Application Report
SPRACE2–January 2018

TMS320C66x XMC Memory Protection

Stanley Liu

ABSTRACT

The C66x DSP inside TDA2x/TDA3x includes memory protection architecture providing the functionalities of data protection against poorly behaving code, preventing illegal memory accesses, and enforcing defined boundaries between supervisor and user mode. In the automotive processor, this can be utilized to create a robust system design. This application report describes how to utilize the XMC memory protection unit on the C66x DSP.

Contents

List of Figures

List of Tables

Trademarks

Code Composer Studio is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

1 Introduction

The C66x DSP offers memory protection support for its local memories (L1P/L1D/L2) and external memory access. The EXtended Memory Controller (XMC) provides memory protection for address outside C66x CorePac. The memory protection scheme is also designed to coordinate with L1/L2 Memory Protection Units and L3/L4 Interconnect Firewalls in the system.

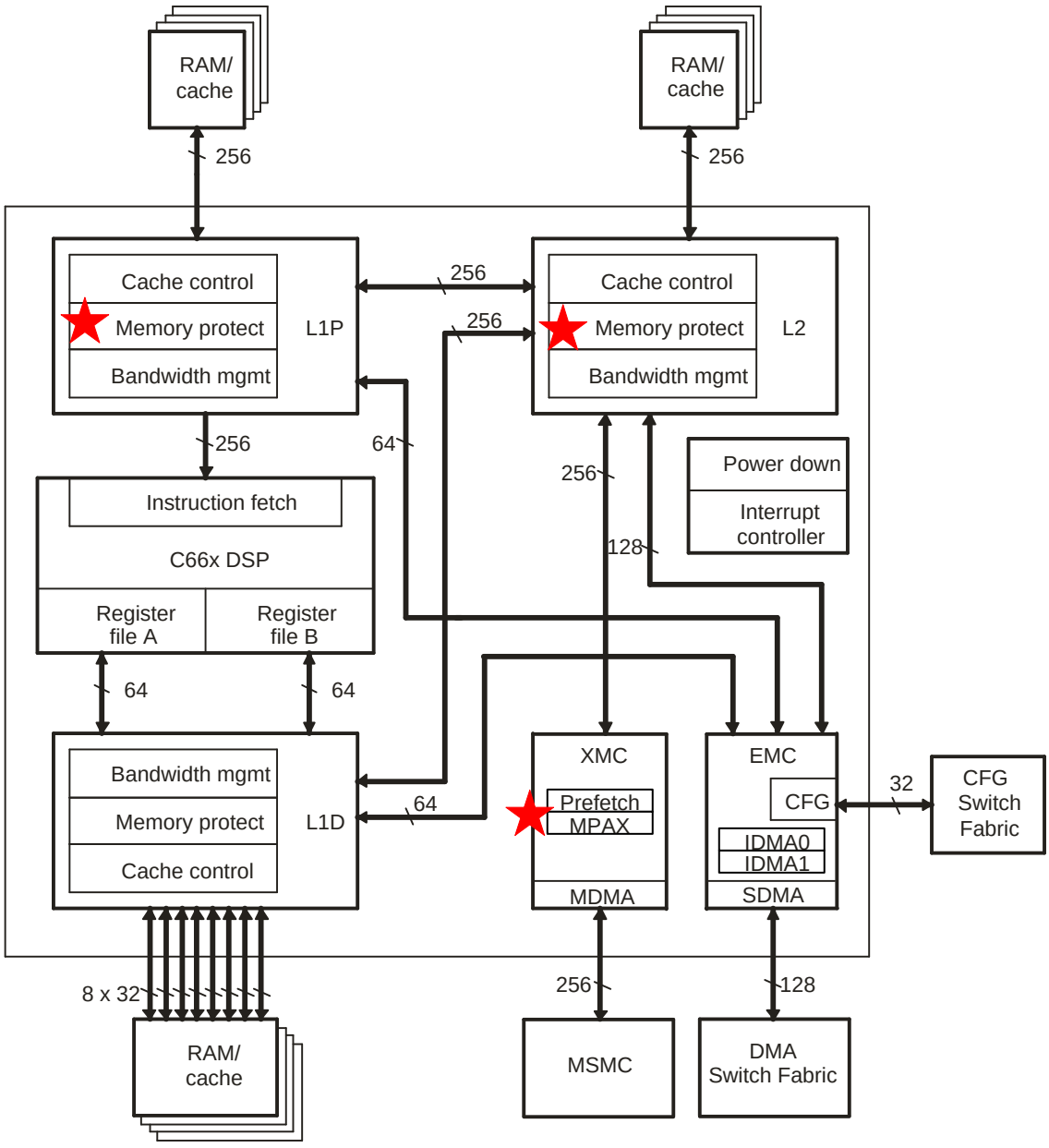

Figure 1. C66x CorePac Block Diagram

2 Memory Protection Register Map

Table 1. Memory Protection Register Map

L2 Memory Protection Registers		
Address	**Name**	**Description**
0x184_A000	L2MPFAR	Level 2 Memory Protection Fault Address Register
0x184_A004	L2MPFSR	Level 2 Memory Protection Fault Set Register
0x184_A008	L2MPFCLR	Level 2 Memory Protection Fault Clear Register
0x184_A200 ~ 0x184_A27C	L2MPPA0 ~ L1PMPPA31	Level 2 Memory Page Protection Attribute Registers
L1P Memory Protection Registers		
0x184_A400	L1PMPFAR	Level 1 Memory Protection Fault Address Register
0x184_A404	L1PMPFSR	Level 1 Memory Protection Fault Set Register
0x184_A408	L1PMPFCLR	Level 1 Memory Protection Fault Clear Register
0x184_A640 ~ 0x184_A67C	L1PMPPA16 ~ L1PMPPA31	Level 1 Memory Page Protection Attribute Registers
L1D Memory Protection Registers		
0x184_AC00	L1DMPFAR	Level 1 Memory Protection Fault Address Register
0x184_AC04	L1DMPFSR	Level 1 Memory Protection Fault Set Register
0x184_AC08	L1DMPFCLR	Level 1 Memory Protection Fault Clear Register
0x184_AE40 ~ 0x184_AE7C	L1DMPPA16 ~ L1DMPPA31	Level 1 Memory Page Protection Attribute Registers
XMC Memory Protection Registers		
0x800_0200	XMPFAR	XMC Memory Protection Fault Address Register
0x800_0204	XMPFSR	XMC Memory Protection Fault Status Register
0x800_0208	XMPFCR	XMC Memory Protection Fault Clear Register
0x800_0000 ~ 0x800_007C	XMPAXL/H0 ~ XMPAXL/H15	XMC Memory Protection and Address eXtension Registers

3 XMC Memory Protection and Address eXtension (MPAX)

The XMC Memory Protection and Address eXtension (MPAX) unit combines memory protection and address extension into one unified process. The memory protection step determines what types of accesses are permitted on various address ranges within C66x CorePac's 32-bit address map. The address extension step projects those accesses onto a larger 36-bit address space.

3.1 XMC MPAX Segment Registers

The MPAX unit defines 16 segments with selectable size and each segment has a corresponding set of permissions to control accesses to that segment. Figure 2 and Figure 3 show the layout of one segment register.

Figure 2. XMPAXH Register Layout

31	12	11	5	4	0
BADDR		Reserved		SEGSZ	

Figure 3. XMPAXL Register Layout

31	8	7	0
BADDR		PERM	

Table 2. XMPAXH/XMPAXL Register Field Descriptions

Field	Name	Description
BADDR	Base Address	Upper bits of address range to match in C66x CorePac's native 32-bit address space
SEGSZ	Segment Size	Segment Size. Table 3 indicates encoding.
RADDR	Replacement Address	Bits that replace and extend the upper address bits matched by BADDR
PERM	Permissions	Access types allowed in this address range

Table 3. XMPAXH.SEGSZ Segment Size Encoding

SEGSZ	Meaning	SEGSZ	Meaning	SEGSZ	Meaning	SEGSZ	Meaning
00000b	Seg disabled	01000b	Rsvd (Disabled)	10000b	128KB	11000b	32MB
000001b	Rsvd (Disabled)	01001b	Rsvd (Disabled)	10001b	256KB	11001b	64MB
000010b	Rsvd (Disabled)	01010b	Rsvd (Disabled)	10010b	512KB	11010b	128MB
00011b	Rsvd (Disabled)	01011b	4KB	10100b	1MB	11011b	256MB
00100b	Rsvd (Disabled)	01100b	8KB	01100b	2MB	11100b	512MB
00101b	Rsvd (Disabled)	01101b	16KB	01101b	4MB	11101b	1GB
00110b	Rsvd (Disabled)	0111bb	32KB	01110b	8MB	11110b	2GB
00111b	Rsvd (Disabled)	01111b	64KB	01111b	16MB	11111b	4GB

The PERM field is divided into various single-bit subfields. Figure 4 and Table 4 ‘‘Summary of Permission Bits in XMPAXL.PERM’’ summarizes the meaning of each bit.

Figure 4. XMPAXL.PERM Subfield Layout

7	6	5	4	3	2	1	0
Reserved	Reserved	SR	SW	SX	UR	UW	UX

Table 4. Summary of Permission Bits in XMPAXL.PERM

Bit	Meaning When Set	Bit	Meaning When Set
SR	Supervisor mode may read from segment	UR	User mode may read from segment
SW	Supervisor mode may write to segment	UW	User mode may write to segment
SX	Supervisor mode may execute from segment	UX	User mode may execute from segment

3.2 XMC MPAX Register Reset Defaults

By default memory segments 2 through 15 are reset to all 0s. XMC configures MPAX segment 0 and 1 so that DSP CPU can access system memory immediately after reset. The default configuration divides the address space into two 2GB segments with full access and with the lower bits of XMPAXL.RADDR equal to the value of XMPAXH.BADDR. The net result is to pass all accesses through unmodified. Figure 5 shows the memory map reset status.

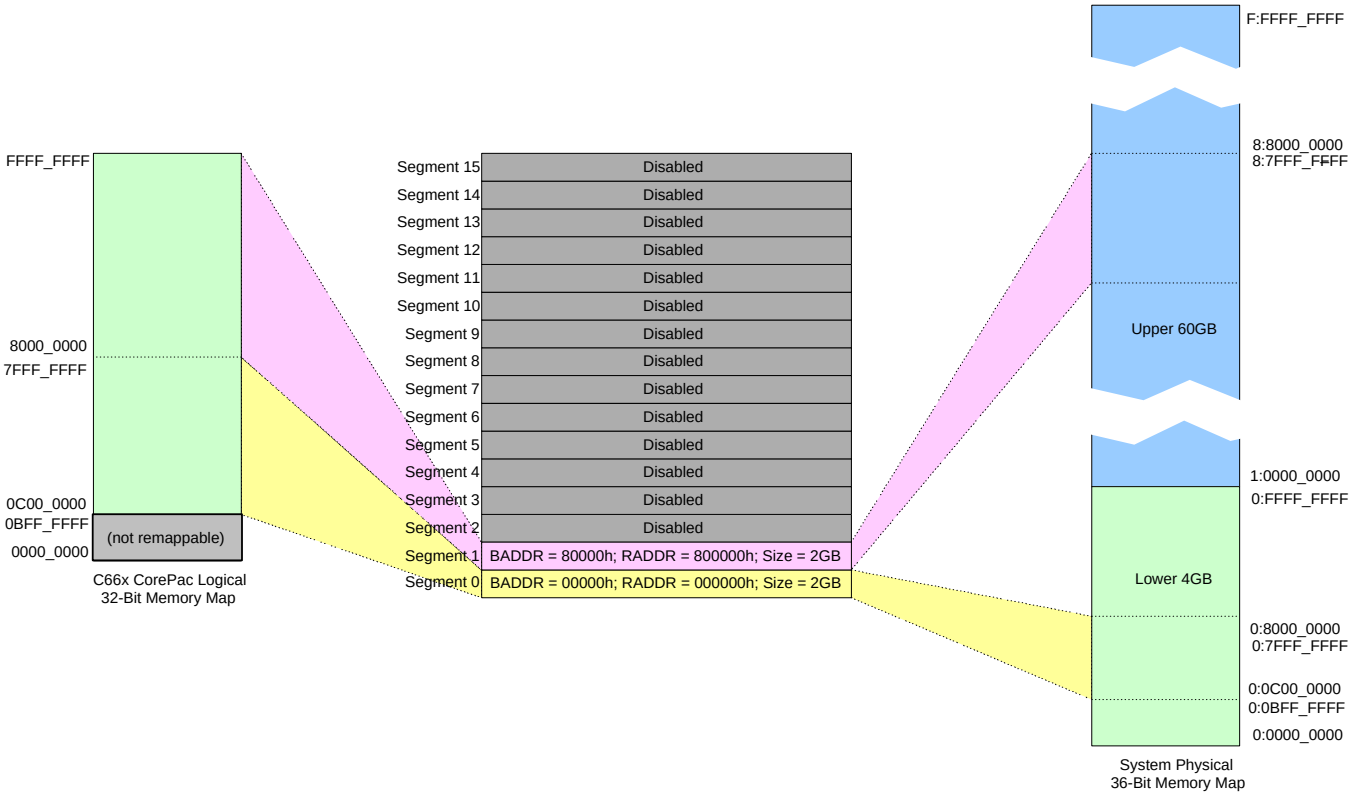

Figure 5. XMPAX Register Memory Map Reset Status

3.3 Memory Protection and Address Extension Process

The MPAX process consists of three main steps: Address range lookup, memory protection check, and address extension.

3.3.1 MPAX Address Range Lookup

XMPAXH.BADDR and XMPAXH.SEGZ fields describe where each MPAX segment resides in C66x CorePac's 32-bit logical address space. XMPAXH.SEGSZ field indicates the size of the segment, from 4KB to 4GB. XMPAXH.BADDR field indicates the start address of that segment in the logical address space.

Segments are always a power-of-2 size and start on a corresponding power-of-2 boundary. Thus, a 4KB segment always starts on a 4K boundary, and a 4GB segment corresponds to the entire 32-bit logical address space.

A logical address resides within a given MPAX segment if its upper address bits match the corresponding bits in the BADDR field. The number of bits compared is a function of the SEGSZ. Examples: For 4KB segments, all 20 bits of the BADDR field must match the upper 20 bits of the logical address. For 16MB segments, the upper 8 bits of the BADDR field must match the upper 8 bits of the logical address; the remaining bits are ignored. For 4GB segments, all logical addresses match regardless of BADDR's value.

Some special cases are described at the followings:

- Matching Multiple Segments

  A given logical address may fall into more than one MPAX segment. Higher numbered segments take precedence over lower numbered segments. The XMC only consults the highest numbered segment among all matches, and ignores all other matches. This allows programs to define complex memory maps with a small number of segment descriptions by creatively overlaying segments.

  In Figure 6, segment 1 matches 8000_0000 through FFFF_FFFF, and segment 2 matches C000_7000 through C000_7FFF. Because segment 2 is higher priority than segment 1, its settings take priority, effectively carving a 4K hole in segment 1's 2GB address space. Furthermore, it maps this 4K space to 0:5004_2000 - 0:5004_2FFF, which overlaps the mapping established by segment 2. This physical address range is now accessible by two logical address ranges.

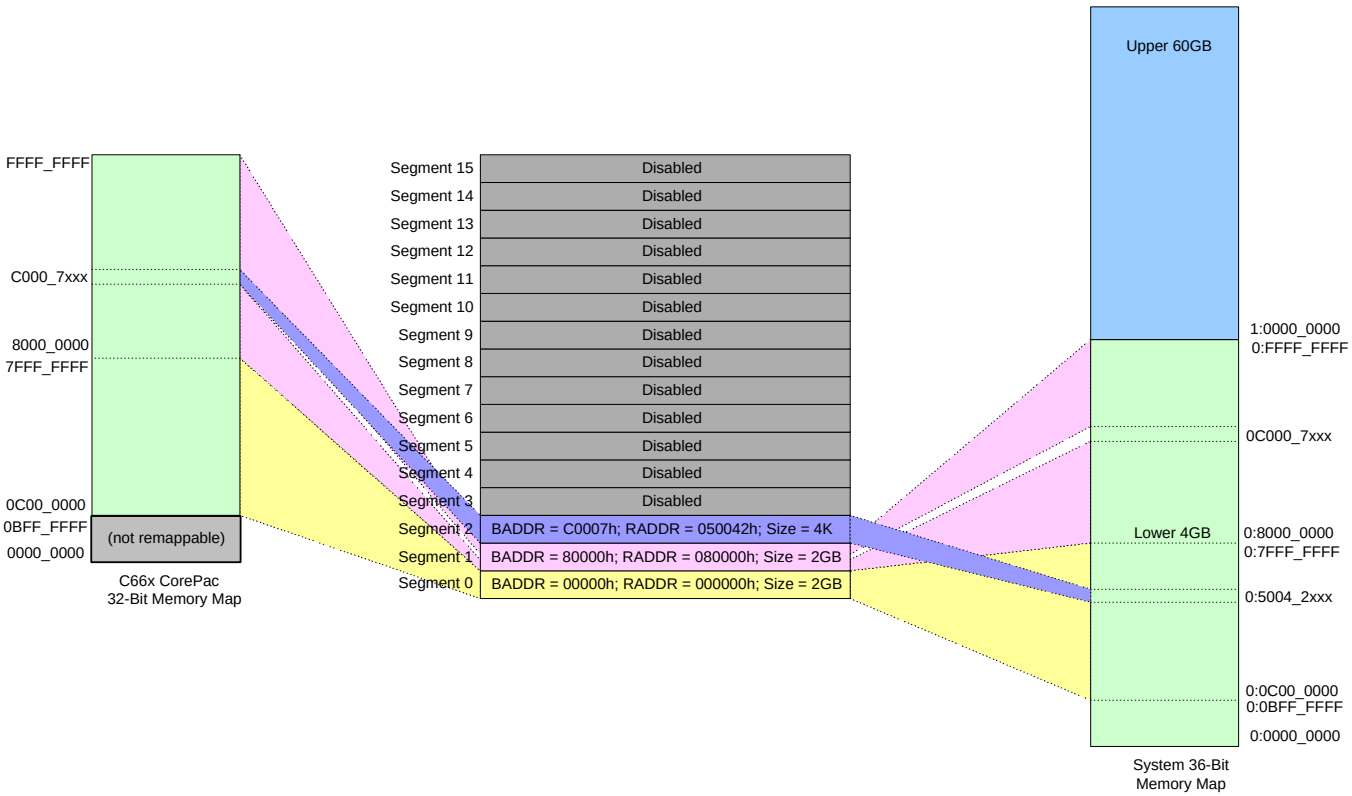

Figure 6. MPAX Segment Priority Example

- Matching No Segments

  It is possible that a given logical address matches no MPAX segments. MPAX treats these requests as matching a segment with zero permission as all accesses disallowed, and will generate a protection fault.

  To establish default permissions (and a default address extension) for the entire memory map, leave the default settings in MPAX segment 0 and 1 unchanged. Addresses that do not match other segments will “fall through” and match one of them.

- Address Ranges Unaffected by MPAX

  XMC considers accesses to 0000_0000 through 0BFF_FFFF as accesses to memory mapped control registers. These addresses never match any segment. The MPAX unit zero-extends these addresses to 36 bits and does not modify them further. XMC does not perform a segment-based protection check for accesses in this range, regardless of whether a segment overlaps this range.

  MPAX segment permissions do not apply to XMC’s memory mapped registers. XMC applies per-register permission checking independently of MPAX’s segment-based permissions.

NOTE: C66x CPU does not present all accesses to XMC. Accesses to addresses 0000_0000 to 07FF_FFFF are decoded internally to CPU and not sent to XMC.

3.3.2 MPAX Memory Protection Checks

Once the XMC determines MPAX segment for a given access, it can compare the access against the permissions associated with the segment. MPAX provides a mixture of immediate and deferred privilege checks depending on the memory access types. For immediate checks, it records faults in XMPFAR/XMPFSR and signals an exception. For deferred checks, MPAX merely returns the corresponding permission of the segment to L1/L2 cache controller and allows the requesting cache to perform the necessary check. In other words, XMC will defer the permission check to cache controller if the external memory access is in the cache region.

3.3.3 MPAX Address Extension

MPAX address extension works by replacing the upper address bits of the logical address (XMPAXH.BADDR) with corresponding bits from the replacement address (XMPAXL.RADDR). The replacement address field is wider than the field it replaces, thus extending the 32-bit logical address to a 36-bit physical address.

3.3.4 Memory Protection Fault Reporting Registers

To enable programs to diagnose a memory protection fault after an exception occurs, the XMC implements two registers (XMPFAR and XMPFSR) dedicated to storing information about the fault, and a third register (XMPFCR) to allow clearing the fault information. Figure 7, 8 and 9 show these registers.

Figure 7. Memory Protection Fault Address Register (XMPFAR)

31	0
Fault Address	
R +0	

(1) Legend: R = Read only; W = Write only; -*n* = Value after reset; -x = value is indeterminate. For more information, see the device-specific data manual.

Figure 8. Memory Protection Fault Status Register (XMPFSR)

31	9	8	7 6	5	0
Reserved		LOCAL	Rsvd	Access Type	
R +0		R +0	R +0	R +0	

(1) Legend: R = Read only; W = Write only; -*n* = Value after reset; -x = value is indeterminate. For more information, see the device-specific data manual.

Figure 9. Memory Protection Fault Clear Register (XMPFCR)

31	1	0
Fault Address		MPFCLR
R +0		

(1) Legend: R = Read only; W = Write only; *-n* = Value after reset; -x = value is indeterminate. For more information, see the device-specific data manual.

The XMPFAR and XMPFSR registers store only enough information for one fault. The hardware records the information about the first fault and generates an exception for that fault only.

The XMC holds the fault information until software clears it by writing to 1 to XMPFCR.MPFCLR. XMC does nothing if software writes 0 to XMPFCR.MPFCLR. The XMC ignores the value written to bits 31:1 of XMPFCR. Programs, however, should write 0s to these bits.

3.4 XMC Memory Protection Error Reporting – MDMAERREVT

XMC MPAX unit will report memory protection violations on accesses to the programmed segments (based on the access privilege settings in the XMPAX segment registers). If the incoming logical address does not match any of the segments (but is within the MDMA space), the XMC controller will consider this as an access with no permissions and report this back to the L2 controller, which will flag it as the MDMAERREVT event and report it in the MDMAERR register.

MDMAERREVT is event #110 in C66x DSP system as shown in Table 5. It can be routed to Exception combiner via EXPMASK3 register to trigger exception. In addition, it can also be enabled from EVTMASK3 register and mapped to one of DSP CPU interrupts in INTMUXn register.

NOTE: Certain memory protection errors will be triggered by L1/L2 cache controllers. The error events are L1P_CMPA/L1D_CMPA/L2_CMPA as shown in Table 5.

Table 5. C66x DSP System Event Map

EVT Number	Event	From	Description
0	EVT0	INT controller	Output of event combiner 0, for events 1 through 31
1	EVT1	INT controller	Output of event combiner 1, for events 32 through 63
2	EVT2	INT controller	Output of event combiner 2, for events 64 through 95
3	EVT3	INT controller	Output of event combiner 3, for events 96 through 127
4-8	Available events		
9	Reserved		
10	Available events		
11-12	Reserved		
13	IDMAINT0	EMC	IDMA channel 0 interrupt
14	IDMAINT1	EMC	IDMA channel 1 interrupt
15-95	Available events		
96	INTERR	INT controller	Dropped DSP interrupt event
97	EMC_IDMAERR	EMC	Invalid IDMA parameters
98	Reserved		
99	Available events		
100-101	Reserved		
102-109	Available events		
110	MDMAERREVT	L2	MDMA bus error event
111	Reserved		
112	Available events		
113	L1P_ED	L1P	Single bit error detected during DMA read
114-115	Available events		
116	L2_ED1	L2	Corrected bit error detected

Table 5. C66x DSP System Event Map (continued)

EVT Number	Event	From	Description
117	L2_ED2	L2	Uncorrected bit error detected
118	PDC_INT	PDC	PDC sleep interrupt
119	SYS_CMPA	SYS	DSP memory protection fault
120	L1P_CMPA	L1P	DSP memory protection fault
121	L1P_DMPA	L1P	DMA memory protection fault
122	L1D_CMPA	L1D	DSP memory protection fault
123	L1D_DMPA	L1D	DMA memory protection fault
124	L2_CMPA	L2	DSP memory protection fault
125	L2_DMPA	L2	DMA memory protection fault
126	EMC_CMPA	EMC	DSP memory protection fault
127	EMC_BUSERR	EMC	CFG bus error event

The MDMA bus error register (MDMAERR) is shown in Figure 10 and described in Table 5. When XMC memory protection error is triggered, MDMAERR.STAT field will be set to 2.

Figure 10. MDMA Bus Error Register (MDMAERR)

31	29	28	16
ERR		Reserved	
R-0			

15	12	11	8	7	3	2	0
Reserved		XID		Reserved		STAT	
R-0		R-0		R-0		R-0	

(1) Legend: R = Read only; W = Write only; *-n* = Value after reset; -x = value is indeterminate. For more information, see the device-specific data manual.

Table 6. MDMA Bus Error Register (MDMAERR) Field Descriptions

Bit	Field	Description
31-29	ERR	Error detected 0h = No error 1h = MDMA read status error detected 2h = MDMA write status error detected 3h-7h = Reserved
28-12	Reserved	Reserved
11-8	XID	Transaction ID Stores the transaction ID (RID or WID) when a read or write error is detected.
7-3	Reserved	Reserved
2-0	STAT	Transaction status 0h = Success (should not cause error to be latched), or unrecognized RID/WID (should cause error to be latched) 1h = Addressing error 2h = Privilege error 3h = Timeout error 4h = Data error 5h-6h = Reserved 7h = Exclusive - operation failure

The exception handler or ISR servicing MDMAERREVT should clear the MDMAERR event in the MDMA Bus Error Clear Register (MDMAERRCLR). Write 1 to MDMAERRCLR.CLR filed to clear the error status.

Figure 11. MDMA Bus Error Clear Register (MDMAERRCLR)

31	1	0
Reserved		CLR
R +0		W-0

(1) Legend: R = Read only; W = Write only; *-n* = Value after reset; -x = value is indeterminate. For more information, see the device-specific data manual.

4 XMC Memory Protection With Cache Regions

The C66x cache hierarchy works in terms of 32-bit logical addresses for its tags. It also caches copies of returned permissions from XMC when accessing cacheable data. Both of these aspects can lead to cache coherence issues when changing MPAX segment registers on the fly.

4.1 General Updates to MPAX Segments

To prevent unwanted operation, programs must take proactive steps to prevent the cache and the MPAX segment registers from being out of sync.

Programs should write back and invalidate address ranges before changing segment mappings that affect those address ranges. The recommended procedure is as follows:

- Write back and invalidate affected address range(s). Use L2WIBAR/L2WIWC on each range, or L2WBINV to force out the contents of the entire cache.
- Perform a fence operation to ensure all write-backs have completed.
- Use XPFCMD to invalidate the prefetch buffer if prefetching is enabled for addresses in the affected region.
- Update the MPAX segment registers as desired.

This sequence ensures that C66x CPU will see the correct permissions and fetch data from (and write data back to) the correct physical addresses.

As this process is very costly, most programs will avoid changing MPAX segment registers often. The cost of the operation is roughly proportional to the size of range affected, and the number of victim writebacks the cache needs to perform in that address range.

4.2 Changing MPAX Settings on Current Program Segment

Programs may need to configure XMPAX segments that cover the range of addresses that the program is currently executing from. The following steps should be followed when programming XMPAX registers under this scenario.

1. Program XMPAXL register with the new PERM field permitting the C66x CPU to continue executing from current program segment and the new RADDR field points the current program segment.
2. Program XMPAXH register with the new BADDR and SEGSZ.

CAUTION

Fail to follow the proper sequence can lead to unwanted XMC error being triggered due to instruction being fetched without the read permission.

4.3 XMC Memory Protection Error Reporting With Cache Regions

XMC does not always report the memory protection errors from the violations in cache regions. This is because XMC defers the permission check to L2 cache and the permission configured for the region in XMC is cached in L2 cache when the cache line is allocated. If L2 cache controller identifies any violation, it will record the fault address and access type in L2MPFAR and L2MPFSR. In addition, L2 cache controller will trigger L2_CMPA error event.

4.4 False XMC Memory Protection Errors With Cache Regions

L2 Cache Writebacks can trigger false XMC errors under the conditions as the followings:

- L2 Cache Block Writeback with Supervisor Write Permission Disabled
- Debugger Access with Supervisor and User Write Permission Disabled

4.4.1 False XMC Memory Protection Error From L2 Cache Block Writeback

In C66x L2 Cache Controller design, L2 Cache controller performs “Writeback Flush with Non-Posted Write” at the end of Cache Block Writeback operation. This feature ensures all victims for the block coherency operation have been flushed to the OCP interconnect. Even if there is no dirty line, this special “Non-Posted Write” will still be issued.

This Non-Posted Write from L2 Cache controller is issued as Supervisor Write with data size 0 from the last cache line of the block operation.

If a memory segment is configured with SW=0 (Supervisor Write Disabled) and Cacheable, one specific XMC error can occur from performing Cache Block Writeback as Supervisor Write violation. This error is the result of the Non-Posted Write as Supervisor issued from L2 Cache Controller.

This specific XMC error can be triggered from Cache Block Writeback even when the cache is empty as long as the above condition is met.

Global Cache Writeback DOES NOT issue a Non-Posted Write at the end of operation.

4.4.2 Software Workaround for L2 Cache Block Writeback False XMC Errors

- **Option 1:** Add a wrapper API to Cache Block WriteBack call. In the wrapper, perform the following steps.
  - Disable HW interrupt.
  - Add an MPAX(N+1) entry which is higher numbered than existing entries and allows Supervisor Write for the 4KB-segment covering the last cache line (base address + size) for the nopost dummy write.

    NOTE: XMC only consults the highest numbered segment among all matches, and ignores all other matches.

  - Perform cache block writeback.
  - Remove the MPAX(N+1) entry.
  - Enable HW interrupt.
  - Pro: No change to existing memory protection policy. Minor performance impact.
  - Con: Code change required to use Wrapper API for Cache Block Writeback.
- **Option 2:** Only perform Global Cache Writeback.
  - Pro: Easy to implement. No change to existing memory protection policy.
  - Con: Potential performance impact depending on how frequently cache maintenance is performed, and code change required to replace Cache Block Writeback with Global Writeback.
- **Option 3:** For memory write, always perform in Supervisor mode and enable Supervisor Write permission.
  - Pro: No change to cache routine. No performance impact.
  - Con: Re-design memory protection policy to allow Supervisor Write to Protected Data region.

4.4.3 False XMC Memory Protection Error From Debugger Access

Debugger access is allowed regardless XMC permission setting in XMPAXL.PERM field. Debugger Write updates the value in External Memory as well as Cache if enabled.

If Debugger writes to the memory location, which is also cached, the cache line will be marked dirty. Performing Cache WB will forward the WB request to XMC. If both Supervisor and User mode Write permission is disabled, XMC will record the Cache Writeback error in XMPFAR (Fault Address) and XMPFSR with a specific value (=0x112).

The examples of Debugger usage that can result in XMC errors are shown as the followings:

- Software Breakpoint (SWBP) set in Program memory without Supervisor or User Write permission (R/X only).
- CIO (Printf) function resides in Program memory without Supervisor or User Write permission (R/X only).
- Modify value in Code Composer Studio™ (CCS) Memory window at the memory segment without Supervisor or User Write permission (R/X only).

4.4.4 Workaround for Debugger Access From SWBPs

To work around SWBP issue, Hardware Breakpoint (HWBP) should be used in CCS when XMC memory protection is enabled without write permission. For the option in CCS to disable SWBP usage, see Figure 12. HWBP will automatically be chosen when setting breakpoints.

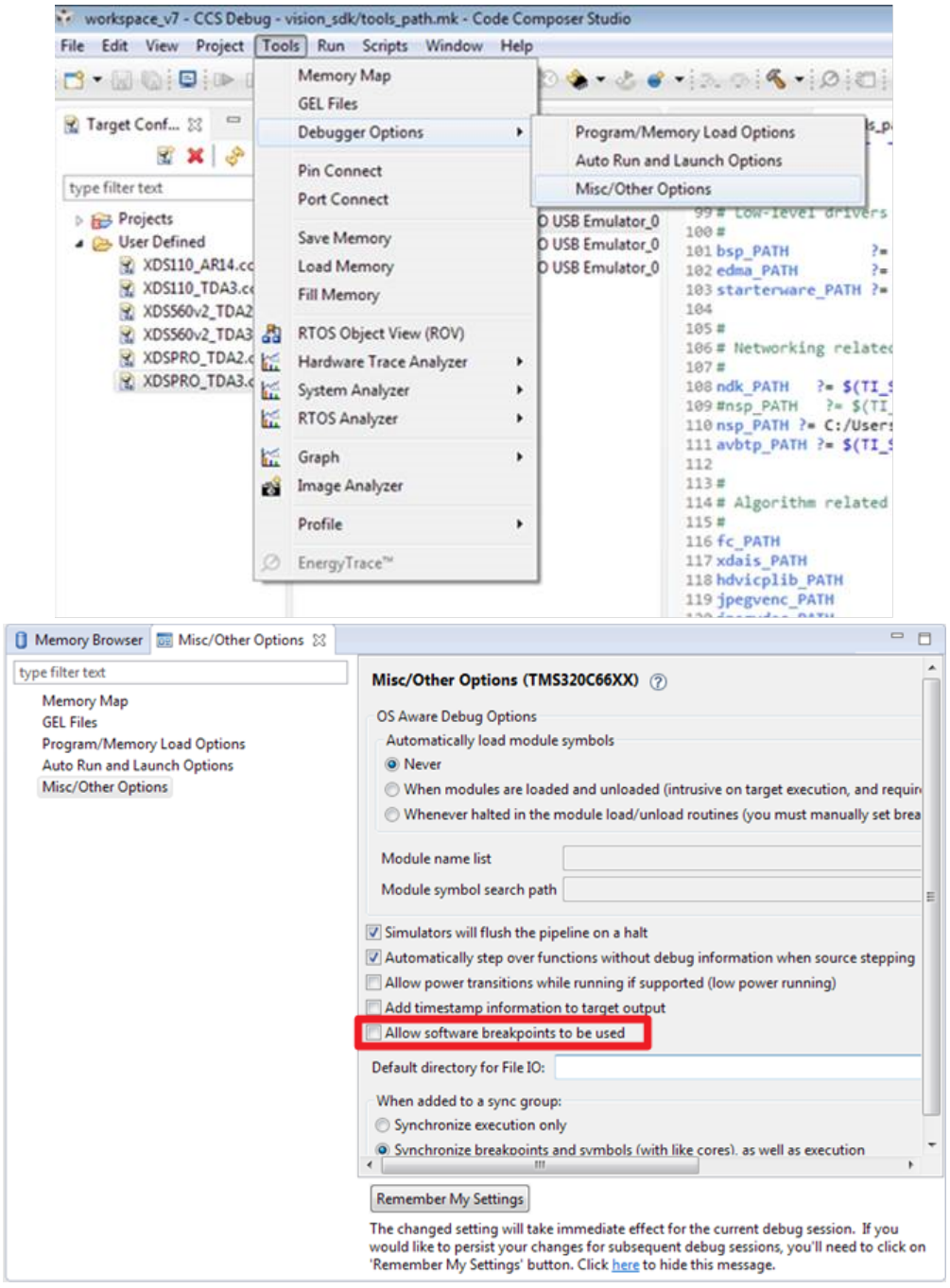

Figure 12. Configure Hardware Breakpoint Option in CCS

5 References

- *TMS320C66x DSP CorePac User's Guide*
- *TDA2x SoC for Advanced Driver Assistance Systems (ADAS) Silicon Revision 2.0, 1.x Technical Reference Manual*

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265