Freescale Semiconductor
Application Note

Document Number: AN2682
Rev. 1, 01/2007

PowerQUICC™ II Parity and ECC Capability

by DSD Applications,
Freescale Semiconductor, Inc.
Austin, TX

Ensuring the integrity of data stored in the memory is an important aspect of memory design. Two primary means of accomplishing this task are parity and error correction code (ECC). Historically, parity has been the most commonly used data integrity method. Parity can detect, but not correct, single-bit errors. ECC is a more comprehensive method of data integrity checking that can detect and correct single-bit error and detect double-bit error. This application note describes the PowerQUICC™ II data error protection mechanism. The devices listed in Table 1 are collectively called PowerQUICC II throughout this document.

Contents

Table 1. PowerQUICC II Families and Devices

MPC8260 Family		MPC8280 Family
0.29µm (HiP3)	**0.25µm (HiP4)**	**0.13µm (HiP7)**
MPC8260 MPC8255	MPC8260A MPC8255A MPC8250 MPC8264 MPC8265 MPC8266	MPC8280 MPC8275 MPC8270

1 Basics of PowerQUICC Parity Checking

The PowerQUICC II has parity checking for both the 60x bus and the local bus, which operate concurrently and independently. Figure 1 shows the overall picture of the parity checking. For .29 µm (HiP3) devices, DPRAM does not support any parity. For .25 µm (HiP4) and .13 µm (HiP7) devices, some parity capabilities are added. See Section 5, "Parity for the 60x External Master Access."

When the PowerQUICC II does a memory write access, it generates parity/ECC. Parity is stored in the external memory along with the data. For such accesses, the PowerQUICC II does not do the parity comparison and never generates parity errors.

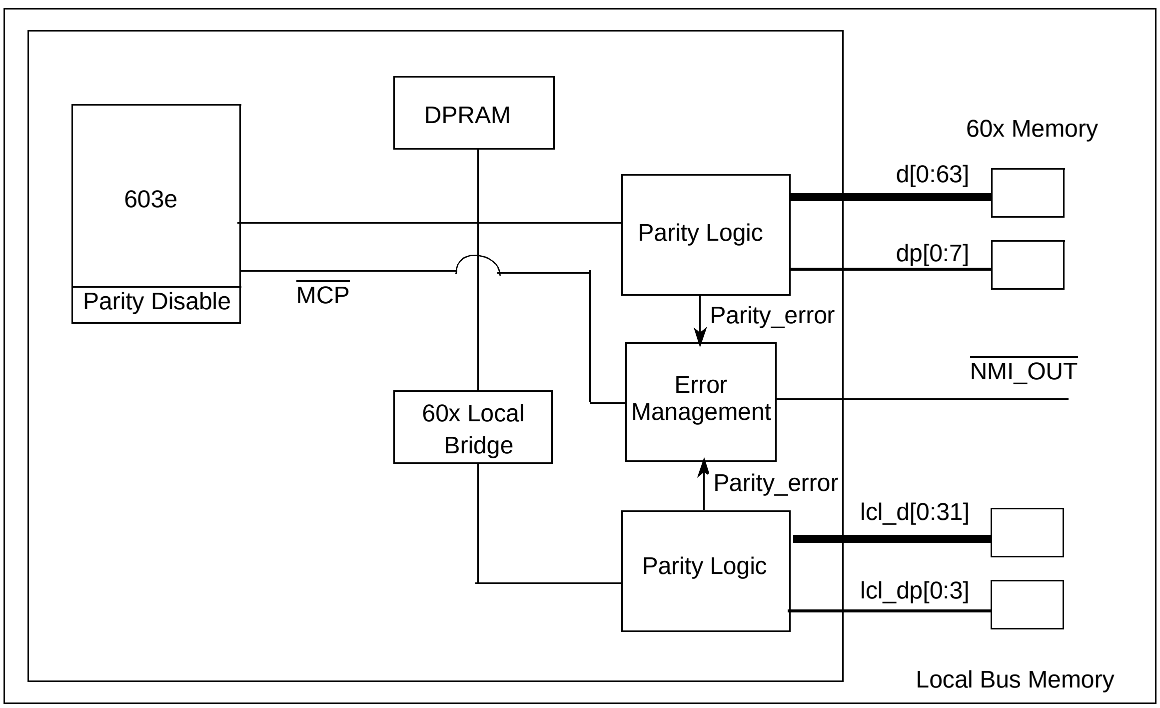

Figure 1. PowerQUICC II Parity Diagram

When the PowerQUICC II performs a memory read access, the external memory drives both data and parity. The PowerQUICC II parity logic calculates the parity based on the data and compares with the parity lines provided by the memory. Any mismatch causes a parity error. The PowerQUICC II reports the error as programmed.

The PowerQUICC II supports three types of parity: normal, read-modify-write, and ECC. The parity is configured on a per-bank (chip select) basis, and BRx[DECC] determines it. Each bank can have a different type of parity or can disable the parity. For the normal parity or read-modify-write parity, users can choose between even or odd parity with BCR[EPAR] for 60x bus and BCR[LEPAR] for the local bus. These two bits are global and apply to all banks on the 60x bus and the local bus. The ECC algorithm is inherently for 64-bit wide data only. The memory bank must be 64 bits wide to use ECC for that bank.

The parity is checked and generated in the memory controller. Note that the 603e core has its own separate parity logic that must be disabled using HID0[EBD] (default is disabled). The parity error is reported to the 603e core using the internal $\overline{\text{MCP}}$ signal. $\overline{\text{MCP}}$ is logically ANDed with HID0[EMCP]. To make $\overline{\text{MCP}}$ take effect, both HID0[EMCP] and MSR[ME] must be set. If the 603e core is disabled, the internal $\overline{\text{MCP}}$ is automatically routed out to the $\overline{\text{NMI\_OUT}}$ pin. In the core disable mode, $\overline{\text{NMI\_OUT}}$ should be connected to the $\overline{\text{MCP}}$ of the host processor. The memory controller asserts $\overline{\text{MCP}}$ in the following cases:

- Parity error
- ECC double-bit error
- ECC single-bit error when the maximum number of the single-bit errors is reached

The 60x bus data parity pins, dp[0:7], are multiplexed with other functionality. They can be configured during hard reset to function as the parity pins by setting HRCW[10:11] in the hard reset configuration word to 01. HRCW[DPPC], data parity pin configuration, is latched into SIUMCR[4:5] during the end of the hard reset process. The other way to configure the parity pins is to program SIUMCR[4:5] to 01 directly.

The local bus has the following dedicated parity pins:

- SIUMCR[PBSE], parity byte select enable. If PBSE is set, PGTA/PUPWAIT/PGPL4/PPBS functions as the 60x bus parity byte select.
- SIUMCR[LPBSE], local parity byte select enable. If LPBSE is set, it configures LGTA/LUPWAIT/LGPL4/LPBS as the local bus parity byte select.

The registers that are closely related to the parity function are TESCR1 and TESCR2 for 60x bus and L_TESCR1, L_TESCR2 for the local bus.

- TESCR1[DMD], data error disable. If set, all data errors (parity and single and double ECC errors) on the 60x bus are disabled.
- TESCR1[PAR] indicates a parity error (either normal or read-modify-write). TESCR1[ECC2] indicates a double-bit ECC error. TESCR1[ECC1] indicates that a single-bit error and error counter exceeds the maximum value.

If DMD is set, all the error bits are not set. If any one of the PAR, ECC2, or ECC1 bits is set, the value of DMD does not matter, and it causes the internal $\overline{\text{MCP}}$ assertion or $\overline{\text{NMI\_OUT}}$ if the core is disabled.

TESCR1[ECNT] indicates the number of the single-bit errors. Each time a single-bit ECC error occurs, the error is corrected but ECNT increments by 1. When it reaches 255, $\overline{\text{MCP}}$ is asserted for all single-bit errors thereafter. The user can write a starting count number to this field. The counter starts from this value instead of zero. Fewer ECC single-bit errors are needed to trigger the $\overline{\text{MCP}}$.

This feature gives the system ability to withstand a few random errors but react to catastrophic failure.

	0	1	2	3	4	5	6	7–9	10	11–15
Field	BM	ISBE	PAR	ECC2	ECC1	WP	EXT	TC	—	TT
Reset	0000_0000_0000_0000									
R/W	R/W									
Addr	0x0x10040									

	16	17	18	19	20	21	22	23	24–31
Field	—	DMD	—	PCIMCP [1]	DER [2]	IRQ0 [2]	SWD [2]	ADO [2]	ECNT
Reset	0000_0000_0000_0000								
R/W	R/W								
Addr	0x10042								

[1] MPC8250, MPC8265, and MPC8266 only. Reserved on all other devices.

[2] Reserved on .29µm (HiP3) Rev A.1 devices.

Note: Bits 0–15 and 19–23 are status bits and are cleared by writing 1s.

Figure 2. 60X Bus Transfer Error Status and Control Register 1 (TESCR1)

TESCR2[PB] indicates which byte lane has a parity error, 1 per 8-bit lane. TESCR2[BNK] indicates which memory bank has an error. That information is useful during the debugging process. L_TESCR1 and L_TESCR2 have the same fields for the local bus.

A potential timing problem occurs when ECC or parity is used. Memory such as SDRAM can output data every cycle. The parity checking requires additional data setup time, and the timing constraints can be very tight. In such a system, the users can set BRx[DR], creating a data pipelining of one stage. Data is latched first and the parity is checked the next cycle, eliminating the additional data setup time requirement.

2 Read-Modify-Write

To support the normal parity, a special type of memory called the parity memory is needed. The parity memory has an extra bit associated with every byte. The extra parity bit is written and read along with the corresponding data byte. Figure 3 shows the organization for 32-bit wide parity memory. Both d[0:7] and dp[0] are addressed by $\overline{BS}$[0]. For a write with the size less than 32 bits, the $\overline{BS}$[0:3] is used to control which byte is written. For example, if it is a 1-byte write to address 0, $\overline{BS}$[0:3] is 0x0111. Then only d[0:7] and dp[0] are written, and d[8:31] and dp[1:3] are masked and untouched. This operation is, of course, the correct one.

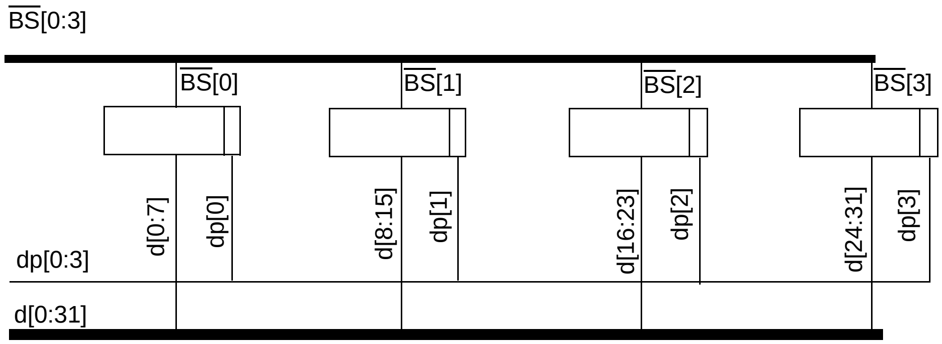

Figure 3. Organization of 32-bit Parity Memory

The low volume causes the parity memory to be significantly more expansive than the normal memory. If cost is an issue, read-modify-write (RMW) parity should be used. The goal is to avoid the expensive parity memory and use the regular non-parity memory for both the data and the parity. First, analyze what happens if data and parity are connected to two separate memories as in Figure 4 and program the memory bank as the normal parity.

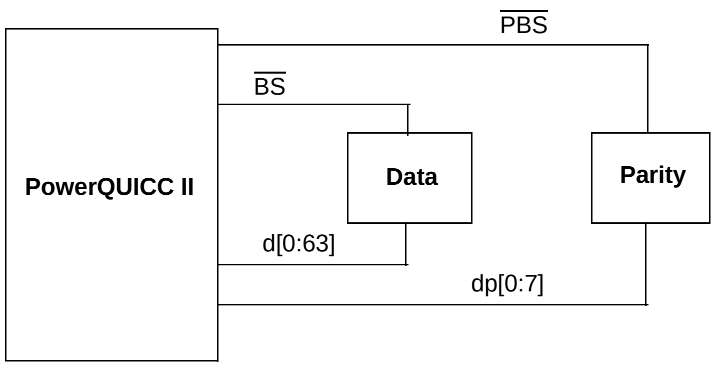

Figure 4. Regular Memory for Both Data and Parity

Note that there is only a 1-byte select signal for the parity memory. For a write with a size less than the port size, for instance, write 1 byte to address 0, then $\overline{BS}$[0:7] is 0x01111111. If we use the $\overline{PBS}$ for the parity byte select, notice that $\overline{PBS}$ is logic OR of all the $\overline{BS}$. If $\overline{BS}$[0] is active, $\overline{PBS}$ is active also. On the data bus, d[0:7] is valid while d[8:63] has invalid data since we are writing 1 byte. The corresponding parity dp[1:7] is generated from the invalid data and therefore invalid as well. During the write, because $\overline{BS}$[1:7] are high, the invalid d[8:63] is not written to the memory. However, for the parity, the invalid dp[1:7] is written along with the valid dp[0]. A write to less than port size corrupts the parity.

The problem that arises from the normal parity and regular non-parity memory is clear. It is easy to understand why a RMW is necessary to overcome the problem. If the bank is programmed as RMW and the write size is less than the port size, then for a write the memory controller does a port size read first, followed by a port size write. Using the write 1 byte to address 0 as the example, the PowerQUICC II first reads in the data at address 0, 1,..., 7. During the write, the data that appears on the data bus is a combination of the intended new write data d[0:7] and the original data d[8:63] that PowerQUICC II just reads back. Therefore, all data on the data bus is valid and correct parity for d[8:63] can be calculated and written back to the parity memory, avoiding the parity corruption problem. With RMW, the benefit is that only regular memory is used and the cost is lower. The trade-off is that for every write less than the port size, the PowerQUICC II must do an extra read, which takes more cycles.

3 Initialization

This section describes the initialization for the ECC and the read-modify-write memories.

3.1 ECC Initialization

The ECC memory bank must be initialized before it is used. This section discusses what happens if the CPU accesses the uninitialized ECC memory bank. After power-on reset, the data and ECC memory contains 'gabbed'. The CPU starts to store the valid data. Normally, the size write is 32 bits or less. Because the bank is ECC-enabled, the port size must be 64 bits. A write of less than 64 bits automatically triggers a read-modify-write sequence because the ECC algorithm is inherently for 64 bits. The PowerQUICC II must read back the 64-bit data and combine the new write data with the original data to form the new data and calculate new ECC parity based on that information. However, during the readback, because memory is uninitialized, the ECC parity mostly does not match the data, triggering undesired ECC errors. To initialize the ECC memory without undesired ECC errors, set TESCR1[DMD] to disable the data error report mechanism. Then initialize all the ECC memory space. After initialization, clear the DMD bit for the normal operation.

3.2 Read-Modify-Write Initialization

Like ECC, the RMW must be initialized before use. Set TESCR1[DMD] and initialize the whole memory bank. Then clear TESCR1[DMD] for the normal operation. Use LTESCR1[DMD] for the local bus.

3.3 Normal Parity

Normal parity does not require initializing.

4 Testing the System Response to the Parity Error

For normal and read-modify-write parity, write the data with BCR[EPAR] or BCR[LEPAR] for the local bus and then invert the EPAR or LEPAR to read back and guarantee that the parity is wrong. System response to the parity error can be tested. However, this method cannot be used for ECC. The PowerQUICC II does not have a built-in mechanism to inject errors and test the system response for ECC. The following procedure is one of the ways to work around this problem.

Use $\overline{\text{PBS}}$ to control the ECC memory byte select signal. Disable the $\overline{\text{PBS}}$ by clearing SIUMCR[PBSE]. Then modify the memory data by one bit or multiple bit. A new ECC parity is generated for the new data. Because $\overline{\text{PBS}}$ is disabled, however, it is not written to the parity memory. Data is modified without the parity being changed accordingly. Enable $\overline{\text{PBS}}$ again for the normal operation, and a read to the modified memory should cause a single-bit or multiple-bit error.

5 Parity for the 60x External Master Access

The following sections discuss parity for the 60x external master access.

5.1 Access to 60x Memory

Parity is disabled.

5.2 Access to DPRAM or the Internal Registers

For .29 mm (HiP3) devices, when an external master reads from an internal memory space (DPR, registers), the PowerQUICC II generates the parity for that transaction. When an external master writes to the internal memory space, parity is not checked.

For .25 mm (HiP4) and .13 mm (HiP7) devices, when an external master reads from an internal memory space (DPR, registers), PowerQUICC II generates the parity for that transaction. When an external master writes into an internal space and BCR[SPAR] = 1, the PowerQUICC II checks normal parity for this transaction. When BCR[SPAR] = 0, parity is not checked.

5.3 Access to Local Bus Memory

For .29 mm (HiP3) devices, when the external master initiates a memory read from the 60x bus that is mapped to the local bus, if the local bus parity is enabled for that bank, the parity is checked on the local bus. When the data read back from the local bus is driven on the 60x for the external master, the parity is generated and driven on to the pins. When the external master initiates a memory write from the 60x bus that is mapped to the local bus and if the local bus parity is enabled for that bank, the local bus generates the parity and drives the local bus parity pins.

For .25 mm (HiP4) and .13 mm (HiP7) devices, When the external master initiates a memory read from the 60x bus that is mapped to the local bus and if the local bus parity is enabled for that bank, the parity is checked on the local bus. When the data read back from the local bus is driven on to the 60x for the external master, the parity is generated by the 60x parity logic and driven on to the parity pins regardless of BCR[SPAR]. Even if the local bus has parity error, the 60x bus still generates the parity based on the data. The parity on the 60x bus has nothing to do with the parity on the local bus.

When the external master initiates a memory write from 60x bus that is mapped to the local bus and if the local bus parity is enabled for that bank, then the local bus generates the parity and drives it on to the local bus parity pins. If the BCR[SPAR]=1, the 60x bus also checks the parity. The parity logic on the 60x bus and the local bus are independent. If parity error is on the 60x bus, the 60x reports the parity error but the local bus still generates the corresponding parity based the data.

6 ECC Encoding

The ECC syndrome equations for the ECC are shown in Figure 5 and Figure 6. The ECC parity bit is equal to the exclusive or of all the data bits with x (see Example 1, in which ^ is exclusive or).

Example 1.

```
ecc_dp[0] = di[0] ^ di[1] ^ di[2] ^ di[3] ^ di[4] ^ di[5] ^ di[6] ^ di[7]
          ^ di[8] ^ di[9] ^ di[10] ^ di[11] ^ di[12] ^ di[13] ^ di[14] ^ di[15]
          ^ di[19] ^ di[23] ^ di[27] ^ di[31] ^ di[34] ^ di[38] ^ di[42] ^ di[46]
          ^ di[51] ^ di[55] ^ di[59] ^ di[61];
```

Syndrome Bit	Data Bit																														
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x				x				x				x				x
1	x				x				x				x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2		x				x				x				x			x				x				x				x			
3			x				x				x				x			x				x				x				x		
4				x				x				x				x			x	x			x	x			x	x			x	x
5					x	x	x	x					x	x	x	x					x	x	x	x					x	x	x	x
6									x	x	x	x	x	x	x	x									x	x	x	x	x	x	x	x
7	x	x	x	x									x	x	x	x	x	x	x					x				x	x	x	x	

Figure 5. ECC Syndrome Encoding for Data Bits [0:31]

Syndrome Bit	Data Bit																														
	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
0			x				x				x				x					x				x				x		x		
1				x				x				x				x	x				x				x						x	
2	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x				x				x						x
3	x				x				x				x						x				x				x		x	x	x	x
4		x	x	x		x	x	x		x	x	x		x	x	x													x	x	x	x
5					x	x	x	x					x	x	x	x	x	x	x	x	x	x	x	x								
6									x	x	x	x	x	x	x	x	x	x	x	x					x	x	x	x	x	x	x	x
7	x	x					x	x			x	x	x	x							x	x	x	x	x	x	x	x		x	x	x

Figure 6. ECC Syndrome Encoding for Data Bits [32:63]

7 Revision History

Table 2 provides a revision history for this application note.

Table 2. Document Revision History

Rev. No.	Substantive Change(s)
0	Initial release
1	Non-substantive formatting.

THIS PAGE INTENTIONALLY LEFT BLANK

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

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)
www.freescale.com/support

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
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN2682
Rev. 1
01/2007