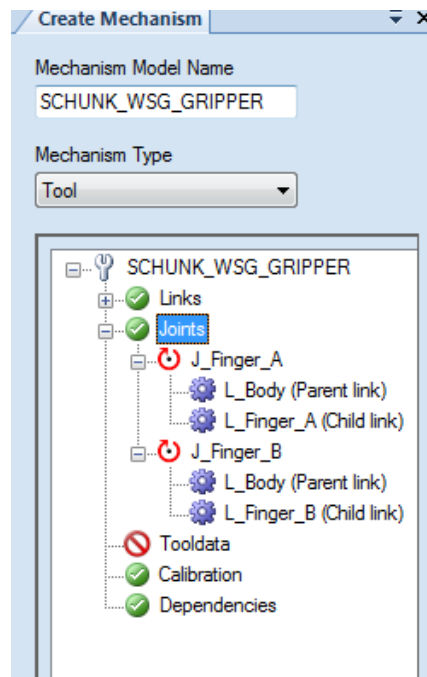
Type what you want to call the joint, select “Prismatic”, since we want to create a parallel gripper. Type “-1” in “Joint Axis”, this will indicate in which direction we want to move the jaw. We are typing “-1” because we want positive values when we are going in a “positive” y-direction. Set the joint limits to “0”, and “53.5”, and press “Apply”.



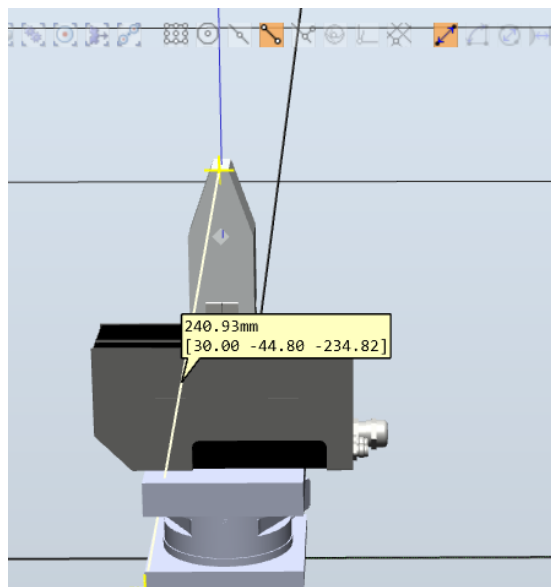
Do the same thing for the other jaw, BUT REMEMBER to select your baselink as “Parent Link”



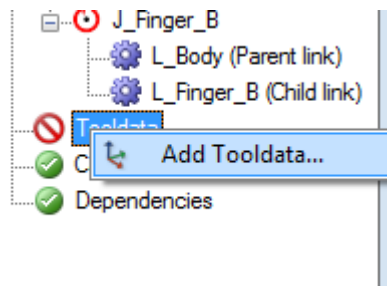
Your joints should now be defined, and it should look like this.



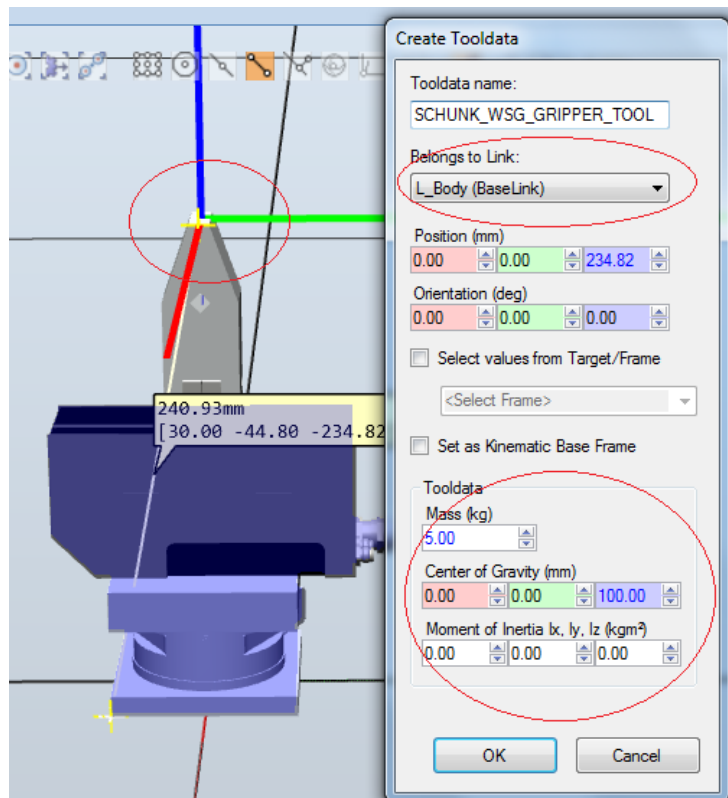
The last thing to do is to define the “Tooldata”, which essentially is your tool frame, otherwise known as the TCP (Tool center point). Where to define your TCP is entirely up to you, but in this example we are going to define it at the tool tip. So first, we measure the distance from the gripper frame to the tool tip. We can see that the tip is allocated “234.82” mm in the z-direction.



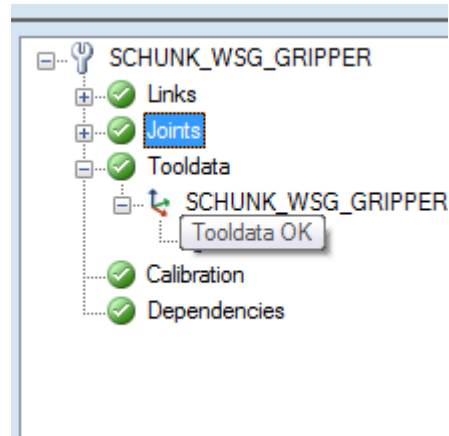
Right click on "Tooldata" and select "Add Tooldata".



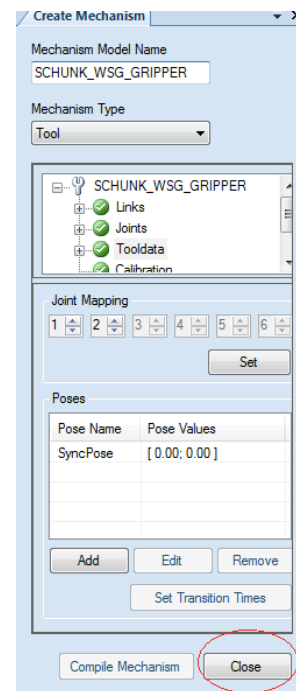
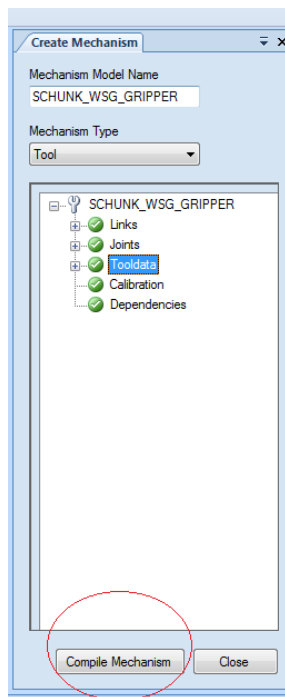
Type the name you want, select that it belongs to the BaseLink (we don't want our tool point to move whenever the jaws are moving), Insert "234.82" in the z-direction, and now you will actually be able to see where its going to create the tool point. The more information you have of your tool, the more "real" will the simulation be. In this case, we just type in some numbers (you decide).



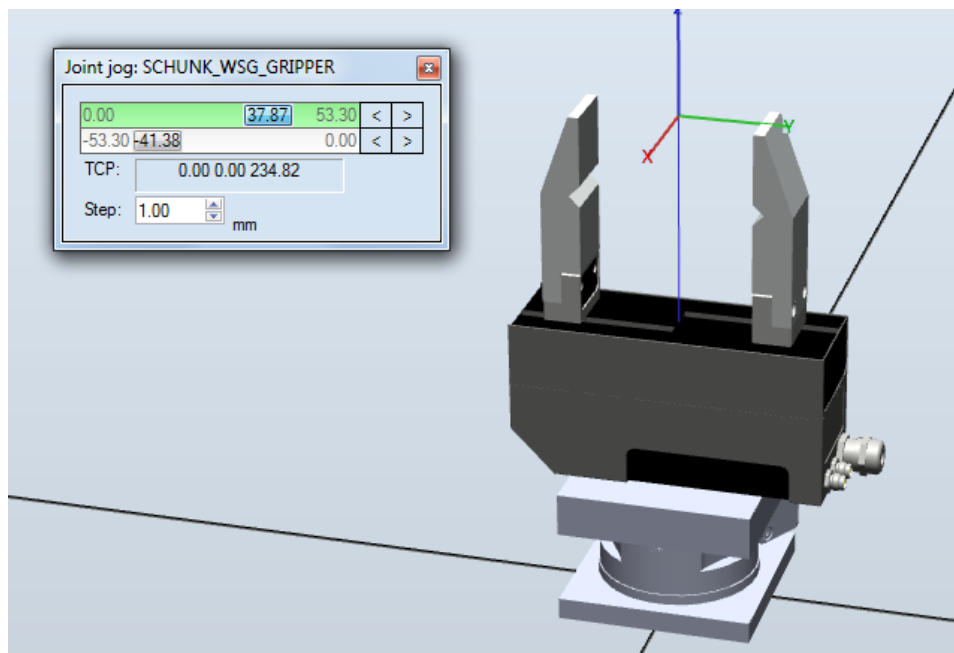
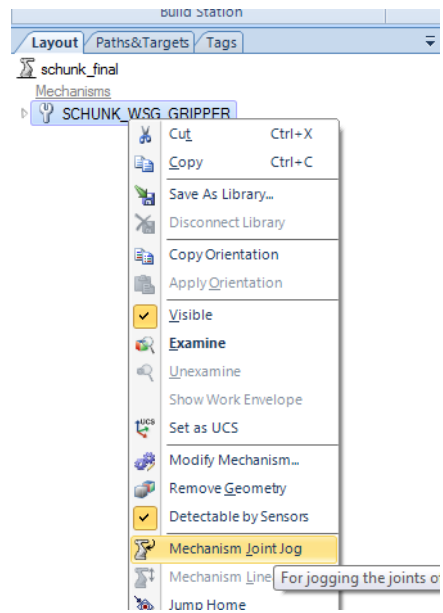
Your Tooldata should now be defined, and it should look like this.



Click “Compile Mechanism”, and afterwards click close.



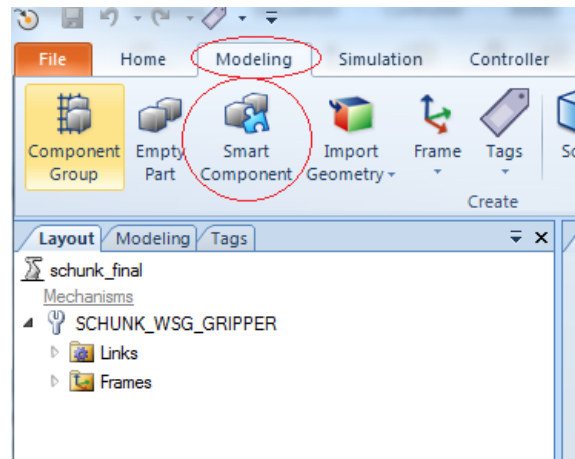
You have now created tool mechanism, feel free to play with it.



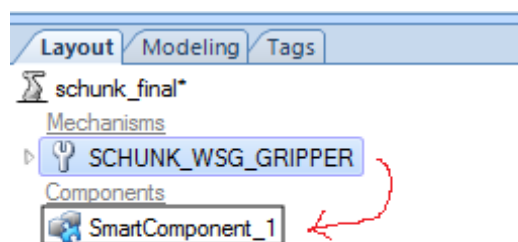
Creating a Smart Component

So now we have successfully created a tool mechanism, but the gripper is at the moment actually not able to grasp objects. To accomplish this we have to create some logic..some SMART logic. In RobotStudio smart components are used to add sensors, logic and actions to, among others, mechanism.

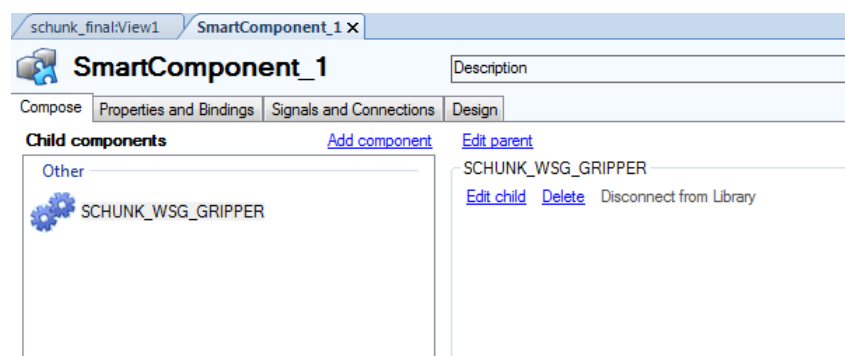
Go to “Modelling” and click “Smart Component”



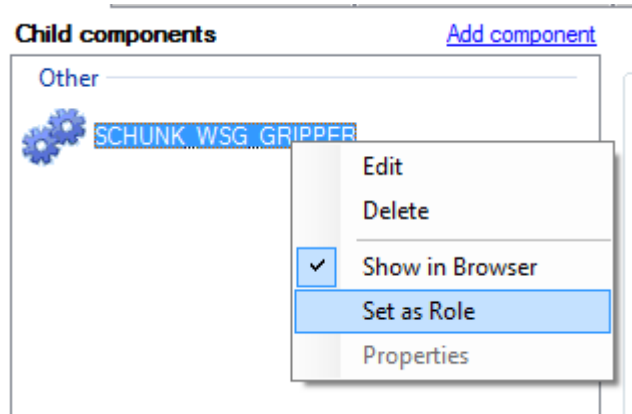
Drag your tool mechanism to the created smart component



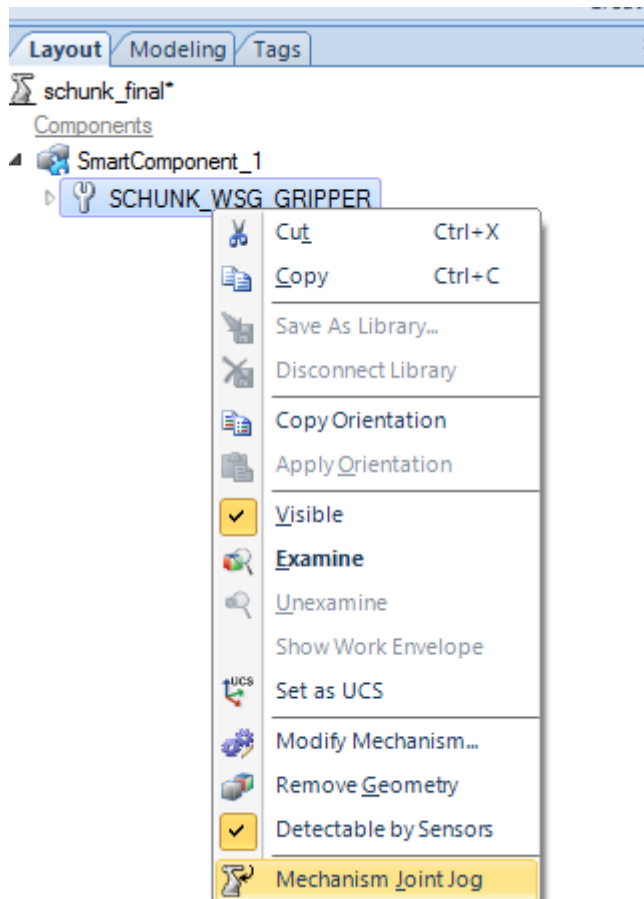
The smart component will then look like this

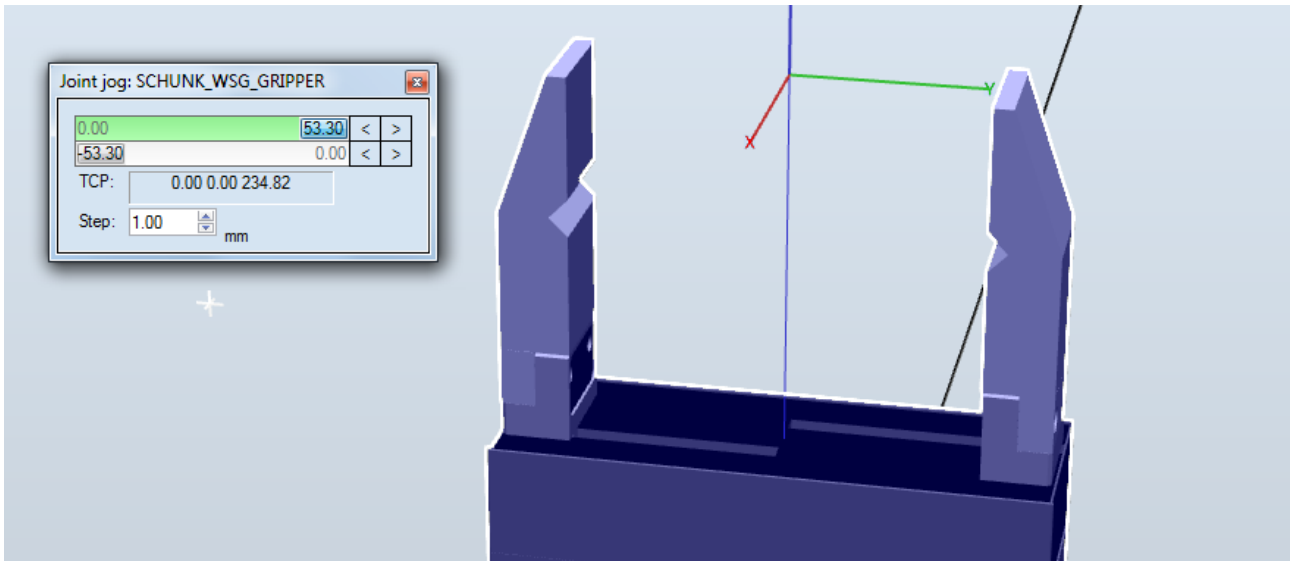


Right click on the component and set it as “Role”. More or less to say that this is our primary component.



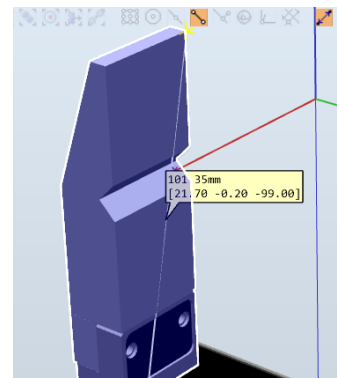
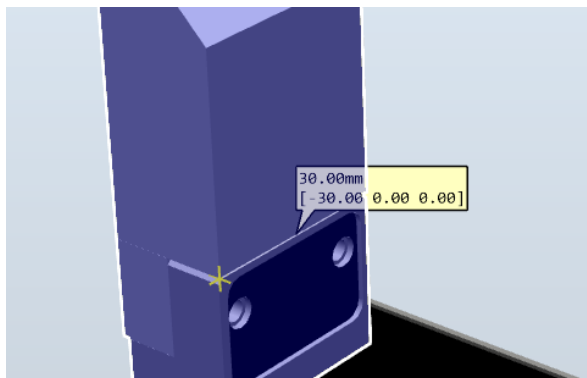
Next jog the jaws, so that the gripper is wide open.



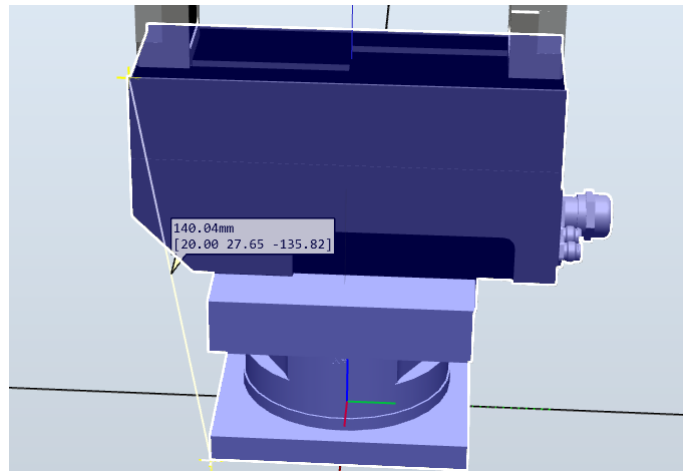


The reason for opening the gripper is that we want to create sensors to detect if there are any objects touching the jaws. To do this, we want an indication of how big the sensors should be, and is much easier when the jaws are accessible to measure. If you already know the dimensions of your jaws, you of coarse do not need to measure them, but opening the gripper will still be a good thing for some of the later steps.

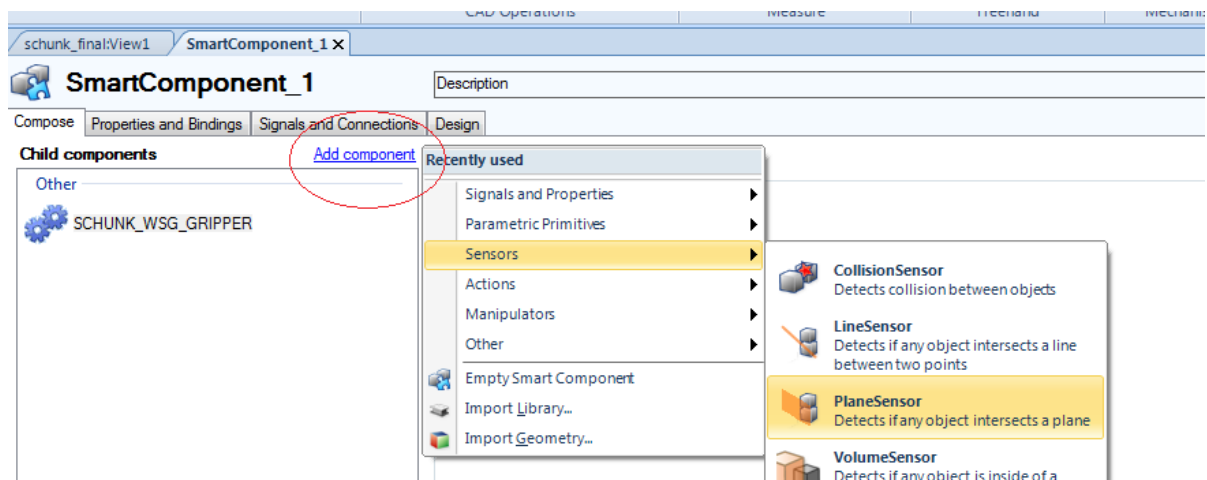
Measure the width and height of your jaws. In this case, the width is 30 mm and the height is 99 mm.



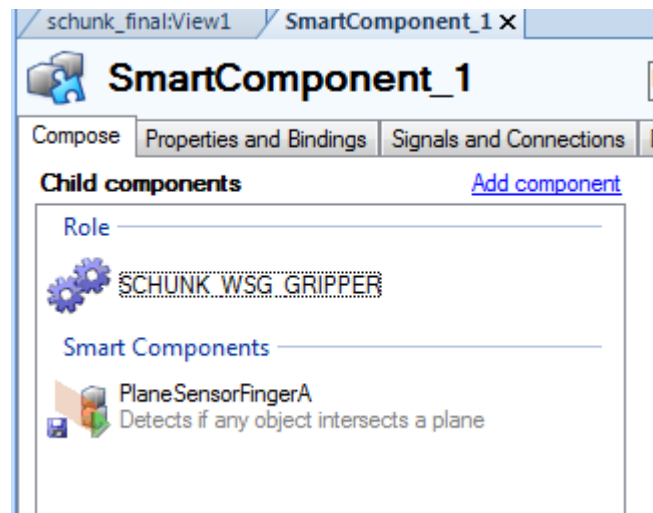
Next, measure the distance from the bottom of the gripper to the surface where the jaws can move. In this case, 135.82 mm in the z-direction.



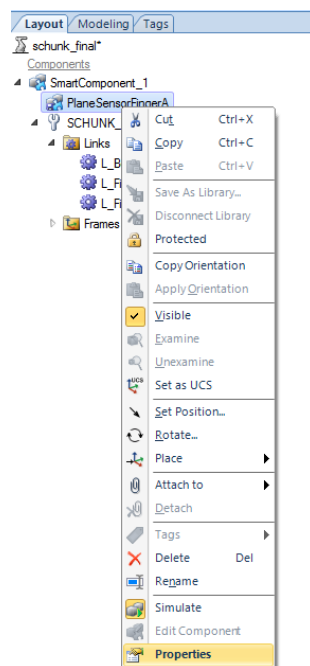
Go back to the smart component view, click “Add component” and select “PlaneSensor”. A sensor plane is basically a sensor that can be used to detect objects.



This will create a plane sensor component. Here the sensor has been renamed to “PlaneSensorFingerA”, since we are going to make two of them (one for each jaw).



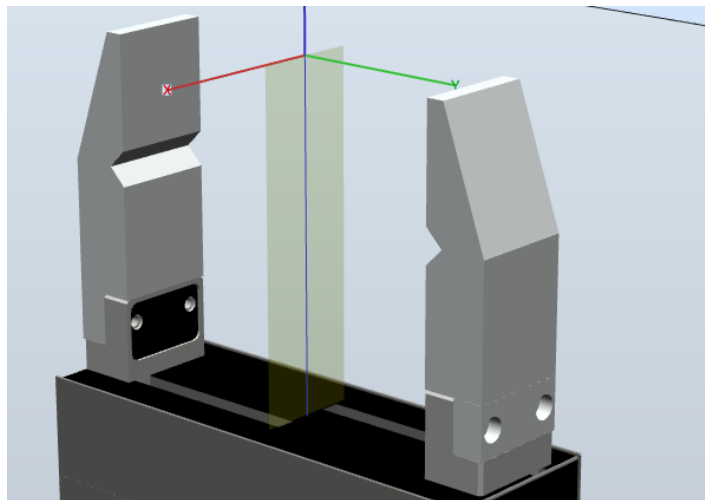
Right click on “PlaneSensorFingerA” in the layout and select “Properties”



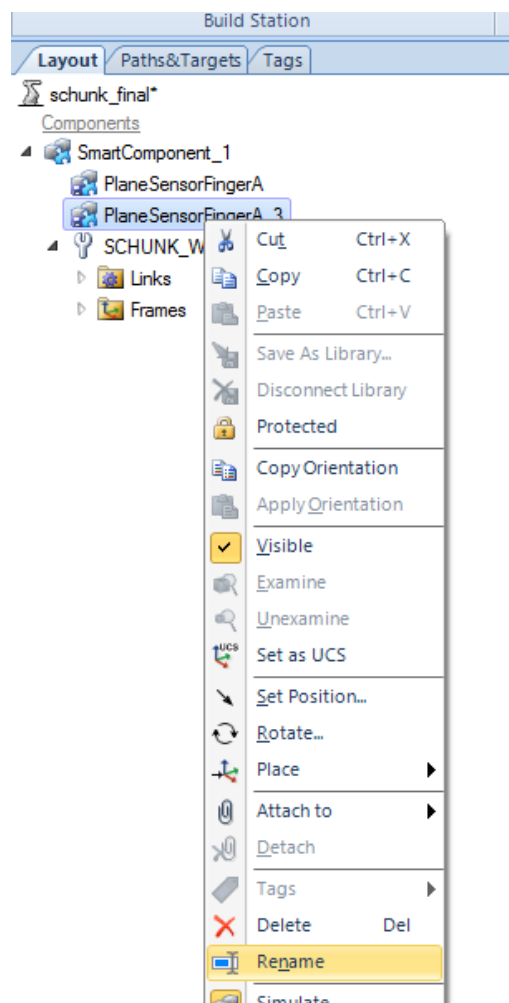
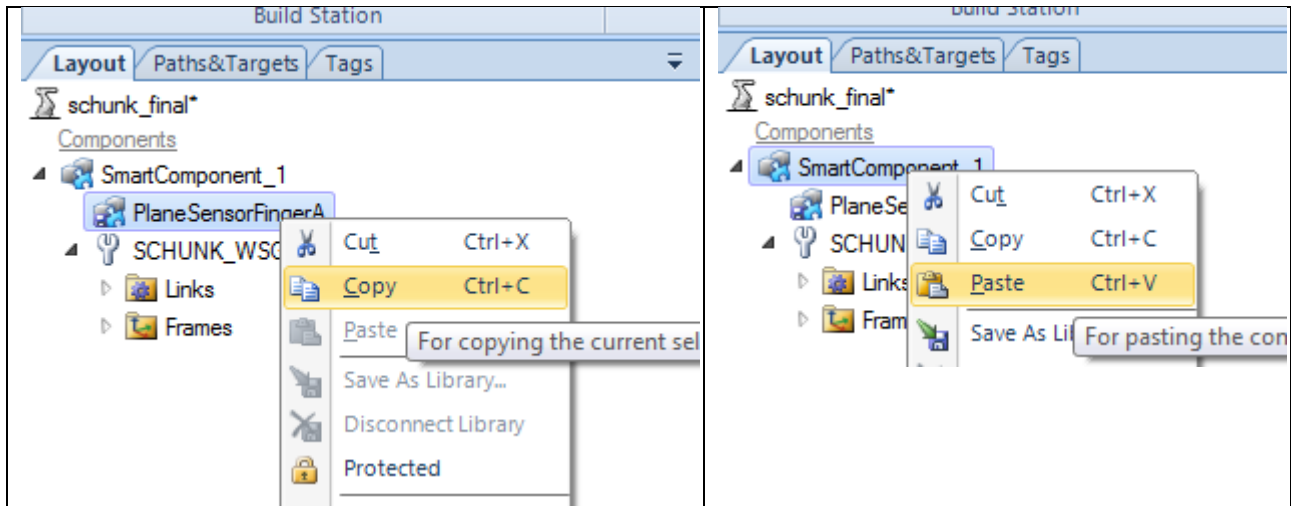
Type in the values that you measured and press apply. The reason for the “-15” in the x-direction is because the frame of the sensor plane has its origin in (0,0,0). We measured a width of 30 mm, and want an evenly distributed sensor at the jaw.

Properties			
Origin (mm)	-15.00	0.00	135.82
Axis1 (mm)	30.00	0.00	0.00
Axis2 (mm)	0.00	0.00	99.00
SensedPart			
Signals			
Active			
SensorOut			
Apply Close			

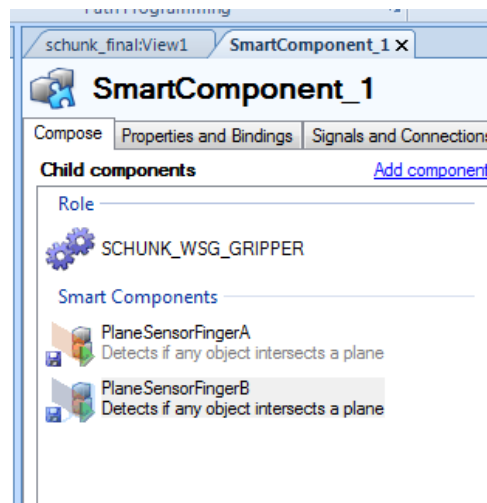
This will create a sensor plane that looks something like this.



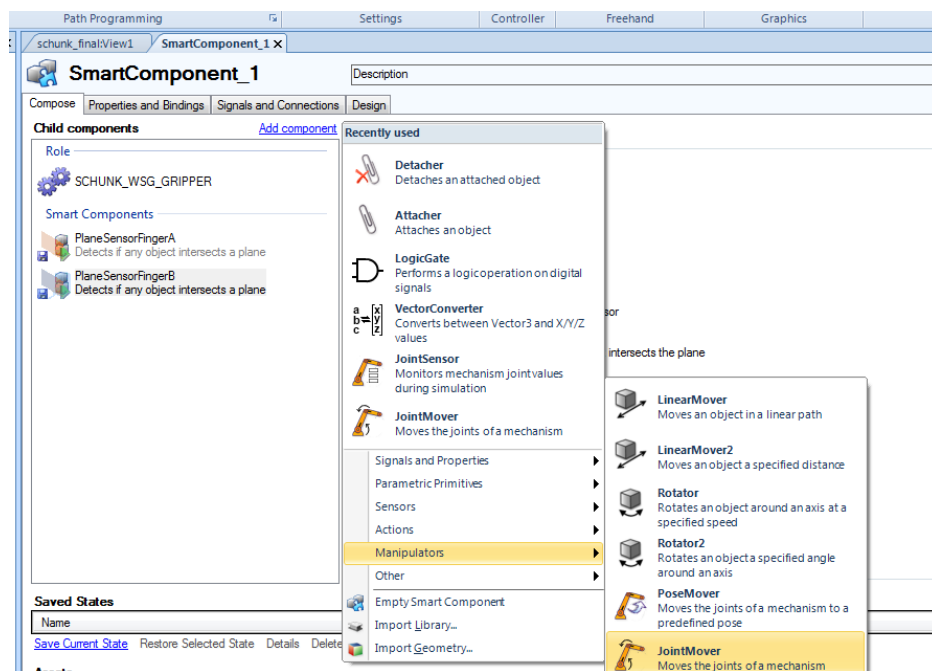
Next, we will create a sensor plane for the other jaw, but since the jaws are identical we can just create a copy of “PlaneSensorFingerA”, paste it in to “SmartComponent_1”, and rename it to “PlaneSensorFingerB”.



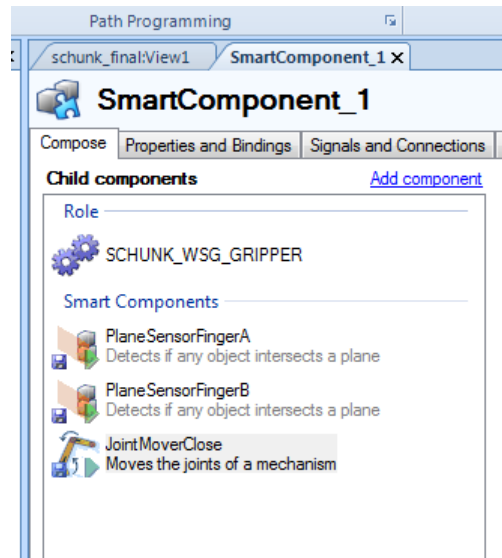
Now your smart component should look like this.



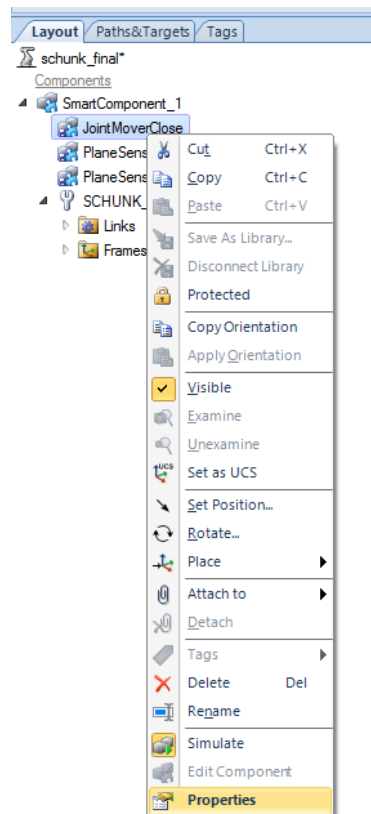
Next, we will add a component called “JointMover”, to actually be able to move the jaws.



Rename it to “JointMoverClose”, and your smart component should look like this.



Right click on “JointMoverClose” and select properties.



Select the “SCHUNK_WSG_GRIPPER” as the mechanism, specify how long you want it take to for the gripper to close (it can be any number you want), and press apply. J1 and J2 are both set to zero, since these are the joint configuration to close the gripper.

Properties: JointMoverClose

Properties

Mechanism
SmartComponent_1/SCHUNK_WSG_GRIPPER

☐ Relative

Duration (s)
2

J1 (mm)
0.00

J2 (mm)
0.00

Signals

GetCurrent

Execute

Pause

Cancel

Executing

Paused

Apply Close

Create another “JointMover” component, and rename this to “JointMoverOpen” and type in these values.

Properties: JointMoverOpen

Properties

Mechanism
SmartComponent_1/Schunk_WSG_Gripper

☐ Relative

Duration (s)
2,0

J1 (mm)
53,50

J2 (mm)
-53,50

Signals

GetCurrent

Execute

Pause

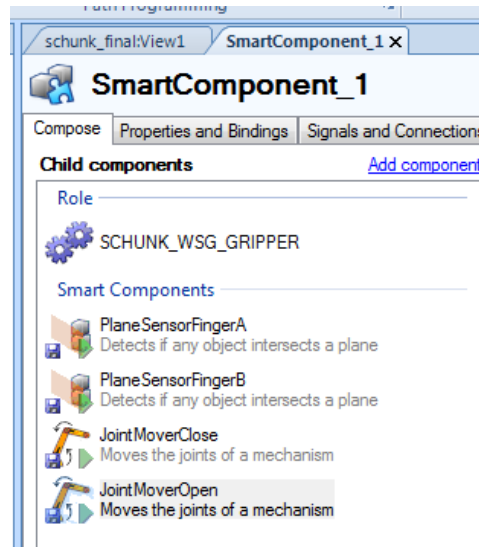
Cancel

Executing

Paused

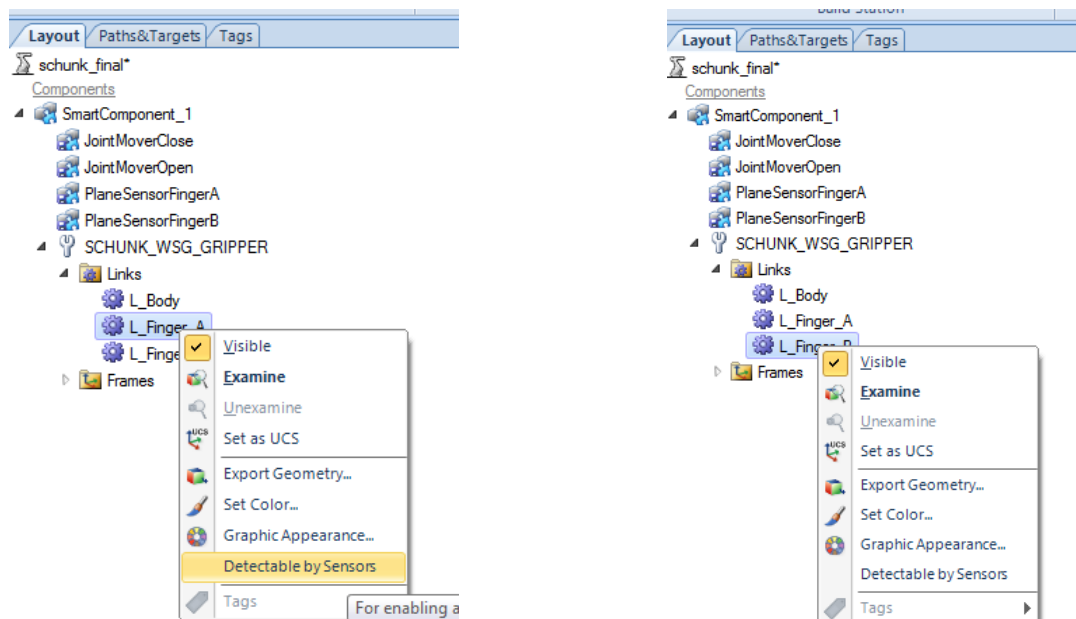
Apply Close

Now your smart component should look like this.

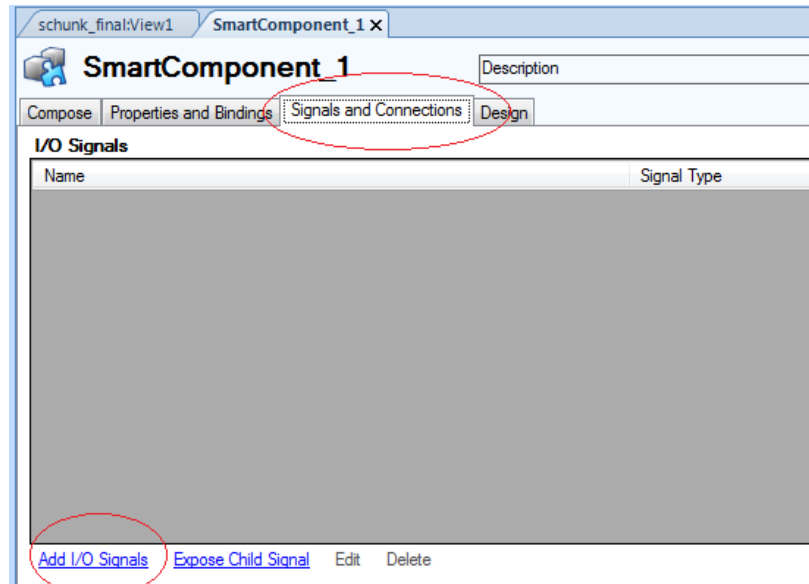


Since our sensor planes are going to be added to the surface of each jaw, they will actually trigger as they have detected an object, because they will have contact with the jaws. There are two ways of handling this: one way is to move the sensor planes so that they will have an offset relative to the jaws, the other way is to disable the jaws to be recognised by sensors. We are going to choose the second approach, well knowing that this would mean that the jaws are not recognised by any sensor that will be placed in the cell. This could be a disadvantage if for example you have some safety sensors that are to detect if the robot (+ tool) is crossing a specific boundary.

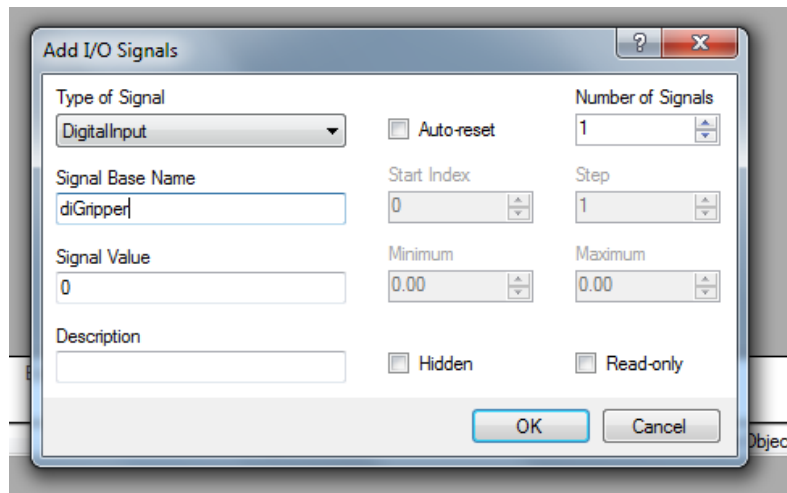
You disable sensor recognition of the jaws like this.



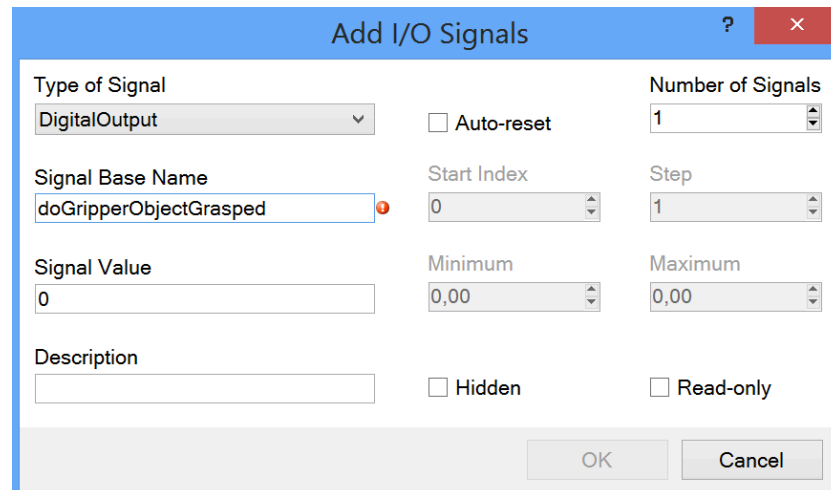
Next we will create some I/O signals for our smart component, so that we can actually command the gripper to grasp an object, and tell us if it has grasped an object. Go to “Signals and Connections” in the smart component view and click “Add I/O signals”.



Create a digital input that we will use to command the gripper to open and close, and name it “diGripper” (or whatever name you want).



Then create two digital output; “doGripperObjectGrasped” that we will use to validate if the gripper has grasped an object, and “doGripperExecuted” that we will use as feedback if a gripper action has been executed.



The 'Add I/O Signals' dialog box is shown with the following settings:

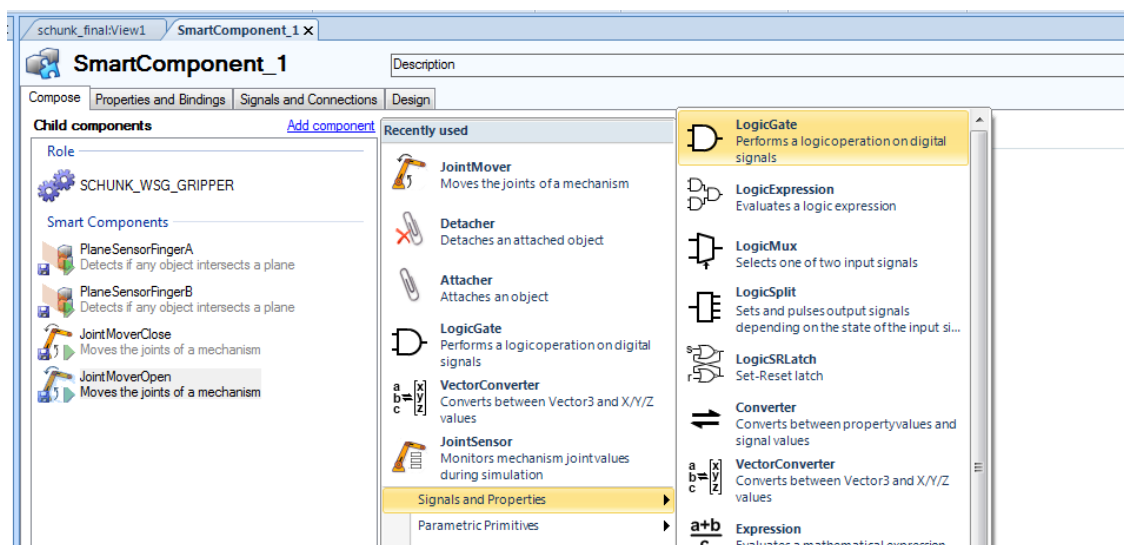
- Type of Signal:** DigitalOutput
- Signal Base Name:** doGripperObjectGrasped
- Signal Value:** 0
- Start Index:** 0
- Step:** 1
- Minimum:** 0,00
- Maximum:** 0,00
- Number of Signals:** 1
- Auto-reset:** ☐
- Hidden:** ☐
- Read-only:** ☐

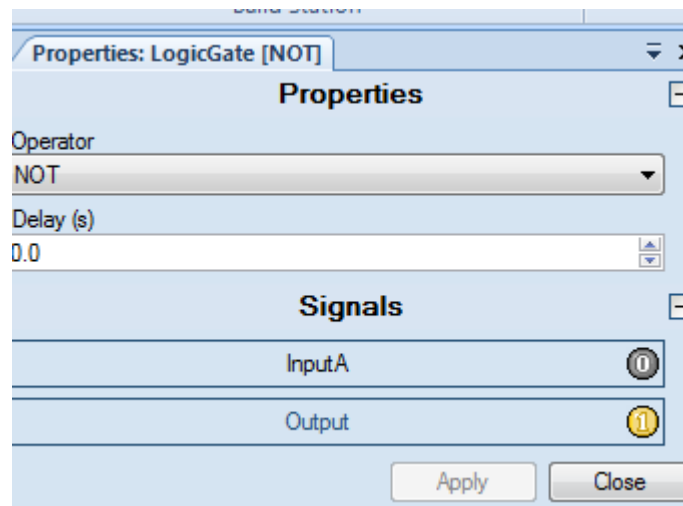
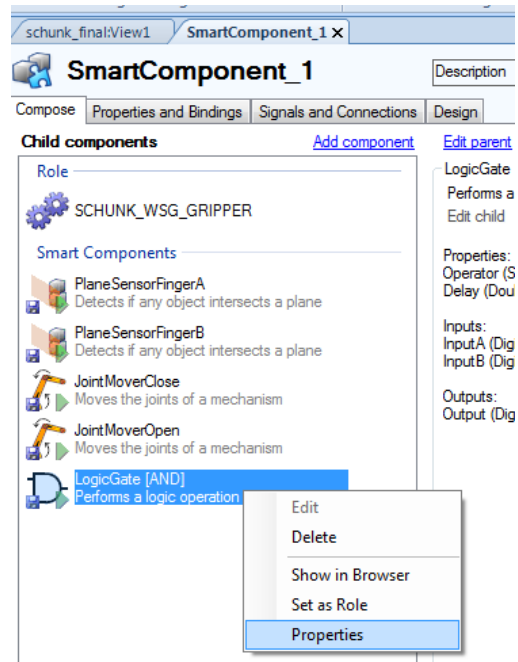
It should now look like this.

Name	Signal Type	Value
diGripper	DigitalInput	0
doGripperObjectGrasped	DigitalOutput	0
doGripperExecuted	DigitalOutput	0

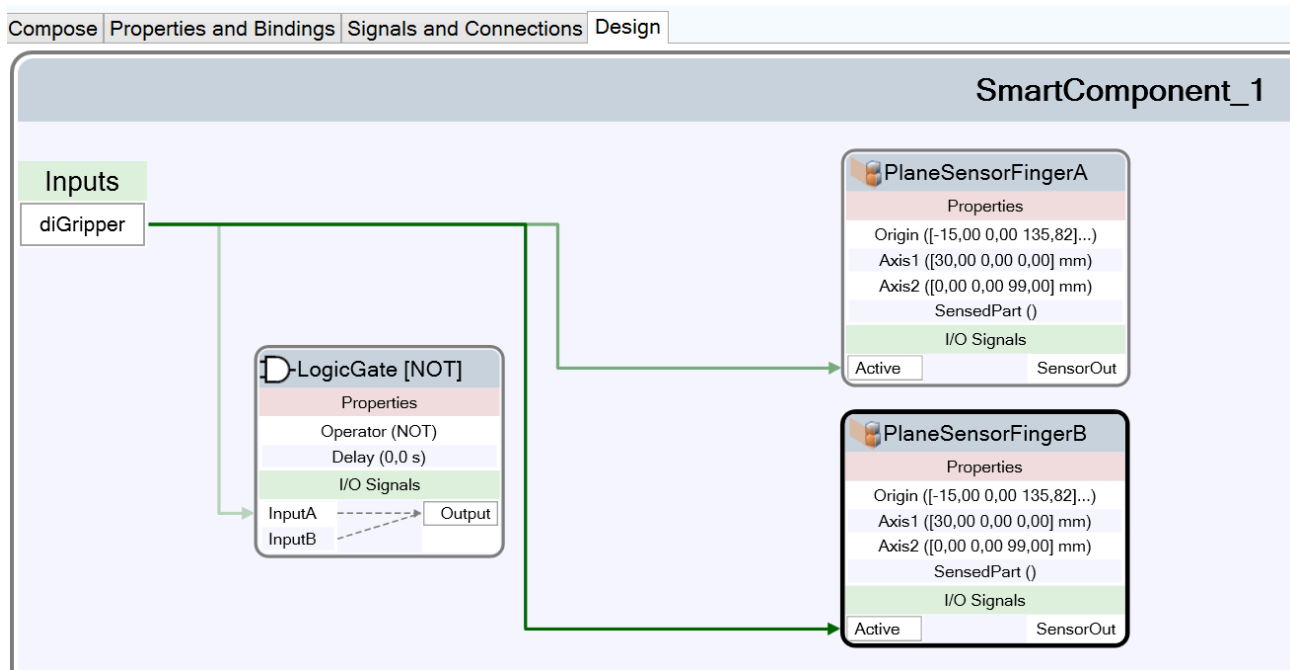
Buttons: Add I/O Signals, Expose Child Signal, Edit, Delete

Next, we will create a “Not” logic gate.

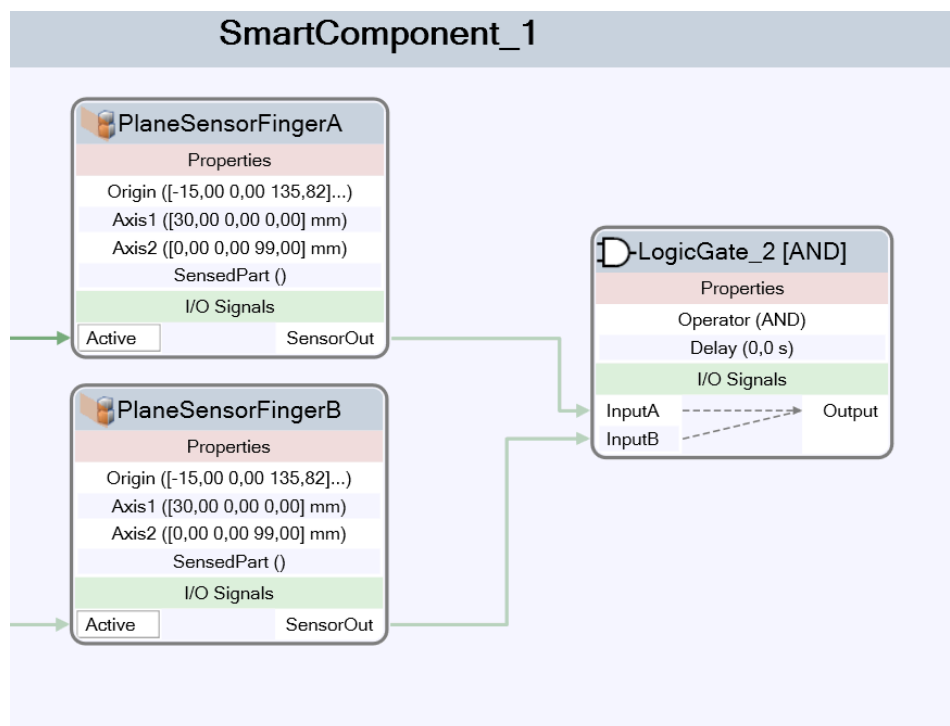




Next, we are going to “wire” the logic together to achieve that the sensor planes will always be active. Go to “Design” in the smart component view and drag the signals to acquire this configuration.

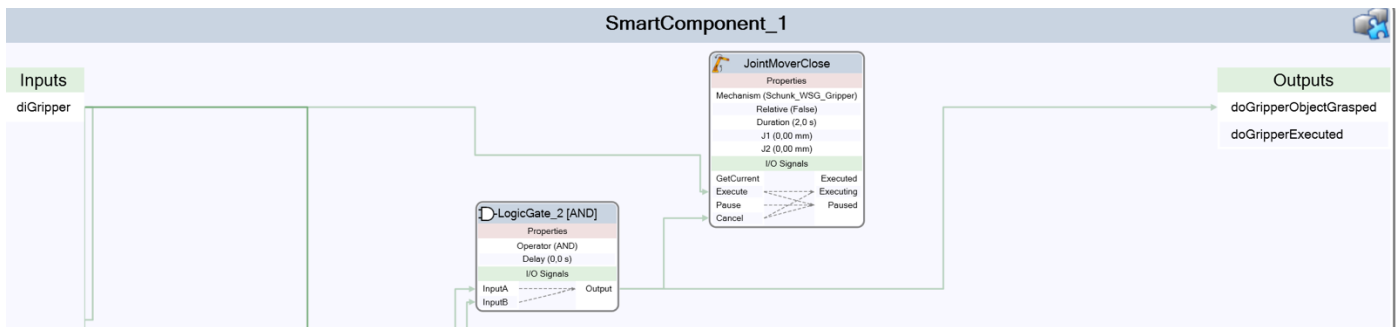


Create another logic gate, this time an “AND” gate. Configure it like this.



This way we will create a signal that will give an indication whether or not both sensors have detected an object. The output of the gate will be used to determine when we will “grasp” an object, and to determine if the gripper should continue moving the jaws.

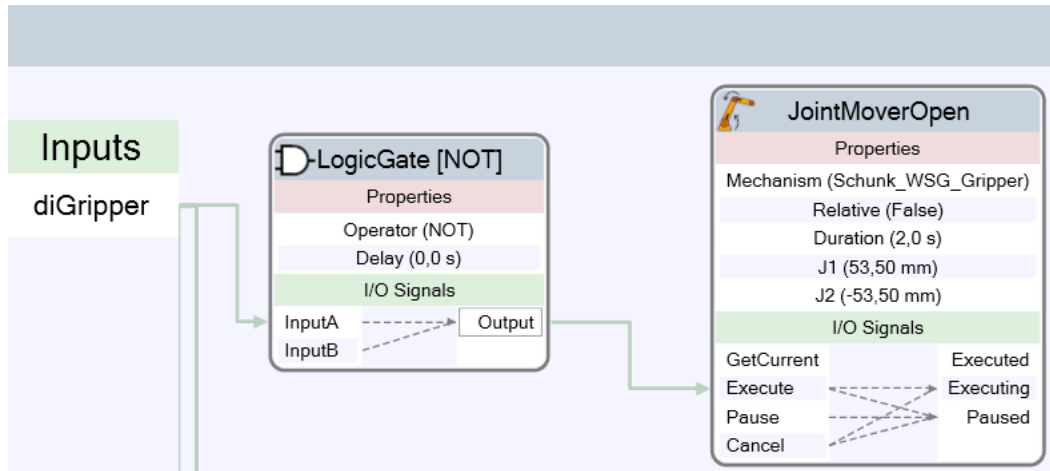
Next, we want to create the logic for closing the gripper.



What we have done is:

If `diGripper` is "1", we will start to close the gripper, but only as long as the two sensor planes do not detect any object, i.e. that the jaws are not grasping an object. In other words, we have made some logic that actually functions as a force feedback feature for the gripper. Furthermore, if both sensor planes detect an object, it means that we have contact with an object, and therefore can give a feedback that we have "grasped" an object.

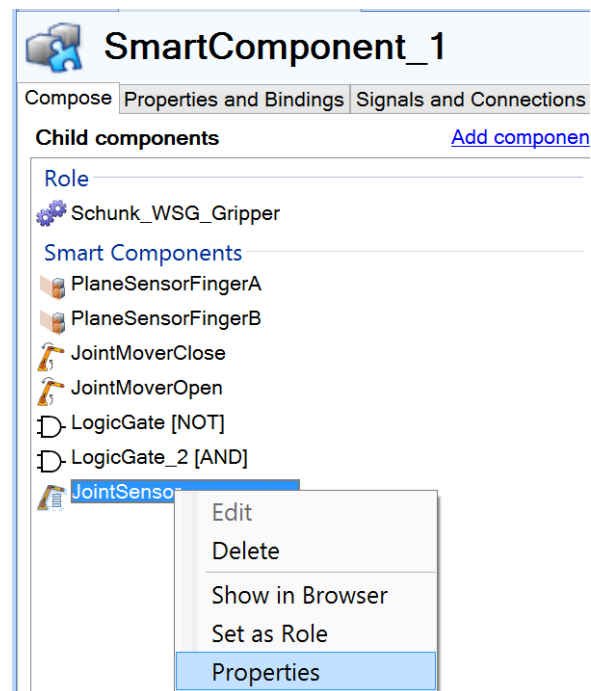
Next, we want to create the logic for opening the gripper. Since we already have the necessary components, we just have to link them together like this.



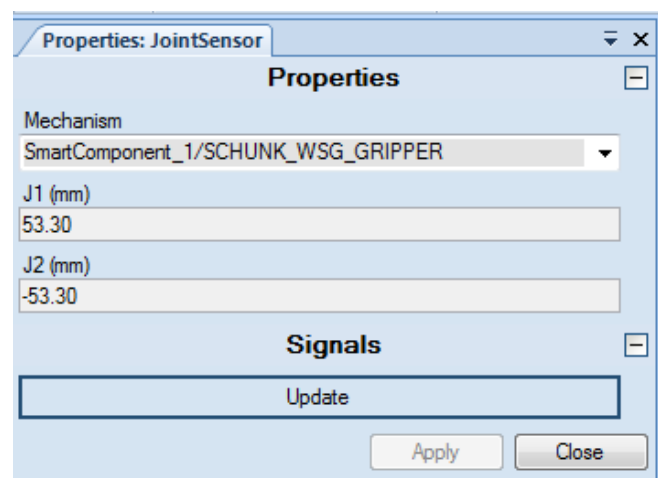
So if diGripper is “0”, we start to open the gripper.

Next, we have to make the sensor planes move together with the jaws, so they are not located in a fixed position. The first step to do that is to continuously monitor the jaws position whenever they are moving. Add a “JointSensor” component like this.

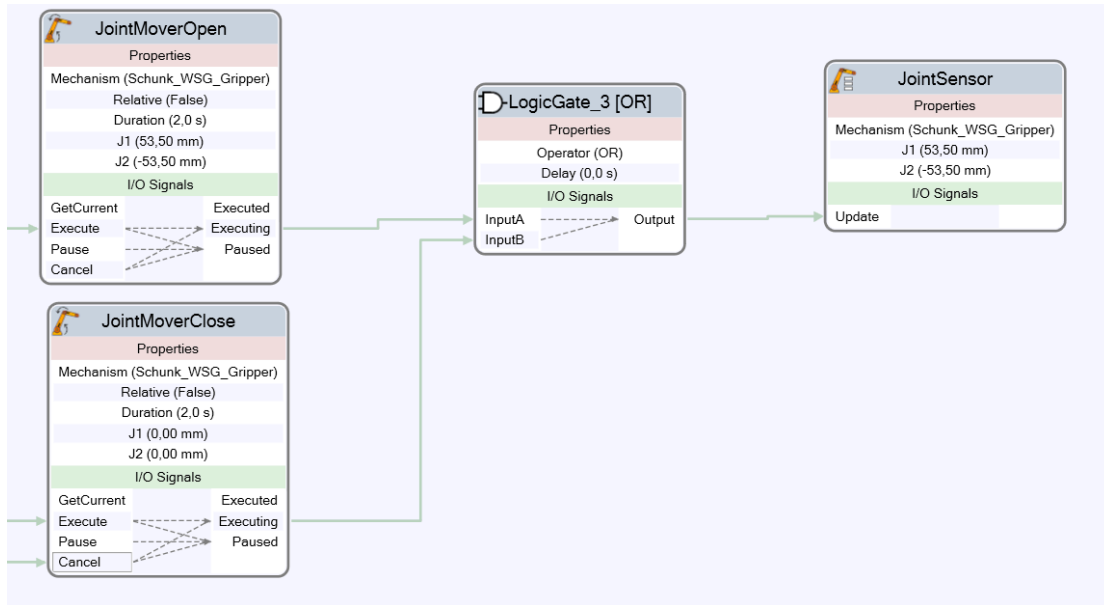
Right click and select properties



Choose the "SCHUNK_WSG_GRIPPER, and will show the current joint values (positions of the jaws).



Create a new “OR” logic gate, and configure the joint sensor like this.



Next, we want to use the values from the joint sensor to update the position of the sensor planes. We will first concentrate on updating “PlaneSensorFingerA”. The first step is to add TWO “VectorConverter” components.

Child components

[Add component](#)

- Role
- Schunk_WSG_Gripper
- Smart Components
- PlaneSensorFingerA
- PlaneSensorFingerB
- JointMoverClose
- JointMoverOpen
- LogicGate [NOT]
- LogicGate_2 [AND]
- JointSensor
- LogicGate_3 [OR]

Saved States

Name	Details
Save Current State	Restore Selected State Details Delete

Assets

Recently used

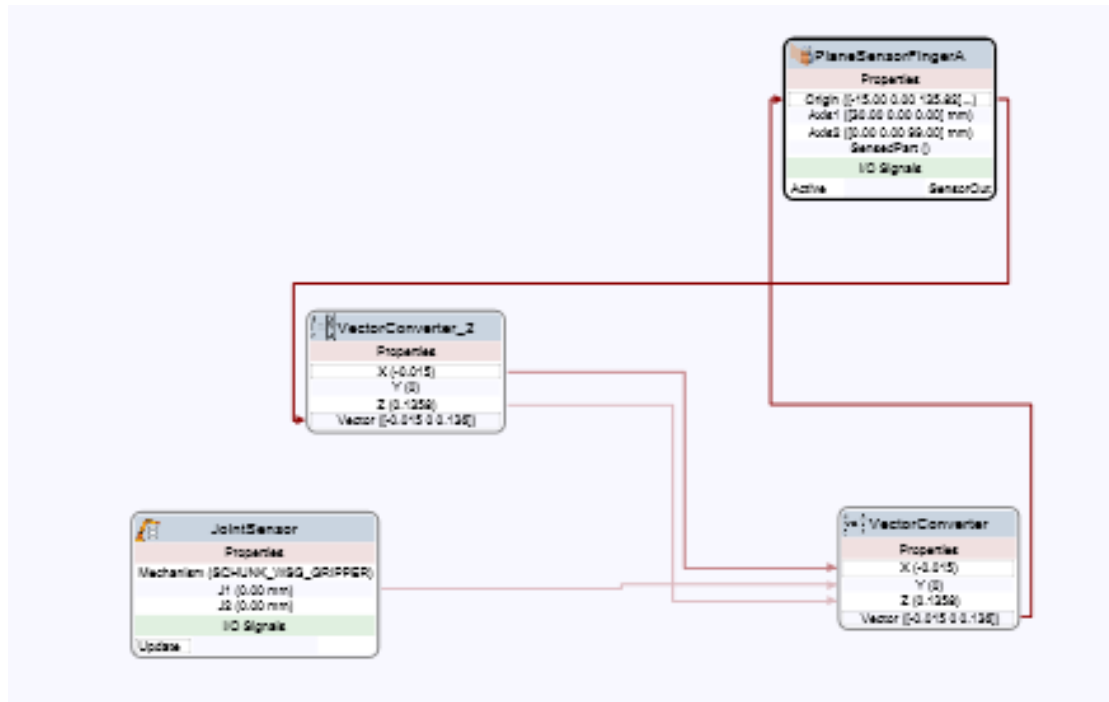
- LogicGate**
Performs a logic operation on digital signals
- JointSensor**
Monitors mechanism joint values during simulation
- JointMover**
Moves the joints of a mechanism
- PlaneSensor**
Detects if any object intersects a plane
- Camera**
Sample Component Description
- MoveToViewpoint**
Moves the active view to a predefined Viewpoint

Signals and Properties

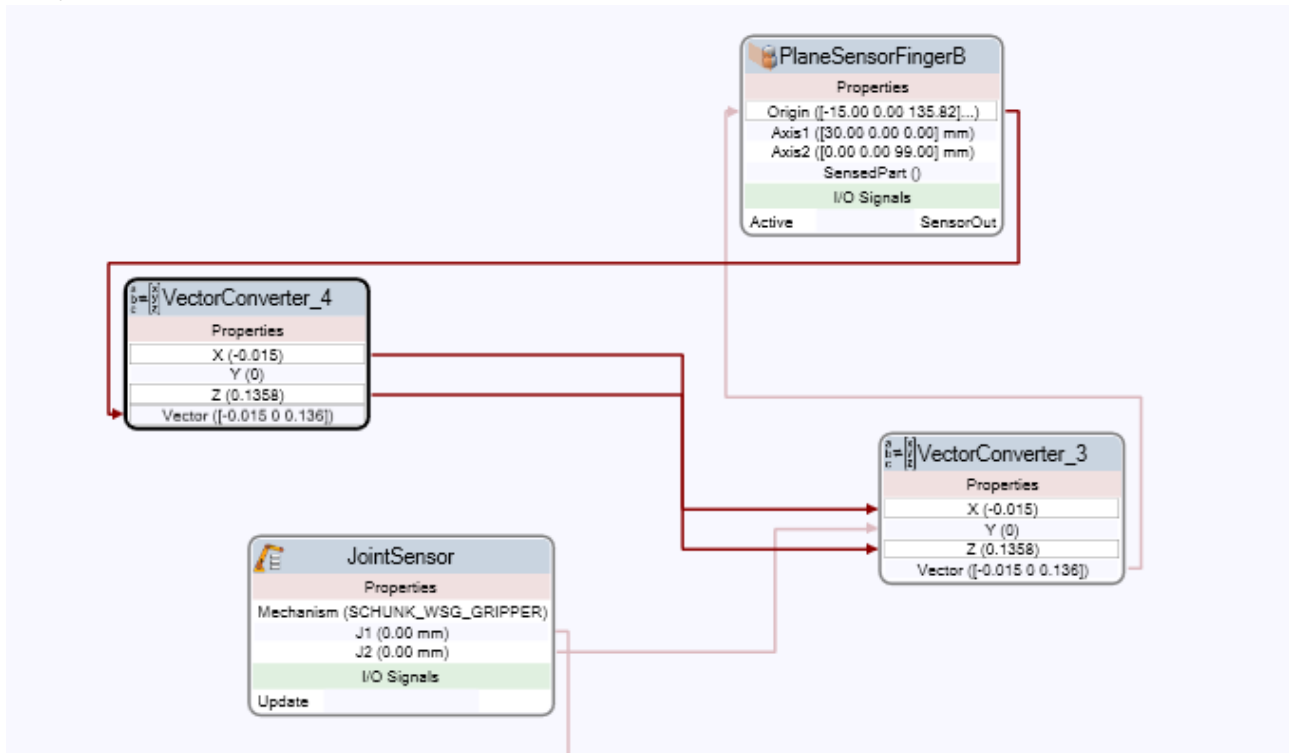
- Parametric Primitives
- Sensors
- Actions
- Manipulators

- LogicGate**
Performs a logic operation on digital signals
- LogicExpression**
Evaluates a logic expression
- LogicMux**
Selects one of two input signals
- LogicSplit**
Sets and pulses output signals depending on the state of the input si...
- LogicSRLatch**
Set-Reset latch
- Converter**
Converts between property values and signal values
- VectorConverter**
Converts between Vector3 and X/Y/Z values
- Expression**
Evaluates a mathematical expression

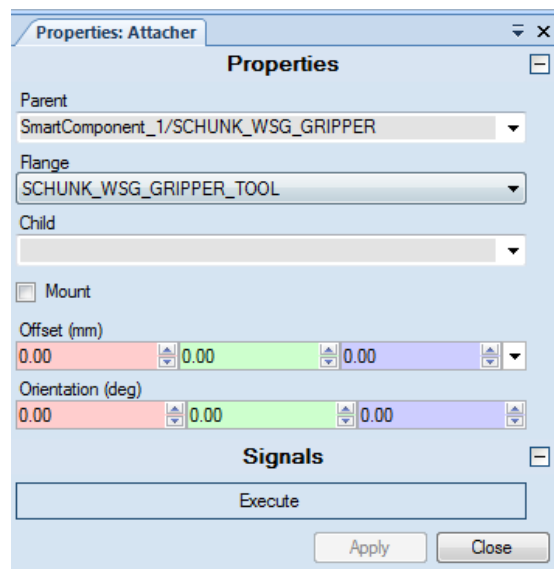
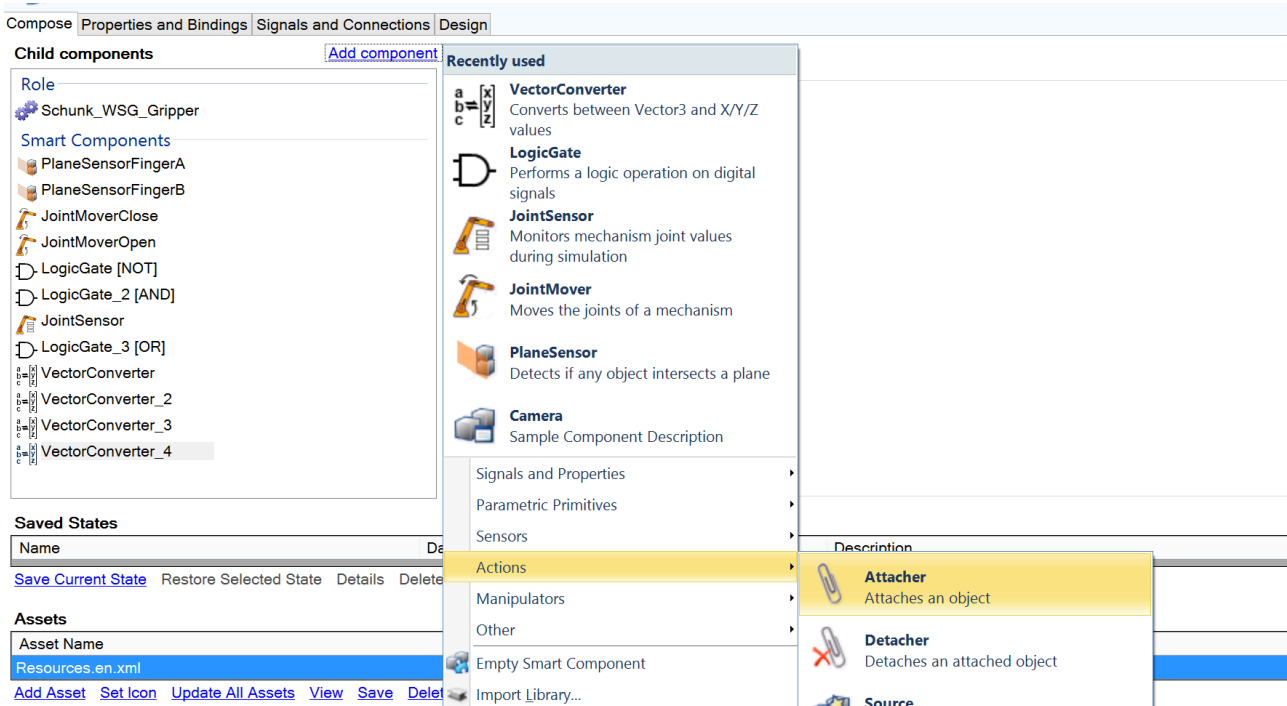
Next, connect them like this (IMPORTANT: the connection between “VectorConverter” and “PlaneSensorFingerA” should be done as the last step).



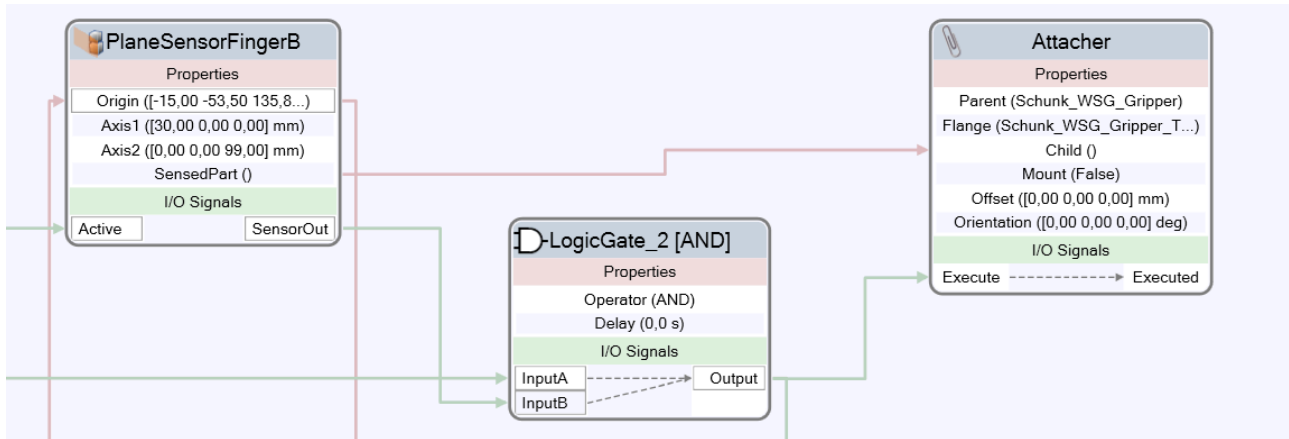
Now do the exact same thing with the “PlaneSensorFingerB”, create TWO MORE “VectorConverter” components, and connect them like this.



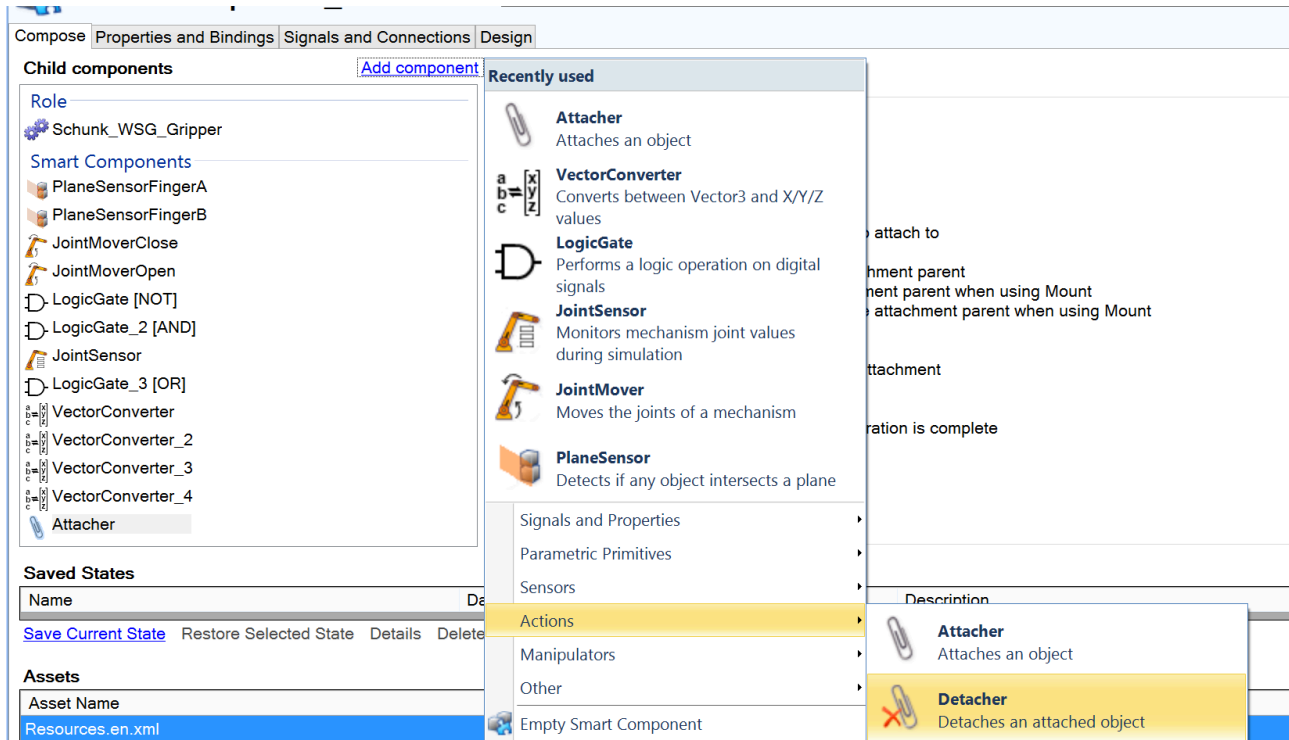
So far we have created a lot of logic, all with the purpose of facilitating the step... making an object graspable by the tool... and here it comes. Create an “Attacher” component like this.



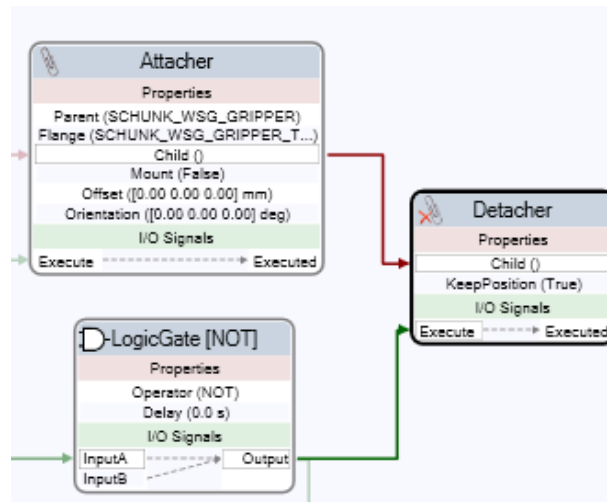
And setup the logic like this



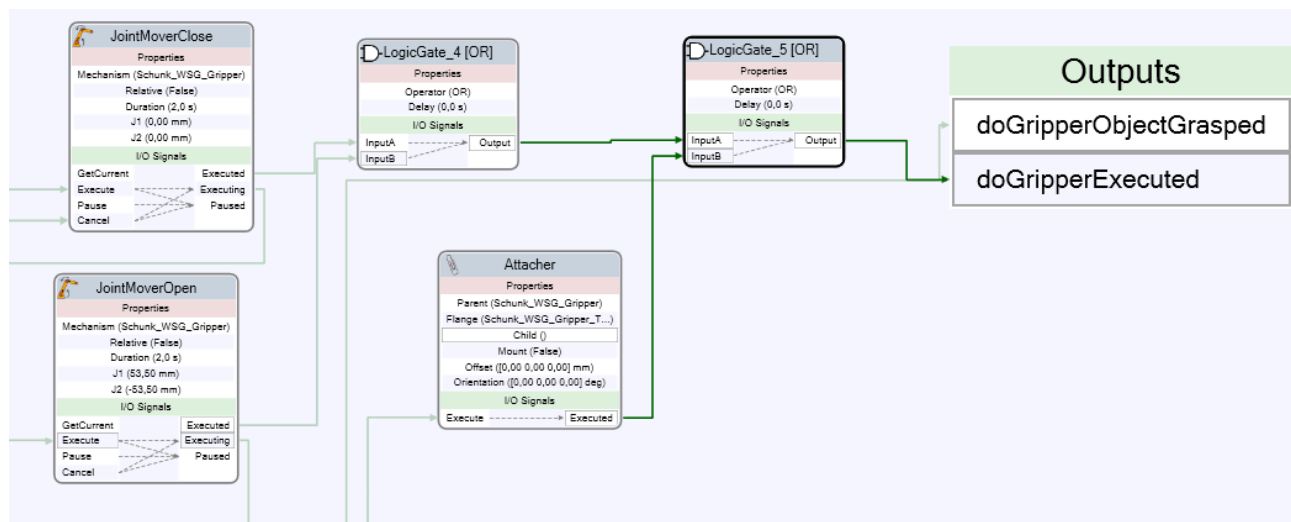
Next, create a “Detacher” component



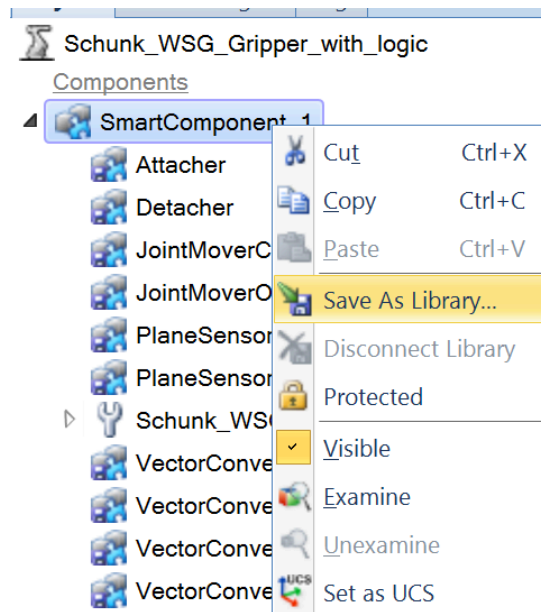
And set it up like this



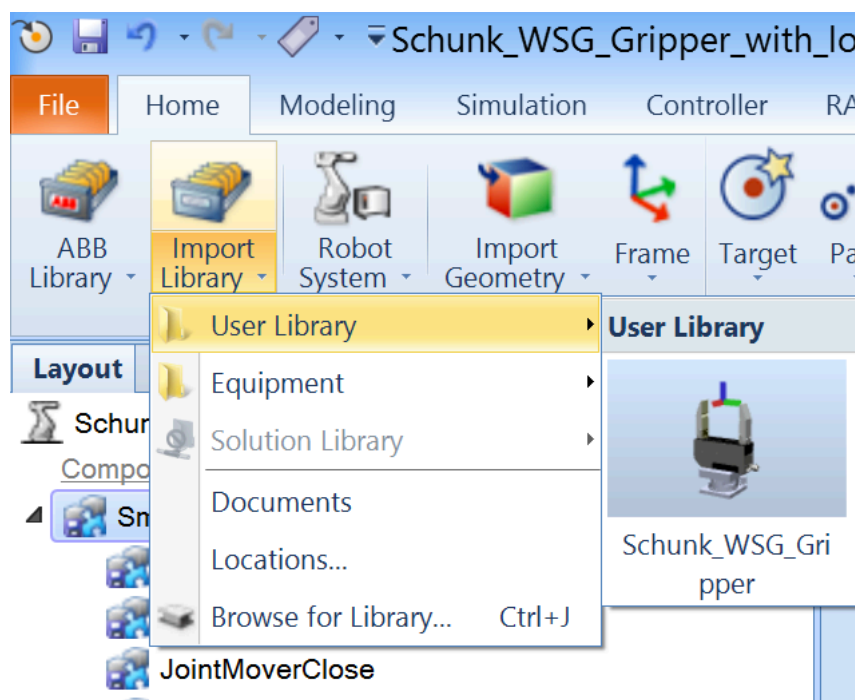
The last step to do is to make the logic for indicating that a grasp action has finished. Create two new “OR” logic gates and set it up like this.



We are now finally done with our smart component, remember to save it in its own station file. In general, if you want to make your created tools available as a library component (so it can be used in other solutions) just right click on the component save it as library.



If you have saved it in the path where RobotStudio has specified the location of a user library (normally something like "C:\Users\{name of user}\Documents\RobotStudio\Libraries"), then it should appear like this.

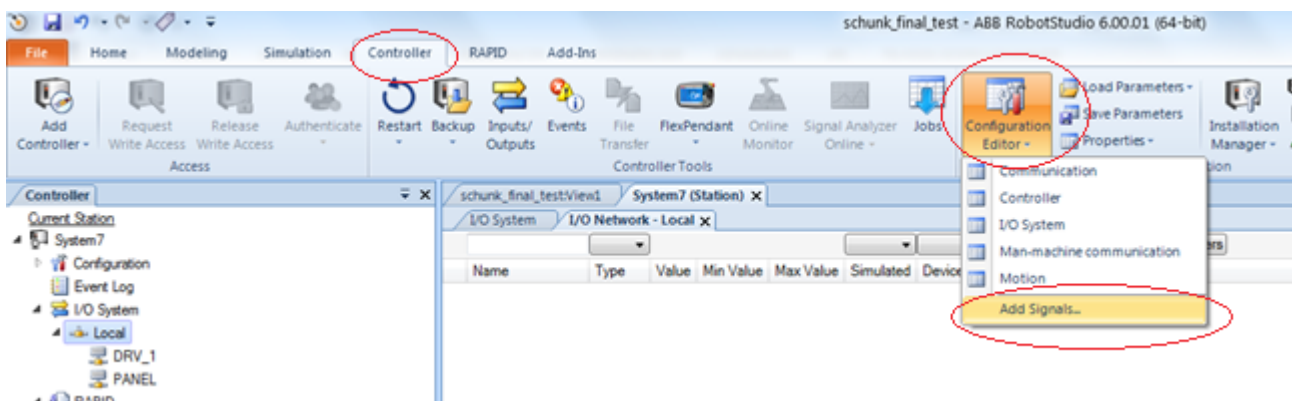


Creating Virtual Controller I/O's to control gripper

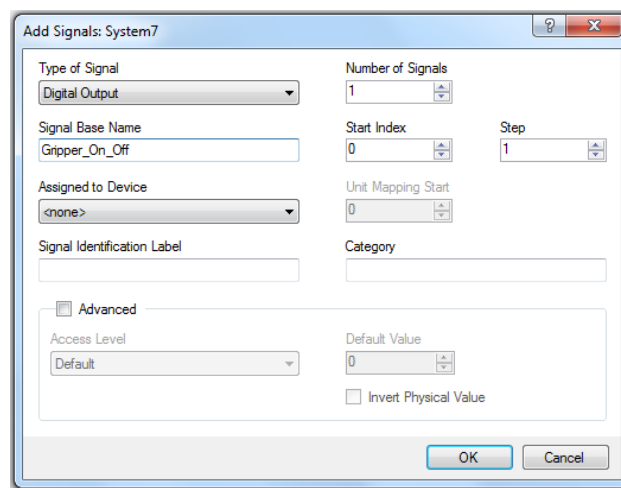
We are almost done with the creation of our tool in RobotStudio. The last thing we need to do is to learn how to setup I/O's that will enable us to activate/deactivate the gripper directly from a robot program.

So create a new empty station, load a robot, load the gripper (smart component), create a new robot system (adding a controller).

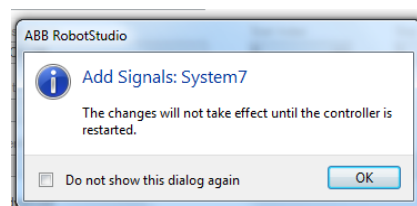
The first thing we need to do after we have added a controller is to create a new I/O in the virtual controller, and is done by going to "Controller", click on the dropdown of "Configuration Editor" and click "Add Signals".



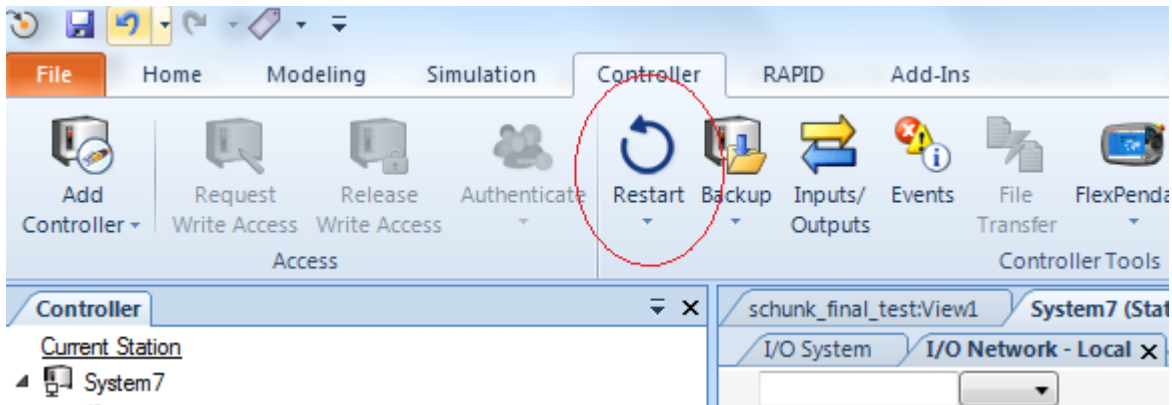
Make a digital output to control the gripper.



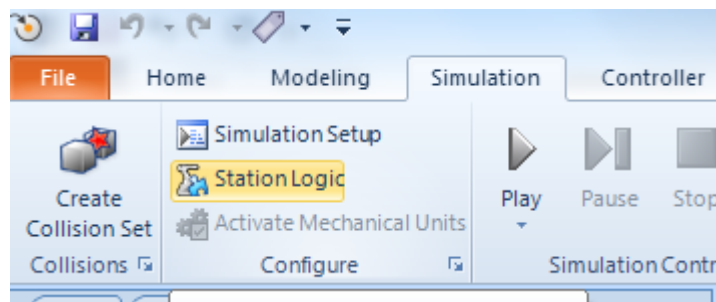
A message will pop up, saying that you need to restart the controller.



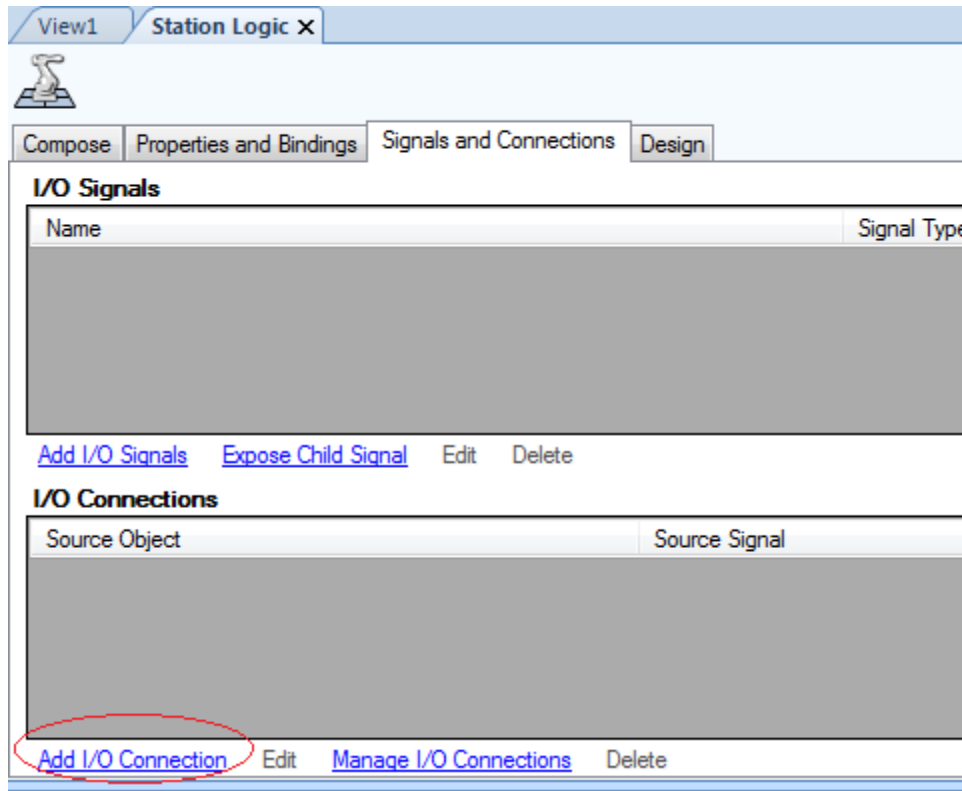
Click “Restart”



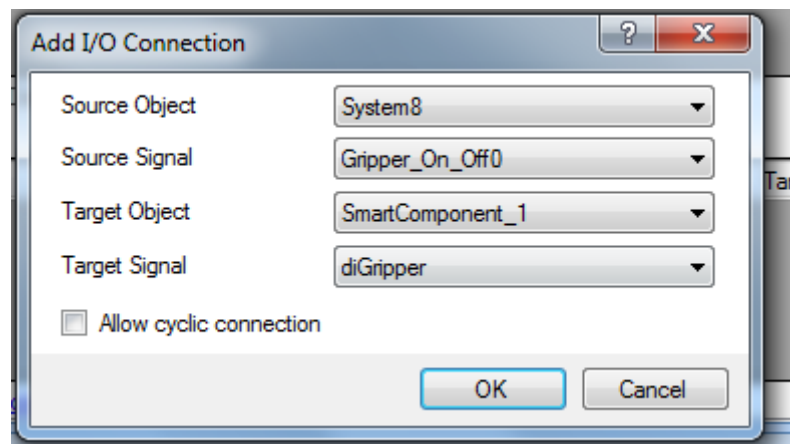
Next go to “Simulation” and click “Station Logic”. Here we can configure all logic that we want to include in our simulation model.



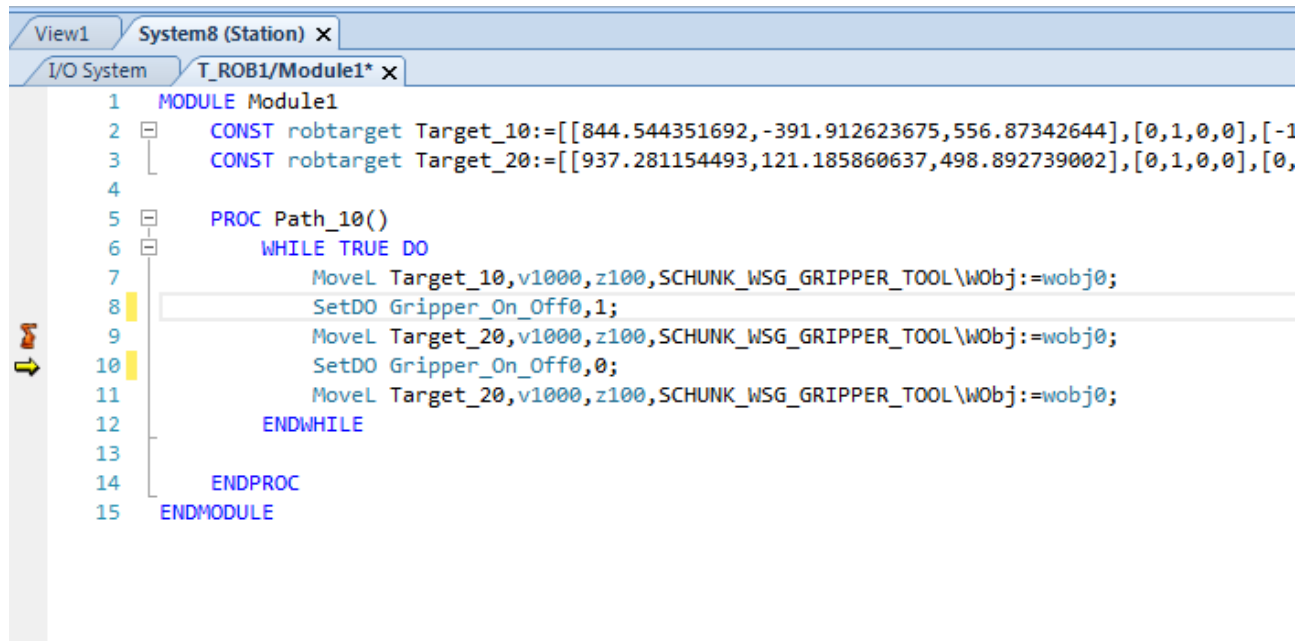
Go to “Signals and Connections” and click “Add I/O Connection”



Setup a connection between the digital output that we made in the virtual controller and to digital input of the gripper.



You will now be able to control the gripper from your robot program, through the command “SetDO”.



```
1  MODULE Module1
2  CONST robtarget Target_10:=[[844.544351692,-391.912623675,556.87342644],[0,1,0,0],[-1
3  CONST robtarget Target_20:=[[937.281154493,121.185860637,498.892739002],[0,1,0,0],[0,
4
5  PROC Path_10()
6  WHILE TRUE DO
7      MoveL Target_10,v1000,z100,SCHUNK_WSG_GRIPPER_TOOL\Wobj:=wobj0;
8      SetDO Gripper_On_Off0,1;
9      MoveL Target_20,v1000,z100,SCHUNK_WSG_GRIPPER_TOOL\Wobj:=wobj0;
10     SetDO Gripper_On_Off0,0;
11     MoveL Target_20,v1000,z100,SCHUNK_WSG_GRIPPER_TOOL\Wobj:=wobj0;
12 ENDWHILE
13
14 ENDPROC
15 ENDMODULE
```

fin ...