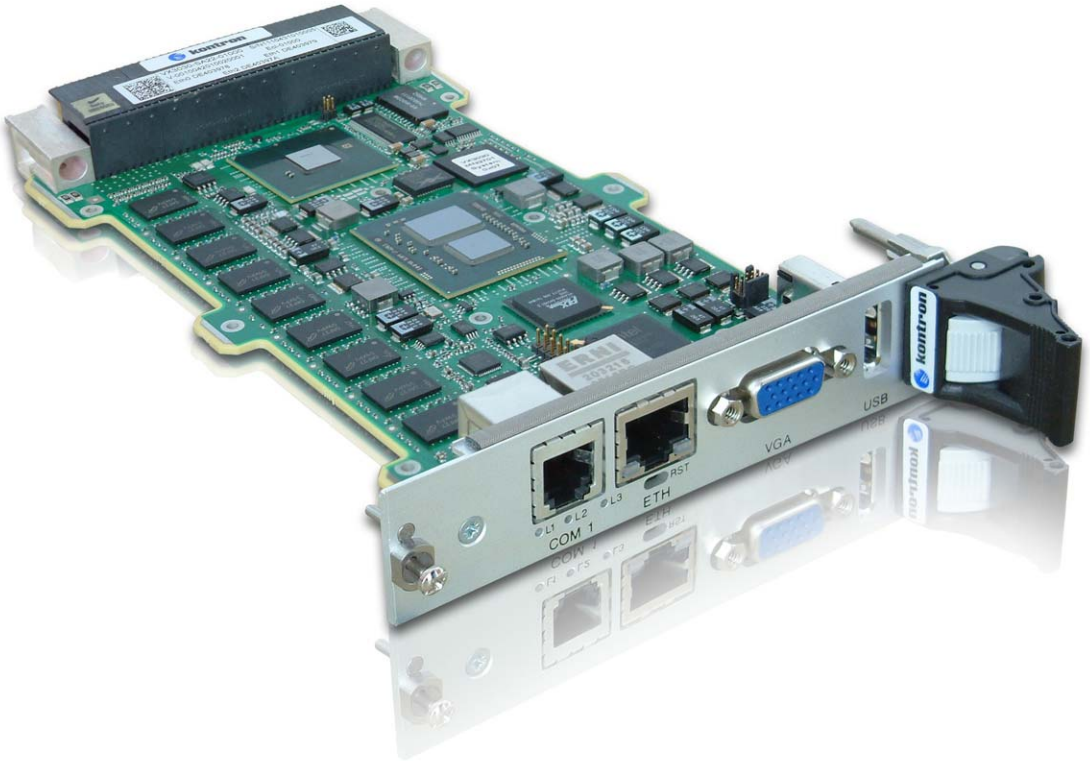
» VX3030 «



PBIT User's Guide

SD.DT.F94-0e - November 2011

Revision History

Publication Title:	VX3030 PBIT User's Guide	
Doc. ID:	SD.DT.F94-0e	
Rev.	Brief Description of Changes	Date of Issue
0e	Initial Version	11-2011

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

Chapter 1 - PBIT Overview

This document describes the PowerOn Built In Test (PBIT) for Kontron VX3030 boards.

PBIT is an optional product available under the VX3030 EFI BIOS shell environment.

PBIT is implemented as a binary executable located in the system Flash and included in the BIOS shell application. PBIT configuration such as tests list and tests result is stored in VX3030 system EEPROM.

PBIT includes among others the following services:

- The PBIT offers a list of tests that can be added or removed from a run list by command according to the desired compromise between time to boot, coverage rate and system dependent configuration.
- PBIT also offers a system test that can quickly spot any configuration change.
- PBIT can be run automatically (when booting firmware) or in an interactive mode (at EFI BIOS Shell firmware prompt).
- Tests configuration and results are stored in the system EEPROM and can also be accessed and reconfigured under Operating System such as Linux or VxWorks . See Chapter 3 page 22.
- Simplified test result is also available in a 8-bit PLD register (register 0x2).

1.1 Related Documents

Hardware:

- VX3030 3U VPX User's Guide CA.DT.A87
- VX3030 Hardware Release Notes CA.DT.A88

BIOS:

- VX3030 AMI BIOS User Reference Manual SD.DT.F81

1.2 PBIT Installation and Activation

The PBIT software comes pre−installed in the system Flash, along with the EFI BIOS firmware, on the VX3030 boards.

The PBIT is an optional product that can be activated on any VX3030. Please contact Kontron support team for more information.

To install a new BIOS version including a new PBIT version please refer to the VX3030 BIOS User's Reference Manual - SD.DT.F81.

1.3 PBIT Configuration

The PBIT must be configured first by an EFI shell command line.

The PBIT is presented as a list of tests to be executed. Each test is focused on a specific device of the VX3030.

The list of tests to be executed can be displayed and modified by using the EFI Shell command "`kdiag`" (see section 1.5 page 7).

1.3.1 Configure PBIT by command line

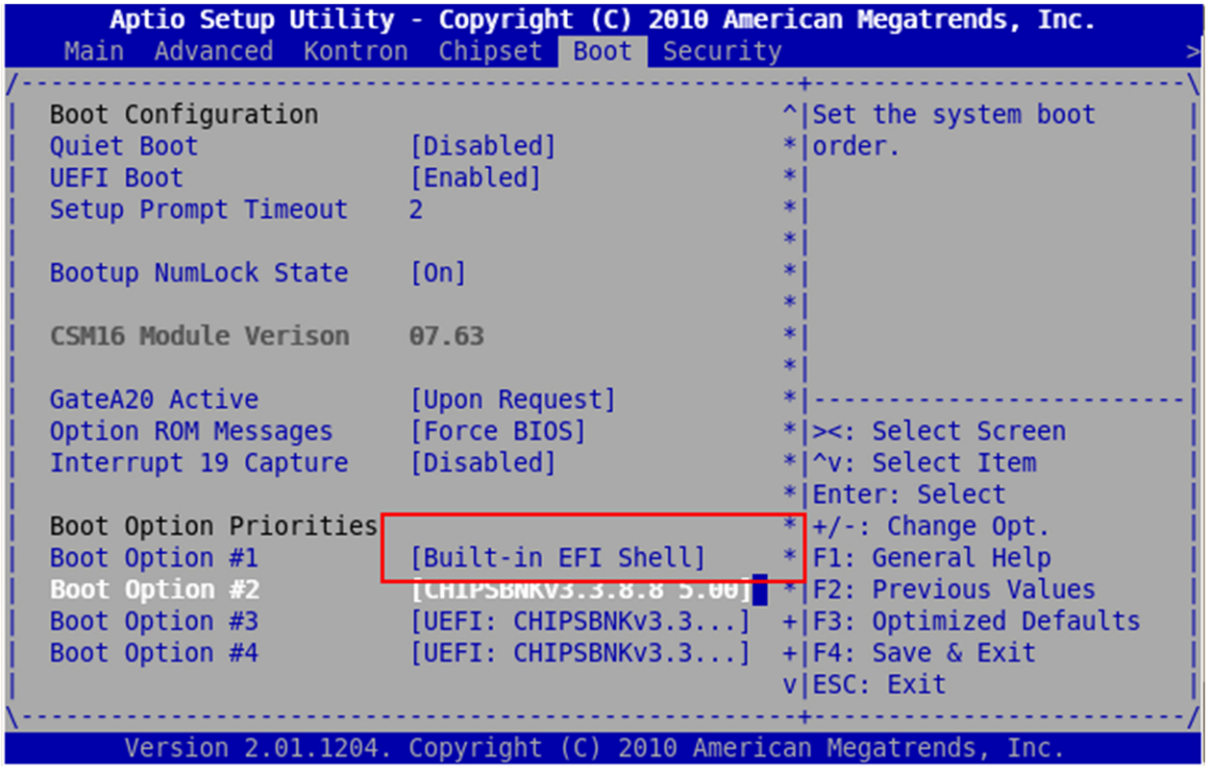
The following explains how to configure and execute PBIT by a command line.

➢ Select "`Built-in EFI Shell`" as the first boot device:

Enter the BIOS setup by pressing the **<F2>** keyboard key and select the Boot menu.

Select "`Built-in EFI Shell`" as the Boot Option #1 (use key **<+>** or **<->**).

Then, in the Save&Exit menu select «`Saving Changes and Reset`».



After reset, the EFI shell prompt is displayed, allowing to enter the PBIT commands.

➢ Verify the PBIT version:

```
VX3030> kdiag version
PBIT VERSION 1.6 ID11314
```

➢ Launch the PBIT manually for verification:

```
VX3030> kdiag run
PBIT "mem_data" (fast,simple)  PASSED
PBIT "mem_addr" (fast,simple)  PASSED
PBIT "mem_pattern1" (slow,simple)  PASSED
PBIT "mem_pattern2" (slow,simple)  PASSED
PBIT "mem_pattern3" (slow,simple)  PASSED
PBIT "mem_pattern4" (slow,simple)  PASSED
PBIT "pcie_vpx_sw" (fast,simple)  PASSED
PBIT "serial" (fast,simple)  PASSED
PBIT "rtc" (fast,simple)   BATTERY NOT EQUIPPED PASSED
PBIT "sysflash" (fast,simple)  PASSED
PBIT "cpld" (fast,simple)  PASSED
PBIT "temp_sensors" (fast,simple)  PASSED
PBIT "temperature" (fast,simple)  PASSED
PBIT "fnvram" (fast,simple)  PASSED
PBIT "ether_loop0" (fast,simple)  PASSED
PBIT "ether_loop1" (fast,simple)  PASSED
PBIT "ether_loop2" (fast,simple)  PASSED
PBIT "ether_loop3" (fast,simple)  PASSED
PBIT "ether_loop4" (fast,simple)  PASSED
PBIT "voltage" (fast,simple)  PASSED
PBIT "sata0_controler" (fast,simple)  PASSED
PBIT "sata1_controler" (fast,simple)  PASSED
PBIT "vpd" (fast,simple)  PASSED
PBIT "eeprom" (fast,simple)  PASSED
PBIT "usb1_controller" (fast,simple)  PASSED
PBIT "usb2_controller" (fast,simple)  PASSED
PBIT "system" (fast,simple)   NOT SAVED PASSED
VX3030>
```

➢ Configure the “system” test:

By default the test named “system” is not “ready”. The end user should record the system configuration when the system is ready for this (all the equipments on PCIe/PCI such as PMC, XMC cards, VPX boards and switches present on the backplane, see section 2.10 page 20 PBIT System learn test for more details).

To record and then activate this test , run the following:

```
VX3030> kdiag system_learn
Seg  Bus  Dev  Func
   ---  ---  ---  ----
    00   00   00    00 ==> Bridge Device - Host/PCI bridge
              Vendor 8086 Device 006A Prog Interface 0
    00   00   01    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 0045 Prog Interface 0
    00   00   02    00 ==> Display Controller - VGA/8514 controller
              Vendor 8086 Device 0046 Prog Interface 0
    00   00   06    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 0047 Prog Interface 0
    00   00   16    00 ==> Simple Communications Controllers - Other communicati
              Vendor 8086 Device 3B64 Prog Interface 0
    . . .
Storing system configuration ...
Done
```

➢ Check the PBIT results:

```
VX3030> kdiag stat
Status of PBITs configured to run from command line :
 PASSED : mem_data (fast,simple)
 PASSED : mem_addr (fast,simple)
 PASSED : mem_pattern1 (slow,simple)
 PASSED : mem_pattern2 (slow,simple)
 PASSED : mem_pattern3 (slow,simple)
 PASSED : mem_pattern4 (slow,simple)
 PASSED : pcie_vpx_sw (fast,simple)
 PASSED : serial (fast,simple)
 PASSED : rtc (fast,simple)
 PASSED : sysflash (fast,simple)
 PASSED : cpld (fast,simple)
 PASSED : temp_sensors (fast,simple)
 PASSED : temperature (fast,simple)
 PASSED : fnvram (fast,simple)
 PASSED : ether_loop0 (fast,simple)
 PASSED : ether_loop1 (fast,simple)
 PASSED : ether_loop2 (fast,simple)
 PASSED : ether_loop3 (fast,simple)
 PASSED : ether_loop4 (fast,simple)
 PASSED : voltage (fast,simple)
 PASSED : sata0_controler (fast,simple)
 PASSED : sata1_controler (fast,simple)
 PASSED : vpd (fast,simple)
 PASSED : eeprom (fast,simple)
 PASSED : usb1_controller (fast,simple)
 PASSED : usb2_controller (fast,simple)
 PASSED : system (fast,simple)

RUN    : 27
PASSED : 27
FAILED : 0
NOT_RUN: 0
VX3030>
```

1.3.2 Configure PBIT to run automatically

The PBIT uses BIOS environment variables to run automatically at the end of the BIOS boot and before the Operating System boot:

> Configure PBIT to be launched at boot time:

The automatic start is activated using the environment variable “bootcmd”.

```
VX3030> set bootcmd "kdiag run"
```

The delay before executing the bootcmd is given by the variable bootdelay which is expressed in second.

Default value is “1”. Value “0” is possible.

```
VX3030> set bootdelay 1
```

> Verify:

```
VX3030> set
    bootcmd   : kdiag run
    bootdelay : 1
```

> Then reset the system:

```
VX3030> reset
```

The PBIT will be launched automatically. When finished, the BIOS boots from the next valid device in the boot list.

1: To stop under the EFI shell after the PBIT execution, define the variable named “bootdontexit”:

```
VX3030> set bootdontexit    1
```

2: To execute a shell script including a list of commands, type:

```
VX3030> set startupAuto      1
VX3030> set startupdelay     1
```

Then plug an USB device including in the top directory the script file named “startup.nsh” including the wanted shell commands.

1.4 Synthetic PBIT Result

A 8-bit synthetic PBIT result can be read in the CPLD register 0x2.

This register is accessible under the Operating System using the CPLD OS facility or a direct memory I/O access at address 0x802.

Under the BIOS EFI shell use the following command:

```
VX3030> kpld -r 2
READ : @0x2 = 0x1
```

The 8-bit register 0x2 content is the following (reset value=0):

Test Fail Number 1.. 128							run
7	6	5	4	3	2	1	0

Bit 0: 0 = NOT RUN 1 = ALL RUN

Bit 1..7: if All 0 => No FAILED test, if NOT 0 then indicates ID number of first failing test

➢ Examples:

0x00 => PBIT not run

0x03 => ALL Tests run and Test 1 FAILED

0x61 => ALL Tests run and Test 48 FAILED (48 = 0x30, 0x30 << 1 = 0x60)

0x01 => ALL Tests run and PASSED

To identify a PBIT test from its number , use the command “`kdiag [PBITnumber]`”

Example:

```
VX3030> kdiag 16
 serial (16) - Checks the serial line COM2
   capabilities : fast,simple/complex
   run mode 1   : fast,simple
```

This register is set to 0 at each hardware reset. It can safely be written to 0 at any time.

1.5 PBIT Tests List

The PBIT tests list comes in two parts: a default list of selected tests and a list of additional not selected tests.

This can be changed by the user to fulfill his specific coverage and execution time requirements. The “kdiag” command displays the 2 lists. Note: the initial default tests list can be restored with the “kdiag default” command

1.5.1 Selected Tests List

The default selected tests list contains all the diagnostics that can be run without any specific equipment. All the tests have been designed to be safe for the system containing a VX3030. No signal on any connector will be modified during the default test execution.

The command “kdiag” displays the default tests list to run:

```
VX3030> kdiag
PBITs configured to run from command line :
mem_data (1) - Checks Memory/ECC data lines
mem_addr (2) - Checks Memory/ECC address lines
mem_pattern1 (6) - Checks Memory/ECC using pattern 0xFFFFFFFF
mem_pattern2 (7) - Checks Memory/ECC using pattern 0x55555555
mem_pattern3 (8) - Checks Memory/ECC using pattern 0xAAAAAAAA
mem_pattern4 (9) - Checks Memory/ECC using pattern 0x00000000
pcie_vpx_sw (13) - Checks the pciExpress backplane switch
serial (16) - Checks the secondary serial line COM1
rtc (20) - Checks the RTC time
sysflash (22) - Checks the BIOS rescue in system flash
cpld (24) - Checks PLD, GeoAddress, watchdog
temp_sensors (31) - Checks if all temperature sensors are detected.
temperature (32) - Checks if temperatures are OK.
fnvram (40) - Checks F-NVRAM device.
ether_loop0 (55) - Checks 82577LM Front Gigabit Ethernet in Loopback mode
ether_loop1 (56) - Checks 82580 Gigabit Rear Interface 1 in Loopback mode
ether_loop2 (57) - Checks 82580 Gigabit Rear Interface 2 in Loopback mode
ether_loop3 (58) - Checks 82580 Gigabit Rear Interface 3 in Loopback mode
ether_loop4 (59) - Checks 82580 Gigabit Rear Interface 4 in Loopback mode
voltage (60) - Checks the voltage sensors & value
sata0_controler (68) - Checks sata0 controller
sata1_controler (69) - Check sata1 controller
vpd (70) - Checks VPD data required for board operation.
eeprom (71) - Checks User EEPROM (0xA2)
usb1_controller (86) - Checks first Ibexpeak usb controller
usb2_controller (87) - Checks second Ibexpeak usb controller
system (88) - Checks system configuration PCIe stability

VX3030>
```

To run the default PBIT, enter the command:

```
VX3030> kdiag run
```

1.5.2 Not Selected Tests List

The second part of the list includes all the tests not currently selected for execution. These tests appear at the end of the "`kdiag`" command after the message "`Other PBITs available but not yet configured`":

```
Other PBITs available but not yet configured :
mem_bitflip (3) - Checks Mem/ECC using bit-flip pattern ((1 << (offset % 32))
mem_addrpat (4) - Checks Memory/ECC using address pattern (offset)
mem_addrpat2 (5) - Checks Memory/ECC using address pattern (~offset)
smbus0 (26) - Check SMBUS0 between PLD and backplane
smbus1 (27) - Check SMBUS1 between PLD and backplane
ether_link0 (50) - Checks the link status is "UP" on 82577LM Gigabit Network.
ether_link1 (51) - Checks the link status is "UP" on 82580 GigaEth Rear 1
ether_link2 (52) - Checks the link status is "UP" on 82580 GigaEth Rear 2
ether_link3 (53) - Checks the link status is "UP" on 82580 GigaEth Rear 3
ether_link4 (54) - Checks the link status is "UP" on 82580 GigaEth Rear 4
sata0_dev_see (63) - Checks if a SATA0 disk on rear P1 is present
sata1_dev_see (64) - Checks if a SATA1 disk on rear P2 is present
sata2_dev_see (65) - Checks if a SATA2 disk on rear P2 is present
sata3_dev_see (66) - Checks if a SATA3 disk on rear P1 is present
sata4_dev_see (67) - Checks if a SATA mezzanine device is present
usb0_dev_see (80) - Check if a USB0 device on mezzanine is present
usb1_dev_see (81) - Check if a USB1 device on front panel is present
usb2_dev_see (82) - Check if a USB2 device on rear P1 is present
usb3_dev_see (83) - Check if a USB3 device on rear P1 is present
usb4_dev_see (84) - Check if a USB4 device on rear P2 is present
usb5_dev_see (85) - Check if a USB5 device on rear P2 is present
faultytest (98) - A dummy test that returns FAIL
hangtest (99) - A dummy test that will hang
```

The unselected tests list contains:

- Memory tests to complete the default memory tests. Long execution time.
- Smbus tests: I2C tests on the backplane, reserved for complex test with specific external equipment. Do not use it.
- External equipment dependent tests to check Ethernet link, Connected SATA and USB Devices. Only use them when it is necessary to verify a device presence.
- Utility tests: "`faultytest`" helps you to test the error reporting mechanism. "`hangtest`" is useful for watchdog recovery checking.

All the USB device detection tests are limited to USB Mass storage devices. They will not detect USB keyboard or mouse.

1.6 PBIT Execution Time

The default PBIT (see section 1.5.1 page 7) runs in about 14 seconds (for a 2 GB DRAM).

Below is the execution time for each test and for a board with 2 GB DRAM.

Tests with a significant duration compared to the other tests appear in red.

Test name	Test ID number	Execution Time
mem_data (fast,simple)	1	680 ms
mem_addr (fast,simple)	2	3.6s
mem_bitflip (slow,simple)	3	1.4s
mem_addrpat (slow,simple)	4	1.4s
mem_addrpat2 (slow,simple)	5	1.4s
mem_pattern1 (slow,simple)	6	1.4s
mem_pattern2 (slow,simple)	7	1.4s
mem_pattern3 (slow,simple)	8	1.4s
mem_pattern4 (slow,simple)	9	1.4s
pcie_vpx_sw (fast,simple)	13	<300 ms
serial (fast,simple)	16	400 ms
rtc (fast,simple)	20	<300 ms
sysflash (fast,simple)	22	<300 ms
cpld (fast,simple)	24	<300 ms
temp_sensors (fast,simple)	31	<300 ms
temperature (fast,simple)	32	<300 ms
fnvram (fast,simple)	40	600 ms
ether_link0 (fast,simple)	50	<300 ms
ether_link1 (fast,simple)	51	<300 ms
ether_link2 (fast,simple)	52	<300 ms
ether_link3 (fast,simple)	53	<300 ms
ether_loop0 (fast,simple)	55	500 ms
ether_loop1 (fast,simple)	56	500 ms
ether_loop2 (fast,simple)	57	500 ms
ether_loop3 (fast,simple)	58	500 ms
ether_loop3 (fast,simple)	59	500 ms
voltage (fast,simple)	60	<300 ms
sata0_dev_see (fast,simple)	63	<300 ms
Sata1_dev_see (fast,simple)	64	<300 ms
Sata2_dev_see (fast,simple)	65	<300 ms
Sata3_dev_see (fast,simple)	66	<300 ms
Sata4_dev_see (fast,simple)	67	<300 ms
sata0_controler (fast,simple)	68	<300 ms

Test name	Test ID number	Execution Time
sata1_controler (fast,simple)	69	<300 ms
vpd (fast,simple)	70	<300 ms
eeprom (fast,simple)	71	400 ms
usb0_dev_see (fast,simple)	80	<300 ms
usb1_dev_see (fast,simple)	81	<300 ms
usb2_dev_see (fast,simple)	82	<300 ms
usb3_dev_see (fast,simple)	83	<300 ms
usb4_dev_see (fast,simple)	84	<300 ms
usb5_dev_see (fast,simple)	85	<300 ms
usb1_controller (fast,simple)	86	<300 ms
usb2_controller (fast,simple)	87	<300 ms
system (fast,simple)	88	<300 ms

Chapter 2 - PBIT Command Line Reference Guide

The PBIT are configured and executed using the EFI Shell command «`kdiag`».

The following section describes the various "`kdiag`" command parameters.

2.1 On-line Help

At EFI Shell prompt enter the command "`help kdiag`" to display the usage messages.

Note: the command formats are

- **[ ]** meaning optional parameters
- | meaning a OR choice between possible parameters
- ... meaning an undetermined number of repeated previous parameters

```
VX3030> help kdiag
kdiag    - perform board diagnostics
  ----- Usage -----
Print list of PBITs and infos about them :
  kdiag [<PBITname>|<PBITnum> ...]
    <PBITname>|<PBITnum> ...
              list of PBIT(s) to display. All if the list is empty.
              PBIT(s) are referenced using their name or their number.
Run PBIT(s) from command line :
  kdiag run [loop <count>] [<PBITname>|<PBITnum> ...]
    loop <count>
              run PBIT(s) <count> times instead of once
    <PBITname>|<PBITnum> ...
              list of PBIT(s) to run. All if the list is empty.
              PBIT(s) are referenced using their name or their number.
Print PBIT(s) status :
  kdiag stat [oside] [<PBITname>|<PBITnum> ...]
    <PBITname>|<PBITnum> ...
              list of PBIT(s) to display. All if the list is empty.
              PBIT(s) are referenced using their name or their number.
    oside (VX6060 board only): print PBIT status of peer CPU side
Clear PBIT(s) status :
  kdiag [oside] clrstat|clrallstat [<PBITname>|<PBITnum> ...]
    clrstat    : Reset status to NOTRUN
    clrallstat : Reset status to NOTRUN and clear the (FAILED once) flag
    <PBITname>|<PBITnum> ...
              list of PBIT(s) to clear. All if the list is empty.
              PBIT(s) are referenced using their name or their number.
    oside (VX6060 board only): clear PBIT status of peer CPU side
Restore default PBIT configuration :
  kdiag [oside] default
    oside (VX6060 board only): restore default PBIT for peer CPU side
```

```
Configure PBIT(s) :
  kdiag  [oside] cfg <cfgarg> ... [<PBITname>|<PBITnum>] ...
    cfg <cfgarg> : Configure one or several PBIT(s).
                   <cfgarg> is either :
                  - "delete" to delete PBIT(s) from the list of configured PBITs
                   - "default" to configure PBIT(s) with a default run mode
                   - a comma separated list of runflags defining a PBIT
                     run mode; for example : fast,complex.
                   valid runflags are :
                   - "SPEED" flags (can NOT be mixed)
                     slow          : run in slow mode (full testing)
                     fast          : run in fast mode (fast testing)
                   - "CONFIG" flags (can NOT be mixed)
                     simple        : run in simple mode (no external hardware)
                     complex     : run in complex mode (needs external hardware)
                   - "HALT" flags (can NOT be mixed)
                     haltonfail    : halt immediately (hang) if test fails
                     promptonfail  : halt at U-Boot prompt (no OS) if test fails
    [<PBITname>|<PBITnum>] ...
                   list of PBIT(s) to configure.
                   PBIT(s) are referenced using their name or their number.
                   All missing tests if the list is empty
    oside (VX6060 board only): Configure PBIT(s) for peer CPU side
Toggle PBIT running log information:
  kdiag silent
Record System Configuration for system Test:
  kdiag system_learn
Display PBIT version :
  kdiag version
```

2.2 Display the List of Selected Tests

To display the selected tests list and their configuration use the command

```
"kdiag [<PBITname>|<PBITnum> ...]".
```

Running the command “kdiag” with no argument prints the list of the tests that are selected to run with the command “kdiag run” and also the other tests not selected by default.

```
VX3030> kdiag
PBITs configured to run from command line :
 mem_data (1) - Checks Memory/ECC data lines
   capabilities : fast,simple
   run mode 1   : fast,simple
 mem_addr (2) - Checks Memory/ECC address lines
   capabilities : fast,simple
   run mode 1   : fast,simple
 mem_pattern1 (6) - Checks Memory/ECC using pattern 0xFFFFFFFF
   capabilities : slow/fast,simple
   run mode 1   : slow,simple
 mem_pattern2 (7) - Checks Memory/ECC using pattern 0x55555555
   capabilities : slow/fast,simple
   run mode 1   : slow,simple
 (...)
Other PBITs available but not yet configured :
 mem_bitflip (3) - Checks Mem/ECC using bit-flip pattern ((1 << (offset % 32))
   capabilities : slow/fast,simple
 mem_addrpat (4) - Checks Memory/ECC using address pattern (offset)
   capabilities : slow/fast,simple
 mem_addrpat2 (5) - Checks Memory/ECC using address pattern (~offset)
   capabilities : slow/fast,simple
 ether_link0 (50) - Checks the link status is "UP" on 82580 Gigabit FrontPanel Interface 0
   capabilities : fast,simple/complex
 ( . . .)
Use ´help kdiag´ to get more info.
VX3030>
```

2.3 Execute PBIT from the Command Line

To run the PBIT selected tests list from the command line, enter:

```
VX3030> kdiag run
PBIT "mem_data" (fast,simple)  PASSED
PBIT "mem_addr" (fast,simple)  PASSED
PBIT "mem_pattern1" (slow,simple)  PASSED
PBIT "mem_pattern2" (slow,simple)  PASSED
PBIT "mem_pattern3" (slow,simple)  PASSED
PBIT "mem_pattern4" (slow,simple)  PASSED
PBIT "pcie_vpx_sw" (fast,simple)  PASSED
PBIT "serial" (fast,simple)  PASSED
PBIT "rtc" (fast,simple)   BATTERY NOT EQUIPPED PASSED
PBIT "sysflash" (fast,simple)  PASSED
PBIT "cpld" (fast,simple)  PASSED
PBIT "temp_sensors" (fast,simple)  PASSED
PBIT "temperature" (fast,simple)  PASSED
PBIT "fnvram" (fast,simple)  PASSED
PBIT "ether_loop0" (fast,simple)  PASSED
PBIT "ether_loop1" (fast,simple)  PASSED
PBIT "ether_loop2" (fast,simple)  PASSED
PBIT "ether_loop3" (fast,simple)  PASSED
PBIT "ether_loop4" (fast,simple)  PASSED
PBIT "voltage" (fast,simple)  PASSED
PBIT "sata0_controler" (fast,simple)  PASSED
PBIT "sata1_controler" (fast,simple)  PASSED
PBIT "vpd" (fast,simple)  PASSED
PBIT "eeprom" (fast,simple)  PASSED
PBIT "usb1_controller" (fast,simple)  PASSED
PBIT "usb2_controller" (fast,simple)  PASSED
PBIT "system" (fast,simple)   NOT SAVED PASSED
```

To run a single test or a limited list of tests, enter:

```
VX3030> kdiag run <PBIT number | PBIT name ...>
```

➢ For example:

```
VX3030> kdiag run usb1_controller sysflash
PBIT "usb1_controller" (fast,simple)  PASSED
PBIT "sysflash" (fast,simple)  PASSED
```

That is equivalent to:

```
VX3030> kdiag run 87 22
PBIT "usb1_controller" (fast,simple)  PASSED
PBIT "sysflash" (fast,simple)  PASSED
```

The “PBIT number” is displayed by the “`kdiag`” command (with no parameter) or with the ”`kdiag <PBIT name>`” command.

2.4 Execute PBIT in Loop Mode

To run PBIT in loop mode, enter:

```
VX3030 > kdiag run loop <count>
```

with <count> being the number of loop to execute.

To run a single test in loop mode, enter:

```
VX3030> kdiag run loop <count> <PBIT number | PBIT name ...>
```

> Example: running 10 times the test ether_loop0 number 55, enter:

```
VX3030> kdiag run loop 10 55
```

It is equivalent to enter:

```
VX3030> kdiag run loop 10 ether_loop0
```

2.5 Get the PBIT Results

To get the PBIT results, use the “kdiag stat” command:

```
VX3030> kdiag stat
Status of PBITs configured to run from command line :
 PASSED : mem_data (fast,simple)
 PASSED : mem_addr (fast,simple)
 PASSED : mem_pattern1 (slow,simple)
 PASSED : mem_pattern2 (slow,simple)
 PASSED : mem_pattern3 (slow,simple)
 PASSED : mem_pattern4 (slow,simple)
 PASSED : pcie_vpx_sw (fast,simple)
 PASSED : serial (fast,simple)
 PASSED : rtc (fast,simple)
 PASSED : sysflash (fast,simple)
 PASSED : cpld (fast,simple)
 PASSED : temp_sensors (fast,simple)
 PASSED : temperature (fast,simple)
 PASSED : fnvram (fast,simple)
 PASSED : ether_loop0 (fast,simple)
 PASSED : ether_loop1 (fast,simple)
 PASSED : ether_loop2 (fast,simple)
 PASSED : ether_loop3 (fast,simple)
 PASSED : ether_loop4 (fast,simple)
 PASSED : voltage (fast,simple)
 PASSED : sata0_controler (fast,simple)
 PASSED : sata1_controler (fast,simple)
 PASSED : vpd (fast,simple)
 PASSED : eeprom (fast,simple)
 PASSED : usb1_controller (fast,simple)
 PASSED : usb2_controller (fast,simple)
 PASSED : system (fast,simple)

RUN    : 27
PASSED : 27
FAILED : 0
NOT_RUN: 0
```

2.6 Clear the PBIT Results

Upon failure of any test, a specific “FAILED ONCE” flag is set. This flag is kept even if this test is successfully PASSED later. This feature has been designed to keep track of intermittent failures.

> To clear the PBIT results enter:

```
VX3030> kdiag clrstat
```

> To clear all the PBIT history including the “FAILED ONCE” flags, enter:

```
VX3030> kdiag clrallstat
```

2.7 Configure PBIT Tests List to Execute

The list of tests to execute can be modified with the "`kdiag`" command.

Each test can be added, removed and configured with a specific run mode. If no run mode is specified then the default run mode (fast,simple) is applied.

2.7.1 Run mode parameters

The possible specific run modes are defined with the following test flags:

- `HALT` flag (can NOT be mixed):

  `haltonfail`: halt immediately (hang) if test fails

  `promptonfail`: halt at BIOS prompt (no OS boot) if test fails

  This flag offers the possibility to halt all test execution when an error is detected.

- `SPEED` flag (can NOT be mixed)

  `slow`: run in slow mode (full testing)

  `fast`: run in fast mode (fast testing)

  In the current PBIT version, no test implements a difference between fast and slow modes.

- `CONFIG` flag (can NOT be mixed)

  `simple`: run in simple mode (no external hardware)

  `complex`: run in complex mode (needs external hardware)

  Complex mode requires external devices to the VX3030 and is reserved for the factory tests.

2.7.2 Adding a Test to the Current Run List

To add a test to the run list, enter:

```
VX3030> kdiag cfg <cfgarg> ... <PBITname>|<PBITnum> ...
```

"`cfgarg`" allows to choose the test run mode.

Use the keyword "`default`" to set the default mode (typically fast and simple)

To add test 16 (serial test) with default mode, enter:

```
VX3030> kdiag cfg default 16
```

To add this test with the promptonfail flag (and the other default flags), enter:

```
VX3030> kdiag cfg promptonfail 16
```

To add this test with the complex and promptonfail flags enter:

```
VX3030> kdiag cfg  complex,promptonfail 16
```

Execute the command "`kdiag`" or "`kdiag 16`" to check the configuration:

```
VX3030> kdiag 16
 serial (16) - Checks the serial line COM2
   capabilities : fast,simple/complex
   run mode 1   : fast,complex,promptonfail
```

2.7.3 Removing a Test from the Current Run List

For example, to remove test number 16, enter:

```
VX3030> kdiag cfg delete 16
```

Verify:

```
VX3030> kdiag 16
PBIT "16" is not configured to run from command line
```

2.7.4 Add All the Not Selected Tests to the Current Run List

To add all the “not selected tests” to the current run list with a default run mode, enter:

```
VX3030> kdiag cfg default
Configuration of all PBITs in default mode only
List of test to configure
--> mem_bitflip (3) - Checks Mem/ECC using bit-flip pattern ((1 << (offset % 32))
--> mem_addrpat (4) - Checks Memory/ECC using address pattern (offset)
--> mem_addrpat2 (5) - Checks Memory/ECC using address pattern (~offset)
--> smbus0 (26) - Check SMBUS0 between PLD and backplane
--> smbus1 (27) - Check SMBUS1 between PLD and backplane
--> ether_link0 (50) - Checks the link status is "UP" on 82577LM Gigabit Network.
--> ether_link1 (51) - Checks the link status is "UP" on 82580 GigaEth Rear 1
--> ether_link2 (52) - Checks the link status is "UP" on 82580 GigaEth Rear 2
--> ether_link3 (53) - Checks the link status is "UP" on 82580 GigaEth Rear 3
--> ether_link4 (54) - Checks the link status is "UP" on 82580 GigaEth Rear 4
--> sata0_dev_see (63) - Checks if a SATA0 disk on rear P1 is present
--> sata1_dev_see (64) - Checks if a SATA1 disk on rear P2 is present
--> sata2_dev_see (65) - Checks if a SATA2 disk on rear P2 is present
--> sata3_dev_see (66) - Checks if a SATA3 disk on rear P1 is present
--> sata4_dev_see (67) - Checks if a SATA mezzanine device is present
--> usb0_dev_see (80) - Check if a USB0 device on mezzanine is present
--> usb1_dev_see (81) - Check if a USB1 device on front panel is present
--> usb2_dev_see (82) - Check if a USB2 device on rear P1 is present
--> usb3_dev_see (83) - Check if a USB3 device on rear P1 is present
--> usb4_dev_see (84) - Check if a USB4 device on rear P2 is present
--> usb5_dev_see (85) - Check if a USB5 device on rear P2 is present
```

To add all the “not selected tests” with the promptonfail flag, enter:

```
VX3030> kdiag cfg promptonfail
Configuration of all PBITs in default mode only
List of test to configure
--> mem_bitflip (3) - Checks Mem/ECC using bit-flip pattern ((1 << (offset % 32))
--> mem_addrpat (4) - Checks Memory/ECC using address pattern (offset)
--> mem_addrpat2 (5) - Checks Memory/ECC using address pattern (~offset)
--> smbus0 (26) - Check SMBUS0 between PLD and backplane
--> smbus1 (27) - Check SMBUS1 between PLD and backplane
--> ether_link0 (50) - Checks the link status is "UP" on 82577LM Gigabit Network.
--> ether_link1 (51) - Checks the link status is "UP" on 82580 GigaEth Rear 1
--> ether_link2 (52) - Checks the link status is "UP" on 82580 GigaEth Rear 2
--> ether_link3 (53) - Checks the link status is "UP" on 82580 GigaEth Rear 3
--> ether_link4 (54) - Checks the link status is "UP" on 82580 GigaEth Rear 4
--> sata0_dev_see (63) - Checks if a SATA0 disk on rear P1 is present
```

```
--> sata1_dev_see (64) - Checks if a SATA1 disk on rear P2 is present
--> sata2_dev_see (65) - Checks if a SATA2 disk on rear P2 is present
--> sata3_dev_see (66) - Checks if a SATA3 disk on rear P1 is present
--> sata4_dev_see (67) - Checks if a SATA mezzanine device is present
--> usb0_dev_see (80) - Check if a USB0 device on mezzanine is present
--> usb1_dev_see (81) - Check if a USB1 device on front panel is present
--> usb2_dev_see (82) - Check if a USB2 device on rear P1 is present
--> usb3_dev_see (83) - Check if a USB3 device on rear P1 is present
--> usb4_dev_see (84) - Check if a USB4 device on rear P2 is present
--> usb5_dev_see (85) - Check if a USB5 device on rear P2 is present

VX3030>
```

Verify for test 3:

```
VX3030> kdiag 3
mem_bitflip (3) - Checks Mem/ECC using bit-flip pattern ((1 << (offset % 32))
   capabilities : slow/fast,simple
   run mode 1   : fast,simple,promptonfail
```

The command will not include the test numbers greater than 95 (hangtest, faulty) because these tests are tool tests for PBIT validation.

2.7.5 Restore the Default Run List

To restore the default run tests list, enter:

```
VX3030> kdiag default
```

2.8 Run Pbit in Silent Mode

To avoid the PBIT to display test messages during execution, use the toggle command:

```
VX3030> kdiag silent
PBIT set in silent mode
```

To disable the silent mode, re-execute the same command:

```
VX3030> kdiag silent
PBIT silent mode removed
```

1: In this mode error messages are displayed anyway.
2: To prevent any output messages to the serial line, use the BIOS setup configuration (Serial Line Console Redirection).

2.9 Display the PBIT Version

To display the PBIT version, enter:

```
VX3030> kdiag version
PBIT VERSION 1.6 ID11314
```

2.10 PBIT System Learn Test

The command "`kdiag system_learn`" is used to record the current system configuration.

It must be run when the system configuration is the correct one to be recorded. All the detected PCI devices are recorded (the list is visible with the BIOS shell command "pci").

Then, the test named "`system`" is used to read the current system configuration and check it against the recorded configuration.

➢ Example:

```
VX3030> kdiag system_learn
   Seg  Bus  Dev  Func
   ---  ---  ---  ----
    00   00   00    00 ==> Bridge Device - Host/PCI bridge
              Vendor 8086 Device 006A Prog Interface 0
    00   00   01    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 0045 Prog Interface 0
    00   00   02    00 ==> Display Controller - VGA/8514 controller
              Vendor 8086 Device 0046 Prog Interface 0
    00   00   06    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 0047 Prog Interface 0
    00   00   16    00 ==> Simple Communications Controllers - Other communicati
              Vendor 8086 Device 3B64 Prog Interface 0
    00   00   16    01 ==> Simple Communications Controllers - Other communicati
              Vendor 8086 Device 3B65 Prog Interface 0
    00   00   16    02 ==> Mass Storage Controller - IDE controller
              Vendor 8086 Device 3B66 Prog Interface 85
    00   00   16    03 ==> Simple Communications Controllers - Serial controller
              Vendor 8086 Device 3B67 Prog Interface 2
    00   00   1A    00 ==> Serial Bus Controllers - USB
              Vendor 8086 Device 3B3C Prog Interface 20
    00   00   1C    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 3B42 Prog Interface 0
    00   00   1C    04 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 3B4A Prog Interface 0
    00   00   1C    06 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 3B4E Prog Interface 0
    00   00   1D    00 ==> Serial Bus Controllers - USB
              Vendor 8086 Device 3B34 Prog Interface 20
    00   00   1E    00 ==> Bridge Device - PCI/PCI bridge
              Vendor 8086 Device 2448 Prog Interface 1
    00   00   1F    00 ==> Bridge Device - PCI/ISA bridge
              Vendor 8086 Device 3B07 Prog Interface 0
    00   00   1F    02 ==> Mass Storage Controller - UNDEFINED
              Vendor 8086 Device 3B2F Prog Interface 1
    00   00   1F    03 ==> Serial Bus Controllers - System Management Bus
              Vendor 8086 Device 3B30 Prog Interface 0
    00   04   00    00 ==> Network Controller - Ethernet controller
              Vendor 8086 Device 150E Prog Interface 0
    00   04   00    01 ==> Network Controller - Ethernet controller
              Vendor 8086 Device 150E Prog Interface 0
    00   04   00    02 ==> Network Controller - Ethernet controller
              Vendor 8086 Device 150E Prog Interface 0
```

```
      00   04   00    03 ==> Network Controller - Ethernet controller
               Vendor 8086 Device 150E Prog Interface 0
      00   05   00    00 ==> Bridge Device - PCI/PCI bridge
               Vendor 10B5 Device 8112 Prog Interface 0
      00   06   04    00 ==> Network Controller - Ethernet controller
               Vendor 8086 Device 1010 Prog Interface 0
      00   06   04    01 ==> Network Controller - Ethernet controller
               Vendor 8086 Device 1010 Prog Interface 0
      00   06   09    00 ==> Bridge Device - Other bridge type
               Vendor 1059 Device 9035 Prog Interface 0

  Storing system configuration ...
  Done

  VX3030> kdiag run system
  PBIT "system" (fast,simple)   PASSED
```

There is no command to clear the system_learn configuration. It must be recorded again if the configuration has changed or a test not required anymore needs to be removed from the run tests list.

Chapter 3 - PBIT and OS Interfaces

The PBIT synthetic result and PBIT detailed results are accessible under the Operating System delivered with the VX3030 board.

The following gives you an overview of the facilities available under Linux, VxWorks and Windows OS Board Support Packages.

Please refer to the appropriate OS Release Notes document for more details.

3.1 Linux

3.1.1 Linux Synthetic PBIT Result

The synthetic PBIT result stored in the CPLD register 0x2 is accessible under Linux with the "`cpldtool`" facility.

Use the "`-a`" option and check the register 0x2 value.

➢ Example:

```
# cpldtool -a
Reg 0x0 - CPLD_ID = 0x0B
        CPLD_ID=0x0
        CPLD_Debug=0x0
        CPLD_Version=0xB
Reg 0x1 - PCB_ID = 0x50
Reg 0x2 - FIRM_PWON = 0x00
Reg 0x3 - PWON_STATUS = 0x00
Reg 0x4 - PWR_RST_CONFIG = 0x21
        PWRON_MODE=0x0
        PLTRST_to_PERST3U_INHIB=0x0
        Alarm_inhib=0x1
        PLTRST_to_PERST6U_INHIB=0x0
(....)
```

3.1.2 Linux Detailed PBIT Result

The detailed PBIT results are accessible under Linux with the "`sysvartool`" facility.

Use the "`-A pbit -l`" options to display the PBIT results.

> Example:

```
# sysvartool -A pbit -l
VX3030 detected
area = 2, arch = 2
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
ether_loop0: PASSED
ether_loop1: PASSED
ether_loop2: PASSED
system: PASSED (FAILED ONCE)
PASSED : 10
FAILED : 0
NOT RUN : 0
TOTAL : 10
```

3.2 VxWorks

3.2.1 VxWorks Synthetic PBIT Result

The synthetic PBIT result stored in the CPLD register 0x2 can be accessed with a memory I/O access at address I/O 0x802.

The command "`sysInByte(0x802)`" will return the synthetic PBIT result.

```
-> sysInByte(0x802)
value = 1 = 0x1
```

3.2.2 VxWorks Detailed PBIT Result

The detailed PBIT results (if implemented) is accessible using the "`pbitdisplay()`" call.

```
-> 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: PASSED
sysflash: PASSED
cpld: PASSED
temp_sensors: PASSED
temperature: PASSED
fnvram: PASSED
ether_link0: PASSED
ether_loop0: PASSED
ether_loop1: PASSED
ether_loop2: PASSED
ether_loop3: PASSED
ether_loop4: PASSED
voltage: PASSED
sata4_dev_see: PASSED
sata0_controler: PASSED
sata1_controler: PASSED
vpd: PASSED
eeprom: PASSED
usb1_controller: PASSED
usb2_controller: PASSED
system: PASSED

PASSED  : 29
FAILED  : 0
NOT RUN : 0
TOTAL   : 29

value = 0 = 0x0
->
```

3.3 Windows XP, XPe and Seven

The Windows XPe BSP for VX3030 provides a Kontron embedded API to access the devices.

The detailed PBIT results are accessible by running the executable “`post.exe`”.

Chapter 4 - Use Cases

The following “Use Cases” are covered by PBIT:

- EVAL (platform EVALuation)
- DEVELOP (application or system DEVELOPment)
- DEPLOY (system production at the factory and DEPLOYment in the field)
- MAINTAIN (specific use case to MAINTAIN the system once deployed or to help repairing it)
- MANUFACTURING (used mostly by Kontron)

4.1 PBIT Features and Benefits

The following describes the PBIT features and benefits according to targeted Use Cases.

4.1.1 EVAL

PBIT can easily be evaluated from the BIOS, thanks to its interactive mode. Access to the PBIT operator interface and to the results can be done through the console serial line or through the VGA or DP (Display Port) screen associated with a USB mouse and keyboard.

Following the detailed sections of this manual, all the PBIT commands can be experimented.

4.1.2 DEVEL

The PBIT tests list can be modified in order to try various coverage/execution time trade-offs.

The “system” test can be used at the end of the development, to capture a complex peripheral configuration (PCIe, PCI, USB, SATA) which becomes the validated final system configuration for the deployment.

According to the complete system test policy, various PBIT results collection methods can be selected at application design time. The Operating System access to the PBIT results gives most of the information needed by a control and monitoring application to take a decision on the system following the boot steps.

The PBIT synthetic result register featured by the board control unit (cPLD) can be used from a management unit such as Kontron CMB or from other boards linked to the same SMB bus on the backplane. From this result, the Control Unit can alter the control registers to modify the behaviour of the board under control.

4.1.3 DEPLOY

PBIT software implements unrivalled features which make Kontron product deployment easier and cheaper. The PBIT code is located in the same device as the BIOS code. It is deployed along with the BIOS (see Appendix A of the BIOS User’s Guide). The PBIT settings (and results) are located in their dedicated non volatile EEPROM device. To copy these settings, boot the Operating System and record the VPD EEPROM content from the offset address 0x1000 up to 0x3FFF and store it in a file. Then use this file to copy back data to the new board VPD EEPROM to configure.

One of the most useful features of PBIT for deployments is implemented in the “system” test (see section 2.10 page 20). As this test scans all the possible I/O ports and devices to compare with a pre-recorded system reference, it can signal any change in the expected I/O devices availability. This can cover I/O device failures, or more subtle system alterations (for example a storage device left unconnected). The “system” test results are organized to pinpoint the faulty I/O port easily, before attempting a complete system boot and having to manage the corresponding cascading failures due to this incorrect peripheral configuration.

4.1.4 MAINTAIN

The following PBIT features can be very useful in the context of long term maintenance and troubleshooting.

The PBIT results record the behaviour of the last PBIT run. In addition to them, each test implements a dedicated “FAILED ONCE” flag which has to be reset separately. In situations where a system is regularly re-started, possibly under the control of a higher level control device, this long term flag can be used by maintenance teams to search for the root cause of previously aborted boots.

Updating the BIOS and the PBIT code is done at the same time, using binary images containing both software, thanks to the same storage device being used by both.

PBIT extension: Thanks to its modular approach, and to the EFI execution environment. PBIT can easily be modified and expanded to match very specific application use case. Please contact support-kom-sa@kontron.com to open a feature request discussion with Kontron.

4.1.5 MANUFACTURING

Thanks to the wide range of services the PBIT software can offer, PBIT is a powerful tool used during the manufacturing process of the boards.

Specific features have been implemented for that purpose.

Appendix A - List of Abbreviations

CPLD	Complex Programmable Logic Device
OS	Operating System
PBIT	Power on Built In Test
PMC	PCI Mezzanine Card
XMC	PCI Express Mezzanine Card

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.