



## ADB3 Driver 1.4.27 for Linux Release Note

### Introduction

This release note accompanies the ADB3 Driver for Linux. The latest version of this driver can be found at:

<https://support.alpha-data.com/pub/admxrcg3/linux>

For support, send e-mail to [support@alpha-data.com](mailto:support@alpha-data.com)

### Operating systems supported

This release of the ADB3 Driver is believed to be compatible with (at least) the following Linux distributions:

- Red Hat Enterprise Linux 8, 9 and 10 and downstream derivatives such as Rocky Linux and AlmaLinux.
- Centos 7.9 with latest available kernel.
- Fedora Linux 33 to 43.
- Ubuntu Linux 18 to 25.
- OpenSUSE Tumbleweed 2026.

The range of Linux kernel versions supported covers 3.x, 4.x.y, 5.x.y & 6.x.y.

However, the Linux kernel is constantly changing in ways that sometimes break out-of-tree device drivers. Alpha Data cannot guarantee that this release of the ADB3 Driver can be successfully built and installed and will work correctly on all versions of all Linux distributions past, present and future. Alpha Data makes best efforts to ensure compatibility with all Linux distributions, but should a problem be encountered, please contact [support@alpha-data.com](mailto:support@alpha-data.com).

### Hardware supported

This release of the ADB3 Driver supports the following Alpha Data hardware:

- ADM-XRC-6TL
- ADM-XRC-6T1
- ADM-XRC-6T-DA1
- ADM-XRC-6TGE and ADM-XRC-6TGEL
- ADM-XRC-6T-ADV8
- ADPE-XRC-6T and ADPE-XRC-6T-L
- ADPE-XRC-6T-ADV
- ADM-XRC-7K1
- ADM-XRC-7V1
- ADM-VPX3-7V2
- ADM-PCIE-7V3
- ADM-PCIE-KU3
- ADM-XRC-KU1 and ADM-XRC-KU1A-T8

- ADM-PCIE-8V3
- ADM-PCIE-8K5
- ADM-AD01478
- ADM-AD01479

## License Agreement

This release of software is licensed according to the terms of GNU Public License Version 2 (GPL V2). A copy of this license can be found in the file **gpl-2.0.txt** within this software package. Please contact Alpha Data if alternative licensing conditions are required.

Alpha Data reserves the right to use a different license agreement for future releases of this software.

## Building and installing

There are two methods for building and installing the ADB3 Driver. They are:

- (a) Build and install the driver for the current running kernel using the Makefiles provided. This is described in [Build and install for the current running kernel](#).
- (b) Build and install everything except the kernel module, using the Makefiles provided, and make DKMS responsible for building and installing the kernel module automatically. This is described in [Build and install with the kernel module under control of DKMS](#).

### Note

Please ensure that, if changing from one method of installation to the other, the appropriate method of uninstallation is used before doing so; see [Uninstalling](#). Failing to uninstall the ADB3 Driver appropriately when changing the installation method can result in the kernel module (**adb3.ko**) existing in two locations in the kernel's module tree, resulting in uncertainty about which kernel module is in use.

## Build and install for the current running kernel

Building and installing for the current running kernel is a two-stage process:

- (i) Build the kernel module and API shared libraries as a non-privileged user.
- (ii) Install the kernel module, **udev** rules files and API shared libraries as a privileged user. The account used to perform this step must be sudo-capable.

Perform the following steps:

1. Change directory to the one containing this document, where you have unpacked the ADB3 Driver package. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
```

2. Issue the following command to build the kernel module (**adb3.ko**) and API shared libraries:

```
make clean all
```

3. Issue the following command to install the kernel module, **udev** rules files and API shared libraries:

```
sudo make install
```

At this point, the kernel module may be loaded by issuing a command such as:

```
sudo modprobe adb3
```

This method of installation installs the following in system directories:

Type	File	Destination directory
File	<b>51-adb3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>51-admxrc3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>libadb3.so.1.4.27</b>	Native shared library directory*
Symbolic link	<b>libadb3.so.1</b>	Native shared library directory*
Symbolic link	<b>libadb3.so</b>	Native shared library directory*
File	<b>libadmxrc3.so.1.4.27</b>	Native shared library directory*
Symbolic link	<b>libadmxrc3.so.1</b>	Native shared library directory*
Symbolic link	<b>libadmxrc3.so</b>	Native shared library directory*
File	<b>adb3.ko</b>	<b>/lib/modules/&lt;kernel version&gt;/kernel/drivers/addon/adb3/</b>

**Table 1 : Installed files and symbolic links**

\* The path of the native shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

## Bi-architecture build

In a machine of **x86\_64** (AMD64) architecture, provided that the Linux distribution in use has not phased out 32-bit support (see below), it is possible to build and install the API shared libraries "bi-architecture". This means that they are built and installed not only for the 64-bit (native) architecture, but also for the 32-bit (compatibility) architecture. This enables 32-bit to use the ADB3 Driver.

In order to build and install the API shared libraries bi-architecture, the **make** commands previously given above must be appended with **BIARCH=yes**. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
make clean all BIARCH=yes
sudo make install BIARCH=yes
```

Note that, as of the date of publication of this document, many Linux distributions have either phased out 32-bit support or have declared intent to do so in the near future. Because this removes 32-bit versions of dependencies required by the ADB3 Driver, such as **glibc** packages, the **BIARCH=yes** option cannot be used in a Linux distribution that has phased out 32-bit support.

Performing a bi-architecture installation creates the following in system directories:

Type	File	Destination directory
File	<b>51-adb3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>51-admxrc3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>libadb3.so.1.4.27 (32-bit)</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadb3.so.1</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadb3.so</b>	Compatibility (32-bit) shared library directory*
File	<b>libadmxrc3.so.1.4.27 (32-bit)</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadmxrc3.so.1</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadmxrc3.so</b>	Compatibility (32-bit) shared library directory*
File	<b>libadb3.so.1.4.27 (64-bit)</b>	Native (64-bit) shared library directory**
Symbolic link	<b>libadb3.so.1</b>	Native (64-bit) shared library directory**

**Table 2 : Installed files and symbolic links (bi-architecture build) (continued on next page)**

Type	File	Destination directory
Symbolic link	<b>libadb3.so</b>	Native (64-bit) shared library directory**
File	<b>libadmxc3.so.1.4.27</b> (64-bit)	Native (64-bit) shared library directory**
Symbolic link	<b>libadmxc3.so.1</b>	Native (64-bit) shared library directory**
Symbolic link	<b>libadmxc3.so</b>	Native (64-bit) shared library directory**
File	<b>adb3.ko</b>	<b>/lib/modules/&lt;kernel version&gt;/kernel/drivers/addon/adb3/</b>

**Table 2 : Installed files and symbolic links (bi-architecture build)**

\* The path of the compatibility (32-bit) shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

\*\* The path of the native (64-bit) shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

## Build and install with the kernel module under control of DKMS

Dynamic Kernel Module Support (DKMS) is a framework for ensuring that out-of-tree Linux kernel modules are automatically rebuilt when a new Linux kernel package is installed. The DKMS project can be found at

<https://github.com/dkms-project/dkms>

In order to use DKMS for managing the building of out-of-tree kernel modules, the appropriate packages that provide DKMS for the Linux distribution in use must be installed. For example, in Red Hat Enterprise Linux and downstream distributions, the following command installs DKMS:

```
sudo dnf install dkms
```

The scope of DKMS is generally limited to building out-of-tree kernel modules, and not any user-mode components such as shared libraries and utilities that might be associated with those kernel modules. This is because installing an updated kernel package in an installation of a Linux distribution generally has no effect on user-mode components, so they do not need to be rebuilt after a kernel update.

### Automatic rebuild of a kernel module using DKMS might sometimes fail

The purpose of DKMS is to remove the need to manually rebuild an out-of-tree kernel module after a kernel upgrade, thus reducing the probability of an outage relating to that kernel module. Because the Linux kernel has no stable API upon which drivers can rely, DKMS might sometimes fail to rebuild a given kernel module after a kernel upgrade. In that event, the kernel module in question must be removed from DKMS and replaced with an updated version that fixes the compilation failure.

Building and installing for the current running kernel using DKMS is a three-stage process:

- (i) Build the kernel module and API shared libraries as a non-privileged user.
- (ii) Install the **udev** rules files and API shared libraries as a privileged user. The account used to perform this step must be sudo-capable.
- (iii) Place the ADB3 Driver's kernel module (**adb3.ko**) under control of DKMS, and instruct DKMS to perform an initial build and installation for the current running kernel. The account used to perform this step must be sudo-capable.

Perform the following steps:

1. Change directory to the one containing this document, where you have unpacked the ADB3 Driver package. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
```

2. Issue the following commands to build the API shared libraries:

```
make clean all
```

- Issue the following command to install the **udev** rules files and API shared libraries:

```
sudo make dkms_install
```

- Issue the following command to place the ADB3 Driver's kernel module (**adb3.ko**) under control of DKMS, building and installing the kernel module for the current running kernel:

```
sudo dkms install ./dkms.conf
```

From this point on, using hooks provided by the Linux distribution's package manager (**apt**, **dnf**, **yum** etc.), DKMS will rebuild and reinstall the kernel module when a new kernel package is installed.

At this point, the kernel module may be loaded by issuing a command such as:

```
sudo modprobe adb3
```

This method of installation installs the following in system directories:

Type	File	Destination directory
File	<b>51-adb3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>51-admxrc3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>libadb3.so.1.4.27</b>	Native shared library directory*
Symbolic link	<b>libadb3.so.1</b>	Native shared library directory*
Symbolic link	<b>libadb3.so</b>	Native shared library directory*
File	<b>libadmxrc3.so.1.4.27</b>	Native shared library directory*
Symbolic link	<b>libadmxrc3.so.1</b>	Native shared library directory*
Symbolic link	<b>libadmxrc3.so</b>	Native shared library directory*
File	<b>adb3.ko</b> or <b>adb3.ko.xz</b>	Somewhere under <b>/lib/modules/&lt;kernel version&gt;/</b> ; depends upon how Linux distribution in use configures DKMS.

**Table 3 : Installed files and symbolic links (with DKMS)**

\* The path of the native shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

## Bi-architecture build (using DKMS)

In a machine of **x86\_64** (AMD64) architecture, provided that the Linux distribution in use has not phased out 32-bit support (see below), it is possible to build and install the API shared libraries "bi-architecture". This means that they are built and installed not only for the 64-bit (native) architecture, but also for the 32-bit (compatibility) architecture. This enables 32-bit to use the ADB3 Driver.

In order to build and install the API shared libraries bi-architecture, the **make** commands previously given above must be appended with **BIARCH=yes**. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
make clean all BIARCH=yes
sudo make dkms_install BIARCH=yes
sudo dkms install ./dkms.conf
```

Note that, as of the date of publication of this document, many Linux distributions have either phased out 32-bit support or have declared intent to do so in the near future. Because this removes 32-bit versions of dependencies required by the ADB3 Driver, such as **glibc** packages, the **BIARCH=yes** option cannot be used in a Linux distribution that has phased out 32-bit support.

Performing a bi-architecture installation creates the following in system directories:

Type	File	Destination directory
File	<b>51-adb3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>51-admxrc3.rules</b>	<b>/etc/udev/rules.d</b>
File	<b>libadb3.so.1.4.27</b> (32-bit)	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadb3.so.1</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadb3.so</b>	Compatibility (32-bit) shared library directory*
File	<b>libadmxrc3.so.1.4.27</b> (32-bit)	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadmxrc3.so.1</b>	Compatibility (32-bit) shared library directory*
Symbolic link	<b>libadmxrc3.so</b>	Compatibility (32-bit) shared library directory*
File	<b>libadb3.so.1.4.27</b> (64-bit)	Native (64-bit) shared library directory**
Symbolic link	<b>libadb3.so.1</b>	Native (64-bit) shared library directory**
Symbolic link	<b>libadb3.so</b>	Native (64-bit) shared library directory**
File	<b>libadmxrc3.so.1.4.27</b> (64-bit)	Native (64-bit) shared library directory**
Symbolic link	<b>libadmxrc3.so.1</b>	Native (64-bit) shared library directory**
Symbolic link	<b>libadmxrc3.so</b>	Native (64-bit) shared library directory**
File	<b>adb3.ko</b> or <b>adb3.ko.xz</b>	Somewhere under <b>/lib/modules/&lt;kernel version&gt;/</b> ; depends upon how Linux distribution in use configures DKMS.

**Table 4 : Installed files and symbolic links (bi-architecture build with DKMS)**

\* The path of the compatibility (32-bit) shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

\*\* The path of the native (64-bit) shared library directory, normally somewhere under **/usr/**, varies between Linux distributions.

## Uninstalling

In order to uninstall the driver, the appropriate method must be used, dependent upon what method was used to install the driver:

- If the driver was built and installed as per [Build and install for the current running kernel](#), see [Uninstall for the current running kernel](#) for uninstallation instructions.
- If the driver was built and installed as per [Build and install with the kernel module under control of DKMS](#), see [Uninstall with the kernel module under control of DKMS](#) for uninstallation instructions.

## Uninstall for the current running kernel

If the ADB3 Driver was previously installed for the current running kernel as per [Build and install for the current running kernel](#), it may be uninstalled as follows:

- If the kernel module (**adb3.ko**) is loaded, unload it using a command such as:

```
sudo rmmod adb3
```

- Change directory to the one containing this document, where you have unpacked the ADB3 Driver package. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
```

- Issue the following command to uninstall the kernel module, the **udev** rules files and the API shared libraries:

```
sudo make uninstall
```

Note that this works whether or not the API shared libraries were built bi-architecture (see [Bi-architecture](#)

build).

## Uninstall with the kernel module under control of DKMS

If the ADB3 Driver was previously installed with the kernel module under control of DKMS, as per [Build and install with the kernel module under control of DKMS](#), it may be uninstalled as follows:

1. If the kernel module (**adb3.ko**) is loaded, unload it using a command such as:

```
sudo rmmod adb3
```

2. Change directory to the one containing this document, where you have unpacked the ADB3 Driver package. For example:

```
cd /path/to/adb3-driver-linux-1.4.27
```

3. Issue the following command to uninstall the **udev** rules files and the API shared libraries:

```
sudo make uninstall
```

Note that this works whether or not the API shared libraries were built bi-architecture (see [Bi-architecture build \(using DKMS\)](#)).

4. Issue the following command to remove the ADB3 Driver's kernel module from control of DKMS, for all kernels in the system:

```
sudo dkms remove adb3/1.4.27 --all
```

## Notes on usage

### Downgrading to an earlier version

Before downgrading to an earlier version of the ADB3 Driver, uninstall ADB3 Driver 1.4.27 or later as per [Uninstalling](#).

### udev rules files

The **udev** service is required in order to create the device nodes in **/dev/** that represent the user-mode interface of the driver. Practically all modern Linux distributions include the **udev** service.

#### The adb3.rc script is no longer provided or required

Starting with ADB3 Driver 1.4.27, the **adb3.rc** script that was provided in previous releases is no longer provided, as it was a source of confusion about how to load the kernel module (**adb3.ko**). Since ADB3 Driver 1.2.1, it is sufficient to load the kernel module with the following command:

```
sudo modprobe adb3
```

The **udev** service uses rules files, e.g. **51-admxrc3.rules** in the case of the ADB3 Driver, to create device nodes automatically when a kernel module (that exports character device interfaces) is loaded using **modprobe** or **insmod**.

The rules files may be customized to change the permissions and/or owner and/or group of the device nodes **/dev/admxrc3\***. See [Security considerations](#) below for more information.

## Security considerations

By default, the **udev** rules file **51-admxrc3.rules** creates device nodes in **/dev** as follows:

- Mode: 664 (octal) => owner read, owner write, group read, group write, other read
- UID: **root**
- GID: **root**



This means that the default permissions are that (i) only **root** and members of the group **root** can open devices, in read-only or read-write mode and (ii) users that are not members of the group **root** can open a device in read-only mode. However, after installing the driver, the file **/etc/udev/rules.d/51-admxrc3.rules** can be customized to relax permissions. See the comments in that file for details.

## VPD write-protection mechanism

The VPD write-protection mechanism described in the ADM-XRC Gen 3 SDK User Guide is implemented as of release 1.1.0. To enable write-to-VPD, the kernel module parameter **EnableVpdWrite** must be nonzero.

This value takes effect when the driver starts, so it can be changed only by unloading the driver and restarting the driver with a different value for **EnableVpdWrite**. If this parameter is not specified, the driver considers it to be zero (write-to-VPD disabled).

## Common buffer support

Beginning with release 1.4.4, the driver can create one or more "common buffers" at startup. The main purpose of this feature is supporting applications that use Direct Master data transfer, such as an ethernet-style I/O interface where the sizes and arrival times of packets of data are not known in advance by software running on the host. These buffers have the following characteristics:

- Persist until the driver is stopped.
- Guaranteed aligned to a specified power-of-2 address boundary.
- Allocated from the appropriate pool of memory in order to be contiguous and visible to bus-master devices.
- Can be mapped into the virtual address space of a user-mode process; see "ADMXRC3 API Specification 1.5.0" or later for details of the new ADMXRC3 API functions relating to common buffers.

The driver parameter **PrimaryCommonBufferCount** determines the number of common buffers allocated; the default is zero, meaning that no common buffers are allocated by default. The parameter

**PrimaryCommonBufferSizeLow** determines the size in bytes of each common buffer; the default is 64 kiB (0x10000). The parameter **PrimaryCommonBufferAlignment** determines the address boundary size to which each common buffer is guaranteed to be aligned; the default is 16 bytes (0x10).

## Preferred read and write width

Starting with release 1.4.25, the preferred word width for load instructions executed as a result of calling **ADMXRC3\_Read** can be specified via the **PreferredReadWidth** driver parameter (default 0). If given a value of 1, 2, 4 or 8, this parameter specifies the word width used, in bytes, when no **ADMXRC3\_ACCESS\_WIDTH\_\*** flag is passed to **ADMXRC3\_Read** in the **flags** parameter.

Similarly, the preferred word width for store instructions executed as a result of calling **ADMXRC3\_Write** can be specified via the **PreferredWriteWidth** driver parameter (default 0). If given a value of 1, 2, 4 or 8, this parameter specifies the word width used, in bytes, when no **ADMXRC3\_ACCESS\_WIDTH\_\*** flag is passed to **ADMXRC3\_Write** in the **flags** parameter.

These parameters take effect when the driver starts, so they can be changed only by unloading the driver and restarting the driver with a different value for **PreferredReadWidth** and/or **PreferredWriteWidth**. A value of 0 for these driver parameters means that the actual word size used is dependent on the implementation of the memory copy primitives provided by the Linux kernel, which is the behavior of ADB3 Driver 1.4.24 and earlier.

## Fixed-local addressing DMA transfers

The flag **ADMXRC3\_DMA\_FIXEDLOCAL** when used with the DMA functions in the ADMXRC3 currently has no effect for Gen 3 hardware.



# Release history

## Release 1.4.27

This release implements ADMXRC3 API Specification version 1.8.4.

### Enhancements:

- 1 Added a **dkms.conf** file for Dynamic Kernel Module Support (DKMS). This allows the ADB3 Driver's kernel module to be automatically rebuilt and reinstalled after a kernel upgrade.
- 2 Overhauled the build system to work with (at least) the most commonly used Linux distributions, and to be aware of the system shared library directory conventions used in those distributions.
- 3 The build system now allows uninstallation of the driver, which removes the kernel module, shared libraries and **udev** rules files from system directories, using the **uninstall** target of the top-level Makefile.
- 4 Moved detailed build and installation instructions from the **README** file into this document.
- 5 Bi-architecture building is now selected by passing **BIARCH=yes** on the make command line.
- 6 The "configure" script, that was used to detect certain features of the Linux distribution and kernel against which the driver is being compiled, is no longer required and is no longer provided in ADB3 Driver releases.
- 7 The **udev** service must now be present in the Linux distribution that is used with the ADB3 Driver, and the **adb3.rc** script is no longer provided in ADB3 Driver releases.

### Corrections:

- 8 Fixes for compilation failures seen when building against new Linux versions such as 6.18.x.
- 9 Added a mitigation for a potential PCIe buffer overflow condition that may affect the Bridge FPGA of the ADM-XRC-KU1, ADM-XRC-KU1A-T8, ADM-AD01478 & ADM-AD01479 when configuring the Target FPGA using the API functions **ADMXRC3\_ConfigureFromFile** & **ADMXRC3\_ConfigureFromBuffer**. The symptom of this issue is the Bridge FPGA dropping from the PCIe tree until the system is rebooted.

## Release 1.4.26

This release implements ADMXRC3 API Specification version 1.8.4.

### Enhancements:

- 1 Added support for ADM-01479.
- 2 Now uses 64-bit PCIe addressing by default, for models and hardware platforms that support it, which may eliminate the need for bounce-buffering, in which case a DMA performance increased might be observed.

### Corrections:

- 3 Fixes for compilation failures seen when building against Red Hat Enterprise Linux 8.x & kernels, caused by changes from later vanilla Linux kernel versions being backported to RedHat's 4.18.0 kernels. These fixes also work for downstream Linux distributions such as Rocky Linux and AlmaLinux.
- 4 Fixes for compilation failures seen when building against Red Hat Enterprise Linux 9.x & kernels, caused by changes from later vanilla Linux kernel versions being backported to RedHat's 5.14.0 kernels. These fixes also work for downstream Linux distributions such as Rocky Linux and AlmaLinux.

## Release 1.4.25

This release implements ADMXRC3 API Specification version 1.8.4.

### Enhancements:

- 1 The new flags defined in ADMXRC3 API Specification version 1.8.4. are now recognized by the **ADMXRC3\_Read** & **ADMXRC3\_Write** API functions: **ADMXRC3\_ACCESS\_WIDTH\_8**, **ADMXRC3\_ACCESS\_WIDTH\_16**, **ADMXRC3\_ACCESS\_WIDTH\_32** & **ADMXRC3\_ACCESS\_WIDTH\_64**.
- 2 Added new driver parameters **PreferredReadWidth** & **PreferredWriteWidth**.

## Release 1.4.24

This release implements ADMXRC3 API Specification version 1.8.3.

Enhancements:

- 1 Added support for ADM-AD01478.

## Release 1.4.23

This release implements ADMXRC3 API Specification version 1.8.3.

Corrections:

- 1 Added a workaround for ADB3 Bridge bug that affects ALL models that prevents correct operation in system with Intel 3rd & 4th Generation Xeon CPUs and may crash the system when the ADB3 Driver starts. The workaround permits the driver to start without crashing the system and have sufficient functionality to permit the ADB3 Bridge firmware to be updated.
- 2 Fixes to support Linux kernel up to 6.8.11.

## Release 1.4.21.1

This release implements ADMXRC3 API Specification version 1.8.3.

Corrections:

- 1 Fixed a compilation failure seen when building the driver against a Linux kernel whose version is in the range 4.4.168 to 4.4.291 inclusive, caused by argument changes in the **get\_user\_pages()** function of the kernel.

## Release 1.4.21

This release implements ADMXRC3 API Specification version 1.8.3.

Enhancements:

- 1 Added support for ADM-XRC-KU1A-T8.

Corrections:

- 2 Various fixes for Linux kernel driver API changes up to and including Linux kernel 5.14.12.
- 3 Changed how kernel build system is invoked in Linux kernel 5.x or later, to use M= instead of SUBDIRS=. The latter will no longer be recognized in Linux kernel 5.3.x onwards.
- 4 Fixed an issue where nodes in /dev are not correctly named, lacking a numerical suffix, resulting in API calls such as **ADMXRC3\_Open** always failing, even when FPGA hardware is present and working correctly.

## Release 1.4.19

This release implements ADMXRC3 API Specification version 1.8.3.

Enhancements:

- 1 Added a function ADB3\_CommandAVR to ADB3 API. This command permits direct communication with the microcontroller on the following models: ADPE-XRC-6T(-L), ADPE-XRC-6T-ADV, ADM-XRC-7K1, ADM-XRC-7V1, ADM-VPX3-7V2.

Corrections:

- 2 Fixed a regression bug introduced in the 1.4.18 release, which prevented a DMA transfer from being aborted by any means (killing the thread that initiated it or calling **ADMXRC3\_CancelDMA**).
- 3 Fixed build failure with Linux kernel 4.11 or later due to new requirement to `#include <sched/signal.h>` in order get `signal_pending` definition.
- 4 Fixed build failure with Linux kernel 4.11 or later due to change in definition of `.fault` member of struct `vm_operations_struct`.
- 5 Fixed build failure with Linux kernel 4.13 or later due to renaming of `wait_queue_t` -> `wait_queue_entry_t`.

- 6 Fixed build failure with Linux kernel 4.15 or later due to change in kernel timer API (init\_timer -> timer\_setup).
- 7 Fixed warning in system log seen when running any application, under a 4.16 or later Linux kernel due to "hardened usercopy whitelisting" feature, which uses non-blocking ADMXRC3 API functions.

## Release 1.4.18

This release implements ADMXRC3 API Specification version 1.8.3.

### Enhancements:

- 1 For Bridge-in-Target FPGA designs, which are exposed as a device whose model code is **ADMXRC3\_MODEL\_GENERIC**, FPGA interrupts can now be consumed by user-mode applications, via API functions such as **ADMXRC3\_RegisterWin32Event** and **ADMXRC3\_StartNotificationWait**.

### Corrections:

- 2 Corrected maximum number of DMA engines exposed for ADM-XRC-KU1-P5HI IP; was 2 (incorrect), now 4 (corrected).
- 3 Corrected incorrect package returned by ADMXRC\_GetFpgaInfo(A,W) for ADM-XRC-KU1 fitted with KU060 FPGA; was FLVA1517 (incorrect), now FFVA1517 (corrected).
- 4 For ADM-XRC-7Z1 only, corrected failure to restore FPGA interrupt enable on power-up; symptom is that after powering up from a low-power state, FPGA interrupts no longer work.
- 5 Corrected a bug affecting the ADM-XRC-KU1, ADM-PCIE-8V3 & ADM-PCIE-8K5 where **ADMXRC3\_WriteVPD** with a block of data whose VPD space address does not start and end on a 512-byte boundary causes the data in the affected 512-byte page of VPD space memory to be shifted down by 2 bytes, cumulative with each such write.
- 6 Helper script lib[32]\_install\_helper.sh now requests to be executed under /bin/bash as opposed to /bin/sh, to avoid problems in Linux distributions where /bin/sh is not aliased to /bin/bash.
- 7 Fixed build failure with a Linux 4.6 (or later) kernel, due to breaking changes affecting the kernel functions get\_user\_pages and page\_cache\_release.
- 8 Fixed build failure with a Linux 4.9 (or later) kernel, due to breaking changes affecting the kernel functions get\_user\_pages\_remote.
- 9 Corrected sensor names and units for the ADM-PCIE-8K5, as reported by **ADMXRC3\_GetSensorInfo**, which had incorrectly been copied and pasted from definitions for the ADM-PCIE-8V3.
- 10 For the ADM-PCIE-8K5, sensor 15 now correctly reflects the FPGA temperature. Previously, it erroneously reflected the board temperature. A new sensor, with index 18, now reflects the board temperature.

### New behavior:

- 11 Device nodes in /dev are now named /dev/adb3i\* & /dev/admxrc3i\* so that the number suffix does not appear to begin with the digit 3. This fixes an issue seen in certain Linux distributions where udev does not name the /dev nodes as expected.  
  
For backwards compatibility, the new udev rules files creates symbolic links for the old names of the /dev nodes (/dev/adb3\* & /dev/admxrc3\* respectively).
- 12 **ADMXRC3\_\*DMA\*** API calls now accept a new flag, **ADMXRC3\_DMA\_BYPASSHW**. This is used by Alpha Data for testing the driver.

## Release 1.4.17

This release implements ADMXRC3 API Specification version 1.8.2.

### Enhancements:

- 1 Added support for the ADM-XRC-KU1, ADM-PCIE-8V3 and ADM-PCIE-8K5.
- 2 Added support for the secondary PCI Express Gen3 x8 endpoint of Alpha Data's ADM-PCIE-KU3-HI2 IP. This requires a system in which the x16 PCIe slot containing the ADM-PCIE-KU3 resides is bifurcated into two x8 links.
- 3 Added DWORD driver parameters **Adpexrc6tForcePcieGen** & **Adpexrc6tlForcePcieGen**, to work

around around the issue of PCIe Gen2 link speed not being automatically negotiated on a particular motherboard (Congatech TS77-i3-3217UE). Set these parameters to 1 or 2 to force retraining to PCIe Gen 1 or 2 (respectively) link speed when the driver starts. Default of 0 means no forced retraining is performed.

Currently, this workaround is performed only for the ADPE-XRC-6T (controlled by driver parameter **Adpexrc6tForcePcieGen**) or for the ADPE-XRC-6T-L (controlled by driver parameter **Adpexrc6tLForcePcieGen**).

## Release 1.4.16

This release implements ADMXRC3 API Specification version 1.7.3.

Enhancements:

- 1 Added clock programming support for the ADM-PCIE-7V3. Six reference clocks are now exposed via the ADMXRC3 API.
- 2 Added support for ADM-PCIE-7V3 sensors. Requires that the PCI revision of the ADM-PCIE-7V3 ADB3 IP is 0x03 or later for the driver to expose the sensors via the ADMXRC3 API.
- 3 Added support for the ADM-PCIE-KU3.
- 4 Added soft-reconfiguration functions using IPROG to ADB3 API, for Bridgeless models (initially ADM-PCIE-7V3 & ADM-PCIE-KU3): **ADB3\_AbortIPROG**, **ADB3\_ScheduleIPROG**, **ADB3\_StatusIPROG**
- 5 On ADM-PCIE-7V3, if PCIe revision is 0x06 or higher, enables 8 simultaneous packets in flight for increased DMA performance.

Corrections:

- 6 The **configure** script should now work correctly when building against a Linux 4.x kernel. Previously, only 2.6.x or 3.x kernel version numbers were recognized as being valid.
- 7 The driver now handles DMA mapping failures (i.e. when `dma_map_page` returns an error value), which could occur in an application which places extreme pressure on the Linux kernel's pool of bounce buffers. Now, if the driver cannot map at least one page of a chunk of a DMA transfer, the driver aborts the DMA transfer and the user-mode application that called the **ADMXRC3\_\*DMA\*** function receives a return value of **ADMXRC3\_NO\_MEMORY**.

In previous releases of the driver, DMA mapping failures were not handled, and if a DMA mapping failure occurred, the call to **ADMXRC3\_\*DMA\*** returned **ADMXRC3\_SUCCESS** but one or more sections of data were not transferred correctly.

- 8 Fixed a regression bug in ADB3 Driver 1.4.14 & 1.4.15 that affects (only) the ADPE-XRC-6T, ADPE-XRC-6T-L & ADPE-XRC-6T-ADV. In the aforementioned two driver releases, if the system monitor generates its "alert" interrupt due to an out-of-range condition on any monitored power supply or temperature sensor, high utilization (40% - 90%) of one CPU core occurs for as long as the out-of-range condition persists, due to failure by the ADB3 Driver to properly service the alert interrupt.
- 9 Changed behavior of **ADMXRC3\_GetStatusStringW** so that in the common case, where the **status** parameter represents a valid **ADMXRC3\_STATUS** value, the returned string is no longer generated by converting a **char** string to a **wchar\_t** string using a statically allocated buffer. This means that when an application calls **ADMXRC3\_GetStatusStringW** twice, the result of the first call is no longer changed by the second call.
- 10 Added a workaround for a GLIBC bug/issue (user-mode) affecting 32-bit processes running under a 64-bit Linux kernel, where **stat** can fail with **errno** set to **EOverflow** if the inode number that it should return cannot be represented by a 32-bit integer. This issue affects the **ADMXRC3\_LoadBitstreamA** & **ADMXRC3\_LoadBitstreamW** API functions, which use **stat** to determine file size. If **stat** fails, **libadmxrc3.so** now falls back to determining file size using **fseek** & **ftell**.

## Release 1.4.15

This release implements ADMXRC3 API Specification version 1.7.1.

Enhancements:

- 1 Added support for the ADM-PCIE-7V3. NOTE: Sensor functionality for the ADM-PCIE-7V3 is not included

in this version of the driver, but will be added in a future release; the number of sensors currently exposed by the ADMXRC3 API is 0.

- 2 Changed the default value of the driver parameter `PrimaryCommonBufferAlignment` to 32 (0x20), to accommodate the ADM-PCIE-7V3, whose native AXI4 / OCP internal bus width is 32 bytes.
- 3 Fixed a kernel oops (non-fatal) in debug Linux kernels when the driver is unloaded (using `rmmod`) due to `dma_free_coherent` called with smaller size to matching call to `dma_alloc_coherent`.

## Release 1.4.14

This release implements ADMXRC3 API Specification version 1.7.1.

Corrections:

- 1 On models using the LM87 system monitor chip (ADM-XRC-6TL, ADM-XRC-6T1, ADM-XRC-6TGE, ADM-XRC-6TGEL and ADM-XRC-6TDA1), fixed temperatures for sensors 8 & 9 reading as unexpectedly large positive values when the actual temperature reading (in degrees Celsius) is negative. Values for sensors 8 & 9 now read correctly as negative values when the actual temperature is negative.
- 2 The description of the sensor whose description was "Time since manufacture" has been corrected, for all applicable models except for the ADM-XRC-6T-ADV8, to read as "Total powered on time". This sensor actually reports the total time spent powered on since manufacture, but the description in previous driver versions incorrectly indicated that it was the time since manufacture.
- 3 The description of the sensor whose description was "Time since manufacture" has been corrected, for the ADM-XRC-6T-ADV8, to read as "Total 3.3VAux on time". This sensor actually reports the total time spent with 3.3VAux powered on since manufacture, but the description in previous driver versions incorrectly indicated that it was the time since manufacture.

## Release 1.4.13

Corrections:

- 1 Improved arithmetic accuracy of calculations that determine initial SI5338 output frequencies at driver startup, and fixed some anomalous frequencies that are generated when certain SI5338 output frequencies are requested via calls to **ADMXRC3\_SetClockFrequency**. Affects following models: ADM-XRC-6TGE, ADM-XRC-6TGEL, ADPE-XRC-6T & ADPE-XRC-6T-L.

## Release 1.4.12

This release implements ADMXRC3 API Specification version 1.7.0.

Enhancements:

- 1 Added support for ADM-VPX3-7V2 board.
- 2 Added support for ADM-XRC-6TGEL board.

## Release 1.4.10

This release implements ADMXRC3 API Specification version 1.6.0.

Enhancements:

- 1 Added support for message-signalled interrupts via **PciUseMsi** driver parameter (default 0), where a nonzero value causes MSI to be used. To enable MSI, load the driver with e.g. **modprobe adb3 PciUseMsi=1**.
- 2 Added new ADMXRC3 API functions: **ADMXRC3\_GetDeviceStatus** and **ADMXRC3\_ClearDeviceErrors**.
- 3 Added support for ADM-XRC-7K1 revision 2 board.
- 4 Added support for ADM-XRC-7V1 revision 2 board.
- 5 Added support for devices implemented using generic ADB3 core in target FPGA (Vendor ID = 0x4144, Device ID = 0xADB3, Subsystem Vendor ID = 0x4144, Subsystem Device ID = 0x0000).
- 6 When a generic ADB3 device (such as an ADB3 core in a target FPGA) is found, now counts number of DMA engines by examining registers in BAR0, instead of assuming 0. This means that DMA engines are

exposed by the ADMXRC3 API when available.

Corrections:

- 7 Corrected sensor names for ADM-XRC-7V1.
- 8 Fixed compile error with Linux 3.7 kernel or later, due to missing symbol **VM\_RESERVED**. Also changed memory-mapping behavior of the driver:
  - The driver now requests (to the Linux kernel) that regions mapped into a process' address space by **ADMXRC3\_MapWindow** are not core-dumped.
  - However, for common buffers (that is, real memory) mapped into a process' address space by **ADMXRC3\_MapCommonBuffer**, the driver's behaviour is unchanged (i.e. it does not request that such mapped regions are not core-dumped).
- 9 Fixed a problem with x86\_64 Gentoo Linux where the presence of the symlink /usr/lib -> lib64 confused the "configure" script, when the "-biarch" option was used, so that the 32-bit shared libraries were erroneously installed into the 64-bit library directory (/usr/lib64) instead of the directory /usr/lib32.

## Release 1.4.9

This release implements ADMXRC3 API Specification version 1.5.1.

Enhancements:

- 1 Added support for the ADM-XRC-6T-DA1 and ADM-XRC-7K1.
- 2 Added preliminary support for the ADM-XRC-7V1.

Corrections:

- 3 The implementation of **ADMXRC3\_MapWindow** and **ADMXRC3\_UnmapWindow** has been revised in order to eliminate a kernel panic that occurs when an application maps a window using **ADMXRC3\_MapWindow** and then invokes fork() to create a child process (the kernel panic occurs when the child process terminates). The new implementation is compatible with the manner in which fork() replicates the parent's virtual address space into the child process, so that **ADMXRC3\_UnmapWindow** works as expected in both the parent and child process.
- 4 On the ADPE-XRC-6T(-L), the debug message about an AVR timeout (visible in system log) has been eliminated. The driver now correctly gets the AVR uC firmware version from the AVR uC instead of timing out.
- 5 On the ADPE-XRC-6T(-L), if a board has a value of 0 (which is always invalid) in VPD for the SI5338 reference clock frequency, works around it by changing it in the in-memory copy of the data to 25000000.
- 6 On the ADM-XRC-6T-ADV, the second bank of Flash memory (dedicated to target FPGA 1) can now be accessed successfully.
- 7 Fixed an unkillable thread hang that can occur when calling **ADMXRC3\_ReadVPD** with a VPD addresses of 0x100 or above on the ADPE-XRC-6T(-L), ADM-XRC-6T-ADV8 and ADPE-XRC-6T-ADV.

## Release 1.4.6

This release implements ADMXRC3 API Specification version 1.5.0.

Enhancements:

- 1 Added preliminary support for ADPE-XRC-6T-ADV.

Corrections:

- 2 Fixed a crash resulting from a regression bug in the 1.4.5 release that results in blocking API calls sometimes failing to wait correctly.
- 3 Added support for FMCs fitted to ADPE-XRC-6T(-L). Now reports FMC information via **ADMXRC3\_GetModuleInfo** for ADPE-XRC-6T(-L) as expected when an FMC is fitted.
- 4 **ADMXRC3\_ReadSensor** now correctly reports the values of sensors whose values are negative (e.g. a temperature sensor whose reading is less than 0 deg. C) instead of a large positive value.
- 5 Corrected name of sensor 1 on ADM-XRC-6T-ADV8; was "12V supply rail", now "5V/12V XMC VPWR"



rail"; setting driver parameter "Admxrc6tadv8CompatSensor1Name" to 1 overrides this and causes the old name to be returned by the driver.

- 6 On the ADPE-XRC-6T(-L), fixed clock generator 1 being incorrectly mapped to SI5338 multisynth 0; now mapped to multisynth 1.

## Release 1.4.5

This release implements ADMXRC3 API Specification version 1.5.0.

Corrections:

- 1 Fixed a crash that can occur when **ADMXRC3\_Unlock** is called with an invalid value for the **ADMXRC3\_BUFFER\_HANDLE** parameter.
- 2 Fixed a crash when multiple queued DMA transfers are cancelled, by **ADMXRC3\_Cancel** or by killing threads, within a small window of vulnerability around to the point at which the DMA transfer would complete normally were it not cancelled.
- 3 Detection in "configure" script of which system library directories are present is now aware of several different conventions.
  - Non-biarchitecture systems which have **/usr/lib** only. This is typically the case for x86 (32-bit) Linux and 32-bit PowerPC Linux.
  - Biarchitecture systems which follow the Redhat / Fedora / CentOS style of 64-bit native libraries in **/usr/lib64** and 32-bit compatibility libraries in **/usr/lib**.
  - Biarchitecture systems which follow the Ubuntu style of 64-bit native libraries in **/usr/lib** and 32-bit compatibility libraries in **/usr/lib32**.

## Release 1.4.4

This release implements ADMXRC3 API Specification version 1.5.0.

Enhancements:

- 1 Added support for Linux 3.x.y kernels.
- 2 Added common buffer functionality with the following ADMXRC3 API functions:
  - **ADMXRC3\_GetCommonBuffer**
  - **ADMXRC3\_GetCommonBufferCount**
  - **ADMXRC3\_MapCommonBuffer**
  - **ADMXRC3\_UnmapCommonBuffer**

Corrections:

- 3 Fixed a race condition that could cause a crash every few hours of constant large DMA transfers (of size 64 physical pages or more) in a typical SMP machine.
- 4 Fixed a regression bug introduced in the 1.4.3 release that results in a small probability of **ADMXRC3\_FinishDMA** failing to return, due to a race condition.
- 5 Corrected the scaling factors for sensors 1 to 10 for the models ADPE-XRC-6T and ADPE-XRC-6T-L.

## Release 1.4.3

This release implements ADMXRC3 API Specification version 1.4.0.

Corrections:

- 6 Fixed a crash that can occur when attempting to do two or more DMA transfers on the same DMA channel.
- 7 Fixed a crash that can occur if the driver is somehow called by a libadmxrc3\*.so from a different driver version, due to the handlers for **ADMXRC3\_GetSensorInfo** and **ADMXRC3\_ReadSensor** failing to properly validate arguments.
- 8 Fixed an issue specific to the ADM-XRC-6T-ADV8 where the driver emitted the debug message "\*\*\*avrlnit: failed to get AVR uC firmware version".



- 9 Fixed a crash whose frequency depends on the rate of calling poll() or select() on an ADB3 device when doing small non-blocking DMA transfers.
- 10 Added preliminary support for the models ADPE-XRC-6T and ADPE-XRC-6T-L.

## Release 1.4.1

This release implements ADMXRC3 API Specification version 1.4.0.

New behavior:

- 1 Default permissions for /dev/admxrc3\* device nodes are now 664 (in previous releases they were 660). This brings the ADB3 Linux driver's behaviour into line with the ADB3 Windows driver.

Corrections:

- 2 Fixed a bug in the SI5338 clock synthesizer code for the ADM-XRC-6TGE that could corrupt memory when programming clock index 4. This clock generator is only available when the **Si5338ExposeAllClocks** driver parameter is nonzero; by default it is not available.
- 3 Support for ADM-XRC-6T-ADV8 is now feature-complete; added support for programming VPD and reading system monitor sensors via ADMXRC3 API.
- 4 Fixed a number of PowerPC-specific build issues, so that cross-building using Embedded Linux Development Kit is considerably simpler than before.

## Release 1.3.1

This release implements ADMXRC3 API Specification version 1.3.0.

New behavior:

- 1 Added support for the ADM-XRC-6TGE.
- 2 Added preliminary support for the ADM-XRC-6T-ADV8.

Enhancements:

- 3 Now exposes (via the ADMXRC3 API) a programmable clock generator with index 0 on the ADM-XRC-6T1 when it has firmware 1.6 (PCI revision 0x06) or later.

Corrections:

- 4 **ADMXRC3\_GetClockFrequency** now correctly returns the current clock frequency for a given clock generator. The driver now interrogates the hardware at startup to determine the current frequencies generated by each clock generator, so that **ADMXRC3\_GetClockFrequency** can return the correct frequency even before any call to **ADMXRC3\_SetClockFrequency**.
- 5 **ADMXRC3\_GetClockFrequency** now correctly validates the pointer argument (3rd argument) passed to it, and returns **ADMXRC3\_NULL\_POINTER** if it is NULL.
- 6 Corrected the maximum frequency allowed for the clock generator with index 0 on the ADM-XRC-6TL. Previously, the driver incorrectly permitted frequencies up to 210 MHz to be requested, whereas 140 MHz is the correct maximum frequency.
- 7 Fixed **ADMXRC3\_EraseFlash** failing to correctly validate the region specified to ensure that it is wholly within the unprotected region of a Flash memory bank.

## Release 1.2.1

This release implements ADMXRC3 API Specification version 1.2.0.

New behavior:

- 1 Added **udev** support. Uses the **udev** subsystem for creating device nodes in /dev/; the script **adb3.rc** is no longer required, except on Linux distributions that do not use **udev**.
- 2 For ADM-XRC-6TL, now recognizes "Extended" temperature range value (2) in VPD at offset 0x3E.
- 3 For ADM-XRC-6T1, now recognizes "Extended" temperature range value (2) in VPD at offset 0x42.

Corrections:

- 4 Fixed a race condition that occurs in (at least) Centos 5.6, between the default catch-all **udev** rule and the **adb3.rc** script. This race may result in unpredictable permissions for the **/dev/admxrc3\*** device nodes. As of release 1.2.1, **udev** is used for creating device nodes and the **adb3.rc** script is obsoleted except for Linux distributions that do not use **udev**.  
  
This issue affects any Linux distribution that has a catch-all **udev** rule which automatically creates device nodes in **/dev** for devices that do not match any other rule. CentOS 5.6 is among the Linux distributions that show this behaviour.
- 5 Fixed a regression bug in the "configure" script of the 1.2.0 release in detecting the correct directory in which to install the **libadmxrc3.so** and **libadb3.so** shared libraries. In 1.2.0, if a non-biarchitecture build and install is performed on a bi-architecture machine, the 64-bit native libraries are (incorrectly) installed in **/usr/lib/** instead of (correctly) in **/usr/lib64/**.

## Release 1.2.0

This release implements ADMXRC3 API Specification version 1.2.0.

New behavior:

- 1 Bi-architecture build is now not performed by default, as most 64-bit Linux distributions do not install (by default) the necessary compatibility packages for building 32-bit binaries. To build both 32-bit and 64-bit binaries for the API libraries, specify '-biarch yes' when running the 'configure' script.

Corrections:

- 2 Fixed some build problems relating to kernel features in RHEL / CentOS 5.5 kernels (derived from Linux 2.6.18). The 'configure' script now detects those kernel features correctly.
- 3 Fixed a thunking issue that may occur if the 32-bit and 64-bit C compilers on the same bi-architecture-capable system have different struct packing rules, where certain ADMXRC3 API calls made by a 32-bit executable running under a 64-bit Linux kernel fail with **ADMXRC3\_UNKNOWN\_ERROR**.
- 4 Fixed a problem where, when a process closes a device handle that has ongoing non-blocking operations, the process may crash due to incorrect cleanup being performed by the driver.

Enhancements:

- 5 Added new API functions for performing DMA transfers with 64-bit local addresses:
  - **ADMXRC3\_ReadDMAEx**
  - **ADMXRC3\_ReadDMALockedEx**
  - **ADMXRC3\_StartReadDMAEx**
  - **ADMXRC3\_StartReadDMALockedEx**
  - **ADMXRC3\_StartWriteDMAEx**
  - **ADMXRC3\_StartWriteDMALockedEx**
  - **ADMXRC3\_WriteDMAEx**
  - **ADMXRC3\_WriteDMALockedEx**
- 6 Added support for new sensors in ADM-XRC-6TL and ADM-XRC-6T1 with firmware 1.4 or later. This provides additional sensors that show internal temperature and voltages in the PCI Express to OCP Bridge.
- 7 Changed the way that permissions are checked on opening a device to allow for a more flexible system that can be customized via the **adb3.rc** script. See the section "[Security considerations](#)" and the comments in the file **driver/adb3.rc** for details.
- 8 Made the driver's build system more compatible with cross-compilation, now using the same ARCH and CROSS\_COMPILE environment variables as the Linux kernel. See the **README** file for details.

## Release 1.1.2

Corrections:

- 1 Fixed a user-mode memory leak that can occur when the API functions **ADMXRC3\_LoadBitstreamA** and

**ADMXRC3\_LoadBitstreamW** return a failure status code.

- 2 Fixed a kernel-mode memory leak that can occur when a device handle is closed when at least one non-blocking operation has not been finished.
- 3 Fixed a bug in **ADMXRC3\_SetClockFrequency** where on failure, the wrong status code was returned.

## Release 1.1.0

### Corrections:

- 1 Fixed a potential memory corruption issue on some architectures due to incorrect byte size calculation in device context structure.
- 2 Fixed a memory leak related to CFI Flash functionality when stopping and restarting the driver.
- 3 Fixed user VPD area (VPD space address range 0x100000-0x1FFFFFF) not being accessible on ADM-XRC-6TL and ADM-XRC-6T1.
- 4 Corrected error codes returned by several ADMXRC3 API functions when invalid parameters are passed:
  - 1 Non-Locked DMA functions now return **ADMXRC3\_INVALID\_BUFFER** if the **pBuffer** and/or **length** parameters are invalid.
  - 2 Locked DMA functions now return **ADMXRC3\_INVALID\_BUFFER\_HANDLE** if the **hBuffer** parameter is invalid.
  - 3 **ADMXRC3\_Unlock** now returns **ADMXRC3\_INVALID\_BUFFER\_HANDLE** if the **hBuffer** parameter is invalid.
- 5 Fixed **ADMXRC3\_Unconfigure** failing with **ADMXRC3\_NOT\_OWNER** even when the target FPGA has no owner.
- 6 Fixed the ADMXRC3 DMA functions not properly co-validating the **local** and **length** parameters.
- 7 Fixed the ADMXRC3 Locked DMA functions not properly co-validating the **offset** and **length** parameters.
- 8 Fixed the ADMXRC3 nonblocking DMA functions incorrectly returning **ADMXRC3\_SUCCESS** instead of **ADMXRC3\_PENDING** when **length** is zero and all other parameters are OK.
- 9 Fixed the ADMXRC3 non-Locked DMA functions not properly co-validating the **pBuffer** and **length** parameters.
- 10 Fixed certain ADMXRC3 API functions not trapping NULL pointers being passed, typically resulting in an application crash.
- 11 Fixed a bug in driver parameter handling so that driver parameters are now correctly applied and have correct default values.
- 12 Implemented VPD write-protection mechanism (now protected by default).
- 13 Added workaround for 4k crossing issue in ADB3 Bridge rev 0x01 and earlier. Workaround is not applied for rev 0x02 or later.

### Enhancements:

- 14 Added support for ADM-XRC-6T1.
- 15 Added API function **ADMXRC3\_OpenEx**, which allows an unprivileged process to open a device in "read only" mode and use a subset of the ADMXRC3 API functions.
- 16 Added API functions **ADMXRC3\_StartNotificationWait** and **ADMXRC3\_FinishNotificationWait**, which allow Linux applications to wait for events (since there is no Linux equivalent of **ADMXRC3\_RegisterWin32Event**).
- 17 Added API function **ADMXRC3\_GetCardInfoEx** (with structure **ADMXRC3\_CARD\_INFOEX**), which returns a superset of data supplied by **ADMXRC3\_GetCardInfoEx**.
- 18 Added diagnostic API functions **ADMXRC3\_GetSensorInfo** and **ADMXRC3\_ReadSensor**, with associated types and structures. These functions allow applications to monitor the health of a Gen 3 reconfigurable computing card.

## Release 1.0.0

This is the first release of the ADB3 Driver for Linux.