Learning DigiShow

7

互动装置应用

Robin Zhang and Labs 2025

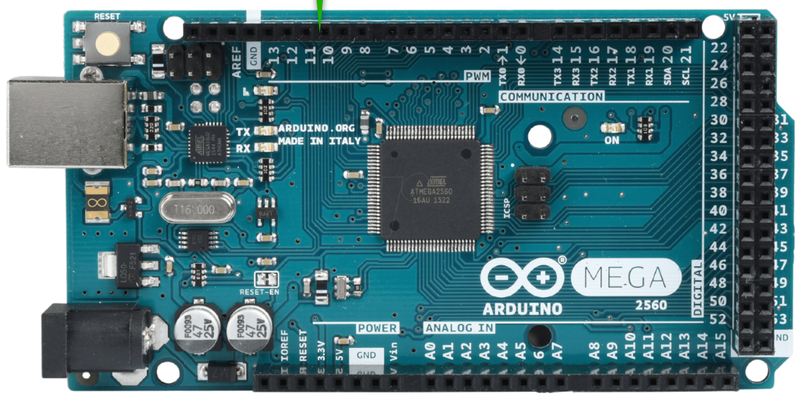

在制作互动道具、互动艺术装置时总离不开控制器、传感器、执行器。

Arduino 接口

什么是 Arduino？

Arduino 是一种开源电子控制器，它的IO引脚可以作为输入连接传感器，也可以作为输出连接驱动LED灯、继电器、电机、舵机、电磁铁、机器人关节等执行器。另外，Arduino PLC 是在 Arduino 控制器核心的基础上提供了对工业级传感器和执行器的适配能力，适合在制作大型装置中使用。

5V 单片机 IO 引脚
可通过连接小功率电子零组件
来驱动传感器和执行器

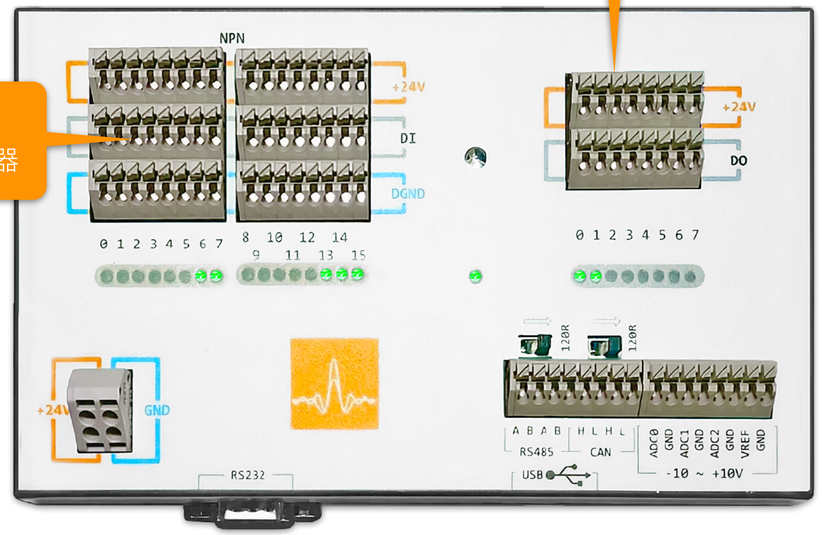


Arduino MEGA

VS

PLC 输出端口, 具有更大功率的驱动能力

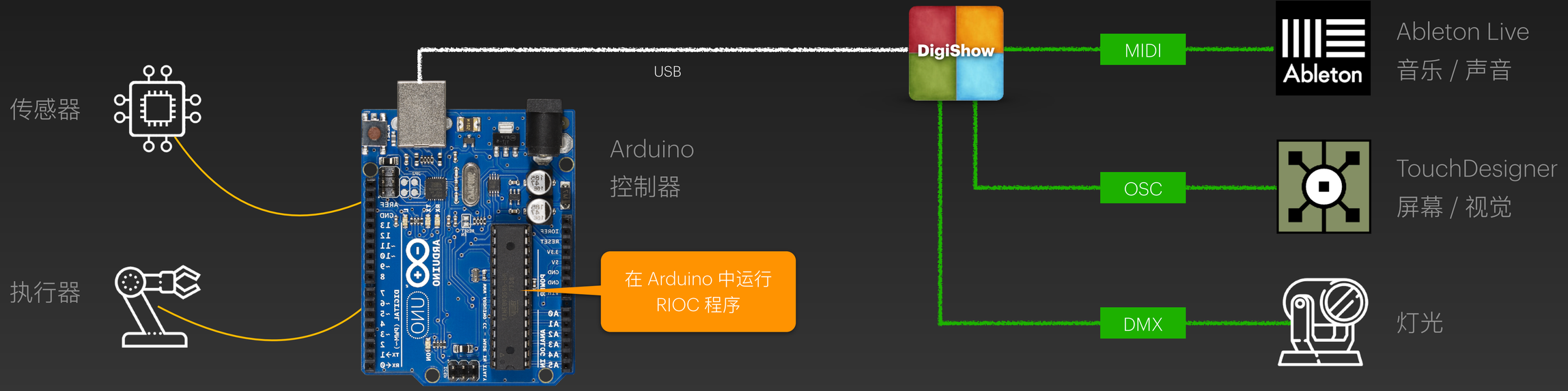
24V PLC 输入端口
可直接连接工业级传感器



Arduino PLC (Aladdin)

与 Arduino 一起工作

在 Arduino 中写入并运行我们预先为你准备好的 RIOC (Remote IO Control) 程序，并将 Arduino 接入电脑的 USB 端口中，它的IO引脚上连接的传感器和执行器就都能被 DigiShow 轻松控制了。更可透过 DigiShow 的信号映射来让其他软件也能获得这种对传感器和执行器的操控能力，而无需任何编程。

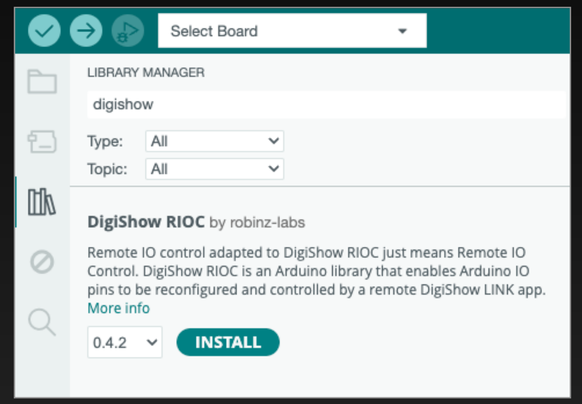


安装 RIOC 程序到 Arduino

首先，我们来准备一个 Arduino

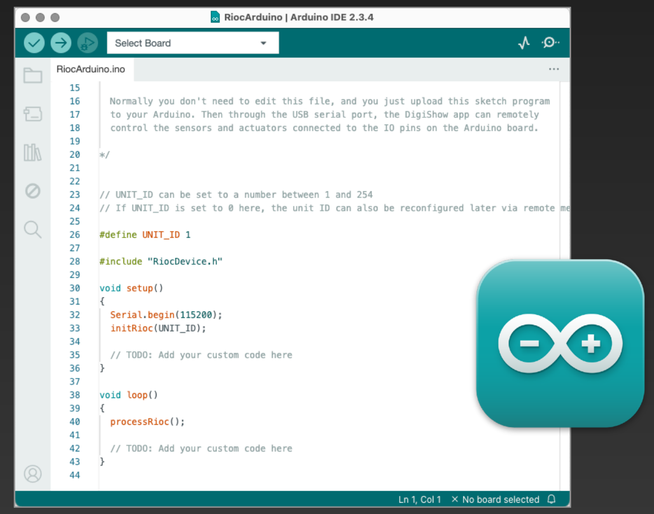
1. 我们建议你选择 Arduino UNO, MEGA 或 Nano (或以 Arduino 为核心的 PLC)，也可以选择各种 ESP32 架构的开发板。

然后，获得 RIOC 程序，将它写入你的 Arduino！



2. 在 Arduino IDE 中的 Library Manager (库管理) 中找到 DigiShow RIOC 库，并点击 INSTALL 完成安装。

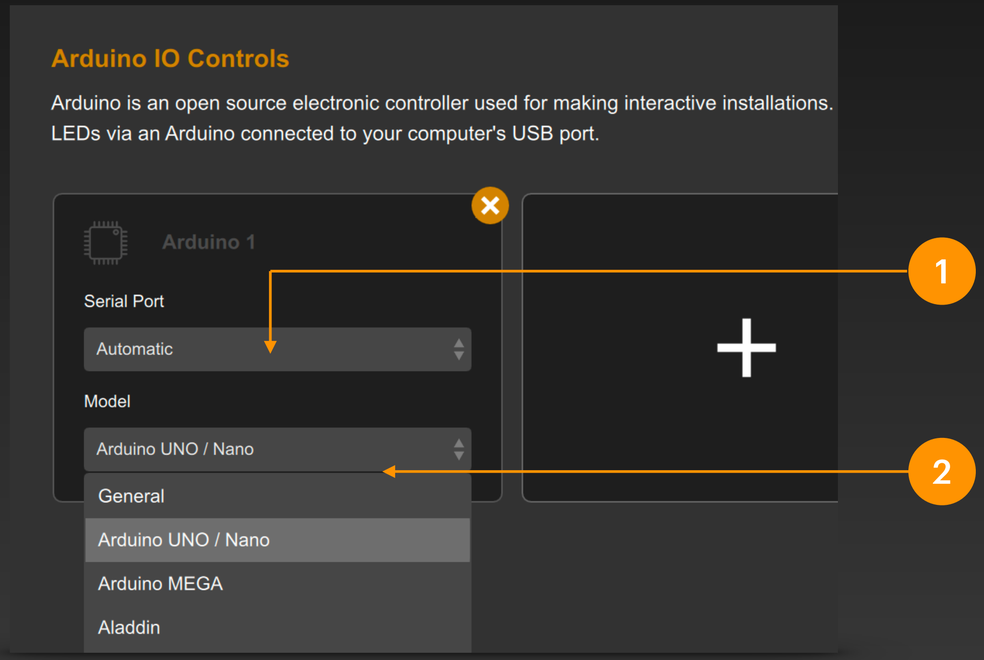
3. 然后在 Arduino IDE 菜单中选择 File > Examples > DigiShow RIOC > RiocArduino，将由此打开的 RIOC 程序写入到你的 Arduino 中。



(可在 www.arduino.cc 下载 Arduino IDE 和获得更多使用 Arduino 的详细指导)

Arduino 接口 - 配置

在把已写入了 RIOC 程序的 Arduino 接入电脑的 USB 端口后，DigiShow 就能通过它来控制所连接的传感器和执行器。用户可以在 Interface Manager 中的 Arduino 分栏中为当前工程添加 Arduino 接口。



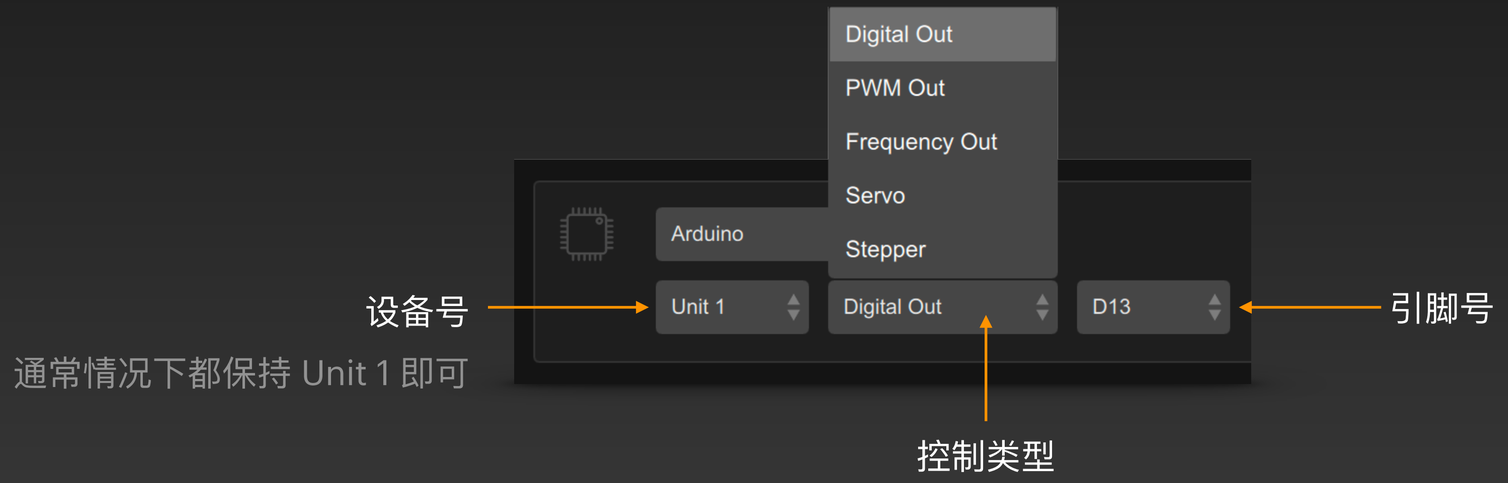
选择你接入的 Arduino 在电脑上占用的 Serial Port (串口)，如果所使用的 Arduino 是与原厂产品完全的兼容的 Arduino UNO 或 MEGA，只需要选择 Automatic (自动)。

选择你接入的 Arduino 的型号，如果是在列表中找不到的型号 (如 ESP32 Arduino)，请选择 General (通用)。

Arduino 接口 - 信号输入输出

在信号链接表中，把信号条的输出端设置为 Arduino，我们就可以在 DigiShow 中操控 Arduino 上的引脚用于输出，其类型可以是：数字输出、PWM输出、频率输出、舵机、步进电机等。

把信号条的输入端设置为 Arduino，我们就也可以在 DigiShow 中操控 Arduino 上的引脚用于输入，其类型可以是：数字输入、模拟输入、编码器输入等。

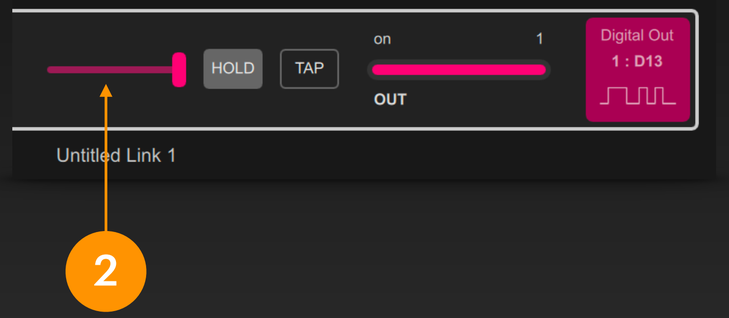


Arduino 接口 - 数字输出

我们可以在 DigiShow 信号条的输出端中操控 Arduino 上的引脚用于数字输出：

在信号条输出端中选择 Arduino 接口，选定一个引脚并设定它的类型为 Digital Out (数字输出)。

此例用于控制 D13 引脚所连接的板载LED灯的点亮与熄灭。

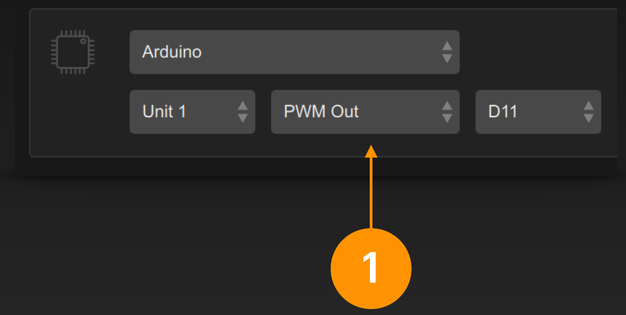


工程启动后，在信号条上拨动推杆就能改变 Digital Out 类型的引脚输出状态，它是一个开关量信号。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

Arduino 接口 - PWM 输出

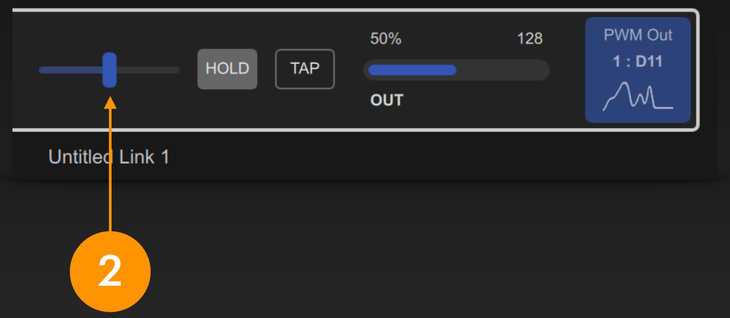
我们可以在 DigiShow 信号条的输出端中操控 Arduino 上的引脚用于PWM输出：



在信号条输出端中选择 Arduino 接口，选定一个引脚并设定它的类型为 PWM Out (PWM输出)。

PWM Out 常被用于 LED 调光，电机调速等。

PWM 输出对应 Arduino 编程中的 analogWrite()。

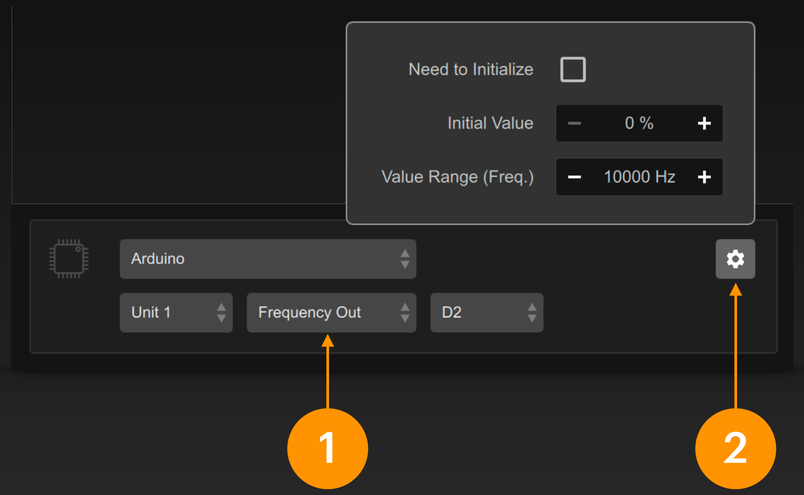


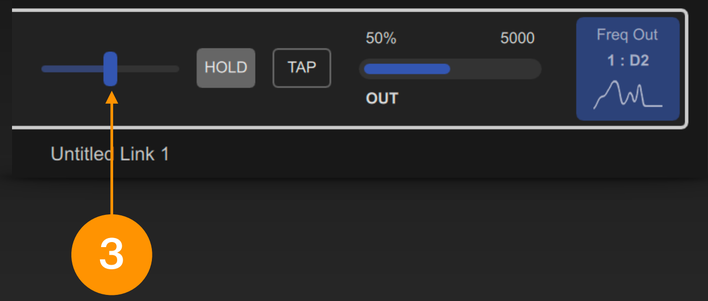
工程启动后，在信号条上移动推杆就能改变 PWM Out 类型引脚的输出值，它是一个模拟量信号，数值范围是 0 ~ 255。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

Arduino 接口 - 频率输出

我们可以在 DigiShow 信号条的输出端中操控 Arduino 上的引脚用于频率输出 (即震荡器输出):





在信号条输出端中选择 Arduino 接口，选定引脚并设定类型为 Frequency Out (频率输出)。

频率输出会产生频率可调的震荡方波，常用于电机驱动的调速或蜂鸣器发声等。

频率输出对应 Arduino 编程中的 tone()。

频率输出的取值范围默认是 0 ~ 1000Hz。也可点击 ⚙ 齿轮按钮，在弹出的选项设置面板内修改 Value Range (Freq.) 中的数值来修改频率调节的范围。

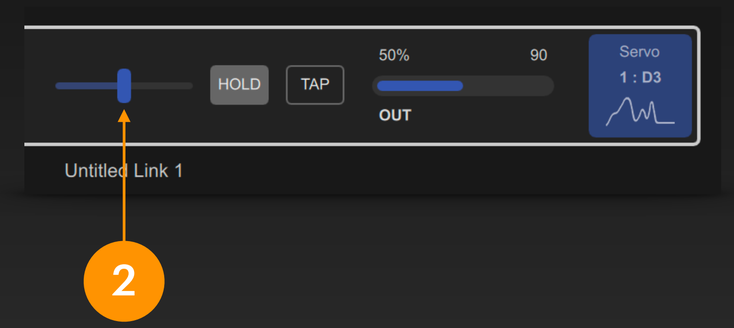
工程启动后，在信号条上移动推杆就能改变 Frequency Out 类型引脚的输出数值，它是一个模拟量信号，数值范围可由用户自定义。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

Arduino 接口 - 舵机

我们可以在 DigiShow 信号条的输出端中操控 Arduino 上的引脚用于舵机信号输出：

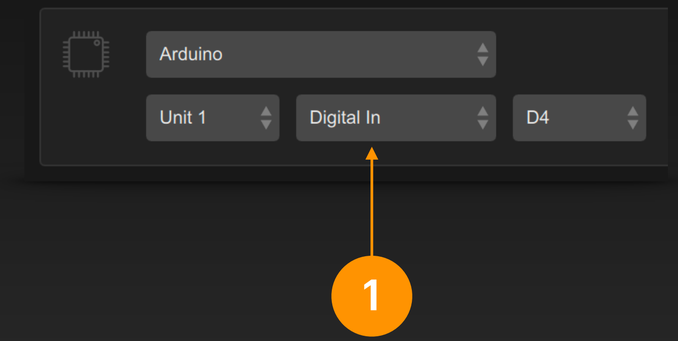
在信号条输出端中选择 Arduino 接口，选定一个引脚并设定它的类型为 Servo (舵机)。



工程启动后，在信号条上移动推杆就能改变 Servo 类型引脚的输出值，即调节舵机的角度位置，它是一个模拟量信号，数值范围是 0 ~ 180度。

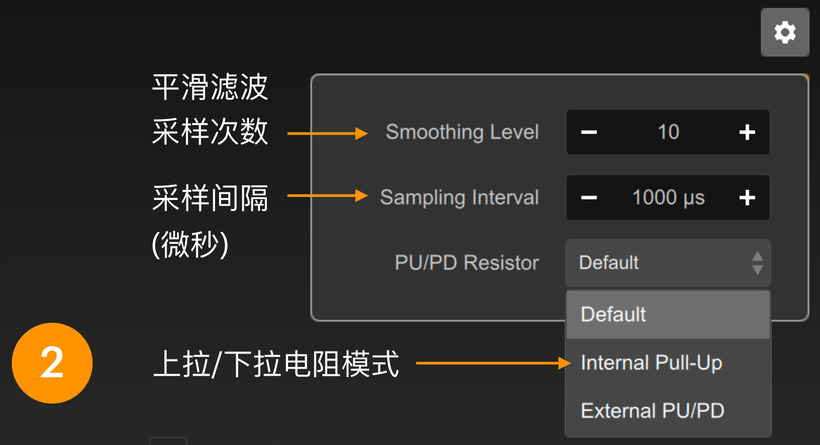
Arduino 接口 - 数字输入

我们可以在 DigiShow 信号条的输入端中操控 Arduino 上的引脚用于数字输入：



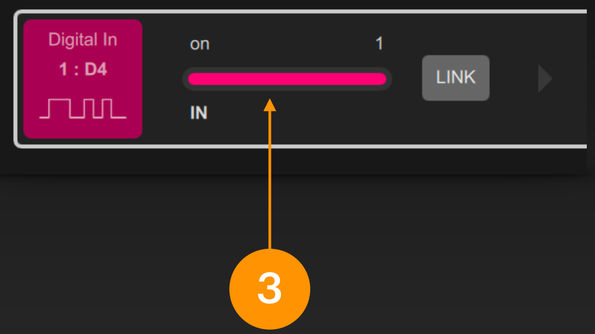
在信号条输入端中选择 Arduino 接口，选定一个引脚并设定它的类型为 Digital In (数字输入)。

数字输入通常用于连接开关、按钮或各种开关量传感器。



也可点击 齿轮按钮，在弹出的选项设置面板内修改此 Digital In 输入端的信号滤波和其他参数。

如上图例所示，若输入端状态发生变化并连续10次以上保持在新的状态，才会触发此输入信号的变化。

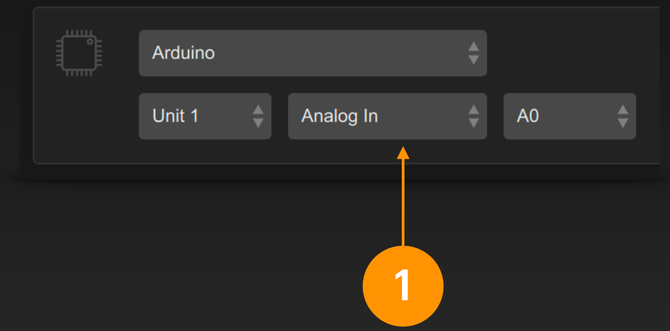


工程启动后，信号条输入端便会显示该引脚中的输入信号状态。它是个开关量信号。

此时，还可以为该信号条再选择一个输出端，以实现相应的信号映射。

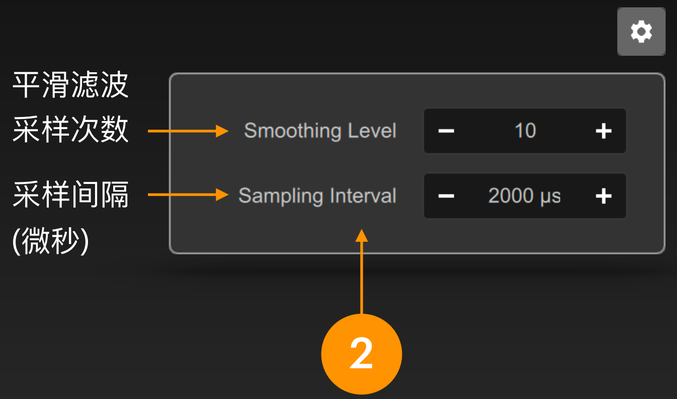
Arduino 接口 - 模拟输入

我们可以在 DigiShow 信号条的输入端中操控 Arduino 上的那些带有ADC的引脚用于模拟输入：



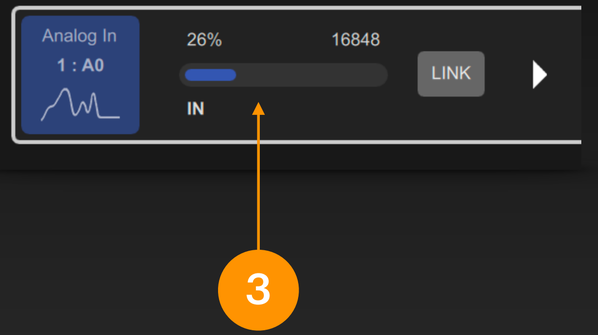
在信号条输入端中选择 Arduino 接口，选定一个引脚并设定它的类型为 Analog In (模拟输入)。

模拟输入通常用于连接电位器或各种模拟量传感器。



也可点击 齿轮按钮，在弹出的选项设置面板内修改此 Analog In 输入端的信号滤波参数。

如上图例所示，应用此滤波参数后，输入端原始信号每隔2000微秒采样一次，该输入信号的数值会是最近10次采样的平均值。



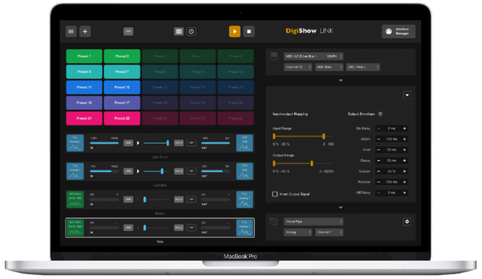
工程启动后，信号条输入端便会显示该引脚中的输入信号数值。它是个模拟量信号，它的数值范围是 0 ~ 65535

此时，还可以为该信号条再选择一个输出端，以实现相应的信号映射。

Modbus 接口

什么是 Modbus？

Modbus 是一种常用于工业场景中自动化设备间的通信协议，由施耐德电气最初在 1979 年制定。DigiShow 可以通过基于串行总线或电脑网络来通信的 Modbus 协议，与机器设备、机器人以及种类繁多的PLC控制器、IO模块、传感器、执行器等实现联机控制。



DigiShow 软件

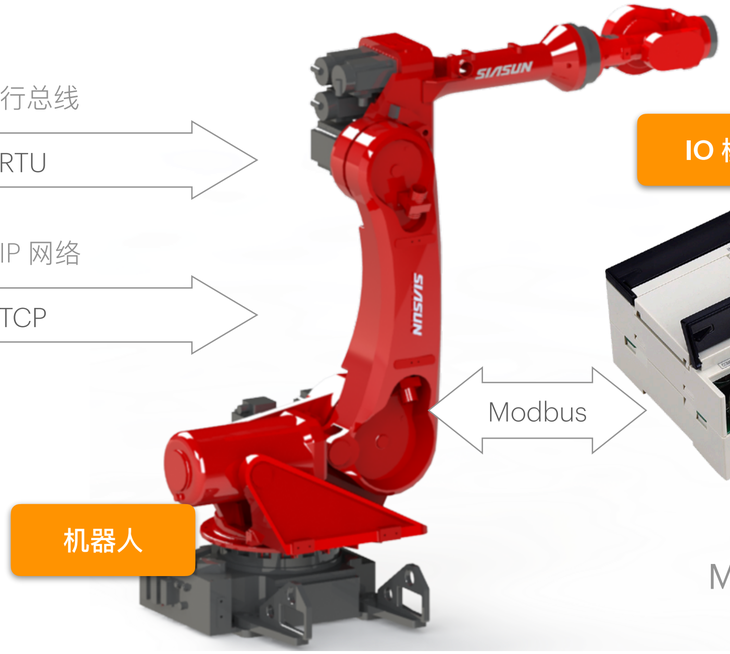
Modbus Master 主机电脑

基于 485 串行总线

Modbus RTU

或基于电脑 IP 网络

Modbus TCP



机器人

Modbus

IO 模块

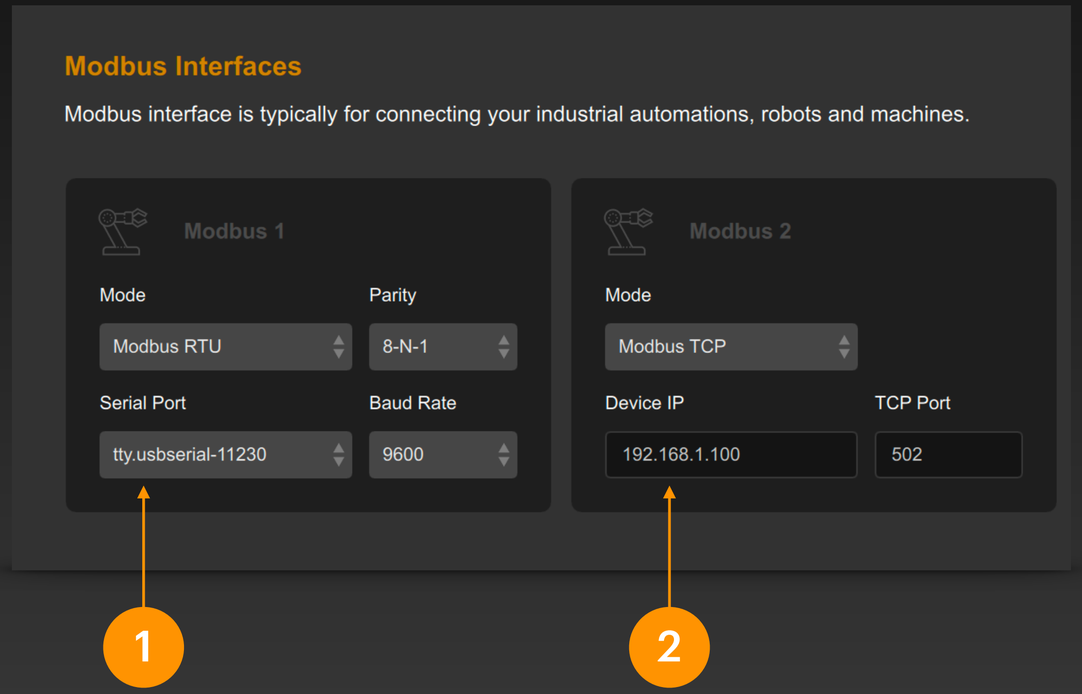
Modbus

传感器

Modbus Slave 从机设备

Modbus 接口 - 配置

DigiShow 支持基于串行总线和基于电脑网络的多种与 Modbus 从机设备通信的方式。用户可以在 Interface Manager 中的 Modbus 分栏中为当前工程添加 Modbus 接口。



1

如果你的 Modbus 设备采用串行总线通信方式，请选择 Mode (模式) 为 Modbus RTU，并设定设备接入电脑时占用的 Serial Port (串口) 和 Baud Rate (通信波特率)、Parity (校验位)等。

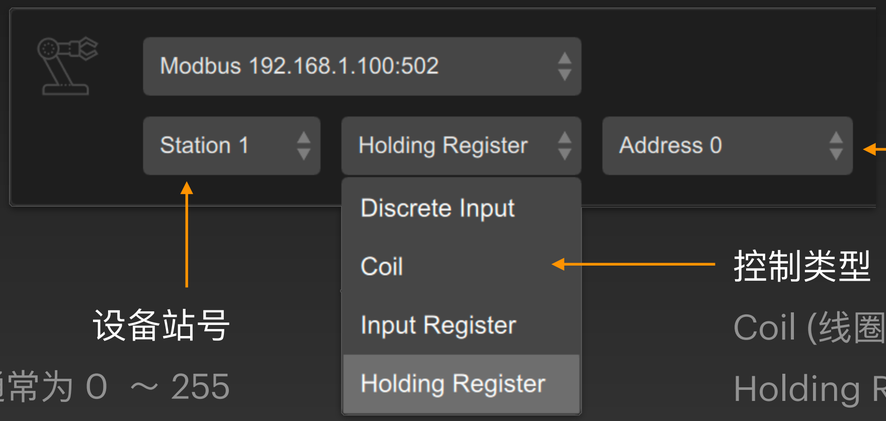
2

如果你的 Modbus 设备采用电脑网络通信方式，请选择 Mode (模式) 为 Modbus TCP，并设定设备在电脑网络中的 Device IP (设备 IP 地址) 和 TCP Port (TCP 通信端口号，通常使用502端口)。

Modbus 接口 - 信号输入输出

在信号链接表中，把信号条的输出端设置为 Modbus，我们就可以在 DigiShow 中操控 Modbus 设备上的输出通道，其类型可以是：Coil (线圈)、Holding Register (保持寄存器)。

把信号条的输入端设置为 Modbus，我们就可以在 DigiShow 中监测 Modbus 设备上的输入通道，其类型可以是：Discrete Input (离散量输入)、Coil (线圈)、Input Register (输入寄存器)、Holding Register (保持寄存器)。



通道地址

通常为 0 ～ 65535 中的一个数

控制类型

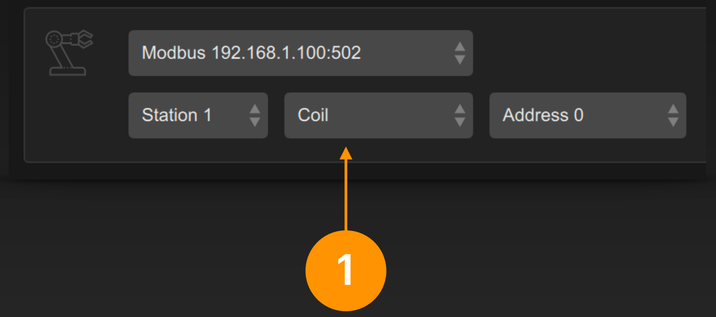
Coil (线圈) 常指设备中的继电器或开关量的控制点，
Holding Register (保存寄存器) 常指设备中的模拟量的控制点。
Discrete Input (离散量输入) 用于其他开关量输入，
Input Register (输入寄存器) 用于其他模拟量输入。

设备站号

通常为 0 ～ 255 中的一个数

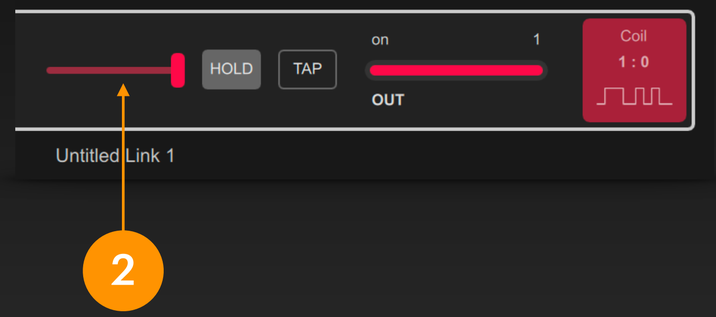
Modbus 接口 - 线圈输出

我们可以在 DigiShow 信号条的输出端中操控 Modbus 设备上的线圈输出：



在信号条输出端中选择 Modbus 接口，选定设备站号，设定控制类型为 Coil (线圈)，并选定该线圈的通道地址。

线圈输出常用来控制设备中的继电器或开关量控制点的开合。

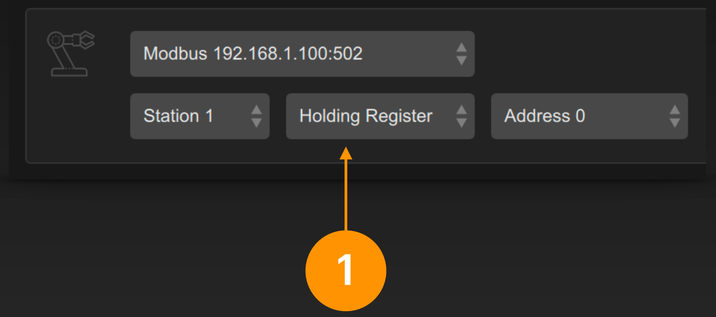


工程启动后，在信号条上拨动推杆就能改变线圈的输出状态，它是一个开关量信号。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

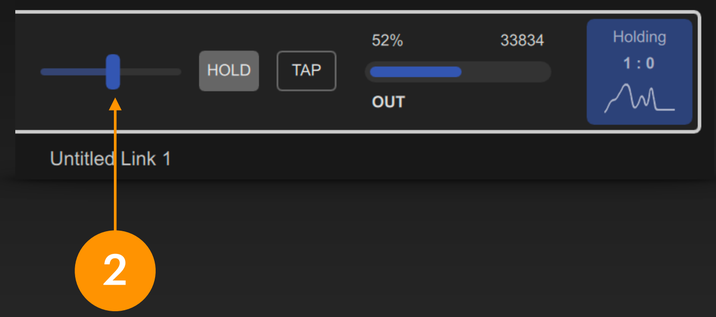
Modbus 接口 - 寄存器输出

我们可以在 DigiShow 信号条的输出端中操控 Modbus 设备上的寄存器输出：



在信号条输出端中选择 Modbus 接口，选定设备站号，设定控制类型为 Holding Register (保持寄存器)，并选定该寄存器的通道地址。

寄存器输出即修改寄存器中的数值，常用来控制设备中的模拟量控制点的状态。

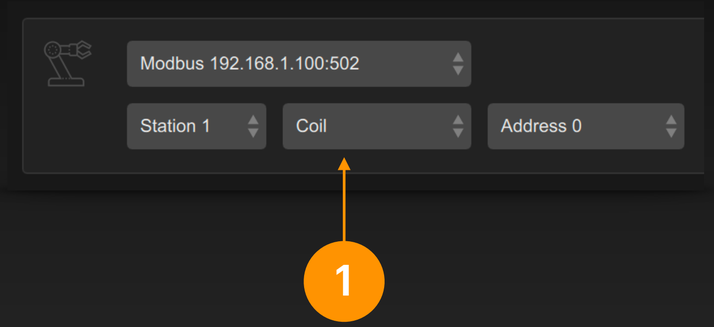


工程启动后，在信号条上移动推杆就能改变寄存器的状态数值，它是一个模拟量信号，数值范围是 0 ~ 65535。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

Modbus 接口 - 线圈输入

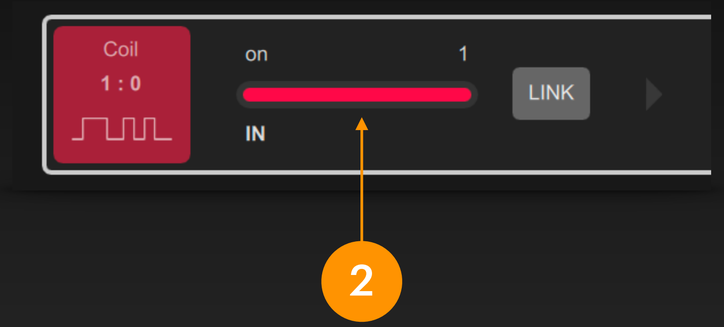
我们可以在 DigiShow 信号条的输入端中监测 Modbus 设备上的线圈输入：



在信号条输入端中选择 Modbus 接口，选定设备站号，设定控制类型为 Coil (线圈)，并选定该线圈的通道地址。

线圈输入常用来监测设备中的继电器或开关量控制点的开合状态。

另外还可以选择 Discrete Input (离散量输入) 类型，用于监测设备上的其他开关量输入。

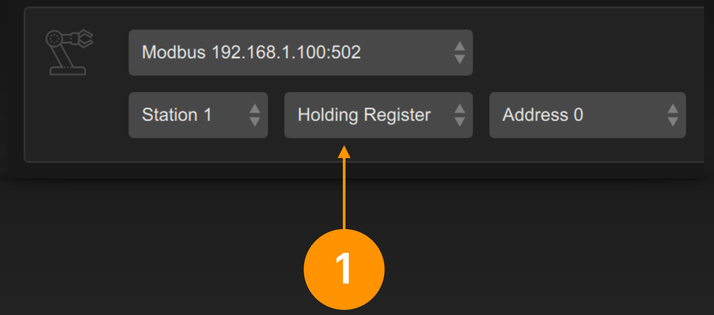


工程启动后，信号条输入端便会显示该线圈的当前状态。它是个开关量信号。

此时，还可以为该信号条再选择一个输出端，以实现相应的信号映射。

Modbus 接口 - 寄存器输入

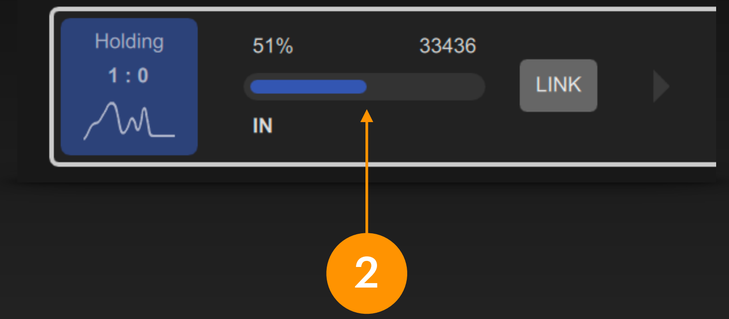
我们可以在 DigiShow 信号条的输入端中监测 Modbus 设备上的寄存器输入：



在信号条输入端中选择 Modbus 接口，选定设备站号，设定控制类型为 Holding Register (保持寄存器)，并选定该寄存器的通道地址。

寄存器输入常用来监测设备中的模拟量控制点的状态。

另外还可以选择 Input Register (输入寄存器) 类型，用于监测设备上的其他模拟量输入。



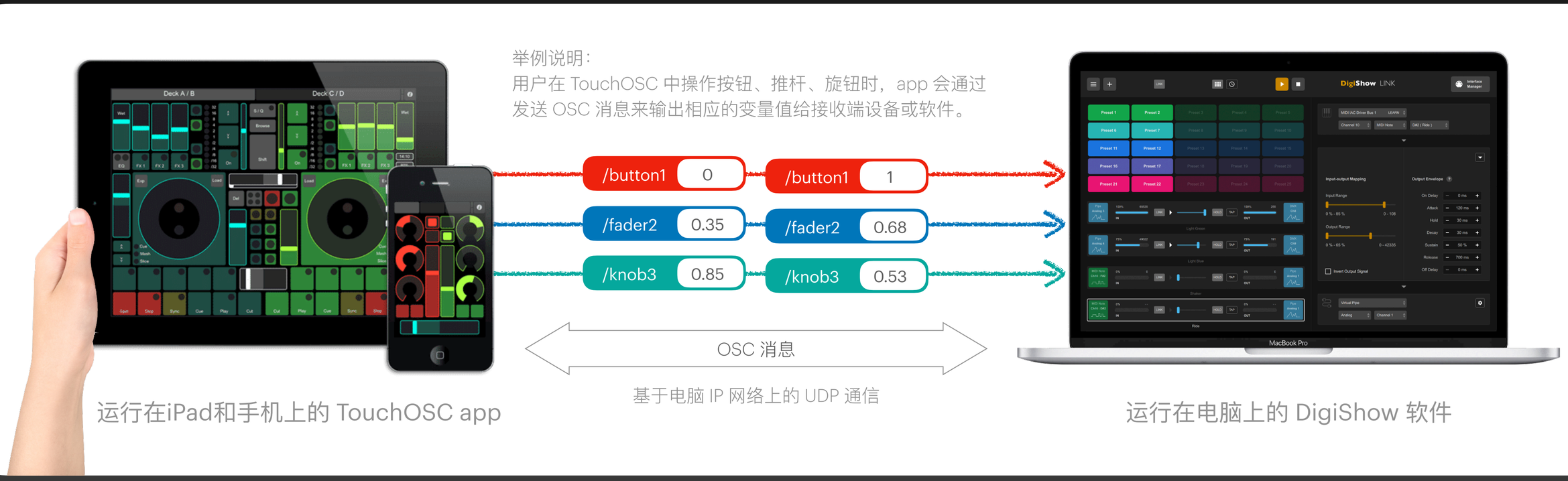
工程启动后，信号条输入端便会显示该寄存器中的当前状态数值。它是个模拟量信号，它的数值范围是 0 ~ 65535。

此时，还可以为该信号条再选择一个输出端，以实现相应的信号映射。

OSC 接口

什么是 OSC？

OSC (Open Sound Control) 是一种用于电脑、声音合成器以及其他多媒体设备之间通信的协议。它由加州大学伯克利分校的新音乐与音频技术中心（CNMAT）于1997年首次提出。OSC 广泛应用于音乐表达、VJ表演、机器人控制以及许多演艺相关的应用软件间的通信。



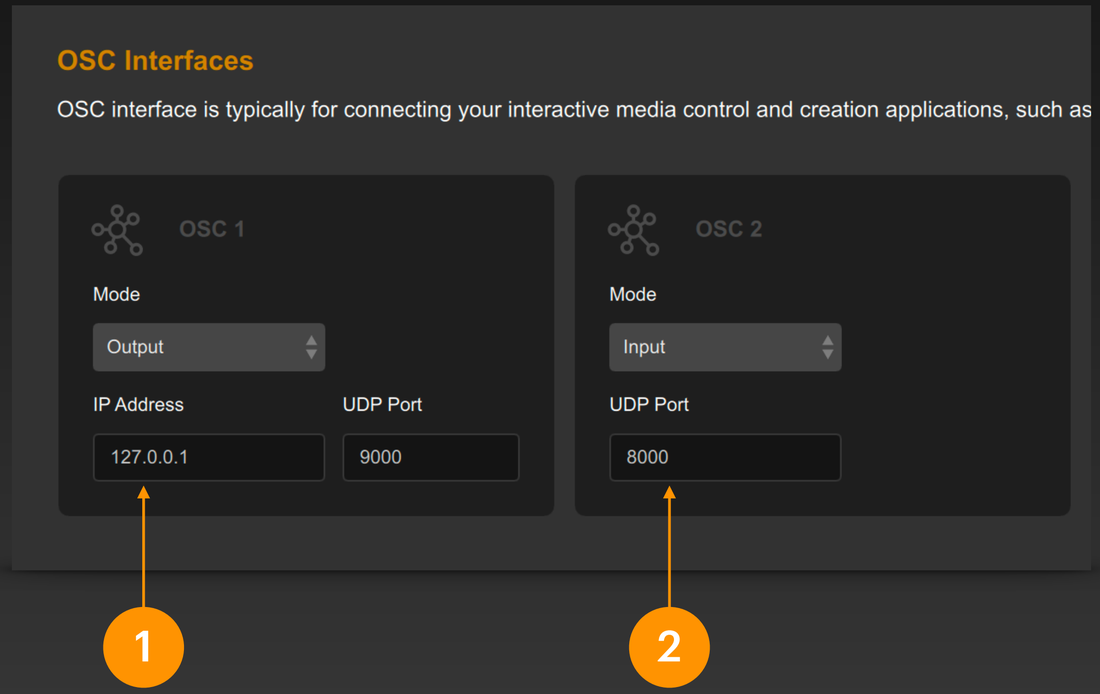
OSC 在应用间的通信

可以在电脑上的多个应用程序间通过 OSC 消息来互相传递信号变量，完成彼此协作。

当 DigiShow 和其他交互媒体创作软件（如 TouchDesigner、Unreal Engine）一起工作时，其他软件也就能立即获得访问 DigiShow 中各种输入输出信号的能力，进而轻松实现对传感器、执行器、机器人、灯光等一系列对象的操控，整个过程无需任何代码编程。

OSC 接口 - 配置

DigiShow 支持与其他应用程序进行双向 OSC 通信，完成信号变量的输入输出。用户可以在 Interface Manager 中的 OSC 分栏中为当前工程添加 OSC 接口。



1

如果你需要让 DigiShow 向其他应用程序发送 OSC 消息 (即向其他应用程序输出信号变量)，请选择 Mode (模式)为 Output (输出)，并设定接收 OSC 消息的应用程序所在电脑的 IP 地址和接收端的 UDP 端口号。

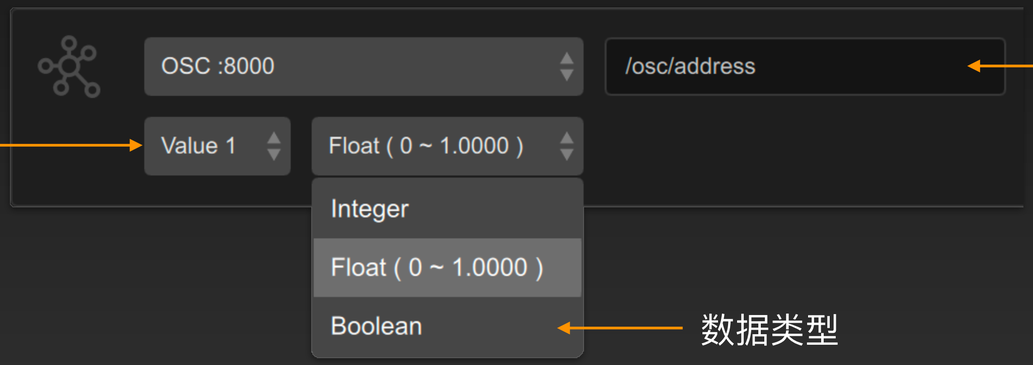
2

如果你需要让 DigiShow 从其他应用程序接收 OSC 消息 (即从其他应用程序输入信号变量)，请选择 Mode (模式)为 Input (输入)，并设定接收 OSC 消息所用的 UDP 端口号。

OSC 接口 - 信号输入输出

在信号链接表中，把信号条的输出端设置为OSC，我们就可以向特定的地址发送OSC消息，将特定的信号变量传输给特定的应用程序。

把信号条的输入端设置为OSC，我们就可以监听在特定端口上所接收到的OSC消息，获取由特定应用程序传输来的特定信号变量。



数据序号

在一个OSC地址中通常可以只包含单个数据，也可以包含多个数据。当地址中包含了多个数据时，就需标记这个数据序号来确定它的存放位置。而在只包含单个数据的地址中，此数据序号保持 1 即可。

OSC 地址

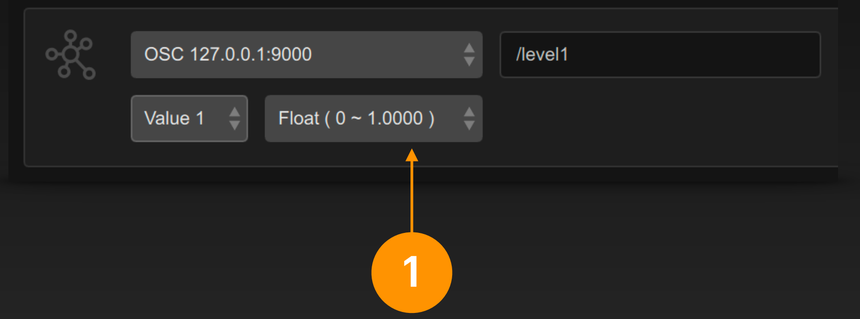
OSC 地址通常是一个类似文件路径的字符串，它可以被看作是 OSC 消息中被传输的信号变量的名称。

数据类型

DigiShow 支持 OSC 消息中的部分数据类型，如：Float (浮点数)，但它的数值范围必须是 0 ～ 1；Integer (整数)，其数值范围是 0 ～ 某个可以由用户设定的最大整数；Boolean (布尔值)，其数值可以是 True 或 False。

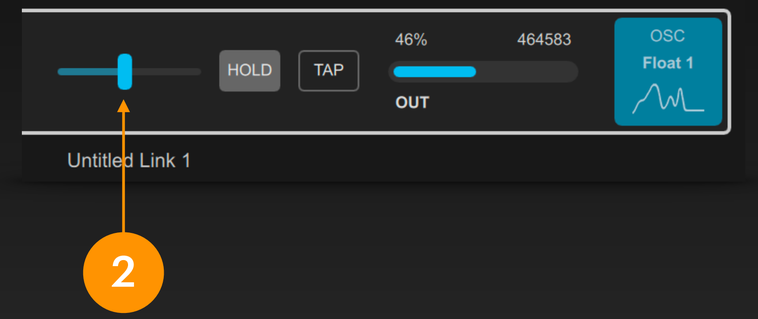
OSC 接口 - 浮点数输出

我们可以在 DigiShow 信号条的输出端中操控 OSC 接口来发送带有浮点数信号变量的 OSC 消息：



在信号条输出端中选择 OSC 接口，设定 OSC 地址和数据序号，选择数据类型为 Float (0~1.0000)(浮点数)。

范围在0～1之间的浮点数类型的数据经常会出现在一些OSC会话中，如 TouchOSC 上的推杆、旋钮数值。

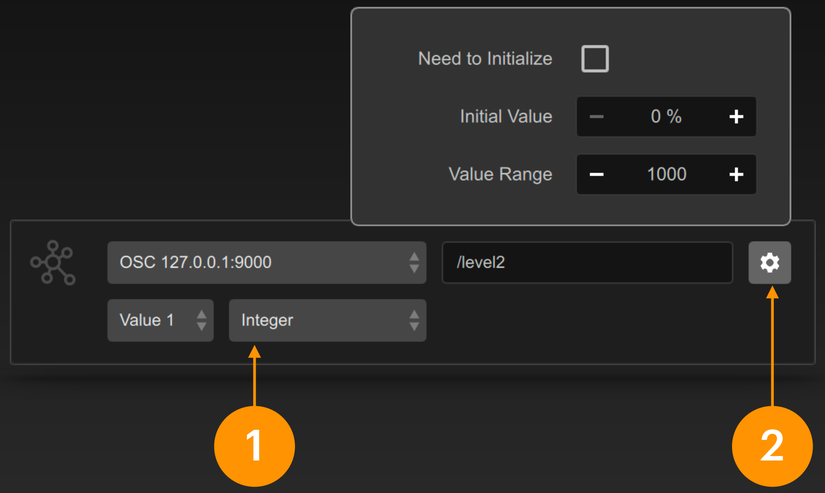


工程启动后，在信号条上移动推杆就能改变信号变量的数值并发送相应的 OSC 消息。在 DigiShow 中它是一个模拟量信号，数值范围是 0 ~ 1000000，而在 OSC 消息中这个整数数值会被自动对应为一个 0 ～ 1.000000 的浮点数。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

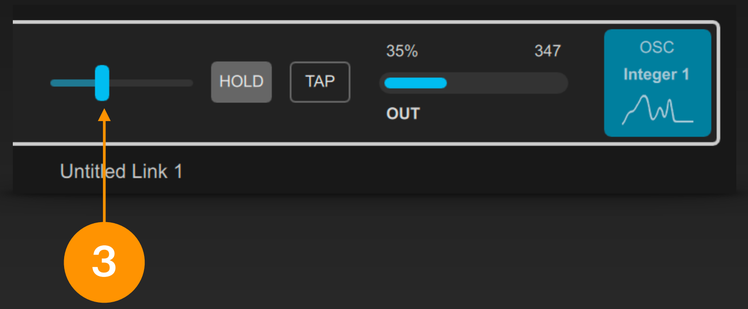
OSC 接口 - 整数输出

我们可以在 DigiShow 信号条的输出端中操控 OSC 接口来发送带有整数信号变量的 OSC 消息：



在信号条输出端中选择 OSC 接口，设定 OSC 地址和数据序号，选择数据类型为 Integer (整数)。

此整数的取值范围可由用户自行设定。请点击 ⚙ 齿轮按钮，在弹出的选项设置面板中修改 Value Range (数值范围) 中的数值。

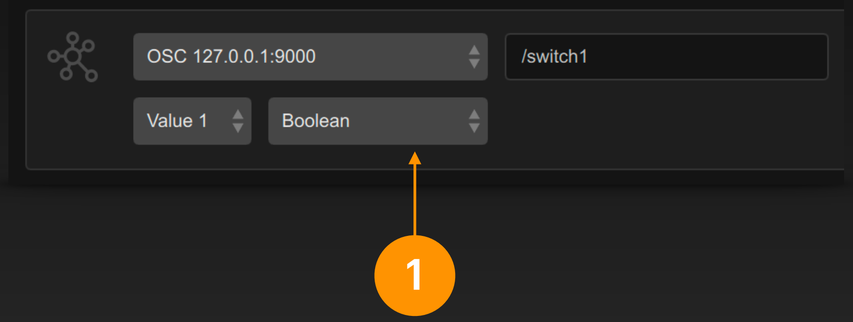


工程启动后，在信号条上移动推杆就能改变信号变量的数值并发送相应的 OSC 消息。它是一个模拟量信号，数值范围是 0 ～ 某个由用户设定的最大整数。

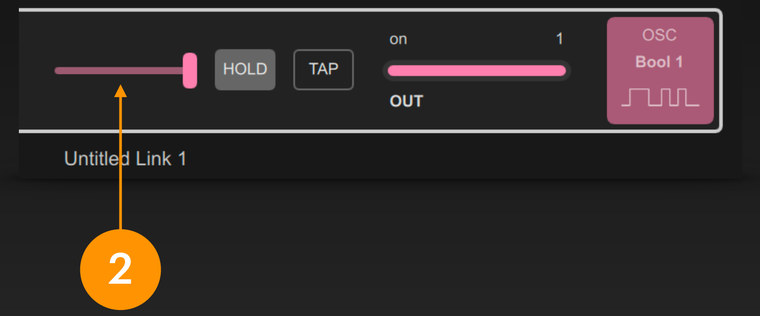
此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

OSC 接口 - 布尔值输出

我们可以在 DigiShow 信号条的输出端中操控 OSC 接口来发送带有布尔值信号变量的 OSC 消息：



在信号条输出端中选择 OSC 接口，设定 OSC 地址和数据序号，选择数据类型为 Boolean (布尔值)。

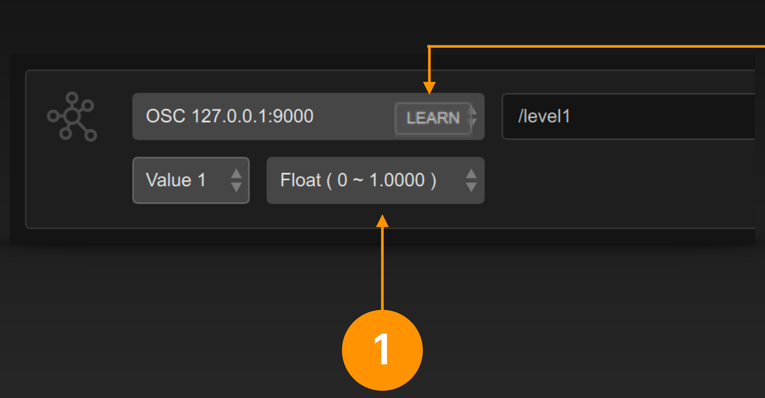


工程启动后，在信号条上拨动推杆就能改变信号变量的状态并发送相应的 OSC 消息。在 DigiShow 中它是一个开关量信号，在 OSC 消息中对应 True 或 False 数据类型。

此时，还可以为该信号条再选择一个输入端，以实现相应的信号映射。

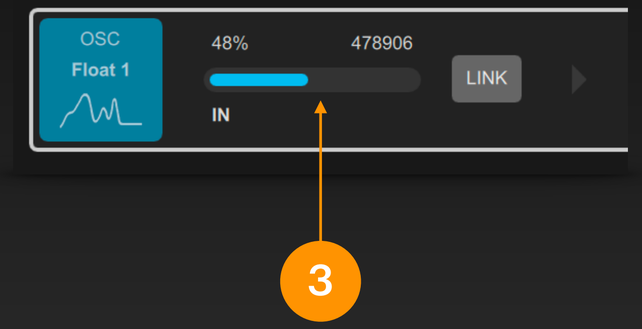
OSC 接口 - 信号输入

把信号条的输入端设置为 OSC，就可以监听在特定端口上所接收到的 OSC 消息，获取由特定应用程序传输来的特定信号变量。



在信号条输入端中选择 OSC 接口，设定 OSC 地址和数据序号，选择数据类型。应用该设置后，此输入端便会接收到 OSC 消息中符合设定要求的信号变量了。

我们也可以主动学习外部软件发出的 OSC 消息。请首先启动工程，此处便会出现 LEARN (学习) 按钮。点击此按钮，并在外部软件 (如 TouchOSC) 中进行操作，其对应发送的 OSC 消息便会被 DigiShow 自动识别，并刷新信号条输入端的参数。

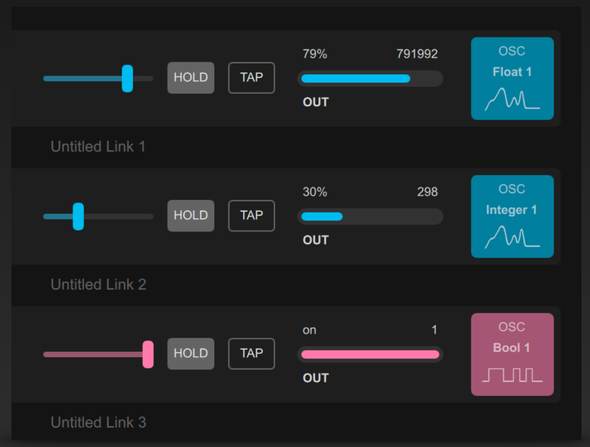


DigiShow 一旦工程启动，信号条输入端便会显示接收到 OSC 消息中符合设定要求的信号变量的数值。

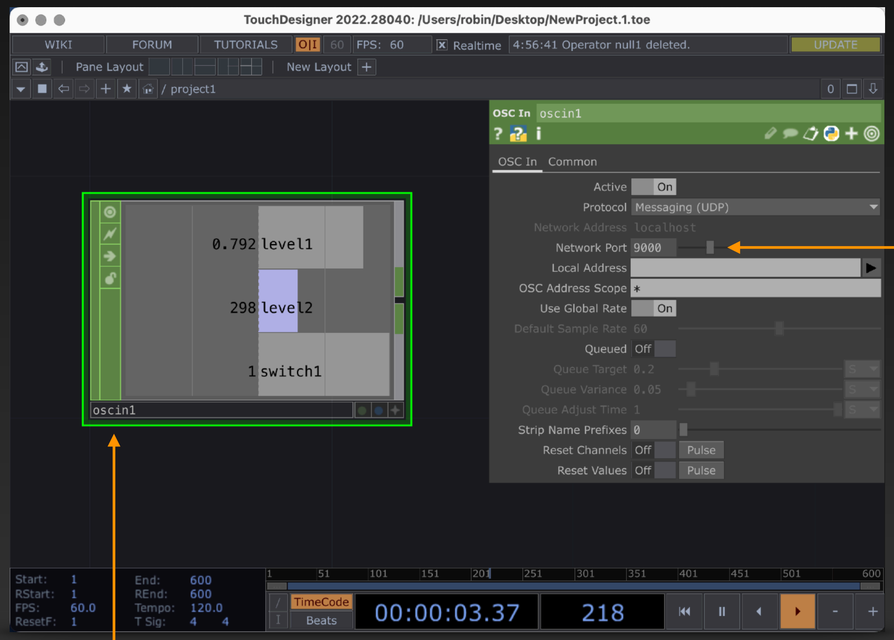
此时，还可以为该信号条再选择一个输出端，以实现相应的信号映射。

OSC 接口 - 输出至 TouchDesigner

可以在 TouchDesigner 软件中轻松接收到由 DigiShow 输出的 OSC 消息，使 TouchDesigner 中的 CHOP 数据与 DigiShow 中的信号变量完成同步更新。



OSC 消息



2

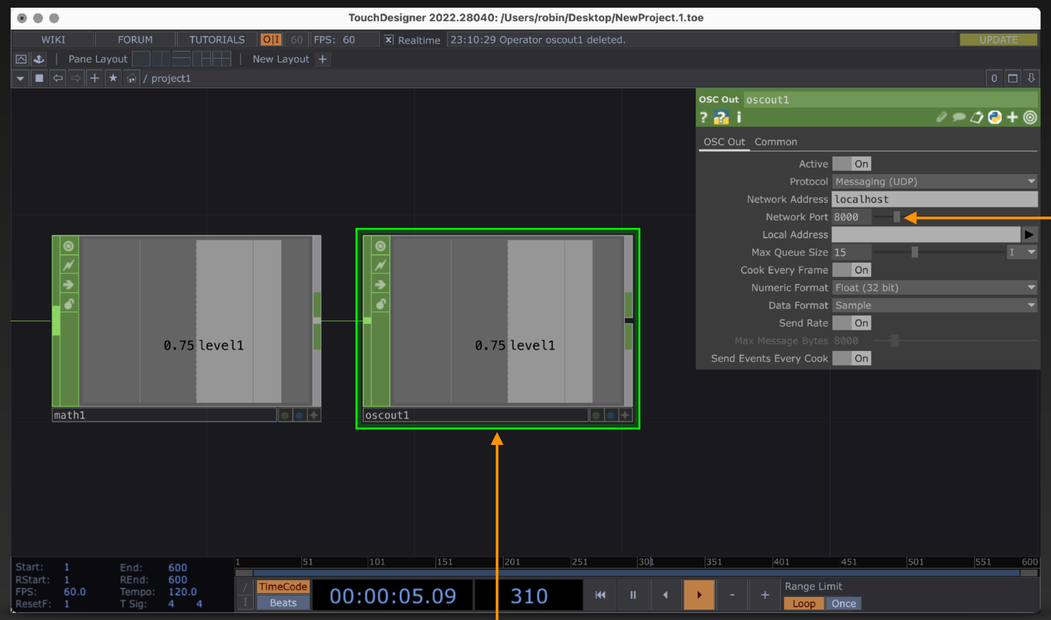
在 OSC In CHOP 的属性面板中设置 Network Port (网络端口)，使此端口号与 DigiShow OSC 输出接口中的 UDP 端口设置保持一致。

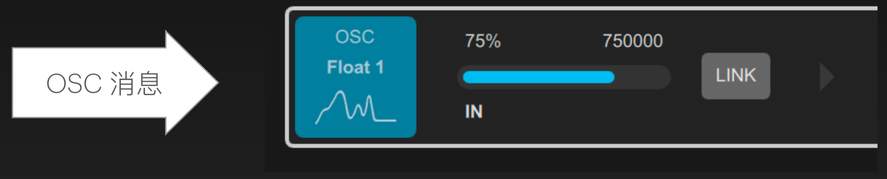
1

在 TouchDesigner 网格图中加入一个 OSC In CHOP

OSC 接口 - 由 TouchDesigner 输入

可以在 DigiShow 中轻松接收到由 TouchDesigner 输出的 OSC 消息，使 DigiShow 中的信号变量与 TouchDesigner 中的 CHOP 数据完成同步更新。





2

在 OSC Out CHOP 的属性面板中设置 Network Port (网络端口)，使此端口号与 DigiShow OSC 输入接口中的 UDP 端口设置保持一致。Numeric Format 设为 Float (32 bit) 或 Int (32 bit)，Data Format 设为 Sample。

1

在 TouchDesigner 网格图中加入一个 OSC Out CHOP，为其设置一个输入数据源。源数据可以是范围在是 0 ～ 1之间的浮点数，或是整数。

本课小结

- 了解 Arduino 和 IO 控制的相关知识
- 制作一片带有 RIOC 程序的 Arduino，为在 DigiShow 中使用做好准备
- 学会在 DigiShow 中使用 Arduino 接口完成信号输入输出的方法
- 了解 Modbus 概念和使用方法
- 了解 OSC 的概念和使用，学会与 TouchDesigner 一起工作的方法