Semiconductors

AN10310

Transfer notes - PNX8525 (Viper 1.0) to PNX8526 (Viper 1.1)

Rev. 01 – 5 November 2004 Application note

Document information

Info	Content
Keywords	PNX8526, PNX8525, DVP2.x, PLL
Abstract	This application note describes the differences and any changes required when replacing PNX8525 with PNX8526.

PHILIPS

Revision history

Rev	Date	Description
01	20041105	Initial version (AN10310)

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 application note describes the differences and any changes required when replacing PNX8525 with PNX8526.

2. Primary differences

2.1 Hardware

Core Power Rail

PNX8525 Vc = 1.8 V

PNX8526 Vc = 1.2 V

2.2 Software

DVP2 x versions

PNX8525 - all DVP releases

PNX8526 - from DVP2.3 B5.1 onwards

2.3 Application

The changes in PNX8526 (Viper1.1) involve the following changes to PLL settings:

Table 1: Changes to PLL settings

Software difference	PNX8525 (Viper1.0)	PNX8526 M0A (Viper1.1 M0A)	PNX8526 M0B (Viper1.1 M0B)	PNX8526 M0C (Viper1.1 M0C)
PLL 0-5 Equations	Fout=Fin(N+2)/(M+2)*f(P)	Fout =Fin*N/M*f(P)	Fout =Fin*N/M*f(P)	Fout =Fin*N/M*f(P)
PLL 4-5 Requires higher P Value	No	Yes	Yes	Yes
D2D Clock frequency settings	0 = 86.4 MHz 1 = 108Mhz	0 =108 MHz 1 = 86.4 MHz	0 = 86.4 MHz 1 = 108 MHz	0 = 86.4 MHz 1 = 108 MHz

3. Explanation of PLL 0-5 equations

PLL 0 to 5 have different equations for Viper1.1 and Viper1.0:

- Viper1.0 Fout = Fin(N+2)/(M+2)*f(P)
- Viper1.1 Fout = Fin N/M*f(P)

The function of the p-value is different between Viper1.0 and Viper1.1

Table 2: Function of P values

P	PNX8525 (Viper1.0) f(P)	PNX8526 (Viper1.1) f(P)
0	1	1
1	2	2
2	2	4
3	4	8

This affects settings for bits 24:16, 12:8, and 3:2 for registers:

PLL0_CTL (0x047000)

PLL1_CTL (0x047004)

PLL2_CTL (0x047008)

PLL3_CTL (0x04700C)

PLL4_CTL (0x047010)

PLL5_CTL (0x047014)

4. Explanation of PLL4-5 P value requirements

In addition the M0A/B/C Fixes, remove a hardware divide by 2 for PLL4 and 5; this can always be compensated for by using a higher f(P) value than was used on Viper1.0.

A working example is:

PNX8525 (Viper1.0)

PLL4 = 0x001e020c = = > P = 3 [f(P)= 4 see Table 2], M = 2, N = 30

Fout = 27 MHz * (N+2) / (M+2)*f(P)

Fout = 27 MHz * (30+2) / (2+2)*4

Fout = 27 MHz * 32 / 4*4

Fout = 864 MHz / 16= 54 MHz / 2 = 27 MHz

Remark: The 54 MHz / 2 is the hardware divide by two which is only present on PNX8525 and has been removed on the PNX8526 M0A, M0B & M0C.

PNX8525 (Viper1.0)

PLL4 = 0x0020040c = = > P = 3 [f(P)= 8 see Table 2], M = 4, N = 32

Fout = 27 MHz * N / M*f(P)

Fout = 27 MHz * 33 / 4*8

Fout = 27 MHz * 32 / 32

Fout = 864 MHz / 32= 27 MHz

5. Explanation of D2D clock frequency settings

The register CLK_D2D_CTL(0x04721C) bit 1 states 0 = 1.728 GHz divide by 20 and 1=1.728 GHz divide by 16.

An initial error in the CAB block meant this is implemented as:

PNX85250 = 1.728 GHz divide by 20 and 1 = 1.728 GHz divide by 16

PNX8526 M0A0 = 1.728 GHz divide by 16 and 1=1.728 GHz divide by 20

PNX8526 M0B0 = 1.728 GHz divide by 20 and 1=1.728 GHz divide by 16

6. Adjusting of PLL M,N & P values

A simple way to calculate PLL changed values is:

Add 2 to the N value

Add 2 to the M Value

For P or f(P) value see Table 2.

7. Application pointers

7.1 Change PLL values in the bootscript

e.g. the bootscript would look like:

Table 3: PNX8525 (Viper1.0) typical bootscript values
e.g. PLL0 = 137.7 MHz

PLL (Address)	Value
PLL0 (0x1be47000)	0x00310304
PLL1 (0x1be47004)	0x00310304
PLL2 (0x1be47008)	0x00310304
PLL3 (0x1be4700c)	0x002a0304
PLL4 (0x1be47010)	0x001e020c
PLL5 (0x1be47014)	0x001e020c

Table 4: PNX8526 (Viper1.1) typical bootscript values
e.g.PLL0 = 137.7 MHz

PLL (Address)	Value
PLL0 (0x1be47000)	0x00330504
PLL1 (0x1be47004)	0x00330504
PLL2 (0x1be47008)	0x00330504
PLL3 (0x1be4700c)	0x002c0504
PLL4 (0x1be47010)	0x0020040c
PLL5 (0x1be47014)	0x0020040c

7.2 Checking the PLL values

It is possible in the HyperTerminal window to check what values have been loaded into PLL0 to PLL5, but this does rely on the PNX8525/6 devices booting up and running the BTM. It is also possible to check these PLL register values by using an EJTAG debug probe, like a Vision Probe or BugTracer and as such this method doesn't require the BTM to be running. However, both methods do require that the BootScript to have been successfully loaded.

Therefore:

- To read & write to the PLL values from the HyperTerminal use the 'e' => edit and 'd' => dump commands.
- When reading and writing to the PLL's use the Virtual Address, so for PLL0 this translates to PLL00x00047000 => 0x0bbe47000 e.g. add 0xbbexxxxx.
- To get a dump of the PLL's values in the HyperTerminal use d bbe47000.
- To edit the PLL's in the HyperTerminal use e bbe47000 00330504 /d.

A working example is:

(C) Copyright 1995-2001 Philips Semiconductors. Boot Manager running on Philips STB85000 ASTB Platform. Board id:0x00521131, CPU Running BE mode, 64MB detected. Boot Manager Ui Shell, Build Mar 21 2002,16:35:05.

```
>e
e        Edit memory.
Syntax : e HexAddress HexVal [/b|/w|/d]
        /b      Edit memory with 8 bit value.
        /w      Edit memory with 16 bit value.
        /d      Edit memory with 32 bit value.

>d bbe47000
0xBBE47000      - 00330504 00330504  00330504 002C0504
0xBBE47010      - 0020040C 0020040C  00000000 DEADABBA

>e bbe47000 00320504 /d
>d bbe47000
0xBBE47000      - 00320504 00330504  00330504 002C0504
0xBBE47010      - 0020040C 0020040C  00000000 DEADABBA
>
```

8. 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.

9. Contents

Date of release: 5 November 2004
Document number: AN10310

Published in The Netherlands