ADB3 Driver 1.4.16 for Wind River VxWorks Release Note

Introduction

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

ftp://ftp.alpha-data.com/pub/admxrcg3/vxworks

For support, send e-mail to support@alpha-data.com

The ADB3 Driver for Wind River VxWorks is built as a downloadable kernel module, to be downloaded to a CPU board after it has booted.

Operating systems supported

This release of the ADB3 Driver for VxWorks supports the following operating systems:

- Wind River VxWorks 5.5 and 6.x

Hardware supported

This release of the ADB3 Driver for VxWorks 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

License agreement

Please refer to the files **license.rtf** or **license.txt** within this software package for the licensing terms that apply to this software. Please contact Alpha Data if alternative licensing terms are required.

Alpha Data reserves the right to use different licensing terms for future releases of this software.

Building the driver

Prerequisites for building the driver are a Linux or Windows host machine with either Tornado 2.2 & Vxworks 5.5 or Workbench & VxWorks 6.x installed on it.

The driver is supplied in source code form so that it can be cross-built for a variety of CPU architectures and hardware platforms. To build the driver, follow the instructions in the appropriate subsection.

Cross-building the driver on a Windows host

To build the driver on a Windows host, follow these steps:

1. Unpack this package somewhere, for example

   ```
   C:\MyTesting\adb3_drv-1.4.16
   ```

   For convenience, the remainder of this document refers to this directory as %ROOT% (although it should be noted that no such environment variable is created nor referenced by the driver's build system).

2. Start a Windows command prompt that is capable of performing command-line VxWorks builds. For VxWorks 6.x, use the "VxWorks Development Shell" shortcut. For VxWorks 5.5, start a normal Windows command prompt, and then execute the **torVars.bat** batch file that can normally be found in

   ```
   C:\Tornado2.2\host\x86-win32\bin
   ```

3. In the command prompt, change directory to %ROOT% from step 1.

4. Execute MAKE with the appropriate options, as described in "MAKE options". For example, to build a debug VxBus driver for a SMP Pentium 4 system, use

   ```
   make CPU=PENTIUM4 VSB=smp clean all
   ```

   In the above command, the options **DEBUG**, **TOOLCHAIN** and **TYPE** (VxBus driver vs. legacy driver) are not specified, so they default to **true**, **gnu** and **vxbus** respectively. Assuming the build is successful, the binaries are:

   ```
   %ROOT%\driver\monolithic\vxworks\bin\vxbus_PENTIUM4gnu_debug_smp\adb3Driver.o
   ut
   %ROOT%\api\modules\admxrc3\vxworks\bin\vxbus_PENTIUM4gnu_debug_smp\admxrc3Api
   .out
   ```

At this point, you are ready to proceed to starting the driver as described in "Starting the driver".

Cross-building the driver on a Linux or UNIX host

To build the driver on a Linux or UNIX host, follow these steps:

TBA

MAKE options

Beginning with ADB3 Driver for VxWorks 1.4.10, the target VxWorks version (e.g. 5.5, 6.7 etc.) is no longer specified as a build option. Instead, this is inferred by the driver's build system from the shell environment, namely the **WIND_PLATFORM** variable. When this variable is not defined, the build system assumes VxWorks 5.5.

The top-level Makefile for the ADB3 VxWorks driver accepts a number of options which are passed on the MAKE command line. These are:

- **CPU=<*architecture*>**
  Specifies the architecture (default **PPC604**) for which the driver is to be built. In general, legal values for this can be seen by inspecting the folder **$(WIND_BASE)/target/h/tool/<*TOOLCHAIN*>**, where **TOOLCHAIN** is one of the supported toolchains (**diab**, **gnu** or **icc**). The files named **make.<CPU>< TOOLCHAIN>** represent the allowed values for **CPU** and **TOOLCHAIN**. For example, for the file

make.ARMARCH7sfdiab, **CPU=ARMARCH7** and **TOOLCHAIN=sfdiab**. See also the **TOOLCHAIN** option described below.

- **CTAG=<*configuration tag*>**
  This option, when specified, overrides the default output directory naming convention. When not specified, **CTAG** takes the value **<TYPE>_<CPU><TOOLCHAIN>_<*debug*|*release*>[_<VSB>]**.
- **CTAG_EXTRA=<*extra configuration tag*>**
  This option, when specified, is appended to **CTAG** in order to further distinguish configurations, if necessary. An underscore or other separator character is not automatically added, so the value of this option should normally begin with an underscore or hyphen.
- **DEBUG=<*false*|*true*>**
  When **true** (default), specifies a debug build in which optimizations are disabled. When **false**, specifies a release build in which normal optimizations are enabled.
- **QUIET=<*false*|*true*>**
  When **true**, the verbose display of build commands is suppressed, enabling warnings to be more easily seen during build. The default is **false**
- **SPECIAL=<*board*>**
  Specified to enable code paths for a single-board computer that requires special-case (non-generic) code in order for the driver to work correctly. Currently-supported boards are:
  - Mercury HCD5220
    Use **SPECIAL=HCD5220**
  - Motorola MVME5500
    Use **SPECIAL=MV5500**
- **TOOLCHAIN=<*gnu*|*sfgnu*|*diab*|*sfdiab*|*icc*>**
  Specifies the toolchain to be used to build the driver. This option is also used to specify whether or not the driver is built for soft-floating point (**sfgnu** or **sfdiab**).
- **TYPE=<*legacy*|*vxbus*>**
  Specifies whether the driver should be built as a legacy driver or a VxBus driver (default). If the build system detects a VxWorks 5.5 environment, **TYPE** is automatically set to **legacy**.
- **VSB=<*variant*>**
  Specifies VxWorks variant libraries, if required. If omitted, the normal libraries are used. This option **must** be specified when building the driver for a SMP system, or for a 64-bit system where the architecture also supports a 32-bit mode. Commonly-used values are **smp**, **lp64_smp** and **lp64**.

  Although VxWorks Source Builds (VSB) was a feature introduced in VxWorks 6.7, the driver makes a special case exception for VxWorks 6.6, which lacks VSB but permits SMP. In the case of VxWorks 6.6, the **VSB=smp** is permitted. This causes the driver's build system to pass -D_VX_

Starting the driver

To start the driver in the target system, follow these steps:

1. Download the modules **adb3Driver.out** and **admxrc3Api.out** to the target system. This can be done using the **ld** command in the VxWorks shell or the target system's console. For example:

   ```
   -> ld <hostname:C:/MyTesting/adb3_drv-1.4.16/driver/monolithic/vxworks/bin/
   legacy_PPC604gnu_debug/adb3Driver.out
   -> ld <hostname:C:/MyTesting/adb3_drv-1.4.16/api/modules/admxrc3/vxworks/bin/
   legacy_PPC604gnu_debug/admxrc3Api.out
   ```

2. To start the driver, use the entry point **adb3DrvStart**:

   ```
   -> adb3DrvStart
   ```

   This entry point accepts two parameters:

   - **debugLevel** (int), default 0
     Verboseness of debug output sent to console using **logMsg**. The release version of a driver produces no output. In the debug version of the driver, a value of 0 results in minimal output and

increasing values (up to 10) result in more output.

- **bLegacyHardware** (int), default 0
  Nonzero to enable support for legacy hardware such as the ADM-XRC-II.

For example, to start the driver with some extra debug output and support for legacy hardware, use:

```
-> adb3DrvStart(2,1)
```

A **debugLevel** value greater than zero may greatly slow down execution of the driver, so 0 is recommended during normal usage.

Starting the driver with a **debugLevel** of 0 should result in output of the following form on the console:

```
-> adb3DrvStart(0,1)
0xf68b790 (tShell): adb3: dfDriverEntry: ADB3 Monolithic Driver, version=1.4.16.26
0xf68b790 (tShell): adb3: identifyPci9656: identified ADM-XRC-II
```

VPD write-protection mechanism

The VPD write-protection mechanism described in the ADM-XRC Gen 3 SDK User Guide is implemented as of release 1.1.1. To enable VPD writes, the global integer variable adb3DrvEnableVpdWrite must be set to 1 either programatically or using the VxWorks shell. This value is checked every time an application attempts to write to the VPD memory, so changes to this variable take effect immediately rather than being sampled when the driver is started.

To set this value to 1 (thus enabling VPD writes) using the VxWorks shell, use:

```
-> adb3DrvEnableVpdWrite=(int)1
```

To set this value to 0 (thus disabling VPD writes) using the VxWorks shell, use:

```
-> adb3DrvEnableVpdWrite=(int)0
```

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 **adb3DrvPrimaryCommonBufferCount** determines the number of common buffers allocated; the default is zero, meaning that no common buffers are allocated by default. The parameter **adb3DrvPrimaryCommonBufferSizeLow** determines the size in bytes of each common buffer; the default is 64 kiB (0x10000). The parameter **adb3DrvPrimaryCommonBufferAlignment** determines the address boundary size to which each common buffer is guaranteed to be aligned; the default is 32 bytes (0x20).

Known issues

Modifications to certain BSPs needed for interrupt delivery

Certain BSPs for single-board computers require a modification to an interrupt vector table in order for interrupts to be delivered to the ADB3 Driver. If the driver behaves as if interrupts are not being delivered to it - for example, if DMA transfers hang, or the "ITest" example from the ADM-XRC Gen 3 SDK fails to work correctly - it may necessary to modify the file **hwConf.c** in your VxWorks kernel image project.

Although Alpha Data cannot in general provide precise instructions for doing this, for many BSPs the necessary steps are as follows:

1. Open **hwConf.c** in your VxWorks kernel image project in your favorite editor, and locate a table of this form:

```
LOCAL const struct intrCtlrInputs loApicInputs[] = {
    { VXB_INTR_DYNAMIC, "yn", 0, 0 },
    { VXB_INTR_DYNAMIC, "gei", 0, 0 },
    ... other entries ...
#if defined (INCLUDE_HPET_MSI)
    { INT_NUM_IA_HPET_TIMER0, "iaHpetTimerDev", 0, 0 },
    ... other entries ...
#endif /* INCLUDE_HPET_MSI */
    ... other entries ...
};
```

2. If you have a single Alpha Data Gen 3 Reconfigurable Computing Device in the target system, add the following entry to the end of the table:

```
    { VXB_INTR_DYNAMIC, "adb3", 0, 0 }
```

   If you have more than one device, add as many entries as you have devices, incrementing the number in the 3rd position of each entry for each entry. For example, if there are 4 devices, add the following entries to the end of the table:

```
    { VXB_INTR_DYNAMIC, "adb3", 0, 0 },
    { VXB_INTR_DYNAMIC, "adb3", 1, 0 },
    { VXB_INTR_DYNAMIC, "adb3", 2, 0 },
    { VXB_INTR_DYNAMIC, "adb3", 3, 0 }
```

3. Rebuild your VxWorks kernel image. It should now be capable of delivering interrupts to the ADB3 Driver.

vxbDmaBufLib symbols missing when downloading the ADB3 Driver

If, when you download **adb3Driver.out** to the target processor, you see messages of the form

```
Warning: module 0xffff80000023b1a0 holds reference to undefined symbol vxbDmaBufMe
mAlloc.
Warning: module 0xffff80000023b1a0 holds reference to undefined symbol vxbDmaBufMa
pCreate.
Warning: module 0xffff80000023b1a0 holds reference to undefined symbol vxbDmaBufTa
gCreate.
... other warnings ...
```

it indicates one of two things:

- The **INCLUDE_DMA_SYS** component is excluded from the kernel image. In that case, add it to the kernel image and rebuild. The ADB3 Driver requires this component in order to be able to perform DMA transfers.
- The **INCLUDE_DMA_SYS** component is included in the kernel image but is being optimized out due to no kernel component using it. In that case, the workaround is to include a kernel component that is known to rely on **INCLUDE_DMA_SYS**, such as **INCLUDE_EHCI**. Then, rebuild the kernel image.

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.

Compiler warnings during builds

The following compiler warnings may be generated when building the driver for certain configurations:

- *../../../modules/admxrc3/admxrc3.c(858) (col. 45): warning #13365: XMM register was modified, but CG_allow_xmm was not specified*
  This warning arises because the 64-bit build specs for the Intel Compiler for VxWorks set the **-fno-implicit-fp-sse** option (no implicit floating point or SSE). This warning should appear only when building the ADMXRC3 API library, and can be safely ignored.
- *../../framework/vxworks/vxbus_pci.c: In function 'getRawBusResources':*
  ../../framework/vxworks/vxbus_pci.c:137: warning: unused variable 'f'
  This warning arises from variables that are required in some VxWorks configurations and not in others, and can safely be ignored.
- *icc: command line warning #10006: ignoring unknown option '-Wbad-function-cast'*
  icc: command line warning #10006: ignoring unknown option '-Wno-sign-conversion'
  This warning arises because the 64-bit build specs for the Intel Compiler for VxWorks set a couple of compiler options that are not supported for that particular version of the compiler. These warnings can safely be ignored.
- *"../../framework/vxworks/legacy_methods.c", line 154: warning (dcc:1500): function pciIntDisconnect2 has no prototype*
  This warning can be ignored, and occurs because the **pciIntDisconnect2** function appears to be missing from the VxWorks 5.5 header files, although it exists and can be used in some VxWorks kernel images.

Release history

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. 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.

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 adb3DrvPrimaryCommonBufferAlignment to 32 (0x20), to accommodate the ADM-PCIE-7V3, whose native AXI4 / OCP internal bus width is 32 bytes.

Release 1.4.14

This release implements ADMXRC3 API Specification version 1.7.1.

Corrections:

1. Fixed a window of vulnerability that could result in a crash when the ADB3 Driver's interrupt service routine (ISR) executed at the exact instant that the Driver polled for deassertion of the system monitor ALERT interrupt (in order to detect the end of an out-of-range condition on supply rails, temperature sensors etc.). In previous versions of the ADB3 Driver, each time the driver polled, there existed a small window of vulnerability which could have resulted in a crash.
2. 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.
3. 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.
4. 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.

New behavior:

3. When building as a VxBus driver for VxWorks 6.9 or later, now uses the VXBUS DMA subsystem for setting up and tearing down DMA transfers, instead of legacy methods. This requires the component "vxBus Driver DMA System" (**INCLUDE_DMA_SYS**) to be included in the VxWorks kernel.

Corrections:

4 Fixed a regression introduced in 1.4.10 driver where debug messages on the console are not displayed correctly in VxWorks versions before 6.9.

Release 1.4.11

This release implements ADMXRC3 API Specification version 1.6.0.

Corrections:

1 Fixed a crash in function **_dfVxwTimerStub**, when starting the driver on certain 64-bit VxWorks platforms.

Release 1.4.10

This release implements ADMXRC3 API Specification version 1.6.0.

Enhancements:

1 Added new ADMXRC3 API functions: ADMXRC3_GetDeviceStatus and ADMXRC3_ClearDeviceErrors.

2 Added support for ADM-XRC-7K1 revision 2 board.

3 Added support for ADM-XRC-7V1 revision 2 board.

4 Introduced a new build system that makes use of the makefiles provided by VxWorks, which eliminates the need to create a rules file for less common configurations.

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 Various code fixes to support building for VxWorks 6.9 and 64-bit VxWorks, and to minimize the number of compiler warnings over a variety of kernel configurations.

8 Corrected sensor names for ADM-XRC-7V1.

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 On the ADM-XRC-6T-ADV, the second bank of Flash memory (dedicated to target FPGA 1) can now be accessed successfully.

4 On the ADPE-XRC-6T(-L), the debug message about an AVR timeout (visible on console) 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 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 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.

3 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.

4 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.

5 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.

Enhancements:

1 Added common buffer functionality with the following ADMXRC3 API functions:

- ADMXRC3_GetCommonBuffer
- ADMXRC3_GetCommonBufferCount
- ADMXRC3_MapCommonBuffer
- ADMXRC3_UnmapCommonBuffer

2 Added support for the models ADPE-XRC-6T and ADPE-XRC-6T-L.

Corrections:

3 Fixed a crash that can occur when ADMXRC3_Unlock is called with an invalid value for the ADMXRC3_BUFFER_HANDLE parameter.

4 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.

5 Fixed a race condition that could cause a crash every few hours of constant large DMA transfers in a typical SMP machine.

6 Corrected the scaling factors for sensors 1 to 10 for the models ADPE-XRC-6T and ADPE-XRC-6T-L.

7 Fixed a crash that can occur when attempting to do two or more DMA transfers on the same DMA channel.

8 Fixed a crash that can occur if the driver is somehow called by an ADMXRC3 API library from a different driver version, due to the handlers for ADMXRC3_GetSensorInfo and ADMXRC3_ReadSensor failing to properly validate arguments.

9 Fixed an issue specific to the ADM-XRC-6T-ADV8 where the driver emitted the debug message "*** avrInit: failed to get AVR uC firmware version".

Release 1.4.1

This release implements ADMXRC3 API Specification version 1.4.0.

Corrections:

1 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.

2 Support for ADM-XRC-6T-ADV8 is now feature-complete; added support for programming VPD and

reading system monitor sensors via ADMXRC3 API.

Release 1.4.0

This release implements ADMXRC3 API Specification version 1.4.0.

Enhancements:

1. Added new API functions for performing DMA transfer to arbitrary PCI-E addresses: ADMXRC3_ReadDMABus, ADMXRC3_StartReadDMABus, ADMXRC3_StartWriteDMABus, ADMXRC3_WriteDMABus.
2. Added caching mechanism for Flash memory; reduces delays in execution of Flash API functions when performing many small write and/or erase operations.

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.
3. For the ADM-XRC-6TL, now recognizes "Extended" temperature range value (2) in VPD at offset 0x3E.
4. For the ADM-XRC-6T1, now recognizes "Extended" temperature range value (2) in VPD at offset 0x42.

Enhancements:

5. 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:

6. 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.
7. ADMXRC3_GetClockFrequency now correctly validates the pointer argument (3rd argument) passed to it, and returns ADMXRC3_NULL_POINTER if it is NULL.
8. 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.
9. 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 1.2.0

This release implements ADMXRC3 API Specification version 1.2.0.

New behavior:

1. For ADM-XRC-6TL and ADM-XRC-6T1, now recognizes "Extended" temperature range value (2) in VPD at offset 0x42.

Corrections:

2. Fixed a problem where, when a task closes a device handle that has ongoing non-blocking operations, the task may crash due to incorrect cleanup being performed by the driver.

Enhancements:

3. 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

4 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.

Release 1.1.2

Enhancements:

1 Improved the build system so that the GNU toolchain can be used for building the driver as well as the DIAB toolchain.

2 Added a rules file for building a VxBus driver for PowerPC PPC85XX and soft-floating point.

Corrections:

3 Fixed a bug where DMA transfers on the ADM-XRC-II (legacy model) do not work when the transfer is large enough to make the linked-list that describes the DMA transfer larger than a single element (may need to be greater than 16 MiB to trigger this condition).

4 Fixed a data corruption issue that can occur for the start and ending cache lines of a DMA transfer when the following conditions are met:

- The platform does not implement hardware cache coherency for DMA transfers.
- The DMA transfer does not start and end on a cache line boundary.
- The DMA transfer direction is from the target FPGA to CPU memory.

5 The driver no longer uses the VxBus functions htole8/16/32/64 for endian-conversion when the driver is built as a VxBus driver because the htole64 function is not implemented by VxWorks.

Release 1.1.1

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

Page Intentionally left blank

Address: 4 West Silvermills Lane,
Edinburgh, EH3 5BD, UK
Telephone: +44 131 558 2600
Fax: +44 131 558 2700
email: sales@alpha-data.com
website: http://www.alpha-data.com

Address: 3507 Ringsby Court Suite 105,
Denver, CO 80216
Telephone: (303) 954 8768
Fax: (866) 820 9956 toll free
email: sales@alpha-data.com
website: http://www.alpha-data.com