Learning DigiShow

7

Interactive Applications

Robin Zhang and Labs 2025

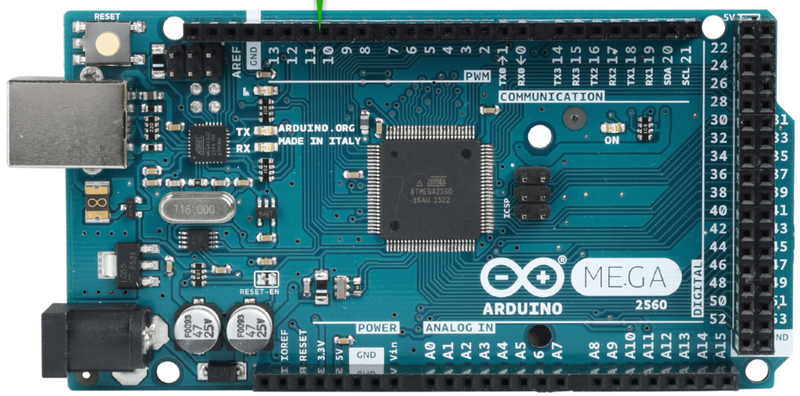

When making interactive props and interactive art installations, controllers, sensors, and actuators are always indispensable.

Arduino Interface

What is Arduino ?

Arduino is an open source electronic controller. IO pins on it can be used as input to connect sensors, or as output to drive actuators such as LED lights, relays, motors, servos, electromagnets, and robot joints. Also, Arduino PLC is based on the Arduino core and provides the ability to adapt to industrial sensors and actuators, suitable for large-scale devices.

5V MCU IO pins can be connected to low-power electronic components to drive sensors and actuators

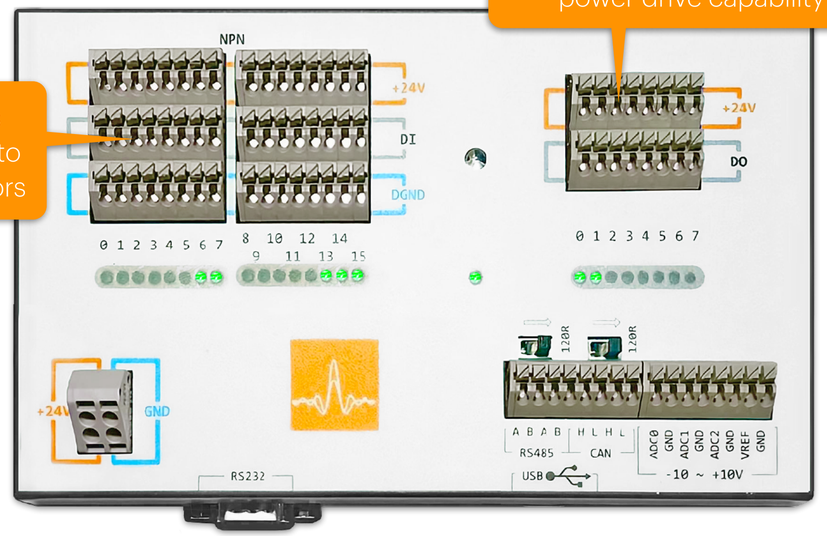


Arduino MEGA

VS

PLC output ports, with greater power drive capability

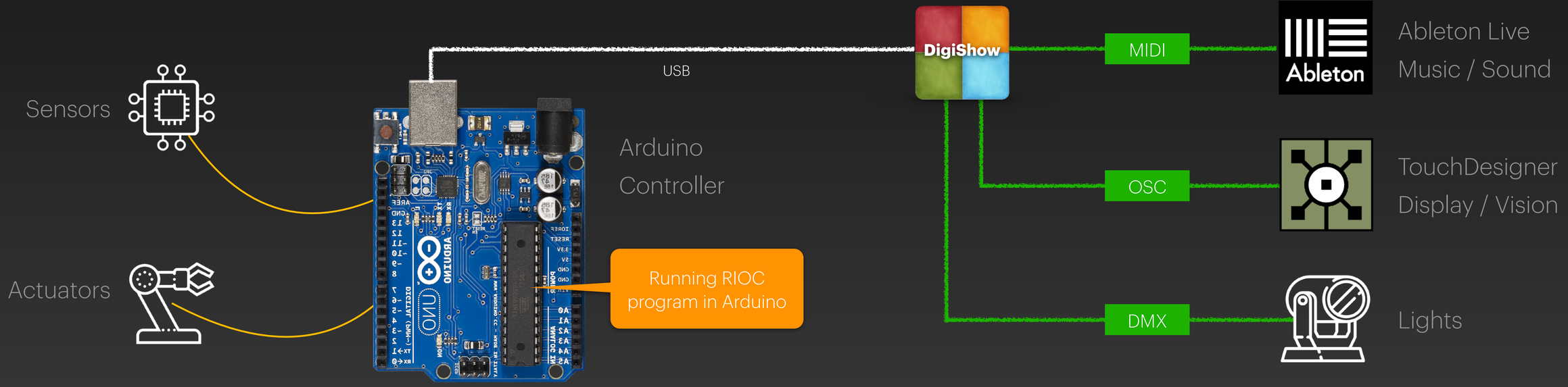
24V PLC input ports can directly connect to industrial-grade sensors



Arduino PLC (Aladdin)

Working with Arduino

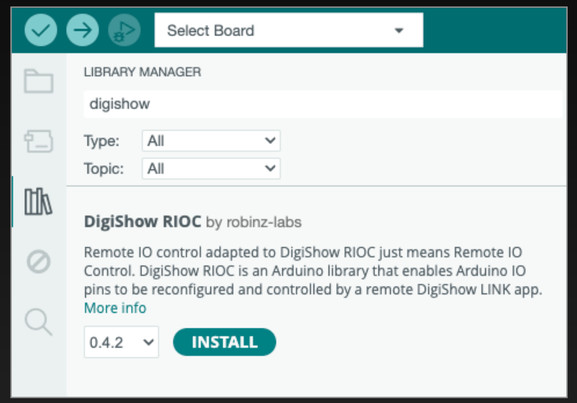
Upload and run the **RIOC** (Remote IO Control) program we prepared for you in Arduino, and connect the Arduino to the USB port of your computer, and the sensors and actuators connected to its IO pins can be easily controlled by DigiShow. Other software can also obtain this control ability of sensors and actuators through DigiShow signal mapping without any programming.



Install RIOC program to Arduino

First, let's prepare an Arduino controller

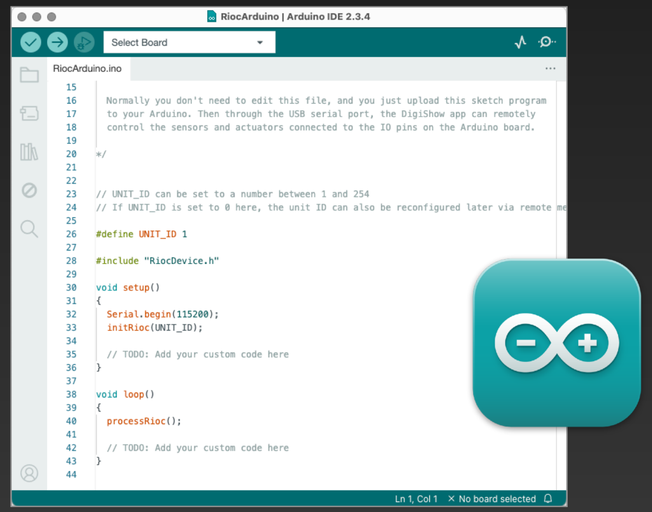
1. We recommend you choose an Arduino UNO, MEGA or Nano board (or a PLC with Arduino core) , or an ESP32 board.



Then, get the RIOC program and upload it to your Arduino!

2. Open Library Manager in the Arduino IDE, find and install the library named DigiShow RIOC .

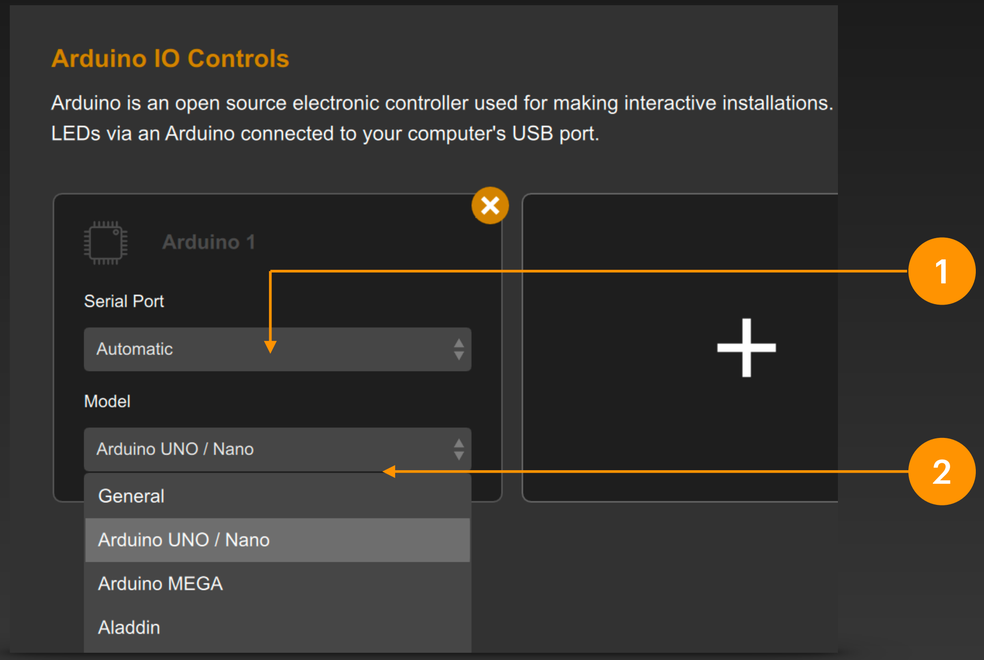
3. Then select File > Examples > DigiShow RIOC > RiocArduino in the Arduino IDE menu and upload the program to your Arduino.



(You can download the Arduino IDE and get more detailed instructions on using Arduino at www.arduino.cc)

Arduino Interface Configuration

After the Arduino with RIOC program is connected to the USB port of the computer, DigiShow can control the connected sensors and actuators through it. Users can add Arduino interfaces to the current project in the Arduino section of the Interface Manager.



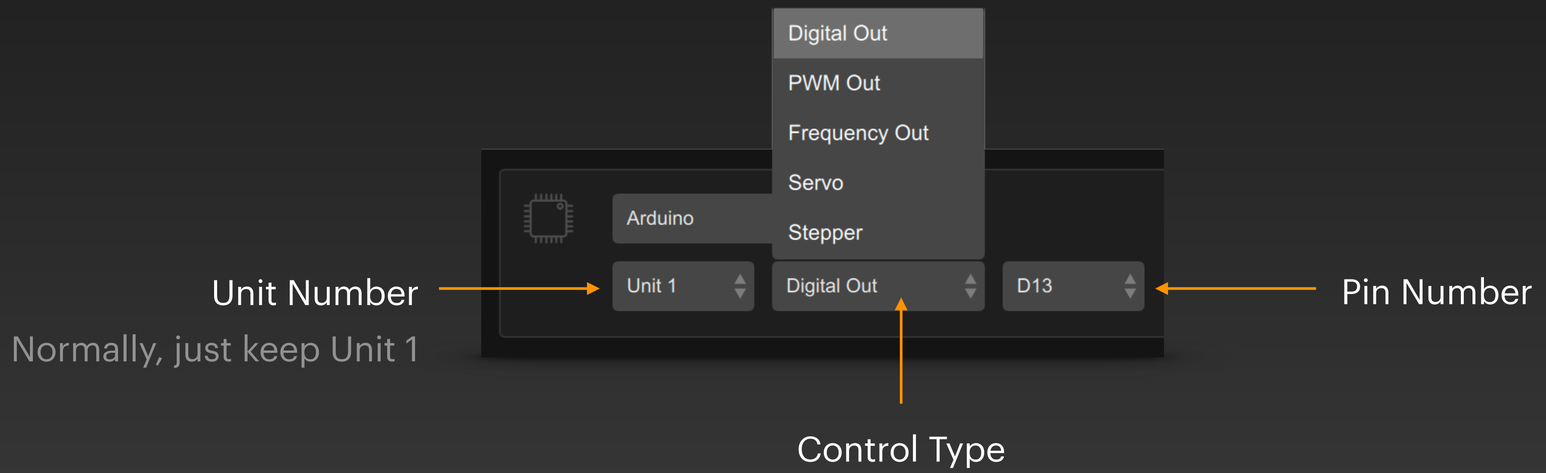
Select the Serial Port that your connected Arduino occupies on your computer. If the Arduino you are using is a fully compatible Arduino UNO or MEGA, just select Automatic.

Select the Model of Arduino you are connecting to. If the model is not in the list (such as ESP32 Arduino), select General.

Arduino Signal Input and Output

In the signal link table, set the **output** end of the signal bar to Arduino, and we can control the pins on the Arduino for output in DigiShow, and its type can be: digital output, PWM output, frequency output, servo, stepper motor, etc.

Set the **input** end of the signal bar to Arduino, and we can also control the pins on the Arduino for input in DigiShow, and its type can be: digital input, analog input, encoder input, etc.

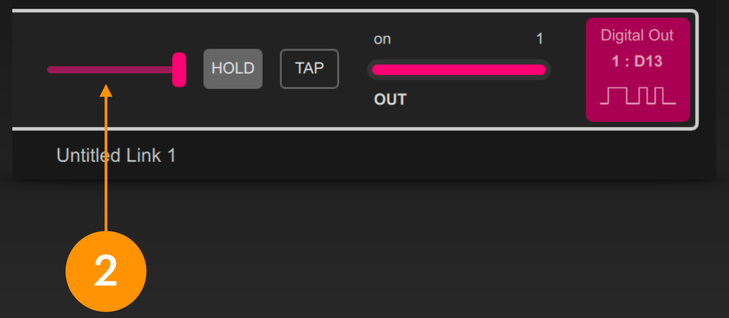


Arduino Digital Output

We can manipulate a pin on the Arduino for digital output in the output end of the DigiShow signal bar:

Select the Arduino interface in the signal bar output, select a pin and set its type to Digital Out.

This example is used to control the onboard LED light connected to the D13 pin on and off.

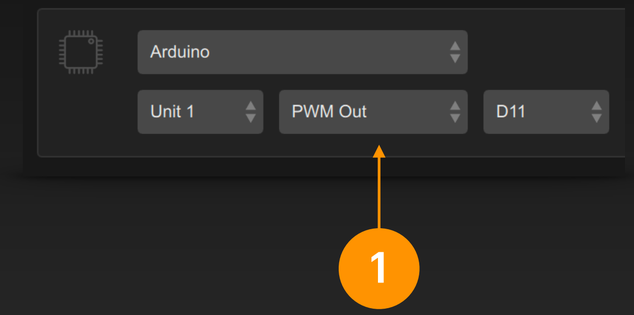


After the project is started, you can change the output state of the Digital Out type pin by switching the fader on the signal bar. It is a binary signal.

At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

Arduino PWM Output

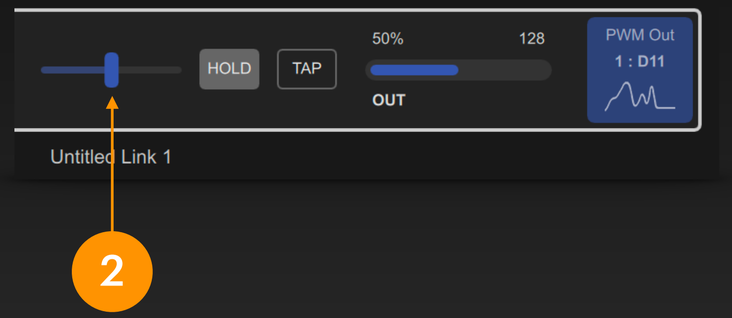
We can manipulate a pin on the Arduino for PWM output in the output end of the DigiShow signal bar:



Select the Arduino interface in the signal bar output, select a pin and set its type to PWM Out.

PWM Out is often used for LED dimming, motor speed control, etc.

PWM output corresponds to analogWrite() in Arduino programming.

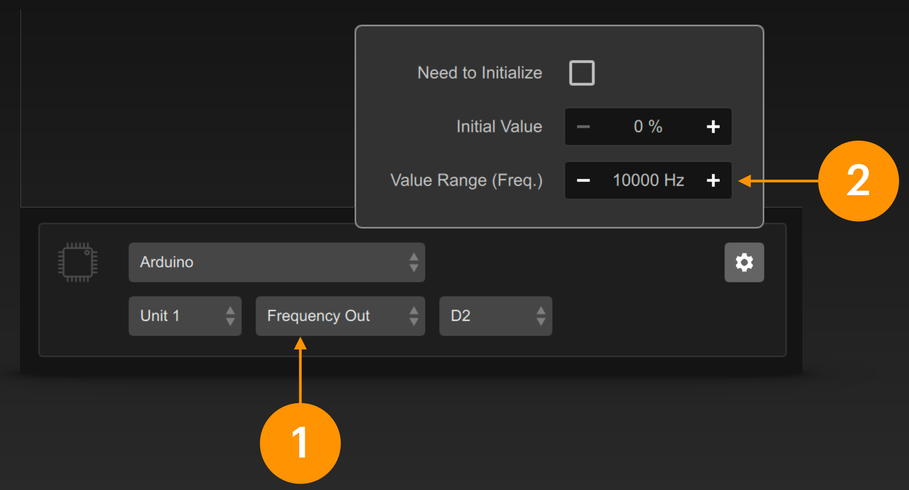


After the project is started, you can change the output value of the PWM Out type pin by moving the fader on the signal bar. It is an analog signal with a value range of 0 ~ 255.

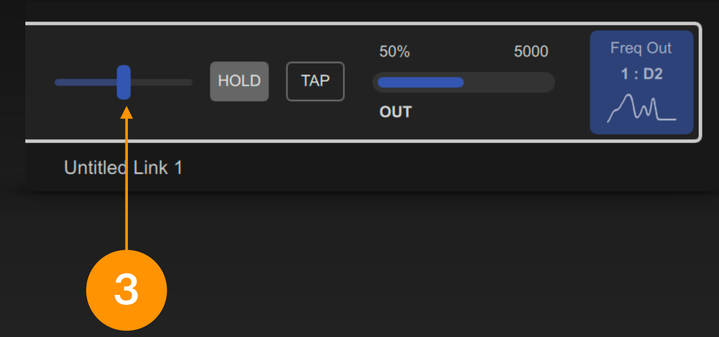
At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

Arduino Frequency Output

We can manipulate a pin on the Arduino for frequency output (i.e. oscillator output) at the output end of the DigiShow signal bar:



The default frequency output range is 0 ~ 1000Hz. You can also click the gear button ⚙ and modify the value in Value Range (Freq.) in the pop-up setting panel to modify the frequency adjustment range.



After the project is started, you can change the output value of the Frequency Out type pin by moving the fader on the signal bar. It is an analog signal with a user-defined value range.

At this point, you can also select an input for the signal bar to achieve the corresponding signal mapping.

Select the Arduino interface in the signal bar output, select the pin and set the type to Frequency Out.

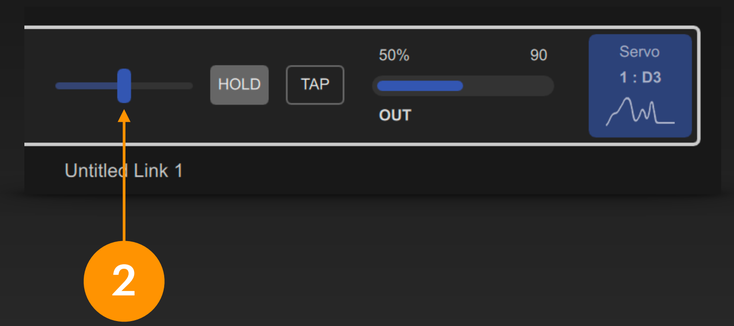
Frequency Out will produce an oscillating square wave with adjustable frequency, which is often used for speed regulation of motor drives or buzzer sound.

Frequency output corresponds to tone() in Arduino programming.

Arduino Servo

We can manipulate a pin on the Arduino for servo signal output at the output end of the DigiShow signal bar:

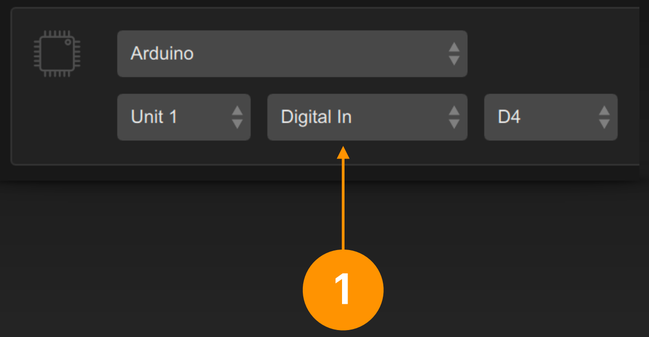
Select the Arduino interface in the signal bar output, select a pin and set its type to Servo.



After the project is started, moving the fader on the signal bar can change the output value of the Servo type pin, that is, adjust the angle position of the servo. It is an analog signal with a value range of 0 ~ 180 degrees.

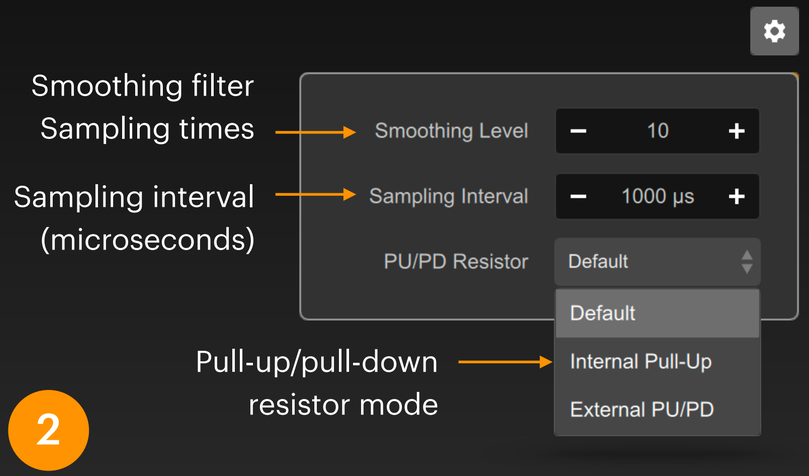
Arduino Digital Input

We can manipulate a pin on the Arduino for digital input in the input end of the DigiShow signal bar:



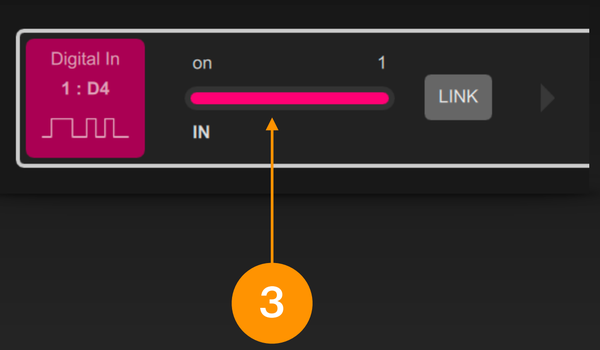
Select the Arduino interface in the signal bar input, select a pin and set its type to Digital In.

Digital input is usually used to connect switches, buttons or various sensors.



You can also click the gear button ⚙ to modify the signal filtering and other parameters of this Digital Input in the pop-up setting panel.

As shown in the example above, if the input state changes and remains in the new state for more than 10 consecutive times, the change of this input signal will be triggered.



After the project is started, the input end of the signal bar will display the input signal state of the pin. It is a binary signal.

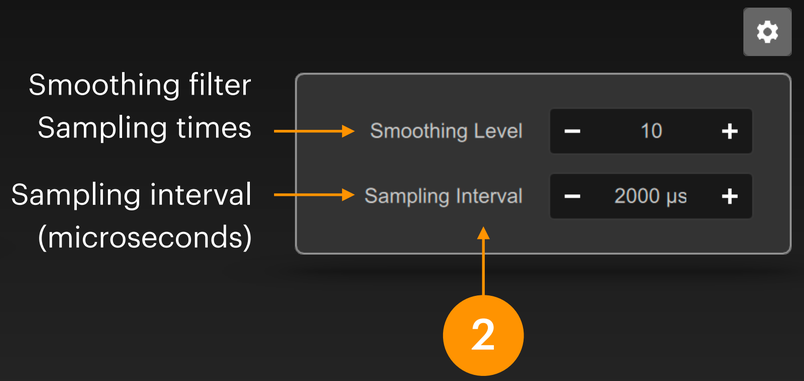
At this time, you can also select an output for the signal bar to achieve the corresponding signal mapping.

Arduino Analog Input

We can manipulate a pin with ADC on the Arduino for analog input in the input end of the DigiShow signal bar:

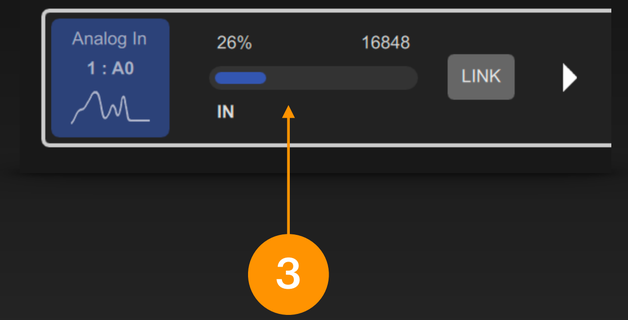
Select the Arduino interface in the signal bar input, select a pin and set its type to Analog In.

Analog input is usually used to connect potentiometers or various analog sensors.



You can also click the gear button ⚙ to modify the signal filter parameters of this Analog Input in the pop-up setting panel.

As shown in the example above, after applying this filter parameter, the original signal at the input is sampled every 2000 microseconds, and the value of the input signal will be the average of the most recent 10 samples.



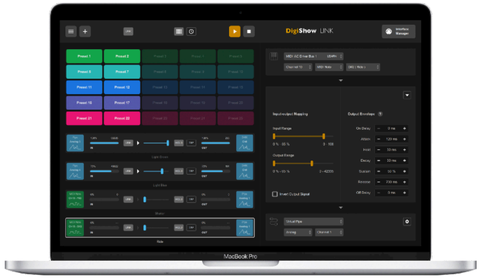
After the project is started, the input end of the signal bar will display the input signal value of the pin. It is an analog signal with a value range of 0 ~ 65535

At this time, you can also select an output for the signal bar to achieve the corresponding signal mapping.

Modbus Interface

What is Modbus ?

Modbus is a communication protocol commonly used between automation devices in industrial scenarios, originally developed by Schneider Electric in 1979. DigiShow can achieve online control with machinery, robots, and a wide variety of PLC controllers, IO modules, sensors, actuators, etc. through the Modbus protocol based on serial bus or IP network communication.



DigiShow app

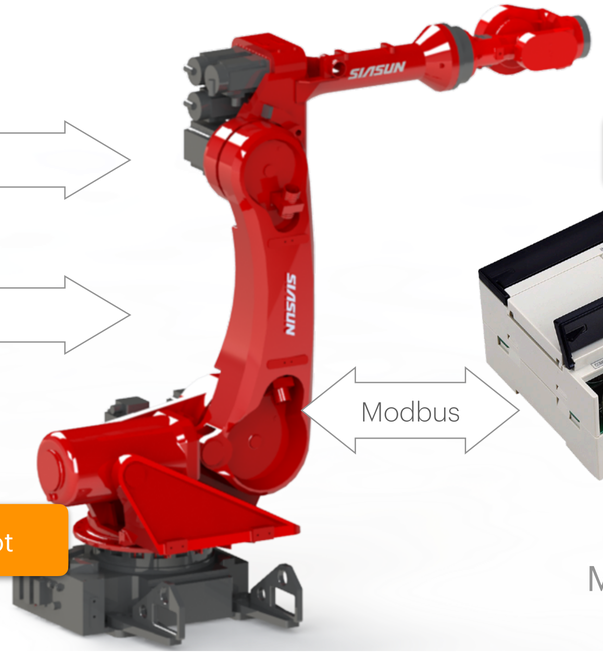
Modbus Master Computer

Modbus RTU

over 485 serial bus

Modbus TCP

over IP network



Robot

Modbus

IO Module

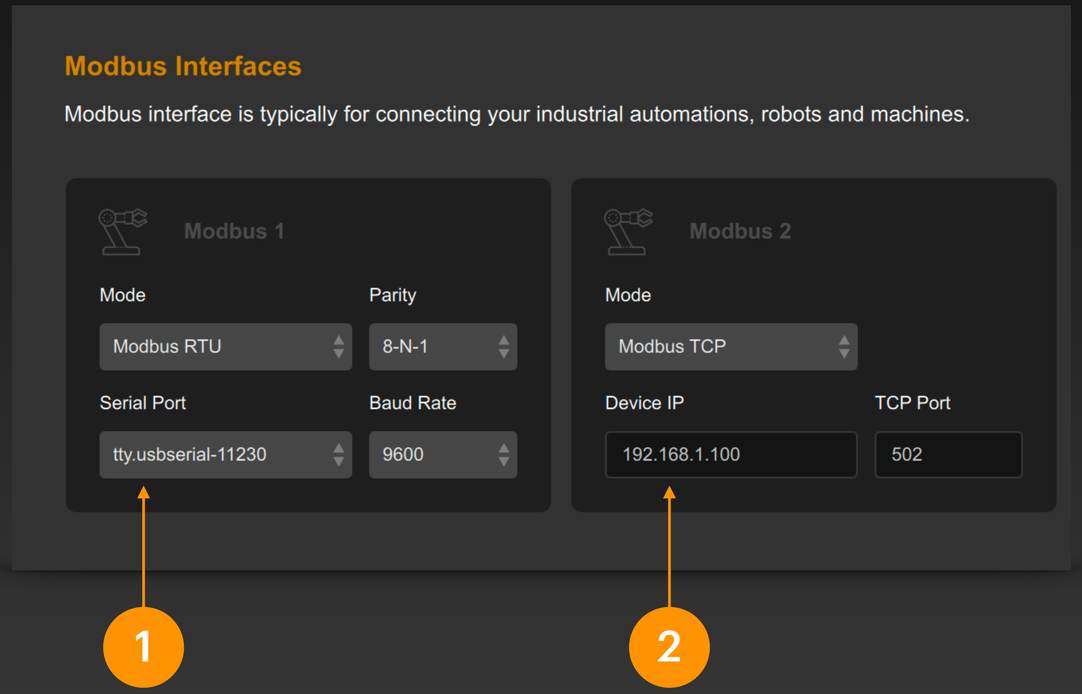
Modbus

Sensor

Modbus Slave Devices

Modbus Interface Configuration

DigiShow supports communicating with Modbus slave devices over serial buses or IP networks. Users can add Modbus interfaces to the current project in the Modbus section of the Interface Manager.



1

If your Modbus device uses serial bus communication, please select Mode as Modbus RTU, select the Serial Port occupied by the device connected to the computer, and set Baud Rate, Parity, etc.

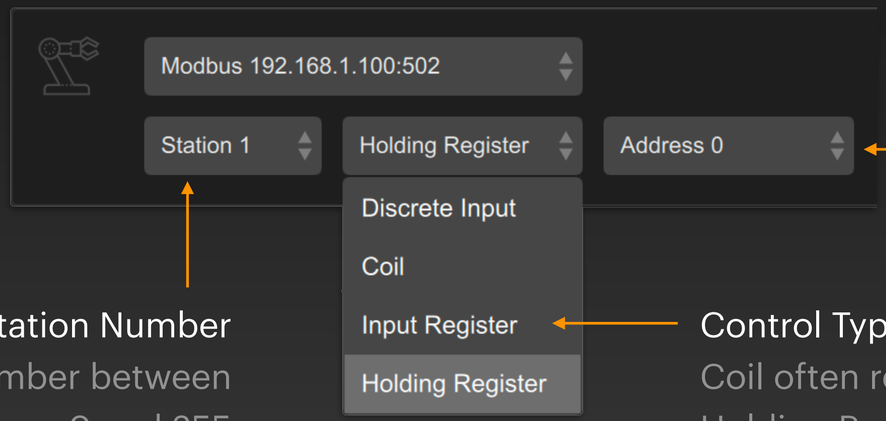
2

If your Modbus device uses IP network communication, please select Mode as Modbus TCP, set the Device IP address in the network and TCP Port for the service (usually port 502).

Modbus Signal Input and Output

In the signal link table, set the **output** end of the signal bar to Modbus, and we can control the output channel of the Modbus device in DigiShow. Its type can be: Coil, Holding Register.

Set the **input** end of the signal bar to Modbus, and we can monitor the input channel of the Modbus device in DigiShow. Its type can be: Discrete Input, Coil, Input Register, Holding Register.



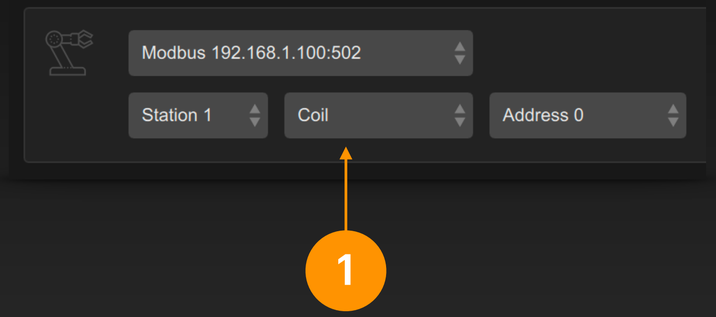
Channel Address
Usually a number between
0 and 65535

Device Station Number
Usually a number between
0 and 255

Control Type
Coil often refers to the relay or binary control point in the device
Holding Register often refers to the analog control point in the device
Discrete Input refers to the alternative binary input in the device
Input Register refers to the alternative analog input in the device

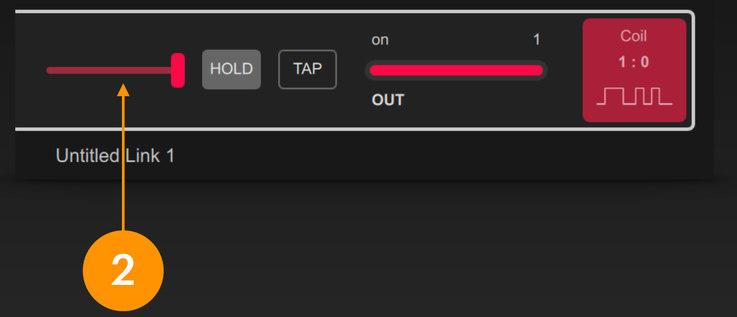
Modbus Coil Output

We can manipulate a coil output on the Modbus device in the output end of the DigiShow signal bar:



Select the Modbus interface in the signal bar output, select the device station number, set the control type to Coil, and select the channel address of the coil.

Coil output is often used to switch On/Off state of relays or binary control points in the device.

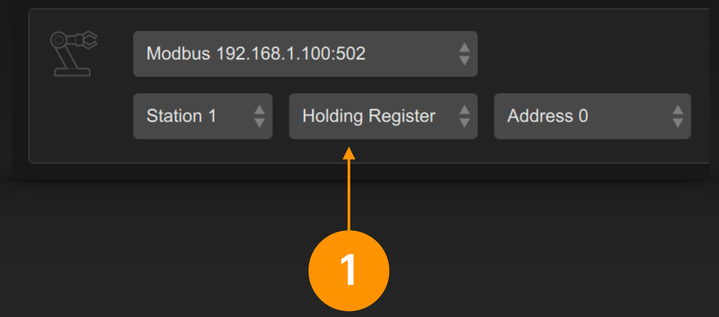


After the project is started, you can change the output state of the coil by switching the fader on the signal bar. It is a binary signal.

At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

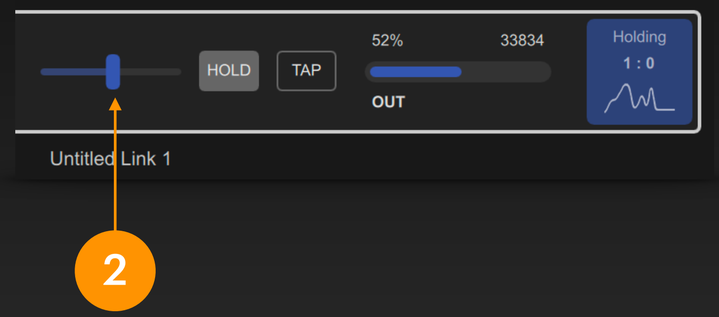
Modbus Register Output

We can manipulate a register output on the Modbus device in the output end of the DigiShow signal bar:



Select the Modbus interface in the signal bar output, select the device station number, set the control type to Holding Register, and select the channel address of the register.

Register output is to modify the value in the register, which is often used to change the status of the analog control point in the device.

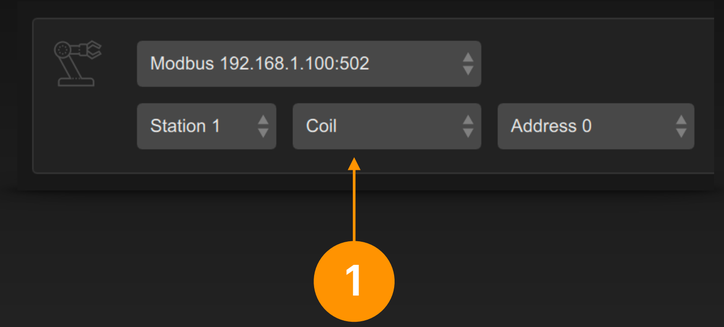


After the project is started, moving the fader on the signal bar can change the status value of the register. It is an analog signal with a value range of 0 ~ 65535.

At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

Modbus Coil Input

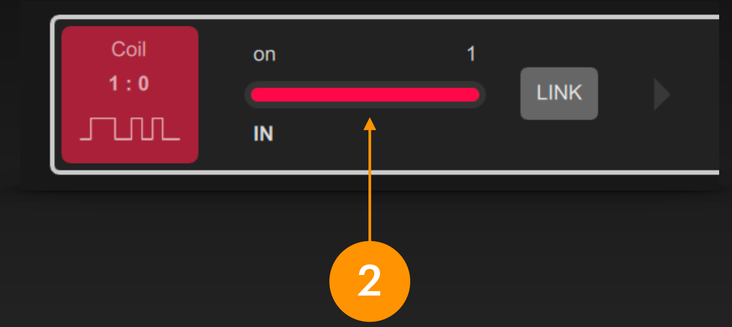
We can monitor a coil input on the Modbus device in the input end of the DigiShow signal bar:



Select the Modbus interface in the signal bar input, select the device station number, set the control type to Coil, and select the channel address of the coil.

Coil input is often used to monitor the On/Off state of relays or binary control points in the device.

You can also select the Discrete Input type to monitor alternative binary inputs on the device.

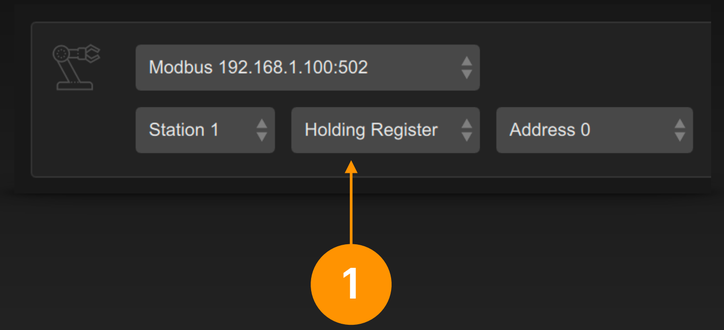


After the project is started, the signal bar input will display the current state of the coil. It is a binary signal.

At this time, you can also select an output for the signal bar to achieve the corresponding signal mapping.

Modbus Register Input

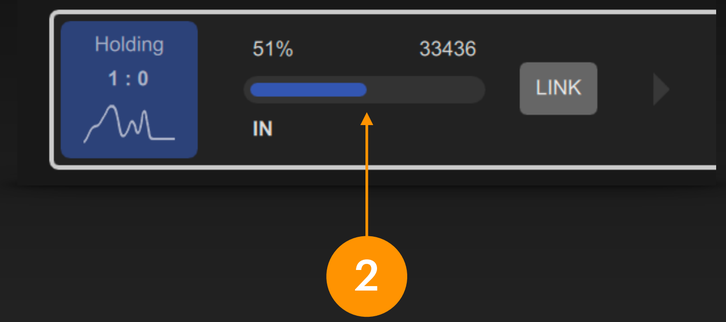
We can monitor a register input on the Modbus device in the input end of the DigiShow signal bar:



Select the Modbus interface in the signal bar input, select the device station number, set the control type to Holding Register, and select the channel address of the register.

Register input is often used to monitor the status of analog control points in the device.

You can also select the Input Register type to monitor alternative analog inputs on the device.



After the project is started, the signal bar input will display the current status value in the register. It is an analog signal with a value range of 0 ~ 65535.

At this time, you can also select an output for the signal bar to achieve the corresponding signal mapping.

OSC Interface

What is OSC ?

OSC (Open Sound Control) is a protocol for communication between computers, synthesizers and multimedia devices. It was first proposed by the Center for New Music and Audio Technologies (CNMAT) at the UC Berkeley in 1997. OSC is widely used in music expression, VJ performance, robot control, and communication between many performing arts related software.

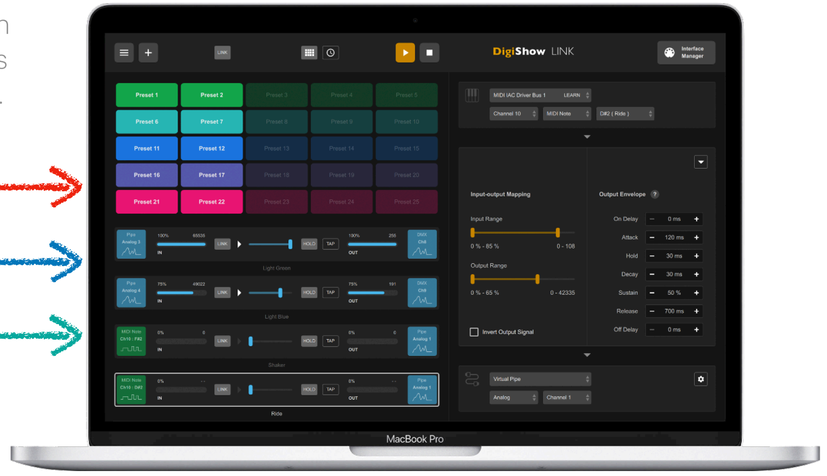
TouchOSC app running
on iPad or iPhone

For example: When a user operates buttons, faders and knobs in TouchOSC, the app will output the corresponding variable values to the destination device or software by sending OSC messages.

/button1	0	/button1	1
/fader2	0.35	/fader2	0.68
/knob3	0.85	/knob3	0.53

OSC Messages

UDP communication over IP networks



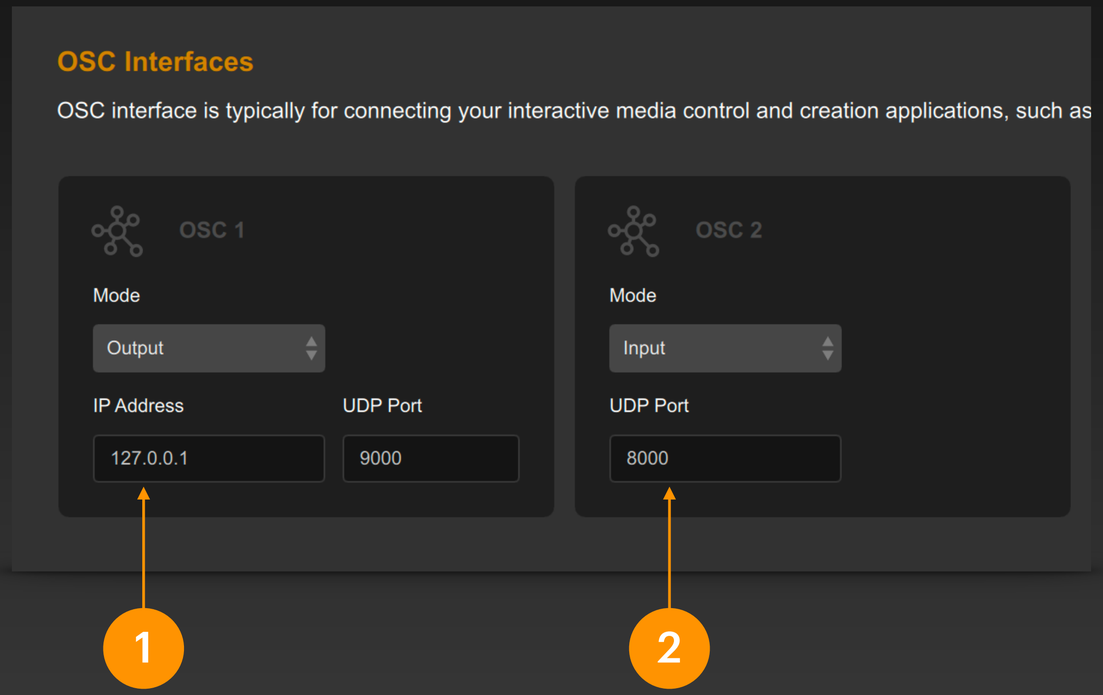
DigiShow app running
on the computer

OSC between Applications

OSC messages can be used to pass signal variables between applications on the computer to collaborate with each other. When DigiShow works with other interactive media creation software (such as TouchDesigner, Unreal Engine), the software can shortly obtain the ability to access input and output signals in DigiShow, and then easily control a series of objects such as sensors, actuators, robots, lights, etc., without any code programming.

OSC Interface Configuration

DigiShow supports bidirectional OSC communication with other applications to complete the input and output of signal variables. Users can add OSC interfaces to the current project in the OSC section of the Interface Manager.



1

If you need DigiShow to send OSC messages to other applications (i.e. output signal variables to other applications), please select Mode as Output and set the IP address of the computer where the application receiving the OSC message is located and the UDP port number of the receiving end.

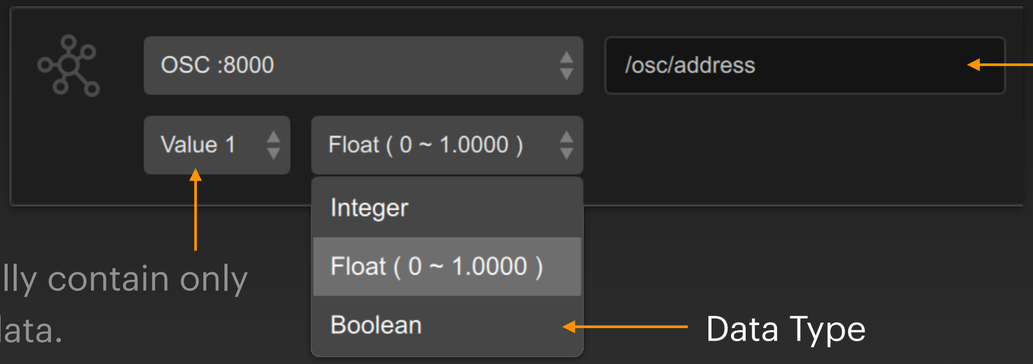
2

If you need DigiShow to receive OSC messages from other applications (i.e. input signal variables from other applications), please select Mode as Input and set the UDP port number used to receive OSC messages.

OSC Signal Input and Output

In the signal link table, by setting the **output** end of the signal bar to OSC, we can send OSC messages to a specific address and transmit specific signal variables to another application.

By setting the **input** end of the signal bar to OSC, we can monitor OSC messages received on a specific port and obtain specific signal variables transmitted from another application.



Data Number

An OSC address can usually contain only a single data or multiple data.
When the address contains multiple data, it is necessary to set the data number to determine its storage location.
When the address contains only a single data, the data number remains 1.

OSC Address

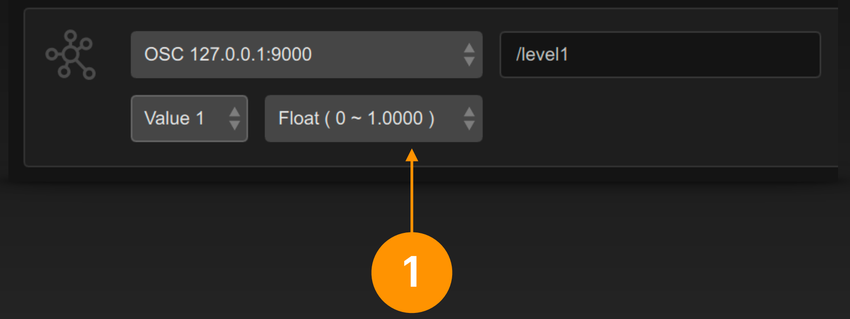
An OSC address is usually a string similar to a file path, which can be regarded as the name of the signal variable transmitted in the OSC message.

Data Type

DigiShow conditionally supports some data types in OSC messages, such as:
Float (0~ 1.0000), its value range must be 0 to 1;
Integer, its value range is 0 to a maximum number set by the user;
Boolean, its value can be True or False.

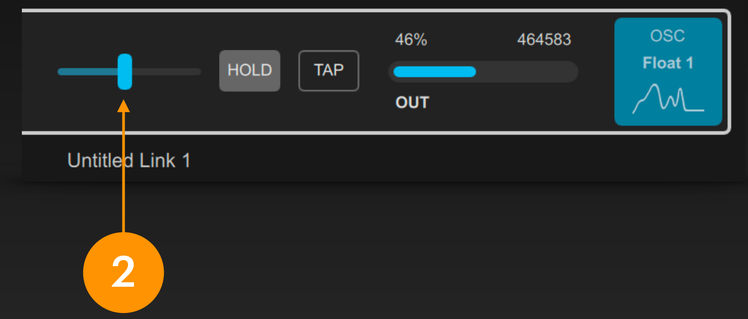
OSC Float Output

We can manipulate the OSC interface in the output end of the DigiShow signal bar to send OSC messages with a specific float signal variable:



Select the OSC interface in the signal bar output, set the OSC address and data number, and select the data type as Float (0~1.0000).

Float data with a range of 0 to 1 often appear in some OSC sessions, such as the fader and knob values on TouchOSC.

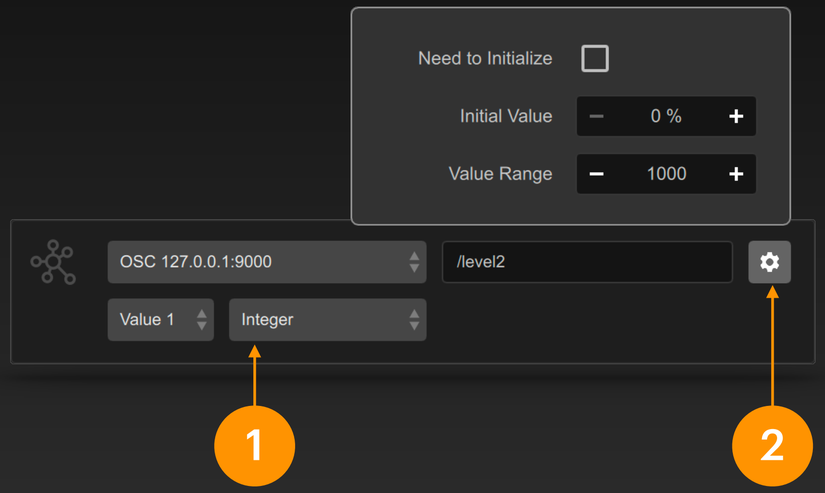


After the project is started, moving the fader on the signal bar can change the value of the signal variable and send the corresponding OSC messages. In DigiShow, it is an analog signal with a value range of 0 ~ 1000000, and in the OSC message, this integer value will be automatically mapped to a floating point number of 0 ~ 1.000000.

At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

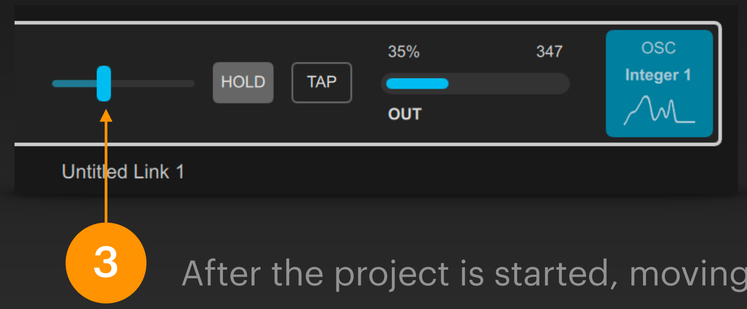
OSC Integer Output

We can manipulate the OSC interface in the output end of the DigiShow signal bar to send OSC messages with a specific integer signal variable:



Select the OSC interface in the signal bar output, set the OSC address and data number, and select the data type as Integer.

The range of this integer can be set by the user. Please click the gear button ⚙ and modify the Value Range in the pop-up setting panel.

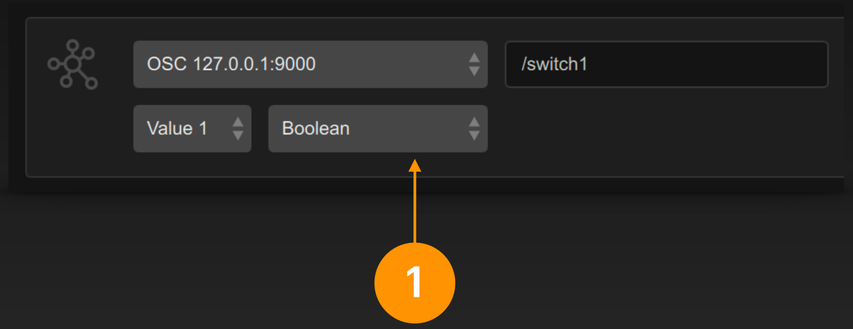


After the project is started, moving the fader on the signal bar can change the value of the signal variable and send the corresponding OSC messages. It is an analog signal with a value range of 0 to a maximum integer set by the user.

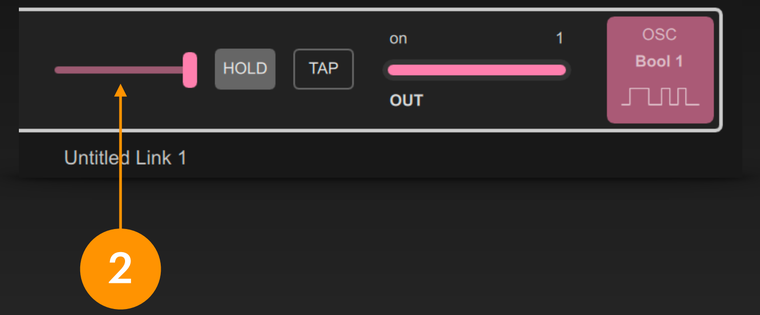
At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

OSC Boolean Output

We can manipulate the OSC interface in the output end of the DigiShow signal bar to send OSC messages with a specific boolean signal variable:



Select the OSC interface in the signal bar output, set the OSC address and data number, and select the data type as Boolean.

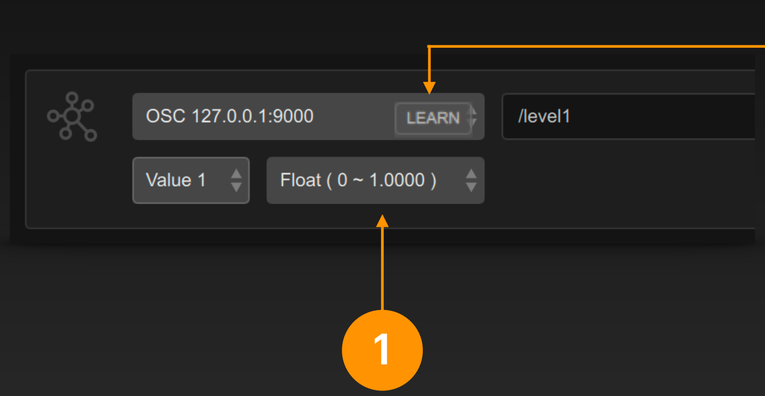


After the project is started, you can change the state of the signal variable and send the corresponding OSC messages by switching the fader on the signal bar. In DigiShow, it is a binary signal, which corresponds to the True or False data type in the OSC message.

At this time, you can also select an input for the signal bar to achieve the corresponding signal mapping.

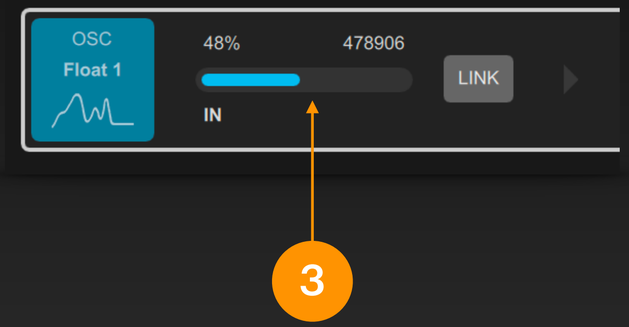
OSC Signal Input

By setting the input end of the signal bar to OSC, we can monitor the received OSC messages with a specific signal variable transmitted from another application.



Select OSC interface in the signal bar input, set OSC address and data number, and select data type. After applying the settings, this input end will receive the signal variable in the OSC messages that meets the set requirements.

We can also actively learn OSC messages sent by external software. Please start the project first, and the LEARN button will appear here. Click this button and operate controls in external software (such as TouchOSC), and the corresponding OSC messages sent will be automatically recognized by DigiShow and refresh the parameters of the signal bar input end.

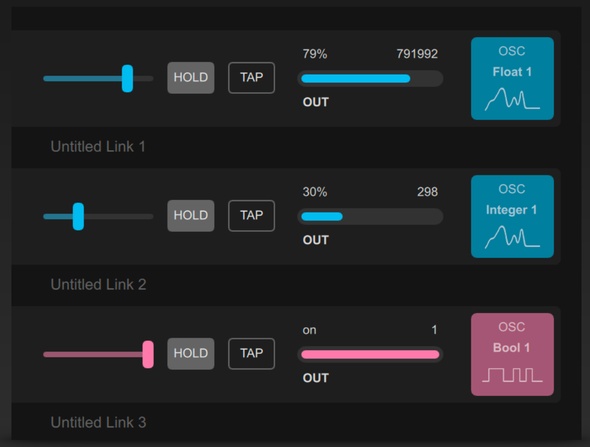


Once the DigiShow project is started, the signal bar input will display the value of the signal variable that meets the set requirements in the received OSC messages.

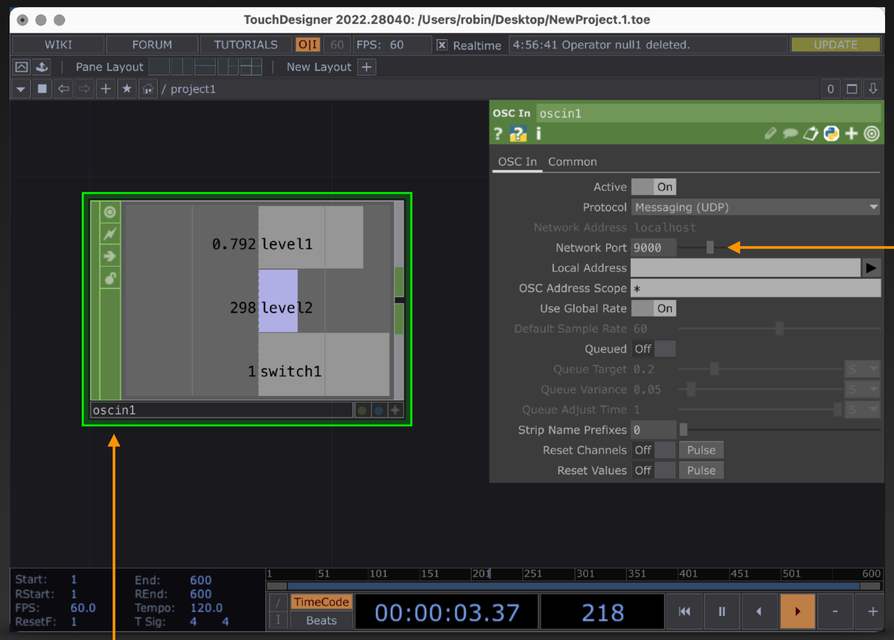
At this time, you can also select an output for the signal bar to achieve the corresponding signal mapping.

OSC Output to TouchDesigner

OSC messages output by DigiShow can be easily received in TouchDesigner, so that the CHOP data in TouchDesigner and the signal variables in DigiShow can be updated synchronously.



OSC Messages



2

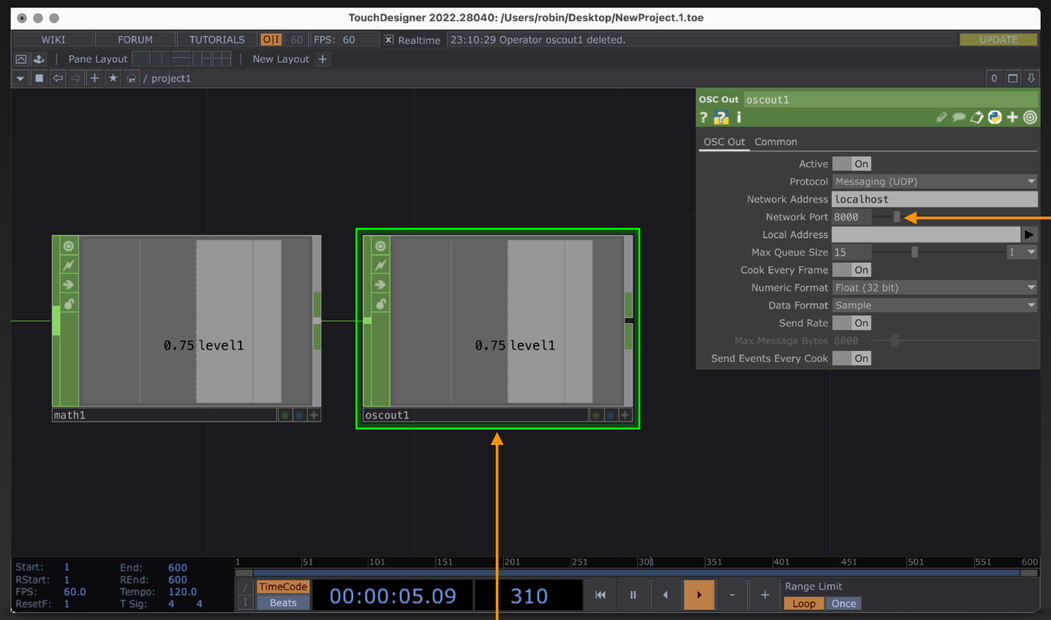
Set the Network Port in the parameters panel of the OSC In CHOP to match the UDP port number set in the DigiShow OSC output interface.

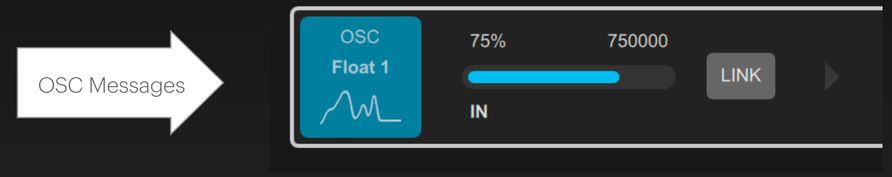
1

Add an OSC In CHOP to the network view in TouchDesigner

OSC Input from TouchDesigner

OSC messages output from TouchDesigner can be easily received in DigiShow, so that the signal variables in DigiShow and the CHOP data in TouchDesigner can be updated synchronously.





2

In the parameters panel of the OSC Out CHOP, set the Network Port to match the UDP port number set in the DigiShow OSC input interface.

Set Numeric Format to Float (32 bit) or Int (32 bit), and Data Format to Sample.

1

Add an OSC Out CHOP to the network view in TouchDesigner, and set an input data source for it. The source data can be a float number between 0 and 1, or an integer.

Summary

- Understand the relevant knowledge of Arduino and IO controls
- Prepare an Arduino with RIOC program for use in DigiShow
- Learn to use Arduino interface for signal input and output in DigiShow
- Understand the concept and usage of Modbus
- Understand the concept and usage of OSC, and learn how to work with TouchDesigner