How to configure, program and simulate with MachineLogic

Contents

Introduction

This guide covers the setup and use of Vention's MachineLogic. MachineLogic is a code-free programming, simulation and deployment tool dedicated to the creation of applications for MachineMotion (using Vention's plug-and-play automation components).
MachineLogic is comprised of three steps:
STEP 1: Configuration (configuration of the automated equipment) STEP 2: Visual Sequence (code-free program) STEP 3: UI Builder (operator interface)
Compatible with MachineMotion v2 only
Note that applications created inside MachineLogic can be directly deployed on the MachineMotion controller (to deploy the MachineLogic program from the CAD, follow this tech doc: Push program to controller). After reading this guide, you will be ready to program and simulate your application inside MachineBuilder (Vention's CAD platform).

Setting up automated equipment in CAD

In order to simulate using MachineLogic, your design requires a few automated components in the CAD. Follow the steps below to learn how to quickly design a simple automated actuator in the CAD.

1. Create a new design and open up MachineBuilder
   Create New Design
   Either start with a blank design or start from a pre-existing design template to use MachineLogic.
2. In order to simulate, ensure you have added at least the following components into the MachineBuilder (refer to figure 1 or 2):
   - An actuator (MO-LM-XXX-XXXX) or pneumatic actuator (MO-AR-00X-XXXX)
   - A gantry plate (MO-LM-001-XXXX) (ensure the gantry plate is properly connected to your actuator)
   - Homing sensor (CE-SN-004-0001)
   - End sensor (CE-SN-004-0001)
   - MachineMotion controller (CE-CL-005-0003 or CE-CL-010-0004):
3. For an example, drag and drop a timing belt actuator from the parts browser into the MachineBuilder. Actuators could be found under the "Linear Motion" part category.

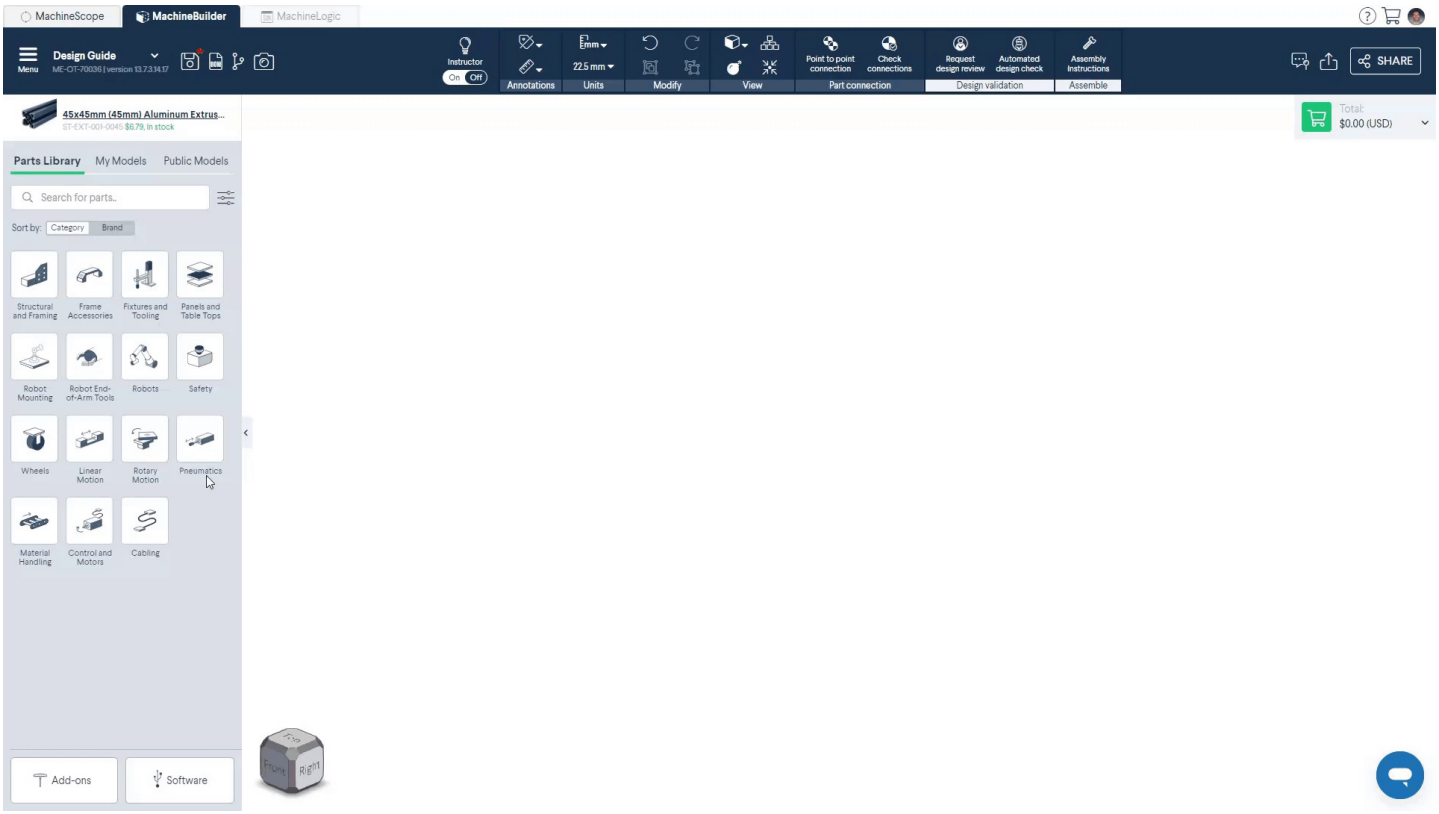

Figure 1: Configuration assistant

Step 1: Configure

Linear/Rotary actuators

Click the MachineLogic tab to begin configuring the automated equipment in your design.

1. Click the “Add Actuator” to begin your machine configuration
2. In the “Type” dropdown menu, select the actuator you would like to configure from your design.

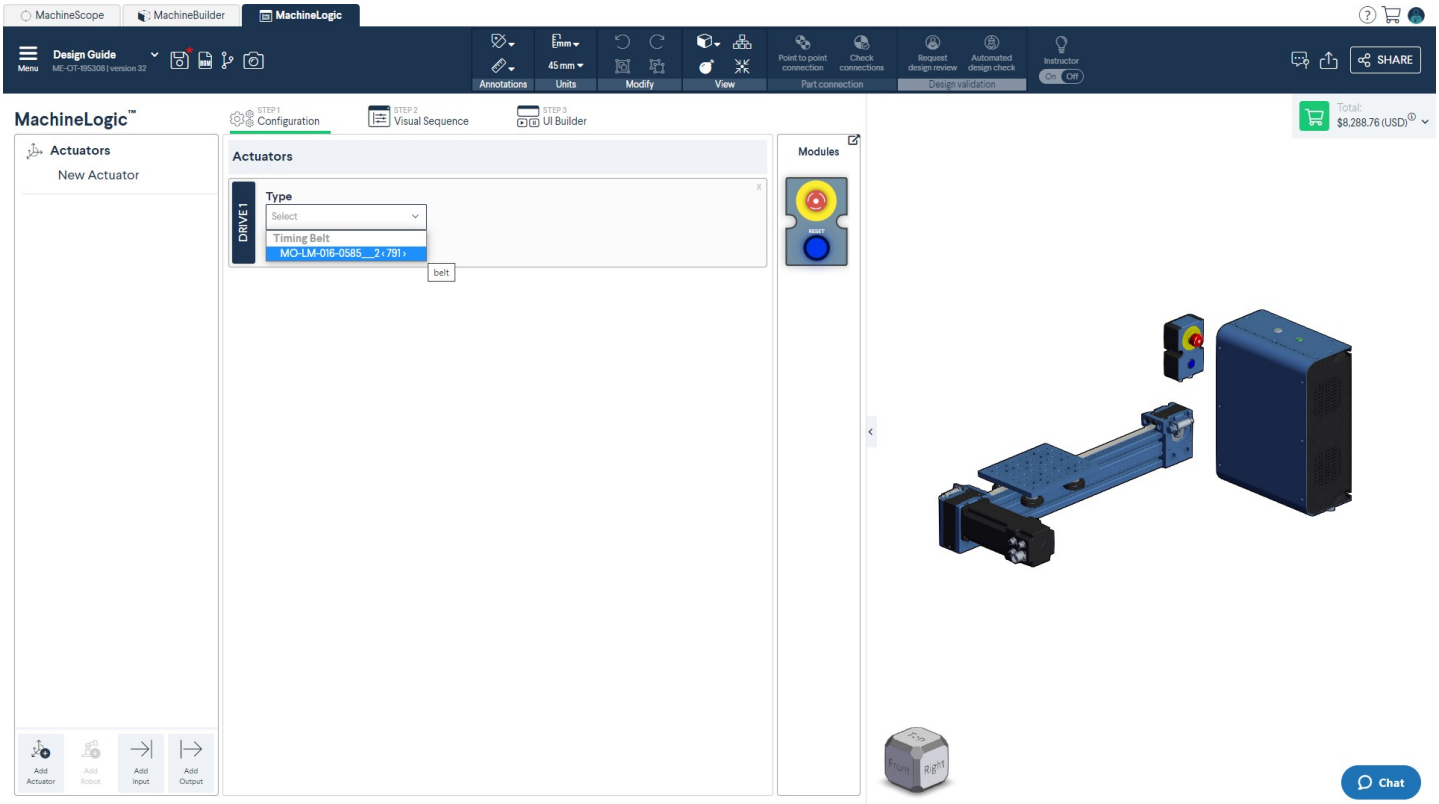

Figure 2: Automatic configuration in MachineLogic

1. The following fields will be auto-populated based on the parts connected to the selected actuator from the “Type” drop down menu:

- **Actuator name**: Friendly name given to the actuator.

- **Motor size**: Motor size connected to the actuator.
- **Homing sensor**: We will guess which sensor should be the gantry's home position. If you would like to reverse the motor direction, switch the homing sensor with the end-stop sensor.
- **End-stop sensor**: We will guess which sensor should be the gantry's end-stop sensor. If you would like to reverse the motor direction, switch the homing sensor with the end-stop sensor.
- **Brake installed**: This checkbox represents the presence of a brake on the associated actuator.
- **Gearbox installed**: This checkbox represents the presence of a gearbox on the associated actuator.
- **Advanced**: Allows you to configure the following fields:
  - **Custom Current**: A different current value for your motor. The default value will be shown.
  - **Tuning profile**: This allows you to tune your step-servo motor using various profiles, to achieve the best performance for your application.

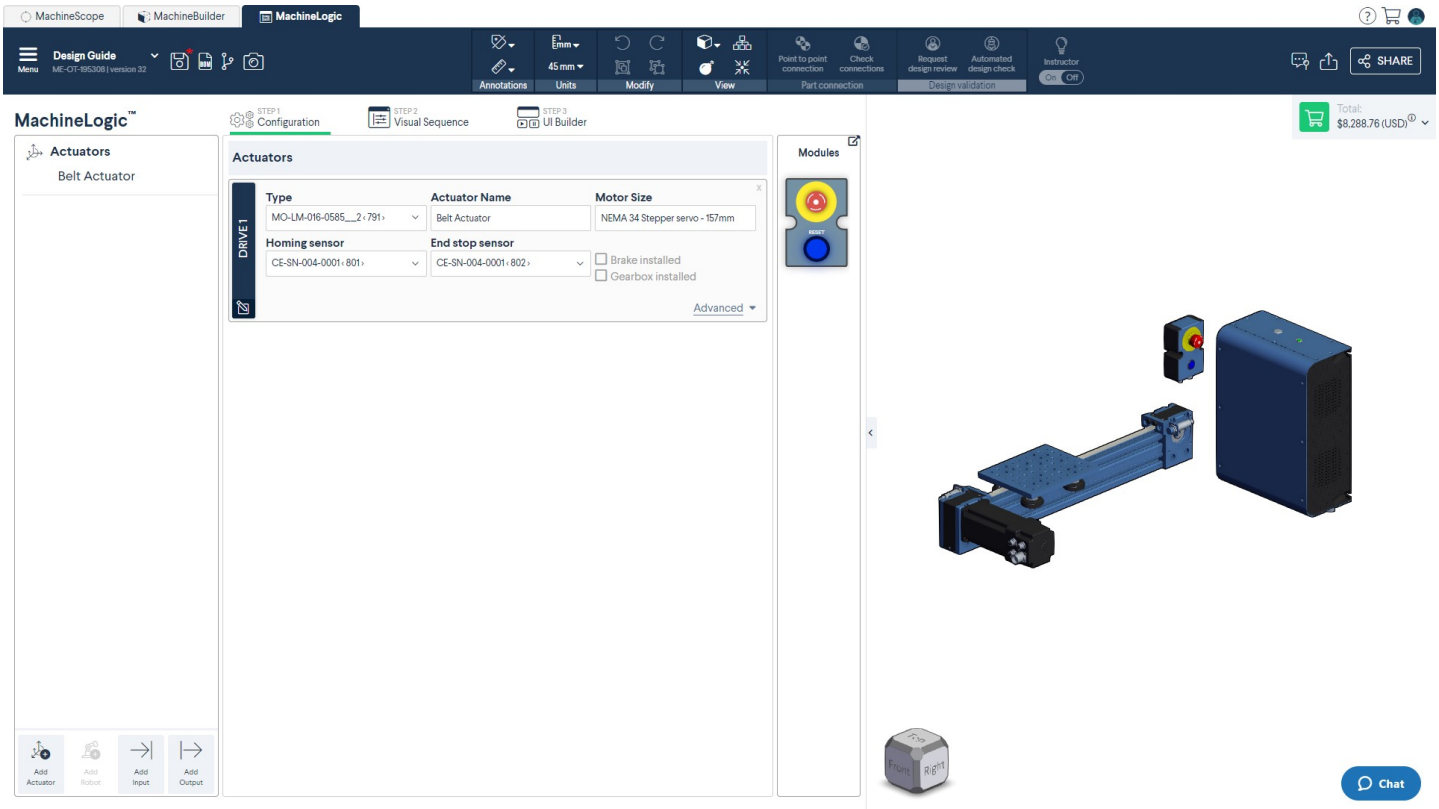

Figure 3: Actuator configuration in MachineLogic

Pneumatic actuators

Click the MachineLogic tab to begin configuring a pneumatic actuator.

1. Click the "Add Actuator" to begin your machine configuration
2. In the "Type" dropdown menu, select the pneumatic actuator you would like to configure from your design.

- **Actuator name**: Friendly name given to the pneumatic actuator.
- **Valve A Push**: Represents the output pin from the digital IO module (CE-MD-001-0001) to activate the extended piston position of the pneumatic actuator.
- **Position Sensor (push)**: *This field is optional*. Represents the pneumatic actuator position sensor (CE-SN-008-0001) to give the feedback if the pneumatic actuator is in the "push" position. Select the input pin that will be connected to this sensor.
- **Valve B Pull**: Represents the output pin from the digital IO module (CE-MD-001-0001) to activate the retracted piston position of the pneumatic actuator.
- **Position Sensor (push)**: *This field is optional*. Represents the pneumatic actuator position sensor (CE-SN-008-0001) to give the feedback if the pneumatic actuator is in the "pull" position. Select the input pin that will be connected to this sensor.

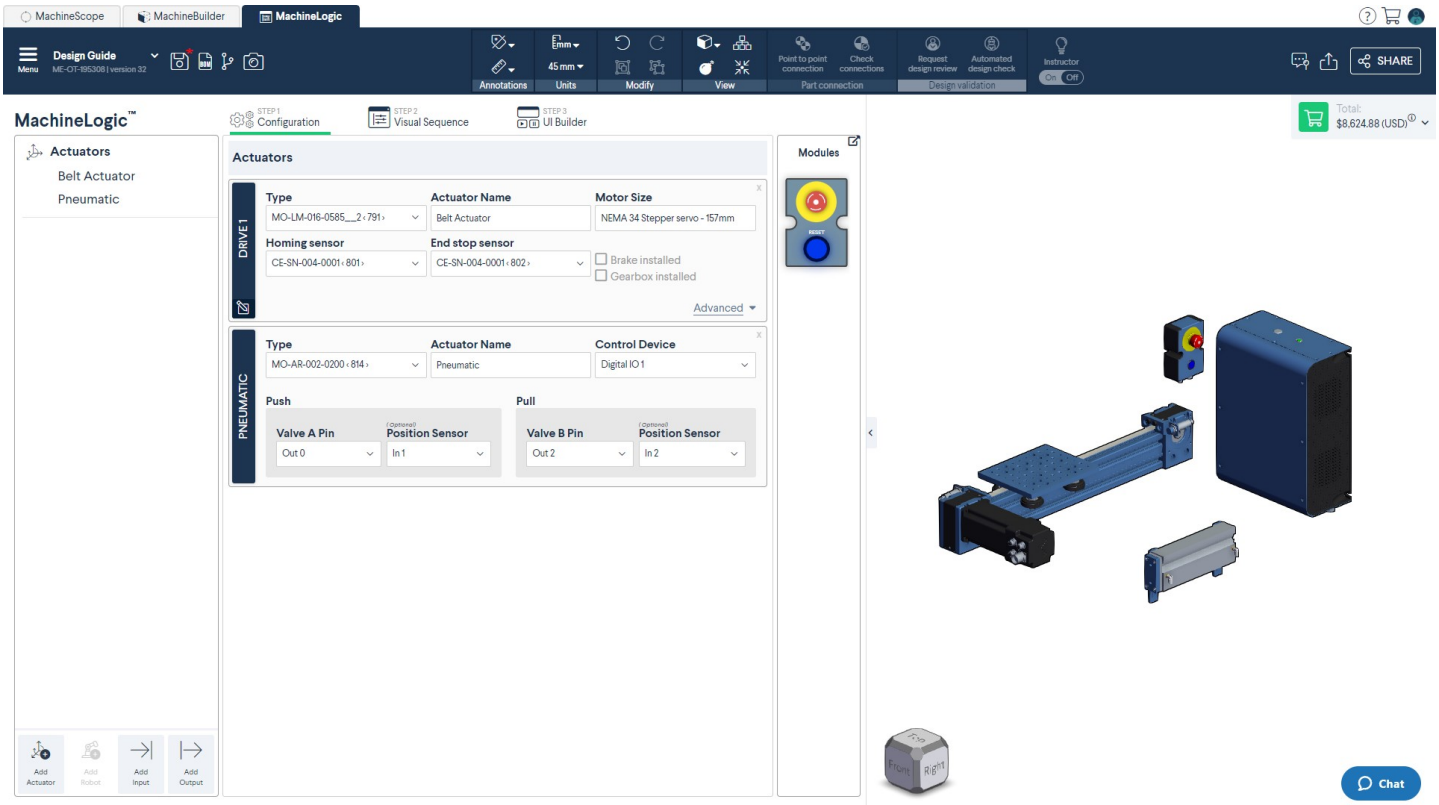

Figure 4: Configuring a pneumatic actuator in MachineLogic

Step 2: Visual Sequence

This step allows you to build programs for the actuators, inputs and outputs you have configured in the first step. Follow the steps below to build a program:

1. Click "Add Application" to start building a new program. An "**Application**" is defined as the parent of all your sequences, and global assets (such as variables). You may create multiple applications within your design.
2. Click "Add Sequence" to add child sequences (sequences can execute within one and another). Child sequences comprise of commands that should be organized as a group or repetitive motions that should be executed throughout the application.
3. Click "Add Command" to add various commands to be organized in your sequences. These definition of each of these commands could be found by clicking here. If you need help understanding the command in the program, click the "?" icon to display an explanation of the command.
4. Once all your motion sequences are organized within your child sequences, you may execute (click "Add Command" › "Add execution") the child sequences through your "**Main Sequence**" (the parent sequence of all child sequences). Only the main sequence can be played.
5. Play the main sequence to execute the simulation (this can also be done through the operator interface in step 3).

Note: All applications and sequences can be renamed.

Variables

Variables gives the ability to change a parameter easily throughout the entirety of the MachineLogic program. It also gives the ability to name the parameter for an operator or a collaborator of the program to easily navigate through the program.

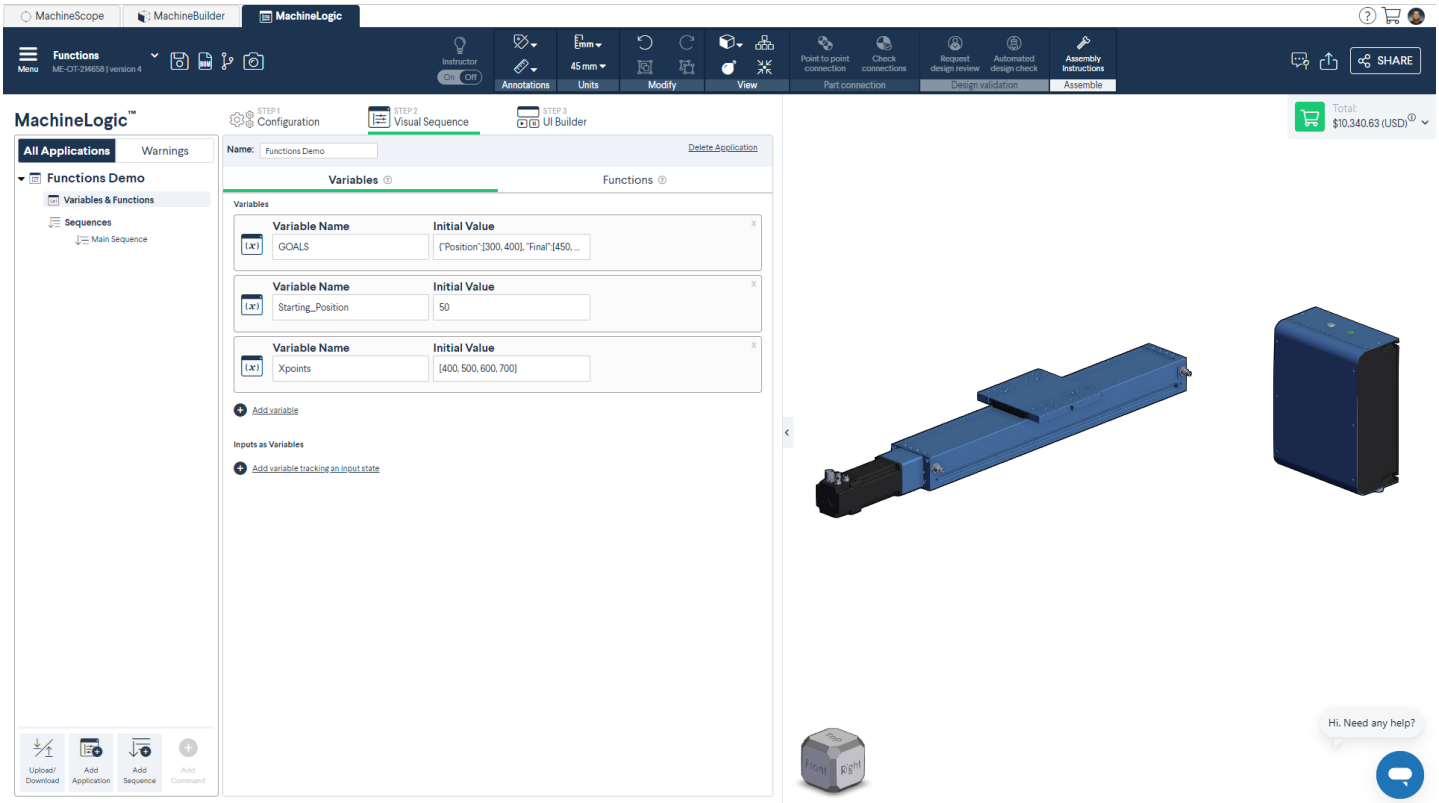

Figure 5: Variables tab

1. Click on the name of the application you would like to create a variable for. Under the application that is being programmed, there will be two tabs; "**Variables**" and "**Functions**".
2. Click on the "**Variables**" tab and click "Add variable".
3. Name the variable and add the initial value of the variable. The initial value field accepts **integers**, **arrays**, **strings**, **objects**, and **JSON** files. The initial value will define the value of the variable when you call it the first time in the MachineLogic program. That variable value may change throughout the program.
4. Once this variable is defined, the variable could be called in:

- Text boxes that have "variable" labelled on it
- "Set variable from device" command
- "Set variable from expression" command
- "Add Motion" commands
- "Add message to information console" command's message field

Type in the first letter of the variable and the text box will pop-up the possible variables that could be used.

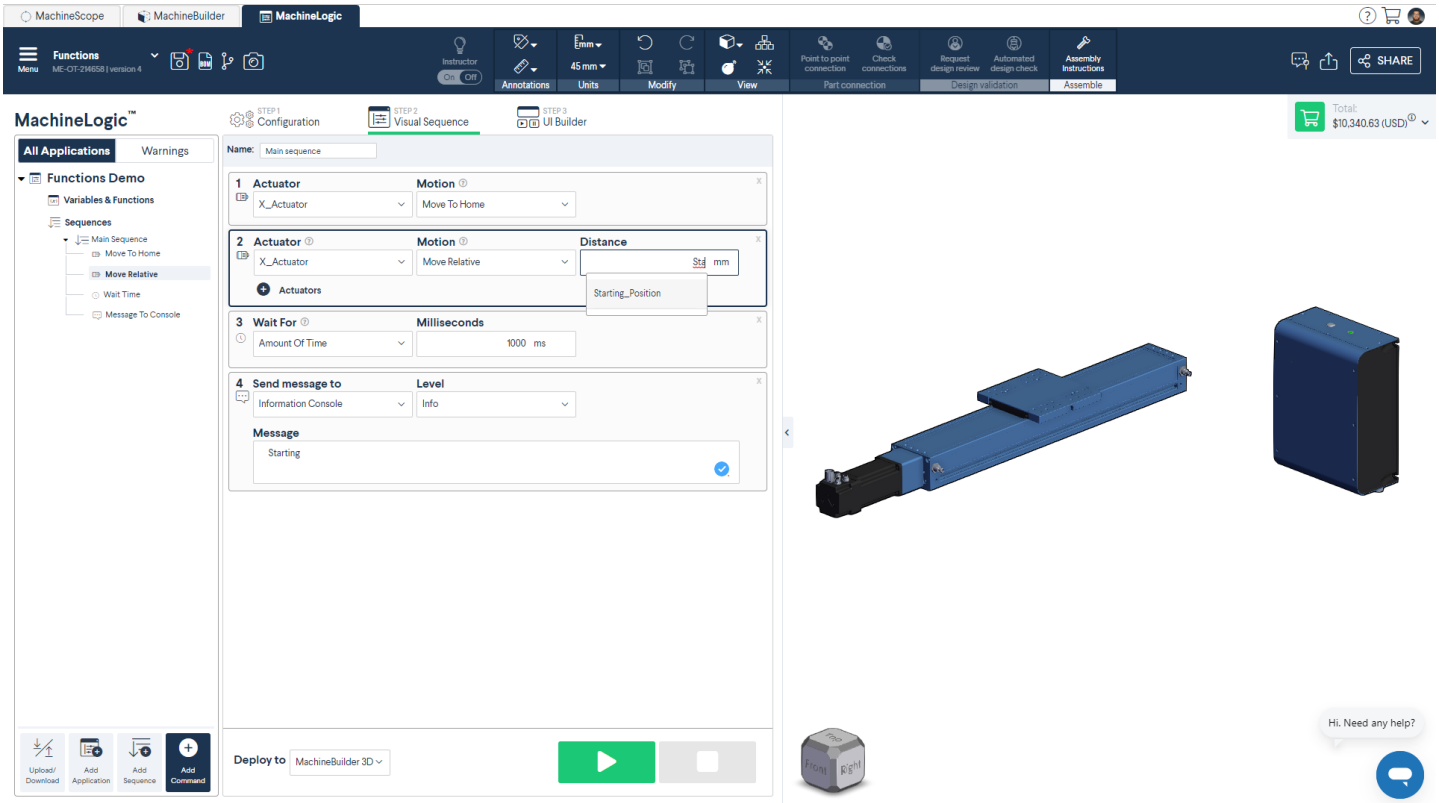

Figure 6: Variable auto-fill

Lambda Functions

Lambda functions give the ability to create complex expressions to return a string, a number, an array or an object, to be inputted in other MachineLogic commands. To use lambda functions, enter a **function name**, **arguments** to allow names to be passed in the body and the actual **Function body**. Lambda Functions accept regular **JavaScript** syntax, meaning *for* or *while loops*, *if-else conditionals*, *operators*, and much more can be written in the Function body. Additionally, **JavaScript objects** such as *Math*, *Array*, or *Map*, and their respective functions can also be called directly in-line. Learn more about what can be done by visiting the official JavaScript Documentation.

Lambda functions can be called in any of the MachineLogic sequence commands found below, in the following format: `FunctionName(Arguments)`.

- Motion commands
- Wait commands
- Output commands
- Set variable commands
- Condition commands
- Loop commands
- Message commands

Functions should be used when similar calculations are needed to be used in the commands mentioned above. For an example, if a position could be computed by adding two numbers together, you could create a function that takes in two arguments and return their sum.

Example:
First, create a function in the functions tab:

```
myFunction(a,b) {
    return a + b
}
```

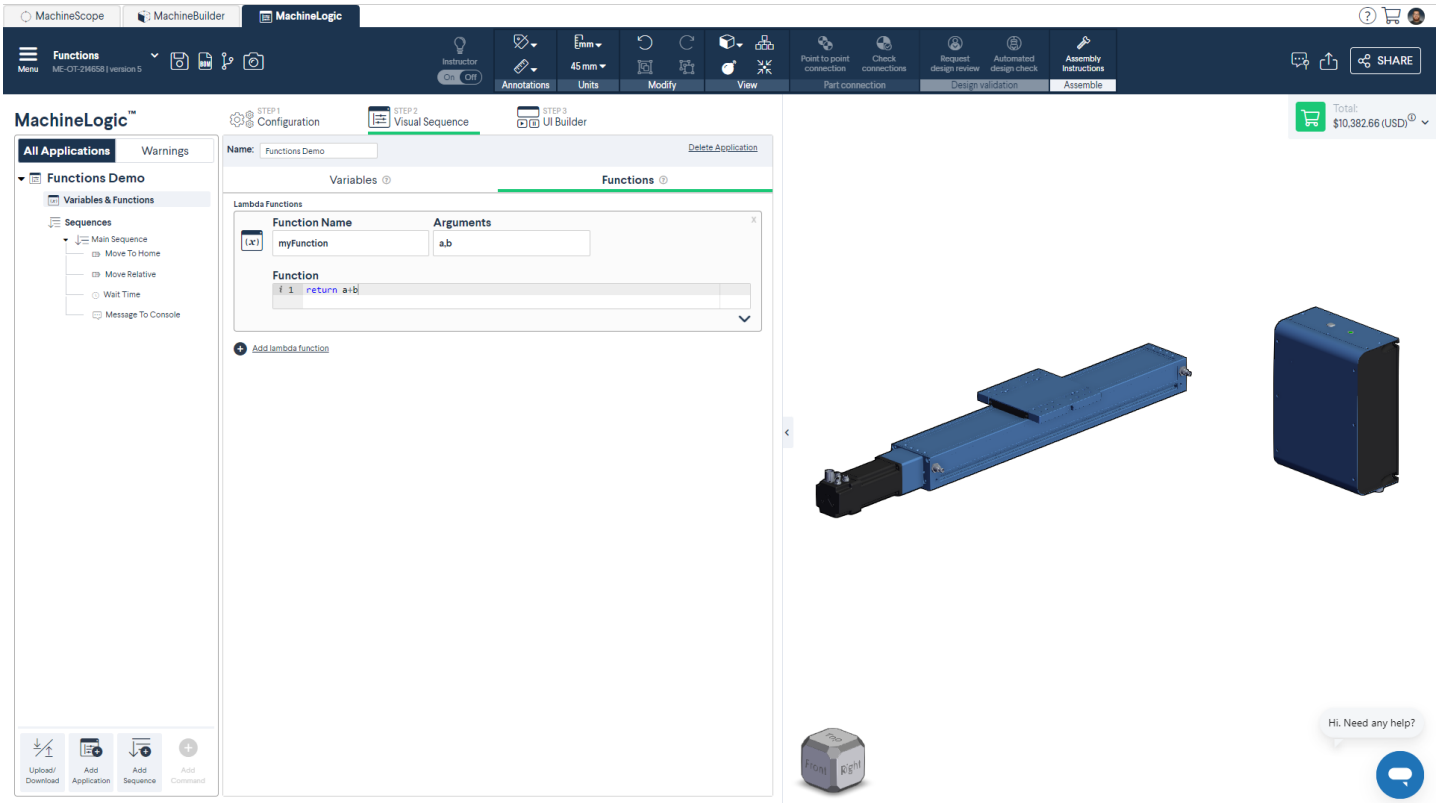

Figure 7: Creating a lambda function

Then in your "add motion" command, you may input `myFunction(200,300)` to return a value of 500.

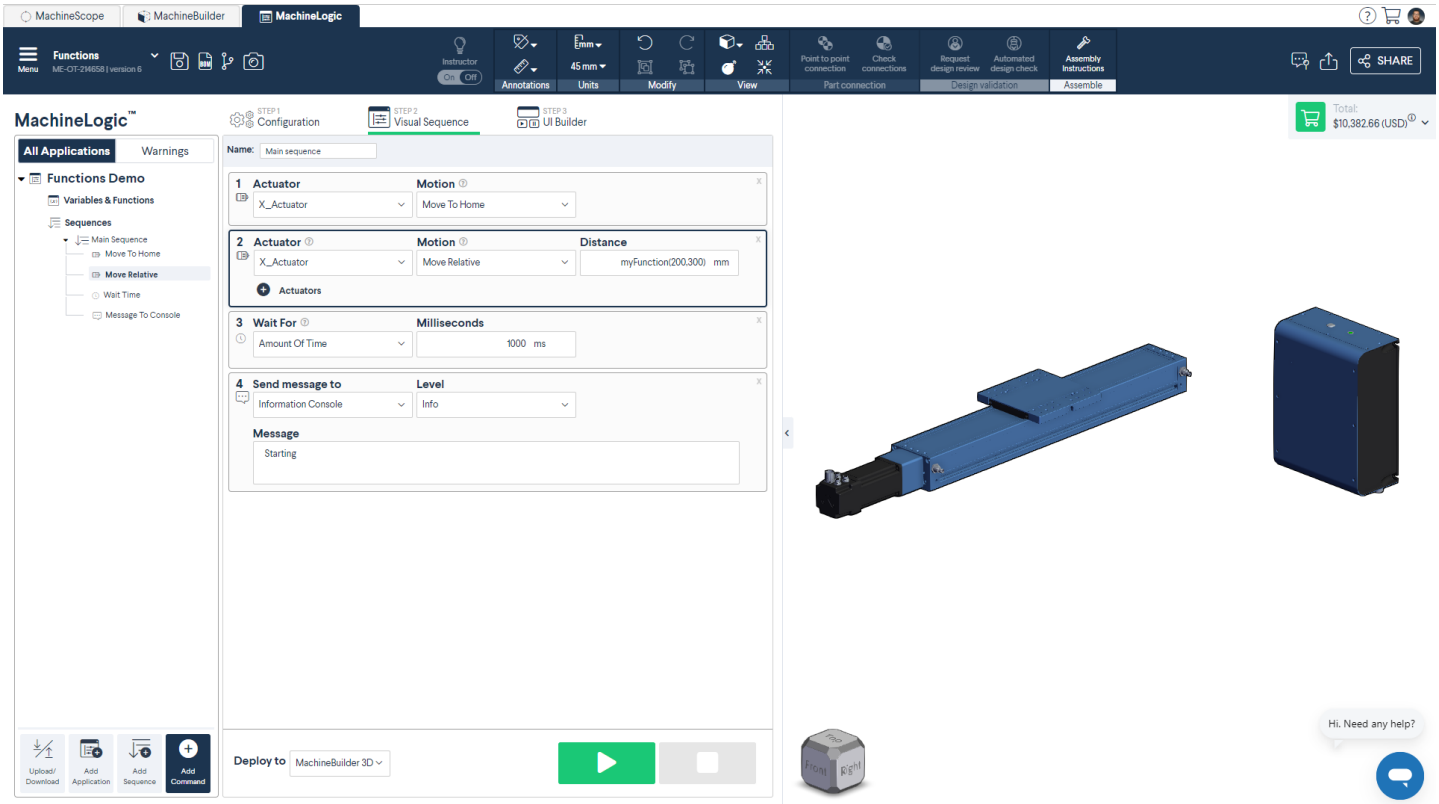

Figure 8: Calling a lambda function in a command

Loops

Loops give the ability to simplify the program by organizing repeating commands under a "Sequence" that will repeatedly execute under certain conditions. Here are the types of loops in MachineLogic:

Forever: This loop will allow a sequence to run forever.
Count: This loop will allow a sequence to run for a user-specified number of times.
While: This is a conditional loop that allows sequences to continuously execute while the comparison of the first "Value/Variable" to the second "Value/Variable" is true.

To add a loop in your sequence, click “Add Loop” at the bottom of the MachineLogic toolbar. Ensure that another child sequence exists in order for the loop to be created.

Conditions

The **If** condition allows a sequence to be executed in series or executed in parallel only if the condition specified is met.
To add a condition in your sequence, click “Add Condition” at the bottom of the MachineLogic toolbar. To create an “if” condition, ensure that another sequence is created in order to execute the group of commands if the condition is true.

Step 3: UI Builder (optional)

The UI Builder allows you to build a custom operator interface based on the application you have built in your “Step 2: Visual Sequence”.

1. Go to the “STEP 3: UI builder” tab
2. Ensure you’re on “Edit Mode”. Drag and drop the widgets from “Construct your UI” menu to the cells within the grid.

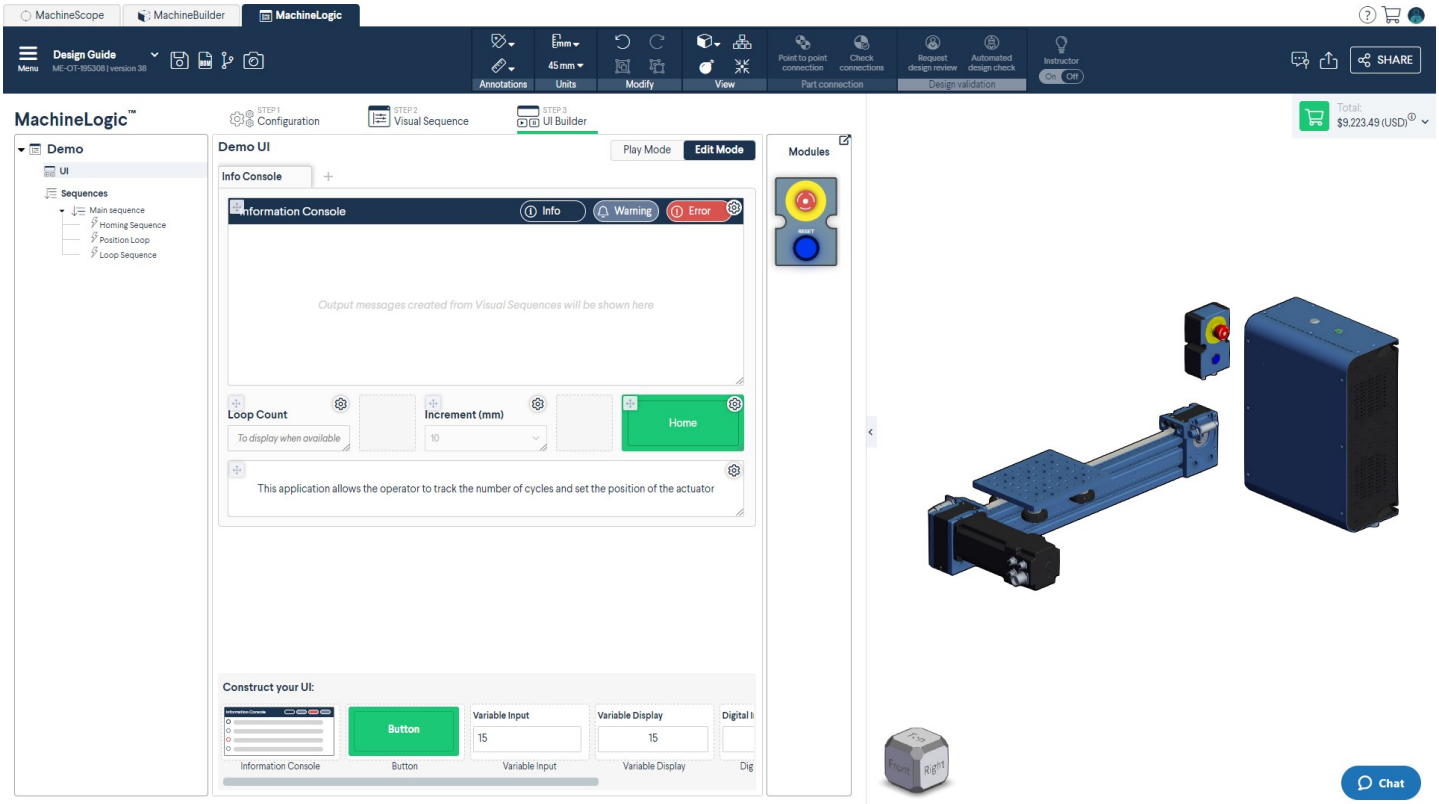

Figure 9: Edit mode

1. Here is how to configure each widget type:

- **Information console**: this is the console where the “Add message” commands added in “STEP 2: Visual sequence” will be outputted.

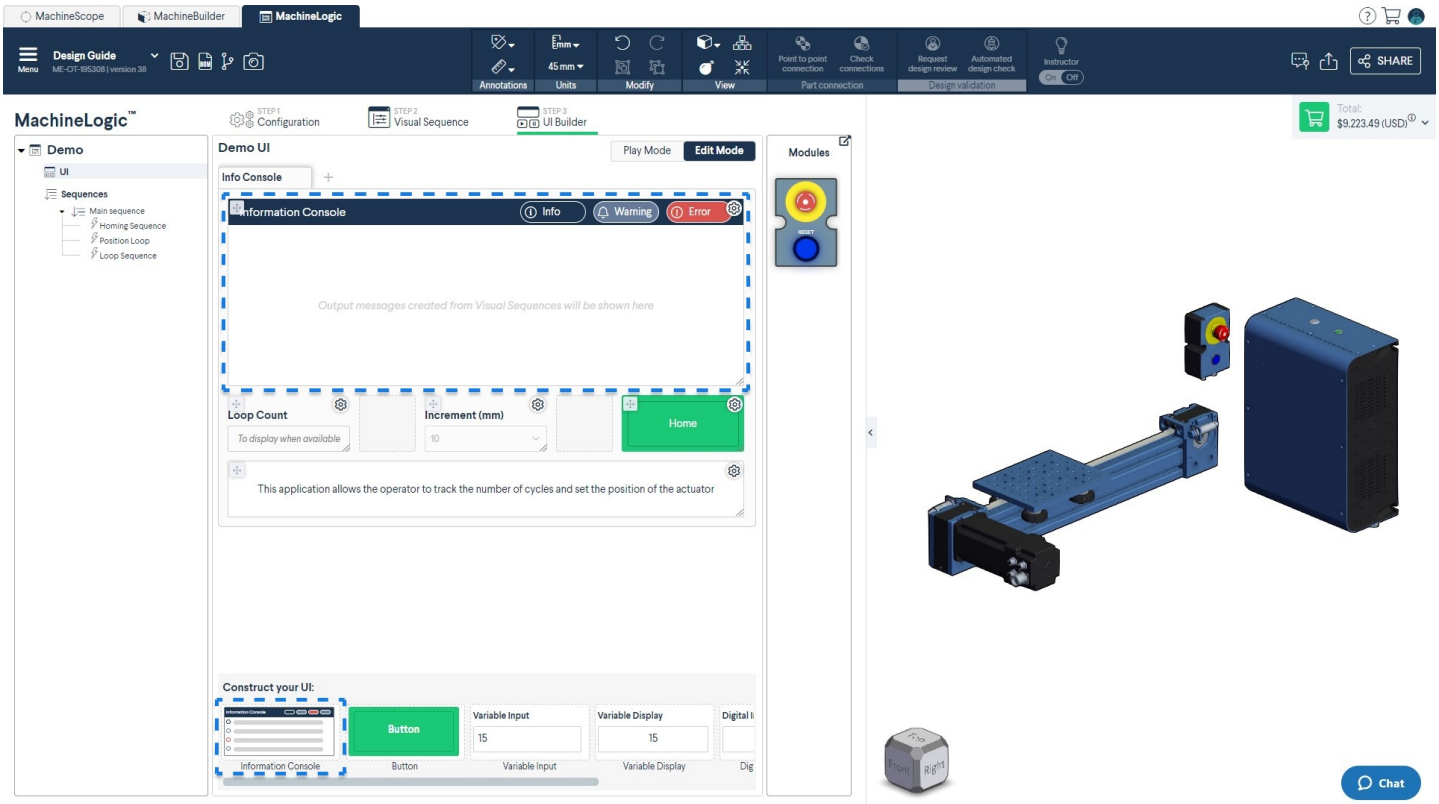

Figure 10: Information console

- **Button**: The button widget allows an operator to generate an event anytime the button is pressed. If you would like given commands to be executed only after the operator presses on the button widget, you can add a "Wait for event" command at "STEP 2: Visual Sequence" right before them. The chosen topic and messages should match between the button widget and the "Wait for event" command.

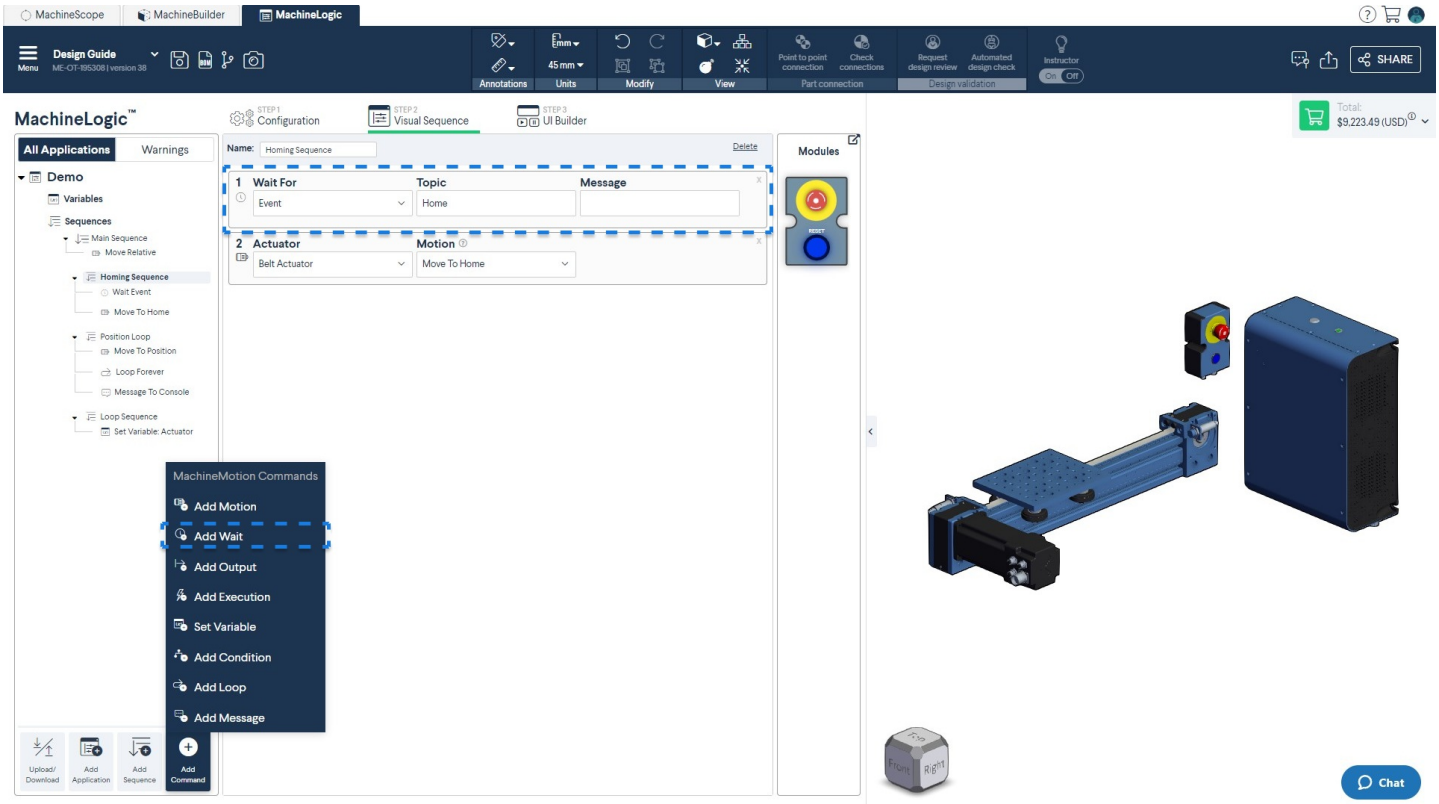

Figure 11: STEP 2: Visual Sequence - Wait for event

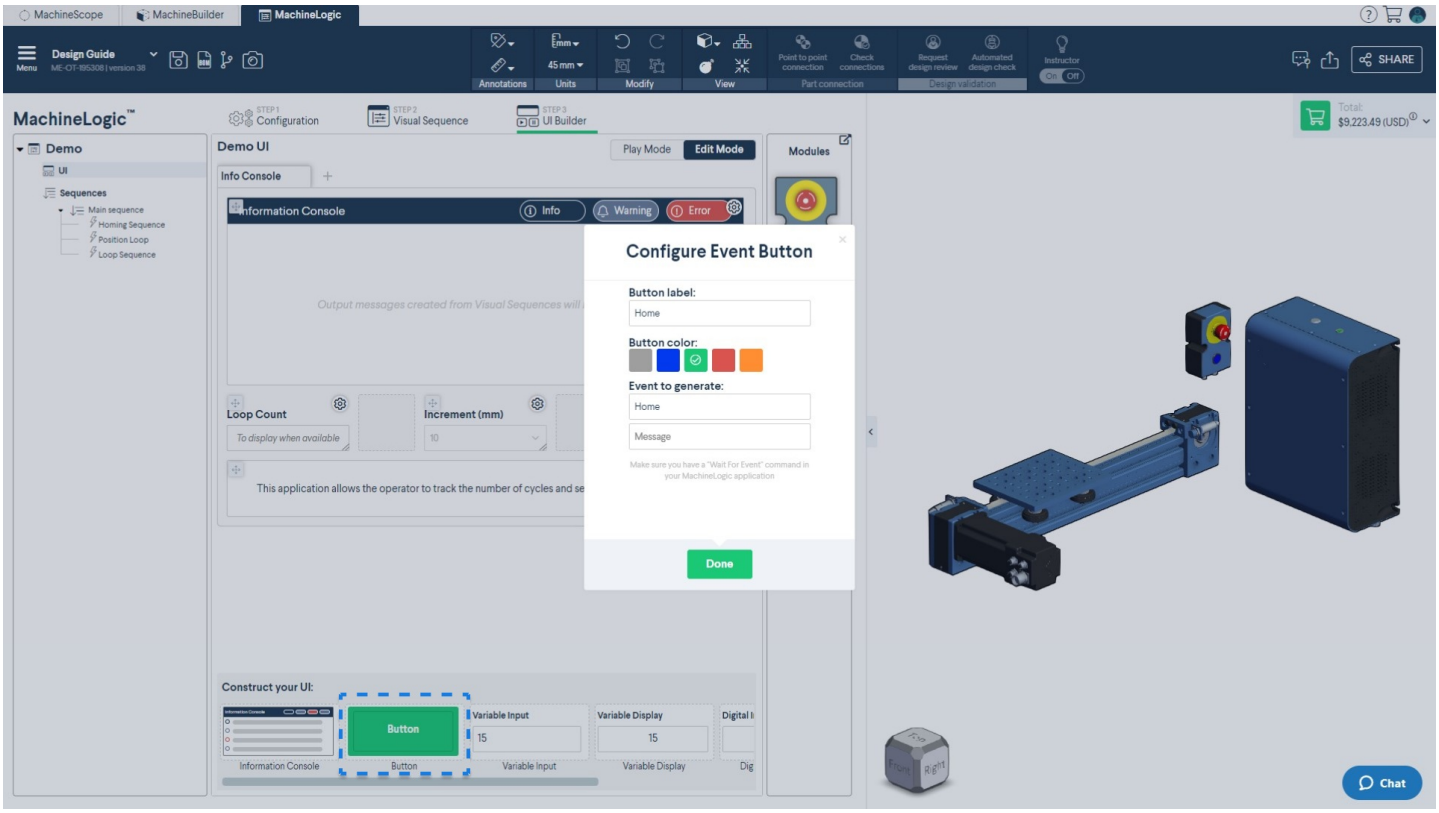

Figure 12: STEP 3: UI Builder - Generate event button

- **Variable Input**: Based on the variables configured from "STEP 2: Visual sequence", the operator may change the initial value of the variable based on a configured dropdown menu, free text or specifying min & max values. The operator may not change the variable when the application is playing.

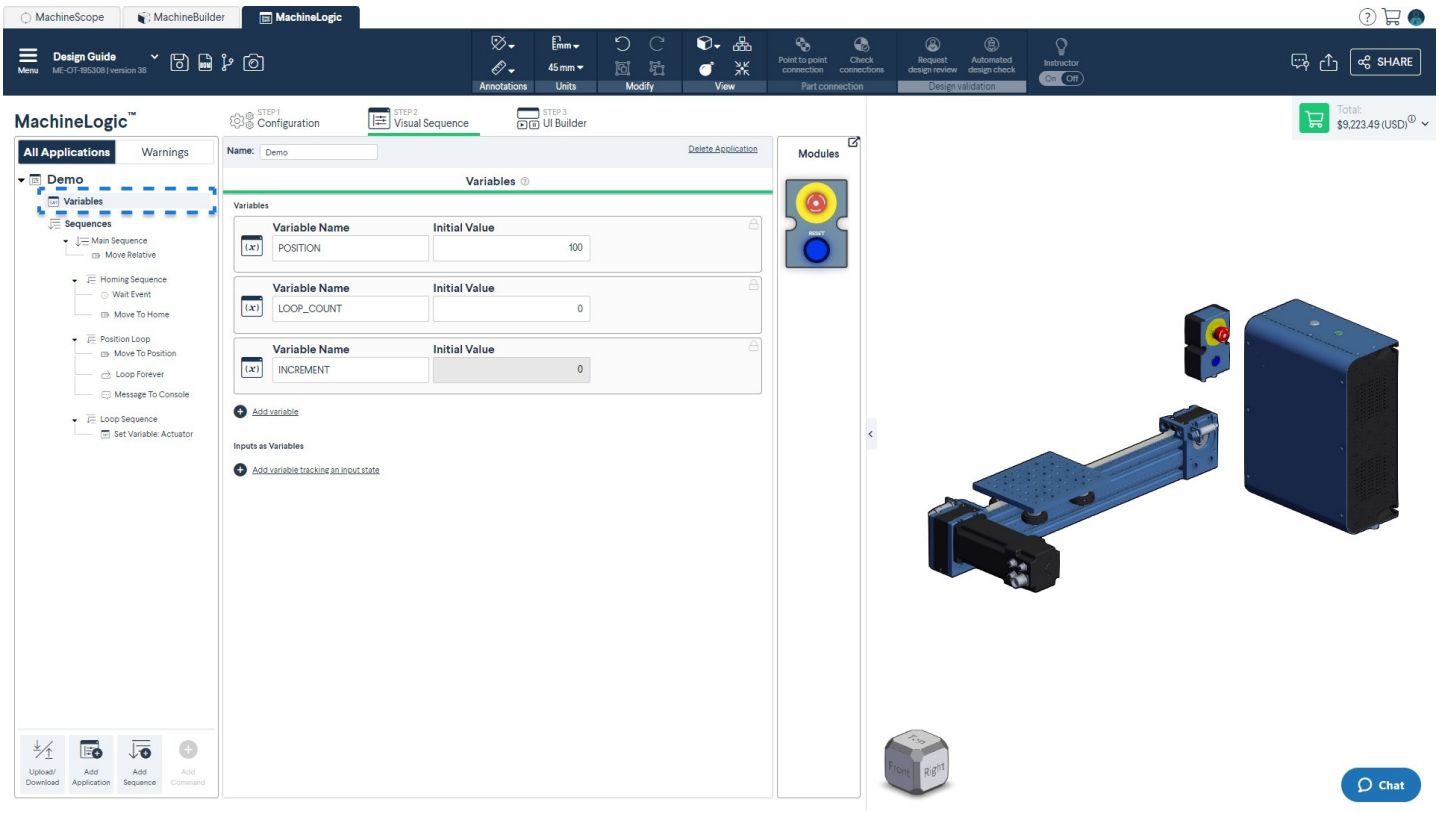

Figure 13: STEP 2: Visual Sequence - Variables

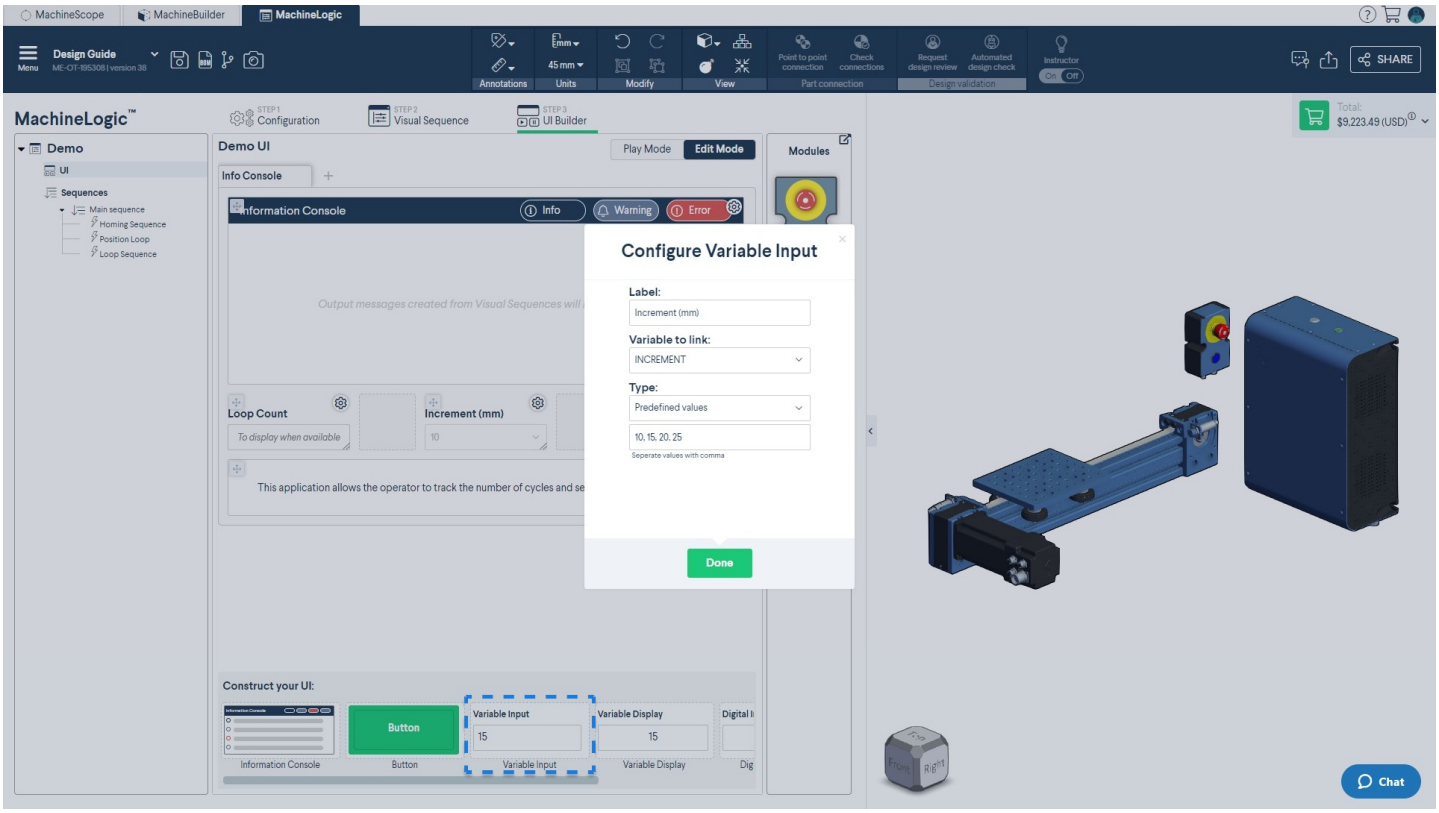

Figure 14: STEP 3: UI Builder - Variable input

- **Variable Display**: Displays the value of the variable in real time as the program runs.

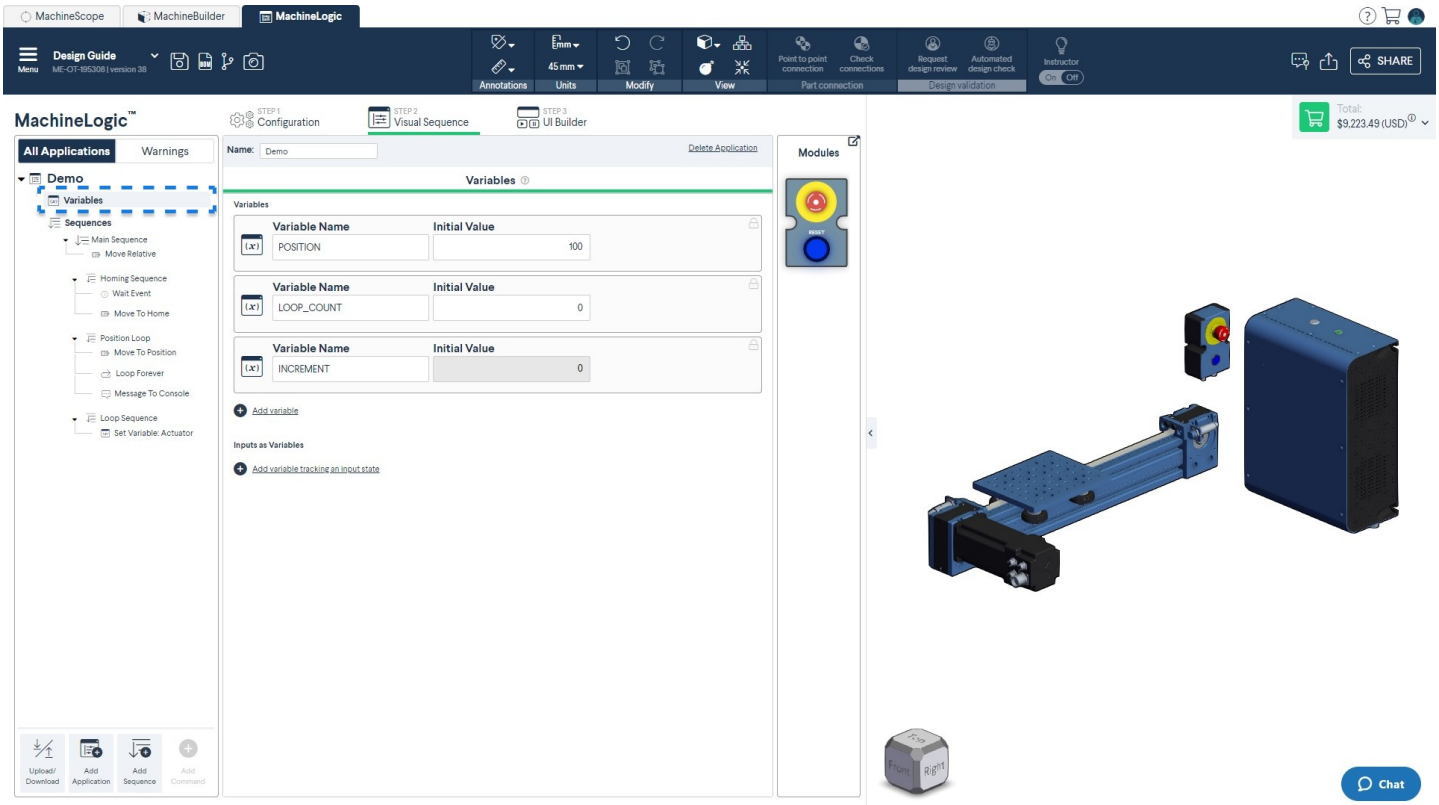

Figure 15: STEP 2: Visual Sequence - Variables

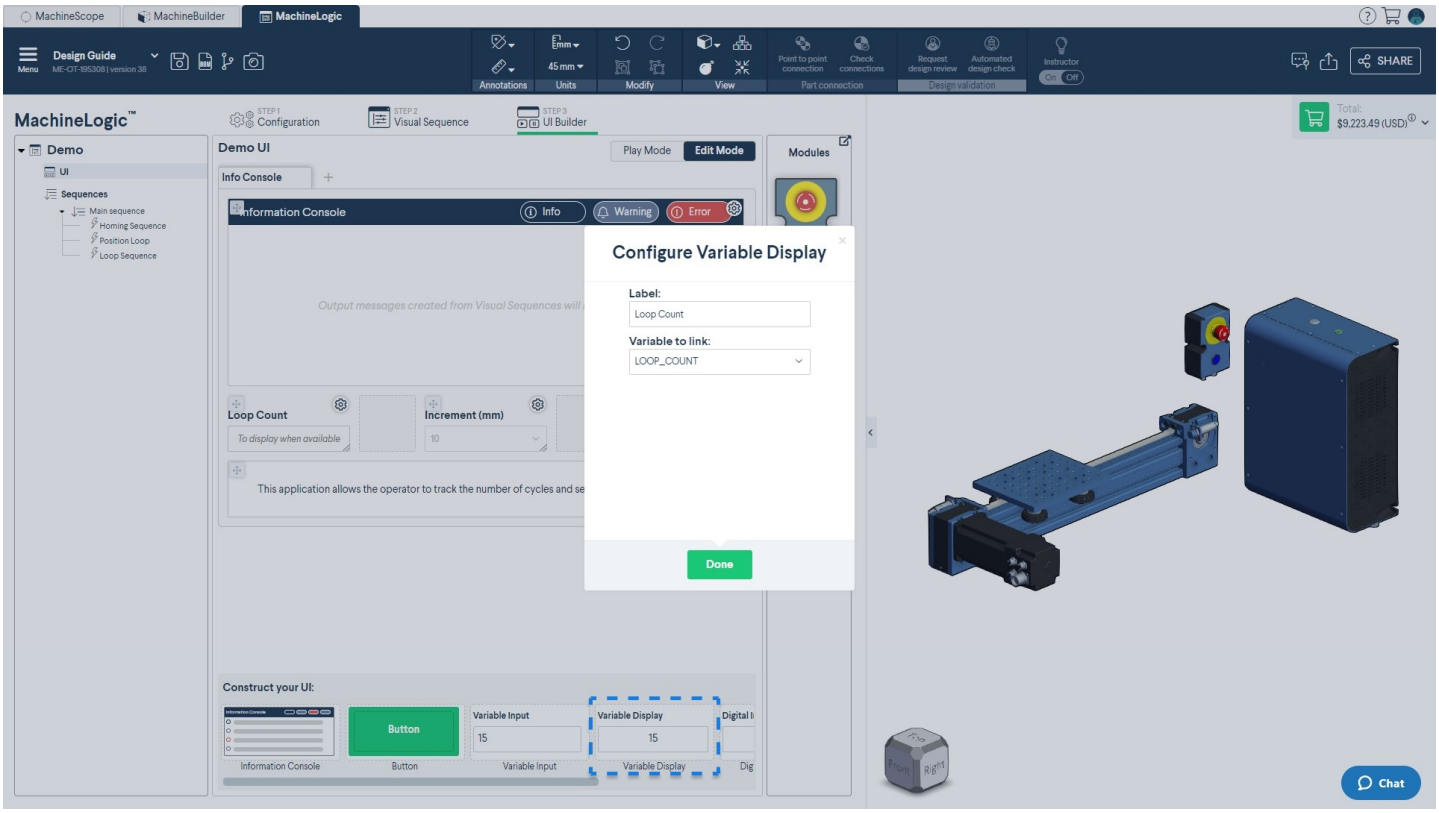

Figure 16: STEP 3: UI Builder - Variable display

- **Digital Input Display**: Displays the state (0 or 1) of a configured input device (from "STEP 1: Configure") in real time as the program runs.

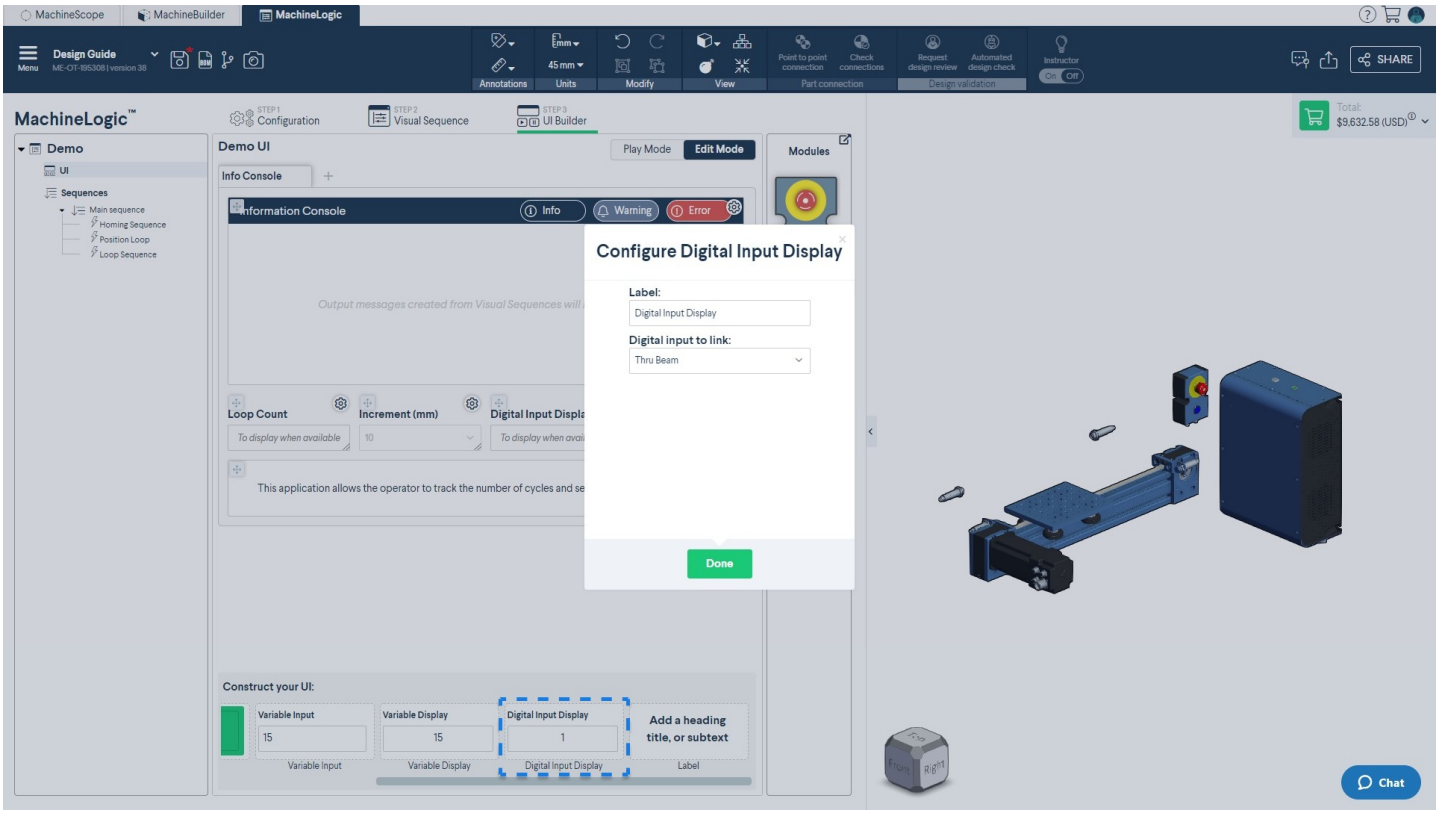

Figure 17: Digital input display

- **Label**: A label allows you to add text to display information to the operator in the operator interface. A few styling options are offered: normal, header, subtext.

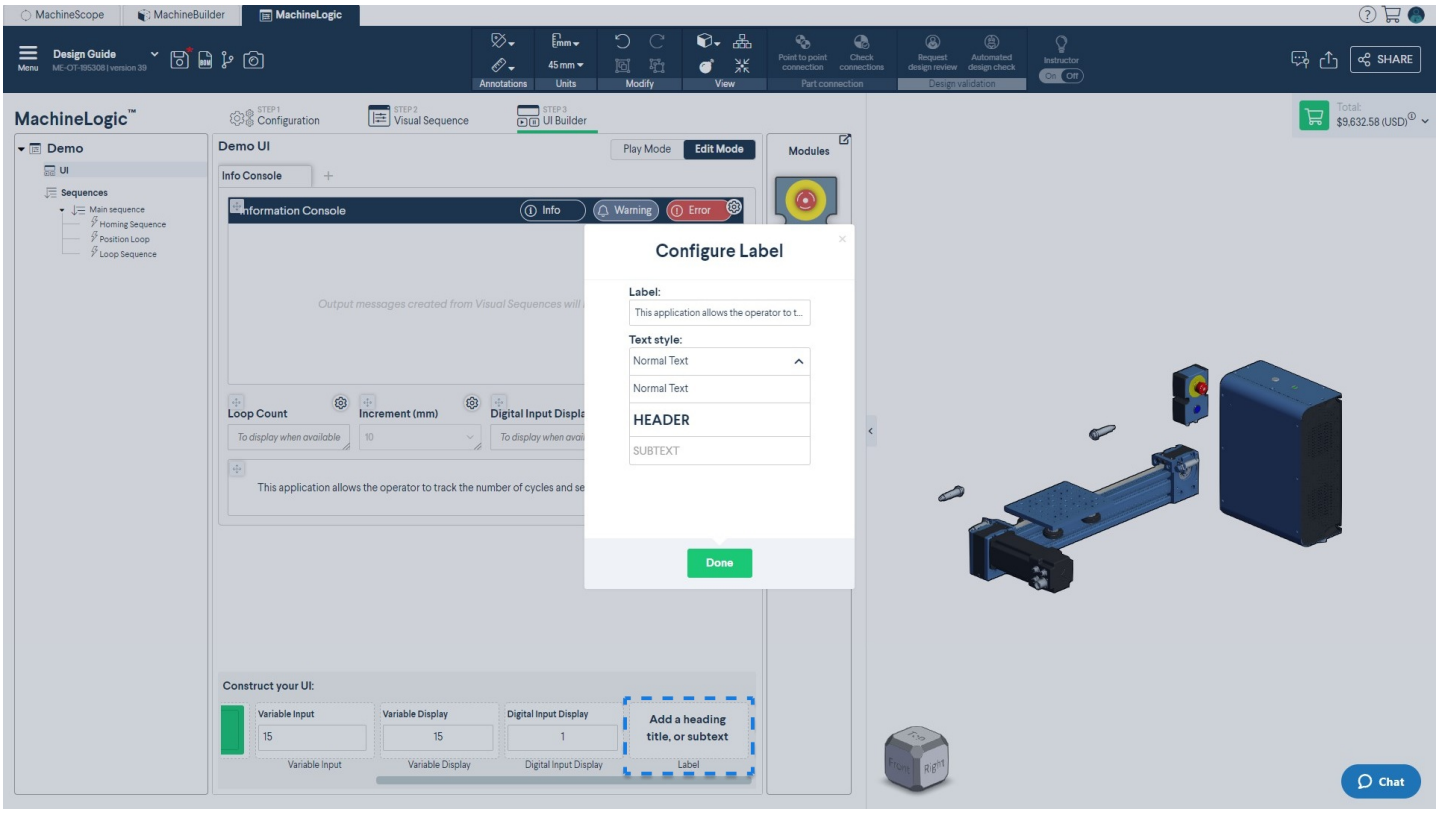

Figure 18: Label

1. Once you are ready to use the operator interface, toggle to "Play Mode" and press the play button.

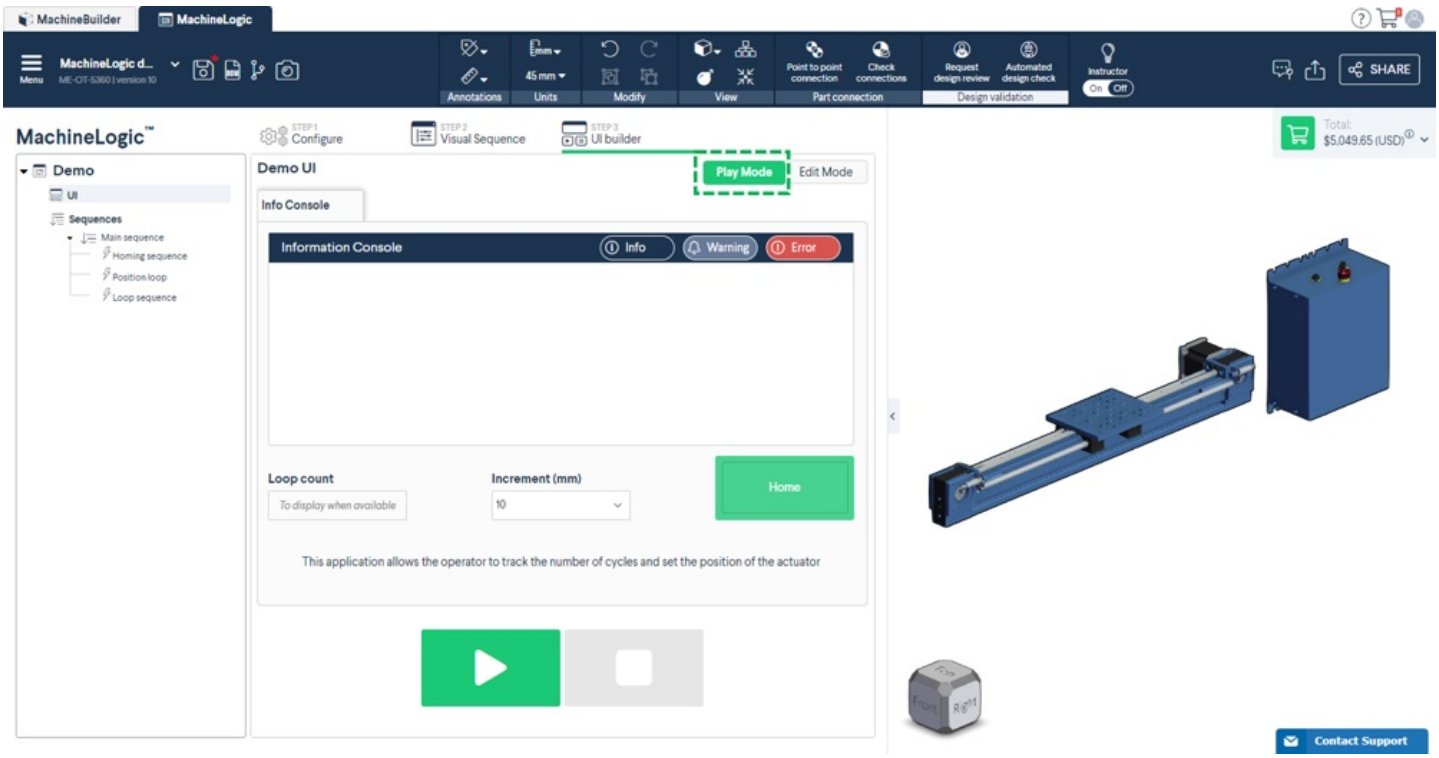

Figure 19: Play mode

Motion command glossary

Move to home	**The "Move to Home" command allows you to position your actuator at its Home position, where the home sensor is located. You need a homing sensor to use this command. This command is recommended at the beginning of each application for each actuator, because its set the "zero" of your position.**

Move to home	The “Move to Home” command allows you to position your actuator at its Home position, where the home sensor is located. You need a homing sensor to use this command. This command is recommended at the beginning of each application for each actuator, because its set the “zero” of your position.
Move Relative	This command allows you to move relative to the current position of the gantry. This will accept both negative or positive distance (mm) values. If the “Move Relative” command exceeds the travel length of the actuator, it will stop at the closest home or end sensor. Example 1. For a timing belt actuator (travel length is 855 mm) at its home position, and you execute a move relative command +1000 mm, the gantry will stop at the end sensor. Example 2. For a rotary actuator if the current position is 90 deg, a move relative command -20 deg would bring the gantry to 70 deg.
Move to position	Move to Position command allows the actuator to move in an absolute position from the home position. Note that a Move to Home command is required before adding the Move to Position command in the application. Move to Position command only allows for a positive value input.
Set System Speed	Configures coasting speed of all actuators in the system. All movement/commands entered after this “Set Speed” command will follow this speed until a new “Set Speed” command is entered.
Set System Acceleration	Configures acceleration of all actuator in the system. All movement/commands entered after this “Set Acceleration” command will follow this acceleration until a new “Set Acceleration” command is entered.
Set angle	Use this function when you would like to redefine an angle on your current rotary actuator position. This command does not move the actuator, but will override its position value.
Move to closest angle	This command allows you to rotate your rotary actuator to the specified position using the desired direction: • **Positive**: Rotates in the positive direction of your motor. Default: Counterclockwise • **Negative**: Rotates in the positive direction of your motor. Default: Clockwise • **Shortest path**: Rotates either clockwise or counterclockwise using the less rotational movement possible Your rotary actuator will always move less than a full turn. The specified angle needs to be between -360 and 360 deg.
Start continuous move	For conveyors and rotary actuators, you have the ability to actuate them continuously. It may be considered a start command to actuate the conveyor/rotary actuator continuously. Typically, until an event happens or under a timeframe.
Stop continuous move	To stop a continuous motion, you should use the “Stop Continuous Move” command. Please note that the “Stop All Motion” command would also stop any continuous moves.
Stop all motion	To stop all motion from the previous commands. This command is useful when you have various “Set continuous move” commands and you would like all movement to stop using one command.
Push	Extends the piston of the selected pneumatic actuator.
Pull	Retracts the piston of the selected pneumatic actuator.
Idle	Disables the output of the pneumatic actuator, allowing the movement of the pneumatic actuator to be “free” (could easily by extended or retracted with any external force).

Add Wait	
	Command Icon
Amount of time	Provides the ability to wait for a certain time delay before performing the next command in a sequence.

Add Wait	
Event	Wait for event command creates the ability to wait for an event topic and/or message to be generated before executing subsequent commands in the sequence. For an example, this gives the ability for an operator to generate an event by clicking a button in the UI builder to execute a sequence that is waiting on an event topic and/or message. Additionally, it provides the ability to wait for a given message on a given MQTT topic before performing the next command in a sequence.
Motion completion	Provides the ability to wait until all moves complete before performing the next command in a sequence. This command will be useful for conditional statements, where you would only execute another command if a certain motion has completed.
Digital input	Provides the ability to wait for a given digital input of an digital IO module on specified state (0 or 1) before performing the next command in a sequence. For an example, depending on the state of a sensor (0: no object sensed, 1: object sensed), you could add a command that will only move an actuator of the state of the sensor is "1".
Digital input edge	Provides the ability to wait for a given digital input of an digital IO module to transition : from 0 to 1 for a rising edge, from 1 to 0 for a falling edge before performing the next command in a sequence. For an example, if you would like to command a conveyor to move only if a sensor detected a box moving past a sensor, triggering the state to go from 0 to 1 and 1 to 0.

Add output	
	Command Icon
Digital output	Add a command to activate your digital I/O compatible components using pins (0, 1, 2, 3). Each pin will trigger an action from that connected component (i.e pneumatic actuator, status light, etc.). Example. If you have set up your configuration with a pneumatic actuator, you could activate your actuator by inputing the output as "Pneumatic actuator" from the drop-down menu. Afterwards, enter your pin associated with that action (0 or 1).
Generate event	Generate Event command gives the ability to generate an event, with a given topic and an optional message. For example, it allows to resolve any active "WaitForEvent" commands that would be waiting on the same topics and messages

Add Execution	
	Command Icon
Execute in parallel	This allows you to play/run a command in parallel with another command. The command you input with the "Execute in parallel" will run in parallel with the next command. Tip: Under your main sequence, you should add most of your execution commands there.
Execute in series	This allows you to play/run a sequence within another sequence, and wait for its completion before moving on to the next instruction. This is useful for repetitive sequential movements. Tip: Under your main sequence, you should add most of your execution commands there.
Terminate	Terminates the execution of the program. If you know your machine is experiencing an error, use this command. This will allow your machine to be in the state of a "software stop".

Set Variable	
	Command Icon

Set Variable	
From device	Set variable command category allows the “Initial Value” of a variable (set from the “Variables tab”) to change. All the commands executed after this command will use that new value. “Set Variable from Device” allows a variable to take the value of one input pin of the digital IO module.
From expression	Set Variable command category allows the “Initial Value” of a variable (set from the “Variables” tab) to change. All the commands executed after this command will use that new value. “Set Variable from Expression” allows a variable (set from the “Variables” tab) to take the value specified in the expression field. The expression field supports numbers, mathematical operators (*, /, +, -) and variable names.
From expression array	**Set Variables from Expression** allows variables (set from the **Variables** tab) to take an array of values specified in the **Expression** field. The **Expression** field supports numbers, mathematical operators (*, /, +, -) and **Variable Names**.

Add Message	
	Command Icon
Information Console	Adding a message to the information console allows you to output a message to the UI builder for the operator to view. Variables, functions and text could be entered as a message. To display the message to the operator, ensure you drag and drop the information console widget in the UI Builder.
URL	Adding a message to a URL allows you to send a message (“Pack message”) to an external server and store the resulting output of the server in the “Unpack Variable Name”.

Operate your simulation

MachineLogic Commands	
	**Execute/Resume** the sequences
	**Pause** the sequences
	**Stop** simulation
	**Download** the application into a file to deploy onto the MachineMotion controller
	**Upload** an existing MachineLogic file