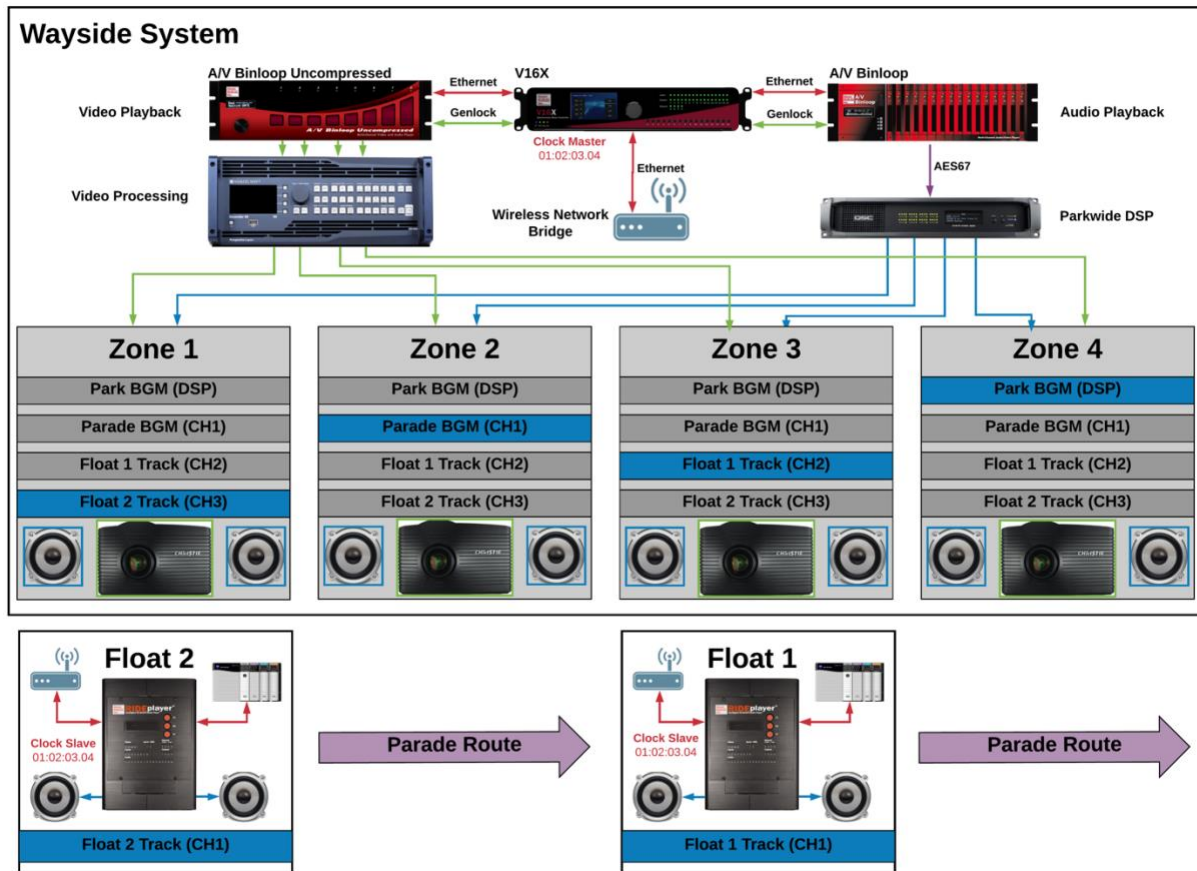


# Parade Application



## Overview

Everyone loves a parade, right? With the beautiful floats, live performers, and engaging soundtrack, they are often a star attraction of a theme park. Since they operate within the common areas of the park and require widely distributed audio, parades can be one of the most technically challenging applications to pull off. Thankfully, V16X and RidePlayer offer a suite of features like advanced network synchronization, integrated show control, and multi-track polyphonic audio playback that make even the toughest parades simple to implement.

# How It's Done

## Introduction

Parades are a high-profile attraction that take place within the heart of a theme park. They involve multiple floats, each with their own unique theme, show, and soundtrack. The floats travel together, so the parade systems and content must be designed to synchronize precisely with one another so that guests have a seamless experience. Also, many guests gather to watch the parade, so we have to consider how to distribute the parade show beyond the proximity of the float itself so that everyone can have a great experience.

With thousands of people gathered to enjoy a show that revolves around a musical soundtrack, audio distribution is a key factor in parade system design. Part of the solution is to source audio from the floats themselves so that guests are drawn to the nearest float. Unfortunately, many guests will not be very close to the floats which creates an interesting dilemma; how do we give everyone a good audio experience? While it would certainly help more people to hear the show, cranking up the audio levels on the floats is not an option. Not only would we blow out the eardrums of everyone close to the floats, the audio from the floats would also bleed into one another. The only good solution to this problem is to incorporate the park-wide audio system (a.k.a. The Wayside) into the show so that audio can be distributed safely to the crowd.



What we're left with is many floats that need to synchronize with sub-millisecond precision to each other as well as the park-wide audio system. We also need to track the position of the floats so that we can route each of their audio tracks to only the wayside speakers that surround them. In case that's not challenging enough, this all has to be done from cramped battery-powered platforms with no hard-wired connection to the wayside. As Jim Carstensen would say, "What could possibly go wrong?"

Since this is a common application for theme parks, many different systems have been built to tackle these challenges. Unfortunately, they often involve a complex integration of many hardware components and highly-customized software to make those components work together. The end result is a system that is extremely difficult to maintain that very few people know how to modify.

This is where products like V16X, RidePlayer and RideAmp change the game. These systems consolidate industry-standard show control, advanced synchronization, DSP, amplification, and powerful media playback capabilities to provide an off-the-shelf solution that makes parades a snap to integrate. This guide explains how to use the V16X and RidePlayer to coordinate synchronization of show and audio between the parade floats and the park-wide systems.

## Integrating the System

As you can see from the application diagram, we're going to implement a parade that incorporates multiple floats and multiple zones of audio and video playback on the wayside. The idea is that the wayside system will coordinate the synchronization of audio between the floats and the wayside. The floats themselves will track their current position using wheel encoders. As the float approaches a zone, it relays its location to the wayside system so that it can route the wayside audio and video tracks associated with that float speakers and projectors in that zone. The result is perfectly synchronized audio for the entire parade system where each float's audio track is reinforced by the park-wide sound system as it travels along the parade route.

## System Components

Let's look at the gear we have designed into this system and its role in implementing the parade.

### RidePlayer – Onboard Synchronous Audio Player and Show Controller

This product is designed specifically to be installed onboard vehicles like dark rides, coasters, and parade floats. Some of the key features include 16 channels of polyphonic audio playback as well as onboard show control. Other useful features like DSP, amplification, network audio, voltage monitoring, and GPS integration are all rolled into a nice compact, rugged, and energy efficient package to make this product the ultimate onboard audio and show control solution.

For this parade system we are using a RidePlayer units on each parade float to manage onboard playback and position tracking. It will coordinate with the wayside V16X to ensure all onboard show elements, including audio, will be synchronized perfectly with the Wayside systems. The unique design of these products ensures that the level of synchronization is extremely precise, even with applications like parades that only have a wireless connection between floats and wayside systems.



## RideAmp-350Q – High-powered Onboard Amplifier

The built-in amplifiers of RidePlayer are powerful enough to drive the main speakers on most vehicles, but parade floats tend to require larger more powerful speakers that greatly benefit from high-power amplification.

To ensure these speakers are driven properly, RideAmp-350Q delivers up to 4 channels of amplification, each capable of up to 350W of output power. This product is also designed to operate from a DC power source, making it ideal for onboard audio applications such as this. To ensure a nice clean audio signal, RideAmp leverages the digital Dante/AES67 network audio interface of RidePlayer. For vehicle configurations that require multiple amplifiers, this connection can be easily daisy-chained without the need for additional network switch hardware.



## Programmable Logic Controllers (PLCs)

PLCs are specialized control devices that are typically used for industrial and safety-critical applications. They are commonly used within parade floats to implement a safe drive system for the float. With the right sensors, they are also quite handy with tracking the state and position of the float.



For this application, we're relying on the PLC to keep track of the float position. This would typically be done in conjunction with devices like wheel encoders, RFID tags, or other sensors used by the PLC to track the position of the float. Since RidePlayer utilizes the powerful Alcorn McBride show control core, it has the ability to communicate directly with the PLC and monitor tags to retrieve position data.

### V16X – Synchronous Show Controller

This show control device manages the wayside systems and coordinates with the RidePlayer units onboard the floats to ensure that both systems synchronize perfectly with one another. In this application, the V16X will distribute a master clock to all RidePlayers using NTP to ensure everyone is triggering based on the same reference. It will also retrieve position information from each float's RidePlayer and leverage the parkwide DSP and video processor to route the wayside audio and video to the speakers and projection surfaces located near the appropriate float.



### A/V Binloop Uncompressed – Multi-channel Synchronous Video Player

The purpose of this unit is to provide multiple channels of uncompressed video playback for the projection surfaces along the parade route. Content is stored on solid-state media drives as uncompressed Targa sequences. This product connects to a display processor via 3G-SDI connections so that the video streams can be dynamically re-routed to various projectors along the parade route. This device also connects to the V16X via Ethernet so that video clips can be played at the appropriate time and synchronized with other devices in the system. Genlock is provided from the V16X as well to ensure that all video playback is perfectly synchronized with the parade timeline.





### A/V Binloop – Multi-channel Synchronous Audio Player

This device provides multiple channels of synchronous audio playback to source the wayside portion of the parade float audio tracks. Content is stored on CompactFlash drives as uncompressed WAV files. This connects to the parkwide DSP system using AES67 network audio so that the parkwide audio system can dynamically route the audio sources to different speakers along the parade route. This device also connects to the V16X via Ethernet so that audio clips can be played at the appropriate time and synchronized with other devices in the system. Genlock is provided from the V16X as well to ensure that all audio playback is perfectly synchronized with the parade timeline.



### Park-wide DSP and Amplifiers

Park-wide audio systems typically use one or more DSP platforms to route and distribute audio within the park. These would be teamed up with high-powered amplifiers that feed the many speakers that are installed throughout the park area.



Since a parade travels through the park, it is absolutely essential for the wayside parade system to integrate with the park-wide DSP. In this application, the A/V Binloop will feed all of the synchronous audio tracks to this DSP via AES67 network audio. The V16X will dispatch commands to the DSP to change audio routing based upon the position of the parade floats. In other words, the wayside parade system basically hijacks the park-wide audio system while a parade is in progress. Once the parade passes through, the park-wide audio system reverts back to normal operation.

## Video Processor

For this application, a video processor is used to route float-specific video content from the A/V Binloop Uncompressed to projectors near the associated float. Just like the park-wide DSP, the V16X will command the video processor to transition between playback sources as the floats move along the parade route.

## Network Infrastructure

Although it is not shown in the system diagram, it is implied that a system like this would consist of network switches, routers, and wireless bridges as needed. A reliable wireless connection between the wayside system and the parade floats is essential to consistent operation. WIFI may not always be the best tool for this, but there are several wireless networking technologies that are well-suited for reliable low-latency networking.

## Implementing Control

Properly implementing the show control interface between the wayside V16X and float RidePlayer systems is critical to synchronous audio and float position tracking. Both of these devices are configured and programmed using our WinScript Live software.

### Show Control Programming

We need to create two show control scripts to properly implement this parade application; one for the wayside V16X and another for the float RidePlayers. The float script will be responsible for monitoring the float's location along the parade route and coordinating with the wayside system to route audio and video to the zone area where the float is located. The wayside script will be responsible for synchronously triggering the parade loop on all devices and routing wayside A/V systems as needed.

These two scripts have been included with this application note and are appropriately named:

- **Parade – Wayside.WSL**
- **Parade – Float.WSL**

Although Alcorn McBride goes through great effort to make this programming significantly easier than many other control systems, there is a learning curve to using WinScript Live and the RidePlayer. If you're looking to learn more about using this interface, Alcorn McBride offers free training in the form of interactive in-person classes and online courses.

### Devices - Wayside

A great first step would be to configure the list of devices that will be connected to the wayside V16X. This involves browsing the comprehensive library of devices in the WinScript library by manufacturer and model number, choosing the device, and then configuring the physical connection to the device (i.e. Ethernet, Serial, etc.).

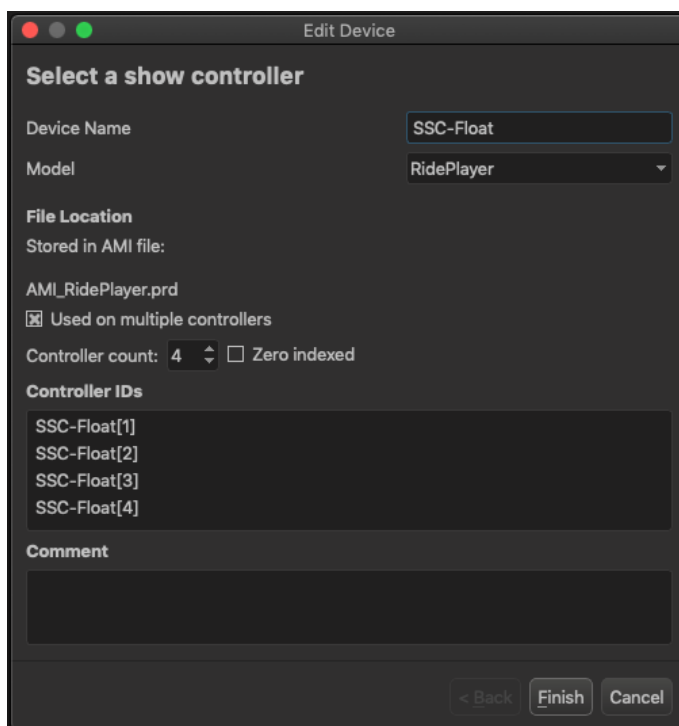
The V16X must communicate with the Binloops, park-wide DSP, video processor, and all of the float RidePlayers:

| #  | D                        | Name            | Device Type                                     | Protocol                                | Connection | Details                   |
|----|--------------------------|-----------------|---|---|------------|---------------------------|
| 1  | <input type="checkbox"/> | SCS-Wayside     | Alcorn McBride, Inc. - V16X                     | local                                   |            |                           |
| 2  | <input type="checkbox"/> | DSP             | QSC - Q-SYS Core                                | Q-SYS External Control Protocol (ASCII) | ethernet A | TCP: 192.168.1.160, 1702  |
| 3  | <input type="checkbox"/> | Binloop-Audio   | Alcorn McBride, Inc. - A/V Binloop              | ASCII                                   | ethernet A | UDP: 192.168.1.161, 2638  |
| 4  | <input type="checkbox"/> | Processor-Video | Analog Way - LiveCore                           | Analog_Way_LiveCore                     | ethernet A | TCP: 192.168.1.170, 10600 |
| 5  | <input type="checkbox"/> | Binloop-Video   | Alcorn McBride, Inc. - A/V Binloop Uncompressed | ASCII                                   | ethernet A | UDP: 192.168.1.171, 2638  |
| 6  | <input type="checkbox"/> | SSC-Float1      | Alcorn McBride, Inc. - RidePlayer               | ASCII remote                            | ethernet A | UDP: 192.168.1.204, 2640  |
| 7  | <input type="checkbox"/> | SSC-Float2      | Alcorn McBride, Inc. - RidePlayer               | ASCII remote                            | ethernet A | UDP: 192.168.1.206, 2640  |
| 8  | <input type="checkbox"/> | SSC-Float3      | Alcorn McBride, Inc. - RidePlayer               | ASCII remote                            | ethernet A | UDP: 192.168.1.208, 2640  |
| 9  | <input type="checkbox"/> | SSC-Float4      | Alcorn McBride, Inc. - RidePlayer               | ASCII remote                            | ethernet A | UDP: 192.168.1.210, 2640  |
| 10 | <input type="checkbox"/> |                 |   |   |            |                           |



## Devices - Floats

We will configure the RidePlayer script so that it can be easily deployed to multiple RidePlayers installed on multiple floats. This configuration method uses a single device name (i.e. “SSC-Float”) that can be referenced throughout the script. When it comes time to deploy this script to multiple vehicles, individual RidePlayers will be uniquely identified using this name combined with an array index (i.e. “SSC-Float[1]”, “SSC-Float[2]”, etc.). These identifiers make it clear which float systems are operating and whether they are up to date with the most recent script and media content. From the connection screen, you can also choose to deploy updated content to individual units one at a time, or to multiple vehicles simultaneously. This approach avoids a tremendous amount of unnecessary duplication by managing large fleets of vehicles with a single script.



Next, we must configure the device list so that the RidePlayers can communicate back to the wayside V16X as well as the on-board PLCs of each float:

| # | D                        | Name        | Device Type                       | Protocol        | Connection       | Details                  |
|---|--------------------------|-------------|-----------------------------------|-----------------|------------------|--------------------------|
| 1 | <input type="checkbox"/> | SSC-Float   | Alcorn McBride, Inc. - RidePlayer | local           |                  | Used on 4 controllers    |
| 2 | <input type="checkbox"/> | SCS-Wayside | Alcorn McBride, Inc. - V16X       | ASCII remote    | ethernet Prim... | UDP: 192.168.1.201, 2640 |
| 3 | <input type="checkbox"/> | PLC         | Allen-Bradley - ControlLogix      | CIP Ethernet IP | ethernet Prim... | (multiple)               |

What’s unique about this is that each float system is likely to have a PLC with its own unique IP address. It’s our job to identify which PLC is associated with each specific RidePlayer unit:

Edit Device

Set up the connection

Controller

Connection Type

ethernet

Show Controller Port

Primary

Protocol

CIP Ethernet IP

Source Port (0 = auto)

0

Find Device

Device

Ethernet Type

ethip\_client

☒ Use different IP addresses for multiple show controllers

| D                        | Controller ID | Device IP     | Port  |
|--------------------------|---------------|---------------|-------|
| <input type="checkbox"/> | SSC-Float[1]  | 192.168.1.151 | 44818 |
| <input type="checkbox"/> | SSC-Float[2]  | 192.168.1.152 | 44818 |
| <input type="checkbox"/> | SSC-Float[3]  | 192.168.1.153 | 44818 |
| <input type="checkbox"/> | SSC-Float[4]  | 192.168.1.154 | 44818 |

< Back

Next >

Finish

Cancel

### PLC Tag Configuration

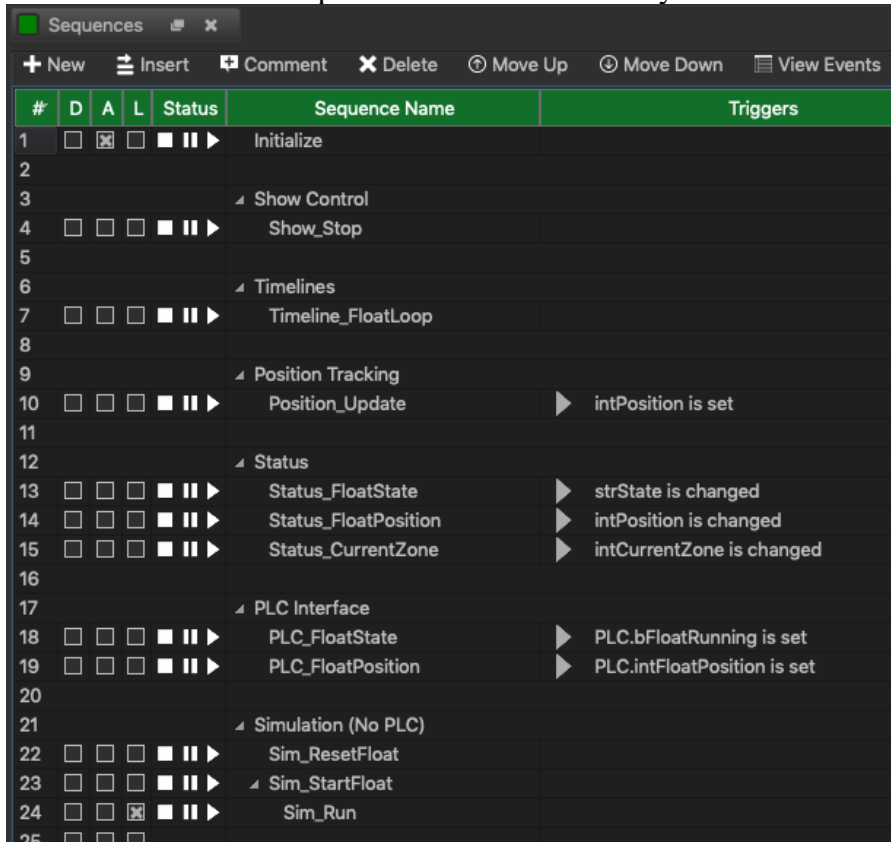
The script needs to be configured to scan the PLC tags so that the RidePlayer can sense the float’s position along the parade route. It’s common for this type of information to be stored in INT or DINT tags. To make this data easier to reference within our script, we can utilize the Alias feature to assign them with descriptive names:

|    |                   |                      |         |   |  |
|----|-------------------|----------------------|---------|---|--|
| 10 | PLC.PL_CTO_SSC    |                      | Integer | 0 | Size: 2. Tag: FLOATSTATUS. Poll: 00:00:00.01. Read Only. |
|    | PLC.PL_CTO_SSC[0] |                      | Integer | 0 |  |
|    | PLC.PL_CTO_SSC[1] | PLC.intFloatPosition | Integer | 0 |  |

## Sequences – Float

Sequences are the heart of the show control script and contain all of the functional events that are programmed. Let's take a moment to walk through the key sequences contained in these example scripts.

Here's what the float sequences within the RidePlayer look like:

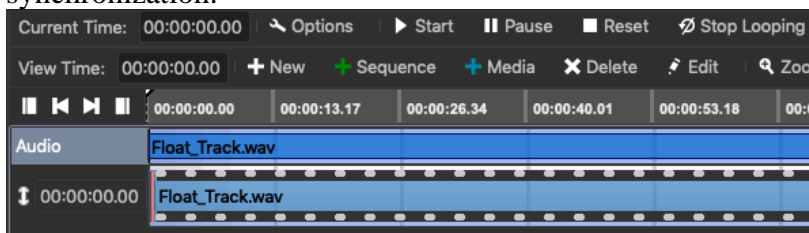


| #  | D                        | A                                   | L                                   | Status                              | Sequence Name        | Triggers                    |
|----|--------------------------|-------------------------------------|-------------------------------------|-------------------------------------|----------------------|-----------------------------|
| 1  | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Initialize           |                             |
| 2  |                          |                                     |                                     |                                     |                      |                             |
| 3  |                          |                                     |                                     |                                     | Show Control         |                             |
| 4  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Show_Stop            |                             |
| 5  |                          |                                     |                                     |                                     |                      |                             |
| 6  |                          |                                     |                                     |                                     | Timelines            |                             |
| 7  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Timeline_FloatLoop   |                             |
| 8  |                          |                                     |                                     |                                     |                      |                             |
| 9  |                          |                                     |                                     |                                     | Position Tracking    |                             |
| 10 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Position_Update      | intPosition is set          |
| 11 |                          |                                     |                                     |                                     |                      |                             |
| 12 |                          |                                     |                                     |                                     | Status               |                             |
| 13 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Status_FloatState    | strState is changed         |
| 14 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Status_FloatPosition | intPosition is changed      |
| 15 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Status_CurrentZone   | intCurrentZone is changed   |
| 16 |                          |                                     |                                     |                                     |                      |                             |
| 17 |                          |                                     |                                     |                                     | PLC Interface        |                             |
| 18 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | PLC_FloatState       | PLC.bFloatRunning is set    |
| 19 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | PLC_FloatPosition    | PLC.intFloatPosition is set |
| 20 |                          |                                     |                                     |                                     |                      |                             |
| 21 |                          |                                     |                                     |                                     | Simulation (No PLC)  |                             |
| 22 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Sim_ResetFloat       |                             |
| 23 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | Sim_StartFloat       |                             |
| 24 | <input type="checkbox"/> | <input type="checkbox"/>            | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | Sim_Run              |                             |
| 25 | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/>            |                                     |                      |                             |

First, there is a sequence called **Initialize** which does nothing more than fill the front-panel display of the RidePlayer with some text. This is common practice since the RidePlayer display is a handy way to display status information to operators and maintenance staff.

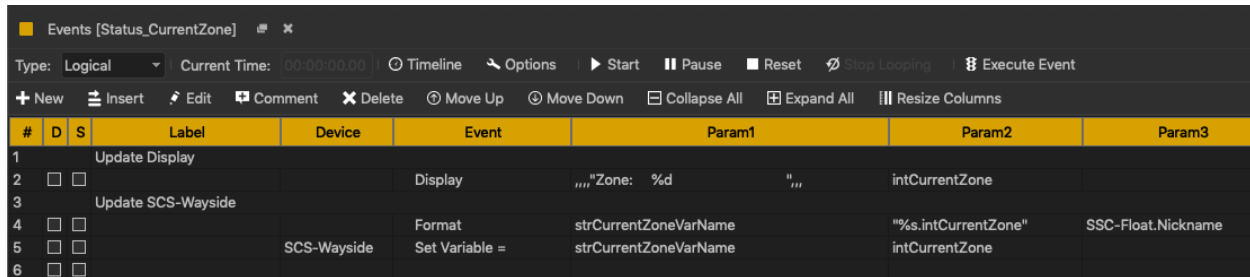
Then we have **Show\_Stop** sequences which can be utilized by the V16X to stop the show (Audio, Video, Lighting, etc.) on the parade float.

Next we have our parade loop timeline called **Timeline\_FloatLoop**. The contents of this timeline are designed to play synchronously with the wayside timeline as well as the timelines of the other floats. It is re-triggered every loop by the wayside V16X to ensure perfect synchronization.



The **Position\_Update** sequence runs every time the position of the float vehicle changes. Its purpose is to determine which parade zone the float is currently located within.

The **Status** sequences play two important roles. The first is updating the front-panel display on each RidePlayer with the float's current state, position, and parade zone. The second, and most important, is relaying all changes of this data to the wayside V16X so that it can route wayside audio and video to the zone closest to the float's current location.



| # | D                        | S                        | Label              | Device      | Event          | Param1                | Param2              | Param3             |
|---|--------------------------|--------------------------|--------------------|-------------|----------------|-----------------------|---------------------|--------------------|
| 1 |                          |                          | Update Display     |             |                |                       |                     |                    |
| 2 | <input type="checkbox"/> | <input type="checkbox"/> |                    |             | Display        | ""Zone: %d            | ""                  | intCurrentZone     |
| 3 |                          |                          | Update SCS-Wayside |             |                |                       |                     |                    |
| 4 | <input type="checkbox"/> | <input type="checkbox"/> |                    |             | Format         | strCurrentZoneVarName | "%s.intCurrentZone" | SSC-Float.Nickname |
| 5 | <input type="checkbox"/> | <input type="checkbox"/> |                    | SCS-Wayside | Set Variable = | strCurrentZoneVarName | intCurrentZone      |                    |
| 6 | <input type="checkbox"/> | <input type="checkbox"/> |                    |             |                |                       |                     |                    |

The **PLC** sequences simply process any updates to the PLC tags and load this information into local variables used to trigger the sequences mentioned above as needed.

The **Simulation** sequences are used for test purposes only when PLCs are not available for use. These sequences allow you to simulate the movement of the floats along the parade route so you can test V16X communication, routing, etc.

## Sequences – Wayside

The sequences of the wayside V16X shares some similarities but have other responsibilities to manage.

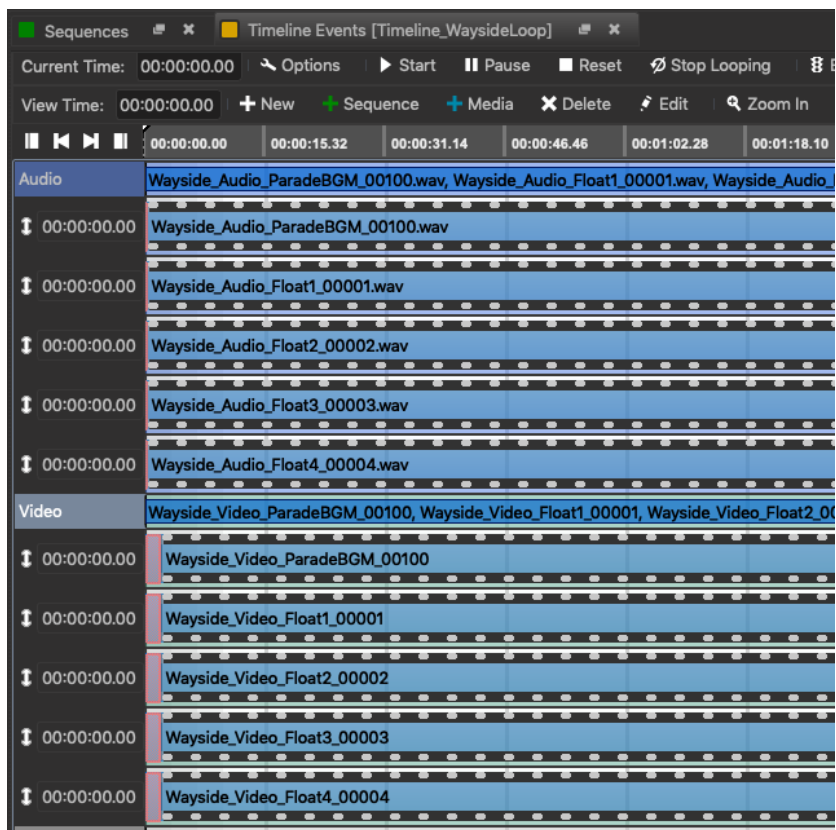
| Sequences   |                          |                                     |                          |                          |                           |                                |  |  |  |
|---|--------------------------|-------------------------------------|--------------------------|--------------------------|---------------------------|--------------------------------|--|--|--|
| + New ≡ Insert 🗨 Comment ✕ Delete ⏶ Move Up ⏷ Move Down 📄 View Events ⚙ |                          |                                     |                          |                          |                           |                                |  |  |  |
| #   | D                        | A                                   | L                        | Status                   | Sequence Name             | Triggers                       |  |  |  |
| 1   | <input type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | Initialize                |                                |  |  |  |
| 2   |                          |                                     |                          |                          |                           |                                |  |  |  |
| 3   |                          |                                     |                          |                          | └ Parade Control          |                                |  |  |  |
| 4   | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Parade_Stop               |                                |  |  |  |
| 5   | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Parade_Start              |                                |  |  |  |
| 6   | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Parade_TriggerLoop        |                                |  |  |  |
| 7   |                          |                                     |                          |                          |                           |                                |  |  |  |
| 8   |                          |                                     |                          |                          | └ Timelines               |                                |  |  |  |
| 9   | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Timeline_WaysideLoop      |                                |  |  |  |
| 10  |                          |                                     |                          |                          |                           |                                |  |  |  |
| 11  |                          |                                     |                          |                          | └ Float Management        |                                |  |  |  |
| 12  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Float_UpdateZones         | ▶ SSC-Float1.intCurrentZone is |  |  |  |
| 13  |                          |                                     |                          |                          |                           |                                |  |  |  |
| 14  |                          |                                     |                          |                          | └ DSP Control             |                                |  |  |  |
| 15  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | DSP_UpdateZoneRouting     | ▶ strZoneState[1] is changed   |  |  |  |
| 16  |                          |                                     |                          |                          |                           |                                |  |  |  |
| 17  |                          |                                     |                          |                          | └ Video Processor Control |                                |  |  |  |
| 18  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | VP_UpdateZoneRouting      | ▶ strZoneState[1] is changed   |  |  |  |
| 19  |                          |                                     |                          |                          |                           |                                |  |  |  |
| 20  |                          |                                     |                          |                          | └ Status                  |                                |  |  |  |
| 21  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Parade             | ▶ strParadeState is changed    |  |  |  |
| 22  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone1              | ▶ strZoneState[1] is changed   |  |  |  |
| 23  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone2              | ▶ strZoneState[2] is changed   |  |  |  |
| 24  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone3              | ▶ strZoneState[3] is changed   |  |  |  |
| 25  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone4              | ▶ strZoneState[4] is changed   |  |  |  |
| 26  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone5              | ▶ strZoneState[5] is changed   |  |  |  |
| 27  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone6              | ▶ strZoneState[6] is changed   |  |  |  |
| 28  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone7              | ▶ strZoneState[7] is changed   |  |  |  |
| 29  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Status_Zone8              | ▶ strZoneState[8] is changed   |  |  |  |
| 30  |                          |                                     |                          |                          |                           |                                |  |  |  |
| 31  |                          |                                     |                          |                          | └ Simulation              |                                |  |  |  |
| 32  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Sim_Reset                 |                                |  |  |  |
| 33  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | └ Sim_Start               |                                |  |  |  |
| 34  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | └ Sim_Run                 |                                |  |  |  |
| 35  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Sim_Float1_Go             |                                |  |  |  |
| 36  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Sim_Float2_Go             |                                |  |  |  |
| 37  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Sim_Float3_Go             |                                |  |  |  |
| 38  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> | Sim_Float4_Go             |                                |  |  |  |
| 39  | <input type="checkbox"/> | <input type="checkbox"/>            | <input type="checkbox"/> | <input type="checkbox"/> |                           |                                |  |  |  |

One of those responsibilities is triggering the parade loops of the wayside system and ALL parade floats in perfect synchronization with each other. This is achieved in the **Parade\_TriggerLoop** sequence.

| # | D | S                                   | Label  | Time | Device     | Event | Param1               |
|---|---|-------------------------------------|--|------|------------|-------|----------------------|
| 1 |   | <input checked="" type="checkbox"/> | Synchronously Trigger Wayside Loop and Float Loops |      |            |       |                      |
| 2 |   | <input checked="" type="checkbox"/> |  |      |            | Start | Timeline_WaysideLoop |
| 3 |   | <input checked="" type="checkbox"/> |  |      | SSC-Float1 | Start | "Timeline_FloatLoop" |
| 4 |   | <input checked="" type="checkbox"/> |  |      | SSC-Float2 | Start | "Timeline_FloatLoop" |
| 5 |   | <input checked="" type="checkbox"/> |  |      | SSC-Float3 | Start | "Timeline_FloatLoop" |
| 6 |   | <input checked="" type="checkbox"/> |  |      | SSC-Float4 | Start | "Timeline_FloatLoop" |

Notice that the checkboxes in the ‘S’ (Sync) column are enabled. This means that these five sequences (Wayside + 4 parade floats) will be scheduled to trigger simultaneously based upon the shared NTP clock.

The **Timeline\_WaysideLoop** sequence manages all playback of wayside audio and video, and also re-triggers the parade loop when it is complete.





The **Float\_UpdateZones** sequence is responsible for processing any changes to parade float location. Whenever the RidePlayer onboard a parade float inform the V16X that it has entered a different zone, this sequence determines the new state of each zone along the entire parade route. The logic within this sequence will make sure that all zones ahead of the first float in the parade are configured to output “Park BGM”. It will also restore all zones behind the last parade float to “Park BGM”. Of course, any zone that has a parade float within it will be configured accordingly (i.e. “Float1”, “Float3”, etc.). Any zones located between the first float and the last float without any floats located within them will be configured for “ParadeBGM”.

Once **Float\_UpdateZones** determines the proper status for all parade zones, **DSP\_UpdateZoneRouting** and **VP\_UpdateZoneRouting** will run as needed to update the audio routing in the DSP and video routing in the Video Processor.

The **Status** sequences do nothing more than update the front-panel display of the V16X so you can easily see the status of the parade loop as well as the current zone routing.

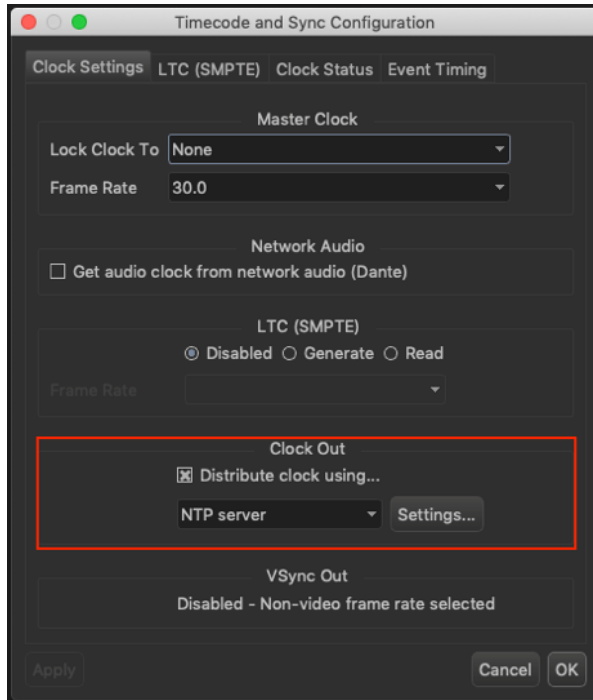
The **Simulation** sequences leverage the simulation feature of the RidePlayer float script to simulate the functionality of the parade without physically driving the floats. This is very handy for creating mockup systems or testing parade hardware and content without driving the floats through the park.

Last but not least, the ShowTouch panel interface is extremely useful in monitoring and controlling the parade. It’s also quite handy in visualizing how this system works as the parade floats progress down the route.



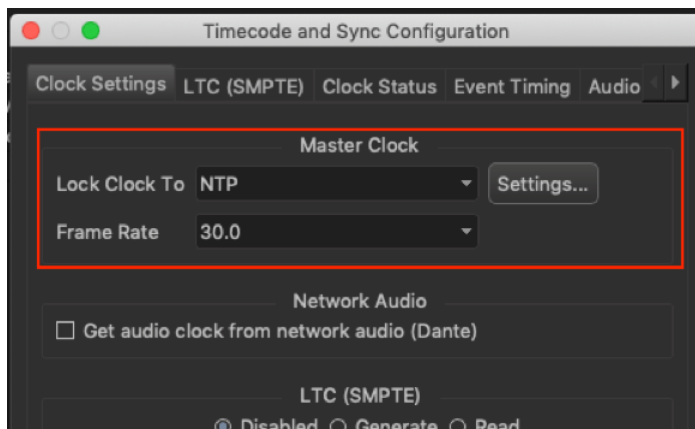
## Clock Configuration – Wayside

For everything to synchronize precisely, we need to configure both the V16X and RidePlayers to share their clocks with one another. For this application, the V16X functions as the clock master and distributes this clock as an NTP Server.



## Clock Configuration – Float

The RidePlayers operate as NTP clients so they can lock to the V16X NTP Server.



With this configuration and proper networking infrastructure, these products can synchronize clocks with sub-millisecond precision to ensure the scene timelines are triggered with incredible accuracy.

## Conclusion

This application note can serve as a starting point in implementing your own parade application. Keep in mind that it's easy to scale the system to include as many parade floats and zones as you need.

Now it's time for you to implement your own project with the V16X, RidePlayer, RideAmp, and our A/V Binloop family of products. Please don't forget that we are here to help you so feel free to contact us with questions.