



AN10308

PNX8525 Limitations List

Rev. 01 – 5 November 2004

Application note

Document information

Info	Content
Keywords	PNX8525, PNX8526
Abstract	Describes the limitations encountered so far in the PNX8525

Revision history

Rev	Date	Description
01	20041105	The format of this document has been redesigned to comply with the new presentation and information standard of Philips Semiconductors.
0	20031211	Split into Generic Section 2 and PNX8525 Specific Section 3 . Added Section 2.44 to Section 2.54 inclusive and Section 3.2 .
	20020610	SMARTCRD03, SMARTCRD04 and SMARTCRD05 added.
	20020225	VMSP04 deleted; not a Limitation any more.
	20020121	PCIXIO13 added
	20020118	SMARTCRD01 modified, USB01 modified, PCIXIO12 modified and VMSP04 added.
	20011221	UART02, AICP03 and PCIXIO12 added.
	20011114	UART01 and MPEG01 added; PCIXIO11 modified.
	20010817	PCIXIO11, USB0 , TSDMA01and GPIO02 added.
	20010614	Format change .TM01, VMSP02, JTAG01, PCIXIO05, PCIXIO08 , TM02, PCXIO09, VMSP03 and PCIXIO10 added.
	20010524	"B1F only" limitations removed. PI burst to XIO added.
	20010419	Cosmetic changes
	20010416	Initial version

Contact information

For additional information, please visit: <http://www.semiconductors.philips.com>

For sales office addresses, please send an email to: sales.addresses@www.semiconductors.philips.com

1. Introduction

This document describes the limitations encountered so far in the PNX8525. It is a living document subject to changes depending on the IC validation/evaluation progress as well as the fixes implemented in future versions.

Each limitation is briefly described. An evaluation of its severity is provided and when possible its workaround is suggested. The severity levels are the followings:

1. **Critical:** The limitation is very severe. It does threaten the usage of the IC in all applications. It can also affect the possibility to evaluate the IC.
2. **Severe:** The limitation makes the affected feature useless but this feature is not critical in typical applications (it could however prevent the usage of the IC in specific applications).
3. **Medium:** The limitation affects a specific non-critical feature. A workaround often exists but is not ideal.
4. **Low:** The limitation can be overcome by an acceptable workaround.

2. PNX852x generic limitations

2.1 GPIO01: GPIO VIC interrupt

2.1.1 Summary

The GPIO block has 5 interrupts routed directly to the PIC. These 5 interrupts are simple ORed together to give you one interrupt which goes to the TriMedia™ VIC.

If TriMedia™ is using the GPIO-VIC as its interrupt source & if both Mips and TriMedia™ processor are using different GPIO interrupts.

Then if a 'Mips' GPIO interrupt is raised; this will also cause an unwanted TriMedia™ interrupt to be raised. There is no way of disabling GPIO certain interrupts to the VIC. Software could handle this case of spurious and unwanted GPIO-VIC interrupts to the TriMedia™ CPU. However it is a waste of the TriMedia's™ time.

2.1.2 Severity

Low.

2.1.3 Workaround

Software has a simple solution to this, which is to use the PIC to handle the TriMedia™ GPIO interrupts, and ignore the GPIO-VIC interrupt.

2.2 GPIO02: GPIO programmed I/O output cannot be used as a GPIO DMA clock

2.2.1 Summary

A programmed i/o GPIO output can not be used as a GPIO DMA clock. Setting a GPIO pin as a GPIO DMA clock forces the clock pin to get its output data from the clock for the DMA queue. This is circular. The clock pin is being driven by itself. So the clock stays at the last level it was at before the GPIO_Evx register was programmed.

2.2.2 Severity

Medium.

2.2.3 Workaround

None.

2.3 MBS01 VIP01: MBS/VIP truncation of LSBs in non-linear scaling leads to complex math

2.3.1 Summary

The cascaded integrator used in the non-linear scaling mechanism in MBS & VIP truncates some LSBs, which makes precise calculating of panorama coefficients (dzoom & ddzoom) unnecessarily complex.

2.3.2 Severity

Low.

2.3.3 Workaround

Use pre-calculated parameters from table if needed.

2.4 VIP02: VIP VBI one line capture is not working

2.4.1 Summary

The VIP VBI capture is not able to capture just one line of data.

2.4.1.1 Severity

Low.

2.4.1.2 Workaround

Software workaround is to capture two lines instead.

2.5 MBS02: MBS 3-field majority select de-interlacing does not work

2.5.1 Summary

MBS 3-field majority select de-interlacing does not work.

2.5.2 Severity

Low.

2.5.3 Workaround

None.

2.6 GLOBAL01: GLBREG2 Read DEADABBA from T-PI modules

2.6.1 Summary

In some cases the registers of all modules on the T-PI bus read back 0xDEADABBA, when read from an external CPU via PCI. This happens sometimes when a TriMedia™ program was started directly after chip reset.

2.6.2 Severity

Low.

2.6.3 Workaround

Clear TM_OWNED_PCI_XIO bit in Global Registers 2.

2.7 PCIXIO01: PCI external master reads swaps MMIO registers in big endian mode

2.7.1 Summary

When Viper is running in big Endian mode 32-bit reads by an external PCI master from MMIO registers are byte-swapped. They should go through without swapping. This might prevent software running in Big Endian mode from an external host.

2.7.2 Severity

Medium.

2.7.3 Workaround

Software workaround.

2.8 PCIXIO02: PCI arbiter setting

2.8.1 Summary

PCI Arbiter can be either external or internal, controlled by PCI Setup Register programming (Offset 0x0010). But this register is only a "Write once" register, written by the boot loader block i.e. the boot scripts, which are hard coded.

So PCI Arbiter configuration is built-in and NOT user programmable.

2.8.2 Severity

Low.

2.8.3 Workaround

None.

2.9 PCIXIO03: XIO profile conflict

2.9.1 Summary

The XIO port may be accessed at any time after the configuration registers have been initialized. Up to three profiles may be enabled at one time - but only if the three profiles have a different type (NAND, NOR flash, IDE, Motorola Device).

If two devices with the same profile are connected to 2 different Chip Select, a bus contention or/and a blocking of PI bus can occurs during an access to one of this device.

2.9.2 Severity

Low.

2.9.3 Workaround

None.

2.10 PCIXIO04: PCI read lifetime timeout

2.10.1 Summary

This bug may appear under the following conditions: External master/host initiates a read cycle that ends in a PCI "retry" because Viper data is not ready. However, If the host does not return for the requested data in the lifetime timeout time (i.e. host speculative reads), the Viper PCI will hang.

2.10.2 Severity

Low.

2.10.3 Workaround

None.

2.11 PCIXIO05: PCI: multi-word read after retry

2.11.1 Summary

If a single word read PCI cycle is initiated by an external PCI master device and this cycle is terminated in a retry and this retry is done by the PCI master with a read of more than 1 word then the first word is gotten and then the PCI/XIO bridge blocks the PCI bus by keeping its DEVSEL line low.

2.11.2 Severity

Low.

2.11.3 Workaround

One workaround is to set the 'en_retry_timer' bit in the pci control register to '0'. The register is located at address offset 0x40014 from MMIO base (Base14).

This will eliminate the "retrys" which is causing the problem. The side-effect from this will be that a PCI access can occupy the PCI bus for an extended amount of time.

2.12 PCIXIO06: PCI issued transaction may fail

2.12.1 Summary

PCI issued transaction may fail following the response to an external master, which stalls the last data phase.

2.12.2 Severity

Low.

2.12.3 Workaround

None.

2.13 PCIXIO07: Direct PI access of XIO with opcodes larger than WDU does not work

2.13.1 Summary

The STB-DMA was used to directly access the XIO. On requests larger than 4 bytes the Pi bus hangs on the last.

2.13.2 Severity

Medium.

2.13.3 Workaround

The built-in PCI-XIO DMA can be used for DMA from/to the PCI -XIO. STB-DMA is functional for SDRAM to SDRAM DMA.

2.14 PCIXIO08: Nand Flash DMA usually limited to 512 bytes

2.14.1 Summary

Most NAND Flash have a segment size of 512 bytes. When reading more than one segment, a "break in access" must be provided. Some Flash vendors provide a "gapless" read option. When using this type of device it may be possible to set the DMA to larger than 128 data phases (512 bytes). Note: The DMA does not monitor the "ACK" signal after DMA has started. This restricts the DMA burst size to 128 data phases when doing conventional flash reads.

2.14.2 Severity

Low.

2.14.3 Workaround

Review NAND Flash specification to determine optimal DMA settings.

2.15 PCIXIO09: GPXIO read data might be lost after a direct XIO read

2.15.1 Summary

When using the GPXIO to read an XIO device, the read data stored in register 40828 must be read before a direct read from another XIO occurs or data will be overwritten with the results of the direct read.

2.15.2 Severity

Low.

2.15.3 Workaround

CF summary.

2.16 PCIXIO10: XIO accesses should be restricted to valid areas within the XIO aperture

2.16.1 Summary

To avoid hanging the bus, XIO accesses should be restricted to valid areas within the XIO aperture. These are where xio_sel0profile, xio_sel1profile, and/or xio_sel2profile are located.

2.16.2 Severity

Low.

2.16.3 Workaround

CF summary.

2.17 PCIXIO11: PCI internal arbiter is not working

2.17.1 Summary

If the PCI internal arbiter is set and if a PCIXIO access takes place as another PCIXIO access is pending, the access will not take place on the PCI bus but the internal state machine will believe it took place. This bug only affect application using external PCI devices. If the application includes only XIO devices, the internal arbitration works OK.

2.17.2 Severity

Medium-high.

2.17.3 Workaround

The work around is to use an external PCI arbiter, which does not park on Viper. A specific external arbiter is available. The code is available in Verilog and VHDL for both 3 and 4 masters.

2.18 MMI01: SDRAM self refresh broken

2.18.1 Summary

The self-refresh mode commonly used for power down is not functional. When Viper initiates self-refresh it does not send the proper command to the SDRAM. This causes the SDRAM to lose its contents.

2.18.2 Severity

Medium.

2.18.3 Workaround

Power saving mode can be achieved by switching the memory clock to the crystal frequency.

2.19 AICP01: AICP KeyFwd feature not working

2.19.1 Summary

The KeyFwd register for controlling the key forward function in the AICP mixer has no effect. The consequence is that color key information can not be forwarded from one layer to another on an individual basis.

2.19.2 Severity

Low.

2.19.3 Workaround

None.

2.20 AICP02: multiple signatures for AICP signature generating mechanism

2.20.1 Summary

The problem is caused due to resynchronization of the vsync into the pi clock domain. This causes the vsync to start the signature generation process with +/- 1clk. This results in two possible signatures for a stable input picture. The two signatures are predictable and therefore SW has only two instead of one signature to compare to validate a given picture.

2.20.2 Severity

Low.

2.20.3 Workaround

See Summary.

2.21 VMSP01: VMSP queue lock; does not output data if head ptr set to beginning of buffer and limit set to full after the first lock

2.21.1 Summary

If the VMSP data queue pointer is set to the beginning of the queue buffer and the lock limit is set to queue Full (0x4) after the VMSP queue has been locked at least once, then no data is being output to the queues. Note that if the queue has not been locked and the same setting is done then data gets output to the queue. At which threshold the queue got locked does not matter.

2.21.2 Severity

Low.

2.21.3 Workaround

Software workaround implemented.

2.22 VMSP02: serial transport in not functional

2.22.1 Summary

The serial transport in, on either TSIN1, or TSIN2, does not pass through to the MSP.

2.22.2 Severity

Medium.

2.22.3 Workaround

Use the parallel transport in.

2.23 VMSP03: VMSP GP packet insertion mode

2.23.1 Summary

In GP and packet insertion modes and in steady state, the first byte of each packet output is the expected last byte of previous packet in VMSP0 and VMSP1.

2.23.2 Severity

Medium.

2.23.3 Workaround

None.

2.24 PI01: PiMi bridge burst transfers issue

2.24.1 Summary

Burst transfers over the PiMi bridge to MMI with length of 64 or 128 bytes will only work when the address is naturally aligned. In general, Viper modules that require large PI burst lengths have a direct path to MMI.

2.24.2 Severity

Low.

2.24.3 Workaround

None.

2.25 139401: 1394 SFTRST bit functionality

2.25.1 Summary

GLOBSCR (0x049018) register of 1394 core inside the Viper has a SFTRST bit; supposedly this bit should reset the 1394 core and bring the core to its reset state. This bit is NOT functional.

2.25.2 Severity

Low.

2.25.3 Workaround

Software workaround.

2.26 139402: 1394 endianness issue

2.26.1 Summary

1394 core inside the Viper does not allow us to set the Endianness of the core. Since Viper MIPS can run as Little or Big Endian, host (MIPS) can pump data in both Endian variations. The bus is in Big Endian order, therefore if host (MIPS or in my case x86) is running Little Endian, heavy swapping required for Asynchronous communication. This is a strictly performance issue.

2.26.2 Severity

Medium.

2.26.3 Workaround

Software workaround.

2.27 139403: 1394 EN_LDTR bit functionality

2.27.1 Summary

ASYCTL has a bit called EN_LDTR, user can set it to 1 or 0, but in Viper case due to speculative reads of Trimedia™, setting to 0 is not operational.

2.27.2 Severity

Low.

2.27.3 Workaround

Software workaround.

2.28 139404: 1394 ping response problem

2.28.1 Summary

According to 1394 spec, when you ping another node on the bus, the pinged node should send self-id packets (responder PHY chips sends this packet). In Viper case (with the TI 41LVA03 phy), we get the first quadlet padded by the link core (0x000000E0) and we get the second quadlet (self-id) but we don't get anything else, we should get the ack quadlet too.

2.28.2 Severity

Low.

2.28.3 Workaround

Software workaround.

2.29 SMARTCRD01: SmartCard interrupt clearing does not work if pi-clk > clk_smc card

2.29.1 Summary

When the pi_clk is greater than clk_smartcard, the SmartCard remains active, even after the Status Register is cleared. The problem does not occur for the frequencies, which is currently used (i.e. pi_clk = 75 MHz and clk_smartcard = 54 MHz).

2.29.2 Severity

Low (clk smartcard = 54 MHz, clk_pi < 108 MHz in Viper).

2.29.3 Workaround

Software workaround - suggested workaround is to clear the Interrupt X twice, instead of clearing the Interrupt X once.

2.30 SMARTCRD02: SmartCard DMA autoflush not working

2.30.1 Summary

See Title.

2.30.2 Severity

Low.

2.30.3 Workaround

Use manual flush.

2.31 2D01: optimizations for 2D missing

2.31.1 Summary

The 2D Graphics block can do its operations either left-to-right or right-to-left. For example, copying a block of pixels up or to the left uses left-to-right, while copying down or to the right uses right-to-left.

On PNX8525, the operations that proceed right-to-left are noticeably slower.

2.31.2 Severity

Low.

2.31.3 Workaround

None.

2.32 JTAG01: synchronization between the TCK and internal clock

2.32.1 Summary

The JTAG block, when used as a Trimedia™ debugger port uses 2 clocks: TCK and PI.

When the Trimedia™ try to clear the lfull bit in the TMdbg, the clear occurs only two TCK rising edge later. Since TCK is not a free running clock, the TM core may clear that bit and read back the old value, seeing it as a new lfull.

2.32.2 Severity

Low.

2.32.3 Workaround

Adding dummy clock cycle after each command/data solve the issue.

2.33 USB01: buffer end in TD descriptor should not en with bits “10”

2.33.1 Summary

When transferring data from the USB host, the transfer does not take place if the two least significant bits of the TD descriptor's Buffer End Address are equal to '10', i.e. addresses ending with 0x2, 0x6, 0xa, 0xe.

2.33.2 Severity

Medium - high.

2.33.3 Workaround

The application or the drivers have to make sure that this event does not happen. For that, one must program both the Buffer start and Buffer end fields in the TD descriptor to ensure that the right transfer size takes place.

2.34 TSDMA01: transfer of multiple of 512 bytes do not happen

2.34.1 Summary

For a TSDMA transfer, if the DMA length is a multiple of 512 bytes (i.e. 512 bytes, 1 KB, 2KB..), the transfer will not happens due to a comparison error in a counter (To start the DMA, the block compare the 9 lsb of the DMA length with 0).

2.34.2 Severity

Low.

2.34.3 Workaround

If a DMA has to be a multiple of 512 byte. It should be broken into two smaller DMA (not multiple of 512 bytes).

2.35 UART01: UART transmitter inserts an extra stop bit

2.35.1 Summary

The Viper UART adds an extra STOP bit after each character. If the UART is programmed for 1 stop bit, it produces 2, if programmed for 2 stop bits, it produces 3. The consequence is transmission speed is slower than expected.

2.35.2 Severity

Low.

2.35.3 Workaround

Program the UART for one less stop bit to achieve desired results.

2.36 MPEG01: ½ HD MPEG mode algorithm in MPEG pipe does not produce acceptable quality

2.36.1 Summary

In ½ HD mode of operation, the I frame is reduced by ½ in the horizontal resolution (e.g. to 960x1080i). The P frames are reconstructed by up-interpolated "I" frame and MV and then written back out to memory in ½ resolution. Under certain MV conditions, this can cause the P frames to accumulate error since the P frames were not constructed from the true non-interpolated "I" or "P" frame. Thus, ½ HD MPEG mode algorithm does not produce a visually acceptable quality.

2.36.2 Severity

Medium.

2.36.3 Workaround

None.

2.37 UART02: Viper Uart glitches TX signal between stop bits when set for 8 bit, parity, 2 stop bits

2.37.1 Summary

When the Viper UART is set for 8 bit, parity, 2 stop bits, the transmitter puts a small glitch on the TX signal between the two stop bits. The glitch is one 16x clock cycle wide.

2.37.2 Severity

Low.

2.37.3 Workaround

Because of problem UART01, the UART adds an extra stop bit, programming the UART for 1 stop bit will produce 2 real stop bits without the glitch.

2.38 AICP03: AICP 8 bit illegal code limiter does not work

2.38.1 Summary

The elimination of illegal codes for 8 bit AICP output formats does not work. The consequence is that foreign 8 bit devices connected to the D1 output of viper will not work properly and could receive false sync information as a result of the illegal codes passed out by Viper.

2.38.2 Severity

Medium.

2.38.3 Workaround

Elimination of these codes could be done either in the Gamma LUT or by subtracting 4 off all Y values with the offset adder in the color space conversion block.

2.39 PCIXIO12: PCI burst lengths exceeding 128 bytes are not supported

2.39.1 Summary

Performing PCI burst transfers greater than 128 bytes, may result in garbage data being sent.

There are known problems in the FIFO full/empty conditions of PCI to/from memory. Limiting the burst length to 128 bytes will avoid these corner cases. This will apply to all external PCI masters as well as the PCI DMA agent.

2.39.2 Severity

Medium.

2.39.3 Workaround

Software needs to limit all PCI burst transactions to 128 bytes or less.

The PCI DMA has a built in burst length restriction mechanism, bits [7:5] of the DMA controls register. This will allow SW to set up the arbitrary DMA length, with the DMA partitioning it into bursts of a maximum length of 128 bytes.

External PCI masters may or may not have this feature. If not, SW will have to partition the bursts itself or rely on the latency timer to limit the burst size.

The same restriction applies to DMA over XIO although it is unlikely to come into play as the XIO is much slower than PCI. Each 4 byte word will be a minimum of 8 PCI clocks verses 1 clock per data phase on PCI.

2.40 PCIXIO13: PCI : DMA Transfers Across PCI bridges are not supported

2.40.1 Summary

Bus master on the secondary bus, of the PCI to PCI Bridge, initiates a DMA transaction to Viper. Viper initiates a read, on the primary bus, from a target on the secondary bus. If the DMA from the secondary bus has not yet completed, and the write posting buffer of the

Bridge is not clear, then the Bridge rejects the read request. Now Viper, having a read request pending, rejects the DMA transaction from the Bridge. At this point there is a Deadlock, with both Viper and Bridge refusing to forward bus transactions.

2.40.2 Severity

Medium.

2.40.3 Workaround

A workaround may be devised specific to the Platform and Hardware being used.

2.41 SMARTCRD03: SMARTCARD generates infinite number of parity errors when acknowledging first parity error

2.41.1 Summary

In T=0 protocol, when the SmartCard Interface receives a parity error, it drives the TDA signal low for 1 ETU. It then releases the TDA without actively driving it high. After 5 internal clock cycles (54 MHz), the internal state machine checks the TDA for a start bit (TDA=0). If the board pull-up is not strong enough, the SmartCard Interface will identify the rising slope as a Start Bit. This is interpreted as if it is receiving a data equal to 0xFF with parity equal to 1, which will again generate a new parity error.

2.41.2 Severity

Low.

2.41.3 Workaround

The value of the pull up on the Board should be low enough to drive the TDA high within the requested time.

2.42 SMARTCRD04: SMARTCARD parity error recovery

2.42.1 Summary

In T=0 mode, when a parity error occurs, the SmartCard Interface signals the error to the SmartCard and stops the DMA. The SmartCard is allowed to resume data transfer after 2 ETU. The software must restart the DMA during the transfer of that first character, which means that the software has 12 ETU after the Parity Error Interrupt to re initialize the DMA. Depending on the Baud Rate and Interrupt latency of the Application, this may not be achievable.

2.42.2 Severity

Low.

2.42.3 Workaround

In T=0 mode, when a Parity Error occurs, the application should reset the SmartCard.

2.43 SMARTCRD05: Data may be lost by SMARTCARD interface when switching from reception to transmission mode

2.43.1 Summary

When the SmartCard UART changes from Reception to Transmission mode right after a byte has been received, there is a high chance that the first byte to be sent is lost. Instead of sending the first byte, the UART only sends out a start bit with an invalid pulse width. At 10.5 ETU after the last start bit, the SmartCard Interface notifies the software that a character is available (if no parity error was detected). But the reception process is not finished yet. Indeed, at 12 ETU, the SmartCard Interface must check that the "flow control" functionality is not activated. So when switching from Reception to Transmission Mode, just after received the character (at 10.5 ETU), the Reception is not stopped properly.

2.43.2 Severity

Low.

2.43.3 Workaround

The following two Workarounds have been suggested for this problem.

1. The proper way to deal with that is to wait 12 ETU at least before switching from Reception to Transmission. The internal configurable timers can be used for that purpose. Disadvantage: This increases Driver complexity, because back to back receive and send operations have to be uncoupled by a timer interrupt
2. Set the extra Guard Time parameter to 1 instead of 0. This way, the SmartCard UART delays 1 ETU before sending a byte. This extra delay is enough to prevent the problem. Disadvantage: This decreases the byte transmission throughput with 8%

2.44 PI-bus hangs when reading from non-existing 1394 register in 1394 address space

2.44.1 Summary

If the address register in the 1394 mmio address space does not exist then the 1394 module will not respond with an ACK and the PI-bus will hang, resulting in a total system freeze. Also when the addressed register in 1394 address space is write only, a read from that register will hang the bus. Especially noticeable for speculative loads.

2.44.2 Severity

Low.

2.44.3 Workaround

Two options:

1. When the 1394 is not used in the system, disable the 1394 module by setting it in power-down mode. The 1394 module will not respond to any access and a DEADABBA will be returned when reading from 1394 address space. Writing to 1394 address space will give a bus error, but the bus will not hang.
2. Deny the Trimedia™ access to the 1394 address space by setting the TM_1394_OWNED bit. The PI bus controller will deny the Trimedia™ access to 1394 address space.

Potential problems for these solutions: - when the Trimedia™ is using the 1394 module speculative load can still hang the bus.

2.45 RGB color clamping of MBS

2.45.1 Summary

The graphics plane showed a lighter shade of black than the video plane. Investigation showed that the MBS clamps our RGB graphics data, while it doesn't clamp our YUV video. This difference is quite noticeable if graphics and video are side-by-side.

Turning on clamping for the YUV video does not have any effect.

Turning off clamping on RGB, the difference disappears.

2.45.2 Severity

Low.

2.45.3 Workaround

The end result is as follows:

1. Disabled color clamping for RGB layers
2. Use the latest version of the MBS task definition code, that should set the color clamping settings correctly

2.46 AICP can not mix a pre-multiplied alpha layer in YUV mode

2.46.1 Summary

When the AICP pipe is in YUV, the component values include offsets. The values are unsigned.

When mixing in a layer with alpha, if it's non pre-multiplied, the offsets combine such that they are properly reconstituted after the mix. When a layer is pre-multiplied (the usual form for a graphics layer that is the result of blending other surfaces), the offsets do NOT work out.

The error is worst when the blend is 50%, and amounts to 25% of the incoming component value in that case. The error function is parabolic, zero at full transparent and at full opaque.

2.46.2 Severity

Medium.

2.46.3 Workaround

After a detailed investigation we concluded that there is no hardware work-around to fix this issue. The only way forward is an application solution, as detailed in (DTV.000547).

2.47 VIP window-end of buffer wrap does not work for ANC capture

2.47.1 Summary

The VIP window/end of buffer wrap does not work correctly for ANC capture. With a less-than-optimum input stream (i.e. stream during a channel change or channel tuning), neither the end-of-window nor end-of-buffer wrap condition are met when capturing ANC data (using end-of-buffer-or-end-of-window wrap - end-of-buffer wrap is OK), causing memory corruption.

2.47.2 Severity

Low.

2.47.3 Workaround

A DVP fix has been implemented, released and confirmed as a valid work-around. Internal reference (DS#2394/DTV.000527/DTV.000544).

Investigation details

A captured stream that showed the memory-overflow problem on both the bench hardware set-up and an RTL simulation of the device. This allowed thorough investigation into the problem such that it is now understood why this happens and the fact it's related to a particular undocumented VIP condition used by the original software. The Software workaround is:

1. Declare a memory buffer equal to the size of a field as a shared buffer for ANC data and AUX window capture.
2. Enable non-overlapping AUX window capture in addition to ANC capture using the shared buffer.
3. Designating the end of the AUX window as the wrap-around point.

The modifications made to the software for a workaround (using a different VIP configuration), we have now reviewed against the findings of the HW investigation. It's now understood why this resolves the failure condition observed. No further enhancement or testing is required to the software workaround.

After joint application testing of this Software fix, the original failure condition can no longer be reproduced. This investigation has now been satisfactorily completed.

2.48 VMSP3 clocking not according to spec

2.48.1 Summary

When using the same values to program the clock control register from VMSP1, VMSP2 and VMSP3. VMSP1 and 2 have no problem to decode high bit rate bit streams. VMSP3 can't.

2.48.2 Severity

Low.

2.48.3 Workaround

Software patch in DVP release to invert VMSP3 clock.

2.49 MBS hangs when processing some full planar data

2.49.1 Summary

MBS processing full planar input data can stall if luma and chroma streams are unlocked (de-interlace or 4:2:0 input).

2.49.2 Severity

Low.

2.49.3 Workaround

No software workaround known or implemented. Full planar data not used with DVP2.3 (just semi-planar).

2.50 MMIO Access to MBS register get ignored during task parsing

2.50.1 Summary

The MBS control registers can either be written by external MMIO access or during task list parsing by internal access. Internal access has priority over external. Since the external access is not halted during internal access the external access will be lost. As a result pipelining of tasks into the tasks fifo will not work reliably.

2.50.2 Severity

Low.

2.50.3 Workaround

MBS devlib does not use direct MMIO programming —no issue.

2.51 Corner case hangs VLD

2.51.1 Summary

This bug affects the `dvi_vld` (VLD) block as well as the `dtv_vmpg` (VMPG) block. The VLD sub-block hangs under the following circumstances: While Parsing macroblocks the Mpeg pipe should decode all of the macroblocks within one slice and then consume the entire the prefix of the next start code.

The definition of a start code prefix is more than 23 zero bits followed by a byte aligned 1. There is no limitation on the number of zero bits. The hang seems to occur as a result of an interaction between the consumption of the start code prefix and the input DMA. I see the VLD hang when the input DMA expires while consuming the start code prefix AND there is some erroneous data within the start code prefix very close to the point where the DMA count expired.

Consider the following example scenarios: (`//` = comment)

```
//-----Scenario 1----- //
```

```
Slice Macroblocks (Start Code Prefix) 0x00000101 ..... 0x00000000 0x00000000
0x00000000 // // DMA Ends Here // More Zeroes | Next Slice 0x00000000 .....
0x00000000 0x00000102 0x00000000
```

```
//-----Scenario 2----- // Slice Macroblocks (Start Code Prefix)
0x00000101 ..... 0x00000000 0x00000000 0x00000000 // // DMA Ends Here // More
Zeroes Next Slice | 0x00000000 ..... 0xa5a5a5a5 0x00000102 0x00000000
```

```
//-----Scenario 3----- // Slice Macroblocks (Start Code Prefix)
0x00000101 ..... 0x00000000 0x00000000 0x00000000 // // DMA Ends Here (or
anywhere in the zeroes) // +->More Zeroes Next Slice 0x00000000 ..... 0xa5a5a5a5
0x00000102 0x00000000
```

```
//-----Scenario 4----- // Slice Macroblocks (Start Code Prefix)
0x00000101 ..... 0x00000000 0x00000000 0x00000000 // // DMA En
```

2.51.2 Severity

Low-medium.

2.51.3 Workaround

No software workaround known or implemented, but very low likelihood for this to happen.

2.52 VMPG timeout is useless

2.52.1 Summary

VMPG times out so often, that most of the time it needs to be ignored. For example, if the pipe encounters a long string of zeroes while parsing, the timeout will be erroneously raised. In addition, the resolution of the timeout counter is too low to be useful.

2.52.2 Severity

Low.

2.52.3 Workaround

Hardware limitation. DVP2.3 software does not use the particular timeout feature.

2.53 TM01: TM32 internal bridge does not check data value on ACK of retried write to same address

2.53.1 Summary

The TM3218 core performs a MMIO write as follows:

- it caches the address & value, and sends a retry on T-PI
- it internally performs the write, using the cached address and value
- when a next request comes in with the same address, it acknowledges it irrespective of the value

It is unusual that two masters target the same MMIO register. There are several registers in the TM32 core however where this is legal:

- The IPENDING register - it is the primary mechanism that the core provides for other processors requesting interrupts
- The SEM register - the system's master multi-processor semaphore

The bug can only get triggered if there is more than 1 external master, and not if there is a TM32 and 1 other master.

2.53.2 Severity

Medium - low.

2.53.3 Workaround

None.

2.54 TM02: The aperture set inside Trimedia™ for PCI/XIO space should not have any holes

2.54.1 Summary

The aperture set inside Trimedia™ for PCI/XIO space should not have any holes. Holes in this case are address spaces that are NOT mapped to either a PCI or XIO device. Accesses to XIO holes means the PCI/XIO module will hang the PI buses and accesses to PCI holes means an error acknowledge is returned. If these holes exist in the PCI/XIO aperture set inside the TriMedia™, that means Trimedia™ can do speculative reads to those addresses, therefore causing PI bus hangs or error acknowledges.

2.54.2 Severity

Medium.

2.54.3 Workaround

None.

3. PNX8525 specific limitations

3.1 SPDIF out carrier problem under high loading conditions

3.1.1 Summary

SPDIF out dropouts occur when high CPU load and slightly lower than nominal core voltage is applied. These SPDIF dropouts are believed to be due to power starvation on the SPDIF internal clock line (DDS7).

External clock input proved stable SPDIF output.

Raising core voltage reduces the SPDIF dropouts seen, but does not make the problem disappear.

3.1.2 Severity

Severe.

3.1.3 Workaround

Hardware limitation, no known workaround. Recommend not using SPDIF out on PNX8525. Problem resolved in PNX8526.

3.2 24-bit RGB mode

3.2.1 Summary

If the PNX8525 is put into the 24 Bit RGB mode then the extra RGB bits needed are obtained by reusing some pins for the red bits, e.g.

Red bits [4] to [7] use the I²S pins / signal lines.

These I²S pins have been found not to operate at a fast enough (max) frequency for all the desired color frequencies in the color spectrum.

3.2.2 Severity

Severe (if the 24 bit RGB mode is needed)

3.2.3 Workaround

Use the PNX8526 device as ALL the pins needed for the 24 Bit RGB mode operate at a suitable maximum frequency.

4. Disclaimers

Life support – These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips Semiconductors customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips Semiconductors for any damages resulting from such application.

Right to make changes – Philips Semiconductors reserves the right to make changes in the products - including circuits, standard cells, and/or software - described or contained herein in order to improve design and/or performance. When the product is in full production (status 'Production'), relevant changes will be communicated via a Customer Product/Process Change Notification (CPCN). Philips Semiconductors assumes no

responsibility or liability for the use of any of these products, conveys no licence or title under any patent, copyright, or mask work right to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified.

Application information – Applications that are described herein for any of these products are for illustrative purposes only. Philips Semiconductors make no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

5. Trademarks

TriMedia – is a trademark of TriMedia Technologies Inc.

6. Contents

1	Introduction	3	2.32	JTAG01: synchronization between the TCK and internal clock	13
2	PNX852x generic limitations	3	2.33	USB01: buffer end in TD descriptor should not en with bits "10"	13
2.1	GPIO01: GPIO VIC interrupt	3	2.34	TSDMA01: transfer of multiple of 512 bytes do not happen	13
2.2	GPIO02: GPIO programmed I/O output cannot be used as a GPIO DMA clock	4	2.35	UART01: UART transmitter inserts an extra stop bit	14
2.3	MBS01 VIP01: MBS/VIP truncation of LSBs in non-linear scaling leads to complex math.	4	2.36	MPEG01: ½ HD MPEG mode algorithm in MPEG pipe does not produce acceptable quality ...	14
2.4	VIP02: VIP VBI one line capture is not working	4	2.37	UART02: Viper Uart glitches TX signal between stop bits when set for 8 bit, parity, 2 stop bits	14
2.5	MBS02: MBS 3-field majority select de-interlacing does not work	4	2.38	AICP03: AICP 8 bit illegal code limiter does not work	15
2.6	GLOBAL01: GLBREG2 Read DEADABBA from T-PI modules	5	2.39	PCIXIO12: PCI burst lengths exceeding 128 bytes are not supported	15
2.7	PCIXIO01: PCI external master reads swaps MMIO registers in big endian mode	5	2.40	PCIXIO13: PCI : DMA Transfers Across PCI bridges are not supported	15
2.8	PCIXIO02: PCI arbiter setting	5	2.41	SMARTCRD03: SMARTCARD generates infinite number of parity errors when acknowledging first parity error	16
2.9	PCIXIO03: XIO profile conflict	6	2.42	SMARTCRD04: SMARTCARD parity error recovery	16
2.10	PCIXIO04: PCI read lifetime timeout	6	2.43	SMARTCRD05: Data may be lost by SMARTCARD interface when switching from reception to transmission mode	17
2.11	PCIXIO05: PCI: multi-word read after retry ...	6	2.44	PI-bus hangs when reading from non-existing 1394 register in 1394 address space	17
2.12	PCIXIO06: PCI issued transaction may fail ...	7	2.45	RGB color clamping of MBS	18
2.13	PCIXIO07: Direct PI access of XIO with opcodes larger than WDU does not work	7	2.46	AICP can not mix a pre-multiplied alpha layer in YUV mode	18
2.14	PCIXIO08: Nand Flash DMA usually limited to 512 bytes	7	2.47	VIP window-end of buffer wrap does not work for ANC capture	19
2.15	PCIXIO09: GPXIO read data might be lost after a direct XIO read	8	2.48	VMSP3 clocking not according to spec	19
2.16	PCIXIO10: XIO accesses should be restricted to valid areas within the XIO aperture	8	2.49	MBS hangs when processing some full planar data	20
2.17	PCIXIO11: PCI internal arbiter is not working .	8	2.50	MMIO Access to MBS register get ignored during task parsing	20
2.18	MMI01: SDRAM self refresh broken	9	2.51	Corner case hangs VLD	20
2.19	AICP01: AICP KeyFwd feature not working ...	9	2.52	VMPG timeout is useless	21
2.20	AICP02: multiple signatures for AICP signature generating mechanism	9	2.53	TM01: TM32 internal bridge does not check data value on ACK of retried write to same address	21
2.21	VMSP01: VMSP queue lock; does not output data if head ptr set to beginning of buffer and limit set to full after the first lock	10	2.54	TM02: The aperture set inside Trimedia™ for PCI/XIO space should not have any holes ..	22
2.22	VMSP02: serial transport in not functional ...	10	3	PNX8525 specific limitations	22
2.23	VMSP03: VMSP GP packet insertion mode ...	10	3.1	SPDIF out carrier problem under high loading conditions	22
2.24	PI01: PiMi bridge burst transfers issue	10	3.2	24-bit RGB mode	23
2.25	139401: 1394 SFTRST bit functionality	11			
2.26	139402: 1394 endianness issue	11			
2.27	139403: 1394 EN_LDTR bit functionality ...	11			
2.28	139404: 1394 ping response problem	12			
2.29	SMARTCRD01: SmartCard interrupt clearing does not work if pi-clk > clk_smcard	12			
2.30	SMARTCRD02: SmartCard DMA autoflush not working	12			
2.31	2D01: optimizations for 2D missing	12			

continued >>

4	Disclaimers.....	24
5	Trademarks.....	24



© Koninklijke Philips Electronics N.V. 2004

All rights are reserved. Reproduction in whole or in part is prohibited without the prior written consent of the copyright owner. The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent- or other industrial or intellectual property rights.

Date of release: 5 November 2004
Document number: AN10308

Published in The Netherlands