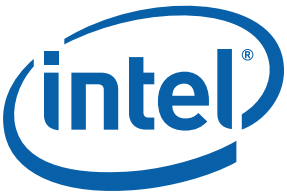

Intel® IXP400 Software v1.x

Software Specification Update

August 2007

Notice: The Intel® IXP400 Software may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Document Number: 273795, Revision: 019US

Contents

Revision History

Date	Revision	Description
August 2007	019	Added the following software issue as “open” for IXP400 software v1.x: 00165437
March 2007	018	Added the following software issues as “open” for IXP400 software v1.x: 00153398, 00153764, 00153855 Removed the following software issues. Those issues have been: root caused as not a defect: 1149(11), 2829, 00050884, 3022, 2644, 3176, 00055754, 00052772, 00049309 Removed v1.1, 1.2.1, 1.2.2 issues as Intel no longer supports these releases.
September 2006	017	Added the following software issues as “open” for IXP400 software v1.3, 1.4, and 1.5: 00153342
July 2006	016	Added the following software issues as “open” for IXP400 software v1.5: 00018943, 00152702, 00152731, 00152701, 00019406.
March 2006	015	Added the following software issues as “open” for IXP400 software v1.5: 00018668
October 2005	014	Added the following software issues as “open” for IXP400 software v1.5: 00056585, 00057521, 00057683, and 00018713.
September 2005	013	Added the following software issues as “open” for IXP400 software v1.5: 00057654.
August 2005	012	Added the following software issues as “open” for IXP400 software v1.5: 00055983. Added documentation issue 00056779.
May 2005	011	Added the following software issues as “open” for IXP400 software v1.5: 00056449, 00056221, 00055503, 00055339, 00055187, 00054053, 00053847, 00053596, 00053591, 00049382, and 00049291.
February 2005	010	Added the following software issues as “open” for IXP400 software v1.5: 00054556, 00054344, 00054427, 00053897, 00053798, 00053691 00049309, 00049294, 00050954, 00049362, 00053699, 00049680, 00049619, 00053660, 00053521, 00053661, 00050962, 00049277, 00053693. Updated the “Summary Table of Issues” to reflect the implication of new issues to previous releases. To improve usability, changed layout of “Summary Table of Issues” , “Software Issues”, and “Documentation Issues” .

Date	Revision	Description
August 2004	009	Added the following software issues as “open” for IXP400 software v1.4: 1149, 3321, 3330, 3331, 3347, 3385, 3395, 3414, 3437, 3454, 3742, 3847, 3864, 3879, 3928, and 4096. Added 3928 to “open” in 1.3. Added the following software issues to “open” in 1.3, “open” in 1.2.1, “open” in 1.2.2, and “open” in 1.1: 3414. Added the following software issue to “open” in 1.2.2, and “open” in 1.3: 3347. Added the following documentation issue: 4086. Modified the resolution for the following software issues in 1.1: 3035, 1832 and 1729. Modified the following resolution in “open” for 1.2.1, 1.2.2 and 1.3: 3035. Added the following software issues to “open” in 1.1, 1.2.1, 1.2.2, 1.3, and 1.4: 3987.
July 2004	008	Added the following software issues as “resolved” for IXP400 software v1.4: 2417, 2649, 2770, 2782, 2792, 2796, 2829, 2848, 2870, 2892, 2895, 2903, 2910, 2919, 2946, 2967, 3039, 3035, 3072, 3104, 3118, 3144, 3150, 3157, 3176, and 3180. Added the following software issues as “open” for IXP400 software v1.4: 2121, 2644, 2739, 3018, 3022, 3182, 3217, and 3263. Removed 2277 from “resolved in software 1.3” table. Moved 1173 from “open for software release 1.3” to “resolved for software release 1.3”.
December 2003	007	Added the following software issues: 1893, 1950, 2026, 2076, 2125, 2231, 2346, 2369, 2417, 2520, 2617, 2618, 2644, 2649, 2715, 2739, 2770, 2782, 2792, 2796, 2829, and 2848. Added the following software issues to “open” in 1.1: 2102, 2113, 2120, and 2151. Added the following software issues to “open” in 1.1 and “resolved” in 1.2.1: 1768 and 1838. Moved the following software issue from “open,” in 1.2.1 and 1.2.2, to “resolved,” in 1.1: 1255 (17). Added documentation issue 2265.
July 2003	006	Changed the title and some introductory paragraphs to convey that document applies to multiple versions of the software.
July 2003	005	Added the following software issues: 1073, 1725, 1729, 1740, 1948, 2141, 2188, 2277, 2296, 2309, 2394, 2419, and 2484. Added documentation issue 2188. Moved the following software issues from “open,” in 1.2.1, to “resolved,” in 1.2.1: 1356 (20) and 1497 (24). Removed software issue 0922 (5) from “open” in 1.1. Removed the following software issues: 0556 (1), 0803 (2), 0878 (3), 1024 (8), 1505 (25), 1525 (28), 1526 (29), 1568 (30), 1589 (32), 1622 (33), 1638 (34), 1639 (35), 1640 (46), 1712 (38), and 1717 (39).

Date	Revision	Description
June 2003	004	Adopted new documentation format and changed issues' sequential numbering to ID numbers. Refined wording of resolved and open software issues. Added the following software issues: 2068, 2082, 2102, 2113, 2120, 2121, and 2151. Determined that the following items were not really issues: 0556, 0878, 1024, 1505, 1525, 1526, 1568, 1622, 1638, 1639, 1640, and 1717. Added documentation change 1434. Removed the software issue previously known as Issue 7.
March 2003	003	Added software issues formerly known as 33 through 48. Updated status for former Software Issue 31 and removed Documentation Changes 1 and 2 from Revision 001.
January 2003	002	Added software issues formerly known as 18 through 32. Updated software issues formerly known as 8, 10, 12, and 17. Changed status of software issues formerly known as 4, 5, 7, 11, 12, and 14.
October 2002	001	Documented software issues formerly known as 1 through 17 and documentation changes formerly known as 1 and 2.

§ §

Preface

This software specification update describes resolved and open software issues for the Intel® IXP400 Software and is an update to the specifications listed in the following, "Affected and Related Documents" table. This document also includes document changes that are subsequently posted to the documents listed in "Documentation Issues" on page 64.

Intel® IXP400 Software supports the Intel® IXP4XX Product Line of Network Processors.

This document supersedes the earlier version of this software specification update, and may contain information that was not previously published. Intel will use commercially reasonable efforts to include all documented defects; however, Intel makes no representations or warranties concerning the completeness of this software specification update.

This document is intended for hardware system manufacturers and software developers of applications, operating systems, or tools.

Affected and Related Documents

Title	Document Number
Intel® IXP400 Software Programmer's Guide	252539
Intel® IXP42X Product Line of Network Processors and IXC1100 Control Plane Processor Developer's Manual	252480

Note: The above documents are available at http://www.intel.com/design/network/products/npfamily/docs/ixp4xx.htm#.

Summary Table of Issues

The tables in this section summarize the resolved and open known software issues of the Intel® IXP400 Software and any changes being made to that product's documentation. The details for these issues are presented in "Software Issues" on page 13 and "Documentation Issues" on page 64.

A change bar — like that shown in the left margin — identifies material that has been added or modified since the previous version of this document.

Organization of Summary Tables

The summary tables are divided into two sections: software issues and documentation issues.

In the upcoming "Software Issues" section, a table lists all known software issues for the various releases of the software.

In the "Documentation Issues" section, a table lists all known documentation issues to be addressed in the next release of each cited document.

Summary-Table Codes

The summary table fields provide the following information:

Table Field	Description
Software Versions	In the "Software Issues" section, the software release(s) that are affected by that table row's software issue. • Blank = Issue resolved in this version of the software (or) Issue does not apply to this version of the software • "X" = Issue applies to this version of the software
ID #	The identification number of the known software or documentation issue.
Title	The title of the known software or documentation issue.
Page	The page — in this document — that gives details about the indicated software or documentation issue.
Affected Document	In the "Documentation Issues" section, the document(s) and version number(s) of the document(s) affected by the documentation issue. The next version of each cited document will resolve the cited issue.

Software Issues

In the following tables, the applicability of each table row's software issue is indicated in the **Software Version** column:

- Blank = Issue resolved in this version of the software (or) Issue does not apply to this version of the software

- "X" = Issue applies to this version of the software

Software Versions 1.x

Intel® IXP400 Software v1.x Issues Summary and Quick Reference (Sheet 1 of 4)

Software Version (X = Issue Open)			ID #	Title	Page
1.5	1.4	1.3			
X			00165437	LLP Dispatcher May Erroneously Enable Disabled Sporadic Queue Notification	63
X	X	X	00153398	Ethernet Collisions in Half Duplex Mode due to Same Seed Number used in Backoff Algorithm	59
X		X	00153764	Incorrect Statistic Counter Value Shown in ixEthAccDataPlaneShow()	61
X	X	X	00153855	Excessive Interrupts to Intel XScale Processor Detected When Sticky Bit Interrupt Enabled	62
X	X	X	00153342	B-1 stepping for Intel® IXP42X Product Line of Network Processors Requires Minor Change to IxFeatureCtrl.c to Accommodate Updated Product Revision ID	58
X			00018943	ixEthAccPortMacReset Function does not Restore MAC Unicast Address	52
X			00152702	Functions to Change Age Type (Static <-> Dynamic) not Working When the Same MAC Entry has been Registered with the EthDB	53
X			00152731	Incorrect VLAN Port Membership/Transmit Tagging Table Update	53
X			00152701	Under Heavy Traffic, EthDB-NPE Acknowledge-Message Timeout Triggers Fatal Warning	54
X			00019406	Memory Leakage in EthACC	55
X	X		00018668	Ethernet MAC Tx Lockup Issue	52
X			00018713	Buffer Length Constraint on Ethernet Rx Path	51
X	X	X	00057683	HSS Port Cannot be Disconnected Successfully when Port is Busy	49
X			00057521	Wrong Configuration of Encryption/Authentication Operation in Crypto Codelet	49
X	X		00056585	Ethernet Transmit Lockup	48
X			00057654	NPE Corrupts Frame on Ingress When Bidirectional Traffic in Operation	48
X			00055983	Unresolved Symbol ixOsalBuffPoolFreeCountGet in Linux Build	48
X			00056221	Spanning Tree Protocol (STP) and VLAN Filtering Does Not Work with the Header Conversion Featured NPE Image	47
X			00056449	If a Network has More Than 511 Individual MAC Addresses, the Ethernet NPE Becomes Unresponsive After Learning all of the Available MAC Addresses	47
X	X		00055503	IxEthAcc Fails to Re-Initialize the Ethernet NPE Interfaces After Unloading the IxEthAcc Module	47
X	X		00055339	Intel® IXP400 Software USB Access-Layer Component Fails to Transmit in Bulk Transfer Mode When the Client Software Sets the 'ENABLE_TX_SEQ' Flag to '1'	46
X			00055187	EthAcc Access-Layer Component is Unable to Replenish Filtered Frames	46
X			00054556	Memory Leaks in VxWorks* When Thread Exits or is Killed	46
X			00054427	IxEthAccRxFrameType Enumerations Definition in IxEthAcc.h Defines Wrong Values	45
X			00054344	IxEthAccCodelet Does Not Convert 802.11 Frames to 802.3 Frames	44
X	X	X	00054053	IxHssAcc Channelized Service Cannot be Restarted After Disabling the Port	44
X			00053897	Linux NPE Ethernet Driver: Typo Mistake has an Impact on Performance Enhancements	44

Intel® IXP400 Software v1.x Issues Summary and Quick Reference (Sheet 2 of 4)

Software Version (X = Issue Open)			ID #	Title	Page
1.5	1.4	1.3			
X			00053825	In VxWorks, Loadable Modules Doesn't Include NpeMicrocode.o	43
X			00053847	VxWorks Build Fails in a Microsoft* Windows Environment	43
X	X	X	00053798	Possible Re-Entrancy Issue with ATM Access-Layer Component	43
X	X	X	00053699	Inconsistency with Inter Frame Gap in IxEthAccMac Access-Layer Component	42
X			00053693	IxCryptoAccCodelet Does Not Display any Output in VxWorks on the Tornado* WindShell	42
X			00053691	Inaccuracy in Timer when using ixOsalSysClkRateGet() API	42
X			00053661	IxEthAccCodelet Does Not Filter Out-Of-Range VID	41
X			00053660	Use of ix_ossl_malloc API Returns Compiler Error	41
	X		00053596	Intel® IXP400 Software Linux Integration Patch 1.1 Fails With MontaVista Linux 3.1 IXDP425 LSP Update	40
X	X	X	00053591	Under MontaVista Linux the Intel® IXP400 Software Does Not **rmmod** Cleanly	40
X			00053521	In VxWorks, Multiple Calls of ixOsalIrqBind() for the Same Interrupt Returns IX_SUCCESS Instead of IX_FAIL	40
X	X	X	00050954	BSS Memory Usage is High in IxPerfProfAcc Access-Layer Component	39
X	X	X	00049619	Very Slow MII Access via 'ixEthAccMiiReadRtn()' and 'ixEthAccMiiWriteRtn()'	39
X	X	X	00049382	IxEthAal5AppCodelet Counters Displays Inconsistent Results	39
X	X	X	00049371	IxEthAcc Control Plane APIs are not Re-Entrant	38
X	X		00049362	The IxPerfProfAccCodelet is incompatible with the Linux NPE Ethernet Driver Running in a Polled Mode	38
X	X		00049294	In Linux, the IxCryptoAccCodelet Cannot be Removed Using the “-1” Option	37
X	X		00049291	Linux NPE Ethernet Driver Drops Packets for an Oscillating Traffic at a High Traffic Rate	37
X	X		00049277	ATM Scheduler Ignores the Peak Cell Rate (PCR) Value	37
	X		4096	Use of ixEthDBFilteringPortMaximumFrameSizeSet( ) API Call Causes a Crash if Only One of the Ethernet NPEs is Initialized	36
	X	X	3987	Cryptographic Processing on NPE C may Halt for a Specific Sequence of Requests to Register and Unregister MD5-Authentication Cryptographic Context	36
X	X	X	3928	HSS Access Component Fails to Initialize When Two Specific NPE Images are Used	35
	X		3879	The OSSL Component’s TICKS_PER_NSEC Macro Returns an Incorrect Value	35
	X		3864	The Queue Descriptor ‘get’ Fails During Rev Key Generation	34
X	X		3847	Linux NPE Ethernet Driver and EthAcc Codelet Does Not Support Intel® LXT973 Revision A3 MII ID 0x00137a11	34
	X		3742	Crypto Request gets Rejected for **HMAC-MD5** Only Operation Following a De-Registration by the WEP Services	33
	X		3454	Unresolved Symbols Messages Appear During Linux Boot Process	33
	X		3437	ixEthDBTriggerPortUpdate() Crashes if Called when EthDB Learning Feature is Disabled Using IxFeatureCtrl	32
	X	X	3414	Ethernet NPE may Incorrectly Filter Incoming Frames Destined for Other Network Interfaces in the System	32
	X		3395	Linux NPE Ethernet Driver Does Not Behave Correctly with Netfilter* Firewall Option is Selected in the Linux Kernel	31

Intel® IXP400 Software v1.x Issues Summary and Quick Reference (Sheet 3 of 4)

Software Version (X = Issue Open)			ID #	Title	Page
1.5	1.4	1.3			
	X		3385	Intel XScale® Processor Does Not Respond to NPE Rebalancing Request after a Port Disable/Enable Cycle	31
	X	X	3347	Ethernet NPE Fails to Respond to a Port Disable Message After the EthAcc Access Component ixEthAccPortDisable( ) is Called	31
	X		3331	While Using the IxEthAal5App Codelet, the ATM Service Parameter PCR (Peak Cell Rate) Does Not Limit the Traffic to a Precise Value as Expected	31
	X		3330	Linux Xcycle Measurement in PerfProfAcc Access Component is Non-Functional	30
	X		3321	Memory Leak Occurs in Linux Kernel Patched with "ebtables" and "br-nf" Patches While Bridging Traffic Through NPE Ethernet Port with Linux NPE Ethernet Driver	30
	X		3263	While Using the EthAcc Component, ixEthAccPortDisable() Fails and Outputs the Following Message: [fatal] IXETHACC:ixEthAccPortDisable: ixEthAccPortDisable Failed port 1 (state = 4)	29
	X	X	3217	Timestamp Register Gives Incorrect Results on Linux When Console UART is set to Low Speed (such as 9600 bps)	29
		X	3202	After Being Used by the Bootloader for Loading the Images, Tx Traffic may be Blocked on the NPE Ethernet Ports	29
		X	3188	On Linux, MAC Address Learning or Aging may Cease to Work After the 'IxEthDB' Component Issues a Warning Message	27
	X		3182	TxDone Traffic on Ethernet NPE Fails to Run With a Certain Boot Configuration	27
		X	3180	Linux NPE Ethernet Driver Does Not Set the Port MAC Address as Supplied by the User	26
		X	3157	Statistics Returned by IxEthAcc API Functions ixEthAccMibIIStatsGet() and ixEthAccMibIIStatsGetAndClear() are not Correct in Little-Endian Mode	26
		X	3150	Linux NPE Ethernet Driver (ixp425_eth) May Crash Due to Execution of Non-Interrupt-Safe Code from Within an Interrupt Context	25
		X	3144	Linux NPE Ethernet Driver (ixp425_eth) Fails to Process Received Traffic After Port is Disabled, and Re-Enabled in Presence of Traffic	25
		X	3118	IX_MBUF's m_nextpkt Field Gets Internally Overwritten by the ethAcc Component	24
		X	3104	Ethernet NPEs Drops Ethernet Frames Greater Than 1522 Bytes	24
		X	3072	No Response From the Ethernet NPE When Ports are Disabled in Quick Succession After Enabling	24
		X	3059	At High Throughput, 'Port Disable' Overflows the Internal FIFOs in the Ethernet NPEs	24
		X	3039	IX_ACC_DATA_CACHE_FLUSH Does Not Drain the MMU Write Buffers	23
		X	3035	Ethernet NPE Corrupts SDRAM Memory	22
	X	X	3018	ix_ossl_thread_kill Does Not Kill Threads in Linux	21
		X	2967	RxStatus Overruns in the Ethernet NPE Firmware	21
		X	2946	IX_OSSERV_READ_BYTE Macro Does Not Work in Little Endian Address Coherent Mode	20
		X	2919	ADSL Driver Displays a Wrong Firmware Version	20
		X	2910	Compilation Error – 'baddr' is Undefined	20
		X	2903	EthDB Access Component Fails to Synchronize with NPE MAC Address Learning Table After Port is Disabled and Re-Enabled	19
		X	2895	Linux NPE Ethernet Driver Does Not Invalidate the mbuf/skbuf Payload	19
		X	2892	Unused Variables in CryptoAcc Component	19

Intel® IXP400 Software v1.x Issues Summary and Quick Reference (Sheet 4 of 4)

Software Version (X = Issue Open)			ID #	Title	Page
1.5	1.4	1.3			
		X	2870	Linux NPE Ethernet Driver Crashes with Transmit Timeout Error	18
		X	2848	Ethernet Access Component dot3StatsFrameTooLongs Counter May be Incremented Incorrectly	18
		X	2796	AtmdAcc Transmit Failure After Repeatedly Disconnecting/Connecting VCs in Polled Mode	18
		X	2792	EthAal5App and fpathAcc Codelets Fail to Run on Intel® IXP421 Network Processor	17
		X	2782	ixTimerCtrlScheduleRepeating(...) Function Does Not Function Properly Under Linux	17
		X	2770	Ethernet AAL-5 Codelet May Transmit Ethernet Packets with Bad CRC	17
	X	X	2739	ixOsServTaskSleep(...) May Not Provide 1-ms Resolution	16
		X	2649	UART-Access Component May Not Clear Request to Send (RTS)	15
		X	2417	ioctrl Issues with the Linux NPE Ethernet Driver	14
	X	X	2121	IRQ Locks Up While Using MontaVista Linux Kernel with ethAcc Codelet Bridge Demo for 64-Byte Packets Over a Prolonged Period of Time for Aggregate Traffic Greater Than 200,000 Bytes	13

Documentation Issues

Documentation Issues 1.x

Affected Documents	ID #	Title	Page
2525329-002 through 005	1434 (22)	ixEthAccPortRxFreeReplenish Needs Update	64
252539-002, 252539-003	4086	The CDVT in the ATM Scheduler Does not Comply with the ATM-TM-4.1 Standard	65
252539-002, 252539-003	2265	Buffers Passed to Access Layer Components Must Have Bits 31-29 Set to 0	65
252539-002, 252480-002, 252479-002, 252702-001, 252741-001, 252725-001, 273811-002, 273810-002	2188	Inconsistent NPE Terminologies Between Code and Documentation	64
252539-002	1434 (22)	ixEthAccPortRxFreeReplenish Needs Update	64

Software Issues

This section gives the details of the software issues summarized in the **Summary Table of Issues** section's "Software Issues" on page 8.

Each software issue's **Version(s)** paragraph lists which release(s) of the software are impacted by that issue.

A change bar — like that shown in the left margin — identifies material that has been added or modified since the previous version of this document.

Note: The identification numbers for software and documentation issues formerly were assigned sequentially, for each release of this document. Issues are assigned a permanent, database-generated identification number.

To assist in the transition from the old numbering system, any previously published issues are identified by their permanent, database ID number, followed — in parentheses — by the issue's previous number. (For example: *Software Issue #4 (SCR 0886)*, from an earlier release of this document, would be identified as *Software Issue #0886 (4)*.

IRQ Locks Up While Using MontaVista* Linux* Kernel with ethAcc Codelet Bridge Demo for 64-Byte Packets Over a Prolonged Period of Time for Aggregate Traffic Greater Than 200,000 Bytes

Reference #: 2121

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: While using the ethAcc codelet bridge demo with MontaVista Linux kernel, sending 64-byte packets bidirectionally might lock up the system after 8 million frames. Following this, a message appears (`IRQ LOCK: IRQ3 is locking the system, disabled`) and the data will not pass again until after reboot.

Implication: This issue only occurs for bidirectional traffic because this issue does not occur for aggregate Ethernet traffic of less than 200,000 byte. At some high interrupt rate, the MontaVista kernel may disable an interrupt. The MontaVista kernel never re-enables the disabled interrupt.

Resolution: The user can break the cycle traffic-interrupt cycle by doing one of the following:

- Adding an extra delay at some threshold
- Ensuring that a lower priority interrupts gets a chance to run.(for example, timer interrupt) at some threshold

Contact your Intel representative to obtain the software patch.

ioctrl Issues with the Linux* NPE Ethernet Driver

Reference #:2417

Product: Intel® IXP400 Software

Version(s): 1.3

Description: Calls to `dev_do_ioctrl(...)` to the **get address of MII PHY in use** command, is missing a return from the switch statement and the error-handing sections of the **Read MII PHY register** and **Write MII PHY register** commands have the arguments in the wrong order.

Implication: If a call is made to dev_do_ioctrl (**get address of MII PHY in use**) the code falls into the **Read MII PHY register** command and may generate an error. If an error occurs in dev_do_ioctrl (**Read** or **Write MII PHY register**), the error message generated has PHY Register and PHY ID switched.

Resolution: Make the following changes to dev_do_ioctrl(...) in `ixp425_eth.c`:

From:

```
/* Get address of MII PHY in use */

case SIOCGMIIPHY:

case SIOCDEVPRIVATE:

    data->phy_id = phy;
```

To:

```
/* Get address of MII PHY in use */

case SIOCGMIIPHY:

case SIOCDEVPRIVATE:

    data->phy_id = phy;

    return 0;
```

From:

```
    case SIOCGMIIREG:
    case SIOCDEVPRIVATE+1:
        down (miiAccessMutex);     /* lock the MII register access mutex */
        if ((res = ixEthAccMiiReadRtn (data->phy_id, data->reg_num, &data-
>val_out)))
        {
        P_ERROR("Error reading MII reg %d on phy %d\n",
               data->phy_id, data->reg_num);
        res = -1;
        }
        up (miiAccessMutex);/* release the MII register access mutex */
        return res;
```

To:

```
    case SIOCGMIIREG:
    case SIOCDEVPRIVATE+1:
        down (miiAccessMutex);     /* lock the MII register access mutex */
        if ((res = ixEthAccMiiReadRtn (data->phy_id, data->reg_num, &data->val_out)))
        {
        P_ERROR("Error reading MII reg %d on phy %d\n",
               data->reg_num, data->phy_id);
        res = -1;
        }
        up (miiAccessMutex);/* release the MII register access mutex */
        return res;
```

From:

```
    /* Write MII PHY register */
    case SIOCSMIIREG:
    case SIOCDEVPRIVATE+2:
        down (miiAccessMutex);     /* lock the MII register access mutex */
        if ((res = ixEthAccMiiWriteRtn (data->phy_id, data->reg_num, data->val_in)))
        {
        P_ERROR("Error reading MII reg %d on phy %d\n",
            data->phy_id, data->reg_num);
        res = -1;
        }
        up (miiAccessMutex);/* release the MII register access mutex */
        return res;
```

To:

```
    /* Write MII PHY register */
    case SIOCSMIIREG:
    case SIOCDEVPRIVATE+2:
        down (miiAccessMutex);     /* lock the MII register access mutex */
        if ((res = ixEthAccMiiWriteRtn (data->phy_id, data->reg_num, data->val_in)))
        {
        P_ERROR("Error reading MII reg %d on phy %d\n",
            data->reg_num, data->phy_id);
        res = -1;
        }
        up (miiAccessMutex);/* release the MII register access mutex */
        return res;
```

UART-Access Component May Not Clear Request to Send (RTS)

Reference #: 2649

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The error condition occurs when going from HW-flow control back to SW-flow control. The RTS signal, set by UART access component, when HW-flow control is enabled, is not cleared while the component reverts to SW-flow control.

Implication: This may cause some issues with terminal-emulation software which expects the RTS output signal from the UART on a Intel® IXP42X product line to be de-asserted while the UART is in SW-flow control mode.

Resolution: Make the following change to the `ixUARTOptsSet()` function in `IxUART.c`:

From:

```
if (!(options & CLOCAL)) /* hardware (RTS/CTS) */
{
IX_UART_REG_READ(pUART, IX_MCR, mcr);
IX_UART_REG_WRITE(pUART, IX_MCR, (mcr | IX_MCR_RTS));
IX_UART_REG_WRITE(pUART, IX_IER, (ier & (~IX_IER_TIE)));
IX_UART_REG_WRITE(pUART, IX_IER, (ier |= IX_IER_MIE)); /* enable modem status
interrupt */
}
else /* software */
{
IX_UART_REG_WRITE(pUART, IX_IER, (ier & ~(IX_IER_MIE))); /* software flow ctrl -
default */
}

pUART->options = options;
return IX_SUCCESS;
}
```

To:

```
if (!(options & CLOCAL)) /* hardware (RTS/CTS) */
{
IX_UART_REG_READ(pUART, IX_MCR, mcr);
IX_UART_REG_WRITE(pUART, IX_MCR, (mcr | IX_MCR_RTS));
IX_UART_REG_WRITE(pUART, IX_IER, (ier & (~IX_IER_TIE)));
IX_UART_REG_WRITE(pUART, IX_IER, (ier |= IX_IER_MIE)); /* enable modem
status interrupt */
}
else /* software */
{
/* Read MCR status */
IX_UART_REG_READ(pUART, IX_MCR, mcr);


/* Reset RTS bit of the MCR */
IX_UART_REG_WRITE(pUART, IX_MCR, mcr & (~IX_MCR_RTS));
IX_UART_REG_WRITE(pUART, IX_IER, (ier & ~(IX_IER_MIE))); /* software flow
ctrl - default */
}

pUART->options = options;
return IX_SUCCESS;
}
```

ixOsServTaskSleep(...) May Not Provide 1-ms Resolution

Reference #:2739

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The ixOsServTaskSleep(...) function utilizes operating system function to provide the requested task sleep time. The default Linux timer has a resolution of 10 ms and the default VxWorks timer has a resolution of 16 ms.

Implication: Calls to ixOsServTaskSleep(...) with the operating system timer defaults, provides at best, 10-ms resolution for Linux and 16-ms resolution for VxWorks.

Resolution: The default operating system timer values can be changed to provide better resolution. In the Linux environment, this requires a recompile of the kernel. In the VxWorks environment, a call can be make to sysClkRateSet(...).

Ethernet AAL-5 Codelet May Transmit Ethernet Packets with Bad CRC

Reference #:2770

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The Ethernet AAL-5 codelet (ixEthAal5AppCodeletMain(...)) does not check the receive status of the AAL-5 frame nor does it verify the Ethernet packet length before submission to be transmitted.

Implication: If a valid AAL-5 frame was not received (for example, due to dropped cycles) the Ethernet AAL-5 codelet submits it to an Ethernet NPE for transmission with an invalid length. This causes an incomplete/invalid Ethernet packet to be sent. This appears as a CRC error at the receiver.

Resolution: The Ethernet AAL-5 codelet should only process AAL-5 PDUs with the status of IX_ATMDACC_AAL5_VALID. All other ATM PDUs should have there mbufs recycled to the ATM receive-free queue.

Since the Ethernet frame length is derived from the SDU length, extracted from the AAL-5 PDU trailer cell, calculate the Ethernet frame length, and verify the value to be the same. If the frame lengths are not equal, recycle the mbufs to the ATM receive-free queue.

ixTimerCtrlScheduleRepeating(...) Function Does Not Function Properly Under Linux*

Reference #:2782

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The ixTimerCtrlScheduleRepeating(...) access function does not consistently generate calls to the user-callback function in the Linux environment.

Implication: The user callback function may not be called at the specified rate.

Resolution: The Linux operating system's timer services should be used instead of the function ixTimerCtrlScheduleRepeating(...).

EthAal5App and fpathAcc Codelets Fail to Run on Intel® IXP421 Network Processor

Reference #:2792

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The IXP421 network processor variant does not have Ethernet NPE-B (also called NPE-C), which is required by the EthaAal5App and fpathAcc codelets. The codelets do not check the capabilities of the target processor before attempting to run.

Implication: The EthAal5App and fpathAcc codelets will not load on an IXP421 network processor.

Resolution: None.

AtmdAcc Transmit Failure After Repeatedly Disconnecting/Connecting VCs in Polled Mode

Reference #:2796

Product: Intel® IXP400 Software

Version(s): 1.3

Description: After AtmdAcc is enabled and VCs are successfully connected for the first time, traffic is transmitted at a high data rate (above 4.45 Mbps). The VCs are then disconnected and reconnected. At this point, AtmdAcc may fail to transmit the traffic at a high data rate, on the Tx side of the VCs. This issue is seen in polled mode.

Implication: After disconnection and reconnection, transmission above 4.45 Mbps may fail. Traffic transmits successfully at lower data rates.

Resolution: Use the ixAtmdAccRxFreeLowCallbackRegister() function to set the RxFree threshold value to 1.

Ethernet Access Component dot3StatsFrameTooLongs Counter May be Incremented Incorrectly

Reference #:2848

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The Ethernet access component dot3StatsFrameTooLongs counter is incremented for *any* Ethernet frames received with a size greater than 1,518 bytes. Valid, VLAN-tagged Ethernet frames — which are greater than 1,518 bytes in size — causes this counter to be incremented.

Implication: The counter is incorrectly incremented for valid Ethernet frames larger that 1,518 bytes.

Resolution: None.

Linux* NPE Ethernet Driver Crashes with Transmit Timeout Error

Reference #:2870

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The Linux NPE Ethernet driver does not start the timer for the last packet to be submitted for transmission

Implication: When an application tries to communicate to the NPE Ethernet device, the kernel crashes. This is because the timer is not set properly. The netwatchdog erroneously stops the device while it detects that the queue is stopped (for example, for transmit overflow), resulting in a crash with print out **NETDEV WATCHDOG: ixp0/1 transmit time out**.

Resolution: In file /linux-2.4.18_mvl30/drivers/net/ixp425_eth.c, add the following line to the end of the dev_hard_start_xmit function.

```
dev-> trans_start=jiffies;
```

Unused Variables in CryptoAcc Component

Reference #:2892

Product: Intel® IXP400 Software

Version(s): 1.3

Issue: In IXP400 software v1.3, AES-CTR implementation is not generic, it is based on the Internet draft - **draft-ietf-ipsec-ciph-aes-ctr-00.txt**, dated July 2002. Since 2002, two more drafts have been published: **draft-ietf-ipsec-ciph-aes-ctr-01.txt**, dated September 2002, and **draft-ietf-ipsec-ciph-aes-ctr-03.txt**, dated January 2003. The difference between the draft is in the implementation of the CTR block format. With the new implementation, SPI is needed in API for construction of the CTR block. If clients wish to implement newer version of code, they have to modify AES CTR block construction code in cryptoAcc to synchronize with the latest draft.

Implication: Only those who have implemented AES-CTR mode using cryptoAcc support are impacted.

Resolution: Remove **UINT32 securityParameterIndex**; from struct IxCryptoAccCipherCtx.

Remove all reference to **securityParameterIndex** from the client's application.

Users should construct the AES CTR counter block based on the Internet draft adopted and passed into cryptoAcc API as an argument of IV (initialization vector).

Linux* NPE Ethernet Driver Does Not Invalidate the mbuf/skbuf Payload

Reference #:2895

Product: Intel® IXP400 Software

Version(s): 1.3

Description: When the Ethernet driver gets its data form the sk_buff pool and then invokes the replenish function to free the buffers, the payload might still exist in the MMU cache line with its dirty bit set to ‘1’. This dirty bit might get flushed at any time by the MMU, causing the new payload (by NPE write) to be overwritten by the old contents.

Implication: When the NPE writes a new mbuf payload, the dirty cache lines from the previous mbuf data can get flushed at any time by the MMU of the Intel XScale® Processor. As a result, the mbuf payload may contain the wrong data.

Resolution: Use IX_ACC_DATA_CACHE_INVALIDATE(addr,len) before using `ixEthAccRxFreeReplenish()`.

EthDB Access Component Fails to Synchronize with NPE MAC Address Learning Table After Port is Disabled and Re-Enabled

Reference #:2903

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The EthDB component holds a copy of the MAC Address learning database, which is constructed by the NPE from the source MAC addresses of all the incoming Ethernet frames. Periodically, and for certain events, the copy in the Intel XScale® processor needs to be resynchronised with the NPE database. However, after disabling and then re-enabling the port, this synchronization may fail without reporting any errors.

Implication: The symptoms are varied, but typically appears that no new MAC addresses are being learned from incoming traffic, or that old MAC addresses reappear in the database. Additionally, it is likely that the NPE will not filter the MAC addresses subsequently provisioned via the EthDB API.

Resolution: None

Compilation Error – 'baddr' is Undefined

Reference #:2910

Product: Intel® IXP400 Software

Version(s): 1.3

Description: There is a mismatch between the macro parameter **bAddr** and the actual used symbol, **baddr**. The IX_OSSERV_BYTE_WRITE_RAW macro uses the wrong parameter.

Implication: The macro doesn't work and causes a compilation error – **baddr** is undefined.

Resolution: Replace **baddr** with "**bAddr** in the macro body, in xscale_sw/src/include/IxOsServicesMemAccess.h.

ADSL Driver Displays a Wrong Firmware Version

Reference #:2919

Product: Intel® IXP400 Software

Version(s): 1.3

Description: API ixAdslShow( ) displays incorrect firmware version information.

Implication: User will not be able to determine the correct firmware version being activated in the ADSL driver.

Resolution: In file IxAdslCtrleLib.c, modify

```
From

CtrlrSWVersion->number3=(unsigned char)swversionbytes[0] & maskuppernibble;

to

CtrlrSWVersion->number3=(unsigned char)swversionbytes[0];
```

IX_OSSERV_READ_BYTE Macro Does Not Work in Little Endian Address Coherent Mode

Reference #:2946

Product: Intel® IXP400 Software

Version(s): 1.3

Description: IX_OSSERV_READ_BYTE macro does not work in Little Endian Address Coherent Mode.

Implication: An address cast operation is placed inside the macro in a place where it can cause compilation errors when used in bit operations. Compilation errors are possible when bit operations are used together with the byte address field.

Resolution: Change the casting in xscale_sw/src/include/IxOsServicesMemAccess.h file as follows:

From

```
#define IX_OSSERV_READ_BYTE(bAddr)
IX_OSSERV_READ_BYTE_IO(IX_OSSERV_LE_AC_BUSTOXSB((volatile UINT8 *)bAddr))
```

to

```
#define IX_OSSERV_READ_BYTE(bAddr) IX_OSSERV_READ_BYTE_IO((volatile UINT8
*)IX_OSSERV_LE_AC_BUSTOXSB(bAddr))
```

RxStatus Overruns in the Ethernet NPE Firmware

Reference #:2967

Product: Intel® IXP400 Software

Version(s): 1.3

Description: Invalid Rx Status information is conveyed to the firmware. Because the Rx Status information in the MAC engine is not pipelined/queued in a FIFO, there is a possibility that the Rx Status register value in the Ethernet MAC engine for the current packet might get overwritten with the subsequent packet Rx status.

Implication: If two packets are received in the order such that the first Ethernet packet is good and the second packet is errored, and if the Rx Status of the first packet gets overwritten by the Rx status of the second packet in the MAC register, then the first packet is interpreted by the firmware as a bad packet. Conversely, a bad packet might also get treated as a good packet.

Resolution: Software fixes have been made to the NPE microcode to insert a checkpoints in the Ethernet Rx MAC firmware to check whether the Rx Status has been overwritten. If it has, it ignores the Rx Status information and flushes the Rx FIFO at the MAC. Note that the microcode images for this fix on IXP400 software v1.4 are not backward-validated on earlier software releases. To resolve this issue, update the IXP400 software to IXP400 software v1.4. If you believe that this issue affects your systems and designs, and cannot use IXP400 software or later, contact your Intel representative.

ix_ossl_thread_kill Does Not Kill Threads in Linux*

Reference #:3018

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: ix_ossl_thread_kill does not kill Linux kernel threads. It only sends a SIGKILL signal to the thread, which wakes up the thread. The kernel threads cannot be killed by using this API call.

Implication: Clean-up operations involving killing threads fails.

Resolution: 1) Define a module-global flag that indicates when the thread should exit as follows:

```
int exitMyThread = 0;
```

2) Set the flag in the clean-up function and wake up the thread as follows:

```
exitMyThread = 1;
mb(); /* memory barrier - ensure flag update is visible */
ix_ossl_thread_kill(threadHandle);
```

3) Inside the thread main processing loop regular checks should be done to see if the flag is set. If the flag is indeed set, the thread should exit. For example:

```
if (exitMyThread == 1)
{
return;}
```

If the main processing loop waits on a mutex or semaphore on each iteration, ix_ossl_thread_kill() use is not required. Set the flag and unlock/post the mutex/semaphore for the thread on which it is blocked, making sure that the flag check is placed in the thread immediately after the mutex/semaphore lock.

Ethernet NPE Corrupts SDRAM Memory

Reference #:3035

Product: Intel® IXP400 Software

Version(s): 1.3

Description: Under heavy traffic conditions, the A0-variant of the IXP425 network processor may experience a system crash while using NPE microcode images that include cryptographic processing on NPE C (also known as Ethernet NPE B). NPE C can handle the processing of Ethernet traffic and cryptographic processing simultaneously. However, under extremely heavy traffic loads — where the processor is processing Ethernet traffic at or near line rate of 100 Mbps and cryptographic processing is occurring on NPE C at the same time — the NPE may mishandle mBuf pointers to SDRAM data. This results in corrupted SDRAM data and may result in a system crash. This is a software issue with the NPE microcode image that runs on NPE C, and is present in IXP400 software v1.3. This issue has been fixed in the NPE microcode images provided in IXP400 software v1.4.

Implication: The effects of SDRAM memory corruption on a running system may vary in their manifestation. For a system design to be exposed to the above issue, all of the following criteria must be true:

1. The system must be using an A0-stepping IXP425 network processor.

2. The system must employ IXP400 software 1.3.
3. NPE C must be running a microcode image that provides both Ethernet processing and crypto processing.
4. Ethernet traffic received by NPE C must be near 100 Mbps at the same time the IxCryptoAcc API is being used for cryptographic processing.
5. Transmit traffic is not affected by this issue. Many designs use NPE C as an uplink port to a lower-speed WAN link, which would be incapable of generating the amount of traffic necessary to encounter this issue.

Resolution: The resolution is to apply the Intel® IXP400 Software microcode patch for the crypto. The patch is available on Intel's Web site at (http://developer.intel.com/design/network/products/npfamily/download_ixp400.htm).The patch includes the updated version of the header file for the NPE-based Ethernet and Crypto interfaces for each of the Intel® IXP400 Software v1.3, and v1.4 release, respectively. Note that the NPE microcode images are only supported on the IXP400 software version for which the image was released.

IX_ACC_DATA_CACHE_FLUSH Does Not Drain the MMU Write Buffers

Reference #:3039

Product: Intel® IXP400 Software

Version(s): 1.3

Description: On a data cache flush, the write operations are not complete until the MMU internal write buffers are empty and the AHB write requests are completed.

Implication: When exchanging data with NPEs with a limited number of mbufs in the queues (approximately less than 4), the NPE may be fast enough and might read the mbuf header or payload from SDRAM before the Intel XScale® processor write operations are complete.
In Linux OS, the required drain operation is always done after each cache flush operation. In VxWorks OS, the drain operation needs to be explicitly called (cachePipeFlush()) if the Write-back mode or the Write-through mode is enabled. Additional impact: CachePipeFlush() stalls the Intel XScale® processor CPU and its use may degrade the performance.

Resolution: It is necessary to drain the MMU write buffer before passing the mbufs to the NPE.

In the “file xscale_sw/src/include/IxOsCacheMMU.h”, change:

From

```
#define IX_ACC_DATA_CACHE_FLUSH(addr,size)  cacheFlush(DATA_CACHE, addr, size)
```

To

```
#define IX_ACC_DATA_CACHE_FLUSH(addr,size)
do

{ cacheFlush(DATA_CACHE, addr, size);
cachePipeFlush();

} while(0)
```

At High Throughput, 'Port Disable' Overflows the Internal FIFOs in the Ethernet NPEs

Reference #:3059

Product: Intel® IXP400 Software

Version(s): 1.3

Description: At high traffic rates, on port disable, the internal Ethernet NPE FIFO gets corrupted and it blocks/corrupts Rx traffic.

Implication: When **port disable** is issued at high traffic rates, corrupted NPE FIFOs block the incoming traffic. Additionally, 'Port Disable' reports error and NPE may crash.

Resolution: Do not perform 'Port Disable' at high traffic rates.

No Response From the Ethernet NPE When Ports are Disabled in Quick Succession After Enabling

Reference #:3072

Product: Intel® IXP400 Software

Version(s): 1.3

Description: There are two issues related to this SCR: The first is when two concurrent executing contexts might corrupt one of NPE registers, resulting in no traffic being received; the second issue is that Intel XScale® processor does not issue 'ELT access request' during 'Port Disable', delaying NPE response.

Implication: The first error condition occasionally could corrupt messaging and Rx path, or even crash the NPE. The second issue blocks the NPE responses for long time (up to 60 seconds) when **Port Disable** is issued following a **Port Enable**.

Resolution: None for the first error condition. For the second error condition, do not perform **Port Disable** less than 60 seconds after it has been enabled.

Ethernet NPEs Drops Ethernet Frames Greater Than 1522 Bytes

Reference #:3104

Product: Intel® IXP400 Software

Version(s): 1.3

Description: Erroneous counting of **too long** frames when frame size limit set to >1522 and frame longer than this limit are received.

Implication: **Too long** frames (larger than the configured frame size limit of 1522 bytes) get counted twice.

Resolution: None

IX_MBUF's m_nextpkt Field Gets Internally Overwritten by the ethAcc Component

Reference #:3118

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The ixEthAcc component internally uses the m_nextpkt field of each MBUF. The user of the IX_MBUF and IxEthAcc API must be aware that the contents of the field m_nextpkt may be overwritten for internal usage. The ethAcc component does not support packet chaining, and ignores the contents of this field in the buffers passed to ixEthAccTxPortSubmit() and ixEthAccRxFreeReplenish().

Implication: Users employing the m_nextpkt field in their application may experience error conditions.

Resolution: The user should be aware that in IXP400 software v1.4, the 'm_nextpkt' field is ignored on input (rxFreeReplenish() and txSubmit()), and zeroed on output of ethAcc (txDone and Rx callbacks).

Linux* NPE Ethernet Driver (ixp425_eth) Fails to Process Received Traffic After Port is Disabled, and Re-Enabled in Presence of Traffic

Reference #:3144

Product: Intel® IXP400 Software

Version(s): 1.3

Description: In the presence of existing traffic, if one of the ports — **ixp0** or **ixp1** — are enabled, disabled, and then re-enabled (for example, **ifconfig ixp0 up**), incoming traffic from the wire will not be received by the driver, but frames are transmitted.

Implication: No traffic is received via the NPE Ethernet port. This affects any bidirectional Ethernet communication, such as TCP, ICMP, and so on.

Resolution: Upgrade to IXP400 software v1.4, which resolves this issue.

Linux* NPE Ethernet Driver (ixp425_eth) May Crash Due to Execution of Non-Interrupt-Safe Code from Within an Interrupt Context

Reference #:3150

Product: Intel® IXP400 Software

Version(s): 1.3

Description: In the Linux Ethernet driver, a crash can occur due to the execution of non-interrupt-safe code from within an interrupt context, resulting in a kernel error message on the console. Examples of interrupts that can trigger this include a Tx-Timeout callback [dev_tx_timeout()] invoked by the NETDEV WATCHDOG context, or the EthDB database maintenance timer callback [maintenance_timer_cb()].

Implication: This issue will crash the system and require a reboot.

Resolution: Locate all the interrupt callbacks in the driver that might execute non-interrupt-safe code, and spawn a task from the interrupt callback to run this code in a task context instead. In particular, look at the callback functions dev_tx_timeout() and maintenance_timer_cb(). To run this code in a task context, move the code from these callbacks into a new function and call schedule_task() from the interrupt callback.

Statistics Returned by IxEthAcc API Functions ixEthAccMibIIStatsGet() and ixEthAccMibIIStatsGetAndClear() are not Correct in Little-Endian Mode

Reference #:3157

Product: Intel® IXP400 Software

Version(s): 1.3

Description: The statistics counters returned by ixEthAccMibIIStatsGet() and ixEthAccMibIIStatsGetAndClear() are not byte-swapped in Little-Endian mode as expected.

Implication: Non-zero MIBII statistics are reported incorrectly. For example, '1' checksum error may be reported as 16777216 (0x10000000) checksum errors.

Resolution: Each member of the struct IxEthEthObjStats, which is returned by the above functions, needs to be byte-swapped before the struct is returned by the function.

Since the byte-swapping is dependent on the MMU Endian and Coherency mode set up for the component, the IX_OSSERV_SWAP_NPE_SHARED_LONG() macro should be used in file xscale_sw/src/ethAcc/IxEthAccMac.c as follows:

```
IxEthAccStatus ixEthAccMibIIStatsGet (IxEthAccPortId portId, IxEthEthObjStats
*retStats )
{ixOsServMutexUnlock(&ixEthAccMacState[portId].MIBStatsGetAccessLock);

/* Endian-swap each member of the retStats struct */

retStats->dot3StatsInternalMacTransmitErrors =
IX_OSSERV_SWAP_NPE_SHARED_LONG(retStatsdot3StatsInternalMacTransmitErrors);
retStats->dot3StatsCarrierSenseErrors =
IX_OSSERV_SWAP_NPE_SHARED_LONG(retStatsdot3StatsCarrierSenseErrors);

return retStats;
}
```

Linux* NPE Ethernet Driver Does Not Set the Port MAC Address as Supplied by the User

Reference #:3180

Product: Intel® IXP400 Software

Version(s): 1.3

Description: If the user wishes to change the MAC address of one of the NPE Ethernet ports (ixp0 or ixp1) by issuing a command (for example, "ifconfig hw ether 0002b3010101 ixp0"), the Linux Ethernet driver may not set the MAC address correctly. This is because the parameter through which the MAC address is passed in the function dev_set_mac_address() is incorrectly used by the function.

Implication: If the MAC address is incorrectly set in the NPE, then the MAC address learning and filtering function in the NPE may not work correctly. At times, all incoming traffic directed at the port (not broadcast or multicast) might get filtered by the NPE and won't reach the Intel XScale® processor.

Resolution: The structure passed to dev_set_mac_address() as a void* needs to be cast to a “struct sockaddr *” so that the field sa_data of this struct points to the actual MAC address. This is the correct value that gets passed down to ixEthAccPortUnicastMacAddressSet() instead of the pointer reference.

TxDone Traffic on Ethernet NPE Fails to Run With a Certain Boot Configuration

Reference #:3182

Product: Intel® IXP400 Software

Version(s): 1.4

Description: TxDone traffic does not get processed by the QMgr because the QMgr flag fails to reflect the status of the **tx done** queue. This occurs after a boot from the NPE Ethernet interface. The bootloader apparently leaves the AQM queues in an undefined state. This leaves the AQM in a confused state while the queues get configured.

Implication: When Tx done traffic is not processed, the Tx done queue starts to fill. When the Tx done queue is full, the NPE stops transmitting and the Tx queue starts to fill. When the Tx queue starts filling, the software queues gets used inside ethAcc component. Following this, the system run out of memory and may crash.
Note that the traffic originated by Intel XScale® processor fails to be transmit after 128 buffers. On receiving a ping from an external equipment for each second, the system might be unable to respond as long as two minutes.

Resolution: Instead of using the QMgr dispatch loop entrypoint, a direct call to the **pull** functions in the ethAcc component drains the Rx queues and TxDone queues. Additionally, these functions help to increase the overall performance. To summarize, the Ethernet driver implementation should be changed as follows:

- All QMgr events should be processed by the QMgr interrupt, except the TxDone and Rx events.
- TxDone Queue and RxQueue should be processed by calling the internal ethAcc pull functions from a timer IRQ.

On Linux*, MAC Address Learning or Aging may Cease to Work After the IxEthDB Component Issues a Warning Message

Reference #:3188

Product: Intel® IXP400 Software

Version(s): 1.3

Description: When running on Linux, IxEthDB may report a warning about receiving an unknown event type. This is because, in Linux, the interrupts can be nested, so if two different messages are received from the NPE in quick succession, the interrupts may overlap (for example, “Ethernet DB: Event processor received an unknown event type (0x0), malformed message?”, or “Warning: trapped insertion of a duplicate MAC address in an NPE search tree”). These messages are issued when the EthDB receives a message callback that was triggered for a registered messageID, but a subsequent interrupt for an unregistered message overwrote the last message.

Implication: IxEthDB ends up losing the original message and receives a message for an unregistered callback. Depending on the message that gets lost, this may have unpredictable results. There is a possibility that the IxEthDB component on Intel XScale® processor may not be able to synchronize its copy of the MAC address learning database with the NPE. As a result of this, the Intel XScale® processor user will not be able to see what MAC addresses that are being filtered by the NPE and will not be able to provision the addresses to filter via the IxEthDB API, causing the MAC address aging in the database to stop working.

Resolution: The solution is to make the received message Interrupt Service routine to disable the interrupts until the received message is processed. This is done in the function ixNpeMhReceiveIsr() in the source file ixp400_xscale_sw/src/npeMh/IxNpeMhReceive.c in IXP400 software as follows:

From:

```
PRIVATE

void ixNpeMhReceiveIsr (int npeId)
{

IX_NPEMH_TRACE0 (IX_NPEMH_FN_ENTRY_EXIT, "Entering "
"ixNpeMhReceiveIsr\n");


/* invoke the message receive routine to get messages from the NPE */
ixNpeMhReceiveMessagesReceive (npeId);



/* update statistical info */
ixNpeMhReceiveStats[npeId].isrs++;
IX_NPEMH_TRACE0 (IX_NPEMH_FN_ENTRY_EXIT, "Exiting " "ixNpeMhReceiveIsr\n");
}
```

to:

```
PRIVATE

void ixNpeMhReceiveIsr (int npeId)
{

int lockKey;
IX_NPEMH_TRACE0 (IX_NPEMH_FN_ENTRY_EXIT, "Entering "
"ixNpeMhReceiveIsr\n");
lockKey = ixOsServIntLock ();


/* invoke the message receive routine to get messages from the NPE */
ixNpeMhReceiveMessagesReceive (npeId);

/* update statistical info */
ixNpeMhReceiveStats[npeId].isrs++;

ixOsServIntUnlock (lockKey);

IX_NPEMH_TRACE0 (IX_NPEMH_FN_ENTRY_EXIT, "Exiting "
"ixNpeMhReceiveIsr\n");
}
```

After Being Used by the Bootloader for Loading the Images, Tx Traffic may be Blocked on the NPE Ethernet Ports

Reference #:3202

Product: Intel® IXP400 Software

Version(s): 1.3

Description: Ethernet NPE soft reset and re-load of firmware does not work under certain conditions.

Implication: Ethernet NPE firmware does not get initialized properly and no packets is transmitted.

Resolution: Before performing an NPE soft reset (once the NPE has already initialized with Ethernet firmware), make sure no packets are pending for transmission. If the Ethernet cable is disconnected and some packets were waiting to be transmitted, the NPE reset and re-load of firmware in this state might not initialize the NPE properly. Once the NPE enters this state (where it is not transmitting packets after reset and reload) a hard reset of the system is necessary.

Timestamp Register Gives Incorrect Results on Linux* When Console UART is set to Low Speed (such as 9600 bps)

Reference #:3217

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The baud rate of the UART on Linux appears to affect the accuracy of the timestamp register on the internal APB bus of the Intel® IXP42X product line. This register is used by functions such as ixOsServTimestampGet() to retrieve timestamps in terms of **ticks per second**, where each tick corresponds to a clock cycle of the APB bus clock (66.66666 MHz). This is possibly related to the fact that the UART FIFOs and configuration registers and the timestamp register, as well as other configuration and status registers, share this internal APB bus.

Implication: This issue affects the accuracy of the timestamps retrieved from ixOsServTimestampGet(), or any other timing information retrieved from other functions also using this register. For example, the throughput rates reported by the IxDmaAcc Codelet are incorrect (lower than expected) when the codelet is executed.

Resolution: The current resolution is to increase the UART baud rates in Linux to 115200 bps, or a sufficiently high rate.

While Using the EthAcc Component, ixEthAccPortDisable() Fails and Outputs the Following Message: “[fatal] IXETHACC:ixEthAccPortDisable: ixEthAccPortDisable Failed port 1 (state = 4)”

Reference #:3263

Product: Intel® IXP400 Software

Version(s): 1.4

Description: Due to certain timing characteristics, the ixEthAccPortDisable() function may fail with the above error message. ixEthAccPortDisable() uses a state-machine

to implement the port disable procedure. If the state-machine does not reach its final state within a fixed timeout, an error such as the one above is reported. This state machine is incomplete, as it does not cover all potential event sequences that may be affected by timing characteristics.

Implication: If ixEthAccPortDisable fails while iterating through its state machine, the Port Disable sequence will not be completed. The result of this is that if the port is re-enabled it may not function as expected. Additionally, there is a possibility that some of the buffers that were supplied to the RxFree and Tx queues might get lost and never returned to the user.

Resolution: Changes to the file “ixp400_xscale_sw/src/ethAcc/IxEthAccMac.c” are required. Contact your Intel representative.

Memory Leak Occurs in Linux* Kernel Patched with "ebtables" and "br-nf" Patches while Bridging Traffic Through NPE Ethernet Port with Linux NPE Ethernet Driver

Reference #:3321

Product: Intel® IXP400 Software

Version(s): 1.4

Description: This issue occurs only on Linux kernels with the bridge-netfilter "br-nf" patch. On setting up a bridge between the two NPE Ethernet ports (ixp0 and ixp1), when traffic is passed in both directions across the bridge, the memory attached to the "nf_bridge" pointer does not get released and results in a memory leak.

Implication: At a reasonably high rate of traffic flowing across the bridge, as the system runs low on memory, the kernel kills off most running processes one-by-one until finally the **init** task is killed and the only things left running in the system are the interrupt handlers. The system is completely unusable at this point.

Resolution: The resolution is to apply the Intel® IXP400 Software Linux Ethernet Device Driver v1.1 patch available on Intel's Web site under the Open Source Components/Patches link (http://www.intel.com/design/network/products/npfamily/ixp400_osc.htm).

Linux* Xcycle Measurement in PerfProfAcc Access Component is Non-Functional

Reference #:3330

Product: Intel® IXP400 Software

Version(s): 1.4

Description: Xcycle measurement in PerfProfAcc Access component does not work since this functionality was not implemented.

Implication: Xcycle is needed in measuring CPU utilization for perfProfAcc access component in IXP400 software v1.4. However, Xcycle measurement in perfProfAcc was not supported in Linux platform. Alternatively, Linux utility “top” can be used to measure the CPU utilization.

Resolution: Linux utility “top” should be used to measure the CPU utilization.

While Using the IxEthAal5App Codelet, the ATM Service Parameter PCR (Peak Cell Rate) Does Not Limit the Traffic to a Precise Value as Expected

Reference #: 3331

Product: Intel® IXP400 Software

Version(s): 1.4

Description: While using the IxEthAal5App codelet, the VBR service does not perform precise calculation. As an example, even though the PCR is set to 8 Mbps for a single VBR channel, the actual traffic being received could be 7.8 Mbps.

Implication: This lack of precision and the VBR service does not comply with the standard.

Resolution: None.

Ethernet NPE Fails to Respond to a Port Disable Message After the EthAcc Access Component ixEthAccPortDisable( ) is Called

Reference #: 3347

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The Ethernet database “ethDB” learning in the Intel XScale® processor should be stopped before disabling the NPE. If this is not done then the NPE messages gets blocked by the Intel XScale® processor while performing an ELT maintenance on issuance of a 'port disable' message.

Implication: When ixEthAccPortDisable( ) is called, the NPE fails to respond to any messages.

Resolution: Disable the “ethDB” learning in Intel XScale® processor before using the EthAcc access component ixEthAccPortDiable( ) function to disable the Ethernet port.

Intel XScale® Processor does Not Respond to NPE Rebalancing Request after a Port Disable/Enable Cycle

Reference #: 3385

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The error condition occurs because the learning feature of the NPE gets enabled in a very early stage (during setting MAC address in NPE), before the ethDB is ready to handle any ELT requests from the NPE — in this case 'rebalancing requests'.

Implication: The Intel XScale® processor ignores any ELT rebalancing request from NPE, which, in turn, blocks the NPE messaging.

Resolution: None

Linux* NPE Ethernet Driver Does Not Behave Correctly with Netfilter* Firewall Option is Selected in the Linux Kernel

Reference #: 3395

Product: Intel® IXP400 Software

Version(s): 1.4

Description: When using the Netfilter* firewall feature with Linux NPE Ethernet driver (ixp425_eth.c), the Linux NPE Ethernet driver recycles the buffers from the kernel for all packets being received, and in doing so it fails to re-initialize certain buffer header fields, which are used by the firewall software within the kernel. This causes the firewall software to process the packets incorrectly.

Implication: When firewall rules are enabled, packets may be incorrectly dropped by the firewall when they should be allowed through, and packets are allowed through when they should be dropped.

Resolution: The resolution is to apply the Intel® IXP400 Software Linux Ethernet Device Driver v1.1 patch available on Intel's Web site under the Open Source Components/Patches link (http://www.intel.com/design/network/products/npfamily/ixp400_osc.htm).

Ethernet NPE may Incorrectly Filter Incoming Frames Destined for Other Network Interfaces in the System

Reference #:3414

Product: Intel® IXP400 Software

Version(s): 1.4

Description: The issue occurs when NPE and non-NPE Ethernet interfaces (for example, PCI) are interconnected in the same system via bridging or routing software. When a host device on an Ethernet interface migrates from a non-NPE Ethernet interface to a NPE interface, the host device will no longer be able to receive packets destined to it from a non-NPE Ethernet interface. This issue occurs because it takes approximately 15 minutes for an address to age-out of the database. In this case, the MAC address gets recorded by the Ethdb component before the host device migrates to a new network and the subsequent interface assumes that the host device is still connected to the first Ethernet interface.

Implication: If a MAC address of a host device, such as a wireless PC, migrates from a network connected to an NPE interface to a network connected to another (non-NPE) interface on the system (such as a WLAN PCI Ethernet interface), the NPE may incorrectly drop received frames destined for that MAC address as if the host device were still connected to the NPE interface.

Resolution: The resolution is to apply the Intel® IXP400 Software Linux Ethernet Device Driver v1.1 patch available on Intel's Web site under the Open Source Components/Patches link (http://www.intel.com/design/network/products/npfamily/ixp400_osc.htm).

ixEthDBTriggerPortUpdate() Crashes if Called when EthDB Learning Feature is Disabled Using IxFeatureCtrl

Reference #:3437

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The ixEthDBTriggerPortUpdate() function is indirectly invoked by several API functions, including ixEthDBDatabaseMaintenance(). The system will crash when a call is made to the ixEthDBTriggerPortUpdate() function when the

Learning feature is disabled. The system crashes when the port update function attempts to increment a semaphore that doesn't get initialized if the Learning feature is disabled.

Implication: The ixEthDBTriggerPortUpdate() function call and other API calls using this function cannot be used if the EthDB learning feature is disabled using IxFeatureCtrl.

Resolution: None.

Unresolved Symbols Messages Appear During Linux* Boot Process

Reference #:3454

Product: Intel® IXP400 Software

Version(s): 1.4

Description: The following two unresolved symbols messages appear during the Linux boot process. The messages occur because of the Linux kernel's “make modules_install” rule being used to build the target file-system.

```
Calculating module dependencies... depmod:

*** Unresolved symbols in /lib/modules/2.4.18_mvl30-ixdp425/kernel/drivers/
ixp400/ixp400_dmaAcc.o depmod:

*** Unresolved symbols in /lib/modules/2.4.18_mvl30-ixdp425/kernel/drivers/
ixp400/ixp400_perfProfAcc.o done.
```

Implication: These unresolved symbols messages have no impact on the successful operation of the IXP400 software.

Resolution: To stop the files from being included in the dependency check, edit the file drivers/ixp400/Makefile in the kernel source tree to change the line.

From:

```
IXP400_MODULES := ixp400.o $(COMPONENTS:%=ixp400_%.o)
$(CODELETS_COMPONENTS%=ixp400_codelets_%.o)
```

To:

```
IXP400_MODULES := ixp400.o
$(CODELETS_COMPONENTS:%=ixp400_codelets_%.o)
```

Crypto Request gets Rejected for "HMAC-MD5" Only Operation Following a De-Registration by the WEP Services

Reference #:3742

Product: Intel® IXP400 Software

Version(s): 1.4

Description: The Crypto request gets rejected if the internal crypto context structure, de-registered from WEP services, gets reused for HMAC-MD5 only operation. This is because during the registration process, the request type set to “WEP_REQ” in previous crypto context does not get updated to “HW_ACCL_REQ” in HMAC-MD5 only operation. When invoking the API to submit HMAC-MD5-only operation requests, the API call does a check on the request type to make sure

the request type matches with the API call before proceeding to submit the packet to NPE. In this case, the request type remains as “WEP_REQ” while the API is for “HW_ACCL_REQ”; therefore the request gets rejected.

Implication: This issue occurs only if the crypto context for WEP are being reused (unregistered for WEP, and being registered for MD5) for HMAC-MD5-only operation. While registering for MD5, the **reqType** field in the crypto context will hold the old value of WEP_REQ instead of HW_ACCL_REQ and therefore all the HW_ACCL requests to this context is rejected. This issue does not occur while registering for a HMAC-SHA1 only operation.

Resolution: Add the following line in the source file **IxCryptoAccCryptoRegister.c** located under
xscale_sw/src/cryptoAcc. This code resets the **reqType** in ixCryptoAccCtx context for AUTH_MD5.

```
* Set Req Type to crypto Hw Accelerator service in crypto context
*/ ixCryptoCtx[cryptoCtxId].reqType = IX_CRYPTO_HW_ACCL_REQ;
```

Linux* NPE Ethernet Driver and EthAcc Codelet Does Not Support Intel® LXT973 Revision A3 MII ID 0x00137a11

Reference #:3847

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: Revision A3 of Intel® LXT973 PHY seems to be non functional when used with the EthAcc codelet or the Linux NPE Ethernet driver. The Revision A3 PHY will not be recognized when used because the ethMii code in the EthAcc component does not support Revision A3 PHY ID as specified in the LXT973 specification update.

Implication: The EthAcc codelet and the Linux Ethernet driver will not support the LXT973 Revision A3 PHY.

Resolution: PHY ID needs to be updated to ID **0x00137a11**. Contact your Intel representative to obtain the modified code to resolve this issue.

The Queue Descriptor ‘get’ Fails During Rev Key Generation

Reference #:3864

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The ixCryptoCCDMgmtKeyCryptoParamRelease function call in ixCryptoRegisterRevAesKeyGenerate function should not have any return status. The return status overwrites the status of Queue write fail/full, and this changes the status of Queue descriptor get to success/fail.

Implication: Depending upon the operation, there are two implications for this:

1. AES decryption-only operation

In this case, the client's register callback will never be called. However, for the rev key generation, the client callback is called in QMgr dispatcher context. The crypto context will not be marked as valid, and therefore the crypto context is invalid. If the client submits the request by using this crypto context, all the requests will get rejected due to an invalid crypto context.

2. AES decryption + HMAC verification

For this case, client callback for rev AES key generation will never be called. However, the crypto context is marked as valid after the registration of the authentication algorithm. The rev AES key gets updated into the crypto param structure to an incorrect random value. This defect will cause the decryption to be wrong as the AES key being used for decryption is wrong. In this case, the authentication will still pass as the digest gets calculated before the cipher text get decrypted. Thus, this defect should not cause the cryptoAcc/ cryptoNPE to hang.

Resolution: The resolution for this is to change the code in file IxCryptoAccCryptoRegister.c located under **xscale_sw/src/cryptoAcc**as shown below:

Search for 2 occurrences of function call to ixCryptoCCDMgmtKeyCryptoParamRelease () in function ixCryptoRegisterRevAesKeyGenerate(), and remove the return status:

From:

```
status = ixCryptoCCDMgmtKeyCryptoParamRelease (keyId);
```

to:

```
ixCryptoCCDMgmtKeyCryptoParamRelease (keyId);
```

The OSSL Component's TICKS_PER_NSEC Macro Returns an Incorrect Value

Reference #:3879

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: The name of the macro TICKS_PER_NSEC currently equates to nsec per tick and returns an incorrect value.

Implication: The macro returns nsec per tick instead of ticks per nsec.

Resolution: Change the macro

From:

```
#define TICKS_PER_NSEC (OEM_CLOCK_FREQ/BILLION)
```

to:

```
#define NSEC_PER_TICKS (BILLION/OEM_CLOCK_FREQ
```

HSS Access Component Fails to Initialize When Two Specific NPE Images are Used

Reference #:3928

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: HSS access component fails to initialize when one of the following NPE images are used:

- IX_NPEDL_NPEIMAGE_NPEA_HSS0_ATM_SPHY_1_PORT
- IX_NPEDL_NPEIMAGE_NPEA_HSS0_ATM_MPHY_1_PORT

Depending on which one is called first, the ATM component or the HSS component will fail to initialize. If the HSS is called before ATM, ATM will fail. If ATM is called before HSS, HSS port '1' will fail.

Implication: NPE images listed above cannot be initialized, and HSS and ATM services cannot be supported simultaneously.

Resolution: Contact your Intel representative to obtain the modified code to resolve this issue.

Cryptographic Processing on NPE C may Halt for a Specific Sequence of Requests to Register and Unregister MD5-Authentication Cryptographic Context

Reference #:3987

Product: Intel® IXP400 Software

Version(s): 1.4, 1.3

Description: When a specific sequence of requests using the IxCryptoAcc access-layer component of the Intel® IXP400 Software occurs, the cryptographic processing on NPE C may halt. The root cause of the issue is related to a timer configuration for the Hashing coprocessor within NPE C. This issue is timing-sensitive, and the runtime required to encounter this issue is unpredictable. The crypto contexts that can induce the issue are limited to MD5 authentication operations. Encrypt-Only or Decrypt-Only operations and SHA-1 or WEP-CRC authentication operations cannot cause the error condition.

Implication: When the issue occurs, the cryptographic processing capabilities on NPE C will stop functioning. Anyway, Ethernet activity on NPE C can still be processed and the Intel XScale® processor and remaining NPEs will continue to operate.

Resolution: The resolution is to apply the Intel® IXP400 Software microcode patch for the crypto. The patch is available on Intel's Web site at (http://developer.intel.com/design/network/products/npfamily/download_ixp400.htm). The patch includes the updated version of the header file for the NPE-based Ethernet and Crypto interfaces for each of the Intel® IXP400 Software v1.3, and v1.4 release, respectively. Note that the NPE microcode images are only supported on the IXP400 software version for which the image was released.

Use of ixEthDBFilteringPortMaximumFrameSizeSet( ) API Call Causes a Crash if Only One of the Ethernet NPEs is Initialized

Reference #:4096

Product: Intel® IXP400 Software

Version(s): 1.4

Description: The Intel Access Layer component EthDB sends messages to both the Ethernet NPEs while only one NPE is initialized for setting up the maximum frame length. This causes a crash in the Intel XScale® processor code. The EthDB code assumes that when the function ixEthDBFilteringPortMaximumFrameSizeSet is called, all Ethernet NPEs are already initialized.

Implication: When ixEthDBFilteringPortMaximumFrameSizeSet( ) is called, and if only one of the NPE is initialized, a crash occurs.

Resolution: The resolution for this issue is to add a check marked below in red, in IxEthDBAPISupport.c located in the xscale_sw/src/EthDB folder:

```
/* update all NPEs with new settings : send the first message */

for (port = 0; port < IX_ETH_DB_NUMBER_OF_PORTS; port++)

{ if ((ixEthDBPortDefinitions[port].type == ETH_NPE) &&

            (ixEthDBSingleEthNpeCheck(port) == IX_ETH_DB_SUCCESS) &&
(ixEthDBPortInfo[port].dependencyPortMap != 0))
```

ATM Scheduler Ignores the Peak Cell Rate (PCR) Value

Reference #:00049277

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: In VBR mode, setting the PCR value has no effect on the number of cells allowed to pass through the ATM scheduler. This is due to an ATM scheduler design issue.

Implication: The ATM scheduler is not able to control the ATM bandwidth using a PCR value by allowing a higher number of cells than is requested to be transmitted.

Resolution: None

Linux* NPE Ethernet Driver Drops Packets for an Oscillating Traffic at a High Traffic Rate

Reference #:00049291

Product: Intel® IXP400 Software Linux* Ethernet Device Driver

Version(s): 1.5, 1.4

Description: By default the netdev_max_backlog value for the Linux NPE Ethernet Driver is set to a high value. When oscillating traffic is encountered, the driver replenishes more aggressively by providing more rxfree buffers to the NPE compared to the networking queue. As a result, the driver will push more buffers to the networking queue than the rate at which the Linux kernel can accept them, which causes the packets to be dropped.

Implication: For oscillating traffic at a high rate, the Linux NPE Ethernet Driver will have lower packet per second performance than it would with steady state traffic.

Resolution: For an oscillating traffic, use the following command/parameters to load the Ethernet driver at the console:

```
echo 300 >/proc/sys/net/ipv4/netdev_max_backlog

insmod ixp400_eth.o netdev_max_backlog=360
```

In Linux*, the IxCryptoAccCodelet Cannot be Removed Using the "-1" Option

Reference #:00049294

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: According to the description in the IxCryptoAccCodelet header file, the user should be able to remove and stop the IxCryptoAccCodelet by passing a "-1" flag to the ixCryptoAccCodeletMain function from the console.

Implication: While executing IxcryptoAccCodelet, passing "-1" as the service Index to the ixCryptoAccCodeletMain function does not remove the module or stop the codelet.

The service index "-1" should not be used for unloading the module.

Resolution: To unload the module or to stop the codelet, use the following commands:

```
# rmmod ixp400_codelets_cryptoAcc

# rmmod ixp400
```

The IxPerfProfAccCodelet is Incompatible with the Linux* NPE Ethernet Driver Running in a Polled Mode

Reference #:00049362

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: In polling mode, the Linux NPE Ethernet driver uses the PMU timer's interrupt. When the IxPerfProfAcc codelet is loaded while the Ethernet driver is in polling mode, the IxPerfProfAcc component tries to reserve the PMU timer interrupt and will implicitly de-register it from the Ethernet driver. This, in turn, causes the Ethernet driver to stop working.

Implication: If the IxPerfProfAccCodelet and the Linux NPE Ethernet driver are used simultaneously, and if the Ethernet driver is enabled to run in polled mode (which is the default behavior), the component that is loaded last is in charge of the interrupt and the component that was loaded first will not work.

Resolution: Using the IxPerfProfAccCodelet while using the Linux NPE Ethernet driver can be accomplished by configuring the Linux driver to run in interrupt mode. The ixp400_eth.o module must be loaded using the parameter: "datapath_poll=0". Note that this resolution may affect the performance.

IxEthAcc Control Plane APIs are not Re-Entrant

Reference #:00049371

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: The Control Plane APIs, as declared in the `IxEthAcc.h` header file, for the IxEthAcc access-layer component are not re-entrant. This is because there is no locking provided in the API implementation.

Implication: Calling Control Plane APIs from different threads at the same time can produce unpredictable results. In the absence of locking, this could lead the MAC hardware to be accessed by concurrent threads that are using the same API.

Resolution: The user application would have to use a locking or a single-thread model for configuring and enabling the IxEthAcc access-layer component to prevent the API from being re-entrant.

IxEthAal5AppCodelet Counters Displays Inconsistent Results

Reference #:00049382

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: While making use of the IxEthAal5AppCodelet, the **ethRxForwarded** and the **atmRxForwarded** counters displays incorrect values. This is because the ixEAANumAtmRxDropBuffers variable in the ixEAAAtmTxDoneCallback and the ixEAANumEthRxDropBuffers variables in the ixEAAEthTxDoneCallback function does not get updated for an invalid PDU or an invalid Ethernet frame, respectively.

Implication: The IxEthAal5AppCodelet counters displays incorrect drop count result.

Resolution: The correct value of the dropped count can be calculated by computing the difference between the atmRxFrames/ethRxFrames counter and the atmRx-Forwarded/ethRxForwarded counters, respectively.

Very Slow MII Access via 'ixEthAccMiiReadRtn()' and 'ixEthAccMiiWri-teRtn()'

Reference #:00049619

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: The default MII command completion timeout (100 ms) could be too large for certain applications and implementations, causing unnecessary delays.

Implication: MII access via 'ixEthAccMiiReadRtn() and 'ixEthAccMiiWriteRtn()' could be unreasonably slow — about 100 milliseconds per access. The reason is the functions have been implemented in a while loop with a delay of 100 milli-seconds to allow the NPE to perform the MII accesses. This might not show up on a device where just a simple PHY has to be set up, but in some devices where more registers have to be set up to configure the switch, it might have a significant impact.

Resolution: To resolve this issue, the value of variable IX_ETH_ACC_MII_10TH_SEC_IN_MILLIS, in ixp400_xscale_sw/src/ethAcc/include/IxEthAccMii_p.h should be changed from 100 millisecond to one milli-second.

BSS Memory Usage is High in IxPerfProfAcc Access-Layer Component

Reference #:00050954

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: The IxPerfProfAcc component uses a large amount of BSS (block starting symbol) memory, which is used to store static variables. The BSS memory is used to store the program counter addresses when an interrupt is called in the interrupt handler, and to hold the sample results.

Implication: Users that include the IxPerProfAcc component in their build might experience **out of memory** system warnings, and eventually the system might hang.

Resolution: Two recommendations for minimizing the impact are:

1. Make IxPerfProfAcc a non-standard module. By default, IxPerfProfAcc should not be included in the standard build. To make IxPerfProf a non-standard module, the **perfProfAcc** tag should be removed from the component list and the codelet list in the makefile located in the \ixp400_xscale_sw directory.
2. Reduce the initial number of samples to a lower number by changing it from 0xFFFF to 0X03E8(~1000). This can be achieved by lowering the count being assigned to #define IX_PERFPROF_ACC_XSCALE_PMU_MAX_PROFILE_SAMPLES located in the IxPerfProfAcc.h header file under the ixp400_xscale_sw\src\perfProfAcc directory.

In VxWorks*, Multiple Calls of ixOsalIrqBind() for the Same Interrupt Returns IX_SUCCESS Instead of IX_FAIL

Reference #:00053521

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The interrupt bind API ixOsalIrqBind returns success when called more than once for the same IRQ. This is because the function returns an error only if the parameter being passed happens to be an invalid parameter. It does not check for multiple calls to bind the same interrupt.

Implication: Multiple calls to connect to the same interrupt will result only in the last being bound to a given interrupt.

Resolution: None. The user should make sure that the IRQ bind is done only once on the desired interrupt request handler.

Under MontaVista* Linux the Intel® IXP400 Software Does Not “rmmod” Cleanly

Reference #:00053591

Product: Intel® IXP400 Software

Version(s): 1.5, 1,4, 1.3

Description: Under MontaVista* Linux, when **rmmod** is used to unload the ixp400.o module, not all system resources are released/freed.

Implication: If ixp400.o is unloaded with rmmod, unpredictable system results may occur. For example, a MontaVista Linux system may crash while performing a cat on /proc/interrupts after the ixp400.o is removed by a **rmmod** command.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

Intel® IXP400 Software Linux* Integration Patch 1.1 Fails With MontaVista Linux 3.1 IXDP425 LSP Update

Reference #:00053596

Product: Intel® IXP400 Software Linux* Integration Patch

Version(s): 1.1 (for IXP400 software v1.4)

Description: The IXP400 software Linux Integration patch 1.1 expects trailing spaces in the MontaVista Linux 3.1 LSP kernel configuration file but the August 30, 2004 update has removed these spaces.

Implication: IXP400 software Linux Integration patch 1.1 will not apply to the MontaVista Linux 3.1 LSP update (August 30, 2004).

Resolution: To obtain a resolution for this issue, contact your Intel representative.

Use of ix_ossl_malloc API Returns Compiler Error

Reference #:00053660

Product: Intel® IXP400 Software

Version(s): 1.5

Description: Applications using ix_ossl_malloc() function will result in compilation errors. This is because the parameter for ix_ossl_malloc being declared in the macro should have been **arg_Size** instead of **arg_size**.

Implication: Use of ix_ossl_malloc results in an **arg_size** undeclared compilation error.

Resolution: Change **arg_Size** to **arg_size** in the ix_ossl_malloc macro in ixp400_xscale/src/include/IxOsalBackwardOssl.h.

IxEthAccCodelet Does Not Filter Out-Of-Range VID

Reference #:00053661

Product: Intel® IXP400 Software

Version(s): 1.5

Description: When the VLAN/QoS feature is enabled it starts in a permissive configuration, the entire VLAN range of 0-4095 gets added to the port membership table by default. As a result of this, while using the IxEthAccCodelet the user is required to remove the VLAN range from the port membership table prior to assigning a desired VLAN ID (VID) range (example 100-200) through use of ixEthDBPortVlanMembershipSet API function to configure a port, or else this will result in all VLAN tag IDs (0-4095) to be accepted.

Implication: If traffic is transmitted to the Ethernet port with an out-of-range VLAN tag, the traffic gets forwarded. This is not expected behavior as only valid VID values of a specified range should be passed.

Resolution: In IxEthAccCodeletSwBridgeQoS.c, the global VLAN range must be removed before adding the desired range. Note that '0' is kept in the membership table by default so that untagged frames are not filtered.

Add the following code in IxEthAccCodeletSwBridgeQoS.c, (before making a call to the ixEthDBPortVlanMembershipRangeAdd() function):

```
if (ixEthDBPortVlanMembershipRangeRemove(firstPortId, 1,
IX_ETH_DB_802_1Q_MAX_VLAN_ID) != IX_ETH_DB_SUCCESS)

{

    printf("SwBridgeQos: Failed to set VLAN membership for port
%u\n", firstPortId);

    return (IX_FAIL);

}
```

Inaccuracy in Timer when using ixOsalSysClkRateGet() API

Reference #:00053691

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The ixOSAL API ixOsalSysClockRateGet() returns a count of 60 when used in VxWorks and Linux instead of returning a count of 66660000. This affects the performance of the ATM Codelet in VxWorks and Linux.

Implication: The periodic poll task of the ATM codelet is affected. As an example, the time of 15 seconds for polling is shorter in VxWorks and Linux. This affects performance under heavy traffic.

Resolution: Use ixOsalTimestampResolutionGet() in place of ixOsalSysClockRateGet(). This function will return 66660000 for all the operating systems.

IxCryptoAccCodelet Does Not Display any Output in VxWorks* on the Tornado* WindShell

Reference #:00053693

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The IxCryptoAccCodelet directs all the output to the serial console and does not display the
user-level output to the Tornado* WindShell.

Implication: If a serial console is not connected to the target platform, the menu and statistical output from the codelet will not be visible to the user. As in the case of other codelets, it is expected that the output is displayed on the Tornado* WindShell while using the ixCryptoAccCodelet.

Resolution: Attach a serial console to view the output. Alternatively, the appropriate ixOsalLog() function calls can be replaced with printf statements in the IxCryptoAccCodelet.c source file.

Inconsistency with Inter Frame Gap in IxEthAccMac Access-Layer Component

Reference #:00053699

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4, 1.3

Description: The Inter Frame Gap defined in the ixEthAccMac header file is in non-compliance to the IEEE802.3 standard. This is because the Inter Frame Gap is currently set to be 76 bits. The IEEE802.3 standard requires the Inter Frame Gap to be 96 bits.

Implication: This non-compliance may result in a slightly higher transmit rate than the rate specified in the IEEE802.3 standard.

Resolution: The value for IX_ETH_ACC_MAC_TX_DEFER_DEFAULT defined in ixp400_xscale_sw/src/ethAcc/include/IxEthAccMac_p.h should be changed from 0x10 to 0x15.

Possible Re-Entrancy Issue with ATM Access-Layer Component

Reference #:00053798

Product: Intel® IXP400 Software

Version(s): 1.5, 1,4, 1.3

Description: The function ixAtmdAccTxDoneDispatch(), which is used to process the ATM transmit done queue by the IxAtmmAcc and IxQMgrAcc access-layer components, is not re-entrant. This is because the function uses a static array to pull the buffer pointers from the transmit done queue for processing.

Implication: There may be a variety of symptoms when multiple instances of the ixAtmdAccTxDoneDispatch() function access the same array or buffer pointers. These include multiple attempts to free the same buffer or memory leaks. Also, PDU transmissions may never be tracked to completion, which may lead to not being able to delete a VPI/VCI.

Resolution: The user must ensure the function ixAtmdAccTxDoneDispatch() is protected from concurrent invocations. To do this, the user needs to determine how the Queue Manager dispatcher function (returned by a call to ixQMgrDispatcher-LoopGet(...)) and the ATM transmit done function (ixAtmmTxDoneHandle(...)) are invoked in their system. Some of the invocation options are via interrupt and/or polling via timer interrupt and/or task level polling. With the invocation information above, the user must ensure the proper locking is put in place to prevent reentrant calls to the ixAtmdAccTxDoneDispatch() function.

In VxWorks*, Loadable Modules Doesn't Include NpeMicrocode.o

Reference #:00053825

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The Makefile links the NpeMicrocode.o into VxWorks.st, but does not link the object file into VxWorks loadable modules.

Implication: Running Intel® IXP400 Software loadable modules on top of VxWorks.st from a previous release will fail.

Resolution: Users should rebuild the vxWorks.st image in Intel® IXP400 Software.

VxWorks* Build Fails in a Microsoft* Windows Environment

Reference #:00053847

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The IXP400 software makefile has an error. Building libIxp425 with "make libIxp425 IX_TARGET=vxbe" option would result in a build errors on a Windows* environment. This is because of an incorrect separator in the makefile.

Implication: Building a make with “make libIxp425 IX_TARGET=vxbe” option would result in build errors.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

Linux* NPE Ethernet Driver: Typo Mistake has an Impact on Performance Enhancements

Reference #:00053897

Product: Intel® IXP400 Software

Version(s): 1.5

Description: When CONFIG_IXP425_ETH_SKB_RECYCLE is enabled in the Linux autoconfig.h header file, a performance improvement for smaller packet size is expected. However, this is not the case. This is because of an incorrect definition in the Linux Ethernet driver source code. It is expected that there should be a reduction of CPU occupancy at the same traffic rate or a higher traffic rate at the same CPU occupancy.

Implication: lower performance is experienced for smaller packet size especially, at high traffic.

Resolution: If SKB recycle is enabled, in ixp425_eth.c, change

```
From
#ifdef CONFIG_IXP425_SKB_RECYCLE

to
#ifdef CONFIG_IXP425_ETH_SKB_RECYCLE
```

IxHssAcc Channelized Service Cannot be Restarted After Disabling the Port

Reference #:00054053

Product: Intel® IXP400 Software

Version(s): 1.5, 1,4, 1.3

Description: When a call is made to the ixHssAccChanPortDisable function to disabled the Channelized service for the first time, the client callback gets removed. When a call is made to the ixHssAccChanPortEnable function to restart the service, the client callback does not get registered again.

Implication: The IxHssAcc channelized service cannot be restarted after it is disabled.

Resolution: To disable and restart the IxHssAcc Channelized service, use the following command sequence at the console:

```
#ixHssAccChanDisconnect

#ixHssAccChanConnect

#ixHssAccChanPortEnable
```

IxEthAccCodelet Does Not Convert 802.11 Frames to 802.3 Frames

Reference #:00054344

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The "Bridge + WiFi header conversion" routine in the IxEthAccCodelet does not convert 802.11 frames back to 802.3 frames as expected.

Implication: The "Bridge + WiFi header conversion" routine in the IxEthAccCodelet cannot be used to demonstrate 802.11 to 802.3 frame header conversions.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

IxEthAccRxFrameType Enumerations Definition in IxEthAcc.h Defines Wrong Values

Reference #:00054427

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The enumeration definition in IxEthAccRxFrameType is incorrectly defined and documented in IxEthAcc.h.

Implication: This defect does not have any implications for the Intel® IXP400 Software v1.5 as none of the values defined by IxEthAccRxFrameType are used in IXP400 software. However, it can introduce errors when used in user application code.

Resolution: Replace the definition of IxEthAccRxFrameType enumeration in IxEthAcc.h located under ixp400_xscale_sw\src\include as follows:

From:

```
typedef enum

{

IX_ETHACC_RX_LLCTYPE = 0x0000,

IX_ETHACC_RX_ETHTYPE = 0x1000,

IX_ETHACC_RX_STATYPE = 0x2000,

IX_ETHACC_RX_APTYPE  = 0x3000

} IxEthAccRxFrameType;
```

To:

```
typedef enum

{

      IX_ETHACC_RX_LLCTYPE = 0x00, /**< 802.3 - 8802 (Witl LLC/
SNAP) */

      IX_ETHACC_RX_ETHTYPE = 0x10, /**< 802.3 without LLC/SNAP */

      IX_ETHACC_RX_STATYPE = 0x20, /**< addressing to a STA */

      IX_ETHACC_RX_APTYPE  = 0x30  /**< addressing to an AP */

} IxEthAccRxFrameType;
```

Memory Leaks in VxWorks* When Thread Exits or is Killed

Reference #:00054556

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The task stack memory and the Task Control Block (TCB) region defined within the stack area gets used by tasks or threads that gets created by using ixOsalThreadCreate API. When a task or thread exits or gets killed, the task stack memory and the task control block does not get deallocated, resulting in memory leaks. This is due to the improper implementation of the Task Create function. The reason for this is that an option flag VX_DEALLOC_STACK is not set. This flag enables deallocation of the stack when a task or thread is exited or terminated.

Implication: The implication of this error condition is that it results in memory leakage errors when a task is deleted or killed.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

EthAcc Access-Layer Component is Unable to Replenish Filtered Frames

Reference #:00055187

Product: Intel® IXP400 Software

Version(s): 1.5

Description: When an incoming frame received from a new source is filtered by the NPE, the EthAcc access-layer component takes the mbuf and sends it to EthDB access-layer component to learn its MAC address. Since this frame is to be filtered, it is expected that the buffer sent to EthDB component is replenished once the MAC address is learned. However, the EthAcc access layer fails to replenish the buffer because the mbuf length field passed to the access-layer component by the NPE gets adjusted to the size of the header instead of the size of the entire data area. Thus, when the EthAcc component tries to replenish the mbuf with the adjusted length, it doesn't get accepted because of the mismatch in length.

Implication: If NPE MAC address learning is used, eventually all mbufs pointers in the rx_free queue of the queue manager is lost as the unlearned addresses from the new sources continues to be filtered.

Workaround: To obtain a resolution for this issue, contact your Intel representative.

Intel® IXP400 Software USB Access-Layer Component Fails to Transmit in Bulk Transfer Mode When the Client Software Sets the ENABLE_TX_SEQ Flag to '1'

Reference #:00055339

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: In a bulk transfer mode, when the ENABLE_TX_SEQ=1 flag is set, it is expected that a '0' byte packet is sent out from the device to the host after the last 64bytes packet of a multiple packet transfer takes place. However, the

Intel® IXP400 Software USB Access-layer Component does not guarantee that a '0' byte packet will always be sent in a bulk transfer mode.

Implication: In a bulk transfer mode, if the host software sets the ENABLE_TX_SEQ flag to '1', it might not receive an end of transfer especially, if it expects to receive a zero-byte packet in return to indicate an end of transfer.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

IxEthAcc Fails to Re-Initialize the Ethernet NPE Interfaces After Unloading the IxEthAcc Module

Reference #:00055503

Product: Intel® IXP400 Software

Version(s): 1.5, 1.4

Description: The ixEthAccUnload function fails to unload the IxEthDB access-layer module.

Implication: The IxEthAcc module cannot be successfully unloaded and then reloaded. When IxEthAcc module is unloaded, it is necessary to unload the ixEthDB module by making a call to the ixEthDBUnload function. However, on re-initializing the Ethernet interface, the IxEthDB module might still fail to enable the learning feature.

Resolution: To obtain a resolution for this issue, contact your Intel representative.

Spanning Tree Protocol (STP) and VLAN Filtering Does Not Work with the Header Conversion Featured NPE Image

Reference #:00056221

Product: Intel® IXP400 Software

Description: 1.5When STP and VLAN filtering features are enabled on the NPE images supporting the WiFi header conversion features, no STP filtering and VLAN filtering support is provided on the Ethernet NPE ports. The affected NPE images are IX_NPEDL_NPEIMAGE_NPEA_ETH_SPAN_FIREWALL_VLAN_QOS_HDR_CONV, IX_NPEDL_NPEIMAGE_NPEB_ETH_SPAN_FIREWALL_VLAN_QOS_HDR_CONV and IX_NPEDL_NPEIMAGE_NPEC_ETH_SPAN_FIREWALL_VLAN_QOS_HDR_CONV.

Implication: STP blocking, VLAN Acceptable Frame-Type Filtering and VLAN Membership Filtering features cannot be used simultaneously with the WiFi HeaderConversion feature.

Resolution: None.

If a Network has More Than 511 Individual MAC Addresses, the Ethernet NPE Becomes Unresponsive After Learning all of the Available MAC Addresses

Reference #:00056449

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The Ethernet NPEs have a learning and filtering capability for up to 511 MAC addresses. If a network has more than 511 MAC addresses, the Intel XScale® processor learning and filtering database attempts to learn more than 511 MAC addresses and the Ethernet NPE becomes unresponsive.

Implication: If a network node connected to an Ethernet NPE has more than 511 individual MAC addresses, and the learning/filtering feature is enabled, then the Ethernet NPE may eventually become unresponsive and fail to pass any traffic.

Resolution: None.

Unresolved Symbol ixOsalBuffPoolFreeCountGet in Linux* Build

Reference #:00055983

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The symbol ixOsalBuffPoolFreeCountGet is not exported under Linux.

Implication: Unresolved symbol error message is reported if the API is called by any component.

Resolution: Add the following line to the end of OSAL\os\linux\src\modules\bufferMgt\IxOsalOsBufferMgtSymbols.c

```
$EXPORT_SYMBOL (ixOsalBuffPoolFreeCountGet);
```

NPE Corrupts Frame on Ingress When Bidirectional Traffic in Operation

Reference #:00057654

Product: Intel® IXP400 Software

Version(s): 1.5

Description: NPE sends corrupted frames to access layer when Rx starvation (that is, Eth Rx Free queue is out of buffer) occurs. This error condition is caused by the design of NPE algorithm in flushing the MAC Rx FIFO whenever Rx starvation happens. The algorithm fails to flush all the frame bytes in the MAC Rx FIFO when traffic is interleaved with short and long frames, which results in the incorrect frames being sent to the access layer.

Implication: Corrupted frames that are submitted to user level via access layer may cause incorrect frames or incorrect IP packet. Apart from that, the multi-corrupted frames may cause the learning table to be filled with incorrect information.

Resolution: Obtain the NPE microcode version 2.0.5 which can be found in the archive of Intel® IXP400 Software v2.0 from:

http://developer.intel.com/design/network/products/npfamily/download_ixp400.htm

Ethernet Transmit Lockup

Reference #:00056585

Product: Intel® IXP400 Software

Version(s): 1.4, 1.5

Description: Chained Tx buffers cause NPE firmware to lockup its transmission when any buffer within the chain has its buffer length set to zero.

Implication: The NPE firmware is programmed to transfer data from each buffer in the chain based on the buffer length value in the buffer header. It will pass the buffer length value to the AHB Coprocessor for the AHB to transfer the data without first checking its value. The NPE will then wait for a transmit-done signal from the AHB, but the AHB may become unresponsive when it is asked to transfer zero bytes. This causes the NPE to wait for the transmit-done acknowledgement — a transmit deadlock scenario.

Resolution: The user application has to ensure that no buffer has zero-byte buffer length inserted during the transmission.

Wrong Configuration of Encryption / Authentication Operation in Crypto Codelet

Reference #:00057521

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The configuration for encryption and authentication operation in crypto codelets is not correct for all combination modes of DES, 3DES, AES, MD5, SHA1. Therefore, the performance collected with crypto codelets for encryption/authentication combination (starting from index 15 to 28) are not correct.

Implication: The operations that have been passed into the cryptoAcc are IX_CRYPTO_ACC_OP_ENCRYPT and IX_CRYPTO_ACC_OP_DECRYPT for all combination operations. These operations only perform encryption/decryption.

Resolution: When performing encryption/authentication operations, configure the operations from index 15 to 28 to IX_CRYPTO_ACC_OP_ENCRYPT_AUTH instead of IX_CRYPTO_ACC_OP_ENCRYPT, and IX_CRYPTO_ACC_OP_AUTH_DECRYPT instead of IX_CRYPTO_ACC_OP_DECRYPT in IxCryptoAccCodelet.c

HSS Port Cannot be Disconnected Successfully when Port is Busy

Reference #:00057683

Product: Intel® IXP400 Software

Version(s): 1.3, 1.4, 1.5

Description: The following error messages were returned when the client is disconnecting the HSS port by calling ixHssAccPCMDisconnect:

[error] ixHssAccPCMDisable: Tried to stop a hdlc Port that isn't connected or isn't started

[error] ixHssAccPCMDisconnect: Unable to disable this port on the NPE

Implication: There is a possibility that the ixHssAccPktTxDoneDisconnectCallback (an ISR routine) may be called before the ixHssAccPortDisable is called in the ixHssAccPCMDisconnect Public API (in IxHssAccPCM.c). If it is, the ixHssAccPktTxDoneDisconnectCallback will set a parameter called ixHssAccPCMClientInfo[hssPortId][hdlcPortId].thisIsConnected flag to 'FALSE'. This will cause the ixHssAccPortDisable function call to return a failure because

it has to check for ixHssAccPCMClientInfo[hssPortId][hdlcPortId].thisIsConnected = TRUE before proceeding to disable the HSS Port.

Resolution: To resolve this issue, the interrupt lock for the ixHssAccPktTxDoneDisconnectCallback ISR should only be released after the ixHssAccPCMPortDisable API has been called within the ixHssAccPCMDisconnect in In IxHssAccPCM.c near line 796.

```
IX_STATUS
ixHssAccPCMDisconnect (IxHssAccHssPort hssPortId,
                                      IxHssAccHdlcPort hdlcPortId)
{


...
...
<near line 850>


 /*   ixOsalIrqUnlock(lockKey); */  <---- masked off/remove this line
/* if started then stop */
    if (ixHssAccPCMClientInfo[hssPortId][hdlcPortId].thisIsEnabled)
    {
    status = ixHssAccPCMPortDisable (hssPortId, hdlcPortId);


    ixOsalIrqUnlock(lockKey);         <----- Insert this line here
    if (status != IX_SUCCESS)
      ....
      ....
    <near line 878>


    return IX_FAIL;
    }
    }
   else                               <---- Insert 4 lines below here
   {
     ixOsalIrqUnlock(lockKey);
   }
```

Buffer Length Constraint on Ethernet Rx Path

Reference #:00018713

Product: Intel® IXP400 Software

Version(s): 1.5

Description: Rx frame corruption may occur if the data buffer is set to 64 bytes when VLAN or WiFi is enabled.

Implication: When VLAN or WiFi is enabled, the payload of the frame is more than 64 bytes. The NPE firmware design does not check the sufficiency of mbuf size and chains more buffer when VLAN or WiFi is enabled for the first block of data. Hence, NPE firmware may corrupt the Rx frames because the mbuf data cluster size is insufficient.

Resolution: A single mbuf or the chained mbufs submitted to the RxFree queue through the ixEthAccPortRxFreeReplenish API must have free data cluster of at least 128 bytes each. It is recommended to use 2k buffer size to avoid buffer chaining and for better performance, but this may be a waste of memory for transmitting small size frames.

Ethernet MAC Tx Lockup Issue

Reference #:00018668

Product: Intel® IXP400 Software

Version(s): 1.4, 1.5

Description: Ethernet MAC engine would perform infinite retries on the wire on a dropped packet. The dropped packet could be due to late collision or due to number of retries has exceeded in the early collision. This may cause the Ethernet MAC engine to become stuck due to its behaviour of infinitely retrying and resulting the Tx FIFO overflowing.

Implication: When this happens, MAC will continuously assert TX_EN signal to PHY. This will trigger PHY to enter **jabber** mode and de-asert TX_EN. But the Ethernet MAC will still be in the lockup state resulting in Ethernet transmit services no longer running. This lockup condition is not recoverable unless the MAC is being reset.

Resolution: None. Please obtain Intel® IXP400 Software releases v2.3 for the workaround.

Note: In general, excessive colisions beyond the Ethernet retry limit is indicative of improper network design and configuration. Customers should take precaution to alleviate the occurances of such excessive network congestion.

ixEthAccPortMacReset Function does not Restore MAC Unicast Address

Reference #:00018943

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The ixEthAccPortUnicastMacAddressSetPriv() does not keep a record of the last configured MAC unicast addresses.

Implication: The MAC unicast address ixEthAccPortMacReset function does not restore MAC unicast addresses that were configured for the MAC before it is reset. Due to this, the MAC will filter out ALL frames when non-promiscuous mode is set.

Resolution: Search for ixEthAccPortUnicastMacAddressSetPriv() in ixp400_xscale_sw/src/ethAcc/IxEthAccMac.c (near line 1314),and insert the following code:

```
  for(i=0;i<IX_IEEE803_MAC_ADDRESS_SIZE;i++)

    {

REG_WRITE(ixEthAccMacBase[portId],

                IX_ETH_ACC_MAC_UNI_ADDR_1 + i*sizeof(UINT32),

                                            macAddr->macAddress[i]);

//-------------------------New code inserted start---------------------------//

        /* Keep a copy ucast address set */

        ixEthAccUcastMacAddr[portId].macAddress[i] = macAddr->macAddress[i];

//-------------------------New code inserted end----------------------------//

    }
```

Functions to Change Age Type (Static <-> Dynamic) not Working When the Same MAC Entry has been Registered with the EthDB

Reference #:00152702

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The matching function within the ixEthDBFilteringStaticEntryProvision(port0, mac0) and ixEthDBFilteringDynamicEntryProvision(port0, mac0) compares the new entry with the entries in Ethernet Learning Database based on the port and MAC address only. In other words, if the port and MAC address pair has been registered (entry found in the database), it will not perform any aging type changes.

Implication: Calling ixEthDBFilteringStaticEntryProvision() will not change the aging type to static if the MAC entry has been registered as dynamic aging type previously. Likewise , this applies to ixEthDBFilteringDynamicEntryProvision function too

Resolution: For software release **BEFORE** Intel® IXP400 Software v2.3, if there is a change on a registered MAC's age type, customer should remove the added MAC entry and follow by readding the same MAC entry on the same port with the intended age type manually. Customer can use ixEthDBFilteringDatabaseShow(portID) to check if the MAC entry is already been registered. This manually removal and adding is NOT needed and is fixed in Intel® IXP400 Software v2.3 and later version.

Incorrect VLAN Port Membership/Transmit Tagging Table Update

Reference #:00152731

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The ixEthDBVlanTableEntryUpdate function has doubled the offset value resulting in the value being updated in the wrong position of the VLAN Port Membership/Transmit Tagging table.

Implication: This failure happens when ixEthDBPortVlanMembershipAdd, ixEthDBPortVlan-MembershipRangeAdd, ixEthDBPortVlanMembershipRangeRemove, ixEthDB-PortVlanTagSet,ixEthDBEgressVlanEntryTaggingEnabledSet, ixEthDBEgressVlanRangeTaggingEnabledSet is used to update a single VLAN ID (any VLAN ID from 8-4094) or a group of VLAN IDs (within a range of a byte, for example VLAN IDs 8-15, 16-23 etc) on VLAN Port Membership/ Transmit Tagging table. The VLAN IDs in the range from 0-7 are not affected by this defect. This results in improper functioning of the VLAN membership filterting and VLAN tagging/tag removal control.

Resolution: Code changes are needed in ixp400_sw_xscale/src/ethDB/IxEthDBVlan.c, search for ixEthDBVlanTableEntryUpdate( ) API near line 75

from

```
FILL_SETPORTVLANTABLEENTRY_MSG(message,IX_ETHNPE_PHYSICAL_ID_TO_LOGICAL_ID(portID
),

        2 * setOffset,

        portInfo->vlanMembership[setOffset],

        portInfo->transmitTaggingInfo[setOffset]);
```

to

```
FILL_SETPORTVLANTABLEENTRY_MSG(message,IX_ETHNPE_PHYSICAL_ID_TO_LOGICAL_ID(portID
),

        setOffset,

        portInfo->vlanMembership[setOffset],

        portInfo->transmitTaggingInfo[setOffset]);
```

Note: This has been resolved in IXP400 software v2.3.

Under Heavy Traffic, EthDB-NPE Acknowledge-Message Timeout Triggers Fatal Warning

Reference #:00152701

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The waiting time in EthDB to receive an acknowledge-message from NPE is 20 ms. It is observed that under very heavy small frame traffic, NPE is too busy and unable to send acknowledge-message to EthDB within 20 ms.The EthDB will post a fatal warning.

Implication: If the EthDB maintenance loop detects a time out condition, the algorithm will assume EthDB learning port has become dead and it will disable the EthDB port. This will cause the learning/filtering and aging facility in EthDB to be malfunction.

Resolution: Change the delay time that is defined in ixp400_sw_xscale/src/ethDB/IxEthDB_p.h. Search for #define IX_ETH_DB_NPE_TIMEOUT. Replace this

```
#define IX_ETH_DB_NPE_TIMEOUT (20) /* NPE response timeout, in ms */
```

with this

```
#define IX_ETH_DB_NPE_TIMEOUT (300) /* NPE response timeout, in ms */
```

Memory Leakage in EthACC

Reference #:00019406

Product: Intel® IXP400 Software

Version(s): 1.5

Description: This defect is found in ixEthRxFrameProcess() in EthAcc component.The access layer frame filtering code algorithm in ixEthRxFrameProcess() does not restore the buffer size according to the original size set by upper layer like Ethernet drivers and instead; it sets the buffer size to 64 bytes by default. On the other hand, the ethernet driver does not initialize IX_OSAL_MBUF_ALLOCATED_BUFF_LEN() accordingly.

Implication: A continuous stream of frames with new source address will get filtered due to their destination mac address being the same as the previously learned source address and this will eventually cause all data buffers to be depleted.

Resolution: For Intel® IXP400 Software v1.5, code changes are required in ixEthAccFrameProcess ( ) API

```
   if (((flags & IX_ETHACC_NE_FILTERMASK) != 0)
        && (ixEthAccMacState[portId].portDisableState == ACTIVE))
//--------------------------New code inserted start----------------------------//
    {
         /*
         * Frame that enters here is internally replenished to RxQ.
         * For buffer received from RxQ, IX_OSAL_MBUF_MLEN(mbufPtr) is the
         * number of valid frame byte in the data cluster. So, before we
         * we replenish this buffer to RxFreeQ, we need to config the field
         * to indicate the number of byte in data cluster that is available
         * for buffering. The update of IX_OSAL_MBUF_MLEN(mbufPtr) field should
         * be catered for chained buffers case.
         */
        IX_OSAL_MBUF *restoreMbufPtr = mbufPtr;
         while(restoreMbufPtr != NULL)
        {
#ifndef NDEBUG
      IX_ETH_ACC_WARNING_LOG("ixEthAccRxFrameProcess: Replenish buffer size=%d on port %d\n",
IX_OSAL_MBUF_ALLOCATED_BUFF_LEN(restoreMbufPtr), portId, 0, 0, 0, 0);
           if (IX_OSAL_MBUF_ALLOCATED_BUFF_LEN(restoreMbufPtr) <
              IX_ETHNPE_ACC_RXFREE_BUFFER_LENGTH_MIN)
 {
            IX_ETH_ACC_FATAL_LOG("ixEthAccRxFrameProcess: Allocated buffer is smaller than %d
on port %d\n", IX_ETHNPE_ACC_RXFREE_BUFFER_LENGTH_MIN, portId, 0, 0, 0, 0);
          }
 #endif
          /* restore IX_OSAL_MBUF_MLEN() with
           * IX_OSAL_MBUF_NEXT_BUFFER_IN_PKT_PTR()
           */
          IX_OSAL_MBUF_MLEN(restoreMbufPtr) = IX_OSAL_MBUF_PKT_LEN(restoreMbufPtr) =
            IX_OSAL_MBUF_ALLOCATED_BUFF_LEN(restoreMbufPtr);


      /* traverse down the chained buffer */
          restoreMbufPtr = IX_OSAL_MBUF_NEXT_BUFFER_IN_PKT_PTR(restoreMbufPtr);
        }
//---------------------------------New code inserted End----------------------//
   /* replenish from here */
        if (ixEthAccPortRxFreeReplenish(portId, mbufPtr) != IX_ETH_ACC_SUCCESS)
        {
          IX_ETH_ACC_FATAL_LOG("ixEthRxFrameProcess: Failed to replenish with filtered frame\
                                    on port %d\n", portId, 0, 0, 0, 0, 0);
        }
```

Code changes are also required in ixEthAccMbufRxQPrepare( ) API. Two parts of code changes are needed as shown below:

```
/*buffer length*/

        len = (IX_OSAL_MBUF_MLEN(mbuf) << IX_ETHNPE_ACC_LENGTH_OFFSET);

        IX_ETHACC_NE_LEN(mbuf) = IX_OSAL_SWAP_BE_SHARED_LONG(len);


/* unchained mbufs : next pointer is null */

 IX_ETHACC_NE_NEXT(mbuf) = 0;

//------------------------New Code inserted Start----------------------------//
        /* Store the allocated buffer size. This is useful for restoring

         * the buffer size internally in ixEthAccRxFrameProcess()

         */

        IX_OSAL_MBUF_ALLOCATED_BUFF_LEN(mbuf) = IX_OSAL_MBUF_MLEN(mbuf);

//--------------------------New Code inserted End-----------------------------//

/* flush shared header after all address conversions */

 IX_ETHACC_NE_CACHE_FLUSH(mbuf);
```

and

```
 /* buffer length */

            len = (IX_OSAL_MBUF_MLEN(ptr) << IX_ETHNPE_ACC_LENGTH_OFFSET);

            IX_ETHACC_NE_LEN(ptr) = IX_OSAL_SWAP_BE_SHARED_LONG(len);

//--------------------------New Code inserted Start-------------------------//

            /*

             * Store the allocated buffer size. This is useful for restoring

             * the buffer size internally in ixEthAccRxFrameProcess()

             */

            IX_OSAL_MBUF_ALLOCATED_BUFF_LEN(ptr) = IX_OSAL_MBUF_MLEN(ptr);

 //------------------------New Code inserted End-----------------------------//

            /* flush shared header after all address conversions */

            IX_ETHACC_NE_CACHE_FLUSH(ptr);
```

You must also ensure that the IX_OSAL_MBUF_ALLOCATED_BUFF_LEN() is configured accordingly in the Ethernet Driver (Intel Ethernet Device Driver-EDD and Enhanced Network Driver-END).

B-1 stepping for Intel® IXP42X Product Line of Network Processors Requires Minor Change to IxFeatureCtrl.c to Accommodate Updated Product Revision ID

Reference #:00153342

Product: Intel® IXP400 Software

Version(s): 1.3, 1.4, 1.5

Description: The product revision field in the ID register of the System Control Coprocessor (CP15) has been updated to reflect the B-1 stepping of the IXP42X product line of network processors. Due to this, certain portions of IXP400 Software that still perform condition checks on the product revision field will not recognize the new ID value.

Implication: The function that reads the ID register *ixFeatureCtrlProductIdRead()* is located in *src\featureCtrl\IxFeatureCtrl.c*. Examples of IXP400 software components that use this function to read the ID register are:

- AtmdAcc
- CryptoAcc
- DmaAcc
- EthAcc
- EthDB
- HssAcc
- NpeDl

In customer software, problems may occur during initialization of IXP42X components due to the ID register check not being able to recognize B-1 stepping.

Resolution: The following source files should be modified:

`ixp425_xscale_sw\src\featureCtrl\IxFeatureCtrl.c` (IXP400 Software v1.3)

`ixp400_xscale_sw\src\featureCtrl\IxFeatureCtrl.c` (IXP400 Software v1.4, v1.5)

Open the source file mentioned above for editing. Search for the function called *ixFeatureCtrlProductIdRead()*. Look for the line below within the function.

```
return (pdId);
```

Modify the return statement above to the following:

```
return ((pdId&0xFFFFFFF0) + ((pdId&IX_FEATURE_CTRL_SILICON_STEPPING_MASK)?1:0));
```

The above modification to *ixFeatureCtrlProductIdRead()* will change the definition of the function to recognize whether the IXP42X silicon is an A-0 stepping or a subsequent B-stepping.

Note: Additional changes to customer software may be required if the customer is using a method other than *ixFeatureCtrlProductIdRead()* to read the ID register. For any questions, please contact your local Intel technical representative.

Ethernet Collisions in Half Duplex Mode Due to Same Seed Number used in Backoff Algorithm

Reference #:00153398

Product: Intel® IXP400 Software

Version(s): 1.3, 1.4, 1.5

Description: Ethernet MAC engine in IXP4XX product line processors is incapable of ensuring true randomness in its pseudorandom generator. When two devices use IXP4XX product line processors, they have a high probability of having the same “random” wait-time sequence.

Implication: This leads to multiple frame collisions and causes high transmission delay. Such delay may cause session protocol to restart the link layer when acknowledgement frame is not received within a short period of time. Multiple link restart is disruptive to the real-time application.

Resolution: The software workaround on this will require enhancing the randomness of the backoff algorithm by introducing random values to time slot. Such random value must be more than 512-bit time.

Changes are needed in file ixp400_xscale/src/ethAcc/IxEthAccMac.c; Search for ixEthAccPortDuplexModeSetPriv() API, see the following:

```
*Set the duplex mode*/

IxEthAccStatus

ixEthAccPortDuplexModeSetPriv (IxEthAccPortId portId,

               IxEthAccDuplexMode mode)

{

    UINT32 txregval;

    UINT32 rxregval;

     </-----------------------NEW Code ----------------------->

    UINT32 randTimeSlotOffset;

    <-----------------------NEW Code ----------------------\>

    /*This is bit 1 of the transmit control reg, set to 1 for half duplex, 0 for
full duplex*/

    IX_ETH_ACC_VALIDATE_PORT_ID(portId);

    ------------------

    -------------------

   if (mode ==  IX_ETH_ACC_FULL_DUPLEX)

    {

    </--------------------NEW Code -------------------------->

        /* In full-duplex mode, we set minimum time-slot

         * for the back-off algorithm.

         */

        REG_WRITE(ixEthAccMacBase[portId],

        IX_ETH_ACC_MAC_SLOT_TIME,

        IX_ETH_ACC_MAC_SLOT_TIME_DEFAULT);

    <---------------------NEW Code ------------------------\>

/*Clear half duplex bit in TX*/

REG_WRITE(ixEthAccMacBase[portId],

                             IX_ETH_ACC_MAC_TX_CNTRL1,

                                                    txregval &
~IX_ETH_ACC_TX_CNTRL1_DUPLEX);

----------------

------------------

  ixEthAccMacState[portId].fullDuplex = TRUE;

    }
```

```
        else if (mode ==  IX_ETH_ACC_HALF_DUPLEX)
    {
    </------------------------NEW Code ---------------------->
        /* Back-off algorithm in half-duplex mode has to be random.
         * To enhance such randomness, we use time stamp and
         * LSB of MAC addr as the random offset of the time slot.
         */
       randTimeSlotOffset = ((ixEthAccUcastMacAddr[portId].macAddress[5] | 0x01)
*

ixOsalTimestampGet()) & 0x7F;
        REG_WRITE(ixEthAccMacBase[portId],
          IX_ETH_ACC_MAC_SLOT_TIME,
          IX_ETH_ACC_MAC_SLOT_TIME_DEFAULT + randTimeSlotOffset);
    <------------------------NEW Code ----------------------\>
/*Set half duplex bit in TX*/
REG_WRITE(ixEthAccMacBase[portId],
                                IX_ETH_ACC_MAC_TX_CNTRL1,
                                                        txregval |
IX_ETH_ACC_TX_CNTRL1_DUPLEX);
```

Incorrect Statistic Counter Value Shown in ixEthAccDataPlaneShow()

Reference #: 00153764

Product: Intel® IXP400 Software

Version(s): 1.3, 1.4, 1.5

Description: The ixEthAccDataPlaneShow () has not taken account of “Rx Filtered” value for “Rx Buffer currently for reception” statistic calculation.

Implication: The ixEthAccDataPlaneShow() is unable to display correct value for “Rx Buffers currently for reception” when “Rx Filtered” counter value is bigger than zero.

Resolution: Search for ixEthAccDataPlaneShow(), near line 2711, replace the following:

```
 numBuffersInRx = rx[portId].rxFreeRepOK +
    rx[portId].rxFreeRepDelayed -
    rx[portId].rxFrameClientCallback -
    rx[portId].rxSwQDuringDisable -
    rx[portId].rxDuringDisable
```

with:

```
numBuffersInRx = rx[portId].rxFreeRepOK +
    rx[portId].rxFreeRepDelayed -
    rx[portId].rxFrameClientCallback -
    rx[portId].rxSwQDuringDisable -
    rx[portId].rxDuringDisable -
    rx[portId].rxFiltered;
```

Excessive Interrupts to Intel XScale Processor Detected When Sticky Bit Interrupt Enabled

Reference #:00153855

Product: Intel® IXP400 Software

Version(s): 1.3, 1.4, 1.5

Description: When sticky bit interrupt was enabled, ixEthTxFrameQMCallback was continuously being invoked after transmitting packets.

Implication: There is no functionality error with this observation.

Resolution: To fix this issue, modification is required in ixQMgrDispatcherLoopRunB0() API in ixp400_xscale_sw/src/qmgr/ixQMgrDispatcher.c near line 565 so that QDispatcher masks off interrupt register bit with interrupt enable register and only service queue has both interrupt register and interrupt enable bit set. See the following.

```
ixQMgrHwQIfQInterruptRegRead (group, &intRegVal);

-------------------------New Code Inserted-----------------

    ixQMgrHwQIfQInterruptEnableRegRead (group, &intEnableRegVal);

-------------------------New Code Inserted-------------------

    intRegValSav = intRegVal; // change to intRegVal &= intEnableRegVal;

    if(!stickyEnabled)

    {

         /* not sticky, write back now */

         ixQMgrHwQIfQInterruptRegWrite (group, intRegVal);

    }

---------------------New Code Inserted---------------------------

    else

    {

        intRegValCopy = intRegVal;

    }

         ......................

    if(stickyEnabled)

    {

        /* Write back saved register to clear the interrupt */

    ixQMgrHwQIfQInterruptRegWrite (group, intRegValSav); // Change to
                          //ixQMgrHwQIfQInterruptRegWrite (group, intRegValCopy);

    }
```

LLP Dispatcher May Erroneously Enable Disabled Sporadic Queue Notification

Reference #:00165437

Product: Intel® IXP400 Software

Version(s): 1.5

Description: The Live Lock Prevention (LLP) Mechanism re-enables queue interrupt notification for all sporadic queues at periodicDone unconditionally.

Implication: This would erroneously enable the queue interrupt notification for those sporadic queues which are originally disabled (and also intended to be disabled).

Resolution: It is advisable to set only Eth Rx queues to sporadic queues.

Documentation Issues

This section gives the details of all documentation changes summarized in "Documentation Issues" on page 12.

ixEthAccPortRxFreeReplenish Needs Update

Reference #:1434 (22)

Issue: If ixEthAccPortRxFreeReplenish is called with an invalid port ID, IX_ETH_ACC_SUCCESS is returned instead of IX_ETH_ACC_INVALID_PORT. The documentation for IX_ETH_ACC_INVALID_PORT needs to have information added, on how to enable this checking.

Section B.3.12.10.25, ixEthAccPortRxFreeReplenish(..), on Page 359 of the *Intel® IXP400 Software Programmer's Guide* (252539-002), is changed as follows:

Returns

- IxEthAccStatus
- IX_ETH_ACC_SUCCESS
- IX_ETH_ACC_FAIL — Buffer has was not able to queue the buffer in the receive service.
- IX_ETH_ACC_INVALID_PORT — Port ID is invalid.
- IX_ETH_ACC_PORT_UNINITIALIZED — Port ID is un-initialized.

Affected Docs: *Intel® IXP400 Software Programmer's Guide* (252539-002)

Inconsistent NPE Terminologies Between Code and Documentation

Reference #:2188

Issue: In the documents listed below, each NPE is referred to by two different names. The following table provides clarification.

NPE Name 1	NPE Name 2	NPE Primary Function
NPE-A	WAN/Voice NPE	High Speed Serial and ATM
NPE-B	Ethernet NPE-A	Ethernet (MII 0)
NPE-C	Ethernet NPE-B	Ethernet (MII 1)

Affected Docs: *Intel® IXP400 Software Programmer's Guide* (252539-002), *Intel® IXP42X Product Line of Network Processors and IXC1100 Control Plane Processor Developer's Manual* (252480-002), *Intel® IXP42X Product Line of Network Processors and IXC1100 Control Plane Processor Datasheet* (252479-002), *Intel® IXP4XX Product Line of Network Processors Specification Update* (252702-001), Intel® IXP400 Software *Release 1.4 Software Release Notes*, *Intel® IXP425 Network Processor Mini-DSLAM Software Release Programmer's Guide* (252741-001), *Intel® IXP400 DSLAM Software Release Notes*, *Intel® IXP400 DSP Software Version 2.3 Programmer's Guide* (252725-001), *Intel® IXP400 DSP Software Version 2.3 API Reference Manual* (273811-002), and *Intel® IXP400 Digital Signal Processing (DSP) Software Specification Update* (273810-002).

Buffers Passed to Access Layer Components Must Have Bits 31-29 Set to 0

Reference #: 2265

Issue: Buffers passed to Access Layer Components must have the upper-most three bits set to 0. These bits are used by the access layer to pass information from the NPE back to the Intel XScale® processor.

Affected Docs: *Intel® IXP400 Software Programmer's Guide* (252539-002 and -003)

The CDVT in the ATM Scheduler Does not Comply with the ATM-TM-4.1 Standard

Reference #: 4086

Issue: Although Table 10 on page 70 of the *Intel® IXP400 Software Programmer's Guide* specifies CDVT as being supported, it should be noted that it only does a basic check to ensure that the table size does not exceed the CDVT.

Affected Docs: *Intel® IXP400 Software Programmer's Guide* (252539-002 and -003)

Real-time Variable Bit Rate ATM Traffic (rt-VBR) is not Supported

Reference #: 00056779

Issue: Table 10 on page 70 of the *Intel® IXP400 Software Programmer's Guide* incorrectly specifies
rt-VBR as being supported. rt-VBR is not supported in versions 1.0 through 2.0 of the IXP400 software.

Affected Docs: *Intel® IXP400 Software Programmer's Guide* (2525329-002 through 005)

§ §