DynamicDisturbanceRecorder

IEC 61131 Library for ACSELERATOR RTAC® Projects

SEL Automation Controllers

Table of Contents

RTAC LIBRARY

DynamicDisturbanceRecorder

Introduction

The DynamicDisturbanceRecorder library contains classes designed to simplify the collection of data points from within the logic engine of the Real-Time Automation Controller (RTAC). All data must be added to the logging object by using the supported bootstrap methods and, as such, the data must originate from a data structure supported by the logging object selected. The maximum rate at which points are logged is dependent on the task cycle of the *Main Task*. See the *ACSELERATOR RTAC® SEL-5033 Software Instruction Manual* for more details on configuring the task cycle.

Each object within the library stores the collected data into a file format specific to that object. The three available file formats are listed below:

- **Time-Aligned CSV**: This format is an ASCII comma-separated value (CSV) file that aligns all point values with a single time value that represents the data. The order in which the data points are bootstrapped determines the position of the data points in the log. A null entry in this format is represented as a blank entry (no characters) delimited by two commas. Supported data structures and simple types include BCR, UDINT, CMV, REAL, DPS, INS, DINT, MV, SPS, BOOL, STR, and STRING(80).
- **SOE CSV**: This format is an ASCII CSV file in which each log represents the status value of a bootstrapped point and a time value. Additional information in the log includes the time code (offset from UTC), station name, and device name. Each detected change generates a log that is a separate line in the file. Supported data structures and simple types include: BCR, UDINT, CMV, REAL, DPS, INS, DINT, MV, SPS, BOOL, STR, and STRING(80).
- **COMTRADE**: Two COMTRADE files are generated that adhere to IEC 60255-24:2013 and IEEE Std. C37.111-2013: a CFG file and a DAT file. The CFG file is an ASCII file that includes information pertinent to the interpretation of the DAT file. The maximum size of the CFG file cannot exceed 10 MB. The DAT file contains the log entries and is a binary file formatted in FLOAT32. Supported data structures and simple types include CMV, MV, REAL, INS, DINT, SPS, and BOOL. The order of the points in the DAT is the order in which the supported data structures are listed above and then in the order the points are bootstrapped. Within each data structure type, the order of the points is determined by the order in which the points are bootstrapped.

For the COMTRADE format, missing data will be filled with hex value 0xFF7FFFFF in the binary DAT file for analog points. For binary points, missing data will be represented as a zero. See the referenced standards above for more detail pertaining to the COMTRADE file format.

Each object also has specific triggering mechanisms that trigger the collection of the bootstrapped points. The objects may contain one or more triggering options, but only one trigger option can be used per instantiated object. When the trigger condition is met, a log or log entry is created. The maximum log or log entry size is 10 MB. The log will at minimum contain a time value and the status data of the bootstrapped points. Additional data may also be present depending on the file format chosen. These triggers are categorized into the following four categories:

- **Timestamp Change**: Available file formats for the Timestamp Change trigger include time-aligned CSV and COMTRADE. This trigger method monitors the bootstrapped points for changes in the dateTime_t data structure. A detected change in any dateTime_t data structure within a bootstrap point creates a log with a time value. Changes detected in simple data types do not generate logs because the data do not contain a time stamp. Each log entry contains all the points for which a time change was detected. By default, the first dateTime_t data structure for which a time change is detected is the time reference of the log entry. This is the time that is associated with the log entry. A user can specify a timeStamp_t data structure that is used as the time reference to better control when log entries are created and what time stamp is associated with the log. Any detected change in the dateTime_t data structure of the time reference generates a log and the time value associated with that log is derived from the dateTime_t data structure of the time reference. Time variance is an optional setting that allows users to set a window around the time reference. If a time change is detected during the scan, but the time value of that data point does not fall within the time window relative to time reference, it will not be included in the log. Points that do not change or are outside the time window have a null entry. This entry maintains the columnar position in the log entry as determined by the file format. The exception is for simple types. All simple types are included in any log generated.
- **Data Change**: The only file format available for the Data Change trigger is SOE CSV. This trigger method monitors the bootstrapped points for changes in the respective status value for the data structure. If a change in the status is detected, a distinct log entry is created for that point that reflects the time of change and data status. The time of change is derived from the dateTime_t data structure of the bootstrapped point. In the case of simple types, system time is assigned to each change in state.

NOTE: For MV and CMV data structures, the monitored quantity is the deadbanded attribute. For example, for a CMV, the mag and ang attributes are monitored.

- **Periodic**: Available file formats for the Periodic trigger include time-aligned CSV and COMTRADE. This trigger method periodically samples all points at the specified interval regardless of whether the time value or the data status value of the bootstrapped point changed. The time value for the log entry is the system time of the RTAC at the time of the periodic interval. All data are sampled, so no null entries are present. This is a snapshot of the points as they are in the RTAC logic engine at the time of the periodic interval.
- **Trigger**: Available file formats include time-aligned CSV and COMTRADE. Triggered file records cannot exceed 10 MB in total file size. This trigger method monitors a Boolean value specified by the user. When the Boolean trigger condition is evaluated as TRUE, a file is generated that contains a configurable number of pre-trigger and post-trigger log entries in addition to a log entry for the time the trigger asserted. New log entries are created each task cycle; the amount of time captured by an individual log can be calculated by multiplying the number of scans (pre-trigger count, post-trigger count, and the trigger scan) by the task cycle time setting of the RTAC. If a trigger condition is detected before the minimum configured pre-trigger cycles

are met, a log will be created that contains the available pre-trigger log entries plus the trigger and the post-trigger log entries. If a trigger condition is detected before a previous trigger event is processed, it will be ignored. Like the Periodic type trigger, a value is populated for each bootstrapped point, regardless of change in data status or time stamp.

Each object is intended to be configured one time at program start. The initial configuration determines the trigger behavior, file parameters, and file management for the object. The type of object instantiated determines the format of the logs in the saved file.

For optimal file system performance, each object must be configured such that the rate of file generation does not exceed the ability of the file system to process the created files. For example, if an object is configured to log data at a fast rate or with a small maximum file size and this configuration results in a file being created every 10 processing cycles, the file system will not be able to process the files at a rate equal to the creation rate. When configuring your object, ensure that files are not created more frequently than every 100 processing cycles. This behavior is tuned by setting the MaxFileSize appropriately for the application. To verify that the object is creating files at an unsustainable rate, monitor the ActiveFileName output pin. If this name is changing faster than once every 100 processing cycles, the object may encounter issues when trying to create or delete files.

Supported Firmware Versions

You can use this library on any device configured using ACSELERATOR RTAC® SEL-5033 Software with firmware version R144 or higher.

Versions 3.5.2.0 and later can be used on RTAC firmware version R144 and later.

Versions 3.5.0.0 and later can be used on RTAC firmware version R139 and later.

To enable DynamicDisturbanceRecorder library support, the device number of your RTAC must include the feature in its model option table (MOT). You cannot download projects that include this library to RTACs that do not support the library. Use the SEL website MOT configuration (https://selinc.com/products/) to ensure that a particular part number has DynamicDisturbanceRecorder support enabled.

Enumerations

Enumerations make code more readable by allowing a specific number to have a readable textual equivalent.

enum_DataLogger

This enumeration defines the types of data loggers used to record data.

Enumeration	Value	Description
TIME_CHANGE	0	When a change is detected in the dateTime_t data structure of a bootstrapped point, a log is generated.
DATA_CHANGE	1	When a change is detected in the respective data status value of a bootstrapped point, a log entry generated.
PERIODIC	2	On a periodic interval, a log entry is generated.
TRIGGER_EVENT	3	When a trigger condition is detected, a log is generated.

Classes

class_TimeAlignedCsv (Function Block)

This class monitors and logs bootstrapped data points. Each time the Run method is called, it scans the bootstrapped points for the trigger criteria described by the DataLogType input setting. If a change is detected based on that criteria, the information is formatted and written to a time-aligned CSV file.

The log is then written into the directory specified by DirectoryPath. A new log will start after MaxFileSize is reached. The number of files stored in the directory is dependent on the following two settings: MaxFolderSize and MaxNumDays. So long as the cumulative size of the stored logs does *not* exceed MaxFolderSize, the file system will store logs for MaxNumDays. If the cumulative number of logs exceeds MaxFolderSize, the directory manager will delete the oldest files until the number of logs is less than MaxFolderSize. File names will begin with the start time of the log and will be appended with FilePostFix. For the trigger condition mechanism, an additional instance number will be appended to ensure uniqueness. This number will increment each time a log is generated and will reset with a settings change or when the maximum number of 65535 is reached.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable data recording.
RecordTimeInUtc	BOOL	Convert log time into UTC time. Information in the timestamp_t of the log source is used for this calculation.
RefTime	timestamp_t	Provides a time-stamp reference for log creation; ignored unless DataLogType TIME_CHANGE trigger.
TimeVariance	UDINT	Time window in milliseconds applied to the time reference. If a change in the dateTime_t structure for a bootstrapped point is detected and within the time window, it is included in the log entry. Set to 0 to disable. Ignored unless DataLogType TIME_-CHANGE trigger.
DataLogType	enum_DataLogger	Specifies what trigger mechanism is used to create logs.
DirectoryPath	STRING(127)	The full directory path at which files generated by this function block can be found.
FilePostfix	STRING(16)	The string appended after the time stamp in the generated file name.

Inputs

Name	IEC 61131 Type	Description
MaxFolderSize	LINT	The maximum size, in bytes, for the specified directory path.
MaxFileSize	DINT	The maximum file size, in bytes, at which a new file will be created for subsequent logging operations. Defaults to 10 MB for TRIGGER_EVENT records.
MaxNumDays	DINT	The maximum number of days for which log files are allowed to persist in a specified directory path.
LoggingInterval	TIME	Time in milliseconds for cyclic recording, ignored unless DataLogType PERIODIC trigger.
TriggerSignal	BOOL	Monitored variable that initiates a record, only for DataLogType TRIGGER_EVENT.
PreTriggerCycles	UDINT	Number of processing cycles prior to the trigger event for which data will be recorded; ignored unless DataLogType TRIGGER_EVENT trigger. Minimum of 1 pre-trigger cycle is required. If configured for less than the minimum, the setting will default to 1.
PostTriggerCycles	UDINT	Number of processing cycles after the trigger event for which data will be recorded; ignored unless DataLogType TRIGGER_EVENT trigger. Minimum of 1 post-trigger cycle is required. If configured for less than the minimum, the setting will default to 1.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The logging block is enabled.
ActiveFileName	STRING(255)	Name of the file that logs are being written to.
ConsumedDirectorySize	ULINT	Size of all files in the monitored directory.
MonitoredPoints	UDINT	Total number monitored points.
Error	BOOL	Something has prevented this block from successfully writing data to file. This can indicate an out of space condition or that too many data points are being monitored for the CPU of the RTAC to handle.
ErrorDesc	STRING(255)	Message describing the source of the *Error* flag.

bootstrap_MonitorBCR (Method)

A configuration method for adding BCR points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the BCR point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	BCR	The BCR point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the BCR point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorUDINT (Method)

A configuration method for adding UDINT points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the UDINT point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	UDINT	The UDINT point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the UDINT point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorCMV (Method)

A configuration method for adding CMV points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the CMV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	CMV	The CMV point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the CMV point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorREAL (Method)

A configuration method for adding REAL points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the REAL point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	REAL	The REAL point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the REAL point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorDPS (Method)

A configuration method for adding DPS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the DPS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	DPS	The DPS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the DPS point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorINS (Method)

A configuration method for adding INS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the INS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	INS	The INS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the INS point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorDINT (Method)

A configuration method for adding DINT points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the DINT point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	DINT	The DINT point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the DINT point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorMV (Method)

A configuration method for adding MV points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the MV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	MV	The MV point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the MV point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSPS (Method)

A configuration method for adding SPS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the SPS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	SPS	The SPS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the SPS point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorBOOL (Method)

A configuration method for adding BOOL points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the BOOL point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	BOOL	The BOOL point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the BOOL point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSTR (Method)

A configuration method for adding STR points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the STR point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	STR	The STR point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the STR point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSTRING (Method)

A configuration method for adding STRING points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the STRING point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	STRING	The STRING point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the STRING point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

Run (Method)

This method must be called every scan to ensure proper functionality, which includes scanning data points and writing files.

Processing

- The first time the method is called, it blocks all future bootstrap methods from adding new points.
- For DataLogType TIME_CHANGE, points are considered changed if the dateTime_t data structure has changed since the previous scan and the change falls within the window, relative to the time reference, specified by TimeVariance. Changes in simple type values do not trigger a log but are included in every log.
- For DataLogType PERIODIC, all points are considered changed if LoggingInterval is reached.
- For DataLogType TRIGGER_EVENT, all points are considered changed when logging data points during the pre- and post-trigger cycles.
- Scans all data points. If they have changed, the data are formatted for logging.
- If a log entry is generated, the class writes one line containing a time value with millisecond precision and one comma-separated column for each data point monitored. If the value of the data point changed since the last scan, the new value is written to the log entry; otherwise, a null entry is created. This ensures that the columnar position of all data points is maintained.

- Manage the total size of the directory path folder to ensure that the cumulative size of the logs does not exceed MaxDirectorySize.
- So long as MaxDirectorySize is not exceeded, the logs will be maintained for MaxNumDays.
- If MaxDirectorySize is exceeded before MaxNumDays is reached, the oldest file is deleted. This repeats until the cumulative directory size is less than MaxDirectorySize.
- Starts a new file if the active log file exceeds the maximum file size, if the project settings are changed, or if trigger condition is detected.
- The EN pin must be TRUE to scan and write new log entries.
- When EN is FALSE, no data point changes are detected.
- A rising edge on EN clears all pending logs so that only changes since the EN toggling to true will be written to future logs.
- Deleting metadata in the .RetainedState and .unsent files in the specified directory may result in inconsistent logging behavior.

class_SoeCsv

This class monitors and logs bootstrapped data points. Each time the Run method is called, it scans the bootstrapped points for the trigger criteria described by the DataLogType input setting. The only supported trigger mechanism for this class is DATA_CHANGE. If a change is detected based on that criteria, the information is formatted and written to a comma-separated value (CSV) file. The log is then written into the directory specified by DirectoryPath. A new log starts after the MaxFileSize is reached or at the beginning of a new day if StartNewLogPerDay is set to TRUE.

The number of files stored in the directory is dependent on two settings: MaxFolderSize and MaxNumDays. So long as the cumulative size of the stored logs does *not* exceed MaxFolderSize, the file system stores logs for MaxNumDays. If the cumulative number of logs exceeds MaxFolderSize, the directory manager deletes the oldest files until the cumulative log directory size is less than MaxFolderSize. File names begin with the start time of the log and are appended with *FilePostFix*.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable data recording.
RecordTimeInUtc	BOOL	Convert log time into UTC time. Information in the timestamp_t of the log source is used for this calculation.
DirectoryPath	STRING(127)	The full directory path at which files generated by this function block can be found.
FilePostfix	STRING (16)	The string which is appended after the time stamp from the file name.
MaxFolderSize	ULINT	The maximum size, in bytes, for the specified directory path.
MaxFileSize	UDINT	The maximum file size, in bytes, at which a new file will be created for subsequent logging operations.

Inputs

Name	IEC 61131 Type	Description
MaxNumDays	UDINT	The maximum number of days log files are allowed to persist in specified directory path.
StartNewFilePerDay	BOOL	If TRUE, a new log file will start at the beginning of a new day.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The logging block is enabled.
ActiveFileName	STRING(255)	Name of the file that logs are being written to.
ConsumedDirectorySize	ULINT	Size of all files in the monitored directory.
MonitoredPoints	UDINT	Total number monitored points.
Error	BOOL	Something has prevented this block from successfully writing data to file. This can indicate an out of space condition or that too many data points are being monitored for the CPU of the RTAC to handle.
ErrorDesc	STRING(255)	Message describing the source of the *Error* flag.

bootstrap_MonitorBCR (Method)

A configuration method for adding BCR points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the BCR point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	BCR	The BCR point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the BCR point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorUDINT (Method)

A configuration method for adding UDINT points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the UDINT point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	UDINT	The UDINT point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the UDINT point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorCMV (Method)

A configuration method for adding CMV points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the CMV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	CMV	The CMV point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the CMV point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorREAL (Method)

A configuration method for adding REAL points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the REAL point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	REAL	The REAL point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the REAL point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorDPS (Method)

A configuration method for adding DPS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the DPS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	DPS	The DPS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the DPS point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorINS (Method)

A configuration method for adding INS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the INS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	INS	The INS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the INS point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorDINT (Method)

A configuration method for adding DINT points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the DINT point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	DINT	The DINT point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the DINT point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorMV (Method)

A configuration method for adding MV points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the MV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	MV	The MV point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the MV point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSPS (Method)

A configuration method for adding SPS points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the SPS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	SPS	The SPS point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the SPS point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorBOOL (Method)

A configuration method for adding BOOL points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the BOOL point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	BOOL	The BOOL point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the BOOL point was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSTR (Method)

A configuration method for adding STR points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the STR point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	STR	The STR point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the STR point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSTRING (Method)

A configuration method for adding STRING points to be monitored by a time-aligned CSV or SOE CSV object. This method will only add the STRING point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	Label describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	STRING	The STRING point to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the STRING point was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

Run (Method)

This method must be called every scan to ensure proper functionality, which includes scanning data points and writing files.

Processing

- The first time the method is called, it blocks all future bootstrap methods from adding new points.
- For DataLogType DATA_CHANGE, points are considered changed if the respective data status value has changed since the previous scan.
- Scan all data points. If they have changed, then prepare the points for logging.
- Write one line containing a time value with millisecond precision, time code, station name, device name, and the data status value.
- Manage the total size of the directory path folder and the number of files in the respective data path based on MaxDirectorySize.
- Start a new file if the active log file exceeds the maximum file size, if NewFilePerDay is TRUE at the start of a new day, or if project settings change.
- The logs will be maintained for MaxNumDays if the MaxDirectorySize is not exceeded.
- If the MaxDirectorySize is exceeded before MaxNumDays of logs is reached, the oldest file is deleted. This repeats until the cumulative directory size is less than the MaxDirectorySize.
- Start a new file if the active log file exceeds the maximum file size, if the project settings are changed, or at the beginning of a new day if StartNewFilePerDay is TRUE.
- The EN pin must be TRUE to scan and write new log entries.
- When EN is FALSE, no data point changes are detected.
- A rising edge on EN clears all pending logs so that only changes since EN toggling to TRUE will be written to future logs.
- Deleting metadata in the .RetainedState and .unsent files in the specified directory may result in inconsistent logging behavior.

class_ComtradeFloat32 (Function Block)

This class monitors and logs bootstrapped data points. Each time the Run method is called, it scans the bootstrapped points for the trigger criteria described by the DataLogType input setting. If a change is detected based on that criteria, the information is formatted and written to a COMTRADE record, which is composed of a CFG and DAT file. The log is then written into the directory specified by DirectoryPath. A subfolder is created that has the date and time of the record to the millisecond, as well as *FilePostFix*. The date, time, and postfix are separated by the "," character. A new log will start after MaxFileSize is reached.

The number of files stored in the directory is dependent on two settings: MaxFolderSize and MaxNumDays. So long as the cumulative size of the stored logs does *not* exceed MaxFolderSize, the file system will store logs for MaxNumDays. If the cumulative number of logs exceeds MaxFolderSize, the directory manager will delete one day of records until the cumulative log directory size is less than MaxFolderSize. File names will be in the same format as the subfolder with either a CFG or DAT extension appended. Two additional folders may be present in the directory. Do not delete or alter the contents of these folders. Doing so may cause errors in the logging object.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable data recording.
RecordTimeInUtc	BOOL	Convert log time into UTC time. Information in the timestamp_t of the log source is used for this calculation.
RefTime	timestamp_t	Provides a time stamp reference for log creation, ignored unless DataLogType TIME_CHANGE trigger.
TimeVariance	UDINT	Time window in milliseconds applied to the time reference. If a change in the dateTime_t structure for a bootstrapped point is detected and within the time window it is included in the log entry. Set to 0 to disable. Ignored unless DataLogType TIME_-CHANGE trigger.
DataLogType	enum_DataLogger	Specifies which trigger mechanism is used to create logs.
DirectoryPath	STRING(50)	The full directory path at which files generated by this function block can be found.
FilePostfix	STRING(50)	The string that is appended after the time stamp from the file name.
MaxFolderSize	ULINT	The maximum size, in bytes, for the specified directory path.
MaxFileSize	UDINT	The maximum file size, in bytes, at which a new file will be created for subsequent logging operations. Defaults to 10 MB for TRIGGER_EVENT records.
MaxNumDays	UDINT	The maximum number of days log files are allowed to persist in specified directory path.
LoggingInterval	TIME	Time in milliseconds for cyclic recording; ignored unless DataLogType PERIODIC trigger.
TriggerSignal	BOOL	Monitored variable that initiates a record; only for DataLogType TRIGGER_EVENT.
PreTriggerCycles	UDINT	Number of processing cycles prior to the trigger event for which data will be recorded; ignored unless DataLogType TRIGGER_EVENT trigger. Minimum 1 pre-trigger cycle is required. If configured for less than the minimum, the setting will default to 1.
PostTriggerCycles	UDINT	Number of processing cycles post-trigger event for which data will be recorded; ignored unless DataLogType TRIGGER_EVENT trigger. Minimum 1 post-trigger cycle is required. If configured for less than the minimum, the setting will default to 1.
SystemFrequency	DINT	Primary frequency of the analog signal measured by the COMTRADE oscillography.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The logging block is enabled.
ActiveFileName	STRING(255)	Name of the file that logs are being written to.
ConsumedDirectorySize	ULINT	Size of all files in the monitored directory.
AnalogPoints	UDINT	Total number of analog points.
DigitalPoints	UDINT	Total number of digital points.
Error	BOOL	Something has prevented this block from successfully writing data to file. This can indicate an out of space condition or that too many data points are being monitored for the CPU of the RTAC to handle.
ErrorDesc	STRING(255)	Message describing the source of the *Error* flag.

bootstrap_MonitorCMV (Method)

A configuration method for adding CMV points to be monitored by a COMTRADE object. This method will only add the CMV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing the phase identifier for the data.
component	STRING(16)	The description to use when describing the physical component that is represented by the data.
unit	STRING(16)	The description to use when describing the engineering units to associate with the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	CMV	The CMV to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the CMV was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorREAL (Method)

A configuration method for adding REAL points to be monitored by a COMTRADE object. This method will only add the REAL point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing the phase identifier for the data.
component	STRING(16)	The description to use when describing the physical component that is represented by the data.
unit	STRING(16)	The description to use when describing the engineering units to associate with the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	REAL	The REAL to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the REAL was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorMV (Method)

A configuration method for adding MV points to be monitored by a COMTRADE object. This method will only add the MV point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing the phase identifier for the data.
component	STRING(16)	The description to use when describing the physical component that is represented by the data.
unit	STRING(16)	The description to use when describing the engineering units to associate with the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	MV	The MV to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the MV was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorINS (Method)

A configuration method for adding INS points to be monitored by a COMTRADE object. This method will only add the INS point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing the phase identifier for the data.
component	STRING(16)	The description to use when describing the physical component that is represented by the data.
unit	STRING(16)	The description to use when describing the engineering units to associate with the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	INS	The INS to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the INS was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorDINT (Method)

A configuration method for adding DINT points to be monitored by a COMTRADE object. This method will only add the DINT point to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing the phase identifier for the data.
component	STRING(16)	The description to use when describing the physical component that is represented by the data.
unit	STRING(16)	The description to use when describing the engineering units to associate with the data.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	DINT	The DINT to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the DINT was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorSPS (Method)

A configuration method for adding SPS points to be monitored by a COMTRADE object. This method will only add the SPS to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing this data in files generated by the data recorder class.
component	STRING(16)	The description to use when describing this data in files generated by the data recorder class.
defaultState	BOOL	The default state.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	SPS	The SPS to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the SPS was added for logging.

Processing

- ➤ Store a reference to the point and the associated metadata for future logging and comparison.
- ➤ Return true if the point will be monitored moving forward, false otherwise.

bootstrap_MonitorBOOL (Method)

A configuration method for adding BOOL points to be monitored by a COMTRADE object. This method will only add the BOOL to be monitored if called before calling the function block itself.

Inputs

Name	IEC 61131 Type	Description
channel	STRING(62)	The description to use when describing the data source.
station	STRING(62)	The description to use when describing the grouping of the data.
ph	STRING(1)	The description to use when describing this data in files generated by the data recorder class.
component	STRING(16)	The description to use when describing this data in files generated by the data recorder class.
defaultState	BOOL	The default state.

Inputs/Outputs

Name	IEC 61131 Type	Description
data	BOOL	The BOOL to be included in the log files.

Return Value

IEC 61131 Type	Description
BOOL	TRUE if the BOOL was added for logging.

Processing

- Store a reference to the point and the associated metadata for future logging and comparison.
- Return true if the point will be monitored moving forward, false otherwise.

Run (Method)

This method must be called every scan to ensure proper functionality, which includes scanning data points and writing files.

Processing

- The first time the method is called, it blocks all future bootstrap methods from adding new points.
- For DataLogType TIME_CHANGE, points are considered changed if the dateTime_t data structure has changed since the previous scan and the change falls within the window, relative to the time reference, specified by TimeVariance. Changes in simple type values do not trigger a log but are included in every log.
- For DataLogType PERIODIC, all points are considered changed if LoggingInterval is reached.
- For DataLogType TRIGGER_EVENT, all points are considered changed when logging points during the pre- and post-trigger cycles.

- Scan all data points. If they have changed, prepare them for logging.
- Write log entry containing the sample number and a time offset from the beginning of the log with millisecond precision and the bootstrapped data. Log entries are stored in the binary DAT file in the FLOAT32 format.
- The log entries and the CFG file information are written to a temporary file until the log is completed. Once completed, the log is stored as a CFG and DAT file in a subdirectory that has the same naming convention as that of the CFG and DAT file.
- Manage the total size of the directory path folder and the number of files in the respective data path based on the MaxDirectorySize.
- Start a new file if the active log file exceeds the maximum file size or if project settings change or if trigger condition is detected.
- The logs will be maintained for the MaxNumDays if the MaxDirectorySize is not exceeded.
- If MaxDirectorySize is exceeded before MaxNumDays of logs is reached, files are deleted in day increments until the cumulative size of the logs is less than MaxDirectorySize.
- The EN pin must be TRUE to scan and write new log entries.
- When EN is FALSE, no data point changes are detected.
- A rising edge on EN clears all pending logs so that only changes since the EN toggling to true will be written to future logs.
- Deleting metadata in the TEMP and SHADOW directories in the specified directory may result in inconsistent logging behavior.

Benchmarks

Benchmark Platforms

The benchmarking tests recorded for this library are performed on the following platforms.

- SEL-3505
  - R134-V1 firmware
- SEL-3530
  - R134-V1 firmware
- SEL-3555
  - Dual-core Intel i7-3555LE processor
  - 4 GB ECC RAM
  - R139 firmware

Benchmark Test Descriptions

class_TimeAlignedCSVManager Performance tests

- **Time-Aligned CSV TIME_CHANGE trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and changes the time stamps of these data points ten times per second. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here.
- **Time-Aligned CSV PERIODIC trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and sets the period to 100 ms. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here.
- **Time-Aligned CSV TRIGGERED trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and triggers once per second. The block is configured to store four pre-trigger cycles and five post-trigger cycles. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here.

class_SoeCsv Performance tests

- **Per-Point CSV DATA_CHANGE trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and changes the data of these points ten times per second. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here.

class_Float32COMTRADE Performance tests

- **COMTRADE TIME_CHANGE trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and changes the time stamps of these data points ten times per second. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here. This test waits for the first transition from temporary data to CFG and DAT files to commence before running.
- **COMTRADE PERIODIC trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and sets the period to 100 ms. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here. This test waits for the first transition from temporary data to CFG and DAT files to commence before running.
- **COMTRADE TRIGGERED trigger**: This test instantiates one manager class, bootstraps it with two data points of each type, and triggers once per second. The block is configured to store four pre-trigger cycles and five post-trigger cycles. The RTAC has a task cycle time of 10 ms and the time required for the `Run()` method is recorded. The average, maximum, minimum, and standard deviation are recorded here.

Benchmark Results

Operation Tested		Platform (time in μs)		
		SEL-3505	SEL-3530	SEL-3555
class_SoeCsv				
DATA_CHANGE	Average	2540.4	1374.7	54.2
	Maximum	10977.8	5904.3	321.1
	Minimum	380.3	251.7	8.1
	Std Deviation	2722.5	1366.7	55.9
class_Float32COMTRADE				
TIME_CHANGE	Average	314.6	204.9	16.7
	Maximum	4636.9	1909.5	297.4
	Minimum	102.1	76.5	3.8
	Std Deviation	548.5	272.1	19.6
PERIODIC	Average	276.9	171.9	12.7
	Maximum	4845.8	1943.9	202.0
	Minimum	66.3	32.2	2.0
	Std Deviation	509.7	283.2	18.6
TRIGGERED	Average	391.3	208.6	14.5
	Maximum	3025.8	1297.5	102.9
	Minimum	163.1	76.9	5.5
	Std Deviation	399.2	227.8	14.8
class_TimeAlignedCSVManager				
TIME_CHANGE	Average	1057.1	631.8	29.2
	Maximum	6555.3	3841.6	399.1
	Minimum	245.9	275.3	4.1
	Std Deviation	749.7	376.6	36.6
PERIODIC	Average	1032.3	574.5	24.3
	Maximum	6433.2	4206.8	398.1
	Minimum	88.5	65.5	2.6
	Std Deviation	734.2	369.9	35.2
TRIGGERED	Average	1564.2	819.3	29.5
	Maximum	5968.8	4456.6	341.6
	Minimum	1000.0	667.5	21.1
	Std Deviation	334.2	243.3	22.1

Examples

These examples demonstrate the capabilities of this library. Do not mistake them as suggestions or recommendations from SEL.

Implement the best practices of your organization when using these libraries. As the user of this library, you are responsible for ensuring correct implementation and verifying that the project using these libraries performs as expected.

Log Data Cyclically to a Time-Aligned CSV File

Objective

A user wishes to record three data points in a time-aligned CSV file format at a one second periodic basis and does not want any log to exceed 100 KB. The user wants to maintain the logs for a maximum of 10 days.

Assumptions

This example assumes that the data tags provided as inputs to the bootstrap methods exist in the project.

Solution

For a periodic logger, the user can create a program as shown in *Code Snippet 1*. The sample log output is shown in *Code Snippet 2*.

Code Snippet 1 prg_CyclicCsvLogger

```
PROGRAM prg_CyclicCsvLogger
VAR
    //Initialization variable
    INIT : BOOL;
    //Class initialization
    MyCSVCyclic : class_TimeAlignedCsv := (EN := TRUE,
        RecordTimeInUtc := FALSE,
        TimeVariance := 0,
        DataLogType := PERIODIC,
        DirectoryPath := 'CSV_Cyclic_1',
        FilePostfix := 'SubC.csv',
        MaxFolderSize := 500000000,
        MaxFileSize := 100000,
        MaxNumDays := 10,
        LoggingInterval := T#1S);
END_VAR
```

```
//Bootstrap the points for logging
IF NOT INIT THEN
    MyCSVCyclic.bootstrap_MonitorCMV(channel := 'Point_CMV0',
         station := 'PMU10', data := DataRecorderTags.Status_0000);
    MyCSVCyclic.bootstrap_MonitorINS(channel := 'Point_INS1',
        station := 'PMU10', data := DataRecorderTags.Status_0001);
    MyCSVCyclic.bootstrap_MonitorSPS(channel := 'Point_SPS3',
        station := 'PMU10', data := DataRecorderTags.Status_0002);

    INIT := TRUE;
END_IF

//It is required to call the Run method every scan
MyCSVCyclic.Run();
```

Code Snippet 2 Sample of a Time-Aligned Cyclic Log

```
   PMU10:Point_INS1,PMU10:Point_SPS3
2017/03/14 22:14:25.899,3172.46,12.1414,123456,1
2017/03/14 22:14:26.900,9.11965E11,3330840.0,654321,0
2017/03/14 22:14:27.901,3172.46,12.1414,123456,1
2017/03/14 22:14:28.902,9.11965E11,3330840.0,654321,0
2017/03/14 22:14:29.903,3172.46,12.1414,123456,1
2017/03/14 22:14:30.904,9.11965E11,3330840.0,654321,0
```

Log Data SOE data to a CSV File

Objective

A user wishes to monitor and record any data status changes to four data points in a SOE CSV file format and does not want any log to exceed 100 KB. The user wants to maintain the logs for a maximum of 30 days.

Assumptions

This example assumes that the data tags provided as inputs to the bootstrap methods exist in the project.

Solution

For an SOE logger, the user can create a program as shown in *Code Snippet 3*. The sample log output is shown in *Code Snippet 4*.

Code Snippet 3 prg_SoeCsvLogger

```
PROGRAM prg_SoeCsvLogger
VAR
    //Initialization variable
    INIT : BOOL;
        //Class initialization
    MySOE : class_SoeCsv := (EN := TRUE,
        RecordTimeInUtc := FALSE,
        DirectoryPath := 'SOE Events 1',
        FilePostfix := 'SubA.csv',
        MaxFolderSize := 500000000,
        MaxFileSize := 100000,
        MaxNumDays := 30,
        StartNewFilePerDay := FALSE);
END_VAR
```

```
//Bootstrap the points for logging
IF NOT INIT THEN
    MySOE.bootstrap_MonitorCMV(channel := 'Point_CMV0', station :=
        'Breaker1',
        data := DataRecorderTags.Status_0000);
    MySOE.bootstrap_MonitorINS(channel := 'Point_INS1', station :=
        'Breaker1',
        data := DataRecorderTags.Status_0001);
    MySOE.bootstrap_MonitorMV(channel := 'Point_MV2', station := 'Breaker1',
        data := DataRecorderTags.Status_0003);
    MySOE.bootstrap_MonitorSPS(channel := 'Point_SPS3', station :=
        'Breaker1',
        data := DataRecorderTags.Status_0002);
    INIT := TRUE;
END_IF

//It is required to call the Run method every scan
MySOE.Run();
```

Code Snippet 4 Sample of an SOE Log

```
Date,Time,Time Code,Station,Device,Value
03/15/2017,10:28:49.528,-7.0,Breaker1,Point_CMV0,9.11965E11@3330840.0
03/15/2017,10:28:49.528,-7.0,Breaker1,Point_INS1,654321
03/15/2017,10:28:49.528,-7.0,Breaker1,Point_MV2,2.99882E-38
03/15/2017,10:28:49.528,-7.0,Breaker1,Point_SPS3,0
03/15/2017,10:28:50.529,-7.0,Breaker1,Point_CMV0,3172.46@12.1414
03/15/2017,10:28:50.529,-7.0,Breaker1,Point_INS1,123456
03/15/2017,10:28:50.529,-7.0,Breaker1,Point_MV2,1.07596E33
03/15/2017,10:28:50.529,-7.0,Breaker1,Point_SPS3,1
```

Log Data to a COMTRADE File

Objective

A user wishes to monitor and record any data updates for two data points, as determined by the time stamp of the data point, in a COMTRADE file format. The user does not want any log to exceed 10 MB and wants to maintain the logs for a maximum of 10 days.

Assumptions

This example assumes that the data tags provided as inputs to the bootstrap methods exist in the project.

Solution

For a COMTRADE time change logger, the user can create a program as shown in *Code Snippet 5*. The sample log output is shown in *Code Snippet 6*.

Code Snippet 5 prgCOMTRADELogger

```
PROGRAM prg_COMTRADELogger
VAR
    //Initialization variable
    INIT : BOOL;
    //Class initialization
    MyCOMTRADETimeChange : class_ComtradeFloat32 := (EN := TRUE,
        RecordTimeInUtc := FALSE,
        TimeVariance := 0,
        DataLogType :=   TIME_CHANGE,
        DirectoryPath := 'CTRADE_Station1',
        FilePostfix := 'Bay1',
        MaxFolderSize := 500000000,
        MaxFileSize := 1000000,
        MaxNumDays := 10,
        SystemFrequency := 60);
END_VAR
```

Code Snippet 5 prgCOMTRADELogger (Continued)

```
//Bootstrap the points for logging
IF NOT INIT THEN
    MyCOMTRADETimeChange.bootstrap_MonitorCMV(channel := 'Point_CMV0',
        station := 'PMU10', ph := 'A',
    Component := 'Breaker1', Unit := 'V', data :=
        DataRecorderTags.Status_0000);
    MyCOMTRADETimeChange.bootstrap_MonitorSPS(channel := 'Point_SPS39',
        station := 'PMU10', ph := 'A',
    Component := 'Breaker1', defaultState := FALSE, data :=
        DataRecorderTags.Status_0002);
    INIT := TRUE;
END_IF

//It is required to call the Run method every scan
MyCOMTRADETimeChange.RUN();
```

Code Snippet 6 Sample of a COMTRADE 2013 CFG File

```
Bay1,RTAC_Archive,2013
5,4A,1D
1,PMU10:Point_CMV0_m,Am,Breaker1,V,1,0,0,-3.4028235E38,3.4028235E38,1,1,P
2,PMU10:Point_CMV0_a,Aa,Breaker1,V,1,0,0,-3.4028235E38,3.4028235E38,1,1,P
1,PMU10:Point_SPS39,A,Breaker1,0
60
1
1,40
15/03/2017,11:51:37.674722
15/03/2017,11:51:37.674722
float32
1000
-7,-7
F,3
```

Release Notes

Version	Summary of Revisions	Date Code
3.5.2.0	➤ Adds support for simple data types. ➤ Increases maximum triggered COMTRADE file size from 1 MB to 10 MB. ➤ Extends COMTRADE pre- and post-trigger sample counts to 18,000. ➤ Updates class_ComtradeFloat32 and class_TimeAlignedCsv to allow the LoggingInterval setting to change during runtime. ➤ COMTRADE files are now compressed to reduce file size. ➤ Must be used with R144 firmware or later.	20190218
3.5.1.0	➤ Allows new versions of ACSELERATOR RTAC to compile projects for previous firmware versions without SEL IEC types "Cannot convert" messages. ➤ Must be used with R143 firmware or later.	20180921
3.5.0.0	➤ Initial release of class_ComtradeFloat32. ➤ Initial release of class_SoeClass. ➤ Initial release of class_TimeAlignedCsv.	20170601