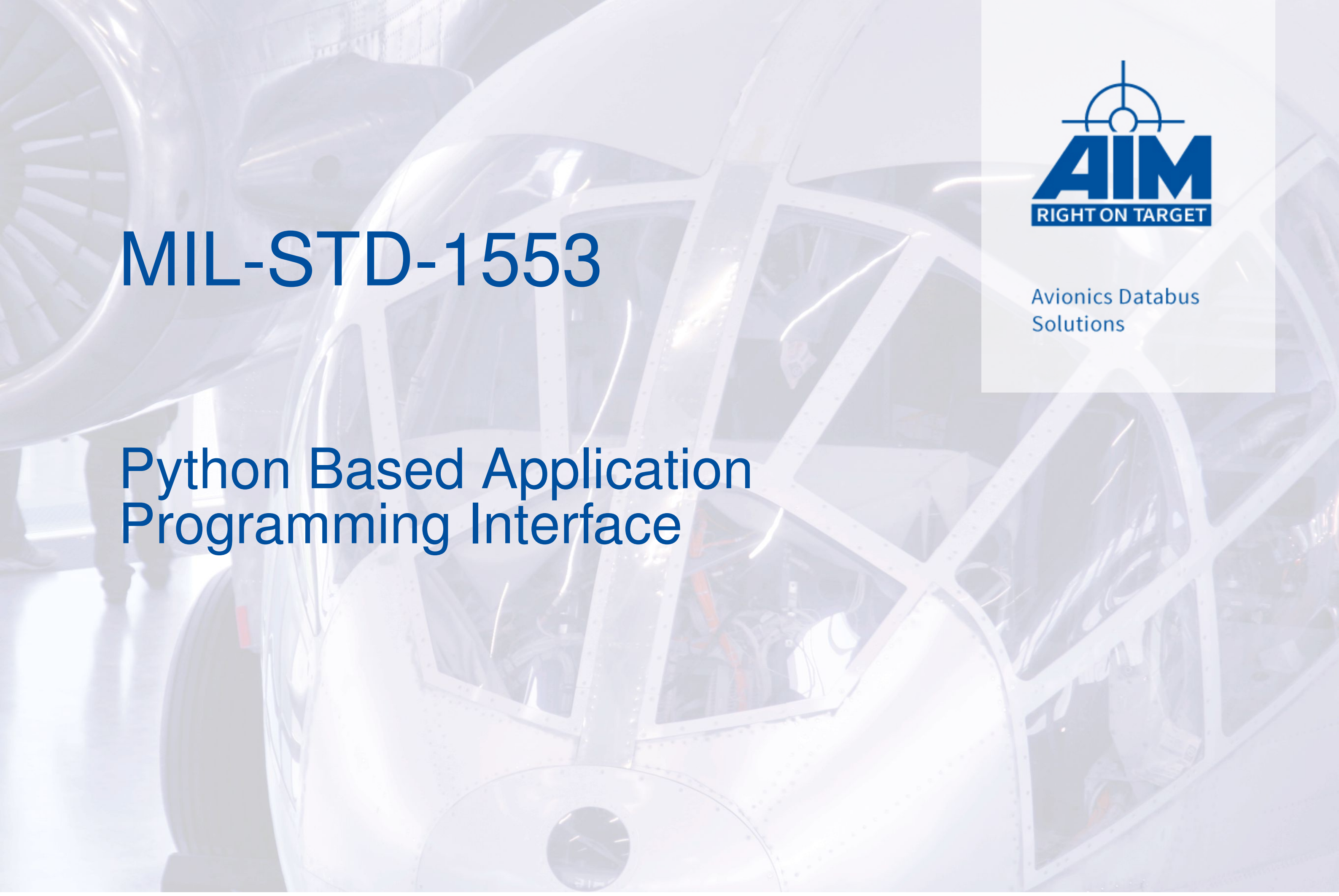


Avionics Databus Solutions

MIL-STD-1553

Python Based Application Programming Interface

Programmer's Guide

Version 24.22.0
December, 2024

www.aim-online.com

MIL-STD-1553

Python Based Application Programming Interface

Programmer's Guide

Version 24.22.0
December, 2024

AIM NO. 60-11945-37-24.22.0

AIM – Gesellschaft für angewandte Informatik und Mikroelektronik mbH

AIM GmbH
Sasbacher Str. 2
D-79111 Freiburg / Germany
Phone +49 (0)761 4 52 29-0
Fax +49 (0)761 4 52 29-33
sales@aim-online.com

AIM GmbH – Munich Sales Office
Terofalstr. 23a
D-80689 München / Germany
Phone +49 (0)89 70 92 92-92
Fax +49 (0)89 70 92 92-94
salesgermany@aim-online.com

AIM UK Office
Cressex Enterprise Centre, Lincoln Rd.
High Wycombe, Bucks. HP12 3RB / UK
Phone +44 (0)1494-446844
Fax +44 (0)1494-449324
salesuk@aim-online.com

AIM USA LLC
Seven Neshaminy Interplex
Suite 211 Trevose, PA 19053
Phone 267-982-2600
Fax 215-645-1580
sales@aim-online.us

TABLE OF CONTENTS

1 Introduction

This Programmer's Guide has been developed to guide users that want to control their AIM mil bus devices using custom applications written in the Python scripting language. This can be done through a dedicated Python wrapper.

The interface is provided with a Python package called *aim_mil*. At this time, the package supports Python 3.x.

Note:
Make sure to install the aim_mil package with pip before starting the sample. Open a command shell in the BSPs add-ons/python folder and run *python -m pip install aim_mil-24.x.y-py3-none-any.whl*

Additionally the BSP provides a Python 3 QT based BSP Tool. It can be used to show versions of all installed MIL devices and check if the versions match versions of a dedicated BSP.

AIM is also a leading designer and manufacturer of other high performance test and simulation modules, data bus analyzer software and systems for MIL-STD-1553 A/B, AFDX/ARINC664, ARINC429, MIL-STD-1760 and CAN/ARINC825 Applications, PANAVIA Serial Link and Fibre Channel. Supported hardware platforms include Peripheral Component Interconnect (PCI), PCI Express (PCIe), Compact PCI (cPCI), Versa Module Eurocard (VME), VME eXtensions for Instrumentation (VXI), PC104+, PC-Card, PCI Mezzanine Card (PMC), Express Card and Universal Serial Bus (USB). Information about all AIM products can be found at `http://www.aim-online.com`.

2 API access using the Python 3 wrapper

At the core, the interface between Python scripts and AIM's mil bus C\C++ based Application Programming Interface (API) library is implemented in a dedicated Python wrapper. The API interface for Python scripts is the AIM MIL Python package *aim_mil*. The package is available in the BSPs *add-ons/python/* folder as a wheel file and can be installed with pip.

The package contains the following files:

mil_bindings_api.py Python wrapper functions for 1553 and generic functions. This wrapper contains a Python function for each C-Function. Each function name has a *Py* prefix. The function arguments are in the same order as in the C-Functions and the MIL-STD-1553 API Reference Manual can be used as a description. The only difference is that the obsolete BIU parameter has been omitted for all functions.

mil_bindings_types.py Python classes for all 1553 and generic C-Structures used in the API. This classes are used as input and return parameters for the wrapper functions above. All classes have the C-Structure name with a *PY_* prefix.

mil_bindings_defines.py Python defines for all 1553 and generic C-Defines used in the API. This defines are used as input and return parameters for the wrapper functions above.

The BSP folder *add-ons/python/* also contains a Python 3 sample application that can be used as further reference on how to implement Python applications for our API.

3 Running the Python 3 sample included in the BSP

3.1 Windows

Before the Python sample is started the *aim_mil* wheel package needs to be installed once with *python -m pip install aim_mil-24.x.y-py3-none-any.whl*.
The sample can then be started by executing *python 1553_sample.py* from within the sample directory.

```
\BSP_DIR\Add-Ons\python>python -m pip install aim_mil-24.x.y-py3-none-any.whl
\BSP_DIR\Add-Ons\python>cd sample
\BSP_DIR\Add-Ons\python\sample>python 1553_sample.py
*********************************************************************
***          This program is licensed under the MIT license.
***
***        Copyright (c) 2023-2024 AIM GmbH <info@aim-online.com>
***
*********************************************************************
action [sample, help, list] ?
server [local] ?
module number [0..31] ?
stream number [1..8] ?
```

3.2 Linux

Before the Python sample is started the *aim_mil* wheel package needs to be installed once with *python -m pip install aim_mil-24.x.y-py3-none-any.whl*.
The sample can then be started by executing *python 1553_sample.py* from within the sample directory.

```
/BSP_DIR/add-ons/python$ python -m pip install aim_mil-24.x.y-py3-none-any.whl
/BSP_DIR/add-ons/python$ cd sample
/BSP_DIR/add-ons/python$ python 1553_sample.py
*********************************************************************
***          This program is licensed under the MIT license.
***
***        Copyright (c) 2023-2024 AIM GmbH <info@aim-online.com>
***
*********************************************************************
action [sample, help, list] ?
server [local] ?
module number [0..31] ?
stream number [1..8] ?
```

3.3 ANET-MxAy

The ANET-MxAy can be accessed from Windows or Linux by specifying the server at the start of the Python sample. Note that the Python path in the ANET-MxAy BSP is different from the example above. The path is "/BSP_DIR/Python API/mil".

Starting with ANET-MxAy BSP 3.0.0 the Python sample can also be executed directly on the ANET-MxAy. The *aim_mil* package is already pre-installed. The Python sample can be copied onto the ANET via SSH and can then be executed on the device locally. For local access on the ANET specify server local, module 0 and stream 1.

```
/BSP_DIR/Python API/mil$ scp -r sample user@ANETMyAy-SN.local:/home/user/
/BSP_DIR/Python API/mil$ ssh user@ANETMyAy-SN.local
/home/user$ sudo chmod a+rw /dev/aim_mil0
/home/user$ cd sample
/home/user/sample$ python 1553\_sample.py
```

3.4 ANET1553

The ANET1553 can be accessed from Windows or Linux by specifying the server at the start of the Python sample. It is currently not possible to run the Python 3 samples on the ANET1553 locally. An update from Python 2 to Python 3 on the ANET1553 is currently not planned.

4 BSP Tool

The BSP Tool is a Python 3 QT GUI that can be used to show and check the versions of installed devices. The BSP Tool needs to be installed with the Python PIP command. The tool requires the aim_mil wheel package to be installed.

Use the following steps to install the BSP Tool and the required dependencies:

Note:
The python interpreter must be in the PATH variable to follow the steps below.

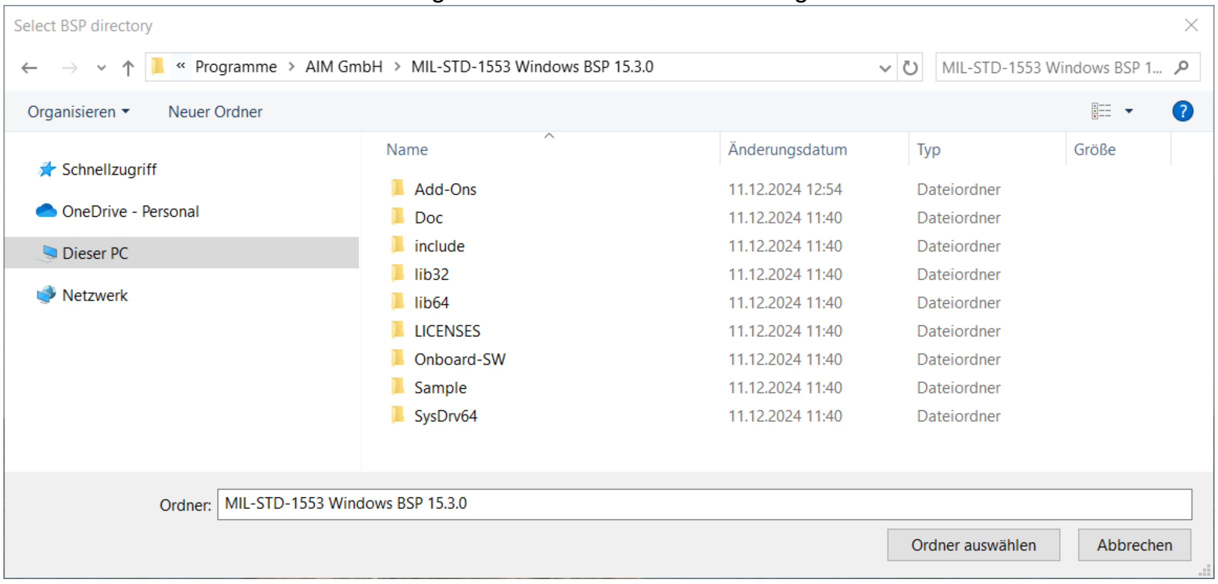
1. Install the *aim_mil* wheel package:
   cd BSP_DIR/add-ons/python
   python -m pip install aim_mil-24.x.y-py3-none-any.whl
2. Install the *BSP Tool* wheel package:
   cd BSP_DIR/add-ons/bsp_tool
   python -m pip install aim_mil_bsp_tool-15.x.y-py3-none-any.whl

   Note:
   This will also install required dependency packages like PyQT5

   Note:
   On Linux systems like Ubuntu the automatically installed PyQT5 package might raise errors when starting the BSP Tool. It is recommended to install the system python3-pyqt5 package before installing the BSP Tool.

After installing the bindings and the BSP Tool, the program can be started by executing *aim_mil_bsp_tool_gui* from a command line. A folder select dialog will open and the BSP containing the expected versions should be selected.

Figure 1: Select BSP folder dialog



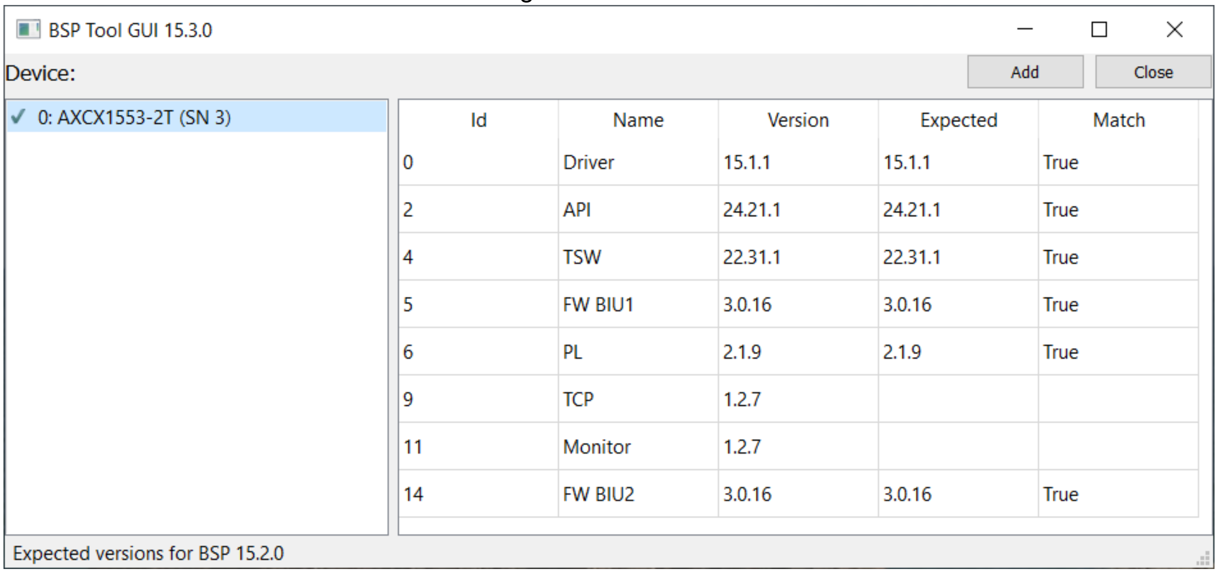
Name	Änderungsdatum	Typ	Größe
Add-Ons	11.12.2024 12:54	Dateiordner	
Doc	11.12.2024 11:40	Dateiordner	
include	11.12.2024 11:40	Dateiordner	
lib32	11.12.2024 11:40	Dateiordner	
lib64	11.12.2024 11:40	Dateiordner	
LICENSES	11.12.2024 11:40	Dateiordner	
Onboard-SW	11.12.2024 11:40	Dateiordner	
Sample	11.12.2024 11:40	Dateiordner	
SysDrv64	11.12.2024 11:40	Dateiordner	

Ordner: MIL-STD-1553 Windows BSP 15.3.0

Note:
Only BSPs with version 15.3.0 or newer contain the required version information for the tool to work.

After selecting the BSP folder the tool will open and will show all local devices.

Figure 2: BSP Tool



Id	Name	Version	Expected	Match
0	Driver	15.1.1	15.1.1	True
2	API	24.21.1	24.21.1	True
4	TSW	22.31.1	22.31.1	True
5	FW BIU1	3.0.16	3.0.16	True
6	PL	2.1.9	2.1.9	True
9	TCP	1.2.7		
11	Monitor	1.2.7		
14	FW BIU2	3.0.16	3.0.16	True

Expected versions for BSP 15.2.0

On the left part of the window is a list of the devices found. If all versions match for a device, a green check mark is displayed to the left.

The right part of the window shows a table with the versions read from the device and the expected versions for the selected BSP.

The bottom left corner shows which BSP version was selected during the startup.

To add ANET devices the button Add in the upper right corner can be used. An input dialog will open that needs to be filled with the network board name or the IP address of the ANET device.

The close button on the upper right corner can be used to close the BSP Tool.

List of Abbreviations

API	Application Programming Interface
cPCI	Compact PCI
PCI	Peripheral Component Interconnect
PCIe	PCI Express
PMC	PCI Mezzanine Card
USB	Universal Serial Bus
VME	Versa Module Eurocard
VXI	VME eXtensions for Instrumentation