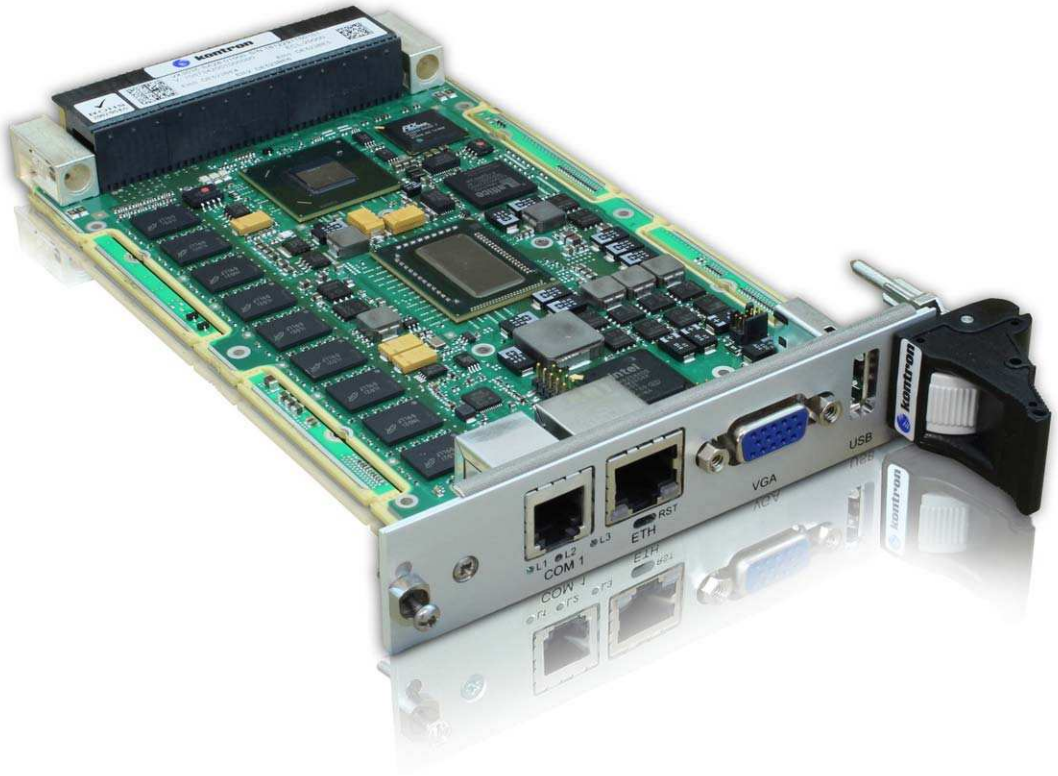
» VX3035 «



Release Notes VxWorks 6.9 on VX3035

SD.DT.G24-0e - April 2013

Revision History

Publication Title:	VxWorks 6.9 on VX3035	
Doc. ID:	SD.DT.G24-0e	
Rev.	Brief Description of Changes	Date of Issue
0e	Initial Issue	04-2013

Proprietary Note

This document contains information proprietary to Kontron. It may not be copied or transmitted by any means, disclosed to others, or stored in any retrieval system or media without the prior written consent of Kontron or one of its authorized agents.

The information contained in this document is, to the best of our knowledge, entirely correct. However, Kontron cannot accept liability for any inaccuracies or the consequences thereof, or for any liability arising from the use or application of any circuit, product, or example shown in this document.

Kontron reserves the right to change, modify, or improve this document or the product described herein, as seen fit by Kontron without further notice.

Trademarks

This document may include names, company logos and trademarks, which are registered trademarks and, therefore, proprietary to their respective owners.

Environmental Protection Statement

This product has been manufactured to satisfy environmental protection requirements where possible. Many of the components used (structural parts, printed circuit boards, connectors, batteries, etc.) are capable of being recycled.

Final disposition of this product after its service life must be accomplished in accordance with applicable country, state, or local laws or regulations.

Environmental protection is a high priority with Kontron.

Kontron follows the DEEE/WEEE directive.

You are encouraged to return our products for proper disposal.

The Waste Electrical and Electronic Equipment (WEEE) Directive aims to:

- reduce waste arising from electrical and electronic equipment (EEE)
- make producers of EEE responsible for the environmental impact of their products, especially when they become waste
- encourage separate collection and subsequent treatment, reuse, recovery, recycling and sound environmental disposal of EEE
- improve the environmental performance of all those involved during the lifecycle of EEE

Conventions

This guide uses several types of notice: Note, Caution, ESD.

Note: this notice calls attention to important features or instructions.


Caution: this notice alert you to system damage, loss of data, or risk of personal injury.


ESD: This banner indicates an Electrostatic Sensitive Device.

All numbers are expressed in decimal, except addresses and memory or register data, which are expressed in hexadecimal. The prefix `0x' shows a hexadecimal number, following the `C' programming language convention.

The multipliers `k', `M' and `G' have their conventional scientific and engineering meanings of $*10^3$, $*10^6$ and $*10^9$ respectively. The only exception to this is in the description of the size of memory areas, when `K', `M' and `G' mean $*2^{10}$, $*2^{20}$ and $*2^{30}$ respectively.

When describing transfer rates, `k' `M' and `G' mean $*10^3$, $*10^6$ and $*10^9$ *not $*2^{10}$ $*2^{20}$ and $*2^{30}$.*

In PowerPC terminology, multiple bit fields are numbered from 0 to n, where 0 is the MSB and n is the LSB. PCI and CompactPCI terminology follows the more familiar convention that bit 0 is the LSB and n is the MSB.

Signal names ending with an asterisk (*) or a hash (#) denote active low signals; all other signals are active high.

Signal names follow the PICMG 2.0 R3.0 CompactPCI Specification and the PCI Local Bus 2.3 Specification.

For Your Safety

Your new Kontron product was developed and tested carefully to provide all features necessary to ensure its compliance with electrical safety requirements. It was also designed for a long fault-free life. However, the life expectancy of your product can be drastically reduced by improper treatment during unpacking and installation. Therefore, in the interest of your own safety and of the correct operation of your new Kontron product, you are requested to conform with the following guidelines.

High Voltage Safety Instructions


Warning!
All operations on this device must be carried out by sufficiently skilled personnel only.


Caution, Electric Shock!
Before installing a not hot-swappable Kontron product into a system always ensure that your mains power is switched off. This applies also to the installation of piggybacks. Serious electrical shock hazards can exist during all installation, repair and maintenance operations with this product. Therefore, always unplug the power cable and any other cables which provide external voltages before performing work.

Special Handling and Unpacking Instructions

ESD Sensitive Device!
Electronic boards and their components are sensitive to static electricity. Therefore, care must be taken during all handling operations and inspections of this product, in order to ensure product integrity at all times

Do not handle this product out of its protective enclosure while it is not used for operational purposes unless it is otherwise protected.

Whenever possible, unpack or pack this product only at EOS/ESD safe work stations. Where a safe work station is not guaranteed, it is important for the user to be electrically discharged before touching the product with his/her hands or tools. This is most easily done by touching a metal part of your system housing.

It is particularly important to observe standard anti-static precautions when changing piggybacks, ROM devices, jumper settings etc. If the product contains batteries for RTC or memory backup, ensure that the board is not placed on conductive surfaces, including anti-static plastics or sponges. They can cause short circuits and damage the batteries or conductive circuits on the board.

General Instructions on Usage

In order to maintain Kontron's product warranty, this product must not be altered or modified in any way. Changes or modifications to the device, which are not explicitly approved by Kontron and described in this manual or received from Kontron's Technical Support as a special handling instruction, will void your warranty.

This device should only be installed in or connected to systems that fulfill all necessary technical and specific environmental requirements. This applies also to the operational temperature range of the specific board version, which must not be exceeded. If batteries are present, their temperature restrictions must be taken into account.

In performing all necessary installation and application operations, please follow only the instructions supplied by the present manual.

Keep all the original packaging material for future storage or warranty shipments. If it is necessary to store or ship the board, please re-pack it as nearly as possible in the manner in which it was delivered.

Special care is necessary when handling or unpacking the product. Please consult the special handling and unpacking instruction on the previous page of this manual.

Table of Contents

Table of Contents

1 INTRODUCTION VX3035

This document is the VX3035 BSP User's Guide. It gives the installation procedure for the VX3035 vxWorks 6.9 BSP complemented with the vxWorks 6.9/Workbench 3.3 patch level required to support this VX3035 BSP.

This file also contains information on the board BIOS configuration, BSP general information, used device drivers and finally limitations and BSP evolutions compared to the previous versions.

The document contains the following first level chapters:

INTRODUCTION
 - VxWorks REQUIREMENTS
 - BIOS REQUIREMENT
 - JUMPERS & SWITCHS
BSP INSTALLATION GUIDE
BOOTING VXWORKS ON VX3035
HARDWARE FEATURES SUPPORTED BY VX3035 BSP
KONTRON DEVICES DRIVERS
BSP INFORMATION
KNOWN PROBLEMS
BSP EVOLUTIONS
BIBLIOGRAPHY
SEE ALSO

1.1 VxWorks Requirements

To support VX3035 BSP, VxWorks 6.9 and Workbench 3.3 must be first updated to

VxWorks 6.9 updated to 6.9.2.2 (Service Pack 2)
Workbench 3.3 updated to 3.3.3 (Update Pack Pack 3)

1.2 BIOS Requirements

The BIOS ID of the VX3035 must be greater or equal to **13010**. The BIOS ID can be verified into the first page of BIOS SETUP.

First of all **restore the default values**. To proceed, enter in the BIOS by pressing **F2** or DEL during the boot process of the VX3035. Then Press
* F3 -> Optimized Defaults

After this is done and the Optimized default BIOS parameters are set then do the following:
* Advanced -> CPU Configuration -> **Hyper-Threading [Disabled]**

If your board is not system controller on VPX backplane it is stronly advised to disable the VPX switch PEX8609. Indeed the VPX switch reserves a big amount of PCI Memory and cause vxWorks 32bit kernel to have less than 1Gbytes of RAM available. To disable the PEX VPX switch on P1 backplane modify SETUP:

* Kontron -> VPX Configuration -> **VPX switch [Disabled]**

If no SATA device is used, it is advised to disable the SATA controller. This will allow to save 5s for bootrom boot time and an additional 5s for kernel boot time (Another way is to rebuild the vxWorks bootrom and kernel without SATA AHCI support). To disable SATA controller into SETUP BIOS do:

* Chipset -> CougarPoint (PCH) Configuration -> SATA Configuration -> **SATA Controller(s) [Disabled]**

Once the setting is done save the parameters for the next boot :

* Save & exit -> **Save changes and Reset**

For PXE boot: Check the PCH LAN Controller is enabled

* Chipset -> CougarPoint (PCH) Configuration -> **PCH LAN Controller: [Enable]**

Enable the PXE OpROM

* Chipset -> CougarPoint (PCH) Configuration -> **PXE ROM : [Enable]**
* Save & Exit -> **Save Changes and Reset**

Then re-enter in the BIOS by pressing **F2** or DEL during the boot process of the VX3035. Enable PXE on the wished interface and disable it on other interfaces. **IBA GE Slot 00C8** is the single 1Gbe 82579 ethernet interface (front side if available) while other **IBA GE Slot** are the Dual 82580 ETH interface on backplane P2 connector.

* Boot -> Network Device BBS Priorities -> **IBA GE Slot XXXX** -> Enabled or Disable
* Boot -> Network Device BBS Priorities -> Boot Option #1: [IBA GE Slot ...]
* F4: Save Changes and Exit
* Go to BIOS SETUP again and set boot device order (PXE: Boot -> Boot Option #1: [IBA GE Slot ...])

1.3 Jumpers & Switches

Before using a board with VxWorks, verify that the board runs in the factory configuration by using vendor-supplied ROMs and jumper settings and checking the RS-232 connection. Refer to 'CA.DT.A95 VX3035 User's Guide' for a detailed description of the board.

2 BSP INSTALLATION GUIDE

2.1 Install vxWorks 6.9 / Workbench 3.3

Get and install the DVDROM of the vxWorks 6.9 release.

DVD-R147826.1-1-01 : VxWorks-6.9 and VxWorks Edition 6.9 Platform

Remark: In the "Installer" Window, when asked to "Choose on line update setting",
Remove selection for the 2 following items:
- check for and apply latest updates for this installer
- check online for the latest Wind River product updates

2.2 Install VxWorks 6.9.2.2

DVD-R147826.1-11-00 : VxWorks 6.9.2.2

Uncompress the archive file delivered by WindRiver that corresponds to **DVD-R147826.1-11-00**. (see the WINDRIVER support web site : support.windriver.com, do not download or unzip it in install/updates directory !).

Go to the **DVD-R147826.1-11-00** folder and launch the appropriate shell script for the host machine.

For Linux :

```
./setup_linux
```

For Windows launch :

```
setup.exe
```

For Solaris :

```
./setup_solaris
```

Install the update pack 2 release

Install the vxWorks 6.9.2.2 release The Sevice Pack 2 vxWorks 6.9 from WindRiver is present in this DVDROM in the following folder :

servicePack2

2.3 Install Workbench Update Pack 3

DVD-R158451.1-1-16b : Wokbench 3.3 Update Pack 3

Uncompress the archive file delivered by WindRiver that corresponds to DVD-R158451.1-1-16b. (see the WINDRIVER support web site : support.windriver.com, do not download or unzip it in install/updates

directory !)

Go to the **DVD-R158451.1-1-16b** folder and launch the appropriate shell script for the host machine.

For Linux :

```
./setup_linux
```

For Windows' launch :

```
setup.exe
```

For Solaris :

```
./setup_solaris
```

Install the Workbench 3.3 update pack 3 release.

The Update Pack 3 Workbench 3.3 from WindRiver is present in the DVDROM in the folder named **Workbench_update**.

2.4 Install the VX3035 BSP

The BSP of the VX3035 is located in **VXW69-BSP-VX3035-xxxxx** folder. Where xxxxx identify the BSP ID.

Files present at the top of the VXW69-BSP-VX3035-xxxxx folder :

bootrom.bin: bootrom to put in a SATA or USB disk (Full feature USB/SATA/GEI/TEI)
bootrom.pxe: bootrom that is able to be launched through PXE protocol (GEI boot device)
vximages : Folder containing the VxWorks images for UP/SMP mode compiled with GNU and DIAB compiler
VXW69-BSP : contains the vxWorks BSP files and docs
vx3035.pdf : Describes the BSP release for vxWorks 6.9 for vx3035
servicePack2: contains the update pack 2 for vxWorks 6.9 from WindRiver
Workbench_update: contains the update pack 3 for Workbench 3.3 from WindRiver
VXW69-BSP-VX3035-xxxxxx.zip: BSP zip file ID xxxxxx to be used for installation

First **set the environment variables** by typing under the WindRiver Installation folder:

Linux OS :

```
./wrenv.linux -p vxworks-6.9
```

Windows OS :

```
wrenv.exe -p vxworks-6.9
```

Go to the DVDROM directory.

Unzip the file **VXW69-BSP-VX3035-xxxxx.zip** in the home vxWorks 6.9 directory as follows :

under **linux OS** :

```
cd $WIND_HOME
unzip /mnt/cdrom/VXW69-BSP-VX3035-xxxxx.zip -d .
```

under **Windows OS** (if **WIND_HOME**=C:\WindRiver and D: is the DVDROM) :

unzip the D:\VXW69-BSP-VX3035-xxxxx.zip in C:\WindRiver

During the unzipping process, in case the following message appears :

```
./vxworks-6.9/target/h/make/rules.bsp? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
```

Type A to replace existing files.

2.5 Recompile libs

Once the patches are installed, the platform needs to be regenerated for the target.

Rename/Recompile an existing file :

-- Under **Linux OS** --

```
mv $WIND_HOME/vxworks-6.9/target/h/hwif/vxbus/vxbMethodDecl.h
 $WIND_HOME/vxworks-6.9/target/h/hwif/vxbus/vxbMethodDecl.h.orig


cd $WIND_HOME/vxworks-6.9/target/src/hwif/methods

make vxbMethodDecl.h
```

The following commands must be performed after each BSP installation to add all new drivers registration:

```
mv $WIND_HOME/vxworks-6.9/target/config/all/vxbUsrCmdLine.c
$WIND_HOME/vxworks-6.9/target/config/all/vxbUsrCmdLine.c.orig

cd $WIND_HOME/vxworks-6.9/target/config/comps/src/hwif

make vxbUsrCmdLine.c
```

-- Under **Windows OS** --

```
mv %WIND_HOME%/vxworks-6.9/target/h/hwif/vxbus/vxbMethodDecl.h
%WIND_HOME%/vxworks-6.9/target/h/hwif/vxbus/vxbMethodDecl.h.orig

cd %WIND_HOME%/vxworks-6.9/target/src/hwif/methods

make vxbMethodDecl.h
```

The following commands must be performed after each BSP installation to add all new drivers registration:

```
mv %WIND_HOME%/vxworks-6.9/target/config/all/vxbUsrCmdLine.c
```

```
%WIND_HOME%/vxworks-6.9/target/config/all/vxbUsrCmdLine.c.orig

cd %WIND_HOME%/vxworks-6.9/target/config/comps/src/hwif

make vxbUsrCmdLine.c
```

Go to :

-- Under **Linux OS** --

```
cd $WIND_HOME/vxworks-6.9/target/src

make CPU=NEHALEM TOOL=gnu rclean && \
make CPU=NEHALEM TOOL=diab rclean && \
make CPU=NEHALEM TOOL=icc rclean && \
make CPU=NEHALEM TOOL=gnu VXBUILD=SMP rclean && \
make CPU=NEHALEM TOOL=diab VXBUILD=SMP rclean && \
make CPU=NEHALEM TOOL=icc VXBUILD=SMP rclean && \
make CPU=NEHALEM TOOL=gnu && \
make CPU=NEHALEM TOOL=diab && \
make CPU=NEHALEM TOOL=icc && \
make CPU=NEHALEM TOOL=gnu VXBUILD=SMP && \
make CPU=NEHALEM TOOL=diab VXBUILD=SMP && \
make CPU=NEHALEM TOOL=icc VXBUILD=SMP
```

-- Under **Windows OS** --

```
cd %WIND_HOME%/vxworks-6.9/target/src

make CPU=NEHALEM TOOL=gnu rclean
make CPU=NEHALEM TOOL=diab rclean
make CPU=NEHALEM TOOL=icc rclean
make CPU=NEHALEM TOOL=gnu VXBUILD=SMP rclean
make CPU=NEHALEM TOOL=diab VXBUILD=SMP rclean
make CPU=NEHALEM TOOL=icc VXBUILD=SMP rclean
make CPU=NEHALEM TOOL=gnu
make CPU=NEHALEM TOOL=diab
make CPU=NEHALEM TOOL=icc
make CPU=NEHALEM TOOL=gnu VXBUILD=SMP
make CPU=NEHALEM TOOL=diab VXBUILD=SMP
make CPU=NEHALEM TOOL=icc VXBUILD=SMP
```

2.6 Compile VX3035 kernel

The installation procedure is ended. The VX3035 kernel can then be compiled by command or the Workbench Project can be launched to configure and compile a new image kernel.

To compile by command, make sure the environment variables are initialized:

Linux OS :

```
./wrenv.linux -p vxworks-6.9
```

Windows OS :

```
wrenv.exe -p vxworks-6.9
```

Then under **Linux OS** or **Windows OS**

```
cd vxworks-6.9/target/config/vx3035
make vxWorks
```

or for standalone kernel (advised)

```
make vxWorks.st
```

or if the SMP (Symmetric Multiprocessing Support) feature is expected

```
cd vxworks-6.9/target/proj ; vxprj create -force -smp vx3035 gnu vx3035_gnu_smp_bspbuild
cd vxworks-6.9/target/proj/vx3035_gnu_smp_bspbuild ; vxprj build
```

Warning: make from the BSP directory **(make vxWorks VXBUILD=SMP) is not supported** with vxWorks 6.9, please move to using the vxprj command line build.

3 BOOTING VXWORKS ON VX3035

When the board is powered on, the system BIOS code loads and executes the bootstrap loader. The bootstrap loader is written in 8088 16-bit assembly language. The BIOS obtains the bootstrap loader from the boot sector, which may be in one of several locations: an USB key, a hard disk, or some other alternatives. When the BIOS finds the bootstrap loader, it transfers execution to it. The bootstrap loader finds the bootrom.sys file, loads it into memory, and transfers execution to romInit.

3.1 Booting VxWorks via PXE

It is possible to boot VxWorks via the network using PXE (the Preboot eXecution Environment). PXE support is provided by the BIOS. The advantage of using PXE is that no floppy or hard disk is required. This is an important point, indeed, support for floppy disk drives is becoming increasingly rare with newer PC systems.

A bootrom.pxe loader is provided for PXE booting VxWorks in addition to the regular bootrom.bin loader. The bootrom.pxe loader is virtually identical to the normal bootrom.bin, except that it has an extra block of padding prepended to it. When booting from disk, the vxld.bin loader is normally loaded by the BIOS at the address 0x7C00. It then in turn loads the bootrom at address 0x8000 and starts it running. By contrast, when booting with PXE, the bootrom is loaded directly at address 0x7C00 by the PXE ROM. A leading block of 1024 NOP instructions is therefore used to force the **romInit( )** entry point to fall at the address 0x8000, just as it would do if it had been loaded from a disk. Once loaded and running, bootrom.pxe is able to load VxWorks images via the network just as the normal bootrom.bin does.

PXE booting requires the use of a DHCP and TFTP server. When the PXE ROM starts up, it will use DHCP to discover an IP address and the path for the bootrom file. Generally, the PXE ROM assumes that the DHCP and TFTP servers will be running on the same machine. The way the DHCP and TFTP servers are configured will depend on what software is used. DHCP and TFTP supports are available for most *NIX/Linux/*BSD systems, as well as for Microsoft Windows (R). The following sample of setup instructions explains how to configure the open source ISC DHCP server running on a FreeBSD system.

Assume that your client machine is on the 10.0.0.0/32 network, and that your DHCP/TFTP server's IP address is 10.0.0.1. When the PXE client is started up, the PXE ROM should display the station address of the ethernet port. We will assume the address is 00:00:E8:01:02:03, and that we want to assign the target an IP address of 10.0.0.3. Given this, we need to create a dhcpd.conf file as follows:

```
# dhcpd.conf
#
# Sample PXE boot configuration file for ISC dhcpd
#

ddns-update-style none;

subnet 10.0.0.0 netmask 255.255.255.0 {
  range 10.0.0.3 10.0.0.3;
  deny unknown-clients;
  host vxWorks_pentium_target {
    hardware ethernet 00:00:e8:01:02:03;
  }
  option routers 10.0.0.1;
  option broadcast-address 10.0.0.255;
```

```
  filename "/bootrom.pxe";
  default-lease-time 600;
  max-lease-time 7200;
}
```

Start the DHCP server with the following command, as superuser:

```
# dhcpd -cf /path/to/dhcpd.conf
```

Next, edit the /etc/inetd.conf file to enable the TFTP service, and insure inetd is started. The enable inetd, edit the /etc/rc.conf file and add a line that says **inetd_enable="YES"** then run the following commands:

```
# sh /etc/rc.d/inetd stop
# sh /etc/rc.d/inetd start
```

Finally, create a /tftpboot directory, and place a copy of the bootrom.pxe loader here.

Check out the BIOS requirement -> PXE Boot in this document to setup the board in PXE mode.

Once this setup has been done, reboot the PXE client system. It should display an output similar to the following:

```
CLIENT MAC ADDR: 00 00 E8 01 02 03  GUID: xxxxxxxx xxxx xxxx xxxx xxxxxxxxxxxx
CLIENT IP: 10.0.0.3  MASK: 255.255.255.0  DHCP IP: 10.0.0.1
GATEWAY IP: 10.0.0.1
TFTP...
```

Right after this output appears, the VxWorks bootrom should start up and the bootrom startup banner should be displayed. At this point, you can enter the desired VxWorks boot parameters and use the bootrom to load a VxWorks image.

Important notes:

- PXE is a 16-bit real mode environment. Consequently, boot applications downloaded via PXE are limited in that they must be able to start up in 16-bit real mode, and they must be less than 640K in size. The VxWorks bootrom.bin image meets these requirements, however standalone vxWorks images might not.

- The PXE environment also provides a universal API for accessing the underlying ethernet device, however VxWorks has no support for this feature. This means that while you can download the VxWorks bootrom via any PXE-enabled ethernet device, a native driver for the ethernet device must be available in VxWorks in order for the bootrom to subsequently download a VxWorks image (or for the downloaded VxWorks image to be able to use the ethernet port).

- The bootrom will be able to save the bootline parameters into VX3035 onboard EEPROM

3.2 Make bootrom for bootable disk (USB, SATA)

This section explains how to build a bootrom image including USB and/or SATA boot features.

If you want to boot vxWorks image from **SATA disk**, enable **INCLUDE_DRV_STORAGE_INTEL_AHCI** in **config.h**, then change **DEFAULT_BOOT_LINE** to **DEFAULT_BOOT_LINE_SATA**. Under linux command line execute.

```
'make clean'
'make bootrom.bin BOOTAPP=y'
```

If you want to boot vxWorks image from **USB** key, the bootrom must be build with the command line (for example under Linux).
Open a VxWorks Development Shell, in the BSP directory "$(**WIND_BASE**)/target/config/vx3035", run the command:

```
'make clean'
'make bootrom.bin BOOTAPP=y BOOTROM_FULL_FEATURE=y'
```

BOOTROM_FULL_FEATURE is mandatory to support USB into the bootrom and so support USB boot. It is not mandatory if you don't want to support USB into the bootrom (this boots faster).

BOOTAPP flag is important for the bootrom. Do not forget it!

The bootrom cannot be generated under the **Workbench tool**

copy the bootrom image ("bootrom.bin") to the disk (renamed bootrom.sys).
copy vxWorks image to disk if you want to load image from it.
Follow the method indicated hereafter to format the boot device (USB, SATA).

Remark: If you are not using SATA interface it is strongly advised to undefine **INCLUDE_DRV_STORAGE_INTEL_AHCI** into bootrom and kernel or to disable SATA controller into BIOS SETUP. This will allows to save 10 s for boot time !

3.3 Install and boot with MKBOOTAHCI vxWorks command on bootable SATA disk

1) Boot vxWorks kernel including mkbootAhci

mkbootAhci command is available in the vxWorks BSP of VX3035. This command can be used to install a bootloader on SATA disk directly from VxWorks.
To do this, you must first build a bootrom correctly configured (see above) and the corresponding standalone kernel (make vxWorks.st). You must first boot this kernel by the way you prefer (PXE bootrom or USB bootrom). Once this kernel is booted then, you should have access to the target SATA disk device.

2) Format the disk and install boot loader under vxWorks Shell command

Execute the following command to prepare the bootable disk. For example, with a SATA

device on SATA port 4 and 4 partitions created. See also **createPart( )** into **partition.c** for a code example on how to prepare a boot on SATA disk.

```
-> xbdCreatePartition "/ahci04:1", 4, 33, 33, 33        (Command must be adapted if SATA po
        Instantiating /ahci00 as rawFs, device = 0x1
        Instantiating /ahci00:1 as rawFs,  device = 0xc0001
        Instantiating /ahci00:2 as rawFs,  device = 0xd0001
        Instantiating /ahci00:3 as rawFs,  device = 0xe0001
        Instantiating /ahci00:4 as rawFs,  device = 0xf0001
        value = 0 = 0x0

-> dosFsVolFormat "/ahci04:1",0x02,0                            (Command must be adapted i
        Formatting /ahci04:1 for DOSFS
   " ...
        OK.
        value = 0 = 0x0

-> dosFsVolFormat "/ahci04:2",0,0                                       (Command must be ac
        Formatting /ahci04:2 for DOSFS
        " ...
        OK.
        value = 0 = 0x0

-> dosFsVolFormat "/ahci04:3",0,0                                       (Command must be ac
        Formatting /ahci00:3 for DOSFS
     " ...
        OK.
        value = 0 = 0x0

-> dosFsVolFormat "/ahci04:4",0,0                                       (Command must be ac
     " ...
        OK.
        value = 0 = 0x0

-> copy "bootrom.bin","/ahci04:1/bootrom.sys"
        value = 0 = 0x0

-> mkMbrAhci(0,4)                                                                       (Co
        interrupt: SATA Drive inserted on ctrl 0, port 4
        value = 0 = 0x0


-> mkbootAhci 0,4,"bootrom.bin"                                         (Command must be ac
        num bytes written = 500728
        value = 0 = 0x0

-> copy "vxWorks.st","/ahci04:1/vxWorks.st"             (Command must be adapted if SATA po
        value = 0 = 0x0

Then reboot and Select SATA device from the BIOS Setup (Enter F2 to Setup)

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++
```

```
                        VxWorks System Boot
Copyright 1984-2008  Wind River Systems, Inc.

 CPU: SYMMETRIC IO SMP VX3035.  Processor #0.
 Memory Size: 0x1fec0000 (~510Mb).  BSP version 1.0/13045.
 Created: Jan 11 2013, 16:39:13
 ED&R Policy Mode: Deployed
```

```
 WDB Comm Type: WDB_COMM_END
 WDB: Ready.


        Press any key to stop auto-boot...
        1

        [VxWorks Boot]: c
'.' = clear field;  '-' = go to previous field;  ^D = quit

        boot device          : gei0 ata=0,0
        processor number     : 1
        host name            : pcsstr2
        file name            : /tmp/vxWorks.st /ahci00:1/vxWorks.st
        inet on ethernet (e) : 172.20.161.25:0xffffff00
        inet on backplane (b):
        host inet (h)        : 172.20.161.184
        gateway inet (g)     :
        user (u)             : vxworks
        ftp password (pw) (blank = use rsh): vxworks
        flags (f)            : 0x8
        target name (tn)     : vx3035
        startup script (s)   :
        other (o)            : gei0

        [VxWorks Boot]: @
        boot device          : ata=0,0
        unit number          : 0
        processor number     : 1
        host name            : pcsstr2
        file name            : /ahci00:1/vxWorks.st
        inet on ethernet (e) : 172.20.161.25:0xffffff00
        host inet (h)        : 172.20.161.184
        user (u)             : vxworks
        ftp password (pw)    : vxworks
        flags (f)            : 0x8
        target name (tn)     : vx3035
        other (o)            : gei0

        Loading /ahci00:1/vxWorks.st...1725728 + 383248

        Starting at 0x408000...
        Attached IPv4 interface to gei unit 0
        Adding 6233 symbols for standalone.

         ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
         ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]       VxWorks 6.9 SMP
         ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]       KERNEL: WIND version 2.13
         ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]       Copyright Wind River Systems, Inc., 1984-2012

         CPU: SYMMETRIC IO SMP VX3035.  Processor #0.
         Memory Size: 0x7a19a000 (~1953Mb).  BSP version 2.0/xxxxx_UD.
         Created: Mar 16 2012, 15:19:54
         ED&R Policy Mode: Deployed
         WDB Comm Type: WDB_COMM_END
         WDB: Agent Disabled.

        -> devs
                drv name
                0 /null
                1 /tyCo/0
                1 /tyCo/1
```

```
        7 pcsstr2:
        10 /vio
        3 /ahci00:1
        3 /ahci00:2
        3 /ahci00:3
        3 /ahci00:4
value = 0 = 0x0
-> ll "/ahci00:1"

Listing Directory /ahci00:1:

-rwxrwxrwx  1 0       0               430432 Jan  1  1980 bootrom.sys
-rwxrwxrwx  1 0       0              2276470 Jan  1  1980 vxWorks.st
```

3.4 Install and boot with MKBOOTUSB vxWorks command on bootable USB disk

1) Boot vxWorks kernel including mkMbrUsb and mkbootUsb

mkMbrUsb and mkbootUsb commands are available in the vxWorks BSP of VX3035 if **INCLUDE_USB** is defined. These commands can be used to install a bootloader on USB disk directly from VxWorks.
To do this, you must first build a bootrom correctly configured (see above, do not forget **BOOTROM_FULL_FEATURE=**y) and the corresponding standalone kernel vxWorks. You must first boot this kernel by the way you prefer (PXE bootrom or SATA bootrom). Once this kernel is booted then, you should have access to the target USB disk device.

2) Format the disk and install boot loader under vxWorks Shell command

Execute the following command to prepare the bootable disk
See also **createPartUSB( )** into **partition.c** for an example on how to prepare a boot on USB disk.

```
-> xbdCreatePartition ("/bd0", 2, 94, 0, 0)
Instantiating /bd0 as rawFs, device = 0x40001
Instantiating /bd0 as rawFs,  device = 0x90001
Instantiating /bd0:2 as rawFs,  device = 0xa0001
value = 0 = 0x0
-> dosFsVolFormat ("/bd0",0x12,0)
Formatting /bd0 for DOSFS
Instantiating /bd0 as rawFs, device = 0x90001
Formatting...Retrieved old volume params with %38 confidence:
Volume Parameters: FAT type: FAT32, sectors per cluster 0
  0 FAT copies, 0 clusters, 0 sectors per FAT
  Sectors reserved 0, hidden 0, FAT sectors 0
  Root dir entries 0, sysId (null)  , serial number e870000
  Label:"           " ...
Disk with 912681 sectors of 512 bytes will be formatted with:
Volume Parameters: FAT type: FAT16, sectors per cluster 16
2 FAT copies, 57012 clusters, 223 sectors per FAT
Sectors reserved 1, hidden 63, FAT sectors 446
Root dir entries 512, sysId VXDOS16 , serial number e870000
Label:"           " ...
OK.
value = 0 = 0x0
-> copy ("bootrom.bin","/bd0/bootrom.sys")
value = 0 = 0x0
-> mkMbrUsb("/bd0")
```

```
        Instantiating /bd0 as rawFs, device = 0x40001
        value = 0 = 0x0
        -> Instantiating /bd0:2 as rawFs,  device = 0xd0001
        mkbootUsb ("/bd0","bootrom.bin")
        Instantiating /bd0 as rawFs, device = 0x40001
        Instantiating /bd0:2 as rawFs,  device = 0x100001
        num bytes written = 576352
        value = 0 = 0x0
        -> copy ("vxWorks","/bd0/vxWorks")
        value = 0 = 0x0


        Then reboot and Select USB device (do not use UEFI !) from the BIOS Setup ([F2] to Enter)

        +++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        +++++++++++++++++++++++++++++++++

                            VxWorks System Boot
Copyright 1984-2008  Wind River Systems, Inc.

 CPU: SYMMETRIC IO SMP VX3035.  Processor #0.
 Memory Size: 0x1fec0000 (~510Mb).  BSP version 1.0/13045.
 Created: Jan 11 2013, 16:39:13
 ED&R Policy Mode: Deployed
 WDB Comm Type: WDB_COMM_END
 WDB: Ready.


        Press any key to stop auto-boot...
        1

        [VxWorks Boot]: c
'.' = clear field;  '-' = go to previous field;  ^D = quit

        boot device          : gei0 fs
        processor number     : 0
        host name            : pcsstr2
        file name            : /tmp/vxWorks.st /bd0/vxWorks
        inet on ethernet (e) : 172.20.161.25:0xffffff00
        inet on backplane (b):
        host inet (h)        : 172.20.161.184
        gateway inet (g)     :
        user (u)             : vxworks
        ftp password (pw) (blank = use rsh): vxworks
        flags (f)            : 0x8
        target name (tn)     : vx3035
        startup script (s)   :
        other (o)            : gei0

        [VxWorks Boot]: @
        boot device          : fs
        unit number          : 0
        processor number     : 0
        host name            : pcsstr2
        file name            : /bd0/vxWorks
        inet on ethernet (e) : 172.20.161.25:0xffffff00
        host inet (h)        : 172.20.161.184
        user (u)             : vxworks
        ftp password (pw)    : vxworks
        flags (f)            : 0x8
        target name (tn)     : vx3035
        other (o)            : gei0
```

```
2908640 + 700916
Starting at 0x408000...


Adding 9221 symbols for standalone.

 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]       VxWorks 6.9 SMP
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]      KERNEL: WIND version 2.13
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]     Copyright Wind River Systems, Inc., 1984-2013

 CPU: SYMMETRIC IO SMP/SMT VX3035.  Processor #0.
Memory Size: 0x1fec0000 (~510Mb).  BSP version 1.0/13045.
Created: Jan 30 2013, 11:10:40
ED&R Policy Mode: Deployed
WDB Comm Type: WDB_COMM_END
WDB: Ready.


-> devs
drv name
  0 /null
  1 /tyCo/0
  1 /tyCo/1
  3 /ahci00:1
  3 /ahci00:2
  9 pcsstr2:
 10 /vio
 11 /tgtsvr
  3 /bd0
  4 /bd0:2
value = 25 = 0x19
->
-> ll "/bd0"

Listing Directory /bd0:
-rwxrwxrwx  1 0       0               576352 Jan  1  1980 bootrom.sys
-rwxrwxrwx  1 0       0              3911408 Jan  1  1980 vxWorks
value = 0 = 0x0
->
```

4 HARDWARE FEATURES SUPPORTED BY VX3035 BSP

4.1 List of the hardware features supported by this BSP

This section gives the hardware features list supported onto the board.

Hardware Interface	Driver/Component	Status	Tested
DDR3 (size of 2GB/4GB)	-	SUPPORTED	Yes
UART:0	**vxbNs16550Sio.c**	SUPPORTED	Yes (RS232 only)
UART:1	**vxbNs16550Sio.c**	SUPPORTED	Yes (RS232 only)
1GB-ETHERNET:0	**gei825xxVxbEnd.c**	SUPPORTED	Yes
DUAL-1GB ETH:1,2	**gei825xxVxbEnd.c**		
RTC (date&time access)	vxbMc146818RtcK .c	SUPPORTED	Yes
RTC (used as timer)	**vxbMc146818Rtc.c**	SUPPORTED	Yes
SMBUS RTC (Set date&time)	**vxSMBusRtc.c**	SUPPORTED	Yes
USB2-HOST	USB EHCI driver	SUPPORTED	Yes
USB1-HOST	USB UHCI driver	SUPPORTED	No
SMBus	**vxbPchSMBus.o**	SUPPORTED	Yes
CPLD Watchdog	VX CPLD	SUPPORTED	Yes
EEPROM	VXI2CEEPROM	SUPPORTED	Yes
FRAM	VXI2CEEPROM	SUPPORTED	No
CPLD LEDS	VX LEDS	SUPPORTED	Yes
CPLD GPIO	VX GPIO	SUPPORTED	Yes
Temp, Voltage Sensor	VX NCT7802Y	SUPPORTED	Yes
Temperature LM73	VX LM73	SUPPORTED	Yes
Voltage Sensor ADS7830	VX ADS7830	SUPPORTED	Yes
AUDIO	n/a	UNSUPPORTED	No
I2C Backplane num 0	VX CPLD	SUPPORTED	Yes
I2C Backplane num 1	VX CPLD	SUPPORTED	Yes
Boot on HDD SATA	SATA AHCI	SUPPORTED	Yes
Boot on SATA Flash module	SATA AHCI	SUPPORTED	Yes
Boot on USB Mass Storage	USB EHCI	SUPPORTED	Yes
Boot on USB Flash module	USB EHCI	SUPPORTED	Yes (Initialize USB disk under vxWorks)
SMB Alert	VX SMBALERT	SUPPORTED	Yes

4.2 Serial Configuration

SIO should be enabled by the BIOS baseboard. Default configuration is COM0:
Baud Rate: 115200
Data: 8 bit
Parity: None
Stop: 1 bit
Flow Control: None

4.3 Network Configuration

This BSP provides support for VxBus drivers, including network drivers. The list of VxBus network drivers available for PCIe devices includes the following:

```
INCLUDE_GEI825XX_VXB_END
INCLUDE_END
```

The following END is supported by this BSP:

Driver	END	END2
GEI	**INCLUDE_GEI825XX_VXB_END**	UNSUPPORTED

5 KONTRON DEVICES DRIVERS

Following is a short description of the device drivers developped by Kontron and specific to VX3035 like GPIO, EEPROM, Watchdog, etc.

5.1 Manage 3rd party drivers

VxBus Drivers developed by Kontron are located in the following directory :

```
$WIND_BASE/target/3rdparty/kontron_vx3035
```

It includes the following vxBus drivers

```
'vxNCT7802Y'  - Voltage and temperature sensors driver (Nuvoton Hardware Monitor)
'ads7830'     - Voltage sensors driver
'vxLm73'      - LM73 1 temperature sensors driver (towards I2C CPLD)
'rtc146818'   - PCH RTC driver for fast date and time access
'vxSMBusRtc'  - Driver for RTC on SMBus (External RTC with battery)
'vxCpld'      - CPLD access driver
'vxI2cEeprom' - EEPROM driver (2 EEPROM and 1 FMRAM towards I2C CPLD)
'vxSMBALERT'  - Driver for Alert & interrupt coming from backplane SMB (GDISCRETE Signal)
'pchsmbus'    - SMBUS driver (for SMBus backplane access)
'vxGPIO'      - GPIO driver (towards CPLD)
'vxLED'       - LED driver  (towards CPLD)
'addons'      - SATA driver for Greenliant SATA Flash Protection (for mezzanine SSD)
```

and a directory named **includes** for *.h include files

For details on any of these drivers please refer to their own documentation located in $**WIND_HOME**/docs/extensions/eclipse/plugins/com.windriver.ide.doc.bsp_vxworks_6.9/vx3035

```
vxbMC146818RtcK.html
vxGPIO.html
vxSMBusRtc.html
vxbPchSMBus.html
vxLED.html
vxbVxCpldDrv.html
vxNCT7802Y.html
vxbVxI2cEepromDrv.html
vxSMBALERT.html
vxbAds7830.html
vxbLm73.html
```

These drivers are accessible by the user in the VX3035 BSP through the two library files :

* **libkontron.a** (UP mode)
* **libkontron_smp.a** (SMP mode)

These drivers do not implement standard I/O control entries but use a WindRiver Method called driverControl. This method allows the User API to access directly to the user driver routine with a structure of function pointers. For example the driver NCT7802Y implements this feature. Note that other drivers implement the same feature. This is the declaration of the method in the driver :

```
LOCAL device_method_t vxNCT7802YDrv_methods [] =
```

```
    {
    DEVMETHOD(driverControl, vxNCT7802YFuncGet),
    { 0, 0}
    };
```

The routine vxNCT7802YFuncGet will return to the user application a pointer to the following structure. A control structure can be retrieved in the same way in each .h driver file.

```
struct vxNCT7802YControl
    {
    /* backward reference to the device instance */
    VXB_DEVICE_ID pInst;

    /* read AD channel */
    STATUS (*chanRead)
        (
        VXB_DEVICE_ID pInst,    /* [in] device instance */
        UINT8 chan,             /* [in] channel # to read */
        UINT16 *pVal            /* [out] data */
        );

    STATUS (*voltGet)
        (
                VXB_DEVICE_ID pInst,    /* [in] device instance */
                UINT8 chan,             /* [in] channel # to read */
                UINT8 **ppName,         /* [out] voltage name */
                UINT32 *pVal            /* [out] voltage value (in mV) */
        );

    /* Get current temperature in degrees */
    STATUS (*tempGet)
        (
        VXB_DEVICE_ID pInst,    /* [in] device instance */
                UINT8 chan,                             /* [in] channel # to read */
                UINT8 **ppName,                 /* [out] temperature name */
        signed short *pTemp                     /* [out] temperature */
        );

    /* Read NCT7802Y register */
    STATUS (*regRead)
        (
        VXB_DEVICE_ID pInst,    /* [in] device instance */
        UINT8 reg,              /* [in] register index */
        UINT8 *pVal             /* [out] register value */
        );

    /* Write to NCT7802Y register*/
    STATUS (*regSet)
        (
        VXB_DEVICE_ID pInst,    /* [in] device instance */
        UINT8 reg,              /* [in] register index */
        UINT8 val               /* [in] new register value */
        );
     };
```

To point to the correct vxBus Driver and retrieve the driverControl Method, we recommend to use the following routine:

```
void *vxbGetCtrlInfo(char *instName, int unit)
```

This routine is present in the VX3035 BSP in the **bspUtil.c** file.

To summarize how the user can use the driverControl Method the vx3035 BSP contains example like the following one and allow the user to get the temperature of the board. This code is present in the BSP in the file name **tempGet.c** :

```
STATUS tempGet(char *name, int drvNum, int sensNum, signed short *temp)
{
        STATUS status;
        struct vxNCT7802YControl *drvCtrl;
        UINT8 *sens_name;

        drvCtrl = vxbGetCtrlInfo(name,drvNum);

        if(drvCtrl == NULL || drvCtrl->pInst == NULL)
        {
                return -1;
        }

        status = drvCtrl->tempGet(drvCtrl->pInst, sensNum, &sens_name, temp);

        return status;
}
```

5.2 RTC Use

An external RTC RV8564 accessible by SMBus is used to manage the current date. This external RTC is there for power battery saving compared to Internal PANTHERPOINT PCH RTC. A vxBus driver has been adapted to use this external RTC and to allow to set and get the current time. The name of this vxBus driver is "vxSMBusRtc". The standard "vxbRtcGet" and "vxbRtcSet" vxbus driver entries can be used. Define **DRV_VXSMBUS_RTC** must be set to include this driver.

See the **examples** directory to find an example code for smbusRtcGet and smbusRtcSet routines

```
-> smbusRtcGet
Year:13, Month:1, Day:15, WDay:2
Hour:11, Min:20, Sec:13
value = 0 = 0x0
->
```

The PCH COUGARPOINT RTC is a duplicate of the External RTC. It can be used for fast RTC date and time access. To use this Kontron PCH RTC vxBus driver "mc146818RtcKontron" for date and time access in project then define the following include:

```
DRV_RTC_MC146818
```

See **vxbMC146818RtcK** driver documentation for some use code example

5.3 CPLD Watchdog

CPLD Watchdog is available through CPLD register access. It uses vxCpld vxBus driver method. Watchdog can be used in interrupt, timer or reset mode. An example on how to use vxCpld and watchdog is given in the "examples" directory in the file **cpldWdt.c**.

```
-> cpldWdt
0: Getting functionality : OK
1: Getting CPLD parameters : OK
Cpld version : 0x5 (1)
Cpld geo id : 0x1d (1)
2: Checking CPLD parameters : OK
3: Activate watchdog (10 sec)  : OK
4: Trigger watchdog in 9 sec  : OK
5: Disable the watchdog in 10 sec : OK
6: Reset the controller (controller must be reset in 5 sec)  :
TOTAL TESTS = 6, FAILED = 0

value = 0 = 0x0
->

Then board Reset
```

5.4 Board Information (VPD)

Board information or Vital Product Data (VPD) like Serial Number can be accessed from VPD EEPROM. If necessary, the user can include and link application located in $**WIND_BASE**/target/3rdparty/kontron_vx3035/vxI2cEeprom/ with 3 files

```
vx6060_vpd.c
vxvariant.c
vxvpdtool.c
```

The VPD can be read with **vpddisplay( )**. This routine is included into the EEPROM vxBus **vxI2cEeprom** folder but it is not loaded by default into the kernel. To include the command **vpddisplay( )** you can invoke it somewhere into the code. For example into usrAppInit (defining **INCLUDE_USER_APPL**)

```
extern STATUS vpddisplay(int dispflags);
void usrAppInit (void)
    {
        vpddisplay(0);
#ifdef  USER_APPL_INIT
        USER_APPL_INIT;
#endif
```

Then compiling will cause the **vpddisplay** command to be included into the kernel :

```
-> vpddisplay(0);
Board type   : VX3035-SA28-01000
EC Level     : 20000
Serial Number: 1812291150099
Variant      : 0187302001100000
Keylist      : /PCB_A/SACLASS/MEZZUSBSATA/BHQUAD/BHDUALMODE/SNBLV2G2/2Gbit_DDR3_1067/DDR_2B/DDR3-1
IRTC/NOXDP/STDCLK/STD_EARTH/BATON/DPBDP/PWRMAGOFF/FP422OFF/TMLVDSOFF/I2CSTD/1INCHES/1SLOT/P80OFF/VS
IBOMOFF/ITIN/LFPROCESS/LFCOMP/

value = 0 = 0x0
-> vpddisplay(1)
0187302001100000
value = 0 = 0x0
```

```
-> vpddisplay(2)
20000
value = 0 = 0x0
->
```

5.5 EEPROMs/FMRAM Accesses

Two EEPROMs and one FMRAM are available on Kontron Core i7 VX3035 boards.

One EEPROM is VPD reserved (Vital Product Data) and the second is User reserved. The User reserved EEPROM can thus be used entirely by an application except the first 0x2000 bytes reserved for bootrom parameters storage and are not accessible to the User by the vxBus method (So usable EEPROM offset can start at 0 as seen by user). The FMRAM is totally free to be used. All these devices are available through the vxBus vxI2CEeprom driver.

An example on how to access FMRAM is given with file **vxFramTool.c** with **target/3rdparty/kontron_vx3035/vxI2cEeprom/vxFramTool.c** routines:

```
STATUS vxprintfmram(int offset, int size)
STATUS vxwritefmram(int offset, int size, UINT8 buffer[])
STATUS vxreadfmram(int offset, int size,UINT8 *buffer)
```

These routines are part of the vxBus library but not linked to the kernel by default. The user needs to invoke it somewhere into the code to include them.

This file example is easy to adapt for **EEPROM**. Instead of calling eeprom->fmramRead or write method user can call any other appropriate method like ->userRead or ->userWrite to have access to the **User EEPROM** etc. See vxI2cEeprom driver documentation itself for a complete vxBus method.

Example of use, dump FMRAM from offset 0 size 40 bytes

```
-> vxprintfmram(0,40)
Raw FMRAM data:
0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0 0x 0
0x 0 0x 0 0x 0 0x 0 0x20 0x20 0x20 0x38 0x34 0x36 0x39 0x33 0x30 0x38 0x38 0x36
0x20 0x20 0x20 0x20 0x20 0x20 0x31 0x36
value = 0 = 0x0
->
```

5.6 Voltage & Temperature Sensors

There is a NCT7802Y voltage and temperature sensors available on the VX3035 (with I2C CPLD). In the "examples" directory, **vxNCT7802Ytest.c** gives an example on how to get and test the voltage and temperature.

```
-> vxNCT7802YTest
0: Getting driver functionality : OK
1: Reading registers : OK
2: Getting voltage values :
Channel 0: VCC(from 3V3) 3268
Channel 1: VCORE 852
Channel 2: VSENS1(1V5 DDR3) 1514
Channel 3: VSENS2(from +12V) 12232
```

```
Channel 4: VSENS3(from 5V) 5056 OK
3: Checking voltage ranges : OK

TOTAL TESTS = 4, FAILED = 0

4: Getting temperature :
Temperature 0: LTD temperature 31.250000 (0)
OK
5: Checking temperature : OK
6: Reading registers :
0 Reg : 0x50 (0) fd
1 Reg : 0xc3 (0) fe
OK
7: Setting upper limit register :
upper-limit Reg : 0x55 (0)OK
8: Checking upper limit register :
upper-limit Reg : 0x56 (0)OK
9: Setting low limit register :
Low-limit Reg : 0xd8 (0) OK
10: Checking low limit register :
Low-limit Reg : 0xd7 (0)OK

TOTAL TESTS = 11, FAILED = 0

value = 0 = 0x0
->
```

DRV_VXNCT7802Y must be defined set to include NCT7802Y driver.

Also the NCT7802Y temperature can be read simply with the command **tempGetShow** included into the BSP in **tempGet.c** file. This command also read the LM73 temperature sensors (if **DRV_LM73** is defined) located behind the CPLD i2C bus.

tempGetShow must be invoked with (0,0) as parameter (driver number,sensor number)

```
STATUS tempGetShow(int num, int sensor);


-> tempGetShow
Temp NCT7802 = 26
Temp LM73    = 26.75
value = 0 = 0x0
->
```

CPU(s) temperature can be read with BSP routine **Read_Thermal_Status_CPU( )**.

ADS7830 Voltage sensors are available behind the Board SMbus. In the "examples" directory, **voltage.c** gives an example on how to get and test the voltage.

```
-> ads7830Test
0: Getting driver functionality : OK
1: Reading registers : OK
2: Getting voltage values :
Channel 0: 12V VPX 11892
Channel 1: 5V VPX 4886
Channel 2: 3V3 3300
```

```
Channel 3: 2V5 6U 2498
Channel 4: 3V3 VPX SB 3222
Channel 5: 1V05S 3U 1025
Channel 6: 1V05 3U PCH 1044
Channel 7: 1V 3U 976 OK
3: Checking voltage ranges :
Channel 0: 12V VPX 11892 [min=11500,max=12500]
Channel 1: 5V VPX 4886 [min=4500,max=5500]
Channel 2: 3V3 3300 [min=3000,max=3500]
Channel 3: 2V5 6U 2498 [min=2300,max=2800]
Channel 4: 3V3 VPX SB 3222 [min=3000,max=3500]
Channel 5: 1V05S 3U 1025 [min=1000,max=1100]
Channel 6: 1V05 3U PCH 1044 [min=1000,max=1100]
Channel 7: 1V 3U 986 [min=950,max=1050] OK

TOTAL TESTS = 4, failed = 0

value = 0 = 0x0
->

DRV_ADS7830 must be defined to include Ads7830 driver.
```

5.7 CPLD GPIO

There is 5 GPIOs available on VX3035. The second GPIO (GPIO1) is reserved for Reseting the board when its direction is set to INPUT so it should only be used as OUTPUT. GPIO are managed with vxBus driver vxGPIO. See **vxbGPIO.h** file to find all GPIO mode that are programmable. See **vxbGPIO.c** for all vxBus method associated with GPIO

GPIO programmable vxBus methods and modes can be :

- GPIO getDirection/setDirection (**GPIO_DIRECTION_IN**, **GPIO_DIRECTION_OUT**)

- GPIO getValue/setValue (**GPIO_VALUE_LO**, **GPIO_VALUE_HI**)

- GPIO getIntMode/setIntMode: level sensitive or edge (**GPIO_INTMODE_LEVEL**, **GPIO_INTMODE_EDGE**)

- GPIO getIntEnabled/setIntEnabled

- GPIO getIntPolarity/setIntPolarity (**GPIO_POLARITY_LO**, **GPIO_POLARITY_HI**)

- GPIO getIntToggle/setIntToggle (**GPIO_TOGGLE_OFF**,**GPIO_TOGGLE_ON**)

When a GPIO is selected in Toggle Mode GPIO, the interrupt Mode and polarity are not taken into consideration. If enabled, then an interruption will be generated if a GPIO has a state that changes.

A test example is given in the **examples** directory file **gpioTest.c** and basic GPIO access service are provided into file **gpioCpld.c** under BSP directory. The test **gpioTest** can work only if the GPIOs are connected two by two. It demonstrates how to program them.

DRV_GPIO_VX must be defined for GPIO to include GPIO driver.

5.8 CPLD LEDs

Two front side LEDs can be controlled by the vxLED vxBus driver. Leds colors and blink modes can be set. See **vxLED.h** for available modes. See **ledTest.c** file in example directory for a LEDs test demonstration.

DRV_LED_VX must be defined to include vxLED driver.

5.9 I2C Backplane

Two SMbus (100Kz) are available on VX3035 P0 VPX backplane. SMbus 0 is always accessible while SMbus 1 is only accessible for a Board System Controller. A program using the CPLD I2C driver is given into BSP file **i2cBpAccess.c**. This program is adapted for the backplane EEPROM device. The Routine **i2cBpAccessScan( )** provides a scan example of SMbus backplane while other routines demonstrate how to read/write into a device (this is adapted for EEPROM device with a page size of 32). In particular **i2cBpAccessScan( )** allows to detect all VPX30xx boards into the backplane including itself. This routine can thus be used to discover a VPX architecture with other VX30xx boards.

```
-> ld < i2cBpAccess.o
value = 136237232 = 0x81ed0b0
-> i2cBpAccessScan

---------- I2C SCAN BUS EXT 0 ------------
8bits accesses devs found  : 0x32
16bits accesses devs found : 0x32
---------- I2C SCAN BUS EXT 1 ------------
8bits accesses devs found  :
16bits accesses devs found :
value = 0 = 0x0
-> basicTestEEPROM(0,0xA0)
** EEPROM 0xa0 on SMB 0 ***
Comparing data
value = 0 = 0x0
->
```

5.10 Get PBIT Result

If available, PBIT (POwer-On Built In Tests) results are read from EEPROM (if PBIT have been ordered with the BIOS). They can be read with the command **pbitdisplay( )** available in the BSP. See file **vxpost.c** in 3rdParty and also **pbitResultShow( )** into **pbitResult.c** file.

```
-> pbitdisplay
POSTs configured to run from command line:
       mem_data: PASSED
       mem_addr: PASSED
   mem_pattern1: PASSED
   mem_pattern2: PASSED
   mem_pattern3: PASSED
   mem_pattern4: PASSED
    pcie_vpx_sw: PASSED
         serial: PASSED
            rtc: FAILED
       sysflash: PASSED
           cpld: PASSED
```

```
   temp_sensors: PASSED
    temperature: PASSED
         fnvram: PASSED
    ether_loop0: PASSED
    ether_loop1: PASSED
    ether_loop2: PASSED
    ether_loop3: PASSED
    ether_loop4: PASSED
        voltage: PASSED
          hwmon: PASSED
sata0_controler: PASSED
sata1_controler: PASSED
            vpd: PASSED
         eeprom: PASSED
usb1_controller: PASSED
usb2_controller: PASSED
         system: PASSED (FAILED ONCE)

PASSED  : 27
FAILED  : 1
NOT RUN : 0
TOTAL   : 28

value = 0 = 0x0
->
```

6 BSP INFORMATION

6.1 Identify BSP Release ID

To identify the version of this VX3035 BSP see **KRELEASE** file into vx3035 BSP directory. Also check the ID that is displayed with BSP kernel is starting

```
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]       VxWorks 6.9 SMP
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]       KERNEL: WIND version 2.13
 ]]]]]]]]]]]]]]]]]]]]]]]]]]]]      Copyright Wind River Systems, Inc., 1984-2013

 CPU: SYMMETRIC IO SMP VX3035.  Processor #0.
 Memory Size: 0x1fec0000 (~510Mb).  BSP version 1.0/13045.
 Created: Jan 15 2013, 17:27:17
```

The BSP command **verModulesShow** gives a complete board information with BSP ID, BIOS ID board Serial Number, Engineering change number, CPLD version ...

```
-> verModulesShow

---------- List of Module Versions ----------

VxWorks                 : 6.9 SMP
BSP ID                  : 1.0/13099
BSP Creation            : Apr 11 2013, 18:26:49

BIOS Vendor             : American Megatrends Inc.
BIOS ID                 : ID13010
BIOS Date               : 01/10/2013

BOARD Vendor            : Kontron
BOARD Version           : 20000_0187302001100000
BOARD S/N               : 1812291150099

BOARD CPLD VER.         : 0x7

IPP Lib Version         : 7.0 (7.0.46)

value = 0 = 0x0
->
```

6.2 SMP mode

The BSP supports SMP (Symmetric Multiprocessing) mode. But **HyperThreading** should be disabled into CPU BIOS SETUP.

6.3 Make Targets

The make targets are listed as the names of object-format files. Other images not listed here may not be tested.

bootrom
bootrom.bin
bootrom.pxe
vxWorks (with **vxWorks.sym**)
vxWorks.st

6.4 BSP Bootloaders and Bootroms

Bootloader/Bootrom	Status	Tested
bootrom	SUPPORTED	Tested
bootrom_uncmp	UNSUPPORTED	Not Tested
vxWorks	SUPPORTED	Tested
vxWorks_rom	UNSUPPORTED	Not Tested
vxWorks_romCompress	SUPPORTED	Not Tested
vxWorks_romResident	UNSUPPORTED	Not Tested

All following commands should be typed under the BSP vx3035 folder. To generate the different BOOTROM target type under the command line:

```
make bootrom BOOTAPP=y

make bootrom.bin BOOTAPP=y

make bootrom.pxe BOOTAPP=y
```

BOOTROM_FULL_FEATURE=y must be also set if USB device muts be accessed by bootrom

To generate the different VXWORKS targets under the command line in UP (instead of SMP):

```
make vxWorks.st TOOL=gnu (for gnu, but  TOOL=diab for diab and TOOL=icc for icc)
```

To generate the different VXWORKS targets under the command line in SMP:

```
make vxWorks VXBUILD=SMP TOOL=gnu (for gnu, but  TOOL=diab for diab and TOOL=icc for icc)
```

Warning: make from the BSP directory **(make vxWorks VXBUILD=SMP) is not supported** with vxWorks 6.9, please move to using the vxprj command line build. See above in this document.

Note that we include **INCLUDE_STANDALONE_SYM_TBL** in **uConfig.h** that includes the symbol file in the BSP

The Kontron BSP provides drivers librairies. These libraries can only be compiled under the command line and generate two library files that are present in the bsp folder of the vx3035:

libkontron.a -> All drivers in UP mode
libkontron_smp.a -> All drivers in SMP mode

6.5 BSP Validated Tools

Tool	Connection/Type	Status
Workbench Debugger	ETHERNET/WDB	SUPPORTED
Workbench System Viewer	ETHERNET/WDB	SUPPORTED
Workbench MemScope	ETHERNET/WDB	UNVALIDATED
Workbench ProfileScope	ETHERNET/WDB	UNVALIDATED
Workbench StethoScope	ETHERNET/WDB	UNVALIDATED
Workbench CoverageScope	ETHERNET/WDB	UNVALIDATED
Workbench Core File Analysis	ETHERNET/WDB	UNVALIDATED
Workbench Sensor Points	ETHERNET/WDB	UNVALIDATED
Workbench OCD Debugger	JTAG	UNSUPPORTED
Workbench OCD Flash	JTAG	UNSUPPORTED

6.6 VxWorks on VGA, PC_CONSOLE mode

This function will use PC monitor (VGA) as a VxWorks console. To use these controller, the **INCLUDE_PC_CONSOLE** directive must be enabled in **config.h**. (this is to be done before Workbench project creation). USB keyboard is supported.

6.7 intelAhciDrv and intelAhciShow

To use this WindRiver SATA Hard disk driver as a boot device, the **INCLUDE_DRV_STORAGE_INTEL_AHCI** directive must be enabled in **config.h**. VxWorks image AHCI SATA support is also enabled by the'**INCLUDE_DRV_STORAGE_INTEL_AHCI**' component.

6.8 SMBus on PCH Controller

To use onboard sensors, the Intel SMbus of Platform Controller Hub (PCH) driver must be included,(define **DRV_SMBUS_INTEL_PCH**). This driver is provided by Kontron as a 3rdparty driver.

6.9 BSP Timer Routines

The BSP provides its own routines to help for time measurement.

gettimeofdayus : gives the current timestamp with microsecond precision
gettimeofdayns : gives the current timestamp with nansecond precision
gettimeofdayShow : shows the current timestamp in seconds and microseconds
gettimeofdaynsShow : shows the current timestamp in seconds and nanoseconds

To use these routines you must include the file **kontron.h** in your application. Then a time measurement can be done for example in microseconds with the following code:

```
#include <sys/times.h>
#include <kontron.h>
```

```
struct timeval start;
struct timeval end;
int totalTime;

gettimeofdayus(&start);

 ... What you need to measure ...

gettimeofdayus(&end);

totalTime = (int)(end.tv_sec*1000000LL+end.tv_usec - (start.tv_sec*1000000LL+start.tv_usec));
printf ("Done in %d us\n", totalTime);
```

6.10 vxbI8253Timer

This WindRiver library contains a board-independent interface to manipulate the timer functions on Intel 8253 and compatible timer chip devices. Timer 0 is used as the system clock. According to different applications, timer 1,2 has different usages. (See **vxbI8253Timer.html** WindRiver documentation for more information)

6.11 MC146818 RTC clock timer driver

This library is the WindRiver vxBus compliant driver which implements the functionality specific to the MC146818 real time clock Timer (define **DRV_TIMER_MC146818** in your project). It doesn't implement the standard MC146818 RTC driver functionality but use the PCH RTC as timer functions. To load this driver, named "mc146818Rtc" it is necessary to define **DRV_TIMER_MC146818**.

See WindRiver documenation for vxBus driver **vxbMc146818Rtc**. For some code examples, see BSP file **kontron.c** routine **startTestTimer(**).

6.12 vxbIntelTimestamp

This WindRiver library is the driver for the timestamp on Intel chipsets to use the timestamp feature, the macro **INCLUDE_TIMESTAMP** must be defined in **config.h**. See **vxbIntelTimestamp.html** for more details.

6.13 USB Host Support

A standard USB EHCI / USB UHCI controller is supported in this BSP. Please refer to Wind River's USB documents.

In vx3035 bsp GEN1 storage device is the default mode. To increase the USB peformance in any case it is necessary to change it to GEN2 mode in project. Exclude **INCLUDE_MS_BULKONLY_INIT** and **INCLUDE_USB_MS_BULKONLY** in Windriver Workbench and include **INCLUDE_USB_GEN2_STORAGE** and **INCLUDE_USB_GEN2_STORAGE_INIT**.

6.14 Boot Devices

The supported boot devices are:

gei - Primary Ethernet (10baseT, 100baseTX, or 1000baseT)

ata - ATA/IDE drive (SATA, CF, PATA)

fs - USB disk

6.15 Boot Methods

The boot methods are affected by the boot parameters. If no password is specified, RSH (remote shell) protocol is used. If a password is specified, FTP protocol is used, or, if the flag is set to 0x80, TFTP protocol is used. Enter **h** command under vxWorks bootrom to have a flag description.

6.16 Software Board reset method

The standard WRS routine **reboot( )** or CTRL-X can be used to reboot on bootrom. **sysHardReset( )** command can be used to reboot the board and restart the board from BIOS. So the reset routines are

reboot - jump on bootrom

sysHardReset - reset board and restart BIOS (also reboot(2) do that)

sysVpxReset - reset board and restart BIOS and possibly reset entire VPX rack

6.17 PCI information

BspUtil.c file provides the following routines to display PCI devices configuration:

sysPciShow - list all PCI devices

sysPciHeaderShow - list PCI configuration regsiter of all PCI device

sysPciConfigFuncShow - list Function, status, PCI MEM/IO of all PCI devices

Example with **sysPciShow**

```
-> sysPciShow
------------------------------------------------------------------------------
Device Instance : 0x00710ef8
------------------------------------------------------------------------------
Scanning device on PCI Bus ...
------------------------------------------------------------------------------
Bus|De|Fu|VendorId|DeviceId| RevId |IntL|     Base Class       |    Sub Class
------------------------------------------------------------------------------
000|00| 0| 0x8086 | 0x0104 | 00.09 |0x00| Bridge               | Host
000|01| 0| 0x8086 | 0x0101 | 00.09 |0x10| Bridge               | PCI-PCI
000|01| 1| 0x8086 | 0x0105 | 00.09 |0x10| Bridge               | PCI-PCI
000|02| 0| 0x8086 | 0x0116 | 00.09 |0x10| Display Controller   | VGA
000|22| 0| 0x8086 | 0x1c3a | 00.04 |0x00| Simple Comms Cntlr   | Other
000|25| 0| 0x8086 | 0x1502 | 00.05 |0x14| Net Controller       | Ethernet
000|26| 0| 0x8086 | 0x1c2d | 00.05 |0x10| Serial Bus Cntlr     | USB
000|28| 0| 0x8086 | 0x1c10 | 11.05 |0x10| Bridge               | PCI-PCI
000|28| 4| 0x8086 | 0x1c18 | 11.05 |0x10| Bridge               | PCI-PCI
000|28| 7| 0x8086 | 0x1c1e | 11.05 |0x13| Bridge               | PCI-PCI
000|29| 0| 0x8086 | 0x1c26 | 00.05 |0x17| Serial Bus Cntlr     | USB
000|31| 0| 0x8086 | 0x1c4f | 00.05 |0x00| Bridge               | ISA
000|31| 2| 0x8086 | 0x1c03 | 00.05 |0x13| Mass Storage         | (null)
000|31| 3| 0x8086 | 0x1c22 | 00.05 |0x12| Serial Bus Cntlr     | (null)
000|31| 6| 0x8086 | 0x1c24 | 00.05 |0x12| Unknown              | Other
```

```
002|00| 0| 0x10b5 | 0x8609 | 11.10 |0x11| Bridge               | PCI-PCI
002|00| 1| 0x10b5 | 0x8609 | 11.10 |0x12| Peripheral           | Other
003|01| 0| 0x10b5 | 0x8609 | 11.10 |0x11| Bridge               | Other
005|00| 0| 0x8086 | 0x1510 | 00.01 |0x10| Net Controller       | Ethernet
005|00| 1| 0x8086 | 0x1510 | 00.01 |0x11| Net Controller       | Ethernet
------------------------------------------------------------------------------
value = 0 = 0x0
->
```

7 KNOWN PROBLEMS

7.1 GPIO Level mode not supported

The GPIO LEVEL mode is not supported. This is a CPLD board restriction. The recommended GPIO mode is Edge mode.

7.2 IPP Performance

The IPP performance is reduced in Kernel Mode comparing to RTP. Should be fixed in the next Service Pack of WindRiver.

7.3 Main memory non contiguous

When the Internal Graphic Display (IGD) is enabled in the BIOS then the available memory is not contiguous under vxWorks. This can be a problem to share memory area and in particular to use **USER_RESERVED_MEM** (see below) In that way the Kontron BSP maps the non-contiguous 512MB regions in the system to have the largest memory available. If the IGD is disabled a contiguous memory region is seen. To do that enter in SETUP by [F2]:

* Chipset -> System Agent (SA) Configuration -> Graphics Configuration -> **Internal Graphics [Disable]**

7.4 USER_RESERVED_MEM limited to 384 Mbytes

As a result of the previous limitation the maximum **USER_RESERVED_MEM** size is limited to 384 Mbytes. The workaround is to disable IGD in SETUP (see above)

7.5 Restriction on bootrom.pxe size

In PXE boot, the legacy BIOS needs to provide the capability to download a 512KB file. It seems that the current PXE file size is less that 500KB. This needs to be improved. That is the reason why the bootrom.pxe file delivered in this release is not full feature. Nevertheless, the bootrom.bin dedicated for the SATA SSD Flash or USB mass storage is full feature boot support.

7.6 BSP and vxBus driver documentation not linked to WRS documentation

When BSP is installed the VX3035 BSP and vxBus driver documentation is not linked to the WRS documentation. It would have been necessary to erase a WRS html file for this. The BSP documentation will be find at
$WIND_HOME/docs/extensions/eclipse/plugins/com.windriver.ide.doc.bsp_vxworks_6.9/vx3035

7.7 SATA Controller(s) initialization takes 5s

During boot process the SATA AHCI controllers initialization takes 5s for bootrom and 5s for kernel with or without any SATA device connected. So if SATA interface is not used SATA controller(s) should

be disabled into SETUP or **INCLUDE_DRV_STORAGE_INTEL_AHCI** must be undefined into bootrom and kernel images.

7.8 Boot on USB device (bootrom + kernel) and Ethernet problem

When booting with bootrom + kernel on USB device and network init is set (like **other: gei0**) in bootrom parameters then a long boot time (more than one minute) occurs if the network is not connected or if the network is connected with no traffic coming. The bug only occurs is SATA controller initialization is also disabled. The same boot time delay will occur at symbol table downloading if the kernel symbol table is downloaded from USB. The workaround is to start by linking the symbol table to the kernel. For the bootrom, the correction consists of initializing USB and having the USB device mounted before the network initialization. This necessitates a modification in WindRiver source file **all/usrConfig.c**. Please contact kontron support at **support-kom-sa@kontron.com** for further details.

7.9 Legacy interrupt on PCI

PCI/PCIe device IRQ could not be supported in Legacy interrupt mode. It is advised to use PCIe devices in MSI (Message Signaled Interrupt) mode. To use PCI devices on PMC in legacy interrupt mode please contact kontron support at **support-kom-sa@kontron.com**

8 BSP EVOLUTIONS

This chapter presents the evolutions to the previous BSP and drivers releases

8.1 BSP VX3035

None, first release

8.2 VX3035 vxBus driver

None first release

9 BIBLIOGRAPHY

Intel Architecture Software Developer's Manual Volume 1 Basic Architecture

Intel Architecture Software Developer's Manual Volume 2 Instruction Set Reference

Intel Architecture Software Developer's Manual Volume 3 System Programming Guide

AP-485, Intel Processor Identification and the CPUID Instruction

PCI System Architecture Fourth Edition Addison-Wesley 1999 ISBN 0-201-30974-2

10 SEE ALSO

VxWorks User's Guide Getting Started, *VxWorks Programmer's Guide Configuration*, *VxWorks Programmer's Guide Architecture Supplement*, *vxworks Application Programmers Guide 6.9*

MAILING ADDRESS
Kontron Modular Computers S.A.S.
150 rue Marcelin Berthelot - BP 244
ZI TOULON EST
83078 TOULON CEDEX - France

TELEPHONE AND E-MAIL
+33 (0) 4 98 16 34 00
sales@kontron.com
support-kom-sa@kontron.com

For further information about other Kontron products, please visit our Internet web site: www.kontron.com.