Marvell Solutions

Marvell® PXA270M Processor

Specification Update

Doc. No. MV-S900957-00 Rev. I, PUBLIC RELEASE
April 19, 2010

MOVING FORWARD FASTER®

MARVELL®

Document Conventions

Note

Provides related information or information of special importance.

Caution

Indicates potential damage to hardware or software, or loss of data.

Warning

Indicates a risk of personal injury.

Document Status	
Doc Status: Rev H Public Release 4-19-10	Technical Publication: 0.xx

For more information, visit our website at: www.marvell.com

Revision History

Revision Date	Version	Description
April 2010	Rev I	Added Errata 95
November 2009	Rev H	Added Specification Clarification S1.
May 2009	Rev G	Re-wrote the description for E94 for greater clarity
April 2009	Rev F	Added Errata E94
February 2009	Rev E	Errata E16 fixed; Added Documentation Change D48; Updated Errata E59
January 2009	Rev D	Added Documentation Change D47
September 2007	012	Added Errata 93 Added Documentation Change 46
September 2007	011	Added Errata 92
July 2006	010	Removed Specification Change 1, 2, 3, 4, 6, 7 and Documentation Change 1, 2, 4, 5, 6, 7, 8, 9, 10, 12, 14, 17, 18, 19, 20, 21, 23, 24, 25, 26, 28, 29, 30, 31, 32 because they have been fixed in the *PXA27x Processor Family Developer's Manual*, January 2006, Revision 003. Added Errata 88, 89, 90, 91 Updated Errata 14, 61 Added Specification Change 13, 14, 15, 16 Updated Specification Change 11 Added Documentation Change 38, 39, 40, 41, 42, 43, 44, 45
December 2005	009	Added Errata 83, 83, 84, 85, 86, 87 Updated Errata 21 Updated Specification Change 3 Added Documentation Change 34, 35, 36, 37
October 2005	008	Added Errata 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82 Added Specification Change 8, 9, 10, 11, 12 Added Documentation Change 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33
July 2005	007	Removed Specification Change 5 and Documentation Change 3, 11, 15, 16 because they have been fixed in the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification*, Revision 004 and *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*, Revision 004. Removed Documentation Change 13 because it has been fixed in the PXA27x Processor Family Design Guide, Revision 002. Added Errata 63, 64, 65, 66, 67, 68, 69, 70, 71 Added Specification Change 6, 7 Updated Specification Change 3 Added Documentation Change 17, 18, 19, 20, 21 Updated Documentation Change 7 Added C0 and C5 columns in Table 4, "Summary of Specification Changes"

Revision Date	Version	Description
April 2005	006	Updated PXA27x Processor Family Package Markings Figure 1 Updated Table 2 Updated Errata 37,60 Added Errata 61, 62 Updated Specification Change 1,2 Added Specification Change 5 Updated Documentation Change 6,11 Added Documentation Change 12-16
February 2005	005	Added Errata 60 Added Documentation Changes 9-11 Updated Documentation Changes 1-8
January 2005	004	Added Errata 55, 56, 57, 58, 59 Updated Errata 17, 25, 26, 29, 30, 40, 42, 43, 45, 46 Added Documentation Change 1, 2, 4, 5, 6, 7, 8 Added Specification Change 1, 2, 3, 4
October 2004	003	Added Errata 51, 52, 53, 54
August 2004	002	Added Errata 48, 49, 50 Added Documentation Changes 5 - 15
April 2004	001	Initial publication

Preface

This document contains updates to the specifications for the PXA27x Processor Family, listed in Table 1. This document is a compilation of device and documentation errata, specification clarifications, and specification changes. It is intended for hardware system manufacturers and software developers of applications, operating systems, and tools.

Marvell Corporation has endeavored to include all documented errata in the consolidation process. However, Marvell makes no representations or warranties concerning the completeness of the *PXA27x Processor Family Specification Update*.

Information types defined in Nomenclature are consolidated into the *PXA27x Processor Family Specification Update* and are no longer published in other documents.

This document might also contain information that was not previously published.

Affected and Related Documents

Table 1 lists the documents affected by and related to this errata update.

Contact a Marvell representative to obtain the latest revisions of these documents.

Table 1. Affected Documents / Related Documents

Title
PXA27x Processor Family Developer's Manual
PXA27x Processor Family Design Guide
PXA27x Processor Family Electrical, Mechanical, and Thermal Specification
PXA270 Processor Family Electrical, Mechanical, and Thermal Specification

Nomenclature

Errata are design defects or errors. These errata might cause behavior of the PXA27x Processor Family to deviate from published specifications. Hardware and software designed to be used with any given processor stepping must assume that all errata documented for that stepping are present on all devices unless otherwise noted.

Documentation changes include typos, errors, and omissions from the current published specifications. These changes will be incorporated in the next release of the document.

Specification clarifications describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications will be incorporated in any new release of the document.

Specification changes are modifications to the current published specifications. These changes will be incorporated in the next release of the document.

Note: Errata remain in the specification update throughout the product's life cycle, or until a particular stepping is no longer commercially available. Under these circumstances, errata removed from the specification update are archived and made available upon request. Specification changes, specification clarifications, and documentation changes are removed from the specification update when the appropriate changes are made to the appropriate product specification or user documentation (data sheets, manuals, and so forth).

PXA27x Processor Family Package Markings

The following figure depicts the location, on specific PXA27x Processor Family packages, where the actual markings are located.

Figure 1.PXA27x Processor Family Package Markings Locations

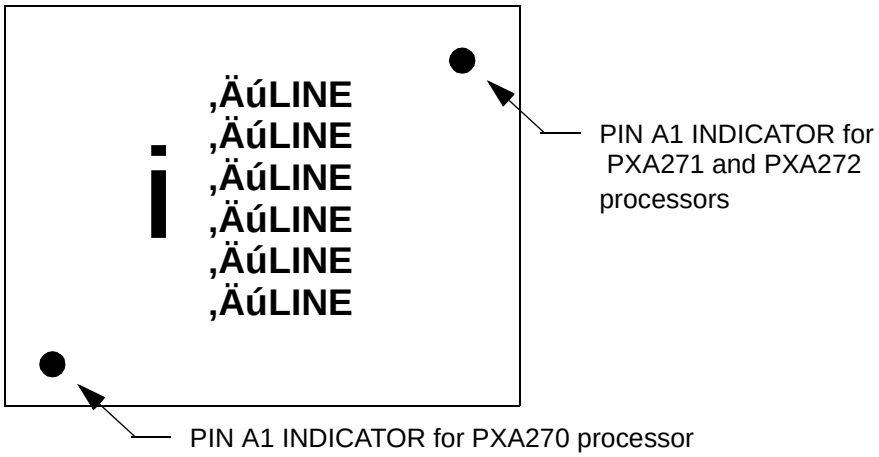


Table 2 summarizes the processor packages.

Note: This table is for example only. It must not be used to determine final production offerings.

Table 2. Processor Package Summary

Product	PXA270	PXA270	PXA271	PXA272
Discrete / MCP	Discrete	Discrete	MCP	MCP
Package	13x13 VFBGA	23x23 PBGA	14x14 FSCSP	14x14 FSCSP
Flash	0 Mb	0 Mb	1x 256L Tyax	2x 256L Tyax
SDRAM	0 Mb	0 Mb	1x 256 Mb	0 Mb
Maximum Frequency	624 MHz	520 MHz	416 MHz	520 MHz
Lead / Lead-Free	Both	Both	Leaded	Both
Temp Range	Commercial	Commercial / Extended	Commercial	Commercial
Stepping	C5	C5	C5	C5

Summary of Changes

The following tables summarize the errata, specification changes, specification clarifications, and documentation changes that apply to the PXA27x Processor Family. Marvell might fix some of the errata in a future stepping of the component and account for the other outstanding issues through documentation or specification changes as noted. These tables use the following notations:

Notation	Meaning
X	This erratum exists in the stepping indicated. Specification change or clarification that applies to this stepping.
(No mark) or (Blank Box)	This erratum is fixed in the listed stepping, or the specification change does not apply to the listed stepping.
Plan Fix	This erratum might be fixed in a future stepping of the product.
Fixed	This erratum has been previously fixed.
No Fix	There are no plans to fix this erratum.
Doc	Marvell plans to update the appropriate documentation in a future revision.
No Bug	This erratum has been determined to be a false erratum.
Shaded	This item is either new or modified from the previous version of the document.

Table 3. Summary of Errata (Sheet 1 of 5)

References			C0	C5	PXA270M A2	Status
Number	Title	Page				
1	“CORE: IFU misses an external abort when a lock command is outstanding.”	16	X	X	X	No Fix
2	“CORE: Aborted store that hits D-cache might mark write-back data as dirty.”	16	X	X	X	No Fix
3	“CORE: Performance monitor unit counts, using performance monitoring event number 0x1, can be incremented erroneously by unrelated core events.”	17	X	X	X	No Fix
4	“CORE: In SDS mode, back-to-back memory operations where the first instruction aborts might hang.”	17	X	X	X	No Fix
5	“CORE: Lock aborts resulting from I-cache or I-TLB lock operations are not presented properly on the trace interface.”	18	X	X	X	No Fix
6	“CORE: CP15 ID register accesses with opcode2 > 0b001 return unpredictable values.”	18	X	X	X	No Fix
7	“CORE: Disabling and re-enabling the MMU can hang the core or cause it to execute the wrong code.”	18	X	X	X	No Fix
8	“CORE: JTAG parallel register updates require an extra TCK rising edge.”	19	X	X	X	No Fix
9	“MMC: SPI mode – if card is deselected, PROG_DONE will not be set.”	19	X	X	X	No Fix
10	“MEMC: No MRS command is given when exiting from alternate bus master mode when SA-1111 address muxing mode is enabled.”	19	X	X	X	No Fix
11	“KBD: Extra keypad matrix interrupt in IMKP mode.”	20	X	X	X	No Fix

Table 3. Summary of Errata (Sheet 2 of 5)

References			C0	C5	PXA270M A2	Status
Number	Title	Page				
12	“USBH: USBH register UHCRHPSx[CCS] bit set incorrectly after power on”	20	X	X	X	No Fix
13	“AC97: Command-done bit remains set after an AC97 cold reset.”	20	X	X	X	No Fix
14	“POWER MANAGER: Fast ramp rates on voltage pins can cause high current consumption.”	21	X	X	X	No Fix
15	“LCD: Reconfiguring the LCD controller retains the previous PPL value for the first line.”	22	X	X	X	No Fix
16	“LCD: Overlay1 is not enabled intermittently after re-enabling LCD.”	22	X			Fixed
17	“POWER MANAGER: Processor ignores BATT/VCC faults while exiting sleep mode.”	23	X	X	X	No Fix
18	“KBD: Keyboard Edge-Detect Status Register Incorrect After Standby Mode Wakeup.”	23	X	X	X	No Fix
19	“UART: Character Timeout interrupt remains set under certain software conditions”	24	X	X	X	No Fix
20	“LCD: LCD not enabling in dual panel mode.”	24	X	X	X	No Fix
21	“UDC: UDC does not correctly support alternate interfaces.”	25	X	X	X	No Fix
22	“ICP: Receiver Aborts randomly occur prematurely and without End of frame/Error in FIFO interrupt”	26	X	X	X	No Fix
23	“SSP: OSTimer counter increments incorrectly for SSP Frames in Network mode”	26	X	X	X	No Fix
24	“LCD: Enabling Overlay 2 for YUV420 hangs LCD controller.”	26	X			Fixed
25	“USBOTG: Unable to measure duration of Single-Ended Zero (SE0) for Session Request Protocol (SRP)”	28	X			Fixed
26	“MEMC: Write/Read to/from SDRAM can collide with alternate bus master mode when MDREFR:ALTREFB is set.”	28	X	X	X	No Fix
27	“POWER MANAGER: Core hangs during voltage change when there are outstanding transactions on the bus”	28	X	X	X	No Fix
28	“MMC: MMC unit in SPI mode always waits a minimum of 1 Ncx cycles, even though the MMC spec dictates that SPI mode CMD9 can have a minimum of 0 Ncx cycles.”	30	X			Fixed
29	“SD: SD Controller in SPI mode not receiving data response for CMD9 and CMD10 from some SD Cards”	31	X			Fixed
30	“MEMC: SDRAM Refresh Commands are issued too often during a VLIO access while BREQ is asserted.”	31	X	X	X	No Fix
31	“INTERRUPT CONTROLLER: Unexpected exception vector when ICCR[DIM]=0 and ICMR=0.”	31	X	X	X	No Fix
32	“SSP: TXD line does not tristate when SSP is Slave to Frame”	32	X	X	X	No Fix
33	“POWER MANAGER: Simultaneous BATT and VDD faults results in going to DeepSleep mode twice.”	33	X	X	X	No Fix
34	“CORE: Non-branch instruction in vector table may execute twice after a thumb mode exception”	33	X	X	X	No Fix
35	“UART: UART does not correctly indicate a Framing Error Interrupt in DMA mode.”	33	X	X	X	No Fix
36	“CLOCKS: System Hangs when enabling RUN/TURBO switching at 520 MHz”	34	X	X	X	No Fix
37	“CLOCKS: System Hangs when enabling HalfTurbo Switching”	35	X	X	X	No Fix

Table 3. Summary of Errata (Sheet 3 of 5)

References			C0	C5	PXA270M A2	Status
Number	Title	Page				
38	“MEMC: Memory Controller hangs when entering Self Refresh Mode.”	35	X	X	X	No Fix
39	“SDIO: SDIO Devices Not Working at 19.5 Mbps”	36	X			Fixed
40	“AC97: Command Done bit is never set when data is written in slot12”	36	X	X	X	No Fix
41	“SD/MMC: SD/MMC controller CRC errors with some SD/MMC cards”	37	X			Fixed
42	“USBH: There is no Individual Power Sense Polarity bit for each Host Port. The Power Sense Polarity bit controls the polarity for all three Host Ports.”	37	X			Fixed
43	“KBD: Keypress wakeup from Standby mode is not reliable”	38	X	X	X	No Fix
44	“USBH: USB Host Port 3 in Transceiverless Mode may not work correctly with an external device.”	38	X			Fixed
45	“SD/MMC: SPI mode commands fail even on cards that are compatible with SPI spec 1.0”	38	X			Fixed
46	“CLOCKS AND POWER: PWM Clock Enables do not work as specified”	39	X	X	X	No Fix
47	“ICP: Occasionally EIF, EOF and CRC interrupt are missed when a CRC error is received”	39	X	X	X	No Fix
48	“POWER MANAGER: Batt Fault does not always re-enable GPIO 0 and GPIO 1 as wake-up sources.”	40	X	X	X	No Fix
49	“POWER MANAGER: The processor does not exit from sleep/deep-sleep mode.”	40	X			Fixed
50	“SDIO: CMD53 multiple-block data transfer with block count set to 0 not supported.”	41	X	X	X	No Fix
51	“LCDC: Disable-done interrupt does not always occur.”	41	X	X	X	No Fix
52	“UDC: RCV can not be tied high during UDC transmission when using an external transceiver.”	41	X	X	X	No Fix
53	“AC97: AC97 CAR[CAIP] bit field can be incorrectly set”	42	X	X	X	No Fix
54	“SD/MMC: SD Command 56 Read Error”	43	X			Fixed
55	“AC97: AC97 Unit Incorrectly Receives An EOC”	43	X			Fixed
56	“MMC: MMC Write CRC Response Error”	44		X	X	No Fix
57	“USB: USB Host Port 3 Transceiverless Mode Restrictions”	45		X	X	No Fix
58	“MMC/SD/SDIO: Read Data Command May Hang The Controller”	45	X	X	X	No Fix
59	“VCORE: Voltage Sensitivity at VVCCC(2,4) May Result in Undetermined System Behavior”	46	X	X	X	Fix
60	“RTC: Stopwatch SWCR alarm inaccuracy”	47	X	X	X	No Fix
61	“MEMC: Memory Controller may hang while clearing MDREFR[K1DB2] or MDREFR[K2DB2]”	48	X	X	X	No Fix
62	“POWER MANAGER: The processor can not execute the nBATT_FAULT or nVDD_FAULT xIDAE abort handler if a fault occurs while in sleep mode.”	49	X	X	X	No Fix
63	“POWER MANAGER: nBATT_FAULT and nVDD_FAULT during sleep mode do not prevent RTC wakeup.”	49	X	X	X	No Fix
64	“POWER MANAGER: Voltage change coupled with power mode change causes the processor to be unable to wake from sleep mode.”	50	X	X	X	No Fix
65	“SSP: GPIO[28] and GPIO[29] may not behave normally in SSP master mode.”	50	X	X	X	No Fix
66	“SSP: SSP may hang when switching from slave mode to master mode.”	50	X	X	X	No Fix

Table 3. Summary of Errata (Sheet 4 of 5)

References			C0	C5	PXA270M A2	Status
Number	Title	Page				
67	“POWER MANAGER: Pins selected in the Keyboard Wake-Up Enable Register (PKWR) may not cause the processor to wake up from sleep mode.”	51	X	X	X	No Fix
68	“UHC: Exiting from sleep/deep sleep mode may cause unexpected USBH2 interrupt.”	51	X	X	X	No Fix
69	“UDC: USB OTG ID pin weak pullup draws 8 mA.”	51	X	X	X	No Fix
70	“CLOCKS: 48 MHz GPIO output stops during a USB suspend operation or if the USB client is not enabled.”	52	X			Fixed
71	“POWER MANAGER: 8uS SYS_EN deassertion during watchdog reset.”	52	X	X	X	No Fix
72	“VCC_USB must be 3.0 V or greater to pass USB signal quality compliance tests.”	52	X	X	X	No Fix
73	“MultiMediaCard Controller: MMCMD line is sensitive to capacitive loading.”	53		X	X	No Fix
74	“UART: Baud rate may not be programmed correctly on back-to-back writes.”	53	X	X	X	No Fix
75	“AC97: PCM transmit FIFO and receive FIFO may not set the FIFO error status bits.”	53	X	X	X	No Fix
76	“UDC: Writing the UDCCSR0[OPC] bit during the status stage can corrupt the next transaction.”	53	X	X	X	No Fix
77	“CI: Quick Capture Interface captures data before line valid signal occurs.”	54	X	X	X	No Fix
78	“MSL: Unexpected BBFREQ operation in 13M mode.”	54	X	X	X	No Fix
79	“KBD: Keypad interrupt missing after wakeup from Standby mode.”	54	X	X	X	No Fix
80	“CLOCKS MANAGER: Possible DMA errors when exiting 13M mode to a frequency where the memory controller clock (CLK_MEM) frequency does not equal the system bus frequency.”	55	X	X	X	No Fix
81	“GPIO: GPIO glitch during power up”	55	X	X	X	No Fix
82	“I^2C: I^2C may hang if an external device holds SCL low without first deasserting SDA.”	55	X	X	X	No Fix
83	“SSP: SSP1 generates a spurious Frame when switching from Frame Slave to Frame Master.”	56	X	X	X	No Fix
84	“MEMC: Data abort on access to disabled SDRAM.”	56	X	X	X	No Fix
85	“UART: TX data corruption when filling an empty FIFO.”	56	X	X	X	No Fix
86	“POWER MANAGER: GPIO83 wake-up is level-triggered, not edge-triggered.”	58	X	X	X	No Fix
87	“SD/MMC: Streaming write (command 20) and streaming read (command 11) transfer one block.”	58	X	X	X	No Fix
88	“SDRAM read and write errors while modifying MDREFR[KxDB2] if DMA is running”	59	X	X	X	No Fix
89	“Limitations on combining core voltage changes with core frequency changes”	60	X	X	X	No Fix
90	“MMC controller sending first FIFO byte twice”	60	X	X	X	No Fix
91	“UART: TX interrupt can be missed when running full duplex”	60	X	X	X	No Fix
92	Caches, TLB (Translation Lookaside Buffer), and BTB (Branch Target Buffer) may become corrupted in Standby mode	60	X	X	X	No Fix

Table 3. Summary of Errata (Sheet 5 of 5)

References			C0	C5	PXA270M A2	Status
Number	Title	Page				
93	USB Host Controller Registers are not reset to their default values after sleep/deep-sleep wakeup.	61	X	X	X	No Fix
94	“USB OTG over-current pin can cause erroneous shutdown in the USB Host.”	62	X	X	X	No Fix
95	“Rapid power cycling may cause a processor hang”	62	X	X	X	No Fix

Table 4. Summary of Specification Changes

Number	Title	Page	C0	C5	Status
8	“Extended Input DC operating conditions for USB interfaces”	63	X	X	Doc
9	“32.768-kHz crystal load capacitance specification updated”	64	X	X	Doc
10	“SDRAM 3.3 V Interface AC Timing Specification Updated”	64	X	X	Doc
11	“Synchronous Flash AC Timing Specification Updated”	66	X	X	Doc
12	“Quick Capture Interface AC Timing Added”	68	X	X	Doc
13	“SD/MMC: Remove section 15.8.4.5 “Stop Data Transmission, Randomly””	69	X	X	Doc
14	“32.768-kHz crystal equivalent series resistance requirements correction”	69	X	X	Doc
15	“LCD controller timing specification updated”	69	X	X	Doc
16	“PXA27x package information updated”	71	X	X	Doc

Table 5. Summary of Specification Clarifications

Number	Document Revision	Page	Status	Specification Clarifications
S1	Spec Update S900957-00	74	Doc	**I/O pin drive strengths and internal resistive pullup/pulldown values may vary**

Table 6. Summary of Documentation Changes (Sheet 1 of 2)

Number	Document Revision	Page	Status	Documentation Changes
22	EMTS 28000304 EMTS 28000205	75	Doc	**“SDRAM Timing Parameter tsdCL Description Correction”**
27	EMTS 28000304 EMTS 28000205	75	Doc	**“VLIO Timing Parameter tvlioDH Correction”**
33	EMTS 28000304 EMTS 28000205	76	Doc	**“Derating Specifications Removed From EMTS”**
34	EMTS 28000205	77	Doc	**“Ball Diameter for 23x23mm Package Added”**
35	Developers Manual 2800003	77	Doc	**“UART Maximum Baud Rate Clarification”**
36	EMTS 28000304 EMTS 28000205	77	Doc	**“MMC/SD/SDIO AC Timings Added”**

Table 6. Summary of Documentation Changes (Sheet 2 of 2)

Number	Document Revision	Page	Status	Documentation Changes
37	EMTS 28000205	79	Doc	"Idle and Low-Power Mode Maximum Power-Consumption Specifications Added"
38	Developers Manual 2800003	81	Doc	"MMC Read Time-Out Register (MMC_RDTO) Description Clarification"
39	EMTS 28000304 EMTS 28000205	81	Doc	"GPIO reset nRESET_OUT and GPIO<1> pulse width correction"
40	EMTS 28000304 EMTS 28000205	83	Doc	"nBATT_FAULT and nRESET power-on reset description correction"
41	EMTS 28000304 EMTS 28000205	83	Doc	"ROM timing parameters tromAVDVF and tromAVDVS correction"
42	Developers Manual 2800003	84	Doc	"Keypad Interface external pull-down resistors specifications added"
43	Developers Manual 2800003	85	Doc	"LCD controller output buffer strength description added"
44	Developers Manual 2800003	86	Doc	"SD/MMC CMD12 no responses handling description added"
45	Developers Manual 2800003	86	Doc	"USB host Port 2 USBHPWR<2> and USBHPEN<2> clarification"
46	Developers Manual 2800003	86	Doc	COPROCESSOR: New CPU ID and JTAG ID Values
47	Specification Update	86	Doc	Incorrect Ball Diameter Listed in S16 for PXA27x Package Information
48	Specification Update	87	Doc	Removed last remaining mention (E16) of Memory Stick from PXA270M Spec Update (not supported in 270M)

Errata

E1. CORE: IFU misses an external abort when a lock command is outstanding.

Problem: A bus abort occurs on a code fetch while an I-TLB lock mcr is outstanding. The IFU fails to abort, instead executing the instruction returned on the aborting transaction. Parity errors are not affected. The bus abort might be due to either an ABORT pin assertion or a multi-bit ECC error on the core. On the core, the bus abort can occur as the result of a master abort, a target abort, or a single data phase disconnect on the system bus.

Implication: TBD

Workaround: Branch flush after every ITLB or I-cache lock.

Status: No Fix

E2. CORE: Aborted store that hits D-cache might mark write-back data as dirty.

Problem: If an aborted store hits clean data in the data cache (data in an aligned four-word range that has not been modified from the core since it was last loaded in from memory or cleaned), the data in the array will not be modified (the store will be blocked), but the dirty bit will be set.

When that line is then aged out of the data cache or explicitly cleaned, the data in that four-word range will be evicted to external memory, even though it has never been changed. In normal operation, this will be nothing more than an extra store on the bus that writes the same data to memory as is already there.

This problem might be visible at the following boundary condition:

1. A cache line is loaded into the cache at address A
2. Another master externally modifies address A
3. An Intel XScale®[1] core store instruction attempts to modify A, hits the cache, aborts because of MMU permissions, and is backed out the of cache. That line should not be marked “dirty”, but because of this defect it will be.
4. The cache line at A then ages out or is explicitly cleaned. The original data from location A will be evicted to external memory, overwriting the data written by the external master.

This situation happens only if software is allowing an external master to modify memory that is write-back or write-allocate in the XScale page tables, and depending on the fact that the data is not dirty in the XScale cache to preclude the cached version from overwriting the external memory version. If there are any semaphores or any other handshaking to prevent collisions on shared memory, this should not be a problem.

Implication: TBD

1. Intel XScale is a trademark or registered trademark of Intel Corporation and its subsidiaries in the United States and other countries.

Workaround: For a shared memory region, mark it as write-through memory in the XScale page table to prevent the data from ever being written out as dirty, so the defect does not appear. Alternatively, ensure that any cached copy of the data in external memory is invalidated if an external agent modifies the external copy.

Status: No Fix

E3. CORE: Performance monitor unit counts, using performance monitoring event number 0x1, can be incremented erroneously by unrelated core events.

Problem: The performance monitor unit can be used to count cycles in which the I-cache cannot deliver an instruction. Performance monitoring event number 0x1 is used for this. According to EAS text, the only cycles counted should be those due to an I-cache miss or an I-TLB miss. The following unrelated events in the core also cause the corresponding count to increment when event number 0x1 is being monitored:

- Any architectural event (e.g. IRQ, data abort)
- mstr instructions that alter the CPSR control bits
- Some branch instructions, including indirect branches and those mis-predicted by the BTB
- CP15 mcr instructions to registers 7, 8, 9, or 10 which involve the I-cache or the I-TLB

Each of the items above might cause the performance monitoring count to increment several times. The resulting performance monitoring count might be higher than expected if the above items occur, but never lower. The performance monitor unit uses ic_instValid_qf2h. This signal asserts not just at the proper time, but also due to the events described above.

Implication: TBD

Workaround: There is no way to obtain the correct number of cycles stalled due to I-cache misses and I-TLB misses. One component of the unwanted noise may be filtered out: extra counts due to branch instructions mis-predicted by the BTB. The number of mispredicted branches can also be monitored using performance monitoring event 0x6 during the same time period as event 0x1. The mispredicted branch number can then be subtracted from the I-cache stall number generated by the performance monitor to get a value closer to the correct one. Depending on the nature of the code monitored, this workaround might have limited value.

Status: No Fix

E4. CORE: In SDS mode, back-to-back memory operations where the first instruction aborts might hang.

Problem: If back-to-back memory operations occur in SDS (special debug state, used by ICE and debug vendors) and the first memory operation gets a precise data abort, the first memory operation is correctly cancelled and no abort occurs. However, depending on the timing, the second memory operation might not work correctly. The data cache might internally cancel the second operation, but the register file may have scoreboarded registers for that second memory operation.

The effect is that the part might hang (due to a permanently scoreboarded register) or that a store operation might be cancelled incorrectly.

Implication: TBD

Workaround: In SDS, any memory operation that might cause a precise data abort should be followed by a write-buffer-drain operation. This operation precludes additional memory operations from being in the pipe when the abort occurs. Do not use load multiple/store multiple, which might cause precise data aborts.

Status: No Fix

E5. CORE: Lock aborts resulting from I-cache or I-TLB lock operations are not presented properly on the trace interface.

Problem: This problem affects only processors that use the core's trace interface. An I-cache or I-TLB lock operation that results in lock abort creates a unique internal pipeline signal timing that causes trouble on the trace interface, which results in the core reporting the event as if it came with the next instruction executed after the aborting lock instruction, the data abort vector.

Implication: TBD

Workaround: None.

Status: No Fix

E6. CORE: CP15 ID register accesses with opcode2 > 0b001 return unpredictable values.

Problem: The XScale core does not implement CP15 ID codes registers other than the Main ID register (opcode2 = 0b000) and the Cache Type register (opcode2 = 0b001). If any of the unimplemented registers are accessed by software (for example, mrc p15, 0, r3, c15, c15, 2), the value of the Main ID register should be returned. Instead, an unpredictable value is returned.

Implication: TBD

Workaround: None.

Status: No Fix

E7. CORE: Disabling and re-enabling the MMU can hang the core or cause it to execute the wrong code.

Problem: If the MMU is disabled via the CP15 control register after being enabled, certain timing cases can cause the processor to hang. In addition, re-enabling the MMU after disabling it can cause the processor to fetch and execute code from the wrong physical address. To avoid these issues, the code sequence below must be employed whenever disabling the MMU or re-enabling it afterwards.

Implication: TBD

Workaround: The following code sequence must be used to disable and/or re-enable the MMU safely. The alignment of the mcr instruction that disables or re-enables the MMU must be controlled carefully, so that it lies in the first word of an instruction cache line:

```
;//@ The following code sequence takes r0 as a parameter. The value of r0 will be
;//@ written to the CP15 control register to either enable or disable the MMU.

    mcr     p15, 0, r0, c10, c4, 1          ;//    @ unlock I-TLB
    mcr     p15, 0, r0, c8, c5, 0           ;//@ invalidate I-TLB
```

```
        mrc     p15, 0, r0, c2, c0, 0           ;//@ CPWAIT
        mov     r0, r0
        sub     pc, pc, #4
        b       AfterAlign                      ;//@ branch to aligned code
        ALIGN 32                                ;//@ align to 32 bytes
AfterAlign
        mcr     p15, 0, r0, c1, c0, 0               ;//@ enable/disable MMU, caches
        mrc     p15, 0, r0, c2, c0, 0               ;//@ CPWAIT
        mov     r0, r0
        sub     pc, pc, #4
```

Status: No Fix

E8. CORE: JTAG parallel register updates require an extra TCK rising edge.

Problem: The IEEE 1149.1 spec states that the effects of updating all parallel JTAG registers should be seen on the falling edge of TCK in the Update-DR state. The XScale core parallel JTAG registers incorrectly require an extra TCK rising edge to make the update visible. Therefore, operations like hold-reset, JTAG break, and vector traps require either an extra TCK cycle by going to run-test-idle or by cycling through the state machine again to trigger the expected hardware behavior.

Implication: TBD

Workaround: If the JTAG interface is polled continuously, this erratum has no effect. If not, an extra TCK cycle can be caused by going to run-test-idle after writing a parallel JTAG register.

Status: No Fix

E9. MMC: SPI mode – if card is deselected, PROG_DONE will not be set.

Problem: If changing SPI chip selects, the PROG_DONE bit does not get updated with the state of the selected card.

Implication: If programming card0, then switch to card1, then come back to card0, there is no way of knowing if card0 ever finished programming.

Workaround: User can switch the MMDAT signal functionality to GPIO functionality and monitor the signal by reading the GPIO Status register until the signal goes high.

Status: No Fix

E10. MEMC: No MRS command is given when exiting from alternate bus master mode when SA-1111 address muxing mode is enabled.

Problem: When using alternate bus master with MDCNFG[SA1110x] set, after the alternate bus master has given the bus back to the processor, if the first access by the processor is a VLIO access, the MRS commands to switch the SDRAMs back into burst-of-4 mode occurs after a long delay. If SDRAM is accessed before the MRS command is performed, data returned to the processor registers will not be valid.

Implication: TBD

Workaround: Use any of the following options to workaround this erratum:

- Do not use SA-1111 address muxing mode with an alternate bus master.
- The alternate bus master can perform the MRS command to put the SDRAM into burst-of-4 mode before de-asserting the MBREQ signal.

Status: No Fix

E11. KBD: Extra keypad matrix interrupt in IMKP mode.

Problem: An unexpected interrupt during keypad-matrix manual scan in ignore-multiple-keypress mode (IMKP). The test consists of pressing one, single, valid key and holding it down, then pressing a second, single, valid key and holding it down, then both pressed keys are released simultaneously. This test results in three vectors to the interrupt-service routine where two are expected.

The first vector to the ISR occurs on first key down. The second vector is unexpected. This is possibly caused by the second key down. The third vector occurs approximately at the time of all keys up.

Debounce interval = 3ms, KPC[IMKP] = 1, KPC[ASACT] = 0; KPC[MI] = 1.

Implication: TBD

Workaround: Do not set KPKDI[Interval] to less than 10ms.

Status: No Fix

E12. USBH: USBH register UHCRHPSx[CCS] bit set incorrectly after power on

Problem: The UHCRHPSx[CCS] bit may be set after power on reset. Under normal conditions, when set, this bit indicates that a USB device is connected to the USB host port. The error is that this bit is being set even though there is no device physically connected to the USB host unit.

Implication: TBD

Workaround: Disable global port power for the USB host unit. Only enable individual port power for the USB host ports that will be used in the system.

Status: No Fix

E13. AC97: Command-done bit remains set after an AC97 cold reset.

Problem: After an AC97 cold reset is asserted (GCR[nCRST]=0) when the AC97 controller unit is enabled again, the command-done bit (GSR[CDONE]) is getting set when it should not.

1. Enable the AC97 unit.
2. Enable the codec.
3. Write some data to the codec register using AClink.
4. Wait for command-done bit (GSR[CDONE]) to set and then clear it.
5. Shut down the codec by writing 0x1000 to codec address 0x26.
6. Wait for command-done bit (GSR[CDONE]) to set and then clear it.

7. Generate wakeup event from Codec and wait for primary codec resume bit (GSR[PRESINT]) or secondary codec resume bit (GSR[SRESINT]) to set.
8. Once resume bit is set, assert a cold reset by writing 0 to GCR[nCRST].
9. Enable the AC97 controller and read the GSR. The command-done bit is set; however, it should be clear.

Implication: TBD

Workaround: After asserting cold reset, the GSR[CDONE] bit must be cleared by writing a 1 to GSR[CDONE].

Status: No Fix

E14. POWER MANAGER: Fast ramp rates on voltage pins can cause high current consumption.[1]

Problem: If voltage rise/stabilization time is too fast for any of the voltage pins (VCC_core, VCC_mem, VCC_batt, etc.), then the internal circuitry causes high current consumption on these voltage pins.

This condition does not damage the internal circuitry of the processor. The duration of the high current consumption is less than 20 µs.

The following describes the current consumption in order of voltage domain bootup sequence.

- If the voltage rise/stabilization time of VCC_BATT is shorter than 500 µs at room temperature, then the extra current on VCC_BATT is less than 100 mA for less than 20 µs.
- If the voltage rise/stabilization time of VCC_USB, VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, VCC_USIM is shorter than 500 µs at room temperature, then the combined current on VCC_USB, VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, VCC_USIM is less than 500mA for less than 20 µs.
- If the voltage rise/stabilization time of VCC_CORE, VCC_SRAM, VCC_PLL is shorter than 500 µs at room temperature, then the combined current on VCC_CORE, VCC_SRAM, VCC_PLL is less than 100mA for less than 20 µs.

Example of battery life consumption:

- Assume square current spike = 850mA = 0.85Coulomb/sec
- Current drain of the current spike lasting 100 µs
- 0.85 Coulomb/sec * 0.0001 sec = 85 mCoulomb total drain
- Assume 150mA-hr battery = 540 Coulombs
- Calculate current spike percentage of total battery capacity: 85mC / 540C = 1.57 e-7
- Current spike consumes only 0.0000157% of battery capacity.

Implication: High current consumption on processor voltage supply pins can cause external power management circuitry to enter into an error/shutdown condition.

Workaround: Voltage rise/stabilization time must be determined by the system designer to mitigate any error condition on external circuitry.

Status: No Fix

1. E14 was corrected and updated as of July 2006

E15. LCD: Reconfiguring the LCD controller retains the previous PPL value for the first line.

Problem: If you program the LCD controller for one configuration, then disable the LCD controller and program it for another configuration and enable it again, the first line retains the pixels-per-line (PPL) value from the previous configuration.

For example, configuring the LCD to 320x200 and then disabling it and reconfiguring it to 640x480, and enabling it again, the first line retains the PPL value from the previous configuration, which is 320.

This issue could effect more than just the first frame, due to the fact that the LCD DMA is initialized with a size to fit the frame size. Data from the first frame could spill over into the second frame, and so on.

Implication: If the wrong number of pixels are sent to the LCD panel for the first line of a frame, the display on the LCD screen could be corrupt.

Workaround: When configuring the LCD the second time, configure it to the desired LCD configuration, enable the LCD, disable it, and then enable it again, for the third time. This ensures that when the LCD controller is enabled for the third time, it will begin the frame with the right PPL value.

Status: No Fix

E16. LCD: Overlay1 is not enabled intermittently after re-enabling LCD.

Problem: When enabling and disabling LCD several times, intermittently, Overlay1 output is all zeros instead of intended frame buffer overlay pattern. Overlay2 works fine.

Implication: TBD

Workaround: Option1:

Users must not use Overlay 1 but can use Overlay 2.

Option2: Users can use Overlay 1 by:

1. Programming the LCD registers
2. Enabling the LCD (LCCR0[ENB] = 1)
3. Disabling the LCD (LCCR0[DIS] = 1)
4. Reprogramming the lcd registers but not the overlay1/cursor registers. Also, do not write DMA FDADR registers
5. Enabling the LCD (LCCR0[ENB] = 1)
6. Performing a quick disable of the LCD (LCCR0[ENB] = 0)
7. Programming the DMA registers
8. Enabling the LCD (LCCR0[ENB] = 1)
9. Updating the overlay/cursor registers if required. Follow the instructions mentioned in the manual in section 7.4.7 for overlay1 and 7.4.10 for cursor as to how to change the overlay/cursor registers dynamically when the LCD is enabled.

Status: Fixed

E17. POWER MANAGER: Processor ignores BATT/VCC faults while exiting sleep mode.

Problem: When asserting nBATT_FAULT or nVCC_FAULT during Sleep mode with the corresponding IDAE set, and the fault is still present after boot, the part does not enter Deep Sleep. However, if the IDAE bit is clear, and the fault is still present after boot, the part enters Deep Sleep as expected.

Implication: If a FAULT continues to occur while the processor is exiting Sleep Mode, then the processor is going to continue to bootup and consume power. There must be enough battery power or the system must be able to tolerate the fault condition until the workaround below is complete.

Workaround: Before entering sleep or deep sleep via software, first write 1 to the PMCR[IAS] bit to enable interrupts and then a separate write must be performed to set the desired PMCR[xIDAE] bits. When the chip exits sleep mode, the interrupt pertaining to Power Management Unit must be unmasked, ICMR[PWR_I2C] = 0b1, to find out if a fault has occurred. Alternatively, the PMCR[INTRS] bit can be checked.

Status: No Fix

E18. KBD: Keyboard Edge-Detect Status Register Incorrect After Standby Mode Wakeup.

Problem: When the processor is in standby mode and it is programmed to wake up by a KeyPad GPIO pulse, the PKSR (Keyboard Edge detect Status) does not correctly report the GPIO that caused the Wakeup.

Scenario 1 (Correct):

If only one keypad GPIO is pulsed, and it is programmed to wakeup from standby in PKWR, the PKSR reports the correct GPIO as the source of the wakeup as expected.

Scenario 2 (Incorrect):

If ALL keypad GPIOs are pulsed but only one is programmed in PKWR to wake up, the PKSR reports all the GPIOs (except for the ones in use) as the source of the wakeup. This is wrong, since only one GPIO was programmed to wake up, regardless of the fact that all GPIOs were pulsed.

Therefore, the following documentation will be added to the Developer's Manual:

"If any of the PKWR bits are enabled, then only the keypad related to that PKWR will cause a wakeup. If there is any other activity (active high) on any other keypad pins at this time, then irrespective of its PKWR setting, it will be registered into the PKSR."

Implication: The user will not be able to detect which keypress woke the processor out of Standby mode.

Workaround: The workaround is to only use PKSR as an indicator that a keypress woke the processor. The user will not be able to tell which key woke the processor.

Status: No Fix

E19. UART: Character Timeout interrupt remains set under certain software conditions

Problem: The issue is that randomly the character timeout interrupt does not clear and the DR bit is not set. The failure was reproduced by adding a software delay loop inside the character timing interrupt routine between reads from the FIFO. The test continuously repeats and increases the software delay. After a few iterations, the tests get into a continuous interrupt loop, where the character timeout interrupt is set, but there is not any data in the FIFO. If the delay loop is placed just outside the loop to read out the data, the test never fails. So the issue looks to be related to the amount of time between reading data out of the FIFO.

Pseudo Code for the Character Timeout Interrupt Handler routine test:

1. Read LSR and check for errors
2. Read Data from FIFO
3. Software Delay (Delay increases after each test)
4. Read LSR, check for errors, and LOOP back to (2) if DR is SET.
5. DONE

if step (3) is placed in front of step (1) the issue never occurs.

Implication: The software servicing the UART can be trapped in an infinite loop.

Workaround: Steps 2 and 6 have been added to the Interrupt Handler routine. Disabling the Receiver Time-Out interrupt via IER[RTOIE] (step 2) prevents the Receiver Time-Out interrupt from becoming continuously set.

1. Read the Line Status Register (LSR) and check for errors
2. Disable the Receiver Time-Out interrupt via IER[RTOIE]
3. Read Data from the UART FIFO
4. Software Delay (Increments after each test)
5. Read LSR, check for errors, and LOOP back to (2) if DR is SET.
6. No more data in FIFO: Re-enable RTO interrupt via IER[4]
7. DONE

Therefore, users must disable the Receiver Time-Out interrupt, then read data from the FIFO, then re-enable the Receiver Time-Out interrupt, before exiting the handler routine.

Status: No Fix

E20. LCD: LCD not enabling in dual panel mode.

Problem: After the LCD unit is dynamically changed from single panel mode to dual panel mode, the LCD unit will not output any data on the LCD pins.

Implication: TBD

Workaround: Do the following sequence to change from single panel mode to dual panel mode:

Configure the LCD to Single panel mode

Enable the LCD

Disable the LCD

Reconfigure the LCD to Dual Panel mode without programming the DMA registers.

Enable the LCD

Do a quick disable (not normal disable).

Program the LCD DMA registers

Enable the LCD for operation in Dual Panel mode.

Status: No Fix

E21. UDC: UDC does not correctly support alternate interfaces.

Problem: When a SET_INTERFACE command is sent from the host, the UDC reallocates the internal FIFO memory.

When USB Client comes up after the USB Reset and Set Configuration is done other than Configuration 0, it does not allow access to all the Interfaces available for that configuration. Until a SET_INTERFACE is issued, only the endpoints assigned to the default interface can be accessed by the host.

Below are excerpts of the USB Specification describing use of Interfaces:

As part of the configuration process, the host sets the device configuration and, where necessary, selects the appropriate alternate settings for the interfaces.

Interfaces are numbered from zero to one less than the number of concurrent interfaces supported by the configuration. Alternate settings range from zero to one less than the number of alternate settings for a specific interface. The default setting when a device is initially configured is alternate setting zero.

see section: Set Interface

This request allows the host to select an alternate setting for the specified interface. This request cannot be used to change the set of configured interfaces (the SetConfiguration() request must be used instead).

see section: Configuration

The descriptor describes the number of interfaces provided by the configuration. Each interface may operate independently. For example, an ISDN device might be configured with two interfaces, each providing 64 Kb/s bi-directional channels that have separate data sources or sinks on the host. Another configuration might present the ISDN device as a single interface, bonding the two channels into one 128 Kb/s bi-directional channel.

An endpoint is not shared among interfaces within a single configuration unless the endpoint is used by alternate settings of the same interface. Endpoints may be shared among interfaces that are part of different configurations without this restriction.

Implication: The processor USB Client does not follow the USB Specification when dealing with different interfaces. External USB host connected to the processor USB Client cannot

switch correctly between alternate interfaces on the processor USB Client. This issue makes the processor USB Client difficult to be a compound device.

Workaround: TBD

Status: No Fix

E22. ICP: Receiver Aborts randomly occur prematurely and without End of frame/Error in FIFO interrupt

Problem: Randomly, Receiver Abort (ICSR0[RAB]=1) occurs prematurely and without the EOF status/interrupt bit set (ICSR1[EOF]=1). When this event occurs, there is always 2 bytes left in the RX FIFO.

A Receiver Abort should always cause an EOF, but in the failing cases, only the Receiver Abort is set.

Adding ~15us delay to the receive data available interrupt (IIR[IID]=0b10) service routine causes this failing case to pass, but limits the transfer size to the size of the FIFO since the added delay will cause an overrun condition.

Implication: TBD

Workaround: Software must do retries after RAB has occurred. Once a RAB event occurs, continue to read data out of the FIFO until the EOF flag sets, then throw away all the data of that frame, and then make a request to the host to resend/retry the last frame of data again.

Status: No Fix

E23. SSP: OSTimer counter increments incorrectly for SSP Frames in Network mode

Problem: When SSP unit is in network mode, when PSP protocol is used, and the SSP is master to Frame (irrespective of sspsclk direction) and FSRT bit is set to 1, OStimer frame counter increments more than the frame is asserted by SSP. For example, 11 frames are asserted as seen on a logic analyzer, but OStimer is showing that the frame signal is asserted for 12 times. OS timer works fine when FSRT is cleared and when in non-network mode.

Implication: TBD

Workaround: Subtract one from the value that is read from the OSTimer frame counter.

Status: No Fix

E24. LCD: Enabling Overlay 2 for YUV420 hangs LCD controller.

Problem: Enabling Overlay 2 in YUV 420 mode causes the LCD controller to stop operating (DMA activity stops and the screen fades away). The test is unable to gracefully shutdown the LCD controller after this failure.

Enabling Overlay 2 in RGB mode has no problems.

Implication: TBD

Workaround: To Enable/Disable Overlay 2 in 4:2:0 mode:

Step 1: a. Enable the LCD with Overlay2 disabled.

Step 2: a. Enable Overlay 2 in RGB mode with minimum size possible so that only one frame worth of data fit in Channel 2 FIFO. Size of Channel 2 FIFO is 128 bytes. Program the Descriptors and write to O2CR1.

b. Run at least 1 frame with Overlay 2.

c. Disable the Overlay 2 by writing to O2CR1 and Frame Branch Registers (FBRx).

d. Wait for 3 base EOF interrupts.

Step 3: a. Enable Overlay 2 in YUV 420 mode. Write to O2CR1.

b. Unmask/Clear the input underrun for Channel 2. Wait for input underrun from Channel 2. Write DMA descriptors for Channels 2, 3, and 4.

Step 4: a. Now Overlay 2 in YUV 420 mode is enabled and running.

Step 5: a. If Overlay 2 needs to be disabled, Disable the Overlay 2 by writing to O2CR1 and Frame Branch Registers (FBRx).

b. If the Overlay 2 needs to be enabled again, Go to Step 3.

Step 6: a. If LCD needs to disabled, disable the Overlay 2 as mentioned in Step 4, then Disable LCD, and go to Step 1.

Pseudo Code to Enable RGB Mode:

Write ovl2c2
Write ovl2c1 (O2EN = 1)
Write fdadr2

Pseudo Code to Disable RGB Mode:

Write ovl2c1 (O2EN = 0)
Clear LCSR1 BS2 bit (LCSR1 = 0x200
Write fbr2 (BRA=BRANCH_AFTER_CURRENT_FRAME, BINT=SET_SR_BS_BIT)
Wait for branch to complete (LCSR1 == 0x200)

Pseudo Code to Enable 4:2:0 mode:

Write ovl2c2
Write ovl2c1 (O2EN = 1)
Clear LCSR1 IU2 bit (LCSR1 = 0x02000000);
Wait for under-run LCSR1 = 0x02000000
Write fdadr2
Write fdadr3
Write fdadr4

Pseudo Code to Disable 4:2:0 mode:

Write ovl2c1 (O2EN = 0)
Clear LCSR1 BS2 bit (LCSR = 0xe00)
Write fbr2 (BRA=BRANCH_AFTER_CURRENT_FRAME BINT=SET_SR_BS_BIT)
Write fbr3 (BRA=BRANCH_AFTER_CURRENT_FRAME BINT=SET_SR_BS_BIT)
Write fbr4 (BRA=BRANCH_AFTER_CURRENT_FRAME BINT=SET_SR_BS_BIT)
Wait for branch to complete (LCSR1=0xe00)

Status: Fixed

E25. USBOTG: Unable to measure duration of Single-Ended Zero (SE0) for Session Request Protocol (SRP)

Problem: According to On-the-Go Supplement to the USB 2.0 Specification, we must be able to detect that a Single-Ended Zero (SE0) condition is driven on the USB bus for at least 2 ms before we can initiate a Session Request Protocol (SRP). See paragraph below.

Sec. 5.3.2..."A second initial condition for starting a new session is that the B-device must detect that both the D+ and D- data lines must have been low (SE0) for at least 2 ms (TB_SE0_SRP min.). This ensures that the A-device has detected a disconnect condition from the device."...

Software has the ability, through reset interrupt, to see when the SE0 state was entered, but it has no way to determine how long it has been in this state, to determine if the 2 ms requirement has been met.

Implication: Dual role devices (A and B devices) are required to be able to initiate and respond to an SRP, therefore, there is the potential that the USB OTG will not pass a USB OTG compliance test.

Workaround: The USB OTG specification says, in section 5.3.2:

"A second initial condition for starting a new session is that the B-device must detect that both the D+ and D- data lines must have been low (SE0) for at least TB_SE0_SRP min. This ensures that the A-device has detected a disconnect condition from the device." Currently, TB_SE0_SRP is specified at 2 msec minimum.

The workaround is to require VBUS to be low for at least 2 msec, and assume that the data lines are low, if VBUS is low. This assumption would work for devices that make the same assumption, but nothing in the specification requires this.

Status: Fixed

E26. MEMC: Write/Read to/from SDRAM can collide with alternate bus master mode when MDREFR:ALTREFB is set.

Problem: When ALTREFB is programmed to 1 and an alternate bus master requests the bus, an SDRAM access in progress may not be allowed to finish before the bus is released. This issue is found in both fly-by mode and non-fly-by mode.

Currently, there is a provision in the specification that ALTREFA and ALTREFB cannot both be set at the same time. Given the workaround below, this will only allow the refreshes to occur both before and after an alternate bus master, or only before an alternate bus master.

Implication: This will potentially cause more refreshes than is necessary according to the SDRAM specification.

Workaround: Never set MDREFR[ALTREFB]. This will cause a refresh cycle to always occur before allowing an alternate bus master to be granted the bus.

Status: No Fix

E27. POWER MANAGER: Core hangs during voltage change when there are outstanding transactions on the bus

Problem: If coprocessor 14 register 7 is written for a voltage change sequence (PWRMODE[VC] = 0b1) and there are outstanding core transactions on the internal bus, the core hangs.

Implication: Unpredictable results can occur if the core hangs.

Workaround: The workaround is to make sure all core transactions are complete and no new core transactions will attempt to get on the internal bus, then initiate a voltage change by setting PWRMODE[VC], then waiting until the PWRMODE[VC] bit is cleared. The VC bit will be cleared within 50ns of the voltage change initiation. If software requires the entire voltage change sequence to complete, i.e. no more communication with the external power manager IC, the software must wait until the PVCR[VCSA] bit is clear.

The following workaround has been tested and will handle all cases.

```
@ WORKAROUND - Core hangs on voltage change at different
@ alignments and at different core clock frequencies
@ To ensure that no external fetches occur, we want to store the next
@ several instructions that occur after the voltage change inside
@ the cache. The load dependency stall near the retry label ensures
@ that any outstanding instruction cacheline loads are complete before
@ the mcr instruction is executed on the 2nd pass. This procedure
@ ensures us that the internal bus will not be busy.

@ -- Begin EnterVoltageChange__asm (void)
      .section.text
      .global EnterVoltageChange__asm
      .align 7                              @ align code to fit in one page
EnterVoltageChange__asm:
      stmfd   sp!, {r4}
      mrs     r0, CPSR                      @ disable interrupts
      mov     r4, r0
      orr     r0, r0, #0xC0
      msr     CPSR_c, r0

      ldr     r0, =0x41300000               @ APB register read and compare
      ldr     r0, [r0]                      @ fence for pending slow apb reads
      cmp     r0, #0

      mov     r0, #8                        @ VC bit for PWRMODE
      movs    r1, #1                        @ don't execute mcr on 1st pass
      mov     r2, #0x0A000000               @ uncacheable memory to force memory read
                                    @ CHANGE THIS ADDRESS for your environment
                                    @ MUST be mapped into pagetable prior to
                                    @ this function call
retry:
      ldreq   r3, [r2]                      @ only stall on the 2nd pass
```

```
        cmpeq   r3, #0                  @ compare causes fence on memory transfers
        cmp     r1, #0                  @ is this the 2nd pass?
        mcreq   p14, 0, r0, c7, c0, 0   @ write to PWRMODE on 2nd pass only

@ Read the VC bit until it is 0, indicates that the VoltageChange is done.
@ On the first pass, we never set the VC bit, so it will be clear already.
VoltageChange_loop:
        mrc     p14, 0, r3, c7, c0, 0
        tst     r3, #0x8
        bne     VoltageChange_loop

        subs    r1, r1, #1              @ update conditional execution counter
        beq     retry

        msr     CPSR_c, r4              @ restore interrupts to original state

        ldmfd   sp!, {r4}
        mov     pc, lr

        .data
@ -- End EnterVoltageChange__asm
```

Status: No Fix

E28. MMC: MMC unit in SPI mode always waits a minimum of 1 Ncx cycles, even though the MMC spec dictates that SPI mode CMD9 can have a minimum of 0 Ncx cycles.

Problem: The MMC unit is not compliant with the MMC Spec 3.2 in SPI Mode.

Version 2.11 of the MMC Spec, in SPI mode, requires 1 or more Ncr cycles between the end of the response and the beginning of the data block for CMD9 (SEND_CSD command).

Version 3.1 and 3.2 of the MMC Spec, in SPI mode, requires 0 or more Ncx cycles between the end of the response and the beginning of the data block for CMD9 (SEND_CSD command).

The MMC unit always waits a minimum of 1 Ncx cycle between the response and the data block, therefore, MMC unit is not compliant with the version 3.2 of the Spec.

Implication: TBD

Workaround: TBD

Status: Fixed

E29. SD: SD Controller in SPI mode not receiving data response for CMD9 and CMD10 from some SD Cards

Problem: The SD unit is not compliant with SD Spec 1.01 in SPI Mode.

Version 1.0 of the SD Spec, in SPI mode, requires 1 or more Ncr cycles between the end of the response and the beginning of the data block for CMD9 (SEND_CSD command).

Version 1.01 of the SD Spec, in SPI mode, requires 0 or more Ncx cycles between the end of the response and the beginning of the data block for CMD9 (SEND_CSD command).

The SD unit always waits a minimum of 1 Ncx cycle between the response and the data block, therefore, the SD unit is not compliant with the 1.01 Spec.

Implication: TBD

Workaround: TBD

Status: Fixed

E30. MEMC: SDRAM Refresh Commands are issued too often during a VLIO access while BREQ is asserted.

Problem: During a VLIO access, if BREQ is asserted, the processor issues Refresh Commands to SDRAM too often. This occurs after BREQ is asserted and before BGNT gets asserted. This situation lasts for as long as VLIO CS is asserted, which can be several microseconds. During this time, the processor is refreshing SDRAM at 100ns interval although programmed interval is around 8usec (80x difference). Everything goes back to normal after BGNT is released.

Implication: TBD

Workaround: TBD

Status: No Fix

E31. INTERRUPT CONTROLLER: Unexpected exception vector when ICCR[DIM]=0 and ICMR=0.

Problem: When ICCR.DIM = 0 and ICMR = 0 and a direct key press is performed, the Interrupt Control IRQ Pending (ICIP) register, does not indicate a pending interrupt after coming up from idle, but the core attempts to vector to an exception vector address (0x000 to 0x01C).

If ICMR=0x0 and ICCR[DIM]=0b0 and a direct key press is performed, then the correct behavior should be:

After the key is pressed, the processor wakes from IDLE, ICPR gets updated, the ICIP (or ICFP) does not get updated, and the processor should not vector to any vector address (0x00 to 0x1C).

The incorrect behavior is:

After the key is pressed, the processor wakes from IDLE, ICPR gets updated, the ICIP (or ICFP) does not get updated, but the processor tries to vector to 0x18 (for IRQ handler) or 0x1C (for FIQ handler), based on the contents of ICLR.

Keypad and DIM Bit and ICMR test failure:

ICCR = 0x00000000, ICMR = 0x04000000.

KPC = 0x000001C3; KPC[DIE] = 1; KPC[DE] = 1;

KPKDI[DIRECT KEY DEBOUNCE INTERVAL] = 20 (msec).

Implication: TBD

Workaround: Code below disables interrupts to the core before going to Idle and re-enables core interrupts after returning from Idle.

```
IDLE:
.global IDLE
    mrs r0, cpsr                    @ read current processor status register
    orr r0, r0, #0xC0               @ disable core interrupts
    msr cpsr_c, r0                  @ update the current processor status register
    mov r0, #1
    mcr p14, 0, r0, c7, c0, 0       @ Set IDLE
    mrc p14, 0, r0, c7, c0, 0       @ CPWAIT ROUTINE
    mov r1, r1                      @ CPWAIT ROUTINE
    sub pc, pc, #4                  @ CPWAIT ROUTINE
    mrs r0, cpsr                    @ read current processor status register
    bic r0, r0, #0xC0               @ enable core interrupts
    msr cpsr_c, r0                  @ update the current processor status register
    mov pc, r14                     @ RETURN
```

Status: No Fix

E32. SSP: TXD line does not tristate when SSP is Slave to Frame

Problem: When the SSP is a Slave to Frame (SFRMDIR=1) and Master of Clock (SCLKDIR=0) using the TI SSP Format, the SSP will not tri-state the TXD line at the end of the last frame when the TTE=1 and TTELP=1 and the SCR>3.

When TI protocol is used, the SSP unit is Slave to Frame and Master/Slave to clock, TTE=1, TTELP = 1, Data Size = 4.

When a data value of 0xA is transmitted, the last data bit is zero, and when a pullup is applied to the TXD pin, the line should go high, but it always remains low.

Similarly, when a data value of 0x5 is transmitted, the last data bit is one, and when a pull-down is applied to the TXD pin, the line should go low, but it always remains high.

When TTE=1, TTELP = 0 is used, the TXD line will tristate at the beginning and end of transfer. Also, the TXD line works fine when the SSP is Master to Frame.

Implication: TBD

Workaround: None.

Status: No Fix

E33. POWER MANAGER: Simultaneous BATT and VDD faults results in going to DeepSleep mode twice.

Problem: When in any mode, asserting simultaneous BATT_FAULT and VDD_FAULT signals, and while BIDAE is not equal to VIDAE, requires the user to go to DeepSleep twice, once automatically due to one of the xIDAE bits being clear, and once via software due to one of the xIDAE bits being set.

Another observation is that a very short (approx. 332usec) nRESET_OUT assertion occurs followed by a bootup (with no change to RCSR).

Implication: TBD

Workaround: PMCR[BIDAE] and PMCR[VIDAE] must be identical.

Status: No Fix

E34. CORE: Non-branch instruction in vector table may execute twice after a thumb mode exception

Problem: If an exception occurs in thumb mode and a non-branch instruction is executed at the corresponding exception vector, that instruction may execute twice. Typically, instructions located at exception vectors must be branch instructions which go to the appropriate handler, but the ARM architecture allows the FIQ handler to be placed directly at the FIQ vector (0x0000001c/0xffff001c) without requiring a branch.

Implication: The first instruction in thumb mode of a FIQ handler may be executed twice if it is not a branch instruction.

Workaround: If a no-op is placed at the beginning of the FIQ handler, the no-op will execute twice and no incorrect behavior will result. If a branch instruction is placed at the beginning of the handler, it will not be executed twice.

Status: No Fix

E35. UART: UART does not correctly indicate a Framing Error Interrupt in DMA mode.

Problem: When DMA is enabled and a “Framing Error” occurs, the UART generates an interrupt, but the IIR register does not show any pending interrupts (IIR[nIP]=0b1).

Reading the registers does not clear the interrupt. The UART will continuously interrupt the CORE.

If the RX FIFO is emptied until there is only 4 bytes of data left in the FIFO, the UART will finally generate a RLS interrupt showing a “framing” error.

The UART handles errors differently in DMA mode. When an error occurs the error gets tagged in the rx fifo with the erroneous byte. In interrupt mode, the error is flagged (processor interrupted) when the erroneous byte is read out of the bottom of the fifo. In DMA mode, the error is flagged as soon as it is detected. In DMA mode, the UART correctly interrupts the processor when the error is first detected but it does not update the IIR register until the error is read out of the FIFO. Thus, the processor is interrupted with no interrupt pending. The correct operation is to both interrupt the processor and update the IIR register when the error is detected.

Implication: The UART generates an interrupt, but the IIR register does not show any pending interrupts.

Workaround: If an interrupt is generated, but the IIR register does not show any pending interrupts, then read data out of the RX FIFO until either it is empty or a valid interrupt is generated. The error interrupt to the core will not clear until the erroneous byte is read out of the RX FIFO.

Status: No Fix

E36. CLOCKS: System Hangs when enabling RUN/TURBO switching at 520 MHz

Problem: The system will hang when performing switching from Run Mode frequency to Turbo Mode frequency.

The test flow is as follows:

- Bootup
- Perform a Frequency / Voltage change to L=16, N=2.5, A=1, B=1, K0DB4=1, K1DB2=1,K2DB2=1. This gives core = 208 MHz (run mode), Sys Bus = 208 MHz, and MEMC = 208 MHz. Note the Turbo bit (T) is not set yet.
- Display a code of “0xFFAA0000”
- Go to Turbo mode by setting the Turbo bit.
- Display a code of “0xFFBB0000”

This test intermittently hangs after the “FFAA” code and before the “FFBB” code. Intermittently means that the test can switch back and forth between Turbo mode and Run mode and a failure will most likely occur between 10,000 iterations and 400,000 iterations.

Implication: TBD

Workaround A: Workaround for going from RUN (208 MHz) to TURBO (520 MHz):

```
Set CCCR[CPDIS]                          // Disable the Core PLL
Clear CCCR[PPDIS]                        // Enable the Peripheral PLL
Set CLKCFG[F]                            // Initiated a Frequency Change
// Core freq = 13MHz, Peripherals = normal freq (PPLL = 312MHz)
Set CCCR[PLL_Early_EN]                   // Allow the Core PLL to ramp up early
Wait for CCSR[CPLCK] to set and CCSR[PPLCK] to set.
Clear CCCR[CPDIS]                        // Enable the Core PLL
Set CLKCFG[F] and Set CLKCFG[T]          // Initiate a Frequency Change
// Core is now running in Turbo mode @ 520Mhz
```

No Workaround needed for switching from TURBO (520 MHz) to RUN (208 MHz). Switching from TURBO to RUN works fine by clearing CLKCFG[T].

Estimated Latency Timings:

From RUN to 13M mode is about 45µs

From 13M to PLL lock is about 15µs
From PLL lock to TURBO is about 100µs

Total latency is about 160µs, where 15µs of the time you can execute code while the core is operating in 13 MHz mode. This time can increase if the PLL lock time changes.

Workaround B: If you choose to go from 208 MHz RUN mode, into a different product point, such as 416 MHz Turbo, then do a full frequency change sequence to switch into 520 MHz Turbo, the latency is about 230µs.

Status: No Fix

Note: For systems using the 520 MHz Core Turbo Freq, 208 MHz CLK_MEM operation mode (520/208/208/104), either workaround is acceptable. However, for systems using the 520MHz Core Turbo Freq, 104MHz CLK_MEM operating mode (520/208/104/104) only Workaround B is effective.

E37. CLOCKS: System Hangs when enabling HalfTurbo Switching

Problem: The system will hang when switching into and out of HalfTurbo, i.e. when going from Run mode to HalfTurbo mode and from Turbo mode to HalfTurbo mode. The failure does not occur when only switching from Run mode to Turbo mode and back.

This failure occurs on, but not limited to, the L=8, N=3, T=1, HT=1, A=1, B=1 product point (i.e. Core = 156 MHz, System Bus = 104 MHz, MEMC = 104 MHz).

Failures on this product point are intermittent. Intermittent means that on powerup, the test will either run or hang. If the first change to HalfTurbo works, then the test will continue to run without failure. If the first change to HalfTurbo fails, then the test hangs, and nothing else can be done.

Implication: TBD

Workaround: Do not use HalfTurbo Mode. If you wish to run the core at a HalfTurbo Frequency, then perform a full frequency change to the desired frequency. For example, if you wish to run the core at 156 MHz, then set L=8, N=1.5, T=1, HT=0, A=1, B=1.

Status: No Fix

E38. MEMC: Memory Controller hangs when entering Self Refresh Mode.

Problem: If software manually puts SDRAM into Self Refresh Mode, then the memory controller will not perform any more activity.

Here are the steps to recreate the issue:

1) MDREFR[K1RUN] = 1 // make sure SDCLK1 is running.

2) MDREFR[K1FREE] = 0 // turn off free running.

3) MDREFR[SLFRSH] = 1 // enter self refresh mode.

4) wait until enter Self Refresh mode. // delay approximately 2 usec.

5) Any non-SDRAM activity will not be performed by the memory controller

Implication: Will not be able to access external non-SDRAM memories after the SDRAMs are put into Self Refresh Mode.

Workaround: The user is allowed to assert the SLFRSH bit to the memory controller placing the SDRAMs in Self Refresh Mode, if and only if, the very next transaction the memory controller receives is the deassertion of the SLFRSH bit. Nothing else is permitted on the memory controller after the assertion of the SLFRSH bit, and before the deassertion of the SLFRSH bit. Essentially, this means that the user may perform internal transfers that do not effect the memory controller, such as reads from Instruction Cache, reads/writes to Data Cache, reads/writes to SRAM, reads/writes to internal peripherals, but cannot perform external accesses such as VLIO or PCMCIA

Status: No Fix

E39. SDIO: SDIO Devices Not Working at 19.5 Mbps

Problem: SD/SDIO controller can only support up to 9.75 Mbps data transfer rate for SDIO card. However, the SD/MMC card works fine at 19.5 Mbps clock rate.

The error reason is “HandleEndCommandInterrupt: response for command 52, contains a CRC error”. Other commands may respond with a CRC error also.

Many cards were tested and some fail and some pass at 19.5 Mbps, however all cards pass at 9.75 Mbps.

Test was conducted under BSP 3.00.029, V2 Beta that supports full SDIO functionality. Test was conducted in both 1-bit and 4-bit mode and both modes fail. Test was conducted on a Mainstone board.

The Mainstone board has an analog switch in the SDIO path. This switch, according to the spec, can cause up to 18ns of delay. The delay causes the clock to be later at the card and then the data gets delayed going back to the processor. According to the SD spec, the output delay maximum is 15ns @ 25MHz, or 5ns of setup time before the rising edge of the clock. This analog switch was taken out of the circuit, however, cards were still failing.

Implication: TBD

Workaround: Slow the speed of the interface down until the card passes. This speed could be as low as 9.75 Mbps.

Status: Fixed

E40. AC97: Command Done bit is never set when data is written in slot12

Problem: The “Codec Done Bit” does not get set upon completion of a write to register 0x54 in Modem IO Space. Moreover, no AC97 controller interrupt will occur to the CPU, telling the software that a register write to 0x54 is complete.

Implication: TBD

Workaround: Software must poll the CAIP bit to determine if a write was accomplished.

Status: No Fix

E41. SD/MMC: SD/MMC controller CRC errors with some SD/MMC cards

Problem: In MMC/SD/SDIO cards, not in SPI mode, CMD2, CMD9, and CMD10 use an R2 type Response. An R2 Response is a 136 bit Response, where:

- [135:128] is 0x3F
- [127:1] is the register being read, where bits [7:1] is the CRC for bits [127:8] of the register
- [0] is the END bit, which is always 1.

The controller ignores bits [135:128] for CRC checking.

The controller should use bits [127:1] for CRC checking. The problem is that the controller is using bits [126:1] for CRC checking. If bit 127 is a 0, then there is no problem, because the CRC checker is always initialized to 0. However, if bit 127 is a 1, then a response CRC error will occur.

Implication: TBD

Workaround: If a Response CRC error occurs for CMD2, CMD9, or CMD10, and bit 127 is a 1, ignore the error.

Status: Fixed

E42. USBH: There is no Individual Power Sense Polarity bit for each Host Port. The Power Sense Polarity bit controls the polarity for all three Host Ports.

Problem: Overcurrent indicator signals for Host Port 1 and 2 are controlled externally by USBHPWR[2:1]. Overcurrent indicator signal for Host Port 3 is not pinned out to an external pin. This signal is internally tied to ground.

The Power Sense Polarity bit (UHCHR[PSPL]) controls the polarity of the Overcurrent indicator signals for all three Host Ports.

If a usage model requires that the polarity of the Host Port 1 or 2 be active low, then this will cause a false overcurrent indication to occur on Host Port 3.

Also, software cannot ignore overcurrent on just Host Port 3. Ignoring overcurrent can only be done across all three ports by setting UHCRHDA[NOCP].

The intended silicon fix is to disconnect the Host Port 3 Overcurrent indicator signal from internal ground and connect it to the Power Sense Polarity (PSPL) bit. This guarantees that Host Port 3 will never see an overcurrent condition.

Note that when switching PSPL, a small glitch might be seen on the Overcurrent indicator for Host Port 3. This should not be an issue since the usage model for PSPL is that it is designed to be set statically at initialization time.

Implication: TBD

Workaround: TBD.

Status: Fixed

E43. KBD: Keypress wakeup from Standby mode is not reliable

Problem: When the processor is put into Standby mode by correctly configuring the PKWR register, a valid keypress does not reliably bring the processor out of Standby mode.

The failure rate is less than 10 times out of 100 attempts to wakeup from Standby mode via a key press.

Implication: TBD

Workaround: Do not put the processor into Standby mode during keypress activity.

Status: No Fix

E44. USBH: USB Host Port 3 in Transceiverless Mode may not work correctly with an external device.

Problem: The USB Host Port 3, when put into Transceiverless Mode by setting UP3OCR[CFG] = 2, expects a certain behavior from the attached external device.

While the device is in receive mode, i.e. while USB_P3_2 (OE_n) is deasserted, the Host Port 3 expects the device to transmit a steady state “1” on the USB_P3_6 (VPO) pin and a steady state “0” on the USB_P3_4 (VMO) pin.

Implication: TBD

Workaround: Use a 2-input OR gate, with one input connected to OE_n from the device, the other input connected to VPO from the device, and the output connected to USB_P3_6 (VPO) of the processor.

Note: Refer to Table 3, “USB Client Controller: Bit definition change in USB Port 2 Output 2 Control Register (UP20CR)” for details on enabling this silicon fix.

Status: Fixed

E45. SD/MMC: SPI mode commands fail even on cards that are compatible with SPI spec 1.0

Problem: When the SD/MMC controller is put into SPI mode, all read commands where data is transmitted in the DATA token, will fail, even if the card is SPI spec 1.0 compliant.

For example:

- C0 step silicon:
  - — CMD9/CMD10 failed on cards compatible with SPI spec 1.0
  - — CMD9/CMD10 failed on cards compatible with SPI spec 1.0.1

Another example:

- CMD52 can do 1 byte reads. The read data is returned in the Response Token and there is no data transfer on DAT line. Therefore, this command will pass.
- CMD53 can read multiple bytes and blocks. The read data is returned on the DAT line. Therefore, this command will fail.

Implication: When the SD/MMC controller is put into SPI mode, all read commands where data is transmitted in the DATA token, will fail.

Workaround: Do not use SD/MMC in SPI mode or use Read Commands that return data in the Response Token.

Status: Fixed

E46. CLOCKS AND POWER: PWM Clock Enables do not work as specified

Problem: According to the specification, CKEN[0] controls the PWM0 and PWM2 Clock Enable, and CKEN[1] controls the PWM1 and PWM3 Clock Enable. However, CKEN[0] and CKEN[1] both have to be disabled in order to disable any of the PWMx clocks.

CKEN[1:0]	Current Spec	New Spec
0b00	All PWMs disabled	All PWMs disabled
0b01	PWM0, PWM2 enabled	All PWMs enabled
0b10	PWM1, PWM3 enabled	All PWMs enabled
0b11	All PWMs enabled	All PWMs enabled

Implication: TBD

Workaround: To disable a PWMx clock, CKEN[1:0] must be 0b00.

Status: No Fix

E47. ICP: Occasionally EIF, EOF and CRC interrupt are missed when a CRC error is received

Problem: On some of the product points, occasionally the EIF, EOF, and CRC interrupts are missed when a CRC error is received.

If you read the data out of the RX FIFO, the last piece of data does set the EOF and CRC status bits, but the EIF bit never gets set.

If the interrupt latency is increased by approximately 20usec, the correct behavior is seen. Interrupt latency equals the time from when the ICP generates the interrupt to the time the processor is able to service the interrupt. Latency can be increased by looping on the OS timer until 20us has past.

Implication: If the ICP received a CRC error and is configured to use interrupts, occasionally the EIF, EOF, and CRC interrupts are missed when a CRC error is received.

Workaround: Configure the ICP to use DMA-mode for data transfers. Do not attempt to use interrupt-mode for data transfers.

Status: No Fix

E48. POWER MANAGER: Batt Fault does not always re-enable GPIO 0 and GPIO 1 as wake-up sources.

Problem: If the 13MHz oscillator is disabled during sleep mode, the processor does not comply to the following statement in section 3.8.1.4 of the *PXA27x Processor Family Developer's Manual*:

"When nVDD_FAULT or nBATT_FAULT is asserted, PWER assumes its reset value, enabling only GPIO<1:0> as wake-up sources."

If the following sequence is performed:

1. The PWER[1:0] bits are cleared, which disables wake-up due to GPIO<1:0> edge detect
2. The PCFR[OPDE] bit is set, which disables the 13MHz oscillator during sleep mode
3. The processor is put into sleep mode
4. A nVDD_FAULT or nBATT_FAULT is asserted
5. A GPIO<1:0> edge is asserted

then there is no guarantee that the processor will wake from sleep mode.

Implication: There is no guarantee that the processor will wake up from sleep mode, if the PWER[1:0] is cleared, and a nVDD_FAULT or nBATT_FAULT is asserted during the sleep mode, and then a GPIO<1:0> edge is asserted.

Workaround: Any of the following workarounds can be used:

1. Keep PWER bit corresponding to the wakeup enabled before entering sleep.
2. Assert GPIO wakeup signal(s) while the fault is asserted.
3. Keep the 13 MHz oscillator running during sleep mode.

Status: No Fix

E49. POWER MANAGER: The processor does not exit from sleep/deep-sleep mode.

Problem: The processor is exhibiting a problem when waking up from sleep and deep-sleep mode. Specifically, the issue is a race condition between clocks in the processor's sleep/deep-sleep wakeup circuitry. When the processor begins to wake from sleep/deep-sleep mode, a reset to the clock unit de-asserts while the internal power and ground supplies to the wakeup circuitry are not yet stable. The effect is that parts of the clock unit will be in an unknown state after the reset has been de-asserted. In most cases, the race condition will end successfully without de-asserting the reset too early and the issue will not be seen. However, in a small number of parts, the reset de-asserts too early, thus causing the issue.

Implication: Early release of the reset prior to a good, stable power and ground supply can lead to unpredictable values during wakeup and can result in the processor stopping execution. The Erratum is also manufacturing process dependent and intermittent, so only a small percentage of devices are ultimately affected.

Workaround: Marvell has found that keeping VCC_Core powered during sleep mode seems to prevent the predictable behavior because it helps keep the correct state of the wakeup

circuitry, even when the reset to the wakeup circuitry is de-asserted early. Additional testing has also shown that entering sleep mode at 91 MHz with VCC_Core at 0.95V (+/- 5%), but then grounding VCC_Core during sleep helps to alleviate the issue in a large number of cases, but not all. Finally, the processor's watchdog timer can be implemented to reset the processor, if the failure occurs. This gives customers the ability to gracefully handle a system failure due to this erratum. Use of the potential workarounds can help greatly reduce or possibly eliminate the occurrence of the erratum. Details of these workarounds can be found in the following Application Note: *Sleep/Deep-Sleep Exit Procedure for the Marvell® PXA27x Processor Family.*

Status: Fixed

E50. SDIO: CMD53 multiple-block data transfer with block count set to 0 not supported.

Problem: A block count of 0 for CMD53 is not supported.

Implication: The block count field in CMD53 is 9 bits (511 maximum blocks). To send more than 511 blocks, set the block count in CMD53 to 0 in the command argument. The PXA27x SDIO controller does not support this functionality.

Workaround: If more than 511 blocks are to be transferred, software should send multiple CMD53 multi-block transfers with a block count that is <= 511.

Status: No Fix

E51. LCDC: Disable-done interrupt does not always occur.

Problem: The LCD controller disable-done and quick-disable-done interrupts are not always issued the second time the LCD controller is enabled and then disabled.

Implication: During normal LCD controller operation, setting the LCD-disable LCCR0[DIS] bit = 1 or LCD-controller-enable LCCR0[ENB] bit = 0 can be used to disable the LCD controller. After the LCD controller has been disabled, a disable-done or quick-disable-done interrupt is issued if their respective mask bits are not enabled. After the LCD controller has been enabled and then disabled for the second time, the disable-done or quick-disable-done interrupts are not issued.

Workaround: Poll the LCD-disable-done flag, LCSR0[LDD], instead of relying on the disable-done or quick-disable-done interrupts.

Status: No Fix

E52. UDC: RCV can not be tied high during UDC transmission when using an external transceiver.

Problem: When the RCV signal is held high during an IN transmission, the UDC may stop transmitting future packets.

Implication: RCV is one of the signals used for the single-ended USB interface option. The RCV input signal is the resultant signal that represents the conversion of the differential pair D+ and D- to a single digital signal. When a USB host makes a request for an IN transmission, if RCV is held high during the IN transmission, the UDC may not acknowledge subsequent ACKs from the host.

Workaround: During an IN transmission, insure that RCV is either held low, or sees the reflection of VP transmit data. VP is another one of the digital signals that is part of a single-ended USB interface. VP is the digital version of the differential signal D+.

Status: No Fix

E53. AC97: AC97 CAR[CAIP] bit field can be incorrectly set

Problem: 1) If software reads the CAR register in the same cycle that the AC97 unit internally clears the lock, the lock is re-acquired in the same cycle. The (new) locked status is returned to software as the value of CAR. Software never detects that the lock was released.

The AC97 controller provides a bit, CAR[CAIP], that can be used by software as a flag to indicate the AC-link interface is currently busy with an I/O cycle. If you read this bit and get a 0, you have the “lock” and it is safe to use the AC Link. The act of reading the CAR[CAIP] bit sets it to 1. When software reads the CAR[CAIP] bit and sees a 1, that means the AC-link interface is already busy with another driver’s I/O cycle.

2) When software reads an AC97 CODEC register other than register 0x54, two AC-link reads occur. The CAR[CAIP] bit gets cleared (by hardware) after each AC-link read. Since the second AC link read is still in progress, the CAR[CAIP] should not be cleared at this point.

Implication: 1) This results in software looping on a wait-for-CAR[CAIP]-free loop.

2) N/A

Workaround: 1) S/W should not read the CAR[CAIP] bit to determine if a CODEC transaction is in progress

2) S/W should not read the CAR[CAIP] bit to determine if a CODEC transaction is in progress

See Table 7, “Recommended Procedure for Accessing AC97 CODEC Registers”.

Table 7. Recommended Procedure for Accessing AC97 CODEC Registers

Operation	Step by step	IO Completion Detection	End of IO clean up
Read a codec register other than register 0x54	1. Read memory space - discard this data 2. Wait for the IO to complete 3. Read memory space again - save this data 4. Wait for the IO to complete 5. Return the saved data	Wait for GSR:SDONE to assert for step 2 Wait for GSR:SDONE to assert for step 4	Clear GSR:SDONE and GSR:CDONE after step 4
Write a codec register other than register 0x54	1. Write to memory. 2. Wait for IO to complete.	Wait for GSR:CDONE to assert for step 2	Clear GSR:CDONE after step 2
Read Modem register 0x54	1. Read from memory for register 0x54. 2. Wait for the IO to complete.	Wait for GSR:SDONE to assert for step 2	Clear GSR:SDONE and GSR:CDONE after step 2
Write Modem register 0x54	1. Write to memory for register 0x54. 2. Wait for the IO to complete	Wait 50 microseconds for step 2.	None

Note: Acquiring the CAR[CAIP] lock is no longer recommended

Status: No Fix

E54. SD/MMC: SD Command 56 Read Error

Problem: When the SD/MMC Controller sends a read command to an SD Card using CMD56 in SD mode, the read does not work correctly.

Implication: When the SD/MMC Controller sends a CMD56 (with MMC_ARGL[ARG_L] = 0x1), software waits for the MMC_I_MASK[RXFIFO_RD_REQ] bit to get set. This does not get set.

Workaround: Software can use CMD17 to perform a Single Block Read.

Status: Fixed

E55. AC97: AC97 Unit Incorrectly Receives An EOC

Problem: When AC97 and MMC are enabled, and DMA services the FIFOs to these controllers, the AC97 Controller may incorrectly detect an End Of Chain (EOC) from the DMA Controller, intended for the MMC Controller.

Problem Description:

The DMA Controller generates an EOC (end of DMA chain) indication to the MMC. The AC97 incorrectly absorbs this EOC, which causes the AC97 to de-assert its Rx DMA request. The DMA deadlocks on the AC97 channel because it is waiting for a Rx DMA request.

Possible EOCs:

- PCMISR[EOC] will be set (the PCM-in request deasserted) anytime the DMA Controller signals an EOC to an address 0xXXX0-0040.

- MCSR[EOC] will be set (the Mic-in request deasserted) anytime DMA signals an EOC to an address 0xXXX0-0060.

- MISR[EOC] will be set (the Modem-in request deasserted) anytime DMA signals an EOC to an address 0xXXX0-0140.

Implication: Hardware incorrectly sets the PCMISR[EOC] bit which signals the end of descriptor chain. The DMA Channel stops after signaling EOC and is not able to service the FIFO further. This causes the AC97 Controller to hang.

Workaround: An EOC will cause an interrupt.

The resulting interrupt service routine must do the following at all times:

- Check if PCMISR[EOC] or MCSR[EOC] or MISR[EOC] is set.

- If any EOC bits are set, software should check to see if the DMA channel servicing the AC97 has also stopped using the DCSRx[STOPINTR].

If DCSRx[STOPINTR] = 0b1 (channel stopped): This indicates a 'true' EOC for one of the AC97 channels. In this case, the EOC occurred and the channel is stopped, indicating that the EOC was correctly received.

If DCSRx[STOPINTR] = 0b0 (channel not stopped): This indicates that the AC97 incorrectly received the EOC.

In this case, clear the EOC bit in the corresponding AC97 status register.

Clearing the EOC bit, will re-assert the DMA request and the FIFO will continue to be serviced by the DMA Controller.

Status: Fixed

E56. MMC: MMC Write CRC Response Error

Problem: The MMC controller incorrectly reads the CRC status bits that the MMC card generates following a MMC controller write, on some cards.

Implication: The MMC controller returns an incorrect “Write CRC error” status, after writing to the MMC card.

- The erratum only effects MMC cards that:
  - — Generate their CRC response based on the positive edge of the MMC clock and
  - — Are Frequency restricted to operation less than 19.5MHz.

Workaround: Cards that support 19.5MHz do not need a workaround.

If the MMC Controller is required to operate at less than 19.5MHz and a “Write CRC error” status occurs, a software corrective action is possible.

Workaround steps: (only needed when the workaround mode is required)

- 1. Before each write operation, copy the write data if necessary. (If the buffer passed by the caller is still available after the write operation, this might not be necessary).
- 2. Perform the write.
- 3. If a CRC error on write status is reported then:
  - a. Read the block from the card into a separate buffer
  - b. Compare the read buffer contents with the write buffer (or copy buffer) contents
  - c. If the data matches, return a “success” status for this write request - this indicates that the CRC error is incorrect.
  - d. If the data comparison fails, return the CRC write error status to the caller.
- 4. Return any other status to the application

Status: No Fix

E57. USB: USB Host Port 3 Transceiverless Mode Restrictions

Problem: When the USB Host Port 3 is in transceiverless mode, it can only be connected to a full speed device and that device cannot use the common method of indicating a connect.

Implication: When USB Host Port 3 is in Transceiverless Mode (UP3OCR[CFG] = 0x2) and UP2OCR[VPMBlockEnbN] = 0b0, the following restrictions apply:

- USB Host Port 3 can only be connected to a full speed (12Mbps) device.
- The USB device cannot use the common method of indicating a connect by pulling the VPO line high, or disconnect by pulling both VPO and VMO low. The USB device must use another method of indicating a connect/disconnect to the Host Port 3, such as using another GPIO, or performing a read to the external device’s status register.

Workaround: None

Status: No Fix

E58. MMC/SD/SDIO: Read Data Command May Hang The Controller

Problem: For MMC/SD/SDIO and SPI mode read commands, when the read data finishes before the response token is sent from the card, the MMC/SD/SDIO controller gets stuck in a state waiting for data. This applies to both DMA and PIO modes.

Implication: If the data transfer starts and finishes before the response token is sent, the MMC/SD/SDIO controller will appear to hang as it waits for data.

If the response token is sent before or at the same time as the data, or the data size is larger than the response, then the controller functions correctly.

Workaround: All read data transfers must be a minimum of 8 bytes for 1-bit mode and 32 bytes for 4-bit mode.

- The response can start 2 - 64 cycles after the command

— And the data can start 2 – (card dependent) cycles after the command

For MMC cards (which only supports 1-bit), the read data transfer must be a minimum data size of 8 bytes.

For SD or SDIO cards using 1-bit and 4-bit mode:

— If in 1-bit mode, the read data transfer must be a minimum data size is 8 bytes.

— If in 4-bit mode, the read data transfer must be a minimum data size is 32 bytes.

Status: No Fix

E59. VCORE: Voltage Sensitivity at VVCCC(2,4) May Result in Undetermined System Behavior

Problem: When operating within the recommended voltage tolerance for 13 MHz (with the peripheral PLL(PPLL) active), 91 MHz, 104 MHz, and 208 MHz (VVCCC(2,4)) the processor may exhibit a voltage sensitivity during boot-up, voltage transitions, or normal operation. This voltage sensitivity only occurs under certain operating conditions and environments.

Implication: When this erratum occurs, it may result in possible system hangs, instability, or a data abort depending on the operating conditions and environment. Systems with a power management integrated circuit (PMIC), requiring a boot voltage higher than the documented boot frequency voltage, may not experience this issue. Systems implementing 13 MHz (with the PPLL deactivated) are not affected by this erratum.

Workaround: Note the following changes that will eliminate E59 as an issue.

VCC_Core Vmin Spec (EMTS)
VVCCC1: Core Voltage and Frequency Range 1 (13/13/13/13 CCCR[CPDIS]=1, CCCR[PPDIS]=1 (See **Table 11** in the *PXA270 EMTS*)
-Typical: change from 0.9 V to 1.0 V
-Minimum: change from 0.9075 V to 0.95 V

VVCCC2: Core Voltage and Frequency Range 2 (13/13/13/13 CCCR[CPDIS]=1, CCCR[PPDIS]=0, (91/45/5/91/45/5) and (104/104/104/104) (See **Table 11** in the *PXA270 EMTS*)
-Typical: change from 0.9 V to 1.0 V
-Minimum: change from 0.855V to 0.95 V

Deep Idle (See **Table 13** in the *PXA270 EMTS*)
-Max: change from 0.935 V to 1.705 V
-Typical: change from 0.85 V to 1.0 V
-Minimum: change from 0.8075 V to 0.95 V

VVCC0: VCC_BATT Voltage (See **Table 9** in the *PXA270 EMTS*)
-Max: change from 2.25 V to 2.40 V

Status: Fixed in the *PXA270 EMTS Rev D*.

E60. RTC: Stopwatch SWCR alarm inaccuracy

Problem: When the RTC Stopwatch Counter Register (SWCR) matches either of the RTC Stopwatch Alarm Registers (SWAR1/2) while the corresponding Stopwatch Alarm Enable bit (RTSR[SWALE1/2]) is set, the RTC Stopwatch alarm occurs. The actual elapsed time may be incorrect by more than 1/4th of one second.

Implication: The SWCR Seconds field is clocked by a 1Hz clock and the SWCR Hundreths field is clocked by a 100Hz clock, therefore the SWCR Seconds field and Hundreths field must synchronize in order to achieve accurate Stopwatch alarm.

Workaround A: Use the 1Hz rising edge signal as the synchronization method between the SWCR Seconds field and Hundreths field.

1) Clear the RTSR[HZ] bit indicating no 1Hz rising edge has been detected.

2) Set the RTSR[HZE] bit to enable the HZ interrupt.

3) Set the RTSR[SWCE] bit. This starts the SWCR counter incrementing.

4) Wait until the 1 Hz rising edge has been detected(RTSR[HZ]=1).

5) Write the SWAR1/2 as desired.

6) Set the RTSR[SWALE1/2] bit to enable the alarm for the desired SWAR1/2 register.

7) If SWCR matches SWAR1/2 the Stopwatch alarm will occur correctly.

8)To prepare for another SWAR1/2 match, software will need to resynchronize by starting at step 1.

Workaround B: Use the SWCR Hundreths field rollover as the synchronization method between SWCR Seconds field and Hundreths field.

1) Set the RTSR[SWCE] bit. This starts the SWCR counter incrementing.

2) Write a non-zero value (SWCR[HUNDRETHS] = 0x1) to the Hundreths field of the SWCR register.

3) Verify the Hundreths field of the SWCR is written successfully.

4) Wait until the Hundreths field reads 0 indicating internal rollover has occurred on a rising edge of the 1Hz signal.

5) Clear the RTSR[SWCE] bit to stop SWCR counter.

6) Write the SWAR1/2 as desired.

7) Set the RTSR[SWALE1/2] bit to enable the alarm for the desired SWAR1/2 register

8) Set RTSR[SWCE] bit. This starts the SWCR counter incrementing.

9) If SWCR matches SWAR1/2, the Stopwatch alarm will occur correctly.

10)To prepare for another SWAR1/2 match, software will need to resynchronize by starting at step 1.

Workaround C: A write operation to the SWAR2 register does not reset the SWCR counter register. Therefore if the Stopwatch Alarm Register 2 (SWAR2) is used for match time, a simple workaround for accurate detection of RTC stopwatch alarm can be used. This workaround does not require any wait time when a new match is desired. A wait of up to one second is performed only once during initialization.

1) Set RTSR[SWCE] to enable the SWCR counter. Let it run continuously as a free running counter.

2) Wait until the Hundreths field of the SWCR is non-zero.

3) Wait until Hundreths field of SWCR is 0. This insures that internal rollover of the Hundreths field has occurred.

4) Load alarm match time (current SWCR counter value + desired alarm interval) in SWAR2.

5) Set the RTSR[SWALE2] bit to enable the alarm for the SWAR2 register.

6)To prepare for another SWAR2 match, return to step 4.

Status: No fix

E61. MEMC: Memory Controller may hang while clearing MDREFR[K1DB2] or MDREFR[K2DB2]

Problem: The SDRAM memory controller may hang while clearing the MDREFR K1DB2 and K2DB2 bits. If the memory controller is in the middle of performing a CBR command to the SDRAM, and the clocks change due to the K1DB2 or K2DB2 bits being changed from 1 to 0, there is a chance the system will hang, with the memory controller staying in the CBR state forever.

Note: The issue only exists when clearing the K1DB2 or K2DB2 bits from 1 to 0. It does not occur when setting these bits from 0 to 1.

Implication: TBD

Workaround: The workaround provided in E88 includes and enhances the workaround for this errata. The prefered solution for future software is to use the workaround for E88.

Before clearing either the K1DB2 or K2DB2 bits, temporarily disable interrupts and then write a large DRI value into the MDREFR. This action delays the next refresh cycle and ensures that the CBR command does not occur when the DB2 bit is cleared. Then write MDREFR[DRI] with proper value and K1DB2 or K2DB2 with 0(all in one write). Enable interrupts at last. This workaround can be completed in seven steps:

1) Disable interrupts.

2) Leave the KxDB2 bits at their current state, but set MDREFR[DRI] to 0xFFF.

3) Read MDREFR one time.

4) Wait 1.6167 µs. This can be accomplished by reading the OSCR0 register until its 3.25MHz counter has incremented 7 times.

5) Clear K1DB2 and/or K2DB2 bits, and set MDREFR[DRI] to the proper value at the same time.

6) Read MDREFR one time.

7) Enable interrupts.

Status: No Fix

E62. POWER MANAGER: The processor can not execute the nBATT_FAULT or nVDD_FAULT xIDAE abort handler if a fault occurs while in sleep mode.

Problem: If nBATT_FAULT or nVCC_FAULT asserts while the processor is in sleep mode with corresponding PMCR[xIDAE] bit set, the processor will wake up from sleep mode. If PMCR[IAS] is cleared, the expected behavior is that the abort handler will be processed. However, the abort handler is not processed as expected in this situation.

Implication: TBD

Workaround: None

Status: No Fix

E63. POWER MANAGER: nBATT_FAULT and nVDD_FAULT during sleep mode do not prevent RTC wakeup.

Problem: While in sleep mode, if deep sleep mode is entered due to a power fault (assertion of nBATT_FAULT or nVDD_FAULT with PMCR[xIDAE] clear), after the fault signal is deasserted the exit from deep sleep is expected to be limited to a wakeup event on GPIO<1> or GPIO<0>. However, if the processor enters deep sleep mode due to a fault signal with an RTC wakeup enabled, the processor may be woken by the RTC alarm after the nBATT_FAULT or nVDD_FAULT signals are de-asserted.

Implication: TBD

Workaround: To implement the workaround:

1. Set PMCR[xIDAE] and PMCR[IAS] bits to force execution of interrupt handler (the Power I^2C interrupt handler) upon assertion of nBATT_FAULT or nVDD_FAULT.
2. Enable the Power I^2C interrupt.
3. Implement the following code in the interrupt handler:

```
IF (RCSR) & 0x4) { /* verify that the processor comes out of sleep */
      PWER &= 0x7fffffff ; /* disable RTC wakeup */
      PWRMODE = 0x7;  /* put the processor into deep sleep mode through software */
}
```

Status: No Fix

E64. POWER MANAGER: Voltage change coupled with power mode change causes the processor to be unable to wake from sleep mode.

Problem: When coupling voltage change with entry into sleep mode by setting the PWRMODE register (PWRMODE=0xB), the voltage is changed as desired and the processor enters sleep mode. But the processor can not wake up from sleep mode in this situation.

Implication: TBD

Workaround: Implement the voltage change and power mode change as separate operations.

Status: No Fix

E65. SSP: GPIO[28] and GPIO[29] may not behave normally in SSP master mode.

Problem: When using GPIO[28] and GPIO[29] both in master mode to drive SSPSFRM and SSPSCLK, the SSP port may fail to operate correctly.

Implication: TBD

Workaround: When using GPIO[28] and GPIO[29] in master mode to drive SSPSFRM and SSPSCLK,

1. Program the direction of SSPSFRM and SSPSCLK in the SSP as outputs in the GPDR0 register,
2. Select Alternate Function 3 for both GPIOs
3. Clear the SCLKDIR bit in the SSP control register 1.
4. Program GPIO[28] and GPIO[29] as inputs in the GPDR0 register. Under this condition, these signals will remain SSP outputs and drive the corresponding SSPFRM and SSPCLK output signals.

E66. SSP: SSP may hang when switching from slave mode to master mode.

Problem: The SSP's state machine can hang when switching from Clock Slave mode (SSCR1[SCLKDIR]=1) to Clock Master mode (SSCR1[SCLKDIR]=0) if SSCR1[SCFR]=0. This hang prevents the SSP from outputting a clock, frame and data for subsequent transfers in the Clock Master mode.

Implication: TBD

Workaround: After finishing the current transaction in Clock Slave mode in which SSCR1[SCLKDIR]=1 and SSCR1[SCFR]=0,

1. Write SSCR1[SCFR] = 1 (leave other SSCR1 bits the same). Do not change the settings in the external device in order to keep it driving SSPSCLK.
2. Wait for SSSR[BSY] to clear.
3. Write SSCR0[SSE] = 0 (or clear all of SSCR0 if desired).
4. Change external device settings appropriately.
5. Write new values to SSCR1, SSTO, SSPSP, SSTSA, SSRSA, SSACD and SSACDD as desired.
6. Write new values to SSCR0 and set SSE (or set SSE later in a separate write if desired) and continue.

Status: No Fix

E67. POWER MANAGER: Pins selected in the Keyboard Wake-Up Enable Register (PKWR) may not cause the processor to wake up from sleep mode.

Problem: If any of the keypad pins enabled in PKWR as a wake up source is held high when the processor enters sleep mode, the wake up event from enabled PKWR pins will not be triggerable until all enabled PKWR signals are released to their inactive low state. This issue could typically be caused by a key being held depressed on entry into sleep mode.

Implication: TBD

Workaround: Make sure all the PKWR wake up sources are low before entering sleep mode, or allow for other enabled wake up keypresses being masked until all enabled PKWR wake up signals are released to their low (no-keypress) value.

Status: No Fix

E68. UHC: Exiting from sleep/deep sleep mode may cause unexpected USBH2 interrupt.

Problem: When the processor is in 13 MHz mode and enters sleep or deep sleep mode, an unexpected USBH2 interrupt may occur when the processor wakes up from sleep or deep sleep mode.

Implication: TBD

Workaround: A workaround can be implemented by clearing the interrupt status bit in UHCSTAT register immediately after waking up.

```
TempReg = UHCSTAT // Read UHCSTAT register.
TempReg &= 0x0001FD80// Set the reserved bits to zero.
UHCSTAT = TempReg// Write back the value. A write of 0b1 will clear the status bit.
```

Status: No Fix

E69. UDC: USB OTG ID pin weak pullup draws 8 mA.

Problem: When setting up GPIO 41 as Alternate Function 2 Input (USB_P2_7) with UP2OCR[IDON] bit set, GPIO 41 should act as OTG ID pin. In this situation, the OTG ID pin draws 8 mA current.

Implication: While a USB OTG cable is connected 8 mA of current will be consumed. This extra power consumption may slightly reduce system battery life.

Workaround: Instead of configuring GPIO 41 for the USBOTG ID function, use GPIO 41 as a regular GPIO input (Alternate Function = 0) with an external 10 kOhm pullup resistor. This will allow detection of cable polarity, cable insertion, and cable removal while consuming reasonably small amounts of current.

Status: No Fix

E70. CLOCKS: 48 MHz GPIO output stops during a USB suspend operation or if the USB client is not enabled.

Problem: If the USB client is in suspend mode and then sees a USB reset operation the internal 48 MHz clock is briefly stopped and restarted. This clock is associated to GPIO 11 and GPIO 12 output alternate function 3 “48 MHz” operation. A USB client reset operation will therefore cause the 48 MHz clock output on either of these GPIOs pins to briefly stop.

Implication: An external device using this 48 MHz signal may see this signal stop briefly.

Workaround: None

Status: Fixed

E71. POWER MANAGER: 8uS SYS_EN deassertion during watchdog reset.

Problem: The PXA27x and PXA270 processors EMTS documents specify the SYS_EN de-assertion timing during watchdog reset should be similar to the power on reset timing of 10 ms. The actual value is closer to 8 µs.

Implication: Customers using a watchdog reset may be affected by this errata. The 8 µs de-assertion may not be long enough for power ICs and other connected devices to complete a full power cycle. If a watchdog reset occurs during a SDRAM read operation, the SDRAM may wrongly enter a Clock Suspend mode and cause memory bus contention immediately after watchdog reset. Some SDRAM devices have shown sensitivity to this issue. For such devices the default low logic level of the processors SDCKE signal immediately after watchdog reset has the effect of holding the SDRAM in Clock Suspend mode, resulting in bus contention during the first instruction fetch. The processor then continues to fetch and execute NOPs without ceasing.

Workaround: If a watchdog reset is used and the short SYS_EN de-assertion time causes a problem, use an external pulse stretching circuit to extend the SYS_EN signal de-assertion time. To implement this workaround, the time period of this stretching circuit must not exceed PSLR[SYS_DEL] - 10 ms. The default value for the SYS_DEL delay, which is configured by the Power Manager Sleep Configuration register PSLR[SYS_DEL], is 125 ms. For systems using the default 125 ms timing, stretching circuits of 10 ms to 60 ms should be more than enough to cycle SYS_EN power supplies during a watchdog reset and then allow these power supplies to ramp before the PXA27x processor internal SYS_DEL time expires.

Status: No Fix

E72. VCC_USB must be 3.0 V or greater to pass USB signal quality compliance tests.

Problem: When using VCC_USB with voltage lower than 3.0 V, USB port may fail the USB eye diagram compliance test.

Implication: TBD

Workaround: Use a VCC_USB voltage of 3.0 V or higher when USB compliance is required for the USB peripheral.

Status: No Fix

E73. MultiMediaCard Controller: MMCMD line is sensitive to capacitive loading.

Problem: The MMCMD pin signal timing may be sensitive to capacitive loading when driven by a MMC card with a weaker drive capability. The failure appears as false CRC errors during initialization when reading from the MMC card. This failure has been observed on some older MMC cards.

Implication: TBD

Workaround: Minimize the loading on MMCMD.

Status: No Fix

E74. UART: Baud rate may not be programmed correctly on back-to-back writes.

Problem: When programming the Divisor Latch registers, Low and High (DLL and DLH), with back-to-back writes, the second register write may not take effect. The result is an incorrect baud rate.

Implication: TBD

Workaround: After programming the first Divisor Latch register, read and verify it before programming the second Divisor Latch register.

Status: No Fix

E75. AC97: PCM transmit FIFO and receive FIFO may not set the FIFO error status bits.

Problem: PCM transmit FIFO overrun error may not set the FIFO error status bit in the PCM Out Status register (POSR[FIFOE]), and PCM receive FIFO overrun error may not set the FIFO error status bit in the PCM In Status register (PCMISR[FIFOE]).

Implication: TBD

Workaround: None

Status: No Fix

E76. UDC: Writing the UDCCSR0[OPC] bit during the status stage can corrupt the next transaction.

Problem: When the USB client controller (UDC) receives a zero length OUT packet at the end of a control transfer it will: (1) automatically acknowledge the packet, and (2) signal the OPC interrupt. If the host sends another SETUP packet during the time that the software writes a 1 to clear the OPC bit in its interrupt handler, the incoming SETUP packet will be corrupted.

Implication: Typically, software can write a 1 to clear the OPC before another SETUP packet arrives. In a heavily loaded system, however, this issue can occur and the UDC may not pass a valid setup command to the software.

Workaround: Minimize the latency in handling the OPC interrupt. For example, in Windows CE this can be done by moving the handling of the interrupt from the interrupt service thread to the interrupt service routine. By moving the code to handle only this specific UDC case from the interrupt service thread into the interrupt service routine, the latency can be reduced sufficiently to avoid the failure.

Status: No Fix

E77. CI: Quick Capture Interface captures data before line valid signal occurs.

Problem: Quick Capture Interface (CI) captures incomplete or incorrect frames.

Implication: Enabling the CI in the middle of a frame may cause the CI to capture data at the wrong time, causing incomplete or incorrect frames to be captured.

Workaround: Enable the CI first and then enable the sensor. When switching video modes, it is necessary to reset the sensor. To resynchronize the CI to the sensor, wait for the current frame to finish before resetting and enabling the CI.

Status: No Fix

E78. MSL: Unexpected BBFREQ operation in 13M mode.

Problem: The baseband transmit clock frequency equals the interface clock (48-MHz) divided by the decimal value of the BBFREQ[DIV] field in any run/turbo mode except 13M mode. It operates at 13 MHz when the processor enters 13M mode. Furthermore, writing the BBFREQ register while the processor is in 13M mode will hang the baseband unit.

Implication: TBD

Workaround: Do not modify BBFREQ register while operating in 13M mode.

Status: No Fix

E79. KBD: Keypad interrupt missing after wakeup from Standby mode.

Problem: After placing the processor in Standby mode and initiating a keypress, the keypad expects to see the MKOUT pins de-assert twice in order to register a correct debounce sequence. MKOUT0 and MKOUT1 lines are both held at a logic high while coming out of Standby mode, instead of being in a de-asserted state immediately after waking up from Standby mode. This condition causes a malformed scan. The keypad controller is able to see a properly formed key press pattern in the 2nd scan. But, because there are differences between the 1st and 2nd scans, the keypad controller does not detect a valid debounced keypress. Without a valid debounced keypress, the keypad controller does not generate the desired interrupt request.

Implication: TBD

Workaround: To implement the workaround, follow these steps:

1. Disable the keypad controller clock in the CKEN register before entering Standby mode.
2. Enter Standby mode.
3. Wake up from Standby mode using a keypress or other wake up sources.
4. Enable keypad controller clock in the CKEN register, if a key was pressed to cause the Standby wakeup the key will be read.

Status: No Fix.

E80. CLOCKS MANAGER: Possible DMA errors when exiting 13M mode to a frequency where the memory controller clock (CLK_MEM) frequency does not equal the system bus frequency.

Problem: Exiting from 13M mode (CCCR[CPDIS]=1, CCCR[PPDIS]= 0 or 1) to a frequency where the system bus frequency is not equal to the memory controller clock frequency may cause DMA to transfer erroneous data.

Implication: TBD.

Workaround: When exiting from 13M mode to a frequency where the system bus and memory clock *are not equal* frequencies, an intermediate frequency change is needed where the system clock and memory clock *are equal*.

To implement the workaround, follow these steps:

1. Save current value of the CCCR register for later use.
2. Reconfigure the SDRAM and LCD timings prior to exiting 13M mode.
3. Set the PLL Early Enable bit CCCR[PLL_EARLY_EN].
4. Write the new value to CCCR, specifically CCCR[L] = 8, CCCR[A] = 1.
   a. Clear the least significant bit of CCCR[2N].
   b. Keep all other bits the same as the desired frequency.
   c. Read CCCR register back.
5. Wait for the CCSR[CPLCK] and CCSR[PPLCK] bits to be set.
6. Clear CCCR[CPDIS] and CCCR[PPDIS] bits and read CCCR register back.
7. Initiate a frequency change by setting the CLKCFG[F] bit.
8. Restore the original value of the CCCR register except for the CCCR[CPDIS] and CCCR[PPDIS] bits, and read CCCR register back.
9. Initiate final frequency change by setting the CLKCFG[F] bit.

Status: No Fix

E81. GPIO: GPIO glitch during power up

Problem: Following a power on reset, when nRESET_OUT de-asserts, a glitch of less than 20 ns may be seen on GPIO pins.

Implication: TBD

Workaround: If external logic is sensitive to a glitch, filter GPIO pins that will be configured as outputs.

Status: No Fix

E82. I^2C: I^2C may hang if an external device holds SCL low without first deasserting SDA.

Problem: The I^2C bus is susceptible to hanging if non-compliant external devices violate the I^2C protocol while the I^2C bus is idle. Specifically a hang condition may occur if the following sequence of events occurs:

1. The I^2C bus is configured as a master device and the bus is in its idle (no activity on the bus) state.

2. A non-compliant external device violates I^2C bus protocol by driving the SCL line low without driving the SDA line low first.
3. If the I^2C samples the SCL line on three consecutive rising edges of the I^2C clock, the I^2C glitch suppression logic will latch in the SCL value. Since the I^2C is in master mode and since there is a feedback loop that connects the I^2C output to the I^2C input, the I^2C master will take over the bus and hold the SCL line low indefinitely even if the external source stops driving the SCL line.

Implication: Devices non-compliant to the I^2C bus specification can cause the PX27x to hang the I^2C bus.

Workaround: None

Status: No Fix

E83. SSP: SSP1 generates a spurious Frame when switching from Frame Slave to Frame Master.

Problem: When switching SSP1 from Frame Slave to Frame Master mode, a spurious Frame signal on SSPSFRM is generated. This spurious Frame signal is only seen on SSP1, when GPIO24 is configured as SSPSFRM.

Implication: TBD

Workaround: Use GPIO28 as SSPSFRM (Alt Function Input 3); Or if GPIO24 is configured for SSPSFRM, then the following workaround can be used:

1. Finish the current data transfer in Frame Slave mode.
2. Set up SSP1 parameters for Frame Master mode for next data transfer. Do not enable SSP1 port or external device to begin data transfer.
3. Enable SSP1 port (Set SSCR0_x[SSE] bit). Then, immediately disable SSP1 port. This generates the spurious Frame signal before any data transfer takes place.
4. Enable SSP1 port and external device to begin data transfer.

Status: No Fix

E84. MEMC: Data abort on access to disabled SDRAM.

Problem: If an 8-bit or 16-bit write is attempted to a disabled SDRAM partition, a Data Abort occurs instead of the specified extra CBR refresh.

Implication: A Data Abort occurs and must be handled. This action can slow performance.

Workaround: Do not access disabled SDRAM partitions. An alternate workaround is to handle the Data Abort exception generated by this type of unusual access in the Data Abort exception handler.

Status: No Fix

E85. UART: TX data corruption when filling an empty FIFO.

Problem: Placing data into an empty UART TX FIFO can result in the loss of the first data byte and the transmission of a zero byte instead. This errata occurs infrequently and is generally observed at VCC_CORE voltages less than or equal to 0.9 V. It is primarily seen on the BTUART and is much less likely to occur on the FFUART. This errata has not been identified as occurring on the STUART. The probability of this errata occurring is determined by the number of times the transmit FIFO empties during normal

operation, the silicon speed path variation from chip to chip, and the internal clock skews following a power up, exit from sleep mode, or exit from 13 MHz mode (CCCR[CPDIS]=1, CCCR[PPDIS]=1).

Implication: TBD

Workaround A: Use hardware flow control and temporarily set the baud divisor to 0.

When software or DMA writes to an empty UART TX FIFO, set the baud divisor (DLL and DLH registers) to 0. This action causes all TX/RX operation to pause, allowing a TX FIFO load to occur without data corruption. Workaround A relies on software manipulating the PXA27x hardware flow control signals to pause the incoming receive data while the baud divisor is 0.

Setup

1. Set the transmit FIFO service/interrupt level to request service when the FIFO is half empty (<32 bytes remaining in the transmit FIFO). This is accomplished by FCR[TIL] = 0.
2. Set the FIFO control register to operate in 32-bit mode. FCR[BUS] = 1. In this mode the transmit FIFO has more elements to send between the time the UART requests service and the time the transmit FIFO actually becomes empty.
3. If using DMA, set the DCMD[SIZE] = 0b11 for the DMA Channel being used.
4. Enable auto-HW flow control MCR[AFE] = 1, MCR[RTS] = 1

Filling the transmit FIFO

1. Check the LSR[TEMT] bit. If the transmit FIFO is not empty, no workaround is required.
2. Set MCR[RTS] = 0. This setting stops auto RTS mode and de-asserts the RTS pin. When RTS is de-asserted, the device transmitting data is required to stop sending data. Many devices send at most, one more character; however, some devices require a longer amount of time to complete their transmission.
3. Wait for receive data to be completely received (usually one packet time, including start, stop, and parity). Some devices require a longer delay.
4. Disable UART by setting IER[UUE]=0
5. Set LCR[DLAB] =1 to enable access to divisor registers
6. Set the DLL[DLL] and DLH[DLH] baud divisor to 0. If both registers need to be changed, after programming the first divisor latch register, read and verify it before programming the second divisor latch register.
7. Set LCR[DLAB] =0 to restore normal register access
8. Re-Enable UART by setting IER[UUE]=1
9. Confirm LCR[DLAB] =0 to ensure the split transaction write has completed before proceeding to Step 10
10. Do one of the following as appropriate:
    a. *If using programmed IO*: Load the transmit FIFO. Check the LSR[TEMT] bit = 0 to indicate data has arrived in the FIFO. Loading more bytes improves performance and generally reduces the number of times the transmit FIFO will reach its empty state.

b. *If using DMA:* Read the DCMD[LEN] value for the DMA channel used for the UART. Set the run bit in the descriptor. Wait for de-assertion of the LSR[TEMT] bit to indicate that data has been placed into the transmit FIFO. DMA then transfers 64 bytes into the DMA FIFO if DCMD[LEN] >=64 or fewer bytes if DCMD[LEN] <64. Wait for this DMA transfer to complete by polling the DCM[LEN] value. When this transfer is complete, go to Step 11.

11. Disable UART by setting IER[UUE]=0
12. Set LCR[DLAB] =1 to enable access to divisor registers
13. Restore the DLL[DLL] and DLH[DLH] baud divisors. If both registers need to be changed, after programming the first divisor latch register, read and verify it before programming the second divisor latch register.
14. Set LCR[DLAB] =0 to restore normal register access
15. Re-Enable UART by setting IER[UUE]=1
16. Confirm LCR[DLAB] =0 to ensure the split transaction write has completed before proceeding to Step 17
17. Restore RTS to auto flow control. MCR[RTS] = 1.

Workaround B: Use 2 UART Ports and temporarily set the baud divisor to 0

If two UARTs can be used to separate the RX and TX operation, Workaround A can be applied to the TX UART without using steps 2, 3, and 17. This workaround works for all flow control and non flow control scenarios.

Workaround C: Disable UART FIFOs

Setting the FCR[TRFIFOE] bit to 0 disables the UART FIFOs and completely prevents this errata at the expense of some UART performance and increased software overhead.

Status: No Fix

E86. POWER MANAGER: GPIO83 wake-up is level-triggered, not edge-triggered.

Problem: When GPIO83 is programmed as a PWER wake-up source, GPIO83 reacts to input signals with level sensitivity instead of edge sensitivity. Thus, if GPIO83 is being driven high before going into sleep, the processor immediately wakes up from the sleep event even though no rising edge event occurs.

Implication: TBD

Workaround: None

Status: No Fix

E87. SD/MMC: Streaming write (command 20) and streaming read (command 11) transfer one block.

Problem: When the SD/MMC Controller performs a streaming write (command 20) or a streaming read (command 11), the SD/MMC controller sends one block. This errata does not affect the behavior of multiple block write (command 25) or multiple block read (command 18) requests.

Implication: Streaming block transfers require a stop command, and a new streaming write or streaming read command, after every block is transferred.

Workaround A: When using streaming IO, use the following process:

1. Send a streaming write (command 20) or a streaming read (command 11) at the start of each block.
2. Send a stop command (command 12) at the end of each block.

Repeat Steps 1 and 2 until all blocks are transferred.

Workaround B: Use multiple block write (command 25) in place of streaming write (command 20). Use multiple block read (command 18) in place of streaming read (command 11).

Status: No Fix

E88. SDRAM read and write errors while modifying MDREFR[KxDB2] if DMA is running

Problem: The SDRAM memory controller may under rare circumstances read or write incorrect data when either or both the MDREFR[K1DB2] and MDREFR[K2DB2] bits are set or cleared. This infrequent issue is only possible if DMA is running simultaneously and PXA27x internal logic signal edges align coincidentally.

Implication: Software executing from SDRAM may hang, DMA data may be read or written incorrectly.

Workaround: This workaround is an enhancement of the E61 workaround. When this workaround is applied, the workaround for E61 does not need to be applied.

Follow these steps from internal SRAM or from a function that has been pre-loaded into the cache.

To set or clear either or both MDREFR[KxDB2] bits:

1. Disable interrupts.
2. Read in the MDREFR register. Leave the KxDB2 bits at their current state, but set MDREFR[DRI] to 0xFFF and MDREFR[APD]= 0. Write these modified bits back to the MDREFR register.
3. Read back the MDREFR register one time.
4. Perform 50 SDRAM dummy reads from an uncached SDRAM address area.
5. Modify the MDREFR[KxDB2] bits (set or clear them), set the correct DRI value, and if using MDRREF[APD] re-enable MDRREF[APD] = 1 in a temporary variable. Write this new modified register value to MDREFR.
6. Read back the MDREFR register one time.
7. Perform an additional two SDRAM dummy reads from an uncached SDRAM address area.
8. Enable interrupts

Note: The dummy reads in steps 4 and 7 of this workaround may be corrupted and should not be left in the cache. To avoid data corruption use either of the two options below:

1. Perform the dummy reads from an uncached address
2. Clean and invalidate the data cache before the workaround and invalidate the data cache after the workaround.

Status: No Fix

E89. Limitations on combining core voltage changes with core frequency changes

Problem: Combining frequency changes with voltage changes by setting PCFR[FVC]=1 may cause FIFO overruns or underruns in some peripherals under certain conditions. If processing is paused for too long during this frequency and voltage change, due to extensive communication on the PWR_I2C bus, FIFOs in peripherals such as the AC97 controller may overrun or underrun when operating at high data transmission rates.

Implication: TBD

Workaround: If this problem identified on a specific hardware platform, setting the frequency and voltage should be performed as separate operations (PCFR[FVC]=0), at least when such high peripheral data rates are present.

Note: During any VCC_CORE voltage change, decrease the core operating frequency before decreasing the core voltage and, conversely, increase the core voltage before increasing the core operating frequency.

Status: No Fix

E90. MMC controller sending first FIFO byte twice

Problem: During a write data transfer where a MMC_TXFIFO swap occurs, if MMCLK is on due to a command-response sequence on the CMD line, the MMC controller may send the first FIFO byte twice before switching to the other MMC_TXFIFO.

Implication: TBD

Workaround: If a command/response sequence to a card has been started, wait until it completes before writing any data to the MMC_TXFIFO (including by DMA). If a data transfer operation to a card has been started (including the writing of any data to the MMC_TXFIFO), wait until it completes before beginning a command/response sequence.

Status: No Fix

E91. UART: TX interrupt can be missed when running full duplex

Problem: The TX FIFO empty interrupt may under rare circumstances be missed when running at full duplex mode. This situation occurs when a higher priority interrupt like RX or Character Timeout occur together with the TX FIFO empty interrupt, and de-assert just before the IIR register is read. This issue occurs at higher baud rates and is rarely found for baud rates below 115200 bps. UART FIFO DMA mode operation is not affected by this issue.

Implication: UART transmission could stall if this issue occurs.

Workaround: Monitor the TX interrupt status after each write to TX FIFO. If no TX FIFO empty interrupt occur after a certain time (several milliseconds for example), use software to emulate a TX interrupt and manually call the TX interrupt handler. There should not be too much performance overhead since the monitoring of TX interrupts can use system timers and events instead of polling.

Status: No Fix

E92. Caches, TLB (Translation Lookaside Buffer) and BTB (Branch Target Buffer) may become corrupted in Standby mode

Problem: On a small number of PXA27x processors cache, TLB (Translation Lookaside Buffer) and BTB (branch target buffer) contents may be corrupted while the processor is in standby mode.

Implication: Data corruption or software crashes may occcur following the exit from standby mode.

Workaround: **Workaround #1:**

1.Clear PCFR[OPDE] bit (leave 13 MHz oscillator on)

2.Disable Instruction/Data cache & BTB;

3.Flush Data cache; Drain write buffer.

4.Invalidate Instruction/Data cache & Instruction/Data TLB

5.Enter standby...

6.On exit from standby...

a.Invalidate Instruction/Data cache & BTB

b.Enable I/D cache & BTB

Workaround #2:

1.To safely enter standby mode ...

a.Disable Instruction/Data caches and the BTB

b.Flush the Data cache

c.Drain the write buffer; Invalidate Instruction/Data caches and I/D TLBs.

d.Execute code to enter standby from SRAM

2.After exiting standby mode ...

a.Wait 10 ms

b.Invalidate Instruction/Data cache and the BTB

c.Re-enable Instruction/Data cache and the BTB

NOTE: Step 1.d & 2.a, 2.b, 2.c should run from internal SRAM

Status: No Fix

E93. USB Host Controller Registers are not reset to their default values after sleep/deep-sleep wakeup.

Problem: When the processor exits either sleep or deep-sleep, the USB configuration settings may be different than expected. This is due to the processors' USB clock being disabled prior to entry into either sleep or deep-sleep.

Implication: The USB is not programmed for operation after exiting sleep or deep-sleep for any stepping of the PXA270 or PXA270M, and needs to be programmed under all circum-stances, to an expected or desired state upon exit from sleep or deep-sleep.

Workaround: Program the USB configuration registers to desired values after exiting sleep or deep-sleep.

Status: No Fix

E94. USB OTG over-current pin can cause erroneous shutdown in the USB Host.

Problem: PXA270M USB OTG [Host Port 2] USBHPWR<2> pin can create erroneous over-current states resulting in the shutdown of the USB Host Port 2. USBHPWR<2> is an alternate function on GPIO<119>, this pin is not bonded out on PXA270 & PXA270M 13x13 discrete packages. The pin is internal to the package.

Implication: The over-current status (indicated by UHCRHS[OCI] = 0b1) will force the USB host port 2 to stop. Software is unable to avoid this condition.

Workaround: As described, GPIO<119> is not bonded out to the package balls. Marvell will apply a fix to the package substrate to workaround this issue. This fix will tie GPIO<119> to the VCC_IO rail.

The updated package will be identified by the assembly date code, which will be provided in a future product update.

NOTE: It is critical that software must NOT drive GPIO<119> low by using the GPCRx register. This will result in excess current draw due to the pin being tied high at the package but driven low by the GPIO pad.

Also, when the GPSR (which allows programmability to the GPIOs during Sleep Mode) is used, software must again not drive GPIO<119> low which would also result in excess current draw.

If global over-current-protection is enabled by clearing UHCRHDA[OCPM] = 0b0, then the incorrect state of GPIO<119> will force all ports to be shutdown. Software must ensure that OCPM = 0b1, this will allow USB Host Port 1 & 3 to be used as usual.

Status: Planned fix within the package. No silicon change.

E95. Rapid power cycling may cause a processor hang

Problem: The processor may hang during a power-on cycle if VCC_BATT does not go down to 0V after the voltage level drops to or below 2.4V on VCC_BATT.

Implication: The processor may hang in a test-case scenario where the processor is (1) power cycled on and off repeatedly, (2) the VCC_BATT power domain drops below 2.4V (minimum spec), and (3) does not go down to 0V, .

Workaround: For the failing scenario, the on and off power cycling must be close enough so that the VCC_BATT rail would never drain to 0V but must drop VCC_BATT to below 2.4V.

This errata affects any use case where the VCC_BATT supply drops below the min spec but does not fall to 0V before the next power on (re-enabling of VCC_BATT).

Toggling the nRESET pin to force a hardware reset does not reset the processor once this hang has occurred. If VCC_BATT goes below 2.4V at any time, then this rail must go down to 0V before VCC_BATT is powered back on.

Status: No Fix

Specification Changes

S8. Extended Input DC operating conditions for USB interfaces

Table 5-8 of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and Table 5-7 of the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* show the DC operating conditions for the I/O pins of the processor. Remove VCC_USB from standard VIH list in Table 5-8 of the PXA270 processor EMTS and add an extra row for VCC_USB for both tables. The VCC_USB power domain's VIH is extended to a maximum of 3.6 V.

Table 5-8 of the PXA270 EMTS currently states:

Table 5-8. Standard Input, Output, and I/O Pin DC Operating Conditions

Symbol	Description	Min	Max	Units	Testing Conditions / Notes
Input DC Operating Conditions (VCC = 1.8V, 2.5, 3.0, 3.3 Typical)					
VIH[1]	Input high voltage, all standard input and I/O pins, relative to applicable VCC (VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, VCC_USB or VCC_USIM)	0.8 * VCC	VCC + 0.1	V	—

It should state:

Table 5-8. Standard Input, Output, and I/O Pin DC Operating Conditions

Symbol	Description	Min	Max	Units	Testing Conditions / Notes
Input DC Operating Conditions (VCC = 1.8V, 2.5, 3.0, 3.3 Typical)					
VIH[1]	Input high voltage, all standard input and I/O pins, relative to applicable VCC (VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, or VCC_USIM)	0.8 * VCC	VCC + 0.1	V	—
VIH_USB	Input high voltage for the USB bus voltage domain (VCC_USB)	0.8 * VCC	3.6	V	—

Table 5-7 of the PXA27x processor EMTS currently states:

Table 5-7. Standard Input, Output, and I/O Pin DC Operating Conditions

Symbol	Description	Min	Max	Units	Testing Conditions / Notes
Input DC Operating Conditions (VCC = 1.8V, 2.5, 3.0, 3.3 Typical)					
VIH[1]	Input high voltage, all standard input and I/O pins, relative to applicable VCC (VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, or VCC_USIM)	0.8 * VCC	VCC + 0.1	V	—

It should state:

Table 5-7. Standard Input, Output, and I/O Pin DC Operating Conditions

Symbol	Description	Min	Max	Units	Testing Conditions / Notes
Input DC Operating Conditions (VCC = 1.8V, 2.5, 3.0, 3.3 Typical)					
VIH[1]	Input high voltage, all standard input and I/O pins, relative to applicable VCC (VCC_IO, VCC_MEM, VCC_BB, VCC_LCD, or VCC_USIM)	0.8 * VCC	VCC + 0.1	V	—
VIH_USB	Input high voltage for the USB bus voltage domain (VCC_USB)	0.8 * VCC	3.6	V	—

S9. 32.768-kHz crystal load capacitance specification updated

Table 5-9 of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and Table 5-8 of the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* list the 32.768-kHz crystal specifications. The Load capacitance (C_L) specification should be updated for both tables.

The Load capacitance of Table 5-9 in the PXA270 EMTS currently states:

Table 5-9. Typical 32.768-kHz Crystal Requirements

Parameter	Minimum	Typical	Maximum	Units
Load capacitance (C_L)	—	12.5	—	pf

It should state:

Table 5-9. Typical 32.768-kHz Crystal Requirements

Parameter	Minimum	Typical	Maximum	Units
Load capacitance (C_L)	6	7.5	12.5	pf

Note: The same changes should be made to Table 5-8 of the PXA27x EMTS.

S10. SDRAM 3.3 V Interface AC Timing Specification Updated

Table 6-15. SDRAM Interface AC Specifications of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* shows the timing parameters of SDRAM. The tsdSDOS and tsdSDOH parameters are now updated.

It currently states:

Table 6-15. SDRAM Interface AC Specifications

Symbols	Parameters	VCC_MEM = 1.8V +20% / –5%[3]			VCC_MEM = 2.5V +/- 10%[4]			VCC_MEM = 3.3V +/- 10%[5]			Units	Notes
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
tsdSDOS	MA<24:10>, MD<31:0>, DQM<3:0>, nSDCS<3:0>, nSDRAS, nSDCAS, nWE, nOE, SDCKE1, RDnWR output setup time to SDCLK<2:1> rise	2.5	—	—	2.5	—		—	—	—	ns	—
tsdSDOH	MA<24:10>, MD<31:0>, DQM<3:0>, nSDCS<3:0>, nSDRAS, nSDCAS, nWE, nOE, SDCKE1, RDnWR output hold time from SDCLK<2:1> rise	1.5	—	—	1.5	—		—	—	—	ns	—
		VCC_CORE = 0.85 V +/– 10%, with 1.71 V<= VCC_MEM <= 3.63 V			VCC_CORE = 1.1 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V			VCC_CORE = 1.3 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V				

It should state:

Table 6-15. SDRAM Interface AC Specifications

Symbols	Parameters	VCC_MEM = 1.8V +20% / –5%[3]			VCC_MEM = 2.5V +/- 10%[4]			VCC_MEM = 3.3V +/- 10%[5]			Units	Notes
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
tsdSDOS	MA<24:10>, MD<31:0>, DQM<3:0>, nSDCS<3:0>, nSDRAS, nSDCAS, nWE, nOE, SDCKE1, RDnWR output setup time to SDCLK<2:1> rise	2.5	—	—	2.5	—		2.5	—	—	ns	—
tsdSDOH	MA<24:10>, MD<31:0>, DQM<3:0>, nSDCS<3:0>, nSDRAS, nSDCAS, nWE, nOE, SDCKE1, RDnWR output hold time from SDCLK<2:1> rise	1.5	—	—	1.5	—		1.5	—	—	ns	—
		VCC_CORE = 0.85 V +/– 10%, with 1.71 V<= VCC_MEM <= 3.63 V			VCC_CORE = 1.1 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V			VCC_CORE = 1.3 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V				

S11. Synchronous Flash AC Timing Specification Updated

Table 6-17. Synchronous Flash Read AC Specifications of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* shows the timing parameters of synchronous flash.

1. The *MIN* values of tffSDOS, tffSDOH, tffSDIS[1] and tffSDIH parameters are now updated.
2. MA<25:0> is removed from the tffSDOH category because the address bus is maintained for the entire flash transaction.
3. The *Parameters* categories of tffSDOS, tffSDOH, tffSDIS and tffSDIH parameters incorrectly list SDCLK<2:1> as clocks for synchronous flash. They should be updated.

It currently states:

1. tffSDIS was updated further as of July 2006.

Table 6-17. Synchronous Flash Read AC Specifications

Symbols	Parameters	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	Units	Notes
		VCC_MEM = 1.8V +20% / -5%[6]			VCC_MEM = 2.5V +/- 10%[7]			VCC_MEM = 3.3V +/- 10%[8]				
tffSDOS	MA<25:0>, MD<31:0>, DQM<3:0>, nCS<3:0>, nSDCAS (nADV), nWE, nOE, RDnWR output setup time to SDCLK<2:1> rise	TBD	—	—	TBD	—	—	TBD	—	—	ns	—
tffSDOH	MA<25:0>, MD<31:0>, DQM<3:0>, nCS<3:0>, nSDCAS (nADV), nWE, nOE, RDnWR output hold time from SDCLK<2:1> rise	TBD	—	—	TBD	—	—	TBD	—	—	ns	—
		VCC_CORE = 0.85 V +/– 10%, with 1.71 V<= VCC_MEM <= 3.63 V			VCC_CORE = 1.1 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V			VCC_CORE = 1.3 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V				
tffSDIS	MD<31:0> read data input setup time from SDCLK<2:0> rise	TBD	—	—	0.5	—	—	0.5	—	—	ns	—
tffSDIH	MD<31:0> read data input hold time from SDCLK<2:0> rise	TBD	—	—	1.8	—	—	1.8	—	—	ns	—

It should state:

Table 6-17. Synchronous Flash Read AC Specifications

Symbols	Parameters	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	Units	Notes
		VCC_MEM = 1.8V +20% / -5%[6]			VCC_MEM = 2.5V +/- 10%[7]			VCC_MEM = 3.3V +/- 10%[8]				
tffSDOS	MA<25:0>, MD<31:0>, DQM<3:0>, nCS<3:0>, nSDCAS (nADV), nWE, nOE, RDnWR output setup time to SDCLK0 rise	8	—	—	8	—	—	8	—	—	ns	—
tffSDOH	MD<31:0>, DQM<3:0>, nCS<3:0>, nSDCAS (nADV), nWE, nOE, RDnWR output hold time from SDCLK0 rise	4.5	—	—	4.5	—	—	4.5	—	—	ns	—
		VCC_CORE = 0.85 V +/– 10%, with 1.71 V<= VCC_MEM <= 3.63 V			VCC_CORE = 1.1 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V			VCC_CORE = 1.3 V +/– 10%, with 1.71 V <= VCC_MEM <= 3.63 V				
tffSDIS	MD<31:0> read data input setup time from SDCLK0 rise	2.2	—	—	2.2	—	—	2.2	—	—	ns	—
tffSDIH	MD<31:0> read data input hold time from SDCLK0 rise	2.9	—	—	2.9	—	—	2.9	—	—	ns	—

S12. Quick Capture Interface AC Timing Added

Section 6.8 Quick Capture Interface AC Timing should be added to the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification*.

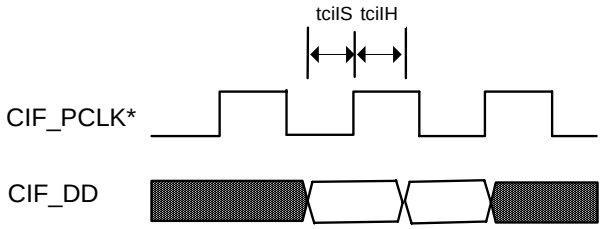
6.8 Quick Capture Interface AC Timing

Table 6-27 lists the timing parameters used in Figure 6-33.

Table 6-27. Quick Capture AC Timing Specification

Symbol	Description	Min	Typical	Max	Units
tciIS	Camera Interface Setup Time	5	—	—	ns
tciIH	Camera Interface Hold Time	5	—	—	ns

Figure 6-33. Quick Capture Interface Timing



* CIF_PCLK Data Sampling edge determined by the CICR 4[PCP] setting

S13. SD/MMC: Remove section 15.8.4.5 “Stop Data Transmission, Randomly”

Section 15.8.4.5 Stop Data Transmission, Randomly of the *PXA27x Processor Family Developer’s Manual* should be removed because the MMC_STRPCL[STOP_CLK_CMD12] bit is no longer supported and cannot be used to stop the MMC clock.

S14. 32.768-kHz crystal equivalent series resistance requirements correction

In Table 5-9 Typical 32.768-kHz Crystal Requirements of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification*, the maximum value of equivalent series resistance (R_S) should be changed from 35 kW to 65 kW.

S15. LCD controller timing specification updated

In Table 6-22 LCD Timing Specifications of the *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification* and *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification*, the maximum values of Tpclkdv, Tpclklv and Tpclkfv should be updated. The associated Figure 6-28 LCD Timing Definitions should be updated to correct the timing relations relative to the LCCR3[PCP] setting.

Table 6-22 currently states:

Table 6-22. LCD Timing Specifications

Symbol	Description	Min	Max	Units	Notes
Tpclkdv	L_PCLK_WR rise/fall to L_LDD<17:0> driven valid	—	14	ns	1
Tpclklv	L_PCLK_WR fall to L_LCLK_A0 driven valid	—	14	ns	2
Tpclkfv	L_PCLK_WR fall to L_FCLK_RD driven valid	—	14	ns	2
Tpclkbv	L_PCLK_WR rise to L_BIAS driven valid	—	14	ns	2

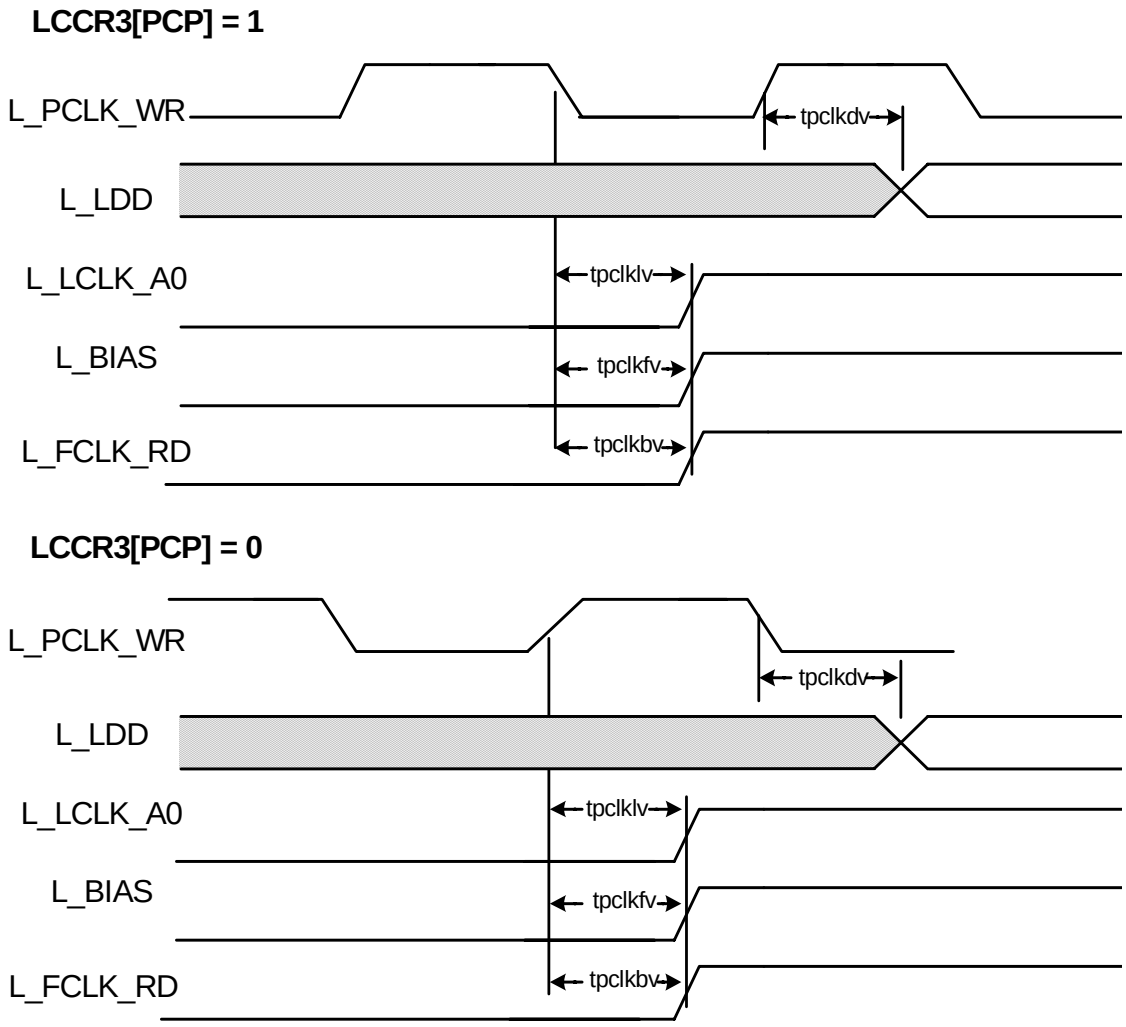
It should state:

Table 6-22. LCD Timing Specifications

Symbol	Description	Min	Max	Units	Notes
Tpclkdv	L_PCLK_WR rise/fall to L_LDD<17:0> driven valid	—	7	ns	1
Tpclklv	L_PCLK_WR fall to L_LCLK_A0 driven valid	—	7	ns	2
Tpclkfv	L_PCLK_WR fall to L_FCLK_RD driven valid	—	7	ns	2
Tpclkbv	L_PCLK_WR rise to L_BIAS driven valid	—	14	ns	2

Figure 6-28 should be replaced with:

Figure 6-28. LCD Timing Definitions



S16. PXA27x package information updated

Section 3.1 Package Information of the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* should be updated with the following changes:

The first paragraph should be updated. It currently states:

The PXA27x Processor Family has the following characteristics:

- Ball pitch: 0.65mm
- Ball diameter: 0.30 mm

- Substrate thickness: 0.21 mm
- Mold thickness: 0.45 mm

It should state:

The PXA27x processor family has the following characteristics:

- Ball pitch: 0.65 mm
- Ball diameter: 0.35 mm

Table 3-1 PXA27x Processor Family Package Information should be removed. Table 3-1 PXA271 Processor Family Package Information and Table 3-2 PXA272 Processor Family Package Information should be added:

Table 3-1. PXA271 Processor Package Information

		Millimeters		
	Symbol	Min	Nom	Max
Pacakge Height	A			1.550
Ball Height	A1	0.180		0.280
Package Body Thickness	A2	1.121	1.158	1.195
Ball (Lead) Width	b	0.300	0.400	0.450
Bottom Package Body Width	D	13.925	14.000	14.075
Bottom Package Body Length	E	13.925	14.000	14.108
Top Package Body Width	F	10.950	11.000	11.050
Top Package Body Length	G	12.950	13.000	13.050
Pitch	[e]		0.650	
Ball (Lead) Count	N		336	
Seating Plane Coplanarity	Y			0.150
Corner to Ball A1 Distance Along D	S1		0.825	
Corner to Ball A1 Distance Along E	S2		0.825	

Table 3-2. PXA272 Processor Package Information

		Millimeters		
	Symbol	Min	Nom	Max
Pacakge Height	A			1.550
Ball Height	A1	0.180		0.280
Package Body Thickness	A2	1.121	1.158	1.195
Ball (Lead) Width	b	0.300	0.400	0.450
Bottom Package Body Width	D	13.925	14.000	14.075
Bottom Package Body Length	E	13.925	14.000	14.075
Top Package Body Width	F	10.950	11.000	11.050
Top Package Body Length	G	7.950	8.000	8.050
Pitch	[e]		0.650	
Ball (Lead) Count	N		336	
Seating Plane Coplanarity	Y			0.150
Corner to Ball A1 Distance Along D	S1		0.825	

Specification Clarifications

S1. I/O Pin Drive Strength

Issue: I/O pin drive strength and internal resistive pu/pd values may vary

Problem: The PXA270M padring drivers and strengths of pullup/pulldown resistors are different across process voltage and temperature (PVT) when compared with the PXA270 processor. Some pins may need to be configured differently when migrating from PxA270 to PxA270M, such as programmable drive strengths may need to be tuned or external pullup/pulldown resistors may have to be changed or added.

Implication: Using the same drive strength configuration on a PXA270M platform may not allow the system to operate correctly using the same configuration for drive strengths as the PXA270. Relying on an internal pullup or pulldown resistor at the system level may have different results between teh PXA270 and PXA270M processors. Some hardware and/or software changes may be required depending on the application.

Workaround: Software may have to be updated or a change may be necessary to the configuration of the drive strength adjustment registers. The PXA270/PXA270M both have the following programmable drive strengths:

1. LCD Buffer Strength Control LCDBSCNTR
2. SDRAM Buffer Strength Control BSCNTRx

Instances may arise where component changes may be required such as the addition or removal of discrete components (resistors or crystals) due to product differences.

Documentation Changes

D22. SDRAM Timing Parameter tsdCL Description Correction

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

In *Section 6.4.2 SDRAM Parameters and Timing Diagrams*, Table 6-15. SDRAM Interface AC Specifications shows the SDRAM timing parameters. The tsdCL description is incorrect.

The Figure 6-7 SDRAM Timing diagram in the PXA270 processor EMTS and PXA27x processor EMTS incorrectly shows tsdCL = 4 and should be updated to match the 3 SDCLK maximum case. The left edge of tsdCL timing should be moved one clock to the right.

Table 6-15 currently states:

Table 6-15. SDRAM Interface AC Specifications

Symbols	Parameters	VCC_MEM = 1.8V +20% / –5%[3]			VCC_MEM = 2.5V +/- 10%[4]			VCC_MEM = 3.3V +/- 10%[5]			Units	Notes
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
tsdCL	nSDRAS to nSDCAS delay	2	MDCNFG [DTCx]	3	2	MDCNFG [DTCx]	3	2	MDCNFG [DTCx]	3	SDCLK	6

It should state:

Table 6-15. SDRAM Interface AC Specifications

Symbols	Parameters	VCC_MEM = 1.8V +20% / –5%[3]			VCC_MEM = 2.5V +/- 10%[4]			VCC_MEM = 3.3V +/- 10%[5]			Units	Notes
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX		
tsdCL	CAS Latency	2	MDCNFG [DTCx]	3	2	MDCNFG [DTCx]	3	2	MDCNFG [DTCx]	3	SDCLK	6

D27. VLIO Timing Parameter tvlioDH Correction

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

Table 6-20 VLIO Timing in the PXA270 EMTS and PXA27x EMTS incorrectly lists tvlioDH parameter values. They should be updated.

It currently states:

Table 6-20. VLIO Timing

Symbols	Parameters	MIN	TYP	MAX[2]	Units[1]	Notes
tvlioDH	MD/DQM hold from nPWE de-asserted	1	—	1	clk_mem	—

It should state:

Table 6-20. VLIO Timing

Symbols	Parameters	MIN	TYP	MAX[2]	Units[1]	Notes
tvlioDH	MD/DQM hold from nPWE de-asserted	2	MSCx[RDN]	30	clk_mem	—

D33. Derating Specifications Removed From EMTS

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

In *Section 6 AC Timing Specifications*, Table 6-1 shows the AC operating conditions. The derating specifications in this table are covered by the published PXA270 and PXA27x IBIS models. Therefore, these specifications (td_{F_H}, td_{R_H}, td_{F_L} and td_{R_L}) should be removed from this table.

It currently states:

Table 6-1. Standard Input, Output, and I/O-Pin AC Operating Conditions

Symbol	Description	Min	Typical	Max	Units
C_{IN}	Input capacitance, all standard input and I/O pins	—	—	10	pf
C_{OUT_H}	Output capacitance, all standard high-strength output and I/O pins	20	—	50	pf
td_{F_H}	Output derating, falling edge on all standard, high-strength output and I/O pins, from 50-pf load.	—	TBD	—	ns/pf
td_{R_H}	Output derating, rising edge on all standard, high-strength output and I/O pins, from 50-pf load.	—	TBD	—	ns/pf
C_{OUT_L}	Output capacitance, all standard low-strength output and I/O pins	20	—	50	pf
td_{F_L}	Output derating, falling edge on all standard, low-strength output and I/O pins, from 50-pf load.	—	TBD	—	ns/pf
td_{R_L}	Output derating, rising edge on all standard, low-strength output and I/O pins, from 50-pf load.	—	TBD	—	ns/pf

It should state:

Table 6-1. Standard Input, Output, and I/O-Pin AC Operating Conditions

Symbol	Description	Min	Typical	Max	Units
C_{IN}	Input capacitance, all standard input and I/O pins	—	—	10	pf
C_{OUT_H}	Output capacitance, all standard high-strength output and I/O pins	20	—	50	pf
C_{OUT_L}	Output capacitance, all standard low-strength output and I/O pins	20	—	50	pf

D34. Ball Diameter for 23x23mm Package Added

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

Figure 3-6 should be updated to reflect the 23x23 mm package ball diameter value, which is 0.60+/-0.10 mm.

D35. UART Maximum Baud Rate Clarification

Affected Docs: *PXA27x Processor Family Developer's Manual*

In *Section 10.4.7 Programmable Baud-Rate Generator*, the first paragraph, the following sentence should be removed:

"For the FFUART and the STUART, the divisor is from 4 to (2^{16} - 1)."

In *Section 10.4.7 Programmable Baud-Rate Generator*, The paragraph below Table 10-2. Theoretical Baud Rate vs. Actual Baud Rate should be updated to reflect the correct UART maximum baud rate.

It currently states:

The divisor's reset value is 0x0002. For the FFUART and the STUART, the divisor must be set to at least 0x0004 before the UART unit is enabled. Changing the baud rate (writing to registers DLL and DLH) is not permitted while actively transmitting or receiving data.

It should state:

The divisor's reset value is 0x0002. Changing the baud rate (writing to registers DLL and DLH) is not permitted while actively transmitting or receiving data.

D36. MMC/SD/SDIO AC Timings Added

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

In *Section 6 AC Timing Specifications*, the following two subsections about MMC/SD/SDIO AC timings are now added.

6.8 MultiMediaCard Timing Specifications

Figure 6-33. MultiMedia Card timing Diagrams

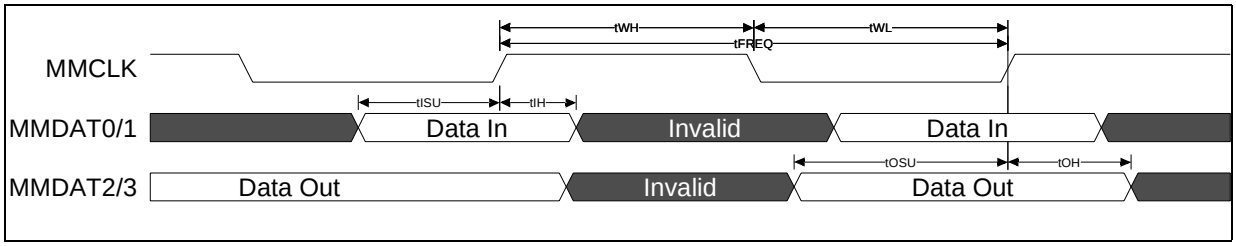


Table 6-27. MultiMedia Card timing specifications

Symbol	Parameter	Min	Max	Unit	Notes
t_{FREQ}	MMCLK Frequency Data Transfer Mode	0	19.5	MHz	
t_{FREQ}	MMCLK Frequency Identification Mode	0	400	KHz	
t_{WH}	Clock high time	10	—	ns	
t_{WL}	Clock low time	10	—	ns	
t_{rise}	Clock rise time	—	10	ns	1
t_{fall}	Clock fall time	—	10	ns	1
t_{ISU}	Data input setup time	3	—	ns	
t_{IH}	Data input hold time	3	—	ns	
t_{OSU}	Output data setup time	13.1	—	ns	
t_{OH}	Output data hold time	9.7	—	ns	

NOTE:
1. Rise and fall times measured from 10% - 90% of voltage level.

t_{WH}	Clock high time	10	—	ns	
t_{WL}	Clock low time	10	—	ns	
t_{rise}	Clock rise time	—	10	ns	1
t_{fall}	Clock fall time	—	10	ns	1

6.9 Secure Digital (SD/SDIO) Timing

Figure 6-34 and Table 6-28 define the Secure Digital (SD/SDIO) controller AC timing specifications.

Figure 6-34. SD/SDIO timing diagrams

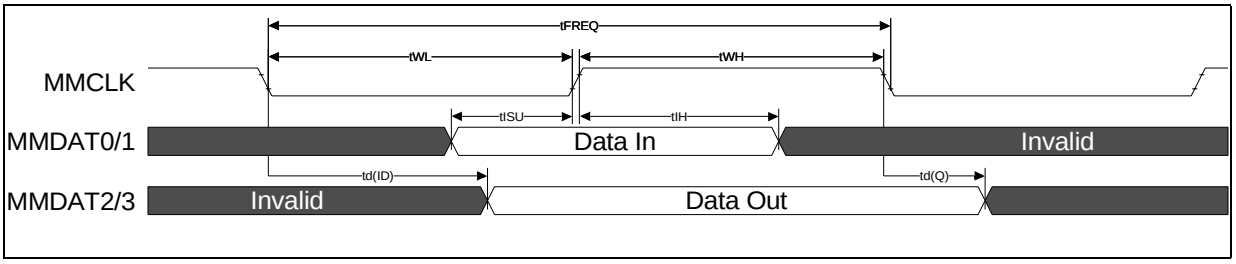


Table 6-28. SD/SDIO Timing Specifications

Symbol	Parameter	Min	Max	Unit	Notes
t_{FREQ}	MMCLK Frequency Data Transfer Mode	0	19.5	MHz	
t_{FREQ}	MMCLK Frequency Identification Mode	0[1]/100	400	KHz	
t_{WH}	Clock high time	50	—	ns	
t_{WL}	Clock low time	50	—	ns	
t_{rise}	Clock rise time	—	10	ns	2
t_{fall}	Clock fall time	—	10	ns	2
t_{ISU}	Data input setup time	5	—	ns	
t_{IH}	Data input hold time	5	—	ns	
$t_{d(Q)}$	Output Delay time during Data Transfer Mode	0	14	ns	
$t_{d(ID)}$	Output Delay time during Identification Mode	0	50	ns	

NOTES:
1. 0 KHz is when the clock is stopped. The minimum 100 KHz frequency range is where continuous clock is required.
2. Rise and fall times measured from 10% - 90% of voltage level.

D37. Idle and Low-Power Mode Maximum Power-Consumption Specifications Added

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

In *Section 5.3 Power-Consumption Specifications*, Table 5-7 Power-Consumption Specifications is renamed to Table 5-7 Typical Power-Consumption Specifications.

A new table, Table 5-8 Maximum Power-Consumption Specifications, is now added.

Table 5-8. Maximum Power-Consumption Specifications (Sheet 1 of 2)

Parameter Description	Maximum	Units	Conditions
Idle Mode Power Consumption			
624MHz Idle Current on VCC_CORE (PX=208MHz)	770	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=1.705 V, VCC_PERI[1]=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
520MHz Idle Current on VCC_CORE (PX=208MHz)	630	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=1.595 V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
416MHz Idle Current on VCC_CORE (PX=208MHz)	500	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=1.485 V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
312MHz Idle Current on VCC_CORE (PX=208MHz)	380	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=1.375 V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
208MHz Idle Current on VCC_CORE (PX=208MHz)	260	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=1.265 V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
104MHz Idle Current on VCC_CORE, (PX=104MHz)	150	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=0.99V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
Idle Current on VCC_PERI[1], All Core Speeds	200	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=any, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
Idle Current on VCC_IO, All Core Speeds	50	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=any, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
Idle Current on VCC_PLL, All Core Speeds	100	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=any, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
Deep-Idle Mode Power Consumption			
13MHz Deep Idle Current on VCC_CORE (LCD off)	105	mA	Temp=85C Tcase, VCC_CORE=VCC_SRAM=VCC_PLL=0.935 V, VCC_PERI=3.63V, VCC_IO=3.63V, VCC_BATT=3.75V
Standby Mode Power Consumption			
Standby Current on VCC_CORE	5	mA	Temp=Room, VCC_CORE=VCC_SRAM=VCC_PLL=1.1V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
Standby Current on VCC_PERI	1.6	mA	Temp=Room, VCC_CORE=VCC_SRAM=VCC_PLL=1.1V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
Standby Current on VCC_IO	1	mA	Temp=Room, VCC_CORE=VCC_SRAM=VCC_PLL=1.1V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
Sleep Mode Power Consumption			
Sleep Current on VCC_CORE	0.15	mA	Temp=Room, VCC_CORE=VCC_PLL=0V, VCC_SRAM=0.95V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
Sleep Current on VCC_PERI	0.47	mA	Temp=Room, VCC_CORE=VCC_PLL=0V, VCC_SRAM=0.95V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V

Table 5-8. Maximum Power-Consumption Specifications (Sheet 2 of 2)

Parameter Description	Maximum	Units	Conditions
Sleep Current on VCC_IO	0.70	mA	Temp=Room, VCC_CORE=VCC_PLL=0V, VCC_SRAM=0.95V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
Sleep Current on VCC_PLL	0.043	mA	Temp=Room, VCC_CORE=VCC_PLL=0V, VCC_SRAM=0.95V, VCC_PERI=1.8V, VCC_IO=VCC_BATT=3.0V
NOTE: 1) VCC_PERI = VCC_MEM + VCC_BB + VCC_USIM + VCC_LCD			

D38. MMC Read Time-Out Register (MMC_RDTO) Description Clarification

Affected Docs: *PXA27x Processor Family Developer's Manual*

In *Section 15.9.7 MMC Read Time-Out Register (MMC_RDTO)*, the definition of MMC Read Time-Out should be corrected.

It currently states:

The MMC Read Time-Out register specifies the number of clocks following the command after which the controller indicates a received-data time-out error. The units for this register is 13128 ns. For example, if MMC_RDTO[READ_TO] is set to 0b10, the controller waits 26256 ns after the response end bit for the data start; if data is not read, it reports a time-out.

It should state:

The MMC Read Time-Out register specifies the number of clocks following the command after which the controller indicates a received-data time-out error. The unit of time for this register is 13128 ns. For example, if MMC_RDTO[READ_TO] is set to 0b10, the controller waits 26256 ns after the command end bit for the data start; if data is not read, it reports a time-out.

The Read Data Time Out state machine runs until one of the following events occur:

1. Following a read data command, the controller receives the first bit of the read data.
2. Software sends a command with MMC_CMDAT[STOP_TRAN]=1. (typically CMD12).
3. The read data timeout occurs when the internal timer has waited the amount of time configured in the MMC_RDTO[READ_TO] register. Note this internal timer relies on the MMC Clock running to measure this period of time.

D39. GPIO reset nRESET_OUT and GPIO<1> pulse width correction

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

In *Section 6.2.4 GPIO Reset Timing*, the description of nRESET_OUT and GPIO<1> pulse width should be clarified and corrected with the following two changes.

The first paragraph of Section 6.2.4 currently states:

GPIO reset is generated externally, and the source is reconfigured as a standard GPIO as soon as the reset propagates internally. The clocks module is not reset by GPIO reset, so the timing varies based on the selected clock frequency. If the clocks and power manager is in a frequency-change sequence

when GPIO reset is asserted (see Section 5.5.1, “32.768-kHz Oscillator Specifications” on page 5-9.), then Figure 6-4 shows the timing of GPIO reset, and Table 6-5 shows the GPIO reset timing specifications.

It should state:

GPIO reset is generated externally, and the reset GPIO source is reconfigured as a standard GPIO as soon as the reset propagates internally. The clocks module is not reset by GPIO reset, so the reset timing varies based on the selected clock frequency. Since GPIO assertions are ignored during a frequency change sequence, if GPIO<1> is asserted during a frequency change sequence, it must remain asserted low for 240 ns after the frequency change completes for the GPIO reset to be recognized. Figure 6-4 shows the timing of GPIO reset, and Table 6-5 shows the GPIO reset timing specifications.

Table 6-5 currently states:

Table 6-5. GPIO Reset Timing Specifications

Symbol	Description	Min	Typical	Max	Units
tA_GPIO<1>	Minimum assert time of GPIO<1>[1] in 13.000-MHz input clock cycles	4[4]	—	—	cycles
tDHW_OUT_A	Delay between GPIO<1> asserted and nRESET_OUT asserted in 13.000-MHz input clock cycles	6[4]	—	8	cycles
tDHW_OUT	Delay between nRESET_OUT asserted and nRESET_OUT de-asserted, run or turbo mode[2]	230	—	—	nsec
tDHW_OUT_F	Delay between nRESET_OUT asserted and nRESET_OUT de-asserted, during frequency change sequence[3]	5	—	380	µs
tCS0[5]	Delay between nRESET_OUT de-assertion and nCS0 assertion	1000	—	—	ns

NOTES:
1. GPIO<1> is not recognized as a reset source again until configured to do so in software. Software must check the state of GPIO<1> before configuring as a reset to ensure that no spurious reset is generated. For details, see the “Clocks and Power Manager” chapter in the *PXA27x Processor Family Developer’s Manual*.
2. Time is 512*N processor clock cycles plus up to 4 cycles of the 13.000-MHz input clock.
3. Time during the frequency-change sequence depends on the state of the PLL lock detector at the assertion of GPIO reset. The lock detector has a maximum time of 350 µs plus synchronization.
4. In standby, sleep, and deep-sleep modes, this time is in addition to the wake-up time from the low-power mode.
5. The tCS0 specification is also applicable to Power-On reset, Hardware reset, Watchdog reset and Deep-Sleep/Sleep mode exit.

It should state:

Table 6-5. GPIO Reset Timing Specifications (Sheet 1 of 2)

Symbol	Description	Min	Typical	Max	Units	Notes
tA_GPIO<1>	Minimum assert time of GPIO<1> in 13.000-MHz input clock cycles	4	—	—	cycles	1, 2, 4
tDHW_OUT_A	Delay between GPIO<1> asserted and nRESET_OUT asserted in 13.000-MHz input clock cycles	6	—	8	cycles	4
tDHW_OUT	Delay between nRESET_OUT asserted and nRESET_OUT de-asserted, run or turbo mode	230	—	—	nsec	

Table 6-5. GPIO Reset Timing Specifications (Sheet 2 of 2)

Symbol	Description	Min	Typical	Max	Units	Notes
tDHW_OUT_F	Delay between nRESET_OUT asserted and nRESET_OUT de-asserted, during frequency change sequence	5	—	380	µs	3
tCS0	Delay between nRESET_OUT de-assertion and nCS0 assertion	1000	—	—	ns	5

NOTES:
1. GPIO<1> is not recognized as a reset source again until configured to do so in software. Software must check the state of GPIO<1> before configuring as a reset to ensure that no spurious reset is generated. For details, see the “Clocks and Power Manager” chapter in the *PXA27x Processor Family Developer’s Manual*.
2. If GPIO<1> reset is asserted during a frequency change sequence, the minimum assert time of GPIO<1> needs to be 512*N processor clock cycles plus up to 4 cycles of the 13.000-MHz input clock cycles in order for the reset to be recognized.
3. Time during the frequency-change sequence depends on the state of the PLL lock detector at the assertion of GPIO reset. The lock detector has a maximum time of 350 µs plus synchronization.
4. In standby, sleep, and deep-sleep modes, this time is in addition to the wake-up time from the low-power mode.
5. The tCS0 specification is also applicable to Power-On reset, Hardware reset, Watchdog reset and Deep-Sleep/Sleep mode exit.

In this table, Note 2 is updated and applied to tA_GPIO<1> instead of tDHW_OUT.

D40. nBATT_FAULT and nRESET power-on reset description correction

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

In the first note of *Section 6.2.1 Power-On Timing Specifications*, the description of nBATT_FAULT and nRESET during power-on reset is incorrect and should be removed.

It currently states:

> ***Note:*** nBATT_FAULT must be high before nRESET is de-asserted. Otherwise, the processor does not begin the power-on sequencing event. nVDD_FAULT is sampled only when the SYS_DEL and PWR_DEL timers have expired. Refer to the *PXA27x Processor Family Developer’s Manual*, “Initial Power On” and “Deep-Sleep Exit States” for a state diagram.

The sentences “nBATT_FAULT must be high before nRESET is de-asserted; otherwise, the processor does not begin the power-on sequencing event.” is incorrect and should be removed.

D41. ROM timing parameters tromAVDVF and tromAVDVS correction

Affected Docs: *PXA270 Processor Family Electrical, Mechanical, and Thermal Specification*

PXA27x Processor Family Electrical, Mechanical, and Thermal Specification

All the appearances of the ROM timing paramters tromAVDVF and tromAVDVS should be replaced with the name tromAVDLF and tromAVDLS.

The description of these two paramters in Table 6-16 should also be corrected.

It currently states:

Table 6-16. ROM AC Specification

Symbols	Parameters	MIN	TYP	MAX	Units†	Notes
tromAS	Address setup to nCS assert	1	—	1	clk_mem	—
tromCES	nCS setup to nOE asserted	—	—	0	clk_mem	—
tromCEH	nCS hold from nOE de-asserted	—	—	0	clk_mem	—
tromDSOH	MD setup to address valid	1.5	—	—	clk_mem	—
tromDOH	MD hold from address valid	0	—	—	clk_mem	—
tromAVDVF	Address valid to data valid for the first read access	2	MSCx[RDF]+2	32	clk_mem	—
tromAVDVS	Address valid to data valid for subsequent reads of non-burst devices	1	MSCx[RDF]+1	31	clk_mem	—
tflashAVDVS	Address valid to data valid for subsequent reads of burst devices	1	MSCx[RDN]+1	31	clk_mem	—
tromCD	nCS de-asserted after a read of next nCS or nSDCS asserted (minimum)	1	MSCx[RRR]*2+1	15	clk_mem	—

† Numbers shown as integer multiples of the clk_mem period are ideal. Actual numbers vary with pin-to-pin differences in loading and transition direction (rise or fall). For more information, refer to the “Memory Control” chapter in the *PXA27x Processor Family Developer’s Manual*.

It should state:

Table 6-16. ROM AC Specification

Symbols	Parameters	MIN	TYP	MAX	Units†	Notes
tromAS	Address setup to nCS assert	1	—	1	clk_mem	—
tromCES	nCS setup to nOE asserted	—	—	0	clk_mem	—
tromCEH	nCS hold from nOE de-asserted	—	—	0	clk_mem	—
tromDSOH	MD setup to address valid	1.5	—	—	clk_mem	—
tromDOH	MD hold from address valid	0	—	—	clk_mem	—
tromAVDLF	Address valid to data latched for the first read access	2	MSCx[RDF]+2	32	clk_mem	—
tromAVDLS	Address valid to data latched for subsequent reads of non-burst devices	1	MSCx[RDF]+1	31	clk_mem	—
tflashAVDVS	Address valid to data valid for subsequent reads of burst devices	1	MSCx[RDN]+1	31	clk_mem	—
tromCD	nCS de-asserted after a read of next nCS or nSDCS asserted (minimum)	1	MSCx[RRR]*2+1	15	clk_mem	—

† Numbers shown as integer multiples of the clk_mem period are ideal. Actual numbers vary with pin-to-pin differences in loading and transition direction (rise or fall). For more information, refer to the “Memory Control” chapter in the *PXA27x Processor Family Developer’s Manual*.

D42. Keypad Interface external pull-down resistors specifications added

Affected Docs: *PXA27x Processor Family Developer’s Manual*

External pull-down resistor specifications for KP_DKIN<7:0> and KP_MKIN<7:0> should be added to Table 18-1 Keypad Interface I/O Signal Descriptions.

It currently states:

Table 18-1. Keypad Interface I/O Signal Descriptions

Name	Type	Description
KP_DKIN<7:0>	Input	Direct key inputs signals from the direct keypad and the rotary encoder sensors. • KP_DKIN<7:4> are dedicated input pins for direct keys 7– 4. • KP_DKIN<3:2> are used as input pins for direct keys 3 and 2 or for rotary-encoder sensor readings for rotary encoder 1, if it is enabled. • KP_DKIN<1:0> are used as input pins for direct keys 1 and 0 or for rotary-encoder sensor readings for rotary encoder 0, if it is enabled.
KP_MKIN<7:0>	Input	Matrix-key return input signals from the matrix keypad and are the matrix-keypad row readings.
KP_MKOUT<7:0>	Output	Matrix-key outputs. The keypad interface sends scan signals to the columns of the matrix keypad to detect any key(s) pressed. If an automatic scan is occurring, these outputs are driven by the automatic scan logic. At other times, they are driven by the settings of bits MS7 to MS0 in the Keypad Interface Control register (KPC described in Table 18-3).

It should state:

Table 18-1. Keypad Interface I/O Signal Descriptions

Name	Type	Description
KP_DKIN<7:0>	Input	Direct key inputs signals from the direct keypad and the rotary encoder sensors. • KP_DKIN<7:4> are dedicated input pins for direct keys 7– 4. • KP_DKIN<3:2> are used as input pins for direct keys 3 and 2 or for rotary-encoder sensor readings for rotary encoder 1, if it is enabled. • KP_DKIN<1:0> are used as input pins for direct keys 1 and 0 or for rotary-encoder sensor readings for rotary encoder 0, if it is enabled. 100 KΩ external pull-down resistors are required for KP_DKIN<7:0> signals.
KP_MKIN<7:0>	Input	Matrix-key return input signals from the matrix keypad and are the matrix-keypad row readings. 100 KΩ external pull-down resistors are required for KP_MKIN<7:0> signals.
KP_MKOUT<7:0>	Output	Matrix-key outputs. The keypad interface sends scan signals to the columns of the matrix keypad to detect any key(s) pressed. If an automatic scan is occurring, these outputs are driven by the automatic scan logic. At other times, they are driven by the settings of bits MS7 to MS0 in the Keypad Interface Control register (KPC described in Table 18-3).

D43. LCD controller output buffer strength description added

Affected Docs: *PXA27x Processor Family Developer's Manual*

Section 7.5.18 LCD Buffer Strength Control Register (LCDBSCNTR) should be updated with the following changes.

Two new paragraphs and a new note should be added following the first paragraph of this section:

The output buffer strength values for memory controller defined in *Section 6.5.7,* Table 6-39 Impedance Selection Options also apply to the LCD controller outputs configured by LCDBSCNTR.

The resistance values in Table 6-39 are provided in case there is a need to reduce electrical noise, such as ringing and reflections. Because of process variation and non-linear behavior of the transistors implementing the drive strength, these resistance values should only be used for signal integrity adjustment. The resistance values should not be used to design analog circuits such as RC time constants or as the current limiter of an LED driver circuit.

Note: Selectable LCD buffer strengths are for package pins, not only LCD signals. If different alternate functions or GPIO functions are selected rather than LCD signals, the drive strengths selected by LCDBSCNTR apply to those selections as well.

D44. SD/MMC CMD12 no responses handling description added

Affected Docs: *PXA27x Processor Family Developer's Manual*

In *Section 15.4.1.1 MMC/SD/SDIO Mode Data Transfers,* the following paragraph about handling of CMD12 no responses should be added to the end of this section:

MMC and SD memory cards must not respond to CMD12 unless they are in the data, rcv or irq states. Therefore, if no response is received for CMD12, it is reasonable to send a status query command such as CMD13, which is valid for all memory card states in data transfer mode, to determine the card state before proceeding to error handling.

D45. USB host Port 2 USBHPWR<2> and USBHPEN<2> clarification

Affected Docs: *PXA27x Processor Family Developer's Manual*

In *Section 20.3.1 Input Signals,* the following note about USBHPWR<2> should be added:

Note: USBHPWR<2> is multiplexed with GPIO<119> and is only available on the PXA271 and PXA272 processors. This GPIO is not available on the PXA270 processor. In an PXA270 design, the USB host Port 2 controller's over-current detection can be disabled by the following procedure:

- Set UHCRHDA[NOCP] as 1 to disable over-current protection support
- Set UHCHR[PSPL] as 0 to set polarity of the USBHPWR<3:1> input signals as active high

System designers wanting to implement over-current detection may use a GPIO and corresponding GPIO interrupt to read digital over-current indication signal from USB power IC.

In *Section 20.3.2 Output Signals,* the following note about USBHPEN<2> should be added:

Note: USBHPEN<2> is multiplexed with GPIO<120> and is only available on the PXA271 and PXA272 processors. This GPIO is not available on the PXA270 processor. Use a GPIO output to control USB host Port 2 power IC in a PXA270 design.

D46. COPROCESSOR: New CPU ID and JTAG ID Values.

Affected Docs: *PXA27x Processor Family Developer's Manual*

The table Table 2-3 Coprocessor: CPU ID and JTAG ID Values in section 2.2.5.1 should be updated to the table below.

It should state:

Table 2-3. Coprocessor: CPU ID and JTAG ID Values

Stepping	CPU ID	JTAG ID
A0	0x69054110	0x09265013
A1	0x69054111	0x19265013
B0	0x69054112	0x29265013
B1	0x69054113	0x39265013
C0	0x69054114	0x49265013
C5	0x69054117	0x79265013
PXA270M -- A1	0x69054118	0x89265013

The table, “Table 2-2 Processor ID Register”, of section 2.2.5.1 Needs to add to the Prod R Field, the final entry of, “ 0b1000 = PXA270M-- A1”. This edit must be performed with the table change above.

D47. Incorrect Ball Diameter Listed in S16 for PXA27x Package Information

Affected Docs: *PXA270M Specification Update* (this document)

S16 in this document states the following:

Section 3.1 Package Information of the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification* should be updated with the following changes:

The first paragraph should be updated. It currently states:

The PXA27x Processor Family has the following characteristics:

- Ball pitch: 0.65mm
- Ball diameter: 0.30 mm
- Substrate thickness: 0.21 mm
- Mold thickness: 0.45 mm

It should state:

The PXA27x processor family has the following characteristics:

- Ball pitch: 0.65 mm
- Ball diameter: 0.35 mm

This change is incorrect. The correct ball diameter is unchanged at 0.30 mm as written in Section 3.1 “Package Information” in the *PXA27x Processor Family Electrical, Mechanical, and Thermal Specification*.

D48. E16: Memory Stick does not come out of SLEEP mode after wakeup process

Affected Docs: *PXA270M Specification Update*

PXA270M does not support Memory Stick, so Errata 16 (E16) found in previous revisions of this document has now been removed from the current version of this document.

Marvell Semiconductor, Inc.
5488 Marvell Lane
Santa Clara, CA 95054, USA

Tel: 1.408.222.2500
Fax: 1.408.752.9028

www.marvell.com

Marvell. Moving Forward Faster