MIL-STD-1553

Linux BSP

Avionics Databus Solutions

Getting Started

Version 15.3.0
December, 2024

MIL-STD-1553

Linux BSP

Getting Started

Version 15.3.0
December, 2024

AIM NO. 60-11220-35-15.3.0

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

1.1 About This Manual

This Getting Started Manual has been developed to assist first time users with software installation, hardware set up, and starting a sample project for the MIL-STD-1553 boards.

Step by step instructions cover MIL-STD-1553 Linux Board Software Package (BSP) installation, board installation, and driver adaptation. The BSP provides users with current documentation, board drivers, and sample development projects. Additionally, the BSP provides utility files that are used to upgrade the MIL-STD-1553 on-board firmware, which is also described in this manual. The current version of all BSP is available from the download area of the AIM website.

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.

1.2 How This Manual Is Organized

This manual is organized as follows:

Section 1 "Introduction" contains an overview of this manual.

Section 2 "Requirements" lists all the requirements the system must meet for the BSP.

Section 3 "BSP And Hardware Installation Procedure" describes the procedure used to install the MIL-STD-1553 Linux BSP software and hardware.

Section 4 "Connecting the AIM device to the Milbus" contains a description and diagrams of the different methods used to connect the AIM 1553 card to the Milbus.

Section 5 "Using MIL-STD-1553 API library" contains the procedure used to build, and run a sample project included in the BSP.

Section 6 "Recompilation And Reinstallation Of The Driver" contains the procedure necessary when the system's kernel is changed.

Section 7 "Signing the AIM Driver for UEFI Secure Boot" contains the procedure for Signing the AIM Driver for UEFI Secure Boot.

2 Requirements

To install the BSP, your system must meet the following requirements.

- An installed x86-64 (64-Bit) Linux Distribution.
- Linux standard kernel versions from 3.2 to 5.15 are supported. With newer or older versions you might experience compilation problems. Please note that Red Hat Enterprise Linux (RHEL) and its derivates CentOS and Rocky Linux contain a modified Linux kernel. The driver might not build even if it shows a supported version number. Please check the Release Notes for information on which Linux distributions and versions have been tested with this BSP.
- The Linux kernel must be configured to provide module support.
- Header files of the kernel must be installed:
  Ubuntu: *sudo apt-get install linux-headers-$(uname -r)*
  RHEL: *sudo yum install kernel-devel kernel-headers*
- The system must provide the *libnl-genl3* library and corresponding header files:
  Ubuntu: *sudo apt-get install libnl-genl-3-200 libnl-genl-3-dev*
  RHEL: *sudo yum install libnl3-devel*
- The system must provide a Python interpreter:
  Ubuntu: *sudo apt-get install python3*
  RHEL: *sudo yum install python3*
- The system must provide GNU make and libtool:
  Ubuntu: *sudo apt-get install make automake libtool pkg-config*
  RHEL: *sudo yum install make which patch elfutils-libelf-devel*
- The system must provide GNU C/C++ compilers:
  Ubuntu: *sudo apt-get install gcc g++*
  RHEL: *sudo yum install gcc gcc-c++*

3 BSP And Hardware Installation Procedure

This section contains the procedure used to install the MIL-STD-1553 Linux BSP software and the hardware.

The BSP contains reference documentation, board drivers, sample development projects and upgrade utility files. These files are automatically loaded onto the system by installing the software package.

The software is available on DVD from AIM or via download.

3.1 BSP Installation

To install the MIL-STD-1553 Linux BSP software from the AIM provided DVD or a download from AIM-Online

1. Uninstall all older MIL-STD-1553 Linux BSPs according to Section 3.4 "Uninstalling The BSP".
2. Change into the directory where you want the BSP and examples to be installed, e.g. your home directory. Then copy the file mil-std-1553-linux-bsp.tar.gz to it.
3. Unpack it with *'tar xfz mil-std-1553-linux-bsp.tar.gz'*
4. Change into the subdirectory *./mil-std-1553-linux-bsp-VERSION/*
5. Run *'sudo ./install.sh'* to compile and install the BSP as superuser.

Note:
Some kernels may require that kernel modules are signed by a key trusted by the kernel in order to be loaded. In this case please have a look to section 7

Note:
In case of an error please make sure that all requirements in section 2 are met and try again before contacting support. If the error is still present please include the file build.log from the current directory in the support case.

The installation will create these files on your system.

/lib/modules/<kernel>/updates/aim_mil.ko
The AIM MIL-STD-1553 PCI device driver compiled for your kernel configuration

/lib/modules/<kernel>/updates/aim1553usb.ko
The AIM 1553 USB device driver compiled for your kernel configuration

/usr/local/lib/libaim_mil.*
The AIM MIL-STD-1553 API library

/usr/local/include/aim_mil*
The AIM MIL-STD-1553 API header files

Note:
The kernel modules directory updates was called extra until vanilla kernel version 6.3.

3.2 Updating the On-board Firmware

General

The installation script updates all boards to the correct firmware version, if allowed.

Note:
A manual update is normally not necessary.

ANET1553 firmware upgrade

The ANET boards cannot be updated with the standard method. They require a separate update procedure, which is described in the ANET Users Manual. Open the ANET web interface update tab and choose boot into maintenance mode. Login to the maintenance web interface an go back to the update tab. Choose the update package (*.tar.gz) for your device and start the update. The update package is located in the BSPs Onboard-SW/update-components folder.

The update might take some time. Please leave the browser window open until the update is finished.

Firmware upgrade of other boards

For a manual update change to the *Onboard-SW* directory beneath the BSPs root directory. Run the **update.sh** file and follow the instructions of the script.

When the upgrade of the firmware is finished, a reboot is required to start the new on-board firmware.

Note:
For some boards an additional reboot may be required. Please check the output on the command prompt and reboot and run the **update.sh** again, if requested.

3.3 Changing Device File Permissions

Each AIM PCI device is represented by a separate device file in your system's */dev* directory, e.g. */dev/aim_mil0* for the first device. Accordingly, each AIM USB device is represented by device files called */dev/aim1553usb0*, */dev/aim1553usb1* and so on. These device files are automatically created by your distribution's *udev* daemon. By default, these device files are created with root as owner and belong to group root. This means, only applications with root rights are able to access the AIM devices. You are able to change these permissions by adding specific rules to your udev configuration. On most systems this can be done by placing 'rules' files into the */etc/udev/rules.d* directory.

For example, create a file called *99-aim_mil.rules* in this directory and write all the AIM device file rules you need into this file.

Each line of the file can contain one specific udev rule.

E,g. if you want to grant all users on the system read and write access to the AIM devices write the rules:

```
ACTION=="add", DRIVERS=="aim_mil", MODE="0666"
ACTION=="add", DRIVERS=="aim1553usb", MODE="0666"
```

Note:
without such a rule normal users don't have the permissions to access the AIM devices

After the new rules have been added reload the rules and the driver or perform a reboot of the system.

```
sudo udevadm control --reload-rules
sudo udevadm trigger
sudo rmmod aim_mil
sudo rmmod aim1553usb
sudo modprobe aim_mil
sudo modprobe aim1553usb
```

Please consult your distributions udev documentation for detailed information on udev rules.

3.4 Uninstalling The BSP

To uninstall the BSP, cd into *./mil-std-1553-linux-bsp-VERSION/* and run *'sudo ./install.sh uninstall'* to uninstall the BSP as superuser. This will remove all the files created during the installation.

4 Connecting the AIM device to the Milbus

The MIL-STD-1553 Specification clearly defines the process of coupling subsystems to the bus. This connection, called a stub, has two coupling options: Direct and Transformer Coupled. In addition to these two methods of connecting to the bus, a Direct to Device (Network) Coupling configuration is also an option. All three methods are described in this section.

4.1 Direct coupling

Direct Coupling connects the subsystem/terminal device directly to the bus and can only be used in connections under one foot in length.

Since a Direct Coupled stub provides only limited isolation in the event of a device (subsystem or terminal) short, Transformer Coupling is normally the recommended method of connecting to the bus.

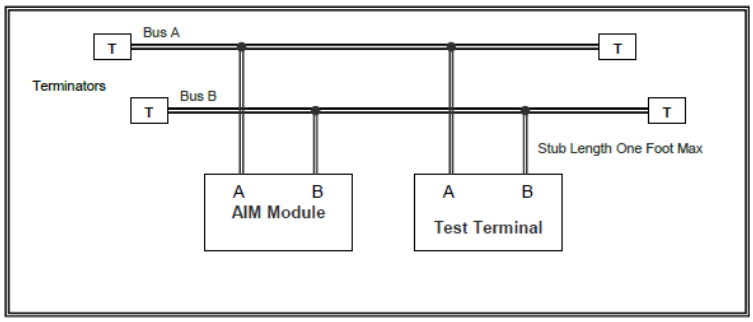


Figure 1: Direct coupling

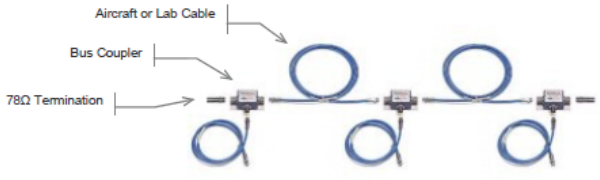
4.2 Transformer coupling

Transformer Coupling utilizes a bus coupler that contains an isolation transformer and isolation resistors. Transformer Coupling extends the stub length to 20 feet and provides electrical isolation, better impedance matching and higher noise rejection characteristics than Direct Coupling. The electrical isolation prevents a terminal fault or stub impedance mismatch from affecting bus performance.

Connecting to the bus using Transformer Coupling requires a Coupler for each subsystem/terminal device and proper termination on the bus.

4.3 Direct to device (Network) coupling

Direct to Device (Network) Coupling is an option with AIM modules. The AIM design includes onboard Bus Network circuitry that is software selectable. This allows the user to connect directly to a single terminal device without the need for any bus coupling. The Network Coupling mode, selected by the user via software controls, provides a terminated MILbus network simulation on the AIM module for direct connection between the AIM module/PBA Analyzer and the terminal under test.



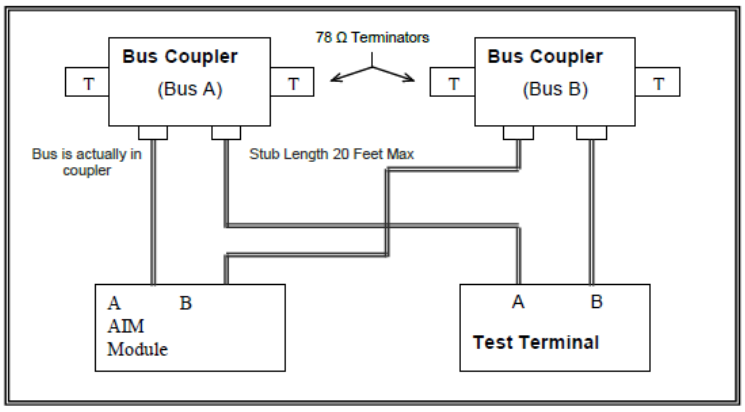


Figure 2: Transformer coupling

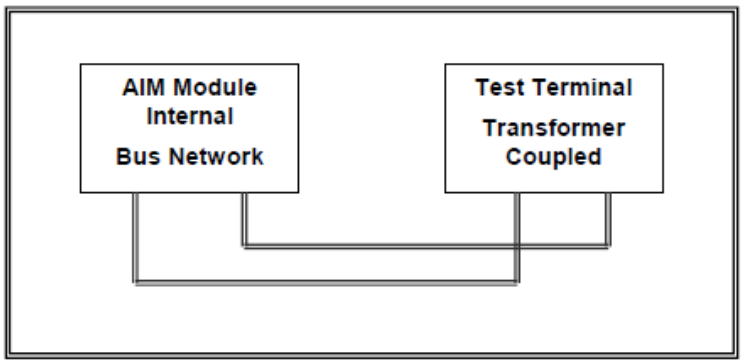


Figure 3: Direct to device (Network) coupling

5 Using MIL-STD-1553 API library

5.1 Developing Custom C/C++ Applications

This section describes how to build a C or C++ based Linux application that makes use of the AIM MIL-STD-1553 library. In this demonstration, the GNU make build system is used. If you are using other build systems, the procedure should be analogous.

1. Include the *Api1553.h* header file in your application's modules that want to make use of API functionality:

```
#include "Api1553.h"

int main(void)
{
    AiInt num_boards = 0;
    num_boards = ApiInit();
    printf("%d boards found.\n", num_boards);

    ...
}
```

2. Add the */usr/local/include/aim_mil* directory to your preprocessor include path and set the preprocessor macro *_AIM_LINUX*. This can be done in your projects *Makefile* using the *CPPFLAGS* variable:

```
CPPFLAGS += -I/usr/local/include/aim_mil -D_AIM_LINUX
```

3. Add the library *libaim_mil* and the library search path */usr/local/lib* to your linker flags. This can be done in your projects *Makefile* using the *LDFLAGS* variable:

```
LDFLAGS += -laim_mil -L/usr/local/lib
```

4. Your application should now successfully build and link against the *libaim_mil* library. Following a complete example *Makefile* that is building an application called *sample* from the source file *main.c*:

```
# Setting preprocessor and linker flags
CPPFLAGS += -I/usr/local/include/aim_mil -D_AIM_LINUX
LDFLAGS += -laim_mil -L/usr/local/lib

# Main make target to build the application 'sample'
```

```
sample : main.o
$(CC) $(CPPFLAGS) $(LDFLAGS) -o sample main.o

# Clean-up make target
clean :
$(RM) sample main.o
```

5.2 The Sample Project

The BSP installation procedure will install a sample application to */usr/local/bin/sample*. This program prototypes some basic use cases for the AIM MIL-STD-1553 Application Programming Interface (API).

If */usr/local/bin* is in your OS path variable the sample can be executed by entering the command *sample* on the command line.

If the sample fails to load because *libaim_mil.so.24* can't be found, then enter *'export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/usr/local/lib/'* on the shell before the sample is launched.

To make the library search path permanent, create a file */etc/ld.so.conf.d/aim.conf* with a line *'/usr/local/lib'* and execute *'sudo ldconfig'*. Then try again.

Sources for this sample application can be found in the BSP subdirectory *sample*. The program consists of several use cases each implemented in a separate source file in the *sample/src* directory. The source code can be adapted to your own needs. In order to re-build the sample application, change into the subdirectory *sample* and run *'sudo make'*. The binary will then be located in *sample/src*. If you want to replace the sample application */usr/local/bin/sample* built during BSP installation, run *'sudo make install'*.

6 Recompilation And Reinstallation Of The Driver

A recompilation of the driver module may be necessary if the kernel configuration has changed fundamentally, e.g. either a different kernel version is used or the kernel is switched from uniprocessor to SMP. To rebuild the driver, re-run the BSP installation script *install.sh* in the BSPs root directory.

7 Signing the AIM Driver for UEFI Secure Boot

7.1 Generate signature file (public/private keys)

Install the following packages to be able to do the signing

Ubuntu: *sudo apt-get install openssl keyutils mokutil*
RHEL: *sudo yum install openssl keyutils mokutil*

Generate the signature files

- Navigate to BSP directory (location of install.sh)
- In the terminal:
  sudo openssl req -new -x509 -newkey rsa:2048 -keyout MOK.priv -outform DER -out MOK.der -nodes -days 36500 -subj "/CN=AIMUser/"

7.2 Register it for the Secure Boot

- *sudo mokutil –import MOK.der*
- This will ask you to create a new password. . . then confirm it
- Finally, restart the computer. A "Perform MOK management" screen will appear with a keyboard wait, press the key that asks you to interrupt the boot.
- Enroll MOK > Continue > and it will ask you for the password
- reboot
- After the system reboots, verify the new key is active with the following command
  sudo keyctl list %:.platform

7.3 BSP installation with signing

- Run *'sudo ./install.sh –enable-signing'* to compile and install the BSP as superuser

List of Abbreviations

API Application Programming Interface

BSP Board Software Package

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