ETAS COSYM V3.4.1

Python Bindings for TargetClient API

SiL Python API Guide

www.etas.com

COSYM V3.4.1 I SiL Python API Guide R01 EN - 04.2024

Contents

1 About this Document

1.1 Classification of Safety Messages

Safety messages warn of dangers that can lead to personal injury or damage to property:

DANGER

DANGER indicates a hazardous situation that, if not avoided, will result in death or serious injury.

WARNING

WARNING indicates a hazardous situation that, if not avoided, could result in death or serious injury.

CAUTION

CAUTION indicates a hazardous situation that, if not avoided, could result in minor or moderate injury.

NOTICE

NOTICE indicates a situation that, if not avoided, could result in damage to property.

1.2 Demands on Technical State of the Product

The following special requirements are made to ensure safe operation:

– Take all information on environmental conditions into consideration before setup and operation (see the documentation of your computer, hardware, etc.).

Further information, refer to Safety advice document which is embedded in the COSYM V3.4.1

2 Introduction

The Python Bindings for TargetClient API provide functions to control the simulation run ("Parameterization APIs" on page 15) without the help of ETAS Experiment Environment.

These functions are applicable only for the SiL ATS use case.

3 System Requirements

It is necessary to install the packages listed below to run the TargetClient APIs of Python.

- **COSYM V3.4.1**
- **Python 3.7** (64-bit version recommended)
- Python is an open source software and can be downloaded from the link; https://www.python.org/downloads/
  - Choose Python 3.7.1
  - Download the Windows x86-64 executable installer.
  - The path should be added to the PATH environment variable. While installing, select the checkbox to add the installation path to PATH environment variable.
- **Pip**
  Pip is a tool which helps to install libraries associated with Python:
  - Open the link "https://bootstrap.pypa.io/get-pip.py"
  - Right click and save the file named '`get-pip.py`' in your local system.
  - Open a command prompt and go to the directory where '`get-pip.py`' file is saved (using "cd <directory path name>").
  - Run the script : "`python get-pip.py`"
  - Run the command : "`pip --version`"
    This will show the installed pip version, if installation is successful.
- **NumPy**
  - Open a command prompt and execute the command below:
    "`pip install numpy`"
- **Avro**
  - Open a command prompt and execute the command below:
    "`pip install avro-python3`"

Note

- If Pip or NumPy installation failed because of the connection error, please verify your proxy settings, enable the correct proxy and try again.
- If your system is connected to VPN and facing the installation issue, then try installing it after disconnecting the VPN.

4 Files in COSYM Installation Folder

The required files are available at <COSYM installation directory>`Simulator\win64\atssimulator\PythonBindings` folder.

File names	Description
`TargetClientPythonAPI.pyd`	Binary generated with `TargetClientPythonAPI.py`
`TargetClientPythonAPI.py`	Contains all the interfaces exposed from `targetClient.dll`. You have to import this file to access required functionalities.

5 Accessing the TargetClient API

In the `TestTCPythonAPI.py` file, an example is shown on how to access the `TargetClientPythonAPI.py` file.

The example file is provided in the COSYM demo data location. The demo data location is the location specified during COSYM installation. The default location is `C:\ETAS\ETASData\COSYM3.4`. In case you have selected a different location for demo data, then the files\folders in the demo data can be found in the location specified by you. Example file is adapted to `3CylATS_Sim` project which is available at the demo data location under `Samples\SiL` folder.

It is necessary to build the project once you open it in the corresponding COSYM version. The build process creates 2 zip files which are listed below. The files folders are required to call the SiL APIs.

- `deployablesWindows.zip`
- `configFiles.zip`

<u>To access the TargetClient API</u>

1. Unzip the `Samples.zip` file which is available in the COSYM demo data. An example file, `TestTCPythonAPI.py` is available in the unzipped `Samples\PythonScripts` folder.
2. Add <COSYM Installation location>`\Simulator\win64\atssimulator\PythonBindings` to 'PYTHONPATH' variable.
3. Open a command prompt and execute below command lines:
   i. Run the below batch file to setup System Environment:
   <COSYM demo data location>`\Samples\PythonScripts\PyEnvSettings.bat`

   ii. Execute the script with below command:
   `python TestTCPythonAPI.py` <Demo data location>`\Samples\SiL\3CylATS_Sim\Systems\SystemATS\Configuration\OS\codegen\deployablesWindows.zip` <Demo data location>`\Samples\SiL\3CylATS_Sim\Systems\SystemATS\Configuration\OS\codegen\configFiles.zip`

   iii. If the script is modified according to any other COSYM project, then generic command to execute the script:
   `python TestTCPythonAPI.py` <COSYM project folder>`\Systems\<System name>\Configuration\OS\codegen\deployablesWindows.zip` <COSYM project folder>`\Systems\<System name>\Configuration\OS\codegen\configFiles.zip`

6 Functions Exposed from `TargetClientPythonAPI.py`

This chapter contains the information about exposed functions, function parameter details, and simulation state mapping.

6.1 Control APIs

Python Interfaces	Functionalities	Input parameters	Returns
GetTargetSessionAPI(ipAddress,userName,passWord)	Establish the connection	ipAddress: 'localhost' or any required IP address userName: Empty ' ' for localhost or corresponding user name passWord: Empty ' ' for localhost or corresponding password	-1 : for SIL case
DeployAPI(targetSessionId,source,type)	Downloads the simulation	targetSessionId: Return value from GetTargetSessionAPI(..) API call source: source project assignment type: type of simulation platform (e.g. SIL)	0- Success / ErrorCode- Failure
UnDeploy()	Closes the simulation and deletes all the running instances related to simulation	None	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
StartSimulation(defaultEndTime=-1)	Starts the simulation for given time (adaptive time). Once time is over, simulation will be stopped automatically	Simulation End Time (default -1)	0- Succes/ ErrorCode- Failure
StopSimulation()	Stops the simulation	None	0- Succes/ ErrorCode- Failure
PauseSimulation()	Pauses the simulation	None	0- Succes/ ErrorCode- Failure
ResumeSimulation()	Resumes the simulation	None	0- Succes/ ErrorCode- Failure
SetSoftRTDelayRatio(delayRatio=0)	Sets the delay ration for soft real time	Delay Ratio (default 0)	0- Succes/ ErrorCode- Failure
GetCurrentSimulationState()	Returns current simulation state	None	First element : 0- Success / ErrorCode- Failure Second element : Simulation State
GetSimulationTime()	Returns current simulation time	None	First element : 0- Success / ErrorCode- Failure Second element : Simulation Time

Python Interfaces	Functionalities	Input parameters	Returns
GetComputationTime()	Returns current computation time	None	First element : 0- Success / ErrorCode- Failure Second element : Computation Time
GetSimulationStepSize()	Returns simulation step size i.e., HCF of task rasters	None	Simulation Step Size: float step size is in seconds

GetPerformanceStatisticsAPI(process_type)	Returns performance statistics information as a list of performance_signal object based on the process type	Process Type can be the following
		– TargetClientPythonAPI.e_download - Download
		– TargetClientPythonAPI.e_init - Init
		– TargetClientPythonAPI.e_exit - Exit
		– TargetClientPythonAPI.e_timer - Timer tasks
		– TargetClientPythonAPI.e_all - All
			list of performance signals
			`struct performance_signal`
			`{`
			`    char signal_path[1024];`
			`    double time_taken;   // in seconds`
			`    double percentage;`
			`};`

6.2 Parameterization APIs

Python Interfaces	Functionalities	Input parameters	Returns
WriteScalarParameter (variableAddress, scalarValue, baseDataType)	Writes a scalar parameter (write only works with parameters)	variableAddress, scalar value to be written, data type of scalar	First element : 0- Success / ErrorCode- Failure Second element : 0- Success / ErrorCode- Failure
ReadScalarVariable(variableAddress, baseDataType)	Reads a scalar variable (any kind of variable, inports outports, parameters, measurements)	variableAddress, data type of scalar	First element : 0- Success / ErrorCode- Failure Second element: Scalar value Third element : 0- Success / ErrorCode- Failure
WriteVectorParameter (variableAddress, data, baseDataType)	Writes a vector parameter(only works with parameter)	variable address, data: input vector (it should be always one dimensional and because of that we call it vector) better to be a numpy array otherwise a python list, data type	First element : 0- Success / ErrorCode- Failure Second element : 0- Success / ErrorCode- Failure
ReadVectorVariable(variableAddress, baseDataType, size, offset=0)	Reads the vector variable in variable address from offset with input size	variable address, base data type of vector elements, size: number of elements to read, offset: starting offset to read from	First element : 0- Success/ErrorCode- Failure Second element : 0- Success/ErrorCode- Failure Third element : vector value in numpy array format

Python Interfaces	Functionalities	Input parameters	Returns
WriteMultiParameter (varNames, baseTypes, values)	Writes multiple parameter in one call	varNames: list of variable addresses baseTypes: list of variable base types in same order of variable names values: list of values to write to the variables in same order of variable names	First element : 0- Success / ErrorCode- Failure Second element : 0- Success / ErrorCode- Failure
ReadMultipleParameter (varNames, baseTypes, sizes)	Read multiple parameters in one call	varNames: list of variable addresses baseTypes: list of variable base types in the same order of variable names sizes: list of variable sizes in the same order of variable names	First element : 0- Success / ErrorCode- Failure Second element : key:value dictionary (key: variable address, value: scal-ar/vector value)
WriteString	Writes a string to the specified address	Variable Address: str data: str	tuple: contains SIL_RESULT (0 - Suc-cess), ERROR_CODE
ReadString	Reads a string from the specified address	Variable Address: str size: int	tuple: contains SIL_RESULT (0 - Suc-cess), ERROR_CODE, string

6.3 Signal Injection APIs

Python Interfaces	Functionalities	Input parameters	Returns
CreateSGInstance(msg)	Creates Signal Generator Instance	createGeneratorInstance_t object: – taskIndex Unique identifier for the signal generator instance. You can start the task index from 0 for the first instance. Added to the list of task table. When we are calling back via SignalGeneratorAPI sgHook(), we have to make use of this task Index. – startTime Recommended to provide start time as zero or positive number. – channelReset We can provide 3 inputs for this 0 – reset signal outport value to 0. 1 – reset signal outport value to start value 2 – reset signal outport value to existing value	First element : 0- Success / ErrorCode- Failure Second element : Signal Id
DeleteSGInstance(sgId)	Deletes Signal Generator instance for given id.	Signal generator id.	0- Success / ErrorCode- Failure
DeleteAllSGInstances()	Deletes all available Signal Generator instances.	None	0- Success / ErrorCode- Failure
StartSGInstance(sgId)	Starts Signal Generator instance for given id.	Signal generator id.	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
StopSGInstance(sgId)	Stops Signal Generator instance for given id.	Signal generator id.	0- Success / ErrorCode- Failure
ResetSGInstance(sgId)	Resets Signal Generator instance for given id.	Signal generator id.	0- Success / ErrorCode- Failure
CreateSGChannel(obj)	Creates Signal Generator channel.	createChannel_t object	First element : 0- Success / ErrorCode- Failure Second element : Channel Id
DeleteSGChannel(obj)	Deletes Signal Generator channel.	deleteChannel_t object	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
SetSGTimeslice(obj)	Sets time slice to signal generator.	setTimeslice_t object: Before starting simulation, we generate a blue print, which tells about how much time each signal should run and which signal should be the next in line. Time slice provides the timing details about the signal like start and end time and number of iterations. Each signal interval is associated with timeslice. If a channel has more than one intervals, then we need more timeslices. Signal interval is fetched based on signal ID provided. – createFlg If create flag is 1 then create new time slice else update existing time slice – sliceNo Indicates position where the timeslice has to be inserted in the existing timeslice list – deltaT End time of a signal interval shall be calculated by adding this delta time value with start time value – startT Start time of a signal interval. Must be greater than zero. If set to 1 then the start time of the signal will be from 1st second – countMax	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
		Denotes maximum repetition of the signal interval in signal generator life cycle.	
DeleteSGTimeslice(obj)	Deletes the time slice.	deleteTimeslice_t object	0- Success / ErrorCode- Failure
SetLinearSignal(obj)	Sets linear signal.	setLinearSignal_t object Range of double data type – createFlg – channelNo – intervalNo – countMax – deltaT – startY – endY	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
SetDataSignal(obj)	Sets Data signal.	setDataSignal_t object – channelNo Indicates specific channel number where the signal needs to be placed – intervalNo Represents data segment interval number – createFlg If create flag is 1 then create new signal interval else update existing signal interval – countMax Denotes maximum repetition of the segment. – nofSamples – intpolType	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
SetSinewaveSignal(obj)	Sets sine wave signal.	setSinewaveSignal_t object Range of double data type – createFlg – channelNo – intervalNo – countMax – deltaT – Frequency – Phase – Amplitude – Offset	0- Success / ErrorCode- Failure
SetExponentialSignal(Obj)	Sets exponential signal.	setExpwaveSignal_t object Range of double data type – createFlg – channelNo – intervalNo – countMax – deltaT – Amplitude – Tau – Offset	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
SetPulseSignal(obj)	Sets pulse signal.	setPulseSignal_t object Range of double data type – countMax – deltaT – Frequency – Amplitude – Offset – dutyCycle	0- Success / ErrorCode- Failure
SetIdleSignal(Obj)	Sets idle signal.	setIdleSignal_t object Range of double data type – countMax – deltaT	0- Success / ErrorCode- Failure
SetRandomSignal(obj)	Sets random signal.	setRandomSignal_t object Range of double data type – countMax – deltaT – Amplitude – Offset	0- Success / ErrorCode- Failure
SetOperationSignal(obj)	Set operation signal	setOperationSignal_t object	0- Success / ErrorCode- Failure
DeleteInterval(obj)	Deletes the interval.	deleteInterval_t object	0- Success / ErrorCode- Failure

Python Interfaces	Functionalities	Input parameters	Returns
GetSGTime(sgId)	gets Signal Generator time.	Signal generator id.	First element : 0- Success/ ErrorCode- Failure Second element : SG Time

LinkPortWithModeAPI(mod-elAddress,
sgoid,eMode,constValue)
	Links SignalGenerator port for given id and
	selected model
		Model Address , signal generator id , value for mode and
		desired constant value.
		Note:
		MD_CONST=0
		MD_MODEL=1
		MD_MODEL_PLUS_CONST=2
		MD_MODEL_MULT_CONST=3
		MD_MODEL_PLUS_SIGNALGENERATOR=4
		MD_MODEL_MULT_SIGNALGENERATOR=5
		MD_SIGNALGENERATOR=6
		MD_SIGNALGENERATOR_PLUS_CONST=7
		MD_SIGNALGENERATOR_MULT_CONST=8
			0- Success / ErrorCode- Failure

SetDataSignalDoublePtr (arrayObj)	Validate and convert a double array to C++ double pointer	Array of double elements	Validated C++ double pointer
FreeMemory(dataSig-nalObject)	Frees memory for signal gen-erator when SetDataSignal API is used	Data Signal object	void

6.4 Data Logger APIs

Python Interfaces	Functionalities	Input parameters	Returns
RegisterDataLogger(channelid, mod-elAddressDict,loggingAddress, dataRate-e=.1)	Registers given channel for data logging.	Channel id, modelAddressDict: Model Address dictionary or 'all' for all signal type, Filename where to be logged and data rate.	0- Success / ErrorCode-Failure
StartLoggingOnChannel(chid)	Starts logging on given channel.	Channel id.	0- Success / ErrorCode-Failure
StopLoggingOnChannel(chid)	Stops logging on given channel.	Channel id.	0- Success / ErrorCode-Failure
CloseAllDataChannels()	Closes all open channels and stop the server	None	0- Success / ErrorCode-Failure
RemoveDataChannel(chid)	Removes the data channel instance of given channel id.	Channel id.	0- Success /ErrorCode-Failure

6.5 Error Info APIs

Python Interfaces	Functionalities	Input parameters	Returns
ErrorDescription(errorCode))	Gets error description for given error code	Error code to be given	Error Message

6.6 SiL Driver APIs

In the SiL Driver APIs, the following classes are available.

- ScalarBuffer class
- ModelLoader class
- ExternalModel class
- SignalGroup class
- SimHandle class

Python API	Description	Parameters	Return Value
convert_cdata	Converts the C data to Python and returns it.	1) data: address/ c void pointer data received in write callback 2) baseDataType: enum type received in write callback 3) isArray: bool true for arrays	python data: can be int/float/bool etc.
get_scalar_buffer	Returns an instance of ScalarBuffer class(which creates and manages a buffer) of specified type. Use ScalarBuffer instance to call set_value()/get_value() to write to/read from buffer.	baseDataType: enum this type represents C/C++ fundamental types	ScalarBuffer instance

6.6.1 ScalarBuffer Class

Manages a buffer of scalar type

Python API	Description	Parameters	Return Value
set_value	writes value to buffer	value of buffer type	none
get_value	reads value from buffer	none	value of buffer type

6.6.2 set_import_timeout and get_model_loader APIs

Python API	Description	Parameters	Return Value
set_import_timeout	Sets timeout in milliseconds for loading exported module data from SiL	time_ms: int time in milliseconds	0 on Success, ErrorCode on Failure
get_model_loader	Gets ModelLoader instance	loader key/name: str	ModelLoader instance

6.6.3 ModelLoader Class

ModelLoader represents a loader (process) in COSYM simulation. Every Loader can contain multiple model instances and should run in a separate process. The loader parses `ModelLoader.config` and identifies the models that needs to be loaded.

The loader parses `SimulationConfiguration.bin` and identifies the location of the models along with their architecture and the scheduling configuration of the models. The loader then invokes the adapter to load the models.

This class exposes following functionalities. You can get model name, number of models, simulation status, loader status, run loader in async/sync mode, reset loader etc.

Python API	Description	Parameters	Return Value
close_sil_loader	Resets loader instance	none	0 on Success, ErrorCode on Failure
is_loader_running	Gets model loader status	none	loader status: int
is_simulation_running	Gets simulation status	none	simulation status: int
run_loader_async	Runs model loader in async state. Simulation loop runs in background till terminated/disconnected.	none	0 on Success, ErrorCode on Failure
run_loader_sync	Runs model loader in sync/blocking mode, till simulation is terminated/disconnected.	none	0 on Success, ErrorCode on Failure
get_number_of_models	Gets number of models from the loader	none	number of models: int
get_model_name	Gets model name using model index	model index: int	model name: str
get_model_by_name	Gets the ExternalModel instance using name	model name: str	ExternalModel instance
get_model_by_index	Gets the ExternalModel instance using index	model index: int	ExternalModel instance

6.6.4 ExternalModel Class

Externall Model represents a model in COSYM system that is configured to run externally. This model will be run by the COSYM in every step and exposes the following functionalities.

You can set callbacks, to be called during init, exit, step, read, and write operations. And also, you can set buffer for variables, get last simulation time, number of sync tasks, task index rate, variable id, signal group, etc.

Python API	Description	Parameters	Return Value
get_last_simulation_time	Gets async last simulation time in microseconds.	none	time: int simulation time in microseconds
get_number_of_sync_tasks	Gets number of sync tasks.	none	number of sync tasks: int
get_task_index_rate	Gets task index rate.	task index: int	task index rate: int
set_init_callback	Sets init callback, which will be called during start simulation.	function with following signature fn()	0 on Success, ErrorCode on Failure
set_exit_callback	Sets exit callback, which will be called during stop simulation.	function with following signature fn()	0 on Success, ErrorCode on Failure
set_step_callback	Sets step callback, which will be called during model step operation. function_name will be received as python bytes object, hence need to be decoded like following. name = function_name.decode()	function with following signature fn(function_name, simulation_time, raster_us)	0 on Success, ErrorCode on Failure

Python API	Description	Parameters	Return Value
set_write_callback	Sets variable set callback, which will be called during write operation. variable_name will be received as python bytes object, hence need to be decoded like following. name = variable_name.-decode() Also, data received needs to be converted to python type using convert_cdata() API.	function with following signature fn(variable_name, datatype, data, size)	0 on Success, ErrorCode on Failure
set_read_callback	Sets variable get callback, which will be called during read operation. variable_name will be received as python bytes object, hence need to be decoded like following. name = variable_name.-decode()	function with following signature fn(variable_name, datatype, data, size)	0 on Success, ErrorCode on Failure
get_variable_id	Gets variable id using name.	variable name: str	variable id: int

Python API	Description	Parameters	Return Value
set_variable_buffer	Sets buffer for variable using variable id/variable name.	1) variable: str/int variable name/variable id 2) buffer: int/ float/ numpy array of int/float For arrays, pass a variable initialized with numpy array.Later, this buffer can be accessed like regular array(eg: buffer[0]) For scalar types pass 0 or temp variable. 3) buffer_instance: ScalarBuffer instance This can be acquired using get_ scalar_buffer(). This is optional for arrays.	0 on Success, ErrorCode on Failure
get_signal_group	Gets SignalGroup instance for inport/out-port	filter: int constant set SGInport for inports or SGOutport for outports	SignalGroup instance

6.6.5 SignalGroup Class

SignalGroup class represents either Inports or Outports group in a Model.

You can get signal name, signal index, signal data type, signal size, set buffer for signals, get signal buffer, get number of signals, etc.

Python API	Description	Parameters	Return Value
get_number_of_signals	Gets number of signals	none	number of signals: int
get_signal_index	Gets signal index using signal name.	signal name: str	signal index: int
get_signal_name	Gets signal name using signal index.	signal index: int	signal name: str
get_signal_size	Gets signal size using signal index.	signal index: int	signal size: int
get_signal_type	Gets signal type using signal index.	signal index: int	signal type: enum this enum represents C/C++ fundamental types
get_signal_buffer	Gets signal buffer using signal index.	signal index: int	signal buffer: address/ c void pointer
set_signal_buffer	Sets buffer for signal using signal id/signal name.	1) signal: int/str signal name/signal index 2) buffer_instance: ScalarBuffer instance this can be acquired using get_scalar_buf-fer()	0 on Success, ErrorCode on Failure

6.6.6 Connect to Running SiL

Python API	Description	Parameters	Return Value
connect_to_running_sil	Connects to running simulation from third party client	session Id: int	0 on Success, ErrorCode on Failure
get_sim_handle	Gets SimHandle instance	none	SimHandle instance
get_sim_handle_offline	Gets SimHandle instance Offline	deployables unzippath: str	SimHandle instance

6.6.7 SimHandle Class

With SimHandle class, you can retrieve information of all available inports/outports/parameters/measurement variables in a system.

Python API	Description	Parameters	Return Value
get_number_of_vari-ables	Gets number of variables	variable type: enum any of VarInport/VarOutport/VarMeasurment/VarParam	number of variables: int
get_variable_info	Gets an updated instance of VariableInfo using index.	1) variable type: enum any of VarInport/VarOutport/VarMeasurment/VarParam 2) variable index: int	VariableInfo instance
find_variable_info	Gets an updated instance of VariableInfo using complete path of variable.	complete path of variable: str	VariableInfo instance

Python API	Description	Parameters	Return Value
find_first_variable	Gets an updated instance of VariableInfo using subpath of variable.	1) subpath of variable: str 2) model_name: str	VariableInfo instance

Note

The 'docstrings' are added for all of the above SiL Driver APIs and classes.
You can get documentation of the same using print (<API>.__doc__)

6.7 Watcher, Capture, and Breakpoint APIs

Python Interfaces	Input parameters	Functionalities	Returns
CreateBreakpointAPI(watcher_descID, state)	– watcher descriptor id – simulation state	Create Breakpoint and return breakpoint descriptor id	Unique breakpoint descriptor id
CreateConditionWatcherAPI(watcher)	– watcher	Gets watcher descriptor id	Unique watcher descriptor id
CreateDurationWatcherByStepAPI(step)	– step	Gets duration watcher descriptor id for given step	Unique duration watcher by steps descriptor id

Python Interfaces	Input parameters	Functionalities	Returns
CreateDurationWatcherByTimeAPI(time)	– time	Gets duration watcher descriptor id for given time	Unique duration watcher by time descriptor id
DeleteBreakpoint(breakpoint_descID)	– break point descriptor id	Deletes specified breakpoint	0- Success / ErrorCode- Failure
DeleteConditionWatcher(watcher_descID)	– watcher descriptor id	Deletes specified condition watcher	0- Success / ErrorCode- Failure
DeleteDurationWatcher(watcher_descID)	– watcher descriptor id	Deletes specified duration watcher	0- Success / ErrorCode- Failure
GetBreakpointAPI()	– NIL	Gets current breakpoint descriptor id	Unique breakpoint descriptor id
GetDataLoggerCaptureStateAPI(chid)	– channel id	Gets current datalogger capture state	capture state
SetBreakpoint(breakpoint_descID)	– break point descriptor id	Sets specified break point	0- Success / ErrorCode- Failure
SetDataLoggerCaptureProperties (ChannelID, startTrigger_watcher_descID, stopTrigger_watcher_descID, preStartTriggerInterval, postStopTriggerInterval)	– ChannelID – startTrigger_watcher_descID – stopTrigger_watcher_descID – preStartTriggerInterval – postStopTriggerInterval	Set start and stop watcher details to datalogger and also sets the pre-trigger and post trigger value for these watchers	0- Success / ErrorCode- Failure

Python Interfaces	Input parameters	Functionalities	Returns
SetSignalGeneratorWatcher(sg_details, watcher_descID)	– signal generator(segment) details – watcher descriptor id	Set watcher details for particular signal segments	0-Success / ErrorCode-Failure
SetDownSampling(channel id, downsampling_value)	– channel id – datalogger downsampling value	Sets the specified downsampling value to the datalogger	0- Success / ErrorCode-Failure
GetDownSampling(channel id)	– channel id	Gets the downsampling value which is been set to the datalogger	Down sampling value / ErrorCode-Failure

6.8 Function Parameter Details

- `modelAddressDict`:
  For example; {"`raster1`":['`MiL_Three_Cyl_7_ECU/Inports/ICE_nEng`'],"`raster2`":['`MiL_Three_Cyl_7_ECU/Inports/ICE_facLoadMf_2`']} or '`all`'
- `modelAddress` :
  For example; `Model0/Inports/In1`
- `loggingAddress`:
  Filename to be logged (e.g. `file://DataBrokerFile.csv`)

6.9 Simulation State Mapping

After calling `GetCurrentSimulationState()` API, it generally returns the state in decimal format. The table below shows the state mapping of the returned decimal value.

Value	State
Accepted States	
7	eStopped
9	eStarted
11	ePaused
Error States	
8	eStarting
10	eStopping
12	ePausing

7 COSYM Workflow

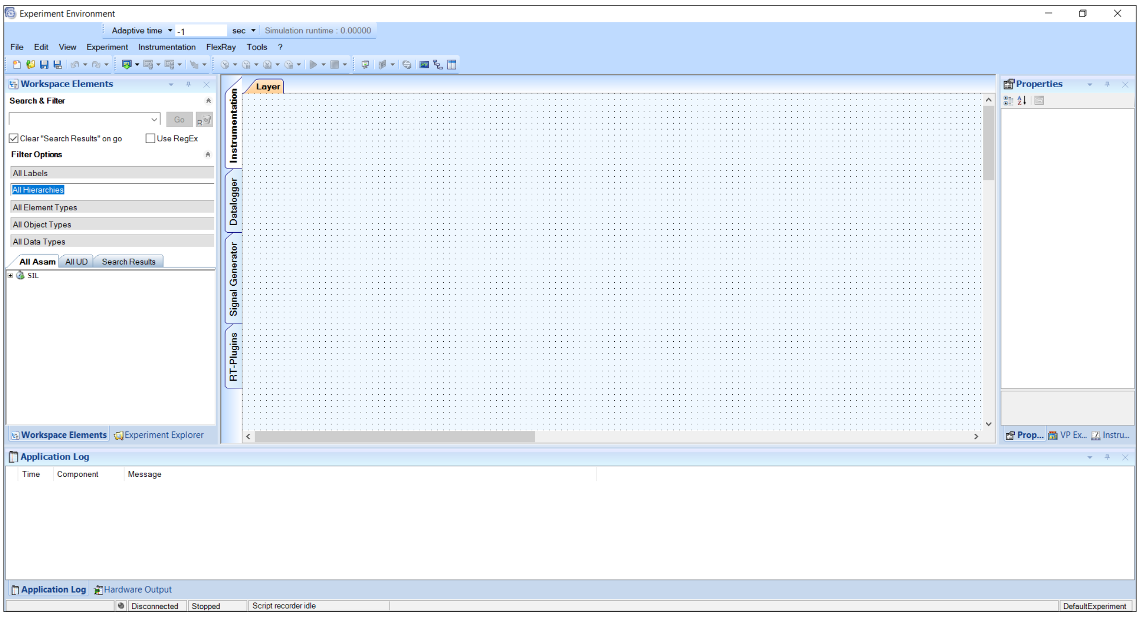
The workflow below shows how to create the `Sample.py` file.

The `deployablesWindows.zip` and `configFiles.zip` folders are required to call the SiL APIs. These will be created during the build process of a COSYM project.

To create `deployablesWindows.zip` and `configFiles.zip` folders

1. Launch COSYM and create a project.
2. Change the target to SiL_ATS.
3. Import the Simulation models and connect those models.
4. Configure the OS.
5. Go to **View** menu → **Build & run experiment**.
6. ETAS Experiment Environment is opened after the successful Build process.

Refer to "Simulating an Idle Speed Controller of a 3-Cylinder Gasoline Engine using ATS use case" section for the complete details to build an ATS project.



Experiment Environment
Adaptive time -1 sec Simulation runtime 0.00000
File Edit View Experiment Instrumentation FlexRay Tools ?
Workspace Elements
Search & Filter
Go
Clear "Search Results" on go
Use RegEx
Filter Options
All Labels
All Hierarchies
All Element Types
All Object Types
All Data Types
All Asam All UD Search Results
SIL
Instrumentation
Datalogger
Signal Generator
RT-Plugins
Layer
Properties
Prop... VP Ex... Instru...
Workspace Elements Experiment Explorer
Application Log
Time Component Message
Application Log Hardware Output
Disconnected Stopped Script recorder idle DefaultExperiment

Fig. 7-1: ETAS Experiment Environment

Once the project is built, the `deployablesWindows.zip` and `configFiles.zip` folders are created in the ‹Project path›`\Systems\System\Configuration\OS\codegen` folder.

7.1 Creating Example File to Use TargetClient Python Model

Create a `Sample.py` file in any of the text editor or python editor by referring to the mentioned steps below:

```
import TargetClientPythonAPI
from time import sleep
import sys
MIN_INT = -2147483648
```

Test ErrorDescription API

```
def GetErrorDescription(errorCode):
    if errorCode != 0:
        retErrMSG=TargetClientPythonAPI.ErrorDescription(errorCode)
        print(retErrMSG)
        raise ValueError('Exception Occurred: ', retErrMSG)
```

Introduced new APIs instead of OpenSimulation()

```
def DownloadSimulation(deployablePath,configPath):
    ipAddress = 'localhost'
    userName = ''
    passWord = ''
    val = TargetClientPythonAPI.GetTargetSessionAPI(ipAd-
dress,userName,passWord)
    print ('\nGetTargetSessionAPI:', val) #Expected value is -1 for
Local Host

    if val == MIN_INT:
        print('\nWrong Session')
    else:
        source = TargetClientPythonAPI.Simulation_artifacts_t()
        source.config_file_path = configPath
        source.deployable_file_path = deployablePath

        val = TargetClientPythonAPI.DeployAPI(val, source, Tar-
getClientPythonAPI.SIL)
        print('\nDeployAPI:', val)
        GetErrorDescription(val)
```

Test ControlClient APIs with newly introduced API

```
def ControlClientTest(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    val=TargetClientPythonAPI.StartSimulation(20)
    print ('\nStartSimulation:',val)
    GetErrorDescription(val)

    currState = TargetClientPythonAPI.GetCurrentSimulationState()
    print ('\nGetCurrentSimulationState:', currState)
    GetErrorDescription(currState[0])

    while currState[1] != TargetClientPythonAPI.networkState_eStopped:
        sleep(1)
        currState = TargetClientPythonAPI.GetCurrentSimulationState()

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

Test SignalGenerator APIs

```
def createChannel(sgId):
    global SimOid
    chObj1=TargetClientPythonAPI.createChannel_t()
    chObj1.chanName="SignalGenerator1/Signal1"
    chObj1.chanNo = 0
    chObj1.sgId = sgId[1]
    SimOid1 = TargetClientPythonAPI.CreateSGChannel(chObj1)

    SimOid=SimOid1[1]
    GetErrorDescription(SimOid1[0])

    return SimOid
```

```
def setSineWaveSignal(sgId):
    sineObj=TargetClientPythonAPI.setSinewaveSignal_t()
    sineObj.sgId = sgId[1]
    sineObj.createFlg = 1
    sineObj.channelNo = 0
    sineObj.intervalNo = 0
    sineObj.countMax = 10000
    sineObj.deltaT = 3.0
    sineObj.frequency = 1.0
    sineObj.phase = 0.0
    sineObj.amplitude = 3.0
    sineObj.offset = 0.0
    retValue = TargetClientPythonAPI.SetSinewaveSignal(sineObj)
    print ('\nsetSineWaveSignal:', retValue)

    GetErrorDescription(retValue[0])
```

```
def setSineWaveSignalWithWatcher(sgId):

    stopWatcher = TargetClientPythonAPI.condition_watcher_info()
    stopWatcher.lpseudoAddr ='MiL_Three_Cyl_7_ECU/ICE_nEng'
    stopWatcher.rpseudoAddr = 'MiL_Three_Cyl_7_ECU/ICE_nEng'
    stopWatcher.timeoutValue = 10000
    stopWatcher.constantValue = 5000
    stopWatcher.rSignalorConstant =TargetClientPythonAPI.eCONSTANT
    stopWatcher.trigger_condition =TargetClientPythonAPI.eEQUAL

    watcherDescID =TargetClientPythonAPI.CreateConditionWatcherAPI
(stopWatcher)
    print('\nWatcherDescID:', watcherDescID)

    sineObj=TargetClientPythonAPI.setSinewaveSignal_t()
    sineObj.sgId = sgId[1]
    sineObj.createFlg = 1
    sineObj.channelNo = 0
    sineObj.intervalNo = 0
    sineObj.countMax = 10000
    sineObj.deltaT = 3.0
    sineObj.frequency = 1.0
    sineObj.phase = 0.0
    sineObj.amplitude = 3.0
    sineObj.offset = 0.0

    details=TargetClientPythonAPI.sg_details()
    details.sgId = sgId[1]
    details.channelNo = 0
    details.intervalNo = 0

    val=TargetClientPythonAPI.SetSignalGeneratorWatcher(details, watch-
erDescID)
    print ('\nSetSignalGeneratorWatcher:', val)
    GetErrorDescription(val[0])

    retValue = TargetClientPythonAPI.SetSinewaveSignal(sineObj)
    print ('\nsetSineWaveSignal:', retValue)
    GetErrorDescription(retValue[0])

    return watcherDescID
```

```
def setDataSignal(sgId):
    dataObj=TargetClientPythonAPI.setDataSignal_t()
    dataObj.sgId = sgId[1]
    dataObj.channelNo = 0
    dataObj.intervalNo = 0
    dataObj.createFlg = 1
    dataObj.countMax = 10000
    dataObj.intpolType = 0
    dataObj.nofSamples = 4
    dataObj.ptrValue=TargetClientPythonAPI.SetDataSignalDoublePtr
([0.0,1.0,2.0,3.0])
    dataObj.ptrTime=TargetClientPythonAPI.SetDataSignalDoublePtr
([0.00,0.01,0.02,0.03])

    retValue = TargetClientPythonAPI.SetDataSignal(dataObj)
    print ('\nsetDataSignal:', retValue)

    TargetClientPythonAPI.FreeMemory(dataObj.ptrValue)
    TargetClientPythonAPI.FreeMemory(dataObj.ptrTime)

    GetErrorDescription(retValue[0])
```

```
def LinkPortWithModeAPI(modelAddress, SimOid, eMode, constValue):
    ret = TargetClientPythonAPI.LinkPortWithModeAPI(modelAddress,
SimOid, eMode, constValue)
    print('LinkPortWithModeAPI:', ret)
    GetErrorDescription(ret[0])
```

```
def InitSignalGenerator(sgId,deployablePath,configPath):
    watcherDescID=0
    SimOid=createChannel(sgId)
    print('\nSimOid:', SimOid)

    #setDataSignal(sgId)
    setSineWaveSignal(sgId) ## Without Watcher Enabled
    #watcherDescID = setSineWaveSignalWithWatcher(sgId) ## With
Watcher Enabled

    timeSliceObj=TargetClientPythonAPI.setTimeslice_t()
    timeSliceObj.sgId = 1
    timeSliceObj.createFlg = 1
    timeSliceObj.sliceNo = 0
    timeSliceObj.deltaT = 300.0
    timeSliceObj.startT = 0.0
    timeSliceObj.countMax = 1000
    retValue = TargetClientPythonAPI.SetSGTimeslice(timeSliceObj)
    print ('\nSetSGTimeslice:', retValue)

    GetErrorDescription(retValue[0])

    retValue = TargetClientPythonAPI.ResetSGInstance(sgId[1])
    print ('ResetSGInstance:', retValue)
    GetErrorDescription(retValue[0])

    modelAddress='MiL_Three_Cyl_7_ECU/ICE_nEng'
    LinkPortWithModeAPI(modelAddress, SimOid, eMode=6, constValue=0)

    retValue = TargetClientPythonAPI.StartSGInstance(sgId[1])
    print ('StartSGInstance:', retValue)
    GetErrorDescription(retValue[0])

    val=TargetClientPythonAPI.StartSimulation(10)
    print ('\nStartSimulation:', val)
    GetErrorDescription(val)

    sleep(5)
    return watcherDescID
```

```
def ExitSignalGenerator(sgId):
    retValue=TargetClientPythonAPI.StopSGInstance(sgId[1])
    print ('\nStopSGInstance:', retValue)
    GetErrorDescription(retValue[0])

    timeValue=TargetClientPythonAPI.GetSGTime(sgId[1])
    print ('GetSGTime:', timeValue)
    GetErrorDescription(timeValue[0])

    modelAddress='MiL_Three_Cyl_7_ECU/ICE_nEng'
    LinkPortWithModeAPI(modelAddress, SimOid, eMode=1, constValue=0)

    retValue=TargetClientPythonAPI.ResetSGInstance(sgId[1])
    print ('ResetSGInstance:', retValue)
    GetErrorDescription(retValue[0])

    dchObj1=TargetClientPythonAPI.deleteChannel_t()
    dchObj1.chanNo = 0
    dchObj1.sgId = sgId[1]
    retValue = TargetClientPythonAPI.DeleteSGChannel(dchObj1)
    print ('DeleteSGChannel:', retValue)
    GetErrorDescription(retValue[0])

    retValue=TargetClientPythonAPI.DeleteAllSGInstances()
    GetErrorDescription(retValue[0])

    sleep(10)

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

```
def SignalGeneratorTest(deployablePath,configPath):
    watcherDescID=0
    DownloadSimulation(deployablePath,configPath)

    msg = TargetClientPythonAPI.createGeneratorInstance_t()

    msg.taskIndex = 0
    msg.startTime = 0.000
    msg.channelReset = 1
    msg.sgName= "SignalGenerator1"

    sgId=TargetClientPythonAPI.CreateSGInstance(msg)
    print ('\nsgId:',sgId)

    GetErrorDescription(sgId[0])

    watcherDescID=InitSignalGenerator(sgId,deployablePath,configPath)
    if watcherDescID != 0 :
        val=TargetClientPythonAPI.DeleteConditionWatcher(watch-
erDescID)
        print ('\nDeleteConditionWatcher:',val)
        GetErrorDescription(val[0])
    ExitSignalGenerator(sgId)
```

Test Parameterization APIs

```
def ParameterizationTestScalar(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    modelAddress='MiL_Three_Cyl_7_ECU/ECU_3Cyl/ECU_AS/AS_PREDICTION_
THROTTLE_CTR/ECU_phiThrottleDesDel_BdcX/Unit Delay/MiL_Three_Cyl_7_
ECU_P_UnitDelay_InitialCondition'
    scalarData=15.5

    retVal=TargetClientPythonAPI.WriteScalarParameter(mod-
elAddress,scalarData,TargetClientPythonAPI.eFloat64)
    print ('\nWriteScalarParameter:', retVal)
    GetErrorDescription(retVal[0])

    retVal=TargetClientPythonAPI.ReadScalarVariable(mod-
elAddress,TargetClientPythonAPI.eFloat64)
    print ('\nReadScalarVariable:', retVal)
    GetErrorDescription(retVal[0])

    val=TargetClientPythonAPI.StartSimulation(-1)
    print ('\nStartSimulation:', val)
    GetErrorDescription(val)

    sleep(10)

    val=TargetClientPythonAPI.StopSimulation()
    print ('\nStopSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

```
def ParameterizationTestVector(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)
    modelAddress='MiL_Three_Cyl_7_ECU/ECU_3Cyl/ECU_IS/Ignition System 
(Is) Ignition Angle Calculation1/ECU_iaDelta2Opt_CUR/MiL_Three_Cyl_7_
ECU_P_ECU_iaDelta2Opt_CUR_tableData'
    dataArr=[44,54,64,74,84,94,104,114]

    retVal=TargetClientPythonAPI.WriteVectorParameter(mod-
elAddress,dataArr,TargetClientPythonAPI.eFloat64)
    print ('\nWriteVectorParameter:', retVal)
    GetErrorDescription(retVal[0])

    retVal=TargetClientPythonAPI.ReadVectorVariable(mod-
elAddress,TargetClientPythonAPI.eFloat64,8,0)
    print ('\nReadVectorVariable:', retVal)
    GetErrorDescription(retVal[0])

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

```
def ParameterizationTestMultiParameter(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    modelAddress=['MiL_Three_Cyl_7_ECU/ECU_3Cyl/ECU_IS/Ignition System 
(Is) Ignition Angle Calculation1/ECU_iaDelta2Opt_CUR/MiL_Three_Cyl_7_
ECU_P_ECU_iaDelta2Opt_CUR_bp01Data',
                    'MiL_Three_Cyl_7_ECU/ECU_3Cyl/ECU_AS/AS_
PREDICTION_THROTTLE_CTR/ECU_facLoadDesPre/ECU_phiThrottleFromMfNormed_
CUR/MiL_Three_Cyl_7_ECU_P_ECU_phiThrottleFromMfNormed_CUR_tableData']
    modelAddress1='MiL_Three_Cyl_7_ECU/ECU_3Cyl/ECU_AS/AS_PREDICTION_
THROTTLE_CTR/ECU_facLoadDesPre/ECU_phiThrottleFromMfNormed_CUR/MiL_
Three_Cyl_7_ECU_P_ECU_phiThrottleFromMfNormed_CUR_tableData'
    dataArr=[[44.78,54,64,74],[34,56.2]]

    retVal=TargetClientPythonAPI.WriteMultiParameter(modelAddress,[Tar-
getClientPythonAPI.eFloat64,TargetClientPythonAPI.eFloat64],dataArr)
    print ('\nWriteMultiParameter:', retVal)
    GetErrorDescription(retVal[0])

    sizes = [2,2]
    retval = TargetClientPythonAPI.ReadMultipleParameter(modelAddress, 
[TargetClientPythonAPI.eFloat64,TargetClientPythonAPI.eFloat64], 
sizes)
    print('\nReadMultipleParameter : ', retval)
    GetErrorDescription(retVal[0])

    retVal=TargetClientPythonAPI.ReadVectorVariable(mod-
elAddress1,TargetClientPythonAPI.eFloat64,2,0)
    print ('\nReadVectorVariable:', retVal)
    GetErrorDescription(retVal[0])

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

Test DataLogger APIs

```
def DataBrokerClientTest(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    if sys.platform == 'linux':
        dataBrokerFile = "file:///tmp/DataBrokerFile.mf4"
    else:
        dataBrokerFile = "file://C:/temp/DataBrokerFile.mf4"
    modelAddressListDict = {"TaskATS":['MiL_Three_Cyl_7_ECU/ICE_
nEng','MiL_Three_Cyl_7_ECU/ICE_facLoadMf_2']}
    #modelAddressListDict = "all"
    val=TargetClientPythonAPI.RegisterDataLogger(2, mod-
elAddressListDict,dataBrokerFile, 0.1)
    print('\nRegisterDataLogger:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.StartLoggingOnChannel(2)
    print('\nStartLoggingOnChannel:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.StartSimulation(-1)
    print ('\nStartSimulation:', val)
    GetErrorDescription(val)

    sleep(10)

    val=TargetClientPythonAPI.StopSimulation()
    print ('\nStopSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.StopLoggingOnChannel(2)
    print ('\nStopLoggingOnChannel:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

Test Condition Watcher APIs

```
def ConditionWatcherTest(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    dataBrokerFile = "file://C:/tem-
p/DataBrokerFileConditionWatcher.mf4"
    modelAddressListDict = {"TaskATS":['MiL_Three_Cyl_7_ECU/ICE_
nEng','MiL_Three_Cyl_7_ECU/ICE_facLoadMf_2']}

    startWatcher = TargetClientPythonAPI.condition_watcher_info()
    startWatcher.lpseudoAddr = 'MiL_Three_Cyl_7_ECU/ICE_nEng'
    startWatcher.rpseudoAddr = 'MiL_Three_Cyl_7_ECU/ICE_facLoadMf_2'
    startWatcher.timeoutValue = -1
    startWatcher.constantValue = 1000
    startWatcher.rSignalorConstant = TargetClientPythonAPI.eCONSTANT
    startWatcher.trigger_condition = Tar-
getClientPythonAPI.eGREATERTHAN

    stopWatcher = TargetClientPythonAPI.condition_watcher_info()
    stopWatcher.lpseudoAddr = 'MiL_Three_Cyl_7_ECU/ICE_nEng'
    stopWatcher.rpseudoAddr = 'MiL_Three_Cyl_7_ECU/ICE_facLoadMf_2'
    stopWatcher.timeoutValue = -1
    stopWatcher.constantValue = 1900
    stopWatcher.rSignalorConstant = TargetClientPythonAPI.eCONSTANT
    stopWatcher.trigger_condition = TargetClientPythonAPI.eGREATERTHAN

    startWatcherDescID=TargetClientPythonAPI.CreateConditionWatcherAPI
(startWatcher)
```

```
    stopWatcherDescID=TargetClientPythonAPI.CreateConditionWatcherAPI
(stopWatcher)

    StartTriggerCondition = startWatcherDescID
    StopTriggerCondition = stopWatcherDescID
    PreStartTriggerInterval = 0.0
    PostStopTriggerInterval = 5.0

    val=TargetClientPythonAPI.RegisterDataLogger(2, mod-
elAddressListDict,dataBrokerFile, 0.1)
    print('\nRegisterDataLogger:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.SetDataLoggerCaptureProperties(2,
StartTriggerCondition, StopTriggerCondition, PreStartTriggerInterval,
PostStopTriggerInterval)
    print('\nSetDataLoggerCaptureProperties:',val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val=TargetClientPythonAPI.StartLoggingOnChannel(2)
    print('\nStartLoggingOnChannel:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val=TargetClientPythonAPI.StartSimulation(-1)
    print ('\nStartSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    sleep(10)

    val=TargetClientPythonAPI.StopLoggingOnChannel(2)
    print ('\nStopLoggingOnChannel:', val)
    GetErrorDescription(val)
    sleep(5)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\GetDataLoggerCaptureStateAPI:',val)

    val=TargetClientPythonAPI.StopSimulation()
    print ('\nStopSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val = TargetClientPythonAPI.RemoveDataChannel(2)
    print('\nRemoveDataChannel:',val)

    val=TargetClientPythonAPI.DeleteConditionWatcher(startWatch-
erDescID)
    print ('\nDeleteConditionWatcher:', val)
    GetErrorDescription(val)
    val=TargetClientPythonAPI.DeleteConditionWatcher(stopWatch-
erDescID)
    print ('\nDeleteConditionWatcher:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

```
            Test Duration Watcher APIs

def DurationWatcherTest(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    dataBrokerFile = "file://C:/tem-
p/DataBrokerFileDurationWatcher.mf4"
    modelAddressListDict = {"TaskATS":['MiL_Three_Cyl_7_ECU/ICE_
nEng','MiL_Three_Cyl_7_ECU/ICE_facLoadMf_2']}

    startWatcherDescID = -1
    stopWatcherDescID = -1

    time = 200.0
    #durationWatcherDescID = -1
    durationWatcherDescID=TargetClientPythonAPI.CreateDur-
ationWatcherByTimeAPI(time)
    print('\nDurationWatcherDescID:',durationWatcherDescID)

    StartTriggerCondition = startWatcherDescID
    StopTriggerCondition = durationWatcherDescID
    PreStartTriggerInterval = 0.0
    PostStopTriggerInterval = 5.0

    val=TargetClientPythonAPI.RegisterDataLogger(2, mod-
elAddressListDict,dataBrokerFile, 0.1)
    print('\nRegisterDataLogger:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.SetDataLoggerCaptureProperties(2,
StartTriggerCondition, StopTriggerCondition, PreStartTriggerInterval,
PostStopTriggerInterval)
    print('\nSetDataLoggerCaptureProperties:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val=TargetClientPythonAPI.StartLoggingOnChannel(2)
    print('\nStartLoggingOnChannel:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val=TargetClientPythonAPI.StartSimulation(-1)
    print ('\nStartSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    sleep(10)

    val=TargetClientPythonAPI.StopLoggingOnChannel(2)
    print ('\nStopLoggingOnChannel:', val)
    GetErrorDescription(val)
    sleep(5)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)


    val=TargetClientPythonAPI.StopSimulation()
    print ('\nStopSimulation:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.GetDataLoggerCaptureStateAPI(2)
    print('\nGetDataLoggerCaptureStateAPI:',val)

    val = TargetClientPythonAPI.RemoveDataChannel(2)
    print('\nRemoveDataChannel:',val)

    val=TargetClientPythonAPI.DeleteDurationWatcher(dur-
ationWatcherDescID)
    print ('\nDeleteDurationWatcher:', val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.UnDeploy()
    print ('\nUnDeploy:', val)
    GetErrorDescription(val)
```

Test Break Point APIs

```
def BreakPointTest(deployablePath,configPath):
    DownloadSimulation(deployablePath,configPath)

    breakpointState1 = TargetClientPythonAPI.eSTOP
    breakpointState2 = TargetClientPythonAPI.ePAUSE
    time=50
    #step=500

    watcherDescID =TargetClientPythonAPI.CreateDur-
ationWatcherByTimeAPI(time)
    #watcherDescID =Ta-
argetClientPythonAPI.CreateDurationWatcherByStepAPI(step)
    print('\nWatcherDescID:',watcherDescID)

    print('\n--------Breakpoint Execution FOR STOP STATE--------' )

    breakpointDescID = TargetClientPythonAPI.CreateBreakpointAPI(watch-
erDescID, breakpointState1)
    print('\nBreakpointDescID for eSTOP:',breakpointDescID)

    val= TargetClientPythonAPI.SetBreakpoint(breakpointDescID)
    GetErrorDescription(val)
    print('\nSetBreakpoint:',val)

    val=TargetClientPythonAPI.StartSimulation(-1)
    GetErrorDescription(val)
    print ('\nStartSimulation:', val)

    while True:
        sleep(10)
        val=TargetClientPythonAPI.GetSimulationTime()
        print('\nGetSimulationTime:',val[1])
        simulationState = TargetClientPythonAPI.GetCur-
rentSimulationState()
```

```
        print('\nSimulationState:',simulationState)
        if (simulationState[1]==TargetClientPythonAPI.networkState_
eStopped):
            break
    val= TargetClientPythonAPI.GetBreakpointAPI()
    print('\nGetBreakpointAPI:',val)

    simulationState = TargetClientPythonAPI.GetCurrentSimulationState
()
    GetErrorDescription(simulationState[0])
    print('\nSimulationState:',simulationState)

    val= TargetClientPythonAPI.DeleteBreakpoint(breakpointDescID)
    GetErrorDescription(val)
    print('\nDeleteBreakpoint:',val)

    print('\n--------Breakpoint Execution FOR PAUSE STATE--------' )

    breakpointDescID = TargetClientPythonAPI.CreateBreakpointAPI(watch-
erDescID, breakpointState2)
    print('\nBreakpointDescID for ePAUSE:',breakpointDescID)

    val= TargetClientPythonAPI.SetBreakpoint(breakpointDescID)
    GetErrorDescription(val)
    print('\nSetBreakpoint:',val)

    val=TargetClientPythonAPI.StartSimulation(-1)
    GetErrorDescription(val)
    print ('\nStartSimulation:', val)

    while True:
        sleep(10)
        val=TargetClientPythonAPI.GetSimulationTime()
        print('\nGetSimulationTime:',val[1])
        simulationState = TargetClientPythonAPI.GetCur-
rentSimulationState()
        print('\nSimulationState:',simulationState)
        if (simulationState[1]==TargetClientPythonAPI.networkState_
ePaused):
            break

    simulationState = TargetClientPythonAPI.GetCurrentSimulationState
()
    GetErrorDescription(simulationState[0])
    print('\nSimulationState:',simulationState)

    val = TargetClientPythonAPI.ResumeSimulation()
    print('\nResumeSimulation : ',val)
    GetErrorDescription(val)

    val=TargetClientPythonAPI.StopSimulation()
    GetErrorDescription(val)
    print ('\nStopSimulation:', val)

    val= TargetClientPythonAPI.DeleteBreakpoint(breakpointDescID)
    GetErrorDescription(val)
    print('\nDeleteBreakpoint:',val)

    val=TargetClientPythonAPI.UnDeploy()
    GetErrorDescription(val)
    print ('\nUnDeploy:', val)
```

7.2 Calling Exposed Functions from Main()

```
def main():

    try:
        deployablePath=str(sys.argv[1])
        configPath=str(sys.argv[2])

        print('\n-----------------------------------------')
        print('\n     Control Client Test Execution:' )
        print('\n-----------------------------------------')
        ControlClientTest(deployablePath,configPath)

        print('\n-----------------------------------------')
        print('\n     Signal Generator Test Execution:' )
        print('\n-----------------------------------------')
        SignalGeneratorTest(deployablePath,configPath)

        print('\n-----------------------------------------')
        print('\n     Data Broker Test Execution:' )
        print('\n-----------------------------------------')
        DataBrokerClientTest(deployablePath,configPath)

        print('\n-----------------------------------------')
        print('\n     Parameterization Test Execution:' )
        print('\n-----------------------------------------')

        print('\n--------Scalar Test with New API:--------' )
        ParameterizationTestScalar(deployablePath,configPath)

        print('\n--------Vector Test with New API:--------' )
        ParameterizationTestVector(deployablePath,configPath)

        print('\n--------Multiparameter Test with New API:--------' )
        ParameterizationTestMultiParameter(deployablePath,configPath)

        print('\n-----------------------------------------')
        print('\n     Watcher and Capturer Test Execution:' )
        print('\n-----------------------------------------')

        print('\n--------Condition Watcher Test:--------' )
        ConditionWatcherTest(deployablePath,configPath)

        print('\n--------Duration Watcher Test:--------' )
        DurationWatcherTest(deployablePath,configPath)

        print('\n-----------------------------------------')
        print('\n     Break Point Test Execution:' )
        print('\n-----------------------------------------')
        BreakPointTest(deployablePath,configPath)


        print('SUCCESS')
        return 0
    except Exception as e:
        print(e)
        print('FAILURE')
        return -1
```

```
if __name__== "__main__":
    main()
```

Similarly, other APIs exposed from "Parameterization APIs" on page 15 can also be used.

7.3 Executing the Python File

Above mentioned example is created for '3CylATS_Sim' project which comes along with installer.

Build the project with newly installed COSYM installer and execute the sample file using the commands below:

– ‹COSYM demo data location›`\Samples\PythonScripts\PyEnvSettings.bat`

– `python Sample.py` ‹COSYM demo data location›`\Samples\SiL\3CylATS_Sim\Systems\SystemATS\Configuration\OS\codegen\deployablesWindows.zip` ‹COSYM demo data location›`\Samples\SiL\3CylATS_Sim\Systems\SystemATS\Configuration\OS\codegen\configFiles.zip`

8 Error Reporting

The ATS API return mechanism is standardized in COSYM and new API is introduced to know the error description.

8.1 Return Codes and its Descriptions

- For the successful execution, it returns the value `SIL_SUCCESS`
- All the error return values must start from 0x80000000
- To simplify, the errors are sub categorized based on the API functionalities.
- Use the API `GetErrorDescription` to get the error description.

Error Codes	Values	Converted Decimal values	Description
//Result codes			
SIL_SUCCESS	0x00000000		Success
SIL_RESULT_ERROR	0x80000000	2147483648	Error
//Error Type			
SIL_RESULT_TYPE_ LICENSE	0x00001000		
SIL_RESULT_TYPE_ INTERNAL	0x00002000		
SIL_RESULT_TYPE_ SIMULATION	0x00003000		

Error Codes	Values	Converted Decimal values	Description
SIL_RESULT_TYPE_ IPC	0x00004000		
SIL_RESULT_TYPE_ PARAM	0x00005000		
SIL_RESULT_TYPE_ BROKER	0x00006000		
SIL_RESULT_TYPE_ DDS	0x00007000		
SIL_RESULT_TYPE_ SIGGEN	0x00008000		
SIL_RESULT_TYPE_ PROBE	0x00009000		
SIL_RESULT_TYPE_ BREAKPOINT	0x000A00		
//Specific	Errors related to license		
SIL_ERR_LICENSE_ NOTLOADED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_LICENSE I 0x00000001	2147487745	License Error. Reason: License Not Loaded.
SIL_ERR_LICENSE_ SIL	SIL_RESULT_ERROR I SIL_RESULT_TYPE_LICENSE I 0x00000002	2147487746	License Error. Reason: SIL License Issue.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_LICENSE_ VNET	SIL_RESULT_ERROR I SIL_RESULT_TYPE_LICENSE I 0x00000004	2147487748	License Error. Reason: vNet License Issue.
SIL_ERR_INVALID_ LICENSE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_LICENSE I 0x00000005	2147487749	License Error. Reason: Invalid License.
//Internal Errors			
SIL_ERR_NOT_ IMPLEMENTED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000001	2147491841	Internal Error. Reason: Function is not implemented.
SIL_ERR_INVALID_ STATE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000002	2147491842	Internal Error. Reason: State is not valid.
SIL_ERR_NULL_ OBJECT	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000003	2147491843	Internal Error. Reason: Object is empty.
SIL_ERR_NULL_ ESSE_CONTROL_ MASTER	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000004	2147491844	Internal Error. Reason: Esse Control Master Object is empty.
SIL_ERR_NULL_ ESSE_SUMILATION_ CONTROL	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000005	2147491845	Internal Error. Reason: Esse Simulation Control Object is empty.
SIL_ERR_NULL_ LICENSE_MANAGER	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000006	2147491846	Internal Error. Reason: License Manager Object is empty.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_UNKNOWN	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000007	2147491847	Internal Error. Reason: Unknown.
SIL_INITIALISE_ FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_INTERNAL I 0x00000008	2147491848	Internal Error. Reason: Initialization failed.
//Simulation info codes			
SIL_ERR_STATE_ STARTING	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 8	2147495944	State Error. Reason: Not a correct state.
SIL_ERR_STATE_ PAUSING	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 12	2147495954	State Error. Reason: Not a correct state.
SIL_ERR_STATE_ STOPPING	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 10	2147495952	State Error. Reason: Not a correct state.
//Simulation Error Codes			
SIL_ERR_FAILED_ INIT	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000001	2147495937	Simulation Error. Reason: Initialization failed.
SIL_ERR_FAILED_ START	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000002	2147495938	Simulation Error. Reason: Filed to start.
SIL_ERR_FAILED_ PAUSE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000003	2147495939	Simulation Error. Reason: Failed to pause.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_FAILED_ STOP	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000004	2147495940	Simulation Error. Reason: Failed to stop.
SIL_ERR_FAILED_ UNKNOWN	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000005	2147495941	Simulation Error. Reason: Unknown.
SIL_ERR_INVALID_ END_TIME	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIMULATION I 0x00000006	2147495942	Simulation Error. Reason: Neg-ative(except -1) and Zero values are invalid .
//Network related error codes			
SIL_ERR_FAILED_ CORBA	SIL_RESULT_ERROR I SIL_RESULT_TYPE_IPC I 0x00000001	2147500033	Network Error. Reason: CORBA failed.
SIL_ERR_FAILED_ NETWORK	SIL_RESULT_ERROR I SIL_RESULT_TYPE_IPC I 0x00000002	2147500034	Network Error. Reason: Other Network error.
//Specific Errors related to Parameterization			
SIL_ERR_FAILED_ TOWRITE_PARAM	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000001	2147504129	Parameterization Error. Reason: Failed to write.
SIL_ERR_FAILED_ TOREAD_PARAM	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000002	2147504130	Parameterization Error. Reason: Failed to read

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_FAILED_ WRONG_ NETWORK_STATE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000003	2147504131	Parameterization Error. Reason: Network state should be Paused or Stopped.
SIL_ERR_FAILED_ INVALID_VARIABLE_ PATH	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000004	2147504132	Parameterization Error. Reason: Couldn't find the variable path.
SIL_ERR_FAILED_ INVALID_FUNCTION_ SEQUENCE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000005	2147504133	Parameterization Error. Reason: Invalid function sequence. Signal generator is enabled for the unconnected inport and write API is used to edit the value of the inport.
SIL_ERR_FAILED_ INPORT_IS_ CONNECTED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PARAM I 0x00000006;	2147504134	write API cannot be used for con-nected port
//Specific Errors related to Broker			
SIL_ERR_FAILED_ ORB	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000001	2147508225	Data Broker Error. Reason: Unknown.
SIL_ERR_NOT_ INITIALIZED_ORB	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000002	2147508226	Data Broker Error. Reason: ORB not Initialized.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_TIMEOUT_ORB	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000003	2147508227	Data Broker Error. Reason: ORB timeout Occurred.
SIL_ERR_COMMUNICATION_FAILED_ORB	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000004	2147508228	Data Broker Error. Reason: ORB Communication Failed.
SIL_ERR_CHANNEL_REMOVED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000005	2147508229	Data Broker Error. Reason: Channel is Removed.
SIL_ERR_ALREADY_INITIALIZED_CHANNEL	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000006	2147508230	Data Broker Error. Reason: Channel Already Initialized.
SIL_ERR_NOT_INITIALIZED_CHANNEL	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000007	2147508231	Data Broker Error. Reason: Channel Not Initialized.
SIL_ERR_CHANNEL_INITIALIZATION_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BROKER I 0x00000008	2147508232	Data Broker Error. Reason: Failed to Initialize Channel.
//Specific Errors related to DDS			
SIL_ERR_DDS_SERVER_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_DDS I 0x00000001	2147512321	DDS Error. Reason: Connection to DDS Server Failed.
//Specific Errors related to Signal Generator			

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_INSTANCE_CREATION_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000001	2147516417	Signal Generator Error. Reason: Failed to Create Signal Generator Instance.
SIL_ERR_TASK_TABLE_NOT_SPECIFIED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000002	2147516418	Signal Generator Error. Reason: Task Table Not Specified.
SIL_ERR_OBJECT_NOT_FOUND	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000003	2147516419	Signal Generator Error. Reason: Object Not Found.
SIL_ERR_INCONSISTENCY	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000004	2147516420	Signal Generator Error. Reason: Inconsistency Occurred.
SIL_ERR_DELETE_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000005	2147516421	Signal Generator Error. Reason: Failed to Delete Signal Generator Instance.
SIL_ERR_INSTANCE_START_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000006	2147516422	Signal Generator Error. Reason: Failed to Start Signal Generator Instance.
SIL_ERR_INSTANCE_STOP_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000007	2147516423	Signal Generator Error. Reason: Failed to Stop Signal Generator Instance.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_ INSTANCE_RESET_ FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000008	2147516424	Signal Generator Error. Reason: Failed to Reset Signal Generator Instance.
SIL_ERR_CHANNEL_ CREATE_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000009	2147516425	Signal Generator Error. Reason: Failed to Create Channel.
SIL_ERR_CHANNEL_ DELETE_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000010	2147516432	Signal Generator Error. Reason: Failed to Delete Channel.
SIL_ERR_TIME_ SLICE_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000011	2147516433	Signal Generator Error. Reason: Time Slice Failed.
SIL_ERR_TIME_ SLICE_NOT_FOUND	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000012	2147516434	Signal Generator Error. Reason: Time Slice Not Found.
SIL_ERR_TIME_ SLICE_DELETE_ FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000013	2147516435	Signal Generator Error. Reason: Failed to Delete Time Slice.
SIL_ERR_CHANNEL_ NOT_FOUND	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000014	2147516436	Signal Generator Error. Reason: Channel Not Found.
SIL_ERR_SIG_ INTERVAL_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000015	2147516437	Signal Generator Error. Reason: Signal Interval Failed.
SIL_ERR_INTERVAL_ NOT_FOUND	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000016	2147516438	Signal Generator Error. Reason: Interval Not Found.

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_ UNKNOWN_SIGGEN	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000017	2147516439	Signal Generator Error. Reason: Unknown
SIL_ERR_LINK_ SIGGEN	SIL_RESULT_ERROR I SIL_RESULT_TYPE_SIGGEN I 0x00000018	2147516440	Signal Generator Error. Reason: Link Not Found.
//Specific Errors related to Probe Config Server			
SIL_ERR_PROBE_ SERVER_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PROBE I 0x00000001	2147520513	Probe Config Error. Reason: Connection to Probe Config Server Failed.
SIL_ERR_ UNKNOWN_PROBE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_PROBE I 0x00000002	2147520514	Probe Config Error. Reason: Unknown
//Specific Errors related to Breakpoint			
SIL_ERR_ BREAKPOINT_ CREATION_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000001	2147524609	Breakpoint creation failed
SIL_ERR_ BREAKPOINT_ DELETION_FAILED	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000002	2147524610	Breakpoint deletion failed
SIL_ERR_INVALID_ BREAKPOINT_ID	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000003	2147524611	Invalid breakpoint Id

Error Codes	Values	Converted Decimal values	Description
SIL_ERR_INVALID_ SIMULATION_STATE	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000004	2147524612	Invalid Simulation state for Break-point Creation
SIL_ERR_ BREAKPOINT_ UNKNOWN_ERROR	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000005	2147524613	Breakpoint Unknown Error
SIL_ERR_ BREAKPOINT_ WATCHER_CONFIG_ INVALID	SIL_RESULT_ERROR I SIL_RESULT_TYPE_BREAKPOINT I 0x00000006	2147524614	Breakpoint creation failed : watcher configuration does not exist

9 Contact Information

Technical Support

For details of your local sales office as well as your local technical support team and product hotlines, take a look at the ETAS website:

www.etas.com/hotlines

ETAS Headquarters

ETAS GmbH

Borsigstraße 24	Phone:	+49 711 3423-0
70469 Stuttgart	Fax:	+49 711 3423-2106
Germany	Internet:	www.etas.com

Index

Glossary

A

API

Application Programming Interface