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. The service supports multiple codecs, but not all are web-compatible. For TwinCAT HMI compatibility, stick with these recommended options:

Codec	FourCC	Description	Web Compatible	Recommended
H.264	`avc1`	Most common, best compatibility	✓	**Yes**

H.264	`avc3`	Alternative H.264 variant	✓	Yes
H.265/HEVC	`hev1`	Better compression, newer	✓	Maybe
H.265/HEVC	`hvc1`	Alternative HEVC variant	✓	Maybe
MPEG-4	`mp4v`	Older standard	✓	No
VP9	`vp09`	Google's codec	✓	No
AV1	`av01`	Newest, best compression	■	No

Tip: For best results, use `avc1` (H.264). It offers the widest compatibility across all web browsers and HMI platforms.

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** - Larger reduction factors (0.1 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

Overview

Event Video Playback is a comprehensive solution for transforming TwinCAT Vision images into event-driven video recordings. This guide will walk you through the installation process using TwinCAT Package Manager.

Prerequisites: Before starting, ensure you have TwinCAT Package Manager installed and have the necessary permissions to install packages.

System Requirements

Before you begin, ensure you have the following installed on your system:

For Engineering Development

- **Windows 10/11**
- **TwinCAT Package Manager**
- **TwinCAT 3.1 XAE** - Build 4026 or higher
- **TwinCAT Vision XAE** - Version 5.8.4 or higher
- **TwinCAT HMI XAE** - Version 14.9 or higher

For Runtime Targets

- **Windows 10/11**
- **TwinCAT Package Manager**
- **TwinCAT 3.1 XAR** - Build 4026 or higher
- **TwinCAT Vision XAR** - Version 5.8.4 or higher

Warning: Previous versions of the 4024 Tc_EventVideoPlayback legacy project must be uninstalled before using the new 4026 build.

Package Signing Configuration

TwinCAT Package Manager only accepts officially signed packages from Beckhoff Automation GmbH & Co. KG by default. To use community packages, you need to temporarily disable signature verification.

Security Notice: Disabling signature verification allows installation of third-party packages. Only use packages from trusted community sources. Review package contents and source code before installation. All community packages are provided "as is" without warranties.

To disable signature checks, run this command in PowerShell as Administrator:

```
tcpkg config unset -n VerifySignatures
```

Installation Methods

Choose the installation method that best fits your environment:

- **Online Installation** - For systems with internet access (recommended)
- **Offline Installation** - For air-gapped systems or manual installation

---

Online Package Feed Connection

Best for: Systems with internet connectivity and access to the Beckhoff USA Community package feed.

Option 1: Add Package Feed via GUI

Add the Beckhoff USA Community package feed to TwinCAT Package Manager:

1. Open the **TwinCAT Package Manager GUI**
2. Click the **Settings** (Gear Icon) in the bottom left corner
3. Select **Feeds**

   Add a new feed with the following settings:

   Feed URL: `https://packages.beckhoff-usa-community.com/stable/v3/index.json`

6. **Feed Name:** `Beckhoff USA Community Stable`

   Set credentials: Deselect (not required)

   Click **Save** and agree to the disclaimer

Option 2: Add Package Feed via PowerShell

Run the following command in PowerShell:

```
tcpkg source add -n "Beckhoff USA Community Stable" -s "https://packages.beckhoff-usa-community.com/stable/v3/index.json"
```

Agree to the disclaimer when prompted.

Offline Package Configuration

Best for: Air-gapped systems, manual installations, or when you need a specific version.

Step 1: Download from GitHub Releases

1. Navigate to the GitHub Releases page
2. Find the latest release (or the version you need)
3. Download the package file (`.zip`)
4. Transfer the files to your target system in a convenient location
5. **Example:** `C:\Program Files\Beckhoff USA Community\Feeds\Local`

Tip: Instead of downloading manually, you can use the PowerShell command: `tcpkg download [package name] -o [output location]`

Step 2: Add Local Package Feed

Option 1: Add via GUI

1. Open the **TwinCAT Package Manager GUI**
2. Click the **Settings** (Gear Icon) in the bottom left corner
3. Select **Feeds**

   Add a new feed with the following settings:

   Feed Path: `C:\Program Files\Beckhoff USA Community\Feeds\Local`

6. **Feed Name:** `Beckhoff USA Community Local`

   Set credentials: Deselect (not required)

Click **Save**

Option 2: Add via PowerShell

Run the following command in PowerShell:

```
tcpkg source add -n "Beckhoff USA Community Local" -s "C:\Program Files\Beckhoff USA Community\Feeds\Local"
```

Install Workloads

After completing the steps above (adding the package feed and disabling signature verification), you can now install the Event Video Playback workloads and packages through TwinCAT Package Manager, just like any other Beckhoff Automation package.

1. Open **TwinCAT Package Manager**
2. Browse the available packages from the Beckhoff USA Community feed
3. Select **Event Video Playback** workloads/packages
4. Click **Install**

Note: The installation will include the necessary XAE/XAR components, PLC libraries, HMI packages, and the Windows service.

Next Steps

After installation is complete, proceed to the Getting Started guide to learn how to configure and use Event Video Playback in your projects.

Troubleshooting

Service Won't Start

- Verify .NET 8 Runtime is installed
- Check Windows Event Viewer for error messages
- Ensure no other service is using ADS port 26129

Videos Not Being Created

- Confirm TwinCAT Vision is saving images to the configured path
- Check service configuration file for correct paths
- Verify sufficient disk space is available

PLC Function Block Errors

- Ensure the library reference is added correctly
- Verify the service is running
- Check ADS communication settings

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
