Overview

Event Video Playback is a comprehensive system that consists of three integrated components:

1. **The EventVideoPlayback Service** - A Windows service that handles creation of the videos
2. **The Event Video Playback Library** - A PLC library that maintains an Image Ring Buffer containing enough images to create videos of configurable length and actively interacts with the EventVideoPlayback Service
3. **The Event Video Playback HMI Control** - A modified Logger control that allows associated videos to be played from log entries

Important: Video viewing is only functional in a published, fully deployed project when viewed from a web browser. Videos cannot be viewed via the HMI Live display.

This guide will walk you through configuring each component to work together in your TwinCAT project.

EventVideoPlayback Service Configuration

The EventVideoPlayback Service controls video creation and automatic deletion of video files based on age and folder size limits.

Note: The service reads the configuration file on startup. You must restart the service for any configuration changes to take effect.

To restart the service: 1. Open the **Windows Services** window 2. Right-click the **EventVideoPlayback** service 3. Choose **Stop** or **Start**

Configuration File Location

The configuration file is located at:

```
C:\Program Files\Beckhoff USA Community\EventVideoPlayback\Service\EventVideoPlaybackService.config.json
```

Open the file with **Notepad** or **Visual Studio** to edit the following parameters:

```
{
 "CodecFourCC": "avc1",
 "VideoDeleteTime": 1,
 "AdsPort": 26129,
 "MaxFolderSize": 250
}
```

CodecFourCC

This is the codec used to create the video from the service. `avc1` is the most common codec that is also compatible with web browsers. For a full list of codecs see below, but not all are web compatible and will display as a black box on Tc HMI. Check here for appropriate codecs.

Note: The most common web-compatible codec is `avc1`. Other codecs may not display correctly in web browsers.

```
OpenCV: FFMPEG: format mp4 / MP4 (MPEG-4 Part 14)
fourcc tag 0x7634706d/'mp4v' codec_id 000C
fourcc tag 0x31637661/'avc1' codec_id 001B
fourcc tag 0x33637661/'avc3' codec_id 001B
fourcc tag 0x31766568/'hev1' codec_id 00AD
fourcc tag 0x31637668/'hvc1' codec_id 00AD
```

```
fourcc tag 0x7634706d/'mp4v' codec_id 0002
fourcc tag 0x7634706d/'mp4v' codec_id 0001
fourcc tag 0x7634706d/'mp4v' codec_id 0007
fourcc tag 0x7634706d/'mp4v' codec_id 003D
fourcc tag 0x7634706d/'mp4v' codec_id 0058
fourcc tag 0x312d6376/'vc-1' codec_id 0046
fourcc tag 0x63617264/'drac' codec_id 0074
fourcc tag 0x7634706d/'mp4v' codec_id 00A3
fourcc tag 0x39307076/'vp09' codec_id 00A7
fourcc tag 0x31307661/'av01' codec_id 801D
fourcc tag 0x6134706d/'mp4a' codec_id 15002
fourcc tag 0x63616c61/'alac' codec_id 15010
fourcc tag 0x6134706d/'mp4a' codec_id 1502D
fourcc tag 0x6134706d/'mp4a' codec_id 15001
fourcc tag 0x6134706d/'mp4a' codec_id 15000
fourcc tag 0x332d6361/'ac-3' codec_id 15003
fourcc tag 0x332d6365/'ec-3' codec_id 15028
fourcc tag 0x6134706d/'mp4a' codec_id 15004
fourcc tag 0x61706c6d/'mlpa' codec_id 1502C
fourcc tag 0x43614c66/'fLaC' codec_id 1500C
fourcc tag 0x7375704f/'Opus' codec_id 1503C
fourcc tag 0x6134706d/'mp4a' codec_id 15005
fourcc tag 0x6134706d/'mp4a' codec_id 15018
fourcc tag 0x6134706d/'mp4a' codec_id 15803
fourcc tag 0x7334706d/'mp4s' codec_id 17000
fourcc tag 0x67337874/'tx3g' codec_id 17005
fourcc tag 0x646d7067/'gpmd' codec_id 18807
fourcc tag 0x316d686d/'mhm1' codec_id 15817
```

VideoDeleteTime

This setting determines how long video files remain on the system before automatic cleanup.

- **Value type:** Floating point
- **Units:** Days
- **Example:** Use `0.5` for half a day (12 hours)

The service automatically deletes video files older than the specified time.

AdsPort

Warning: Do not change this value. This is the ADS Port that the service is hosting on. This should remain at port 26129 at all times in order for the PLC function blocks to work properly.

MaxFolderSize

This setting limits the total size of the video storage folder.

- **Value type:** Integer
- **Units:** MB (megabytes)

When a new video is created, the service checks the folder size. If the limit is exceeded, the oldest video will be automatically deleted to free up space.

PLC Function Block Parameters

The `FB_ImageToVideo` function block maintains an image buffer in Router memory. The memory requirements are directly related to:

- **FramesPerSecond** - Higher frame rates require more memory
- **Record time** - Longer videos require more memory
- **Image size** - Larger images require more memory
- **ReductionFactor** - Smaller reduction factors (closer to 1.0) require more memory

Important: Larger values of these parameters require more Router memory. Plan your Router memory allocation accordingly.

```
VAR
   // ImageToVideo Instance
   Playback : FB_ImageToVideo := (CameraName := 'Camera1',
                                  FramesPerSecond := 10,
                                  TimeBeforeEvent := T#3S,
                                  TimeAfterEvent := T#3S,
                                  VideoOutputDirectory := 'C:\EventVideos',
                                  ReductionFactor := 0.25);

   // Event Trigger Boolean
   TriggerEvent1 : BOOL;
   TriggerEvent2 : BOOL;
END_VAR
```

Parameter Descriptions

CameraName

A unique identifier for each `FB_ImageToVideo` instance. - **Examples:** `Camera1`, `Infeed_Camera`, `Arm_Camera` - **Requirement:** Must be unique across all instances

FramesPerSecond

The rate at which images are added to the Image Ring Buffer. - **Value type:** Integer - **Example:** `10` frames per second

TimeBeforeEvent and TimeAfterEvent

These times combined determine the total video duration. - **Value type:** TIME (e.g., `T#3S`) - **Units:** Seconds - **Total video length:** TimeBeforeEvent + TimeAfterEvent

VideoOutputDirectory

The storage location for created videos. Videos are saved in a subfolder named after the CameraName. - **Example:** If set to `C:\EventVideos` and CameraName is `Camera1`, videos will be saved to `C:\EventVideos\Camera1\`

ReductionFactor

Scales down the original image before storing it in the buffer to reduce memory usage. - **Value type:** Decimal - **Range:** 0.1 to 1.0 - **Example:** `0.25` = 25% of original size (reduces memory by 75%)

PLC Event Video Playback Library Parameters

These library-level parameters ensure adequate Router Memory is available for your configuration. On startup, the system evaluates available Router Memory. If insufficient memory is detected, a log entry will be generated and Event Video Playback will not function.

MAX_NUMBER_IMAGES_2_VIDEO

Defines the maximum number of images in the ring buffer.

Formula: Must be at least 2 greater than: `FramesPerSecond × (TimeBeforeEvent + TimeAfterEvent)`

Example: For 10 FPS and 6 seconds total time: - Minimum required: (10 × 6) + 2 = **62 images**

MAX_PERCENT_ROUTER_MEM_FOR_BUFFER

Sets the maximum percentage of Router Memory allowed per `FB_ImageToVideo` instance.

Single Instance Example: - If using **one** `FB_ImageToVideo` instance, set this to `60` to allocate up to 60% of Router Memory

Multiple Instance Example: - If using **three** instances and want to allocate 60% total: - Set this to `60 / 3 = 20%` per instance - This ensures all three instances combined use no more than 60% of Router Memory

ADS_PORT_FOR_IMAGE_TO_VIDEO

Warning: Do not change this value. This is the ADS Port that the EventVideoPlayback service is hosting on. This should remain at port 26129 at all times in order for the PLC function blocks to work properly.

LARGE_ROUTER_MEMORY

This parameter specifies the Router Memory size to use for memory checks when the actual value cannot be read programmatically.

Note: In TwinCAT 4026.18, Router Memory larger than 4026 MB cannot be read programmatically. If you have more than 4026 MB of Router Memory, manually set this parameter to your actual Router Memory size.

Add to Existing TwinCAT Vision PLC Project

Prerequisites: Install Event Video Playback XAE and XAR packages using the TwinCAT Package Manager. See the Installation guide for details.

You can easily integrate Event Video Playback with existing TcVision projects by following these steps:

1. Add the Event Video Playback Library to the References

Add the Event Video Playback Library to the References section of the PLC Project.

2. Instantiate FB_ImageToVideo and TriggerEvent(s)

```
VAR
   // ImageToVideo Instance
   EVP1 : FB_ImageToVideo := (CameraName := 'Camera1',
                              FramesPerSecond := 10,
                              TimeBeforeEvent := T#3S,
```

```
                                TimeAfterEvent := T#3S,
                                VideoOutputDirectory := 'C:\TcEventVideos',
                                ReductionFactor := 0.25);

    // Event Trigger Boolean
    TriggerEvent1 : BOOL;
    TriggerEvent2 : BOOL;
END_VAR
```

3. Add a Reset Call to the First Scan

Add a Reset call to the first scan of POU:

```
EVP1.Reset();
```

4. Add the CyclicLogic Call

Add the CyclicLogic call to the main body of the POU. This **MUST** be called cyclically to work.

```
EVP1.CyclicLogic();
```

5. Add the AddImage Method

Add the AddImage method to the "new image" section of your program. This will add an image to the buffer of the Playback block.

```
EVP1.AddImage(ipImageIn := ImageIn);
```

6. Add the Trigger Logic

Add the trigger logic somewhere in your program. The `TriggerAlarmForVideoCapture` method only needs to be called once to start Event processing. Multiple event names can be used for events generating a log entry for the same ImageToVideo Instance.

```
IF TriggerEvent1 THEN
    TriggerEvent1 := FALSE;
    EVP1.TriggerAlarmForVideoCapture(LogEntryName := 'Log Entry Name1');
END_IF

IF TriggerEvent2 THEN
    TriggerEvent2 := FALSE;
    EVP1.TriggerAlarmForVideoCapture(LogEntryName := 'Log Entry Name2');
END_IF
```

HMI Configuration

Add the EventVision NuGet Package

1. Open the **NuGet Package Manager** in your HMI project
2. Go to the **Browse** window
3. Search for and add the **EventVision** package

Tip: If the package does not appear, verify that the Package source is set to **Beckhoff Offline Packages**.

Add the EventVisionControl

Once the package is installed, add the **EventVisionControl** to your HMI from the Toolbox.

HMI EventVisionControl Properties

After adding the EventVisionControl, set the properties:

Important: The Virtual drive setting must match that specified in the HMI Publish configuration. The Time Zone Info is required if the viewing browser system time is in a different time zone than the TwinCAT HMI Server.

HMI Publish Configuration

Configure a virtual directory to allow the HMI to access video files stored on the system.

Configure Virtual Directory

1. In the **Solution Explorer**, navigate to **Server** → **TcHmiSrv**
2. Add a virtual directory pointing to your video storage location

Important: Be aware of your Publish Configuration. If using a "Remote" Publish Configuration, ensure you add the virtual directory to the corresponding "Remote" configuration via the dropdown menu on the TcHmiSrv page.

Next Steps

Now that you have configured Event Video Playback, you can:

- Test video capture by triggering events in your PLC code
- Review captured videos in your HMI
- Adjust service configuration parameters as needed
- Configure additional cameras by creating more `FB_ImageToVideo` instances

Support

Need help? Here are some resources:

- GitHub Issues - Report bugs or request features
- Beckhoff USA Community - Community support and discussions
- Documentation - Additional guides and references