Freescale Semiconductor
Application Note

AN3045
Rev. 3, 5/2006

A Comparison of the MPC5200B (Mask Set M62C) with Prior MPC5200 Versions

by: Charles Melear
Infotainment, Multimedia, and Telematics Division

1 Overview

This document describes the basic differences between the MPC5200B (Mask Set M62C Rev. 1) and prior revisions of the MPC5200 microprocessors. In the context of this document, MPC5200B always refers to the device built with Mask Set M62C Rev. 1. The MPC5200B adds enhancements to the device, including an updated SDRAM memory controller, upgrades to the programmable serial controller, and other functions to improve the operation of the device. Fixes to known errata were also incorporated into the MPC5200B, as well as features for improved testability. The MPC5200B microcontroller is based on an e300 core using the PowerPC™ instruction set.

The MPC5200B pinout is identical to its predecessor, the MPC5200.

While the MPC5200B is pin compatible with prior versions of the MPC5200, there are significant differences in the electrical characteristics of the pins, particularly with respect to the drive strength and slew rate capabilities on the SDRAM Bus.

Table of Contents

1 Overview 1
2 Changes to the Hardware Specification 2
2.1 Electrical and Timing Characteristics. 2
2.2 AC Electrical Characteristics—Memory Interface. 2
2.3 I/O Pins 3
2.4 HRESET 3
2.5 Reset Configuration. 4
3 Changes to the Programmer's Interface. 5
3.1 CDM Clock Enable Register —MBAR + 0x0214. 5
3.2 CDM 48-MHz Fractional Divider Configuration Register—MBAR + 0x0210 . . . 6
3.3 SDRAM Memory Controller 8
3.4 BestComm. 11
3.5 PCI. 18
3.6 PSC 33
3.7 Registers Added To Implement the Enhanced AC97 Mode. 41
3.8 I^2C 50
3.9 LPC/GPIO 55
3.10 MSCAN 56
3.11 ID Codes 58
4 Documentation 58

freescale™ semiconductor

When upgrading an existing MPC5200 design to use the MPC5200B, it is imperative that circuit board layout and signal line impedance matching issues be checked for proper performance. In particular, series termination resistors, as shown in Figure 1, must match the PC board line impedance as closely as possible.

NOTE

Place termination resistors as close to package pins as possible.

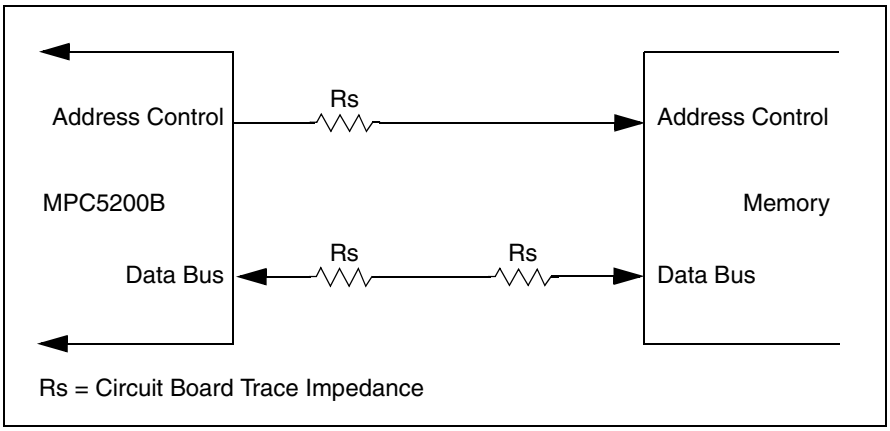

Figure 1. Proper Use of Series Termination Resistors

2 Changes to the Hardware Specification

2.1 Electrical and Timing Characteristics

The target for the electrical and timing characteristics of the MPC5200B is the MPC5200 hardware specifications.

2.2 AC Electrical Characteristics—Memory Interface

The improved DDR interface (DQS) results in a changed timing specification. Refer to the *MPC5200B User's Guide* for details.

2.3 I/O Pins

Additional Schmitt Trigger inputs have been added to these pins for better noise rejection:

- PSC1_3
- PSC2_3
- PSC3_2
- PSC6_3
- JTAG_TCK

The pins associated with the PSC ports can be used as the BITCLOCK for the AC97 interface or the CLOCK for the CODEC interface. The JTAG_TCK pin is the clock input for the JTAG interface. Schmitt Triggers on these inputs improve noise immunity/rejection as well as system performance and reliability. This improvement does not require any system level design changes.

2.4 HRESET

The MPC5200B HRESET pin is asserted when Power On Reset is asserted, regardless of whether the SYS_XTAL clock is running or not.

On prior versions of the MPC5200, a running system clock was required to configure the internal circuitry driving the HRESET pin.

This means that for the period of time between the application of power and the start of the SYS_XTAL clock, the HRESET pin was not driven to a specified value. If by random chance the HRESET pin assumed a logic 1 as its output value, the HRESET pin would rise along with Vdd during power up until the system clock started. As soon as the SYS_XTAL clock started, the HRESET pin would be actively driven to a logic 0 (assuming that POR is currently being driven to a logic 0).

Some applications used the HRESET pin to hold various peripherals, including certain FLASH memory devices, in the reset state. By allowing the HRESET pin to float to a logic 1 during the time that power was being applied, these peripherals would momentarily be taken out of reset. In some cases, this condition was responsible for memory corruption issues.

Actively forcing HRESET on the MPC5200B to a logic 0 while Power On Reset is asserted, even if the SYS_XTAL clock has not started, avoids these reset issues.

2.5 Reset Configuration

During reset ($\overline{\text{HRESET}}$ and $\overline{\text{PORRESET}}$), the reset configuration word is latched in the related reset configuration word register with each rising edge of the SYS_XTAL signal. If both resets ($\overline{\text{HRESET}}$ and $\overline{\text{PORRESET}}$) are inactive (high), the contents of this register get locked.

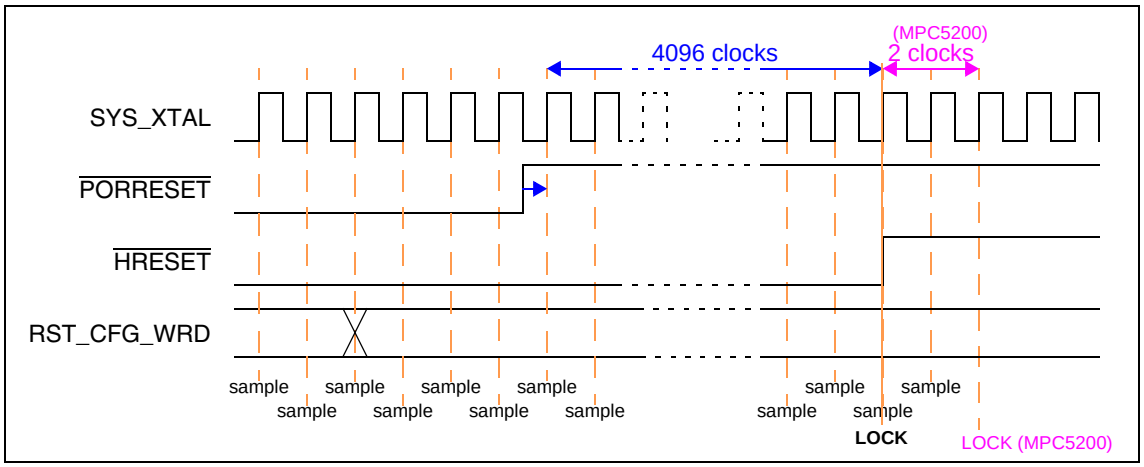

Figure 2. Reset Configuration Word Locking

For the MPC5200B, as well as prior versions of the device, pins that are configured by the assertion of Power On Reset/HRESET will maintain their reset state as long as POR or HRESET is asserted. That is, even though the configuration word is sampled on each rising edge of the SYS_XTAL clock, the actual microcontroller pins maintain their reset state until HRESET is released. Upon the release of HRESET, the various pins assume the state controlled by the configuration word.

In versions of the MPC5200 prior to the MPC5200B, the configuration word was latched and locked into the reset configuration register on the second rising edge of the SYS_XTAL clock after the release of HRESET. This meant that the external circuitry driving the various configuration pins of prior versions of the MPC5200 had to continue to drive the configuration pins for two additional SYS_XTAL clock cycles after HRESET was released to insure that the configuration word was properly latched. Also, any other devices that could possibly drive the configuration pins had to be held in reset for these two SYS_XTAL clock cycles so that potential conflict issues would be avoided.

The MPC5200B latches and locks the configuration word into the reset configuration register on the first rising edge of SYS_XTAL after the release of HRESET. This change makes external configuration word circuitry easier to design as it significantly reduces the hold time for the logic driving the reset configuration word.

3 Changes to the Programmer's Interface

There are some differences in the programmer's interface between the MPC5200B and prior versions of the MPC5200. Some registers were added, and some reserved bits are used to provide extra functionality.

3.1 CDM Clock Enable Register—MBAR + 0x0214

Bit 24 (psc345_clk_en) is separated into 3 bits psc3_clk_en, psc4_clk_en, and psc5_clk_en. The function of bits IR_TX and IR_RX are replaced by psc4_clk_en and psc5_clk_en, respectively.

Only bits 22, 23 and 24 have changed between the MPC5200B and prior versions of the MPC5200.

On prior versions of the MPC5200, bits 22 and 23 of the CDM clock enable register were associated with an IR function. The IR function has been removed from the MPC5200B, along with the functionality of these two bits. Bit 24 of this register on prior versions of the MPC5200 was used to enable the IP bus clock to PSC3, PSC4, and PSC5. On the MPC5200B, bits 22 and 23 are now used to enable the IP bus clock to PSC5 and PSC4 respectively. Now, bit 24 only enables the IP bus clock to PSC3.

In all versions of the MPC5200, the RESET state of all the clock enable bits in the CDM clock enable register is a logic 1. That is, all of the clock enable bits are set to enable. The software for prior versions of the MPC5200 may have to be changed to work on the MPC5200B. Specifically, if the user's current software wrote bits 22 or 23 to a logic 0 and the user wants to use PSC4 or PSC5, these bits will have to be written to a logic 1. If the user's software does not write to this register, then no software changes will be required. That is, PSC4 and PSC5 will be enabled after the release of reset.

3.1.1 CDM Clock Enable Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved Write 0												mem_ clk_en	pci_ clk_en	lpc_ clk_en	slt_ clk_en
W																
RESET:	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	scom_ clk_en	ata_ clk_en	eth_ clk_en	usb_ clk_en	spi_ clk_en	bdlc_clk_ en	psc5_ clk_en	psc4_ clk_en	psc3_ clk_en	psc2_ clk_en	psc1_ clk_en	psc6 clk_en	mscan_ clk_en	i2c_ clk_en	timer_ clk_en	gpio_ clk_en
W																
RESET:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 3. CDM Clock Enable Register for MPC5200B

3.1.2 CDM Clock Enable Register for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved Write 0												mem_ clk_en	pci_ clk_en	lpc_ clk_en	slt_ clk_en
W																
RESET:	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	scom_ clk_en	ata_ clk_en	eth_ clk_en	usb_ clk_en	spi_ clk_en	bdlc_clk_ en	irrx_ clk_en	irtx_ clk_en	psc345_ clk_en	psc2_ clk_en	psc1_ clk_en	psc6 clk_en	mscan_ clk_en	i2c_ clk_en	timer_ clk_en	gpio_ clk_en
W																
RESET:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 4. CDM Clock Enable Register for Prior Versions of the MPC5200

Note: The MPC5200B does not contain the IR module that was present on prior versions of the MPC5200. The clock enable bits referred to above enabled the clocks to this module; however, in the PSC module, there is a CODEC mode that does implement an IR function. This note is for information only.

3.2 CDM 48-MHz Fractional Divider Configuration Register—MBAR + 0x0210

Bit 5 (ext_usb_clk_sync) is added.

The ability to synchronize the external USB clock with the internal buses was added to improve USB performance above 1 Mbit/second. The user should set this bit when using USB at higher speeds.

The default state of bit 5 is a logic 0, which causes the external 48-MHz clock to not be synchronized with the internal clock. If the user's software for prior versions of the MPC5200 wrote this bit to a logic 0, then no changes to the software will be required for the MPC5200B; however, if the user wants to take advantage of this feature, the software will have to be modified. That is, bit 5 will need to be set to a logic 1.

Only bit 5 of this register is affected.

3.2.1 CDM 48-MHz Fractional Divider Configuration Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	Reserved Write 0					ext_usb_ sync_en	ext_usb_ 48mhz_en	ext_irda_ 48mhz_en	Reserved Write 0							fd_en
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R W	Rsrvd Write 0	cfgd_p3_cnt			Rsrvd Write 0	cfgd_p2_cntt			Rsrvd Write 0	cfgd_p1_cnt			Rsrvd Write 0	cfgd_p0_cnt		
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5. CDM 48-MHz Fractional Divider Configuration Register for MPC5200B

3.2.2 CDM 48-MHz Fractional Divider Configuration Register for Prior Versions of the MPC5200

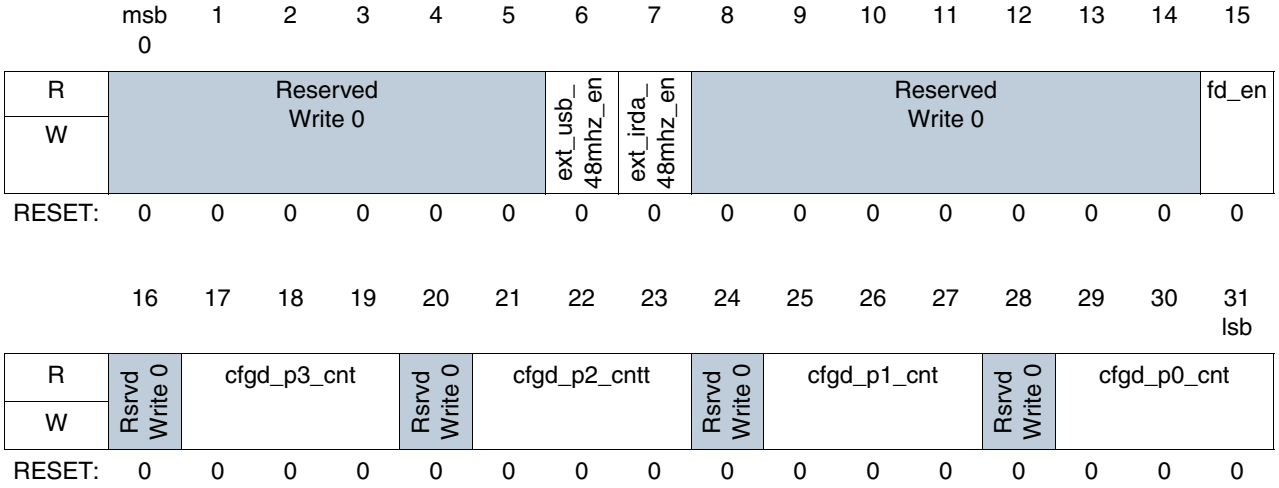

Figure 6. CDM 48-MHz Fractional Divider Configuration Register for Prior Versions of the MPC5200

3.3 SDRAM Memory Controller

3.3.1 New Module Design

The hardware for the SDRAM memory controller has been redesigned; however, the module has complete backward software compatibility with prior versions of the MPC5200.

3.3.2 Multiple Open Pages

The support of multiple open pages is an internal function of the MPC5200B memory controller. It adds functionality to the SDRAM interface but does not require any change to software.

3.3.3 Support of 16-bit Memory Interface

The SDRAM bus now supports both 16-bit and 32-bit memory structures. Bit 18 of the SDRAM bus control register (MBAR + 0x0104) has been added to select between the modes. To convert an existing MPC5200 design to the MPC5200B, bit 18 should be a logic 0 (select 32-bit port size).

3.3.4 SDRAM Bus Memory Controller Control Register—MBAR + 0x0104

3.3.4.1 Memory Controller Control Register for the MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	mode_en	cke	ddr	ref_en	Rsvd			hi_addr	Rsvd	drive_rule	ref_interval[0:5]					
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R W	Reserved		mem_ps	Rsvd	dqs_oe				Reserved					soft_ref	soft_pre	Rsvd
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7. Memory Controller Control Register for the 20065200B

3.3.4.2 Memory Controller Control Register for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R W	mode_en	cke	ddr	ref_en	Rsvd			hi_addr	Rsvd	drive_rule	ref_interval[0:5]					
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R W	Reserved				dqs_oe				Reserved					soft_ref	soft_pre	Rsvd
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8. Memory Controller Control Register for Prior Versions of the MPC5200

3.3.5 BURST TERMINATION

BURST TERMINATION is an SDRAM controller command. This command was not functional on revisions of the MPC5200 prior to the MPC5200B. The BURST TERMINATION command is fully functional on the MPC5200B.

The BURST command is issued automatically by the SDRAM bus memory controller to terminate a burst read or write when only one word of data needs to be transferred. This function is handled automatically by the SDRAM bus memory controller.

This is a hardware change inside the SDRAM memory controller and has no effect on system software. The BURST TERMINATION command simply improves performance on the SDRAM bus.

3.3.6 H/W Wakeup from Deep Sleep Mode

The MPC5200B SDRAM memory controller automatically handles putting the external memory devices into self-refresh mode when the processor enters the deep sleep mode. Likewise, when the deep sleep mode of the MPC5200B is exited, the SDRAM memory controller automatically causes the SDRAM to exit self-refresh mode.

Note that SDRAMs from various manufacturers require certain amounts of delay time between exiting the self-refresh mode and resuming data fetches. It is the responsibility of the software writer to insure that this delay is enforced.

When entering the deep sleep mode, the SDRAM memory controller automatically issues a series of commands to the SDRAM memory devices to put them into self-refresh mode. The SDRAM memory controller also controls the state of the clock enable pin to properly put the SDRAM devices into the self-refresh mode.

Hardware wakeup from deep sleep causes the SDRAM memory controller to automatically issue a series of commands to the SDRAM memory devices to take them out of self-refresh mode. Also, the state of the MPC5200B clock enable line is properly set by the SDRAM memory controller. No change to software is required because of this hardware change, other than to enforce the proper amount of delay between exiting the self-refresh mode of the SDRAM devices and the first memory access to those devices.

3.3.7 New 32-bit to 64-bit XLB Gasket

The new XL bus gasket and the improved read clock recovery circuit are internal improvements that do not affect system software. This change is a hardware change only and is for improvement of internal system bus performance.

3.3.8 New DQS Read Data Sampling Circuitry

Configuration register 1 MBAR + 0x0108 - Bit [4] has been added to the swt2rwp field.

The width of the single write to read-write-precharge delay field has been increased from 3 to 4 bits. The default state of the added bit is a logic 0. On prior versions of the MPC5200, this bit was reserved.

The user software for versions of the MPC5200 prior to the MPC5200B should be checked to see that the added bit was written to a logic 0 to insure compatibility.

3.3.9 SDRAM Bus Memory Controller Configuration Register 1—MBAR + 0x0108

3.3.9.1 Memory Controller Configuration Register 1 for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	srd2rwp				swt2rwp				rd_latency				Rsvd	act2rw		
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Rsvd	pre2act			ref2act				Rsvd	wr_latency			Reserved			
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 9. Memory Controller Configuration Register 1 for MPC5200B

Table 1. Memory Controller Configuration Register 1 Field Descriptions

Bit	Name	Description
4:7	swt2rwp	Single Write to Read/Write/Precharge delay. Limiting case is Write to Precharge. For DDR, suggested value = 0x3 (t_{WR} + 1 clk) For SDR, suggested value = 0x2 (t_{WR})

3.3.9.2 Memory Controller Configuration Register 1 for Prior Versions of the MPC5200

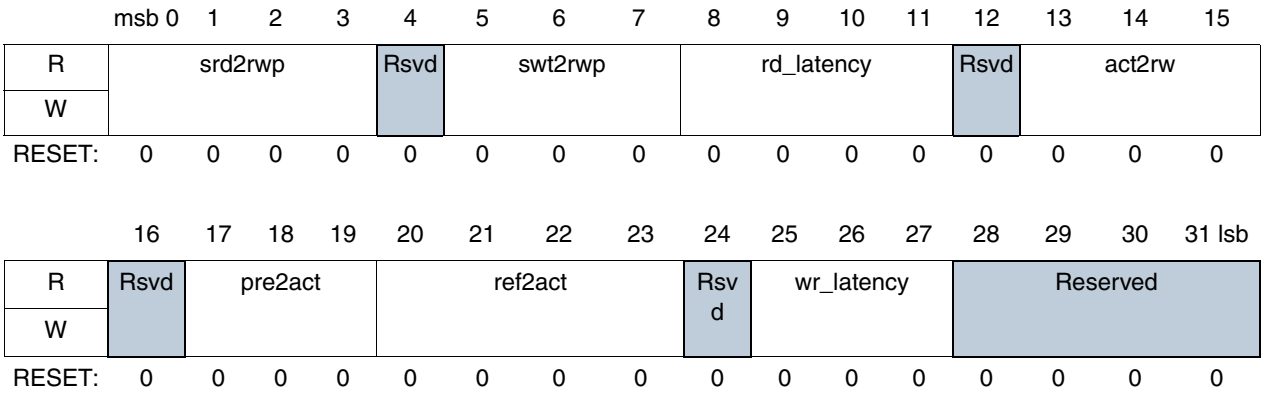

Figure 10. Memory Controller Configuration Register 1 for Prior Versions of the MPC5200

Table 2. Memory Controller Configuration Register 1 Field Descriptions

Bit	Name	Description
4	—	Reserved
5:7	swt2rwp	Single Write to Read/Write/Precharge delay. Limiting case is Write to Precharge. For DDR, suggested value = 0x3 (t_{WR} + 1 clk) For SDR, suggested value = 0x2 (t_{WR})

3.3.10 Read Clock Recovery (RCR)—Registers Added

The read clock recover registers are located from MBAR + 0x090 to MBAR + 0xBC. User software MUST NOT write to these registers.

User software should be checked to make sure that these memory locations are not accessed to insure MPC5200B compatibility.

3.4 BestComm

3.4.1 Bus Snoop Enable Bit Added in the XLB Arbiter

Bit 15 of the arbiter configuration register has been added. This bit enables BestComm XL address bus snooping. This feature is enabled from the release of reset.

The BestComm unit fetches data from memory and places it in its read line buffer in preparation for transmitting that data through one of the peripheral elements, such as the Ethernet port. If the CPU writes to the memory location just accessed by the BESTComm unit but before the data is transmitted from the read line buffer, the bus snooping logic will mark the data as dirty in the read line buffer and cause the XL bus arbiter to re-fetch the fresh data from memory.

User software should be checked to ensure that this bit is set to the desired setting.

3.4.2 BESTComm Arbiter Configuration Register—MBAR + 0x1F40

3.4.2.1 Arbiter Configuration Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	PLDIS	Rsvd														BSDIS
W																
RESET:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	SE	USE_WWF	TBEN	Rsvd	WS	SP			Rsvd	PM		Rsvd	BA	DT	AT	Rsvd
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Figure 11. Arbiter Configuration Register for MPC5200B

Table 3. Arbiter Configuration Field Descriptions

Bit	Name	Description
15	BSDIS	BestComm snooping disable. 0 BestComm XLB address snooping enabled. 1 BestComm XLB address snooping disabled.

3.4.2.2 Arbiter Configuration Register for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	PLDIS	Rsvd														
W																
RESET:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	SE	USE_WWF	TBEN	Rsvd	WS	SP			Rsvd	PM		Rsvd	BA	DT	AT	Rsvd
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Figure 12. Arbiter Configuration Register for Prior Versions of the MPC5200

Table 4. Arbiter Configuration Field Descriptions

Bit	Name	Description
1:15	—	Reserved

3.4.3 External Breakpoint

The SDMA debug module control register, for both the MPC5200B and prior versions of the MPC5200, are exactly alike; however, in versions of the MPC5200 prior to Revision B, the "E" and "EB" bits did not work. In the MPC5200B, the bits do work as described. If these bits were set to a logic 1 in versions of the MPC5200 prior to Revision B, external breakpoints would not cause the BestComm unit to halt. If software written for versions of the MPC5200 prior to Revision B, set the "E" and "EB" bits to a logic 1, and this software is used on the MPC5200B, external breakpoints will cause the BestComm unit to halt. For compatibility, check the software to see that the "E" and "EB" bits are set to a logic 0 when upgrading to the MPC5200B. This may affect existing user software that uses the SDMA debug module control register at MBAR + 0x1278.

3.4.3.1 SDMA Debug Module Control Register—MBAR + 0x1278

3.4.3.2 SDMA Debug Module Control Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Block Tasks															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	AA	B	Comparator Type 1			Comparator Type 2			and/ or	EU breakpoints				E	I	EB
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13. Debug Module Control Register for MPC5200B

Table 5. SDMA Debug Module Control Field Descriptions

Bit	Name	Description
0:15	Block Tasks	Specify for each of tasks 15-0, whether to block that task with detection of a breakpoint (bit 0 halts TASK 15, bit 1 halts TASK 14, etc) 0 Do not block task 1 Block the task
16	AA	AutoArm—specifies whether or not the triggered bit dbgStatusReg[16] will be automatically reset to 0 following the saving of context for a breakpoint. This bit is set to 0 at reset. 0 Triggered bit will not be automatically reset 1 Triggered bit will be automatically reset
17	B	Breakpoint—This bit specifies whether or not to take a breakpoint. This bit is set to 0 at reset. 0 Disable breakpoints 1 Enable breakpoints
18:20	Comparator Type 1	Comparator 1 type—These bits specify the type of data that has been loaded into comparator 1.

Table 5. SDMA Debug Module Control Field Descriptions (continued)

Bit	Name	Description
21:23	Comparators Type 2	Comparator 2 type—These bits specify the type of data that has been loaded into comparator 2.
24	and / or	AND/OR—This specifies what type of operation is to be used with the comparators. This bit is set to 0 at reset. 0 Indicates an OR'ing of the comparators 1 Indicates an AND'ing of the comparators
25:28	EU breakpoints	euBreakpoint These bits indicate that a breakpoint has occurred in one of the four execution units. Each execution unit has one bit dedicated to it. A 1 in any of these bits indicates that the associated execution unit has issued breakpoint. These bits are sticky and must be overwritten to continue. These bits are cleared to zero at reset. MPC5200B has integrated only EU3.
29	E	Enable external breakpoint to cause HALT condition. 0 Do not enable external breakpoint to cause a halt condition 1 Allow external breakpoint to cause a halt condition
30	I	Enable internal breakpoint 0 Do not enable internal breakpoint to cause a halt condition 1 Allow internal breakpoint to cause a halt condition
31	EB	Master breakpoint enable (this bit must be always set to allow any kind of breakpoint to halt the task) 0 Disable external breakpoint 1 Enable external breakpoint

3.4.3.3 SDMA Debug Module Control Register for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Block Tasks															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	AA	B	Comparator Type 1			Comparator Type 2			and/ or	EU breakpoints				E	I	EB
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14. SDMA Debug Module Control Register for Prior Versions of the MPC5200

Table 6. SDMA Debug Module Control Field Descriptions

Bit	Name	Description
0:15	Block Tasks	Specify for each of tasks 15-0, whether to block that task with detection of a breakpoint (bit 0 halts TASK 15, bit 1 halts TASK 14, etc) 0 Do not block task 1 Block the task
16	AA	AutoArm—specifies whether or not the triggered bit dbgStatusReg[16] will be automatically reset to 0 following the saving of context for a breakpoint. This bit is set to 0 at reset. 0 Triggered bit will not be automatically reset 1 Triggered bit will be automatically reset
17	B	Breakpoint—This bit specifies whether or not to take a breakpoint. This bit is set to 0 at reset. 0 Disable breakpoints 1 Enable breakpoints
18:20	Comparator Type 1	Comparator 1 type—These bits specify the type of data that has been loaded into comparator 1.
21:23	Comparators Type 2	Comparator 2 type—These bits specify the type of data that has been loaded into comparator 2.
24	and / or	AND/OR—This specifies what type of operation is to be used with the comparators. This bit is set to 0 at reset. 0 Indicates an OR'ing of the comparators 1 Indicates an AND'ing of the comparators
25:28	EU breakpoints	euBreakpoint These bits indicate that a breakpoint has occurred in one of the four execution units. Each execution unit has one bit dedicated to it. A 1 in any of these bits indicates that the associated execution unit has issued breakpoint. These bits are sticky and must be overwritten to continue. These bits are cleared to zero at reset. MPC5200B has integrated only EU3.
29	E	Enable external breakpoint to cause HALT condition. 0 Do not enable external breakpoint to cause a halt condition 1 Allow external breakpoint to cause a halt condition
30	I	Enable internal breakpoint 0 Do not enable internal breakpoint to cause a halt condition 1 Allow external breakpoint to cause a halt condition
31	EB	Master breakpoint enable (this bit must be always set to allow any kind of breakpoint to halt the task) 0 Disable external breakpoint 1 Enable external breakpoint

3.4.4 External BestComm Request (GPIO)

The ability to use simple and interrupt GPIO as BestComm requestors has been added.

The SDMA requestor muxcontrol register encoding values have been expanded to use the values %01 and %10. These new values allow various simple and interrupt GPIO pins to be selected as BestComm requestors.

This may affect user software. User code should be checked to make sure that reserved values were not used in software written for MPC5200 devices prior to MPC5200B to insure software compatibility.

3.4.5 SDMA Requestor MuxControl—MBAR + 0x125C

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Req31		Req30		Req29		Req28		Req27		Req26		Req25		Req24	
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Req23		Req22		Req21		Req20		Req19		Req18		Req17		Req16	
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 15. SDMA Request MuxControl

Table 7. SDMA Request MuxControl Field Descriptions

Bit	Name	Description for the MPC5200B (Mask Set M62C Rev. 1)	Description for Prior Versions of the MPC5200
0:1	Req31	00 Requestor (RESERVED) 01 GPIO_PSC2_3 10 GPIO_IRDA_1 11 Always Requestor 31	00 Requestor (RESERVED) 01-10 No active requestor 11 Always Requestor 31
2:3	Req30	00 Requestor (RESERVED) 01 GPIO_PSC2_2 10 GPIO_IRDA_0 11 Always Requestor 30	00 Requestor (RESERVED) 01-10 No active requestor 11 Always Requestor 30
4:5	Req29	00 Requestor (RESERVED) 01 GPIO_PSC2_1 10 GPIO_ETH_3 11 Always Requestor 29	00 Requestor (RESERVED) 01-10 No active requestor 11 Always Requestor 29
6:7	Req28	00 Requestor (RESERVED) 01 GPIO_PSC2_0 10 GPIO_ETH_2 11 Always Requestor 28	00 Requestor (RESERVED) 01-10No active requestor 11 Always Requestor 28
8:9	Req27	00 Requestor (RESERVED) 01 GPIO_PSC1_3 10 GPIO_ETH_1 11 Always Requestor 27	00 Requestor (RESERVED) 01-10 No active requestor 11 Always Requestor 27
10:11	Req26	00 Requestor IrDA TX (PSC_6) 01 GPIO_PSC1_2 10 GPIO_ETH_0 11 Always Requestor 26	00 Requestor IrDA TX (PSC_6) 01-10 No active requestor 11 Always Requestor 26
12:13	Req25	00 Requestor IrDA RX (PSC_6) 01 GPIO_PSC1_1 10 GPIO_USB_3 11 Always Requestor 25	00 Requestor IrDA RX (PSC_6) 01-10 No active requestor 11 Always Requestor 25
14:15	Req24	00 Requestor I2C1_TX 01 GPIO_PSC1_0 10 GPIO_USB_2 11 Always Requestor 24	00 Requestor I2C1_TX 01-10 No active requestor 11 Always Requestor 24

Table 7. SDMA Request MuxControl Field Descriptions (continued)

Bit	Name	Description for the MPC5200B (Mask Set M62C Rev. 1)	Description for Prior Versions of the MPC5200
16:17	Req23	00 Requestor I2C1_RX 01 GPIO_SINT_7 10 GPIO_USB_1 11 Always Requestor 23	00 Requestor I2C1_RX 01-10 No active requestor 11 Always Requestor 23
18:19	Req22	00 Requestor I2C2_TX 01 GPIO_SINT_6 10 GPIO_USB_0 11 Always Requestor 22	00 Requestor I2C2_TX 01-10 No active requestor 11 Always Requestor 22
20:21	Req21	00 Requestor I2C2_RX 01 GPIO_SINT_5 10 GPIO_PSC3_5 11 Always Requestor 21	00 Requestor I2C2_RX 01-10 No active requestor 11 Always Requestor 21
22:23	Req20	00 Requestor PSC4_TX 01 GPIO_SINT_4 10 GPIO_PSC3_4 11 Always Requestor 20	00 Requestor PSC4_TX 01-10 No active requestor 11 Always Requestor 20
24:25	Req19	00 Requestor PSC4_RX 01 GPIO_SINT_3 10 GPIO_PSC3_3 11 Always Requestor 19	00 Requestor PSC4_RX 01-10 No active requestor 11 Always Requestor 19
26:27	Req18	00 Requestor PSC5_TX 01 GPIO_SINT_2 10 GPIO_PSC3_2 11 Always Requestor 18	00 Requestor PSC5_TX 01-10 No active requestor 11 Always Requestor 18
28:29	Req17	00 Requestor PSC5_RX 01 GPIO_SINT_1 10 GPIO_PSC3_1 11 Always Requestor 17	00 Requestor PSC5_RX 01-10 No active requestor 11 Always Requestor 17
30:31	Req16	00 Requestor LP 01 GPIO_SINT_0 10 GPIO_PSC3_0 11 Always Requestor 16	00 Requestor LP 01-10 No active requestor 11 Always Requestor 16

3.5 PCI

3.5.1 SC PCI—Packet Size Increase

- Tx packet size PCITPSR(RW)—0x3800—The Packet_Size field has been increased in width from 16 to 32 bits.
- Tx done counts PCITDCR(R)—0x3818—The Bytes_Done field has been increased from 16 to 32 bits. The Packets_Done field has been moved to the Tx packets done counts PCITPDCR(R) at MBAR + 0x3820, and this field has been increased from 16 to 32 bits.
- Add Tx packets done counts register—PCITPDCR(R) at MBAR + 0x3820—for changed Packets_Done and increased field size from 16 to 32 bits.

- Rx packet size PCIRPSR(RW)—0x3880—The Packet_Size field has been increased in width from 16 to 32 bits.
- RxDone counts PCIRDCR(R)—0x3898—The Bytes_Done field has been increased from 16 to 32 bits. The Packets_Done field has been moved to the Rx packets done counts PCIRPDCR(R) at MBAR + 0x38A0, and this field has been increased from 16 to 32 bits.
- Add Rx packets done counts register—PCIRPDCR(R) at MBAR + 0x38A0—for changed Packets_Done and increased field size from 16 to 32 bits.

3.5.2 Tx Packet Size PCITPSR(RW)—MBAR + 0x3800

3.5.2.1 Tx Packet Size PCITPSR(RW) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Packet_Size[31:16]															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Packet_Size[15:2]														PacketSize[1:0]	
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 16. Tx Packet Size PCITPSR(RW) for MPC5200B

Table 8. Tx Packet Size PCITPSR(RW) Field Descriptions

Bits	Name	Description
0:31	Packet_Size	User writes the number of bytes for transmit controller to send over PCI.The two low bits are hardwired low; only 32-bit data transfers to the FIFO are allowed. Writing to this register also completes a Restart Sequence as long as the Master Enable bit, PCITER[ME], is high and Reset Controller bit, PCITER[RC], is low.

3.5.2.2 Tx Packet Size PCITPSR(RW) for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Packet_Size[16:2]														Packet_Size[1:0]	
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17. Tx Packet Size PCITPSR(RW) for Prior Versions of the MPC5200

Table 9. Tx Packet Size PCITPSR(RW) Field Descriptions

Bits	Name	Description
0:15	Packet_Size	User writes the number of bytes for transmit controller to send over PCI.The two low bits are hardwired low; only 32-bit data transfers to the FIFO are allowed. Writing to this register also completes a restart sequence as long as the master enable bit, PCITER[ME], is high and reset controller bit, PCITER[RC], is low.
16:31	Reserved	Unused. Software should write zero to these bits.

3.5.3 Tx Bytes Done Counts PCITDCR(R)—MBAR + 0x3818

3.5.3.1 Tx Bytes Done Counts PCITDCR(R) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18. Tx Bytes Done Counts PCITDCR(R) for MPC5200B

Table 10. Tx Bytes Done Counts PCITDCR(R) Field Descriptions

Bits	Name	Description
0:31	Bytes_Done	This status register indicates the number of bytes transmitted since the start of a packet. It is updated at the end of each successful PCI data beat. For normally terminated packets, the Bytes_Done value and the Packet_Size values will be equal. If continuous mode is active the Bytes_Done value will read zero at the end of a successful packet, and the Packets_Done field will be incremented.

3.5.3.2 Tx Done Counts PCITDCR(R) for Prior Versions of the MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Packets_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 19. Tx Done Counts PCITDCR(R) for Prior Versions of the MPC5200B

Table 11. Tx Done Counts PCITDCR(R) Field Descriptions

Bits	Name	Description
0:15	Bytes_Done	This status register indicates the number of bytes transmitted since the start of a packet. It is updated at the end of each successful PCI data beat. For normally terminated packets, the Bytes_Done value and the Packet_Size values will be equal. If continuous mode is active, the Bytes_Done value will read zero at the end of a successful packet, and the Packets_Done field will be incremented.
16:31	Packets_Done	This status register indicates the number of packets transmitted and is active only if continuous mode is in effect. The counter is reset if the following occurs • Reset controller bit, PCITER[RC], is asserted (normal way to restart continuous mode) • Master enable bit, PCITER[ME], becomes negated Master enable can reset Packets_Done status without disturbing continuous mode addressing. At any point in time, the total number of bytes transmitted can be calculated as (Packets_Done x Packet_Size) + Bytes_Done This assumes Packet_Size is the same for all restart sequences.

3.5.4 Rx Packet Size PCIRPSR(RW)—MBAR + 0x3880

3.5.4.1 Rx Packet Size PCIRPSR(RW) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Packet_Size[31:16]															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Packet_Size[15:2]														Packet_Siz[1:0]	
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 20. Rx Packet Size PCIRPSR(RW) for MPC5200B

Table 12. Rx Packet Size PCIRPSR(RW) Field Descriptions

Bits	Name	Description
0:31	Packet_Size	The user writes this register with the number of bytes for receive controller to fetch over PCI. The two low bits are hardwired low; only 32-bit data transfers to the FIFO are allowed. Writing to this register also completes a restart sequence as long as master enable bit, PCIRER[ME], is high and reset controller bit, PCIRER[RC], is low.

3.5.4.2 Rx Packet Size PCIRPSR(RW) for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Packet_Size[16:2]														Packet_Size[1:0!]	
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 21. Rx Packet Size PCIRPSR(RW) for Prior Versions of the MPC5200

Table 13. Rx Packet Size PCIRPSR(RW) Field Descriptions

Bits	Name	Description
0:15	Packet_Size	The user writes this register with the number of bytes for receive controller to fetch over PCI. The two low bits are hardwired low; only 32-bit data transfers to the FIFO are allowed. Writing to this register also completes a restart sequence as long as master enable bit, PCIRER[ME], is high and reset controller bit, PCIRER[RC], is low.
16:31	Reserved	Unused bits. Software should write zero to these bits. No bus error is generated

3.5.5 Rx Bytes Done Counts PCIRDCR(R) —MBAR + 0x3898

3.5.5.1 Rx Bytes Done Counts PCIRDCR(R) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22. Rx Bytes Done Counts PCIRDCR(R) for MPC5200B

Table 14. Rx Bytes Done Counts PCIRDCR(R) Field Descriptions

Bits	Name	Description
0:31	Bytes_Done	This status register indicates the number of bytes received since the start of a packet. It is updated at the end of each successful PCI data beat. For normally terminated packets, the Bytes_Done value and the Packet_Size values are equal. If continuous mode is active, the Bytes_Done value reads 0 at the end of a successful packet, and the Packets_Done field is incremented.

3.5.5.2 RxDone Counts PCIRDCR(R) for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Bytes_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Packets_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 23. RxDone Counts PCIRDCR(R) for Prior Versions of the MPC5200

Table 15. xDone Counts PCIRDCR(R) Field Descriptions

Bits	Name	Description
0:15	Bytes_Done	This status register indicates the number of bytes received since the start of a packet. It is updated at the end of each successful PCI data beat. For normally terminated packets, the Bytes_Done value and the Packet_Size values are equal. If continuous mode is active, the Bytes_Done value reads 0 at the end of a successful packet, and the Packets_Done field is incremented.
16:31	Packets_Done	This status register indicates the number of packets received. It is active only if continuous mode is in effect. If the following occurs, the counter is reset: • Reset controller bit, PCIRER[RC], is asserted (normal way to restart continuous mode) • Master enable bit, PCIRER[ME], is negated In this way, master enable can be used to reset Packets_Done status without disturbing continuous mode addressing. At any point in time the total number of Bytes received can be calculated as (Packets_Done x Packet_Size) + Bytes_Done This assumes Packet_Size is the same for all restart sequences.

3.5.6 Tx Packets Done Counts PCITPDCR(R) —MBAR + 0x3820

3.5.6.1 Tx Packets Done Counts PCITPDCR(R) for MPC5200B

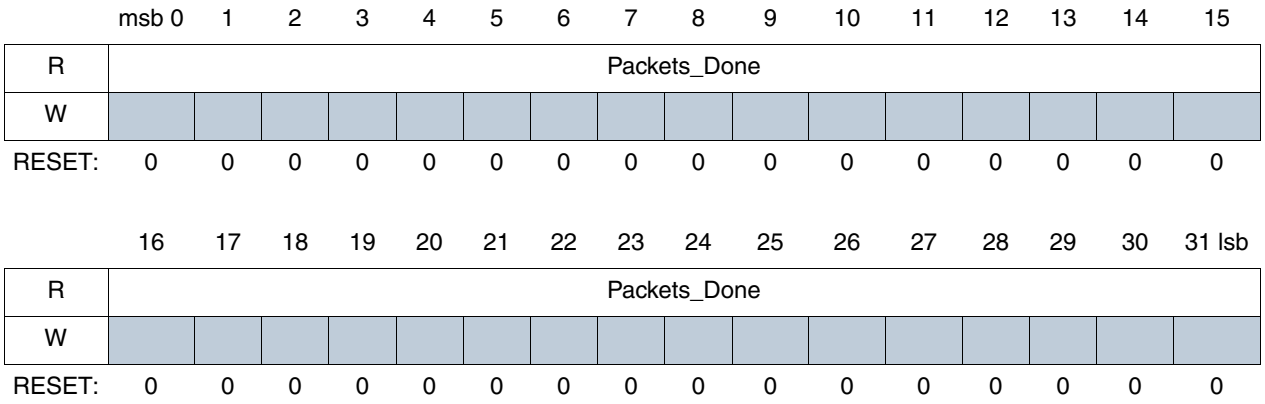

Figure 24. x Packets Done Counts PCITPDCR(R) for MPC5200B

Table 16. Tx Packets Done Counts PCITPDCR(R) Field Descriptions

Bits	Name	Description
0:31	Packets_Done	This status register indicates the number of packets transmitted and is active only if continuous mode is in effect. The counter is reset if the following occurs • Reset controller bit, PCITER[RC], is asserted (normal way to restart continuous mode) • Master enable bit, PCITER[ME], becomes negated Master enable can reset Packets_Done status without disturbing continuous mode addressing. At any point in time, the total number of bytes transmitted can be calculated as (Packets_Done x Packet_Size) + Bytes_Done This assumes Packet_Size is the same for all restart sequences.

The Tx packets done counts register has been added to the MPC5200B. There is no corresponding register in prior versions of the MPC5200.

3.5.7 Rx Packets Done Counts PCIRPDCR(R) —MBAR + 0x38A0

3.5.7.1 Rx Packets Done Counts PCIRPDCR(R) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Packets_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Packets_Done															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 25. Rx Packets Done Counts PCIRPDCR(R) for MPC5200B

Table 17. Rx Packets Done Counts PCIRPDCR(R) Field Descriptions

Bits	Name	Description
0:31	Packets_Done	This status register indicates the number of packets received. It is active only if continuous mode is in effect. If the following occurs, the counter is reset: • Reset controller bit, PCIRER[RC], is asserted (normal way to restart continuous mode) • Master enable bit, PCIRER[ME], is negated In this way, master enable can be used to reset Packets_Done status without disturbing continuous mode addressing. At any point in time the total number of Bytes received can be calculated as (Packets_Done x Packet_Size) + Bytes_Done This assumes Packet_Size is the same for all restart sequences.

The Rx packets done counts register has been added to the MPC5200B. There is no corresponding register in prior versions of the MPC5200.

3.5.8 XTPCI Write Combining Programmability

The target control register PCITCR(RW)—0xD6C—has additional bits for the write combine feature:

- Bit 23—WCD—write combine disable
- Bits[24:31]—WCT[7:0] write combine timer

The purpose of this feature, added to the MPC5200B, allows partial packets of incoming data to be transferred to the XL bus if a break in incoming PCI data occurs greater than specified by the write combine timer.

This feature is a hardware function that was added to improve system bus performance and does not affect user software. This feature can be disabled by the WCD bit in the PCITCR register.

NOTE

The MPC5200B allows an external burst write to the internal XL Bus. Prior versions of the MPC5200 do not allow an external PCI initiator to do a write burst on the XL Bus.

3.5.9 Target Control Register PCITCR(RW) —MBAR + 0x0D6C

3.5.9.1 Target Control Register PCITCR(RW) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved							LD	Reserved							P
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved							WCD	Write Combine Timer [7:0]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 26. Target Control Register PCITCR(RW) for MPC5200B

Table 18. Target Control Field Descriptions

Bits	Name	Description
16:22	Reserved	Unused bits. Software should write zero to this register.
23	Write Combine Disable (WCD)	This control bit applies only when MPC5200 is Target. When set, it prevents the PCI Controller from automatically combining write data to be sent out on the XL bus as a burst, if possible. Instead, data is transferred as soon as possible on the XL bus as single-beat transactions. Better target write performance is achieved when this bit cleared.
24:31	Write Combine Timer (WCT)	This register contains the timer value, in PCI clocks, used when a partial burst has been buffered in the target write data path and write data stops being transferred to local memory from the external PCI device. Every time a sequential beat of write data is stored in the buffer, the counter is reset with this value. If partial burst data has been buffered, thereby activating the count-down counter, and this field is reprogrammed to a value less than the current counter value, the counter will jump down to the new write combine timer value. This way, software can force the write buffer to flush data to the XL bus more quickly than when the counter was initialized. The reset value of the write combine timer is 0x08. All 8 bits are programmable.

3.5.9.2 Target Control Register PCITCR(RW) for Prior Versions of the MPC5200

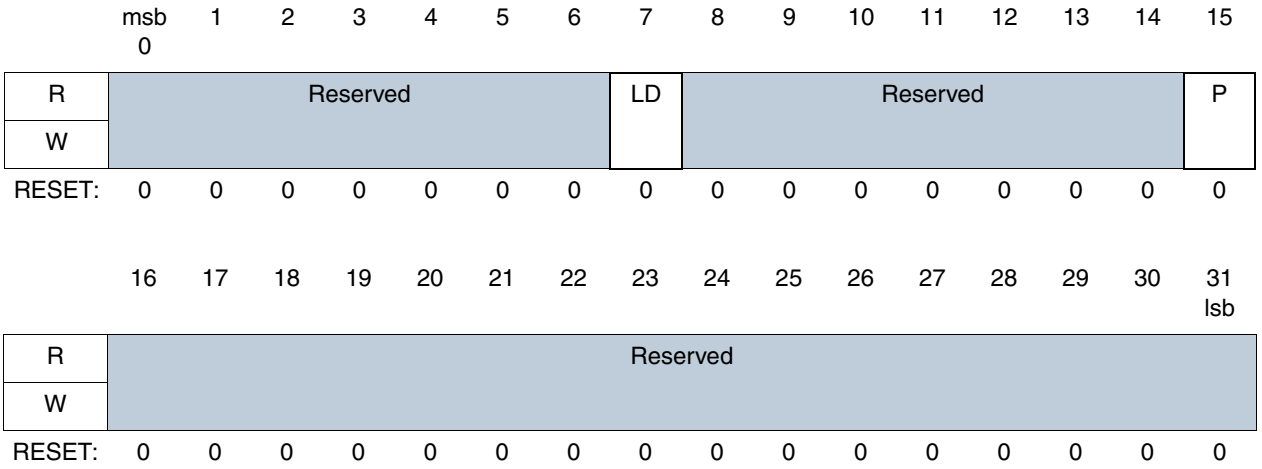

Figure 27. Target Control Register PCITCR(RW) for Prior Versions of the MPC5200

Table 19. Target Control Field Descriptions

Bits	Name	Description
16:31	Reserved	Unused bits. Software should write zero to this register.

3.5.10 PCITTCR and PCIRTCR

- Tx transaction control register PCITTCR
- Rx transaction control register PCIRTCR

On prior revisions of the MPC5200, setting the Max_Retries field of either the PCITTCR or PCIRTCR registers to 0xFF or 0x00 would allow INFINITE retries. The MPC5200B only uses the value 0x00 to allow INFINITE retries. Now, the value of 0xFF will specify 255 retires.

3.5.11 Tx Transaction Control Register PCITTCR(RW) —MBAR + 0x3808

3.5.11.1 Tx Transaction Control Register PCITTCR(RW) for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved				PCI_cmnd				Max_Retries							
W																
RESET:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved					Max_Beats			Reserved			W	Reserved			DI
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 28. Tx Transaction Control Register PCITTCR(RW) for MPC5200

Table 20. Tx Transaction Control Field Descriptions

Bits	Name	Description
8:15	Max_Retries	The user writes this field with the maximum number of retries to permit perpacket. The retry counter is reset when the packet completes normally or is terminated by a master abort, target abort, or an abort due to exceeding the retry limit. A slow or malfunctioning target might issue infinite disconnects and therefore permanently tie up the PCI bus. A finite (0x01 to 0xff) Max_Retries value will detect this condition and generate an interrupt. Setting Max_Retries to 0x00 will not generate any interrupt.

3.5.11.2 Tx Transaction Control Register PCITTCR(RW) for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved				PCI_cmnd				Max_Retries							
W																
RESET:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved					Max_Beats			Reserved			W	Reserved			DI
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 29. x Transaction Control Register PCITTCR(RW) for Prior Versions of the MPC5200

Table 21. Transaction Control Field Descriptions

Bits	Name	Description
8:15	Max_Retries	The user writes this field with the maximum number of retries to permit perpacket. The retry counter is reset when the packet completes normally or is terminated by a master abort, target abort, or an abort due to exceeding the retry limit. A slow or malfunctioning Target might issue infinite disconnects and therefore permanently tie up the PCI bus. A finite (0x01 to 0xfe) Max_Retries value will detect this condition and generate an interrupt. Setting Max_Retries to 0x00 or 0xff will not generate any interrupt.

3.5.12 Rx Transaction Control Register PCIRTCR(RW)—MBAR + 0x3888

3.5.12.1 Rx Transaction Control Register PCIRTCR(RW) for MPC5200B

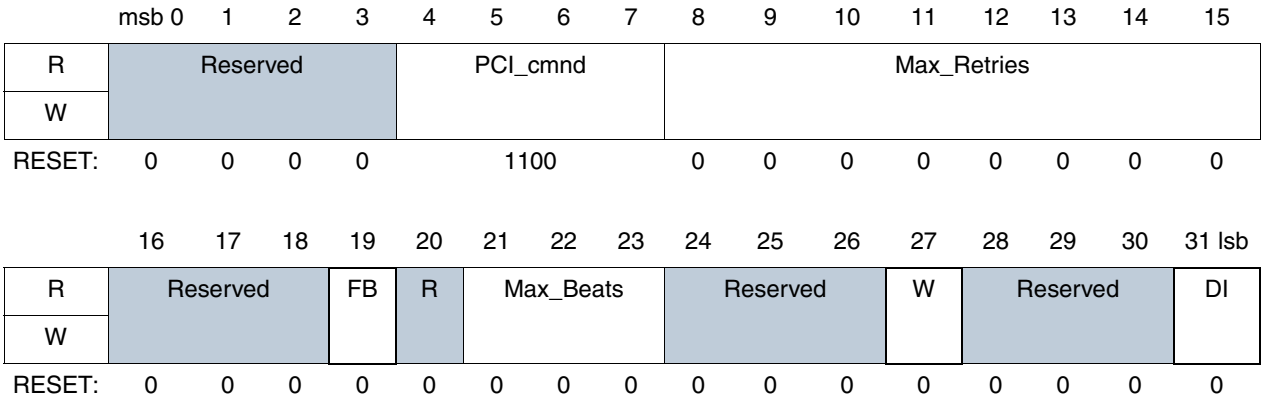

Figure 30. Rx Transaction Control Register PCIRTCR(RW) for MPC5200B

Table 22. Rx Transaction Control Field Descriptions

Bits	Name	Description
8:15	Max_Retries	The user writes this field with the maximum number of retries to permit per packet. The retry counter is reset when the packet completes normally or is terminated by a master abort, target abort, or an abort due to exceeding the retry limit. A slow or malfunctioning Target might issue infinite disconnects and therefore permanently tie up the PCI bus. A finite (0x01 to 0xff) Max_Retries value will detect this condition and generate an interrupt. Setting Max_Retries to 0x00 will not generate any interrupt.

3.5.12.2 Rx Transaction Control Register PCIRTCR(RW) for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Reserved				PCI_cmnd				Max_Retries							
W																
RESET:	0	0	0	0	1100				0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved			FB	R	Max_Beats			Reserved			W	Reserved			DI
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31. Rx Transaction Control Register PCIRTCR(RW) for Prior Versions of the MPC5200

Table 23. Rx Transaction Control Field Descriptions

Bits	Name	Description
8:15	Max_Retries	The user writes this field with the maximum number of retries to permit per packet. The retry counter is reset when the packet completes normally or is terminated by a master abort, target abort, or an abort due to exceeding the retry limit. A slow or malfunctioning Target might issue infinite disconnects and therefore permanently tie up the PCI bus. A finite (0x01 to 0xfe) Max_Retries value will detect this condition and generate an interrupt. Setting Max_Retries to 0x00 or 0xff will not generate any interrupt.

3.6 PSC

3.6.1 New Serial Interface Control Register Bits SICR (MBAR + PSCx + 0x40)

Bit 9 MultiWd functionality is moved to bit 14 and renamed to ESAI.

More than one audio sample is transferred at the start of each frame when this mode is enabled. The number of samples transferred per frame is determined by the CCR (frame sync frequency)., CTUR (frame sync length), and Codec mode (8-,16-, or 32-bit).

Bit 9 becomes I2S bit.

When this bit is enabled, one data sample is transferred on every transition of the frame signal. The CCR (frame sync frequency) determines the frame length, and the Codec mode (8-,16-, or 32-bit) determines the size of the data transferred.

Bit 14 becomes the ESAI bit.

Bit 15 is added to enable the enhanced AC97 mode.

Bit 21 is added to enable/disable EOF generation on the occurrence of UART errors.

These changes will affect user software in CODEC applications.

3.6.1.1 Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of MPC5200B

	msb 0	1	2	3	4	5	6	7
R	ACRB	AWR	DTS1	SHDIR	SIM[3:0]			
W								
RESET:	0	0	0	0	0	0	0	0

	8	9	10	11	12	13	14	15
R	GenClk	I2S	ClkPol	SyncPol	CellSlave	Cell2xClk	ESAI	EnAC97
W								
RESET:	0	0	0	1	0	0	0	0

	16	17	18	19	20	21	22	23 lsb
R	SPI	MSTR	CPOL	CPHA	UseEOF	Disable_EOF	Reserved	
W								
RESET:	0	0	0	0	0	0	0	0

Figure 32. Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of MPC5200B

Table 24. Serial Interface Control Field Descriptions

Bit	Name	Description
9	I2S	**Codec**—I2S mode 0 no I2S mode supported 1 PSC works in I2S mode **Other modes**—Reserved
14	ESAI	**Codec**—enhanced serial audio interface 0 PSC doesn't support the ESAI mode. 1 PSC supports the ESAI mode. This mode allows the PSC to send and receive more the one data word per frame, if the frame length is greater than the word length. The PSC sends only complete data words. **Other modes**—Reserved
15	EnAC97	**Codec**—enhanced AC97 mode—takes effect only when the AC97 mode is selected (SIM = 0x3) 0 No effect 1 If the AC97 mode was selected, the PSC uses the enhanced AC97 mode to transmit and receive the data. **Other modes**—Reserved
21	Disable_ EOF	**UART/SIR** —Disable EOF generation 0 The UART receiver generates an EOF tag if an UART error was detected. For more information's regarding the UART errors (RB, FE,PE, CDE,). 1 The UART receiver doesn't generate an EOF tag if an UART error was detected **Other modes**—Reserved
22:23	—	Reserved

3.6.1.2 Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7 lsb
R	ACRB	AWR	DTS1	SHDIR	SIM[3:0]			
W								
RESET:	0	0	0	0	0	0	0	0

	8	9	10	11	12	13	14	15
R	GenClk	MultiWd	ClkPol	SyncPol	CellSlave	Cell2xClk	Reserved	
W								
RESET:	0	0	0	1	0	0	0	0

	16	17	18	19	20	21	22	23 msb
R	SPI	MSTR	CPOL	CPHA	UseEOF	Reserved		
W								
RESET:	0	0	0	0	0	0	0	0

Figure 33. Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of Prior Versions of the MPC5200

Table 25. Serial Interface Control Field Descriptions

Bit	Name	Description
9	MultiWd	**Codec**—Multi word mode 0 PSC sends and receives only one data word per frame, even if the frame length is greater than the word length. 1 PSC sends and receive more the one data word per frame, if the frame length is greater than the word length. The PSC send only complete data words. This bit is used to support the I2S mode. **Other modes**—Reserved
14:15	—	Reserved
21:23	—	Reserved

3.6.2 Infrared Control 1 (MBAR + PSCx + 0x44) IRCR1

This register controls the configuration in one of the IrDA modes (SIR/MIR/FIR).

Bit 2—INV_RX bit added for inverting of the incoming receive signal.

This register controls the configuration in one of the IrDA modes (SIR/MIR/FIR).

3.6.2.1 Infrared Control 1 (MBAR + PSCx + 0x44) for SIR Mode for MPC5200B

	msb 0	1	2	3		5	6	7 lsb
R W	Reserved		INV_RX	Reserved		FD	Reserved	SPUL
RESET:	0	0	0	0	0	0	0	0

	msb 0	1	2	3	4	5	6	7 lsb
R W	Reserved					FD	SIPEN	Reserved
RESET:	0	0	0	0	0	0	0	0

Figure 34. Infrared Control 1 (MBAR + PSCx + 0x44) for MIR/FIR Modes for MPC5200B

Table 26. Infrared Control 1 Field Descriptions

Bit	Name	Description
2	INV_RX	**SIR / MIR / FIR**—Invert the RX line 0 The receiver doesn't invert the receive line. 1 The receiver invert the receive line. **Other modes**—Reserved

3.6.2.2 Infrared Control 1 MBAR + PSCx + (0x44) for SIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7 lsb
R	Reserved					FD	Reserved	SPUL
W								
RESET:	0	0	0	0	0	0	0	0

	msb 0	1	2	3	4	5	6	7 lsb
R	Reserved					FD	SIPEN	Reserved
W								
RESET:	0	0	0	0	0	0	0	0

Table 27. Infrared Control 1 (MBAR + PSCx + 0x44) for MIR/FIR Modes for Prior Versions of the MPC5200

Table 28. Infrared Control 1 Field Descriptions

Bit	Name	Description
0:4	—	Reserved

This bit has been added for additional functionality when using the SIR/MIR/FIR modes of the programmable serial controller. This bit allows the incoming IR data to be inverted. The default state of this bit is a logic 0 that is compatible with prior versions of the MPC5200.

3.6.3 Enhanced AC97 Mode

A number of changes were made to assist software using the AC97 interface of the PSC. In addition, the PSC will use the hardware sample rate conversion used by some AC97 codecs. The PSC now creates the AC97 frames in hardware using new registers and bits in the PSC when the enhanced AC97 mode is enabled.

ISR/IMR/SR (PSC_MBAR + 0x14 and 0x04) bits [12:15] have been added for the new event status of the AC97 enhanced mode.

3.6.4 Interrupt Status Register (MBAR + PSCx + 0x14)

3.6.4.1 Interrupt Status Register (MBAR + PSCx + 0x14) for UART / SIR Mode for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	IPC	Reserved		ORERR	TxEMP	DB	RxRDY FFULL	TxRDY	Reserved	Error	Reserved					
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 35. Interrupt Status Register (MBAR + PSCx + 0x14) for UART / SIR Mode for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	IPC	Reserved		ORERR	URERR	Reserved	RxRDY FFULL	TxRDY	DEOF	Error	Reserved		CMD_SEND	DATA_OVR	DATA_VALID	UNEX_RX_SLOT
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36. Interrupt Status Register (MBAR + PSCx + 0x14) Other modes

Table 29. Interrupt Status Field Descriptions

Bit	Name	Description
9	Error	**Error** This bit is identical to the error bit in the SR register. To clear this interrupt, use the reset error status command in the CR register.
12	CMD_SEND	**Enhanced AC97 mode**—command send ready This bit is identical to the CMD_SEND bit in the SR register. To clear this interrupt, use the reset error status command in the CR register. **Other modes**—Reserved
13	DATA_OVR	**Enhanced AC97 mode**—receive data overwrite This bit is identical to the DATA_OVR bit in the SR register. To clear this interrupt, use the reset error status command in the CR register. **Other modes**—Reserved
14	DATA_VALID	**Enhanced AC97 mode**—received status data This bit is identical to the DATA_VALID bit in the SR register. To clear this interrupt, use the reset error status command in the CR register. **Other modes**—Reserved
15	UNEX_RX_SLOT	**Enhanced AC97 mode**—unexpected RX slots detect This bit is identical to the UNEX_RX_SLOT bit in the SR register. To clear this interrupt, use the reset error status command in the CR register. **Other modes**—Reserved

3.6.4.2 Interrupt Status Register (MBAR + PSCx + 0x14) for UART / SIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	IPC	Reserved		ORERR	URERR	DB	RxRDY FFULL	TxRDY	Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 37. Interrupt Status Register (MBAR + PSCx + 0x14) for UART / SIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	IPC	Reserved		ORERR	URERR	Reserved	RxRDY FFULL	TxRDY	DEOF	Reserved						
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 38. Interrupt Status Register (MBAR + PSCx + 0x14) Other Modes

Table 30. Interrupt Status Field Descriptions

Bit	Name	Description
9:15	—	Reserved

SICR bit 15, EnAC97, enables the enhanced AC97 mode. If this mode is not enabled, the interface behaves the same as MPC5200 Rev A.

3.6.4.3 Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes

3.6.4.4 Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of MPC5200B

	msb 0	1	2	3	4	5	6	7
R	ACRB	AWR	DTS1	SHDIR	SIM[3:0]			
W								
RESET:	0	0	0	0	0	0	0	0

	8	9	10	11	12	13	14	15
R	GenClk	I2S	ClkPol	SyncPol	CellSlave	Cell2xClk	ESAI	EnAC97
W								
RESET:	0	0	0	1	0	0	0	0

	16	17	18	19	20	21	22	23 lsb
R	SPI	MSTR	CPOL	CPHA	UseEOF	Disable_EOF	Reserved	
W								
RESET:	0	0	0	0	0	0	0	0

Figure 39. Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of MPC5200B

Table 31. Serial Interface Control Field Descriptions

Bit	Name	Description
9	I2S	**Codec**—I2S mode 0 no I2S mode supported 1 PSC works in I2S mode **Other modes**—Reserved
14	ESAI	**Codec**—enhanced serial audio interface 0 PSC doesn't support the ESAI mode. 1 PSC supports the ESAI mode. This mode allows the PSC to send and receive more the one data word per frame, if the frame length is greater than the word length. The PSC sends only complete data words. **Other modes**—Reserved
15	EnAC97	**Codec** —enhanced AC97 mode—takes effect only when the AC97 mode is selected. (SIM = x3) 0 No effect 1 If the AC97 mode was selected the PSC use the "Enhanced AC97" mode to transmit and receive the data. **Other modes**—Reserved
21	Disable_ EOF	**UART/SIR** —disable EOF generation 0 The UART receiver generate an EOF tag if an UART error is detected. For more information's regarding the UART errors (RB, FE,PE, and CDE). 1 The UART receiver doesn't generate an EOF tag if an UART error is detected. **Other modes**—Reserved
22:23	—	Reserved

3.6.4.5 Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7 lsb
R	ACRB	AWR	DTS1	SHDIR	SIM[3:0]			
W								
RESET:	0	0	0	0	0	0	0	0

	8	9	10	11	12	13	14	15
R	GenClk	MultiWd	ClkPol	SyncPol	CellSlave	Cell2xClk	Reserved	
W								
RESET:	0	0	0	1	0	0	0	0

	16	17	18	19	20	21	22	23 msb
R	SPI	MSTR	CPOL	CPHA	UseEOF	Reserved		
W								
RESET:	0	0	0	0	0	0	0	0

Figure 40. Serial Interface Control Register (MBAR + PSCx + 0x40) for All Modes of Prior Versions of the MPC5200

Table 32. Serial Interface Control Field Descriptions

Bit	Name	Description
9	MultiWd	**Codec**—multi word mode 0 PSC sends and receives only one data word per frame, even if the frame length is greater than the word length. 1 PSC sends and receives more the one data word per frame, if the frame length is greater than the word length. The PSC send only complete data words. This bit is used to support the I2S mode. **Other modes**—Reserved
14:15	—	Reserved
21:23	—	Reserved

3.7 Registers Added To Implement the Enhanced AC97 Mode

The following registers have been added to the MPC5200B to implement the enhanced AC97 mode:

- AC97_SLOTS (PSC_MBAR + 0x24)—defines the active data slots in the AC97 frame
- AC97_CMD (PSC_MBAR + 0x28)—interface for AC97 mixer registers
- AC97_DATA (PSC_MBAR + 0x2C)—interface for AC97 mixer register read

3.7.1 AC97 Slots Register (0x24)—AC97Slots

This register has been added to the MPC5200B to implement the enhanced AC97 mode.

This write-only register defines which slots are expected in a receive AC97 frame and which slots will be sent in a AC97 TX frame. If the received frame doesn't match the expected slots, the SR [UNEXP_RX_SLOTS] bit will be set. This register has affect only if AC97 mode is selected in the SICR register, and the EnAC97 bit is active.

3.7.1.1 AC97 Slots Register (MBAR + PSCx + 0x24)—AC97Slots

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved															
W	Reserved						TX_Slots[3:12]									
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	Reserved															
W	Reserved						RX_Slots[3:12]									
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 41. AC97 Slots Register (MBAR + PSCx + 0x24)—AC97Slots

Table 33. AC97 Slots Field Descriptions

Bit	Name	Description
0:5	—	Reserved
6:15	TX_Slots[3:12]	**Enhanced AC97 mode**—expected receive slots The bits in this register specify which data slots [3:12] will be sent in an AC97 TX frame. The AC97 transmitter will use this information to generate the Slot0 and read out the according number of data words from the TXFIFO. If the TXFIFO is empty an empty AC97 frame will be sent until new data are available. **Other modes**—Reserved
16:21	—	Reserved
22:3‘	RX_Slots[3:12]	**Enhanced AC97 mode**—expected receive slots The bits in this register specify which data slots [3:12] in the receive AC97 frame must contain valid data. The AC97 Codec selects the valid data slots by setting the according data valid bit in Slot0[12:3]. If the received valid slots not match the expected slots the unexpected slot received state occurs. Only if the received slots match the expected slots, will the received data written to the RXFIFO. If the receiver detects an AC97 frame without data (frame is empty or contains only status data), the unexpected slot received state will not occur. **Other modes**—Reserved

3.7.2 AC97 Command Register (MBAR + PSCx + 0x28)—AC97CMD

This register has been added to the MPC5200B to implement the enhanced AC97 mode.

This register contains the AC97 address for transmit slot1 and the AC97 command data for transmit slot 2. A write access to any byte of this register will set the SR [CMD_SEND] bit to one. The AC97 transmitter generates a frame with valid slot1 and slot2 and paste the values of this register to the next transmitted slot1 and slot2. If the data was sent, then the SR [CMD_SEND] bit will be cleared by the transmitter.

3.7.2.1 AC97 Command Register (MBAR + PSCx + 0x28)—AC97CMD

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	A97 CMD	AC97 Control Register Index							AC97 Command Data[15:8]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
R	AC97 Command Data[7:0]								Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42. AC97 Command Register (MBAR + PSCx + 0x28)—AC97CMD

Table 34. AC97 Command Field Descriptions

Bit	Name	Description
0	AC97 CMD	**Enhanced AC97 mode**—AC97 command This bit indicates if the access to the Control Register is a read or write access. It will be pasted to the Slot1 bit 19. 0 write access 1 read access **Other modes**—Reserved
1:7	AC97 Control Register Index	**Enhanced AC97 mode**—AC97 address register This register contains target control register address. It will be pasted to the Slot1 bit 18 to 12. **Other modes**—Reserved
6:23	AC97 Command Data	**Enhanced AC97 mode**—AC97 command data register This register is used to define the command data value for a write command. It will be pasted to the Slot2 bit 19 to 4. **Other modes**—Reserved
24:31	—	Reserved

3.7.3 AC97 Status Data Register (MBAR + PSCx + 0x2C)—AC97Data

This register has been added to the MPC5200B to implement the enhanced AC97 mode.

This read-only register contains the received response of a AC97 read command. If this register contains new data, then the SR [DATA_VALID] will be set to one by the receiver. A read access to this register clears the SR [DATA_VALID] bit.

3.7.3.1 AC97 Status Data Register (MBAR + PSCx + 0x2C)—AC97Data

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved	AC97 Register Index Echo							AC97 Control Register Read Data[15:8]							
W	Reserved															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	39	31
R	AC97 Control Register Read Data[7:0]								Reserved							
W	Reserved															
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 43. AC97 Status Data Register (MBAR + PSCx + 0x2C)—AC97Data

Table 35. AC97 Status Data Field Descriptions

Bit	Name	Description
0	—	Reserved
1:7	AC97 Register Index Echo	**Enhanced AC97 mode**—AC97 register index echo This register contains the received register index echo from the RX Slot0. **Other modes**—Reserved
6:23	AC97 Control Register ReadData	**Enhanced AC97 mode**—AC97 control register read data This register contains the received control data from Rx Slot2. **Other modes**—Reserved
24:31	—	Reserved

The BitClkDiv[0:7] field of the Codec clock register has been moved from CCR bits [8:15] to CCR bits [16:23]. CCR bits [8:15] are now BitClkDiv[8:15].

If the CODEC mode is used, software written for prior versions of the MPC5200 will need to be modified.

3.7.4 Codec Clock Register (MBAR + PSCx + 0x20)—CCR

This register defines the divider for the Frame and BitClk generation for Codec mode. This register value only has an effect if the GenClk bit in the PSC control register SICR is set to one. In UART, SIR, and AC97 mode, this register is reserved.

3.7.4.1 Codec Clock Register (MBAR + PSCx + 0x20)—CCR for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	FrameSyncDiv[0:7]								BitClkDiv[8:15]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	msb 16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	BitClkDiv[0:7]								Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for Codec Mode for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved								BitClkDiv[8:15]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	msb 16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	BitClkDiv[0:7]								Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for MIR/FIR Mode for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	msb 16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 46. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for Other Modes for MPC5200B

Table 36. Codec Clock Field Descriptions

Bit	Name	Description
0:7	FrameSyncDiv	**Codec**—frame sync divider FrameSync is generated internally by dividing down the Bit Clock. The FrameSyncDiv defines the number of Bit clock cycles between two active frame edges: FrameSync Length = FrameSyncDiv[0:7] + 1 **Codec / SPI**—delay before SCK (DSCKL) When the PSC is in SPI mode (SICR[SPI] = 1), the FrameSyncDiv divider is used to determine the length of time the PSC delays after SS goes low/active before the first SCK transition of the serial transfer. This is a feature that exists in a QSPI. The following equation determines the actual delay before SCK: **Other modes**—Reserved **Note:** The value 0x00 stops this counter and disables the clock generator.
8:23	BitClkDiv	**Codec**—bit clock divider Bit clock is generated internally by dividing down the Mclk frequency as follows: **Codec SPI**—baud rate SCK is generated internally by dividing down the Mclk frequency as follows: **MIR / FIR**—Irda clock IrdaClk is generated internally by dividing down the Mclk frequency as follows: **Other modes**—Reserved **Note:** The value 0x00 stops this counter and disables the clock generator.

3.7.4.2 Codec Clock Register (MBAR + PSCx + 0x20)—CCR for Codec Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	FrameSyncDiv[0:7]								BitClkDiv[0:7]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 47. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for Codec Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved								BitClkDiv[0:7]							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 48. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for MIR/FIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 49. Codec Clock Register (MBAR + PSCx + 0x20)—CCR for Other Modes for Prior Versions of the MPC5200

Table 37. Codec Clock Field Descriptions

Bit	Name	Description
0:7	FrameSyncDiv	**Codec**—frame sync divider Frame sync is generated internally by dividing down the Bit Clock frequency as follows: FrameSync frequency = BitClk / (FrameSyncDiv[0:7] +1) **Codec / SPI**—delay before SCK (DSCKL) When the PSC is in SPI mode (SICR[SPI] = 1), the FrameSyncDiv divider is used to determine the length of time the PSC delays after SS goes low/active before the first SCK transition of the serial transfer. This is a feature that exists in a QSPI. The following equation determines the actual delay before SCK: **Other modes**—Reserved **Note:** The value 0x00 stops this counter and disables the clock generator.
8:15	BitClkDiv	**Codec**—bit clock divider Bit clock is generated internally by dividing down the Mclk frequency as follows: **Codec SPI**—baud rate SCK is generated internally by dividing down the Mclk frequency as follows: **MIR / FIR**—Irda clock IrdaClk is generated internally by dividing down the Mclk frequency as follows: **Other modes**—Reserved **Note:** The value 0x00 stops this counter and disables the clock generator.

3.7.5 More Clock Dividers for UART Baud Rate Generation

A value of 0xF in the TCS or RCS field of the CSR (PSC_MBAR + 0x04) will select a divide by 4 prescaler.

A value of 0xE in the TCS or RCS field of the CSR (PSC_MBAR + 0x04) will disable clock generation.

In the UART mode, the TCS and RCS fields can now be programmed to %1110 (disable the clock generation) and %1111 (choose the divide by 4 prescaler for UART receive clock generation). In the SIR mode, the TCS and RCS fields can now be programmed to %1110 (disable the clock generation). These codes are invalid in versions of the MPC5200 prior to the MPC5200B.

As these codes were not used in versions of the MPC5200 prior to the MPC5200B, existing user software should be compatible with the MPC5200B.

3.7.6 Clock Select Register (MBAR + PSCx + 0x04)

3.7.6.1 Clock Select Register for UART / SIR Mode for MPC5200B0

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R																
W	RCS				TCS				Reserved							
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 50. Clock Select Register (MBAR + PSCx + 0x04) for UART / SIR Mode for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51. Clock Select Register (MBAR + PSCx + 0x04) for Other Modes for MPC5200B

Table 38. Clock Select Field Descriptions

Bit	Name	Description
0:3	RCS	**UART**—receiver clock select register 0000 -1101 choose the prescaler by 32 for the UART receive clock generation 1110 disable the clock generation 1111 choose the prescaler by 4 for the UART receive clock generation **SIR** —clock select register 1110 disable the clock generation others choose the prescaler by 32 for the SIR receive clock generation **Other modes**—Reserved

Table 38. Clock Select Field Descriptions (continued)

Bit	Name	Description
4:7	TCS	**UART**—transmitter clock select register 0000 -1101 choose the prescaler by 32 for the UART transmit clock generation 1110 disable the clock generation 1111 choose the prescaler by 4 for the UART transmit clock generation **SIR** —clock select register 1110 disable the clock generation others choose the prescaler by 32 for the SIR transmit clock generation **Other modes**—Reserved
8:15	—	Reserved

3.7.6.2 Clock Select Register for UART / SIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R																
W	RCS				TCS				Reserved							
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52. Clock Select Register (MBAR + PSCx + 0x04) for UART / SIR Mode for Prior Versions of the MPC5200

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 53. Clock Select Register (MBAR + PSCx + 0x04) for Other Modes for Prior Versions of the MPC5200

Table 39. Clock Select Field Descriptions

Bit	Name	Description
0:3	RCS	**UART / SIR** —receiver clock enable 0000 -1101 enable the clock generation 1110 -1111 disable the clock generation **Other modes**—Reserved
4:7	TCS	**UART / SIR** —transmitter clock enable 0000 -1101 enable the clock generation 1110 -1111 disable the clock generation **Other modes**—Reserved
8:15	—	Reserved

3.8 I²C

3.8.1 I²C Frequency Divider Register (MFDR) MBAR + 0x3D04

Bits 0 and 1 were added as additional prescaler bits. These bits have been added to increase the length of the length of the I²C prescaler. The default state of these bits from the release of RESET is %00, which is compatible with prior versions of the MPC5200. The user software should be checked to see if the state of these bits is changed. If not, the user code for prior versions of the MPC5200 should be compatible with the MPC5200B.

3.8.2 I²C Frequency Divider Register (MFDR)—MBAR + 0x3D04 / 0x3D44

3.8.2.1 I²C Frequency Divider Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	FDR[7:0]								Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 54. ²C Frequency Divider Register for MPC5200B

Table 40. I²C Frequency Divider Field Descriptions for MPC5200B

Bit	Name	Description
0:1	FDR[7:6]	These 2 bits act as a prescale divider of the input module clock.

3.8.2.2 II^{2}C Frequency Divider Register for Prior Versions of the MPC5200

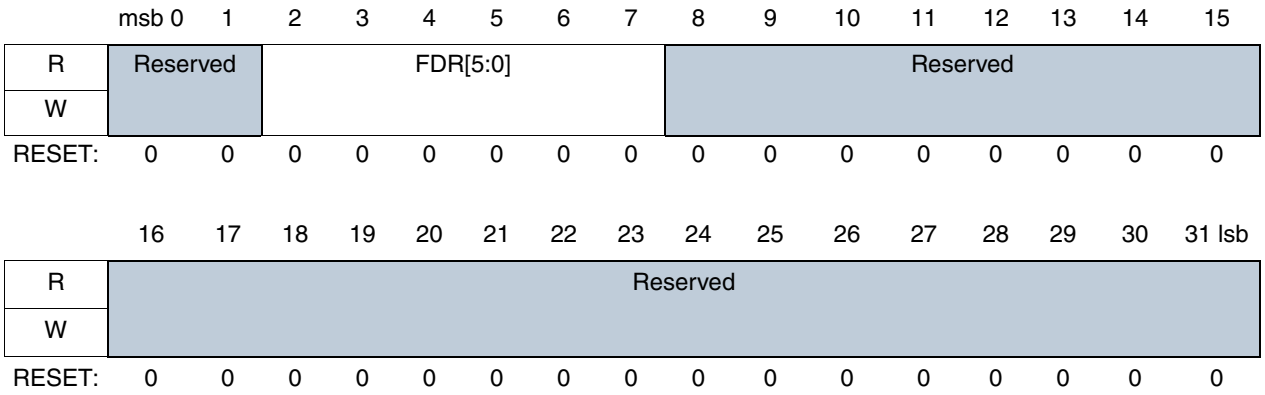

Figure 55. ^{2}C Frequency Divider Register for Prior Versions of the MPC5200

Table 41. I^{2}C Frequency Divider Field Descriptions for Prior Versions of the MPC5200

Bit	Name	Description
0:1	—	Reserved

3.8.3 I2C Filter Register (IFR) MBAR + 0x3D24—New Register

Bits [0:7] control the configuration of the I^{2}C glitch filters. These bits add the functionality to the MPC5200B of programming a filter to reject noise pulses on the I^{2}C input. Pulses up to 15 IP bus clock cycles wide can be rejected. The default state of these bits from the release of RESET is %00000000, which is compatible with prior versions of the MPC5200.The user software should be checked to see if the state of these bits is changed. If not, the user code for prior versions of the MPC5200 should be compatible with the MPC5200B.

3.8.3.1 I²C Filter Register for MPC5200BI

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	Resv	Resv	Resv	Resv	FR3	FR2	FR1	FR0	Reserved							
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56. I²C Filter Register for MPC5200B

Table 42. I²C Filter Register Field Descriptions

Bit	Name	Description
0:3	—	Reserved
4:7	FR[7:4]	Bits 7 to 4 contain the programming controls for the width of glitch (in terms of IPBUS clock cycles) that the filter should absorb; that is, the filter will not let pass glitches less than or equal to this width setting. FR[] 3210 0000 No Filter / Bypass 0001 Filter glitches up to width of 1 IPBUS clock cycle 0010 Filter glitches up to width of 2 IPBUS clock cycles 0011 Filter glitches up to width of 3 IPBUS clock cycles 0100 Filter glitches up to width of 4 IPBUS clock cycles 0101 Filter glitches up to width of 5 IPBUS clock cycles 0110 Filter glitches up to width of 6 IPBUS clock cycles 0111 Filter glitches up to width of 7 IPBUS clock cycles 1000 Filter glitches up to width of 8 IPBUS clock cycles 1001 Filter glitches up to width of 9 IPBUS clock cycles 1010 Filter glitches up to width of 10 IPBUS clock cycles 1011 Filter glitches up to width of 11 IPBUS clock cycles 1100 Filter glitches up to width of 12 IPBUS clock cycles 1101 Filter glitches up to width of 13 IPBUS clock cycles 1110 Filter glitches up to width of 14 IPBUS clock cycles 1111 Filter glitches up to width of 15 IPBUS clock cycles
8:31	—	Reserved

3.8.4 I2S/GPIO

Additional functionality has been added to the MPC5200B. These changes allow the pins on the I²C port to function as general purpose output.

3.8.5 GPIO Output-Only Enables Register MBAR + 0x0B18

These bits have been added to enable the individual pins on the I^2C port as general purpose outputs. The default state is a logic 0. This leaves the pins under the control of the I^2C port configuration bits.

The user software should be checked to see if the state of these bits is changed. If not, the user code for prior versions of the MPC5200 should be compatible with the MPC5200B.

3.8.5.1 GPS GPIO Output-Only Enables Register for MPC5200B

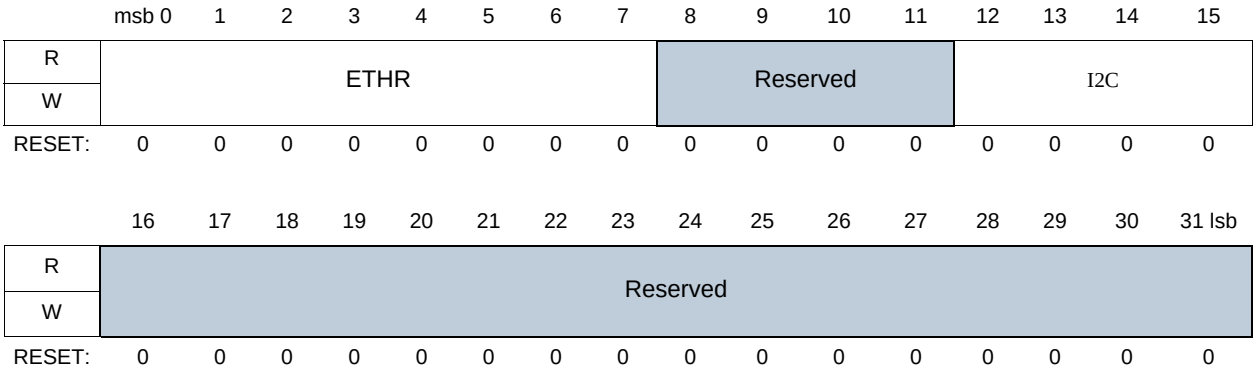

Figure 57. GPS GPIO Output-Only Enables Register for MPC5200B

Table 43. GPS GPIO Output-Only Enables Field Descriptions

Bit	Name	Description
0:7	ETHR	Individual bits to enable each output only GPIO pin—all reside on the Ethernet port. bit 0 controls GPIO_ETHO_7 (ETH_7 pin) bit 1 controls GPIO_ETHO_6 (ETH_6 pin) bit 2 controls GPIO_ETHO_5 (ETH_5 pin) bit 3 controls GPIO_ETHO_4 (ETH_4 pin) bit 4 controls GPIO_ETHO_3 (ETH_3 pin) bit 5 controls GPIO_ETHO_2 (ETH_2 pin) bit 6 controls GPIO_ETHO_1 (ETH_1 pin) bit 7 controls GPIO_ETHO_0 (ETH_0 pin) 0 Disabled for GPIO use (default) 1 Enabled for GPIO use
8 :11	—	Reserved
12:15	I2C	Individual bits to enable each output only GPIO pin—all reside on the I2C ports. bit 12 controls I2C2_CLK (I2C_2 pin) bit 13 controls I2C2_IO (I2C_3 pin) bit 14 controls I2C1_CLK (I2C_0 pin) bit 15 controls I2C1_IO (I2C_1 pin) 0 Disabled for GPIO use (default) 1 Enabled for GPIO use This bits can be used to toggle the clock (SCL) and data (SDA) lines of the I2C interface.
16:31	—	Reserved

3.8.6 GPIO Output-Only Data Value Out Register MBAR + 0x0B1C

If these bits are enabled as general purpose outputs, the logic value applied to the respective pins can be read using this register.

3.8.6.1 GPS GPIO Output-Only Data Value Out Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	ETHR								Reserved				I2C			
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	Reserved															
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 58. GPS GPIO Output-Only Data Value Out Register for MPC5200B

Table 44. GPS GPIO Output-Only Data Value Out Field Descriptions

Bit	Name	Description
0:7	ETHR	Individual bits to control the state of enabled output only GPIO pins. bit 0 controls GPIO_ETHO_7 (ETH_7 pin) bit 1 controls GPIO_ETHO_6 (ETH_6 pin) bit 2 controls GPIO_ETHO_5 (ETH_5 pin) bit 3 controls GPIO_ETHO_4 (ETH_4 pin) bit 4 controls GPIO_ETHO_3 (ETH_3 pin) bit 5 controls GPIO_ETHO_2 (ETH_2 pin) bit 6 controls GPIO_ETHO_1 (ETH_1 pin) bit 7 controls GPIO_ETHO_0 (ETH_0 pin) 0 Drive 0 on the pin (default) 1 Drive 1 on the pin
8 :11	—	Reserved
12:15	I2C	Individual bits to control the state of enabled output only GPIO pins — all reside on the I2C ports. bit 12 controls I2C2_CLK (I2C_2 pin) bit 13 controls I2C2_IO (I2C_3 pin) bit 14 controls I2C1_CLK (I2C_0 pin) bit 15 controls I2C1_IO (I2C_1 pin) 0 Drive 0 on the pin (default) 1 Drive 1 on the pin This bits can be used to toggle the clock (SCL) and data (SDA) lines of the I2C interface.
16:31	—	Reserved

3.9 LPC/GPIO

- GPS port configuration register MBAR + 0x0B00
- Bit 1—LPTZ was added to the GPS port configuration register

This bit was added to change the functionality of certain pins on the LocalPlus bus when operating in the non-muxed mode (LPC). If this bit is left as a logic 0 (default condition), the pins will retain their original functionality and no changes in software will be required.

When this bit is at a logic 0, the GPIO_WKUP_7 and TEST_SEL_1 pins retain the functionality of the prior versions of the MPC5200. When this bit is set, the functionality of GPIO_WKUP_7 is switched to TSIZ1 and the functionality of TEST_SEL_1 is switched to TSIZ2.

3.9.1 GPS Port Configuration Register—MBAR + 0x0B00

3.9.1.1 GPS Port Configuration Register for MPC5200B

	msb 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
R	CS1	LPTZ	ALTs		CS7	CS6	ATA		IR_USB_CLK	IRDA			Ether			
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31 lsb
R	PCI_DIS	USB_SE	USB		PSC3				Rsvd	PSC2			Rsvd	PSC1		
W																
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 59. GPS Port Configuration Register for MPC5200B

Table 45. GPS Port Configuration Field Descriptions

Bit	Name	Description
1	LPTZ	LocalPlus non-muxed TSIZ bit 0 gpio_wkup_7 and test_sel_1 1 TSIZ 1 on gpio_wkup_7 and TSIZ 2 on test_sel_1

3.9.1.2 GPS Port Configuration Register for Prior Versions of the MPC5200

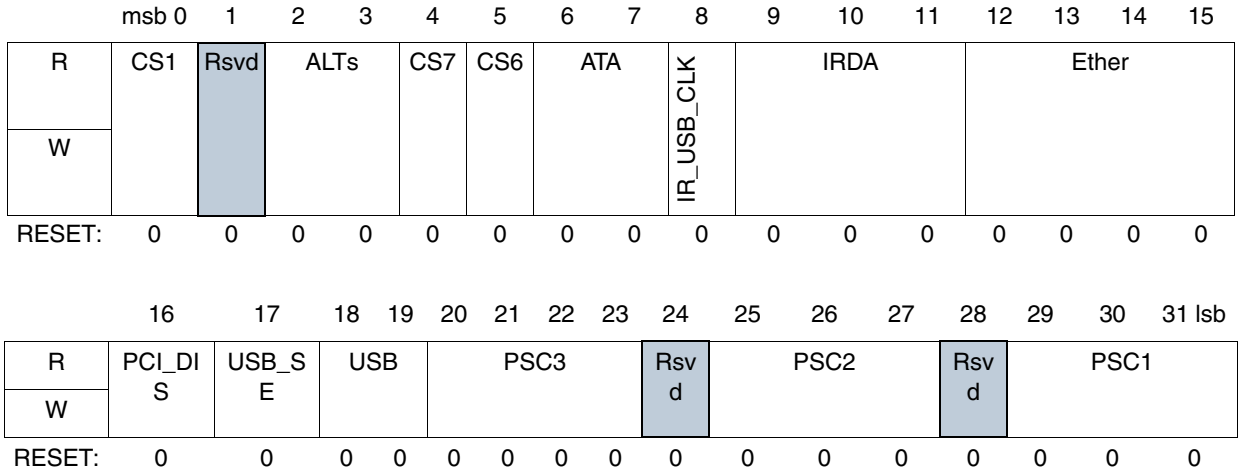

Figure 60. GPS Port Configuration Register for Prior Versions of the MPC5200

Table 46. GPS Port Configuration Field Descriptions

Bit	Name	Description
1	—	Reserved

For software from prior versions of the MPC5200, this bit should be checked to see that it remains at a logic 0 for compatibility with the MPC5200B.

3.10 MSCAN

- MSCAN 1 control register 1 (CANCTL1)MBAR + 0x0901
- MSCAN 2 control register 1 (CANCTL1)MBAR + 0x0981
- Bit 1—CLKSRC of CANCTL1

In prior versions of the MPC5200, the same clock source had to be used for both MSCAN modules. This requirement has changed. Now, the clock source for each MSCAN module can be independently selected.

The user software should be checked to make sure that the CLKSRC bit is set appropriately in each MSCAN module's control register 1 to select the desired MSCAN clock source.

3.10.1 MSCAN Control Register 1 (CANCTL1)—MBAR + 0x0901 / 0x981

3.10.1.1 MSCAN Control Register 1 for MPC5200B

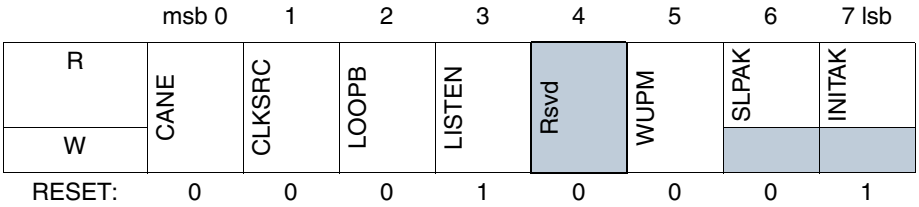

Figure 61. MSCAN Control Register 1 for MPC5200B

Table 47. MSCAN Control Field Descriptions

Bit	Name	Description
1	CLKSRC	MSCAN clock source—bit defines MSCAN module clock source (only for systems with a system clock generation module. 0 MSCAN clock source is the IP bus clock (IP CLK) 1 MSCAN clock source is the oscillator clock (SYS_XTAL_IN) **Note:** The two MSCAN modules can have different selected clock sources.

3.10.1.2 MSCAN Control Register 1 for Prior Versions of the MPC5200B

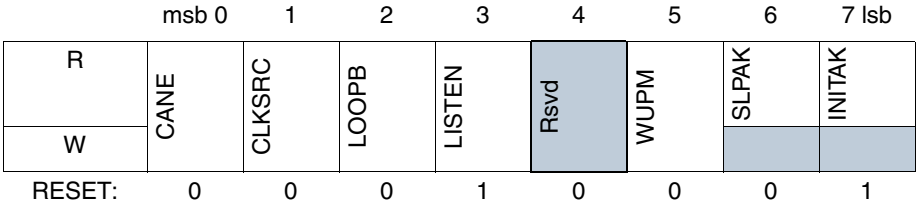

Figure 62. MSCAN Control Register 1 for Prior Versions of the MPC5200B

Table 48. MSCAN Control Field Descriptions

Bit	Name	Description
1	CLKSRC	MSCAN clock source—bit defines MSCAN module clock source (only for systems with a system clock generation module. 0 MSCAN clock source is the IP bus clock (IP CLK) 1 MSCAN clock source is the oscillator clock (SYS_XTAL_IN) **Note:** Both MSCAN modules can have only the same selected clock source.To select the oscillator clock the CLKSRC bit in the CANCTL1 register must be set in MSCAN1 OR/AND in MSCAN2.

3.11 ID Codes

The MPC5200B has different ID codes from prior versions of the MPC5200. The ID codes for MPC5200B are presented in Table 49.

Table 49. ID Codes

ID Code	MPC5200	MPC5200B
PVR_COP	8082 2011	8082 2014 --- (G2_LE rev1.4)
SVR_COP	8011 0012 --- (rev1.2)	8011 0020 --- (1st samples) 8011 0022 --- (prod. samples)
XLBArb VER	0001	0001
PCI Device ID	5803 1057	5809 1057
JTAG IDCODE	0001 101d	1001 101d

4 Documentation

The following documents and models are available.

- MPC5200B *User Manual*
- “MPC5200B Data Sheet”
- “MPC5200B Errata”
- MPC5200B IBIS Model
- MPC5200B BSDL File

THIS PAGE INTENTIONALLY 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

AN3045
Rev. 3, 5/2006

freescale™
semiconductor