Application Note

AN2569
Rev. 0.1, 01/2004

Example Software for the PowerQUICC II™: IMA Initialization Using Internal or External TC Layer Implementation

Paul Wilson
& Michael Johnston
NCSD Applications
East Kilbride,
Scotland

In today's telecommunications industry, asynchronous transfer mode (ATM) is widely used as the high speed transport mechanism for data, voice and multimedia. With the use of ATM adaptation layers (AALs), it is possible to provide a variety of services using the same 53-Byte ATM cell format with guaranteed Quality of Service (QoS).

To help increase accessibility to ATM services, in 1997 the ATM Forum approved the *Inverse Multiplexing for ATM (IMA) Specification*, which defines a new physical-layer protocol to bridge the bandwidth gap between T1/EI and T3/E3 line rates for access and trunking applications. In most cases today, the cost of multiple T1/E1s is more economical than a full T3/E3 when fewer than eight T1/E1s are needed.

The MPC8264, the MPC8266, and the MPC8280, all members of Motorola's PowerQUICC II™ family of integrated communications processors, implement both ATM level control and IMA functions using a combination of hardware and microcode within the ATM controller of their communication processor module (CPM). PowerQUICC II family members contain PowerPC™ processor cores.

This application note presents example C code to initialize the CPM's resources and the IMA microcode in a simple IMA configuration (one group, four links) using internal transmission convergence (TC) layer hardware or an external TC layer device. This example does not support the full IMA reference model; that is, layer management and plane management are not handled by this example.

The following topics are addressed:

1 IMA Overview

IMA enables demultiplexing/deconstruction (transmit) of an ATM cell stream into multiple links. When receiving, the IMA microcode multiplexes/reconstructs incoming cells from multiple links into the original ATM cell stream. IMA must compensate for differences in clock rate and delay over the multiple links.

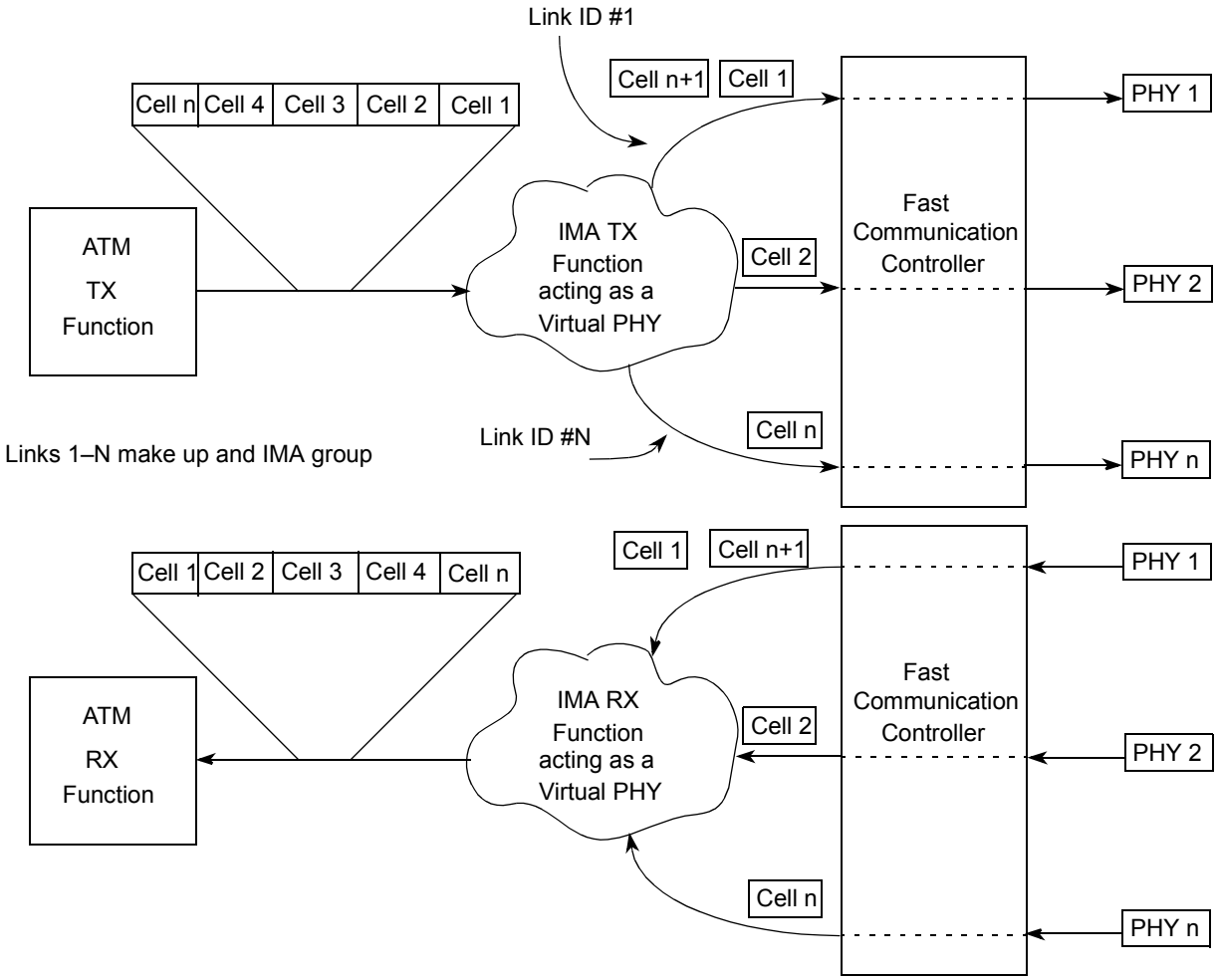

Figure 1. IMA Conceptual Diagram

The data multiplexing performed by IMA is cell based, where cells are distributed among the links in the IMA group in a round-robin cycle. In order to compensate for different clock rates, IMA must periodically insert 'stuff' cells into faster links in order to maintain a consistent average data rate over the links of the group. Furthermore, IMA must compensate for potential differences in delay between the links of the group. According to the IMA specification, the allowable delay differential for DS1/E1 links is 25 ms, which at E1 rates is equivalent to approximately 118 cells. IMA on the PowerQUICC II allows the user to define the allowable delay differential by means of a delay compensation buffer of programmable length.

IMA accomplishes these goals by the periodic insertion of special operations and maintenance (OAM) cells, which (among other things) define M-cell frame boundaries and provide frame sequence numbers and stuffing information. This framing information is used by the receiver to correlate the received cell streams and extract cells in order from the links of the IMA group, thereby reconstructing the original cell stream.

1.1 IMA Frame

An IMA frame consists of M number of cells (M = 32, 64, 128, or 256). Each frame consists of payload and control cells. There is at least one control cell, called an ICP (IMA control protocol) cell, in each frame. An additional ICP cell may be included in a frame to compensate for timing differences between the links in an IMA group (if, for example, one link is slightly faster than the other). The insertion of additional ICP cells to compensate for timing differences between links is called a stuff event. The transmitter is responsible for inserting stuff ICP (SICP) cells, and the receiver monitors for stuff indication and discards SICP cells. The location of the ICP cell in an IMA frame is determined during the IMA start-up sequence.

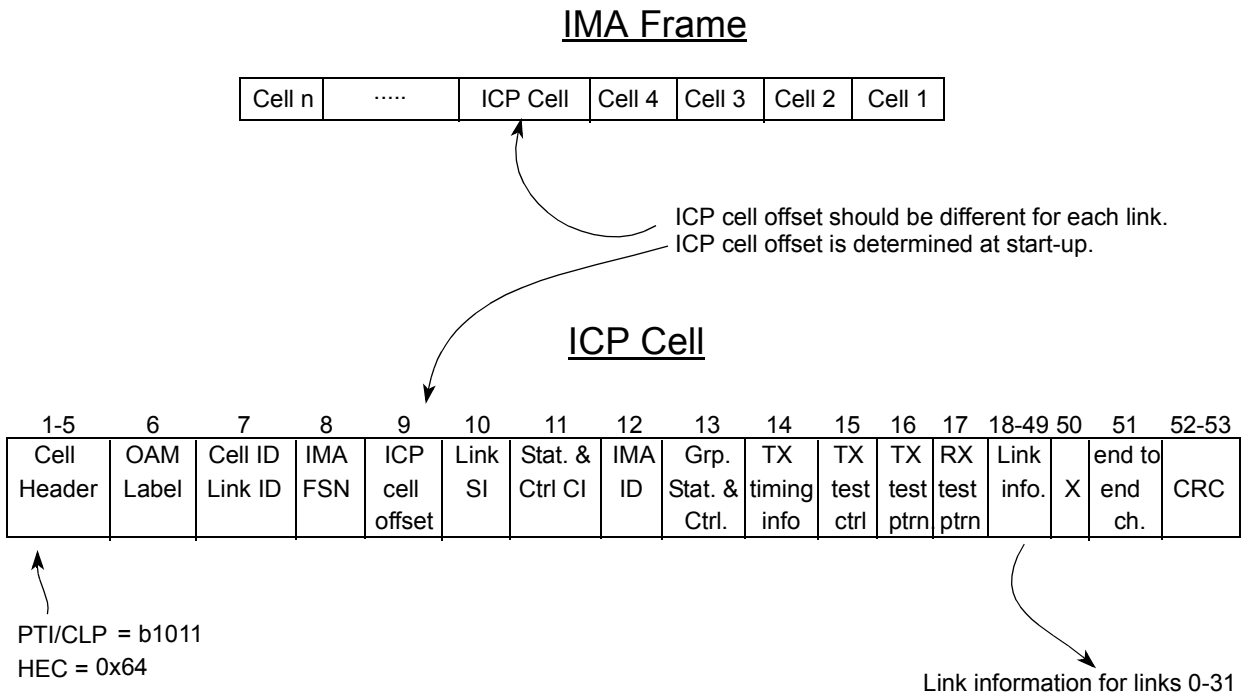

Figure 2. IMA Frame and ICP Cell Formats

1.2 References

For a better understanding of IMA concepts and terminology and the PowerQUICC II programming model, users are encouraged to familiarize themselves with the following references:

- ATM forum IMA specifications, available from http://www.atmforum.com:
  - *Inverse Multiplexing for ATM (IMA) Specification, Version 1.0* (AF-PHY-0086.000)
  - *Inverse Multiplexing for ATM (IMA) Specification, Version 1.1* (AF-PHY-0086.001)
- Freescale MPC8260 product documentation, available from http://www.mot.com/semiconductors:
  - *MPC8260 PowerQUICC II Family Reference Manual* (MPC8260UM)
  - *MPC8280 PowerQUICC II Specification (Addendum to the MPC8260 PowerQUICC II Family Reference Manual)* (MPC8280UMAD)
  - *MPC82xx Family Application Development System Users Manual* (PQ2FADSZURM)
  - *PowerQUICC II API (Drivers and Examples)* (MPC8260API)
  - IMA API Software Reference System Version 1.3

2 Hardware Description

This example C driver has been developed and tested using different hardware platforms which depend on the TC layer implementation:

- External TC layer—PQ2FADS-ZU pilot board as a system platform in connection with the multi-PHY slave board
- Internal TC layer—PQ2FADS-ZU pilot board as a system platform in connection with a TCOM or ECOM board

2.1 Hardware Configuration Using External TC Layer

Figure 3 below shows the hardware configuration using the multi-PHY slave board, which implements the TC layer using external hardware.

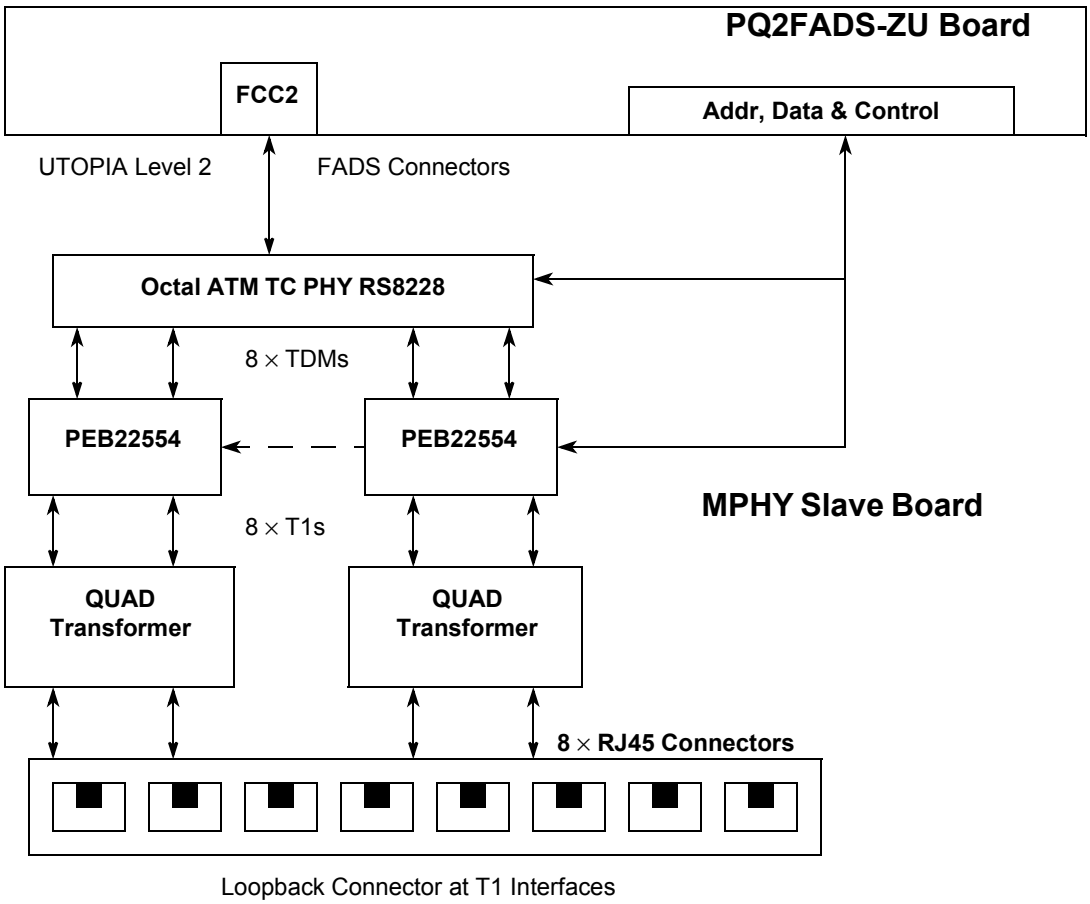

Figure 3. IMA Hardware Configuration Using External TC layer

The multi-PHY slave board allows ATM cells to be received and transmitted over 8 T1 lines using the UTOPIA level-2 8-bit interface supported on FCC2. This board uses one Mindspeed RS8228 octal ATM TC PHY device, which translates the UTOPIA level-2 bus to 8 T1/E1 lines and vice versa. Each T1/E1 port in the RS8228 has its own separate clock and can be independently configured for operation using the PowerQUICC II bus. Framing is controlled through the Infineon QuadFALC PEB22554 T1/E1 framer. The multi-PHY slave board is connected to the PQ2FADS-ZU board using two 128-pin DIN connectors.

2.2 Hardware Configuration Using Internal TC Layer

Figure 4 below shows the hardware configuration using the internal TC layer hardware on the PowerQUICC II with the PQ2FADS-ZU board and a TCOM or ECOM board.

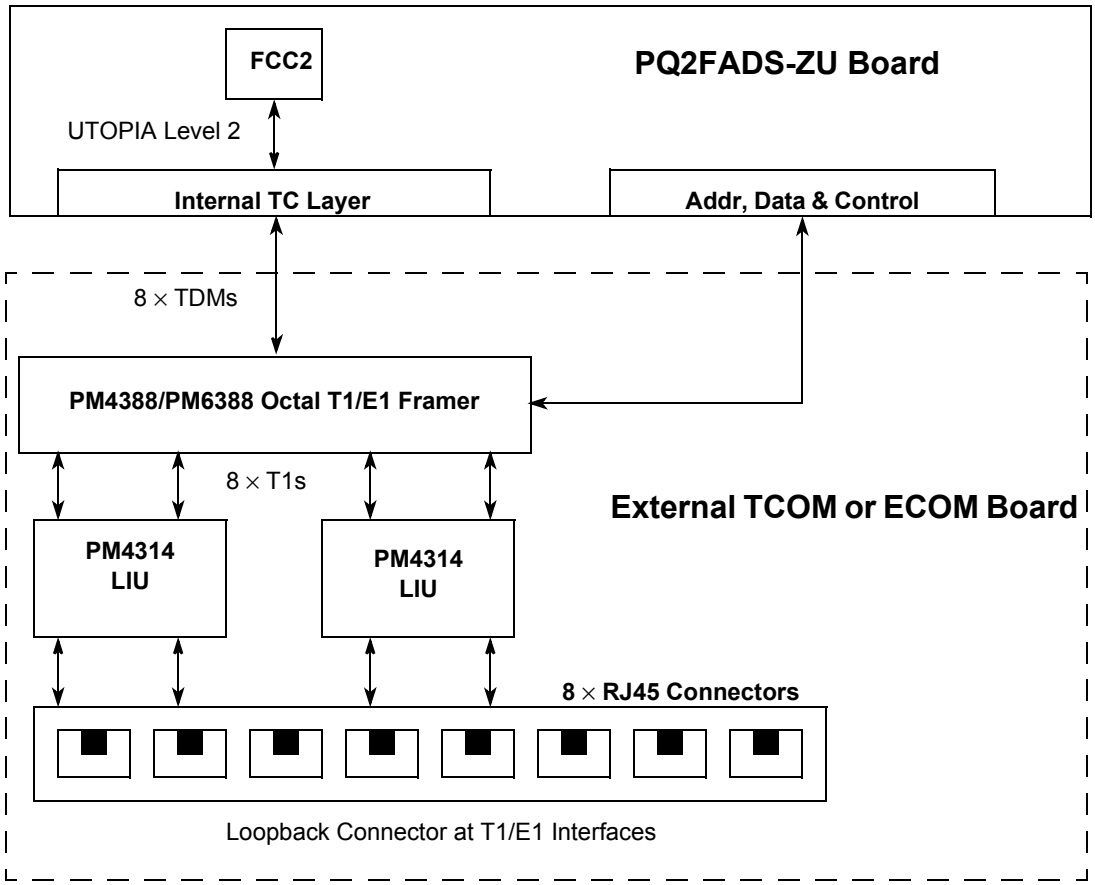

Figure 4. Hardware Configuration for Internal TC layer

The TCOM/ECOM board supports up to 8 T1/E1 interfaces and uses one PMC-Sierra PM4388/PM6388 octal T1/E1 framer and two PMC-Sierra PM4314 line interface units (LIUs). The TCOM/ECOM board is connected to the PQ2FADS-ZU board through two 128-pin DIN connectors. The TC layer is implemented using internal hardware connected to FCC2 on the MPC8264, MPC8266, and MPC8280 members of the PowerQUICC II family .

3 Software Description

The focus of this application note is the implementation of an example IMA software driver for use on the PowerQUICC II. The IMA software driver is responsible for initializing the PowerQUICC II IMA data structure, establishing and tearing down connections, handling of alarms, keeping statistics, and controlling protocol state machines. The IMA microcode interfaces to the software-implemented functions (layer management and plane management) by providing received ICP cells and by interrupts. The software-implemented functions control the microcode and the system through the IMA root, group, and link parameters, and by providing an ICP cell template to the microcode for ICP cell transmission.

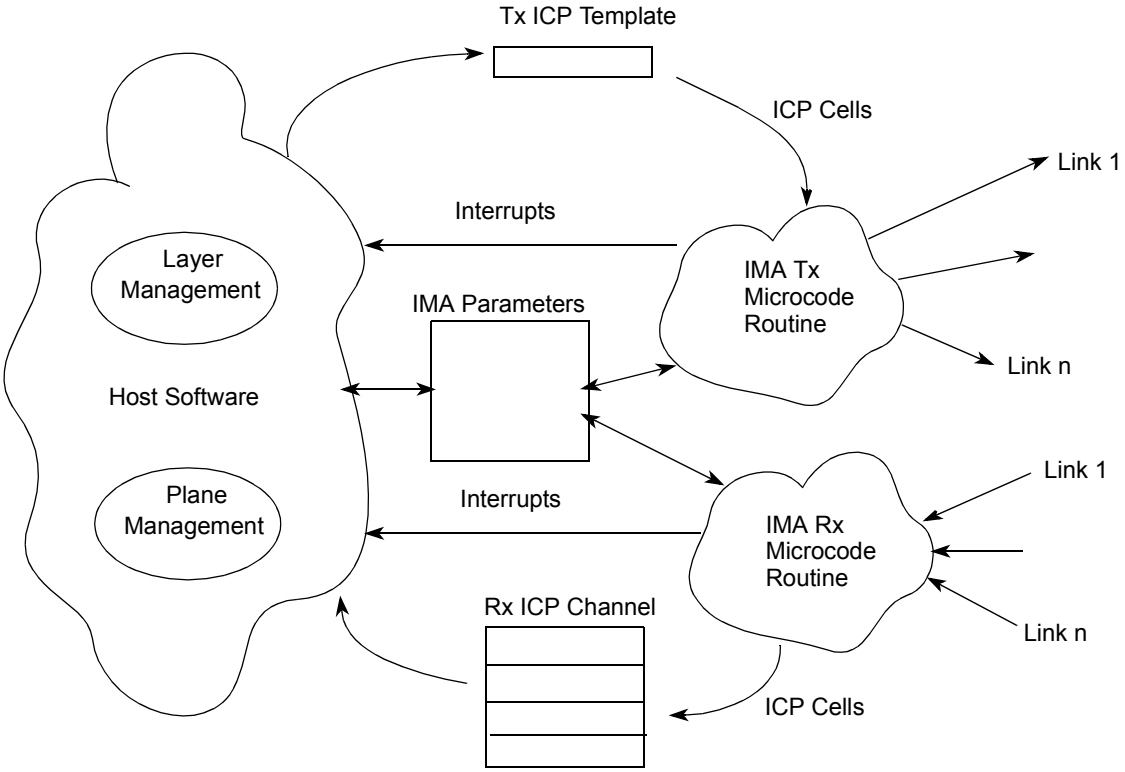

Figure 5. IMA Microcode/Software Interaction

3.1 IMA Driver Files

The PowerQUICC II IMA software driver example was written and compiled specifically to run on the PQ2FADS-ZU board, although it can be readily adapted to run on other PowerQUICC II–based boards. The source files implemented include one assembly startup file, one main C file and four header files, as listed below:

- PQ2_ima.c—Main IMA driver C source file
- fcc_ima.h—Main header file containing IMA-specific data structures
- fcc_atm.h—Local header file containing ATM and IMA constants, definitions, and data structures
- mpc82xx.h—General data structures for the FCC parameter RAM and internal memory map register (IMMR) for the MPC826x registers
- netcomm.h—Header file containing global data type definitions
- startup.s—Initializes the stack, instruction and data caches, memory management unit (MMU) and local bus

- ucode.c—IMA RAM microcode release 1.2 for MPC8264/MPC8266 HiP4 Rev A0 and B1 and MPC8280 HiP7 Rev 0.0 silicon (silicon revision selectable by the user)
- PQ2_ima.lcf—Codewarrior link configuration file
- PQ2_ima.elf—Downloadable ELF file

3.2 IMA Functions and Usage

The PQ2_ima.c driver uses certain functions to initialize and implement IMA on the PowerQUICC II. The functions and their purpose are given in Table 1 below.

Table 1. PQ2_ima.c functions

Function	Purpose
main()	Responsible for calling the respective initialization routines. After all the parameters have been initialized, the scheduling of cells begins by enabling BRG5 and activating the transmitter and receiver.
IMA_Test_Loop	Continuously transmits and receives AAL5 frames under certain conditions. Compares the transmitted frames to the received frames. Buffer mismatch results in the “Bad_Buffer_Cnt” variable being incremented. “Test_count” contains the total number of frames sent. See Section 3.4, “IMA Test Sequence—IMA_Test_Loop().”
Init_IMA_Root	Initializes parameters in the IMA root table including filler cell template, transmit queue parameters, PHY management parameters, and base pointers for group/link tables
Init_IMA_Groups	Initializes IMA group parameters in IMA Rx group 4 and IMA Tx group 3
Init_IMA_Links	Initializes IMA transmit and receive links parameters for link tables 0–3
FCC_init	Initializes FCC ATM parameters including IMA root table pointer. This routine also enables BRG clocks for UTOPIA and FCC for IMA operation.
InitCTs	Initializes two receive connection tables (RCTs) in DPRAM for the raw cell and ICP cell connections. In addition, this routine initializes the transmit and receive connection tables for CH#257 where data is transmitted and received.
Init_RCT	Initializes the receive connection table (RCT) for the channel specified in the parameter passed.
Init_TCT	Initializes the transmit connection table (TCT) for the channel specified in the parameter passed
Init_VPT	Initializes VP table entries to zero. All unused VP table entries point to the first VC table entry, which is initialized to indicate “no match,” resulting in the cell being discarded by the CP.
Get_VPT_addr	Determines the VP table entry address by performing address compression of the PHY/VPI value passed
Update_VPT	Updates the respective VP table entry with the VC mask and VC table offset values. The corresponding VC table entries are also initialized.
Update_VCT	Updates the respective VC table entry with the channel number (RCT table number) to use for a successful match of the VPI/VCI value
InitAPC	Initializes the ATM pace controller parameters in DPRAM
Init_int_queue	Initializes the ATM event interrupt queue in external memory. The contents of the interrupt queue are reset to zero with the exception of the last entry, which contains the “wrap” bit.
Init_TX_BDs	Initializes transmit buffer descriptors (BDs) for the channel specified in the BD parameter

Table 1. PQ2_ima.c functions (continued)

Function	Purpose
Init_RX_BDs	Initializes receive buffer descriptors (BDs) for the channel specified in the BD parameter
Init_IMA_Links	Initialize IMA transmit and receive links parameters for link tables 0–3
VPVC_Init	Hardcoded initialization of the VP/VC tables for IMA RVPHYNUM 5
InitTxBuffers	Initializes the receive buffers to zero to make sure they are in a known state before data is received. (This helps in debugging.) The transmit buffers (frames) can have the following patterns: Buffer 1: 0x00,01,02,03...10,11,12...20,21,22...8F Buffer 2: 0xA5,A5,A5,A5 Buffer 3: 0x0x9A,9A,9A....6A,6A
InterruptInit	Copies interrupt handler code from its current address to the specified PowerPC interrupt vector
Init_Decrementer	Copies the ISR for the decrementer exception from its current address to the specified PowerPC interrupt vector location. The decrementer exception is enabled by default and therefore an ISR must be provided to service this exception. Since this example does not use the decrementer, an **rfi** (Return From Interrupt) instruction is all that is needed.
BDRxError	Returns TRUE if buffer descriptor status bd_cstatus indicates receive error; returns FALSE otherwise. Note: receive errors are as follows: 0x8: Receive frame was aborted. 0x2: Length error 0x1: CRC error
LastBD	Returns TRUE if buffer descriptor with status and control register bd_cstatus is last in frame; return FALSE otherwise
BDEmpty	Returns TRUE if buffer descriptor status bd_cstatus is empty; returns FALSE otherwise.
ExtIntHandler	Processes external interrupt (assumes only interrupts from FCC*x*) Main processing steps: 1. Get a copy of FCCx's event register. 2. Check that it's an “External Vector” and that the correct FCC generated the interrupt. 3. Verify that all frames were transmitted. 4. Verify the contents of the received frames. 5. Clear the ATM interrupt queue. 6. Clear FCCx's event and pending registers. Note: This ISR handles only global interrupt events (found in the FCC event register). The type of global interrupt is further defined in the interrupt queue. Global interrupts handled: received frame & transmit buffer.
SetIMMR	Changes the base address of dual-port RAM NOTE: The base of the internal memory mapped register is defined at power up by the hard reset configuration word (internal space base, ISB). The ISB for the is initialized so that the internal memory-mapped registers start at 0x0470_0000.
Led	Turns on/off either the green or red LED on the PQ2FADS-ZU pilot board
FlashLed	Flashes the red LED on the PQ2FADS-ZU pilot board
Init_Header	This function initializes the HEADER value for AAL0 cells. If the header is contained in the TCT, the header value in the xmit buffer (cell) is cleared; otherwise, it is initialized to the “header” value passed in the function.
LoadUCODE	Enables IMA functionality in ROM or loads IMA RAM microcode release 1.2 for HiP4 Rev A0 or B1, or HiP7 Rev 0.0 silicon and sets the trap registers.

Table 1. PQ2_ima.c functions (continued)

Function	Purpose
Init_idle_base	Initialize IDLE cell template
Init_EXT_TC_PHY	Specific to external TC layer configuration: On the MPHY board this routine initializes ATM TC PHY device (Mindspeed RS8228) for either internal loopback mode or normal operation.
Init_MPHY_T1_framers	Specific to external TC layer configuration: On the MPHY board this routine initializes T1 framers (8 T1s) devices from Infineon PEB22554.
Init_IO_ports_EXTTC	Specific to external TC layer configuration: Configures parallel I/O port pins on the PowerQUICC II for UTOPIA 8-bit MPHY operation on FCC2 in order to interface with the external ATM TC PHY device (Mindspeed RS8228)
Init_8228_UPM	Specific to external TC layer configuration: Configures UPM to use $\overline{CS6}$ for RS8228 (memory map this device to 0x4000_0000).
Init_IO_ports_INTTC	Specific to internal TC layer configuration: Configures parallel I/O port pins on the PowerQUICC II for 8 T1/E1 links in order to interface with the external framer device (PMC-Sierra PM4388/PM6388) on the external TCOM or ECOM board.
Init_Clocks_INTTC	Specific to internal TC layer configuration: Configures parallel I/O port pins on the PowerQUICC II for 8 T1/E1 links in order to interface with the external framer device (PMC-Sierra PM4388/PM6388) on the external TCOM or ECOM board.
TCOM_ECOM_Mem_Init	Specific to internal TC layer configuration: Configure UPM to use $\overline{CS6}$ to memory map the octal T1/E1 framer (PMC-Sierra PM4388/6388) and LIUs (PMC-Sierra PM4314) on the TCOM/ECOM Board to address 0x6000_0000.
TCOM_Phy_Init	Specific to internal TC layer configuration: On the TCOM board this routine initializes the T1 interfaces on the framer device and LIUs.
ECOM_Phy_Init	Specific to internal TC layer configuration: On the ECOM board this routine initializes the E1 interfaces on the framer device and LIUs.
INTTC_Layer_Init	Specific to internal TC layer configuration: This routine initializes the mode register and cell delineation registers for all 8 internal TC layer blocks.
TCOM_SIRAM_Init	Specific to internal TC layer configuration: This routine initializes transmit/receive SIRAM for 8 T1 connections.
ECOM_SIRAM_Init	Specific to internal TC layer configuration: This routine initializes transmit/receive SIRAM for 8 E1 connections.

3.3 Memory Map

The memory map is provided here for the benefit of the reader. However, the location and size of memory allocation are dependent on the specific project requirements. The starting address of dual-port RAM (DPR_BASE) is defined by the internal space base (ISB) value in the internal memory map register (IMMR). The IMMR is configured by the PQ2FADS-ZU pilot board with a value of 0x0470_0000. This address is hard coded in fcc_atm.h. The memory map can be easily changed by modifying the address/offset definitions in fcc_atm.h.

The ATM / IMA data structures with this example C code are located on the local bus.

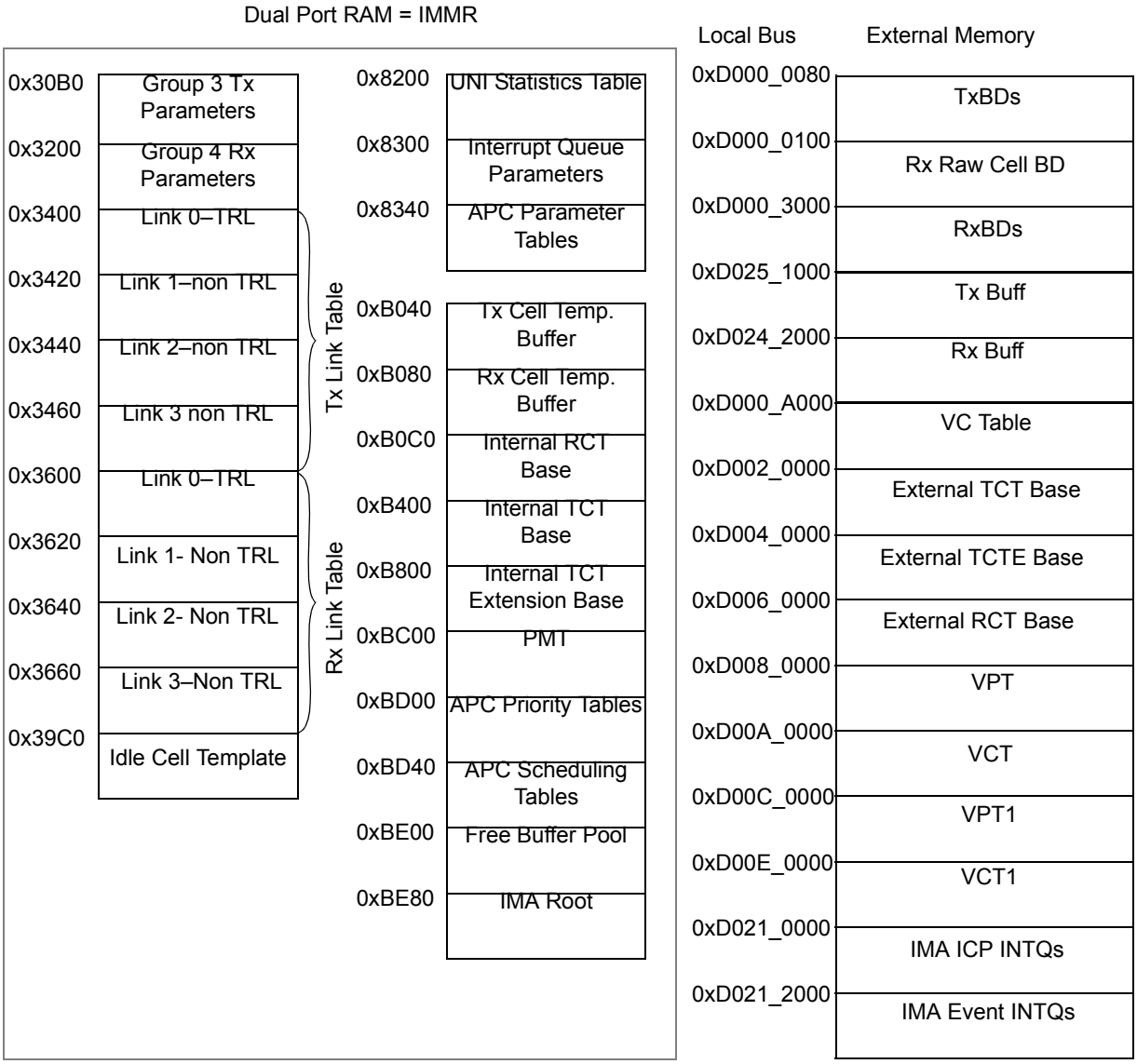

Figure 6. Memory Map for DPRAM and External Memory

3.4 IMA Test Sequence—IMA_Test_Loop()

The following test sequence, implemented by this example C code in the routine IMA_Test_Loop(), is used to verify that:

- IMA link frame synchronization is working
- IMA group synchronization has been achieved
- ATM data transmitted is received okay in external loopback at T1 links

For an IMA configuration which includes one IMA group with four IMA links:

1. Wait for four ICP cells to be received. (Decrement passcount for each ICP cell received.)
2. When all expected ICP cells are received, assign links 0 to 3 to IMA group 4 ILRCNTL[GA]= 1.
3. Wait for the IMA frame synchronization working (IFSW) events on links 0 to 3.

4. When all links are synchronized, enable IMA group synchronization (set IGRSTATE[GDSS] to 0b01.
5. Wait until IMA Group synchronization is achieved. Check for GDS event in the IMA interrupt queue.
6. Reset IMA interrupt queue.
7. Enable Tx BDs and activate ATM ch#257 for data transmission.
8. Check buffer descriptors to ensure data is transmitted and received. Ensure IMA group is still synchronized by setting IGRSTATE[GDSS] to 0b11.
9. If unexpected IMA_events–Re-initialize transmit & receive BDs
10. After BD status updated (ready flag cleared), compare transmit buffers against receive buffers.
    - If receive and transmit buffers match, increment counter Good_Buffer_Cnt.
    - If receive and transmit buffers do not match, increment counter Bad_Buffer_Cnt.
11. If a link event (IFDS, DCBO, LS, TQU, TQO) is detected in the IMA interrupt queue, the CT_Init_Flag is set. Start a new transmission sequence. (Go back to step 7 above.)

3.5 User-definable variables

This table details the user-definable options for the example IMA software driver.

Table 2. User Definable Variables in PQ2_ima.c.

Variables	Description	Current Setting	Alternate Setting
FCC_num	Selection of FCC to be used	FCC2	FCC1 [1]
Frame_length	Size of AAL5 transmit and receive frames (bytes). Default is 96 (2x48).	144 bytes	User Defined
UCODE_type	Enable IMA functionality in ROM or use microcode patch to support latest IMA errata fixes for HiP4 Rev A0 and B1, and HiP7 Rev 0.0 silicon.	IMA_ROM	IMA_HiP4REVA IMA_HiP4REVB IMA_HiP7REV00
Trl_link	Designate which link is the timing reference link	0	1, 2, 3
TC_Layer_type	Enables IMA example C code for two different hardware configurations based on TC layer implementation, internal or external	INTERNAL	EXTERNAL
COM_Board	Select TCOM or ECOM board	ECOM	TCOM
CT_init_flag	Set if a link event (IFDS, DCBO, LS, TQU, TQO) is detected in the IMA interrupt queue	FALSE	TRUE
pass_count	Number of ICP events expected (IMA_Test_Loop)	4	User Defined

[1] Due to the hardware platforms used to verify this IMA initialization code, only FCC2 can be used with this example C code. However this does not prevent the user from modifying this code to use FCC1 using their own hardware configuration which supports an external TC layer device.

4 Development/Test Environment

The following development tools were used:

- Metrowerks Codewarrior 8.0 compiler and debugger with WIRETAP
- PQ2FADS-ZU: MPC82xx Family Application Development System
- MPC8260 TCOM and ECOM boards
- MPC8260 multi-PHY slave board (used for external TC layer implementation only)

This example code transmits three ATM AAL5 frames continuously over four T1 links using the IMA protocol using external loopback connections at the T1/E1 interfaces. On the receive side, if IMA frame synchronization (IFSW event for each link) and group synchronization (GDS event) are achieved, this example processes the transmit ATM AAL5 frames into a receive buffer. If the data received matches the data transmitted then the green general-purpose 0 LED on the PQ2FADS-ZU board is illuminated. If the data transmitted does not match the data received, or if an unexpected error occurs, the red general-purpose 1 LED on the PQ2FADS-ZU is illuminated.

Below are some useful addresses to check to verify that this example code has been executed successfully:

- 0x0470_3600—IMA receive link 0 table1. (Check that ILRSTATE[IFSS] = 1x and ILRSTATE[FSES] = 00.)
- 0x0470_3620—IMA receive link 1 table1. (Check that ILRSTATE[IFSS] = 1x and ILRSTATE[FSES] = 00.)
- 0x0470_3640—IMA receive link 2 table1. (Check that ILRSTATE[IFSS] = 1x and ILRSTATE[FSES] = 00.)
- 0x0470_3660—IMA receive link 3 table1. (Check that ILRSTATE[IFSS] = 1x and ILRSTATE[FSES] = 00.)
- 0x0470_3200—IMA receive group 3 table. (Check that IGRSTATE[GDSS] = 11.)
- 0xD021_2000—IMA interrupt queues
- 0xD000_0080—Transmit buffer descriptors
- 0xD000_3000—Receive buffer descriptors
- 0xD020_5100—Transmit buffers
- 0xD024_2000—Receive buffers

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com